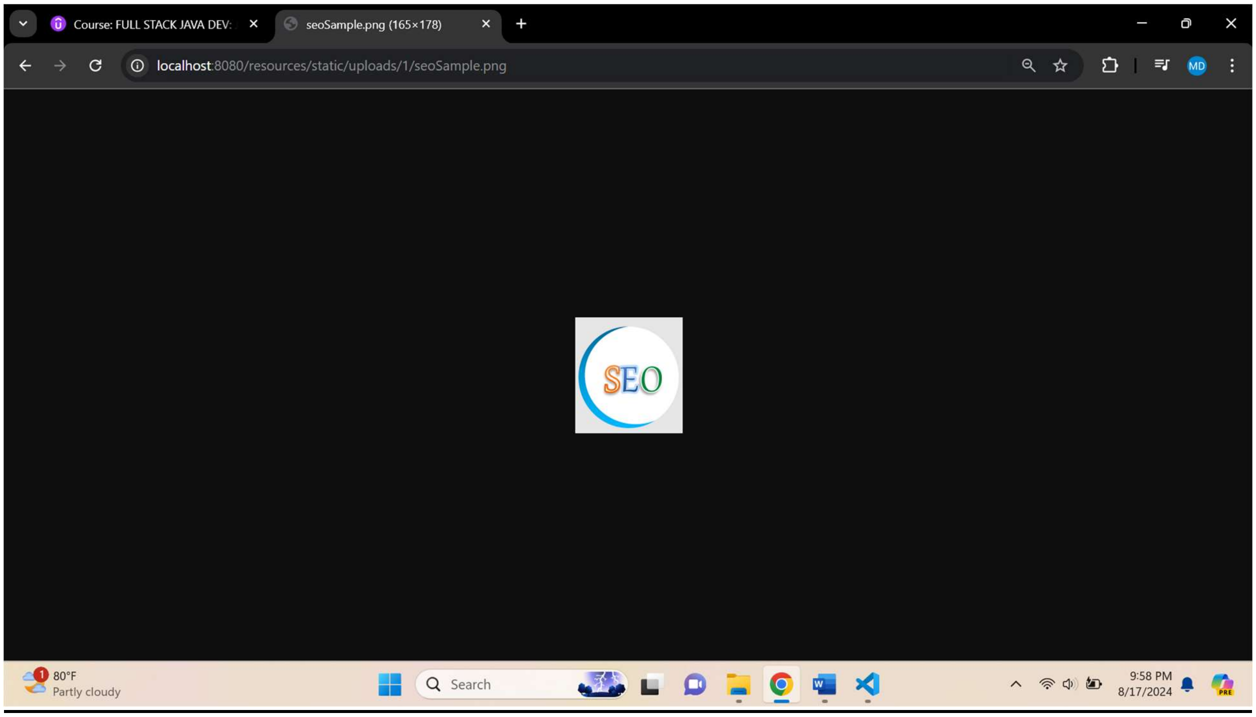
java_springboot_restful_29_AlbumController – Build Files Upload for Album

In application. properties; add below code

```
# File Settings

spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB
spring.mvc.static-path-pattern=/resources/static/**
```

Access to the static file

URL: http://localhost:8080/resources/static/uploads/1/seoSample.png


Course: FULL STACK JAVA DEV
seoSample.png (165×178)
localhost:8080/resources/static/uploads/1/seoSample.png
SEO
80°F
Partly cloudy
Search
9:58 PM
8/17/2024

Create a model -> Photo.java -> in folder -> models

```
package org.studyeasy.SpringRestdemo.model;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.JoinColumn;
import jakarta.persistence.ManyToOne;
```

```java
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;
import lombok.ToString;

@Entity
@Getter
@Setter
@ToString
@AllArgsConstructor
@NoArgsConstructor
public class Photo {

    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private long id;

    private String name;

    private String description;

    private String originalFileName;

    private String fileName;

    @ManyToOne
    @JoinColumn(name = "album_id", referencedColumnName = "id", nullable = false)
    private Album album;
}
```

Create a PhotoRepository.java in repository folder

```java
package org.studyeasy.SpringRestdemo.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import org.studyeasy.SpringRestdemo.model.Photo;

@Repository
public interface PhotoRepository extends JpaRepository<Photo, Long>{

}
```

