Universidad Autónoma de Campeche

Licenciatura en Ingeniería en Sistemas Computacionales

Materia:

Temas selectos de programación

Trabajo:

Conceptos básicos de Spring web

Alumno:

Axel Alessandro Chavez Moreno

Docente:

Jose C. Aguilar Canepa

Semestre y grupo:

8 A

Conceptos básicos de Spring Web

Este proyecto es una aplicación web básica de inicio de sesión desarrollada con Spring Boot. Permite al usuario ingresar un nombre de usuario y contraseña mediante un formulario HTML. Si las credenciales son correctas, accede a una página de bienvenida personalizada. Si no, se le muestra nuevamente el formulario con un mensaje de error.

Su objetivo consiste en desarrollar una aplicación funcional que simule un proceso de autenticación sencillo utilizando una programación orientada a objetos en Java con spring Boot como framework backend y thymeleaf como motor de plantillas para renderizar vistas HTML.

El principal objetivo de esta aplicación es: Implementar un sistema básico de inicio de sesión en una aplicación web utilizando Spring Boot y Thymeleaf, que permita validar credenciales del usuario y mostrar diferentes vistas según el resultado de la autenticación.

Las tecnologías utilizadas fueron:

- Java 17+
- Spring Boot
- Spring Web
- Thymeleaf
- Maven
- HTML5 + CSS3
- IntelliJ IDEA Community

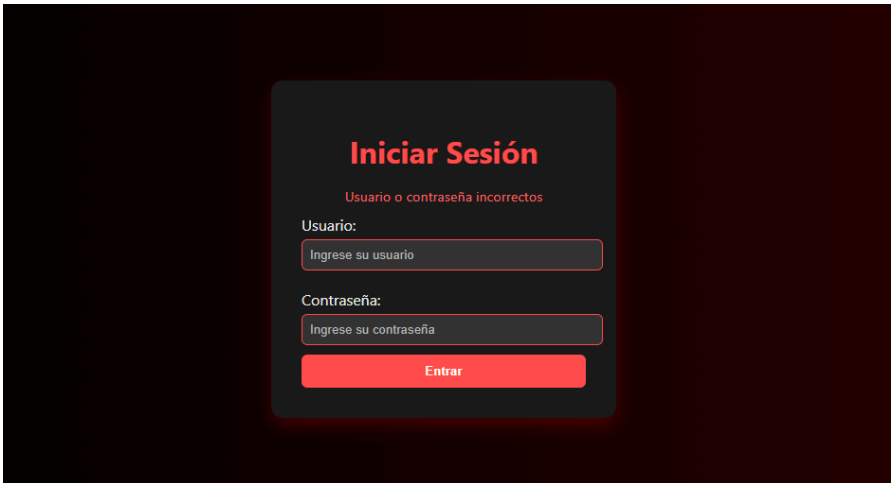
La intención de este código es brindar una base sólida para comprender cómo funciona el flujo de autenticación en una aplicación web construida con Spring Boot. A través de un formulario de inicio de sesión sencillo, se permite al usuario ingresar su nombre y contraseña, los cuales son procesados por un controlador que valida la información y responde adecuadamente, mostrando una página de bienvenida si los datos son correctos o devolviendo el formulario con un mensaje de error si no lo son.

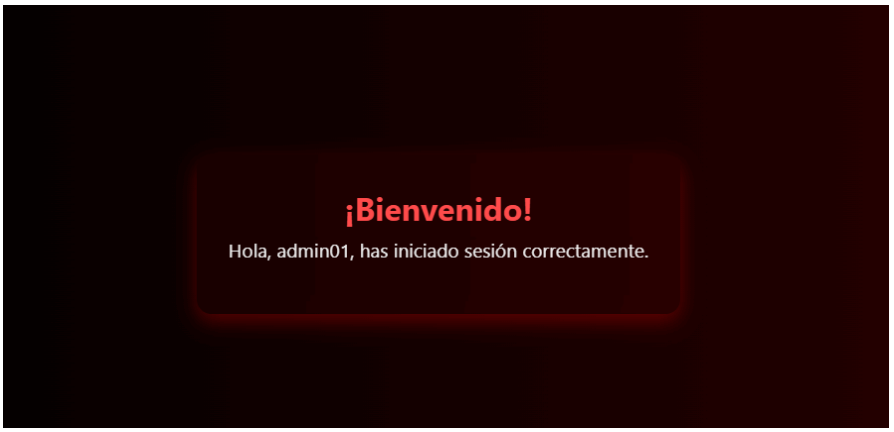
Esta interacción refleja el patrón Modelo-Vista-Controlador (MVC), donde se separan claramente los datos (modelo), la lógica de negocio (controlador) y la interfaz de usuario (vista).

