Old Feature List from Milestone 2 (changes reflected by underline, priority reflected by bold):
Priority Scoring: 1(highest level of importance) - 5(lowest level of importance)

Project Features List:

Game Features:

1. Title: invite friends
    a. After logging in, players can invite friends by sending them the room ID, and their friends will be able to join the game after inputting the unique room ID. **(priority: already implemented)**
2. Playing: game mechanics
    a. Once you've started the game and invited other players you will be directed into a browser-sized map where all players are shown in the form of a tank. After a countdown, players will be free to use the arrow keys (WASD) to move their vehicle, and mouse to direct their tank's gun. The goal of the game is to collect power-ups and use them to fire upon and kill opposing tanks. The last man standing wins the round.
    (priority: 1, being able to implement all the game mechanics without mistakes is very important)

Web-Browser Features:

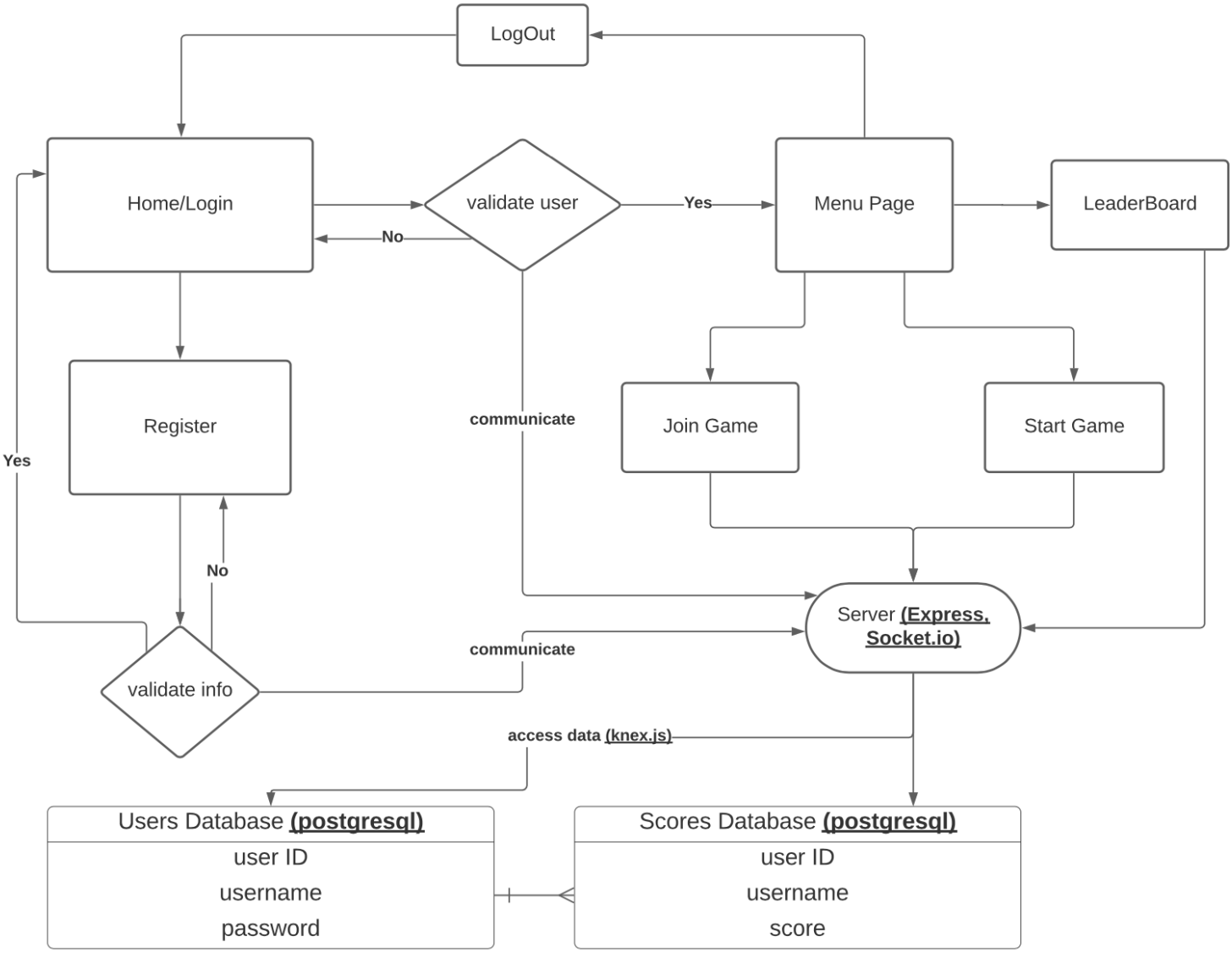
1. Aesthetically pleasing design that is easy to navigate and allows users of all skill levels to find controls and other game features. **(priority: 3, important but not as urgent)**
2. Leaderboard that provides real-time updates on who is winning/has the most points
    a. Involves connecting to the database in order to obtain this real-time information
        i. See Backend Server Features section for more insight on what features this will tie into **(priority: 4)**
3. Login and/or create user to access the game
    a. Be able to change screen name (name displayed during game and leaderboard), but username is unique and used for log in **(priority: 1, the players ability to create and edit their account is vital to the game canvas)**
4. Additional browsing pages (about us, game rules)
    a. Users are able to browse additional information, such as how to play the game, or know our team, by browsing the nav bar **(priority: already implemented)**

