

CosmicRace

Ran Nissan | Amit Swisa | Dvir Magen

Project description

In this project we will realize a game, which will be presented in 2D. This project describes a game that developed in Unity, called Cosmic Race. The game is a running contest, that the purpose of this game is get to the finish line first, but it isn't that easy. In the race the player will have to avoid obstacles, and attacks from the other players that trying get to the finish line before him.

Game Process

When the player starts our game – he will need to select one of two options modes that he would like to play in.

The first option is Online mode. In this mode, the player will have the opportunity to play against people all over the world and show them who is the best.

The second option is called Friends mode. In this mode, friends that stays at the same place, and have a computer in their area will be able to start and compete against each other.

In order to start a game in this mode – they first have to generate a room number from the computer, and then each of the participants needs to browse CosmicRace's website in his mobile and enter the room number. After all the participants is in, they need to click Start option from the computer, and each of the participants mobile will now be his game controller.

In our game. The players will compete each other in order to get the finish line first, but skills and technique is not the only things they need to have in order to win the contest.

Each player will have the opportunity to improve his skills by collecting coins that will be placed all over the map, which he can use them later in the shop in order to improve it stats (speed, jump and more).

Project Architecture

Since in our project we have two modes – Online and Offline, we had to use two communication protocols.

For the online mode we use Sockets – that communicates with our game server. We decided to use sockets in order to give the player the best game experience and prevent lags that hurt game experience.

For the offline mode we use Web Sockets, for the same reason.

Since the logic of the code is similar between the online mode and the offline mode, we decided to create our main classes abstracts, in order to run the game in the same way, and with a little bit change of code for the different protocols.

The game server is written in Java, and dealing with regular game issues like timeouts, lags, and interrupts.

The game itself developed in Unity – a cross-platform game engine. The code written in C#.

We also used Azure to store our game data in Database and introduce our servers. Our Web server is written in NodeJS, and he is handling our database operations.

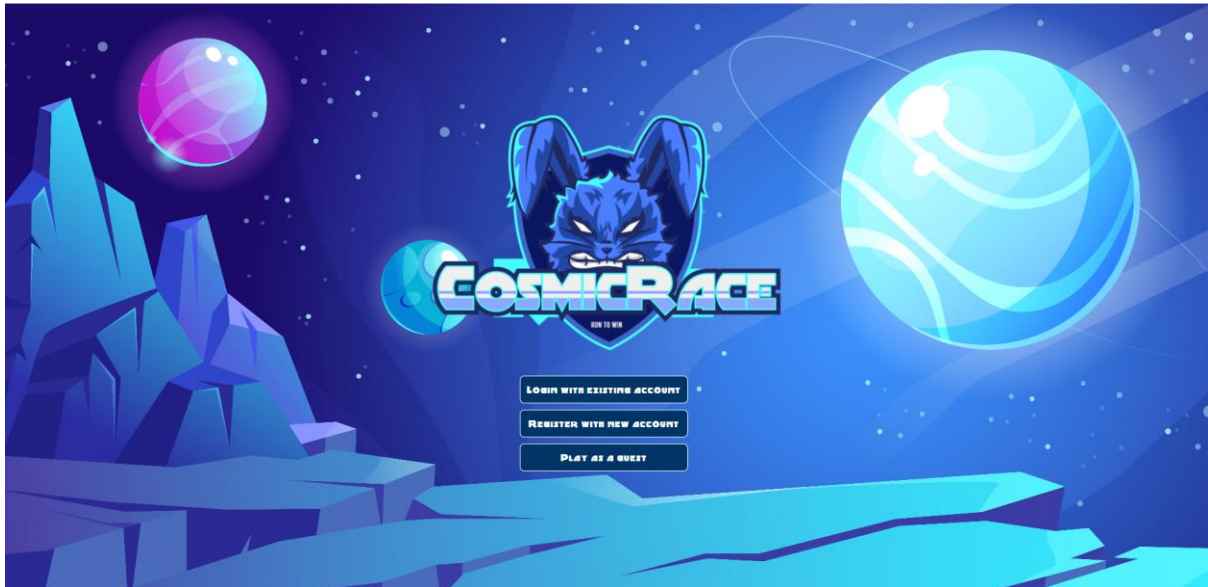
The CosmicRace's Web App written in ReactJS. Base web application for register, login, view match history and play offline mode.



