

Descripción, Despliegue y Funcionamiento: Expense Manager



Iordache Mihai Laurentiu - 2º DAM

ÍNDICE DE CONTENIDOS

1. Descripción del proyecto	3
1.1. Tecnologías utilizadas	3
2. Despliegue	3
3. Configuración de bases de datos y funcionamiento	7
3.1. Configuración de parámetros de acceso	8
4. Funcionamiento de la aplicación	10
4.1. Pantalla principal	10
4.2. Login y Registro	11
4.3. Index - Dashboard	12
4.4. Manager - Add	12
4.5. Manager - Manage	13
4.6. Logout	13
5. Acceso a la documentación	13

Descripción, Despliegue y Funcionamiento: Expense Manager

1. Descripción del proyecto

El proyecto en sí trata sobre un **manejador de gastos**. Este proyecto lo tenía pensado como proyecto personal, pero ya que ha surgido el tema de desarrollar un proyecto para la clase de HLC pues al final he decidido finalizarlo y presentarlo en este módulo.

El usuario puede **añadir, quitar, buscar** gastos sobre cosas que hayan comprado este mes en concreto. No se pueden añadir cosas que se hayan comprado en meses anteriores.

Es un CRUD sencillo pero que tiene su dificultad desarrollarlo bajo la arquitectura habitual de una aplicación en Spring (sobre todo siendo principiante en esta tecnología).

1.1. Tecnologías utilizadas

Las tecnologías principales son: **Java** y **Javascript** aunque para ser específicos son:

- **Java**
- **Spring Framework (JPA, Security)**
- **Hibernate ORM**
- **Vue Js**
- **Axios**
- **jQuery**
- **Bootstrap + CSS + HTML**

2. Despliegue

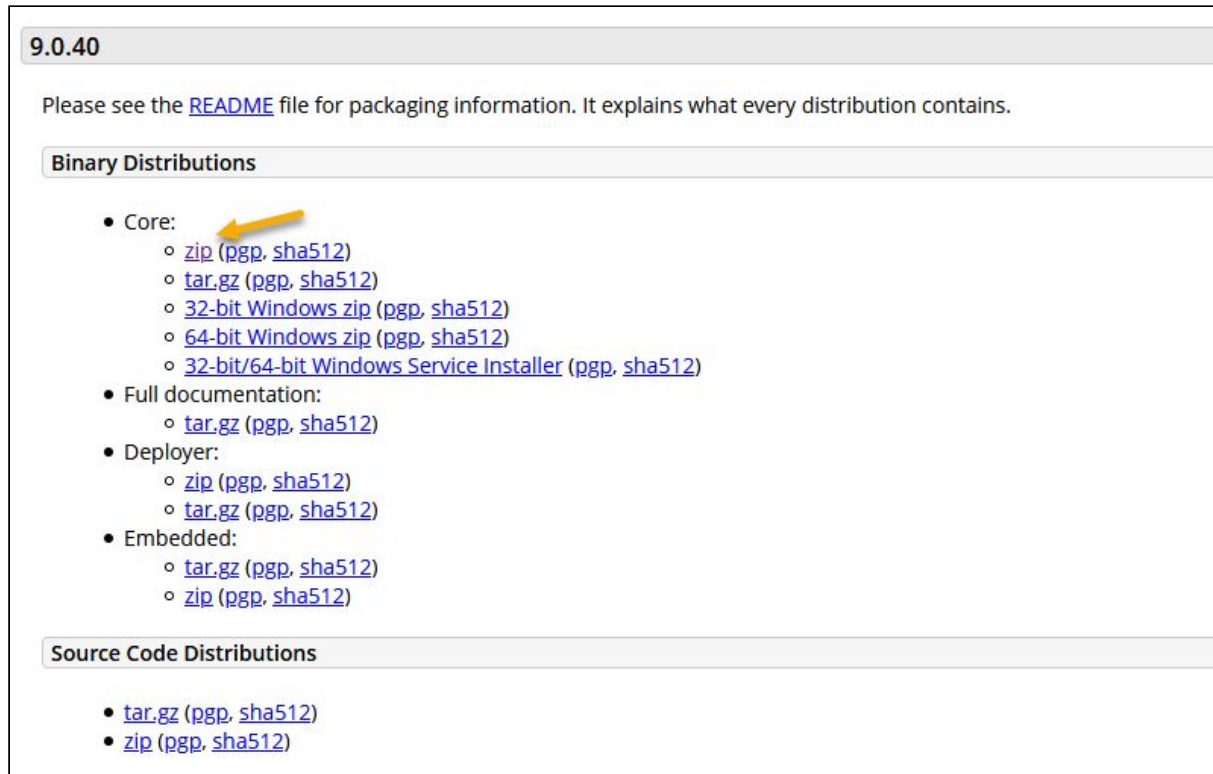
Vamos a comenzar con el despliegue de la aplicación. El despliegue se explicará para el IDE **Eclipse**.

Lo primero es, evidentemente, tener instalado **Eclipse para JAVA EE**. Es muy importante que sea la versión para Java empresarial. Lo podemos descargar desde <https://www.eclipse.org/downloads/packages/release/kepler/sr2/eclipse-ide-java-ee-developers>.

