Building a Personal Operating System

A Neurodivergent Engineer's Blueprint for Taming Complexity and Unleashing Creative Flow.

The Challenge: Taming the Storm of ‘Metal Vapor-Lock’

For an engineer with auADHD, the modern digital world presents a unique conflict:

- **The ASD Brain** craves deep structure, logic, and systems.
- **The ADHD Brain** produces ‘popcorn thoughts’—rapid, non-linear ideas and tangents.

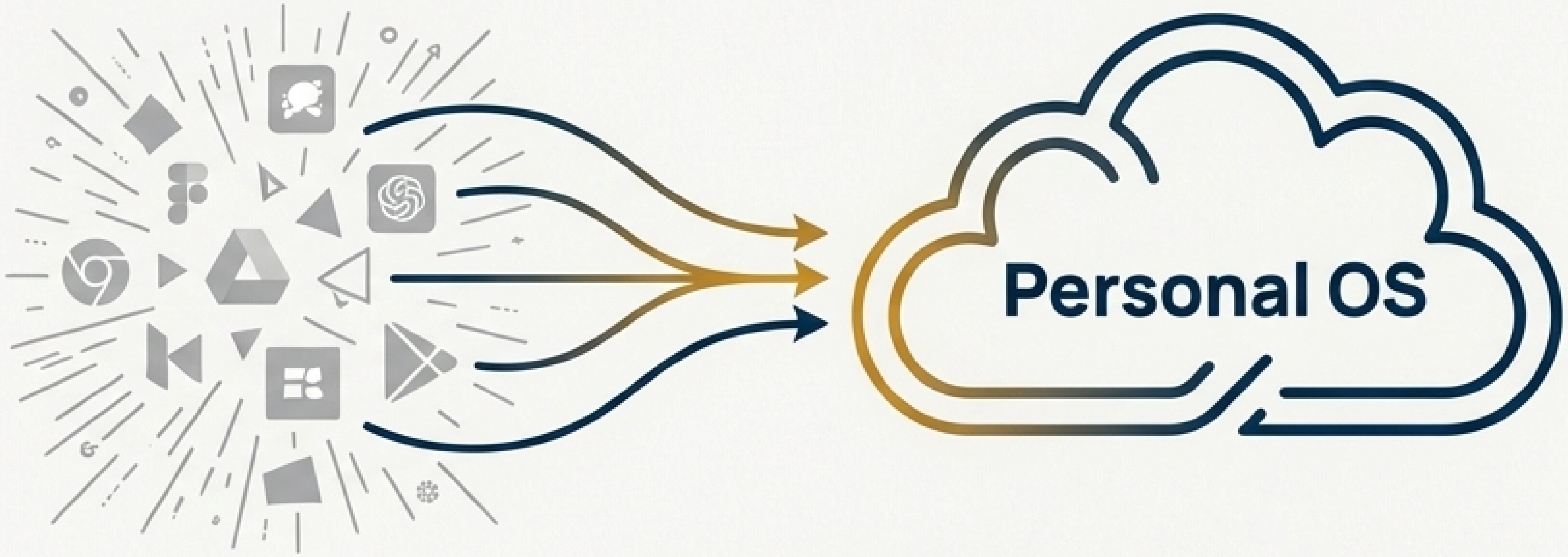
This leads to:

- **Information Overload:** Drowning in tabs, tools, and data streams.
- **The Paradox of Choice:** Paralyzing fragmentation when every tool is a potential rabbit hole.
- **Executive Function Gaps:** The ‘toil’ of moving data between systems creates friction that halts momentum.

> ‘The largest challenge is to avoid the fragmentation that causes the metal vapor-lock that options create, and to eliminate the friction of capturing ideas and data so they can be acted upon to provide value.’

The Solution: A Unified ‘Local Cloud’

Instead of fighting a collection of tools, we will build a single, cohesive machine. This is a ‘Personal Operating System’ designed like a high-end production studio, with every component assigned a specific responsibility.



Low Friction

Reduce the cognitive load of capturing ideas and managing tasks to zero.

Low Maintenance

Automate the mundane to conserve creative energy. Use off-the-shelf solutions over custom code where possible.

