MEMORIA DAGSS

Java Enterprise Edition

Alejandro Gómez González

1.Breve descripción general del proyecto

El proyecto se basa en la creación de un gestor de contabilidad o de facturación para empresas o cualquier tipo de actividad comercial, pudiendo gestionar clientes, facturas y pagos asociados a dichas facturas. Se desarrollará una aplicación Web Java donde se incluyen las historias de usuario definidas en las especificaciones técnicas del proyecto.

A pesar de que el proyecto tiene dos tipos de usuarios, Administrador y Usuario, las historias solo involucran a los Usuarios teniendo que desarrollar la Gestión de Facturación y Pagos.

2. Descripción de las funcionalidades

A continuación, se explican las funcionalidades diseñadas en el proyecto, es condición indispensable para acceder a la información estar registrado y loggueado en la web.

Funcionalidades Gestión Facturas:

- **Filtro de facturas** según el DNI del cliente propietario de la factura.
- **Listado facturas del Usuario** muestra la información relativa a todas las facturas del usuario, así como sus componentes, como son: Nº Factura, NIF Cliente, Estado, Fecha Emisión, Importe. Se puede editar accionando el correspondiente botón.
- **Litado líneas de factura**: Accionando el correspondiente botón se pueden visualizar las líneas de facturas que componen determinada factura, así como sus determinados componentes: Nº Línea, Concepto, Cantidad e Importe Total. Se puede editar accionando el correspondiente botón.
- **Formularios para añadir** tanto Facturas como las líneas de factura que componen a una factura, para ello se deben accionar los correspondientes botones.

Funcionalidades Gestión Pagos:

- **Filtrado de pagos** según el DNI del cliente propietario de la factura.
- **Listado pagos del Usuario** muestra la información relativa a todos los pagos del usuario, así como sus componentes, como son: Nº Pago, Nº Factura, NIF Cliente, Estado e Importe. Se puede editar accionando el correspondiente botón.

3. Descripción de los elementos que conforman la implementación realizada

Capa de presentación (JSF):

Vistas xhtml (Facelets)

Gestión de Facturas:

- listado.xhtml: muestra la lista de Facturas asociadas al Usuario.
- fragmento-editar.xhtml: muestra el formulario que sirve para añadir una nueva factura y editar una existente.
- fragmento-ver-linas.xhtml: muestra la lista de líneas de Facturas asociadas a una Factura.
- fragmento-editar-linea.xhtml: muestra el formularo que sirve para añadir una nueva línea de factura de una factura o editar una existente.

Gestión de Pagos:

- listado.xhtml: muestra la lista de Pagos asociados al Usuario.
- fragmento-editar.xhtml: muestra el formulario para editar un pago existente.

Managed Beans (CDI)

- FacturaController.java: Es el controlador encargado de gestionar las funcionalidades de la vista de Facturas, cabe destacar en este controlador los métodos:
  - cargaInicial(): Al cargarse la vista devuelve todas las facturas
  - actualizarPagos(): Se encarga de ocultar los pagos en blanco creados al inicializar una factura sin Lineas de Factura
  - calcularImporte(): Devuelve el importe total de la factura contando los importes parciales de todas sus Líneas de Factura.
- LineasFacturaController.java: Es el controlador encargado de gestionar las funcionalidades de la vista de líneas de factura, insertada dentro de las Vistas de la gestión de Facturas.

- cargaInicial(): Al seleccionar una factura se cargan las líneas de Facturas de dicha Factura.
- doNuevaLinea(): Crea una nueva Línea de Factura

- PagoController.java: Es el controlador encargado de gestionar las funcionalidades de la vista de la Gestión de Pagos.
  - cargaInicial(): Al cargarse la vista devuelve todos los pagos.
  - calcularImporte(): Devuelve el importe del pago corriente en función del importe total y de la periodicidad de ese tipo de pagos.

En cuento a entidades se han creado 3 nuevas:

- Factura.java: Es la entidad encargada de representar a las Facturas.
- LineaFactura.java: Es la entidad encargada de representar a las líneas de Factura, tiene una relación de uno a muchos con respecto a las Facturas, una factura puede tener muchas líneas de factura, pero una línea de factura solo puede pertenecer a una Factura.
- Pago.java: Es la entidad encargada de representar a los pagos, tiene una relación de uno a muchos con respecto a las Facturas, una factura puede tener muchos pagos asociados pero un pago solo puede pertenecer a una Factura.

Capa de lógica

EJBs

- FacturaDAO.java:
  - buscarPorUsuario(usuario): Devuelve todas las facturas que tienen como propietario a usuario.
  - buscarPorCliente(cliente): Devuelve todas las facturas que tienen como Cliente a cliente.

- LineaFacturaDAO.java:
  - buscarPorFactura(factura): Devuelve todas las líneas de factura que tienen como Factura("padre") a factura.
- PagoDAO.java:
  - buscarPorUsuario(usuario): Devuelve todos los pagos que tienen como propietario a usuario.
  - buscarPorCliente(cliente): Devuelve todos los pagos que tienen como Cliente a cliente.
  - buscarPorFactura(factura): Devuelve todos los pagos de factura que tienen como Factura("padre") a factura.

4. Revisión crítica de la arquitectura del proyecto Java EE

Creo que esta práctica es muy buena para entender el funcionamiento de Java EE, que es una tecnología que las empresas en el mundo laboral valoras bastante positivamente es interesante y sorprendente ver como con solo objetos de Java es bastante sencillo implementar una solución WEB, lo que en el patrón Modelo-Vista-Controlador en PHP sería una implementación bastante más larga y costosa en tiempo con JEE se simplifica bastante.
Creo que el punto fuerte de JEE es el tema de desentenderte de la comunicación con la Base de Datos que al principio puede ser confuso por como se creen las entidades, pero su utilizas bien las etiquetas no resulta problemático.
Por otro lado, la capa de lógica creo que aporta facilidad para controlar que datos están disponibles en cada momento, a través de los contextos. A pesar de esta ventaja, a veces
se hace difícil de ver si realmente es así y no heredan algunos elementos otro estado.
Además, la inyección de dependencias aporta mucha facilidad a nivel de diseño para reducir el acoplamiento entre clases.

Para mi el gran escollo ha sido la capa de presentación, acostumbrado a programar en PHP/HTML como los atributos se inyectan aquí en las vistas me resultó complicado además de que al principio no entendía muy bien como o donde estaban definidos el código JS y CSS, en programación web que yo llevo hecho uso mucho JS para la carga dinámica de datos por lo que JEE simplifica esta parte, aunque al principio estaba perdido en ese aspecto.

Por último, creo que la práctica sirve muy bien como toma de contacto de esta tecnología, y sobre todo para cambiar el punto de vista de otras asignaturas como IU que son de PHP puro que te hace ver que las cosas pueden ser aún más simples por un lado pero un poco más complicadas por otro, como por ejemplo las parte de presentación.