COM S/ SE 319: Construction of User Interfaces
Fall 2019
Group No. 55: Final Release Report

1. **Successful Implemented Story Cards for Final Release:**
    - Story Card 3:
        - Name of the Story: As a user, I would like to have different text cards for each round so that they aren't the same every time
        - Assigned Team Member: Emma Paskey
        - Tasks Accomplished For This Story Card:
            - Create a list of 30 keywords stored in a variable list
            - Randomly select a keyword/prompt and render on the game component
            - Ensure prompts will not repeat in game
            - Judge chooses prompt and passes to all other users

    - Story Card 6:
        - Name of the Story: As a user, I'd like to see the scoreboard as to see where I rank with other players
        - Assigned Team Member: Emma Paskey
        - Tasks Accomplished For This Story Card:
            - Create a scoreboard table listing all usernames in game
            - Add rounds won under each username
            - Update scoreboard values with judging choice
            - List usernames in order of most points to least

    - Story Card 7:
        - Name of the Story: As a user, I'd like to be able to receive a new card at the end of a turn so I have new giphs
        - Assigned Team Member: Jamie Sampson
        - Tasks Accomplished For This Story Card:
            - A button exists after judging is completed to go to next round
            - Reset game interface upon clicking next round
                - Render new giphs
                - Choose new judge
                - Reset selection logic

    - Story Card 14:
        - Name of the Story: As the judge, I'd like to select my favorite gif and end the round.
        - Assigned Team Member: Jamie Sampson
        - Tasks Accomplished For This Story Card:

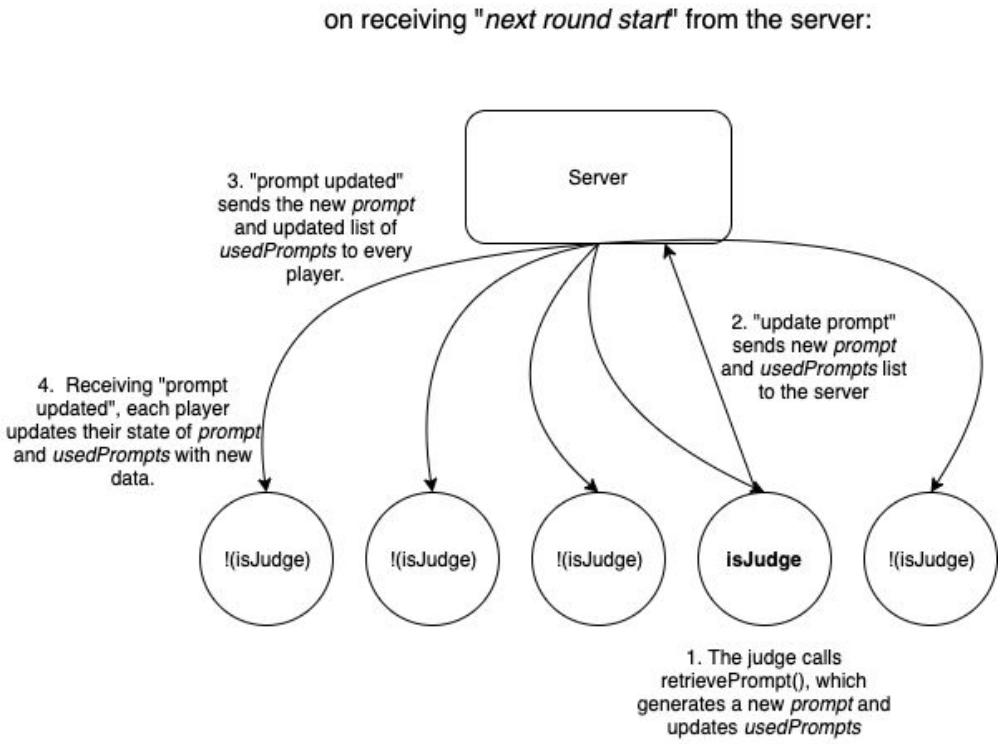
        - On submit by judge, send results to all other users
        - Allow collective cards to be selected by judge (only one similar to player's setup)
        - Show usernames on all cards for all clients
        - Pass judge to next player

- Story Card 16:
    - Name of the Story: Setup Firebase Hosting (Client)
    - Assigned Team Member: Jamie Sampson
    - Tasks Accomplished For This Story Card:
        - Setup firebase hosting
        - Ensure that application shows up on given url

