DISYS - Mandatory Exercise 1

IT University of Copenhagen
Lecturer: Marco Carbone and Agata Przybyszewska
GitHub repo: https://github.com/Arneproductions/DISYS-Exercise-1

Albert Rise Nielsen (albn)
Amalie Bøgild Sørensen (abso)
Andreas Nicolaj Tietgen (anti)
Asger Brødslev Mathiasen (asgm)

We know less and less the more we know
5. oktober 2021

1 REST service

To take a look at our REST service go to the github repository release called "Client and API server".

1.1 What operations should be using GET, PUT, POST, DELETE?

GET is used when wanting to retrieve a specific resource, fx the different courses, students, etc.
PUT is used when updating a specific resource, fx updating a user or adding a user to a course.
POST is used when wanting to create a new resource or to perform an action, fx adding a course or a satisfaction.
DELETE is used when wanting to remove a specific resource, fx deleting a student or deleting a student from a specific course.

2 Examine if our service is a microservice

To take a look at our microservice go to the github repository release called "microservice".

2.1 Which endpoints of the web service *could* have a different lifecycle / supporting team?

If our API is sold as a SaaS tool, we will need to make regular updates to the product API. In this example, the endpoint with "workloads" could have a lifecycle, which required faster updates or that it was expanded with more features.

Therefore, the workload lifecycle might need extra step where the users of the API were monitored or interviewed about the new updates so that the support team can improve the API.

2.2 Which operations could be asynchronous?

- When course wants to know which students are taking the course, we need an asynchronous operation between the course microservice and the student microservice.
- When wanting to add a workload to a course we need an asynchronous operation between the course microservice and the workload microservice.
- When wanting to add a student's degree of satisfaction with a course, we need an asynchronous operation between the course microservice, the satisfaction microservice and the student microservice.

2.3 What is the difference between a RESTful API and a set of microservices?

A RESTful API is a service that satisfy some different rules. For example it has some specific verbs: GET, PUT, POST, DELETE. Furthermore, it has to rely on HTTP coding.

A microservice is a service that is implemented in such a way that is consists of small, individual services, where each of them can follow the REST principles. Within the application, each microservice has one clearly defined job, eg. creating users.

3 Using gRPC

To take a look at our gRPC endpoint go to the github repository release called "gRPC endpoint".

3.1 When should gRPC be favored, and when should REST be favored?

In case you are building an application which is designed to prioritize speed more than browser compatibility, it will be better to use gRPC instead of REST. This is due to gRPC uses a binary formatter.

gRPC uses the binary protocol to send data, which is faster than using JSON/XML like REST does. Though, due to the low browser compatibility, gRPC is best suited for internal and private systems.