# Assignment 1 – Student Management System
# Analysis and Design Document

**Student: Ana-Maria Nanes**
**Group: 30432**

# Table of Contents

# 1. Requirements Analysis

## 1.1 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.

## 1.2 Functional Requirements

The functional requirements of the system are the following:

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

## 1.3 Non-functional Requirements

The non-functional requirements :

- Security - making distinction between the two types of users – students and administrators – the data is protected.

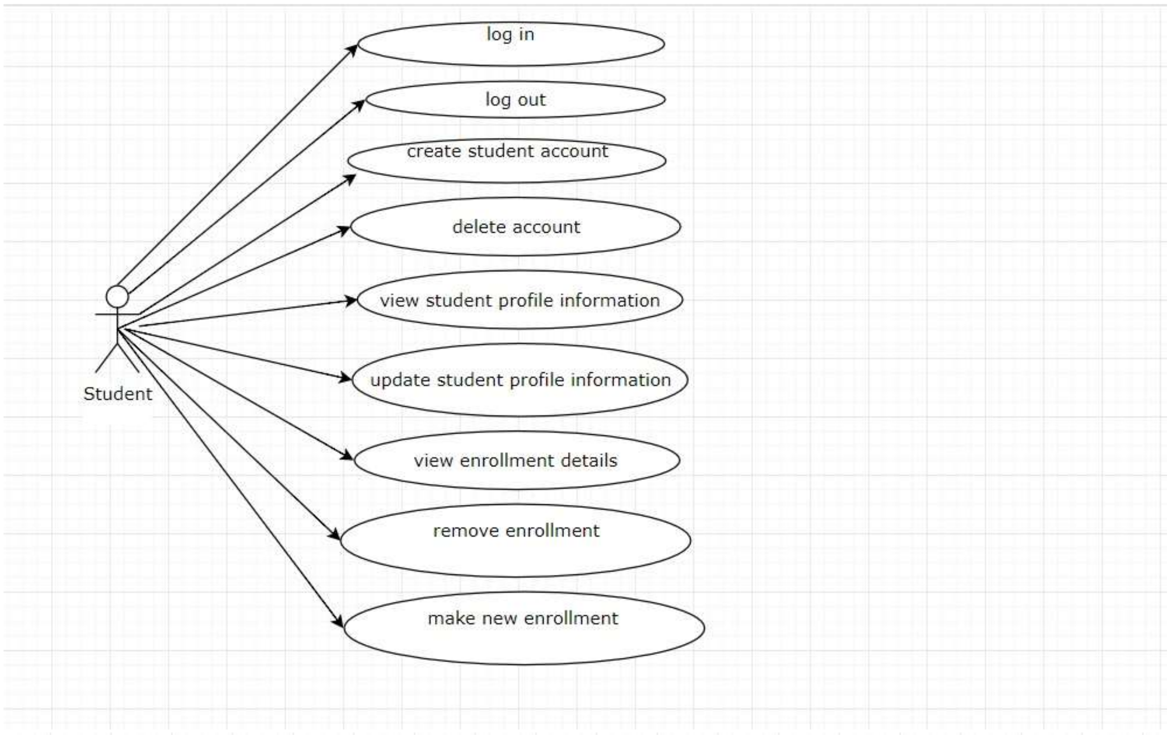
- Data integrity – is implemented using data validation each time new information is added to the system.

