Tribhuvan University

Academia International College

Final Year Project Report
On
RPG game: Path of wanderer
[CSC 412]

Under the supervision of
"Er. Anup Shrestha"

Submitted by

Kritan Shrestha (T.U. Symbol No. 29012/078)
Smriti Tamang (T.U. Symbol No. 29031/078)

Submitted to
Department of Computer Science and Information Technology
Academia International College
Institute of Science and Technology
Tribhuvan University
September 2025

Tribhuvan University

Academia International College

Final Year Project Report

On

"RPG game: Path of wanderer"

[CSC 412]

A final year project submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Information Technology awarded by Tribhuvan University

Submitted by

Kritan Shrestha (T.U. Symbol No. 29012/078)

Smriti Tamang (T.U. Symbol No. 29031/078)

Submitted to

Department of Computer Science and Information Technology

Academia International College

Institute of Science and Technology

Tribhuvan University

September 2025

Tribhuvan University

Institute of Science and Technology

Academia International College

Department of Computer Science and Information Technology

Email: mail@academiacollege.edu.np

Supervisor's Recommendation

I hereby recommend that the project work report prepared under my supervision Mr. Kritan Shrestha (T.U. Symbol No. 29012/078), Ms. Smriti Tamang (T.U. Symbol No. 29031/078) entitled "RPG game: Path of wanderer" be accepted as fulfilling in partial requirements for the degree of Bachelor of Science in Computer Science and Information Technology. In my best knowledge, this is an original work in Computer Science and Information Technology.

.....

Er. Anup Shrestha

Project Supervisor

Department of Computer Science and Information Technology

Academia International College,

Gwarko, Lalitpur

Tribhuvan University

Department of Computer Science and Information Technology

Academia International College

Certificate of Approval

This is to certify that this project prepared by Ms. Smriti Tamang, and Mr. Kritan Shrestha entitled "RPG game" in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Information Technology has been well studied. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

…….. Er. Anup Shrestha Project Supervisor Department of Computer Science and IT Academia International College	…….. Mr. Bishwas Mathema HOD/Program Coordinator Department of Computer Science and IT Academia International College
…….. Internal Examiner Academia International College	…….. External Examiner Central Department of CSIT Tribhuvan University

Acknowledgement

We owe our most profound appreciation to Academia International College for giving us a chance to work on this project as part of our syllabus.

Special thanks to our supervisor, Er. Anup Shrestha (Academia International College), for his consistent guidance, support, and feedback throughout the report's creation. We are generously obligated to him for providing this excellent opportunity to expand our knowledge. It helped us a lot to realize what we studied for.

We would like to express our sincere gratitude to all those individuals, families, friends, colleagues, and teachers for supporting and helping us a lot in finalizing this project within the limited time frame by providing valuable insights and feedback on the report.

Thanking You,

Kritan Shrestha (T.U. Symbol No. 29012/078)

Smriti Tamang (T.U. Symbol No. 29031/078)

Abstract

The "Path of Wanderer" is a single-player RPG game application that serves to provide an immersive and engaging fantasy gaming experience to its users. It is a story-driven role-playing game that towards provides narrative depth, character progression, and strategic combat mechanics without the burden of pay-to-win models or excessive monetization to the players. This game features a comprehensive fantasy world where players can explore diverse environments, complete meaningful quests, engage in tactical combat with various monsters, and ultimately face a challenging boss battle and various educational mini games such as wordle and jigsaw puzzle. The game includes essential RPG elements such as character progression, inventory management, quest tracking, and NPC interactions, all wrapped in an intuitive user interface designed for seamless navigation. The game will be developed using Unity engine with C# as the programming language. Visual assets will be created using industry-standard tools, and the project will utilize version control systems for collaborative development. The game is designed to run efficiently on PC platforms, ensuring accessibility for players with varying hardware specifications. Path of Wanderer represents a return to traditional RPG values, emphasizing storytelling, exploration, and player agency over competitive multiplayer mechanics. The project demonstrates the potential of Unity for creating engaging single-player experiences that prioritize player enjoyment and narrative immersion over profit-driven design decisions.

Keywords: Single-player RPG, narrative-driven, Unity Engine, C#, fantasy setting, character progression, NPC interaction, inventory system, quest system, exploration, story-rich, intuitive UI, PC gaming

Table of Contents

List of figures

List of tables

List of Abbreviation

Git	Version Control System (Git)
HUD	Heads Up Display
LTS	Long-Term Support
NPC	Non-Player Character
OOP	Object-Oriented Programming
PC	Personal Computer
RPG	Role-Playing Game
XP	Experience Points
UI	User Interface
UX	User Experience

Chapter 1: Introduction

1.1. Introduction

One of the first ever released adventure games was Colossal Cave Adventure in 1976. Eventually, as the technologies got better, the visual adventure games started to be born. In 1980s, a game named Mystery House from Sierra On-line was created which was considered one of the first visual adventure game. However, it still required input texts. After the passing of time, new generation game started to develop 3D gameplay. In this modern era, where everything is surrounded by technologies. There are tons of adventure games available from all around the world and on the play store. However, according to my research, most games don't provide educational purposes to their users or players. Most of the games available on the market right now is mostly based on providing entertainment to their users which is one of the biggest weaknesses on the market. For that specific purpose, this RPG (Role Playing Game) is developed to fill up that gap and provide educational purposes while entertaining the users and the players playing the game.

This 3D adventure game is a single player educational fantasy game. In this game, the player controls the actual character to freely walk and climb around the platforms. The game has several quests completely different from the first one to make it interesting for the player and to keep them engaged in playing the game. As the game progresses, the players can experience various levels in the game and objectives, difficulty varying according to the quests of the game. In this game, players can explore a fantasy world, complete educational quests, fight monsters, and defeat them to win. The game will include simple RPG mechanics, such as quest tracking, combat systems, and player progression. The game is designed for casual players, specifically designed for a small group of friends and teachers and will be available on PC only.

The main gameplay and mechanics of this game were developed using Unity and the C# programming language. The game's artwork consists of 2D or 3D models in a fantasy setting with old time theme. The goal of the project is to develop an engaging simple RPG experience that showcases the potential of Unity for creating accessible and fun games.

1.2. Problem Statement

These days, the gaming industry has been focusing on competitive, monetized, and fast-paced games. As a result, story driven RPGs that emphasize exploration, immersion, and creativity have become less popular. Most modern games focus more on multiplayer mechanics and making money over core gameplay. The goal of this project is to bring back the spirit of classic RPGs through a single player experience using Unity, focusing on design quality and core gameplay over profits.

Some of the key problems are:

- **Over Competitiveness**
  Competitive games involve players competing against others online, either directly or strategically. Casual players may feel discouraged to play games. Over competitiveness also creates a "toxic" community or environment that drains the joy out of gaming.

- **Pay-to-Win and Monetization Models**
  These days, in-game purchases and paid upgrades are common in games. This project removes all forms of monetization, offering a complete experience without microtransactions or locked features.

- **Game Engines with a Graphics Hunger**
  Many modern games demand high-end hardware, inaccessible to users with less powerful PCs. To create immersive graphics while preserving wide system compatibility, this project maximizes visual.

1.3. Objectives

The primary objective of this project is to create a simple yet engaging RPG game. The objectives we hope to achieve within this project are as follows:

- To develop a basic RPG game in Unity with core RPG features: character movement, combat systems, quests, and enemy AI.
- To create a program that can read user input provided by players and respond appropriately.
- To build a simple but detailed environment that will be both educational and entertaining.

1.4. Scope and Limitations

1.4.1. Scopes

The project entails using C# and Unity to create a window-based gaming software application. The main categories of features and functionalities that are necessary for the creation of this game are listed below:

- Player character movement and animation
- Basic combat system and ranged attacks
- Inventory system with equipment slots
- NPC interaction system
- Quest tracking system
- Save/load functionality
- Special sound effect. For example: character interaction, dying, etc.
- Actual world gravitational physics

All the above-mentioned components will be combined to form a complete game.

1.4.2. Limitations

The limitations include:

- Single-player gameplay only
- Fixed camera perspective
- Restricted world size
- Simplified combat and mechanics
- No limited character customization

1.5. Methodology

The Agile methodology has been used during the development of this project, as it focuses more on iterative and flexible process of development. Here development phase is divided into smaller pieces of steps known as sprints. Each sprint was carried out to focus on quick development of working, manageable ad functioning part of game.

