

STUDENT, Justa

B00007I@student.staffs.ac.uk

**[Outpost Zero]**

Technical Design Document

Contents

[**Project Introduction** 1](#_Toc144997739)

[Project Goals 1](#_Toc144997740)

[Challenges and Risks 2](#_Toc144997741)

[Hardware Requirements 2](#_Toc144997742)

[**Platforms** 2](#_Toc144997743)

[Target Platform 2](#_Toc144997744)

[Engine Specific Specifications and Limitations 2](#_Toc144997745)

[Engine Summary 2](#_Toc144997746)

[**Systems and Diagrams** 2](#_Toc144997747)

[System 1 2](#_Toc144997748)

[System 2 2](#_Toc144997749)

[System 3 2](#_Toc144997750)

[**Optimisation and Profiling** 2](#_Toc144997751)

[Profiling Systems 3](#_Toc144997752)

[Profiling Graphics 3](#_Toc144997753)

[Profiling Network/Multiplayer (If Applicable) 3](#_Toc144997754)

[**Coding Standards** 3](#_Toc144997755)

[Programming Standards 3](#_Toc144997756)

[Style Guide 3](#_Toc144997757)

[Commenting Rules 3](#_Toc144997758)

[Code Review Procedures 3](#_Toc144997759)

[**Production Overview** 3](#_Toc144997760)

[Moscow 3](#_Toc144997761)

[Timeline 3](#_Toc144997762)

[Budgeting 4](#_Toc144997763)

# Project Introduction

Outpost Zero is a small space station builder game where the goal is to build your station in a way that makes it withstand solar storms and more. The player must unlock new station parts and modifiers using the tech tree system. They will have to manage resources like science, electricity, water, ammo and crew. Each module will provide and consume different resources allowing for some strategic gameplay, you’ll need enough of all of these to survive the disasters. The main goal is to survive for as long as you can, changing your strategy each time.

## Project Goals

The goal for this project is to create a few systems that I haven’t yet created (like a building system and a tech tree) and create these in an efficient and easily scalable manner. I’d also like to explore parts of the engine I’ve yet to try like using data assets and further refining my ability to create thematic UI elements. There should be around 1-5 minutes of gameplay depending on the players skill level, however this can easily be increased or decreased by altering a few gameplay elements.

## Challenges and Risks

The biggest problem I’ll face with this project is to do with the fact I’ve never created any systems like this before. As all of this will be new to me, I need to ensure that I can still deliver the project on time and don’t plan too many things to implement. Although developing new systems may bring some unexpected challenges, I also know that it’ll further improve my UE5 skill level which will come in handy as a games developer.

## Hardware Requirements

**The PC specs that will be used to develop this prototype are:**

Windows 11 (64bit)

4.20Ghz Processor

48GB RAM

RTX 3060 Ti (8GB VRAM)

**The PC specs recommended to play this prototype are:**

Windows 10 (64bit)

2Ghz Processor

4GB RAM

GTX 960 (2GB VRAM)

750MB Available Storage

# Platforms

## Target Platform

This game prototype is both made and tested on and intended for PC users. For this project I’ll only be adding support for PC keyboards and mouse input and none for console. This is primarily because I want to focus on the systems and mechanics themselves as opposed to getting it to work on multiple platforms. I also don’t own a console or controller which would make it very hard to test.

## Engine Specific Specifications and Limitations

As this is only a small prototype, I don’t need to worry too much about disk space when it comes to limitations. That said I do have to keep in mind that the prototype should be well optimised as that will be one of my main focuses for this project. One of the main areas where I need to specifically look out for optimisation would be in the building system itself. This should primarily be made using two blueprints a component and a building blueprint. With this method I should get away with only needing to cast once on begin play. This is a good starting point as removing the obvious bad practices will help keep this project well optimised.

Other areas where I’ll look out for are not using the ‘On Tick’ node, not having timer events happen too often, not using ‘Get All Actors’ nodes and finally avoiding delay nodes. Most things mentioned here can be replaced by other methods which are seen as a much better practice when using blueprints.

