



Orbital 2025

Nanban Requiem

Team ID: 7366

Team Name: Spicy Tartar Chicken Nanban Don

Luke Aidan Tan (A0307717B)

Chen Ziming (A0308962W)

Team Name

Spicy Tartar Chicken Nanban Don

Proposed Level of Achievement

Apollo 11

Liftoff Poster and Video

Poster:  7366.png


Video:  7366.mp4

Milestone 1 Poster and Video


Poster:  7366.png

Video:  7366.mp4

Project Log

 Nanban Requiem Project Log

Current Build

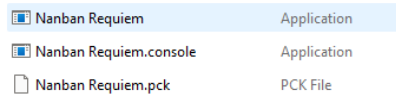
 Nanban Requiem

Instructions

1. Download the zip file from the **Current Build** section and extract it.



2. You will now see a folder named **Nanban Requiem**. Open this folder.



3. Launch the application Nanban Requiem, highlighted in the image above. If Windows Defender flags it as unsafe, click **run anyway**.

Motivation

We are both huge fans of the popular 2D tower defense game, Arknights. It offers highly addictive gameplay, encouraging players to strategise and plan their defense given the situations in each map. What separates it from other common tower defense games is that it introduces unique elements into gameplay that forces you to think outside the box, juggling limited resources and defending against enemies in a time constraint, while keeping the gameplay fresh and non repetitive. We decided to try our hand at making a tower defense game that also offers this level of enjoyment, delivering an immersive experience for new and experienced tower defense players, while adding our own flair to it.

User Stories

- As a strategic player, I can deploy both melee and ranged towers with unique abilities so that I can experiment with different combinations and adapt to enemy types.
- As a player who values immersion, I can see detailed animations and visual effects for towers, enemies, and projectiles so that the gameplay feels more dynamic and engaging.
- As a challenge-seeking player, I can face waves of increasingly powerful enemies while managing limited resources, encouraging me to think critically and weigh the priorities of different choices.
- As a creative player, I can interact with the terrain/environment so that I can influence the battlefield and come up with unique strategies.

Storyline

What began as a peaceful lunch break has erupted into a full-blown lunch time war. You were just about to dig into your lunch, when all of a sudden, hordes of hungry enemies came rushing in. They have a single objective - to partake in your bowl of Spicy Tartar Chicken Nanban Don.

But you refuse. And you are not defenseless.

Every second counts. With the allies standing beside you, strategically deploy your towers across the map, be it close quarters or ranged warfare. Every tower matters. Every bite is at stake.

This is the Nanban Requiem.

Scope

Project Scope

Nanban Requiem is a 2D top-down tower defense game developed using Godot engine, where the core gameplay revolves around strategically deploying two types of towers: ranged and melee, to defend the base from waves of incoming enemies.

Description

Towers serve as the primary defense mechanism, where ranged towers attack enemies from a distance while melee towers block enemy progress and engage in close combat, with a limited block capacity. There will be a cap for the number of towers deployed at any given time to introduce critical thinking involved in understanding the enemy and using the towers' abilities to one's advantage.

Enemies consist of both melee and ranged types, navigate a predefined path using a path-following algorithm. Ranged enemies can attack towers or the base from a distance, while melee enemies must reach their target directly.

The win condition is achieved when all enemy waves on the map are cleared. The lose condition occurs if the Base HP is reduced to zero by enemies entering the base.

Tech Stack

Our game will be developed using the following tools and technologies:

Godot Engine 4.4.1

- A lightweight, open-source game engine used to build our 2D gameplay systems and visuals.

GDScript

- Godot's built-in scripting language, ideal for rapid prototyping and implementing core game logic.

Git & GitHub

- Used for version control, collaboration, issue tracking, and managing the development process.

Feature Timeline

Liftoff (12 May - 19 May)

1. Designing liftoff poster and presentation video
2. Learn how to effectively version control using Git/Github through workshops

Milestone 1 - Ideation (20 May - 2 June)

1. Creation of Main Menu
2. Creation of Map 1
3. Creation of melee and ranged tower classes
4. Creation of melee and ranged enemy classes
5. Basic UI for gameplay (Build Mode, Pause/Play, Fast Forward)
6. Dedicated targeting system for towers and enemies
7. Projectile class, projectile spawning and tracking logic
8. Basic path following algorithm for enemies
9. Tower and Enemy despawn logic
10. Towers blocking enemy logic
11. Base HP and Game Over condition

Milestone 2 - Prototype (3 June - 30 June)

1. Integration of C# in addition to GDScript for OOP/SWE
2. Manage tower deployment and deployment resources
3. Creation of settings menu (volume modulation etc)
4. Addition of Music soundtrack
5. Creation of more towers and enemy types/classes
6. Addition of tower and enemy skills
7. Polishing of Map 1 as well as the addition of Map 2 and 3
8. Polishing of UI and sprites
9. User testing to gather feedback and refine the game experience

Milestone 3 - Extended system (1 July - 28 July)

1. More towers and enemy types
2. Enemies dynamically choose or change paths based on certain conditions

3. Instant Redeployment to respond to smart enemy behaviour
4. Terrain interactables
5. Smart wave generation + difficulty scaling
6. Achievements

Current Progress

Full documentation of these features can be found in the next section.

