Assignment A2
Analysis and Design Document

Student: Nicolae-Florian Onica
Group: 30432

Table of Contents

1. Requirements Analysis

1.1 Assignment Specification

The assignment consists of a Java application for the management of students in the CS Department at TUCN. The application will have two types of users (student and teacher/administrator user).

1.2 Functional Requirements

The application should allow the following user to do the following operations:
For the regular user:
- 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).

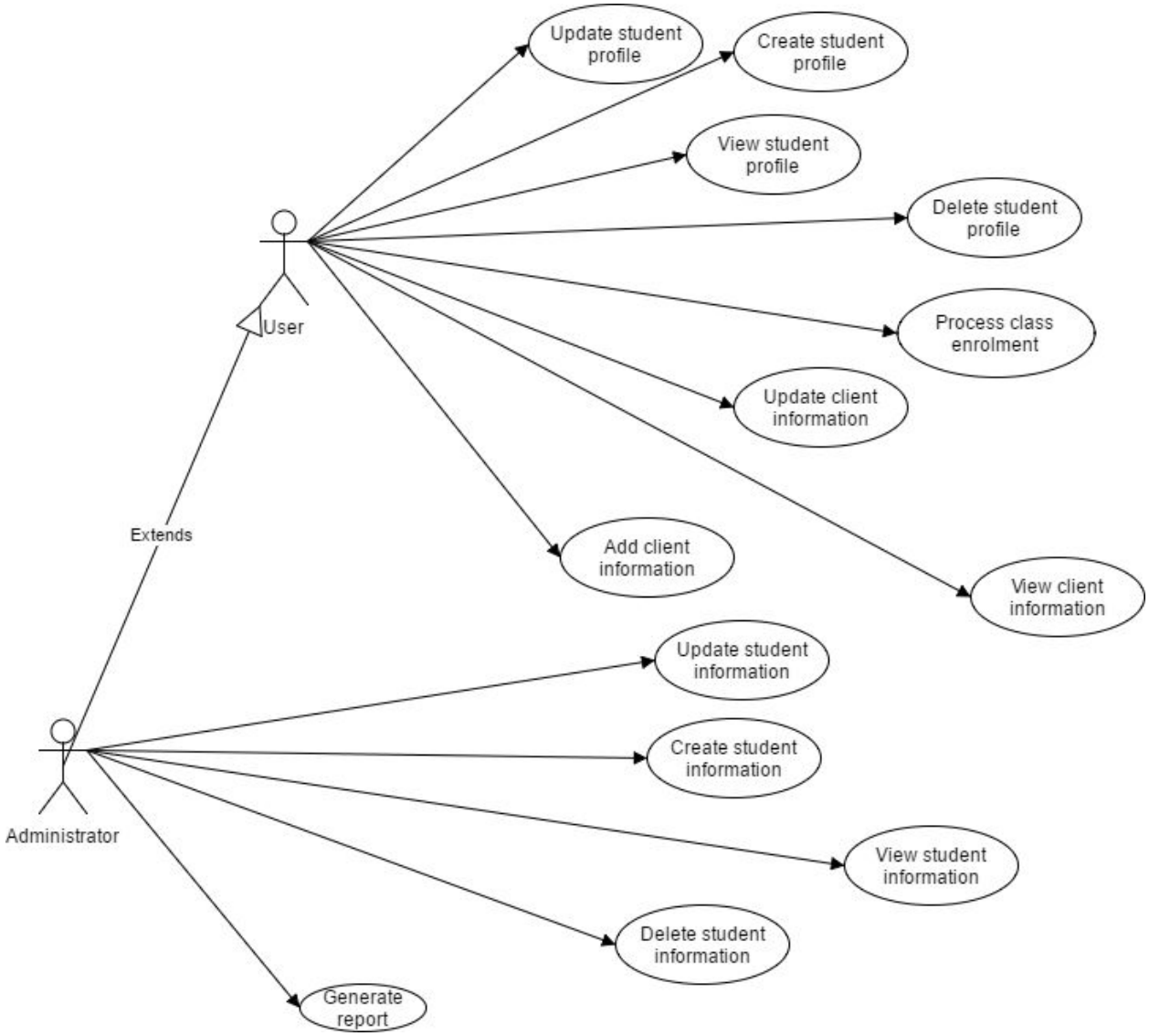
For the administrator:
- CRUD on students information.
- Generate reports for a particular period containing the activities performed by a student.