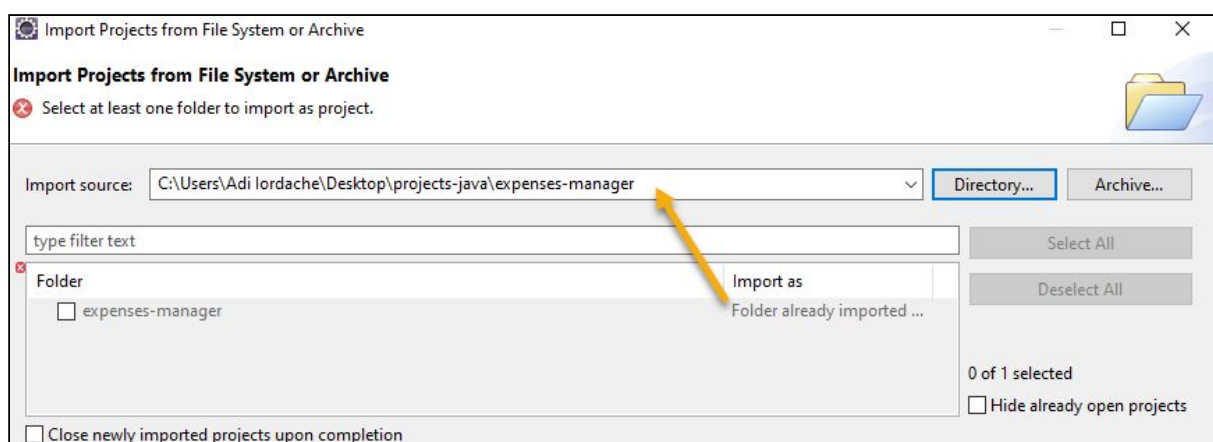
Lo segundo es **descargar el servidor de aplicaciones**. En este proyecto he utilizado **Tomcat** así que será el que utilizaremos.

Accedemos a la web: <https://tomcat.apache.org/download-90.cgi>. Nos vamos al apartado **Core** y pulsamos la **primera opción** que dice “zip”.

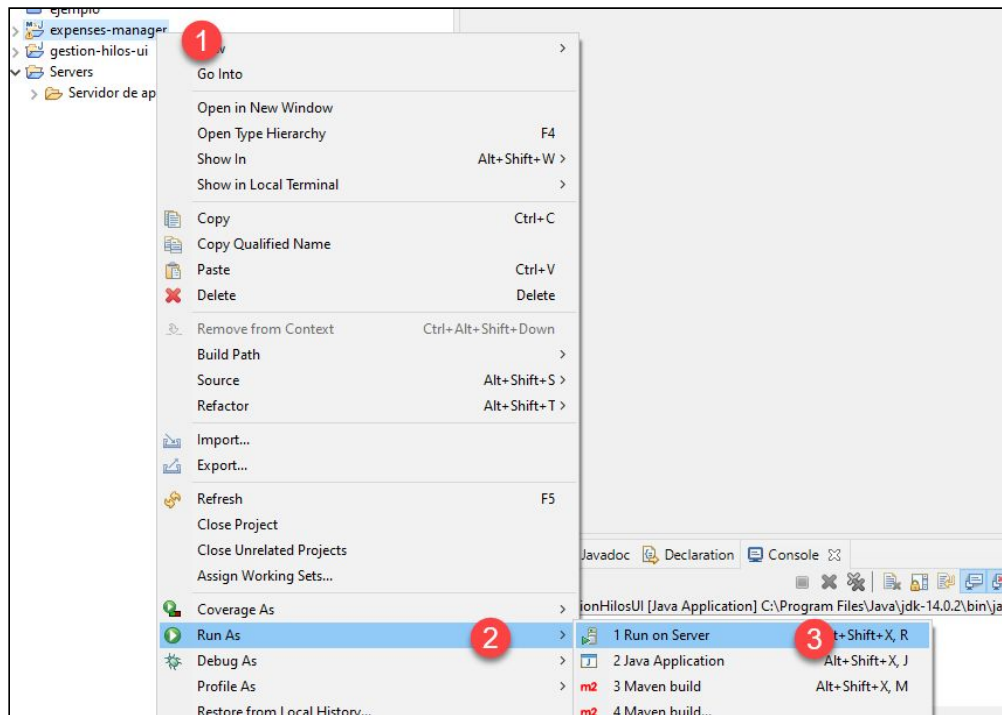
Lo descomprimos y lo dejamos en una carpeta bien localizada.



