Student management

Analysis and Design Document

Student: Bercean Andrei

**Group: 30233**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 3

5. Class Design 3

6. Data Model 3

7. System Testing 3

8. Bibliography 3

1. Requirements Analysis

# Assignment Specification

Design and implement a Java application for the management of students in the CS Department at TUCN. The application should have two types of users (student and teacher/administrator user) which have to provide a username and a password in order to use the application.

# Functional Requirements

The regular user can perform the following operations:

* Add/update/view client information (name, identity card number, personal numerical

code, address, etc.).

* Create/update/delete/view student profile (account information: identification number,

group, enrolments, grades).

* Process class enrolment (enroll, exams, grades).

The administrator user can perform the following operations:

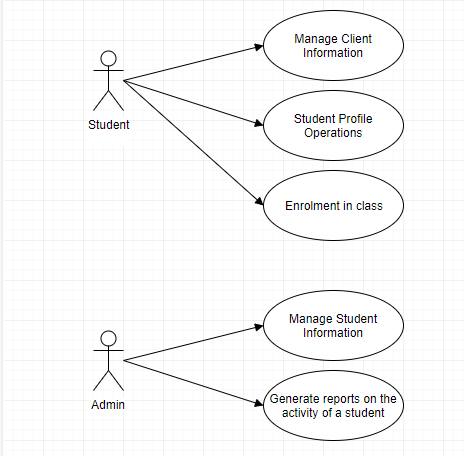
- CRUD on student’s information.

- Generate reports for a particular period containing the activities performed by a student.

# Non-functional Requirements

A necessary requirement of the non-functional sort is the guaranteed integrity of data, which is guaranteed by the usage of a SQL database for storing the information.

2. Use-Case Model

**

Use case: Student enrolls in a class

Level: user-goal level

Primary actor: Student

Main success scenario: After the login, the user selects the correct class and selects the enrollment operation on said class. The enrollment takes place and the information in the database is updated.

Extensions: Should the class ID not be found in the database, the enrollment will result in failure. Failure may result from providing the wrong information in the login screen, hence not being able to access the enrollment operation.

3. System Architectural Design

**3.1 Architectural Pattern Description**

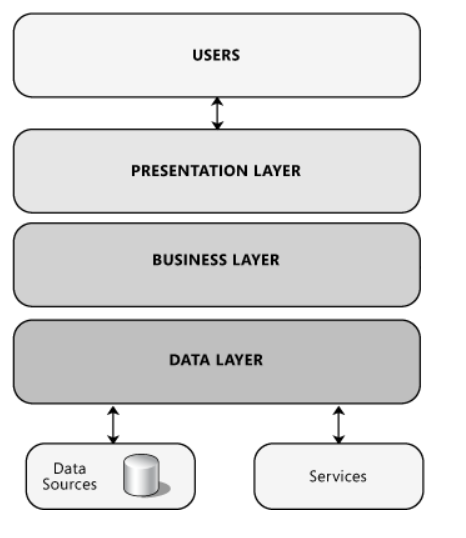
The pattern used in this application is the Layered Architectural Pattern.

This pattern consists of decomposing the design of the application into logical groupings called layers. These layers make it easier to differentiate between the tasks performed by the components found in each layer.

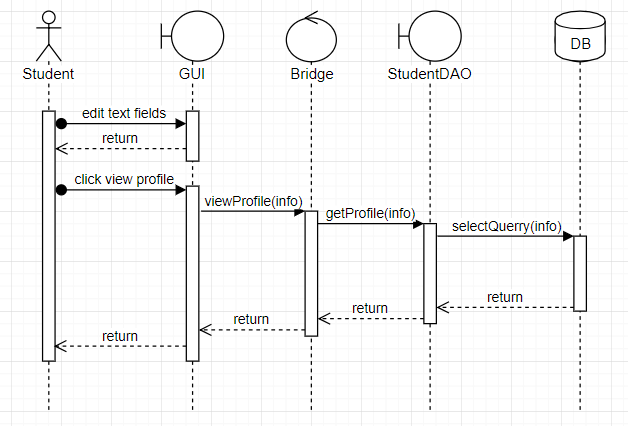
The common three-layer design, which will be used in the implementation of this application, consists of the following layers:

* **Presentation Layer**
  + contains the GUI responsible with the interaction between the user and the system
  + it is the bridge between the operations implemented in the Business Logic Layer and the user
* **Business Logic Layer**
  + in this layer the core functionality of the system is implemented, also known as business logic
* **Data Access Layer**
  + this layer provides access to the data required by the system, either from the database of the system or obtained from other services

**3.2 Diagrams**



4. UML Sequence Diagrams



Sequence diagram for the viewing of a profile for a Student type of user.

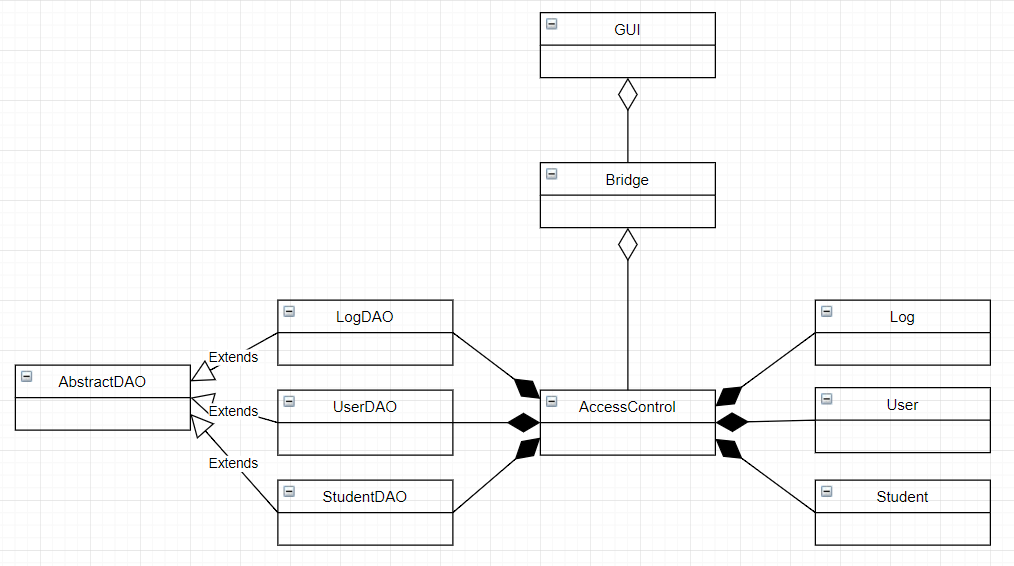
5. Class Design

**5.1 Design Patterns Description**

The design pattern used is called Design by contract. By using this design pattern the programmer specifies preconditions, postconditions and invariants for the operations that are contained in the Business Logic Layer.

**5.2 UML Class Diagram**

A final UML Class Diagram will be developed after the implementation of the project.



6. Data Model

There are three classes that are used in the system’s implementation.

* **User** which contains all the data from the SQL table with the same name :
  + ID
  + Name
  + Age
  + Password
* **Student** which contains:
  + User ID
  + group
* **Log** which contains:
  + Student ID
  + Class
  + Grade

7. System Testing

All the operations of the system will be tested, first with wrong information so as to observe the Design by Contract pattern and then with correct data, so the behavior of the system may be observed.

8. Bibliography

<https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/>

<https://www.oracle.com/technetwork/testcontent/gettingstartedwithusecasemodeling-133857.pdf>

<https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff650706(v=pandp.10)>

<https://github.com/utcn-cs-software-design-tudorv-2019/LabInfo/blob/master/resources/L1_Revision.pdf>

<https://en.wikipedia.org/wiki/Design_by_contract>

<https://creately.com/blog/diagrams/sequence-diagram-tutorial/>