



TECNOLOGIAS WEB

Presenta :

- Avila Ibarra Andres
- Castillo Chávez Gerardo
- Castillo Lozada Norma Evelyn
- Paredes Lazcano Janet

Mayo-Agosto 2024

UNA UNIVERSIDAD PARA LA INVESTIGACIÓN



Objetivo

- Proporcionar una comprensión integral de las tecnologías web, la arquitectura cliente-servidor y el modelo de diseño MVC (Modelo-Vista-Controlador).

Introducción

- En la era digital actual, la web se ha convertido en una plataforma fundamental para la entrega de servicios, aplicaciones y contenidos a nivel mundial. Detrás de cada aplicación web exitosa, existe una combinación de tecnologías y arquitecturas que permiten su funcionamiento eficiente y su capacidad de respuesta ante las demandas de los usuarios.

Figura 1. Proceso de acondicionamiento de señal

Tecnologías Web

- Son un conjunto de herramientas y estándares que permiten la creación y el funcionamiento de sitios web y aplicaciones web.

Abarcan desde lenguajes de programación hasta protocolos de comunicación y frameworks de desarrollo.



Figura 1. Tecnologías Web

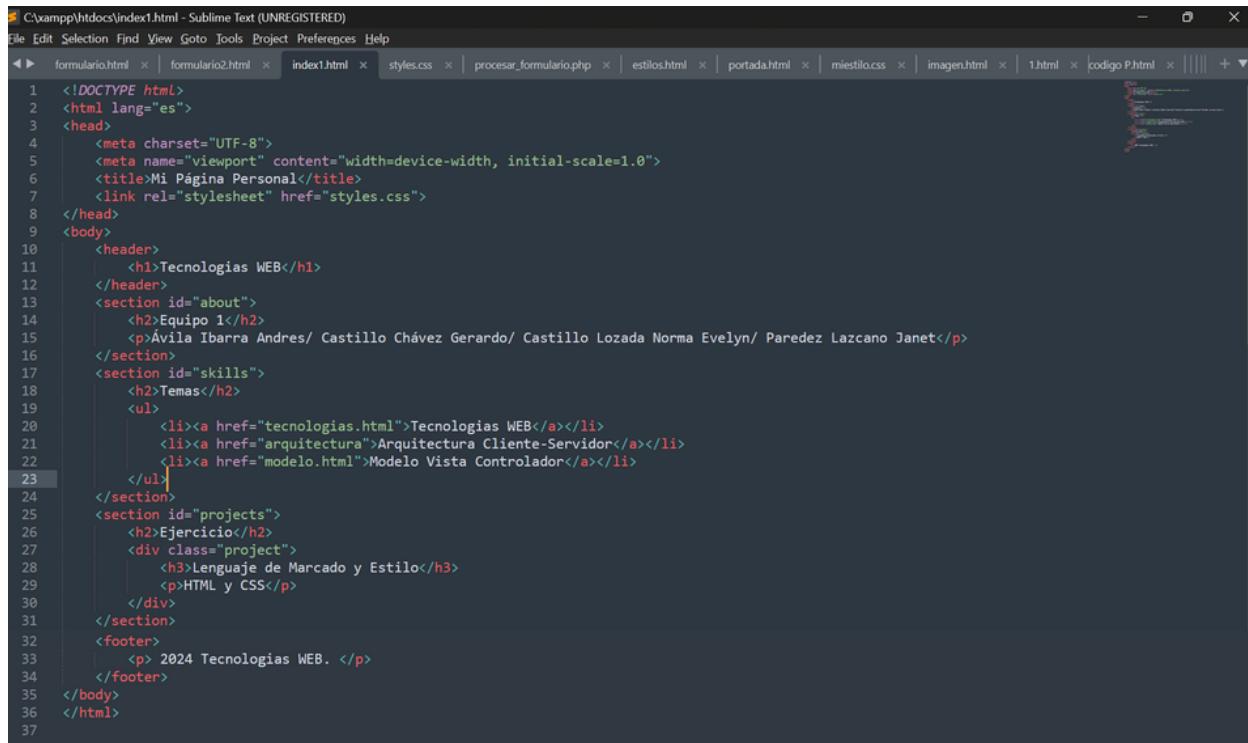


Lenguajes de Marcado y Estilo

- **HTML (HyperText Markup Language):** Es el lenguaje de marcado principal utilizado para crear la estructura de las páginas web. Define elementos como encabezados, párrafos, enlaces, imágenes, etc.
- **CSS (Cascading Style Sheets):** Es el lenguaje utilizado para describir la presentación de un documento HTML. Permite aplicar estilos como colores, fuentes, espaciados y diseño en general.

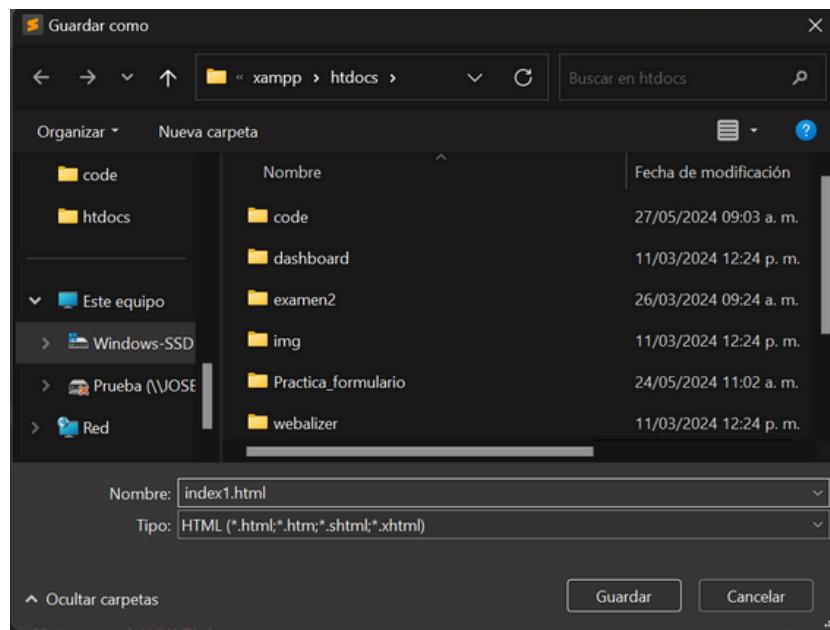
Ejercicio 1:

- Paso 1.
Crear un formulario en html y guardarlo en una carpeta dentro de Xampp/htdocs.



The screenshot shows a Sublime Text window with multiple tabs open, including 'formulario.html' which is the active tab. The code in 'formulario.html' is as follows:

```
1 <!DOCTYPE html>
2 <html lang="es">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Mi Página Personal</title>
7     <link rel="stylesheet" href="styles.css">
8   </head>
9   <body>
10    <header>
11      <h1>Tecnologías WEB</h1>
12    </header>
13    <section id="about">
14      <h2>Equipo 1</h2>
15      <p>Ávila Ibarra Andres/ Castillo Chávez Gerardo/ Castillo Lozada Norma Evelyn/ Paredez Lazcano Janet</p>
16    </section>
17    <section id="skills">
18      <h2>Temas</h2>
19      <ul>
20        <li><a href="tecnologias.html">Tecnologías WEB</a></li>
21        <li><a href="arquitectura">Arquitectura Cliente-Servidor</a></li>
22        <li><a href="modelo.html">Modelo Vista Controlador</a></li>
23      </ul>
24    </section>
25    <section id="projects">
26      <h2>Ejercicio:</h2>
27      <div class="project">
28        <h3>Lenguaje de Marcado y Estilo</h3>
29        <p>HTML y CSS</p>
30      </div>
31    </section>
32    <footer>
33      <p> 2024 Tecnologías WEB. </p>
34    </footer>
35  </body>
36 </html>
37
```



Paso 2.

Crear un segundo archivo .css y guardarlo en una carpeta dentro de Xampp/htdocs.

```
File Edit Selection Find View Goto Tools Project Preferences Help
formulario.html x | formulario2.html x | index1.html x | styles.css x | formulario.php x |||| + ▾
1 /* Reset de márgenes y padding */
2 * {
3     margin: 0;
4     padding: 0;
5     box-sizing: border-box;
6 }
7
8 /* Estilos para el cuerpo */
9 body {
10    font-family: Arial, sans-serif;
11    line-height: 1.6;
12    background-color: #f4f4f4;
13    color: #333;
14    padding: 20px;
15 }
16
17 /* Estilos del encabezado */
18 header {
19    background: #333;
20    color: #fff;
21    padding: 10px 0;
22    text-align: center;
23 }
24
25 /* Estilos de las secciones */
26 section {
27    margin-bottom: 20px;
28 }
```

```
30 h2 {
31     border-bottom: 2px solid #333;
32     padding-bottom: 5px;
33     margin-bottom: 10px;
34 }
35
36 /* Estilos para la lista de habilidades */
37 ul {
38     list-style: none;
39     padding-left: 0;
40 }
41
42 ul li {
43     background: #fff;
44     margin-bottom: 5px;
45     padding: 10px;
46     border-left: 4px solid #333;
47 }
48
49 /* Estilos para los proyectos */
50 .project {
51     background: #fff;
52     margin-bottom: 10px;
53     padding: 10px;
54     border-left: 4px solid #333;
55 }
56
57 /* Estilos del pie de página */
58 footer {
59     text-align: center;
60     padding: 10px 0;
61     background: #333;
62     color: #fff;
63 }
64 }
```



Resultados.

