CARRERA DE ANÁLISIS DE SISTEMAS

DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA EL CONTROL DE COMISIONES PARA BROKER DE SEGUROS EN LA EMPRESA QUALITYSEG S.A. UBICADA EN GUAYAQUIL, ABRIL – SEPTIEMBRE 2018

Trabajo de Titulación previo a la obtención del título de Tecnólogo en Análisis de Sistemas

AUTOR: López Viteri Christian Javier

DIRECTOR: Ing. Basantes Basantes Jaime Neptalí

Quito, 2018

TECNOLÓGICO SUPERIOR CORDILLERA

FORMULARIO 005 Unidad de Titulación — ACTA DE APROBACIÓN FINAL DE PROYECTOS

ACTA DE APROBACIÓN DEL PROYECTO DE GRADO

Quito, 18 de Diciembre de 2018.

El equipo asesor del Trabajo de Titulación del Sr. (Srta.) (Sra.) **LOPEZ VITERI CHRISTIAN JAVIER** de la Carrera de Análisis de Sistemas cuyo tema de investigación fue: **"DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB DE CONTROL DE COMISIONES PARA BROKER DE SEGUROS EN LA EMPRESA QUALITYSEG S.A. UBICADA EN GUAYAQUIL, ABRIL - SEPTIEMBRE 2018."** una vez considerados los objetivos del estudio, coherencia entre los temas y metodologías desarrolladas; adecuación de la redacción, sintaxis, ortografía y puntuación con las normas vigentes sobre la presentación del escrito, resuelve: ***APROBAR*** el proyecto de grado, certificando que cumple con todos los requisitos exigidos por la Institución.

BASANTES BASANTES JAIME NEPTALÍ
Tutor del Proyecto

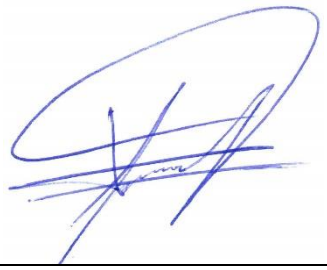
GARZON JACOME ELSA PATRICIA
Lector del Proyecto

INSTITUTO TECNOLÓGICO SUPERIOR "CORDILLERA"
DIRECCIÓN DE CARRERA
HEREDIA MAYORGA HUGO PATRICIO.
Director de Carrera
Análisis de Sistemas

INSTITUTO TECNOLÓGICO SUPERIOR "CORDILLERA"
CONSEJO DE CARRERA
CORONEL ORDOÑEZ JOHMMY PATRICIO
Delegado Unidad de Titulación
Análisis de Sistemas

DECLARACIÓN DE AUTORÍA

Yo, Christian Javier López Viteri, declaro bajo juramento que la investigación es absolutamente original, auténtica, es de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes. Las ideas, doctrinas, resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad



Christian Javier López Viteri

C.C: 1722630801

LICENCIA DE USO NO COMERCIAL

Yo, Christian Javier López Viteri portador de la cédula de ciudadanía signada con el No. 1722630801 de conformidad con lo establecido en el Artículo 110 del Código de Economía Social de los Conocimientos, la Creatividad y la Innovación (INGENIOS) que dice: "En el caso de las obras creadas en centros educativos, universidades, escuelas politécnicas, institutos superiores técnicos, tecnológicos, pedagógicos, de artes y los conservatorios superiores, e institutos públicos de investigación como resultado de su actividad académica o de investigación tales como trabajos de titulación, proyectos de investigación o innovación, artículos académicos, u otros análogos, sin perjuicio de que pueda existir relación de dependencia, la titularidad de los derechos patrimoniales corresponderá a los autores. Sin embargo, el establecimiento tendrá una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra con fines académicos. Sin perjuicio de los derechos reconocidos en el párrafo precedente, el establecimiento podrá realizar un uso comercial de la obra previa autorización a los titulares y notificación a los autores en caso de que se traten de distintas personas. En cuyo caso corresponderá a los autores un porcentaje no inferior al cuarenta por ciento de los beneficios económicos resultantes de esta explotación. El mismo beneficio se aplicará a los autores que hayan transferido sus derechos a instituciones de educación superior o centros educativos.", otorgo licencia gratuita, intransferible y no exclusiva para el uso no comercial del proyecto denominado "DESARROLLO E IMPLEMENTACION DE UN SISTEMA WEB DE CONTROL DE COMISIONES PARA BROKER DE SEGUROS EN LA EMPRESA QUALITYSEG S.A UBICADA EN GUAYAQUIL, ABRIL - SEPTIEMBRE 2018" con fines académicos al Instituto Tecnológico Superior Cordillera.

Christian Javier López Viteri

C.C: 1722630801

Quito, 17 de diciembre de/2018

AGRADECIMIENTO

Yo, Christian Javier López Viteri, agradezco inmensamente el apoyo por parte de toda mi familia ya que en el proceso de mis estudios fueron a pesar de todos los contratiempos presentados siempre fueron un pilar importante para no dejar de seguir adelante.

Agradezco el apoyo por parte de los docentes de la institución que fueron el eje vital en cada ciclo que se fue avanzando, en especial agradezco la amistad que se ha podido formar con la gran mayoría de los docentes, ahora en mi proceso de titulación solo espero poder seguir contando con cada una de las personas.

Soy grato con la institución que a pesar de los inconvenientes que se presentan supieron acogerme como alumno y por mi parte creo que me he desempañado de manera adecuada siempre dejando en alto el nombre de la institución.

DEDICATORIA

Este proyecto es dedicado a mi familia, mi madre al ser el pilar de un hogar de cuatro hermanos siempre nos enseñó los mejores valores, honestidad, integridad y respeto.

Estos cuatro valores siempre fueron fomentados y con ejemplo me han permitido poder compartirlos con las personas que son allegadas a mi persona.

También va dedicado a mi esposa, ella junto a mi compartió malas noches para poder cumplir este sueño siempre apoyándome en equipo, cada uno siempre nos damos la mano cuando el uno está por rendirse siempre con palabras de amor y aliento ha conseguido levantarme cada día a pesar de los problemas que puedan presentarse.

A mi hija porque cada día me alegra con su sonrisa desde el día que la pude tener en mis brazos pude darme cuenta que el amor verdadero y eterno existe.

Mis hermanos que con sus consejos han podido brindarme sabiduría, su apoyo incondicional en todo ámbito me ha ayudado a ver que todo es posible con el apoyo de la familia.

Mi familia siempre se encuentra unida y por ese motivo me permito dedicar este proyecto que no fue nada fácil lograrlo.

CONTENIDO GENERAL

INDICE DE TABLAS

INDICE DE FIGURAS

INDICE DE ANEXOS

RESUMEN EJECUTIVO

La investigación del proyecto se realizó con el fin de poder brindar una herramienta que permita a la empresa Qualityseg S.A. reducir tiempos en los procesos de operación.

Brindando grandes beneficios a los distintos colaboradores de las distintas áreas de la empresa, como son Operaciones, Comercial, Jefaturas y Gerencias.

Se ha realizado el cumplimiento de cada objetivo trazado desde el inicio del proyecto, cada área planteó sus objetivos los cuales fueron estructurados y analizados para poder brindar una solución a cada uno de ellos.

Con el desarrollo del sistema se logra centralizar la información de la empresa referente a las emisiones de sus pólizas de vehículos y de ramos generales.

Con dicha información se ha podido realizar varios reportes gerenciales los cuales permiten al encargado de cada área poder ver el rendimiento de sus operadores, estos son reportes de producción emitida, reporte de producción pendiente, cotizaciones pendientes, cuotas impagas del cliente, facturas pendientes, comisiones pendientes y comisiones pagadas.

ABSTRACT

The investigation of the project I realize in order to be able to offer a tool that allows the company Qualityseg S.A. to reduce times in the processes of operation.

Offering big benefits to the different collaborators of the different areas of the company, since they are Operations, Commercial, Headquarters and Managements.

There has been realized the fulfillment of every aim planned from the beginning of the project, every area I raise his aims which were structured and analyzed to be able to offer a solution to each of them.

With the development of the system it is achieved to centralize the information of the company relating to the emission of his policies of vehicles and of general branches.

With the above-mentioned information, one could have realized several managerial reports which allow to the manager of every area to be able to see the performance of his operators, these are reports of issued production, report of hanging production, hanging prices, unpaid quotas of the client, hanging invoices, hanging commissions and full commissions.

CAPÍTULO I

1. Antecedentes

1.01 Contexto

Qualityseg S.A. Agencia Asesora Productora de Seguros, se dedica a la gestión de contratos de seguros, su oficina matriz se encuentra ubicada en Quito en el edificio Metropolitan localizado sobre la Av. Naciones Unidas entre Núñez de Vela e Iñaquito, dispone de varios puntos de servicio en Quito, Guayaquil, Ambato, Riobamba y Santo Domingo.

Qualityseg fue constituida en el 2011 donde da inició sus actividades, tiene como misión brindar a sus clientes individuales y corporativos asesoramiento especializado en todas las ramas de seguros. Cuenta con el respaldo de las mejores compañías de seguros tales como AIG Metropolitana, Latina Seguros, Latina Salud, Seguros del Pichincha, Generaly, QBE, Seguros Equinoccial, Humana, Chubb Seguros, Seguros Sucre, BMI del Ecuador, Equivida, entre otras.

Qualityseg es miembro del Grupo Mavesa, por lo que su principal rama de negocios es el de seguros vehiculares, adicional maneja seguros de personas, seguros generales y fianzas.

Actualmente se mantiene varios inconvenientes con la forma que se paga comisiones al personal, no se lleva un control bien estructurado y varios procesos se los realiza manualmente, dando como consecuencia pagos retrasados, incongruencia con los pagos de la compañía de seguros y un gran desorden en la información. Por este motivo se desea una herramienta que permita administrar los procesos de emisión, registro de pagos de los clientes, facturación a la compañía de seguros,

cobro de comisiones a la compañía de seguros y liquidación de comisiones a personal de la empresa.

En el desarrollo de la investigación se espera poder mejorar el flujo de procesos y poder mejorar el control de la información que se tiene disponible, brindando un fácil acceso a la información de l cliente y que dicha información se confiable.

1.02 Justificación

El principal problema que tiene la empresa es el control de comisiones al momento de realizar los pagos a los asesores comerciales, mismos que se encargan de registrar sus ventas de seguros en un Excel por lo que no se puede establecer un control sobre las ventas realizadas, por este motivo la empresa decide contratar un sistema especializado en el manejo de corredores de seguros, el mismo cuenta con interfaces no amigables lo que les ocasiona inconvenientes en el trabajo y reduce el tiempo de procesos operacionales, considerando que teniendo ya el sistema más de 4 años no se ha llegado al objetivo que es poder realizar los pagos a los asesores desde el sistema.

Considerando estos antecedentes se propone el desarrollo de una aplicación la cual solucionara los inconvenientes presentados en el área comercial, área de cobranzas y el área de pago de comisiones.

La implementación de dicha aplicación permitirá reducir el tiempo en trabajo operativo, reducción de costos en gasto de papel impreso, se mejoraría costos ya que contarían con una herramienta propia y realizada a sus necesidades, la información tendría alta disponibilidad, es decir en caso de un cliente puede realizar consultas en línea, estas mejoras presentarán un impacto tanto social, económico y ambiental que mejorarán a la empresa.

1.03 Definición del problema central

Tabla 1.

Matriz de Fuerzas

Situación Empeorada	**Situación Actual**				**Situación Mejorada**
Pérdida de ingresos por deficit en el cobro de comisiones a compañía de seguros	Gestión manual en registro, cobro y pago de comisiones en el corredor de seguros				Sistematización de procesos en liquidación de comisiones.
Fuerzas Impulsadoras	**I**	**PC**	**I**	**PC**	**Fuerzas Bloqueadoras**
Realizar un control manual y revisado por jefe de operaciones	4	5	3	5	Empleados presentan un déficit de cultura informática para el manejo de un sistema informático.
Empezar a manejar con la producción emitida en el sistema	3	5	5	5	Insuficiente apoyo para sistematizar procesos por parte de los directores ejecutivos de la empresa.
Cambio de política de comisiones en base a lo emitido al sistema	4	3	4	4	Déficit de recursos económicos para implementación de sistemas informáticos.

Nota: Matriz de fuerzas T permite determinar las fuerzas bloqueadoras e impulsadoras del proyecto.

Análisis de Fuerzas Impulsadoras

FI 1: Realizar un control manual y revisado por jefe de operaciones.

I = 4: Fuerza impulsadora tiene una intensidad con escala de valor cuatro por motivo de que la empresa actualmente lleva cerca del 80% de registros de manera manual.

PC = 5: La fuerza impulsadora tiene un potencial de cambio con escala de valor de cinco ya que su principal objetivo es dejar los archivos manuales y automatizar procesos.

FI 2: Empezar a manejar con la producción emitida en el sistema.

I = 3: Fuerza Impulsadora tiene una intensidad con escala de valor de tres, debido a que el sistema actual no cumple con las expectativas del usuario.

PC = 5: Fuerza Impulsadora tiene un potencial de cambio con escala de valor de cinco, ya que la idea de implementar un sistema acorde a sus necesidades es vital para centralizar la información.

FI 3: Cambio de política de comisiones en base a lo emitido al sistema.

I = 4: Fuerza impulsadora tiene una intensidad con escala de valor cuatro, debido a que con el cambio que se presenta en políticas no se puede evidenciar realmente que se cobra y que se paga a los asesores comerciales.

PC = 3: Fuerza impulsadora tiene un potencial de cambio de tres debido a que al implementar un sistema con el cual se puede manejar la política, las comisiones pueden variar debido a que existe un mejor control de las mismas.

Análisis de Fuerzas Bloqueadoras

FB 1: Empleados presentan un déficit de cultura informática para el manejo de un sistema informático.

I = 3: Fuerza bloqueadora tiene una intensidad con escala de tres, debido a que la mayoría del personal en la empresa tiene muy poca cultura informática, dando como consecuencia el temor al cambio en los procesos que ejecutan en la actualidad.

PC = 5: Fuerza bloqueadora tiene un potencial de cambio con escala de cinco, el personal al ver que carecen de cultura informática toma la decisión de acoplarse a los cambios, solicitando capacitaciones sobre los procesos que se implemente.

FB 2: Insuficiente apoyo por parte de los directores ejecutivos de la empresa.

I = 5: Fuerza bloqueadora con una intensidad en escala de cinco, debido al insuficiente apoyo en la toma de decisiónes por parte de los directores de la empresa, no se ha podido avanzar con la automatización de procesos.

PC = 5: Fuerza bloqueadora con un potencial de cambio en escala de cinco, debido a que al presentar una propuesta accesible para poder sistematizar los procesos y demostrando que su productividad aumentaría, los directivos toman la decisión de apoyar el proyecto.

FB 3: Déficit de recursos económicos para implementación de sistemas informáticos.

I = 4: Fuerza bloqueadora con una intensidad en escala de cuatro, debido a que los sistemas que actual mente permiten automatizar procesos del bróker son demasiado costosos y no se acoplan a las necesidades al 100%.

PC = 4: Fuerza bloqueadora con un potencial de cambio de cuatro, debido a que el desarrollo del proyecto presenta factibilidad económica para la empresa, restando gastos a la empresa y optimizando recursos.

CAPÍTULO II

2. Análisis de involucrados

2.01.01 Situación Actual.

Actualmente la empresa sigue un flujo de procesos el cual permite realizar el cobro de comisiones, los cuales se detalla a continuación.

Cotización

Un cliente al solicitar una cotización sobre su seguro solicita varias alternativas, dando como resultado un cuadro el cual permite realizar comparaciones de los productos que la compañía se seguros nos ofrece, mismos que se presentan en sus cotizadores.

Emisión

Una vez que el cliente toma la decisión de un seguro en base a la cotización presentada, el asesor comercial realiza la solicitud de emisión a la compañía de seguros seleccionada por el cliente y con las condiciones que se establece en el producto seleccionado, en la presente solicitud el asesor adjunta documentación importante que la compañía de seguros con el formulario de vinculación.

La compañía de seguros recepta la solicitud de emisión y genera la póliza, enviando al asesor comercial los contratos del seguro tres copias, la primera se denomina original que tiene como destino el cliente, la segunda es un duplicado firmado por el cliente la misma debe ser devuelta a la aseguradora y la tercera se denomina agente la cual es un duplicado para el corredor de seguros en este caso Qualityseg.

Cobranzas

Una vez emitida la póliza el asesor comercial solicita al cliente que realice el pago de su primera cuota o el pago total según los acuerdos establecidos en la cotización, una vez que se realiza el pago el asesor comercial está encargado de re direccionar el comprobante o los cheques al área de cobranzas para que pueda registrar dicho pago y notificar de igual maneada a la compañía de seguros.

En caso de que el cliente no realice ningún pago de su póliza, la emisión de la misma será anulada y perderá cobertura de su seguro.

Adicional a esto el ejecutivo de cobranzas tiene como labor realizar la gestión de cobros de las cuotas restantes y gestionar la cartera vencida de los clientes.

De igual manera en caso de que el cliente deje de pagar su póliza, el ejecutivo de cobranzas solicita la cancelación de la póliza, dando como resultado la invalidez del contrato de seguros.

Pre liquidación de comisiones

El Jefe Operativo solicita el detalle de comisiones a liberar en el mes a las diferentes compañías de seguros con las cuales se mantiene un convenio, dicho detalle permite al Jefe de Operaciones validar lo que la compañía de seguros se le va a facturar.

Facturación

Con el detalle de comisiones se proceder a realizar una pre - factura la misma nos presenta un detalle general de lo que se desea facturar, el mismo nos da un valor total con el cual se procede a generar la factura electrónica.

Esta factura se la genera por medio del sistema JD el mismo es el sistema de facturación de Mavesa, de la factura generada se registra de manera manual en un archivo Excel el valor y el número de factura.

Cobro de comisiones

Una vez que la compañía de seguros recibe la factura generada procede a liberar el pago el mismo que se lo registra en el mismo archivo que se almaceno la información con los datos de la factura.

Liquidación de comisiones

Al recibir el pago de la factura por parte de la compañía de seguros se procede a revisar uno a uno el detalle con el que se generó dicha factura con el reporte de producción emitida en el sistema EFI.

Una vez realizada la revisión se procede a enviar el detalle de liquidación de comisiones de cada asesor en base a las pólizas que vendieron se procede a enviar un detalle de cada asesor por medio de correo electrónico al área de talento humano para que se pueda liberar la comisión a los asesores comerciales.

2.01.02 Visión y alcance.

Visión

El proyecto a desarrollarse tiene como finalidad reestructurar los procesos que se aplican al momento de realizar el registro, cobro y pago de comisiones de un corredor de seguros, optimizando tiempos en cada proceso.

Alcance.

La finalidad del sistema es brindar a Qualityseg una herramienta informática que permita facilitar el proceso de emisión, cobranzas, facturación y pago de comisiones.

Módulo de seguridad

El presente módulo permite al colaborador un acceso al sistema solicitando un usuario y clave que será entregado oportunamente, esto redireccionará a una pantalla de inicio donde se presenta los accesos de acuerdo al perfil que se encuentre asignado.

Módulo de Mantenimiento o Ajustes

El módulo de mantenimiento permite realizar la administración de todos los parámetros que requiera el sistema para su funcionamiento con la opción de registrar, modificar y eliminar.

Si no se administra este módulo el manejo del sistema presentará contratiempos por insuficiente información en los procesos que se presenta de acuerdo al manejo del usuario.

Módulo producción

En el módulo de producción se dividirá en tres grupos los cuales permitirán manejar de manera ordenada el flujo del proceso.

Solicitud de Emisión

En el proceso de emisión se podrá realizar las solicitudes de emisión de pólizas directamente a la compañía de seguros manejando notificaciones en base a las condiciones que el cliente solicita, este tipo de notificaciones se las realiza por medio de correo electrónico o por carta.

Emisión

Permite el registro de la póliza emitida por la compañía de seguros, detallando los números de contrato, cliente, factura, con eso se finaliza la emisión en el sistema.

Consulta de pólizas

Permite realizar la consulta de pólizas ingresadas al sistema, únicamente que se encuentren con número de factura y numero de contrato.

Dicha consulta permitirá evidenciar los datos del seguro registrado y sus características, forma de pago, cuotas, valores y vigencias.

Módulo de cobranzas

En el presente modulo se registrará todo lo referente a pagos realizados por el cliente, de igual manera se podrá realizar consultas de los mismos, para mejorar la ejecución se lo divide en tres grupos.

Registro de pagos individuales

Este módulo permitirá el registro de pagos de las cuotas que tiene un cliente de su póliza de seguros, determinando la fecha de pago, se podrá aplicar abonos a las cuotas y pagos totales, una vez aplicado el pago se enviará una notificación al cliente por mail.

Reversión de pagos

Permite realizar la restitución de los pagos aplicados, siempre y cuando la póliza no se encuentre facturada para el cobro de comisiones.

Consulta de cobranzas

Permite realizar la consulta de los pagos registrados por cliente y por número de cuotas, detallando el usuario que registro el pago.

Módulo para Cobro de comisiones

En el modo de cobro de comisiones se podrá registrar todo lo referente a cobros y pagos de comisiones para el corredor de seguros, dichos pagos son ejecutados por la compañía de seguros en base a las facturas emitidas.

Preliquidación

Módulo el cual permitirá realizar la comparación del detalle enviado por la compañía de seguros con la información ingresada en el sistema.

Al procesar la información se generará un registro de pre - factura el cual detalla los valores que se desea factura a la compañía de seguros.

Envió de detalle para factura

Se presentará una pantalla donde se podrá realizar la consulta de las prefecturas generadas mismo que permitirá seleccionar una o más de una para generar un total y será enviado por medio de correo electrónico al área de facturación.

Registro de factura

Módulo presentará una pantalla donde se debe registrar los datos de la factura generada poniendo un estado de pendiente de cobro.

El detalle de la factura generada se tomará en base al detalle que se registra en la pre - factura.

Registro de cobro de facturas

Pantalla la cual permite registrar si una factura que fue emitida por el corredor de seguros a la compañía de seguros fue pagada, adicional se presentará el detalle de la fecha de pago y el usuario quien registra en el sistema.

Consulta de facturación

Módulo que permite revisar las facturas emitidas a las distintas compañías de seguros que se encuentren en los estados pendientes de cobro y facturas pagadas.

De igual manera permitirá realizar consultas del detalle de cada factura para poder evidenciar que es lo que se ha facturado para cada compañía de seguros.

Módulo de liquidación de comisiones

El módulo de liquidación permitirá realizar el pago de comisiones a los asesores comerciales o subagentes, en el mismo se presentará una pantalla la cual se debe escoger el subagente y permitirá consultar las pólizas que fueron emitidas por el subagente seleccionado y marcar las que se realizara el pago, cabe mencionar que

no se presentará los registros que no hayan sido facturados y mucho menos cobrado la factura.

2.01.03 Entrevistas

Se realiza una serie de preguntas al personal involucrado para el correcto funcionamiento del flujo del proceso, con el fin de conocer cómo se maneja cada área del a empresa e identificar los procesos que se puedan automatizar.

Ver anexo A. Entrevista

2.01.04 Matriz de requerimientos

Describe los requerimientos funcionales y no funcionales que son parte del desarrollo del proyecto, tomando en cuenta la prioridad, los involucrados, y estado en que se encuentra.

Ver anexo B. Matriz de requerimientos Funcionales.

Ver anexo C. Matriz de requerimientos No Funcionales.

2.01.5 Descripción detallada

Representa un de manera detallada cada requerimiento tanto funcional como no funcional con el fin de poder definir cada proceso que se ejecuta en cada requerimiento.

Tabla 2.

Requerimiento funcional RF001

Identificador:	RF 001
Nombre:	El sistema informático debe permitir realizar el envió de cotizaciones mediante correo electrónico adjuntando la oferta que se genera en el cotizador de la compañía de seguros.
Descripción:	Cuando un cliente solicita una cotización, el asesor comercial debe enviar una oferta con los valores que le costara el seguro a un cliente, dicha oferta puede variar en base a cada compañía de seguros.
Actores:	Jefe Comercial, Ejecutivo Comercial o subagente.
Prioridad:	Alta
Precondiciones:	• Se debe disponer de los cotizadores de cada compañía de seguros. • El cliente debe estar registrado en el sistema. • Se debe contar con al menos una compañía de seguros registrada. • Se debe contar con los distintos ramos de seguros para generar la cotización.
Flujo de Eventos:	• Al ingresar a la pantalla de seleccionar cotización, se debe presionar el botón nuevo. • Se debe seleccionar un cliente registrado que debe ser al que se envía la cotización. • Al seleccionar el cliente se presenta una interfaz de registro la cual presenta los datos del cliente seleccionado y la vigencia de la cotización. • Se debe seleccionar la compañía y ramo de seguros a cotizar. • Se ingresa las posibles vigencias del seguro lo cual permite que se habilite la opción de registrar un nuevo ítem asegurado. • Al presionar el botón agregar objeto, se presenta un panel el cual permite el registro del nuevo objeto asegurado. • Una vez ingresado al menos un objeto asegurado, se debe guardar la información, habilitando el botón de envío de cotización. • Al ingresar al botón de envío de cotización, se presenta un cuadro de texto con la estructura base del email a enviarse con los datos registrados de la cotización. • Se debe confirmar el correo del cliente y presionar el botón enviar.
Flujo de Eventos Alternativos:	• Al seleccionar el ramo vehículos, se presenta un panel de registro con los datos y características del vehículo, si no es ramo vehículos se registra una ubicación con sus objetos asegurados. • En caso de que no se registre la fecha de vigencia no será posible el registro del ítem asegurado.
Postcondiciones:	Con el registro de una cotización en el sistema, da paso a registrar la solicitud de emisión o directamente la emisión en el sistema

Nota: Requerimiento funcional detalla el proceso ideal para poder registrar una cotización y que sea enviada al cliente.

Tabla 3.

Requerimiento Funcional RF 002

Identificador:	RF 002
Nombre:	El sistema debe permitir enviar la solicitud de emisión de una póliza mediante correo electrónico al contacto registrado de la compañía de seguros.
Descripción:	Una vez que el cliente aprueba la cotización, el trabajo del asesor comercial es solicitar la emisión de la póliza al ejecutivo encargado dentro de la compañía de seguros.
Actores:	Jefe Comercial, Asesor comercial o Subagente.
Prioridad:	Alta
Precondiciones:	Debe existir una cotización registrada en el sistema para poder enviar la solicitud de emisión a la compañía de seguros.
Flujo de Eventos:	• Al ingresar a la pantalla de emisión, se debe buscar el registro de cotización previamente registrado. • Se lo selecciona y se envía a la pantalla de emisión. • En la pantalla de emisión se debe ir al botón forma pago, en el cual se registra la modalidad de pago. Puede ser crédito, debido bancario, tarjeta de crédito o de contado. • Una vez registrada la forma de pago el sistema habilita las opciones para enviar la solicitud de emisión, dicha solicitud puede ser por carta que genera el sistema o mediante correo electrónico.
Flujo de Eventos Alternativos:	• En caso de que no se registre la forma de pago no se podrá realizar la orden de emisión en el sistema. • En caso de que ya se disponga de la emisión de una póliza, el usuario podrá registrar como emitida la póliza en el sistema siempre y cuando se registre el número de contrato y el número de factura por parte de la compañía de seguros.
Postcondiciones:	Con la emisión de la póliza, el sistema da paso a poder registrar los pagos que se recepta de las cuotas por parte del cliente.

Nota: Requerimiento funcional que detalla el flujo de eventos que se realiza para la emisión y solicitud de emisión de una póliza a la compañía de seguros.

Tabla 4.

Requerimiento Funcional RF 003

Identificador:	RF 003
Nombre:	El sistema debe permitir registrar la emisión de una póliza con los datos enviados por la compañía de seguros.
Descripción:	Cuando se solicita una emisión a la compañía de seguros, el ejecutivo de la compañía de seguros envía los contratos y factura al asesor comercial, adjuntando factura por dicho contrato y un número de póliza mismos que validan la emisión del seguro.
Actores:	Ejecutivo compañía de seguros, Ejecutivo Comercial.
Prioridad:	Alta
Precondiciones:	El sistema debe disponer de cotizaciones registradas para poder emitirlas.
Flujo de Eventos:	• en la pantalla de emisiones pendientes, se selecciona la cotización registrada. • Al ingresar se presenta la información de la cotización con los datos del cliente, compañía de seguros, ramo de seguros, objetos asegurados y forma de pago. • para emitir la póliza se debe registrar el número de contrato y numero de factura. • Registrada la información se presiona el botón guardar.
Flujo de Eventos Alternativos:	• El número de factura del contrato de seguros, debe constar de 15 dígitos, en caso de que no sea de esta manera, el sistema impide la emisión de la póliza.
Postcondiciones:	Con la emisión de una póliza, el sistema permite registrar cobro de las cuotas del seguro.

Nota: Requerimiento funcional que permite ver el flujo de eventos ideal para la emisión de una póliza en el sistema.

Tabla 5.

Requerimiento Funcional RF 004

Identificador:	RF 004
Nombre:	El sistema debe permitir registrar pagos que son realizados por un cliente, dichos pagos son en base a las cuotas de su seguro.
Descripción:	Cuando una póliza se encuentra emitida, lo siguiente es registrar los pagos de las cuotas o letras que paga el cliente basándose en la forma de pago escogida por el cliente.
Actores:	Ejecutivo de Cobranzas, Cliente.
Prioridad:	Alta
Precondiciones:	El sistema debe disponer de pólizas emitidas y registradas con número de factura y numero de contrato.
Flujo de Eventos:	• En la pantalla de cobranzas, se debe realizar la consulta de las pólizas registradas por número de contrato, nombres o apellidos del cliente, numero de factura o todos. • Una vez realizada la consulta, se debe seleccionar el registro sobre el cual se desea registrar el pago y presionar el botón siguiente. • Se presenta un panel el cual muestra las cuotas del cliente. • Se seleccionará una o más de una sobre las que se aplicará el pago y presionar el botón pago. • Esto presenta un panel con el detalle del valor a pagar de las cuotas seleccionadas, en dicho panel se debe seleccionar el medio de pago, puede ser cheques, efectivo, tarjeta, transferencia. Adicional se registra si el pago es realizado al bróker o a la compañía de seguros y quien es el cobrador. • Una vez registrado los datos se debe presionar el botón pagar, esto presenta un mensaje de confirmación, el cual al aceptar emite un recibo el cual el asesor de cobranzas puede imprimirlo o enviarlo por correo electrónico.
Flujo de Eventos Alternativos:	• En caso de que se seleccione una sola cuota a pagar y el pago no sea completo se registrara como abono de la cuota. • Si no se registra los datos de pago, no será posible generar el recibo de pago.
Postcondiciones:	Con el registro de pago de las cuotas del cliente, se habilitará la opción de cobro de comisiones a la compañía de seguros.

Nota: Requerimiento funcional permite verificar el flujo de eventos ideal y posibles alternativas a presentarse para el proceso de registro de pagos por parte del cliente.

Tabla 6.

Requerimiento Funcional RF 005

Identificador:	RF 005
Nombre:	El sistema debe permitir enviar el detalle de pagos por parte los clientes a cada compañía de seguros.
Descripción:	El ejecutivo de cobranzas tiene la obligación de notificar diariamente los pagos que se registró por parte de los clientes, siempre y cuando hayan sido receptados en el bróker.
Actores:	Ejecutivo de Cobranzas.
Prioridad:	Media
Precondiciones:	Debe existir en el sistema registros de pagados por parte del cliente y que sean realizados en el bróker.
Flujo de Eventos:	• En la pantalla de remisiones de pagos, se presenta una lista de compañías de seguros y un rango de fechas para generar una consulta de los pagos registrados. • Al generar la consulta se debe seleccionar los registros a enviar. • Una vez seleccionados, se habilita el botón remisión de pagos. • Al ingresar al botón se presenta un reporte con el detalle de los registros cobrados y que fueron seleccionados. • Dicho reporte el usuario puede imprimirlo o guardarlos para que sea enviado por correo electrónico.
Flujo de Eventos Alternativos:	• Si no se registra pagos en el sistema no será posible enviar el reporte de pagos a la compañía de seguros.
Postcondiciones:	No aplica.

Nota: Requerimiento funcional detalla el flujo de eventos presentados para el envío del detalle de pagos por parte de los clientes a la compañía de seguros.

Tabla 7.

Requerimiento Funcional RF 006

Identificador:	RF 006
Nombre:	El sistema deberá permitir comparar el detalle de las comisiones enviado por la compañía de seguros con la información de la producción pagada en el sistema.
Descripción:	Cundo la compañía de seguros verifica los pagos realizados por el cliente, envía el detalle de las comisiones listas para ser liberadas, con dicho detalle el jefe de operaciones realiza la comparación de la información con el detalle de producción pagada.
Actores:	Jefe de Operaciones, Aseguradora.
Prioridad:	Alta.
Precondiciones:	Debe existir el registro de pago de la cuota en la que se libera la comisión.
Flujo de Eventos:	• El sistema presenta una pantalla de pre facturas, en la cual se debe seleccionar una compañía de seguros del listado en pantalla, un rango de fechas que hace referencia al pago de la cuota. • Al generar la consulta el usuario debe seleccionar los registros que se encuentra en el detalle de comisiones proporcionado por la compañía de seguros. • Cada que se selecciona un registro se presenta el valor de comisión que se cobrara en base al porcentaje de comisión acordado por la compañía de seguros. • Una vez seleccionado todos los registros se detalla el total a recibir de comisión, mismo que debe coincidir con el detalle de la compañía de seguros. • Finalmente se presiona el botón pre factura, generando un detalle de los registros seleccionados.
Flujo de Eventos Alternativos:	• En caso de que una póliza registrada no se encuentre registrada el pago por parte del cliente, no se presentará en la pantalla para generar la pre factura. • En caso de que quede registros pendientes de seleccionar, se debe revisar con la compañía de seguros el motivo por el cual no se libera la comisión de dichos contratos.
Postcondiciones:	Con las pre facturas registradas, da paso a registrar la factura emitida a la compañía de seguros, con el detalle de los registros seleccionados o a cobrarse.

Nota: Requerimiento funcional detalle el flujo de eventos para realizar la comparación del detalle de comisiones con la información en el sistema.

Tabla 8.

Requerimiento Funcional RF 007

Identificador:	RF 007
Nombre:	El sistema debe permitir registrar la factura que se envía a la compañía de seguros con el detalle de valores de los contratos que se cobra comisión.
Descripción:	Una vez que el jefe de operaciones realiza la comparación del detalle de comisiones enviado por la compañía de seguros, se genera una factura a la aseguradora, misma que se registra en el sistema para poder llevar un control de cobros de comisión al bróker.
Actores:	Jefe de Operaciones, Aseguradora.
Prioridad:	Alta
Precondiciones:	Debe constar registrada una pre factura en el sistema.
Flujo de Eventos:	• En la pantalla de registro de facturas, se debe seleccionar una aseguradora del listado presentado y consultar los registros de pre facturas que se generaron. • Al presentar la consulta se debe seleccionar los registros deseados y presionar el botón registrar factura. • Presentará un panel de registro el cual solicita el número de factura y fecha en la que fue emitida.
Flujo de Eventos Alternativos:	Si no se dispone de pre facturas registradas no se podrá registrar facturas en el sistema. No se genera el documento de factura, ya que solo se lleva un registro de cobros en el sistema.
Postcondiciones:	Con la facturación registrada, se podrá liberar comisiones a los asesores comerciales.

Nota: Requerimiento funcional detalla el flujo de eventos presentados al momento de registrar una factura que fue emitida a la compañía de seguros, cabe aclarar que el sistema no genera el documento de factura ya que el registro únicamente es para poder llevar un control de cobros a la aseguradora.

Tabla 9.

Requerimiento Funcional RF 008

Identificador:	RF 008
Nombre:	El sistema debe permitir registrar el pago de las facturas emitidas a la compañía de seguros.
Descripción:	Cuando la compañía de seguros genera el pago de la factura emitida por el bróker, esta debe ser registrada.
Actores:	Jefe Operaciones.
Prioridad:	Alta
Precondiciones:	Debe existir facturas registradas en el sistema.
Flujo de Eventos:	• Al ingresar a la pantalla de cobro de facturas, se presenta un listado de aseguradoras para consultar las facturas registradas de dicha compañía. • Al presentar la consulta, seleccionar la factura que se desea registrar como pagada. • Esto presentará la fecha de pago al día en curso y se debe presionar grabar.
Flujo de Eventos Alternativos:	En caso de que no se seleccione ningún registro no se podrá tomar como pagada ninguna factura.
Postcondiciones:	Con el registro de pago de las facturas automáticamente se puede liberar la comisión a los asesores comerciales.

Nota: Requerimiento funcional detalla el flujo de eventos presentado para el registro de pago de una factura en el sistema.

Tabla 10.

Requerimiento Funcional RF 009

Identificador:	RF 009
Nombre:	El sistema debe permitir generar el detalle de comisiones para cada asesor comercial en base a los registros ingresados y que fueron facturados y cobrados a la compañía de seguros.
Descripción:	Cuando una factura fue cobrada a la compañía de seguros, esto permite realizar el pago de los contratos emitidos por cada asesor, en base al porcentaje de comisión que se acordó en el bróker.
Actores:	Jefe de operaciones.
Prioridad:	Alta
Precondiciones:	Para realizar el pago de comisión a los asesores, debe existir registradas facturas a la compañía de seguros.
Flujo de Eventos:	• En la pantalla de pago de comisiones subagentes, se presenta un listado de los subagentes o asesores comerciales, al seleccionar uno se debe presionar el botón buscar. • Esto presenta el detalle de los contratos en los cuales puede cobrar comisión dicho asesor. • Se selecciona los registros a pagar, generando un total de lo que se va a pagar. • Una vez seleccionados todos los registros, se debe presionar el botón liquidación de comisiones, mismo que genera un reporte con el detalle de comisiones para el asesor seleccionado.
Flujo de Eventos Alternativos:	En caso de que no se encuentren emitidas facturas no se presentará información de los contratos para que sea liberada la comisión al asesor comercial.
Postcondiciones:	No aplica.

Nota: Requerimiento funcional detalla el flujo de eventos a realizarse para poder realizar el pago de comisiones a un asesor comercial.

Tabla 11.

Requerimiento no Funcional RNF 001

Identificador:	RNF 001
Nombre:	El sistema tiene que controlar el acceso y se lo permitirá solo usuarios autorizados.
Descripción:	Proceso que permitirá dar un control de accesos al sistema dependiendo del perfil que corresponda a cada uno.
Actores:	Todos los usuarios.
Prioridad:	Media.
Precondiciones:	Debe existir creado el usuario y asignado una contraseña.
Flujo de Eventos:	• El usuario debe ingresar por medio de un explorador web al link que será entregado. • Al acceder al link se presentará una pantalla de acceso. • Dicha pantalla solicitara un usuario y una clave. • Al ingresar los dos parámetros de manera correcta el sistema permitirá el acceso al menú principal.
Flujo de Eventos Alternativos:	• En caso de no se posea un usuario o una clave el sistema impedirá el acceso. • Si el usuario o la contraseña son incorrectos el sistema alertara con un mensaje informando que no existe usuario y/o contraseña. • En caso de que el usuario exista y realice el ingreso erróneo de su clave por más de tres veces, el sistema procederá a bloquear el usuario impidiendo su acceso.
Postcondiciones:	• Al ingresar los datos correctos el sistema presentará una pantalla de inicio con los accesos correspondientes a su perfil.

Nota: Requerimiento no funcional el cual detalla el proceso de ingreso al sistema mediante usuario y contraseña.

Tabla 12.

Requerimiento no Funcional RNF 002

Identificador:	RNF 002
Nombre:	El sistema deberá permitir el registro de usuarios, cada usuario contará con un nombre de usuario, contraseña, nombres, apellidos, cargo que desempeña y un perfil de acceso.
Descripción:	El registro de un usuario es importante para que cada colaborador pueda acceder al sistema con sus credenciales de acceso.
Actores:	Administrador del sistema
Prioridad:	Media.
Precondiciones:	Contar con un listado de perfiles.
Flujo de Eventos:	• Al acceder al sistema en el menú principal se encuentra la opción parámetros, dentro se presenta un submenú denominado usuarios. • Para agregar un nuevo usuario se debe dar clic en el botón registrar, esto presentará una ventana de registro. • En dicha ventana se presenta los campos de ingreso que son nombres, apellidos, cargo, perfil, nombre de usuario y una contraseña. • Al completar los campos, se debe presionar el botón grabar, este proceso presentará un mensaje detallando que el registro fue exitoso.
Flujo de Eventos Alternativos:	• La pantalla de usuario permitirá realizar la modificación y des habilitación de usuarios. • Para modificar un usuario se deberá ubicarse sobre el registro del listado en pantalla. • Esto direccionara a la pantalla similar a la de ingreso, capturando la información el usuario a modificar. • En caso de que no se ingrese la información principal el sistema presentará un mensaje el cual indica que el registro no puede ser grabado.
Postcondiciones:	Con el registro de un usuario, se da los privilegios correspondientes de acuerdo al área y funciones, con esto permite el ingreso al sistema.

Nota: El requerimiento permite el registro de usuarios, los cuales pueden acceder al sistema con el perfil asignado.

Tabla 13.

Requerimiento no Funcional RNF 003

Identificador:	RNF 003
Nombre:	El sistema debe permitir registrar distintos perfile, cada perfil dará un distinto acceso al sistema a los usuarios registrados.
Descripción:	Al disponer de perfiles por usuario, se otorga un nivel de acceso al sistema a cada usuario.
Actores:	Administrador del sistema.
Prioridad:	Alta
Precondiciones:	Establecer el listado de accesos por perfil.
Flujo de Eventos:	• Al acceder al sistema, en la opción parámetros, se presenta el submenú perfiles. • Al acceder a la pantalla de perfiles, se presenta una pantalla la cual presenta el listado de perfiles, al ubicarse sobre cada perfil, se presenta el listado de acceso para dicho perfil. • En caso de registrar un perfil, se debe dar clic sobre el botón agregar, esto permitirá definir un nuevo perfil y solicitará el listado de accesos que deberá tener dicho perfil. • Al seleccionar los accesos se deberá presionar el botón grabar.
Flujo de Eventos Alternativos:	En caso de querer registrar un perfil sin accesos, el sistema inhabilitara dicho perfil.
Postcondiciones:	Asignación de perfiles a los usuarios registrados.

Nota: Requerimiento que solicita el ingreso de perfiles con sus respectivos accesos.

Tabla 14.

Requerimiento no Funcional RNF 004

Identificador:	RNF 004
Nombre:	El sistema debe permitir registrar los datos de la empresa y de ser el caso ingresar cada sucursal.
Descripción:	Es importante el registro de la empresa para poder realizar la identificación del bróker para reportes y cartas que se puedan generar a las distintas entidades.
Actores:	Administrador del sistema.
Prioridad:	Media
Precondiciones:	Disponer de los datos de la empresa como el ruc, razón social y nombre comercial.
Flujo de Eventos:	• Al ingresar al sistema en al menú principal se debe seleccionar la opción parámetros, dentro de ella existirá la opción datos del corredor. • Al ingresar a la pantalla de datos del corredor, se presentará una pantalla la cual permite visualizar la compañía registrada y sus sucursales. • Para agregar una nueva sucursal se debe presionar sobre el botón agregar, esto presentará un panel de registro. • En el panel de registro se solicitará el nombre de la compañía, el ruc y la razón social, en caso de ser sucursal se deberá seleccionar cual es la compañía matriz. • Al llenar los campos se debe presionar el botón grabar, esto cerrará el panel de ingreso y se presentará un mensaje informando que el registro fue grabado exitosamente.
Flujo de Eventos Alternativos:	• Al no ingresar los datos obligatorios, el sistema mostrara un mensaje informando que el registro no pudo ser grabado.
Postcondiciones:	Permite identificar la compañía en la que se está trabajando actualmente.

Nota: Requerimiento no funcional, que permite el registro del nombre de la empresa que utiliza el sistema, con el fin de realizar parámetros iniciales.

Tabla 15.

Requerimiento no Funcional RNF 005

Identificador:	RNF 005
Nombre:	El sistema debe permitir registrar N número de contactos por compañía de seguros, cada contacto debe contar con nombres, apellidos, correo electrónico, cargo y teléfonos.
Descripción:	Al disponer de información de contactos, se puede enviar las notificaciones a la compañía de seguros.
Actores:	Jefe Operativo, Gerente Comercial, Gerente General.
Prioridad:	Media
Precondiciones:	• Debe existir registrada ya una compañía de seguros.
Flujo de Eventos:	• En la pantalla de aseguradoras, se presentará un botón denominado contactos. • Al dar clic se presenta un panel con el listado de contactos de la compañía seleccionada. • Para registrar se presentará un botón de ingreso, esto presentará otro panel de registro. • En el panel de registro se debe ingresar los datos del contacto, como nombres, apellidos, correo electrónico, teléfonos y cargo. • Al completar de ingresar la información se debe dar clic en el botón grabar el cual cerrara el panel y actualizara la lista de contactos.
Flujo de Eventos Alternativos:	• En caso de que se quiera registrar información sin datos el sistema presentará un mensaje informando que el registro no puede ser grabado.
Postcondiciones:	• El contacto permite ser visualizado al momento de enviar las notificaciones a la compañía de seguros.

Nota: La presente tabla detalla el proceso para el registro de contactos para la compañía de seguros.

Tabla 16

Requerimiento no Funcional RNF 006

Identificador:	RNF 006
Nombre:	El sistema debe tener un registro de seguimiento para los clientes, en el cual se podrá registrar el detalle de lo que se ha tratado con el cliente desde el proceso de cotización hasta completar la emisión.
Descripción:	Cada que un asesor comercial realiza el contacto con un cliente se lo considera como seguimiento, considerando que actividad realizo con el cliente.
Actores:	Ejecutivo comercial.
Prioridad:	Media.
Precondiciones:	Debe existir un registro de cotización en trámite.
Flujo de Eventos:	• En la pantalla de cotización, visualizará un botón denominado seguimiento, al dar clic se presentará el listado del seguimiento que se ha realizado, y los campos para realizar un nuevo registro. • En caso de registrar un nuevo seguimiento, se debe llenar los campos y presionar el botón grabar, • Al grabar la información se actualizará la lista de seguimiento colocando como activo el ultimo seguimiento registrado.
Flujo de Eventos Alternativos:	• En caso de querer registrar un seguimiento sin datos se presentará un mensaje informando que no es posible guardar la información.
Postcondiciones:	• Los registros ingresados en el seguimiento, serán visualizados en un reporte de seguimientos por asesor comercial.

Nota: Detalle del registro de seguimiento que se realiza por cliente.

Tabla 17.

Requerimiento no Funcional RNF 007

Identificador:	RNF 007
Nombre:	El sistema debe permití realizar consultas de las pólizas emitidas, en las que se podrá visualizar los objetos asegurados, vigencias, valores y forma de pago.
Descripción:	Al poder realizar la consulta en el sistema se puede evidenciar si los movimientos fueron emitidos de manera correcta, o si se puede consultar algún tipo de información adicional.
Actores:	Jefe Operativo, Jefe Comercial, Gerente General.
Prioridad:	Media
Precondiciones:	Debe existir pólizas emitidas en el sistema.
Flujo de Eventos:	• Al acceder al sistema se presentará en el menú una opción de consultas, dentro de ella se reflejará la opción consulta de pólizas. • Al ingresar se presentará una pantalla para consultar por nombre, apellidos, número de póliza, ramo, aseguradora y factura, cada uno de dichos campos son como parámetros de búsqueda. • Para visualizar los datos de la póliza se debe dar clic sobre el registro deseado, esto abrirá una ventana en la cual se visualiza los datos de la póliza emitida.
Flujo de Eventos Alternativos:	En caso de que no se presente información el sistema presentará un mensaje informando que no existe datos para la consulta realizada.
Postcondiciones:	No aplica.

Nota: La tabla presenta el requerimiento para realizar la consulta de pólizas emitidas, para validación de información.

Tabla 18.

Requerimiento no Funcional RNF 009

Identificador:	RNF 009
Nombre:	El sistema debe permitir realizar la reversión de pagos registrados por error.
Descripción:	La reversión de un pago registrado es importante realizarla ya que puede existir error humano en el proceso.
Actores:	Administrador del Sistema, Jefe Operativo, Usuarios autorizados.
Prioridad:	Alta
Precondiciones:	Debe existir pagos ingresados al sistema, caso contrario no se presentará información.
Flujo de Eventos:	• Al acceder al sistema en el menú principal en la opción cobranzas se presentará un submenú denominado reversar cobranza. • Al acceder a esta opción se presentará una pantalla para realizar la consulta, dicha consulta se la puede realizar por nombres, apellidos, número de póliza y factura, en caso de no registrar ningún campo se presentará toda la información de pagos registrados. • Para el reverso se debe seleccionar uno de los registros presentados por la consulta y presionar el botón reversar. • Esto redireccionará a una venta la cual presentará las cuotas pagadas de la póliza. • Se debe seleccionar cuál de las cuotas se realizará el reverso. • Al seleccionar existirá el botón reversar pago. • Esta acción presentará un mensaje de confirmación de reverso, al dar clic en si se presenta un mensaje informando que el proceso se ejecutó correctamente.
Flujo de Eventos Alternativos:	• En caso de que se desee reversar un registro el cual ya fue generado factura y pre-factura el sistema presentará una alerta informando que el registro ya está en proceso de cobro de comisiones.
Postcondiciones:	Al realizar la reversión de un pago, dicho registro podrá ser visible para aplicar nuevamente el pago correspondiente.

Nota: La siguiente tabla presenta el proceso de reversión de pagos en el sistema.

Tabla 19.

Requerimiento no Funcionala RFN 010

Identificador:	RNF 010
Nombre:	El sistema tendrá una pantalla de consulta de las cuotas pagadas, con el detalle de valor pagado, fecha, usuario que registro el pago, origen de pago, numero de cuota, estado y forma de pago.
Descripción:	Al poder realizar una consulta de los pagos realizados por el cliente se puede evidenciar su estado de cuenta.
Actores:	Ejecutivo de cobranzas, Ejecutivo Comercial, Jefe Operativo, Gerente General.
Prioridad:	Media
Precondiciones:	Debe existir pagos registrados en el sistema.
Flujo de Eventos:	• Al acceder al sistema, en el menú se visualiza la opción cobranzas, dentro de esta opción se presenta el submenú consulta cobranzas. • Al acceder se presenta un apantalla de consulta la cual solicita parámetro de nombres, apellidos o número de factura, al ingresar un parámetro se presiona buscar. • A su vez es posible consultar información sin parámetros, esto presentará todo lo registrado como pagado o no pagado.
Flujo de Eventos Alternativos:	• En caso de que no exista información el sistema presentará mensaje informando que no hay datos que mostrar.
Postcondiciones:	No aplica.
Criterios de aceptación:	

Nota: Con la pantalla de consulta de cobranzas se podrá verificar si las cuotas del cliente fueron pagadas o siguen pendientes.

Tabla 20.

Requerimiento no Funcional RFN 011.

Identificador:	RNF 011
Nombre:	El sistema debe permitir realizar la consulta de las facturas, tanto cobradas y que se encuentran pendientes de cobro.
Descripción:	Al poder identificar que facturas están pagadas y cuales están pendientes de cobro se podrá realizar un seguimiento a la compañía de seguros.
Actores:	Jefe Operativo.
Prioridad:	Alta
Precondiciones:	Debe existir facturas emitidas.
Flujo de Eventos:	• Al acceder al sistema en el menú principal en la opción facturación, se presentará el submenú consulta facturas. • En dicha pantalla se presentará un listado de aseguradoras y fecha que son los parámetros de búsqueda. • Al seleccionar algún parámetro y dar clic en el botón buscar se presentará la información de las facturas. • Cada registro presentará un detalle sobre la factura seleccionada.
Flujo de Eventos Alternativos:	Si no existe información se presentará un mensaje de información indicando que no existen datos.
Postcondiciones:	No aplica.

Nota: La presente tabla detalla la funcionalidad de que se pueda realizar la consulta de facturas emitidas a la compañía de seguros.

2.02 Mapa de involucrados

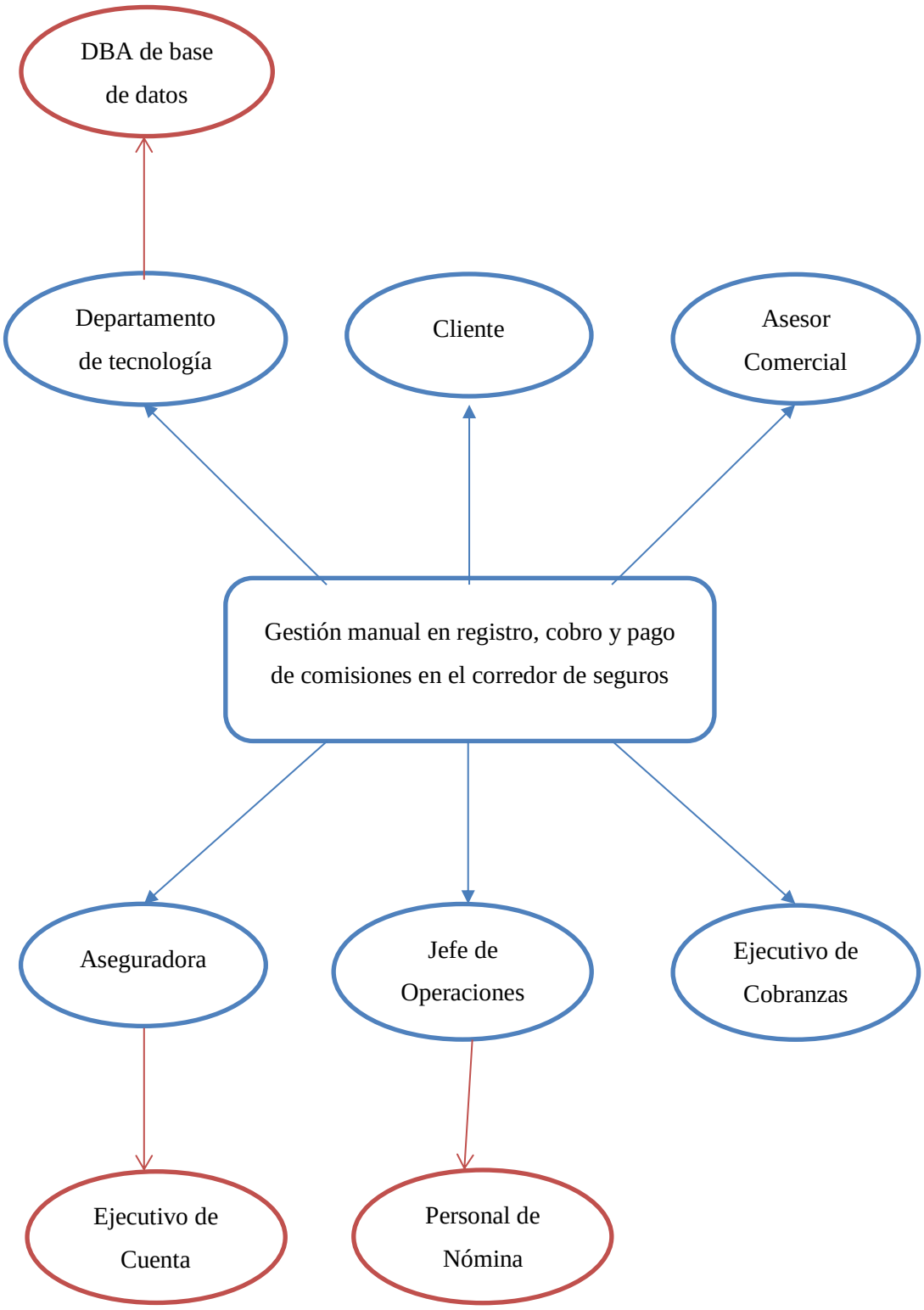


Figura 1. Mapa de Involucrados.

Permite identificar los involucrados o actores para la solución de la investigación.

2.03 Matriz de análisis de involucrados

Tabla 21.

Matriz de Involucrados

Actores Involucrados	Intereses sobre el problema central	Problemas percibidos	Recursos, mandatos y capacidades	Intereses sobre el proyecto	Conflictos potenciales
Cliente	Mejorar el canal de comunicación de parte del corredor de seguros.	Falta de comunicación al cliente		Mayor eficiencia en la comunicación.	
Ejecutivo Comercial	Agilizar el proceso de emisión de una póliza.	Procesos manuales y perdida de información		Mayor efectividad en el ingreso de producción	Falta de conocimiento de los productos
Ejecutivo cobranzas	Agilizar el proceso de cobro a los clientes y mejorar la notificación a la aseguradora	Demora en el registro de pagos a la compañía de seguros.		Mayor control en el registro de pagos del cliente.	Retraso en la comunicación del pago realizado.
Jefe Operativo	Facilitar el proceso de preliquidación de comisiones.	La preliquidación solo se toma lo reportado por cada asesor		Mejor efectividad al proceso de pre liquidación y pago de comisiones	Demora en la recepción del detalle de comisiones
Aseguradora	Mejorar el sistema de comunicación con el corredor de seguros.	Descoordinación con el corredor de seguros		Reducir el índice de error con la información del corredor de seguros	Falta de información por parte del corredor de seguros
Departamento de TI	Implementar un sistema informático que permita realizar el seguimiento de cobro y pago de comisiones.	Falta de un sistema informático		Agilizar el proceso que desempeña el bróker.	

Nota: Tabla de matriz de involucrados, detalla los involucrados del área administrativa.

CAPÍTULO III

3. Problema y objetivos:

3.01 Árbol del Problema

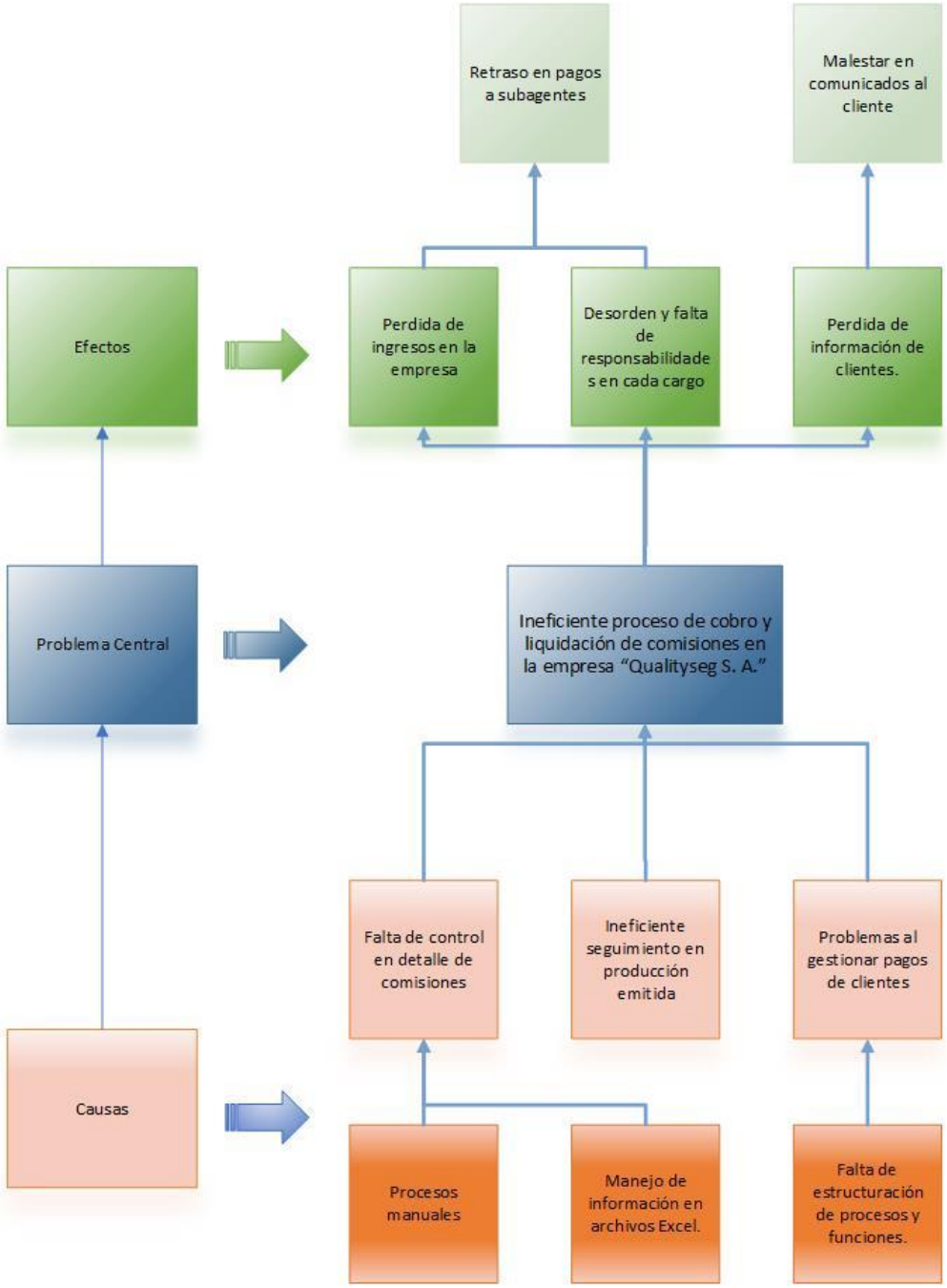


Figura 2. Árbol de Problemas.

Define los problemas que se presentan en la investigación realizada a la empresa Qualityseg S.A., con el fin de poder identificar los conflictos que se presenta en el corredor de seguros.

3.02 Árbol de objetivos

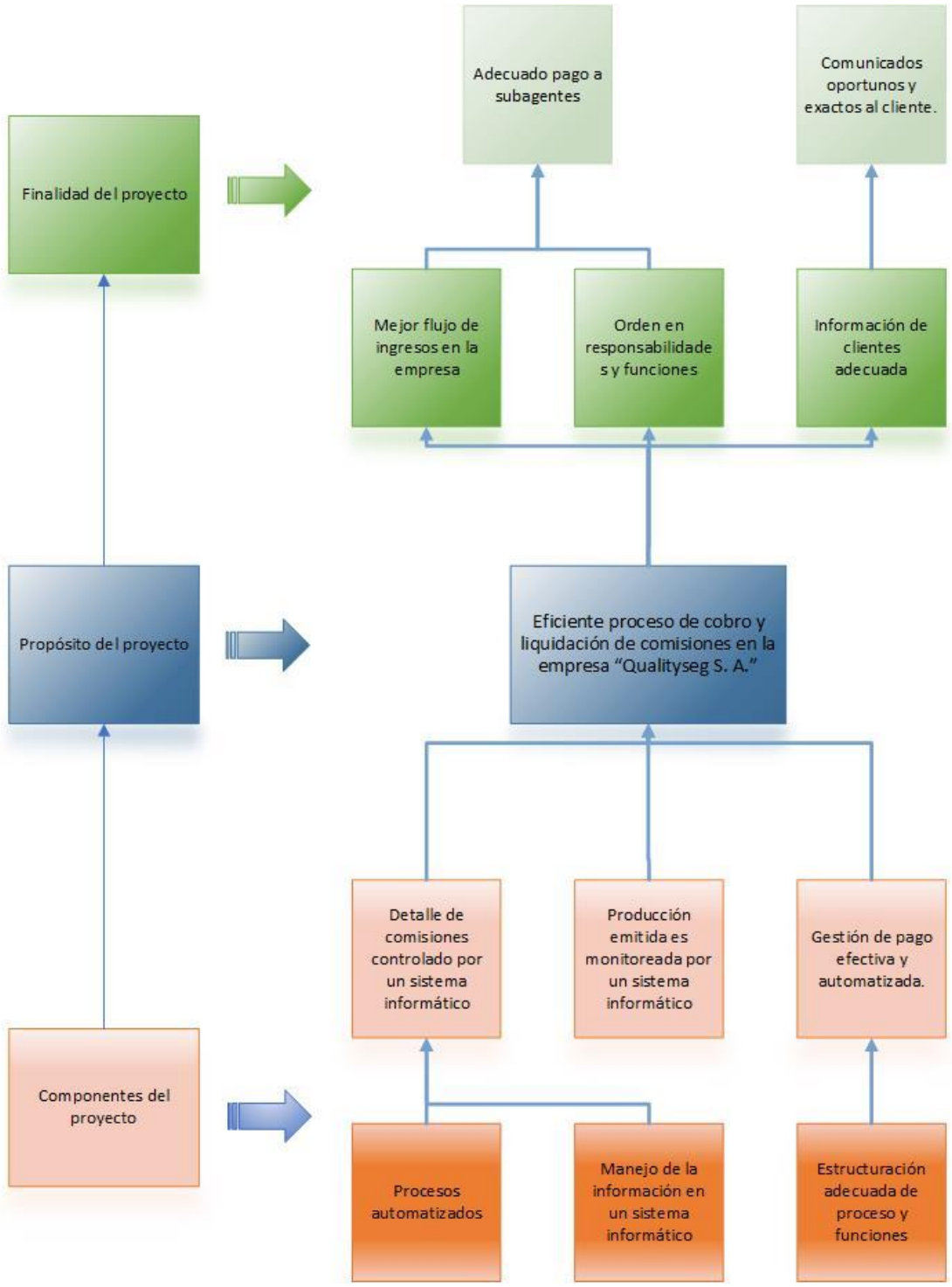


Figura 3. Árbol de Objetivos.

Detalla los objetivos que la empresa Qualityseg desea obtener en el transcurso del desarrollo de la investigación.

3.03 Casos de uso

Con los diagramas de caso de uso se puede evidenciar la relación que tiene cada actor en cada proceso.

Caso de uso CU001: Proceso general de un corredor de seguros.

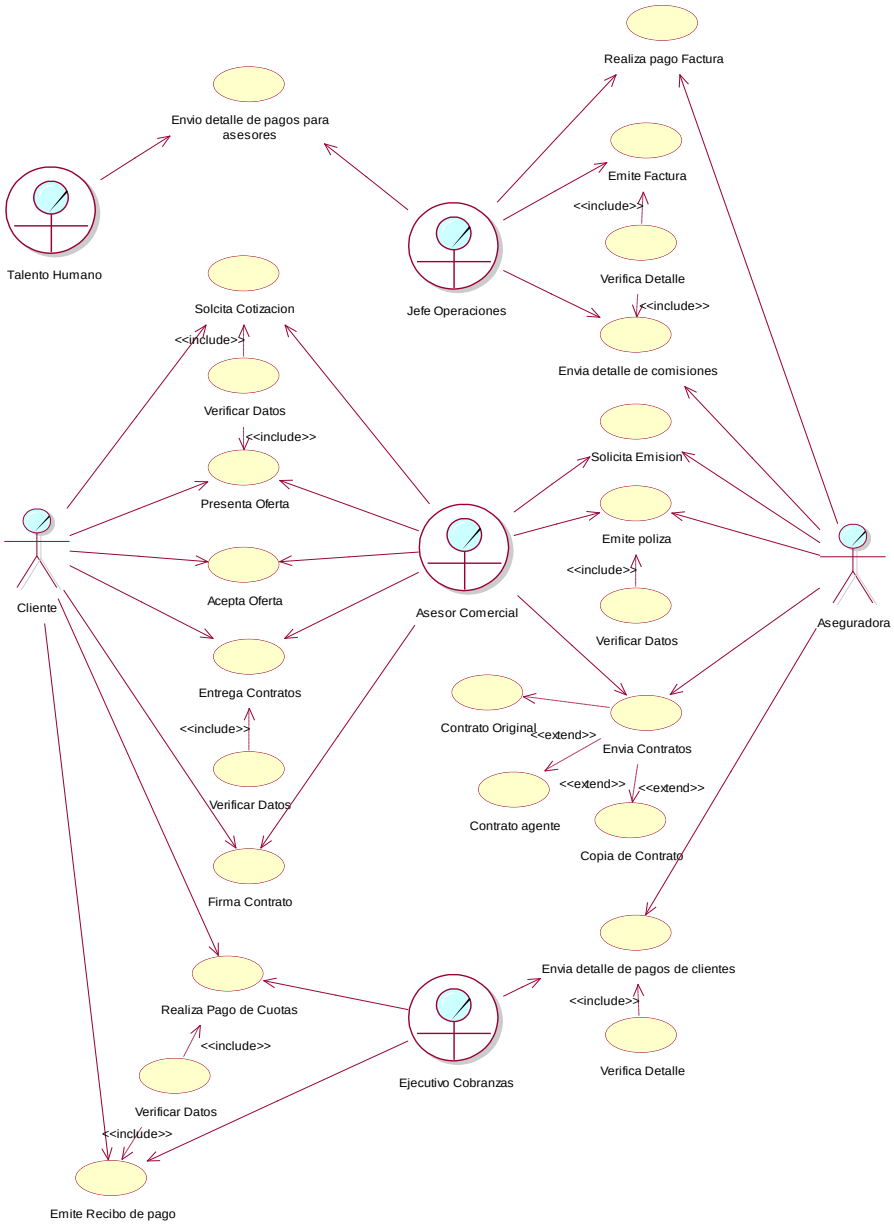


Figura 4. Diagrama de caso de uso general.

Representa todos los actores y procesos a los cuales se encuentran relacionados.

Caso de uso CU002: Proceso de cotización a un cliente.

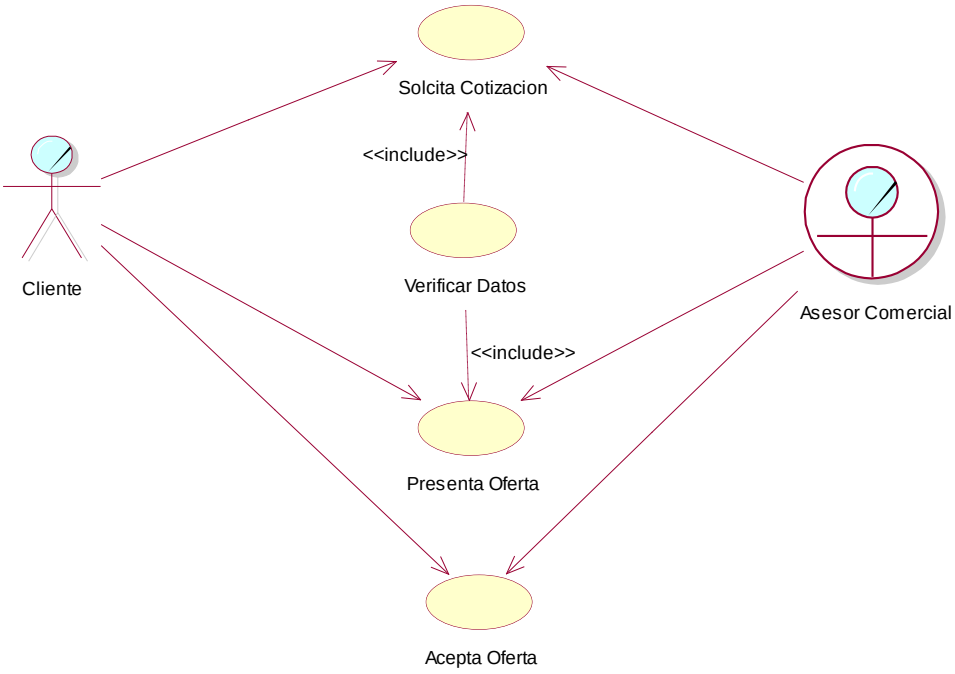


Figura 5. Diagrama de caso de uso, Cotización Cliente.

Detalla el proceso que se genera entre los actores cliente y el asesor comercial.

Caso de uso CU003: Proceso de emisión de pólizas.

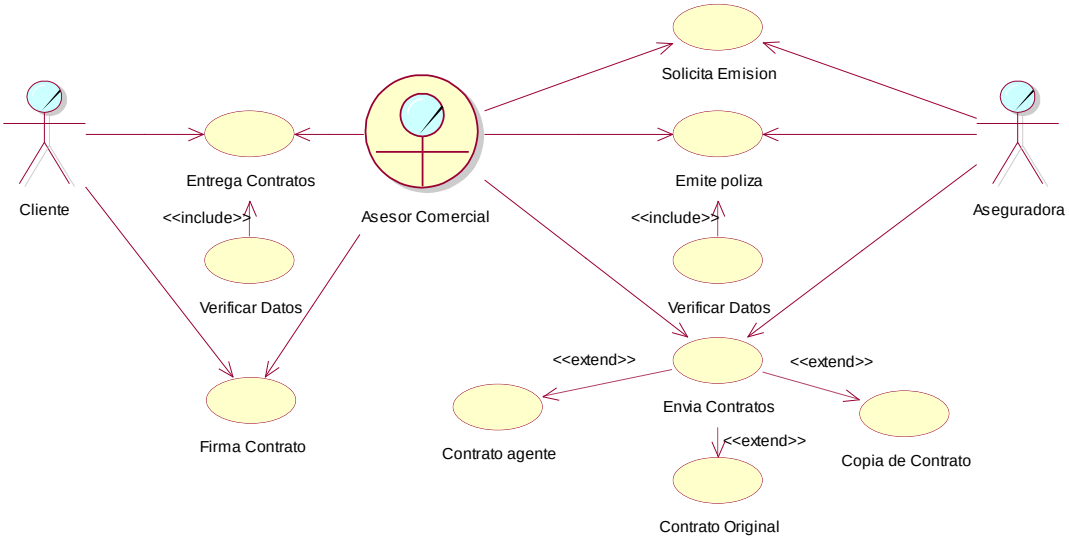


Figura 6. Diagrama de caso de uso, Emisión Pólizas.

Detalla el proceso que ejecuta el autor ejecutivo de cuenta con el cliente y la compañía de seguros para poder realizar la emisión de una póliza.

Caso de uso CU004: Proceso de cobranzas a clientes.

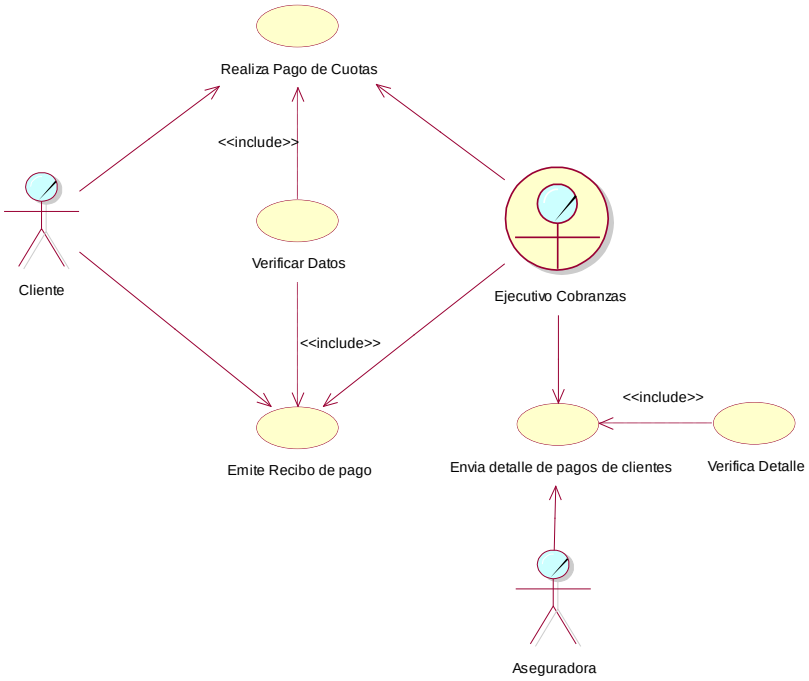


Figura 7. Diagrama de caso de uso, Cobros Clientes.

Detalla el proceso que realiza el ejecutivo de cobranzas al momento de receptar el pago de un cliente.

Caso de uso CU005: Proceso de facturación a Aseguradora.

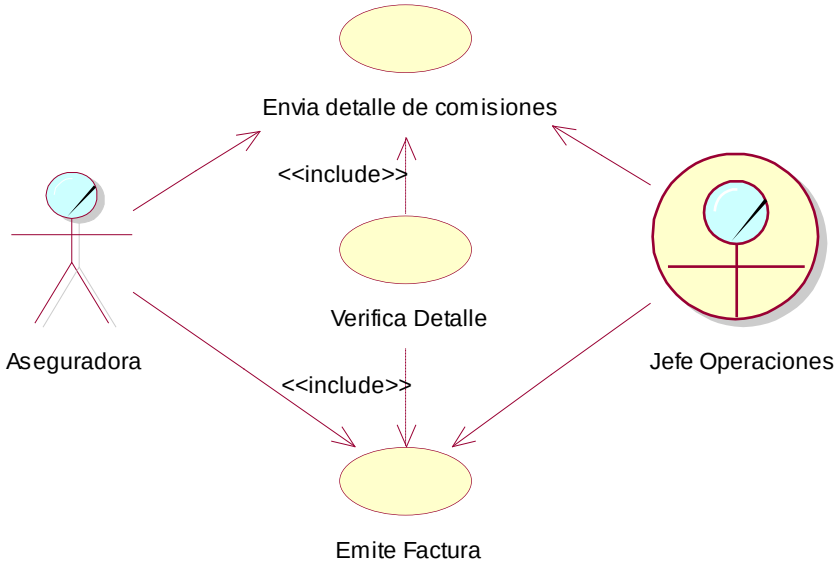


Figura 8. Diagrama de caso de uso, Facturación.

Detalla el proceso que se realiza para emitir la factura para el pago de comisiones que se generó de los negocios vendidos y que fueron cobrados a los clientes.

Caso de uso CU006: Proceso de cobro de facturas y pago de comisiones.

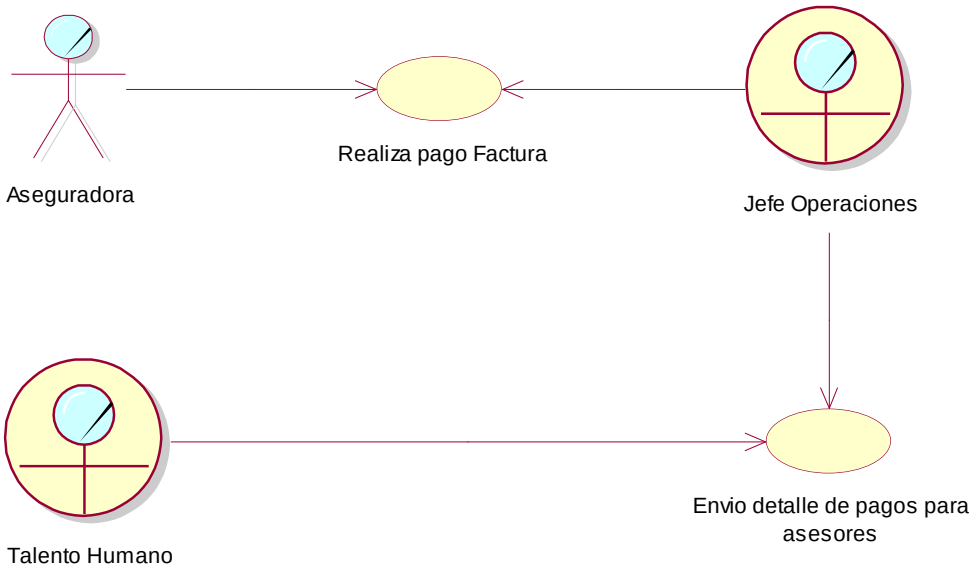


Figura 9. Diagrama de caso de uso, Cobro Facturas.

Detalla el proceso que ejecuta el jefe de operaciones desde el cobro de facturas que se emite a la compañía de seguros.

3.04 Especificación de casos de uso

La especificación de los casos de uso hace referencia a una descripción detallada de cada una de las partes definidas para poder describir un proceso de manera completa, definiendo los involucrados y el o los pasos a seguir de cada uno de ellos en cada caso de uso.

Se debe definir una descripción (Detalle corto y concreto del caso de uso), los flujos de eventos Evento ideal de un proceso), flujos de eventos alternativos (Eventos alternativos que pueden suscitar en un proceso), las fuentes (Personal de la empresa que ayudo con el caso que se presenta.), las precondiciones y post condiciones y finalmente los puntos de inclusión y exclusión que se ejecutan con el caso de uso a detallar.

Tabla 22.

Caso de uso Solicitar cotización.

Caso de uso	CU1. Solicitar Cotización
Fuentes	Hernán Ochoa (Gerente General) Luis González (Jefe de Operaciones) Paul Chalem (Jefe Comercial)
Actor	Act1. Cliente Act2. Asesor Comercial.
Descripción	Solicitud de cotización de parte del cliente a un asesor comercial del corredor de seguros.
Flujo básico	1. El cliente se pone en contacto con el asesor comercial de seguros. 2. Al contactarse solicita una cotización de un seguro. 3. El asesor comercial genera varias ofertas las cuales son presentadas al cliente. 4. El cliente revisa las ofertas presentadas y toma la decisión sobre una de las ofertas presentadas. 5. Fin del caso de uso.
Flujo alternativo	1. El cliente puede solicitar cotizaciones para distintos tipos de seguros, estos pueden ser seguro de vehículos, seguro de incendios, seguro para equipos y maquinarias, seguros para equipos electrónicos y seguros de fianzas. 2. En caso de que el cliente no acepte una de las ofertas presentadas, dichas cotizaciones deben ser rechazadas.
Pre - condiciones	
Post - condiciones	1. Una vez aceptada la cotización, el asesor comercial puede realizar el proceso de emisión.
Puntos de inclusión	1. El asesor comercial verifica datos del cliente para poder emitir una cotización.
Puntos de extensión	No aplica.

Nota: Descripción de caso de uso de solicitud de cotización, y su proceso de funcionamiento.

Tabla 23.

Caso de uso Presentar Oferta.

Caso de uso	CU2. Presentar Oferta de cotización
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial)
Actor	Act1. Cliente Act2. Asesor Comercial.
Descripción	El asesor comercial presenta varias ofertas al cliente.
Flujo básico	1. Cuando el asesor comercial recibe la solicitud por parte del cliente, revisa los productos disponibles. 2. Definido el producto, se genera la oferta al cliente con los datos del objeto que requiere asegurar el cliente.
Flujo alternativo	1. Al revisar los productos disponibles, se evidencia que no existe algo que el cliente solicita. 2. No se genera la cotización ya que es una renovación del seguro.
Pre - condiciones	1. Se debe tener definidos los productos para ofrecer al cliente con al menos una compañía de seguros. 2. Se debe contar con el cotizador proporcionado por la compañía de seguros. 3. El cliente debe proporcionar los datos del o los objetos que desea asegurar.
Post - condiciones	1. Con la presentación de oferte, el cliente podrá tomar una decisión si acepta la cotización enviada.
Puntos de inclusión	El cliente verifica los datos de la oferta, lo cual le permite tomar una decisión para aceptar o no la oferta.
Puntos de extensión	No aplica

Nota: Especificación del caso de uso para presentación de una oferta o cotización al cliente, detalla el proceso el cual se realiza para poder generar una oferta y pueda ser presentada al cliente.

Tabla 24.

Caso de uso Aceptar Oferta.

Caso de uso	CU3. Aceptar Oferta
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial)
Actor	Act1. Cliente Act2. Asesor Comercial.
Descripción	El cliente realiza la aprobación de la oferta proporcionada por el asesor comercial
Flujo básico	1. Cuando el asesor comercial ha presentado su oferta el cliente la verifica. 2. Una vez revisada la oferta, el cliente procede a aceptar la oferta. 3. La aceptación la notifica al asesor comercial para que proceda con la emisión de la póliza.
Flujo alternativo	1. El cliente al verificar la cotización, no acepta las condiciones ofertadas, por lo tanto, no continua con el proceso. 2. Los datos de la oferta no son correctos sobre los riesgos a asegurar.
Pre - condiciones	No Aplica.
Post - condiciones	1. Cuando el cliente acepta la oferta, da apertura para que el asesor comercial pueda enviar la solicitud de emisión a la compañía de seguros.
Puntos de inclusión	No aplica
Puntos de extensión	No aplica

Nota: Detalle de caso de uso para aceptar oferta, detalla el proceso el cual se ejecuta cuando un cliente realiza la aprobación de una oferta proporcionada por el asesor comercial del corredor de seguros.

Tabla 25.

Caso de uso Solicitud de Emisión

Caso de uso	CU4. Solicitud de Emisión
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial)
Actor	Act1. Asesor Comercial Act2. Aseguradora
Descripción	Cuando un cliente acepta la cotización, el asesor comercial solicita la emisión a la compañía de seguros.
Flujo básico	1. El asesor comercial solicita a la compañía de seguros se realice la inspección del riesgo. 2. Se solicita al cliente que envié los documentos de vinculación que son: formulario de vinculación de cada compañía de seguros, cédula del titular y cédula del cónyuge de ser el caso, papeleta de votación del titular y del cónyuge y planilla de servicio básico. 3. Una vez que se obtiene toda la información, se envía a la compañía de seguros la solicitud de emisión por correo electrónico.
Flujo alternativo	• El ejecutivo comercial no dispone de la información de los documentos de vinculación que se solicita al cliente. • En la solicitud de emisión no se presenta los valores correctos de valores asegurados.
Pre - condiciones	• Se debe solicitar documentos de vinculación al cliente.
Post - condiciones	Da paso a la compañía de seguros a emitir la póliza.
Puntos de inclusión	No aplica
Puntos de extensión	No aplica

Nota: Especificación de caso de uso solicitud de emisión, detalla cada paso que se ejecuta para el proceso de solicitud de emisión, se detalla de igual manera que posibles flujos alternativos.

Tabla 26.

Caso de uso Emisión de póliza

Caso de uso	CU5. Emisión de Póliza.
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial) Carina Tufiño (Asesora Comercial)
Actor	Act1. Asesor comercial Act2. Aseguradora
Descripción	Cuando un asesor comercial envía la solicitud de emisión, la compañía de seguros emite un contrato denominado póliza.
Flujo básico	1. La compañía de seguros genera la revisión de la solicitud de cotización. 2. En base a las condiciones enviadas genera la emisión del contrato de seguros.
Flujo alternativo	1. En caso de que se presente inconsistencias en la solicitud de emisión, la compañía de seguros puede generar una emisión errónea.
Pre - condiciones	1. La solicitud de emisión debe contener toda la información del cliente. 2. Debe existir un informe de inspección 3. Debe existir los documentos de vinculación del cliente.
Post - condiciones	Emitida la póliza la compañía de seguros genera él envió de contratos al asesor comercial para que pueda hacer firmar al cliente la copia del contrato.
Puntos de inclusión	La compañía de seguros genera la verificación de datos de la solicitud de emisión.
Puntos de extensión	No aplica.

Nota: Especificación de caso de uso que representa la secuencia de pasos que se ejecuta para generar la emisión de una póliza para el cliente.

Tabla 27.

Caso de uso envió de contratos

Caso de uso	CU6. Envió de contratos de seguro.
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial)
Actor	Act1. Asesor comercial Act2. Aseguradora
Descripción	Cuando la compañía de seguros procede a emitir una póliza envía tres duplicados de contratos al asesor comercial, dichos contratos y denominan original, duplicado y agente.
Flujo básico	1. Una vez emitida la póliza por la compañía de seguros, se genera el envío de contratos al asesor de seguros. 2. El asesor de seguros realiza la verificación de los contratos. 3. El contrato se divide en tres copias, la primera es el contrato al cliente, el segundo es la copia que será firmada por el cliente y el tercero es la copia de agente que se queda con el corredor de seguros.
Flujo alternativo	1. En caso de que exista errores en la emisión, el asesor comercial deberá solicitar el cambio en la emisión de la póliza.
Pre - condiciones	No aplica
Post - condiciones	Con los contratos que envía la compañía de seguros, el asesor comercial comunica al cliente que se encuentra listos para que puedan ser firmados.
Puntos de inclusión	No aplica
Puntos de extensión	• Cada contrato es una copia del original, estos contratos envían la compañía de seguros y son Contrato Original, Copia firmada y Copia agente.

Nota: especificación de caso de uso muestra el detalle del proceso que se genera al momento que la compañía de seguros envía los contratos de la póliza emitida.

Tabla 28.

Caso de uso Entrega de contratos al cliente.

Caso de uso	CU7. Entrega Contratos.
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial)
Actor	Act1. Asesor comercial Act2. Cliente
Descripción	El ejecutivo comercial realiza la entrega de contratos al cliente sobre su póliza de seguros.
Flujo básico	1. Cuando el asesor recibe los contratos enviados por la compañía de seguros, notifica al cliente. 2. Cuando el cliente se reúne con el asesor de seguros, entrega los contratos al cliente para que puedan ser firmados.
Flujo alternativo	1. Exista demora en la entrega de contratos por parte de la compañía de seguros. 2. Los contratos presenten errores que el cliente solicite cambiar.
Pre - condiciones	El asesor comercial debe contar con los contratos para el cliente
Post - condiciones	Con la entrega de contratos el cliente puede proceder con la firma de los mismos y podrán ser enviados a la compañía de seguros.
Puntos de inclusión	El cliente verifica los datos de los contratos que le son entregados.
Puntos de extensión	No aplica

Nota: Especificación de caso de uso para detallar los pasos que se ejecuta en el proceso de entrega de contratos al cliente.

Tabla 29.

Caso de uso Firmar Contrato

Caso de uso	CU8. Firmar contrato por parte del cliente
Fuentes	Hernán Ochoa (Gerente General) Paul Chalem (Jefe Comercial) Carina Tufiño (Asesora Comercial)
Actor	Act1. Cliente Act2. Asesor de Seguros.
Descripción	Cuando el cliente recibe los contratos, los firma para que sean válidos y su póliza disponga de cobertura.
Flujo básico	1. Cual el cliente recibe los contratos por parte del asesor comercial los verifica. 2. Una vez verificados, procede a firmar la copia del original. 3. Una vez firmado es devuelto el contrato firmado al asesor comercial.
Flujo alternativo	1. En caso de que el cliente verifique algún error en los contratos no procede con la firma. 2. Es importante que el cliente sea contactado por el asesor de lo contrario no podrá firmar el contrato.
Pre - condiciones	El cliente debe recibir los contratos por parte del asesor de seguros.
Post - condiciones	Una vez firmado el contrato el cliente tiene cobertura de la póliza
Puntos de inclusión	No aplica
Puntos de extensión	No aplica

Nota: especificación de caso de uso detalla el paso a paso que se desarrolla cuando un cliente firma el contrato.

Tabla 30.

Caso de uso realizar pago de cuotas.

Caso de uso	CU9. Realizar pago de clientes.
Fuentes	Carina Tufiño (Ejecutiva de cuenta) Tatiana Ladines (Ejecutiva de Cobranzas) Luis González (Jefe de Operaciones)
Actor	Act1. Cliente Act2. Ejecutivo de Cobranzas.
Descripción	Cuando un cliente contrata un seguro, tiene la obligación de pagar sus diferentes cuotas
Flujo básico	1. El ejecutivo de cobranzas notifica al cliente que debe pagar las cuotas pendientes. 2. El cliente realiza el pago, puede hacerlo en la compañía de seguros, en el corredor de seguros o directo al banco.
Flujo alternativo	1. El cliente nunca fue notificado por lo cual no genera el pago de las cuotas. 2. Al no pagar su póliza queda inmediatamente anulada por lo tanto no cuenta con cobertura sobre su seguro.
Pre - condiciones	El cliente debe disponer de una póliza de seguros.
Post - condiciones	Con el pago de cuotas por parte del cliente la compañía de seguros envía el detalle de comisiones.
Puntos de inclusión	Cuando un cliente realiza el pago el ejecutivo de cobranzas realiza la verificación de datos para ver si la cuota pagada es la correspondiente y los valores son los adecuados.
Puntos de extensión	No aplica

Nota: especificación de caso de uso el cual detalla los pasos que un cliente sigue para realizar el pago de sus cuotas.

Tabla 31.

Caso de uso Emisión de recibos de pagos.

Caso de uso	CU10. Emitir contrato de seguros.
Fuentes	Tatiana Ladines (Ejecutiva de Cobranzas) Luis González (Jefe de Operaciones)
Actor	Act1. Cliente Act2. Ejecutivo de Cobranzas.
Descripción	El ejecutivo de cobranzas emite un recibo del pago realizado por parte del cliente.
Flujo básico	1. Una vez que el cliente emite el pago, el ejecutivo de cobranzas lo registra en un archivo de Excel. 2. Una vez registrado emite el recibo de pago al cliente. 3. El cliente recepta dicho recibo como constancia de haber realizado el pago de su cuota del seguro.
Flujo alternativo	1. El ejecutivo de cobranzas envía el recibo por medio de correo electrónico. 2. Se genera un recibo de pago del cliente porque pago solo una parte.
Pre - condiciones	1. El cliente debe haber registrado el pago de su cuota del seguro. 2. Debe existir un registro de pago en un sistema informático.
Post - condiciones	No aplica.
Puntos de inclusión	El cliente al recibir su recibo verifica los datos para ver si coinciden con el valor que genero el pago.
Puntos de extensión	No aplica.

Nota: Especificación de caso de uso para la emisión de un recibo de pago de cuotas del cliente.

Tabla 32.

