



Daffodil International University
Department of Software Engineering SE-431

Project/thesis Project Documentation

“Lactos World” A Open world Unreal Engine Game.

Submitted by

MD Sunbir Hasan

ID: 201-35-537

Department of Software Engineering
Daffodil International University

Supervised by

MD Rajib Mia

Lecturer

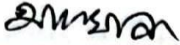
Department of Software Engineering
Daffodil International University

This Project report has been submitted in fulfilment of the requirements for the Degree
of Bachelor of Science in Software Engineering

APPROVAL

This project titled on “**Lactos World-A Open world Unreal Engine Game.**”, submitted by **MD SUNBIR HASAN (ID: 201-35-537)** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS



Afsana Begum
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Chairman



Md Rajib Mia
Lecturer
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 1



Musabbir Hasan Sammak
Lecturer
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 2



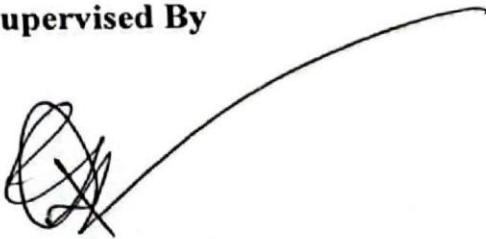
Mohammad Abu Yousuf, PhD
Professor
Institute of Information Technology
Jahangirnagar University

External Examiner

DECLARATION

I announce that I am rendering this study document under Md. Rajib Mia, Lecturer, Department of Software Engineering, Daffodil International University. I therefore, state that this work or any portion of it was not proposed here therefore for Bachelor's degree or any graduation.

Supervised By



Md. Rajib Mia

Lecturer

Department of Software Engineering
Daffodil International University

Submitted by



Md. Sunbir Hasan

ID: 201-35-537

Department of Software Engineering
Daffodil International University

ACKNOWLEDGE

Throughout the system's planning and development phases, my supervisor, MD. Rajib Mia, a lecturer in the Department of Software Engineering (SWE), was a tremendous help. I could not have progressed to this stage of my development without his shrewd counsel and tactical direction. He was a major contributor to the project's success, coming up with the original concept and offering constant assistance all along.

I want to express my appreciation to my friends, family, and coworkers for your unwavering support under all circumstance.

ABSTRACT

"Lactos World A Open world Unreal Engine Game."

An ambitious open-world game project called Lactos World aims to maximize player interaction and immersion. This ambitious project delivers a gaming experience that goes beyond accepted conventions by fusing innovative technology, imaginative design, and player-centric features.

This abstract provides a succinct summary of the extensive defensive documentation and provides an overview of the main elements of the Lactos World project.

INDEX

Approval.....	i
Declaration	ii
Acknowledgement.....	iii
Abstract	iv
Chapter 1 Introduction.....	1-3
1.1 Project Overview.....	1
1.2 Project Purpose.....	2
1.2.1 Proposed System	3
Chapter 2 System Analysis... ..	4-9
Feasibility Analysis.....	4
2.1.1 Technical Feasibility	4
2.1.2 Operational Feasibility.....	4
2.2 Functional Requirements... ..	5-6
2.3 System Requirements	7
2.4 Non-Functional Requirements... ..	7
2.5 Performance	8-9

Chapter 3 System Design	10-19
3.1 Development Model.....	10
3.1.1 Use Case Diagram.....	11
3.1.2 Use Case Descriptions.....	12-15
3.2 Activity Diagram.....	16ssn
3.3 Sequence Diagram.....	17
3.4 ERD Diagram.....	18
3.5 Class Diagram	19
Chapter 4 Development Tool and Technology... ..	20
4.1 Integrated Development Environment (IDE).....	20
4.2 Programming Language	20
4.3 User Interface Design.....	20
Chapter 5 System Testing... ..	21-25
5.1 Testing Features... ..	21
5.1.1 Feature to be tested.....	21
5.2 Testing Strategies... ..	22
5.2.1 Test Approach	22
5.2.2 Pass/Fail Criteria	23
5.2.3 Testing Schedule	24
5.3 Test Cases.....	2

Chapter 6 Gameplay Manual 26-34

6.1 Landing Page 26-34

Chapter 7 Conclusion 35

7.1 Future Roadmap... 35

Reference 35

CHAPTER 1: INTRODUCTION

The ambitious open-world game project Lactos World transports players to a huge and engrossing virtual realm. The game offers an unmatched gaming experience by fusing gorgeous graphics, inventive gameplay mechanisms, and an engaging story.

1.1 Project Overview

The demands of an ardent player base, technological improvements, and creative innovation are propelling the gaming industry's extraordinary evolution. The Lactos World project comes to light as a trailblazing effort that has the potential to completely alter the open-world gaming experience in response to this changing scenario.

1.2 Project Purpose

The Lactos World project aims to provide a unique gaming experience through a blend of innovative technology, artistic vision, and unwavering dedication. The general objectives of the project can be summed up as follows:

Elevating Player Experience:

Lactos World is conceived with the primary objective of providing players with a gaming experience that transcends conventional boundaries. The purpose is to immerse players in a meticulously crafted, expansive open world, offering a dynamic and captivating adventure that resonates with a broad audience.

Innovation in Gameplay:

At the core of the project is a commitment to innovate in gameplay mechanics. Whether through the seamless exploration of a vast and diverse world, the intuitive combat system, or the intricate crafting and economic systems, Lactos World seeks to push the envelope of what players expect from an open-world game.

Storytelling and Player Agency:

The project aims to weave a rich narrative tapestry, immersing players in a captivating storyline with deep lore. Importantly, Lactos World places significant emphasis on player agency, allowing individual choices

to influence both personal experiences and the overarching narrative, providing a sense of ownership in the game world.

Technical Excellence:

Lactos World is built on the robust Unreal Engine 4, incorporating advanced graphics and technical features. The purpose is to deliver a visually stunning and technically impressive gaming environment while ensuring smooth performance across various gaming platforms.

Cross-Platform Accessibility:

The decision to develop Lactos World for PC, PlayStation, and Xbox platforms is driven by the purpose of reaching a diverse gaming audience. Cross-platform accessibility ensures that players can engage with the game regardless of their preferred gaming device, fostering inclusivity.

1.2.1 Proposed System

The comprehensive framework of the Lactos World proposed system is intended to realize the project's goals. It includes a range of parts, tools, and techniques to guarantee the game's effective creation, implementation, and upkeep. An outline of the suggested system is provided here:

Game Engine:

- Utilize Unreal Engine 5.3 as the primary game engine for its powerful capabilities in rendering, physics, and scalability.
- Leverage the engine's Blueprint system for rapid prototyping and iterative development.

Performance Optimization:

- Conduct rigorous performance testing to optimize the game for various hardware configurations.
- Implement level-of-detail (LOD) systems, occlusion culling, and other optimization techniques to maintain smooth gameplay.

Post-Launch Support:

- Commit to providing post-launch support with regular updates, addressing player feedback, and addressing emerging issues.
- Plan and execute downloadable content (DLC) releases to expand the game universe and introduce new features.

Chapter 2: SYSTEM ANALYSIS

2.1 Feasibility Analysis

The feasibility analysis for Lactos World assesses the practicality and viability of the project from various perspectives, including technical, operational, economic, and scheduling considerations.

2.1.1 Technical Feasibility:

- **Technology Stack:** The use of Unreal Engine 4 provides a robust foundation for developing a visually stunning and technically sophisticated game. The engine's capabilities in rendering, physics, and cross-platform compatibility contribute to the technical feasibility of the project.
- **Platform Compatibility:** The decision to develop for PC, PlayStation, and Xbox ensures broad market reach. Cross-platform development, although challenging, is technically feasible with proper planning and resource allocation.
- **Scalability:** The technical architecture allows for scalability, accommodating future updates, expansions, and the integration of advanced features.

2.1.2 Operational Feasibility:

- **Development Team:** Assessing the availability of skilled developers, designers, and other key personnel is crucial. The feasibility analysis considers the team's expertise and experience with Unreal Engine 4 and open-world game development.
- **Collaborative Development:** Collaboration tools and methodologies are implemented to facilitate effective communication and coordination among team members, promoting a streamlined development process.
- **Iterative Development:** The feasibility of adopting an iterative development approach is crucial for adapting to changing requirements and player feedback during the development cycle.

2.2 Functional Requirements

Functional requirements describe the specific features and functionalities that the Lactos World game must have to meet its objectives. Here is a set of functional requirements for the game:

➤ **Character Creation:**

- Players can create and customize their in-game characters, including appearance, skills, and abilities.

➤ **Open World Exploration:**

- Seamless and expansive open-world environment for players to explore.
- Diverse landscapes, regions, and ecosystems with unique challenges and rewards.

➤ **Quest System:**

- Varied and engaging quests catering to different playstyles.
- Quest tracking and journal system for player reference.
- A mix of main storyline quests, side quests, and dynamic events.

➤ **Combat System:**

- Intuitive combat mechanics supporting various weapons and skills.
- Realistic animations and responsive controls.
- Enemies with diverse behaviors and challenges.

➤ **Crafting and Economy:**

- Robust crafting system allowing players to gather resources and create items.
- In-game economy influenced by player actions, trade, and supply and demand dynamics.

➤ **Dynamic Day-Night Cycle:**

- Realistic and dynamic transitions between day and night.
- Influences on gameplay, creature behavior, and environmental conditions.

➤ **Storyline and Lore:**

- Gripping narrative with deep lore and character development.
- Player choices impacting the unfolding story.
- Branching storylines based on player decisions.

➤ **Graphics and Visual Effects:**

- High-definition visuals with advanced graphics features.