## Engine Summary

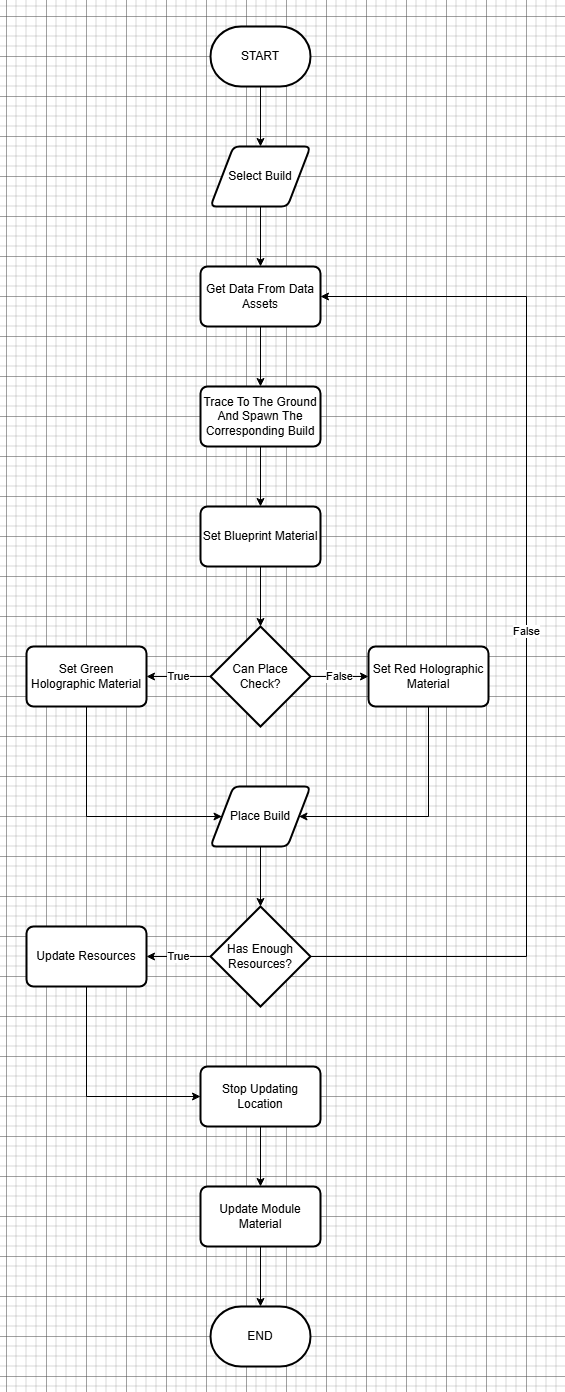
For this game prototype, I’ll be using the default UE5 editor with no plug-ins. Using Unreal Engine 5.4.4 as this is the editor used by the University as well as the one I have installed on my home PC.

# Systems and Diagrams

## System – Building System

The building system for this prototype is like what would be used for a city builder. The player would select a build first using some buttons on the HUD, once this is selected the build would follow the user’s mouse while snapping to the grid. The modular space station parts will be made with the right scale so that when any part is built within the grid, it will look like it’s connected to the stations docking ports. The player can also press ‘R’ which will rotate the selected part by 90 degrees, this can only be done before it is placed down. Finally, there should be a series of checks carried out to make sure that it can only be placed when next to another sections docking port. One check will make sure that the selected part isn’t touching the station, another check will make sure that there is another station part directly next to the docking port. This should block the player from placing a build in an unintended area.

