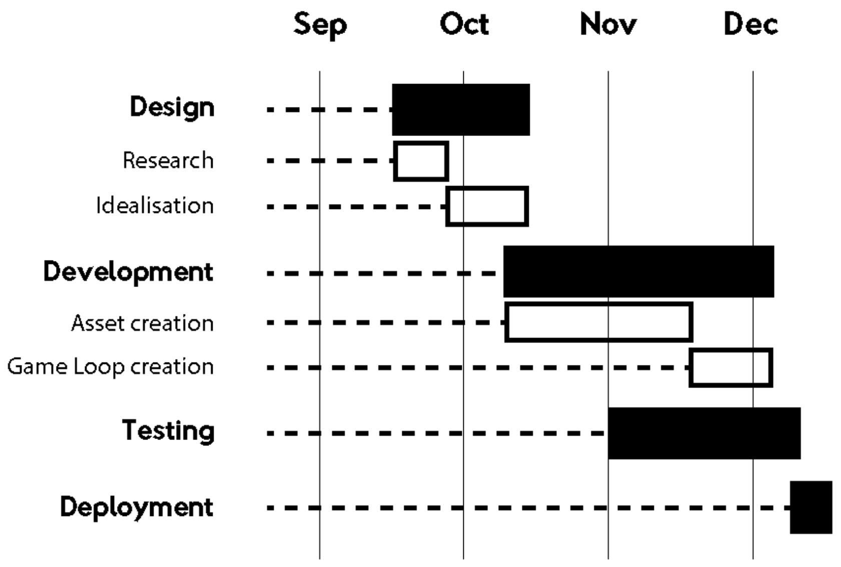
Adam Kraus 1907800

CM4114: Games Development

Project Plan

I started by looking into concepts that I could tie in with the assignment's theme, “Beyond”, and wanted to make a rogue-like game where you set off from a “home base” and defend the threats from beyond. I decided to make this in an isometric view as I originally wanted to explore the possibilities of displaying 2D objects in 3D space. Working from here, I created my base concepts for the game and sketched the main components, which gave me a better idea of the final product and allowed for a smooth creation and gathering of assets. I started implementing the basic features and creating simple prefabs and interactions. Having started testing around this point allowed me to gradually move on to implementing the main game loop, which ended up posing a larger challenge than I expected. Eventually, with the project ready, everything was exported and compiled.



Requirements Analysis

The requirements analysis reflects on the design goals and outlines the common objectives for the gameplay and the assessment requirements. While this provides a reasonable basis for the implementation, each requirement will need to be broken down much further after understanding the problems at hand. Completing the requirements categorised by the MoSCoW methodology should ensure a fully sustainable game loop and satisfy the project's requirements. In addition to the essential functions, features like key rebinding and dynamic soundtrack were added in line with the design.

Functional requirements

- The player must be able to launch and close the game;
- the player must be able to control the character;
- the player must be able to control the volume;
- the player must be able to damage/kill enemies;
- enemies should be present and able to attack;
- the player should be able to upgrade skills/abilities;
- the player should be able to gain back health;

- multiple levels with scaling difficulty should be present;
- the player could be able to gain experience points by killing enemies;
- a trace of player movement could be added;
- a difficulty setting could be implemented;
- multiple types of enemies could be implemented;
- a traced field of vision could be implemented;
- the player could be able to remap buttons.

Non-functional requirements

- Contextual graphical hints should be added.
- a loading screen could be added,
- and contextual music could be added.

Extensive Accessibility Analysis

While rogue-like games might be considered "hard" and have a steep learning curve, they have unique gameplay that is hard to replicate. While it may feel that the game's progression is based on the players' hardships, there are ways the game designers can use this to aid the player through the story and/or enhance the gameplay itself.

Case 1: Binding of Isaac

Due to the similarities in desired gameplay, a quick look at this rogue-like game presents several considerations that should be implemented in the project.

1) **Controls** – Using four-directional input for both movement and attack and utilising additional action buttons is the most traditional for this genre due to its web-based origins and the accessibility of full controls on laptops without the need for a mouse. While the mouse controls can be desirable as they allow for much more extensive freedom of movement, the keyboard provides a reliably accessible way of controlling the game. It should be considered first, with the option to change to mouse controls if possible. Additionally, the controls should be displayed as soon as possible after launching the game.

