



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

**Desarrollo de una Aplicación Web en PHP con el Modelo MVC
para la Gestión de Libros y Autores**

NOMBRE:

ADRIANA BORJA
ALAN QUILUMBAQUIN
GENESIS DEL ROCIO TITO
CAMILA QUIROLA

APLICACIÓN TECNOLOGÍAS WEB

NRC: 17707

DOCENTE:

ING. ANGEL GEOVANNY CUDCO POMAGUALLI

GitHub:

<https://github.com/Genesisrocio12/Actividad-de-aprendizaje-N-3>

FECHA DE ENTREGA:

28-08-2024

Actividad de aprendizaje n.º 3

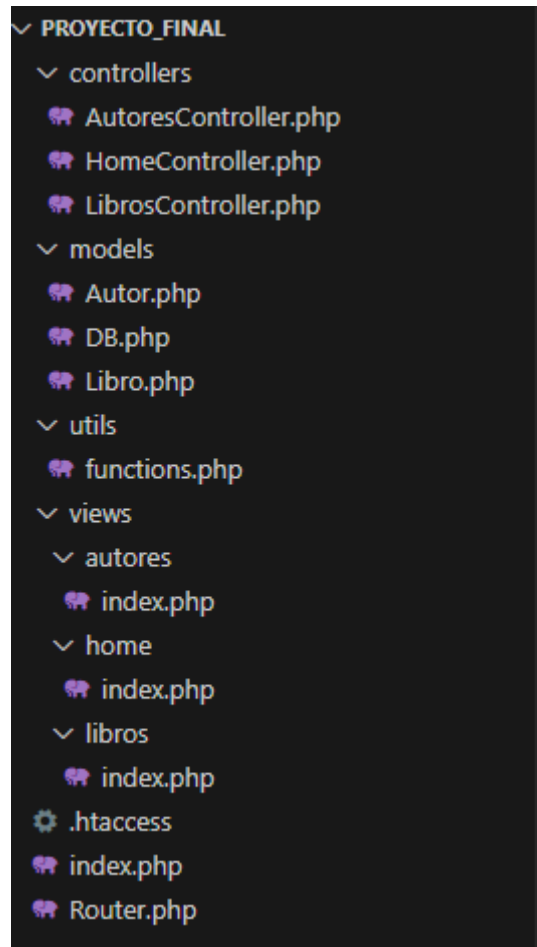
1. Introducción

En esta actividad, se desarrolló una aplicación web utilizando el modelo de arquitectura MVC (Modelo-Vista-Controlador) en PHP para la gestión de libros y autores. La aplicación implementa un menú de navegación dinámico, gestiona peticiones AJAX mediante Axios, y utiliza Bootstrap para el diseño de la interfaz con modales para la gestión de formularios. Además, se configuraron rutas mediante un archivo router.php y .htaccess para manejar URLs amigables.

2. Estructura del Proyecto

El proyecto está organizado en las siguientes carpetas y archivos:

- **controllers/**: Contiene los controladores de la aplicación.
 - HomeController.php
 - LibrosController.php
 - AutoresController.php
- **models/**: Contiene los modelos que interactúan con la base de datos.
 - DB.php
 - Libro.php
 - Autor.php
- **views/**: Contiene las vistas que se presentan al usuario.
 - **home/**: Vistas relacionadas con la página de inicio.
 - index.php
 - **libros/**: Vistas para la gestión de libros.
 - index.php
 - **autores/**: Vistas para la gestión de autores.
 - index.php
- **utils/**: Contiene funciones auxiliares.
 - functions.php
- **.htaccess**: Configura las rutas amigables.
- **index.php**: Punto de entrada principal.
- **router.php**: Manejador de rutas.

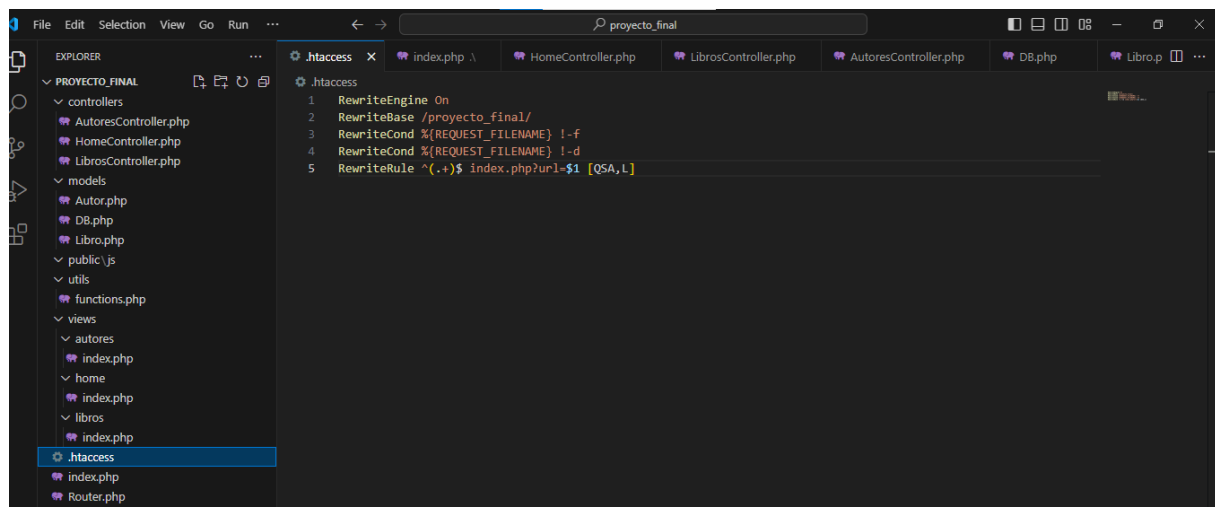


3. Implementación

3.1. Configuración de Rutas y .htaccess

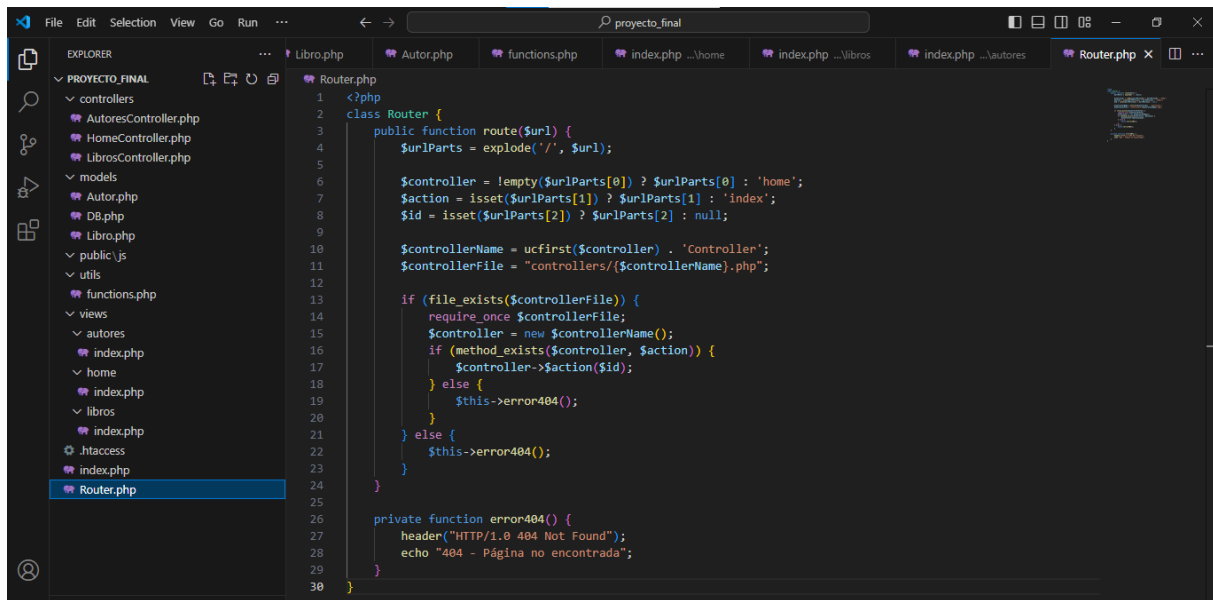
- ✓ El archivo .htaccess está configurado para redirigir todas las solicitudes a index.php, donde se maneja la lógica de enrutamiento.

