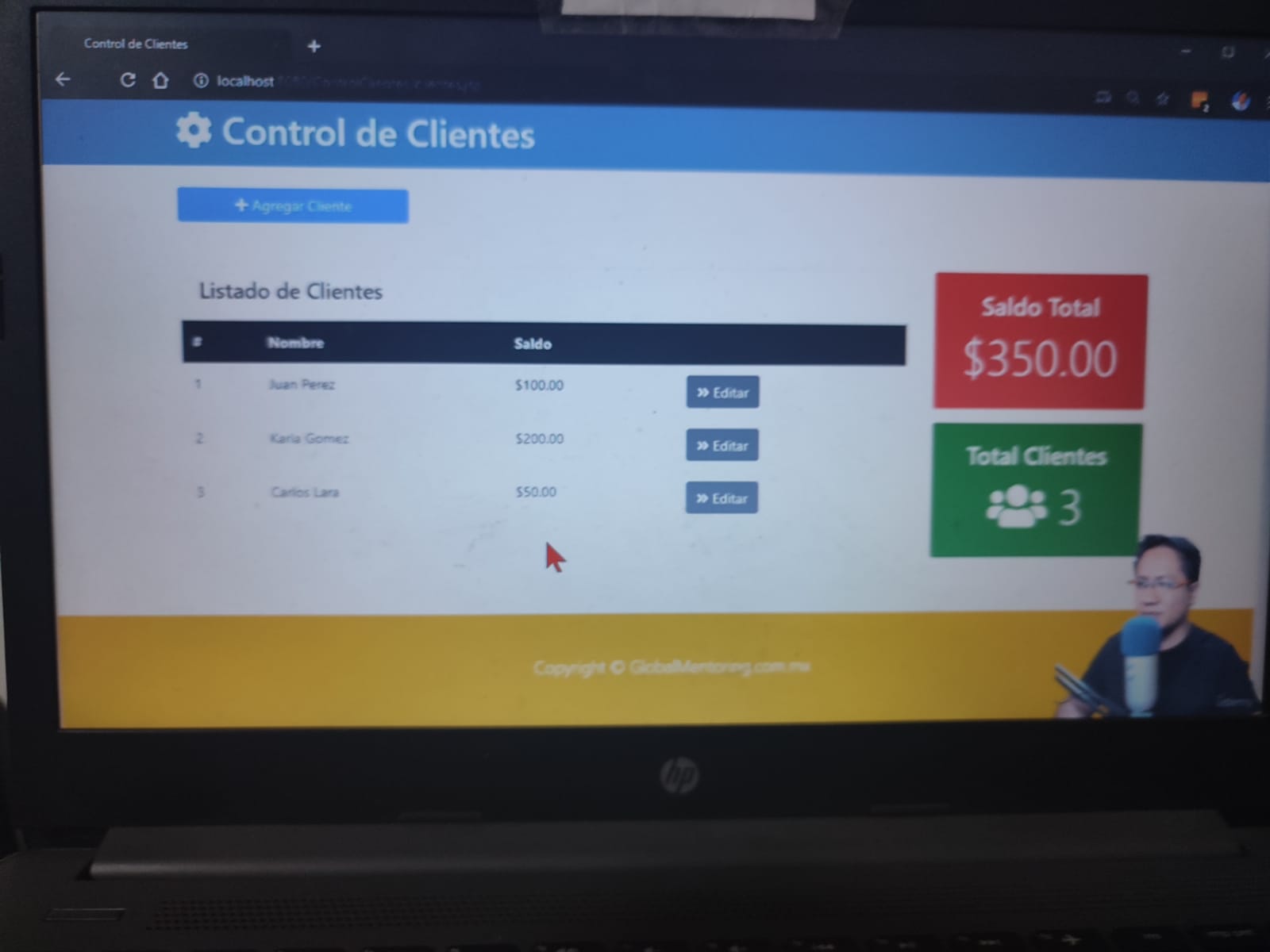
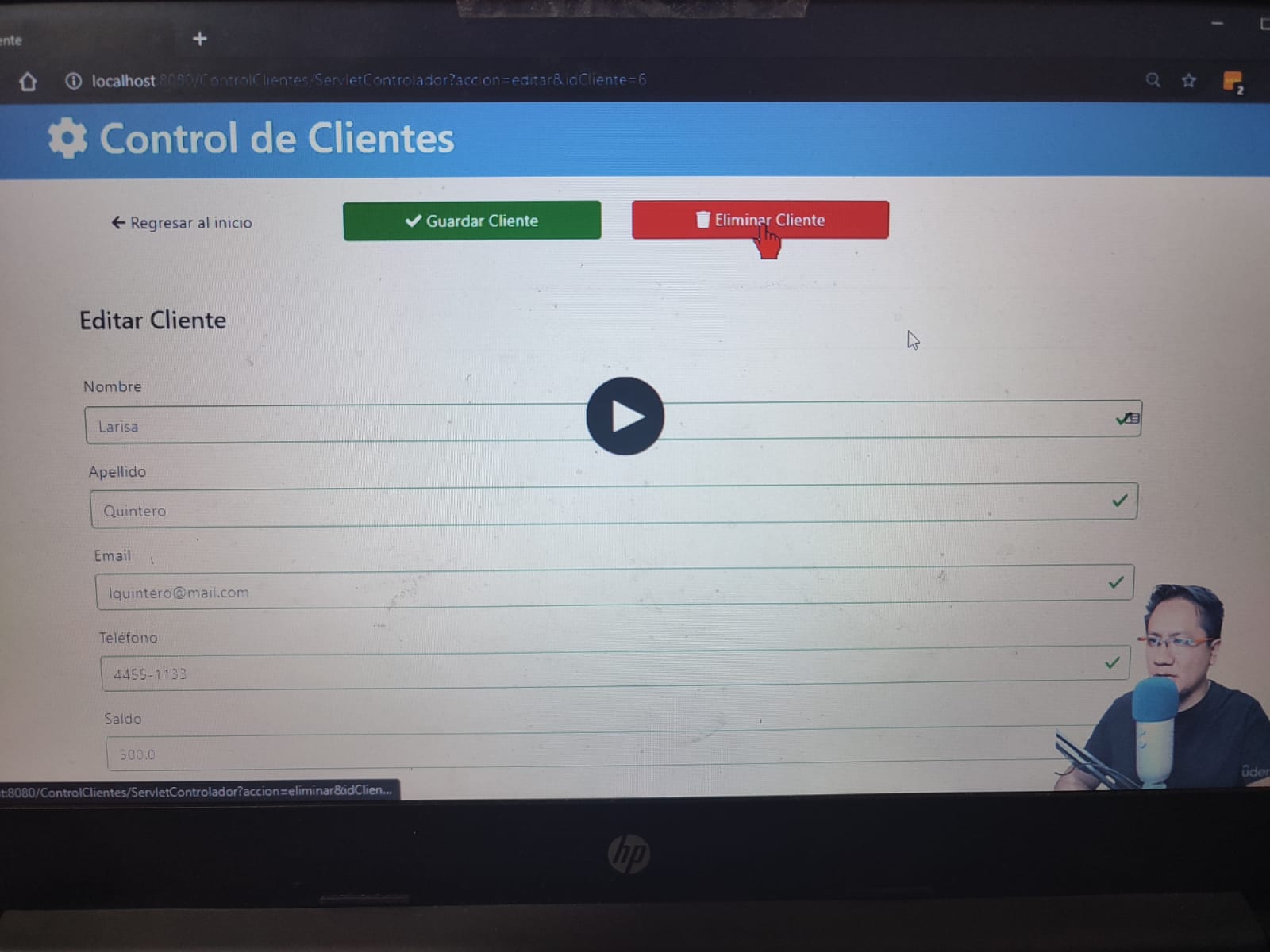
