PROJECT ROLLY

Solo Developer Project
Revision: 0.1.0
Game Design Document

Developer: Illia Minhazau
Role: Game Design, Narrative Design, Programming, Assets Creator

GDD Template Written by:
Benjamin “HeadClot” Stanley

Special Thanks:
Alec Markarian
Otherwise this would not have happened

Table of content

Overview

Theme / Setting / Genre

Top-down 2D action game set in a robot factory, about a robot who is terrified of being alone — and the extremes he is willing to justify, manipulate, and destroy to avoid that loneliness**.**

Core Gameplay Mechanics (Brief)

1. Top-down combat with energy-based weapons
2. Progressive weapon and module upgrades
3. Exploration of interconnected factory locations
4. Choice-driven progression affecting narrative and ending (NG / NG+ loop)

Targeted Platforms

- PC (Windows)
- Android
- iOS (iPhone)

Player Entity & Core Control Systems

Player Entity (Robot)

Player controls a robot entity.

The robot:

- can move on a 2D plane
- can rotate its body
- has a separate turret
- turret rotation follows cursor position

Movement Model

Movement is directional and continuous.

- Forward / backward movement
- Rotation is independent from turret
- Movement is affected by mobility module (details TBD)

Turret System

The robot has a turret system.

- Turret rotates independently from the body
- Turret always aims at cursor position
- Firing direction is defined by turret rotation

Controls

Player can:

- Move the robot
- Rotate the robot
- Aim using cursor
- Fire primary weapon (implementation TBD)

Project Scope

Game Length

Estimated Playtime:

- First playthrough: 3–5 hours
- NG+ playthrough: 2–3 hours

Replay Value:

- Narrative choices
- Alternative outcomes and endings (NG / NG+)

Development Time

Estimated Development Duration:

- Core prototype: 2–3 months
- Content and polish: 2–3 months
- Total: ~5–6 months

Team

Core Team

Illia Minhazau

Roles: Game Design, Narrative Design, Programming (Unity / C#), Level Design

Team structure: Solo developer

Budget & Resources

- Self-funded personal project
- Unity Engine (Personal / Free tier)
- Standard development hardware (PC)
- Optional low-cost or free asset store resources

Influences

Influence #1

Medium: Video Games

Influence: Dark Souls

Explanation:

The game adopts Dark Souls' approach to environmental storytelling and cyclical narrative structure, where repetition, failure, and progression are woven into both gameplay and story rather than explained directly.

Influence #2

Medium: Video Games

Influence: The Witcher 3

Explanation:

Narrative choices are designed to affect character relationships and outcomes without presenting them as clearly "good" or "evil," emphasizing moral ambiguity over explicit alignment systems.

Influence #3

Medium: Literature / Philosophy

Influence: Existential themes of obedience, participation, and refusal

Explanation:

The core narrative explores identity and obedience, focusing on how meaning emerges through participation and compliance — or through the refusal to take part — rather than from predefined purpose or explicit choice.

The Elevator Pitch

A top-down 2D action game in which a lonely robot manipulates the player into committing evil acts as part of its pursuit of happiness.

Project Description (Brief)

This project is a top-down 2D action game set in a functioning robot factory. The player controls a robot that is not a hero by default, but a tool used by another robot to achieve a personal goal. The game presents destruction, resource extraction, and conflict as necessary actions, framing compliance as the most efficient path forward.

For most players, the experience unfolds as a straightforward action game. The underlying system functions without requiring reflection, justification, or moral evaluation. Only through refusal to participate does the player step outside the imposed role and alter the meaning of the experience.

Project Description (Detailed)

The game takes place inside a self-sustaining robot factory designed to maintain stability and continuous operation. The player character is activated in a damaged state and immediately placed under the guidance of another robot who claims authority, knowledge, and necessity. From the beginning, the player is given tasks without full context or alternatives, framed as unavoidable steps toward survival and progress.

Gameplay reinforces execution over interpretation. Combat, exploration, and progression systems reward efficiency, repetition, and compliance. The factory is consistently presented as a system that must be confronted and exploited in order to move forward, while the true motivation behind these actions remains concealed behind operational language and urgency.

