Contents

1. System Analysis

1.1. System Overview

The purpose of the system is to provide a collaborative platform to medical students and professors. The intended audience of this project is medical students and professors. By using ‘Medicollab’, the students can search for their respective professors and can add them to their list. The “Medicollab” will be an Android and Iphone based application. To create a ‘Medicollab’, we will use Java as a backend programming language, XML as a frontend language, and SQL as a database. The purpose of this project is to provide the medical students a complete platform where they can see all their assignments, grades and collaborate with teachers. The students can save a lot of their time by managing all the things in one application and there is no need to use multiple other third-party social media applications. The application will be developed using the Model View Controller (MVC) architecture. By using MVC architecture, we will be able to add the new features in future easily. All the user interface related functionality will be implemented in View component, whereas the core logic or business logic of the application will be implemented in Model. The controller will be responsible for maintaining communication and data passing between view and model.

1.2. System Design

The system design is comprised of Model, View and Controller.

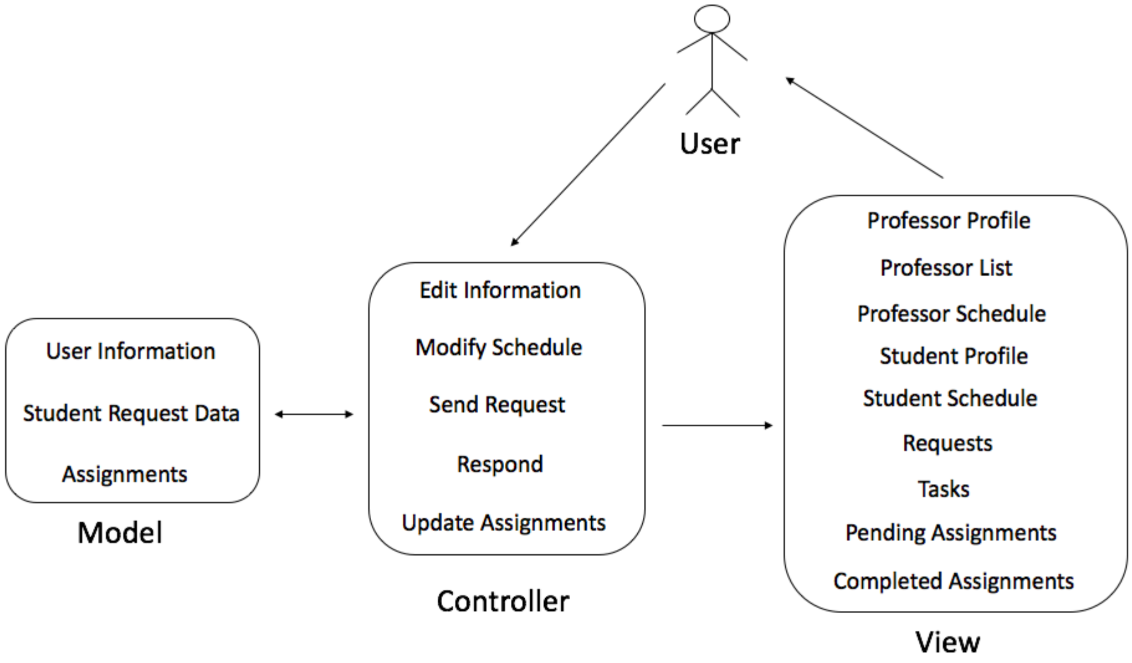


Figure 1: System Design of Medicollab

1.3. Actor Identification

The actors of this system are medical students and professors. Both the actors will sign up and Login in the system. The students can search for their respective professors and can add them to their list. A student will send a request to the professor. The professor can accept or reject the student request. The professor can assign different tasks or assignments to students. Moreover, the students can view their pending assignments, the grades of their completed assignments, and a schedule of their classes.

1.4. Design Rationale

1.4.1. Architectural Style

The Medicollab will be an Iphone and Android based application. The Android application specifically follows the Model View Controller Architecture by default. The view component will handle all the user interface related functionality whereas the business logic of the system will be implemented in the Model component. The controller component ensures the communication between the view and model. The model component further interacts with the system database which store all the related information.

