CSE 5236 Project Description

Instructor: Adam C. Champion

Autumn 2016 Semester

Total: 60 points

The team project (2–3 students per team) for this class involves conceptualizing, designing, and developing a mobile application on the Android SDK. The application should meet the set of functional and non-functional requirements described below. Teams should follow a well-managed development process that results in well-designed, well-documented software.

High-Level Requirements

We discussed in class that a mobile device serves as two things: (a) it serves as your gateway to everything else; and (b) it represents you. Thus, the app that you conceptualize must be you, represent you, and be unique to you. In addition, the app must meet the following requirements:

- Must uniquely meet a clear need;
- Must be well-designed (by following the process outlined in the course);
- Must have a UI (i.e., it can’t only be a background service);
- Must have a rich set of domain objects (at least three);
- Must have data that is persistent across sessions;
- Must use one or more external services (e.g., maps);
- Must use one or more sensors (e.g., GPS, accelerometer, light sensor, etc.).

Note: All student teams in a class section need to design original distinct apps. There should not be multiple team apps that meet the same need (e.g., multiple apps that “find nearby events around me” using GPS, Google Maps, and Facebook). I expect that each team designs and develops an original app that is distinct from apps that are already available on the market as well as apps designed by other teams in class. Since the project involves designing and implementing an original app that represents you, use this opportunity to show your creativity in your design. In addition, your app can be a game that meets the above requirements and originality considerations.

Stages of Evaluation

In the following, “submit” means to submit to your group folder on BuckeyeBox. Alternatively, you can submit files to Google Drive, Dropbox, or a similar website. If so, please tell the grader and me where you submitted them. Please do not email your app project code to the grader or me. There have been numerous cases where emails with project code have been rejected due to malware detection (even when no malware was present).

Part (1): Android Environment Setup and Running Sample App

Due Date: Monday, Sep. 5, 2016, 11:59 p.m.

Points: 5/60

Submit: Screenshot(s) of Tic-Tac-Toe app running from the Android Studio IDE

Show: The Tic-Tac-Toe app running from the Android Studio IDE

Evaluation Criteria: App comes up on the emulator (or mobile device).
Part (2): App Requirements and Design

This part involves envisioning and designing a mobile app that meets the requirements specified in the “High-Level Requirements” section.

Due Date: Monday, Sep. 26, 2016, 11:59 p.m.

Points: 5/60

Submit and Show:

- App concept notes: Narratives, sketches, categorized list of use cases, database schema, screen layouts, and screen flows;
- Object-oriented design of application domain model.

Evaluation Criteria: Clear evidence of the design process.

Part (3): Installing, Debugging, Profiling, App Lifecycle Management, Logging

This part involves demonstrating your ability to do the following:

- Installing your app on a device;
- Debugging your app (e.g., stepping through code and setting breakpoints);
- Profiling your app using one or more profilers (e.g., Android Studio’s CPU, GPU, or memory profilers or Instruments for iOS);
- The app lifecycle and logging. Here you must implement a simple part of your app and demonstrate using the Android log that you can force the app’s lifecycle methods (onPause(), onResume(), etc.) to be triggered.

Due Date: Monday, Oct. 17, 2016, 11:59 p.m.

Points: 10/60

Submit:

- A screenshot of at least one type of profiler (e.g., CPU, GPU, or memory);
- Screenshot(s) of the app’s lifecycle methods being triggered
- Your app project as a zip file.

Show:

- A mobile device running a “recent version” of the OS (e.g., Android 5+);
- Installing your app on a device;
- Debugging your app (setting breakpoints and stepping);
- Profiling your app;
- Invocation and logging of lifecycle methods.

Evaluation Criteria:

- Demonstrated understanding of the use of the IDE;
- Demonstrated understanding of the Android application lifecycle.

Part (4): Functional Demonstration of Your App

For this part you have to demonstrate a functioning app. The app should work from beginning to end, although it need not be optimized for performance, or be resilient to failures such as network failures, GPS failures, etc.

Due: Monday, Nov. 7, 2016, 11:59 p.m.
Points: 20/60
Demo:
- Your app working on a device (if it works only on an emulator, you’ll only receive partial credit)
Submit:
- A list of use cases (only a list, no descriptions)
- Screenshots of your app running
- All code (simply export and zip the project)
Evaluation Criteria: TA will check off working use cases.

Part (5): Demonstration of Non-Functional Characteristics of App

For this part you have to demonstrate the non-functional capabilities of the app such as usability, performance, availability, maintainability, modifiability, and scalability. The app should work from beginning to end, should be optimized for performance, and be resilient to failures such as network failures, GPS failures, screen rotation, killing by the OS, etc. Non-functional requirements (NFRs) should be specific to the application, and, ideally, quantified.
The minimum requirements are as follows:
- Performance NFR addressed and demonstrated using “before and after” profiler snapshots
- At least one other NFR addressed (e.g., security or design enhancement)
- Unit testing using an Android test framework demonstrated (example frameworks include JUnit, Robotium, and Roboelectric)
Due: Monday, Apr. 28, 2016, 11:59 p.m.
Points: 20/60
Show:
- The working app on a device demonstrated to work under failures (network connectivity, GPS, screen rotation)
Submit:
- List of use cases and NON-functional requirements met
- Screenshots of the app
- Profiler screenshots showing areas where performance was improved as well as the “baseline” before improvement (e.g., “before-and-after” screenshots of CPU, GPU, or memory consumption)
- All code (simply export and zip the project).
Evaluation Criteria: Number and quality of working use cases and non-functional requirements met.

Final Report

Due: (Fixed) Day of the final exam.
The final report for this class will be a team report that analyzes and critiques the design process described in class. The report should consist of three sections as follows:
- Describe the design process and its intended goals in your own words.
Describe how you translated the design into the implementation of the app. In this section, describe what worked well, and where there were gaps in the process. For these gaps, explain how you bridged them.

Suggest changes or improvements in the design process that you believe would make the design more complete and more easily translatable into the implementation.

The report should be 4–5 pages (single-spaced) written in 10–12 point font. Sample report templates are provided on the course website.