EPITECH. TECHNOLOGY

PICTSMANAGER

TAKE, STORE, TAG AND HANDLE PICTURES

PICTSMANAGER

Let's build a mobile application and the associated server to take and share pictures.
Pictures are classified in albums, that have an owner and a list of people who can view it.

There are some requirements for this application, driven by usability:

- ✓ The application only stores pictures' names and associated ID. Pictures, metadatas and other informations are stored in the distant back server and can be retrieved and/or updated with specific requests.
- ✓ Pictures you take from your phone camera need to be compressed before being stored. You have to find a compromise between performance and quality, but the pictures should be visually acceptable.

Do all views require the same compression ratio?

- ✓ The user that created an album becomes its only owner, as well as the owner of all the pictures it contains. This ownership cannot be shared, transferred or resigned.
- ✓ Only the owner of a picture/album can modify/delete it.
- ✓ Users can only see pictures/albums they own or have been granted access to.

✓ There must be a research engine implemented explicitly (no external library is allowed).

✓ The application should be reactive. There should not be a lag when accessing pictures.

Select carefully the algorithms and database models you use, and optimize whatever you can!

✓ The architecture of the application must be designed with scalability in mind.

You may re-use the structure of an application you developed during a previous project, but this time you will have to focus on more advanced topics, including storing and parsing data efficiently.

UX is fundamental in such an application.
You may follow guidelines for design development, or you could create and justify yours.
Especially, **accessibility** is important. By integrating accessibility features and services, you could improve your app's usability.

Delivery

We require you to containerize each of the services with docker and orchestrate them with **docker-compose**.

The construction of your mobile applcation and your back must be automated and run within the container. The docker project will compile, test, package and deploy your program, including the database.

As usual:

1. the code must be duly documented *(preferably following the JavaDoc syntax)* and you must provide Software Architecture Specification and Software Qualification ;
2. a testing policy must be implemented ;
3. all errors and important messages should logged in a clear manner.

To fully demonstrate all the functionalities of your app, you should fill it with **many** albums and pictures from **various** users.

Bonuses

You can improve this project in many ways, including:

- ✓ adding a real authentication protocol ;
- ✓ increasing your applcation's security ;
- ✓ building an admin view for the server ;
- ✓ adding formal specification documents ;
- ✓ conducting a full functional testing process ;
- ✓ providing automated tag suggestion based on ML.

v 2.0.1