- Realistic lighting, particle effects, and environmental details.
- **Inventory System:**
 - Manageable inventory system for storing items and equipment.
 - Sorting and categorization options for player convenience.
- **Dynamic Weather System:**
 - Realistic and dynamic weather conditions, impacting gameplay.
 - Environmental effects such as rain, snow, and storms.
- **User Interface (UI):**
 - Intuitive and user-friendly interface for easy navigation.
 - HUD elements displaying essential information, such as health, inventory, and quest status.
- **Sound and Music:**
 - Immersive sound effects corresponding to in-game actions and events.
 - Evocative background music enhancing the gaming experience.
- **Save and Load System:**
 - Autosave and manual save options for player progress.
 - Load system allowing players to resume gameplay from saved points.
- **Accessibility Features:**
 - Customizable controls and settings for players with different preferences.
 - Subtitle options for dialogue and in-game audio.

2.3 System Requirements

➤ Hardware Requirements:

Operating System: Windows 10/11 (64-bit)

Processor: Intel Core i3-9100F or AMD Ryzen 3 3200G

Memory: 8 GB RAM

Graphics: NVIDIA GeForce GTX 1050Ti 4GB or AMD Radeon RX 580 8GB

DirectX: Version 11

Storage: 65 GB available space

Sound Card: DirectX compatible soundcard or onboard chipset

2.4 Non-Functional Requirements

➤ Performance:

- Load Time: The game should load within 15 seconds on average hardware configurations.
- Response Time: The game should respond to user inputs within 100 milliseconds to maintain a smooth and responsive experience.

➤ Scalability:

- The system should be scalable to accommodate a growing player base, ensuring stability and performance even during peak usage.

➤ Availability:

- The game should have a minimum uptime of 99.5%, allowing for routine maintenance and updates.

➤ Reliability:

- The system should be reliable, with a low probability of crashes or unexpected downtime.
- The game should autosave progress at regular intervals to prevent data loss.

2.5 Performance

➤ **Response Time:**

- The game should exhibit low latency, responding to user inputs quickly and efficiently.
- Aim for an average response time of 100 milliseconds or less to maintain smooth and responsive gameplay.

➤ **Load Time:**

- Optimize the game's loading times to enhance the overall user experience.
- Implement efficient resource loading strategies to minimize initial load times and level transitions.

➤ **Frame Rate:**

- Ensure a consistent and high frame rate to provide smooth animation and visual experiences.
- Target a minimum of 30 frames per second (FPS) for consoles and 60 FPS for PC, with support for higher frame rates on capable hardware.

➤ **Scalability:**

- Design the game to scale across a variety of hardware configurations, allowing players with different setups to enjoy the game.
- Implement graphics settings and optimization options to accommodate a wide range of performance capabilities.

➤ **Network Performance:**

- Optimize network performance to minimize latency and provide a smooth multiplayer experience.
- Implement efficient networking code, employ server-client prediction mechanisms, and consider the impact of network conditions on player interactions.

➤ **Memory Management:**

- Implement efficient memory management to minimize resource usage and prevent memory leaks.
- Optimize textures, models, and other assets to ensure the game runs smoothly within the constraints of various hardware specifications.

➤ **Rendering Efficiency:**

- Optimize rendering techniques to maximize graphical fidelity while maintaining performance.
- Implement level-of-detail (LOD) systems, occlusion culling, and other rendering optimizations to balance visual quality and performance.

➤ **Stability:**

- Conduct rigorous testing to identify and address bugs, crashes, and performance bottlenecks.
- Implement effective error handling and logging mechanisms to diagnose issues promptly.

➤ **Update Performance:**

- Develop a streamlined process for delivering updates and patches to minimize disruption to players and maintain a smooth gaming experience.

Chapter 3: System Design

The choice of a specific system design approach in software engineering often depends on various factors, including the complexity of the project, development team expertise, project requirements, and scalability considerations. For a sophisticated and expansive open-world game like Lactos World, a combination of several system design principles and methodologies would be suitable.

1. Object-Oriented Design (OOD):

- Utilize object-oriented principles for modeling and organizing the game's components. This helps in encapsulating behavior, promoting code reuse, and managing the complexity of a large-scale project.

2. Service-Oriented Architecture (SOA):

- Implement a service-oriented architecture to break down the game into modular and loosely coupled services. This allows for easier scalability, maintainability, and the potential for incorporating microservices.

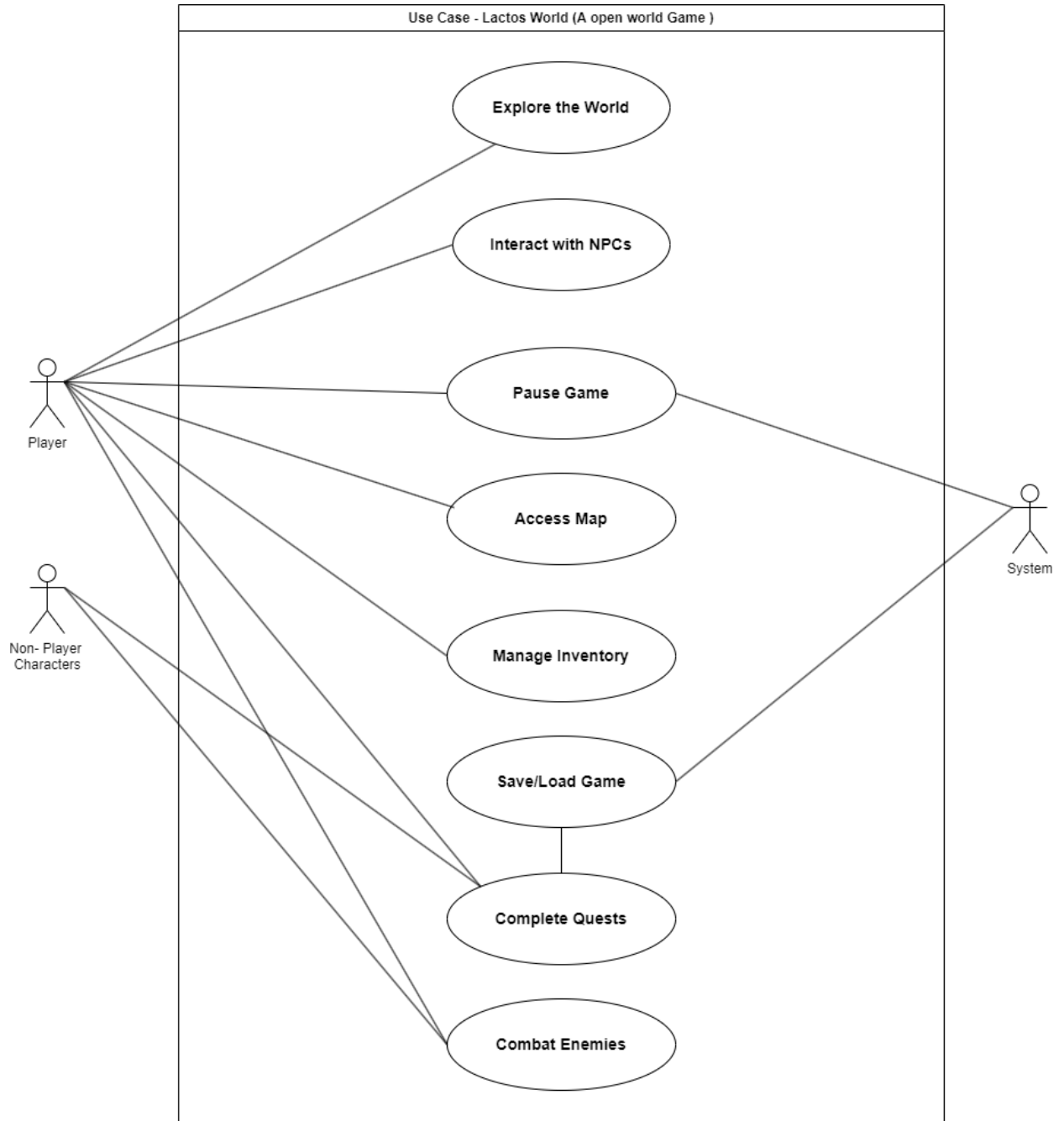
3. Model-View-Controller (MVC):

- Apply the MVC design pattern to separate the game's internal representation (model), user interface (view), and user input (controller). This enhances maintainability and allows for easier updates to specific components.

4. Entity-Component-System (ECS):

- Utilize an entity-component-system architecture for managing game entities and their behaviors. This provides flexibility, scalability, and efficient performance, especially in the context of an open-world environment with numerous interactive elements.

5. 3.1.1 Use Case Diagram



3.1.2 Use Case Descriptions

1. Explore the World:

Description	The player can freely explore the open-world environment.
Pre-Condition	Have to Install the game files on device.
Flow And Counts	<ul style="list-style-type: none">• Control Up and Down and Round with mouse• Go forward and backward with keyboard (W,S,A,D)

2. Interact with NPCs:

Description	The player can interact with non-playable characters (NPCs) for dialogues, trading, or quests.
Pre-Condition	Player Should Go Near the polyzoon of NPC.
Flow And Counts	<ul style="list-style-type: none">• Go near NPC• Press “E” to talk with NPC

3. Access Map:

Description	The player can access a map of the open world, helping with navigation.
Pre-Condition	Open map using Keyboard's (M) Button
Flow And Counts	<ul style="list-style-type: none">• Open map• Select Location

4. Manage Inventory:

Description	The player can manage and organize their inventory, including items, weapons, and armor.
Pre-Condition	Player Should Go Near the polyzoon of NPC.
Flow And Counts	<ul style="list-style-type: none">• Go near NPC• Press “E” to talk with NPC

5. Complete Quests:

Description	The player can accept and complete quests provided by NPCs or some certain location.
Pre-Condition	By completing all task
Flow And Counts	<ul style="list-style-type: none">• Complete all task• Pick up some specific item.

