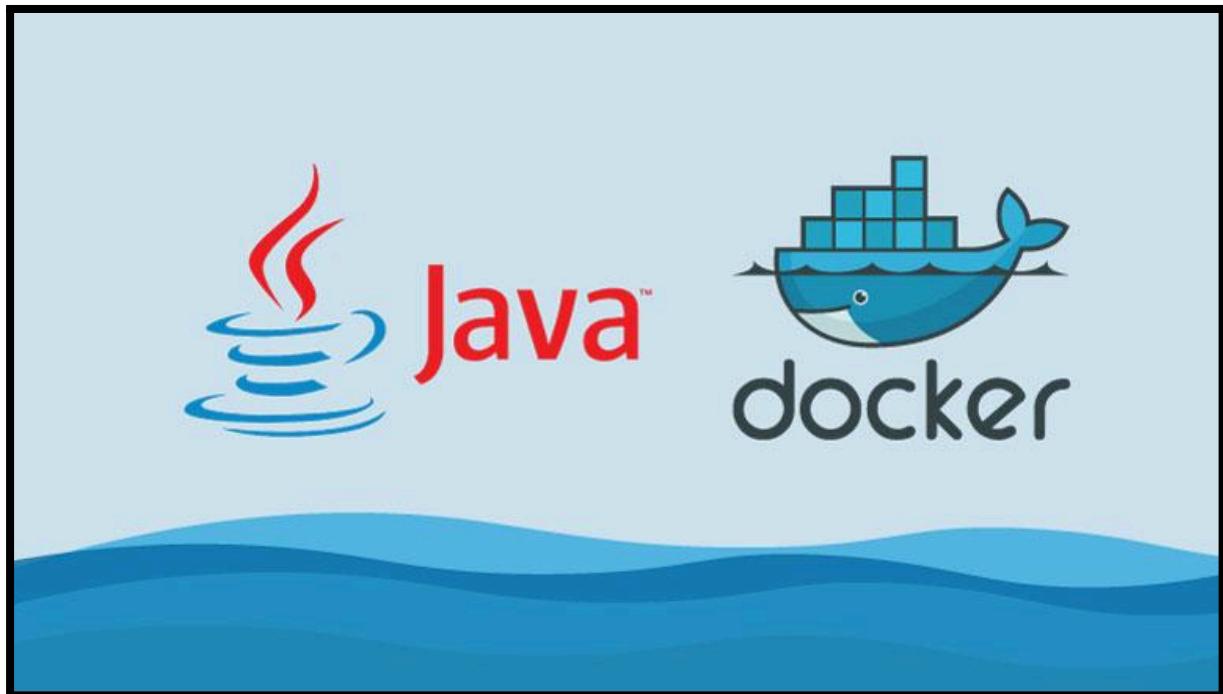


# *Despliegue de una mini aplicación*

## JAVA



<i>Nombre y Apellido</i>	Laura San Román
<i>Curso</i>	2º - DAW
<i>Fecha</i>	12 / 02 / 2026
<i>Asignatura</i>	Despliegue de aplicaciones web

## SERVIDOR WEB VIRTUALIZADO

### La estructura es:

Carpeta llamada servidor-java

- app
  - DockerFile
  - index.html
  - InsertarUsuarioServlet.java
  - mariadb-java-client-3.5.6.jar
- db
  - init.sql
- docker-compose.yml

Sabiendo la estructura que tiene, pasamos a la configuración para luego construir el contenedor.

