Version 1.1 2024/25

University of Staffordshire

Feudal Living

Technical Design Document

SLASKI, Jack

S013671M@student.staffs.ac.uk

Contents

Project Introduction 2
Project Goals 2
Challenges and Risks 2
Hardware Requirements 2
Platforms 2
Target Platform 2
Engine Summary 3
Mechanics and Diagrams 3
Main Level Game Mode 3
Player Controller 3
Camera Pawn 4
Needs 5
Skills 6
Interactable Objects 6
Character Data 9
World Time 10
Working / Doors 10
Interactable Manager 11
AI Takeover 11
UI 13
Coding Standards 14
Programming Standards 14
Style Guide .... **Error! Bookmark not defined.**
Commenting Rules 14
Code Review Procedures 15
Production Overview 15
Moscow 15
Timeline 16

Project Introduction

The idea for this game is that it will be a 'Sims-like' medieval life simulator named "Feudal Living". In this game, the player will take control of a character who is living in medieval England and attempt to control their life by managing their needs, improving their skills, earning them money, and designing their home.

Project Goals

The ultimate goal for this project is to have a solid and functional foundation for a medieval life simulator game. This foundation could then act as a base for possible future expansion and project creation. By having a solid foundation that is modular, easy to understand, and efficient, future work and expansion can be made very easy.

Challenges and Risks

The challenges this project is aiming to solve are as follows:

- Having modular interactable objects that can easily be created to solve multiple needs and skills.
- Having and AI system that roams and interacts with objects on their own.
- Having a way to earn and spend money.
- Spawning, building, placing, selling, and moving interactable and decorative objects around the world.
- Having multiple different types of objects.
- Having time controls that can stop, play, and fast forward time.
- Having a custom world time that gives the user some more world context.
- Having a realistic environment, objects, needs, and skills for the time period.
- Easy to use and concise camera and player controls.

Hardware Requirements

Recommended Hardware:

Operating System: Windows 10 64-bit version 1909 revision .1350 or higher, or versions 2004 and 20H2 revision .789 or higher

Processor: Quad-core Intel or AMD, 2.5 GHz or faster

Memory: 32 GB RAM

Graphics RAM: 8 GB or more

Graphics Card: DirectX 11 or 12 compatible graphics card with the latest drivers

Platforms

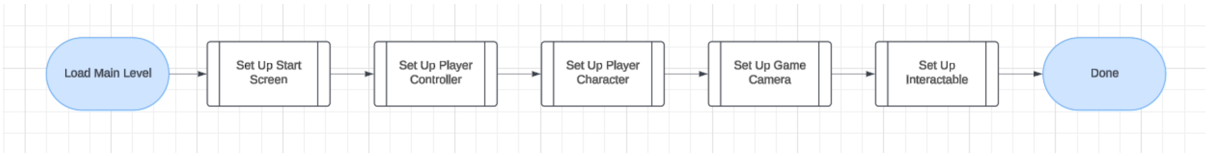
Target Platform

The target platform for this project is PC as the project's controls can be complex and mostly requires a mouse cursor as well as multiple keyboard controls. Although, in the future a console version could be made through the use of a console cursor and smart console control mapping.

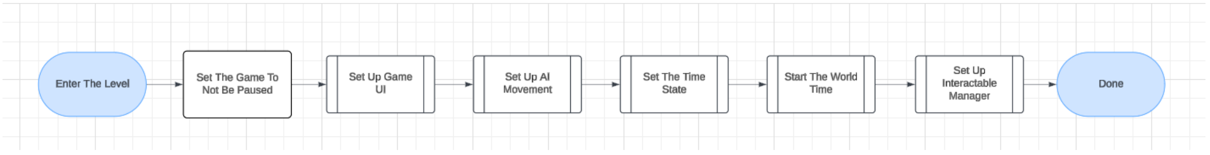
Engine Summary
This project was created in Unreal Engine Version: 5.5 with no additional plugins used.

Mechanics and Diagrams

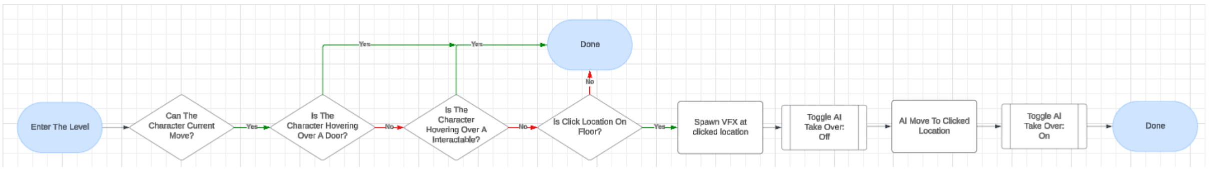
Main Level Game Mode
Here is a flowchart showing the logic of the main game mode loading the main level:



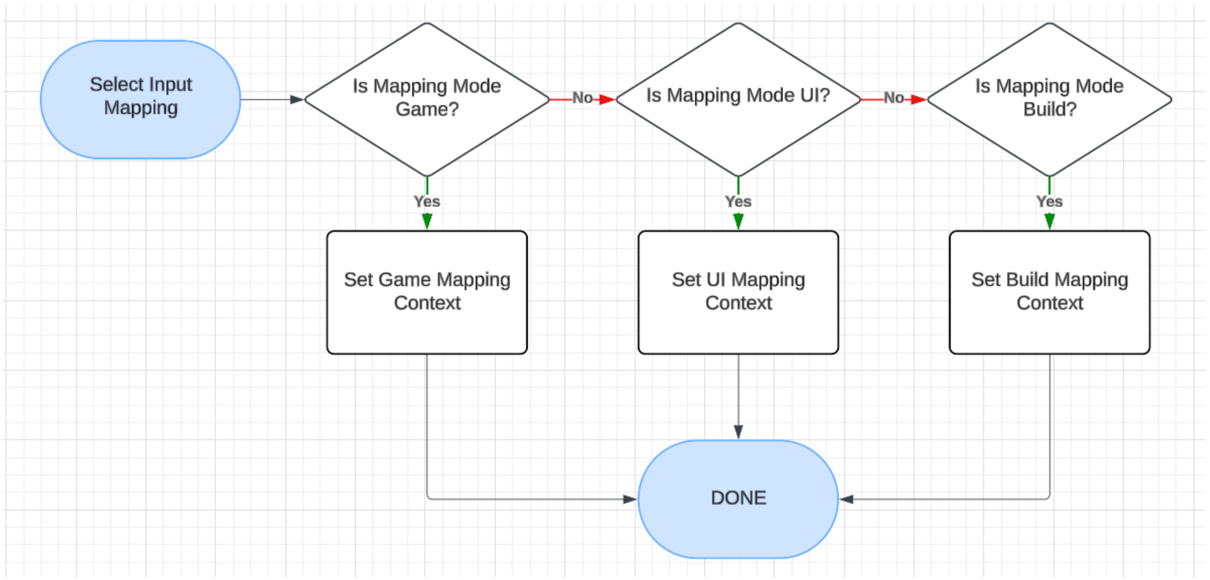
Here is a flowchart showing the logic of entering the main level in the game mode:



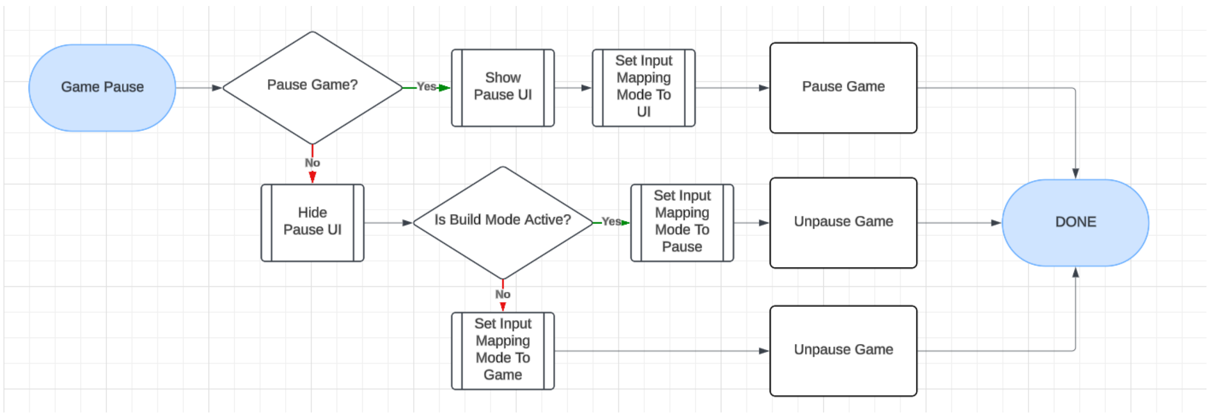
Player Controller
Here is a flowchart showing the logic of a character clicking in the world:


Enter The Level
Can The Character Current Move?
Yes
Is The Character Hovering Over A Door?
Yes
No
Is The Character Hovering Over A Interactable?
Yes
No
Is Click Location On Floor?
No
Done
Yes
Spawn VFX at clicked location
Toggle AI Take Over: Off
AI Move To Clicked Location
Toggle AI Take Over: On
Done

