Game Proposal: Nighthood

CPSC 427 – Video Game Programming

Team: 12

James Cabaral 31046386
Zongyu (Ricky) Yang 95364667
Linghan (Roger) Qi 98746993
Dylan Williams 29033891
Jiageng(Aaron) Chen 71752323
Aaron Tsang 34709725

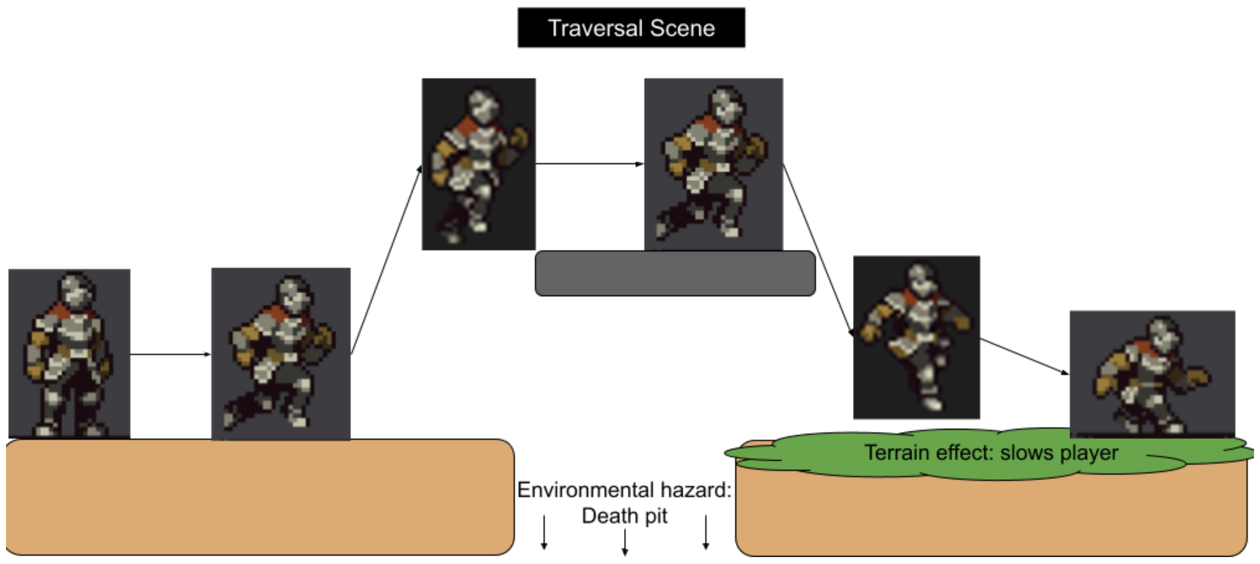
Story:

Nighthood is a 2D platformer that uses a hack and slash melee combat system and Metroidvania style level design.

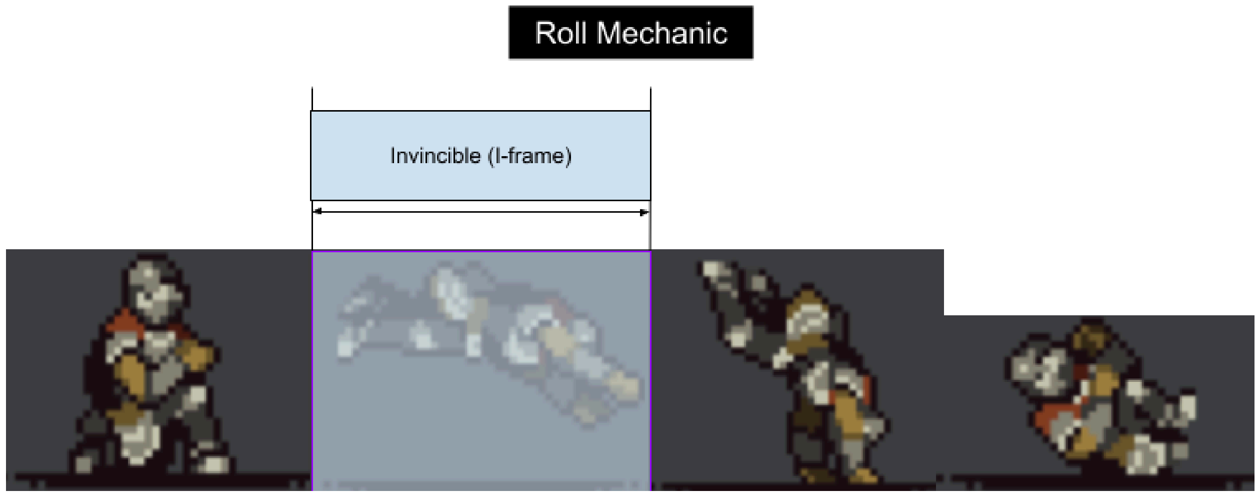
Each level involves the player controlling “the Dishonoured One”. The game’s non-linear levels will require the player to find their own path to the next level while battling enemies and carefully navigating through difficult terrains and environmental hazards. As the player defeats more enemies in a level, the Dishonoured one’s “heroism” level rises which improves the buff given to him in a coming boss fight. Along with the Dishonoured One’s desire to reclaim his honour, this rewards the player for exploration and being brave enough to battle more enemies.