Tecnologías WEB

Equipo 1

Ávila Ibarra Andres/ Castillo Chávez Gerardo/ Castillo Lozada Norma Evelyn/ Paredes Lazcano Janet

Temas

- [Tecnologías WEB](#)
- [Arquitectura Cliente-Servidor](#)
- [Modelo Vista Controlador](#)

Ejercicio

Lenguaje de Marcado y Estilo
HTML y CSS

2024 Tecnologías WEB.



Lenguajes de Programación

- **Lenguajes del lado del cliente:**

1. **JavaScript:** Es un lenguaje de programación esencial para el desarrollo web que permite crear contenido dinámico y interactivo en los sitios web.
2. **TypeScript:** Es un superconjunto de JavaScript que agrega tipos estáticos. Se compila a JavaScript y se utiliza para mejorar la escalabilidad y mantenibilidad del código JavaScript.
3. **HTML (HyperText Markup Language):** Aunque técnicamente no es un lenguaje de programación, es fundamental para la estructura de las páginas web. Define el contenido y la estructura de una página web.
4. **CSS (Cascading Style Sheets):** Tampoco es un lenguaje de programación, pero es crucial para la presentación y el diseño de las páginas web.



5. **WebAssembly:** Es un lenguaje de bajo nivel que puede ejecutarse en navegadores web. Permite ejecutar código de lenguajes como C, C++ y Rust en el navegador, proporcionando un rendimiento cercano al nativo.

- **Lenguajes del lado del servidor:**

1. **PHP:** Es un lenguaje ampliamente utilizado para el desarrollo de aplicaciones web. Se integra fácilmente con bases de datos y servidores web.
2. **Python:** Utilizado en el desarrollo web a través de frameworks como Django y Flask. Es conocido por su sintaxis clara y su amplio ecosistema de bibliotecas.
3. **Ruby:** Utilizado principalmente con el framework Ruby on Rails. Es conocido por su simplicidad y productividad.

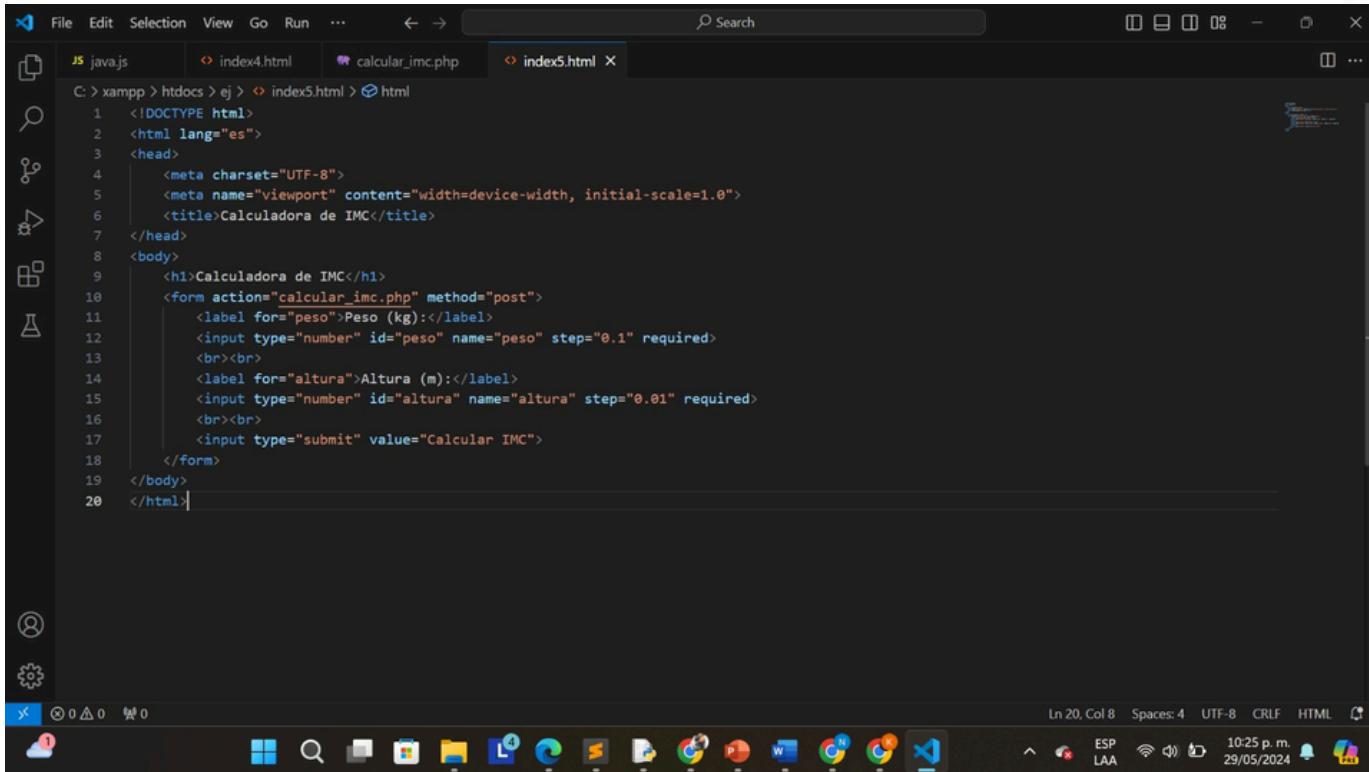


4. **Java:** Utilizado en el desarrollo web con frameworks como Spring. Es un lenguaje robusto y escalable.
5. **C#:** Utilizado con el framework .NET de Microsoft para el desarrollo de aplicaciones web y de escritorio.
6. **Node.js:** Aunque JavaScript se ejecuta tradicionalmente en el navegador, Node.js permite ejecutar JavaScript en el servidor. Es conocido por su alta eficiencia y capacidad para manejar grandes cantidades de conexiones simultáneas.
7. **Perl:** Aunque ha perdido popularidad, aún se usa en algunas aplicaciones web heredadas y scripts del lado del servidor.
8. **Go:** Un lenguaje desarrollado por Google que se destaca por su rendimiento y concurrencia. Es cada vez más popular para el desarrollo de servicios web.

Ejercicio 2:

- **Paso 1.**

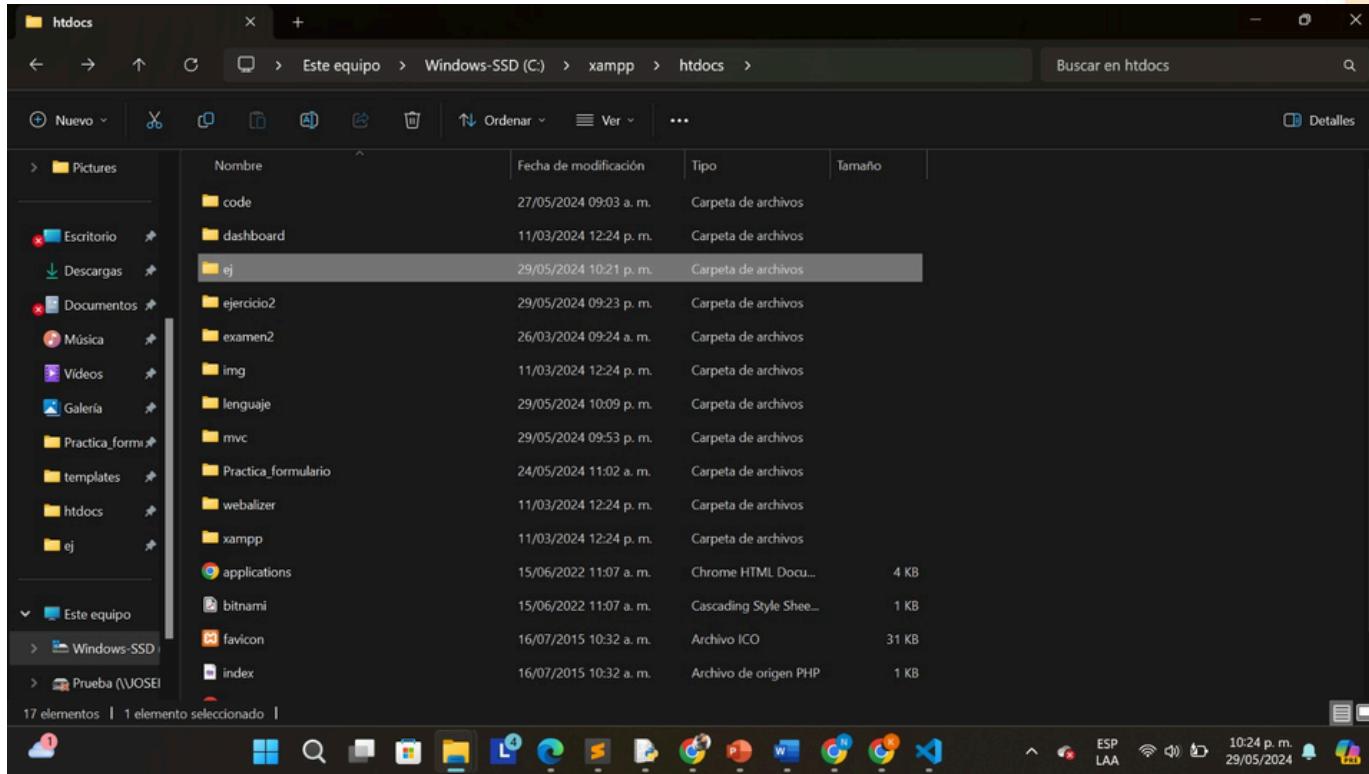
Crear un calculador del indice de masa corporal de una persona, dentro carpeta creada en xampp/htdocs llamada ej donde se guardara el index.html



```
File Edit Selection View Go Run ... ⏪ ⏩ Search
JS java.js index4.html calcular_imc.php index5.html
C:\xampp\htdocs\ej>index5.html
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Calculadora de IMC</title>
7  </head>
8  <body>
9      <h1>Calculadora de IMC</h1>
10     <form action="calcular_imc.php" method="post">
11         <label for="peso">Peso (kg):</label>
12         <input type="number" id="peso" name="peso" step="0.1" required>
13         <br><br>
14         <label for="altura">Altura (m):</label>
15         <input type="number" id="altura" name="altura" step="0.01" required>
16         <br><br>
17         <input type="submit" value="Calcular IMC">
18     </form>
19 </body>
20 </html>
```

