C.L.O.E. & Finch Top

This is the top-level page for C.L.O.E. & Finch, containing all development content.

📖 DESIGN

Summary

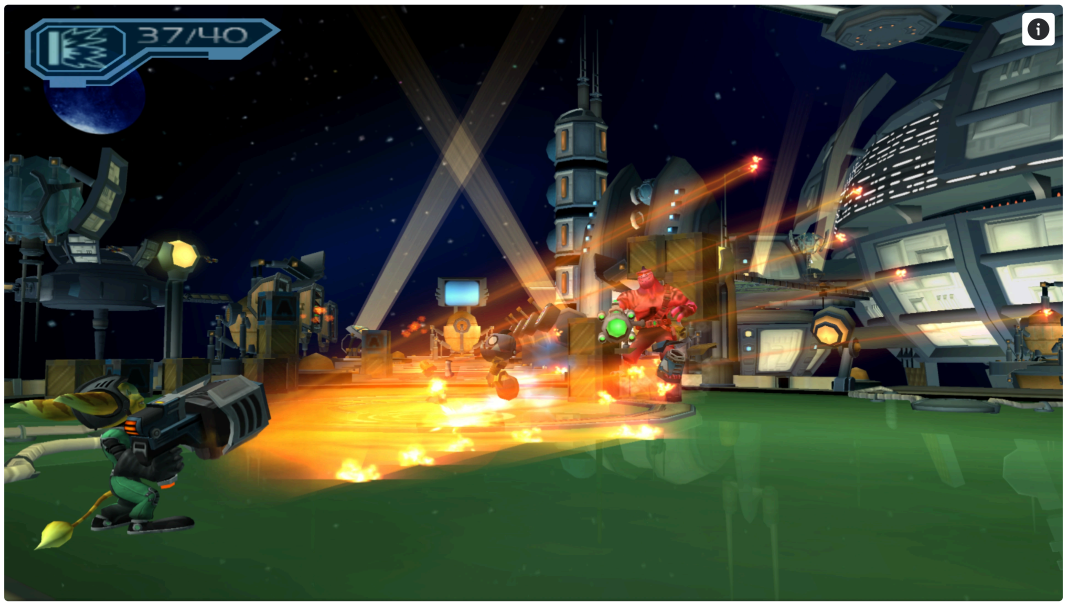
C.L.O.E. & Finch is a third-person action-adventure game that blends fast-paced combat, platforming, and exploration across a wide variety of handcrafted levels. Players control C.L.O.E., a highly agile combat android, and her companion Finch, and are tasked with traversing levels while switching between weapons and gadgets to overcome environmental puzzles, defeat enemies, and uncover secrets hidden in the ruins of an over-corporatized world.

Combat is dynamic and arena-based, featuring reactive enemy AI, a growing arsenal of upgradable weapons, and gadgets that open new paths for the player to explore. Each new zone introduces distinct challenges, unique enemies in both appearance and behavior, and various traversal mechanics, from gravity-defying rooftops to claustrophobic factory interiors. Outside of combat, players use C.L.O.E.'s mobility and Finch's clever tools to solve puzzles, unlock hidden paths, and uncover pieces of the world's story.

Inspiration/Similar Games

Ratchet & Clank: Going Commando

is an action-platformer developed by Insomniac Games that blends fast-paced third-person shooting with light RPG elements, featuring upgradable weapons, character progression, and diverse level-based combat arenas. As this game is serving as my prototype's main inspiration, there are a few things I plan to replicate: The player movement and camera, the feel of combat, platforming elements, and, in ways, the over-arching theme surrounding mega-corporations. Below is an in-game screenshot from Fandom that shows Ratchet firing the shotgun-type Blitz Gun in combat.



Titanfall 2

is a first-person shooter developed by Respawn Entertainment that seamlessly combines high-speed infantry combat with large-scale mech warfare, blending tight gunplay, parkour movement, and cinematic storytelling into a fluid, genre-bending experience. I plan to replicate some of the game's traversal mechanics a bit further down the line if I can (i.e., wallrunning), but the main "thing" I want to pull from Titanfall 2 is a bit more vague. I want to replicate the feeling of companionship between BT and the pilot. Below I've included a promotional screenshot from.. RedBull? Showing both the wallrunning *and* the companionship.

