Assignment 2
Analysis and Design Document

Dan Fulga
Department of Computer Science
Technical University of Cluj-Napoca
Group 30432
danfulga96@yahoo.com

April 12, 2018

Contents

1 Requirements Analysis

1.1 Assignment Specification

The task of this assignment is to implement a Java application which can be used for the management of students in the Computer Science department at UTCN.

1.2 Functional Requirements

The application should have two types of users (student and teacher/administrator user) which both have to provide an username and a password in order 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, enrollments, grades)
-Process class enrollment(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-funtional Requirements

1.Data integrity

Data integrity is a very important requirement whenever dealing with an application that stores personal information. In this application, every data to be inserted is validated, and the system responds accordingly to invalid inputed data.

2.Extensibility/Manageability

This application can be easily further extended and due to its layered-

architecture components, manageability is also increased.

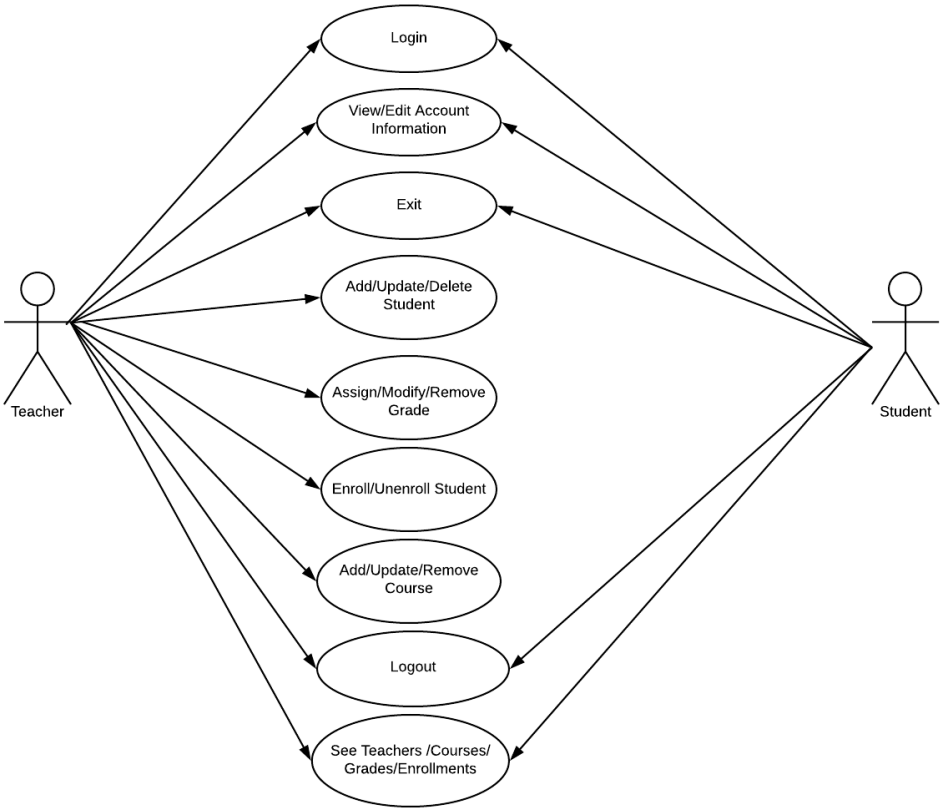
3.Security

Considering that in order to view or manage certain data it is required to login using an existing account, the application is secured. Also, every role in the application(student and teacher) only sees what he is supposed to see.

2 Use-Case Model

Use Case: Update Student Profile
Level : User-goal level
Primary actor: Student
Main success scenario:
1.The student introduces an username and a password and he is successfully logged in.
2.The student will click on the appropriate button to update his profile.
The student will insert data in all the required fields and click on the save button.

