The Last Project: Brian Liu, Caden Khuu, Claire Song, Tracy Ye
SoftDev
P05
2025-06-06
Time Spent: 3 hrs

DESIGN DOCUMENT (VERSION 3)

Project Description

Our project is a multiplayer mafia game where players can create or join public lobbies and get a randomly assigned role: mafia, doctor, cop, fool, or civilian. Once the game starts, it will progress through day cycle phases (night, dawn, morning, evening, and dusk). At night, the mafia, doctor, and cop will pick someone to kill, save, or investigate respectively. At dawn, players will see who got killed or saved and the cop will see what their target's role is. In the morning, players will type to each other in the chat to discuss who they think the mafia is. In the evening, they will vote for the most suspicious player. At dusk, the player with the most votes will be executed and the game will either end or the day cycle will repeat. The game ends when the mafia is eliminated, all the innocent players are eliminated, or the fool is executed.

Program Components

Front-end Components

1. NodeJS & React
    a. JoinRoom.js
        i. Lets users join waiting rooms
    b. WaitingRoom.js
        i. Shows the list of players in the waiting room
        ii. Starts the game if 5 players are in the room
    c. Game.js
        i. Moves the players through the different day cycle phases & shows the timing of each one
    d. Night.js
        i. Lets users with special roles (mafia, doctor, or cop) choose their targets
        ii. Shows descriptions of each role
    e. Dawn.js
        i. Shows player deaths from the night phase
        ii. Shows the cop target's role to the cop
    f. Morning.js
        i. Has a chat to let players discuss who they think the mafia is
    g. Evening.js
        i. Allows alive players to vote for who they think the mafia is
    h. Dusk.js
        i. Reveals which player got voted out
    i. Win.js
        i. Shows who won the game
        ii. Closes the room
2. Tailwind CSS - Frontend Framework

a. Tailwind provides user customization for accessibility, a clean user interface, and easy in-line styling that allows for different styling amongst the same HTML tags.
b. Features:
    i. Responsive design, animations, sidebars/navigation bars, and resizing based on screen size

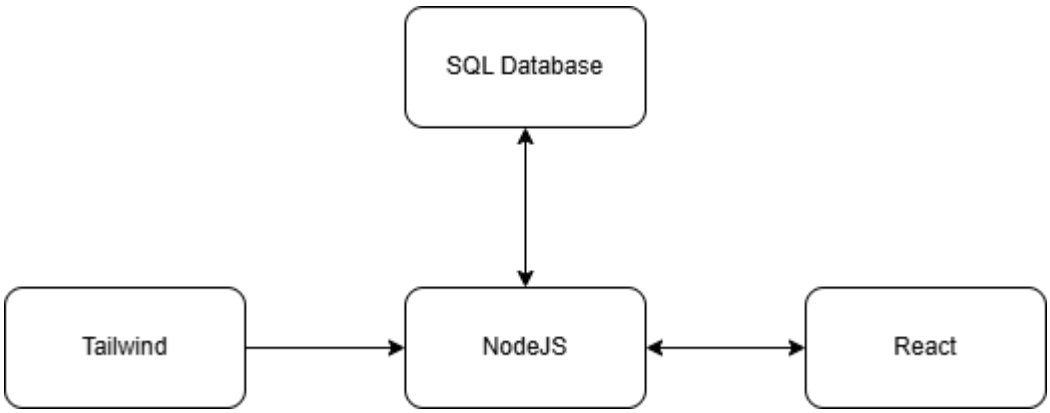
Back-end Components

1. NodeJS
    a. index.js
        i. Handles the timing of each phase
        ii. Adds users to rooms
        iii. Includes functions that allow users to send messages, see the alive/dead user lists, count and process the condemn votes, check if win conditions are met, give users roles, and game logic
        iv. Includes database functions
2. SQLite Databases - Stores information from datasets
    a. Store necessary information about rooms to ensure the game runs smoothly

Site Map

*For our project, everything happens on one page because there isn't a need for users to have to switch to a different page because all that they can do is join a room and play the game.

Component Map



Database Organization (SQL)

User Table (rooms) - stores information of each room, including its ID, user ID, role, username, and information to facilitate game logic

user_id	TEXT	Unique identifier for each member in the party
room_id	TEXT	Unique identifier for each game room/party
role	TEXT	Role of the user
username	TEXT	Username of the user

spectating	REAL	Whether or not the user is spectating (dead)
condemnCnt	REAL	The number of people who voted them guilty during Evening
phase	REAL	The current phase of the room the user is in

APIs

We will not be using any APIs.

Libraries

Cors
Express
Socket.io

Task Breakdown

Brian Liu (PM) – Node.js

1. Implementing functions that help the game run using JS, such as role functionality.

Caden Khuu – Frontend

1. Create templates and make website look nice with Tailwind and CSS
2. Help out with NodeJS

Claire Song – SQLite Database

1. Creating tables to keep track of user information and player information
2. Implement narration and animation into the game

Tracy Ye – Node.js

1. Linking pages and backend using NodeJS
2. Implement multiplayer, party system, and real-time updating of information within the game.


Errors
Warnings
Info
Logs
React DevTools for a better development experience: https://react.dev/link/react-devtools
Request Blocked: The Same Origin Policy disallows reading the remote resource at http://localhost:3001/socket.io/?EIO=4&transport=polling&t=slvzh0lg. (Reason: CORS request did not succeed). Status code: (null). [Learn More]
Request Blocked: The Same Origin Policy disallows reading the remote resource at http://localhost:3001/socket.io/?EIO=4&transport=polling&t=slwzdufv. (Reason: CORS request did not succeed). Status code: (null). [Learn More]
Request Blocked: The Same Origin Policy disallows reading the remote resource at http://localhost:3001/socket.io/?EIO=4&transport=polling&t=slv6lfey. (Reason: CORS request did not succeed). Status code: (null). [Learn More]
Request Blocked: The Same Origin Policy disallows reading the remote resource at http://localhost:3001/socket.io/?EIO=4&transport=polling&t=sm1sz8d8. (Reason: CORS request did not succeed). Status code: (null). [Learn More]
Request Blocked: The Same Origin Policy disallows reading the remote resource at http://localhost:3001/socket.io/?EIO=4&transport=polling&t=sm5r3piv. (Reason: CORS request did not succeed). Status code: (null). [Learn More]