The Dishonoured One can do basic 2D movements (walk left/right and jump), a melee attack and a roll. The roll is a manoeuvre that makes the Dishonoured One invincible for a very brief moment to use as a means for defence or advancing.

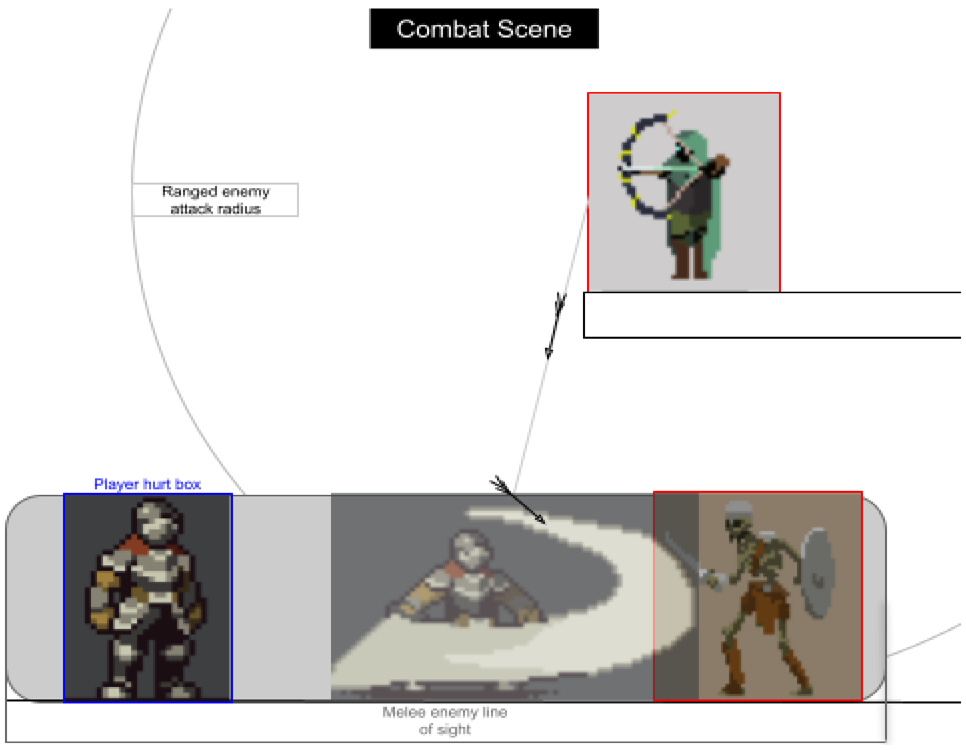
Scenes:



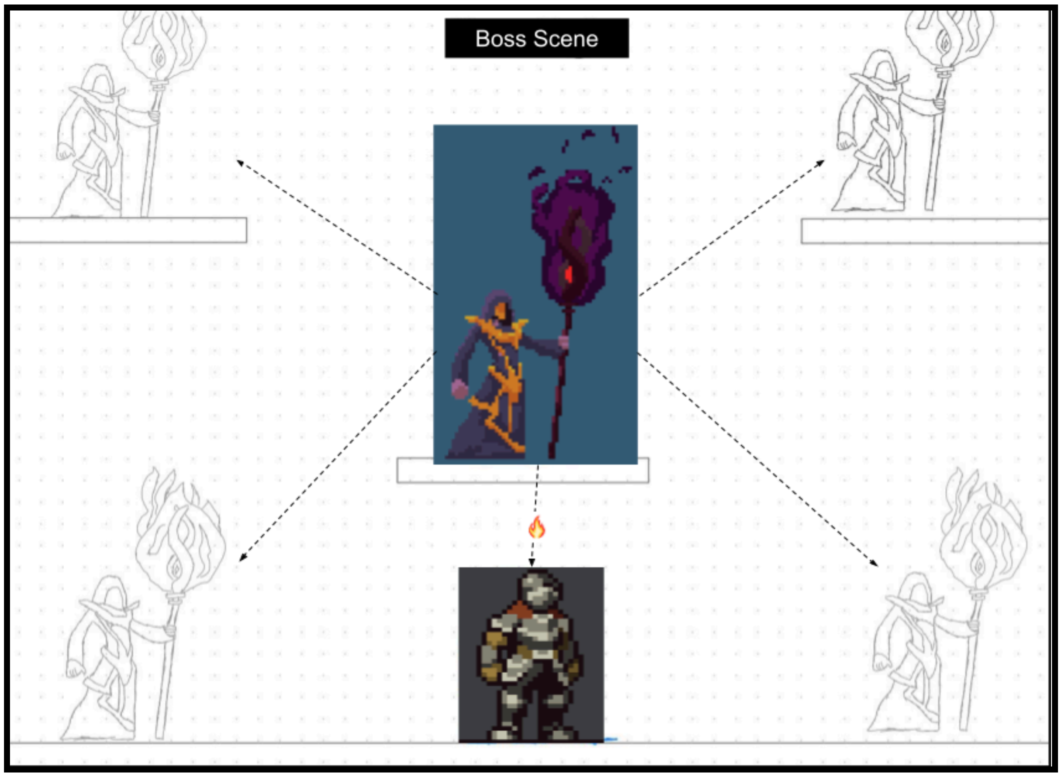
Traversal scene displaying the player's 2D movement and environmental hazards/effects. The player will appear out of screen and dying upon falling into pit. Dishonoured one being slowed will show as a crouched animation.



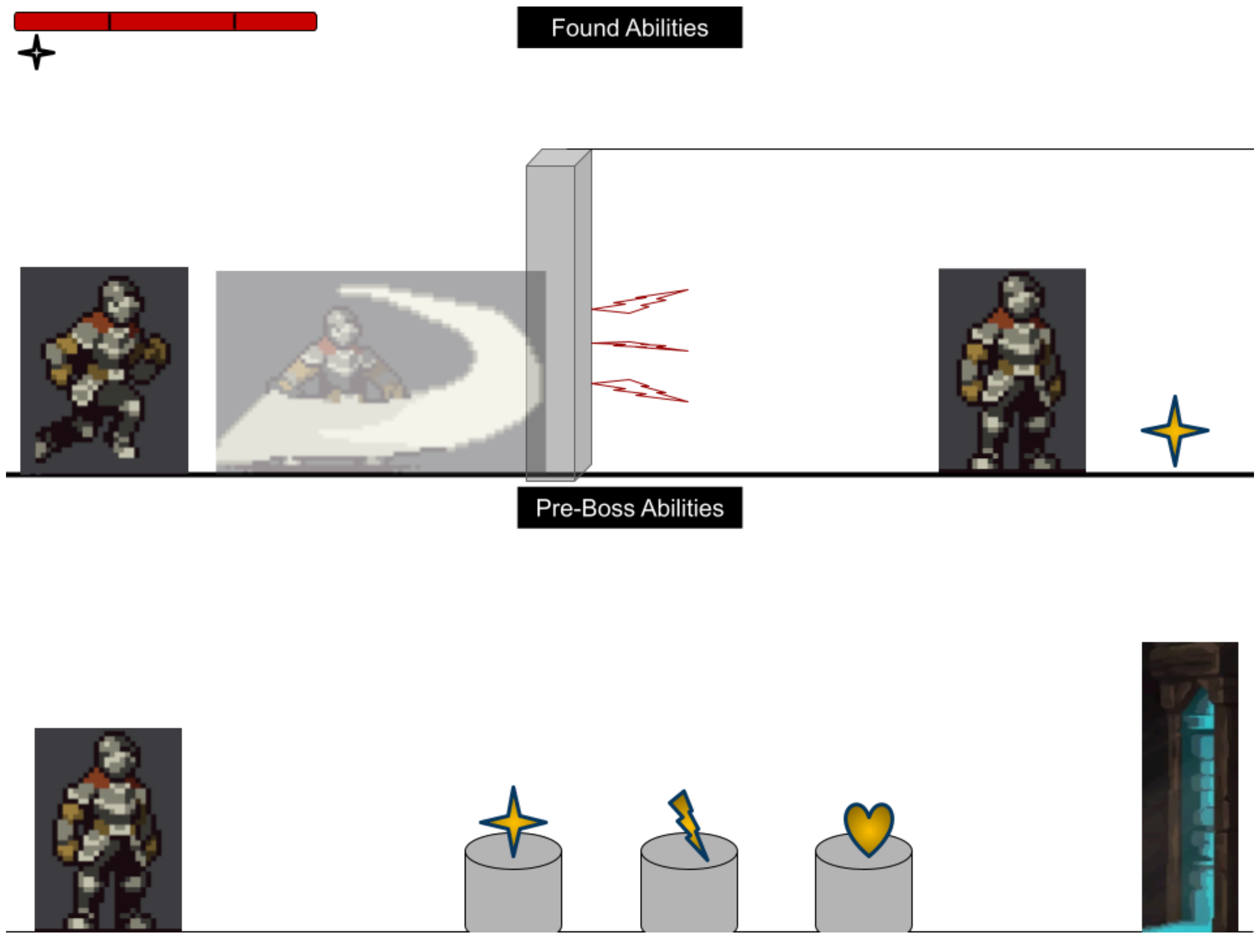
Demonstration of roll mechanic. Player rolls forward to a direction and has a brief moment of invincibility.



