CSC 214 PROJECT 2

DUE SUNDAY APRIL 9th AT 11:59PM

The goal of Project 2 is to implement a social networking application that supports multiple users on the same device. Don't worry, *actual* networking will not be required as part of this assignment. Instead, users will be able to create accounts, log in, browse their network feed, and log out so that another may log in on the same device.

It is strongly recommended that you design your database schema before beginning to code. Changes in the schema may require you to wipe out the database that currently exists on the system by uninstalling the application. This could potentially be *very* annoying if you have to create user accounts, profiles, and feed posts every time you test.

Please note that there will be **absolutely no extensions granted** for this project. There simply is not enough time to move the deadline on project 2 and still have enough time in the semester to finish project 3. It is **strongly recommended** that you look at your schedule for the next three weeks *now* and determine when you will be able to devote time to the project. You are given three weeks to work on this project because I expect that it will take that long for you to complete it. You are not given three weeks so that you can wait until the third week to start and ask for an extension. **Time management is your responsibility**.

0. Meta Requirements

- The user should not be able to access the main application without first logging in. The *only* part of the application that should be accessible is the login/account creation screen.
- Once logged in, the user should be able to navigate between parts of the application using toolbar or menu options. Any part of the application should be accessible from any other part.
- The application should include "up navigation" that will return the user to the main network feed from any other part of the application.
- Once logged in the user should be able to log out from any part of the application. This should return the user to the login/account creation screen.
- Every part of the application should support orientation changes without losing state.

1. Account Creation and Logging In

- All data for the user accounts must be stored persistently in a SQLite Database in the file system. All data should persist across launches of the application.
- When the application is first launched, users must be given the option to create a new account with an email address and a password.
    - If an account with the same email already exists, the user should receive an error message.

- Users that already have an account should have the option to log in with their username and password.
    - If the username does not exist, the user should receive an error message.
    - If the password is incorrect, the user should receive an error message.
- No other part of the application should be available unless and until the user has logged in.

2. Profile Creation

- All data for the user profiles must be stored persistently in a SQLite Database in the file system. All data should persist across launches of the application.
- A user must be able to specify features of their profile. At a minimum, the user must be able to:
    - Take a photo and use it as a profile picture. The photo may be stored in the file system (not in the SQLite database) but the path to the photo file should be stored in the database.
    - Specify a full name.
    - Specify a birthdate.
    - Specify a home town.
    - Write a short BIO (e.g. a few sentences of text)

3. Social Networking

- All data for the user social networks must be stored persistently in a SQLite Database in the file system. All data should persist across launches of the application.
- A user must be able to browse a list of other users and
    - View the profile of any user in the list (including the profile photo)
    - “Favorite” the users that they want to see posts from.
    - See users that are already favorited highlighted in some way (e.g. a different colored background or an icon).
    - “Un-Favorite” previously favorited users so that the user will no longer see posts from the other user.

4. The Feed

- All data for the user feeds must be stored persistently in a SQLite Database in the file system. All data should persist across launches of the application.
- A user must be able to browse their feed. They should see posts by any user in their favorites list *in reverse chronological order*.
- A user must be able to post status updates that include text and/or images taken with the device’s camera.

Additional Enhancements (Extra Credit Opportunities)

You may earn extra credit for any enhancements above and beyond the most basic implementation of the features above or by adding additional features. You may receive up to 20% extra credit in total, but you **must** mention your extra features in your readme in order to be considered for extra credit.

Grading Criteria

You will be graded according to the following criteria.

Class Definitions (20%)

You are expected to practice the Model View Controller (MVC) design pattern. This means that any classes related to application logic (e.g. the game rules) should be implemented in model classes and NOT in widgets (the view) or the controllers (the activities). Your model should contain classes for things like accounts (usernames, passwords), profiles, and posts. The database schema that you design should include tables and columns that reflect the attributes of your model. **Again, I cannot emphasize enough: you should design your schema *before* you begin coding.** Figure out what attributes are needed in your classes, and then make sure that a table/column exists for each attribute. Remember that you will need keys to uniquely identify some classes (e.g. users) and those keys will appear in multiple tables. For example, each user in the user table should have a unique key to identify them, and that key should be used in the posts table to identify posts written by that user. As much as possible, the controller should receive events from the view and translate these into commands (method calls) on the model. The controller should then respond to changes in the model by updating the view.

Login/Account Creation (10%)

Users should not be able to access the application without first logging in. This should be the only screen that is accessible when the application first launches.

Application Implementation (70% total)

- Meta-Requirements: persistent toolbar options, orientation changes (10%)
- Profile Creation (20%)
- Social Network (20%)
- The Feed (20%)

Code Style (Lose up to 10%)

- Modularity of design - Objects that make sense. High cohesion, low coupling.
- Comments - Where appropriate!
- Names - Variables, Methods, Classes, and Parameters
- Indentation and White Space

Documentation (Lose up to 10%)

In addition to submitting the Java code (.java files), you are required to submit a README text document which specifies the following:

1. A one paragraph description of your class designs and schema.
2. The typed version of the academic honesty pledge: “I affirm that I did not give or receive any unauthorized help on this project, and that all work will is my own.”
3. Full documentation for any enhancements that you wish to be considered for extra credit. YOU must explain what your enhancements are, and how the TA should enable them or otherwise detect their presence.

HAND IN

Projects will be submitted as ZIP archives via Blackboard. You will be given 3 weeks to complete the project, and there will not be an assignment due the same week that the project is due. This is more than enough time to complete the project.You will also have unlimited attempts to submit your project (only the last submission will be graded), which means that you will be able to upload your solution as soon as you have something working, and try for improvements or extra credit afterwards. Because of this, late submissions **will not** be accepted.

You will be expected to submit:
- Your Android Studio project complete with all resources and source files.
- Documentation described in the section above. If you submitted any additional files, make sure to explain their function in your documentation.

Grading

Each TA will run your code in Android studio on a virtual device. If your program produces only partial functionality, you will receive a grade for the features that you did complete. Each TA will also conduct a code review to evaluate the correctness of your code and the code design and to determine whether or not it was likely stolen.