Virtual Board Game Engine

Seng 499

Project Group 37

University of Victoria

Group ID: 37

Faculty supervisor: Dr. Miguel Nacenta

Co-supervisor (if any):

Team Information:

S. No.	Name	V Number
	Juan Cormish	V00839267
	Oleg Matveenko	V00844836
	Michail Roesli	V00853253
	Emily Sluis	V00865746

Table of Contents

Table of Figures

I Introduction

Playing games are a good way to pass the time, connect with family, and develop better cognitive skills as we grow up. Due to the recent COVID-19 Pandemic, people are staying home and avoiding contact with family and friends to reduce potential exposure. Therefore, people aren't able to meet up for an evening of card games around a kitchen table. Furthermore, many people have their own house rules, which may not be available online. This means that we may not be able to play games to pass the time with those we enjoy spending time with. To address this problem, we chose to create an online board game engine so that people can play fully customizable board games with unique house rules they wish from the comfort and safety of their own homes.

The goal of this project is to design a solution that allows friends and families to gather around a virtual space while apart and play a regular card game or their own niche version. The UI and UX design should be simple and intuitive, and the product should be accessible to anyone with a device. In the short term, our goal is to create a minimum viable product that allows users to play and build their own board games.

We will accomplish this by creating a board game engine where rules are enforced by the players like in person rather than creating the rules ourselves. By not creating the rules ourselves, we Simplify the process of the user designing their board game on our platform. In our initial implementation, we wish to provide users a solution to a subset of board games such as card games so that we can focus on providing a good user interface and experience. In future iterations, we can expand on providing users with board customization so that users have more options to choose from.

II Literature Survey & Specifications

There exist several solutions to what we have proposed. One of the most popular ones is called Tabletop simulator which is a game that is offered through the Steam game client by Valve. They average 16k users concurrently [1] and is offered as a paid game on steam. They include ingame physics and allow users to create their own games the way they want to by moving customizable virtual 3d pieces. Some games that can be played include chess, poker, jigsaw, puzzles, dominoes, and mahjong. One of the reasons we found this style to be appealing is how well it tries to simulate real world games through its 3d physics with the ability to even flip the board [2].

Another solution is playingcards.io which focuses more on the 2d experience but instead of being offered as a game through a game client, it is available to anyone with internet access on the web. Template board games include checkers, chess, crazy eights, go fish, hearts, and match up. They also offer the capability to create custom games and move pieces at will to simulate the real experience. One aspect that we liked about this solution is that they allow the user to edit the board and provide constraints on where cards can be placed on the board to provide some level of organization and also better usability. We found that this could enhance the real world experience by using things that a computer can provide that cannot be in the real world. Although it provided some useful features, we found that it had some minor missing features.

III Team Duties & Project Planning

In this section we'll describe the roles that exist for the project and to which deliverables they may pertain. As outlined in our original project proposal, we've assigned several roles to our group members. The roles were assigned as shown below.

- Front-end designer game play - Emily
- Front-end designer game design - Juan
- Back-end for media management (user uploaded cards, images) – Emily
- Back-end designer for general functionality – Misha, Juan
- Authentication/user designer – Oleg
- UML diagram manager – Juan
- Environment and maintenance specialist – Oleg
- Dev-Ops coordinator – Misha
- Faculty Sponsor Liaison – Emily
- Documentation Lead – Misha
- Database lead - Oleg

In creating our roles we considered some of the key milestones which we've identified to include:

1. Project Planning & Selection
2. Selecting a Stack & Environment
3. Design Review
4. App & UI Design
5. Minimum Viable Product Implementation
6. Prototype Improvements
7. Final Overview

Our first observation from these milestones was that to ensure that we're on the right track, we need to be able to keep in contact with our faculty supervisor. We found the best way to do this was to have one person responsible so that the supervisor is not overwhelmed with emails. We also identified that as we progress through this project, it is important that we document why we made certain design decisions, what was discussed in meetings and ensuring that we have instructions on how to execute our code so that it can then be shared in the future. In addition to documenting how to run the code, we thought that it would be valuable to document our design of the code using UML diagrams as this would show how our project is evolving over time, and ensure that the implementation is using the same frame of reference.

This leads to implementation roles; we decided to separate the project similarly to how it is done in larger organizations where we separate the backend, frontend and maintenance which provided us with the front-end, backend and dev-ops roles. In the frontend roles, we split it further to consider two main aspects of the game: playing and the game design. Similarly, for the backend we've split it into the backend for the media and images, the general backend concurrency tracking data, and the actual database management. We also thought that keeping track of users is a large enough deliverable to warrant a role for it as well, hence the authentication and user design role. Lastly, to ensure that all this development is done smoothly, we added an environment and maintenance specialist to ensure that the workflow and source code management is done correctly.

One of the potential bottlenecks that we identified is in the transition from the App & UI design to the MVP implementation phase. To be able to start working on the minimum viable product we need to have some sort of base infrastructure and initial design concepts. This highlights the importance of the MVP implementation milestone since it relies so much on the completion of these previous milestones. If changes are required to the stack later in development then this could result in a worse product as shown with the laws of software evolution; if changes are made later on in the product, the more costly this is in time, and money. To ensure that we are able to complete the expected deliverables for the bottleneck, we ensured that we had already implemented the base infrastructure for the selected stack, and we created the initial design as a group. As we iteratively continue designing the system we will distribute the work required for designing the different key features of the application. Each design is reviewed as a team to ensure there is no miscommunication or differing mental models to avoid requiring reimplementation of features. In the case that this strategy fails, then we will reevaluate the scope and minimum requirements for our MVP and focus on the key features we want to implement.

IV Milestone & Progress Made

To elaborate on our roles, we're going to break down the project into smaller tasks for each identified milestone and depict who is responsible (R), accountable (A), consulted (C), and informed (I) for each deliverable. It is important to note that most of the team will be expected to assist in various aspects of the project, however, we'll ensure that each member has a particular task that they are responsible for. Table 1 below depicts the RACI chart and how we plan on breaking down our tasks.

Table 1. RACI Chart Task Breakdown

Deliverable	Deadline	Juan	Oleg	Michail	Emily	Status
Project Planning & Selection	May 29th					✓
Explore project topics	May 13th	R	R, C	R, A	R, I	✓
Exploring existing solution to selected project	May 15th	R	A	C	I	✓
Distribution of Roles	May 21th	I	C	A	R	✓
Exploring work log apps	May 20th	C	A	R	I	✓
Create project draft sketches	May 27th	I	R	A	C	✓
Find a faculty supervisor	May 29th	I	C	A	R	✓
Create Project Proposal	May 29th	R, I	R	R, A	R, C	✓
Selecting a Stack & Environment	June 4th					✓
Select Backend	June 3rd	R	R,C	R	A,I	✓
Select Front-End	June 4th	C	A	I	R	✓
Select CI Tools	June 4th	C	I	R	A	✓
Create Github Repository	June 4th	C	I	R	A	✓
Design Review	June 15th					✓
Create Virtual Environment for Development	June 10th	C	A	R	I	
Ensure base infrastructure works	June 13th	R	C	R	I, A	✓
UML diagrams	June 14th	R	I	C	A	✓