.htaccess



- ✓ El archivo router.php gestiona la redirección a los controladores adecuados basándose en los parámetros de la URL.

router.php

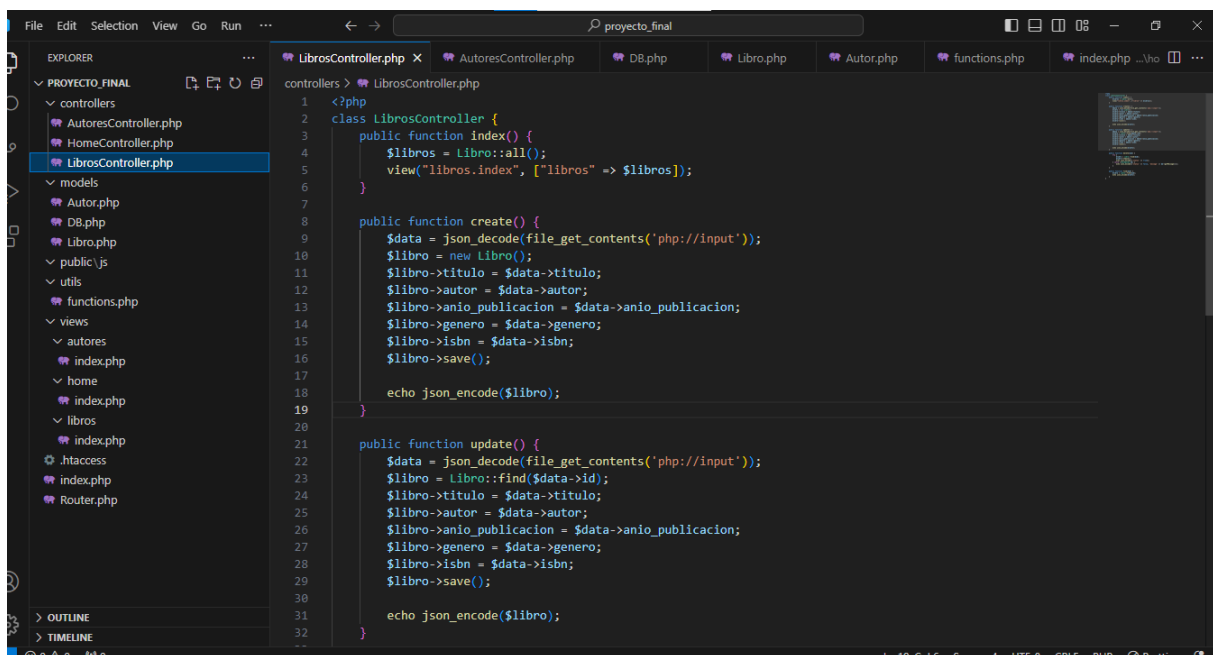


```
1 <?php
2 class Router {
3     public function route($url) {
4         $urlParts = explode('/', $url);
5
6         $controller = (empty($urlParts[0]) ? $urlParts[0] : 'home');
7         $action = isset($urlParts[1]) ? $urlParts[1] : 'index';
8         $id = isset($urlParts[2]) ? $urlParts[2] : null;
9
10        $controllerName = ucfirst($controller) . 'Controller';
11        $controllerFile = "controllers/{$controllerName}.php";
12
13        if (file_exists($controllerFile)) {
14            require_once $controllerFile;
15            $controller = new $controllerName();
16            if (method_exists($controller, $action)) {
17                $controller->$action($id);
18            } else {
19                $this->error404();
20            }
21        } else {
22            $this->error404();
23        }
24    }
25
26    private function error404() {
27        header("HTTP/1.0 404 Not Found");
28        echo "404 - Página no encontrada";
29    }
30 }
```

3.2. Controladores

- ✓ Los controladores gestionan las peticiones y la lógica de negocio. Por ejemplo, el LibrosController maneja la creación, actualización, eliminación y visualización de libros.

LibrosController.php

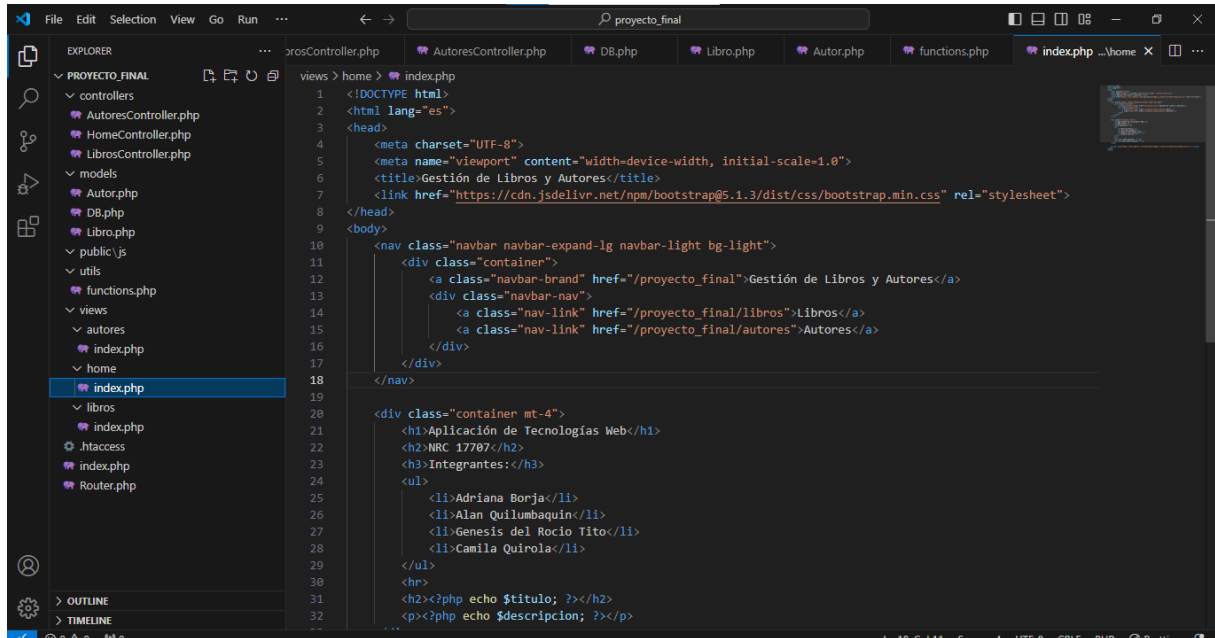


```
1 <?php
2 class LibrosController {
3     public function index() {
4         $libros = Libro::all();
5         view("libros.index", ["libros" => $libros]);
6     }
7
8     public function create() {
9         $data = json_decode(file_get_contents('php://input'));
10        $libro = new Libro();
11        $libro->titulo = $data->titulo;
12        $libro->autor = $data->autor;
13        $libro->anio_publicacion = $data->anio_publicacion;
14        $libro->genero = $data->genero;
15        $libro->isbn = $data->isbn;
16        $libro->save();
17
18        echo json_encode($libro);
19    }
20
21    public function update() {
22        $data = json_decode(file_get_contents('php://input'));
23        $libro = Libro::find($data->id);
24        $libro->titulo = $data->titulo;
25        $libro->autor = $data->autor;
26        $libro->anio_publicacion = $data->anio_publicacion;
27        $libro->genero = $data->genero;
28        $libro->isbn = $data->isbn;
29        $libro->save();
30
31        echo json_encode($libro);
32    }
33 }
```

3.4. Vistas

Las vistas presentan la información al usuario. En `views/home/index.php`, se muestra la página de inicio con el título y la descripción del proyecto.

index.php (Home)