Se busca reforzar el uso de tecnologías clave como Thymeleaf para el manejo dinámico de vistas y Spring Boot para simplificar la configuración del backend. El código, aunque básico, ilustra conceptos fundamentales como la inyección de modelos en las vistas, la gestión de rutas HTTP y el uso de objetos Java como puente entre el frontend y el backend. En conjunto, permite a quienes están aprendiendo a desarrollar aplicaciones web en Java comprender los principios esenciales de la autenticación de usuarios.

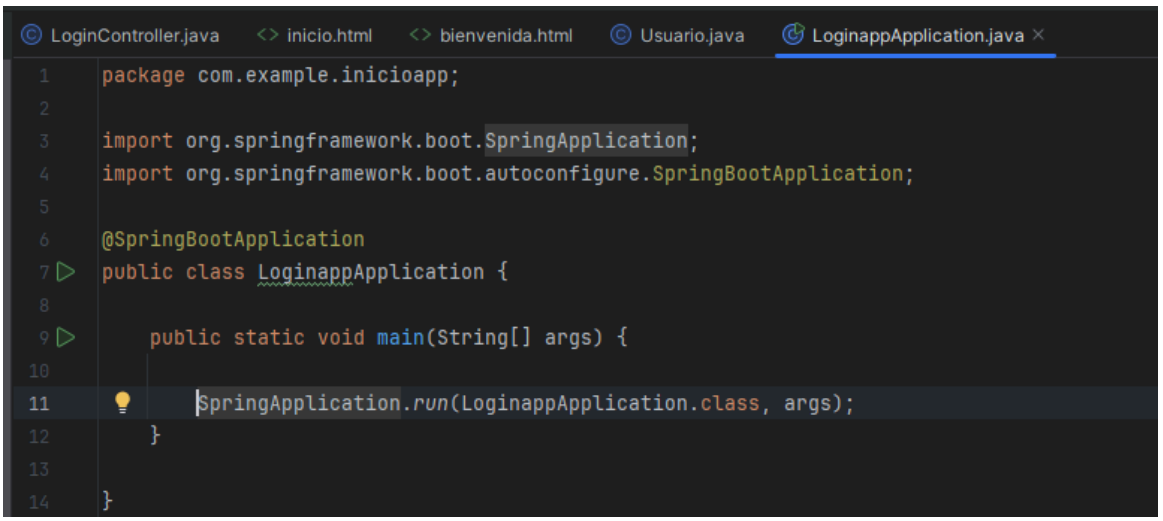
Uso del sistema:

- Ejecutar LoginAppApplication.java
- Enviar una solicitud a: http://localhost:8080/inicio
- Ingresar usuario “admin01” y “am1234”
- Con estas contraseñas el acceso será posible, en caso de escribir un usuario o contraseña diferente el acceso será denegado indicando “Usuario o contraseña incorrectos”.

