Technical Design Document

Contents

Table of Contents **1**

Game Details 2

Team Members **2**

Game Concept 2

Technical Goals **3**

Technical Goal 1 – Player Movement/Input 3

Technical Goal 2 – NPC A.I. **3**

Technical Goal 3 – UI and Menus 3

Technical Risks **3**

Technical Risk 1 – Familiarity with Software Package (Unity) 3

Technical Risk 2 – Running out of time **4**

Features/Mechanics/Tasks 4

Deliverables **5**

System Requirements 5

Target Device 1 - Desktop PC (Windows) **5**

Third Party Tools 6

File Formats **6**

Coding Conventions 6

Source Control **6**

Game Flow 7-8

Game Objects and Scripts **9-10**

Gameplay Systems 11

Gameplay System 1 – Player Control **11**

Gameplay System 2 – NPC A.I 12

Input Method(s) **13**

User Interface 14-15

Revision History Table **16**

Game Details

- **Game Name:** Midnight Blood
- **Team Name:** HiveMind Productions

Team Members

List of technical team members and broad overview of their roles.

Name	Job Title	Responsibilities/Roles
Andrew Osborn	Controls Programmer	Creation of player controller and input
Fredrick Bancan	Gameplay Programmer	NPC A.I and gameplay systems integration
Sam Nagy	UI Programmer	Menus and user interface creation

Game Concept

Describe the game concept here in 2-3 sentences. Focus on what the player can DO.

Player spawns each "night"(iteration of gameplay/map/level) with a specific amount of energy value. This energy value will slowly decline, once reaching zero the player loses. To replenish this energy value the player must use movement to find villager NPC's and attack them from behind, by clicking their back in melee range.

The game level/map has a timer for the sun rising, which will cause the overcast skybox to increase in brightness and the map's global illumination to increase. Possibly with a clock UI element to show how much time is left. The player must go back to their spawn area and confirm they want to finish the night before the timer hits 0/ sun rises. Otherwise the player loses.

Once player has killed first victim, they will be covered in blood (boolean), after this point, if they are within a villagers line of site, in field of view and within range, the villager will be set to alert state and flee from the player. Once the villager is out of line of sight of the player, an authority NPC will spawn and seek the player, the player will be notified of this in the HUD. Once the authority NPC touches the player, the player loses.

Technical Goals

The team aims to deliver gameplay and an environment which is suitable enough to support the games suggested systems, but not too complex that they take too long to develop.

Intuitive and fun player movement and input, believable A.I movement/behaviours and intuitive U.I.

Technical Goal 1 – Player Movement/Input

Who's Responsible: Andrew

Description: Successful creation and implementation of player input and movement physics. Must be modular enough to work with other gameplay systems, and well implemented enough to feel intuitive to the user.

Technical Goal 2 – NPC A.I

Who's Responsible: Fred

Description: Successful creation and implementation of NPC movement, logic and state changes. Must be suitable for the suggested gameplay elements and systems, as well as being realistic and intuitive enough to be believable / immersive.

Technical Goal 3 – U.I and Menus

Who's Responsible: Sam

Description: Successful creation and implementation of U.I and Menu System and graphics. Must fit the suggested game systems and display all required information to the user. As well as allow the user to make any decisions they are required too, such as quitting the game, or changing a setting.

Technical Risks

What are the technical (i.e. related to programming) features and ideas most likely to cause problems? E.g A mechanic requires learning new design patterns and 3rd party libraries you've never used before. Or none of the programmers have experience with AI.

What can you do to reduce the risk? E.g Bob will perform additional research and will spend the first two days making a test project to prove this idea is possible. If not, we will cut the idea.

Technical Risk 1 – Familiarity with software package *(Unity)*

What's the risk about: The programming team's combined experience of the Unity game engine is a bit lacking. Sam having had experience building out a game in 2d, Andrew having worked on a 2d space-shooter demo and some 3d basics, and Fred having had little direct experience with Unity.

These could present challenges in building out the mechanics of the game in a timely manner.

How will risk be mitigated: There will be additional research into translating our game mechanics from 2d into a 3d environment. Having a stripped-back gameplay system to start with will also allow us to keep on schedule.

