**Student & Admin Portal System - Complete Solution**

**🎯 System Overview**

**Portals:**

* **Main Home** (/) → Choose Student or Admin Portal
* **Student Portal** (/student/) → Registration & Login (no student list)
* **Admin Portal** (/admin/) → Login & View All Students

**Features:**

1. ✅ **Student Registration & Login**
2. ✅ **Admin Authentication & Dashboard**
3. ✅ **Role-based Access Control**
4. ✅ **Student Management for Admins**
5. ✅ **Separate Portals with Clean UI**

**📁 Updated Project Structure**

student-admin-portal/

├── src/main/java/com/example/demo/

│   ├── StudentSystemApplication.java

│   ├── config / DataInitializer.java

│   ├── controller/

│   │   ├── MainController.java

│   │   ├── StudentController.java

│   │   └── AdminController.java

│   │   └── TestEmailController.java

│   ├── model/

│   │   ├── Student.java

│   │   └── Admin.java

│   ├── repository/

│   │   ├── StudentRepository.java

│   │   └── AdminRepository.java

│   └── service/

│       └── EmailService.java

├── src/main/resources/

│   ├── templates/

│   │   ├── index.html (Main Home)

│   │   ├── student/

│   │   │   ├── home.html

│   │   │   ├── register.html

│   │   │   ├── login.html

│   │   │   ├── dashboard.html

│   │   │   ├── verify-otp.html

│   │   │   └── forgot-password.html

│   │   └── admin/

│   │       ├── login.html

│   │       ├── edit-student.html

│   │       └── dashboard.html

│   └── application.properties

└── pom.xml

**🛠 Step 1: pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.1.5</version>

<relativePath/>

</parent>

<groupId>com.example</groupId>

<artifactId>student-admin-portal</artifactId>

<version>1.0.0</version>

<name>student-admin-portal</name>

<description>Student and Admin Portal System</description>

<properties>

<java.version>17</java.version>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

<dependencies>

<!-- Spring Boot Web -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- Thymeleaf -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<!-- JPA + MySQL -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<scope>runtime</scope>

</dependency>

<!-- Email -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-mail</artifactId>

</dependency>

<!-- Development Tools -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<!-- Test -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.11.0</version>

<configuration>

<source>17</source>

<target>17</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**🛠 Step 2: Main Application Class**

**File:** src/main/java/com/example/demo/StudentSystemApplication.java

**package com.example.demo;**

**import org.springframework.boot.SpringApplication;**

**import org.springframework.boot.autoconfigure.SpringBootApplication;**

**@SpringBootApplication**

**public class StudentSystemApplication {**

**public static void main(String[] args) {**

**SpringApplication.run(StudentSystemApplication.class, args);**

**}**

**}**

**Datainitializer.java**

package com.example.demo.config;

import com.example.demo.model.Admin;

import com.example.demo.repository.AdminRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

@Component

public class DataInitializer implements CommandLineRunner {

@Autowired

private AdminRepository adminRepository;

@Override

public void run(String... args) throws Exception {

if (!adminRepository.existsByUsername("admin")) {

Admin admin = new Admin("admin", "admin123", "System Administrator");

adminRepository.save(admin);

System.out.println("Default admin created: username=admin, password=admin123");

}

}

}

**🛠 Step 3: Models**

**Student Model**

**File:** src/main/java/com/example/demo/model/Student.java

package com.example.demo.model;

import jakarta.persistence.\*;

import java.time.LocalDateTime;

@Entity

@Table(name = "students")

public class Student {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(unique = true, nullable = false)

private String studentId;

@Column(nullable = false)

private String name;

@Column(unique = true, nullable = false)

private String email;

@Column(nullable = false)

private String password;

@Column(name = "otp\_code")

private String otp;

@Column(name = "otp\_expiry")

private LocalDateTime otpExpiry;

@Column(name = "is\_verified")

private boolean verified = false;

public Student() {}

public Student(String studentId, String name, String email, String password) {

this.studentId = studentId;

this.name = name;

this.email = email;

this.password = password;

this.verified = false;

}

public Long getId() { return id; }

public void setId(Long id) { this.id = id; }

public String getStudentId() { return studentId; }

public void setStudentId(String studentId) { this.studentId = studentId; }

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public String getEmail() { return email; }

public void setEmail(String email) { this.email = email; }

public String getPassword() { return password; }

public void setPassword(String password) { this.password = password; }

public String getOtp() { return otp; }

public void setOtp(String otp) { this.otp = otp; }

public LocalDateTime getOtpExpiry() { return otpExpiry; }

public void setOtpExpiry(LocalDateTime otpExpiry) { this.otpExpiry = otpExpiry; }

public boolean isVerified() { return verified; }

public void setVerified(boolean verified) { this.verified = verified; }

}

**Admin Model**

**File:** src/main/java/com/example/demo/model/Admin.java

**package com.example.demo.model;**

**import jakarta.persistence.\*;**

**@Entity**

**@Table(name = "admins")**

**public class Admin {**

**@Id**

**@GeneratedValue(strategy = GenerationType.*IDENTITY*)**

**private Long id;**

**@Column(unique = true, nullable = false)**

**private String username;**

**@Column(nullable = false)**

**private String password;**

**@Column(name = "full\_name")**

**private String fullName;**

**@Column(nullable = false)**

**private boolean active = true;**

**// Constructors**

**public Admin() {}**

**public Admin(String username, String password, String fullName) {**

**this.username = username;**

**this.password = password;**

**this.fullName = fullName;**

**}**

**// Getters and Setters**

**public Long getId() {**

**return id;**

**}**

**public void setId(Long id) {**

**this.id = id;**

**}**

**public String getUsername() {**

**return username;**

**}**

**public void setUsername(String username) {**

**this.username = username;**

**}**

**public String getPassword() {**

**return password;**

**}**

**public void setPassword(String password) {**

**this.password = password;**

**}**

**public String getFullName() {**

**return fullName;**

**}**

**public void setFullName(String fullName) {**

**this.fullName = fullName;**

**}**

**public boolean isActive() {**

**return active;**

**}**

**public void setActive(boolean active) {**

**this.active = active;**

**}**

**}**

**🛠 Step 4: Repositories**

**Student Repository**

**File:** src/main/java/com/example/demo/repository/StudentRepository.java

package com.example.demo.repository;

import com.example.demo.model.Student;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import java.util.Optional;

@Repository

public interface StudentRepository extends JpaRepository<Student, Long> {

Optional<Student> findByStudentId(String studentId);

Optional<Student> findByEmail(String email);

boolean existsByStudentId(String studentId);

boolean existsByEmail(String email);

}

**Admin Repository**

**File:** src/main/java/com/example/demo/repository/AdminRepository.java

package com.example.demo.repository;

import com.example.demo.model.Admin;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import java.util.Optional;

@Repository

public interface AdminRepository extends JpaRepository<Admin, Long> {

Optional<Admin> findByUsername(String username);

boolean existsByUsername(String username);

}

**🛠 Step 5: Email Service**

**File:** src/main/java/com/example/demo/service/EmailService.java

**package com.example.demo.service;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.beans.factory.annotation.Value;**