Caso de uso Enviar detalle de pagos de clientes.

Caso de uso	CU11. Enviar detalle de pagos de los clientes a la compañía de seguros.
Fuentes	Tatiana Ladines (Ejecutiva de Cobranzas) Luis González (Jefe de Operaciones)
Actor	Act1. Ejecutivo de Cobranzas. Act2. Aseguradora
Descripción	El ejecutivo de cobranzas envía un listado en el cual se detalla los pagos que realizan los clientes sobre su seguro.
Flujo básico	1. El ejecutivo de cobranzas agrupa todos los pagos realizados en el día por parte de los clientes. 2. Genera un archivo con los datos de pago de cada cliente, el mismo se denomina remisión de pagos, misma que es enviada a la compañía de seguros. 3. La compañía de seguros lo recibe y procesa internamente dicha información.
Flujo alternativo	1. Los pagos enviados a la compañía de seguros presentan errores, 2. El cliente pudo haber realizado pago directo a la compañía de seguros, por lo tanto, no debe constar en el listado de pagos registrados.
Pre - condiciones	1. Debe existir pagos registrados por parte del cliente. 2. El pago que el cliente registra debe ser realizado directo al banco o al corredor de seguros.
Post - condiciones	No aplica.
Puntos de inclusión	La compañía de seguros verifica los detalles de pagos para verificar si se encuentran ingresados de manera correcta.
Puntos de extensión	No aplica

Nota: Especificación de caso de uso, el cual permite detallar paso a paso el del envió de pagos que realiza cada cliente sobre su seguro contratado.

Tabla 33.

Caso de uso Envió detalle de comisiones.

Caso de uso	CU12. Enviar detalle de comisiones por parte de la compañía de seguros.
Fuentes	Tatiana Ladines (Ejecutiva de Cobranzas) Luis González (Jefe de Operaciones)
Actor	Act1. Aseguradora Act2. Jefe de Operaciones
Descripción	La compañía de seguros envía el detalle de comisiones por pagar.
Flujo básico	1. La compañía de seguros envía un detalle de las comisiones que se encuentran listas para ser pagadas. 2. El jefe de operaciones verifica el detalle recibido por la compañía de seguros.
Flujo alternativo	La compañía de seguros no envía el detalle de comisiones por qué no registra pagos por parte del cliente.
Pre - condiciones	El jefe operativo debe solicitar el detalle a la compañía de seguros.
Post - condiciones	Con el detalle de comisiones el jefe operativo puede realizar la facturación a la compañía de seguros.
Puntos de inclusión	El jefe operativo realiza la revisión del detalle de comisiones con el detalle de producción emitida y pagada.
Puntos de extensión	No aplica.

Nota: Especificación de caso de uso permite identificar el proceso paso a paso para el envío del detalle de comisiones por parte del a compañía de seguros, con este detalle se puede proceder a generar la factura a dicha compañía.

Tabla 34.

Caso de uso Emisión de factura.

Caso de uso	CU13. Enviar detalle de pagos de los clientes a la compañía de seguros.
Fuentes	Tatiana Ladines (Ejecutiva de Cobranzas) Luis González (Jefe de Operaciones)
Actor	Act1. Jefe Operativo Act2. Aseguradora.
Descripción	Jefe operativo emite una factura con un detalle sobre el cual se cobra comisiones a la compañía de seguros.
Flujo básico	1. Cuando el jefe operativo realiza la revisión de comisiones genera una pre - factura de dicho detalle. 2. Con la pre factura generada se procede con la emisión de la factura. 3. Dicha factura es entregada a la compañía de seguros.
Flujo alternativo	1. Puede existir más de una pre - factura que sea incluida en una factura. 2. Puede existir más de un pre – factura para cada factura, siempre y cuando la pre - factura corresponda a la compañía de seguros que se emitirá la factura.
Pre - condiciones	Se debe contar con el detalle de comisiones para poder generar la pre - factura.
Post - condiciones	Emitida la factura a la compañía de seguros, dicha compañía debe proceder a realizar el pago de comisiones que se detalla en la factura emitida.
Puntos de inclusión	La compañía de seguros verifica el detalle de la factura para validar que sea lo correspondiente para proceder a realizar el pago de comisiones.
Puntos de extensión	No aplica.

Nota: Especificación de caso de uso, permite definir el paso a paso que se realiza para poder emitir una factura a una compañía de seguros.

Tabla 35.

Caso de uso Pagos de Facturas a bróker.

Caso de uso	CU14. Realizar pago de factura a bróker.
Fuentes	Tatiana Ladines (Ejecutiva de Cobranzas) Luis González (Jefe de Operaciones)
Actor	Act1. Jefe Operativo Act2. Aseguradora.
Descripción	La compañía de seguros paga la factura emitida por el corredor de seguros.
Flujo básico	1. La compañía de seguros al recibir la factura emitida por el corredor de seguros, emite un cheque de pago. 2. Dicho cheque es enviado al corredor de seguros. 3. El jefe operativo recepta dicho pago y lo registra en el sistema.
Flujo alternativo	1. La compañía de seguros no genera el pago completo de la factura. 2. El detalle de la factura no corresponde por lo que devuelven la factura para que sea cambiado.
Pre - condiciones	Debe existir una factura emitida para que pueda ser pagada por la compañía de seguros.
Post - condiciones	Con el pago de la factura se puede liberar la comisión a los asesores comerciales del corredor de seguros.
Puntos de inclusión	No aplica
Puntos de extensión	No aplica

Nota: Especificación de caso de uso para poder detallar el flujo de proceso que se ejecuta cuando una compañía de seguros realiza el pago de la factura al corredor de seguros.

Tabla 36.

Caso de uso envió detalle de pago para asesores.

Caso de uso	CU15. Enviar detalle de pago para asesores comerciales
Fuentes	Luis González (Jefe de Operaciones)
Actor	Act1. Jefe Operativo Act2. Talento humano.
Descripción	Cuando la compañía de seguros realiza el pago de una factura, el jefe de operaciones divide el detalle por asesor comercial y lo envía a nomina para que pueda realizar el pago correspondiente.
Flujo básico	1. El jefe de operaciones, revisa el pago generado por la compañía de seguros. 2. De la revisión genera un detalle por asesor comercial. 3. Cada detalle envía a nomina con los valores correspondientes que deben ser pagados a cada asesor. 4. Nomina recepta dicho pago para poder incluir en pagos mensuales.
Flujo alternativo	En caso de que no exista pagos por parte de la compañía de seguros no se podrá enviar el detalle de comisiones a nómina.
Pre - condiciones	Para enviar detalle, la factura debe ser pagada por la compañía de seguros.
Post - condiciones	No aplica.
Puntos de inclusión	No aplica
Puntos de extensión	No aplica

Nota: Especificación de caso de uso detalla el paso a paso para poder enviar el detalle de pagos de comisiones para cada asesor de seguros al área de nómina o talento humano.

3.05 Casos de uso de realización.

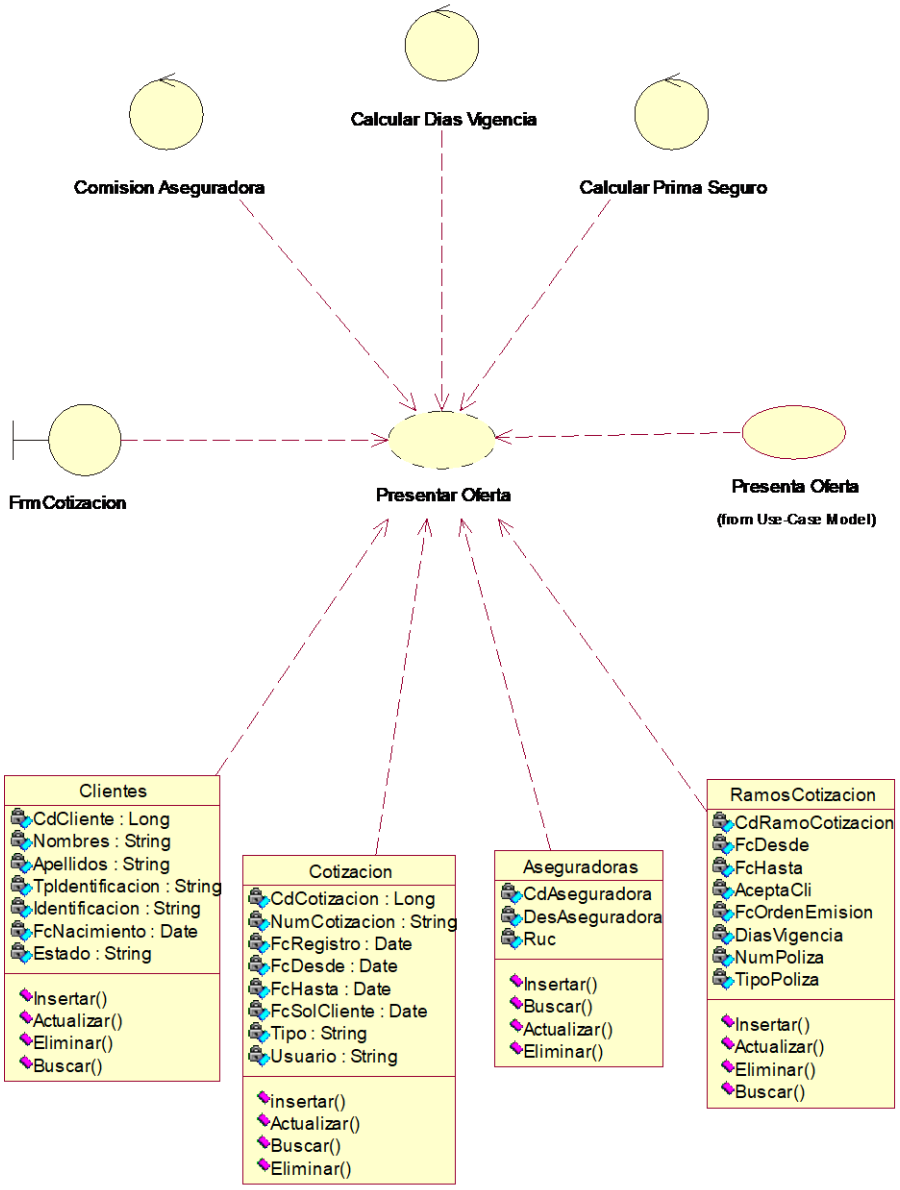


Figura 10.Caso de uso de realización, Presentar Oferta.

Se presenta los controles que se presenta para la interfaz de Cotización, adicional a esto se presenta las clases que se utilizara para dicho proceso.

Tabla 37.

Caso de uso de realización Presentar Oferta.

Caso de Uso	Presentar Oferta
Identificador	RCU 001.
Curso Típico De Eventos	
Usuario	**Sistema**
El usuario ingresa a la pantalla de cotización.	El sistema presenta una pantalla con una tabla donde se presenta las cotizaciones pendientes, adicional dos botones uno para ejecutar dicha consulta y otro para ingresar una nueva cotización.
El usuario presiona el botón nuevo en el sistema	El sistema presenta un panel de clientes, en caso de no existir presenta un botón para registrar un nuevo cliente.
El usuario busca a su cliente en el sistema y lo selecciona.	El sistema redirecciona a la pantalla de nueva cotización.
El usuario selecciona la aseguradora y el ramo	El sistema muestra el porcentaje de comisión para la aseguradora y ramo seleccionados.
El usuario ingresa las vigencias del seguro y fecha de solicitud del cliente	El sistema habilita el botón para agregar objetos asegurados.
El usuario da clic sobre el botón para agregar objetos asegurados	El sistema solicita los valores del seguro y sus especificaciones.
Una vez ingresado los objetos asegurados, se habilita el botón enviar cotización	El sistema presenta una pantalla donde le permite enviar un email al cliente con los datos de su cotización.
Flujo alternativo	
En caso de que no se ingrese las fechas de vigencia del seguro, no se permitirá registrar los objetos asegurados, de la misma manera si no se ingresa un ramo no podrá registrar la cotización ni enviar al cliente.	

Nota: Especificación de caso de uso de realización para presentar una oferta, detalla el flujo de eventos ideal para que un usuario pueda enviar una cotización a un cliente.

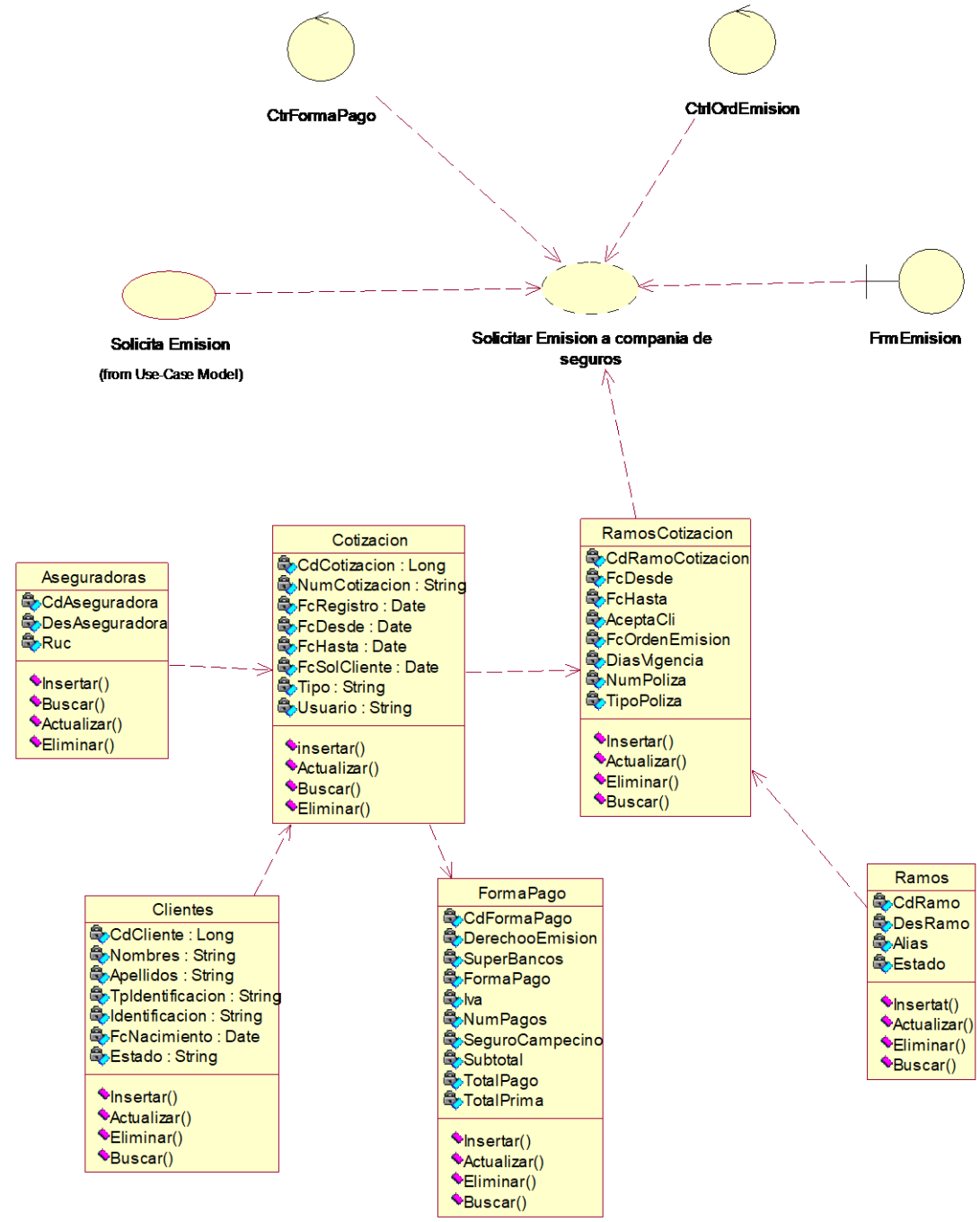


Figura 11. Caso de uso de realización, Solicitud de Emisión.

Detalla los componentes que requiere el caso de uso para que se pueda generar la solicitud de emisión de una póliza en el sistema.

Tabla 38.

Caso de uso de realización Solicitar Emisión

Caso de Uso	Solicitar Emisión
Identificador	RCU 002.
Curso Típico De Eventos	
Usuario	**Sistema**
El usuario ingresa a la pantalla de emisión	El sistema solicita presenta una interfaz para generar una emisión.
El usuario ingresa los datos de la cotización presentada al cliente	El sistema verifica que la cotización se encuentre registrada.
El usuario ingresa a la cotización	El sistema presenta una interfaz similar a la de una cotización nueva, adicionando los campos de fecha de emisión, numero de póliza y factura.
El usuario registra la forma de pago del cliente	El sistema habilita la opción de solicitud de emisión a la compañía de seguros.
El usuario presiona el botón solicitar emisión.	Se presenta un panel de correo electrónico con los datos del cliente y el valor cotizado, dirigido al contacto en la compañía de seguros.
Flujo alternativo	
En caso de que no se registre una forma de pago el sistema no habilitara la opción de envió de correo electrónico.	

Nota: Especificación de diagrama de realización del proceso de solicitud de emisión de una póliza, por medio de correo electrónico.

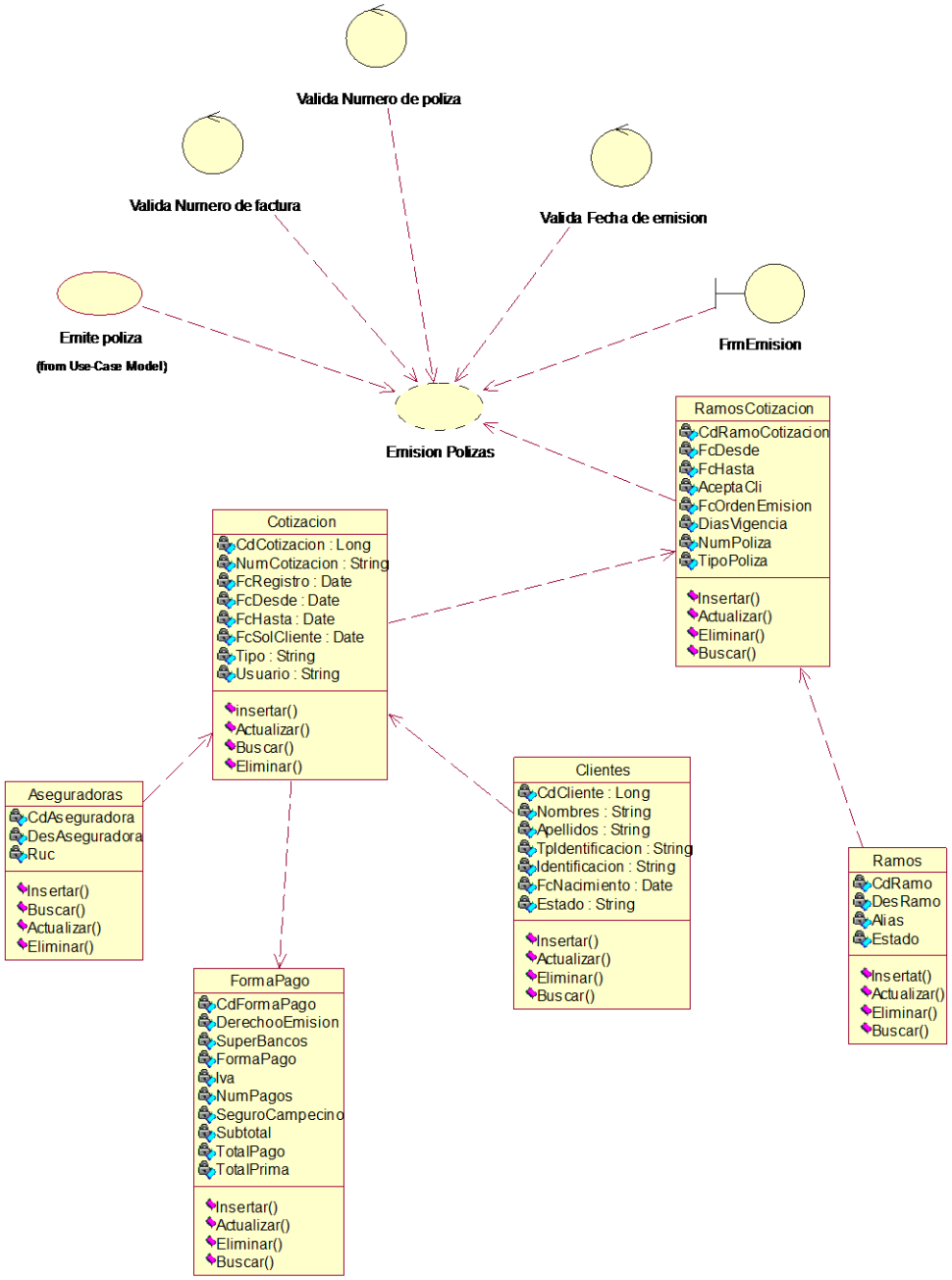


Figura 12. Caso de uso de realización, Emisión Póliza.

Se define los controles y la interfaz que será aplicada en el proceso de emisión.

Tabla 39.

Especificación de caso de uso de realización emisión de pólizas.

Caso de Uso	Emisión de pólizas
Identificador	RCU 003.
Curso Típico De Eventos	
Usuario	**Sistema**
El usuario al recibir los documentos de la póliza emitida ingresa al sistema en la pantalla de emisiones pendiente.	El sistema presenta una pantalla donde se encuentran todas las pólizas pendientes de emisión, es decir que se generó una solicitud de emisión.
El usuario buscara el registro a emitir por los parámetros de consulta	El sistema presentará el registro pendiente.
El usuario ingresa al registro.	El sistema presenta la interfaz de emisión, donde solicita el número de factura, numero de póliza y fecha de emisión.
El usuario con los datos de factura y numero de póliza procede a ingresarlos.	El sistema presenta un mensaje indicando que la emisión fue exitosa.
Flujo alternativo	
En caso de que no exista comisión para el ramo y para la compañía de seguros seleccionada, no se podrá continuar con la emisión. En caso de que no se cuente con comisión para el subagente asignado al cliente, no se podrá continuar con la emisión.	

Nota: Especificación de caso de uso de realización, se detalla los pasos a seguir para que un usuario pueda registrar la emisión de una póliza con los datos proporcionados por la compañía de seguros, de igual manera se presenta un flujo de posibles contratiempos que se pueden presentar en el proceso.

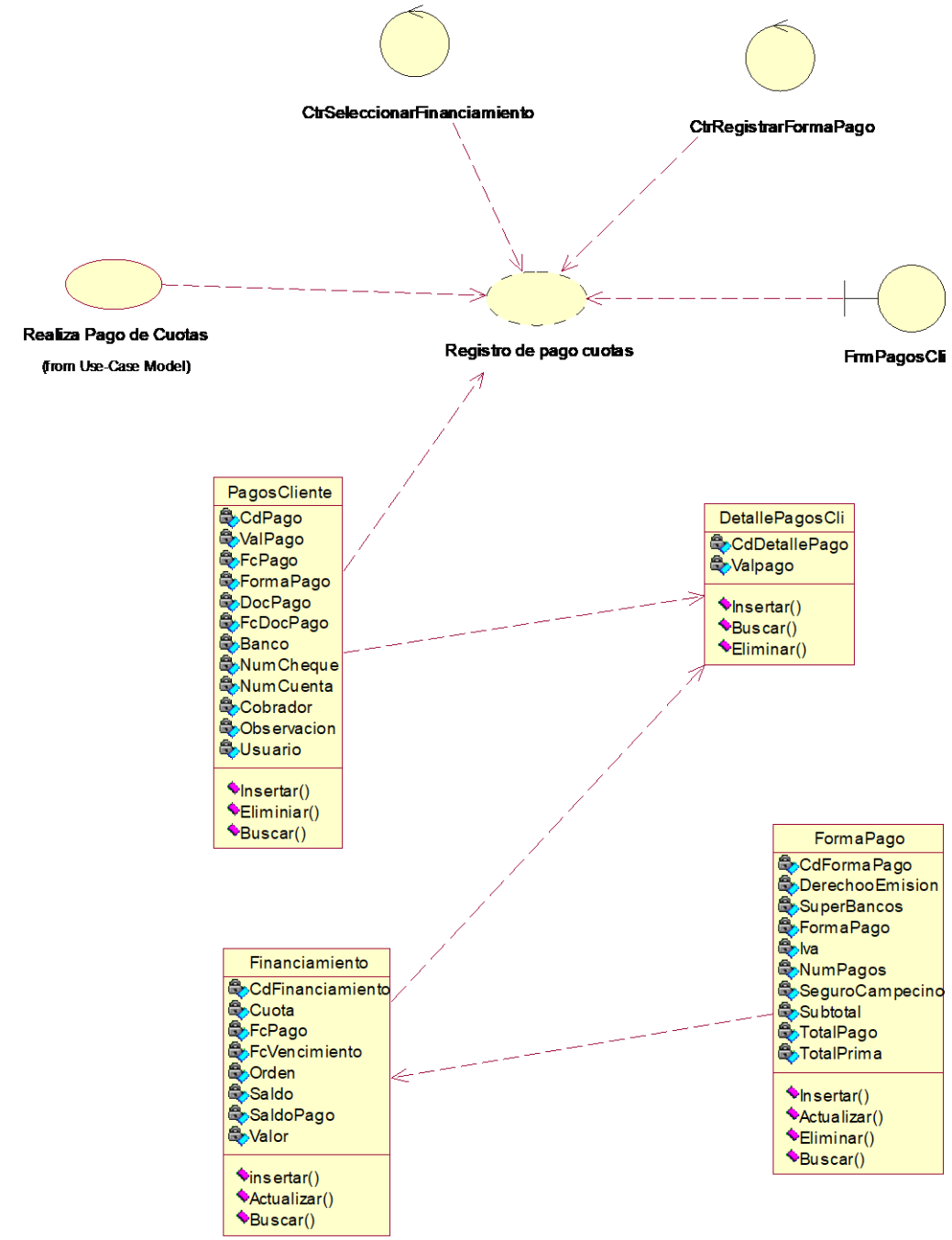


Figura 13. Caso de uso de realización, Registro pagos cuotas cliente.

Se detalla las clases y controles que se utilizara para el proceso de registro de pagos realizados por el cliente.

Tabla 40.

Especificación de caso de uso de realización Registro pago de cuotas

Caso de Uso	Realiza Pago Cuotas
Identificador	RCU 004.
Curso Típico De Eventos	
Usuario	**Sistema**
El usuario al recibir un pago por parte del cliente ingresa al sistema.	El sistema presenta una interfaz en la cual se presenta el parámetro de búsqueda de la cuota que se está registrando el pago.
El usuario ingresa los datos de la factura en los parámetros de búsqueda y presiona el botón buscar	El sistema presenta la información del cliente, con el detalle de la cuota o cuotas por pagar.
El usuario ingresa al registro que desea aplicar el pago	El sistema presenta una interfaz donde se presenta las cuotas del cliente, mismas que para ser registradas se debe seleccionar cual es la que se aplicara el pago.
El usuario selecciona la o las cuotas a pagar y presiona el botón pago.	El sistema muestra un panel donde se debe registrar el monto a pagar y los detalles del comprobante de pago.
El usuario una vez ingresado los valores a pagar presiona el botón registrar pago.	El sistema presenta un recibo el cual es entregado al cliente.
Flujo alternativo	
En caso de que los pagos se encuentren ya registrados no se presentará la información en la pantalla de consulta.	

Nota: Especificacion de diagrama de casos de uso detalla el proceso que un usuario sigue en el sistema para poder registrar el pago de un cliente sobre su seguro.

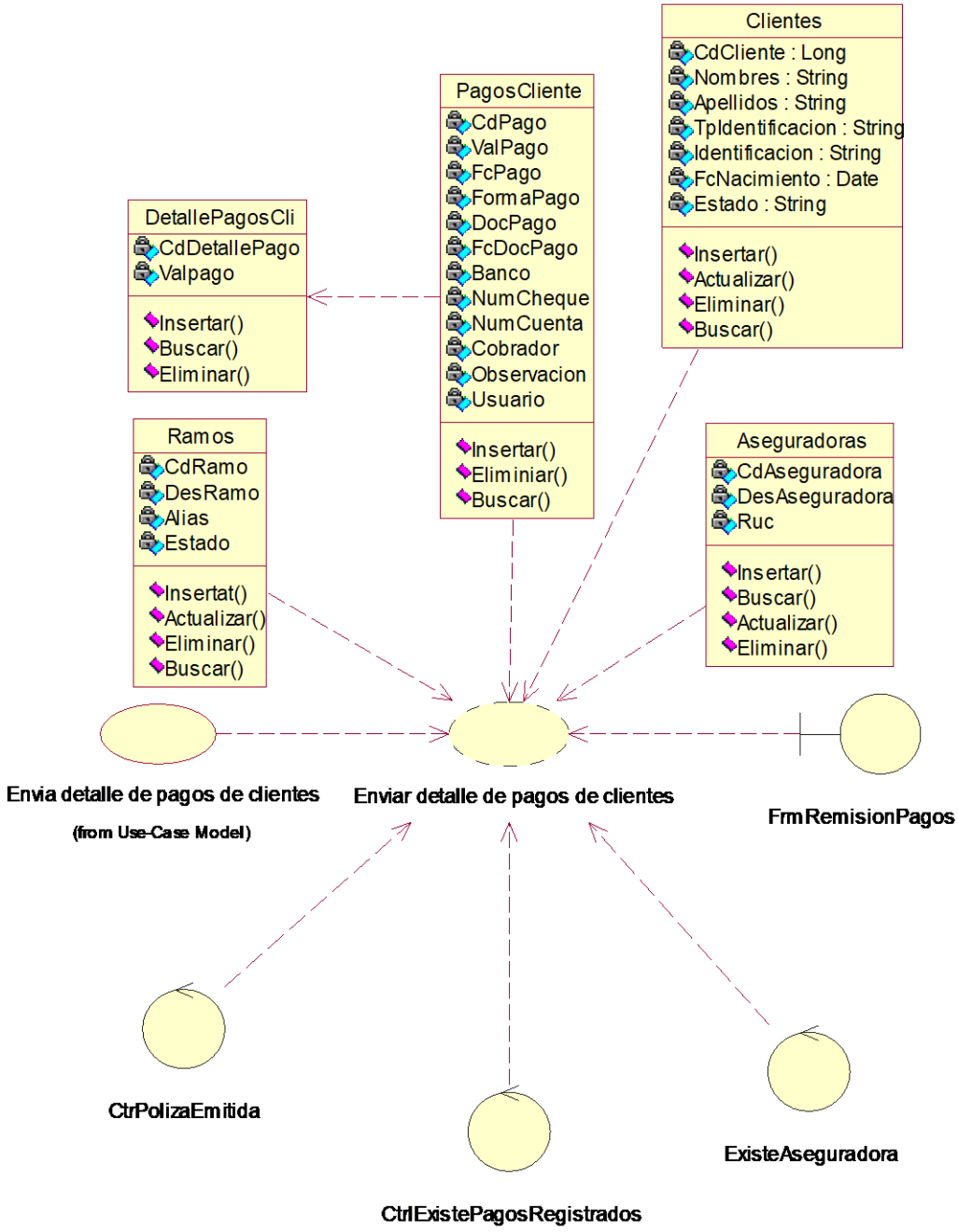


Figura 14. Caso de uso de realización, Envió Detalle de Pagos Clientes.

Presenta los controles y componentes necesarios para poder generar el detalle de pagos realizados por un cliente.

Tabla 41.

Especificación de caso de uso de realización Envió de pagos Clientes

Caso de Uso	Envía detalle de pagos clientes
Identificador	RCU 005.
Curso Típico De Eventos	
Usuario	**Sistema**
El usuario ingresar al sistema al módulo de cobranzas.	El sistema presenta la opción de remisión de pagos.
El usuario al ingresar debe seleccionar la compañía de seguros.	El sistema mostrará los registros cobrados para dicha compañía de seguros.
El usuario selecciona los registros presentados y que serán enviados a la compañía de seguros.	El sistema calcula el valor total recaudado de los registros seleccionados por el usuario.
El usuario presiona el botón generar remisión	El sistema presenta un reporte con el detalle de las cuotas de los clientes y que será enviado a la compañía de seguros.
El usuario puede enviar el reporte por correo electrónico al contacto de la compañía de seguros o vía carta.	El sistema muestra los datos del contacto para que se envié el correo electrónico.
Flujo alternativo	
En caso de que no se encuentre registrados pagos no se presenta en el listado para seleccionar y generar el reporte de remisión.	

Nota: Especificación de caso de uso detalla el flujo de eventos para el diagrama de realización, se detalle el flujo ideal y el flujo alternativo de los posibles eventos a presentarse

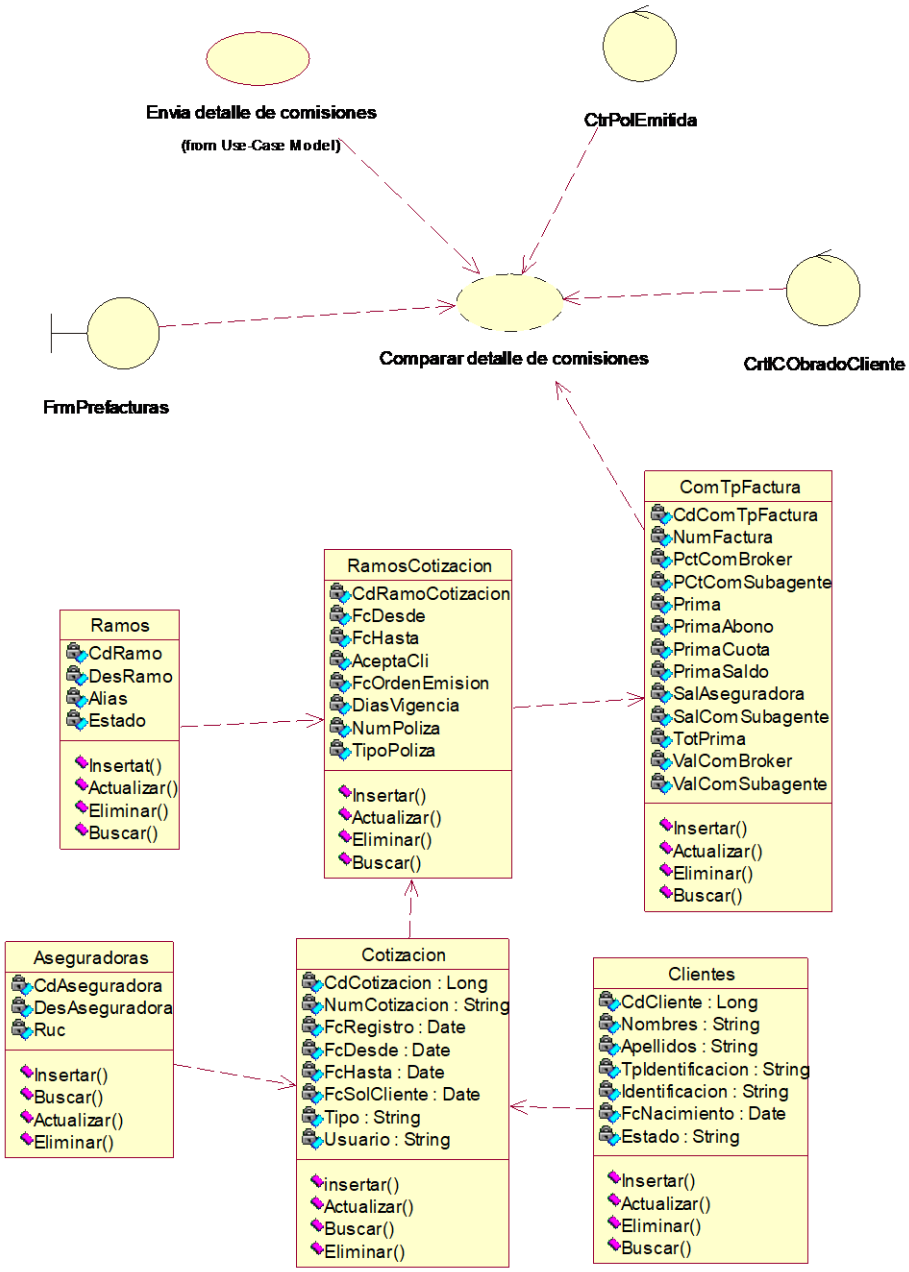


Figura 15. Caso de Uso de Realización, Envió de detalle de Comisiones

Permite identificar el proceso que se realiza para comprar el detalle proporcionado por la aseguradora con el detalle del sistema.

Tabla 42. *Caso de uso realización envío de detalle de comisiones.*

Especificación de caso de uso de realización envío de detalle de comisiones.

Caso de Uso	Envió detalle de comisiones.
Identificador	RCU 006.
Curso Típico De Eventos	
Usuario	**Sistema**
El usuario ingresa al sistema en el módulo de prefecturas	El sistema presenta una interfaz donde se puede evidenciar un filtro por aseguradora
El usuario selecciona una aseguradora y presiona el botón buscar	El sistema muestra un detalle de los registros de cuotas cobradas al cliente donde se libera comisión.
El usuario compara el detalle enviado por la compañía de seguros y selecciona los registros que se encuentran en el sistema.	El sistema realiza un cálculo de los registros seleccionados en base al porcentaje de comisión parametrizado.
El usuario verifica que los datos sean iguales o que no varíen por grandes cantidades.	El sistema presenta los datos calculados por el porcentaje parametrizado.
El usuario presiona el botón generar prefectura el cual genera un detalle de los registros seleccionados.	El sistema presenta un reporte con los datos de los registros seleccionados y con los valores totales.
Flujo alternativo	
En caso de que una póliza no sea registrada su pago no se podrá comprar con el registro de la compañía de seguros ya que el sistema no me recuperara dicha información.	

Nota: Especificación de diagrama de realización para el proceso de comparación de detalle de comisiones proporcionado por la compañía de seguros.

3.06 Diagramas de secuencia.

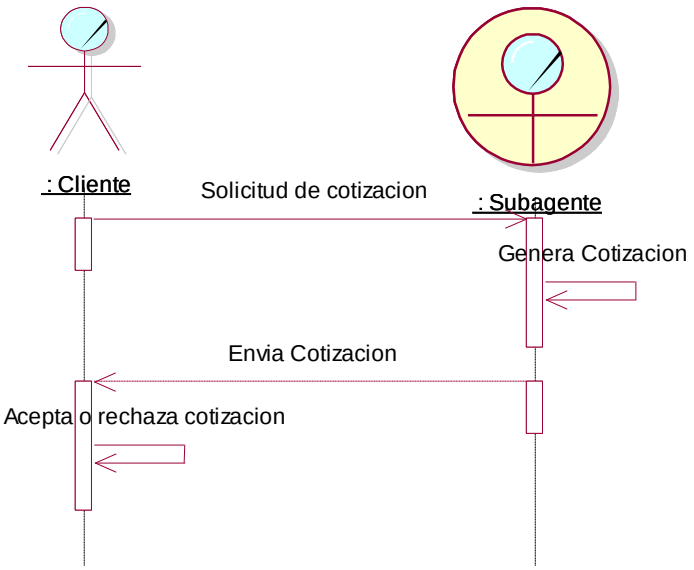


Figura 16. Diagrama de secuencia proceso de cotización

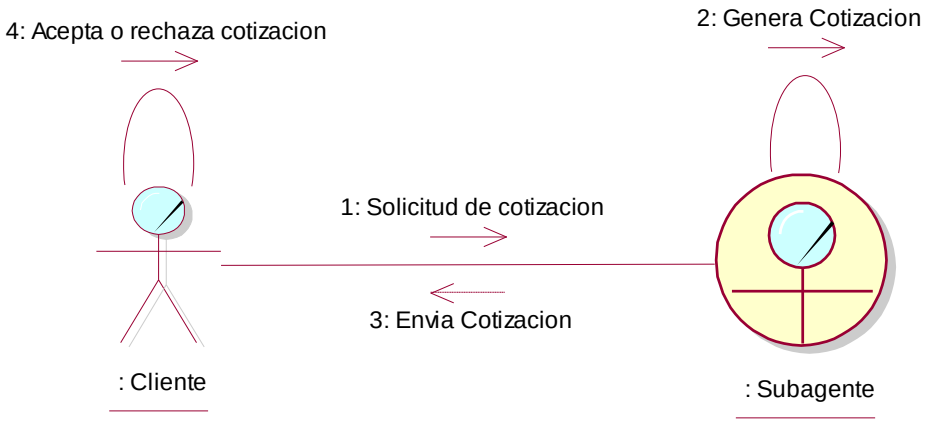


Figura 17. Diagrama de Colaboración proceso de cotización.

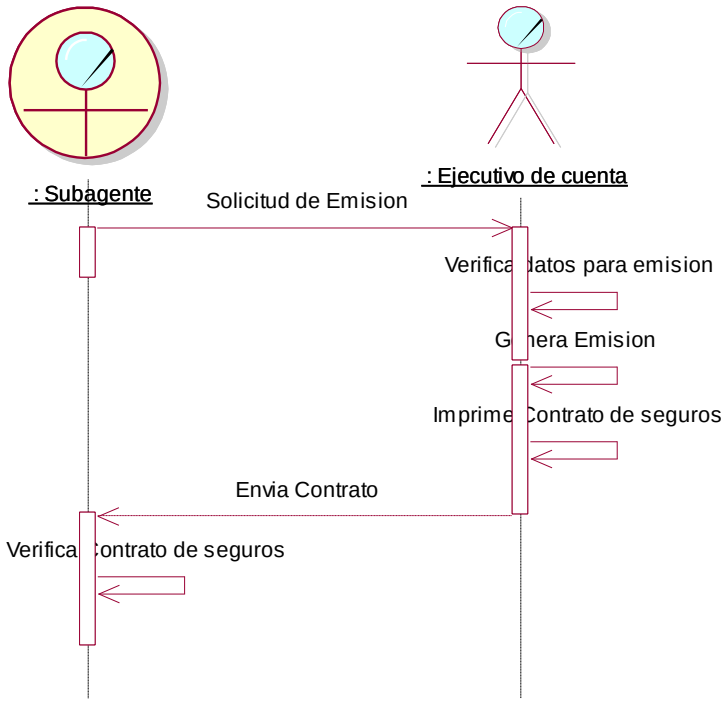


Figura 18. Diagrama de secuencia proceso de emisión.

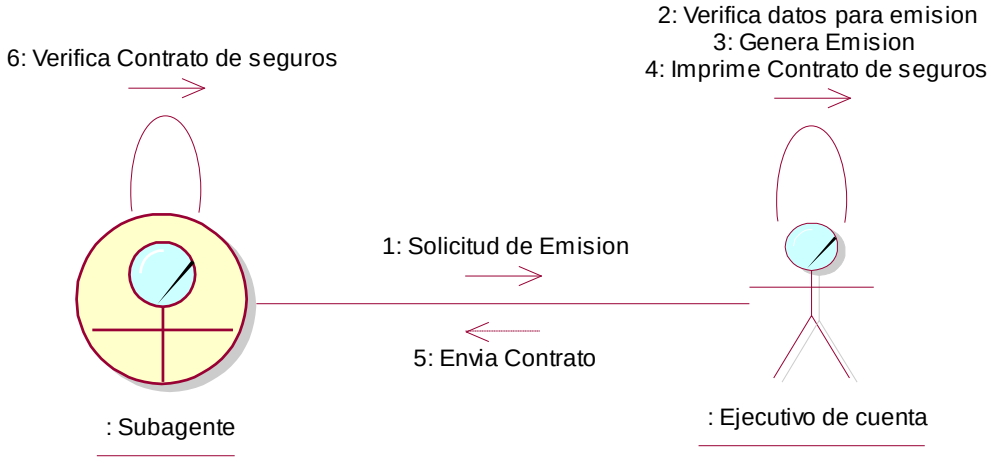


Figura 19. Diagrama de colaboración proceso de emisión

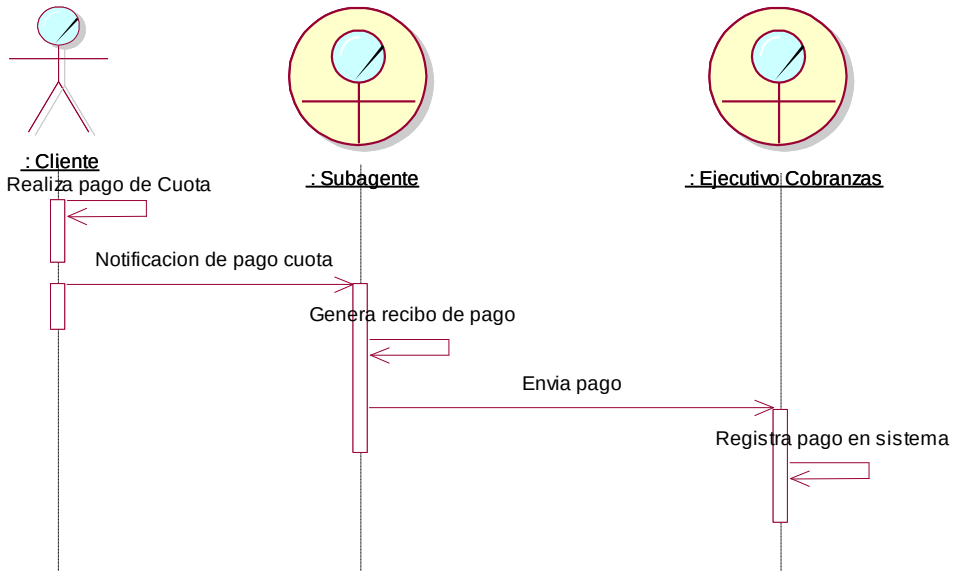


Figura 20. Diagrama de secuencia proceso de cobranzas al cliente.

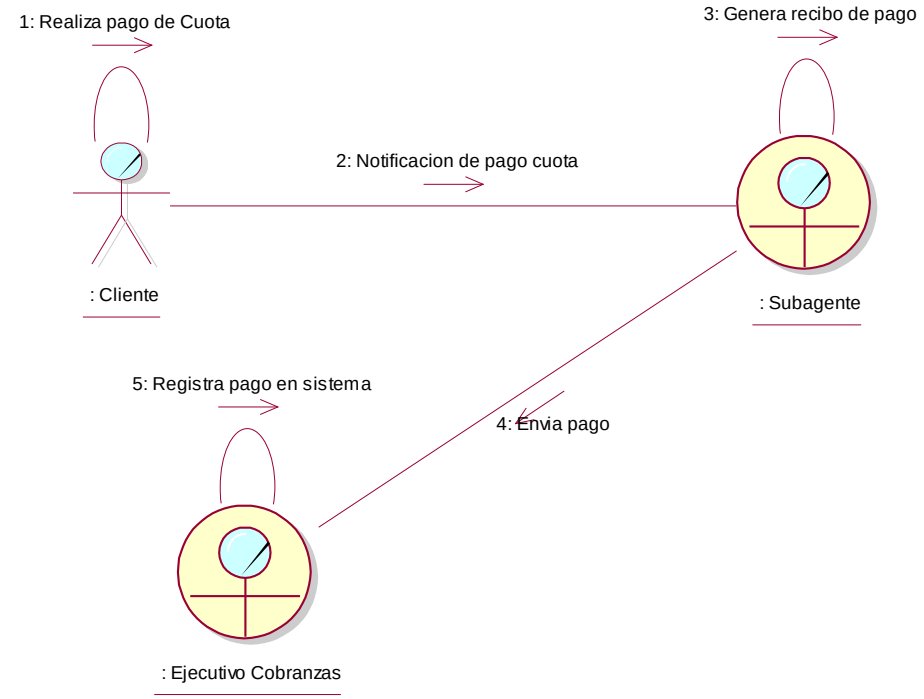


Figura 21. Diagrama de Colaboración proceso de cobranzas a cliente.

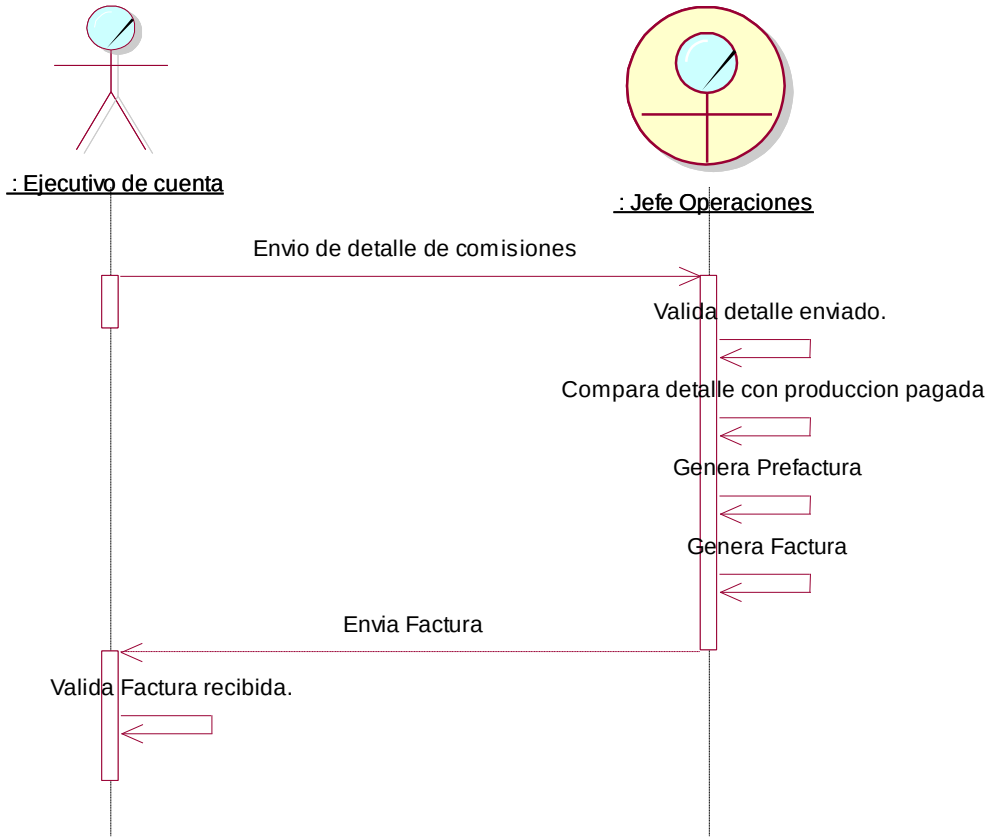


Figura 22. Diagrama de Secuencia proceso de facturación.

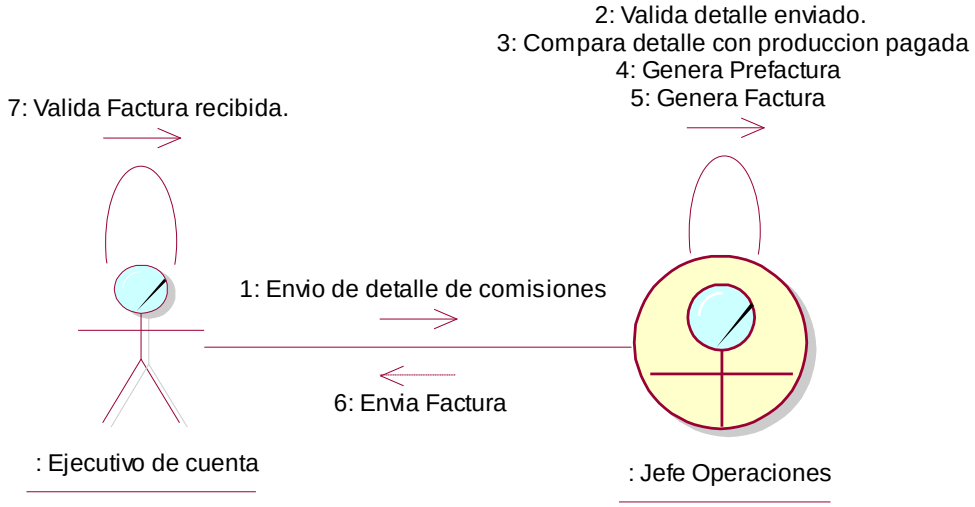


Figura 23. Diagrama de Colaboración proceso de facturación.

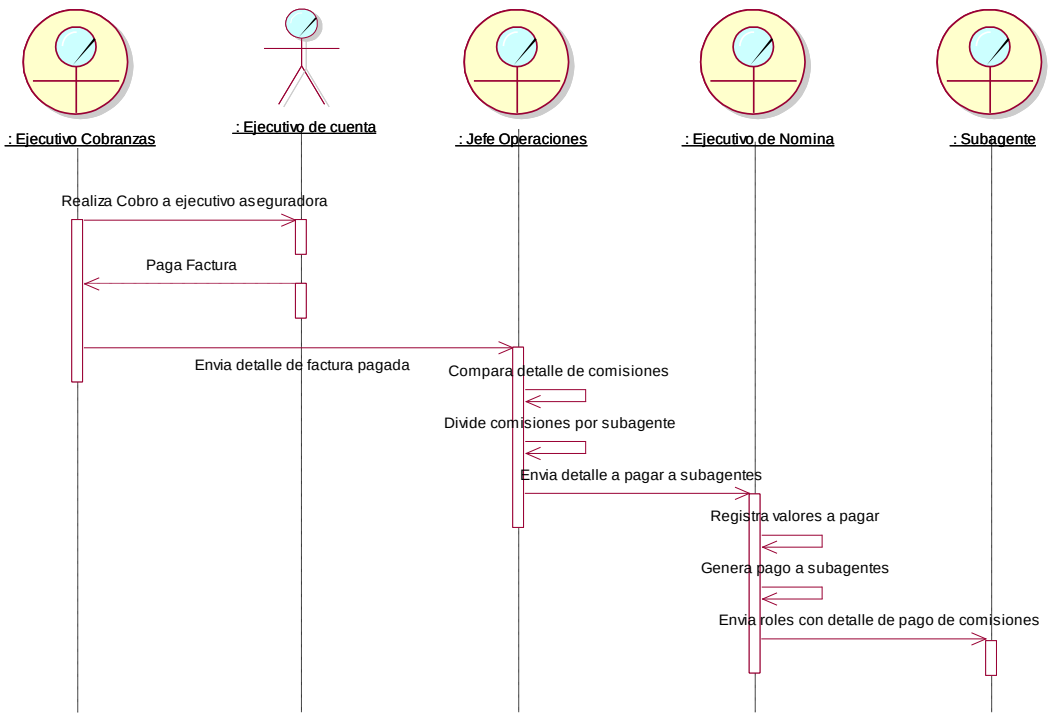


Figura 24. Diagrama de secuencia proceso de pago comisiones.

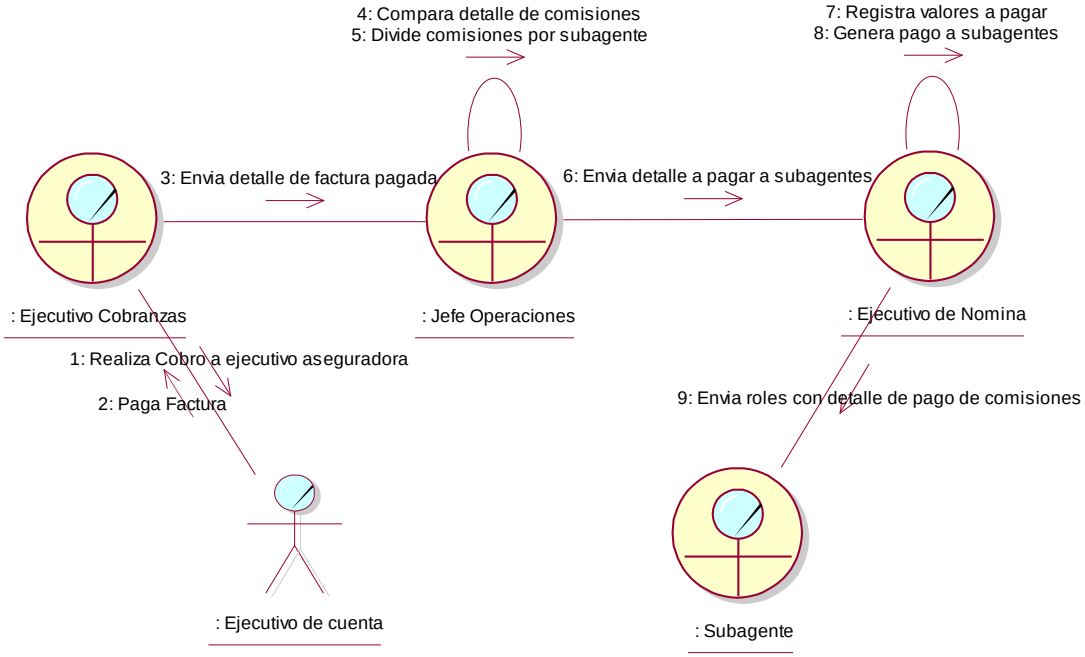


Figura 25. Diagrama de Colaboración proceso pago de comisiones.

CAPÍTULO IV

4. Análisis de alternativas.

4.01 Matriz de análisis de alternativas.

Es un análisis de las alternativas que se puede presentar en el desarrollo del proyecto la cual nos permitirá identificar el objetivo principal del proyecto, de igual manera poder identificar los medios posibles que permitan que el proyecto sea alcanzado con los recursos más adecuados.

Tabla 43.

Matriz de Análisis de Alternativas

Objetivos	Impacto	Fact. Técnica	Fact. Financiera	Fact. Social	Fact. Política	Total	Categoría
Capacitación del personal sobre un sistema que permita agilizar procesos	**4**	**5**	**4**	**3**	**4**	**20**	**Medio Alta**
Implementar una herramienta informática que gestione el flujo de procesos.	**3**	**4**	**4**	**5**	**3**	**19**	**Medio Alta**
Generar un repositorio de gestión masiva de clientes por medio de correo electrónico	**3**	**3**	**2**	**2**	**1**	**11**	**Media**
Definir cargos y áreas de trabajo en base al perfil profesional	**4**	**3**	**5**	**4**	**3**	**19**	**Medio Alta**
Medir carga de trabajo por colaborador.	**5**	**4**	**5**	**5**	**5**	**24**	**Alta**
TOTAL	**19**	**19**	**20**	**19**	**16**	**93**	

Nota: Matriz de análisis de alternativas, permite evidenciar la viabilidad del proyecto para y que alternativas son factibles en su ejecución.

4.02 Matriz de análisis de impactos de los objetivos.

Con la matriz de análisis de impactos se puede evidenciar que impacto tiene cada objetivo que se ha plateado en el desarrollo del proyecto, definiendo la factibilidad de cada uno de los objetivos.

Tabla 44.

Matriz de análisis de impactos.

Objetivo	Factibilidad de lograrse	Impacto en genero	Impacto ambiental	Relevancia	Sostenibilidad
Capacitación del personal sobre un sistema que permita agilizar procesos	Facilita el manejo del sistema a implementarse	La capacitación será proporcionada a todos los usuarios	En la capacitación se utilizará herramientas tecnológicas.	Con una capacitación a los usuarios, dichos usuarios podrán hacer uso del sistema	Mejorar el conocimiento de los usuarios
Implementar una herramienta informática que gestione el flujo de procesos.	Cuenta con el apoyo por parte de la empresa	Participación de hombres y mujeres por igual.	Contribuye a proteger el entorno ambiental	Beneficia al área con mayor vulnerabilidad	Fortalece a la organización en general
Generar un repositorio de gestión masiva de clientes por medio de correo electrónico	Existe tecnología que permite realizar dicha gestión	Podrá ser utilizado por todos los usuarios tanto hombres como mujeres	Con esta herramienta se podrá fortalecer la educación ambiental	Es una prioridad para los usuarios que operaran el sistema	Mejora la participación de los usuarios.
Definir cargos y áreas de trabajo en base al perfil profesional	Es aceptable y conveniente para la empresa y cada usuario.	Participación de hombres y mujeres es por igual.		Responde a las expectativas de los usuarios y la empresa	Fortalece el rendimiento de la empresa mejorando sus ingresos
Medir carga de trabajo por colaborador.	Cuenta con soporte por parte de la empresa.	Fortalece la aplicación de los derechos de la mujer	Mejora el entorno cultural en cada área de trabajo	Los beneficios son idóneos para cada usuario.	Se puede conseguir mejor orden y manejo a futuro.

Nota: Matriz de Análisis de impactos de los objetivos nos permite identificar qué impacto tiene cada objetivo en los ámbitos detallados.

4.03 Diagrama de estrategias.

Representa las estrategias que serán utilizadas en base a los objetivos planteados, con dichas estrategias se requiere cumplir con cada uno de los objetivos de esta manera se podrá realizar el proyecto de manera adecuada.

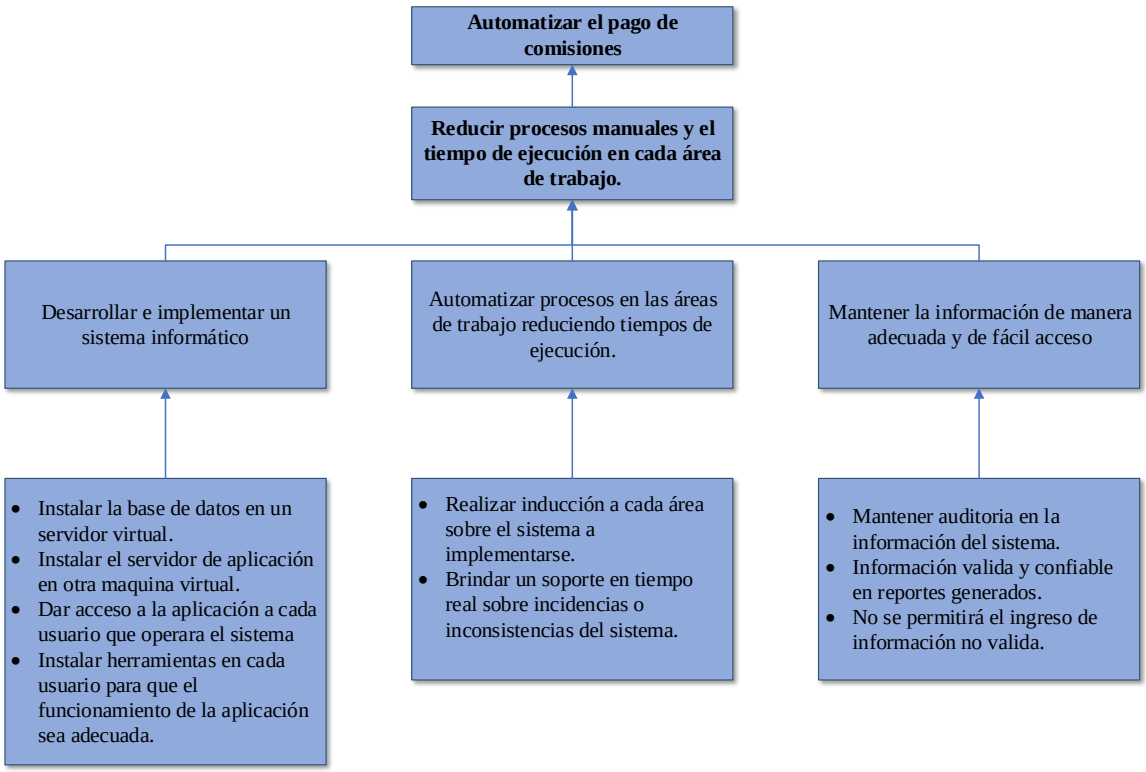


Figura 26. Diagrama de estrategias.

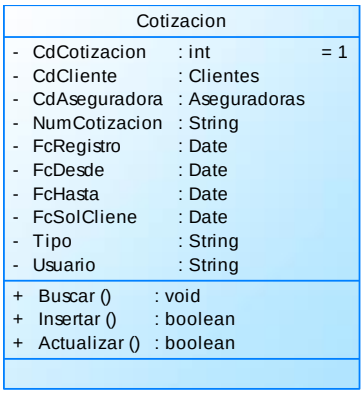
Presenta las estrategias que se aplicara en el desarrollo del proyecto, con el fin de poder cumplir con todos los objetivos planteados en el inicio del proyecto.

4.03.01 Diseño de Clases.

El diseño de clases presenta una descripción lógica de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.

Clase

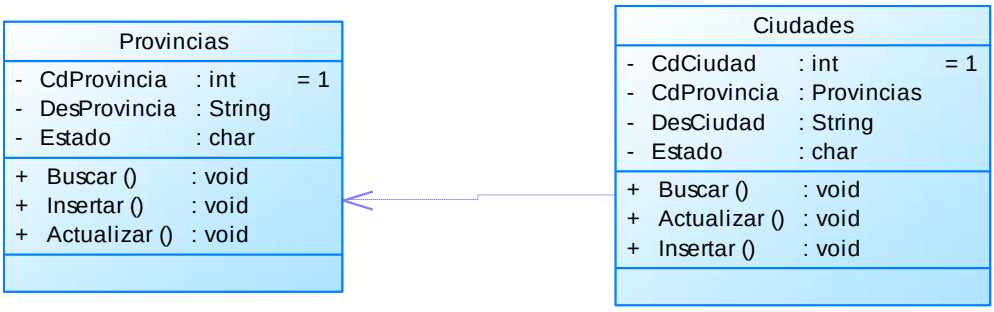
Se la representa con un rectángulo dividido en tres partes, la parte superior presenta el nombre de la clase, la parte intermedia presenta los atributos y el tipo de dato que tiene un objeto y la parte inferior representa los métodos que utiliza dicha clase.



Relaciones entre clases

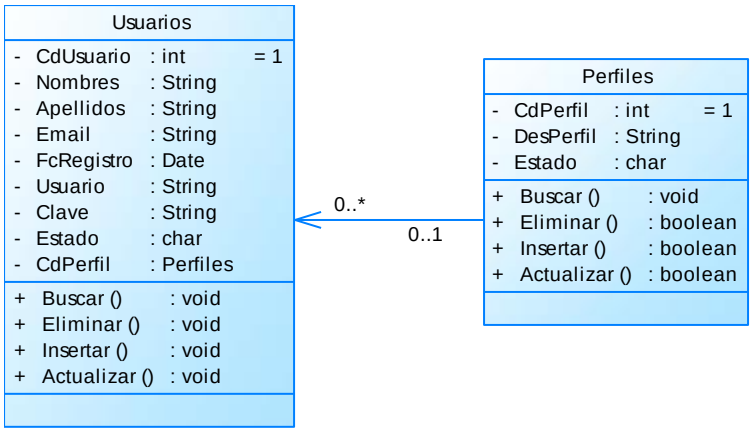
Relación de dependencia.

Es la relación que se establece entre una clase A con una clase B.



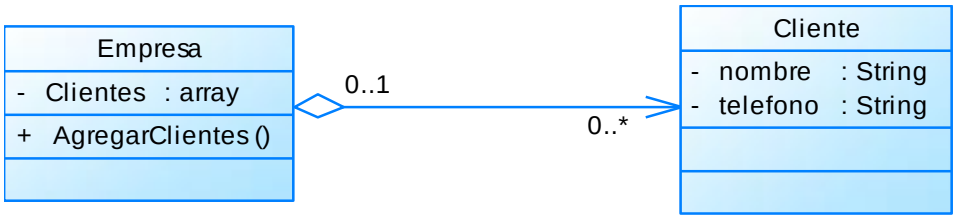
Relación de Asociación.

Es un tipo de relación la cual, a diferencia de la anterior, existe una dependencia entre las clases que es permanente en el tiempo, y tiene una implicación directa en la estructura estática de la clase.



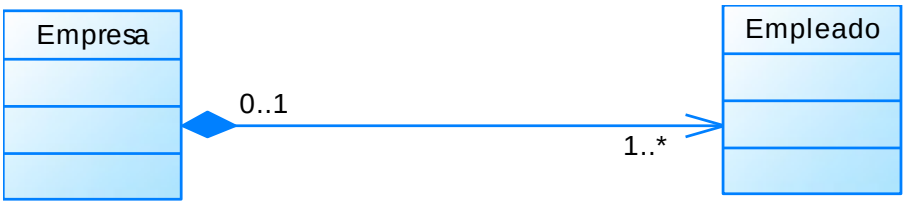
Relación de Agregación.

Es un tipo de relación en la que se puede identificar que existe un concepto que está formado por partes, se la conoce como relación todo – partes.



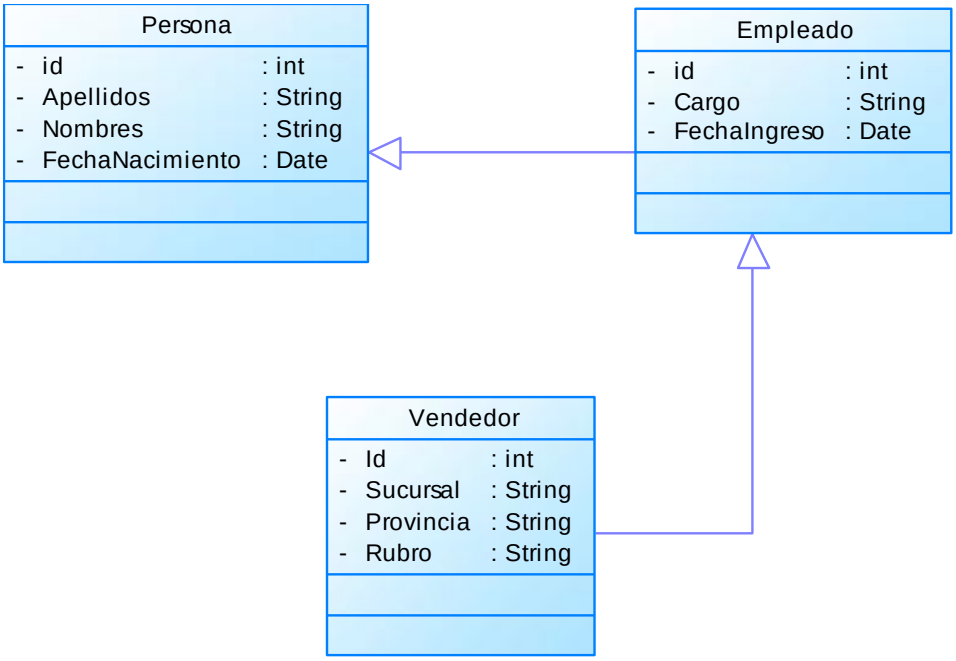
Relación de Composición.

Una relación de composición es una agregación, en la que la parte, pertenece exclusivamente a su correspondiente todo. Es decir, no se puede compartir una parte por varios todos.



Relación de Herencia.

La herencia, es una relación en la que existe un elemento generalizador y uno especializado, razón por la cual se le conoce también como Generalización-Especialización. La señal más inminente de que se está frente a una relación de Herencia, entre la clase A y la Clase B, es cuando se puede decir naturalmente que la clase A es un B.



Multiplicidad.

La multiplicidad es el número de instancias que tiene una clase en relación con otra clase pueden ser de varias maneras.

Tabla 45.

Multiplicidad.

Multiplicidad	Significado
1	Uno y solo uno
0..1	Cero o uno
N..M	Desde N hasta M
*	Cero o varios
0.. *	Cero o varios
1.. *	Uno a varios

Nota: representación de multiplicidad para diseño de clases.

4.03.02 Diagrama de clases.

Representa un diagrama estructurado que representa la estructura del sistema detallando las clases del sistema, sus atributos, métodos y las distintas relaciones que existe entre ellas.

Revisar Anexo D. Diagrama de Clases

4.03.03 Modelo lógico - físico.

El modelo físico representa un modelo de datos de tipo relacional, como tablas, columnas, claves primarias y claves foráneas, y la relación que existe entre ellas. Revisar Anexo E Modelo Físico.

El modelo lógico representa un modelo no especifico que permite describir los aspectos relacionados de una organización, con el fin de poder recopilar datos y sus relaciones entre sí.

Revisar Anexo F. Modelo Lógico.

4.03.04 Diagrama de componentes.

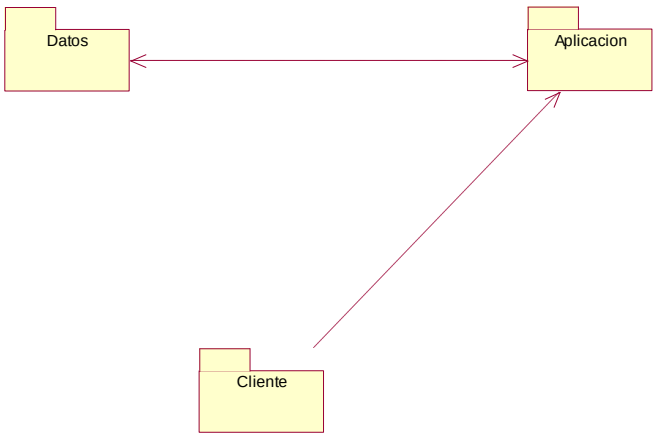


Figura 27. Diagrama de componentes General.

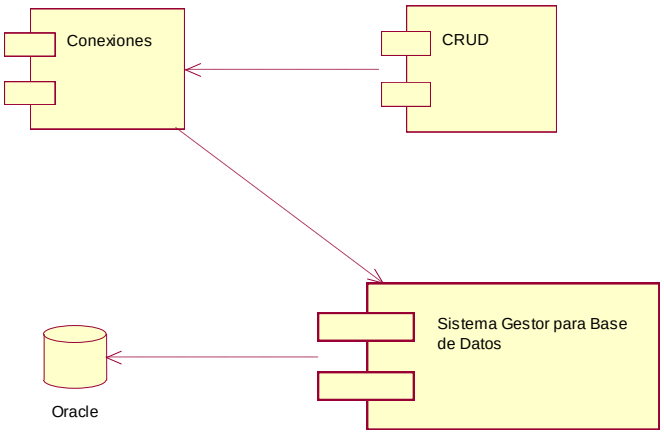


Figura 28. Diagrama de componentes paquete de Datos.

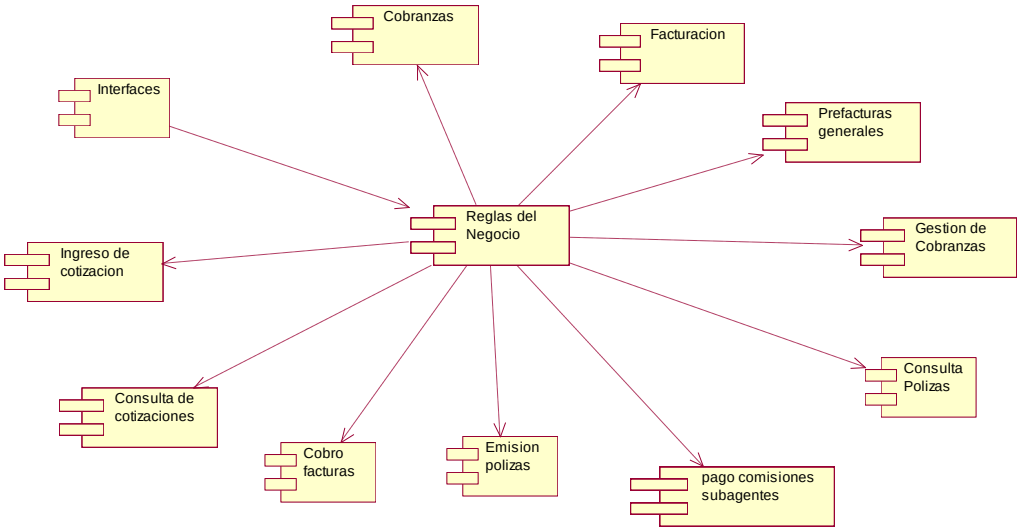


Figura 29. Diagrama de componentes paquete de aplicación.

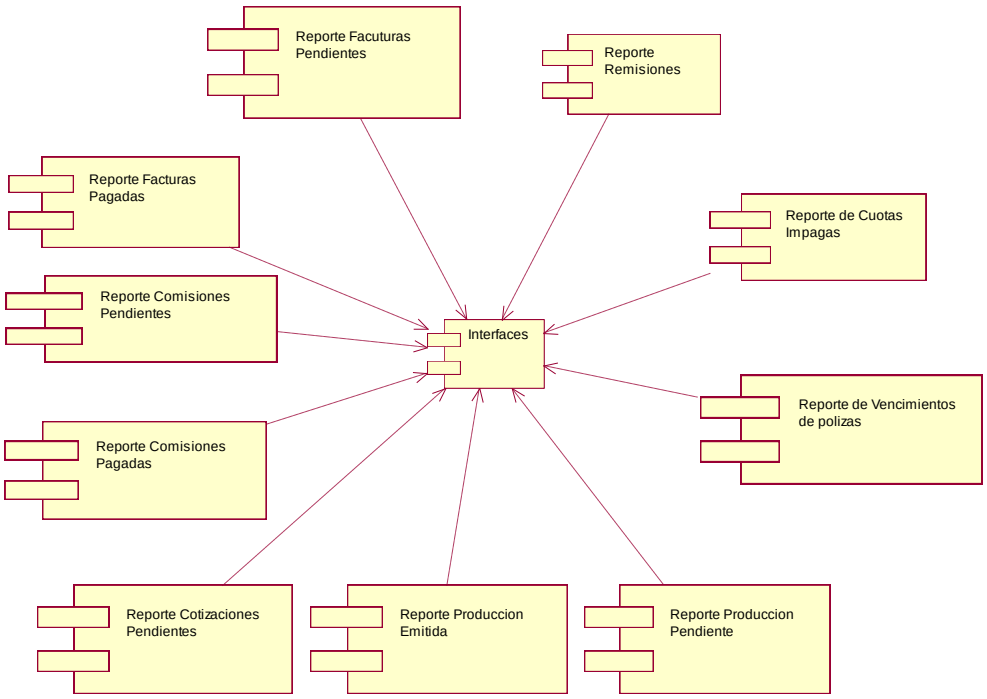


Figura 30. Diagrama de componentes paquete de Cliente

4.04 Matriz de marco lógico (MML).

Tabla 46.

Matriz de Marco Lógico.

	INDICADORES	MEDIOS DE VERIFICACION	SUPUESTO
FINALIDAD			
Sistematizar el proceso de registro, control y cobro de comisiones.	Agilizar los procesos de pago de comisiones en un 80%	Evaluación de tiempos en la ejecución de cada proceso para definir si el sistema es bueno, malo o regular.	Reducir el tiempo de ejecución de cada proceso.
PROPOSITO			
Mejorar el proceso de cobro y pago de comisiones.	Reportes de cobros pendientes y cobros pagados, de igual manera reportes de producción emitida y cobranzas.	Realizar la comparación de los reportes generados actualmente con la información que se tiene de históricos.	Realizar mejoras sobre los incidentes que se presentan con la información obtenida en los reportes.
COMPONENTES			
1. Detalle de comisiones controlado por un sistema informático. 2. Producción emitida monitoreada por un sistema informático. 3. Gestión de pago efectiva y sistematizada.	- Con los convenios de pago de comisiones realizados con la compañía de seguros. - Con el ingreso de producción al sistema se puede verificar el monto de ingresos para la compañía de seguros que se está gestionando. - El pago a los asesores será calculado por en base a los porcentajes acordados con cada asesor y en base a su producción ingresada.	- Reporte de comisiones pendientes y cobradas al bróker. - Reportes de producción emitida, emisiones pendientes y producción pagada. - por medio de reportes de pago y pendientes de pago de comisiones a subagentes.	Aprobación de parte de la gerencia en base al manejo de los reportes presentados.
ACTIVIDADES	**PRESUPUESTO**		
- Capacitar sobre el uso del aplicativo web. - Conocer el flujo de procesos general de la empresa. - Capacitar al personal comercial sobre los productos de las compañías de seguros.	- El costo de capacitación es de $40 por hora posterior a las 40 horas que se incluye en la implementación. - El desarrollo del aplicativo consta de 320 horas laborales a un costo de $40 por hora dando un total de $12800.00. - Acompañamiento $720. - Total implementación $13520.00	Realizar un análisis del cumplimiento que realiza el sistema en base a los requerimientos y alcance del mismo.	Verificar los procesos que se ejecuta en el aplicativo y que procesos podrían faltar de implementar.

Nota: Matriz de marco lógico permite resumir en un solo documento lo que se desea lograr del proyecto.

4.04.01. Vistas arquitectónicas.

4.04.02. Vista lógica.

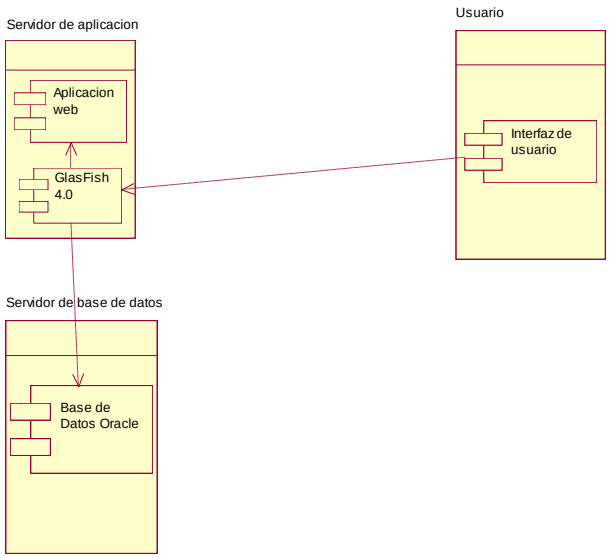


Figura 31. Vista lógica.

Esquema que se maneja para el proyecto para su correcto funcionamiento.

4.04.03. Vista física.

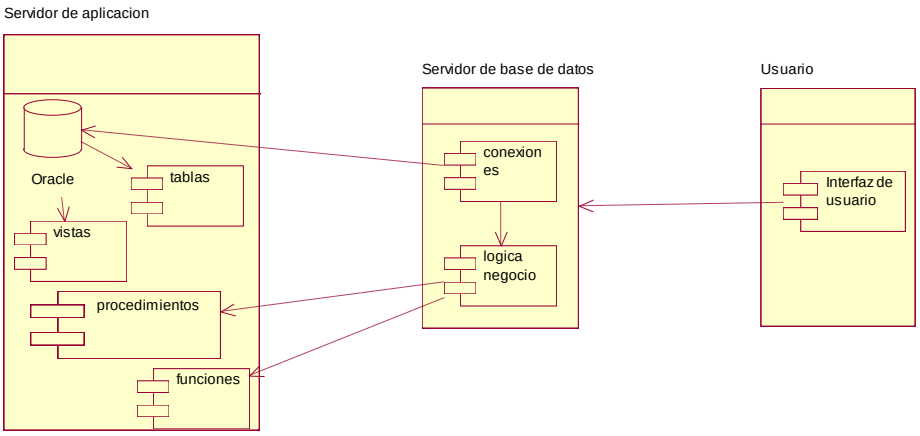


Figura 32. Vista Física.

Esquema manejado para el proyecto, para manejo de capas con usuario, aplicación y base de datos.

4.04.04. Vista de desarrollo.

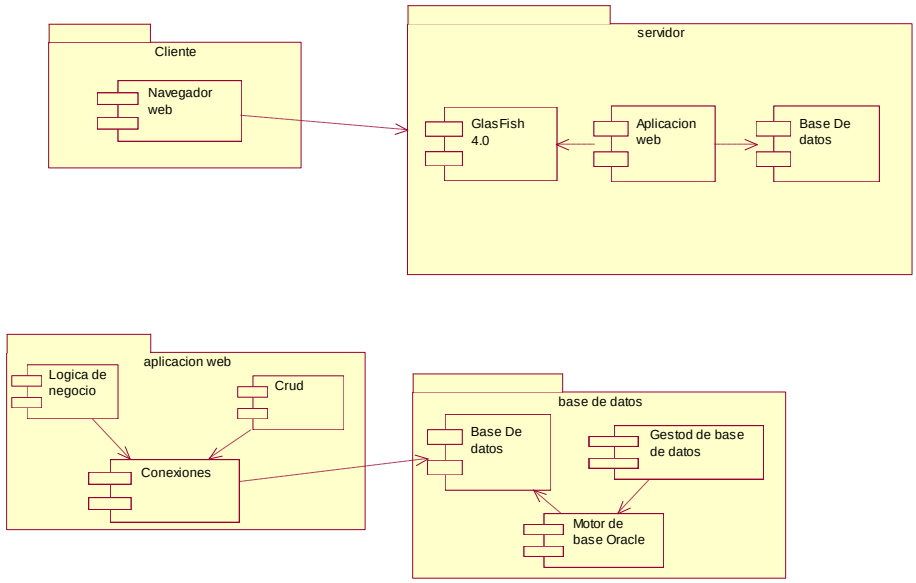


Figura 33. Vista de desarrollo.

Muestra la estructura que se utiliza para el desarrollo de la aplicación web.

4.04.05. Vista de procesos.

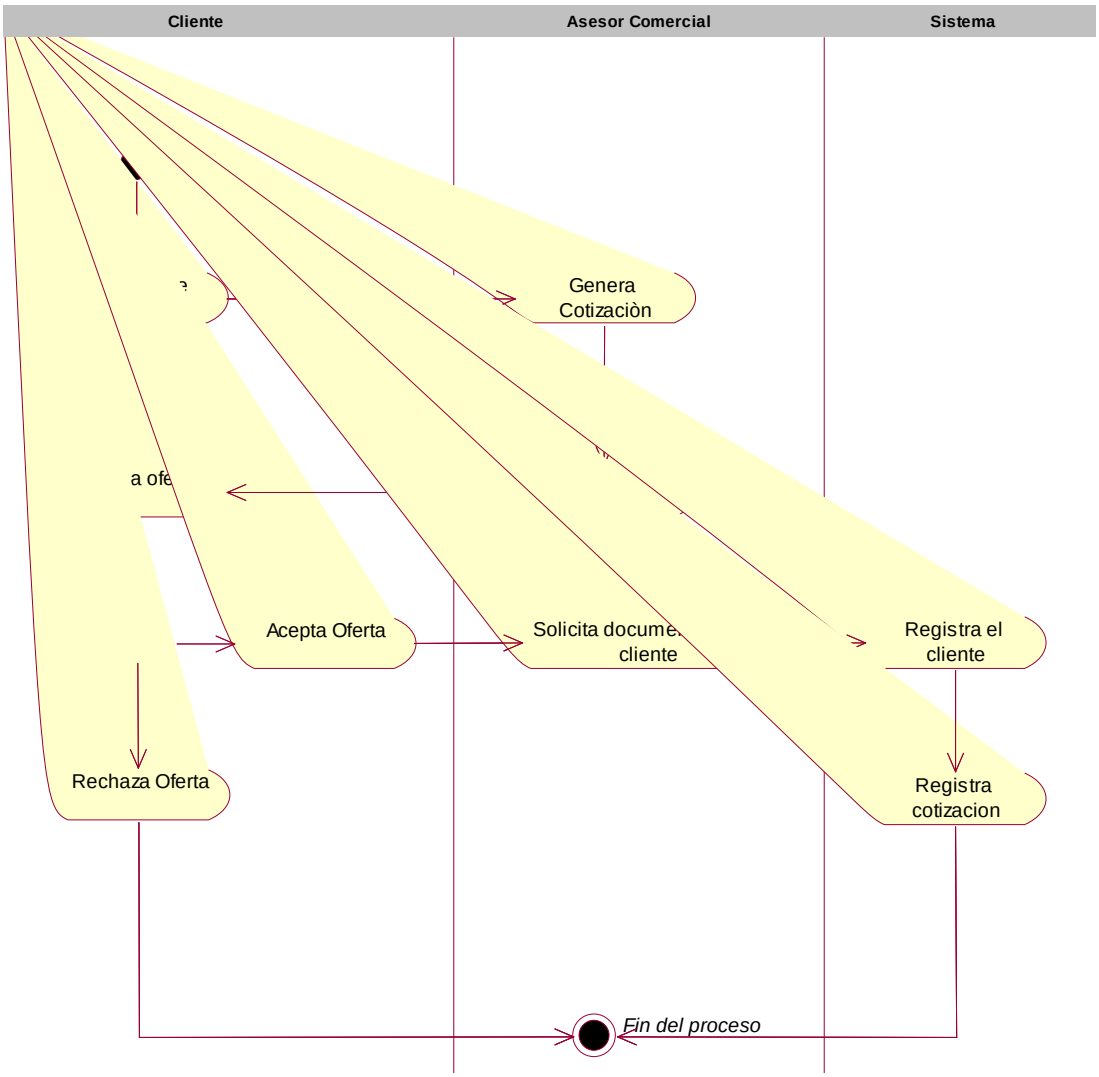


Figura 34. Vista de Proceso generación de cotización.

Presenta los posibles eventos a realizarse, desde que un cliente solicita al asesor comercial una cotización, en caso de que el cliente no deseara la oferta presentada, puede rechazarla.

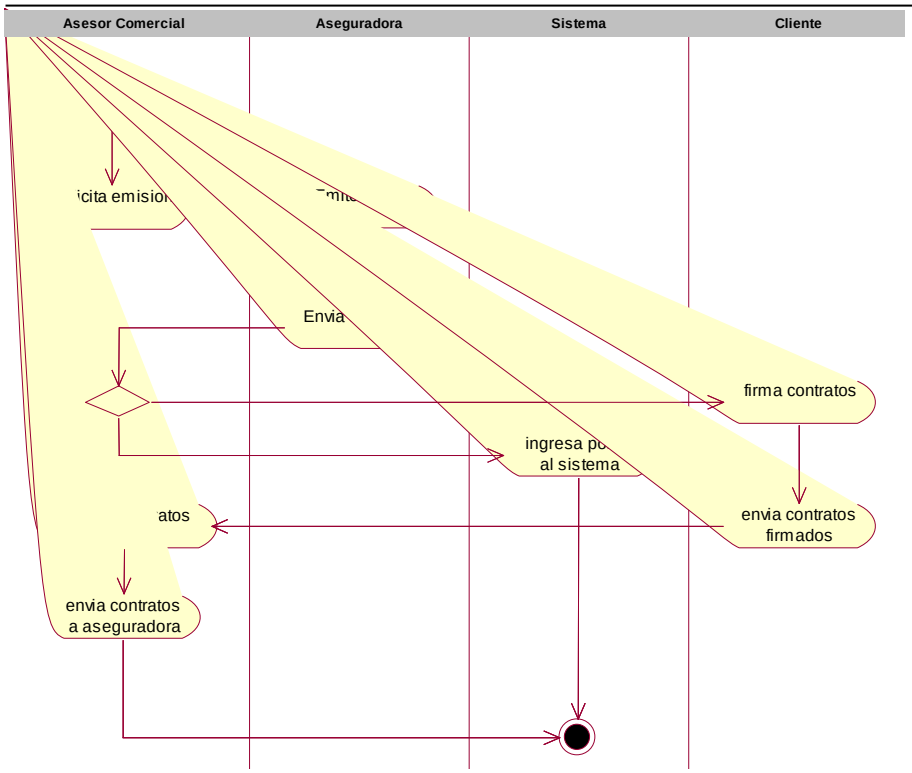


Figura 35. Vista de procesos para la solicitud de emisión

Presenta el proceso de emisión de una póliza por parte del asesor comercial a la compañía de seguros.

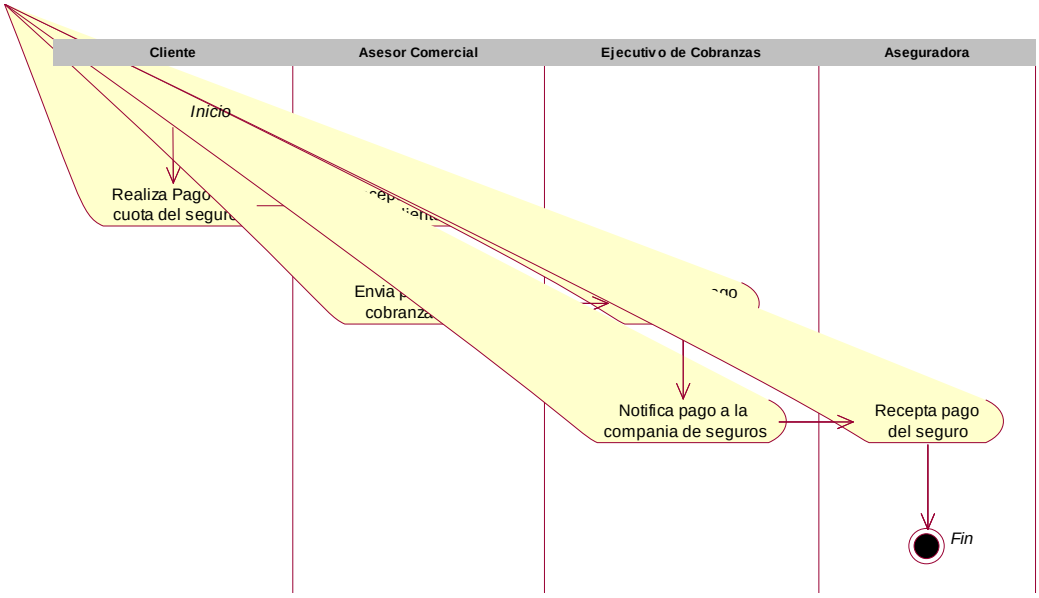


Figura 36. Vista de procesos, proceso registro de pagos clientes.

Presenta el proceso que se realiza para el cobro de las cuotas a los clientes.