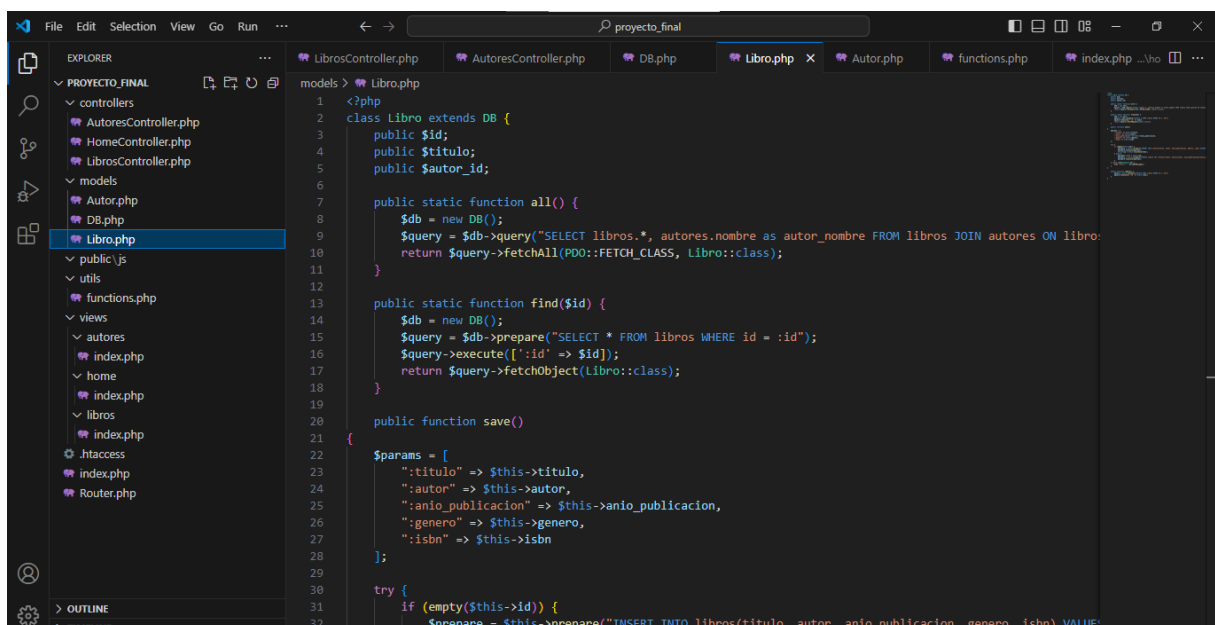


```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Gestión de Libros y Autores</title>
7 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">
8 </head>
9 <body>
10 <nav class="navbar navbar-expand-lg navbar-light bg-light">
11 <div class="container">
12 <a class="navbar-brand" href="/proyecto_final">Gestión de Libros y Autores</a>
13 <div class="navbar-nav">
14 <a class="nav-link" href="/proyecto_final/libros">Libros</a>
15 <a class="nav-link" href="/proyecto_final/autores">Autores</a>
16 </div>
17 </div>
18 </nav>
19
20 <div class="container mt-4">
21 <h1>Aplicación de Tecnologías Web</h1>
22 <h2>MRC 17707</h2>
23 <h3>Integrantes:</h3>
24 <ul>
25 <li>Adriana Borja</li>
26 <li>Alan Quilumbaquin</li>
27 <li>Genesis del Rocio Tito</li>
28 <li>Camila Quirola</li>
29 </ul>
30 <hr>
31 <h2><?php echo $titulo;></h2>
32 <p><?php echo $descripcion;></p>
```

3.3. Modelos

Los modelos gestionan las operaciones de base de datos. Por ejemplo, el modelo Libro realiza las operaciones CRUD sobre la tabla de libros.

Libro.php

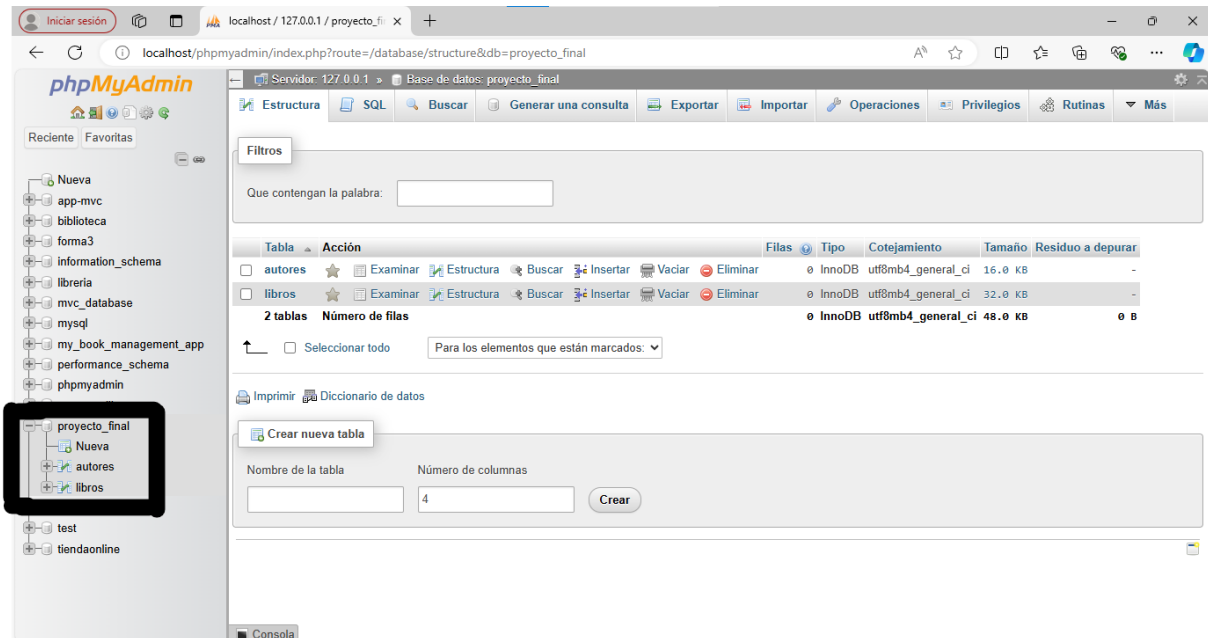


```
1 <?php
2 class Libro extends DB {
3     public $id;
4     public $titulo;
5     public $autor_id;
6
7     public static function all() {
8         $db = new DB();
9         $query = $db->query("SELECT libros.*, autores.nombre as autor_nombre FROM libros JOIN autores ON libros.autor_id = autores.id");
10        return $query->fetchAll(PDO::FETCH_CLASS, Libro::class);
11    }
12
13    public static function find($id) {
14        $db = new DB();
15        $query = $db->prepare("SELECT * FROM libros WHERE id = :id");
16        $query->execute([':id' => $id]);
17        return $query->fetchObject(Libro::class);
18    }
19
20    public function save()
21    {
22        $params = [
23            ":titulo" => $this->titulo,
24            ":autor" => $this->autor_id,
25            ":anio_publicacion" => $this->anio_publicacion,
26            ":genero" => $this->genero,
27            ":isbn" => $this->isbn
28        ];
29
30        try {
31            if (empty($this->id)) {
32                $prepare = $this->prepare("INSERT INTO libros(titulo, autor, anio_publicacion, genero, isbn) VALUE
```

3.5. Base de Datos

La base de datos se crea con dos tablas: autores y libros, con las relaciones correspondientes.

