



Tribhuvan University
Faculty of Humanities and Social Sciences

Game Development
with
Perlin Noise Algorithm and A* Algorithm
A PROJECT PROPOSAL

Submitted to
Department of Computer Application
Asian College of Higher Studies

In partial fulfillment of the requirements for the Bachelors in Computer Application

Submitted by
Ashesh Shakya
Suja Maharjan
2023/08/11

Under the Supervision of
Mr. Bimal Adhikari



Tribhuvan University
Faculty of Humanities and Social Sciences
Asian College of Higher Studies

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by Ashesh Shakya and Suja Maharjan entitled “**Game Development with Perlin Noise Algorithm and A* Algorithm**” in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

.....

Mr. Bimal Adhikari

Lecturer

Asian College of Higher Studies



Tribhuvan University
Faculty of Humanities and Social Sciences
Asian College of Higher Studies

LETTER OF APPROVAL

This is to certify that this project prepared by Ashesh Shakya and Suja Maharjan entitled “**Game Development with Perlin Noise Algorithm and A* Algorithm**” in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

<p>.....</p> <p>SIGNATURE of Supervisor</p> <p>Mr. Bimal Adhikari</p> <p>Lecturer</p> <p>Asian College of Higher Studies</p> <p>Lalitpur</p>	<p>-----</p> <p>SIGNATURE of HOD/ Coordinator</p> <p>Mr. Pranaya Nakarmi</p> <p>Coordinator</p> <p>Asian College of Higher Studies</p> <p>Lalitpur</p>
<p>-----</p> <p>SIGNATURE of Internal Examiner</p> <p>Internal Examiner</p>	<p>-----</p> <p>SIGNATURE of External Examiner</p> <p>External Examiner</p>

ABSTRACT

In today's gaming market, video games have emerged as a highly popular form of entertainment, captivating a wide audience. Engaging environments are crucial for the success of many games. Procedural generation, a technique enabling the creation of dynamic environments and diverse content, holds great promise for game design, art, and production.

The objective of this report is to explore a game development project centered around a procedurally generated world, allowing players to freely navigate and explore an immersive game environment. The project focuses on the development of the game world using seed-based generation, where the seed can be either random or input by the player. This approach ensures that with each new game, the leading character will encounter unique landscapes throughout their journey. Two key algorithms were implemented for this game development project: the Perlin Noise algorithm for generating the game world and the A* algorithm for efficient pathfinding between different towns.

Through the exploration of procedural generation and the implementation of these algorithms, this report aims to demonstrate the potential and capabilities of creating captivating and ever-changing game environments.

KEYWORDS: *Procedural generation, Point of Interest (POI), Game Development, Perlin Noise algorithm, A* algorithm*

ACKNOWLEDGEMENT

We could like to express our special thanks of gratitude to our Supervisor (Mr. Bimal Adhikari) as well as our coordinator (Mr. Pranaya Nakarmi) who gave us a golden opportunity to do this wonderful project on the topic (Game Development with Perlin Noise Algorithm and A* Algorithm), which also helped us in doing a lot of research and we came to know about so many new things we really thankful to them. Secondly, we would also like to thank our parents and friends and or mentor who helped us a lot in finalizing this project within the limited time frame.

Your Sincerely

Ashesh Shakya

Suja Maharjan

TABLE OF CONTENTS

ABSTRACT.....	iii
ACKNOWLEDGEMENT.....	iv
LIST OF ABBREVIATIONS	vii
LIST OF FIGURE	viii
LIST OF TABLES	ix
CHAPTER 1: INTRODUCTION.....	1
1.1. Introduction.....	1
1.2. Problem Statement	2
1.3. Objectives	2
1.4. Scope and Limitation	2
1.4.1. Scope.....	2
1.4.2. Limitation.....	2
1.5. Development Methodology	2
1.6. Report Organization.....	3
CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW.....	4
2.1. Background study	4
2.2. Literature Review.....	4
CHAPTER 3: SYSTEM ANALYSIS AND DESIGN	6
3.1. System Analysis.....	6
3.1.1. Requirement Analysis	6
i. Functional Requirements	6
ii. Non Functional Requirements	7
3.1.2. Feasibility Analysis.....	7
i. Technical Analysis.....	7
ii. Operational Analysis.....	7
iii. Economic Analysis.....	7
iv. Schedule Analysis	8
3.1.3. Object Modelling using Class and Object Diagrams.....	9
3.1.4. Dynamic Modelling using State and Sequence Diagrams	11
3.1.5. Process Modelling using Activity Diagrams	12
3.2. System Design	24

3.2.1.	Refinement of Class, Object, State, Sequence and Activity diagrams	24
3.2.2.	Deployment Diagrams	25
3.3.	Algorithm Details (if any).....	26
CHAPTER 4: IMPLEMENTATION AND TESTING		29
4.1.	Implementation	29
4.1.1.	Tools Used	29
4.1.2.	Implementation Details of Modules.....	30
4.2.	Testing.....	30
4.2.1	Test Cases for Unit Testing.....	30
4.2.2	Test Case for System Testing	32
CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATIONS		33
5.1.	Conclusion	33
5.2.	Future Recommendations	33
APPENDICES		
REFERENCES		

LIST OF ABBREVIATIONS

ACSII	American Standard Code for Information Interchange
BFS	Breadth First Search
CSS	Cascading Style Sheets
DFS	Depth First Search
GPU	Graphical Processing Unit
HTML	HyperText Markup Language
JRPGS	Japanese Role Play Games
NPCS	Non Playable Characters
OS	Operating System
RAM	Random Access Memory
RPG	Role Playing Game
UML	Unified Modeling Language

LIST OF FIGURE

Figure 1.1: Iterative Model	3
Figure 3. 1: Use Case Diagram of Game	6
Figure 3. 2: Gantt Chart	8
Figure 3. 3: Class Diagram of World Generation	9
Figure 3. 4: Class Diagram of Game	10
Figure 3. 5: Sequence Diagram of Input Handling	11
Figure 3. 6: World Generation Activity Diagram	12
Figure 3. 7: Towns Generation Activity Diagram	13
Figure 3. 8: Player Movement Activity Diagram	14
Figure 3. 9: Collision Detection Activity Diagram.....	15
Figure 3. 10: Spiral Graph Traversal Activity Diagram	16
Figure 3. 11: NPCs Generation Activity Diagram.....	17
Figure 3. 12: Special NPCs Generation Activity Diagram	18
Figure 3. 13: Player Movement in world.....	19
Figure 3. 14: Running Command/ Cheats.....	20
Figure 3. 15: Add Item in Inventory	21
Figure 3. 16: Remove Item in Inventory	22
Figure 3. 17: Waypoint Interaction process	23
Figure 3. 18: Refinement Class Diagram of World Generation	24
Figure 3. 19: Deployment Diagram of Game	25

LIST OF TABLES

Table 4. 1: Test Case for New Player Placing	30
Table 4. 2: Test Case for Player Movement	31
Table 4. 3: Test Case for Town Visit.....	31
Table 4. 4: Test Case for Teleportation	31

CHAPTER 1: INTRODUCTION

1.1. Introduction

Games have been an integral part of human culture for centuries, and today they have become a multi-billion dollar industry. The gaming industry has evolved rapidly over the past few decades, with technological advancements leading to the development of games.

".co" will be a 2D tile based strategy game in which the main game play loop will be exploring a vast procedurally generated world and interacting with other Non-playable Characters (NPCs) to aid you in your journey. This game will be made with direct inspiration from games like *"Fire Emblem"* and *"Octopath Traveler"*.

Fire Emblem is a tactical RPG where you move characters around a grid-like battlefield, taking turns to position them strategically and engage in combat with enemy units. In this type of game, you move characters around a grid-like battlefield and strategically position them to attack enemy units or defend your own. The movement system in this game is turn based, which means that you take turns moving your characters and making decisions before the enemy gets a chance to do the same.

Octopath Traveler, on the other hand, is a game with a turn-based combat system that involves choosing different attacks and abilities for each character in your party during each turn. In this game, you can target different parts of your enemy's body to weaken them and gain an advantage in combat.

The procedural generation of the game world means that each playthrough would be unique, as the game would generate new maps and environments every time you start a new game. This would keep things fresh and exciting, as you explore new areas and encounter different enemies with each playthrough.

So, imagine a game that combines the movement system of Fire Emblem with the combat system of Octopath Traveler. You would move your characters around a tactical battlefield and position them strategically to engage in turn-based combat. During combat, you would be able to choose from a variety of different attacks and abilities for each character, much like in Octopath Traveler but in a procedurally generated world.

1.2. Problem Statement

There have been many JRPGs (Japanese Role Playing Games) and games with procedurally generated maps but a game with both of the concept placed together have not been found. So this game will focus on combining the best aspect of both worlds to create a new game.

1.3. Objectives

- i To use Perlin noise algorithm for procedural world generation.
- ii To creates path between different Point of Interests (POIs) using A* path solving algorithm.
- iii To only use JavaScript and Bootstrap component to create a game.

1.4. Scope and Limitation

1.4.1. Scope

- As this game is browser based, so it is easily distributable.
- The scope of this project encompasses the development of a game prototype that showcases the potential of procedural generation.
- The game will feature a character that can navigate the generated world, collect coins, and encounter various landscapes based on a seed value.

1.4.2. Limitation

- In procedurally generated worlds we cannot achieve same level of detail as hand drawn worlds.
- Because JavaScript is used, files can only be read but not written on.

1.5. Development Methodology

Game development is a complicated process that requires us to work parallel, so we will be using the Iterative methodology which is an approach for developing products using short iterations.

The following phases were selected for the completion of the project:

- i. Initial Planning
- ii. Planning and Requirements
- iii. Design and Implementation
- iv. Development
- v. Testing

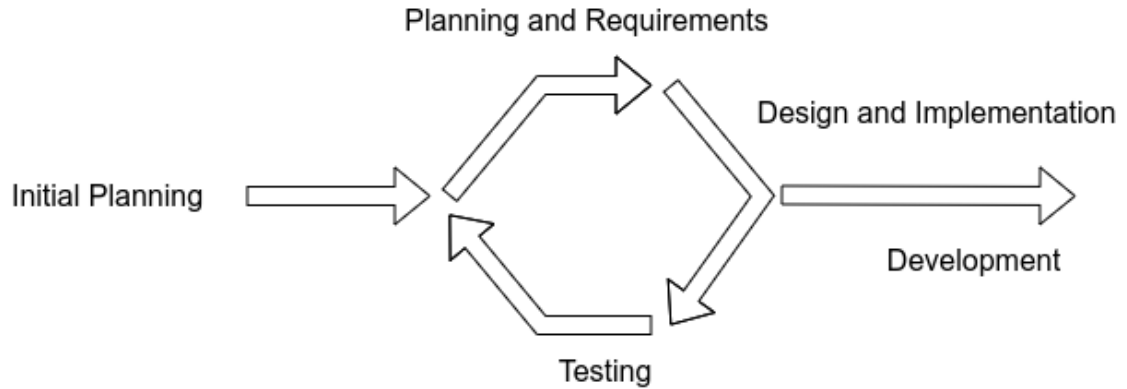


Figure 1.1: Iterative Model

1.6. Report Organization

Chapter 1: Introduction

This chapter serves as an introduction to the project, presenting the problem statement, project objectives, as well as the project's scope and limitations. It aims to provide readers with a comprehensive overview of the project.

Chapter 2: Background study and Literature review

This chapter enclose about background study and literature review, focusing on a similar project and the relevant concepts associated with the current project. Its aim is to provide understanding of the project by examining previous research.

Chapter 3: System Analysis and Design

This chapter encompasses the workings of the project including feasibility analysis, functional and non-functional analysis along with the schema and architectural design and the data and process modeling diagrams.

Chapter 4: Implementation and Testing

This chapter encompasses the testing and implementation phase of the project, which involves the design of test cases to ensure the system and its components function as intended during the development process. It also includes an overview of the tools utilized in the construction of the project.

Chapter 5: Conclusion and Future Recommendations

It contains the conclusion of the entire project. In addition to these things the future recommendations of the project are also listed in the end of chapter 5.

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1. Background study

While software developers have applied procedural generation techniques for years, few products have employed this approach extensively. Procedurally generated elements have appeared in earlier video games: The Elder Scrolls II: Daggerfall takes place in a mostly procedurally generated world, giving a world roughly two thirds the actual size of the British Isles. Soldier of Fortune from Raven Software uses simple routines to detail enemy models, while its sequel featured a randomly-generated level mode. Avalanche Studios employed procedural generation to create a large and varied group of detailed tropical islands for Just Cause. No Man's Sky, a game developed by games studio Hello Games, is all based upon procedurally generated elements.