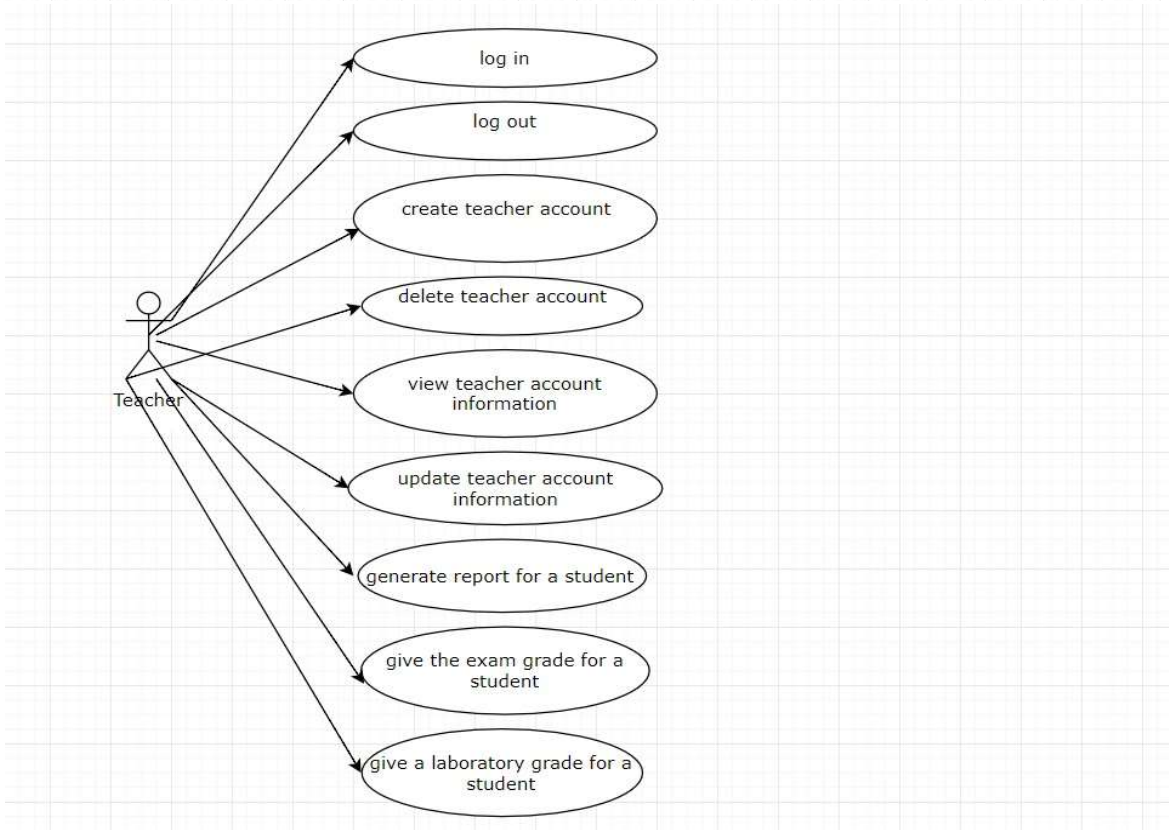
- Extensibility - due to the Layers Architectural Pattern it is easy to extend the system in more directions.

- BackUp - the system`s information could be also stored in files – make a backup after a certain period of time.

## 2. Use-Case Model

The following diagram is the use-case diagram:




log in
log out
create teacher account
delete teacher account
view teacher account information
update teacher account information
generate report for a student
give the exam grade for a student
give a laboratory grade for a student
Teacher

The two type of users are:

- The student.
- The administrator.

Use case: Create student profile.
Level: Subfunction.
Primary actor: A student in the CS Department at TUCN.
Main success scenario:

- Provide a user name.
- Provide a password.
- Provide the required information.
- Validate the input data.
- Perform the registration operation.

Extensions:

Scenarios of failure :

- the username is already taken.
- the name is already taken.
- the password does not respect the imposed structure.

# 3. System Architectural Design

## 3.1 Architectural Pattern Description

The architectural pattern used are the following:
Layered Architectural Pattern

Layered Architectural Pattern

This pattern can be used in order to structure the application such that it can be decomposed into groups of subtasks. Each layer implements subtasks at a particular level of abstraction and each layer provides services to the next higher level.

The following 3 layers will be used in order to offer the application the accurate structure:

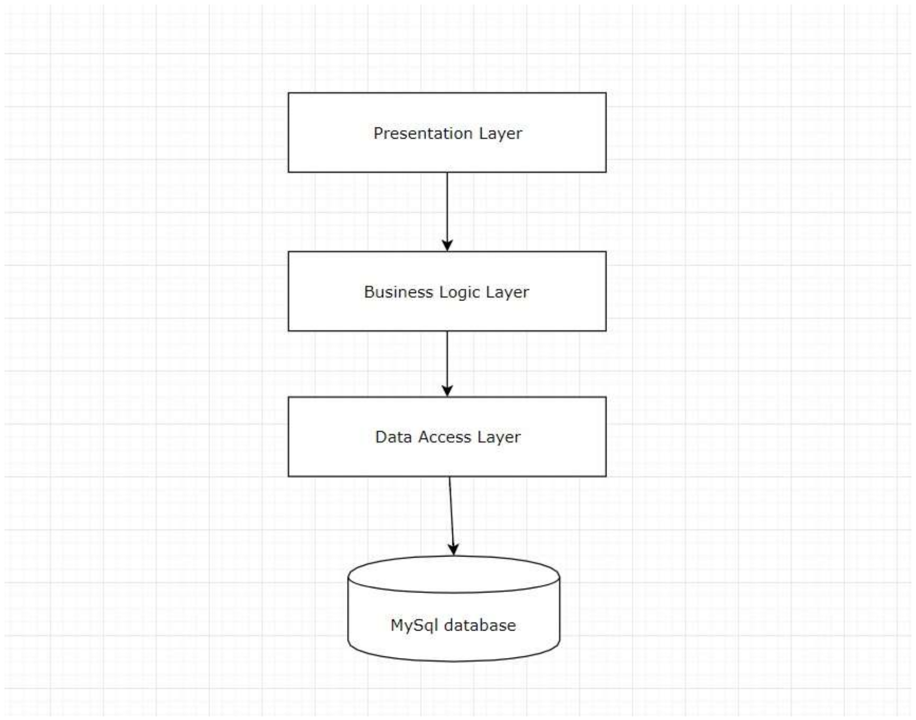
- Presentation Layer
- Business Logic Layer
- Data Access Layer

These packages are modeled using different packages inside the application.

## 3.2 Diagrams

- The system conceptual diagram:

The layers are depicted in the next system conceptual diagram:



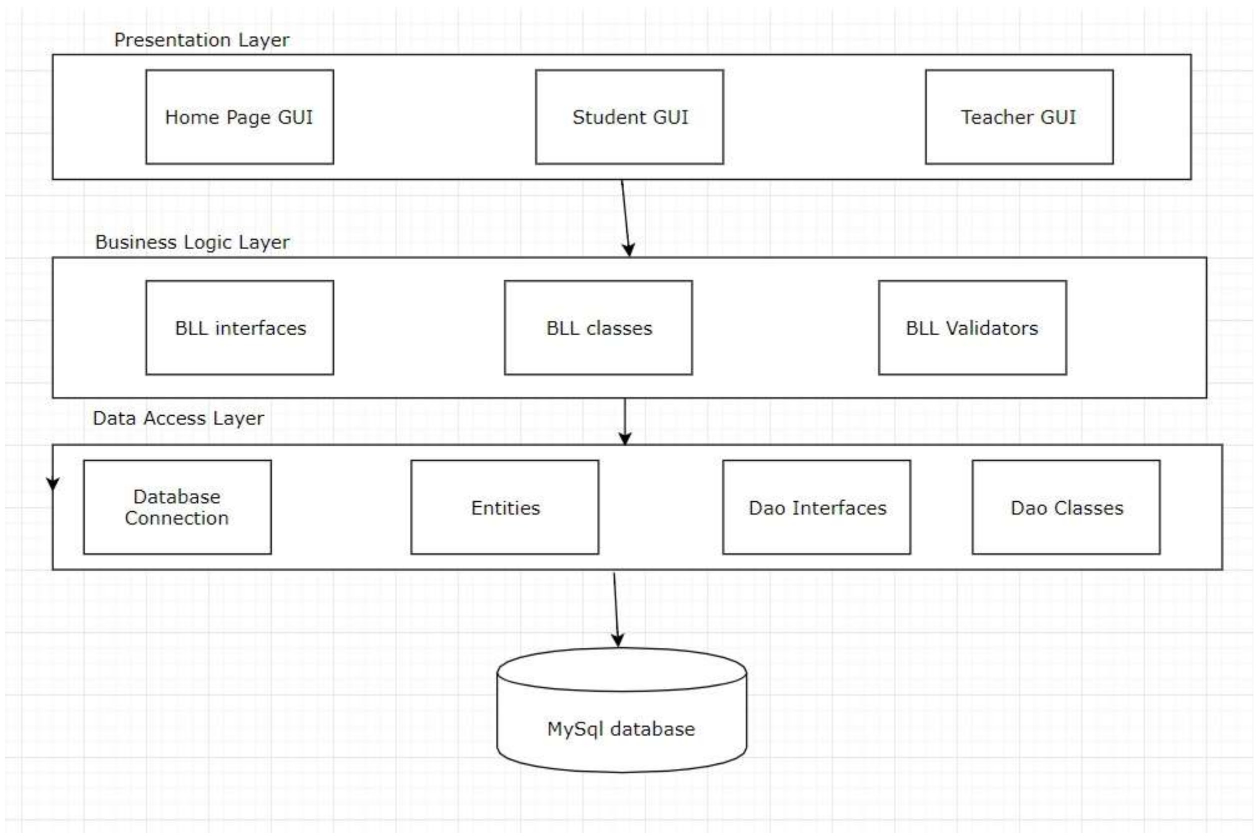
How the Layered Architectural Pattern is applied:

The Data Access Layer provides acces to the data hosted within the boundaries of the system, and data exposed by other networked systems; perhaps accessed through services.

The Business Layer implements the core functionality of the system, and encapsulates the relevant business logic. It consists of components which expose service interfaces other callers can use.

The Presentation Layer contains the user oriented functionality responsible for managing user interaction with the system,and generally consists of components that provide a bridge into the core business logic encapsulated in the business layer.

A more detailed overview is the following, in which we can see the classes that can be found in each layer:



Classes in the Presentation Layer:

- Home Page View classes
- Student View classes
- Teacher View classes

Classes in the Business Logic Layer:

- Business Logic Interfaces
- Business Logic implementations of the interfaces
- Business Logic data validators

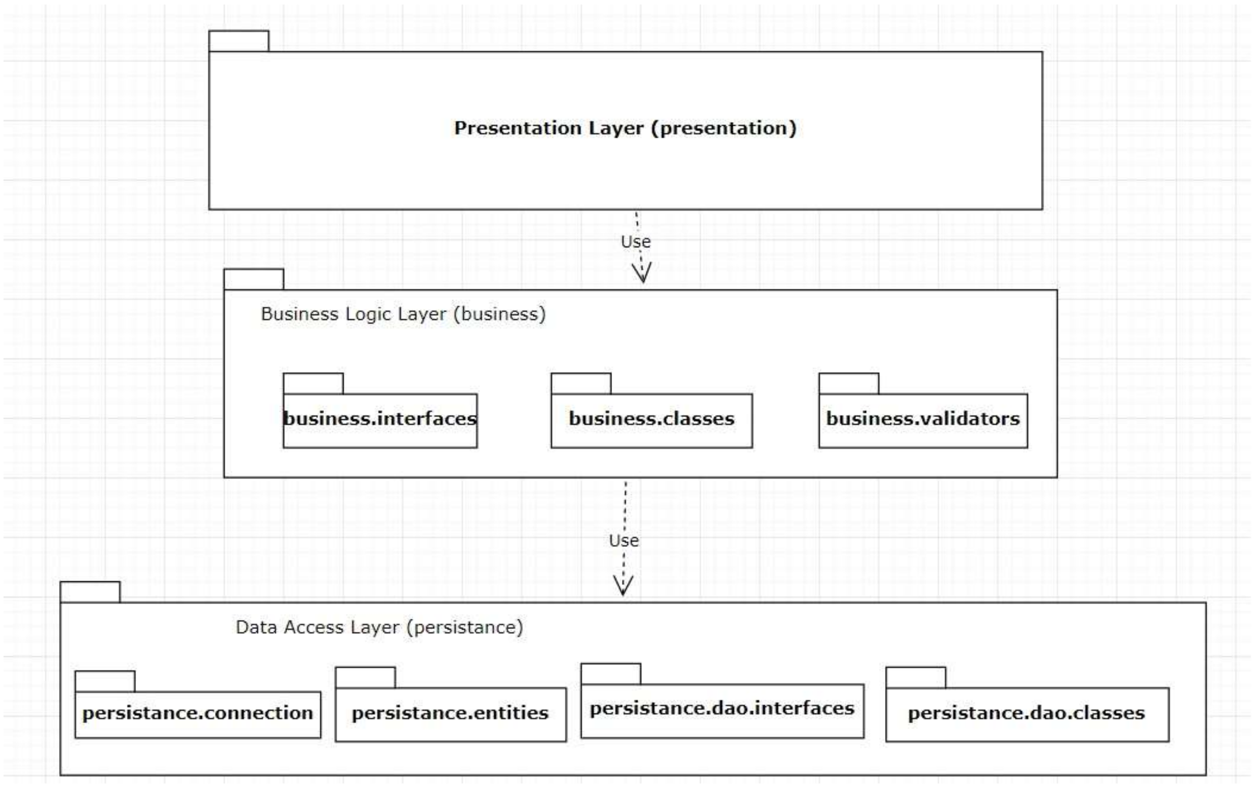
Classes in the Persistance Layer:

- Database connection class
- Entities
- Data Access Interfaces
- Data Access Classes

❖ The package diagram:

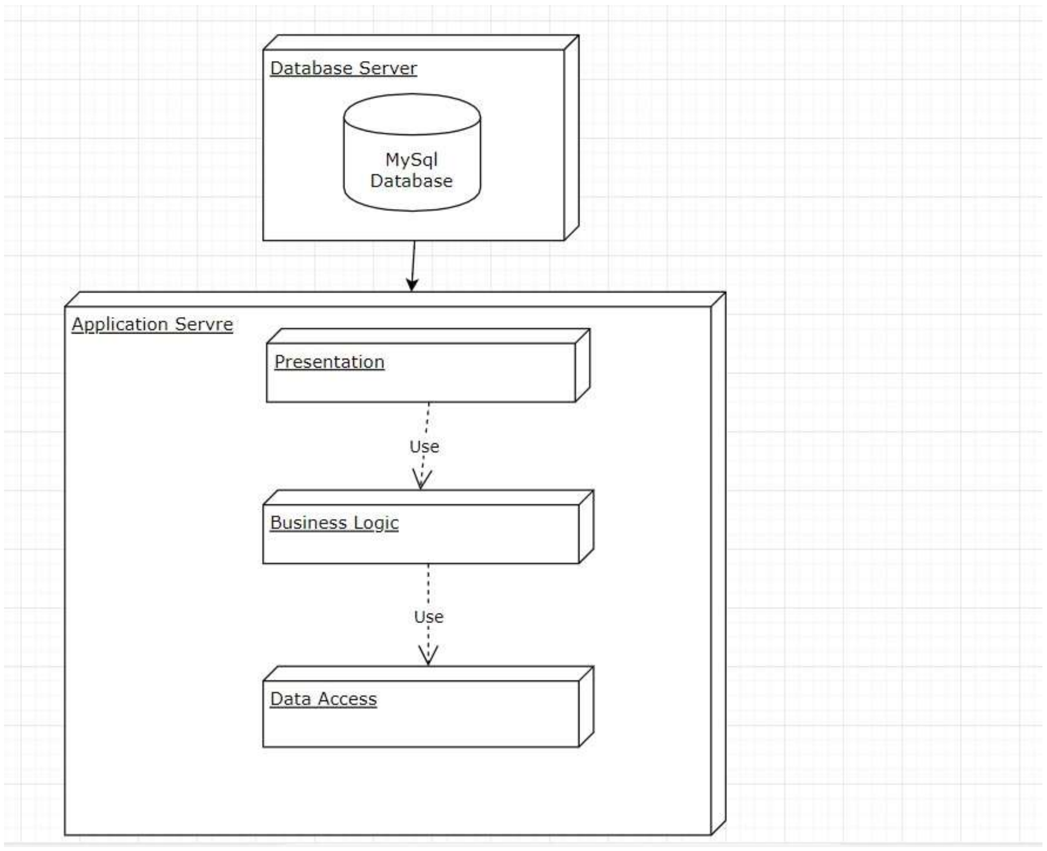
The Layered achitectural pattern is implemented using a separation of the application classes into packages according to their functionalities.
In the application we have the following packages:

- presentation
- business with the subpackages
    - business.intrefaces
    - business.classes
    - busisness.validators
- persistance with the subpackages:
    - persistance.dao.interfaces
    - persistance.dao.classes
    - persistance.entities
    - persistance.connection



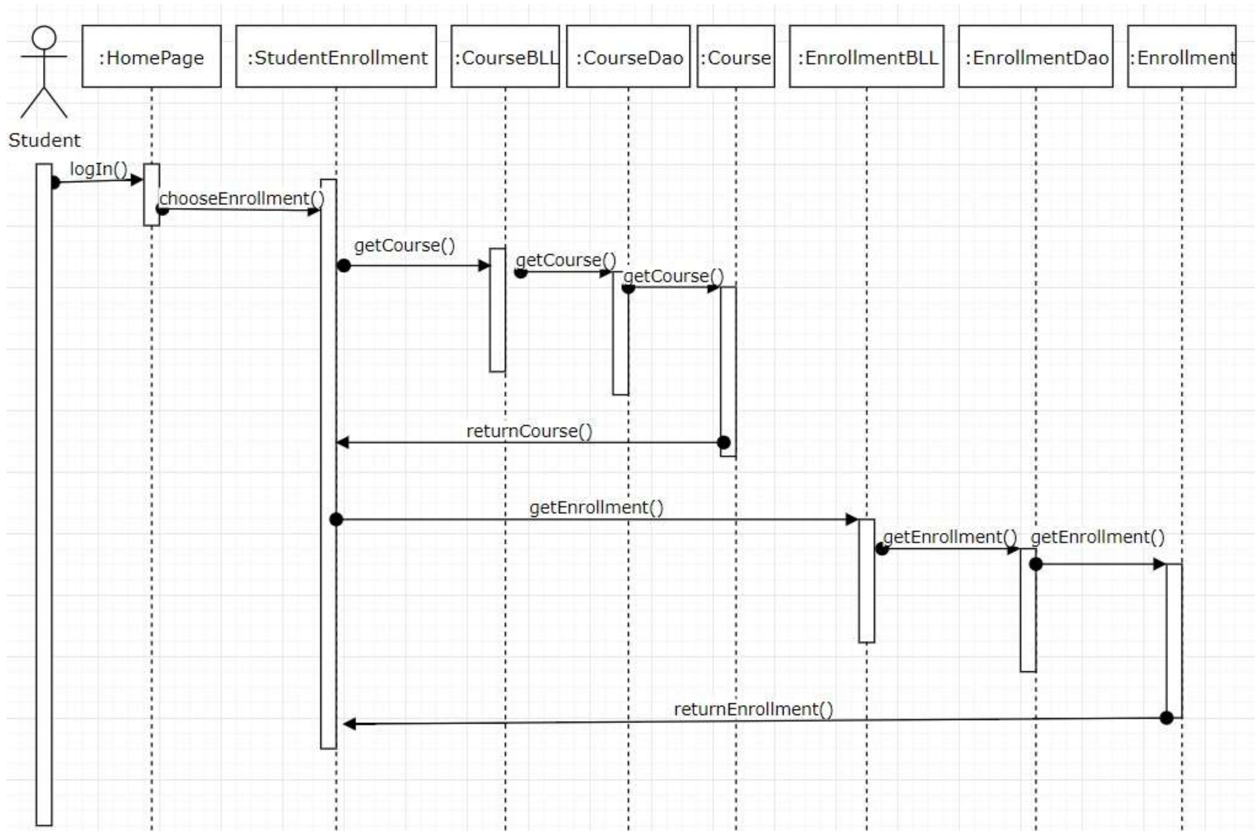
❖ The deployment diagram:

The deployment diagram is a structure diagram through which we see the architecture of our application distributed through artifacts. The artifacts represent physical elements from the real world that are a result of a development process. In our application, we only have 2 components, i.e the Database Server and the app itself, which is a desktop application.
The principal elements of the  deployment diagram are the nodes.