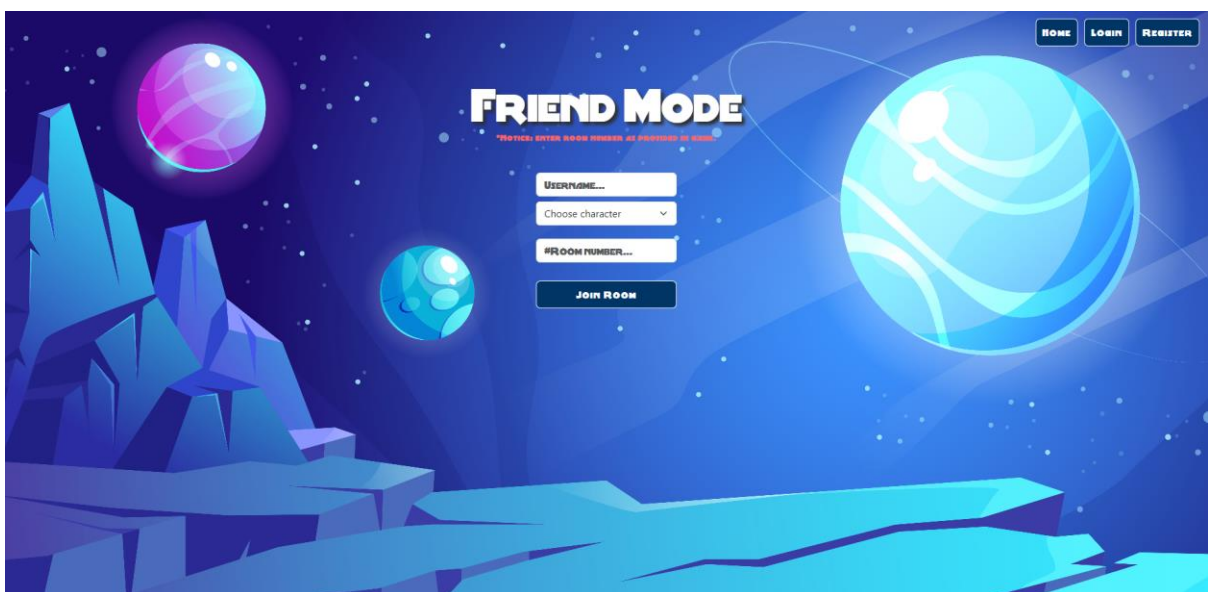
Scenes

Web:

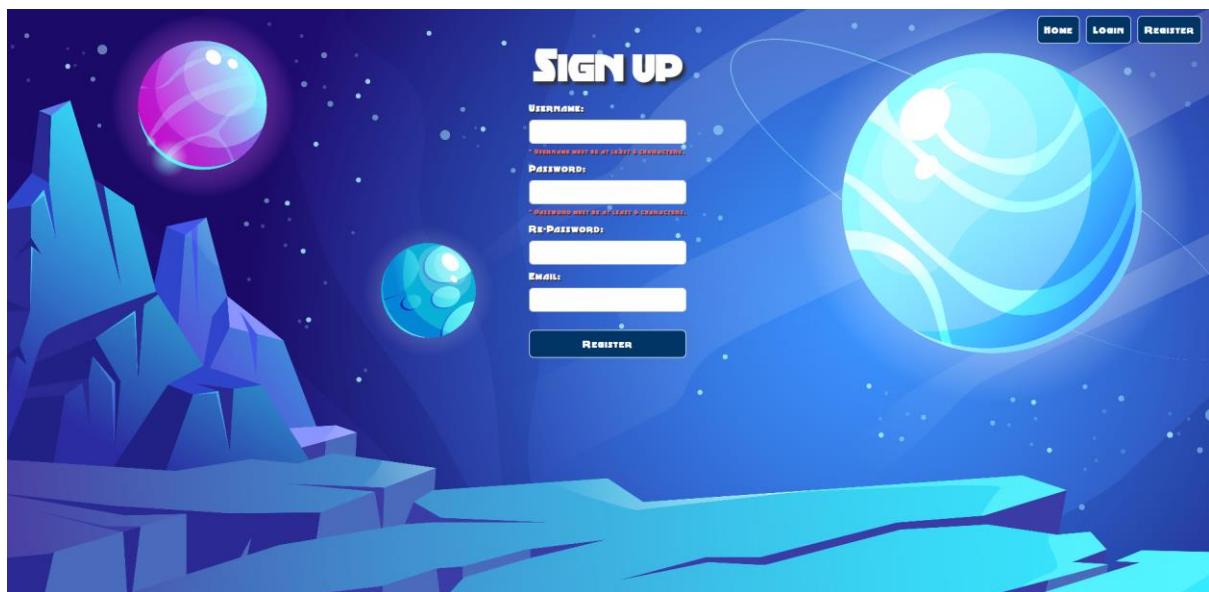
Login Page:



Friend Mode:



Registration:



The registration form is titled "SIGN UP" and is set against a vibrant space-themed background featuring a large blue planet, a pink planet, and rocky terrain. The form includes fields for Username, Password, Re-Password, and Email, each with a small red error message below it. A "REGISTER" button is at the bottom. Navigation links for "HOME", "LOGIN", and "REGISTER" are in the top right corner.

SIGN UP

USERNAME:

* username must be at least 6 characters

PASSWORD:

* password must be at least 6 characters

RE-PASSWORD:

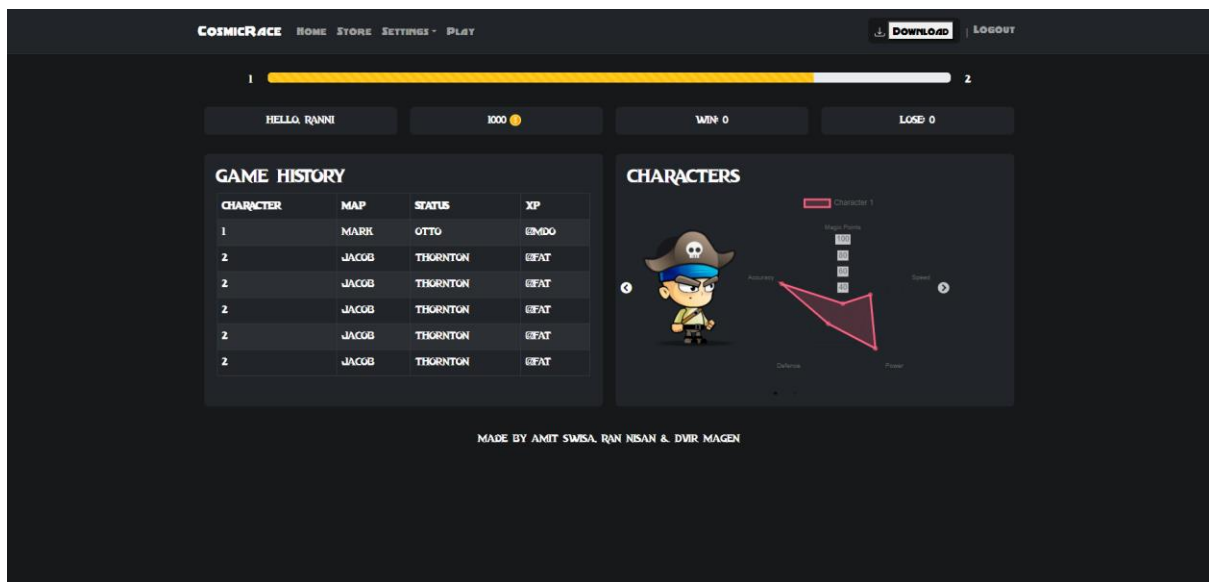
* password must be at least 6 characters

EMAIL:

REGISTER


[HOME](#) [LOGIN](#) [REGISTER](#)

Home Page:



The home page dashboard for "CosmicRace" features a dark theme with a navigation bar at the top. The main content area includes a progress bar, a greeting, a balance display, and two main sections: "GAME HISTORY" and "CHARACTERS".

