

Update Album API

Add below code in AlbumController.java

```
//Update Album
    @PutMapping(value = "/albums/{album_id}/update", consumes =
"application/json", produces = "application/json")
    @ResponseStatus(HttpStatus.CREATED)
    @ApiResponse(responseCode = "400", description = "Please add valid name a
description")
    @ApiResponse(responseCode = "204", description = "Album update")
    @Operation(summary = "Update an Album")
    @SecurityRequirement(name = "studyeasy-demo-api")
    public ResponseEntity<AlbumViewDTO> update_Album(@Valid @RequestBody
AlbumPayloadDTO albumPayloadDTO,
        @PathVariable long album_id, Authentication authentication) {
        try {

            String email = authentication.getName();
            Optional<Account> optionalAccount =
accountService.findByEmail(email);
            Account account = optionalAccount.get();

            Optional<Album> optionaAlbum = albumService.findById(album_id);
            Album album;
            if (optionaAlbum.isPresent()) {
                album = optionaAlbum.get();
                if (account.getId() != album.getAccount().getId()) {
                    return
ResponseEntity.status(HttpStatus.FORBIDDEN).body(null);
                }
            } else {
                return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(null);
            }

            album.setName(albumPayloadDTO.getName());
            album.setDescription(albumPayloadDTO.getDescription());
            album = albumService.save(album);
            List<PhotoDTO> photos = new ArrayList<>();
            for(Photo photo: photoService.findByAlbumId(album.getId())){
                String link =
"/albums/"+album.getId()+"/photos/"+photo.getId()+"/download-photo";
                photos.add(new PhotoDTO(photo.getId(), photo.getName(),
photo.getDescription()),
```

```

        photo.getFileName(), link));

    }

    AlbumViewDTO albumViewDTO = new AlbumViewDTO(album.getId(),
album.getName(), album.getDescription(), photos);
    return ResponseEntity.ok(albumViewDTO);

} catch (Exception e) {
    return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(null);
}

}

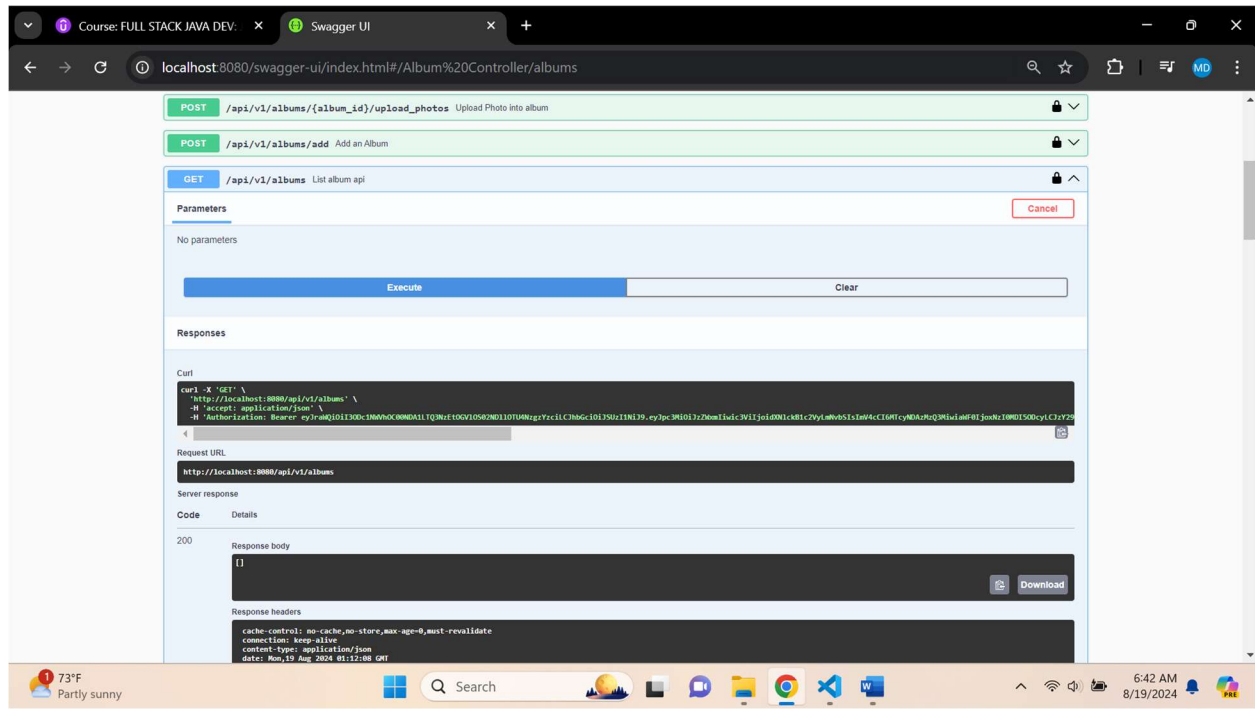
```

