Software Requirements and Design Document

For

Group <3>

Version 1.0

Authors:
Grace Brill
Elias Diez
Jason Graham
Miguel Malayto
Santiago Ruiz

1. Overview (5 points)

The purpose of the project is to develop a first person parkour game called Owl's Omen. The game revolves around the plot of the main character, Nim, who is an acrobat in a bankrupt circus. Nim is approached by a pair of wealthy siblings about smuggling goods into the poorer area of the city to help out those in need. As the game progresses, these smuggling missions get riskier and more challenging. The system is designed around this plot and incorporates features to mimic acrobatic movements.
The movement system allows for traditional WASD movement, with jumps. It also allows for launching off of any surface and pulling towards any surface (to/from a specific pointed-to location in the surface), as well as "recovery jumps" that can be done mid-air to regain push/pull charges at the cost of mid-air directional adjustments for some time.

There are surfaces that affect movement by: increasing player speed (unidirectionally), slowing down player speed (multi-directionally), and providing the player with a jump boost given that they are already jumping. The game also includes a rudimentary dialogue system, which allows the main character to interact with NPCs. This game also includes a functioning enemy AI that can track and follow the main character. This enemy AI can not only track the main player, but it can also patrol points spread around the map and also has a character design fitting the world and some animation. The game also features a HUD with various UI elements that include (but are not limited to) a boost cooldown meter, push/pull charge capacity, distance-to-goal indicator, and a functional minimap.

The game also includes a Main Menu that will load the game (any scene spacified), open settings(same scene, new panel), and quit the game (ends application). The settings menu allows users to change technical options like the volume, resolution, graphics, and toggle fullscreen. There is also a Pause Menu that pauses the game and allows the user to resume, return to the menu or quit the game.

2. Functional Requirements (10 points)

1. Player movement system (Priority: High) (in the following, 'x' is used to mean an arbitrary value exposed in the inspector to be tuned when balancing the game, each time x is mention it is a different arbitrary value)
    a. Movement abilities while on ground:
        i. Move forwards/backwards/left/right at speed x based on first-person view on using W/S/A/D keys
        ii. Press space to jump into the air by x amount
        iii. Click on a point at a surface that is no more than x meters away from the player to receive a push in the opposite direction they are facing with force x or towards where they are facing with force x.
            1. Limited by x amount of charges, which are shared between pushes/pulls
            2. Controls: Push down LMB to aim a push off a surface, release to execute the push. Push down RMB to aim a pull onto a surface, release to execute the pull. If the player presses RMB while LMB is pressed down, the force is changed to match RMB, and vice-versa.
        iv. When the player lands, their charges are reset.
    b. Movement abilities while on air
        i. Same as ground i, but wasd movement has slight smoothing applied such that it is easier to land on smaller platforms
        ii. Pressing space instead triggers a "leap" angled in the current movement direction, until the player touches the ground or uses another boost, their wasd

movement has a lower effect than normal, such that they lose a significant amount of control and must instead plan before leaping.
            1. Doing this has the benefit of gaining some distance, but primarily it resets the charges the player has
            2. This is on a cooldown
        iii. Same as on-ground.
    c. Ground momentum conservation
        i. Some airspeed (if not all) will be retained by the player upon landing on the ground given that they do not take any hard turns (anything higher than x degrees over y time) or stop moving (holding down whichever wasd key corresponds to their current momentum vector)
2. First person arm animation: (Priority: Medium)
    a. The staff model that the player holds sways with acceleration to give feedback on current speed to the player
    b. The staff model also angles itself when looking down or up by an amount equal to the degree the player is looking down/up by
    c. The staff model should spin to point towards the current surface whenever the player is aiming
    d. Staff is a diegetic UI, meaning that the information normally displayed in the UI overlay is instead displayed “in-world” as parts of the staff.
        i. Charge counter is at the tip of it, lunge meter is a charging segment of it
    e. Headbob
        i. Gets more intense the faster the player is moving on the ground
3. General VFX: (Priority: Low)
    a. Boosts to/from places, colorful explosions
    b. Leaps, some air effect around the screen to show the speed
    c. High speed, a general-purpose way to indicate the user is moving over x speed, could just be increased screenshake.
4. Main level (Priority: High)
    a. The level where the missions will take place, this level should accommodate for many interesting locations and interesting / diverse paths between them.
5. Speed Boost Surface (Priority: Medium)
    a. When the player enters the speed boost region, they receive a unidirectional speed boost. The player must enter from the correct side of the speed boost in order to receive the boost.
    b. The speed boost surface is denoted by the blue spinning vortex animation, which is moving in the direction of the speed boost.
    c. The rationale of this feature is to provide players with a new layer of strategy when playing the game and to allow them more choices with their motion.
6. Slow Down Surface (Priority: Medium)
    a. When the player enters the slow down region, they receive a multidirectional slow down, meaning that no matter what direction they move in (x,y, or z), their velocity is reduced.
    b. The slow down region is denoted by a pulsating red animation.
    c. The purpose of this feature is to provide a challenge for users when navigating the parkour environment. Players will have to alter their movement and pay attention to the environment to avoid the slow regions.
7. Jump Boost Surface (Priority: Medium)
    a. The jump boost surface provides a player that is already jumping with an additional boost. This means that the player’s velocity is checked and if it is greater than -10 m/s,