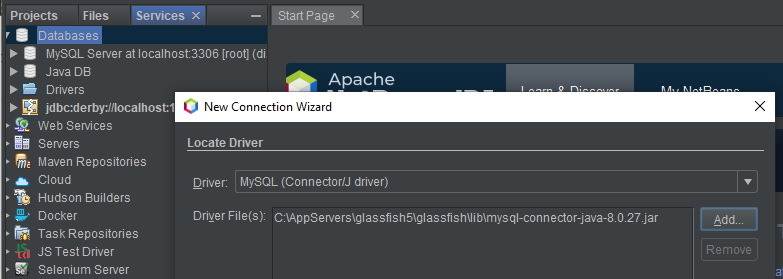
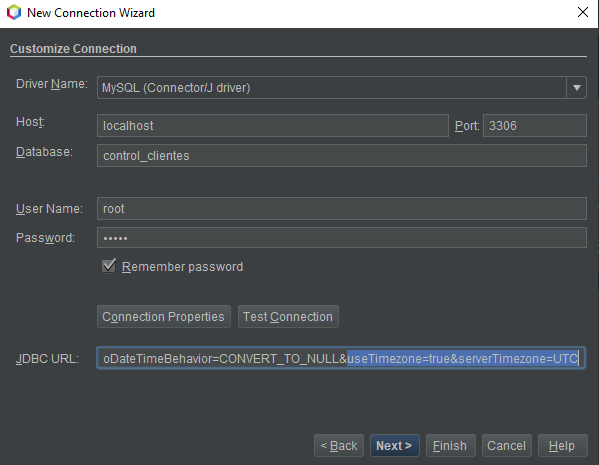
