**Software Implementation and Testing Document**

**For**

**Group Phaze 5**

Version 1.0

**Authors**:

Faith Miller

Mackenzie Knight

Eduardo Antonini

Tristan Garcia

Katie Rombeiro

# **1.** **Programming Languages (5 points)**

* JavaScript with React: Used for game implementation and in all of the front-end design as well as the server
* HTML/CSS: Used for the front-end design to make the web app look pretty

We chose these JavaScript because we researched that JavaScript is the most common with webapp games and React is great for front end aspects. We chose HTML/CSS because those are most widely used for front-end design aspects.

# **2.** **Platforms, APIs, Databases, and other technologies used (5 points)**

We used Atom for all of our coding and we used Express and Node.js for the server even though the server is not currently being used. We are doing everything on localhost at the moment.

# **3.** **Execution-based Functional Testing (10 points)**

We have not performed a large amount of execution-based functional testing, but we have done a minor amount of testing so far. For starters, those who have been coding the user interface in HTML and CSS with React have been checking to see what the UI looks like on the localhost after every few changes to the code. Checking the UI on the localhost an adequate amount of times while coding is essential because it allows us to see if there are any errors in our code causing the program to not compile. In addition, when creating a React App, a file called App.test.js automatically gets created in our project. We have renamed that file Homepage.test.js because it correlates with the Homepage file. In the next increment we will be researching how the test.js file works and we will potentially do some of our testing for the Homepage in this file.

Those who have been coding the game implementation do not have enough code to perform a large amount of execution-based functional testing yet. Although there is not a large amount of code for the game implementation, we have given each person in the group the task of reviewing another person’s code each day to make sure that there are no issues with the code. This concept of peer reviewing is essential because it allows for us to find minor mistakes in another person’s code that they might have not noticed.

# **4.** **Execution-based Non-Functional Testing (10 points)**

We have not yet performed a large amount of non-functional testing for our project yet because we do not have quite enough code to begin these testing procedures. Although there is not a large amount of code, we have been testing to see if the state continuously gets updated on the front end and back end. This is essential because state plays a big role in our Phase 5 game. For example, when a person enters that they will be playing the game with 3 players the program updates the state of number of players to 3, but we have been testing to make sure that the state gets updated if the user changes their mind about the number of players.

# **5.** **Non-Execution-based Testing (10 points)**

After a decent amount of code has been added to the project, we have been using the Inspect tool on google Chrome to see if there are any errors. This allows us to see what errors are occurring and if there are any warnings that are being thrown. In addition, we have designated a specific person to each person in our group so that they can peer review each other's code. This allows us to find any errors in our code that we may have missed. We definitely need to perform a larger amount of non-execution-based testing, but we will focus on that in our next sprint.