Fluffy Assistant Desktop

A Detailed Technical Analysis & System Documentation

Project Overview

🐰 Fluffy is a lightweight, privacy-focused system monitor and security guardian.

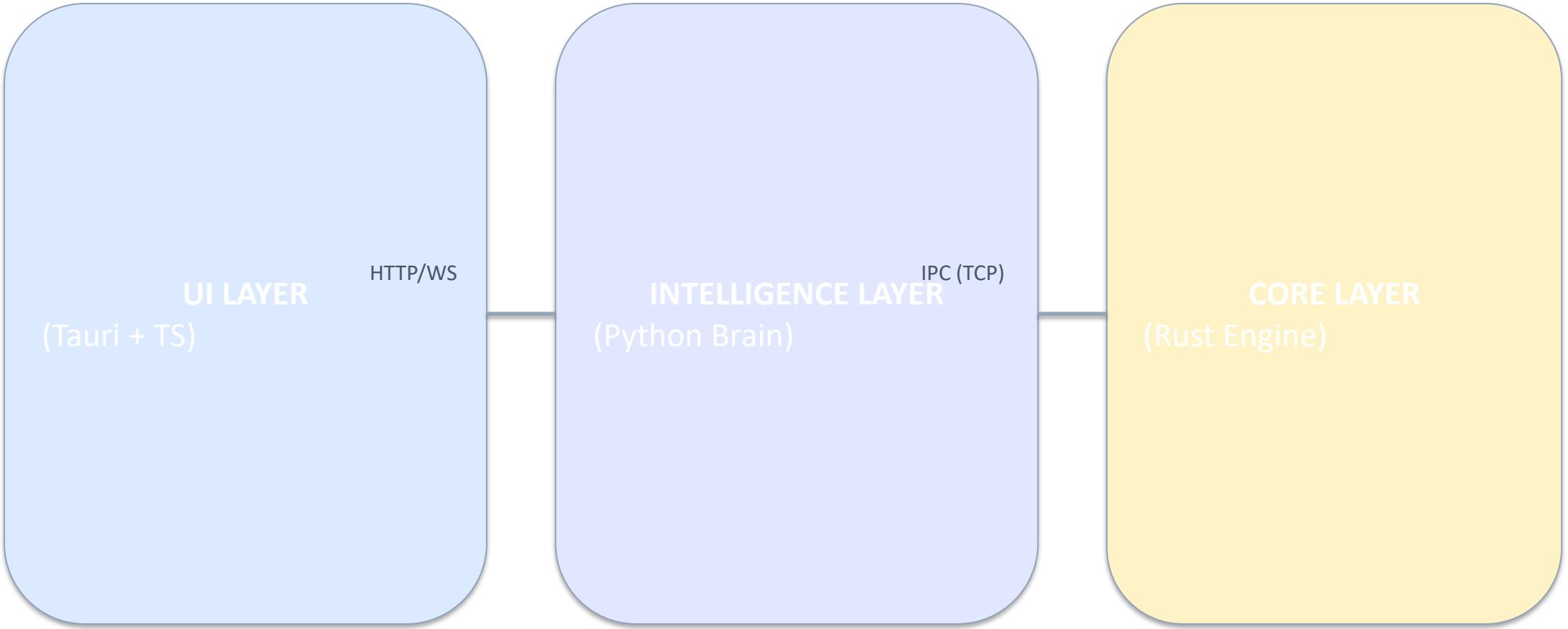
🛠️ Built for Windows using a high-performance multi-language stack (Rust, Python, TS).

🛡️ Features signature-less behavioral threat detection (Guardian).

🗣️ Integrated Voice Control and LLM Chat for an intelligent desktop experience.

🔒 100% Local Processing - Zero data exfiltration by design.

Hexagonal Architecture



File Structure Analysis

/core - Rust Monitoring Engine (Performance/System Access)

/brain - Python Intelligence Layer (Security/Logic/Memory)

/ui/tauri - Desktop Shell and IPC Hub

/ui/frontend - Modern Dashboard Assets (Vite/TypeScript)

/ai - LLM Connectors and Intent Classifiers

/voice - STT (Vosk) and TTS (Piper) Implementations

/fluffy_data - Persistent storage (Memory/Baselines/History)

Core Service: The Rust Engine

- 🏎️ Native precision: Collects system metrics every 100ms.
- 📡 Telemetry Hub: Broadcasts JSON states over TCP port 9001.
- ⚙️ Hardware Controller: Direct access to Volume and Brightness APIs.
- 🛡️ Process Manager: Safe termination and tree traversal via 'sysinfo'.
- 🌐 Network Monitor: Uses ETW for per-process packet tracking.

