SOUL DEFENDERS

GROUP MEMBER:

- TRẦN NGUYÊN KHÁNH – ITDSIU22171
- ĐẶNG HUỲNH MINH PHÚC - ITITIU22126

SOUL DEFENDERS
A tower defense game

Strategy **games** where the goal is to defend a player's territories or possessions by obstructing the enemy attackers

Table of content

I / Introduction

- Goals:

Variety of Tower and Enemy Types: various types of towers and enemies.

Fun of the game: Entertaining, challenging, and interesting.

Intuitive Interface: simple and readable.

Replay Value: Players are able to edit and create custom level

Optimization: Lightweight, minimal bugs or performance issues.

I / Introduction

Objective:

- Improve Java programming skill
- Understand the OOP concept
- Another way to approach programming

II / Methodology

Application

- Coding:
  - Visual Studio Code
  - IntelliJ IDEA

- Drawing Sprite:
  - Aseprite

Website

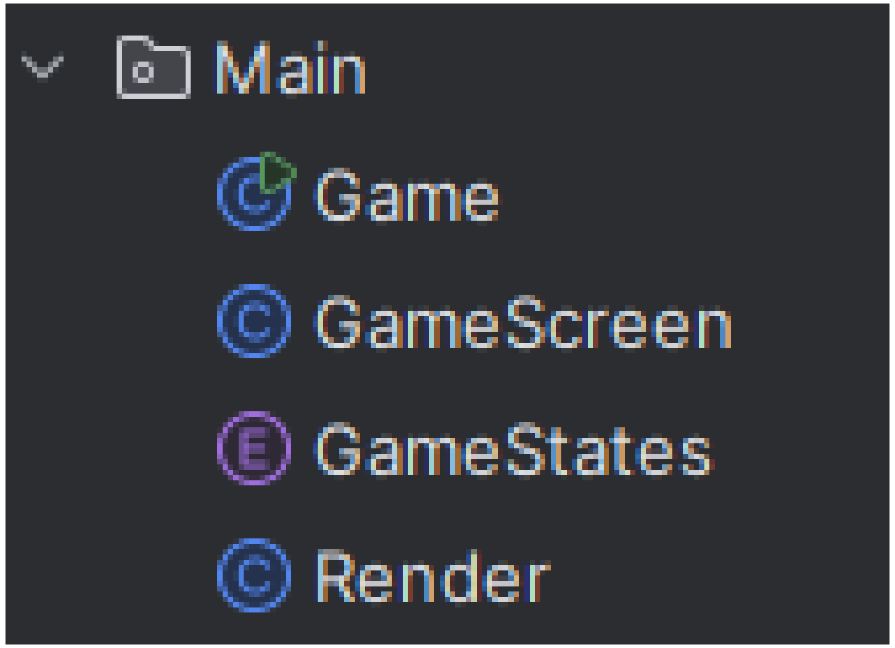
- Draw.io
- ERDplus
- Github

Communication

- Discord
- Messenger
- Google Drive (for file)

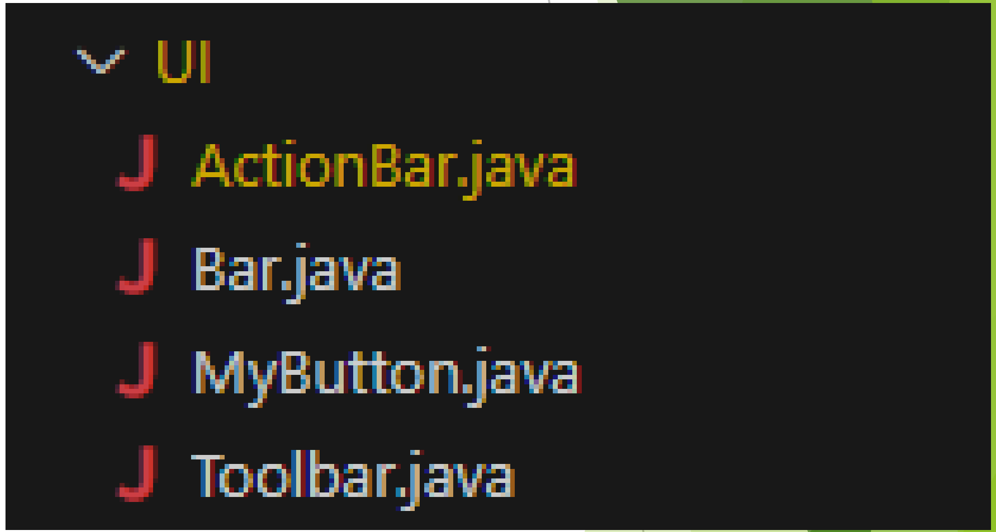
MAIN

- Game: the main class of the game.
- GameScreen: handling display and managing input events from the user.
- GameStates: defines the different states of the game.
- Render: allows to render depending on cases