Output

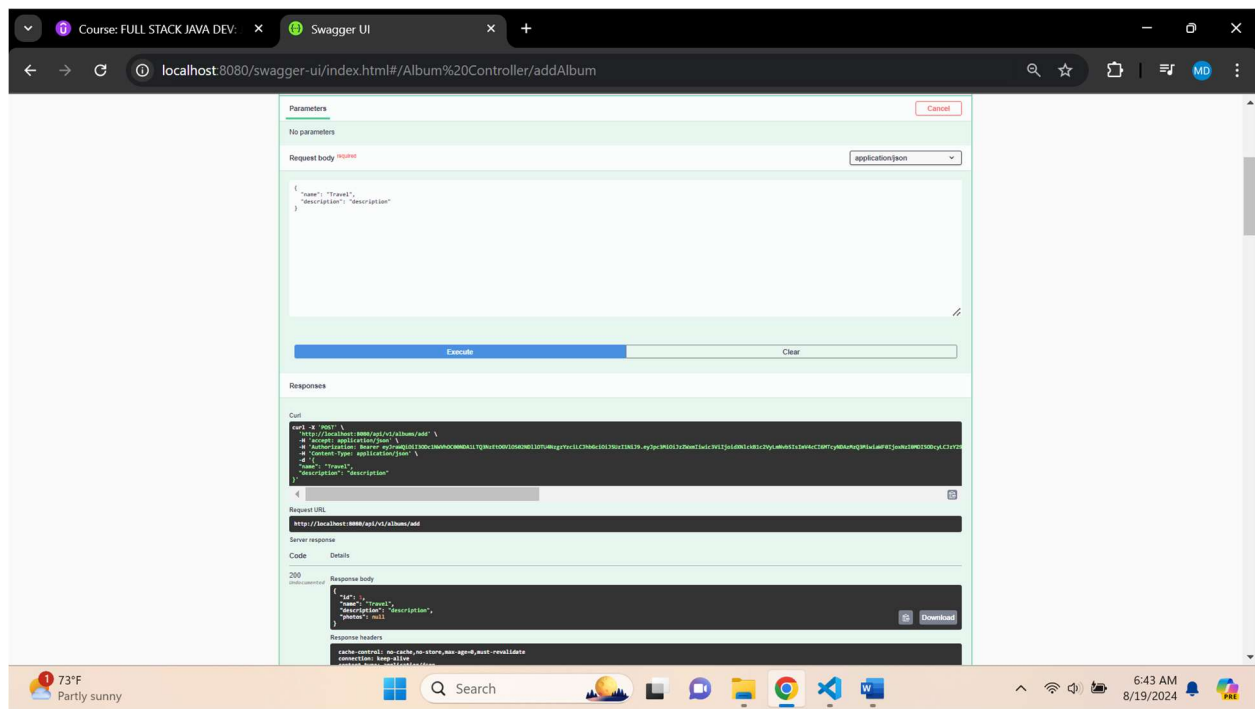
Token Generated to authrozie for user

The screenshot shows the Swagger UI interface for a REST API. The browser address bar displays `localhost:8080/swagger-ui/index.html#/Auth%20Controller/token`. The interface includes an **Execute** button and a **Clear** button. Below these, the **Responses** section is visible, showing the **Curl** command for a POST request to `http://localhost:8080/api/v1/auth/token`. The **Request URL** is also displayed. The **Server response** section shows a **200** status code. The **Response body** contains a long JWT token. The **Response headers** section lists various headers including `cache-control`, `connection`, `content-type`, `date`, `expires`, `keep-alive`, `pragma`, `transfer-encoding`, `vary`, `x-content-type-options`, `x-frame-options`, and `x-xss-protection`. The **Links** section is also visible at the bottom.

