***Librería Todo Libros***

Este proyecto es una página web de una librería online donde los usuarios pueden ver un catálogo de libros, ver imágenes y enviar un formulario de contacto.

¿Qué nuevas funcionalidades se agregaron?

* Se agregó un formulario de contacto funcional que puede procesar y validar datos en el servidor.
* Se implementó una vista dinámica de libros, que puede adaptarse en el futuro para cargar información desde una base de datos.
* Se estructuró el proyecto siguiendo el patrón MVC (Modelo-Vista-Controlador), lo cual mejora el mantenimiento y la escalabilidad del sistema.

Cómo se integraron los controladores y vistas:

La integración entre los controladores y las vistas se realizó utilizando el patrón MVC (Modelo-Vista-Controlador) que ofrece Spring Boot.

1. **Definición de Controladores (@Controller)**:  
   Se crearon clases Java anotadas con @Controller que manejan las rutas HTTP (@GetMapping, @PostMapping). Por ejemplo, para mostrar el formulario de contacto o el catálogo de libros.
2. **Conexión con Vistas (Thymeleaf)**:  
   Desde los controladores, se retorna el nombre de una vista HTML que se encuentra dentro del directorio src/main/resources/templates. Estas vistas están construidas con Thymeleaf, lo cual permite inyectar datos dinámicos usando expresiones como th:text, th:each, th:href, etc.
3. **Procesamiento de Formularios**:  
   En el formulario de contacto, se usó @PostMapping para recibir los datos ingresados por el usuario, vinculándolos a un objeto Java con @ModelAttribute. También se incorporó validación con anotaciones como @NotBlank y @Email.

Qué problemas se presentaron y cómo se resolvieron.

1. Las vistas no se mostraban correctamente

Problema: Al ejecutar la aplicación, algunas páginas HTML no se mostraban o aparecía error 404.

Causa: El controlador no retornaba bien el nombre del archivo.

Solución: Se ajustaron los nombres retornados desde los controladores.

2. No se renderizaban correctamente los datos con Thymeleaf

Problema: Al cargar una lista de libros o al mostrar el formulario, algunas etiquetas no mostraban los datos dinámicos.

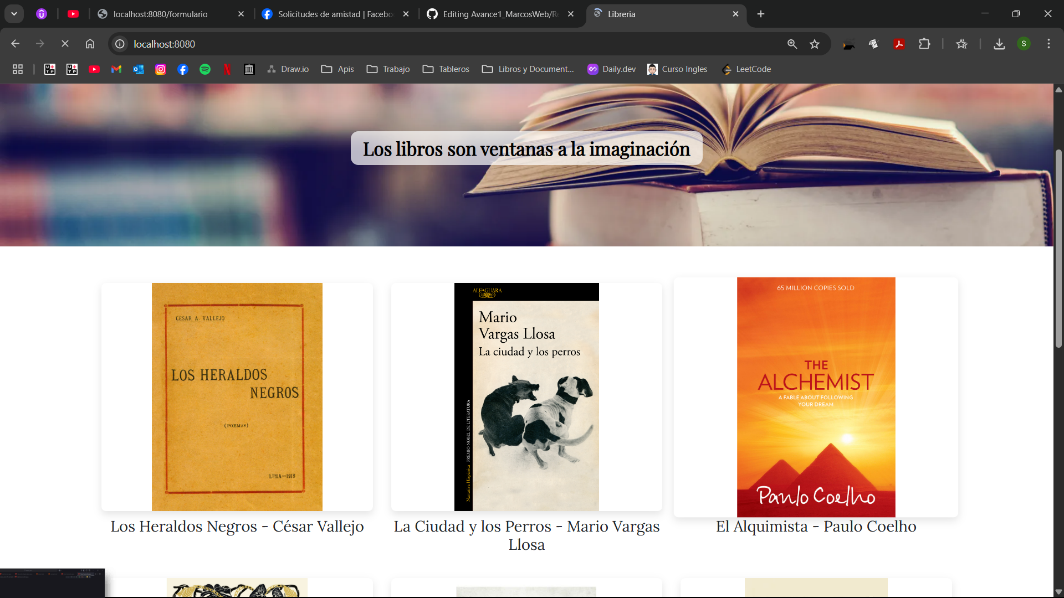
Causa y solución: Se estaban usando atributos incorrectos como value o text en lugar de th:value o th:text.

3. Errores al enviar el formulario

Problema: El formulario de contacto no enviaba datos correctamente al backend.

Causa: Faltaba la anotación @ModelAttribute

Solución: Se validó que el objeto del formulario estuviera correctamente definido en el modelo (model.addAttribute)

Capturas de Pantalla de Funcionamiento

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico, Sitio web

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Plataforma de control de versiones

git clone https://github.com/StivenBravo/Avance2\_Marcos