Technical Risk 2 – Running out of time

What's the risk about: The programming team has a limited amount of time to implement all of the required systems. If they can not implement these systems completely or correctly within the given time span, the game may not be playable or as polished.

How will risk be mitigated: Cutting back on extra technical details/systems which are not required or do not require the suggested amount of complexity/detail, if required.

Features/Mechanics/Tasks

A list of exactly what systems exist in your game and who is responsible, including scheduled dates for completion.

Feature/Mechanic	Who's responsible	Scheduled Date
Player walking / looking around	Andrew	Alpha, 27/11/2020
Player attacking/feeding	Andrew	Alpha, 27/11/2020
NPC movement	Fred	Alpha, 27/11/2020
NPC logic and state changes	Fred	Alpha, 27/11/2020
Night/Day transition mechanic	Grant	Alpha, 27/11/2020
Menu System	Sam	Alpha, 27/11/2020
UI and HUD	Sam	Alpha, 27/11/2020
Map	Grant	Alpha, 27/11/2020
Polishing Player movement / interactions	Andrew	Beta, 2/12/2020
Polishing NPC movement/logic	Fred	Beta, 2/12/2020
Polishing night/day transitions and timing	Grant	Beta, 2/12/2020
Polishing ui and menus	Sam	Beta, 2/12/2020
Polishing map	Grant	Beta, 2/12/2020

Deliverables

Deliverable	Who's Responsible	Who's the Owner
Final project build/executable.	Grant	Client

System Requirements

- 2GB ram
- Windows 10
- core i3 processor
- intel integrated graphics
- 10 gb HDD space

Target Device 1 - Desktop PC (Windows)

Recommended Hardware:

- 4 gb ram
- Windows 10
- core i5 processor
- nvidia GTX 560 or higher
- 15gb HDD space

Third Party Tools

What third-party tools are you using? List Unity with version number and any other tools you might need. Include any asset packs you plan to use from the asset store.

We will be using Unity version 2020.1.12f1 to build our project.

We will be utilising VisualStudio 2019 for writing our scripts.

GitHub Desktop for Windows, to manage source control.

File Formats

What file formats will be used for models, textures, sounds and other assets.

If your project will be using a custom file format, explain in detail the format (eg. Text/Binary, order and size of each element in the file, file/byte offsets (if necessary))

- Sound: .Wav file.
- Texture:.tga, .png
- Models:.obj, .fbx
- Scripts: .cs (C#) files.

Coding Conventions

Our team will be following a coding convention outlined in the guideline found at this URL: https://csharpcodingguidelines.com

Source Control

Source Control Repository: *GitHub.*

Source Control Client Tools: *GitHub Desktop for Windows*

Source Control Remote Repo URL: https://github.com/rabbet35/MidnightBlood

Ignore/Config file: https://github.com/rabbet35/MidnightBlood/blob/main/.gitignore (Visual studio default Unity .gitignore file)

Commit message formats: Effective description of changes made in commit.