App and UI Design	June 29th					✓
Create Initial Design Sketch	June 26th	C, I	C, I	C, A	R	✓
Examine Improvements	June 29th	R	I	A	C	✓
MVP Implementation	July 15					
Create first working prototype	July 13	I	R	A	C	
Test prototype	July 14	R	A	C	I	
Examine Improvements	July 15	A	C	I	R	
Prototype Improvements	July 25th					
Select major improvements for final copy	July 16th	R	I	A	C	
Complete improvements	July 25th	A	R	I	C	
Final Overview	July 31st					
Finalize documentation	July 27th	C	I	R	A	
Gather Images	July 27th	R	A	C	I	
Create Poster	July 28th	A	C	I	R	
Create Presentation	July 29th	C	A	I	R	
Edit presentation recording	July 30th	R	A	C	I	
Create Report	July 31st	A	I	R	C	
Hand in Poster/ Presentation & Report	July 31st	A, I	A	R	A, C	

Below we'll discuss each of the milestones listed above and what will or has been done in more detail.

IV.I Project Planning & Selection

After assembling a team of four software engineering students we primarily considered software based projects. Our initial ideas were a plant monitor; a website for 3d designers, 3d printer owners, and clients to connect and set up three-way transactions; and our final choice of a way to play and create board games online. This project was evaluated to be most feasible and most relevant to the current circumstances of playing board games with COVID-19.

IV.II Selecting a Stack & Environment

Our first decision to make was where to host our project, for simplicity and accessibility we decided to go with a web interface, this way anyone with a device can use our platform. Our development will still primarily consider users on large screen devices such as laptops and tablets.

The selection for the software stack was made based on the team's previous experience with web development. We used Python with Django for our backend, and HTML with Vue and CSS for our front end. Python with Django is a simple way to host a website and use the python language for handling the backend, we decided on this because Python is easy to learn and some of us have experience using it for web development. Vue is a platform for building user interaction on the front end of a website that none of the group members have used but it is fairly popular in the industry and makes sense for our application.

To handle the communication between the Vue frontend and the Django REST backend we decided on using Axios. The choice of Axios [3] over standard JQuery with Ajax was an extension of the decision to use Vue. It works well with Vue and provides automatic transforms for JSON data which we found to be valuable for sending our JSON data for user actions. Vue recommends the use of Axios as the current standard [4][5]. Further research into examples of using the combination of Django, Vue, and Axios was done to ensure we had enough instructional resources on its implementation.

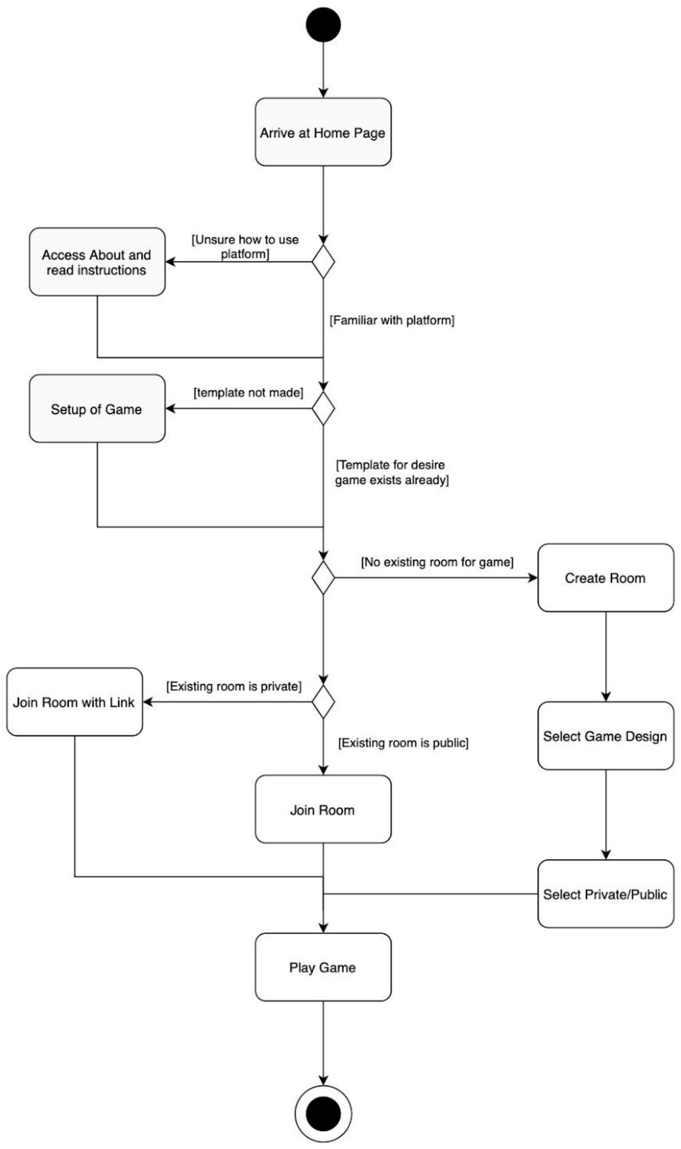
Before we finalized our decision to use Axios we created a proof of concept prototype that uses Axios to handle a HTTP request and display the information of the request in a Vue webpage, it was added to the stack. To make HTTP requests easier while not adding an extensive library to the requirements of the application we decided to install Axios inline.

The main environment we chose to use to collaborate distributively is Github. In Github we are able to add plugins that will allow us to keep track of our integration and deployment status using tools like TravisCI or CircleCI, and keeping track of code quality using tools like Code Climate. The workflow that we chose to use alongside these tools is Github flow since it is a simple workflow that will allow us to iterate on our design and deploy more quickly [6]. In our git repository we added a virtualenv to provide easier initialization with our required packages but we later realized that all it provided was an empty virtual environment and we'd have to install the packages regardless. We then moved to creating a requirements.txt file and moved all the required packages as a list there that can then be downloaded using pip install.

IV.III Design Review

Since our project is primarily a human computer interaction problem we decided we needed a professor who specializes in this field. Our first choice was Dr. Nacenta who happened to be interested in our project. During our first meeting he helped iron out the details of the project such as excluding a rules engine and letting users settle rules disputes by a third party platform, our final design is made to be used in conjunction with a communication platform such as zoom or skype.

After initial discussions and planning of the core functionality of the application a series of Activity Diagrams were drafted as seen in figures 1-3. We used the flow of users through the application to highlight the necessary portions of the application that needed design sketches that will be discussed later.