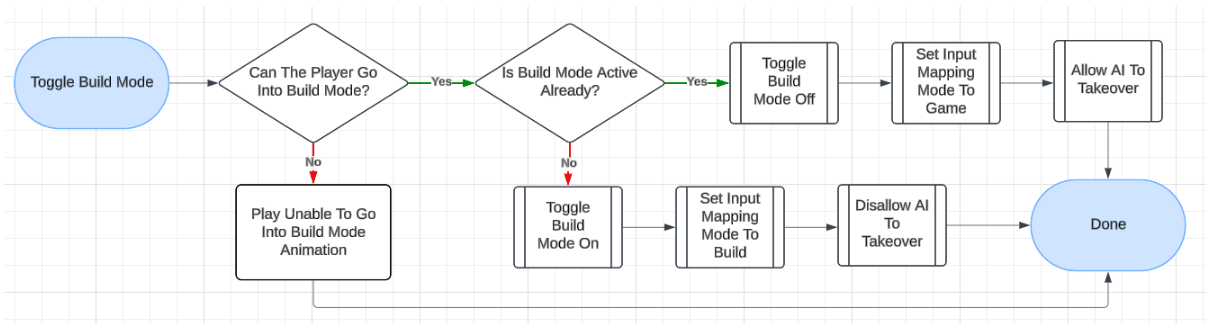
Here is a flowchart showing the logic of selecting an input mapping context:



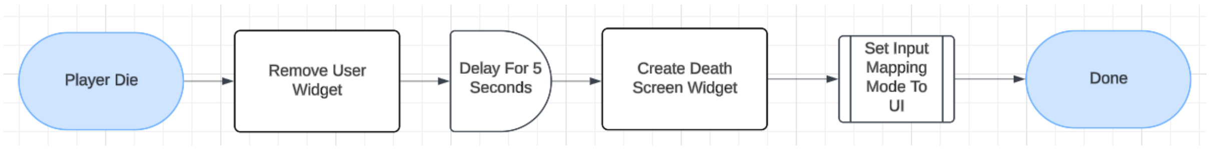
Here is a flowchart showing the logic for pausing the game:



Here is a flowchart to show the logic of toggling build mode:

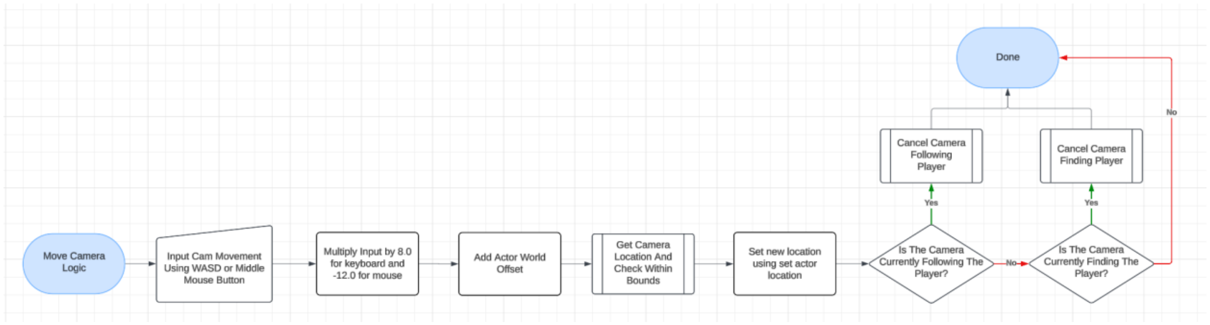

Toggle Build Mode
Can The Player Go Into Build Mode?
Yes
Is Build Mode Active Already?
Yes
Toggle Build Mode Off
Set Input Mapping Mode To Game
Allow AI To Takeover
No
Play Unable To Go Into Build Mode Animation
No
Toggle Build Mode On
Set Input Mapping Mode To Build
Disallow AI To Takeover
Done

Here is a flowchart to show the logic of a player dying:

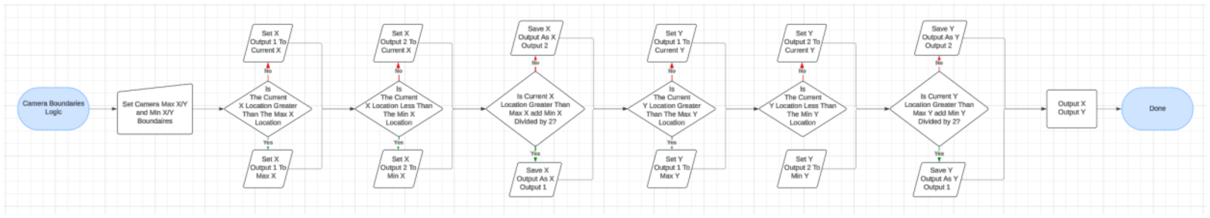


Camera Pawn

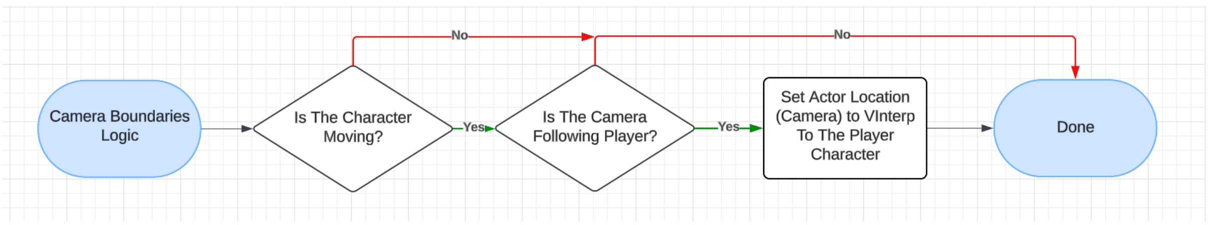
Here is a flowchart showing the camera movement logic:


Done
No
Cancel Camera Following Player
Cancel Camera Finding Player
Yes
Yes
Move Camera Logic
Input Cam Movement Using WASD or Middle Mouse Button
Multiply Input by 8.0 for keyboard and -12.0 for mouse
Add Actor World Offset
Get Camera Location And Check Within Bounds
Set new location using set actor location
Is The Camera Currently Following The Player?
No
Is The Camera Currently Finding The Player?

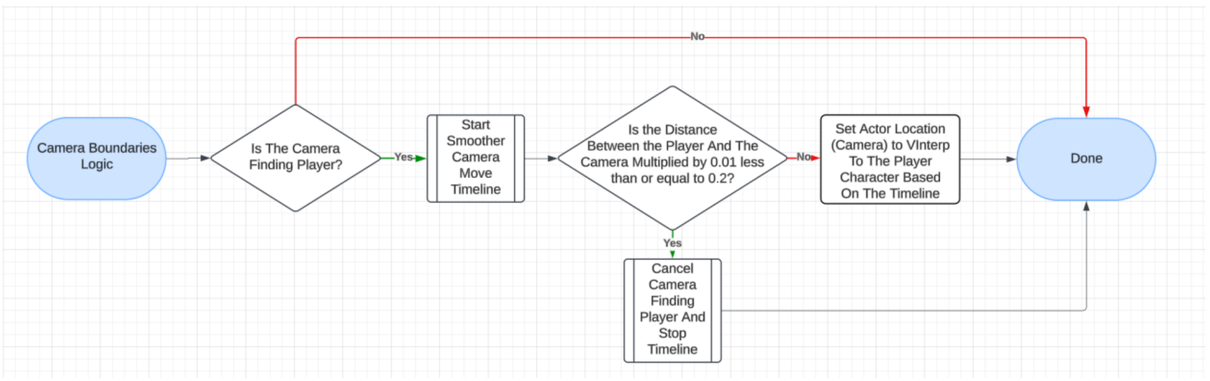
Here is a flowchart showing the camera boundaries logic:


Camera Boundaries Logic
Set Camera Max X/Y and Min X/Y Boundaires
Set X Output 1 To Current X
No
Is The Current X Location Greater Than The Max X Location
Yes
Set X Output 1 To Max X
Set X Output 2 To Current X
No
Is The Current X Location Less Than The Min X Location
Yes
Set X Output 2 To Min X
Save X Output As X Output 2
No
Is Current X Location Greater Than Max X add Min X Divided by 2?
Yes
Save X Output As X Output 1
Set Y Output 1 To Current Y
No
Is The Current Y Location Greater Than The Max Y Location
Yes
Set Y Output 1 To Max Y
Set Y Output 2 To Current Y
No
Is The Current Y Location Less Than The Min Y Location
Set Y Output 2 To Min Y
Save Y Output As Y Output 2
No
Is Current Y Location Greater Than Max Y add Min Y Divided by 2?
Yes
Save Y Output As Y Output 1
Output X Output Y
Done