Prior to graphically oriented video games, roguelike games, a genre directly inspired by Dungeons & Dragons adopted for solitaire play, heavily utilized procedural generation in the same manner that tabletop systems had done. Such early games include Beneath Apple Manor (1978) and the genre's namesake, Rogue (1980). The procedural generation system in roguelikes would create dungeons in ASCII- or regular tile-based systems and define rooms, hallways, monsters, and treasure to challenge the player. Roguelikes, and games based on the roguelike concepts, allow the development of complex gameplay without having to spend excessive time in creating a game's world. [1]

2.2. Literature Review

The first computer games appeared in the sixties and seventies of the 20th centuries, who can imagine that these interactive classic games like chess or pong will become one of the biggest industries in the world? This industry generates more money than the film industry.

With the growth of the mobile devices and more powerful computers and consoles the consumers are interested to pay for subscriptions services or gaming services, although this is not quite popular in mobile devices is becoming more popular for the companies

to include in-app purchases to their product while the game is free for download. Also, many consumers want their consoles for more than just gaming, for this reason the companies are maximising other aspects like video or audio on demand . For all this reasons is possible to say that video games have become part of the culture of the people hence the benefits from them are only growing more and more every year.

Game development can be divided in three phases: preproduction, production and post-production. The first phase consists to evaluate all the ideas available to until one stands out and then the game proposal is created. After that different prototypes are created and played internally to get feedback in order to help the designers and managers to understand which one is the best and fits well in a commercial perspective. The production phase starts when all the team knows what the want to build and how they plan to create it, this means that they can start to create real assets that will be used in the game. Finally, post-production is the final phase of the game development cycle and this involves the marketing strategies and final distribution of the product. [2]

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

3.1. System Analysis

3.1.1. Requirement Analysis

i. Functional Requirements

➤ User

- Can create new games
- Can save and load the game
- Will be able to move from one town to another
- Will be able to import and export game data
- Can quest
- Can get the inventory
- Can execute command

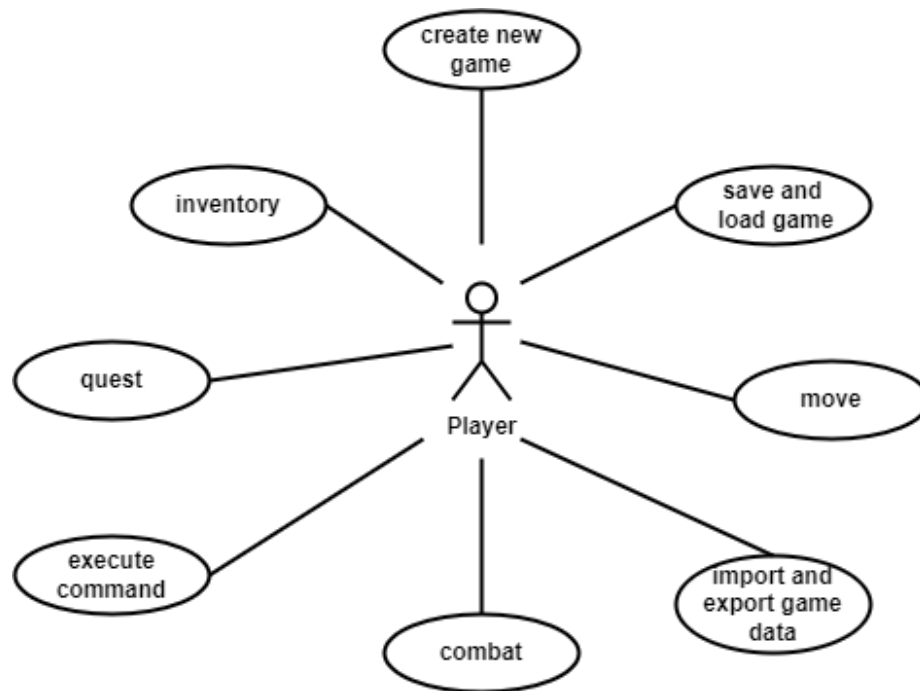


Figure 3. 1: Use Case Diagram of Game

ii. Non Functional Requirements

The requirements required for the smooth running for the new system was analyzed and the following minimum necessities were considered necessary:

- Requires a 64-bit processor and operating system
- OS: Windows 10 or above
- Processor: Dual Core 2 GHz
- Memory: 2 GB RAM
- Graphics: Intel HD 4600
- Storage: 500 MB available space

3.1.2. Feasibility Analysis

This chapter describes all the feasibilities that comes as question to both the developers and other users during the development of software. The chapter contains technical feasibility, operational feasibility and economic feasibility.

i. Technical Analysis

The tools and technology that were used in the making of this game are:

Game Engine: Javascript

Code Editor: Visual Studio Code

Programming Language: Javascript

These mentioned above technologies are completely free for students. There were no other additional tools required to make this game. The simplicity of the project along with the facts mentioned above proves that this software is technically feasible.

ii. Operational Analysis

The game can be easily played by the users who frequently play such games but also it can be easily by learned by new player. The game is targeted towards the people who like JRPGs (Japanese Role Playing Games).

iii. Economic Analysis

The resources that are required for this projects are:

- Development machine: Any regular laptop/PC with Minimum Ram of 2GB and a decent GPU can be used for the development of this game.

- Technical tools and software: The tools needed to develop this software are available to developers at no charge.
- Game Developers and Graphic Designers.

For ingame sprites, sound and other assets, websites like craftpix.net will be used to download free sprites and sound effects. Piskel app will be used to create art assets.

iv. Schedule Analysis

To develop any game from scratch without using game engine requires more time. So it may be hard to complete development within 3 month. Therefore we had schedule time as below chart:

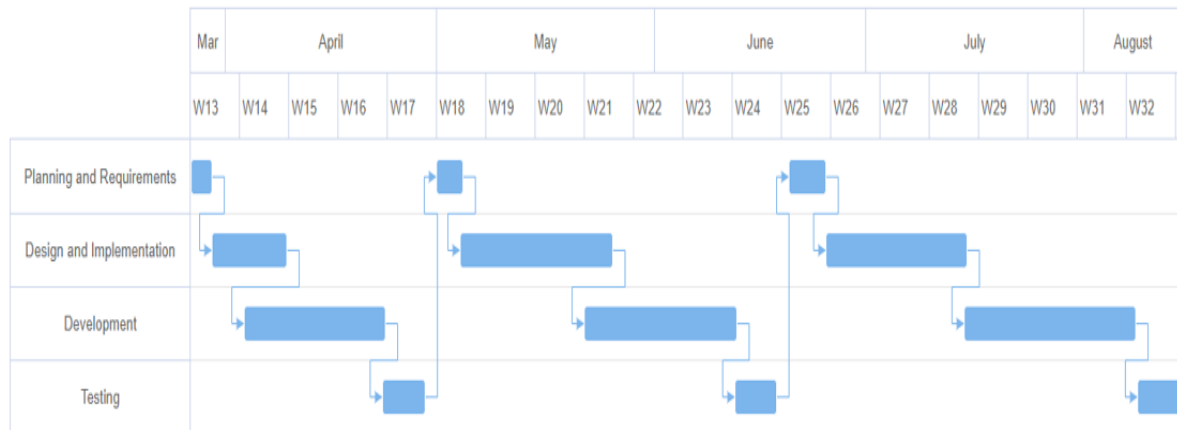


Figure 3. 2: Gantt Chart

3.1.3. Object Modelling using Class and Object Diagrams

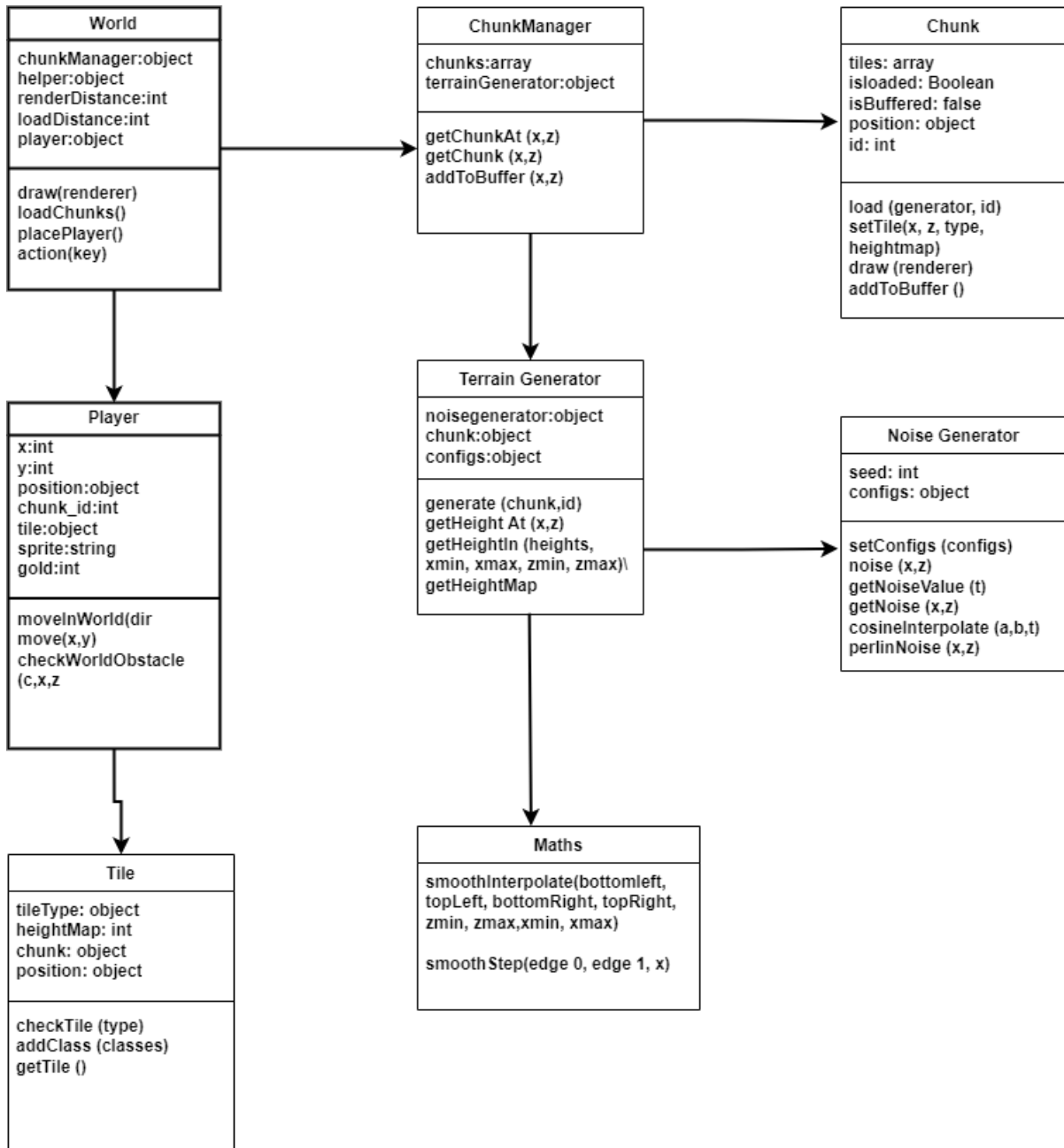


Figure 3. 3: Class Diagram of World Generation

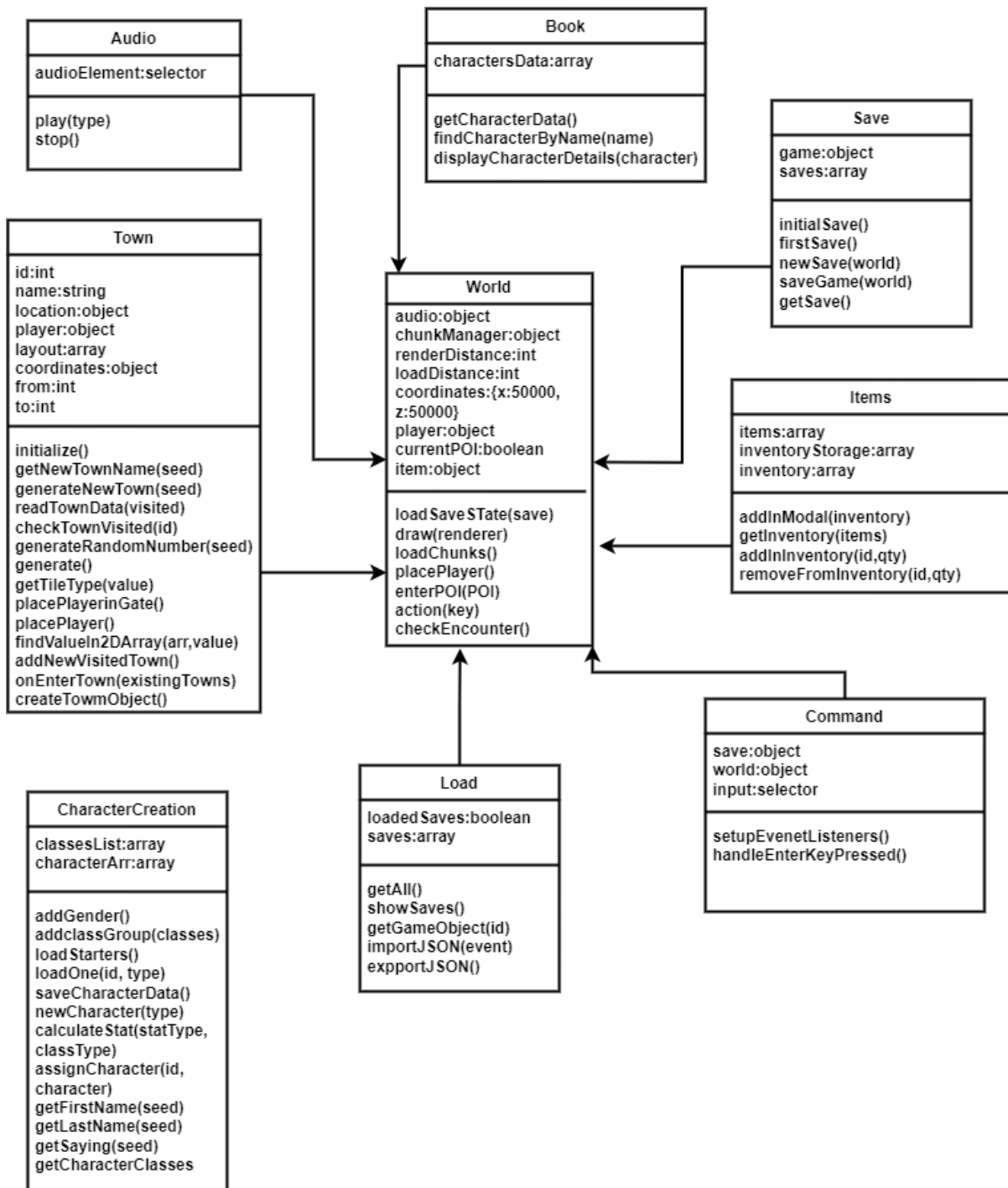


Figure 3. 4: Class Diagram of Game

3.1.4. Dynamic Modelling using State and Sequence Diagrams

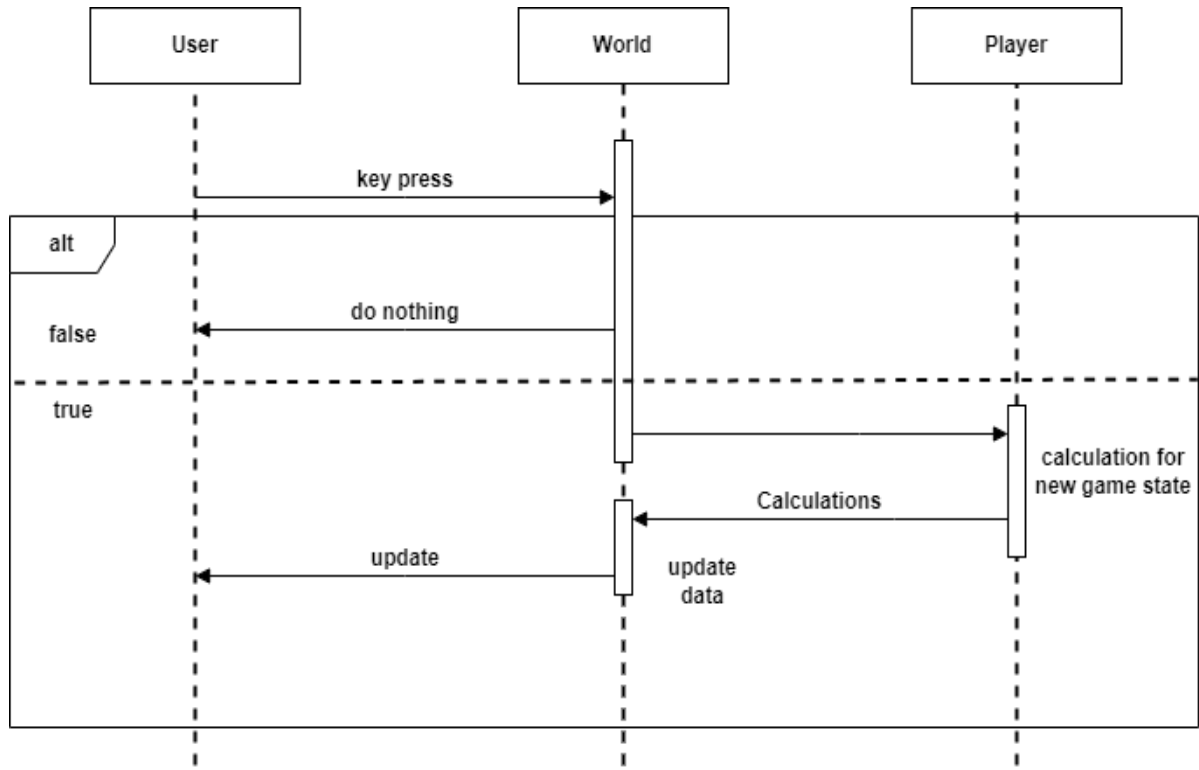


Figure 3. 5: Sequence Diagram of Input Handling

In this sequence diagram, process of handling the input is described. Firstly, when user press key, world will receive the key and with respective to that key world will check if action has been handled or not. If action is being handled then world will send action to player else do nothing. After player get action, player will makes calculation as per the action line movement and then new calculation is send back to world and world will update the data and show updated data to the user.

3.1.5. Process Modelling using Activity Diagrams

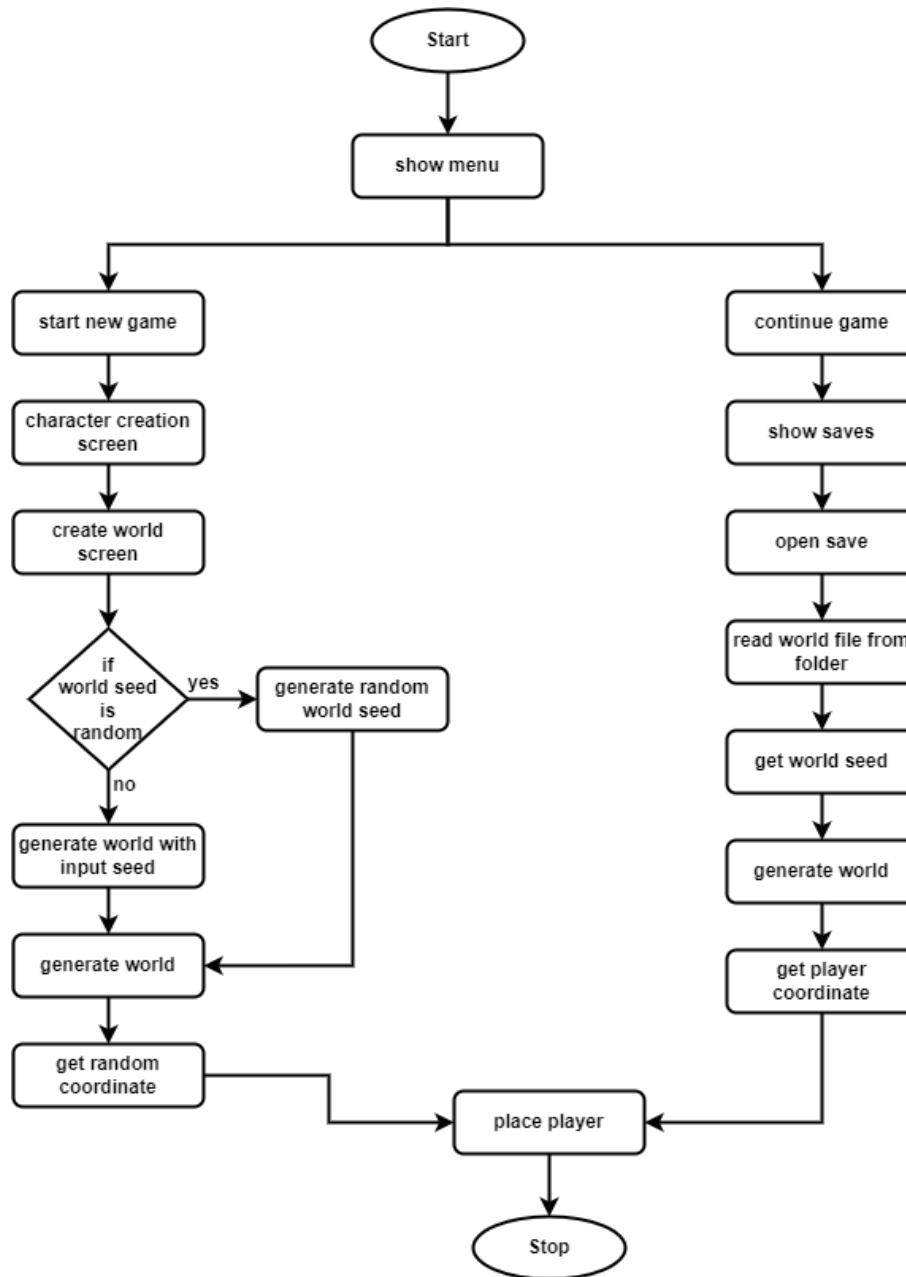


Figure 3. 6: World Generation Activity Diagram

The world generation process start with displaying menu where new game and continue button will be shown. When new button is clicked, character creation page will be open. In character creation page, User will input the required data and after all data are input then move to the seed generation page. Here, user will be given two option i.e. to generate world with random seed or to generate world with input seed and get random coordinate (x, y) to place player.

When Continue button is clicked, the saved game and open the game as we had saved. Then read world file from the folder, from where we can get the world seed and generate the world and get the player coordinate as they were before they save the game and place the player.

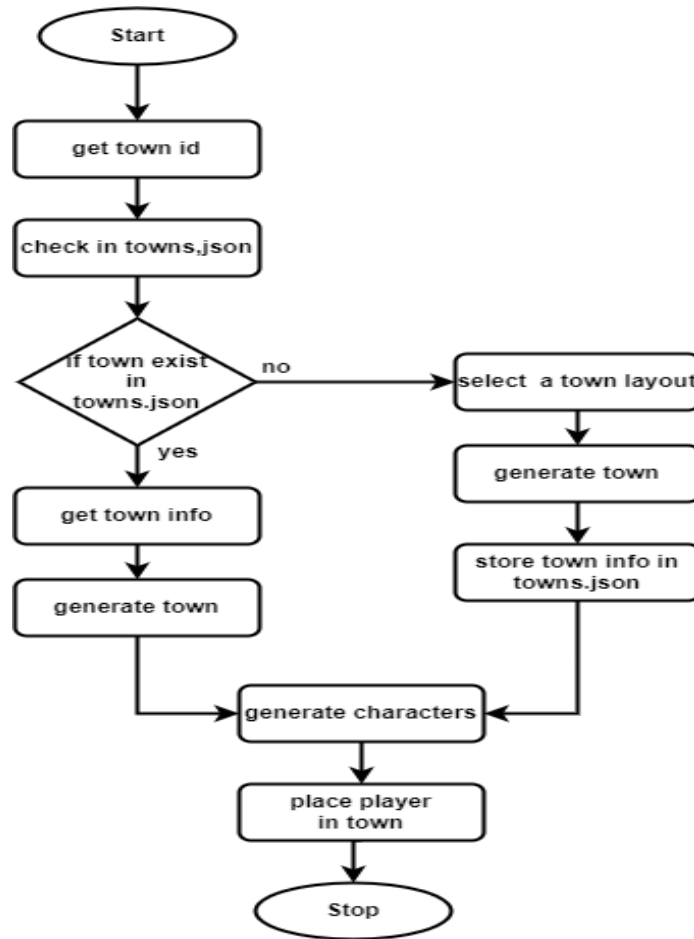


Figure 3. 7: Towns Generation Activity Diagram

Firstly, town id is get and Secondly, town is checked if it exist on town.json file or not. If town exist on json file, the information about town is retrieve and according to that information town is generated else the town layout is selected and town is generated with storing information about town in town.json file. At last character is generated and the player is placed in the town.