- Story Card 17:
    - Name of the Story: As a user, I'd like to see the final scoreboard at the end of the game
    - Assigned Team Member: Emma Paskey
    - Tasks Accomplished For This Story Card:
        - Add cap of 10 rounds to a room
        - After the 10th round, "Next Round" button changes to "Final Results"
        - Clicking "Final Results" only shows the scoreboard and player stats
        - A button exists to go back to the home screen

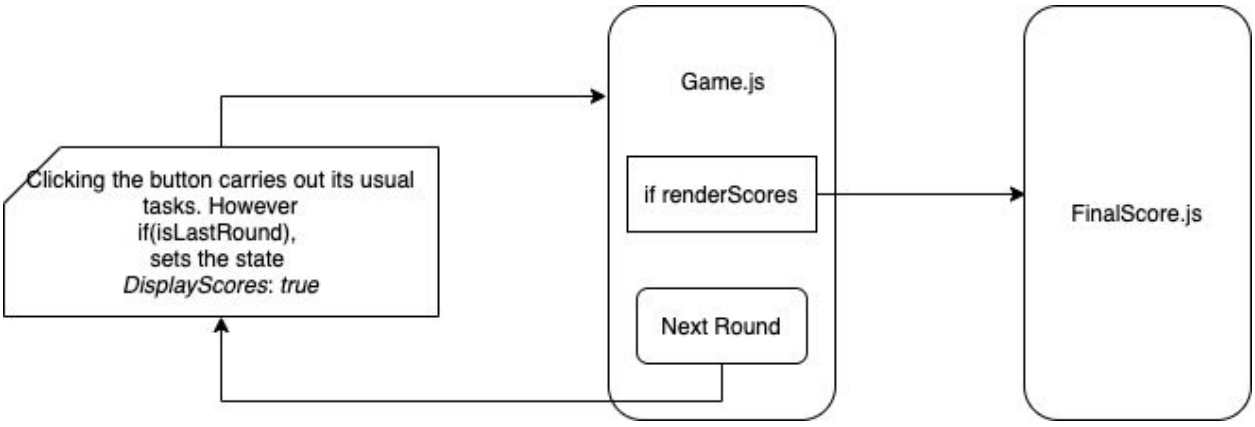
- Story Card 18:
    - Name of the Story: As admin, I'd like to host my application server off of localhost / allow other people to interact with server logic
    - Assigned Team Member: Jamie Sampson
    - Tasks Accomplished For This Story Card:
        - Server runs on one port
        - Install and make client side compatible with ngrok address

2. **<u>Design Documentation (UML Diagram for Story Cards):</u>**
    - *Figure 1 - Use case diagram for story card 3*



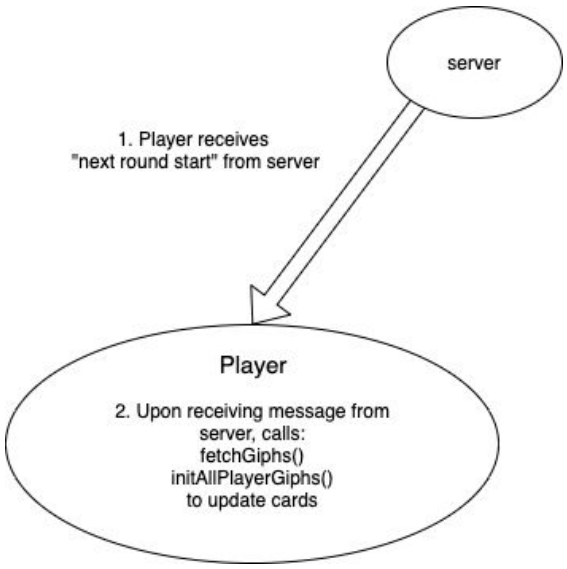
1 - "As a user, I would like to connect to a game so I can play on the server side."