6. Combat Enemies:

Description	The player engages in combat with in-game enemies, such as monsters or hostile NPCs.
Pre-Condition	Have Specific item or weapon to attack
Flow And Counts	<ul style="list-style-type: none">• Pick up weapon form location or creates.• Pick ammo for weapon

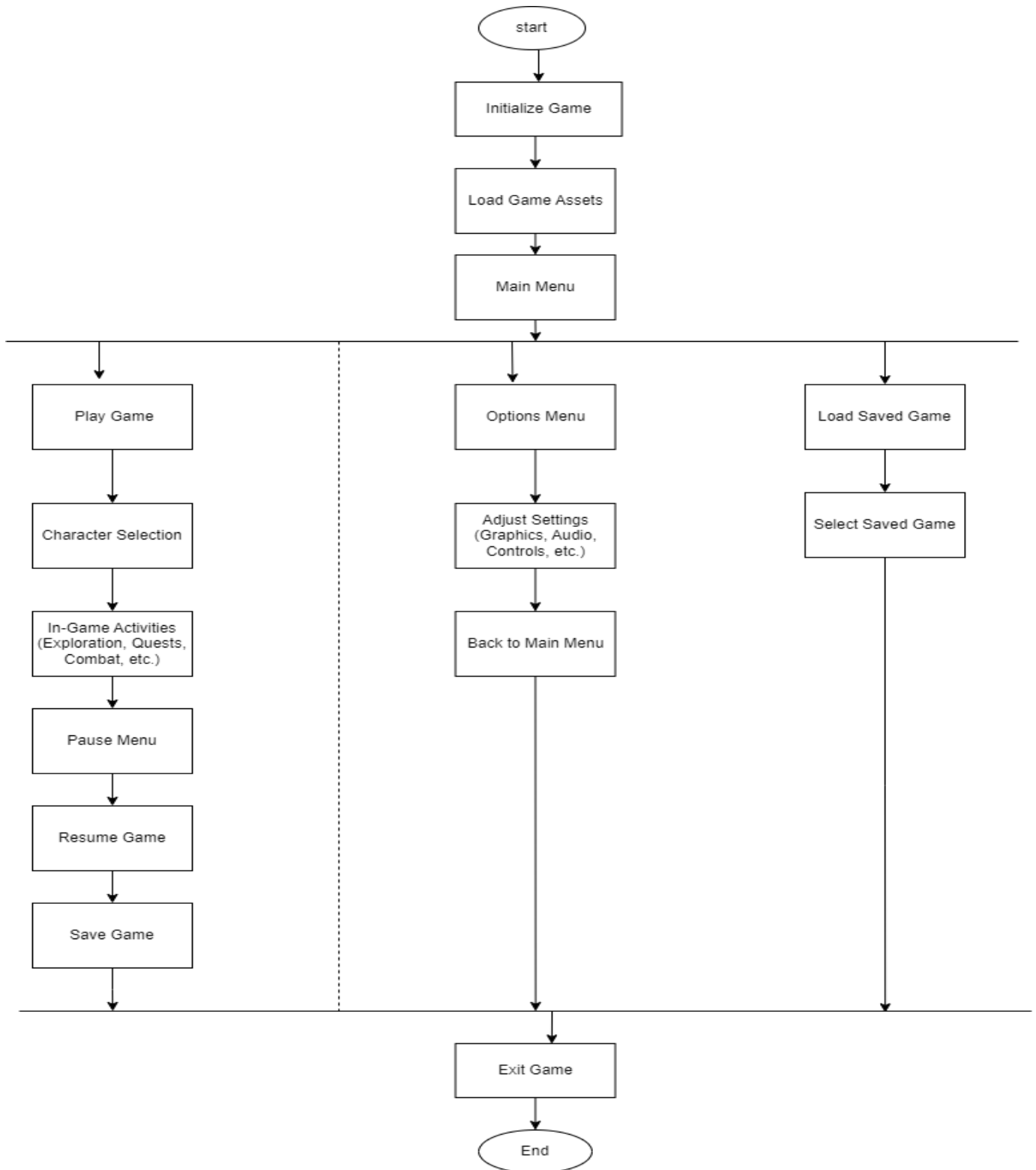
7. Pause Game:

Description	The player can pause the game to access settings, controls, or take a break.
Pre-Condition	“ESC” Button for Pause the game

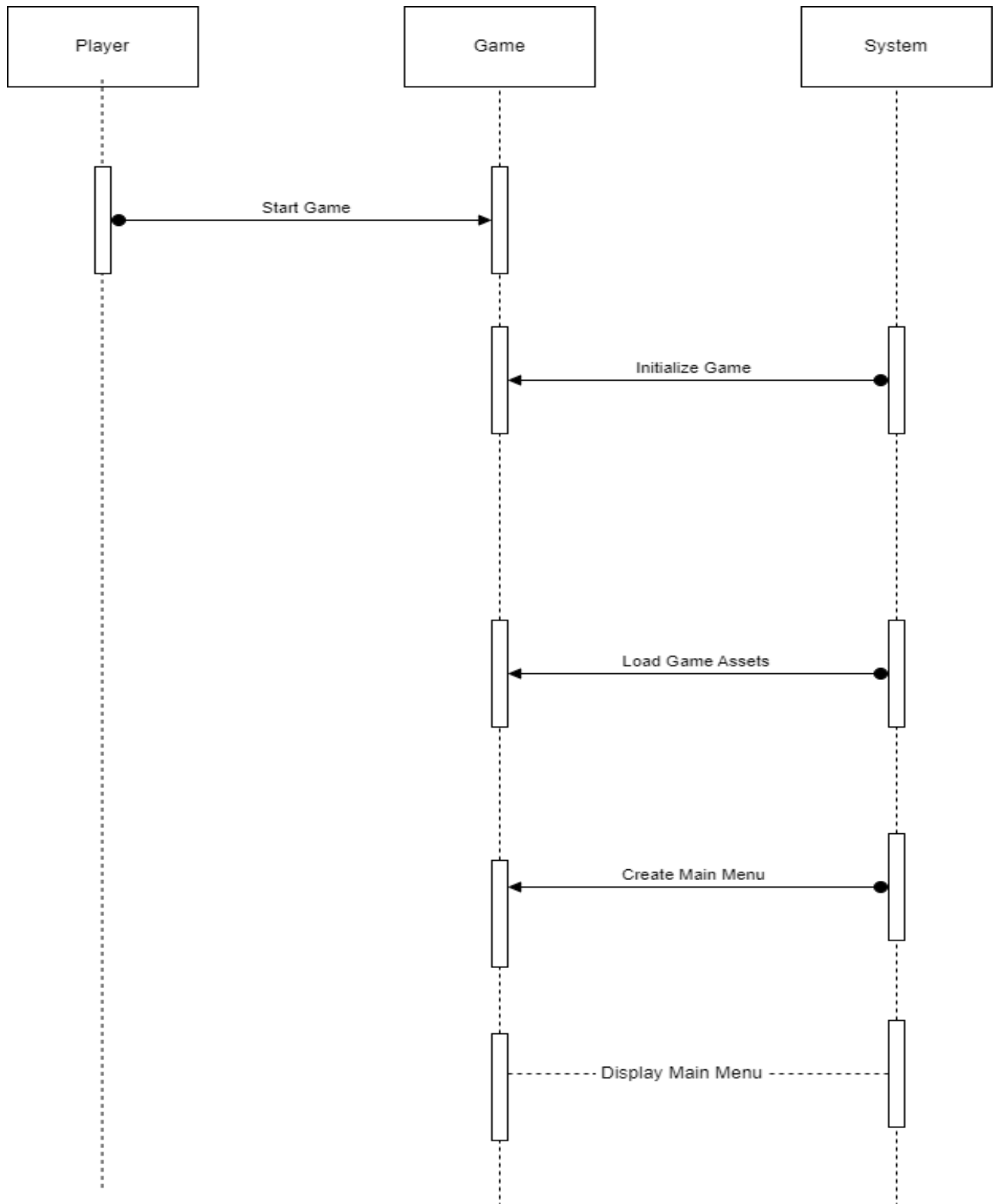
8. Save/Load Game:

Description	The player can save and load game progress.
Pre-Condition	Auto save when a task in complete