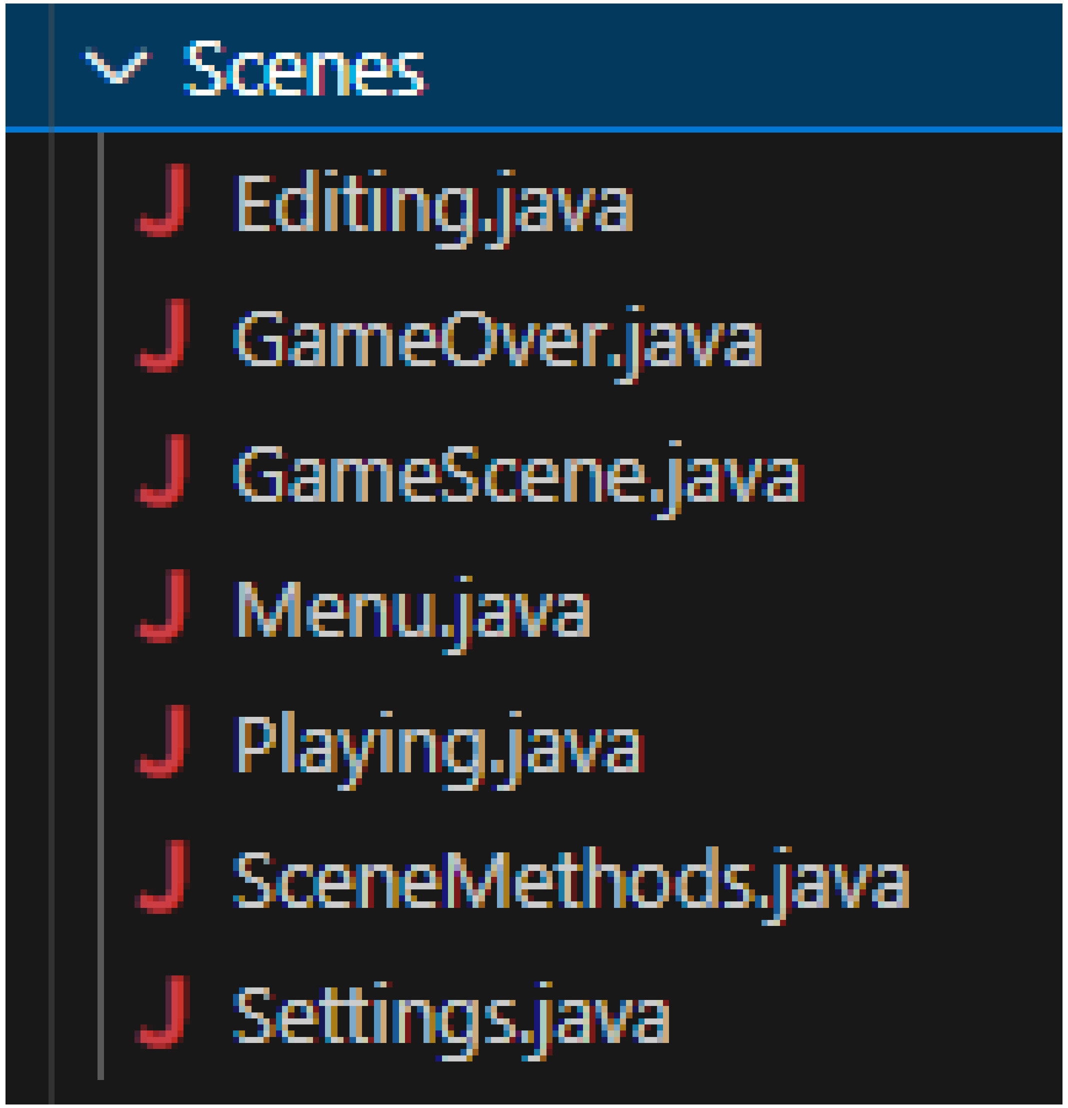

Main
Game
GameScreen
GameStates
Render

UI

- Bar: UI elements that represent a rectangular area on the screen (action bar and toolbar).
- MyButton: represents a clickable button in the user interface
- ActionBar: extends the Bar class and represents a specific area of the UI that displays gamerelated
- Toolbar: extends the Bar class and represents a specific area of the user interface dedicated to editing functionalities



```
UI
  ActionBar.java
  Bar.java
  MyButton.java
  Toolbar.java
```