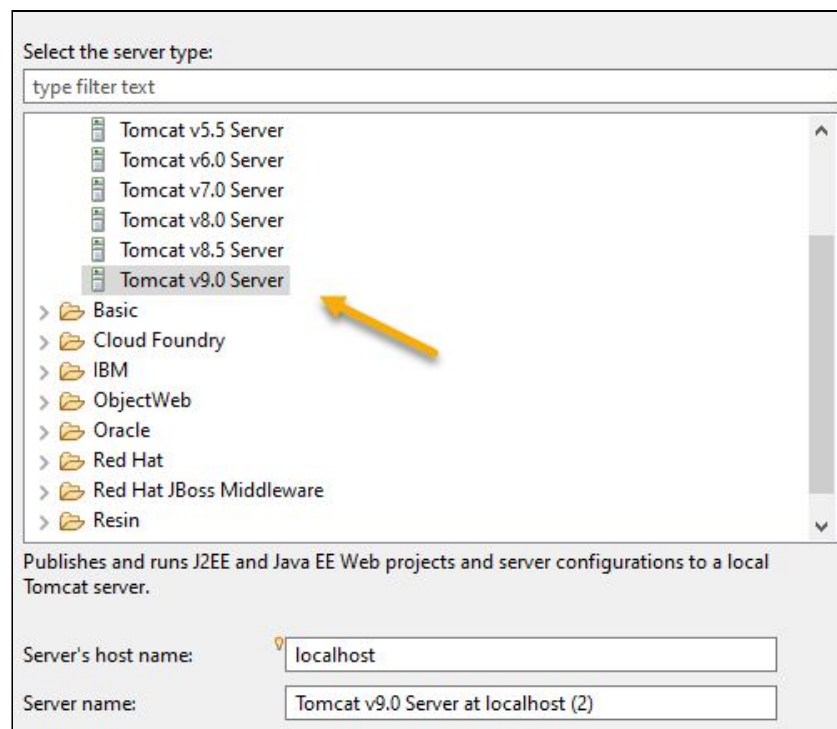
Importamos el proyecto pulsando en **File → Open projects from filesystem**. Una vez hecha la importación, esperamos a que se resuelvan todo el conjunto de dependencias.



Una vez finalizada la importación, pulsamos sobre el proyecto **botón derecho** → **Run As** → **Run on Server**.

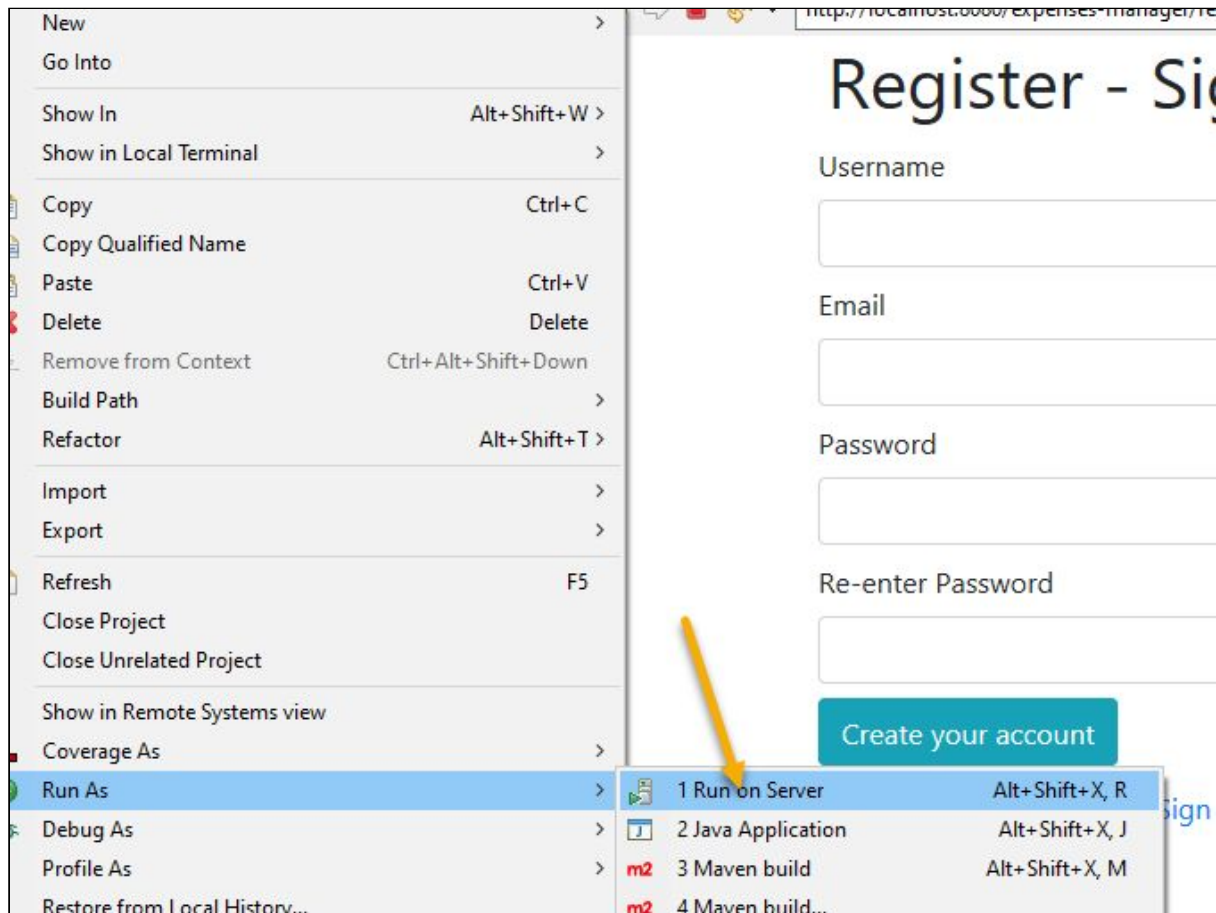
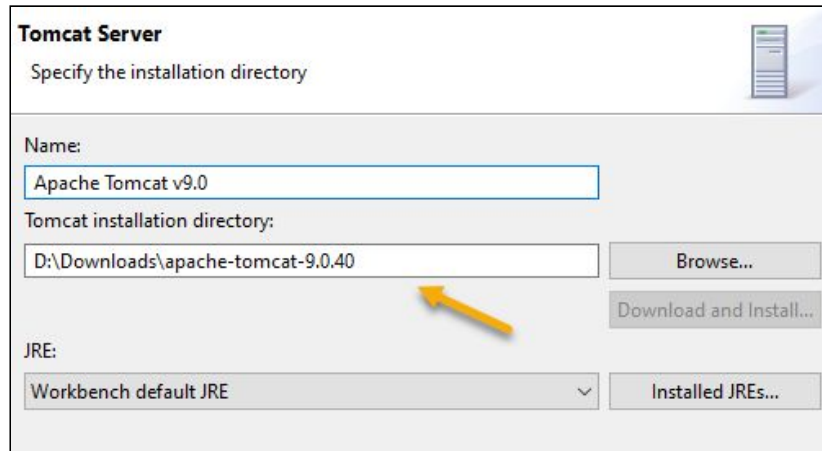