Features	Details
Main Menu	Functional “New Game” and “Quit” buttons
Game Map	Only 1 map has been created so far
Towers	4 unique towers <ul style="list-style-type: none">• 2 melee• 2 ranged
Enemies	3 unique enemies <ul style="list-style-type: none">• 2 melee• 1 ranged

Features

Main Menu



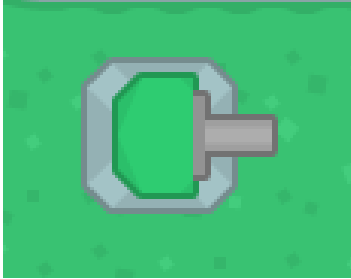
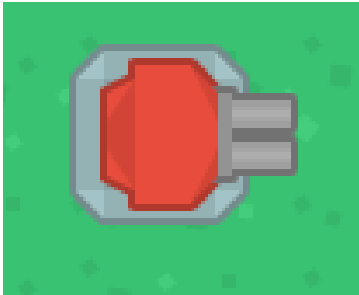

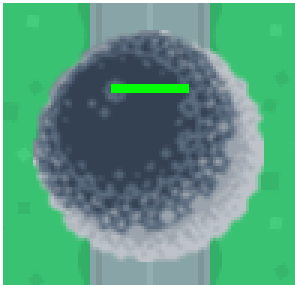
The Main Menu consists of the New Game, Settings, Credits and Quit buttons.

New Game loads the first map scene, where the gameplay begins.




Quit closes the game.

Settings and Credits buttons are a work in progress.

Towers

 Pew Pew	<p>Pew Pew is the first ranged tower in the game. It shoots projectiles at the enemies, constantly targeting the enemy closest to the base.</p> <p>HP: 1000 Ranged attack damage: 10 Attack speed: 3</p>
 Rocketeer	<p>Rocketeer has the same stats as Pew Pew, but deals Area of Effect (AoE) damage to enemies</p> <p>HP: 1000 Ranged attack damage: 10 Attack speed: 1</p>
 The Boulder	<p>The Boulder is the first melee tower that blocks enemies. It serves as a deployable obstacle or a meatshield to hinder enemy movement. It is also unable to attack enemies. Respect the Boulder.</p> <p>HP: 1000 Block count: 2</p>
 Inverse Boulder	<p>The Inverse Boulder is a copy of The Boulder that offers melee attacks.</p> <p>HP: 1000 Block count: 2 Melee attack damage: 30</p>