Esquema de Base de Datos

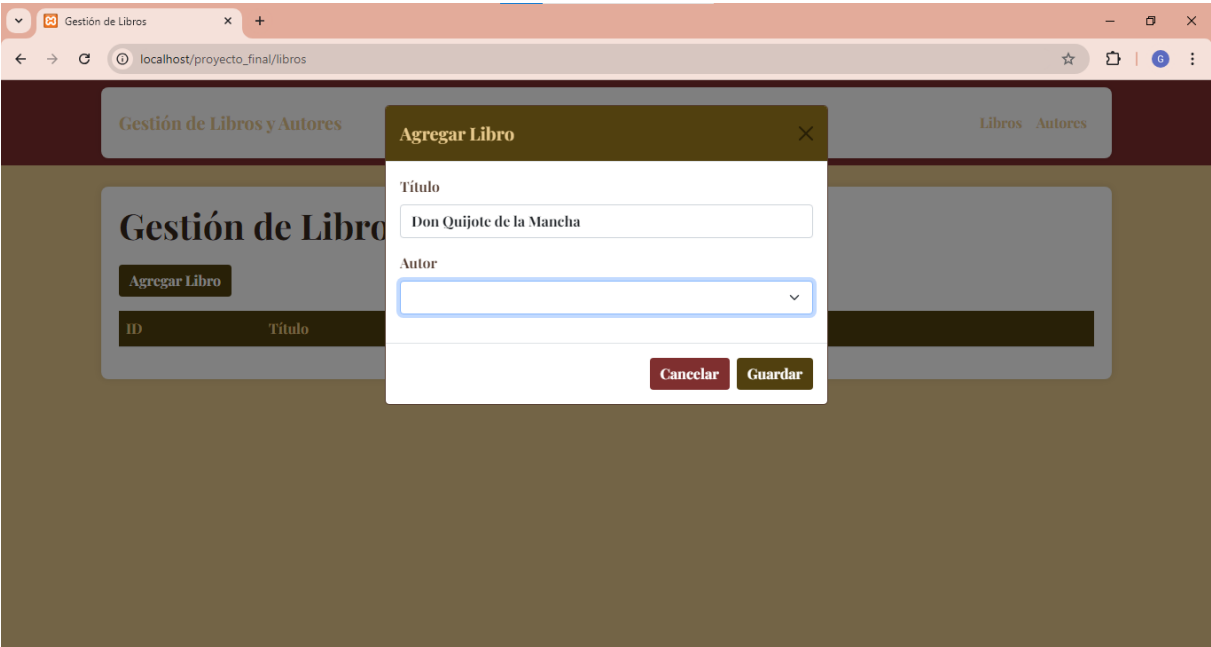


4. Capturas de Pantalla de la Ejecución

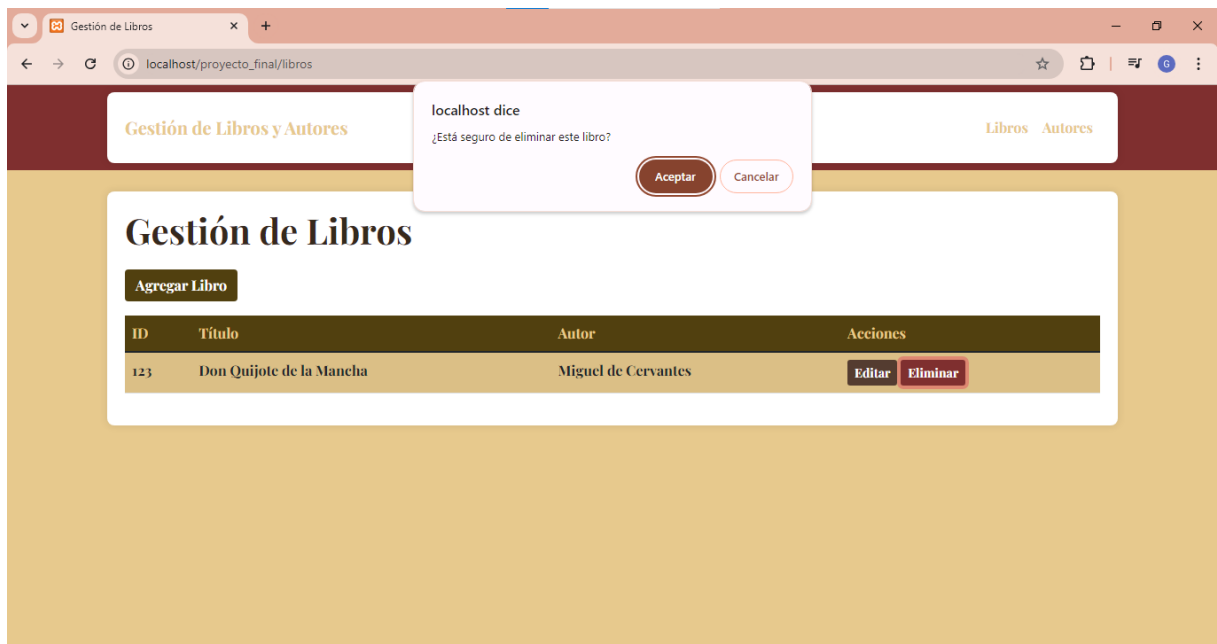
1. Página de Inicio: Muestra cómo se ve la página de inicio.



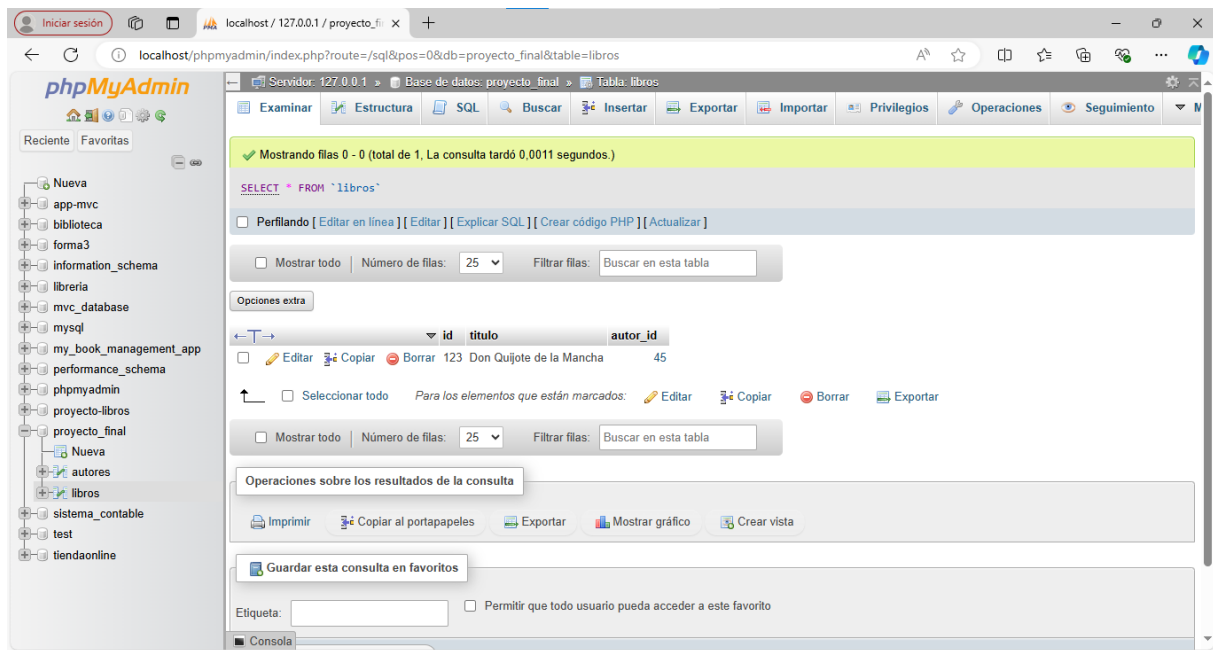
2. Gestión de Libros: Captura de la vista con la lista de libros y los modales para añadir o editar.