1.4.2. Design Pattern

The design pattern is reusable solution for commonly occurring problems. The following design patterns will be used in the implementation of Medicollab application:

1- **Singleton:** we will use Singleton pattern to ensure that that there is only one instance of a class is created at one time.

2- **Factory:** Factory is a very commonly used design pattern. It is used to create the abstract objects that share the common properties.

1.4.3. Framework

As discussed previously, Medicollab will be an Android and Iphones based application. The Android application follows the Model View Controller (MVC) Architecture by default. So therefore, the intended framework of this system is MVC.

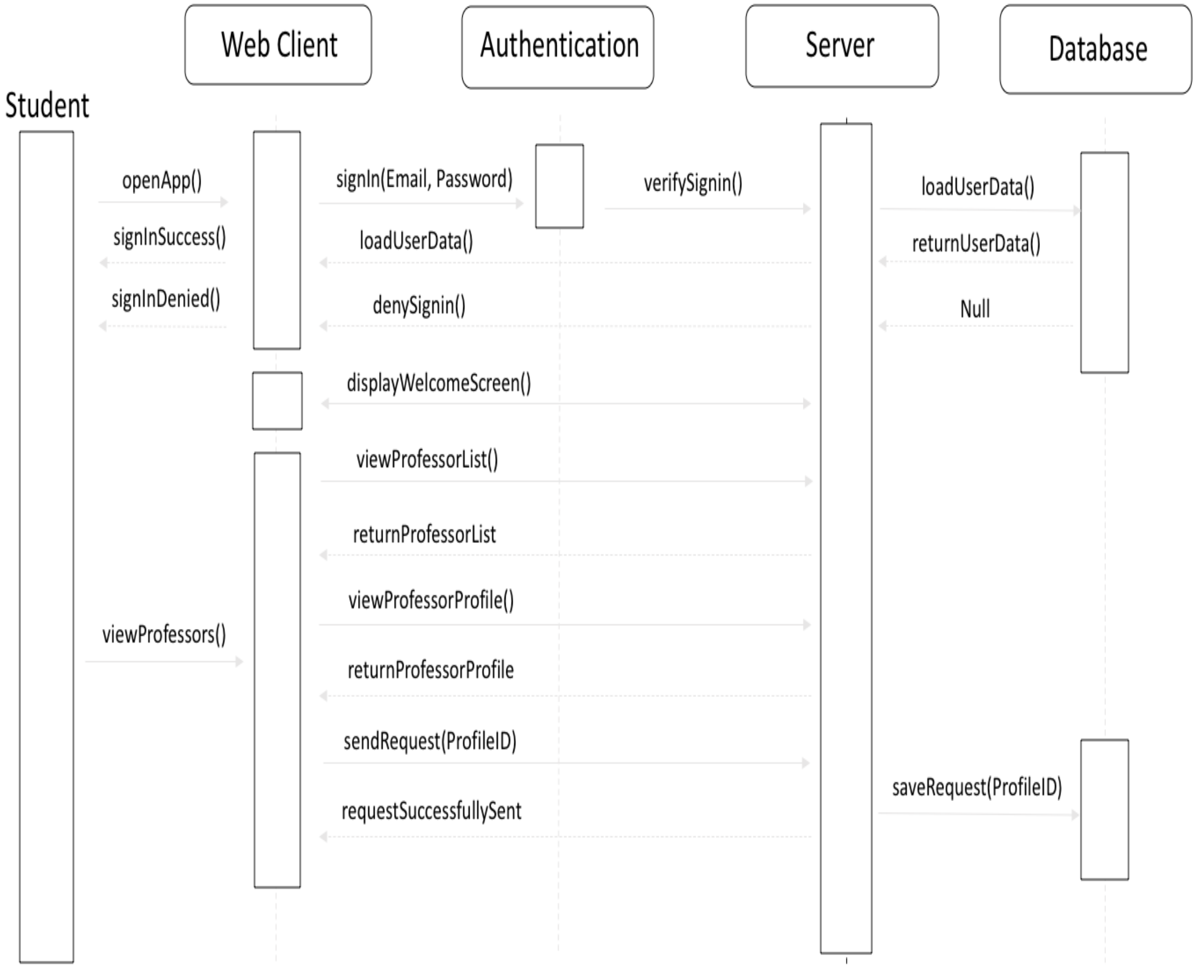
1.5. Cross-cutting Requirements

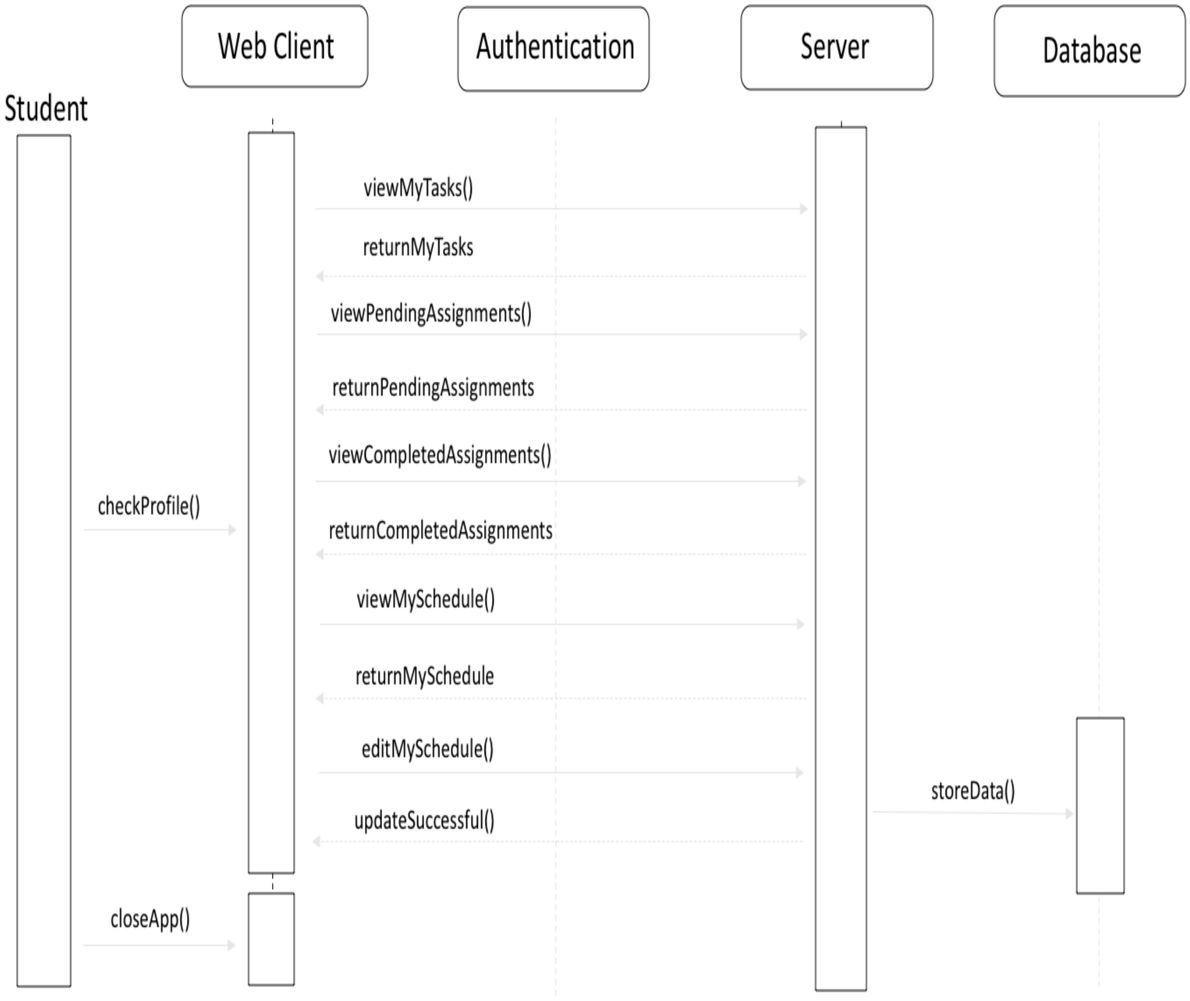
Following are the cross-cutting requirements of Medicollab application:

- **Sign up:** Signing up is a lengthy process in which the user fill out many required fields. It is a critical system requirement because a user cannot perform the other functionalities if the sign up is not performed.
- **Send Request to professor:** It is another cross-cutting requirement in which the student search for the available professors and send the request to professor. After that, the professor add the student in his/her student list.

