UTCN Student Management app v2

Student: Turcu Lucian 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

The application’s role is to keep track of the students’ records. A regular user can add/update/view student 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 can perform CRUD operations on students’ information. Generate reports for a particular period containing the activities performed by a student.

# Functional Requirements

A regular user (aka the student) should be able to add/update/view student 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 should be able to perform CRUD operations on students’ information. Generate reports for a particular period containing the activities performed by a student.

# Non-functional Requirements

The application must use SQL as a DB engine and have an initial data set created through an SQL script.

Implementation must be done using the Layers architectural pattern. And using sp

2. Use-Case Model

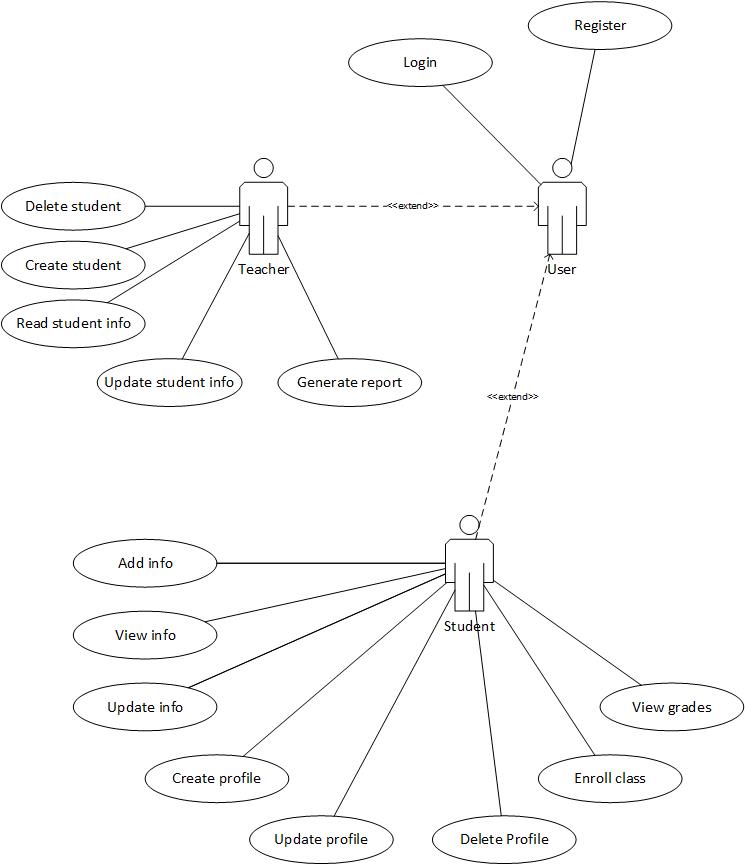
Use case: Enroll

Level: User goal

Primary actor: Student

Main success scenario: Push enroll button, input course id, push submit.

Extensions: Student not eligible for course, fail.

**

*Use case diagram*

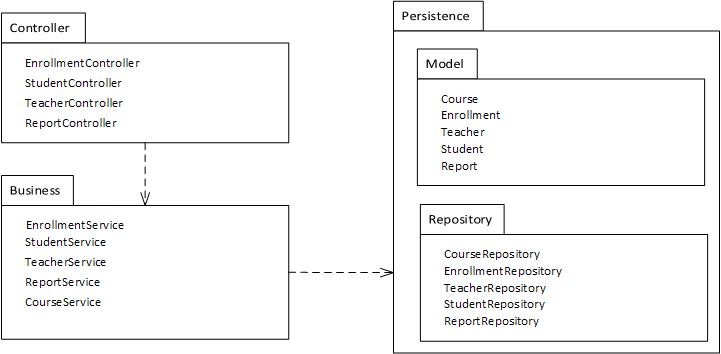
3. System Architectural Design

**3.1 Architectural Pattern Description**

Layered pattern: it is used to decompose programs into groups of subtasks, each of which is considered a layer and provides services to the next higher one. The layers used are the application layer (aka service layer), business logic layer (aka controller layer) and data access layer (aka the repository layer).