The Last of Us: Part Two

is a narrative-driven action-adventure game developed by Naughty Dog that merges stealth, survival, and third-person combat with emotionally complex storytelling, creating a harrowing and immersive post-apocalyptic experience. While it may seem like a bit of a curveball, I do intend to use the tense, slower-paced stealth sections as inspiration for parts of my project. Just like in Ratchet & Clank, I intend to have sections of the game playable as the side character, it's here that I plan to use the stealth elements from The Last of Us, namely the player's ability to crawl/hide in tall grass/bushes and stealth attack enemies from behind. Below is an in-game screenshot showing the grass mechanic from IGN.

Ghost of Tsushima

is an open-world action-adventure game developed by Sucker Punch Productions that blends cinematic storytelling with fluid sword combat and stealth mechanics. Set in feudal Japan during the Mongol invasion, it immerses players in a richly detailed world where every decision—whether made in battle or through exploration—shapes the journey of Jin Sakai, a samurai torn between tradition and survival. With a strong emphasis on atmosphere, player expression, and honor-versus-necessity themes, the game delivers both visceral action and emotional depth. I plan to base the remaining inspired mechanics on those found in this game. The remaining stealth mechanics for the companion, the world/level design, traversal mechanics for the main character, and the mood, all just brought over to an over-corporatized, futuristic, and alien setting.

Design Pillars

1. Arena-Based Combat

Summary:
Combat takes place in tightly constructed, enemy-rich zones designed for dynamic encounters that challenge the player's ability to adapt, evade, and respond using both ranged and melee tools. The game alternates between moments of intensity and calm, with "refuge trails" providing breathing room and narrative pacing.

Supporting Mechanics:

- C.L.O.E. uses a mix of melee and ranged weapons (with unique functionality)
- Finch engages in stealth combat using gadgets and distractions
- Combat zones are handcrafted and distinct, with varied enemy types and hazards
- Health drops, ammo, and buffs are found in combat arenas
- Cover-based shooting
- Arena pacing: intense clusters followed by downtime/exploration paths ("refuge")
- Dynamic enemy AI (some fight each other, allowing for strategy)
- Environmental hazards (electric floors, toxic waste, machinery)

---

2. Companionship

Summary:
The bond between C.L.O.E. and Finch is at the heart of gameplay and story. Their differences, both physically and emotionally, create gameplay synergy. Emphasizing the need for mutual support and highlighting their relationship through both mechanics and narrative.

Supporting Mechanics:

- Combined abilities (e.g., Finch rides on C.L.O.E. to access tight areas or hack while she protects him)
- Partner-based puzzle solving (dual switches, climb assists, etc.)
- Tag-team sections where players control each character at different times
- Companion banter that evolves based on story events
- Upgrade system where Finch improves C.L.O.E.'s systems/weapons
- Sections that require teamwork to bypass otherwise inaccessible areas
- Emotional moments between the two that impact gameplay or unlock abilities

---

3. Introspective & Thematic Storytelling

Summary:
Despite the whimsical tone, the game wrestles with themes like worker exploitation, pollution, class divide, and self-identity. These are told through characters, environmental storytelling, enemy factions, and reactive dialogue that encourage reflection while keeping things entertaining.

Supporting Mechanics:

- Dialogue and worldbuilding that reflect dystopian industrial life
- Levels layered with environmental storytelling (protest graffiti, propaganda signs, ruined homes)
- Enemies that represent different factions or ideologies (e.g., rival corporate security, rebellious workers)
- Moral grey areas in story events and choices
- Flashback sequences revealing each character’s emotional backstory
- Comical banter that occasionally shifts into serious introspection
- Situational irony and dark humor to contrast the absurdity of the world

---

4. Momentum-Driven Traversal/Gameplay

Summary:
Movement and exploration are built around flow and verticality. The world is layered and reactive, encouraging fluid transitions between traversal, stealth, and combat. Each character moves differently, reinforcing identity and rhythm.