Extensions:
1. In case of login failure, the student will see errors such as "user not found" or "invalid password" and his access to the application will be denied.
2. Data-related errors, for example invalid password such as: "Password format is not valid. Please satisfy the following conditions: a digit must occur at least once, a lower case letter must occur at least once, an upper case letter must occur at least once, a special character must occur at least once, no whitespace allowed in the entire string, anything, at least eight characters though".



3 System Architectural Design

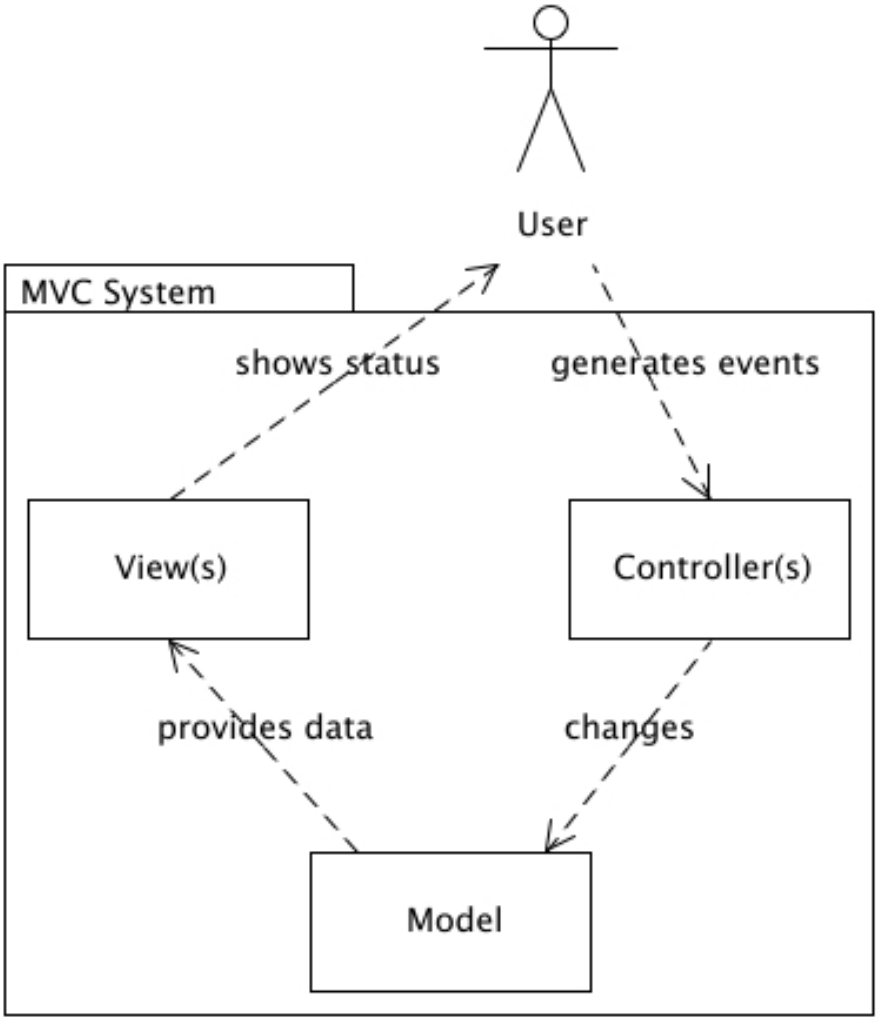
The main design pattern used for implementing this application is the MVC pattern. MVC proposes three types of objects in an application, the Model, Views and Controllers.

1.Model objects hold data and define the logic for manipulating that data.Model objects are not directly displayed. They often are reusable, distributed, persistent and portable to a variety of platforms.