**import org.springframework.mail.SimpleMailMessage;**

**import org.springframework.mail.javamail.JavaMailSender;**

**import org.springframework.stereotype.Service;**

**import java.util.Random;**

**@Service**

**public class EmailService {**

**@Autowired**

**private JavaMailSender mailSender;**

**@Value("${spring.mail.username}")**

**private String fromEmail;**

**public void sendOTPEmail(String toEmail, String otp, String studentName) {**

**try {**

**System.*out*.println("=== EMAIL DEBUG INFO ===");**

**System.*out*.println("From email: " + fromEmail);**

**System.*out*.println("To email: " + toEmail);**

**System.*out*.println("OTP: " + otp);**

**System.*out*.println("Student name: " + studentName);**

**System.*out*.println("========================");**

**SimpleMailMessage message = new SimpleMailMessage();**

**message.setFrom(fromEmail);**

**message.setTo(toEmail);**

**message.setSubject("Email Verification - Student Portal");**

**message.setText("Hello " + studentName + ",\n\n" +**

**"Thank you for registering with Student Portal!\n\n" +**

**"Your OTP for email verification is: " + otp + "\n\n" +**

**"This OTP will expire in 10 minutes.\n" +**

**"Please enter this OTP to complete your registration.\n\n" +**

**"If you didn't request this, please ignore this email.\n\n" +**

**"Best regards,\nStudent Portal Team");**

**mailSender.send(message);**

**System.*out*.println("✅ OTP email sent successfully to: " + toEmail);**

**} catch (org.springframework.mail.MailAuthenticationException e) {**

**System.*err*.println("❌ MAIL AUTHENTICATION FAILED!");**

**System.*err*.println("Error: " + e.getMessage());**

**System.*err*.println("Check your Gmail App Password in application.properties");**

**throw new RuntimeException("Email authentication failed. Please check Gmail app password configuration.");**

**} catch (org.springframework.mail.MailSendException e) {**

**System.*err*.println("❌ MAIL SEND EXCEPTION!");**

**System.*err*.println("Error: " + e.getMessage());**

**if (e.getMessage().contains("Invalid Addresses")) {**

**throw new RuntimeException("Invalid email address format.");**

**}**

**throw new RuntimeException("Failed to send email. Please check network connection and email configuration.");**

**} catch (Exception e) {**

**System.*err*.println("❌ UNEXPECTED EMAIL ERROR!");**

**System.*err*.println("Error type: " + e.getClass().getSimpleName());**

**System.*err*.println("Error message: " + e.getMessage());**

**e.printStackTrace();**

**throw new RuntimeException("Failed to send OTP email: " + e.getMessage());**

**}**

**}**

**public void sendResetPasswordEmail(String toEmail, String newPassword) {**

**try {**

**SimpleMailMessage message = new SimpleMailMessage();**

**message.setFrom(fromEmail);**

**message.setTo(toEmail);**

**message.setSubject("Password Reset - Student Portal");**

**message.setText("Hello,\n\nYour password has been reset successfully.\n" +**

**"Your new password is: " + newPassword +**

**"\n\nPlease login and change your password.\n\n" +**

**"Best regards,\nStudent Portal Team");**

**mailSender.send(message);**

**System.*out*.println("✅ Password reset email sent successfully to: " + toEmail);**

**} catch (Exception e) {**

**System.*err*.println("❌ Failed to send password reset email: " + e.getMessage());**

**e.printStackTrace();**

**throw new RuntimeException("Failed to send password reset email: " + e.getMessage());**

**}**

**}**

**public String generateOTP() {**

**Random random = new Random();**

**int otp = 100000 + random.nextInt(900000);**

**return String.*valueOf*(otp);**

**}**

**public boolean testEmailConnection() {**

**try {**

**System.*out*.println("🧪 Testing email configuration...");**

**SimpleMailMessage message = new SimpleMailMessage();**

**message.setFrom(fromEmail);**

**message.setTo(fromEmail);**

**message.setSubject("Test Email - Student Portal");**

**message.setText("This is a test email to verify email configuration.\n" +**

**"If you receive this, your email setup is working correctly!");**

**mailSender.send(message);**

**System.*out*.println("✅ Test email sent successfully!");**

**return true;**

**} catch (Exception e) {**

**System.*err*.println("❌ Test email failed!");**

**System.*err*.println("Error: " + e.getMessage());**

**e.printStackTrace();**

**return false;**

**}**

**}**

**}**

**🛠 Step 6: Controllers**

**Main Controller**

**File:** src/main/java/com/example/demo/controller/MainController.java

package com.example.demo.controller;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

@Controller

public class MainController {

@GetMapping("/")

public String index(Model model) {

model.addAttribute("title", "Educational Portal");

return "index";

}

}

**Student Controller**

**File:** src/main/java/com/example/demo/controller/StudentController.java

package com.example.demo.controller;

import com.example.demo.model.Student;

import com.example.demo.repository.StudentRepository;

import com.example.demo.service.EmailService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.\*;

import jakarta.servlet.http.HttpSession;

import java.time.LocalDateTime;

import java.util.Optional;

import java.util.UUID;

@Controller

@RequestMapping("/student")

public class StudentController {

@Autowired

private StudentRepository studentRepository;

@Autowired

private EmailService emailService;

// Student Portal

@GetMapping("/")

public String studentHome(Model model) {

return "student/home";

}

@GetMapping("/register")

public String showRegisterForm(Model model) {

model.addAttribute("student", new Student());

return "student/register";

}

@PostMapping("/register")

public String register(@ModelAttribute Student student, Model model, HttpSession session) {

try {

if (student.getStudentId() == null || student.getStudentId().trim().isEmpty()) {

model.addAttribute("error", "Student ID is required!");

model.addAttribute("student", new Student());

return "student/register";

}

if (student.getName() == null || student.getName().trim().isEmpty()) {

model.addAttribute("error", "Name is required!");

model.addAttribute("student", new Student());

return "student/register";

}

if (student.getEmail() == null || student.getEmail().trim().isEmpty()) {

model.addAttribute("error", "Email is required!");

model.addAttribute("student", new Student());

return "student/register";

}

if (student.getPassword() == null || student.getPassword().trim().isEmpty()) {

model.addAttribute("error", "Password is required!");

model.addAttribute("student", new Student());

return "student/register";

}

student.setStudentId(student.getStudentId().trim());

student.setName(student.getName().trim());

student.setEmail(student.getEmail().trim().toLowerCase());

if (studentRepository.existsByStudentId(student.getStudentId())) {

model.addAttribute("error", "Student ID already exists!");

model.addAttribute("student", new Student());

return "student/register";

}

if (studentRepository.existsByEmail(student.getEmail())) {

model.addAttribute("error", "Email already exists!");

model.addAttribute("student", new Student());

return "student/register";

}

String otp = emailService.generateOTP();

student.setOtp(otp);

student.setOtpExpiry(LocalDateTime.now().plusMinutes(10));

student.setVerified(false);

Student savedStudent = studentRepository.save(student);

try {

emailService.sendOTPEmail(student.getEmail(), otp, student.getName());

session.setAttribute("pendingStudentId", savedStudent.getId());

session.setAttribute("pendingStudentEmail", savedStudent.getEmail());

model.addAttribute("message", "Registration initiated! Please check your email for OTP verification.");

model.addAttribute("email", student.getEmail());

return "student/verify-otp";

} catch (Exception emailException) {

studentRepository.deleteById(savedStudent.getId());

model.addAttribute("error", "Failed to send verification email. Please try again later.");

model.addAttribute("student", new Student());

return "student/register";

}

} catch (Exception e) {

model.addAttribute("error", "Registration failed: " + e.getMessage());

model.addAttribute("student", new Student());

return "student/register";

}

}

@GetMapping("/verify-otp")

public String showOtpVerification(Model model, HttpSession session) {

Long pendingStudentId = (Long) session.getAttribute("pendingStudentId");

String pendingStudentEmail = (String) session.getAttribute("pendingStudentEmail");

if (pendingStudentId == null || pendingStudentEmail == null) {

model.addAttribute("error", "Session expired. Please register again.");

return "redirect:/student/register";

}

model.addAttribute("email", pendingStudentEmail);

return "student/verify-otp";

}

@PostMapping("/verify-otp")

public String verifyOtp(@RequestParam String otp, Model model, HttpSession session) {

