**// VaccinationCenterApplication.java**

package com.vaccination;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class VaccinationCenterApplication {

public static void main(String[] args) {

SpringApplication.run(VaccinationCenterApplication.class, args);

}

}

**// CitizenController.java**

package com.vaccination.controller;

import java.util.List;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

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

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

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

import com.vaccination.entity.Citizen;

import com.vaccination.entity.VaccineCenter;

import com.vaccination.service.CitizenService;

import com.vaccination.service.VaccineCenterService;

@Controller

@RequestMapping("/citizens")

public class CitizenController {

private final CitizenService citizenService;

private final VaccineCenterService vaccineCenterService;

@Autowired

public CitizenController(CitizenService citizenService, VaccineCenterService vaccineCenterService) {

this.citizenService = citizenService;

this.vaccineCenterService = vaccineCenterService;

}

@GetMapping

public String getAllCitizens(Model model) {

List<Citizen> citizens = citizenService.getAllCitizens();

model.addAttribute("citizens", citizens);

return "citizen-details";

}

@GetMapping("/{id}")

public String viewCitizen(@PathVariable("id") Long id, Model model) {

Citizen citizen = citizenService.getCitizenById(id);

if (citizen != null) {

model.addAttribute("citizen", citizen);

return "view-citizen";

} else {

// Citizen not found, handle the error scenario

return "redirect:/citizens";

}

}

@GetMapping("/add")

public String addCitizen(Model model) {

List<VaccineCenter> vaccineCenters = vaccineCenterService.getAllVaccineCenters();

model.addAttribute("vaccineCenters", vaccineCenters);

model.addAttribute("citizen", new Citizen());

return "add-citizen";

}

@PostMapping("/add")

public String processAddCitizenForm(@ModelAttribute("citizen") Citizen citizen,

@RequestParam("vaccineCenterId") Long vaccineCenterId) {

VaccineCenter vaccineCenter = vaccineCenterService.getVaccineCenterById(vaccineCenterId);

citizen.setVaccinationCenter(vaccineCenter);

citizenService.addCitizen(citizen);

return "redirect:/citizens";

}

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

public String showEditCitizenForm(@PathVariable("id") Long id, Model model) {

Citizen citizen = citizenService.getCitizenById(id);

if (citizen != null) {

List<VaccineCenter> vaccineCenters = vaccineCenterService.getAllVaccineCenters();

model.addAttribute("citizen", citizen);

model.addAttribute("vaccineCenters", vaccineCenters);

return "edit-citizen";

} else {

// Citizen not found, handle the error scenario

return "redirect:/citizens";

}

}

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

public String updateCitizen(@PathVariable("id") Long id, @ModelAttribute("citizen") Citizen updatedCitizen,

@RequestParam("vaccineCenterId") Long vaccineCenterId) {

VaccineCenter vaccineCenter = vaccineCenterService.getVaccineCenterById(vaccineCenterId);

Citizen citizen = citizenService.getCitizenById(id);

if (citizen != null && vaccineCenter != null) {

citizen.setName(updatedCitizen.getName());

citizen.setCity(updatedCitizen.getCity());

citizen.setNoOfDoses(updatedCitizen.getNoOfDoses());

citizen.setVaccinationStatus(updatedCitizen.getVaccinationStatus());

citizen.setVaccinationCenter(vaccineCenter);

// Update other fields as needed

citizenService.updateCitizen(citizen);

}

return "redirect:/citizens";

}

@GetMapping("/{id}/delete")

public String deleteCitizen(@PathVariable("id") Long id) {

citizenService.deleteCitizen(id);

return "redirect:/citizens";

}

}

**// UserController.java**

package com.vaccination.controller;

import org.springframework.http.ResponseEntity;

import org.springframework.stereotype.Controller;

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

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

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

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

import com.vaccination.entity.User;

import com.vaccination.repository.UserRepository;

import com.vaccination.service.UserService;

@Controller

@RequestMapping("/")

public class UserController {

private final UserService userService;

private final UserRepository userRepository;

public UserController(UserService userService, UserRepository userRepository) {

this.userService = userService;

this.userRepository = userRepository;

}

@GetMapping

public String showRLoginPage() {

return "redirect:/login.html";

}

@PostMapping("/login")

public String processLogin(@RequestParam("username") String username, @RequestParam("password") String password) {

User user = userRepository.findByUsername(username);

