83

2. Submit to Moodle the link to your GitHub repository to mark your delivery.

Evaluation Rubric (out of 50 points)

- R1 Architecture section inadequate or missing: -10 pts
- R2 Technology section inadequate or missing: -10 pts
- R3 Data representation description is inadequate or missing: -10 pts
- R4 Coding Standards section inadequate or missing: -10 pts
- R5 (as promised) Over Inflated Story Point Estimations: -5 pts for over inflated story points. Any over inflation must be fixed before the next group assignment or incur additional penalties.
- R6 `pdflatex` fails: -10 pts
- R7 No GitHub Repository: -50 pts.

R1 no $ named architecture
mvc under design
-5 main modules missing

R2
fw ✓
-2. frameworks missing
DB ✓
testing ✓

R3
-1 ER dia
-4 description of main Table/Structures

✓ R4
test 20%
cC

✓ R5
✓ R6
R7
-2 what is there is very good. I think it is incomplete, check your stories
R8
-3 create acct, admin screens, others?

ASN2

William Grimmer, Jack Formato, Victoria Shelton

October 2024

1 Design

For our design we chose to do a layered architecture with a three tiered architecture model. It will have the GUI as the top layer, the application as the middle layer and the database as the bottom layer. We chose this as it works well with JavaScript and although MVC is commonly used we felt as if it would be easier to learn and understand layered architecture.

1.1 Language

For our language we will use JavaScript as we understand it is one of the premier languages for web applications and has tons of frameworks and libraries that support it such as Node and React. In addition Victoria has also worked with the language before which will help significantly with the learning curve.

1.2 Frameworks/Libraries

For our Frameworks and Libraries we will almost certainly use Node.js and React.js. Node.js is what allows for server side JavaScript hosting and will be necessary in order to create a web application. React.js helps to build user interfaces and the front end for web applications and will be very useful. For our testing framework we will be using Jest which is the newest and most common testing framework for JavaScript.

1.3 Database

For our Database we will use MySQL. We chose a SQL database over a noSQL version as Will and Jack are familiar with SQL from taking DBMS with Dr. Rocha. For that same reason we will use MySQL as our choice of SQL database as once again we are familiar with it.

Figure 1: The Entity Relations Diagram for SQL.

1.4 coding standards

For our coding standards we will follow Agile development and work iteratively in coding sprints. The goal is to prioritize working code and not commit anything that covers less than 70% of test cases. Additionally we will use CamelCase as a naming convention to ensure consistency throughout the code.

2 UML

Figure 2 is our UML Diagram. This illustrates the class relationships in our project. Users will have Messages and UserStats. We have an Admin class which will extend. Admins, unlike normal Users, will be able to change a User's password. The Game Class has a Host, Players and a Deck. A Player will have a user, in order to add balance and update stats. A deck will be composed of Cards which are dependent on the deck and only exists with it. Players will have a hand, which consists of cards. A game will also have bots. Just like Players, Bots have a hand. These relationships should allow the program to run smoothly and efficiently.

3 Website UI Mockup

The UI for the website should be simple and intuitive. For the main page (see Figure 3a), the user has four options: create a game, join a game, play the tutorial, or check their account. The account menu page (see Figure 3b) allows users to check their account information and update their password. The create game page in Figure 3c allows users to host a game, set a password, manage players, etc. Players can also join games through the join game page in Figure 3d. When players are in a game, they see the screen in Figure 3e. Lastly, when a game ends, players will see the end screen in Figure 3f, which shows a message based on whether they win or lose and allows them to return to the main screen.

a lobby is-a collection of games does that warrant a class?

class friend list

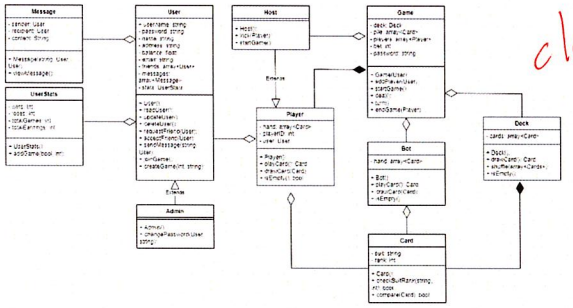
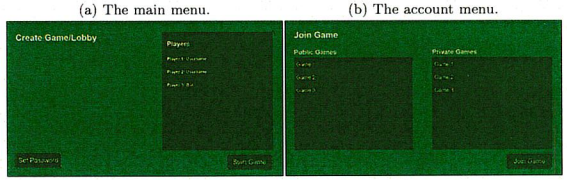
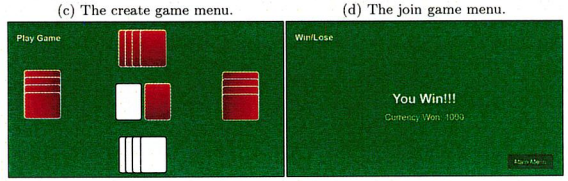


Figure 2: High-level OOA UML Class Diagram

(a) The main menu. (b) The account menu.



(c) The create game menu. (d) The join game menu.



(e) The game menu. (f) The end screen.