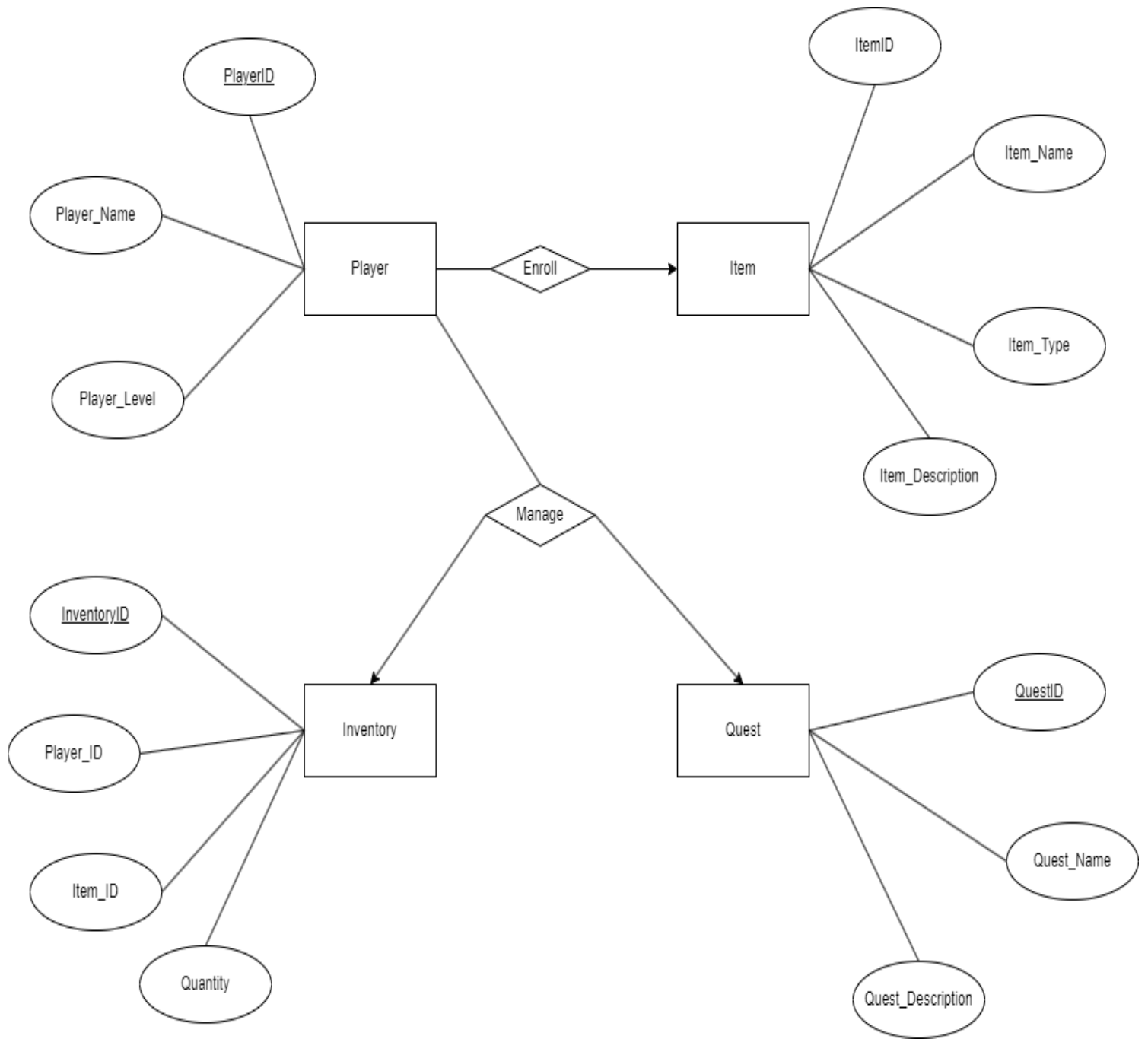
3.2 Activity Diagram



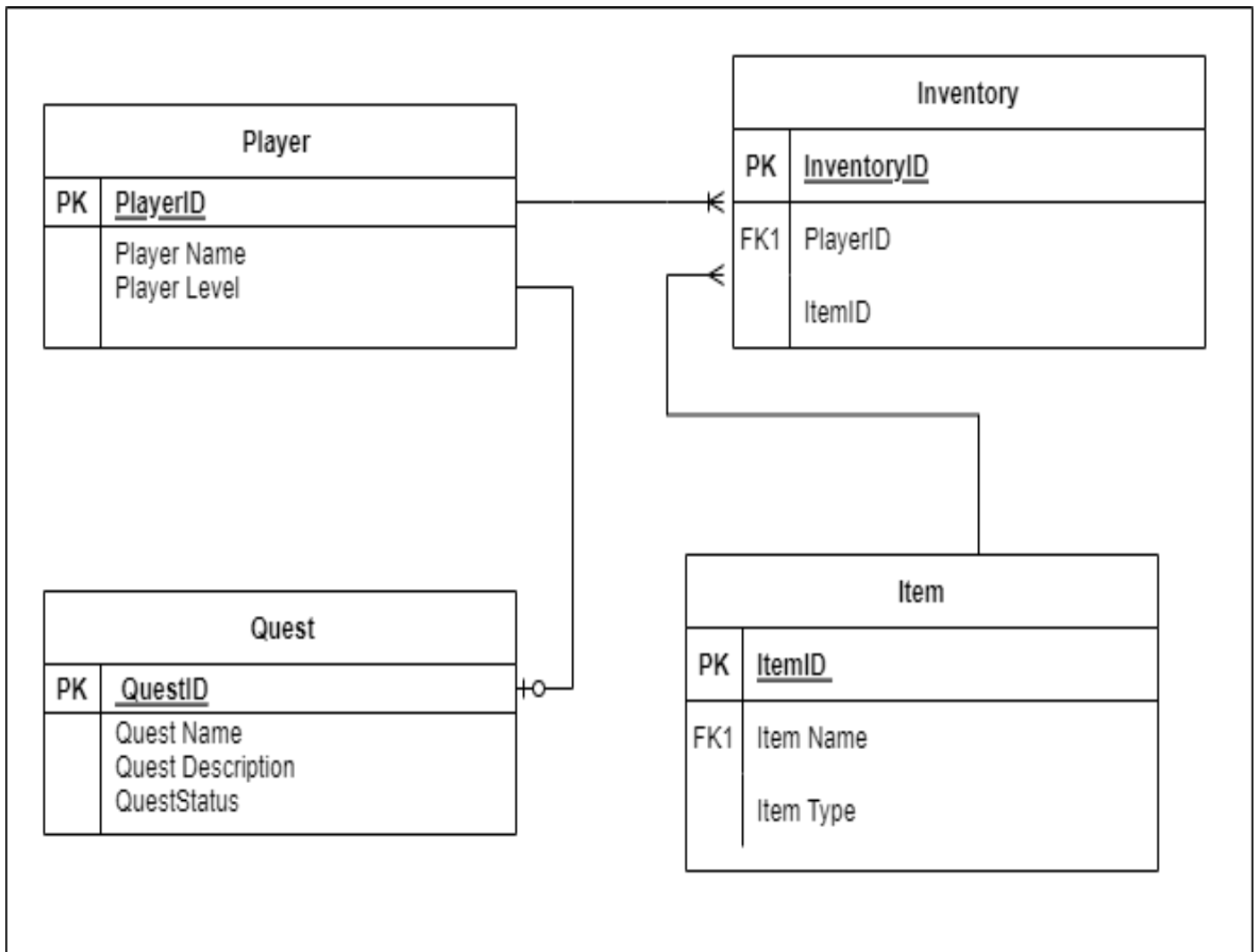
3.3 Sequence Diagram



3.4 ER Diagram



3.5 Class Diagram



Chapter 4: Development Tool and Technology

4.1 Integrated Development Environment (IDE)

- Unreal Engine 5.1/5.2/5.3
- Blender
- Meta Human
- Ableton Live

4.2 Programming Language

- C#

4.3 User Interface Design

- Adobe Creative Cloud Suite
- Photoshop
- Illustrator



Chapter 5: System Testing

Test the core gameplay mechanics, including character movement, combat interactions, quest systems, and crafting elements. Verify that the main storyline and side quests progress logically, considering player choices and consequences.

5.1.1 Feature to be tested

Features	Priority Description
Gameplay Mechanics	Exploration mechanics, such as jumping, climbing, and swimming
Quests and Narrative	Main storyline progression and branching narratives
Crafting and Economy	Trading and player-driven economy interactions.
Player Progression	Skill trees and character customization.
Dynamic Day-Night Cycle	Realistic transitions between day and night.
User Interface (UI)	Clarity and intuitiveness of menus and HUD elements.

5.2 Testing Strategies

5.2.1 Test Approach:

Testing strategies and test approaches for Lactos World should be comprehensive and tailored to the unique characteristics of an open-world game. Here is an outline of potential testing strategies and approaches:

1. Test Levels:

➤ Unit Testing:

- Focus on testing individual components, functions, and methods in isolation.
- Utilize automated unit tests to ensure the correctness of code at the smallest level.

➤ Integration Testing:

- Verify the interactions and data flow between different components.
- Conduct tests to ensure the seamless integration of various subsystems, such as AI, physics, and rendering.

➤ System Testing:

- Evaluate the entire system's functionality, including core gameplay mechanics, quests, multiplayer features, and UI.
- Perform end-to-end testing to simulate real-world player scenarios.

➤ Acceptance Testing:

- Involve stakeholders and real players to validate that the game meets their expectations.
- Execute acceptance tests to ensure that the delivered product aligns with the defined requirements.

5.2.2 Pass/Fail Criteria:

Unit Testing:

Pass Criteria:

- All unit tests pass without errors.
- Code coverage for unit tests meets or exceeds the defined threshold (e.g., 80%).
- No critical defects are identified during unit testing.

Fail Criteria:

- Any unit test fails to execute or returns unexpected results.
- Code coverage is below the defined threshold.
- Critical defects are identified and not addressed.

Integration Testing:

Pass Criteria: Data flow between subsystems is seamless, and data integrity is maintained.

Fail Criteria: Critical defects remain unresolved.

System Testing:

Pass Criteria: Core gameplay mechanics work as intended. Performance meets predefined standards under normal conditions.

Fail Criteria: Multiplayer features experience significant issues.

Performance Testing:

Pass Criteria: Acceptable frame rates are maintained under various conditions. Loading times meet or exceed defined thresholds.

Fail Criteria: Unacceptable frame rates affect gameplay experience.

5.2.3 Testing Schedule

Test Phase Time
Testing plan create 1 Week.
Unit testing During development time.
Integration Testing Phase time 3 weeks.
System Testing Phase 4 Week.
Performance testing 2 Week.
Compatibility Testing Phase 2 Week.
Regression Testing and Automation Phase 2 weeks.
Localization Testing Phase 2 weeks
Beta Testing and Player Feedback Integration 2 weeks.

5.3 Test Cases:

❖ **Player Movement:**

- Test Case 1: Verify that the player character moves in response to directional input.
- Test Case 2: Confirm that the player can perform basic movements such as walking, running, and jumping.

❖ **Combat Mechanics:**

- Test Case 3: Validate the accuracy of player attacks and weapon interactions.
- Test Case 4: Ensure that defensive actions, such as blocking or dodging, are functioning correctly.

❖ **Quest System:**

- Test Case 5: Test the initiation and completion of a main storyline quest.
- Test Case 6: Verify that side quests progress appropriately based on player choices.