2) **Difficulty considerations** – the rogue-like and -lite genre is often considered difficult. While some games like Hades provide accessibility features to make gameplay easier, Isaac is one of the few games of this genre to feature a difficulty setting. While implementing this can significantly aid accessibility, unique solutions will feel more fluid and honest. To simulate such a feature, "God damage" was implemented on the "J" key that deals flat damage to all entities. While alternate ways of playing the game are introduced to increase the replayability, they can also aid the accessibility. Making player decisions matter more within a game's context allows for difficult self-regulation. To allow for this, the enemies were made never to stop moving towards the player after they get near, making the player take on only as much as they can.

3) **Sound** – using secondary queues for actions and events such as sound, particles, or haptic feedback helps the players notice possible actions and consequences of their actions. In addition to primary visual queues, the player can be notified of something happening on or off

the screen without the need for an expensive solution. To adapt to this, I decided to create my own music for the game as well as get sound effects for the in-game actions.

4) **Contextual hints** – While the freedom the player has in the Binding of Isaac is the core value, it can lead to the player not utilising the entire arsenal of tools. While this might be regarded as a learning curve, actions could and should be taken to aid players in utilising all the tools and features provided. For example, if an action has not been used in a long time, a hint should be displayed that directs the player to press the key and perform the allocated action.

Case 2: Don't Starve Together

Visually very close to the initially intended style, this game is also placed in a semi-isometric world, where you can rotate the environment rather than be fixed to the south-north direction.

1) **Colour coordination** – As well as extensively utilising visual contextual hints, this game uses a system of light that serves both as a safe zone and a visual component. Providing this constant link between mechanics actively introduces this “classification” into the gameplay and allows for further expansion by adding more types of coloured lights. These can be used to signify hostility or allegiance. While colour-typing objects is a common practice, utilising environmental connections can strengthen this link and aid in directing the player through the game.

2) **2D 3D** – Displaying a 2D world in a 3D environment is an approach not many games take. While there are some exceptions, like Fez, the movement is usually solved by translating the movement in a similar way you would with a normal character controller. However, if verticality or other mechanics are required, unique solutions can and should be considered for the movement to design “from” accessibility rather than “for” accessibility. Utilising this with Raycasting could produce some interesting solutions deemed out of this project's scope but feasible within the current environment.

Case 3: Vampire Survivors

This enemy hell looter game features a vast map bursting with enemy loot dying to get collected. This simple 2D world can be tricky to navigate as you constantly need to be moving.

1) **Movement and drops** – a large part of the satisfaction you get playing this game comes from the large amounts of experience you collect and how powerful you feel. While managing your movement to collect all the drops is undoubtedly a part of the gameplay, having a magnet on the player for these desired drops and making the collection part easier can lower the load on the player and lead them toward the correct solutions.

2) **Visual overload** – as in any game that features a large number of objects on screen at one time, it is vital that the player is still able to receive game queues for their actions despite being possibly hidden. While we can utilise different colour theory methods to ensure enough

contrast, simply positioning the player object in front of the other enemy layers can serve as a primary solution. This, besides other features such as flash prevention and colour-blind mode, can significantly aid the gameplay of visually overwhelming games.

3) **Object management** – While enemies should ideally collide to lend more credibility to the in-game world, destroying the enemy objects when killed is desirable to prevent the player or other enemies from getting blocked. Leaving a trace or a corpse in the environment means the player still has some visual reference to the number of enemies.

Testing

Apart from actively testing throughout development on my own, I opted for peer qualitative testing. I would observe players without aid and then direct them to perform specific tasks or let them continue exploring independently. In the cases where not everything was covered, a short verbal evaluation followed. I tried to get test runs of newly implemented features soon throughout development to allow me to test different solutions recursively.

Case 1: S. *(active new gamer, 25)*
S. ignored the hint screen and got killed immediately after running into enemies, pressing only one key at a time. After three deaths, he started feeling more comfortable with the controls and cleared the first level. Here, he needed to be directed to the upgrades menu, which he used somewhat effectively throughout the rest of the gameplay. However, despite hinting, he still did not use the dash action but reflected that he wanted but could not remember the keys. Additionally, the audio queues have proven less effective than hoped, as there is no indication that the player still needs to perform a task. In addition, he expressed a wish for the environment to be more interactable or expressive.