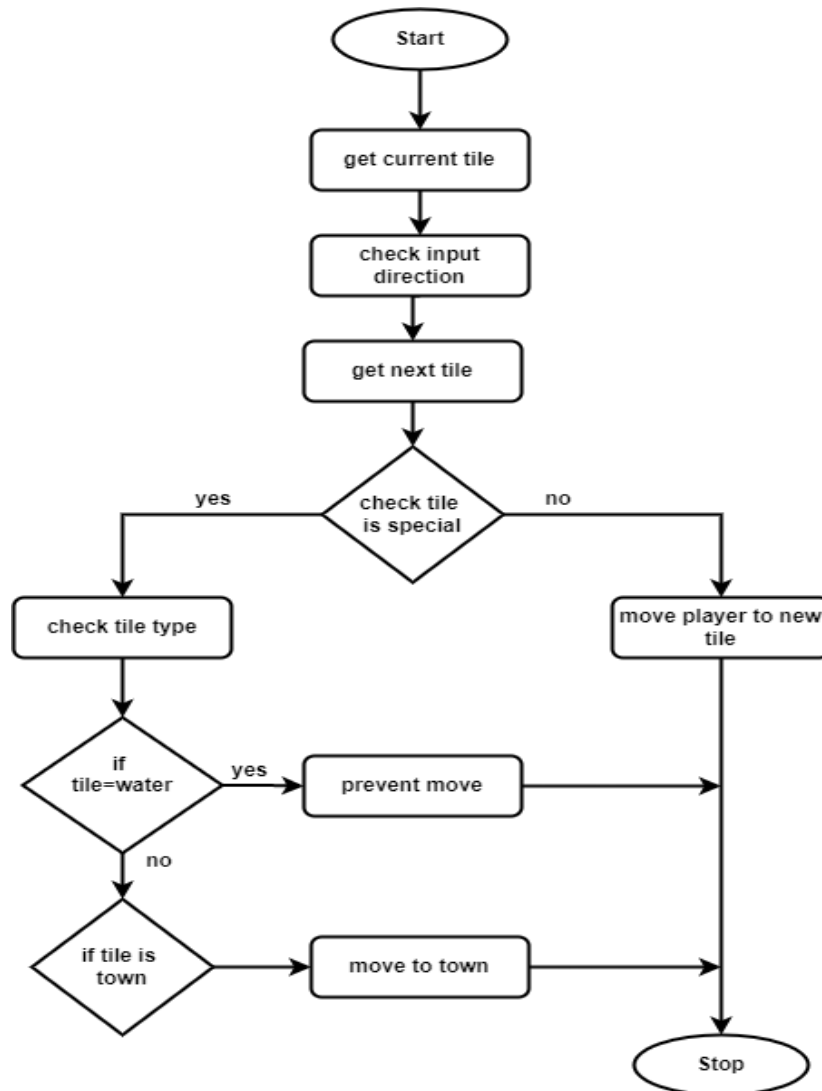


Figure 3. 8: Player Movement Activity Diagram

Figure 3.8 illustrate the player process movement where the first method is to get current tile id where player is placed and when player want to move to another tile, the direction is check and get the next tile id. After the new tile id is get, it is check to find the tile type i.e. whether it is water or town. If the next tile is water then the movement of player is prevent and if the next tile is town then player will move to that town.

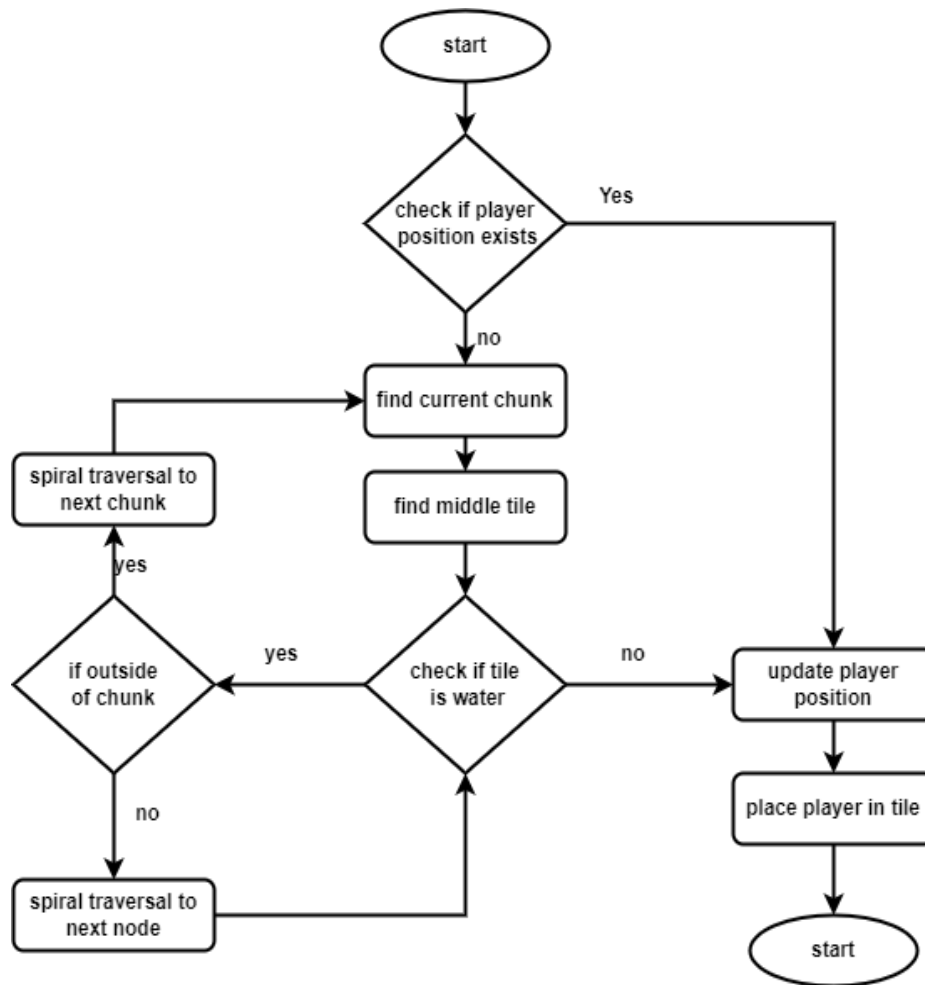


Figure 3. 9: Collision Detection Activity Diagram

Figure 3.9 describes an activity diagram of collision detection. It start with checking the player position if exist or not. If player exist then update the player position else find the current chunk i.e. center chunk and then find the middle tile. When middle tile is found, the tile will be check whether it is water or not. If tile is not water then update player position else again it is check whether it is outside of chunk or not else . If it is true then spiral traversal to next chunk and then return back to the step i.e. finding current chunk else spiral traversal to next tile and again check if the tile is water and continue with the same process as above.

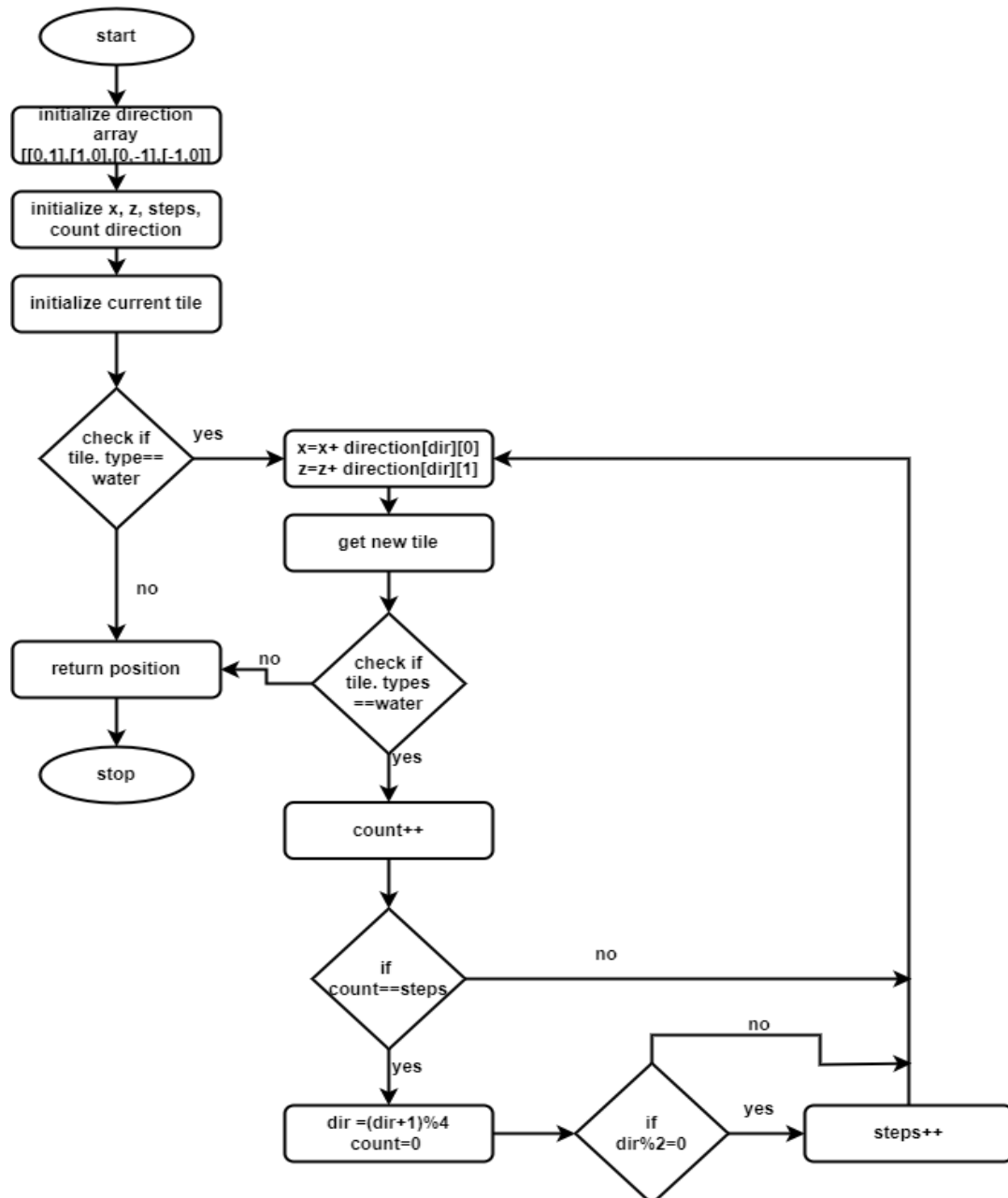
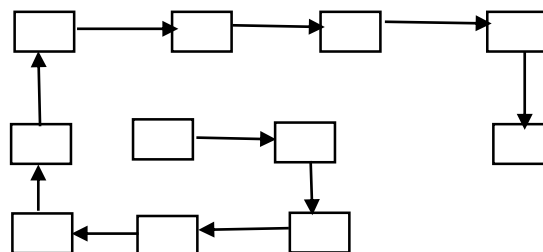


Figure 3. 10: Spiral Graph Traversal Activity Diagram

Initially when new game start, if player starting tile is water then move in a spiral pattern by checking if tiles are placable. This spiral pattern looks in a way of



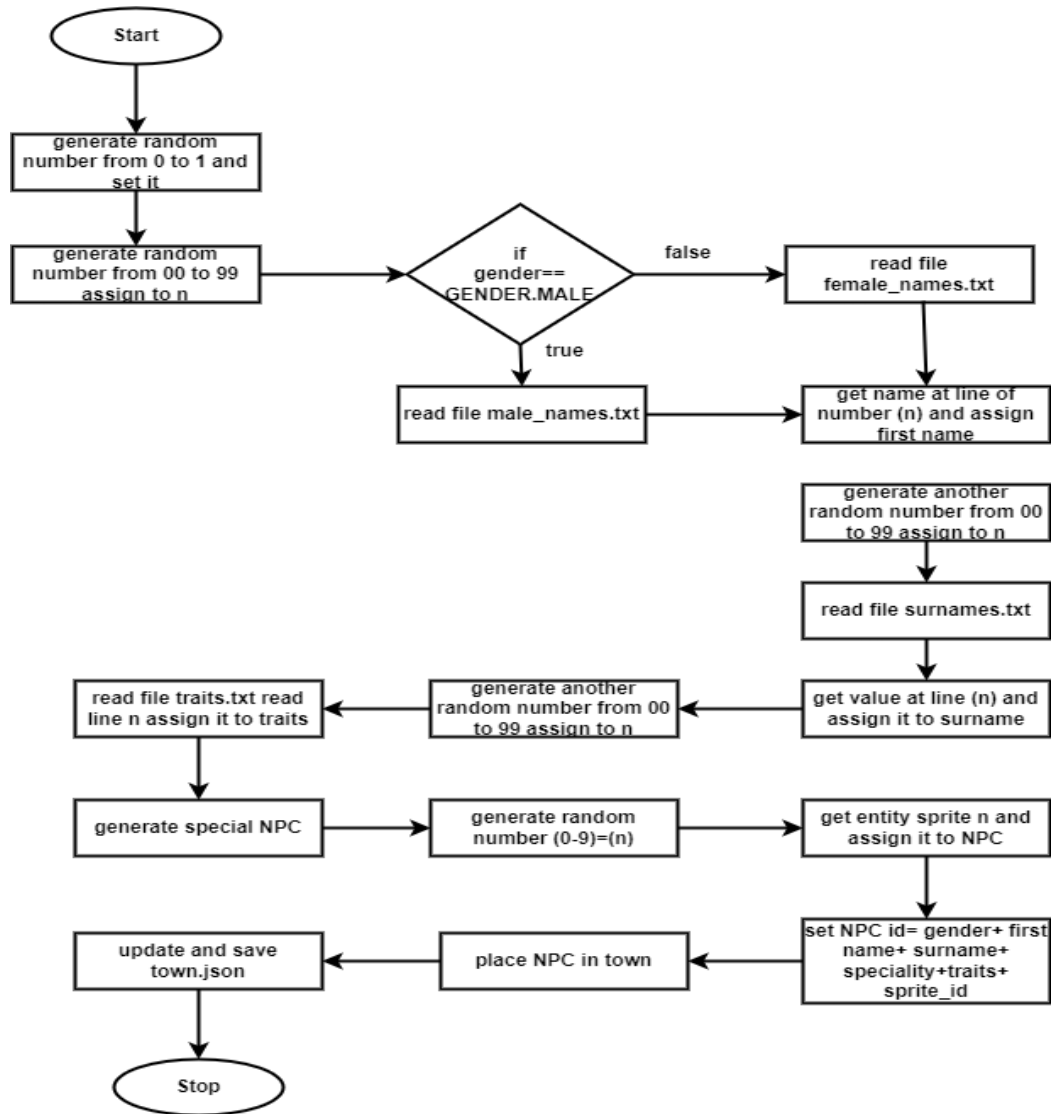


Figure 3. 11: NPCs Generation Activity Diagram

For generating NPCs in towns which are generated for the first time we try to provide each NPC with a unique id which uses multiple random numbers to generate a 10 digit number in the format of id = 1-12-12-12-12-1 (gender – first name – last name – specialty – trait – sprite_id) For this, first a random number is generated between 0 and 1 to assign in gender. Another 2 digit random number is generated which is used along side gender value to read first name files to assign the first name parameter, similar things are done for traits and specializations and surnames but surnames are not affected by gender. At last a sprite id is randomly selected for the new character and character id is set and placed in town and the town data is updated and saved.