Create a PhotoService.java in service folder

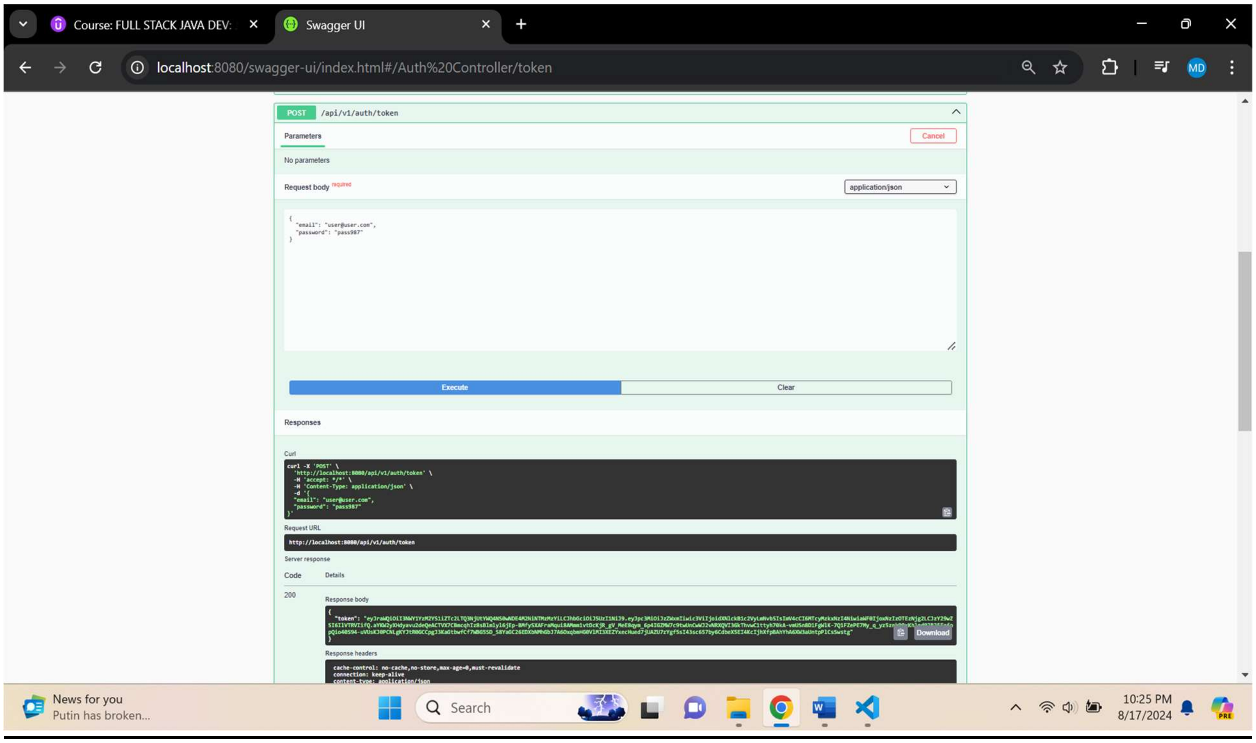
```
package org.studyeasy.SpringRestdemo.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.studyeasy.SpringRestdemo.model.Photo;
import org.studyeasy.SpringRestdemo.repository.PhotoRepository;

@Service
public class PhotoService {

    @Autowired
    private PhotoRepository photoRepository;

    public Photo save(Photo photo){
        return photoRepository.save(photo);
    }
}
```

Updated photos method in AlbumController

```
@PostMapping(value = "{album_id}/upload_photos", consumes = { "multipart/form-
data" })
    @Operation(summary = "Upload Photo into album")
    @SecurityRequirement(name = "studyeasy-demo-api")
    @ApiResponse(responseCode = "400", description = "Please check the payload of
token")
    public ResponseEntity<List<String>> photos(@RequestPart(required = true)
MultipartFile[] files,
            @PathVariable long album_id, Authentication authentication) {
        String email = authentication.getName();
        Optional<Account> optionalAccount = accountService.findByEmail(email);
        Account account = optionalAccount.get();
        Optional<Album> optionalAlbum = albumService.findById(album_id);
        Album album;
        if (optionalAlbum.isPresent()) {
            album = optionalAlbum.get();
            if (account.getId() != album.getAccount().getId()) {
                return ResponseEntity.status(HttpStatus.BAD_GATEWAY).body(null);
            }
        } else {
            return ResponseEntity.status(HttpStatus.BAD_GATEWAY).body(null);
    }

    List<String> fileNamesWithSuccess = new ArrayList<>();
    List<String> fileNamesWithError = new ArrayList<>();
    Arrays.asList(files).stream().forEach(file -> {
      // Checking the type of the file is correct or not
      String contentType = file.getContentType();

      if (contentType.equals("image/png")
          || contentType.equals("image/jpg")
          || contentType.equals("image/jpeg")) {
        fileNamesWithSuccess.add(file.getOriginalFilename());
  // When we are storing the file in the database, there is a possibility that
  // file name from the user
  // is repeating and if that happens, then file from the user will get
repeated and in server will get replaced.
  // In order to prevent that, we need to create a random string.;
        int length = 10;
        boolean useLetters = true;
        boolean useNumbers = true;
        try {
          String fileName = file.getOriginalFilename();
          String generatedString = RandomStringUtils.random(length,
useLetters, useNumbers);
          String final_photo_name = generatedString + fileName;
          String absolute_fileLocation =
AppUtil.get_photo_upload_path(final_photo_name, album_id);
          Path path = Paths.get(absolute_fileLocation);
          Files.copy(file.getInputStream(), path,
StandardCopyOption.REPLACE_EXISTING);
          Photo photo = new Photo();
          photo.setName(fileName);
          photo.setFileName(final_photo_name);
          photo.setOriginalFileName(fileName);
          photo.setAlbum(album);
          photoService.save(photo);
        } catch (Exception e) {

        }
      }else{
        fileNamesWithError.add(file.getOriginalFilename());
      }
    });
    return ResponseEntity.ok(fileNamesWithSuccess);
}
```