CosmicRace [HOME](#) [STORE](#) [SETTINGS](#) [PLAY](#) [Download](#) [Logout](#)

1  2

HELLO, RANNI 1000 WIN 0 LOSE 0

GAME HISTORY

CHARACTER	MAP	STATUS	XP
1	MARK	OTTO	ENVOO
2	JACOB	THORNTON	BEAT
2	JACOB	THORNTON	BEAT
2	JACOB	THORNTON	BEAT
2	JACOB	THORNTON	BEAT
2	JACOB	THORNTON	BEAT

CHARACTERS

Character 1

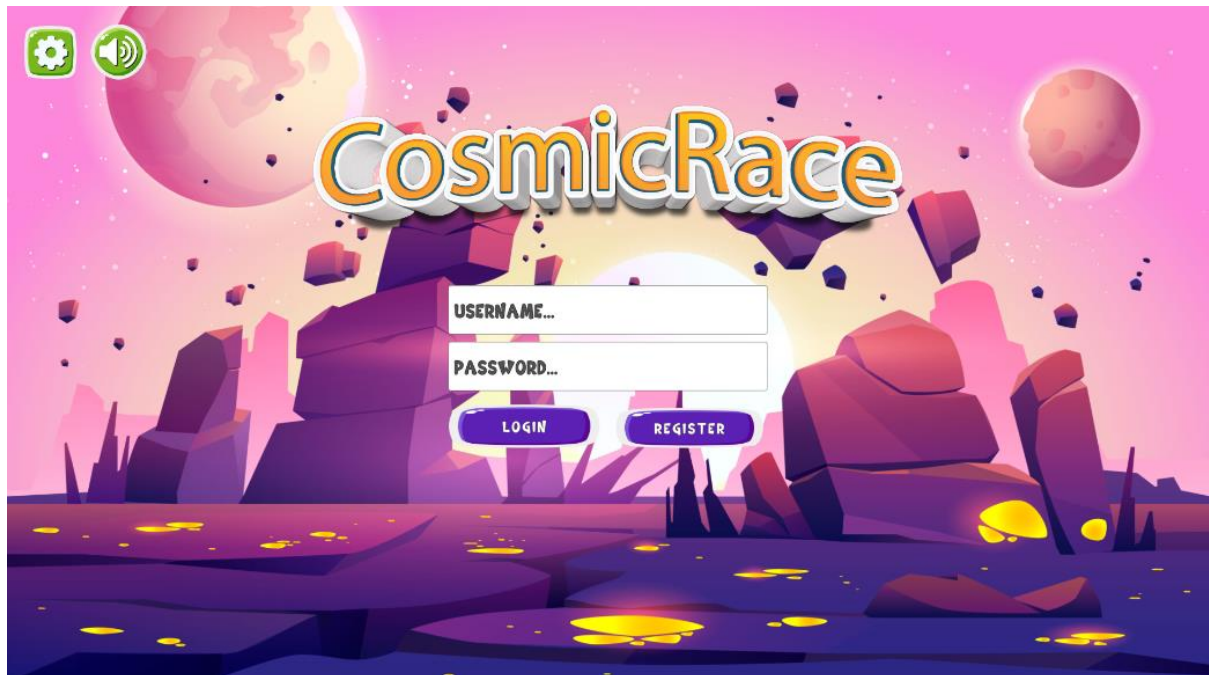
Attack Power

Defense Power

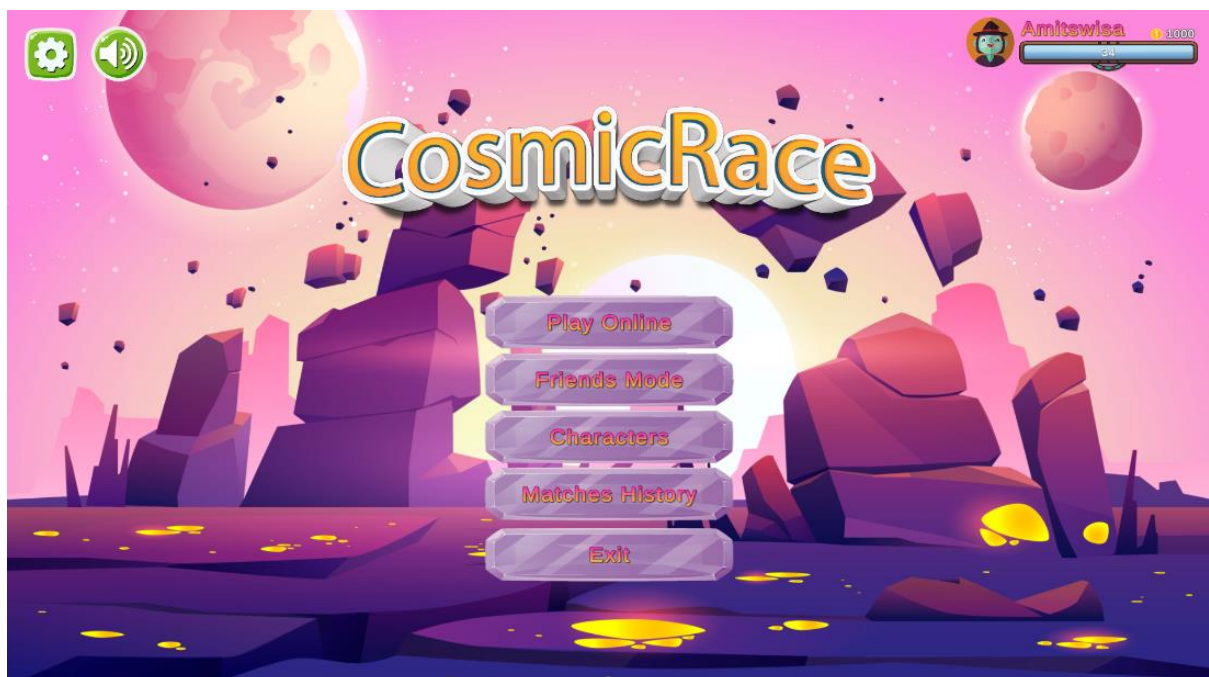
MADE BY AMIT SWISA, RAN NISAN & Dvir Magen

PC:

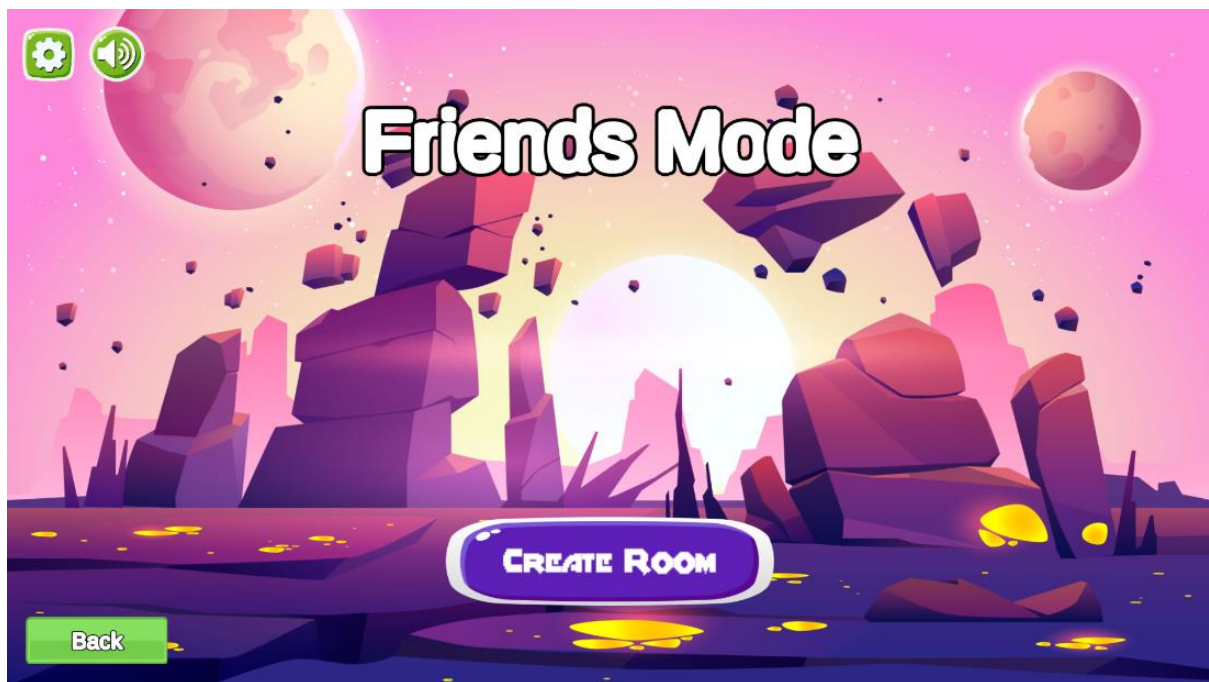
Login Page:



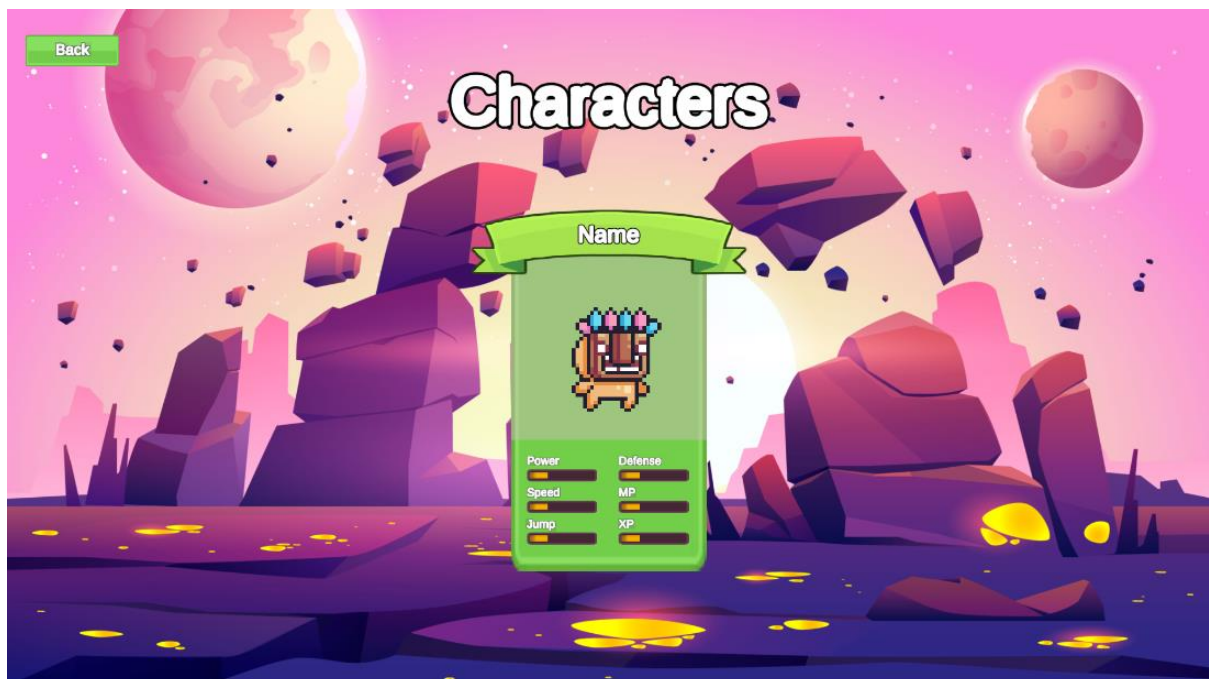
Home Page:



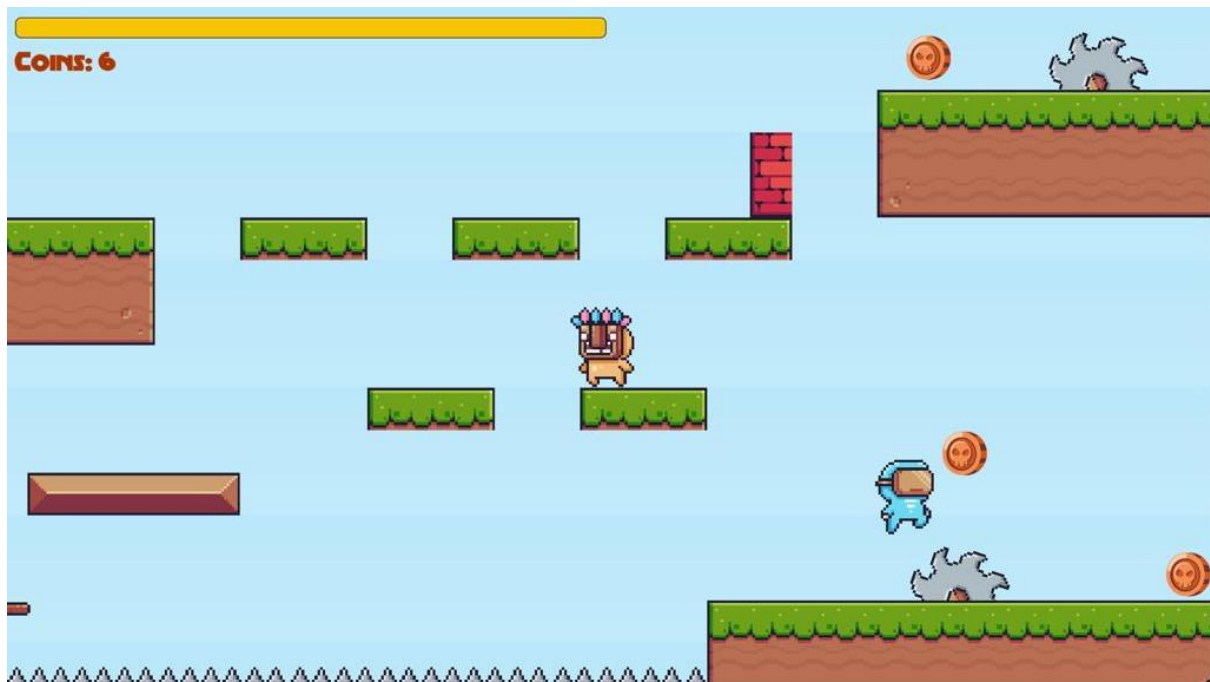
Friends Mode:



Character Info:



Game Scene:



Class Diagram

WebSocketServer- server-side implementation that enables real-time, bidirectional communication between clients and the server over WebSocket protocol.

GameServerSocket- server-side implementation that facilitates real-time, bidirectional communication between clients and the server using regular sockets for multiplayer gaming applications.

MatchService- responsible for managing and coordinating multiplayer game matches, handling player matchmaking, game session creation, and ensuring fair and balanced gameplay experiences.

PlayerEntity- representing a player or character in a game, containing relevant attributes for their state and interactions.

ConnectionModel- used to manage and store information about network connections, including details such as connection status, IP addresses, and other relevant data for communication between different devices or systems.