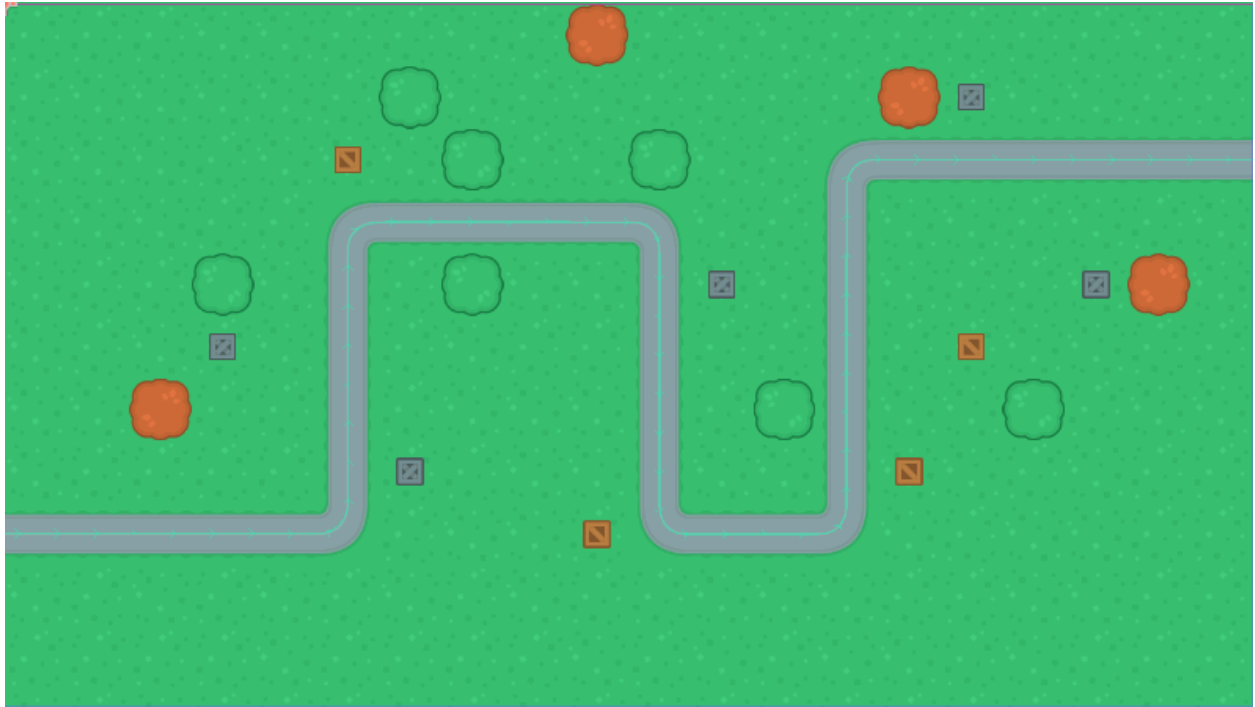
Enemies

 Samurai	<p>The Samurai is the first enemy encounter in the game. It is a melee enemy that only attacks towers in the path.</p> <p>HP: 300 Movement Speed: 100 Melee Attack Dmg: 100 Melee Attack speed: 0.5</p>
 Rocket Samurai	<p>The Rocket Samurai is a ranged variant of the Samurai with both melee and ranged modes.</p> <p>When blocked by a melee tower, it will initiate melee mode, attacking the tower until it is dead.</p> <p>When not blocked, it will launch projectiles at the nearest tower.</p> <p>Rocket Samurai deals twice more damage in ranged mode than in melee mode.</p> <p>HP: 300 Movement Speed: 75 Melee Attack Dmg: 50 Melee Attack speed: 0.5 Ranged Attack Dmg: 100 Ranged Attack speed: 1</p>
 Mighty Whitey	<p>Mighty Whitey is a buffed variant of the Samurai, with double its melee attack damage and movement speed, but half its HP.</p> <p>HP: 150 Movement Speed: 200 Melee Attack Dmg: 200 Melee Attack speed: 0.5</p>

Maps

As of now, there is only one map in the game being used for testing. Enemy waves are being spawned infinitely until the end of time.

Map 1 (Highway to Spice)



The first map offers a basic linear path that gives players time to experiment with the unique features of each tower and defend against the waves of enemies.

In-Game UI

Settings



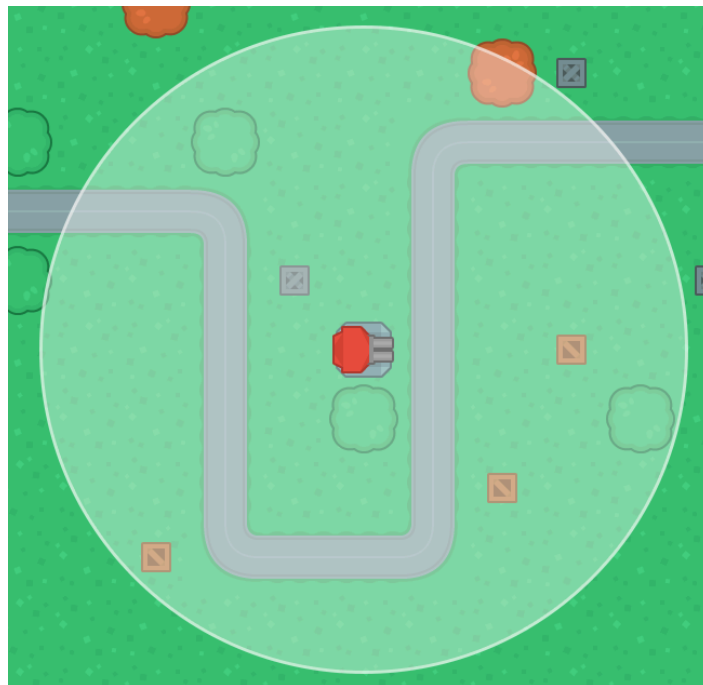
The In-Game UI consists of Pause / Play, Fast Forward and Restart.

The Pause / Play button gives players full control over the game's flow by allowing them to pause and resume gameplay at any time.

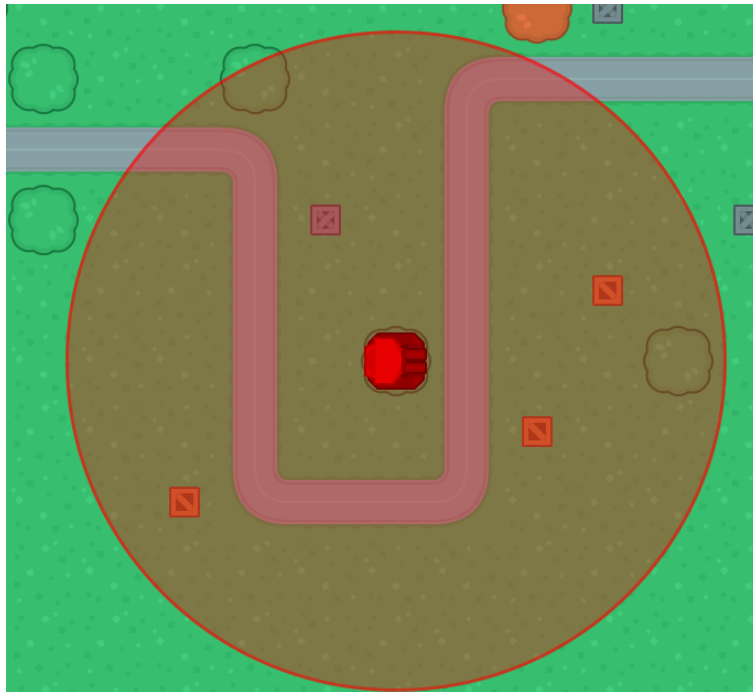
The Fast Forward button toggles between 1x and 2x speed to fit different players' needs. We added this feature as we felt that this was a much needed QoL feature in tower defense games, that addresses a common pain point in tower defense games: extended downtime between waves or during slower combat segments.