❖ **Crafting and Inventory:**

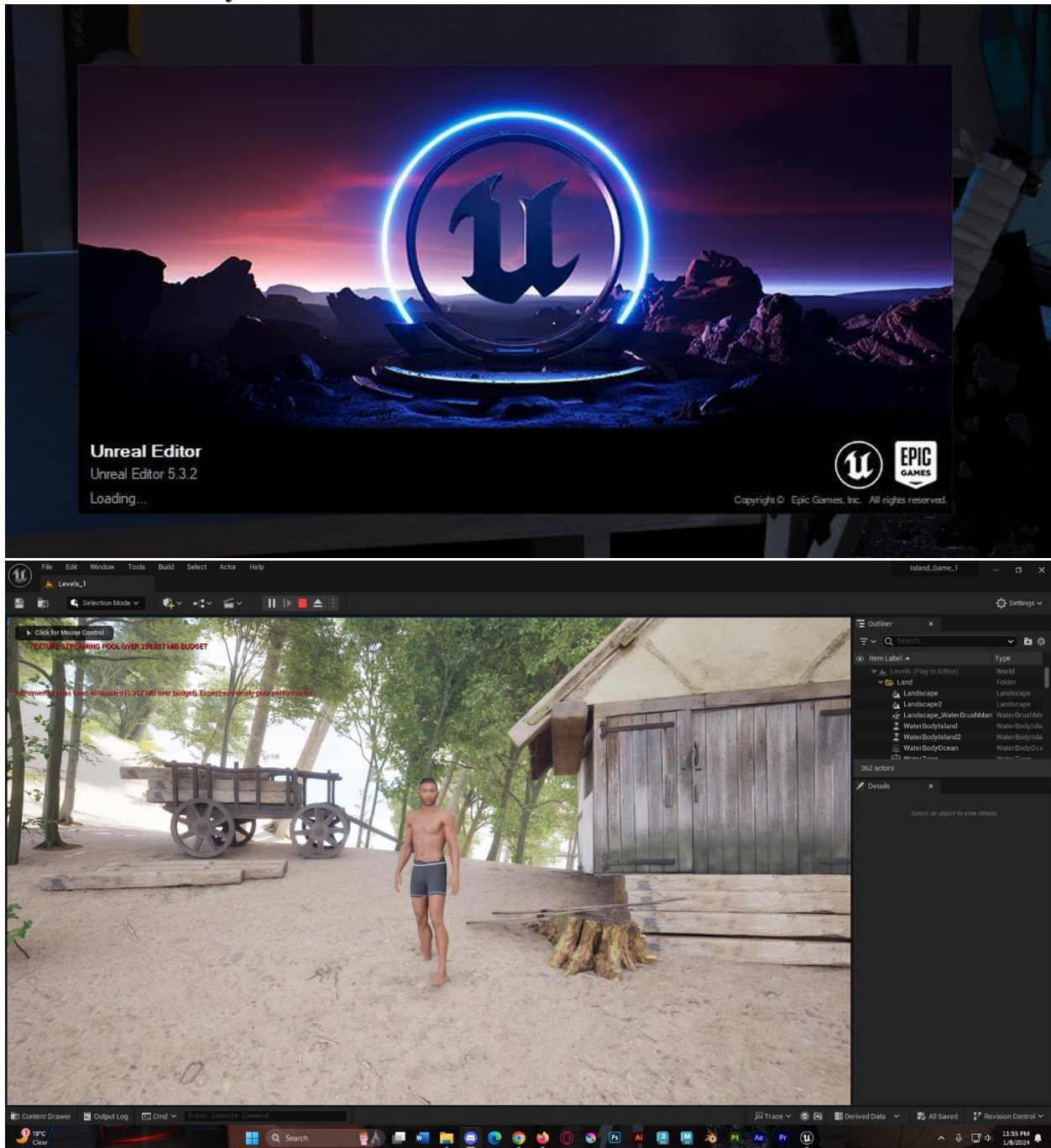
- Test Case 7: Confirm that players can gather resources and use them for crafting.
- Test Case 8: Validate the inventory system for proper item storage and retrieval.

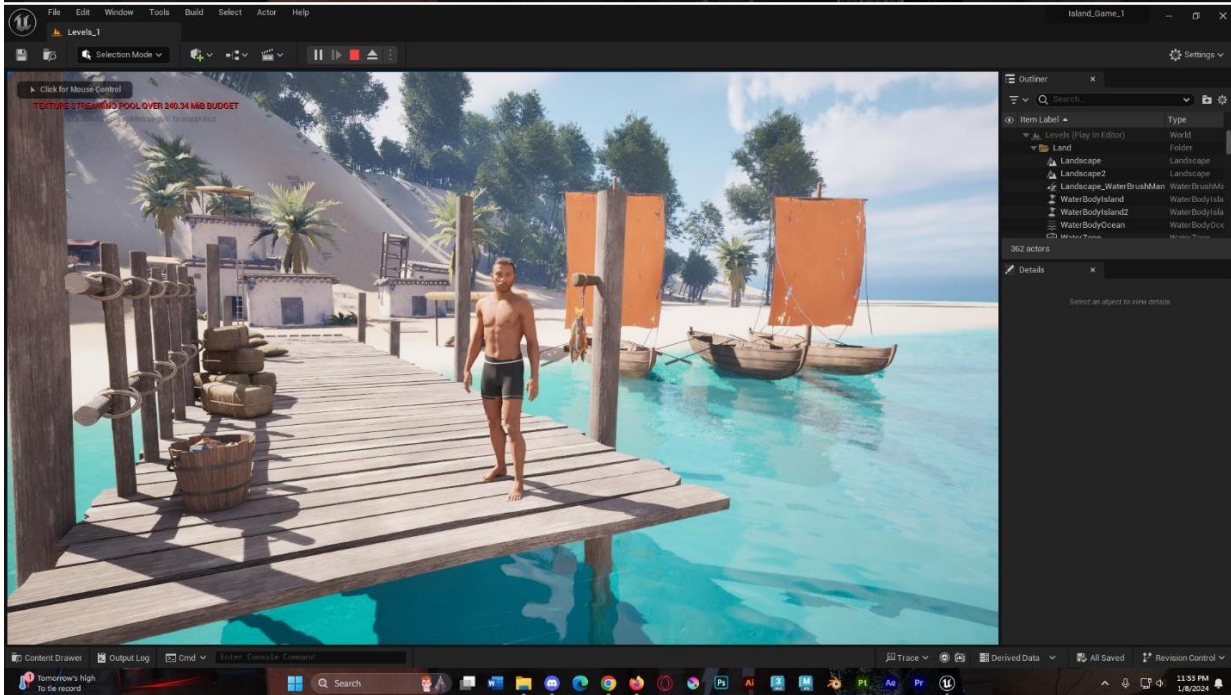
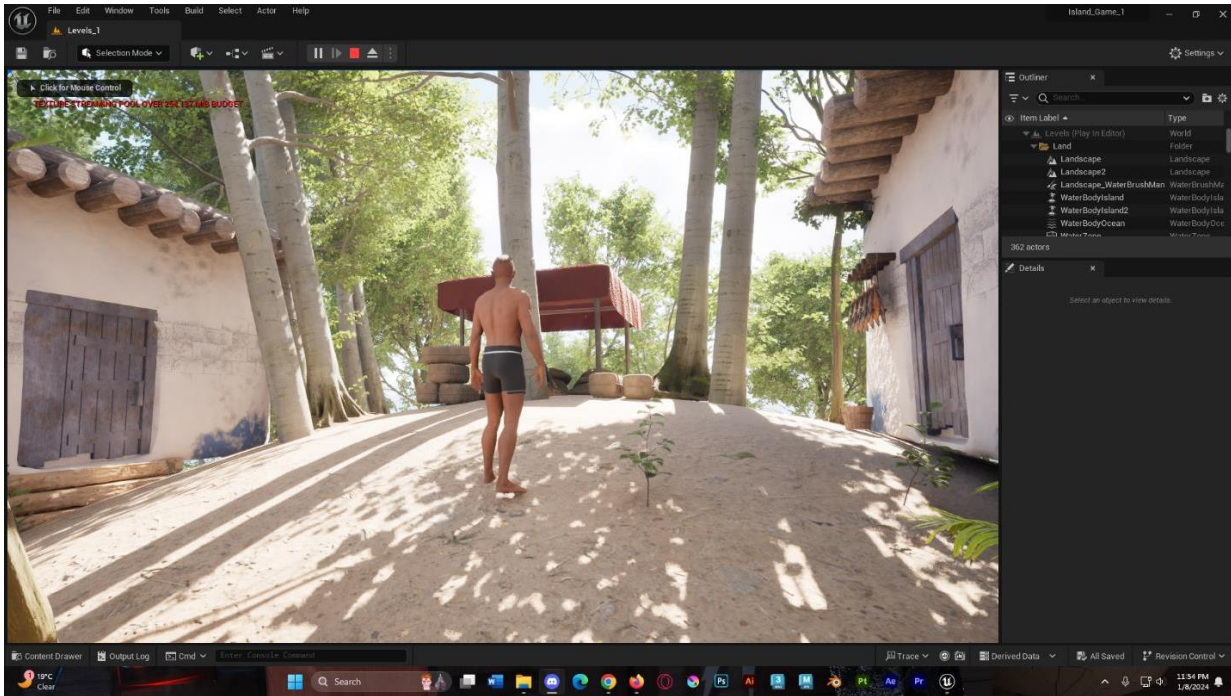
❖ **Integration of Gameplay Elements:**

