Assignment 1

Analysis and Design Document

Student: Ciucescu Vlad Andrei

**Group: 30432**

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.

# Functional Requirements

The application should have two types of users (student and teacher/administrator

user) which must provide a username and a password to use the application.

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 period containing the activities performed by a student

2. Use-Case Model

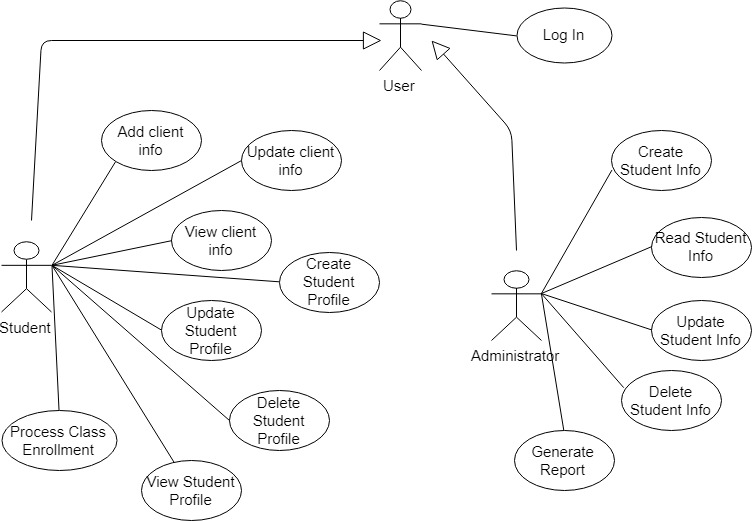
Use case: Update Profile

Level: user-goal level

Primary actor: Student

Main success scenario: The Student presses the “Update Profile” button, edits some fields, presses the “Save Changes” button and the changes to the profile are saved.

Extensions: alternatively, the student could press the “Cancel” button, which cancels any modifications to the profile. If the student introduced invalid data, pressing the “Update Profile” button will not update the profile, but instead will display an error message.



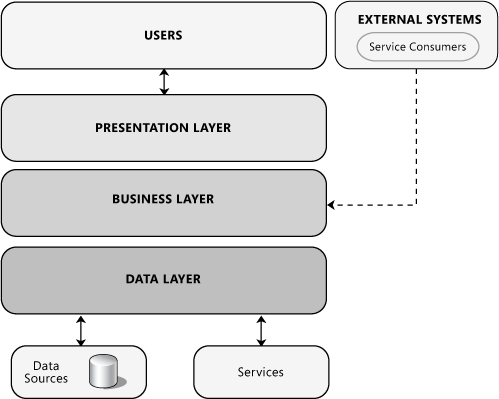
3. System Architectural Design

**3.1 Architectural Pattern Description**

The main architectural pattern used for the application is the layered architecture pattern. Irrespective of the type of application being created, one can decompose the design into logical groupings of software components. These logical groupings are called layers. Layers help to differentiate between the different kinds of tasks performed by the components, making it easier to create a design that supports reusability of components. Each logical layer contains several discrete component types grouped into sub layers, with each sub layer performing a specific type of task.

**3.2 Diagrams**

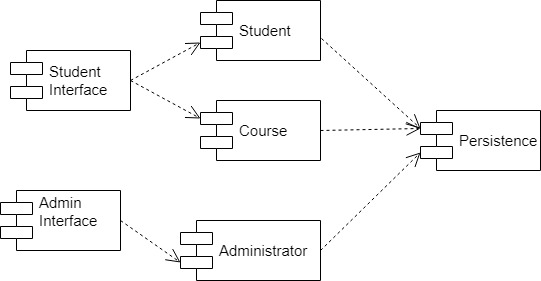
Conceptual Architecture



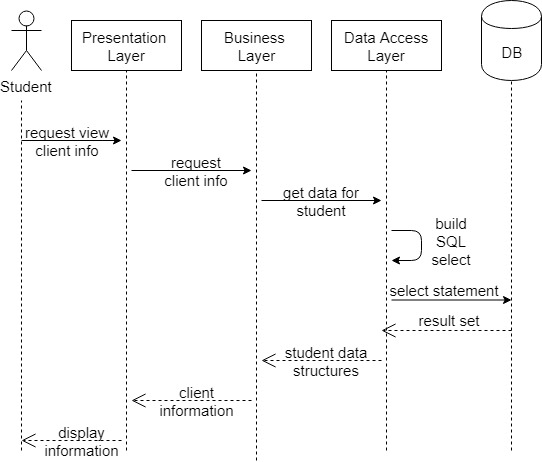
Package Diagram

****

Component Diagram



The is a standalone application, supposed to run on a single computer. As such, no deployment diagram is provided. (deployment diagrams are used for applications that are deployed to several machines)

4. UML Sequence Diagrams

5. Class Design

**5.1 Design Patterns Description**

*[Describe briefly the used design patterns.]*

**5.2 UML Class Diagram**

*[Create the UML Class Diagram and highlight and motivate how the design patterns are used.]*

6. Data Model

*[Present the data models used in the system’s implementation.]*

7. System Testing

*[Present the used testing strategies (unit testing, integration testing, validation testing) and testing methods (data-flow, partitioning, boundary analysis, etc.).]*

8. Bibliography