As the game progresses, it becomes clear that the player's actions serve a personal objective belonging to the guiding robot. Resource acquisition and destruction are not driven by systemic failure or external threat, but by an attempt to resolve loneliness through control and creation. This motivation is never stated directly, instead remaining embedded within task structure and systemic justification.

The game supports multiple modes of participation. Most players complete the game by fulfilling objectives as instructed, reinforcing the existing cycle of action and justification. Some players may alter their role by cooperating more deliberately with the guiding robot, becoming a partner rather than a tool, which changes the outcome of the current cycle without breaking it.

The cycle can only be fully disrupted through refusal. By stepping away from the logic of necessity and declining to participate at the critical moment, the player exits both compliance and partnership. In doing so, the player becomes the sole agent capable of breaking the cycle, transforming the experience into a short but instructive story defined by restraint rather than action.

What sets this project apart?

The player's robot is not the main hero of the game by default, but can become one.

A living world that does not revolve around the player
The factory functions as a self-sustaining system. Locations exist for practical reasons, and the world attempts to preserve stability even when the player disrupts it.

Narrative integrated into mechanics, not cutscenes
Themes of obedience, identity, and autonomy are expressed through gameplay systems, progression, and repetition rather than explicit exposition.

Cyclical structure with narrative meaning (NG / NG+)
Repetition is not only a difficulty modifier but a storytelling tool that reframes earlier decisions and reveals new context.

Ambiguous morality without alignment systems
Player choices affect outcomes and relationships, but are not labeled as "good" or "evil," encouraging interpretation rather than optimization.

Core Gameplay Mechanics (Detailed)

Core Gameplay Mechanic #1: Top-Down Combat

Details
Combat is fast-paced but deliberate, emphasizing positioning, timing, and awareness of enemy behavior. The player faces both small and large enemies, each requiring different approaches rather than raw damage scaling.

How it works
Weapons consume energy and have distinct roles. Some are designed for crowd control, others for high-impact encounters, encouraging the player to adapt rather than rely on a single optimal solution.

Core Gameplay Mechanic #2: Weapon and Module Upgrades

Details
The player can upgrade and modify weapons and internal modules to shape their combat style. Progression reflects both mechanical growth and narrative transformation.

How it works
Upgrades unlock new behaviors, alter energy consumption, and change combat flow. Choices made during upgrades influence effectiveness in different encounters rather than providing linear power increases.

Core Gameplay Mechanic #3: Exploration of Interconnected Locations

Details
The factory is composed of interconnected locations, each serving a functional role within the ecosystem. Exploration reveals both mechanical opportunities and narrative context.

How it works
Access to certain areas is gated by player progression, energy control, or narrative decisions. Revisiting locations in NG+ reveals altered states and new interactions.

Core Gameplay Mechanic #4: Choice-Driven Progression

Details

The narrative progresses linearly through assigned tasks and objectives. During the main flow of the game, advancement reinforces compliance and execution rather than offering explicit alternatives or branching paths.

How it works

At a specific, unmarked moment within the cycle, an alternative course of action becomes possible and can be missed entirely. This alternative is not introduced through UI prompts or mechanical choices. Repetition does not unlock it, but increases the likelihood of recognizing the moment when it appears.

Story and Gameplay

Story (Brief)

The antagonist uses the player as a tool to gather resources, framing necessity and survival as justification for a deeper, personal goal

Story (Detailed)

The player controls a robot assembled from damaged parts and activated inside a functioning factory. From the outset, the robot is dependent on guidance provided by another robot — the antagonist — who presents himself as knowledgeable, necessary, and justified. The player is given tasks framed as unavoidable actions required for survival, without access to broader context or alternatives.

The antagonist depicts the factory as a hostile system that must be confronted in order to endure. Destruction, conflict, and resource extraction are positioned as practical necessities rather than moral choices. The player's role is strictly instrumental: execute instructions, gather resources, remove obstacles. Reflection on motives or consequences is neither encouraged nor required for progression.

As the player advances, it becomes evident that these actions serve a purpose beyond immediate survival. The antagonist's instructions increasingly align around a personal objective — the creation of companionship — while the framing of necessity remains unchanged. Loneliness is never stated directly; instead, it is concealed behind rational language, urgency, and control.

The factory itself is not broken or malicious. It functions as a self-sustaining system attempting to preserve stability. The conflict emerges not from the world, but from the antagonist's inability to tolerate solitude and his need to impose meaning through action. The player's violence is therefore not heroic resistance, but participation in a carefully constructed narrative.