Output

Generate a token for the user and authorize



```
{
  "email": "user@user.com",
  "password": "pass987"
}
```

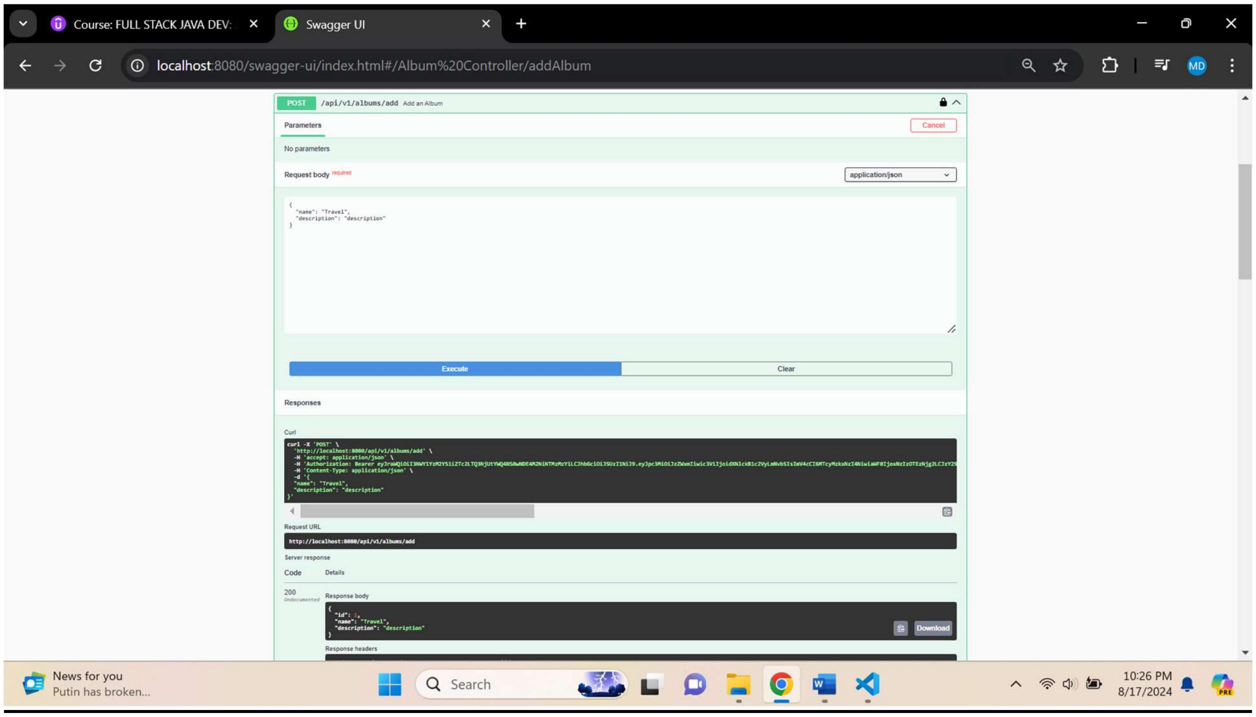
```
curl -X 'POST' \
  'http://localhost:8080/api/v1/auth/token' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "email": "user@user.com",
  "password": "pass987"
}'
```

```
http://localhost:8080/api/v1/auth/token
```

```
{
  "token": "eyJraWQiOiI3NWY1YzM2YS1iZTc2LTQ3NjUtYWQ4NS0wNDE4M2NiNTMzMzYiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1c2VyLmNvbSIsImV4cCI6MTcyMzkxNzI4NiwiaWF0IjoxNzIzOTEzNjg2LCJzY29wZSI6IlVTRVIifQ.aYKW2yXHdyavu2deQeACTVX7CBmcqhIz8s8lmly16jEp-BMfySXAFraMqui8AMmm1vtDcKjR_gV_MeE8qym_6p4IGZMW7c9twUnCwWJ2vNRXQVI3GkThvwC1ttyh70kA-vmUSn8D1FgWlK-7Q1FZePE7My_q_yz5z
pQio40594-uVUsKJ0PCNLgKYJtR0GCCpgJ3KaGtbwfCf7WBG5SD_S8YaGC26EDXbNMhGbJ7A6OxqbmHG0VlMI3XEZYxecHued7jUAZU7zYgf5sI43sc657by6CdbeX5EI4KcIjhXfpBAhYhA6XW3aUntpPlCsSwstg"
}
```

```
cache-control: no-cache,no-store,max-age=0,must-revalidate
connection: keep-alive
content-type: application/json
```

