**Software Requirements and Design Document**

**For**

**Group Phaze 5**

Version 1.0

**Authors**:

Faith Miller

Mackenzie Knight

Eduardo Antonini

Tristan Garcia

Katie Rombeiro

**1.** **Overview (5 points)**Phaze 5 will be a Web App of a shorter version of the card game Phase 10. Anywhere from 2 to 6 players will be able to play the game. The first step in our project is to get our Web App up and running on localhost, where the game will be played by passing the computer around between players. Our end goal is to eventually use the server that we set up to deploy our web app.  
  
The frontend of our Phaze 5 Web App is created using React, JavaScript, HTML, and CSS. The backend implementation of the game is in Javascript. The server consists of Node.js and Express. If we have time, we will use localtunnel, ngrok or now in order to expose our localhost to deploy our Web Application.

# **2.** **Functional Requirements (10 points)**

Functional requirements for the backend of our Web App include (1) coding the game implementation, which consists of classes Player and Deck, along with all of their methods. Methods in the Deck Class include shuffle() and dealCard(). Methods in the Player class include draw(), sortHand(), and makePlay(). The game implementation requirement is essential for the functioning of our app, so it has high priority.

Functional requirements for the frontend of our Web App include (2) using react and HTML to code the user interface of the game that takes in player moves in the game and (3) making the app look appealing to the eye. Coding the user interface of the game has high priority, as it will be used to send game players’ moves to the backend implementation of the game. Making the app look pretty has low to medium priority. It is important to have an appealing user interface, but the interface will be improved once we have a functioning app. The frontend of our Webapp that involves making the connection with the backend to make player moves is high priority because without player moves there is no game.

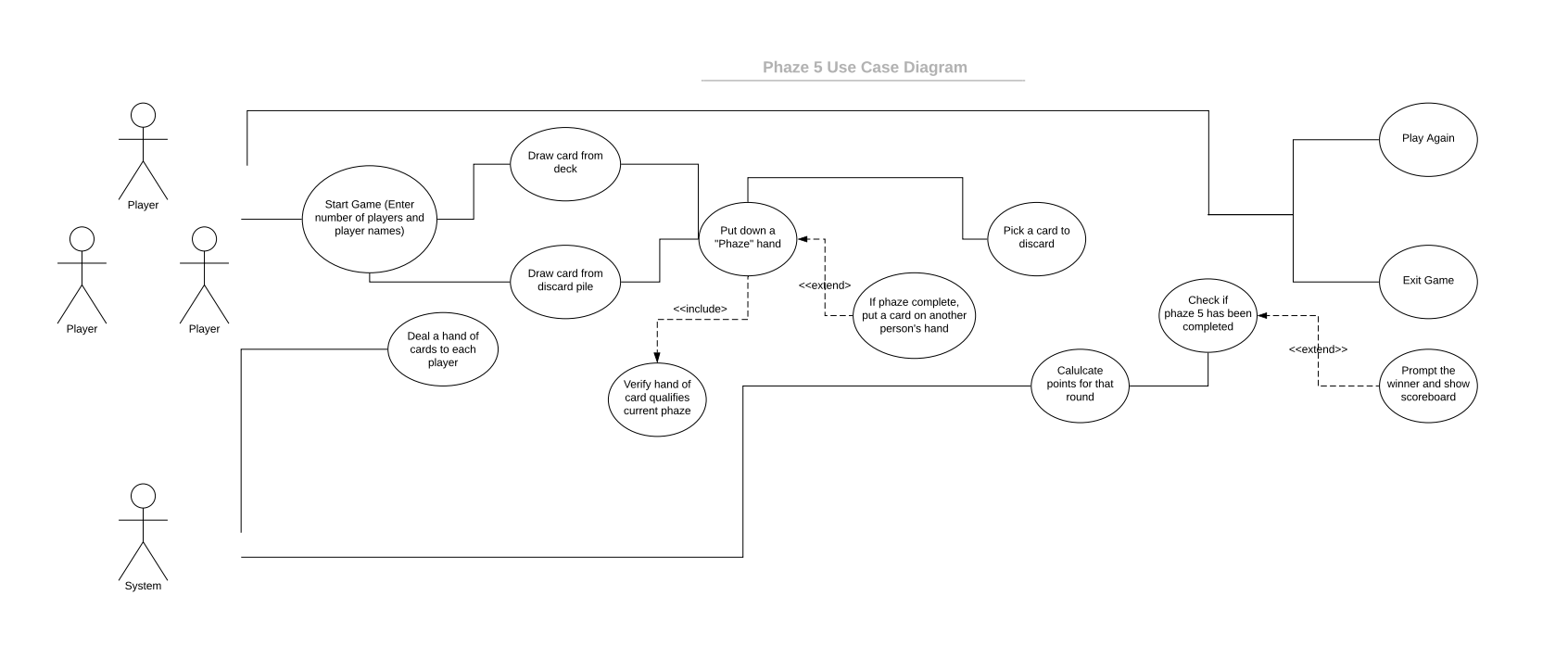
Another functional requirement for the frontend is (4) allowing the user to choose how many players that will be playing a game and what each of their usernames will be. This is currently a high priority because to play the game it is necessary to know how many players there are and what names, so that we can differentiate between each player.