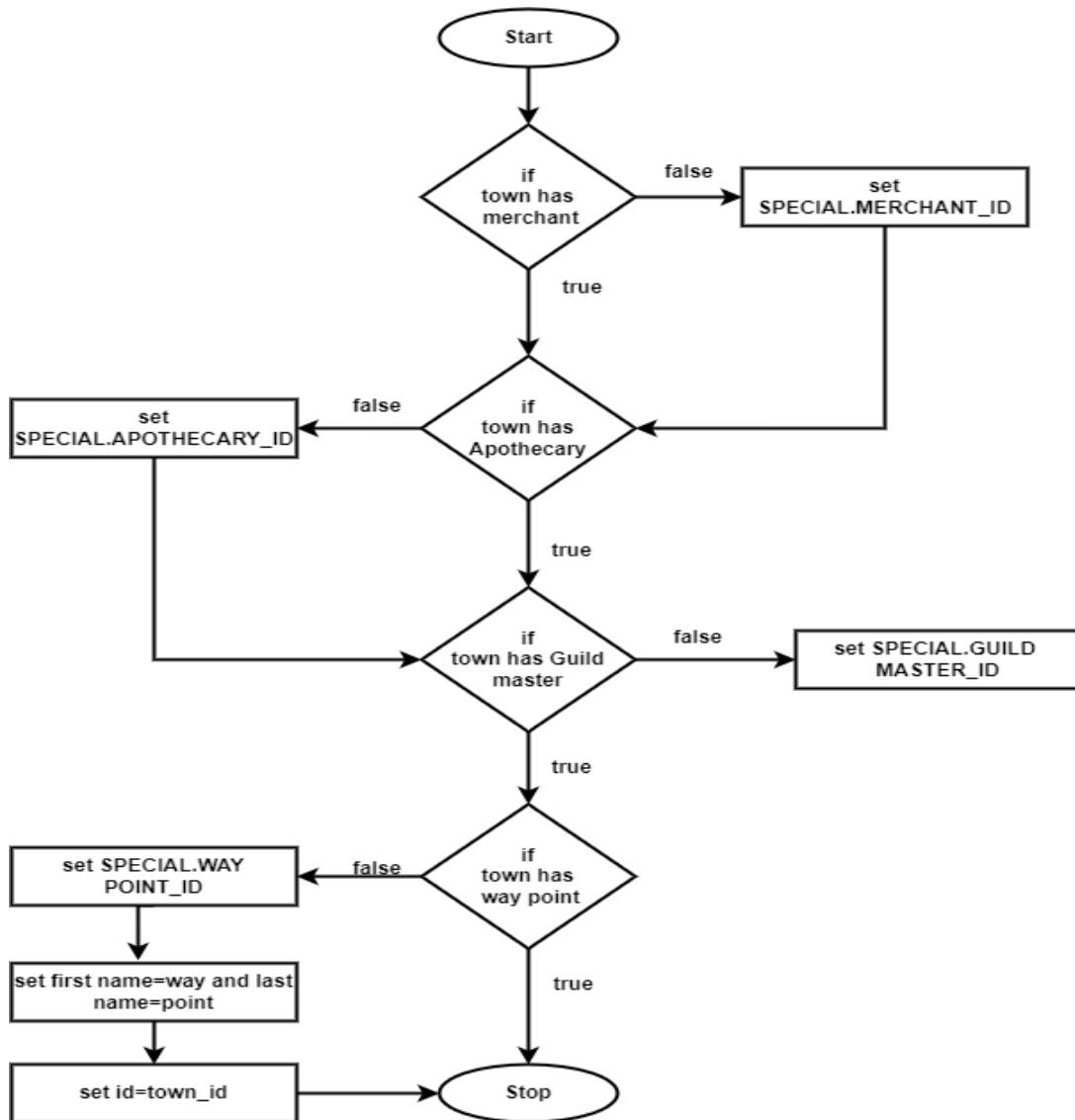
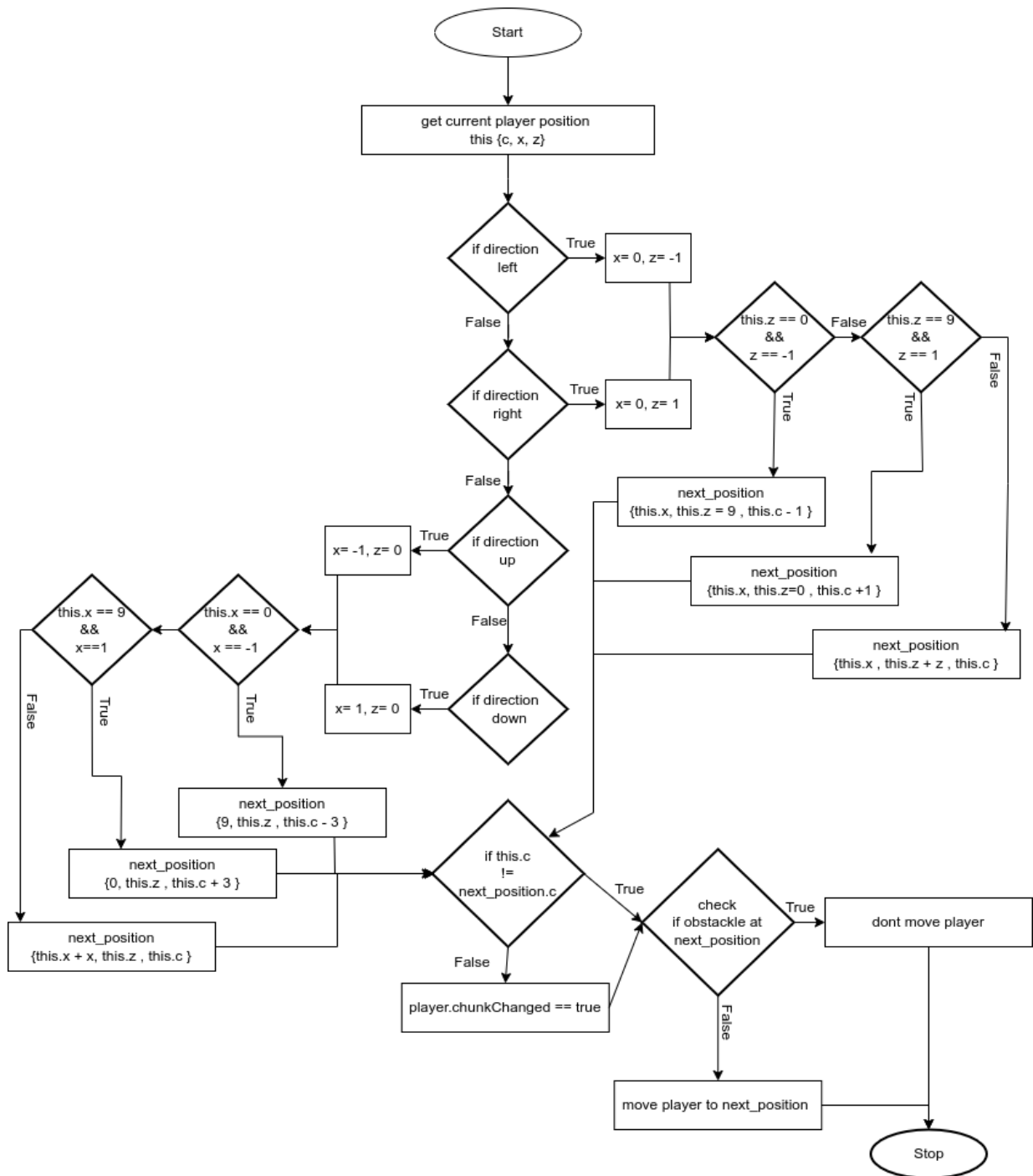


Figure 3. 12: Special NPCs Generation Activity Diagram

In any town there must be 4 characters of types merchant, apothecary, guild master and a way-point but none repeating. So check if they exist and if not assign the created character that specialty. In case of way-points, they are objects but for continence they are given the title of NPC and are assigned town_id which be used for its functionality.



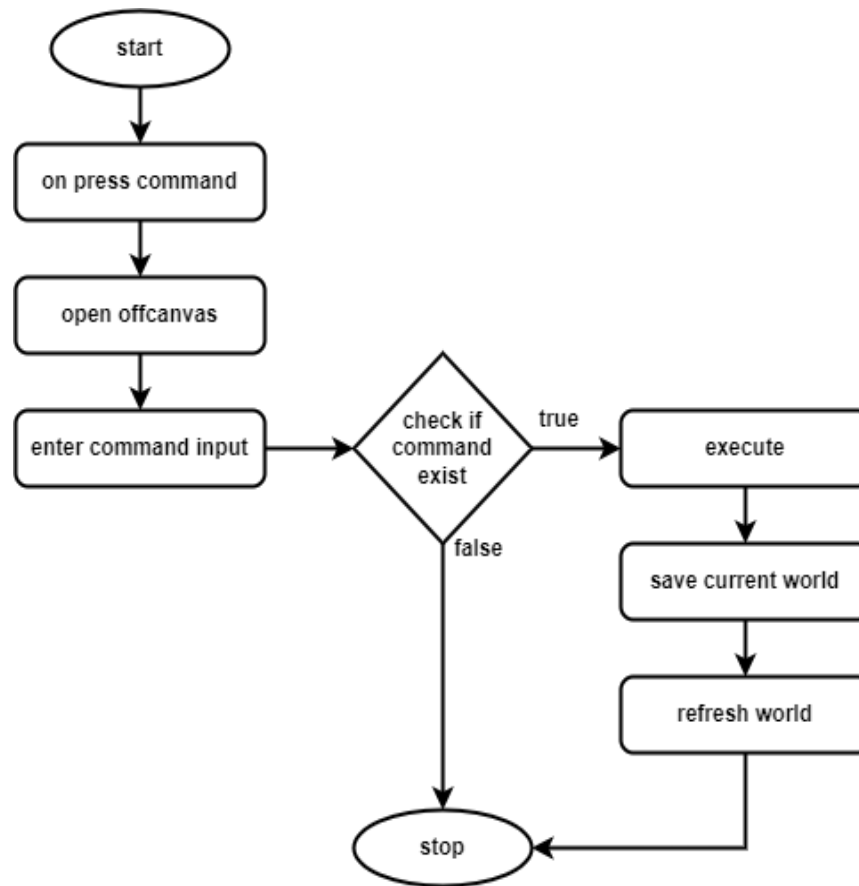


Figure 3. 14: Running Command/ Cheats

This flow chart describe about command or cheat that player can use. First press command box which appear an off canvas where player can enter the command or cheat they want. Then the command will be checked, if command exist then it will be executed else it will not be executed. After the command is executed, the current state of world will be saved and refresh the world.