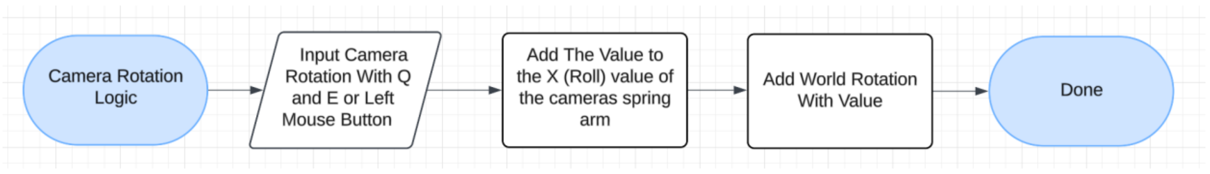
Here is a flowchart showing the camera follow player logic:


Camera Boundaries Logic
Is The Character Moving?
No
Yes
Is The Camera Following Player?
No
Yes
Set Actor Location (Camera) to VInterp To The Player Character
Done

Here is a flowchart showing the camera find player logic:


Camera Boundaries Logic
Is The Camera Finding Player?
No
Yes
Start Smoother Camera Move Timeline
Is the Distance Between the Player And The Camera Multiplied by 0.01 less than or equal to 0.2?
No
Yes
Set Actor Location (Camera) to VInterp To The Player Character Based On The Timeline
Cancel Camera Finding Player And Stop Timeline
Done

Here is a flowchart showing the camera rotation logic:

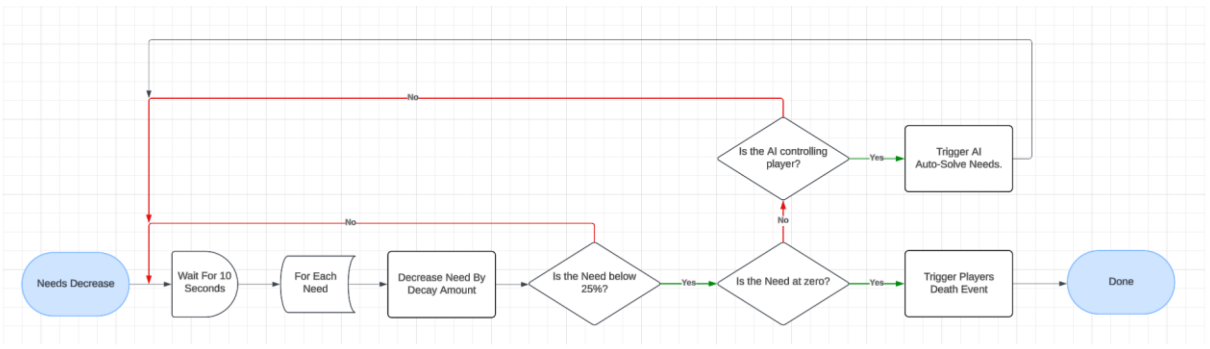

Camera Rotation Logic
Input Camera Rotation With Q and E or Left Mouse Button
Add The Value to the X (Roll) value of the cameras spring arm
Add World Rotation With Value
Done

Needs

Here is a table showing all of the in-game needs and their values.

Need Type	Current Value	Max Value	Decay Amount	Need Colour
Hunger	100	100	1.2	980000FF
Thirst	100	100	1.5	000098FF
Energy	100	100	0.5	988800FF
Hygiene	100	100	1.0	009800FF
Belief	100	100	0.2	4A0098FF

Here is a flowchart showing the process of the characters needs decaying:

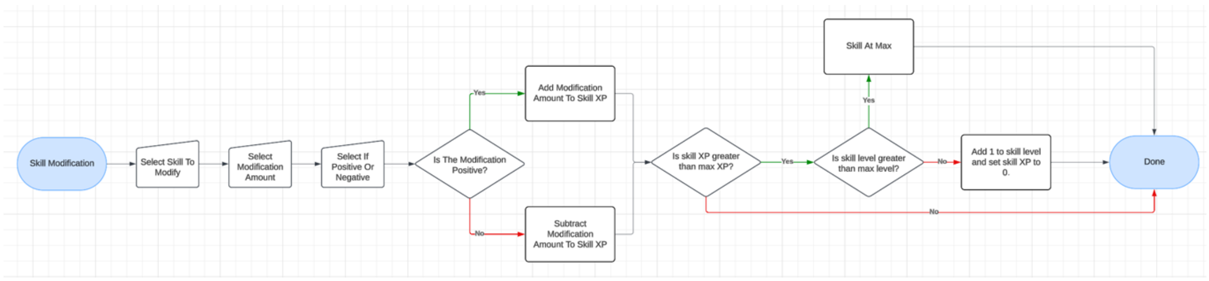

Needs Decrease
Wait For 10 Seconds
For Each Need
Decrease Need By Decay Amount
Is the Need below 25%?
No
Yes
Is the Need at zero?
No
Yes
Is the AI controlling player?
No
Yes
Trigger AI Auto-Solve Needs.
Trigger Players Death Event
Done

Skills

Here is a table showing all of the in-game skills and their data:

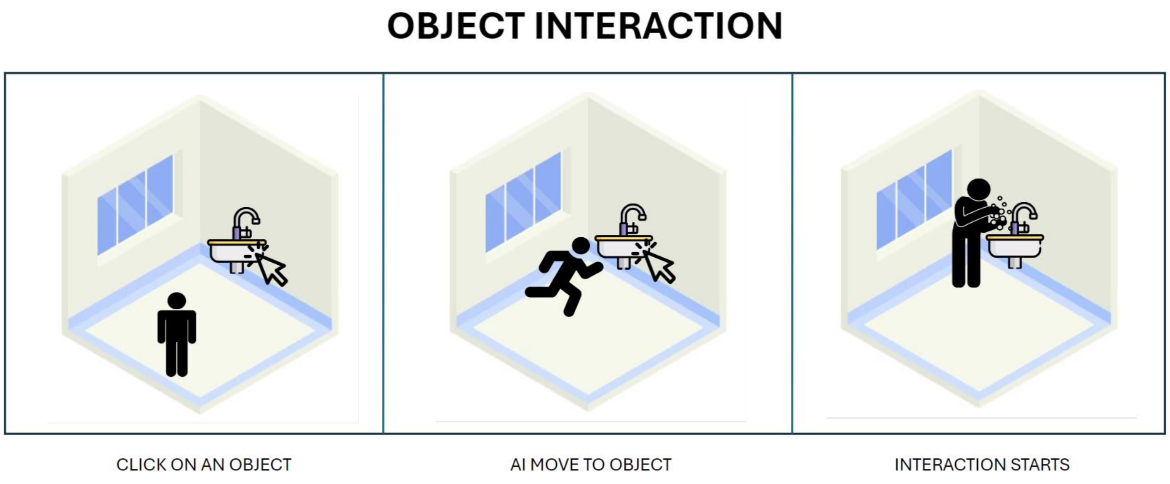
Skill	XP	Max XP	Level	Max Level	Colour
Faith	0	1.0	1	10	230099FF
Plumbing	0	1.0	1	10	000898FF
Crafting	0	1.0	1	10	983900FF
Cookery	0	1.0	1	10	5B0600FF

Here is a flowchart showing the process of modifying the characters skills:


Skill Modification
Select Skill To Modify
Select Modification Amount
Select If Positive Or Negative
Is The Modification Positive?
Yes
Add Modification Amount To Skill XP
No
Subtract Modification Amount To Skill XP
Is skill XP greater than max XP?
Yes
Is skill level greater than max level?
Yes
Skill At Max
No
Add 1 to skill level and set skill XP to 0.
No
Done

Interactable Objects

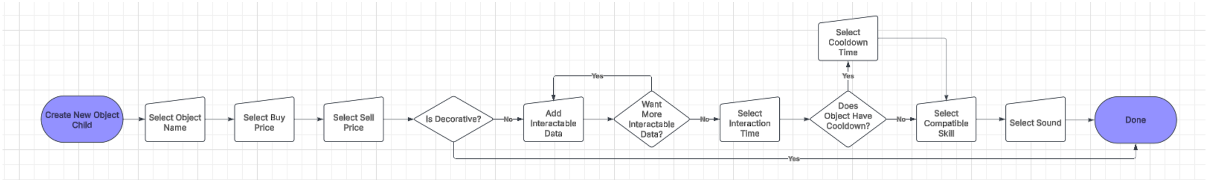
Here is a diagram made to show how the interactable objects will work:



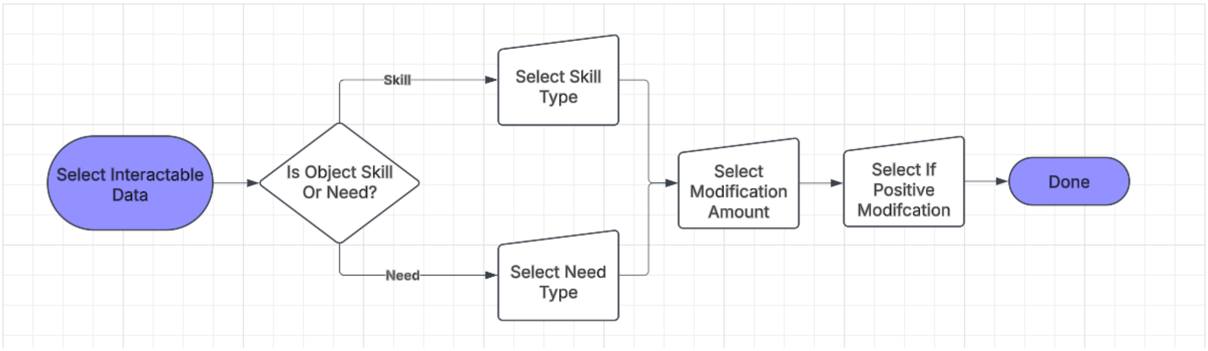
Here is a table depicting the settings that can be changed when creating an object:

Setting	Type	What It Does?
Object Name	String	Sets the name of the interactable.
Is Decorative?	Boolean	Decides if the object is decorative or interactable.
Interactable Data	Structure	Sets the modification data of the interactable.
Interaction Time	Float	Sets the interaction time of the object.
Object Has Cooldown?	Boolean	Decides if the object has a cooldown or not.
Cooldown Time	Float	Sets the cooldown time of the interactable.
Buy Price	Integer	Sets the buy price of the interactable.
Sell Price	Integer	Sets the sell price of the interactable.
Icon	Texture 2D	Sets the icon of the interactable.
Compatible Skill	Enum	Sets the compatible skill of the interactable
Object Sound	Sound Cue	Sets the sound that plays during the object interaction.

