SOFTWARE REQUIREMENTS
AND
DESIGN SPECIFICATION

Video Game:
Punchman

Fight Club
Joseph Johnson, Isaiah Dorsey, Nasir Hill, Demarieo James

4/12/2024

Table of Contents

1. Planning

1.1 Project Details

1.1.1 Project Title: Punchman
1.1.2 Team Name: Fight Club
1.1.3 Time frame: 1 year
1.1.4 Project Contacts:

- Isaiah Dorsey/ 2055310234/ idorsey@bulldogs.aamu.edu
- Nasir Hill/ 7063589099/ nasir.hill@bulldogs.aamu.edu
- Demarieo James/ 3347824411/ DemarieoJ2@gmail.com
- Joseph Johnson/ 2055413301/ jjohn361@bulldogs.aamu.edu

1.2 Project Summary

1.2.1 Background and Justification

- The history of our project is we wanted to learn how to make a video game. Our project doesn't have any problems to solve

1.2.2 Project Objectives

- Understand the concepts of how to make a video game
- How the process of making a video game works
- Use the skills we will learn to make a video game

1.3 Project Methodology

1.3.1 Project Approach

- We start off researching what we need to know about making a video game.
- We then list what we need to learn and how much time we need to learn it.
- Document everything we need to know and what they are.
- Come together and try to use the skill we learned to make a rough draft version of the game.
- Finalize the game by fixing any bugs and errors that might be happening in the game.
- Keep providing support for the game after finishing it.

1.3.2 Work Breakdown and Task Estimates

- Framework:
  a. Communication
    i. Talk to the Customer (3 hours/ $0 required)
      1. What type of game is wanted?
        a. Work Product: The customer wants a 2D fighting game with multiple different levels and stages.
      2. How much time is given to make it?

a. Description: The customer has given us a full year to make the video game.

3. What features are wanted for the game?
    a. Work Product: The customer wants the fighting to be simple, but fun(ex. Pressing the X button makes the character do a 2-hit combo). They also want the character to be able to use different weapons depending on the user's preference. There will also be fighting requirements such as: “Must have a sword equipped for the fight”

ii. Talk to the Team (2 hours/ $0 required)

1. What software can be used to make the game?
    a. Work Product: The team did research about what software would be best suited to complete the task without spending money. They concluded on using GODOT4.
2. How can we split the tasks up for the job?
    a. Work Product: We separated the task for the job based on the skills from each member of the team. The tasks fall under the Front-End or Back-End.
3. What can be done to make the job less stressful?
    a. Work Product: Make a work deadline for the tasks that need to be completed.

b. Planning

i. Project timeline

1. What parts of the job do we need to have done by a certain day?
    a. Work Product: We need to have all learned how to use GoDot 4 before our project can begin.

c. Modeling

i. Skills/software requirement (4 hours/$0 required)

1. What type of skills do we need to complete the job?
    a. Work Product: We needed coding skills with C# and C++. We also needed skills in GoDot 4 because that is the software we are using for our project.
2. What software can we use to complete the job in the time given?
    a. Work Product: We are using GoDot 4 because it is free to use and gives us a wide range of resources to use.

ii. Design (1 month/ $0 required)

1. Making the Front-End of the game
    a. Work Product: We start making the game from the user's point of view. We start making the layout of the

video game and the character’s looks and different moves. We would make all the video game’s necessary things that don't require coding.

d. Construction
    i. Code Generation (1 month/$0 required)
        1. Coding the Back-End of the game
            a. Work Product: We begin coding the Back-End of the video game. This allows us to start the actual process of making the game fun for the users. We start making the levels of the video game and how the fighting is going to work. We can generate how the opponent will fight depending on the difficulty of the fight and the stage that the user is on.
    ii. Testing (5 Days / $0 required)
        1. Testing out game
            a. Work Product: We start testing our game to look for bugs or errors that we need to fix before we give the game to the client. This allows us to make sure the customer is satisfied with the product before delivering it to them.

e. Deployment
    i. Uploading PunchMan
        1. Upload to github
            a. Work Product: We will upload the game Punchman to github as a playable game downloadable for computer and Mobile.