Low Power

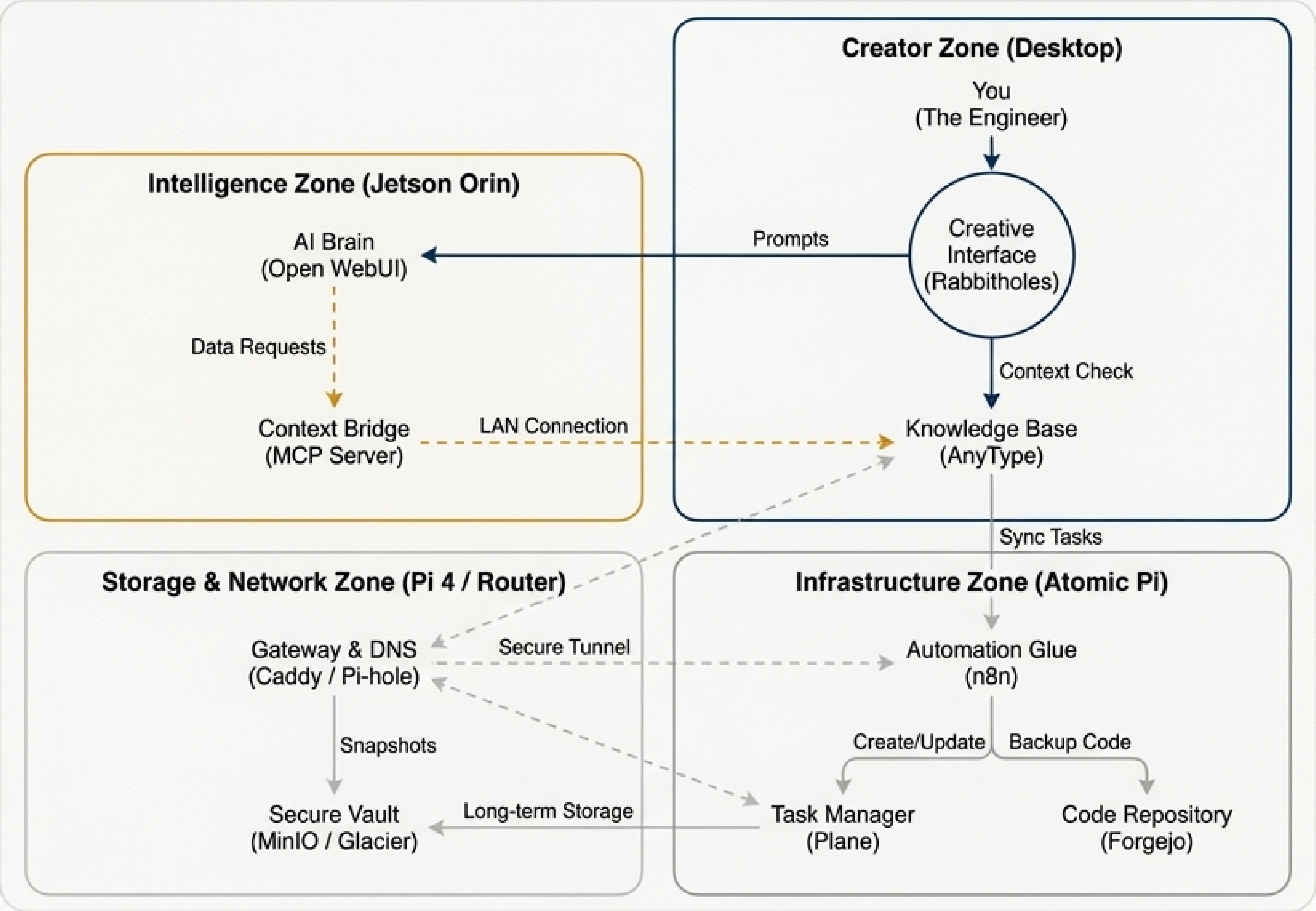
Design a distributed, energy-efficient system that respects environmental and budget constraints.

Full Ownership

Ensure data privacy and system resilience through a self-hosted architecture.

The Architectural Blueprint: The Four Zones of Operation

Our Personal OS is organized into four distinct, interconnected zones, each hosted on hardware best suited for its task. This architecture separates the creative process from the underlying infrastructure.



NotebookLM

The Creator Zone: The Workbench for Ideas

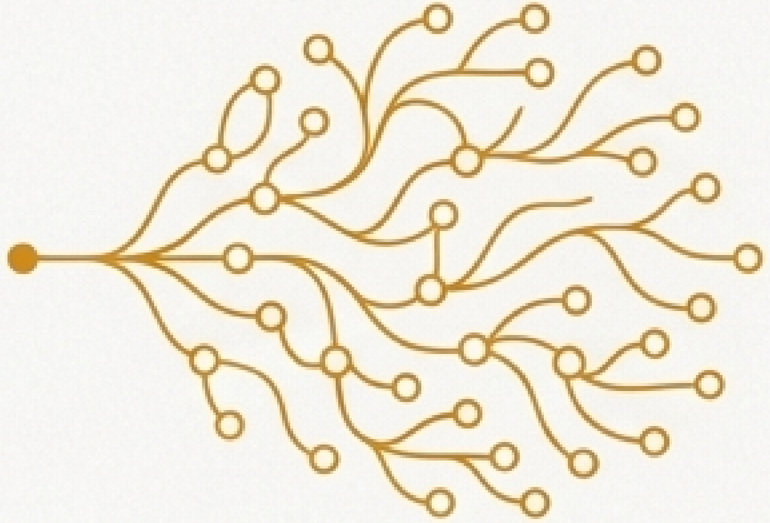
The desktop is the primary workstation. All tools here are designed for direct interaction, separating volatile brainstorming from permanent knowledge.