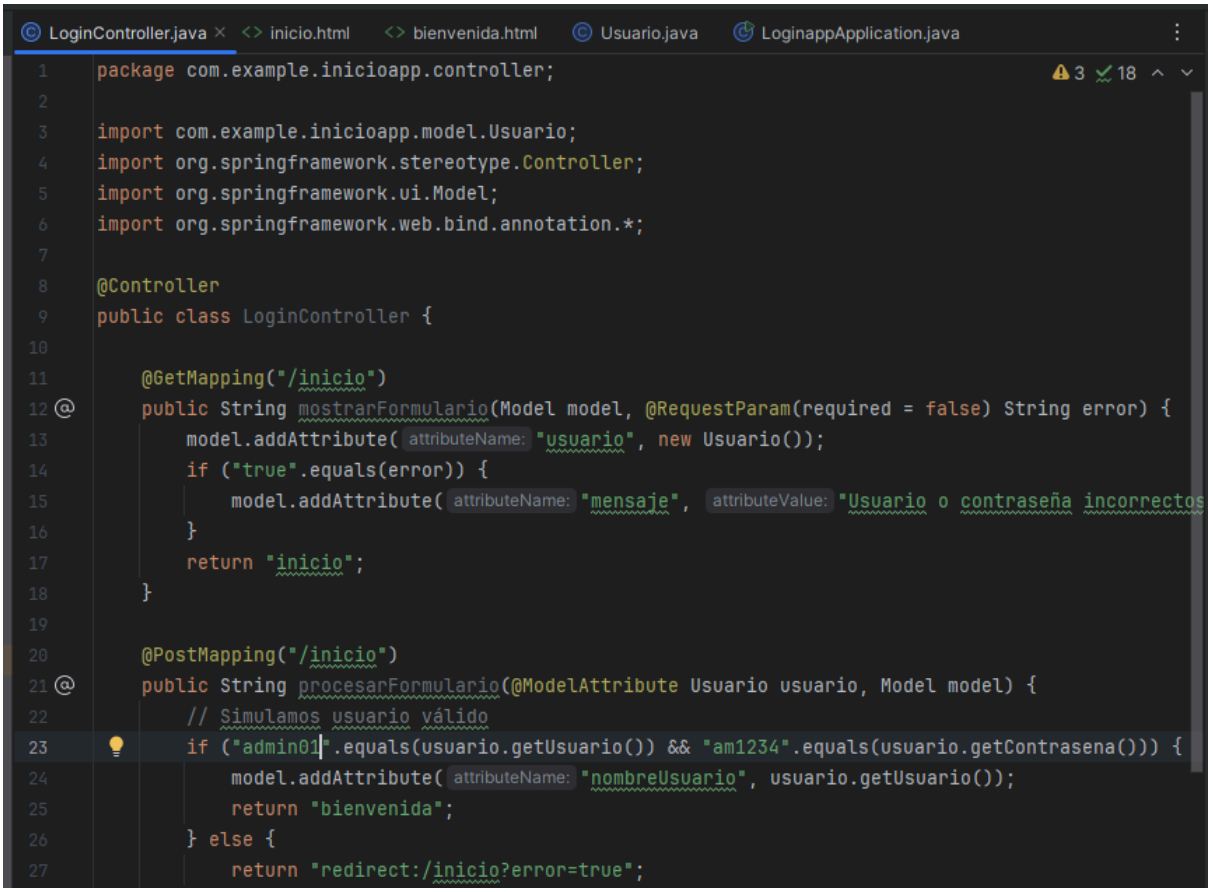




Clase principal que ejecuta el proyecto Spring Boot: LoginAppApplication.java

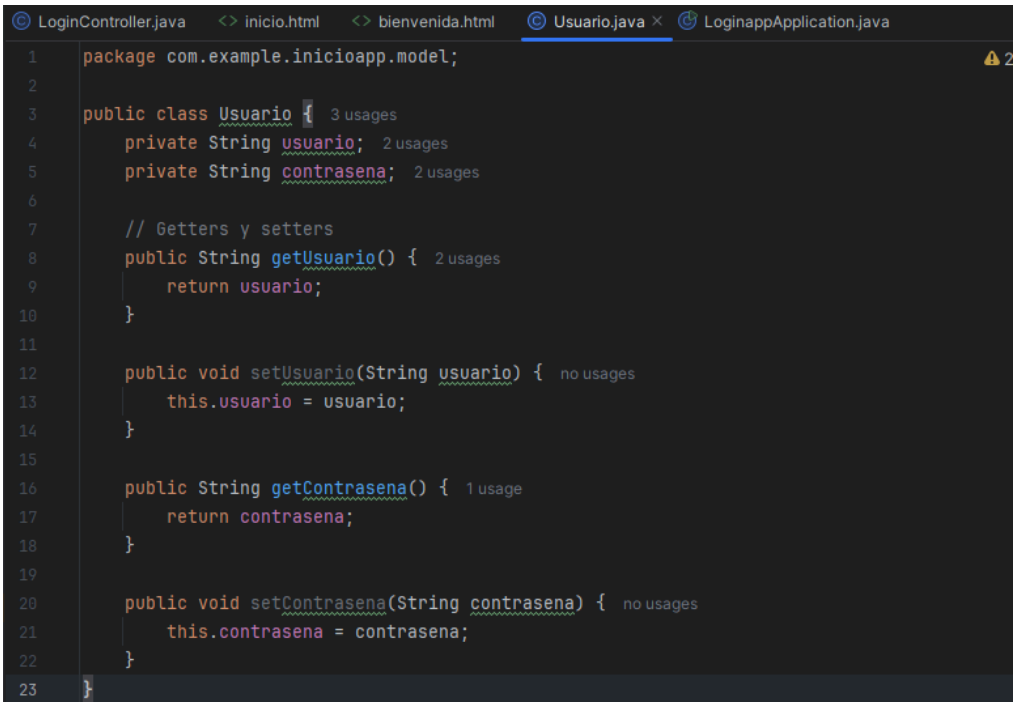


LoginController.java: Controlador que maneja las solicitudes GET y POST a /inicio. Verifica si el usuario ingresó correctamente y redirige a la vista correspondiente.