Bajo la opción de **Apache** seleccionamos la versión de Tomcat 9 (que es la versión que hemos descargado). Una vez lo hagamos pulsamos en next.

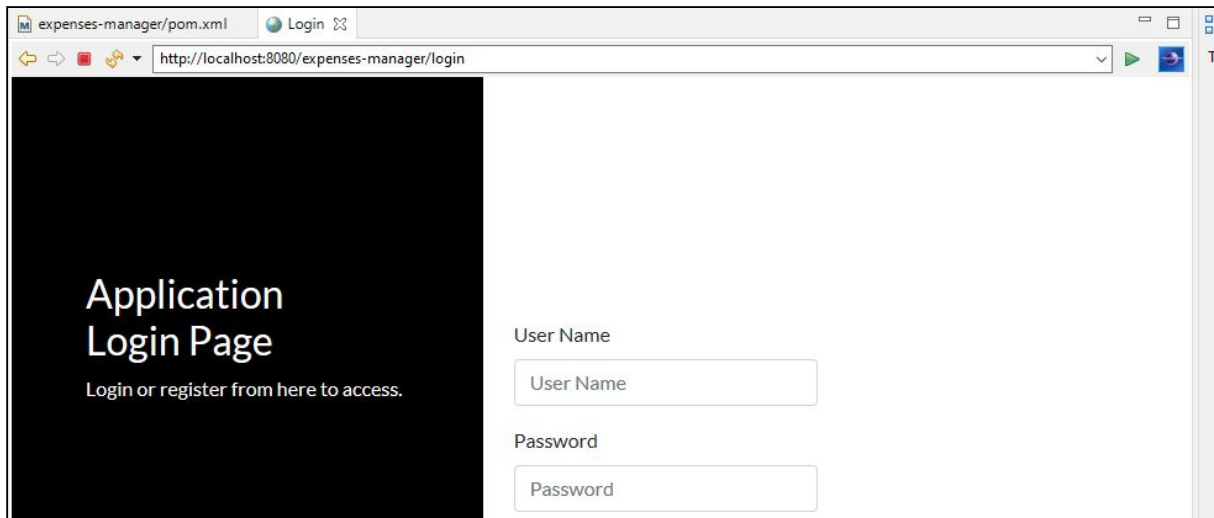


Seleccionamos la **ruta de tomcat**, la ruta a la carpeta que se ha extraído recientemente tras la descarga del servidor. Pulsamos en **finalizar**.