Arrive at Home Page
[Unsure how to use platform]
Access About and read instructions
[Familiar with platform]
[template not made]
Setup of Game
[Template for desire game exists already]
[No existing room for game]
Create Room
[Existing room is private]
Join Room with Link
[Existing room is public]
Select Game Design
Join Room
Select Private/Public
Play Game

Figure 1. Overall Activity Behaviour UML Diagram Version 1

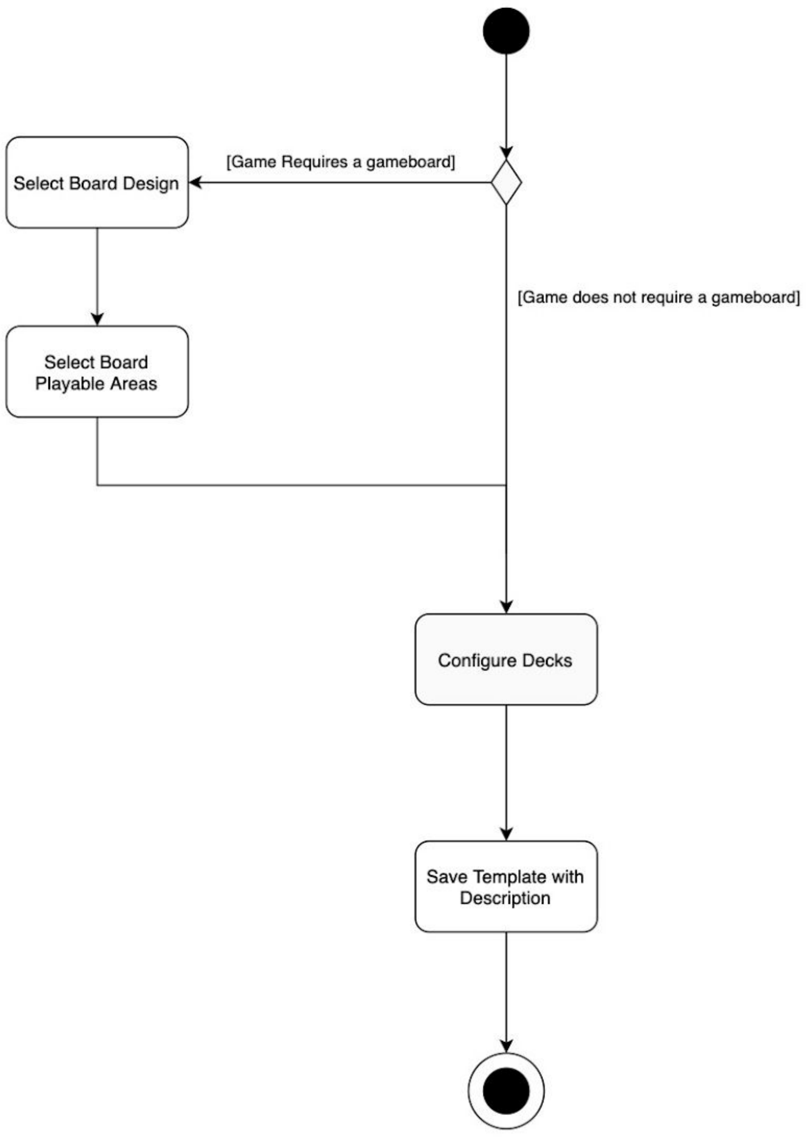

[Game Requires a gameboard]
Select Board Design
Select Board Playable Areas
[Game does not require a gameboard]
Configure Decks
Save Template with Description

Figure 2. Game Design Activity Behaviour UML Diagram Version 1

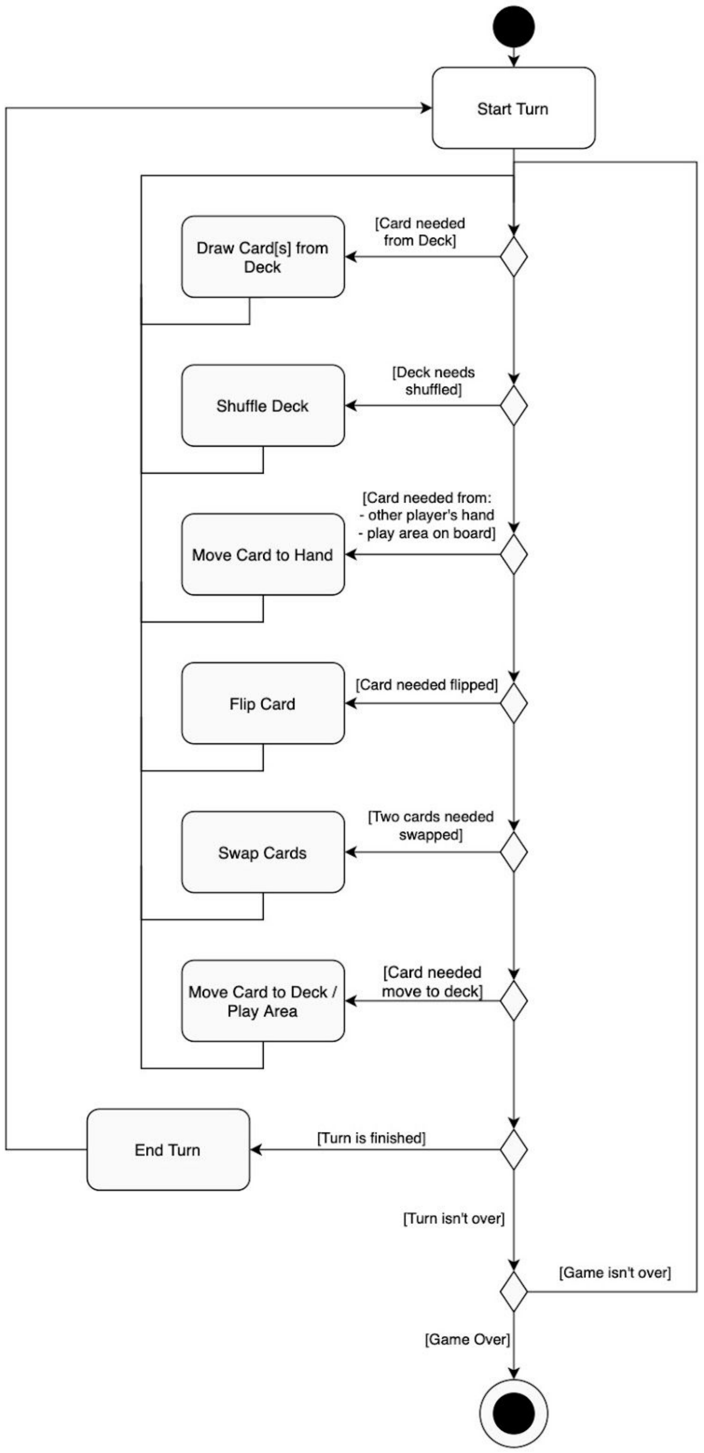


Figure 3. Gameplay Activity Behaviour UML Diagram Version 1

With the hopes of beginning implementation early an initial class diagram was also drafted as seen in figure 4. Key considerations included in this diagram was the main design choice of open synchronous gameplay versus constrained turn based gameplay. In the initial design the focus was on more constrained gameplay and led to the concept of a tiles object on the gameboard that would behave both as play areas and as “Deck”s as referred to in figure 4.