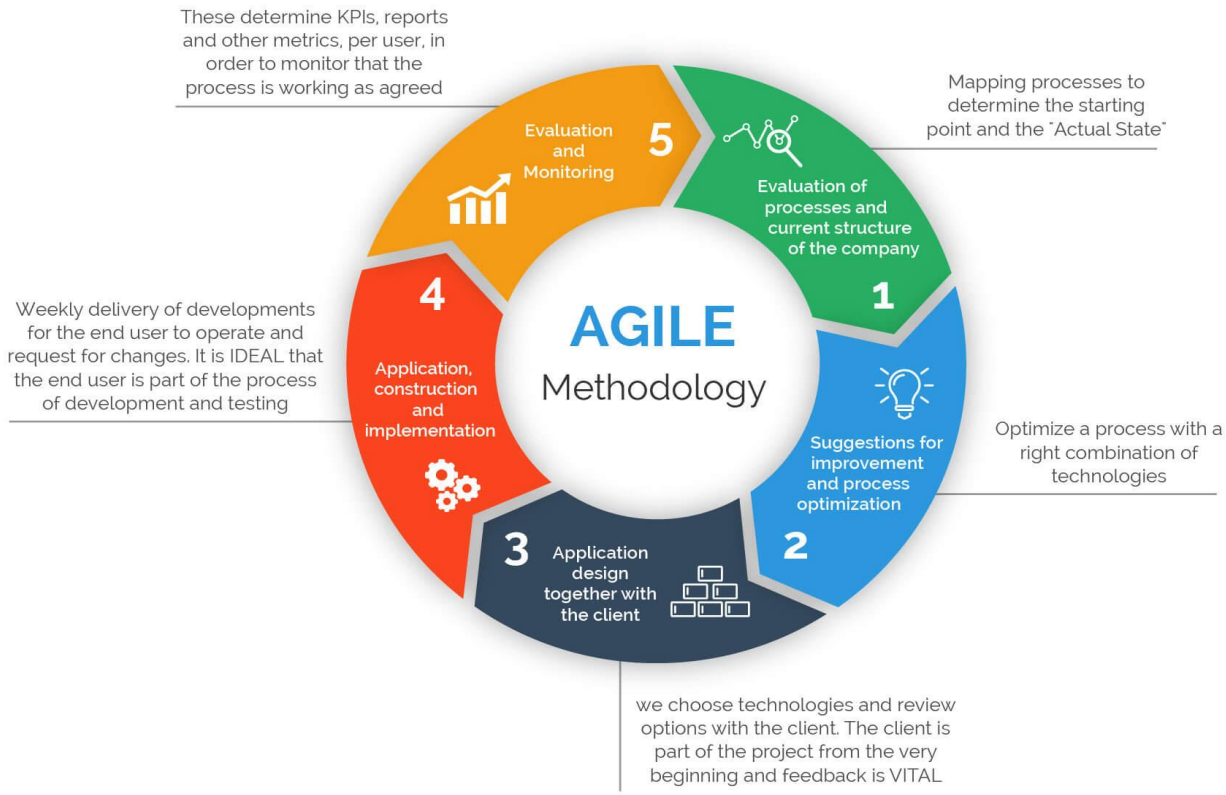


Figure 1.1: Agile Methodology of software Development

1. Evaluation of Processes and Current Structure

We began by analyzing existing systems and user expectations in similar applications. This included:

- Studying well-known games and apps for reference (e.g., how Genshin handles progression, or how The Witcher structures quests).
- Identifying the limitations casual users face (e.g., complex lore, monetization).
- Mapping out our starting point with minimal mechanics and systems.

2. Suggestions for Improvement and Process Optimization

From that evaluation, we proposed improvements:

- Making the interface beginner friendly.
- Removing pay-to-win or grind-heavy systems.
- Designing a lightweight quest and combat system tailored to casual and educational use.

3. Application Design

We designed the application hand-in-hand with feedback from users (like testers or mentors):

- Quests and features (e.g., educational challenges, moral choices) were brainstormed collaboratively.
- Sequence diagrams were drafted to clarify flows (combat, learning progression).
- Design iterations reflected real-time feedback to ensure the app felt intuitive.

4. Application, Construction, and Implementation

The development phase involved:

- Implementing core gameplay mechanics.
- Developing and testing combat logic (as shown in your sequence diagram).
- Adding educational quest systems.
- Weekly builds or module testing to gather insights from users and incorporate their feedback.

5. Evaluation and Monitoring

After each build:

- We evaluated player interaction and ease of use.
- Metrics such as task completion, learning engagement, or play session length were reviewed.
- Based on results, we re-adjusted quest flow, combat balance, or UI responsiveness.

1.6 Report Organization

Project Organization refers to systematic management and coordination of all tasks, activities, and resources required to complete the project successfully. A clear and logical organization is required to ensure that every phase of the project is completed efficiently, on schedule, and according to the outlined objectives. The report is organized into six chapters, each with its own content.

1. **Chapter 1: Introduction**

   Introduces the project, discusses the background, defines the scope and objectives, and outlines the problem that the project aims to address.

2. **Chapter 2: Literature Review**

   Summarizes existing research and related work, explores similar systems and technologies.

3. **Chapter 3: Methodology and System Design**

   Explains the tools, techniques, and design approaches used, including software requirements, system architecture, and conceptual models such as use case and sequence diagrams.

4. **Chapter 4: Analysis and Design**

   Provides detailed analysis and design specifications. This includes data flow diagrams, entity-relationship diagrams, and interface designs, illustrating the structural and functional aspects of the system.

5. **Chapter 5: Implementation and Testing**

   Describes the implementation process, tools and platforms used, module by module functionality, and testing strategies. It also discusses unit testing, integration testing, and system testing with corresponding test cases and results.

6. **Chapter 6: Conclusion and Future Recommendations**

   Summarizes the entire project's accomplishments, discusses its drawbacks, and suggests potential improvements or features that could enhance the project in future iterations.

Chapter 2: Background Study and Literature Review

2.1 Background Study

Role-Playing Games (RPGs) have become one of the most influential genres in the gaming industry, offering players interactive storytelling, exploration, and character development. In particular, fantasy adventure RPGs immerse players in open-world environments where quests, collectables, and combat systems drive engagement. Modern 3D RPGs rely on a combination of game design principles, artificial intelligence, physics simulations, and real-time graphics to deliver an immersive experience. [1]

For this project, we studied the core concepts of RPG systems. These include player movement in a 3D world, combat and quest progression, inventory and resource management, and non-player character (NPC) interactions. We explored physics and collision handling, including Rigidbody mechanics, collision detection, and triggers, which ensure realistic movement and interactions within the game world. [2] Additionally, we applied scripting and programming concepts such as event handling, state machines, and object-oriented design to structure game systems efficiently. [4]The UI and HUD design was also an important focus, enabling us to create functional inventory systems, health bars, quest logs, and feedback mechanisms for the player. [5]

Alongside theoretical knowledge, we learned and applied several tools and technologies. The Unity Game Engine served as the foundation for world-building and mechanics implementation, while C# scripting allowed us to design player controls, quest logic, and inventory systems. We utilized Blender to create and customize 3D assets, implemented Unity's NavMesh for AI navigation, and applied GitHub for version control to manage collaboration and iterative development effectively. [6]

During development, we encountered several challenges, such as integrating multiple systems into a seamless experience, ensuring smooth player-NPC interactions, and managing the performance of physics simulations and AI pathfinding in the open world. By studying these concepts and tools in depth, we were able to overcome these difficulties and construct a functional prototype of a 3D fantasy RPG game. [7]This background knowledge provided both the theoretical foundation and practical skills necessary to achieve the project's goals and deliver an engaging gaming experience.

2.2 Literature Review

The Role-Playing gaming (RPG) genre has been the foundation of the video gaming industry for a long time, giving players deep stories, character growth, and strategic gameplay. RPGs range from complex open-world experiences to minimalist indie titles, adapting to evolving player preferences and hardware capabilities. This literature review explores key RPG titles across various sub-genres, highlighting core mechanics, user experience design, and their relevance to our project's scope and objectives.[4]

The Witcher 3: Wild Hunt

This game is renowned for its narrative depth and morally complex choices. Its character progression system and branching dialogue trees contribute to high replay value. However, the game's length and dense storytelling may not cater to users seeking short, casual gameplay sessions. Its relevance lies in demonstrating how quality storytelling can elevate user engagement an area we aim to simplify for our casual target audience.

Dota 2

Dota 2 is a highly competitive multiplayer online battle arena (MOBA) game developed by Valve Corporation. It emphasizes strategic team-based combat, with players controlling unique heroes that have distinct abilities and roles. The game features complex mechanics, including item management, map control, and real-time decision-making. Dota 2 has a large, active community and supports frequent tournaments, contributing to its global popularity. However, its steep learning curve and high-pressure gameplay can be intimidating for casual players. Our RPG addresses this by offering simpler mechanics, gradual learning, and a more accessible, single-player experience while still keeping gameplay engaging and strategic.

Genshin Impact

As a recent action-RPG, Genshin Impact showcases high-quality visuals, real-time combat, and a gacha monetization model. While it excels in polish and world-building, its heavy monetization and grind-centric progression systems present barriers to entry for casual players. These drawbacks further support our decision to exclude monetization from our own project, ensuring accessibility and fair gameplay.

R.E.P.O.

R.E.P.O. (Retrieve, Extract and Profit Operation), is a cooperative survival horror game released in early access in February 2025. It allows up to six players to explore procedurally generated facilities, collect valuable items, and transport them to extraction points under time pressure and escalating threats Players use in-game earnings to purchase permanent upgrades. However, its steep difficulty and punishing mechanics can overwhelm casual players. Our RPG avoids such barriers by keeping challenges simple and accessible.

PEAK

PEAK is a cooperative climbing game where players work together to survive dangerous mountain climbs. It is known for its creative gameplay and social fun. However, it focuses more on mechanics than story or learning. Our RPG balances both gameplay and educational content for deeper engagement.