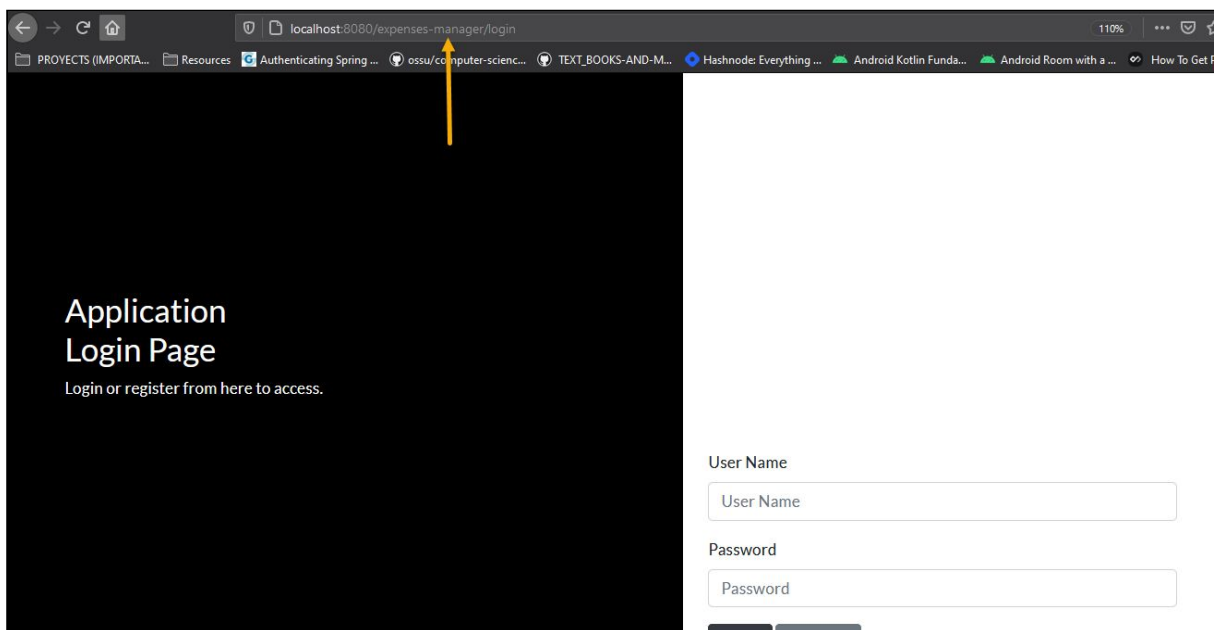
Una vez finalizado, pulsamos **botón derecho sobre la carpeta del proyecto → Run as → Run on Server**.



Se nos abrirá el proyecto en un navegador propio de eclipse. Copiamos la URL y la llevamos a un navegador tipo Chrome o Firefox.



Si introducimos la URL en el navegador, vemos que se nos abre la página de Login. El proyecto está desplegado.



3. Configuración de bases de datos y funcionamiento

Vamos a realizar en este punto, la configuración de la **base de datos de los usuarios** y de la **base de datos encargada de las entidades de tipo gasto**. Posteriormente, veremos el funcionamiento de la aplicación. Para ello conviene usar una herramienta como Laragon o XAMPP para montar un servidor de base de datos.

Existen **dos ficheros sql** que he adjunto al fichero comprimido:

- La base de datos de usuarios
- La base de datos de gastos

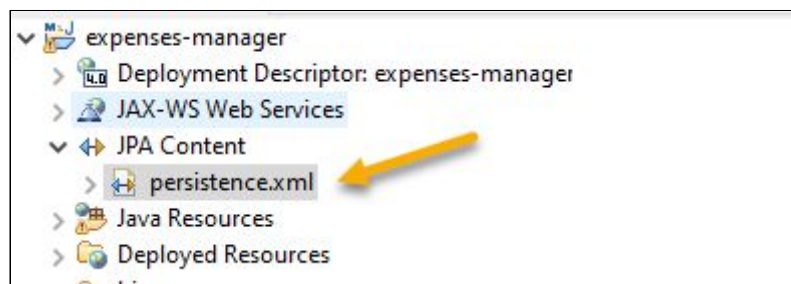
Creamos las dos bases de datos en el SGBD con el mismo nombre que los ficheros: **expensemanager** y **usuarios** respectivamente. Posteriormente, **importamos** los ficheros sql en las bases de datos.

Debemos hacer esto puesto que a pesar de haber elegido un ORM para la creación de la base de datos, he decidido que la base de datos sería escrita por mi y no la crearía un ORM a la hora de lanzar la aplicación.

3.1. Configuración de parámetros de acceso

Necesitamos configurar los parámetros de acceso a las bases de datos. Para ello, primero configuramos el acceso a **expensemanager**.

En la estructura del proyecto, buscamos **JPA Content** y pulsamos en **persistence.xml**, que es el fichero de persistencia de las entidades.



Vemos la URL de acceso a la base de datos **expensemanager**. Normalmente, el puerto de acceso a MySQL es 3306. Lo modificamos en caso de ser necesario.