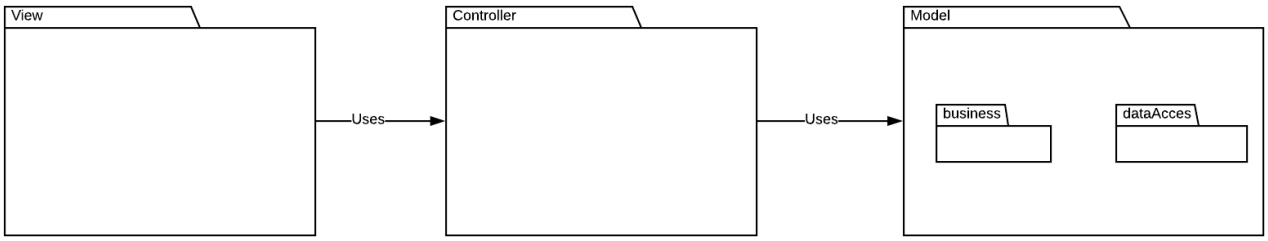
2.View objects represent something visible in the user interface, for example a panel or button. There, data from the model objects is displayed in a customed way by the developer.

3.The Controller object acts as a Mediator between the Model and View objects. A Controller object communicates data back and forth between the Model objects and the View objects. Since controllers are application specific they usually do not find any reuse in other applications.