- Umbrella:
- Software Project Management
    - We had to manage how the work needed for the project to be distributed among the team members
- Formal Technical Reviews
    - We tested each screen before we added it to the main game to ensure it worked.
- Software quality assurance
    - We had to test each feature to make sure it did as it was intended to do. Once it passed that, It was added to the main game and tested again.
- Software Configuration management
    - We had no need to use this activity
- Work product preparation production
    - We had a spreadsheet of what was done and what needed to be completed.
- Reusability management
    - There wasn’t anything that was able to be reused.
- Measurement

    - We measured the time it would take do the tasks; however, there were some tasks that exceeded our measured time.
- Risk management
    - We took a risk trying to do a leaderboard, but wasn't able to do it.

- Project Deliverables
    - Documentation
    - SlideShow
    - Github with the code

1.4 Project Risk Management

1.5 Milestones

1.5.1 Completed the first Demo in November 2023.
1.5.2 Score system: February 2024.
1.5.3 Game over Menu: March 2024

2. Software Requirements List

- GoDot 4 (free game development software),

- Godot Docs (server needed for the game),

- Domain

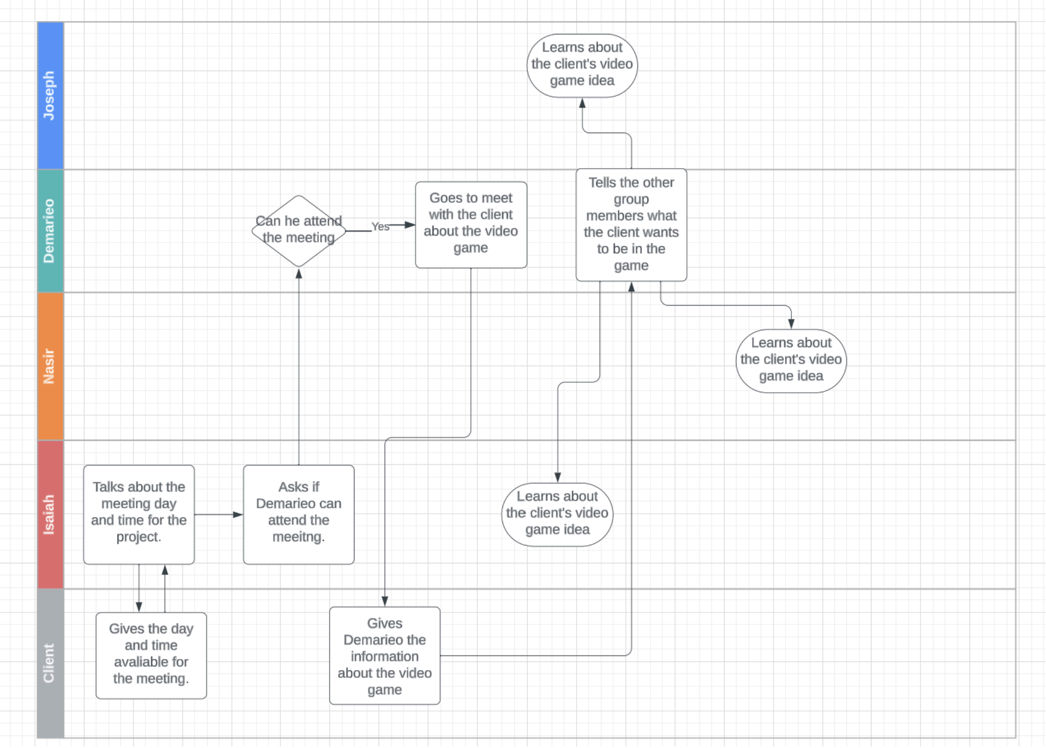
3. Requirement Workflows

3.1 Workflow for Communication (First action)

This workflow satisfies the following Requirements:

- REQ #1: shows interaction with client
- REQ #2: shows connection between all group members and client
- REQ #3: shows who had the meeting with the client

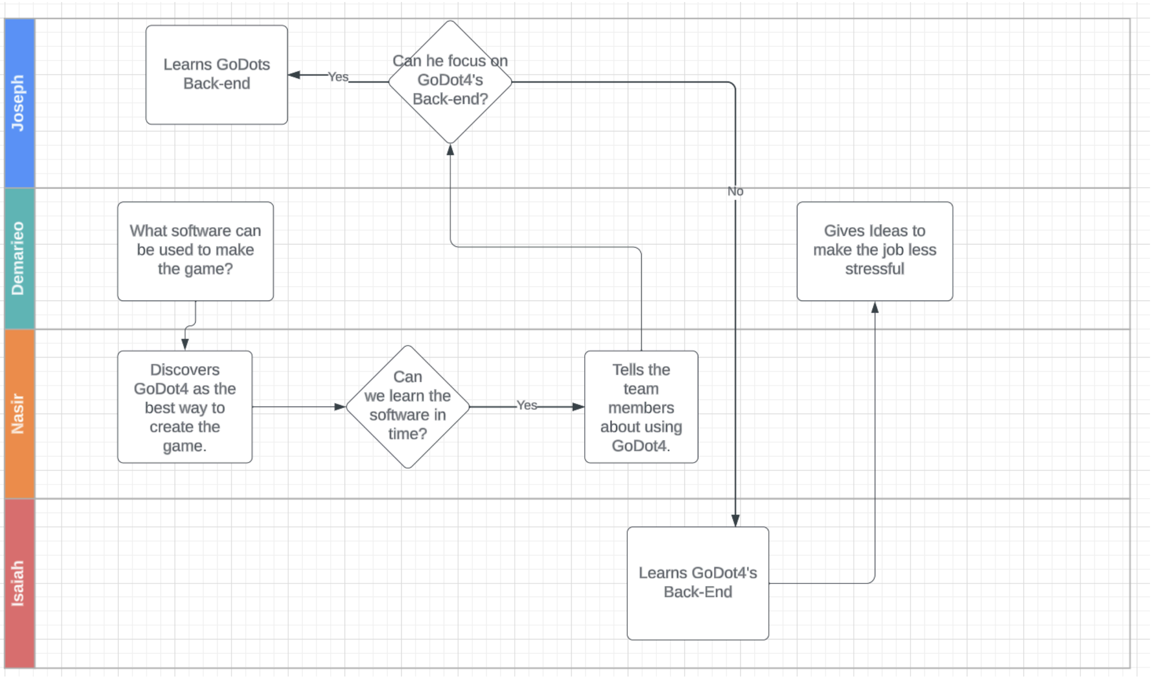
3.1.1


Joseph
Demarieo
Nasir
Isaiah
Client
Learns about the client's video game idea
Can he attend the meeting
Yes
Goes to meet with the client about the video game
Tells the other group members what the client wants to be in the game
Learns about the client's video game idea
Talks about the meeting day and time for the project.
Asks if Demarieo can attend the meeitng.
Learns about the client's video game idea
Gives the day and time avaliable for the meeting.
Gives Demarieo the information about the video game

3.2

This workflow satisfies the following Requirements:

- REQ #1: shows communication between all group members.


Joseph
Demarieo
Nasir
Isaiah
Learns GoDots Back-end
Yes
Can he focus on GoDot4's Back-end?
No
What software can be used to make the game?
Gives Ideas to make the job less stressful
Discovers GoDot4 as the best way to create the game.
Can we learn the software in time?
Yes
Tells the team members about using GoDot4.
Learns GoDot4's Back-End