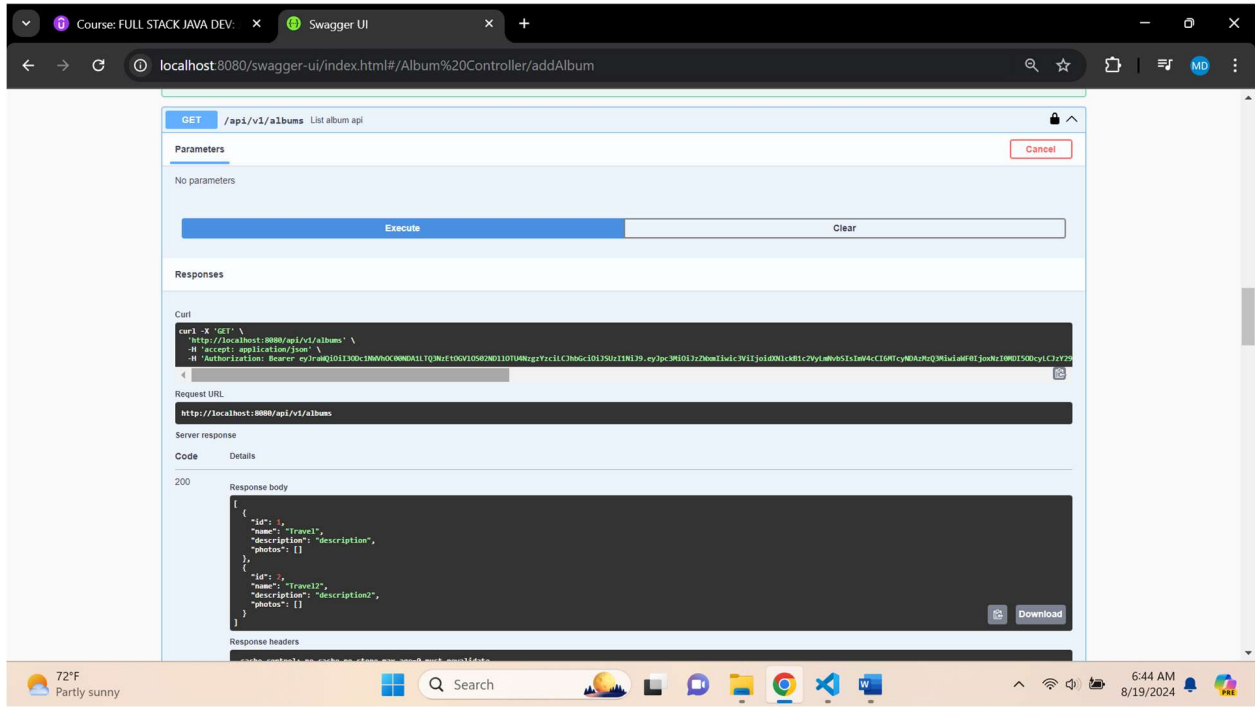
List of Albums - Starting



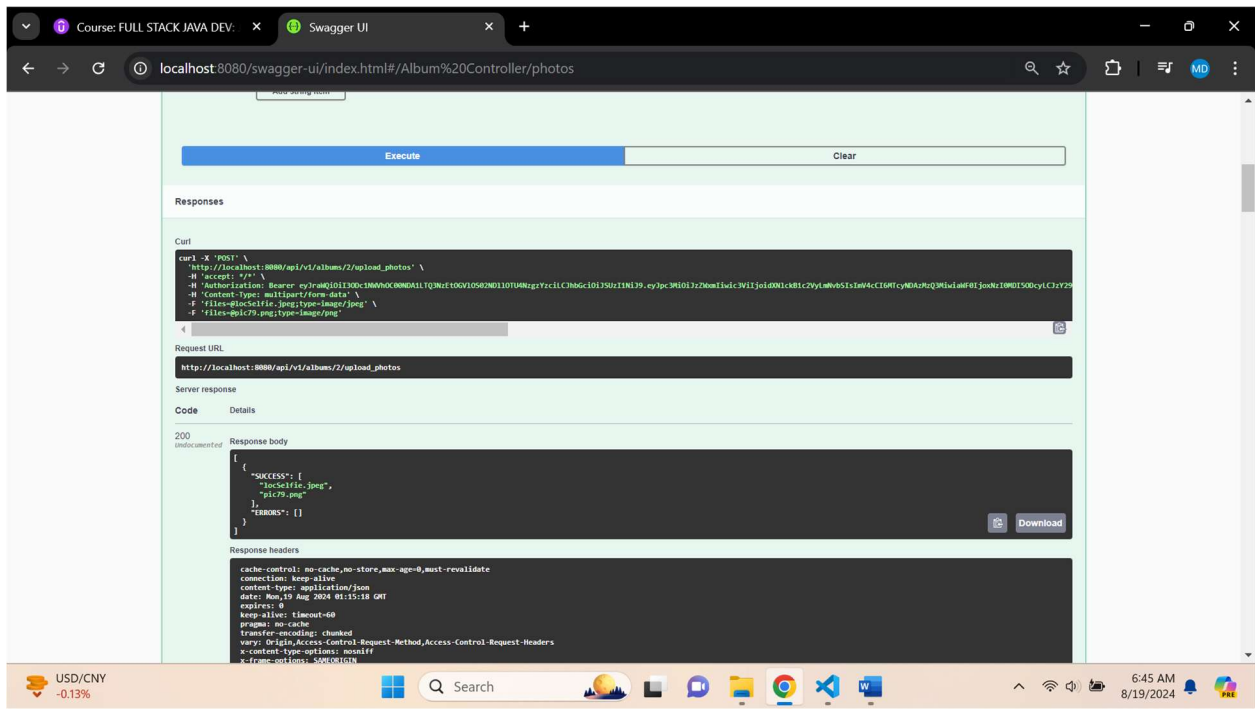