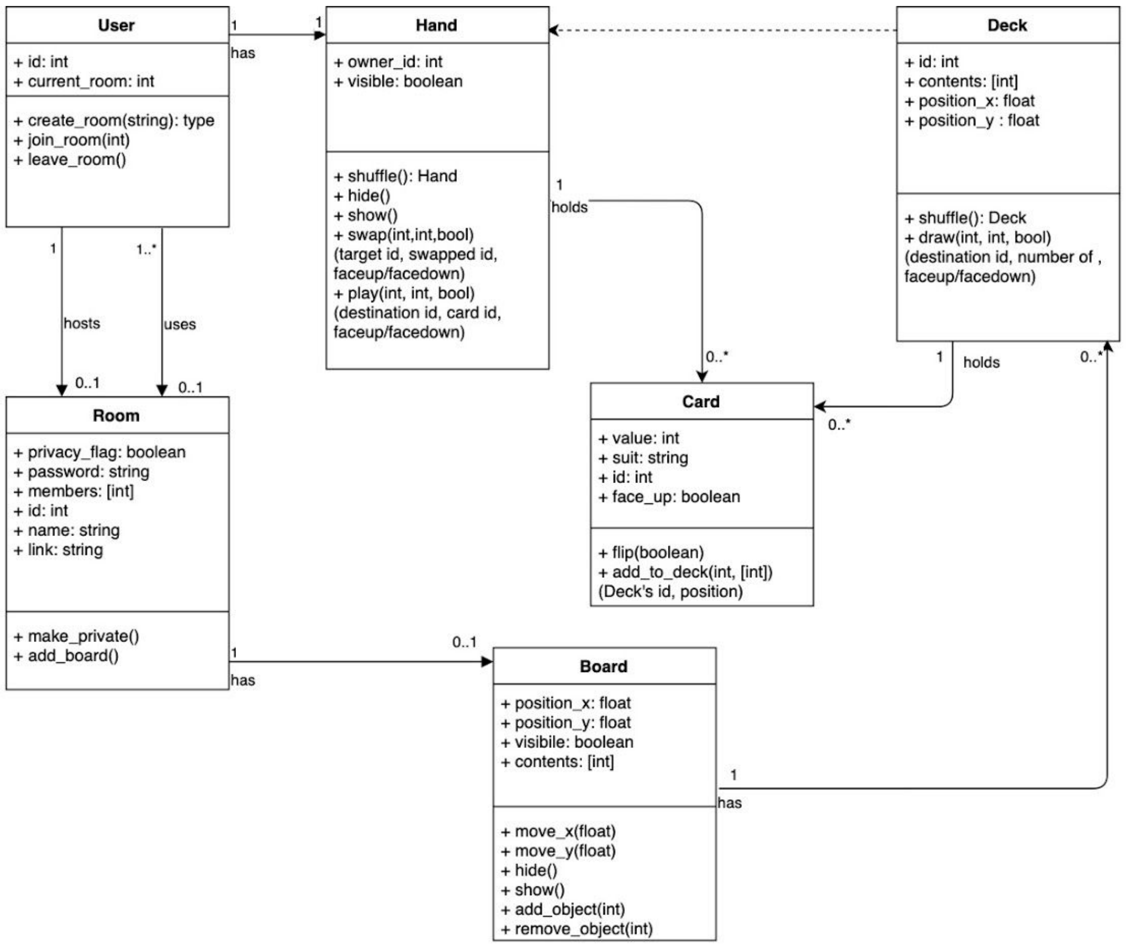


Figure 4. Card Game Class Structure UML Diagram Version 1

IV.IV App and UI Design

After feedback from Dr. Nacenta further considerations were put into iteratively designing the UI and a series of low fidelity prototypes were created. Every member of the design team made their own set of sketches which were combined in the first round of designs to form the first design as seen in figure 5 and was sent to Dr. Nacenta. Higher quality photos of each individual page on the application can be found in Appendix A.

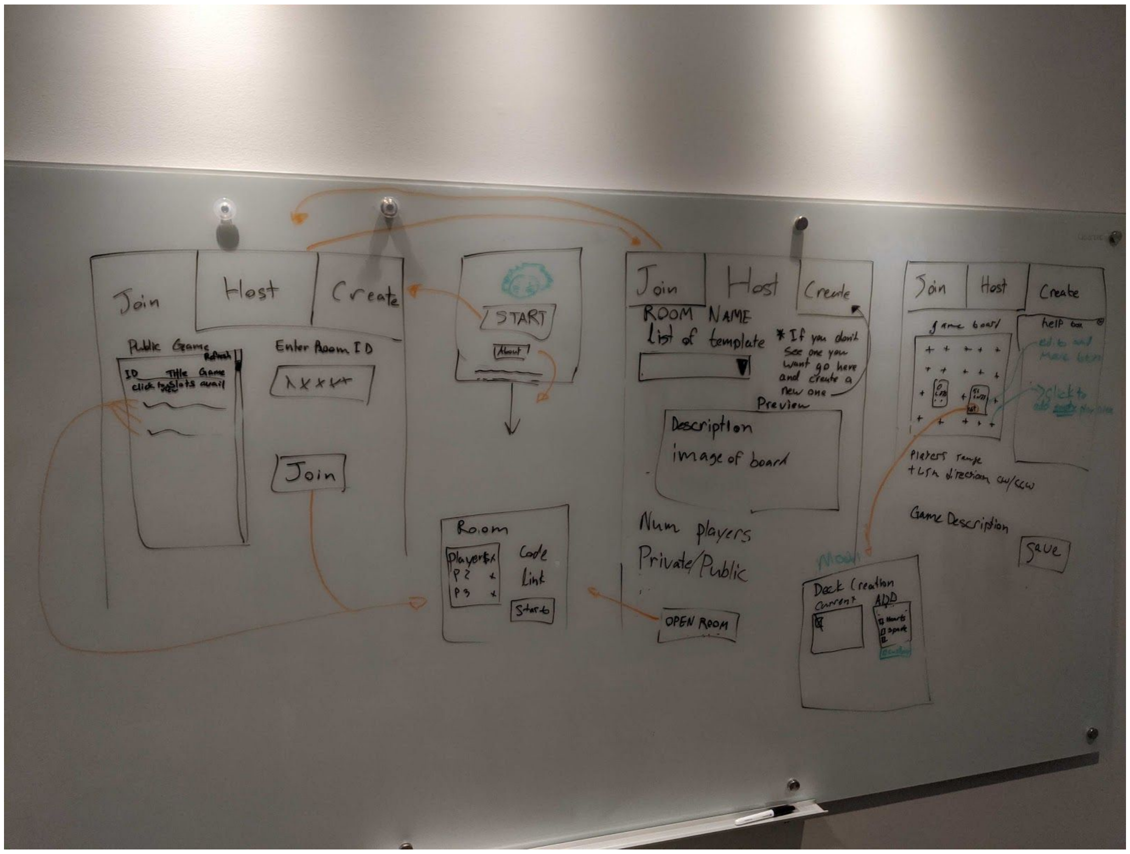


Figure 5. Full Overview UI Sketch

Feedback from this set of design sketches is currently under consideration.

V Summary & Future Work

The goal of this project is to develop a minimum viable product of an online board game engine. Ultimately, the app will offer a customizable, virtual game space to allow users to join their family and friends online to play any of their favourite card games, no matter how niche their rules are.