- *Figure 2 - Use case diagram for story card 6*



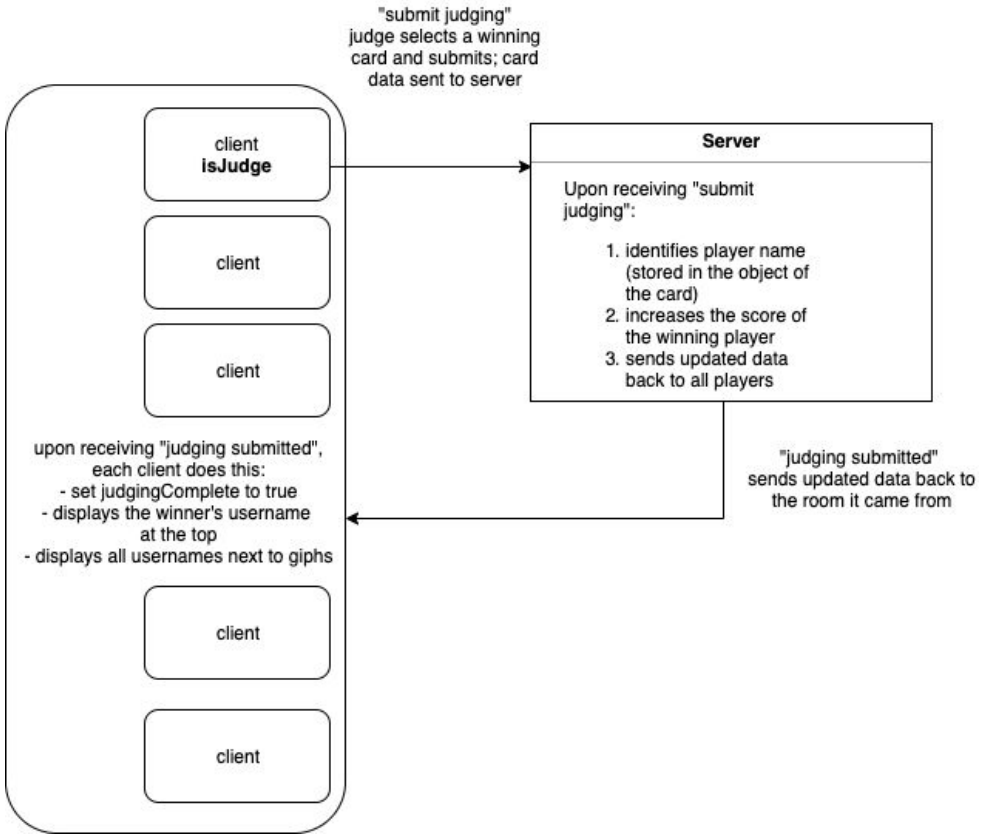
7 - "As a user, I would like to connect to a game so I can play on the server side."

- *Figure 3 - Use case diagram for story card 7*



7 - “As a user, I would like to connect to a game so I can play on the server side.”

- *Figure 2 - Use case diagram for story card 4*



1 - “As a user, I would like to connect to a game so I can play on the server side.”

3. **Implementation Outline:**
    - Platform
        i. Web
    - Front-End Language/Technologies/Framework
        i. JavaScript
        ii. ReactJS (framework)
        iii. Jest (testing)
        iv. Npm (task runners)
        v. CSS
        vi. HTML
    - Badk-End Language/Technologies/Framework
        i. NodeJS
        ii. ExpressJS (framework)
        iii. Socket.io
        iv. Jest (testing)
        v. Npm (package manager)
    - Database, Server, IDE and any design/UML tools:
        i. WebStorm & Microsoft Visual Studios (IDE)
        ii. Draw.io (UML tool)
        iii. Ngrok (server tunnel)
        iv. Firebase Hosting (web hosting)

4. **UI Description with Screenshots:**

Story Card 6: As a user, I'd like to see the scoreboard as to see where I rank with other players

The initial screen after starting a game. You can see the scoreboard with each player's username and their score at the top.