// Check if the user exists and the password matches

if (user != null && user.getPassword().equals(password)) {

// Redirect to the user dashboard or any other desired page

return "redirect:/vaccine-centers";

} else {

return "redirect:/errorpage.html";

}

}

@GetMapping("/register")

public String showRegisterPage() {

return "redirect:/register.html";

}

@PostMapping("/register")

public ResponseEntity<String> registerUser(@RequestParam("username") String username,

@RequestParam("email") String email, @RequestParam("password") String password) {

User user = new User(username, email, password);

userService.registerUser(user);

return ResponseEntity.ok("User registered successfully");

}

}

**// VaccineCenterController.java**

package com.vaccination.controller;

import java.util.List;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

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

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

import com.vaccination.entity.Citizen;

import com.vaccination.entity.VaccineCenter;

import com.vaccination.service.CitizenService;

import com.vaccination.service.VaccineCenterService;

@Controller

@RequestMapping("/vaccine-centers")

public class VaccineCenterController {

private final VaccineCenterService vaccineCenterService;

private final CitizenService citizenService;

@Autowired

public VaccineCenterController(VaccineCenterService vaccineCenterService, CitizenService citizenService) {

this.vaccineCenterService = vaccineCenterService;

this.citizenService = citizenService;

}

@GetMapping

public String getAllVaccineCenters(Model model) {

List<VaccineCenter> vaccineCenters = vaccineCenterService.getAllVaccineCenters();

model.addAttribute("vaccineCenters", vaccineCenters);

return "vaccine-center-details";

}

@GetMapping("/add")

public String showAddVaccineCenterForm(Model model) {

model.addAttribute("vaccineCenter", new VaccineCenter());

return "add-vaccine-center";

}

@PostMapping("/add")

public String addVaccineCenter(@ModelAttribute("vaccineCenter") VaccineCenter vaccineCenter) {

vaccineCenterService.addVaccineCenter(vaccineCenter);

return "redirect:/vaccine-centers";

}

@GetMapping("/{id}")

public String viewVaccineCenter(@PathVariable("id") Long id, Model model) {

VaccineCenter vaccineCenter = vaccineCenterService.getVaccineCenterById(id);

List<Citizen> citizens = citizenService.getCitizensByVaccinationCenter(vaccineCenter);

model.addAttribute("vaccineCenter", vaccineCenter);

model.addAttribute("citizens", citizens);

return "vaccine-center-view";

}

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

public String showEditVaccineCenterForm(@PathVariable("id") Long id, Model model) {

VaccineCenter vaccineCenter = vaccineCenterService.getVaccineCenterById(id);

if (vaccineCenter != null) {

model.addAttribute("vaccineCenter", vaccineCenter);

return "edit-vaccine-center";

} else {

// Vaccine center not found, handle the error scenario

return "redirect:/vaccine-centers";

}

}

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

public String updateVaccineCenter(@PathVariable("id") Long id,

@ModelAttribute("vaccineCenter") VaccineCenter updatedVaccineCenter) {

VaccineCenter vaccineCenter = vaccineCenterService.getVaccineCenterById(id);

if (vaccineCenter != null) {

vaccineCenter.setCenter(updatedVaccineCenter.getCenter());

vaccineCenter.setCity(updatedVaccineCenter.getCity());

// Update other fields as needed

vaccineCenterService.updateVaccineCenter(vaccineCenter);

}

return "redirect:/vaccine-centers";

}

@GetMapping("/{id}/delete")

public String deleteVaccineCenter(@PathVariable("id") Long id) {

vaccineCenterService.deleteVaccineCenter(id);

return "redirect:/vaccine-centers";

}

}

**// Citizen.java**

package com.vaccination.entity;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.FetchType;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.JoinColumn;

import jakarta.persistence.ManyToOne;

import jakarta.persistence.Table;

@Entity

@Table(name = "citizens")

public class Citizen {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "name")

private String name;

@Column(name = "city")

private String city;

@Column(name = "no\_of\_doses")

private int noOfDoses;

@Column(name = "vaccination\_status")

private String vaccinationStatus;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "vaccination\_center")

private VaccineCenter vaccinationCenter;

// Constructors, getters, setters, and other properties

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

public int getNoOfDoses() {

return noOfDoses;

}

public void setNoOfDoses(int noOfDoses) {

this.noOfDoses = noOfDoses;

