Group 102-5 MILESTONE 4

Revised List of Features

- Physics Engine - Priority: HIGH
    - Provides an approximate simulation of physical phenomenon (in this case gravity, velocity, acceleration, normal force). Allows in game characters to jump, land, run etc. in a realistic manner.
    - The physics engine is a high priority because every feature relies on it to translate the user input into a visual representation for the user.

- Game Mechanics - Priority: HIGH
    - Working hand-in-hand with the physics engine, the mechanics make up how the game operates and what the user needs to do to interact with the game itself.
    - The game mechanics are the rules that dictate how the user will interact with the game - including input recognition, object interaction, data handling (including integration with leaderboard)

- Leaderboard (and what it involves) - Priority: HIGH
    - Keeps track of, and ranks high scores of players
    - Working to read data from a csv into the database to be sorted and displayed on Leaderboard.
    - The leaderboard is one of the main features of the project alongside the game itself and as such is considered to be high priority.

- Levels - Priority: MEDIUM
    - Levels will be either in a sequential order or have branches to create a web of levels that the user can interact with. Levels help keep the attention span of the users, create navigable space (thereby giving purpose to the physics engine), and make the game's mechanics enjoyable.
    - The level is something we will adjust based on how object interaction, physics limitations and game mechanics function together, making levels a medium priority for the moment. In other words, while there will certainly be levels, their complexity may change based on time constraints.

- Items/Power Ups - Priority: MEDIUM
    - Gives the user the opportunity to interact with items that are placed on the map to enhance their movement speed, extend their time window or even give them an extra life
    - The items will be objects that trigger game mechanics, physics changes, data entry and game status changes.

    - These are a medium priority because they are essentially a secondary user input, the users interaction with visible objects in the game will be how functions are called within the game, therefore we can use them as place holders for the time being until the mechanics are ready to be integrated with the user input.

- Music/Sounds - Priority: LOW
    - Governs how music will be integrated into the gameplay experience, and how sounds will be implemented (and activated) upon interaction with environments.
    - Similarly to the Art style feature, sound design is a feature that can be handled as other features are finished.

- Art Style/Graphics - Priority: LOW
    - Dynamic graphics that change as the player transverses, and interacts with their environment.
    - Actions taken by the player should match the movement so that the user understands what their inputs are doing. e.g jumping, running and picking up items animation.
    - This feature is a lower priority because it is a luxury feature that the functionality of the game depends on. More time investment means smoother animation; animation may be crude to non-existent otherwise.

Architecture Diagram

Using draw.io to draw a diagram of the architecture
https://drive.google.com/file/d/1yU9EiCBjybx4Y-ZEyaWWs7fHP1LhWqQP/view?usp=sharing

Front End design

Using figma to give a front end wireframe:
https://www.figma.com/file/JADUzCv0y1WztIJMzKglQB/Untitled?node-id=0%3A1

Notes: We are assuming we are going to inbedd the unity game in the html website as it makes it easier to develop leaderboard

Database design

- We will be using a simple PostgreSQL database to hold the players' usernames and high scores.
    - We can expand this as much as we want (add additional specific stats about the players' games), but for now we are sticking to a leaderboard with names and scores solely.

- Using PHP, we can integrate an SQL database into Unity, in which Unity can both retrieve data and add to it with the PHP script.
    - We will use either JavaScript or C# to have a controller that actually loads in scores and connects to the database through the PHP script.
- We figured we did not ned to include an entity diagram, as there is only one entity (the score itself) with a primary key as the username (as a string), and the score as an integer.