ADInt

Laboratory 5

In this laboratory students will continue on the development of Web applications (with REST Web Services) with the support of an ORM.

In this laboratory students will modify the work done on the previous laboratories to implement a library for courses related material.

Although student will complement the previous flask application with new endpoints, no changes on the user interface should be done.

Students will only implement REST endpoints that can be called from other applications (for instance python) or accessed with specific debug tools (such as postman)

1 Exercise

Student should implement a web application that will store files (as with the previous laboratory) but will allow users to assign extra information to each file (such as course information and material type).

Each file will be assigned one specific course, such as ADInt and a type of material (exam, lab_assignment, exercise_resolution, …)

This new system will work as follows:

- files are uploaded into the system using the forms created in the previous class.
- To assign a course or file type, students should implement the new REST endpoints.

1.1 REST Endpoints

In the development of the following endpoints , student should follow a simple set of rules:

- All the GET endpoint should return data in JSON
- All POST and PUT methods should receive data in JSON
- In case of success the endpoints should return the 200 HTML Code
- In case of error the endpoint should return the suitable HTML error code

List of endpoints to implement:

- **GET /API/courses/** – Lists all the courses that have been created on the system
- **POST /API/courses/** - Creates a new course. This endpoint receives in the message body a json similar to: **`{“course”: “ADInt”, “name”: ”Aplicações Distribuídas sobre a Internet”}`**

- **GET /API/courses/<course_id>/** - return the information about a course: name and number of assigned files such as in this example **`{“course”: “Aplicações Distribuídas sobre a Internet”, “n_files”: 10}`**
- **GET /API/courses/<course_id>/files/** - return the list of files that are assigned to such course. This list contains the type of file: **`[{“file”: “exam1.pdf”, “type:”exam”}, ...]`**
- **POST** /**AP/\courses/<course_id>/files/** - Inserts a files into a course. This endpoint receives in the message body a json similar to: **`{“file”: “exame1.txt”, “type”:”exam-resolution”}`**
- **GET /API/courses/<course_id>/types/** - Return the list of the type of the files that are assigned to such course, sucha as: **`[”exam-resolution”, ”exam”]`**
- **GET /API/courses/<course_id>/types/<types>/** - Return the list of the files assigned to such course and of the specified type.
- **GET /API/files/** – Lists all the files in the system
- **GET /API/files/<file_name>/** - send the content of the file or the file information. If the client requests the data to be in **json** format then the endpoint should return the file information: **`{“name”: “xxx.txt”, “course”:”ADInt”, “type”: “exam”, “downloads”: 10}`**, otherwise should return the file content.

1.2 JSON

To easily transform python data-structures to JSON and configure the response, use the **`flask.json.jsonify`** function.

This function is automatically called if the method returns an object (that is transformed to json):

https://flask.palletsprojects.com/en/1.1.x/quickstart/#about-responses

1.3 Error Codes

Each endpoint/operation should return a specific HTTP error code if it is impossible to correctly run it.

Verify what are possible HTTP error codes and return them from flask:

https://developer.mozilla.org/pt-PT/docs/Web/HTTP/Status

https://flask.palletsprojects.com/en/1.1.x/quickstart/#redirects-and-errors

1.4 Data model

The system should implement new tables that contains the relation between files and courses and types.

Students that already know SQL can create a two other table so that the database contains the following tables:

- Files
- Courses
- File_downloads
- Files_Assignments

2 Invocation of endpoints

2.1 Debugging

The endpoints can be called from any program (python or JavaScript) but for testing purposes it is easier to call them from a specific program.

In order to experiment with the API without programming it is possible to use a REST API client that provides an UI for the definition of the request and performs all the communication.

https://alternativeto.net/software/postman/?license=opensource

https://www.slant.co/topics/7913/~rest-api-clients

2.2 Python

Implement a simple Pyhton program that list all Courses and files.

This program runs o the terminal and interacts with the services, calling the provided endpoints using the python requests library

https://requests.readthedocs.io/en/latest/