}

public String getVaccinationStatus() {

return vaccinationStatus;

}

public void setVaccinationStatus(String vaccinationStatus) {

this.vaccinationStatus = vaccinationStatus;

}

public VaccineCenter getVaccinationCenter() {

return vaccinationCenter;

}

public void setVaccinationCenter(VaccineCenter vaccinationCenter) {

this.vaccinationCenter = vaccinationCenter;

}

}

**// User.java**

package com.vaccination.entity;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "users")

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(nullable = false, unique = true)

private String username;

@Column(nullable = false, unique = true)

private String email;

@Column(nullable = false)

private String password;

public User() {

}

public User(String username, String email, String password) {

this.username = username;

this.password = password;

this.email = email;

}

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 getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

**// VaccineCenter.java**

package com.vaccination.entity;

import jakarta.persistence.Column;

import jakarta.persistence.Entity;

import jakarta.persistence.GeneratedValue;

import jakarta.persistence.GenerationType;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "vaccine\_centers")

public class VaccineCenter {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "center")

private String center;

@Column(name = "city")

private String city;

public VaccineCenter() {

// Default constructor required by JPA

}

public VaccineCenter(String center, String city) {

this.center = center;

this.city = city;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getCenter() {

return center;

}

public void setCenter(String center) {

this.center = center;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

}

**// CitizenRepository.java**

package com.vaccination.repository;

import java.util.List;

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

import org.springframework.stereotype.Repository;

import com.vaccination.entity.Citizen;

import com.vaccination.entity.VaccineCenter;

@Repository

public interface CitizenRepository extends JpaRepository<Citizen, Long> {

// Add custom query methods if needed

List<Citizen> findByVaccinationCenter(VaccineCenter vaccinationCenter);

}

**// UserRepository.java**

package com.vaccination.repository;

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

import org.springframework.stereotype.Repository;

import com.vaccination.entity.User;

@Repository

public interface UserRepository extends JpaRepository<User, Long> {

User findByUsername(String username);

}

**// VaccineCenterRepository.java**

package com.vaccination.repository;

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

import org.springframework.stereotype.Repository;

import com.vaccination.entity.VaccineCenter;

@Repository

public interface VaccineCenterRepository extends JpaRepository<VaccineCenter, Long> {

// Add custom query methods if needed

}

**// CitizenService.java**

package com.vaccination.service;

import java.util.List;

import com.vaccination.entity.Citizen;

import com.vaccination.entity.VaccineCenter;

public interface CitizenService {

void addCitizen(Citizen citizen);

void updateCitizen(Citizen citizen);

void deleteCitizen(Long id);

Citizen getCitizenById(Long id);

List<Citizen> getAllCitizens();

List<Citizen> getCitizensByVaccinationCenter(VaccineCenter vaccineCenter);

}

**// CitizenServiceImpl.java**

package com.vaccination.service;

import java.util.List;

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

import org.springframework.stereotype.Service;

import com.vaccination.entity.Citizen;

import com.vaccination.entity.VaccineCenter;

import com.vaccination.repository.CitizenRepository;

@Service

public class CitizenServiceImpl implements CitizenService {

@Autowired

private CitizenRepository citizenRepository;

@Override

public void addCitizen(Citizen citizen) {

citizenRepository.save(citizen);

}

@Override

public void updateCitizen(Citizen citizen) {

citizenRepository.save(citizen);

}

@Override

public void deleteCitizen(Long id) {

citizenRepository.deleteById(id);

}

@Override

public Citizen getCitizenById(Long id) {

return citizenRepository.findById(id).orElseThrow(() -> new NotFoundException("Citizen not found"));

}

@Override

public List<Citizen> getAllCitizens() {

return citizenRepository.findAll();

}

@Override

public List<Citizen> getCitizensByVaccinationCenter(VaccineCenter vaccineCenter) {

return citizenRepository.findByVaccinationCenter(vaccineCenter);

}

public class NotFoundException extends RuntimeException {

public NotFoundException(String message) {

super(message);

}

}

}

**// UserService.java**

package com.vaccination.service;

import org.springframework.stereotype.Service;

import com.vaccination.entity.User;

import com.vaccination.repository.UserRepository;

@Service

public class UserService {

private final UserRepository userRepository;

public UserService(UserRepository userRepository) {

this.userRepository = userRepository;

}

public void registerUser(User user) {

userRepository.save(user);