Long pendingStudentId = (Long) session.getAttribute("pendingStudentId");

if (pendingStudentId == null) {

model.addAttribute("error", "Session expired. Please register again.");

return "redirect:/student/register";

}

Optional<Student> studentOpt = studentRepository.findById(pendingStudentId);

if (studentOpt.isEmpty()) {

model.addAttribute("error", "Student not found. Please register again.");

return "redirect:/student/register";

}

Student student = studentOpt.get();

if (student.getOtp() != null && student.getOtp().equals(otp.trim()) &&

student.getOtpExpiry() != null && LocalDateTime.now().isBefore(student.getOtpExpiry())) {

student.setVerified(true);

student.setOtp(null); // Clear OTP

student.setOtpExpiry(null);

studentRepository.save(student);

session.removeAttribute("pendingStudentId");

session.removeAttribute("pendingStudentEmail");

model.addAttribute("success", "Email verified successfully! You can now login.");

return "student/login";

} else {

model.addAttribute("error", "Invalid or expired OTP. Please try again.");

model.addAttribute("email", student.getEmail());

return "student/verify-otp";

}

}

@PostMapping("/resend-otp")

public String resendOtp(Model model, HttpSession session) {

Long pendingStudentId = (Long) session.getAttribute("pendingStudentId");

if (pendingStudentId == null) {

return "redirect:/student/register";

}

Optional<Student> studentOpt = studentRepository.findById(pendingStudentId);

if (studentOpt.isEmpty()) {

return "redirect:/student/register";

}

Student student = studentOpt.get();

try {

String newOtp = emailService.generateOTP();

student.setOtp(newOtp);

student.setOtpExpiry(LocalDateTime.now().plusMinutes(10));

studentRepository.save(student);

emailService.sendOTPEmail(student.getEmail(), newOtp, student.getName());

model.addAttribute("message", "New OTP sent to your email!");

model.addAttribute("email", student.getEmail());

return "student/verify-otp";

} catch (Exception e) {

model.addAttribute("error", "Failed to resend OTP. Please try again.");

model.addAttribute("email", student.getEmail());

return "student/verify-otp";

}

}

@GetMapping("/login")

public String showLoginForm(Model model) {

return "student/login";

}

@PostMapping("/login")

public String login(@RequestParam String studentId,

@RequestParam String password,

HttpSession session,

Model model) {

try {

Optional<Student> studentOpt = studentRepository.findByStudentId(studentId.trim());

if (studentOpt.isPresent()) {

Student student = studentOpt.get();

if (!student.isVerified()) {

model.addAttribute("error", "Please verify your email first! Check your email for OTP.");

return "student/login";

}

if (student.getPassword().equals(password)) {

session.setAttribute("student", student);

return "redirect:/student/dashboard";

} else {

model.addAttribute("error", "Invalid Student ID or Password!");

return "student/login";

}

} else {

model.addAttribute("error", "Invalid Student ID or Password!");

return "student/login";

}

} catch (Exception e) {

model.addAttribute("error", "Login failed. Please try again.");

return "student/login";

}

}

@GetMapping("/dashboard")

public String dashboard(HttpSession session, Model model) {

Student student = (Student) session.getAttribute("student");

if (student == null) {

return "redirect:/student/login";

}

Optional<Student> refreshedStudent = studentRepository.findById(student.getId());

if (refreshedStudent.isPresent()) {

model.addAttribute("student", refreshedStudent.get());

} else {

return "redirect:/student/login";

}

return "student/dashboard";

}

@GetMapping("/logout")

public String logout(HttpSession session) {

session.invalidate();

return "redirect:/student/";

}

@GetMapping("/forgot-password")

public String forgotPasswordPage() {

return "student/forgot-password";

}

@PostMapping("/forgot-password")

public String forgotPassword(@RequestParam String email, Model model) {

try {

Optional<Student> studentOpt = studentRepository.findByEmail(email.trim().toLowerCase());

if (studentOpt.isPresent()) {

Student student = studentOpt.get();

if (!student.isVerified()) {

model.addAttribute("error", "Please verify your email first!");

return "student/forgot-password";

}

String newPassword = UUID.randomUUID().toString().substring(0, 8);

student.setPassword(newPassword);

studentRepository.save(student);

emailService.sendResetPasswordEmail(email, newPassword);

model.addAttribute("message", "New password has been sent to your email!");

} else {

model.addAttribute("error", "Email not found in our records!");

}

} catch (Exception e) {

model.addAttribute("error", "Failed to process request. Please try again.");

}

return "student/forgot-password";

}

}

**Admin Controller**

**File:** src/main/java/com/example/demo/controller/AdminController.java

package com.example.demo.controller;

import com.example.demo.model.Admin;

import com.example.demo.model.Student;

import com.example.demo.repository.AdminRepository;

import com.example.demo.repository.StudentRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.\*;

import jakarta.servlet.http.HttpSession;

import java.util.Optional;

@Controller

@RequestMapping("/admin")

public class AdminController {

@Autowired

private AdminRepository adminRepository;

@Autowired

private StudentRepository studentRepository;

@GetMapping("/")

public String adminHome() {

return "redirect:/admin/login";

}

@GetMapping("/login")

public String showLoginForm(Model model) {

return "admin/login";

}

@PostMapping("/login")

public String login(@RequestParam String username,

@RequestParam String password,

HttpSession session,

Model model) {

Optional<Admin> adminOpt = adminRepository.findByUsername(username);

if (adminOpt.isPresent() && adminOpt.get().getPassword().equals(password)) {

session.setAttribute("admin", adminOpt.get());

return "redirect:/admin/dashboard";

} else {

model.addAttribute("error", "Invalid username or password!");

return "admin/login";

}

}

@GetMapping("/dashboard")

public String dashboard(HttpSession session, Model model) {

Admin admin = (Admin) session.getAttribute("admin");

if (admin == null) {

return "redirect:/admin/login";

}

model.addAttribute("admin", admin);

model.addAttribute("students", studentRepository.findAll());

model.addAttribute("totalStudents", studentRepository.count());

return "admin/dashboard";

}

@GetMapping("/edit-student/{id}")

public String showEditStudentForm(@PathVariable Long id, HttpSession session, Model model) {

Admin admin = (Admin) session.getAttribute("admin");

if (admin == null) {

return "redirect:/admin/login";

}

Optional<Student> studentOpt = studentRepository.findById(id);

if (studentOpt.isEmpty()) {

// Student not found, redirect back to dashboard with error

return "redirect:/admin/dashboard?error=Student not found";

}

model.addAttribute("student", studentOpt.get());

return "admin/edit-student";

}

@PostMapping("/edit-student/{id}")

public String editStudent(@PathVariable Long id,

@ModelAttribute Student updatedStudent,

HttpSession session,

Model model) {

Admin admin = (Admin) session.getAttribute("admin");

if (admin == null) {

return "redirect:/admin/login";

}

try {

Optional<Student> existingStudentOpt = studentRepository.findById(id);

if (existingStudentOpt.isEmpty()) {

model.addAttribute("error", "Student not found!");

return "redirect:/admin/dashboard";

}

Student existingStudent = existingStudentOpt.get();

if (updatedStudent.getStudentId() == null || updatedStudent.getStudentId().trim().isEmpty()) {

model.addAttribute("error", "Student ID is required!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

}

if (updatedStudent.getName() == null || updatedStudent.getName().trim().isEmpty()) {

model.addAttribute("error", "Name is required!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

}

if (updatedStudent.getEmail() == null || updatedStudent.getEmail().trim().isEmpty()) {

model.addAttribute("error", "Email is required!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

}

if (updatedStudent.getPassword() == null || updatedStudent.getPassword().trim().isEmpty()) {

model.addAttribute("error", "Password is required!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