Add album



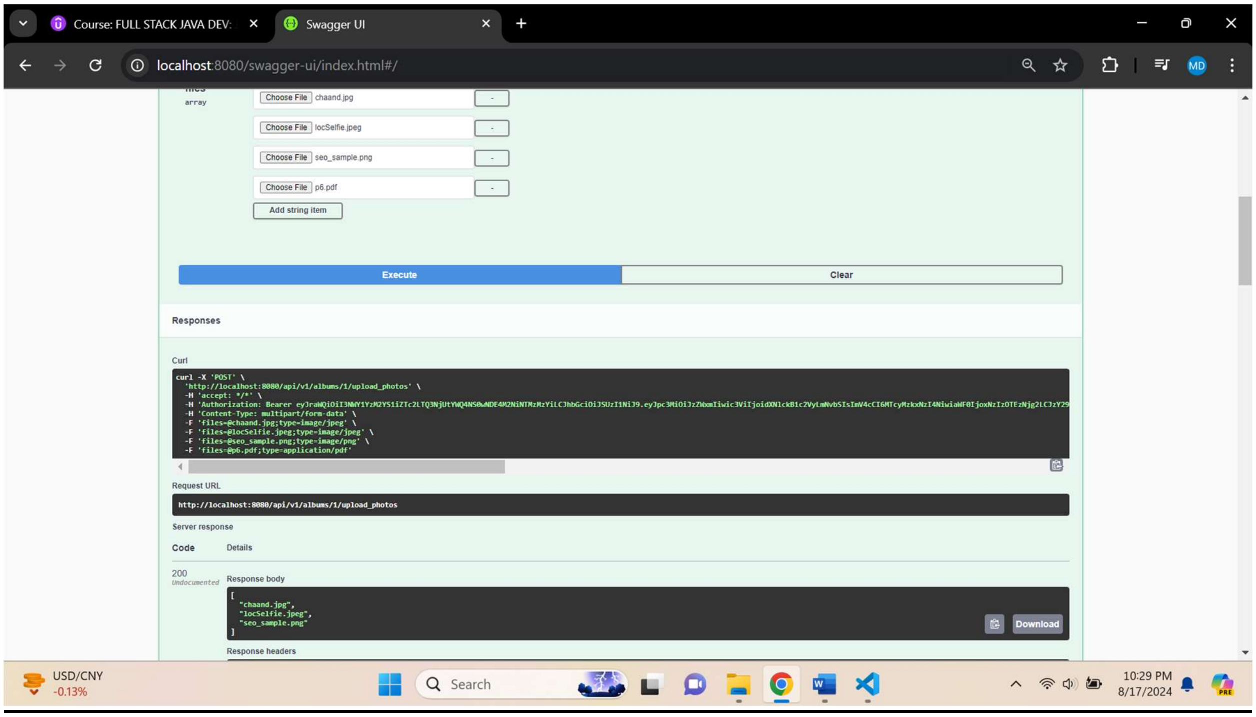
```
{
  "name": "Travel",
  "description": "description"
}
```

```
curl -X 'POST' \
  'http://localhost:8080/api/v1/albums/add' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer eyJraWQiOiI3NWY1YzM2YS1iZTc2LTQ3NjUtYWQ4NS0wNDE4M2NiNTMzMzYiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1c2VyLmNvbSIsImV4cCI6MTcyMzkxNzI4NiwiaWF0IjoxNzIzOTEzNjg2LCJzY29
  -H 'Content-Type: application/json' \
  -d '{
  "name": "Travel",
  "description": "description"
}'
```

```
http://localhost:8080/api/v1/albums/add
```

```
{
  "id": 1,
  "name": "Travel",
  "description": "description"
}
```

After adding images to Api


Course: FULL STACK JAVA DEV:
Swagger UI
localhost:8080/swagger-ui/index.html#/
array
Choose File chaand.jpg
Choose File locSelfie.jpeg
Choose File seo_sample.png
Choose File p6.pdf
Add string item
Execute
Clear
Responses