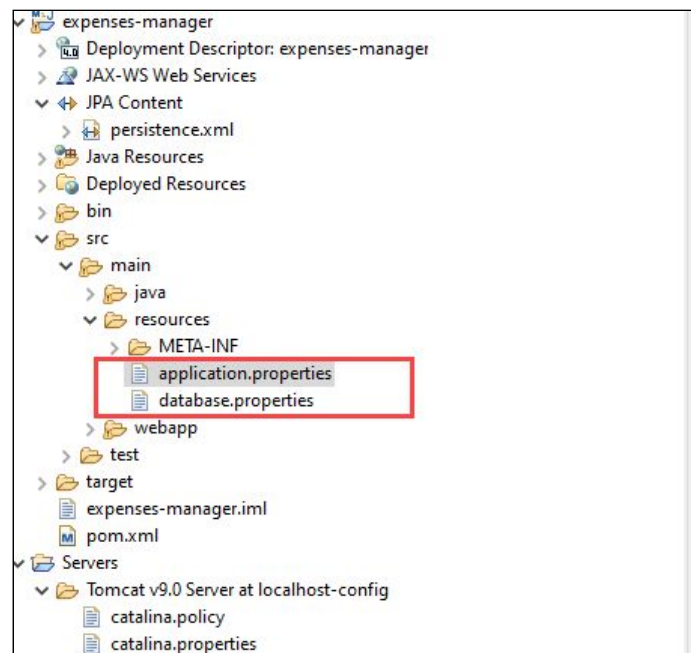
Posteriormente, introducimos los datos de: **usuario** y **contraseña**. En mi caso es **'root'** y vacío.


```
<persistence-unit name="expenseManager">
  <properties>
    <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver" />
    <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/expensemanager" />
    <property name="javax.persistence.jdbc.user" value="root" />
    <property name="javax.persistence.jdbc.password" value="" />

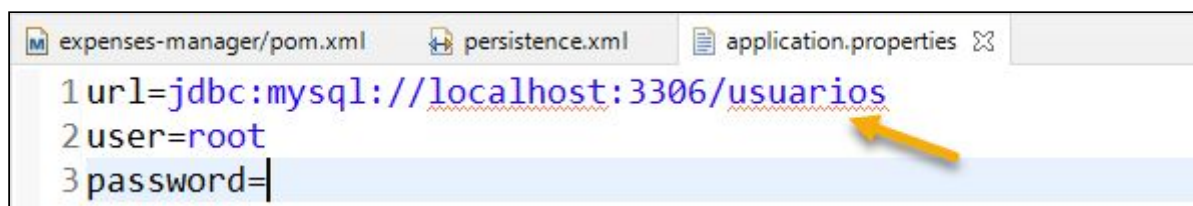
    <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect" />
    <property name="hibernate.show_sql" value="true" />
  </properties>
</persistence-unit>
```

Una vez realizado esta configuración, vamos a hacer la configuración a la base de datos de **usuarios**.

Ahora vamos a los dos ficheros siguientes: **application.properties** y **database.properties**.

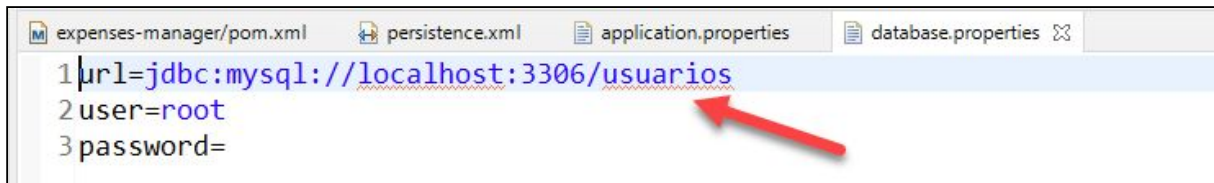


Establecemos la url para el acceso a la base de datos **usuarios** y también establecemos el **usuario** y **contraseña**.



```
1 url=jdbc:mysql://localhost:3306/usuarios
2 user=root
3 password=
```

Lo mismo hacemos en el fichero **database.properties**. Una vez llegado a este punto, ya podemos usar la aplicación.



```
expenses-manager/pom.xml  persistence.xml  application.properties  database.properties  X
1 url=jdbc:mysql://localhost:3306/usuarios
2 user=root
3 password=
```

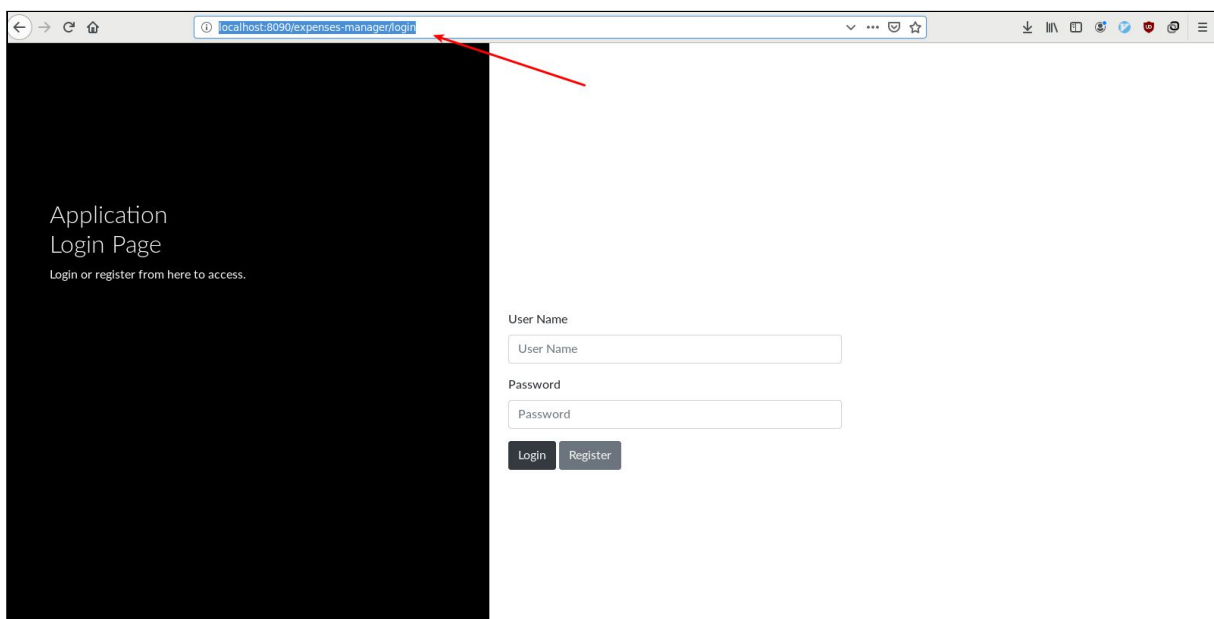
4. Funcionamiento de la aplicación

Vamos a probar el funcionamiento de la aplicación.

