Team Magikarp - Sean Ging (PM), Wen Hao Dong, Reng Zheng, Haotian Gan
Design Doc:

Concept: Battleship

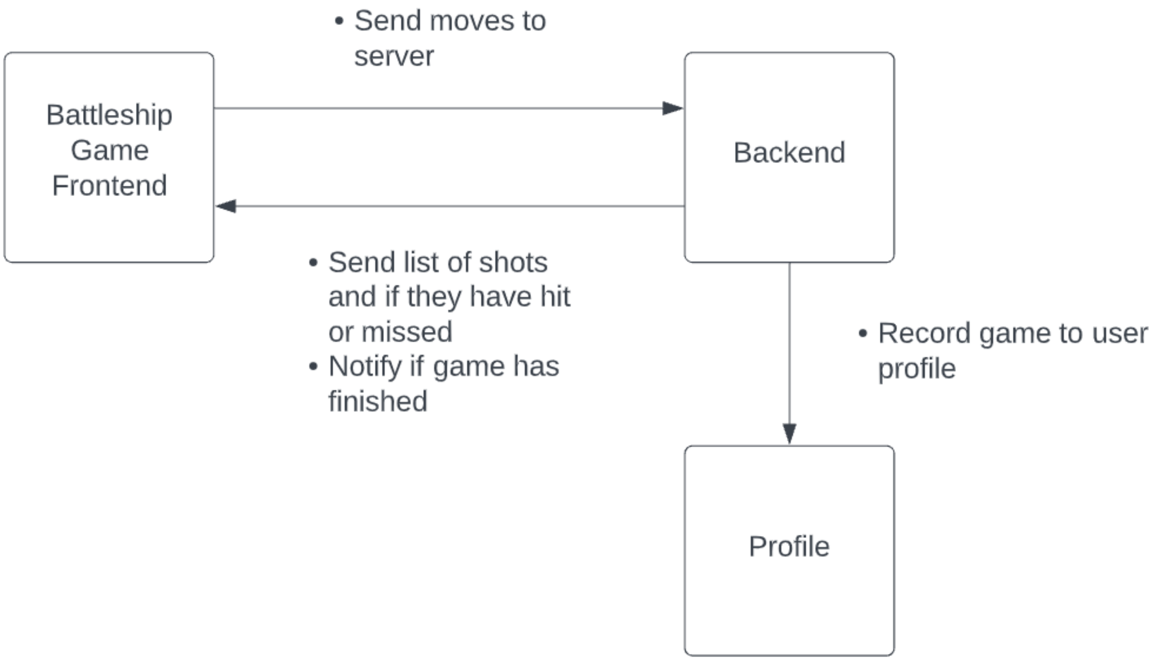
- Users registered with the site can play Battleship in two different modes:
    - Use the mouse pointer to make moves, or WASD/Arrow Keys + Enter/Spacebar
    - Multiplayer on the same computer (Pass and Play)
    - Singleplayer (Versus Computer)
        - Computer player uses a probability based algorithm and several optimizations outlined here:
            - https://datagenetics.com/blog/december32011/index.html
    - **Optional:** Multiplayer, played between two accounts, but not real-time.
        - A player makes their move, then presses “Submit Move”.
        - The other player receives a notification that a move has been made, and clicks a link to view the board and make their own move. They click “Submit Move” once they have made their move, and so forth until the game is over.
        - The notification feature is implemented with short-polling (Fetch request to server every five seconds to check if there are any notifications)
- Battleship games are played over a 7x7 board with 5 ships
    - The 5 ships are:
        - Carrier (occupies 5 spaces),
        - Battleship (4)
        - Cruiser (3)
        - Submarine (3)
        - Destroyer (2)
    - **Optional:** A gamemode where the user can create their own ship shapes.
        - Would require an implementation of a more complicated battleship computer player algorithm: https://pageperso.lis-lab.fr/guilherme.fonseca/battleship_conf.pdf
        - Would require an algorithm for placing arbitrary user-created shapes onto the board
- Games that you play are recorded on your user profile.
    - You can view the full details of each game you win or lose
    - If you were playing a game and did not finish it (exited midway), the profile is how you return to that game

