ECE 1778 – Creative Applications for Mobile Devices

January 2021

Programming Assignment P1, for Programmers

Introducing Yourself & Dev Environment & Instagram Front-End

PART I

A key part of this course is to form an inter-disciplinary group to make a new and exciting software application. To help form those groups, and to make sure you have sufficient background as a programmer, please do the following:

1. Create a text description of your background, with the following items:
    - A list of your degree(s), and where you received them. Be sure to include the field(s) you were studying.
    - A list of the computer programming courses that you have taken.
    - A list of the programming *projects* you have undertaken (give the name of the project and a one-line description), together with the size (in approximate number of lines of code) and the computer language used.
    - A list of any companies that you have worked for as a programmer, if any, with a brief description of your responsibilities.

    These will be reviewed by the instructor to make sure that your programming background is at an appropriate level for this course.

2. Create a video of yourself describing the most significant programming project that you have done in the past – its goal and motivation, your role in its creation, and how well it worked. **Also**, give a quick sense of the kinds of projects you might be interested in working on. (This will help begin the key process of forming a team). **The video should be no more than 2 minutes long.** Upload this video to YouTube, and publish it as 'unlisted' (which means someone must have the link to it to see it).

3. Next, go to the Piazza website for this course, and click on the **programmer_intros** folder. (You'll see a video of the instructor in that folder, under the title 'Welcome Programmers!') To do this, go to the **Piazza** course website (instructions for how to enroll into the Piazza are given in an announcement on Quercus) and click on the **programmer _intros** folder at the top left of the site. Click the 'new post' button, and insert a new 'post.' You should place both your written text (from (1) above) and a link to your YouTube video (from (2) above) into this post. Don't forget to paste in your written answers to part 1.

Part I is due on Tuesday January 19th at 6pm. Sooner is better, though! A penalty will be levied if late. See below for Part II.

PART II

The goal of this part of the assignment is to set up the development environment that you will use throughout this course, to use Google's Firebase system to authenticate users, and store user data in a backend server without having to manually manage any servers, and to get started building the app that you will continue to work on and expand in this and the following programming assignments.

For Android Programmers

You will need access to a Windows, Linux or Mac computer, all of which are supported by the Android development environment. For instructions on how to download, install, and use the Android Studio IDE, please complete Lesson 1 of Unit 1 of the Codelabs for Android Developer Fundamentals (V2):
https://developer.android.com/courses/fundamentals-training/toc-v2

Lesson 1.1: *Android Studio and Hello World* will get you completely up and running with Android development.

Learn the Basic Environment

Once you've done the installation and first application, continue on and complete all of Lessons 1, 2, 3 and 4.

This will expose you to the basic development environment, as well as the structure of an Android Project's files, and the Android Studio environment. The later lessons move on to describe *activities* (which are the pages that an app user sees), and how to lay these out. This includes user interfaces such as buttons and text fields, and how to display images.

For iOS (iPhone) Programmers

To develop for an iPhone, you must have an Apple mac computer. To download the environment, open the Apple Mac App Store, and download the latest version of Xcode, version 11.3. Next, go to the website https://developer.apple.com and make sure you can login to the developer website using your AppleID. These credentials will be necessary for using Xcode.

For iOS-based devices (iPhone, iPad) the suggested language to use is the Swift 5 Programming Language. A reasonable textbook, which is free to UofT students, is **Beginning iPhone Development with Swift 5** by Wallace Wang. It is a free download if you are inside the University of Toronto network: https://link.springer.com/book/10.1007/978-1-4842-4865-2

For this assignment, read and do the exercises in Chapters 1, 2 and 3

Assignment P1

Introduction

The three programming assignments (P1 through P3) in ECE1778 are designed to have you complete a single, full-featured app created in these three stages. The goal is to create some of the functionality of the *Instagram* app (https://www.instagram.com): a picture sharing app that allows users to register and then take and upload pictures, all of which are shared with other users of the app. Over the course of the three assignments you will learn the fundamentals of Android (or iOS) development. You will also learn how to build a server-less backend using Google's Firebase (to handle user authentication and management, and data storing and sharing among users of the app). Note that, as mentioned in class, most of your learning will be driven by your own reading and doing. Please feel free to ask questions on the Piazza course discussion website.

Preparation

1. Install and set up your development environment as described at the beginning of Part II of this document. For Android programmers, it is important that you work through Lessons 1 through 4 of the **Codelabs for Android Developer Fundamentals (V2)**:
2. https://developer.android.com/courses/fundamentals-training/toc-v2
   For iOS programmers, you should read and do the work in Chapters 1, 2 and 3 of **Beginning iPhone Development with Swift 5.**

3. Throughout this course we will require that you make use proper source-code control, beginning with this assignment. Since every assignment builds upon the last, it is critical that you maintain a healthy codebase with the ability to undo/rollback errors. If you are not already familiar with any form of source code control, please be sure to review the side video on this topic.