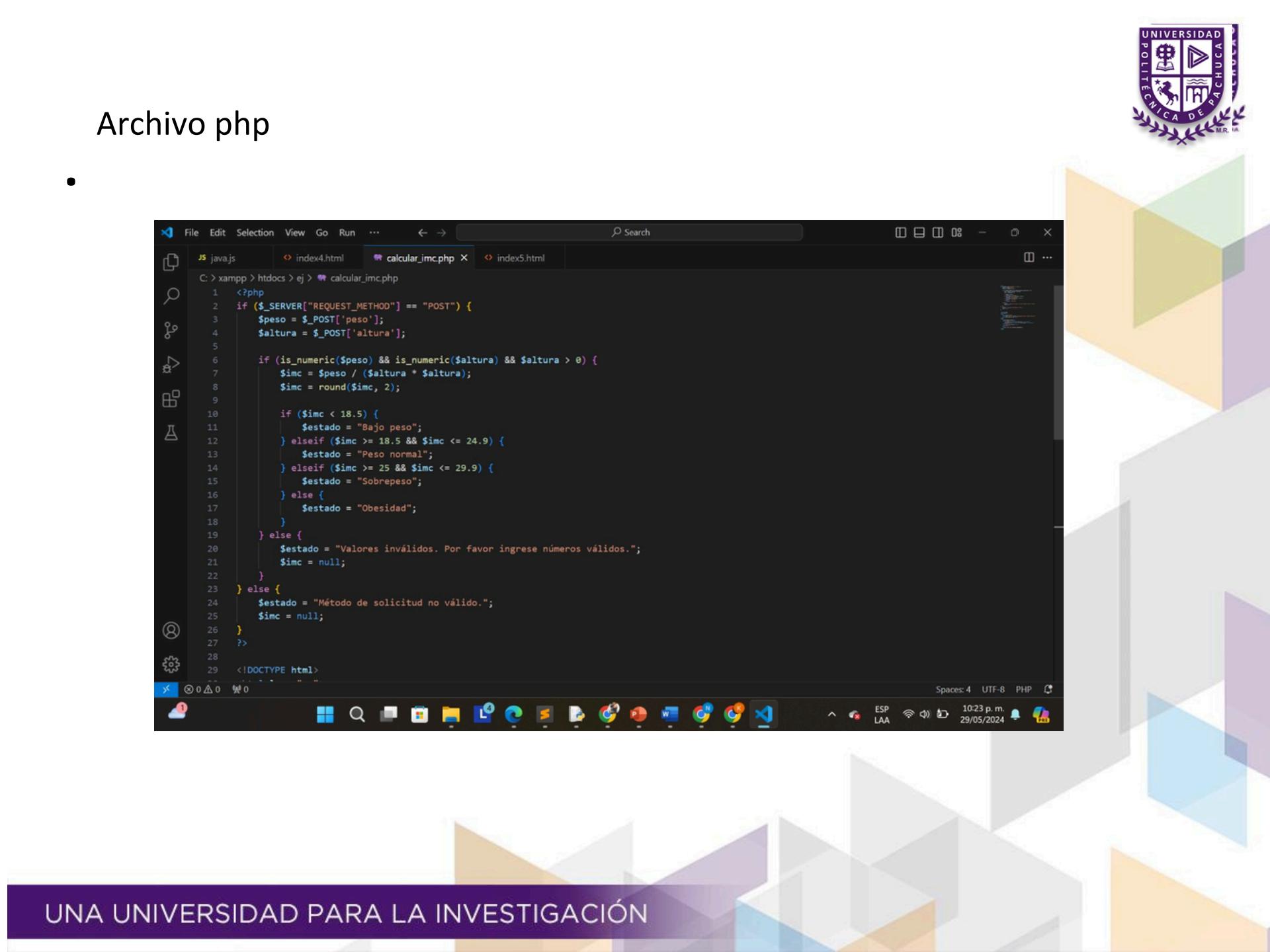
Paso 2.

Crear un archivo en php y guardarlo dentro de la misma carpeta





Archivo php



```
File Edit Selection View Go Run ... ← → ⌂ Search
C: > xampp > htdocs > ej > calcular_imc.php
1  <?php
2  if ($_SERVER["REQUEST_METHOD"] == "POST") {
3      $peso = $_POST['peso'];
4      $altura = $_POST['altura'];
5
6      if (is_numeric($peso) && is_numeric($altura) && $altura > 0) {
7          $imc = $peso / ($altura * $altura);
8          $imc = round($imc, 2);
9
10         if ($imc < 18.5) {
11             $estado = "Bajo peso";
12         } elseif ($imc >= 18.5 && $imc <= 24.9) {
13             $estado = "Peso normal";
14         } elseif ($imc >= 25 && $imc <= 29.9) {
15             $estado = "Sobrepeso";
16         } else {
17             $estado = "Obesidad";
18         }
19     } else {
20         $estado = "Valores inválidos. Por favor ingrese números válidos.";
21         $imc = null;
22     }
23 } else {
24     $estado = "Método de solicitud no válido.";
25     $imc = null;
26 }
27 ?>
28
29 <!DOCTYPE html>
Spaces: 4  UTF-8  PHP
10:23 p.m.
ESP LAA 29/05/2024
```



Resultados

Calculadora de IMC

Peso (kg):

Altura (m):

[Calcular IMC](#)

Resultado del IMC

Su IMC es: **24.62**

Estado de peso: **Peso normal**

[Calcular nuevamente](#)