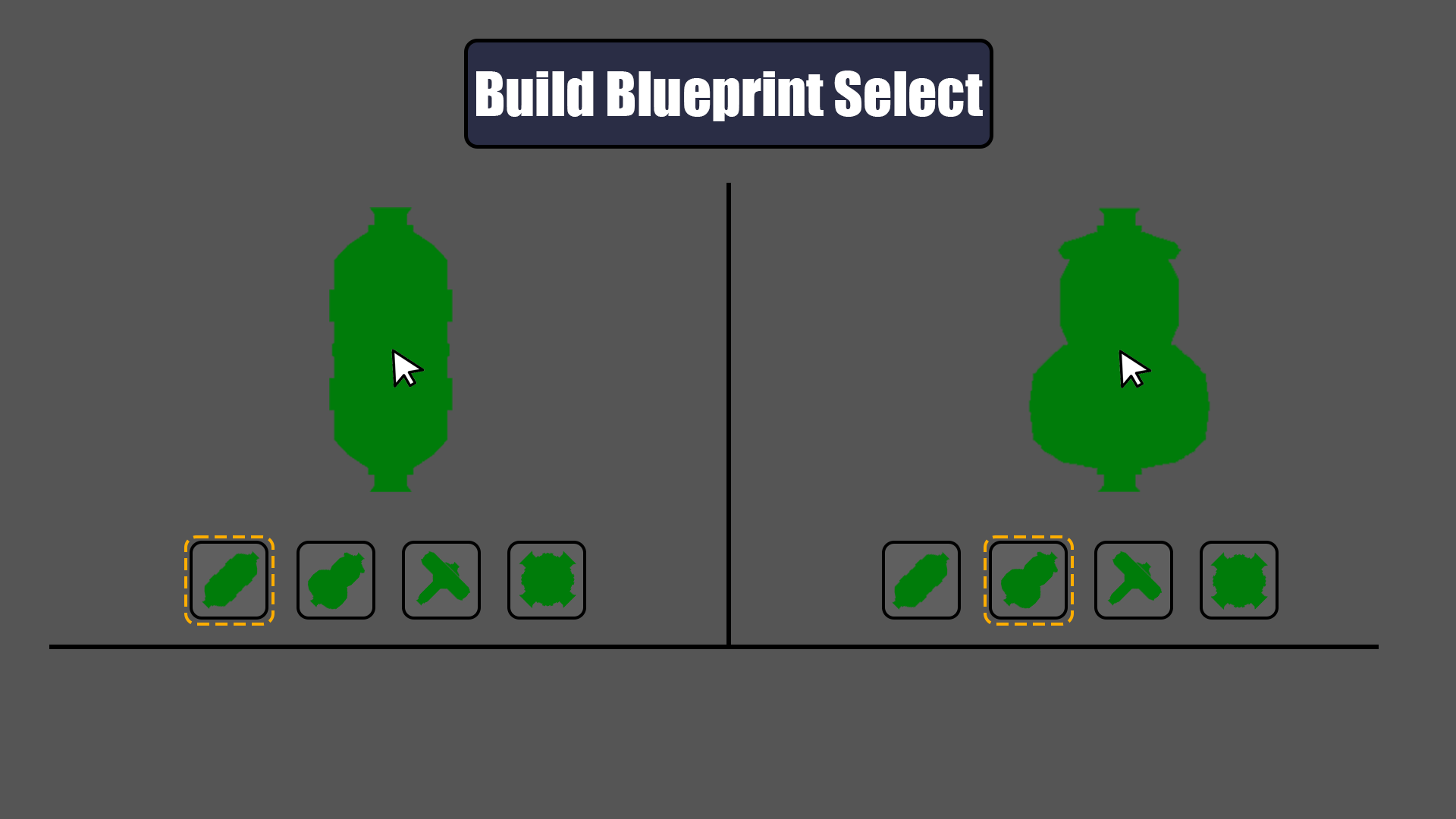
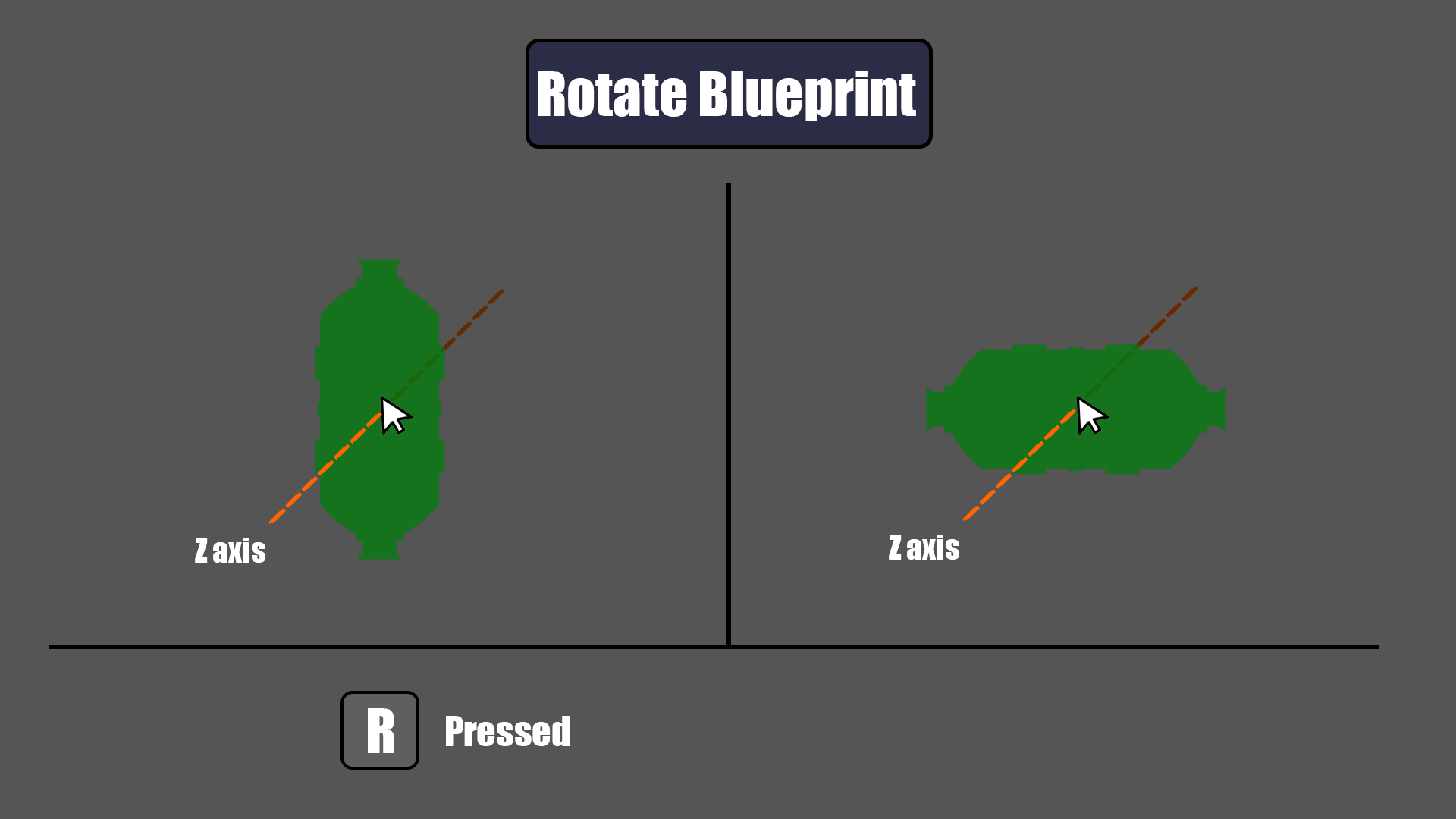