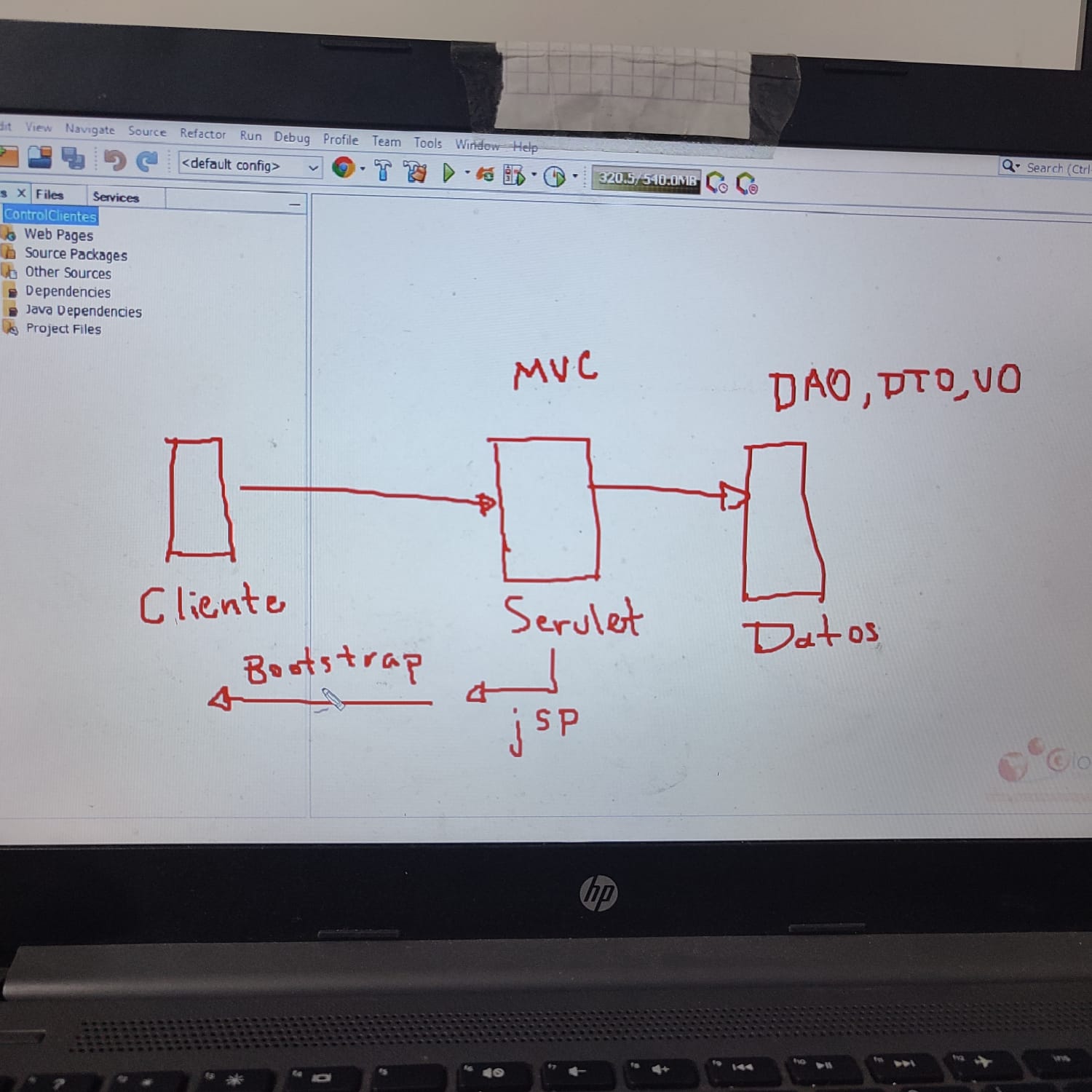
PROYECTO FINAL

