Jarvis WSL Backend – Architectural Blueprint

This document defines the architectural-blueprint for the Jarvis-WSL backend. It is intended as a long-term reference for system design, responsibilities, workflows, and evolution strategy. This is not an implementation guide; it is a decision-level document.

1. Role Definition

The WSL backend is the cognitive core of the system. It is responsible for reasoning, planning, memory, decision-making, and learning. It does not execute OS-level actions directly. All atomic actions are delegated to the Windows controller.

2. Responsibility Split

Windows Controller:
Executes atomic actions, has no memory, no planning, no intelligence.

WSL Backend:
Interprets intent, plans multi-step tasks, evaluates risk, manages memory, and orchestrates execution.

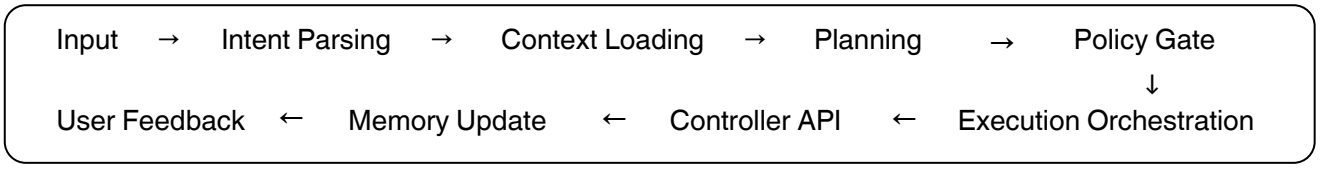
3. Core Backend Capabilities

- **Intent Understanding**: Convert raw input into structured intent with confidence.
- **Context Assembly**: Load conversation state, user profile, environment state, and relevant memories.
- **Planner**: Decompose intent into ordered, dependency-aware task graphs.
- **Policy & Safety Layer:** Enforce permissions, confirmations, and risk control.
- **Execution Orchestrator:** Track steps, handle failures, retry or replan.
- **Speech Loop:** Speech-to-text input and text-to-speech output.
- **Learning Loop:** Improve future behavior based on outcomes and feedback.

4. Memory Architecture

- **Short-term Memory:** Active conversation and task state.
- **Long-term Memory:** User preferences, habits, historical decisions.
- **Skill Memory:** Previously successful task plans and workflows.

5. Workflow Graph



This loop is continuous and stateful. Every execution updates the system's future behavior.

6. Design Philosophy

This backend is designed to be task-centric, not command-centric. Single-action commands are considered implementation details. The system is judged by its ability to complete goals, not trigger functions.