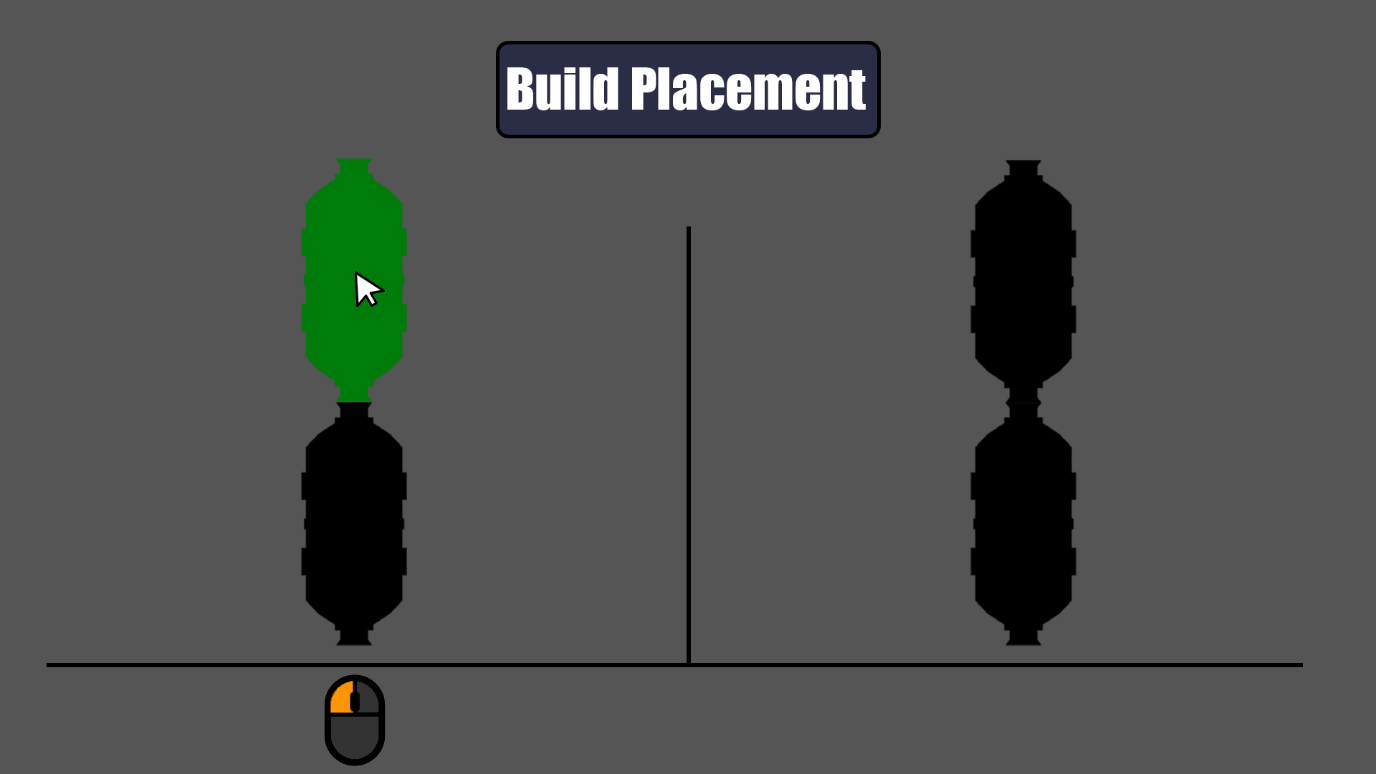
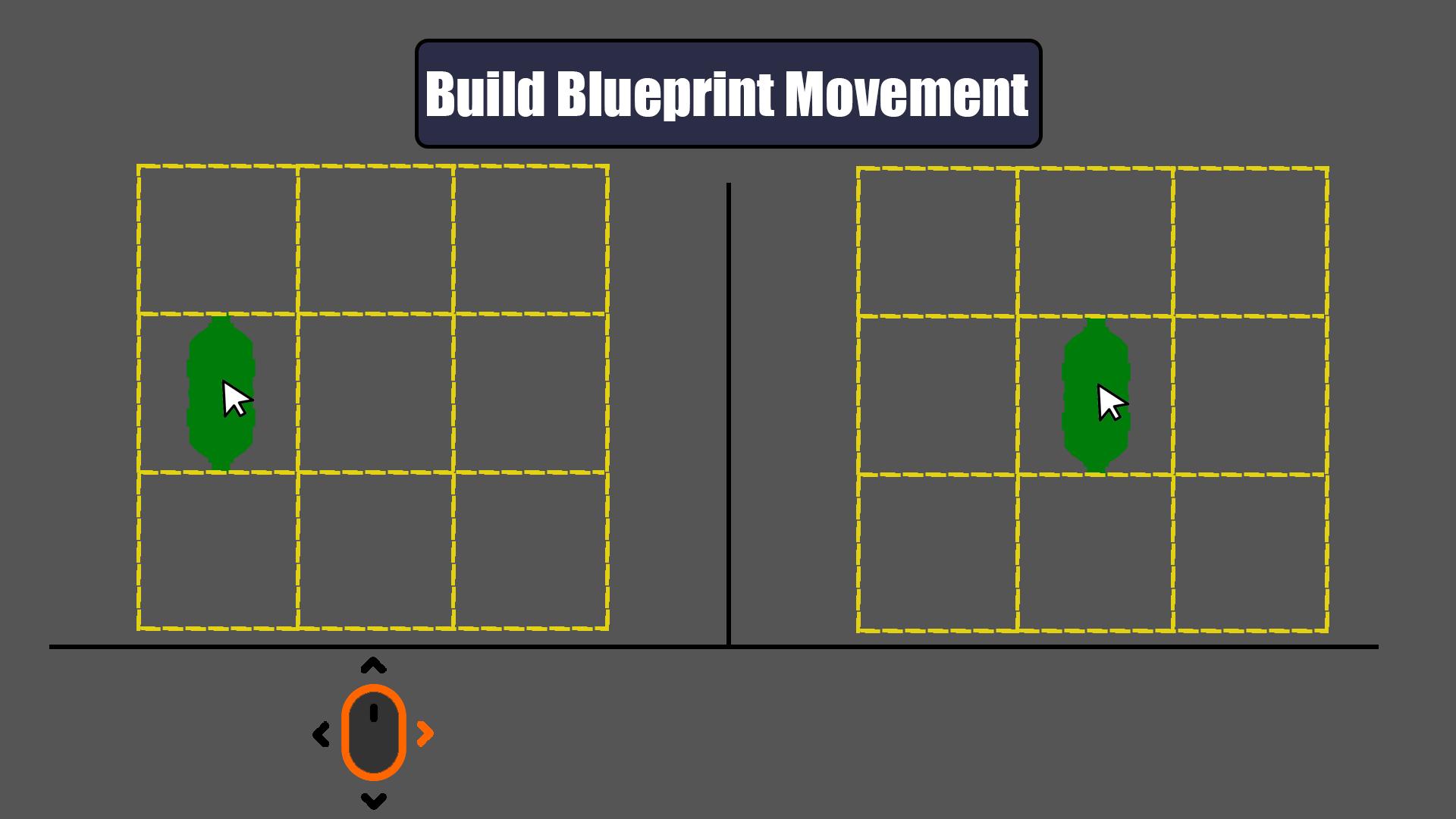