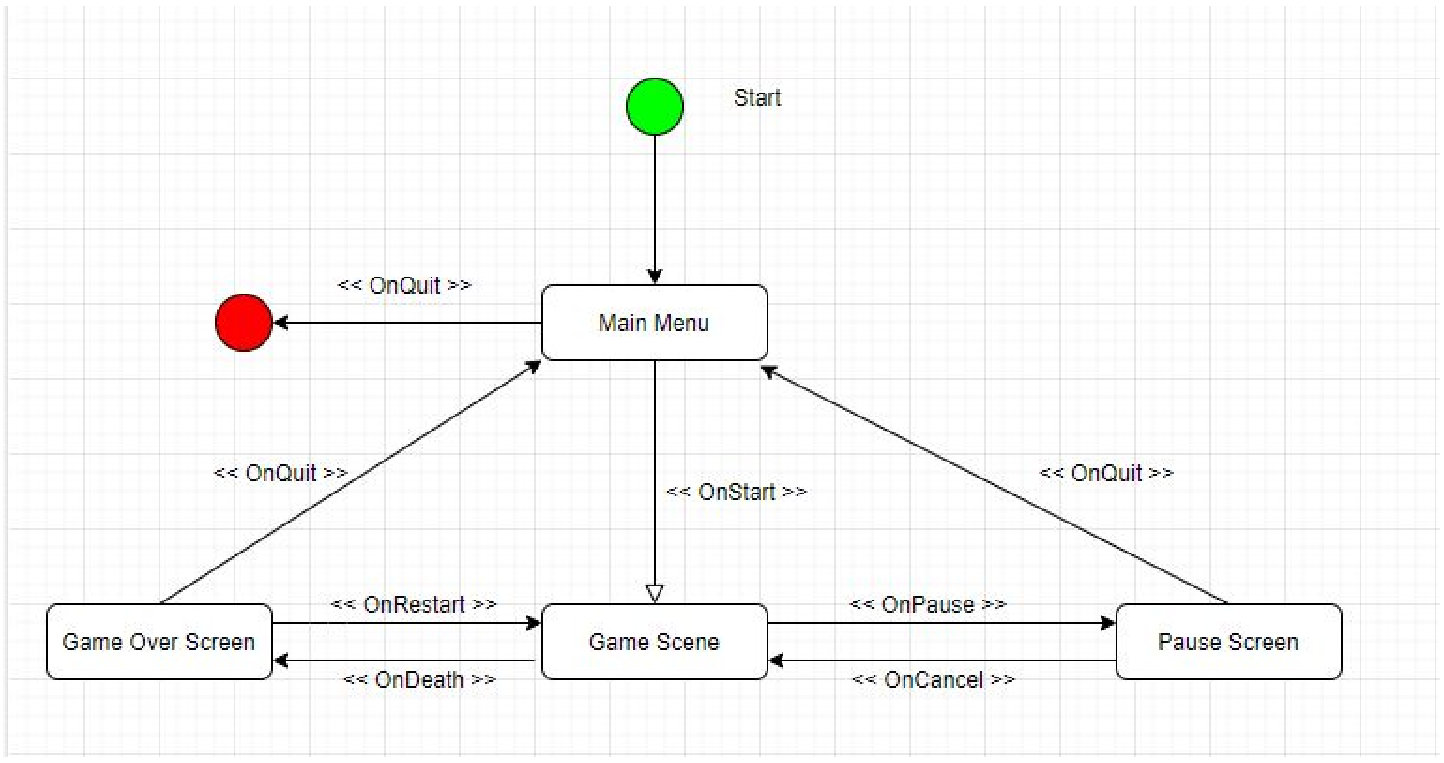
Other repo notes: Each team member will only work on a certain file at a time to avoid conflicts.

Game Flow

List the scenes in the game, and a short description of what the scene is responsible for.