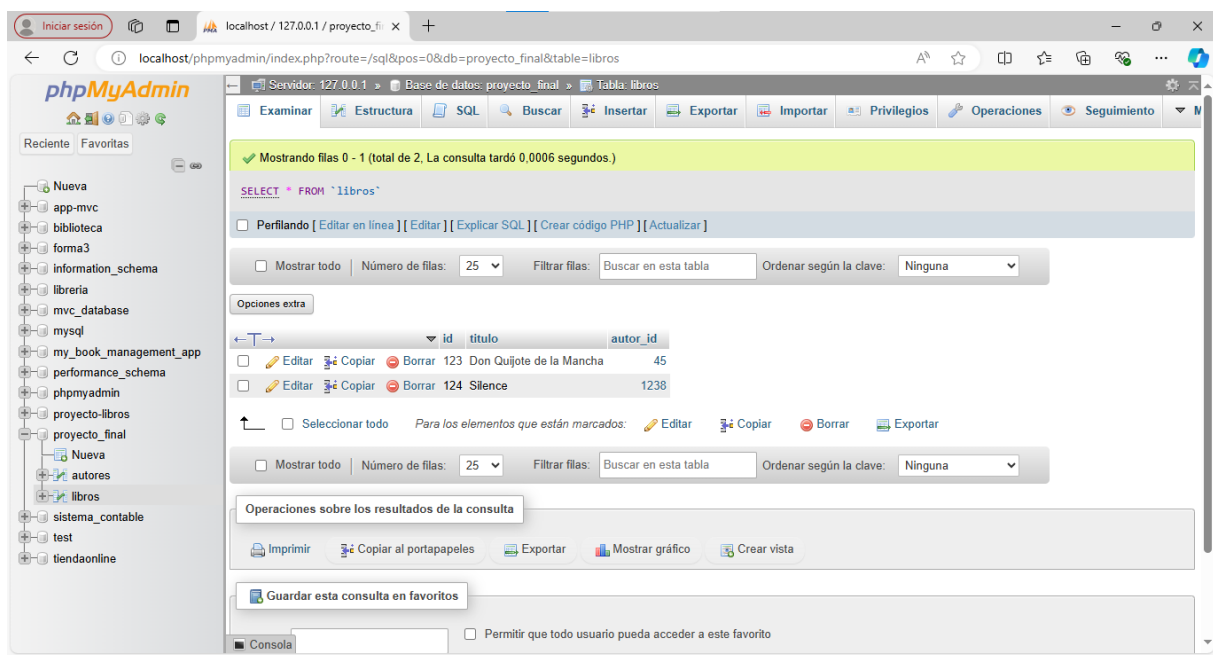




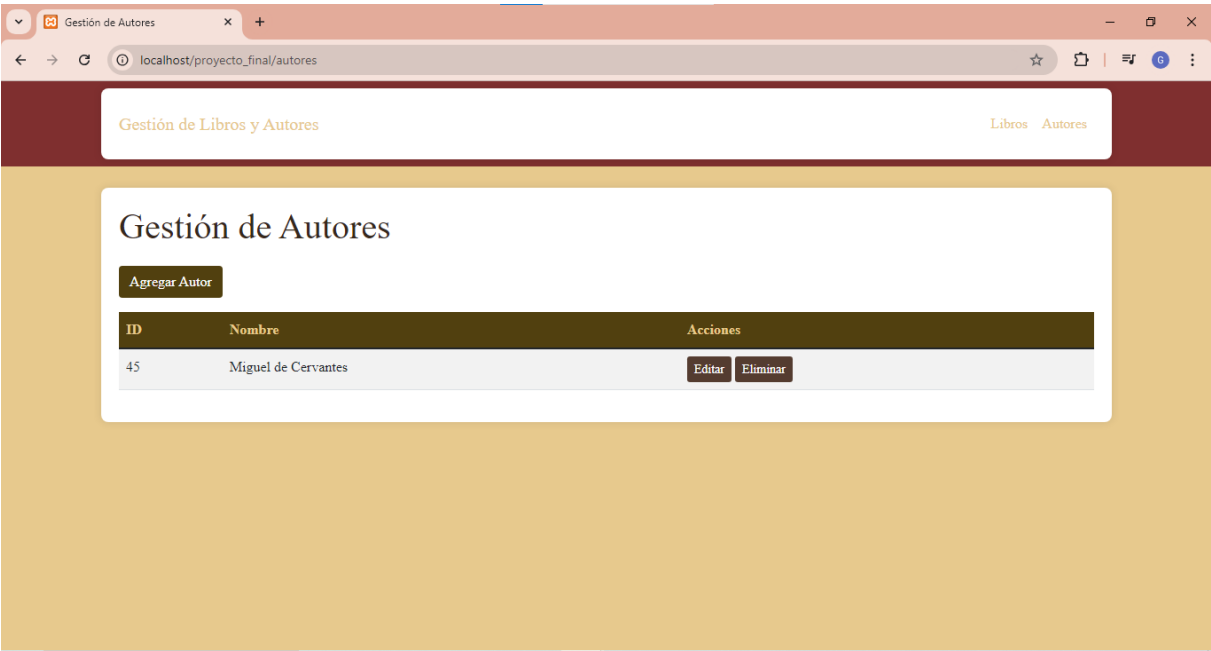
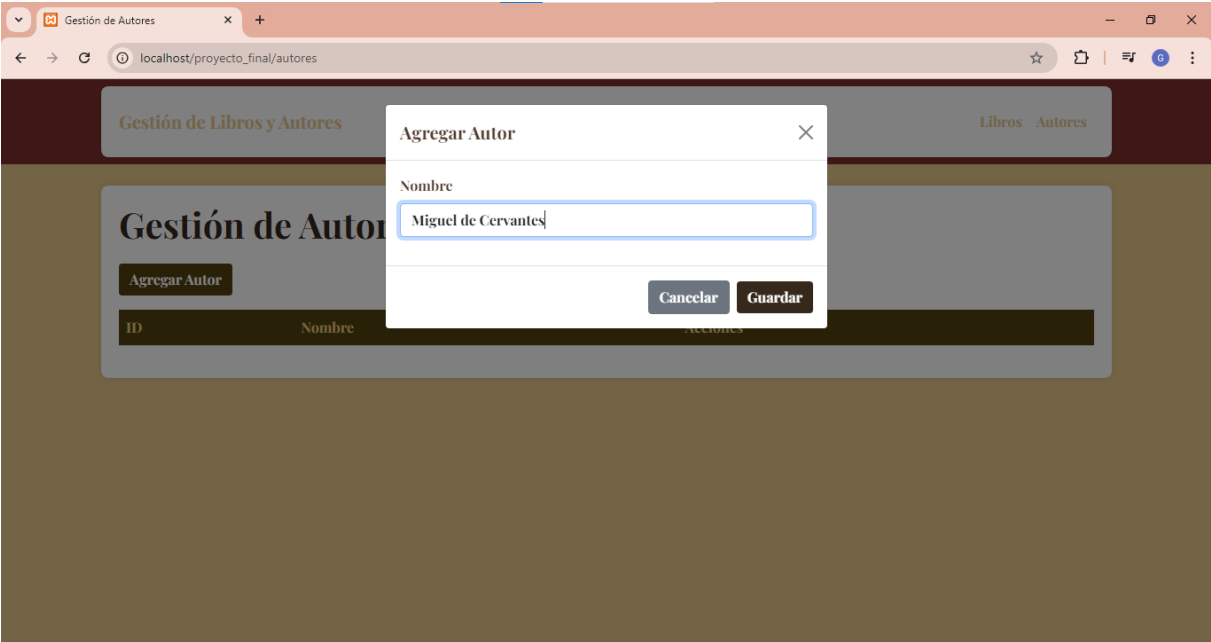
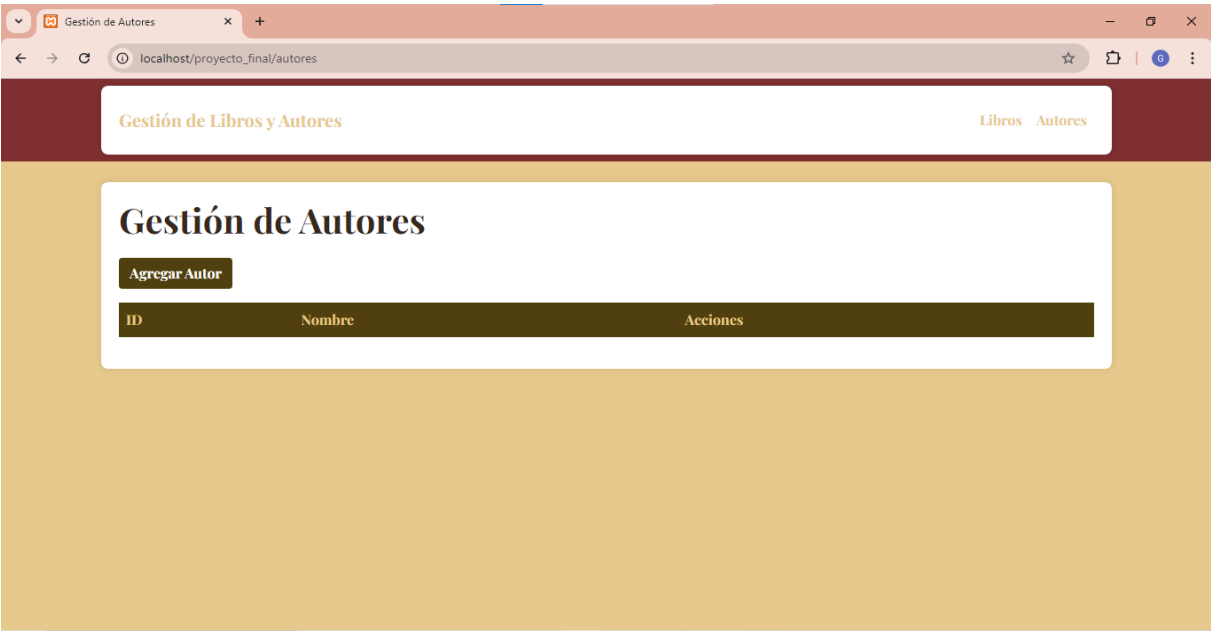
➤ **Ingreso de datos en la Base de Datos:**