1.3 Non-functional Requirements

2. Use-Case Model

Add_student: the student is added in the database
Level: user-goal level
Primary actor: teacher
Main success scenario: the teacher inputs correct student data, is logged in and the student is inserted into the database
Extensions: Failure case: the teacher inputs invalid data and is redirected to the add student form

Use case diagram:



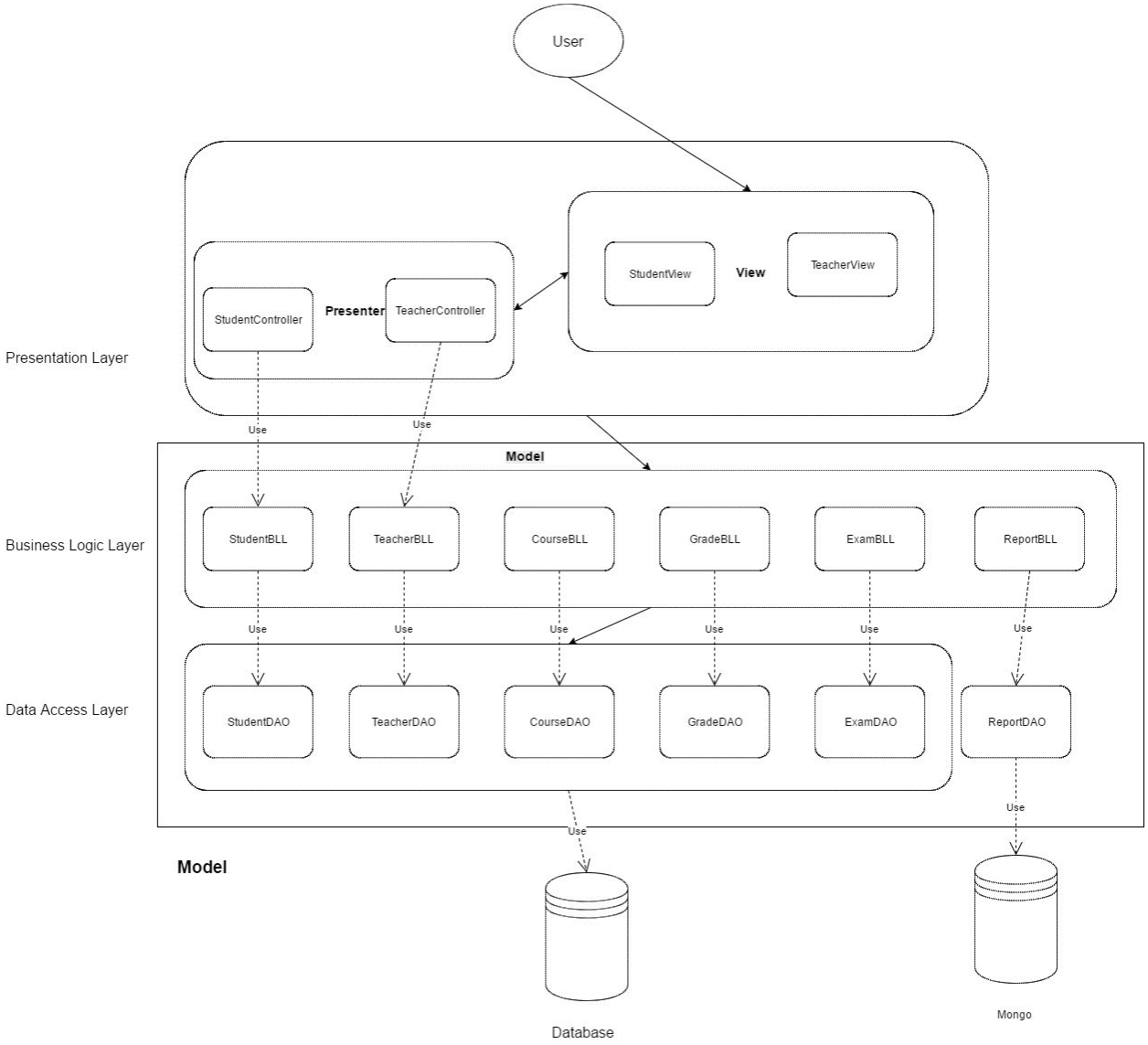
3. System Architectural Design

3.1 Architectural Pattern Description

The architectural pattern used for this application is the layered architecture pattern. The components within this pattern are organized into horizontal layers, each layer performing a specific role within the application (e.g., presentation logic or business logic). In this case three layers will be used: presentation, business and database layer.