The 'RAM vs. SSD' Workflow

Volatile RAM (Processing & Exploration)

Tool: Rabbitholes AI

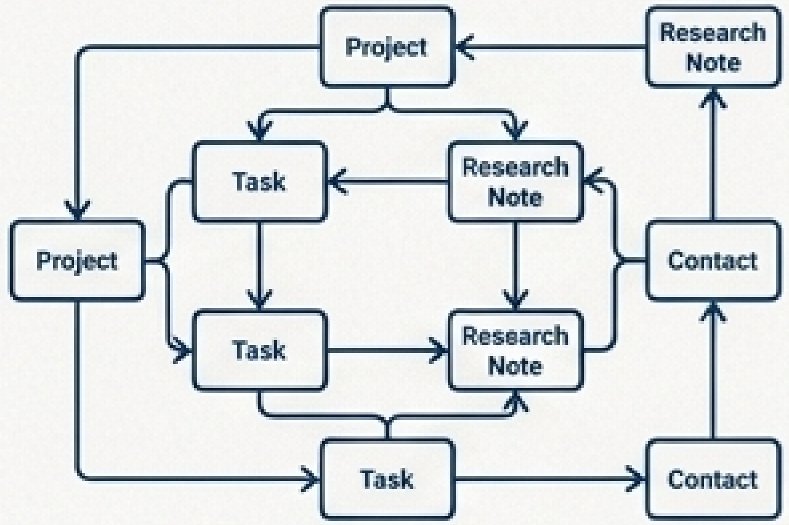
Function: The "Git for Thought." A visual interface for LLMs that structures conversations as branching trees. Perfect for following ADHD tangents without losing the main context. It's the sandbox for "blue sky" thinking.



Structured SSD (Permanent Knowledge)

Tool: AnyType

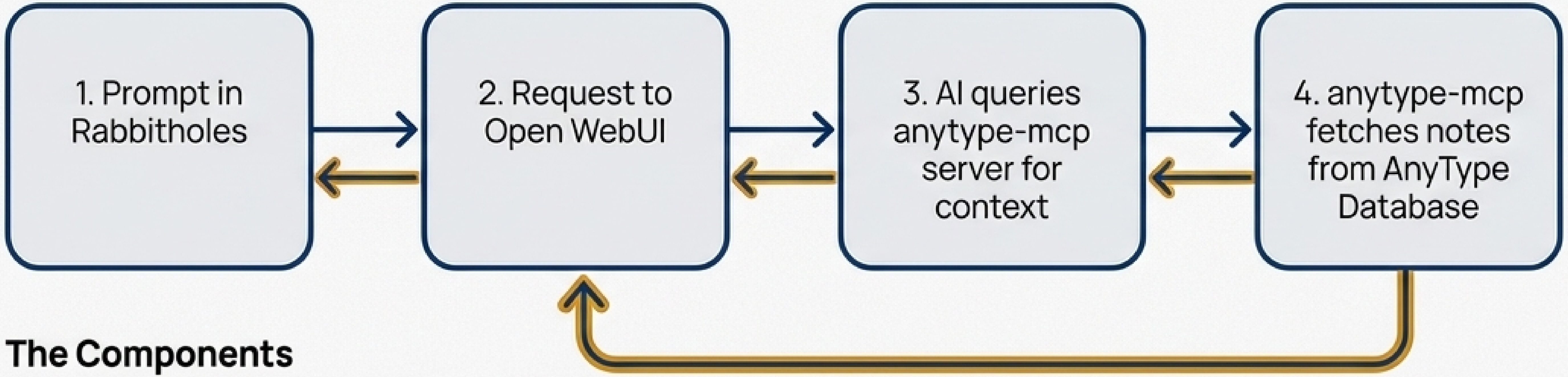
Function: The object-first "Database" and single source of truth. It enforces structure, creating relational objects for projects, tasks, and research notes. This is where crystallized ideas are committed.