Chapter 3: System Analysis

3.1. System Analysis

3.1.1. Requirement Analysis

i. **Functional Analysis:**

- Character navigation and movement
- Interaction with NPCs and objects in the game world
- Inventory system
- Quest system for tracking and managing tasks
- Basic combat mechanics (attacking monsters and boss)
- UI for health, inventory, and settings

ii. **Non-functional:**

- Optimized performance for smooth gameplay on mid-range PCs
- Modular and maintainable codebase using OOP and Unity best practices
- Accessibility through readable fonts, intuitive UI, and basic color contrast

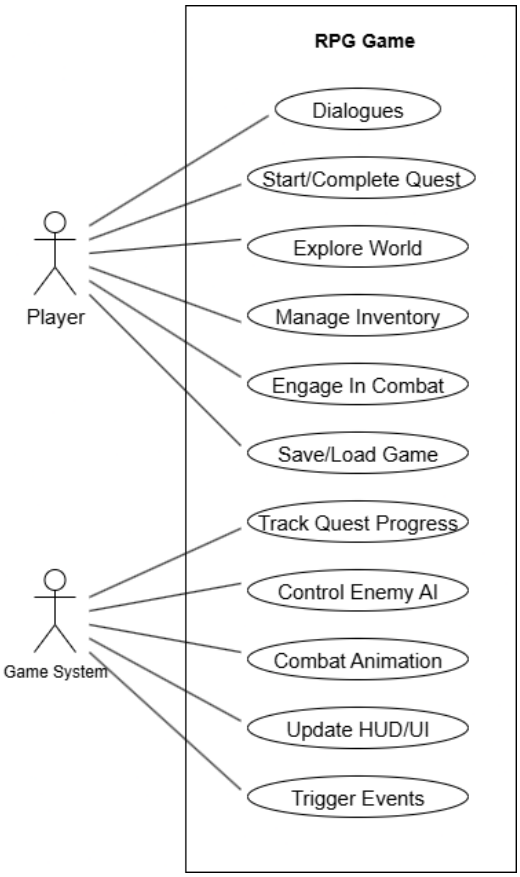


Figure 3.1: Use Case Diagram

3.1.2. Feasibility Study

1. **Technical:**
   - Game development using Unity (LTS version) with C# for scripting
   - Visual Studio as the development environment
   - PC-only build targeting low to mid-range hardware
2. **Operational:**
   - Use of Unity version control to track changes and maintain project history
   - Play tested internally by friends and teachers
   - Play tested internally by friends and teachers
3. **Economic:**
   - No licensing cost or third-party paid assets involved
   - No monetization planned; game is for academic learning only

Table 3.1: Gantt Chart

Activity	Start Week	Duration Weeks
Project Initiation & Planning	1	2
System Design	3	3
Environment Setup & Prototyping	6	3
Core Mechanics Development	9	4
Integration	13	2
Testing & QA	15	2
Documentation	1	16

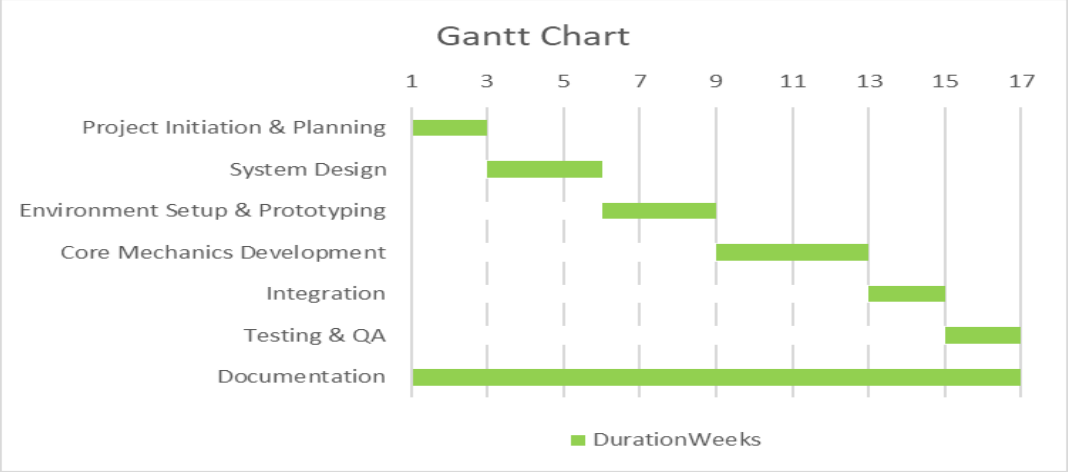


Figure3.2: Gantt chart

3.1.3 Analysis (Object Oriented)

This project of developing a mobile game follows a object-oriented approach as the programming language used (C#) is an object oriented programming language.

i. Object modelling using Class and Object Diagrams

The Class and Object diagram for our project is as follows:

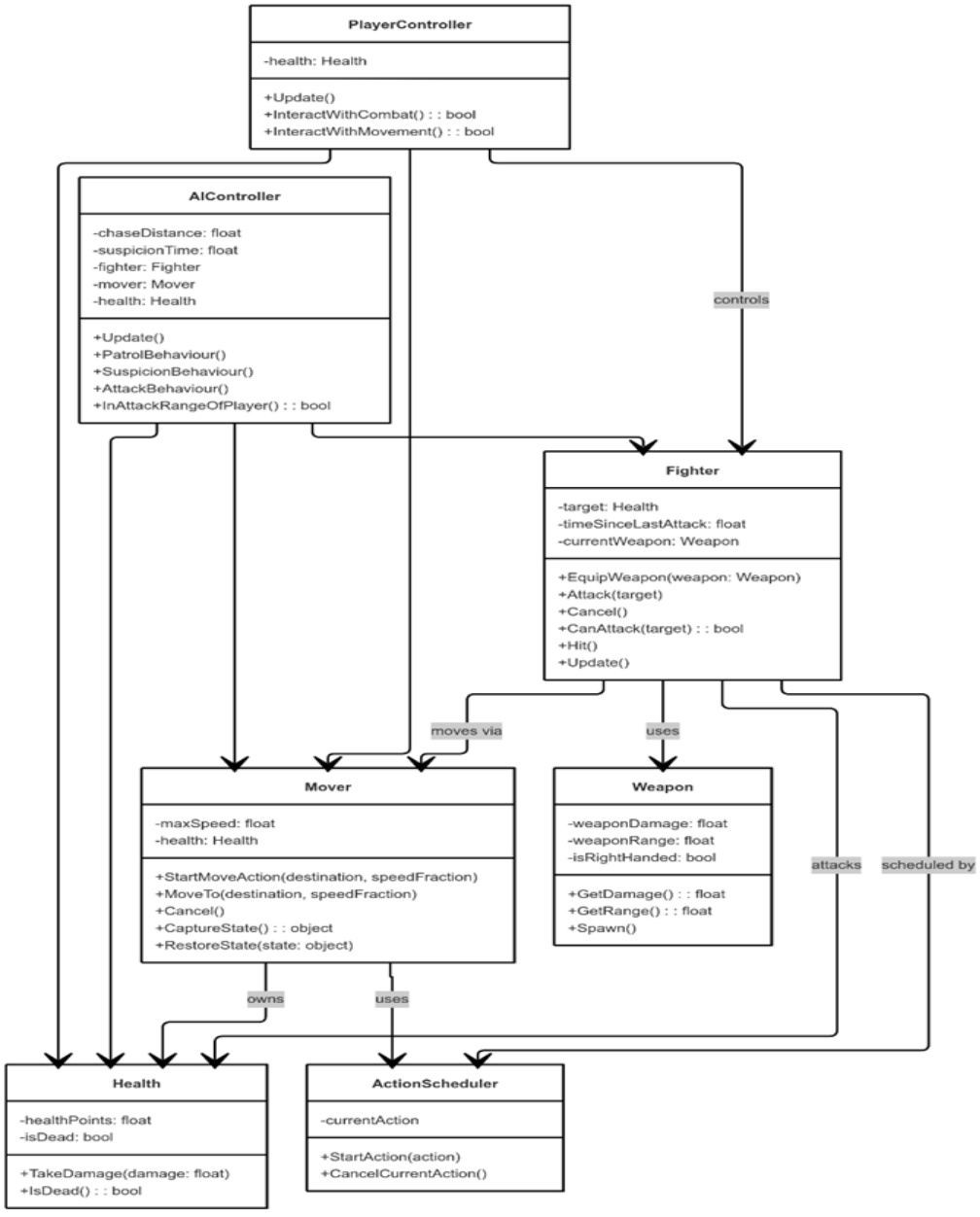


Figure 3.3: Class and Object Diagram

ii. Dynamic modelling using Sequence Diagrams

The sequence diagram for our project is as follows:

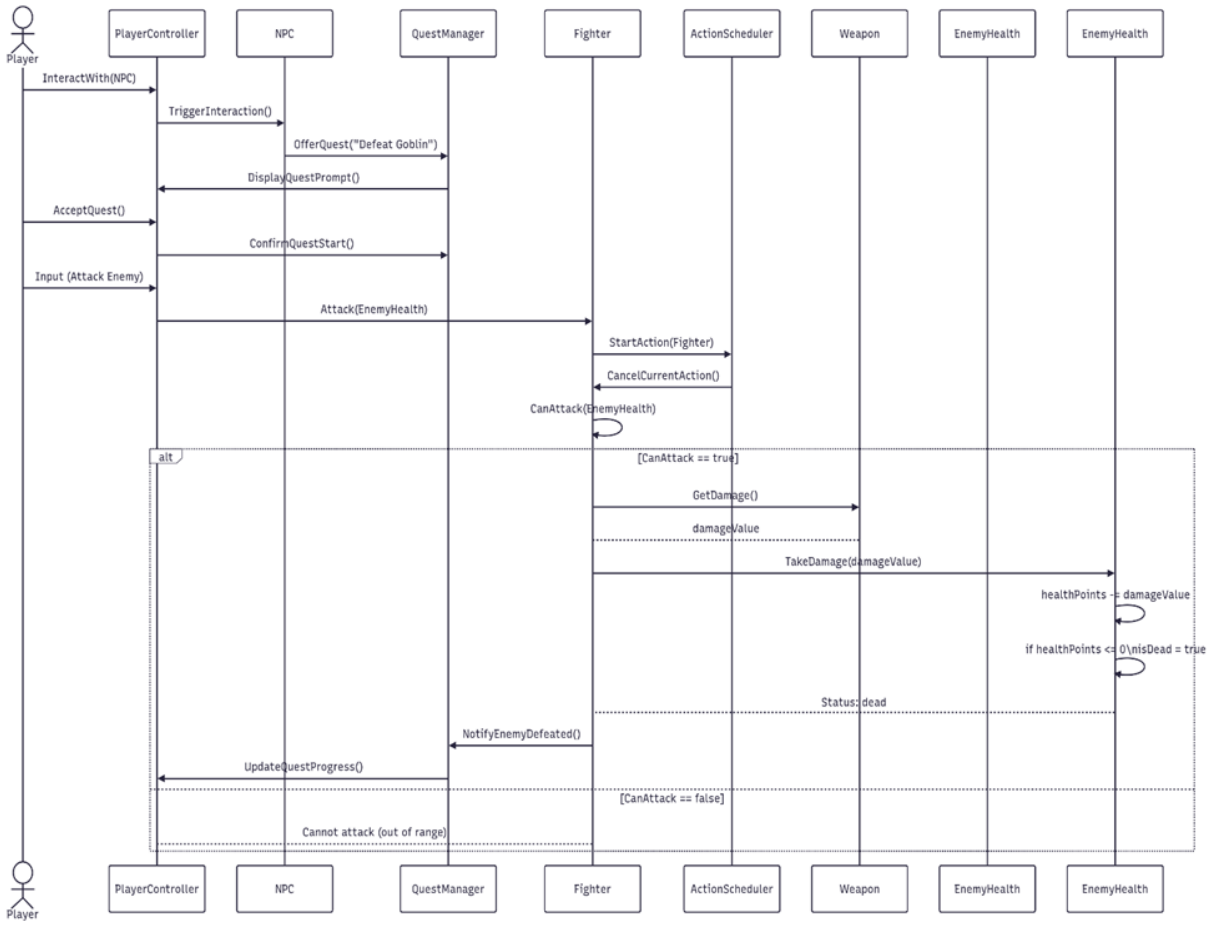


Figure 3.4: Sequence Diagram

iii. Process modelling using Activity Diagrams

Activity diagrams are simply an advanced from of flow-chart diagram that model the working of the system from one activity to another. An activity diagram for our project is as follows:

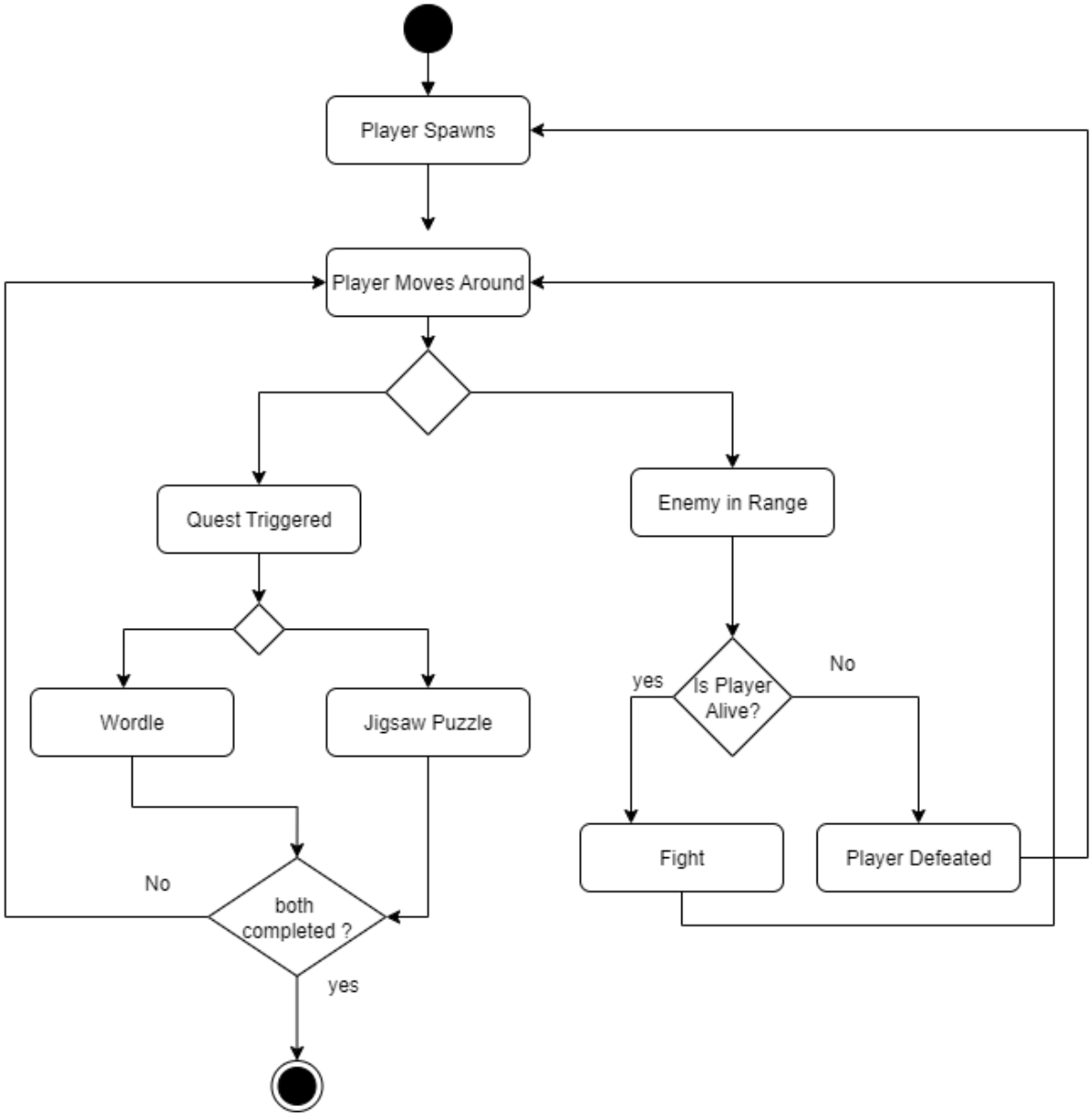


Figure 3.5: Activity Diagram

Chapter 4: System Design

4.1 Design (Object Oriented)

As discussed in the analysis chapter, the system design also follows an object-oriented approach.

- Refinement of Class, Object and Activity Diagram

  Since this is a very basic project with not much complexity, the class diagrams, object diagrams and activity diagrams designed earlier are already refined enough to be applicable within the project.