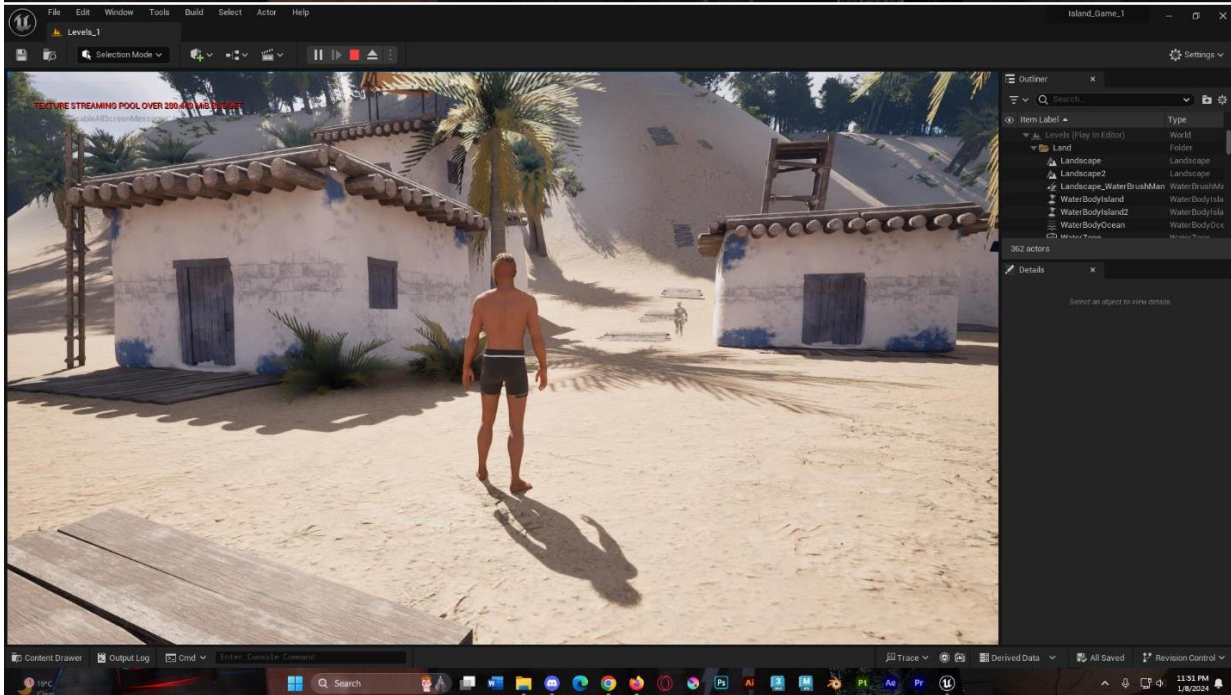
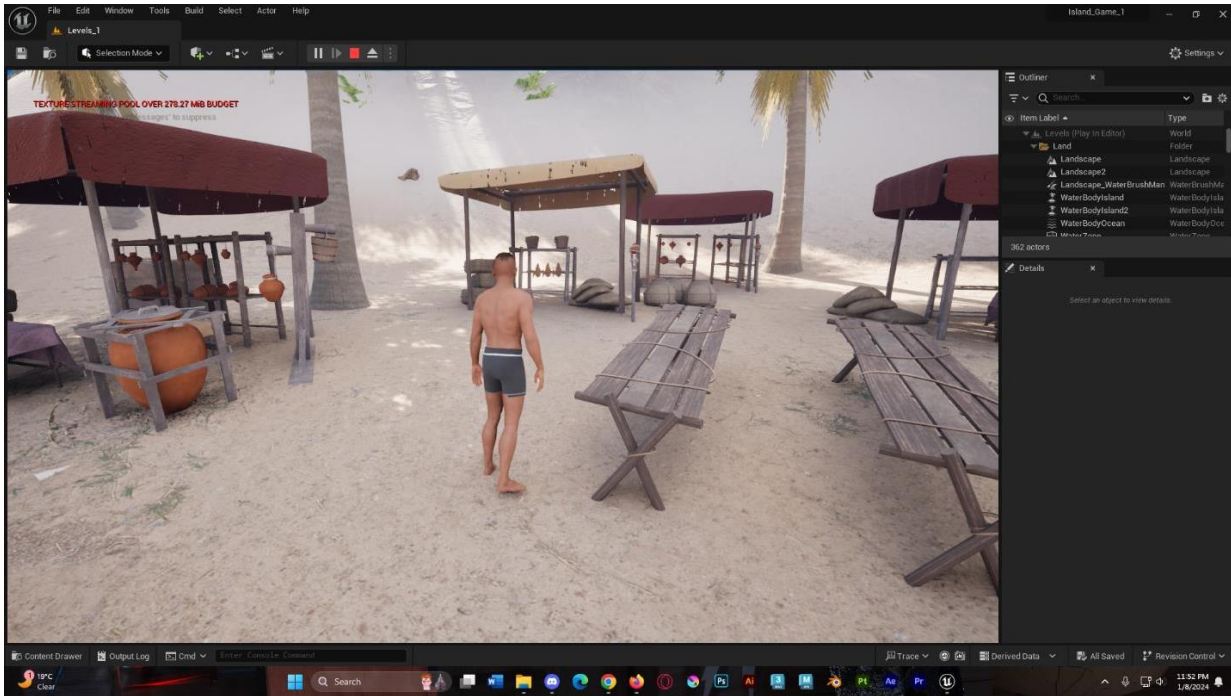
- Test Case 9: Verify that combat mechanics integrate seamlessly with the quest system.
- Test Case 10: Confirm that player movement interacts appropriately with the environment.

Chapter 6: Gameplay Manual

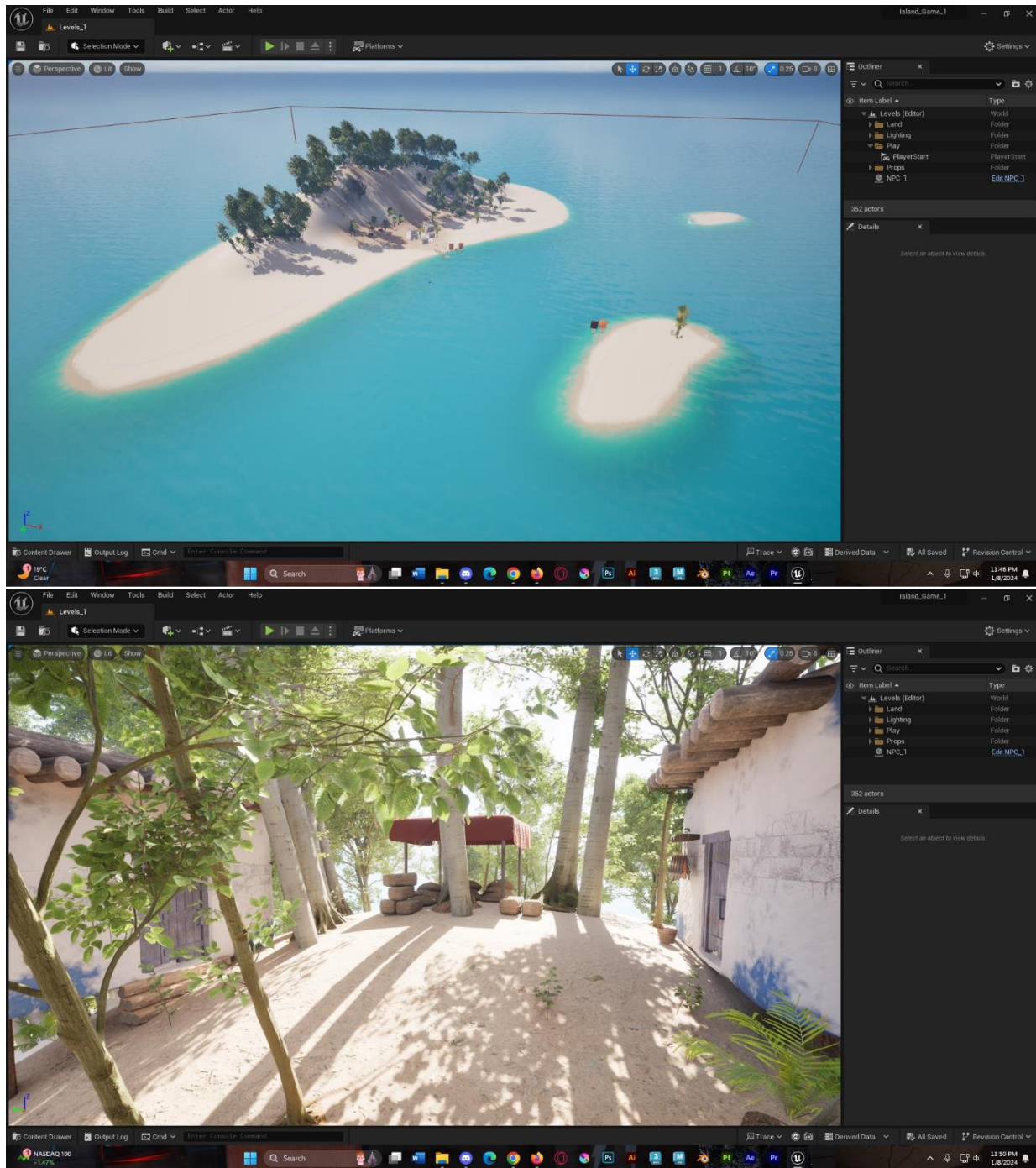
6.1 Game Play :