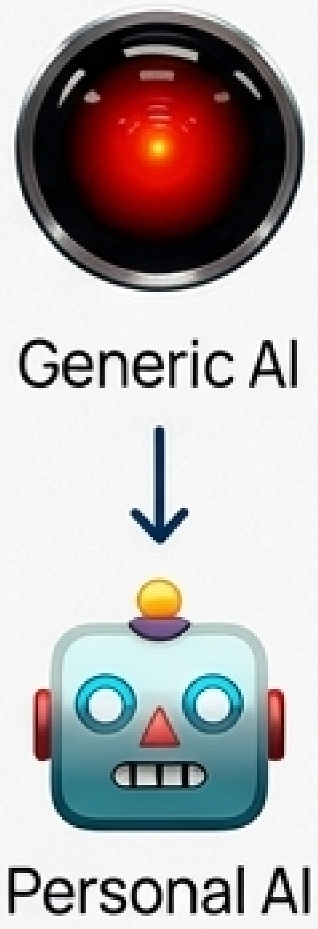
The Intelligence Zone: An AI That Knows Your Data

The Jetson Orin acts as a dedicated, low-power AI compute node. Its purpose is to offload heavy inference tasks from the desktop. The key component is the bridge that connects the AI to personal data.

5-Step Data Context Workflow


1. Prompt in Rabbitholes
2. Request to Open WebUI
3. AI queries anytype-mcp server for context
4. anytype-mcp fetches notes from AnyType Database

5. AI generates response using your data



The Components

The components of the anytie-mcp.

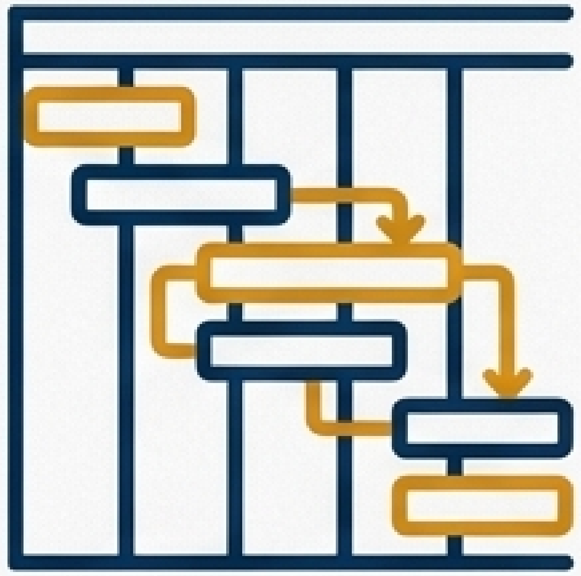
- **AI Brain:** Open WebUI provides the interface to various local and remote LLMs.
- **The Context Bridge:** The `anytype-mcp` (Model Context Protocol) server runs on the desktop, creating a secure 'phone line' between the AI and the AnyType knowledge base.

Result: Zero-friction context retrieval. The AI becomes a true personal assistant.

NotebookLM

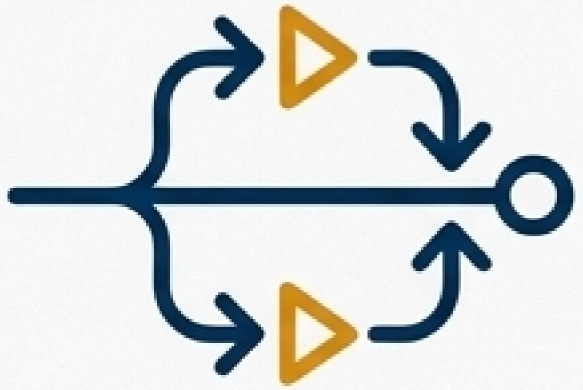
The Infrastructure Zone: The 24/7 Project Office

Hosted on the x86 Atomic Pi for maximum compatibility, this zone contains the services that turn ideas into deliverables. This is the system's "Prefrontal Cortex" for planning and execution.



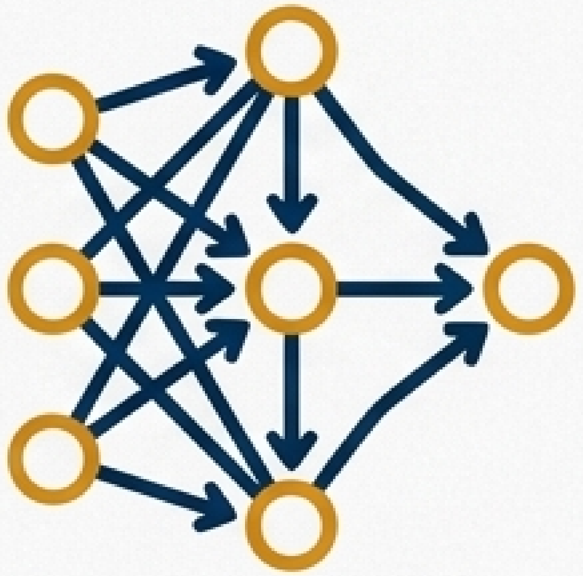
The Task Manager (Plane)

A self-hosted project management suite. Fills the gap left by AnyType, providing Gantt charts, timelines, sprint management. It defines *When* and *What* needs to be done.



