William McLaughlin
Same Britten
Kyle Leonesio
Joe Taylor

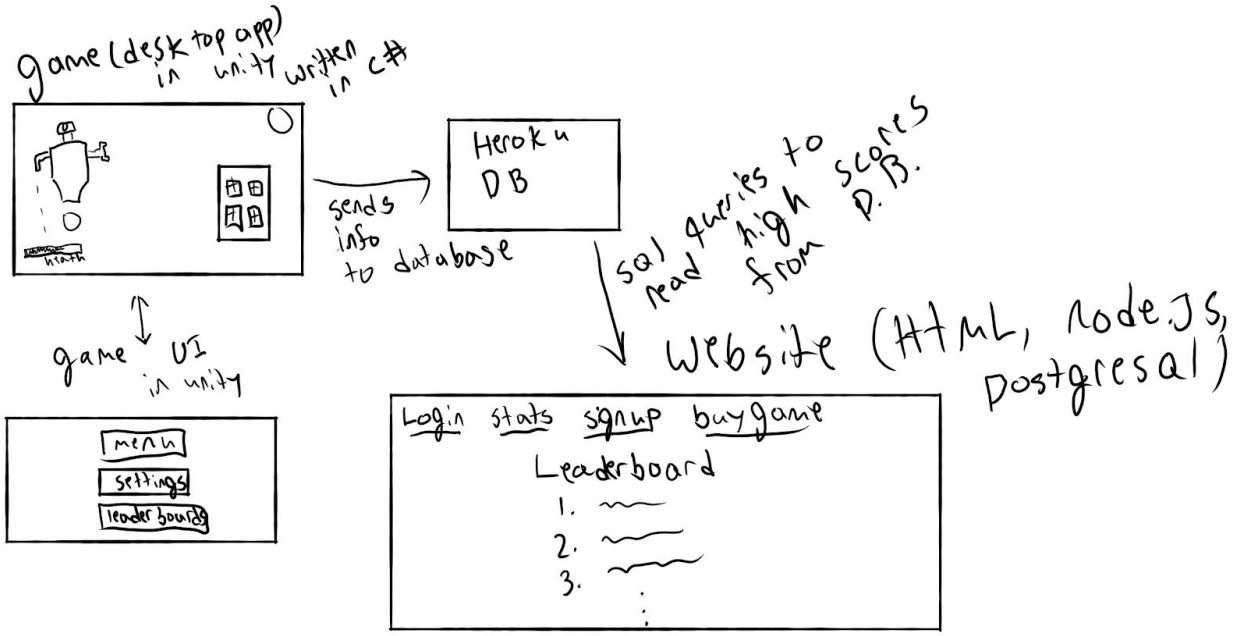
Project Milestone 4

Features

- Website:
    - A leaderboard page displaying the highest all-time scores in the game, pulls from the remote heroku database to stay updated.
    - A registration page so users can sign-up with an email and password. (To avoid bots and spam downloads etc.)
    - A wiki-like page with gameplay-related information
    - Game download page for the user to be able to download the game (must be logged in to do so)
    - (possible later) game development blog to show the games' design process
- Database:
    - High score storage with two columns: name [as input by user in game] and score achieved.
    - User registration information stored on a separate table.
    - Website will call to this database to update the leaderboard page
    - Game will prompt the user upon death to enter a name then send their high score with their name to the database.
- Player:
    - A character the player can move around the game environment
        - Collectible/unlockable other characters with different abilities
    - Weapon with which to shoot and kill enemies
        - Weapon variations for dynamic gameplay
- Level:
    - Enemies for the player to fight, and kill, to increase score
        - Variations in attack/speed/behavior
        - Have chance to drop items for player scaling
        - Scale up in damage/speed/hp/count over time so the game gets harder
    - Items for the player to pick up to scale and add variation to gameplay
        - Drop from enemies and/or collectable loot around the map
    - One relatively large map for the user to play through the entire game
        - Areas of different terrain/variation to keep it interesting as the player goes around the map
- Basic game functionality:
    - Created a build
    - Have enemy movement and tracking
    - Basic inventory
    - Start screen
    - Pickupable items

Architecture

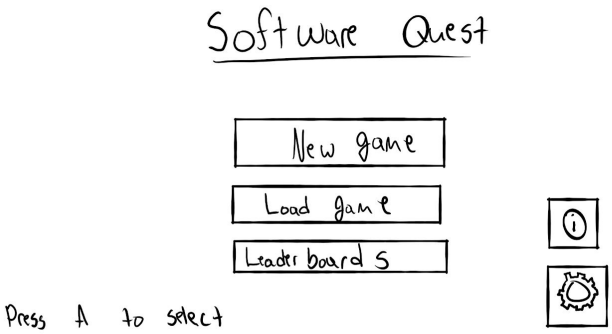
The architecture consists of the game, the website, and the database. The game will send scores to the database, and the website will retrieve them and update the leaderboards.