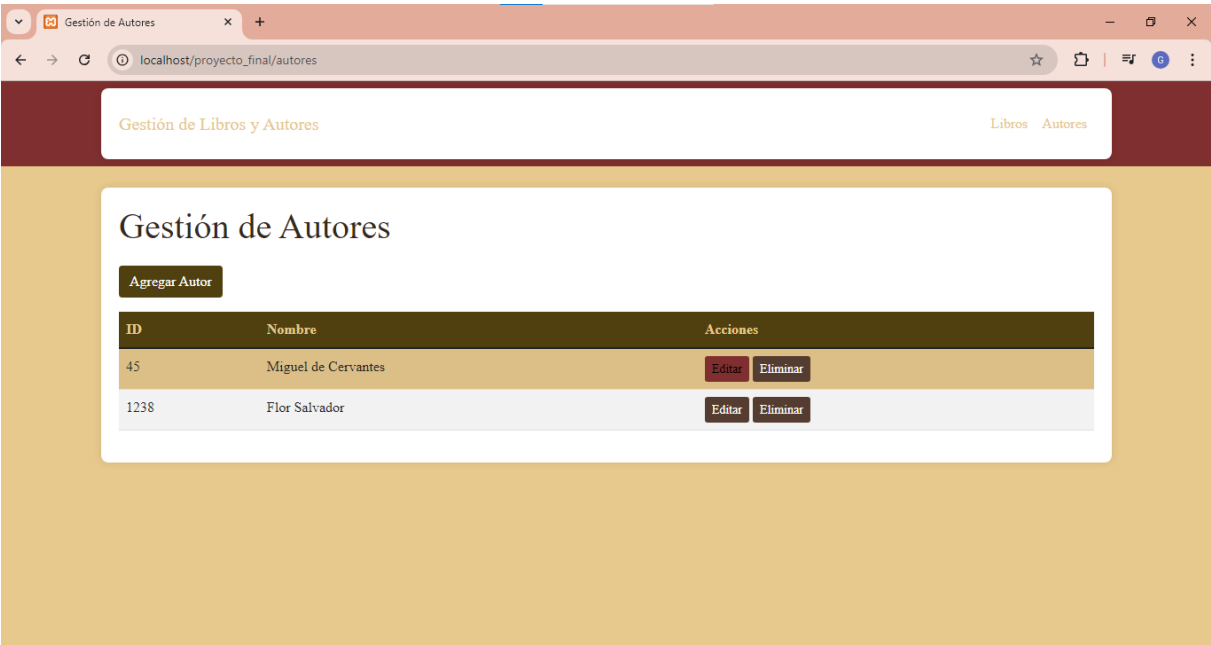
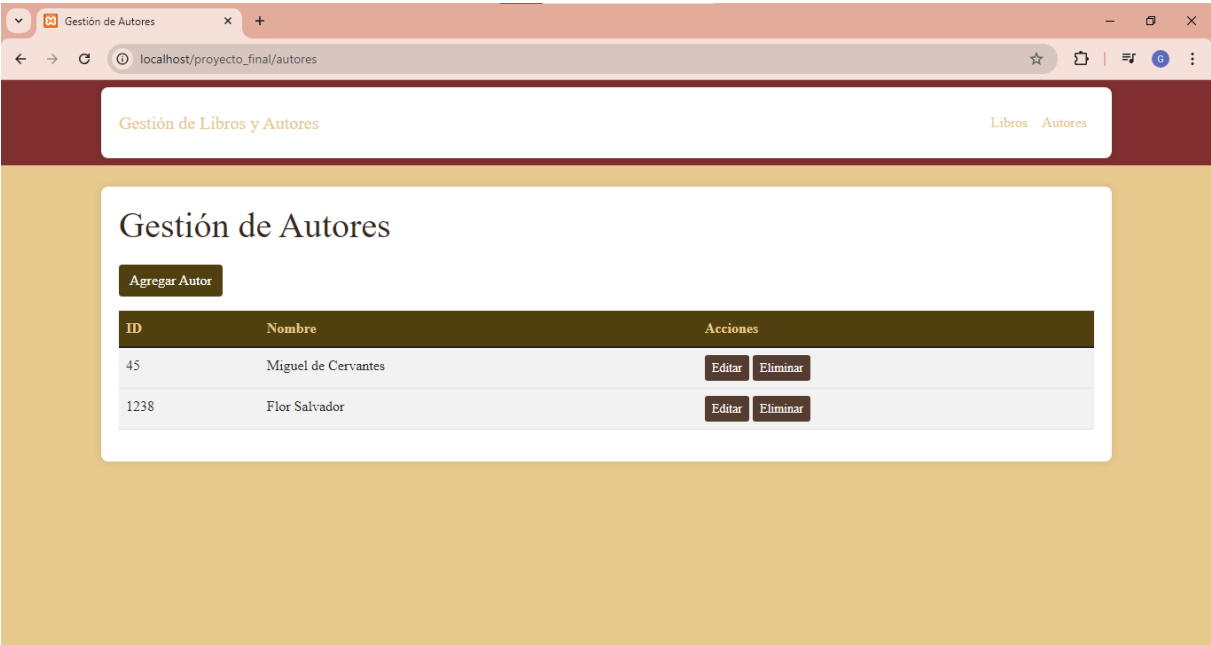
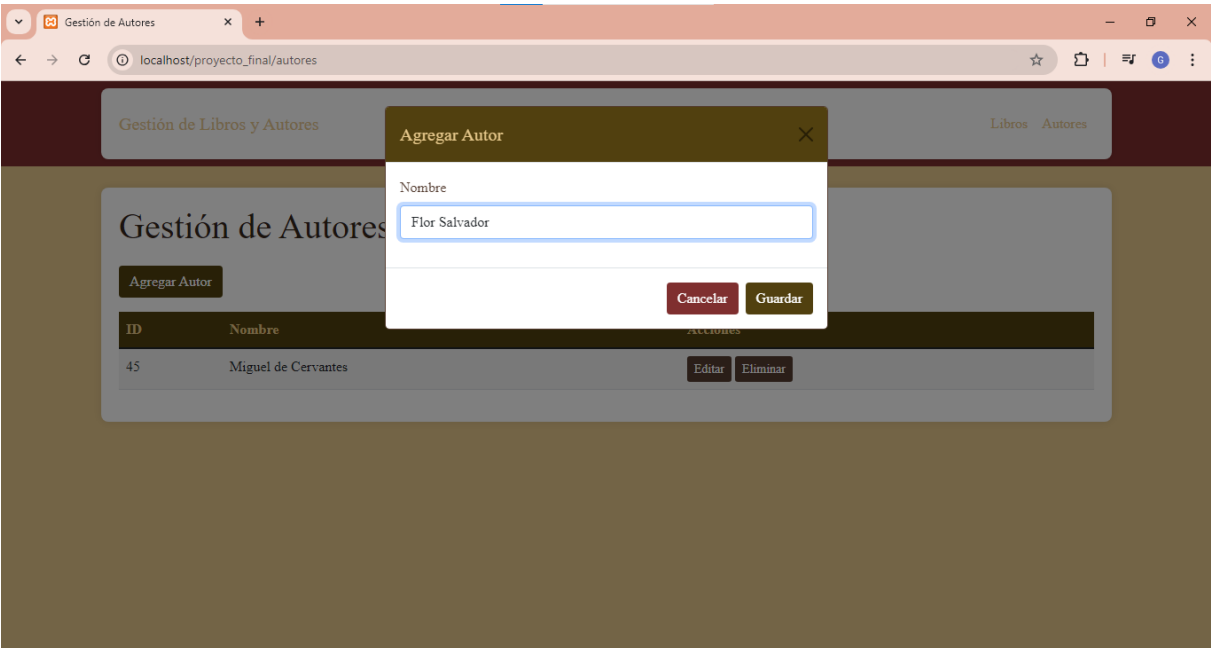