Curl

```
curl -X 'POST' \
  'http://localhost:8080/api/v1/albums/1/upload_photos' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJraWQiOiI3NWY1YzM2YS1iZTc2LTQ3NjUtYWQ4NS0wNDE4M2NiNTMzMzYiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJzZWxmIiwic3ViIjoidXNlckB1c2VyLmNvbSIsImV4cCI6MTcyMzkxNzI4NiwiaWF0IjoxNzIzOTEzNjg2LCJzY29
  -H 'Content-Type: multipart/form-data' \
  -F 'files=@chaand.jpg;type=image/jpeg' \
  -F 'files=@locSelfie.jpeg;type=image/jpeg' \
  -F 'files=@seo_sample.png;type=image/png' \
  -F 'files=@p6.pdf;type=application/pdf'
```

Request URL

```
http://localhost:8080/api/v1/albums/1/upload_photos
```

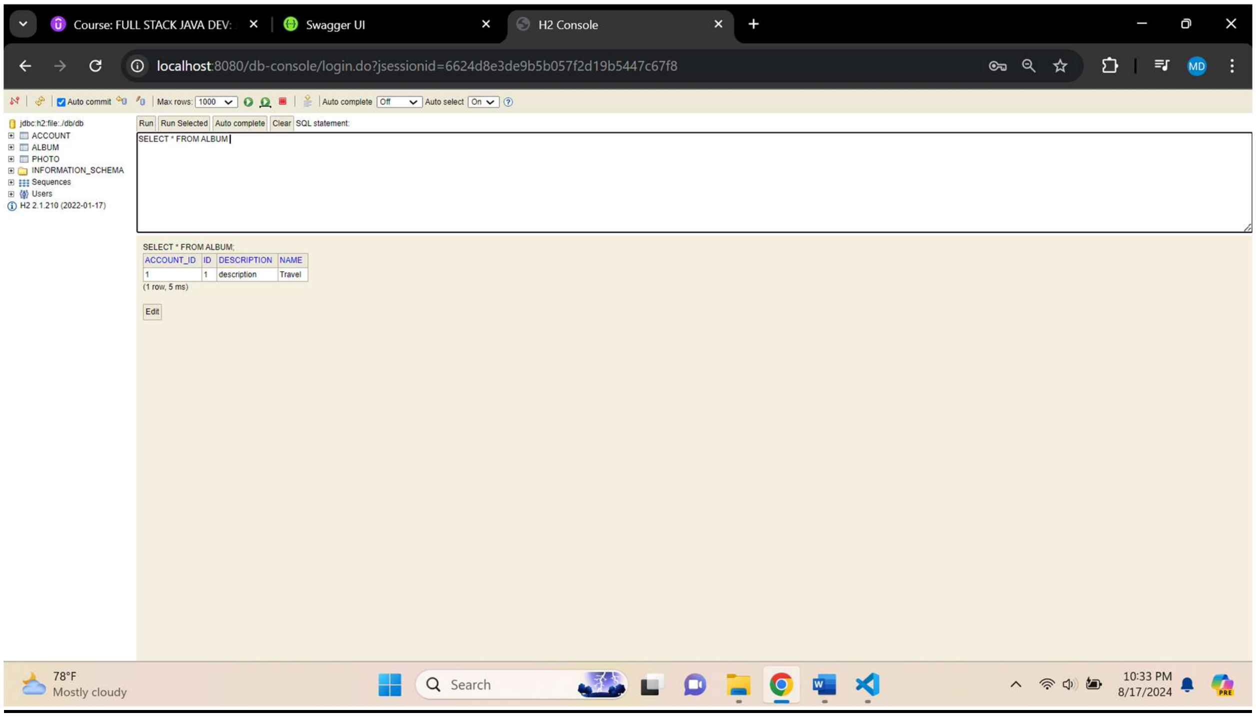
Server response

Code Details

200 *Undocumented* Response body

```
[
  "chaand.jpg",
  "locSelfie.jpeg",
  "seo_sample.png"
]
```

