package com.becoder.controller;  
  
import com.becoder.model.Document;  
import com.becoder.repository.DocumentRepository;  
import jakarta.servlet.http.HttpSession;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.core.io.ClassPathResource;  
import org.springframework.core.io.Resource;  
import org.springframework.core.io.UrlResource;  
import org.springframework.http.HttpHeaders;  
import org.springframework.http.MediaType;  
import org.springframework.http.ResponseEntity;  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.multipart.MultipartFile;  
  
import java.io.File;  
import java.net.MalformedURLException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.nio.file.StandardCopyOption;  
import java.util.List;  
  
@Controller // Обозначает класс как контроллер Spring  
public class DocumentController {  
  
 @Autowired // Автоматическая инъекция DocumentRepository  
 private DocumentRepository documentRepo;  
  
 @GetMapping("/") // GET-запрос к корневому URL-адресу  
 public String index(Model m) {  
 // ... (код для отображения документов, если требуется)  
  
 // Получить список документов из хранилища  
 List<Document> documents = documentRepo.findAll();  
  
// Перебрать список и добавить элементы в модель  
 for (Document doc : documents) {  
 m.addAttribute("document", doc);  
 }  
  
// Возвратить шаблон, который отобразит список документов  
 return "index";  
  
 }  
  
 // POST-запрос к URL-адресу /documentUpload  
 @PostMapping("/documentUpload")  
 public String documentUpload(  
 @RequestParam("doc") MultipartFile doc,  
 @RequestParam("author") String author,  
 @RequestParam("description") String description,  
 @RequestParam("genre") String genre,  
 @RequestParam("releaseYear") Integer releaseYear,  
 HttpSession session) {  
  
 Document document = new Document(); // Создание нового объекта Document  
 document.setDocumentName(doc.getOriginalFilename());  
 document.setAuthor(author);  
 document.setDescription(description);  
 document.setGenre(genre);  
 document.setReleaseYear(releaseYear);  
  
 Document savedDocument = documentRepo.save(document); // Сохранение Document в хранилище  
  
 if (savedDocument != null) { // Обработка загрузки файла  
 try {  
 File saveDir = new ClassPathResource("static/documents").getFile(); // Определение каталога сохранения  
 Path path = Paths.*get*(saveDir.getAbsolutePath() + File.*separator* + doc.getOriginalFilename()); // Путь для сохранения файла  
 Files.*copy*(doc.getInputStream(), path, StandardCopyOption.*REPLACE\_EXISTING*); // Копирование загруженного файла  
  
 // Обновление пути к файлу в базе данных  
 savedDocument.setPath(path.toString());  
 documentRepo.save(savedDocument);  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 // Обработайте исключение должным образом  
 session.setAttribute("msg", "Ошибка при загрузке документа");  
 return "redirect:/";  
 }  
 }  
  
 session.setAttribute("msg", "Документ загружен успешно");  
 return "redirect:/";  
 }  
  
  
  
  
 @GetMapping("/documents")  
 public String listFiles(Model model) {  
 List<Document> documents = documentRepo.findAll();  
 model.addAttribute("documents", documents); // Передаем список документов в модель  
 return "documents"; // Возвращаем имя шаблона страницы с документами  
 }  
  
  
  
 @GetMapping("/download/{id}")  
 public ResponseEntity<Resource> downloadFile(@PathVariable Integer id) throws MalformedURLException {  
 Document document = documentRepo.findById(id).orElseThrow(() -> new RuntimeException("File not found"));  
 Path path = Paths.*get*(document.getPath());  
 Resource resource = new UrlResource(path.toUri());  
  
 return ResponseEntity.*ok*()  
 .contentType(MediaType.*APPLICATION\_OCTET\_STREAM*)  
 .header(HttpHeaders.*CONTENT\_DISPOSITION*, "attachment; filename=\"" + resource.getFilename() + "\"")  
 .body(resource);  
 }  
  
  
}

package com.becoder.model;  
  
import jakarta.persistence.Entity;  
import jakarta.persistence.GeneratedValue;  
import jakarta.persistence.GenerationType;  
import jakarta.persistence.Id;  
  
@Entity  
public class Document {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private int id;  
 private String documentName;  
  
 private String path ;  
 private String author;  
 private String description;  
 private String genre;  
 private Integer releaseYear;  
  