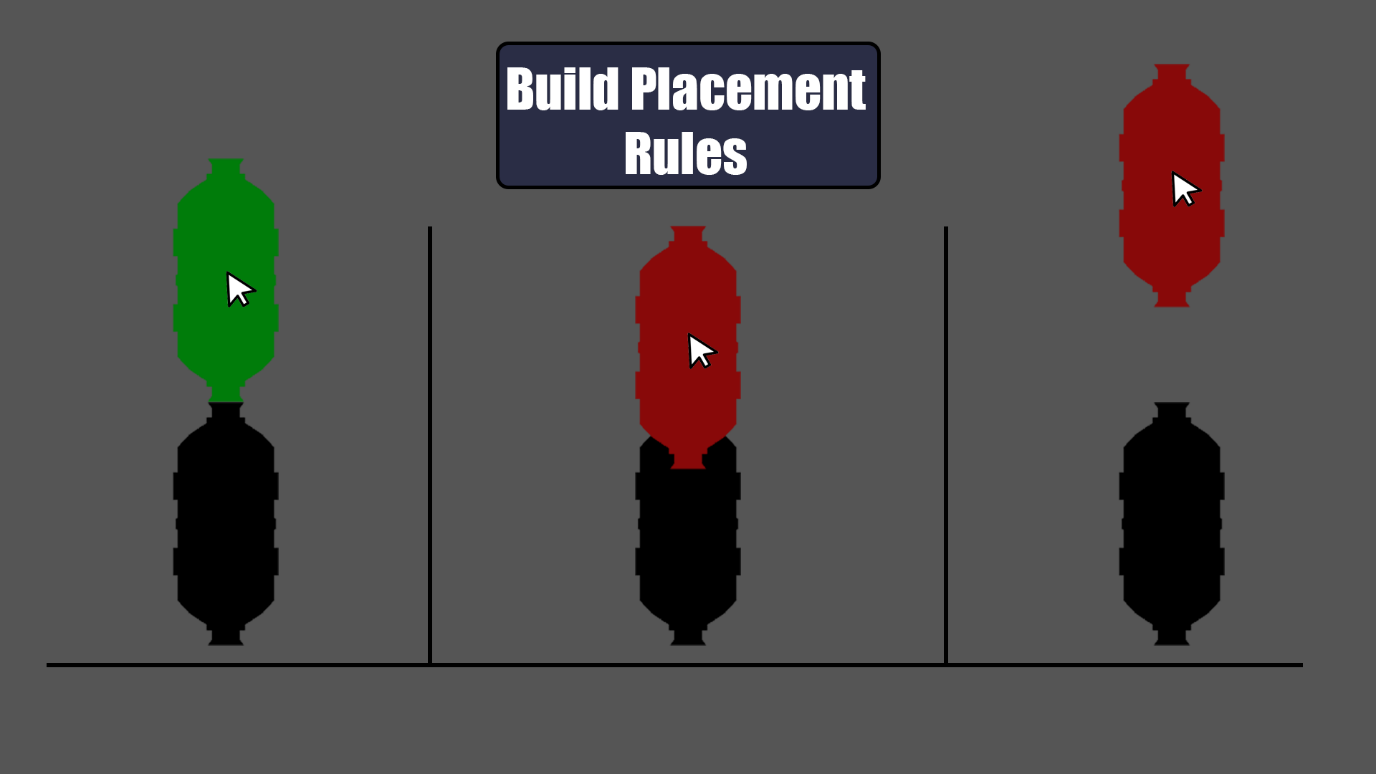
**Build System**