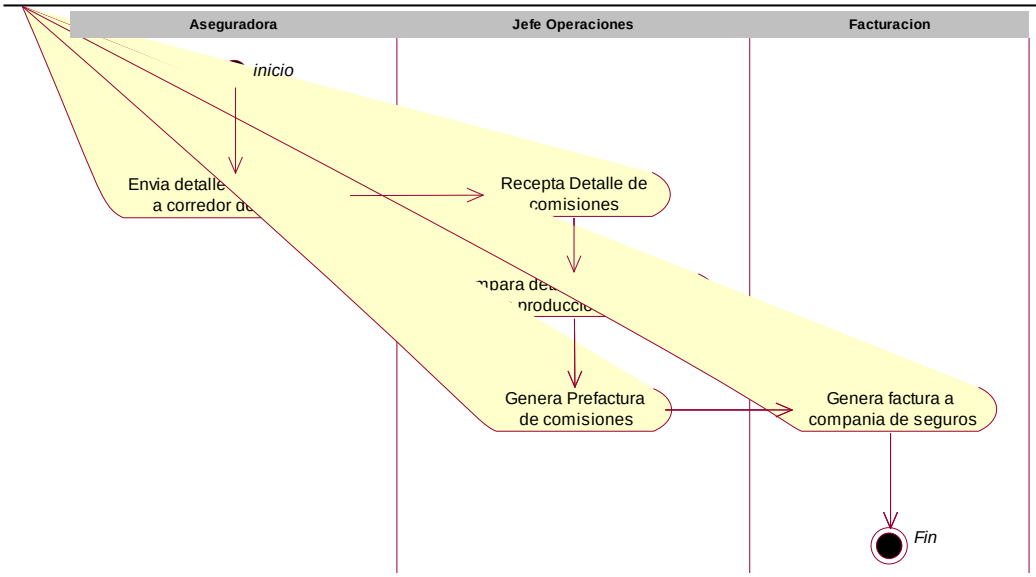


Figura 37. Vista de proceso, Revisión detalle de comisiones.

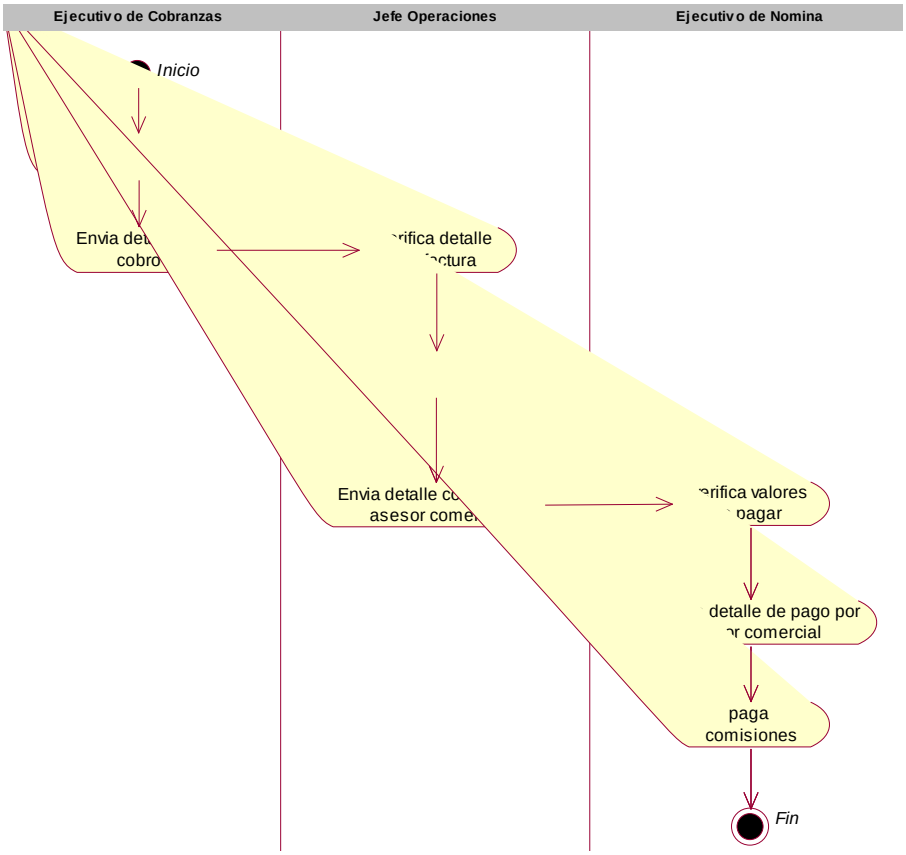


Figura 38. Vista de procesos, proceso de pago de comisiones.

CAPÍTULO V

5. Propuesta.

5.01 Antecedentes.

La empresa Qualityseg S.A. ha presentado varios inconvenientes al intentar manejar un sistema de este tipo, una de las principales causas fue el factor económico, ya que en el mercado existe una aplicación llamada EFI desarrollada por la empresa Tefisoft, esta aplicación demanda un costo demasiado alto anualmente, por tal motivo Qualityseg toma la decisión de no renovar el soporte de dicho sistema.

5.02 Descripción.

La aplicación desarrollada al ser en ambiente web, tendrá un acceso más ágil y de fácil operación, se podrá automatizar procesos al momento de emitir pólizas y poder generar el pago de manera eficiente y sin errores.

5.03 Formulación.

Este aspecto nos permite identificar con ciertos cuestionamientos si el sistema desarrollado tiene una factibilidad en la implementación para la empresa, para ello se ha realizado las siguientes preguntas

¿El sistema cumple con la necesidad de la empresa, en el manejo de comisiones?

El sistema fue desarrollado para realizar el control de la producción que se encuentra emitida, cobranzas y finalmente regularizar el pago de comisiones para cada asesor.

Con los objetivos planteados, el sistema fue desarrollado a medida que el usuario solicito los requerimientos, y definidos en el alcance como tal.

¿Existe facilidad de manejo por parte de los usuarios?

El sistema mantiene una interfaz amigable para el usuario por lo que es da fácil intuición para el manejo, de todas maneras, se proporciona manuales escritos para que puedan hacer uso en caso de que se presente inquietudes al respecto.

Adicional a los manuales se ha realizado un trabajo de soporte a usuarios el cual permite brindar una capacitación personalizada a cada usuario y manteniendo clara su función en el sistema.

¿Qué procesos fueron automatizados con la implementación del sistema?

El principal proceso fue el de pago de comisiones, debido a que se reduce el cuadre manual para cada asesor, al establecer un control en la producción que se emite, el sistema calcula automáticamente la comisión para cada asesor en base al negocio vendido, de esta manera se logra automatizar y se reduce el manejo de archivos manuales que pueden ser modificados por usuarios externos.

5.04 Especificación de estándares de programación.

Tabla 47.

Estándares para base de datos.

Tipo	Prefijo	Ejemplo
Tablas	QLT_T	QLT_T_USUARIO
Vistas	QLT_V	QLT_V_PRODUCCION
Procedimientos almacenados	SP	SP_REP_VENCIMIENTOS
Funciones	F	F_BUSCA_CORREO
Triggers	TRG	TRG_ELIMINA_DUPLICADO
Foreign Key	FK	FK_USUARIO_PERFIL
Primary Key	PK	PK_USUARIOS
Unique	U	U_USUARIO

Nota: Estándares para identificación en nomenclaturas de la base de datos.

Tabla 48. ***Estándares para Interfaces***

Estándares para Interfaces.

Tipo	Prefijo	Ejemplo
InputText	Txt	txtNombre
Button	Btn	btnBuscar
Link	Lnk	lnkEditar
Panel	Pnl	pnlUsuario
DataTable	Dt	dtUsuarios
ConfirmDialog	Cd	CdConfirmar
Menu	Mn	mnPrincipal
Growl	Grl	grlMensajes
Messages	Msg	msgMensajes
GraphicImage	Gpi	gpiLogoEmpresa
PanelGrid	Pg	PgCliente

Nota: Estándares para identificación en nomenclaturas en el diseño de interfaces de usuario.

Tabla 49.

Clases Java.

Tipo	Prefijo	Ejemplo
Unidad de persistencia	PU	BrokerPU.
Paquetes	com.broker	Com.broker.modelo
Clases	No aplica	Usuarios
Clase abstractas	DAOImpl	AccesosDAOImpl
Clase interface	DAO	UsuariosDAO

Nota: Denominación para las distintas clases java que serán utilizadas en el desarrollo de la aplicación.

Tabla 50.

Tipos de Datos.

Nombre	Tipo
Int	Entero
Long	Entero
Float	Decimal Simple
Double	Decimal Doble
Char	Carácter Simple
Boolean	Valor Verdadero o falso
String	Cadena de caracteres
List	Lista de Objetos
Date	Fechas

Nota: *Tipos de datos a usarse para el desarrollo del aplicativo.*

5.05 Diseño de interfaces de usuario.

Para el desarrollo de las interfaces de usuario se utilizó el framework primefaces, con su témplate ultima.

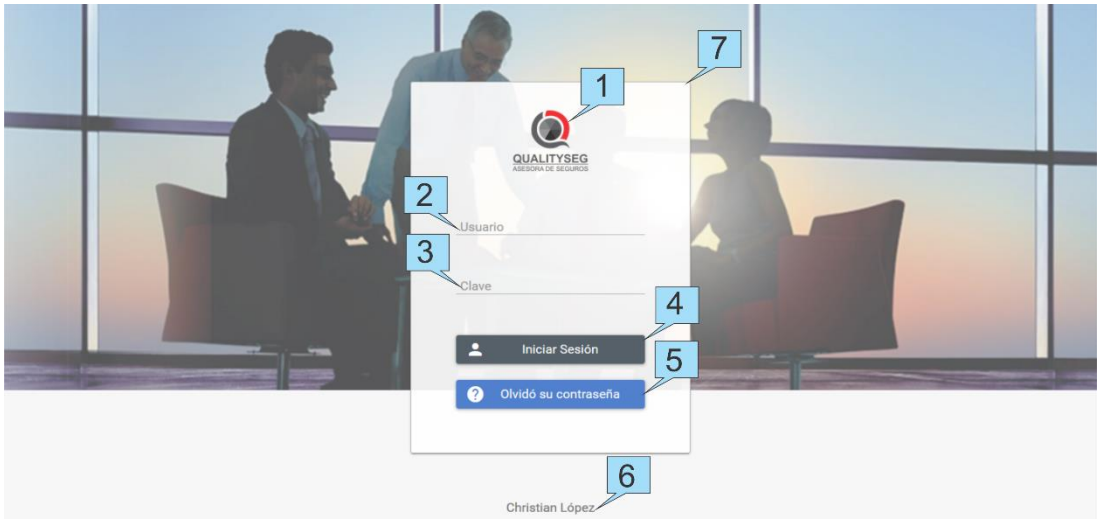


Figura 39. Interfaz de usuario, acceso al sistema.

1.- Etiqueta GraphicImage para logo de la empresa; 2.- Etiqueta Label detalla el nombre del campo; 3.- InputText caja de texto para capturar el valor solicitado y detallado por la etiqueta label; 4.- CommandButton botón que permite el acceso al sistema; 5.- CommandButton botón que permite el cambio de clave en caso de olvido; 6.- Label que detalla un nombre, en este caso el nombre del desarrollador del sistema.

Figura 40. Interfaz de usuario, Página de inicio.

1.- GraphicImage presenta el logo del grupo de compañías; 2.- Button que permite ocultar el menú de accesos; 3.- GraphicImage con el logo de la compañía; 4.- OutputLabel que presenta el nombre del usuario que accede al sistema; 5.- Menú, detalla los accesos que puede tener un usuario en el sistema; 6.- Panel que detalla el pie de página de cada pantalla.

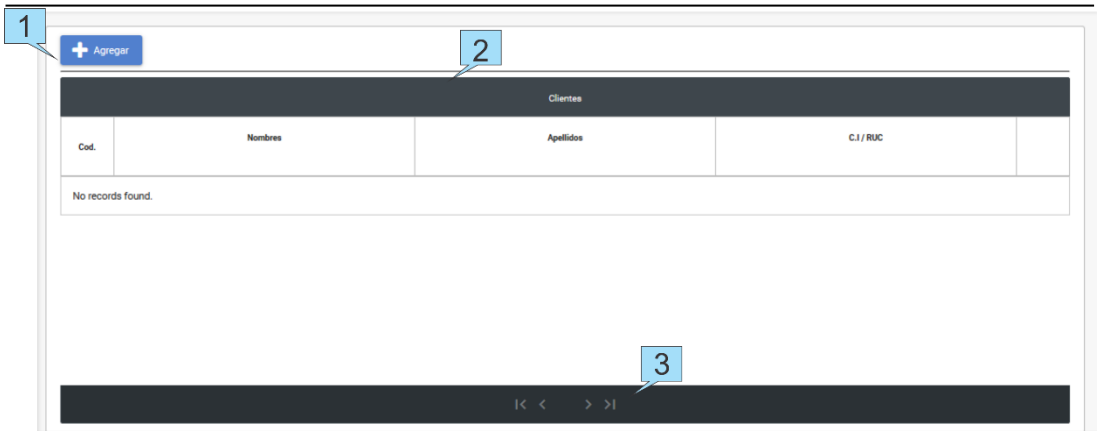


Figura 41. Interfaz de usuario Creación Clientes.

1.- CommandBotton para agregar un nuevo usuario; 2.- DataTable presenta la lista de los clientes registrados; 3.- Paginación, permite paginar la información presentada por el datatable.

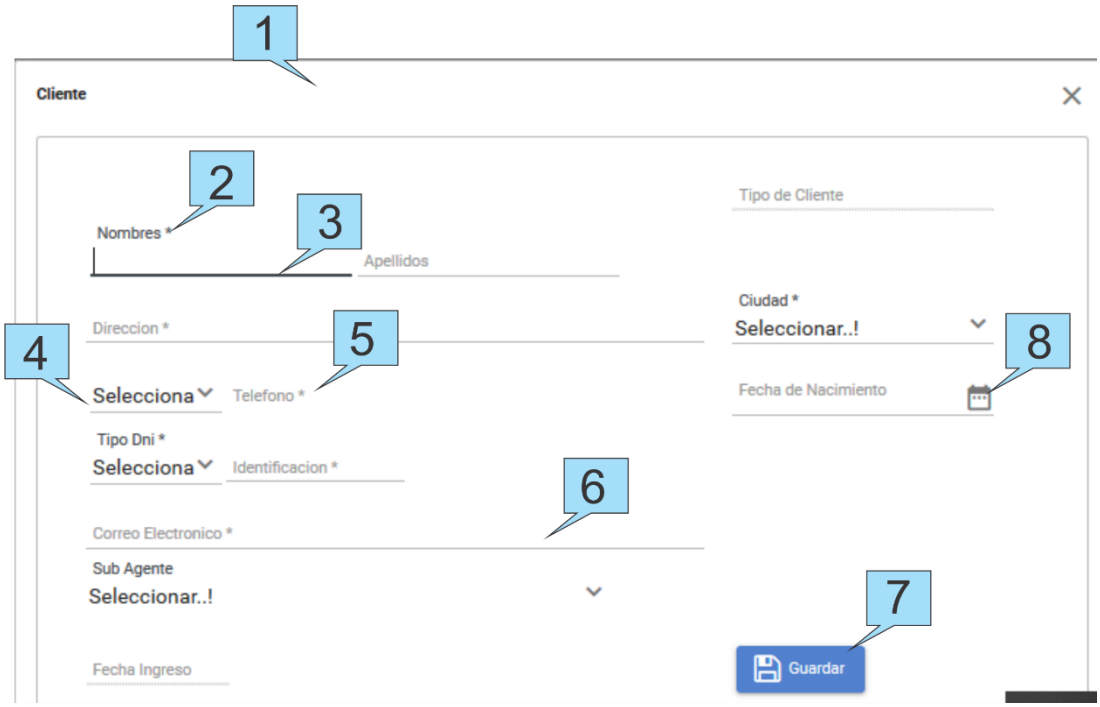


Figura 42. Interfaz de usuario Registro de aseguradoras.

1.- Dialog, panel de dialogo; 2.- outputLabel etiqueta para identificar el parámetro a ingresar; 3.- inputtext etiqueta que permite capturar el parámetro de entrada; 4.- SelectOneMenu, lista de opciones seleccionables; 5.- label identificador de un campo; 6.- inputText para registro de dato de correo electrónico; 7.- CommandBotton ejecuta una acción en este caso guarda la información registrada; 8.- Calendar, permite el ingreso de fechas por medio de un calendario o digitando por teclado.

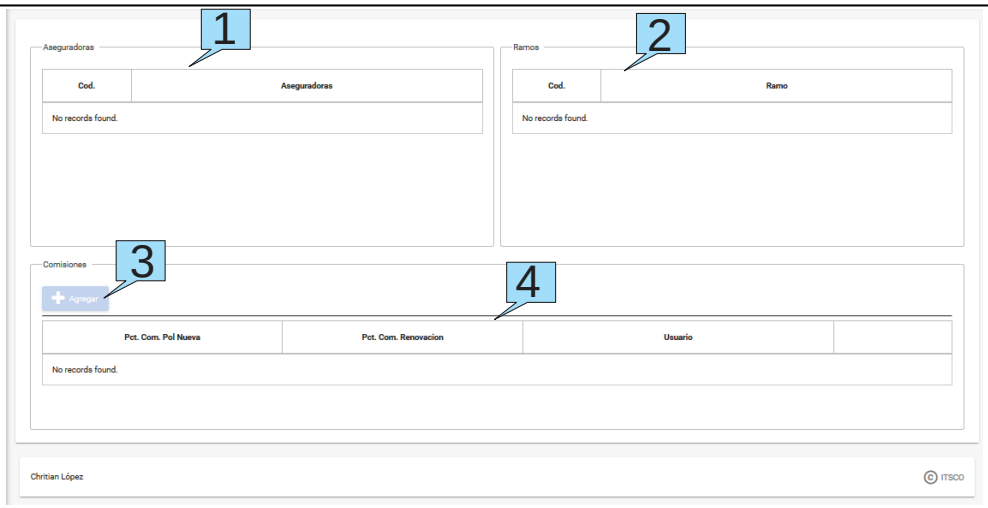


Figura 43. Interfaz de usuario, Comisiones bróker.

Presenta la interfaz de parametrización del porcentaje de comisiones que paga una compañía de seguros al corredor de seguros.

Ítem 1: Listado de Compañías de seguros registradas

Ítem 2: Listado de ramos registrados.

Ítem 3: botón para registrar un nuevo porcentaje de comisión.

Ítem 4: Listado de comisiones para el ramo y aseguradoras seleccionadas.

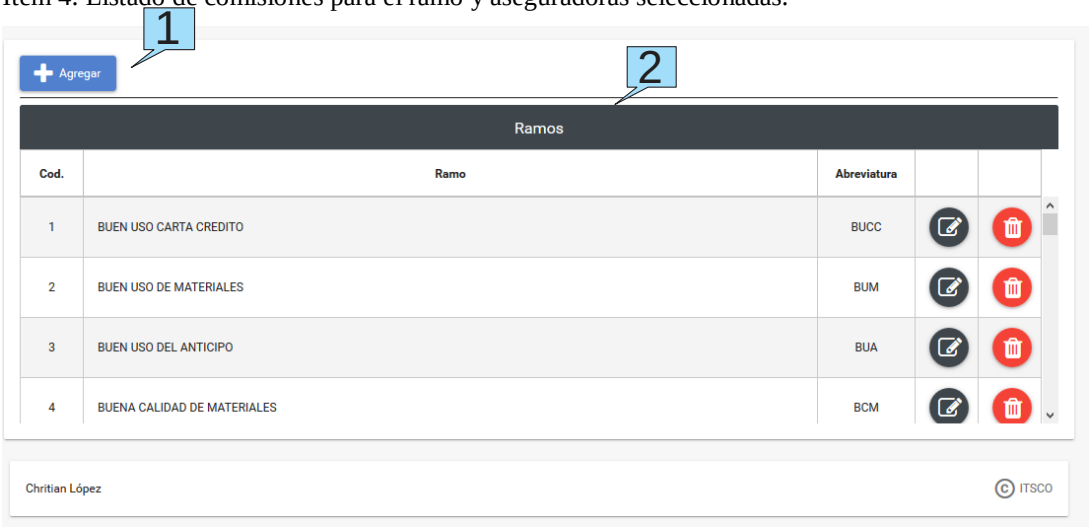


Figura 44. Interfaz de usuario, Ramos.

Presenta el listado de ramos que se encuentra registrado en el sistema., con una abreviatura la cual será referenciada en reportes.

Ítem 1: botón para registrar un nuevo ramo

Ítem 2: listado de ramos registrados.

5.06. Especificación de pruebas de unidad.

Tabla 51.

Pruebas de unidad, Validación de acceso al sistema

Identificador de prueba de unidad	PU001
Método a probar	Acceso al sistema
Objetivo de la prueba	Evidenciar los parámetros necesarios para poder acceder al sistema e identificar los posibles casos en caso de que no exista un usuario registrado
Datos de entrada	
En la pantalla de acceso se solicita dos parámetros, en la primera caja de texto se ingresa el nombre de usuario y la segunda caja de texto la clave.	
Resultado Esperado	
El sistema valida que el usuario y contraseña coincidan con los registrados, en caso de que no exista un usuario el sistema presenta mensajes indicando que no existe usuario, de igual manera si la clave no es correcta se presenta un mensaje indicando que el usuario y/o clave no son correctos.	
Comentarios	
Para acceder a la pantalla de acceso al sistema se debe ingresar por medio de un navegador de internet, de preferencia Google Chrome o Mozilla Firefox.	

Nota: Detalles de prueba de unidad para el ingreso al sistema.

Tabla 52.

Prueba de unidad, Validación de registro de usuarios.

Identificador de prueba de unidad	PU002
Método a probar	Registro de usuarios
Objetivo de la prueba	Evidenciar que las condiciones de ingreso para cada campo sean correctas, es decir que sean validas
Datos de entrada	
En la pantalla de usuarios, el administrador del sistema debe ingresar el nombre, apellidos, email, perfil y el nombre de usuario.	
Resultado Esperado	
Al ingresar todos los datos de entrada, se espera que los datos ingresados permitan a un usuario acceder al sistema en base al perfil asignado.	
Comentarios	
Con el registro de usuarios y la validación de que exista información correcta en su ingreso se espera poder validar que el ingreso no presente problemas con los tipos de datos.	

Nota: Prueba unitaria para verificación de registro de usuarios al sistema.

Tabla 53.

Pruebas de Unidad, verificación de redundancia de datos.

Identificador de prueba de unidad	PU003
Método a probar	Redundancia al momento de ingresar información al sistema
Objetivo de la prueba	Identificar que no exista redundancia de datos al momento de ingresar información al sistema.
Datos de entrada	
Pantalla de registro de usuarios, campo usuario.	
Resultado Esperado	
El sistema debe controlar o presentar en un mensaje en el caso que se intente ingresar un nombre de usuario que ya se encuentra registrado en la base de datos, impidiendo que el registro sea ingresado y solicitando el cambio en el nombre de usuario.	
Comentarios	
Con esta validación se podrá evitar que exista usuarios que deseen ingresar al sistema y que posiblemente tengan nombres similares.	

Nota: Prueba de unidad para verificación de redundancia de datos.

Tabla 54.

Prueba de unidad validación de campos numéricos y de texto.

Identificador de prueba de unidad	PU004
Método a probar	Validación de ingreso de datos en campos numéricos y de texto exclusivamente.
Objetivo de la prueba	Validar que los campos que sean únicamente de tipo texto no permitan ingresar caracteres numéricos, y en campos de tipo numérico no permita ingresar caracteres de tipo texto.
Datos de entrada	
Caja de texto para datos numéricos, caja de texto para datos de texto.	
Resultado Esperado	
Cada campo debe permitir ingresar únicamente el tipo de dato para el que se utiliza, es decir si se desea ingresa un número de identificación, solo se permite el registro de datos numéricos. Si se desea ingresar el nombre de un cliente o de una persona, solo debe ser permitido ingresar letras no datos numéricos.	
Comentarios	
Los campos cumplen con la validación, así mismo existen campos mixtos donde se ingresa los dos tipos de datos, para estos campos no se registra ninguna validación adicional.	

Nota: *prueba de unidad para la validación de ingreso de datos en los campos exclusivos de texto y numéricos.*

Tabla 55.

Prueba de unidad digito verificador cédula.

Identificador de prueba de unidad	PU005
Método a probar	Validación de cédula
Objetivo de la prueba	Poder identificar que el número de identificación de un cliente es válido por medio del ultimo digito mejor conocido como digito verificador de la cédula.
Datos de entrada	
Al momento de registrar un cliente se debe ingresar el dato de identificación.	
Resultado Esperado	
Al momento de ingresar el dato de identificación, este validara dependiendo el tipo de identificación que se seleccione, puede ser cédula, ruc o pasaporte. Para que se ejecute la validación se debe seleccionar ruc o cédula, en caso de que no sea correcto, se presenta un mensaje el cual informa que no es un dato valido impidiendo el registro del nuevo cliente.	
Comentarios	
Con el digito verificador se puede validar que la identificación de un cliente es correcta, de lo contrario puede existir errores al momento de ingresar la información al sistema y poder generar las notificaciones a las entidades de control.	

Nota: prueba de unidad para verificación de validez de un número de cédula.

Tabla 56.

Prueba de unidad validación de fechas

Identificador de prueba de unidad	PU006
Método a probar	Validación de fechas
Objetivo de la prueba	Verificar que el dato que se ingresa en los campos de fechas sea del tipo fecha, de lo contrario no debe permitir continuar con el ingreso.
Datos de entrada	
Campos de tipo fecha, como fechas de nacimiento, fechas de vigencia en pólizas.	
Resultado Esperado	
El sistema valida que los campos ingresados son de tipo fecha, este dato se lo puede registrar digitando la fecha o abriendo un calendario en el mismo campo.	
Comentarios	
Con la validación correcta de ingreso de fechas, se puede mantener una integridad de información en cuanto a las pólizas ingresadas al sistema manejando un dato correcto y evitando fallas.	

Nota: Prueba unitaria para el registro de fechas en los campos que solicita una fecha sea fecha de nacimiento o fechas de inicio y fin de vigencia en una póliza.

Tabla 57.

Prueba de unidad, funcionalidad de botones.

Identificador de prueba de unidad	PU007
Método a probar	Funcionalidad de botones.
Objetivo de la prueba	Validar que la funcionalidad de cada botón ejecute el proceso para el cual fue colocado.
Datos de entrada	
Botones en interfaces de usuario.	
Resultado Esperado	
Los botones en las interfaces ejecutan su funcionalidad, de acuerdo a lo indicado en la etiqueta de cada botón.	
Comentarios	
El sistema presenta botones con etiquetas fáciles de identificar, los mismos permiten al usuario identificar cada botón por medio de gráficos amigables los cuales representan su funcionalidad.	

Nota: prueba unitaria para la validación de funcionalidad de cada botón del sistema.

5.07. Pruebas de aceptación.

Las pruebas de aceptación corresponden a las últimas etapas previas a la liberación de una nueva versión al ambiente de producción, con el fin de determinar si cumple con las necesidades y/o requerimientos determinados por la empresa y sus funcionarios.

Tabla 58.

Prueba de aceptación, acceso al sistema

Identificador de la prueba	PA001
Tipo de usuario	Todos los usuarios
Objetivo de la prueba	Validar el funcionamiento de la pantalla de acceso al sistema, utilizado por todos los usuarios
Secuencia de eventos	
• El usuario ingresa al sistema por medio de un navegador web, en el mismo coloca la dirección o link de conexión al sistema. • Al acceder a la pantalla el sistema solicitara un usuario y una contraseña. • El usuario ingresa con las credenciales asignadas por el administrador del sistema.	
Resultados esperados	
Con el flujo de procesos detallado se espera que el usuario pueda acceder al sistema con su propio usuario y clave que será asignado por el administrador de sistema.	
Comentarios	
El usuario maneja la pantalla con facilidad y no presenta inconvenientes en su ejecución.	
Estado	
En ejecución.	

Nota: Tabla que refleja el flujo de eventos para las pruebas de aceptación por parte del cliente.

Tabla 59.

Prueba de aceptación, registro de usuarios.

Identificador de la prueba	PA002
Tipo de usuario	Administrador
Objetivo de la prueba	Validar que la funcionalidad de la pantalla sea sencilla para el manejo del administrador encargado en la empresa.
Secuencia de eventos	
• Una vez ingresado al sistema, el usuario debe ir al menú configuración, en el cual se encuentra la opción usuarios. • En la pantalla usuarios se presenta un listado de usuarios, para poder registrar un nuevo usuario debe dar clic sobre el botón agregar. • Al dar clic en el botón agregar se despliega un panel de registro, el mismo solicita los datos de nombres, apellidos, email, perfil y usuario. • Al registrar todos los parámetros se debe presionar el botón grabar. • Una vez presionado el botón grabar, se presenta un mensaje informando que el registro fue almacenado correctamente y actualiza el listado de la pantalla principal	
Resultados esperados	
El ejecutar en la prueba el usuario que realiza las operaciones como administrador presenta un óptimo funcionamiento en el proceso, de igual manera con las interfaces amigables no presenta contratiempos en el registro.	
Comentarios	
Al registrar un usuario nuevo, el sistema no solicita ningún tipo de clave a asignar a dicho usuario, esto se debe a que como es un usuario nuevo su clave será el mismo nombre de usuario, cuando acceda por primera vez al sistema, este solicitará su cambio de manera inmediata.	
Estado	
En ejecución.	

Nota: Prueba de aceptación realizada por parte del administrador del sistema, detalla la secuencia de eventos realizados para poder registrar un nuevo usuario.

Tabla 60.

Prueba de aceptación, registro de cotización.

Identificador de la prueba	PA003
Tipo de usuario	Ejecutivo de cuenta
Objetivo de la prueba	Verificar el funcionamiento en el proceso de ingreso de una cotización al sistema.
Secuencia de eventos	
• El usuario con perfil ejecutivo de cuenta podrá acceder al menú de producción, el mismo presentará una opción de cotización. • Al ingresar a esta pantalla se podrá consultar las cotizaciones pendientes, o si es el caso registrar una nueva cotización. • Al dar clic sobre el botón nuevo, el sistema redireccionara a la pantalla de registro de cotizaciones. • En dicha pantalla el usuario debe seleccionar el cliente al cual se enviará la cotización, la compañía de seguros, el ramo y por último el producto. • Una vez seleccionados los parámetros iniciales se habilitará el botón para el registro de los objetos asegurados. • En caso de ser un ramo de vehículos, se solicitará las características del vehículo. • Ingresado el vehículo se lo debe asignar un valor asegurado el mismo en base al producto generara un valor el cual será el valor que costara el seguro. • Para finalizar y enviar al cliente el sistema presenta un botón el cual indica enviar cotización, al dar clic este presenta una pantalla con los datos ingresados. • Dicha pantalla presenta nuevamente un botón con el nombre enviar el cual al dar clic envía mediante correo electrónico un email al cliente.	
Resultados esperados	
Se ingresa al sistema la información del cliente, la misma que será visualizada por el ejecutivo de cuenta, finalmente envía por correo electrónico la cotización la misma que el cliente podrá revisar una vez enviado.	
Comentarios	
El proceso al parecer es un poco extenso ya que el corredor de seguros genera cotizaciones desde los cotizadores de cada compañía de seguros.	
Estado	
Pendiente	

Nota: Prueba de aceptación para el registro de cotizaciones y envió al cliente.

5.08. Especificación de pruebas de carga.

Son pruebas que se realiza para determinar lo rápido que realiza una tarea el sistema en especial en ambiente de producción, adicional a esto este tipo de pruebas sirve para visualizar o evidenciar otros atributos como escalabilidad, fiabilidad y uso de los recursos con los que cuenta el sistema.

Tabla 61.

Pruebas de carga del sistema

Identificador de la prueba	PC001
Tipo de usuario	Manejo de grandes cantidades de información
Objetivo de la prueba	Verificar la operatividad del sistema en un ambiente con excesiva carga de información.
Descripción	
• Se realizará la medición de tiempos a cada una de las ventanas del sistema. • Se realizará el ingreso de información errónea para evidenciar el comportamiento del sistema ante posibles inconsistencias del mismo. • Manejo de gran cantidad información en los módulos de producción, cobranzas y pago de comisiones, de tal manera se evidencie si existe problemas al procesar dicha información.	
Resultados esperados	
• Se espera que el sistema cumpla con su funcionalidad indistintamente la cantidad de información que se encuentre registrada. • La validación de los campos que se encuentra en las pantallas, con el fin de verificar que cumpla con las validaciones de ser el caso. • El sistema no debe colapsar al procesar gran cantidad de información, las consultas deben ser transparentes y en tiempos adecuados.	
Comentarios	
El sistema soporta de manera adecuada las pruebas realizadas, dando confianza en el manejo de procesos y brindando fluidez en cada una de las actividades y tareas.	

Nota: Pruebas de carga realizadas al sistema, presenta las descripciones y los resultados que se espera en la ejecución.

5.09. Configuración del ambiente mínimo.

Tabla 62.

Requerimientos mínimos servidor de base de datos.

BASE DE DATOS ORACLE 11g	
Requerimiento	**Valor mínimo**
Memoria RAM	512MB
Espacio de disco Duro	1.5GB
Sistema operativo	Windows server 2008 o superiores. Windows 7 o superiores.
Arquitectura del sistema operativo	32bits o 64bits

Nota: Requisitos mínimos para funcionamiento de base de datos Oracle 11g.

Tabla 63.

Requerimientos mínimos servidor de aplicación.

GLASFISH 4.0	
Requerimiento	**Valor mínimo**
Memoria RAM	1GB
Espacio de disco Duro	250GB
Sistema operativo	Windows server 2008 o superiores. Windows 7 o superiores.
Máquina virtual de Java	JDK 6 de 32 o 64 bits

Nota: Requisitos mínimos para funcionamiento del servidor de aplicación con Glasfish 4.0.

CAPÍTULO VI

6. Aspectos administrativos.

6.01 Recursos

Los recursos utilizados para el desarrollo de la aplicación web son humanos y técnicos, mismos que permiten el correcto funcionamiento y ejecución.

Tabla 64.

Recurso Humano.

Cargo	Nombre	Institución
Tutor del Proyecto	Ing. Jaime Basantes	Docente Instituto Cordillera
Lector	Lcda. Patricia Garzón	Docente Instituto Cordillera
Estudiante	Christian López	Estudiante Instituto Cordillera
Gerente General	Hernán Ochoa	Qualityseg S.A
Jefe de Operaciones	Luis González	Qualityseg S.A

Nota: Detalle del recurso humano que se utilizó para el desarrollo del proyecto.

Tabla 65.

Recursos Técnicos.

Ítem	Detalle
Sistema Operativo	Windows server 2008 R2 Standard 64Bits
Memoria RAM	4.00 GB
Procesador	Intel Xenón CPU E5-2640 2.40Ghz
Base de datos	Oracle 11g XE
Servidor de aplicación	Glasfish 4.0
IDE de programación	NetBeans 8.1
Gestor de Base de datos	Toad For Oracle 12
Frimework	Prime faces 6.0
Tema Primefaces	Primefaces Ultima v1.1.4

Nota: Detalle de recursos Tecnológicos aplicados en el desarrollo del proyecto

6.02 Presupuesto.

En el transcurso del desarrollo del proyecto se intenta presentar un presupuesto el cual debería ser el costo que representa el desarrollo de la solución a la problemática que se ha presentado, de este modo se ha definido varios puntos, mismos que se detallan en la siguiente tabla de presupuestos aplicados al proyecto.

Estos costos representan el ahorro que realiza la empresa al permitir implementar la aplicación.

Tabla 66.

Tabla De Presupuesto Del Proyecto

Descripción	Cantidad	Valor Unitario	Valor total
Implementos de escritorio		20.00	20.00
Hojas de papel bond	200	0.05	10.00
Tóner de impresora	1	40.00	40.00
Base de datos	1	0.00	0.00
ID de programación	1	0.00	0.00
Tema Primefaces	1	79.00	79.00
Energía Eléctrica	5	10.00	50.00
Internet	5	15.00	75.00
Gastos Varios		30.00	30.00
Total			**304.00**

Nota: Tabla de presupuesto donde se detalla los costos de inversión en el desarrollo del proyecto, dichos costos son el valor que se ahorra la empresa con el desarrollo de la aplicación.

6.03 Cronograma.

Se detalla en una línea de tiempo las fechas que se debe presentar cada etapa del desarrollo del proyecto, con el fin de poder establecer que tareas son primordiales y cuales son dependientes de otras.

En este caso se realiza un cronograma con un flujo lineal definiendo que una tarea depende de otra, esto ya que con el avance de cada tarea se puede desarrollar la siguiente.

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		**Proyecto de Titulacion**	**87 días**	**jue 28/06/18**	**vie 26/10/18**
2		**1. Antecedentes**	**2 días**	**jue 28/06/18**	**vie 29/06/18**
3		1.1 Contexto	1 día	jue 28/06/18	jue 28/06/18
4		1.2 Justificacion	1 día	vie 29/06/18	vie 29/06/18
5		**2. Analisis de involucrados**	**11 días**	**lun 02/07/18**	**lun 16/07/18**
6		2.1 Vision y Alcance	1 día	lun 02/07/18	lun 02/07/18
7		2.2 Entrevistas	2 días	mar 03/07/18	mié 04/07/18
8		2.3 Matriz de requerimientos	2 días	jue 05/07/18	vie 06/07/18
9		2.4 Descripcion detallada	3 días	lun 09/07/18	mié 11/07/18
10		2.5 Mapa de involucrados	1 día	jue 12/07/18	jue 12/07/18
11		2.6 Matriz de analisis de involuc	2 días	vie 13/07/18	lun 16/07/18
12		**3. Problema y objetivos**	**14 días**	**mar 17/07/18**	**vie 03/08/18**
13		3.1 Arbol de problemas	1 día	mar 17/07/18	mar 17/07/18
14		3.2 Arbol de objetivos	1 día	mié 18/07/18	mié 18/07/18
15		3.3 Casos de uso	5 días	jue 19/07/18	mié 25/07/18
16		3.4 Especificacion de casos de u:	2 días	jue 26/07/18	vie 27/07/18
17		3.5 Casos de uso de realizacion	3 días	lun 30/07/18	mié 01/08/18
18		3.6 Diagramas de secuencias	2 días	jue 02/08/18	vie 03/08/18
19		**4. Analisis de alternativas**	**16 días?**	**lun 06/08/18**	**lun 27/08/18**
20		4.1 Matriz de análisis de alterna	1 día	lun 06/08/18	lun 06/08/18
21		4.2 matriz de impactos de objeti	1 día	mar 07/08/18	mar 07/08/18
22		4.3 Diagrama de estratégias	1 día	mié 08/08/18	mié 08/08/18
23		4.4 Diseño de clases	2 días	jue 09/08/18	vie 10/08/18
24		4.5 Diagrama de clases	2 días	lun 13/08/18	mar 14/08/18
25		4.6 Modelo logico - físico	3 días	mié 15/08/18	vie 17/08/18
26		4.7 Diagrama de componentes	1 día	lun 20/08/18	lun 20/08/18
27		4.8 Matriz de marco lógico	1 día	mar 21/08/18	mar 21/08/18
28		**4.9 Vistas arquitectonicas**	**4 días?**	**mié 22/08/18**	**lun 27/08/18**
29		4.9.1 Vista Lógica	1 día	mié 22/08/18	mié 22/08/18
30		4.9.2 Vista Física	1 día	jue 23/08/18	jue 23/08/18
31		4.9.3 Vista de Desarrollo	1 día	vie 24/08/18	vie 24/08/18
32		4.9.4 Vista de procesos.	1 día?	lun 27/08/18	lun 27/08/18
33		**5. Propuesta**	**37 días**	**mar 28/08/18**	**mié 17/10/18**
34		5.1 Antecedentes	1 día	mar 28/08/18	mar 28/08/18
35		5.2 Descripcion	1 día	mié 29/08/18	mié 29/08/18
36		5.3 Formulacion	1 día	jue 30/08/18	jue 30/08/18
37		5.4 Especificacion de estandares de programacion	2 días	vie 31/08/18	lun 03/09/18
38		5.6 Diseño de interfaces de usu:	15 días	mar 04/09/18	lun 24/09/18
39		5.7 Especificacion de pruebas de unidad	4 días	mar 25/09/18	vie 28/09/18
40		5.7 Pruebas de aceptacion	5 días	lun 01/10/18	vie 05/10/18
41		5.8 Especificacion de pruebas de carga	7 días	lun 08/10/18	mar 16/10/18
42		5.9 Configuracion del ambiente minimo	1 día	mié 17/10/18	mié 17/10/18
43		**6. Aspectos administrativos**	**6 días**	**jue 18/10/18**	**jue 25/10/18**
44		6.1 Recursos	2 días	jue 18/10/18	vie 19/10/18
45		6.2 Presupuesto	4 días	lun 22/10/18	jue 25/10/18

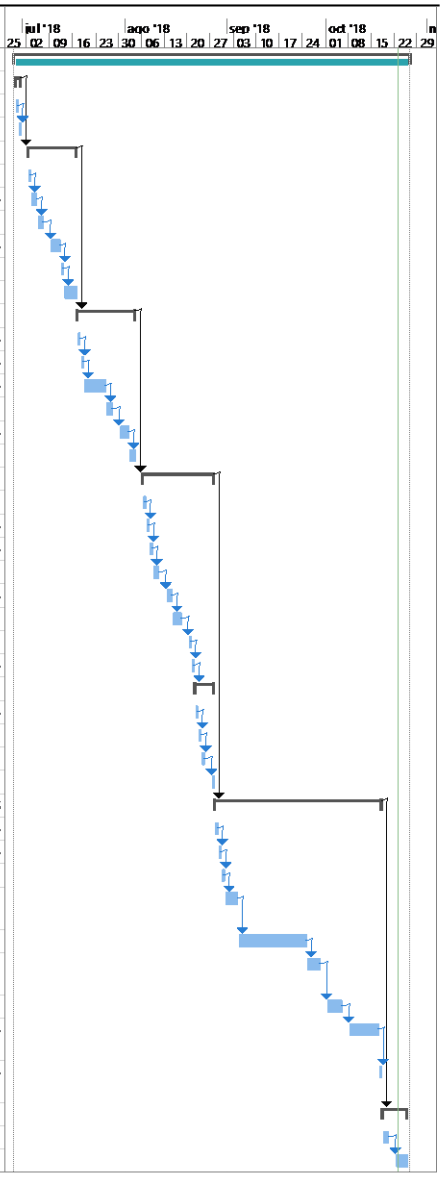


Figura 45. Cronograma de actividades.

Presenta cronograma de actividades que se realizara para la ejecución y desarrollo del proyecto.

CAPÍTULO VII

7. Conclusiones y recomendaciones.

7.01 Conclusiones

Con el desarrollo de la aplicación web para la empresa Qualityseg S.A. se ha podido solucionar el inconveniente que se presentaba al momento de manejar la información en archivos manuales, de igual manera se ha podido automatizar el pago de comisiones para cada asesor comercial, y se logra tener un flujo correcto en el cobro de comisiones a cada compañía de seguros.

En el desarrollo del proyecto únicamente se globalizo el ingreso de producción, es decir solo pólizas nuevas y renovaciones para ramos del área de generales y fianzas, por lo que la aplicación no maneja el área de personas y tampoco maneja lo que son anexos.

En base a los conocimientos adquiridos en el transcurso de la carrera de análisis de sistemas se puede agregar que para cada proceso se debe estandarizar con manuales y políticas para que de esta manera no exista inconvenientes en caso de cambio de personal, de igual manera es importante implementar una cultura informática a cada colaborador en la empresa con el fin de que la herramienta implementada pueda tener éxito y pueda evolucionar en la empresa.

Finalmente, con el desarrollo de la aplicación se ha podido evidenciar una reducción de tiempos en el proceso de emisión, cobranzas, mejor apreciación de su información para cada cliente, y sobre todo mejora el trabajo en equipo, ya que cada proceso va de la mano de otro.

7.02 Recomendaciones.

Se recomienda que se priorice el uso del sistema con la información de manera adecuada, ya que en caso de que la información no sea ingresada de manera

correcta puede existir contratiempos en los procesos que ocasionen malestar y pérdidas para la empresa.

Se recomienda seguir con el análisis de alternativas para poder implementar los procesos que se encuentran pendientes, con el fin de poder llegar a un resultado del 100% de efectividad que el sistema pueda brindar.

En base a los conocimientos presentados por parte de la carrera de análisis de sistemas es recomendable bridar capacitaciones periódicas a cada área, con el fin de fortalecer las debilidades que puedan presentar en el flujo del manejo del sistema.

Finalmente, como última recomendación se debe tomar en cuenta los tiempos que se emplea para ingresar los datos al sistema, ya que en caso de que no sea ingresados correctos puede presentar problemas en procesos posteriores y conflictos para el manejo de nuevas operaciones.

REFERENCIAS BIBLIOGRÁFICAS

Oracle. Sin fecha. Requisitos de Hardware y software. Recuperado de: https://docs.oracle.com/cd/E19226-01/821-1335/abpaj/index.html

Alfonso Pérez Rodríguez. s.f. Estructuración y Especificación de Casos de Uso. Recuperado de: https://sites.google.com/site/alfonsoperezr/investigacion/estructuracin-y-especificacin-de-casos-de-uos

José María Megino Barquinero. (29 de noviembre de 2013). Especificación detallada de los Casos de Uso. Recuperado de: https://www.seas.es/blog/informatica/especificacion-detallada-de-los-casos-de-uso-uml/

José María Megino Barquinero. (18 de marzo de 2013). Tipos de relaciones en diagramas de casos de uso. Recuperado de: https://www.seas.es/blog/informatica/tipos-de-relaciones-en-diagramas-de-casos-de-uso-uml/

Ingenio Empresa. (17 de febrero 2017). Análisis De Involucrados. Recuperado de: https://ingenioempresa.com/analisis-involucrados-marco-logico/

Ingenio Empresa. (9 de agosto 2016). Árbol de objetivos. Recuperado de: https://ingenioempresa.com/arbol-de-objetivos/

ANEXOS

Anexo A. Entrevista.

ENTREVISTA

Identificador:		
Preguntas	**Objetivos**	**Análisis Posterior**
¿Se registra algún tipo de seguimiento al cotizar un seguro al cliente?	Identificar como se genera el proceso al cotizar a un cliente y ver que seguimiento se realiza	Se desea conocer cómo se lleva el registro de clientes y como actualiza su seguimiento
¿Qué tiempo se demora en el proceso de emisión de una póliza?	Medir el tiempo que se demora el proceso de emisión desde la solicitud por parte del asesor comercial hasta la confirmación de emisión por parte de la compañía de seguros	Se intenta reducir los tiempos de ejecución en los procesos de emisión y verificar hasta qué punto es factible automatizarlos.
¿Antes de emitir una póliza existe algún proceso adicional?	Verificar si al emitir una póliza se realiza algún proceso operativo adicional.	Se identifica que se realiza una solicitud de inspección para que se pueda proceder con la emisión de la póliza, esto debe ser realizado como un seguimiento de emisión.
¿Qué proceso realiza cuando un cliente realiza un pago?	Identificar como se lleva el control de pagos realizados por parte del cliente.	Se intenta identificar como se realiza la aplicación del pago del cliente a las distintas compañías de seguros.
¿Existe algún tipo de notificación al cliente cuando se registra el pago de una cuota?	Conocer si se realiza notificaciones al cliente cuando se aplica el pago a la compañía de seguros	Verificar como se notifica al cliente que su pago fue registrado para que no pierda seguimiento de su seguro.
¿Cómo se notifica a la compañía de seguros que se aplicó un pago por parte del cliente?	Identificar el proceso operativo cuando se realiza él envió de los pagos realizados por clientes a una compañía de seguros	Identificar que se recepta de parte de la compañía de seguros al notificar que se realizó el pago de una cuota por parte del cliente.
¿Cuál es el proceso de preliquidación actualmente?	Conocer claramente que acciones e involucrados intervienen en el proceso de preliquidación	Con la idea clara del proceso que realizan actualmente se puede verificar que aspectos se puede realizar la automatización con un sistema informático
¿Cómo se realiza la verificación del archivo enviado por la compañía de seguros?	Identificar si el proceso es automático o manual, y como garantizan que la información es fiable	Actualmente realizan un cruce manual de lo enviado con la compañía de seguros con el registro de emisión el mismo que se lo maneja de manera manual.
¿Cómo se lleva un detalle de facturas enviadas a las compañías de seguros?	Identificar cual es el proceso que se realiza cuando una factura es enviada a la compañía de seguros y donde se la registra para su seguimiento.	Según lo determinado el proceso lo registran en un documento Excel, según factura enviada a la compañía de seguros, este proceso puede ser automatizado en el sistema a desarrollarse.

¿Cómo realiza el pago de comisiones a los asesores comerciales?	Identificar el proceso que se realiza para realizar el pago de comisiones a los asesores comerciales e identificar que corresponde a cada asesor	Actualmente el pago lo realizan de manera manual, obteniendo la información de cada asesor enviada por un archivo Excel y se realiza la comparación con el detalle de comisiones de la compañía de seguros.
¿Cómo se puede evidenciar lo que se está pagando a cada asesor?	Verificar si existe algún registro de pagos pendientes y pagos realizados de la producción emitida por cada asesor.	Se identifica que no hay un control que se pueda evidenciar, solo se envía el detalle a cada asesor de lo que se le está pagando.
¿Cree que se puede automatizar los procesos por medio de un sistema informático?	Conocer qué proceso fueron se puede automatizar en el bróker definiendo la operativa actual	Al revisar todas las inquietudes del proceso, se puede identificar que actualmente existen varios procesos los que pueden ser automatizados.
¿Qué reportes le gustaría manejar en su sistema informático?	Conocer que reportes son indispensables para el control de la información y notificación a las entidades correspondientes.	Se presenta una variedad de reportes como producción emitida, producción pagada, cuotas pendientes, emisiones pendientes, preliquidaciones, detalle para facturar, comisiones pendientes, facturas cobradas, facturas pendientes y comisiones pagadas.

Nota: Detalle de preguntas realizadas a las distintas áreas de la empresa Qualityseg S.A.

Anexo B. Matriz de Requerimientos.

MATRIZ DE REQUERIMIENTOS						
ID	**Descripción**	**Fuente**	**Prioridad**	**Tipo**	**Estado**	**Involucrados**
REQUERIMIENTOS FUNCIONALES						
RF 001	El sistema deberá permitir el registro de N número de aseguradoras. Cada aseguradora debe tener un ruc, nombre o razón social y un nombre corto.	Jefe de operaciones	Alta	Funcional	Pendiente	Administrador del sistema. Jefe de operaciones. Gerente General
RF 002	El sistema debe permitir registrar ramos, cada ramo tendrá un nombre, un nombre corto y un área.	Jefe de operaciones	Alta	Funcional	Pendiente	Administrador del sistema. Jefe de operaciones. Gerente General
RF 003	El sistema debe permitir registrar áreas. Cada área agrupa a los ramos de acuerdo a su naturaleza.	Jefe de operaciones	Alta	Funcional	Pendiente	Administrador del sistema. Jefe de operaciones. Gerente General
RF 004	El sistema debe permitir registrar el porcentaje de comisión que paga cada compañía de seguros por cada diferente ramo	Jefe de operaciones	Alta	Funcional	Pendiente	Administrador del sistema. Jefe de operaciones. Gerente General
RF 005	El sistema debe permitir registrar N número de subagentes para que puedan asignados a cada cliente	Jefe de operaciones	Alta	Funcional	Pendiente	Administrador del sistema. Jefe de operaciones. Gerente General
RF 006	El sistema debe permitir registrar el porcentaje de comisión a los subagentes, de igual manera se lo debe realizar por ramo	Jefe de operaciones	Alta	Funcional	Pendiente	Administrador del sistema. Jefe de operaciones. Gerente General
RF 007	El sistema deberá permitir el registro de clientes, cada cliente debe manejar un estado	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales

	el cual identifica si es prospecto o contratante					
RF 008	El sistema tendrá que permitir generar cotizaciones. Cada cotización debe permitir seleccionar un cliente o registrarlo colocándolo en estado de prospecto	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 009	El sistema tendrá que agregar uno o varios ramos en la cotización dependiendo el modelo de contrato	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 010	El sistema tendrá que permitir seleccionar una compañía de seguros con la que se está realizando la cotización	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 011	El sistema debe permitir registrar uno o varios objetos asegurados a dependiendo el ramo que se esté utilizando	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 012	Si es el objeto asegurado es del ramo vehículos el sistema debe permitir ingresar el tipo de vehículo o vehículos y sus características como placa, motor, chasis, color, modelo, marca y si dispone dispositivo	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 013	Cada objeto asegurado debe tener un valor comercial por lo que el sistema deberá permitir registrar un valor	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales

	asegurado del cual se realiza el cálculo de la prima del seguro en base a la tasa y vigencia					
RF 014	Una vez ingresado los valores y objetos asegurados, el sistema debe permitir generar una forma de pago para el cliente, dicha forma de pago puede ser tarjeta de crédito, de contado, débitos bancarios y crédito	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 015	Para validar la emisión en el sistema se tendrá que validar el ingreso de numero de factura y numero del contrato del seguro si estos dos campos no se encuentran registrados no constara como emitido	Personal de emisión	Alta	Funcional	Pendiente	Ejecutivos comerciales
RF 016	El sistema debe permitir registrar el cobro de cuotas de pólizas emitidas, en base a la forma de pago registrada, si la póliza no se encuentra emitida no se puede registrar pagos	Departamento de cartera o Cobranzas	Alta	Funcional	Pendiente	Ejecutivo de cobranzas
RF 017	El sistema debe permitir comparar el detalle de comisiones enviado por la compañía de seguros con el detalle de pólizas emitidas y aplicadas el pago, si una póliza no se ha registrado el pago, no se	Personal de Operaciones	Alta	Funcional	Pendiente	Jefe Operativo

	presentará en la lista para comparación					
RF 018	Realizada la comparación con el detalle de comisiones, el sistema debe permitir generar una pre factura, las pre facturas deben ser ligadas a una sola compañía de seguros, no puede existir dos compañías a una pre factura	Personal de Operaciones	Alta	Funcional	Pendiente	Jefe Operativo
RF 019	Una vez generada la o las pre facturas, el sistema debe permitir enviar el detalle que va a ser facturado, él envió de este detalle registra automáticamente un detalle de la factura que va a ser generada	Personal de Operaciones	Alta	Funcional	pendiente	Jefe Operativo
RF 020	Una vez registrada la factura, el sistema debe permitir registrar si la factura fue cobrada con la fecha que se realizó el cobro a la compañía de seguros	Personal de Operaciones	Alta	Funcional	pendiente	Jefe Operativo
RF 021	El sistema debe permitir escoger el ejecutivo comercial al que se le va a realizar el pago de comisiones en base a las pólizas emitidas por dicho asesor comercial	Personal de Operaciones	Alta	Funcional	pendiente	Jefe Operativo
RF 022	Una vez seleccionado el asesor al que se realizara la liquidación, el sistema debe	Personal de Operaciones	Alta	Funcional	Pendiente	Jefe Operativo

	permitir realizar la consulta de las pólizas emitidas por dicho asesor, solo se presentará las pólizas que se encuentren facturadas a la compañía de seguros y que se haya registrado el cobro de la factura					
RF 023	Al presentar el listado, se debe seleccionar los registros que se realizara el pago de comisión al asesor comercial, una vez seleccionado se debe procesar la información	Jefe Operativo	Alta	Funcional	Pendiente	Jefe Operativo
RF 024	Una vez procesada la información, el sistema debe permitir enviar vía correo electrónico al personal de talento humano el detalle que se está liberando la comisión	Jefe operativo	Alta	Funcional	Pendiente	Jefe Operativo Talento Humano

Nota: *La presente matriz muestra los requerimientos funcionales que necesita el sistema para su funcionamiento principal.*

Anexo C. Matriz de requerimientos no Funcionales.

MATRIZ DE REQUERIMIENTOS						
ID	**Descripción**	**Fuente**	**Prioridad**	**Tipo**	**Estado**	**Involucrados**
REQUERIMIENTOS NO FUNCIONALES						
RNF 001	El sistema tiene que controlar el acceso y permitirá solo a usuarios autorizados. Cada usuario debe ingresar al sistema con una identificación de usuario y una contraseña	Administrador sistemas	Alta		Pendiente	Administrador Gerente general. Jefe Operativo. Asesores comerciales. Ejecutivos de cobranzas.
RNF 002	El sistema deberá permitir el registro de usuarios. Cada usuario contara con una identificación de usuario, contraseña, Nombres, Apellidos, Cargo que desempeña, un perfil	Administrador sistema	Alta		Pendiente	Administrador sistema
RNF 003	El sistema debe permitir registrar distintos	Administrador sistema	Alta		Pendiente	Administrador

	perfiles por usuario, A cada perfil se le debe asignar los distintos niveles de acceso				
RNF 004	El sistema debe permitir registrar los datos de la empresa y de ser el caso ingresar las sucursales	Administrador sistema	Alta	Pendiente	Administrador
RNF 005	El sistema debe permitir registrar N número de contactos por compañía de seguros, cada contacto debe contar con Nombres, Apellidos, Correo electrónico y cargo	Jefe operativo	Alta	Pendiente	Jefe Operativo
RNF 006	El sistema debe tener un registro de seguimientos a los clientes que mantengan su estado en prospecto	Asesores Comercial	Media	Pendiente	Asesor Comercial

RNF 007	Para registrar un objeto asegurado el sistema deberá identificar la naturaleza del ramo o a que área pertenece	Jefe Operativo	Alto	Pendiente	Asesor comercial
RNF 008	El sistema debe permitir enviar una solicitud de emisión directamente al contacto de la compañía de seguros mediante correo electrónico con copia al asesor comercial, dicha notificación tendrá un formato y contener toda la información ingresada anteriormente	Asesor comercial	Medio	Pendiente	Asesor Comercial
RNF 009	El sistema debe permitir realizar consultas de las pólizas emitidas, en	Asesor Comercial	Medio	Pendiente	Asesor Comercial Jefe Operativo. Jefe Comercial

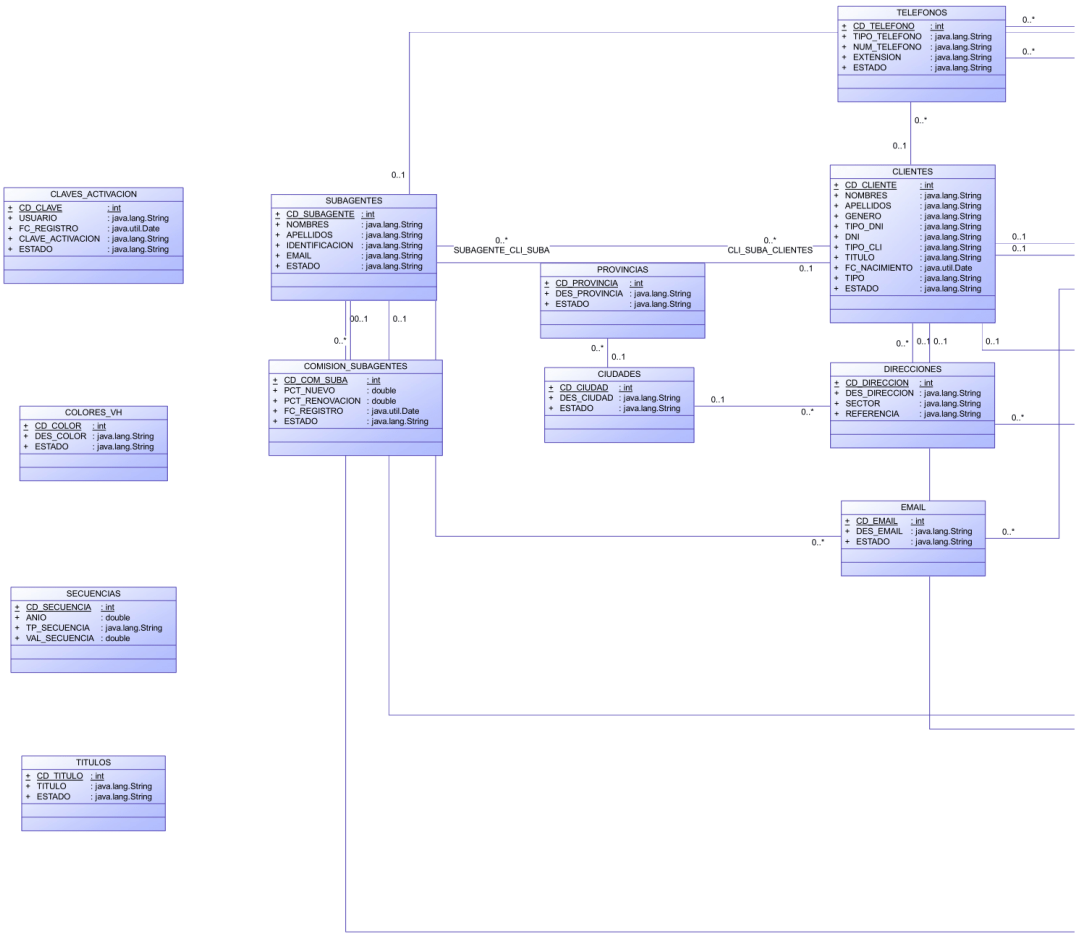
	las que se podrá visualizar los objetos asegurados, vigencias, valores y forma de pago				Gerente General
RNF 010	El sistema debe permitir realizar la reversión de pagos registrados por error, esto solo lo podrá realizar el administrador del sistema o los usuarios autorizados	Ejecutivo de Cobranzas	Alto	Pendiente	Jefe Operativo Administrador Usuarios permitidos
RNF 011	El sistema tendrá una pantalla de consulta de las cuotas pagadas, con el detalle de valor pagado, fecha, usuario que registro el pago, origen de pago, numero de cuota, estado,	Ejecutivo de Cobranzas	Medio	Pendiente	Ejecutivo de cobranzas Jefe operativo Gerente general Ejecutivo comercial

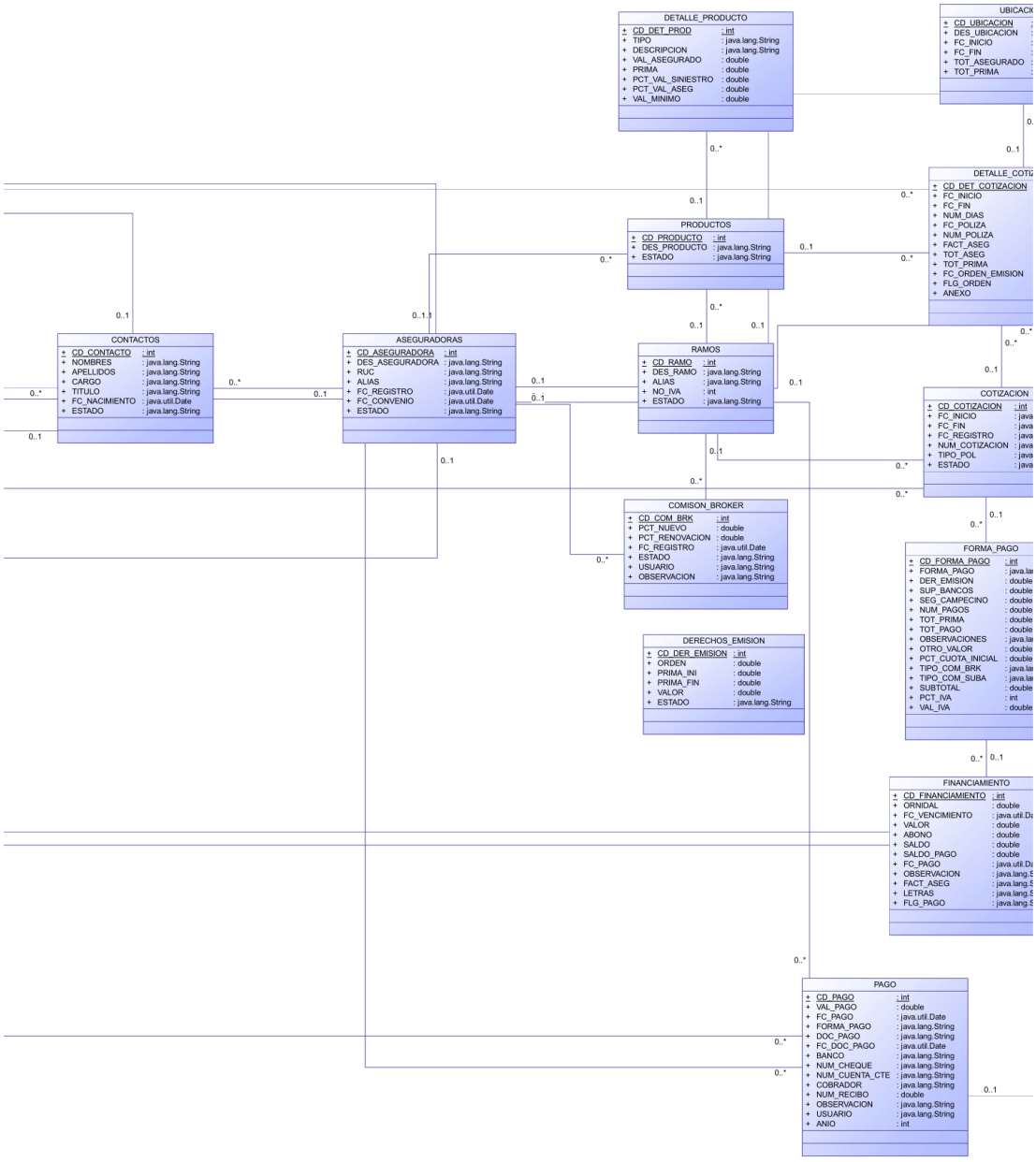
	forma de pago				
RNF 012	El sistema debe permitir realizar un seguimiento a los clientes que tengan cuotas pendientes, en dicho seguimiento se debe registrar fecha de contacto, observaciones y resultado de llamada	Ejecutivo de Cobranzas	Medio	Pendiente	Ejecutivo de cobranza
RNF 013	El sistema debe permitir manejar un secuencial de facturas, dicho secuencial permitirá identificar en el sistema que facturas se emite a las compañías de seguros	Jefe Operativo	Medio	Pendiente	Jefe Operativo Administrador
RNF 014	El sistema debe permitir realizar la consulta de las facturas, tanto cobradas y que se encuentran	Jefe Operativo	Medio	Pendiente	Jefe Operativo

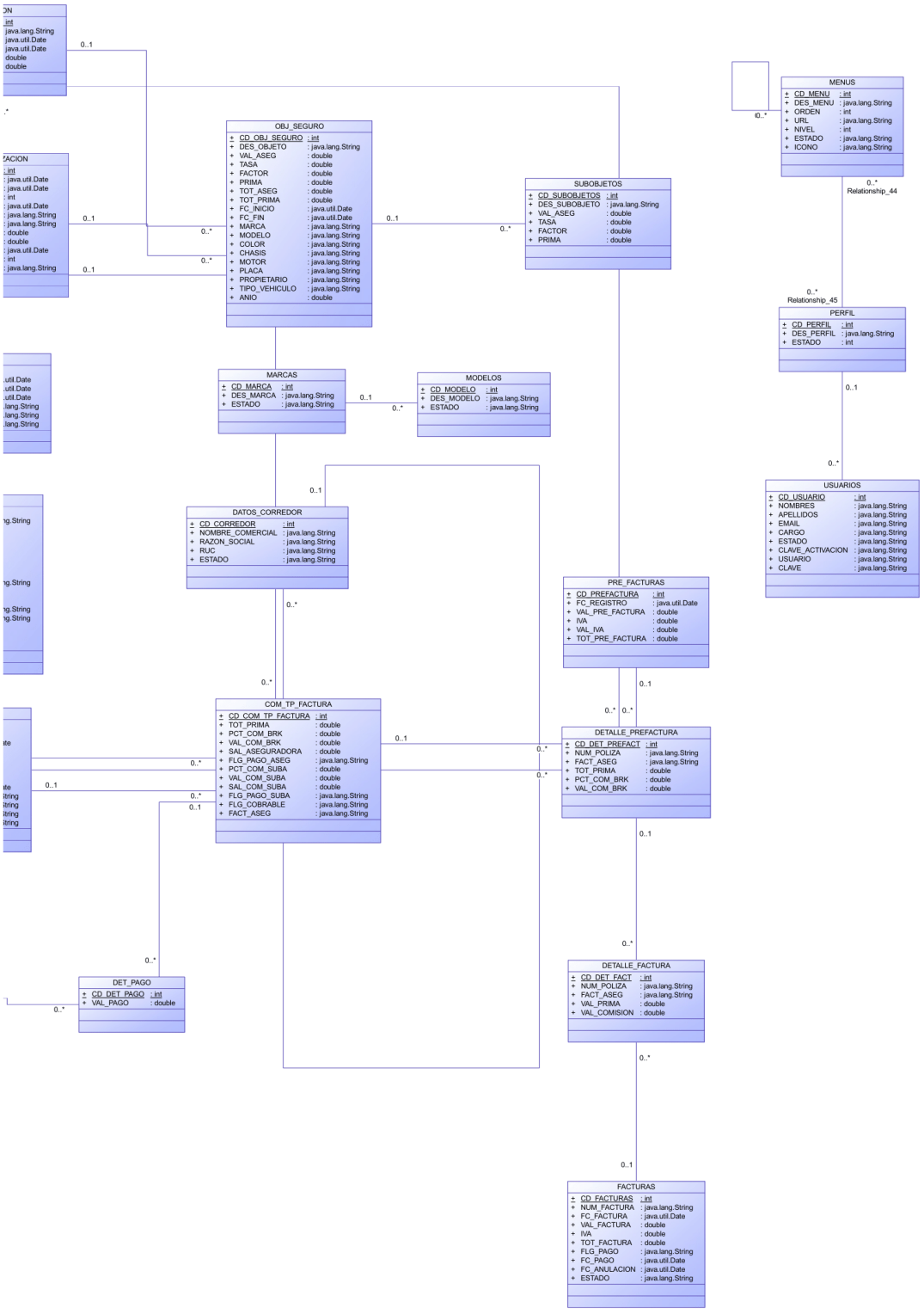
	pendientes de cobro

Nota: Matriz de requerimientos no funcionales, detalla los requerimientos que son necesarios, pero no interrumpen el funcionamiento del sistema.

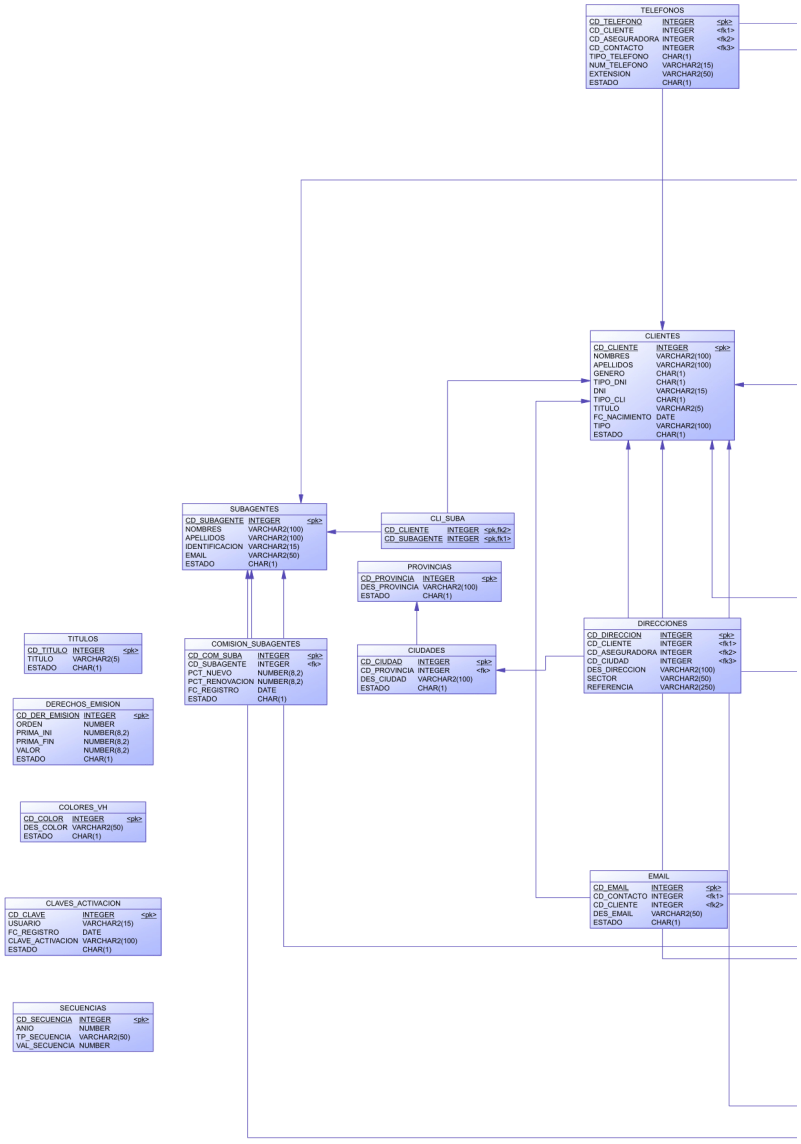
Anexo D. Modelo de Clases.

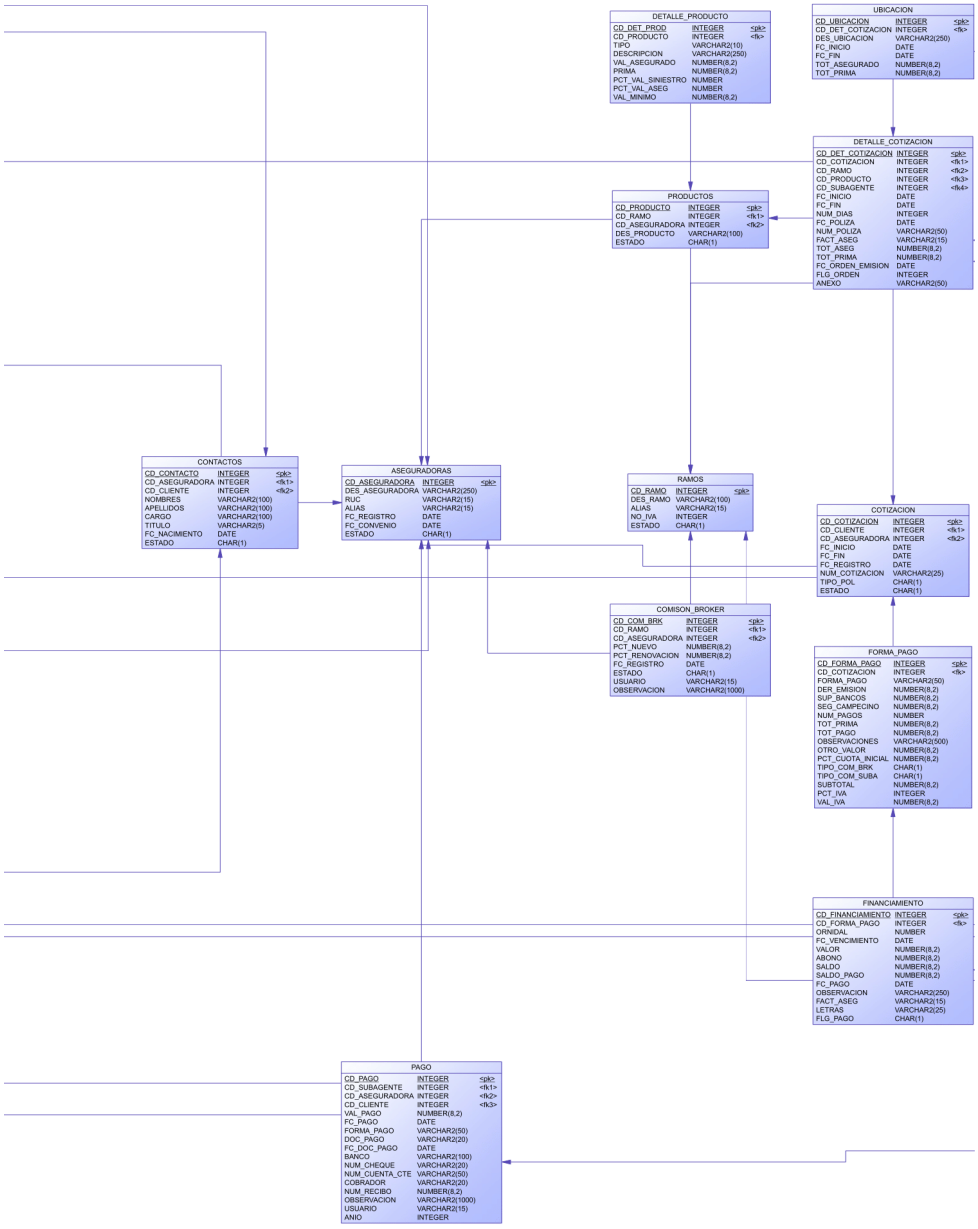


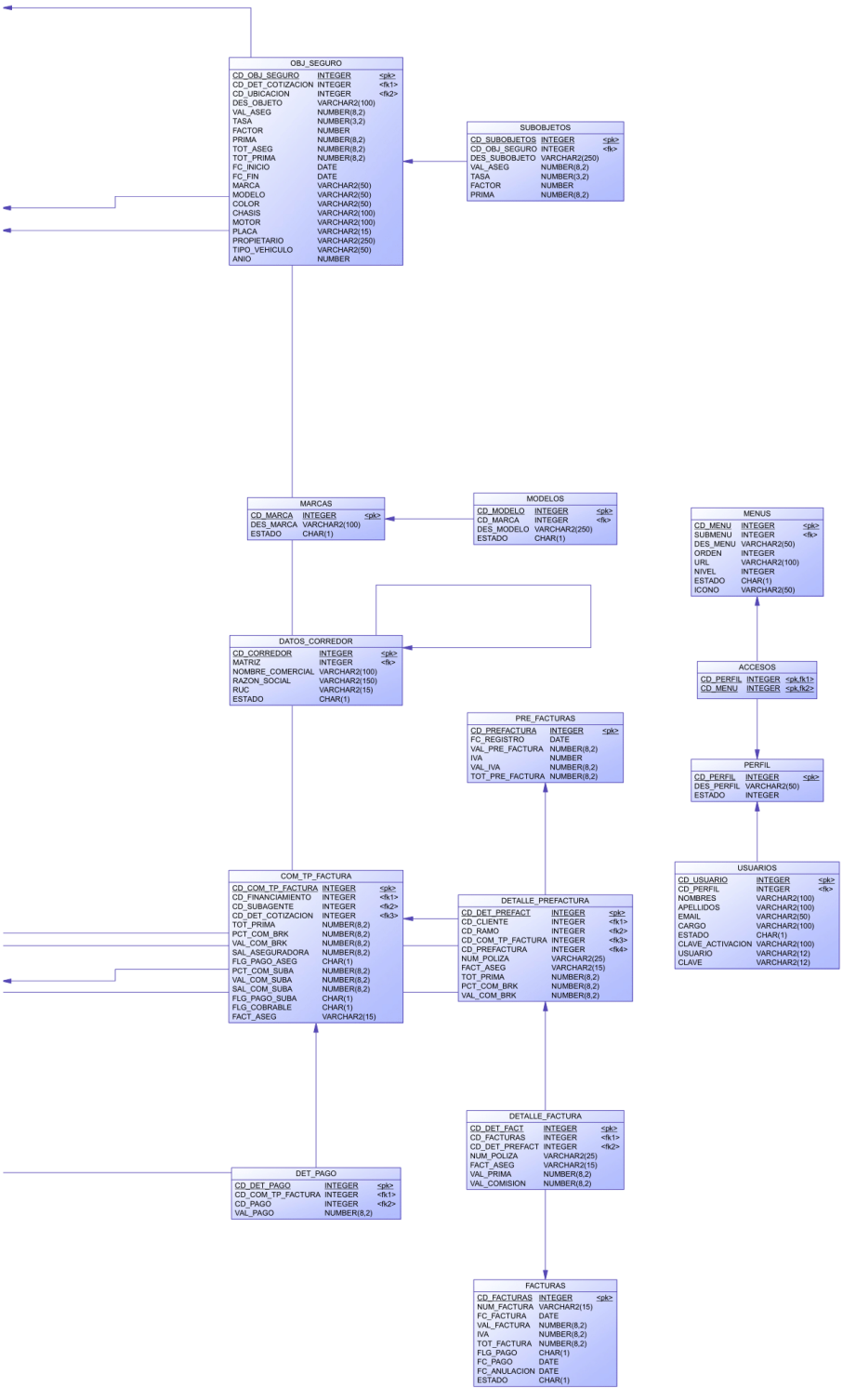

DETALLE_PRODUCTO
+ CD_DET_PROD : int
+ TIPO : java.lang.String
+ DESCRIPCION : java.lang.String
+ VAL_ASEGURADO : double
+ PRIMA : double
+ PCT_VAL_SINIESTRO : double
+ PCT_VAL_ASEG : double
+ VAL_MINIMO : double
PRODUCTOS
+ CD_PRODUCTO : int
+ DES_PRODUCTO : java.lang.String
+ ESTADO : java.lang.String
CONTACTOS
+ CD_CONTACTO : int
+ NOMBRES : java.lang.String
+ APELLIDOS : java.lang.String
+ CARGO : java.lang.String
+ TITULO : java.lang.String
+ FC_NACIMIENTO : java.util.Date
+ ESTADO : java.lang.String
ASEGURADORAS
+ CD_ASEGURADORA : int
+ DES_ASEGURADORA : java.lang.String
+ RUC : java.lang.String
+ ALIAS : java.lang.String
+ FC_REGISTRO : java.util.Date
+ FC_CONVENIO : java.util.Date
+ ESTADO : java.lang.String
RAMOS
+ CD_RAMO : int
+ DES_RAMO : java.lang.String
+ ALIAS : java.lang.String
+ NO_IVA : int
+ ESTADO : java.lang.String
COMISON_BROKER
+ CD_COM_BRK : int
+ PCT_NUEVO : double
+ PCT_RENOVACION : double
+ FC_REGISTRO : java.util.Date
+ ESTADO : java.lang.String
+ USUARIO : java.lang.String
+ OBSERVACION : java.lang.String
DERECHOS_EMISION
+ CD_DER_EMISION : int
+ ORDEN : double
+ PRIMA_INI : double
+ PRIMA_FIN : double
+ VALOR : double
+ ESTADO : java.lang.String
PAGO
+ CD_PAGO : int
+ VAL_PAGO : double
+ FC_PAGO : java.util.Date
+ FORMA_PAGO : java.lang.String
+ DOC_PAGO : java.lang.String
+ FC_DOC_PAGO : java.util.Date
+ BANCO : java.lang.String
+ NUM_CHEQUE : java.lang.String
+ NUM_CUENTA_CTE : java.lang.String
+ COBRADOR : java.lang.String
+ NUM_RECIBO : double
+ OBSERVACION : java.lang.String
+ USUARIO : java.lang.String
+ ANIO : int
FORMA_PAGO
+ CD_FORMA_PAGO : int
+ FORMA_PAGO : java.la
+ DER_EMISION : double
+ SUP_BANCOS : double
+ SEG_CAMPECINO : double
+ NUM_PAGOS : double
+ TOT_PRIMA : double
+ TOT_PAGO : double
+ OBSERVACIONES : java.la
+ OTRO_VALOR : double
+ PCT_CUOTA_INICIAL : double
+ TIPO_COM_BRK : java.la
+ TIPO_COM_SUBA : java.la
+ SUBTOTAL : double
+ PCT_IVA : int
+ VAL_IVA : double
FINANCIAMIENTO
+ CD_FINANCIAMIENTO : int
+ ORNIDAL : double
+ FC_VENCIMIENTO : java.util.Da
+ VALOR : double
+ ABONO : double
+ SALDO : double
+ SALDO_PAGO : double
+ FC_PAGO : java.util.Da
+ OBSERVACION : java.lang.S
+ FACT_ASEG : java.lang.S
+ LETRAS : java.lang.S
+ FLG_PAGO : java.lang.S


OBJ_SEGURO
+ CD_OBJ_SEGURO : int
+ DES_OBJETO : java.lang.String
+ VAL_ASEG : double
+ TASA : double
+ FACTOR : double
+ PRIMA : double
+ TOT_ASEG : double
+ TOT_PRIMA : double
+ FC_INICIO : java.util.Date
+ FC_FIN : java.util.Date
+ MARCA : java.lang.String
+ MODELO : java.lang.String
+ COLOR : java.lang.String
+ CHASIS : java.lang.String
+ MOTOR : java.lang.String
+ PLACA : java.lang.String
+ PROPIETARIO : java.lang.String
+ TIPO_VEHICULO : java.lang.String
+ ANIO : double
SUBOBJETOS
+ CD_SUBOBJETOS : int
+ DES_SUBOBJETO : java.lang.String
+ VAL_ASEG : double
+ TASA : double
+ FACTOR : double
+ PRIMA : double
MENUS
+ CD_MENU : int
+ DES_MENU : java.lang.String
+ ORDEN : int
+ URL : java.lang.String
+ NIVEL : int
+ ESTADO : java.lang.String
+ ICONO : java.lang.String
Relationship_44
Relationship_45
PERFIL
+ CD_PERFIL : int
+ DES_PERFIL : java.lang.String
+ ESTADO : int
USUARIOS
+ CD_USUARIO : int
+ NOMBRES : java.lang.String
+ APELLIDOS : java.lang.String
+ EMAIL : java.lang.String
+ CARGO : java.lang.String
+ ESTADO : java.lang.String
+ CLAVE_ACTIVACION : java.lang.String
+ USUARIO : java.lang.String
+ CLAVE : java.lang.String
MARCAS
+ CD_MARCA : int
+ DES_MARCA : java.lang.String
+ ESTADO : java.lang.String
MODELOS
+ CD_MODELO : int
+ DES_MODELO : java.lang.String
+ ESTADO : java.lang.String
DATOS_CORREDOR
+ CD_CORREDOR : int
+ NOMBRE_COMERCIAL : java.lang.String
+ RAZON_SOCIAL : java.lang.String
+ RUC : java.lang.String
+ ESTADO : java.lang.String
PRE_FACTURAS
+ CD_PREFACTURA : int
+ FC_REGISTRO : java.util.Date
+ VAL_PRE_FACTURA : double
+ IVA : double
+ VAL_IVA : double
+ TOT_PRE_FACTURA : double
COM_TP_FACTURA
+ CD_COM_TP_FACTURA : int
+ TOT_PRIMA : double
+ PCT_COM_BRK : double
+ VAL_COM_BRK : double
+ SAL_ASEGURADORA : double
+ FLG_PAGO_ASEG : java.lang.String
+ PCT_COM_SUBA : double
+ VAL_COM_SUBA : double
+ SAL_COM_SUBA : double
+ FLG_PAGO_SUBA : java.lang.String
+ FLG_COBRABLE : java.lang.String
+ FACT_ASEG : java.lang.String
DETALLE_PREFACTURA
+ CD_DET_PREFACT : int
+ NUM_POLIZA : java.lang.String
+ FACT_ASEG : java.lang.String
+ TOT_PRIMA : double
+ PCT_COM_BRK : double
+ VAL_COM_BRK : double
DET_PAGO
+ CD_DET_PAGO : int
+ VAL_PAGO : double
DETALLE_FACTURA
+ CD_DET_FACT : int
+ NUM_POLIZA : java.lang.String
+ FACT_ASEG : java.lang.String
+ VAL_PRIMA : double
+ VAL_COMISION : double
FACTURAS
+ CD_FACTURAS : int
+ NUM_FACTURA : java.lang.String
+ FC_FACTURA : java.util.Date
+ VAL_FACTURA : double
+ IVA : double
+ TOT_FACTURA : double
+ FLG_PAGO : java.lang.String
+ FC_PAGO : java.util.Date
+ FC_ANULACION : java.util.Date
+ ESTADO : java.lang.String

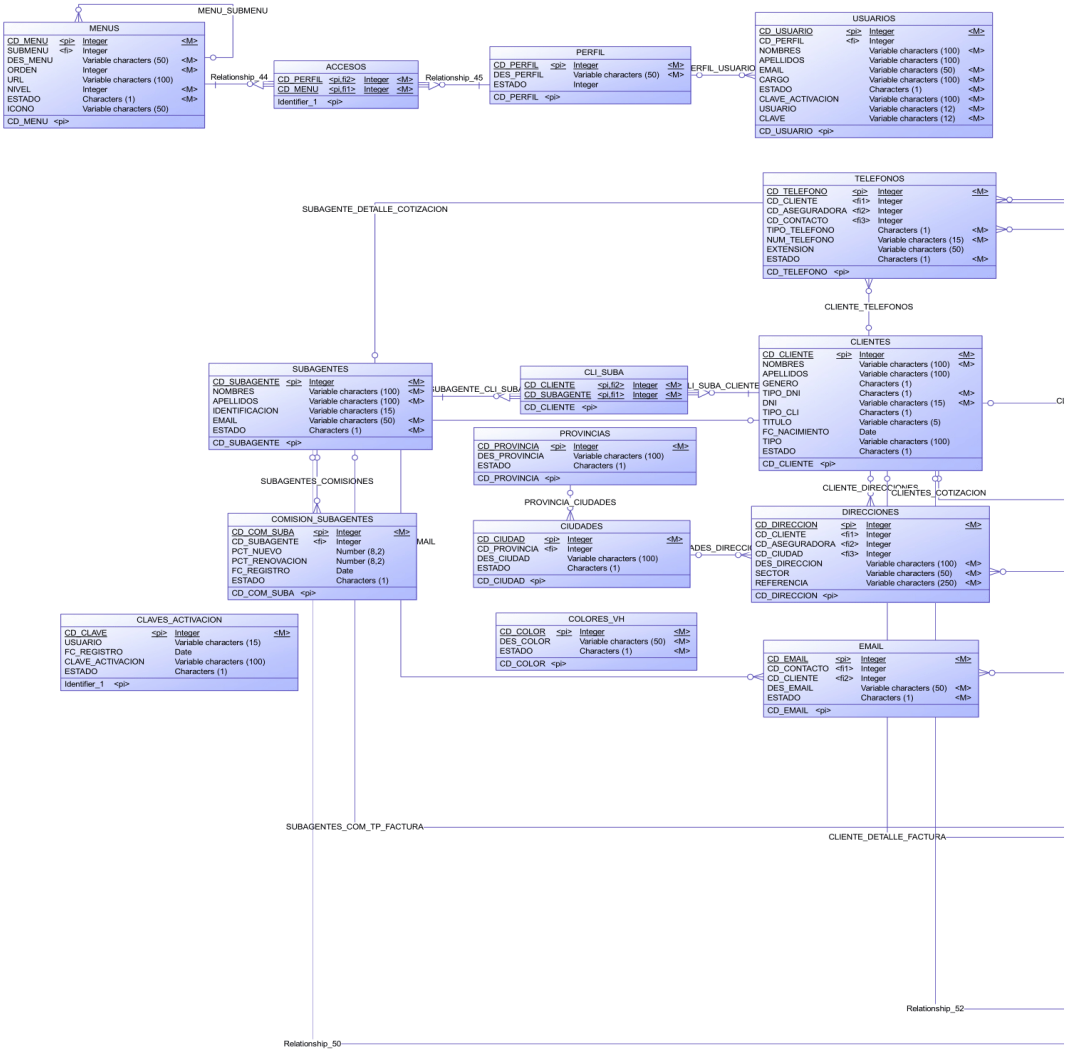
Anexo E. Modelo Físico.