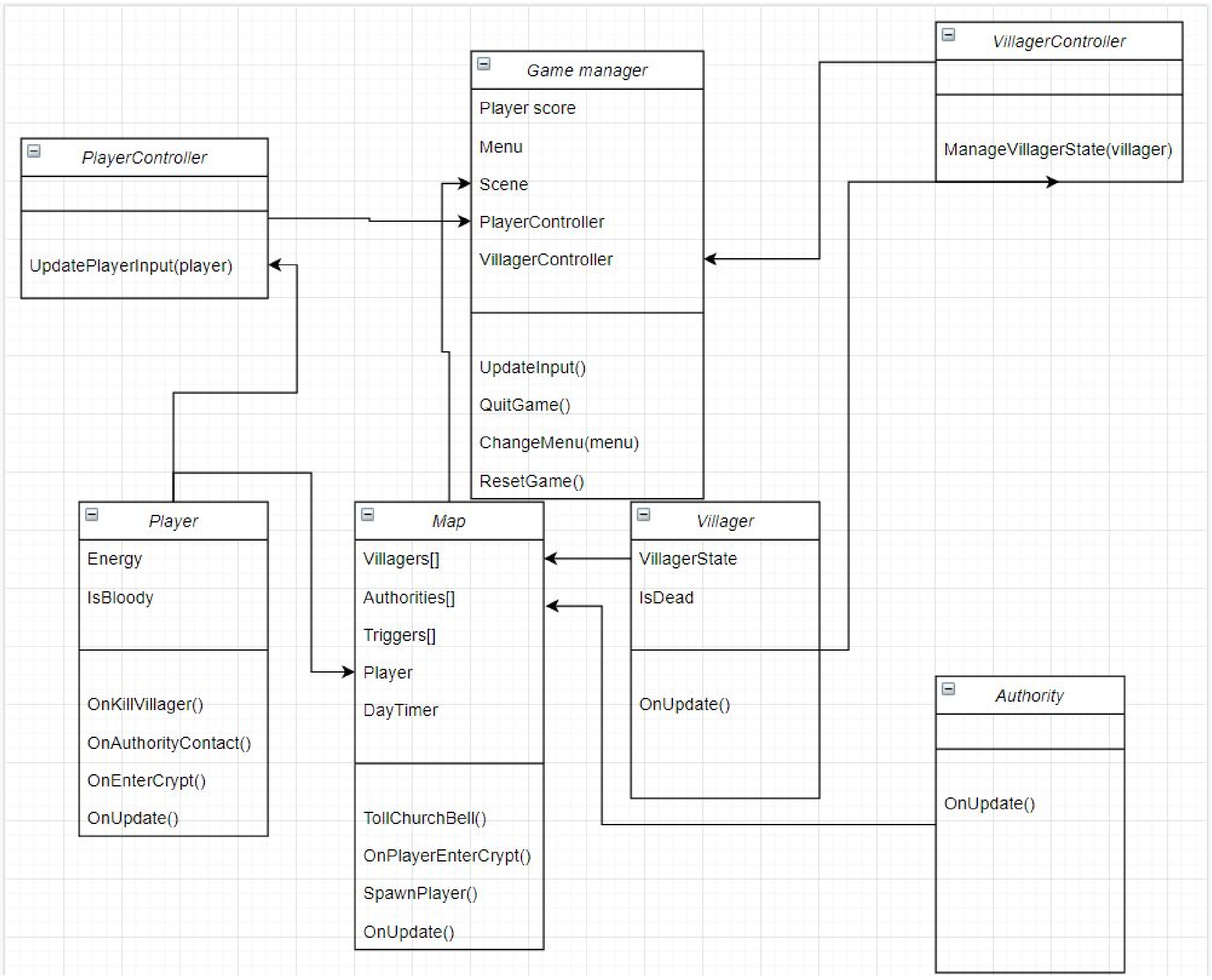
Scene	Who's responsible	What is does
Enter each game menu		
Main Menu	Sam	Starting menu, allows player to choose to start playing or quit.
Pause Menu	Sam	Menu with options to quit game, or restart game. Also shows some stats about the player/current game.
Game Over Menu	Sam	Menu which is shown when the player loses the game with options to quit and try again. Also shows stats about the game which the player just lost.
Main Game Level	Grant	Load the main game level and manages the main gamestate. Can be interrupted by the player when they chose to open the Pause menu via keyboard input.

Include State Diagrams to describe menus & basic game flow (example below).