Backend Server Features: Server built on Express for Node.js

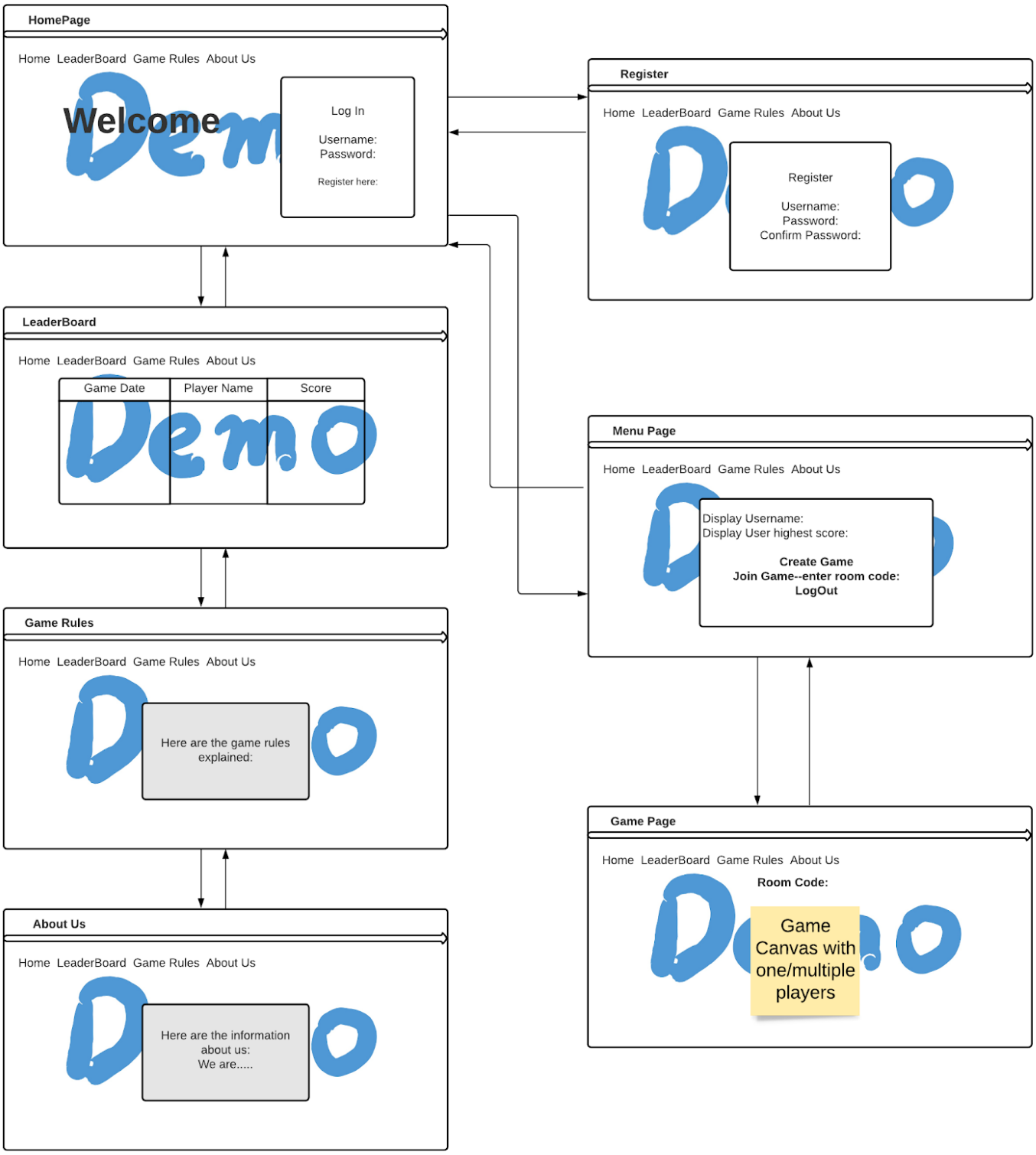
1. Once the game page has loaded, ensure the user is able to connect to the server. **(priority:4)**
    a. If unable to connect, throw an error message.
    b. Once connected, show the menu to allow creating games/inviting other players.