****

**Can Place Check**

**A diagram of a multiply module

AI-generated content may be incorrect.**

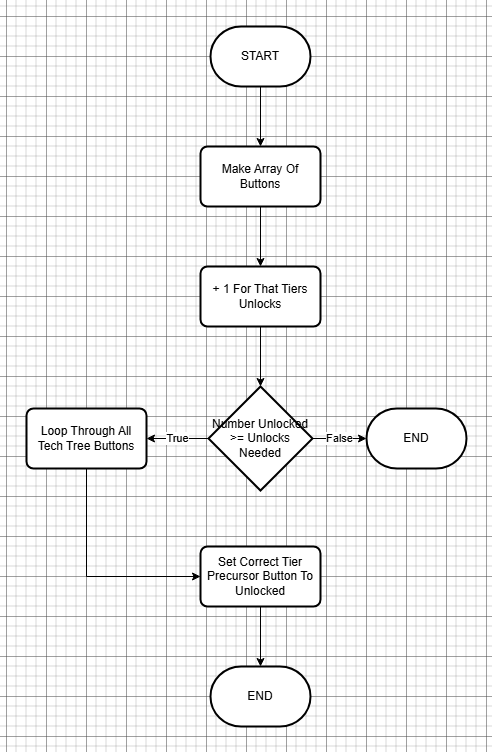


## System – Tech Tree

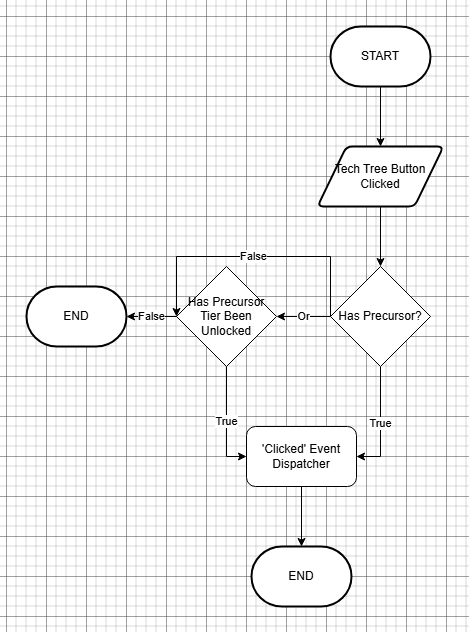
Another system in this prototype will be the tech tree. This will serve to prolong gameplay and prevent the player from having access to everything the game has to offer right away. The idea is that you’ll start with the basic modules needed and the rest I’ll be unlocked via the tech tree. Not only that, but the tech tree will also have buffs you can unlock like increased power production or decreased water consumption which will help the player grow the station.