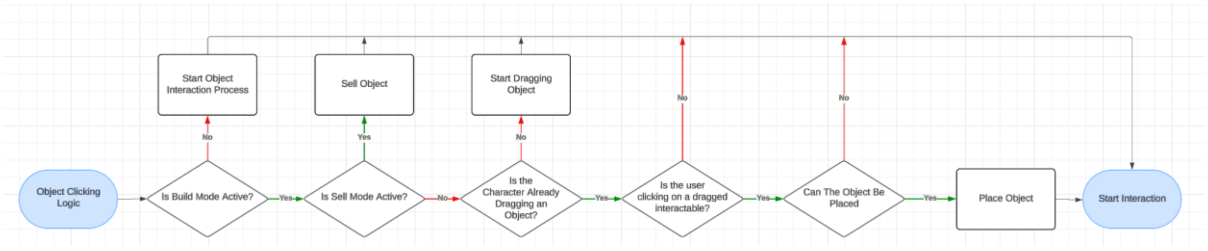
Here is a flowchart showing the process of creating an interactable object during the creation of the game:



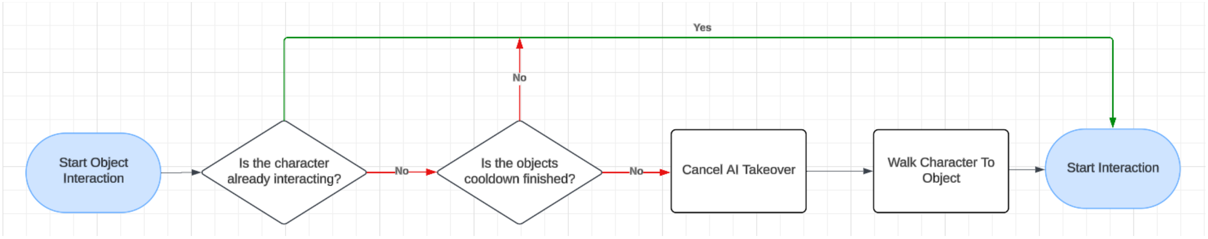
Here is a flowchart to show the process of selected the interactable data for an object during the creation of the game:



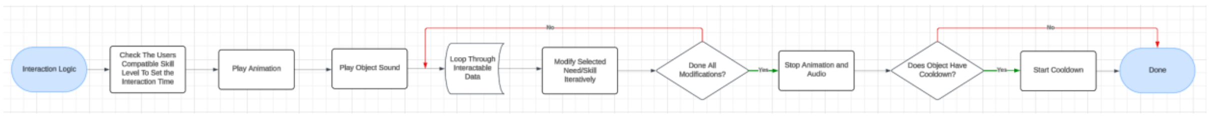
Here is a flowchart to show the logic of the process of when an interactable object is clicked:



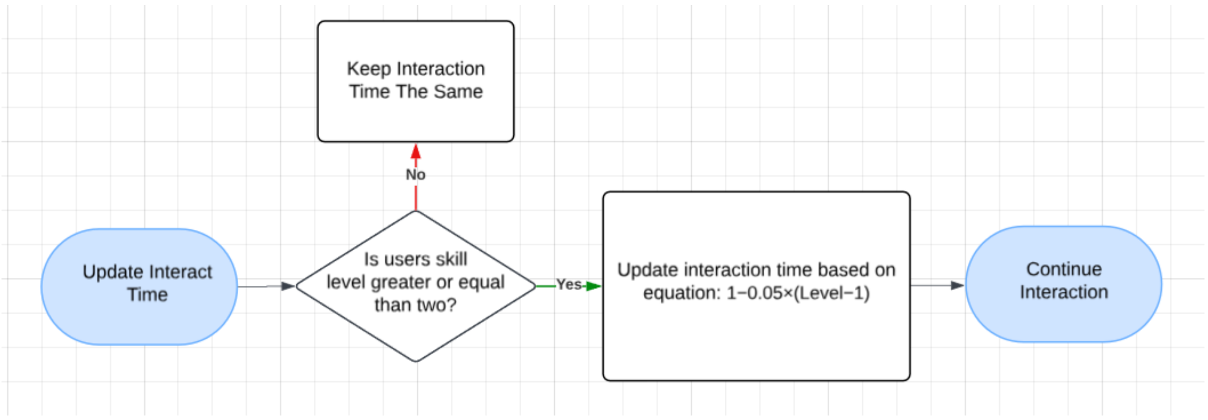
Here is a flowchart to show the logic of the starting the object interaction process:



Here is a flowchart to show the logic of the interaction process with an interactable object:


Interaction Logic
Check The Users Compatible Skill Level To Set the Interaction Time
Play Animation
Play Object Sound
Loop Through Interactable Data
Modify Selected Need/Skill Iteratively
Done All Modifications?
No
Yes
Stop Animation and Audio
Does Object Have Cooldown?
No
Yes
Start Cooldown
Done

Here is a flowchart to show the process of checking a user's compatible skill level and updating the interaction time:

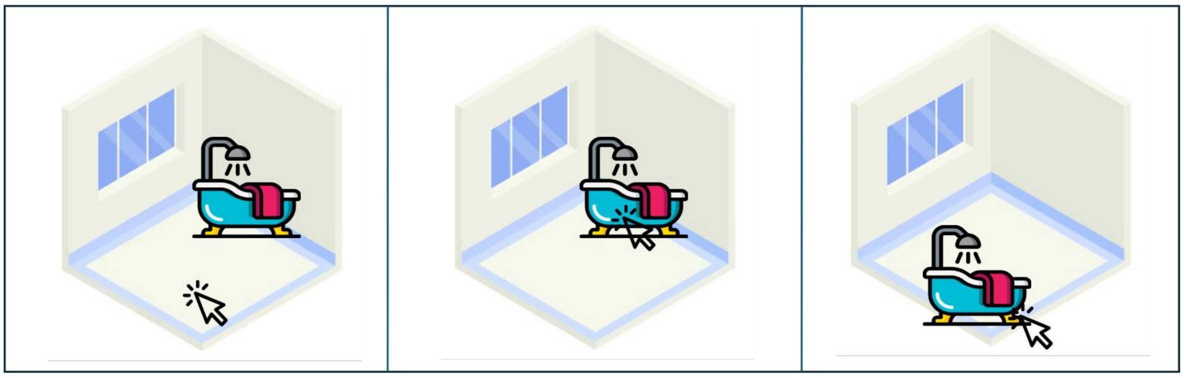

Keep Interaction Time The Same
No
Update Interact Time
Is users skill level greater or equal than two?
Yes
Update interaction time based on equation: 1−0.05×(Level−1)
Continue Interaction

Here is a table showing all of the levels and their converted time:

Level	Equation	% of Full Time
Level 2	1−0.05×(2−1) = 0.95	95%
Level 3	1−0.05×(3−1) = 0.9	90%
Level 4	1−0.05×(4−1) = 0.85	85%
Level 5	1−0.05×(5−1) = 0.8	80%
Level 6	1−0.05×(6−1) = 0.75	75%
Level 7	1−0.05×(7−1) = 0.7	70%
Level 8	1−0.05×(8−1) = 0.65	65%
Level 9	1−0.05×(9−1) = 0.6	60%
Level 10	1−0.05×(10−1) = 0.55	55%

Here is a design showing how object placing works:

OBJECT BUILDING

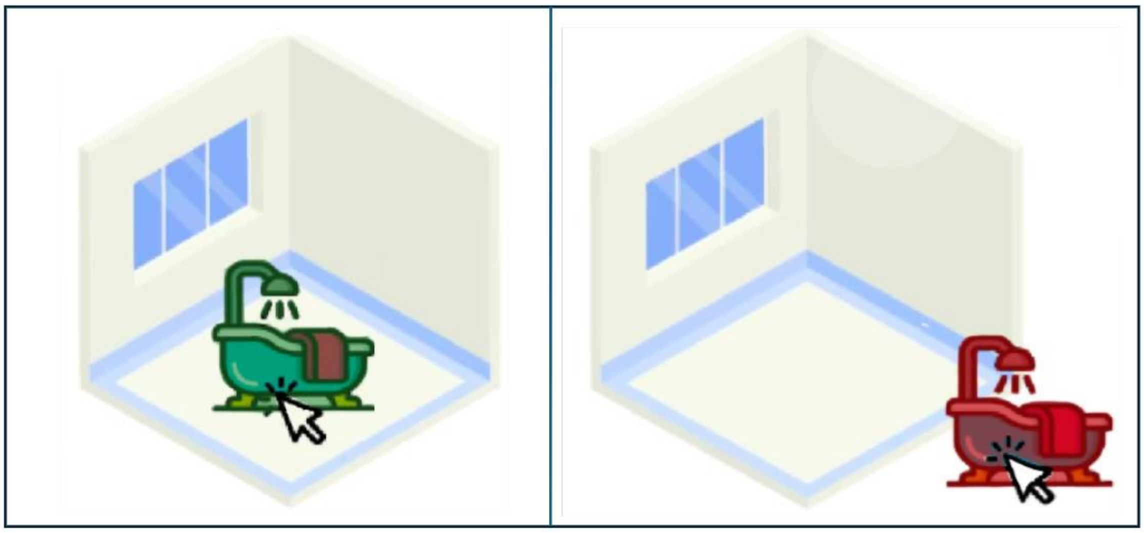


LOCATE OBJECT IN WORLD

CLICK ON THE OBJECT IN BUILD MODE

DRAG THE OBJECT TO DESIRED LOCATION

OBJECT PLACING

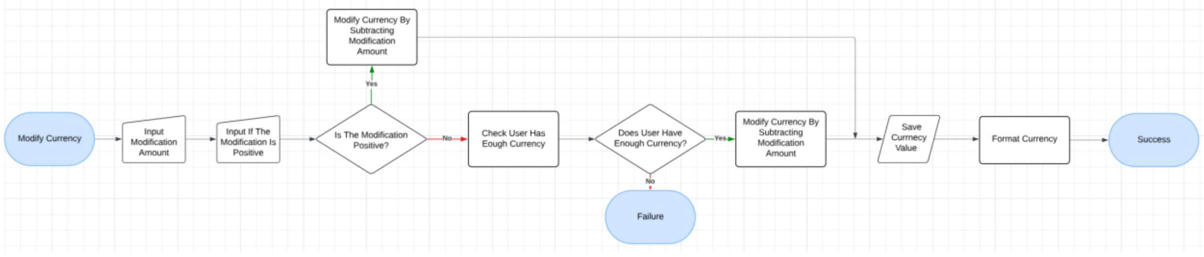


OBJECT CAN BE PLACED

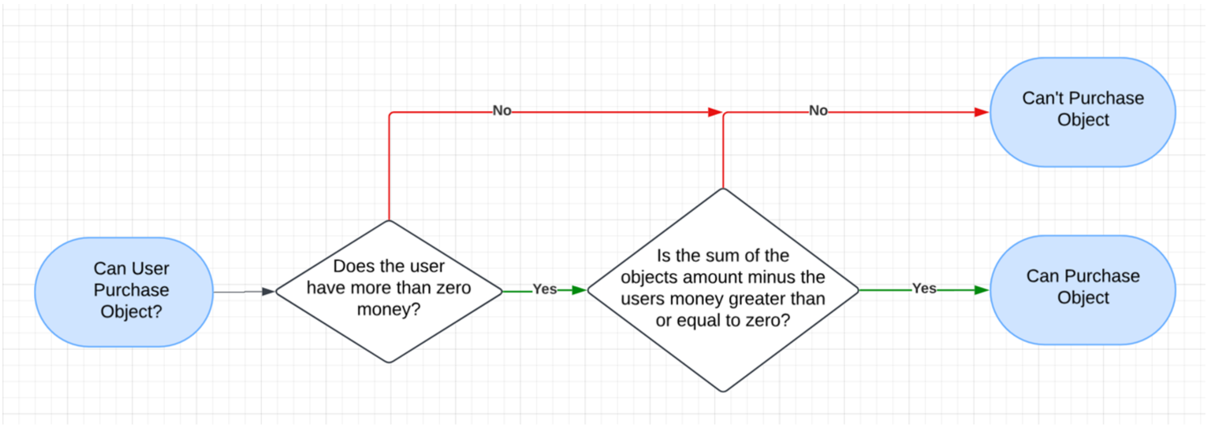
OBJECT CANNOT BE PLACED

Character Data

Here is a flowchart to show the process of modifying a user's currency:


Modify Currency
Input Modification Amount
Input If The Modification Is Positive
Is The Modification Positive?
Yes
Modify Currency By Subtracting Modification Amount
No
Check User Has Eough Currency
Does User Have Enough Currency?
Yes
No
Failure
Modify Currency By Subtracting Modification Amount
Save Currnecy Value
Format Currency
Success

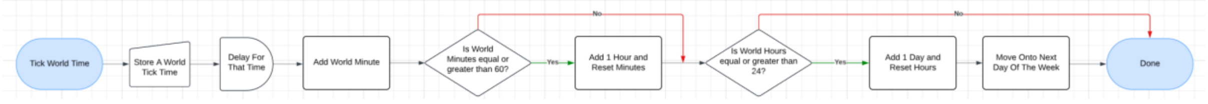
Here is a flowchart to show the process of checking a user's currency:


Can User Purchase Object?
Does the user have more than zero money?
No
Yes
Is the sum of the objects amount minus the users money greater than or equal to zero?
No
Yes
Can't Purchase Object
Can Purchase Object

Here is how the currency is format: **'£xxx,xxx'**

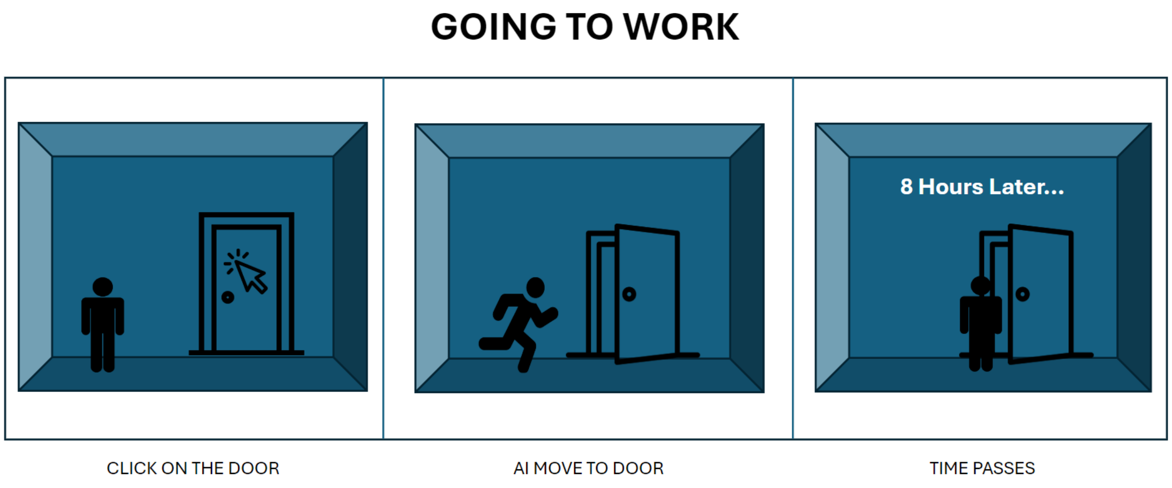
World Time

Here is a flowchart showing the process of the world time ticking:

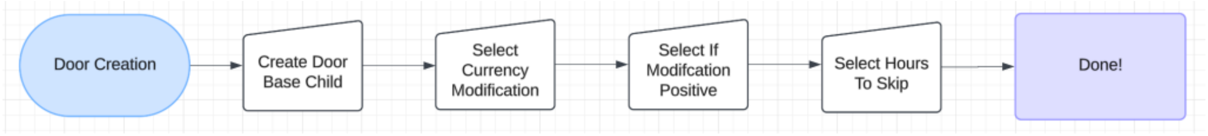

Tick World Time
Store A World Tick Time
Delay For That Time
Add World Minute
Is World Minutes equal or greater than 60?
No
Yes
Add 1 Hour and Reset Minutes
Is World Hours equal or greater than 24?
No
Yes
Add 1 Day and Reset Hours
Move Onto Next Day Of The Week
Done