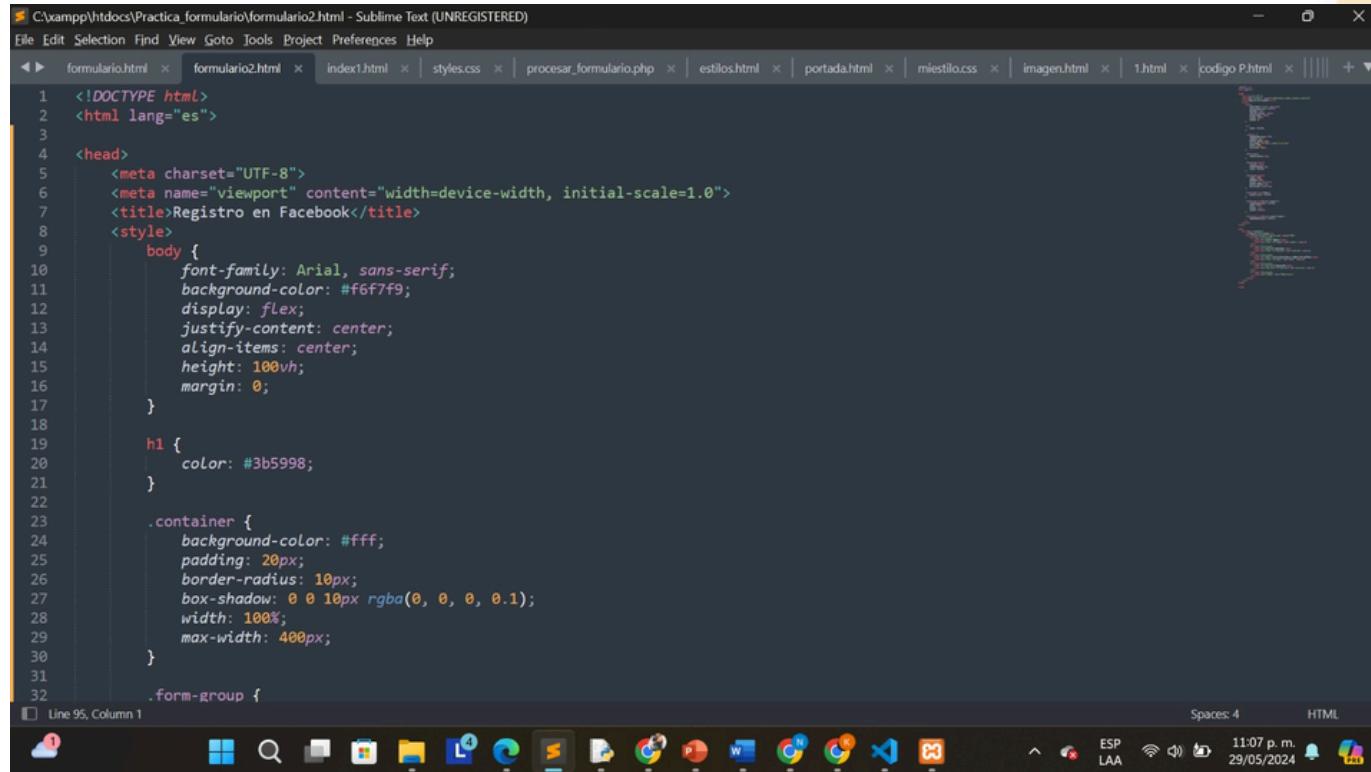


Bases de Datos

- **SQL (Structured Query Language):** Utilizado para manejar bases de datos relacionales como MySQL, PostgreSQL, etc.
- **NoSQL:** Bases de datos no relacionales como MongoDB, CouchDB, etc., que son útiles para datos no estructurados o semi-estructurados.

Ejercicio 3:

Paso 1: Crear un formulario en HTML llamado formulario2.html



```
C:\xampp\htdocs\Practica_formulario\formulario2.html - Sublime Text (UNREGISTERED)
File Edit Selection Find Goto Tools Project Preferences Help
formulario.html x formulario2.html x index1.html x styles.css x procesar_formulario.php x estilos.html x portada.html x miestilo.css x imagen.html x 1.html x codigo_P.html x + | |
1 <!DOCTYPE html>
2 <html lang="es">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Registro en Facebook</title>
8   <style>
9     body {
10       font-family: Arial, sans-serif;
11       background-color: #f6f7f9;
12       display: flex;
13       justify-content: center;
14       align-items: center;
15       height: 100vh;
16       margin: 0;
17     }
18
19     h1 {
20       color: #3b5998;
21     }
22
23     .container {
24       background-color: #fff;
25       padding: 20px;
26       border-radius: 10px;
27       box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
28       width: 100%;
29       max-width: 400px;
30     }
31
32     .form-group {
Line 95, Column 1
Spaces: 4
HTML
? S D L E P G W C N X ^ F ESP LAA 11:07 p. m. 29/05/2024 B C D E
```



C:\xampp\htdocs\Practica_formulario\formulario2.html - Sublime Text (UNREGISTERED)

File Edit Selection Find Goto Tools Project Preferences Help

```
< > formulario.html x formulario2.html x index1.html x styles.css x procesar_formulario.php x estilos.html x portada.html x mienstilo.css x imagen.html x 1.html x codigo P.html x |||
```

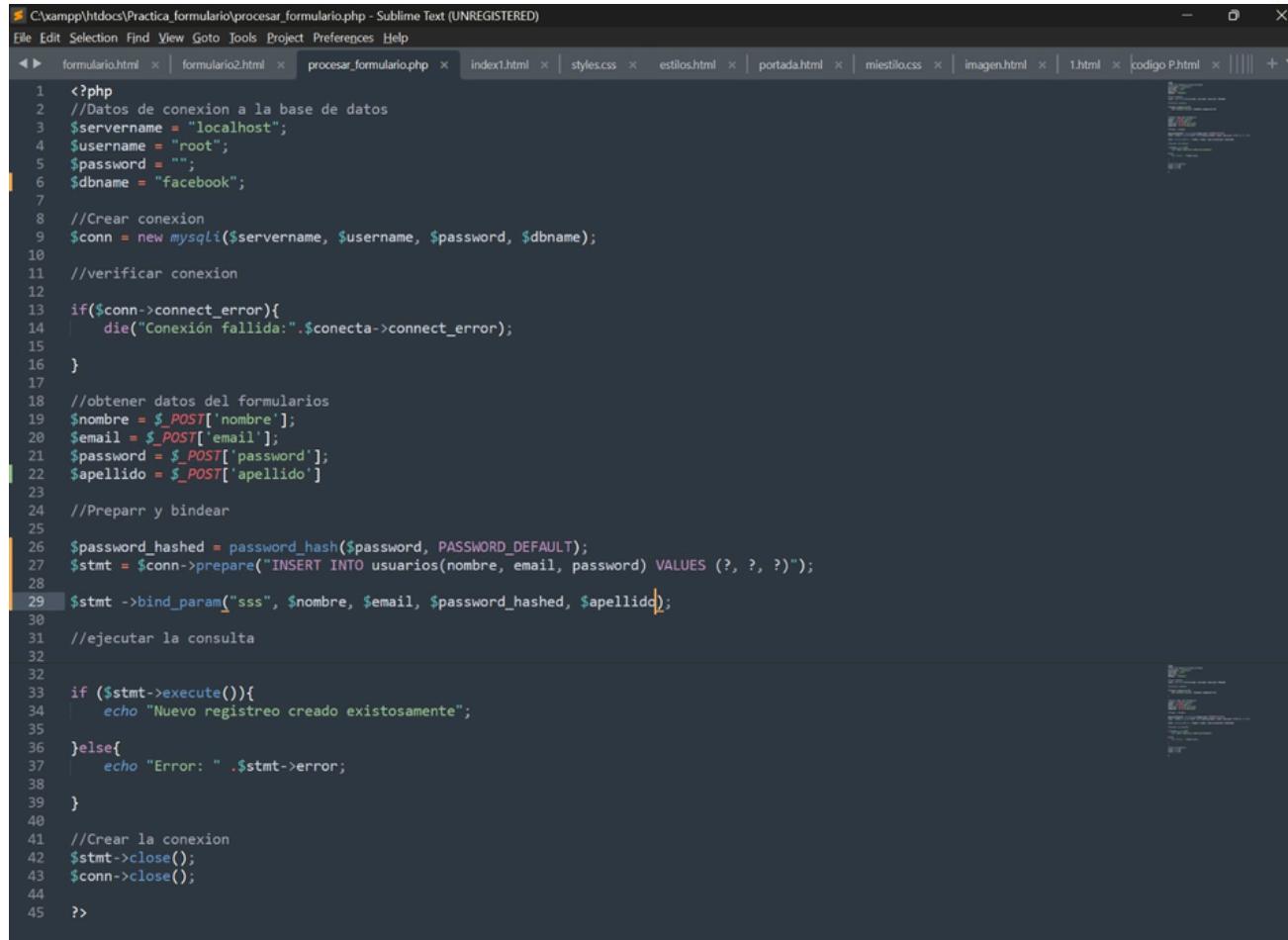
32 .form-group {
33 margin-bottom: 15px;
34 }
35
36 .form-group label {
37 display: block;
38 margin-bottom: 5px;
39 color: #3b5998;
40 }
41
42 .form-group input {
43 width: 100%;
44 padding: 10px;
45 border: 1px solid #ccc;
46 border-radius: 5px;
47 box-sizing: border-box;
48 }
49
50 .form-group input:focus {
51 border-color: #3b5998;
52 }
53
54 .form-group input[type="submit"] {
55 background-color: #3b5998;
56 color: #fff;
57 border: none;
58 cursor: pointer;
59 }
60
61 .form-group input[type="submit"]:hover {
62 background-color: #2d4373;



```
C:\xampp\htdocs\Practica_formulario\formulario2.html - Sublime Text (UNREGISTERED)
File Edit Selection Find Goto Tools Project Preferences Help
formulario.html x formulario2.html x index1.html x styles.css x procesar_formulario.php x estilos.html x portada.html x miestilo.css x imagen.html x 1.html x codigo P.html x ||| | |
63     }
64   </style>
65 </head>
66
67 <body>
68   <div class="container">
69     <h1>Registro en Facebook</h1>
70     <form action="procesar_registro.php" method="POST">
71       <div class="form-group">
72         <label for="nombre">Nombre</label>
73         <input type="text" id="nombre" name="nombre" required>
74       </div>
75       <div class="form-group">
76         <label for="apellido">Apellido</label>
77         <input type="text" id="apellido" name="apellido" required>
78       </div>
79       <div class="form-group">
80         <label for="email">Correo electrónico o número de teléfono</label>
81         <input type="text" id="email" name="email" required>
82       </div>
83       <div class="form-group">
84         <label for="password">Contraseña</label>
85         <input type="password" id="password" name="password" required>
86       </div>
87       <div class="form-group">
88         <input type="submit" value="Registrarse">
89       </div>
90     </form>
91   </div>
92 </body>
93
94 </html>
```

Paso 2:

Crear un archivo en php y guardalo en una carpeta creada en xampp/htdocs



The screenshot shows a Sublime Text window with multiple tabs open. The active tab contains PHP code for creating a new user in a MySQL database. The code includes connecting to a local MySQL server, preparing and executing an INSERT query with hashed passwords, and echoing a success or error message. The code is as follows:

```
<?php
//Datos de conexion a la base de datos
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "facebook";

//Crear conexion
$conn = new mysqli($servername, $username, $password, $dbname);

//verificar conexion
if($conn->connect_error){
    die("Conexión fallida:". $conn->connect_error);
}

//obtener datos del formularios
$nombre = $_POST['nombre'];
$email = $_POST['email'];
$password = $_POST['password'];
$apellido = $_POST['apellido']

//Preparar y bindear
$password_hashed = password_hash($password, PASSWORD_DEFAULT);
$stmt = $conn->prepare("INSERT INTO usuarios(nombre, email, password) VALUES (?, ?, ?)");
$stmt ->bind_param("sss", $nombre, $email, $password_hashed, $apellido);

//ejecutar la consulta

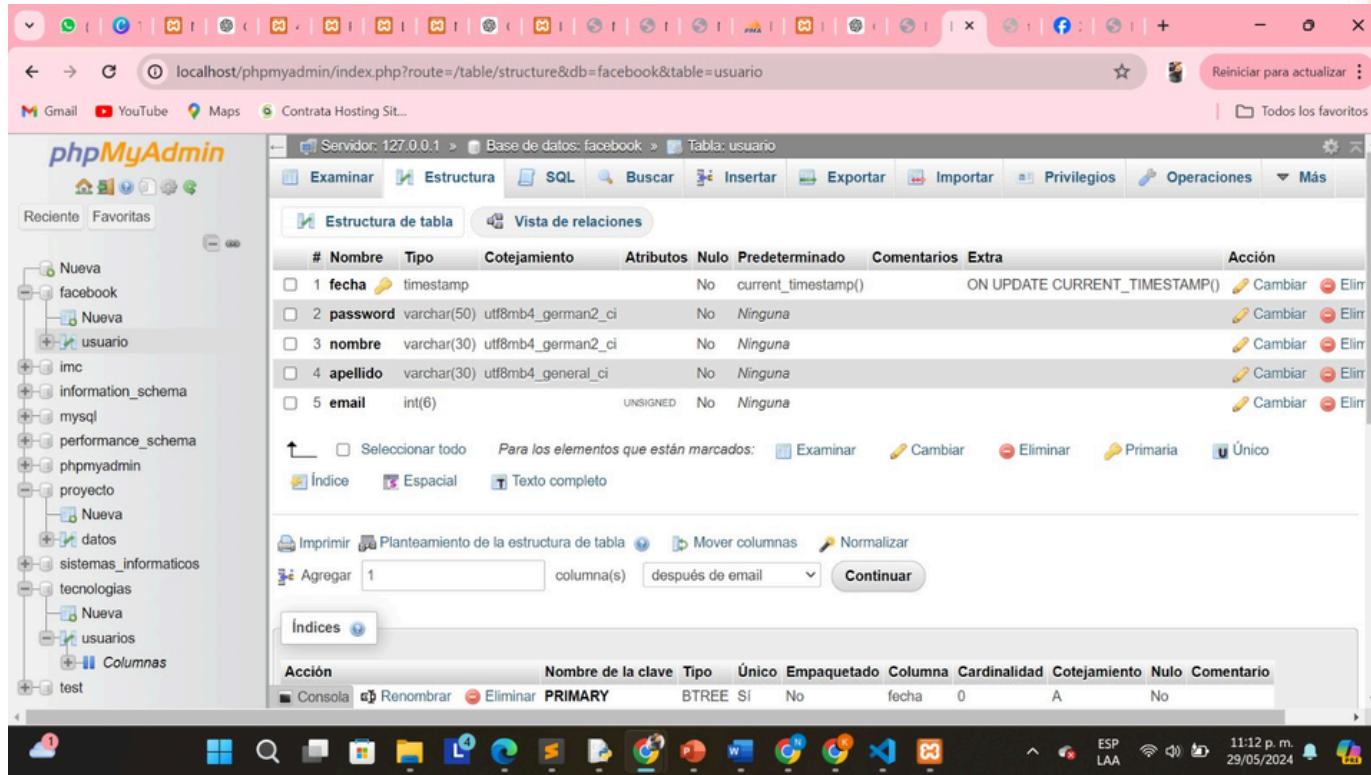
if ($stmt->execute()){
    echo "Nuevo registro creado exitosamente";
} else{
    echo "Error: " . $stmt->error;
}

//Crear la conexion
$stmt->close();
$conn->close();

?>
```

Paso 3:

- Crea una base de datos en SQL, al igual que crea una tabla llamada usuario con las variables asignadas en html.



The screenshot shows the phpMyAdmin interface for the 'facebook' database. The 'Estructura de tabla' (Table Structure) tab is selected, displaying the columns of the 'usuario' table. The columns are:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	fecha	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Cambiar Eliminar
2	password	varchar(50)	utf8mb4_german2_ci		No	Ninguna			Cambiar Eliminar
3	nombre	varchar(30)	utf8mb4_german2_ci		No	Ninguna			Cambiar Eliminar
4	apellido	varchar(30)	utf8mb4_general_ci		No	Ninguna			Cambiar Eliminar
5	email	int(6)		UNSIGNED	No	Ninguna			Cambiar Eliminar

Below the table structure, there is a section for adding a new column ('Agregar') and a section for managing indices ('Índices').



- Resultado

Registro en Facebook

Nombre

Apellido

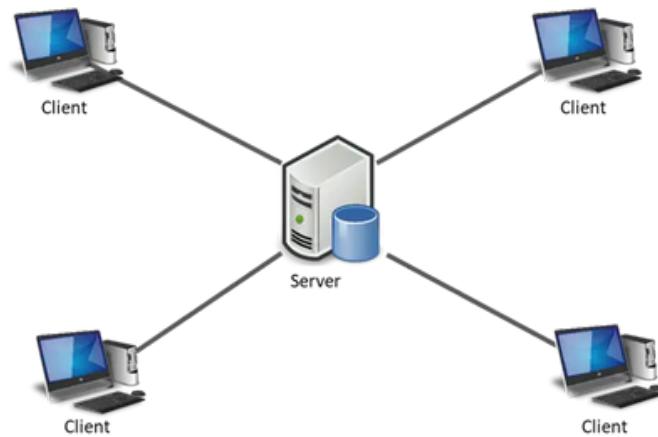
Correo electrónico o número de teléfono

Contraseña

Registrarse

Arquitectura Cliente-Servidor

- Modelo de diseño de sistemas distribuidos en el cual los componentes se dividen en clientes y servidores. Esta arquitectura es fundamental para muchas aplicaciones en red, incluyendo la web, servicios de correo electrónico, bases de datos, etc.





Arquitectura Cliente-Servidor

- **Cliente:** El cliente es un dispositivo conectado a una red, cuya función es realizar peticiones al servidor. Estas peticiones pueden ser consultas o solicitudes de servicios, tales como la descarga de archivos o la transmisión de datos. Normalmente, los clientes se encuentran en dispositivos como PCs, teléfonos móviles y tabletas, entre otros.
- **Servidor:** El servidor es el encargado de responder a las peticiones del cliente y proporcionar los servicios que se le hayan solicitado. Un servidor también puede almacenar y administrar información para su posterior acceso por parte del usuario. Suelen trabajar con sistemas operativos como Linux o Windows Server.



Flujo de Trabajo

- **Solicitud del Cliente:** El cliente inicia una solicitud enviando un mensaje al servidor. Esta solicitud puede ser, por ejemplo, una petición de una página web, datos de una base de datos, etc.
- **Procesamiento del Servidor:** El servidor recibe la solicitud, la procesa y prepara una respuesta. El procesamiento puede incluir la consulta a una base de datos, la ejecución de lógica de negocio, etc.
- **Respuesta del Servidor:** El servidor envía la respuesta al cliente. Esta respuesta puede incluir HTML, archivos, etc.
- **Recepción del Cliente:** El cliente recibe la respuesta y la procesa según sea necesario, como mostrar una página web en un navegador o actualizar una interfaz de usuario en una aplicación.

Tipos de arquitectura cliente-servidor

- **Arquitectura de dos niveles (2-Tier):** Es un modelo de diseño de software en el que una aplicación se divide en dos capas o niveles principales.
 1. **Capa de Cliente (Client Tier):** Esta capa es la interfaz de usuario de la aplicación, donde los usuarios interactúan directamente con el sistema.
 2. **Capa de Servidor (Server Tier):** Esta capa gestiona la lógica de negocio y el acceso a los datos. Se encuentra en un servidor al que los clientes se conectan para realizar operaciones. Incluye la base de datos y, a menudo, la lógica del servidor que procesa las solicitudes del cliente.



Ventajas de la arquitectura 2-Tier

- **Simplicidad:** Es más simple de implementar y entender en comparación con arquitecturas más complejas como la de tres niveles (3-Tier).
- **Rendimiento:** Puede ser más rápida ya que hay menos capas de comunicación entre el cliente y el servidor.