At this point in the project, we spent a large portion of our work time in design and planning, which should set us up for success. Through thoughtful project selection and roll assignments we have ensured that the project will have a positive social impact in the era of quarantine, that it is of interest to all team members, and that the work can be evenly distributed amongst members.

Team consultation led to careful selection of the tool in our stack. We also ensure that every piece of technology we are using is familiar to at least one member of the team, or one or two members were assigned to that piece to learn it thoroughly enough to take the lead with it.

We have spent a large portion of our work time on an iterative design process to explore the UI and UX possibilities of our platform. Through many iterations of low-fidelity and hand-drawn wireframes, we were able to design a clean and intuitive UI. These designs will be incorporated into higher-fidelity mocks in the next few weeks and then will be implemented to the front end of the platform.

We have implemented the base infrastructure using django, postgres, html and vue, and are able to deploy a prototype version of our web platform. Basic functionality such as database calls and url navigation have been implemented as well. Each team member has successfully set up their local environments and are able to contribute to the code base.

Our next steps are to complete the mocks and build a high fidelity prototype using UI software. We will then implement the logic and designs demonstrated in those mocks. Steps beyond that will be determined in an interactive process as preliminary testing and demonstrations will lead to new requirements throughout the process.

References

[1] *steamdb*, 2020. [Online]. Available: https://steamdb.info/app/286160/graphs/. [Accessed: 29- Jun- 2020].

[2] T. Simulator, "Tabletop Simulator on Steam", *Store.steampowered.com*, 2020. [Online]. Available: https://store.steampowered.com/app/286160/Tabletop_Simulator/. [Accessed: 29- Jun- 2020].

[3] "axios/axios", *GitHub*, 2020. [Online]. Available: https://github.com/axios/axios. [Accessed: 29- Jun- 2020].

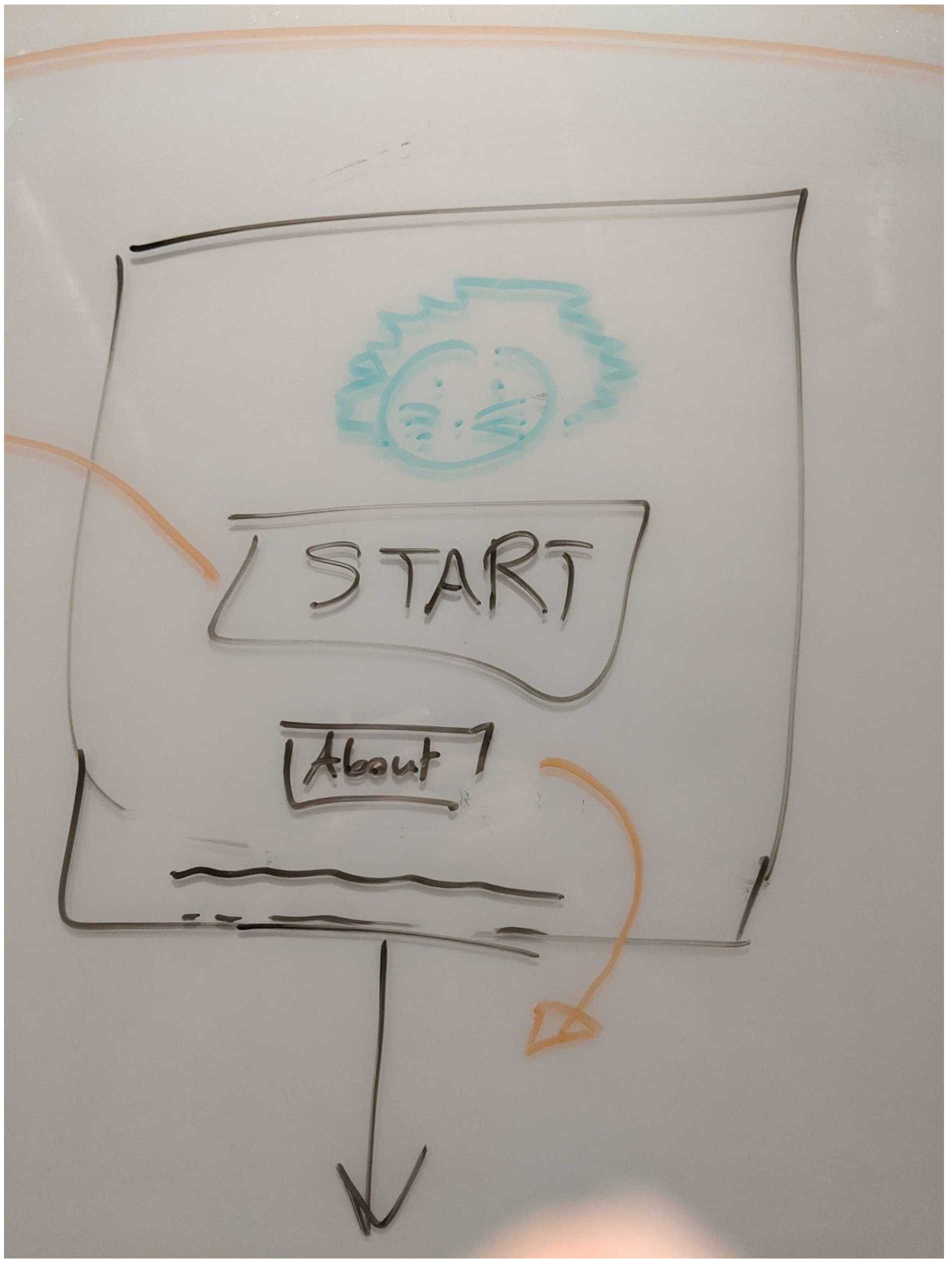
[4] "Using Axios to Consume APIs", *Vuejs.org*, 2020. [Online]. Available: https://vuejs.org/v2/cookbook/using-axios-to-consume-apis.html. [Accessed: 30- Jun- 2020].

[5] S. Triepels, "How to use Vue.JS and Django", *DjangoWaves*, 2020. [Online]. Available: https://djangowaves.com/tutorial/how-to-use-vue-and-django/#2-use-vuejs-in-django-templates. [Accessed: 29- Jun- 2020].