The Code Repository (Forgejo)

The "factory floor" where code is stored, versioned, and managed.



The Automation Glue (n8n)

The "automated butler" or "nervous system." It watches for events across services and performs actions, eliminating the toil of manual data entry.

NotebookLM

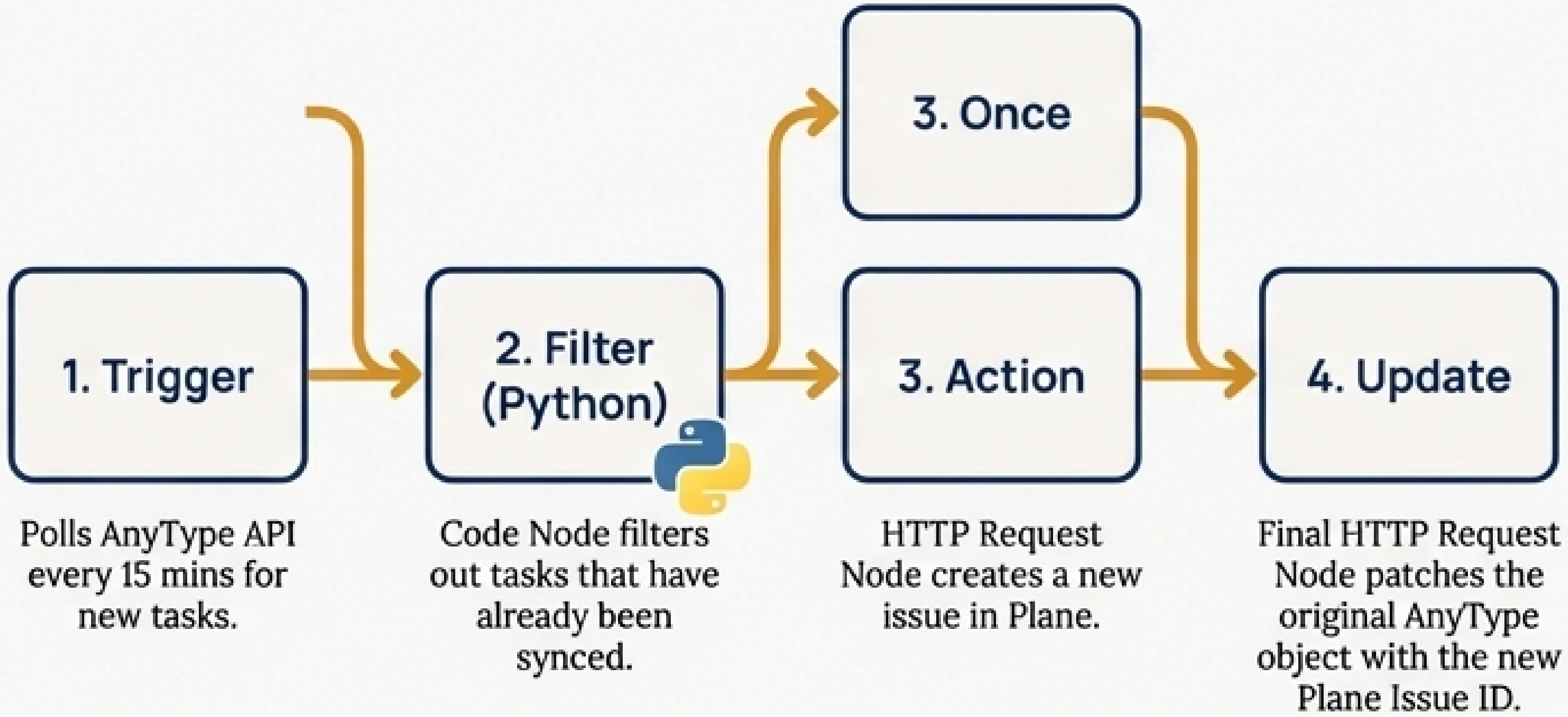
The Automation Glue: A Hybrid Approach with n8n

To minimize maintenance overhead, n8n serves as the central automation hub. Its visual interface reduces cognitive load and makes workflows self-documenting. We use a "Low-Code First" philosophy.

The Strategy

- **Orchestration Hub**: n8n manages the flow, timing, and error handling of all integrations using its library of pre-built nodes.
- **Embedded Python Power**: A custom n8n Docker image includes Python and the `anytype-client` library. This allows for powerful, specialized logic to be run inside a "Code Node" without managing external scripts or daemons.