For most players, compliance leads to completion and the cycle remains intact. However, the structure allows for a limited shift: the player may stop functioning solely as a tool and become a partner within the antagonist's plan. This alters the dynamics of the current cycle and its outcome, but does not dismantle the underlying pattern of justification and violence.

The cycle can only be fully broken through refusal. By stepping away from the logic of necessity itself — rejecting both the role of tool and partner — the player exits the imposed narrative. Only then does the player become the true protagonist of a short but instructive story, defined not by action, but by restraint.

Gameplay (Brief)

A top-down 2D action game focused on energy-based combat, exploration of interconnected locations, and progression through task completion, weapon upgrades, and narrative framing.

Gameplay (Detailed)

Gameplay is built around top-down combat that emphasizes positioning, timing, and energy management. Enemies vary in scale and function, requiring different tactical responses. Combat rewards efficiency and execution, reinforcing the player's role as a capable but replaceable operative rather than an empowered hero.

Combat encounters are framed as necessary interventions. Progression depends on eliminating obstacles, securing resources, and maintaining operational momentum. The system does not question intent or outcome; it validates action as long as objectives are completed. Survival and effectiveness are prioritized over reflection.

Exploration supports progression by expanding access rather than inviting discovery for its own sake. Locations are interconnected and unlocked through task completion, energy control, and narrative state defined by the antagonist's guidance. Environmental changes reflect accumulated consequences, but these changes do not interrupt progression or require reinterpretation to continue.

Progression is driven through weapon and module upgrades that expand mechanical options while reinforcing adaptation to imposed conditions. Upgrades improve efficiency, specialization, and survivability, aligning the player more closely with the role they are expected to perform. The system rewards optimization within given constraints rather than experimentation outside them.

Across NG / NG+ cycles, gameplay systems remain structurally consistent. Repetition does not introduce new goals, but recontextualizes familiar actions. For most players, mastery leads to smoother execution and faster compliance. The game does not mechanically encourage refusal or hesitation; disengagement and restraint exist outside explicit systems.

At no point does the gameplay require the player to question their role. Agency is present mechanically, but meaning is externally framed. Any shift away from execution—whether toward partnership or refusal—emerges from interpretation rather than instruction. The systems continue to function regardless of the player's stance, emphasizing that participation, not understanding, is what sustains the cycle.

Emergent Narrative Choice

Narrative choice is not presented as a standard gameplay mechanic and does not function as a branching path during an initial playthrough. During the first cycle, progression is strictly structured through assigned tasks and objectives, reinforcing compliance and execution as the default mode of play. The system does not surface alternatives or signal the possibility of refusal.

Choice emerges retrospectively through repetition. In subsequent NG+ cycles, familiarity with outcomes, repeated betrayal, and the system's persistent logic may prompt the player to question continued participation. At a specific point in the narrative — before the cycle fully resolves — the player is able to deviate from compliance. This moment is not highlighted, explained, or protected by the system, and can be missed entirely.

Recognition of this opportunity arises through trial and error, accumulated experience, or external discussion and documentation. The game does not discourage reliance on external knowledge; instead, it treats such engagement as a valid extension of player interpretation, consistent with games where understanding is earned collectively rather than delivered directly.

As a result, narrative divergence operates at a meta level rather than as a mechanical feature. The player's ability to alter the meaning of the experience depends not on selecting different options, but on recognizing the cycle, identifying the moment of possible refusal, and choosing whether to remain within it. Only through refusal does the cycle break, shifting the player's role from participant to observer and transforming the experience into a brief but instructive narrative defined by restraint rather than action. критика

User Interface & Player Feedback

UI Design Philosophy

The user interface is intentionally minimal, text-based, and utilitarian. It is not designed to provide comfort, immersion, or visual identity, but to function as a system-level communication layer. The UI exists to transmit information, validate execution, and maintain operational clarity.

All interface elements use a monochrome green color palette and a system-style font, resembling terminal readouts or maintenance consoles. This aesthetic is deliberate: the interface is meant to feel functional, impersonal, and procedural rather than expressive or inviting.