### La arquitectura de IP es:

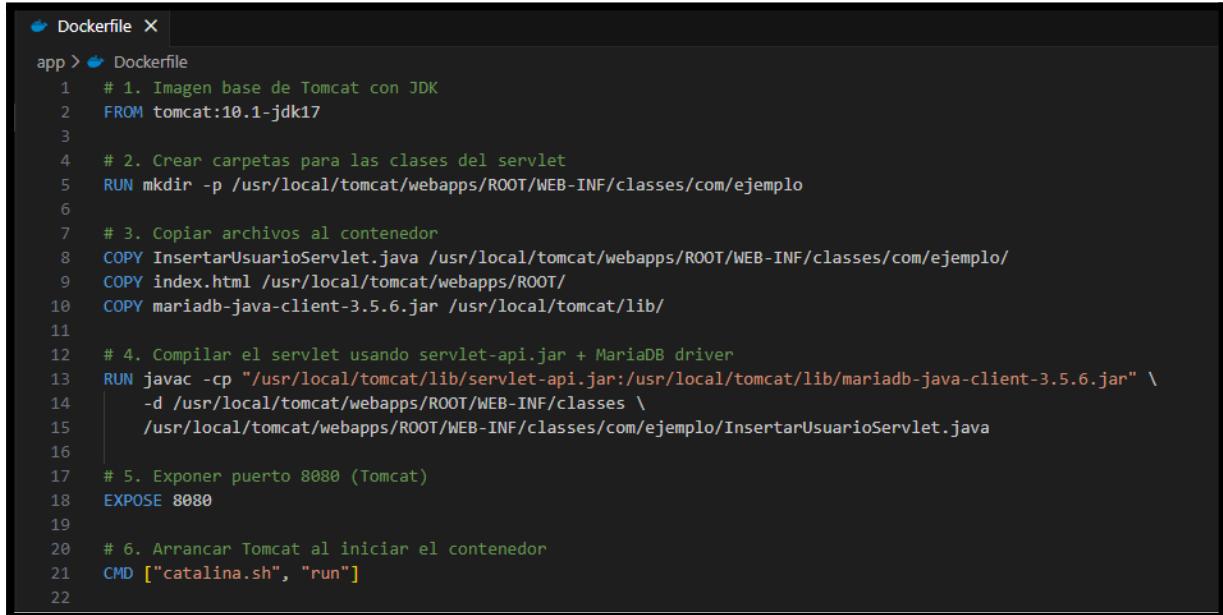
- Apache -> “8080:80”
- Tomcat-> “8081:80”
- MariaDB-> “3309:3306”
- PhpMyAdmin-> “5004:80”

## **CONFIGURACIÓN DE LA CARPETA APP**

**Paso 1.** Entrar a la cmd -> cd C:\Users\dawmi\servidor-java\app

**Paso 2.** Crear un .txt llamado DockerFile donde se guardará la siguiente información

->

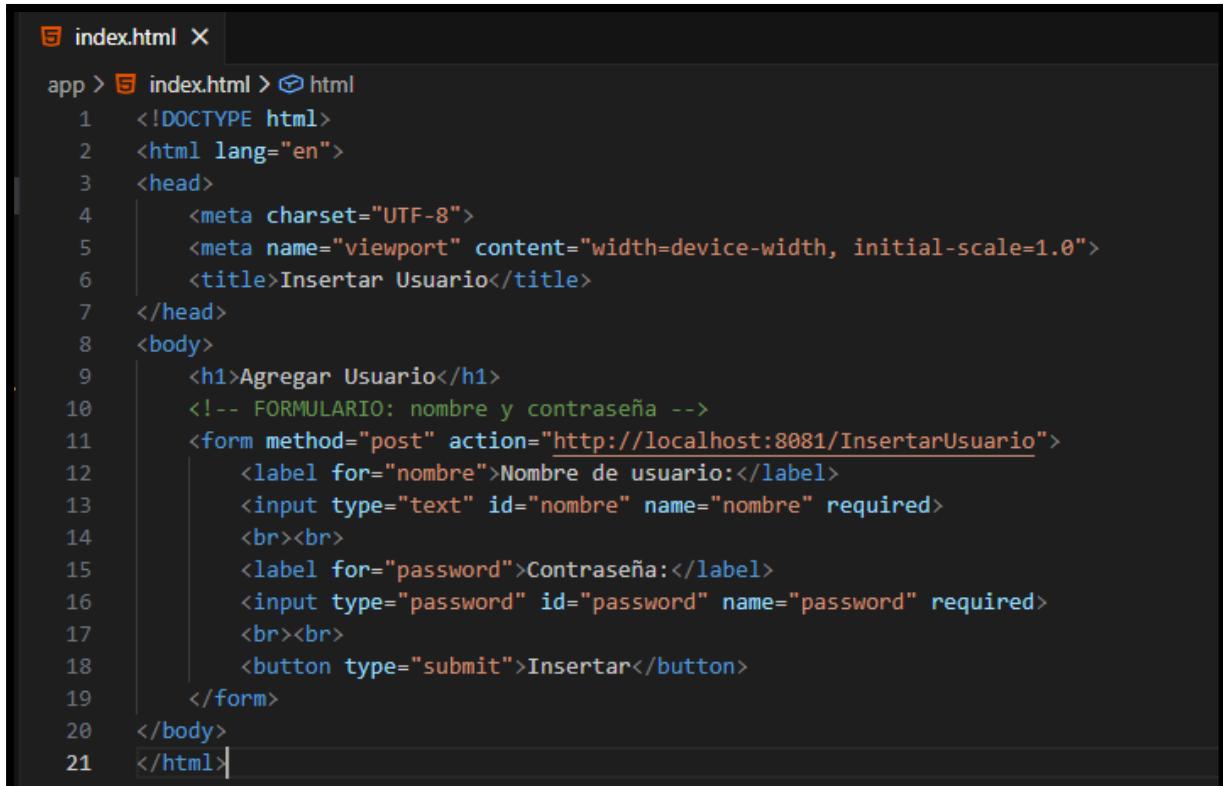


```
Dockerfile X
app > Dockerfile
1 # 1. Imagen base de Tomcat con JDK
2 FROM tomcat:10.1-jdk17
3
4 # 2. Crear carpetas para las clases del servlet
5 RUN mkdir -p /usr/local/tomcat/webapps/ROOT/WEB-INF/classes/com/ejemplo
6
7 # 3. Copiar archivos al contenedor
8 COPY InsertarUsuarioServlet.java /usr/local/tomcat/webapps/ROOT/WEB-INF/classes/com/ejemplo/
9 COPY index.html /usr/local/tomcat/webapps/ROOT/
10 COPY mariadb-java-client-3.5.6.jar /usr/local/tomcat/lib/
11
12 # 4. Compilar el servlet usando servlet-api.jar + MariaDB driver
13 RUN javac -cp "/usr/local/tomcat/lib/servlet-api.jar:/usr/local/tomcat/lib/mariadb-java-client-3.5.6.jar" \
14     -d /usr/local/tomcat/webapps/ROOT/WEB-INF/classes \
15     /usr/local/tomcat/webapps/ROOT/WEB-INF/classes/com/ejemplo/InsertarUsuarioServlet.java
16
17 # 5. Exponer puerto 8080 (Tomcat)
18 EXPOSE 8080
19
20 # 6. Arrancar Tomcat al iniciar el contenedor
21 CMD ["catalina.sh", "run"]
22
```