# **3.** **Non-functional Requirements (10 points)**

The software quality and performance must be stable enough to support the multiplayer game in its entirety. The system must be reliable enough to store player information, such as the running score against each player and a state indicating what phase each player is on in any given round. Safety and security are not requirements for our game, as it does not contain any personal information of players. In addition, the state must be kept on the frontend, so we can see if someone changes their mind about their decision when making a move or if someone puts a new value in for the number of players and their usernames.

**4.** **Use Case Diagram (10 points)**

Link to PDF version of diagram: <https://www.lucidchart.com/publicSegments/view/bb20ea11-da9a-4a5b-aa11-6dc4c68ffb69/image.pdf>



**5.** **Class Diagram and/or Sequence Diagrams (15 points)**

**

# **6.** **Operating Environment (5 points)**

If we fail to produce the intended program, our fall-back plan is to have our game run locally on one machine, likely a Mac computer, which should also support our program and allow it to run with no problems.

# 

# **7.** **Assumptions and Dependencies (5 points)**

There are a lot of assumed factors that could affect the end result of the project. The implementation of a web-based game application was an idea that everyone agreed was an interesting and involving choice, but was also a barrier for everyone due to the inclusion of multiple tasks and subjects (like the knowledge of the language JavaScript) within the application that some of us have never done or have touched on before. This then led us to put in a lot of individual research for this project, allowing us all to come together to work on our separate pieces of the project. Our main assumed factor as of now is the concern of how long the implementation of the deployment of a lobby for the web app would take. As much as the other factors, like a few of us not being familiar with JavaScript, are only a minor barrier, we are aware that the concept of a lobby could potentially arise some issues with completing the project in time. Another assumed factor is human resource availability; as we are all full-time college students, we sometimes face the issue of pressured time frames. Additionally, another issue to keep in mind is the lack of source code access, due to only some of the group members having access to the GitHub repository, potentially leading to mixed code or simply some sections of code being left out. However, another assumed factor of prioritizing the important functions has been minimized due to our new knowledge of which functions are the main ones needed for the program. But without these functions, the majority of the program cannot function leading to the potential issue of either being unable to complete these functions in time or being unable to work in general. Some mandatory dependencies are some things mentioned earlier within the assumed factors, being the relevant software backend functions. These are time-dependent, due to without these functions we cannot start the frontend view of the game with its respectable connections to backend functions. This is important for the majority of the project, therefore leaving the discretionary dependencies to be the CSS and HTML code for the user interface, since it is not as time dependent. Although we do not reuse code from another project, we do consistently turn to the use of the internet for explanations of certain topics within the project that we are not fully knowledgeable of.