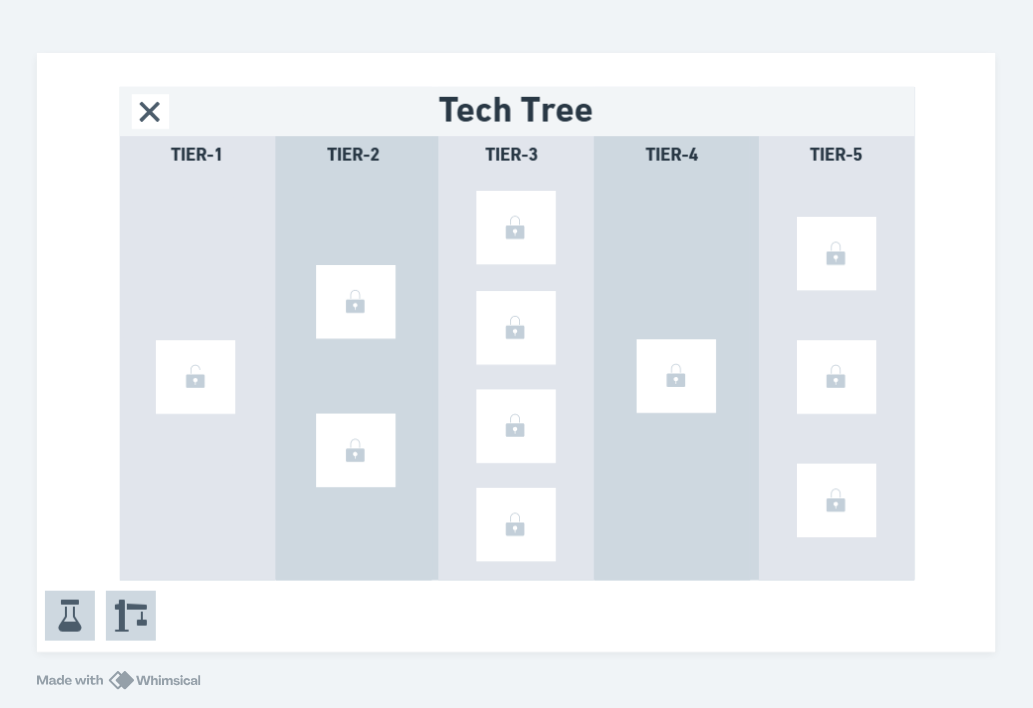
To unlock a tech tree tier, you’ll need to unlock all the previous sections that are connected to the one you’d like to connect. This system will also require only one style of buttons, which will then be altered in engine for all the different states.

**Tech Tree Panel**

****

**Tech Tree Button**

****

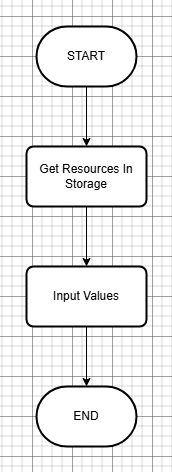


## System – Resource Management System

This prototype will also include a resource management system. The players resources will be displayed at the top of the screen. These resources will change based on the modules you’ve built onto the space station. A tooltip will allow the player to see what resources that specific module uses and produces. If the player has enough resources, then once the module is built the ‘used’ resources will be removed and the ‘produced’ resources will be added onto the player’s total. If there aren’t enough resources, then the player won't be able to place that selected build down until all resources needed have been obtained.

Resources can also be altered through the tech tree. Some tech tree unlocks will allow modules to produce more of a specific product making everything much more efficient. This will also affect any modules already placed.

**Update Resources UI**



*A close-up of a card

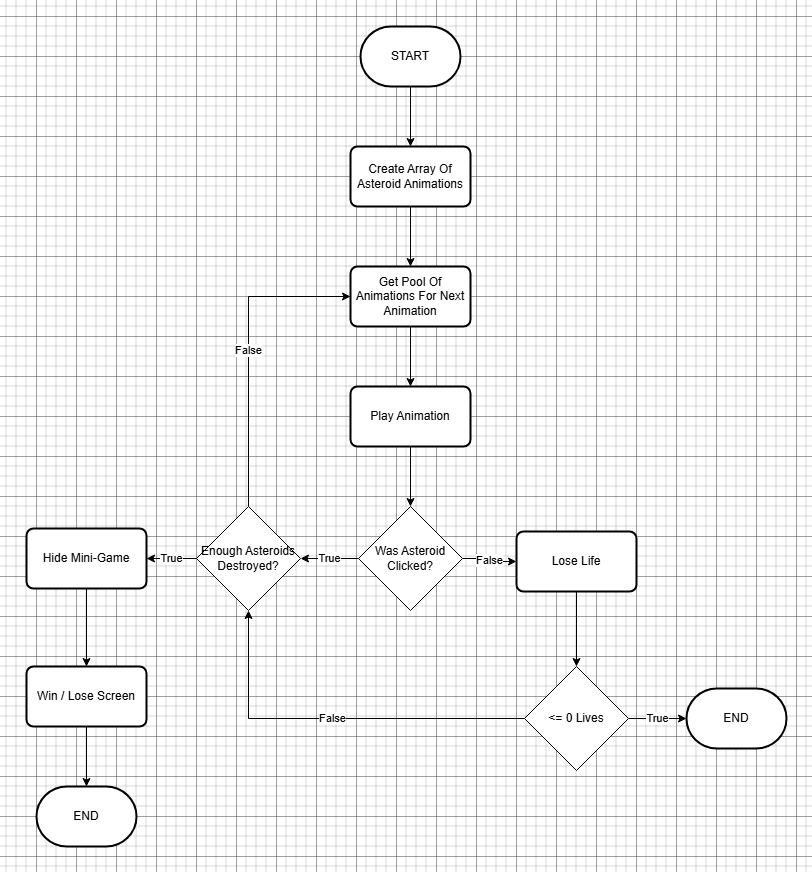
AI-generated content may be incorrect.*

A screenshot of a video game

AI-generated content may be incorrect.

