package iuh.fit.se.entities;

import jakarta.persistence.\*;

import jakarta.validation.constraints.\*;

import java.time.LocalDate;

import org.springframework.format.annotation.DateTimeFormat;

@Entity

@Table(name = "flights")

public class Flight {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int flightId;

@NotBlank(message = "Airplane name is required.")

@Pattern(regexp = "Airbus 0001|Airbus 0002|Boeing 0001|Boeing 0002", message = "Invalid airplane name.")

private String airplaneName;

@NotBlank(message = "Departure airport is required.")

private String departureAirport;

@NotBlank(message = "Arrival airport is required.")

private String arrivalAirport;

@FutureOrPresent(message = "Departure date must be in the present or future.")

@DateTimeFormat(pattern = "yyyy-MM-dd")

private LocalDate departureDate;

@Future(message = "Arrival date must be in the future.")

@DateTimeFormat(pattern = "yyyy-MM-dd")

private LocalDate arrivalDate;

@Min(value = 1, message = "Seat fare must be greater than 0.")

private double seatFare;

@Min(value = 0)

@Max(value = 1)

private int flightStatus;

}

-- **repository**

@Repository

public interface FlightRepository extends JpaRepository<Flight, Integer> {

boolean existsByAirplaneNameAndDepartureDateBetween(String airplaneName, LocalDate startDate, LocalDate endDate);

}

-- **services**

public interface FlightService {

List<Flight> getAllFlights();

Flight getFlightById(int id);

void addFlight(Flight flight);

void deleteFlight(int id);

void updateFlight(Flight flight);

}

-- **servicesIMPL**

package iuh.fit.se.servicesIml;

import java.util.List;

import org.springframework.stereotype.Service;

import iuh.fit.se.entities.Flight;

import iuh.fit.se.repository.FlightRepository;

import iuh.fit.se.services.FlightService;

@Service

public class FlightServiceImpl implements FlightService {

private final FlightRepository flightRepository;

// Constructor Injection

public FlightServiceImpl(FlightRepository flightRepository) {

this.flightRepository = flightRepository;

}

@Override

public List<Flight> getAllFlights() {

return flightRepository.findAll();

}

@Override

public Flight getFlightById(int id) {

return flightRepository.findById(id).orElse(null); // Xử lý Optional

}

// @Override

// public void addFlight(Flight flight) {

// flightRepository.save(flight);

// }

@Override

public void deleteFlight(int id) {

flightRepository.deleteById(id); // Xóa bằng deleteById

}

@Override

public void addFlight(Flight flight) {

if (flight.getDepartureDate().isAfter(flight.getArrivalDate())) {

throw new IllegalArgumentException("Departure date must be before arrival date.");

}

boolean exists = flightRepository.existsByAirplaneNameAndDepartureDateBetween(flight.getAirplaneName(),

flight.getDepartureDate(), flight.getArrivalDate());

if (exists) {

throw new IllegalArgumentException("A flight with the same airplane and overlapping dates already exists.");

}

flightRepository.save(flight);

}

@Override

public void updateFlight(Flight flight) {

Flight existingFlight = flightRepository.findById(flight.getFlightId()).orElse(null);

if (existingFlight == null) {

throw new IllegalArgumentException("Flight not found for update.");

}

if (flight.getDepartureDate().isAfter(flight.getArrivalDate())) {

throw new IllegalArgumentException("Departure date must be before arrival date.");

}

// Cập nhật thông tin chuyến bay

existingFlight.setAirplaneName(flight.getAirplaneName());

existingFlight.setDepartureAirport(flight.getDepartureAirport());

existingFlight.setArrivalAirport(flight.getArrivalAirport());

existingFlight.setDepartureDate(flight.getDepartureDate());

existingFlight.setArrivalDate(flight.getArrivalDate());

existingFlight.setSeatFare(flight.getSeatFare());

existingFlight.setFlightStatus(flight.getFlightStatus());

flightRepository.save(existingFlight); // Lưu thay đổi vào DB

}

}

**--Controller**

package iuh.fit.se.controller;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

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

import iuh.fit.se.entities.Flight;

import iuh.fit.se.services.FlightService;

import jakarta.validation.Valid;

@Controller

@RequestMapping("/flights")

public class FlightController {

@Autowired

private FlightService flightService;

@GetMapping

public String listFlights(Model model) {

model.addAttribute("flights", flightService.getAllFlights());

return "list\_flights";

}

@GetMapping("/add")

public String showAddFlightForm(Model model) {

model.addAttribute("flight", new Flight());

return "add\_flight";

}

@PostMapping("/add")

public String addFlight(@Valid @ModelAttribute("flight") Flight flight, BindingResult result, Model model) {

if (result.hasErrors()) {

return "add\_flight"; // Trả về trang thêm nếu có lỗi

}

try {

flightService.addFlight(flight);

} catch (IllegalArgumentException e) {

model.addAttribute("errorMessage", e.getMessage());

return "add\_flight"; // Trả về trang thêm với lỗi logicc

}

return "redirect:/flights"; // Chuyển về danh sách sau khi thành công

}

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

public String deleteFlight(@PathVariable int id, Model model) {

Flight flight = flightService.getFlightById(id);

if (flight == null) {

model.addAttribute("errorMessage", "Flight not found.");

return "redirect:/flights";

}

if (flight.getFlightStatus() != 0) {

model.addAttribute("errorMessage", "Only flights with status 'NEW' can be deleted.");

return "redirect:/flights";

}

flightService.deleteFlight(id);

return "redirect:/flights";