}

}

**// VaccineCenterService.java**

package com.vaccination.service;

import java.util.List;

import com.vaccination.entity.VaccineCenter;

public interface VaccineCenterService {

VaccineCenter addVaccineCenter(VaccineCenter vaccineCenter);

void updateVaccineCenter(VaccineCenter updatedVaccineCenter);

void deleteVaccineCenter(Long id);

VaccineCenter getVaccineCenterById(Long id);

List<VaccineCenter> getAllVaccineCenters();

}

**// VaccineCenterServiceImpl.java**

package com.vaccination.service;

import java.util.List;

import org.springframework.stereotype.Service;

import com.vaccination.entity.VaccineCenter;

import com.vaccination.repository.VaccineCenterRepository;

@Service

public class VaccineCenterServiceImpl implements VaccineCenterService {

private final VaccineCenterRepository vaccineCenterRepository;

public VaccineCenterServiceImpl(VaccineCenterRepository vaccineCenterRepository) {

this.vaccineCenterRepository = vaccineCenterRepository;

}

@Override

public VaccineCenter addVaccineCenter(VaccineCenter vaccineCenter) {

return vaccineCenterRepository.save(vaccineCenter);

}

@Override

public void updateVaccineCenter(VaccineCenter updatedVaccineCenter) {

VaccineCenter existingCenter = vaccineCenterRepository.findById(updatedVaccineCenter.getId())

.orElseThrow(() -> new NotFoundException("Vaccine center not found"));

existingCenter.setCenter(updatedVaccineCenter.getCenter());

existingCenter.setCity(updatedVaccineCenter.getCity());

// Update other fields as needed

vaccineCenterRepository.save(existingCenter);

}

@Override

public void deleteVaccineCenter(Long id) {

VaccineCenter existingCenter = vaccineCenterRepository.findById(id)

.orElseThrow(() -> new NotFoundException("Vaccine center not found"));

vaccineCenterRepository.delete(existingCenter);

}

@Override

public VaccineCenter getVaccineCenterById(Long id) {

return vaccineCenterRepository.findById(id)

.orElseThrow(() -> new NotFoundException("Vaccine center not found"));

}

@Override

public List<VaccineCenter> getAllVaccineCenters() {

return vaccineCenterRepository.findAll();

}

public class NotFoundException extends RuntimeException {

public NotFoundException(String message) {

super(message);

}

}

}

**//errorpage.html**

<!DOCTYPE html>

<html>

<head>

<title>Error - Vaccination Project</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container">

<h1 class="mt-4">Error</h1>

<div class="alert alert-danger" role="alert">

Invalid username or password. Please try again or register.

</div>

<a href="/register" class="btn btn-primary">Register</a>

<a href="/" class="btn btn-secondary">Try Again</a>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>

**//login.html**

<!DOCTYPE html>

<html>

<head>

<title>Login - Vaccination Project</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container">

<h1 class="mt-4">Login</h1>

<form action="/login" method="POST">

<div class="form-group">

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

<input type="text" class="form-control" id="username" name="username" required>

</div>

<div class="form-group">

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

<input type="password" class="form-control" id="password" name="password" required>

</div>

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

</form>

<p class="mt-3">Don't have an account? <a href="/register">Register here</a>.</p>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@1.16.1/dist/umd/popper.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>

**//register.html**

<!DOCTYPE html>

<html>

<head>

<title>Register - Vaccination Project</title>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>

<div class="container">

<h1 class="mt-4">Register</h1>

<form action="/register" method="POST">

<div class="form-group">

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

<input type="text" class="form-control" id="username" name="username" required>

</div>

<div class="form-group">

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

<input type="email" class="form-control" id="email" name="email" required>

</div>

<div class="form-group">

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

<input type="password" class="form-control" id="password" name="password" required>

</div>

<button type="submit" class="btn btn-primary">Register</button>

</form>

<p class="mt-3">Already have an account? <a href="/">Login here</a>.</p>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@1.16.1/dist/umd/popper.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>

**//application.properties**

spring.datasource.url=jdbc:mysql://localhost:3306/vaccination?useSSL=false&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=Un$ammu#2982

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=update

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

logging.level.org.hibernate.SQL=DEBUG

**// VaccinationCenterApplicationTests.java**

package com.vaccination;

import org.junit.jupiter.api.Test;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class VaccinationCenterApplicationTests {

@Test

void contextLoads() {

}

}