phase3-End project

Vaccination Center

Source code

Citizen controller:

```
package com.ecommerce.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.ecommerce.entity.Citizen;
import com.ecommerce.entity.VaccineCenter;
import com.ecommerce.service.CitizenService;
import com.ecommerce.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";
	}
}
```

User controller:

```
package com.ecommerce.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.ecommerce.entity.User;
import com.ecommerce.repository.UserRepository;
import com.ecommerce.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);
   if (user != null && user.getPassword().equals(password)) {

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");
}
}
```

Vaccine center controller:

```
package com.ecommerce.controller;

import java.util.List;

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.ecommerce.entity.Citizen;
import com.ecommerce.entity.VaccineCenter;
import com.ecommerce.service.CitizenService;
import com.ecommerce.service.VaccineCenterService;

@Controller
@RequestMapping("/vaccine-centers")
public class VaccineCenterController {

private final VaccineCenterService vaccineCenterService;
private final CitizenService citizenService;

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 {

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());
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:

```
package com.ecommerce.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:

```
package com.ecommerce.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;
}
```

Vaccine center:

```
package com.ecommerce.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;
}
}
}
```

Citizen repository:

```
package com.ecommerce.repository;


import java.util.List;

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

import com.ecommerce.entity.Citizen;
import com.ecommerce.entity.VaccineCenter;

public interface CitizenRepository extends JpaRepository<Citizen, Long> {

List<Citizen> findByVaccinationCenter(VaccineCenter vaccinationCenter);
}
```

User repository:

```
package com.ecommerce.repository;
import org.springframework.data.jpa.repository.JpaRepository;

import com.ecommerce.entity.User;

public interface UserRepository extends JpaRepository<User, Long> {

User findByUsername(String username);

}
```

Vaccine repository:

```
package com.ecommerce.repository;

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

import com.ecommerce.entity.VaccineCenter;

public interface VaccineCenterRepository extends JpaRepository<VaccineCenter, Long> {
// Add custom query methods if needed
```

```
}
```

Citizen service:

```
package com.ecommerce.service;

import java.util.List;

import com.ecommerce.entity.Citizen;
import com.ecommerce.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);
}
```

Citizen imply:

```
package com.ecommerce.service;

import java.util.List;

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

import com.ecommerce.entity.Citizen;
import com.ecommerce.entity.VaccineCenter;
import com.ecommerce.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);
		}
	}
}
```

User service:

```
package com.ecommerce.service;

import org.springframework.stereotype.Service;

import com.ecommerce.entity.User;
import com.ecommerce.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);
}
}
```

Vaccine center service:

```
package com.ecommerce.service;

import java.util.List;

import com.ecommerce.entity.VaccineCenter;

public interface VaccineCenterService {

VaccineCenter addVaccineCenter(VaccineCenter vaccineCenter);

void updateVaccineCenter(VaccineCenter updatedVaccineCenter);

void deleteVaccineCenter(Long id);

VaccineCenter getVaccineCenterById(Long id);

List<VaccineCenter> getAllVaccineCenters();
}
```

Vaccine center service imply:

```
package com.ecommerce.service;

import java.util.List;

import org.springframework.stereotype.Service;

import com.ecommerce.entity.VaccineCenter;
import com.ecommerce.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());

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);
}
}
}
```

Vaccine application:

```
package com.ecommerce;

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

@SpringBootApplication
```

```
public class VaccinationCentreApplication {

	public static void main(String[] args) {
		SpringApplication.run(VaccinationCentreApplication.class, args);
	}

}
```

Vaccine center appli test:

```
package com.ecommerce;

import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
class VaccinationCentreApplicationTests {

	@Test
	void contextLoads() {
	}

}
```

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>3.1.0</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
	<artifactId>Vaccination_Centre</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>Vaccination_Centre</name>
	<description>Demo project for Spring Boot</description>
	<properties>
		<java.version>17</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-jpa</artifactId>
```

```
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-thymeleaf</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</artifactId>
		</dependency>

		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-devtools</artifactId>
			<scope>runtime</scope>
			<optional>true</optional>
		</dependency>
		<dependency>
			<groupId>com.mysql</groupId>
			<artifactId>mysql-connector-j</artifactId>
			<scope>runtime</scope>
		</dependency>
		<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>
		</plugins>
	</build>

</project>
```