Supporting Mechanics:

- C.L.O.E. has a heavy sprint, jump boosts, and a glide/slowfall function
- Finch is nimble; can high jump, cling to walls, and crawl through vents
- Momentum paths: ramps, launch pads, wall runs, and chainable traversal sections
- “Refuge trails” between combat arenas serve as quiet traversal puzzles or story delivery routes
- Traversal puzzles: environmental manipulation, power rerouting, lift systems
- Optional collectibles that reward exploration and creative movement
- Some areas emphasize stealth movement or timing (for Finch especially)

Feature Breakdowns

Player Avatar: C.L.O.E.

Priority: Highest

Objective:

Establish C.L.O.E. as the player’s primary avatar with responsive movement, intuitive inputs, and a camera system that supports both exploration and combat in a dynamic third-person perspective.

Summary:

C.L.O.E. is the main playable character, built for mobility, precision, and direct action. Her avatar blends tight third-person controls with responsive camera behavior to create a smooth gameplay experience. Whether navigating complex environments or facing enemies in fast-paced arena encounters, C.L.O.E. must feel like an extension of the player’s intent. Her movement suite supports the game’s core combat and traversal mechanics, while the camera adapts intelligently to context to ensure clarity and control at all times.

Core Mechanics:

- **Movement:**
  - **Walk/Run:** Analog stick movement; run by default.
  - **Jump:** Standard jump with double jump.
  - **Dodge/Roll:** Quick evade with invincibility frames and directional control.
  - **Climb/Grab Ledges:** Auto-climb when near ledges; manual input for longer climbs.
  - **Interact:** As often as possible, use the player's movement system. (pressure plates, detection activation, or have the player shoot/melee to interact)
- **Input:**
  - **Controller & Keyboard/Mouse Support:** Unified input scheme.
  - **Remappable Controls:** Full customization for accessibility.
  - **Visual Feedback:** Button prompts for interactables and context-sensitive actions.
- **Camera:**
  - **Third-Person Follow:** Dynamic follow camera with slight shoulder offset.
  - **Combat Lock-On/Strafe:** Optional lock-on for targeted melee/ranged attacks.
  - **Auto-Recenter:** Camera recenters after movement lulls or sharp turns.
  - **Environmental Cues:** Subtle shakes, zooms, and pans emphasize key moments.
  - **Obstacle Avoidance:** Camera dynamically adjusts to avoid walls or tight spaces.

    - **Manual Adjustment:** Player can always rotate or zoom the camera (within limits).

Stress & Growth:

- **Stresses the Player:**
    - Requires awareness of space, positioning, and camera angles mid-combat.
    - Demands quick reactions for dodging, aiming, and platforming.
    - Encourages mastering subtle input timing.
- **Player Relieves Stress By:**
    - Learning optimal control mappings and muscle memory.
    - Customizing camera behavior to personal comfort.
    - Upgrading mobility options or unlocking smoother movement perks.
- **How the Player Grows:**
    - Gains confidence in navigating diverse terrain and arenas.
    - Learns to anticipate and manipulate camera angles for advantage.
    - Becomes fluent in chaining movement and attacks efficiently.
- **Post-Success Progression:**
    - Unlock new movement abilities (e.g., dodge).
    - Upgrade dodge mechanics (e.g., blink dash, charged evade).
    - Camera behavior adjusts further to reflect C.L.O.E.'s upgrades and combat style.

Design Risks:

- **Camera Disorientation:** Poor handling in tight or vertical spaces could frustrate players.
- **Animation Responsiveness:** Delayed inputs or sluggish transitions may break immersion.
- **Overloaded Controls:** Need to ensure complexity doesn't outweigh fluidity.
- **Player Discomfort:** Poor FOV or camera shake settings could affect comfort or accessibility.

References:

Ratchet & Clank: Going Commando - Core inspiration.

Returnal - 3rd person bullet hell with responsive movement/camera.

Ratchet & Clank: Rift Apart - Dodge; Snappy movement, engaging avatar control with platform-combat blend.

User Story:

As a player, I want to control C.L.O.E. so that I can progress through both combat and exploration confidently and intuitively.