Y se guardará como \*todos los archivos para que no tenga extensión ya que sino, no funciona.

**Paso 3.** Construir el index.html

->



```
index.html X
app > index.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Insertar Usuario</title>
7  </head>
8  <body>
9      <h1>Agregar Usuario</h1>
10     <!-- FORMULARIO: nombre y contraseña -->
11     <form method="post" action="http://localhost:8081/InsertarUsuario">
12         <label for="nombre">Nombre de usuario:</label>
13         <input type="text" id="nombre" name="nombre" required>
14         <br><br>
15         <label for="password">Contraseña:</label>
16         <input type="password" id="password" name="password" required>
17         <br><br>
18         <button type="submit">Insertar</button>
19     </form>
20  </body>
21  </html>
```

#### Paso 4. Meter el código java (código del Aula Virtual) y añadir la parte de “password”

->

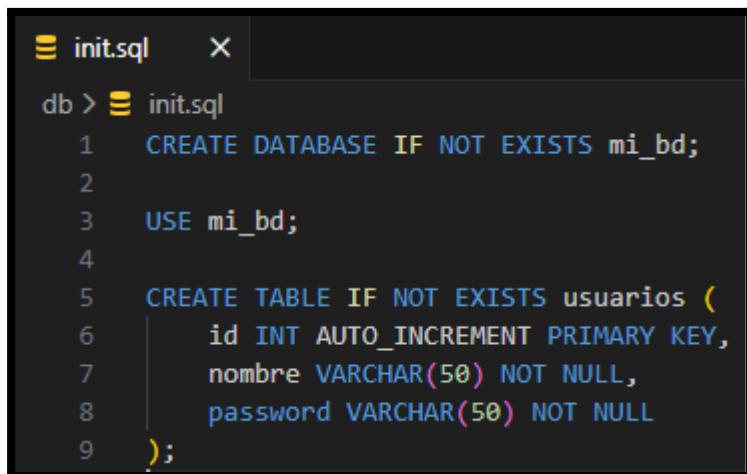
```
InsertarUsuarioServlet.java X
app > InsertarUsuarioServlet.java
1 package com.ejemplo;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.sql.Connection;
6 import java.sql.DriverManager;
7 import java.sql.PreparedStatement;
8 import java.sql.SQLException;
9
10 import jakarta.servlet.ServletException;
11 import jakarta.servlet.annotation.WebServlet;
12 import jakarta.servlet.http.HttpServlet;
13 import jakarta.servlet.http.HttpServletRequest;
14 import jakarta.servlet.http.HttpServletResponse;
15
16 @WebServlet("/InsertarUsuario")
17 public class InsertarUsuarioServlet extends HttpServlet {
18
19     // Configura estos valores según tu entorno
20     private static final String URL = "jdbc:mariadb://mariadb-daw:3306/mi_bd";
21     private static final String USER = "usuario";
22     private static final String PASS = "password";
23
24     @Override
25     protected void doPost(HttpServletRequest request, HttpServletResponse response)
26         throws ServletException, IOException {
27
28         request.setCharacterEncoding("UTF-8");
29         response.setContentType("text/html;charset=UTF-8");
30
31         // Recoger los parámetros del formulario
32         String nombre = request.getParameter("nombre");
33         String password = request.getParameter("password"); // NUEVO
34
35         try (PrintWriter out = response.getWriter()) {
36
37             // 1. Cargar el driver de MariaDB
38             try {
39                 Class.forName("org.mariadb.jdbc.Driver");
40             } catch (ClassNotFoundException e) {
41                 out.println("No se encontró el driver de MariaDB: " + e.getMessage());
42                 return;
43             }
44
