CS Capstone Design

Technical Demo Grading Sheet (100 pts)

TEAM: HealthLit

Overview: The main purpose of the "Technical Demos" is to very clearly communicate the extent to which the team has identified key challenges in the project, and has proven solutions to those challenges. Grading is based on how complete/accurate the list of challenges is, and how convincingly and completely the given demos cover the given challenges.

This template is fleshed out by the team, approved by CS mentor, and brought to demo as a grading sheet.

Risky technical challenges

Based on our requirements acquisition work and current understanding of the problem and envisioned solution, the following are the key technical challenges that we will need to overcome in implementing our solution:

C1: Module Implementation: Module implementation will be difficult because the team will need to make sure that the back-end and front-end of the application is able to withhold and display the modules. *We will need to create faux module(s) to make sure that the app is able to retain the modules*.

C2: Game Creation: It will be challenging to ensure that we can translate the game ideas properly into the app. *Therefore, we will need to show that we can make a basic game and push it into HTML format.*

C3: Modeling Language: We must ensure that the modeling language that we chose (UML) can properly represent our app's many components and relationships. *Therefore, we will need to show part or most of the app modeled in UML.*

C4: Database Communication: Our database(Google Firebase) will need to hold all of our user's information. The database has to interact with both Flutter for client-side communication, and PHP for admin-side testing and management. *Therefore, we will need to show that communication with our database will work with both reading and writing.*

C5: Dart: It will be slightly challenging to learn the Dart programming language since our team is not familiar with it.. *Therefore, we will need to show basic mastery of the Dart programming language..*

Challenges covered by demos:

In this section, we outline the demonstrations we have prepared, and exactly which of the challenge(s) each one of them proves a solution to.

Demonstration 1: Modeling Language Demo

Challenges addressed: Modeling Language

Flight Plan: Step by step overview of demo

1. Create a partial UML diagram of the system.

Evaluation:

- ✔ Convincingly demo'd each of the listed challenges?
- ✔ Other evaluative comments:

Demonstration 2: Module Demo

Challenges addressed: Module Implementation

Flight Plan: Step by step overview of demo

1. Create a document that does not retain any real health literacy information.
2. Upload document on back-end (firebase) to test how it will be retained.
3. From the back-end, upload the module to the front end of the application to see how it will be displayed and make adjustments as needed.

Evaluation:

- ✔ Convincingly demo'd each of the listed challenges?
- ✔ Other evaluative comments:

Demonstration 3: Game Demo

Challenges addressed: Game Creation

Flight Plan: Step by step overview of demo

1. Create a basic game with GMS2 that has user input and sprite-based graphics.
2. Demonstrate that the game can conduct those operations.
3. Show a way for the game itself to close after completion.

Evaluation:

- ✔ Convincingly demo'd each of the listed challenges?
- ✔ Other evaluative comments:

Demonstration 4: Database Demo

Challenges addressed: Database Communication

Flight Plan: Step by step overview of demo

1. Create a Project in Flutter
2. Connect Firebase to Flutter
3. Add a Database Entry
4. Read that same Database Entry and Display it

Evaluation:

✔ Convincingly demo'd each of the listed challenges?

✔ Other evaluative comments:

Other challenges recognized by not addressed by demo:

If there were challenges you listed earlier that were *not* covered by a demo, list here. This will hopefully be a short list…but better to be clear about where you are. If you have items here, you could list (if applicable) any pending plans to reduce these risks.

- GMS2 conversion to HTML5
  - Despite this being a simple fix, it will not be shown in our demo. We just need to pay for a license to export GMS2 projects to the HTML format.
- Learning Dart
  - Another simple fix, our team will simply need to take some time together to learn the syntax and behaviors of Dart to demonstrate basic mastery of the language.