**3.2 Diagrams**

*Package and component diagrams*

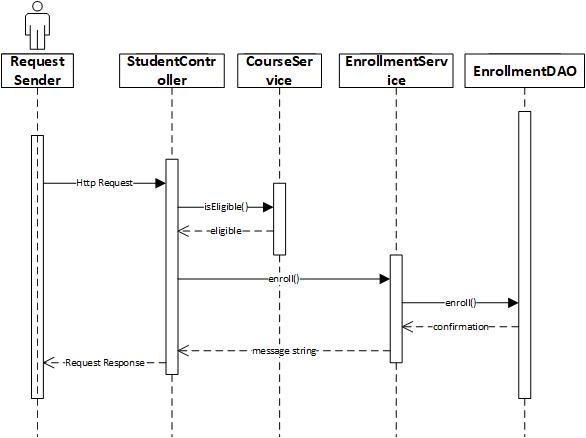
**

*<package diagram>*

The package diagram illustrates the implementation based on the Layers architectural pattern. The dependencies exist between adjacent layers. In this case, Controller > Service > Repository, while model is accessed from everywhere.

4. UML Sequence Diagrams

*Use case diagram for course enrollment.*

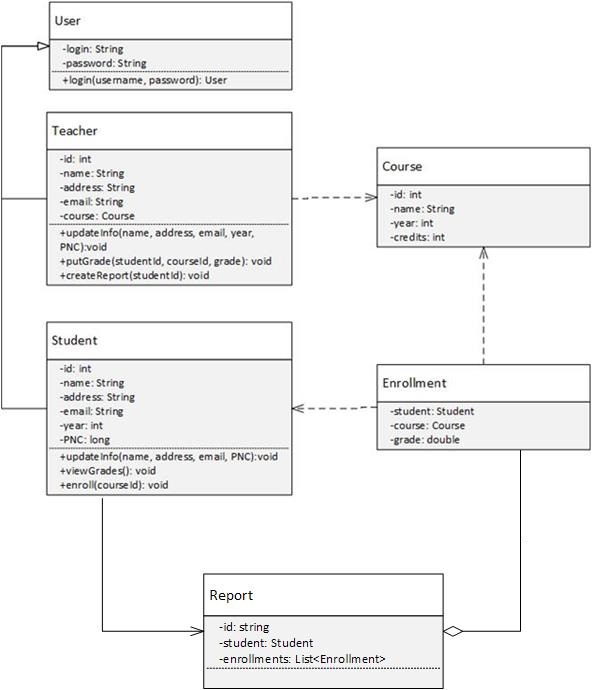


5. Class Design

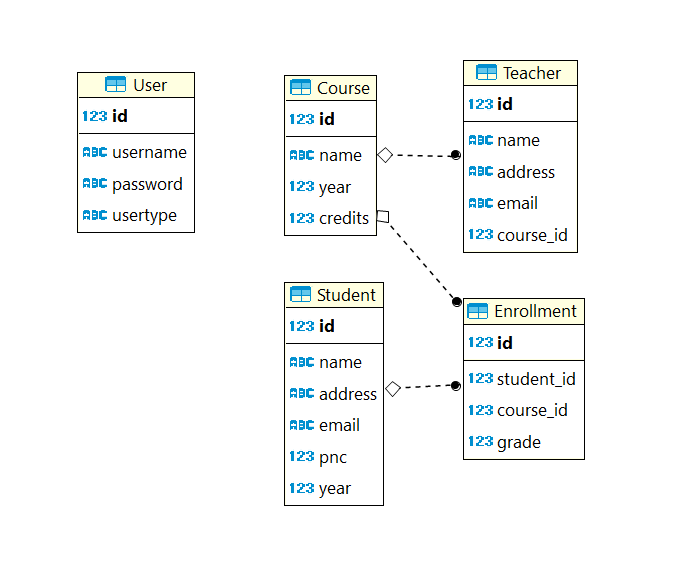
**5.1 Design Patterns Description**

The application will implement the singleton design pattern for the database connection. This will ensure that only a single connection object is created and reused for all database access events.

**5.2 UML Class Diagram**

**

6. Data Model



7. System Testing

Testing will be done using Junit tests for most of the use cases. Additional testing will be done during the development phase for critical functionalities of the system.

8. Bibliography

<https://msdn.microsoft.com/en-us/library/ff650706.aspx>

<http://stackoverflow.com>