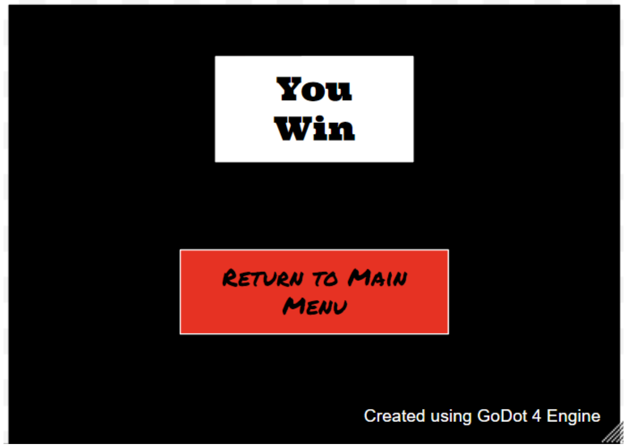
4. UI Design

4.1 Win Screen

4.1.1 UI Description and Related Requirements

- This UI is the screen that shows up when the user has beaten the boss.

4.1.2 UI Screenshot

- 

4.1.3 UI Detailed Description

-

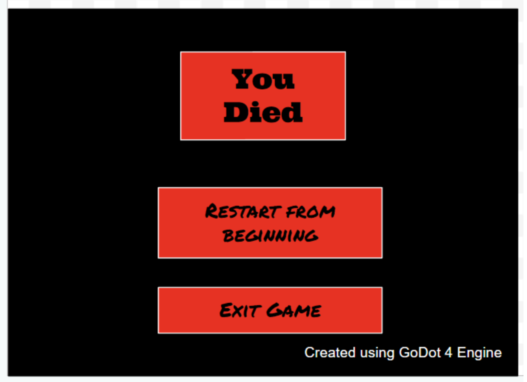
Field Name	Type	Purpose and/or Action(s) to Perform
You Win	Label	To let the user know that they have beaten the game
Return to Main Menu	Button	To return back to the main menu screen

4.2 Lose

4.2.1 UI Description and Related Requirements

- This UI is what shows up when the user has run out of health.
- REQ #1: Has Labels
- REQ #2: Has the necessary Buttons

4.2.2 UI Screenshot

- 

4.2.3 UI Detailed Description

-

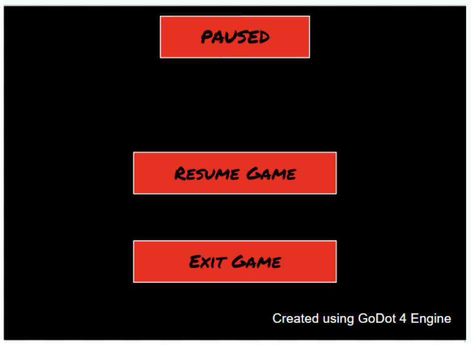
Field Name	Type	Purpose and/or Action(s) to Perform
You Died	Label	Tell the user that they have lost the game
Restart from beginning	Button	Start the game over from the beginning.
Exit Game	Button	Ends the Game

4.3 Paused

4.3.1 UI Description and Related Requirements

- This UI is what is seen when the user wants to pause the game
- REQ #1: Has Labels
- REQ #2: Has the necessary Buttons
- REQ #3: Stops the game

4.3.2 UI Screenshot

- 

4.3.3 UI Detailed Description

-

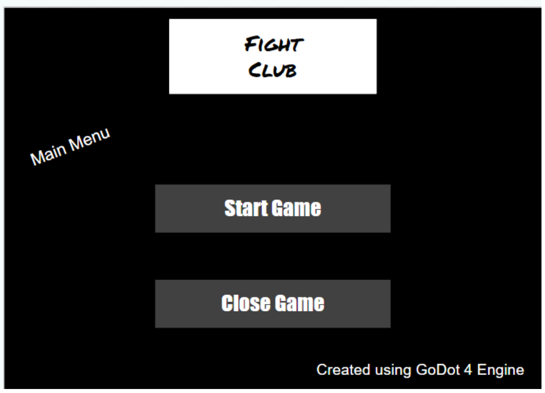
Field Name	Type	Purpose and/or Action(s) to Perform
Paused	Label	Says that the game is paused
Resume Game	Button	Starts the game from the current point
Exit Game	Button	Ends the game