Add Album – More than 2



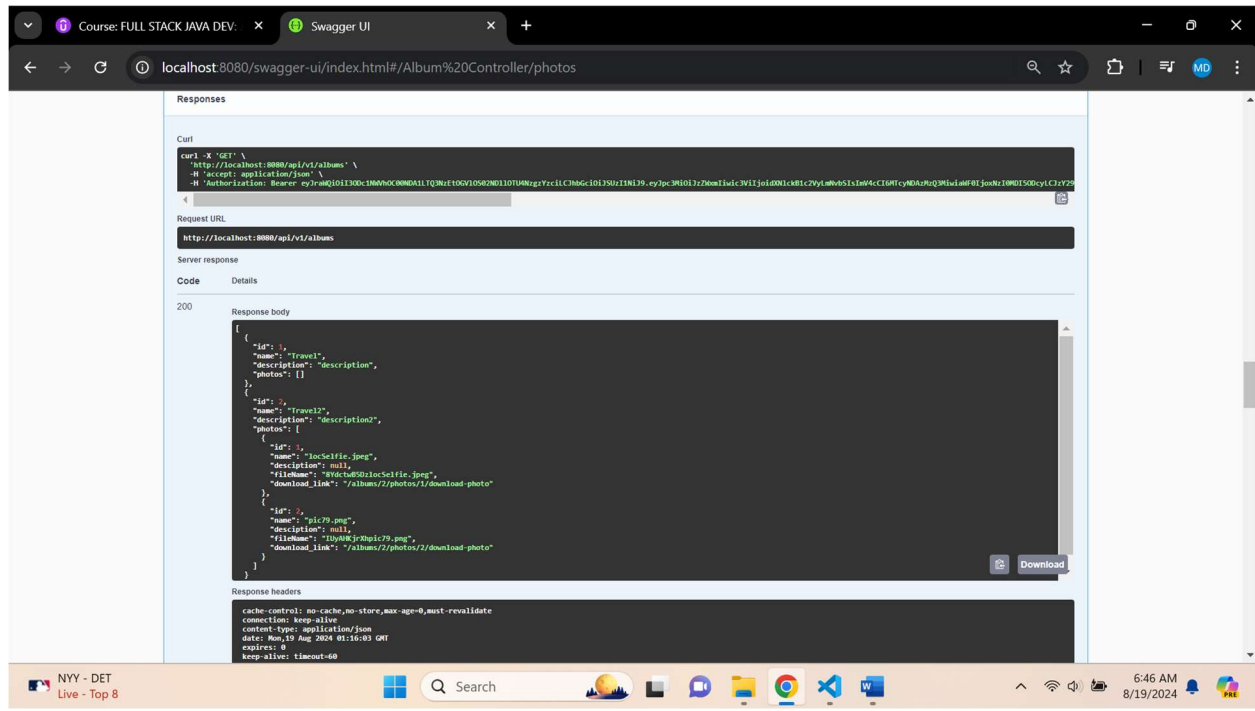
List of Albums with