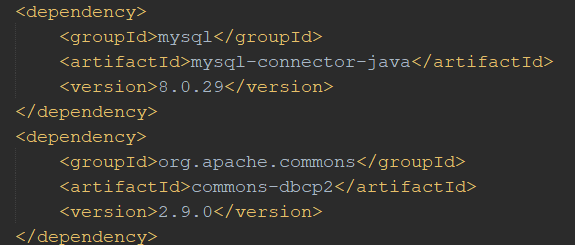
**Control Clientes**

La aplicación será par administrar clientes el modelo final será el siguiente





1. Crear MER (Base de Datos)
   1. Esquema: control\_clientes
   2. Tabla: cliente
2. Agregar (Descargar) librería de la BD (MySql) al servidor
   1. Platform independent
   2. Descargar archivo zip
   3. <https://dev.mysql.com/downloads/connector/j/>
3. Crear proyecto en IDE
4. Conectar la BD al IDE
   1. 
   2. 
5. Capas de la aplicación
   1. 
6. Ajustando el IDE
   1. Agregando dependencias en el POM
      1. Mysql y orgApache



1. Capa de datos lógica
   1. Clase datos.Conexión
      1. Permite conectar a la bd y obtener una conexión de un pool de conexiones
   2. Fuente
      1. <https://github.com/GenserBeltran/jdbcudmy/blob/Pooldeconexiones/ManejoJDBC_v4/src/main/java/datos/Conexion.java>
2. Capa de datos – Capa de presentación
   1. Clase dominio.Cliente
      1. Se realiza el mapeo de la BD de la tabla Cliente
      2. Esta clase también se conoce como DTO ClienteDTO
   2. Fuente
      1. <https://github.com/GenserBeltran/jdbcudmy/blob/Pooldeconexiones/ManejoJDBC_v4/src/main/java/domain/PersonaDTO.java>
3. Capa de datos – Accediendo a la información identidad del cliente
   1. Clase CleinteDaoJDBC
      1. Si se quiere agregar más código puedo crear una INTERFAZ, pero para este caso no se utilizara la INTERFAZ
         1. Con Interfaz
            1. <https://github.com/GenserBeltran/jdbcudmy/blob/Pooldeconexiones/ManejoJDBC_v4/src/main/java/datos/PersonaDao.java>
            2. <https://github.com/GenserBeltran/jdbcudmy/blob/Pooldeconexiones/ManejoJDBC_v4/src/main/java/datos/PersonaDaoJDBC.java>
         2. Sin Interfaz
            1. <https://github.com/GenserBeltran/jdbcudmy/blob/Pooldeconexiones/ManejoJDBC_v2/src/main/java/datos/PersonaDAO.java>
      2. Aquí se crean los métodos que se emplearan en la bd como decir el CRUD
4. Modificando la página de INDEX.html ojo como se esta usando EL será **INDEX.jsp**
   1. Realizara el llamado a JSP Controlador
   2. 0 segundos se refresca la pagina
   3. Se asigna el url que se mandara a llamar una vez se haga refresh, por medio de **EL**
   4. Se agrega el **ServletControlador**
5. Creando Jsp **web.ServletControlador;** este recuperara el listado de clientes y compartirla a **clientes.jsp**
   1. EL método **doGet** se encargará de recuperar el listado de tipo clientes que se compartirar en alguno de los alcances disponibles, en este caso **request**
   2. Hacer redirect a clientes.jsp
   3. Declarar y manejar Exepciones
6. Creando **clientes.jsp**
   1. Iterar los elementos de la lista de clientes
   2. Agregar librería de JSTL (jpa-estándar-library), librería core
   3. Iterando la lista no ordenada <ul>
   4. Con la librería core realizar un foreach
   5. Mostrar los elementos de la tabla cliente

Se puede corroborar que la conexión se ha realizado bien en las diferentes capas:

Capa de presentación aplicando el patrón de diseño MVC en el servlet controlador, se recuperó la información de la capa de datos apoyándonos de la clase de clienteJDBC y regresa un listado de objetos de tipo cliente aplicando el patrón de diseño DTO o Value Object y la información recuperada se comparte en algún alcance y se reenvía la información hacia el jsp de cliente.jsp

