CPEN 321 Software Engineering
Winter 2023

M9: Final Product (Thursday, November 30, 11:59pm PT).

The deliverable for this milestone is your final product. This includes:

1. Refined specification of your project (requirements, design, test design, etc.)
2. Implementation of your project's front-end, back-end, and tests
3. Video presentations of different project components
4. APK of your application, which works against the back-end that is up and running until grading is completed.

You **can** use assistive AI technology (namely, ChatGPT 3.5) when working on this assignment. No points will be deducted for documented use. However, undocumented use will be considered academic misconduct and will be treated accordingly.

Submission: The submission will include four parts.

PART I – Final Project Report. Submit a pdf file named "YourProjectName-part1.pdf", which includes the following information in the order specified below:

1. The up-to-date description of your project.
   - At this stage, we expect the project
     - to be in-scope w.r.t. the course expectations;
     - have the implementation aligned with requirement and design documentation;
     - have a good design and implementation;
     - have no unjustified issues flagged by Codacy;
     - have tests that are well-designed, achieve high coverage, and pass.
2. Your refined project requirements, as defined in M2 (use case diagram, use case formal specification, non-functional requirements). Mismatches between your requirements and implementation will cause you to lose marks.
3. For the three use cases for which you implemented front-end tests, specify the design of the test, as defined in item 6.2 of the M6 spec. The test design should be listed **right after** the corresponding use case specification in item 2.
4. Your refined project design, as defined in M3 (components, interfaces, component diagram, description/design of the "complexity" function). Mismatches between your design and implementation will cause you to lose marks.
5. For each exposed interface of your back-end, specify the location of the test for the API and what mocks were implemented to test API, if any, as defined in item 5.2 of the M6 spec. This information should be listed **right after** the corresponding interface specification in item 4, for each API exposed to the front-end. The remaining tests that do not belong to any particular API, e.g., those of recurrent events, should be listed separately.
6. The link to your GitHub repository and the commit SHA of your final released version.
7. The locations of your front-end and back-end tests in the repository.
8. Screenshots of Jest coverage reports for the back-end tests and logs of the front-end tests, as specified in M6.
9. The description and logs of tests of the non-functional requirements, as specified in M6.
10. The "Issues Breakdown" table of running Codacy on the final release version, as specified in M5.
11. The manufacturer and model of a physical device your app is running on (not an emulator).
12. All necessary user accounts or similar details we need to run your app.

13. The list of limitations of your project.
14. "Humblebrag": a description of any extra part of your project you are particularly proud of.
15. Reflections on using Generative AI technologies for software engineering tasks. You must answer and elaborate on all the following questions in 1-2 paragraphs each:
    - For which phases of the overall Software Engineering process and which concrete tasks in these phases was ChatGPT the most helpful and why? Discuss phases such as requirements elicitation and documentation, design, implementation, code review, testing, and project refinement (for the final release).
    - For which phases of the overall Software Engineering process and which concrete tasks in these phases was ChatGPT the least helpful and why? Discuss phases such as requirements elicitation and documentation, design, implementation, code review, testing, and project refinement (for the final release).
    - What is your summary / overall assessment of the appropriateness of ChatGPT to help with software engineering tasks?
    - In the context of software engineering, for which tasks do you anticipate you will use ChatGPT or other types of AI technologies in the future? Which types of AI technologies?
    - Is there anything else you would like to share about AI's impact on your software engineering tasks?
16. The contributions of each team member to the work done on the project (2-3 sentences for each team member).

PART II – Project Presentation. Submit a pdf file named "YourProjectName-part2.pdf", which includes a presentation describing your project. The presentation should contain exactly 9 slides, as specified below:

1. Slide 1: The name of the project, the group members, and the contribution of each group member to the work done in the project.
2. Slide 2: The project's high-level idea and target audience.
3. Slide 3: Description of the actors and three most central use cases of the project.
4. Slide 4: The high-level design of the project.
5. Slide 5: The "complexity" idea, its design, and implementation.
6. Slide 6: Breakdown of the remaining Codacy issues and test coverage results.
7. Slide 7: List of non-functional requirements, your methodology for verifying the requirements, and verification results.
8. Slide 8: Highlights of the workflow that you followed (Scrum meetings, GitHub Actions, etc.)
9. Slide 9: The most important lesson you learned from the project.

PART III – Project Videos. Submit a zip file named "YourProjectName-part3.zip", which includes videos describing specific aspects of your project. Each team member should record one of the videos #2-#5 in full. Any / all team members can participate in recording video #1.

Each video should show both the presenter and their screen. Your video must be on for the presentation. When you show code or text on your screen, make sure it is of an appropriate size and clearly visible. You can use Zoom for recording.

Each video should be in mp4 format and be named "YourProjectName-video1.mp4", etc. **Make sure no video exceeds 50-60MB, to ease uploading in Canvas**. Allocate yourself enough time to upload videos in Canvas, to avoid issues caused by high traffic close to the deadline.

The required content and length of each video are specified below:

- Video #1 – **The App** (12 minutes): Present the slides you submitted in PART II and then demonstrate your app on a real physical device (not an emulator). To demonstrate live updates, use an emulator or a second device and record both. Make sure to clearly identify all main use cases and the complexity idea of your app. Cover all other noteworthy parts about your app.
- Video #2 – **Front-end** (4 minutes): Present the structure of your front-end code and show a few representative code snippets. Show the implementation of your tests and describe how you checked for expected results. Run the tests and show how they execute and the results they produce. If your non-functional requirements relate to the front-end, describe the requirements and how they are tested. Provide any additional info you find relevant.
- Video #3 – **Back-end code** (3 minutes): Present the structure of your back-end code and show a few representative code snippets. Describe the back-end components (using a diagram from the slides in Part II) and show how they are mapped to code. If your non-functional requirements relate to the back-end, describe the requirements and how they are tested. Provide any additional info you find relevant.
- Video #4 – **Back-end tests** (3 minutes): Describe the structure of your tests. Show a few representative code snippets. Demonstrate how you mocked external dependencies of your back-end (APIs of external components, databases, etc.) Run the tests for 1-2 central interfaces and show the coverage and test status results. Run all tests combined and show coverage information. Provide any additional info you find relevant.
- Video #5 – **Codacy + GitHub Actions** (3 minutes): Show your GitHub Actions configuration file and explain its structure. Explain which functional and non-functional back-end tests are triggered by GitHub Actions and how. Make a commit to your repository and show that the tests are running (and pass). Show that Codacy is configured to run with the provided setup, run it for the latest commit in your project on the main branch and describe the results, as specified in PART I (i.e., demonstrate that no issues are detected or provide an appropriate justification why the remaining issues cannot be fixed).

PART IV – Mobile Application. Submit the APK of your application, named "YourProjectName.apk". Make sure: (a) the app runs on a Google Pixel 3 emulator running Android Q (API 29) and (b) the back-end is up and running until the grading is completed.

GRADING

Grading will be based on

- Presentation quality for PARTS I-III (covers all required material, timing, clarity of explanations, etc.)
- App feature-completeness and usefulness (the app is in-scope, all requirements are implemented and are fully functional, the app is easy to use and clear)
- Code quality (well-designed and implemented, checks for error conditions, secure, performant, etc.)
- Test quality (coverage, completeness, checks for error conditions, degree of automation, especially in UI and non-functional requirements test, etc.)
- Ad-hoc code and test reviews performed by the TAs.

Good luck!