| | | | |
|---|---|---|---|
| Assignment | Final assignment Data Processing | Programme | **Computer science/technical computer science** |

| | |
|---|---|
| **Student name** | **Student number** |
| **Programming language(s)** | **Framework(s)** |
| **Link to repository** | |
| Maximum number of points to be scored | 100 |
| Number of points by which the final assignment was achieved | 55 |

**Final assignment conditions:**

- The cover sheet should be completed and handed in by the end of week 1 at the latest.
  The following data must be supplied:
  - Student name and numbers of participants group
  - Chosen programming language(s)
  - Chosen Frameworks.
- For each survey moment, the cover sheet is refilled and modified where necessary. For example: modifications to programming languages or chosen frameworks. Also provide the necessary information requested in this document.
- The deadline for submitting the final assignment is **19 January 2023 17:00**.
- The resit is two weeks after the grade is announced via Progress.
- If the entire assignment is not met, an NBB (Not Assessable) is issued instead of a full assessment.

## Assignment description Data Processing

The final assignment consists of 3 different parts. Building an API, maintaining a database and programming a front-end suitable for data visualisation.

Before any programming is allowed, the necessary preliminary work will have to be done to establish what will be built. In the first phase of this assignment, a design will have to be created that creates clarity about how the entire system will work and what requirements it will meet. We will call this information the *system design*.

A use case is provided for background information, read it carefully to understand the customer's intention. Ask questions where necessary.

## System design

The system design will contain all necessary information about the system to be built and will include at least the following information.

About the API, the system design will include a *class diagram* that provides insight into how the API is built. Furthermore, it contains the information about the endpoints to be built and the relevant data that will serve as input and output for these endpoints. The API section will be written using the standards of the *OPENAPI specification* found at https://spec.openapis.org/oas/v3.1.0.

About the database, the system design will contain an ERD that provides information about all tables, relationships and attributes within the database system. The ERD contains relationships with both maximum and minimum cardinality indicated. Furthermore, there is room within the system design to map other components of the database system, think about the views, stored procedures, triggers, etc. used.

With regard to the entire system, the system design will include an architecture slab that provides insight into how the various components (front-end, API and database) interact and how the exchange of information with each other will be expressed. This design can be supported with examples of endpoints, requests or other examples that capture the imagination.

## API

The API will function as a *middleman* system between the front-end and the database and will satisfy at least the following conditions:

- The API aims to send and receive large amounts of data.
- The API interfaces with the underlying database system.
- The API accepts data in an established format and can present this data in at least two different formats. For example, XML, JSON, CSV, etc.

- The API has several endpoints available, each of which has the ability to return a correct response with the corresponding status code.
- The API has authentication and authorisation capabilities for different users.
- The API can validate input data and provide an appropriate response for invalid data.
- The API must interface with an external public API.

The API will have to be built using a framework. The choice of framework will have to be submitted to the examiners for determination. This can be done by submitting the cover sheet of this document.

## Database

To set up a database, an SQL file will be supplied. This SQL file contains a number of tables with associated data on which to base the database. **PLEASE NOTE:** The database is incomplete and will need to be modified. Think of adding relations, new tables, etc. An ERD must be designed based on these tables, which will form the basis of the database system to be realised.

In addition to design and implementation, measures will have to be taken to improve database integrity. Consider, for example, the creation of views, stored procedures and referential integrity. The options available and their implementation depend on the chosen DBMS.

In terms of authorisation, the database will need to have several users. The 'Front-End' section introduces these users. The users will need to have appropriate access to the database with appropriate GRANT and DENY options designated where necessary.

Besides integrity and authorisation, a backup protocol will also have to be developed. This will include how often backups should be made and where they are kept. The choice of backup type and the justification for this choice will also have to be mentioned in this protocol.

## Front-END

The front-end user panel will have to make all information insightful based on user input. Based on the use case, it will have to be determined exactly what information is relevant. All information made available by the API will have to be given a place in the user panel.

There are three different users available for the user panel: *Juniors*, *Mediors* and *Seniors*. Seniors have access to all relevant information, with no exceptions.

Mediors have access to all relevant information with the exception of information related to the financial side of the use case.

Juniors have access to all relevant information with the exception of information related to finances and privacy-sensitive information (name and address details).

These different users will have to be 'constrained' at the API and database level.