Completion Criteria:

- Implement C.L.O.E.'s movement set: run, jump, dodge.
- Set up input system.
- Build third-person camera system with manual control and auto-adjustments.
- Design lock-on and free-look camera modes with context switching.
- Implement camera smoothing and obstacle avoidance logic.

- Ensure avatar movement integrates with combat and platforming sections fluidly.

Gun Base Class

Priority: Highest

Objective:

Provide a flexible, extendable gun system for ranged combat by establishing a base class with modular functionality, enabling the creation of unique weapon variants without rewriting core logic.

Summary:

The Gun feature is built on a base class that encapsulates the fundamental behavior of all projectile weapons in the game. This includes projectile spawning, range detection, and lock-on targeting. Each gun variant inherits from this base, allowing designers to define new weapons with custom models, ranges, effects, and firing logic while maintaining consistent system structure. The gun system supports both single-target and multi-target lock-on mechanics and integrates with the player’s inputs for responsive ranged gameplay. This modular setup encourages experimentation and future content scalability.

Core Mechanics:

- **Gun Base Class Functionality:**
  - **Projectile Spawning:** Fires projectiles using a pre-defined socket or spawn location on the gun.
  - **Range Detection:** Checks for enemies within a customizable range and stores them in an array.
  - **Target Lock-On Array:** Dynamically fills a list of valid targets within range for lock-on systems.
  - **Activation Input:** On player input (e.g., right trigger or mouse click), fires at a selected or auto-targeted enemy.
  - **Cooldown/Fire Rate:** Configurable delay between shots to balance gameplay.
  - **Ammo Management:** Placeholder for ammo consumption or energy cost integration.
- **Modular Variation (Child Classes):**
  - **Custom Models:** Each variant may have unique geometry and visual effects.
  - **Custom Firing Logic:** Overridable functions for spread shot, charged shot, homing projectiles, etc.
  - **Custom Ranges:** Each weapon defines its own detection radius.

Stress & Growth:

- **Stresses the Player:**
  - Requires quick reflexes and spatial awareness to lock onto multiple enemies in range.
  - Challenges the player to choose the right weapon variant for the right situation.
  - Demands timing and ammo/resource management under pressure.
- **Player Relieves Stress By:**
  - Learning which weapon variants excel in which combat scenarios.
  - Mastering the rhythm and pacing of each gun’s firing and cooldown cycles.
  - Unlocking and experimenting with new weapon types to find their preferred playstyle.
- **How the Player Grows:**
  - Gains proficiency in ranged combat tactics, including timing, positioning, and target prioritization.
  - Learns to evaluate enemy layout and environmental spacing to optimize weapon choice.
  - Develops a personal loadout strategy based on gun type and expected encounter type.

- **Post-Success Progression:**
  - Unlock new gun variants with expanded abilities (e.g., piercing rounds, AoE shots, ricochet rounds).
  - Upgrade existing guns to improve stats (e.g., faster fire rate, extended range, larger lock-on capacity).
  - Access optional content gated behind mastery of specific ranged challenges.

Design Risks:

- **System Bloat:** Too many weapon variants could dilute core gameplay feel if not curated properly.
- **Lock-On Confusion:** Without clear feedback, players may not know which enemies are being targeted.
- **Overpowered Ranged Play:** Guns must be balanced to not overshadow melee or stealth options.
- **Performance Impact:** Frequent projectile spawning or overlap checks may strain lower-end systems if unoptimized.

References:

Ratchet & Clank Series – Modular weapon design, variety in function and visual identity.

Returnal – Procedural weapon mods and projectiles with smooth auto-targeting systems.

User Story:

As a player, I want to use a variety of guns, so that I can engage enemies in dynamic, satisfying ways tailored to my preferred playstyle.

Completion Criteria:

- Implement base gun class with modular projectile logic and range detection.
- Enable spawning and management of projectiles from gun socket.
- Implement target detection and lock-on list generation within a definable radius.
- Integrate player input for weapon activation and cooldown handling.
- Develop UI feedback for lock-on, weapon ready/cooldown, and targeting.
- Create sample gun variants to validate modular system flexibility.

Enemy Units

Priority: Highest

Objective:

Create modular enemy units that mirror C.L.O.E.'s systems while expanding with AI-specific behavior to challenge the player, support world-building, all while maintaining an easily expandable and modular framework.