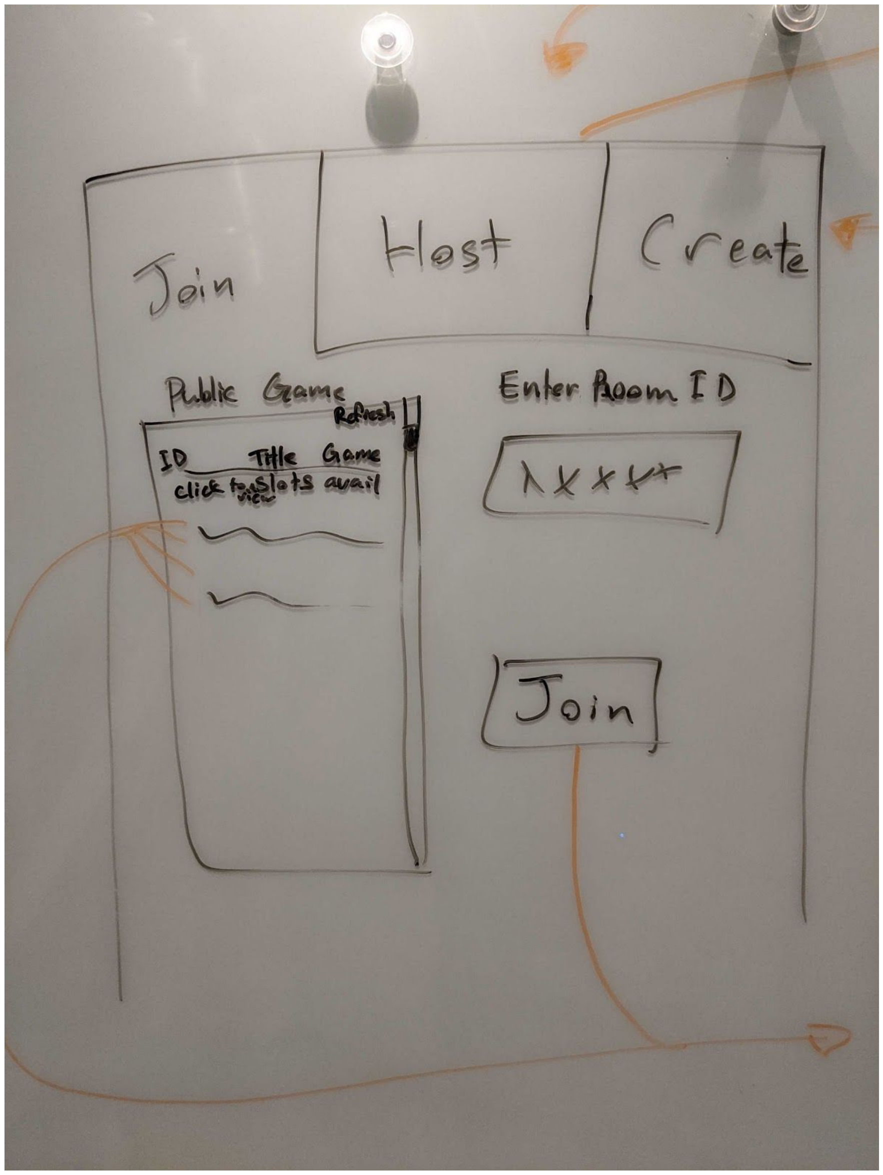
[6] "Understanding the GitHub flow · GitHub Guides", *Guides.github.com*, 2017. [Online]. Available: https://guides.github.com/introduction/flow/. [Accessed: 29- Jun- 2020].

