Assignment A1-student management App

Analysis and Design Document

Student: Razvan Pasca

**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

This application serves as a management tool for teachers in the school department. It comes in the form of a desktop app with a database connection and contains all the functionality needed to reduce the workload and improve the management of the courses and students. The system has different types of users: admins and regular users, each being able to perform different activities.

# Functional Requirements

The system presents different use cases, each coming with its own functional requirements. However, some functional requirements are shared between the use cases:

* Before proceeding with the operations, each user must be logged in
* The password must be longer than 6 characters
* The student can choose to move only in existing groups, attend only existing courses and take only the exams corresponding to the courses he enrolled for
* The student can see only his own grades but can see the name/courses attended etc for the other students.
* The admins of the courses are notified when a student wants to enroll and accept/decline their request

# Non-functional Requirements

*[Discuss the non-functional requirements for the system]*

2. Use-Case Model

*Use case: Student view grades*

*Level: User goal level*

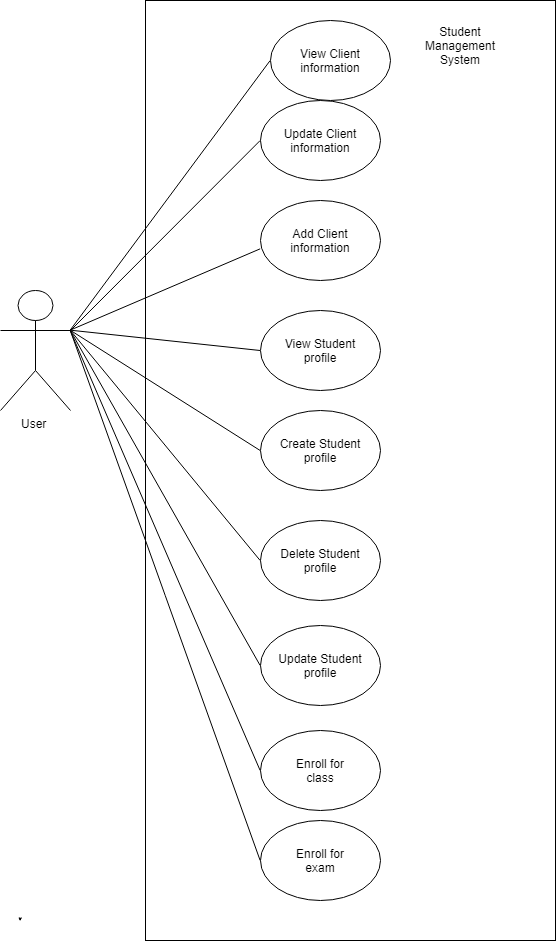
*Primary actor: student*

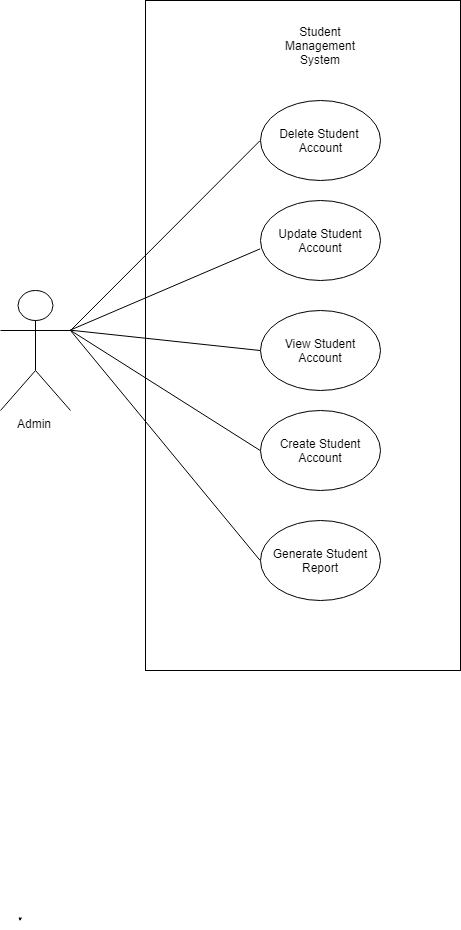
*Main success scenario:*

Pre-conditions: student is logged in

1. The user clicks “View student profile” button
2. A new window shows up with options for the student such as: view grades, view courses etc
3. The user clicks on “View grades” button
4. A list with the student’s grades shows up

