

Team Number: 103-4

Team Name: TeamName

Team Members: Austin Lucas (A-Luc), Christopher McCarroll-Gilbert (chrismcg14), Hayes Vavpetic (hayes-vavpetic), Pranav Subramanian (pranav-super), Justin Murillo (jumu3668)

Application Name: Cards Against Profanity

Project Milestone 7

PART 1 - PROJECT DETAILS

Title: Cards Against Profanity

Who: Austin Lucas, Christopher McCarroll-Gilbert, Hayes Vavpetic, Pranav Subramanian, and Justin Murillo.

Project Description:

Cards Against Profanity is a novel take on the classic “Cards Against Humanity”/“Apples to Apples” type game. All 3 of these games operate under the same premise. Every player is dealt a few cards, which we can call “response cards”. Every single round, a new card, which we refer to as a “topic card” is provided. The topic card might look something like, “I like to ____.” It’s now the players’ duty to select one of their response cards, which could be really *any word*, and use those cards to respond to the topic card. If my best response card is just the word “cry”, I’d play that and hope nobody else put something better. Every round, a judge is picked (on a rotation), and the judge is the only player that, instead of submitting a card this round, judges the submissions and picks the best response. The judge picks a winning card, and the player who played that card wins the topic card, or an associated number of points!

The novel feature that we provide is the fact that we have migrated this card game to be a playable mobile application, and unlike “Cards Against Humanity”, which is geared towards adults, and “Apples to Apples”, which is geared towards kids, our game lets players choose if they want to play with kid-friendly or adult-friendly cards.

Also, by virtue of being an application, we offer user authentication services that allow players to have their own accounts, so that they can keep track of their own statistics, including information about how many games they've won! We also offer an in-game chat functionality, a live scoreboard, as well as a lobby creation system based on lobby keys, so that you can play locally with your friends by letting them know what the lobbykey is!

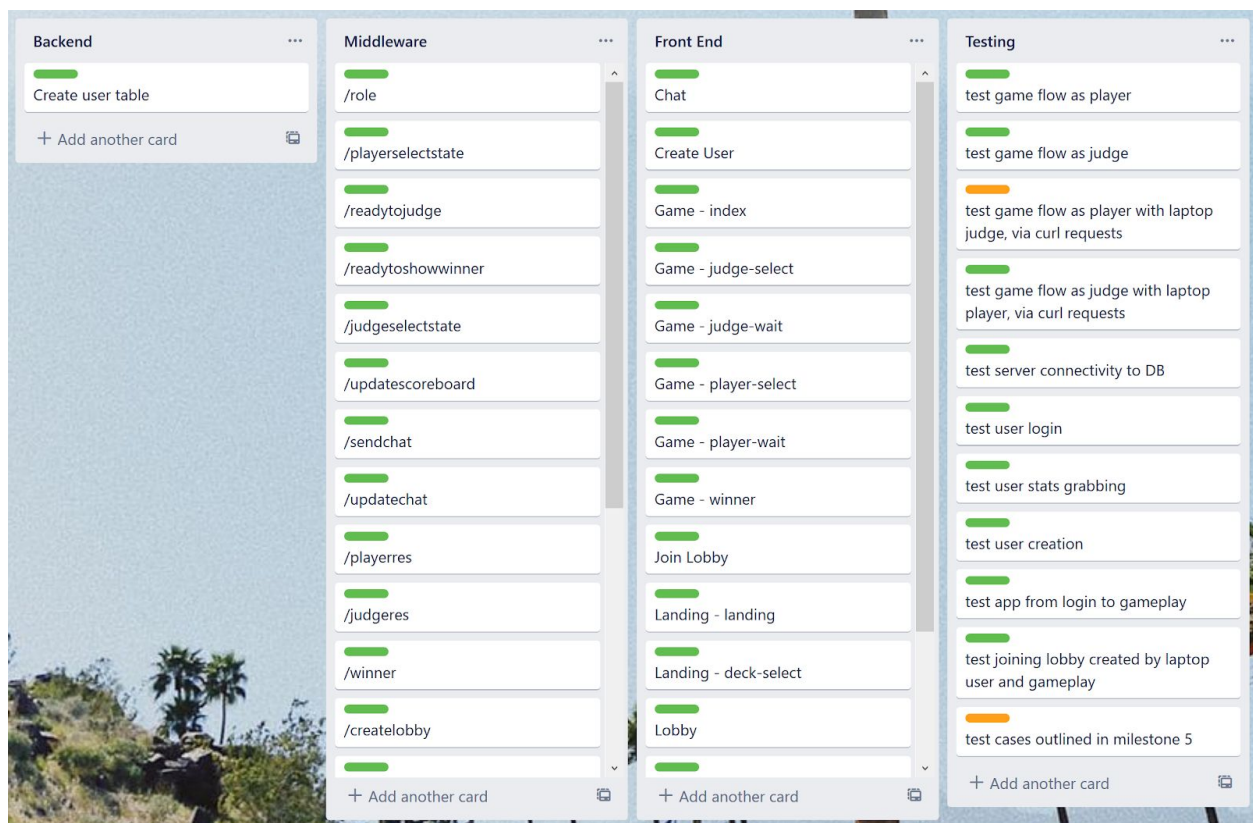
PART 2 - ADMINISTRATIVE DETAILS

Project Tracker:

Our project was tracked primarily via Trello, although a lot of it was decentralized and assigned more informally via Slack. That being said, our Trello can be found here:

<https://trello.com/invite/b/v5n9oPIP/131e2d788e52451f10b614504a024829/work>.

If you don't want to join the Trello, but simply wish to view what our project tracker and task listing look like, here is a screenshot of it! It lists all the major tasks that we had to work on in the frontend, backend, and middleware. A green label indicates that a task is completed, while an orange one indicates that a task is completed to the best of our ability (we make reference a lot to an unfixable bug that we encountered, as a result of the primary library that we use, React-Navigation, still dealing with an app-interrupting bug, some of which ruin gameplay).



Video:

You can find our video on our project's GitHub, here:

https://github.com/pranav-super/csci_3308_103-4_project/tree/master/Additional%20Project%20Information. The file is called 'FunctioningDemoVideo.mp4'.

VCS:

The repository can be found here: https://github.com/pranav-super/csci_3308_103-4_project/.

This includes the following:

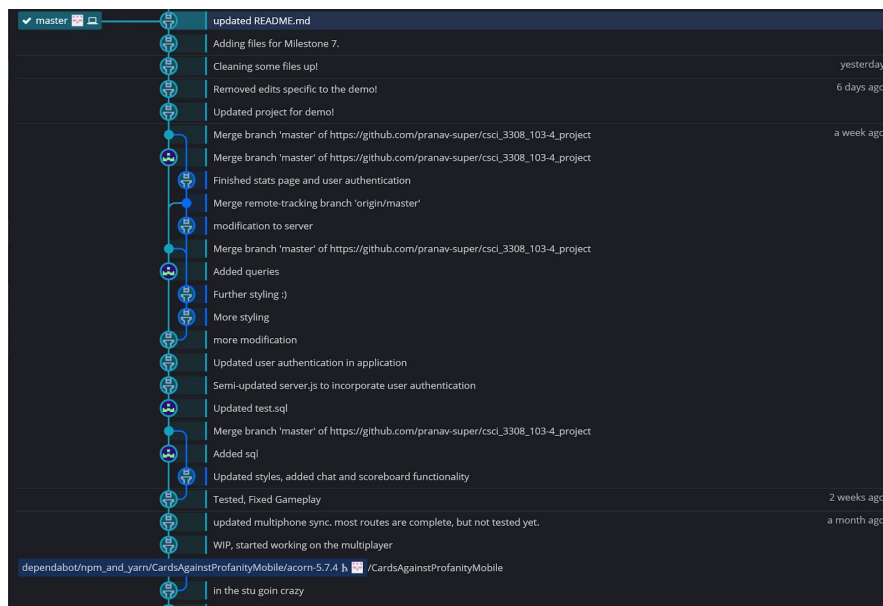
- Source Code
- Test Cases
- Video Demo
- Project Milestone 7
- A README.md explaining the project, the repo's organization (including where to find the files mentioned above), and how to deploy it.

Contributions:

There were a lot of commits made throughout the history of this project. Including screenshots of all the commits would be pretty inefficient, so we've just included the most recent ones.

However, you can access the whole list here:

https://github.com/pranav-super/csci_3308_103-4_project/commits/master.



This screenshot comes from GitKraken, the primary desktop client that we used for this project.

Now, one thing worth noting is that a lot of the commit history might look pretty imbalanced. When we did our work meetings, we all contributed to and discussed the code and debugging. For something like an app, we found that it would work a lot easier if we just called each other on Zoom and had one person actually do the typing, while we as a collective figured out what to do. Having multiple people contribute commits at various times to a project like a mobile application seemed daunting, due to how frequently small changes could mess up the entire project. Since Pranav's IP address was written in as the local host, we had to send him all of our work to test. We would always test while all of us were on Zoom so we could come up solutions together. Since the final code was always on Pranav's computer, he was always the one to commit the work to the repo as it didn't make sense for him to email it to us to commit.

