Use Case: Login With Saved Progress

Primary Actor: Player/User

Stakeholders and Interests:

- Player: wants to log into the game with a tracked high score and progress so as to compete against others and enjoy the achievement of having progressed through the game.
- Developers: wants a system to accurately, reliably, and securely save the user's information so as to ensure the consistent return of the user to progress further in their game. Also wants to observe the ease or difficulty with which players play the game. Statistics like time to beat a level, frequency of gaining points, and magnitude of high scores are all aspects a developer might desire to observe in order to improve the game and its appeal to the user.

Preconditions:

- User successfully signs up with an arbitrary username and password

Success Guarantee(Postconditions):

- Player's high score will always be maintained and tracked. Player's level will always be saved and they will always start at the level they left at with all level parameters such as difficulty or reward frequency intact

Main Success Scenario

1. User comes to title page and clicks the start button or redirects themselves to the login page via the sign in page
2. User enters their username and password
3. Both the username and password are correct
4. The user is authenticated and is redirected to the main game

Alternative Flows:

Alt1: User does not enter correct credentials

1. The login module will reject the credentials and will not send the user to the game

Alt2: User has not created an account

1. The login page will ceaselessly reject whatever credentials the user puts in until they enter the credentials of some other existing user, assuming one exists
2. The user may go to the sign in page via a link provided at the bottom of the login module

Exceptions:

If the system runs into any exception such as attempting to access a locked database, the system will report the error on the page

Special Requirements:

The site must inform the user of an incorrect credential entry upon every trial. The login module must be stylized beyond a bare bones information entry form

Open Issues

- What can the user do if they forget their password?
- How can the user change their credentials if they wish to?

Use Case: Occasionally Load a New, Randomly Generated Map

Primary Actor: Player/User

Stakeholders and Interests:

- Player: theoretically wants a more dynamic and stimulating gameplay which can be created via the occasional creation of a new, unpredictable map for the player to play on.
- Developers: wants a system to efficiently load a map for the player with the time taken being enough to show the player some interesting map generation visuals but not so long as to bore or agitate the player. This helps both the servers and the player. Also want to prevent any time based advantage for a player who might simply take time to memorize a map and use that to find the best paths to traverse as they play the game

Preconditions:

- User successfully logs in with an arbitrary username and password

Success Guarantee(Postconditions):

- The user will watch as a new map is generated quadrant by quadrant until all 4 quadrants are made. At that point, the user will be redirected to play the actual game

Main Success Scenario:

1. User logs in
2. User watches as map is generated
3. User is redirected to the game

Alternative Flows:
N/A

Exceptions:
If the user's browser fails to load the map, the next user to authenticate into the website will be responsible for loading the map

Special Requirements:
The site must load the map efficiently and lightly. This means the loading must be somewhat fast and must not tax the user's browser too much

Open Issues

- What map size is optimal to load?
- What to do when a user disconnects before loading a map?

Use Case: Play Spawn(The Game)

Primary Actor: Player/User

Stakeholders and Interests:

- Player: wants a fun and challenging game to play possibly involving some competition with others.
- Developers: want players to enjoy a game, compete, and keep returning to progress further

Preconditions:

- User successfully logs and possibly loads a map

Success Guarantee(Postconditions):

- The user will play the game
    - The goal of the game will be to gather enough points to move to then next level without touching invalid objects,
        - Invalid objects include adversaries, your own tail, which grows as you gather points, and walls, which will be at random positions on the map
    - Difficulty scales with level
    - Each level has adversaries that will randomly spawn across the map
    - Coins will also randomly spawn on the map. Touching these coins is how points are acquired
    - The user's points, level(quadrant) and high score will be saved for the next time they return
    - Once the user reaches level 4, it is essentially a game of endurance as they will not go higher than level 4

Main Success Scenario:

4. User logs in
5. User possibly loads a map
6. User is redirected to and plays the game

Alternative Flows:
N/A

Exceptions:
Exceptions or bugs during the game will ideally be displayed on screen and handled internally

Special Requirements:
The game must be efficient and non-tasking for the browser to run

Open Issues

- How exactly should difficulty scale?