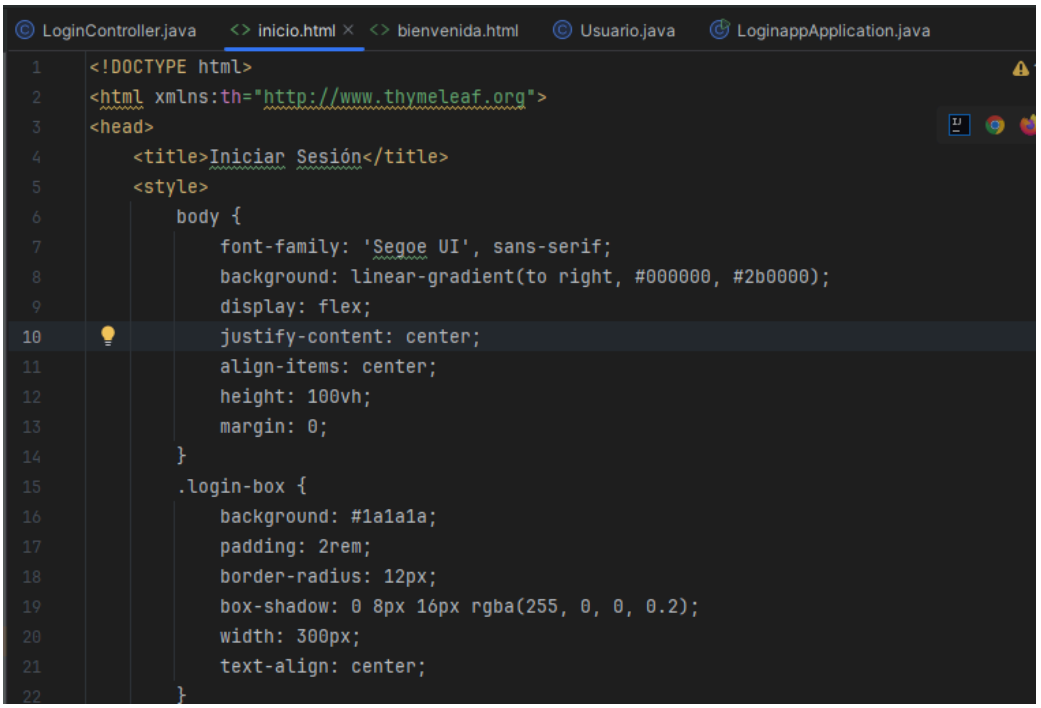
```java
package com.example.inicioapp.controller;

import com.example.inicioapp.model.Usuario;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

@Controller
public class LoginController {

    @GetMapping("/inicio")
    public String mostrarFormulario(Model model, @RequestParam(required = false) String error) {
        model.addAttribute( attributeName: "usuario", new Usuario());
        if ("true".equals(error)) {
            model.addAttribute( attributeName: "mensaje", attributeValue: "Usuario o contraseña incorrectos
        }
        return "inicio";
    }

    @PostMapping("/inicio")
    public String procesarFormulario(@ModelAttribute Usuario usuario, Model model) {
        // Simulamos usuario válido
        if ("admin01".equals(usuario.getUsuario()) && "am1234".equals(usuario.getContrasena())) {
            model.addAttribute( attributeName: "nombreUsuario", usuario.getUsuario());
            return "bienvenida";
        } else {
            return "redirect:/inicio?error=true";
```

Usuario.java: Modelo que representa los datos del formulario: usuario y contraseña.