Code Deep-Dive: Rust Telemetry

```
loop {
    system.refresh_cpu_all();
    system.refresh_processes(ProcessesToUpdate::All);

    let stats = transform_to_fluffy_json(system);
    let message = json!({
        "type": "telemetry",
        "data": stats
    }).to_string();

    ipc_server.broadcast(message); // Port 9001
    thread::sleep(Duration::from_millis(2000));
}
```

- refreshes only necessary components to save CPU.
- serializes to a standardized 'Fluffy' schema.
- uses thread-safe broadcast to multi-client Brains.

Brain Service: The Python Intelligence

- 🧠 Semantic Monitor: Converts raw metrics into health signals.
- 🗄️ State Manager: Thread-safe repository of current system health.
- 🛡️ Guardian Engine: The primary behavioral analysis pipeline.
- 🔌 Flask API: Bridge between the native UI and the internal logic.
- 🎙️ Audio Manager: Coordinates STT input and TTS feedback.

Security Guardian: Detection DNA

1. Path Integrity: Execution from %TEMP% flagged immediately (+30).
2. Resource Spikes: CPU >3x baseline detected via EMA analysis (+20).
3. Child Spawning: Rapid creation of CLI sub-processes tracked (+25).
4. Persistence: Monitoring 'Run' registry keys and Startup folders (+40).

🚀 Learning Phase: A 5-minute baseline established on first startup.

Algorithm: Adaptive Learning (EMA)

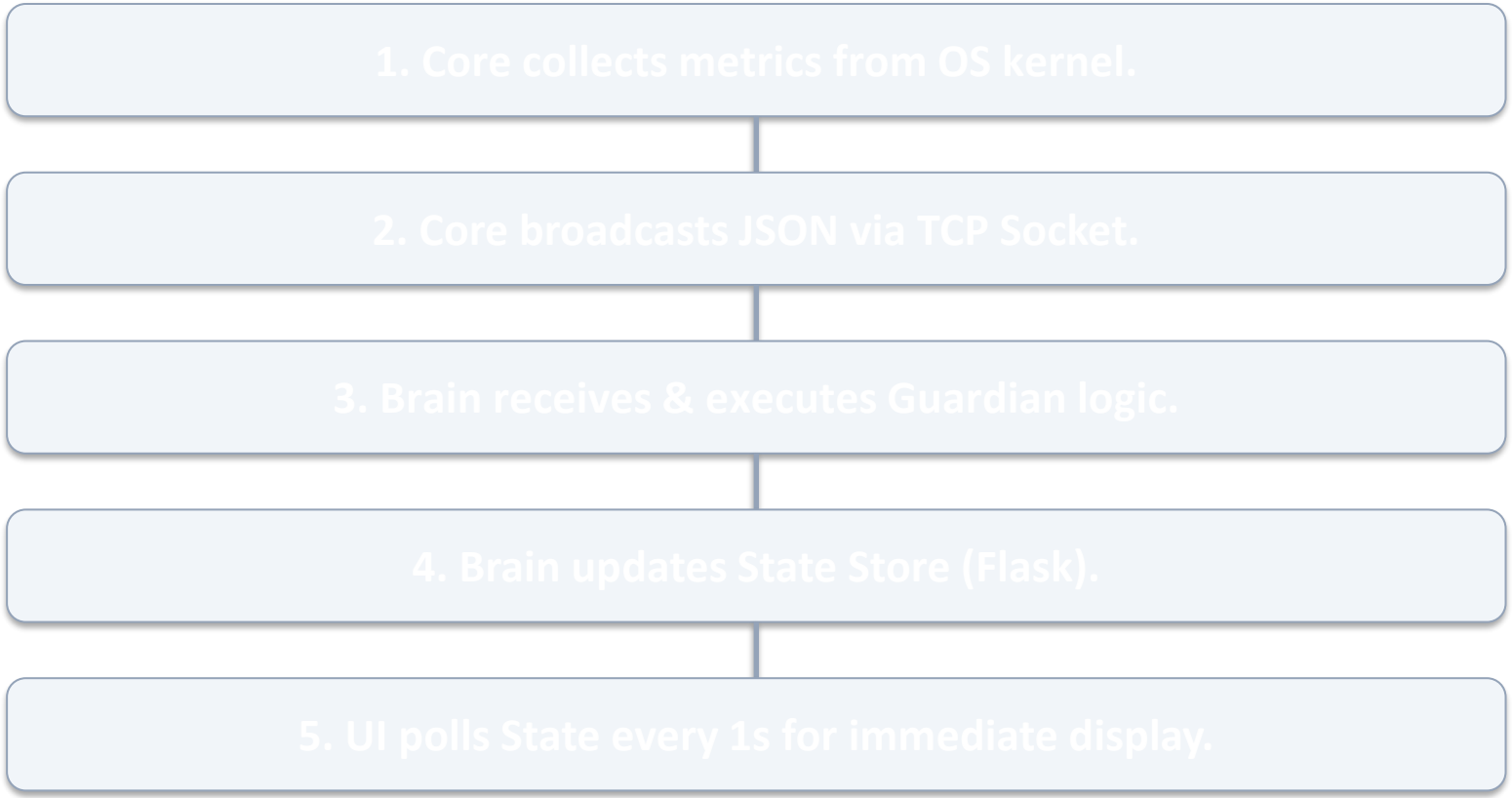
```
# math.py logic
def update_baseline(old_avg, new_val, alpha=0.01):
    return (alpha * new_val) + ((1 - alpha) * old_avg)

# Guardian update
baseline['avg_cpu'] = update_baseline(
    baseline['avg_cpu'],
    current_cpu
)
```

