Gin Rummy

Team Scuttle Crab: Ryan Scott, Brian Nguyen, Julian Manaois, Janelle Lara

Architecture

MVC Framework

Technologies used:

- JS
- EJS
- SCSS - used for automating css for repeated/similar elements
- CSS
- Express
- PostgreSQL
- Socket.io
- Render

Completed Requirements

- User authentication (register, login, logout)
    - Logged in users have access to game rooms they enter, not accessible to guests or other users if they are not in the room
- Chat enabled in lobby and game rooms
- Game states persisted in database, and are updated real time via websockets
    - Users receive game states that are relevant to them
    - Server is only source of truth
    - Users can connect into games they previously entered using room id in url
- Users can enter any amount of games in separate tabs
- Design is not terrible

Challenges

- Familiarizing ourselves with the game and its rules because we’ve never played it before
- Implementing game logic (specifically knocking and melding), as it involves many event interactions, and gathering/sending data back and forth to/from the database.
- Start of game turn-progression logic, giving a user option to pass their turn but not allowing after the first round
- Lay-off phase, somehow tell database which opponent meld we are melding with
- Constraining certain functions because of game state, like discarding when not your turn, etc.
- Scheduling conflicts. It was difficult to organize group gathering due to other school work/events and work.

What we learned

- How to implement different states using a database and updating them real time to users
- Using CSS mapping to get portions of an image as our cards to reduce amount of files
- How to implement chat feature using websockets
- How to use migrations for databases
- Balancing multiple projects/working in crunch time and working with a team
- How to implement difficult game logic(meld, knocking, layoff phase of game)
- How to use Render to deploy web apps

Demo