The Restart button sends the player back to the main menu if he/she wants to restart a new game or quit the game.

Build Mode



In Build Mode, the tower selected will be shown in a Tower Preview, where said tower will follow your mouse to where you want to deploy it on the map. If the build location is valid, the tower will be deployed successfully.



If the build location is invalid, the tower and its range indicator will flash red. If the player attempts to build on an invalid location, the Build Mode will be cancelled. Invalid locations include specific tiles and existing towers on the map.

Base HP



The total remaining base HP is indicated on the top left hand corner of the screen. At full HP, the HP bar is green. With more HP lost, the colour of the bar changes to yellow, followed by red. The base HP reduces by 1 with each enemy entering the base. Upon the base HP reaching zero, the game will end and the player will be redirected to the Main Menu.

Testing

Integration Testing

We adopted functionality testing to ensure our game functions as intended. It involves testing the game's core features, mechanics, and gameplay to identify and fix any bugs or issues that could hinder the player's experience. This also verifies that the game adheres to all technical and functional requirements outlined in design documents and specifications.

No.	Test Case	Results
1	Click Quit on Main Menu	Game window closes
2	Click New Game on Main Menu	Map1 loads up
3	Click Pause button during gameplay	Gameplay is paused
4	Click Play button during gameplay	Gameplay is resumed
5	Click Fast Forward button	Gameplay speeds up to 2x speed
6	Click Restart button during gameplay	Stage progress is restarted
7	Click Tower build button	Tower Preview mode entered, selected tower and its range indicator follows the mouse to any point on the map
8	Hover mouse over a valid map tile while in Tower Preview mode	Selected tower and range indicator in tower preview remain unchanged
9	Hover mouse over an invalid map tile while in Tower Preview mode	Selected tower and range indicator in tower preview turn red
10	Hover mouse over an existing Tower while in Tower Preview mode	Selected tower and range indicator in tower preview turn red
11	Click a valid map tile while	Tower is successfully deployed onto the

	in Tower Preview mode	selected map tile
12	Click an invalid map tile while in Tower Preview mode	Build Mode is cancelled
13	Click a map tile with an existing Tower while in Tower Preview mode	Build Mode is cancelled
14	Samurai enters attack range of Ranged Tower	Ranged Tower attacks Samurai with projectiles Visual Health Bar of Samurai decreases
15	Samurai leaves the attack range of Ranged Tower	Ranged Tower stops attacking Samurai with projectiles
16	Samurai gets killed by Ranged Tower	Samurai despawns from the map
17	Ranged Tower enters the attack range of Rocket Samurai	Rocket Samurai attacks Ranged Tower with projectiles Visual Health Bar of Ranged Tower decreases
18	Ranged Tower is no longer in attack range of Rocket Samurai	Rocket Samurai stops attacking Ranged Tower with projectiles
19	Ranged Tower gets killed by Rocket Samurai	Ranged Tower despawns from the map
20	Enter Build Mode and click on a tile with a previously despawned Tower	Tower is successfully deployed onto the selected map tile
21	Melee Tower placed in front of enemy path	Enemies are blocked by the Melee Tower. Rocket Samurai changes from ranged to melee mode Samurai performs melee attacks
22	Melee Tower is destroyed	Enemies continue moving along the path. Rocket Samurai returns to ranged mode.
23	Enemy enters base	Base HP reduces by 1
24	Base HP reduces to zero	Game is redirected back to Main Menu