Working / Doors

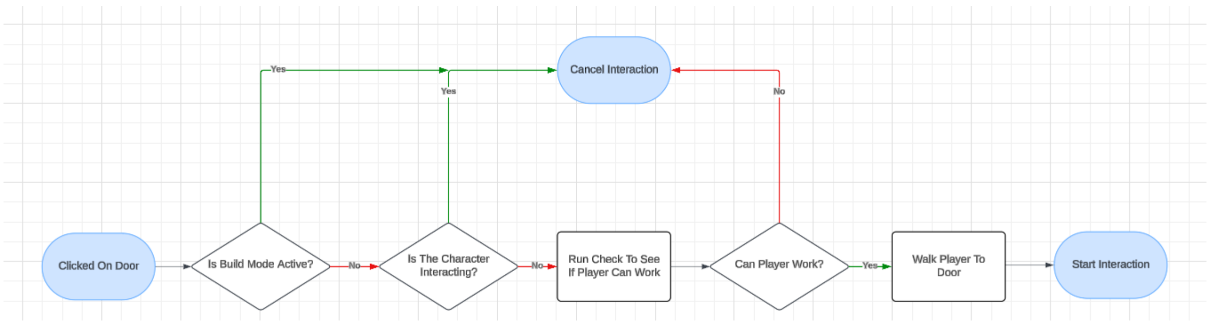
Here is a diagram I made to show the character going to work:



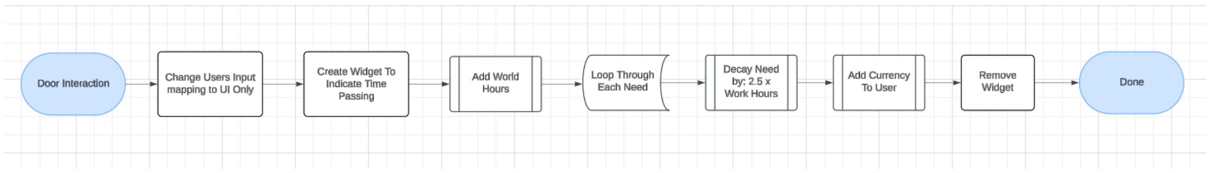
Here is a flowchart showing the process of creating a door child during the creation of the game:



Here is a flowchart showing the process of clicking on a door:

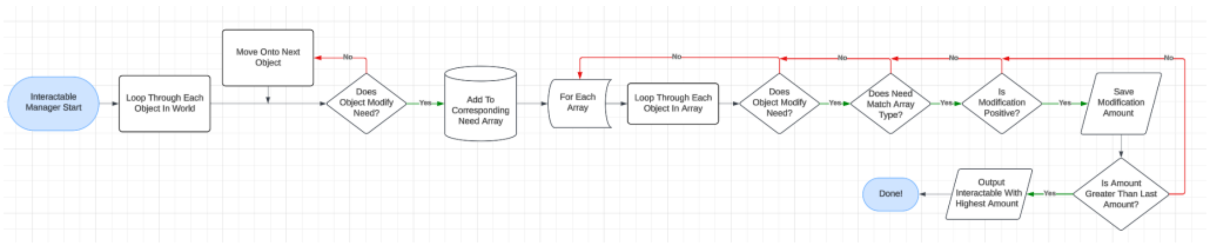


Here is a flowchart showing the interaction logic for a door:


Door Interaction
Change Users Input mapping to UI Only
Create Widget To Indicate Time Passing
Add World Hours
Loop Through Each Need
Decay Need by: 2.5 x Work Hours
Add Currency To User
Remove Widget
Done

Interactable Manager

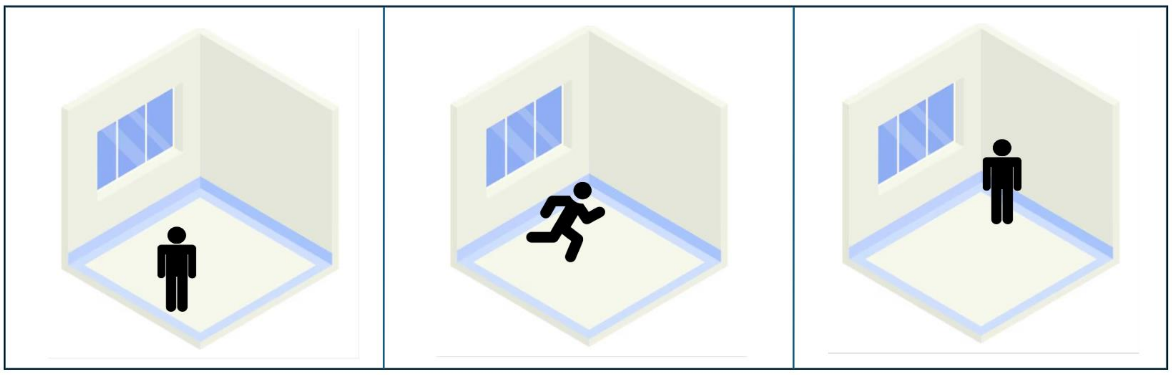
Here is a simplified flowchart showing the logic and flow of how the interactable manager sorts and outputs interactable objects that give the highest modification value per need:


Interactable Manager Start
Loop Through Each Object In World
Move Onto Next Object
No
Does Object Modify Need?
Yes
Add To Corresponding Need Array
For Each Array
Loop Through Each Object In Array
No
Does Object Modify Need?
Yes
No
Does Need Match Array Type?
Yes
No
Is Modification Positive?
Yes
No
Save Modification Amount
Is Amount Greater Than Last Amount?
Yes
Output Interactable With Highest Amount
Done!

AI Takeover

Here is a diagram I created to show how the AI Takeover will work:

AI TAKEOVER

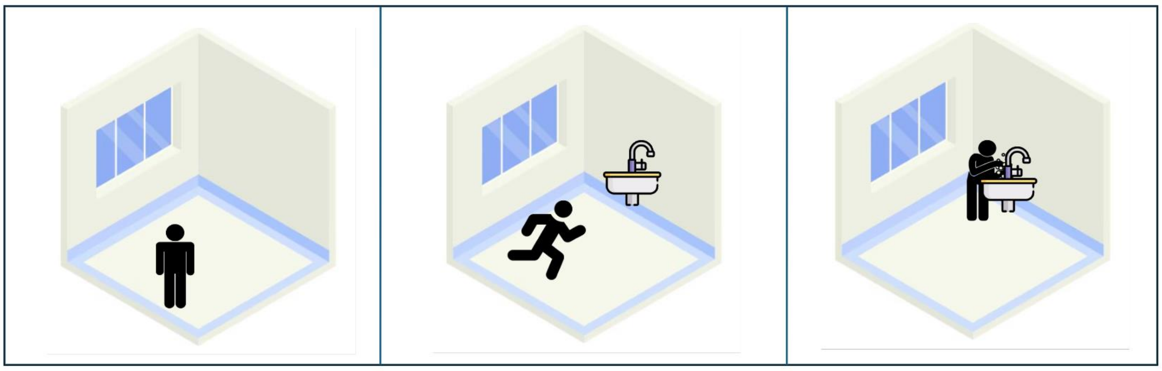


IDLE FOR 20 SECONDS

AI MOVE TO RANDOM LOCATION

WAIT FOR 5 SECONDS AND MOVE AGAIN

AI TAKEOVER (WITH OBJECT)

