java_springboot_restful_30_AlbumController – Build Photo Upload for Album With Thumbnails

Updated AppUtil.java

```java
package org.studyeasy.SpringRestdemo.util.AppUtils;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

//Receiving the filename and also the album_id
public class AppUtil {

    public static String get_photo_upload_path(String fileName, String 
folder_name ,long album_id) throws IOException {
        String path = "src\\main\\resources\\static\\uploads\\" + album_id + "\\" 
+folder_name;
        Files.createDirectories(Paths.get(path));
        return new File(path).getAbsolutePath() + "\\" + fileName;
    }

}
```

Updated AlbumError.java

```java
package org.studyeasy.SpringRestdemo.util.constants;

public enum AlbumError {
    ADD_ALBUM_ERROR,
    PHOTO_UPLOAD_ERROR
}
```

Add this in pom.xml

```xml
 <!-- Adding Scalar-->
 <dependency>
     <groupId>org.imgscalr</groupId>
     <artifactId>imgscalr-lib</artifactId>
      <version>4.2</version>
      <type>jar</type>
      <scope>compile</scope>
</dependency>
```

Updated AppUtil.java

```java
package org.studyeasy.SpringRestdemo.util.AppUtils;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

import javax.imageio.ImageIO;

import org.imgscalr.Scalr;
import org.springframework.web.multipart.MultipartFile;
import java.awt.image.BufferedImage;

//Receiving the filename and also the album_id
public class AppUtil {

    public static String get_photo_upload_path(String fileName, String 
folder_name, long album_id) throws IOException {
        String path = "src\\main\\resources\\static\\uploads\\" + album_id + "\\" 
+ folder_name;
        Files.createDirectories(Paths.get(path));
        return new File(path).getAbsolutePath() + "\\" + fileName;
    }

    public static BufferedImage getThumbnail(MultipartFile orginalFile, Integer 
width) throws IOException {
        BufferedImage thumbImg = null;
        BufferedImage img = ImageIO.read(orginalFile.getInputStream());
        thumbImg = Scalr.resize(img, Scalr.Method.AUTOMATIC, 
Scalr.Mode.AUTOMATIC, width, Scalr.OP_ANTIALIAS);
        return thumbImg;
    }

}
```

Updated photo method in AlbumController.java

```java
  @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<HashMap<String, 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, PHOTOS_FOLDER_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);

                    BufferedImage thumbImg = AppUtil.getThumbnail(file,
THUMBNAIL_WIDTH);
                    File thumbnail_location = new
File(AppUtil.get_photo_upload_path(final_photo_name, THUMBNAIL_FOLDER_NAME,
album_id));
                    ImageIO.write(thumbImg, file.getContentType().split("/")[1],
thumbnail_location);

                } catch (Exception e) {
                    log.debug(AlbumError.PHOTO_UPLOAD_ERROR.toString()+ ":
"+e.getMessage());
                    fileNamesWithError.add(file.getOriginalFilename());
                }
            }else{
                fileNamesWithError.add(file.getOriginalFilename());
            }
        });

        HashMap<String, List<String>> result = new HashMap<>();
        result.put("SUCCESS", fileNamesWithSuccess);
        result.put("ERRORS", fileNamesWithError);

        List<HashMap<String, List<String>>> response = new ArrayList<>();
        response.add(result);
        return ResponseEntity.ok(response);
    }
```