## 4. UML Sequence Diagrams

A relevant scenario is the one in which a student views the course and the enrollment details for a chosen course:



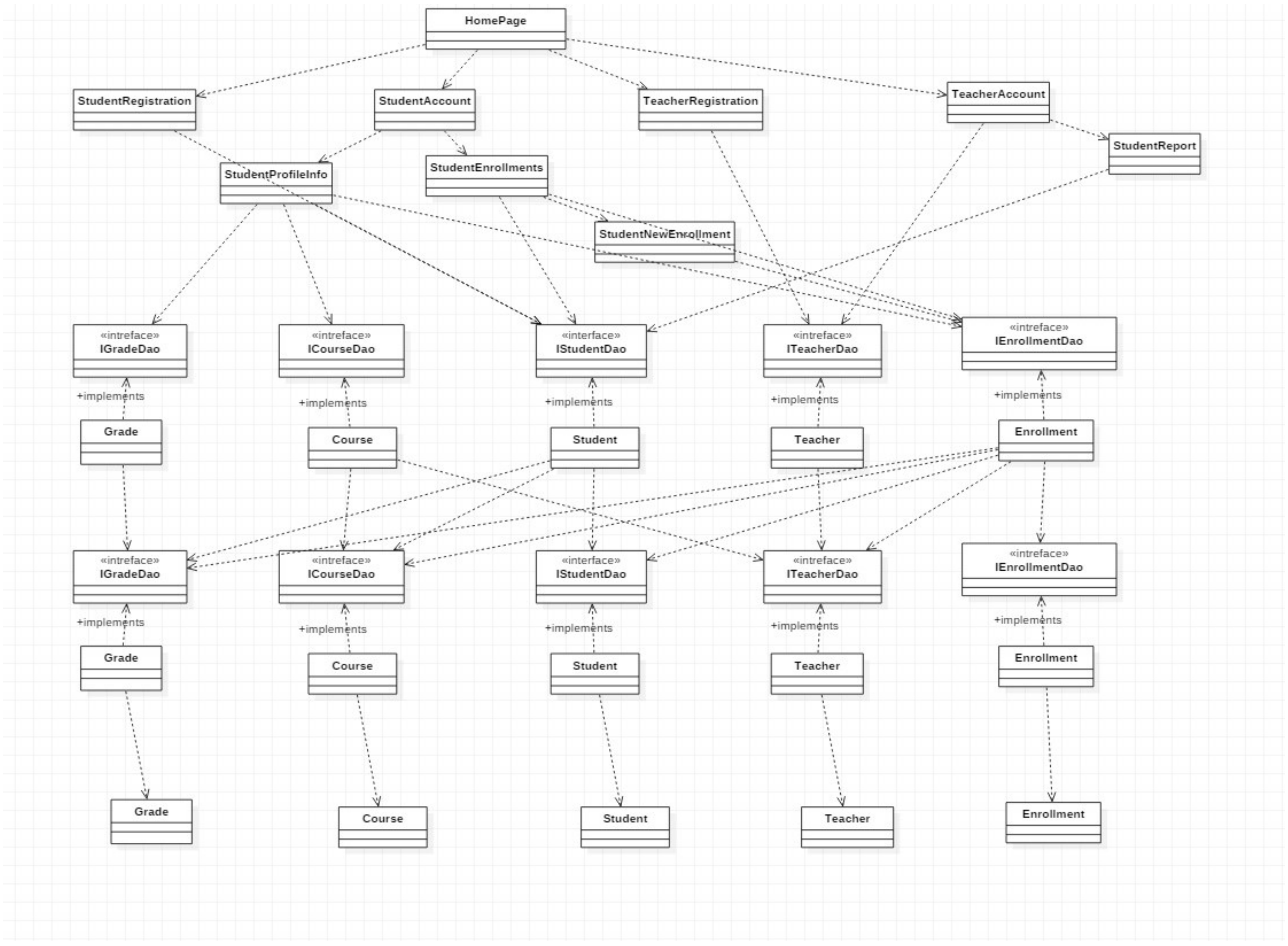
# 5. Class Design

## 5.1 Design Patterns Description

I have chosen to use the Singleton Design Pattern: in order to model the fact that in the entire system only a single instance of the class DatabaseConnection should be created.

In order to achieve this we use a private constructor and a methos getInstance that returns the one and only instance of the class if it was created or which creates it if it had not been previously created.