 // ... getters and setters  
  
  
 public int getId() {  
 return id;  
 }  
  
 public void setId(int id) {  
 this.id = id;  
 }  
  
 public String getDocumentName() {  
 return documentName;  
 }  
  
 public String getPath() {  
 return path;  
 }  
  
 public void setPath(String path) {  
 this.path = path;  
 }  
  
 public void setDocumentName(String documentName) {  
 this.documentName = documentName;  
 }  
 public String getAuthor() {  
 return author;  
 }  
  
 public void setAuthor(String author) {  
 this.author = author;  
 }  
  
 public String getDescription() {  
 return description;  
 }  
  
 public void setDescription(String description) {  
 this.description = description;  
 }  
  
 public String getGenre() {  
 return genre;  
 }  
  
 public void setGenre(String genre) {  
 this.genre = genre;  
 }  
  
 public Integer getReleaseYear() {  
 return releaseYear;  
 }  
  
 public void setReleaseYear(Integer releaseYear) {  
 this.releaseYear = releaseYear;  
 }  
}

package com.becoder.repository;  
  
import com.becoder.model.Document;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface DocumentRepository extends JpaRepository<Document, Integer> {  
}

package com.becoder;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class DemoApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(DemoApplication.class, args);  
 }  
  
}

<!Doctype html>  
<html lang="ru" xmlns:th="http://www.thymeleaf.org">  
<head>  
 <meta charset="UTF-8">  
 <title>Сайт для загрузки документов</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"  
 rel="stylesheet"  
 integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"  
 crossorigin="anonymous">  
</head>  
<body>  
<nav class="navbar navbar-expand-lg navbar-dark bg-success">  
 <div class="container-fluid">  
 <a class="navbar-brand" href="#">Сайт для загрузки документов</a>  
 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">  
 <span class="navbar-toggler-icon"></span>  
 </button>  
 <div class="collapse navbar-collapse" id="navbarSupportedContent">  
 <div class="text-center">  
 <a href="/documents" class="btn btn-secondary">Список документов</a>  
 </div>  
 </div>  
 </div>  
</nav>  
  
<div class="container">  
 <th:block th:if="${session.msg}">  
 <p class="text-center fs-4 text-success">[[${session.msg}]]</p>  
 </th:block>  
  
 <div class="row">  
 <div class="col-md-6 offset-md-3 mt-4">  
 <form action="/documentUpload" enctype="multipart/form-data" method="post">  
 <div class="mb-3">  
 <label for="doc" class="form-label">Выберите документ:</label>  
 <input type="file" name="doc" id="doc" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="author" class="form-label">Автор:</label>  
 <input type="text" name="author" id="author" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="description" class="form-label">Описание:</label>  
 <textarea name="description" id="description" class="form-control" required></textarea>  
 </div>  
 <div class="mb-3">  
 <label for="genre" class="form-label">Жанр:</label>  
 <input type="text" name="genre" id="genre" class="form-control" required>  
 </div>  
 <div class="mb-3">  
 <label for="releaseYear" class="form-label">Год выпуска:</label>  
 <input type="number" name="releaseYear" id="releaseYear" class="form-control" required>  
 </div>  
 <div class="text-center">  
 <button type="submit" class="btn btn-primary">Загрузить</button>  
 </div>  
 </form>  
 </div>  
 </div>  
</div>  
  
<hr>  
  
<div class="container">  
 <div class="row">  
 </div>  
</div>  
  
</body>  
</html>

<!DOCTYPE html>  
<html lang="ru" xmlns:th="http://www.w3.org/1999/xhtml">  
<head>  
 <meta charset="UTF-8">  
 <title>Загруженные документы</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"  
 rel="stylesheet"  
 integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"  
 crossorigin="anonymous">  
</head>  
<body>  
<div class="container mt-4">  
 <h2 class="mb-3">Список документов</h2>  
 <div class="row">  
 <div class="col-md-12">  
 <ul class="list-unstyled">  
 <li th:each="document : ${documents}" class="mb-3">  
 <div class="card">  
 <div class="card-body">  
 <h5 class="card-title" th:text="${document.documentName}">Название документа</h5>  
 <h6 class="card-subtitle mb-2 text-muted" th:text="'Автор: ' + ${document.author}">Автор</h6>  
 <p class="card-text" th:text="'Описание: ' + ${document.description}">Описание</p>  
 <p class="card-text" th:text="'Жанр: ' + ${document.genre}">Жанр</p>  
 <p class="card-text" th:text="'Год выпуска: ' + ${document.releaseYear}">Год выпуска</p>  
 <a th:href="@{'/download/' + ${document.id}}" class="card-link">Скачать</a>  
 </div>  
 </div>  
 </li>  
 </ul>  
 </div>  
 </div>  
</div>  
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"  
 integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"  
 crossorigin="anonymous"></script>  
</body>  
</html>

spring.datasource.url=jdbc:mysql://localhost:3306/doc3  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.username=root  
spring.datasource.password=Elchi2001#  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect  
spring.jpa.hibernate.ddl-auto=update  
  
spring.servlet.multipart.max-file-size=10MB  
spring.servlet.multipart.max-request-size=10MB