   Install and set up a Git version control/source control tool for use with your ECE1778 programming assignments. We ask that you please use Git because later in the course we will be asking you to push your project's codebase to a repository under our control.

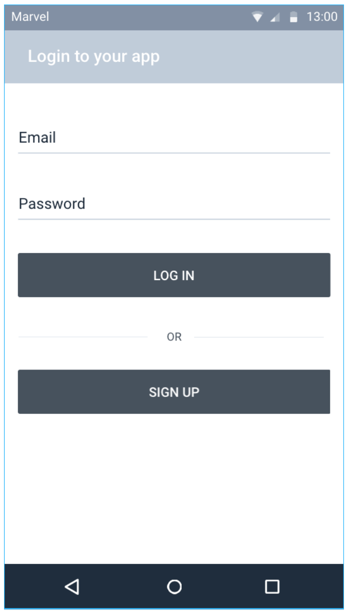
   We suggest you use Github's Student developer pack:
   https://education.github.com/pack which is a free set of software development tools, including **private** Github repositiories (these normally cost money). If you choose to use a hosting service (like Github or Bitbucket) for your repository, we must insist that you **please do not push your assignment code to a public repository**! This would enable others to plagiarize your work.

Assignment Specification

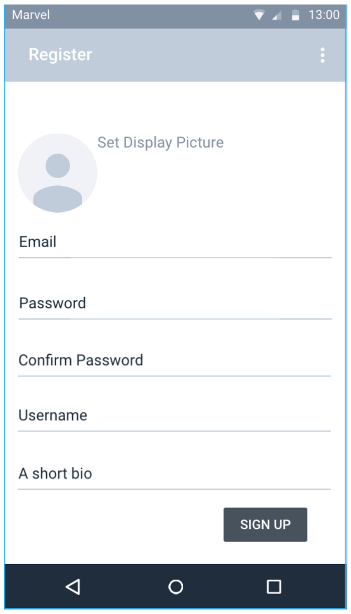
You are to write a mobile application allows a User to *register* for the Photo Sharing app. There are two parts to this – creating the screens to collect information, and then performing authentication using Google's **Firebase** online system.

Screen Specification
The application will have the following three screens:

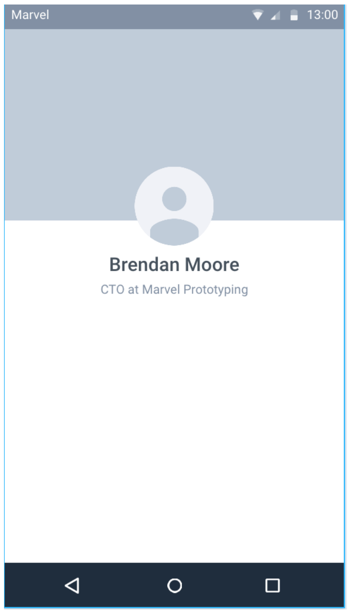
1. Log In / Sign Up screen:
   This screen (illustrated below) prompts a user to sign in using their email address and a password, with one button to Log In (to an existing account) and one button labelled SIGN UP. The SIGN UP button should take the user to the screen described below in point 2.



2. Registration/Sign up Screen:
   This screen (shown below) asks the user for the necessary information to sign up/register. There should be a clickable element at the top of the page to allow the user to launch the camera and take a picture which will serve as their display picture. Textboxes should be used to ask the user for the email and password that they would like to use for authentication, in addition to their display name and a short biography (which should be just a single sentence). Finally, there should be a button ("Signup"). The functionality of that button is described below in the section labelled **Authentication Using Google Firebase**.



3. Profile Page
   Render an empty Profile page that shows the user's icon (a thumbnail of the picture taken with the camera on the previous page), the user's display name, and their short bio.



Authentication Using Google Firebase

We will be using Google's Firebase cloud-based system for a number of things in the three assignments. For this assignment, we'll make use of its authentication system, which creates login IDs and passwords for user authentication. Below we describe the required functionality of the screens above, in concert with Firebase authentication.

Once you've read this specification, you'll need to learn about Firebase as described below in the section **Learning Firebase.**

1. On the Sign-in Page:
    a. The App should launch to this page if a user is not logged in; otherwise it should launch to the Profile page.
    b. The "Log In" button should attempt to authenticate a user's given email and password using **Firebase** authentication, described below. If login succeeds, the "Profile Page" should be loaded, but if it fails, it should give an appropriate message on the sign-in page.
    c. The "Sign Up" button should take you to the "Registration Page"
2. On the Sign up/Registration Page:
    a. Prompts the user for all the described data. A **Firebase** User should be created for the given email and password (using Firebase Authentication).
    b. The display picture taken by the user does not need to be saved to Firebase for this assignment.
    c. The user's username and bio should be saved as a document within Firebase Cloud **Firestore**. All of these documents should be saved within a collection called 'users' in the **Firestore** database.
    d. Once the above steps are completed, the user should be taken to the "Profile Page"
3. Profile Page:
    a. Somewhere on the Profile Page there should be a button to log out the user and return them to the Sign-in Page.

