1 Term Project Details

This document provides details about the project such as how iterations will work, and general application constraints.

1.1 Iterations

1.1.1 Team Retrospective

Each iteration (except the first) starts off with some reflection on how the last iteration went. This is key to Agile methodologies because the team owns the process, so the team evolves the process by analyzing how things are going.

As a team, discuss the following questions and record a brief answer to each in the `docs/retrospectiveN.txt` file where N is the iteration number, e.g. `retrospective2.txt`. There will be no `retrospective1.txt`.

1. What did we observe during the iteration? What was noteworthy?
2. What suggestions do we have for next time? How can we be a smarter, happier, more productive team?
3. What did we do well that we want to keep doing?

1.1.2 New and Updated Use Cases

The project kickoff slides presented a long-term road-map for each app. These are feature ideas that may or may not be required for some iteration.

Each iteration will come with new and/or updated use cases for each app that describe the new functionality to be implemented. The use cases will be written from the user's point of view with plenty of room for interpretation by your team. This not only simulates how real users may give requirements but also allows your team some flexibility in adding your own ideas when implementing features. If you would like to implement a feature in a way that is contrary to how it is described in the use case, please talk to the customer representative (the instructor) for clarification.

1.1.3 Scrum Master

Each iteration, the group selects a new scrum master. This must rotate among the group members. Nobody shall be the scrum master twice.

The scrum master is responsible for:

1. Organizing group meetings and helping to keep meetings on topic.

2. The only person on a team who can contact the customer representative (the instructor) and ask questions related to features for that iteration. Anyone may ask about other aspects of the project, such as marking, or raise concerns to the instructor. Basically, if it's a question the customer should answer, the scrum master must ask it.
3. Organizing how activities during the iteration are completed, such as team retrospective, task-list updating, and project submission.
4. Being the scrum master does not give anyone management powers over team members. Remember, he or she is a facilitator.

1.1.4 Task List

In the `docs/` folder of your project, create a `tastlistN.txt` where N is again the iteration number, e.g. `tasklist1.txt`. In it, state who is the scrum master for this iteration. Also list the development tasks you have come up with based on the use cases. Beside each task state who will take the lead on implementing/completing the task. This need not be an extensive list or 100% accurate, but should be a good indication of who has been doing what.

1.1.5 Submission & Marking

Each iteration your team must submit one `ZIP` file of your project to CourSys for your working software. The submission should not contain the folders holding your compiled classes (to keep the file size down). Recall that `git` can help you create such a `ZIP` file. Your project must contain at least:

1. Android Studio project files.
2. Java code, Android resources (`strings.xml`, ...), and project "assets" (images, sounds...)
3. `docs/` folder including:

- `tasklistN.txt`: states who is the scrum master for this iteration, and lists task breakdown from the use cases.
- `gitN.jpg`: Screen-shot showing `git` commit history for your project. Show at least this iteration on the screen. N refers to the iteration number. Take this picture within GitLab.
- `debug.keystore`: A single, shared keystore for the group, used to hold keys for, e.g., the Google Maps API. If you use `git` to create your `ZIP` file, remember to add the files in `docs/` to the repository before creating the `ZIP` file so that they are included in the archive.

Each iteration a marking guide is also posted so you know what is expected and how much it is worth. All iterations will be marked by the TA for your official

grade. Iteration 3 will also be evaluated by the customer for feedback about meeting their needs. Possible bonus and prizes for outstanding work meeting the customer's needs.

Questions about the marking of your work should be directed to the TA (via the cmpt-276-help@sfu.ca email list). If you are unable to resolve your question with the TA, then please bring your question to me.

Additional requirements apply to the 3rd iteration's delivery. See updates to the course website for more information.

1.2 Application Constraints

1. All development must be in the `git` repository creating for your group created in GitLab and visible to the TA and instructor.
2. Apps must run on Android 4.2 (API 17)
3. Well built UI, including:
   - Well laid out usable screens.
   - Works on different sized screens. Example tests on 3", 4", 7", 10".
   - Use of icons and images where possible.
   - Strings that the user can see must be in strings.xml to support internationalization. This will be required for features in later iterations.
4. Clean Code
   - Good class, method, and variable names.
   - Perfectly formatted code.
   - Comments on all classes (not needed inside classes, just on the class level)
5. Simple OOD supporting not much more (if anything) than the current set of requirements.
6. Good commit comments on (almost) all commits.