Crear una Página Web con Servlet y JSP usando Jakarta EE y Tomcat - 2da parte 🚀

1️⃣ Crear el Proyecto Maven con Tomcat

1. Abre tu IDE y crea un nuevo proyecto Maven Archetype.
2. Usa las siguientes configuraciones:
   - **Name**: `app`
   - **Localización**: `Donde Quieras`
   - **Catalog**: `internal`
   - **Archetype**: `escribimos web y seleccionamos la que aparece`
3. Espera que cargue todo.
4. Click en el `pom.xml` y luego en RUN, nos aparece un formulario de configuración.
5. Seleccione en USE CLASSPATH OF MODULE: **app**
6. Aplly y RUN.
7. En la terminal aparecerá un link http://localhost:8080/app. Deberás ver tu JSP.
8. Detenemos la App
9. Configura el archivo `pom.xml`:
10. Te dejo las dependencias de ejemplo pom.xml

```
    <!-- JPA (Hibernate) -->
<dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-core</artifactId>
  <version>6.2.7.Final</version>
</dependency>

<!-- JPA (API) -->
<dependency>
  <groupId>jakarta.persistence</groupId>
  <artifactId>jakarta.persistence-api</artifactId>
  <version>3.1.0</version>
</dependency>
<!--Conector mysql workbench (8.0.21)-->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.21</version>
</dependency>

<!-- Jakarta Servlet (API) -->
<dependency>
  <groupId>jakarta.servlet</groupId>
  <artifactId>jakarta.servlet-api</artifactId>
  <version>5.0.0</version>
  <scope>provided</scope>
</dependency>
```

2️⃣ Configurar tu conexión a la DB - Clases pasada 🐆

1. Abre tu Workbench e importa una DB. Te dejo una de ejemplo apple.sql
2. Configura el archivo `persistence.xml`: Te dejo una de ejemplo persistence.xml
3. En el paquete persistence: o ConfigJPA: Esta clase se encargará de configurar Hibernate. o PersonaJPA: Esta clase será un DAO (Data Access Object) para realizar operaciones CRUD sobre la entidad Persona.
4. En el paquete entities: o Persona: Esta entidad mapeará la tabla "persona" de tu base de datos. Anótalo con las anotaciones de Hibernate para definir la correspondencia entre las propiedades de la clase y las columnas de la tabla.

3️⃣ Configurar el Directorio del Proyecto 🗂️

Estructura básica del proyecto: Archivos.java

```
src/
└── main/
    └── java/
        └── com/
            └── example/
                ├── controllers/
                │   └── PersonaController.java
                ├── entities/
                │   ├── Persona.java
                ├── exceptions/
                │   ├── PersonaException.java
                ├── persistence/
                │   ├── ConfigJPA.java
                │   └── GenericoJPA.java
                └── servlets/
                    ├── PersonaFormServlet.java
                    └── PersonaServlet.java
```

Detalles del Proyecto

`controllers`

- Contiene clases como `PersonaController` que controlan la lógica principal del proyecto.

`entities`

- Almacena las clases del modelo, como `Persona`, `Producto` y `Tarjeta`.
- Estas son típicamente entidades JPA que representan las tablas de una base de datos.

`exceptions`

- Se reserva para definir excepciones personalizadas, aunque no se muestran en la imagen.

`persistence`

- Contiene clases para la configuración y gestión de la capa de persistencia, como `ConfigJPA` y `GenericoJPA`.

`servlets`

- Incluye los servlets como `PersonaFormServlet` y `PersonaServlet`, que manejan las solicitudes HTTP.

3️⃣ Configurar el Directorio del Proyecto 🗂️

Estructura básica del proyecto: JSP

```
src/
└── main/
    └──webapp/
        ├── partials/
        │   └── header.jsp
        ├── public/
        │   ├── css/
        │   ├── img/
        │   └── js/
        ├── WEB-INF/
        │   └── web.xml
        ├── index.jsp
        ├── persona.jsp
        └── personaForm.jsp
```

Descripción de las carpetas y archivos

`partials/`

- Contiene fragmentos JSP reutilizables como `header.jsp`, que puede incluirse en otros archivos JSP.

`public/`

- `css/`: Almacena archivos CSS para la estilización.
- `img/`: Almacena imágenes del proyecto.
- `js/`: Almacena archivos JavaScript para la funcionalidad del cliente.

`WEB-INF/`

- `web.xml`: Archivo de configuración de la aplicación web (descriptor de despliegue).

Archivos JSP

- `index.jsp`: Página principal de la aplicación.
- `persona.jsp`: Página relacionada con la entidad `Persona`.
- `personaForm.jsp`: Página del formulario de la entidad `Persona`.

---

4 Crear un Servlet para la ruta /persona

1. Crea el archivo `PersonaServlet.java` en `src/main/java/com/example/`.
2. Añade el siguiente código:

```
@WebServlet(urlPatterns = "/persona")
public class PersonaServlet extends HttpServlet {

    private PersonaController personaController= new PersonaController();
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletExcepti
        //buscar info en la DB
        List<Persona> listado = personaController.findAll();

        //enviarle info a la vista
        request.setAttribute("listado", listado);

        // Redirigir a la vista JSP
        request.getRequestDispatcher("persona.jsp").forward(request, response);
    }
}
```

5 Crear la Vista persona.jsp 🌐

1. Crea el archivo `persona.jsp` en `src/main/webapp/`.
2. Te recomendamos trabajar los archivos JSP desde VSC, nos ayudara con las etiquetas html.
3. Añade el siguiente contenido:

```
<%-- Configurar metadatos --%>
<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%-- Importar las clases --%>
<%@ page import="java.util.List, com.example.entities.Persona" %>

<!DOCTYPE html>
<html lang="en">
<body>
    <h1>Persona listado</h1>

    <ul>
         <%  List<Persona> listado = (List<Persona>) request.getAttribute("listado");

            for( Persona persona: listado ) { %>

            <li>Cliente: <%= persona.getNombre() %> -
                Apellido: <%= persona.getApellido()%> -
                Tarjeta :  <%= persona.getTarjetas().stream().findFirst().get().getTipo() %>
            </li>
        <% } %>
    </ul>



</body>
</html>
```

6️⃣ Levanta la App con el Tomcat 🖥️

1. Ir a Run y ejecutar.
2. Abre la página http://localhost:8080/app/persona.

Has creado una página web básica con un Servlet, JSP y Tomcat. ¡Felicidades! 🎉

🥳 ¡Listo!