4.4 Main Menu

4.4.1 UI Description and Related Requirements

- This UI is the first screen that is seen when the video game is opened.
- REQ #1: Has Labels
- REQ #2: Has the necessary Buttons

4.4.2 UI Screenshot

- 

4.4.3 UI Detailed Description

-

Field Name	Type	Purpose and/or Action(s) to Perform
Fight Club	Label	This is the title of the game
Start Game	Button	This button runs the game when pressed.
Close Game	Button	This button closes the game.
Main menu	Label	Tells the user this is the main menu

5. Database Design

5.1 Score

5.1.1 This table holds information about the player.

- REQ #1: Stores score

5.1.2 Table Detailed Description

Column Name	Type	PK\FK	DescriptionPurpose and/or Action(s) to Perform
Score	Char(20)		The score the player had at the end of the game

6. Process Design

6.1 Requirements

6.1.1 These are the requirements we needed in order to make a video game

- A software that could be used to create video games
- Have an idea about what we wanted our game to look like
- What features we wanted it to have

6.2 Design

6.2.1 For this process, we worked on getting an idea of how we wanted the game and the different screens to look like when playing the game.

- We will have examples of different UI screens drawn up, but it would change after

6.3 Development

6.3.1 We are in the process of following this process now. This allows us to show a little bit of what we are making.

- This is where we try to make the video game according to the design phase; however, changes might be made.
- More about the game will be given during the Demo.

6.4 Testing

6.4.1 Due to the way our project is, we test the project as it is being made.

- The test are conducted after a certain thing has happened
- Ex. A character is added to the game. We make sure the character is doing what it is supposed to do before we continue with the project.

7. Implementation

- For the demo of our game, we give the idea of what type of game that is being created.
- For the Demo, there is going to be 1-2 stages for the video game.
- We also have the controls that have him moving left and right. He also jumps with the space bar.
- Link to the Github: https://github.com/Kingdomheartz/Video-Game-Project

8. Unit Testing

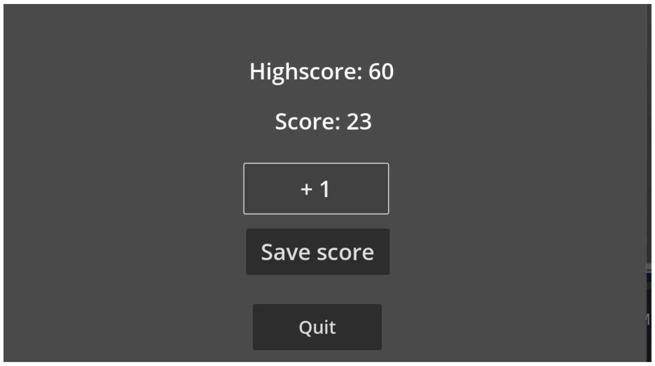
8.1 Test Case Name (Copy and repeat for each unit test case)

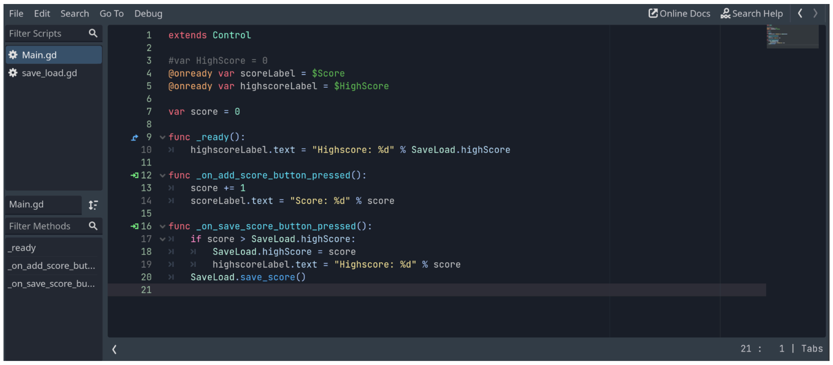
8.1.1 Test Case Description and Related Requirements