Case 2: Tony *(male, active rogue-like player, 23)*
Tony took on the mechanics and controls quickly, although did not use the upgrade menu until the end of the level. He noted more visual feedback, dynamicity in the environment, and more types of enemies. While he struggled to understand the exact function of the fire at first, he quickly adapted. Additionally, he suggested increasing the speed of enemies and the loot magnet's strength to help the gameplay's fluidity, as well as additional drops and level-ups.

Case 3: Ryan *(male, active player, 24)*
Ryan was the only subject that had the chance to test the games at multiple points throughout development. While his learning curve was very smooth, he had trouble adjusting to the initial difficulty and actively using all the mechanics. With the experience across multiple runs, the impact of the audio was much better, as killing the last enemy is immediately translated into a music change. Based on his selections, the level system was balanced to reward early investment into damage.

Case 4: Lilli *(female, active player, 26)*
While Lilli struggled to grasp the controls, rebinding the dash to shift the gameplay made it easier to control. Here, a bug was also discovered, where the enemies would still play sounds after death, which made it into the submission. She felt a bit frustrated by the hints at some points but still thought they

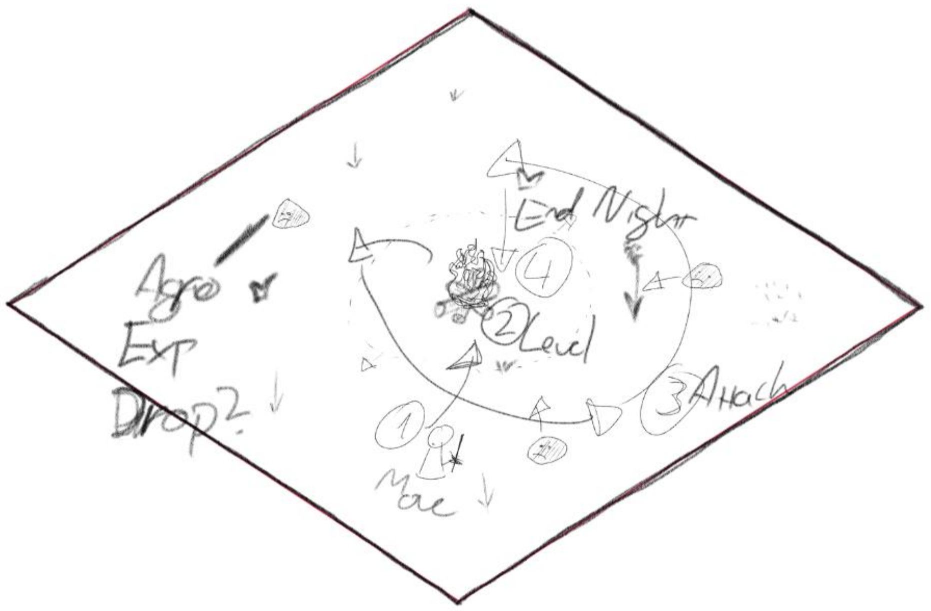
were helpful, just as the dynamic hints that she actively used and reflected on. Lilli suggested expanding on the map, move set, and some other features.