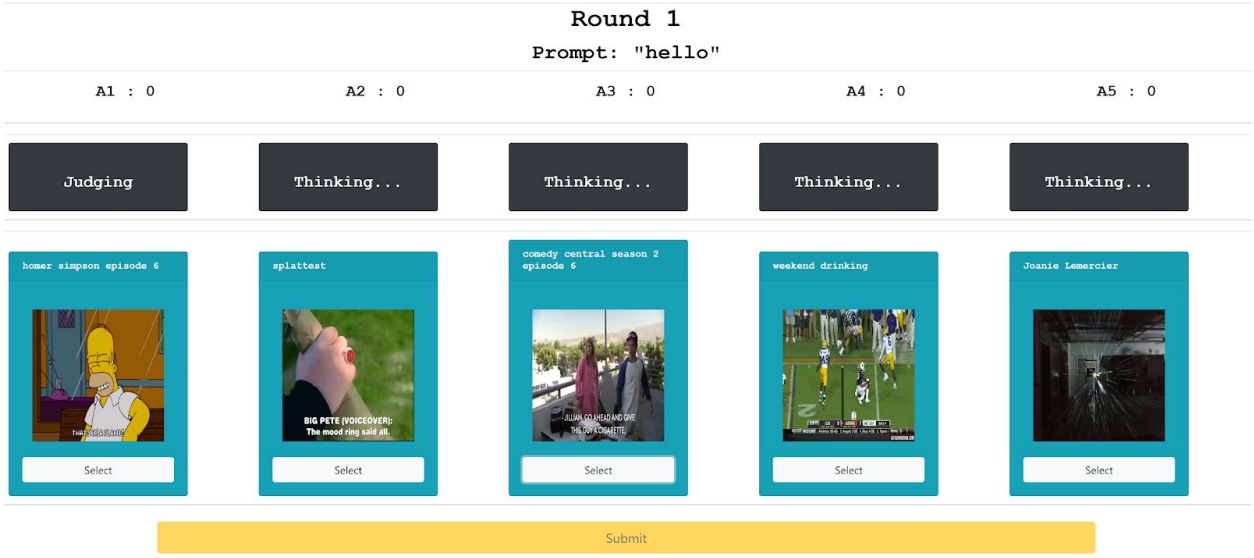


Figure 3: Story Card 6: As a user, I'd like to see the scoreboard as to see where I rank with other players

Story Card 14: As the judge, I'd like to select my favorite gif and end the round.

The judge's view as other players are selecting their cards. Same setup of being able to select and unselect a card before submitting. Can only submit upon selecting a card.

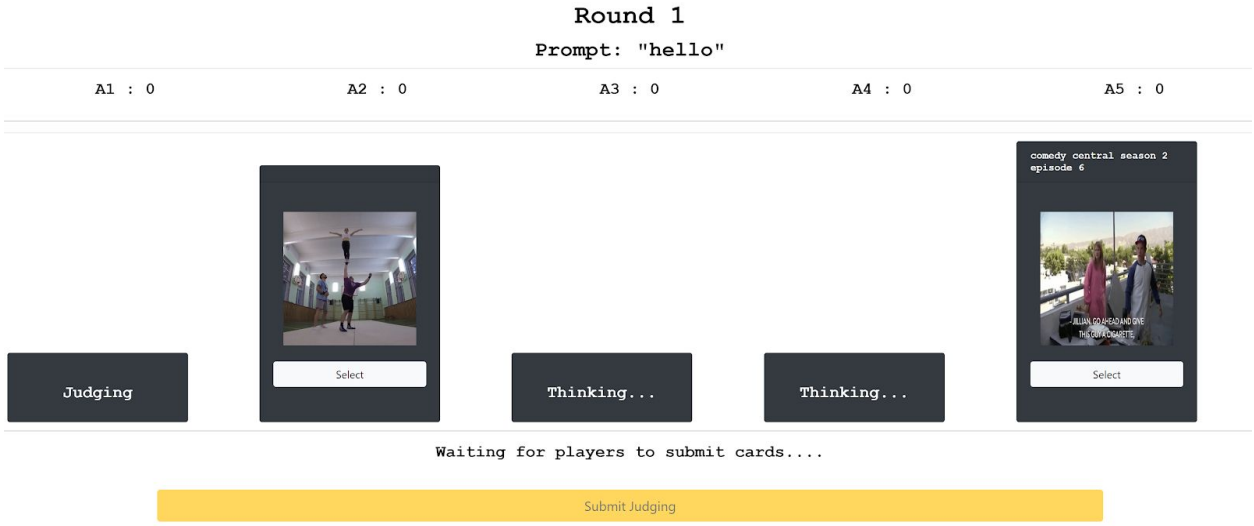


Figure 4: Story Card 14: As the judge, I'd like to select my favorite gif and end the round.

Story Card 14: As the judge, I'd like to select my favorite gif and end the round.