45             // 2. Obtener conexión
46             try (Connection conn = DriverManager.getConnection(URL, USER, PASS)) {
47
48                 // 3. Instrucción SQL preparada para INSERT (nombre + password)
49                 String sql = "INSERT INTO usuarios (nombre, password) VALUES (?, ?)";
50
51                 try (PreparedStatement stmt = conn.prepareStatement(sql)) {
52                     stmt.setString(1, nombre);
53                     stmt.setString(2, password); // NUEVO
54
55                     int filas = stmt.executeUpdate(); // ejecuta INSERT
56
57                     if (filas > 0) {
58                         out.println("Usuario insertado correctamente.");
59                     } else {
60                         out.println("No se pudo insertar el usuario.");
61                     }
62                 }
63
64             } catch (SQLException e) {
65                 out.println("Error de base de datos: " + e.getMessage());
66             }
67         }
68     }
69 }
70 }
```

**Paso 5.** Descargar el .jar de mariadb y meterlo en la carpeta “app”  
-> [mariadb-java-client-3.5.6.jar](#)

## **CONFIGURACIÓN DE LA CARPETA DB**

**Paso 1.** Añadir el init.sql

->



```
init.sql      X
db > init.sql
1   CREATE DATABASE IF NOT EXISTS mi_bd;
2
3   USE mi_bd;
4
5   CREATE TABLE IF NOT EXISTS usuarios (
6       id INT AUTO_INCREMENT PRIMARY KEY,
7       nombre VARCHAR(50) NOT NULL,
8       password VARCHAR(50) NOT NULL
9   );
```

## CONFIGURACIÓN DEL DOCKER-COMPOSE.YML

Paso 1. El docker-compose.yml contiene lo siguiente

->

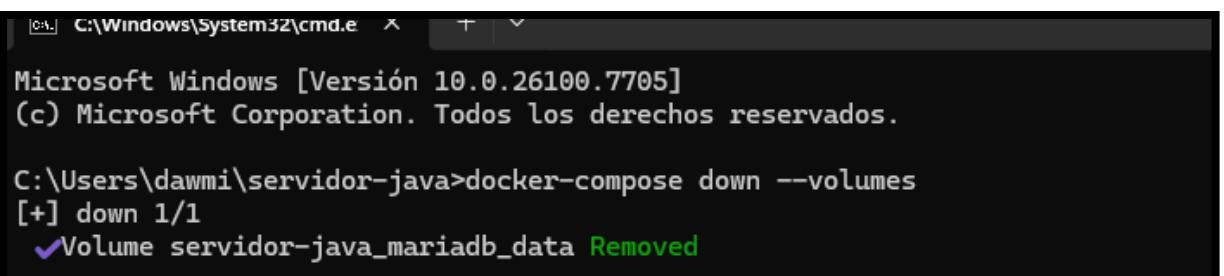
```
❶ docker-compose.yml X
❷ docker-compose.yml
❸
❹ services:
❺   apache-web:
❻     image: httpd:latest
❼     container_name: apache-web
❼     ports:
❼       - "8080:80"
❼     volumes:
❼       - ./app/index.html:/usr/local/apache2/htdocs/index.html
❼
❼   tomcat:
❼     build: ./app
❼     container_name: servidor-java-app
❼     ports:
❼       - "8081:8080"
❼     depends_on:
❼       - mariadb
❼
❼   mariadb:
❼     image: mariadb:latest
❼     container_name: mariadb-daw
❼     environment:
❼       MARIADB_ROOT_PASSWORD: root
❼       MARIADB_DATABASE: mi_bd
❼       MARIADB_USER: usuario
❼       MARIADB_PASSWORD: password
❼     ports:
❼       - "3309:3306"
❼     volumes:
❼       - mariadb_data:/var/lib/mysql
❼       - ./db:/docker-entrypoint-initdb.d
❼
❼   phpmyadmin:
❼     image: phpmyadmin/phpmyadmin
❼     container_name: phpmyadmin-java
❼     environment:
❼       PMA_HOST: mariadb-daw
❼       PMA_USER: usuario
❼       PMA_PASSWORD: password
❼     ports:
❼       - "5004:80"
❼     depends_on:
❼       - mariadb
❼
❼   volumes:
❼     mariadb_data:
```