Learning Firebase

To begin integrating Firebase into your app and learning how to use its features, please read through the links below.

1.1 Setting up Firebase dependencies

Begin by adding the basic Firebase setup to your app, according to the following guides:

Android: https://firebase.google.com/docs/android/setup
iOS: https://firebase.google.com/docs/ios/setup

Next, add Firebase **Authentication** can be added according to the following guides:

Android: https://firebase.google.com/docs/auth/android/start
iOS: https://firebase.google.com/docs/auth/ios/start

Next, Firebase **Firestore** can be added by according to the following guide:

Android & iOS: https://firebase.google.com/docs/firestore/quickstart

Finally, Firebase **Storage** (which won't be needed for this assignment) can be added according to the following guides:

Android: https://firebase.google.com/docs/storage/android/start
iOS: https://firebase.google.com/docs/storage/ios/start

1.2 Firebase Sample Apps

Google provides very helpful sample apps complete with source code and written guides which demonstrate how to use all the necessary features of Authentication, Firestore, and Storage which you will need to complete the labs. We provide links to these sample apps below.

Authentication – user ID and password authentication
Android: https://github.com/firebase/quickstart-android/tree/master/auth
iOS: https://github.com/firebase/quickstart-ios/tree/master/authentication

Firestore – storage of information in Google's cloud servers
Android: https://github.com/firebase/quickstart-android/tree/master/firestore
iOS: https://github.com/firebase/quickstart-ios/tree/master/firestore

Storage – larger file storage in the cloud. (This will be needed for Assignment 2)
Android: https://github.com/firebase/quickstart-android/tree/master/storage
iOS: https://github.com/firebase/quickstart-ios/tree/master/storage

2 Important Notes

This section provides some clarification on the specification and also gives some important information which will save you from common pitfalls. Also, you can see a video of one possible implementation here.

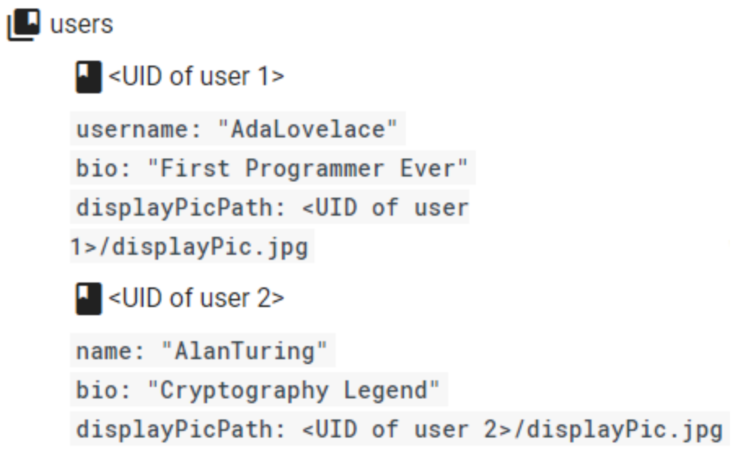
2.1 Suggested Organization for Data in Firestore Database

First, familiarize yourself with Firestore's data model, here:

https://firebase.google.com/docs/firestore/data-model

There are many ways to organize your data, and you're free to choose your own organization. One way that you can organize your data in Firestore is to maintain two collections: a collection of "user" documents called "users", and a collection of photo references called "photos" (to be used later).

The “users” collection would be a flat collection of “user” documents, where each “user” document uses the associated user’s UID as its key, and saves the associated username, bio, and reference to display picture as data:



users

<UID of user 1>

```
username: "AdaLovelace"
bio: "First Programmer Ever"
displayPicPath: <UID of user
1>/displayPic.jpg
```

<UID of user 2>

```
name: "AlanTuring"
bio: "Cryptography Legend"
displayPicPath: <UID of user 2>/displayPic.jpg
```

Further Instruction and Tips

Be sure to write your code cleanly so that it can be extended in assignment P2!

To obtain the user’s display picture with the camera, Android programmers can review this resource https://developer.android.com/training/camera/photobasics.

iOS programmers can review this resource:
https://developer.apple.com/documentation/avfoundation/cameras_and_media_capture/

Follow the steps which show you how to request the camera feature, take a photo with the camera app, and then get the thumbnail. The image thumbnail is enough for out purposes here.