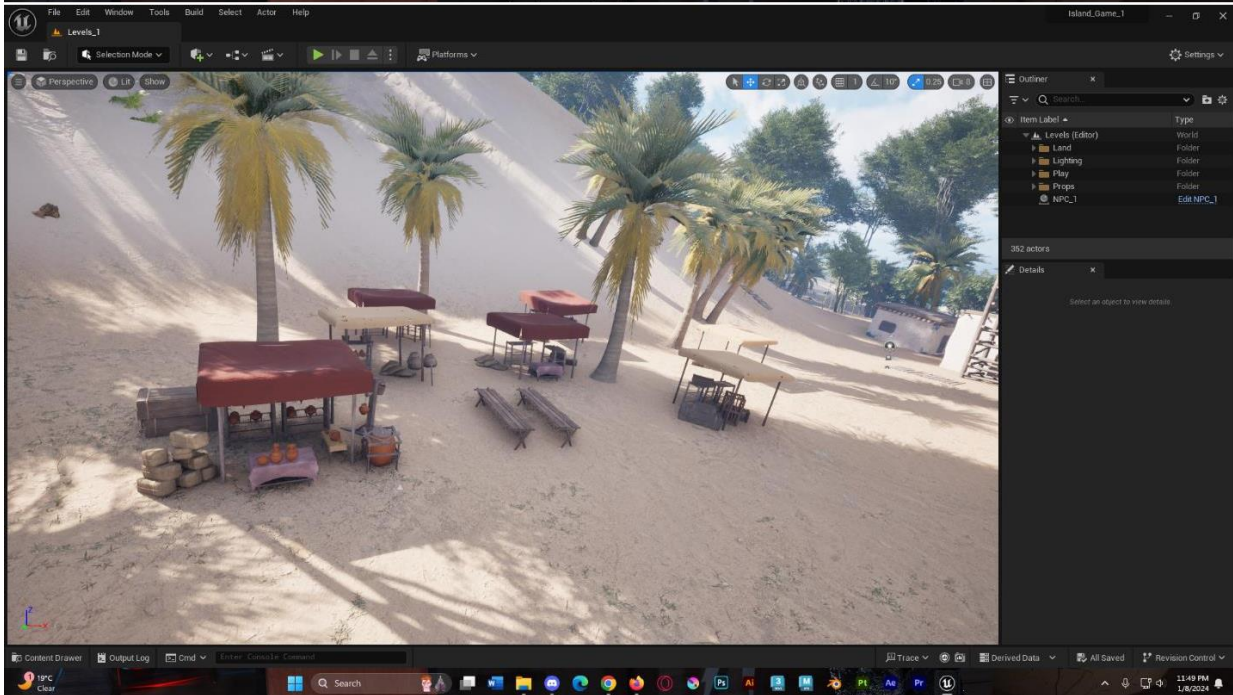
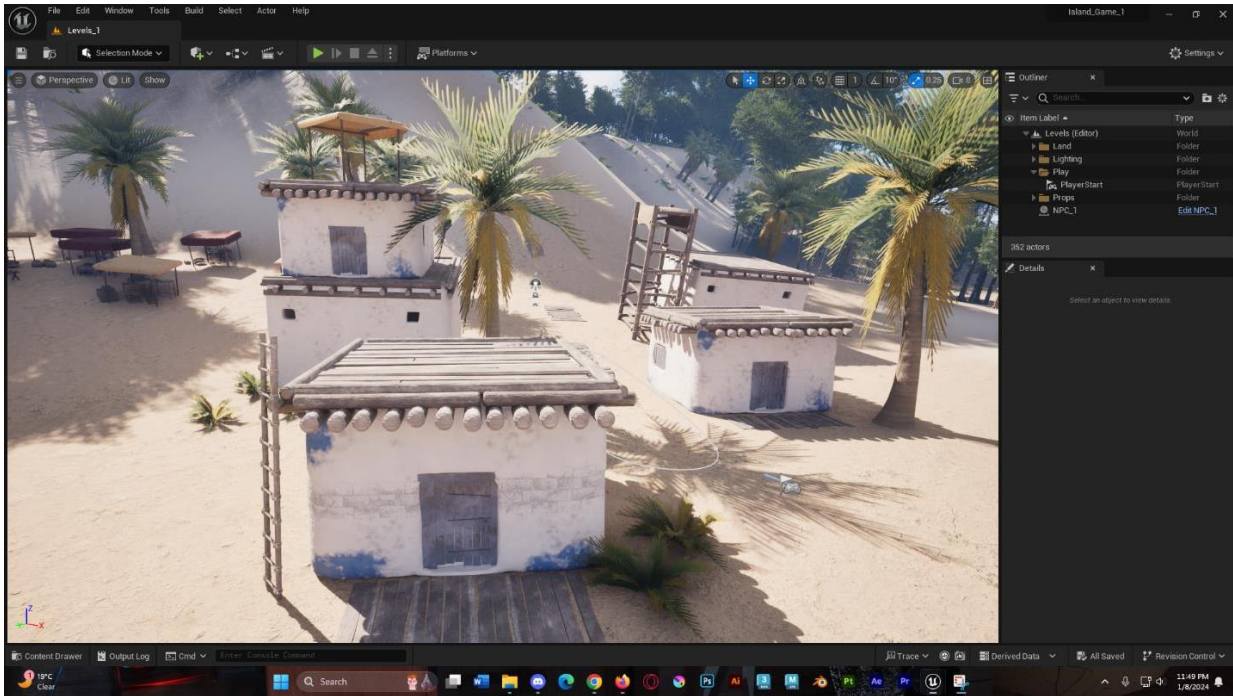


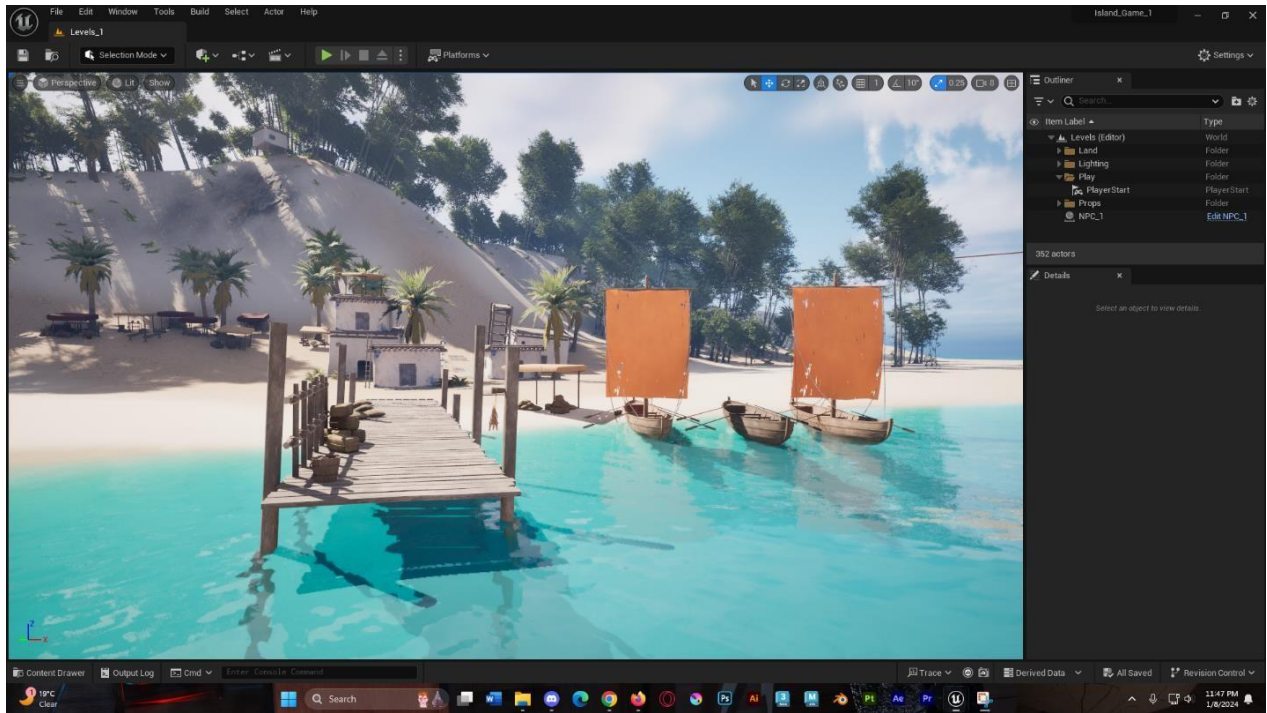




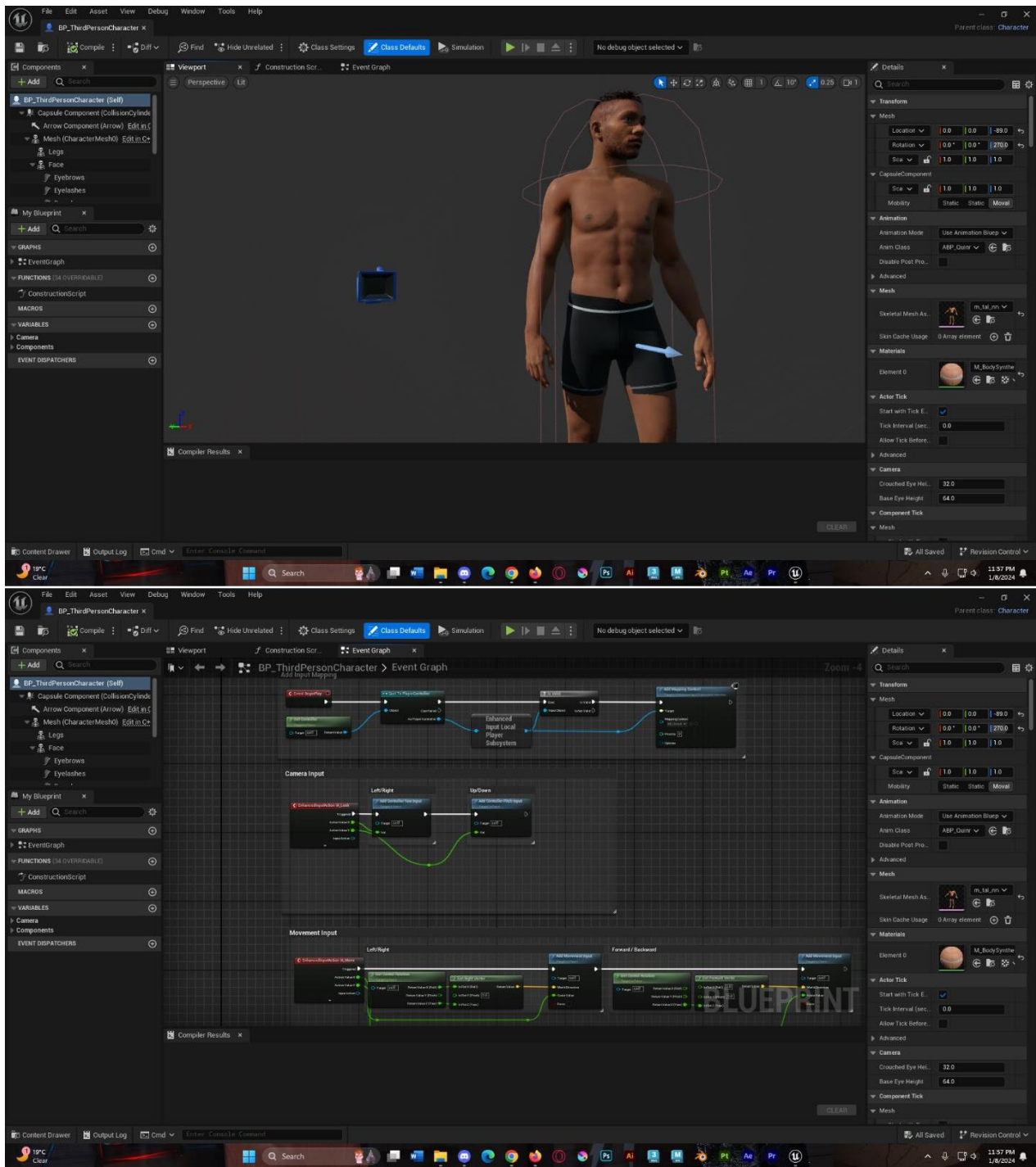
6.2 Locations:







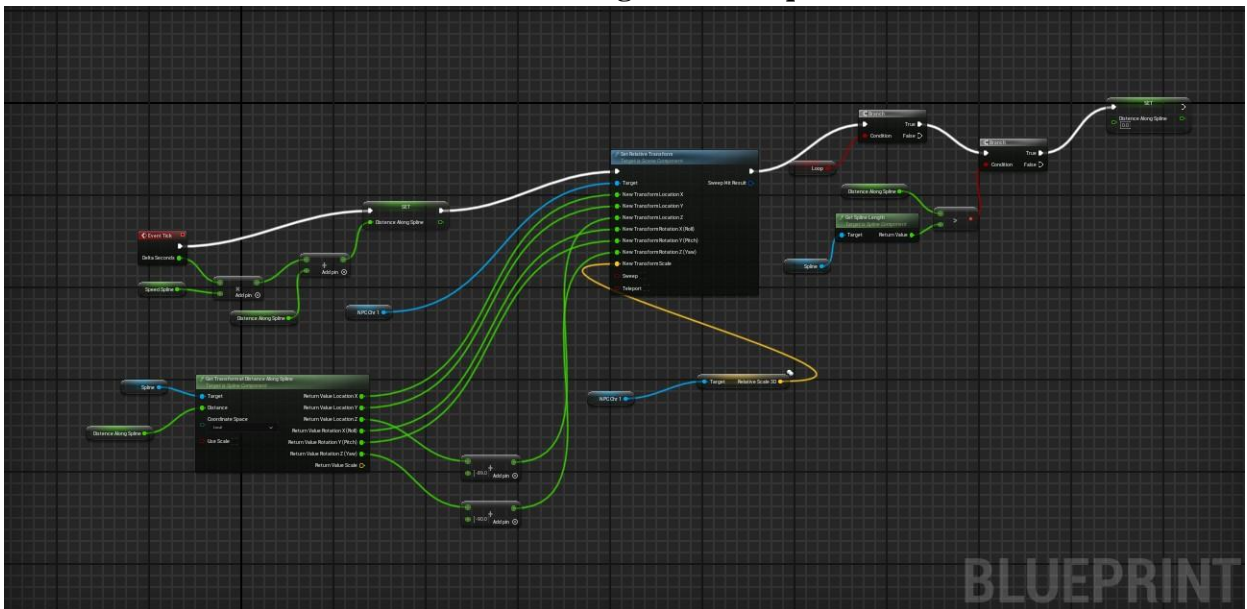
6.2 Character:

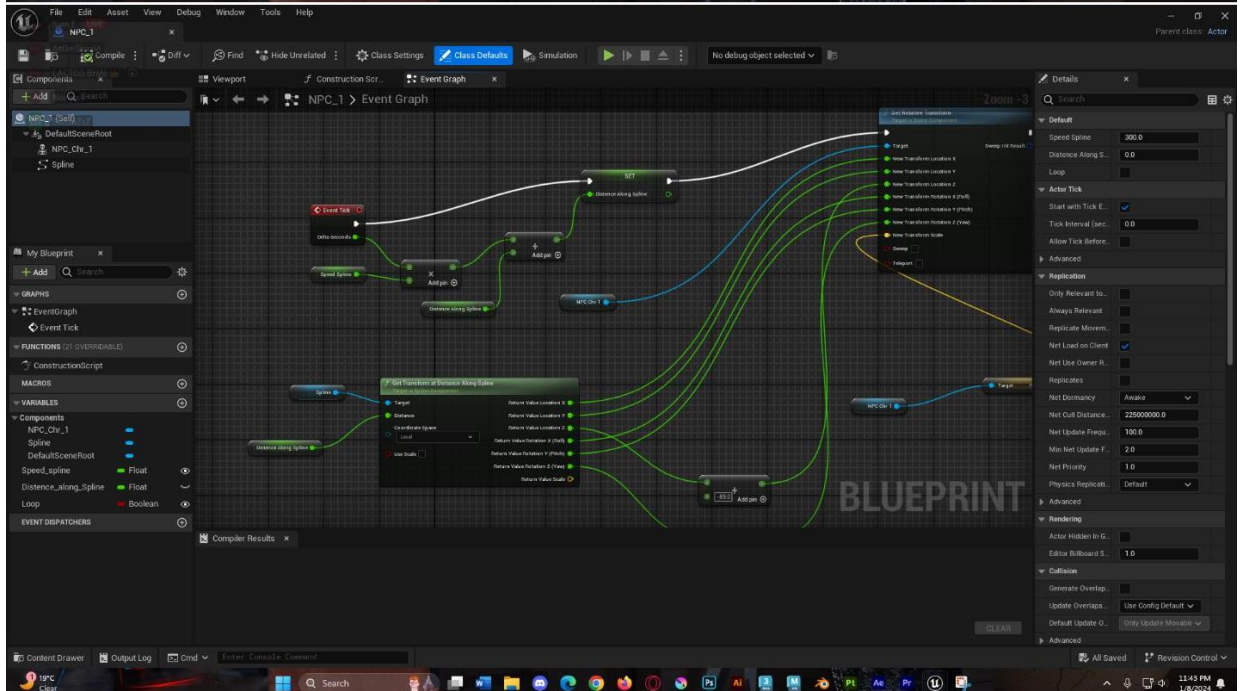
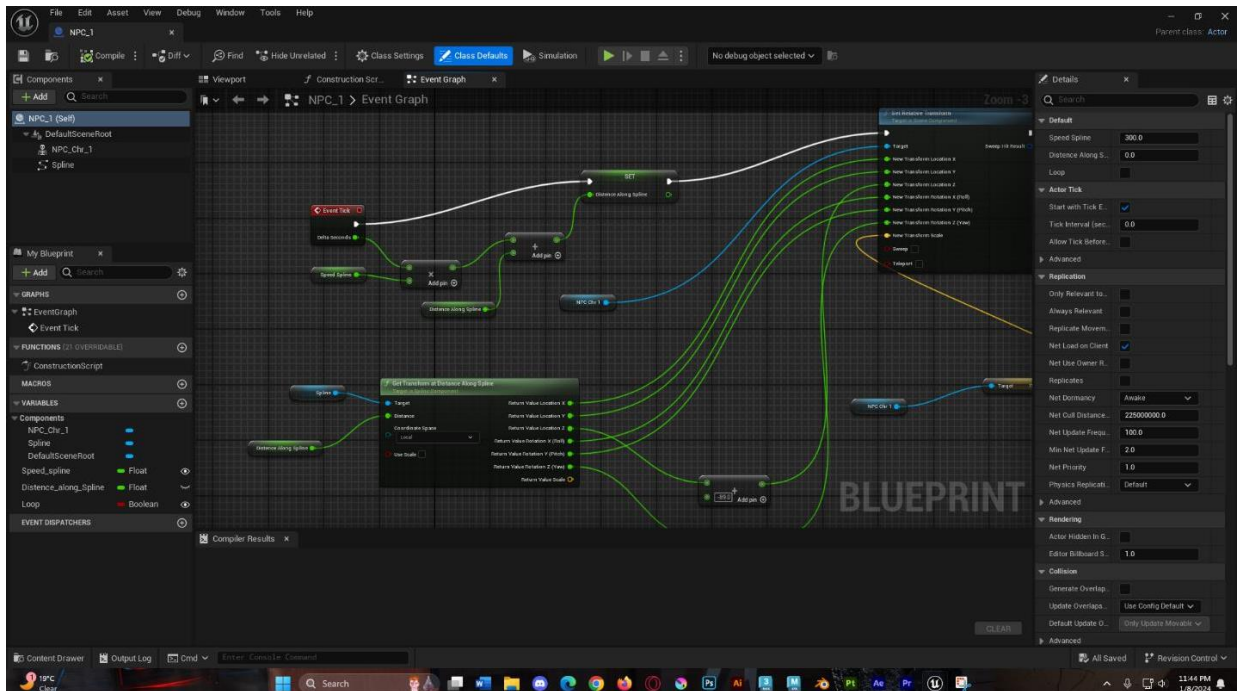


6.3 NPCs



N.B Unreal Engine Mannequin





Chapter 7: Conclusion

We've strived to create not just a game but a living, breathing universe where players can escape into a realm of endless possibilities. "Lactos World" is a testament to what can be achieved when passion, innovation, and a commitment to excellence come together.

7.1 Future Roadmap:

- ✓ **Expansions and Additional Content:** Lactos World envisions a future filled with exciting expansions and additional content, introducing new lands, quests, and challenges for players to conquer. The roadmap includes a commitment to delivering fresh and engaging experiences for the growing player base.
- ✓ **Technological Advancements:** Embracing advancements in gaming technology, Lactos World looks forward to incorporating new features, optimizing performance, and exploring emerging platforms to ensure a cutting-edge gaming experience.
- ✓ **Community Engagement:** The game's success is intrinsically tied to the vibrant community of players. Ongoing community engagement initiatives, such as events, forums, and social media interactions, foster a sense of belonging and shared enthusiasm among players.

Reference:

1. Game: Games like "The Elder Scrolls", "Cyber punk 2077", series (e.g., Skyrim) and "The Legend of Zelda: Breath of the Wild" offer expansive open worlds with rich narratives and intricate gameplay. And also Watch Dogs, Call Of duty series By Activision.

2. Unreal Engine Showcase: Explore projects and games that showcase the capabilities of the Unreal Engine.

<https://docs.unrealengine.com/4.26/en-US/Resources/SampleGames/>

3. Unreal Engine's scripting language (C++ and Blueprints). <https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ClassCreation/CodeAndBlueprints/>