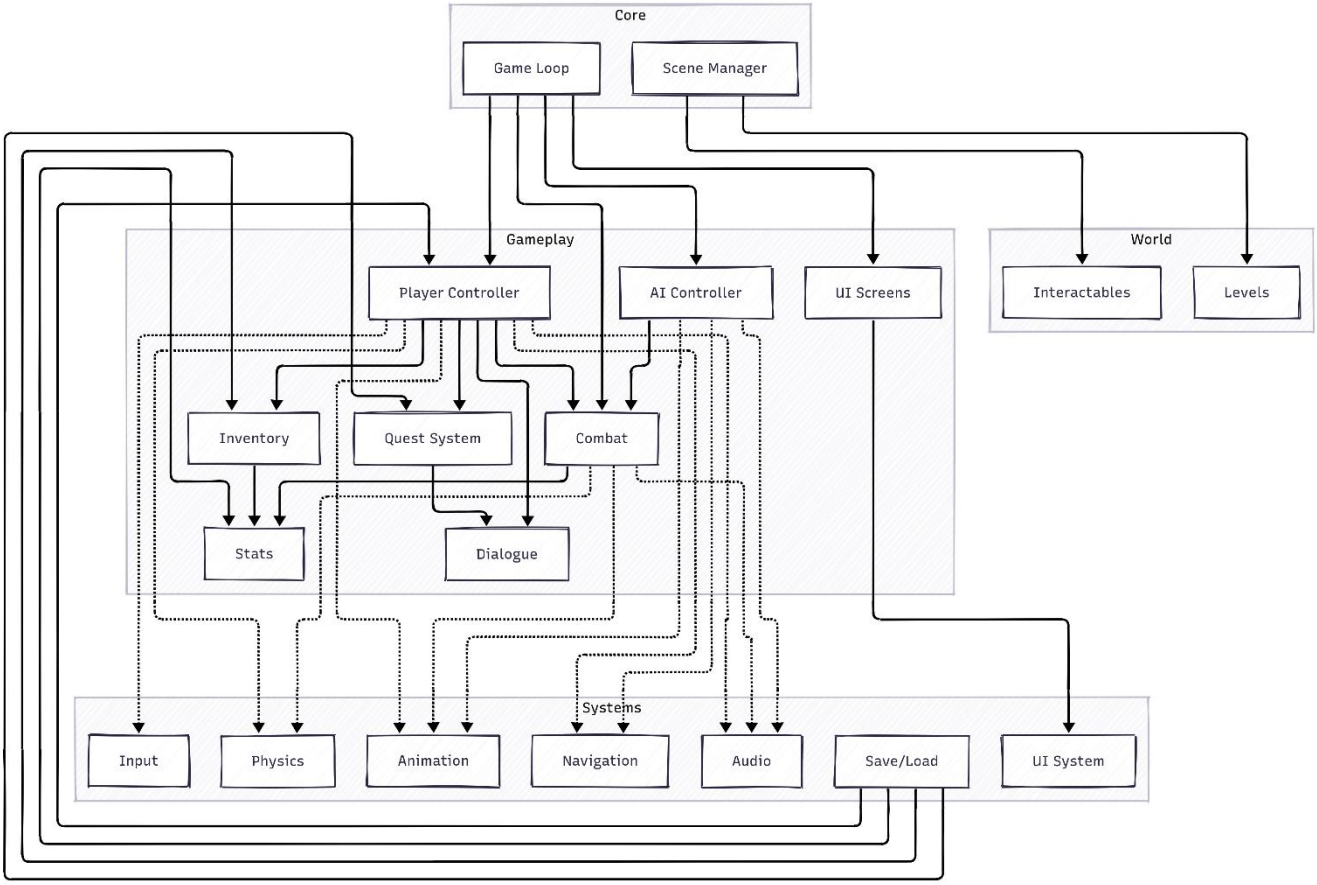


Figure 4.1: Component Diagram

4.2. Algorithm Details

4.2.1. Pathfinding (A* Algorithm)

Pathfinding algorithm is a step-by-step procedure which is used to find optimal path between starting point and target point. Here in this game A* Algorithm is used as it combines elements of Dijkstra's algorithm and Greedy Best-First Search for efficiency and optimality. The main purpose of it in our game is to enable NPCs and enemies to navigate the game world.

Example:

Enemies use A* to chase the player through a complex map while avoiding walls or traps.

4.2.2. Physics Simulation Algorithm

A physics simulation algorithm is a step-by-step computational method that imitates how objects move and interact according to the laws of physics. It is the backbone of our game. Each object has properties that can be mass, position, speed and velocity accordingly. It also implements gravity, friction and collision. The main purpose of this algorithm is to make realistic movement and interaction of body.

Example:

A rigid body is used so that it does not deform under forces or collisions.

4.2.3 Graphics Rendering Algorithm

Graphics rendering algorithms are methods that are used to generate images from 2D or 3D models in a game or simulation. The main purpose of this algorithm is to make models and scene realistic in game.

Example:

Material, shader, texture etc. in game.

4.2.4 Artificial Intelligence (AI) Algorithm

Artificial Intelligence (AI) algorithms are methods that are used to adapt, behave and make decision for non-Player Characters (NPCs). The main purpose of this algorithm is to make object react to player.

Example:

Applying animation, nav mesh path and path finding algorithm also falls under this category.

4.2.5. Combat Mechanics Algorithm

Resolve Combat mechanics in the system resolve interactions between the player and enemies by calculating damage during attacks. The algorithm retrieves the attacker’s damage stat and the defender’s defense stat, then adjusts the final damage based on these values. Weapon effects and projectiles are also considered, ensuring that each attack reflects both character stats and equipment.

Example:

Damage is calculated by dividing the attacker's damage by the defender's defense factor:

```
float damage = GetComponent<BaseStats>().GetStat(Stat.Damage);
float defence = target.GetComponent<BaseStats>().GetStat(Stat.Defence);
damage /= 1 + defence / damage.
```

4.2.6. Collision Detection Algorithm

Collision detection algorithms are methods that are used to detect when two or more objects in a game or simulation intersect, touch, or come into contact. The main purpose of this algorithm is to detect and respond to interactions between objects, ensuring that characters, projectiles, and environmental elements behave realistically.

Example:

Collision response stops the player from falling through the platform and allows the player to stand on it.

Chapter 5: Implementation and Testing

5.1. Implementation

This report outlines how key algorithms and architectural patterns are implemented practically in the RPG game, including quests, combat, NPC dialogues, and event-driven interactions. The focus is on integrating these algorithms into Unity or a similar game engine.

5.1.1. Tools Used

The tools we used for implementing this project are as follows:

a. **Unity**

Unity is a cross-platform game engine developed by Unity Technologies, that supports a variety of desktop, mobile as well as console platforms. It gives users the ability to create games and experiences in both 2D and 3D, and the engine offers a primary scripting API in C# using Mono, for both the Unity editor in the form of plugins, and games themselves, as well as drag and drop functionality.

b. **C#**

C# is a general-purpose high-level programming language based on object-oriented programming created by Microsoft. It is a popular and easy to learn as well as simple to use programming language having huge community support.

c. Adobe Photoshop

Adobe Photoshop is a vector graphics editor and design software developed and marketed by Adobe that allows for the creation of everything from single design elements to entire compositions.

d. **Git**

Git is one of the most widely used platforms for hosting code. It allows for version control, to track the changes made to the project and the state of progress of the project. It also allows to collaborate on projects from anywhere and monitor the activities of the team members on the project.

5.1.2. Implementation Details of Modules

i. Game Core Module

This is the heart of the game, responsible for all core mechanics and systems that make the world come to life.

- **Gameplay Mechanics**:
  Handles player movement, jumping, running, and special abilities.
- **Combat and AI**:
  Manages enemy behavior, pathfinding, combat interactions, hit detection, and damage calculation.
- **Physics & Collision**:
  Controls gravity, character-environment interaction, and collision with terrain and props.

ii. UI Module

Responsible for every visual interface the player interacts with during gameplay.

- **Main Menu:**
  Entry point to the game, allowing new games, load games and quit game.
- **Inventory & Equipment Screens**:
  Displays player's items, gear, and stats.
- **Dialogue & Notification Popups**:
  Handles on-screen conversations.

iii. Audio Module

Manages all sound-related features to enhance immersion.

- **Background Music and Sound Effects**:
  Plays ambient themes for villages, forests, dungeons, and boss battles along with sound effects such as footsteps, sword swings, spell casting and environment effects.

iv. Asset Module

Controls efficient loading, management, and organization of all game assets.

- **3D Models & Textures**:
  Manages character models, NPCs, creatures, weapons, and world scenery.
- **Animations**:
  Controls animation clips for movement, combat combos, emotes, and environment elements.
- **Data Storage & Optimization**:
  Handles asset memory usage to ensure smooth gameplay performance.

5.2 Testing

5.2.1 Unit Testing

Unit testing checks individual components separately to ensure they behave as expected.

Test Case 1: Player Movement Function

Objective: Verify that the player character responds correctly to movement commands.

Table 5.1: Player Movement Test

Test Input	Expected Output	Actual Output	Status
Right Click Mouse	Character moves to that spot	Character moves to that spot	Pass
WASD keyboard	Character moves to that spot	Character moves to that spot	Pass

Test Case 2: Obstacle Collision Function

Objective: Verify player detects an obstacle and collides with it.

Table 5.2: Obstacle Collision Test

Test Input	Expected Output	Actual Output	Status
Player hits obstacle	Player collides with an obstacle.	Player collides with an obstacle.	Pass

Test Case 3: Inventory Add/Remove Function

Objective: Ensure items can be correctly added and removed from the player's inventory.

Table 5.3: Inventory Function Test

Test Input	Expected Output	Actual Output	Status
Add item to inventory	Item appears in inventory list	Item appears in inventory list	Pass
Remove item from inventory	Item is no longer in inventory list	Item is no longer in inventory list	Pass
Equip item from Inventory	Item is equipped in the inventory	Item is equipped in the inventory	Pass

Test Case 4: Health Management Function

Objective: Confirm health increases and decreases appropriately based on game events.

Table5.4: Health System Test

Test Input	Expected Output	Actual Output	Status
Player takes damage	Health decreases	Health decreases	Pass
Player uses health potion	Health increases	Health increases	Pass

5.2.2 Integration Testing

Integration testing ensures that independently developed modules work together seamlessly.

Test Case 5: Player Data Persistence

Objective: Verify that the player and quest progress are correctly saved and loaded.

Table 5.5: Player Data Persistence Test

Test Input	Expected Output	Actual Output	Status
Save game & reload	Data loaded correctly	Data loaded correctly	Pass

Test Case 6: Dialogue Integration

Objective: Check that quest triggers initiate appropriate dialogues and quest tracking.

Table 5.6: Quest & Dialogue Integration Test

Test Input	Expected Output	Actual Output	Status
Start quest and talk to NPC	Dialogue initiates	Dialogue initiates	Pass

5.2.3 System Testing

System testing validates the game as a whole has all components integrated or not.

Test Case 4: Full Gameplay Run-Through

Objective: Verify that the game runs without crashes from start to finish.