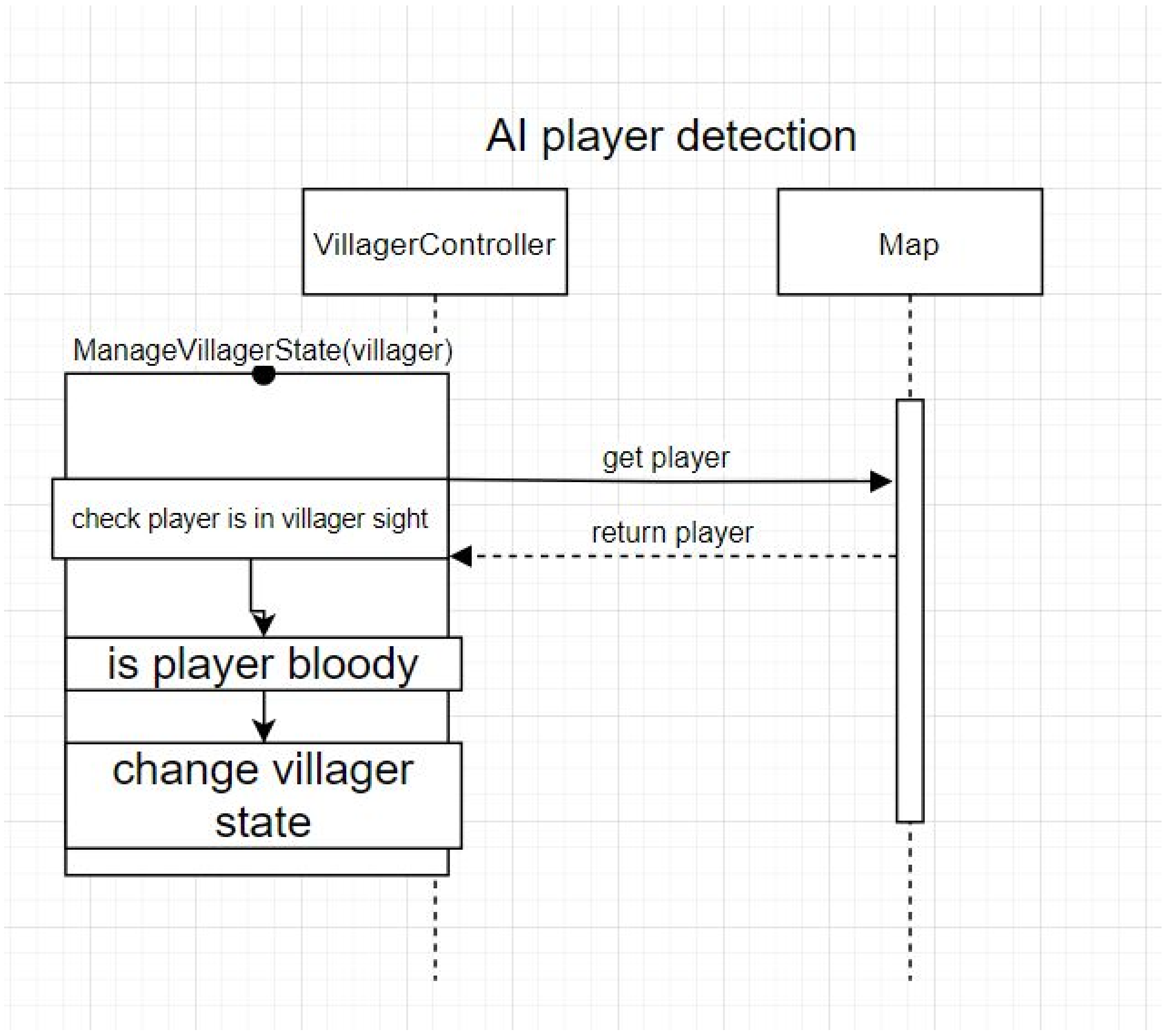

Start
Main Menu
<< OnQuit >>
<< OnQuit >>
<< OnStart >>
<< OnQuit >>
Game Over Screen
<< OnRestart >>
<< OnDeath >>
Game Scene
<< OnPause >>
<< OnCancel >>
Pause Screen