## System – Asteroid Mini-game

After a timer runs out, I’d want the player to face some sort of challenge (a reason to build up the space station). An asteroid mini game like among us would fit my game perfectly. The way this would work is I would create 3 different animations for 3 different buttons (asteroids). Then I’d get a random animation from the pool of 3 (excluding the previously used one) and play it. If the player loses all their lives, then the game ends and they must restart. If they hit the asteroid, then another one will spawn until the required number of asteroids have been destroyed. There will be three rounds, each round spawning one extra asteroid meaning that in round 3 there will be three asteroids on screen at once. After each round is completed a message will appear that congratulates the player, until the last round where the game complete screen will be displayed.



## HUD

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Quick Points – Buildable

**Resource – Output**

**A table with text and numbers

AI-generated content may be incorrect.**

**Resource – Input**

**A table with text and numbers

AI-generated content may be incorrect.**

**Tech Tree – Info**

A table with numbers and text

AI-generated content may be incorrect.

# Optimisation and Profiling

## Profiling Systems

While developing a system or mechanic I will continuously check that it works in the way I have intended it to. After it’s at a point I’m happy with I’ll attempt to break the system through different methods like spam clicking or using it in a way it wasn’t intended to be used in. Once I have ironed out the issues I have found, I will conduct a slightly larger scale testing session where I gather feedback from other users and see if any new bugs arise. This is also a great opportunity to gain feedback on how the systems or mechanics could be improved upon.

## Profiling Graphics

When using the UE5 FPS overlay I aim to be hitting a minimum of 60 fps, this is to ensure smooth gameplay and gives me a good idea of when my project is leaning more towards the unoptimized side. I’ll ensure to test this around once every two weeks to ensure that it’s constantly preforming well.

# Coding Standards

## Programming Standards

I will be sticking, as closely as possible, to the standard UE naming conventions when creating new elements which can be found [*here.*](https://dev.epicgames.com/documentation/en-us/unreal-engine/recommended-asset-naming-conventions-in-unreal-engine-projects) Additionally, when it comes to naming different events, functions and comments, I will use names that describe what that specific code does. Comments should also be coloured based on what that chunk does for example core events – Red, UI – Brown, Functions – Purple and other smaller chunks will be coloured and grouped off as deemed appropriate.

As mentioned earlier, to make sure my code is efficient I will avoid using the basic bad practices as much as possible like ‘Event Tick’, ‘Get All Actors’ and ‘Cast’ nodes. In addition, I’ll use functions for code that will often be repeated and or called from within other blueprints, Macros will be used for condensing repeating code but only called from the same blueprint.

## Style Guide

I will ensure that each function, event or comment that I create will be named appropriately so that anyone could read it and make out roughly what that system does. I’ll also ensure that my nodes are aligned neatly as spaghetti code is hard to read as well as debug. If I ensure that things are kept neat and tidy, then the project will be a lot easier to finish as well as debug.

## Commenting Rules

As mentioned previously, all code should be commented with a basic heading that describes what that chunk of code does. If something is more complex, then there should be a brief description that helps other people understand what is being done. All comments should be colour coded to help the programmer quickly and at a glance see where they need to look when de-bugging.

## Code Review Procedures

Similar to testing my projects performance, I’ll constantly be trying to create my systems in the best way I know. That said, I’ll also ask my peers if I’m uncertain about how to go about a system in the best way possible. Additionally, I hope to get a lecturer to glance through my project at least two times throughout development. Ensuring that I’m not going too far off standard procedure.

# Production Overview

## Moscow

|  |  |  |  |
| --- | --- | --- | --- |
| Must | Should | Could | Won’t |
| Grid Based Build System | Tech Tree | VFX & SFX | Moveable Camera |
| Rotate/Zoom Camera | Different UI For Different Modes | Camera Shake |  |
| Resource Management | Tool Tips | Asteroid Mini-Game |  |

## Timeline

|  |  |  |
| --- | --- | --- |
| Task | Start Date | Completion Date |
| Initial Research / Ideas | 09/12/24 | 20/12/24 |
| Building System | 23/12/24 | 10/01/25 |
| Resource Sytem | 10/01/25 | 17/01/25 |
| Basic UI | 15/01/25 | 20/01/25 |
| Basic Polish | 20/01/25 | 24/01/25 |
| First Testing | 20/01/25 | 24/01/25 |
| Tech Tree | 27/01/25 | 06/02/25 |
| Asteroid Mini-Game | 06/02/25 | 16/02/25 |
| Audio | 11/02/25 | 18/02/25 |
| Second Testing | 16/02/25 | 18/02/25 |

## Budgeting

This project will be 100% as I’m creating most of the 3D assets. As well as this when looking for audio, icons and fonts I’ll ensure to find some that are free for personal and commercial uses. I’ll make sure to spend a minimum of 30-40 hours a week on this project where my time will be split between blueprinting, debugging, research and planning work.