    then the player receives an additional jump boost. Otherwise, the player receives no jump boost.
    b. The jump boost surface is denoted by a white box with a purple lightning animation.
    c. The purpose of this feature is to add a new layer of strategy for users when jumping. Users will have to aim for a specific landing spot in order to receive the boost, which could help them jump over obstacles and navigate the environment more easily.

8. Dialogue (Priority: High)
    a. A basic dialogue system is being developed. In its early stages, it involves a Canvas, which will contain all UI elements. In the Canvas, there is an image, which is a white rectangle, and the text.
    b. To show the text that you want shown, create a dialogue object with elements that contain the text you want shown.
    c. Currently, the text is written out in the typewriter style, meaning that letters are written one at a time. This feature can be customized by changing the writing speed.
    d. This requirement was derived from the idea of having the user interact with non-player characters (NPCs) to make the game more engaging. Having a dialogue system is necessary for the game to feel more engaging and to incorporate the plot of the game, which is also in the process of being developed.
9. Menus (Priority: High)
    a. Maine menu, settings, and pause menu have been implemented.
    b. Fix any UI or on pause bugs with the pause menu as new features are added.
10. Loading Screen (Priority: Low)
    a. This will be a scene transition between all scene loads to make the transition less jarring.
11. Audio (Priority: Medium)
    a. Implement basic sound effects and music in the menu and to the player.
    b. Add music and click noises to buttons on the main menu and pause menu.
    c. Add music and player noises to ingame.
12. Save/Load System (Priority: High)
    a. Implement the save and load buttons in the pause menu and the load button in the main menu.
    b. Add a panel with multiple save slots to the main menu.
    c. When the save button is pressed the player's level, health, position, and cooldown status will be saved.
13. Enemy AI (Priority: High)
    a. Track player and go around obstacles and buildings efficiently.
    b. Enemy patrols different patrol points randomly.
    c. Find enemy within a vision cone and efficiently chase the player
    d. Have enemy turn from movement animation to attack animation when approaching player
    e. Have enemy react to environmental variables
14. HUD elements (Priority: High)
    a. Contain UI elements that visualize manageable resources (ex. ammo).
    b. Contain functional minimap
        i. Follow player movement + rotation.
        ii. Render terrain objects as simple sprites (icons) for map layout.
        iii. Render enemies, goals, and other game objects as icons.
    c. Contain indicator that measures distance from goal and is seeable through all objects.
    d. Add more elements as game development progresses.

3. Non-functional Requirements (10 points)

Runs at a minimum of 60fps on any normal gaming rig.

Inputs are responsive in that any input will cancel any current action that would conflict with it (for example, clicking somewhere to boost off a surface should not lock player into a “launching off the ground” animation that is impossible to cancel, any other boost should cancel the animation of the current one.) In other words, inputs over animations / visuals--ideally both in harmony (VFX that “feel right” even with twitchy inputs).

6. Operating Environment (5 points)

The software is built on Unity 2021.1.1f1 to operate in MacOS, Windows (primarily), and Linux systems.

7. Assumptions and Dependencies (5 points)

TextMeshPro
Odin (https://odininspector.com/)
Octave3D (https://assetstore.unity.com/packages/tools/level-design/octave3d-level-design-45021)
Other Unity Store packs (primarily for visual assets and FX).