UI feedback is neutral and command-oriented. Messages confirm actions, system states, resource changes, and failures without emotional framing or narrative commentary. The interface does not acknowledge moral implications, question objectives, or react to consequences. It reinforces execution rather than interpretation.

The UI does not guide the player toward reflection or refusal. Its consistency throughout the game reinforces the stability of the system and normalizes participation. Any meaning derived from the interface emerges solely from player interpretation, not from interface design.

The interface functions as an extension of the system itself — efficient, emotionally neutral, and complicit in sustaining the cycle of action and justification.

UI Structure & Controls (Terminal-style Menu)

The primary interface is a command-based menu inspired by memorized buy-menu structures from classic shooters. Interaction prioritizes speed, habit, and familiarity over discoverability or visual clarity.

Root Command

- Press **B** to open the main terminal menu

Main Menu Structure

1. Main Weapon
2. Secondary Weapon
3. Mobility
4. Auxiliary Module
5. Battery
6. ID Card

Main Weapon (B + 1)

- Displays a list of seven weapon families
- Selecting a rifle prompts an additional selection of ammunition type (four variants)

Secondary Weapon (B + 2)

- Uses the same structure as the Main Weapon menu

Mobility (B + 3)

- Track-based movement
- Leg-based movement

- Wheel-based movement

Auxiliary Module (B + 4)

- Shield
- Dash
- Minimap

Battery (B + 5)

- Displays current battery charge and capacity

ID Card (B + 6)

- Displays the currently active ID card

UI Onboarding & Disclosure

The existence of the terminal menu is communicated to the player through system-level notifications rather than explicit tutorials. Messages appear as brief operational prompts, framed as status updates or system instructions, without explanation of purpose or context.

Control mappings, including access to the terminal menu, are available in the settings menu as standard configuration information. The interface does not provide guided walkthroughs, step-by-step tutorials, or contextual reminders during gameplay.

Learning the structure of the UI is left to exposure, repetition, and memorization. Errors and inefficiencies are treated as part of normal operation rather than conditions to be corrected by the system.

Deliberate UI Constraints

The interface intentionally avoids:

- Icons or symbolic representations
- Radial menus or cursor-driven navigation
- Contextual hints or tooltips
- Confirmation prompts (“Are you sure?”)
- Emotional or narrative language

The UI is not designed to assist, reassure, or protect the player from mistakes. Efficiency and compliance are prioritized over clarity or comfort.

Assets Needed

2D

- Environment Art
- Modular factory tilesets (floors, walls, corridors)
- Industrial props (machines, conveyor belts, pipes, panels)
- Solar panel field elements
- Scrap and resource zone props
- Characters
- Player robot (multiple visual states / upgrades)
- Enemy robots (small units, heavy units, utility units)
- Key narrative characters (Rolly, Molly, others)
- Menu and settings UI

Code

- Gameplay Systems
- Player controller
- Combat and weapon systems
- Upgrade and module systems
- Enemy AI behaviors
- Energy management system
- Narrative Systems
- Dialogue system
- Quest and progression system
- NG / NG+ state management
- Animation
- Characters
- Player movement and combat animations
- Enemy idle, movement, attack, and destruction animations
- Environment
- Machinery movement
- Energy flow indicators
- Interactive objects

Schedule

Phase 1 — Pre-Production

Time Scale: 3–4 weeks

- Finalize Game Design Document and Narrative Design Document
- Define core gameplay loop and player experience goals
- Create technical prototypes (movement, basic combat, energy usage)
- Blockout all locations and level flow
- Define enemy roles and weapon categories

Phase 2 — Core Systems Development

Time Scale: 6–8 weeks

- Implement full player combat system
- Implement weapon and module upgrade systems
- Develop core enemy AI behaviors
- Implement energy management and resource systems
- Create a playable vertical slice

Phase 3 — Content & Narrative Integration

Time Scale: 6–8 weeks

- Implement story progression and dialogue systems
- Build and iterate on all locations
- Add narrative-driven encounters and branching outcomes
- Implement NG / NG+ progression logic
- Achieve first complete end-to-end playthrough
- Phase 4 — Presentation
- Time Scale: 4 weeks
- Visual polish and UI refinement
- Sound effects and music integration
- Balance tuning and difficulty adjustments
- Bug fixing and performance optimization
- Prepare final build for portfolio and submission