Summary:

Enemy characters are duplicated from C.L.O.E.'s base avatar and gun class, providing a consistent and readable visual and functional baseline. These enemies utilize the same weapon and movement systems but are enhanced with AI layers that govern behavior, targeting, and environmental responsiveness. Enemies can engage the player and optionally other NPCs, creating opportunities for factional conflict and emergent gameplay. Additional player-facing information (PFI) ensures that enemies feel alive within the game world, engaging with their surroundings, reacting to changes, and supporting the emotional and narrative tone of each encounter.

Core Mechanics:

- **Shared Systems with C.L.O.E.:**
  - **Avatar Base:** Uses the same skeletal mesh, animation blueprint, and movement capabilities as C.L.O.E.
  - **Gun System:** Enemies utilize a subclass of the modular gun base, allowing for consistent ranged combat.
- **AI Behavior:**
  - **Detection System:** Perception-based logic (sight, sound) to detect the player and other hostile NPCs.
  - **Target Selection:** Optional system to prioritize threats.
  - **Movement AI:** Navigation mesh-based pathfinding for patrols, pursuit, flanking, or retreat.
  - **Combat Logic:** Fires projectiles using the same logic as C.L.O.E., with built-in cooldown and lock-on selection.
  - **Group Coordination:** Squad-based behavior like alert sharing, coordinated attacks, or retreating together.

Stress & Growth:

- **Stresses the Player:**
  - Creates consistent pressure through AI positioning and coordinated aggression.
  - Encourages dynamic movement and cover usage by forcing reaction to ranged and flanking tactics.
  - Introduces unpredictable behavior when enemies interact with other NPC factions.
- **Player Relieves Stress By:**
  - Learning enemy patterns and AI cues to anticipate behavior.
  - Mastering ranged tactics to pick off threats before group alerts.
  - Using the environment or Finch to divide and distract groups.
- **How the Player Grows:**
  - Learns how to exploit gaps in patrols.
  - Recognizes visual/audio cues to identify enemy threat level and behavior state.
  - Adapts to multi-faction scenarios where enemies may not be focused solely on the player.
- **Post-Success Progression:**
  - Introduce elite variants with upgraded gear, faster reaction times, or squad command logic.
  - Unlock enemy intel codexes or logs with behavioral weaknesses and lore insights.
  - Design dynamic encounters with multiple AI factions reacting to the player and the environment differently.

Design Risks:

- **AI Complexity:** Poorly tuned AI may feel either too passive or too omniscient.
- **Combat Clarity:** If enemy behavior too closely mirrors C.L.O.E. without distinction, players may notice.
- **Performance Cost:** Simultaneous perception, movement, and logic for many enemies can strain performance.
- **Over-telegraphing:** Excessive audio/visual feedback could ruin tension or make enemies feel robotic.

References:

- **Metal Gear Solid V** – Smart enemy AI that escalates based on player patterns.
- **The Last of Us** – Enemy dialog and world interactions enhance immersion.
- **Ratchet & Clank: Rift Apart** – Diverse enemy types with readable silhouettes, responsive audio/visual PFI, and clear behavior states.

User Story:

As a player, I want enemies to behave like intelligent counterparts to C.L.O.E., so that combat feels fair, immersive, and dynamically reactive to my choices.

Completion Criteria:

- Duplicate C.L.O.E.'s avatar and gun system for baseline enemy behavior.
- Implement AI logic for detection, pathfinding, targeting, and gun usage.
- Enable optional multi-faction targeting system for dynamic enemy encounters.
- Create idle and reaction animations for immersive polish.
- Design death and hit feedback systems for clarity and impact.
- Build patrol and combat test scenarios to validate AI behavior under stress.
- Integrate voice lines or audio cues for increased player feedback and world immersion.

Enemy Spawners

Priority: High

Objective:

Enable level designers to spawn enemies dynamically during gameplay, allowing for reactive encounters and flexible level pacing.

