

# Technical: Project Cartesian

## Technical Architecture

### Layer 1: The Foundation (Arch Linux)

- **Base OS:** Arch Linux.
- **Reasoning:** Rolling release provides latest CUDA/ROCM drivers and AI libraries (llama.cpp updates) without dependency hell.
- **Init System:** [systemd](#).
- **Compositor:** [Hyprland](#) (Wayland). With optional [X11](#) Fallback.
- **Audio Server:** PipeWire (Required for complex channel routing).

### Layer 2: The "Cartesian Core"

- **Framework:** Rust Native (Iced).
- **Role:** The Monolith. Replaces separate daemon scripts.
- **Modules:**
  - [lobotomy.rs](#): The Process Monitor & Signal Injector.
  - [main.rs](#): Multi-threaded Event Loop & Tauri Bridge (Mutex state management).
  - [inference.rs](#): Bindings for llama-cpp-rs (text) and moondream (vision).
  - [witness.rs](#): Manages the Ring Buffer and Vision Pipeline.
  - [hippocampus.rs](#): Manages ChromaDB vector storage and decay logic.

- **Safety Protocol:**
  - **No rm -rf:** File deletion MUST use [send2trash](#) (Freedesktop.org Trash Spec compliance).
  - **Strict Typing:** No [unwrap\(\)](#) allowed in production code.

## Layer 3: The "Face" (Rust / Iced)

- **Role:** The Visuals & Interaction Layer.
- **Stack:** Iced (Rust Native GUI)
- **Communication:** Native Rust Channels (Tokio/MPSC) & Iced Subscriptions.
- **UX Design:**
  - **Holo-Badge:** Transparent, click-through overlay.
  - **Audio-First:** Push-to-Talk (Windows Key, Fn Key, etc) interaction to minimize focus stealing during games.

## Layer 4: The "Mind" (AI Model Stack)

- **Manager Model:** Gemma 2 9B (Quantized 4-bit).
  - **Role:** Complex reasoning, macro generation.
  - **Hardware:** Requires GPU VRAM.
- **Sidekick Model:** Gemma 2 2B (Quantized 4-bit).
  - **Role:** Quick Wiki lookups, Chat.
  - **Hardware:** System RAM (CPU).
- **Vision Sidekick:** Moondream2 (1.6B).
  - **Role:** "What is this item?" analysis.
  - **Latency:** ~3-5s on CPU.

# Feature Implementation Details

## The "Witness" System (Adaptive Pipeline)

### Path A: The Replay Buffer (Hard-Coded)

- **Mechanism:** GPU Render -> NVENC/VAAPI (Hardware Encode) -> Ring Buffer (/dev/shm).
- **Constraint:** Uses dedicated Encoder Silicon, protecting Game FPS.
- **Format:** H.264/HEVC (Compressed).

### Path B: The AI Eye (Adaptive)

- **Primary (Wayland/Modern):** DMA-BUF Zero-Copy. The AI reads raw textures directly. Ultra-low latency.
- **Fallback (X11/Legacy):** XShm or Screen Copy. The AI decodes the latest frame from the Replay Buffer. Higher latency, but broad compatibility.

## The "Lobotomy" Logic (Resource Governor)

### Scenario A (God Mode): Manager (9B) on GPU.

**Condition:** No Heavy Game detected + VRAM available.

**State:** Manager (9B) loaded on GPU. High intelligence.

### Scenario B (Sidekick/Gaming Mode):

**Prerequisite:** > 6 CPU Cores AND > 16GB System RAM.

**Trigger:** Gaming Process Detected (Steam, Gamescope, Lutris).

**Action:**

- Evacuate GPU: Unload Manager Model immediately.

- Load Sidekick: Load [Gemma 2B](#) into [System RAM](#).

### The Leash:

- Apply [nice -n 19](#) (Lowest Priority).
- **E-Core Pinning:** Force affinity to [Efficiency Cores](#) (Intel 12th+ Gen) to protect Game FPS.
- **Panic Switch:** If Frame Pacing variance > 20ms, send SIGSTOP to pause AI.

**Scenario C (Potato Mode):** Hard disable AI.

**Condition:** < 6 Cores OR < 16GB RAM.

**Action:** Hard Disable.

- **Zero Idle Strategy:** AI is completely unloaded. Pressing PTT triggers a "Cold Boot" (load from NVMe -> RAM), incurs ~1.5s latency, performs inference, then auto-unloads.

## Gaming Compatibility

**Proton Integration:** The OS ships with steam and proton-ge-custom.

**Wine Prefix Management:** A Rust wrapper automatically creates and manages prefixes for non-Steam games, applying [dxvk](#) and [vkd3d](#) patches

# Development Environment (The Factory)

## Directory Structure

```
ProjectCartesian/
    └── pkg/                                # The Distribution Factory
        ├── PKGBUILD
        ├── cartesian-admin.sh
        ├── org.cartesian.policy
        └── 50-cartesian.rules
    |
    └── src/                                  # The Source Code
        └── cartesian-core/                  # The Rust Monolith
            ├── Cargo.toml
            └── src/
                ├── main.rs
                ├── lobotomy.rs
                ├── witness.rs
                ├── inference.rs
                └── assets/
                    └── fonts/                 # Custom Sci-Fi Fonts (.ttf)
    |
    └── iso/                                  # ArchISO Configuration
        └── (To be initialized...)
```