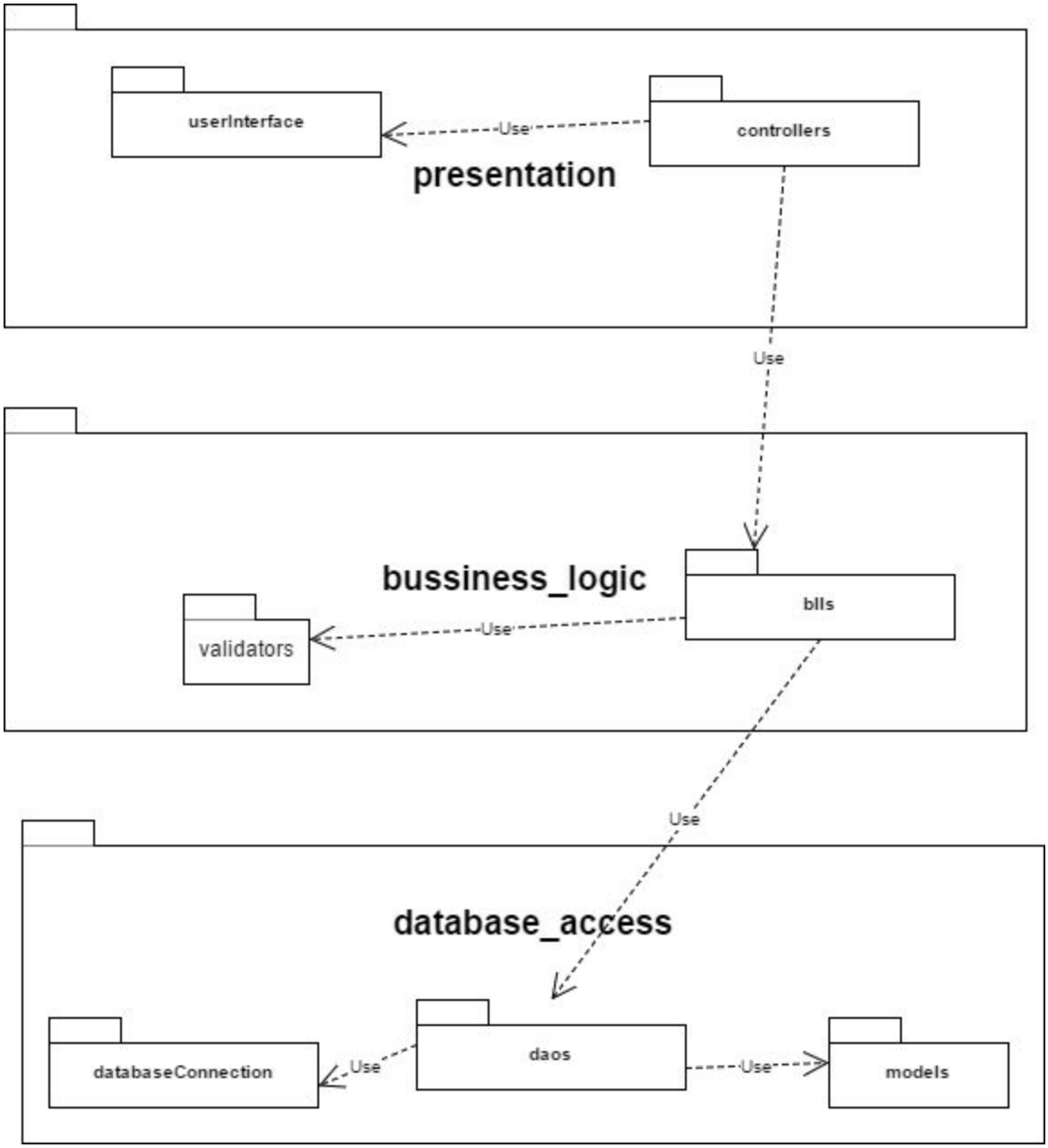
MVP pattern

This architecture is a variation of the MVC pattern used to facilitate automated unit testing and improve the separation of distinct sections in the presentation layer. The model here is an interface defining the data which will be displayed or modified in the user interface. The view is a passive interface that displays the model and directs user commands to the presenter to act upon that data. The presenter acts upon the model and the view. It retrieves data from repositories, and formats it for display in the view.