Automated Unit Testing

```
Finished 0.0s

res://Unit Tests/test_MeleeTower.gd
* test_take_damage
* test_damage_equals_hp
* test_damage_higher_than_hp
3/3 passed.

res://Unit Tests/test_RangedTower.gd
* test_take_damage
* test_damage_equals_hp
* test_damage_higher_than_hp
3/3 passed.

res://Unit Tests/test_RocketSamurai.gd
* test_take_damage
* test_damage_equals_hp
* test_damage_higher_than_hp
3/3 passed.

res://Unit Tests/test_Samurai.gd
* test_take_damage
* test_damage_equals_hp
* test_damage_higher_than_hp
3/3 passed.
```



```
=====
= Run Summary
=====

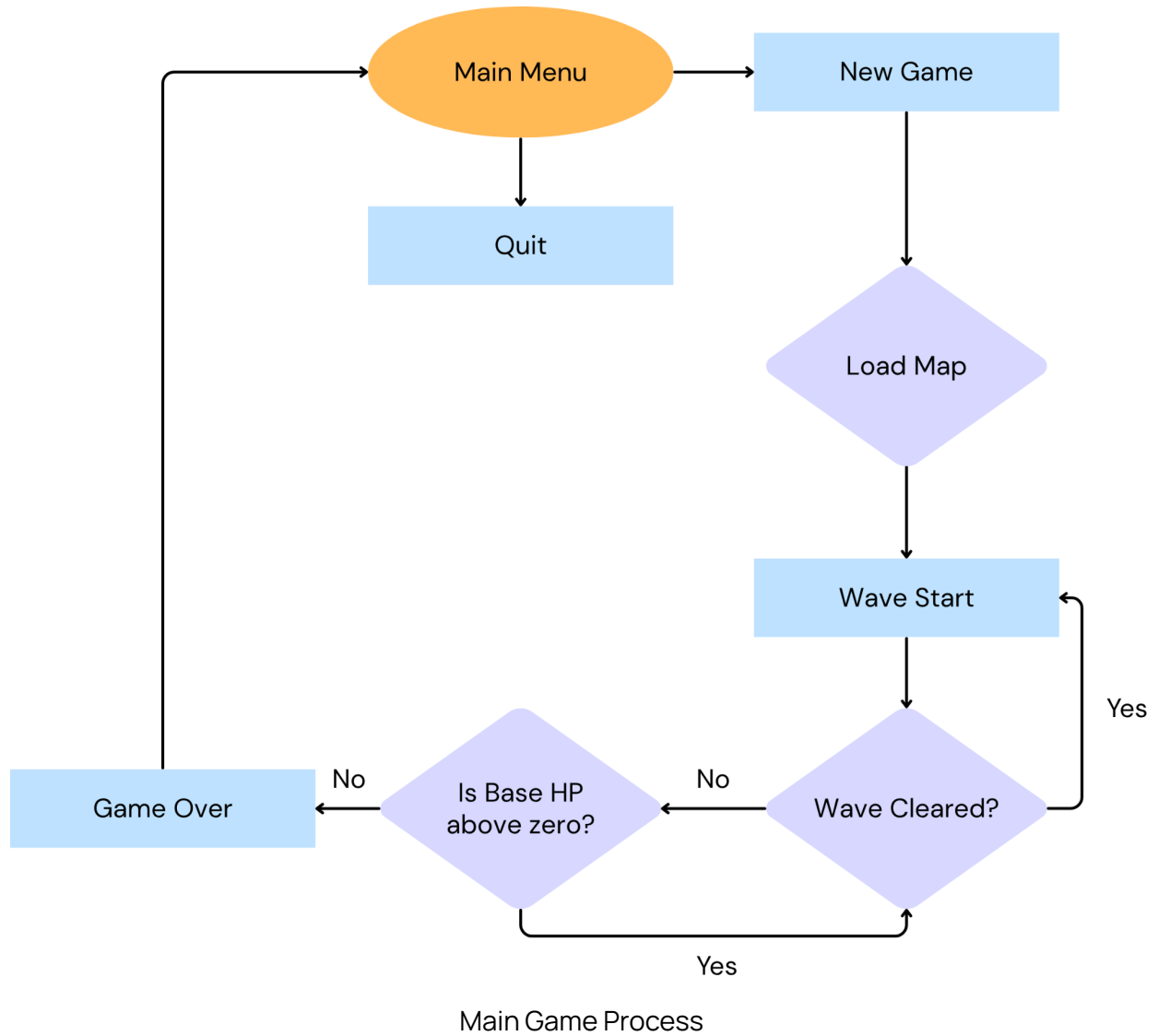
---- Totals ----
Scripts          4
Tests            12
  Passing        12
Asserts          12
Time             0.016s

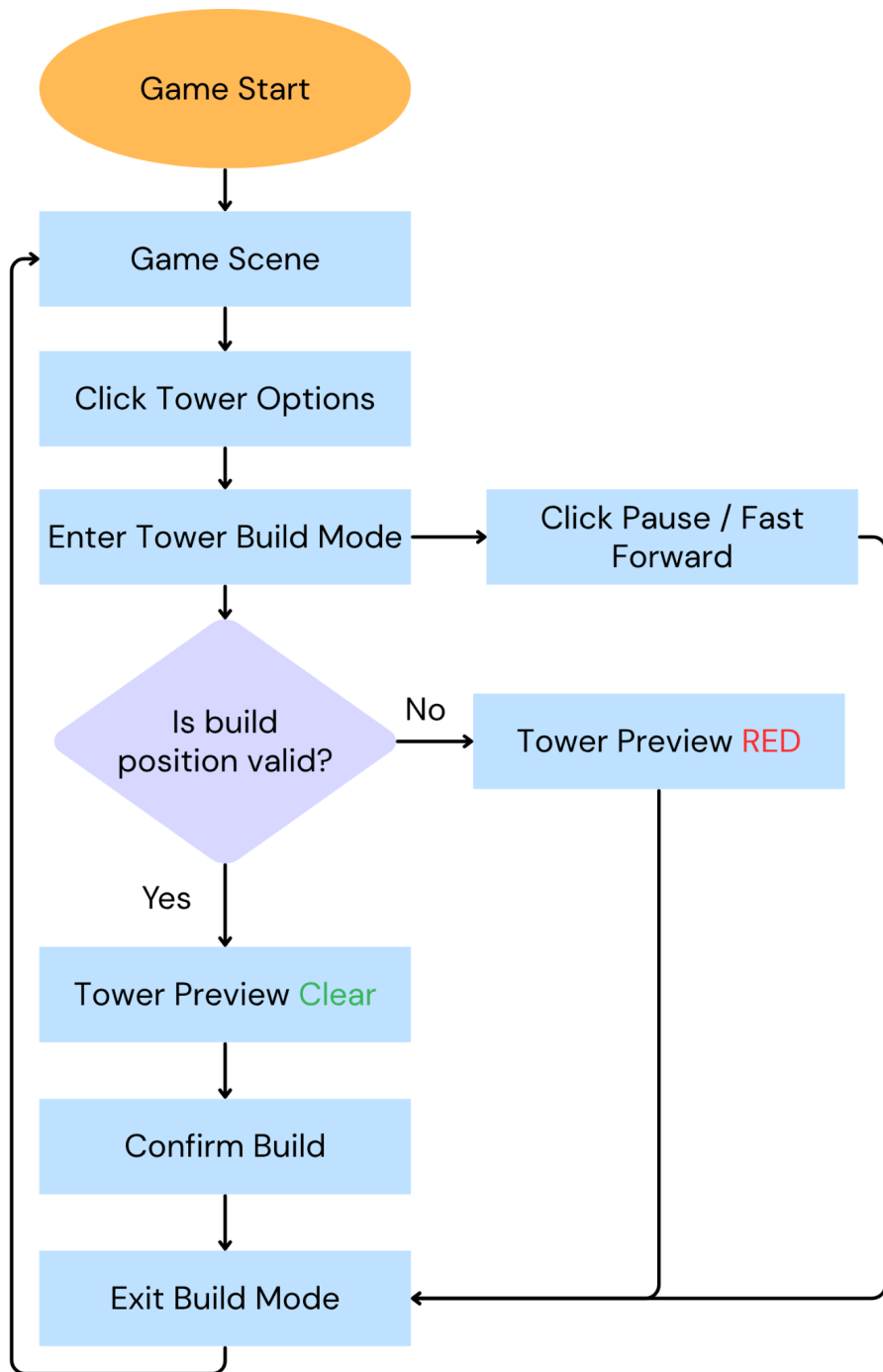
---- All tests passed! ----
```