## RESULTADOS

**Paso 1.** Desde la cmd de “servidor-java”. Haremos los dos siguiente comandos.

-> docker-compose down --volumes

Para eliminar configuraciones previas.



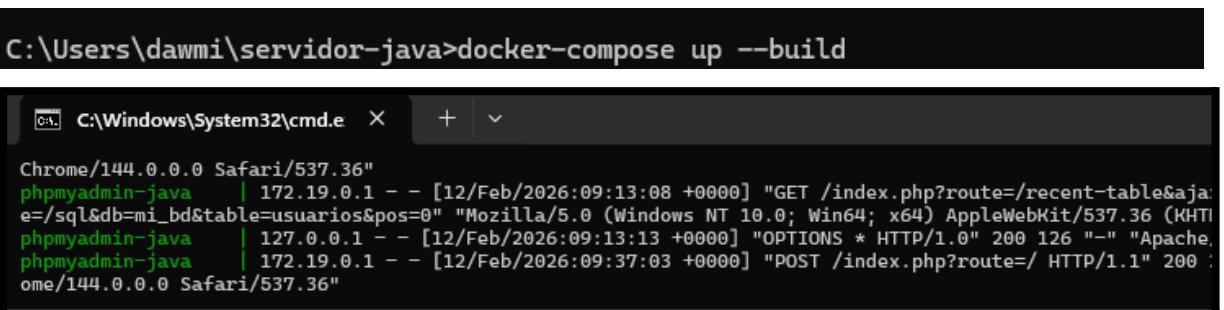
```
C:\Windows\System32\cmd.e Microsoft Windows [Versión 10.0.26100.7705]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\dawmi\servidor-java>docker-compose down --volumes
[+] down 1/1
✓ Volume servidor-java_mariadb_data Removed
```