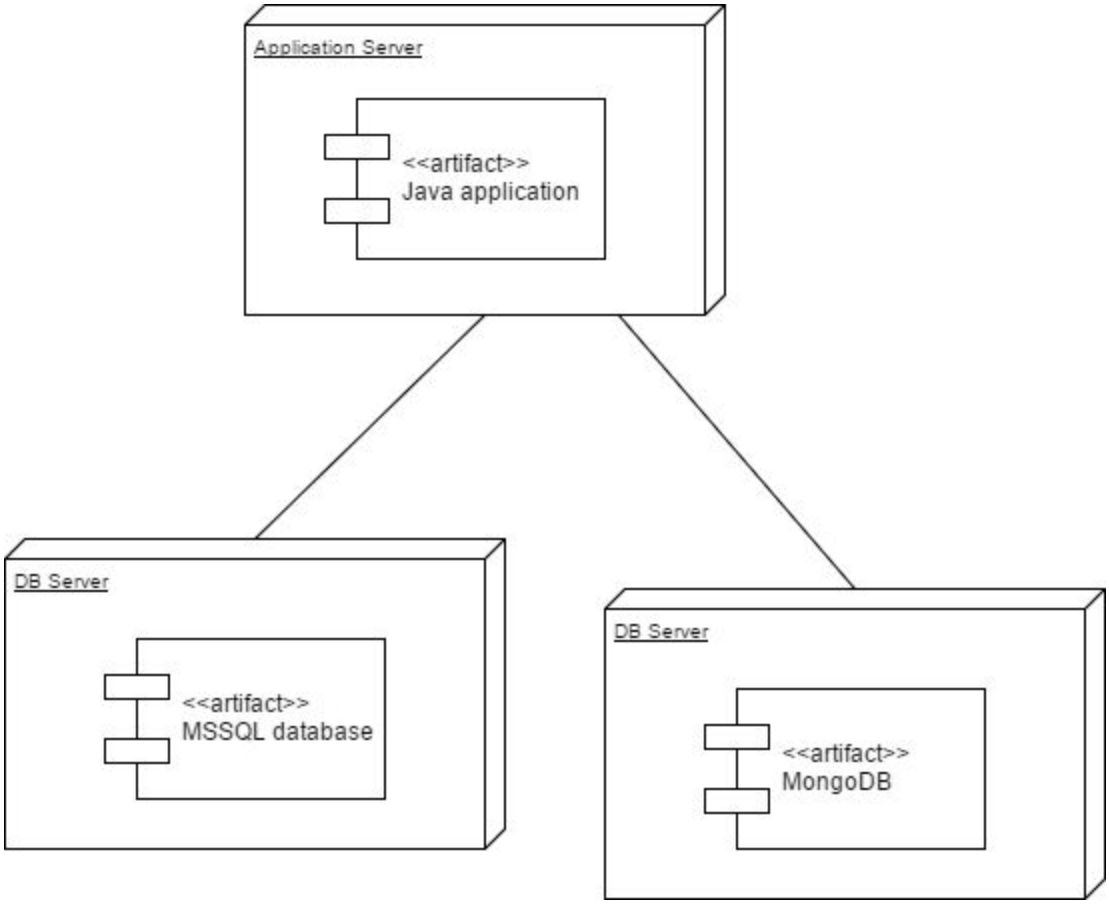


3.2 Diagrams

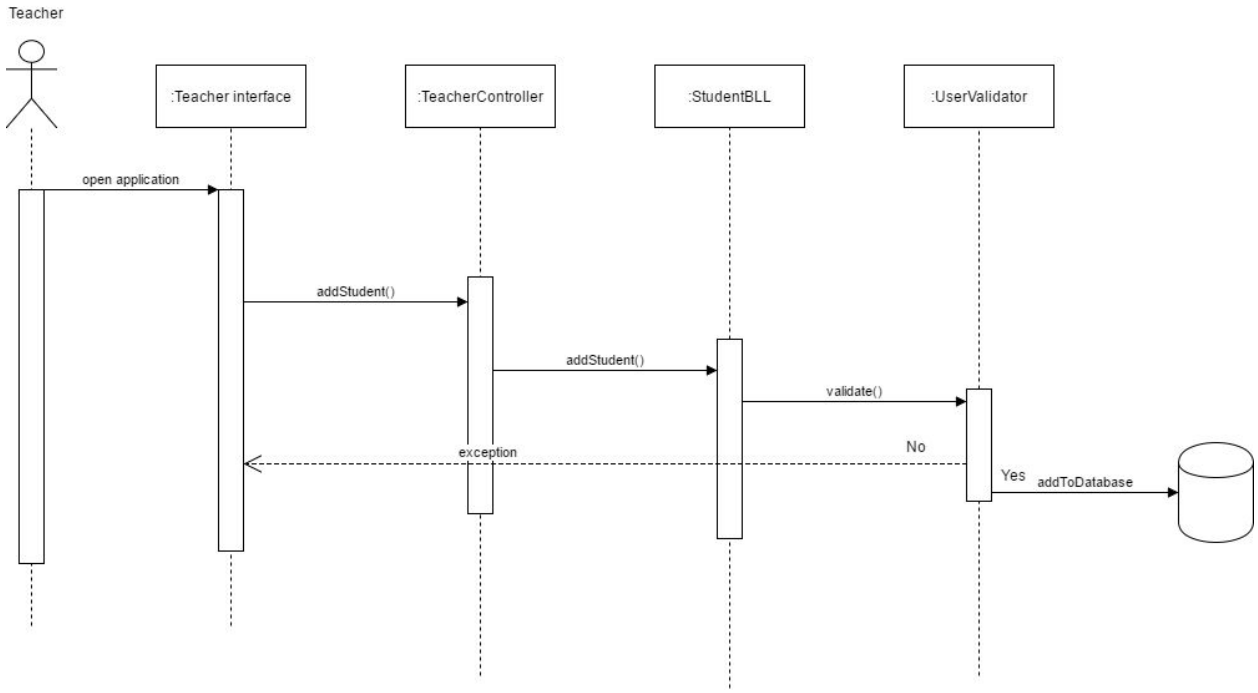
Package diagram



Deployment diagram



4. UML Sequence Diagrams

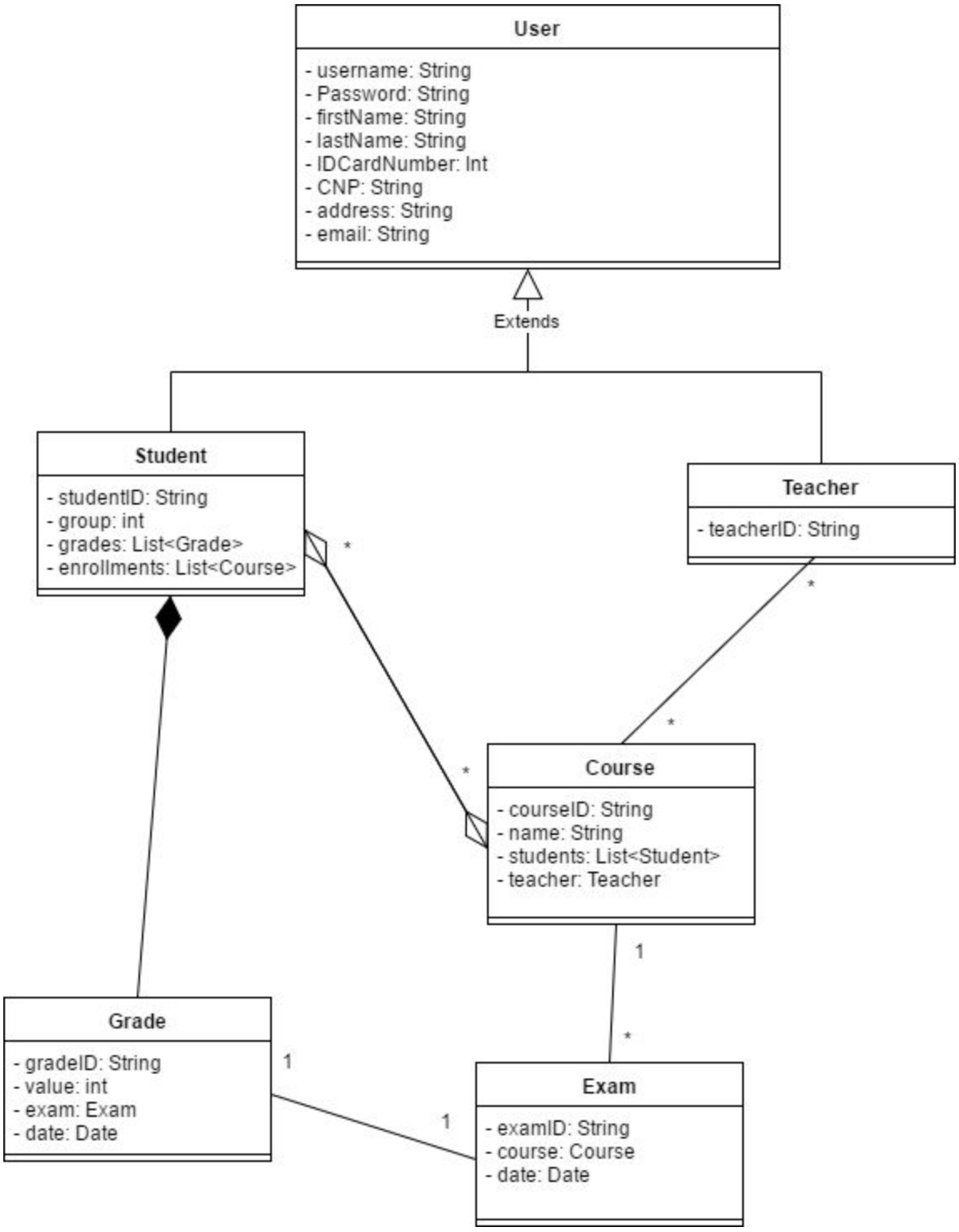


5. Class Design