While the majority of what we did was a collective effort, here is a list of features (features which we outlined in Project Milestone 4) and which members contributed to them, followed by a brief from each team member about their contributions:

- *Gameplay*
 - This effort was carried out partly by Pranav, with contributions from the rest of the team during select work meetings on Zoom.
 - Hayes, Chris, and Justin contributed to this effort at various points by giving suggestions and helping debug via the Zoom call.
- *Multiphone Sync*
 - This effort was carried out mostly by Pranav, with Justin working on trying to deploy the app publically (although we later decided against it, see the *Deployment* section of this milestone).
- *Matchmaking*
 - This effort was carried out by Hayes and Chris, with contributions from the rest of the team during select work meetings on Zoom.
 - Pranav helped build and test the features.
- *In Game Chat*
 - This effort was carried out mostly by Pranav, and was done concurrently with the Multiphone Sync feature. The same was the case for the scoreboard, which was functionally a modified copy of the In Game Chat code.
- *User Account System*
 - This effort was carried out mostly by Justin and Austin.
 - Justin and Austin iteratively designed and refined the database (which we ultimately worked down to a single table to be hosted on a MySQL Server).
 - Pranav wrote the framework for the routes to link to the database, and Justin finished it by filling in the actual SQL queries, as well as connection information, to the database.
- *Deck Selection*
 - This effort was carried out by Hayes and Chris, while Pranav helped build the features.

We can summarize this list by team member, as well:

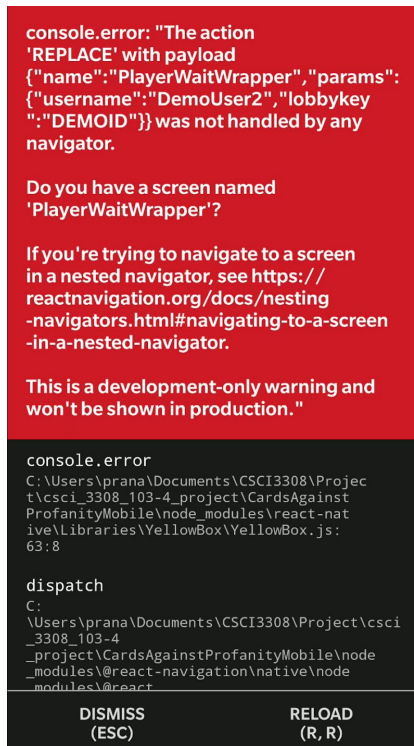
- Pranav handled the gameplay, multiphone sync, chat, and helped with the user account system.
- Hayes worked on on the gameplay, the deck selection, and matchmaking.
- Justin worked on sql database framework and the gameplay, as well as parts of the multiphone sync.
- Chris worked on on the gameplay, the deck selection, and matchmaking.
- Austin helped work on the database.

Deployment:

You can find all the information on this outlined in the README.md of our repository, under the “Deployment” heading (there’s a lot to it, and we didn’t want to be redundant by essentially pasting the same information twice); it talks about the setup and deployment of our app. Note that we chose to deploy our application locally - we spent a significant time attempting to get our application to run remotely, but encountered a lot of strange errors, including some with React Native itself! As a result, we decided to keep it at a stage where it worked, which was a local deployment.

An Error:

We encountered a pretty serious bug that had to do with our navigation library. As a matter of fact, this bug even showed up during our demo! The bug has to do with our dependency that we use for navigation between views, “*react-navigation*”, not properly handling a transition between views! This error arises sometimes in transitioning between views in the game - since we have a lot of transitions between views and rely heavily on *react-navigation* in our app, sometimes it falls apart and misses handling a transition, which manifests itself as either the game getting stuck on a certain view, or as an error, which you can see below:



```
console.error: "The action
'REPLACE' with payload
{"name":"PlayerWaitWrapper","params":
{"username":"DemoUser2","lobbykey
":"DEMOID"}} was not handled by any
navigator.

Do you have a screen named
'PlayerWaitWrapper'?

If you're trying to navigate to a screen
in a nested navigator, see https://
reactnavigation.org/docs/nesting
-navigators.html#navigating-to-a-screen
-in-a-nested-navigator.

This is a development-only warning and
won't be shown in production."

console.error
C:\Users\prana\Documents\CSCI3308\Projec
t\csci_3308_103-4_project\CardsAgainst
ProfanityMobile\node_modules\react-nat
ive\libraries\YellowBox\YellowBox.js:
63:8

dispatch
C:
\Users\prana\Documents\CSCI3308\Project\csci
_3308_103-4
_project\CardsAgainstProfanityMobile\node
_modules\@react-navigation\native\node
_modules\@react

DISMISS (ESC) RELOAD (R, R)
```

It is unable to reach the view that we want, which, in this case is the (explicitly) defined PlayerWaitWrapper view. This happens with a variety of views, and, while researching the bug, was found to be a fault in react-navigation. The solutions we encountered were either outdated or simply didn’t work with our application. You can find a video demo of this in the repository under the “Additional Project Information” directory (it only happens at the end, but the video is 2 minutes long).