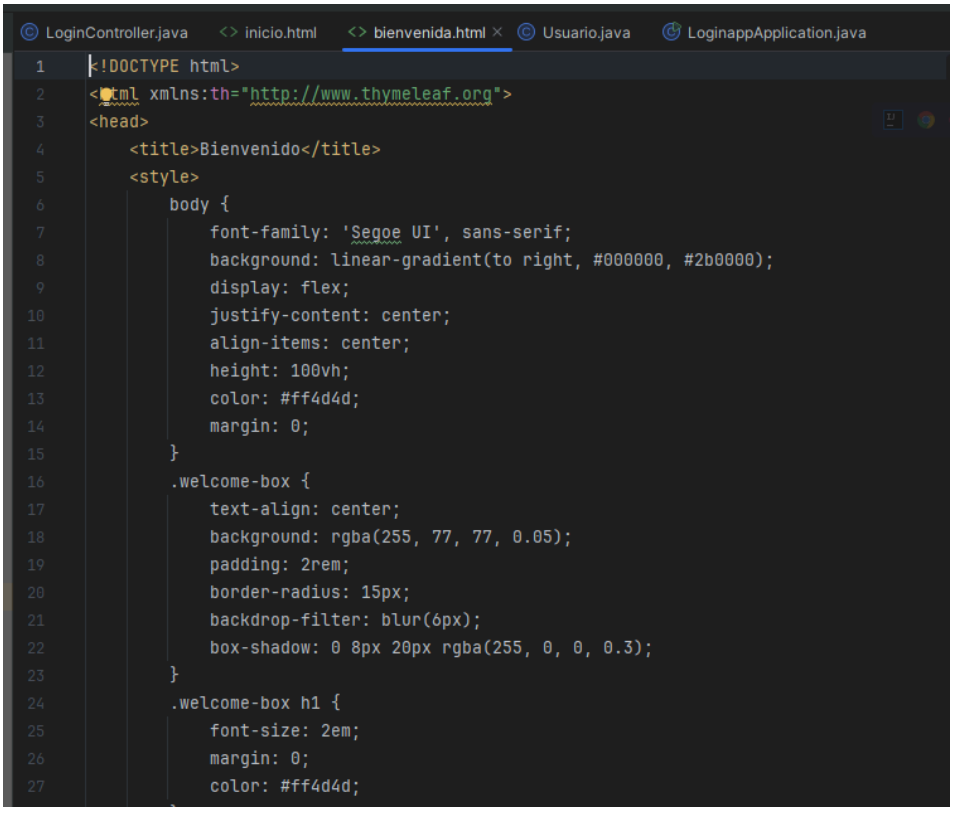


```java
package com.example.inicioapp.model;

public class Usuario {  3 usages
    private String usuario;  2 usages
    private String contrasena;  2 usages

    // Getters y setters
    public String getUsuario() {  2 usages
        return usuario;
    }

    public void setUsuario(String usuario) {  no usages
        this.usuario = usuario;
    }

    public String getContrasena() {  1 usage
        return contrasena;
    }

    public void setContrasena(String contrasena) {  no usages
        this.contrasena = contrasena;
    }
}
```

inicio.html: Es la vista del formulario de inicio de sesión. Estilizada con CSS para una mejor experiencia con campos: usuario y contraseña y botón de "Entrar" donde muestra mensaje de error si aplica



bienvenida.html: Vista que se muestra tras un login exitoso. Presenta un saludo personalizado al usuario.



Sus características destacables:

- Interfaz de inicio de sesión con diseño moderno y responsivo.
- Validación de usuario y contraseña en el backend.
- Redirección automática a una vista personalizada si el inicio es exitoso.
- Mensaje de error visible en caso de fallo en la autenticación.
- Uso de Thymeleaf para enlazar datos entre el formulario HTML y el controlador Java.
- Arquitectura basada en el patrón MVC con separación clara de responsabilidades.

Este proyecto demuestra cómo construir una aplicación web funcional y bien estructurada utilizando Spring Boot, aplicando conceptos clave como el manejo de formularios, validación de datos y renderizado dinámico de vistas con Thymeleaf. A través de un flujo sencillo de inicio de sesión, se refuerzan habilidades esenciales para cualquier desarrollador backend, como la creación de controladores, el uso de modelos Java para representar datos y la separación de responsabilidades mediante el patrón MVC.

El diseño limpio y la lógica clara hacen que este proyecto sea ideal como punto de partida para aplicaciones más complejas que requieran autenticación real con bases de datos o integración de herramientas de seguridad como Spring Security. En resumen, se trata de un ejercicio práctico que no solo cumple con su función de validar accesos, sino que también ofrece una base extensible para futuros desarrollos en entornos empresariales o académicos.