The view after the judge has submitted the winning card. It's highlighted in green. The winner's username is posted at the top, and the scoreboard is incremented by the choice.

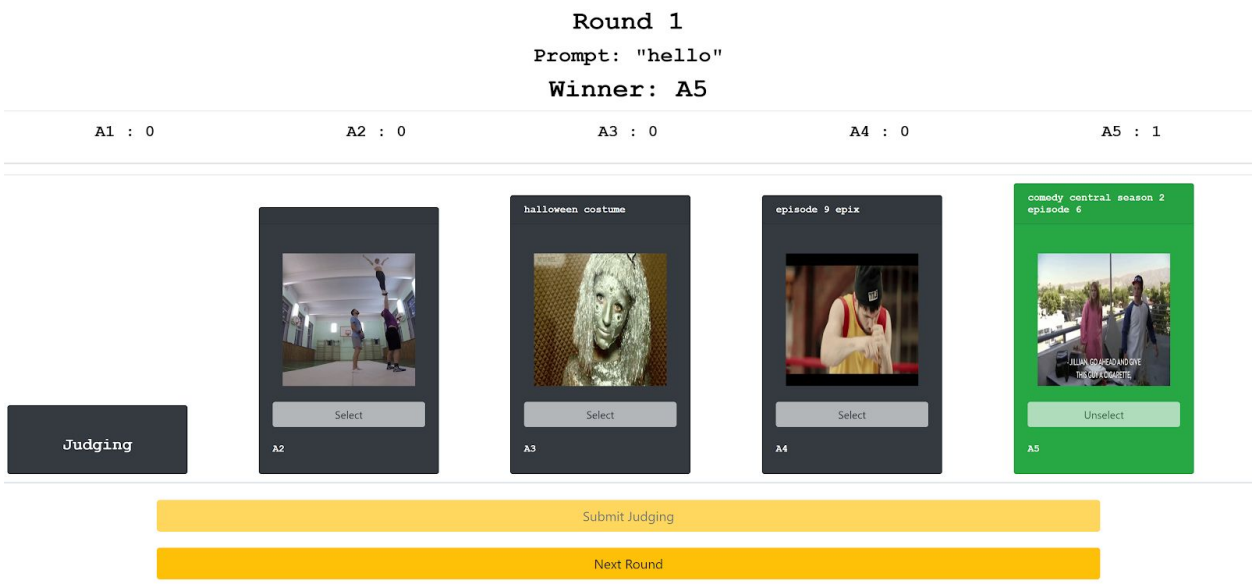


Figure 5: Story Card 14: As the judge, I'd like to select my favorite gif and end the round.

Story Card 7: As a user, I'd like to be able to receive a new card at the end of a turn so I have new giphs

Story Card 6: As a user, I'd like to see the scoreboard as to see where I rank with other players

Here you can see different giphs are rendered, the round is incremented, the prompt category has changed, and the screen has reset for this new round.