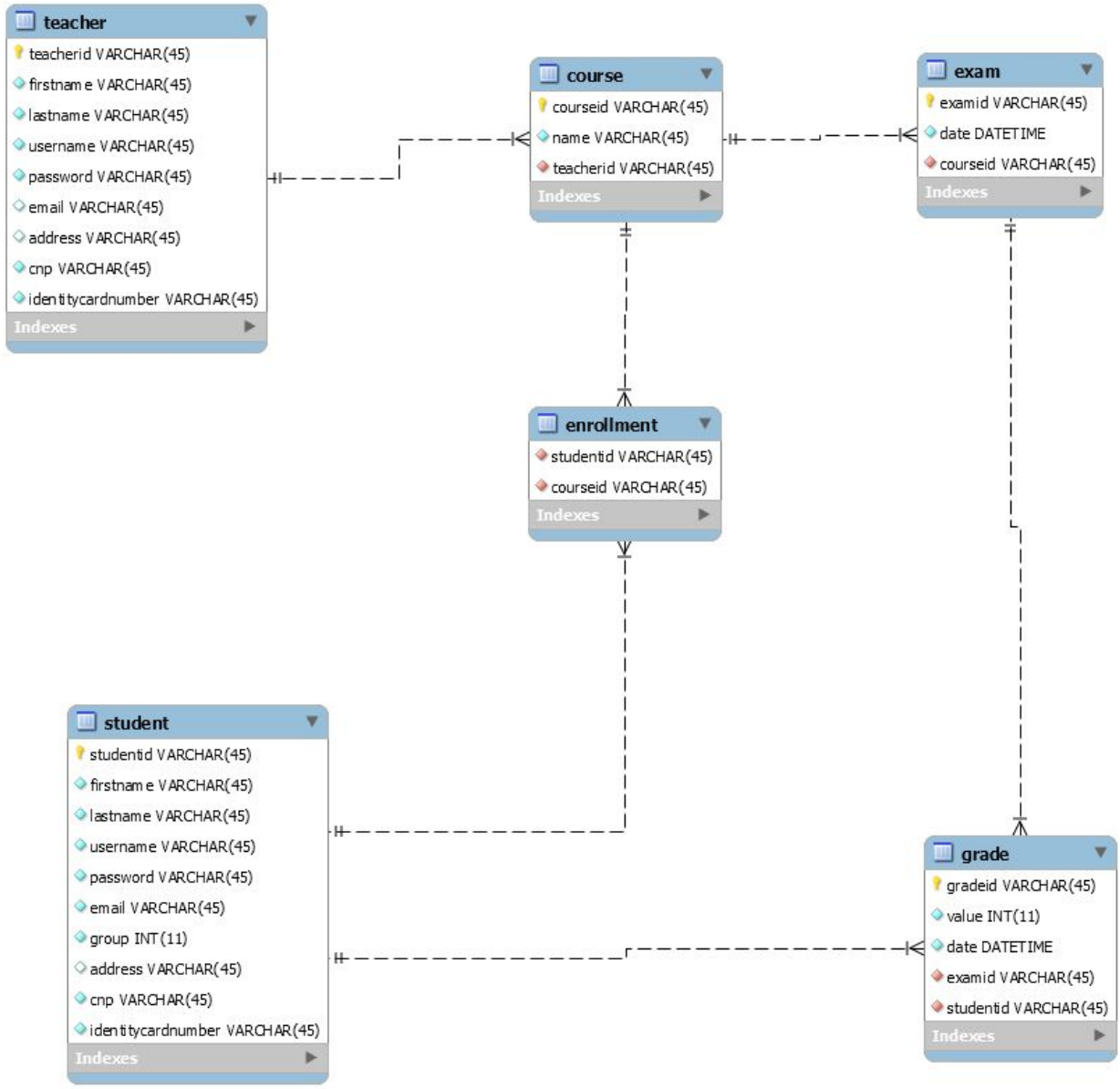
5.1 Design Patterns Description

The design pattern used here is the DAO pattern. It is used to separate low level data accessing or operations from high level business services. The Data Access Object pattern consists of: an interface defines the standard operations to be performed on a model object, the concrete class which implements it and the model class which contains the get/set methods used to store data retrieved with the previous class.

5.2 UML Class Diagram



6. Data Model



7. System Testing

The testing strategy used is unit testing which implies separating the code into unit and testing each unit to see if they meet the required functionality.

8. Bibliography