}

updatedStudent.setStudentId(updatedStudent.getStudentId().trim());

updatedStudent.setName(updatedStudent.getName().trim());

updatedStudent.setEmail(updatedStudent.getEmail().trim().toLowerCase());

updatedStudent.setPassword(updatedStudent.getPassword().trim());

Optional<Student> duplicateStudentId = studentRepository.findByStudentId(updatedStudent.getStudentId());

if (duplicateStudentId.isPresent() && !duplicateStudentId.get().getId().equals(id)) {

model.addAttribute("error", "Student ID already exists for another student!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

}

Optional<Student> duplicateEmail = studentRepository.findByEmail(updatedStudent.getEmail());

if (duplicateEmail.isPresent() && !duplicateEmail.get().getId().equals(id)) {

model.addAttribute("error", "Email already exists for another student!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

}

existingStudent.setStudentId(updatedStudent.getStudentId());

existingStudent.setName(updatedStudent.getName());

existingStudent.setEmail(updatedStudent.getEmail());

existingStudent.setPassword(updatedStudent.getPassword());

studentRepository.save(existingStudent);

model.addAttribute("success", "Student updated successfully!");

model.addAttribute("student", existingStudent);

return "admin/edit-student";

} catch (Exception e) {

System.*err*.println("Error updating student: " + e.getMessage());

e.printStackTrace();

// Get the original student data to display in case of error

Optional<Student> originalStudent = studentRepository.findById(id);

if (originalStudent.isPresent()) {

model.addAttribute("student", originalStudent.get());

}

model.addAttribute("error", "Failed to update student. Please try again.");

return "admin/edit-student";

}

}

// Delete Student

@PostMapping("/delete-student/{id}")

public String deleteStudent(@PathVariable Long id, HttpSession session) {

Admin admin = (Admin) session.getAttribute("admin");

if (admin == null) {

return "redirect:/admin/login";

}

try {

studentRepository.deleteById(id);

} catch (Exception e) {

// Handle delete error silently or add flash message

System.*err*.println("Error deleting student: " + e.getMessage());

}

return "redirect:/admin/dashboard";

}

// Admin Logout

@GetMapping("/logout")

public String logout(HttpSession session) {

session.invalidate();

return "redirect:/admin/login";

}

}

**TestEmail Controller**

**File:** src/main/java/com/example/demo/controller/TestEmailController.java

**package** com.example.demo.controller;

**import** com.example.demo.service.EmailService;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

**public** **class** TestEmailController {

@Autowired

**private** EmailService emailService;

@GetMapping("/test-email")

**public** String testEmail() {

**try** {

**boolean** success = emailService.testEmailConnection();

**if** (success) {

**return** "✅ Email configuration is working! Check your email inbox.";

} **else** {

**return** "❌ Email configuration failed. Check console logs for details.";

}

} **catch** (Exception e) {

**return** "❌ Email test failed: " + e.getMessage();

}

}

@GetMapping("/test-otp")

**public** String testOTP() {

**try** {

String otp = emailService.generateOTP();

emailService.sendOTPEmail("testmail.project2004@gmail.com", otp, "Test User");

**return** "✅ OTP email sent successfully! OTP: " + otp;

} **catch** (Exception e) {

**return** "❌ OTP email failed: " + e.getMessage();

}

}

}

**🛠 Step 7: HTML Templates**

**Main Home Page**

**File:** src/main/resources/templates/index.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Educational Portal - Home</title>

<style>

/\* Base and Reset \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f4f6fb;

color: #333;

min-height: 100vh;

display: flex;

align-items: center;

justify-content: center;

padding: 20px;

}

.container {

text-align: center;

max-width: 700px;

background-color: #fff;

padding: 40px 30px;

border-radius: 12px;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.08);

}

.title {

font-size: 2.5rem;

color: #2c3e50;

margin-bottom: 10px;

}

.subtitle {

font-size: 1.1rem;

color: #666;

margin-bottom: 30px;

}

.portal-grid {

display: flex;

flex-wrap: wrap;

justify-content: center;

gap: 30px;

}

.portal-card {

background-color: #f9f9f9;

padding: 25px 20px;

width: 250px;

border-radius: 10px;

text-decoration: none;

color: inherit;

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.05);

transition: transform 0.3s, box-shadow 0.3s;

}

.portal-card:hover {

transform: translateY(-5px);

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

}

.portal-icon {

font-size: 3rem;

margin-bottom: 15px;

}

.portal-title {

font-size: 1.4rem;

margin-bottom: 8px;

font-weight: 600;

color: #34495e;

}

.portal-desc {

font-size: 0.95rem;

color: #666;

}

.admin-card {

background-color: #f1f5ff;

}

@media (max-width: 600px) {

.portal-grid {

flex-direction: column;

gap: 20px;

}

.portal-card {

width: 100%;

}

}

</style>

</head>

<body>

<div class="container">

<h1 class="title">🎓 Educational Portal</h1>

<p class="subtitle">Choose your portal to continue</p>

<div class="portal-grid">

<a href="/student/" class="portal-card">

<div class="portal-icon">👨‍🎓</div>

<h2 class="portal-title">Student Portal</h2>

<p class="portal-desc">Register, Login & Manage your account</p>

</a>

<a href="/admin/" class="portal-card admin-card">

<div class="portal-icon">👨‍💼</div>

<h2 class="portal-title">Admin Portal</h2>

<p class="portal-desc">Manage students & system administration</p>

</a>

</div>

</div>

</body>

</html>

**Student Dashboard**

**File:** src/main/resources/templates/student/dashboard.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Student Dashboard</title>

<style>

/\* Reset some default styles \*/

\* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: #f9fafb;

color: #333;

line-height: 1.6;

min-height: 100vh;

display: flex;

justify-content: center;

align-items: flex-start;

padding: 20px;

}

.container {

background: #fff;

max-width: 900px;

width: 100%;

border-radius: 8px;

box-shadow: 0 4px 15px rgba(0, 0, 0, 0.1);

padding: 30px 40px;

}

.header {

text-align: center;

margin-bottom: 30px;

}

.header h1 {

font-size: 2.5rem;

margin-bottom: 10px;

color: #2c3e50;

}

.header p {

font-size: 1.1rem;

color: #666;

}

.welcome-message h2 {

font-size: 1.8rem;

color: #34495e;

margin-bottom: 8px;

}

.welcome-message p {

color: #555;

margin-bottom: 30px;

font-size: 1.1rem;

}

.info-grid {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(220px, 1fr));

gap: 20px;

margin-bottom: 30px;

}

.info-card {

background: #f1f5f9;

border-radius: 8px;

padding: 20px;

box-shadow: 0 1px 6px rgba(0, 0, 0, 0.05);

transition: box-shadow 0.3s ease;

}

.info-card:hover {

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);

}

.info-card h3 {

font-size: 1.2rem;

margin-bottom: 10px;

color: #34495e;

display: flex;

align-items: center;

gap: 8px;

}

.info-card p {

font-size: 1.1rem;

color: #222;

word-break: break-word;

}

.actions {

text-align: center;

}

.logout-btn {

display: inline-block;

background-color: #e74c3c;

color: white;

padding: 12px 28px;

border-radius: 30px;

text-decoration: none;

font-weight: 600;

font-size: 1.1rem;

box-shadow: 0 4px 8px rgba(231, 76, 60, 0.4);

transition: background-color 0.3s ease, box-shadow 0.3s ease;

}

.logout-btn:hover {

background-color: #c0392b;

box-shadow: 0 6px 14px rgba(192, 57, 43, 0.6);

}

@media (max-width: 480px) {

.container {

padding: 20px;

}

.header h1 {

font-size: 2rem;

}

.welcome-message h2 {

font-size: 1.5rem;