Combat scene showing entity hurt boxes and enemy pathfinding. Melee enemies approach player when player is in their line of sight. Ranged enemies shoot at player when they are in their attack radius. Projectiles (like arrows) can be hit for defence.



Boss fires magic projectiles at player. Boss will be in one of five positions (centre, top-left, top-right, bottom-left, bottom-right) and will move after taking a certain amount of damage.



Different abilities can be found in various locations in a level. This scene also shows levels' breakable walls that could lead to abilities. Equipped ability is shown underneath the health bar. There is also a room before the boss fight where the player can equip an ability of their choice for the coming fight.

Technical Elements:

Rendering: 2D Terrain using geometry such as the ground, platforms, drops. The terrain will have obstacle blocks, dead ends, breakable objects/walls. And may have some hidden terrains for some collection goals.

Geometric/Sprite/Other assets: Models and textures for knight, Enemies (Skeleton Melee, Skeleton Range, Boss). Assets for terrain, walls, doors, torches, sword, bow, arrow. May include weather systems to make some buff or debuff for player and enemies.

2D Geometry manipulation: The player character will be able to move around in a 2D plane. We will also have collisions between the player, enemy, arrows, breakable walls/objects.

Gameplay Logic/AI: Logic for the player movement and AI will have logic which allows the AI to target or attack the player.

Physics: Entities will be affected by gravity and will have collision detection between other entities and terrain. Entities such as projectile(s) will have velocity associated.

Audio: Audio effects for moving, jumping, swinging sword, arrows shot by AI, ambient sound/music. As well as background music may be available for the main theme and during the game.

Advanced Technical Elements:

- Attacking projectiles midair which breaks and blocks the projectile.
- Acceleration attached to entities such as projectiles decreasing/increasing velocity or an enemy that ramps up speed.

Devices:

Keyboard Input/Controls:
- W: Look up during attack
- S: Look down during attack
- D: Move left
- A: Move right
- Roll: Shift
- Space: Jump
- Enter: Attack

Tools:

No libraries have been chosen yet besides OpenGL.

Team management:

Our group has agreed to meet on Discord at least twice a week (potentially more if milestone calls for it). Tasks are assigned with respect to the agreed upon roles in the first milestone by categorising the requirements of each milestone evenly.

Assigned tasks will be recorded on a shared document to allow each team member to update their respective parts and for each member to be able to check on each others' progress at any time.

Development Plan:

Week 1:

- Build basic class hierarchy/structure ECS
- Skeletal OpenGL 2D GUI
- Keyboard input detection
- Simple collision detection/resolution
- Rendering
  - Basic 2D transformations
  - Key-frame/state interpolation
- Game testing/bug reporting

Main features to implement: Player entity/sprite, floor/platform, 2D movement from keyboard input.

Week 2

- Rendering
  - Textured geometry
- Coded action
- Stabilise framerate/minimise game lag.
- Define game space boundaries
- Game testing/bug reporting

Main features to implement: Enemy dummy sprite, combat coded action, finalise level and graphics.

Milestone 2: Minimal Playability

Week 1

- Basic Enemy AI
- Embed basic assets
- More physics for elements

Week 2

- Game menu
- Round 1 setup

Milestone 3: Playability

Week 1

- Complex Enemy AI
- Round 2,3 Setup

Week 2

- Game progress save

Milestone 4: Final Game

Week 1

- Effects for all interaction
- Gameplay testing & debug

Week 2

- Gameplay testing & debug