CSC 667 - Term Project

Final Documentation

Team Members

Vern Saeteurn
Jessica Serrano
Jair Gonzalez
Rodrigo Gallardo

GitHub Repository Link

https://github.com/sfsu-csc-667-spring-2021-roberts/term-project-team-jjrv

Application Link

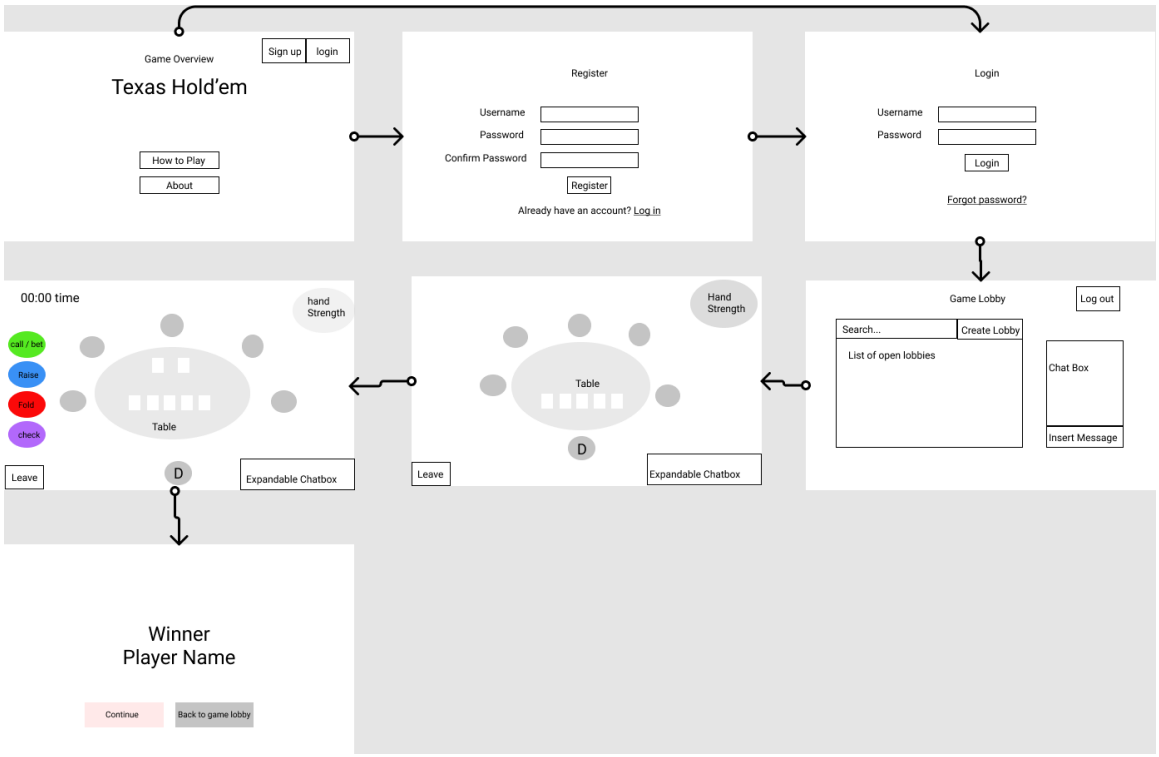
https://project667.herokuapp.com/

Contents

Introduction

The web application that our group decided to work on for our project is a classic poker game, Texas Hold Em'. It allows users to register and log in, chat with other players in either the lobby or game lobby, and join game lobbies to play with other players.