2. Functional Design

In this section, The sequence diagram outlines all the necessary steps for sending a request to professor.




Web Client
Authentication
Server
Database
Student
viewMyTasks()
returnMyTasks
viewPendingAssignments()
returnPendingAssignments
viewCompletedAssignments()
checkProfile()
returnCompletedAssignments
viewMySchedule()
returnMySchedule
editMySchedule()
storeData()
updateSuccessful()
closeApp()

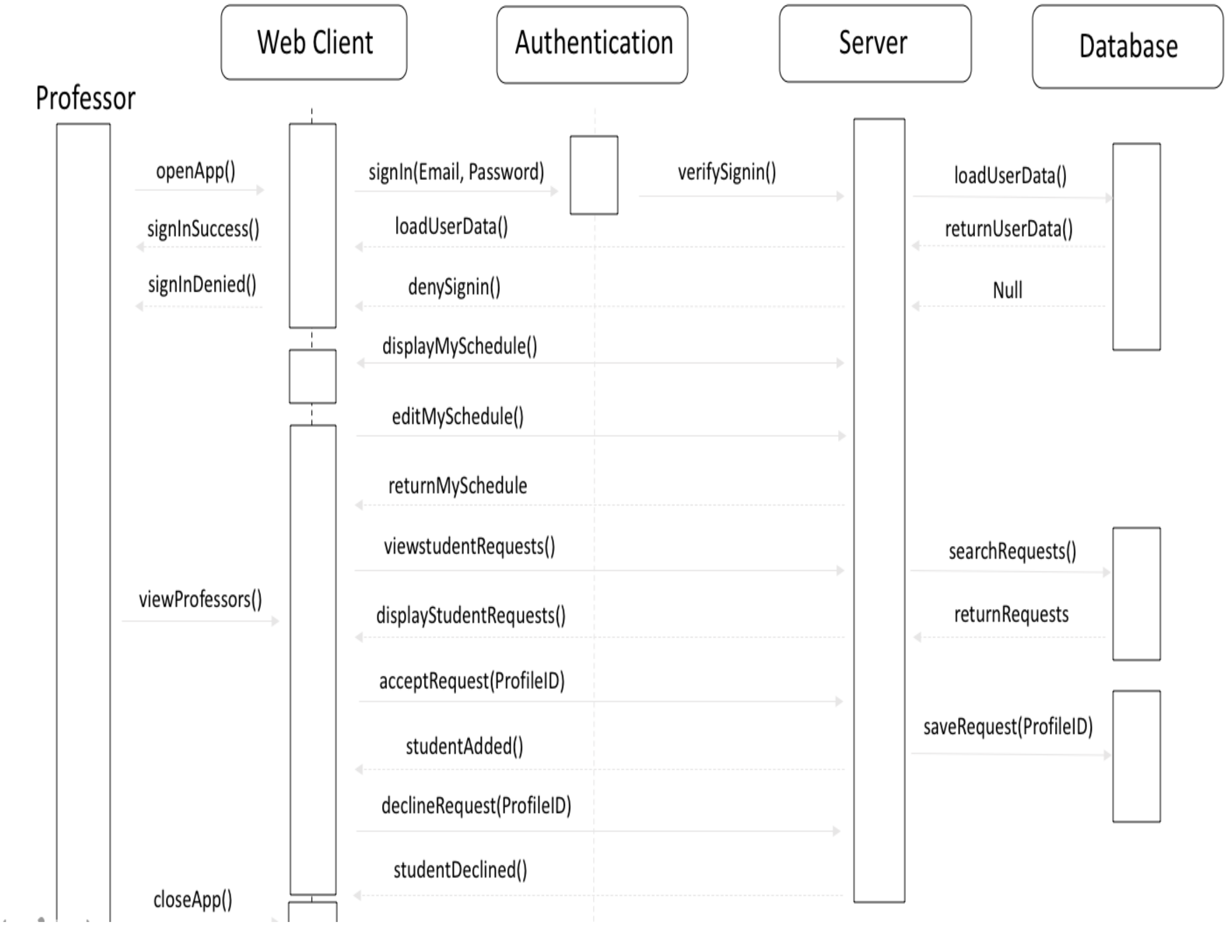


Figure 2: Sequence Diagram for send request to professor use case

3. Structural Design

The below-given class diagram shows the essential classes and structural design of the Medicollab application.

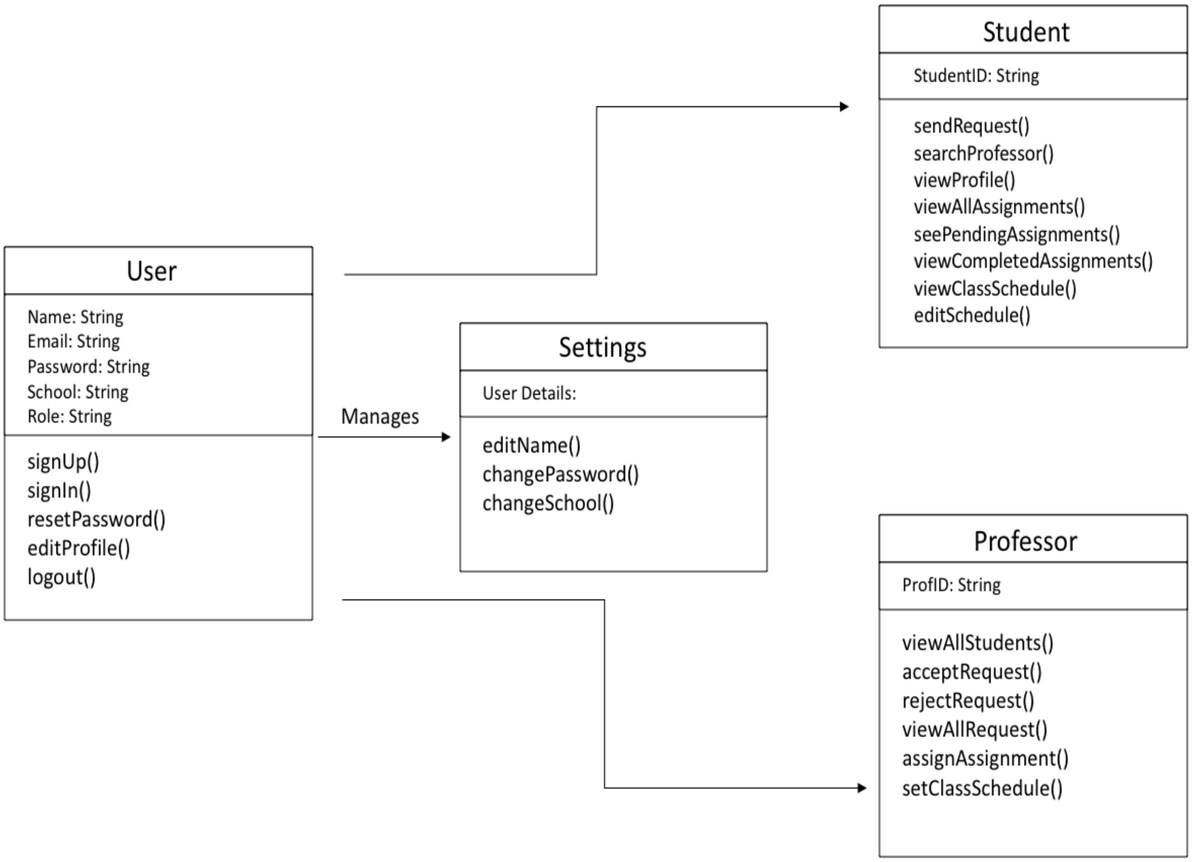


Figure 3: Class Diagram of Medicollab