no photo uploaded



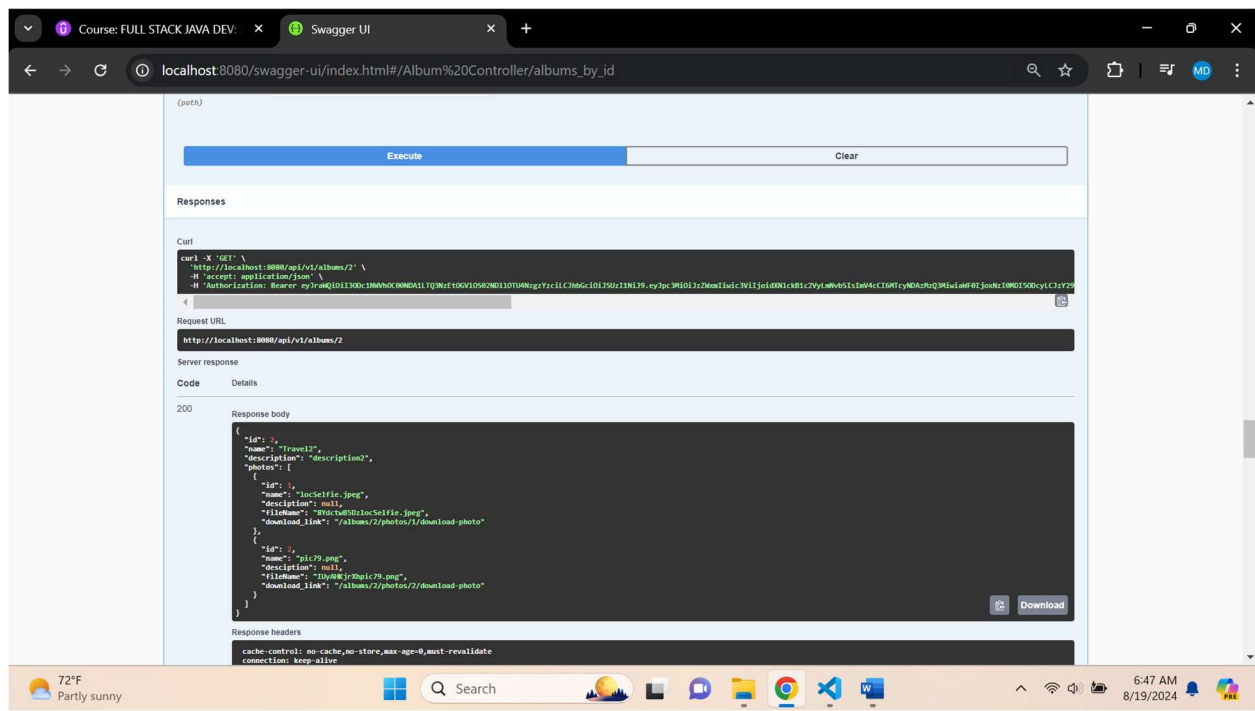
Updating Photo into album (id =2)