Scenes

Editing.java

GameOver.java

GameScene.java

Menu.java

Playing.java

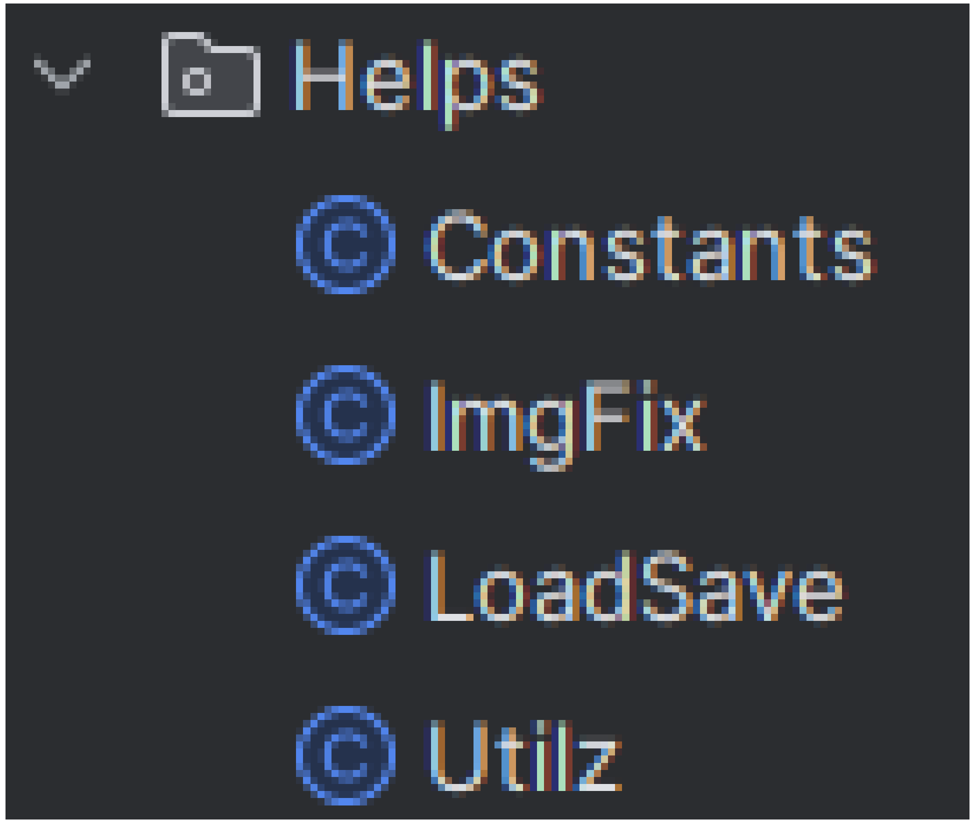
SceneMethods.java

Settings.java

SCENES

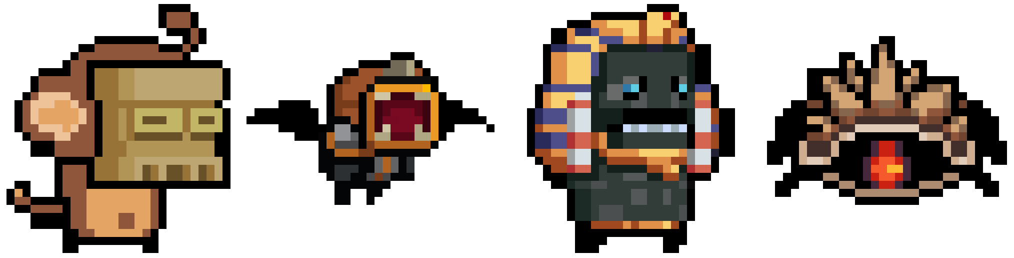
- ScencesMethods: Interface that contains mouse input methods and render method
- GameScene: Getting sprite from atlas by getSprite() and running a loop of animation with updatetick()
- Menu: Draw buttons “Play”, “Edit”, “Settings”, “Quit” and background
- GameOver: draw Menu and Replay.