Appendix A: Design Sketches

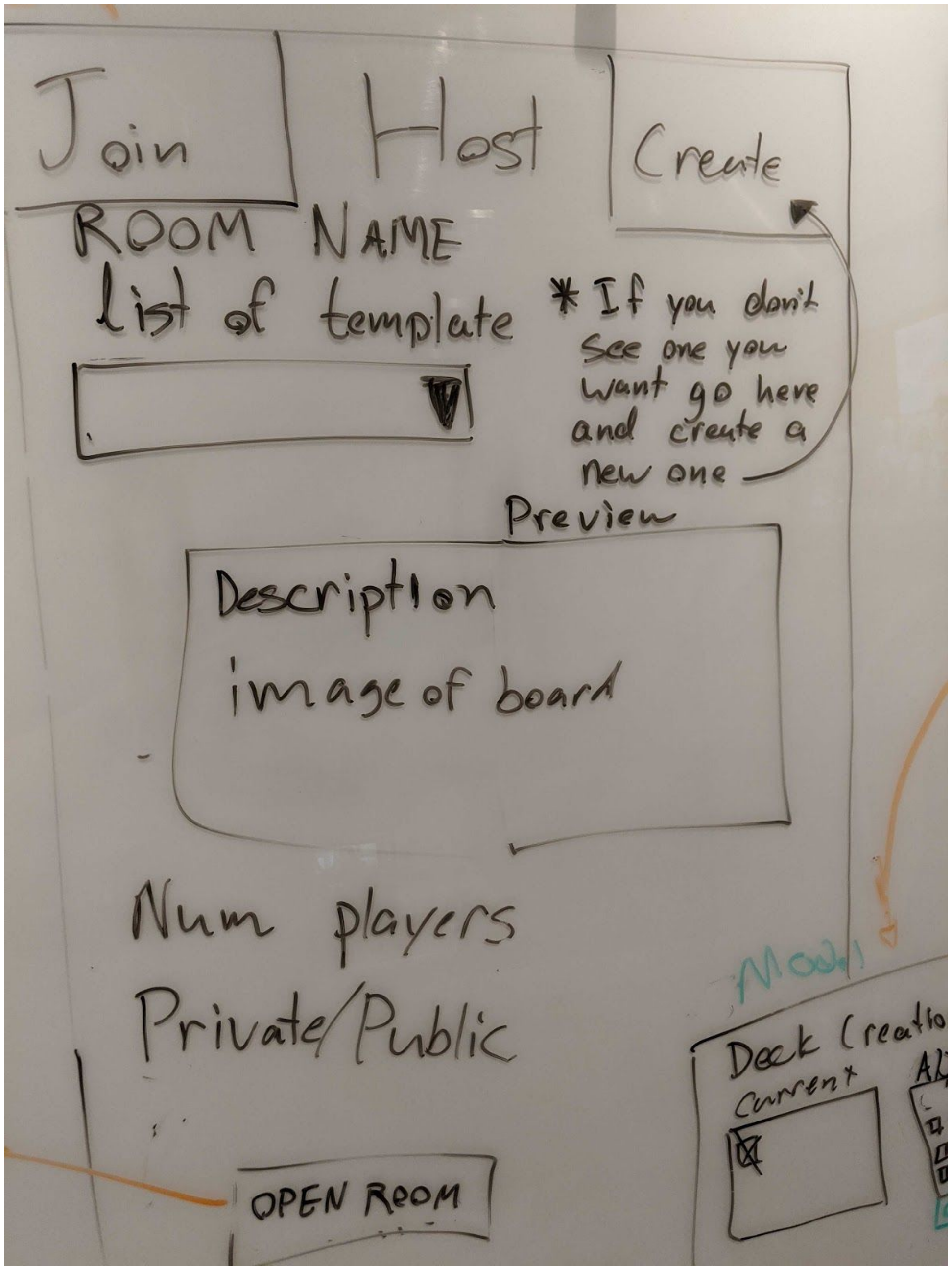
Start



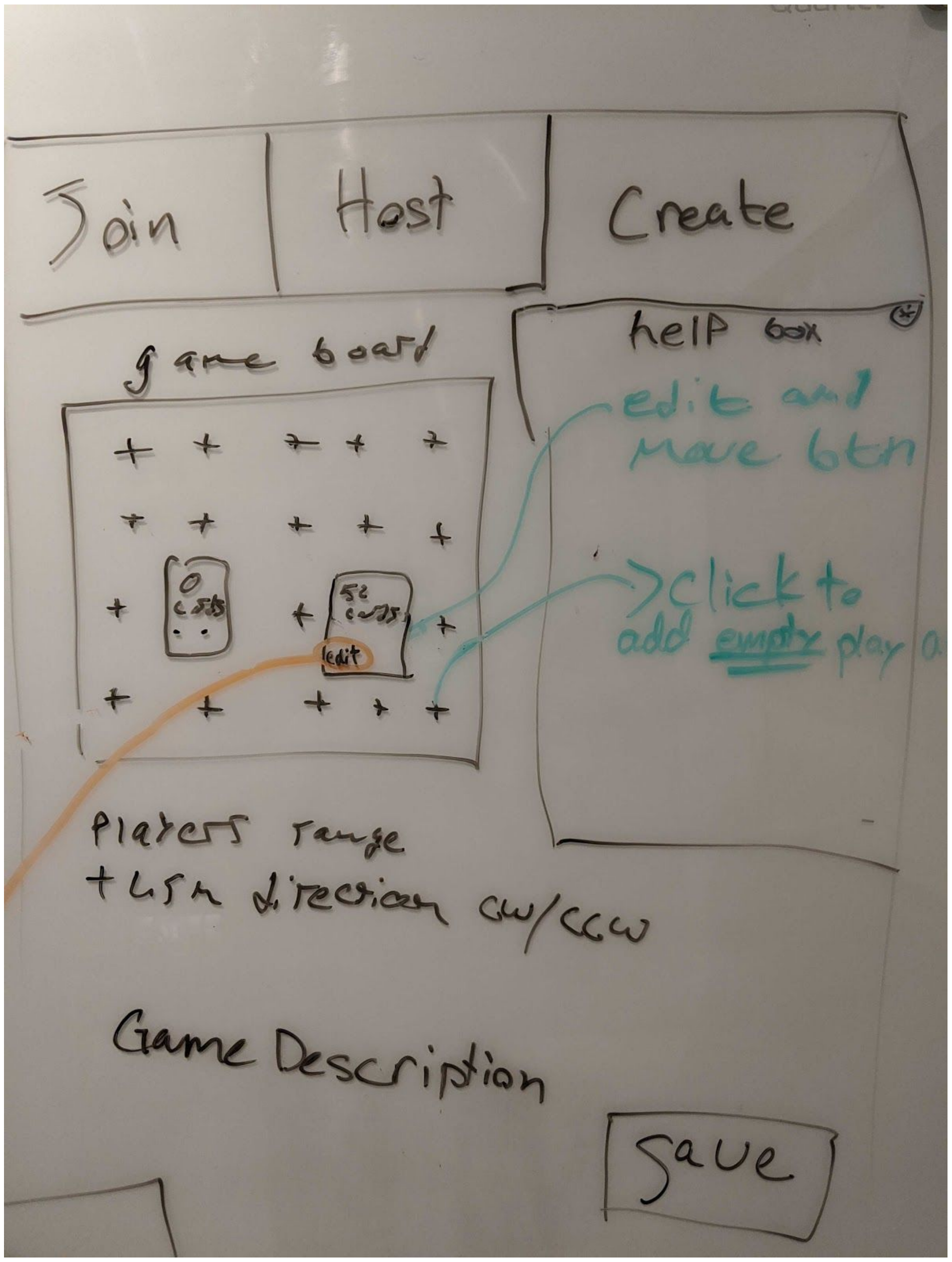
Join


Join
Host
Create
Public Game
Refresh
ID Title Game
click to view Slots avail
Enter Room ID
XXXXX
Join

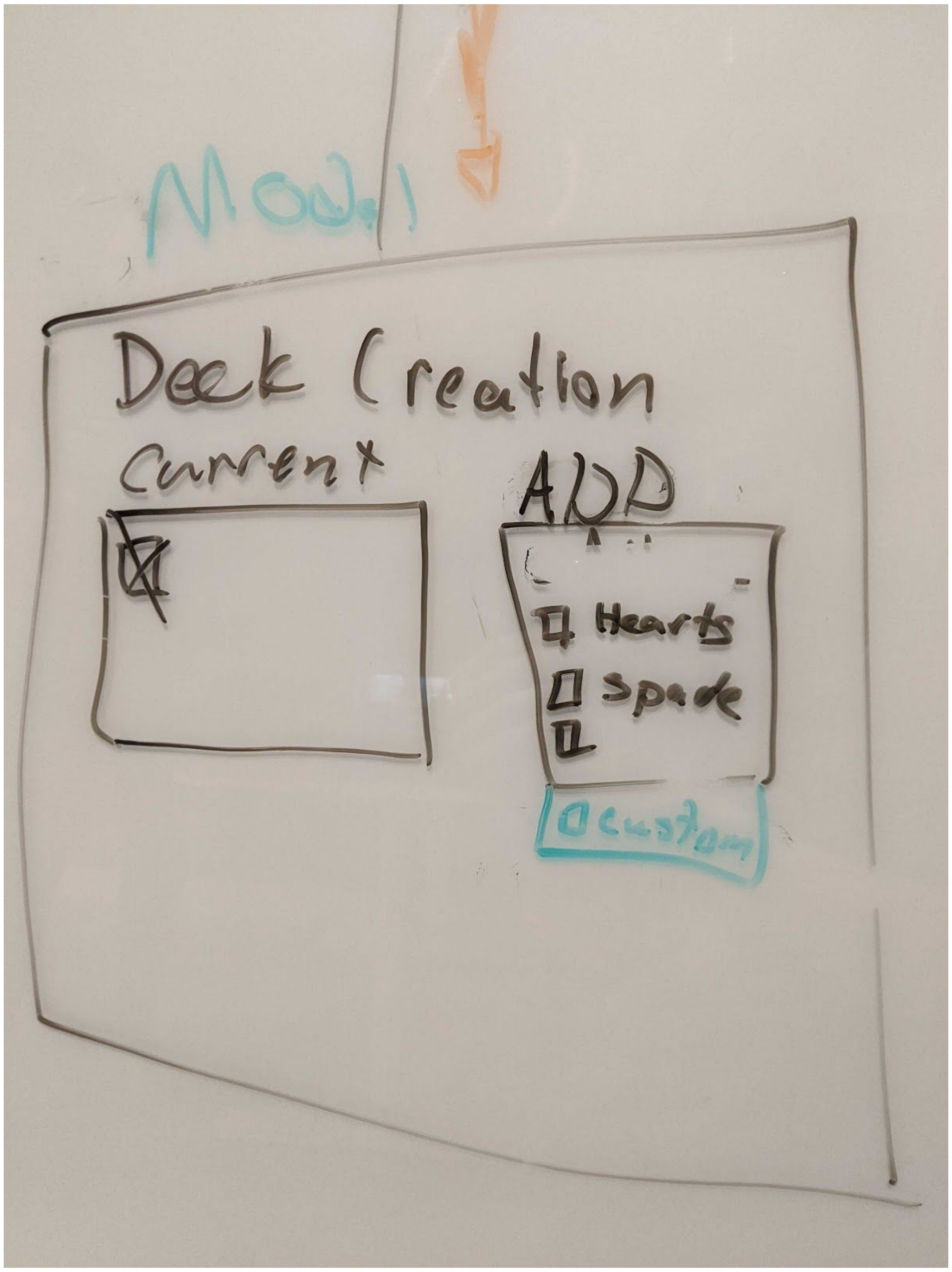
Host


Join
Host
Create
ROOM NAME
list of template
*If you don't see one you want go here and create a new one
Preview
Description
image of board
Num players
Private/Public
OPEN ROOM
Modal
Deck Creatio
Current
Al