Game Objects and Scripts

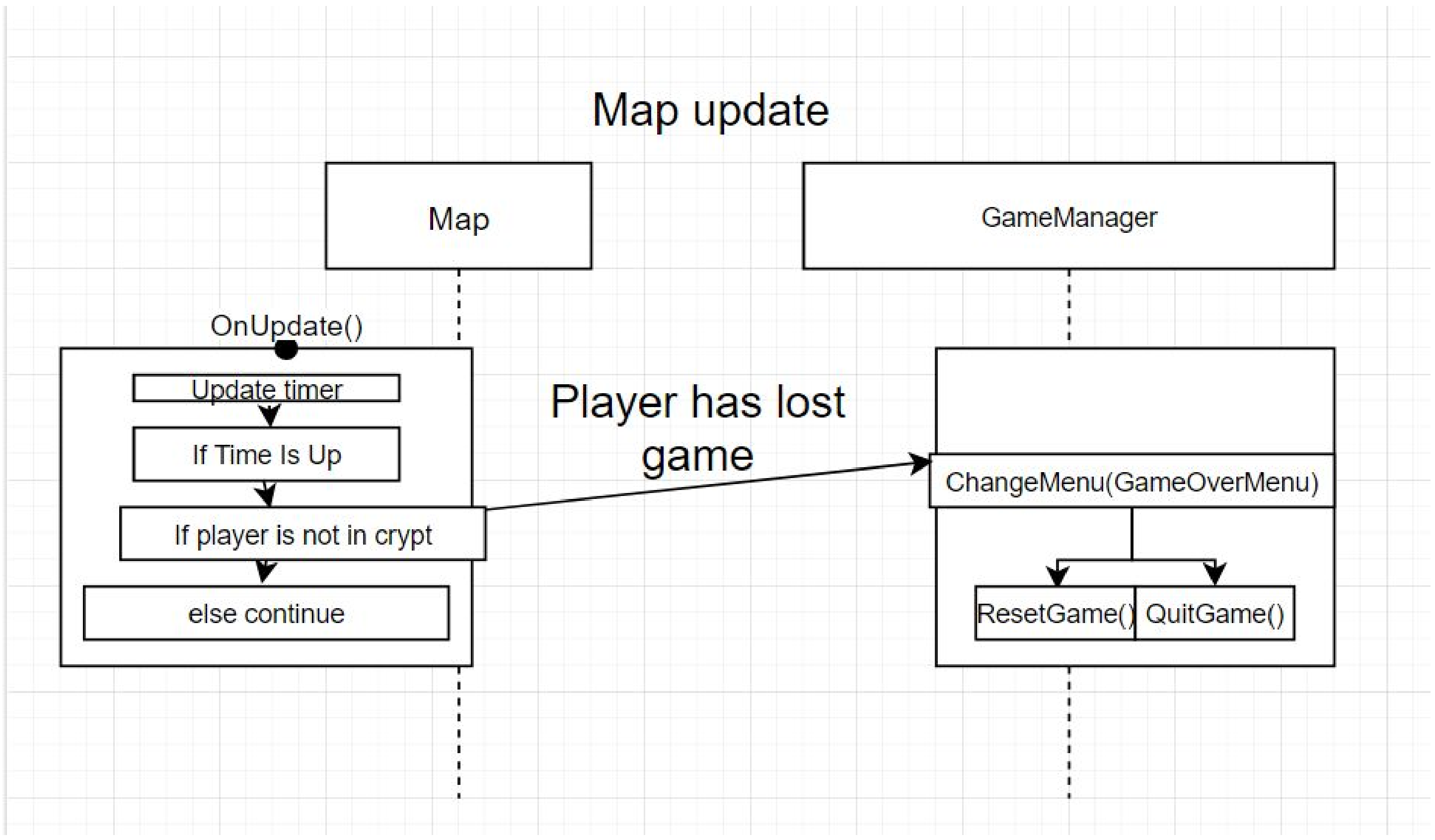
Main architecture:



A.I Player detection:



Map updating:



Gameplay Systems

The players' input is a gameplay system of how they can move and interact with the NPCs in the game, as well as the environment. The player can move using keys on the keyboard (presumably W A S and D) and use the mouse to move the camera (left and right, up and down. maximum vertical angle of 90 degrees).

The NPCs have states and logic which moves them in a natural and unpredictable manner. They have systems for checking for variables such as if they can see the player while the player is bloody. The Authority NPC is more simple and simply runs at the player after it has been spawned in the world.