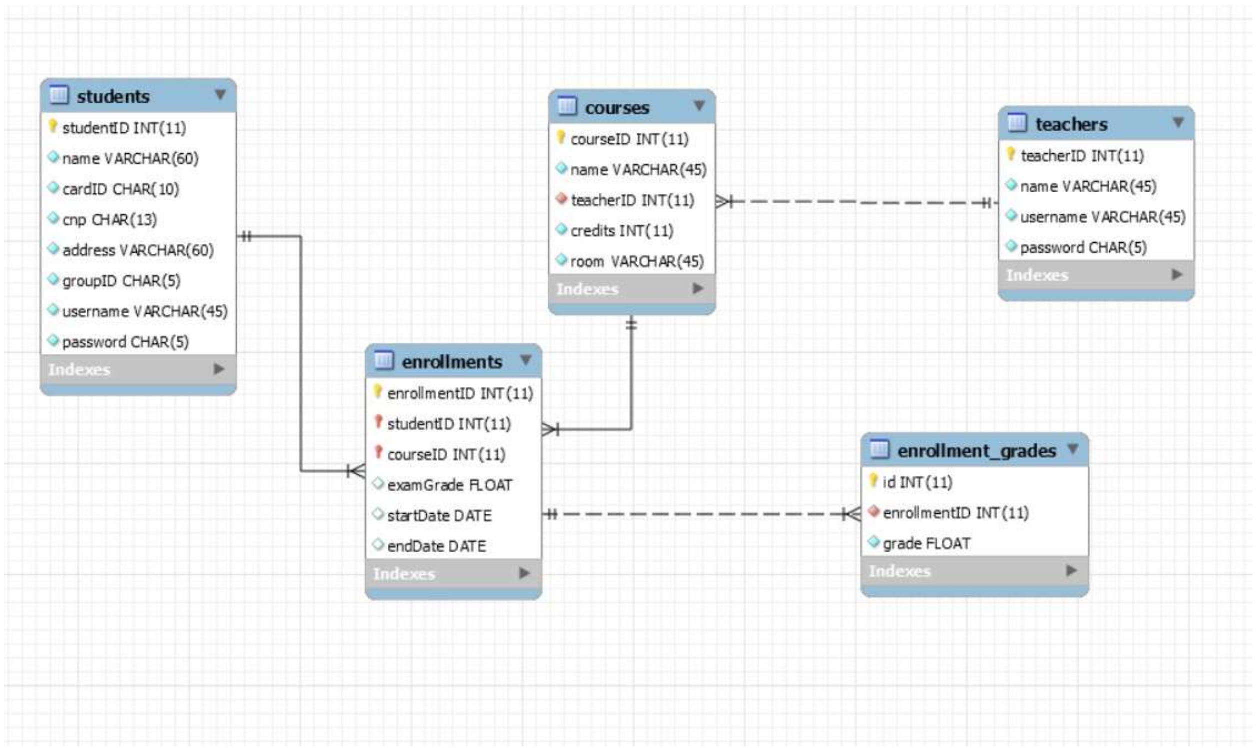
## 5.2 UML Class Diagram



## 6. Data Model

I have chosen to represent the data model used in the system`s implementation using the Relational Database Model. In this way both the data, and the exact relationships between it are clearly specified.

The database will be created in MySql.



## 7. System Testing

➢ The used testing strategies are the following:

<u>Unit Testing</u>

By unit testing the developer writes a piece of code that executes a specific functionality in the code to be tested ans asserts a certain behavior or state.

I choose to use Unit Testing to test methods that implement the CRUD operations on student information.

Individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.

When we combine different units that were separately tested (unit testing) we test if the units can be correctly combined in order to obtain the entire architecture of the system.

In perform Junit testing by testing some of the DAO classes.We create the test classes StudentDaoTest and TeacherDaoTest in order to test some of the methods these classes that implement some intefaces, provide. We use assertions to check if the provided result after is the expected one.

## 8. Bibliography

http://softwaretestingfundamentals.com/unit-testing/
https://en.wikipedia.org/wiki/Software_verification_and_validation
https://www.safaribooksonline.com/library/view/software-architecture-patterns/9781491971437/ch01.html
https://en.wikipedia.org/wiki/Multitier_architecture