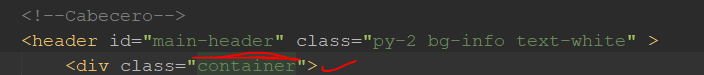
Ok OBTENIDA LA INFORMACION

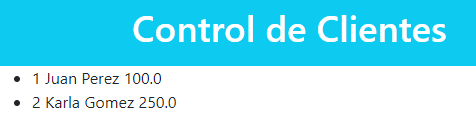
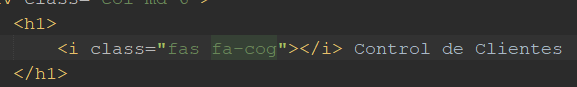
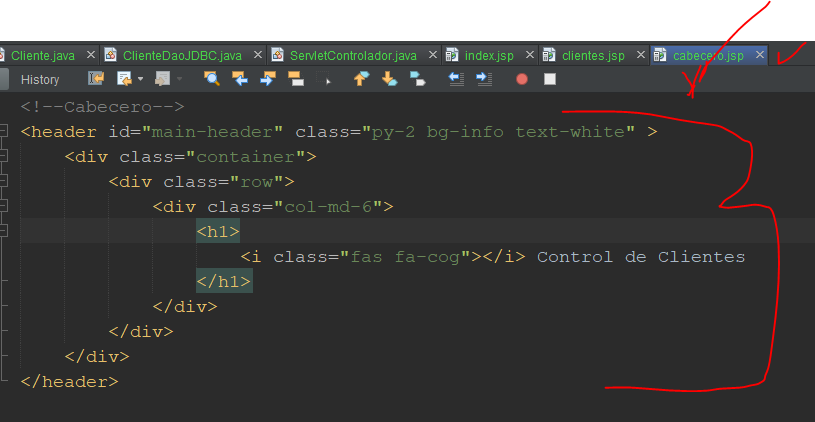
**Diseño y Front**

1. Instalación de Bootstrap
   1. **Bootstrap**: <https://getbootstrap.com/docs/5.1/getting-started/introduction/> Diseñe y personalice rápidamente sitios móviles con capacidad de respuesta con Bootstrap, el conjunto de herramientas de código abierto de front-end más popular del mundo, que incluye variables y mixins de Sass, un sistema de cuadrícula receptivo, amplios componentes preconstruidos y potentes complementos de JavaScript.
      1. Uso de JQuery
      2. Uso de Popper.js
      3. Agregar el meta y el link bootstrap
      4. Agregar las líneas de código antes de cerrar el body
2. Instalando la librería Font Awesome
   1. Copiar y pegar el link kit en el proyecto
   2. Fuente <https://fontawesome.com/kits/21ecf7bd0b/use> **Font Awesome** es la biblioteca de íconos y el conjunto de herramientas de Internet, utilizado por millones de diseñadores, desarrolladores y creadores de contenido.
3. Trabajando con Bootstrap
   1. Agregando contenedores o bloque, columnas
   2. Vinculación de clases CSS
   3. Topografía
   4. Etc
   5. Recursos <https://www.w3schools.com/bootstrap4/>
4. Ajustes Front clientes.jsp
   1. Se agrega el header
   2. Agregar div para contenedor, columna y titulo