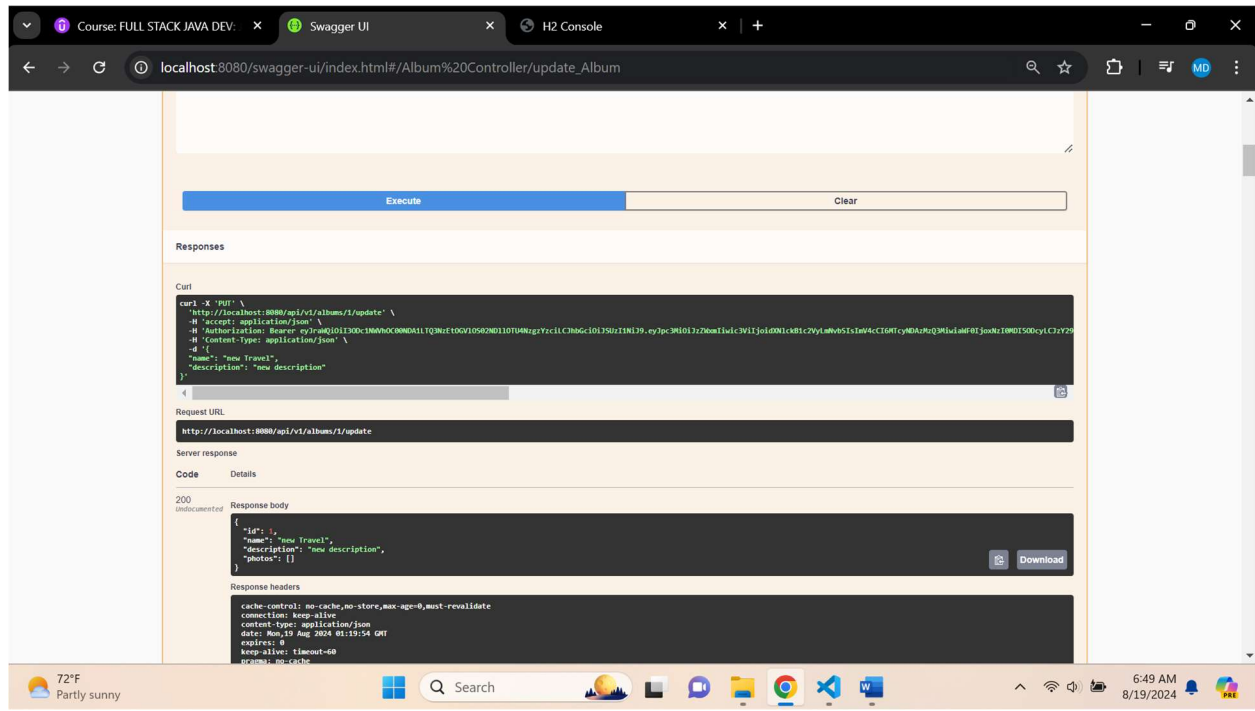
List of Albums with photo uploaded – id = 2