Desventajas de la arquitectura 2-Tier:

- **Escalabilidad:** Puede tener problemas de escalabilidad, especialmente si muchos clientes están accediendo al servidor simultáneamente.
- **Mantenimiento:** Las actualizaciones y cambios pueden ser más complicados de gestionar, ya que cualquier cambio en la lógica del negocio puede requerir actualizaciones en la capa de cliente.
- **Seguridad:** La lógica de negocio en la capa de cliente puede ser más vulnerable a ataques, y mover la lógica de negocio al cliente puede exponer la estructura interna de la aplicación.



Ejemplos de Arquitectura 2-Tier:

- **Microsoft Access:** Es un sistema de gestión de bases de datos que permite a los usuarios crear y gestionar bases de datos en su computadora local. La aplicación de escritorio actúa como el cliente, y la base de datos almacenada localmente actúa como el servidor.
- **Sistemas de Punto de Venta (POS) de Pequeñas Empresas:** Estos sistemas suelen tener una aplicación instalada en las terminales de venta que se conectan directamente a una base de datos central para registrar las ventas, gestionar el inventario y procesar transacciones.

- **Arquitectura de tres niveles (3-Tier):** En la cual se introduce un intermediario entre el cliente y el servidor que tiene normalmente la responsabilidad de aplicar una capa de lógica de negocio.

1. **Capa de Presentación:** Es la interfaz de usuario de la aplicación, donde los usuarios interactúan directamente con el sistema. Incluye elementos como páginas web, aplicaciones móviles o aplicaciones de escritorio que muestran la información y permiten la entrada de datos.
2. **Capa de Lógica de Negocio (Business Logic Tier):** Esta capa actúa como intermediaria entre la capa de presentación y la capa de datos, procesando las solicitudes de la capa de presentación, aplicando la lógica de negocio y comunicándose con la capa de datos.

3. **Capa de Datos (Data Tier):** Es responsable del almacenamiento y la gestión de los datos. Incluye sistemas de gestión de bases de datos (DBMS), archivos y cualquier otro sistema de almacenamiento de datos.

Ventajas de la arquitectura 3-Tier

- **Escalabilidad:** Cada capa puede escalarse independientemente, permitiendo que la aplicación maneje un mayor número de usuarios y transacciones.
- **Mantenibilidad:** La separación de preocupaciones facilita el mantenimiento y la actualización de la aplicación. Los cambios en una capa no afectan directamente a las otras capas.
- **Reutilización:** La lógica de negocio y la capa de datos pueden ser reutilizadas por diferentes interfaces de usuario, como aplicaciones web y móviles.

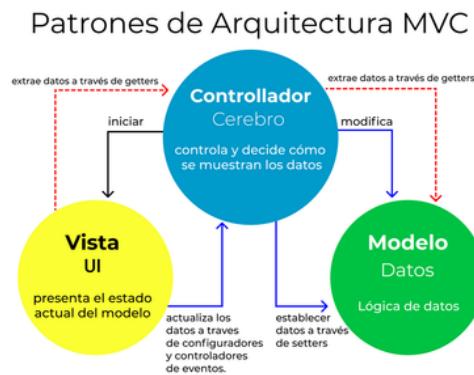


Ejemplos de Arquitectura 3-Tier:

- **WordPress:** Utiliza una capa de presentación (interfaz de usuario para gestionar el contenido), una capa de lógica de negocio (procesamiento de contenido, gestión de usuarios) y una capa de datos (base de datos MySQL).
- **Banca en Línea:** Una aplicación de banca en línea tiene una capa de presentación (interfaz de usuario web o móvil), una capa de lógica de negocio (procesamiento de transacciones, gestión de cuentas) y una capa de datos (base de datos de clientes y transacciones).
- **SAP:** Utiliza una arquitectura 3-Tier para separar la interfaz de usuario, la lógica de negocio y el almacenamiento de datos, permitiendo una gestión eficiente de los recursos empresariales.

Modelo Vista Controlador

- Es un patrón en el diseño de software comúnmente utilizado para implementar interfaces de usuario, datos y lógica de control. Enfatiza una separación entre la lógica de negocios y su visualización.
 - **Modelo:** El backend que contiene toda la lógica de datos
 - **Vista:** El frontend o interfaz gráfica de usuario (GUI)
 - **Controlador:** El cerebro de la aplicación que controla como se muestran los datos.



Componentes del MVC

- **Modelo:** Representa los datos y la lógica de negocio de la aplicación. El modelo gestiona la información, lógica y reglas del negocio.
 - Acceso y gestión de los datos.
 - Validación de los datos.
 - Notificación a la vista de cambios en los datos.
- **Vista:** Representa la interfaz de usuario. La vista muestra los datos del modelo al usuario y envía los comandos del usuario al controlador.
 - Renderización de la interfaz de usuario.
 - Captura de la entrada del usuario y presentación de esta entrada al controlador.

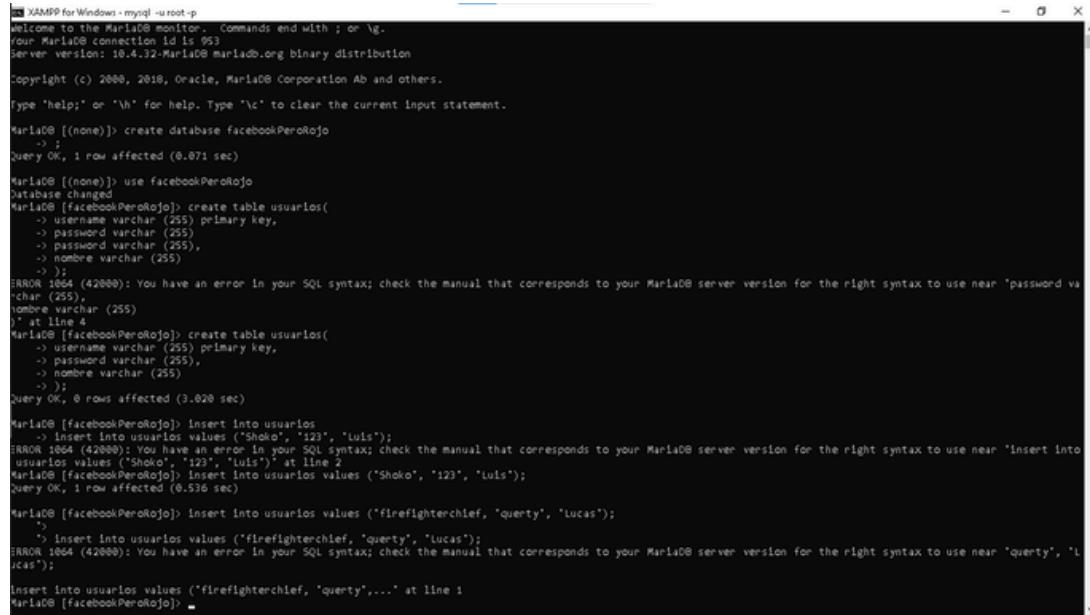


- **Controlador:** Actúa como un intermediario entre el modelo y la vista. El controlador recibe las entradas del usuario desde la vista, las procesa (mediante el modelo), y devuelve la respuesta para ser mostrada en la vista.
 - Interpretación de las acciones del usuario.
 - Comunicación con el modelo para actualizar los datos.
 - Actualización de la vista para reflejar los cambios en el modelo.

Ejercicio MVC:

Paso 1.

- Crear una base de datos mediante la ejecución de XAMPP, en la terminal de Shell



```
[root@XAMPPforWindows-x64- mysql -u root -p
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 953
Server version: 10.4.32-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database facebookPeroRojo
MariaDB [(none)]>
Query OK, 1 row affected (0.071 sec)

MariaDB [(none)]> use facebookPeroRojo
Database changed
MariaDB [facebookPeroRojo]> create table usuarios(
    ->     username varchar (255) primary key,
    ->     password varchar (255),
    ->     password varchar (255),
    ->     nombre varchar (255)
    -> );
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'password va
char (255),
nombre varchar (255)
' at line 4
MariaDB [facebookPeroRojo]> create table usuarios(
    ->     username varchar (255) primary key,
    ->     password varchar (255),
    ->     nombre varchar (255)
    -> );
Query OK, 0 rows affected (3.020 sec)

MariaDB [facebookPeroRojo]> insert into usuarios
    ->     insert into usuarios values ('Shoko', '123', 'Luis');
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'insert into
usuarios values ('Shoko', '123', 'Luis')' at line 2
MariaDB [facebookPeroRojo]> insert into usuarios values ('Shoko', '123', 'Luis');
Query OK, 1 row affected (0.536 sec)

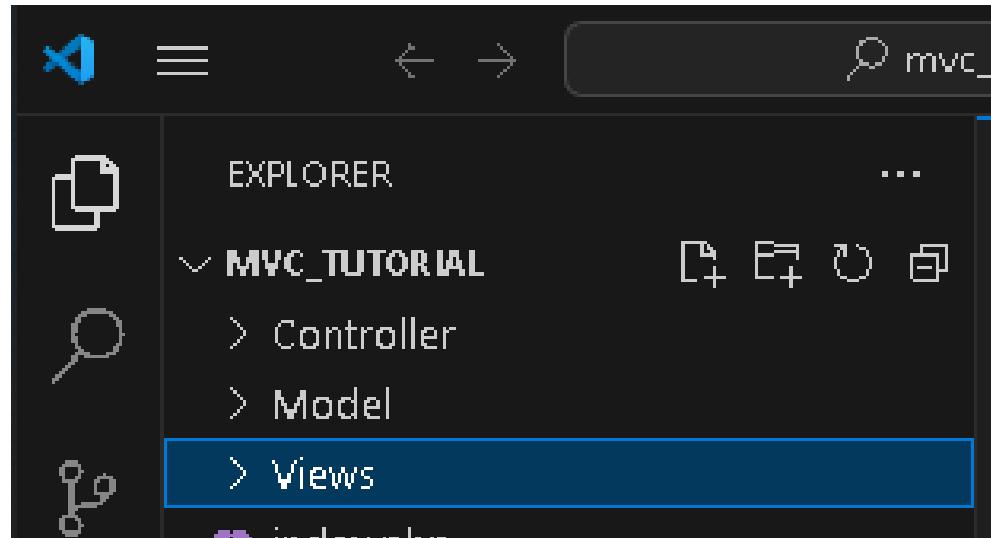
MariaDB [facebookPeroRojo]> insert into usuarios values ('firefighterchief', 'query', 'Lucas');
->
->     insert into usuarios values ('firefighterchief', 'query', 'Lucas');
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'query', 'L
ucas');

Insert into usuarios values ('firefighterchief', 'query',...) at line 1
MariaDB [facebookPeroRojo]>
```