HELPS

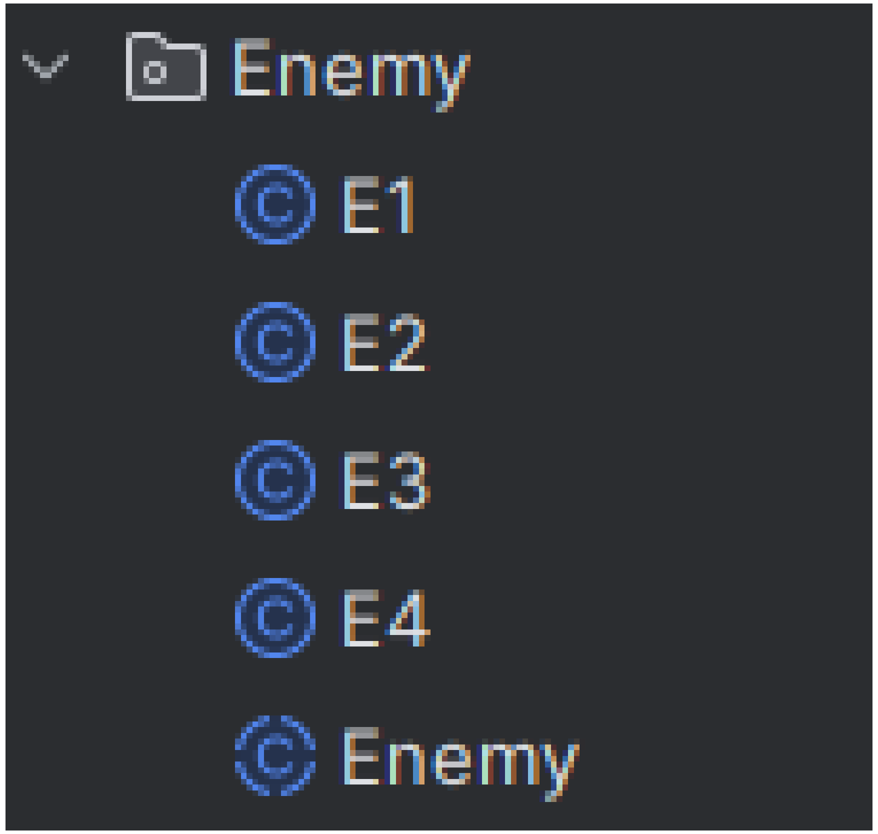


Helps
Constants
ImgFix
LoadSave
Utilz

- Constants: sets statistics for both towers and enemies
- ImgFix: provides methods for image manipulation, including rotation and image construction from multiple images
- LoadSave: provides methods to load and save data for the game
- Utilz: provides methods for handling arrays and calculating distances.



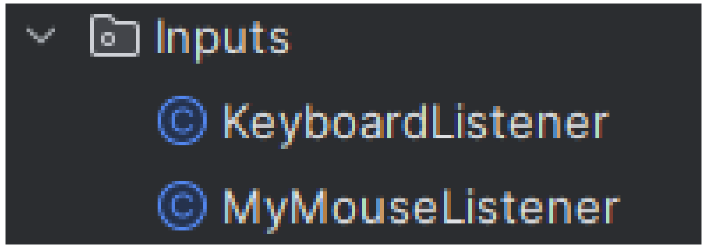
ENEMY



Enemy
E1
E2
E3
E4
Enemy

- E1, E2, E3: defines kinds of enemy.
- Enemy: abstract class for E1, E2, E3, define their statistics and behavior.