}

}

</style>

</head>

<body>

<div class="container">

<div class="header">

<h1>🎓 Student Dashboard</h1>

<p>Welcome to your personal dashboard</p>

</div>

<div class="content">

<div class="welcome-message">

<h2>Hello, <span th:text="${student.name}">Student Name</span>! ��</h2>

<p>You are successfully logged into the Student Portal</p>

</div>

<div class="info-grid">

<div class="info-card">

<h3>📋 Student ID</h3>

<p th:text="${student.studentId}">Student ID</p>

</div>

<div class="info-card">

<h3>👤 Full Name</h3>

<p th:text="${student.name}">Student Name</p>

</div>

<div class="info-card">

<h3>✉️ Email Address</h3>

<p th:text="${student.email}">student@email.com</p>

</div>

<div class="info-card">

<h3>🔐 Account Status</h3>

<p style="color: #28a745; font-weight: 600;">Active</p>

</div>

</div>

<div class="actions">

<a href="/student/logout" class="logout-btn">Logout</a>

</div>

</div>

</div>

</body>

</html>

**Student Forgot Password**

**File:** src/main/resources/templates/student/forgot-password.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Forgot Password</title>

<style>

/\* Reset \*/

\* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f3f6f9;

color: #333;

min-height: 100vh;

display: flex;

justify-content: center;

align-items: center;

padding: 20px;

}

.container {

background: #fff;

max-width: 400px;

width: 100%;

padding: 30px 40px;

border-radius: 10px;

box-shadow: 0 6px 15px rgba(0,0,0,0.1);

text-align: center;

}

.back-link {

display: inline-block;

margin-bottom: 20px;

font-size: 0.9rem;

color: #555;

text-decoration: none;

transition: color 0.3s ease;

}

.back-link:hover {

color: #f5576c;

}

.title {

font-size: 2rem;

margin-bottom: 10px;

color: #2c3e50;

}

.description {

font-size: 1rem;

color: #555;

margin-bottom: 25px;

}

.error, .message {

margin-bottom: 20px;

padding: 12px 15px;

border-radius: 6px;

font-weight: 600;

font-size: 0.95rem;

}

.error {

background-color: #fdecea;

color: #e74c3c;

border: 1px solid #e74c3c;

}

.message {

background-color: #e6ffed;

color: #27ae60;

border: 1px solid #27ae60;

}

form {

text-align: left;

}

.form-group {

margin-bottom: 20px;

}

label {

display: block;

font-weight: 600;

margin-bottom: 8px;

color: #34495e;

}

input[type="email"] {

width: 100%;

padding: 12px 15px;

border: 1.5px solid #ccc;

border-radius: 8px;

font-size: 1rem;

transition: border-color 0.3s ease;

}

input[type="email"]:focus {

border-color: #f5576c;

outline: none;

box-shadow: 0 0 5px rgba(245, 87, 108, 0.5);

}

.btn {

width: 100%;

background-color: #f5576c;

color: white;

padding: 14px 0;

font-size: 1.1rem;

border: none;

border-radius: 30px;

cursor: pointer;

font-weight: 700;

box-shadow: 0 6px 12px rgba(245, 87, 108, 0.5);

transition: background-color 0.3s ease, box-shadow 0.3s ease;

}

.btn:hover {

background-color: #d94359;

box-shadow: 0 8px 16px rgba(217, 67, 89, 0.7);

}

p:last-of-type {

margin-top: 25px;

font-size: 0.95rem;

color: #666;

}

p:last-of-type a {

color: #f5576c;

font-weight: 600;

text-decoration: none;

transition: color 0.3s ease;

}

p:last-of-type a:hover {

color: #d94359;

}

@media (max-width: 480px) {

.container {

padding: 25px 20px;

}

.title {

font-size: 1.7rem;

}

.btn {

font-size: 1rem;

padding: 12px 0;

}

}

</style>

</head>

<body>

<div class="container">

<a href="/student/login" class="back-link">← Back to Login</a>

<h1 class="title">🔑 Forgot Password</h1>

<p class="description">Enter your registered email address and we'll send you a new password.</p>

<div th:if="${error}" class="error" th:text="${error}"></div>

<div th:if="${message}" class="message" th:text="${message}"></div>

<form th:action="@{/student/forgot-password}" method="post">

<div class="form-group">

<label for="email">Email Address:</label>

<input type="email" id="email" name="email" required placeholder="Enter your registered email">

</div>

<button type="submit" class="btn">Send New Password</button>

</form>

<p style="text-align: center; margin-top: 20px; color: #666;">

Remember your password? <a href="/student/login" style="color: #f5576c; text-decoration: none; font-weight: 600;">Login here</a>

</p>

</div>

</body>

</html>

**Student home**

**File:** src/main/resources/templates/student/home.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Student Portal - Home</title>

<style>

/\* Reset and base styling \*/

\* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f3f6f9;

color: #333;

min-height: 100vh;

padding: 30px 20px;

display: flex;

justify-content: center;

align-items: center;

}

.container {

max-width: 500px;

width: 100%;

background: #ffffff;

padding: 40px 30px;

border-radius: 12px;

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

text-align: center;

}

.title {

font-size: 2.2rem;

margin-bottom: 10px;

color: #2c3e50;

}

.container p {

font-size: 1rem;

color: #555;

margin-bottom: 30px;

}

.back-link {

position: absolute;

top: 20px;

left: 20px;

text-decoration: none;

color: #555;

font-size: 0.95rem;

transition: color 0.3s ease;

}

.back-link:hover {

color: #f5576c;

}

.action-grid {

display: flex;

flex-direction: column;

gap: 20px;

}

.action-btn {

display: block;

padding: 20px;

background-color: #f5576c;

color: white;

border-radius: 10px;

font-size: 1.1rem;

font-weight: 600;

text-decoration: none;

box-shadow: 0 6px 12px rgba(245, 87, 108, 0.3);

transition: background-color 0.3s ease, transform 0.2s ease;

}

.action-btn:hover {

background-color: #d94359;

transform: translateY(-3px);

}

.register-btn {

background-color: #1abc9c;

box-shadow: 0 6px 12px rgba(26, 188, 156, 0.3);

}

.register-btn:hover {

background-color: #16a085;

}

.btn-icon {

font-size: 1.5rem;

display: block;

margin-bottom: 5px;

}

@media (min-width: 600px) {

.action-grid {

flex-direction: row;

justify-content: center;

}

.action-btn {

flex: 1;

margin: 0 10px;

}

}

@media (max-width: 480px) {

.container {

padding: 30px 20px;

}

.title {

font-size: 1.8rem;

}

.action-btn {

font-size: 1rem;

padding: 16px;

}

}

</style>

</head>

<body>

<a href="/" class="back-link">← Back to Main</a>

<div class="container">

<h1 class="title">👨‍🎓 Student Portal</h1>

<p>Welcome to the Student Portal. Please choose an option below:</p>

<div class="action-grid">

<a href="/student/register" class="action-btn register-btn">

<span class="btn-icon">��</span>

<strong>Register</strong><br>

Create new account

</a>

<a href="/student/login" class="action-btn">

<span class="btn-icon">��</span>

<strong>Login</strong><br>

Access your account

</a>

</div>

</div>

</body>

</html>

**Student login**

**File:** src/main/resources/templates/student/login.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Student Login</title>

<style>

/\* Reset and base \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f3f6f9;

min-height: 100vh;

display: flex;

justify-content: center;

align-items: center;

padding: 20px;

color: #333;

}

.container {

background: #fff;

width: 100%;

max-width: 400px;

padding: 35px 30px;

border-radius: 12px;

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

position: relative;

text-align: center;

}

.back-link {

position: absolute;

top: 15px;

left: 20px;

font-size: 0.9rem;

text-decoration: none;

color: #555;

transition: color 0.3s;