Gameplay System 1 – Player control

Who's Responsible: Andrew

Description: Player movement and other input

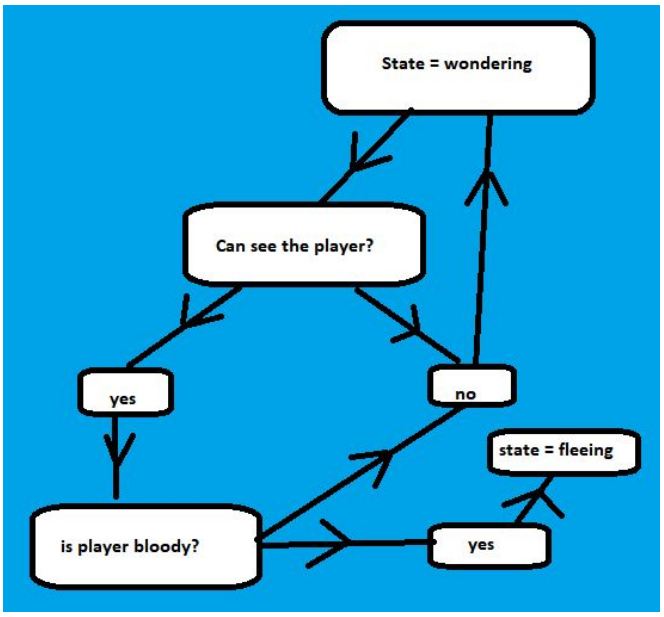
Diagrams: *none*

Gameplay System 2 – NPC A.I

Who's Responsible: Fred

Description: NPC Movement and interaction with player and environment (state machine based)

Diagrams:



Gameplay Systems

Input Method(s)

Describe the Input method for each target platform (e.g PC / VR / Console).

Target Platform	Input System	Who is responsible
PC	Mouse/Keyboard	Andrew

Keyboard Input will be a version of first-person controls
WASD keys will be used for traversing the map