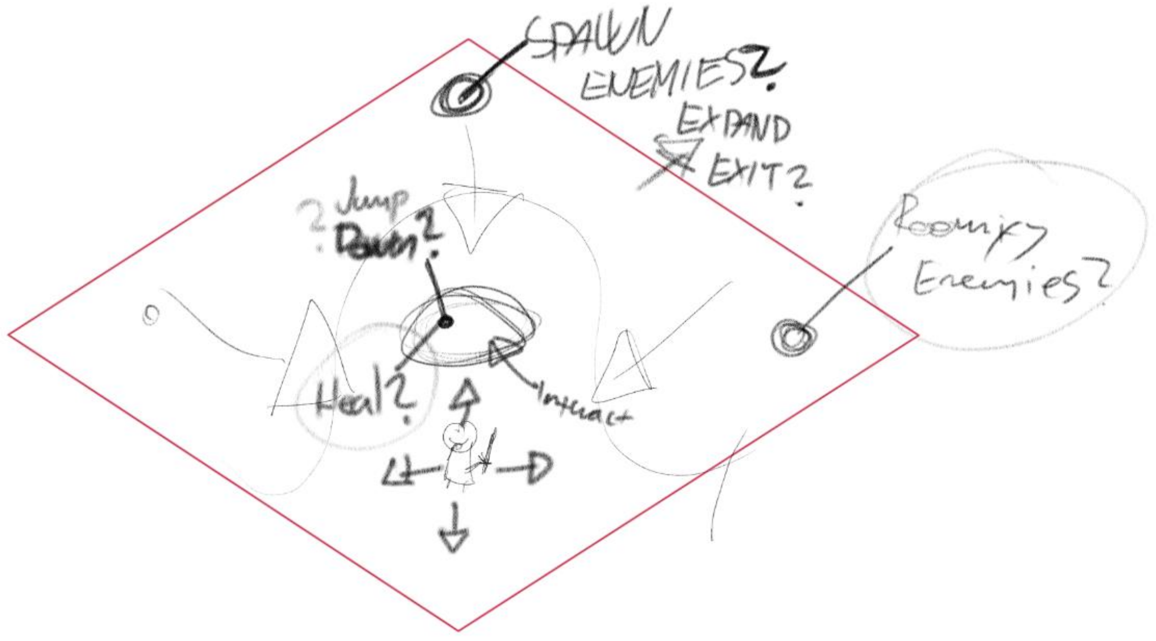
Resources

Mood Board

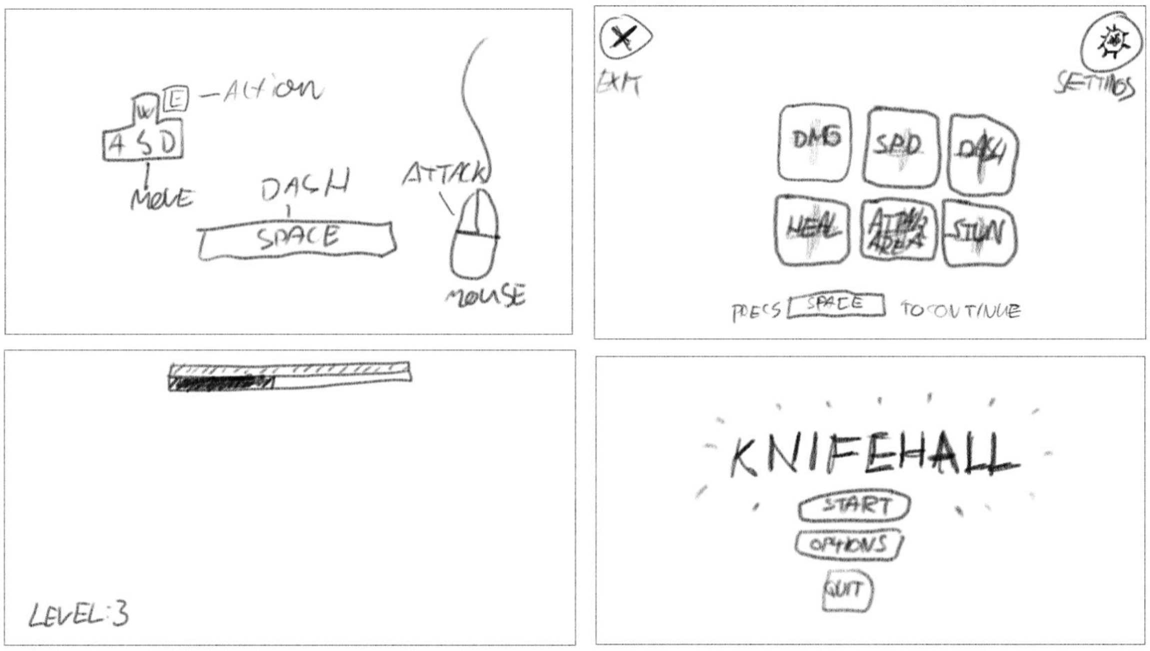
KNIFEHALL
Paper-Cut Isometric Enemy-Hell
VectorStock