- A score counter and a highscore counter was created to see most points ever recorded during a playthrough.
- Requirements:

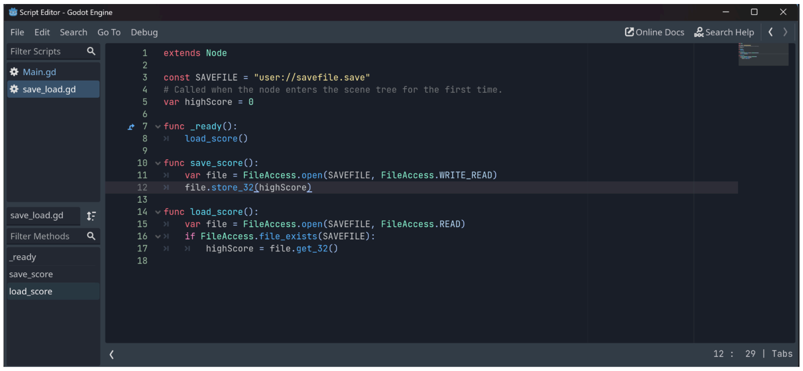
8.1.1.●.1 Godot 4: FileAccess knowledge

8.1.2 Test Case Steps





```
extends Control

#var HighScore = 0
@onready var scoreLabel = $Score
@onready var highscoreLabel = $HighScore

var score = 0

func _ready():
    highscoreLabel.text = "Highscore: %d" % SaveLoad.highScore

func _on_add_score_button_pressed():
    score += 1
    scoreLabel.text = "Score: %d" % score

func _on_save_score_button_pressed():
    if score > SaveLoad.highScore:
        SaveLoad.highScore = score
        highscoreLabel.text = "Highscore: %d" % score
    SaveLoad.save_score()

```



```
extends Node

const SAVEFILE = "user://savefile.save"
# Called when the node enters the scene tree for the first time.
var highScore = 0

func _ready():
    load_score()

func save_score():
    var file = FileAccess.open(SAVEFILE, FileAccess.WRITE_READ)
    file.store_32(highScore)

func load_score():
    var file = FileAccess.open(SAVEFILE, FileAccess.READ)
    if FileAccess.file_exists(SAVEFILE):
        highScore = file.get_32()