Example Workflow: Syncing an Idea to a Task


1. Trigger

Polls AnyType API every 15 mins for new tasks.

2. Filter (Python)

Code Node filters out tasks that have already been synced.

3. Once

3. Action

HTTP Request Node creates a new issue in Plane.

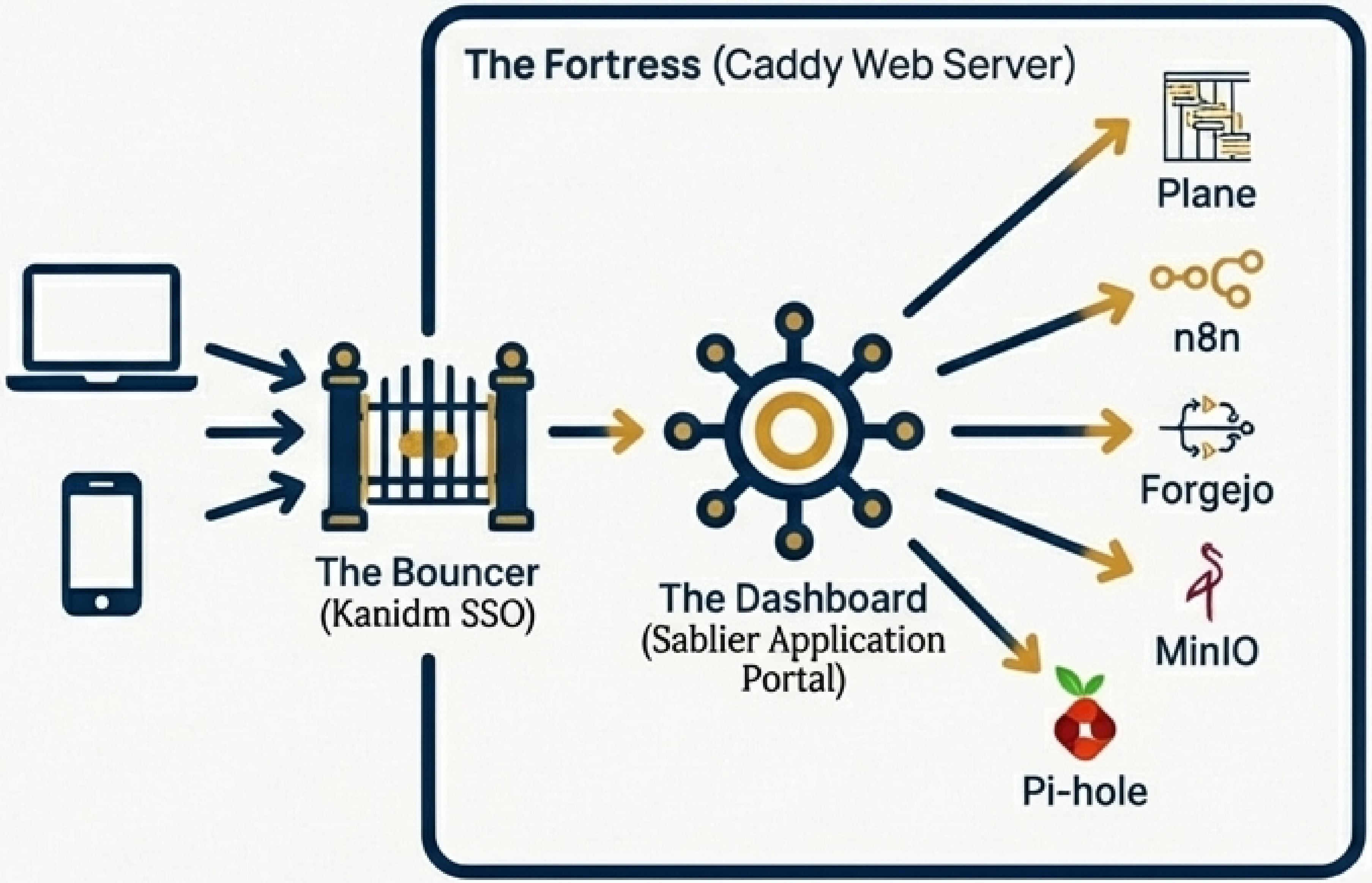
4. Update

Final HTTP Request Node patches the original AnyType object with the new Plane Issue ID.