Summary:

Enemy spawners are simple blueprint actors that spawn a selected enemy type at the tip of an arrow. They're triggered by any actor that implements the "Activate" blueprint interface, making them plug-and-play for a variety of scripted scenarios like ambushes, area lockdowns, or timed waves. Designed for modularity and ease of use in the level editor.

Core Mechanics:

- Largely a dev-side feature with:
  - Arrow component for spawn direction.
  - Variable to assign enemy class.
  - "Activate" interface support.
- Spawns enemy when triggered, facing arrow direction.
- Optional delay, repeat toggle, or FX cues for player feedback.

Stress & Growth:

- **Stresses the Player:**
  - Enables surprise spawns and pressure pacing.
- **Player Relieves Stress By:**
  - Recognizing environmental cues and adapting mid-combat.
- **Post-Success Progression:**
  - Used in challenge rooms, elite waves, or puzzle-based combat setups.

Design Risks:

- Overuse or unclear PFI may frustrate players.
- Visible spawner placement can lead to exploitation.

References:

- **Ratchet & Clank: Rift Apart** – Clear and exciting enemy spawn telegraphs.
- **DOOM Eternal** – Spawn-driven combat waves with readable pacing.

User Story:

As a level designer, I want to use enemy spawners so that I can control when and where enemies appear.

Completion Criteria:

- Blueprint spawner with enemy class variable and arrow for orientation.
- Triggers on “Activate” blueprint interface from any actor.
- Supports optional delay, repeat, and (optionally) spawn FX.

HUD: Core Player UI

Priority: High

Objective:

Present essential player information in a clean, intuitive layout that aligns with character perspective and emphasizes situational awareness without visual clutter.

Summary:

The HUD in *C.L.O.E. & Finch* is minimal and diegetically aligned. Health and ammo are displayed in the bottom-left—mirroring C.L.O.E.’s on-screen position—while collected currency appears in the top-right. Enemy health bars are hidden by default and only appear when the player locks onto a target, keeping the screen clear until combat focus is necessary.

Core Elements:

- **Health & Ammo:**
  - Positioned bottom-left for spatial cohesion with C.L.O.E.’s screen orientation.
  - Ammo updates dynamically based on current weapon.
- **Currency:**
  - Displayed top-right for quick glance during downtime or post-combat.
- **Enemy Healthbars:**
  - Hidden by default.
  - Appear and track the targeted enemy only while lock-on is active.
  - Attached directly to the enemy as a world-space widget.

Stress & Growth:

- **Stresses the Player:**
  - Minimal HUD increases reliance on spatial awareness and PFI.
- **Player Relieves Stress By:**

    - Locking on to reveal critical enemy info only when needed.
- **Post-Success Progression:**
    - Future HUD upgrades may include:
        - Alternate ammo types.
        - Finch-specific UI when controlled.
        - Status effects or ability cooldowns.

Design Risks:

- Over-minimalism could lead to player confusion in chaotic encounters.
- World-space widgets must be legible from all angles.

References:

- **Ratchet & Clank: Rift Apart** – Clean HUD layout, contextual info pop-ups.
- **Dead Space** – Spatially motivated HUD placement and minimal clutter.
- **Horizon Zero Dawn** – Lock-on healthbar reveals and diegetic clarity.

User Story:

As a player, I want a simple and readable HUD so that I can stay informed without losing immersion.

Completion Criteria:

- Display player health and ammo in bottom-left.
- Display currency in top-right.
- Show enemy healthbars only when locked on.
- Ensure legibility of world-space health widgets.
- Align HUD elements to screen layout and player orientation.

Currency

Priority: Medium

Objective:

Provide players with a satisfying, “physical” collectible that reinforces progress and incentivizes exploration and combat.

Summary:

Currency is a small, rolling pickup that adds to the player’s total when collected. It consists of two blueprints: the currency item itself and a spawner. The item uses physics and a subtle idle rotation for visual appeal, while the spawner handles randomized launch and placement. The result is a dynamic, tactile reward that reacts to the world and feels good to collect.

Core Elements:

- **Currency Blueprint:**

    - Mesh + rotating movement component.
    - Sphere collision with physics enabled on spawn.
    - After a short delay: disables physics, destroys on overlap, plays sound, adds 1 to currency count.
- **Currency Spawner:**
    - Spawns currency with random Z rotation.
    - Adds upward + forward impulse for natural scatter.

