AAD Backend Development

Assignment 2: design and implementation

Assignment 2: design and implementation

The goal of the second assignment is to expand one of the prototypes to create a full backend API for your product and for the chosen advanced track. You will create a technical design in the process, including a formal API specification (such as OpenAPI or some other standard). You will create a backend application that is stable and robust and adheres to the standards of our profession. For maintainability you will create tests and dockerize all parts of the application.

You are allowed to implement a system for your own idea, this means that you have a lot a freedom. Because the research phase has been done, and you already have a substantial part of the design done, this assignment will be largely up to you. But you are still responsible for creating something that is complex enough for 5 ECTS.

Submission details for assignment 2

The **minimum requirements** for grading are:

- There is a technical specification with a formal API definition
- The application implements the basic application requirements (see below)
- The application implements the basic requirements for the advanced track (see below)
- There are no build artifacts (like node_modules) in your submission
- The goal of this assignment is to design and create a full backend AP

The **basic application requirements** are:
- For persistence the application uses a database that can be self-hosted. Sqlite is not a valid option, because it can't be hosted. Firebase is also not a valid option, because it can't be self-hosted. Postgres, MS SQL, RavenDB, MongoDB and MariaDB are valid options, because all can be hosted in a container.
- The application implements a formal API using the API standard in the technical spec. This can be REST or GraphQL or SPARQL.
- There are at least five resources exposed by the API, excluding "meta-resources" that are needed for logging, auth, etc.
- The code is layered (following the separation of concerns principles)
- The application is fully dockerized, without extra steps
- There is an installation guide including sample credentials to test the application with

Additional requirements for the application are
- Your application support multiple users with multiple roles and different access to the system.
- Your application includes tests to prove it is functional
- Your document includes test reports to show that the application works

Technical documentation should include the following:

- A context description and a list of (technical) requirements.
- An overview of the components and structure of your application. Describe each component:
  - Explain and illustrate the technical details, add diagrams where it seems fit
  - Include justification of implementation choices you have made (and alternatives you considered).
  - Describe third-party libraries used. If there are alternatives, describe them and also describe why you chose the library you did.
  - Describe the structure and subcomponents. Describe each subcomponent by repeating this list. Repeat until you have enough detail as you seem fit.
- Also describe the deployment and tests
- Include details of your advanced track.

On Blackboard you submit the following **artifacts**:

- A technical design document (pdf)
- The application, code, docker configuration, etc (zipped)
- A show-and-tell document that "sells" the features of the advanced track. This can be a tutorial-like document with screenshots that showcases your work (pdf).
- A reading guide for Backend Development (see below)
- additional documents for the advanced track (pdf)

You submit most artifacts at the generic submission point for all three the courses DE, BD and HD. You submit a reading guide specific for BD at the submission point for BD.

In week 9 or 10 all three courses will be graded in a single **assessment**.

The deadlines can be found on Blackboard.

Please note: we expect a ready-for-prime-time, professional backend; an application that you can deploy or deliver to a client.