import javax.persistence.\*;

import java.util.HashSet;

import java.util.Set;

@Entity

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String username;

@OneToMany(mappedBy = "user", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private Set<Image> images = new HashSet<>();

// getters and setters

}

import javax.persistence.\*;

@Entity

public class Image {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String fileName;

private String fileType;

private byte[] data;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "user\_id", nullable = false)

private User user;

// getters and setters

}

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

public interface UserRepository extends JpaRepository<User, Long> {

}

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

public interface ImageRepository extends JpaRepository<Image, Long> {

}

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

import org.springframework.stereotype.Service;

import org.springframework.web.multipart.MultipartFile;

import java.io.IOException;

import java.util.Optional;

@Service

public class ImageService {

@Autowired

private ImageRepository imageRepository;

@Autowired

private UserRepository userRepository;

public Image storeImage(MultipartFile file, Long userId) throws IOException {

User user = userRepository.findById(userId)

.orElseThrow(() -> new RuntimeException("User not found"));

Image image = new Image();

image.setFileName(file.getOriginalFilename());

image.setFileType(file.getContentType());

image.setData(file.getBytes());

image.setUser(user);

return imageRepository.save(image);

}

public Optional<Image> getImage(Long imageId) {

return imageRepository.findById(imageId);

}

}

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

import org.springframework.http.HttpHeaders;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

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

import org.springframework.web.multipart.MultipartFile;

import java.io.IOException;

import java.util.Optional;

@RestController

@RequestMapping("/api/images")

public class ImageController {

@Autowired

private ImageService imageService;

@PostMapping("/upload")

public ResponseEntity<String> uploadImage(@RequestParam("file") MultipartFile file, @RequestParam("userId") Long userId) {

try {

Image image = imageService.storeImage(file, userId);

return ResponseEntity.status(HttpStatus.OK)

.body("Image uploaded successfully: " + image.getFileName());

} catch (IOException e) {

return ResponseEntity.status(HttpStatus.INTERNAL\_SERVER\_ERROR)

.body("Could not upload the image: " + e.getMessage());

}

}

@GetMapping("/{imageId}")

public ResponseEntity<byte[]> getImage(@PathVariable Long imageId) {

Optional<Image> image = imageService.getImage(imageId);

if (image.isPresent()) {

return ResponseEntity.ok()

.header(HttpHeaders.CONTENT\_DISPOSITION, "attachment; filename=\"" + image.get().getFileName() + "\"")

.body(image.get().getData());

}

return ResponseEntity.status(HttpStatus.NOT\_FOUND).body(null);

}

}

spring.servlet.multipart.enabled=true

spring.servlet.multipart.max-file-size=2MB

spring.servlet.multipart.max-request-size=2MB