1. Project Title and Authors

1.1. Team Number 37

1.2. Team Members

1.2.1. Jeffrey Lin - jlin3778@usc.edu
1.2.2. Vincent Li Ruan - liruan@usc.edu
1.2.3. Charles Lee - clee2981@usc.edu

2. Preface

The document serves as documentation of our team's technical implementation, which includes any changes we made to the requirements. This document is used for our internal teams to review our minimum viable product, and examine features that were implemented and those that were not.

3. Introduction

This document outlines all the changes we have made to ship our product's version 1. This includes changes to our Architectural Design.

4. Architectural Design Change

Our Architectural Design largely remained the same. We have a smooth implementation of the Presentation Layer and the Logic Layer, where each individual pages/functionalities had their own layout file — Log in / Sign up / View Course / Review Course/ Chat. And each layout had corresponding Java files to implement the logic. The Persistence Layer involved our implementation of Firebase to store User data, when they register for an account. However, we did not necessarily modularize the use of Firebase (our only external service). Instead, we were calling Firebase in the Logic Layer. For the Database Layer, we create individual classes like Profile Data to store data, and upload/store the data in Firebase.

1) **The Presentation Layer**
   a) The Presentation Layer could also be looked at as the UI Layer. This is where user interaction will occur and where it will act as the interface that handles user interaction in our application. Specifically, our login/logout page, Find Classmate page and Course Bin Page.
2) **The Logic Layer**

a) The Logic Layer will have the functions that implement the logic in our application. Specifically, the process of selecting a class, adding a class, dropping a class, checking whether a classmate is in the class pool and also rating certain classes. This is the core of our Find My Classmates utility. It will not only retrieve information but it will also process the data from our Presentation Layer. When a user adds a class, the Logic Layer will handle the event and processes.

3) The Persistence Layer

a) The Persistence Layer will contain code that interacts mainly with our database. Mainly functions that handle CRUD(create, read, update and delete) operations and API calls. It will also be where we store, retrieve and update data in our persistent storage system.

4) The Database Layer

a) The Database Layer is where all the user profile, class and chat information will be stored. This layer is to ensure that our data is available, consistent and secure over a long period of time.

5. Detailed Design Change

1. We created a class to store the information of the user, which is stored in the **profileData** class. In the profile data class, we store the name of the user, status of the user, as well as the image url of the user. We store the registered course and review of the users in the firebase, instead of the class. In the database, we attempt to store the enrolled courses using a dictionary. The key would be the user, and the value would be a set of all the enrolled classes. We chose a set since users can't enroll in the same class twice, and there are no duplicates. Also, it is easier to drop classes than just remove them from the set.

- It will look something like->

Enrolled classes for each student

User: [Class] mapping (dictionary)

User1: {CS310, CS350, CS360}
User2: {CS270, CS201}
User3: {CS102, CS103}
User4: []
....

Display the courses each user is enrolled in in the profile page
The key of the dictionary is the user email, which can be retrieved from firebase auth

Also, we will use firebase to store the reviews. We will use a dictionary in this case as well, where the key of the dictionary is the class, and there is a list of all the reviews for this course. We will retrieve the reviews from the database and display them in the course page.

- It will look something like->

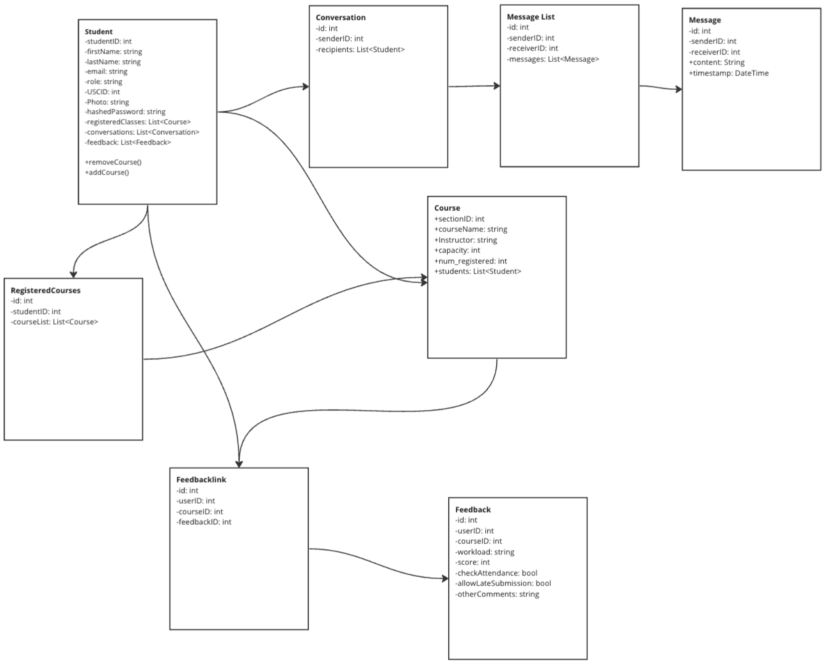
  Class: [Reviews]

  CS310: [review1, review2]
  CS350: [review5, review6, review7]

Finally, we will use firebase again to display the enrolled students in every class. We can use a dictionary as well where the key is the class and the value is a set of students. Then, in the courses, we will go through the set and display all the students that are enrolled in the course.

Enrolled Students in class(to display all students)
CS310: {student1, student2, student5}
CS350: {student2, student6, student8, student10}

Class Diagram



6. Requirements Change

a)

Some changes that we have made:

- Showing just profile picture / allowing user to upload profile picture in profile
- Added a review button for courses
- A feature that we have not implemented as according to the requirements is our chat feature. We are planning to implement this feature in a later assignment in 2.4.
- We haven’t implemented a page that showcases all the reviews that users have given a course, which will be implemented in a later assignment.

b)

For the changed requirements:

- By changing to use the user uploaded image rather than USC ID Card as mentioned in the Project D description, it doesn't really change our design because it will just be an image instead of the ID card.
- We added a review button for courses so that users could go to the review course page to write their review of the course rather than strictly using the course page to add / drop courses. It doesn't change the design other than allowing the user to navigate to the review course page.
- For the chat feature that has not been implemented yet, it does change the design for that feature because it is not able to be accessed by users, but as we implement it later on there won't be any changes in design.
- For the page that showcases all the reviews on the course that hasn't been implemented, this won't change the design. Users just won't have access to all the previous reviews that have been done on the course until we implement this page.