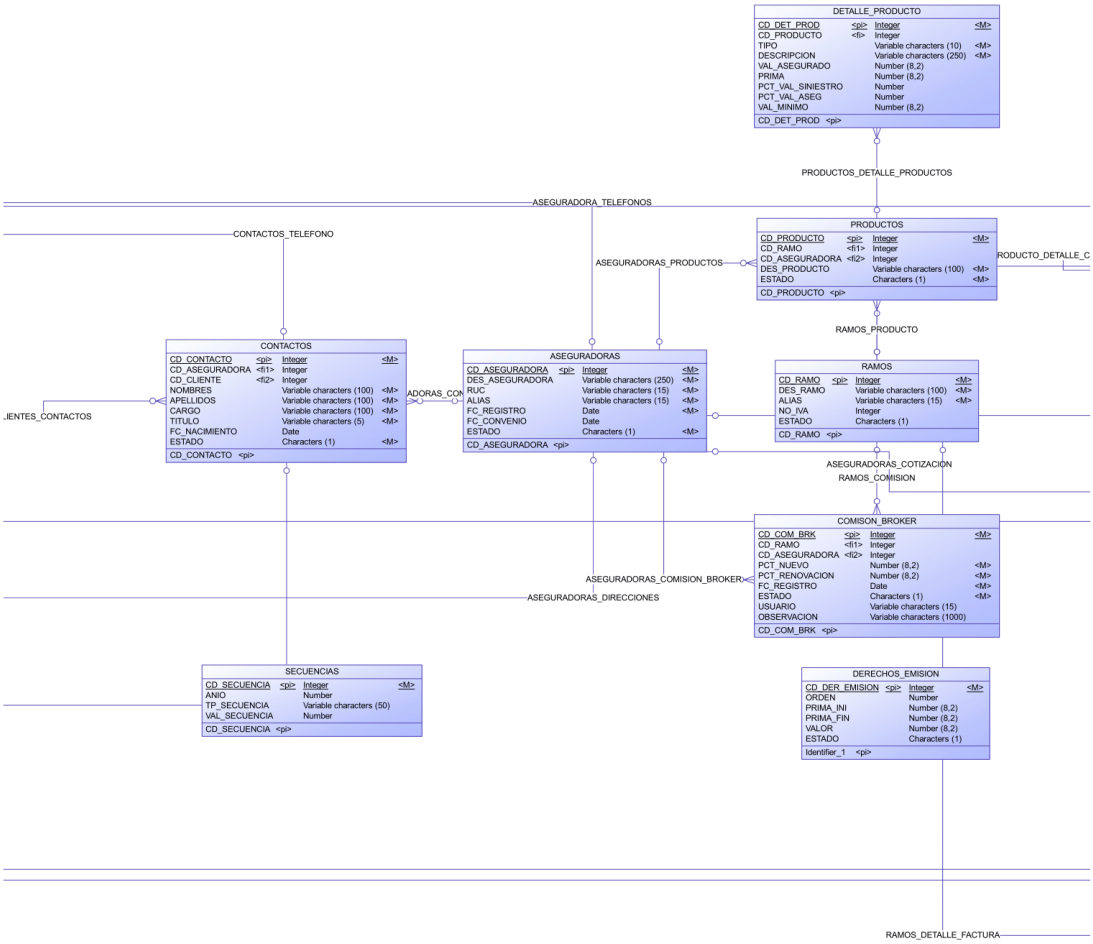


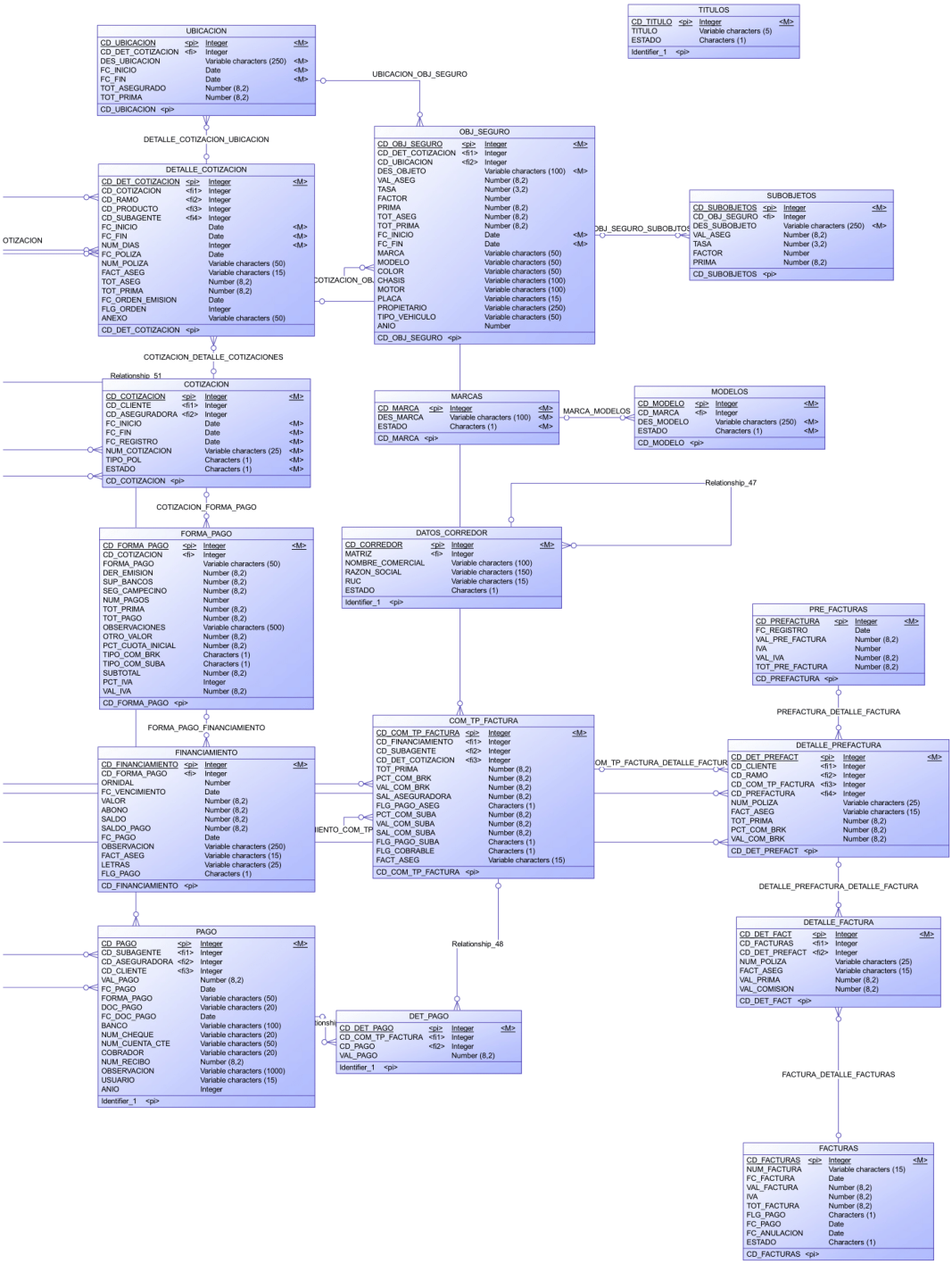

DETALLE_PRODUCTO
CD_DET_PROD INTEGER <pk>
CD_PRODUCTO INTEGER <fk>
TIPO VARCHAR2(10)
DESCRIPCION VARCHAR2(250)
VAL_ASEGURADO NUMBER(8,2)
PRIMA NUMBER(8,2)
PCT_VAL_SINIESTRO NUMBER
PCT_VAL_ASEG NUMBER
VAL_MINIMO NUMBER(8,2)
UBICACION
CD_UBICACION INTEGER <pk>
CD_DET_COTIZACION INTEGER <fk>
DES_UBICACION VARCHAR2(250)
FC_INICIO DATE
FC_FIN DATE
TOT_ASEGURADO NUMBER(8,2)
TOT_PRIMA NUMBER(8,2)
DETALLE_COTIZACION
CD_DET_COTIZACION INTEGER <pk>
CD_COTIZACION INTEGER <fk1>
CD_RAMO INTEGER <fk2>
CD_PRODUCTO INTEGER <fk3>
CD_SUBAGENTE INTEGER <fk4>
FC_INICIO DATE
FC_FIN DATE
NUM_DIAS INTEGER
FC_POLIZA DATE
NUM_POLIZA VARCHAR2(50)
FACT_ASEG VARCHAR2(15)
TOT_ASEG NUMBER(8,2)
TOT_PRIMA NUMBER(8,2)
FC_ORDEN_EMISION DATE
FLG_ORDEN INTEGER
ANEXO VARCHAR2(50)
PRODUCTOS
CD_PRODUCTO INTEGER <pk>
CD_RAMO INTEGER <fk1>
CD_ASEGURADORA INTEGER <fk2>
DES_PRODUCTO VARCHAR2(100)
ESTADO CHAR(1)
CONTACTOS
CD_CONTACTO INTEGER <pk>
CD_ASEGURADORA INTEGER <fk1>
CD_CLIENTE INTEGER <fk2>
NOMBRES VARCHAR2(100)
APELLIDOS VARCHAR2(100)
CARGO VARCHAR2(100)
TITULO VARCHAR2(5)
FC_NACIMIENTO DATE
ESTADO CHAR(1)
ASEGURADORAS
CD_ASEGURADORA INTEGER <pk>
DES_ASEGURADORA VARCHAR2(250)
RUC VARCHAR2(15)
ALIAS VARCHAR2(15)
FC_REGISTRO DATE
FC_CONVENIO DATE
ESTADO CHAR(1)
RAMOS
CD_RAMO INTEGER <pk>
DES_RAMO VARCHAR2(100)
ALIAS VARCHAR2(15)
NO_IVA INTEGER
ESTADO CHAR(1)
COTIZACION
CD_COTIZACION INTEGER <pk>
CD_CLIENTE INTEGER <fk1>
CD_ASEGURADORA INTEGER <fk2>
FC_INICIO DATE
FC_FIN DATE
FC_REGISTRO DATE
NUM_COTIZACION VARCHAR2(25)
TIPO_POL CHAR(1)
ESTADO CHAR(1)
COMISON_BROKER
CD_COM_BRK INTEGER <pk>
CD_RAMO INTEGER <fk1>
CD_ASEGURADORA INTEGER <fk2>
PCT_NUEVO NUMBER(8,2)
PCT_RENOVACION NUMBER(8,2)
FC_REGISTRO DATE
ESTADO CHAR(1)
USUARIO VARCHAR2(15)
OBSERVACION VARCHAR2(1000)
FORMA_PAGO
CD_FORMA_PAGO INTEGER <pk>
CD_COTIZACION INTEGER <fk>
FORMA_PAGO VARCHAR2(50)
DER_EMISION NUMBER(8,2)
SUP_BANCOS NUMBER(8,2)
SEG_CAMPECINO NUMBER(8,2)
NUM_PAGOS NUMBER
TOT_PRIMA NUMBER(8,2)
TOT_PAGO NUMBER(8,2)
OBSERVACIONES VARCHAR2(500)
OTRO_VALOR NUMBER(8,2)
PCT_CUOTA_INICIAL NUMBER(8,2)
TIPO_COM_BRK CHAR(1)
TIPO_COM_SUBA CHAR(1)
SUBTOTAL NUMBER(8,2)
PCT_IVA INTEGER
VAL_IVA NUMBER(8,2)
FINANCIAMIENTO
CD_FINANCIAMIENTO INTEGER <pk>
CD_FORMA_PAGO INTEGER <fk>
ORNIDAL NUMBER
FC_VENCIMIENTO DATE
VALOR NUMBER(8,2)
ABONO NUMBER(8,2)
SALDO NUMBER(8,2)
SALDO_PAGO NUMBER(8,2)
FC_PAGO DATE
OBSERVACION VARCHAR2(250)
FACT_ASEG VARCHAR2(15)
LETRAS VARCHAR2(25)
FLG_PAGO CHAR(1)
PAGO
CD_PAGO INTEGER <pk>
CD_SUBAGENTE INTEGER <fk1>
CD_ASEGURADORA INTEGER <fk2>
CD_CLIENTE INTEGER <fk3>
VAL_PAGO NUMBER(8,2)
FC_PAGO DATE
FORMA_PAGO VARCHAR2(50)
DOC_PAGO VARCHAR2(20)
FC_DOC_PAGO DATE
BANCO VARCHAR2(100)
NUM_CHEQUE VARCHAR2(20)
NUM_CUENTA_CTE VARCHAR2(50)
COBRADOR VARCHAR2(20)
NUM_RECIBO NUMBER(8,2)
OBSERVACION VARCHAR2(1000)
USUARIO VARCHAR2(15)
ANIO INTEGER


OBJ_SEGURO
CD_OBJ_SEGURO INTEGER <pk>
CD_DET_COTIZACION INTEGER <fk1>
CD_UBICACION INTEGER <fk2>
DES_OBJETO VARCHAR2(100)
VAL_ASEG NUMBER(8,2)
TASA NUMBER(3,2)
FACTOR NUMBER
PRIMA NUMBER(8,2)
TOT_ASEG NUMBER(8,2)
TOT_PRIMA NUMBER(8,2)
FC_INICIO DATE
FC_FIN DATE
MARCA VARCHAR2(50)
MODELO VARCHAR2(50)
COLOR VARCHAR2(50)
CHASIS VARCHAR2(100)
MOTOR VARCHAR2(100)
PLACA VARCHAR2(15)
PROPIETARIO VARCHAR2(250)
TIPO_VEHICULO VARCHAR2(50)
ANIO NUMBER
SUBOBJETOS
CD_SUBOBJETOS INTEGER <pk>
CD_OBJ_SEGURO INTEGER <fk>
DES_SUBOBJETO VARCHAR2(250)
VAL_ASEG NUMBER(8,2)
TASA NUMBER(3,2)
FACTOR NUMBER
PRIMA NUMBER(8,2)
MARCAS
CD_MARCA INTEGER <pk>
DES_MARCA VARCHAR2(100)
ESTADO CHAR(1)
MODELOS
CD_MODELO INTEGER <pk>
CD_MARCA INTEGER <fk>
DES_MODELO VARCHAR2(250)
ESTADO CHAR(1)
MENUS
CD_MENU INTEGER <pk>
SUBMENU INTEGER <fk>
DES_MENU VARCHAR2(50)
ORDEN INTEGER
URL VARCHAR2(100)
NIVEL INTEGER
ESTADO CHAR(1)
ICONO VARCHAR2(50)
DATOS_CORREDOR
CD_CORREDOR INTEGER <pk>
MATRIZ INTEGER <fk>
NOMBRE_COMERCIAL VARCHAR2(100)
RAZON_SOCIAL VARCHAR2(150)
RUC VARCHAR2(15)
ESTADO CHAR(1)
ACCESOS
CD_PERFIL INTEGER <pk,fk1>
CD_MENU INTEGER <pk,fk2>
PRE_FACTURAS
CD_PREFACTURA INTEGER <pk>
FC_REGISTRO DATE
VAL_PRE_FACTURA NUMBER(8,2)
IVA NUMBER
VAL_IVA NUMBER(8,2)
TOT_PRE_FACTURA NUMBER(8,2)
PERFIL
CD_PERFIL INTEGER <pk>
DES_PERFIL VARCHAR2(50)
ESTADO INTEGER
USUARIOS
CD_USUARIO INTEGER <pk>
CD_PERFIL INTEGER <fk>
NOMBRES VARCHAR2(100)
APELLIDOS VARCHAR2(100)
EMAIL VARCHAR2(50)
CARGO VARCHAR2(100)
ESTADO CHAR(1)
CLAVE_ACTIVACION VARCHAR2(100)
USUARIO VARCHAR2(12)
CLAVE VARCHAR2(12)
COM_TP_FACTURA
CD_COM_TP_FACTURA INTEGER <pk>
CD_FINANCIAMIENTO INTEGER <fk1>
CD_SUBAGENTE INTEGER <fk2>
CD_DET_COTIZACION INTEGER <fk3>
TOT_PRIMA NUMBER(8,2)
PCT_COM_BRK NUMBER(8,2)
VAL_COM_BRK NUMBER(8,2)
SAL_ASEGURADORA NUMBER(8,2)
FLG_PAGO_ASEG CHAR(1)
PCT_COM_SUBA NUMBER(8,2)
VAL_COM_SUBA NUMBER(8,2)
SAL_COM_SUBA NUMBER(8,2)
FLG_PAGO_SUBA CHAR(1)
FLG_COBRABLE CHAR(1)
FACT_ASEG VARCHAR2(15)
DETALLE_PREFACTURA
CD_DET_PREFACT INTEGER <pk>
CD_CLIENTE INTEGER <fk1>
CD_RAMO INTEGER <fk2>
CD_COM_TP_FACTURA INTEGER <fk3>
CD_PREFACTURA INTEGER <fk4>
NUM_POLIZA VARCHAR2(25)
FACT_ASEG VARCHAR2(15)
TOT_PRIMA NUMBER(8,2)
PCT_COM_BRK NUMBER(8,2)
VAL_COM_BRK NUMBER(8,2)
DETALLE_FACTURA
CD_DET_FACT INTEGER <pk>
CD_FACTURAS INTEGER <fk1>
CD_DET_PREFACT INTEGER <fk2>
NUM_POLIZA VARCHAR2(25)
FACT_ASEG VARCHAR2(15)
VAL_PRIMA NUMBER(8,2)
VAL_COMISION NUMBER(8,2)
DET_PAGO
CD_DET_PAGO INTEGER <pk>
CD_COM_TP_FACTURA INTEGER <fk1>
CD_PAGO INTEGER <fk2>
VAL_PAGO NUMBER(8,2)
FACTURAS
CD_FACTURAS INTEGER <pk>
NUM_FACTURA VARCHAR2(15)
FC_FACTURA DATE
VAL_FACTURA NUMBER(8,2)
IVA NUMBER(8,2)
TOT_FACTURA NUMBER(8,2)
FLG_PAGO CHAR(1)
FC_PAGO DATE
FC_ANULACION DATE
ESTADO CHAR(1)

Anexo F. Modelo Lógico.




DETALLE_PRODUCTO
CD_DET_PROD <pi> Integer <M>
CD_PRODUCTO <fi> Integer
TIPO Variable characters (10) <M>
DESCRIPCION Variable characters (250) <M>
VAL_ASEGURADO Number (8,2)
PRIMA Number (8,2)
PCT_VAL_SINIESTRO Number
PCT_VAL_ASEG Number
VAL_MINIMO Number (8,2)
CD_DET_PROD <pi>
PRODUCTOS_DETALLE_PRODUCTOS
ASEGURADORA_TELEFONOS
CONTACTOS_TELEFONO
PRODUCTOS
CD_PRODUCTO <pi> Integer <M>
CD_RAMO <fi1> Integer
CD_ASEGURADORA <fi2> Integer
DES_PRODUCTO Variable characters (100) <M>
ESTADO Characters (1) <M>
CD_PRODUCTO <pi>
ASEGURADORAS_PRODUCTOS
RODUCTO_DETALLE_C
RAMOS_PRODUCTO
CONTACTOS
CD_CONTACTO <pi> Integer <M>
CD_ASEGURADORA <fi1> Integer
CD_CLIENTE <fi2> Integer
NOMBRES Variable characters (100) <M>
APELLIDOS Variable characters (100) <M>
CARGO Variable characters (100) <M>
TITULO Variable characters (5) <M>
FC_NACIMIENTO Date
ESTADO Characters (1) <M>
CD_CONTACTO <pi>
IENTES_CONTACTOS
ADORAS_CON
ASEGURADORAS
CD_ASEGURADORA <pi> Integer <M>
DES_ASEGURADORA Variable characters (250) <M>
RUC Variable characters (15) <M>
ALIAS Variable characters (15) <M>
FC_REGISTRO Date <M>
FC_CONVENIO Date
ESTADO Characters (1) <M>
CD_ASEGURADORA <pi>
RAMOS
CD_RAMO <pi> Integer <M>
DES_RAMO Variable characters (100) <M>
ALIAS Variable characters (15) <M>
NO_IVA Integer
ESTADO Characters (1)
CD_RAMO <pi>
ASEGURADORAS_COTIZACION
RAMOS_COMISION
COMISON_BROKER
CD_COM_BRK <pi> Integer <M>
CD_RAMO <fi1> Integer
CD_ASEGURADORA <fi2> Integer
PCT_NUEVO Number (8,2) <M>
PCT_RENOVACION Number (8,2) <M>
FC_REGISTRO Date <M>
ESTADO Characters (1) <M>
USUARIO Variable characters (15)
OBSERVACION Variable characters (1000)
CD_COM_BRK <pi>
ASEGURADORAS_COMISION_BROKER
ASEGURADORAS_DIRECCIONES
SECUENCIAS
CD_SECUENCIA <pi> Integer <M>
ANIO Number
TP_SECUENCIA Variable characters (50)
VAL_SECUENCIA Number
CD_SECUENCIA <pi>
DERECHOS_EMISION
CD_DER_EMISION <pi> Integer <M>
ORDEN Number
PRIMA_INI Number (8,2)
PRIMA_FIN Number (8,2)
VALOR Number (8,2)
ESTADO Characters (1)
Identifier_1 <pi>
RAMOS_DETALLE_FACTURA


TITULOS
CD_TITULO <pi> Integer <M>
TITULO Variable characters (5)
ESTADO Characters (1)
Identifier_1 <pi>
UBICACION
CD_UBICACION <pi> Integer <M>
CD_DET_COTIZACION <fi> Integer
DES_UBICACION Variable characters (250) <M>
FC_INICIO Date <M>
FC_FIN Date <M>
TOT_ASEGURADO Number (8,2)
TOT_PRIMA Number (8,2)
CD_UBICACION <pi>
UBICACION_OBJ_SEGURO
DETALLE_COTIZACION_UBICACION
OBJ_SEGURO
CD_OBJ_SEGURO <pi> Integer <M>
CD_DET_COTIZACION <fi1> Integer
CD_UBICACION <fi2> Integer
DES_OBJETO Variable characters (100) <M>
VAL_ASEG Number (8,2)
TASA Number (3,2)
FACTOR Number
PRIMA Number (8,2)
TOT_ASEG Number (8,2)
TOT_PRIMA Number (8,2)
FC_INICIO Date <M>
FC_FIN Date <M>
MARCA Variable characters (50)
MODELO Variable characters (50)
COLOR Variable characters (50)
CHASIS Variable characters (100)
MOTOR Variable characters (100)
PLACA Variable characters (15)
PROPIETARIO Variable characters (250)
TIPO_VEHICULO Variable characters (50)
ANIO Number
CD_OBJ_SEGURO <pi>
SUBOBJETOS
CD_SUBOBJETOS <pi> Integer <M>
CD_OBJ_SEGURO <fi> Integer
DES_SUBOBJETO Variable characters (250) <M>
VAL_ASEG Number (8,2)
TASA Number (3,2)
FACTOR Number
PRIMA Number (8,2)
CD_SUBOBJETOS <pi>
OBJ_SEGURO_SUBOBJTOS
DETALLE_COTIZACION
CD_DET_COTIZACION <pi> Integer <M>
CD_COTIZACION <fi1> Integer
CD_RAMO <fi2> Integer
CD_PRODUCTO <fi3> Integer
CD_SUBAGENTE <fi4> Integer
FC_INICIO Date <M>
FC_FIN Date <M>
NUM_DIAS Integer <M>
FC_POLIZA Date
NUM_POLIZA Variable characters (50)
FACT_ASEG Variable characters (15)
TOT_ASEG Number (8,2)
TOT_PRIMA Number (8,2)
FC_ORDEN_EMISION Date
FLG_ORDEN Integer
ANEXO Variable characters (50)
CD_DET_COTIZACION <pi>
OTIZACION
COTIZACION_OBJ
COTIZACION_DETALLE_COTIZACIONES
Relationship_51
COTIZACION
CD_COTIZACION <pi> Integer <M>
CD_CLIENTE <fi1> Integer
CD_ASEGURADORA <fi2> Integer
FC_INICIO Date <M>
FC_FIN Date <M>
FC_REGISTRO Date <M>
NUM_COTIZACION Variable characters (25) <M>
TIPO_POL Characters (1) <M>
ESTADO Characters (1) <M>
CD_COTIZACION <pi>
MARCAS
CD_MARCA <pi> Integer <M>
DES_MARCA Variable characters (100) <M>
ESTADO Characters (1) <M>
CD_MARCA <pi>
MARCA_MODELOS
MODELOS
CD_MODELO <pi> Integer <M>
CD_MARCA <fi> Integer
DES_MODELO Variable characters (250) <M>
ESTADO Characters (1) <M>
CD_MODELO <pi>
Relationship_47
COTIZACION_FORMA_PAGO
FORMA_PAGO
CD_FORMA_PAGO <pi> Integer <M>
CD_COTIZACION <fi> Integer
FORMA_PAGO Variable characters (50)
DER_EMISION Number (8,2)
SUP_BANCOS Number (8,2)
SEG_CAMPECINO Number (8,2)
NUM_PAGOS Number
TOT_PRIMA Number (8,2)
TOT_PAGO Number (8,2)
OBSERVACIONES Variable characters (500)
OTRO_VALOR Number (8,2)
PCT_CUOTA_INICIAL Number (8,2)
TIPO_COM_BRK Characters (1)
TIPO_COM_SUBA Characters (1)
SUBTOTAL Number (8,2)
PCT_IVA Integer
VAL_IVA Number (8,2)
CD_FORMA_PAGO <pi>
DATOS_CORREDOR
CD_CORREDOR <pi> Integer <M>
MATRIZ <fi> Integer
NOMBRE_COMERCIAL Variable characters (100)
RAZON_SOCIAL Variable characters (150)
RUC Variable characters (15)
ESTADO Characters (1)
Identifier_1 <pi>
PRE_FACTURAS
CD_PREFACTURA <pi> Integer <M>
FC_REGISTRO Date
VAL_PRE_FACTURA Number (8,2)
IVA Number
VAL_IVA Number (8,2)
TOT_PRE_FACTURA Number (8,2)
CD_PREFACTURA <pi>
PREFACTURA_DETALLE_FACTURA
FORMA_PAGO_FINANCIAMIENTO
FINANCIAMIENTO
CD_FINANCIAMIENTO <pi> Integer <M>
CD_FORMA_PAGO <fi> Integer
ORNIDAL Number
FC_VENCIMIENTO Date
VALOR Number (8,2)
ABONO Number (8,2)
SALDO Number (8,2)
SALDO_PAGO Number (8,2)
FC_PAGO Date
OBSERVACION Variable characters (250)
FACT_ASEG Variable characters (15)
LETRAS Variable characters (25)
FLG_PAGO Characters (1)
CD_FINANCIAMIENTO <pi>
COM_TP_FACTURA
CD_COM_TP_FACTURA <pi> Integer <M>
CD_FINANCIAMIENTO <fi1> Integer
CD_SUBAGENTE <fi2> Integer
CD_DET_COTIZACION <fi3> Integer
TOT_PRIMA Number (8,2)
PCT_COM_BRK Number (8,2)
VAL_COM_BRK Number (8,2)
SAL_ASEGURADORA Number (8,2)
FLG_PAGO_ASEG Characters (1)
PCT_COM_SUBA Number (8,2)
VAL_COM_SUBA Number (8,2)
SAL_COM_SUBA Number (8,2)
FLG_PAGO_SUBA Characters (1)
FLG_COBRABLE Characters (1)
FACT_ASEG Variable characters (15)
CD_COM_TP_FACTURA <pi>
OM_TP_FACTURA_DETALLE_FACTUR
DETALLE_PREFACTURA
CD_DET_PREFACT <pi> Integer <M>
CD_CLIENTE <fi1> Integer
CD_RAMO <fi2> Integer
CD_COM_TP_FACTURA <fi3> Integer
CD_PREFACTURA <fi4> Integer
NUM_POLIZA Variable characters (25)
FACT_ASEG Variable characters (15)
TOT_PRIMA Number (8,2)
PCT_COM_BRK Number (8,2)
VAL_COM_BRK Number (8,2)
CD_DET_PREFACT <pi>
DETALLE_PREFACTURA_DETALLE_FACTURA
DETALLE_FACTURA
CD_DET_FACT <pi> Integer <M>
CD_FACTURAS <fi1> Integer
CD_DET_PREFACT <fi2> Integer
NUM_POLIZA Variable characters (25)
FACT_ASEG Variable characters (15)
VAL_PRIMA Number (8,2)
VAL_COMISION Number (8,2)
CD_DET_FACT <pi>
FACTURA_DETALLE_FACTURAS
FACTURAS
CD_FACTURAS <pi> Integer <M>
NUM_FACTURA Variable characters (15)
FC_FACTURA Date
VAL_FACTURA Number (8,2)
IVA Number (8,2)
TOT_FACTURA Number (8,2)
FLG_PAGO Characters (1)
FC_PAGO Date
FC_ANULACION Date
ESTADO Characters (1)
CD_FACTURAS <pi>
IENTO_COM_TP
PAGO
CD_PAGO <pi> Integer <M>
CD_SUBAGENTE <fi1> Integer
CD_ASEGURADORA <fi2> Integer
CD_CLIENTE <fi3> Integer
VAL_PAGO Number (8,2)
FC_PAGO Date
FORMA_PAGO Variable characters (50)
DOC_PAGO Variable characters (20)
FC_DOC_PAGO Date
BANCO Variable characters (100)
NUM_CHEQUE Variable characters (20)
NUM_CUENTA_CTE Variable characters (50)
COBRADOR Variable characters (20)
NUM_RECIBO Number (8,2)
OBSERVACION Variable characters (1000)
USUARIO Variable characters (15)
ANIO Integer
Identifier_1 <pi>
Relationship_48
DET_PAGO
CD_DET_PAGO <pi> Integer <M>
CD_COM_TP_FACTURA <fi1> Integer
CD_PAGO <fi2> Integer
VAL_PAGO Number (8,2)
Identifier_1 <pi>

Anexo G. Manual de Usuario.

CARRERA ANÁLISIS DE SISTEMAS

MANUAL DE USUARIO

AUTOR: CHRISTIAN JAVIER LOPEZ VITERI.

Quito, 2019

Objetivo.

Brindar al usuario una guía del manejo y correcto funcionamiento del sistema.

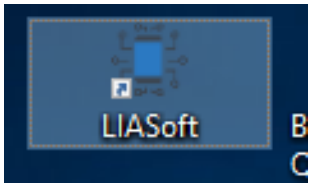
Antecedentes.

El sistema se encuentra desarrollado en un entorno web por lo cual el usuario deberá acceder al mismo desde un explorador web puede ser Google Chrome o Mozilla Firefox.

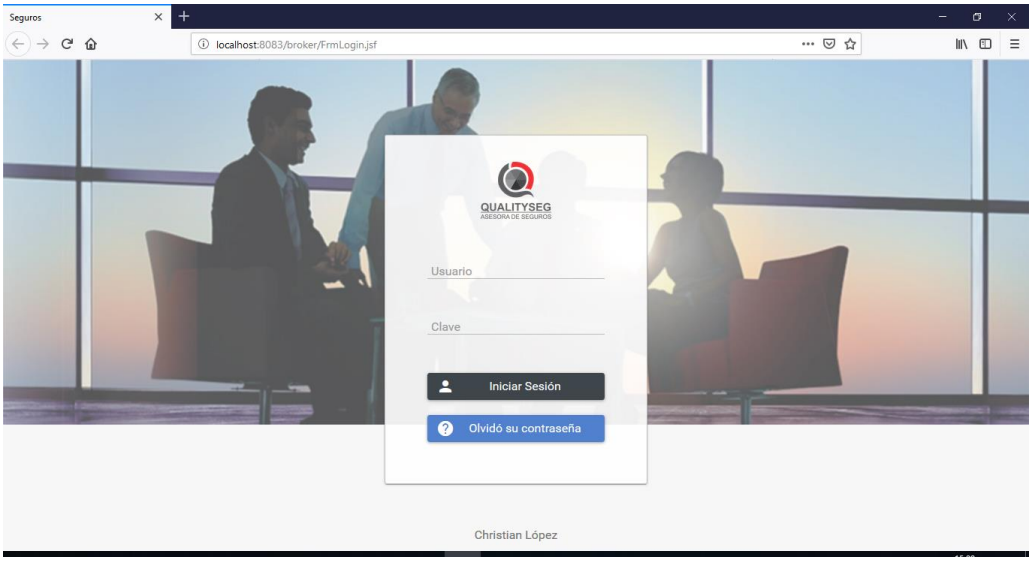
Instrucciones.

Ingreso al sistema:

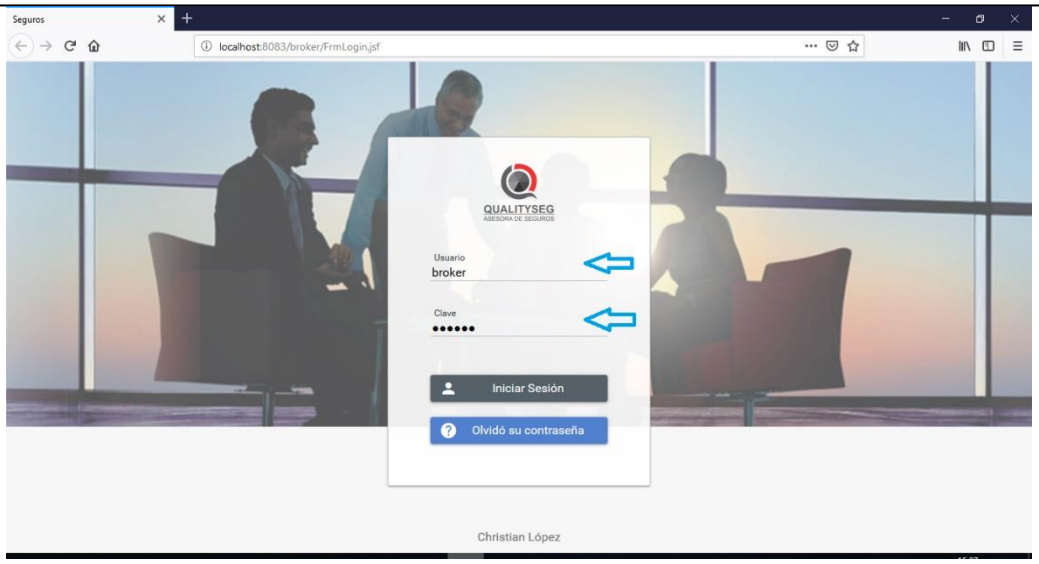
El usuario encontrara en su escritorio un icono con el nombre de LIASoft.

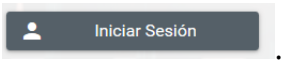


Al dar clic en dicho icono se abrirá un explorador de internet redireccionando a la página de acceso del sistema.



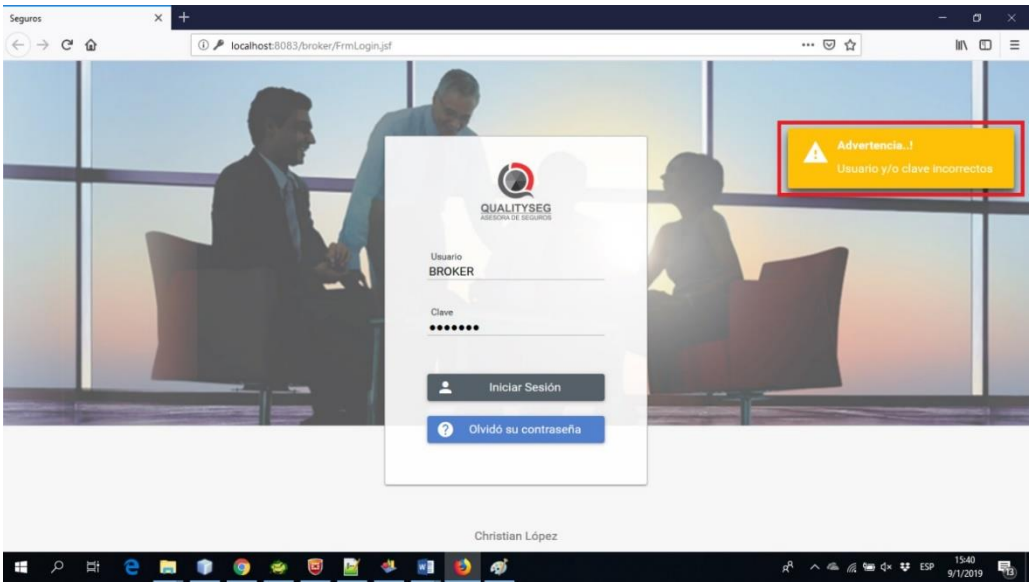
Para acceder al sistema se debe ingresar el usuario y clave que solicita el panel de acceso.



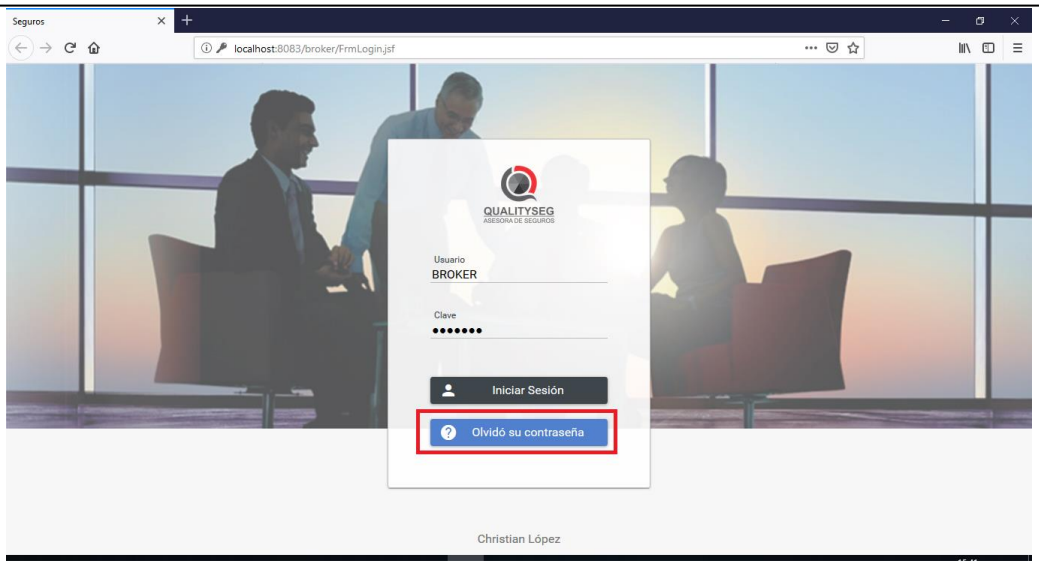
Finalmente se debe dar clic sobre el botón iniciar sesión 

.

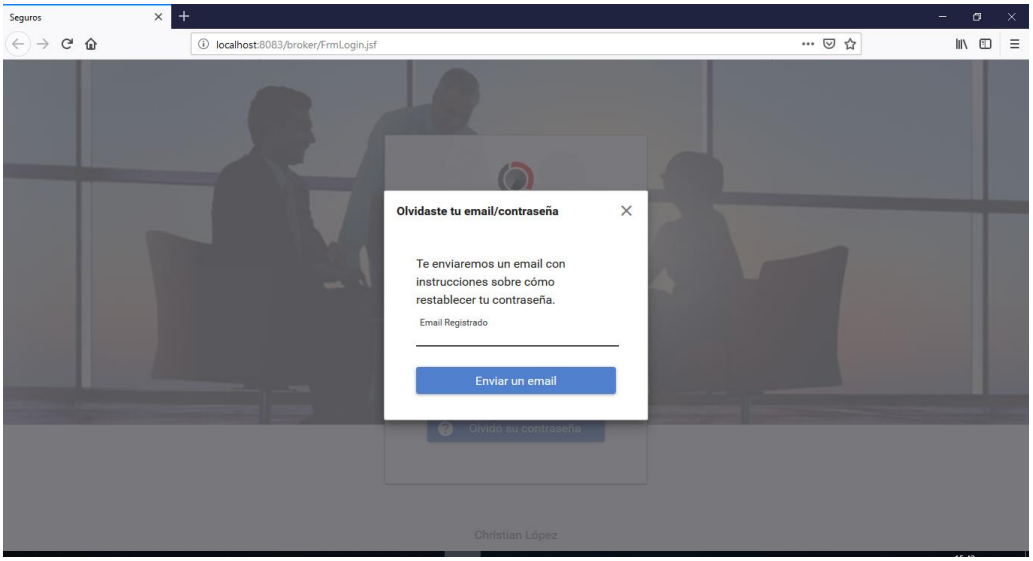
Si el ingreso es exitoso se redirecciona a la pantalla de inicio, de lo contrario presentara un mensaje indicando que la clave no es correcta



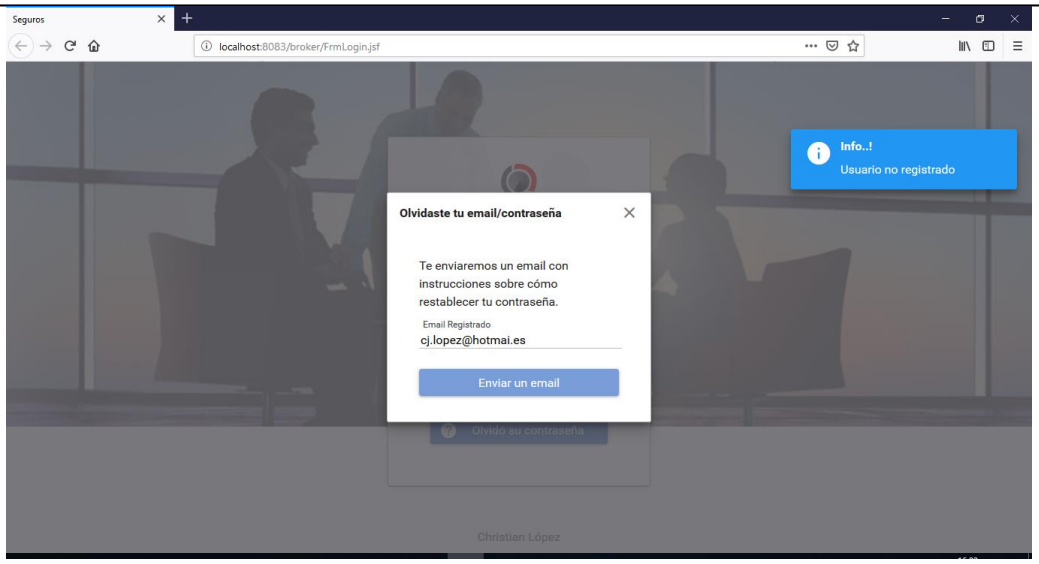
En caso de que no recuerde su clave de acceso puede dar clic sobre el botón olvido su contraseña



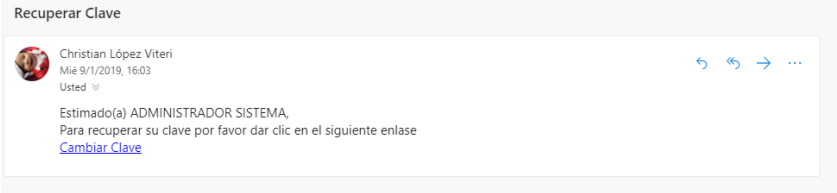
Esto presentara un panel de dialogo con los pasos a seguir para recuperar su clave.



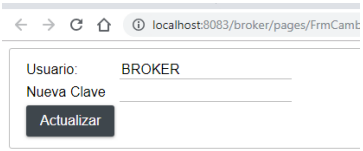
Al ingresar la cuenta de correo electrónico con la cual fue creado su usuario y presionar el botón enviar se enviará un correo electrónico con un link para poder recuperar la clave, en caso de que el correo no sea correcto se presentara un mensaje indicando que el usuario no existe.



Al verificar el correo en nuestra bandeja de entrada se presenta de la siguiente manera:

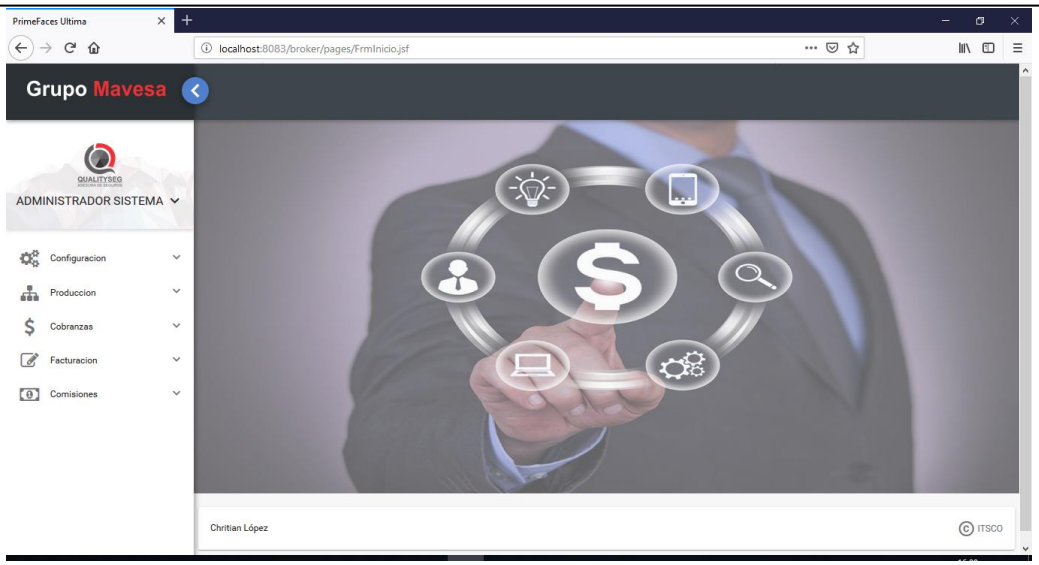


Al dar clic sobre el link cambiar clave se abre una interfaz para realizar el cambio.



Pantalla de inicio y Navegación por el menú del sistema.

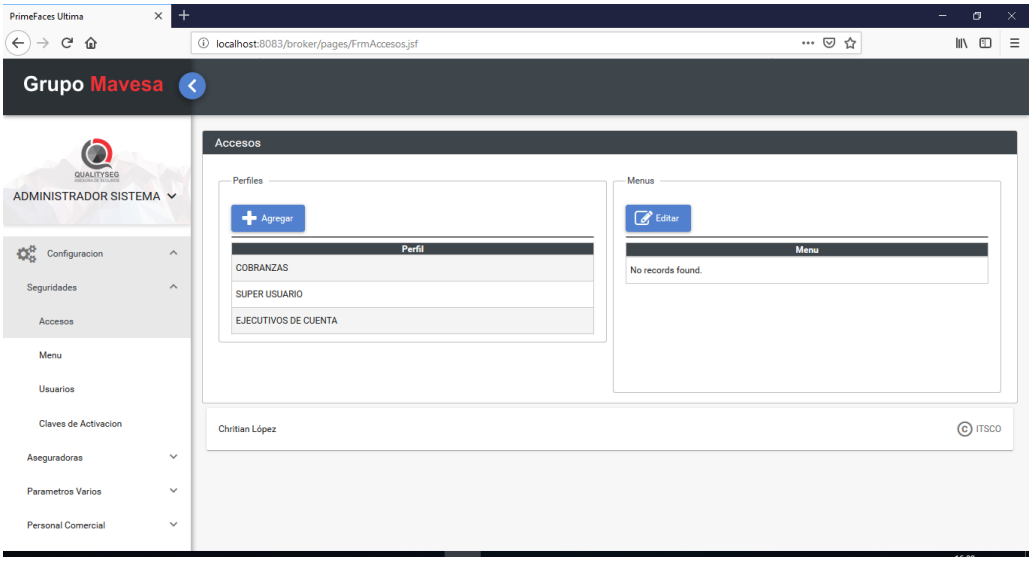
Una vez que se logró ingresar al sistema con las credenciales proporcionadas por el administrador, se presenta la pantalla de inicio de la siguiente manera:



Aquí encontramos el menú de opciones con las cuales trabaja el sistema.

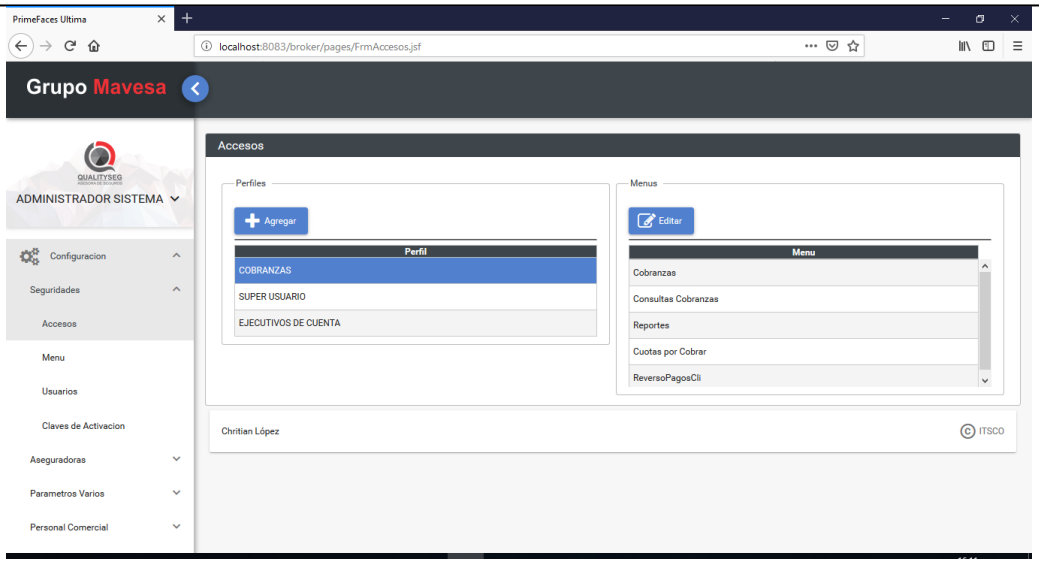
Accesos.

Son los accesos que tiene un usuario de acuerdo al perfil que selecciona, al dar clic en este menú se despliega la siguiente pantalla:

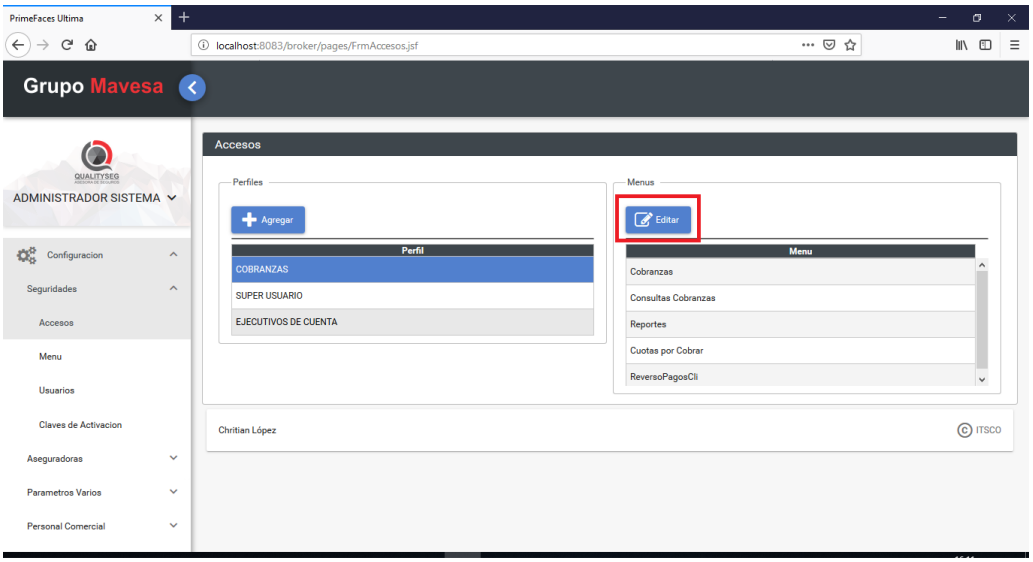


En dicha pantalla presentamos dos paneles, el primero se detalla todos los perfiles que cuenta el sistema, al lado derecho un panel de los menús de acceso.

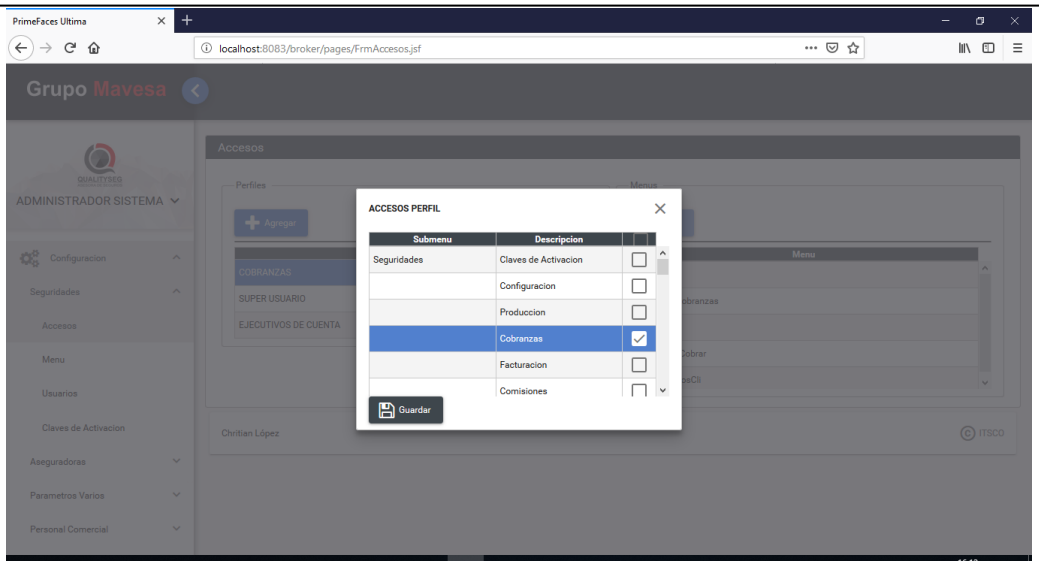
Para poder verificar que accesos tiene un perfil solo se debe seleccionar dicho perfil.



En caso de querer editar los accesos solo se debe presionar el botón editar ubicado en la parte superior del panel de menús de accesos.

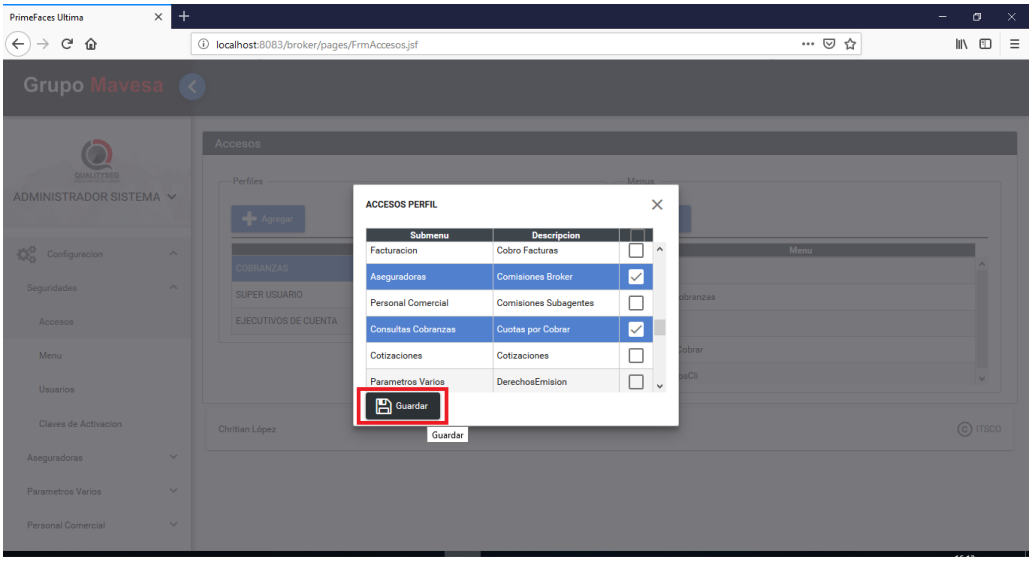


Al dar clic en editar se presenta el siguiente panel para seleccionar los accesos

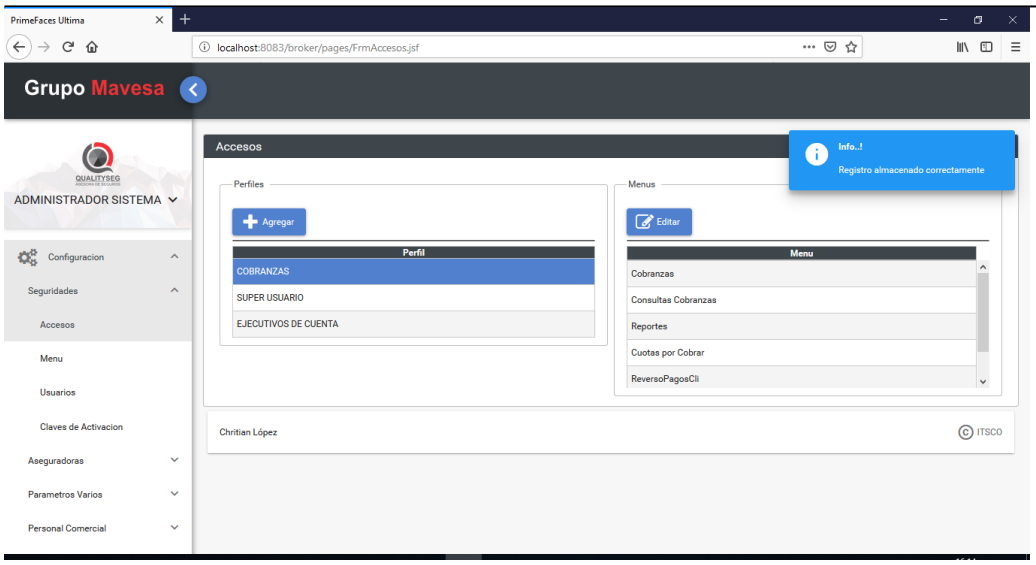


En dicho panel solo se debe marcar o desmarcar las opciones que desee para otorgar el acceso a dicho perfil de usuario.

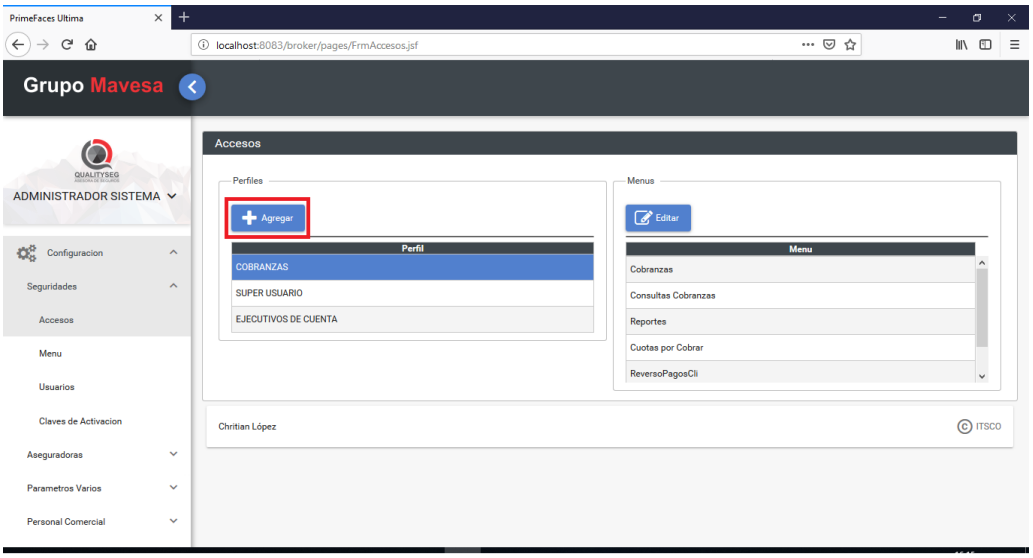
Una vez estandarizado los accesos se debe presionar el botón grabar



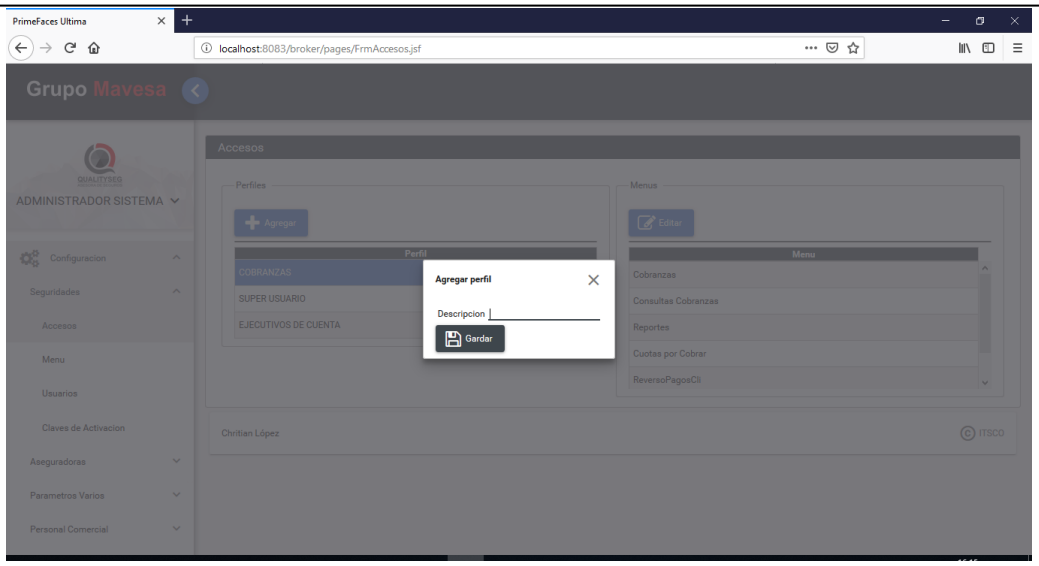
Al guardar se presenta un mensaje en la pantalla principal indicando que el registro se guardó de manera correcta.



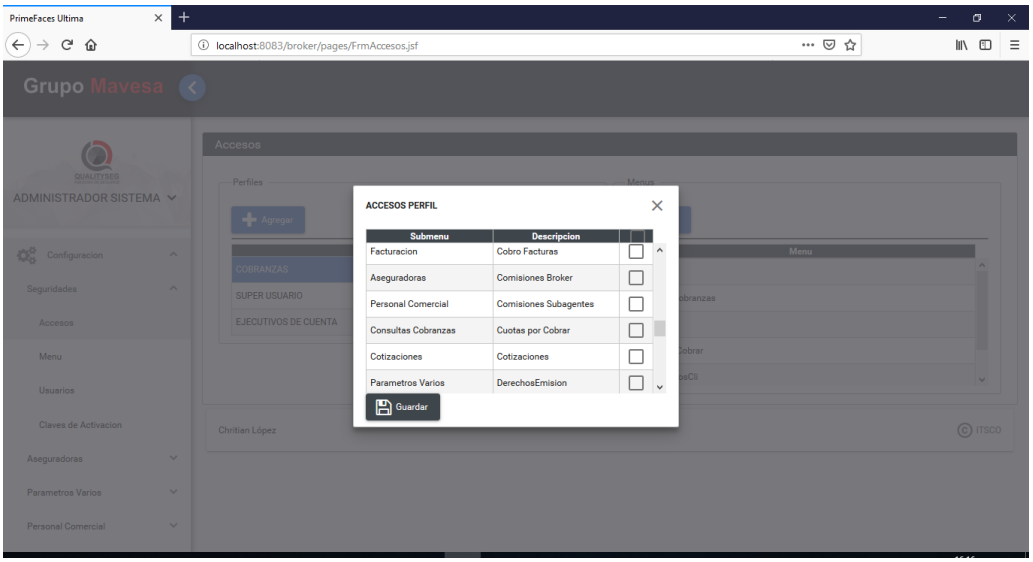
De la misma manera en caso de que se requiera crear un nuevo perfil se debe presionar el botón agregar.



Esto presentara un panel de registro del nuevo perfil.



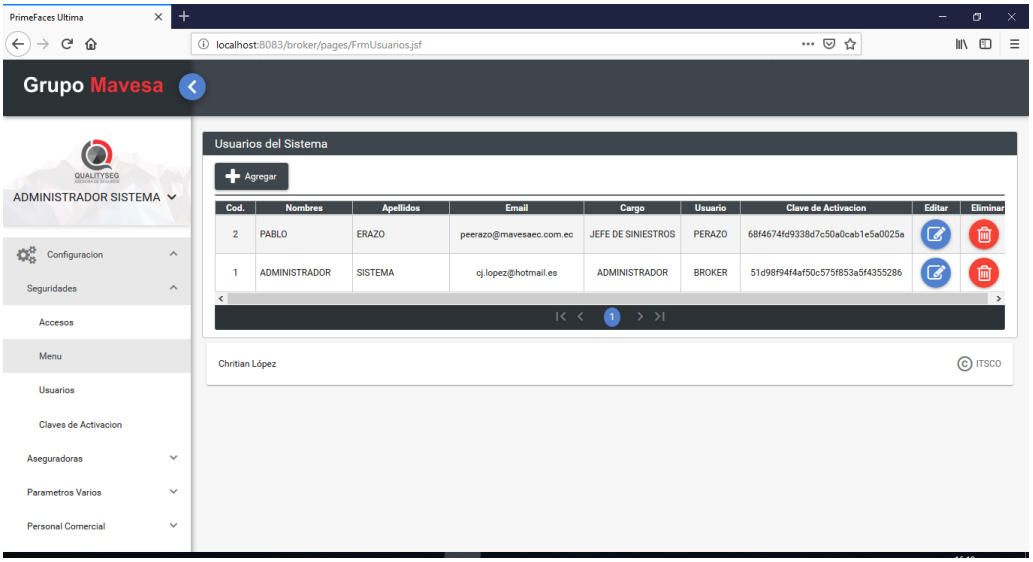
Al ingresar el dato se debe presionar el botón grabar para registrar el nuevo perfil y presentara el panel de accesos para configurarlos.



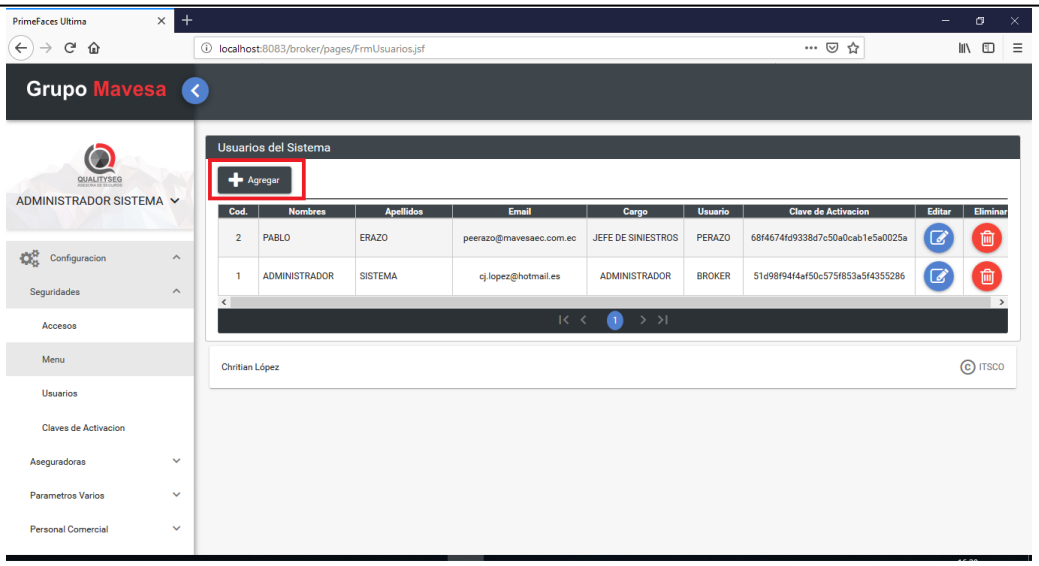
Se selecciona los accesos a brindar al perfil y se da en guardar.

Usuarios

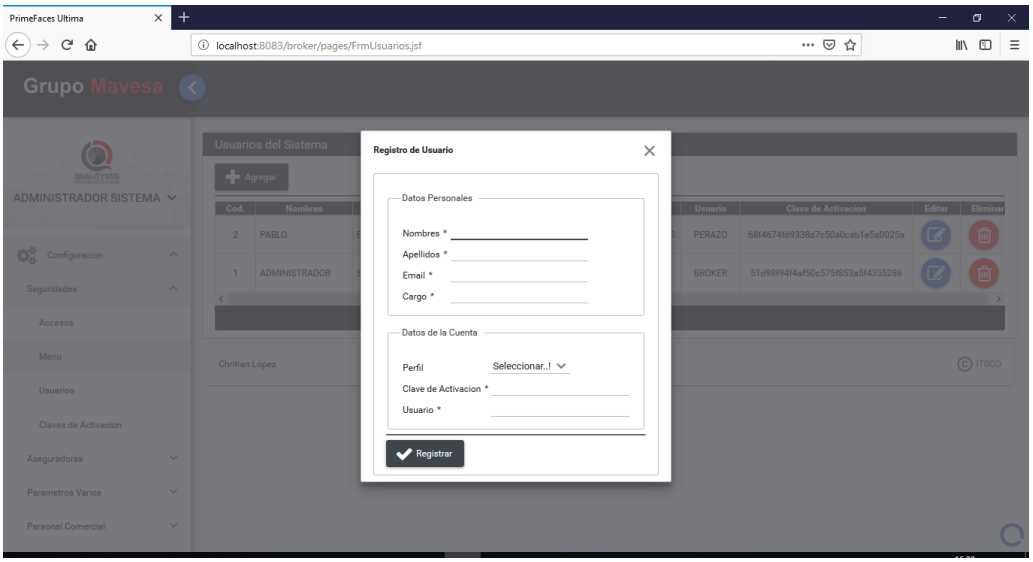
Al dar clic en el menú usuario, se presenta la pantalla de usuarios, en la cual se detalla todos los usuarios registrados.



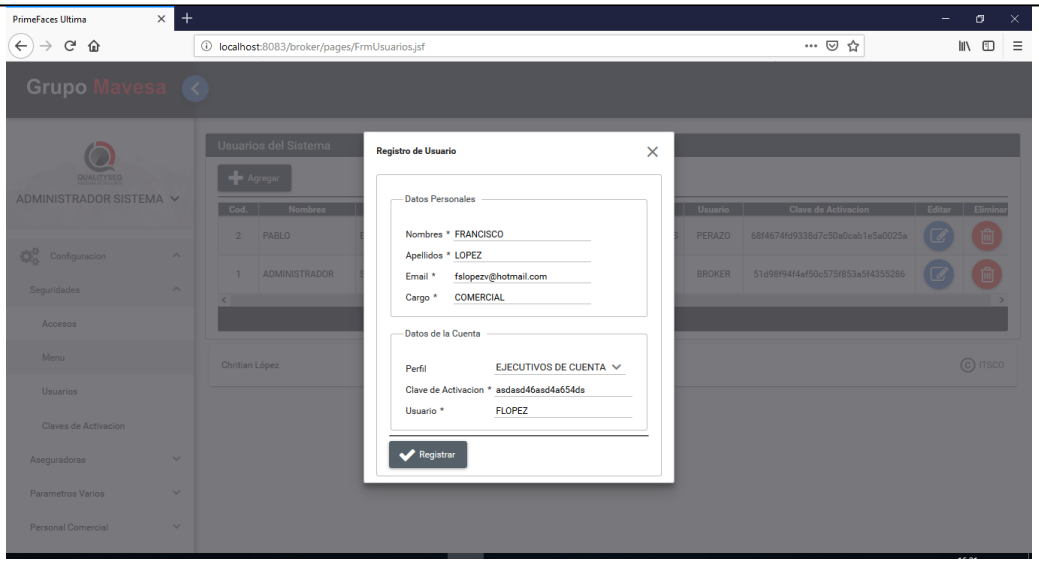
Para registrar un nuevo usuario se debe presionar el botón agregar.



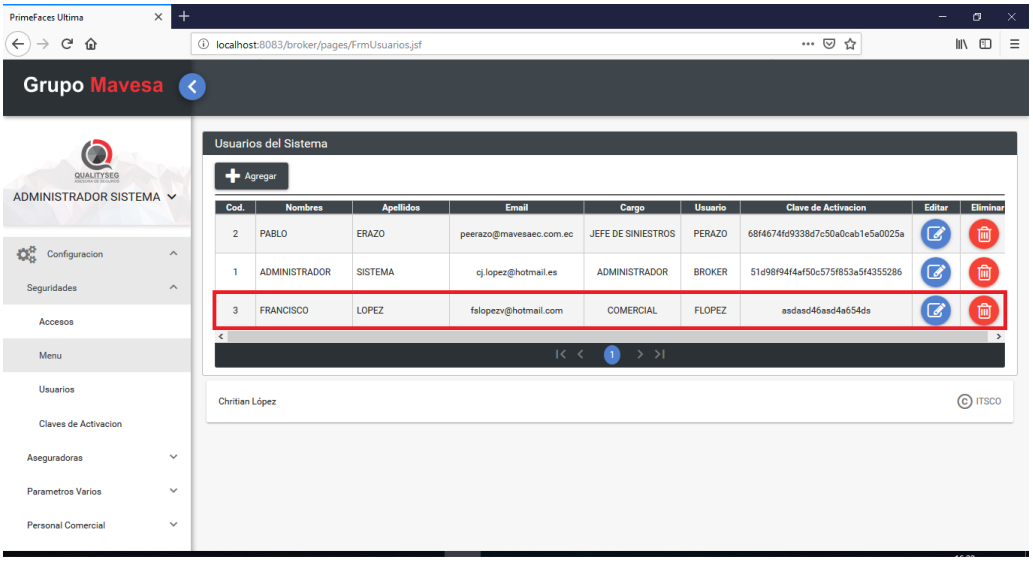
Esto presenta un panel de registro



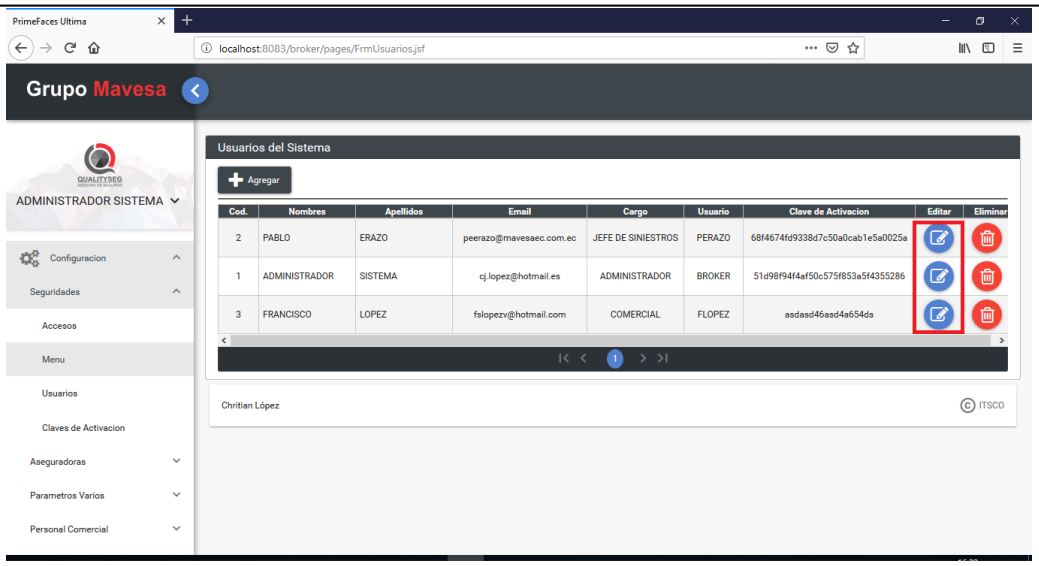
Se debe ingresar todos los campos ya que son requeridos por el sistema



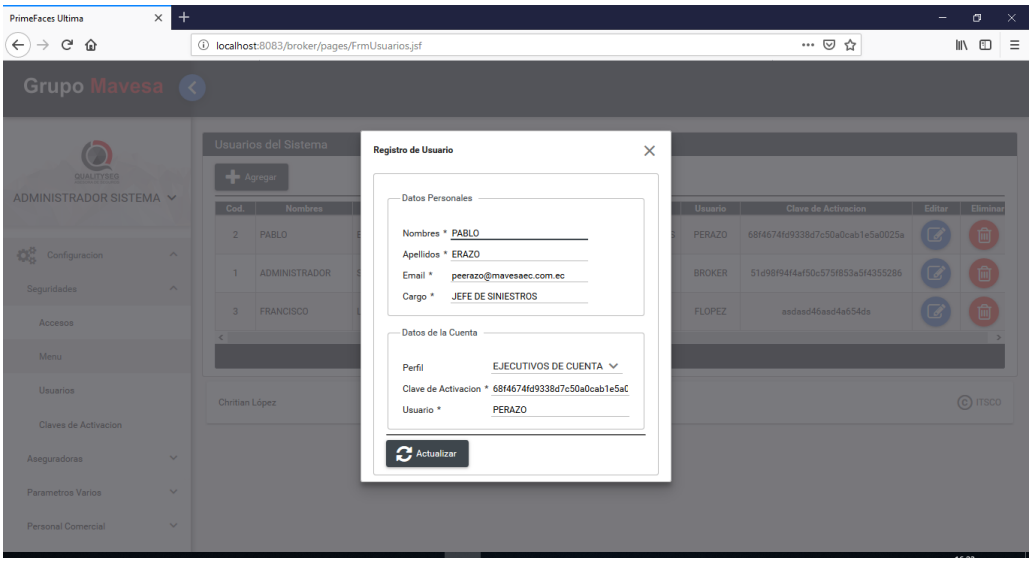
Al finalizar el registro se actualiza la pantalla de usuario con el dato del nuevo usuario



En el caso de que se requiera actualizar la información, se debe presionar el botón editar ubicado para cada usuario registrado.

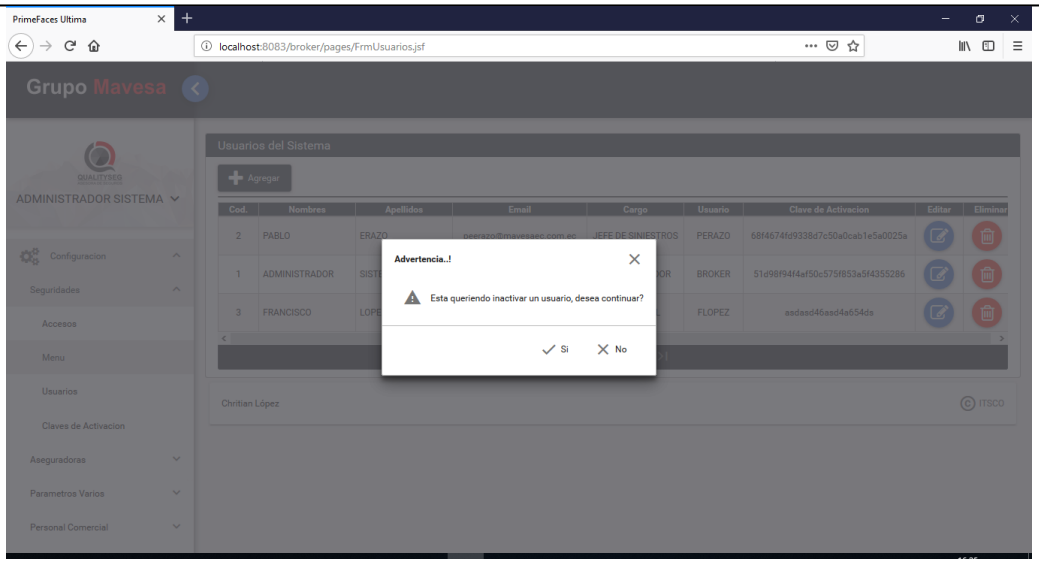


Esto nos presenta el panel con los datos del usuario a actualizar.

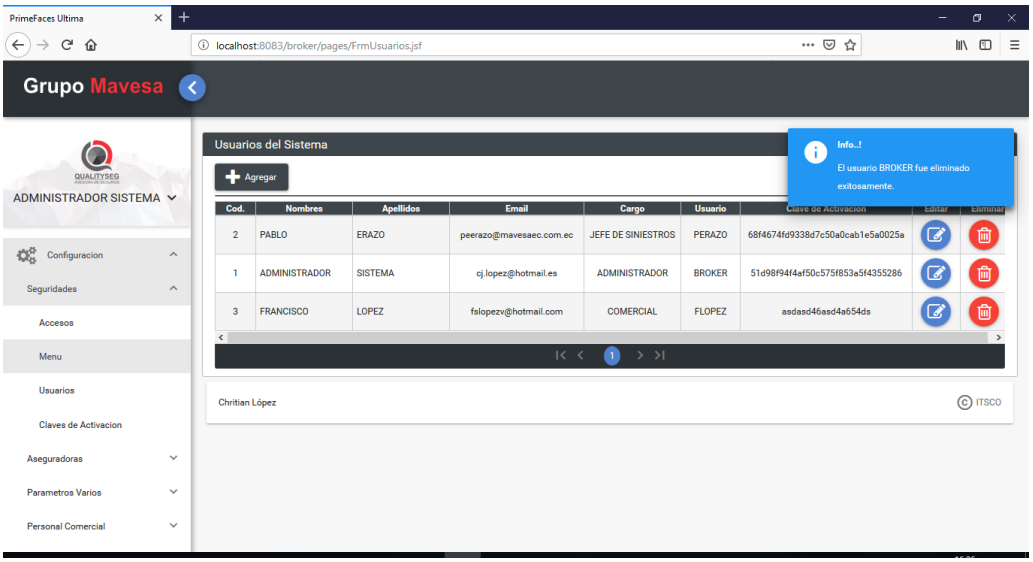


Finalmente, para guardar solo se debe presionar el botón actualizar.

Para desactivar un usuario únicamente se debe presionar el botón eliminar de la tabla usuario, la misma presentara un mensaje de confirmación para dicha acción.



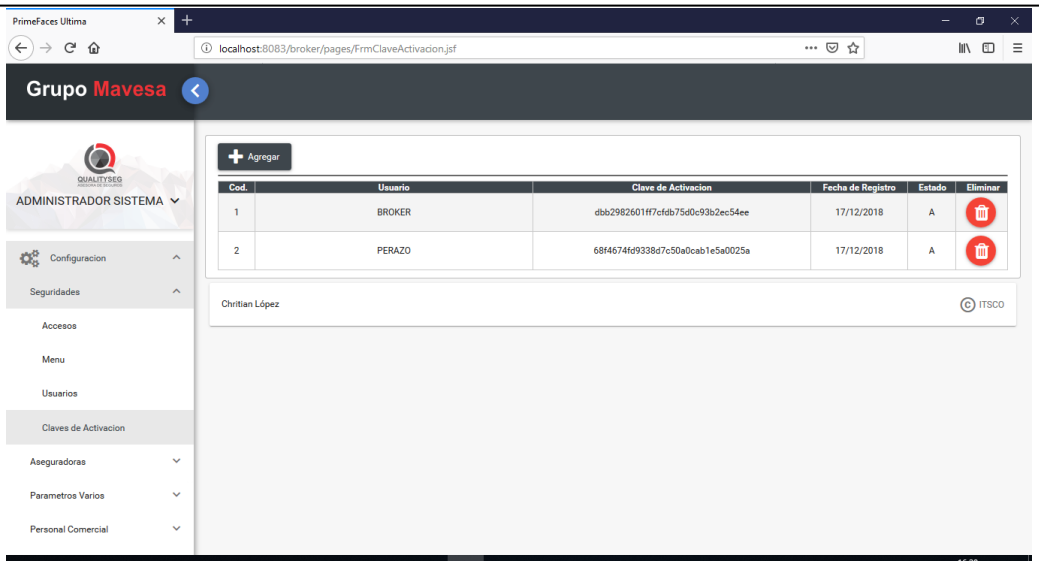
Al aceptar la confirmación del mensaje el usuario quedara inactivo impidiendo el acceso al sistema.



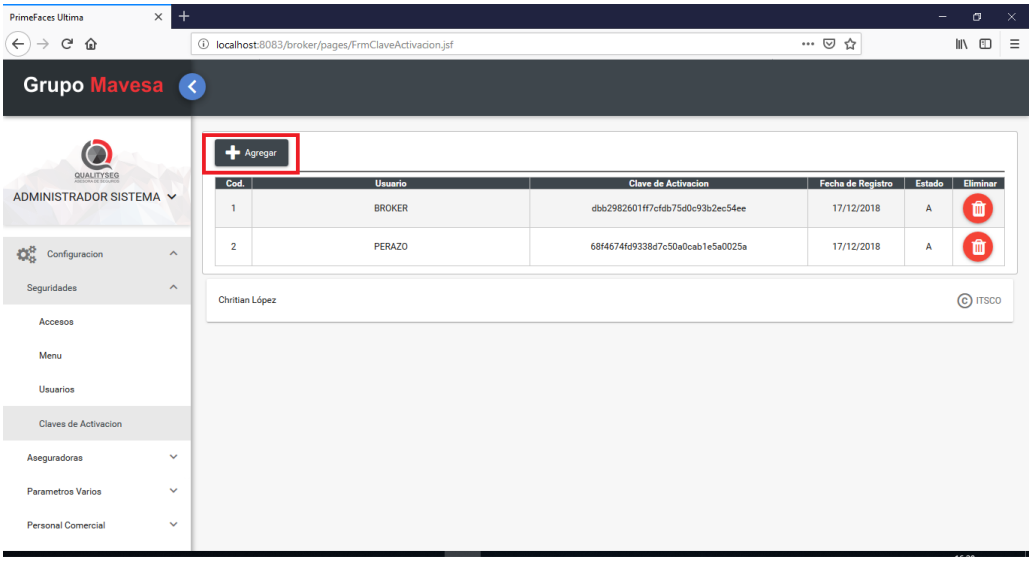
Claves de activación

Las claves de activación son generadas para un nuevo usuario que desee utilizar el sistema, en caso de que la clave ingresada al momento de registrar un usuario no sea correcta no se podrá acceder al sistema.

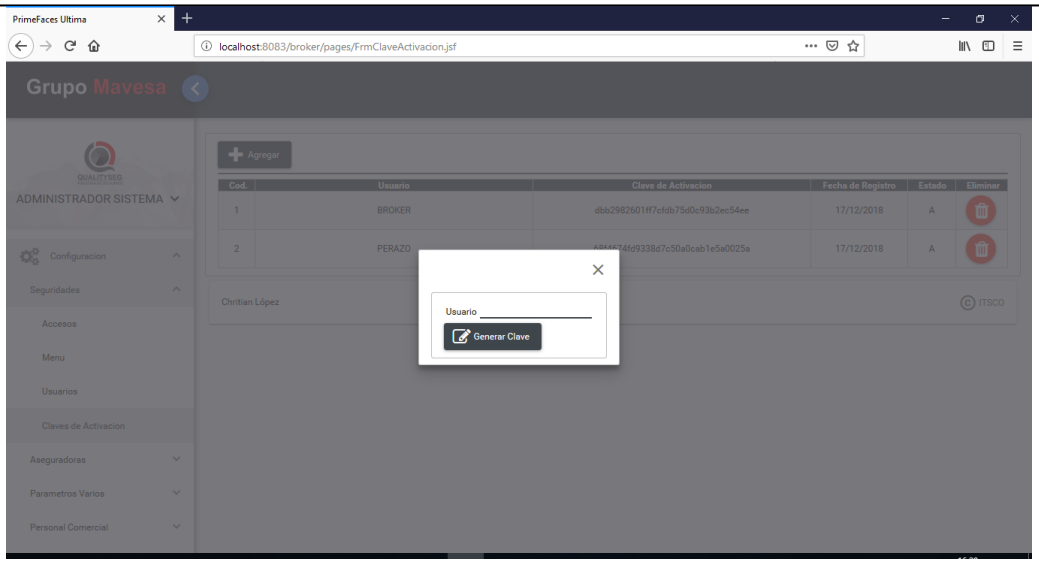
Para generar dichas claves se debe ir al menú de Configuración>Seguridades> Claves de Activación.



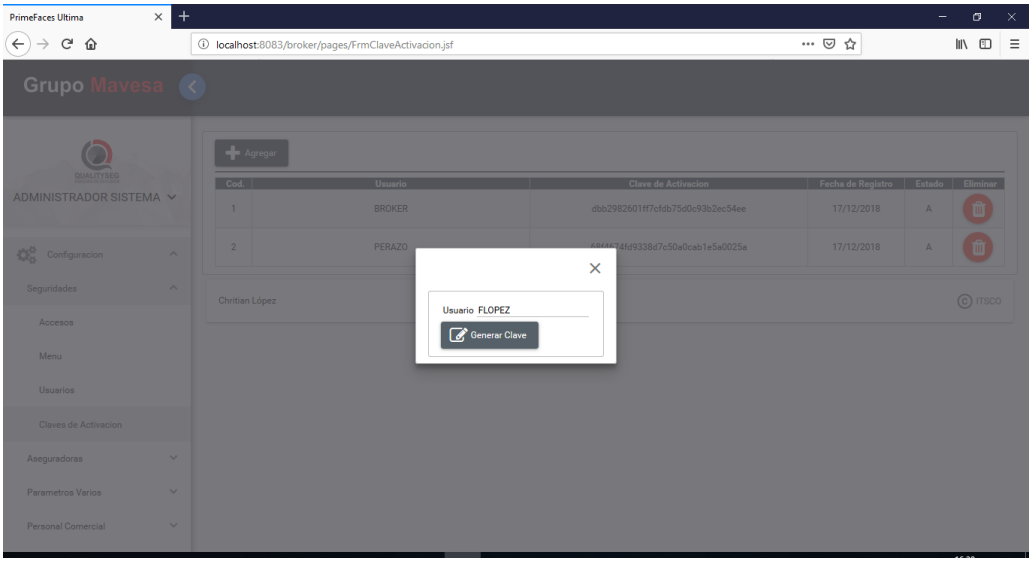
En dicha pantalla para agregar una nueva clave se debe presionar el botón agregar.

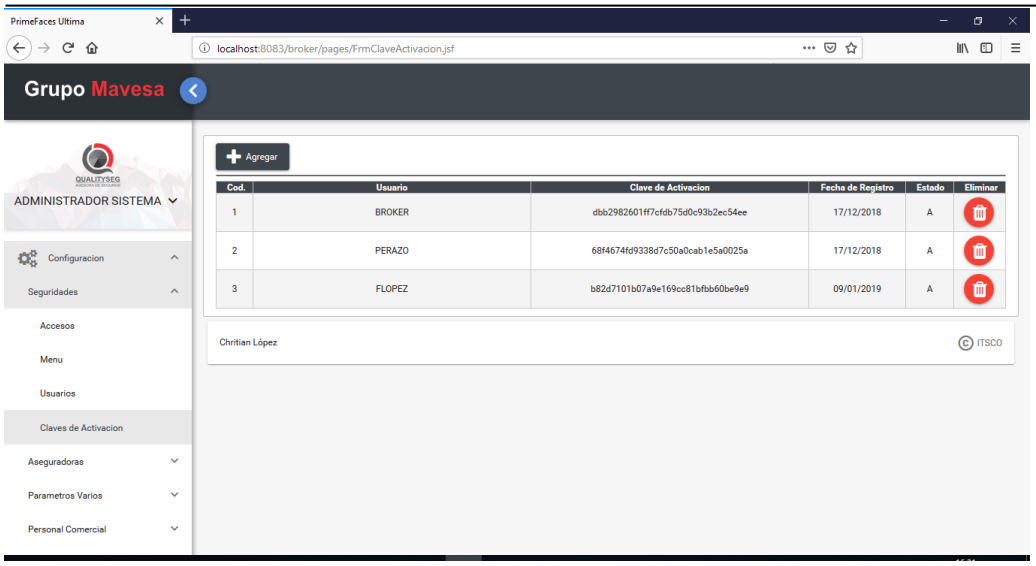


Se presentará un panel solicitando el usuario que será registrado.

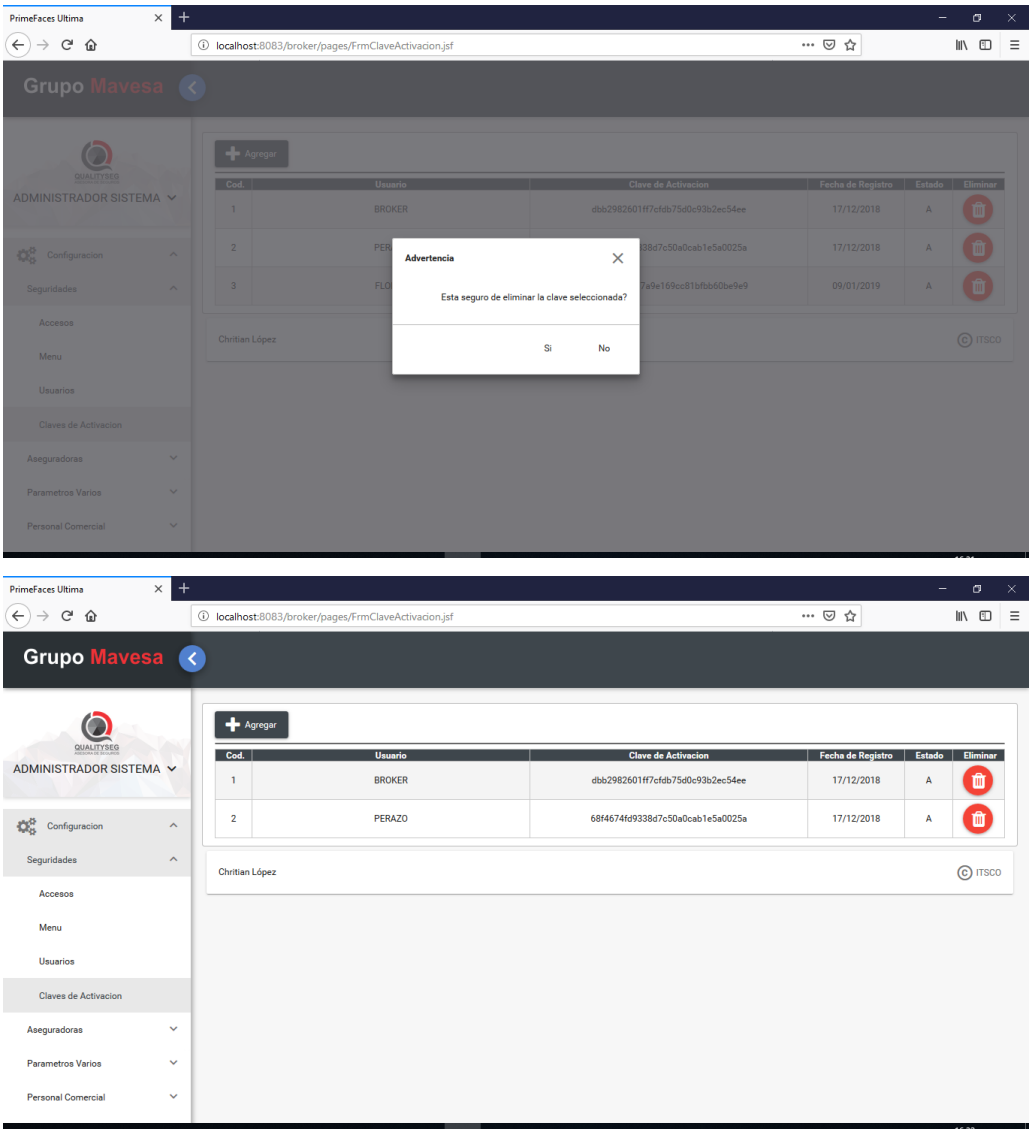


Al ingresar el usuario se debe presionar el botón generar clave, la misma que será almacenada.