}

.back-link:hover {

color: #f5576c;

}

.title {

font-size: 2rem;

margin-bottom: 20px;

color: #2c3e50;

}

.error {

background-color: #fdecea;

color: #e74c3c;

border: 1px solid #e74c3c;

border-radius: 6px;

padding: 12px;

margin-bottom: 20px;

font-weight: 600;

font-size: 0.95rem;

}

form {

text-align: left;

}

.form-group {

margin-bottom: 20px;

}

label {

display: block;

margin-bottom: 8px;

font-weight: 600;

color: #34495e;

}

input[type="text"],

input[type="password"] {

width: 100%;

padding: 12px 15px;

border-radius: 8px;

border: 1.5px solid #ccc;

font-size: 1rem;

transition: border-color 0.3s ease;

}

input:focus {

border-color: #f5576c;

outline: none;

box-shadow: 0 0 6px rgba(245, 87, 108, 0.4);

}

.btn {

width: 100%;

background-color: #f5576c;

color: white;

padding: 14px 0;

font-size: 1.1rem;

border: none;

border-radius: 30px;

font-weight: bold;

box-shadow: 0 6px 12px rgba(245, 87, 108, 0.4);

cursor: pointer;

transition: background-color 0.3s, box-shadow 0.3s;

}

.btn:hover {

background-color: #d94359;

box-shadow: 0 8px 16px rgba(217, 67, 89, 0.6);

}

.forgot-password {

margin-top: 15px;

text-align: right;

}

.forgot-password a {

color: #f5576c;

font-size: 0.9rem;

text-decoration: none;

transition: color 0.3s;

}

.forgot-password a:hover {

color: #d94359;

}

p {

font-size: 0.95rem;

margin-top: 25px;

color: #666;

}

p a {

color: #f5576c;

text-decoration: none;

font-weight: 600;

}

p a:hover {

color: #d94359;

}

@media (max-width: 480px) {

.container {

padding: 30px 20px;

}

.title {

font-size: 1.8rem;

}

.btn {

font-size: 1rem;

padding: 12px 0;

}

}

</style>

</head>

<body>

<div class="container">

<a href="/student/" class="back-link">← Back to Student Portal</a>

<h1 class="title">🔐 Student Login</h1>

<div th:if="${error}" class="error" th:text="${error}"></div>

<form th:action="@{/student/login}" method="post">

<div class="form-group">

<label for="studentId">Student ID:</label>

<input type="text" id="studentId" name="studentId" required placeholder="Enter your student ID">

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" id="password" name="password" required placeholder="Enter your password">

</div>

<button type="submit" class="btn">Login</button>

</form>

<div class="forgot-password">

<a href="/student/forgot-password">Forgot Password?</a>

</div>

<p style="text-align: center; margin-top: 20px; color: #666;">

Don't have an account? <a href="/student/register" style="color: #f5576c; text-decoration: none; font-weight: 600;">Register here</a>

</p>

</div>

</body>

</html>

**Student register**

**File:** src/main/resources/templates/student/register.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Student Registration</title>

<style>

/\* Reset and base styles \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f3f6f9;

min-height: 100vh;

display: flex;

justify-content: center;

align-items: center;

padding: 20px;

color: #333;

}

.container {

background: #fff;

width: 100%;

max-width: 450px;

padding: 35px 30px;

border-radius: 12px;

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

position: relative;

text-align: center;

}

.back-link {

position: absolute;

top: 15px;

left: 20px;

font-size: 0.9rem;

text-decoration: none;

color: #555;

transition: color 0.3s;

}

.back-link:hover {

color: #f5576c;

}

.title {

font-size: 2rem;

margin-bottom: 20px;

color: #2c3e50;

}

.error,

.success {

padding: 12px;

margin-bottom: 20px;

border-radius: 6px;

font-weight: 600;

font-size: 0.95rem;

}

.error {

background-color: #fdecea;

color: #e74c3c;

border: 1px solid #e74c3c;

}

.success {

background-color: #eafaf1;

color: #28a745;

border: 1px solid #28a745;

}

form {

text-align: left;

}

.form-group {

margin-bottom: 20px;

}

label {

display: block;

margin-bottom: 8px;

font-weight: 600;

color: #34495e;

}

input[type="text"],

input[type="email"],

input[type="password"] {

width: 100%;

padding: 12px 15px;

border-radius: 8px;

border: 1.5px solid #ccc;

font-size: 1rem;

transition: border-color 0.3s ease;

}

input:focus {

border-color: #f5576c;

outline: none;

box-shadow: 0 0 6px rgba(245, 87, 108, 0.4);

}

.btn {

width: 100%;

background-color: #f5576c;

color: white;

padding: 14px 0;

font-size: 1.1rem;

border: none;

border-radius: 30px;

font-weight: bold;

box-shadow: 0 6px 12px rgba(245, 87, 108, 0.4);

cursor: pointer;

transition: background-color 0.3s, box-shadow 0.3s;

}

.btn:hover {

background-color: #d94359;

box-shadow: 0 8px 16px rgba(217, 67, 89, 0.6);

}

p {

font-size: 0.95rem;

margin-top: 25px;

color: #666;

}

p a {

color: #f5576c;

text-decoration: none;

font-weight: 600;

}

p a:hover {

color: #d94359;

}

@media (max-width: 480px) {

.container {

padding: 30px 20px;

}

.title {

font-size: 1.8rem;

}

.btn {

font-size: 1rem;

padding: 12px 0;

}

}

</style>

</head>

<body>

<div class="container">

<a href="/student/" class="back-link">← Back to Student Portal</a>

<h1 class="title">📝 Student Registration</h1>

<div th:if="${error}" class="error" th:text="${error}"></div>

<div th:if="${success}" class="success" th:text="${success}"></div>

<form th:action="@{/student/register}" th:object="${student}" method="post">

<div class="form-group">

<label for="studentId">Student ID:</label>

<input type="text" id="studentId" th:field="\*{studentId}" required placeholder="Enter your student ID">

</div>

<div class="form-group">

<label for="name">Full Name:</label>

<input type="text" id="name" th:field="\*{name}" required placeholder="Enter your full name">

</div>

<div class="form-group">

<label for="email">Email:</label>

<input type="email" id="email" th:field="\*{email}" required placeholder="Enter your email address">

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" id="password" th:field="\*{password}" required placeholder="Create a password">

</div>

<button type="submit" class="btn">Register Now</button>

</form>

<p style="text-align: center; margin-top: 20px; color: #666;">

Already have an account? <a href="/student/login" style="color: #f5576c; text-decoration: none; font-weight: 600;">Login here</a>

</p>

</div>

</body>

</html>

**Student otp verify**

**File:** src/main/resources/templates/student/verify-otp.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Verify OTP - Student Portal</title>

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

min-height: 100vh;

display: flex;

align-items: center;

justify-content: center;

padding: 20px;

}

.container {

background: white;

padding: 40px;

border-radius: 15px;

box-shadow: 0 20px 40px rgba(0,0,0,0.1);

width: 100%;

max-width: 400px;

text-align: center;

}

.back-link {

position: absolute;

top: 20px;

left: 20px;

color: white;

text-decoration: none;

font-weight: 600;

transition: all 0.3s ease;

}

.back-link:hover {

color: #f0f0f0;

}

.title {

color: #333;

margin-bottom: 10px;

font-size: 28px;

font-weight: 700;

}

.subtitle {

color: #666;

margin-bottom: 30px;

font-size: 16px;

}

.email-info {

background: #f8f9fa;

padding: 15px;

border-radius: 8px;

margin-bottom: 20px;

border-left: 4px solid #007bff;

}

.form-group {

margin-bottom: 20px;

text-align: left;