The Foundation: A Unified & Secure Edge

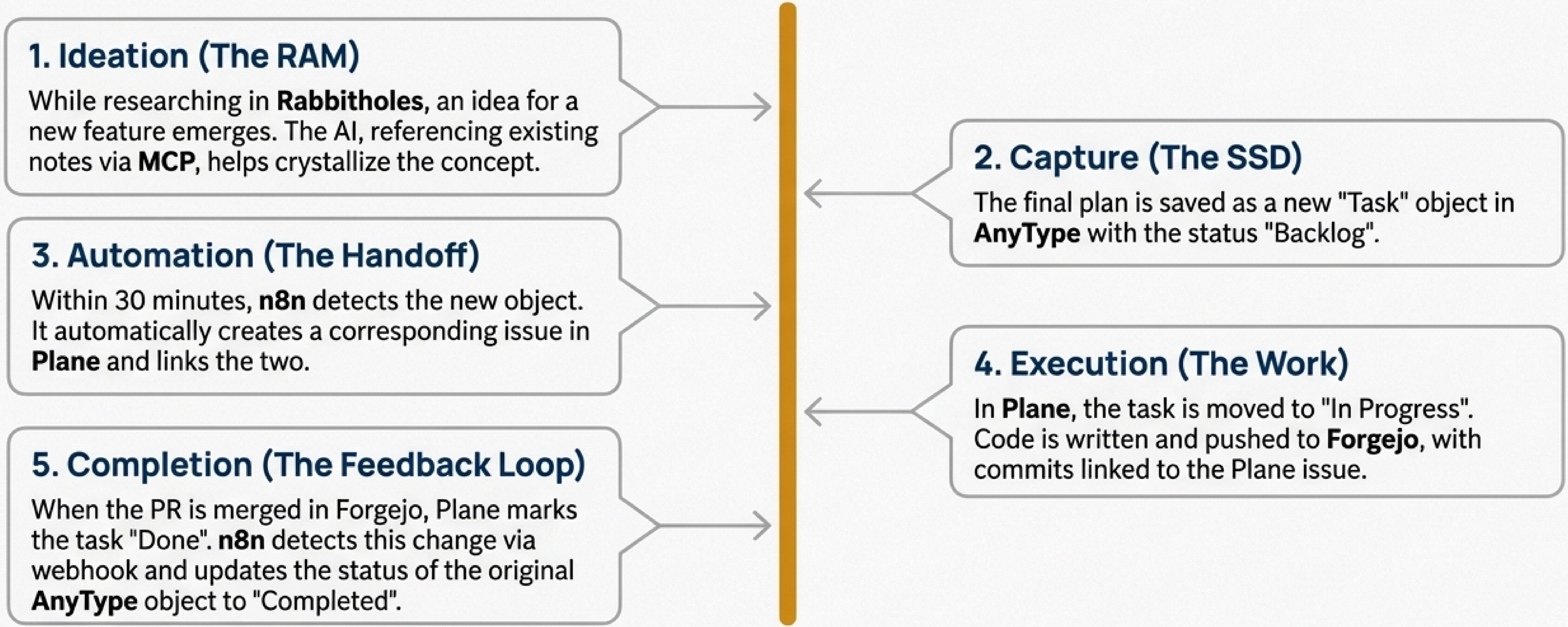
Running on a Raspberry Pi, the network and storage layer provides security, discovery, and resilience for the entire system. The core is a highly-customized Caddy web server.

- **The Bouncer (Caddy Security + Kanidm):** Implements Single Sign-On (SSO). Instead of logging into each service, a single login to Kanidm grants access to all protected applications.
- **The Dashboard (Sablier):** An “Application Portal” at `home.bitwise.ws’ that provides a visual grid of all services, their status, and one-click access. This eliminates URL hunting and reduces executive function load.
- **The Vault (MinIO + resticprofile):** A self-hosted S3-compatible object store for backups, with automated, incremental snapshots of critical data.
- **The Filter (Pi-hole):** Network-wide DNS filtering.



A Day in the Life: From Idea to 'Done'

This is how the system works in a typical development session.



1. Ideation (The RAM)

While researching in **Rabbitholes**, an idea for a new feature emerges. The AI, referencing existing notes via **MCP**, helps crystallize the concept.

2. Capture (The SSD)

The final plan is saved as a new "Task" object in **AnyType** with the status "Backlog".

3. Automation (The Handoff)

Within 30 minutes, **n8n** detects the new object. It automatically creates a corresponding issue in **Plane** and links the two.

4. Execution (The Work)

In **Plane**, the task is moved to "In Progress". Code is written and pushed to **Forgejo**, with commits linked to the Plane issue.

5. Completion (The Feedback Loop)

When the PR is merged in Forgejo, Plane marks the task "Done". **n8n** detects this change via webhook and updates the status of the original **AnyType** object to "Completed".

Result: A fully documented, end-to-end workflow with zero manual data transfer.