Wireframes


Agro
Exp
Drop?
End Night
4
2 Level
3 Attack
1
Move

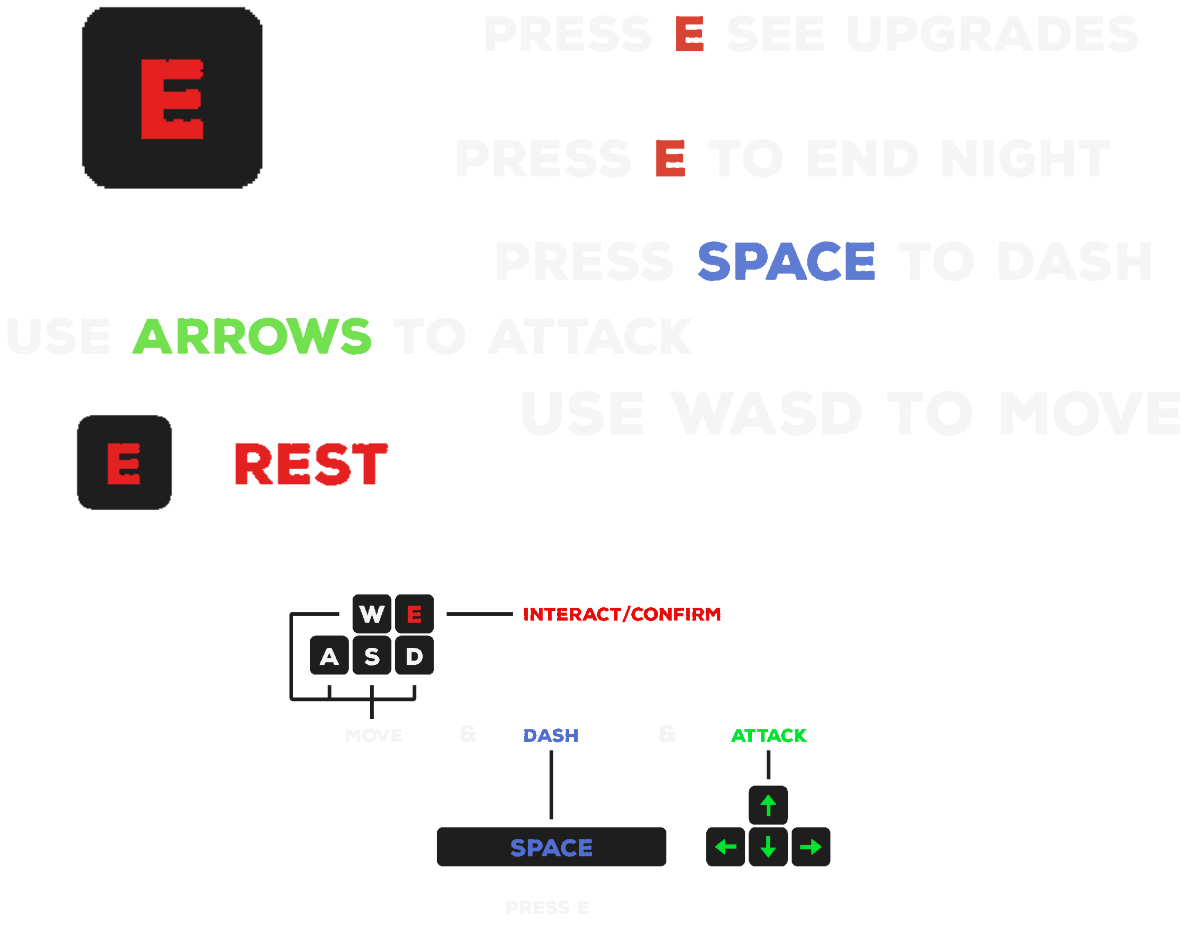

SPAWN
ENEMIES?
EXPAND
~~?~~ EXIT?
Roaming
Enemies?
? Jump
Down?
Heal?
Interact

UI/HUD



Own elements

All Music in the game (not sound effects).



References

Throughout the development process, I used ChatGPT [https://chat.openai.com/chat] and Phind [https://www.phind.com]. Mostly while debugging and searching for the proper functions, but occasionally, as a starting point for code, I would struggle to start on my own.

YouTube tutorials and journals, mainly channels: Randy, Brackeys, Game Makers Toolkit and Pirate Software.

Additionally, parts of the tutorial code have been used for some of the assets downloaded from the Unity Asset Store when available.