Table 5.7: Full Gameplay Test

System Component	Test Step	Expected Result	Status
Game Initialization	Start new game	Game loads into tutorial scene without issues	Pass
Movement & Navigation	Move player across map	Player moves smoothly with pathfinding	Pass
Quest System	Accept, track, and complete quests	Quests update and complete as designed	Partially Pass
Combat System	Engage and defeat various enemies	Health updates, enemy AI responds, combat feels smooth	Pass
Inventory System	Pick up, use, and drop items	Items behave correctly in inventory	Pass
Dialogue System	Interact with multiple NPCs	Dialogue trees trigger based on quest state	Pass
Save/Load	Save game, exit, reload	Game state restores correctly	Pass
User Interface	Use menus, HUD, inventory screens	UI is responsive and displays correct information	Pass
Performance & Stability	Play 1+ hour session with high activity (combat, quests, etc.)	No frame drops, crashes, or memory leaks	Partially Pass
Final Objective	Reach and complete final mission or boss	Game ends as designed with proper cutscene/end trigger	Pass

5.3 Result Analysis

After completing all the testing phases, we carefully analyzed the results to find any errors or performance problems. Each issue we found was documented and fixed quickly. To further improve the game, we also organized a playtest with a small group of people. Their feedback helped us adjust the game balance, fix bugs, and make the controls smoother. Since the game has three sub-modes 3D RPG Mode, Jigsaw Puzzle Mode, and Wordle Mode we tested each mode separately to make sure they worked well.

In the 3D RPG Mode, we focused on movement, quests, combat, and saving/loading. Testers suggested making the enemies more challenging and improving player movement, so we added sprinting and better enemy difficulty.

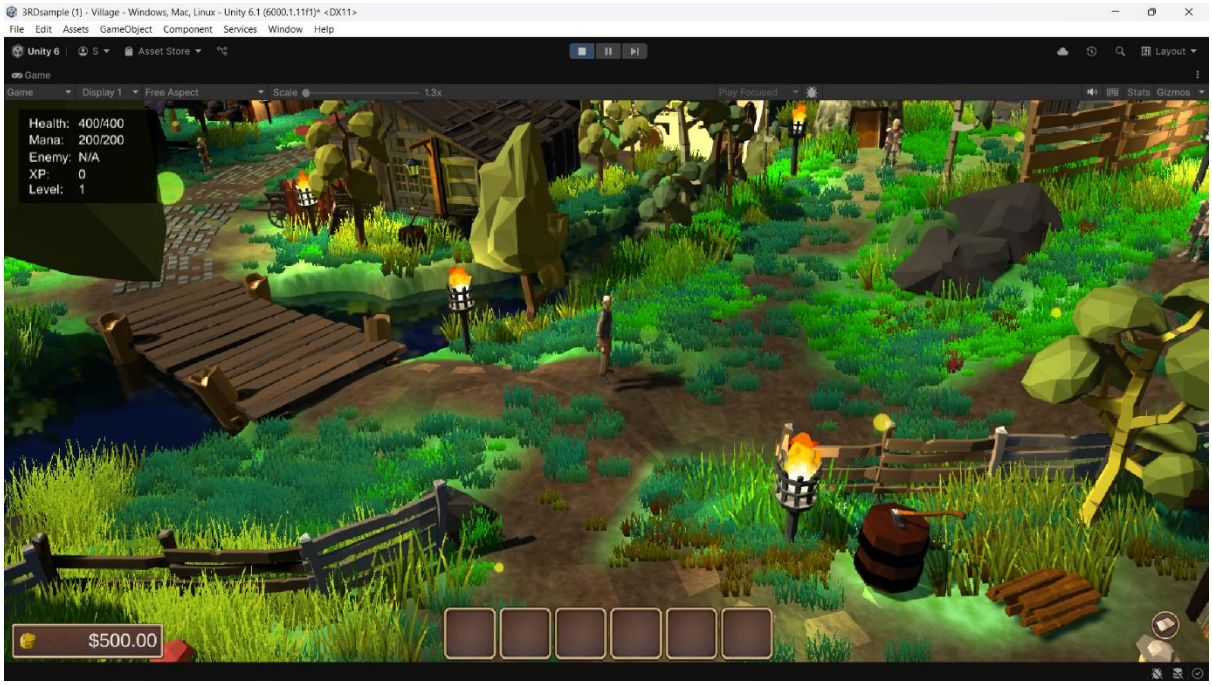


Figure 5.1: Improved player movement and sprinting

In the Jigsaw Puzzle Mode, we fixed issues with piece snapping and added a hint by attaching reference picture in background added timer so it would be more challenging.

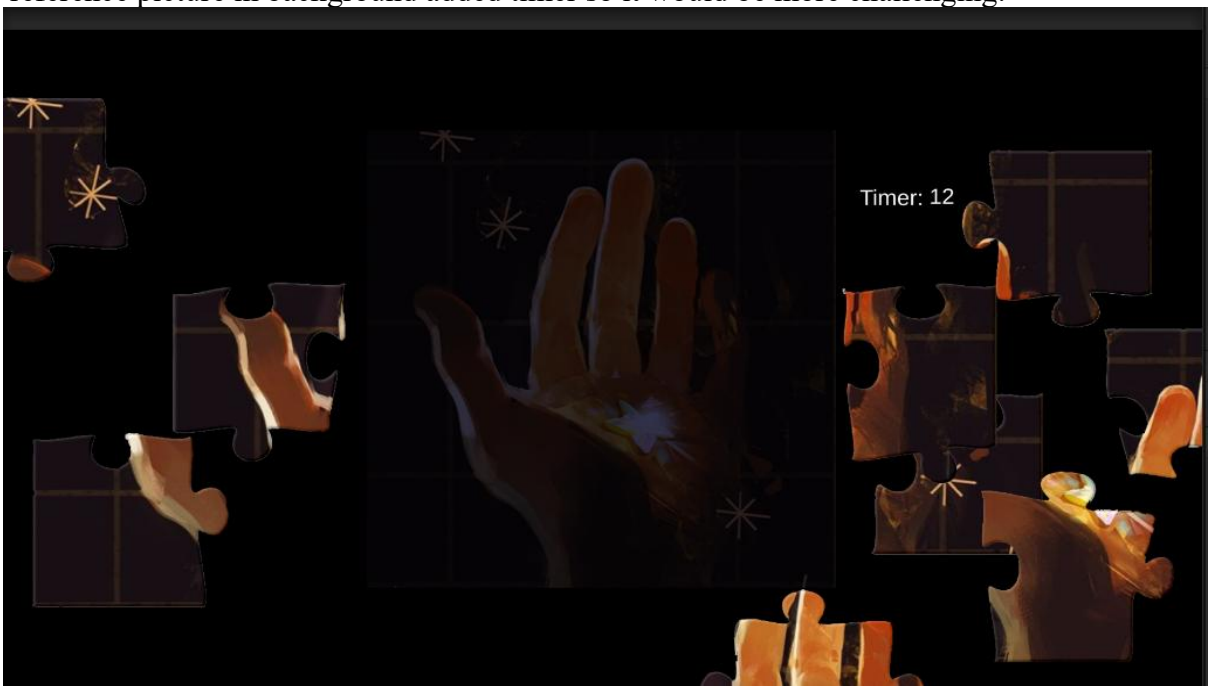


Figure 5.2: Solving Puzzle

Similarly, in the Wordle Mode, we improved the word list and fixed color for guesses.

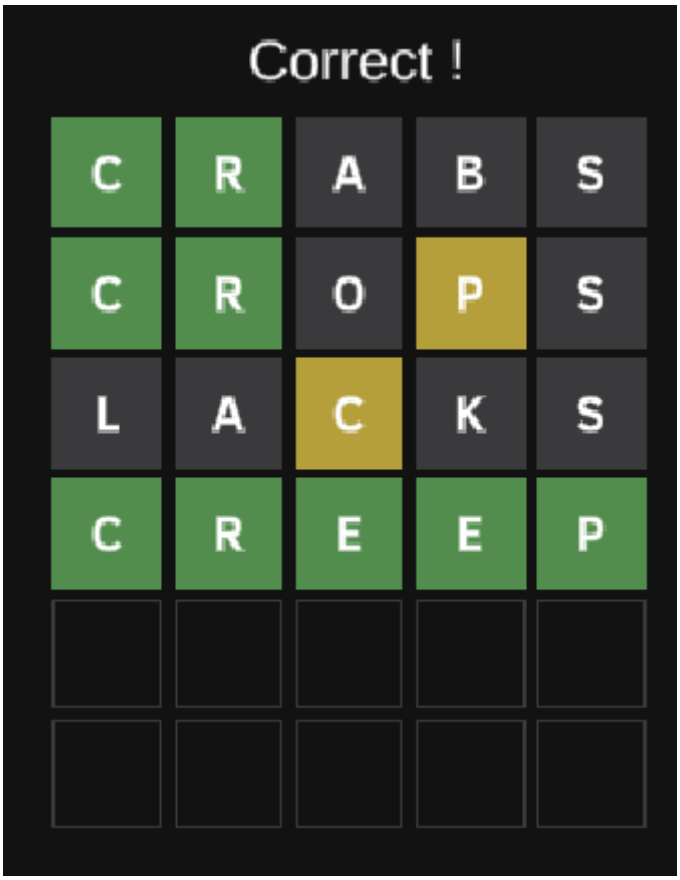


Figure 5.3: Added color according to testing and feedback for more improved and polished version

Chapter 6: Conclusion and Future Recommendations

6.1 Conclusion

This 3D open-world RPG project aims to create a rich and immersive role-playing experience set in a vast, explorable fantasy world. Built using the C# programming language and the powerful Unity Game Engine, the game integrates 3D models, textures, and animations created with asset tools like Blender and Substance Painter. Core gameplay mechanics such as questing, combat, and dialogue with non-player characters are driven by well-structured algorithms and event-driven architecture to ensure smooth, responsive interactions.