The ‘Green’ Cluster Map: Right-Sizing Workloads to Silicon

The system is distributed across a heterogeneous cluster of low-power SBCs and a primary desktop. Each service is strategically placed on hardware that matches its compute requirements and architecture, minimizing the overall carbon footprint.

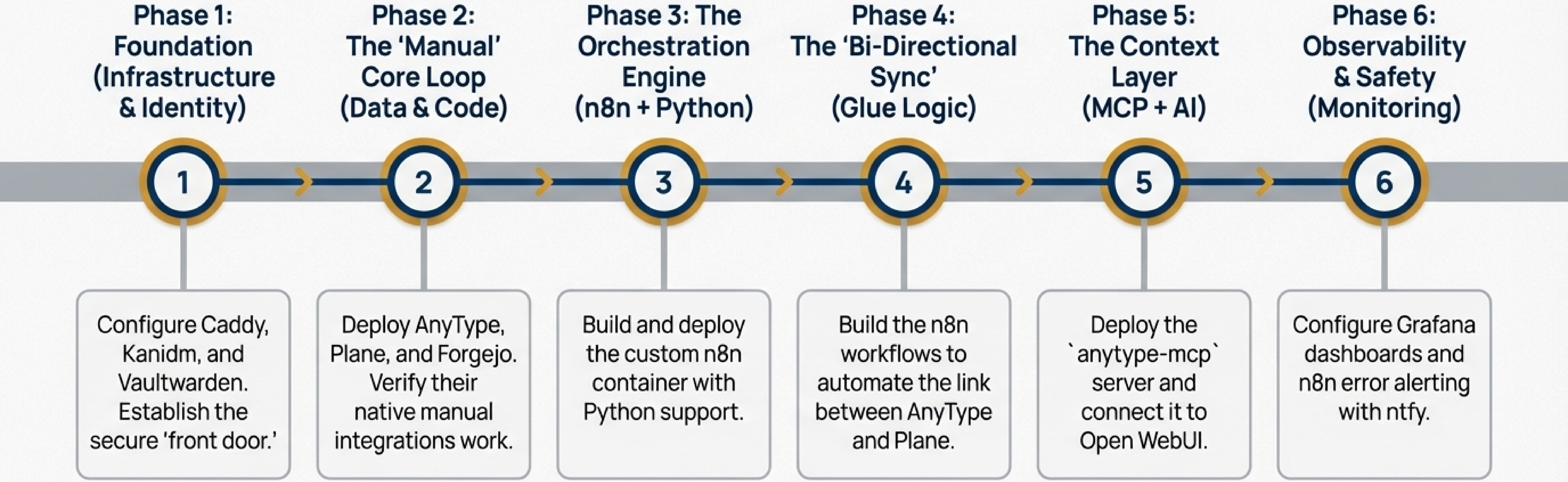
The Deployment Map

Node	Hardware	Role	Workloads
Desktop	AMD FX-8350	**Creator**	AnyType Client, Rabbitholes, IDE
Jetson Orin	ARM64 / 8GB RAM	**AI Hub**	Open WebUI, Local LLMs
Atomic Pi	x86_64 / Atom CPU	**Orchestrator**	n8n, Plane, Forgejo, MariaDB
RPi 4/5	ARM64	**Data Backbone**	Caddy, MinIO, AnyType Middleware

Key Principle: The power-hungry desktop is an ‘intermittent’ node, only active during work sessions. The low-power SBCs form the ‘always-on’ backbone, ensuring services like project management and code hosting are always available.

The Implementation Roadmap: An Incremental Build

The system is built in testable phases to ensure a sound foundation and isolate issues. Each phase delivers a functional piece of the stack.



Phase 1: Foundation (Infrastructure & Identity)

1

Configure Caddy, Kanidm, and Vaultwarden. Establish the secure 'front door.'

Phase 2: The 'Manual' Core Loop (Data & Code)

2

Deploy AnyType, Plane, and Forgejo. Verify their native manual integrations work.

Phase 3: The Orchestration Engine (n8n + Python)

3

Build and deploy the custom n8n container with Python support.

Phase 4: The 'Bi-Directional Sync' (Glue Logic)

4

Build the n8n workflows to automate the link between AnyType and Plane.

Phase 5: The Context Layer (MCP + AI)

5

Deploy the `anytype-mcp` server and connect it to Open WebUI.

Phase 6: Observability & Safety (Monitoring)

6

Configure Grafana dashboards and n8n error alerting with ntfy.

The Guiding Principle: Conforming the System to the Mind

This architecture is more than a collection of self-hosted services. It is a deliberate act of digital environment design.

The ultimate goal is to create a system that works *with* the grain of a neurodivergent mind—one that allows for non-linear exploration while providing an automated structure to capture the results.

It is a purpose-built machine for turning cognitive friction into creative flow.