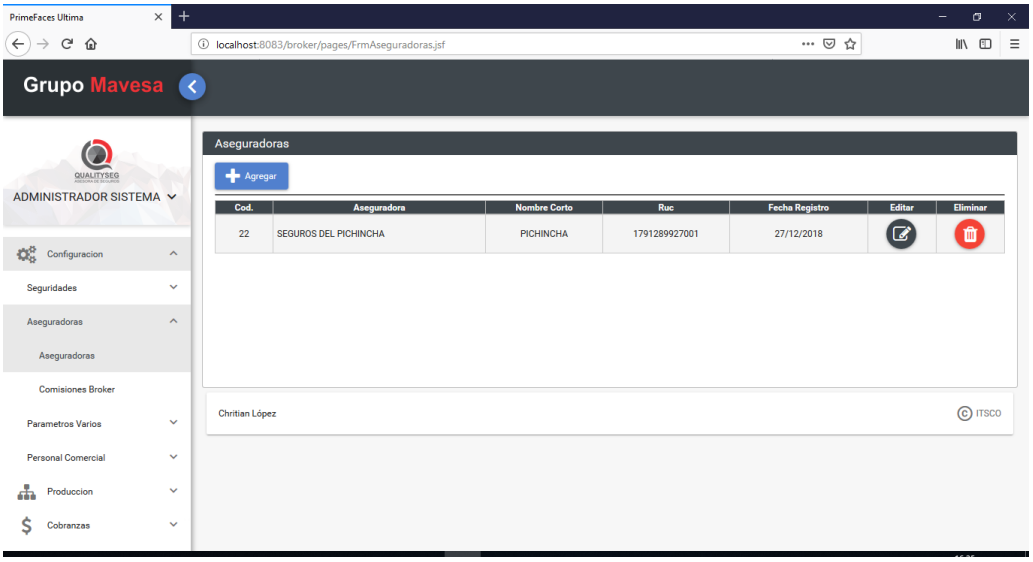
En caso de que se desee eliminar una clave se debe presionar el botón eliminar, el cual presentara un panel de confirmación.



Aseguradoras

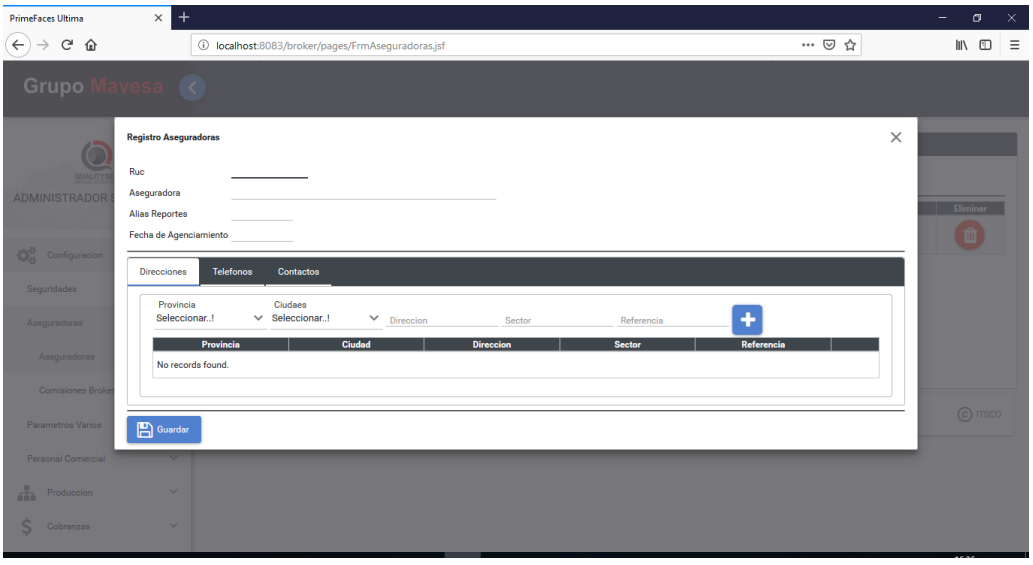
En el sistema se debe registrar compañías de seguros con las cuales se trabaja como corredor de seguros.

para verificar que compañías se encuentran registradas se debe ir al menú Configuración> Aseguradoras > Aseguradoras.

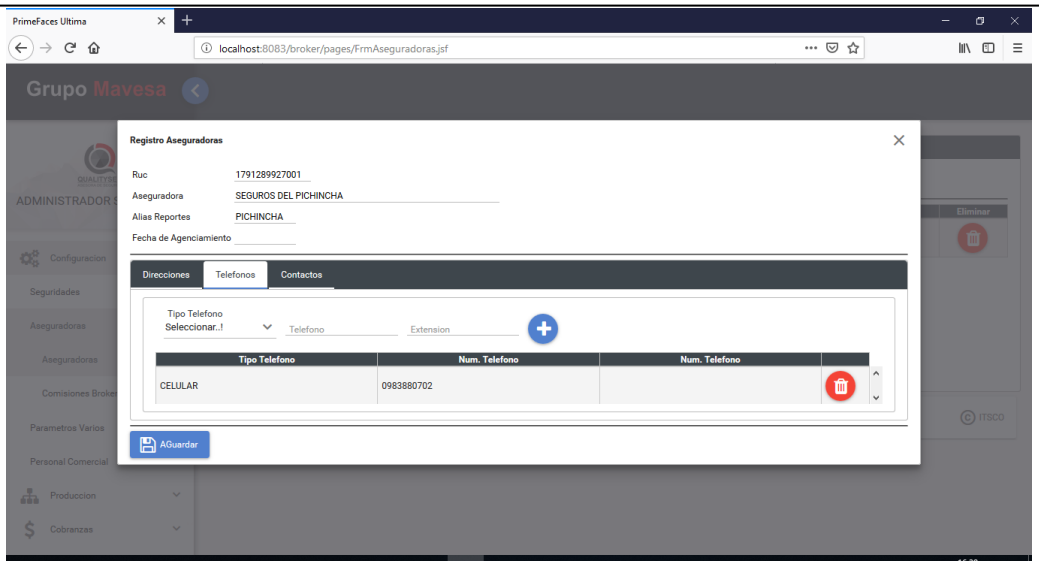


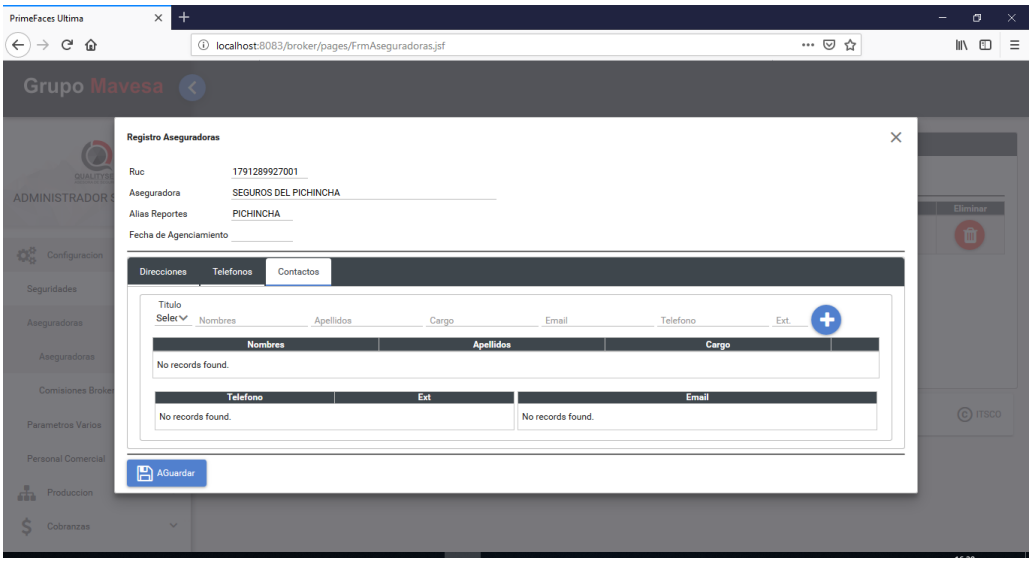
Para agregar una nueva compañía se debe dar clic sobre el botón agregar.

Presentará un panel de registro



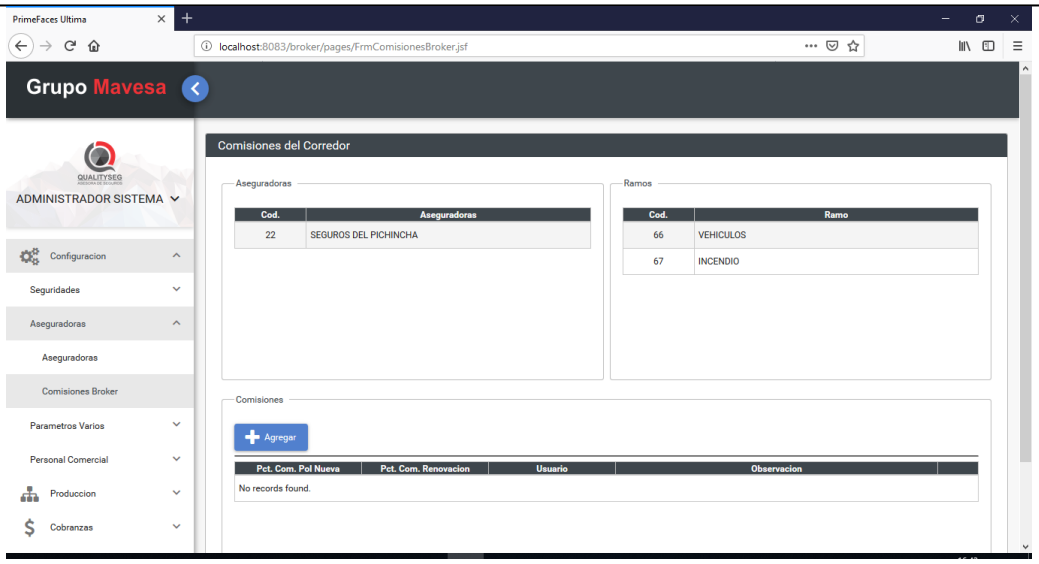
En el panel de registro se debe ingresar la información solicitada, adicional se debe ingresar lo datos de dirección, teléfonos y contactos de ser el caso.



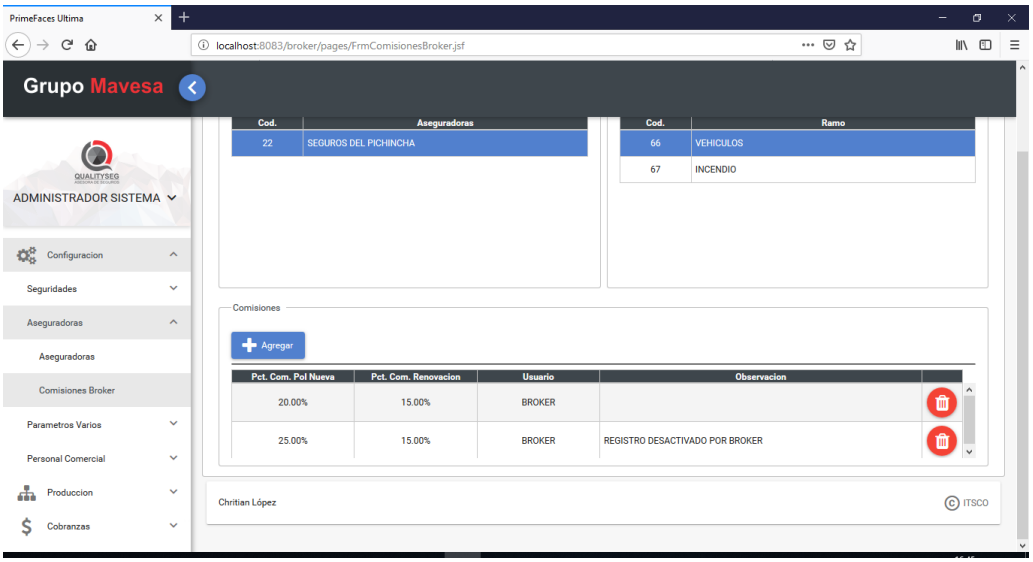


Comisión bróker

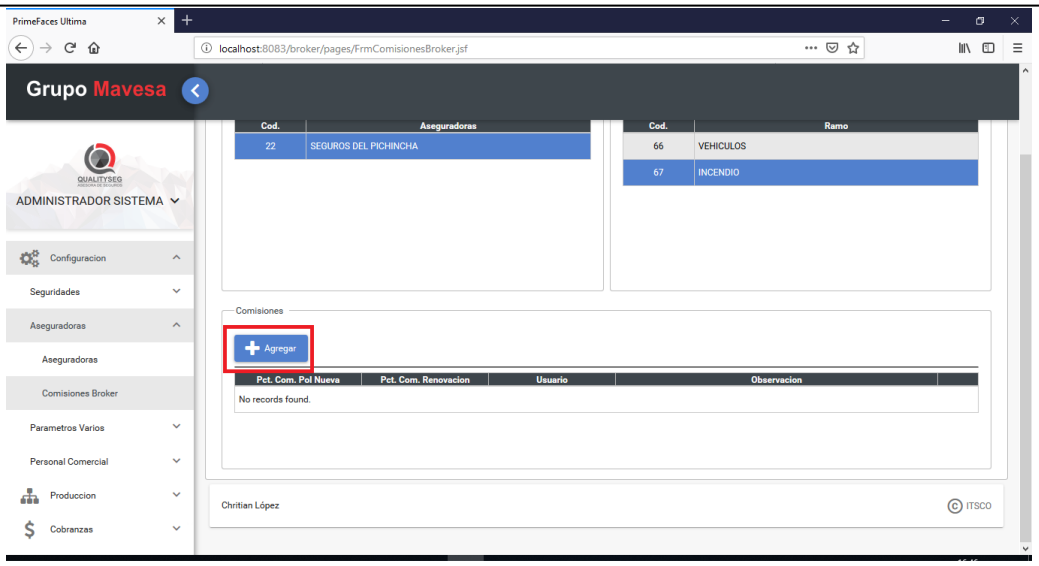
Al ingresar al menú comisión bróker se presenta un listado de compañías de seguros y los ramos de seguros que se cobra comisión.



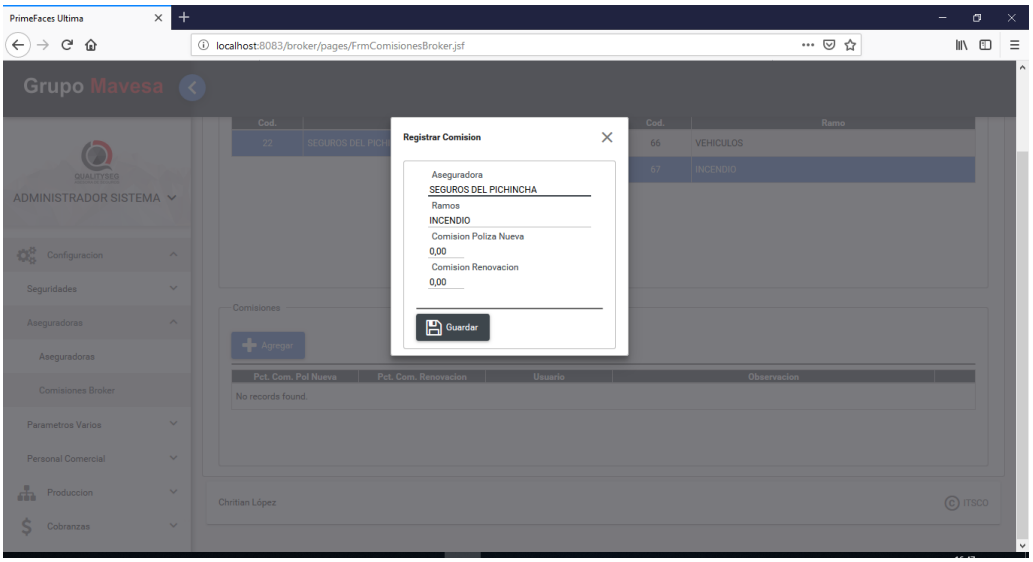
Para verificar la comisión de un ramo se debe seleccionar la compañía de seguros y el ramo de seguros.



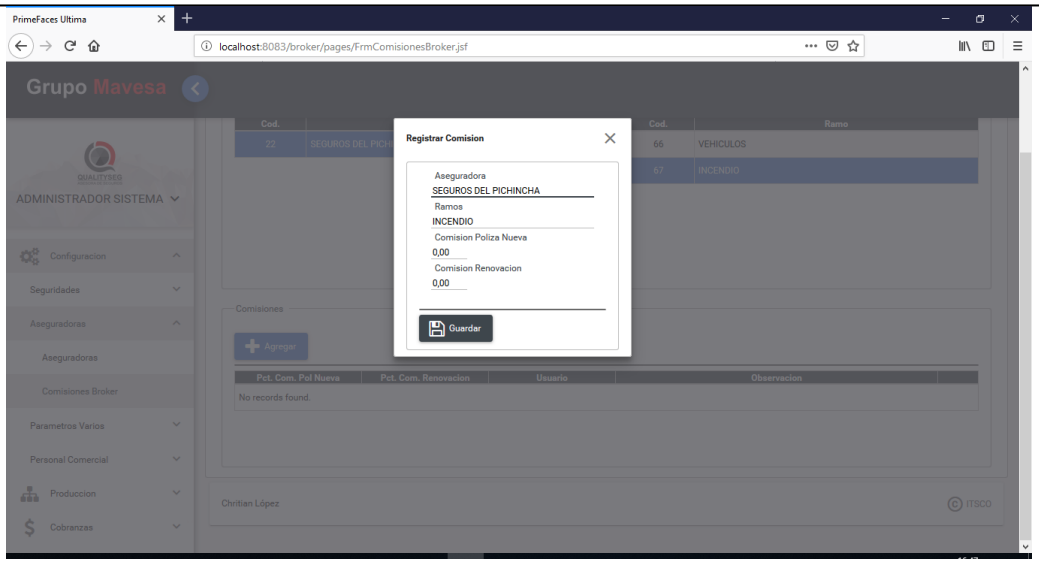
En caso de querer registrar la comisión para un ramo se debe presionar agregar una vez seleccionada la compañía de seguros y el ramo.



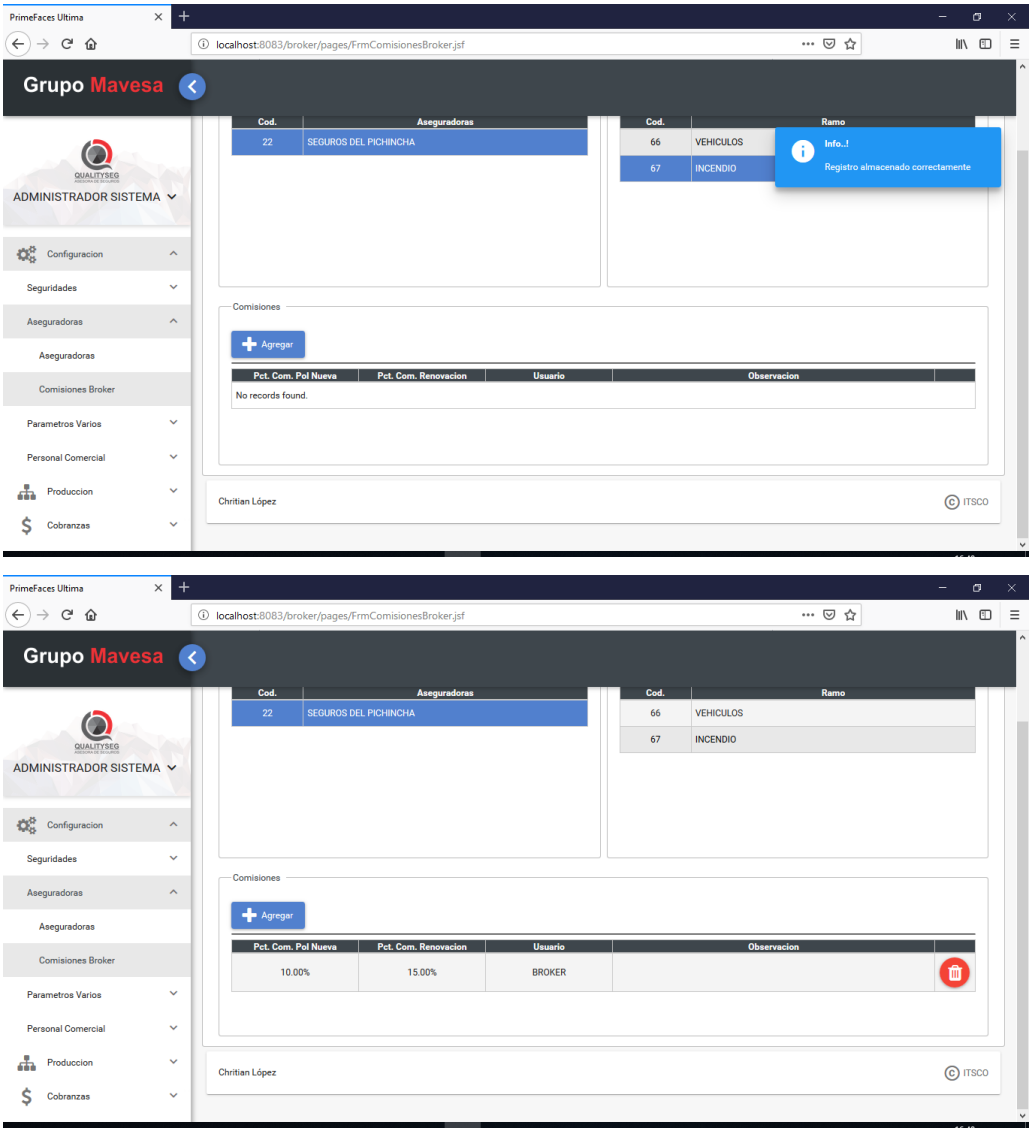
Se presentará un panel de registro con los datos de la compañía de seguros y el ramo.



Se debe registrar los valores de comisión para pólizas nuevas y pólizas en renovación.

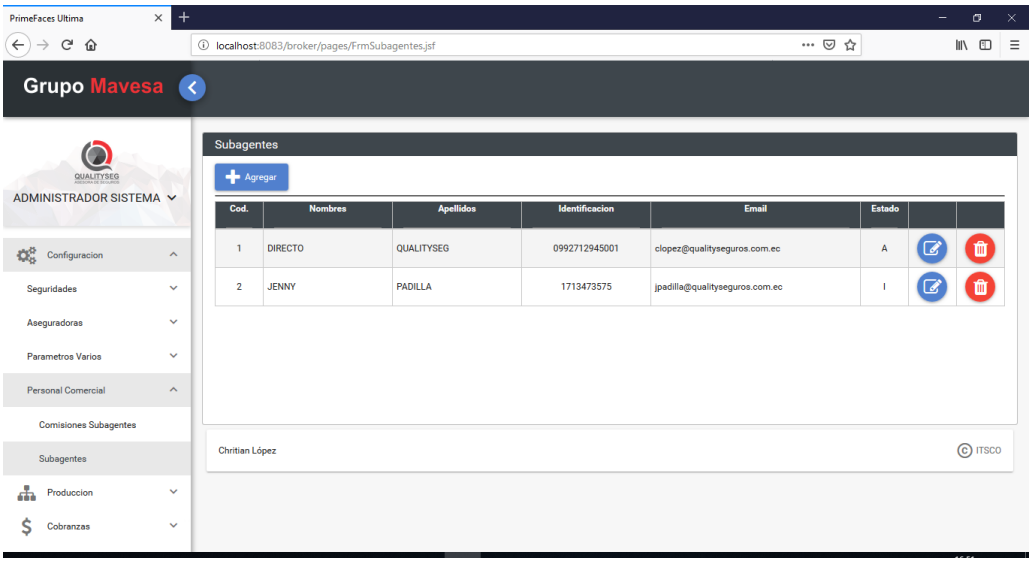


Finalmente presionar el botón grabar.

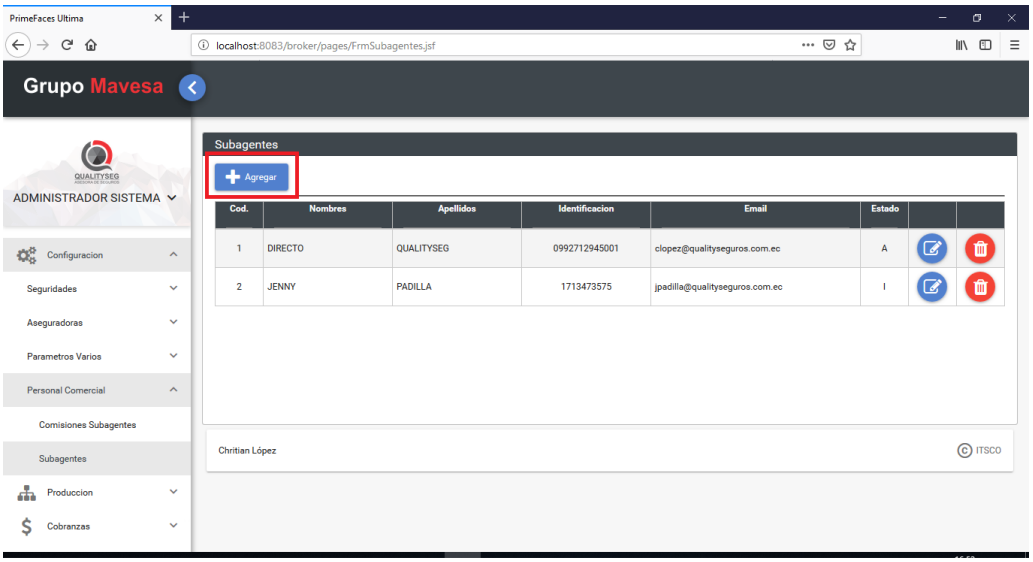


Subagentes

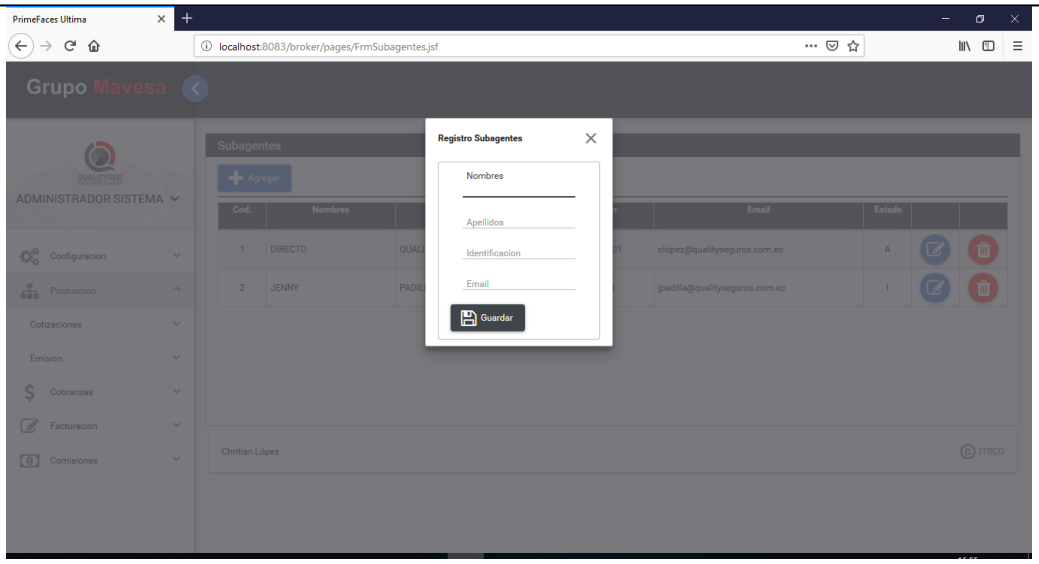
Al ingresar al menú subagentes se presenta el listado de todos los subagentes.



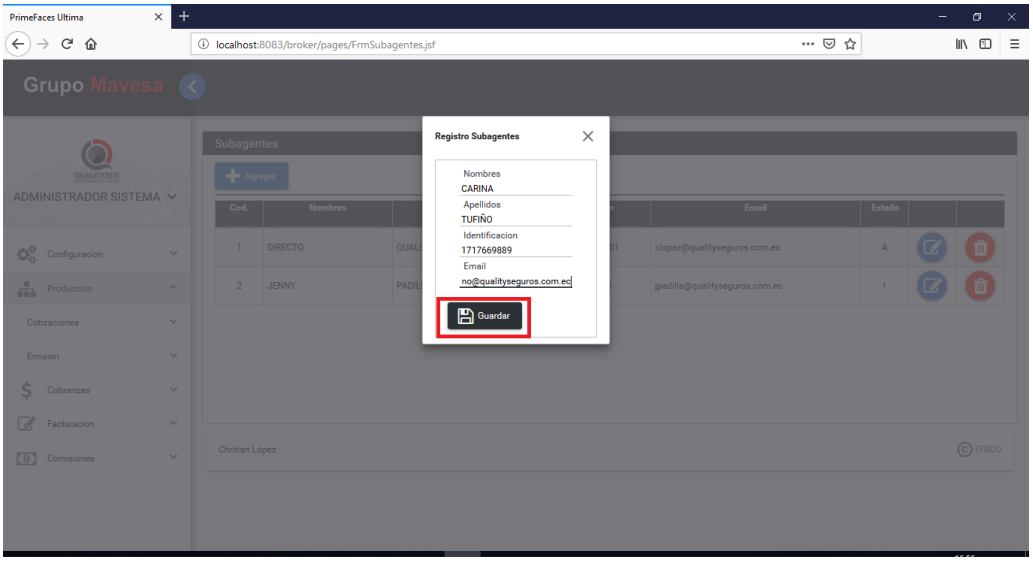
Para registrar un nuevo subagente, se debe presionar el botón agregar.



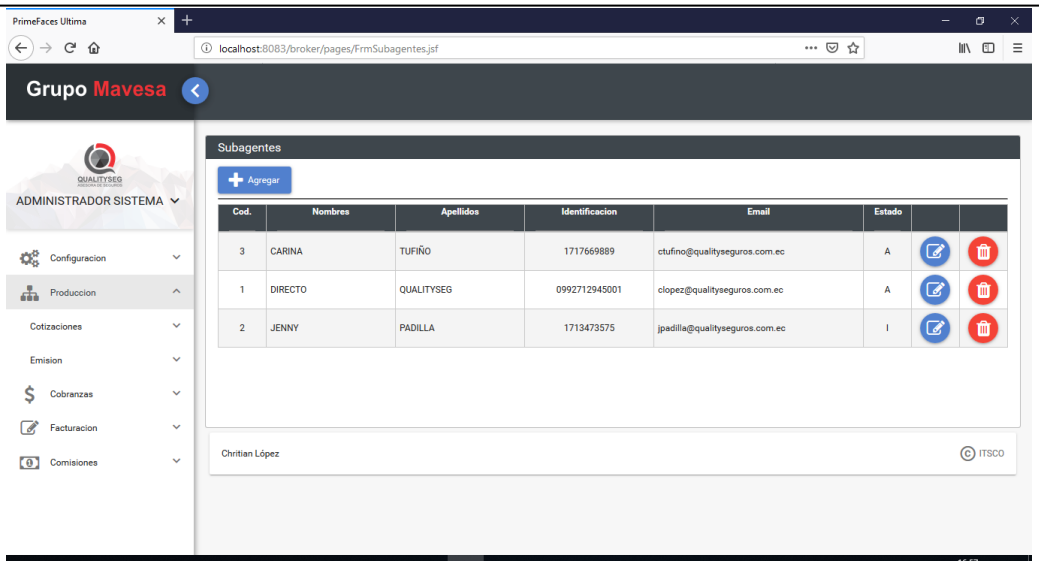
Se presentará un panel de registro.



Al ingresar todos los datos solo se debe presionar el botón guardar.

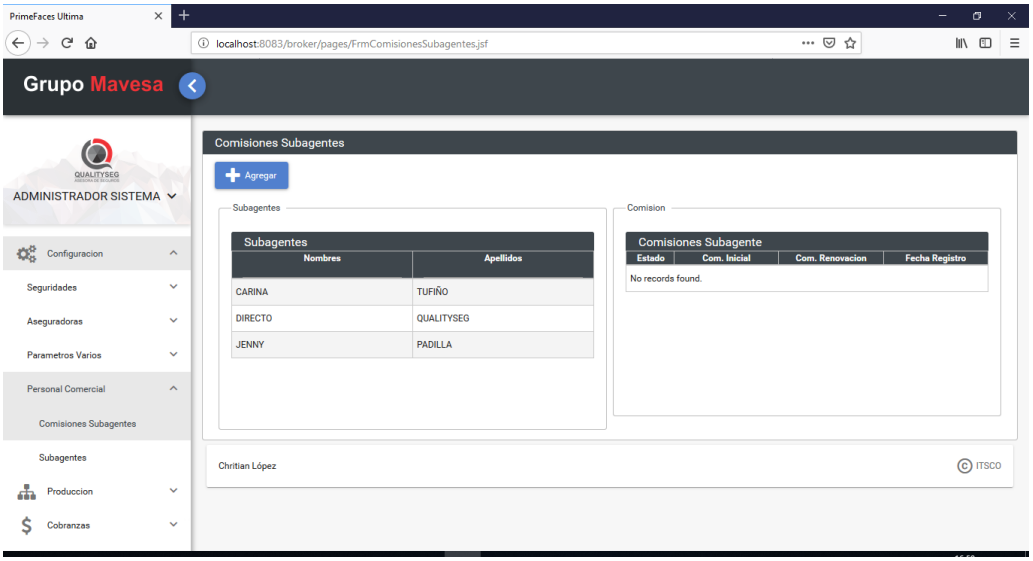


Esto registrara el nuevo subagente y lo presentara en lista.

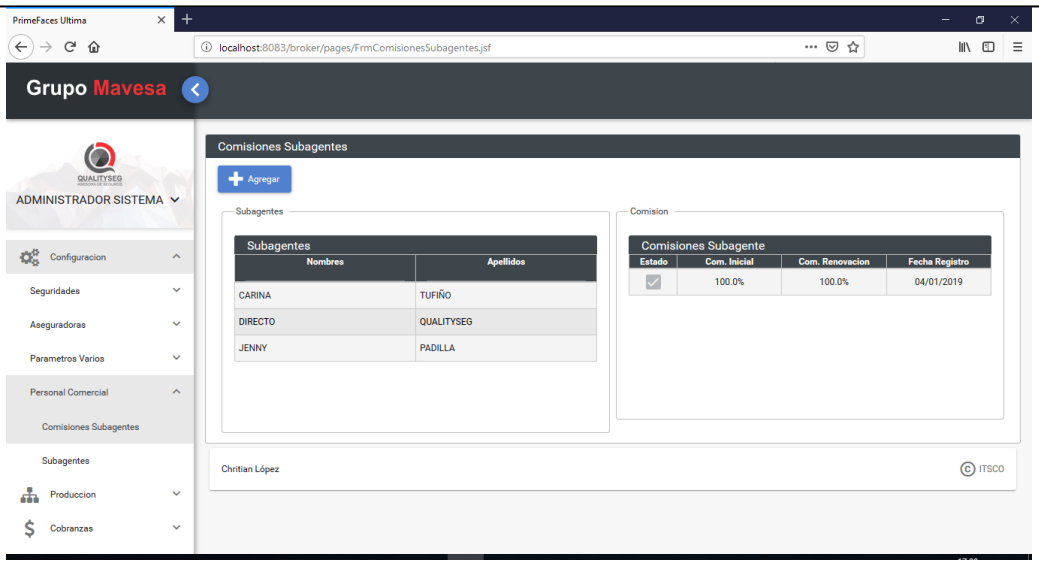


Comisión Subagente

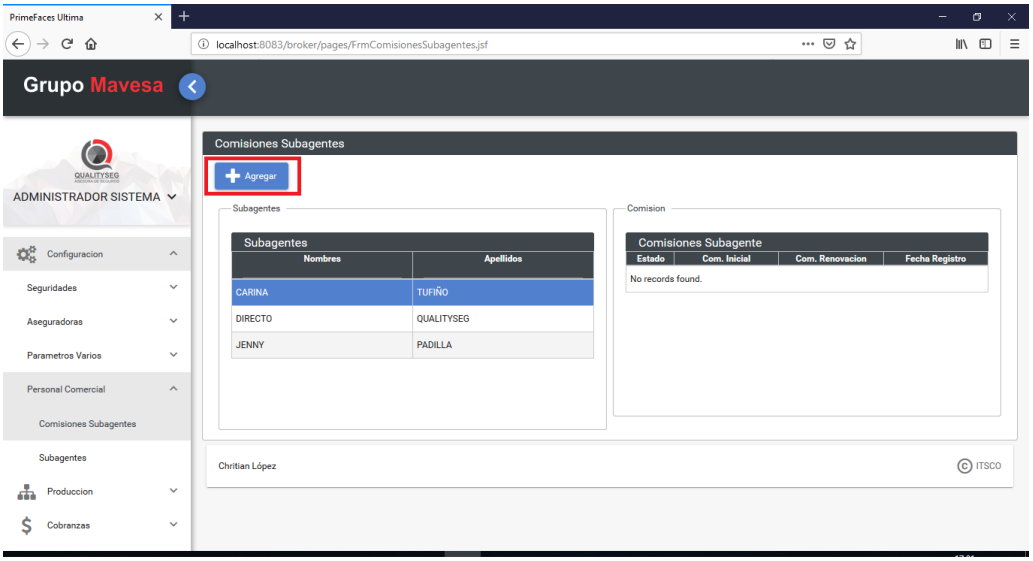
Para emitir una póliza es importante parametrizar el valor de comisión para un subagente, por lo tanto, ingresamos al menú Configuración> Personal Comercial > Comisión Subagentes.



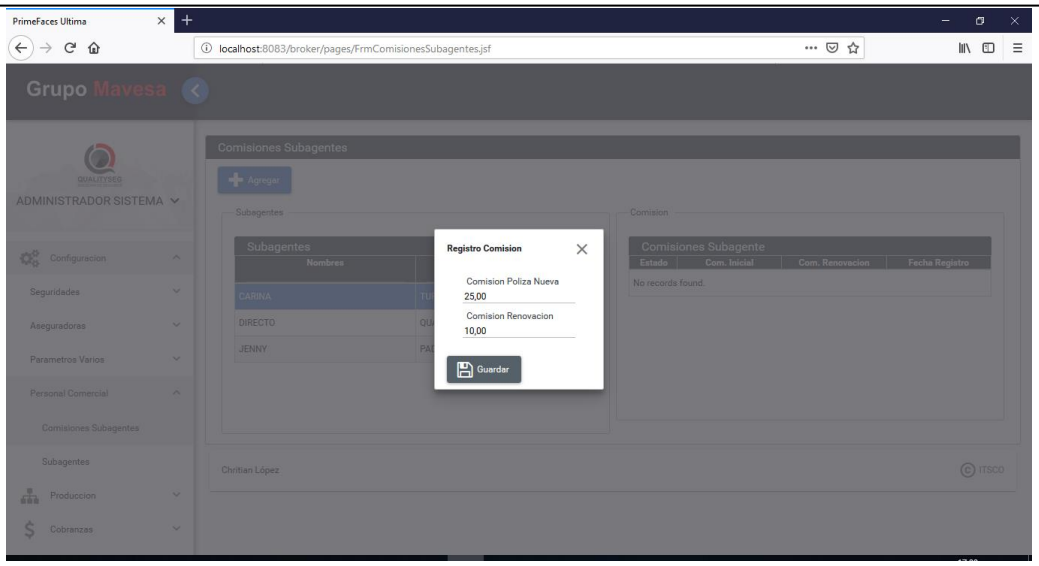
Para verificar la comisión de un subagente solo se debe seleccionarlo.



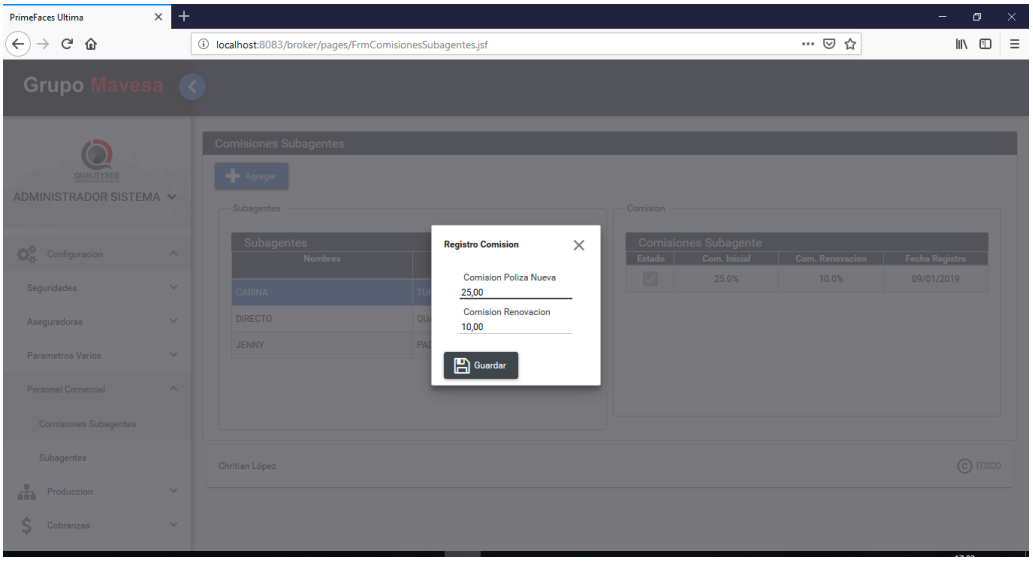
En caso de que un subagente no disponga de comisión ingresada se debe seleccionar el subagente y presionar el botón agregar.



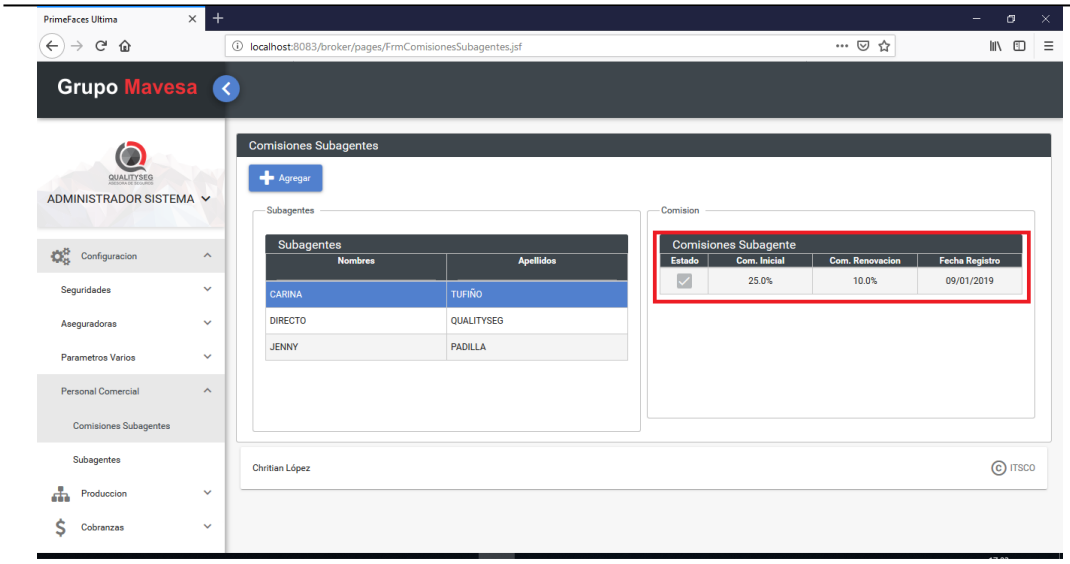
Se presentará un panel donde se solicita el porcentaje para pólizas nuevas y pólizas renovadas.



Al ingresar los datos de comisión se debe dar clic sobre el botón guardar

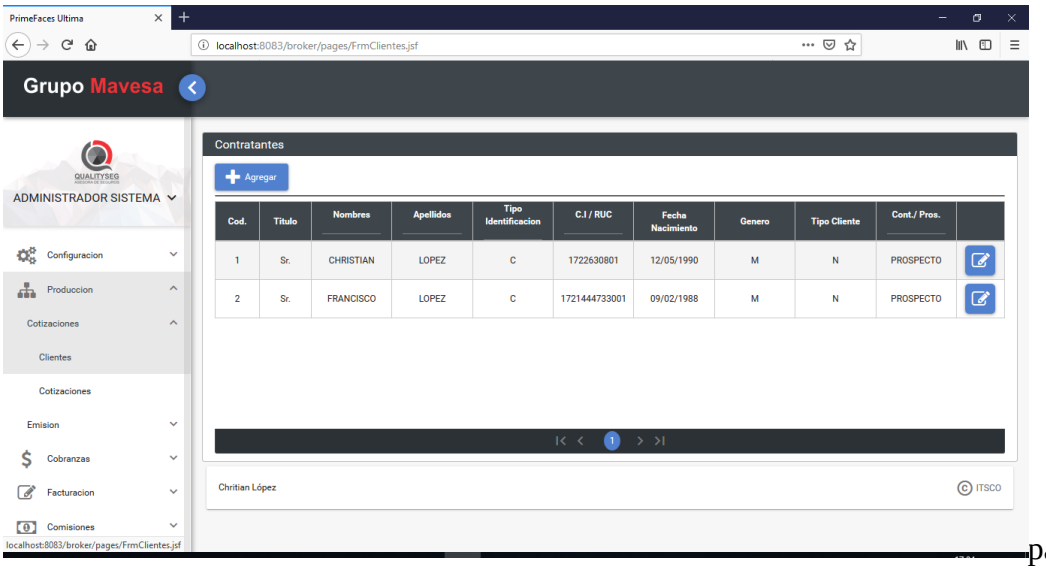


Finalmente, se verá reflejado en el panel de comisiones.

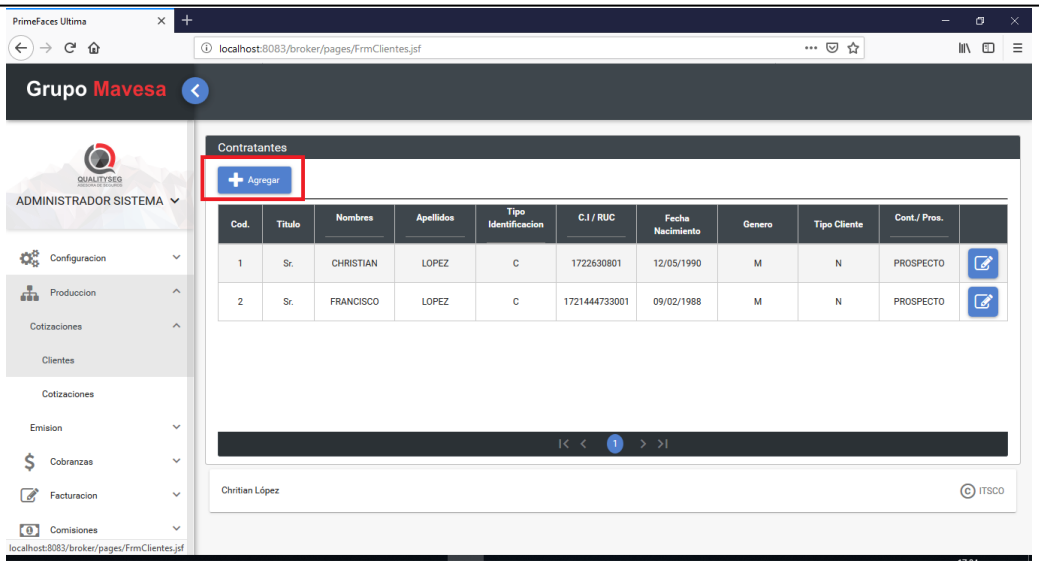


Clientes

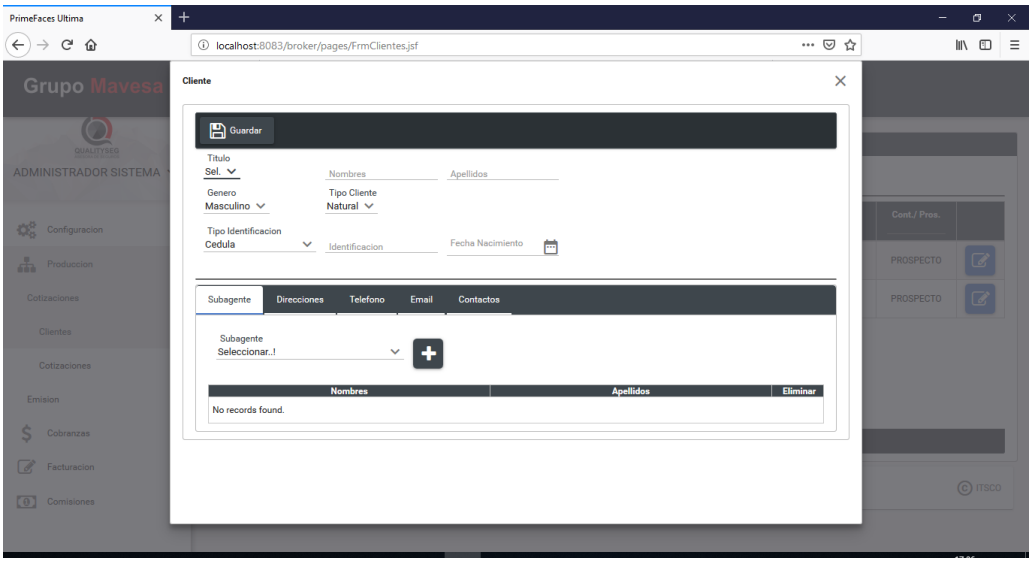
Para verificar los clientes registrados o registrar un nuevo cliente se debe ir al menú producción > Cotizaciones > Clientes



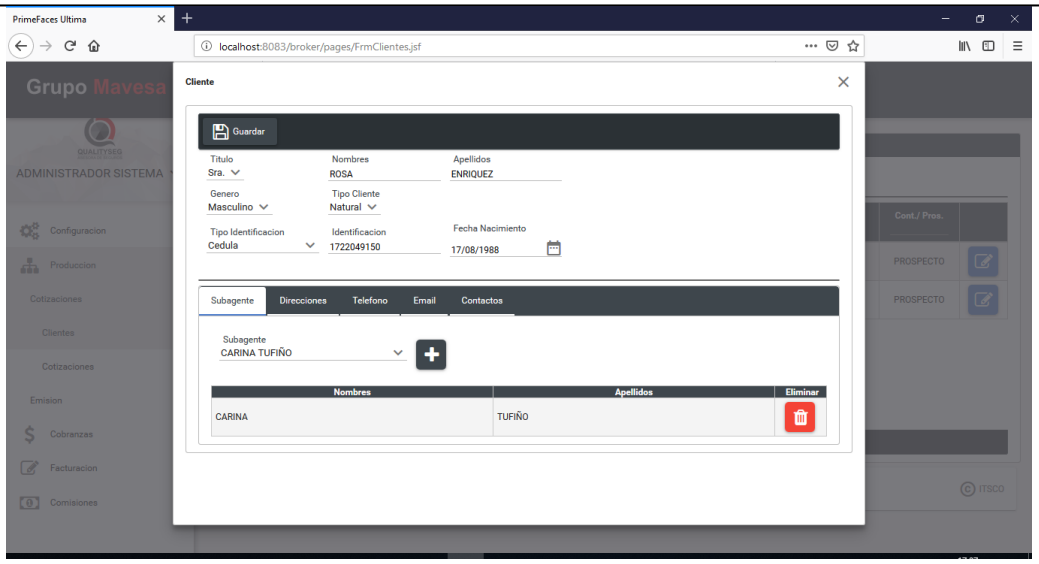
para agregar un nuevo cliente se debe dar clic sobre el botón agregar.

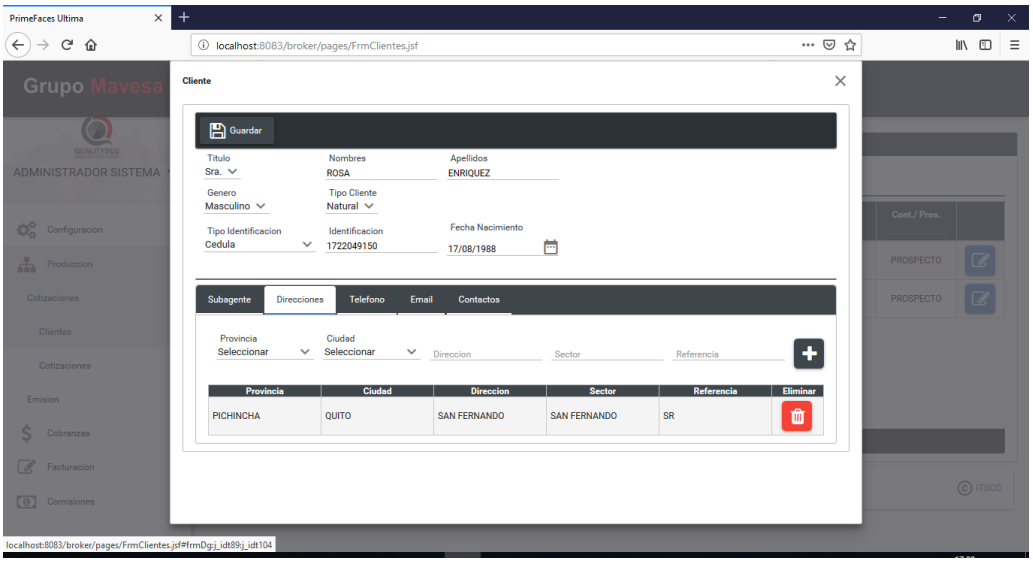


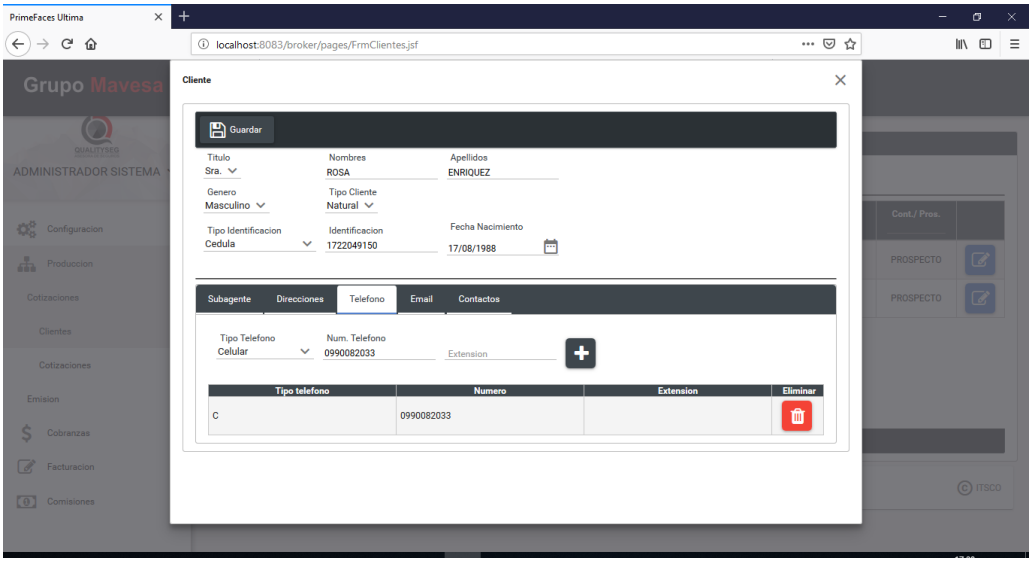
Se presentará un panel de registro.

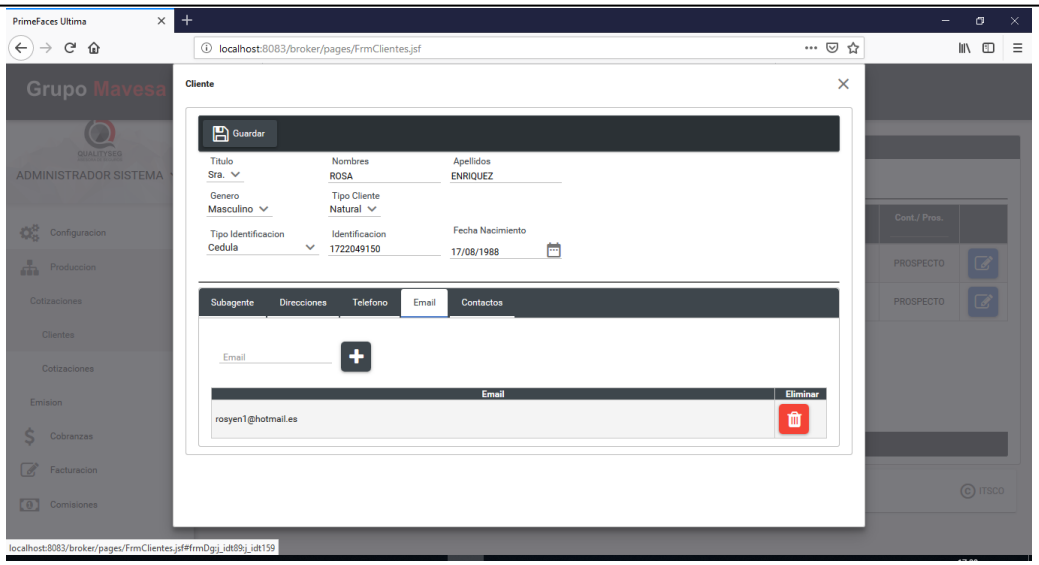


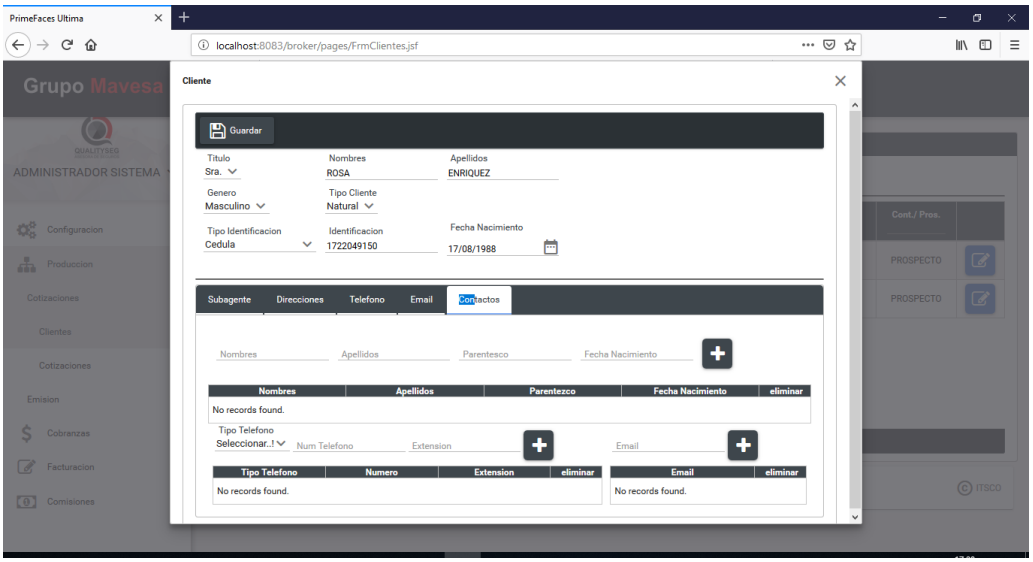
Al llenar los datos debe quedar de la siguiente manera:


PrimeFaces Ultima
localhost:8083/broker/pages/FrmClientes.jsf
Grupo Mavesa
Cliente
Guardar
Titulo
Sra.
Nombres
ROSA
Apellidos
ENRIQUEZ
Genero
Masculino
Tipo Cliente
Natural
Tipo Identificacion
Cedula
Identificacion
1722049150
Fecha Nacimiento
17/08/1988
Subagente
Direcciones
Telefono
Email
Contactos
Subagente
CARINA TUFIÑO
Nombres
Apellidos
Eliminar
CARINA
TUFIÑO


PrimeFaces Ultima
localhost:8083/broker/pages/FrmClientes.jsf
Grupo Mavesa
Cliente
Guardar
Titulo
Sra.
Nombres
ROSA
Apellidos
ENRIQUEZ
Genero
Masculino
Tipo Cliente
Natural
Tipo Identificacion
Cedula
Identificacion
1722049150
Fecha Nacimiento
17/08/1988
Subagente
Direcciones
Telefono
Email
Contactos
Provincia
Seleccionar
Ciudad
Seleccionar
Direccion
Sector
Referencia
Provincia
Ciudad
Direccion
Sector
Referencia
Eliminar
PICHINCHA
QUITO
SAN FERNANDO
SAN FERNANDO
SR

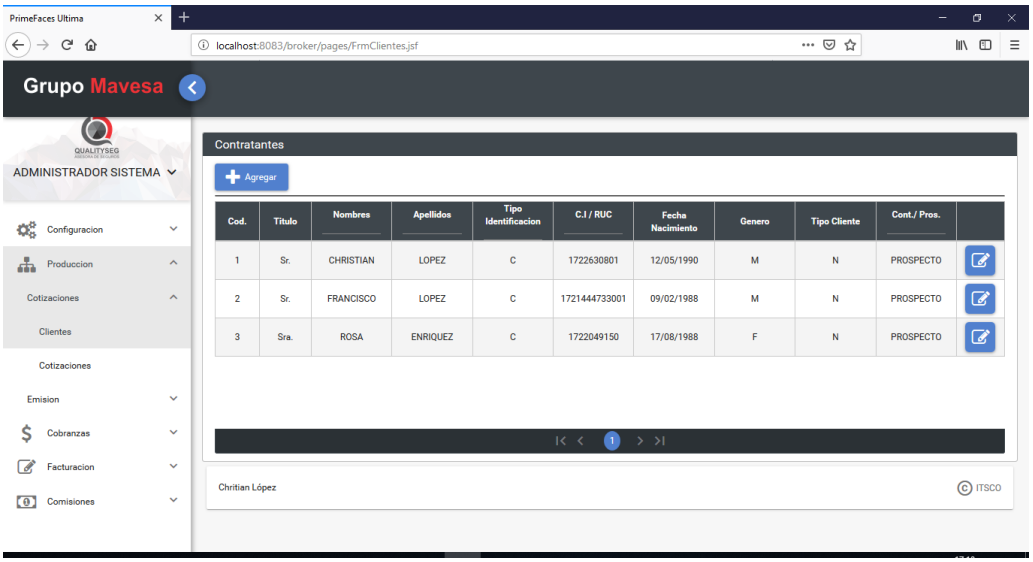

PrimeFaces Ultima
localhost:8083/broker/pages/FrmClientes.jsf
Grupo Mavesa
Cliente
Guardar
Titulo
Sra.
Nombres
ROSA
Apellidos
ENRIQUEZ
Genero
Masculino
Tipo Cliente
Natural
Tipo Identificacion
Cedula
Identificacion
1722049150
Fecha Nacimiento
17/08/1988
Subagente
Direcciones
Telefono
Email
Contactos
Tipo Telefono
Celular
Num. Telefono
0990082033
Extension
Tipo telefono
Numero
Extension
Eliminar
C
0990082033





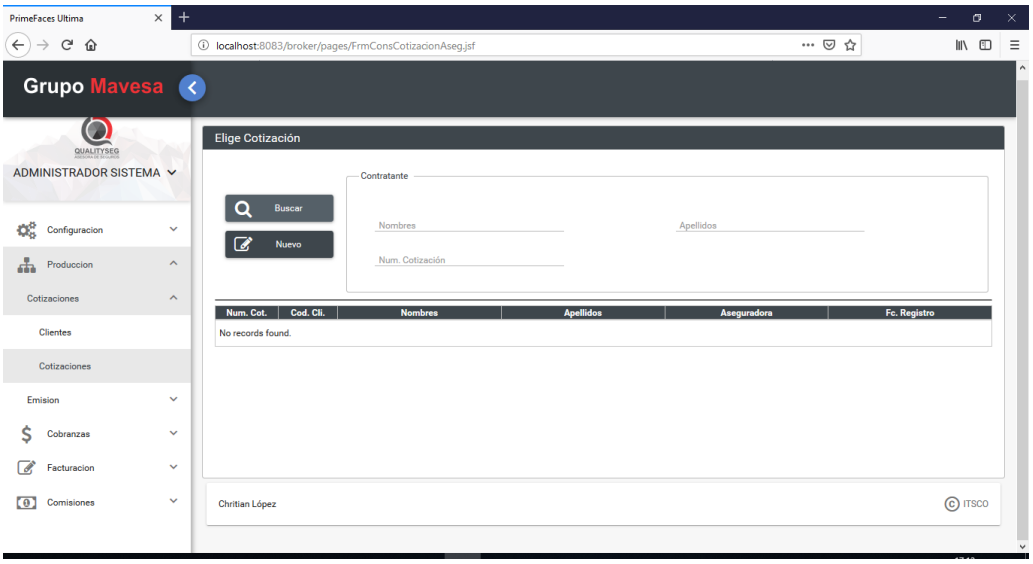
Finalmente presionar el botón grabar

Se registrar el nuevo cliente y se actualiza la tabla de clientes.

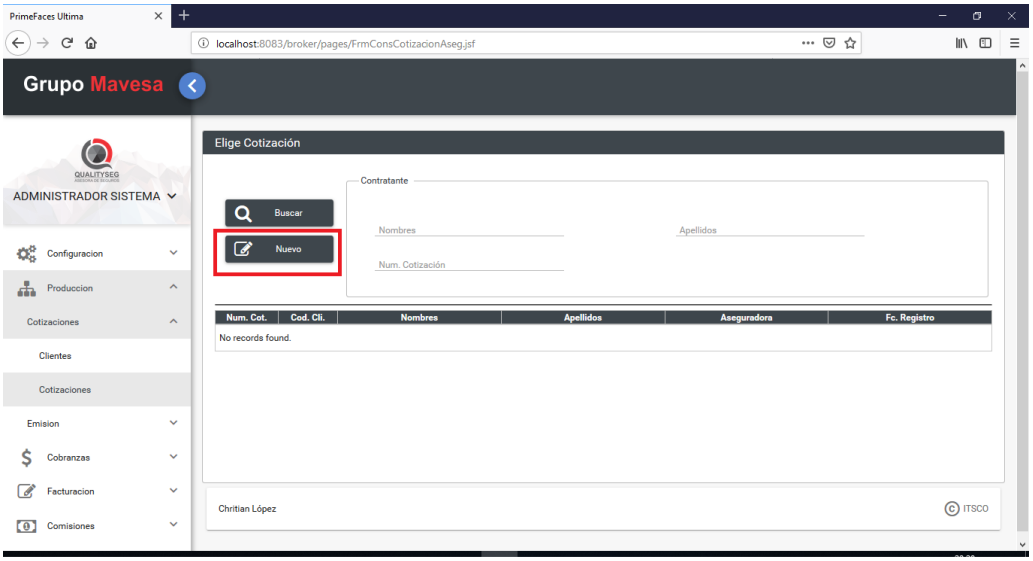


Cotizaciones.

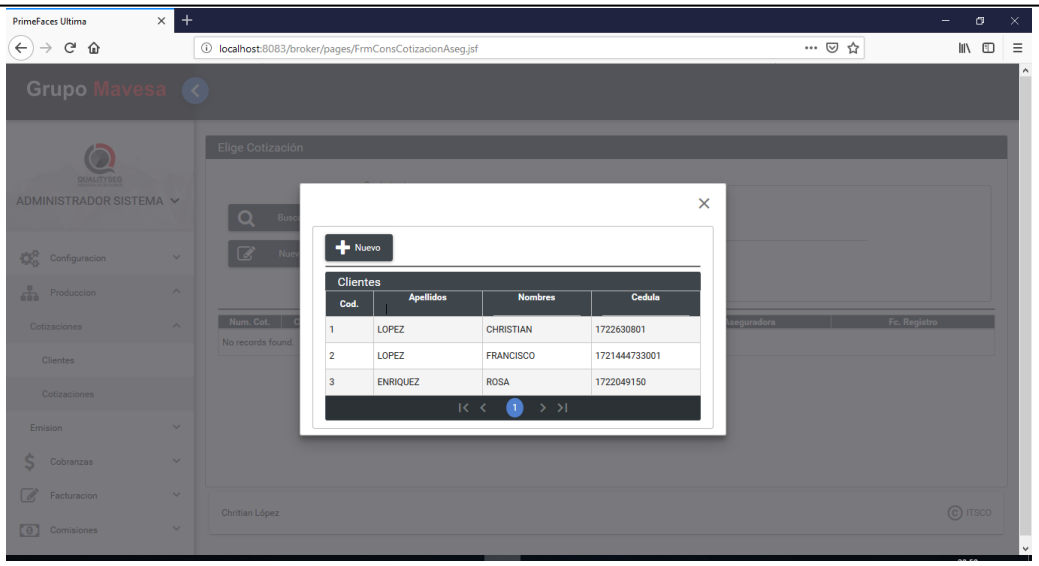
Para buscar una cotización o registrar una nueva se debe ir al menú Producción> Cotizaciones> Cotizaciones.



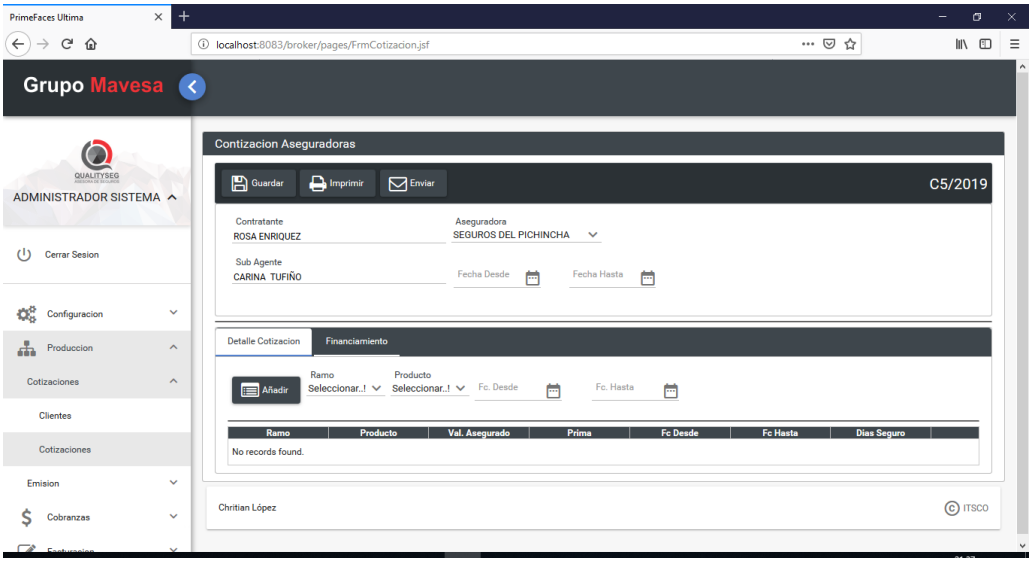
Para crear una nueva cotización se debe dar clic sobre el botón Nuevo



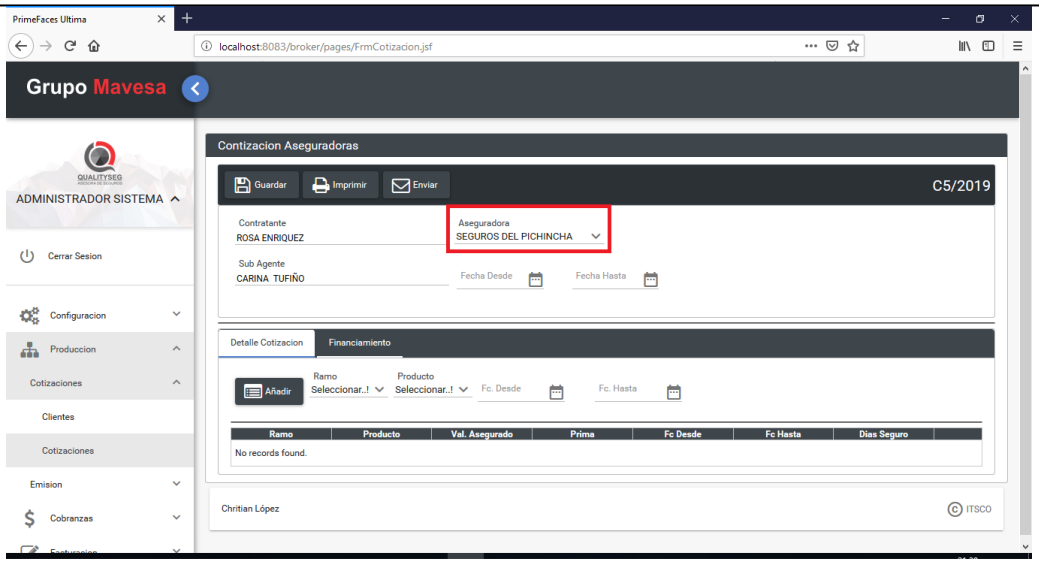
Se presentará un panel con la lista de clientes.



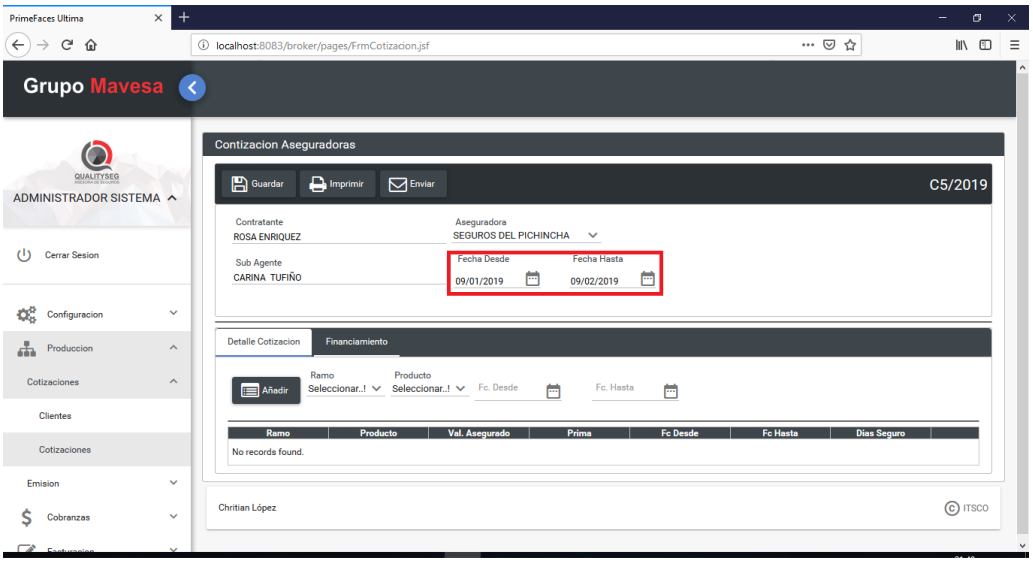
Si el cliente existe se lo debe seleccionar, esto redireccionara a la pantalla de cotización.



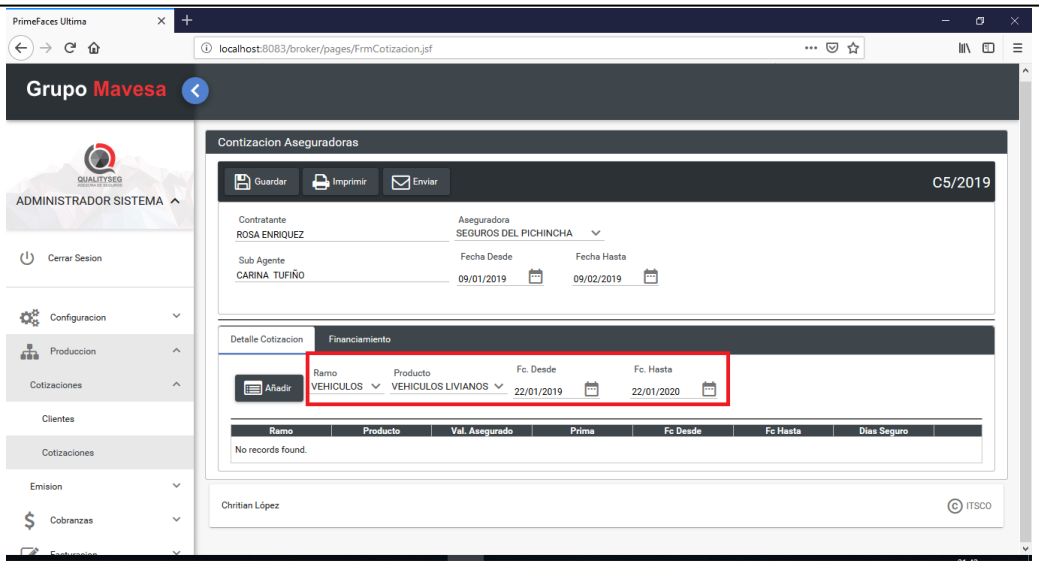
Una vez ingresados a la pantalla de cotización, lo primero a realizar es seleccionar la compañía de seguros.



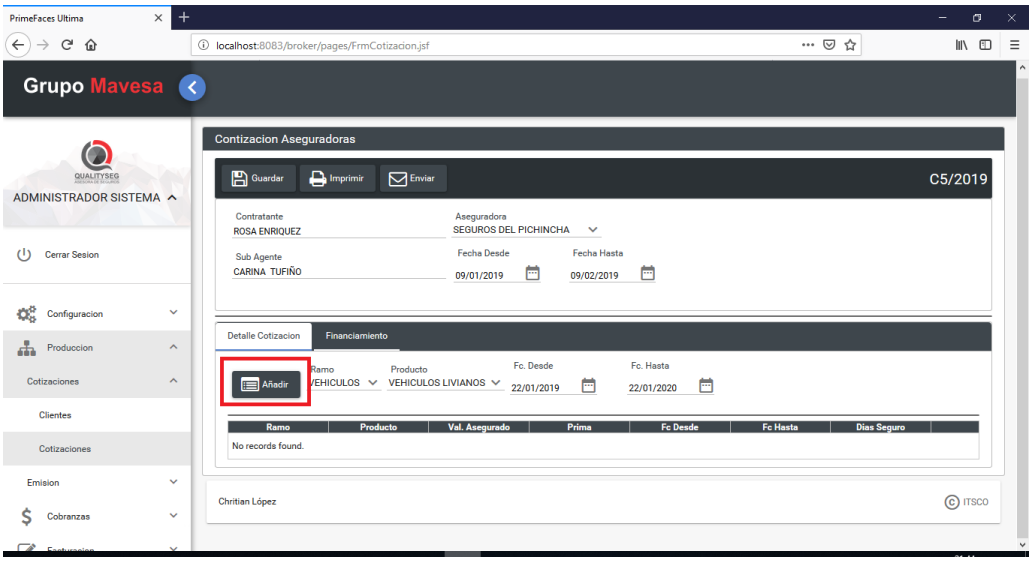
Se presentará dos fechas las cuales son la vigencia de la cotización al ingresar la fecha de inicio se calcula automáticamente un mes de vigencia para la cotización que es lo que las compañías dan de plazo de vigencia en una cotización.



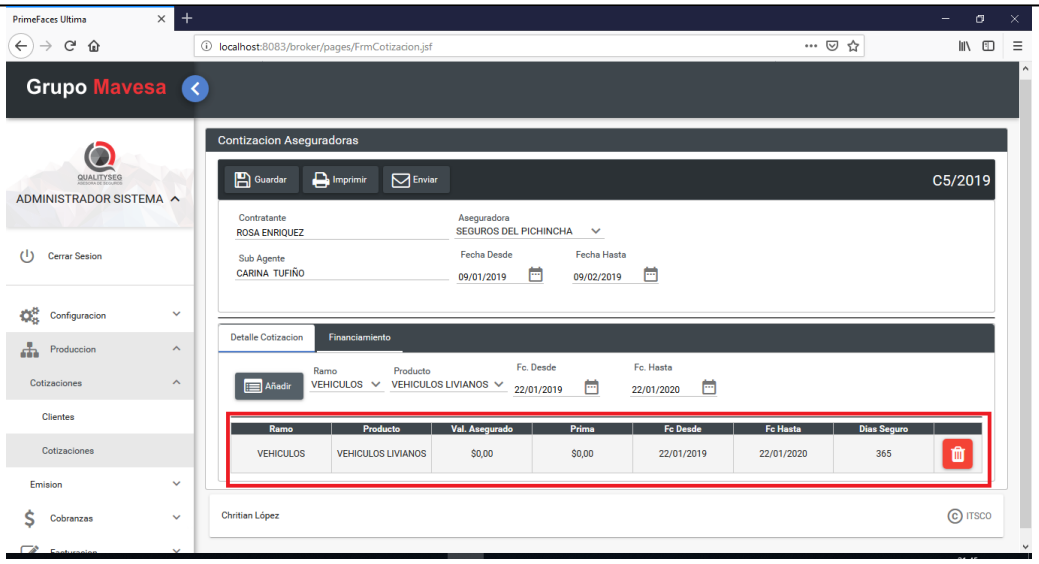
Luego se debe registrar el detalle de la cotización, es decir el ramo asegurado con el producto y la vigencia que tendrá el seguro del cliente.



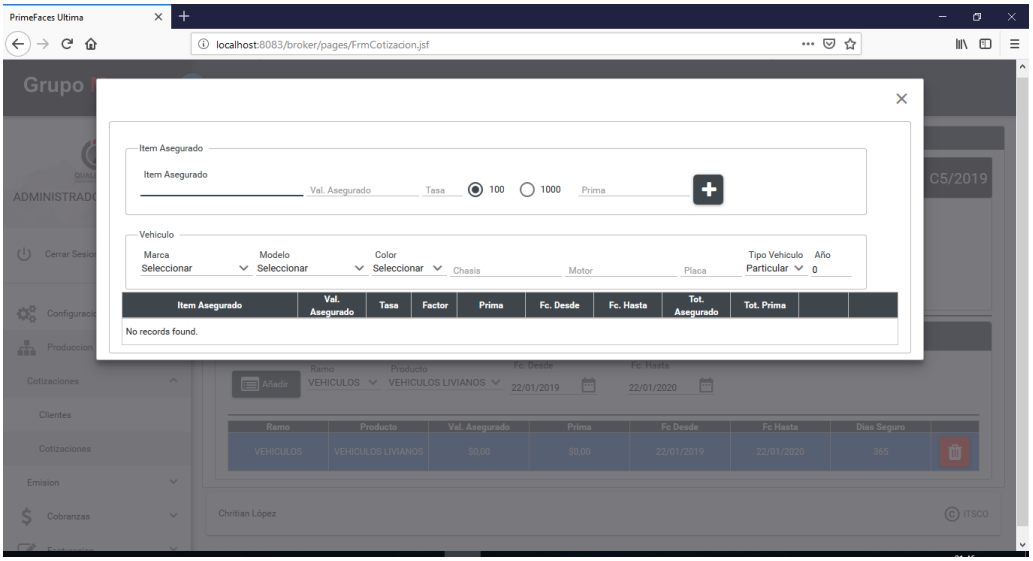
Al ingresar los datos se debe presionar el botón añadir:



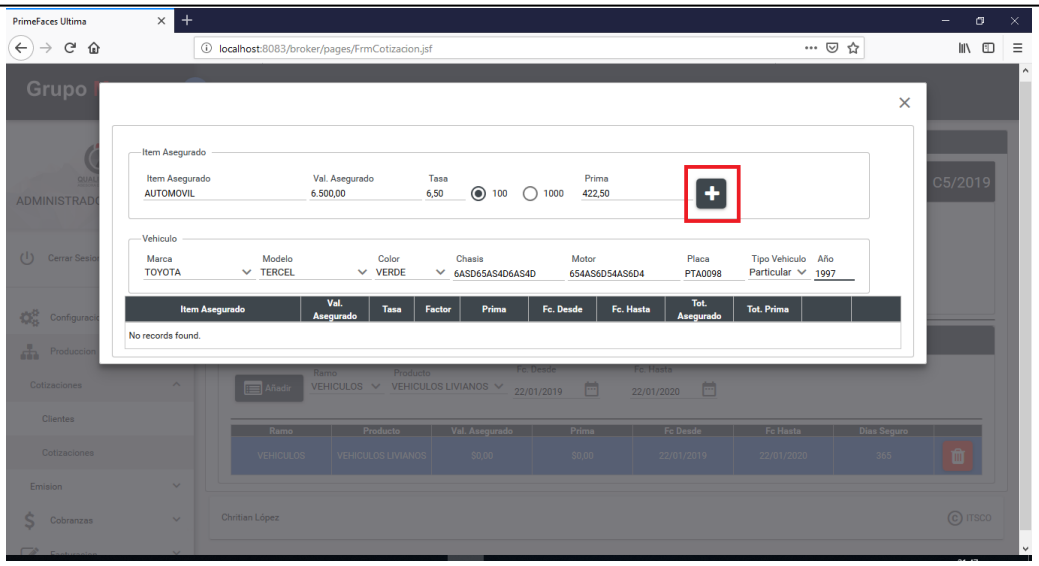
Esta acción agregara los datos a la tabla de detalle de cotización.



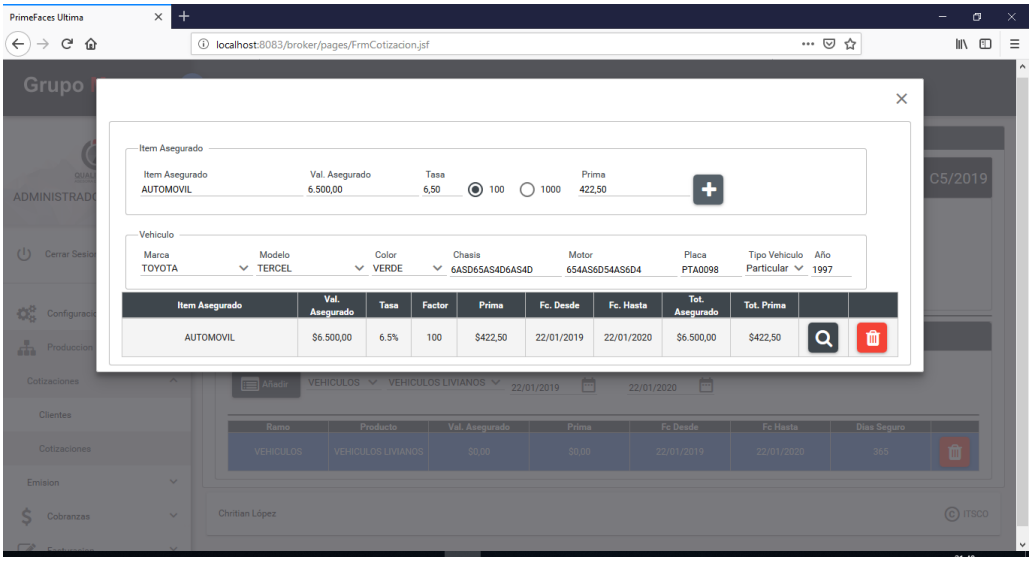
Para poder ingresar el objeto asegurado se debe dar clic sobre el registro de detalle, esta acción presentara un panel para poder registrar los N número de objetos asegurados que tenga la póliza del cliente.



Al registrar los datos se debe presionar el botón en pantalla para que se registre el objeto asegurado.

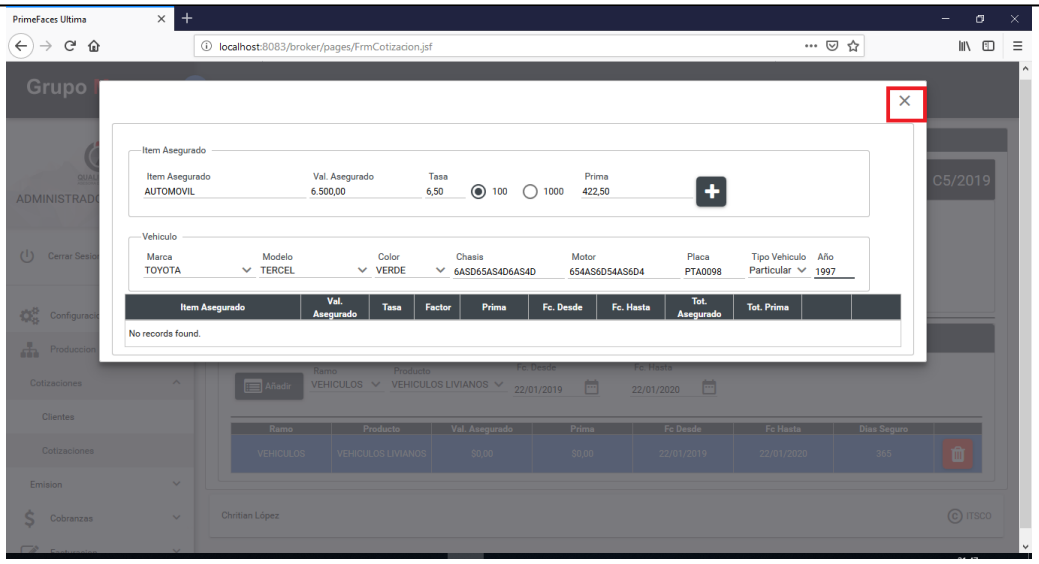


Esta acción alimentara la tabla de objetos asegurados

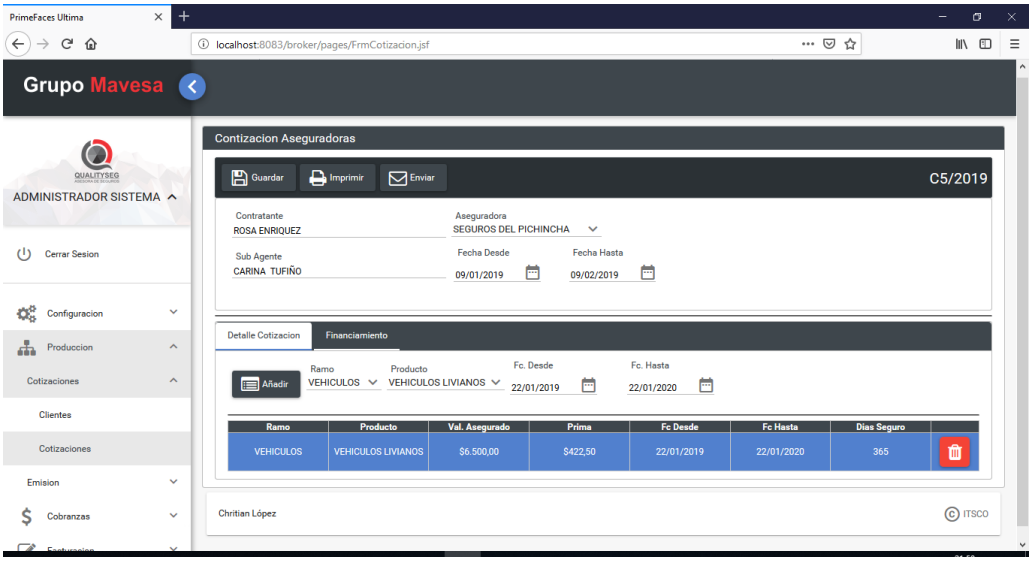


En caso de querer agregar más objetos se debe repetir la acción.

Para regresar se presiona la tecla Esc. O en la X de la esquina superior.