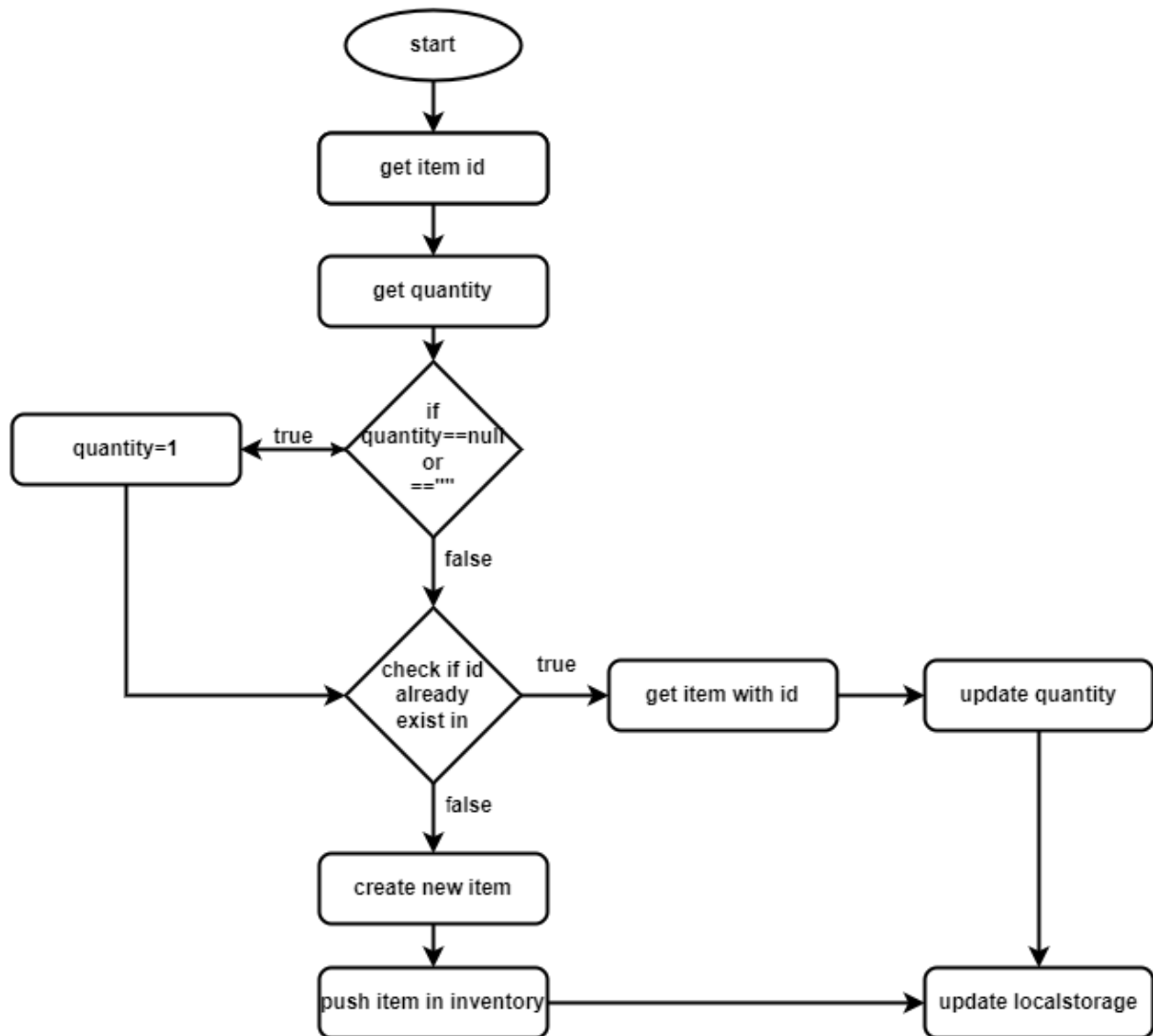


Figure 3. 15: Add Item in Inventory

In the process outlined in Figure 3.15, which pertains to adding an item to the inventory, the initial steps involve retrieving the item's ID and quantity from the Item.js module. The procedure then verifies whether the quantity is null or not. If the quantity is indeed null, the quantity is set to 1. Subsequently, the system checks whether the ID already exists within the dataset stored in the local storage, referred to as the Inventory.

If the ID is located in the Inventory data, the associated quantity is updated. Conversely, if the ID is not found, a new item entry is generated, and this new item is appended to the local storage dataset to ensure its preservation.

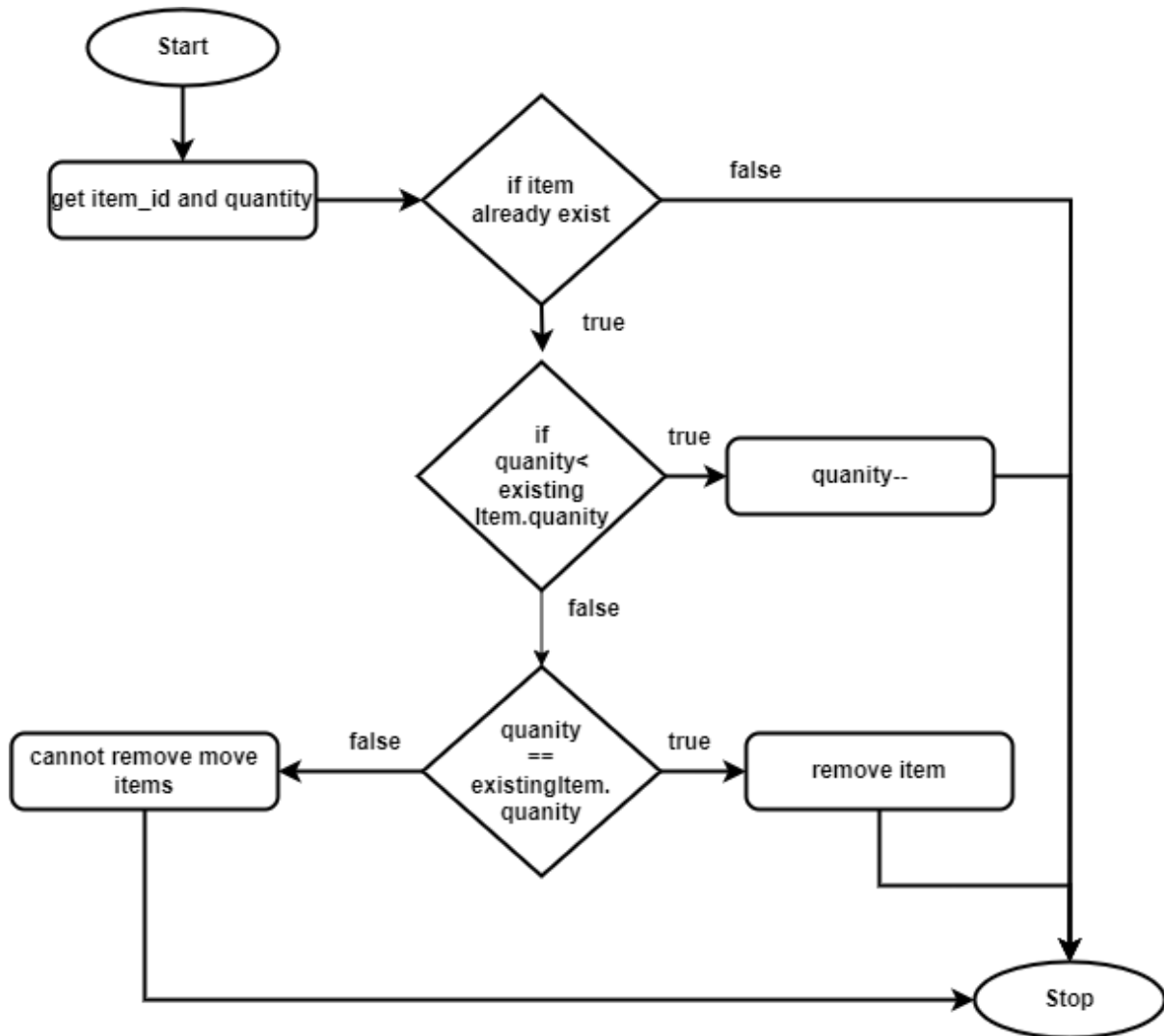


Figure 3. 16: Remove Item in Inventory

Above figure describe about removing item from Inventory. Firstly item id and quantity is retrieve from Item.js. The system checks whether the ID already exists within the dataset stored in the local storage, referred to as the Inventory. If id is located in Inventory data, the associated quantity will be check with existing quantity. If quantity is less than existing quantity then quantity data will be subtracted and if quantity is equal to existing quantity, item will be successfully remove.

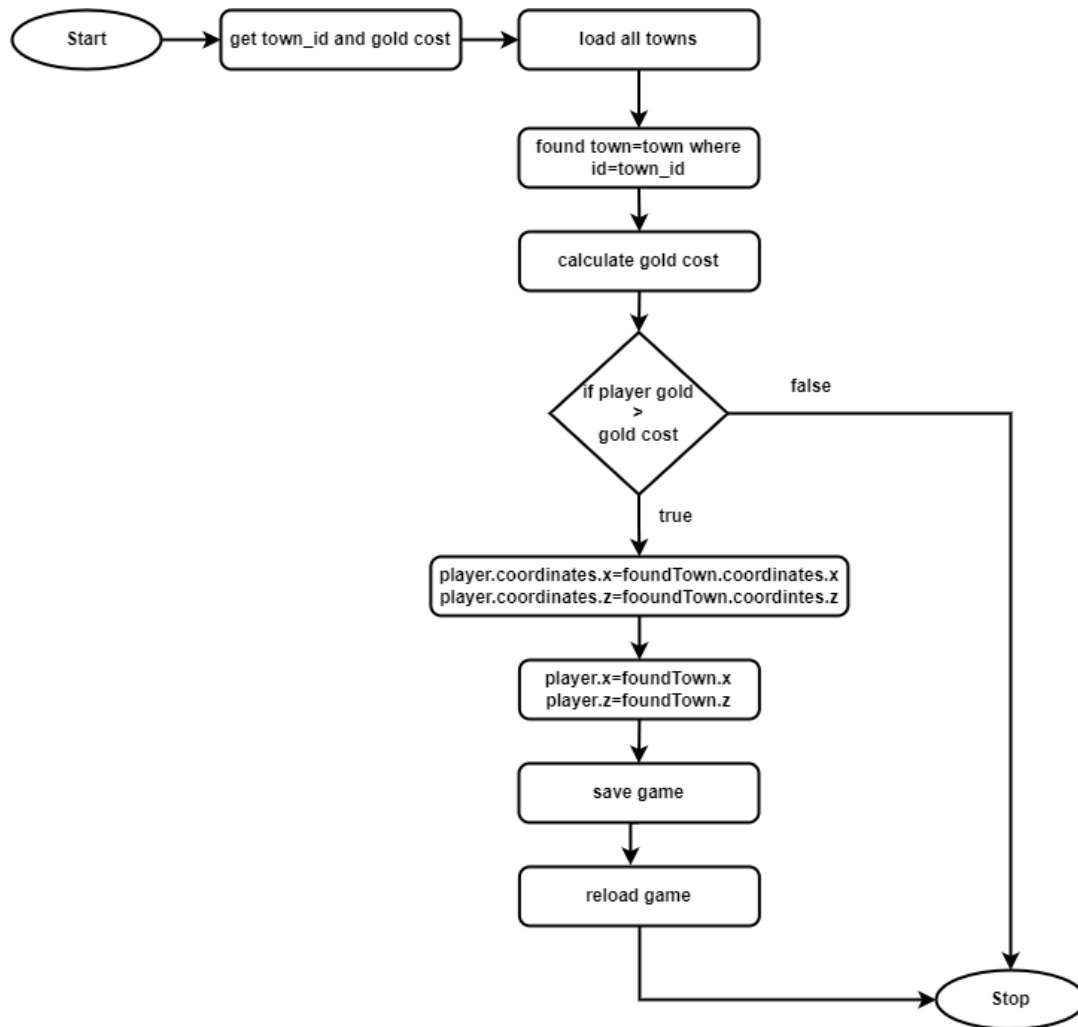


Figure 3. 17: Waypoint Interaction process

The depicted diagram illustrates the sequence of interactions for navigating waypoints. The process commences by acquiring the town's identification and its corresponding gold cost. Subsequently, all towns, excluding the present one, are loaded into the system.

Before initiating a teleportation to a different town, a check is conducted to determine whether the player possesses sufficient gold. Should the player's gold fall short, the ability to transition to another town is denied. Conversely, if the player's gold surpasses the required amount for the target town, the player's coordinates are updated to match those of the desired town. This progress is followed by a game save, and upon reloading, the player finds themselves successfully transported to their chosen destination.