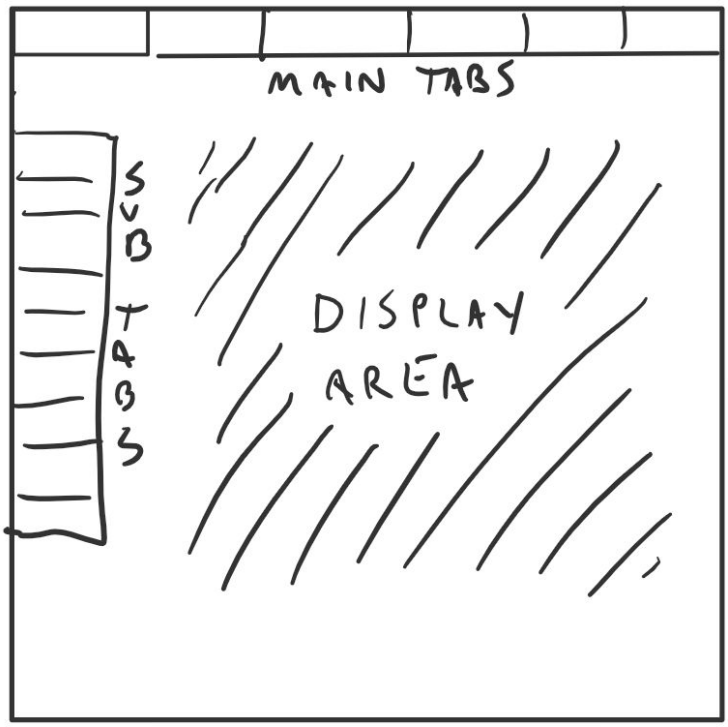
-
- GameOver screen that you can submit score from attached to a username

Front End design

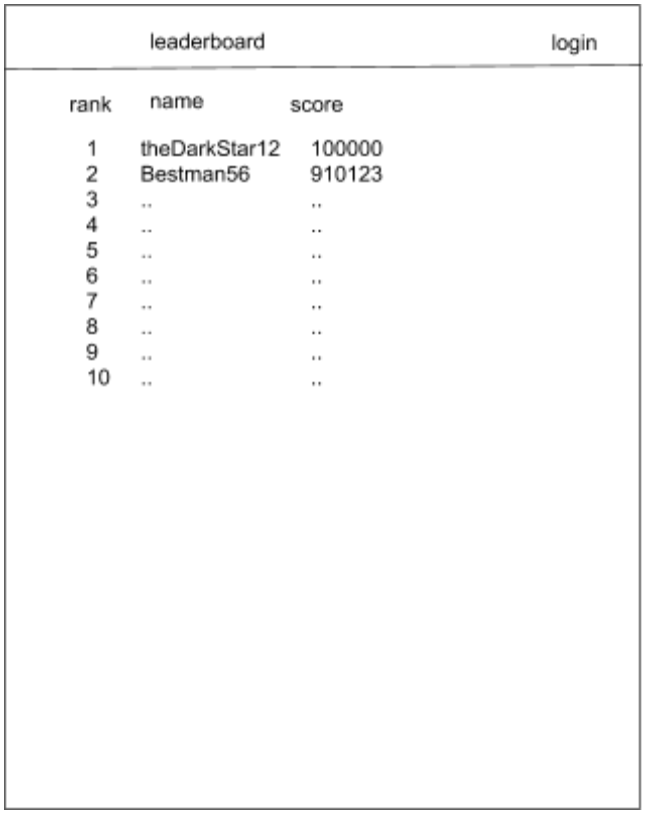
The front-end of our project is the game itself. This will consist of a main menu, multiple UI elements in-game, and the actual gameplay.



The general website design will consist of a display area and tabs on the top and the right to lead to other parts of the website



The front end of the leaderboard Page. Consists of three rows one for rank one for player name and one for score



Web Service Design

No web-service/API are being used.

Database design

PostgreSQL database hosted via heroku.

Table 1: High Scores

- High Score ID (PK, Auto-Increment Integer 1-100)
- Initials (Char 3)
- Score (Integer)

Table 2: Registration

- Username (PK, Varchar 45)
- Password (Varchar 45)

Individual Contributions

Joe Taylor:

- Created the webpage framework in react.
- Created the first iteration of the database in MySQL.
- Planned final database with high scores and registration information.
- Created basic website layout for final site.
- https://github.com/CSCI-3308-CU-Boulder/204-5/tree/master/Website

Kyle Leonesio:

- Created the builds of the game
- Created a hotbar and an inventory to store items
- Added functionality to be able to pick up items
- Created a start screen
- Added functionality to have certain objects or scripts persist between levels
- Installed system.data.dll
- https://github.com/CSCI-3308-CU-Boulder/204-5/tree/master/Code%20Components/kyle's_Code

Sam Britten:

- Created game assets in blender
- Worked on getting blender assets to work properly in unity
- Created shaders in unity for use on 3D models
- Finished enemy design
- Animated character
- Began implementing character in unity
- https://github.com/CSCI-3308-CU-Boulder/204-5/blob/master/assets/bot.blend

William McLaughlin:

- Worked on initial teams set up for unity collaboration
- Created initial player character to walk and shoot
- Added enemies that can be hit by the player [targets]
- Added a way for enemies to follow the player around the map [with pathing]
- Began work on weapon variations - projectile hit detection over raycast
- Began design on items and scaling possibilities
- https://github.com/CSCI-3308-CU-Boulder/204-5/commit/e8772d48a803b17cb203c48dca54c634b191a627

Challenges

1. Database connection to game. Having no experience with making a unity game connect to a remote database makes it difficult, though we have found many resources to help do it and this is our current main focus.

    a. Backup plan: Either have a manual entering of scores or force the game design process into a web application.
2. Integrating the website with the database and allowing players to view a variety of game data.
    a. If this proves difficult, we will go with the simplest possible variation and only pull high scores from the database.
3. Splitting development into modular pieces that can be divided amongst our team. Learning modular website design such as EJS, to allow for simultaneous development of modules by multiple team members.