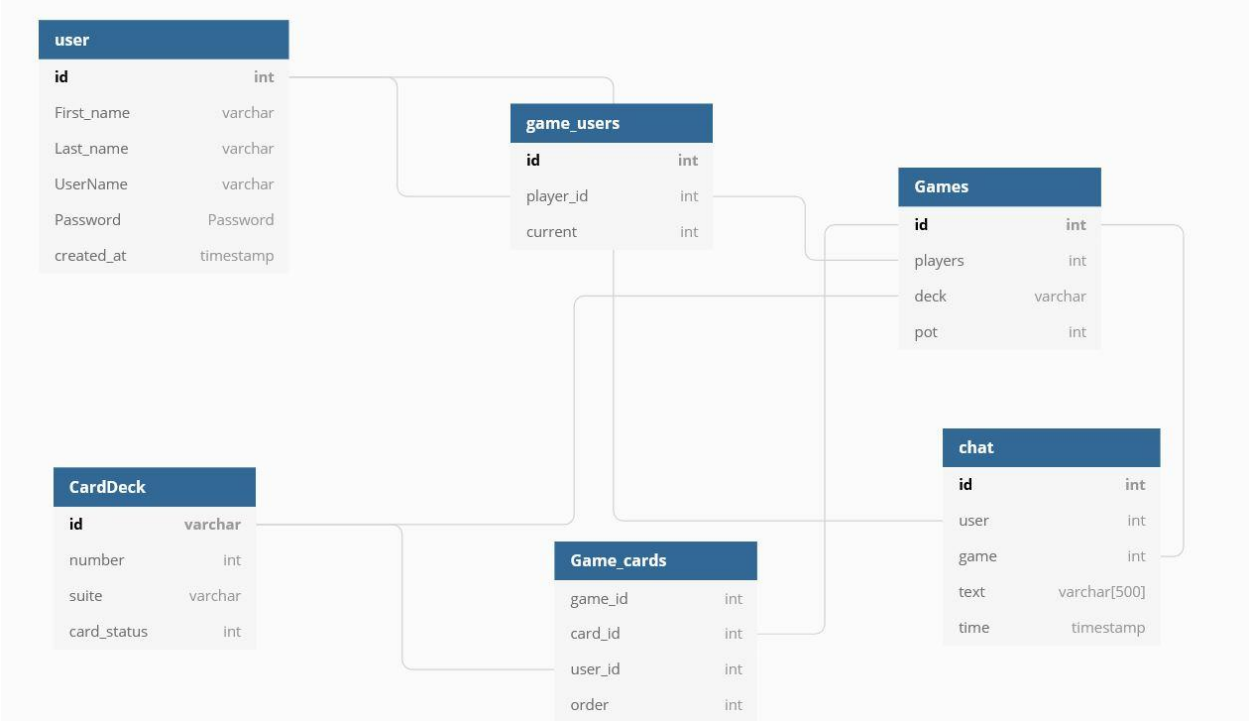
Wireframe Overview



Technology

- PostgreSQL
- Heroku
- Nodemon
- Node.js
- Express.js
- Frontend: Pug and CSS
- Sockets: Pusher.js

Database Structure



SQL Queries

```
CREATE TABLE public.cards
(
    card_id integer NOT NULL,
    rank integer NOT NULL,
    suite character varying(30) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT cards_pkey PRIMARY KEY (card_id)
)
CREATE TABLE public.game_cards
(
    game_id integer,
    card_id integer,
    user_id integer,
    card_order integer
)
```

```
CREATE TABLE public.game_users
(
    "order" integer NOT NULL,
    game_id integer NOT NULL DEFAULT nextval('game_users_game_id_seq'::regclass),
    current integer NOT NULL,
    users_in_game integer NOT NULL DEFAULT nextval('game_users_users_in_game_seq'::regclass),
    CONSTRAINT game_users_pkey PRIMARY KEY (game_id)
)

CREATE TABLE public.games
(
    game_id integer NOT NULL,
    pot integer
)
CREATE TABLE public.messages
(
    game_id integer,
    user_id integer,
    "timestamp" timestamp with time zone DEFAULT now(),
    content character varying(500) COLLATE pg_catalog."default"
)
CREATE TABLE public.users
(
    user_id integer NOT NULL DEFAULT nextval('users_user_id_seq'::regclass),
    username character varying(30) COLLATE pg_catalog."default" NOT NULL,
    first_name character varying(100) COLLATE pg_catalog."default" NOT NULL,
    last_name character varying(100) COLLATE pg_catalog."default" NOT NULL,
    password character varying(500) COLLATE pg_catalog."default" NOT NULL,
    created_at time without time zone NOT NULL DEFAULT now(),
```

```
    CONSTRAINT users_pkey PRIMARY KEY (user_id),
    CONSTRAINT users_username_key UNIQUE (username)
)
```

Functional Requirements

Category	Requirement	Completed	Comments
Login and Registration	Users can register and make an account	✔	
	One account is associated with a single username	✔	
	Registered users can log in	✔	
	Login requires username and password	✔	
Main Lobby	Users can create a new game lobby	✔	
	Users can join a game lobby		
	Users can return to a game lobby		
	List of games is updated live		
Game Lobby	Users can see all other players in game lobby		
	Users can leave game to go back to main lobby		
	Up to six players can join game lobby		
Chat	All users can chat in main lobby	✔	
	Users in a game lobby can chat with each other		
Game Logic	Players can call or place a bet		
	Players can raise a bet		
	Players can check/pass on betting		
	Players can fold or discard their current hand		
	Players can keep playing until they run out of chips		

	Showdown occurs when there are only two players left in play		
	Deck is shuffled after every round is finished		
Code Quality	Well-organized into meaningful file structure	✔	
	Well-formatted and readable (appropriate white space and indentation)	✔	
	Single responsibility principle (methods implemented serve a single function and are small in size)	✔	

System Design Decisions

- Chat
    - When a player sends a message in chat in a game lobby
        - The message is seen by all players in the same game lobby
- Shuffling Deck
    - Deck is shuffled at the end of a showdown & before the start of a new round
        - The cards will be shuffled and ready to deal
    - When a player calls
        - His opponents will place a bet or the round is over
- Showdown
    - Occurs when there are at least 2 players still in play at the end of the last round
        - Players will reveal cards to decide who has stronger hand and wins the round
- Winning a Round
    - A player wins a round by either having the strongest hand in a showdown or all other players fold before a showdown
        - Player will be prompted with the amount of chips won
- No Chips Left
    - When a player runs out of chips
        - The player is unable to continue playing and is not dealt a hand

Challenges

Non-technical:

- Team Communication
    - Difficulties scheduling meetings to work on project
- Time Management

Technical:

- Deploying app to Heroku
    - Issues with set up to the db via heroku

    - SSL
- Configuring .env files for team members
- issues with set up with Pusher on heroku
- Working across different computer platforms
    - Such as windows OS vs. MacOSX
- New technologies to learn

Conclusions

Our team believes that this project was a very good learning experience although many of the functional requirements were not met. Game development along with full stack web development encompassed many new frontiers for all members of the team. Learning new technologies and methodologies, then shortly applying them to create a solution, proved to be a very challenging experience. When overviewing specific technologies within this project, that we believe to have brought us a great value. We look at Pusher and Sequelize as key importance to this project. Pusher, a hosted API service that brings real time data into our application had a large learning curve but after testing and research, we were able to create a chat system that successfully delivers messages to all members within the same subscribe channel. Sequelize brought middleware that connected our node.js project to our database. Although there is much room for improvement this project opened many of our eyes, indicating areas in which we could further improve as individuals.