3.2. System Design

3.2.1. Refinement of Class, Object, State, Sequence and Activity diagrams

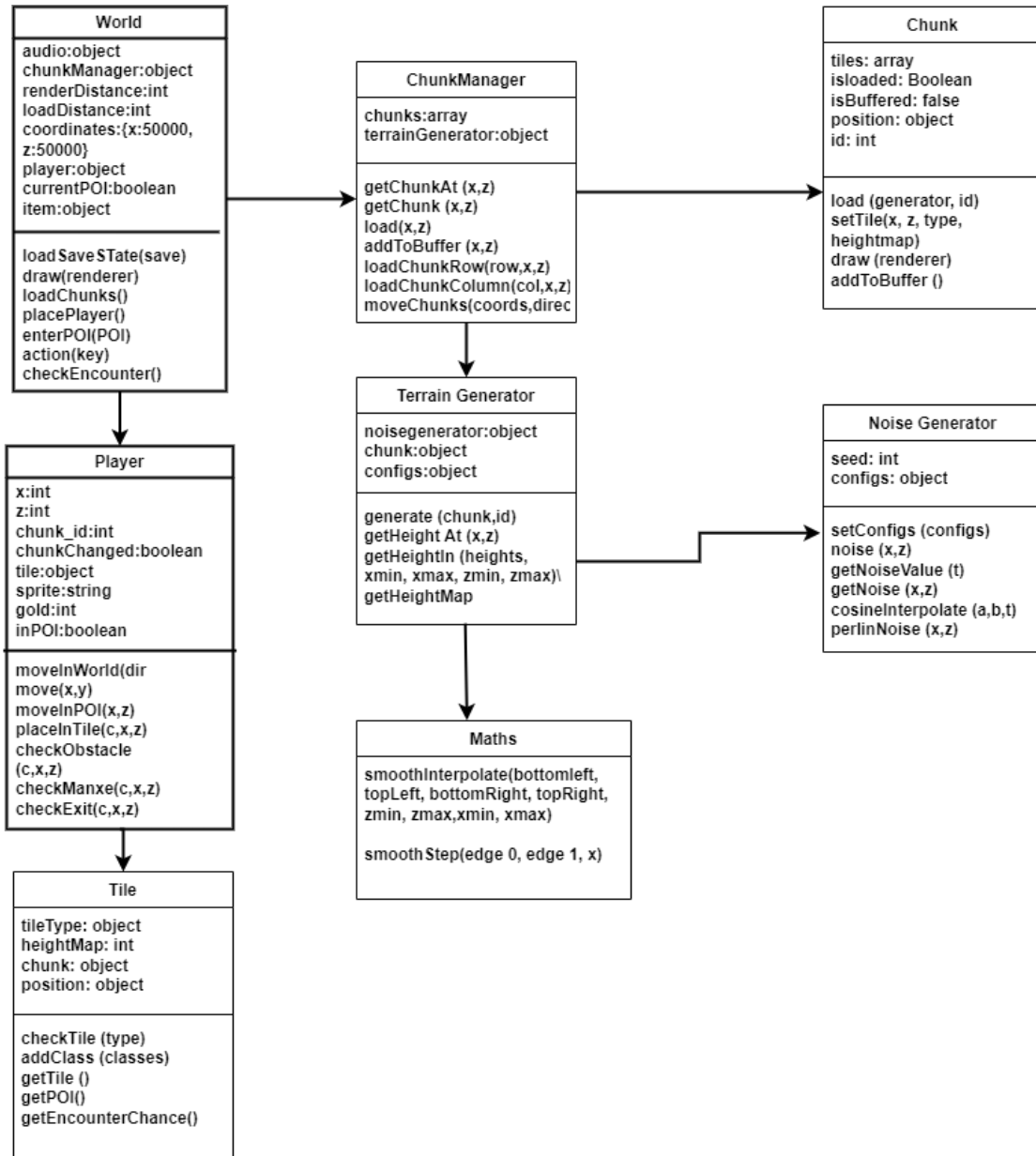


Figure 3. 18: Refinement Class Diagram of World Generation

3.2.2. Deployment Diagrams

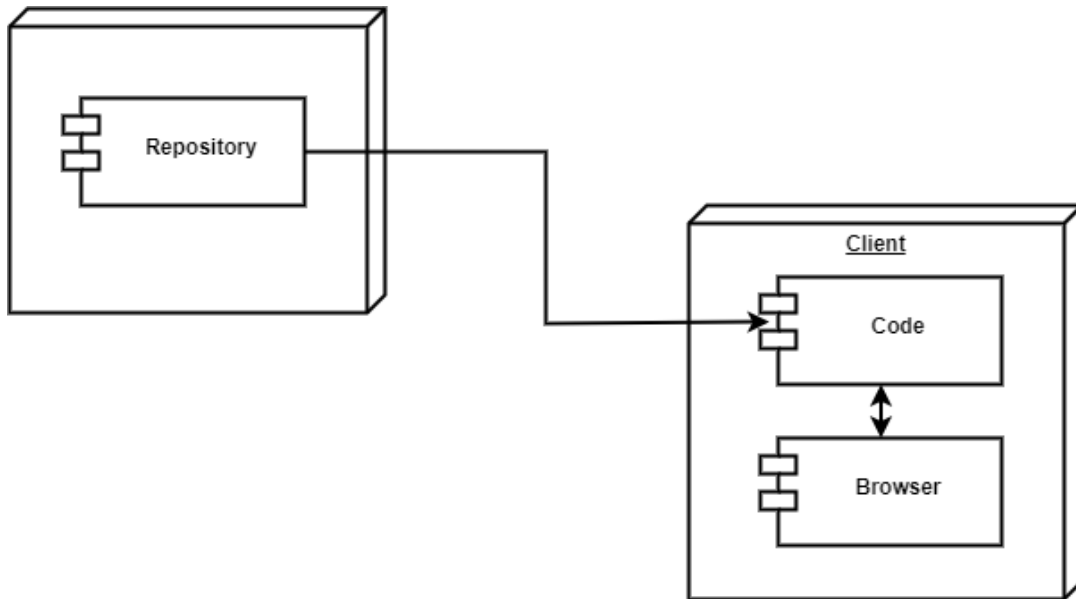


Figure 3. 19: Deployment Diagram of Game

The project is made in such a way that it doesn't have any online dependencies like servers to run. It needs only to be downloaded and run on the browser.

3.3. Algorithm Details (if any)

i Description of Algorithm

a. Perlin Noise

Perlin noise is a popular procedural generation algorithm invented by Ken Perlin. It can be used to generate things like textures and terrain procedurally, meaning without them being manually made by an artist or designer. The algorithm can have 1 or more dimensions, which is basically the number of inputs it gets. In this article, I will use 2 dimensions because it's easier to visualize than 3 dimensions. There is also a lot of confusion about what Perlin noise is and what it is not. It is often confused with value noise and simplex noise. There is basically 4 type of noise that are similar and that are often confused with one another : classic Perlin noise, improved Perlin noise, simplex noise, and value noise. Improved Perlin noise is an improved version of classic Perlin noise. Simplex noise is different but is also made by Ken Perlin. Value noise is also different. A rule of thumb is that if the noise algorithm uses a (pseudo-)random number generator, it's probably value noise. This article is about improved Perlin noise. First, how to use it.

The algorithm takes as input a certain number of floating point parameters (depending on the dimension) and return a value in a certain range (for Perlin noise, that range is generally said to be between -1.0 and +1.0 but it's actually a bit different). Let's say it is in 2 dimensions, so it takes 2 parameters: x and y. Now, x and y can be anything but they are generally a position. To generate a texture, x and y would be the coordinates of the pixels in the texture (multiplied by a small number called the frequency but we will see that at the end). So for texture generation, we would loop through every pixel in the texture, calling the Perlin noise function for each one and decide, based on the return value, what color that pixel would be. [3]

In this project perlin noise algorithm will be used to create a noise map which will be translated into the ingame terrain where the player will interact on.

b. A* Algorithm

A-Star (A*)search algorithm is an intelligent algorithm to solve a graph problem. Contrary to Depth First Search (DFS) and Breadth First Search (BFS), A* is an informed search algorithm which means that it takes into account the n of the goal while searching for it and hence it searches quite a few nodes to reach to the goal.

The way A* works is that it assigns a cost to each of the cells of the maze and the algorithm selects the path with minimum cost. The cost of a cell (n) has two parts and is defined as:

$$f(n) = g(n) + h(n)$$

Where f(n) is the total cost to reach the cell n and g(n) and h(n) are defined as:

g(n) → It is the actual cost to reach cell n from the start cell.

h(n) → It is the heuristic cost to reach to the goal cell from cell n. It is the estimated cost to reach the goal cell from cell n. [4]

In this project, A* algorithm will be used to find and generate one or more paths between different Point of interest (towns). Said path will provide a specific bonus like less enemy encounter chance or less gold loss.

$$\text{Encounter chance} = \text{steps taken} + \text{tile encounter rate} + \text{random number}$$

c. Insertion Sorting

Insertion sort is a simple sorting algorithm that works by building a sorted array gradually. It divides the input array into two parts: a sorted subarray and an unsorted subarray. The algorithm repeatedly takes an element from the unsorted subarray and inserts it into its correct position within the sorted subarray. This process continues until the entire array is sorted. Here's how insertion sort works:

- **Start with the second element:** Imagine you have an array, and you start with the second element (index 1). The first element (index 0) is considered already sorted.
- **Compare and insert:** Take the second element and compare it with the first element. If it's smaller (or larger, depending on the sorting order), you move the first element one position to the right to make space for the second element. Then, you insert the second element into the empty position.

- **Extend the sorted portion:** Now you have the first two elements in sorted order. You take the third element (index 2) and compare it with the second element. If it's smaller, you shift the second element to the right, and similarly, you continue this process until you find the correct position for the third element.
- **Repeat:** You continue this process for each subsequent element in the array, extending the sorted portion with each iteration.
- **Final sorted Array:** You continue this process for each subsequent element in the array, extending the sorted portion with each iteration.

CHAPTER 4: IMPLEMENTATION AND TESTING

4.1. Implementation

4.1.1. Tools Used

➤ Case tools

- **Diagram.net**

Diagrams.net/draw.io is an open source technology stack for building diagramming applications, and the world's most widely used browser-based end-user diagramming software. Its interface can be used to create diagrams such as flowcharts, wireframes, UML diagrams, organizational charts, and network diagrams.

- **Visual Studio Code**

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

➤ Markup Languages

- **HTML**

HTML(HyperText Markup language) is a standard markup language for web creation. HTML was used to give structure to the project. It was used as the core programming language to show the users the GUI based system.

- **CSS**

The sole purpose of using CSS was to beautify the interface created using HTML.

- **Javascript**

JavaScript is an object-oriented computer programming language. JavaScript was mostly used for core business logic in the client side. This includes form validations and showing pop up messages of errors along with showing responsive content to the user.

➤ Piskelapp

Piskel is a free online sprite editor for creating 2d pixel art and animations for game development. Piskel provides a user-friendly interface and a range of features specifically designed for pixel art creation.

4.1.2. Implementation Details of Modules

In this section, we will explore the implementation details of the various modules in our game development project. Each module plays a crucial role in creating a immersive gaming experience.

i. World Generation Module

- This module utilized the Perlin Noise Algorithm to generate the game world.
- The Perlin Noise algorithm generates coherent terrain, incorporating feature like mountains, forest, and sand.
- It takes into account various parameters such as seed value.

ii. Character Movement Module

- This module enables the player to control the character's movement within the game world.
- It utilized input handling mechanism to capture user commands, such as keyboard or controller inputs.

4.2. Testing

4.2.1 Test Cases for Unit Testing

Table 4. 1: Test Case for New Player Placing

Test case	Test Data	Expected outcome	Test Result
New Player placement	World Seed:12211	Player is placed on chunk 5(4,7)	pass
	World Seed:5120531	Player is placed on chunk 5(5,5)	pass
	World Seed:1311432	Player is placed on chunk 5(4,4)	pass

Table 4. 2: Test Case for Player Movement


Test Case	Test Data	Expected Outcome	Test Result
Player Movement	 Player move up	Cannot move up because of water tile	pass
	Player move left	Can move to left	pass
	Player move right	Can move to right	pass
	Player move down	Can move down	pass

Table 4. 3: Test Case for Town Visit


Test Case	Test Data	Expected Outcome	Test Result
Town	 Player move up	Enter visited town	pass
	Player move up	Enter new town	pass

Table 4. 4: Test Case for Teleportation

Test Care	Test Data	Expected Outcome	Test Result
Teleportation	/town 41541	Player moved to town with id 41541	pass
	/town randomId	Player is not moved	pass
	/tp 4545 4545	Player placed in (45454,4545) at tile(5,5)	pass
	Click teleport button on waypoint of town 41541	Player placed in town at gate of 41541	pass
	Gold=30 waypoint use cost=60 on use waypoint	Player is shown bot enough gold and not teleported	pass
	Gold=30 waypoint use cost=10 on use waypoint	Player is teleported to gate to town	pass

4.2.2 Test Case for System Testing

Scenario 1: for new game creation

- First user open the site and sees landing page.
- In landing page, user need to click new page button.
- When button is clicked, user will redirect to the character creation page.
- He/she can choose any character as per their choice in this page.
- After choosing character new game will be created.

Scenario 2: World generation

- User need to open the seed generation page.
- He/ she will be given two option i.e. user can give the seed or user can use random seed.
- As per the seed, world will be generated.

Scenario 3: Save and load game

- To save the game, player should press 'esc' button, leading to appearance of modal box.
- He/she will be given different option like "save", "save and quit", "quit" where "save" button will only save the game state whereas "save and quit" saved the game and exit out of the game.
- To load game, user need to open landing page and press "Continue" button which will lead player to the same game state which they leave before saving.

Scenario 3: Import and Export

- User move to landing page.
- He/she will press setting button, which appear a modal with import and export button.
- He/she need to click export button to export the data of game.
- He/she need to click import button to import the data.

CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATIONS

5.1. Conclusion

On overall, in development of this project, we have utilized Perlin Noise algorithm to generate world, providing unique and diverse landscape for player to explore and make the movement of the player as per input handling. In this project, player also moved by checking collision detection i.e. if the tile is water player are not allowed to move. Also, player can give command or cheats to move from one town to another or to gain or remove items. Player can teleport from one town to another using waypoint located inside the town based on gold cost. Therefore, the development of world generation depend on seed based generation.

5.2. Future Recommendations

The game which we have been developing is working properly but as for the future many features can be added like

- Unique characteristics of characters and items.
- Additional types of enemies and items
- Additional types of POIs
- A more complex combat system with magic, statuses(poison, burn, sleep) and weapon triangles

APPENDICES

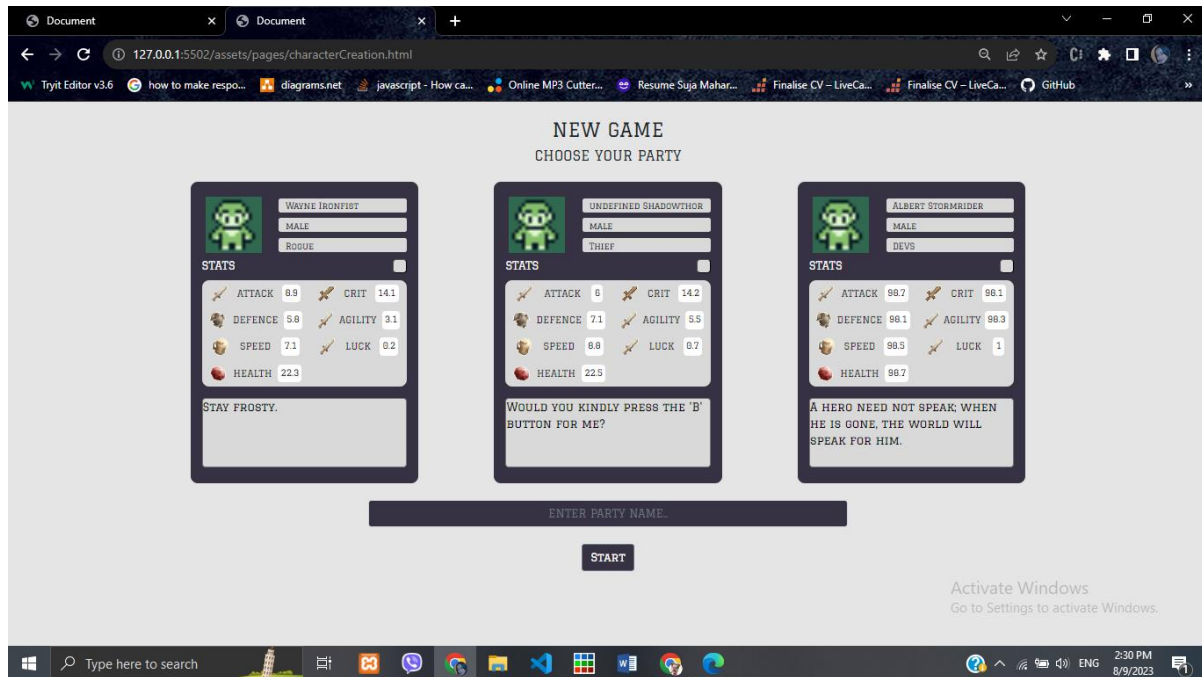


Figure 6. 1: Character Creation Page

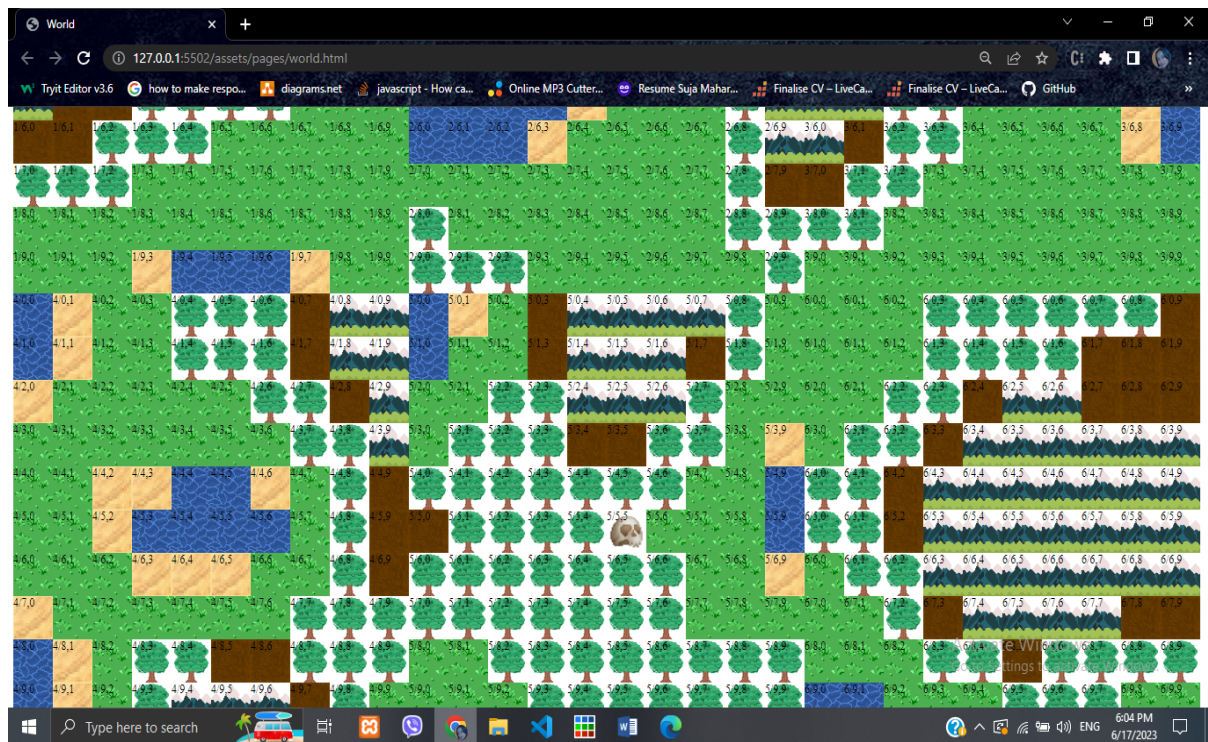


Figure 6. 2: World Generation Page

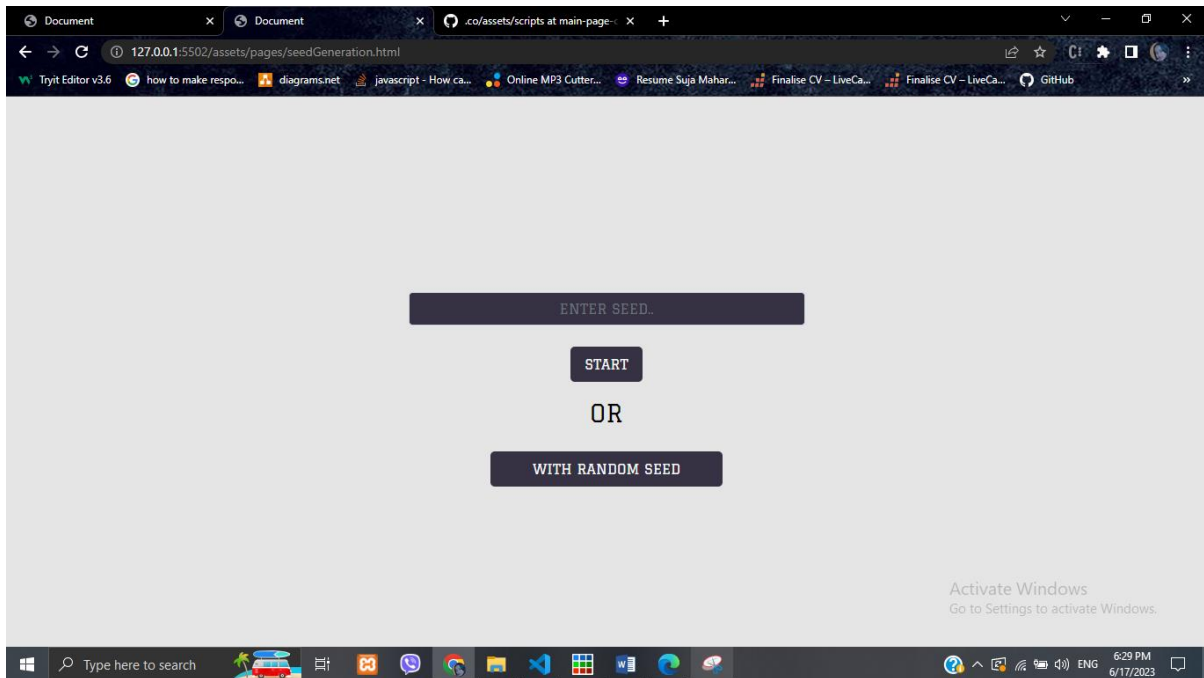


Figure 6. 3: Seed Generation Page

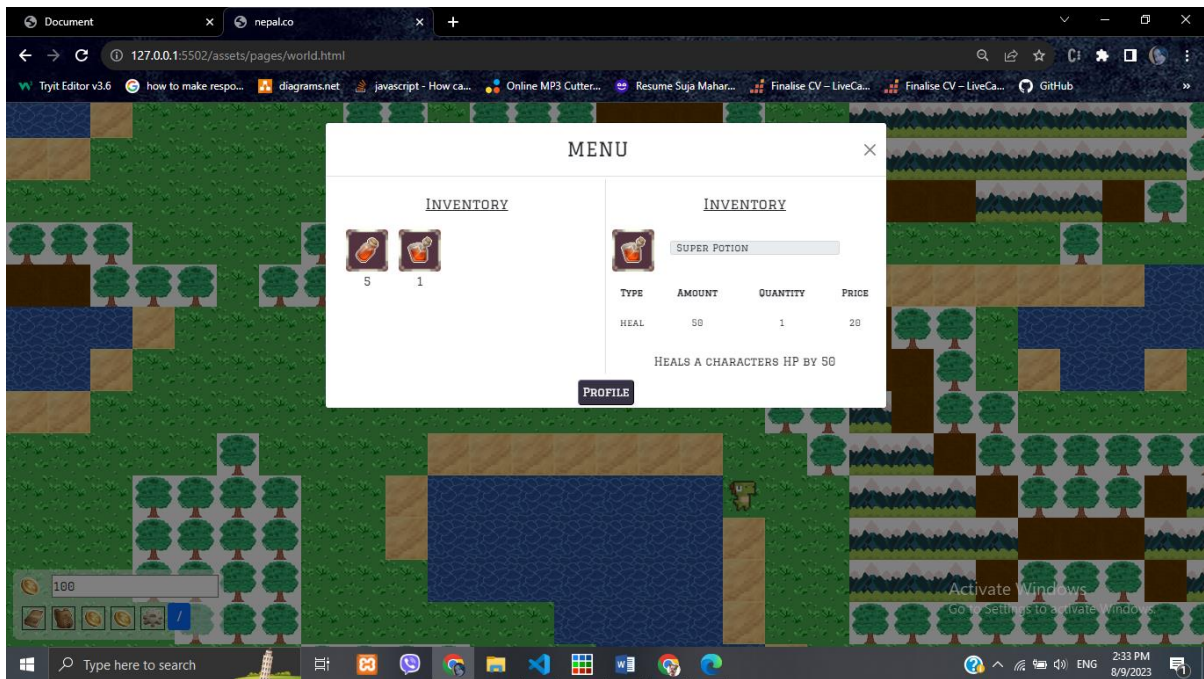


Figure 6. 4: Inventory Modal







SAVES		
(62806885992)	CREATED AT: 2023-08-06	
CO		
SEED :33007495939705		
 100	 3	
LOAD	DELETE	LAST SAVE: NULL
(122300882)	CREATED AT: 2023-08-07	
TRY		
SEED :439081015410215		
 100	 3	
LOAD	DELETE	LAST SAVE: NULL
(95827595604)	CREATED AT: 2023-08-09	
NEPAL		
SEED :823854052763788		
 100	 3	
LOAD	DELETE	LAST SAVE: NULL

Figure 6. 5: Saves Data

REFERENCES

- [1] T. Anome, "wikipedia," 7 march 2023. [Online]. Available:
https://en.wikipedia.org/wiki/Procedural_generation.
- [2] O. Tobar, "Game Development-From the vision to the final product," Laooeenranta
University of Technology.
- [3] R. Blog, "Perlin Noise: A Procedural Generation Algorithm," [Online]. Available:
<https://rtouti.github.io/graphics/perlin-noise-algorithm>.
- [4] M. A. Naeem, "levelup," 18 October 2021. [Online]. Available:
<https://levelup.gitconnected.com/a-star-a-search-for-solving-a-maze-using-python-with-visualization-b0cae1c3ba92>.