*Extensions: 4’. If the student has no grades yet, an appropriate message shows up*





3. System Architectural Design

**3.1 Architectural Pattern Description**

*For this application we are going to employ the layered architectural pattern, in its classical form, namely: Presentation, Business and Data layer. This approach was considered because it the app we are developing is a small one, and there is no need to develop into further components.*

*Moreover, the type of the application is the classical use of the layered pattern, since it implies:*

* *Connection with the database*
* *DAO components for the model classes*
* *Business logic to implement the functionalities*
* *Presentation layer for the GUI*

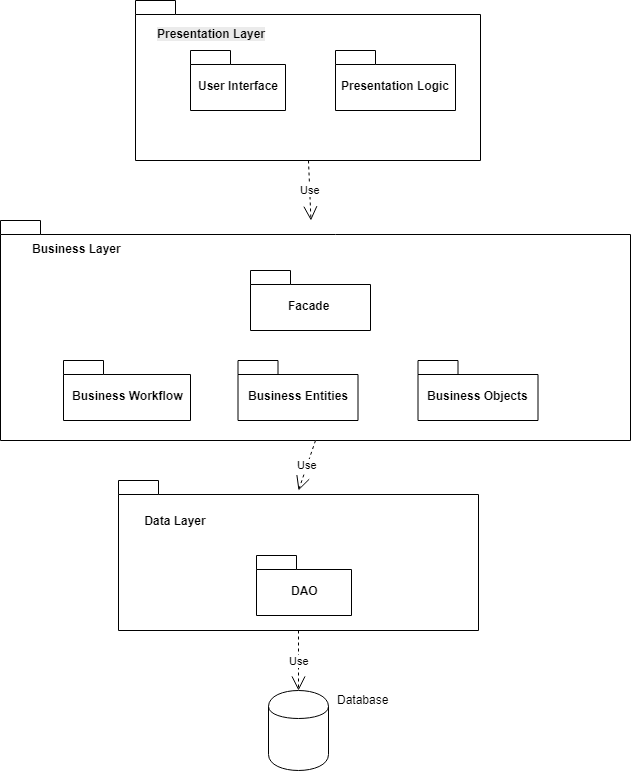
*Thus, we can see that it is easy to develop a logical separation between the layers based on their responsibilities and functionalities. This enables a lot of flexibility for the future, in case we opt to offer a web or mobile version of the system, since theoretically we would have to modify just the presentation layer.*