Ejercicio MVC:

Paso 2.

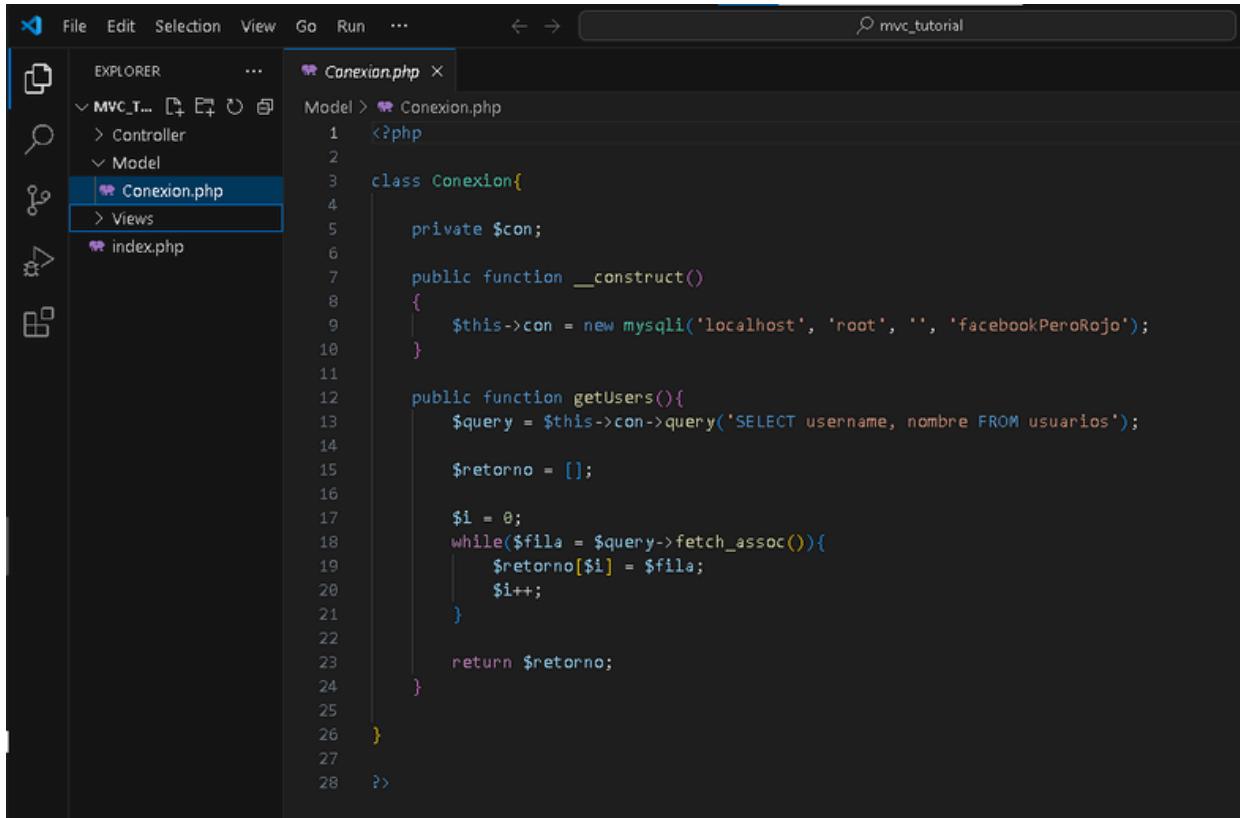
- Crear las carpetas MVC, dentro de la carpeta XAMPP



Ejercicio MVC:

Paso 3.

- Empezaremos creando el modelo con un archivo llamado conexion.php



The screenshot shows a code editor window with the following details:

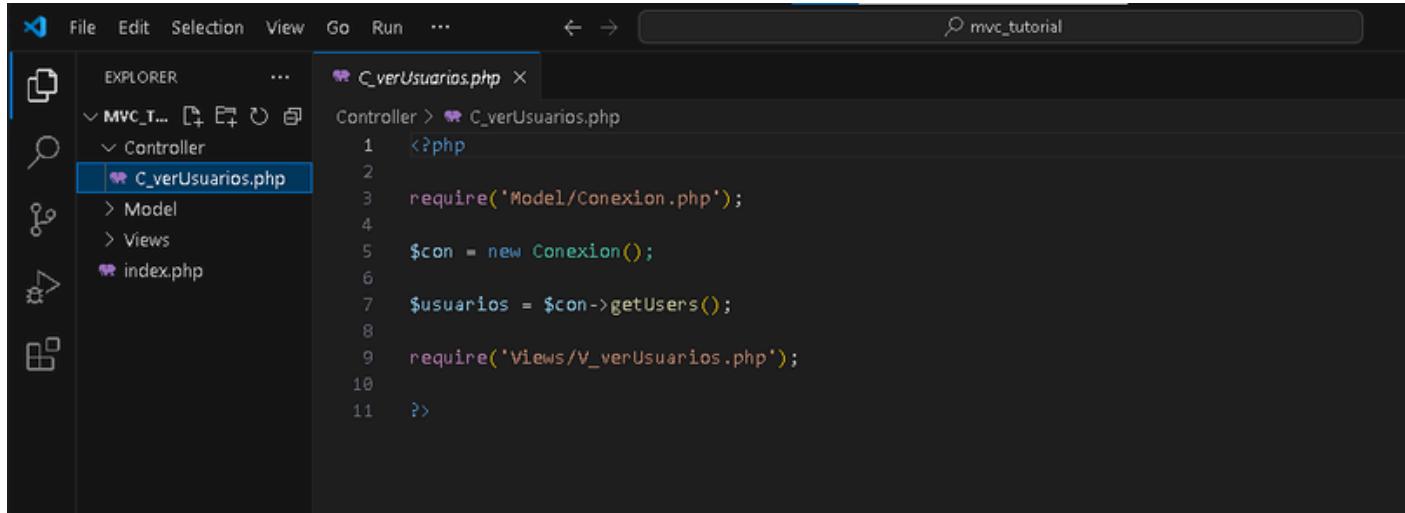
- Title Bar:** The title bar displays "Conexion.php" and "mvc_tutorial".
- Explorer Panel:** On the left, the "EXPLORER" panel shows a project structure:
 - A root folder named "MVC_T..." with a plus sign icon.
 - "Controller" and "Model" folders.
 - "Conexion.php" (selected) and "index.php" files under the "Model" folder.
 - "Views" folder.
- Code Editor:** The main area contains the PHP code for the "Conexion.php" class:

```
<?php
class Conexion{
    private $con;
    public function __construct()
    {
        $this->con = new mysqli('localhost', 'root', '', 'facebookPeroRojo');
    }
    public function getUsers(){
        $query = $this->con->query("SELECT username, nombre FROM usuarios");
        $retorno = [];
        $i = 0;
        while($fila = $query->fetch_assoc()){
            $retorno[$i] = $fila;
            $i++;
        }
        return $retorno;
    }
}>?
```

Ejercicio MVC:

Paso 4.

