« 5.2 Cloud software in practice (/css/2... 5.4 Assignment: asynchronous messagi...

CS-E4190 (/css/2022/) / 5. Design principles and practices (/css/2022/design/) / 5.3 Assignment: API design with OpenAPI

Assignment: API design with OpenAPI

In this assignment you will learn about OpenAPI specification and documentation. You may read more about OpenAPI specification here (https://swagger.io/specification).

Photo management API

Your task is to create an OpenAPI 3 specification in YAML format that serves basic CRUD (Create, Read, Update, Delete) for photo management via HTTP requests. Basics about HTTP requests and response codes can be found here (http://www.restapitutorial.com/lessons/httpmethods.html).

The specification should be able to process each request and provide the output in JSON format. The paths and schemes required for the specification are described below.

Schema name: `Photo`

Attribute	Type	Conditions
id	string	readOnly:true
name	string	required:true - maxlength:20
description	string	maxlength:100
access	string	required:true - enum:[public, private]
location	string	required:true
file	string	required:true
created_date	string	required:true - format:date-time
modified_date	string	required:true - format:date-time

Schema name: `Photos`

Definition	Type	Description
[Photo]	array	An array of photos

Schema name: `Success`

Attribute	Type	Conditions
message	string	–
id	string	–

Schema name: `Error`

Attribute	Type	Conditions
message	string	–

Paths

Request	Type	Route	Request body	Response Body
Create	POST	/photo	Photo	`Success` : {message: 'Photo successfully created', id: `photo_id` } - **Status code**: 201 (Created)
				`Error` : {message: `err` } - **Status code**: 422 (Unprocessable entity)
READ	GET	/photos	access(*)	`Photos` : [{ `Photo` }, ..] - **Status code**: 200
				`Error` : {message: `err` } - **Status code**: 404 (Not Found)
READ	GET	/photo/{photo_id}	–	`Photo` : {id: `id` , name: `name` , description: `description` , access: `access` , location: `location` , created_date: `created_date` , modified_date: `modified_date` } - **Status code**: 200
				`Error` : {message: `err` } - **Status code**: 404 (Not Found)
UPDATE	PUT	/photo/{photo_id}	Photo	`Success` : {message: 'Photo successfully updated', id: `photo_id` } - **Status code**: 200
				`Error` : {message: `err` } - **Status code**: 404 (Not Found)
				`Error` : {message: `err` } - **Status code**: 422 (Unprocessable entity)
DELETE	DELETE	/photo/{photo_id}	–	**Status code**: 204 (No Content)
				`Error` : {message: `err` } - **Status code**: 404 (Not Found)

Note

(*) The requestBody in the GET statements should be sent as query parameters. The remaining routes receive request body through JSON.

Note

All the paths should have a default response using the `Error` schema.

Attention

You should use all the required fields according to the OpenAPI specification version 3.0.0 or above. This assignment unit test uses “openapi 3.0.0”. It is mandatory that you write all the relevant parameters, objects and data types in order to get a successful request and response. You can test you OpenAPI specification at the following link (https://editor.swagger.io/).

Grading

Submission are evaluated by means of an automated system. You only need to submit a single file, namely the OpenAPI specification file. Once your submission is graded you will receive the result of the evaluation.

Note

The assignment runs multiple unit tests which give fractional points based on how the requirements in task are fulfilled, according to the table below.

Test	Points
Required paths exist	9
Required schemas exist	9
Required fields in schemas exist	9
Required schema properties exist	9
Data-type of photos are correct	8
Methods of the paths are correct	9
Required responses exist	9
Required parameters exist	8

⚠ The deadline for the assignment has passed (Wednesday, 23 November 2022, 14:30).

Microservices design: the OpenAPI 3 API description format

Upload YAML file

You have to upload a YAML file for this assignment. The name that you choose for the uploaded file is irrelevant and does not affect the automated grading.

openapi.yaml

Choose File No file chosen

Submit

« 5.2 Cloud software in practice (/css/2... 5.4 Assignment: asynchronous messagi...