}

label {

display: block;

margin-bottom: 8px;

color: #333;

font-weight: 600;

}

input[type="text"] {

width: 100%;

padding: 15px;

border: 2px solid #e1e5e9;

border-radius: 8px;

font-size: 16px;

transition: all 0.3s ease;

text-align: center;

letter-spacing: 2px;

font-weight: bold;

}

input[type="text"]:focus {

outline: none;

border-color: #007bff;

box-shadow: 0 0 0 3px rgba(0,123,255,0.1);

}

.btn {

width: 100%;

padding: 15px;

background: linear-gradient(135deg, #007bff, #0056b3);

color: white;

border: none;

border-radius: 8px;

font-size: 16px;

font-weight: 600;

cursor: pointer;

transition: all 0.3s ease;

margin-bottom: 15px;

}

.btn:hover {

transform: translateY(-2px);

box-shadow: 0 10px 20px rgba(0,123,255,0.3);

}

.resend-btn {

background: linear-gradient(135deg, #28a745, #20c997);

padding: 12px 20px;

font-size: 14px;

}

.error {

background: #f8d7da;

color: #721c24;

padding: 15px;

border-radius: 8px;

margin-bottom: 20px;

border: 1px solid #f5c6cb;

}

.message {

background: #d4edda;

color: #155724;

padding: 15px;

border-radius: 8px;

margin-bottom: 20px;

border: 1px solid #c3e6cb;

}

.otp-instructions {

background: #fff3cd;

color: #856404;

padding: 15px;

border-radius: 8px;

margin-bottom: 20px;

border: 1px solid #ffeaa7;

font-size: 14px;

}

.timer {

color: #dc3545;

font-weight: bold;

margin-top: 10px;

}

</style>

</head>

<body>

<a href="/student/register" class="back-link">← Back to Registration</a>

<div class="container">

<h1 class="title">📧 Verify Your Email</h1>

<p class="subtitle">Please enter the 6-digit OTP sent to your email</p>

<div class="email-info">

<strong>Email sent to:</strong><br>

<span th:text="${email}">student@email.com</span>

</div>

<div th:if="${error}" class="error" th:text="${error}"></div>

<div th:if="${message}" class="message" th:text="${message}"></div>

<form th:action="@{/student/verify-otp}" method="post">

<div class="form-group">

<label for="otp">Enter OTP:</label>

<input type="text" id="otp" name="otp" required

placeholder="000000" maxlength="6" pattern="[0-9]{6}"

title="Please enter a 6-digit OTP">

</div>

<button type="submit" class="btn">Verify OTP</button>

</form>

<form th:action="@{/student/resend-otp}" method="post" style="display: inline;">

<button type="submit" class="btn resend-btn">Resend OTP</button>

</form>

<div class="timer">

<small>Didn't receive the email? Check your spam folder or click Resend OTP</small>

</div>

</div>

<script>

// Auto-format OTP input

document.getElementById('otp').addEventListener('input', **function**(e) {

this.value = this.value.replace(/[^0-9]/g, '');

});

// Auto-submit when 6 digits are entered

document.getElementById('otp').addEventListener('input', **function**(e) {

if (this.value.length === 6) {

setTimeout(() **=>** {

this.form.submit();

}, 500);

}

});

</script>

</body>

</html>

**Admin Login**

**File:** src/main/resources/templates/admin/login.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Admin Login</title>

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f4f6fb;

padding: 20px;

color: #333;

}

.container {

max-width: 500px;

margin: auto;

background: #fff;

padding: 30px;

border-radius: 12px;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.07);

}

.back-link {

display: inline-block;

margin-bottom: 20px;

color: #2e86de;

text-decoration: none;

font-weight: bold;

}

.admin-header {

text-align: center;

margin-bottom: 25px;

}

.admin-icon {

font-size: 3rem;

}

.title {

font-size: 2rem;

color: #2c3e50;

margin-top: 10px;

}

.subtitle {

font-size: 1rem;

color: #777;

margin-top: 5px;

}

.form-group {

margin-bottom: 20px;

}

label {

display: block;

font-weight: 600;

margin-bottom: 6px;

}

input[type="text"],

input[type="password"] {

width: 100%;

padding: 10px 12px;

border: 1px solid #ccc;

border-radius: 6px;

font-size: 1rem;

transition: border-color 0.3s;

}

input:focus {

border-color: #2e86de;

outline: none;

}

.btn {

width: 100%;

background-color: #2e86de;

color: #fff;

border: none;

padding: 12px;

border-radius: 6px;

font-size: 1rem;

font-weight: bold;

cursor: pointer;

transition: background-color 0.3s;

}

.btn:hover {

background-color: #1b4f72;

}

.error {

background-color: #ffe6e6;

color: #cc0000;

padding: 10px;

border-radius: 5px;

margin-bottom: 20px;

text-align: center;

}

.admin-info {

margin-top: 25px;

background: #f9f9f9;

border-left: 5px solid #f39c12;

padding: 10px 15px;

font-size: 0.95rem;

color: #555;

}

@media (max-width: 600px) {

.container {

padding: 20px;

}

.admin-icon {

font-size: 2.5rem;

}

.title {

font-size: 1.6rem;

}

}

</style>

</head>

<body>

<div class="container">

<a href="/" class="back-link">← Back to Main</a>

<div class="admin-header">

<div class="admin-icon">👨‍💼</div>

<h1 class="title">Admin Portal</h1>

<p class="subtitle">Administrative Access Only</p>

</div>

<div th:if="${error}" class="error" th:text="${error}"></div>

<form th:action="@{/admin/login}" method="post">

<div class="form-group">

<label for="username">Admin Username:</label>

<input type="text" id="username" name="username" required placeholder="Enter admin username">

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" id="password" name="password" required placeholder="Enter admin password">

</div>

<button type="submit" class="btn">Admin Login</button>

</form>

<div class="admin-info">

<strong>Note:</strong> This area is restricted to authorized administrators only.

</div>

</div>

</body>

</html>

**Edit student**

**File:** src/main/resources/templates/admin/edit-student.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Edit Student - Admin Portal</title>

<style>

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 20px;

min-height: 100vh;

}

.container {

max-width: 600px;

margin: 0 auto;

background: white;

border-radius: 10px;

box-shadow: 0 10px 30px rgba(0,0,0,0.3);

overflow: hidden;

}

.header {

background: grey;

color: white;

padding: 30px;

text-align: center;

}

.back-link {

display: inline-block;

color: white;

text-decoration: none;

margin-bottom: 15px;

font-weight: 500;

}

.back-link:hover {

text-decoration: underline;

}

.content {

padding: 30px;

}

.form-group {

margin-bottom: 20px;

}

label {

display: block;

margin-bottom: 8px;

font-weight: 600;

color: #333;

}

input[type="text"],

input[type="email"],

input[type="password"] {

width: 100%;

padding: 12px;

border: 2px solid #e1e5e9;

border-radius: 8px;

font-size: 16px;

transition: border-color 0.3s;

box-sizing: border-box;

}

input:focus {

outline: none;

border-color: #667eea;

}

.btn {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: white;

padding: 12px 30px;

border: none;

border-radius: 25px;

font-size: 16px;

font-weight: 600;

cursor: pointer;

transition: transform 0.2s;

margin-right: 10px;

}

.btn:hover {

transform: translateY(-2px);

}

.cancel-btn {

background: #6c757d;

color: white;

padding: 12px 30px;

border: none;

border-radius: 25px;

font-size: 16px;

font-weight: 600;

cursor: pointer;

text-decoration: none;

display: inline-block;

}

.cancel-btn:hover {

background: #5a6268;

text-decoration: none;

color: white;

}

.error {

background: #f8d7da;

color: #721c24;

padding: 12px;

border-radius: 8px;

margin-bottom: 20px;

border: 1px solid #f5c6cb;