Db Console Output

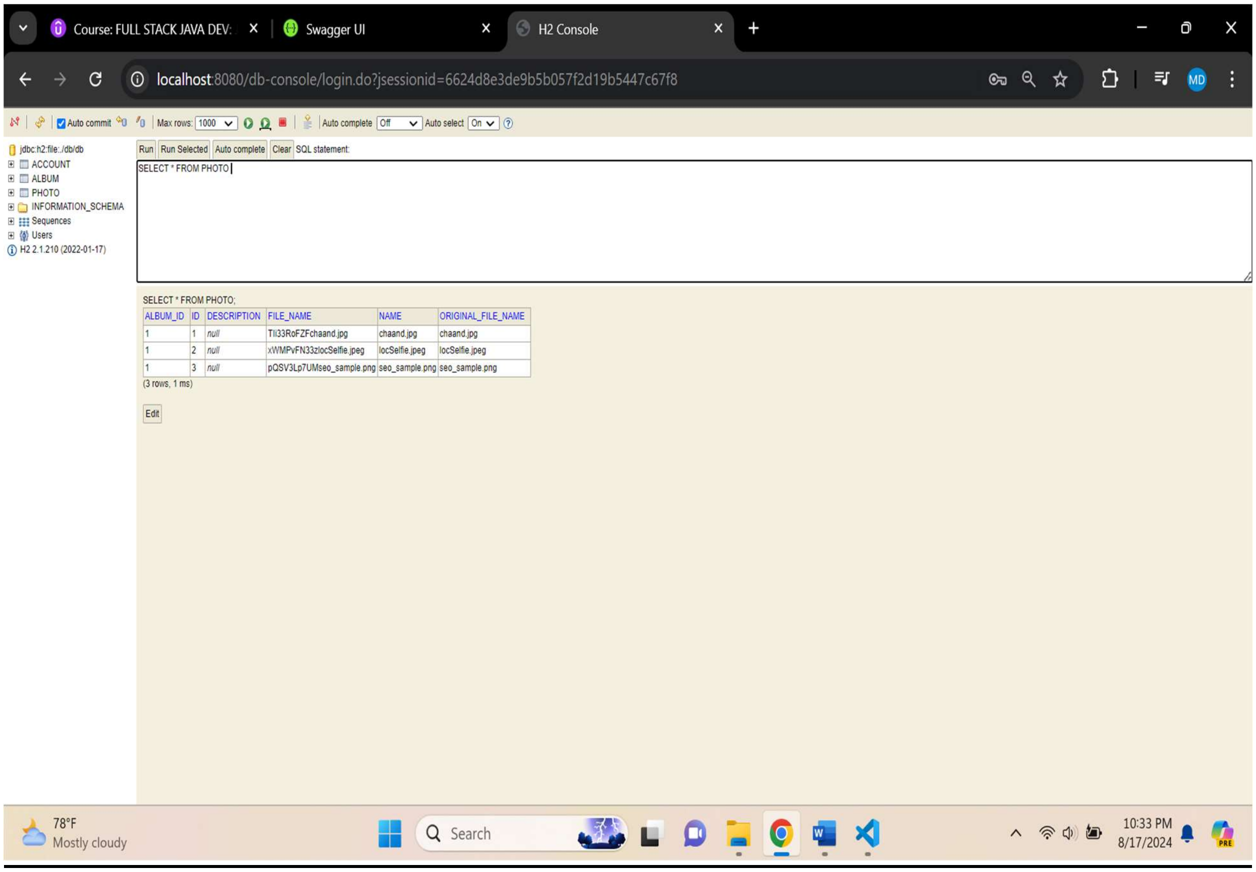


```
SELECT * FROM ALBUM
```

SELECT * FROM ALBUM;

ACCOUNT_ID	ID	DESCRIPTION	NAME
1	1	description	Travel

(1 row, 5 ms)