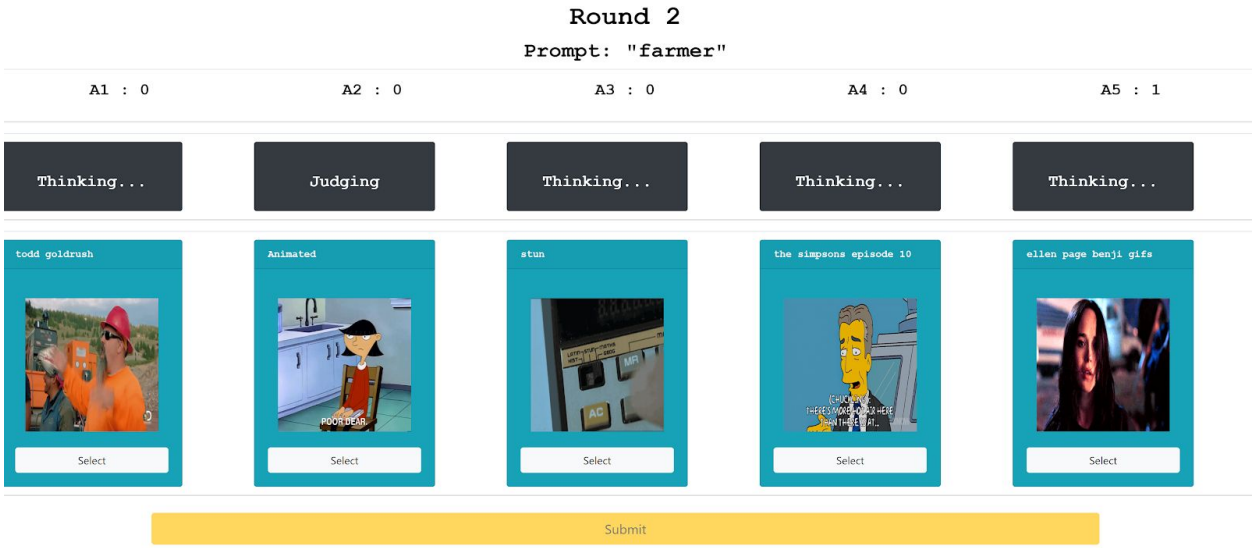


Figure 6:

Story Card 7: As a user, I'd like to be able to receive a new card at the end of a turn so I have new giphs

Story Card 6: As a user, I'd like to see the scoreboard as to see where I rank with other player

Story Card 17: As a user, I'd like to see the final scoreboard at the end of the game

After the 10th round, the next round button transitions to a final results button.

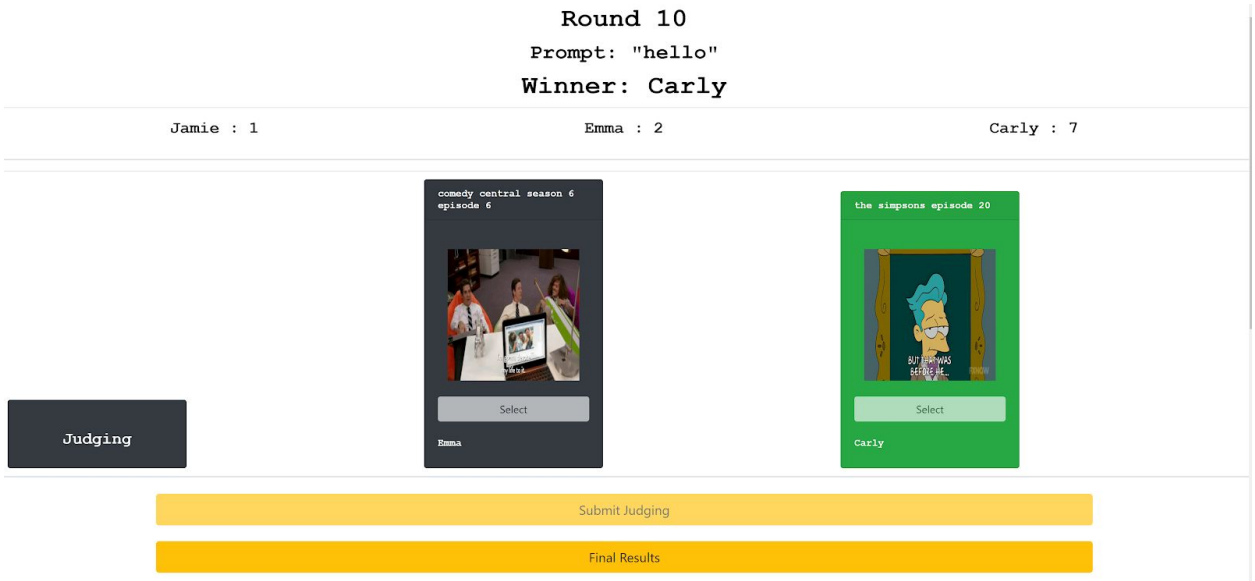


Figure 7:

Story Card 17: As a user, I'd like to see the final scoreboard at the end of the game

Story Card 17: As a user, I'd like to see the final scoreboard at the end of the game

After clicking “Final Results”, users are taken to this page to see the final scoreboard and who was the overall winner.

