# Project: MOVEMENT ROGUELITE FPS

## 1. Game Summary

**Project Title (Working):** *TBD*

**Concept:** A fast-paced, single-player, first-person arena shooter with roguelite progression. The game emphasizes smooth, expressive movement and simple but satisfying gunplay.

**Inspirations:** The core gameplay feel aims for the fluid movement of **ULTRAKILL**, while the aesthetic is a surreal, psychedelic style reminiscent of **Cruelty Squad**, characterized by low-fidelity models and vibrant, uncohesive textures.

**Player Experience:** The player is an agile combatant thrown into a series of surreal arenas. They must survive waves of enemies using skillful movement and gunplay. After clearing each arena, they choose a temporary upgrade for that run. Completing runs unlocks new weapons, abilities, and harder difficulties for future attempts, encouraging high replayability.

## 2. Core Design Pillars

- **Fluid Movement is King:** The primary focus is making the player feel powerful and agile. Movement should be intuitive, responsive, and the core skill for players to master for both offense and defense.
- **Fast-Paced, Readable Combat:** Combat should be quick and intense. Enemies are simple, but their numbers and placement create challenges that force the player to stay on the move.
- **High Replayability:** Through procedural level sequences, in-run upgrades, and permanent meta-unlocks, no two runs are exactly the same. The goal is to create a loop that makes players want to say "just one more run."
- **Achievable Scope:** This plan is designed to be completed within a 4-week timeframe by a 4-person team. We will prioritize a polished vertical slice over a large quantity of unpolished features.

## 3. Gameplay Mechanics

### Movement (Priority #1)

- **Base:** Walk, Sprint, Jump, Momentum-based physics.
- **Core Mechanics:**
  - **Slide:** Perform a slide on any surface that preserves and builds momentum.
  - **Slide Jump:** Jumping out of a slide provides a significant speed and height boost.
  - **Dash:** A short, directional burst of speed on a cooldown.
- **Advanced Mechanics:**
  - **Wall Jump / Wall Kick**: The ability to jump off walls to gain height and change direction.
- **Stretch Goal (To be implemented only if time permits):**
  - **Grapple Hook:** A utility for rapid traversal across arenas.

### Combat

- **Gunplay:** Simple and satisfying point-and-shoot mechanics.
- **Weapons (MVP Goal: 2-3):**
  1. **Pistol:** A reliable, basic starting weapon.
  2. **Shotgun:** A high-damage, close-quarters weapon.
  3. **Machine Gun:** A rapid-fire weapon effective at medium range.
- **Enemies (MVP Goal: 2):**
  1. **Melee Grunt:** Moves towards the player and attacks on contact.
  2. **Ranged Turret:** A stationary enemy that fires slow-moving projectiles.

## 4. Progression Systems

### Run Structure

A single "run" consists of the player progressing through a sequence of **10 arena-levels**.

1. Player starts a run, choosing from unlocked starting gear.
2. Enter Arena -> Survive enemy waves.
3. Arena Cleared -> A UI appears, offering a choice of 3 random, temporary upgrades.
4. Player selects an upgrade and moves to the next arena.
5. After the 10th level, the run is "won." The game can loop for an infinite mode or end.

### In-Run Upgrades (Roguelite)

These are temporary buffs that reset after each run. Examples:

- `+10% Movement Speed`
- `+25% Max Health`
- `+1 Additional Dash Charge`
- `20% Faster Weapon Switching`

### Meta-Progression (Permanent)

This system encourages long-term play and is saved between sessions using `PlayerPrefs` in WebGL.

- **Weapon Unlocks:** Reaching certain milestones (e.g., "clear level 5 for the first time") permanently unlocks new weapons for the starting loadout.
- **Ability Unlocks:** Unlocks new starting "Perks" (e.g., Start with more health, Start with an extra dash).
- **Difficulty Modifiers:** Beating the game on "Standard" difficulty unlocks "Hard," which in turn could unlock "Insane." These modes could modify enemy health, speed, or damage.

# 5. Technical Details

- **Engine:** Unity
- **Platform:** WebGL (for hosting on Unity Play)
- **Version Control:** Git & GitHub
- **Data Persistence:** `PlayerPrefs` for saving meta-progression.

# 6. 4-Week Project Timeline

**Team:**

- **Programming:** 2 (CS Students)
- **Art/Level Design:** 2 (Art Student, Art Student w/ CS Minor)

---

### Week 0: Setup (ASAP)

- **All:**
  - Create the GitHub repository.
  - Establish project structure (e.g., `_Scripts`, `_Prefabs`, `_Art`, `_Scenes` folders).
  - Ensure everyone can clone, commit, and push to the repository. **Do a test commit for everyone.**

---

### Week 1: Foundation & Prototyping

- **Goal:** Get the player moving and establish the visual style.
- **Programming (2):**
  - **Primary Task:** Develop the core `PlayerCharacterController`. Focus on making movement feel good in a test scene with simple blocks, ramps, and walls.
  - Implement: Walk, Sprint, Jump, Slide, Slide Jump, Dash.
  - Set up the basic first-person camera.
- **Art/Level Design (2):**
  - **Primary Task:** Define the art style. Create texture swatches and mood boards.
  - Create a "modular kit" of simple environmental assets (floors, walls, ramps, platforms).
  - Design and model a simple placeholder for the Melee Grunt enemy.
  - The Art/CS student can assist with setting up the test scene in Unity.

---

### Week 2: Core Loop Implementation

- **Goal:** Create a playable, end-to-end gameplay loop.
- **Programming (2):**
  - Implement a basic weapon system. Start with the Pistol.
  - Create the simple Melee Grunt AI (pathfinds to player, attacks).
  - Implement the core game loop logic: spawn enemies, check for wave clear, trigger "level end."
  - Start work on the Wall Jump mechanic.
- **Art/Level Design (2):**
  - **Primary Task:** Use the modular kit to build **3-4 polished arena-levels**.
  - Finalize textures and apply them to the kit pieces.
  - Model and texture the first weapon (Pistol).
  - Animate the Melee Grunt (walk, attack).

---

### Week 3: Systems & Content Expansion

- **Goal:** Build out the progression systems and add more content.
- **Programming (2):**
  - **Primary Task:** Implement the **In-Run Upgrade System** (UI, logic to apply stats).
  - Implement the **Meta-Progression System** using `PlayerPrefs` for unlocks.
  - Add the second weapon (e.g., Shotgun).
  - Add the Ranged Turret enemy.
- **Art/Level Design (2):**
  - **Primary Task:** Build the remaining arena-levels to reach the goal of **7-10 total**.
  - Model and texture the Shotgun.
  - Create the Ranged Turret model.
  - Create basic UI assets for the Main Menu, HUD (Health, Ammo), and Upgrade Selection screen.

---

## Week 4: Polish, Audio & Shipping

- **Goal:** Bug fix, add final touches, and prepare the build. **NO NEW FEATURES!**
- **Programming (2):**
  - **Primary Task:** Bug fixing. Playtest relentlessly and address issues.
  - Implement audio hooks ( `PlaySound("Player_Jump")` ) and work with the art team to add sound effects.
  - Implement the Main Menu and scene transitions.
  - Optimize performance for the WebGL build.
- **Art/Level Design (2):**
  - **Primary Task:** Polish. Tweak lighting, post-processing, and level layouts for maximum fun.
  - Find and implement royalty-free sound effects for jumps, weapons, enemy hits, etc.
  - Create a simple background music track if time allows.
  - Help with extensive playtesting.
- **Final Task (All):** Build the game for WebGL, upload to Unity Play, and write the final report/presentation.