We used Godot Unit Testing (GUT) from the Godot assets library and ran a couple of test cases for Ranged and Melee Tower and Enemy functions. This helped to ensure that each unit of code in the game worked as intended and met the requirements, improving the overall quality of the game. It also allowed us to detect and fix issues early before they snowballed into bigger problems.

Game Flow

Game Flow Diagram

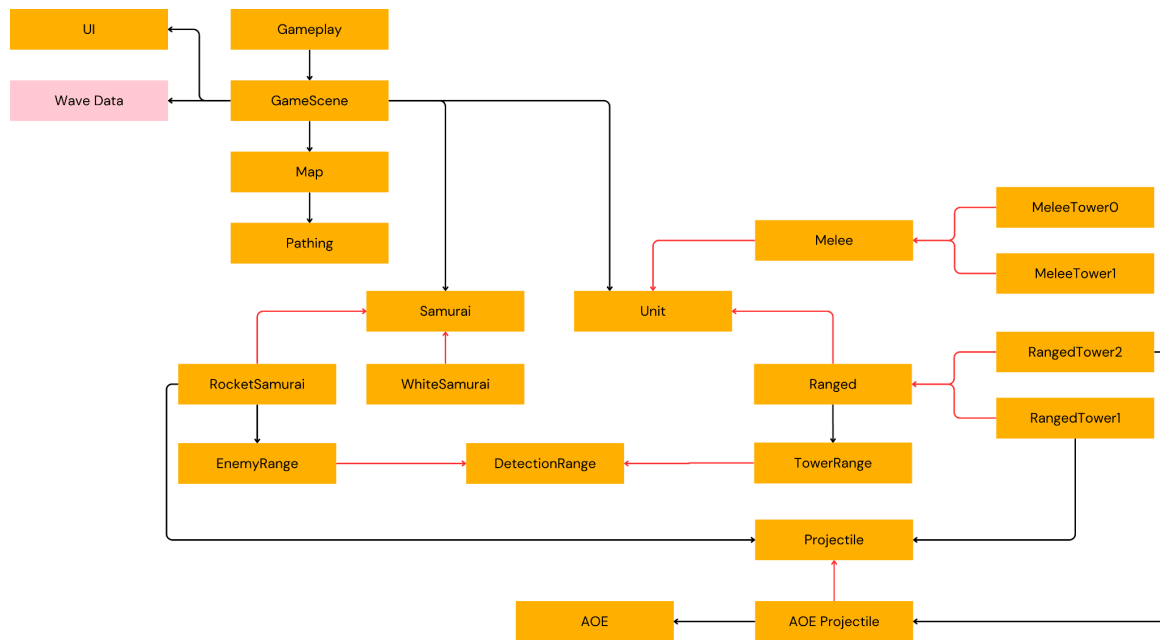




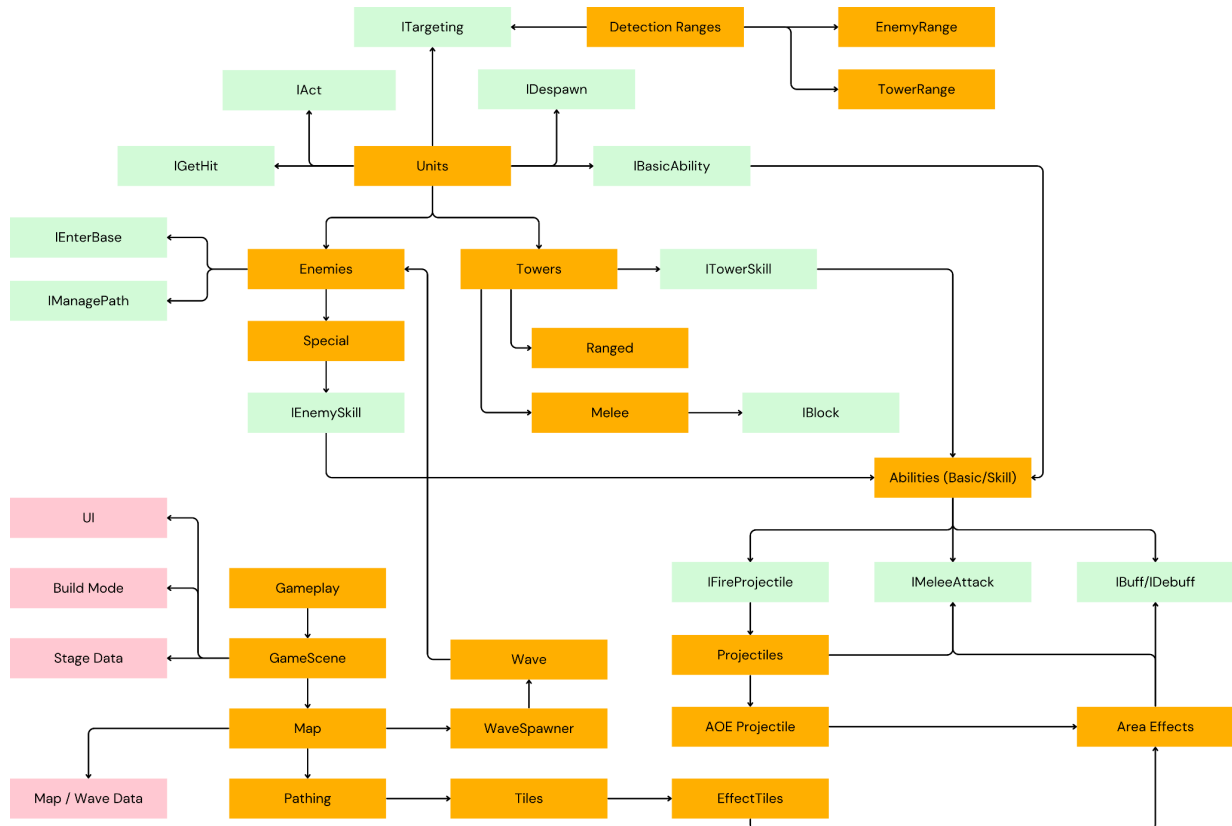
Build Mode / Tower Preview process

Class Diagram

Current



Planned



SWE Principles

Single Responsibility Principle

An example is the AOE class - its only responsibility is to apply damage to enemies within its range. An example for improvement would be the current DetectionRange class - which handles both detection and targeting. In our proposed diagram, we would shift the targeting to a separate class implementing the ITargeting interface.

Open-Closed Principle

In our proposed diagram, we plan to use interfaces in C# to define modular classes with specific roles - such as IFireProjectile for firing projectiles, or IBuff/Debuff for applying a self-buff or enemy debuff. These classes will allow us to define different units using the same base class by extending its behaviour in different ways.

Liskov Substitution Principle

Our class hierarchy is created based on the LSP. For example, Unit class is only able to receive hits, while its subclass Melee is able to both receive hits and block enemies. This allows for situations such as in Samurai, where we are able to refer to any subtype of Unit because we are sure it responds to hits within Unit's expectations.