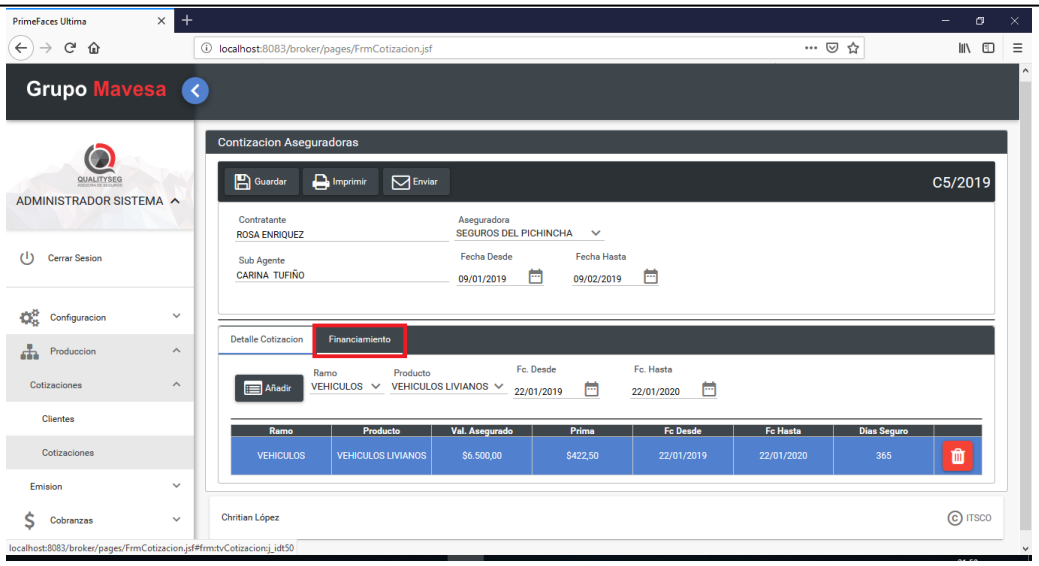


Como resultado regresamos a la pantalla de cotización.

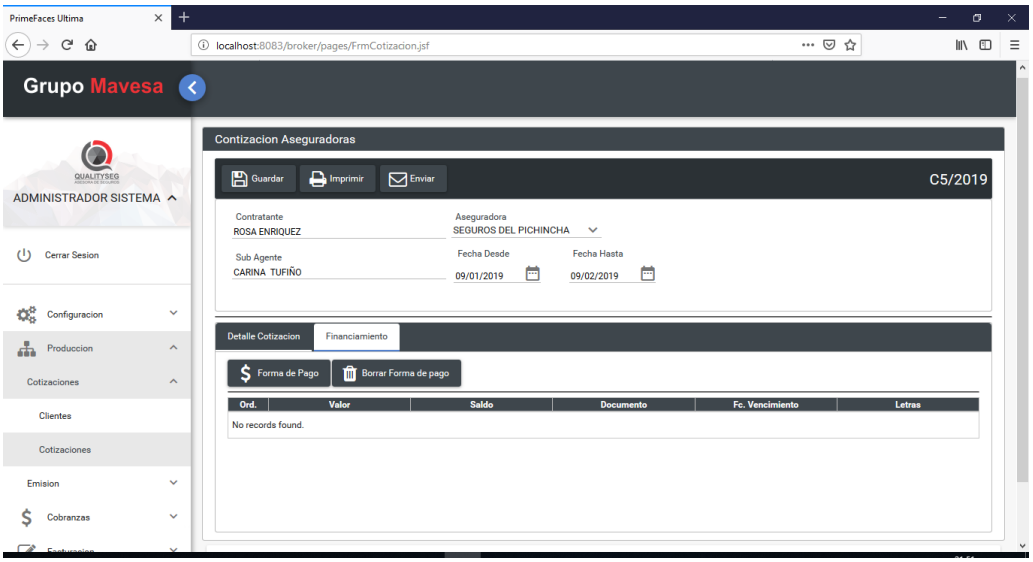


dicha pantalla se encuentra actualizada los valores de prima y valor asegurado en base a los objetos asegurados registrados.

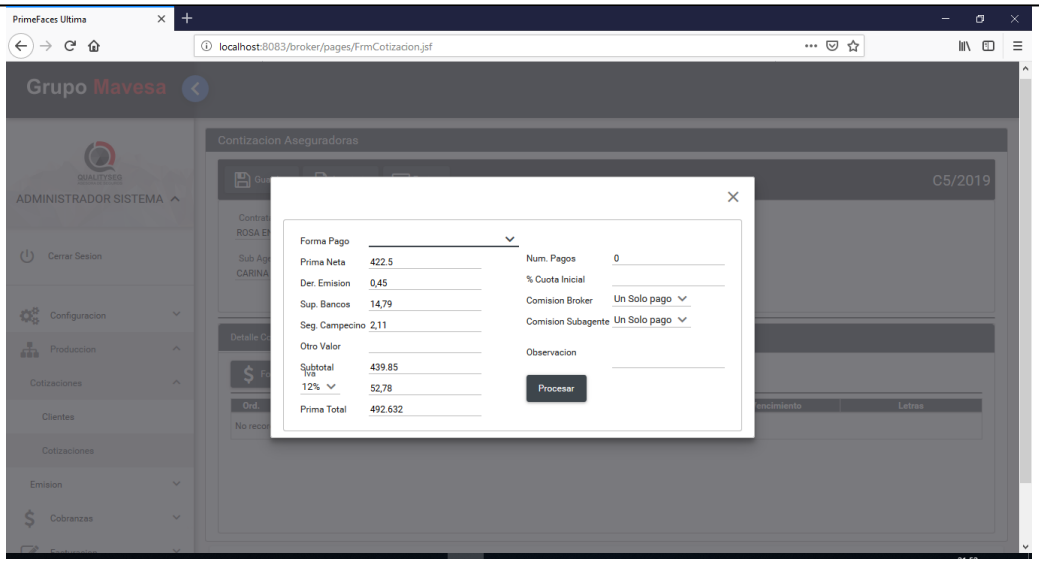
Para agregar la forma de pago o el financiamiento que desee el cliente se debe ir a la pestaña financiamiento.



Al ingresar se presenta un botón para registrar el financiamiento del cliente.

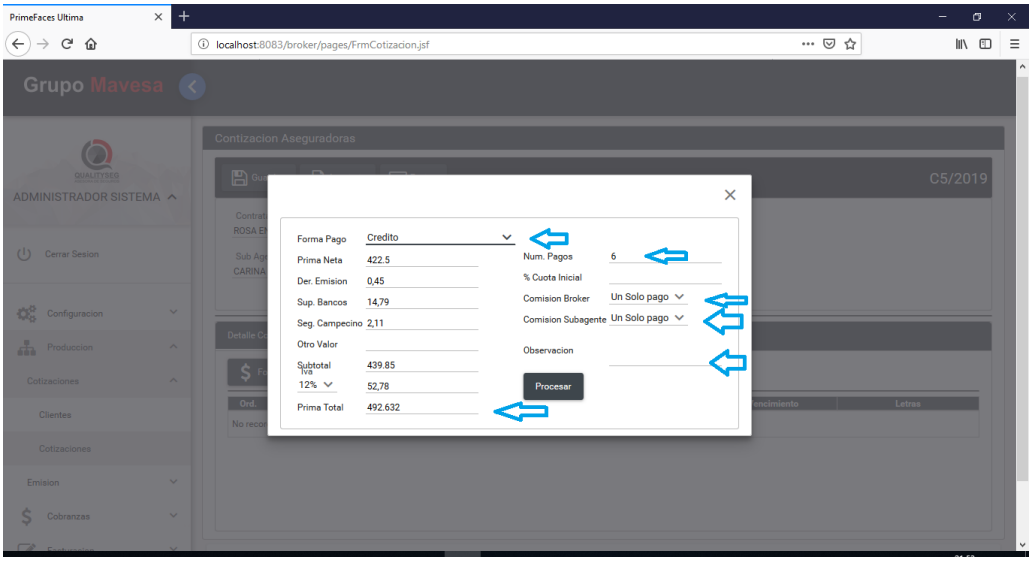


Dar clic en el botón forma de pago y se presentara un panel que muestra los valores de impuestos y el total que pagara el cliente por su seguro.

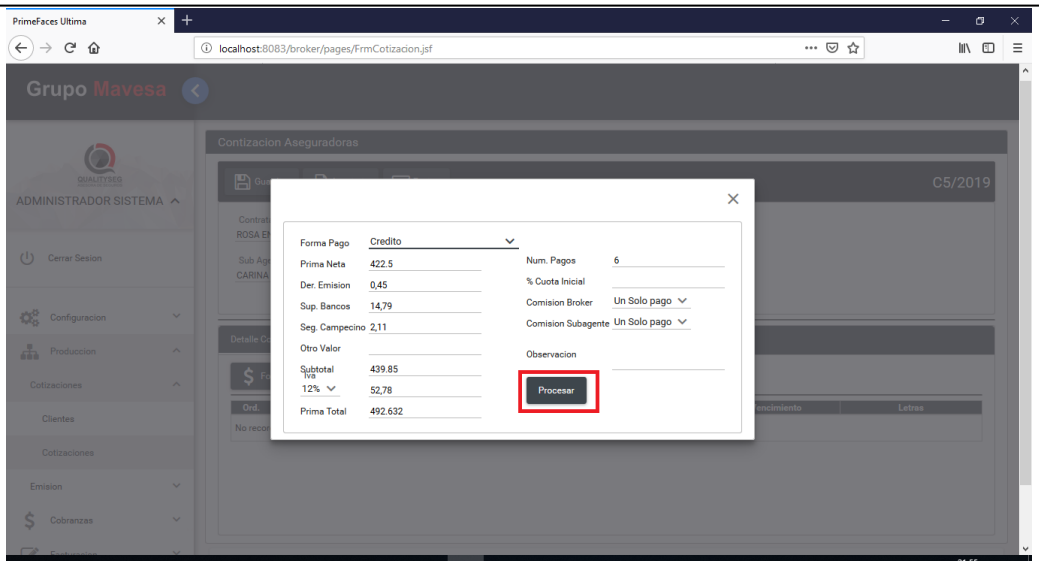


En dicha pantalla se debe seleccionar la forma de pago que realizara el cliente y de ser el caso el número de pagos, como se cobrara la comisión y una observación.

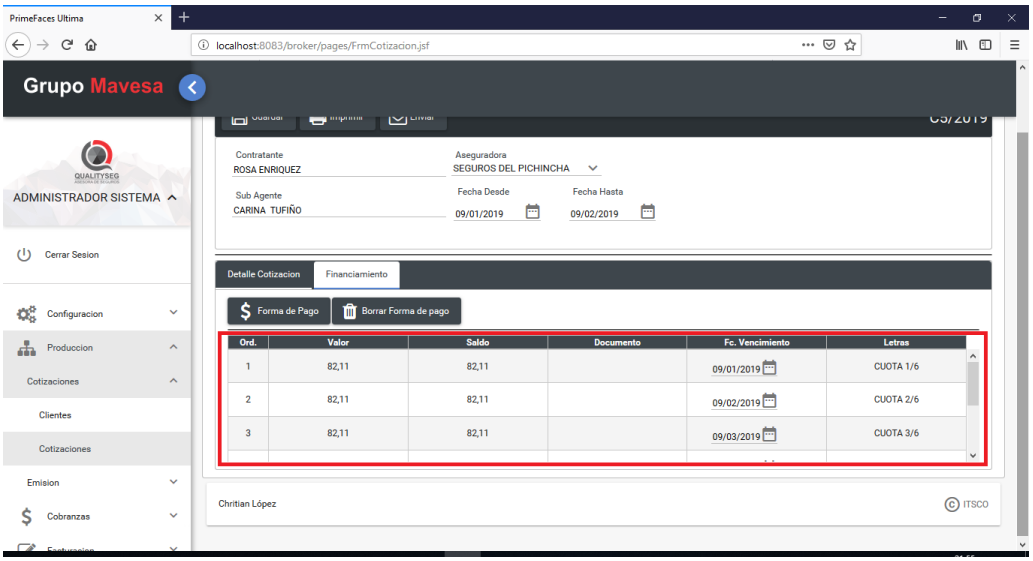
Aquí se mostrará el valor total a pagar por el cliente.



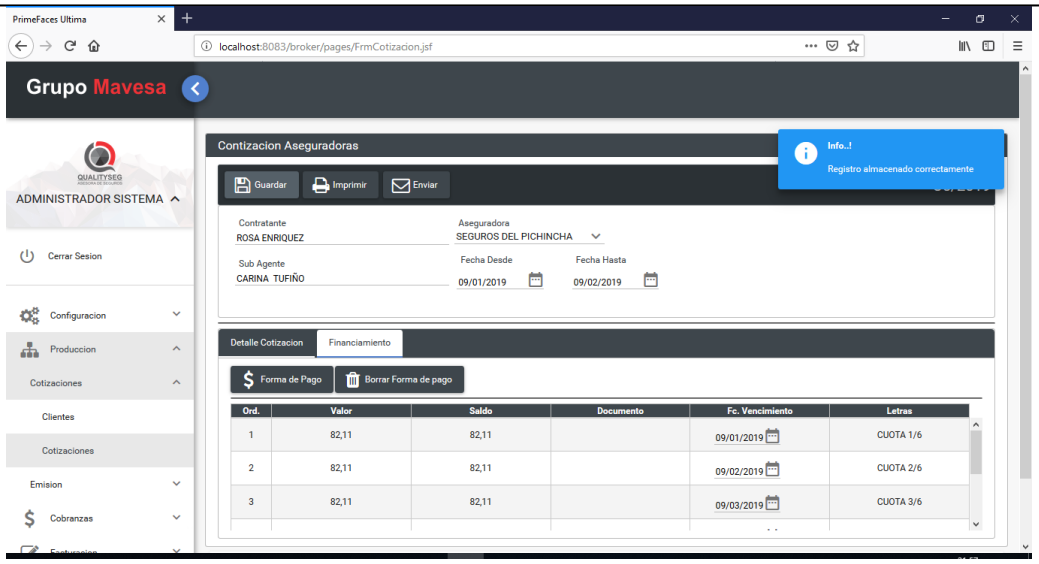
Para poder registrar se debe presionar el botón procesar.



Esta acción mostrara el financiamiento escogido por el cliente con los valores a pagar y su fecha de vencimiento de cada cuota.

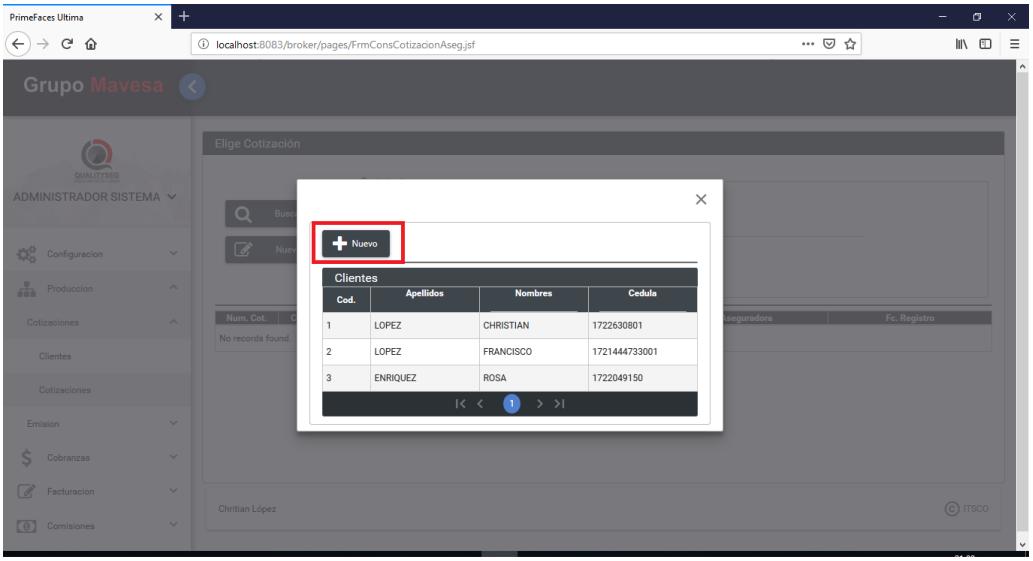


Para poder finalizar se debe guardar la información.

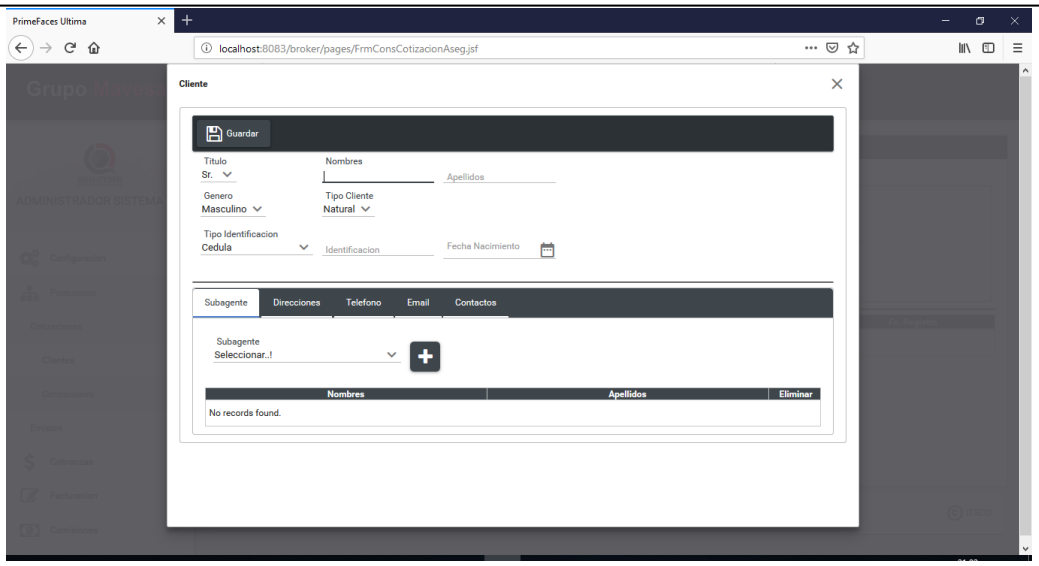


Presentará un mensaje de información indicando que todo se encuentra guardado.

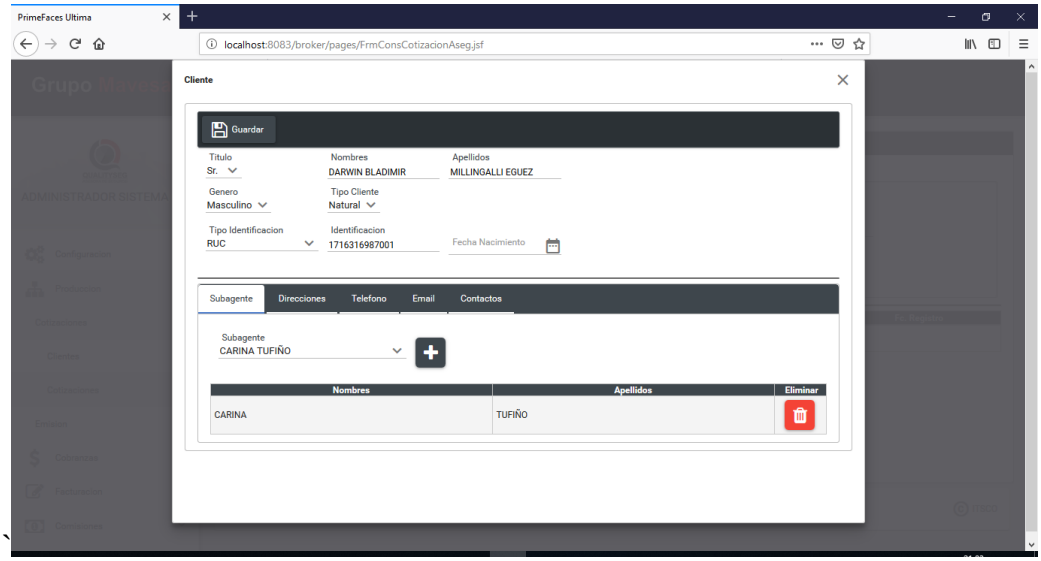
En caso de que un cliente no se encuentre registrado en la lista de contratantes al momento de registrar una cotización, se debe dar clic sobre el botón nuevo.



Se presentará el panel de registro de clientes

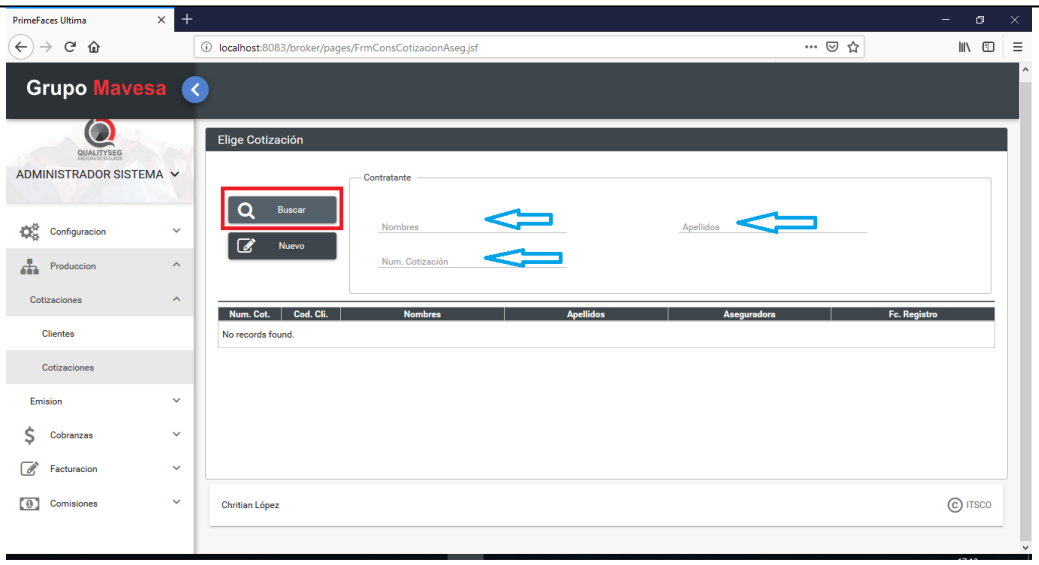


El cual se deberá registrar los datos del cliente para que pueda ser creado.

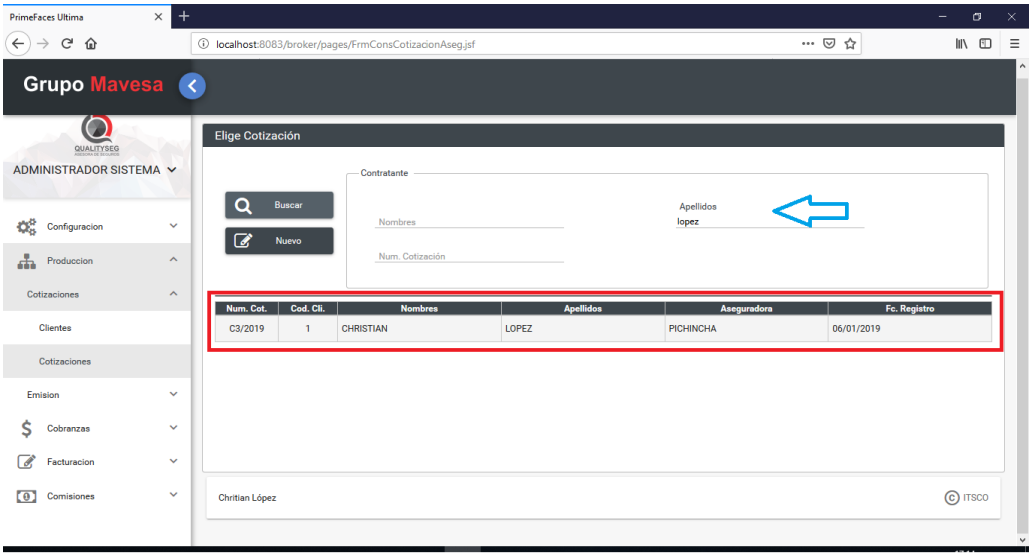


Al ingresar los datos y guardar la información se redirecciona a la pantalla de cotización y se deberá realizar el proceso anterior

Si se dispone de cotizaciones pendientes se debe ingresar los parámetros de nombre, apellido o número de cotización para realizar la búsqueda, en caso de que no se disponga solo se presiona buscar



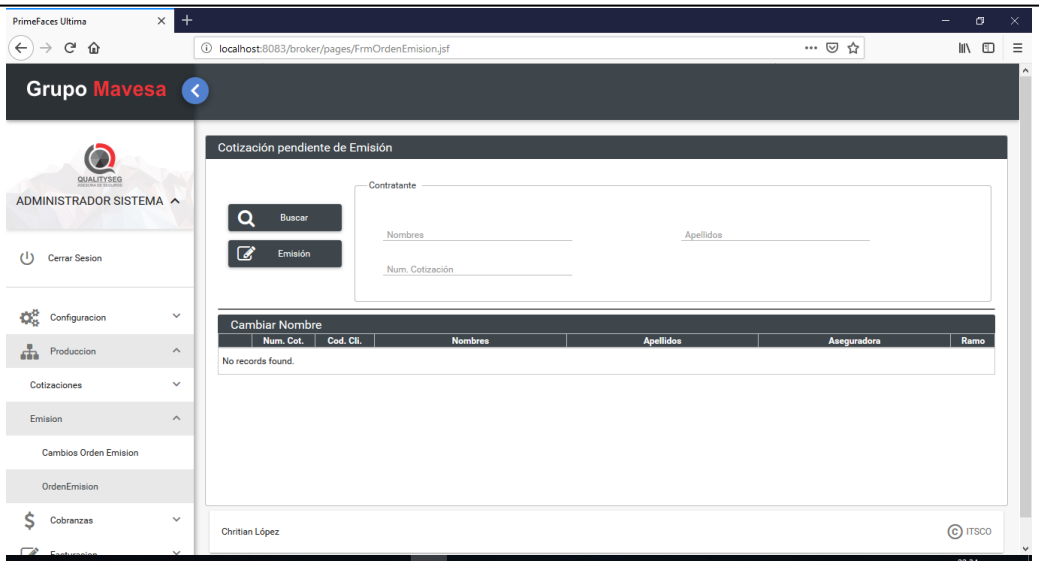
En caso de encontrar información con los parámetros ingresados, se desplegará en la tabla de cotizaciones.



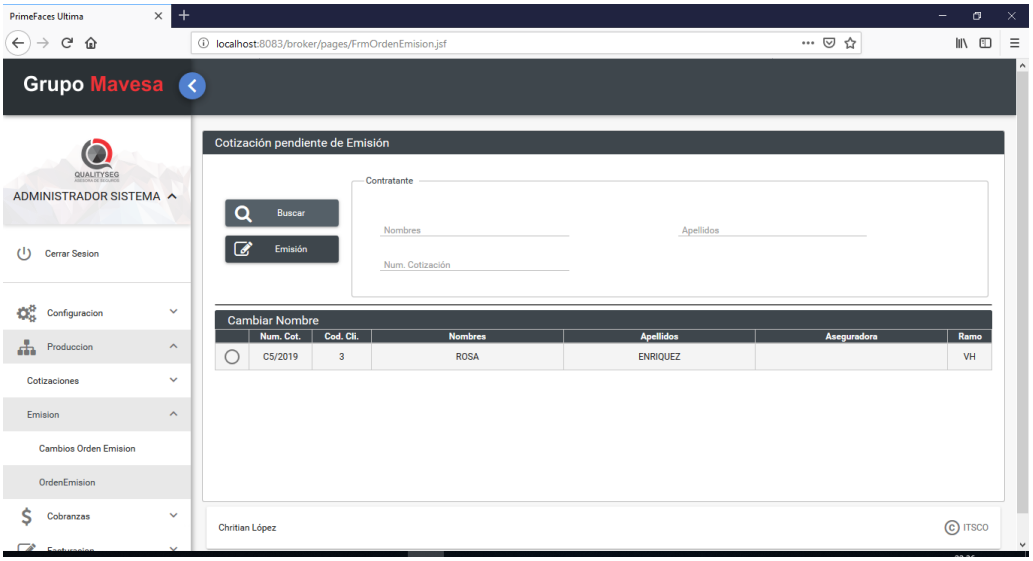
Se da clic en el registro que se encontró y se redirecciona a la pantalla de cotización y realizar el proceso de registro de cotización.

Orden de Emisión

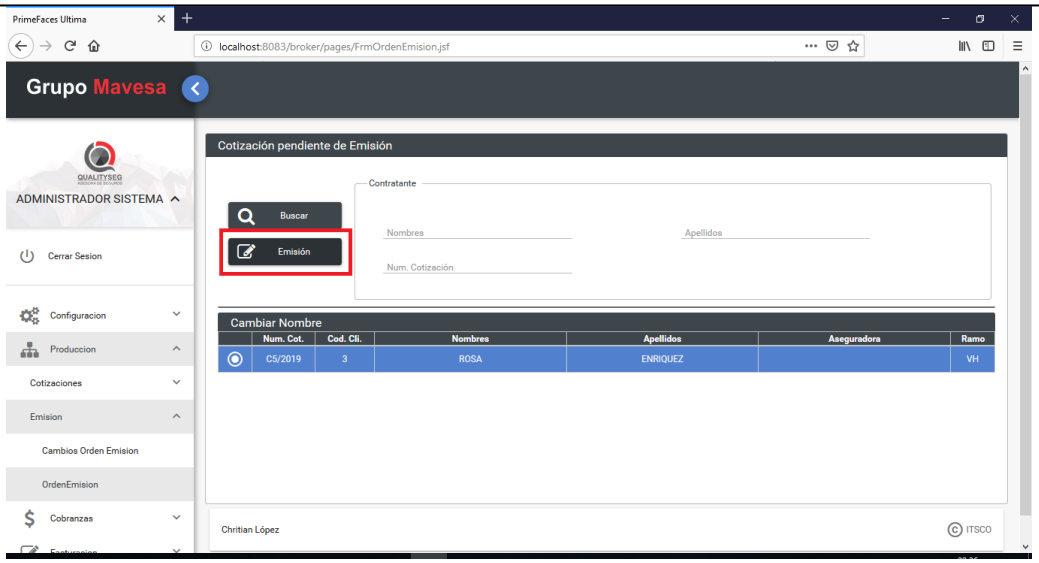
Para poder emitir una póliza se debe ir en el menú Producción > Emisión > Orden de emisión.



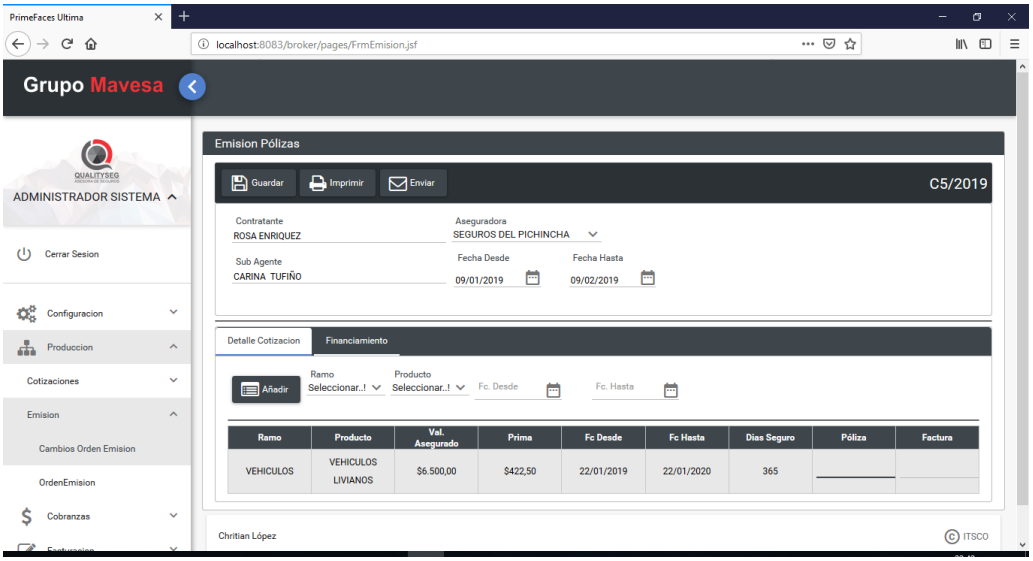
En la presente pantalla se podrá buscar todas las cotizaciones pendientes que no hayan sido emitidas o que estén pendientes de emisión, para ello se debe ingresar los datos de los parámetros o si deseamos generar una consulta global solo presionar el botón buscar.



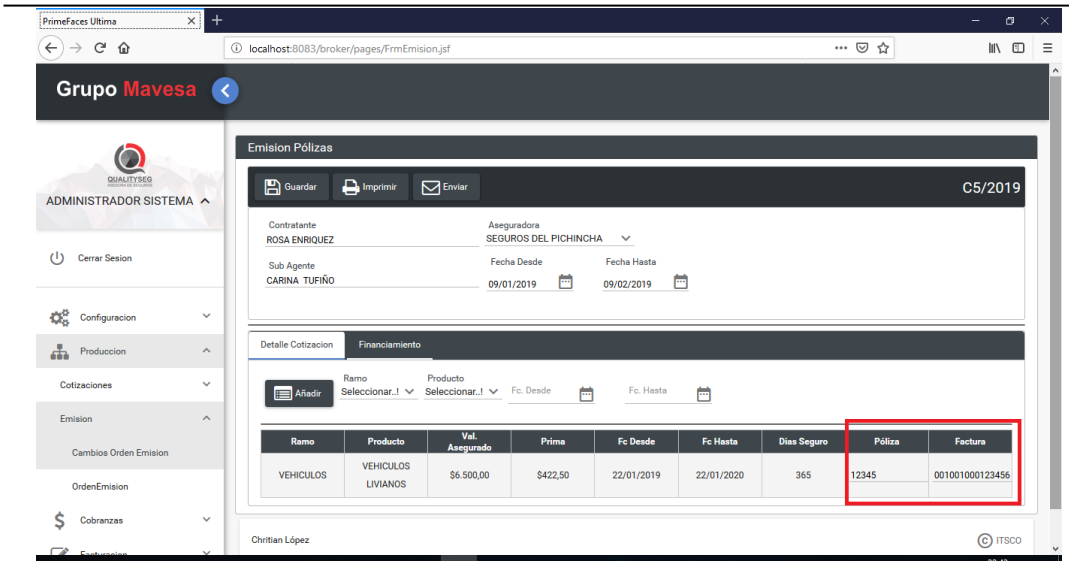
Se presentará el registro que se desea emitir, se lo selecciona y se da clic en el botón emisión.



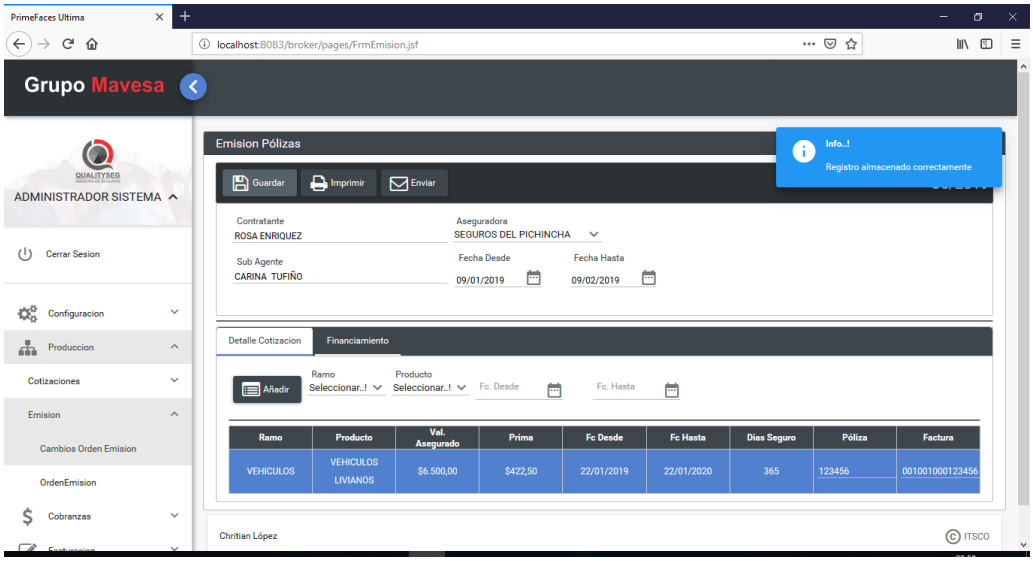
Esta acción redireccionara a la pantalla de emisión, similar a la pantalla de cotización.



En la tabla de detalle cotización, se debe registrar el número de póliza y factura que proporciona la compañía de seguros, con estos dos datos se completa la emisión.



Al guardar la información la póliza queda emitida en el sistema.



Anexo H. Manual de Instalación.

CARRERA ANÁLISIS DE SISTEMAS

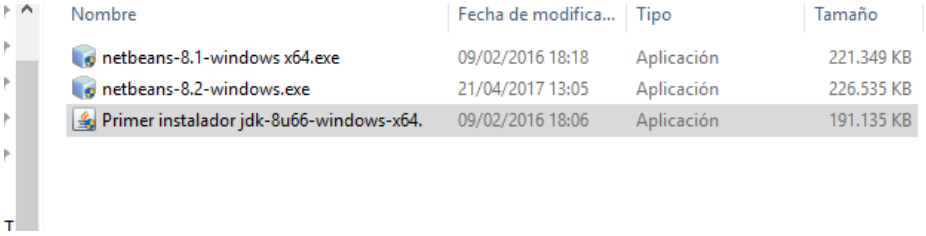
MANUAL DE INSTALACIÓN

AUTOR: CHRISTIAN JAVIER LOPEZ VITERI.

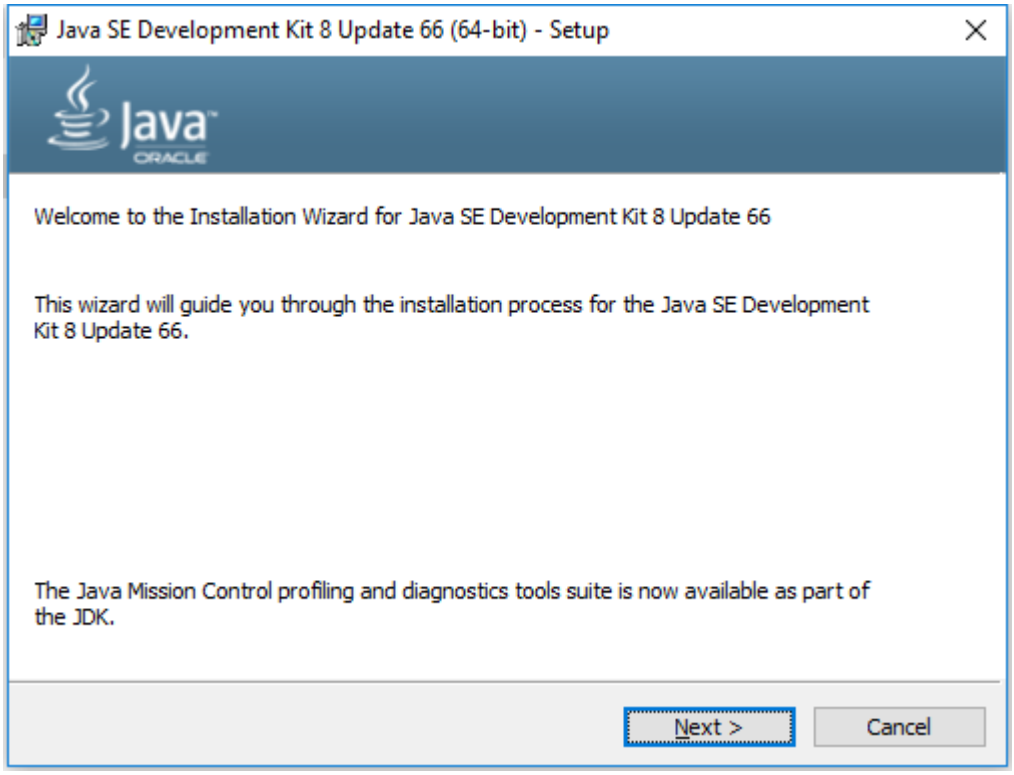
Quito, 2019

Instalación IDE de programación NetBeans 8.1

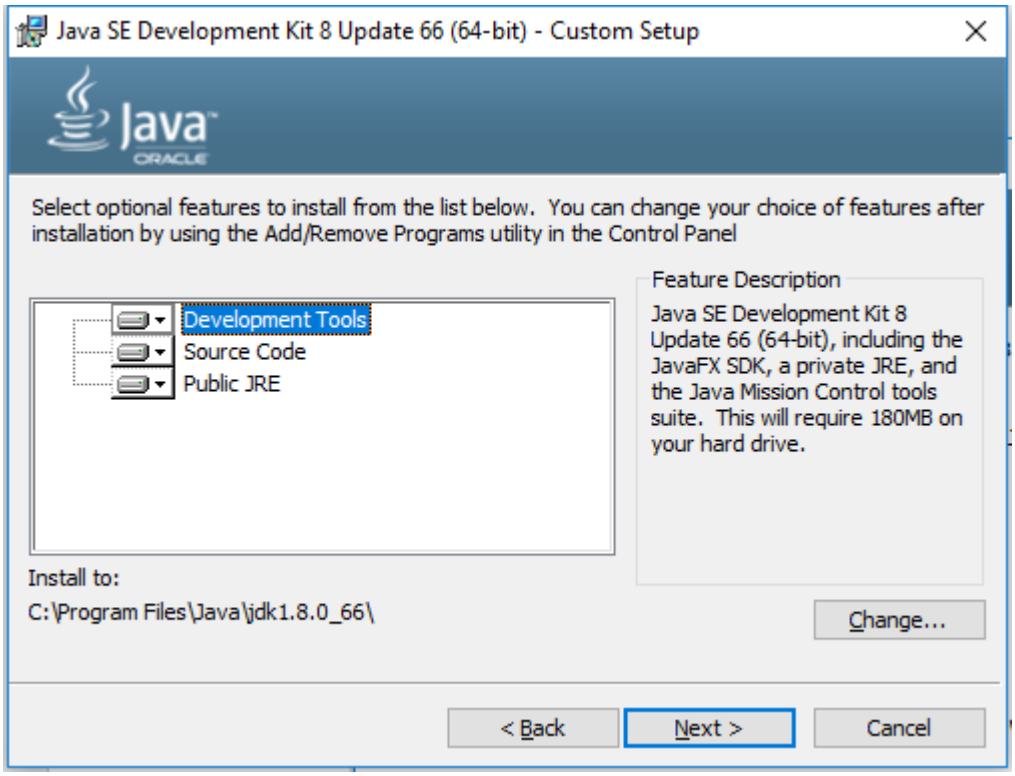
1. Primero se debe instalar la herramienta jdk de java.



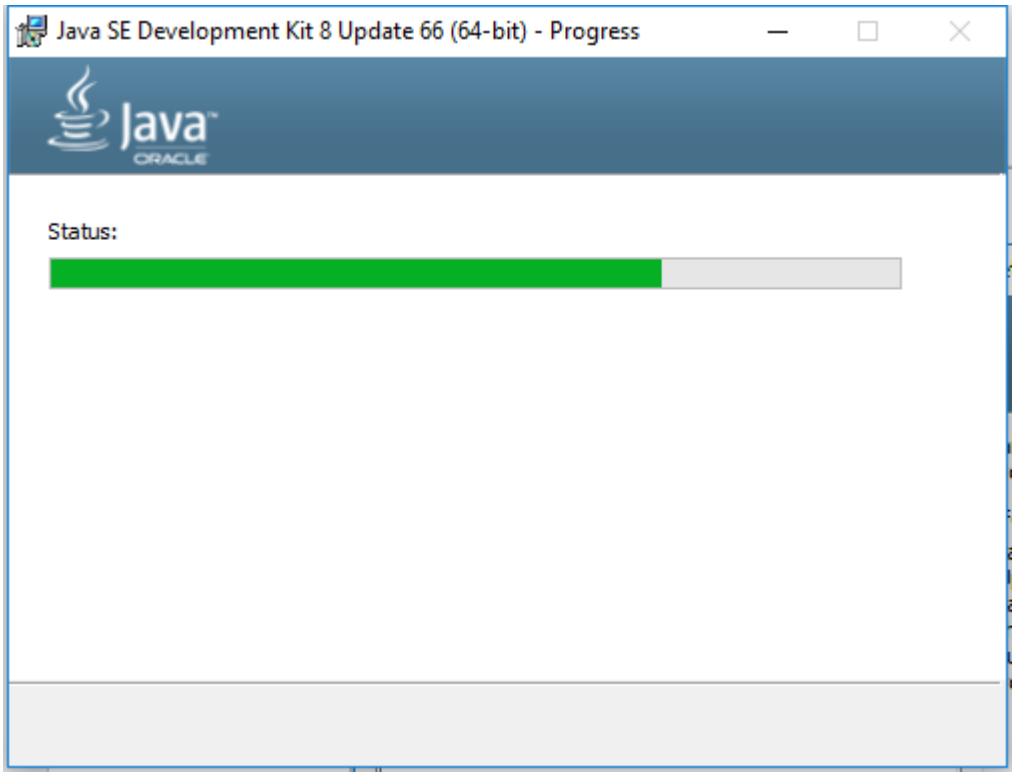
2. Presionamos siguiente.



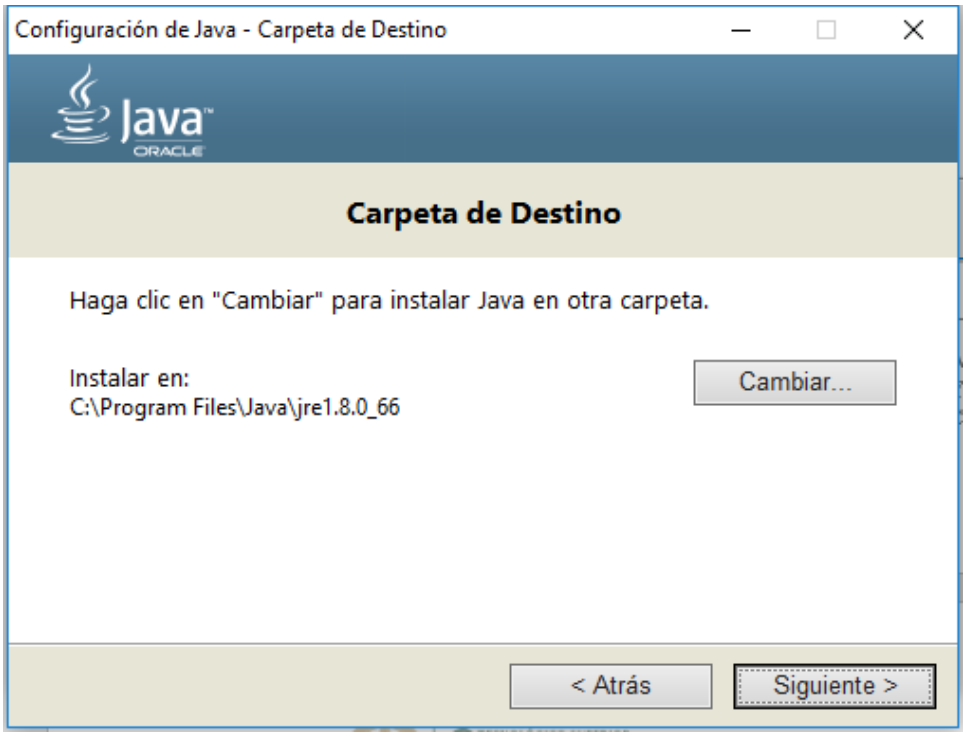
3. Nuevamente presionamos siguiente.



4. Se mostrará el estatus de la instalación.

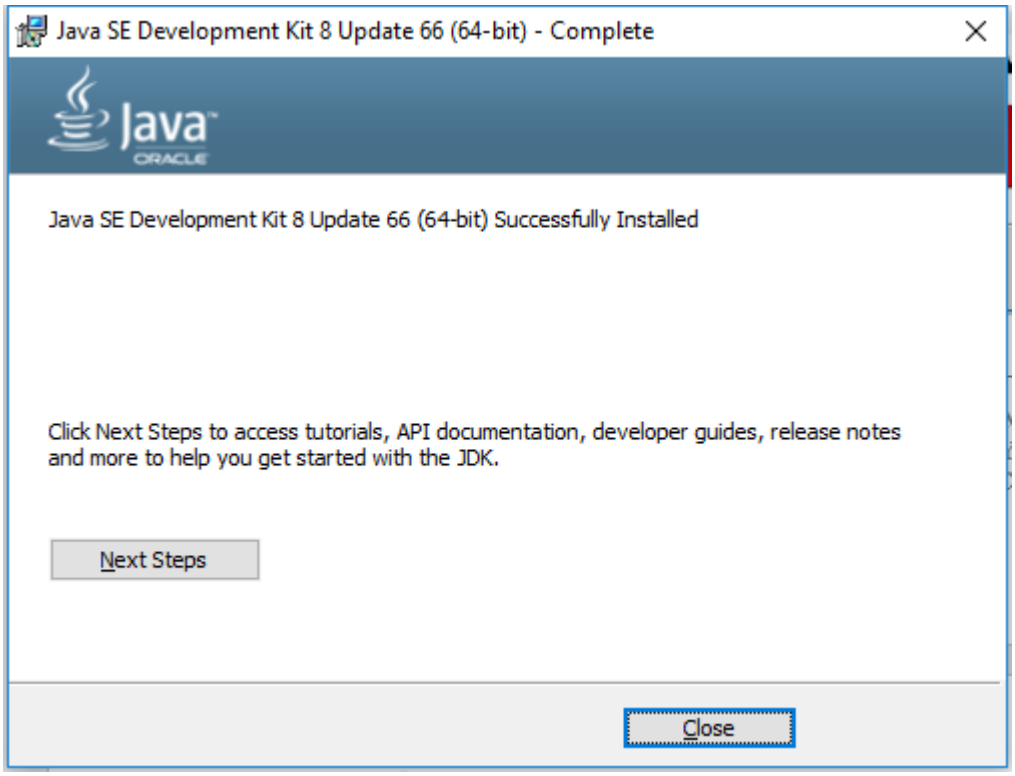


5. Se indica que se instalara el JRE de java, mantenemos la ubicación por default.

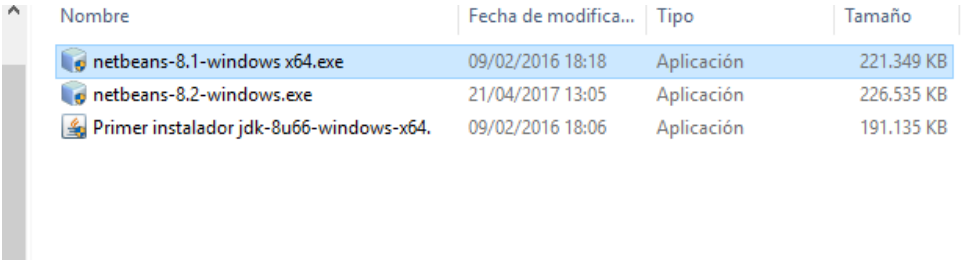


6. Presentará el progreso de instalación.

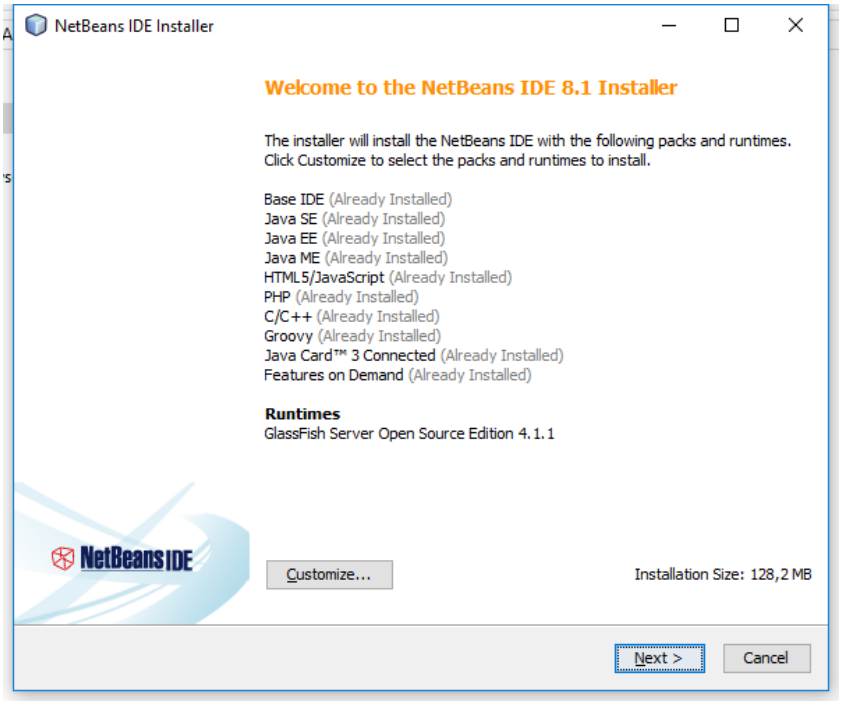
7. Al concluir la instalación presionamos cerrar.



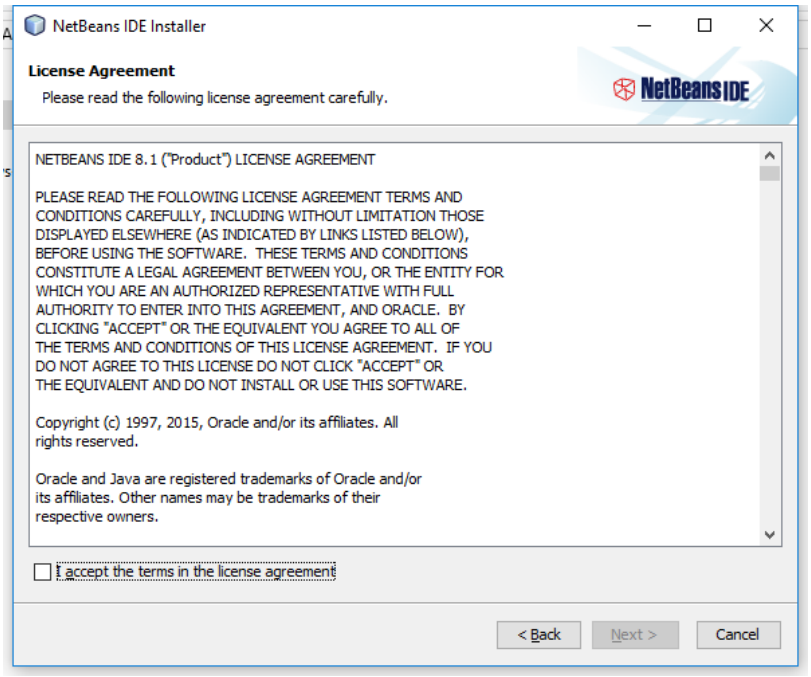
8. Ahora procedemos con la instalación del IDE NetBeans.



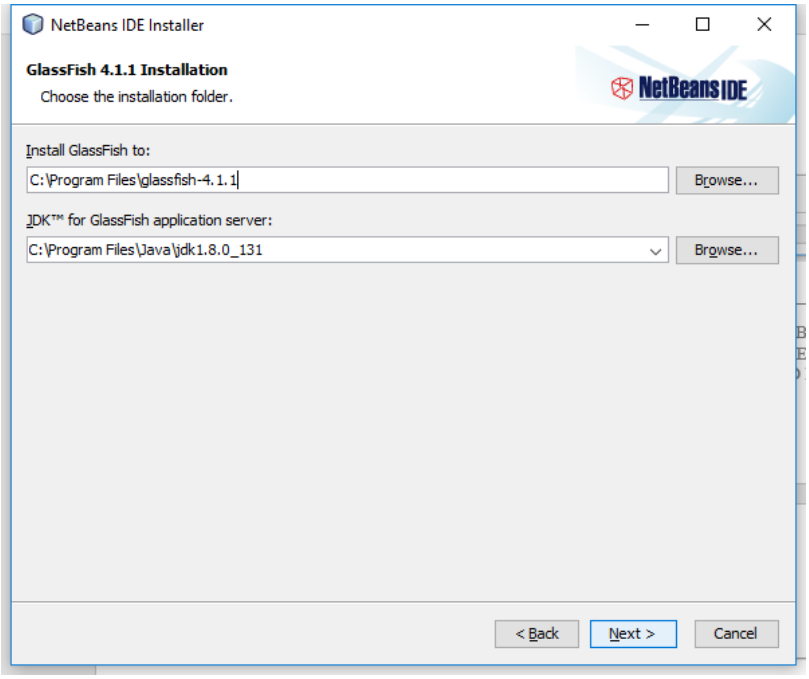
9. En caso de tener instalado se presenta los componentes instalados, de ser necesario se puede agregar más componentes.



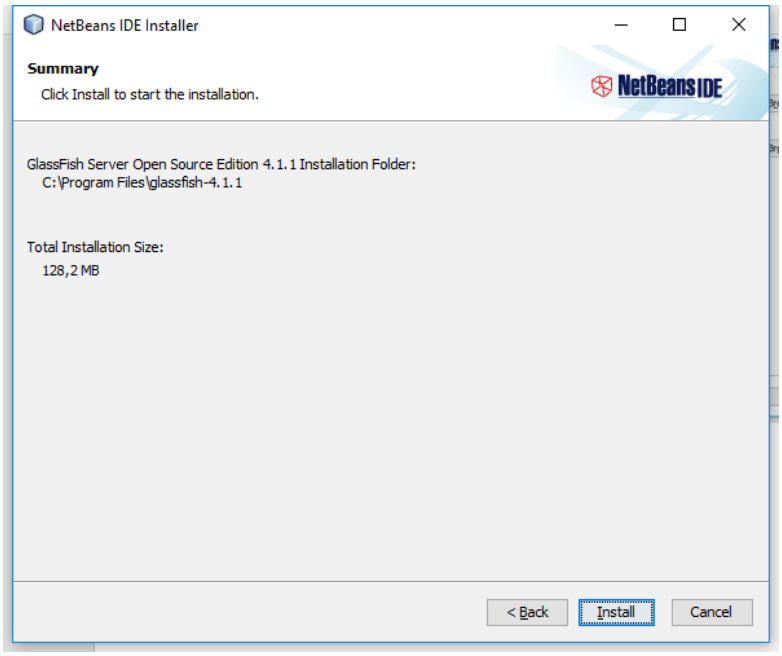
10. Al presionar siguiente se presenta los términos de licenciamiento de NetBeans, aceptamos los términos y presionamos siguiente.



11. Presentará la ubicación del servidor de aplicación Glasfish y el JRE de java.

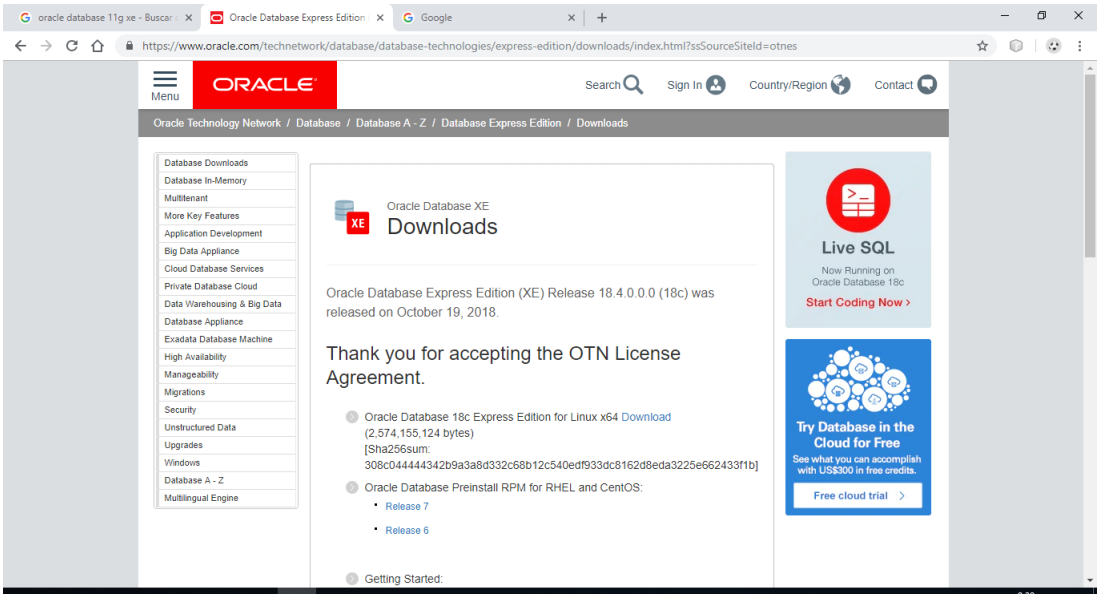


12. Al presionar siguiente presenta el tamaño total de instalación, solo se debe presionar instalar.

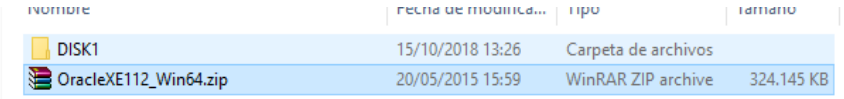


Base de datos Oracle xe 11G.

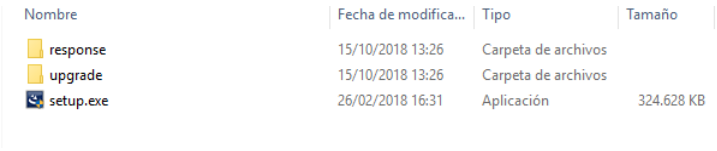
1. Se debe descargar el instalador desde la página oficial de Oracle.



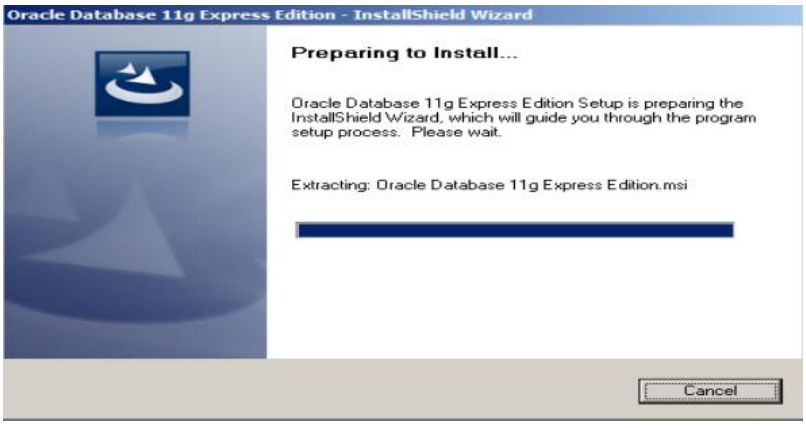
2. Posterior a esto se debe ejecutar el instalador descargado



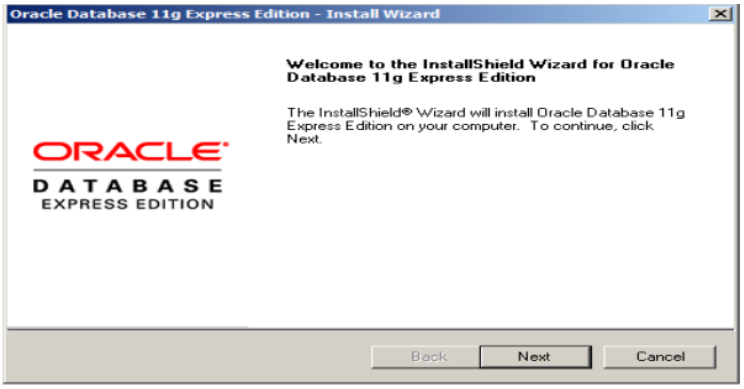
3. Ejecutamos el setup.exe.



4. Se presentará el panel de instalación.

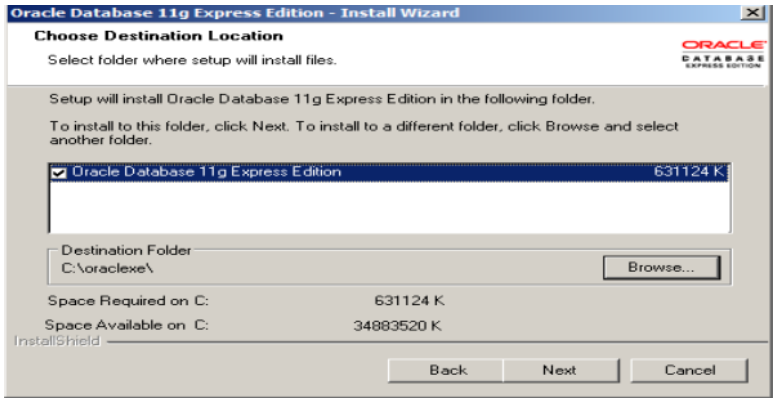


5. Una vez iniciado, presenta la pantalla de bienvenida, solo se debe presionar siguiente.

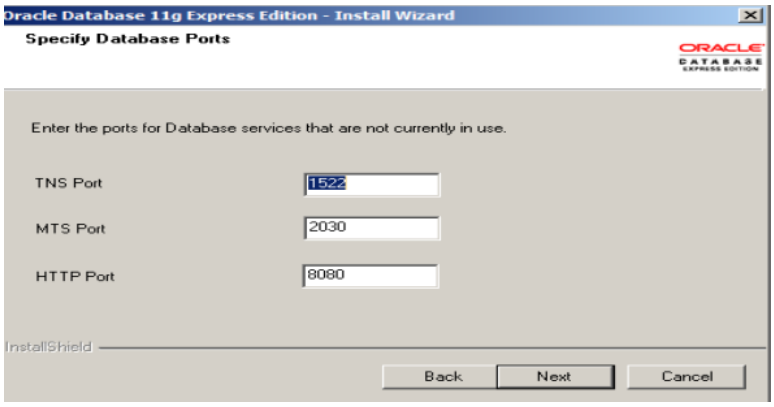


6. Nos presenta los términos y condiciones de licencia de Oracle Data Base 11g, se marca la opción aceptar condiciones y siguiente.

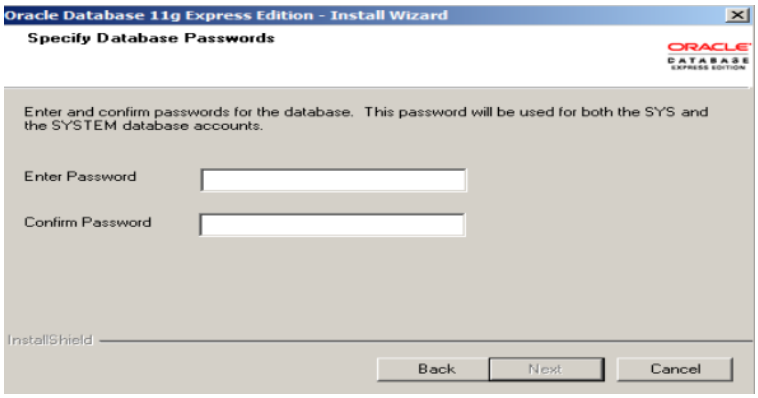
7. Se debe seleccionar la ruta donde se instalará los paquetes del motor de base de datos, lo cual mantenemos como está la información y presionamos siguiente.



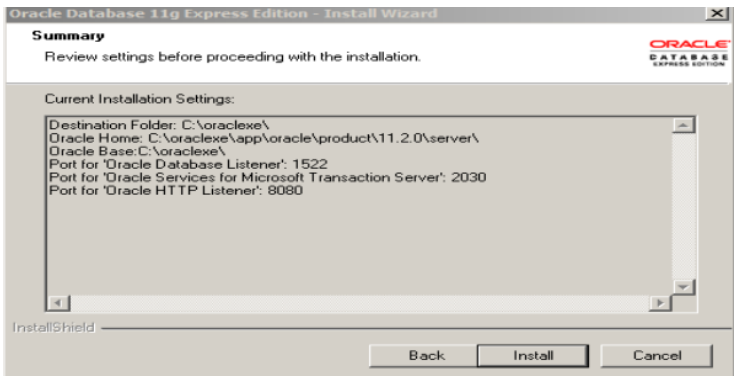
8. En caso de que se encuentre ocupados los puertos que utiliza Oracle presentara un panel para poder cambiar los puertos al momento de instalación, mantendremos los que viene por default y presionamos siguiente.



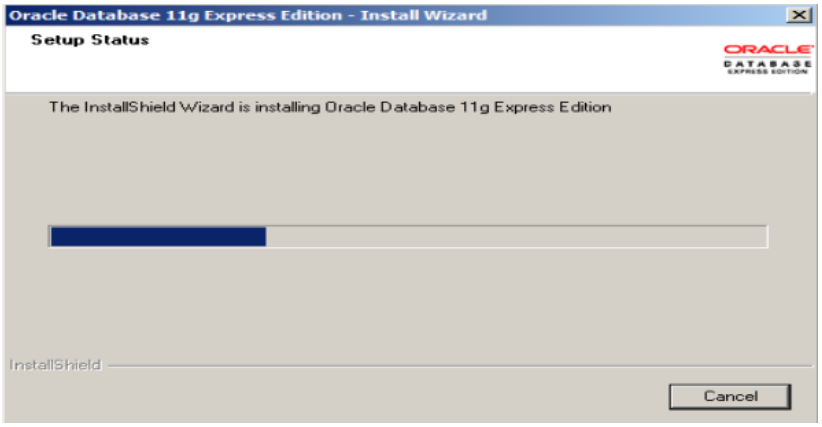
9. Solicita una clave y su confirmación para acceder con el usuario sys o system, son los super usuario y usuario con rol dba del motor de base de datos, al colocar la clave presionamos siguiente.



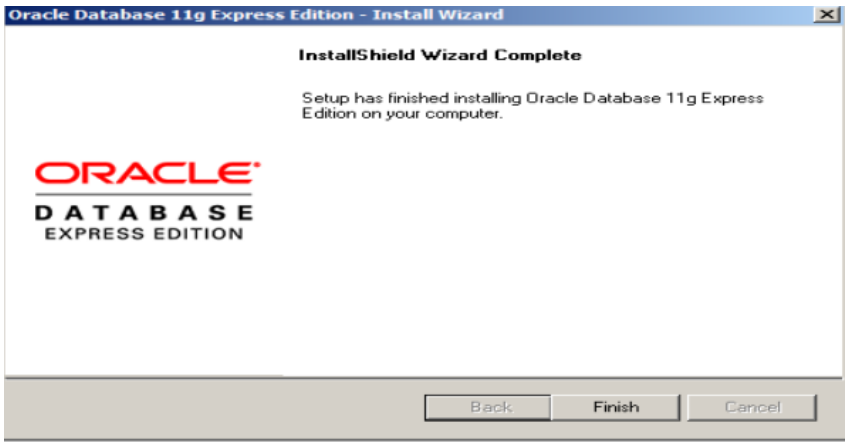
10. Presentará un resumen de los datos con rutas y puertos de lo que se instalará, presionamos instalar.



11. Se presenta el proceso de instalación y la configuración de la base de datos en el equipo.



12. Finalmente se presenta un panel informando que la instalación fue exitosa por lo que solo presionamos el botón finalizar.



Anexo I. Manual Técnico.

CARRERA ANÁLISIS DE SISTEMAS

MANUAL TÉCNICO DEL SISTEMA

AUTOR: CHRISTIAN JAVIER LOPEZ VITERI.

Quito, 2019

1.0 Capas de desarrollo.

Código de desarrollo en IDE de programación NetBeans 8.1

1.1 Paquete para el modelo de clases.

Permite agrupar todas las clases que manejan la persistencia con la base de datos, definiendo los mismos atributos que contiene una tabla.

1.2 Ejemplos de clases en el paquete de clases.

1.2.1. Clase Cotización.

```
package com.broker.modelo;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlTransient;

/**
 *
 * @author Christian Javier
 */
@Entity
@Table(name = "QLT_T_COTIZACION")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Cotizacion.findAll", query = "SELECT c FROM Cotizacion c"),
    @NamedQuery(name = "Cotizacion.findByCdCotizacion", query = "SELECT c FROM
Cotizacion c WHERE c.cdCotizacion = :cdCotizacion"),
    @NamedQuery(name = "Cotizacion.findByFcInicio", query = "SELECT c FROM
Cotizacion c WHERE c.fcInicio = :fcInicio"),
```

```
    @NamedQuery(name = "Cotizacion.findByFcFin", query = "SELECT c FROM
Cotizacion c WHERE c.fcFin = :fcFin"),
    @NamedQuery(name = "Cotizacion.findByFcRegistro", query = "SELECT c FROM
Cotizacion c WHERE c.fcRegistro = :fcRegistro"),
    @NamedQuery(name = "Cotizacion.findByNumCotizacion", query = "SELECT c FROM
Cotizacion c WHERE c.numCotizacion = :numCotizacion"),
    @NamedQuery(name = "Cotizacion.findByTipoPol", query = "SELECT c FROM
Cotizacion c WHERE c.tipoPol = :tipoPol"),
    @NamedQuery(name = "Cotizacion.findByEstado", query = "SELECT c FROM
Cotizacion c WHERE c.estado = :estado")})
public class Cotizacion implements Serializable {

    private static final long serialVersionUID = 1L;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal fields consider
using these annotations to enforce field validation
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "CD_COTIZACION")
    private Long cdCotizacion;
    @Column(name = "FC_INICIO")
    @Temporal(TemporalType.DATE)
    private Date fcInicio;
    @Column(name = "FC_FIN")
    @Temporal(TemporalType.DATE)
    private Date fcFin;
    @Column(name = "FC_REGISTRO", insertable = false)
    @Temporal(TemporalType.DATE)
    private Date fcRegistro;
    @Column(name = "NUM_COTIZACION")
    private String numCotizacion;
    @Column(name = "TIPO_POL")
    private Character tipoPol;
    @Column(name = "ESTADO")
    private Character estado;
    @OneToMany(mappedBy = "cdCotizacion", cascade = {CascadeType.ALL})
    private List<DetalleCotizacion> detalleCotizacionList;
    @OneToMany(mappedBy = "cdCotizacion", cascade = {CascadeType.ALL})
    private List<FormaPago> formaPagoList;
    @JoinColumn(name = "CD_ASEGURADORA", referencedColumnName =
"CD_ASEGURADORA")
    @ManyToOne
    private Aseguradoras cdAseguradora;
    @JoinColumn(name = "CD_CLIENTE", referencedColumnName = "CD_CLIENTE")
    @ManyToOne
    private Clientes cdCliente;

    public Cotizacion() {
        detalleCotizacionList = new ArrayList<>();
        formaPagoList = new ArrayList<>();
    }

    public Cotizacion(Long cdCotizacion) {
        this.cdCotizacion = cdCotizacion;
```

```
    }

    public Cotizacion(Long cdCotizacion, Date fcInicio, Date fcFin, Date fcRegistro, String
numCotizacion, Character tipoPol, Character estado) {
        this.cdCotizacion = cdCotizacion;
        this.fcInicio = fcInicio;
        this.fcFin = fcFin;
        this.fcRegistro = fcRegistro;
        this.numCotizacion = numCotizacion;
        this.tipoPol = tipoPol;
        this.estado = estado;
    }

    public Long getCdCotizacion() {
        return cdCotizacion;
    }

    public void setCdCotizacion(Long cdCotizacion) {
        this.cdCotizacion = cdCotizacion;
    }

    public Date getFcInicio() {
        return fcInicio;
    }

    public void setFcInicio(Date fcInicio) {
        this.fcInicio = fcInicio;
    }

    public Date getFcFin() {
        return fcFin;
    }

    public void setFcFin(Date fcFin) {
        this.fcFin = fcFin;
    }

    public Date getFcRegistro() {
        return fcRegistro;
    }

    public void setFcRegistro(Date fcRegistro) {
        this.fcRegistro = fcRegistro;
    }

    public String getNumCotizacion() {
        return numCotizacion;
    }

    public void setNumCotizacion(String numCotizacion) {
        this.numCotizacion = numCotizacion;
    }
```

```
public Character getTipoPol() {
    return tipoPol;
}

public void setTipoPol(Character tipoPol) {
    this.tipoPol = tipoPol;
}

public Character getEstado() {
    return estado;
}

public void setEstado(Character estado) {
    this.estado = estado;
}

@XmlTransient
public List<DetalleCotizacion> getDetalleCotizacionList() {
    return detalleCotizacionList;
}

public void setDetalleCotizacionList(List<DetalleCotizacion> detalleCotizacionList) {
    this.detalleCotizacionList = detalleCotizacionList;
}

@XmlTransient
public List<FormaPago> getFormaPagoList() {
    return formaPagoList;
}

public void setFormaPagoList(List<FormaPago> formaPagoList) {
    this.formaPagoList = formaPagoList;
}

public Aseguradoras getCdAseguradora() {
    return cdAseguradora;
}

public void setCdAseguradora(Aseguradoras cdAseguradora) {
    this.cdAseguradora = cdAseguradora;
}

public Clientes getCdCliente() {
    return cdCliente;
}

public void setCdCliente(Clientes cdCliente) {
    this.cdCliente = cdCliente;
}

@Override
public int hashCode() {
    int hash = 0;
```

```
        hash += (cdCotizacion != null ? cdCotizacion.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
        if (!(object instanceof Cotizacion)) {
            return false;
        }
        Cotizacion other = (Cotizacion) object;
        if ((this.cdCotizacion == null && other.cdCotizacion != null) || (this.cdCotizacion !=
null && !this.cdCotizacion.equals(other.cdCotizacion))) {
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "com.broker.modelo.Cotizacion[ cdCotizacion=" + cdCotizacion + " ]";
    }

}
```

1.2.2. Clase DetalleCotizacion

```
package com.broker.modelo;

import java.io.Serializable;
import java.math.BigDecimal;
import java.math.BigInteger;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
```

```
import javax.xml.bind.annotation.XmlTransient;

/**
 *
 * @author Christian Javier
 */
@Entity
@Table(name = "QLT_T_DETALLE_COTIZACION")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "DetalleCotizacion.findAll", query = "SELECT d FROM
DetalleCotizacion d"),
    @NamedQuery(name = "DetalleCotizacion.findByCdDetCotizacion", query = "SELECT
d FROM DetalleCotizacion d WHERE d.cdDetCotizacion = :cdDetCotizacion"),
    @NamedQuery(name = "DetalleCotizacion.findByFcInicio", query = "SELECT d FROM
DetalleCotizacion d WHERE d.fcInicio = :fcInicio"),
    @NamedQuery(name = "DetalleCotizacion.findByFcFin", query = "SELECT d FROM
DetalleCotizacion d WHERE d.fcFin = :fcFin"),
    @NamedQuery(name = "DetalleCotizacion.findByNumDias", query = "SELECT d
FROM DetalleCotizacion d WHERE d.numDias = :numDias"),
    @NamedQuery(name = "DetalleCotizacion.findByFcPoliza", query = "SELECT d FROM
DetalleCotizacion d WHERE d.fcPoliza = :fcPoliza"),
    @NamedQuery(name = "DetalleCotizacion.findByNumPoliza", query = "SELECT d
FROM DetalleCotizacion d WHERE d.numPoliza = :numPoliza"),
    @NamedQuery(name = "DetalleCotizacion.findByFactAseg", query = "SELECT d
FROM DetalleCotizacion d WHERE d.factAseg = :factAseg"),
    @NamedQuery(name = "DetalleCotizacion.findByTotAseg", query = "SELECT d FROM
DetalleCotizacion d WHERE d.totAseg = :totAseg"),
    @NamedQuery(name = "DetalleCotizacion.findByTotPrima", query = "SELECT d
FROM DetalleCotizacion d WHERE d.totPrima = :totPrima")})
public class DetalleCotizacion implements Serializable {

    @Column(name = "NUM_DIAS")
    private int numDias;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal fields consider
using these annotations to enforce field validation
    @Column(name = "TOT_ASEG")
    private Double totAseg;
    @Column(name = "TOT_PRIMA")
    private Double totPrima;
    @Column(name = "FLG_ORDEN")
    private int flgOrden;
    @OneToMany(mappedBy = "cdDetCotizacion")
    private List<ComTpFactura> comTpFacturaList;

    private static final long serialVersionUID = 1L;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal fields consider
using these annotations to enforce field validation
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "CD_DET_COTIZACION")
    private Long cdDetCotizacion;
    @Column(name = "FC_INICIO")
```

```
    @Temporal(TemporalType.DATE)
    private Date fcInicio;
    @Column(name = "FC_FIN")
    @Temporal(TemporalType.DATE)
    private Date fcFin;
    @Column(name = "FC_POLIZA")
    @Temporal(TemporalType.DATE)
    private Date fcPoliza;
    @Size(max = 50)
    @Column(name = "NUM_POLIZA")
    private String numPoliza;
    @Size(max = 15)
    @Column(name = "FACT_ASEG")
    private String factAseg;
    @Column(name = "FC_ORDEN_EMISION")
    @Temporal(TemporalType.DATE)
    private Date fcOrdenEmision;

    @Column(name = "ANEXO")
    private String anexo;

    @JoinColumn(name = "CD_COTIZACION", referencedColumnName =
"CD_COTIZACION")
    @ManyToOne
    private Cotizacion cdCotizacion;
    @JoinColumn(name = "CD_PRODUCTO", referencedColumnName =
"CD_PRODUCTO")
    @ManyToOne
    private Productos cdProducto;
    @JoinColumn(name = "CD_RAMO", referencedColumnName = "CD_RAMO")
    @ManyToOne
    private Ramos cdRamo;
    @JoinColumn(name = "CD_SUBAGENTE", referencedColumnName =
"CD_SUBAGENTE")
    @ManyToOne
    private Subagentes cdSubagente;
    @OneToMany(mappedBy = "cdDetCotizacion", cascade = {CascadeType.ALL})
    private List<Ubicacion> ubicacionList;
    @OneToMany(mappedBy = "cdDetCotizacion", cascade = {CascadeType.ALL})
    private List<ObjSeguro> objSeguroList;

    public DetalleCotizacion() {
        ubicacionList = new ArrayList<>();
        objSeguroList = new ArrayList<>();
    }

    public DetalleCotizacion(Long cdDetCotizacion) {
        this.cdDetCotizacion = cdDetCotizacion;
    }

    public DetalleCotizacion(Long cdDetCotizacion, Date fcInicio, Date fcFin, int numDias)
{
        this.cdDetCotizacion = cdDetCotizacion;
```

```
    this.fcInicio = fcInicio;
    this.fcFin = fcFin;
    this.numDias = numDias;
}

public Long getCdDetCotizacion() {
    return cdDetCotizacion;
}

public void setCdDetCotizacion(Long cdDetCotizacion) {
    this.cdDetCotizacion = cdDetCotizacion;
}

public Date getFcInicio() {
    return fcInicio;
}

public void setFcInicio(Date fcInicio) {
    this.fcInicio = fcInicio;
}

public Date getFcFin() {
    return fcFin;
}

public void setFcFin(Date fcFin) {
    this.fcFin = fcFin;
}

public int getNumDias() {
    return numDias;
}

public void setNumDias(int numDias) {
    this.numDias = numDias;
}

public Date getFcPoliza() {
    return fcPoliza;
}

public void setFcPoliza(Date fcPoliza) {
    this.fcPoliza = fcPoliza;
}

public String getNumPoliza() {
    return numPoliza;
}

public void setNumPoliza(String numPoliza) {
    this.numPoliza = numPoliza;
}
```

```
public String getFactAseg() {
    return factAseg;
}

public void setFactAseg(String factAseg) {
    this.factAseg = factAseg;
}

public Double getTotAseg() {
    return totAseg;
}

public void setTotAseg(Double totAseg) {
    this.totAseg = totAseg;
}

public Double getTotPrima() {
    return totPrima;
}

public void setTotPrima(Double totPrima) {
    this.totPrima = totPrima;
}

public Cotizacion getCdCotizacion() {
    return cdCotizacion;
}

public void setCdCotizacion(Cotizacion cdCotizacion) {
    this.cdCotizacion = cdCotizacion;
}

public Productos getCdProducto() {
    return cdProducto;
}

public void setCdProducto(Productos cdProducto) {
    this.cdProducto = cdProducto;
}

public Ramos getCdRamo() {
    return cdRamo;
}

public void setCdRamo(Ramos cdRamo) {
    this.cdRamo = cdRamo;
}

public Subagentes getCdSubagente() {
    return cdSubagente;
}

public void setCdSubagente(Subagentes cdSubagente) {
```

```
        this.cdSubagente = cdSubagente;
    }

    public Date getFcOrdenEmision() {
        return fcOrdenEmision;
    }

    public void setFcOrdenEmision(Date fcOrdenEmision) {
        this.fcOrdenEmision = fcOrdenEmision;
    }

    public int getFlgOrden() {
        return flgOrden;
    }

    public void setFlgOrden(int flgOrden) {
        this.flgOrden = flgOrden;
    }
    @XmlTransient
    public List<Ubicacion> getUbicacionList() {
        return ubicacionList;
    }

    public void setUbicacionList(List<Ubicacion> ubicacionList) {
        this.ubicacionList = ubicacionList;
    }

    @XmlTransient
    public List<ObjSeguro> getObjSeguroList() {
        return objSeguroList;
    }

    public void setObjSeguroList(List<ObjSeguro> objSeguroList) {
        this.objSeguroList = objSeguroList;
    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (cdDetCotizacion != null ? cdDetCotizacion.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
        if (!(object instanceof DetalleCotizacion)) {
            return false;
        }
        DetalleCotizacion other = (DetalleCotizacion) object;
        if ((this.cdDetCotizacion == null && other.cdDetCotizacion != null) ||
(this.cdDetCotizacion != null && !this.cdDetCotizacion.equals(other.cdDetCotizacion))) {
            return false;
```

```
        }
        return true;
    }

    @Override
    public String toString() {
        return "com.broker.modelo.DetalleCotizacion[ cdDetCotizacion=" + cdDetCotizacion +
" ]";
    }

    @XmlTransient
    public List<ComTpFactura> getComTpFacturaList() {
        return comTpFacturaList;
    }

    public void setComTpFacturaList(List<ComTpFactura> comTpFacturaList) {
        this.comTpFacturaList = comTpFacturaList;
    }

    public String getAnexo() {
        return anexo;
    }

    public void setAnexo(String anexo) {
        this.anexo = anexo;
    }
}
```

1.2.3. Clase Financiamiento

```
package com.broker.modelo;

import java.io.Serializable;
import java.math.BigInteger;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
```

```
import javax.xml.bind.annotation.XmlTransient;

/**
 *
 * @author Christian Javier
 */
@Entity
@Table(name = "QLT_T_FINANCIAMIENTO")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Financiamiento.findAll", query = "SELECT f FROM
Financiamiento f"),
    @NamedQuery(name = "Financiamiento.findByCdFinanciamiento", query = "SELECT f
FROM Financiamiento f WHERE f.cdFinanciamiento = :cdFinanciamiento"),
    @NamedQuery(name = "Financiamiento.findByOrnidal", query = "SELECT f FROM
Financiamiento f WHERE f.ornidal = :ornidal"),
    @NamedQuery(name = "Financiamiento.findByFcVencimiento", query = "SELECT f
FROM Financiamiento f WHERE f.fcVencimiento = :fcVencimiento"),
    @NamedQuery(name = "Financiamiento.findByValor", query = "SELECT f FROM
Financiamiento f WHERE f.valor = :valor"),
    @NamedQuery(name = "Financiamiento.findByAbono", query = "SELECT f FROM
Financiamiento f WHERE f.abono = :abono"),
    @NamedQuery(name = "Financiamiento.findBySaldo", query = "SELECT f FROM
Financiamiento f WHERE f.saldo = :saldo"),
    @NamedQuery(name = "Financiamiento.findBySaldoPago", query = "SELECT f FROM
Financiamiento f WHERE f.saldoPago = :saldoPago"),
    @NamedQuery(name = "Financiamiento.findByFcPago", query = "SELECT f FROM
Financiamiento f WHERE f.fcPago = :fcPago"),
    @NamedQuery(name = "Financiamiento.findByObservacion", query = "SELECT f
FROM Financiamiento f WHERE f.observacion = :observacion"),
    @NamedQuery(name = "Financiamiento.findByFactAseg", query = "SELECT f FROM
Financiamiento f WHERE f.factAseg = :factAseg"),
    @NamedQuery(name = "Financiamiento.findByLetras", query = "SELECT f FROM
Financiamiento f WHERE f.letras = :letras")})
public class Financiamiento implements Serializable {

    private static final long serialVersionUID = 1L;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal fields consider
using these annotations to enforce field validation
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "CD_FINANCIAMIENTO")
    private Long cdFinanciamiento;
    @Column(name = "ORNIDAL")
    private int ornidal;
    @Column(name = "FC_VENCIMIENTO")
    @Temporal(TemporalType.DATE)
    private Date fcVencimiento;
    @Column(name = "VALOR")
    private Double valor;
    @Column(name = "ABONO")
    private Double abono;
    @Column(name = "SALDO")
```

```
    private Double saldo;
    @Column(name = "SALDO_PAGO")
    private Double saldoPago;
    @Column(name = "FC_PAGO")
    @Temporal(TemporalType.DATE)
    private Date fcPago;
    @Size(max = 250)
    @Column(name = "OBSERVACION")
    private String observacion;
    @Size(max = 15)
    @Column(name = "FACT_ASEG")
    private String factAseg;
    @Size(max = 25)
    @Column(name = "LETRAS")
    private String letras;
    @Column(name = "FLG_PAGO")
    private Character flgPago;
    @JoinColumn(name = "CD_FORMA_PAGO", referencedColumnName =
"CD_FORMA_PAGO")
    @ManyToOne
    private FormaPago cdFormaPago;
    @OneToMany(mappedBy = "cdFinanciamiento", cascade = {CascadeType.ALL})
    private List<ComTpFactura> comTpFacturaList;

    public Financiamiento() {
        comTpFacturaList = new ArrayList<>();
    }

    public Financiamiento(Long cdFinanciamiento) {
        this.cdFinanciamiento = cdFinanciamiento;
    }

    public Long getCdFinanciamiento() {
        return cdFinanciamiento;
    }

    public void setCdFinanciamiento(Long cdFinanciamiento) {
        this.cdFinanciamiento = cdFinanciamiento;
    }

    public int getOrnidal() {
        return ornidal;
    }

    public void setOrnidal(int ornidal) {
        this.ornidal = ornidal;
    }

    public Date getFcVencimiento() {
        return fcVencimiento;
    }

    public void setFcVencimiento(Date fcVencimiento) {
```

```
    this.fcVencimiento = fcVencimiento;
}

public Double getValor() {
    return valor;
}

public void setValor(Double valor) {
    this.valor = valor;
}

public Double getAbono() {
    return abono;
}

public void setAbono(Double abono) {
    this.abono = abono;
}

public Double getSaldo() {
    return saldo;
}

public void setSaldo(Double saldo) {
    this.saldo = saldo;
}

public Double getSaldoPago() {
    return saldoPago;
}

public void setSaldoPago(Double saldoPago) {
    this.saldoPago = saldoPago;
}

public Date getFcPago() {
    return fcPago;
}

public void setFcPago(Date fcPago) {
    this.fcPago = fcPago;
}

public String getObservacion() {
    return observacion;
}

public void setObservacion(String observacion) {
    this.observacion = observacion;
}

public String getFactAseg() {
    return factAseg;
```

```
    }

    public void setFactAseg(String factAseg) {
        this.factAseg = factAseg;
    }

    public String getLetras() {
        return letras;
    }

    public void setLetras(String letras) {
        this.letras = letras;
    }

    public Character getFlgPago() {
        return flgPago;
    }

    public void setFlgPago(Character flgPago) {
        this.flgPago = flgPago;
    }



    public FormaPago getCdFormaPago() {
        return cdFormaPago;
    }

    public void setCdFormaPago(FormaPago cdFormaPago) {
        this.cdFormaPago = cdFormaPago;
    }

    @XmlTransient
    public List<ComTpFactura> getComTpFacturaList() {
        return comTpFacturaList;
    }

    public void setComTpFacturaList(List<ComTpFactura> comTpFacturaList) {
        this.comTpFacturaList = comTpFacturaList;
    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (cdFinanciamiento != null ? cdFinanciamiento.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
        if (!(object instanceof Financiamiento)) {
            return false;
```

```
        }
        Financiamiento other = (Financiamiento) object;
        if ((this.cdFinanciamiento == null && other.cdFinanciamiento != null) ||
(this.cdFinanciamiento != null && !this.cdFinanciamiento.equals(other.cdFinanciamiento)))
{
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "com.broker.modelo.Financiamiento[ cdFinanciamiento=" + cdFinanciamiento +
" ]";
    }

}
```

1.2.4. Clase factura

```
package com.broker.modelo;

import java.io.Serializable;
import java.util.Date;
import java.util.List;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.XmlTransient;

/**
 *
 * @author Christian Javier
 */
@Entity
@Table(name = "QLT_T_FACTURAS")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Facturas.findAll", query = "SELECT f FROM Facturas f"),
    @NamedQuery(name = "Facturas.findByCdFacturas", query = "SELECT f FROM
Facturas f WHERE f.cdFacturas = :cdFacturas"),
```

```
    @NamedQuery(name = "Facturas.findByNumFactura", query = "SELECT f FROM
Facturas f WHERE f.numFactura = :numFactura"),
    @NamedQuery(name = "Facturas.findByFcFactura", query = "SELECT f FROM
Facturas f WHERE f.fcFactura = :fcFactura"),
    @NamedQuery(name = "Facturas.findByValFactura", query = "SELECT f FROM
Facturas f WHERE f.valFactura = :valFactura"),
    @NamedQuery(name = "Facturas.findByIva", query = "SELECT f FROM Facturas f
WHERE f.iva = :iva"),
    @NamedQuery(name = "Facturas.findByTotFactura", query = "SELECT f FROM
Facturas f WHERE f.totFactura = :totFactura"),
    @NamedQuery(name = "Facturas.findByFlgPago", query = "SELECT f FROM Facturas f
WHERE f.flgPago = :flgPago"),
    @NamedQuery(name = "Facturas.findByFcPago", query = "SELECT f FROM Facturas f
WHERE f.fcPago = :fcPago"),
    @NamedQuery(name = "Facturas.findByFcAnulacion", query = "SELECT f FROM
Facturas f WHERE f.fcAnulacion = :fcAnulacion"),
    @NamedQuery(name = "Facturas.findByEstado", query = "SELECT f FROM Facturas f
WHERE f.estado = :estado")})
public class Facturas implements Serializable {

    private static final long serialVersionUID = 1L;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal fields consider
using these annotations to enforce field validation
    @Id
    @Basic(optional = false)
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "CD_FACTURAS")
    private Long cdFacturas;
    @Size(max = 15)
    @Column(name = "NUM_FACTURA")
    private String numFactura;
    @Column(name = "FC_FACTURA")
    @Temporal(TemporalType.TIMESTAMP)
    private Date fcFactura;
    @Column(name = "VAL_FACTURA")
    private Long valFactura;
    @Column(name = "IVA")
    private Long iva;
    @Column(name = "TOT_FACTURA")
    private Long totFactura;
    @Column(name = "FLG_PAGO")
    private Character flgPago;
    @Column(name = "FC_PAGO")
    @Temporal(TemporalType.TIMESTAMP)
    private Date fcPago;
    @Column(name = "FC_ANULACION")
    @Temporal(TemporalType.TIMESTAMP)
    private Date fcAnulacion;
    @Column(name = "ESTADO")
    private Character estado;
    @OneToMany(mappedBy = "cdFacturas")
    private List<DetalleFactura> detalleFacturaList;
```

```
public Facturas() {
}

public Facturas(Long cdFacturas) {
   this.cdFacturas = cdFacturas;
}

public Long getCdFacturas() {
   return cdFacturas;
}

public void setCdFacturas(Long cdFacturas) {
   this.cdFacturas = cdFacturas;
}

public String getNumFactura() {
   return numFactura;
}

public void setNumFactura(String numFactura) {
   this.numFactura = numFactura;
}

public Date getFcFactura() {
   return fcFactura;
}

public void setFcFactura(Date fcFactura) {
   this.fcFactura = fcFactura;
}

public Long getValFactura() {
   return valFactura;
}

public void setValFactura(Long valFactura) {
   this.valFactura = valFactura;
}

public Long getIva() {
   return iva;
}

public void setIva(Long iva) {
   this.iva = iva;
}

public Long getTotFactura() {
   return totFactura;
}

public void setTotFactura(Long totFactura) {
   this.totFactura = totFactura;
```

```
    }

    public Character getFlgPago() {
        return flgPago;
    }

    public void setFlgPago(Character flgPago) {
        this.flgPago = flgPago;
    }

    public Date getFcPago() {
        return fcPago;
    }

    public void setFcPago(Date fcPago) {
        this.fcPago = fcPago;
    }

    public Date getFcAnulacion() {
        return fcAnulacion;
    }

    public void setFcAnulacion(Date fcAnulacion) {
        this.fcAnulacion = fcAnulacion;
    }

    public Character getEstado() {
        return estado;
    }

    public void setEstado(Character estado) {
        this.estado = estado;
    }

    @XmlTransient
    public List<DetalleFactura> getDetalleFacturaList() {
        return detalleFacturaList;
    }

    public void setDetalleFacturaList(List<DetalleFactura> detalleFacturaList) {
        this.detalleFacturaList = detalleFacturaList;
    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (cdFacturas != null ? cdFacturas.hashCode() : 0);
        return hash;
    }

    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
```

```
        if (!(object instanceof Facturas)) {
            return false;
        }
        Facturas other = (Facturas) object;
        if ((this.cdFacturas == null && other.cdFacturas != null) || (this.cdFacturas != null &&
!this.cdFacturas.equals(other.cdFacturas))) {
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "com.broker.modelo.Facturas[ cdFacturas=" + cdFacturas + " ]";
    }

}
```

1.3 Paquete para clases Bean

Paquete que permite almacenar las clases que permite el control de las interfaces y contienen la lógica del negocio.