}

.success {

background: #d4edda;

color: #155724;

padding: 12px;

border-radius: 8px;

margin-bottom: 20px;

border: 1px solid #c3e6cb;

}

.student-info {

background: #f8f9fa;

padding: 15px;

border-radius: 8px;

margin-bottom: 20px;

}

</style>

</head>

<body>

<div class="container">

<div class="header">

<a href="/admin/dashboard" class="back-link">← Back to Dashboard</a>

<h1>✏️ Edit Student</h1>

<p>Modify student information</p>

</div>

<div class="content">

<div class="student-info">

<strong>Editing Student:</strong> <span th:text="${student.name}">Student Name</span><br>

<strong>Current ID:</strong> <span th:text="${student.studentId}">Student ID</span>

</div>

<div th:if="${error}" class="error" th:text="${error}"></div>

<div th:if="${success}" class="success" th:text="${success}"></div>

<form th:action="@{/admin/edit-student/{id}(id=${student.id})}" th:object="${student}" method="post">

<div class="form-group">

<label for="studentId">Student ID:</label>

<input type="text" id="studentId" th:field="\*{studentId}" required placeholder="Enter student ID">

</div>

<div class="form-group">

<label for="name">Full Name:</label>

<input type="text" id="name" th:field="\*{name}" required placeholder="Enter full name">

</div>

<div class="form-group">

<label for="email">Email Address:</label>

<input type="email" id="email" th:field="\*{email}" required placeholder="Enter email address">

</div>

<div class="form-group">

<label for="password">Password:</label>

<input type="password" id="password" th:field="\*{password}" required placeholder="Enter password">

</div>

<div style="text-align: center; margin-top: 30px;">

<button type="submit" class="btn">Update Student</button>

<a href="/admin/dashboard" class="cancel-btn">Cancel</a>

</div>

</form>

</div>

</div>

</body>

</html>

**Admin Dashboard**

**File:** src/main/resources/templates/admin/dashboard.html

<!DOCTYPE html>

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Admin Dashboard</title>

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

background-color: #f4f6fb;

color: #333;

padding: 20px;

}

.container {

max-width: 1200px;

margin: auto;

background: #fff;

padding: 30px;

border-radius: 12px;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.07);

}

.header {

display: flex;

justify-content: space-between;

align-items: center;

border-bottom: 2px solid #eee;

padding-bottom: 15px;

margin-bottom: 25px;

}

.header h1 {

font-size: 2rem;

color: #2c3e50;

}

.header p {

margin-top: 5px;

font-size: 1rem;

color: #666;

}

.logout-btn {

background-color: #e74c3c;

color: #fff;

padding: 8px 16px;

border-radius: 6px;

text-decoration: none;

font-weight: bold;

transition: background-color 0.3s;

}

.logout-btn:hover {

background-color: #c0392b;

}

.content {

padding: 30px;

}

.stats-grid {

display: flex;

gap: 20px;

margin-bottom: 30px;

}

.stat-card {

flex: 1;

background-color: #f1f5ff;

padding: 20px;

border-radius: 10px;

text-align: center;

box-shadow: 0 4px 12px rgba(0, 0, 0, 0.05);

}

.stat-number {

font-size: 2.5rem;

font-weight: bold;

color: #2e86de;

}

.stat-label {

font-size: 1.1rem;

margin-top: 8px;

color: #555;

}

.section-title {

color: #333;

margin-bottom: 20px;

font-size: 24px;

}

.table-container {

background: #f8f9fa;

border-radius: 10px;

overflow: hidden;

}

.no-students {

text-align: center;

padding: 40px;

color: #666;

font-size: 18px;

}

.students-table {

width: 100%;

border-collapse: collapse;

}

.students-table th {

background: #343a40;

color: white;

padding: 15px;

text-align: left;

font-weight: 600;

}

.students-table td {

padding: 15px;

border-bottom: 1px solid #dee2e6;

}

.students-table tr:hover {

background: #e9ecef;

}

.verification-status {

padding: 4px 12px;

border-radius: 15px;

font-size: 12px;

font-weight: 600;

}

.verified {

background: #d4edda;

color: #155724;

}

.unverified {

background: #f8d7da;

color: #721c24;

}

.action-buttons {

display: flex;

gap: 10px;

}

.edit-btn {

background: #007bff;

color: white;

padding: 6px 12px;

border-radius: 4px;

text-decoration: none;

font-size: 12px;

font-weight: 600;

}

.edit-btn:hover {

background: #0056b3;

text-decoration: none;

color: white;

}

.delete-btn {

background: #dc3545;

color: white;

padding: 6px 12px;

border: none;

border-radius: 4px;

font-size: 12px;

font-weight: 600;

cursor: pointer;

}

.delete-btn:hover {

background: #c82333;

}

</style>

</style>

</head>

<body>

<div class="container">

<div class="header">

<div class="header-left">

<h1>👨‍💼 Admin Dashboard</h1>

<p>Welcome, <span th:text="${admin.fullName}">Admin</span></p>

</div>

<a href="/admin/logout" class="logout-btn">Logout</a>

</div>

<div class="content">

<div class="stats-grid">

<div class="stat-card">

<div class="stat-number" th:text="${totalStudents}">0</div>

<div class="stat-label">Total Students</div>

</div>

<div class="stat-card">

<div class="stat-number">Active</div>

<div class="stat-label">System Status</div>

</div>

</div>

<h2 class="section-title">📚 Student Management</h2>

<div class="table-container">

<div th:if="${#lists.isEmpty(students)}" class="no-students">

<p>No students registered yet.</p>

</div>

<table th:if="${!#lists.isEmpty(students)}" class="students-table">

<thead>

<tr>

<th>ID</th>

<th>Student ID</th>

<th>Name</th>

<th>Email</th>

<th>Status</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

<tr th:each="student : ${students}">

<td th:text="${student.id}">1</td>

<td th:text="${student.studentId}">STU001</td>

<td th:text="${student.name}">Student Name</td>

<td th:text="${student.email}">student@email.com</td>

<td>

<span th:if="${student.verified}" class="verification-status verified">Verified</span>

<span th:unless="${student.verified}" class="verification-status unverified">Unverified</span>

</td>

<td>

<div class="action-buttons">

<a th:href="@{/admin/edit-student/{id}(id=${student.id})}" class="edit-btn" >

Edit

</a>

<form th:action="@{/admin/delete-student/{id}(id=${student.id})}" method="post" style="display: inline;">

<button type="submit" class="delete-btn"

onclick="return confirm('Are you sure you want to delete this student?')">

Delete

</button>

</form>

</div>

</td>

</tr>

</tbody>

</table>

</div>

</div>

</div>

</body>

</html>

**Application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/student\_portal\_db?createDatabaseIfNotExist=true&useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

spring.jpa.properties.hibernate.format\_sql=true

spring.mail.host=smtp.gmail.com

spring.mail.port=587

spring.mail.username=testmail.project2004@gmail.com

spring.mail.password=cbll mmkm cicm mxbh

spring.mail.properties.mail.smtp.auth=true

spring.mail.properties.mail.smtp.starttls.enable=true

spring.mail.properties.mail.smtp.starttls.required=true

spring.mail.properties.mail.smtp.ssl.trust=smtp.gmail.com

spring.mail.properties.mail.smtp.connectiontimeout=5000

spring.mail.properties.mail.smtp.timeout=5000

spring.mail.properties.mail.smtp.writetimeout=5000

spring.mail.properties.mail.debug=true

server.port=8082

spring.thymeleaf.mode=HTML

spring.thymeleaf.cache=false

logging.level.com.example.demo=DEBUG

logging.level.org.springframework.mail=DEBUG

logging.level.org.springframework.web=DEBUG