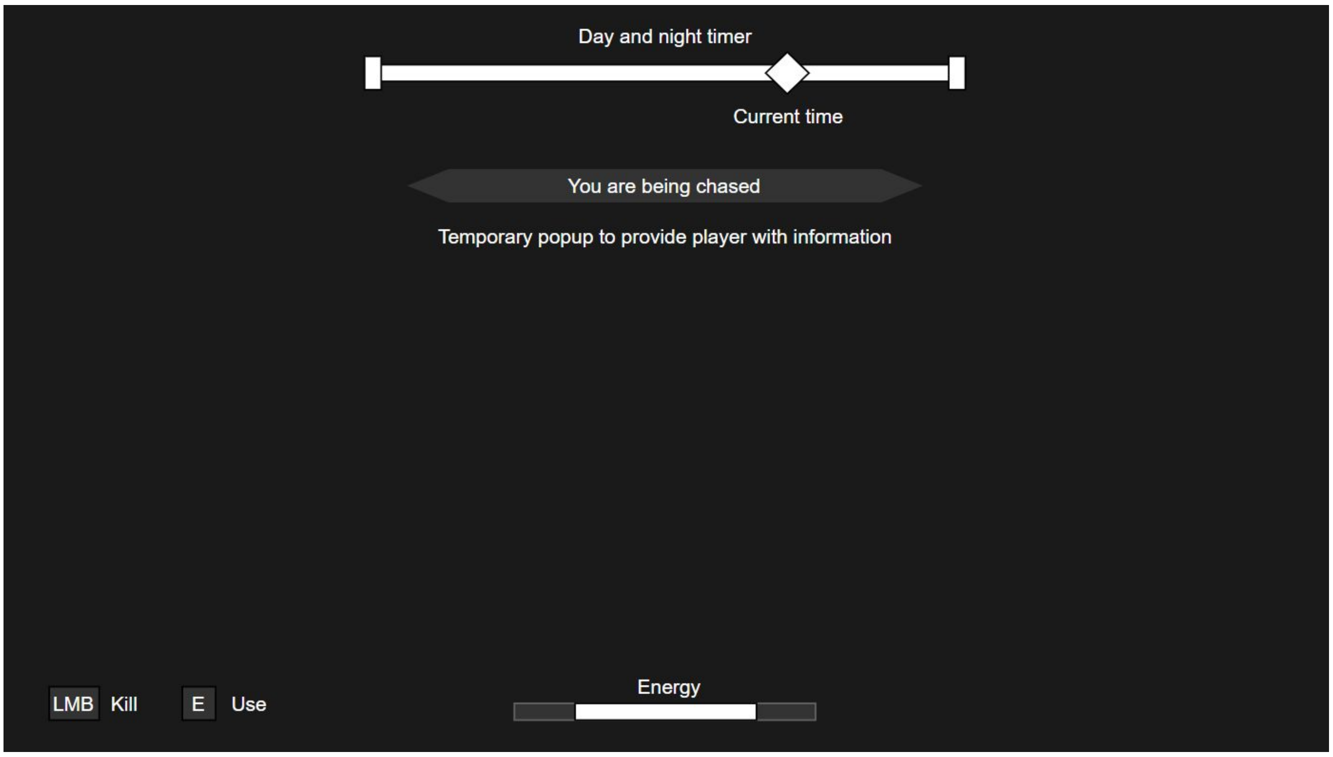
Esc to bring up the Pause Menu

mouse movement to look around

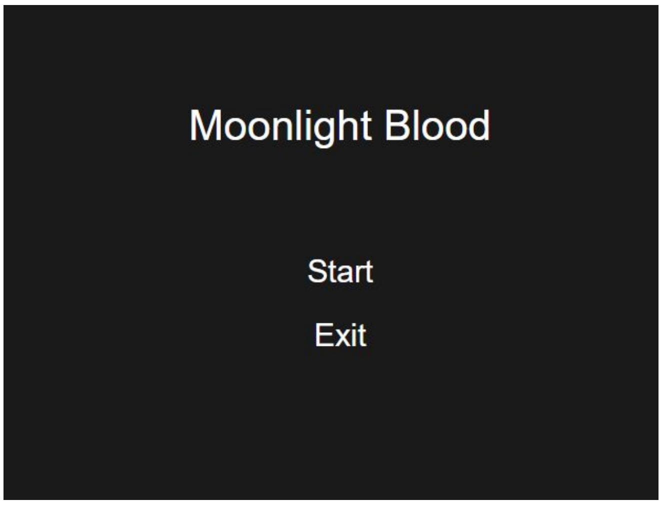
left-click on mouse to attack/feed.

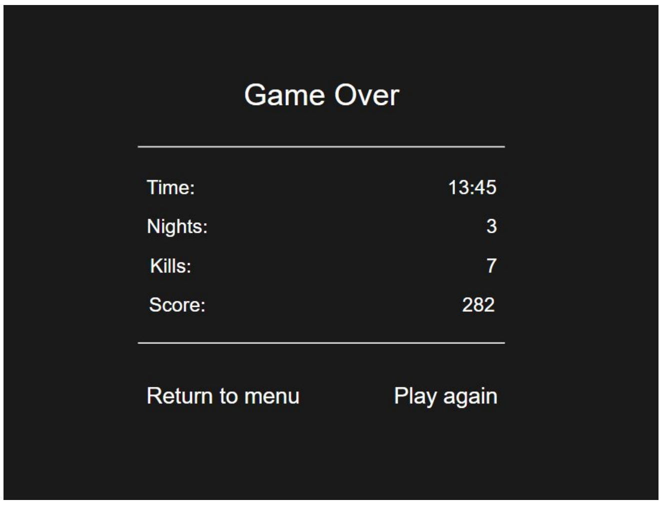
User Interface

Basic in-game HUD

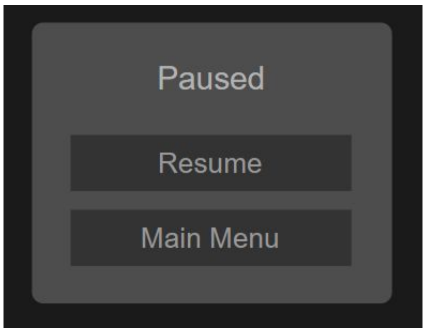


Simple start, end, and pause menu





Time: 13:45
Nights: 3
Kills: 7
Score: 282



Revision History Table

Version	Description	Editor
v0.0	Initial document creation	Fred
v.0.1	Filled out game info and concept	Fred
v0.2	Filled out Gameplay Systems page	Fred
v0.3	Filled out technical goals section	Fred
v0.4	Added Technical Risk, worked on Features/Mechanics/Tasks list	Andrew
v0.5	Added to Input Method(s)	Andrew
v0.6	Filled out Third Party Tools	Andrew
v0.7	Added technical risk 2 and filled out features/mechanics/tasks	Andrew and Fred
v0.8	Updated Third Party Tools and Source Control	Andrew and Fred
v0.9	Added U.I mockups	Sam
v1.0	Added game flow state diagram	Andrew
v1.1	field out file formats	Fred
v1.2	Added script diagrams under game objects and scripts	Fred
v1.3	Proof Read and update Table of Contents	Andrew