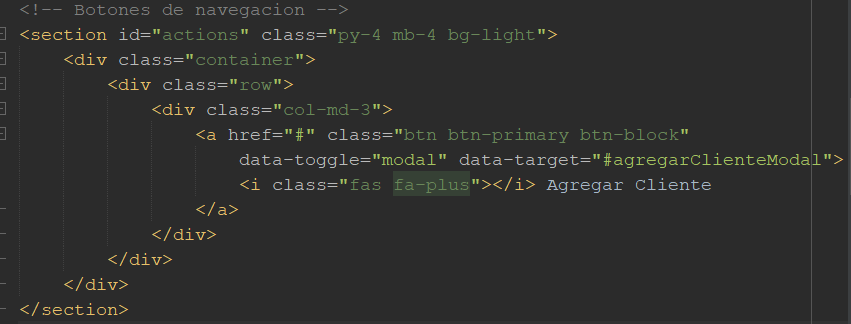
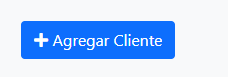
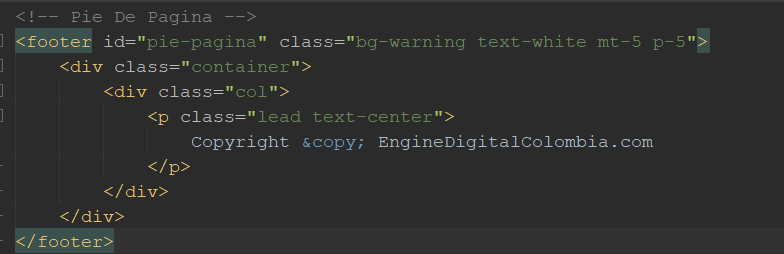
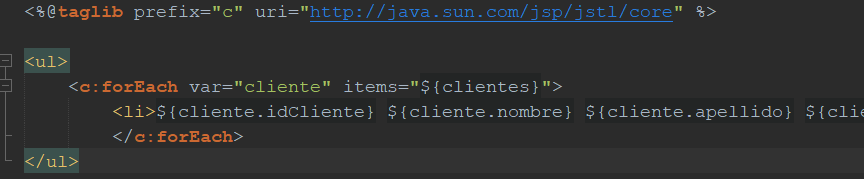
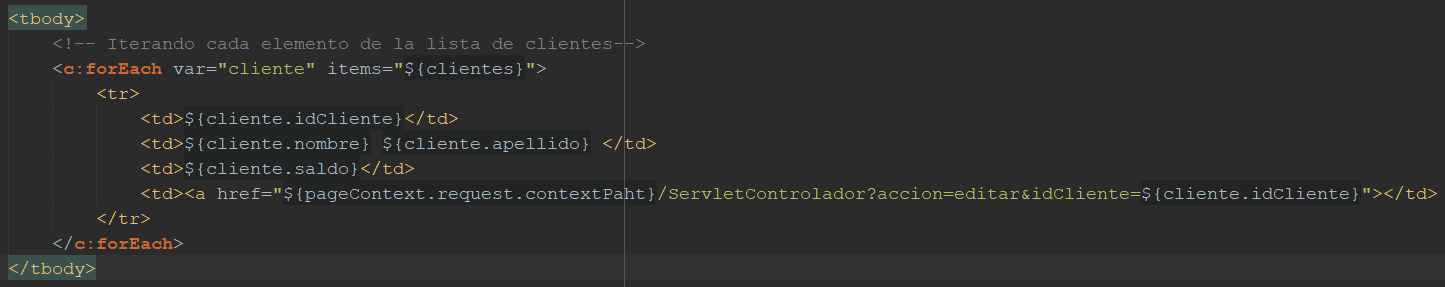
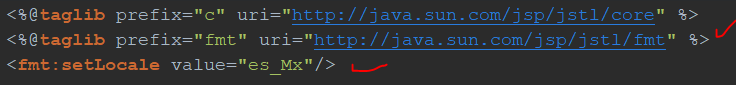
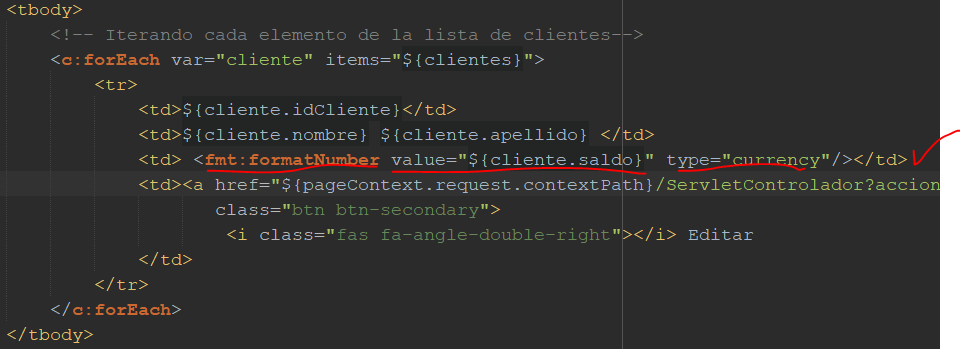


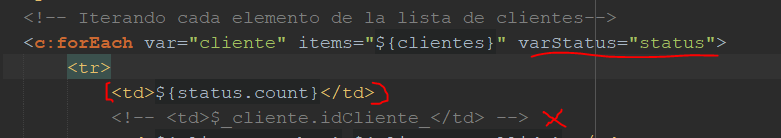
* 1. Modificando Cabecero de control de clientes con Bootstrap
     1. 
     2. 
  2. Se agrega el la clase contenedor para centrar el header



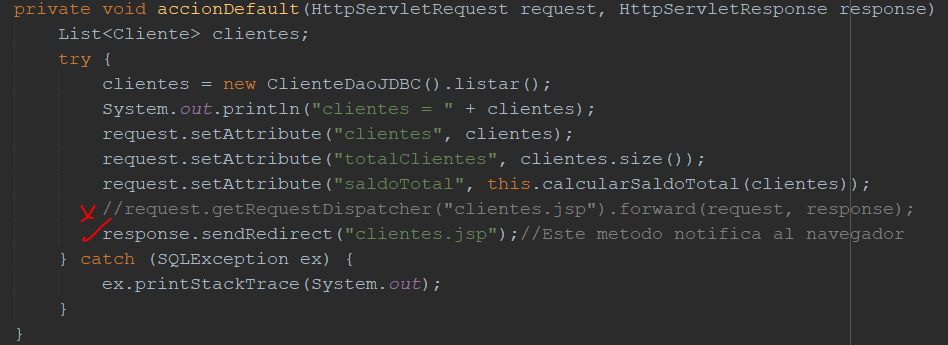
* + 1. 
  1. Se agregar al otro contenedor la fila y va contener 5 columnas
  2. Se agrega el icono
     1. 
     2. 
  3. Separando el código del cabecero en un jsp privado
     1. Creo la ruta donde se encontrarán los jsp WEB-INF/paginas/comunes
     2. 
     3. En el clientes.jsp incluyo el jsp para el cabecero