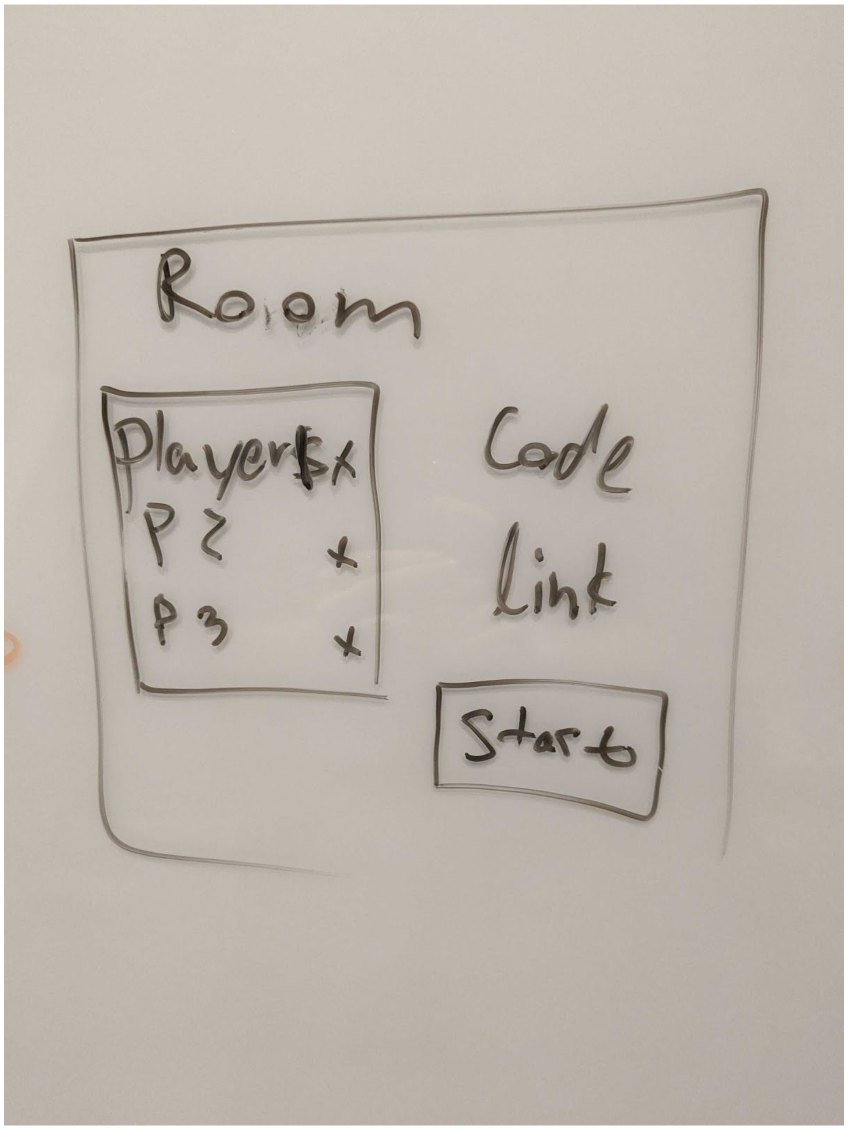
Create


Join
Host
Create
game board
help box
edit and Move btn
0 cards
52 cards
edit
click to add empty play
players range
turn direction cw/ccw
Game Description
Save

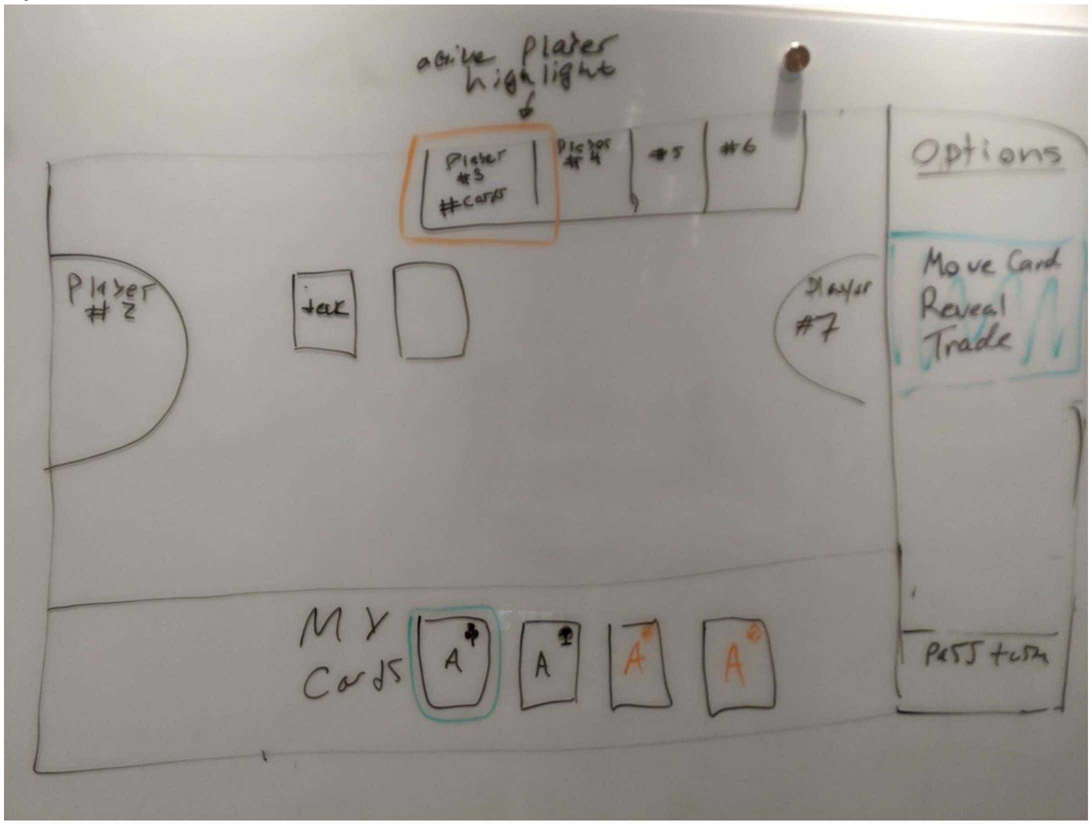
Deck Creation



Room



Play



active Player highlight
Player #3 #cards
Player #4
#5
#6
Options
Player #2
Player #7
Move Card
Reveal
Trade
My Cards
A
A
A
A
Pass turn