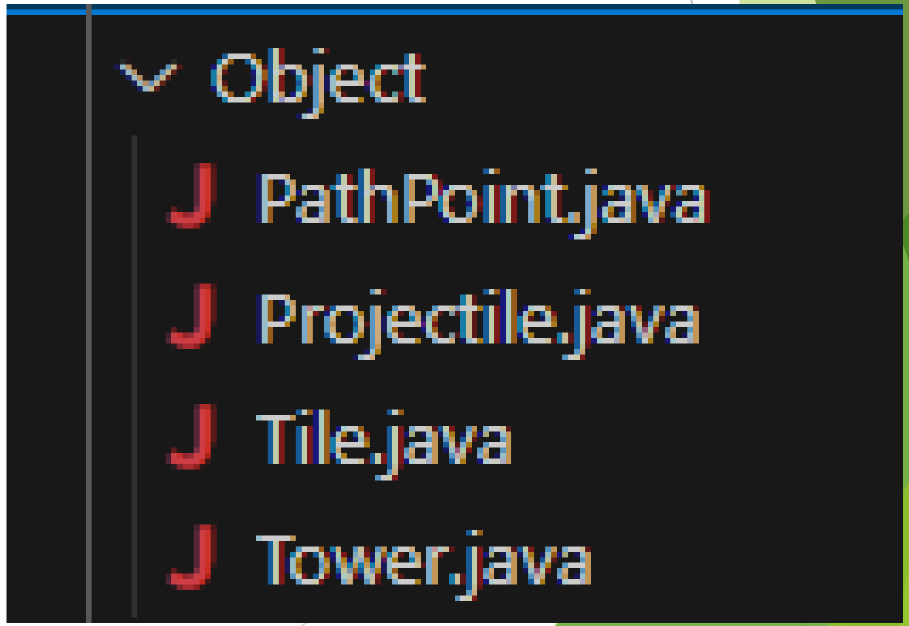
INPUTS


Inputs
KeyboardListener
MyMouseListener

- KeyboardListener: Making methods for keyboard inputs
- MyMouseListener: Making methods for mouse inputs

OBJECT

- Pathpoint: represent a point in a 2D space using two coordinates: **xCord** and **yCord**.
- Projectile: represent a projectile
- Tile: represent a tile, used to create the game level environment.
- Tower: manage towers, with different attacks and types. Handle attack cooldowns, upgrades, placement

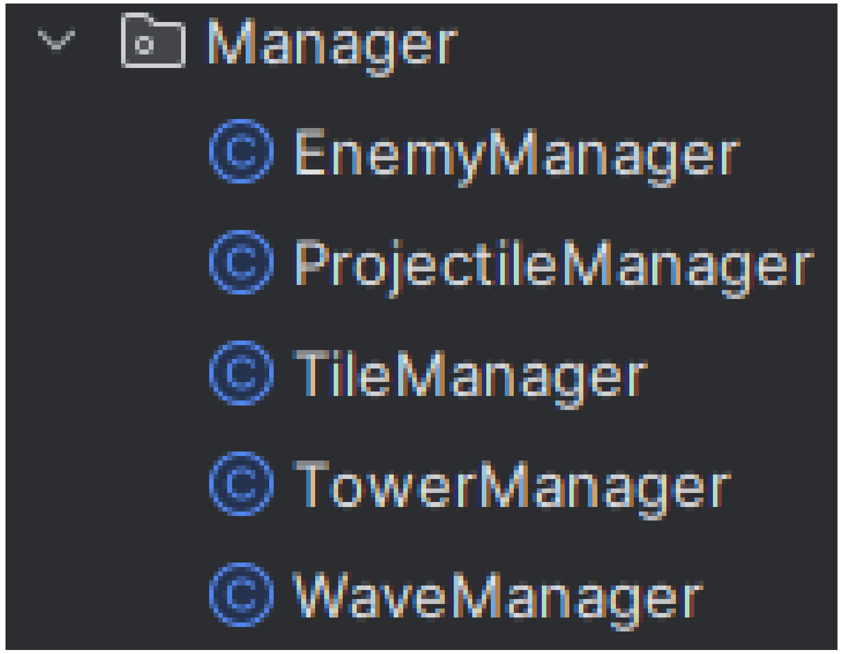


```
∨ Object
    J PathPoint.java
    J Projectile.java
    J Tile.java
    J Tower.java
```