1.3.1. Clase BeanCotizacion.

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.broker.bean;

import com.broker.dao.AseguradorasDAO;
import com.broker.dao.ColoresVhDAO;
import com.broker.dao.ComTpFacturaDAO;
import com.broker.dao.CotizacionDAO;
import com.broker.dao.DerechosEmisionDAO;
import com.broker.dao.DetalleCotizacionDAO;
import com.broker.dao.MarcasDAO;
import com.broker.dao.ProductosDAO;
import com.broker.dao.RamoDAO;
import com.broker.modelo.Aseguradoras;
import com.broker.modelo.Clientes;
import com.broker.modelo.ColoresVh;
import com.broker.modelo.ComTpFactura;
import com.broker.modelo.Cotizacion;
import com.broker.modelo.DerechosEmision;
import com.broker.modelo.DetalleCotizacion;
import com.broker.modelo.Financiamiento;
import com.broker.modelo.FormaPago;
import com.broker.modelo.Marcas;
import com.broker.modelo.ObjSeguro;
import com.broker.modelo.Productos;
import com.broker.modelo.Ramos;
import com.broker.modelo.Subagentes;
import com.broker.modelo.Subobjetos;
import com.broker.modelo.Ubicacion;
import com.broker.util.Funciones;
import com.broker.util.Mensajes;
import java.io.Serializable;
```

```
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
import javax.faces.context.FacesContext;
import org.primefaces.context.RequestContext;

/**
 *
 * @author Christian Javier
 */
@ManagedBean
@ViewScoped
public class BeanCotizacion implements Bean, Serializable {

    private Clientes clientes;

    @EJB(beanName = "CotizacionDAOImpl")
    private CotizacionDAO cotizacionDAO;
    private Cotizacion cotizacion;

    private FormaPago formaPago;
    private Double iva;

    private Financiamiento financiamiento;

    @EJB(beanName = "AseguradorasDAOImpl")
    private AseguradorasDAO aseguradorasDAO;
    private Aseguradoras aseguradoras;
    private List<Aseguradoras> listaAseguradoras;
    private Long cdAseguradora;

    @EJB(beanName = "DetalleCotizacionDAOImpl")
    private DetalleCotizacionDAO detalleCotizacionDAO;
    private DetalleCotizacion detalleCotizacion;
    private DetalleCotizacion detalleCotizacionSel;
    private Long cdDetCotizacion;

    private Subagentes subagentes;

    @EJB(beanName = "RamoDAOImpl")
    private RamoDAO ramoDAO;
    private Ramos ramos;
    private List<Ramos> listaramos;
    private Long cdRamo;

    @EJB(beanName = "ProductosDAOImpl")
    private ProductosDAO productosDAO;
    private Productos productos;
    private List<Productos> listaProductos;
    private Long cdProducto;

    private Ubicacion ubicacion;
    private Ubicacion ubicacionSel;
```

```
    private ObjSeguro objSeguro;
    private ObjSeguro objSeguroSel;
    private ObjSeguro vehiculo;

    private Subobjetos subobjetos;

    @EJB(beanName = "MarcasDAOImpl")
    private MarcasDAO marcasDAO;
    private List<Marcas> listaMarcas;
    private Marcas marcas;

    @EJB(beanName = "ColoresVhDAOImpl")
    private ColoresVhDAO coloresVhDAO;
    private List<ColoresVh> listaColores;

    @EJB(beanName = "DerechosEmisionDAOImpl")
    private DerechosEmisionDAO derechosEmisionDAO;
    private DerechosEmision derechosEmision;

    private ComTpFactura comTpFactura;

    private Double TotalAseg;
    private Double TotalPrima;

    @PostConstruct
    @Override
    public void init() {
       try {
          cancel();
          findAll();
          cotizacion = (Cotizacion)
FacesContext.getCurrentInstance().getExternalContext().getSessionMap().get("cotizacion");
          clientes = cotizacion.getCdCliente();
          cdAseguradora = cotizacion.getCdAseguradora().getCdAseguradora();
          if (!clientes.getSubagentesList().isEmpty()) {
             subagentes = clientes.getSubagentesList().get(0);
          }

          if(!cotizacion.getFormaPagoList().isEmpty()){
             formaPago = cotizacion.getFormaPagoList().get(0);
          }
       } catch (Exception e) {
       }
    }

    @Override
    public void create() {

    }

    @Override
    public void delete() {

    }

    public void deleteDetalleCotizacion(DetalleCotizacion detalle) {
       try {
          cotizacion.getDetalleCotizacionList().remove(detalle);
          detalleCotizacionDAO.delete(detalle);
```

```
            cotizacion = cotizacionDAO.findById(cotizacion.getCdCotizacion());
        } catch (Exception e) {
            Mensajes.mensajeError(e.getCause().getCause().getCause().getMessage());
        }
    }

    @Override
    public void update() {
        try {
            cotizacionDAO.update(cotizacion);
            Mensajes.mensajeGrabarCorrecto();
            cotizacion = cotizacionDAO.findById(cotizacion.getCdCotizacion());
        } catch (Exception e) {
            Mensajes.mensajeError(e.getMessage());
        }

    }

    @Override
    public void findAll() {
        listaAseguradoras = aseguradorasDAO.findByFk("WHERE t.estado = 'A'");
        listaramos = ramoDAO.findByFk("WHERE t.estado = 'A'");
    }

    @Override
    public void find() {

    }

    @Override
    public void cancel() {
        cdAseguradora = 0L;
        cdProducto = 0L;
        cdRamo = 0L;
        clientes = new Clientes();
        cotizacion = new Cotizacion();
        detalleCotizacion = new DetalleCotizacion();
        financiamiento = new Financiamiento();
        formaPago = new FormaPago();
        objSeguro = new ObjSeguro();
        productos = new Productos();
        ramos = new Ramos();
        subagentes = new Subagentes();
        subobjetos = new Subobjetos();
        ubicacion = new Ubicacion();
        vehiculo = new ObjSeguro();
    }

    @Override
    public Boolean validate() {
        return true;
    }

    public void selectRamo() {
        listaProductos = productosDAO.findByFk("where t.cdRamo.cdRamo = " + cdRamo +
"""
                + " AND t.cdAseguradora.cdAseguradora = " + cdAseguradora);
    }
```

```
    public void selectMarca() {
        marcas = marcasDAO.findByFk("WHERE t.desMarca = '" + objSeguro.getMarca() +
"'").get(0);
    }

    public void calcPrima() {
        try {
            objSeguro.setPrima(Funciones.calculaPrima(detalleCotizacionSel.getNumDias(),
objSeguro.getValAseg(), objSeguro.getFactor(), objSeguro.getTasa()));
        } catch (Exception e) {

        }
    }

    public int diasEntreFechas(Date fecha1, Date fecha2) {
        long startTime = fecha1.getTime();
        long endTime = fecha2.getTime();
        long diffTime = endTime - startTime;
        long diffDays = diffTime / (1000 * 60 * 60 * 24);
        return (int) diffDays;
    }

    public void finVigenciaCot() {
        cotizacion.setFcFin((new Date(cotizacion.getFcInicio().getYear(),
cotizacion.getFcInicio().getMonth() + 1, cotizacion.getFcInicio().getDate())));
    }

    public void finVigencia() {
        detalleCotizacion.setFcFin((new Date(detalleCotizacion.getFcInicio().getYear() + 1,
detalleCotizacion.getFcInicio().getMonth(), detalleCotizacion.getFcInicio().getDate())));
    }

    public void addDetalleCotizacion() {
        try {
            detalleCotizacion.setCdRamo(ramoDAO.findById(cdRamo));
            detalleCotizacion.setCdSubagente(subagentes);
            detalleCotizacion.setCdProducto(productosDAO.findById(cdProducto));
            detalleCotizacion.setTotAseg(0D);
            detalleCotizacion.setTotPrima(0D);
            detalleCotizacion.setCdCotizacion(cotizacion);
            detalleCotizacion.setNumDias(diasEntreFechas(detalleCotizacion.getFcInicio(),
detalleCotizacion.getFcFin()));
            cotizacion.getDetalleCotizacionList().add(detalleCotizacion);
            detalleCotizacion = new DetalleCotizacion();
        } catch (Exception e) {
            Mensajes.mensajeError(e.getCause().getCause().getMessage());
        }
    }

    public void consObjSeguro() {
        try {

            if (!detalleCotizacionSel.getCdRamo().getDesRamo().equals("VEHICULOS")) {
                objSeguro.setFactor(100);
                RequestContext.getCurrentInstance().execute("PF('dgObjSeguro').show();");
            } else {
                objSeguro.setFactor(100);
                listaMarcas = marcasDAO.findByFk("WHERE t.estado = 'A' order by
t.desMarca");
```

```
            listaColores = coloresVhDAO.findByFk("WHERE t.estado = 'A'");
            RequestContext.getCurrentInstance().execute("PF('dgObjSeguro').show();");
        }
    } catch (Exception e) {
        System.err.println(e.getMessage());
    }
}

public void addUbicacion() {
    try {
        ubicacion.setCdDetCotizacion(detalleCotizacionSel);
        detalleCotizacionSel.getUbicacionList().add(ubicacion);
        ubicacion = new Ubicacion();
    } catch (Exception e) {
    }
}

public void addObjSeguro() {
    try {

        if (ubicacionSel == null) {
            objSeguro.setFcFin(detalleCotizacionSel.getFcFin());
            objSeguro.setFcInicio(detalleCotizacionSel.getFcInicio());
            objSeguro.setTotAseg(objSeguro.getValAseg());
            objSeguro.setTotPrima(objSeguro.getPrima());
            objSeguro.setCdDetCotizacion(detalleCotizacionSel);
            detalleCotizacionSel.getObjSeguroList().add(objSeguro);
        } else {
            objSeguro.setFcFin(ubicacionSel.getFcFin());
            objSeguro.setFcInicio(ubicacionSel.getFcInicio());
            objSeguro.setTotAseg(objSeguro.getValAseg());
            objSeguro.setTotPrima(objSeguro.getPrima());
            objSeguro.setCdUbicacion(ubicacionSel);
            ubicacionSel.getObjSeguroList().add(objSeguro);
        }

        objSeguro = new ObjSeguro();
    } catch (Exception e) {
    }
}

public void addSubobjeto() {
    try {
        subobjetos.setCdObjSeguro(objSeguroSel);
        objSeguroSel.getSubobjetosList().add(subobjetos);
        subobjetos = new Subobjetos();
    } catch (Exception e) {
    }
}

public void carVehiculos(ObjSeguro seguro) {
    vehiculo = seguro;
    RequestContext.getCurrentInstance().execute("PF('dgCarVh').show();");
}

public void totAseguradoUbicacion() {
    try {
        detalleCotizacionSel.setTotAseg(0D);
        detalleCotizacionSel.setTotPrima(0D);
```

```
            for (Ubicacion ubi : detalleCotizacionSel.getUbicacionList()) {
                detalleCotizacionSel.setTotAseg(detalleCotizacionSel.getTotAseg() +
ubi.getTotAsegurado());
                detalleCotizacionSel.setTotPrima(detalleCotizacionSel.getTotPrima() +
ubi.getTotPrima());
            }
        } catch (Exception e) {
        }
    }

    public void totAseguradoObj() {
        try {

            if (detalleCotizacionSel.getCdRamo().getDesRamo().equals("VEHICULOS")) {
                detalleCotizacionSel.setTotAseg(0D);
                detalleCotizacionSel.setTotPrima(0D);
                for (ObjSeguro obj : detalleCotizacionSel.getObjSeguroList()) {
                    detalleCotizacionSel.setTotAseg(detalleCotizacionSel.getTotAseg() +
obj.getTotAseg());
                    detalleCotizacionSel.setTotPrima(detalleCotizacionSel.getTotPrima() +
obj.getTotPrima());
                }
            } else {
                ubicacionSel.setTotAsegurado(0D);
                ubicacionSel.setTotPrima(0D);
                for (ObjSeguro obj : ubicacionSel.getObjSeguroList()) {
                    ubicacionSel.setTotAsegurado(ubicacionSel.getTotAsegurado() +
obj.getTotAseg());
                    ubicacionSel.setTotPrima(ubicacionSel.getTotPrima() + obj.getTotPrima());
                }
            }
        } catch (Exception e) {
        }
    }

    public void totAsegSubobjeto() {
        try {
            objSeguroSel.setTotAseg(objSeguroSel.getValAseg());
            objSeguroSel.setTotPrima(objSeguroSel.getPrima());
            for (Subobjetos subobjeto : objSeguroSel.getSubobjetosList()) {
                objSeguroSel.setTotAseg(objSeguroSel.getTotAseg() + subobjeto.getValAseg());
                objSeguroSel.setTotPrima(objSeguroSel.getTotPrima() + subobjeto.getPrima());
            }
        } catch (Exception e) {
        }
    }

    public void addFormaPago() {
        try {
            formaPago.setTotPrima(cotizacion.getDetalleCotizacionList().get(0).getTotPrima());
            derechosEmision = derechosEmisionDAO.findByFk("WHERE t.primaIni <= " +
formaPago.getTotPrima() + " "
                    + "AND t.primaFin >= " + formaPago.getTotPrima()).get(0);
            formaPago.setDerEmision(derechosEmision.getValor());
            formaPago.setSupBancos(formaPago.getTotPrima() * 0.035);
            formaPago.setSegCampecino(formaPago.getTotPrima() * 0.005);
            formaPago.setSubtotal(formaPago.getTotPrima() + formaPago.getDerEmision() +
formaPago.getSupBancos() + formaPago.getSegCampecino());
            formaPago.setPctIva(12);
```

```
            formaPago.setValIva((formaPago.getSubtotal()) * (formaPago.getPctIva()/100D ));
            formaPago.setTotPago(formaPago.getSubtotal() + formaPago.getValIva());
            RequestContext.getCurrentInstance().execute("PF('dgFormaPago').show();");
        } catch (Exception e) {
        }
    }

    public void tipoPago(){
        try {
            if(formaPago.getFormaPago().equals("CONTADO")){
                formaPago.setNumPagos(1);
            }else{
                formaPago.setNumPagos(6);
            }
        } catch (Exception e) {
        }
    }

    public void procesarFinanciamiento(){
        try {

            Double valCuota = formaPago.getTotPago() / formaPago.getNumPagos();
            valCuota =  Math.rint(valCuota*100)/100;
            Double valTotal = formaPago.getTotPago();
            formaPago.setCdCotizacion(cotizacion);
            cotizacion.setFormaPagoList(new ArrayList<>());
            cotizacion.getFormaPagoList().add(formaPago);
            for (int i = 0; i < formaPago.getNumPagos(); i++) {
                financiamiento.setCdFormaPago(formaPago);
                financiamiento.setOrnidal(i + 1);
                financiamiento.setFcVencimiento(Funciones.sumarMeses(new Date(), i));
                if((i+1) == formaPago.getNumPagos()){
                    financiamiento.setValor(valTotal);
                    financiamiento.setSaldo(valTotal);
                }else{
                    financiamiento.setValor(valCuota );
                    financiamiento.setSaldo(valCuota);
                    valTotal = valTotal - valCuota;
                }
                financiamiento.setAbono(0D);
                financiamiento.setSaldoPago(0D);
                financiamiento.setLetras("CUOTA "+(i+1)+"/"+formaPago.getNumPagos());
                formaPago.getFinanciamientoList().add(financiamiento);
                financiamiento = new Financiamiento();
            }
            RequestContext.getCurrentInstance().execute("PF('dgFormaPago').hide()");
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }

    public void fcVencimiento() {
        int cont = 0;
        Date fecha = formaPago.getFinanciamientoList().get(0).getFcVencimiento();
        for( Financiamiento f: formaPago.getFinanciamientoList() ){
            f.setFcVencimiento(Funciones.sumarMeses(fecha, cont));
            cont++;
        }
    }
```

```
public Clientes getClientes() {
    return clientes;
}

public void setClientes(Clientes clientes) {
    this.clientes = clientes;
}

public Cotizacion getCotizacion() {
    return cotizacion;
}

public void setCotizacion(Cotizacion cotizacion) {
    this.cotizacion = cotizacion;
}

public FormaPago getFormaPago() {
    return formaPago;
}

public void setFormaPago(FormaPago formaPago) {
    this.formaPago = formaPago;
}

public Financiamiento getFinanciamiento() {
    return financiamiento;
}

public void setFinanciamiento(Financiamiento financiamiento) {
    this.financiamiento = financiamiento;
}

public Aseguradoras getAseguradoras() {
    return aseguradoras;
}

public void setAseguradoras(Aseguradoras aseguradoras) {
    this.aseguradoras = aseguradoras;
}

public List<Aseguradoras> getListaAseguradoras() {
    return listaAseguradoras;
}

public void setListaAseguradoras(List<Aseguradoras> listaAseguradoras) {
    this.listaAseguradoras = listaAseguradoras;
}

public Long getCdAseguradora() {
    return cdAseguradora;
}

public void setCdAseguradora(Long cdAseguradora) {
    this.cdAseguradora = cdAseguradora;
}

public DetalleCotizacion getDetalleCotizacion() {
    return detalleCotizacion;
```

```
}

public void setDetalleCotizacion(DetalleCotizacion detalleCotizacion) {
   this.detalleCotizacion = detalleCotizacion;
}

public Subagentes getSubagentes() {
   return subagentes;
}

public void setSubagentes(Subagentes subagentes) {
   this.subagentes = subagentes;
}

public Ramos getRamos() {
   return ramos;
}

public void setRamos(Ramos ramos) {
   this.ramos = ramos;
}

public List<Ramos> getListaramos() {
   return listaramos;
}

public void setListaramos(List<Ramos> listaramos) {
   this.listaramos = listaramos;
}

public Long getCdRamo() {
   return cdRamo;
}

public void setCdRamo(Long cdRamo) {
   this.cdRamo = cdRamo;
}

public Productos getProductos() {
   return productos;
}

public void setProductos(Productos productos) {
   this.productos = productos;
}

public List<Productos> getListaProductos() {
   return listaProductos;
}

public void setListaProductos(List<Productos> listaProductos) {
   this.listaProductos = listaProductos;
}

public Long getCdProducto() {
   return cdProducto;
}

public void setCdProducto(Long cdProducto) {
```

```
    this.cdProducto = cdProducto;
}

public Ubicacion getUbicacion() {
    return ubicacion;
}

public void setUbicacion(Ubicacion ubicacion) {
    this.ubicacion = ubicacion;
}

public ObjSeguro getObjSeguro() {
    return objSeguro;
}

public void setObjSeguro(ObjSeguro objSeguro) {
    this.objSeguro = objSeguro;
}

public Subobjetos getSubobjetos() {
    return subobjetos;
}

public void setSubobjetos(Subobjetos subobjetos) {
    this.subobjetos = subobjetos;
}

public DetalleCotizacion getDetalleCotizacionSel() {
    return detalleCotizacionSel;
}

public void setDetalleCotizacionSel(DetalleCotizacion detalleCotizacionSel) {
    this.detalleCotizacionSel = detalleCotizacionSel;
}

public ObjSeguro getObjSeguroSel() {
    return objSeguroSel;
}

public void setObjSeguroSel(ObjSeguro objSeguroSel) {
    this.objSeguroSel = objSeguroSel;
}

public List<Marcas> getListaMarcas() {
    return listaMarcas;
}

public void setListaMarcas(List<Marcas> listaMarcas) {
    this.listaMarcas = listaMarcas;
}

public Marcas getMarcas() {
    return marcas;
}

public void setMarcas(Marcas marcas) {
    this.marcas = marcas;
}
```

```
    public List<ColoresVh> getListaColores() {
        return listaColores;
    }

    public void setListaColores(List<ColoresVh> listaColores) {
        this.listaColores = listaColores;
    }

    public ObjSeguro getVehiculo() {
        return vehiculo;
    }

    public void setVehiculo(ObjSeguro vehiculo) {
        this.vehiculo = vehiculo;
    }

    public Double getIva() {
        return iva;
    }

    public void setIva(Double iva) {
        this.iva = iva;
    }

    public Double getTotalAseg() {
        return TotalAseg;
    }

    public void setTotalAseg(Double TotalAseg) {
        this.TotalAseg = TotalAseg;
    }

    public Double getTotalPrima() {
        return TotalPrima;
    }

    public void setTotalPrima(Double TotalPrima) {
        this.TotalPrima = TotalPrima;
    }

}
```

1.3.2. Clase BeanEmision.

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.broker.bean;

import com.broker.dao.AseguradorasDAO;
import com.broker.dao.ColoresVhDAO;
import com.broker.dao.ComisionBrokerDAO;
import com.broker.dao.ComisionesSubagentesDAO;
import com.broker.dao.CotizacionDAO;
import com.broker.dao.DerechosEmisionDAO;
import com.broker.dao.DetalleCotizacionDAO;
```

```
import com.broker.dao.MarcasDAO;
import com.broker.dao.ProductosDAO;
import com.broker.dao.RamoDAO;
import com.broker.modelo.Aseguradoras;
import com.broker.modelo.Clientes;
import com.broker.modelo.ColoresVh;
import com.broker.modelo.ComTpFactura;
import com.broker.modelo.ComisionBroker;
import com.broker.modelo.ComisionSubagentes;
import com.broker.modelo.Cotizacion;
import com.broker.modelo.DerechosEmision;
import com.broker.modelo.DetalleCotizacion;
import com.broker.modelo.Financiamiento;
import com.broker.modelo.FormaPago;
import com.broker.modelo.Marcas;
import com.broker.modelo.ObjSeguro;
import com.broker.modelo.Productos;
import com.broker.modelo.Ramos;
import com.broker.modelo.Subagentes;
import com.broker.modelo.Subobjetos;
import com.broker.modelo.Ubicacion;
import com.broker.util.Funciones;
import com.broker.util.Mensajes;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
import javax.faces.context.FacesContext;
import org.primefaces.context.RequestContext;

/**
 *
 * @author Christian Javier
 */
@ManagedBean
@ViewScoped
public class BeanEmision implements Bean, Serializable {

    private Clientes clientes;

    @EJB(beanName = "CotizacionDAOImpl")
    private CotizacionDAO cotizacionDAO;
    private Cotizacion cotizacion;

    private FormaPago formaPago;
    private Double iva;

    private Financiamiento financiamiento;
```

```
@EJB(beanName = "AseguradorasDAOImpl")
private AseguradorasDAO aseguradorasDAO;
private Aseguradoras aseguradoras;
private List<Aseguradoras> listaAseguradoras;
private Long cdAseguradora;

@EJB(beanName = "DetalleCotizacionDAOImpl")
private DetalleCotizacionDAO detalleCotizacionDAO;
private DetalleCotizacion detalleCotizacion;
private DetalleCotizacion detalleCotizacionSel;
private DetalleCotizacion detCot;
private Long cdDetCotizacion;

private Subagentes subagentes;

@EJB(beanName = "RamoDAOImpl")
private RamoDAO ramoDAO;
private Ramos ramos;
private List<Ramos> listaramos;
private Long cdRamo;

@EJB(beanName = "ProductosDAOImpl")
private ProductosDAO productosDAO;
private Productos productos;
private List<Productos> listaProductos;
private Long cdProducto;

private Ubicacion ubicacion;
private Ubicacion ubicacionSel;

private ObjSeguro objSeguro;
private ObjSeguro objSeguroSel;
private ObjSeguro vehiculo;

private Subobjetos subobjetos;

@EJB(beanName = "MarcasDAOImpl")
private MarcasDAO marcasDAO;
private List<Marcas> listaMarcas;
private Marcas marcas;

@EJB(beanName = "ColoresVhDAOImpl")
private ColoresVhDAO coloresVhDAO;
private List<ColoresVh> listaColores;

@EJB(beanName = "DerechosEmisionDAOImpl")
private DerechosEmisionDAO derechosEmisionDAO;
private DerechosEmision derechosEmision;

private ComTpFactura comTpFactura;

private Double TotalAseg;
private Double TotalPrima;
```

```
    @EJB(beanName = "ComisionBrokerDAOImpl")
    private ComisionBrokerDAO comisionBrokerDAO;
    private ComisionBroker comisionBroker;

    @EJB(beanName = "ComisionesSubagentesDAOImpl")
    private ComisionesSubagentesDAO comisionesSubagentesDAO;
    private ComisionSubagentes comisionSubagentes;

    @PostConstruct
    @Override
    public void init() {
        try {
            cancel();
            findAll();
            cotizacion = (Cotizacion) FacesContext.getCurrentInstance().getExternalContext().getSessionMap().get("cotizacion");
            detCot = (DetalleCotizacion) FacesContext.getCurrentInstance().getExternalContext().getSessionMap().get("detalleCotizacion");
            clientes = cotizacion.getCdCliente();
            cdAseguradora = cotizacion.getCdAseguradora().getCdAseguradora();
            if (!clientes.getSubagentesList().isEmpty()) {
                subagentes = clientes.getSubagentesList().get(0);
            }

            if (!cotizacion.getFormaPagoList().isEmpty()) {
                formaPago = cotizacion.getFormaPagoList().get(0);
            }
        } catch (Exception e) {
        }
    }

    @Override
    public void create() {

    }

    @Override
    public void delete() {

    }

    public void deleteDetalleCotizacion(DetalleCotizacion detalle) {
        try {
            cotizacion.getDetalleCotizacionList().remove(detalle);
            detalleCotizacionDAO.delete(detalle);
            cotizacion = cotizacionDAO.findById(cotizacion.getCdCotizacion());
        } catch (Exception e) {
            Mensajes.mensajeError(e.getCause().getCause().getCause().getMessage());
        }
    }
```

```
    @Override
    public void update() {
        try {

            for (DetalleCotizacion det : cotizacion.getDetalleCotizacionList()) {
                if ((!det.getFactAseg().equals("") || det.getFactAseg() != null) &&
(!det.getNumPoliza().equals("") || det.getNumPoliza() != null)) {
                    det.setFcPoliza(new Date());
                    emision();
                }
            }
            cotizacionDAO.update(cotizacion);
            Mensajes.mensajeGrabarCorrecto();
            cotizacion = cotizacionDAO.findById(cotizacion.getCdCotizacion());
        } catch (Exception e) {
            Mensajes.mensajeError(e.getMessage());
        }

    }

    public void emision() {
        try {
            comisionBroker = comisionBrokerDAO.findByFk("WHERE
t.cdAseguradora.cdAseguradora = " +
cotizacion.getCdAseguradora().getCdAseguradora() + ""
                    + " AND t.cdRamo.cdRamo = " + detCot.getCdRamo().getCdRamo() +
" "
                    + " AND t.estado = 'A'").get(0);

            comisionSubagentes = comisionesSubagentesDAO.findByFk("WHERE
t.cdSubagente.cdSubagente = " + subagentes.getCdSubagente() + ""
                    + " AND t.estado = 'A'").get(0);
            Double pctComBrk = comisionBroker.getPctNuevo();
            Double pctComSuba = comisionSubagentes.getPctNuevo();
            for (FormaPago forma : cotizacion.getFormaPagoList()) {
                switch (forma.getTipoComBrk()) {
                    case 'P':
                        for (Financiamiento f : forma.getFinanciamientoList()) {
                            comTpFactura.setCdFinanciamiento(f);
                            comTpFactura.setCdSubagente(subagentes);
                            comTpFactura.setPctComBrk(pctComBrk);
                            comTpFactura.setPctComSuba(pctComSuba);

comTpFactura.setFactAseg(cotizacion.getDetalleCotizacionList().get(0).getFactAse
g());
                            if (f.getOrnidal() == 1) {
                                comTpFactura.setTotPrima(forma.getTotPrima());
                                comTpFactura.setValComBrk(comTpFactura.getTotPrima()     *
(pctComBrk / 100D));

comTpFactura.setSalAseguradora(comTpFactura.getValComBrk());
```

```
                        comTpFactura.setValComSuba(comTpFactura.getValComBrk()
* (pctComSuba / 100D));

comTpFactura.setSalComSuba(comTpFactura.getValComSuba());
                        comTpFactura.setFlgCobrable('1');
                    } else {
                        comTpFactura.setTotPrima(0D);
                        comTpFactura.setValComBrk(comTpFactura.getTotPrima()     *
(pctComBrk / 100D));

comTpFactura.setSalAseguradora(comTpFactura.getValComBrk());
                        comTpFactura.setValComSuba(comTpFactura.getValComBrk()
* (pctComSuba / 100D));

comTpFactura.setSalComSuba(comTpFactura.getValComSuba());
                        comTpFactura.setFlgCobrable('0');
                    }
                    f.getComTpFacturaList().add(comTpFactura);
                    comTpFactura = new ComTpFactura();
                }
                break;
            case 'C':
                Double cuota = forma.getTotPrima() / forma.getNumPagos();
                Double totPrima = forma.getTotPrima();
                cuota = Math.rint(cuota * 100) / 100;
                for (Financiamiento f : forma.getFinanciamientoList()) {
                    comTpFactura.setCdFinanciamiento(f);
                    comTpFactura.setCdSubagente(subagentes);
                    comTpFactura.setPctComBrk(pctComBrk);
                    comTpFactura.setPctComSuba(pctComSuba);

comTpFactura.setFactAseg(cotizacion.getDetalleCotizacionList().get(0).getFactAse
g());

                    if (f.getOrnidal() != forma.getNumPagos()) {
                        comTpFactura.setTotPrima(cuota);
                        comTpFactura.setValComBrk(cuota * (pctComBrk / 100D));
                        totPrima = totPrima - cuota;
                    } else {
                        comTpFactura.setTotPrima(totPrima);
                        comTpFactura.setValComBrk(totPrima * (pctComBrk / 100D));
                    }

comTpFactura.setSalAseguradora(comTpFactura.getValComBrk());
                        comTpFactura.setValComSuba(comTpFactura.getValComBrk()   *
(pctComSuba / 100D));
                        comTpFactura.setSalComSuba(comTpFactura.getValComSuba());
                        comTpFactura.setFlgCobrable('1');
                        f.getComTpFacturaList().add(comTpFactura);
                        comTpFactura = new ComTpFactura();
                    }
                    break;
            }
```

```
            }

        } catch (Exception e) {
        }
    }

    @Override
    public void findAll() {
        listaAseguradoras = aseguradorasDAO.findByFk("WHERE t.estado = 'A'");
        listaramos = ramoDAO.findByFk("WHERE t.estado = 'A'");
    }

    @Override
    public void find() {

    }

    @Override
    public void cancel() {
        cdAseguradora = 0L;
        cdProducto = 0L;
        cdRamo = 0L;
        clientes = new Clientes();
        cotizacion = new Cotizacion();
        detalleCotizacion = new DetalleCotizacion();
        financiamiento = new Financiamiento();
        formaPago = new FormaPago();
        objSeguro = new ObjSeguro();
        productos = new Productos();
        ramos = new Ramos();
        subagentes = new Subagentes();
        subobjetos = new Subobjetos();
        ubicacion = new Ubicacion();
        vehiculo = new ObjSeguro();
        comTpFactura = new ComTpFactura();
    }

    @Override
    public Boolean validate() {
        return true;
    }

    public void selectRamo() {
        listaProductos = productosDAO.findByFk("where t.cdRamo.cdRamo = " +
cdRamo + ""
                + " AND t.cdAseguradora.cdAseguradora = " + cdAseguradora);
    }

    public void selectMarca() {
        marcas = marcasDAO.findByFk("WHERE t.desMarca = '" +
objSeguro.getMarca() + "'").get(0);
    }
```

```
    public void calcPrima() {
        try {

objSeguro.setPrima(Funciones.calculaPrima(detalleCotizacionSel.getNumDias(),
objSeguro.getValAseg(), objSeguro.getFactor(), objSeguro.getTasa()));
        } catch (Exception e) {

        }
    }

    public int diasEntreFechas(Date fecha1, Date fecha2) {
        long startTime = fecha1.getTime();
        long endTime = fecha2.getTime();
        long diffTime = endTime - startTime;
        long diffDays = diffTime / (1000 * 60 * 60 * 24);
        return (int) diffDays;
    }

    public void finVigenciaCot() {
        cotizacion.setFcFin((new                Date(cotizacion.getFcInicio().getYear(),
cotizacion.getFcInicio().getMonth() + 1, cotizacion.getFcInicio().getDate())));
    }

    public void finVigencia() {
        detalleCotizacion.setFcFin((new Date(detalleCotizacion.getFcInicio().getYear()
+                1,                detalleCotizacion.getFcInicio().getMonth(),
detalleCotizacion.getFcInicio().getDate())));
    }

    public void addDetalleCotizacion() {
        try {
            detalleCotizacion.setCdRamo(ramoDAO.findById(cdRamo));
            detalleCotizacion.setCdSubagente(subagentes);
            detalleCotizacion.setCdProducto(productosDAO.findById(cdProducto));
            detalleCotizacion.setTotAseg(0D);
            detalleCotizacion.setTotPrima(0D);
            detalleCotizacion.setCdCotizacion(cotizacion);

detalleCotizacion.setNumDias(diasEntreFechas(detalleCotizacion.getFcInicio(),
detalleCotizacion.getFcFin()));
            cotizacion.getDetalleCotizacionList().add(detalleCotizacion);
            detalleCotizacion = new DetalleCotizacion();
        } catch (Exception e) {
            Mensajes.mensajeError(e.getCause().getCause().getMessage());
        }
    }

    public void consObjSeguro() {
        try {

            if
(!detalleCotizacionSel.getCdRamo().getDesRamo().equals("VEHICULOS")) {
                objSeguro.setFactor(100);
```

```
RequestContext.getCurrentInstance().execute("PF('dgObjSeguro').show();");
            } else {
                objSeguro.setFactor(100);
                listaMarcas = marcasDAO.findByFk("WHERE t.estado = 'A' order by
t.desMarca");
                listaColores = coloresVhDAO.findByFk("WHERE t.estado = 'A'");

RequestContext.getCurrentInstance().execute("PF('dgObjSeguro').show();");
            }
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }

    public void addUbicacion() {
        try {
            ubicacion.setCdDetCotizacion(detalleCotizacionSel);
            detalleCotizacionSel.getUbicacionList().add(ubicacion);
            ubicacion = new Ubicacion();
        } catch (Exception e) {
        }
    }

    public void addObjSeguro() {
        try {

            if (ubicacionSel == null) {
                objSeguro.setFcFin(detalleCotizacionSel.getFcFin());
                objSeguro.setFcInicio(detalleCotizacionSel.getFcInicio());
                objSeguro.setTotAseg(objSeguro.getValAseg());
                objSeguro.setTotPrima(objSeguro.getPrima());
                objSeguro.setCdDetCotizacion(detalleCotizacionSel);
                detalleCotizacionSel.getObjSeguroList().add(objSeguro);
            } else {
                objSeguro.setFcFin(ubicacionSel.getFcFin());
                objSeguro.setFcInicio(ubicacionSel.getFcInicio());
                objSeguro.setTotAseg(objSeguro.getValAseg());
                objSeguro.setTotPrima(objSeguro.getPrima());
                objSeguro.setCdUbicacion(ubicacionSel);
                ubicacionSel.getObjSeguroList().add(objSeguro);
            }

            objSeguro = new ObjSeguro();
        } catch (Exception e) {
        }
    }

    public void addSubobjeto() {
        try {
            subobjetos.setCdObjSeguro(objSeguroSel);
            objSeguroSel.getSubobjetosList().add(subobjetos);
            subobjetos = new Subobjetos();
```

```
        } catch (Exception e) {
        }
    }

    public void carVehiculos(ObjSeguro seguro) {
        vehiculo = seguro;
        RequestContext.getCurrentInstance().execute("PF('dgCarVh').show();");
    }

    public void totAseguradoUbicacion() {
        try {
            detalleCotizacionSel.setTotAseg(0D);
            detalleCotizacionSel.setTotPrima(0D);
            for (Ubicacion ubi : detalleCotizacionSel.getUbicacionList()) {
                detalleCotizacionSel.setTotAseg(detalleCotizacionSel.getTotAseg()      +
ubi.getTotAsegurado());
                detalleCotizacionSel.setTotPrima(detalleCotizacionSel.getTotPrima()    +
ubi.getTotPrima());
            }
        } catch (Exception e) {
        }
    }

    public void totAseguradoObj() {
        try {

            if
(detalleCotizacionSel.getCdRamo().getDesRamo().equals("VEHICULOS")) {
                detalleCotizacionSel.setTotAseg(0D);
                detalleCotizacionSel.setTotPrima(0D);
                for (ObjSeguro obj : detalleCotizacionSel.getObjSeguroList()) {
                    detalleCotizacionSel.setTotAseg(detalleCotizacionSel.getTotAseg()  +
obj.getTotAseg());
                    detalleCotizacionSel.setTotPrima(detalleCotizacionSel.getTotPrima() +
obj.getTotPrima());
                }
            } else {
                ubicacionSel.setTotAsegurado(0D);
                ubicacionSel.setTotPrima(0D);
                for (ObjSeguro obj : ubicacionSel.getObjSeguroList()) {
                    ubicacionSel.setTotAsegurado(ubicacionSel.getTotAsegurado()        +
obj.getTotAseg());
                    ubicacionSel.setTotPrima(ubicacionSel.getTotPrima()                +
obj.getTotPrima());
                }
            }
        } catch (Exception e) {
        }
    }

    public void totAsegSubobjeto() {
        try {
            objSeguroSel.setTotAseg(objSeguroSel.getValAseg());
```

```
            objSeguroSel.setTotPrima(objSeguroSel.getPrima());
            for (Subobjetos subobjeto : objSeguroSel.getSubobjetosList()) {
               objSeguroSel.setTotAseg(objSeguroSel.getTotAseg()                    +
subobjeto.getValAseg());
               objSeguroSel.setTotPrima(objSeguroSel.getTotPrima()                  +
subobjeto.getPrima());
            }
         } catch (Exception e) {
         }
      }

      public void addFormaPago() {
         try {

formaPago.setTotPrima(cotizacion.getDetalleCotizacionList().get(0).getTotPrima())
;
            derechosEmision = derechosEmisionDAO.findByFk("WHERE t.primaIni <=
" + formaPago.getTotPrima() + " "
                  + "AND t.primaFin >= " + formaPago.getTotPrima()).get(0);
            formaPago.setDerEmision(derechosEmision.getValor());
            formaPago.setSupBancos(formaPago.getTotPrima() * 0.035);
            formaPago.setSegCampecino(formaPago.getTotPrima() * 0.005);
            formaPago.setSubtotal(formaPago.getTotPrima()                        +
formaPago.getDerEmision()          +          formaPago.getSupBancos()          +
formaPago.getSegCampecino());
            formaPago.setPctIva(12);
            formaPago.setValIva((formaPago.getSubtotal())  *  (formaPago.getPctIva()  /
100D));
            formaPago.setTotPago(formaPago.getSubtotal() + formaPago.getValIva());

RequestContext.getCurrentInstance().execute("PF('dgFormaPago').show();");
         } catch (Exception e) {
         }
      }

      public void tipoPago() {
         try {
            if (formaPago.getFormaPago().equals("CONTADO")) {
               formaPago.setNumPagos(1);
            } else {
               formaPago.setNumPagos(6);
            }
         } catch (Exception e) {
         }
      }

      public void procesarFinanciamiento() {
         try {

            Double valCuota = formaPago.getTotPago() / formaPago.getNumPagos();
            valCuota = Math.rint(valCuota * 100) / 100;
            Double valTotal = formaPago.getTotPago();
            formaPago.setCdCotizacion(cotizacion);
```

```
            cotizacion.setFormaPagoList(new ArrayList<>());
            cotizacion.getFormaPagoList().add(formaPago);
            for (int i = 0; i < formaPago.getNumPagos(); i++) {
               financiamiento.setCdFormaPago(formaPago);
               financiamiento.setOrnidal(i + 1);
               financiamiento.setFcVencimiento(Funciones.sumarMeses(new Date(), i));
               if ((i + 1) == formaPago.getNumPagos()) {
                  financiamiento.setValor(valTotal);
                  financiamiento.setSaldo(valTotal);
               } else {
                  financiamiento.setValor(valCuota);
                  financiamiento.setSaldo(valCuota);
                  valTotal = valTotal - valCuota;
               }
               financiamiento.setAbono(0D);
               financiamiento.setSaldoPago(0D);
               financiamiento.setLetras("CUOTA " + (i + 1) + "/" +
formaPago.getNumPagos());
               formaPago.getFinanciamientoList().add(financiamiento);
               financiamiento = new Financiamiento();
            }
            RequestContext.getCurrentInstance().execute("PF('dgFormaPago').hide()");
         } catch (Exception e) {
            System.err.println(e.getMessage());
         }
      }

      public void fcVencimiento() {
         int cont = 0;
         Date fecha = formaPago.getFinanciamientoList().get(0).getFcVencimiento();
         for (Financiamiento f : formaPago.getFinanciamientoList()) {
            f.setFcVencimiento(Funciones.sumarMeses(fecha, cont));
            cont++;
         }
      }

      public Clientes getClientes() {
         return clientes;
      }

      public void setClientes(Clientes clientes) {
         this.clientes = clientes;
      }

      public Cotizacion getCotizacion() {
         return cotizacion;
      }

      public void setCotizacion(Cotizacion cotizacion) {
         this.cotizacion = cotizacion;
      }

      public FormaPago getFormaPago() {
```

```
        return formaPago;
    }

    public void setFormaPago(FormaPago formaPago) {
        this.formaPago = formaPago;
    }

    public Financiamiento getFinanciamiento() {
        return financiamiento;
    }

    public void setFinanciamiento(Financiamiento financiamiento) {
        this.financiamiento = financiamiento;
    }

    public Aseguradoras getAseguradoras() {
        return aseguradoras;
    }

    public void setAseguradoras(Aseguradoras aseguradoras) {
        this.aseguradoras = aseguradoras;
    }

    public List<Aseguradoras> getListaAseguradoras() {
        return listaAseguradoras;
    }

    public void setListaAseguradoras(List<Aseguradoras> listaAseguradoras) {
        this.listaAseguradoras = listaAseguradoras;
    }

    public Long getCdAseguradora() {
        return cdAseguradora;
    }

    public void setCdAseguradora(Long cdAseguradora) {
        this.cdAseguradora = cdAseguradora;
    }

    public DetalleCotizacion getDetalleCotizacion() {
        return detalleCotizacion;
    }

    public void setDetalleCotizacion(DetalleCotizacion detalleCotizacion) {
        this.detalleCotizacion = detalleCotizacion;
    }

    public Subagentes getSubagentes() {
        return subagentes;
    }

    public void setSubagentes(Subagentes subagentes) {
        this.subagentes = subagentes;
```

```
}

public Ramos getRamos() {
   return ramos;
}

public void setRamos(Ramos ramos) {
   this.ramos = ramos;
}

public List<Ramos> getListaramos() {
   return listaramos;
}

public void setListaramos(List<Ramos> listaramos) {
   this.listaramos = listaramos;
}

public Long getCdRamo() {
   return cdRamo;
}

public void setCdRamo(Long cdRamo) {
   this.cdRamo = cdRamo;
}

public Productos getProductos() {
   return productos;
}

public void setProductos(Productos productos) {
   this.productos = productos;
}

public List<Productos> getListaProductos() {
   return listaProductos;
}

public void setListaProductos(List<Productos> listaProductos) {
   this.listaProductos = listaProductos;
}

public Long getCdProducto() {
   return cdProducto;
}

public void setCdProducto(Long cdProducto) {
   this.cdProducto = cdProducto;
}

public Ubicacion getUbicacion() {
   return ubicacion;
}
```

```
public void setUbicacion(Ubicacion ubicacion) {
   this.ubicacion = ubicacion;
}

public ObjSeguro getObjSeguro() {
   return objSeguro;
}

public void setObjSeguro(ObjSeguro objSeguro) {
   this.objSeguro = objSeguro;
}

public Subobjetos getSubobjetos() {
   return subobjetos;
}

public void setSubobjetos(Subobjetos subobjetos) {
   this.subobjetos = subobjetos;
}

public DetalleCotizacion getDetalleCotizacionSel() {
   return detalleCotizacionSel;
}

public void setDetalleCotizacionSel(DetalleCotizacion detalleCotizacionSel) {
   this.detalleCotizacionSel = detalleCotizacionSel;
}

public ObjSeguro getObjSeguroSel() {
   return objSeguroSel;
}

public void setObjSeguroSel(ObjSeguro objSeguroSel) {
   this.objSeguroSel = objSeguroSel;
}

public List<Marcas> getListaMarcas() {
   return listaMarcas;
}

public void setListaMarcas(List<Marcas> listaMarcas) {
   this.listaMarcas = listaMarcas;
}

public Marcas getMarcas() {
   return marcas;
}

public void setMarcas(Marcas marcas) {
   this.marcas = marcas;
}
```

```
    public List<ColoresVh> getListaColores() {
        return listaColores;
    }

    public void setListaColores(List<ColoresVh> listaColores) {
        this.listaColores = listaColores;
    }

    public ObjSeguro getVehiculo() {
        return vehiculo;
    }

    public void setVehiculo(ObjSeguro vehiculo) {
        this.vehiculo = vehiculo;
    }

    public Double getIva() {
        return iva;
    }

    public void setIva(Double iva) {
        this.iva = iva;
    }

    public Double getTotalAseg() {
        return TotalAseg;
    }

    public void setTotalAseg(Double TotalAseg) {
        this.TotalAseg = TotalAseg;
    }

    public Double getTotalPrima() {
        return TotalPrima;
    }

    public void setTotalPrima(Double TotalPrima) {
        this.TotalPrima = TotalPrima;
    }


}
```

1.3.3. Clase BeanLogin

```
package com.broker.bean;

import com.broker.dao.PerfilesDAO;
import com.broker.dao.UsuariosDAO;
import com.broker.modelo.Perfil;
import com.broker.modelo.Usuarios;
import com.broker.util.EncriptarClaves;
import com.broker.util.EnviarMails;
import com.broker.util.Mensajes;
```

```
import java.io.Serializable;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import javax.faces.context.FacesContext;
import javax.servlet.http.HttpServletRequest;
import org.primefaces.context.RequestContext;

/**
 *
 * @author Christian Javier
 */
@ManagedBean
@SessionScoped
public class BeanLogin implements Serializable {

    private String user;
    private String clave;

    private String nuevaClave;
    private String confirmarClave;

    @EJB(beanName = "UsuariosDAOImpl")
    private UsuariosDAO usuariosDAO;
    private Usuarios usuario;
    private List<Usuarios> listaUsuarios;

    @EJB(beanName = "PerfilesDAOImpl")
    private PerfilesDAO perfilDao;
    private Perfil perfil;

    @PostConstruct
    public void init() {
        try {
            usuario = new Usuarios();
            perfil = new Perfil();
            if (usuariosDAO.findAll().isEmpty()) {
                RegistroAdmin();
            }
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }

    }

    public void RegistroAdmin() {
        try {
            usuario.setApellidos("SISTEMA");
            usuario.setCargo("ADMINISTRADOR");
            usuario.setCdPerfil(perfilDao.findById(1L));
            usuario.setCdUsuario(1L);
```

```
            usuario.setClaveActivacion(EncriptarClaves.getMD5("BROKER"));
            usuario.setEmail("ADMIN@ADMIN.COM");
            usuario.setEstado('A');
            usuario.setNombres("ADMINISTRADOR");
            usuario.setUsuario("BROKER");
            usuario.setClave(EncriptarClaves.getMD5("BROKER"));
            usuariosDAO.create(usuario);
        } catch (Exception e) {

        }
    }

    public void acceso() {
        try {
            listaUsuarios = usuariosDAO.findByFk("WHERE t.usuario = '" +
user.toUpperCase() + "' "
                    + "and t.clave = '" + EncriptarClaves.getMD5(clave.toUpperCase()) +
"'");
            if (listaUsuarios.size() > 0) {
                for (Usuarios us : listaUsuarios) {

FacesContext.getCurrentInstance().getExternalContext().getSessionMap().put("usua
rio", us);

FacesContext.getCurrentInstance().getExternalContext().getSessionMap().put("perfi
l", us.getCdPerfil());
                }

FacesContext.getCurrentInstance().getExternalContext().redirect("pages/FrmInicio.j
sf");
            } else {
                Mensajes.mensajeAdvertencia("Usuario y/o clave incorrectos");
            }
        } catch (Exception e) {
            Mensajes.mensajeError(e.getMessage());
        }
    }

    public void recuperarClave() {
        if (!usuariosDAO.findByFk("where t.email = '" + usuario.getEmail() +
"'").isEmpty()) {
            usuario = usuariosDAO.findByFk("where t.email = '" + usuario.getEmail() +
"'").get(0);
            String url = "<a
href='http://localhost:8083/broker/pages/FrmCambiarClave.jsf?cdUsuario=" +
usuario.getCdUsuario() + "'>Cambiar Clave</a>";
            String mensaje = "Estimado(a) " + usuario.getNombres() + " " +
usuario.getApellidos() + ", <br> "
                    + "Para recuperar su clave por favor dar clic en el siguiente enlase <br>
"
                    + url + "<br>";
            if (EnviarMails.SendMail(usuario.getEmail(), "", "Recuperar Clave",
mensaje)) {
```

```
                Mensajes.mensajeInformacion("Se envio un email a su correo para que
pueda realizar el cambio de clave.");
            } else {
                Mensajes.mensajeInformacion("No se pudo enviar el email, favor verificar
su cuenta de correo electronico o existen problemas de conexion");
            }
        } else {
            Mensajes.mensajeInformacion("Usuario no registrado");
        }
    }

    public String getUser() {
        return user;
    }

    public void setUser(String user) {
        this.user = user;
    }

    public String getClave() {
        return clave;
    }

    public void setClave(String clave) {
        this.clave = clave;
    }

    public Usuarios getUsuario() {
        return usuario;
    }

    public void setUsuario(Usuarios usuario) {
        this.usuario = usuario;
    }

    public String getNuevaClave() {
        return nuevaClave;
    }

    public void setNuevaClave(String nuevaClave) {
        this.nuevaClave = nuevaClave;
    }

    public String getConfirmarClave() {
        return confirmarClave;
    }

    public void setConfirmarClave(String confirmarClave) {
        this.confirmarClave = confirmarClave;
    }

}
```

1.3.4. Clase BeanComisionBroker.

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.broker.bean;

import com.broker.dao.AseguradorasDAO;
import com.broker.dao.ComisionBrokerDAO;
import com.broker.dao.RamoDAO;
import com.broker.modelo.Aseguradoras;
import com.broker.modelo.ComisionBroker;
import com.broker.modelo.Ramos;
import com.broker.modelo.Usuarios;
import com.broker.util.Mensajes;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
import javax.faces.context.FacesContext;
import org.primefaces.context.RequestContext;

/**
 *
 * @author Christian Javier
 */
@ManagedBean
@ViewScoped
public class BeanComisionBroker implements Serializable, Bean {

    @EJB(beanName = "AseguradorasDAOImpl")
    private AseguradorasDAO aseguradorasDAO;
    private Aseguradoras aseguradoras;
    private List<Aseguradoras> listaAseguradoras;
    private Long cdAseguradora;

    @EJB(beanName = "RamoDAOImpl")
    private RamoDAO ramosDAO;
    private List<Ramos> listaRamos;
    private Ramos ramos;
    private Long cdRamo;

    @EJB(beanName = "ComisionBrokerDAOImpl")
    private ComisionBrokerDAO comisionBrokerDAO;
    private ComisionBroker comisionBroker;
    private List<ComisionBroker> listaComisiones;
    private List<ComisionBroker> listaComisionesActivas;

    private Usuarios usuario;

    @PostConstruct
    @Override
    public void init() {
```

```
            cancel();
            usuario = (Usuarios)
FacesContext.getCurrentInstance().getExternalContext().getSessionMap().get("usuario");
        }

        @Override
        public void create() {
            try {
                comisionBroker.setCdAseguradora(aseguradoras);
                comisionBroker.setCdRamo(ramos);
                comisionBroker.setUsuario(usuario.getUsuario());
                if (listaComisionesActivas.isEmpty()) {
                    comisionBrokerDAO.create(comisionBroker);
                    RequestContext.getCurrentInstance().execute("PF('dgRamoComision').hide()");
                } else {
                    for (ComisionBroker comision : listaComisiones) {
                        if (!comision.getEstado().equals('I')) {
                            comision.setEstado('I');
                            comision.setObservacion("REGISTRO DESACTIVADO POR CAMBIO DE
COMISION REALIZADO POR " + usuario.getUsuario());
                            comisionBrokerDAO.update(comision);
                        }
                    }
                    comisionBrokerDAO.create(comisionBroker);
                    RequestContext.getCurrentInstance().execute("PF('dgRamoComision').hide()");

                }
                cancel();
                find();
                Mensajes.mensajeGrabarCorrecto();
            } catch (Exception e) {
                Mensajes.mensajeError(e.getLocalizedMessage());
            }
        }

        @Override
        public void delete() {

        }

        public void eliminar(ComisionBroker comision) {
            try {
                comision.setEstado('I');
                comision.setObservacion("REGISTRO DESACTIVADO POR " +
usuario.getUsuario());
                comisionBrokerDAO.update(comision);
                find();
            } catch (Exception e) {
                Mensajes.mensajeError(e.getLocalizedMessage());
            }
        }

        @Override
        public void update() {

        }

        @Override
        public void findAll() {
```

```
        listaAseguradoras = aseguradorasDAO.findByFk("where t.estado = 'A'");
        listaRamos = ramosDAO.findByFk("where t.estado = 'A'");
    }

    @Override
    public void find() {
        listaComisiones = comisionBrokerDAO.findByFk("WHERE
t.cdAseguradora.cdAseguradora = " + aseguradoras.getCdAseguradora() + ""
                + " and t.cdRamo.cdRamo = " + ramos.getCdRamo());
        listaComisionesActivas = comisionBrokerDAO.findByFk("WHERE
t.cdAseguradora.cdAseguradora = " + aseguradoras.getCdAseguradora() + ""
                + " and t.cdRamo.cdRamo = " + ramos.getCdRamo()+" "
                + "and t.estado = 'A'");
    }

    @Override
    public void cancel() {
        findAll();
        ramos = new Ramos();
        aseguradoras = new Aseguradoras();
        comisionBroker = new ComisionBroker();
        listaComisiones = new ArrayList<>();
        listaComisionesActivas = new ArrayList<>();
    }

    @Override
    public Boolean validate() {
        Boolean validacion = true;
        return validacion;
    }

    public void selectAseguradora() {
        ramos = new Ramos();
        listaComisiones = new ArrayList<>();
    }

    public List<Aseguradoras> getListaAseguradoras() {
        return listaAseguradoras;
    }

    public void setListaAseguradoras(List<Aseguradoras> listaAseguradoras) {
        this.listaAseguradoras = listaAseguradoras;
    }

    public Long getCdAseguradora() {
        return cdAseguradora;
    }

    public void setCdAseguradora(Long cdAseguradora) {
        this.cdAseguradora = cdAseguradora;
    }

    public List<Ramos> getListaRamos() {
        return listaRamos;
    }

    public void setListaRamos(List<Ramos> listaRamos) {
        this.listaRamos = listaRamos;
    }
```

```
    public Long getCdRamo() {
        return cdRamo;
    }

    public void setCdRamo(Long cdRamo) {
        this.cdRamo = cdRamo;
    }

    public ComisionBroker getComisionBroker() {
        return comisionBroker;
    }

    public void setComisionBroker(ComisionBroker comisionBroker) {
        this.comisionBroker = comisionBroker;
    }

    public Aseguradoras getAseguradoras() {
        return aseguradoras;
    }

    public void setAseguradoras(Aseguradoras aseguradoras) {
        this.aseguradoras = aseguradoras;
    }

    public Ramos getRamos() {
        return ramos;
    }

    public void setRamos(Ramos ramos) {
        this.ramos = ramos;
    }

    public List<ComisionBroker> getListaComisiones() {
        return listaComisiones;
    }

    public void setListaComisiones(List<ComisionBroker> listaComisiones) {
        this.listaComisiones = listaComisiones;
    }

    public Usuarios getUsuario() {
        return usuario;
    }

    public void setUsuario(Usuarios usuario) {
        this.usuario = usuario;
    }

}
```

5. Codigo de vistas del sistema.

Para las vistas del sistema se utilizo el frimework primefaces el cual permite un manejo mas rápido en el desarrollo.

5.1.Vista FrmCotizacion

```
<?xml version='1.0' encoding='UTF-8' ?>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:p="http://primefaces.org/ui"
      xmlns:f="http://xmlns.jcp.org/jsf/core"
      xmlns:c="http://xmlns.jcp.org/jsp/jstl/core">

    <body>

        <ui:composition template="./../WEB-INF/template.xhtml">

            <ui:define name="head">
            </ui:define>

            <ui:define name="content">
                <h:form id="frm">
                    <p:growl id="growl" autoUpdate="true" showDetail="true"
life="5000"/>
                    <p:panel id="pnlCotizacion">
                        <f:facet name="header">
                            Contizacion Aseguradoras
                        </f:facet>
                        <p:toolbar>
                            <f:facet name="left">
                                <h:panelGrid columns="3">
                                    <p:commandButton icon="fa fa-save"
                                                     value="Guardar"
                                                     action="#{beanCotizacion.update()}"
                                                     title="Guardar Cotizacion"
                                                     update=":frm,:frmDg"/>
                                    <p:commandButton icon="fa fa-print" value="Imprimir"
title="Imprimir Cotizacion"/>
                                    <p:commandButton icon="fa fa-envelope-o"
value="Enviar" title="Enviar por correo"/>
                                </h:panelGrid>
                            </f:facet>
                            <f:facet name="right" >
                                <p:outputLabel
value="#{beanCotizacion.cotizacion.numCotizacion}" style="font-size: 20px"/>
                            </f:facet>
                        </p:toolbar>



                        <p:fieldset style="margin: auto">
                            <h:panelGrid columns="2" style="border-spacing: 5px 20px;">
                                <h:panelGroup class="md-inputfield">
```

```
                        <p:inputText id="txtContratante"
                                value="#{beanCotizacion.clientes.nombres}
#{beanCotizacion.clientes.apellidos}"
                                size="40"
                                readonly="true"/>
                        <label>Contratante</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu id="somAseguradoras"
                                value="#{beanCotizacion.cdAseguradora}"
                                style="width: 200px"
                                filter="true"
                                autoWidth="false">

                            <f:selectItem itemLabel="Seleccionar..!"
itemValue="0"/>
                            <f:selectItems
value="#{beanCotizacion.listaAseguradoras}"
                                var="aseg"
                                itemLabel="#{aseg.desAseguradora}"
                                itemValue="#{aseg.cdAseguradora}"/>
                        </p:selectOneMenu>
                        <label>Aseguradora</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:inputText
value="#{beanCotizacion.subagentes.nombres}
#{beanCotizacion.subagentes.apellidos}"
                                size="40"
                                readonly="true"/>
                        <label>Sub Agente</label>
                    </h:panelGroup>

                    <h:panelGrid columns="2" >
                        <h:panelGroup class="md-inputfield">
                            <p:calendar id="calFcInicio"
                                locale="es"
                                mask="true"
                                pattern="dd/MM/yyyy"
                                autocomplete="true"
                                navigator="true"
                                size="15"
                                showOn="button"
                                value="#{beanCotizacion.cotizacion.fcInicio}">
                              <p:ajax event="dateSelect" update="calFcFin"
listener="#{beanCotizacion.finVigenciaCot()}"/>
                            </p:calendar>
```

```
                        <label>Fecha Desde</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:calendar id="calFcFin"
                                    locale="es"
                                    mask="true"
                                    pattern="dd/MM/yyyy"
                                    autocomplete="true"
                                    navigator="true"
                                    size="15"
                                    showOn="button"
                                    value="#{beanCotizacion.cotizacion.fcFin}"/>
                        <label>Fecha Hasta</label>
                    </h:panelGroup>
                </h:panelGrid>

            </h:panelGrid>

        </p:fieldset>
        <p:separator/>
        <p:tabView id="tvCotizacion">
            <p:tab title="Detalle Cotizacion">
                <h:panelGrid columns="5" style="border-spacing: 5px
15px">
                    <p:commandButton icon="fa fa-list-alt"
                                     value="Añadir"

action="#{beanCotizacion.addDetalleCotizacion()}"
                                     update="dtDetalleCotizacion, :frm:growl"/>
                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu
value="#{beanCotizacion.cdRamo}">
                            <p:ajax listener="#{beanCotizacion.selectRamo()}"
update="somProducto"/>
                            <f:selectItem itemLabel="Seleccionar..!"
itemValue="0"/>
                            <f:selectItems value="#{beanCotizacion.listaramos}"
                                           var="ramo"
                                           itemLabel="#{ramo.desRamo}"
                                           itemValue="#{ramo.cdRamo}"/>
                        </p:selectOneMenu>
                        <label>Ramo</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu id="somProducto"
value="#{beanCotizacion.cdProducto}">
```

```
                        <f:selectItem itemLabel="Seleccionar..!"
itemValue="0"/>
                        <f:selectItems
value="#{beanCotizacion.listaProductos}"
                                var="producto"
                                itemLabel="#{producto.desProducto}"
                                itemValue="#{producto.cdProducto}"/>
                    </p:selectOneMenu>
                    <label>Producto</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:calendar id="fcDesde"
                            locale="es"
                            mask="true"
                            pattern="dd/MM/yyyy"
                            autocomplete="true"
                            navigator="true"
                            size="15"
                            showOn="button"

value="#{beanCotizacion.detalleCotizacion.fcInicio}">
                        <p:ajax event="dateSelect"
listener="#{beanCotizacion.finVigencia()}" update="fcHasta"/>
                    </p:calendar>
                    <label>Fc. Desde</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:calendar id="fcHasta"
                            locale="es"
                            mask="true"
                            pattern="dd/MM/yyyy"
                            autocomplete="true"
                            navigator="true"
                            size="15"
                            showOn="button"

value="#{beanCotizacion.detalleCotizacion.fcFin}"/>
                    <label>Fc. Hasta</label>
                </h:panelGroup>
            </h:panelGrid>
            <p:separator/>
            <p:dataTable id="dtDetalleCotizacion"
                    var="detalleCotizacion"

value="#{beanCotizacion.cotizacion.detalleCotizacionList}"
                    selection="#{beanCotizacion.detalleCotizacionSel}"
                    selectionMode="single"
```

```
rowKey="#{detalleCotizacion.cdCotizacion.cdCotizacion}">
                        <p:ajax event="rowSelect"
listener="#{beanCotizacion.consObjSeguro()}"
update=":frmDg:pnlObjSeguro"/>
                        <p:column headerText="Ramo" style="text-align:
center">
                            <p:outputLabel
value="#{detalleCotizacion.cdRamo.desRamo}"/>
                        </p:column>

                        <p:column headerText="Producto" style="text-align:
center">
                            <p:outputLabel
value="#{detalleCotizacion.cdProducto.desProducto}"/>
                        </p:column>

                        <p:column headerText="Val. Asegurado" style="text-
align: center">
                            <p:outputLabel value="#{detalleCotizacion.totAseg}">
                              <f:convertNumber type="currency"
currencySymbol="$" />
                            </p:outputLabel>
                        </p:column>

                        <p:column headerText="Prima" style="text-align:
center">
                            <p:outputLabel
value="#{detalleCotizacion.totPrima}">
                              <f:convertNumber type="currency"
currencySymbol="$" />
                            </p:outputLabel>
                        </p:column>

                        <p:column headerText="Fc Desde" style="text-align:
center">
                            <p:outputLabel value="#{detalleCotizacion.fcInicio}">
                              <f:convertDateTime pattern="dd/MM/yyyy"/>
                            </p:outputLabel>
                        </p:column>

                        <p:column headerText="Fc Hasta" style="text-align:
center">
                            <p:outputLabel value="#{detalleCotizacion.fcFin}">
                              <f:convertDateTime pattern="dd/MM/yyyy"/>
                            </p:outputLabel>
                        </p:column>
```

```
                        <p:column headerText="Dias Seguro" style="text-align:
center">
                            <p:outputLabel
value="#{detalleCotizacion.numDias}"/>
                        </p:column>

                        <p:column width="35" style="text-align: center">
                            <p:commandButton icon="fa fa-trash"
                                             style="border-radius: 5px"
                                             styleClass="red-btn"

action="#{beanCotizacion.deleteDetalleCotizacion(detalleCotizacion)}"
                                             update=":frm">
                                <p:confirm header="Confirmation" message="Esta
Seguro de Eliminar?" icon="pi pi-exclamation-triangle" />
                            </p:commandButton>
                            <p:confirmDialog global="true" showEffect="fade"
hideEffect="fade">
                                <p:commandButton value="Si" type="button"
styleClass="ui-confirmdialog-yes" icon="pi pi-check" />
                                <p:commandButton value="No" type="button"
styleClass="ui-confirmdialog-no" icon="pi pi-times" />
                            </p:confirmDialog>
                        </p:column>

                    </p:dataTable>
                </p:tab>

                <p:tab title="Financiamiento">
                    <p:commandButton icon="fa fa-dollar"
                                     value="Forma de Pago"
                                     action="#{beanCotizacion.addFormaPago()}"
                                     update=":frmDg:pnlFormaPago"/>

                    <p:commandButton icon="fa fa-trash"
                                     value="Borrar Forma de pago"
                                     update=":frmDg:pnlFormaPago"/>
                    <p:separator/>
                    <p:dataTable id="dtFinanciamiento"
                                 scrollable="true"
                                 scrollRows="4"
                                 scrollHeight="150"

value="#{beanCotizacion.formaPago.financiamientoList}"
                                 var="financiamiento">
                        <p:column headerText="Ord." width="25" style="text-
align: center">
                            <p:outputLabel value="#{financiamiento.ornidal}"/>
                        </p:column>
```

```
                        <p:column headerText="Valor" style="text-align: center">
                            <p:outputLabel value="#{financiamiento.valor}">
                                <f:convertNumber maxFractionDigits="2"
currencySymbol="$"/>
                            </p:outputLabel>
                        </p:column>
                        <p:column headerText="Saldo" style="text-align: center">
                            <p:outputLabel value="#{financiamiento.saldo}">
                                <f:convertNumber maxFractionDigits="2"
currencySymbol="$"/>
                            </p:outputLabel>
                        </p:column>
                        <p:column headerText="Documento" style="text-align:
center">
                            <p:outputLabel value="#{financiamiento.factAseg}"/>
                        </p:column>
                        <p:column  style="text-align: center" >
                            <f:facet name="header">
                                <p:commandLink value="Fc. Vencimiento"
style="color: #fff" action="#{beanCotizacion.fcVencimiento()}"
update="dtFinanciamiento"/>
                            </f:facet>
                            <p:calendar value="#{financiamiento.fcVencimiento}"
size="11"  mask="true" pattern="dd/MM/yyyy" showOn="button"/>
                        </p:column>
                        <p:column headerText="Letras" style="text-align:
center">
                            <p:outputLabel value="#{financiamiento.letras}"/>
                        </p:column>
                    </p:dataTable>
                </p:tab>
            </p:tabView>
        </p:panel>
    </h:form>

    <h:form id="frmDg">
        <p:dialog id="dgFormaPago"
                  widgetVar="dgFormaPago"
                  modal="true"
                  responsive="true"
                  closeOnEscape="true"
                  showEffect="drop"
                  hideEffect="drop"
                  position="center"
                  style="margin-top: 25px">

            <p:panel id="pnlFormaPago">
                <h:panelGrid columns="2">
                    <h:panelGrid columns="2">
```

```
                        <p:outputLabel value="Forma Pago"/>
                        <p:selectOneMenu
value="#{beanCotizacion.formaPago.formaPago}"
                                         autoWidth="false"
                                         style="width: 200px">
                           <p:ajax listener="#{beanCotizacion.tipoPago()}"
update="txtNumPagos"/>
                           <f:selectItem itemLabel=" " itemValue=" "/>
                           <f:selectItem itemLabel="Contado"
itemValue="CONTADO"/>
                           <f:selectItem itemLabel="Credito"
itemValue="CREDITO"/>
                           <f:selectItem itemLabel="Debito Bancario"
itemValue="DEBITO"/>
                        </p:selectOneMenu>

                        <p:outputLabel value="Prima Neta"/>
                        <p:inputText
value="#{beanCotizacion.formaPago.totPrima}" readonly="true"/>

                        <p:outputLabel value="Der. Emision"/>
                        <p:inputNumber
value="#{beanCotizacion.formaPago.derEmision}" />

                        <p:outputLabel value="Sup. Bancos"/>
                        <p:inputNumber
value="#{beanCotizacion.formaPago.supBancos}" />

                        <p:outputLabel value="Seg. Campecino"/>
                        <p:inputNumber
value="#{beanCotizacion.formaPago.segCampecino}" />

                        <p:outputLabel value="Otro Valor"/>
                        <p:inputNumber
value="#{beanCotizacion.formaPago.otroValor}" />

                        <p:outputLabel value="Subtotal"/>
                        <p:inputText id="txtSubtotal"
value="#{beanCotizacion.formaPago.subtotal}" readonly="true" />

                        <h:panelGroup class="md-inputfield">
                           <p:selectOneMenu
value="#{beanCotizacion.formaPago.pctIva}">
                              <f:selectItem itemLabel="12%" itemValue="12"/>
                              <f:selectItem itemLabel="14%" itemValue="14"/>
                              <f:selectItem itemLabel="0%" itemValue="0"/>
                           </p:selectOneMenu>
                           <label>Iva</label>
                        </h:panelGroup>
```

```
                        <p:inputNumber id="txtValIva"
value="#{beanCotizacion.formaPago.valIva}" />

                        <p:outputLabel value="Prima Total"/>
                        <p:inputText id="txtTotPago"
value="#{beanCotizacion.formaPago.totPago}" readonly="true" />
                    </h:panelGrid>
                    <h:panelGrid columns="2">
                        <p:outputLabel value="Num. Pagos"/>
                        <p:inputText id="txtNumPagos"
value="#{beanCotizacion.formaPago.numPagos}" onkeypress="return
soloNumeros(event)"/>

                        <p:outputLabel value="% Cuota Inicial"/>
                        <p:inputText
value="#{beanCotizacion.formaPago.pctCuotaInicial}" onkeypress="return
soloNumeros(event)"/>

                        <p:outputLabel value="Comision Broker"/>
                        <p:selectOneMenu
value="#{beanCotizacion.formaPago.tipoComBrk}">
                            <f:selectItem itemLabel="Un Solo pago"
itemValue="P"/>
                            <f:selectItem itemLabel="Cuotas" itemValue="C"/>
                        </p:selectOneMenu>

                        <p:outputLabel value="Comision Subagente"/>
                        <p:selectOneMenu
value="#{beanCotizacion.formaPago.tipoComSuba}">
                            <f:selectItem itemLabel="Un Solo pago"
itemValue="P"/>
                            <f:selectItem itemLabel="Cuotas" itemValue="C"/>
                        </p:selectOneMenu>


                        <p:outputLabel value="Observacion"/>
                        <p:inputTextarea
value="#{beanCotizacion.formaPago.observaciones}"/>

                        <p:commandButton value="Procesar"

action="#{beanCotizacion.procesarFinanciamiento()}"
                                         update=":frm:tvCotizacion:dtFinanciamiento"/>
                    </h:panelGrid>
                </h:panelGrid>
            </p:panel>

        </p:dialog>
```

```
                <p:dialog id="dgUbicacion"
                        widgetVar="dgUbicacion"
                        modal="true"
                        responsive="true"
                        closeOnEscape="true"
                        showEffect="drop"
                        hideEffect="drop"
                        position="top"
                        style="margin-top: 25px">
                    <p:ajax event="close"
listener="#{beanCotizacion.totAseguradoUbicacion()}"
update=":frm:tvCotizacion:dtDetalleCotizacion"/>
                    <h:panelGrid columns="2">
                       <h:panelGroup class="md-inputfield">
                          <p:inputText
value="#{beanCotizacion.ubicacion.desUbicacion}"/>
                          <label>Ubicacion</label>
                       </h:panelGroup>
                       <p:commandButton icon="fa fa-plus"
                                style="border-radius: 5px"
                                action="#{beanCotizacion.addUbicacion()}"
                                update="dtUbicacion"/>
                    </h:panelGrid>

                    <p:separator/>
                    <p:dataTable id="dtUbicacion"

value="#{beanCotizacion.detalleCotizacionSel.ubicacionList}"
                             var="ubicacion">
                       <p:column headerText="Ubicacion">
                          <p:outputLabel value="#{ubicacion.desUbicacion}"/>
                       </p:column>

                       <p:column headerText="Fecha Desde">
                          <p:outputLabel value="#{ubicacion.fcInicio}"/>
                       </p:column>

                       <p:column headerText="Fecha Hasta">
                          <p:outputLabel value="#{ubicacion.fcFin}"/>
                       </p:column>

                       <p:column headerText="Tot. Asegurado">
                          <p:outputLabel value="#{ubicacion.totAsegurado}">
                             <f:convertNumber type="currency" currencySymbol="$"
/>
                          </p:outputLabel>
                       </p:column>

                       <p:column headerText="Tot. Prima">
```

```
                        <p:outputLabel value="#{ubicacion.totPrima}">
                            <f:convertNumber type="currency" currencySymbol="$"
/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="">
                        <p:commandButton icon="fa fa-inbox"
                                         style="border-radius: 5px"
                                         title="Ver o Agregar Objetos asegurados"
                                         />
                    </p:column>

                    <p:column headerText="">
                        <p:commandButton icon="fa fa-trash" style="border-radius:
5px">
                            <p:confirm header="Confirmation" message="Desea
eliminar la ubicacion seleccionada y los objetos que contiene?" icon="fa fa-
exclamation-triangle" />
                        </p:commandButton>
                        <p:confirmDialog global="true" showEffect="fade"
hideEffect="fade">
                            <p:commandButton value="Si" type="button"
styleClass="ui-confirmdialog-yes" icon="pi pi-check" />
                            <p:commandButton value="No" type="button"
styleClass="ui-confirmdialog-no" icon="pi pi-times" />
                        </p:confirmDialog>
                    </p:column>
                </p:dataTable>
            </p:dialog>
            <!--
                ************************************************

              REGISTRO OBJ SEGURO

                ************************************************
            -->


            <p:dialog id="dgObjSeguro"
                      widgetVar="dgObjSeguro"
                      modal="true"
                      responsive="true"
                      closeOnEscape="true"
                      showEffect="drop"
                      hideEffect="drop"
                      position="top"
                      style="margin-top: 25px"
                      width="1100">
```

```
                    <p:ajax event="close"
listener="#{beanCotizacion.totAseguradoObj()}"
update=":frm:tvCotizacion:dtDetalleCotizacion, :frmDg:dtUbicacion"/>
                    <p:panel id="pnlObjSeguro">
                       <h:panelGrid columns="1">
                          <p:fieldset legend="Item Asegurado">
                             <h:panelGrid columns="9">
                                <h:panelGroup class="md-inputfield">
                                   <p:inputText
value="#{beanCotizacion.objSeguro.desObjeto}" size="30"/>
                                   <label>Item Asegurado</label>
                                </h:panelGroup>

                                <h:panelGroup class="md-inputfield">
                                   <p:inputNumber
value="#{beanCotizacion.objSeguro.valAseg}">
                                      <p:ajax event="blur"
listener="#{beanCotizacion.calcPrima()}" update="txtPrimaSeg"/>
                                   </p:inputNumber>
                                   <label>Val. Asegurado</label>
                                </h:panelGroup>

                                <h:panelGroup class="md-inputfield">
                                   <p:inputNumber
value="#{beanCotizacion.objSeguro.tasa}" size="6">
                                      <p:ajax event="blur"
listener="#{beanCotizacion.calcPrima()}" update="txtPrimaSeg"/>
                                   </p:inputNumber>
                                   <label>Tasa</label>
                                </h:panelGroup>
                                <p:selectOneRadio id="sorFactor"
value="#{beanCotizacion.objSeguro.factor}">
                                   <f:selectItem itemLabel="100" itemValue="100"/>
                                   <f:selectItem itemLabel="1000" itemValue="1000"/>
                                </p:selectOneRadio>
                                <h:panelGroup class="md-inputfield">
                                   <p:inputNumber id="txtPrimaSeg"
value="#{beanCotizacion.objSeguro.prima}"/>
                                   <label>Prima</label>
                                </h:panelGroup>

                                <p:commandButton icon="fa fa-plus"
                                                 title="Agregar Objeto"
                                                 style="border-radius: 5px"
                                                 action="#{beanCotizacion.addObjSeguro()}"
                                                 update="dtObjSeguro,
:frm:tvCotizacion:dtDetalleCotizacion"/>
                             </h:panelGrid>
                          </p:fieldset>
```

```
                    <p:fieldset  legend="Vehiculo"
rendered="#{(beanCotizacion.detalleCotizacionSel.cdRamo.desRamo ==
'VEHICULOS')}">
                        <h:panelGrid columns="9">
                            <h:panelGroup class="md-inputfield">
                                <p:selectOneMenu id="somMarcas"
value="#{beanCotizacion.objSeguro.marca}"
                                                 autoWidth="false"
                                                 filter="true"
                                                 style="width: 150px">
                                    <p:ajax
listener="#{beanCotizacion.selectMarca()}" update="somModelo"/>
                                    <f:selectItem itemLabel="Seleccionar"
itemValue="0"/>
                                    <f:selectItems
value="#{beanCotizacion.listaMarcas}"
                                                   var="marca"
                                                   itemLabel="#{marca.desMarca}"
                                                   itemValue="#{marca.desMarca}"/>
                                </p:selectOneMenu>
                                <label>Marca</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:selectOneMenu id="somModelo"
value="#{beanCotizacion.objSeguro.modelo}"
                                                 autoWidth="false"
                                                 filter="true"
                                                 style="width: 150px">
                                    <f:selectItem itemLabel="Seleccionar"
itemValue="0"/>
                                    <f:selectItems
value="#{beanCotizacion.marcas.modelosList}"
                                                   var="modelo"
                                                   itemLabel="#{modelo.desModelo}"
                                                   itemValue="#{modelo.desModelo}"/>
                                </p:selectOneMenu>
                                <label>Modelo</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:selectOneMenu id="somColor"
value="#{beanCotizacion.objSeguro.color}"
                                                 autoWidth="false"
                                                 style="width: 100px"
                                                 filter="true">
                                    <f:selectItem itemLabel="Seleccionar"
itemValue="0"/>
```

```
                                    <f:selectItems
value="#{beanCotizacion.listaColores}"
                                                  var="color"
                                                  itemLabel="#{color.desColor}"
                                                  itemValue="#{color.desColor}"/>
                                </p:selectOneMenu>
                                <label>Color</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:inputText
value="#{beanCotizacion.objSeguro.chasis}"/>
                                <label>Chasis</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:inputText
value="#{beanCotizacion.objSeguro.motor}"/>
                                <label>Motor</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:inputText
value="#{beanCotizacion.objSeguro.placa}" size="10"/>
                                <label>Placa</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:selectOneMenu id="somTipoVh"

value="#{beanCotizacion.objSeguro.tipoVehiculo}">
                                    <f:selectItem itemLabel="Particular"
itemValue="PARTICULAR"/>
                                    <f:selectItem itemLabel="Comercial"
itemValue="COMERCIAL"/>
                                </p:selectOneMenu>
                                <label>Tipo Vehiculo</label>
                            </h:panelGroup>

                            <h:panelGroup class="md-inputfield">
                                <p:inputText
value="#{beanCotizacion.objSeguro.anio}" size="6" onkeypress="return
soloNumeros(event)"/>
                                <label>Año</label>
                            </h:panelGroup>
                        </h:panelGrid>
                    </p:fieldset>
                </h:panelGrid>
```

```
                    <!--
                    ***********************************************

                    TABLA OBJ SEGURO

                    ***********************************************
                    -->
                    <p:dataTable id="dtObjSeguro"

value="#{beanCotizacion.detalleCotizacionSel.objSeguroList}"
                                 var="objSeguro"
                                 scrollable="true"
                                 scrollWidth="max-content"
                                 selectionMode="single"
                                 selection="#{beanCotizacion.objSeguroSel}"

rowKey="#{objSeguro.cdDetCotizacion.cdCotizacion.cdCotizacion}">
                        <p:column headerText="Item Asegurado" width="200"
style="text-align: center">
                            <p:outputLabel value="#{objSeguro.desObjeto}"/>
                        </p:column>
                        <p:column headerText="Val. Asegurado" width="60"
style="text-align: center">
                            <p:outputLabel value="#{objSeguro.valAseg}">
                              <f:convertNumber type="currency"
currencySymbol="$" />
                            </p:outputLabel>
                        </p:column>
                        <p:column headerText="Tasa" width="30" style="text-align:
center">
                            <p:outputLabel value="#{objSeguro.tasa}%"/>
                        </p:column>

                        <p:column headerText="Factor" width="30" style="text-
align: center">
                            <p:outputLabel value="#{objSeguro.factor}"/>
                        </p:column>

                        <p:column headerText="Prima" width="60" style="text-
align: center">
                            <p:outputLabel value="#{objSeguro.prima}">
                              <f:convertNumber type="currency"
currencySymbol="$" />
                            </p:outputLabel>
                        </p:column>

                        <p:column headerText="Fc. Desde" width="60" style="text-
align: center">
                            <p:outputLabel value="#{objSeguro.fcInicio}">
```

```
                        <f:convertDateTime pattern="dd/MM/yyyy"/>
                    </p:outputLabel>
                </p:column>

                <p:column headerText="Fc. Hasta" width="60" style="text-
align: center">
                    <p:outputLabel value="#{objSeguro.fcFin}">
                        <f:convertDateTime pattern="dd/MM/yyyy"/>
                    </p:outputLabel>
                </p:column>

                <p:column headerText="Tot. Asegurado" width="60"
style="text-align: center">
                    <p:outputLabel value="#{objSeguro.totAseg}">
                        <f:convertNumber type="currency"
currencySymbol="$" />
                    </p:outputLabel>
                </p:column>

                <p:column headerText="Tot. Prima" width="60" style="text-
align: center">
                    <p:outputLabel value="#{objSeguro.totPrima}">
                        <f:convertNumber type="currency"
currencySymbol="$" />
                    </p:outputLabel>
                </p:column>

                <p:column width="35" style="text-align: center"
rendered="#{(beanCotizacion.detalleCotizacionSel.cdRamo.desRamo ==
'VEHICULOS')}">
                    <p:commandButton icon="fa fa-search"
                                     title="Caracteristicas"
                                     style="border-radius: 5px"
                                     update=":frmDg:pnlVehiculo"

action="#{beanCotizacion.carVehiculos(objSeguro)}"/>
                </p:column>

                <p:column width="35" style="text-align: center">

                    <p:commandButton icon="fa fa-trash" title="Eliminar"
style="border-radius: 5px" styleClass="red-btn">
                        <p:confirm header="Confirmation" message="Desea
eliminar el Objeto asegurado?" icon="fa fa-exclamation-triangle" />
                    </p:commandButton>
                    <p:confirmDialog global="true" showEffect="fade"
hideEffect="fade">
                        <p:commandButton value="Si" type="button"
styleClass="ui-confirmdialog-yes" icon="pi pi-check" />
```

```
                        <p:commandButton value="No" type="button"
styleClass="ui-confirmdialog-no" icon="pi pi-times" />
                    </p:confirmDialog>
                </p:column>
            </p:dataTable>
        </p:panel>
    </p:dialog>

    <p:dialog id="dgCarVh"
            widgetVar="dgCarVh"
            modal="true"
            responsive="true"
            closeOnEscape="true"
            showEffect="drop"
            hideEffect="drop">
        <p:panel id="pnlVehiculo">
            <h:panelGrid columns="1" style="border-spacing: 5px 20px">
                <h:panelGroup class="md-inputfield">
                    <p:inputText
value="#{beanCotizacion.vehiculo.marca}"/>
                    <label>Marca</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText
value="#{beanCotizacion.vehiculo.modelo}"/>
                    <label>Modelo</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText value="#{beanCotizacion.vehiculo.color}"/>
                    <label>Color</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText
value="#{beanCotizacion.vehiculo.marca}"/>
                    <label>Placa</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText
value="#{beanCotizacion.vehiculo.chasis}"/>
                    <label>Chasis</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText
value="#{beanCotizacion.vehiculo.motor}"/>
```

```
                        <label>Motor</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:inputText
value="#{beanCotizacion.vehiculo.tipoVehiculo}"/>
                        <label>Tipo Vehiculo</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:inputText value="#{beanCotizacion.vehiculo.anio}"/>
                        <label>Año</label>
                    </h:panelGroup>
                </h:panelGrid>
            </p:panel>
        </p:dialog>


        <p:dialog id="dgSubobjeto"
                  widgetVar="dgSubobjeto"
                  modal="true"
                  responsive="true"
                  closeOnEscape="true"
                  showEffect="drop"
                  hideEffect="drop">
            <p:ajax event="close"
listener="#{beanCotizacion.totAsegSubobjeto()}"
update=":frmDg:dtObjSeguro"/>
            <p:panel id="pnlSubobjetos">

            </p:panel>

        </p:dialog>
    </h:form>


    </ui:define>

</ui:composition>

</body>
</html>
```

5.2.Vista FrmAccesos

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
```

```
    xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
    xmlns:p="http://primefaces.org/ui"
    xmlns:h="http://xmlns.jcp.org/jsf/html"
    xmlns:f="http://xmlns.jcp.org/jsf/core">

  <body>

    <ui:composition template="./../WEB-INF/template.xhtml">

      <ui:define name="head">

      </ui:define>

      <ui:define name="content">
        <h:form id="frm">
          <p:growl id="grlAccesos" autoUpdate="true" showDetail="true"
life="5000"/>
          <p:panel id="pnlAccesos">
            <f:facet name="header">
              Accesos
            </f:facet>


            <table width="100%" >
              <tr>
                <td valign="top" width="50%">

                  <p:fieldset legend="Perfiles" style="vertical-align:
middle">
                    <p:commandButton id="btnAgregar"
                                     value="Agregar"
                                     onclick="PF('dgAccesos').show();"
                                     icon="fa fa-plus"
                                     styleClass="secondary"/>
                    <p:separator/>

                    <p:dataTable id="dtPerfiles"
                                 value="#{beanAccesos.listaPerfiles}"
                                 var="perfil"
                                 selectionMode="single"
                                 selection="#{beanAccesos.perfil}"
                                 rowKey="#{perfil.cdPerfil}">
                      <p:ajax event="rowSelect"
listener="#{beanAccesos.find()}" update=":frm:dtAccesos" />
                      <p:column headerText="Perfil">
                        <p:outputLabel value="#{perfil.desPerfil}"/>
                      </p:column>
                    </p:dataTable>
```

```
                        </p:fieldset>
                    </td>
                    <td valign="top" width="50%">
                        <p:fieldset legend="Menus" style="vertical-align:
middle">
                            <p:commandButton id="btnEditar"
                                    value="Editar"
                                    onclick="PF('dgAccesosPerfil').show();"
                                    icon="fa fa-edit"
                                    styleClass="secondary"/>
                            <p:separator/>
                            <p:dataTable id="dtAccesos"
                                    value="#{beanAccesos.selectedMenu}"
                                    var="acceso"
                                    scrollable="true"
                                    scrollHeight="170">
                                <p:column headerText="Menu">
                                    <p:outputLabel value="#{acceso.desMenu}"/>
                                </p:column>

                            </p:dataTable>
                        </p:fieldset>
                    </td>
                </tr>
            </table>
        </p:panel>
    </h:form>

    <h:form id="frmDgAccesos">
        <p:dialog id="dgAccesos"
                widgetVar="dgAccesos"
                modal="true"
                showEffect="drop"
                hideEffect="drop"
                header="Agregar perfil">
            <p:ajax event="open" listener="#{beanAccesos.cancel()}"/>
            <h:panelGrid columns="2">
                <p:outputLabel value="Descripcion"/>
                <p:inputText id="txtDesPerfil"
                        value="#{beanAccesos.perfil.desPerfil}"
                        onkeydown="this.value =
this.value.toUpperCase()"/>
            </h:panelGrid>
            <p:commandButton value="Gardar"
                        icon="fa fa-floppy-o"
                        actionListener="#{beanAccesos.create()}"/>
        </p:dialog>
    </h:form>
```

```
            <h:form id="frmDgAccesosPerfil">
              <p:dialog id="dgAccesosPerfil"
                        widgetVar="dgAccesosPerfil"
                        modal="true"
                        showEffect="drop"
                        hideEffect="drop"
                        header="ACCESOS PERFIL
#{accesosBean.perfiles.desPerfil}">
                <p:ajax event="open" listener="#{beanAccesos.editar()}"
update=":frmDgAccesosPerfil:dtMenu"/>
                <p:ajax event="close" listener="#{beanAccesos.cancel()}"
update=":frmDgAccesosPerfil:dtMenu"/>
                <p:dataTable id="dtMenu"
                             var="menu"
                             scrollable="true"
                             scrollHeight="200"
                             value="#{beanAccesos.listaMenu}"
                             selection="#{beanAccesos.selectedMenu}"
                             rowKey="#{menu.cdMenu}"
                             style="width: 400px">
                  <f:facet name="Header">
                    Menus del Sistema
                  </f:facet>
                  <p:column headerText="Submenu">
                    <p:outputLabel value="#{menu.submenu.desMenu}"/>
                  </p:column>
                  <p:column headerText="Descripcion">
                    <p:outputLabel value="#{menu.desMenu}"/>
                  </p:column>
                  <p:column selectionMode="multiple"
style="width:16px;text-align:center"/>
                </p:dataTable>
                <p:commandButton value="Guardar"
                                 title="Guardar"
                                 actionListener="#{beanAccesos.update()}"
                                 icon="fa fa-floppy-o"
                                 update=":frm, :frmDgAccesos,
:frmDgAccesosPerfil"/>

              </p:dialog>
            </h:form>
          </ui:define>
        </ui:composition>

    </body>
</html>
```