}

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

public String showEditFlightForm(@PathVariable int id, Model model) {

Flight flight = flightService.getFlightById(id);

if (flight == null) {

model.addAttribute("errorMessage", "Flight not found");

return "redirect:/flights";

}

model.addAttribute("flight", flight);

return "edit\_flight";

}

@PostMapping("/edit")

public String editFlight(@ModelAttribute("flight") Flight flight, BindingResult result, Model model) {

if (result.hasErrors()) {

return "edit\_flight";

}

try {

flightService.addFlight(flight); // Gọi lại phương thức lưu hoặc sửa

} catch (IllegalArgumentException e) {

model.addAttribute("errorMessage", e.getMessage());

return "edit\_flight";

}

return "redirect:/flights"; // Chuyển về danh sách sau khi sửa xong

}

}

--**html**

<!DOCTYPE html>

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

<head>

<title>Add Flight</title>

<style type="text/css">

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color: #f9f9f9;

}

h1 {

text-align: center;

color: #333;

margin-top: 20px;

}

form {

width: 400px;

margin: 50px auto;

padding: 20px;

background-color: #fff;

border: 1px solid #ddd;

border-radius: 5px;

}

label {

display: block;

margin-bottom: 5px;

font-size: 14px;

color: #555;

}

input, select {

width: calc(100% - 20px);

padding: 8px;

margin-bottom: 15px;

font-size: 14px;

border: 1px solid #ccc;

border-radius: 3px;

box-sizing: border-box;

}

button, a {

display: inline-block;

padding: 8px 15px;

font-size: 14px;

border: none;

border-radius: 3px;

text-decoration: none;

text-align: center;

cursor: pointer;

margin-top: 10px;

}

button {

background-color: #007bff;

color: #fff;

}

button:hover {

background-color: #0056b3;

}

a {

background-color: #f1f1f1;

color: #333;

}

a:hover {

background-color: #ddd;

}

</style>

</head>

<body>

<h1>Add Flight</h1>

<form action="#" th:action="@{/flights/add}" th:object="${flight}" method="post">

<div th:if="${errorMessage}" style="color:red;">

<p th:text="${errorMessage}"></p>

</div>

<div th:if="${#fields.hasErrors('airplaneName')}" style="color:red;">

<p th:text="${#fields.errors('airplaneName')}"></p>

</div>

<label>Airplane Name:</label>

<select th:field="\*{airplaneName}" required>

<option value="" disabled selected>Select Airplane</option>

<option value="Airbus 0001">Airbus 0001</option>

<option value="Airbus 0002">Airbus 0002</option>

<option value="Boeing 0001">Boeing 0001</option>

<option value="Boeing 0002">Boeing 0002</option>

</select><br>

<div th:if="${#fields.hasErrors('departureAirport')}" style="color:red;">

<p th:text="${#fields.errors('departureAirport')}"></p>

</div>

<label>Departure Airport:</label>

<input type="text" th:field="\*{departureAirport}" required><br>

<div th:if="${#fields.hasErrors('arrivalAirport')}" style="color:red;">

<p th:text="${#fields.errors('arrivalAirport')}"></p>

</div>

<label>Arrival Airport:</label>

<input type="text" th:field="\*{arrivalAirport}" required><br>

<div th:if="${#fields.hasErrors('departureDate')}" style="color:red;">

<p th:text="${#fields.errors('departureDate')}"></p>

</div>

<label>Departure Date:</label>

<input type="date" th:field="\*{departureDate}" required><br>

<div th:if="${#fields.hasErrors('arrivalDate')}" style="color:red;">

<p th:text="${#fields.errors('arrivalDate')}"></p>

</div>

<label>Arrival Date:</label>

<input type="date" th:field="\*{arrivalDate}" required><br>

<div th:if="${#fields.hasErrors('seatFare')}" style="color:red;">

<p th:text="${#fields.errors('seatFare')}"></p>

</div>

<label>Seat Fare:</label>

<input type="number" th:field="\*{seatFare}" step="0.01" required><br>

<label>Flight Image:</label>

<input type="file" name="flightImage" accept="image/\*" required><br>

<label>Flight Status:</label>

<select th:field="\*{flightStatus}">

<option value="0">New</option>

<option value="1">Scheduled</option>

</select><br>

<a href="/flights">Back</a>

<button type="submit">Register</button>

</form>

</body>

</html>

**<!DOCTYPE html>**

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

<head>

<title>Airline Reservation System</title>

</head>

<body>

<h1>STT23\_NGUYEN DUC NHAT\_21059221</h1>

<h2>Airline Reservation System</h2>

<table border="1">

<thead>

<tr>

<th>Airplane Name</th>

<th>Departure Airport</th>

<th>Arrival Airport</th>

<th>Departure Date</th>

<th>Arrival Date</th>

<th>Seat Fare</th>

<th>Flight Status</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

<tr th:each="flight : ${flights}">

<td th:text="${flight.airplaneName}"></td>

<td th:text="${flight.departureAirport}"></td>

<td th:text="${flight.arrivalAirport}"></td>

<td th:text="${flight.departureDate}"></td>

<td th:text="${flight.arrivalDate}"></td>

<td th:text="${flight.seatFare}"></td>

<td th:text="${flight.flightStatus == 0 ? 'New' : 'Scheduled'}"></td>

<td>

<a th:href="@{'/flights/edit/' + ${flight.flightId}}">Edit</a>

<a th:href="@{'/flights/delete/' + ${flight.flightId}}" th:if="${flight.flightStatus == 0}"

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

Delete

</a>

</td>

</tr>

</tbody>

<!-- Button add fligt, bottom in the form -->

<a id="buttonAdd" href="/flights/add">Add Flight</a>

</table>

</body>

</html>