Dorset College Dublin

Game Design - BSC30924

CA4 - Vertical Slice Game Prototype

Final Report

Shadow Maze: Ghost Escape

Filipe Lutz Mariano - 25956

Vinicius Miranda - 70973

Lecture: **Seamus Hickey**

Table of Contents

1. Introduction

Shadow Maze: Ghost Escape is a suspenseful 3D exploration and stealth game built in Unity, where players must navigate a haunted maze before their time runs out. It is designed as a vertical slice prototype that captures a focused and polished segment of intended full gameplay. The player moves in darkness with a limited light radius, evading patrolling enemies while racing against a ticking clock.

This project represents a leap from the original 2D pixel-art concept into an atmospheric 3D experience. Core systems were reimagined and adapted for Unity's 3D environment, including procedural maze generation, enemies, audio-visual feedback systems, and modular UI interfaces. Special attention was given to creating immersive tension through timed objectives, lighting, and sound design.

---

2. Gameplay Summary

- **Goal:** Escape the maze before the timer expires.
- **Challenge:** Avoid ghosts and gargoyles; if caught, the game ends.
- **Time Pressure:** Start with 3:10 on the clock. The first 10 seconds show a map overlay to help orient the player.
- **Escape:** Reach the exit in time to win.

---

3. Key Mechanics

1. **Player Movement:** Controlled with WASD or Arrow keys.
2. **Timer:** Starts at 3:10. The maze map is visible for the first 10 seconds, then disappears and UI Game Menu Panel appears.
3. **Ghosts & Gargoyles:**
    - Ghosts patrol and trigger game over on contact.
    - Gargoyles stand still, scanning the maze with lanterns. If a light ray detects the player, they are caught and it is game over.

4. **Collectibles:** Initially planned Time & Immunity Boxes but excluded to preserve balance.
5. **Mini Map:** Replaced by a static overlay shown at game start.
6. **Menus:**
    - Main Menu includes Start and Quit options.
    - Game Menu appears after 10 seconds: Restart, Pause/Resume, Exit.
7. **Sound FX:** Custom and sourced sounds for footsteps, ambient tension, and UI feedback.

3.1 Unique Time-Based Mechanics

- **Start Time:** 03:10.
- **First 10 Seconds:** Map overlay is visible.
- **At 3:00:** Map disappears, and the Game Menu panel becomes visible.
- **Timer Colour Changes:**
    - Green first 10 seconds
    - White normal gameplay
    - Red below 30 seconds
- **Final 10 Seconds:**
    - Timer blinks
    - A warning sound plays once (sourced from Pixabay)

---

4. Game Elements

Theme: Escape a cursed, dimly lit maze while being hunted by supernatural guardians. Only cunning, timing, and awareness will keep the player alive.

Characters:

- **Player:** A cat/human-like figure with a restricted vision radius.
- **Ghosts:** Patrol corridors and trigger game over on collision.
- **Gargoyles:** Static watchers with lanterns that scan for the player using raycasts.

5. Assets

- **3D Models:** From *John Lemon's Haunted Jaunt* (Unity Learn).
- **Maze Generation:** Adapted from *Catlike Coding Maze Tutorial* and *John Lemon's Haunted Jaunt*.
- **UI & Timer:** Based on Unity UI Toolkit examples.
- **Sound FX:** Mixed sources from Pixabay and Unity Asset Store.
- **UI Icons:** Default Unity UI assets and minimalistic icons.

---

6. UI Overview

6.1 In-Game UI

- Timer (TextMeshPro) displayed at the top left.
- Map overlay visible for the first 10 seconds.
- The Game Menu panel appears after the map disappears.

6.2 Menus

- **Main Menu:** Start, Quit.
- **Game Menu Panel:** Restart, Pause/Resume, Exit to Main Menu.
- **Game Over Screen:** Triggers fade, plays sound, then redirects.

---

7. Game Backlog Implementation

- **Maze Layout:** Fully implemented via procedural logic.
- **Player Movement:** Complete, animated, and includes audio feedback.
- **Ghost and Gargoyle:** Implemented with raycasting detection.
- **UI Menus:** Fully functional with button logic.
- **Collectibles:** Skipped for better gameplay balance.
- **Mini Map:** Replaced by a static overlay shown at game start.
- **Time Mechanics:** Fully implemented, with colour change, sound, and blink.
- **Game Over Logic:** Fade, sound, and scene transition.
- **Sound System:** Includes pause-aware audio source control.

7.1 Development Workflow

Throughout the development of *Shadow Maze: Ghost Escape*, task planning and production tracking were organised using a collaborative Trello board.

- **Trello Workspace Board - Shadow Maze: Ghost Escape**

The board was used to maintain the game backlog, plan tasks, and track progress through To Do, Doing, and Done stages. It ensured effective teamwork and scope control during the vertical slice development.

---

8. Challenges and Difficulties

1. **Balancing Difficulty:**
   Originally planned power-ups like time bonuses and immunity boxes made the game too easy. These were removed to maintain pressure and pacing.
2. **Font Visibility in Build:**
   Text rendering inconsistencies in Unity builds led to migrating from legacy Text to TextMeshPro, which ensured proper font bundling and styling.
3. **UI Scaling & Layering:**
   Some UI elements disappeared in builds due to canvas sorting order or incorrect layer setups. This was resolved by enforcing consistent canvas settings and screen space scaling.
4. **Audio Pausing:**
   Not all audio sources paused correctly when the game was paused. This was solved by using FindObjectsOfType<AudioSource>() to globally pause/resume active audio.

9. Conclusion

Shadow Maze: Ghost Escape delivers an immersive, polished vertical slice game experience. It features tightly scoped design, dynamic audio/visual feedback, and systems that build suspense through time pressure and limited visibility. While some original features were removed for balance, the result is a cohesive and engaging prototype that could easily be expanded into a larger game.

The project demonstrates strong command over Unity's systems for animation, lighting, and UI, and highlights a sharp understanding of scope control in production. This vertical slice prototype is fully playable and primed for user testing or future development.

---

10. References

1. **Make a Health Bar with UI Toolkit - Unity Learn**
   https://learn.unity.com/tutorial/make-health-bar-with-UItoolkit?start=true#
2. **John Lemon's Haunted Jaunt: 3D Beginner**
   https://learn.unity.com/project/john-lemon-s-haunted-jaunt-3d-beginner?uv=2020.3
3. **Maze Tutorial - Catlike Coding**
   https://catlikecoding.com/unity/tutorials/maze/
4. **Unity Asset Store**
   https://assetstore.unity.com/
5. **Pixabay - Time Warning Sound Effects**
   https://pixabay.com/sound-effects/search/time%20warning/