5.3.Vista FrmComisionBroker

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:p="http://primefaces.org/ui"
      xmlns:f="http://xmlns.jcp.org/jsf/core">

    <body>

        <ui:composition template="./../WEB-INF/template.xhtml">

            <ui:define name="head">

            </ui:define>

            <ui:define name="content">
                <h:form id="frm">
                    <p:growl id="growl" showDetail="true" life="5000"/>
                    <p:panel id="pnlComisionesBroker">
                        <f:facet name="header">
                            Comisiones del Corredor
                        </f:facet>
                        <table width="100%">
                            <tr style="width: 50%; vertical-align: top">
                                <td>
                                    <p:fieldset legend="Aseguradoras">
                                        <p:dataTable id="dtAseguradoras"
                                                     var="aseg"

value="#{beanComisionBroker.listaAseguradoras}"
                                                     selectionMode="single"

selection="#{beanComisionBroker.aseguradoras}"
                                                     rowKey="#{aseg.cdAseguradora}"
                                                     scrollable="true"
                                                     scrollHeight="200">
                                            <p:ajax event="rowSelect"
listener="#{beanComisionBroker.selectAseguradora()}" update=":frm:dtRamos,
:frm:dtComision"/>
                                            <p:column headerText="Cod." width="1"
style="text-align: center">
                                                <p:outputLabel value="#{aseg.cdAseguradora}"/>
                                            </p:column>
                                            <p:column headerText="Aseguradoras" width="90">
                                                <p:outputLabel
value="#{aseg.desAseguradora}"/>
```

```
                        </p:column>
                    </p:dataTable>
                </p:fieldset>
            </td>
            <td>
                <p:fieldset legend="Ramos">
                    <p:dataTable id="dtRamos"
                                 var="ramo"
                                 value="#{beanComisionBroker.listaRamos}"
                                 selectionMode="single"
                                 selection="#{beanComisionBroker.ramos}"
                                 rowKey="#{ramo.cdRamo}"
                                 scrollable="true"
                                 scrollHeight="200">
                        <p:ajax event="rowSelect"
listener="#{beanComisionBroker.find()}" update=":frm:dtComision "/>
                        <p:column headerText="Cod." width="1"
style="text-align: center">
                            <p:outputLabel value="#{ramo.cdRamo}"/>
                        </p:column>
                        <p:column headerText="Ramo" width="90">
                            <p:outputLabel value="#{ramo.desRamo}"/>
                        </p:column>
                    </p:dataTable>
                </p:fieldset>
            </td>
        </tr>
        <tr style="width: 50%; vertical-align: top">
            <td colspan="2">
                <p:fieldset legend="Comisiones">
                    <p:commandButton id="btnAgregar"
                                     onclick="PF('dgRamoComision').show();"
                                     value="Agregar"
                                     disabled="#{(empty
beanComisionBroker.ramos)}"
                                     icon="fa fa-plus"
                                     styleClass="secondary"/>
                    <p:separator/>
                    <p:dataTable id="dtComision"

value="#{beanComisionBroker.listaComisiones}"
                                 var="comision"
                                 scrollable="true"
                                 scrollHeight="100"
                                 scrollWidth="max-content">

                        <p:column headerText="Pct. Com. Pol Nueva"
style="text-align: center">
                            <p:outputLabel value="#{comision.pctNuevo}">
```

```
                                    <f:convertNumber pattern="#####,00%"/>
                                </p:outputLabel>
                            </p:column>
                            <p:column headerText="Pct. Com. Renovacion"
style="text-align: center">
                                <p:outputLabel
value="#{comision.pctRenovacion}">
                                    <f:convertNumber pattern="#####,00%"/>
                                </p:outputLabel>
                            </p:column>

                            <p:column headerText="Usuario" style="text-align:
center">
                                <p:outputLabel value="#{comision.usuario}"/>
                            </p:column>
                            <p:column headerText="Observacion" width="400">
                                <p:outputLabel
value="#{comision.observacion}"/>
                            </p:column>

                            <p:column width="25" style="text-align: center" >
                                <p:commandButton id="btnEliminar"
                                                 icon="fa fa-trash"
                                                 styleClass="red-btn"

action="#{beanComisionBroker.eliminar(comision)}"
                                                 update=":frm:growl, :frm:dtComision">
                                    <p:confirm header="Advertencia"
message="Esta por desactivar una comision, desea continuar?" icon="pi pi-
exclamation-triangle" />
                                </p:commandButton>
                                <p:confirmDialog global="true"
showEffect="fade" hideEffect="fade">
                                    <p:commandButton value="Si" type="button"
styleClass="ui-confirmdialog-yes" icon="pi pi-check" />
                                    <p:commandButton value="No" type="button"
styleClass="ui-confirmdialog-no" icon="pi pi-times" />
                                </p:confirmDialog>

                            </p:column>
                        </p:dataTable>

                    </p:fieldset>
                </td>
            </tr>
        </table>
    </p:panel>
</h:form>
```

```
            <h:form id="frmDgRamoComision">
               <p:dialog id="dgRamoComision"
                         widgetVar="dgRamoComision"
                         modal="true"
                         showEffect="drop"
                         hideEffect="drop"
                         position="top"
                         style="margin-top: 75px"
                         draggable="false"
                         responsive="true"
                         header="Registrar Comision">
                  <p:ajax event="open" update="pnlRamosComision"/>
                  <p:panel id="pnlRamosComision">
                     <h:panelGrid columns="1" cellspacing="15">
                        <h:panelGroup styleClass="md-inputfield">
                           <p:inputText
value="#{beanComisionBroker.aseguradoras.desAseguradora}" readonly="true"
size="30"/>
                           <label>Aseguradora</label>
                        </h:panelGroup>
                        <h:panelGroup styleClass="md-inputfield">
                           <p:inputText
value="#{beanComisionBroker.ramos.desRamo}" readonly="true" size="30"/>
                           <label>Ramos</label>
                        </h:panelGroup>
                        <h:panelGroup styleClass="md-inputfield">
                           <p:inputNumber
value="#{beanComisionBroker.comisionBroker.pctNuevo}"  size="5"/>
                           <label>Comision Poliza Nueva</label>
                        </h:panelGroup>
                        <h:panelGroup styleClass="md-inputfield">
                           <p:inputNumber
value="#{beanComisionBroker.comisionBroker.pctRenovacion}" size="5" />
                           <label>Comision Renovacion</label>
                        </h:panelGroup>

                     </h:panelGrid>
                     <p:separator/>
                     <p:commandButton id="btnGuardar"
                                      value="Guardar"
                                      icon="fa fa-floppy-o"
                                      actionListener="#{beanComisionBroker.create()}"
                                      update=":frm:growl, :frm:dtComision"/>
                  </p:panel>
               </p:dialog>
            </h:form>
         </ui:define>
      </ui:composition>
   </body>
```

```
</html>
```

5.4.Vista FrmMarcasModelosVh

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
    xmlns:h="http://xmlns.jcp.org/jsf/html"
    xmlns:p="http://primefaces.org/ui"
    xmlns:f="http://xmlns.jcp.org/jsf/core">

  <body>

    <ui:composition template="./../WEB-INF/template.xhtml">
      <ui:define name="head">

      </ui:define>

      <ui:define name="content">
        <h:form id="frm">
          <p:growl id="growl" showDetail="true" life="5000"/>
          <p:panel>
            <f:facet name="header" >
              Marcas y Modelos de Vehiculos
            </f:facet>
            <table width="100%" >
              <tr>
                <td valign="top" width="50%">
                  <p:fieldset legend="Marcas">
                    <p:commandButton value="Agregar" icon="fa fa-plus"
onclick="PF('dgMarcas').show()"/>
                    <p:separator/>
                    <p:dataTable id="dtMarcas"
                                 value="#{beanMarcasModelos.listaMarcas}"
                                 var="marca"
                                 scrollable="true"
                                 scrollHeight="200"
                                 selectionMode="single"
                                 selection="#{beanMarcasModelos.marca}"
                                 rowKey="#{marca.cdMarca}">
                      <p:ajax event="rowSelect"
listener="#{beanMarcasModelos.find()}" update=":frm:dtModelos,
:frm:btnAgregarModelo" />
                      <p:column headerText="Cod." width="35">
                        <p:outputLabel value="#{marca.cdMarca}"/>
                      </p:column>
```

```
                        <p:column headerText="Marca"
filterBy="#{marca.desMarca}">
                            <p:outputLabel value="#{marca.desMarca}"/>
                        </p:column>
                    </p:dataTable>

                </p:fieldset>
            </td>
            <td valign="top" width="50%">
                <p:fieldset legend="Modelos">
                    <p:commandButton id="btnAgregarModelo"
                                     value="Agregar"
                                     icon="fa fa-plus"
                                     onclick="PF('dgModelos').show();"
                                     disabled="#{not
(beanMarcasModelos.marca.cdMarca != null)}"
                                     update=":frmDg:pnlModelo"/>
                    <p:separator/>
                    <p:dataTable id="dtModelos"
                                 value="#{beanMarcasModelos.listaModelos}"
                                 var="modelo"
                                 scrollable="true"
                                 scrollHeight="200">
                        <p:column headerText="Cod." width="35">
                            <p:outputLabel value="#{modelo.cdModelo}"/>
                        </p:column>
                        <p:column headerText="Modelo"
filterBy="#{modelo.desModelo}">
                            <p:outputLabel value="#{modelo.desModelo}"/>
                        </p:column>
                    </p:dataTable>

                </p:fieldset>
            </td>
        </tr>
    </table>
</p:panel>
</h:form>

<h:form id="frmDg">
    <p:dialog id="dgMarcas"
              widgetVar="dgMarcas"
              header="Registrar marca"
              hideEffect="drop"
              showEffect="drop"
              modal="true">
        <p:panel id="pnlMarcas">
            <h:panelGrid columns="1" cellspacing="20">
                <h:panelGroup class="md-inputfield">
```

```
                        <p:inputText
value="#{beanMarcasModelos.marcas.desMarca}"/>
                        <label>Marca</label>
                    </h:panelGroup>
                </h:panelGrid>
                <p:commandButton value="Grabar"
                        icon="fa fa-floppy-o"
                        actionListener="#{beanMarcasModelos.create()}"
                        update=":frm:dtMarcas, :frm:dtModelos, :frm:growl,
pnlMarcas"/>
            </p:panel>
        </p:dialog>


        <p:dialog id="dgModelos"
            widgetVar="dgModelos"
            header="Registrar Modelo"
            hideEffect="drop"
            showEffect="drop"
            modal="true">
          <p:panel id="pnlModelo">
            <h:panelGrid columns="1" cellspacing="20">
              <h:panelGroup class="md-inputfield">
                <p:inputText
value="#{beanMarcasModelos.marca.desMarca}" readonly="true"/>
                <label>Marca</label>
              </h:panelGroup>

              <h:panelGroup class="md-inputfield">
                <p:inputText
value="#{beanMarcasModelos.modelos.desModelo}"/>
                <label>Modelo</label>
              </h:panelGroup>
            </h:panelGrid>
            <p:commandButton value="Grabar"
                        icon="fa fa-floppy-o"

actionListener="#{beanMarcasModelos.createModelo()}"
                        update=":frm:dtModelos, :frm:growl, pnlModelo"/>
          </p:panel>
        </p:dialog>
      </h:form>
    </ui:define>
  </ui:composition>

</body>
</html>
```

5.5.Vista FrmEmision

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
    xmlns:h="http://xmlns.jcp.org/jsf/html"
    xmlns:p="http://primefaces.org/ui"
    xmlns:f="http://xmlns.jcp.org/jsf/core">

  <body>

    <ui:composition template="./../WEB-INF/template.xhtml">

      <ui:define name="head">

      </ui:define>

      <ui:define name="content">
        <h:form id="frm" target="_blank">
          <p:growl id="growl" autoUpdate="true" showDetail="true"
life="5000"/>
            <p:panel id="pnlCotizacion">
              <f:facet name="header">
                Emision Pólizas
              </f:facet>
              <p:toolbar>
                <f:facet name="left">
                  <h:panelGrid columns="3">
                    <p:commandButton icon="fa fa-save"
                                     value="Guardar"
                                     action="#{beanEmision.update()}"
                                     title="Guardar Cotizacion"
                                     update=":frm,:frmDg"/>
                    <p:commandButton icon="fa fa-print" value="Imprimir"
title="Imprimir Cotizacion"/>
                    <p:commandButton icon="fa fa-envelope-o" value="Enviar"
title="Enviar por correo"/>
                  </h:panelGrid>
                </f:facet>
                <f:facet name="right" >
                  <p:outputLabel
value="#{beanEmision.cotizacion.numCotizacion}" style="font-size: 20px"/>
                </f:facet>
              </p:toolbar>
```

```
                <p:fieldset style="margin: auto">
                  <h:panelGrid columns="2" style="border-spacing: 5px 20px;">
                    <h:panelGroup class="md-inputfield">
                      <p:inputText id="txtContratante"
                                   value="#{beanEmision.clientes.nombres}
#{beanEmision.clientes.apellidos}"
                                   size="40"
                                   readonly="true"/>
                      <label>Contratante</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                      <p:selectOneMenu id="somAseguradoras"
                                       value="#{beanEmision.cdAseguradora}"
                                       style="width: 200px"
                                       filter="true"
                                       autoWidth="false">

                        <f:selectItem itemLabel="Seleccionar..!" itemValue="0"/>
                        <f:selectItems value="#{beanEmision.listaAseguradoras}"
                                       var="aseg"
                                       itemLabel="#{aseg.desAseguradora}"
                                       itemValue="#{aseg.cdAseguradora}"/>
                      </p:selectOneMenu>
                      <label>Aseguradora</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                      <p:inputText value="#{beanEmision.subagentes.nombres}
#{beanEmision.subagentes.apellidos}"
                                   size="40"
                                   readonly="true"/>
                      <label>Sub Agente</label>
                    </h:panelGroup>

                    <h:panelGrid columns="2" >
                      <h:panelGroup class="md-inputfield">
                        <p:calendar id="calFcInicio"
                                    locale="es"
                                    mask="true"
                                    pattern="dd/MM/yyyy"
                                    autocomplete="true"
                                    navigator="true"
                                    size="15"
                                    showOn="button"
                                    value="#{beanEmision.cotizacion.fcInicio}">
                          <p:ajax event="dateSelect" update="calFcFin"
listener="#{beanEmision.finVigenciaCot()}"/>
                        </p:calendar>
```

```
                        <label>Fecha Desde</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:calendar id="calFcFin"
                                    locale="es"
                                    mask="true"
                                    pattern="dd/MM/yyyy"
                                    autocomplete="true"
                                    navigator="true"
                                    size="15"
                                    showOn="button"
                                    value="#{beanEmision.cotizacion.fcFin}"/>
                        <label>Fecha Hasta</label>
                    </h:panelGroup>
                </h:panelGrid>

            </h:panelGrid>

        </p:fieldset>
        <p:separator/>
        <p:tabView id="tvCotizacion">
            <p:tab title="Detalle Cotizacion">
                <h:panelGrid columns="5" style="border-spacing: 5px 15px">
                    <p:commandButton icon="fa fa-list-alt"
                                     value="Añadir"
                                     action="#{beanEmision.addDetalleCotizacion()}"
                                     update="dtDetalleCotizacion, :frm:growl"/>
                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu value="#{beanEmision.cdRamo}">
                            <p:ajax listener="#{beanEmision.selectRamo()}"
update="somProducto"/>
                            <f:selectItem itemLabel="Seleccionar..!"
itemValue="0"/>
                            <f:selectItems value="#{beanEmision.listaramos}"
                                           var="ramo"
                                           itemLabel="#{ramo.desRamo}"
                                           itemValue="#{ramo.cdRamo}"/>
                        </p:selectOneMenu>
                        <label>Ramo</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu id="somProducto"
value="#{beanEmision.cdProducto}">
                            <f:selectItem itemLabel="Seleccionar..!"
itemValue="0"/>
                            <f:selectItems value="#{beanEmision.listaProductos}"
                                           var="producto"
```

```
                                    itemLabel="#{producto.desProducto}"
                                    itemValue="#{producto.cdProducto}"/>
                        </p:selectOneMenu>
                        <label>Producto</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:calendar id="fcDesde"
                                locale="es"
                                mask="true"
                                pattern="dd/MM/yyyy"
                                autocomplete="true"
                                navigator="true"
                                size="15"
                                showOn="button"

value="#{beanEmision.detalleCotizacion.fcInicio}">
                                <p:ajax event="dateSelect"
listener="#{beanEmision.finVigencia()}" update="fcHasta"/>
                        </p:calendar>
                        <label>Fc. Desde</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:calendar id="fcHasta"
                                locale="es"
                                mask="true"
                                pattern="dd/MM/yyyy"
                                autocomplete="true"
                                navigator="true"
                                size="15"
                                showOn="button"
                                value="#{beanEmision.detalleCotizacion.fcFin}"/>
                        <label>Fc. Hasta</label>
                    </h:panelGroup>
                </h:panelGrid>
                <p:separator/>
                <p:dataTable id="dtDetalleCotizacion"
                        var="detalleCotizacion"

value="#{beanEmision.cotizacion.detalleCotizacionList}"
                        selection="#{beanEmision.detalleCotizacionSel}"
                        selectionMode="single"

rowKey="#{detalleCotizacion.cdCotizacion.cdCotizacion}">
                    <p:ajax event="rowSelect"
listener="#{beanEmision.consObjSeguro()}" update=":frmDg:pnlObjSeguro"/>
                    <p:column headerText="Ramo" style="text-align: center">
```

```
                        <p:outputLabel
value="#{detalleCotizacion.cdRamo.desRamo}"/>
                    </p:column>

                    <p:column headerText="Producto" style="text-align: center">
                        <p:outputLabel
value="#{detalleCotizacion.cdProducto.desProducto}"/>
                    </p:column>

                    <p:column headerText="Val. Asegurado" style="text-align:
center">
                        <p:outputLabel value="#{detalleCotizacion.totAseg}">
                            <f:convertNumber type="currency" currencySymbol="$"
/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Prima" style="text-align: center">
                        <p:outputLabel value="#{detalleCotizacion.totPrima}">
                            <f:convertNumber type="currency" currencySymbol="$"
/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Fc Desde" style="text-align: center">
                        <p:outputLabel value="#{detalleCotizacion.fcInicio}">
                            <f:convertDateTime pattern="dd/MM/yyyy"/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Fc Hasta" style="text-align: center">
                        <p:outputLabel value="#{detalleCotizacion.fcFin}">
                            <f:convertDateTime pattern="dd/MM/yyyy"/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Dias Seguro" style="text-align:
center">
                        <p:outputLabel value="#{detalleCotizacion.numDias}"/>
                    </p:column>

                    <p:column headerText="Póliza">
                        <p:inputText value="#{detalleCotizacion.numPoliza}"/>
                    </p:column>

                    <p:column headerText="Factura">
                        <p:inputText value="#{detalleCotizacion.factAseg}"/>
                    </p:column>
                </p:dataTable>
```

```
                    </p:tab>

                    <p:tab title="Financiamiento">
                      <p:commandButton icon="fa fa-dollar"
                                       value="Forma de Pago"
                                       action="#{beanEmision.addFormaPago()}"
                                       update=":frmDg:pnlFormaPago"/>

                      <p:commandButton icon="fa fa-trash"
                                       value="Borrar Forma de pago"
                                       update=":frmDg:pnlFormaPago"/>
                      <p:separator/>
                      <p:dataTable id="dtFinanciamiento"
                                   scrollable="true"
                                   scrollRows="4"
                                   scrollHeight="150"
                                   value="#{beanEmision.formaPago.financiamientoList}"
                                   var="financiamiento">
                        <p:column headerText="Ord." width="25" style="text-align:
center">
                          <p:outputLabel value="#{financiamiento.ornidal}"/>
                        </p:column>
                        <p:column headerText="Valor" style="text-align: center">
                          <p:outputLabel value="#{financiamiento.valor}">
                            <f:convertNumber maxFractionDigits="2"
currencySymbol="$"/>
                          </p:outputLabel>
                        </p:column>
                        <p:column headerText="Saldo" style="text-align: center">
                          <p:outputLabel value="#{financiamiento.saldo}">
                            <f:convertNumber maxFractionDigits="2"
currencySymbol="$"/>
                          </p:outputLabel>
                        </p:column>
                        <p:column headerText="Documento" style="text-align:
center">
                          <p:outputLabel value="#{financiamiento.factAseg}"/>
                        </p:column>
                        <p:column  style="text-align: center" >
                          <f:facet name="header">
                            <p:commandLink value="Fc. Vencimiento" style="color:
#fff" action="#{beanEmision.fcVencimiento()}" update="dtFinanciamiento"/>
                          </f:facet>
                          <p:calendar value="#{financiamiento.fcVencimiento}"
size="11"  mask="true" pattern="dd/MM/yyyy" showOn="button"/>
                        </p:column>
                        <p:column headerText="Letras" style="text-align: center">
                          <p:outputLabel value="#{financiamiento.letras}"/>
                        </p:column>
```

```
                    </p:dataTable>
                </p:tab>
            </p:tabView>
        </p:panel>
    </h:form>

    <h:form id="frmDg">
        <p:dialog id="dgFormaPago"
                  widgetVar="dgFormaPago"
                  modal="true"
                  responsive="true"
                  closeOnEscape="true"
                  showEffect="drop"
                  hideEffect="drop"
                  position="center"
                  style="margin-top: 25px">

            <p:panel id="pnlFormaPago">
                <h:panelGrid columns="2">
                    <h:panelGrid columns="2">
                        <p:outputLabel value="Forma Pago"/>
                        <p:selectOneMenu
value="#{beanEmision.formaPago.formaPago}"
                                         autoWidth="false"
                                         style="width: 200px">
                            <p:ajax listener="#{beanEmision.tipoPago()}"
update="txtNumPagos"/>
                            <f:selectItem itemLabel=" " itemValue=" "/>
                            <f:selectItem itemLabel="Contado"
itemValue="CONTADO"/>
                            <f:selectItem itemLabel="Credito"
itemValue="CREDITO"/>
                            <f:selectItem itemLabel="Debito Bancario"
itemValue="DEBITO"/>
                        </p:selectOneMenu>

                        <p:outputLabel value="Prima Neta"/>
                        <p:inputText value="#{beanEmision.formaPago.totPrima}"
readonly="true"/>

                        <p:outputLabel value="Der. Emision"/>
                        <p:inputNumber
value="#{beanEmision.formaPago.derEmision}" />

                        <p:outputLabel value="Sup. Bancos"/>
                        <p:inputNumber
value="#{beanEmision.formaPago.supBancos}" />

                        <p:outputLabel value="Seg. Campecino"/>
```

```
                        <p:inputNumber
value="#{beanEmision.formaPago.segCampecino}" />

                        <p:outputLabel value="Otro Valor"/>
                        <p:inputNumber
value="#{beanEmision.formaPago.otroValor}" />

                        <p:outputLabel value="Subtotal"/>
                        <p:inputText id="txtSubtotal"
value="#{beanEmision.formaPago.subtotal}" readonly="true" />

                        <h:panelGroup class="md-inputfield">
                           <p:selectOneMenu
value="#{beanEmision.formaPago.pctIva}">
                              <f:selectItem itemLabel="12%" itemValue="12"/>
                              <f:selectItem itemLabel="14%" itemValue="14"/>
                              <f:selectItem itemLabel="0%" itemValue="0"/>
                           </p:selectOneMenu>
                           <label>Iva</label>
                        </h:panelGroup>
                        <p:inputNumber id="txtValIva"
value="#{beanEmision.formaPago.valIva}" />

                        <p:outputLabel value="Prima Total"/>
                        <p:inputText id="txtTotPago"
value="#{beanEmision.formaPago.totPago}" readonly="true" />
                     </h:panelGrid>
                     <h:panelGrid columns="2">
                        <p:outputLabel value="Num. Pagos"/>
                        <p:inputText id="txtNumPagos"
value="#{beanEmision.formaPago.numPagos}" onkeypress="return
soloNumeros(event)"/>

                        <p:outputLabel value="% Cuota Inicial"/>
                        <p:inputText
value="#{beanEmision.formaPago.pctCuotaInicial}" onkeypress="return
soloNumeros(event)"/>

                        <p:outputLabel value="Comision Broker"/>
                        <p:selectOneMenu
value="#{beanEmision.formaPago.tipoComBrk}">
                           <f:selectItem itemLabel="Un Solo pago" itemValue="P"/>
                           <f:selectItem itemLabel="Cuotas" itemValue="C"/>
                        </p:selectOneMenu>

                        <p:outputLabel value="Comision Subagente"/>
                        <p:selectOneMenu
value="#{beanEmision.formaPago.tipoComSuba}">
                           <f:selectItem itemLabel="Un Solo pago" itemValue="P"/>
```

```
                        <f:selectItem itemLabel="Cuotas" itemValue="C"/>
                    </p:selectOneMenu>


                    <p:outputLabel value="Observacion"/>
                    <p:inputTextarea
value="#{beanEmision.formaPago.observaciones}"/>

                    <p:commandButton value="Procesar"

action="#{beanEmision.procesarFinanciamiento()}"
                                     update=":frm:tvCotizacion:dtFinanciamiento"/>
                </h:panelGrid>
            </h:panelGrid>
        </p:panel>

    </p:dialog>

    <p:dialog id="dgUbicacion"
              widgetVar="dgUbicacion"
              modal="true"
              responsive="true"
              closeOnEscape="true"
              showEffect="drop"
              hideEffect="drop"
              position="top"
              style="margin-top: 25px">
        <p:ajax event="close"
listener="#{beanEmision.totAseguradoUbicacion()}"
update=":frm:tvCotizacion:dtDetalleCotizacion"/>
        <h:panelGrid columns="2">
            <h:panelGroup class="md-inputfield">
                <p:inputText
value="#{beanEmision.ubicacion.desUbicacion}"/>
                <label>Ubicacion</label>
            </h:panelGroup>
            <p:commandButton icon="fa fa-plus"
                             style="border-radius: 5px"
                             action="#{beanEmision.addUbicacion()}"
                             update="dtUbicacion"/>
        </h:panelGrid>

        <p:separator/>
        <p:dataTable id="dtUbicacion"
                     value="#{beanEmision.detalleCotizacionSel.ubicacionList}"
                     var="ubicacion">
            <p:column headerText="Ubicacion">
                <p:outputLabel value="#{ubicacion.desUbicacion}"/>
            </p:column>
```

```
                    <p:column headerText="Fecha Desde">
                        <p:outputLabel value="#{ubicacion.fcInicio}"/>
                    </p:column>

                    <p:column headerText="Fecha Hasta">
                        <p:outputLabel value="#{ubicacion.fcFin}"/>
                    </p:column>

                    <p:column headerText="Tot. Asegurado">
                        <p:outputLabel value="#{ubicacion.totAsegurado}">
                            <f:convertNumber type="currency" currencySymbol="$" />
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Tot. Prima">
                        <p:outputLabel value="#{ubicacion.totPrima}">
                            <f:convertNumber type="currency" currencySymbol="$" />
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="">
                        <p:commandButton icon="fa fa-inbox"
                                         style="border-radius: 5px"
                                         title="Ver o Agregar Objetos asegurados"
                                         />
                    </p:column>

                    <p:column headerText="">
                        <p:commandButton icon="fa fa-trash" style="border-radius:
5px">
                            <p:confirm header="Confirmation" message="Desea eliminar
la ubicacion seleccionada y los objetos que contiene?" icon="fa fa-exclamation-
triangle" />
                        </p:commandButton>
                        <p:confirmDialog global="true" showEffect="fade"
hideEffect="fade">
                            <p:commandButton value="Si" type="button" styleClass="ui-
confirmdialog-yes" icon="pi pi-check" />
                            <p:commandButton value="No" type="button"
styleClass="ui-confirmdialog-no" icon="pi pi-times" />
                        </p:confirmDialog>
                    </p:column>
                </p:dataTable>
            </p:dialog>
            <!--
                ***********************************************

            REGISTRO OBJ SEGURO
```

```
                ***********************************************
            -->


            <p:dialog id="dgObjSeguro"
                    widgetVar="dgObjSeguro"
                    modal="true"
                    responsive="true"
                    closeOnEscape="true"
                    showEffect="drop"
                    hideEffect="drop"
                    position="top"
                    style="margin-top: 25px"
                    width="1100">
                <p:ajax event="close" listener="#{beanEmision.totAseguradoObj()}"
update=":frm:tvCotizacion:dtDetalleCotizacion, :frmDg:dtUbicacion"/>
                <p:panel id="pnlObjSeguro">
                  <h:panelGrid columns="1">
                    <p:fieldset legend="Item Asegurado">
                      <h:panelGrid columns="9">
                        <h:panelGroup class="md-inputfield">
                          <p:inputText
value="#{beanEmision.objSeguro.desObjeto}" size="30"/>
                          <label>Item Asegurado</label>
                        </h:panelGroup>

                        <h:panelGroup class="md-inputfield">
                          <p:inputNumber
value="#{beanEmision.objSeguro.valAseg}">
                            <p:ajax event="blur"
listener="#{beanEmision.calcPrima()}" update="txtPrimaSeg"/>
                          </p:inputNumber>
                          <label>Val. Asegurado</label>
                        </h:panelGroup>

                        <h:panelGroup class="md-inputfield">
                          <p:inputNumber
value="#{beanEmision.objSeguro.tasa}" size="6">
                            <p:ajax event="blur"
listener="#{beanEmision.calcPrima()}" update="txtPrimaSeg"/>
                          </p:inputNumber>
                          <label>Tasa</label>
                        </h:panelGroup>
                        <p:selectOneRadio id="sorFactor"
value="#{beanEmision.objSeguro.factor}">
                          <f:selectItem itemLabel="100" itemValue="100"/>
                          <f:selectItem itemLabel="1000" itemValue="1000"/>
                        </p:selectOneRadio>
```

```
                        <h:panelGroup class="md-inputfield">
                            <p:inputNumber id="txtPrimaSeg"
value="#{beanEmision.objSeguro.prima}"/>
                            <label>Prima</label>
                        </h:panelGroup>

                        <p:commandButton icon="fa fa-plus"
                                         title="Agregar Objeto"
                                         style="border-radius: 5px"
                                         action="#{beanEmision.addObjSeguro()}"
                                         update="dtObjSeguro,
:frm:tvCotizacion:dtDetalleCotizacion"/>
                    </h:panelGrid>
                </p:fieldset>
                <p:fieldset  legend="Vehiculo"
rendered="#{(beanEmision.detalleCotizacionSel.cdRamo.desRamo ==
'VEHICULOS')}">
                    <h:panelGrid columns="9">
                        <h:panelGroup class="md-inputfield">
                            <p:selectOneMenu id="somMarcas"
value="#{beanEmision.objSeguro.marca}"
                                             autoWidth="false"
                                             filter="true"
                                             style="width: 150px">
                                <p:ajax listener="#{beanEmision.selectMarca()}"
update="somModelo"/>
                                <f:selectItem itemLabel="Seleccionar"
itemValue="0"/>
                                <f:selectItems value="#{beanEmision.listaMarcas}"
                                               var="marca"
                                               itemLabel="#{marca.desMarca}"
                                               itemValue="#{marca.desMarca}"/>
                            </p:selectOneMenu>
                            <label>Marca</label>
                        </h:panelGroup>

                        <h:panelGroup class="md-inputfield">
                            <p:selectOneMenu id="somModelo"
value="#{beanEmision.objSeguro.modelo}"
                                             autoWidth="false"
                                             filter="true"
                                             style="width: 150px">
                                <f:selectItem itemLabel="Seleccionar"
itemValue="0"/>
                                <f:selectItems
value="#{beanEmision.marcas.modelosList}"
                                               var="modelo"
                                               itemLabel="#{modelo.desModelo}"
                                               itemValue="#{modelo.desModelo}"/>
```

```
                        </p:selectOneMenu>
                        <label>Modelo</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu id="somColor"
value="#{beanEmision.objSeguro.color}"
                                         autoWidth="false"
                                         style="width: 100px"
                                         filter="true">
                            <f:selectItem itemLabel="Seleccionar"
itemValue="0"/>
                            <f:selectItems value="#{beanEmision.listaColores}"
                                           var="color"
                                           itemLabel="#{color.desColor}"
                                           itemValue="#{color.desColor}"/>
                        </p:selectOneMenu>
                        <label>Color</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:inputText
value="#{beanEmision.objSeguro.chasis}"/>
                        <label>Chasis</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:inputText
value="#{beanEmision.objSeguro.motor}"/>
                        <label>Motor</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:inputText value="#{beanEmision.objSeguro.placa}"
size="10"/>
                        <label>Placa</label>
                    </h:panelGroup>

                    <h:panelGroup class="md-inputfield">
                        <p:selectOneMenu id="somTipoVh"

value="#{beanEmision.objSeguro.tipoVehiculo}">
                            <f:selectItem itemLabel="Particular"
itemValue="PARTICULAR"/>
                            <f:selectItem itemLabel="Comercial"
itemValue="COMERCIAL"/>
                        </p:selectOneMenu>
                        <label>Tipo Vehiculo</label>
                    </h:panelGroup>
```

```
                        <h:panelGroup class="md-inputfield">
                            <p:inputText value="#{beanEmision.objSeguro.anio}"
size="6" onkeypress="return soloNumeros(event)"/>
                            <label>Año</label>
                        </h:panelGroup>
                    </h:panelGrid>
                </p:fieldset>
            </h:panelGrid>

            <!--
            ********************************************

            TABLA OBJ SEGURO

            ********************************************
            -->
            <p:dataTable id="dtObjSeguro"

value="#{beanEmision.detalleCotizacionSel.objSeguroList}"
                         var="objSeguro"
                         scrollable="true"
                         scrollWidth="max-content"
                         selectionMode="single"
                         selection="#{beanEmision.objSeguroSel}"

rowKey="#{objSeguro.cdDetCotizacion.cdCotizacion.cdCotizacion}">
                <p:column headerText="Item Asegurado" width="200"
style="text-align: center">
                    <p:outputLabel value="#{objSeguro.desObjeto}"/>
                </p:column>
                <p:column headerText="Val. Asegurado" width="60"
style="text-align: center">
                    <p:outputLabel value="#{objSeguro.valAseg}">
                        <f:convertNumber type="currency" currencySymbol="$" />
                    </p:outputLabel>
                </p:column>
                <p:column headerText="Tasa" width="30" style="text-align:
center">
                    <p:outputLabel value="#{objSeguro.tasa}%"/>
                </p:column>

                <p:column headerText="Factor" width="30" style="text-align:
center">
                    <p:outputLabel value="#{objSeguro.factor}"/>
                </p:column>

                <p:column headerText="Prima" width="60" style="text-align:
center">
```

```
                        <p:outputLabel value="#{objSeguro.prima}">
                            <f:convertNumber type="currency" currencySymbol="$" />
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Fc. Desde" width="60" style="text-
align: center">
                        <p:outputLabel value="#{objSeguro.fcInicio}">
                            <f:convertDateTime pattern="dd/MM/yyyy"/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Fc. Hasta" width="60" style="text-
align: center">
                        <p:outputLabel value="#{objSeguro.fcFin}">
                            <f:convertDateTime pattern="dd/MM/yyyy"/>
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Tot. Asegurado" width="60"
style="text-align: center">
                        <p:outputLabel value="#{objSeguro.totAseg}">
                            <f:convertNumber type="currency" currencySymbol="$" />
                        </p:outputLabel>
                    </p:column>

                    <p:column headerText="Tot. Prima" width="60" style="text-
align: center">
                        <p:outputLabel value="#{objSeguro.totPrima}">
                            <f:convertNumber type="currency" currencySymbol="$" />
                        </p:outputLabel>
                    </p:column>

                    <p:column width="35" style="text-align: center"
rendered="#{(beanEmision.detalleCotizacionSel.cdRamo.desRamo ==
'VEHICULOS')}">
                        <p:commandButton icon="fa fa-search"
                                         title="Caracteristicas"
                                         style="border-radius: 5px"
                                         update=":frmDg:pnlVehiculo"

action="#{beanEmision.carVehiculos(objSeguro)}"/>
                    </p:column>

                    <p:column width="35" style="text-align: center">

                        <p:commandButton icon="fa fa-trash" title="Eliminar"
style="border-radius: 5px" styleClass="red-btn">
```

```
                        <p:confirm header="Confirmation" message="Desea
eliminar el Objeto asegurado?" icon="fa fa-exclamation-triangle" />
                    </p:commandButton>
                    <p:confirmDialog global="true" showEffect="fade"
hideEffect="fade">
                        <p:commandButton value="Si" type="button"
styleClass="ui-confirmdialog-yes" icon="pi pi-check" />
                        <p:commandButton value="No" type="button"
styleClass="ui-confirmdialog-no" icon="pi pi-times" />
                    </p:confirmDialog>
                </p:column>
            </p:dataTable>
        </p:panel>
    </p:dialog>

    <p:dialog id="dgCarVh"
            widgetVar="dgCarVh"
            modal="true"
            responsive="true"
            closeOnEscape="true"
            showEffect="drop"
            hideEffect="drop">
        <p:panel id="pnlVehiculo">
            <h:panelGrid columns="1" style="border-spacing: 5px 20px">
                <h:panelGroup class="md-inputfield">
                    <p:inputText value="#{beanEmision.vehiculo.marca}"/>
                    <label>Marca</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText value="#{beanEmision.vehiculo.modelo}"/>
                    <label>Modelo</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText value="#{beanEmision.vehiculo.color}"/>
                    <label>Color</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText value="#{beanEmision.vehiculo.marca}"/>
                    <label>Placa</label>
                </h:panelGroup>

                <h:panelGroup class="md-inputfield">
                    <p:inputText value="#{beanEmision.vehiculo.chasis}"/>
                    <label>Chasis</label>
                </h:panelGroup>
```

```
                        <h:panelGroup class="md-inputfield">
                           <p:inputText value="#{beanEmision.vehiculo.motor}"/>
                           <label>Motor</label>
                        </h:panelGroup>

                        <h:panelGroup class="md-inputfield">
                           <p:inputText
value="#{beanEmision.vehiculo.tipoVehiculo}"/>
                           <label>Tipo Vehiculo</label>
                        </h:panelGroup>

                        <h:panelGroup class="md-inputfield">
                           <p:inputText value="#{beanEmision.vehiculo.anio}"/>
                           <label>Año</label>
                        </h:panelGroup>
                     </h:panelGrid>
                  </p:panel>
               </p:dialog>


               <p:dialog id="dgSubobjeto"
                     widgetVar="dgSubobjeto"
                     modal="true"
                     responsive="true"
                     closeOnEscape="true"
                     showEffect="drop"
                     hideEffect="drop">
                  <p:ajax event="close"
listener="#{beanEmision.totAsegSubobjeto()}" update=":frmDg:dtObjSeguro"/>
                  <p:panel id="pnlSubobjetos">

                  </p:panel>

               </p:dialog>
            </h:form>
         </ui:define>
      </ui:composition>
   </body>
</html
```

1.4 Base de datos.

```
CREATE TABLE QLT_T_ASEGURADORAS
(
 CD_ASEGURADORA   INTEGER                NOT NULL,
 DES_ASEGURADORA  VARCHAR2(250 BYTE)     NOT NULL,
 RUC              VARCHAR2(15 BYTE)      NOT NULL,
 ALIAS            VARCHAR2(15 BYTE)      NOT NULL,
 FC_REGISTRO      DATE                   DEFAULT trunc(sysdate)    NOT NULL,
 FC_CONVENIO      DATE,
 ESTADO           CHAR(1 BYTE)           DEFAULT 'A'               NOT NULL,
 CONSTRAINT PK_QLT_T_ASEGURADORAS
```

```
 PRIMARY KEY
 (CD_ASEGURADORA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_CLAVES_ACTIVACION
(
 CD_CLAVE         INTEGER                NOT NULL,
 USUARIO          VARCHAR2(15 BYTE)      NOT NULL,
 FC_REGISTRO      DATE                   DEFAULT trunc(sysdate)      NOT NULL,
 CLAVE_ACTIVACION  VARCHAR2(100 BYTE)    NOT NULL,
 ESTADO           CHAR(1 BYTE)           DEFAULT 'A',
 CONSTRAINT PK_CLAVES_ACTIVACION
 PRIMARY KEY
 (CD_CLAVE)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
```

```
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_CLIENTES
(
 CD_CLIENTE     INTEGER                    NOT NULL,
 NOMBRES        VARCHAR2(100 BYTE)         NOT NULL,
 APELLIDOS      VARCHAR2(100 BYTE),
 GENERO         CHAR(1 BYTE),
 TIPO_DNI       CHAR(1 BYTE)               NOT NULL,
 DNI            VARCHAR2(15 BYTE)          NOT NULL,
 TIPO_CLI       CHAR(1 BYTE),
 TITULO         VARCHAR2(5 BYTE),
 FC_NACIMIENTO  DATE,
 TIPO           VARCHAR2(100 BYTE)         DEFAULT 'PROSPECTO',
 ESTADO         CHAR(1 BYTE)               DEFAULT 'A',
 CONSTRAINT PK_QLT_T_CLIENTES
 PRIMARY KEY
 (CD_CLIENTE)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
 ENABLE VALIDATE
)
```

```
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_COLORES_VH
(
 CD_COLOR   INTEGER                      NOT NULL,
 DES_COLOR  VARCHAR2(50 BYTE)            NOT NULL,
 ESTADO     CHAR(1 BYTE)                 DEFAULT 'A'               NOT NULL,
 CONSTRAINT PK_QLT_T_COLORES_VH
 PRIMARY KEY
 (CD_COLOR)
 USING INDEX
   TABLESPACE USERS
   PCTFREE    10
   INITRANS   2
   MAXTRANS   255
   STORAGE    (
               INITIAL          64K
               NEXT             1M
               MAXSIZE          UNLIMITED
               MINEXTENTS       1
               MAXEXTENTS       UNLIMITED
               PCTINCREASE      0
               BUFFER_POOL      DEFAULT
               FLASH_CACHE      DEFAULT
               CELL_FLASH_CACHE DEFAULT
              )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
```

```
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_CONTACTOS
(
 CD_CONTACTO     INTEGER                NOT NULL,
 CD_ASEGURADORA  INTEGER,
 CD_CLIENTE      INTEGER,
 NOMBRES         VARCHAR2(100 BYTE)     NOT NULL,
 APELLIDOS       VARCHAR2(100 BYTE)     NOT NULL,
 CARGO           VARCHAR2(100 BYTE)     NOT NULL,
 TITULO          VARCHAR2(5 BYTE),
 FC_NACIMIENTO   DATE,
 ESTADO          CHAR(1 BYTE)           DEFAULT 'A'            NOT NULL,
 CONSTRAINT PK_QLT_T_CONTACTOS
 PRIMARY KEY
 (CD_CONTACTO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL        64K
              NEXT           1M
              MAXSIZE        UNLIMITED
              MINEXTENTS     1
              MAXEXTENTS     UNLIMITED
              PCTINCREASE    0
              BUFFER_POOL    DEFAULT
              FLASH_CACHE    DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_ASEGURADORAS_CONTACTOS
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_CLIENTES_CONTACTOS
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
```

```
        )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_COTIZACION
(
 CD_COTIZACION   INTEGER                NOT NULL,
 CD_CLIENTE      INTEGER                NOT NULL,
 CD_ASEGURADORA  INTEGER,
 FC_INICIO       DATE,
 FC_FIN          DATE,
 FC_REGISTRO     DATE                   DEFAULT TRUNC(SYSDATE),
 NUM_COTIZACION  VARCHAR2(25 BYTE),
 TIPO_POL        CHAR(1 BYTE),
 ESTADO          CHAR(1 BYTE)           DEFAULT 'C',
 CONSTRAINT PK_QLT_T_COTIZACION
 PRIMARY KEY
 (CD_COTIZACION)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL          64K
             NEXT             1M
             MAXSIZE          UNLIMITED
             MINEXTENTS       1
             MAXEXTENTS       UNLIMITED
             PCTINCREASE      0
             BUFFER_POOL      DEFAULT
             FLASH_CACHE      DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_ASEGURADORAS_COTIZACION
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_CLIENTES_COTIZACION
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL          64K
           NEXT             1M
           MAXSIZE          UNLIMITED
           MINEXTENTS       1
           MAXEXTENTS       UNLIMITED
           PCTINCREASE      0
           BUFFER_POOL      DEFAULT
           FLASH_CACHE      DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
```

```
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_DATOS_CORREDOR
(
 CD_CORREDOR      INTEGER                NOT NULL,
 NOMBRE_COMERCIAL  VARCHAR2(100 BYTE),
 RAZON_SOCIAL     VARCHAR2(150 BYTE),
 RUC              VARCHAR2(15 BYTE),
 MATRIZ           INTEGER,
 ESTADO           CHAR(1 BYTE)           DEFAULT 'A',
 CONSTRAINT QLT_T_DATOS_CORREDOR_PK
 PRIMARY KEY
 (CD_CORREDOR)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL        64K
              NEXT           1M
              MAXSIZE        UNLIMITED
              MINEXTENTS     1
              MAXEXTENTS     UNLIMITED
              PCTINCREASE    0
              BUFFER_POOL    DEFAULT
              FLASH_CACHE    DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_MATRIZ
 FOREIGN KEY (MATRIZ)
 REFERENCES QLT_T_DATOS_CORREDOR (CD_CORREDOR)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_DERECHOS_EMISION
```

```
(
 CD_DER_EMISION  INTEGER                 NOT NULL,
 ORDEN          NUMBER,
 PRIMA_INI      NUMBER(8,2),
 PRIMA_FIN      NUMBER,
 VALOR          NUMBER(8,2),
 ESTADO         CHAR(1 BYTE)             DEFAULT 'A',
 CONSTRAINT PK_QLT_T_DERECHOS_EMISION
 PRIMARY KEY
 (CD_DER_EMISION)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
           PCTINCREASE     0
           BUFFER_POOL     DEFAULT
           FLASH_CACHE     DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_EMAIL
(
 CD_EMAIL     INTEGER                    NOT NULL,
 CD_CONTACTO  INTEGER,
 CD_CLIENTE   INTEGER,
 DES_EMAIL    VARCHAR2(50 BYTE)          NOT NULL,
 ESTADO       CHAR(1 BYTE)               DEFAULT 'A',
 CONSTRAINT PK_QLT_T_EMAIL
 PRIMARY KEY
 (CD_EMAIL)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
```

```
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_EM_CLIENTES__QLT_T_CL
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_EM_CONTACTOS_QLT_T_CO
 FOREIGN KEY (CD_CONTACTO)
 REFERENCES QLT_T_CONTACTOS (CD_CONTACTO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_FACTURAS
(
 CD_FACTURAS   INTEGER                  NOT NULL,
 NUM_FACTURA   VARCHAR2(15 BYTE),
 FC_FACTURA    DATE,
 VAL_FACTURA   NUMBER(8,2),
 IVA           NUMBER(8,2),
 TOT_FACTURA   NUMBER(8,2),
 FLG_PAGO      CHAR(1 BYTE),
 FC_PAGO       DATE,
 FC_ANULACION  DATE,
 ESTADO        CHAR(1 BYTE)             DEFAULT 'E',
 CONSTRAINT PK_QLT_T_FACTURAS
 PRIMARY KEY
 (CD_FACTURAS)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
```

```
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_FORMA_PAGO
(
 CD_FORMA_PAGO      INTEGER                NOT NULL,
 CD_COTIZACION      INTEGER,
 FORMA_PAGO         VARCHAR2(50 BYTE),
 DER_EMISION        NUMBER(8,2),
 SUP_BANCOS         NUMBER(8,2),
 SEG_CAMPECINO      NUMBER(8,2),
 NUM_PAGOS          NUMBER,
 TOT_PRIMA          NUMBER(8,2),
 TOT_PAGO           NUMBER(8,2),
 OBSERVACIONES      VARCHAR2(500 BYTE),
 OTRO_VALOR         NUMBER(8,2),
 PCT_CUOTA_INICIAL  NUMBER(8,2),
 TIPO_COM_BRK       CHAR(1 BYTE),
 TIPO_COM_SUBA      CHAR(1 BYTE),
 SUBTOTAL           NUMBER(8,2),
 PCT_IVA            INTEGER,
 VAL_IVA            NUMBER(8,2),
 CONSTRAINT PK_QLT_T_FORMA_PAGO
 PRIMARY KEY
 (CD_FORMA_PAGO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
```

```
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_FO_COTIZACIO_QLT_T_CO
 FOREIGN KEY (CD_COTIZACION)
 REFERENCES QLT_T_COTIZACION (CD_COTIZACION)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
           PCTINCREASE     0
           BUFFER_POOL     DEFAULT
           FLASH_CACHE     DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_MARCAS
(
 CD_MARCA   INTEGER                     NOT NULL,
 DES_MARCA  VARCHAR2(100 BYTE)              NOT NULL,
 ESTADO     CHAR(1 BYTE)                DEFAULT 'A',
 CONSTRAINT PK_QLT_T_MARCAS
 PRIMARY KEY
 (CD_MARCA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
```

```
          )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_MENUS
(
 CD_MENU   INTEGER                      NOT NULL,
 SUBMENU   INTEGER,
 DES_MENU  VARCHAR2(50 BYTE)            NOT NULL,
 ORDEN     INTEGER                      NOT NULL,
 URL       VARCHAR2(100 BYTE),
 NIVEL     INTEGER                      NOT NULL,
 ESTADO    CHAR(1 BYTE)                 DEFAULT 'A',
 ICONO     VARCHAR2(50 BYTE),
 CONSTRAINT PK_QLT_T_MENUS
 PRIMARY KEY
 (CD_MENU)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_ME_MENU_SUBM_QLT_T_ME
 FOREIGN KEY (SUBMENU)
 REFERENCES QLT_T_MENUS (CD_MENU)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
```

```
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_MODELOS
(
 CD_MODELO   INTEGER                      NOT NULL,
 CD_MARCA    INTEGER,
 DES_MODELO  VARCHAR2(250 BYTE)           NOT NULL,
 ESTADO      CHAR(1 BYTE)                 DEFAULT 'A',
 CONSTRAINT PK_QLT_T_MODELOS
 PRIMARY KEY
 (CD_MODELO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL          64K
              NEXT             1M
              MAXSIZE          UNLIMITED
              MINEXTENTS       1
              MAXEXTENTS       UNLIMITED
              PCTINCREASE      0
              BUFFER_POOL      DEFAULT
              FLASH_CACHE      DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_MO_MARCA_MOD_QLT_T_MA
 FOREIGN KEY (CD_MARCA)
 REFERENCES QLT_T_MARCAS (CD_MARCA)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
```

```
        BUFFER_POOL      DEFAULT
        FLASH_CACHE      DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_PERFIL
(
 CD_PERFIL   INTEGER                    NOT NULL,
 DES_PERFIL  VARCHAR2(50 BYTE)              NOT NULL,
 ESTADO      CHAR(1 BYTE)                DEFAULT 'A',
 CONSTRAINT PK_QLT_T_PERFIL
 PRIMARY KEY
 (CD_PERFIL)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_PRE_FACTURAS
(
 CD_PREFACTURA    INTEGER                NOT NULL,
 FC_REGISTRO      DATE,
```

```
 VAL_PRE_FACTURA  NUMBER(8,2),
 IVA            NUMBER,
 VAL_IVA        NUMBER(8,2),
 TOT_PRE_FACTURA  NUMBER(8,2),
 CONSTRAINT PK_QLT_T_PRE_FACTURAS
 PRIMARY KEY
 (CD_PREFACTURA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_PROVINCIAS
(
 CD_PROVINCIA   INTEGER                  NOT NULL,
 DES_PROVINCIA  VARCHAR2(100 BYTE),
 ESTADO         CHAR(1 BYTE)             DEFAULT 'A',
 CONSTRAINT PK_QLT_T_PROVINCIAS
 PRIMARY KEY
 (CD_PROVINCIA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
```

```
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_RAMOS
(
 CD_RAMO   INTEGER                      NOT NULL,
 DES_RAMO  VARCHAR2(100 BYTE)               NOT NULL,
 ALIAS     VARCHAR2(15 BYTE)              NOT NULL,
 NO_IVA    INTEGER,
 ESTADO    CHAR(1 BYTE)                 DEFAULT 'A',
 CONSTRAINT PK_QLT_T_RAMOS
 PRIMARY KEY
 (CD_RAMO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
```

```
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_SECUENCIAS
(
 CD_SECUENCIA   INTEGER                    NOT NULL,
 ANIO           NUMBER,
 TP_SECUENCIA   VARCHAR2(50 BYTE),
 VAL_SECUENCIA  NUMBER,
 CONSTRAINT PK_QLT_T_SECUENCIAS
 PRIMARY KEY
 (CD_SECUENCIA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL          64K
              NEXT             1M
              MAXSIZE          UNLIMITED
              MINEXTENTS       1
              MAXEXTENTS       UNLIMITED
              PCTINCREASE      0
              BUFFER_POOL      DEFAULT
              FLASH_CACHE      DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
```

```
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_SUBAGENTES
(
 CD_SUBAGENTE    INTEGER                 NOT NULL,
 NOMBRES         VARCHAR2(100 BYTE)          NOT NULL,
 APELLIDOS       VARCHAR2(100 BYTE)          NOT NULL,
 IDENTIFICACION  VARCHAR2(15 BYTE),
 EMAIL           VARCHAR2(50 BYTE)           NOT NULL,
 ESTADO          CHAR(1 BYTE)              DEFAULT 'A',
 CONSTRAINT PK_QLT_T_SUBAGENTES
 PRIMARY KEY
 (CD_SUBAGENTE)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL        64K
              NEXT           1M
              MAXSIZE        UNLIMITED
              MINEXTENTS     1
              MAXEXTENTS     UNLIMITED
              PCTINCREASE    0
              BUFFER_POOL    DEFAULT
              FLASH_CACHE    DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_TELEFONOS
(
 CD_TELEFONO     INTEGER                 NOT NULL,
 CD_CLIENTE      INTEGER,
 CD_ASEGURADORA  INTEGER,
```

```
 CD_CONTACTO     INTEGER,
 TIPO_TELEFONO   CHAR(1 BYTE),
 NUM_TELEFONO    VARCHAR2(15 BYTE)           NOT NULL,
 EXTENSION       VARCHAR2(50 BYTE),
 ESTADO          CHAR(1 BYTE)                DEFAULT 'A',
 CONSTRAINT PK_QLT_T_TELEFONOS
 PRIMARY KEY
 (CD_TELEFONO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_ASEGURADORA_TELEFONOS
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_TE_CLIENTE_T_QLT_T_CL
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_TE_CONTACTOS_QLT_T_CO
 FOREIGN KEY (CD_CONTACTO)
 REFERENCES QLT_T_CONTACTOS (CD_CONTACTO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
           PCTINCREASE     0
           BUFFER_POOL     DEFAULT
           FLASH_CACHE     DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_TITULOS
(
 CD_TITULO  INTEGER,
```

```
 TITULO     VARCHAR2(5 BYTE)               NOT NULL,
 ESTADO     CHAR(1 BYTE)                   DEFAULT 'A',
 CONSTRAINT QLT_T_TITULOS_PK
 PRIMARY KEY
 (CD_TITULO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_USUARIOS
(
 CD_USUARIO       INTEGER                NOT NULL,
 CD_PERFIL        INTEGER,
 NOMBRES          VARCHAR2(100 BYTE)        NOT NULL,
 APELLIDOS        VARCHAR2(100 BYTE),
 EMAIL            VARCHAR2(50 BYTE)         NOT NULL,
 CARGO            VARCHAR2(100 BYTE)        NOT NULL,
 ESTADO           CHAR(1 BYTE)              DEFAULT 'A'               NOT NULL,
 CLAVE_ACTIVACION  VARCHAR2(100 BYTE)       NOT NULL,
 USUARIO          VARCHAR2(12 BYTE)         NOT NULL,
 CLAVE            VARCHAR2(100 BYTE)        NOT NULL,
 CONSTRAINT PK_QLT_T_USUARIOS
 PRIMARY KEY
 (CD_USUARIO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
```

```
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL         64K
              NEXT            1M
              MAXSIZE         UNLIMITED
              MINEXTENTS      1
              MAXEXTENTS      UNLIMITED
              PCTINCREASE     0
              BUFFER_POOL     DEFAULT
              FLASH_CACHE     DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_USUARIO_PERFIL
 FOREIGN KEY (CD_PERFIL)
 REFERENCES QLT_T_PERFIL (CD_PERFIL)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE SEQUENCE
(
 SEQ_NAME   VARCHAR2(50 BYTE)              NOT NULL,
 SEQ_COUNT  NUMBER(38),
 PRIMARY KEY
 (SEQ_NAME)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL         64K
              NEXT            1M
              MAXSIZE         UNLIMITED
              MINEXTENTS      1
              MAXEXTENTS      UNLIMITED
              PCTINCREASE     0
              BUFFER_POOL     DEFAULT
              FLASH_CACHE     DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
```

```
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE INDEX ASEGURADORAS_CONTACTOS_FK ON QLT_T_CONTACTOS
(CD_ASEGURADORA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX ASEGURADORAS_COTIZACION_FK ON QLT_T_COTIZACION
(CD_ASEGURADORA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
```

```
NOPARALLEL;


CREATE INDEX ASEGURADORA_TELEFONOS_FK ON QLT_T_TELEFONOS
(CD_ASEGURADORA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX CLIENTES_CONTACTOS_FK ON QLT_T_CONTACTOS
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX CLIENTES_COTIZACION_FK ON QLT_T_COTIZACION
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;
```

```
CREATE INDEX CLIENTES_EMAIL_FK ON QLT_T_EMAIL
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX CLIENTE_TELEFONOS_FK ON QLT_T_TELEFONOS
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX CONTACTOS_EMAIL_FK ON QLT_T_EMAIL
(CD_CONTACTO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX CONTACTOS_TELEFONO_FK ON QLT_T_TELEFONOS
```

```
(CD_CONTACTO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
        )
NOPARALLEL;


CREATE INDEX COTIZACION_FORMA_PAGO_FK ON QLT_T_FORMA_PAGO
(CD_COTIZACION)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
        )
NOPARALLEL;


CREATE INDEX MARCA_MODELOS_FK ON QLT_T_MODELOS
(CD_MARCA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
        )
NOPARALLEL;


CREATE INDEX MENU_SUBMENU_FK ON QLT_T_MENUS
(SUBMENU)
LOGGING
```

```
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX PERFIL_USUARIOS_FK ON QLT_T_USUARIOS
(CD_PERFIL)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;

CREATE TABLE QLT_T_ACCESOS
(
 CD_PERFIL  INTEGER                    NOT NULL,
 CD_MENU    INTEGER                    NOT NULL,
 CONSTRAINT PK_QLT_T_ACCESOS
 PRIMARY KEY
 (CD_PERFIL, CD_MENU)
 USING INDEX
   TABLESPACE USERS
   PCTFREE    10
   INITRANS   2
   MAXTRANS   255
   STORAGE    (
               INITIAL         64K
               NEXT            1M
               MAXSIZE         UNLIMITED
               MINEXTENTS      1
               MAXEXTENTS      UNLIMITED
               PCTINCREASE     0
               BUFFER_POOL     DEFAULT
               FLASH_CACHE     DEFAULT
               CELL_FLASH_CACHE DEFAULT
              )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_AC_RELATIONS_QLT_T_ME
 FOREIGN KEY (CD_MENU)
 REFERENCES QLT_T_MENUS (CD_MENU)
```

```
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_AC_RELATIONS_QLT_T_PE
 FOREIGN KEY (CD_PERFIL)
 REFERENCES QLT_T_PERFIL (CD_PERFIL)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_CIUDADES
(
 CD_CIUDAD     INTEGER                     NOT NULL,
 CD_PROVINCIA  INTEGER,
 DES_CIUDAD    VARCHAR2(100 BYTE),
 ESTADO        CHAR(1 BYTE)                DEFAULT 'A',
 CONSTRAINT PK_QLT_T_CIUDADES
 PRIMARY KEY
 (CD_CIUDAD)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL         64K
              NEXT            1M
              MAXSIZE         UNLIMITED
              MINEXTENTS      1
              MAXEXTENTS      UNLIMITED
              PCTINCREASE     0
              BUFFER_POOL     DEFAULT
              FLASH_CACHE     DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_CI_PROVINCIA_QLT_T_PR
 FOREIGN KEY (CD_PROVINCIA)
 REFERENCES QLT_T_PROVINCIAS (CD_PROVINCIA)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
```

```
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
           PCTINCREASE     0
           BUFFER_POOL     DEFAULT
           FLASH_CACHE     DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_CLI_SUBA
(
 CD_CLIENTE    INTEGER                     NOT NULL,
 CD_SUBAGENTE  INTEGER                       NOT NULL,
 CONSTRAINT PK_QLT_T_CLI_SUBA
 PRIMARY KEY
 (CD_CLIENTE, CD_SUBAGENTE)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_CLI_SUBA_CLIENTES
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_CL_SUBAGENTE_QLT_T_SU
 FOREIGN KEY (CD_SUBAGENTE)
 REFERENCES QLT_T_SUBAGENTES (CD_SUBAGENTE)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
```

```
        PCTINCREASE      0
        BUFFER_POOL      DEFAULT
        FLASH_CACHE      DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_COMISION_BROKER
(
 CD_COM_BRK      INTEGER                NOT NULL,
 CD_RAMO         INTEGER,
 CD_ASEGURADORA  INTEGER,
 PCT_NUEVO       NUMBER(8,2)            NOT NULL,
 PCT_RENOVACION  NUMBER(8,2)            NOT NULL,
 FC_REGISTRO     DATE                   DEFAULT TRUNC(SYSDATE)       NOT NULL,
 ESTADO          CHAR(1 BYTE)           DEFAULT 'A'                  NOT NULL,
 USUARIO         VARCHAR2(15 BYTE),
 OBSERVACION     VARCHAR2(1000 BYTE),
 CONSTRAINT PK_QLT_T_COMISON_BROKER
 PRIMARY KEY
 (CD_COM_BRK)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE,
 CONSTRAINT FK_ASEG_COM_BRK
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_CO_RAMOS_COM_QLT_T_RA
 FOREIGN KEY (CD_RAMO)
 REFERENCES QLT_T_RAMOS (CD_RAMO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
```

```
        BUFFER_POOL      DEFAULT
        FLASH_CACHE      DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_COMISION_SUBAGENTES
(
 CD_COM_SUBA     INTEGER                  NOT NULL,
 CD_SUBAGENTE    INTEGER,
 PCT_NUEVO       NUMBER(8,2),
 PCT_RENOVACION  NUMBER(8,2),
 FC_REGISTRO     DATE                     DEFAULT TRUNC(SYSDATE),
 ESTADO          CHAR(1 BYTE)             DEFAULT 'A',
 CONSTRAINT PK_QLT_T_COMISION_SUBAGENTES
 PRIMARY KEY
 (CD_COM_SUBA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
           INITIAL        64K
           NEXT           1M
           MAXSIZE        UNLIMITED
           MINEXTENTS     1
           MAXEXTENTS     UNLIMITED
           PCTINCREASE    0
           BUFFER_POOL    DEFAULT
           FLASH_CACHE    DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
 ENABLE VALIDATE,
 CONSTRAINT FK_SUBAGENTES_COMISIONES
 FOREIGN KEY (CD_SUBAGENTE)
 REFERENCES QLT_T_SUBAGENTES (CD_SUBAGENTE)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
```

```
MONITORING;


CREATE TABLE QLT_T_DIRECCIONES
(
 CD_DIRECCION    INTEGER                 NOT NULL,
 CD_CLIENTE      INTEGER,
 CD_ASEGURADORA  INTEGER,
 CD_CIUDAD       INTEGER,
 DES_DIRECCION   VARCHAR2(100 BYTE)      NOT NULL,
 SECTOR          VARCHAR2(50 BYTE),
 REFERENCIA      VARCHAR2(250 BYTE),
 CONSTRAINT PK_QLT_T_DIRECCIONES
 PRIMARY KEY
 (CD_DIRECCION)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_ASEGURADORAS_DIRECCIONES
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_CLIENTE_DIRECCIONES
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DI_CIUDADES__QLT_T_CI
 FOREIGN KEY (CD_CIUDAD)
 REFERENCES QLT_T_CIUDADES (CD_CIUDAD)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
           PCTINCREASE     0
           BUFFER_POOL     DEFAULT
           FLASH_CACHE     DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
```

```
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_FINANCIAMIENTO
(
 CD_FINANCIAMIENTO  INTEGER                NOT NULL,
 CD_FORMA_PAGO      INTEGER,
 ORNIDAL            NUMBER,
 FC_VENCIMIENTO     DATE,
 VALOR              NUMBER(8,2),
 ABONO              NUMBER(8,2),
 SALDO              NUMBER(8,2),
 SALDO_PAGO         NUMBER(8,2),
 FC_PAGO            DATE,
 OBSERVACION        VARCHAR2(250 BYTE),
 FACT_ASEG          VARCHAR2(15 BYTE),
 LETRAS             VARCHAR2(25 BYTE),
 FLG_PAGO           CHAR(1 BYTE),
 CONSTRAINT PK_QLT_T_FINANCIAMIENTO
 PRIMARY KEY
 (CD_FINANCIAMIENTO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL         64K
              NEXT            1M
              MAXSIZE         UNLIMITED
              MINEXTENTS      1
              MAXEXTENTS      UNLIMITED
              PCTINCREASE     0
              BUFFER_POOL     DEFAULT
              FLASH_CACHE     DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_FI_FORMA_PAG_QLT_T_FO
 FOREIGN KEY (CD_FORMA_PAGO)
 REFERENCES QLT_T_FORMA_PAGO (CD_FORMA_PAGO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
```

```
MONITORING;


CREATE TABLE QLT_T_PAGO
(
 CD_PAGO        INTEGER,
 CD_SUBAGENTE    INTEGER,
 CD_ASEGURADORA  INTEGER,
 CD_CLIENTE      INTEGER,
 VAL_PAGO        NUMBER(8,2),
 FC_PAGO         DATE,
 FORMA_PAGO      VARCHAR2(50 BYTE),
 DOC_PAGO        VARCHAR2(20 BYTE),
 FC_DOC_PAGO     DATE,
 BANCO           VARCHAR2(100 BYTE),
 NUM_CHEQUE      VARCHAR2(20 BYTE),
 NUM_CUENTA_CTE  VARCHAR2(50 BYTE),
 COBRADOR        VARCHAR2(20 BYTE),
 NUM_RECIBO      NUMBER,
 OBSERVACION     VARCHAR2(1000 BYTE),
 USUARIO         VARCHAR2(15 BYTE),
 ANIO            INTEGER,
 CONSTRAINT PK_PAGO
 PRIMARY KEY
 (CD_PAGO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL          64K
              NEXT             1M
              MAXSIZE          UNLIMITED
              MINEXTENTS       1
              MAXEXTENTS       UNLIMITED
              PCTINCREASE      0
              BUFFER_POOL      DEFAULT
              FLASH_CACHE      DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_PAGO_ASEGURADORA
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_PAGO_CLIENTE
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_PAGO_SUBAGENTE
 FOREIGN KEY (CD_SUBAGENTE)
 REFERENCES QLT_T_SUBAGENTES (CD_SUBAGENTE)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL          64K
           NEXT             1M
           MAXSIZE          UNLIMITED
```

```
        MINEXTENTS       1
        MAXEXTENTS       UNLIMITED
        PCTINCREASE      0
        BUFFER_POOL      DEFAULT
        FLASH_CACHE      DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_PRODUCTOS
(
 CD_PRODUCTO     INTEGER                  NOT NULL,
 CD_RAMO         INTEGER,
 CD_ASEGURADORA  INTEGER,
 DES_PRODUCTO    VARCHAR2(100 BYTE)       NOT NULL,
 ESTADO          CHAR(1 BYTE)             DEFAULT 'A'              NOT NULL,
 CONSTRAINT PK_QLT_T_PRODUCTOS
 PRIMARY KEY
 (CD_PRODUCTO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL        64K
             NEXT           1M
             MAXSIZE        UNLIMITED
             MINEXTENTS     1
             MAXEXTENTS     UNLIMITED
             PCTINCREASE    0
             BUFFER_POOL    DEFAULT
             FLASH_CACHE    DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_ASEGURADORAS_PRODUCTOS
 FOREIGN KEY (CD_ASEGURADORA)
 REFERENCES QLT_T_ASEGURADORAS (CD_ASEGURADORA)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_PR_RAMOS_PRO_QLT_T_RA
 FOREIGN KEY (CD_RAMO)
 REFERENCES QLT_T_RAMOS (CD_RAMO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
```

```
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE INDEX ASEGURADORAS_DIRECCIONES_FK ON QLT_T_DIRECCIONES
(CD_ASEGURADORA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX ASEGURADORAS_PRODUCTOS_FK ON QLT_T_PRODUCTOS
(CD_ASEGURADORA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX CIUDADES_DIRECCIONES_FK ON QLT_T_DIRECCIONES
(CD_CIUDAD)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
```

```
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX CLIENTE_DIRECCIONES_FK ON QLT_T_DIRECCIONES
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX CLI_SUBA_CLIENTES_FK ON QLT_T_CLI_SUBA
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX FK_ASEG_COMISION_BROKER ON QLT_T_COMISION_BROKER
(CD_ASEGURADORA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
```

```
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX FORMA_PAGO_FINANCIAMIENTO_FK ON QLT_T_FINANCIAMIENTO
(CD_FORMA_PAGO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX MENU_ACESOS_FK ON QLT_T_ACCESOS
(CD_MENU)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX PERFIL_ACCESOS_FK ON QLT_T_ACCESOS
(CD_PERFIL)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
```

```
NOPARALLEL;

CREATE INDEX PROVINCIA_CIUDADES_FK ON QLT_T_CIUDADES
(CD_PROVINCIA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX RAMOS_COMISION_FK ON QLT_T_COMISION_BROKER
(CD_RAMO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX RAMOS_PRODUCTO_FK ON QLT_T_PRODUCTOS
(CD_RAMO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;
```

```
CREATE INDEX SUBAGENTE_CLI_SUBA_FK ON QLT_T_CLI_SUBA
(CD_SUBAGENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX SUBAGENTES_COMISIONES_FK ON QLT_T_COMISION_SUBAGENTES
(CD_SUBAGENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE TABLE QLT_T_DETALLE_COTIZACION
(
 CD_DET_COTIZACION  INTEGER                NOT NULL,
 CD_COTIZACION      INTEGER                NOT NULL,
 CD_RAMO            INTEGER                NOT NULL,
 CD_PRODUCTO        INTEGER,
 CD_SUBAGENTE       INTEGER                NOT NULL,
 FC_INICIO          DATE,
 FC_FIN             DATE,
 NUM_DIAS           INTEGER,
 FC_POLIZA          DATE,
 NUM_POLIZA         VARCHAR2(50 BYTE),
 FACT_ASEG          VARCHAR2(15 BYTE),
 TOT_ASEG           NUMBER(8,2),
 TOT_PRIMA          NUMBER(8,2),
 FC_ORDEN_EMISION   DATE,
 FLG_ORDEN          INTEGER,
 ANEXO              VARCHAR2(50 BYTE),
 CONSTRAINT PK_QLT_T_DETALLE_COTIZACION
 PRIMARY KEY
 (CD_DET_COTIZACION)
 USING INDEX
   TABLESPACE USERS
```

```
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL         64K
              NEXT            1M
              MAXSIZE         UNLIMITED
              MINEXTENTS      1
              MAXEXTENTS      UNLIMITED
              PCTINCREASE     0
              BUFFER_POOL     DEFAULT
              FLASH_CACHE     DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_COTIZACIO_QLT_T_CO
 FOREIGN KEY (CD_COTIZACION)
 REFERENCES QLT_T_COTIZACION (CD_COTIZACION)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_PRODUCTO__QLT_T_PR
 FOREIGN KEY (CD_PRODUCTO)
 REFERENCES QLT_T_PRODUCTOS (CD_PRODUCTO)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_SUBAGENTE_QLT_T_SU
 FOREIGN KEY (CD_SUBAGENTE)
 REFERENCES QLT_T_SUBAGENTES (CD_SUBAGENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_RAMOS_DETALLE_COTIZACION
 FOREIGN KEY (CD_RAMO)
 REFERENCES QLT_T_RAMOS (CD_RAMO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_DETALLE_PRODUCTO
(
 CD_DET_PROD       INTEGER                NOT NULL,
 CD_PRODUCTO       INTEGER,
 TIPO              VARCHAR2(10 BYTE)      NOT NULL,
 DESCRIPCION       VARCHAR2(250 BYTE)     NOT NULL,
 VAL_ASEGURADO     NUMBER(8,2),
 PRIMA             NUMBER(8,2),
 PCT_VAL_SINIESTRO  NUMBER,
```

```
 PCT_VAL_ASEG      NUMBER,
 VAL_MINIMO        NUMBER(8,2),
 CONSTRAINT PK_QLT_T_DETALLE_PRODUCTO
 PRIMARY KEY
 (CD_DET_PROD)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_PRODUCTOS_QLT_T_PR
 FOREIGN KEY (CD_PRODUCTO)
 REFERENCES QLT_T_PRODUCTOS (CD_PRODUCTO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_UBICACION
(
 CD_UBICACION      INTEGER               NOT NULL,
 CD_DET_COTIZACION  INTEGER,
 DES_UBICACION     VARCHAR2(250 BYTE)       NOT NULL,
 FC_INICIO         DATE                  NOT NULL,
 FC_FIN            DATE                  NOT NULL,
 TOT_ASEGURADO     NUMBER(8,2),
 TOT_PRIMA         NUMBER(8,2),
 CONSTRAINT PK_QLT_T_UBICACION
 PRIMARY KEY
 (CD_UBICACION)
 USING INDEX
  TABLESPACE USERS
```

```
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_UB_DETALLE_C_QLT_T_DE
 FOREIGN KEY (CD_DET_COTIZACION)
 REFERENCES QLT_T_DETALLE_COTIZACION (CD_DET_COTIZACION)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE INDEX FK_COTIZACION_DET_COTIZACIONES ON QLT_T_DETALLE_COTIZACION
(CD_COTIZACION)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;
```

```
CREATE INDEX FK_DET_COT_UBICACION ON QLT_T_UBICACION
(CD_DET_COTIZACION)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX FK_SUBAGENTE_DET_COTIZACION ON QLT_T_DETALLE_COTIZACION
(CD_SUBAGENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;

CREATE INDEX PRODUCTO_DETALLE_COTIZACION_FK ON QLT_T_DETALLE_COTIZACION
(CD_PRODUCTO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX PRODUCTOS_DETALLE_PRODUCTOS_FK ON QLT_T_DETALLE_PRODUCTO
(CD_PRODUCTO)
LOGGING
```

```
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE INDEX RAMOS_DETALLE_COTIZACION_FK ON QLT_T_DETALLE_COTIZACION
(CD_RAMO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;


CREATE TABLE QLT_T_COM_TP_FACTURA
(
 CD_COM_TP_FACTURA  INTEGER                NOT NULL,
 CD_FINANCIAMIENTO  INTEGER,
 CD_SUBAGENTE       INTEGER,
 TOT_PRIMA          NUMBER(8,2),
 PCT_COM_BRK        NUMBER(8,2),
 VAL_COM_BRK        NUMBER(8,2),
 SAL_ASEGURADORA    NUMBER(8,2),
 FLG_PAGO_ASEG      CHAR(1 BYTE),
 PCT_COM_SUBA       NUMBER(8,2),
 VAL_COM_SUBA       NUMBER(8,2),
 SAL_COM_SUBA       NUMBER(8,2),
 FLG_PAGO_SUBA      CHAR(1 BYTE),
 FLG_COBRABLE       CHAR(1 BYTE),
 FACT_ASEG          VARCHAR2(15 BYTE),
 CD_DET_COTIZACION  INTEGER                NOT NULL,
 CONSTRAINT PK_QLT_T_COM_TP_FACTURA
 PRIMARY KEY
 (CD_COM_TP_FACTURA)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
```

```
            INITIAL         64K
            NEXT            1M
            MAXSIZE         UNLIMITED
            MINEXTENTS      1
            MAXEXTENTS      UNLIMITED
            PCTINCREASE     0
            BUFFER_POOL     DEFAULT
            FLASH_CACHE     DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
  ENABLE VALIDATE,
  CONSTRAINT FK_DETALLE_COTIZACION
  FOREIGN KEY (CD_DET_COTIZACION)
  REFERENCES QLT_T_DETALLE_COTIZACION (CD_DET_COTIZACION)
  ON DELETE CASCADE
  ENABLE VALIDATE,
  CONSTRAINT FK_QLT_T_CO_FINANCIAM_QLT_T_FI
  FOREIGN KEY (CD_FINANCIAMIENTO)
  REFERENCES QLT_T_FINANCIAMIENTO (CD_FINANCIAMIENTO)
  ENABLE VALIDATE,
  CONSTRAINT FK_SUBAGENTES_COM_TP_FACTURA
  FOREIGN KEY (CD_SUBAGENTE)
  REFERENCES QLT_T_SUBAGENTES (CD_SUBAGENTE)
  ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_DETALLE_PREFACTURA
(
  CD_DET_PREFACT     INTEGER                NOT NULL,
  CD_CLIENTE         INTEGER,
  CD_RAMO            INTEGER,
  CD_COM_TP_FACTURA  INTEGER,
  CD_PREFACTURA      INTEGER,
  NUM_POLIZA         VARCHAR2(25 BYTE),
  FACT_ASEG          VARCHAR2(15 BYTE),
  TOT_PRIMA          NUMBER(8,2),
  PCT_COM_BRK        NUMBER(8,2),
  VAL_COM_BRK        NUMBER(8,2),
  CONSTRAINT PK_QLT_T_DETALLE_PREFACTURA
  PRIMARY KEY
  (CD_DET_PREFACT)
  USING INDEX
```

```
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL        64K
          NEXT           1M
          MAXSIZE        UNLIMITED
          MINEXTENTS     1
          MAXEXTENTS     UNLIMITED
          PCTINCREASE    0
          BUFFER_POOL    DEFAULT
          FLASH_CACHE    DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE,
 CONSTRAINT FK_CLIENTE_DETALLE_FACTURA
 FOREIGN KEY (CD_CLIENTE)
 REFERENCES QLT_T_CLIENTES (CD_CLIENTE)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_COM_TP_FA_QLT_T_CO
 FOREIGN KEY (CD_COM_TP_FACTURA)
 REFERENCES QLT_T_COM_TP_FACTURA (CD_COM_TP_FACTURA)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_PREFACTUR_QLT_T_PR
 FOREIGN KEY (CD_PREFACTURA)
 REFERENCES QLT_T_PRE_FACTURAS (CD_PREFACTURA)
 ENABLE VALIDATE,
 CONSTRAINT FK_RAMOS_DETALLE_FACTURA
 FOREIGN KEY (CD_RAMO)
 REFERENCES QLT_T_RAMOS (CD_RAMO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_DET_PAGO
(
 CD_DET_PAGO       INTEGER,
 CD_PAGO           INTEGER,
 VAL_PAGO          NUMBER,
 CD_COM_TP_FACTURA  INTEGER,
 CONSTRAINT QLT_T_DET_PAGO_PK
 PRIMARY KEY
```

```
 (CD_DET_PAGO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
              INITIAL        64K
              NEXT           1M
              MAXSIZE        UNLIMITED
              MINEXTENTS     1
              MAXEXTENTS     UNLIMITED
              PCTINCREASE    0
              BUFFER_POOL    DEFAULT
              FLASH_CACHE    DEFAULT
              CELL_FLASH_CACHE DEFAULT
             )
 ENABLE VALIDATE,
 CONSTRAINT FK_DET_PAGO
 FOREIGN KEY (CD_PAGO)
 REFERENCES QLT_T_PAGO (CD_PAGO)
 ENABLE VALIDATE,
 CONSTRAINT FK_DETPAGO_COMTPFAC
 FOREIGN KEY (CD_COM_TP_FACTURA)
 REFERENCES QLT_T_COM_TP_FACTURA (CD_COM_TP_FACTURA)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
            INITIAL        64K
            NEXT           1M
            MAXSIZE        UNLIMITED
            MINEXTENTS     1
            MAXEXTENTS     UNLIMITED
            PCTINCREASE    0
            BUFFER_POOL    DEFAULT
            FLASH_CACHE    DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_OBJ_SEGURO
(
 CD_OBJ_SEGURO     INTEGER                NOT NULL,
 CD_DET_COTIZACION  INTEGER,
 CD_UBICACION      INTEGER,
 DES_OBJETO        VARCHAR2(100 BYTE)       NOT NULL,
 VAL_ASEG          NUMBER(8,2),
 TASA              NUMBER(3,2),
 FACTOR            NUMBER,
 PRIMA             NUMBER(8,2),
 TOT_ASEG          NUMBER(8,2),
 TOT_PRIMA         NUMBER(8,2),
 FC_INICIO         DATE                   NOT NULL,
 FC_FIN            DATE                   NOT NULL,
```

```
 MARCA             VARCHAR2(50 BYTE),
 MODELO            VARCHAR2(50 BYTE),
 COLOR             VARCHAR2(50 BYTE),
 CHASIS            VARCHAR2(100 BYTE),
 MOTOR             VARCHAR2(100 BYTE),
 PLACA             VARCHAR2(15 BYTE),
 PROPIETARIO       VARCHAR2(250 BYTE),
 TIPO_VEHICULO     VARCHAR2(50 BYTE),
 ANIO              NUMBER,
 CONSTRAINT PK_QLT_T_OBJ_SEGURO
 PRIMARY KEY
 (CD_OBJ_SEGURO)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL        64K
             NEXT           1M
             MAXSIZE        UNLIMITED
             MINEXTENTS     1
             MAXEXTENTS     UNLIMITED
             PCTINCREASE    0
             BUFFER_POOL    DEFAULT
             FLASH_CACHE    DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_OB_DETALLE_C_QLT_T_DE
 FOREIGN KEY (CD_DET_COTIZACION)
 REFERENCES QLT_T_DETALLE_COTIZACION (CD_DET_COTIZACION)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_OB_UBICACION_QLT_T_UB
 FOREIGN KEY (CD_UBICACION)
 REFERENCES QLT_T_UBICACION (CD_UBICACION)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL        64K
           NEXT           1M
           MAXSIZE        UNLIMITED
           MINEXTENTS     1
           MAXEXTENTS     UNLIMITED
           PCTINCREASE    0
           BUFFER_POOL    DEFAULT
           FLASH_CACHE    DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE TABLE QLT_T_SUBOBJETOS
(
 CD_SUBOBJETOS  INTEGER                  NOT NULL,
```

```
 CD_OBJ_SEGURO  INTEGER,
 DES_SUBOBJETO  VARCHAR2(250 BYTE)          NOT NULL,
 VAL_ASEG       NUMBER(8,2),
 TASA           NUMBER(3,2),
 FACTOR         NUMBER,
 PRIMA          NUMBER(8,2),
 CONSTRAINT PK_QLT_T_SUBOBJETOS
 PRIMARY KEY
 (CD_SUBOBJETOS)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
             INITIAL         64K
             NEXT            1M
             MAXSIZE         UNLIMITED
             MINEXTENTS      1
             MAXEXTENTS      UNLIMITED
             PCTINCREASE     0
             BUFFER_POOL     DEFAULT
             FLASH_CACHE     DEFAULT
             CELL_FLASH_CACHE DEFAULT
            )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_SU_OBJ_SEGUR_QLT_T_OB
 FOREIGN KEY (CD_OBJ_SEGURO)
 REFERENCES QLT_T_OBJ_SEGURO (CD_OBJ_SEGURO)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
           MAXSIZE         UNLIMITED
           MINEXTENTS      1
           MAXEXTENTS      UNLIMITED
           PCTINCREASE     0
           BUFFER_POOL     DEFAULT
           FLASH_CACHE     DEFAULT
           CELL_FLASH_CACHE DEFAULT
          )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE INDEX CLIENTE_DETALLE_FACTURA_FK ON QLT_T_DETALLE_PREFACTURA
(CD_CLIENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
           INITIAL         64K
           NEXT            1M
```

```
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX FK_COM_TP_FACT_DET_FACTURA ON QLT_T_DETALLE_PREFACTURA
(CD_COM_TP_FACTURA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX FK_DET_COT_OBJ_SEGURO ON QLT_T_OBJ_SEGURO
(CD_DET_COTIZACION)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX FK_FINANCI_COM_TP_FACTURA ON QLT_T_COM_TP_FACTURA
(CD_FINANCIAMIENTO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
```

```
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX OBJ_SEGURO_SUBOBJTOS_FK ON QLT_T_SUBOBJETOS
(CD_OBJ_SEGURO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;

CREATE INDEX PREFACTURA_DETALLE_FACTURA_FK ON QLT_T_DETALLE_PREFACTURA
(CD_PREFACTURA)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;



CREATE INDEX RAMOS_DETALLE_FACTURA_FK ON QLT_T_DETALLE_PREFACTURA
(CD_RAMO)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
```

```
        PCTINCREASE      0
        BUFFER_POOL      DEFAULT
        FLASH_CACHE      DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX SUBAGENTES_COM_TP_FACTURA_FK ON QLT_T_COM_TP_FACTURA
(CD_SUBAGENTE)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE INDEX UBICACION_OBJ_SEGURO_FK ON QLT_T_OBJ_SEGURO
(CD_UBICACION)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL        64K
        NEXT           1M
        MAXSIZE        UNLIMITED
        MINEXTENTS     1
        MAXEXTENTS     UNLIMITED
        PCTINCREASE    0
        BUFFER_POOL    DEFAULT
        FLASH_CACHE    DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
NOPARALLEL;


CREATE TABLE QLT_T_DETALLE_FACTURA
(
 CD_DET_FACT     INTEGER                 NOT NULL,
 CD_FACTURAS     INTEGER,
 CD_DET_PREFACT  INTEGER,
 NUM_POLIZA      VARCHAR2(25 BYTE),
 FACT_ASEG       VARCHAR2(15 BYTE),
 VAL_PRIMA       NUMBER(8,2),
 VAL_COMISION    NUMBER(8,2),
 CONSTRAINT PK_QLT_T_DETALLE_FACTURA
 PRIMARY KEY
 (CD_DET_FACT)
 USING INDEX
  TABLESPACE USERS
  PCTFREE    10
```

```
  INITRANS   2
  MAXTRANS   255
  STORAGE    (
          INITIAL         64K
          NEXT            1M
          MAXSIZE         UNLIMITED
          MINEXTENTS      1
          MAXEXTENTS      UNLIMITED
          PCTINCREASE     0
          BUFFER_POOL     DEFAULT
          FLASH_CACHE     DEFAULT
          CELL_FLASH_CACHE DEFAULT
         )
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_DETALLE_P_QLT_T_DE
 FOREIGN KEY (CD_DET_PREFACT)
 REFERENCES QLT_T_DETALLE_PREFACTURA (CD_DET_PREFACT)
 ENABLE VALIDATE,
 CONSTRAINT FK_QLT_T_DE_FACTURA_D_QLT_T_FA
 FOREIGN KEY (CD_FACTURAS)
 REFERENCES QLT_T_FACTURAS (CD_FACTURAS)
 ENABLE VALIDATE
)
TABLESPACE USERS
RESULT_CACHE (MODE DEFAULT)
PCTUSED    0
PCTFREE    10
INITRANS   1
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
LOGGING
NOCOMPRESS
NOCACHE
NOPARALLEL
MONITORING;


CREATE INDEX FACTURA_DETALLE_FACTURAS_FK ON QLT_T_DETALLE_FACTURA
(CD_FACTURAS)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
        INITIAL         64K
        NEXT            1M
        MAXSIZE         UNLIMITED
        MINEXTENTS      1
        MAXEXTENTS      UNLIMITED
        PCTINCREASE     0
        BUFFER_POOL     DEFAULT
        FLASH_CACHE     DEFAULT
        CELL_FLASH_CACHE DEFAULT
       )
```

```
NOPARALLEL;


CREATE INDEX FK_DET_PREFACTURA_DET_FACTURA ON QLT_T_DETALLE_FACTURA
(CD_DET_PREFACT)
LOGGING
TABLESPACE USERS
PCTFREE    10
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            NEXT             1M
            MAXSIZE          UNLIMITED
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
            FLASH_CACHE      DEFAULT
            CELL_FLASH_CACHE DEFAULT
           )
NOPARALLEL;
```

VEHÍCULOS | VIDA | ASISTENCIA MÉDICA | VIAJES | GENERALES | FIANZAS

QUALITYSEG
ASESORA DE SEGUROS

CERTIFICADO DE FUNCIONAMIENTO E IMPLEMENTACION

Quito, 08 de enero de 2019

Señores

INSTITUTO TECNOLOGICO SUPERIOR CORDILLERA

Presente.

De mi consideración

Me permito emitir el presente certificado, correspondiente a la entrega e implementación del Software desarrollado en el Instituto Tecnológico Superior Cordillera, ya que ha cumplido con los requisitos solicitados por parte de nuestra institución (Qualityseg S.A.) y permitido implementar el sistema de control y seguimiento de comisiones para corredores de seguros de manera profesional al Sr. Christian Javier López Viteri con numero de cedula 1722630801.

El trabajo sobre **DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA EL CONTROL DE COMISIONES PARA BROKER DE SEGUROS EN LA EMPRESA QUALITYSEG S.A.**, se encuentra terminado e implementado satisfactoriamente desde el 01 de diciembre de 2018.

Es todo cuanto puedo decir en honor a la verdad.

Atentamente.

QUALITYSEG S.A.
Firma Autorizada

Sr. Hernán Ochoa

GERENTE GENERAL

Guayaquil, Banjamín Carrión y calle Emilio Romero, Edif. City Office, piso 3, oficina 315.
Puntos de Servicio en Quito, Cuenca, Ambato, Machala y Santo Domingo de los Colorados.
contacto@qualityseguros.com.ec
www.qualityseguros.com.ec
0992747387

URKUND

Urkund Analysis Result

Analysed Document: CHRISTIAN LÓPEZ.pdf (D43612537)
Submitted: 11/7/2018 12:23:00 AM
Submitted By: cj.lopez@hotmail.es Significance: 4 %

Sources included in the report:

urkund_mejia_stefany_sistemas_15.docx (D15738072)
Pallo_Yautibug_Roberto_Sistemas.pdf (D15723231)
Vera Kevin Tesis.pdf (D30555686) TESIS
FINAL AH.pdf (D19690204)

Instances where selected sources appear:

12

Ing. Jaime Basantes, MSC.
Tutor del Proyecto

TECNOLÓGICO SUPERIOR CORDILLERA

INSTITUTO TECNOLÓGICO SUPERIOR CORDILLERA

ANÁLISIS DE SISTEMAS

ORDEN DE EMPASTADO

Una vez verificado el cumplimiento de los requisitos establecidos para el proceso de Titulación, se **AUTORIZA** realizar el empastado del trabajo de titulación, del alumno(a) **CHRISTIAN JAVIER LOPEZ VITERI,** portador de la cédula de identidad Nº 1722630801, previa validación por parte de los departamentos facultados.

Quito, 31 de octubre del 2018

20 NOV 2018

Sra. Mariela Balseca
CAJA

30 NOV 2018

Ing. Samira Villalba
PRÁCTICAS PREPROFESIONALES

INSTITUTO TECNOLÓGICO SUPERIOR "CORDILLERA"

CONSEJO DE CARRERA

Ing. Johnny Coronel
DELEGADO DE LA UNIDAD DE TITULACIÓN

INSTITUTO TECNOLÓGICO SUPERIOR "CORDILLERA"

DIRECCIÓN DE CARRERA

Ing. Hugo Heredia
DIRECTOR DE CARRERA

BIBLIOTECA INSTITUTO TECNOLÓGICO SUPERIOR "CORDILLERA"

Ing. William Parra
BIBLIOTECA

INSTITUTO TECNOLÓGICO SUPERIOR "CORDILLERA"

12 DIC. 2018

Tgla. Carolina Guerra
SECRETARIA ACADÉMICA

Nuestro reto formar seres humanos con iguales derechos, deberes y obligaciones