4.1. Pantalla principal

Lo primero que se nos muestra es el apartado de **login**, ya que así ha sido indicado en **Spring Security**.

Como vemos, tenemos principalmente **dos opciones**: Login y Registro. Tras haber arrancado el proyecto, justo después de hacer la importación de las bases de datos, debemos hacer un registro.



4.2. Login y Registro

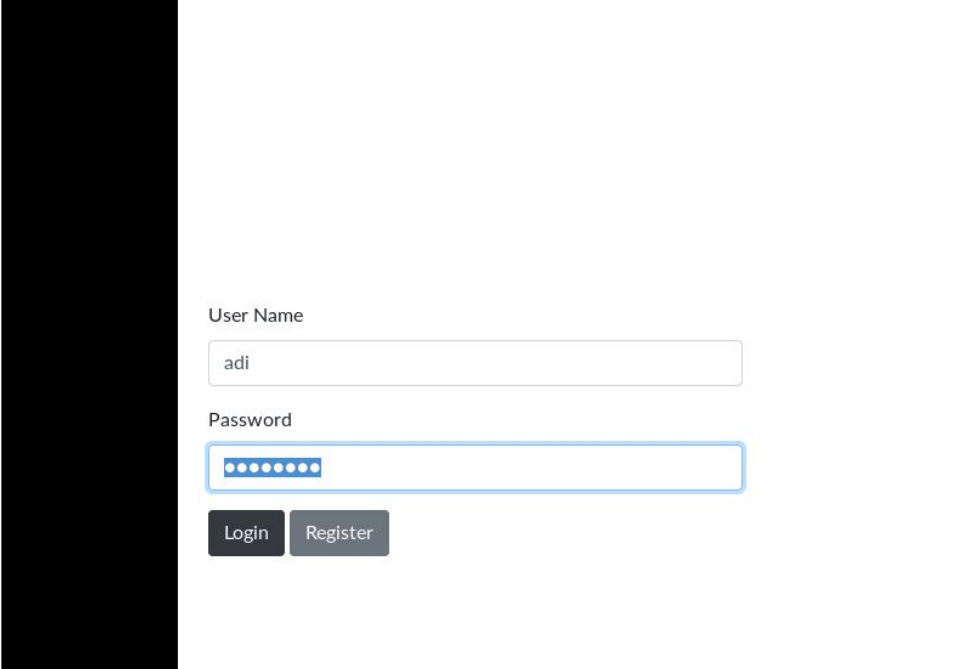
En la página de **registro**, cada campo tiene su validación. Desde la contraseña hasta el **username** y **correo**, por lo tanto, **no se podrán repetir nombres de usuario ni correos**. Si creamos la cuenta exitosamente nos lleva al **login**



The screenshot shows a web browser window with the address bar displaying 'localhost:8090/expenses-manager/register'. The page title is 'Register - Sign Up'. The form contains the following fields and elements:

- Username:** A text input field containing the value 'adi'.
- Email:** A text input field containing the value 'usuario@usuario.es'.
- Password:** A password input field with masked characters '.....'. A red border and an error icon indicate a validation failure. Below the field, a red message states: 'Password must be at least 8 characters'.
- Re-enter Password:** A password input field with masked characters '.....'.
- Buttons:** A teal button labeled 'Create your account' and a blue link labeled 'Already have an account? Sign In'.

En la página de **login**, una vez registrados, introducimos los datos. En el caso de que sean erróneos, se nos indicará.



The screenshot shows a login form with a black sidebar on the left. The form contains the following fields and elements:

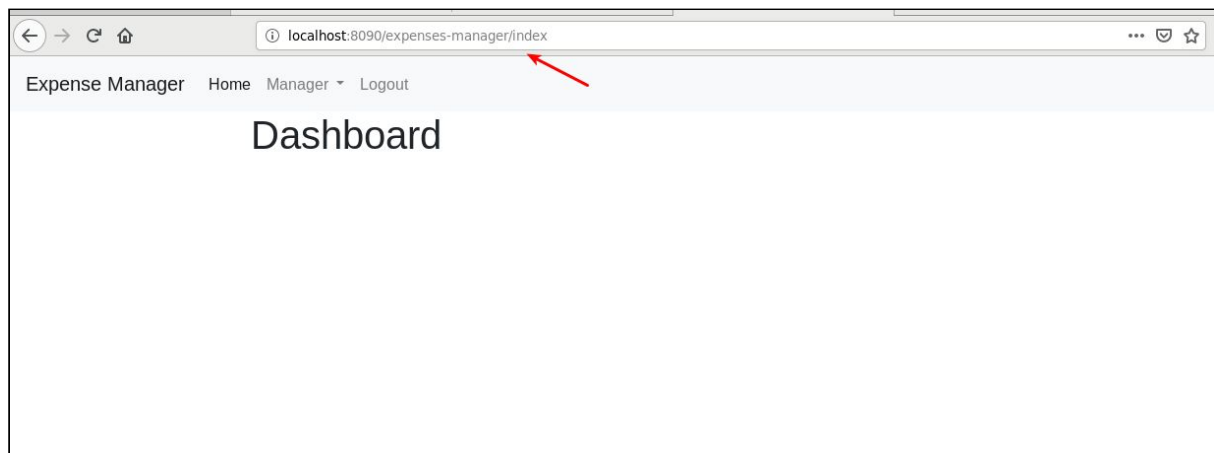
- User Name:** A text input field containing the value 'adi'.
- Password:** A password input field with masked characters '.....'.
- Buttons:** Two buttons labeled 'Login' and 'Register'.

4.3. Index - Dashboard

Esta es la primera pantalla que se nos muestra al iniciar sesión. Esta pantalla no hace nada en concreto, simplemente es una pantalla más que está asociada a una acción de un controlador.

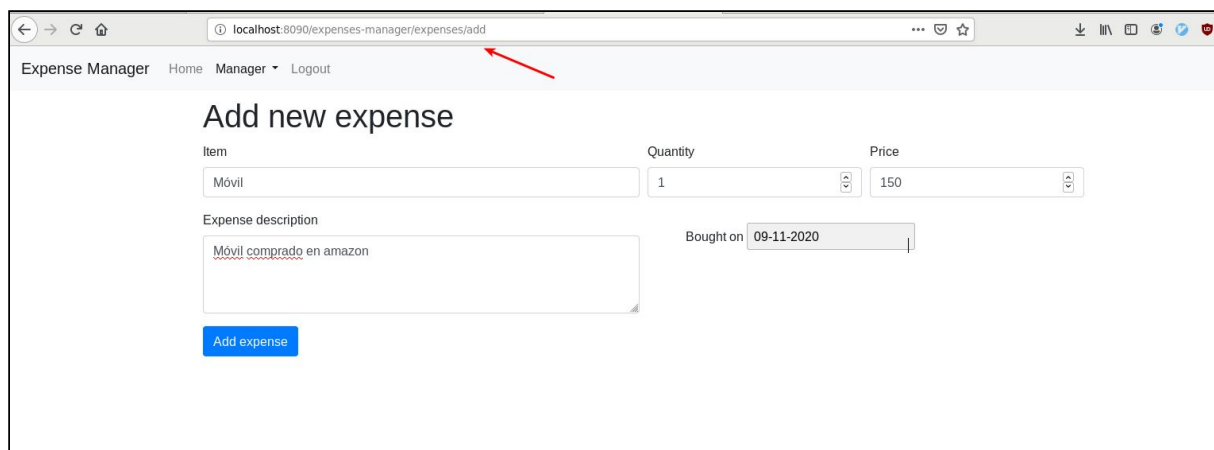
Vemos que disponemos de tres botones: **Home**, **Manager** y **Logout**.

Evidentemente, todas las acciones de **login** y **registro** estarán restringidas una vez hayamos iniciado sesión. Lo que quiere decir es que **no hay forma de acceder a las acciones de registro o login una vez iniciado sesión**.



4.4. Manager - Add

Vamos a ver la primera opción del **manager**: Añadir. En esta opción, añadimos **gastos** para el **usuario logueado**. Todo el formulario tiene control de errores. Solo se pueden añadir gastos del mes actual y hasta el día de hoy.



4.5. Manager - Manage

Una vez añadido, se nos abrirá la página de **manage**, donde se nos muestran todos nuestros gastos. Si recientemente hemos añadido uno, se nos indicará.

What you looking for?

Item: 'Móvil' added

#	Item	Description	Quantity	Bought On	Price (per item)	Action
1	Móvil	Móvil comprado en amazon	1	9/11/2020	150 \$	Delete

En la parte superior tenemos la opción de filtrar, según los productos que hayamos añadido.

localhost:8090/expenses-manager/expenses/manage

Expense Manager Home Manager Logout

por

Item: 'Portatio' added

#	Item	Description	Quantity	Bought On	Price (per item)	Action
2	Portatio	Portatio	1	17/11/2020	120 \$	Delete

La opción de **delete** borra el gasto en cuestión.

What you looking for?

Item: 'Portatio' deleted

#	Item	Description	Quantity	Bought On	Price (per item)	Action
1	Móvil	Móvil comprado en amazon	1	9/11/2020	150 \$	Delete

4.6. Logout

La opción de **logout** borra la sesión y las cookies del usuario actual y nos devuelve a la página de **login**.

5. Acceso a la documentación

Si así se desea, se puede acceder a la documentación de las clases. En la carpeta **doc** se encuentran los **javadoc** generados del proyecto.