```

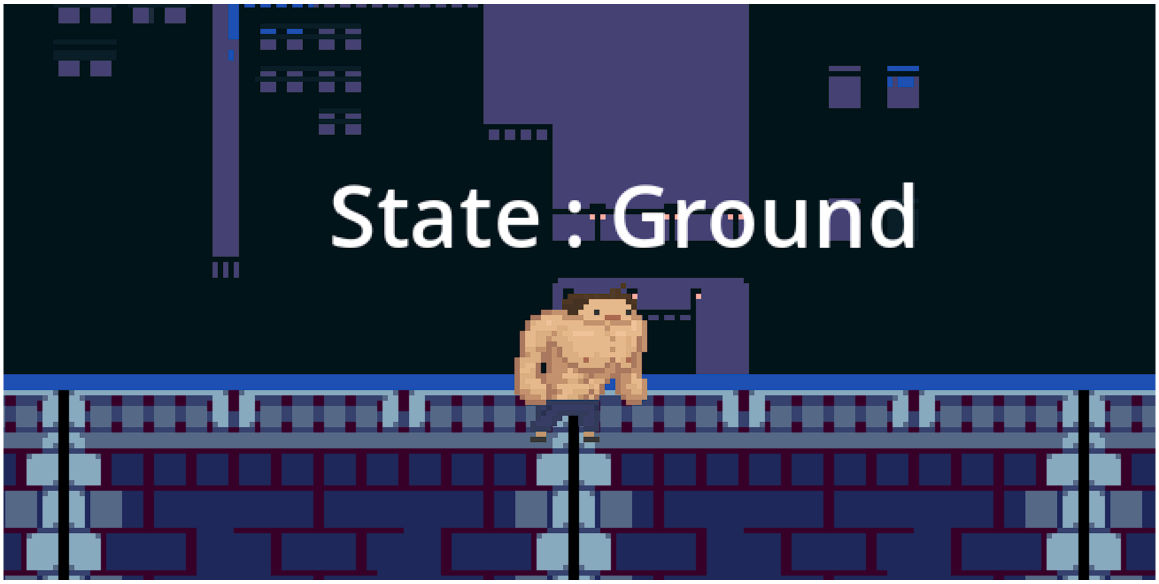
9. System Testing

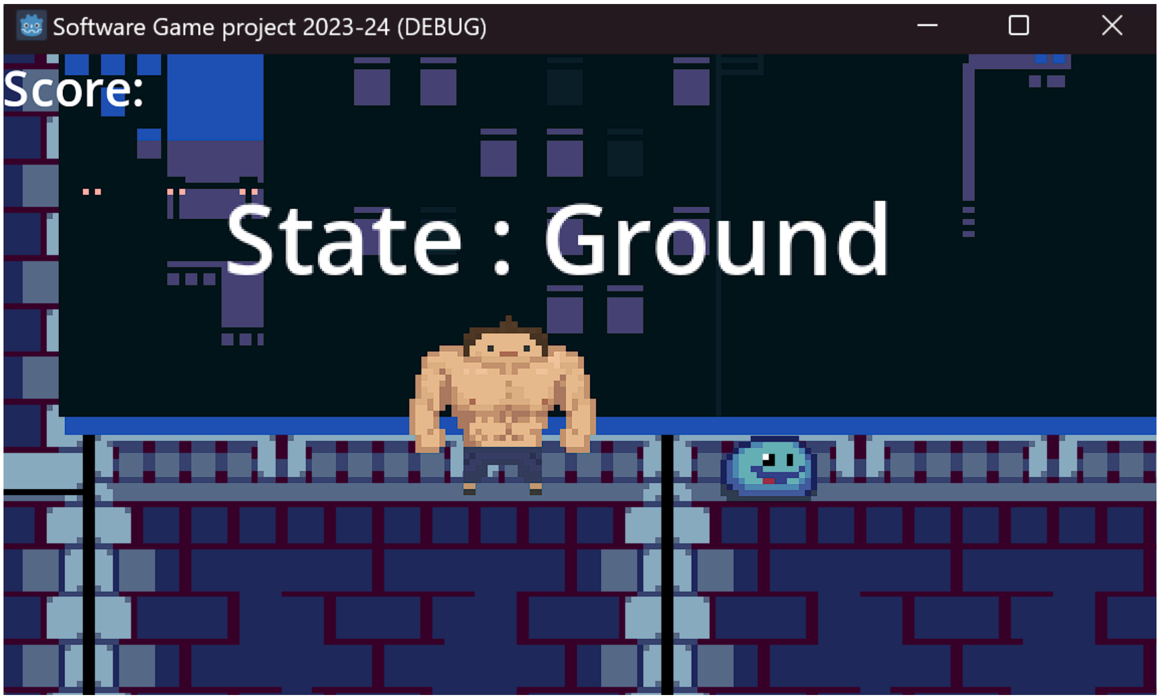
9.1 Test Case Name (Copy and repeat for each system test case)

9.1.1 Test Case Description and Related Requirements

- We tested the video game to ensure it was working properly

9.1.2 Test Case Steps


State : Ground


Software Game project 2023-24 (DEBUG)
Score:
State : Ground

10. Appendices

10.1 Document Version History

Date	**Version**	**Last Modified**	**Modified By**	**Description**
1/22/24	0.3	The character's movements	Nasir Hill	Updated the characters actions
2/2/24	0.4	The UI Screens	Demarieo James	Added the Lose Menu
2/15/24	0.5	UI screens	Joseph Johnson	Added the pause menu
3/10/24	0.7	Highscore	Isaiah Dorsey	Added the score and high score feature
4/12/24	1.0	Enemy Movements	Nasir Hill	Added the enemy actions

10.2 Supplementary Information (as required)