Battleship Rules:
https://www.cs.nmsu.edu/~bdu/TA/487/brules.htm#:~:text=Once%20the%20guessing%20begins%2C%20the,%2C%20and%20Destroyer%20(2)

Program Components:

- Login and Signup
    - Pick username
- User Profile
    - View W/L ratio
    - Match history
    - Change profile picture
    - Start New Game
- Battleship Game Frontend:
    - Two 7x7 boards (One for the enemy, and one for the player)
        - In “Pass and Play”, the ships, once placed, are hidden from both players on both boards to prevent cheating
    - Responsible for allowing the user to drag and drop ships onto the board during the initial ship-placement phase, and performing client-side validation that all the ships are placed correctly.
    - Given the enemy’s hits and moves, and the player’s hits and moves, must correctly render the two boards
    - If the gamemode is “Pass and Play”, the client has full knowledge of where the battleship pieces are. The client sends the entire game state to the server after each move to save the game in case the client doesn’t complete the game in one sitting.
    - If the gamemode is versus a computer, the client does not have full knowledge of where the enemy’s battleship pieces are. The client is only given info about the computer’s hits and moves, the player’s hits and moves. The client is responsible for sending moves made by the player to the server.
- Battleship Game Backend:
    - POST endpoint for saving a game that the user wants to return to playing in the future.
    - Saves ongoing and finished games to the user’s profile
    - If the gamemode is “Pass and Play”, the server is only responsible for receiving game states from the client, and saving those game states.
    - If the gamemode is singleplayer, the server is responsible for:
        - Receiving the client’s moves
        - Telling the client the squares they have hit or missed
        - As the computer opponent, deciding on a move and telling the client their move
        - Keeping track of if the game is over or not
        - Saving the game on each turn to the user’s profile, in case they don’t complete the game in one sitting.

Component Map:



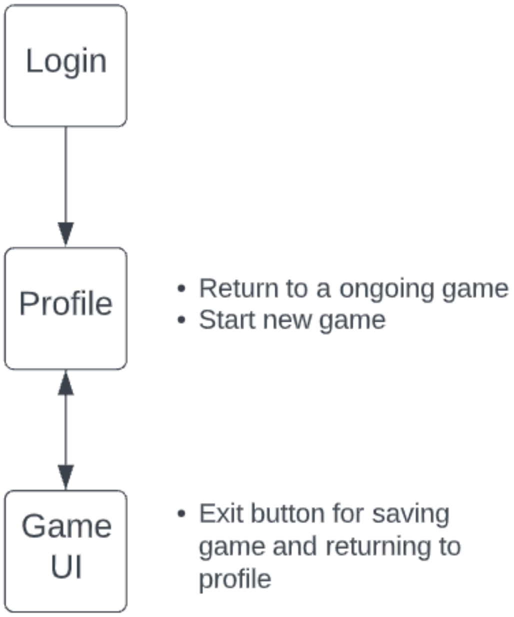
• Send moves to server
Battleship Game Frontend
Backend
• Send list of shots and if they have hit or missed
• Notify if game has finished
• Record game to user profile
Profile

Database Organization:

- Login and Signup
    - Username (Unique String),
    - Password (String)
- User Profile
    - Win Loss Table
        - Username (String),
        - Matches Won (Integer),
        - Matches Lost (Integer)
    - Match History Table
        - Match ID (Primary Key),
        - Player 1 Username (String),
        - Player 2 Username (String),
        - Player 1 Went First (Boolean)
        - List of Moves (JSON Object stored as String),
        - Game Finished (Boolean)
        - Ship Placements (JSON Object stored as String)

Code Components: Flask, Sqlite3, Javascript, CSS, Bootstrap

Site Map:



- Return to a ongoing game
- Start new game

- Exit button for saving game and returning to profile

Roles:
Sean Ging (PM): Backend (Move Validation, Move Saving, Profile)
Wen Hao Dong: Frontend, Backend (Game Engine, Singleplayer)
Reng Zheng: Frontend, Backend (Game Engine, Computer Player)
Haotian Gan: Frontend, Backend (Game Engine, Computer Player)

APIs:
No APIs

Bootstrap or Foundation:
Bootstrap, for its modals

Target Ship Date:
3/14/2022