Interface Segregation Principle

In our proposed diagram, we define many different interfaces with their own specific responsibility. When creating tower classes, we only implement interfaces that are relevant to the tower's role - such as the IBlock interface that allows melee towers to block enemies. Ranged towers do not implement this as they do not block enemies.

Dependency Inversion Principle

The proposed ITargeting interface is an example of an abstraction - it only defines the action of picking a target, not how the target is picked. A tower only needs to ensure that its targeting class returns a target, allowing for modifications to the targeting logic without requiring changes to the tower class, leading to less coupling.

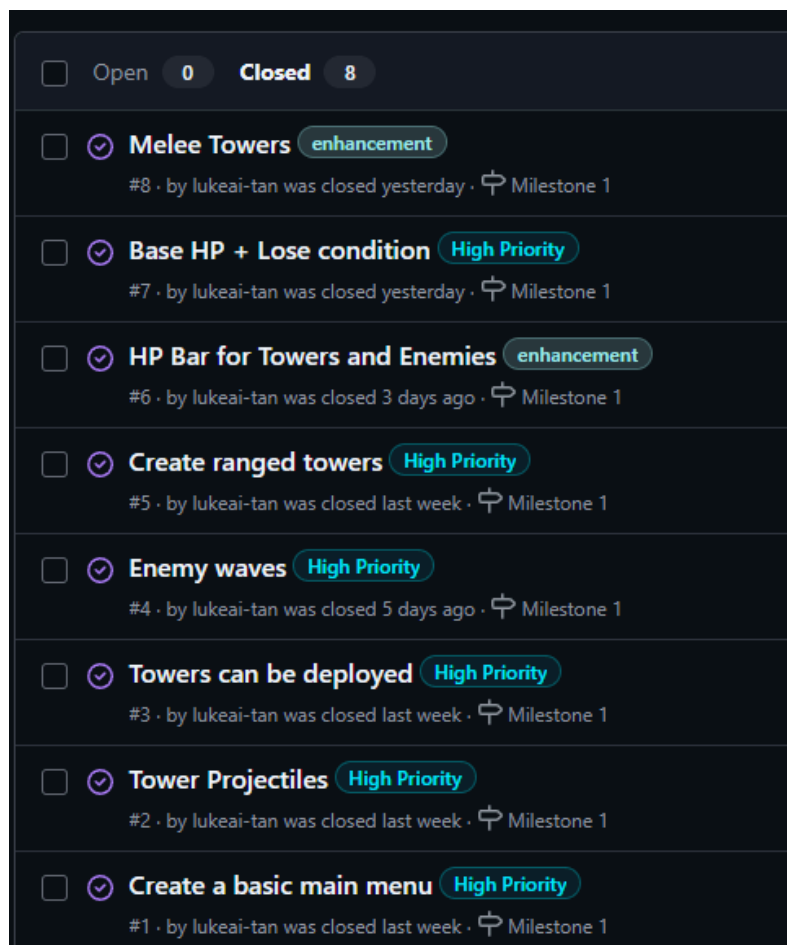
Project Management

Agile Methodology

Each milestone is completed in the form of 4 week sprints, where we detail the core features and extensions to be implemented in each milestone using a feature timeline. Although the milestone deadlines were set by the Orbital program, we still maintained internal agility by breaking down milestones into weekly goals, continuously integrating and testing new features, as well as using GitHub issues, labels and milestones to track task completion









Issues / Labels / Milestones


To effectively manage development and ensure progress aligned with our goals, we utilised GitHub's project management tools, specifically milestones, issues, and labels, in conjunction with our outlined feature timeline.



















Version Control

Using Git, we commit changes frequently and merge them into the main branch before pushing to GitHub. This workflow ensures continuous integration, minimises merge conflicts and supports efficient team collaboration.

Update Map1.tscn	lukeai-tan committed 6 hours ago	1e38ad9		
updated tower and enemy stats	lukeai-tan committed 6 hours ago	2bbf944		
Update RangedTower2.gd	lukeai-tan committed 6 hours ago	98f68d6		
changed stats for Ranged Tower 2	lukeai-tan committed 6 hours ago	f913618		

 Commits on May 29, 2025

changed MeleeTower1 to MeleeTower0 	lukeai-tan committed yesterday	a9e19df		
added attack animation for samurai and rocket samurai	lukeai-tan committed yesterday	f27a96e		
added Melee Towers to Build Mode + unit tests	lukeai-tan committed yesterday	709988e		
Base enemy attack method 	Zmzzz3 committed 2 days ago	e93b5c9		
added Unit class and Obstacle class 	Zmzzz3 committed 2 days ago	4c969c7		
Merge branch 'main' of https://github.com/lukeai-tan/nanban-requiem-orbital	Zmzzz3 committed 2 days ago	962cab0		
changes to targeting logic 	Zmzzz3 committed 2 days ago	5c8836c	