SENG440: Topics in Mobile Computing Project 2

1 Overview

Your responsibility in this project is to make an app that fully embraces mobility by interacting with the surrounding environment through sensors or proximity-based networks. The design and purpose of the app is mostly unspecified. You are encouraged to make something that interests you or for a community that you care about.

2 Requirements

The requirements for this project fall into two categories: mandatory requirements and grade-bearing requirements. For your project to be eligible for marking, you must satisfy all of the mandatory requirements by the end of the due date. If these requirements are met, the grade you receive will be calculated based on how many of the grade-bearing requirements you meet, your collaboration effort, and the overall quality of the product.

Complete the following **mandatory requirements** by the due date:

- Collaborate in a group of 3. (I will consider other group sizes but only with good reason).
- Create an app that has a particular and meaningful purpose for some audience. It should not just be a sandbox app in which you cobble together disconnected features.
- Use Kotlin for program logic, not Java. Create your user interface declaratively with XML or Jetpack Compose, not imperatively through Kotlin.
- Maintain and submit your project in a Git repository on the departmental GitLab server.
- Document your app with a post mortem post on Slack. In your post, tell us a story about the app's purpose and development process. Describe the contributions of each team member. Include screenshots. At the end of your post, enumerate which of the grade-bearing requirements you believe you have met and how. Be brief but specific in your enumeration.

Each of the following requirements will contribute to your grade:

1. Interact with the nearby physical world in some way. This interaction might rely on sensors, GPS/location, the camera, or Bluetooth to create a local area network.
2. Interact with the nearby physical world in some additional way.
3. Choose **one** of the following:
    a. Handle one type of input based on a non-simple (i.e. non-click) Gesture in a non-standard way, e.g. fling, drag, multitouch, etc.
    b. Use ML Kit library, Tensor flow lite, or an equivalent toolkit to run non-trivial analysis on user input or sensor data.

4. Provide a facility for “openness”—for a user to interoperate with entities beyond their phone or beyond your app. This could take many forms. You could allow the user to export a backup of your app's data that could be imported by someone else or on another device. You could provide support for deep-linking, such that certain patterns of URLs would trigger your app. You could allow the user to share achievements via texting or social media. If your idea for this feature situates your app within the larger community, it probably qualifies.
5. Gracefully handle configuration changes, not losing any of the user's data.
6. Use a local database to persist data, preferably using `Room`.
7. Send the user notifications related to your app in some way.
8. Integrate an action bar in at least one activity.
9. Provide a preference screen using the modern AndroidX Preference Library.
10. Add a multi-resolution launcher icon.
11. Support both landscape and portrait orientations in all views—unless your content demands a fixed orientation, as in a game. In other words, all widgets should be able to be made fully visible in either orientation.
12. Use string resources for all static text on the user interface.
13. Write a set of user stories for your app and include them in your post mortem.
14. Find two people who are not part of your group, have them test your app, and record their feedback in your post mortem.
15. Incorporate two other Android features not mentioned above into your app.
16. Successful demo of your app during the last week of class.

Possible substitution: I will consider substituting 2 requirements above for publishing your app on the Play Store. (Publishing will not count as extra credit, only as a substitution.) Be forewarned that preparing an app for publication is a lot of work and costs a wee bit of money—by some definitions of *wee*. **If you are interested in this substitution, declare your intentions to me within the first 2 weeks of term 2.** The review process for Android apps takes a few days, so you need to plan ahead and finish your app before the deadline if you choose this option. Your app must also be a complete app, not work-in-progress with unfinished features – Google will not approve your app in that state.

Part of your grade will be awarded based on your investment in the collaboration. It will be calculated as a function of your number of individual commits to the shared repository, mentions of your contributions in the Slack reports, and feedback from your teammates.

Finally, I will assess the quality and creativity of the overall product. For this I will not be comparing different groups’ apps to one another, but rather I will assess the overall creativity and thought put into the design of your app *as well as its suitability for its intended purpose and audience as indicated by your initial proposal and post mortem*. If your app simply rehashes tutorial code and is not fit-for-purpose, then you will not receive a high mark in this area.

If you have questions about any of these, communicate with your instructor. Do so early and often. Please share your questions in #general.

3 Submission

To submit your project for grading, complete the following before the due date:

- Commit and push your app to your Git repository.
- Verify that the push succeeded by visiting your repository via your provider's web interface.
- Publish your post mortem on Slack in the #project2 channel.