2. Download all the information required to play the game from the server **(priority:3)**
3. Once the game has started, the user needs to be able to see any updates that happen in game according to game mechanics. **(priority: 3)**
4. Keep the game updated according to the players' new information. **(priority: 4)**

Architecture Diagram:



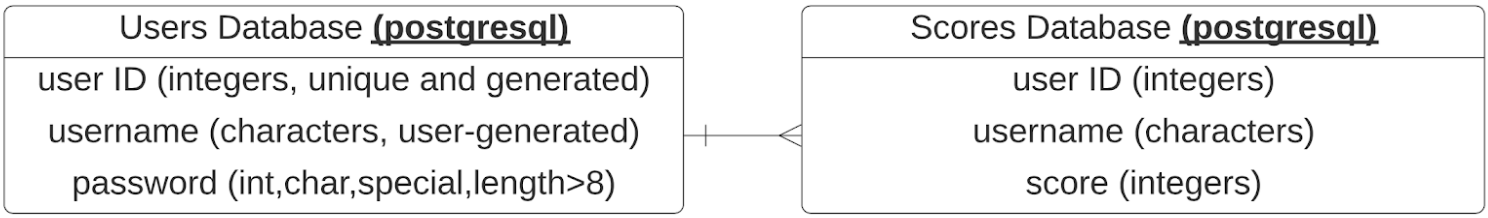
Front End design:



Web Service Design:

Our application is not using any Web Services.

Database design:



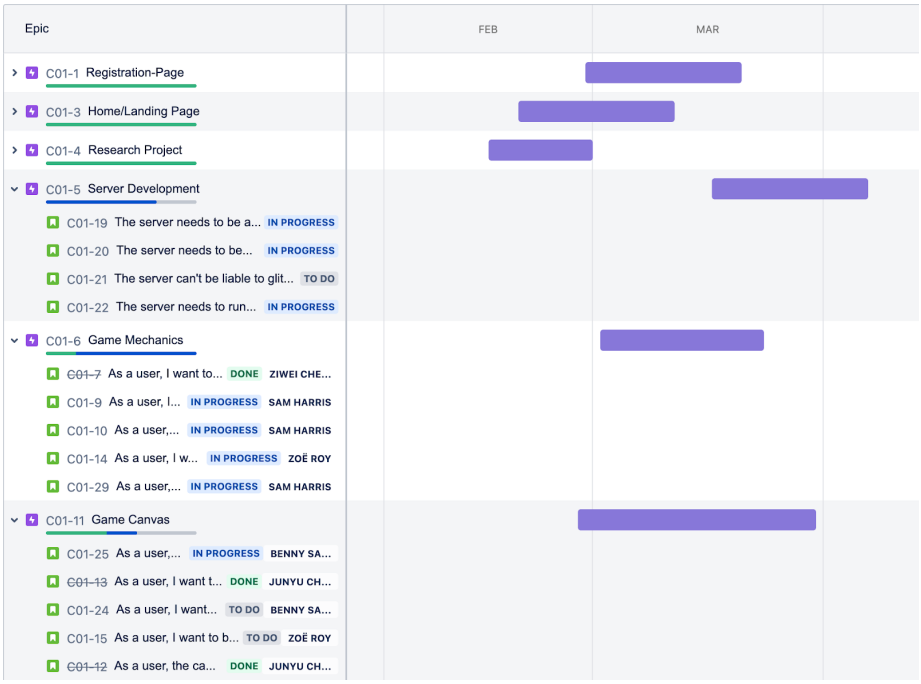
Individual Contributions:

- Zoe Roy: About Us, Game Rules Pages, Updating Jira Board.
- Ziwei Cheng: Created prototype_home and architecture diagrams. Styled front page after home, menu and login are joined together.
- Junyu Chen: Connect prototype_home (the pretty landing page) to the prototype (the working game). Prototype setup (socketio, express, html canvas…)
- Sam Harris: Projectile/bullet implementation, shooting mechanics (click to shoot, shoot in direction of mouse), working on bugs with bullets/projectiles, trying to implement sprites, but currently drawing all images to figure out proper speed and positioning of bullet.
- Aiden Colley: Player and gun movement
- Benny Sakiewicz: Create a database for storing user info (username, password, top 50 scores), as well as the top scores by all players.

Contribution Link:

Link: https://github.com/CSCI-3308-CU-Boulder/3308SP21_section013_1/commits/master

Jira Screenshot:



Challenges:

- Proper collision mechanics are hard to implement and design. It also seems as though there are limited Javascript libraries around collisions or collider objects. This means we would need to completely design, from scratch, a collision detection system.
- Connecting database and using passport. We are still finding ways to connect the database and store the user information. We also need to learn more about using passport to implement the user login.
- One challenge we will face is time and how much will be left to create the power up system. One of the last features that will be added are the power ups that players will be able to collect. However, with so much of our time going into the foundations of our game mechanics, there may be a limited amount left for this particular feature.

Risk Management Plan:

- We plan to focus on the foundations of our game with the highest priority. Every week we will discuss how much time each feature will take and if we do not see a way to create every feature within the time constraints, the team will make decisions about the necessity of each feature. If needed, features will be reworked or even removed.
- If passport ends up becoming more a hindrance, than a help, we will consider our options and decide how to connect the login to our user database. It is unlikely this challenge will not be solved.
- The database will be connected by the end of the project. In the meantime, it's important to create a comprehensive plan for our database and what we plan to do if it fails.