localhost:8080/db-console/login.do?jsessionid=6624d8e3de9b5b057f2d19b5447c67f8

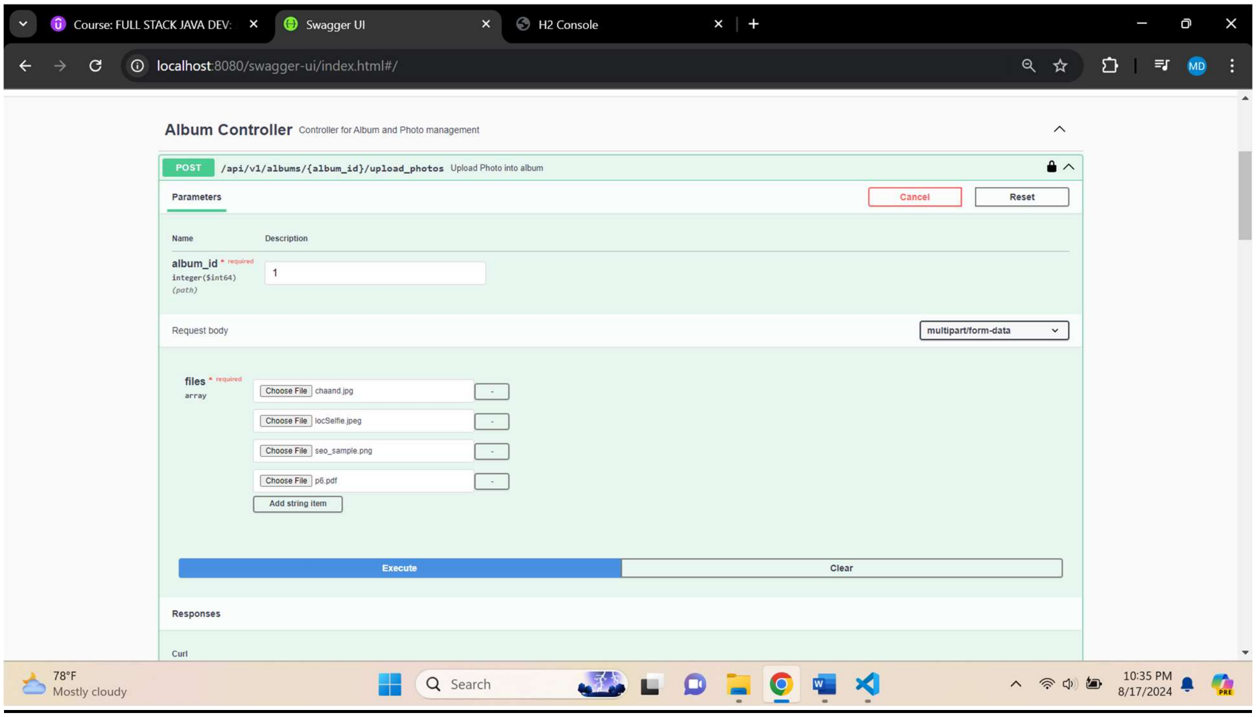
SQL statement:

```
SELECT * FROM PHOTO
```

SELECT * FROM PHOTO;

ALBUM_ID	ID	DESCRIPTION	FILE_NAME	NAME	ORIGINAL_FILE_NAME
1	1	*null*	Tli33RoFZFchaand.jpg	chaand.jpg	chaand.jpg
1	2	*null*	xWMPvFN33zlocSelfie.jpeg	locSelfie.jpeg	locSelfie.jpeg
1	3	*null*	pQSV3Lp7UMseo_sample.png	seo_sample.png	seo_sample.png

(3 rows, 1 ms)