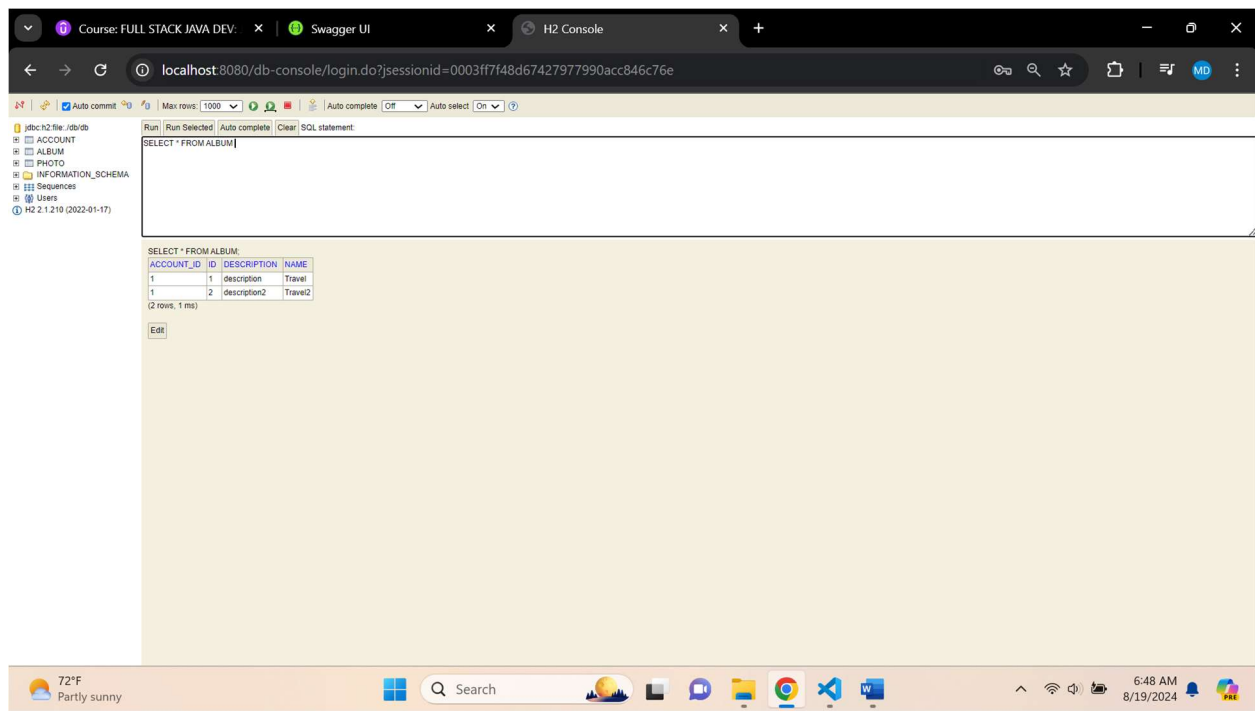
List album by album id



Updating Album



Db Console – Album – Before Update



Db Console – Album – After Update

Run (Run Selected) | Auto complete | Clear | SQL statement:

```
SELECT * FROM ALBUM
```

| ACCOUNT_ID | ID | DESCRIPTION | NAME |
|------------|----|-----------------|------------|
| 1 | 1 | new description | new Travel |
| 1 | 2 | description2 | Travel2 |

(2 rows, 1 ms)

changes Made

Final Frontend API Swagger View for AlbumController

Servers: http://localhost:8080 - Generated server uri

Authorize

Album Controller

Controller for Album and Photo management

- PUT /api/v1/albums/{album_id}/update Update an Album
- POST /api/v1/albums/{album_id}/upload_photos Upload Photo into album
- POST /api/v1/albums/add Add an Album
- GET /api/v1/albums List album api
- GET /api/v1/albums/{album_id} List album by album ID
- GET /api/v1/albums/{album_id}/photos/{photo_id}/download-thumbnail
- GET /api/v1/albums/{album_id}/photos/{photo_id}/download-photo

Auth Controller

Controller for Account management

- PUT /api/v1/auth/users/{user_id}/update-authorities Update authorities
- PUT /api/v1/auth/profile/update-password Update profile