The game encourages players to explore diverse environments, uncover secrets, and grow stronger by completing missions, collecting resources, and defeating enemies. By utilizing procedural systems like random enemy spawning and dynamic quest generation, the game keeps every session engaging and unpredictable.

6.2 Future Recommendations

- **Player Customization & Progression**:
  Implement a deeper customization system that offers unlockable gear, player skill trees, mounts, and rare items as rewards for completed quests.
- **Multiplayer & Co-Op**:
  Include optional multiplayer features that let friends or complete strangers join each other's worlds for competitive combat or cooperative adventures.
- **Economy & Trading Systems**:
  Add an in-game economy with marketplaces, crafting systems, and player-to-player trading to encourage immersion and strategy.
- **Online Progress & Leaderboards**:
  Implement a global leaderboard and account system to encourage competitive or cooperative play across the player base.

References

[1] Brackeys, "How to make an RPG in Unity," 2018. [Online]. Available: https://www.youtube.com/playlist?list=PLPV2KyIb3jR4KLGCCAciWQ5qHudKtYeP7.

[2] V. Karamian, Building an RPG with Unity 2018: Leverage the power of Unity, Birmingham-mumbai: Packt>, 2018.

[3] U. Learn, "Learn game development with Unity," Unity Technologies, [Online]. Available: https://learn.unity.com. [Accessed 2 5 2025].

[4] H. Ferrone, Learning C# by Developing Games with Unity 2019 Fourth Edition Code in C# and build 3D games with Unity, BIRMINGHAM - MUMBAI: Packt Publishing , 2019.

[5] J. M. &. P. Buttfield-Addison, Mobile Game Development with Unity build once, Deploy Anywhere, 1005 Gravenstein Highway North, Sebastopol: O'Reilly Media, Inc., 2017.

[6] J.Hocking, Unity in Action: Multiplatform Game Development in C# with Unity 5, Manning Publication, 2015.

[7] E. L. Jr., Getting Started with Unity 2018: A beginner's Guide to 2Dand 3D Game Development with Unity, Mumbai: Packt Publishing, 2018.

[8] R. D. G. T. Ben Tristem, "RPG Core Combat Creator: Learn Intermediate Unity C# Coding," 2025. [Online]. Available: https://www.udemy.com/course/unityrpg.

[9] M. Geig, Sams Teach Yourself Unity 2018 Game Development in 24 Hours, Sams Publishing, 2018.

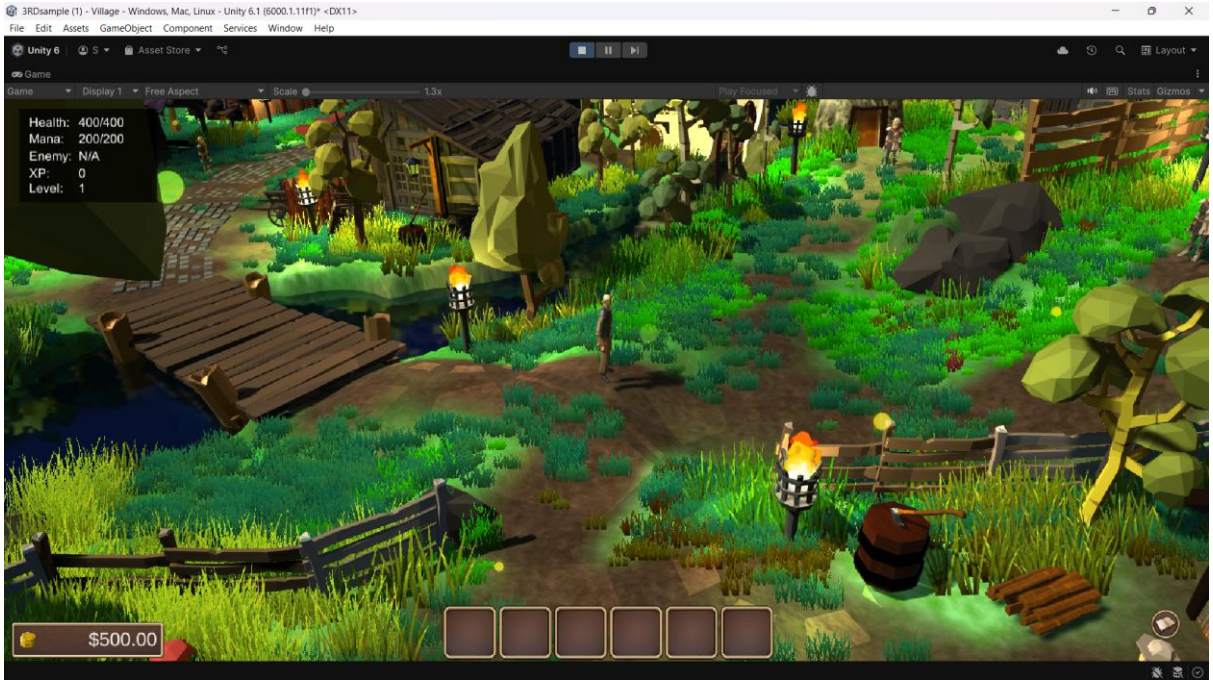
[10] T. lintrami, Game Development Essentials: Build fully functional 2D and 3D games with realistic environments, sounds, physics, special effects, and more!,, Packt Publishing, 2018.

Appendix

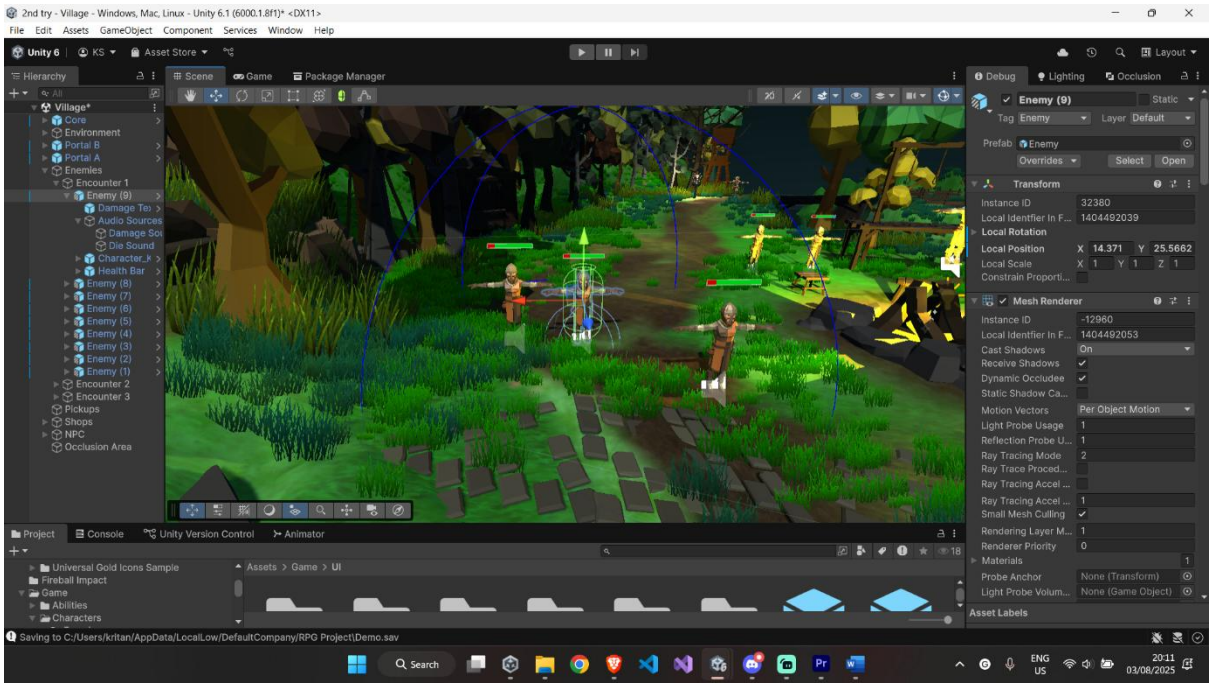
Screenshots of Game:

Main Menu:

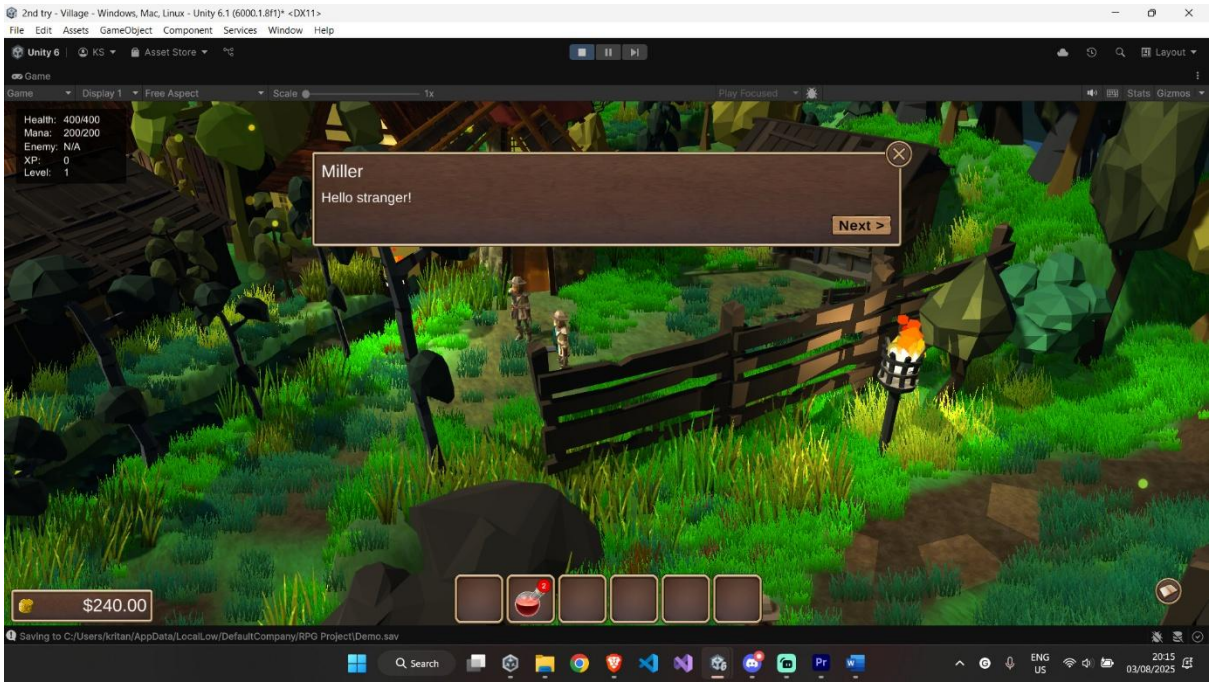
Beginning of Game:



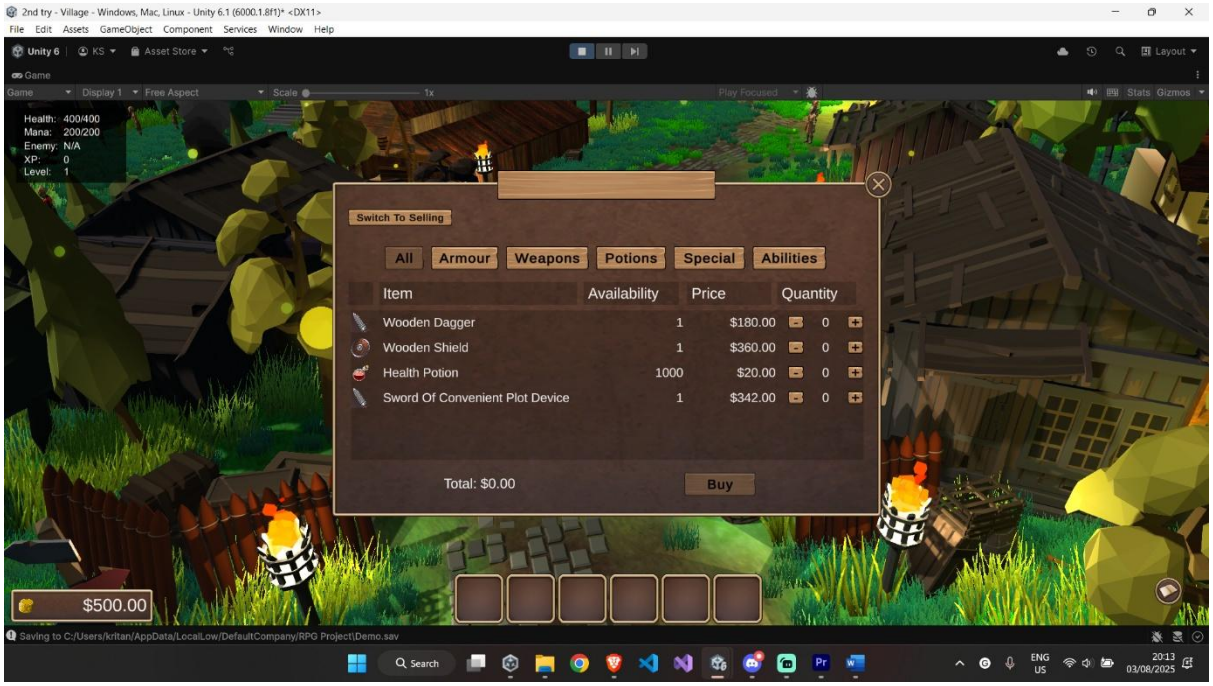
Enemy of Game:



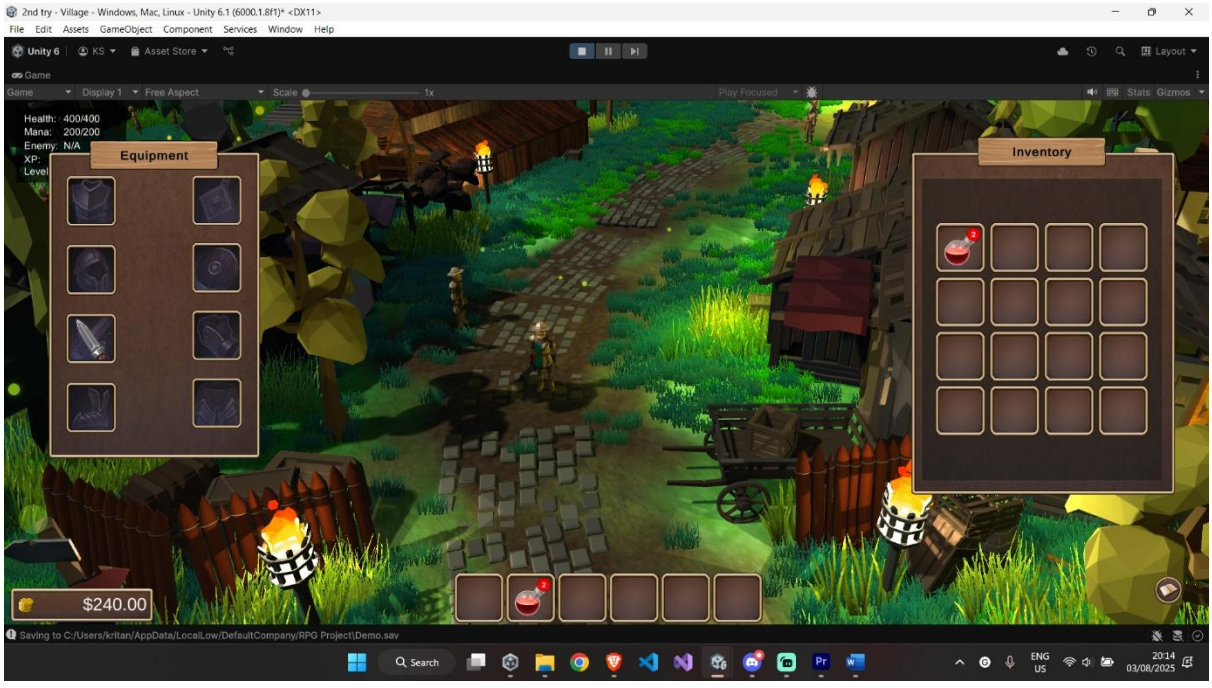
Dialog:



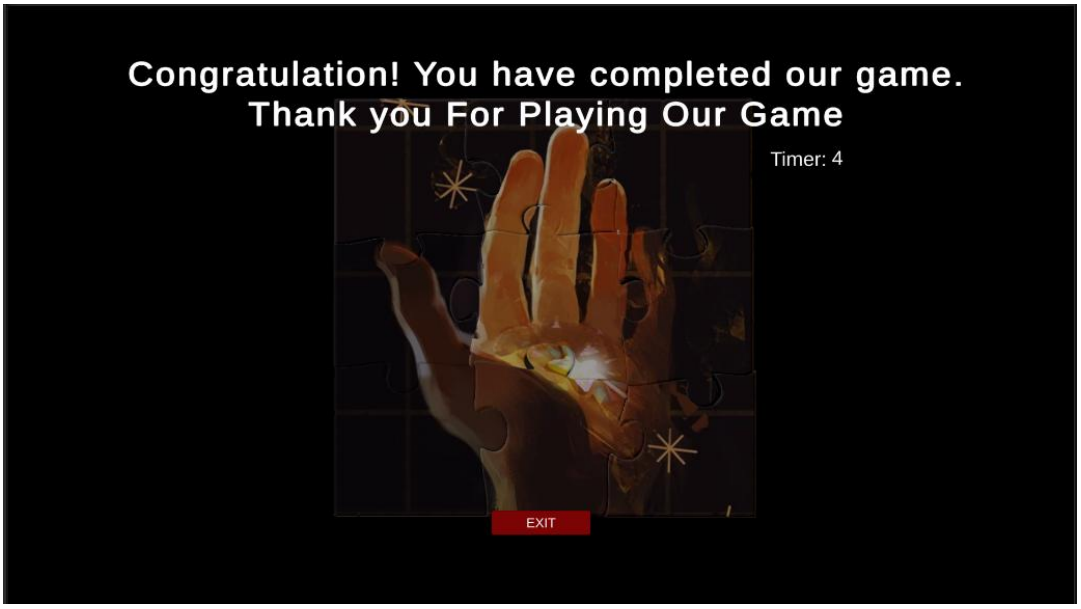
Shop:



Inventory system:



Puzzle:



Wordle:

Algorithm

Follow player Script:

Enemy AI Controller :