**Paso 2.** Desde la cmd de “servidor-java”. Segundo comando.

-> docker-compose up --build

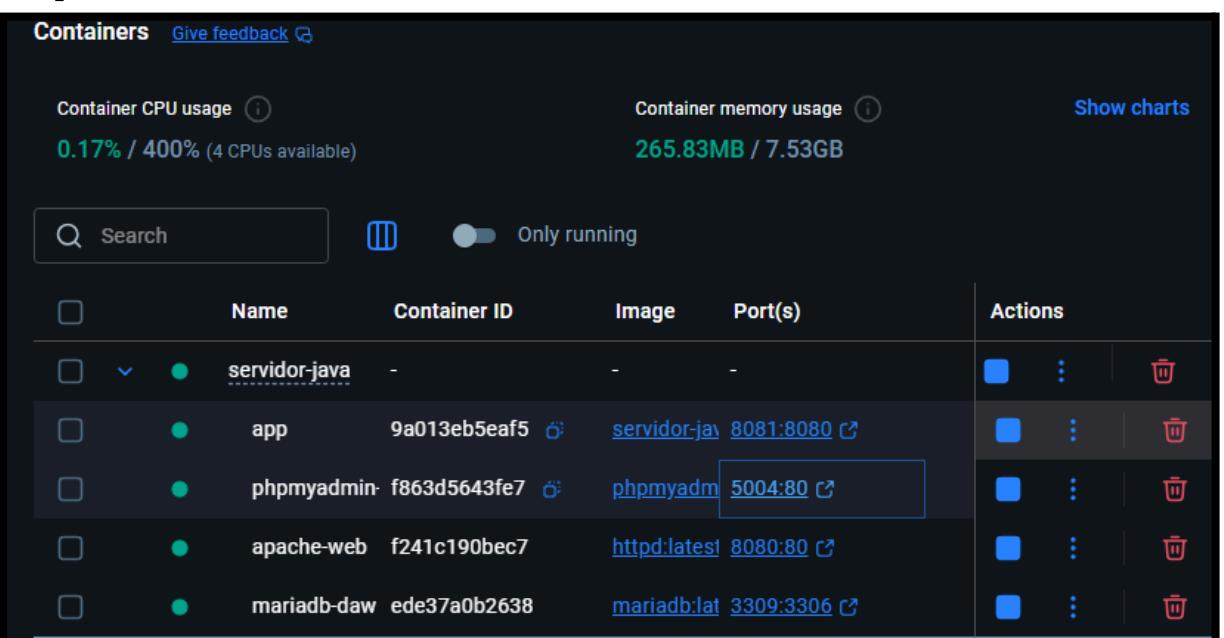
Para construir y levantararlo.



```
C:\Users\dawmi\servidor-java>docker-compose up --build
```

```
Chrome/144.0.0.0 Safari/537.36"
phpmyadmin-java | 172.19.0.1 - - [12/Feb/2026:09:13:08 +0000] "GET /index.php?route=/recent-table&ajax=true&sqlDb=mi_bd&table=usuarios&pos=0" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36"
phpmyadmin-java | 127.0.0.1 - - [12/Feb/2026:09:13:13 +0000] "OPTIONS * HTTP/1.0" 200 126 "-" "Apache/2.4.41 PHP/8.1.10"
phpmyadmin-java | 172.19.0.1 - - [12/Feb/2026:09:37:03 +0000] "POST /index.php?route=/ HTTP/1.1" 200 144.0.0.0 Safari/537.36"
```

Así quedaría el docker:



Containers		Give feedback		
		Container CPU usage	Container memory usage	
		0.17% / 400% (4 CPUs available)	265.83MB / 7.53GB	
Search		Only running	Show charts	
Name	Container ID	Image	Port(s)	Actions
servidor-java	-	-	-	<span>⋮</span> <span>⋮</span> <span>⋮</span>
app	9a013eb5eaf5	servidor-java	8081:8080	<span>⋮</span> <span>⋮</span> <span>⋮</span>
phpmyadmin	f863d5643fe7	phpmyadmin	5004:80	<span>⋮</span> <span>⋮</span> <span>⋮</span>
apache-web	f241c190bec7	httpd:latest	8080:80	<span>⋮</span> <span>⋮</span> <span>⋮</span>
mariadb-daw	ede37a0b2638	mariadb:lat	3309:3306	<span>⋮</span> <span>⋮</span> <span>⋮</span>

Al abrir el index.html

->

The screenshot shows a web browser window with three tabs. The active tab is titled 'Insertar Usuario' and has the URL 'localhost:8081'. The browser address bar also displays 'localhost:8081'. The page content is titled 'Agregar Usuario' and contains two input fields: 'Nombre de usuario:' with the value 'usuario1' and 'Contraseña:' with the value '.....'. A 'Insertar' button is present below the fields.

->

The screenshot shows a web browser window with three tabs. The active tab is titled 'localhost:8081/InsertarUsuario' and has the URL 'localhost:8081/InsertarUsuario'. The browser address bar also displays 'localhost:8081/InsertarUsuario'. The page content displays the message 'Usuario insertado correctamente.'

Y esto quedaría reflejado en la base de datos.

->

The screenshot shows the phpMyAdmin interface for the 'mi\_bd' database. The 'usuarios' table is selected, displaying two rows of data:

		id	nombre	password
<input type="checkbox"/>	<input type="checkbox"/> Editar	1	laura	123
<input type="checkbox"/>	<input type="checkbox"/> Editar	2	usuario1	usuario1

Below the table, there are buttons for 'Seleccionar todo', 'Para los elementos que están marcados:', and 'Operaciones sobre los resultados de la consulta' (Print, Copy to clipboard, Export, Show graphic, Create view).

## VERIFICACIÓN DEL ENTORNO

**Paso Final.** Para comprobar que ha sido desplegada en mi entorno local (mi ordenador).

->