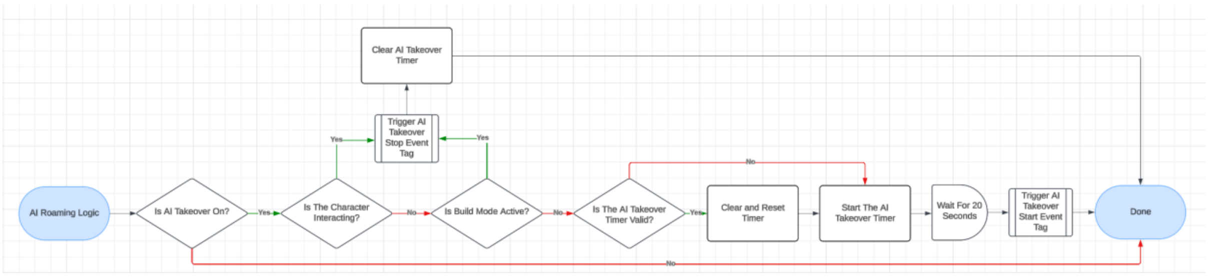


IDLE FOR 20 SECONDS AND NEED BELOW 25%

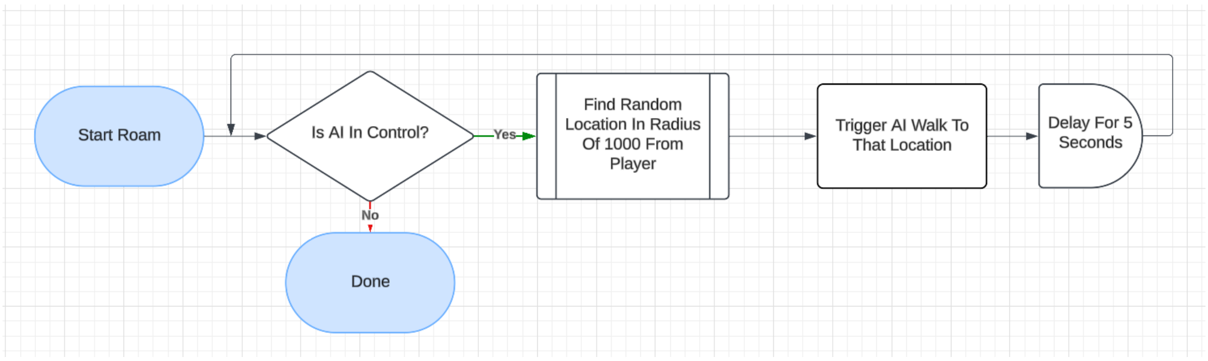
AI MOVE TO OBJECT TO FULFIL NEED

COMPLETE INTERACTION AND RETURN TO ROAM

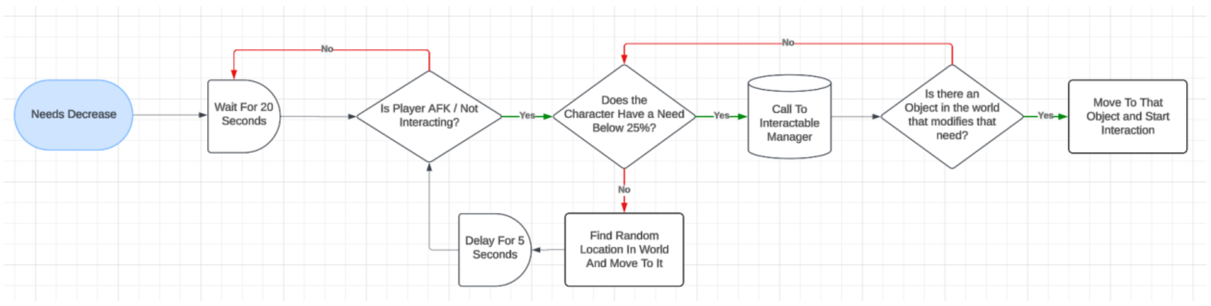
Here is a flowchart showing the logic of the AI roaming event:


AI Roaming Logic
Is AI Takeover On?
Yes
Is The Character Interacting?
Yes
Trigger AI Takeover Stop Event Tag
Clear AI Takeover Timer
No
Is Build Mode Active?
Yes
No
Is The AI Takeover Timer Valid?
No
Yes
Clear and Reset Timer
Start The AI Takeover Timer
Wait For 20 Seconds
Trigger AI Takeover Start Event Tag
Done
No

Here is a flowchart showing the logic of the state tree roam functionality:

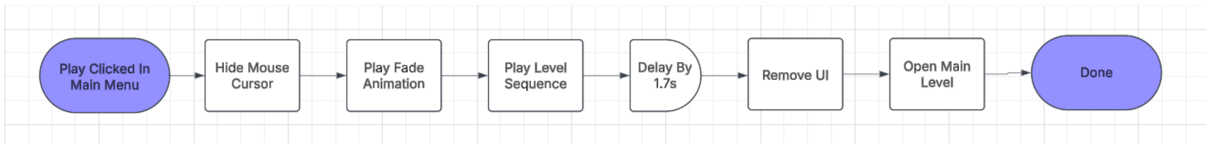

Start Roam
Is AI In Control?
Yes
Find Random Location In Radius Of 1000 From Player
Trigger AI Walk To That Location
Delay For 5 Seconds
No
Done

Here is a flowchart showing the logic of the AI Auto Refill Needs functionality:

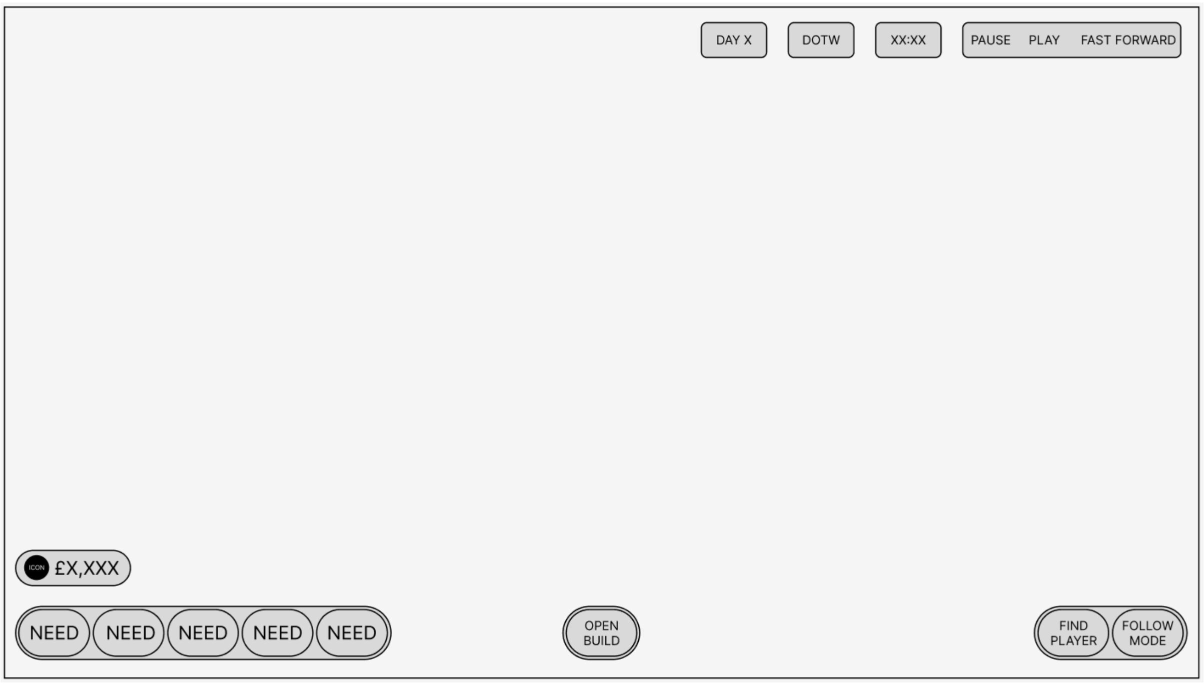

Needs Decrease
Wait For 20 Seconds
No
Is Player AFK / Not Interacting?
Yes
Does the Character Have a Need Below 25%?
Yes
No
Call To Interactable Manager
Is there an Object in the world that modifies that need?
No
Yes
Move To That Object and Start Interaction
Find Random Location In World And Move To It
Delay For 5 Seconds

UI

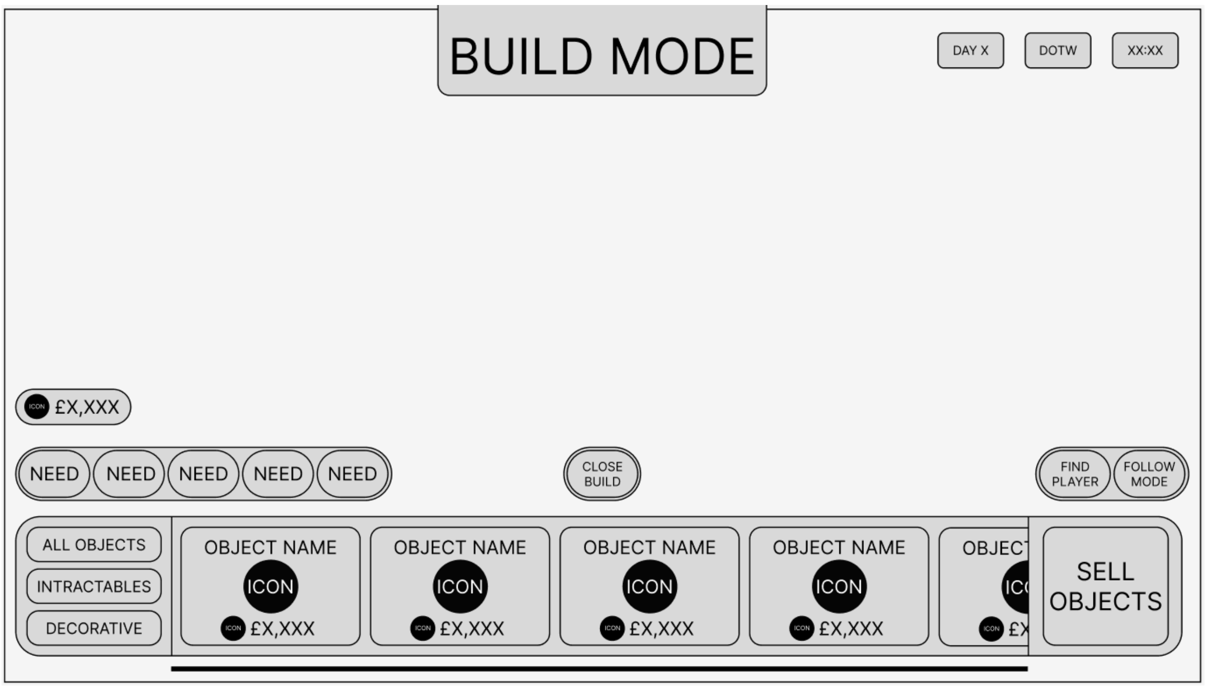
Here is a flowchart showing the logic of how the game is started and loaded from the main menu:



Here is a wireframe design showing how the main UI HUD will be laid out (no build mode):



Here is a wireframe design showing how the main UI HUD will be laid out (build mode):



Coding Standards

Programming Standards

For naming conventions, I will attempt to follow the unreal engine recommended asset naming conventions linked here:

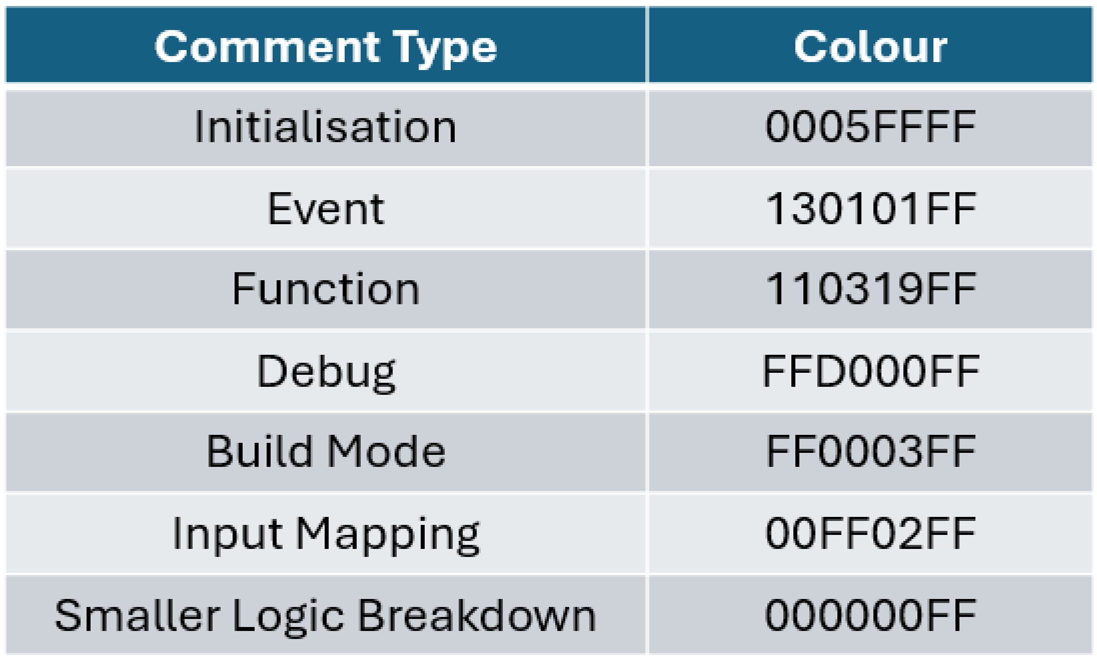
https://dev.epicgames.com/documentation/en-us/unreal-engine/recommended-asset-naming-conventions-in-unreal-engine-projects

For efficiency rules, I will attempt to make as much of the blueprint code as modular as I possibly can. This means creating large parent actors to create other world objects and actors, the use of enumeration and structures to ensure that data is correctly and efficiently laid out.

Commenting Rules

For commenting rules, I will be commenting all larger and important blocks of code with a simple comment title that briefly explains the functionality of the block. For comment colours I also plan on using certain colours for certain types of code. For example, using yellow for debug code and blue for initialisation code. This is being done to ensure that the code is readable. If code is readable it allows for collaborative work with other developers, future project development after a long time a way (comments act as reminders), or even the ability to move the project on to someone else or another company.

Here is a table showing my comment types and their colours:



Comment Type	Colour
Initialisation	0005FFFF
Event	130101FF
Function	110319FF
Debug	FFD000FF
Build Mode	FF0003FF
Input Mapping	00FF02FF
Smaller Logic Breakdown	000000FF

Code Review Procedures

The code in my project will be reviewed very regularly. This is because I am going to be learning a lot about unreal during this development process, I plan on noting down areas of inefficient code during the creation process so that they may be returned to later. Furthermore, as my knowledge of the best coding practices increases as well as my skill in unreal, I will be consistently updating and improving code throughout the project.

Production Overview

Moscow

Must Have:

- A wide range of camera controls including movement, rotation (with keyboard and mouse), zoom controls, and player find/follow features.
- Game time controls such as stop, play and fast forward.
- Character needs that decay over time.
- Interactable objects that increase player's needs.
- Character currency that can be gained and spent.

Should Have:

- Basic building system that allows for interactable objects to be bought, placed, and sold.
- Character skills that can be gained and has an impact on a player's interaction with different world objects.
- AI system that takes control of the character and automatically solves their needs.

Could Have:

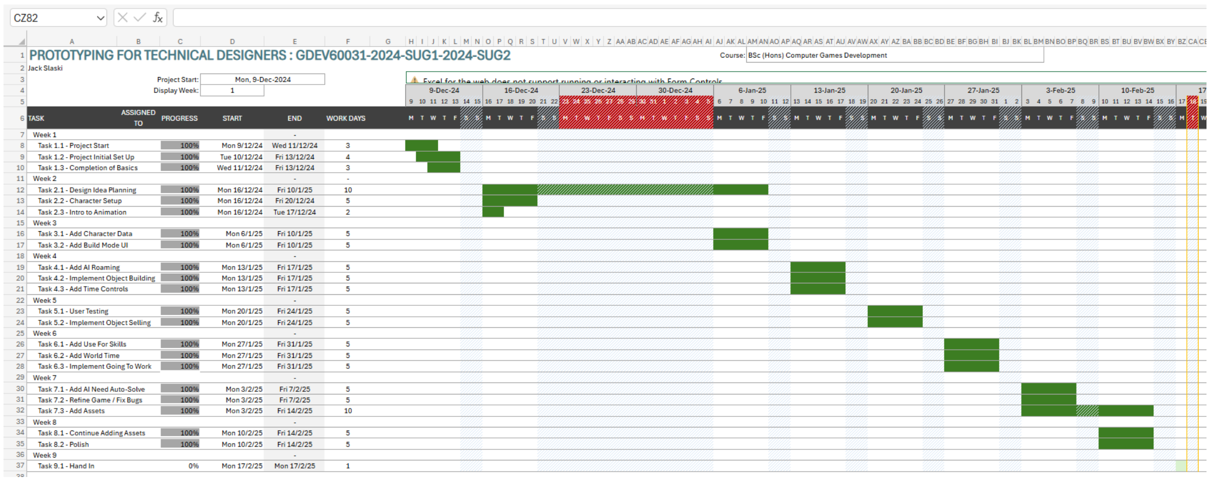
- A more refined building system with more items and snapping.
- Object UI to allow multiple interaction types per object.
- Detailed interaction animations for each object.

Won’t Have:

- Character Creator
- Other characters / NPCs
- House building system
- Other locations

Timeline

PTDGanttChart.xlsx



PROTOTYPING FOR TECHNICAL DESIGNERS : GDEV60031-2024-SUG1-2024-SUG2

Course: BSc (Hons) Computer Games Development

Jack Slaski

Project Start: Mon, 9-Dec-2024

Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END	WORK DAYS
Week 1				-	
Task 1.1 - Project Start		100%	Mon 9/12/24	Wed 11/12/24	3
Task 1.2 - Project Initial Set Up		100%	Tue 10/12/24	Fri 13/12/24	4
Task 1.3 - Completion of Basics		100%	Wed 11/12/24	Fri 13/12/24	3
Week 2				-	-
Task 2.1 - Design Idea Planning		100%	Mon 16/12/24	Fri 10/1/25	10
Task 2.2 - Character Setup		100%	Mon 16/12/24	Fri 20/12/24	5
Task 2.3 - Intro to Animation		100%	Mon 16/12/24	Tue 17/12/24	2
Week 3					
Task 3.1 - Add Character Data		100%	Mon 6/1/25	Fri 10/1/25	5
Task 3.2 - Add Build Mode UI		100%	Mon 6/1/25	Fri 10/1/25	5
Week 4				-	
Task 4.1 - Add AI Roaming		100%	Mon 13/1/25	Fri 17/1/25	5
Task 4.2 - Implement Object Building		100%	Mon 13/1/25	Fri 17/1/25	5
Task 4.3 - Add Time Controls		100%	Mon 13/1/25	Fri 17/1/25	5
Week 5				-	
Task 5.1 - User Testing		100%	Mon 20/1/25	Fri 24/1/25	5
Task 5.2 - Implement Object Selling		100%	Mon 20/1/25	Fri 24/1/25	5
Week 6				-	
Task 6.1 - Add Use For Skills		100%	Mon 27/1/25	Fri 31/1/25	5
Task 6.2 - Add World Time		100%	Mon 27/1/25	Fri 31/1/25	5
Task 6.3 - Implement Going To Work		100%	Mon 27/1/25	Fri 31/1/25	5
Week 7				-	
Task 7.1 - Add AI Need Auto-Solve		100%	Mon 3/2/25	Fri 7/2/25	5
Task 7.2 - Refine Game / Fix Bugs		100%	Mon 3/2/25	Fri 7/2/25	5
Task 7.3 - Add Assets		100%	Mon 3/2/25	Fri 14/2/25	10
Week 8				-	
Task 8.1 - Continue Adding Assets		100%	Mon 10/2/25	Fri 14/2/25	5
Task 8.2 - Polish		100%	Mon 10/2/25	Fri 14/2/25	5
Week 9				-	
Task 9.1 - Hand In		0%	Mon 17/2/25	Mon 17/2/25	1