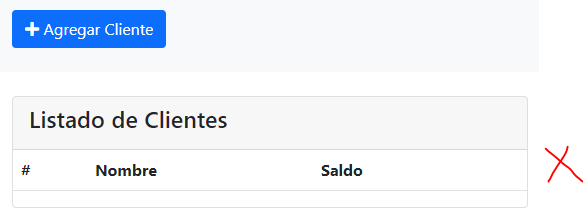
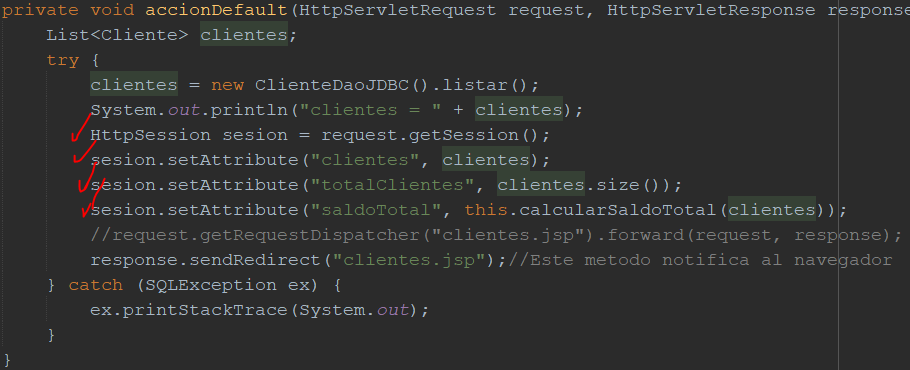
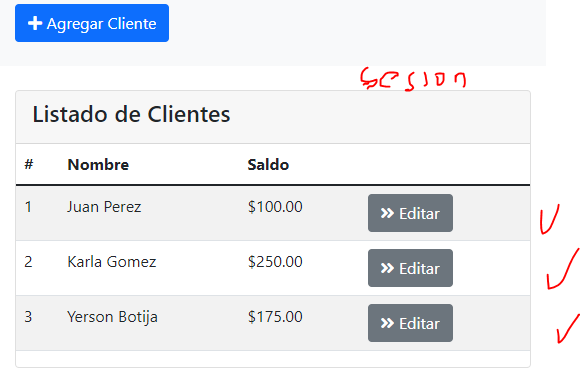
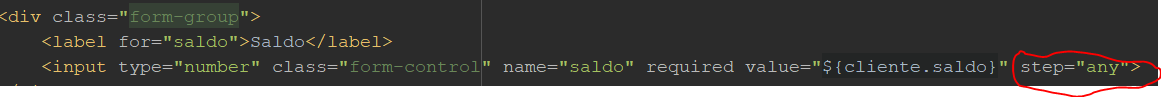


1. Incluyendo el botón agregar cliente 🡪 botonesNavegacion.jsp
   1. 
   2. 
2. Agregando el footer 🡪 piePagina.jsp
   1. 
   2. 
3. Modificando el listado de clientes TABLA
   1. Creo un jsp privado WEB-INF/paginas/cliente listadoClientes.jsp que contendrá la tabla o lista de datos de los clientes
   2. Agregar el taglig en listadoClientes.jsp
      1. 
   3. Empezamos a editar todo el jsp de listadoClientes.jsp
      1. La sección en dos columnas de ancho 9 columnas y 3 para las tarjetas
      2. Se agrega la tabla Sección HEADER con los 3 campos, item, nombre, saldo, vacío(editar)
      3. Agregar la sección BODY
         1. Agregar un for Each para iterar cada elemento de la lista
         2. Agregar el valor de cada columna de la tabla cliente (idCliente Nombre\_Apellido Saldo Editar(link))
      4. 
      5. Agregar librería para dar formato al signo de la moneda que se trabaje según la moneda (E $ $$)
      6. 
      7. 
4. Agregando as tarjetas de total de saldo y total de clientes
   1. Ajustar 3 columnas en listadoClientes.jsp
   2. Crear header y body para la tarjeta uno y dos
   3. Crear variable total clientes en clase ServletControlador
   4. Crear método calcularSaldoTotal en clase ServletControlador
   5. Compartir las variables en el alcance Request
5. Creando nuevo jsp para crear cliente NUEVO 🡪 agregarCliente.jsp
   1. Incluir agregarCliente.jsp en listadoClientes.jsp que es el jsp MODAL
   2. El id del “pop up” es el que se utilizo en botonesNavegacion.jsp de crear cliente **agregarClienteModal**
   3. Agregar el botón con símbolo de la **x** para cerrar la ventana
   4. Agregar el formulario y dentro el body de la sección
   5. Agregar los datos requeridos del cliente y requeridos según el tipo de dato
6. Procesando los datos recibidos del formulario y dar respuesta al cliente
   1. Agregar método doPost en ServletControlador.jsp, sobreescribiendo
   2. Recuperando parámetro de acción
   3. Revisar si se tiene información en la variable acción con un if y un switch
   4. Crear método insertarCliente
   5. En el caso por default se retorna a la pag de inicio pero como esta en el método doGet se debe enviar refrescada la lista de clientes, y se vuelve a calcular los totales de cliente y saldo y luego se redirige a clientes.jsp
      1. Refactorizando el punto e ya que se ejecuta constantemente este procedimiento
      2. Crear el método accionDefault para que nos regrese al listado de clientes, sera lo del punto e, entonces el método doGet llamara a este método por default
   6. En caso de que no se cumple el if del item c el else utilizara nuevamente la opción por default
   7. Recuperar la información del cliente del item d
      1. Creo las variables que contiene los datos del cliente, para saldo como debe ser un doble creo una sentencia para validar si es nulo o la cadena es vacía hacer la conversión del valor que se está recibiendo
   8. Crear el objeto de cliente con los datos recibidos (MODELO)
      1. Crear un objeto en la clase Cliente para agregar el nuevo usuario
      2. Insertamos el nuevo objeto en la base de datos
      3. Redirigiendo a la acción por default
7. Ajustando el idCliente, no siempre el de la BD va hacer el secuencial del front… Se dejara secuencial en el front.



