***CSC 667 Project Documentation***

***Fall 2022***

***Thursday Section 02:***

***Aleksandr Gusev***

***Alexandre Sauner***

***Nicholas Thomas***

***Monday Section 01:***

***Jesus Arambula***

***GitHub Repository:***

[***https://github.com/csc-667-fall-2022-sfsu-roberts/term-project-teamn***](https://github.com/csc-667-fall-2022-sfsu-roberts/term-project-teamn)

## 1 Introduction

### 1.1 Project Overview

This assignment was to create a server to host a game of UNO for anyone to join. The server would take in any number of players and let the users select the game room from a list where they want to join. This server also included both a global chat feature and a game room chat feature where users can interact with each other. This server also allowed users to create and login to their own accounts to participate in the game rooms. Anyone with the link to the website would be available to view the site and create an account and add it to the list of all other user accounts in the database.

### 1.2 Technical Overview

This server is composed of multiple languages primarily http and hypertext languages. These languages include HTML, CSS, PostgreSQL and JavaScript with the NodeJs framework. The app is primarily run on the Javascript code while the HTML and CSS only serve as the front end of the project. We also used PostgreSQL for the database for our project to store game variables and user accounts.Most of the information in the database will be fetched through the API in the back end with NodeJS and displayed to the webpage.

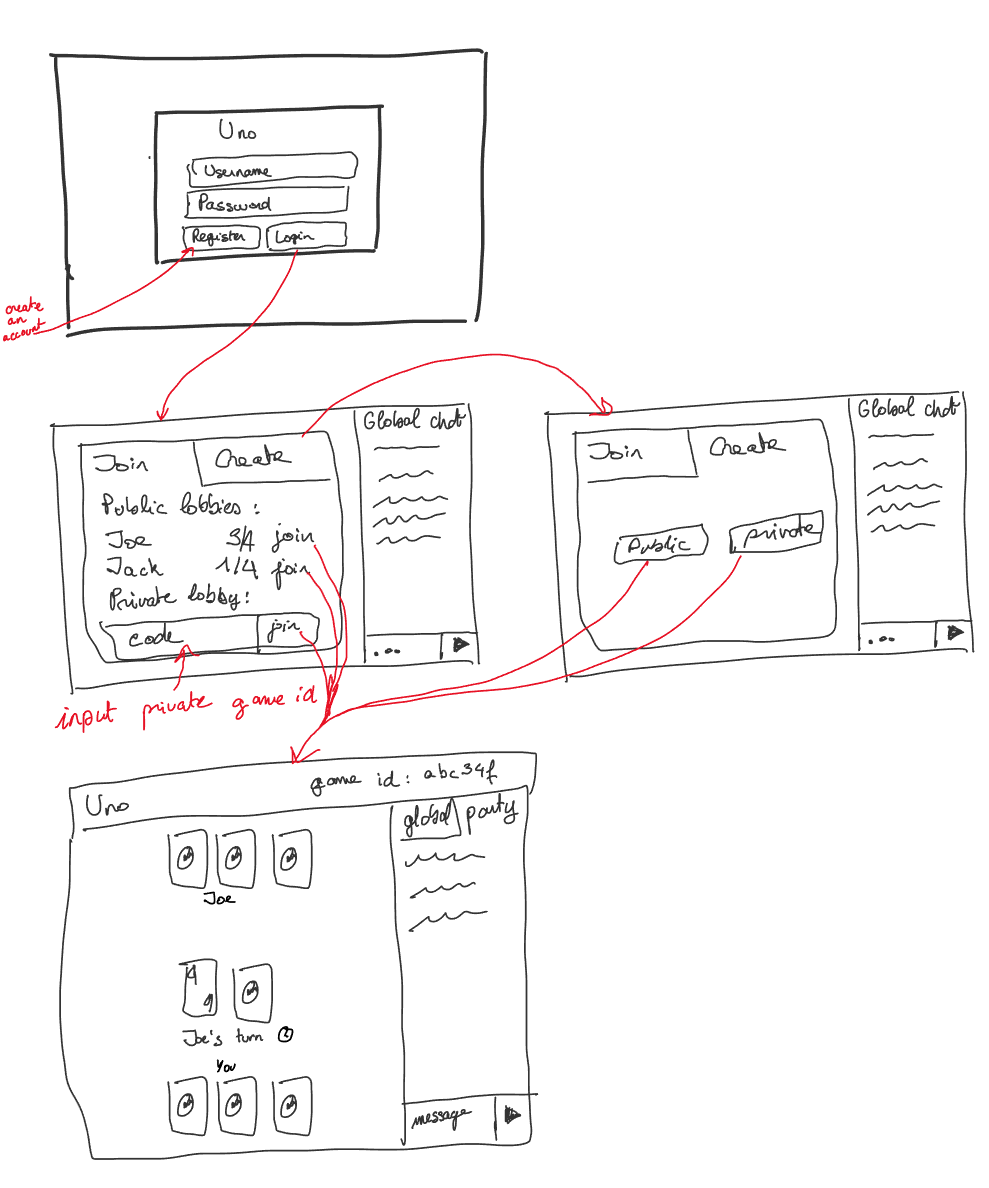
The way this website works is through the use of routes in the backend. Based on the url displayed in the search bar on top of the browser. Each route will route to a different page in the server when a user visits that particular route in the website.

### 1.3 Summary of Work Completed

For the work completed it was primarily JavaScript with a use of Embedded JavaScript templates (EJS) to provide for the views of the website. The project was divided in directories to make the organization more clean and easier to work with.

* app-config:
  + Authentication for login, registration, and lobby views
* bin:
  + Includes module dependencies for app, http, and socket initialization
  + Creates the HTTP server and listens on port provided (3000)
  + Also has handles for error and listening events
* config:
  + Contains the card information for the game (colors, numbers, draw cards, etc.)
* db:
  + Contains all database variables and functions related to creating, reading, updating, and deleting information from the database.
  + Also contains user registration and login functions
* migrations:
  + Contains the database migrations
* routes:
  + Has chat JavaScript code for user’s game chat feature.
  + Validates user input when registering, logging in, and handles login error and logout requests.
  + Renders the website home page
  + Also contains the directory which holds the file that handles all of the routes for the website (‘/mygames’, ‘/create’, ‘/join’, etc.)
  + And gets the list of all users
* sockets:
  + Initializes the server socket and app configurations, and makes a connection to the server.
* views:
  + Contains all of the views for the website as Embedded JavaScript templates.
  + These views include the game lobby view, login, register, create, etc.
  + Also contains miscellaneous .ejs files for header, footer, error, etc.

## 2 Wireframe



## 3 Development Environment

**Technologies Versions:**

Nodejs version 18.12.1

PostgreSQL 14.6

Embedded JavaScript template version 3.1.8

CSS 2.1

**IDE Version:**

Visual Studio 2022 version 17.4.1

## 4 Project Reflection

To many of us working on this project was the first time since we had to learn new technologies outside of the ones we already know. It was especially difficult to run on local machines given the file structure of the project. For others it was also a change of pace switching from MySQL to PostgreSQL given that we don’t have a GUI to work with and only with the terminal.