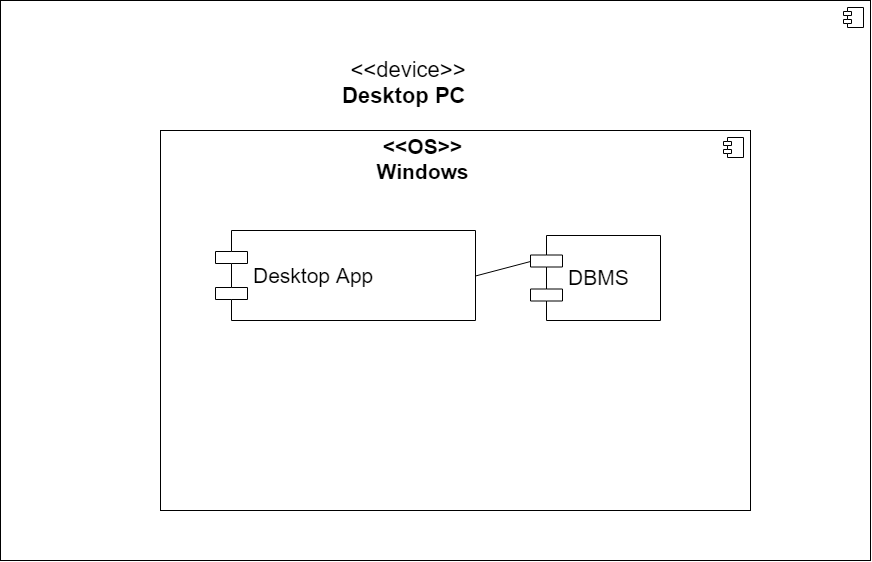
*Another pattern which we are going to employ is the MVC pattern. This is a regular solution when developing user interfaces, since it splits the application logic into three separate parts. This increases the modularity of the system and gives us flexibility when it comes to extending the app and the user base.*

**3.2 Diagrams**

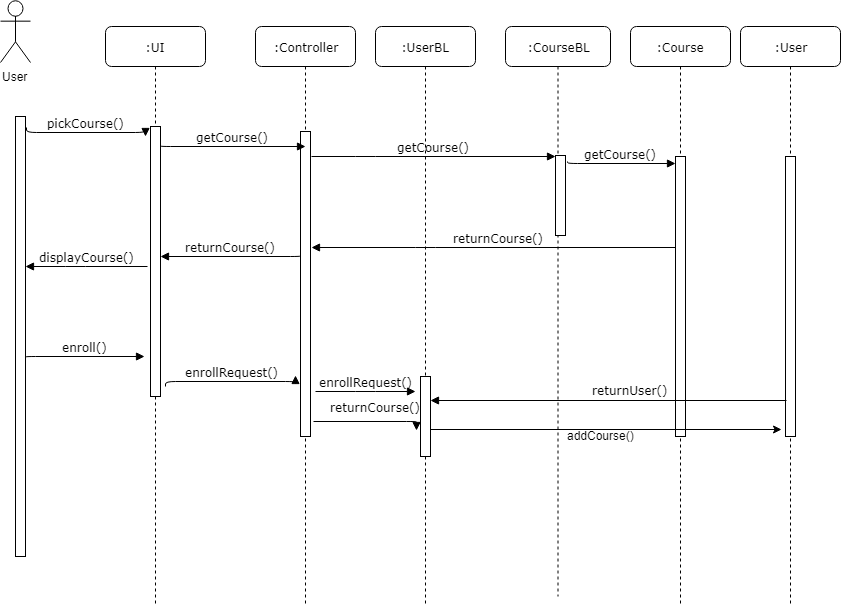
*For the layered architecture, we are going to employ the classical package division of the components. Moreover, in the initial phase, the whole system is going to be placed on one component. At further iterations, we can employ a multi-tier architecture, where the database and the BLL are deployed on one component and the desktop app is delivered independently.*

*For the MVC, we can also employ different implementations of the view component, depending on the type of the host device: desktop or mobile. Talking about the dependencies, the controller is responsible for handling the interactions with the business layer through the façade provided.*

**

**

4. UML Sequence Diagrams



5. Class Design

**5.1 Design Patterns Description**

*Façade Design Pattern: this pattern has been employed to simplify the interface provided by the business logic. In this way, the interaction with the system is simplified, and reduces the outside dependencies to the BLL.*

*Observer Design Pattern: this pattern has been employed to notify the admins about any changes in the courses they teach (e.g. a student wants to enroll). Also, the students are notified about changes in the courses they are enrolled in (e.g. an exam is anounced or the date is changed).*

**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

1. https://en.wikipedia.org/wiki/Facade\_pattern#UML\_class\_and\_sequence\_diagram

2. <https://msdn.microsoft.com/en-us/library/ee658109.aspx>

3.