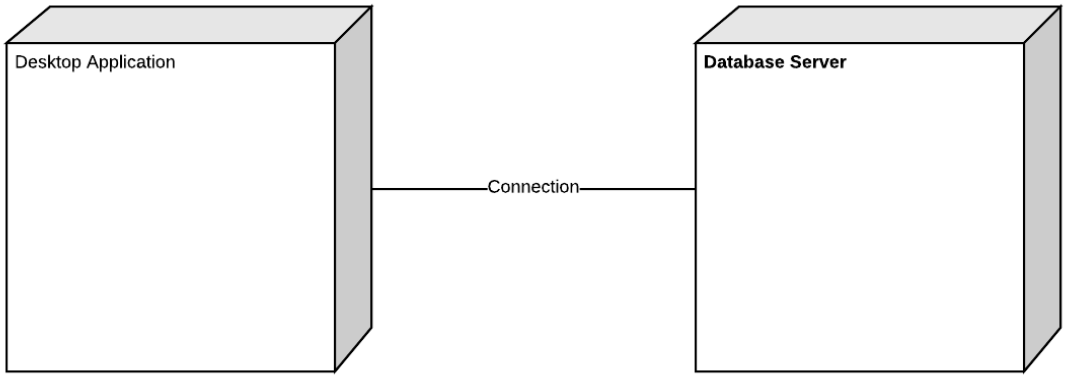
4 Diagrams



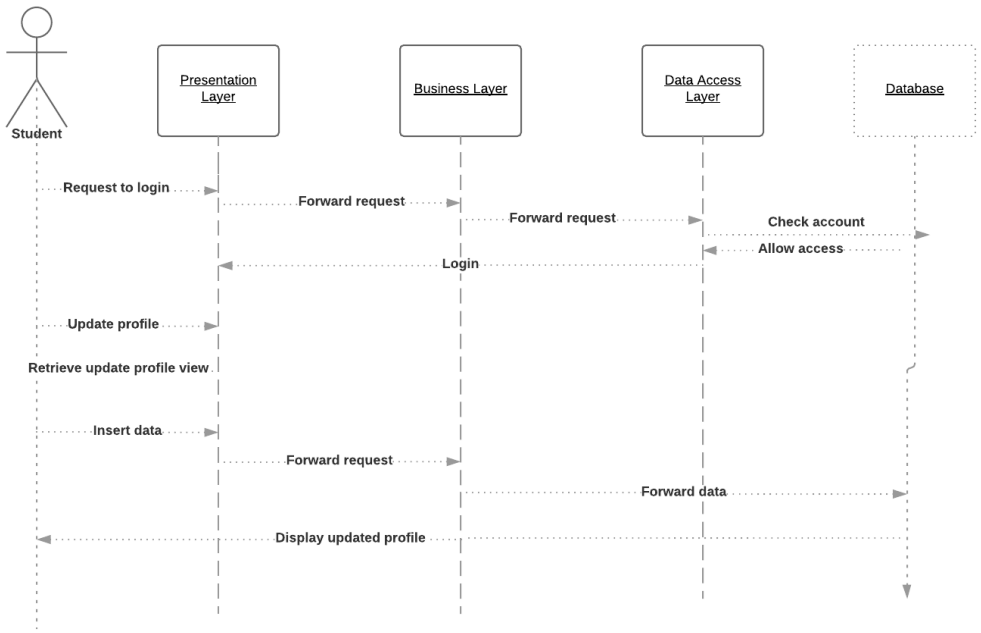
4.1 Package Diagram



4.2 Deployment Diagram



5 UML Sequence Diagram



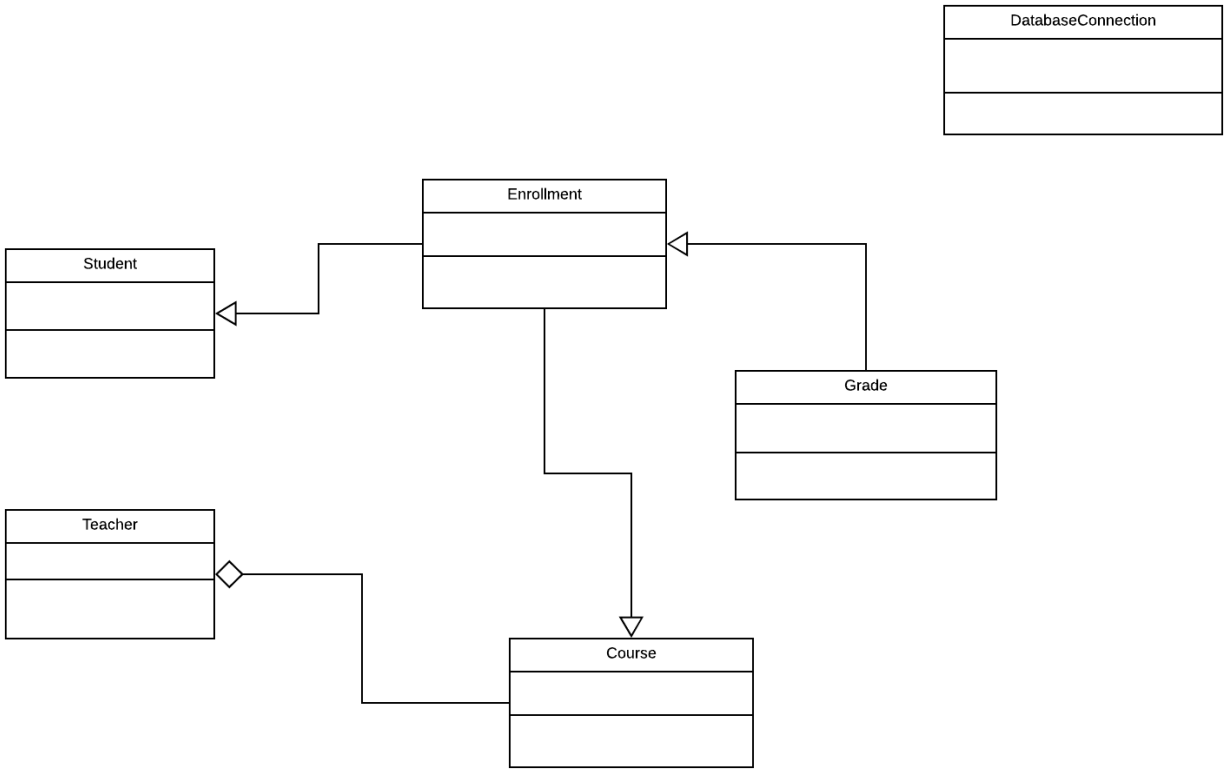
6 Class Design

6.1 Design Pattern Description

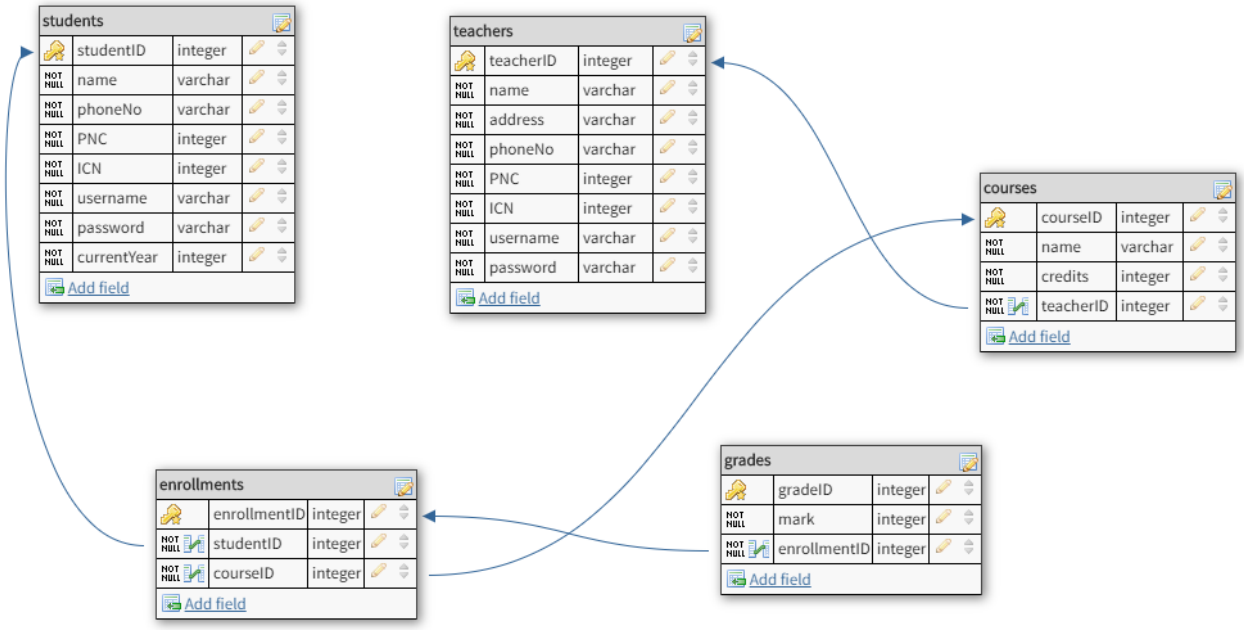
Layering is a common architectural pattern employed to help break up complicated software. Each layer is responsible for exposing a principal service, and layers are typically divided based on their primary functional responsibilities. Packages are an excellent way to represent the logical layers composing an application. Higher-level layers (or a package representing a layer) sit atop lower-level layers, using their services. Lower-level layers, however, know nothing of the higher-level layers that use them. Furthermore, each layer usually hides its lower layers from the layers above, though the transparency of individual layers must be a conscious decision. In this application, the layers are:
-Presentation Layer : responsible for display of application user interface components and handling user requests
-Business Layer : responsible for the application specific business logic
-Data Layer : facilitates communications with underlying services such as

database connectivity.
In the creation of the class ConnectionFactory, the Singleton Pattern is also used. Its purpose is to achieve the status in which we only have one object that can communicate directly with the database. Therefore, the method getConnection returns the unique instance of the class.

6.2 UML Class Diagram



7 Data Model



8 System Testing

The application will be tested using JUnit. JUnit is a test framework which uses annotations to identify methods that specify a test. Some advantages when using JUnit are:
- is the standard library for unit testing in Java
- is supported by all major IDEs
- has extensions for various purposes beyond unit testing Java objects.

9 Bibliography

https://coderanch.com/t/95022/engineering/pros-cons-JUnit
https://www.lucidchart.com
https://www.techrepublic.com/article/commonly-used-architectural-patterns-in-java-applications/