- Despues vamos a crear un archivo dentro de la carpeta Controller para ver la lista de usuarios en una tabla C_verUsuarios.php

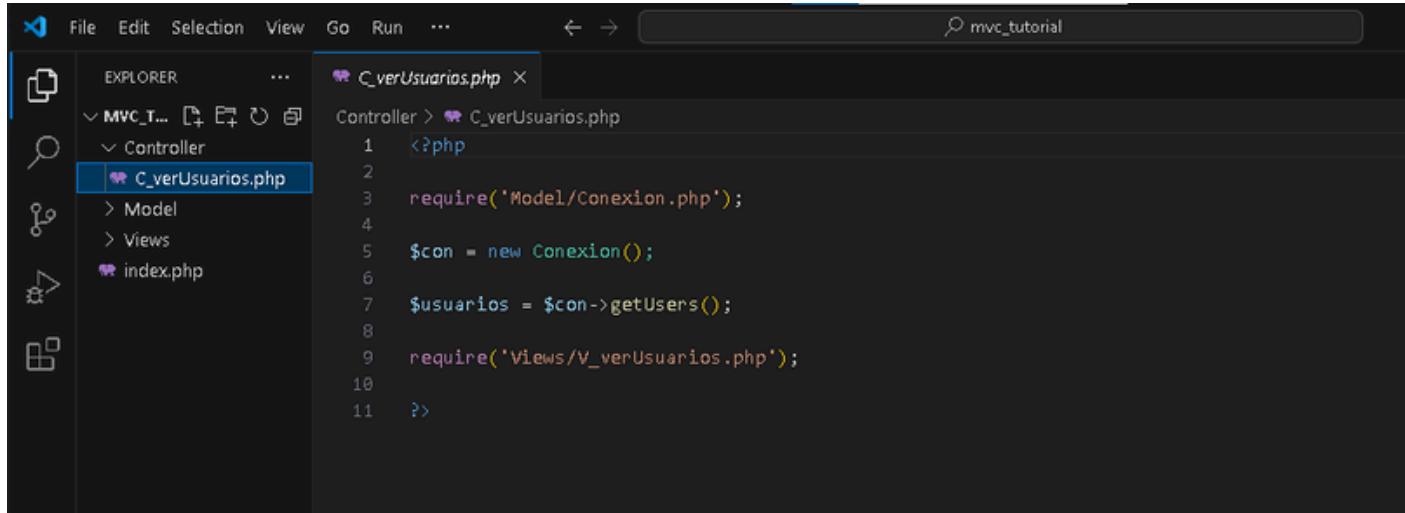


```
<?php
require('Model/Conexion.php');
$con = new Conexion();
$usuarios = $con->getUsers();
require('Views/V_verUsuarios.php');
```

Ejercicio MVC:

Paso 5.

- Despues vamos a crear un archivo dentro de la carpeta Controller para ver la lista de usuarios en una tabla llamada C_verUsuarios.php

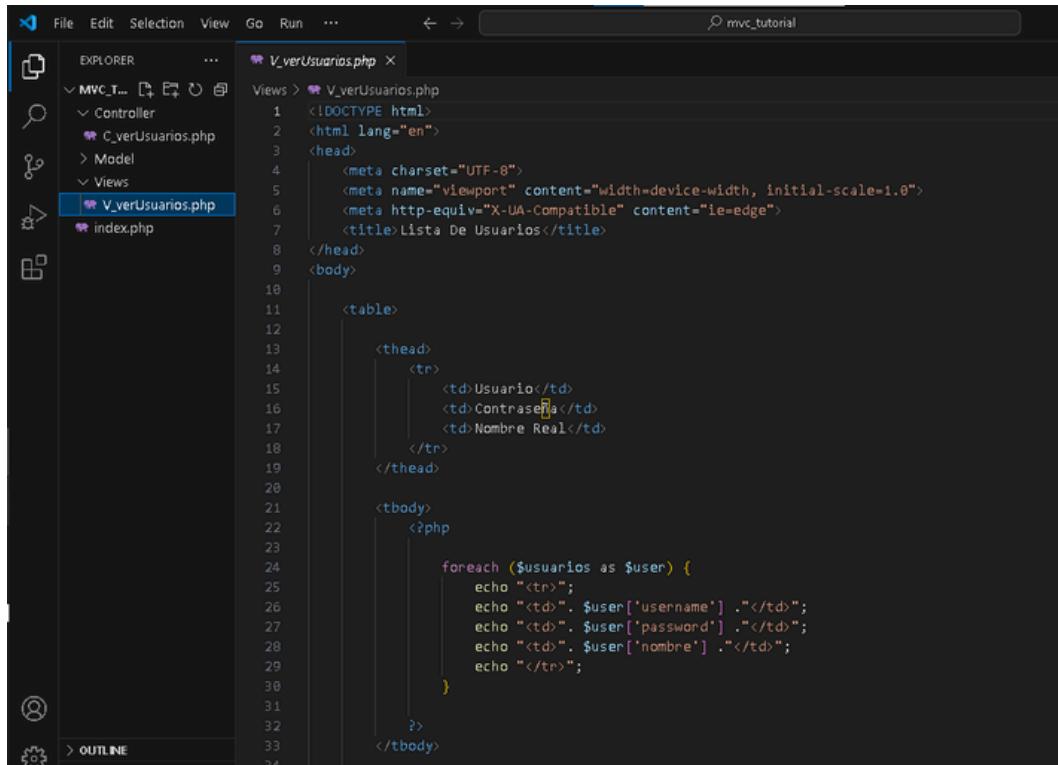


```
<?php
require('Model/Conexion.php');
$con = new Conexion();
$usuarios = $con->getUsers();
require('Views/V_verUsuarios.php');
```

Ejercicio MVC:

Paso 6.

- Para mostrar la tabla, vamos a crear una dentro de la carpeta Views, donde el archivo se va a llamar V_verUsuarios.php



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Lista De Usuarios</title>
</head>
<body>

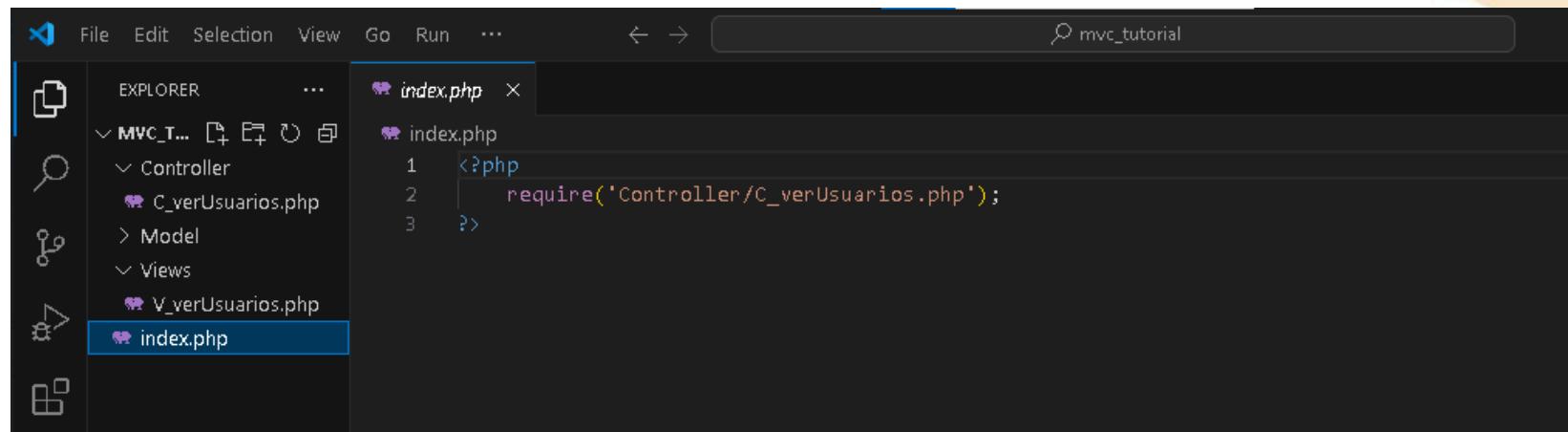
    <table>
        <thead>
            <tr>
                <td>Usuario</td>
                <td>Contraseña</td>
                <td>Nombre Real</td>
            </tr>
        </thead>
        <tbody>
            <?php
                foreach ($usuarios as $user) {
                    echo "<tr>";
                    echo "<td>". $user['username'] ."</td>";
                    echo "<td>". $user['password'] ."</td>";
                    echo "<td>". $user['nombre'] ."</td>";
                    echo "</tr>";
                }
            <?>
        </tbody>
    </table>

</body>
</html>
```

Ejercicio MVC:

Paso 7.

- Tenemos que requerir al controlado, por eso se lleva a cabo la sintaxis de index.php



The screenshot shows a code editor interface with the following details:

- Project Explorer:** Shows the project structure under "MVC_Tutorial".
 - Controller folder contains "C_verUsuarios.php".
 - Model folder.
 - Views folder contains "V_verUsuarios.php".
 - "index.php" is selected and highlighted in blue.
- Code Editor:** Displays the "index.php" file content:

```
<?php
require('Controller/C_verUsuarios.php');
?>
```
- Toolbar:** Includes standard file operations like File, Edit, Selection, View, Go, Run, etc., and a search bar at the top right.



Ejercicio MVC:

Paso 8.

- Finalmente ingresamos la dirección correcta, para visualizar la tabla

The screenshot shows a Microsoft Edge browser window with the title bar "Aplicación CRUD PHP, MYSQL, ..." and a tab labeled "Lista De Usuarios". The address bar shows the URL "localhost/mvc_tutorial/". The main content area displays a table with three columns: "Usuario", "Contraseña", and "Nombre Real". There are two rows of data:

Usuario	Contraseña	Nombre Real
Andres	N/A	Andres
Shoko	N/A	Luis