localhost:8080/swagger-ui/index.html#/

Album Controller Controller for Album and Photo management

POST `/api/v1/albums/{album_id}/upload_photos` Upload Photo into album

album_id * required
integer($int64)
(path)
1

Request body: multipart/form-data

files * required
array
chaand.jpg
locSelfie.jpeg
seo_sample.png
p6.pdf

Small Little changes in the API

Updated AlbumController.java

```
package org.studyeasy.SpringRestdemo.controller;

import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardCopyOption;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Optional;

import org.apache.commons.lang3.RandomStringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.security.core.Authentication;
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.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestPart;
import org.springframework.web.bind.annotation.ResponseStatus;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.multipart.MultipartFile;
import org.studyeasy.SpringRestdemo.model.Account;
import org.studyeasy.SpringRestdemo.model.Album;
import org.studyeasy.SpringRestdemo.model.Photo;
import org.studyeasy.SpringRestdemo.payload.auth.album.AlbumPayloadDTO;
import org.studyeasy.SpringRestdemo.payload.auth.album.AlbumViewDTO;
import org.studyeasy.SpringRestdemo.service.AccountService;
import org.studyeasy.SpringRestdemo.service.AlbumService;
import org.studyeasy.SpringRestdemo.service.PhotoService;
import org.studyeasy.SpringRestdemo.util.AppUtils.AppUtil;
import org.studyeasy.SpringRestdemo.util.constants.AlbumError;

import io.swagger.v3.oas.annotations.Operation;
import io.swagger.v3.oas.annotations.responses.ApiResponse;
import io.swagger.v3.oas.annotations.security.SecurityRequirement;
import io.swagger.v3.oas.annotations.tags.Tag;
import jakarta.validation.Valid;
import lombok.extern.slf4j.Slf4j;

@RestController
@RequestMapping("api/v1")
@Tag(name = "Album Controller", description = "Controller for Album and Photo
management")
@Slf4j
public class AlbumController {

    @Autowired
    private AccountService accountService;

    @Autowired
    private AlbumService albumService;

    @Autowired
    private PhotoService photoService;

    @PostMapping(value = "/albums/add", consumes = "application/json", produces =
"application/json")
    @ResponseStatus(HttpStatus.CREATED)
    @ApiResponse(responseCode = "400", description = "Please add valid name a
description")
    @ApiResponse(responseCode = "201", description = "Account added")
    @Operation(summary = "Add an Album")
    @SecurityRequirement(name = "studyeasy-demo-api")
    public ResponseEntity<AlbumViewDTO> addAlbum(@Valid @RequestBody
AlbumPayloadDTO albumPayloadDTO,
            Authentication authentication) {
        try {
            Album album = new Album();
            album.setName(albumPayloadDTO.getName());
            album.setDescription(albumPayloadDTO.getDescription());

            // Extract Account
            String email = authentication.getName();

            Optional<Account> optionalAccount =
accountService.findByEmail(email);
            Account account = optionalAccount.get();
            album.setAccount(account);
            album = albumService.save(album);
            AlbumViewDTO albumViewDTO = new AlbumViewDTO(album.getId(),
album.getName(), album.getDescription());
            return ResponseEntity.ok(albumViewDTO);
        } catch (Exception e) {

            log.debug(AlbumError.ADD_ALBUM_ERROR.toString() + ": " +
e.getMessage());
            return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(null);
        }
    }

    // List all the albums based on the logged in users
    @GetMapping(value = "/albums", produces = "application/json")
    @ApiResponse(responseCode = "200", description = "List of albums")
    @ApiResponse(responseCode = "401", description = "Token missing")
    @ApiResponse(responseCode = "403", description = "Token Error")
    @Operation(summary = "List album api")
    @SecurityRequirement(name = "studyeasy-demo-api")
    public List<AlbumViewDTO> albums(Authentication authentication) {
        String email = authentication.getName();
        Optional<Account> optionalAccount = accountService.findByEmail(email);
        Account account = optionalAccount.get();
        List<AlbumViewDTO> albums = new ArrayList<>();
        for (Album album : albumService.findByAccount_id(account.getId())) {
            albums.add(new AlbumViewDTO(album.getId(), album.getName(),
album.getDescription()));
        }
        return albums;
    }

    @PostMapping(value = "/albums/{album_id}/upload_photos", consumes = {
"multipart/form-data" })
    @Operation(summary = "Upload Photo into album")
    @SecurityRequirement(name = "studyeasy-demo-api")
    @ApiResponse(responseCode = "400", description = "Please check the payload of
token")
    public ResponseEntity<List<String>> photos(@RequestPart(required = true)
MultipartFile[] files,
            @PathVariable long album_id, Authentication authentication) {
        String email = authentication.getName();
        Optional<Account> optionalAccount = accountService.findByEmail(email);
        Account account = optionalAccount.get();
        Optional<Album> optionalAlbum = albumService.findById(album_id);
        Album album;
        if (optionalAlbum.isPresent()) {
            album = optionalAlbum.get();
            if (account.getId() != album.getAccount().getId()) {
                return ResponseEntity.status(HttpStatus.BAD_GATEWAY).body(null);
            }
        } else {
            return ResponseEntity.status(HttpStatus.BAD_GATEWAY).body(null);
        }

        List<String> fileNamesWithSuccess = new ArrayList<>();
        List<String> fileNamesWithError = new ArrayList<>();
        Arrays.asList(files).stream().forEach(file -> {
            // Checking the type of the file is correct or not
            String contentType = file.getContentType();

            if (contentType.equals("image/png")
                    || contentType.equals("image/jpg")
                    || contentType.equals("image/jpeg")) {
                fileNamesWithSuccess.add(file.getOriginalFilename());
                // When we are storing the file in the database, there is a
possibility that
                // file name from the user
                // is repeating and if that happens, then file from the user will
get repeated
                // and in server will
                // get replaced.
                // In order to prevent that, we need to create a random string.;
                int length = 10;
                boolean useLetters = true;
                boolean useNumbers = true;
                try {
                    String fileName = file.getOriginalFilename();
                    String generatedString = RandomStringUtils.random(length,
useLetters, useNumbers);
                    String final_photo_name = generatedString + fileName;
                    String absolute_fileLocation =
AppUtil.get_photo_upload_path(final_photo_name, album_id);
                    Path path = Paths.get(absolute_fileLocation);
                    Files.copy(file.getInputStream(), path,
StandardCopyOption.REPLACE_EXISTING);
                    Photo photo = new Photo();
                    photo.setName(fileName);
                    photo.setFileName(final_photo_name);
                    photo.setOriginalFileName(fileName);
                    photo.setAlbum(album);
                    photoService.save(photo);
                } catch (Exception e) {

                }
            }else{
                fileNamesWithError.add(file.getOriginalFilename());
            }
        });
        return ResponseEntity.ok(fileNamesWithSuccess);
    }

}
```

Updated Account.java

```
package org.studyeasy.SpringRestdemo.model;

import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Entity
@Setter
@Getter
@ToString
public class Account {

    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private long id;

    @Column(unique = true)
    private String email;

    private String password;

    private String Authorities;
}
```