Image codes

package com.vdc.vignan.degree.college.Entity;

import jakarta.persistence.\*;

import java.util.Arrays;

@Entity

public class Image {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

@Lob

private byte[] data;

// Getters and setters

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 byte[] getData() { return data; }

public void setData(byte[] data) { this.data = data; }

@Override

public String toString() {

return "Image [id=" + id + ", name=" + name + ", data=" + Arrays.toString(data) + "]";

}

}

package com.vdc.vignan.degree.college.Repository;

import com.vdc.vignan.degree.college.Entity.Image;

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

public interface ImageRepository extends JpaRepository<Image, Long> {

}

package com.vdc.vignan.degree.college.Service;

import com.vdc.vignan.degree.college.Entity.Image;

import com.vdc.vignan.degree.college.Repository.ImageRepository;

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.\*;

@Service

public class ImageService {

@Autowired

private ImageRepository imageRepository;

public Map<String, Object> uploadImage(MultipartFile file) throws IOException {

Image image = new Image();

image.setName(file.getOriginalFilename());

image.setData(file.getBytes());

Image saved = imageRepository.save(image);

Map<String, Object> response = new HashMap<>();

response.put("id", saved.getId());

response.put("name", saved.getName());

response.put("message", "Image uploaded successfully");

return response;

}

public List<Map<String, Object>> getAllImages() {

List<Image> images = imageRepository.findAll();

List<Map<String, Object>> response = new ArrayList<>();

for (Image img : images) {

Map<String, Object> map = new HashMap<>();

map.put("id", img.getId());

map.put("name", img.getName());

map.put("data", Base64.getEncoder().encodeToString(img.getData()));

response.add(map);

}

return response;

}

public Map<String, Object> getImageById(Long id) throws NoSuchElementException {

Image image = imageRepository.findById(id).orElseThrow();

Map<String, Object> map = new HashMap<>();

map.put("id", image.getId());

map.put("name", image.getName());

map.put("data", Base64.getEncoder().encodeToString(image.getData()));

return map;

}

public void deleteImage(Long id) {

if (!imageRepository.existsById(id)) {

throw new NoSuchElementException("Image not found with id: " + id);

}

imageRepository.deleteById(id);

}

}

package com.vdc.vignan.degree.college.Controller;

import com.vdc.vignan.degree.college.Service.ImageService;

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

import org.springframework.http.\*;

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

import org.springframework.web.multipart.MultipartFile;

import java.util.\*;

@RestController

@RequestMapping("/api/vignan/images")

@CrossOrigin(origins = "\*")

public class ImageController {

@Autowired

private ImageService imageService;

@PostMapping("/upload")

public ResponseEntity<?> uploadImage(@RequestParam("file") MultipartFile file) {

try {

return ResponseEntity.ok(imageService.uploadImage(file));

} catch (Exception e) {

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

.body("Failed to upload image: " + e.getMessage());

}

}

@GetMapping

public ResponseEntity<?> getAllImages() {

try {

return ResponseEntity.ok(imageService.getAllImages());

} catch (Exception e) {

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

.body("Failed to fetch images: " + e.getMessage());

}

}

@GetMapping("/{id}")

public ResponseEntity<?> getImageById(@PathVariable Long id) {

try {

return ResponseEntity.ok(imageService.getImageById(id));

} catch (NoSuchElementException e) {

return ResponseEntity.status(HttpStatus.NOT\_FOUND).body("Image not found with id: " + id);

} catch (Exception e) {

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

.body("Failed to fetch image: " + e.getMessage());

}

}

@DeleteMapping("/{id}")

public ResponseEntity<?> deleteImage(@PathVariable Long id) {

try {

imageService.deleteImage(id);

return ResponseEntity.ok(Map.of("message", "Image deleted successfully"));

} catch (NoSuchElementException e) {

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

}

}

}