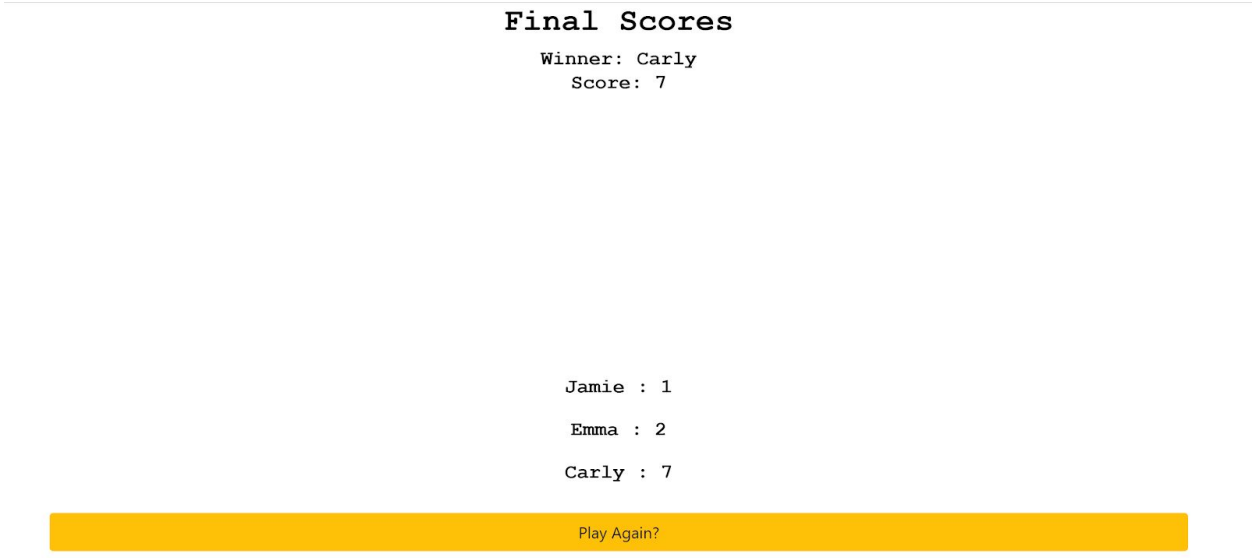


Figure 8:

Story Card 17: As a user, I'd like to see the final scoreboard at the end of the game

5. **Testing:**

All of our testing was UI testing. We ran ‘npm start’ in both the server and client folder on localhost. Testing involved adding console.log statements, using the chrome console, inspect, source, and reactJS extension. For the firebase hosting, testing that our application was hosted (front - end) at this url (https://giphs-against-humanity.firebaseapp.com/). We also simulated a

full mock game (changed the max rounds to 3 for testing sake). To test the server running remotely. You have to run `npm start` in the server folder and have ngrok installed. Open the executable and run the command `ngrok http 900`. You then need to take the given server string and replace it in the client side code in the `Game.js` file and the `Mode.js` file for the api addresses. After that, you will have to redeploy to firebase, so will need to run `npm run-build script`. You will need the firebase hosting account, and then you run `firebase init hosting` using the `build` folder, `y`, and `n` to overwriting the index.html file. After that's done, run `firebase deploy` and the version with the remote server api address will be hosted.

6. **Summary:**

Overall, we consider our project to be a success. We were able to meet just about all of the main goals we had laid out at the start of the assignment. The final implementation is fully playable with a clean and organized user interface; it handles gameplay very well, and functions with only a few minor bugs. We feel happy with what we implemented in the time frame given to us.

In regards to what was most beneficial, our team met on a weekly basis outside of class to discuss our code, ask questions of each other, bounce ideas back and forth, and prioritize tasks. This allowed us to maintain the project's scope and go over any potential issues we may have with our implementation. We were able to walk through some of the basics of the ReactJS components and socket.io together, which proved helpful throughout the course of the project. Meeting up in person really allowed us to get more out of each interaction. Additionally, the story cards were very useful for managing what stages our tasks were in. It was a great reference to look back to when we identified our next steps.

With any project, there were a few things we could have done differently. The tools we initially set out to use were slightly different from the ones we chose after learning more about the project composition. For future projects, we would acquire a better understanding of these tools and whether they would be a good fit for what we hope to achieve. We also faced some challenges with the file structure of our program. While our final implementation works really well, it was hard to identify. If these types of resources are available, it might be helpful to ask for outside feedback, or look for any similar projects written by someone in the industry that are available to the public. These would allow us to ensure our file structure was built with good practice in mind, and save us from a few instances where we decided to reorganize.

This project was a great learning experience. We enjoyed the open-ended aspect of the project, the ability to get creative, and some of the freedoms that come with it. Our group left with several positive takeaways and areas of growth to keep in mind.