**CS683 Project Assignment   
Iteration 3  
WeFriend  
Tianxin Chen**

**Instructions**

* This document template is used for iteration 1-4 (project assignment 2 - 5). Please specify the proper iteration number in the title and the filename.
* Please name your report as CS683\_<Last Name><First Name>\_<ProjectTitle>\_IterX. It can be either a PDF or Word document.
* Please submit your source code in a zip file named CS683\_<Last Name><First Name>\_<ProjectTitle>\_IterX.zip. You should create a zip file in AS ( using the Menu item File -> Export to zip file … ).
* Please provide your feedback in the “Add comments” section when submitting your lab report. Thanks!

# 

[**Overview**](#_g6igqliy7rm) **2**

[**Requirement Analysis and Test Result**](#_9dheewbiht5g) **2**

[**Design and Implementation**](#_312k3b3li0xh) **3**

[**Project Structure**](#_hkcglxnjhrt2) **9**

[**Timeline**](#_hf7rmhk3rqx2) **10**

[**References**](#_i76cy1p2ent3)11

# 

# 

# Overview

WeFriend is an App for students in college who want to make friends who’s studying in the same college. Because of the outbreak of the cov-19, most schools are holding online courses. It will be hard to make new friends in the new semester. WeFriend will help the user to find someone who study in the same college/major/class. Then the user can send a request to that person, and if that person accept the request, they can start chatting. The potential customer is college students who want to make friends in the new semester.

# Requirement Analysis and Testing

Requirement I have worked on this iteration is mainly the database settings and user information storage of the App. I added features(AuthenticationStateChangeListener ) for both login and register modules of the App so the App will check the sign in status of the account and promote to the next stage of UI automatically. I will show the details in the rest of this Iteration. I also created a setting page which can allow users to log out the account. User can access the page from the top right navigation menu.

I also connect the EditAccount activity to the project structure. User can open the profile page from the navigation Menu. However, since the storage of image has not been set up yet, right now the function of uploading personal image has not been implemented. The Edit Account page will be implemented in the future.

Bellow is the Edit Account page of the app.

A picture containing drawing

Description automatically generated

For the searching function, I did a simple query that can search all user documents with wanted name of school and will return then names of users from searching result. In fact, it took me a lot of time to learn how to implement the Google Firebase database and query. It’s good that I figured it out. In the future, the App will implements more queries with different functionalities. And the Account and profile classes will be used to associated with getting data from online database and store those data with local objects.

Bellow is the searching potential matchups function in the main screen.

A screenshot of a cell phone

Description automatically generated

Now I want to show the login and register work flow of my App.

This is the sign-in page of the App:

A screenshot of a cell phone

Description automatically generated

Assume I don’t have an account and I need to create one, I just click on the hint text at the bottom.

This is the register page:

A screenshot of a cell phone

Description automatically generated

Then I just enter my email address and password. After the code check that the information is correct and the email has not been used, it will send a verification email from Firebase and open the login page.

This is the verification message:

A screenshot of a social media post

Description automatically generated

And This is the Firebase Authentication after click on the link:

A screenshot of a cell phone

Description automatically generated

This is after I clicked on the link:

A screenshot of a cell phone

Description automatically generated

Then we can sign in with the new account. After entered the email address and password and click on login button, it will check if there is already a file created for the user. If not which means it’s the first time the user activate the account, some screens of getting users’ information will be showed.

The page right after click on login button:

A screenshot of a cell phone

Description automatically generated

The next page:

A screenshot of a cell phone

Description automatically generated

The next page:

A screenshot of a cell phone

Description automatically generated

Because our search method is all based on schools, the school cannot be unselected. The spinner bar is a searchable spinner which I found a useful library for it. Because there are over 1,000 colleges in the US, it’s better to have search function instead of just a drop down spinner. The list of colleges is in the String.xml file.

This is the college spinner view:

A screenshot of a cell phone

Description automatically generated

Then we finish the initial set up of the account. All information is now added to the online database.

This is the new document of the just created account:

A screenshot of a social media post

Description automatically generated

We can log out the account from the top right navigation menu:

A screenshot of a cell phone

Description automatically generated

And then we will be taken to an introduction page which user can choose either login or sign up for an account:

A person looking at the camera

Description automatically generated

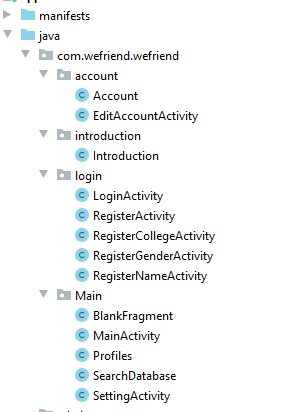
\*The picture is from another app. Because this project is not accomplished yet, I don’t have the proper picture for introduction. It will be replaced for the final deliverable.

# Design and Implementation

To implement the state check in sign in function, I need to connect the project to the Firebase authentication, Firebase databse, and Firebase Authentication state change listener. The state change listener has been added to both login, register, and introduction activities. Firebase database is a NoSQL database. Unlike the database that’s introduced by this class, Firebase does not create table to store data. Instead, it has index. Each document in the same collection can have different fields and data. This feature gives developer freedom to define different users, but also increase the difficulty of querying and error handling.

On the UI side, there are still something I need to figure out in the future. Such as the navigation bar feature. And the edit profile needs to be connected with an online storage of image files. I used menu layout to implement the navigation menu for this App.

# Project Structure



# 

The above pictures shows current structure of my project. This week I have mainly worked on the login and register modules and a little bit of search function. Not the structure of classes becomes more clear as I separate the login and account packages. I have more Register-related classes now. Next week, a search package will be added to the src file.

# Timeline

(Please provide an updated status and plan to specify in which iteration each requirement is implemented (or to be implemented) using which Android features.

|  |  |  |
| --- | --- | --- |
| Iteration | Application Requirements(E/D/O) | Android Components and Features |
| 1 | User Interface | Activities, Animations |
| 2 | Login and account functions | Firebase Authorization |
| 3 | Search functions and query | Firebase query, login/logout connected with Firebase Auth and Cloud Database, |
| 4 | Match up and chatting function, query, file storage, in-App image display | Firebase Storage, Firebase Query, bitmap(TBD)… |
| 5 | Push up notations, Debug and test | Media players(optional) |

# 

# References

# 

1. “[Firebase](https://firebase.google.com/)”. Firebase, inc. Retrieved June 2, 2020.
2. ["Firebase Auth"](https://firebase.google.com/docs/auth/). Firebase, Inc. Retrieved June 2, 2020.