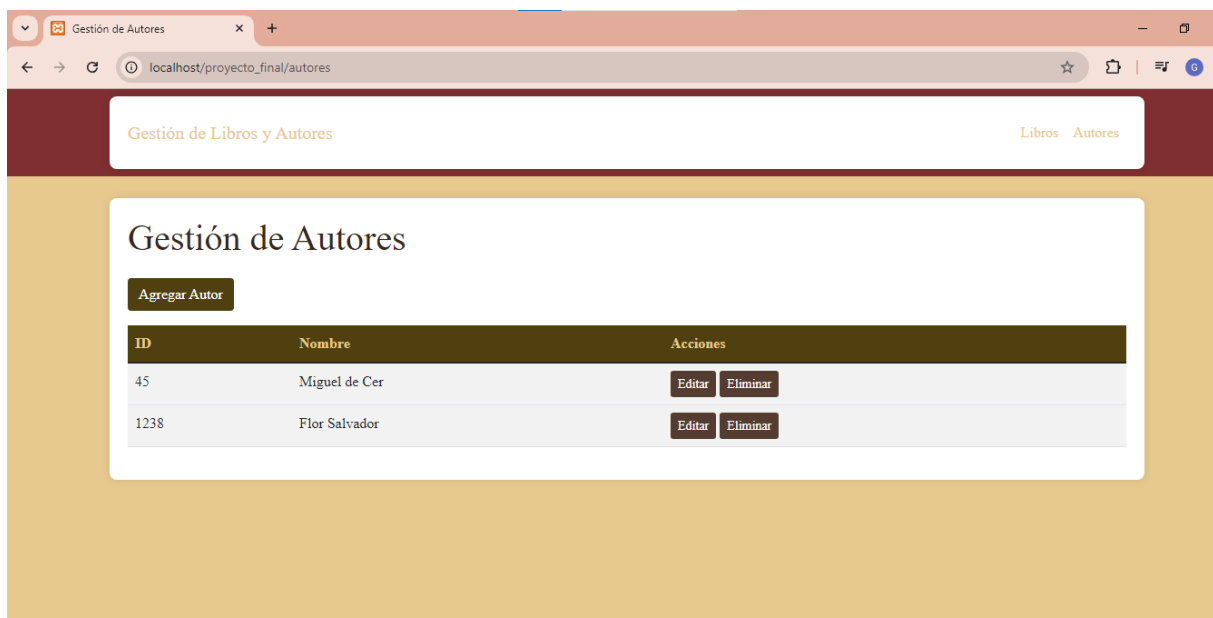
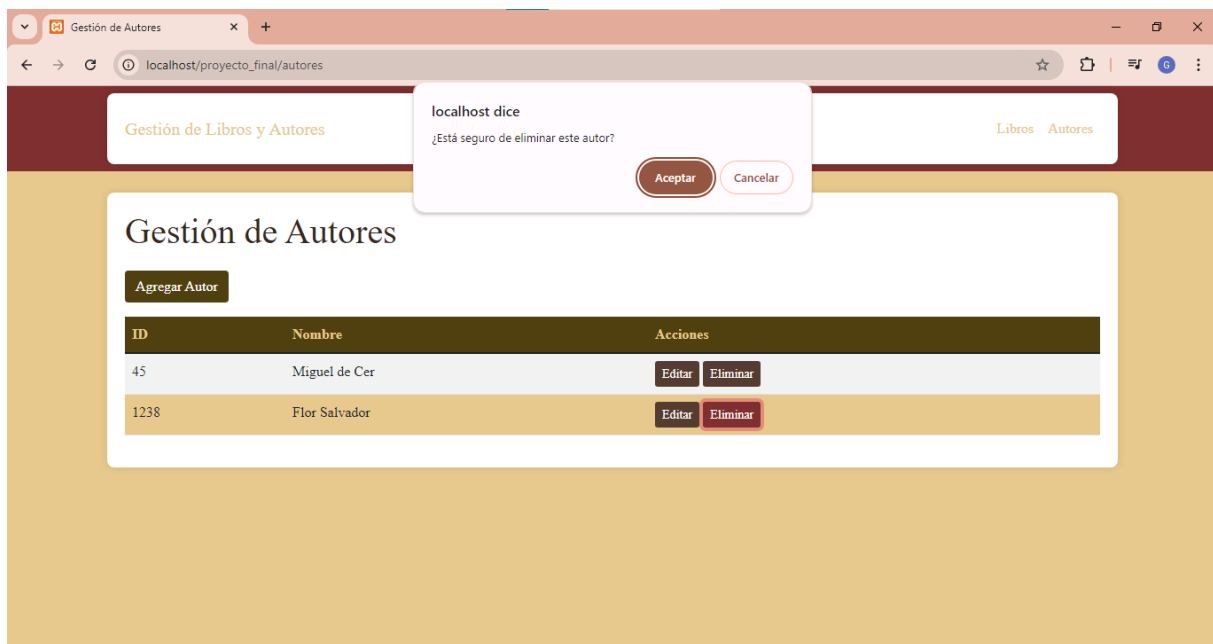
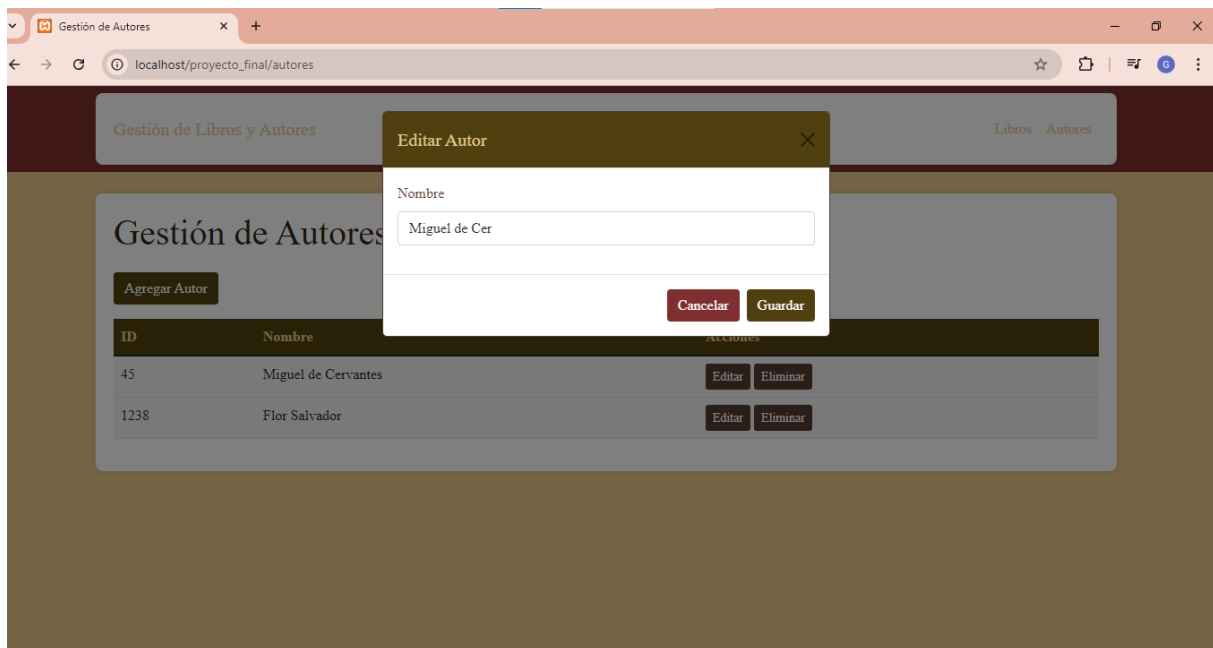
➤ Ingreso de dato en la Base de Datos después de lo que se agrego:



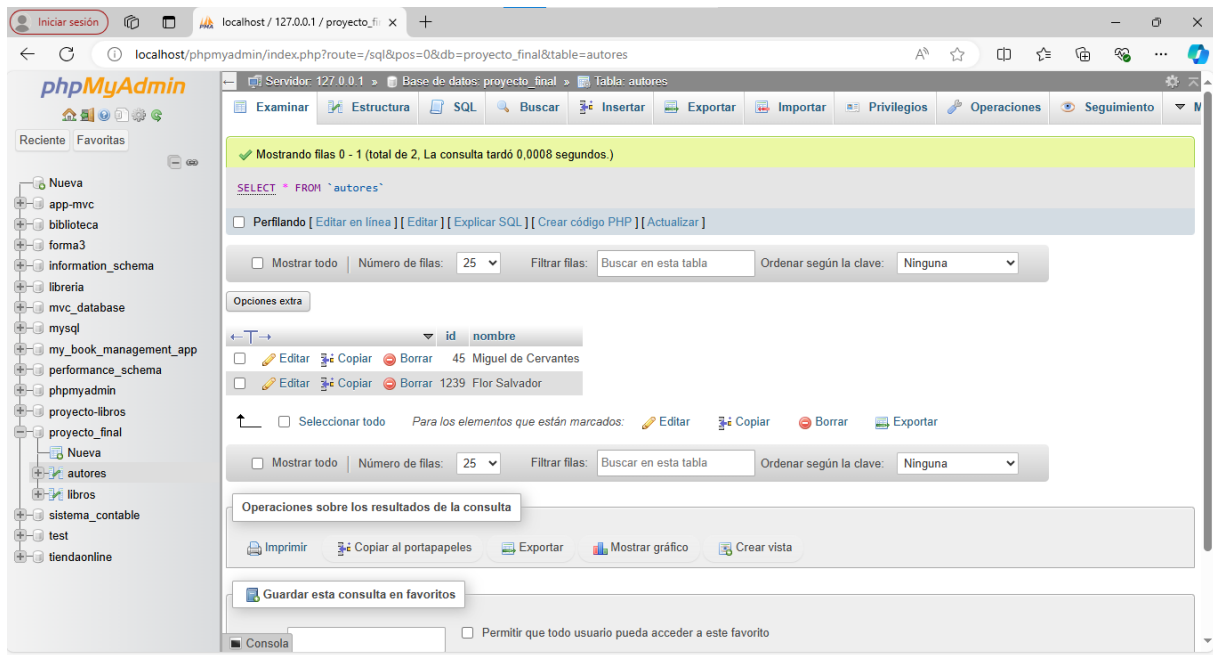
3. Gestión de Autores: Captura de la vista con la lista de autores y los modales para añadir o editar.



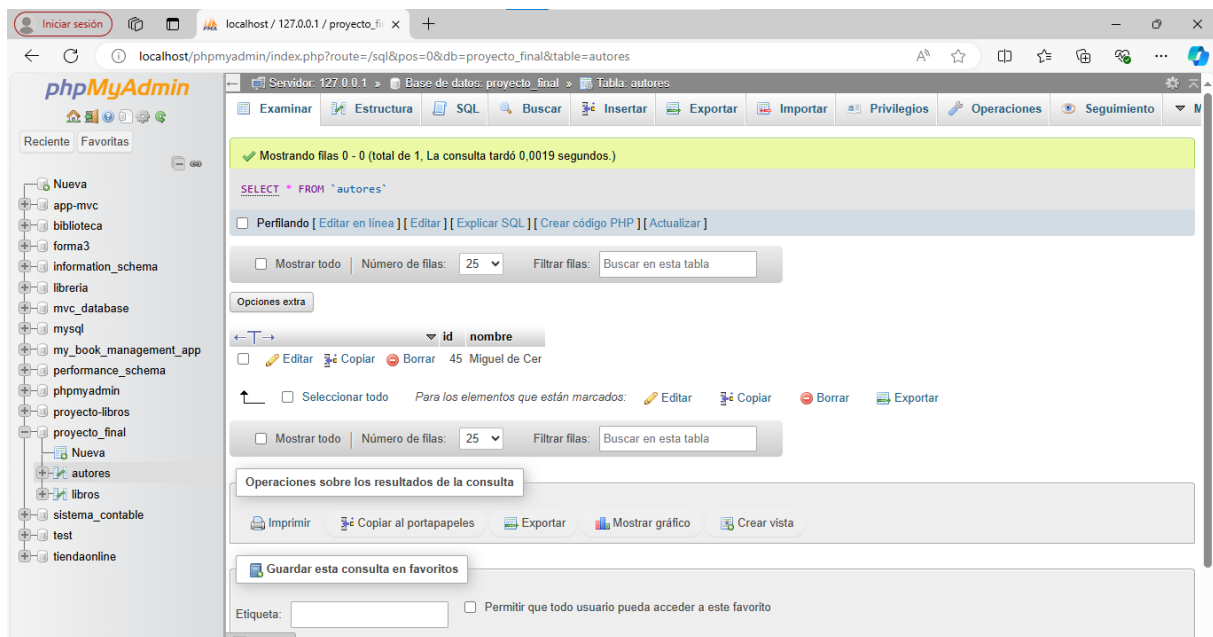




➤ Ingreso de datos en la Base de Datos:



➤ Ingreso de dato en la Base de Datos después de lo que se modificó:



5. Conclusiones

En esta actividad, hemos implementado una aplicación web completa utilizando PHP y el patrón MVC. La aplicación permite gestionar libros y autores, utilizando un diseño limpio y funcional con Bootstrap y modales para la interacción del usuario. La configuración de rutas y la estructura del proyecto siguen las mejores prácticas para asegurar un mantenimiento y escalabilidad adecuados.

➤ Subida de Evidencias

Código Fuente en GitHub

Hemos subido el código fuente completo del proyecto a un repositorio en GitHub, al que se puede acceder a través del siguiente enlace:

<https://github.com/Genesisrocio12/Actividad-de-aprendizaje-N-3>

Actividad-de-aprendizaje-N-3 Public

00ec4b8 · 1 minute ago · 5 Commits

File	Commit	Time
controllers	Proyecto Final	13 hours ago
models	Proyecto Final	13 hours ago
utils	Proyecto Final	13 hours ago
views	Proyecto Final	13 hours ago
.htaccess	Proyecto Final	13 hours ago
Proyecto_final_Tito_Genesis_Borja_Adriana_Quilumba...	Create Proyecto_final_Tito_Genesis_Borja_Adriana_Quilumba...	13 hours ago
README.md	README.md	1 minute ago
Router.php	Proyecto Final	13 hours ago
index.php	Proyecto Final	13 hours ago

README

Proyecto: Gestión de Libros y Autores

Descripción

Este proyecto es una aplicación web desarrollada en PHP utilizando el modelo MVC (Modelo-Vista-Controlador) para la gestión de libros y autores. La aplicación permite a los usuarios gestionar (crear, leer, actualizar y eliminar) libros y autores a través de una interfaz web intuitiva y moderna. La aplicación incluye funcionalidades de peticiones mediante Axios, diseño de interfaz con Bootstrap, modales para la gestión de formularios, y un menú de navegación dinámico.

Tecnologías Utilizadas

- **PHP:** Lenguaje de programación para el desarrollo del backend.
- **Modelo MVC:** Estructura para organizar el código en Modelos, Vistas y Controladores.
- **Bootstrap:** Framework CSS para el diseño de la interfaz.
- **Axios:** Librería para realizar peticiones AJAX.
- **MySQL:** Sistema de gestión de bases de datos para almacenar la información.
- **.htaccess:** Configuración para gestionar URLs amigables y redirigir solicitudes.
- **XAMPP/WAMP/LAMP:** Servidor web para ejecutar la aplicación localmente.

Languages

PHP 100.0%

Suggested workflows

Based on your tech stack

- PHP** Configure
Build and test a PHP application using Composer
- Symfony** Configure
Test a Symfony project.
- SLSA Generic generator** Configure
Generate SLSA3 provenance for your existing release workflows

[More workflows](#) [Dismiss suggestions](#)

➤ Colaboradores en GitHub

General

Access

🔒 Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Environments

Codespaces

Pages

Security

Who has access

PUBLIC REPOSITORY

This repository is public and visible to anyone.

Manage

DIRECT ACCESS


3 users have access to this repository. [3 collaborators](#).


Manage access

Select all Type


Find a collaborator...


☐

 adryborja
Collaborator



☐

 ALINFINITY
Collaborator



General

Access

🔒 Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Environments

Codespaces

Pages

Security

Code security and analysis

Deploy keys

Secrets and variables

Who has access

PUBLIC REPOSITORY

This repository is public and visible to anyone.

Manage

DIRECT ACCESS

3 users have access to this repository. [3 collaborators](#).


Manage access

Select all Type


Find a collaborator...


☐

 adryborja
Collaborator




☐

 ALINFINITY
Collaborator



☐

 caquirola
Collaborator