Stress & Growth:

- **Player Relieves Stress By:**
    - Hearing satisfying pickup sounds and watching currency react physically.

Design Risks:

- Physics bugs may cause clipping or erratic motion.
- Too much scatter can delay pickups in combat scenarios.

References:

- **Ratchet & Clank Series** – Visually and physically satisfying currency collection.

User Story:

As a player, I want to collect currency so that I can earn rewards and feel progression.

Completion Criteria:

- Implement currency item with mesh, rotation, and pickup logic.
- Implement spawner with randomized force and rotation.
- Ensure audio feedback and counter increment on pickup.

Pickups

Priority: Medium

Objective:

Deliver instant, tangible benefits to the player through collectible world objects that reinforce survival and combat readiness.

Summary:

Pickups are simple world items based on a modular class. Each pickup features a visible mesh, idle rotation, and collision box that triggers its effect upon player contact. Designed for flexibility, the base class supports easily customized children like health and ammo, allowing designers to quickly populate levels with helpful rewards.

Core Elements:

- **Base Pickup Class:**
  - Static mesh (visual).
  - Rotating movement component (visual cue).
  - Box collision triggers event on overlap.
  - On trigger: calls child-defined effect, then destroys self.
- **Variants (Children):**
  - **Health Pickup:** Restores a set amount of player health.
  - **Ammo Pickup:** Refills weapon-specific ammo.

Stress & Growth:

Player Relieves Stress By:

- Gaining critical resources during or after intense encounters.

Design Risks:

- Poorly balanced placement can trivialize challenge or starve the player.

References:

- **Ratchet & Clank: Rift Apart** – Clear, satisfying collectible interactions.
- **Halo** – Simple pickup behavior with meaningful effects.

User Story:

As a player, I want to collect pickups so that I can restore resources and stay combat-ready.

Completion Criteria:

- Create base pickup blueprint with:
  - Mesh, rotation, collision trigger, self-destruction logic.
- Implement child classes for:
  - Health restoration.
  - Ammo refill.
- Ensure effects apply correctly and are responsive on collection.

Melee Attack: High-Risk, High-Impact

Priority: Medium

Objective:

Give the player a powerful melee option that emphasizes commitment, timing, and precision, offering high damage in exchange for momentary vulnerability.

Summary:

C.L.O.E.'s melee is a targeted, high-impact move triggered while locked onto an enemy. When activated, it commits the player to a short animation window where movement is disabled and the player is exposed. The system is tightly integrated with the existing lock-on and target selection logic to ensure seamless execution and clear player intention. Designed as a clutch tool, it rewards aggression but punishes recklessness.

Core Mechanics:

- **Activation:**
  - Requires lock-on to a valid enemy target.
  - Activated via a specific input (e.g., Square or F).
- **Execution:**
  - Locks the player into a brief melee animation (no movement control).
  - During this window, C.L.O.E. is vulnerable to damage.
  - If successful: deals heavy damage or staggers the enemy.
- **Integration:**
  - Uses the same target array from the gun system.
  - Prioritizes closest or most dangerous target within lock-on range.

Stress & Growth:

- **Stresses the Player:**
  - Demands timing and spatial awareness.
  - Punishes careless usage with vulnerability during execution.
- **Player Relieves Stress By:**
  - Using melee as a finisher or interrupt.
  - Learning to recognize safe windows for aggressive plays.
- **Post-Success Progression:**
  - Potential upgrades could reduce vulnerability window or add knockback effects.

Design Risks:

- Could feel frustrating if enemy interrupts are too frequent.
- Needs clear PFI (Player-Facing-Information) to telegraph activation and vulnerability period.

References:

- **Ratchet & Clank: Rift Apart** – Close-range finishers with exaggerated feedback.
- **God of War (2018)** – Risk/reward melee sequences with impactful animations.

User Story:

As a player, I want to perform a powerful melee attack on my target so that I can deal high damage at the cost of brief vulnerability.

Completion Criteria:

- Integrate melee input with existing lock-on system.