EVENTS

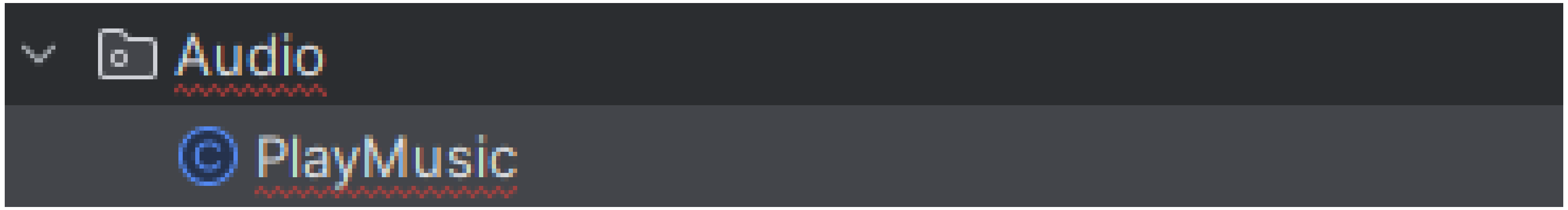
- Wave : defines a enemy wave

MANAGER

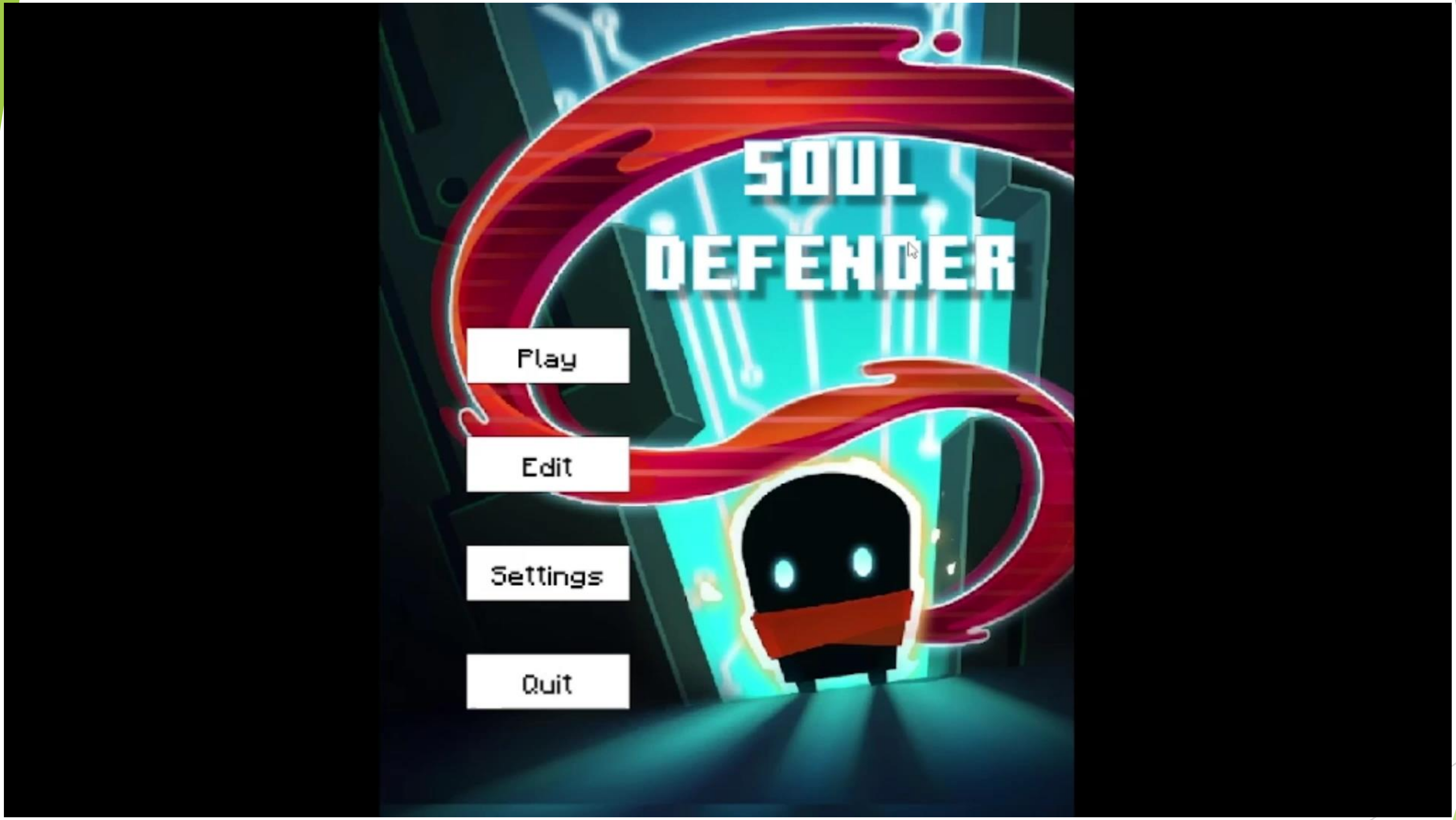

Manager
EnemyManager
ProjectileManager
TileManager
TowerManager
WaveManager

- EnemyManager: manages enemies in the game.
- ProjectileManager: manages effects of shooting and explosions
- TileManager: manages kinds of tiles in the game.
- TowerManager: manages kinds of towers.
- WaveManager: manages waves of enemies.

AUDIO



- PlayMusic: Defines method to play, stop, and continue music



RESULT

CONCLUSION & FUTURE WORKS

- Summary:
  - Experiencing programming game
  - Understand core concept of OOP

- Future Works:
  - Adding more features
  - Working on animation
  - Fixing bugs

THANKS FOR LISTENING