You should only need an emulator to do this assignment, not an actual phone. For Android, make sure that you target the highest level of the Android SDK, with a minimum SDK level of 23 (Android 6). For iOS, please sure your code runs on the most recent version of iOS version 13.

With all assignments, including this one, we’d like you to produce applications that work well and robustly, as good training for the app that you’ll make in the project. As such, we require that you follow **Braiden Brousseau’s guide to Quality Apps**, which is appended below.

- **Additional Application Behavior**

In addition to all the behavior specified above, the application should *also* behave in the following ways:

1. Uninstalling and reinstalling the app (or clearing the app's data) should not delete any data from the cloud storage "backend". If a user were to delete and reinstall the app and then log back in to their account, the app should display all their original data on the Profile page (i.e., username, and bio). This will be tested.
2. The app should support multiple users. For example, a user should be able to register an account ("account 1"). Afterwards, if they were to logout and register a new account ("account 2"), account 1's data should still persist and should be accessible if the user logs out of account 2 and logs back in to account 1. This will be tested.

<u>To Hand In</u>

For Part I: Due Tuesday January 19th at 6pm, as described above.

For Part II: Due Tuesday January 26st, 6pm. Marked out of 10, 0.5 marks off every hour late.

What to submit:

Android: a zip file containing both a final .apk of your assignment and your complete project, runnable in android studio.

iOS: a zip file of complete project, runnable on Xcode 12.3.

Submit your zip file through the **Quercus Assignment P1** page in this course. **(**Please do not accidentally submit it to the Assignment S1 page).

Braiden Brousseau's Guide to Quality Apps

The purpose of this guide is to ensure that the software you create in this course – both the project and the assignments, meet a certain standard of quality and robustness. The assignments will include grades allocated towards these guidelines, as an additional motivation. Applying these concepts will also improve the quality of the larger project app. These guidelines come from previous years' experience with marking assignments and projects. These guidelines are written for Android programmers, but similar concepts apply for iOS.

Don't let the User crash the App

The user shouldn't be able to crash an app by pressing touch objects in the "wrong" order or by excessive clicking of an object. For example, if an activity requires Button 1 to be pressed before Button 2 (for example to select what data file should be used before processing), then pressing Button 2 first should *not* cause a crash. Either nothing should happen, or better yet the user could be notified that they must select a data file first by pressing Button 1.

Don't Make the User Wait

Avoid unnecessary slowdowns caused by overuse of new activities and fragments where not appropriate.

Avoid waiting for large files to be read or written to storage. For example, if you need to load a very high-resolution image, consider loading just a thumbnail version of the picture first, while waiting for the full resolution version to load. As another example: when saving a large piece of data consider doing so asynchronously if the required save time noticeable impedes use of the app.

Make use of time while a user is idling. In an app that shows a user the top stories of the day from a website, load the data for story 2 while the user is reading story 1. It is very likely the next action from the user will be to click for the next story. This creates a better experience than having to wait for the download to happen.

Lists and other scrollable User Interface elements should always scroll smoothly.

In the context of the assignments in this course your app should never 'feel" slow or laggy. Most of the time the code you write will result in your apps behaving in fast and smooth manner, without any intentional consideration while programming them. However, when they don't behave smoothly, it is usually a sign that it is implemented in an inefficient way, and you are encouraged to research and evaluate other techniques and implement something, that might be more complex but gives superior user experience.

Use UI Elements Appropriately

Example: In Android, you can programmatically set the text in an “EditText” box, if the app has no intention of the user entering text here use a “TextView” or another element that the user cannot change.

Content-Independent Interface

The UI should behave and look similar regardless of what data is loaded or entered. For example, loading a picture of different sizes should not cause buttons in the app to physically move around to different places on the screen. Loading and displaying a large text file shouldn't push UI elements off of the bottom of the screen, or somewhere else unreachable. If the UI changes based on what content is loaded it should be intentional.

Appropriately Sized and Labeled Touch Targets.

Avoid making UI elements (that the user must touch) very small and close to other elements they might be touched by accident. It should be clear what can be touched and what action will be taken when the element is touched. One can use text labels, icons, pictures, colors etc. to convey to this type of information.

Fill Space Appropriately

UI elements should have fairly commonsense space occupancy. For example, suppose an activities' main purpose is to measure how long it takes to run a certain distance and show the current time along the way. Making the running time font size 8sp in the upper right-hand corner while making a ‘share my time to twitter’ button 3/4 of the screen is probably bad design.

Explore the apps you use for inspiration on this type of design. While we are not requiring outstanding UI design, we do want you to learn and demonstrate your ability to finely control UI elements. Becoming comfortable with this during the assignments will pay significant dividends when the project works starts as you have short development cycles in between presentations of your progress.