- Trigger heavy damage animation that locks player movement.
- Disable player control and apply vulnerability during attack.
- Apply damage to locked-on enemy if within valid range.
- Implement PFI cues (e.g., sound, slow-mo, screen shake) to sell impact and risk.

Playable Companion: Finch

Priority: Medium

Objective:

Allow the player to control Finch in specific gameplay segments to encourage tactical multitasking and strategic planning under pressure.

Summary:

Finch is a playable companion who excels at traversal, sabotage, and support. The player can switch between C.L.O.E. and Finch during specific combat or puzzle encounters to overcome multi-layered challenges that neither character could solve alone. While C.L.O.E. is built for direct confrontation, Finch's gameplay is about **stealth, speed,** and **clever use of the environment**. Mastering Finch means understanding enemy patterns, prioritizing threats, and coordinating with C.L.O.E. for seamless synergy. This duality challenges the player to think on two levels and rewards strategic flexibility.

Design Breakdown:

Core Mechanics:

- **Character Swap Mechanic:**
  - Switch between C.L.O.E. and Finch with a button press (e.g., Tab on keyboard or L1 on controller).
  - Swapping is limited in combat unless Finch-specific triggers are present.

Finch's Abilities:
Combat

- **Sabotage:** Disable enemy shields, turrets, and alarms.
- **Distraction:** Throw objects or emit noises to lure enemies away.
- **Hide:** Hide in tall grass or under props to avoid prying eyes.

Puzzle/Exploration

- **Traversal:** Crawl through vents, wall-run, or climb surfaces C.L.O.E. can't access.
- **Hack/Puzzle Mini-Games:** Solve tech puzzles to open doors or disable security systems.

Teamwork Interactions (examples):

- Finch disables a generator while C.L.O.E. holds off waves of enemies.
- Finch distracts a boss while C.L.O.E. lands heavy damage.

Stress & Growth:

- **Stresses the Player:**
  - Encourages mental juggling between direct action (C.L.O.E) and indirect problem-solving (Finch).
  - Forces the player to learn how to optimize Finch's pathing and impact in real-time scenarios.
  - Requires keen awareness of level layout, enemy placement, and timing.
- **Player Relieves Stress By:**

    - Learning effective Finch routes.
    - Perfecting timing and rhythm in switching between characters.
    - Upgrading Finch’s tools for greater impact.
- **How the Player Grows:**
    - Becomes more fluid in switching roles mid-encounter.
    - Develops foresight in setting up future advantages with Finch before combat escalates.
    - Learns to handle asymmetrical gameplay systems in harmony.
- **Post-Success Progression:**
    - New upgrades unlocked for Finch’s toolkit (e.g., invisibility cloak, EMP bomb).
    - Optional side content that requires high mastery of Finch’s mechanics (secret areas, challenge arenas).

Design Risks:

- **Cognitive Load:** Too much micromanagement could overwhelm some players.
- **Control Complexity:** Input scheme must remain simple despite two-character control.
- **Character Balance:** Finch must feel important but not overpowered compared to C.L.O.E.
- **AI Support When Not Controlled:** If C.L.O.E./Finch is AI-controlled when not played, their behavior must be consistent and helpful without trivializing challenges.

References:

The Last of Us Part II – Companion stealth and AI support behavior.

It Takes Two/**Split Fiction** – Dual-character puzzle-solving and environment interaction.

A Plague Tale: Innocence – Use of a non-combat companion for distraction and puzzle-solving.

User Story:

As a player, I want to switch to Finch, so that I can disable obstacles and outmaneuver enemies to support C.L.O.E. in and out of combat.

Completion Criteria:

1. Implement Finch as a playable character with distinct yet familiar controls.
2. Allow player to swap between C.L.O.E. and Finch during gameplay.
3. Design sabotage, distraction, and traversal abilities unique to Finch.
4. Build level segments requiring Finch’s abilities to progress.
5. Create/Modify existing enemy systems that respond to Finch's presence differently than C.L.O.E.
6. Create puzzles or cramped areas only accessible by Finch.
7. Implement UI feedback to indicate when Finch is needed or useful..
8. Develop AI fallback behavior for C.L.O.E./Finch when not player-controlled.