1. Ajustar el reenvio del formulario en el metodo accionDefault quitamos el forward porque el servidor nunca nota cuando estamos regresando a la pagina principal



1. Cambiar al alcance de sesión para que tenga mas duración y guarde la información de todos los clientes y sus saldos
   1. 
   2. 
   3. 
2. Caso editar clientes 🡪 Realizar el ajuste en ServletControlador y es una petición get porque no es un formulario si no un link
   1. Es similar al método doPost
   2. Quitar el meotodo doGet el llamado al método por default
   3. Crear método de editarCliente
      1. Se recupera el idCliente y realizar conversión
      2. Recuperar el objeto cliente de la BD
      3. Colocar el objeto en algún alcance
      4. Para realizar la misma petición no usare sendredire, entonces reo una varibale con el path del jsp a utilizar
   4. Crear editarCliente.jsp
      1. Como es un jsp similar al de clientes, nuevamente debo definir enlaces de Bootstrap y caracteres de HTML, me puedo apoyar de clientes.jsp
      2. Mostrara un formulario para editar un cliente
   5. Crear botones de editar cliente
      1. Inicio y guardar
   6. La llamada al al jsp de botones de navegación editar debe estar en un formulario
   7. Agregar el path al formulario
      1. Agregar al formulario los valores del cliente para editar
      2. Reutilizar el formulario de agregar cliente
      3. En la clase daoJDBC se debe ajustar el método encontrar cuando el resulset rs… llama al método absolute, se cambia por el método next()
      4. Ajustar que saldo sea decimales y no enteros
         1. 
3. Ajustando botón de **Guardar Cliente**
   1. Ajustando el ServletControlador como el formulario de editar cliente esta en el método post procedo a crearle un nuevo caso MODIFICAR
   2. Metodo modificarCliente
      1. //Recuperando lso valores del formulario agregar client, principal el idCliente
      2. //Creamos el objeto de cliente con los datos recibidos(modelo)
      3. //Modificamos el nuevo objeto en la base de datos
      4. //Redirigiendo a la accion por default
4. Ajustando botón de eliminar cliente
   1. Este método se trabajara en get porque la petición se realiza por medio de un link declarando en botonesNavegacionEditar.jsp la acción de eliminar por medio del paht usando EL(Expression Language)
   2. Crear método eliminarCliente y agregando el caso ELIMINAR en el método get de ServletControlador.jsp
      1. //Recuperando el idCliente para elimnar el cliente
      2. //Creamos el objeto de cliente con los datos recibidos(modelo)
      3. //Eliminando el objeto en la base de datos
      4. //Redirigiendo a la accion por default
5. Mejoras del proyecto:
6. Configurar una variable basicDatasouce en la clase de conexión. Servirá cuando se llame el método getConection cada que solicitamos una conexión no se cree un objeto nuevo en DataSource; SI no se hace el ajuste cada que se cree un objeto nuevo solicitara 50 conexiones por el pool de conexiones establecido
   1. Valuidar la condición cuando el ds es null 