- alpha=0.01 provides 'long-term memory'.
- prevents false positives from transient spikes.
- adapts to user's daily software habits.

UI Layer: Modern Native Experience

- ✨ Windows Native performance via Tauri.
- 📊 Dynamic charting (EMA-smoothed) for CPU/RAM visualization.
- 🌳 Process Tree: Real-time parent-child hierarchy display.
- 💬 Interactive FAB: Floating chat button for instant AI access.
- 🎨 Glassmorphism design system with responsive layouts.

Workflow: Telemetry Data Pipeline



Persistent Memory System

- 💾 JSON Long-Term Store: Saves preferences, trusted PIDs, and identities.
- 🧠 Multi-Session Context: Remembers previous chat topics and actions.
- 🛡️ Trusted Whitelist: Processes marked as safe persist across reboots.
- 📝 Session Buffer: Tracks dangerous actions for delayed confirmation.

Safety Control: Interrupt Commands

Keywords: "Stop", "Cancel", "Abort", "Shut up", "Nevermind".

- ⚡ Immediate Cancellation: Halts TTS playback in <50ms.
- 🛑 Logic Reset: Clears all pending intents and confirmation prompts.
- 🔔 Visual Feedback: Toast notifications confirm the interrupt.

Natural Language Interface

- 🎤 STT: Vosk Offline Engine (privacy-centric).
- 📢 TTS: Piper Neural Voices (highly realistic).
- 🤖 LLM Support: OpenAI, Claude, Groq, and Ollama (Local).
- 🧪 Intent Mapping: Maps speech to 15+ internal system commands.

The Self-Improvement Cycle

- 👁️ Observer: Tracks user failures or 'I don't know' responses.
- 🏗️ Architect: Suggests new Python extensions to handle new commands.
- 🔨 Generator: Auto-writes and installs extensions into /brain/extensions.
- 📦 Dynamic Loading: New features become active without a system restart.

The 'Normalize' Feature

- 📢 Sound: Resets volume to 50% for optimal environment.
- 🌞 Visuals: Forces brightness to 70-80% to avoid eye strain.
- 🧹 Disk: Purges Windows /Temp folders of bloated logs.
- 🔍 Security: Triggers a deep Guardian scan for all active PIDs.
- 🎯 Goal: Bring messy systems back to a pristine baseline.

Conclusion & Future

- ✅ Technical Excellence: Multi-service architecture for speed and safety.
- ✅ Privacy First: No dependencies on third-party cloud data mining.
- ✅ Future: Cross-platform support and decentralized P2P telemetry.
- 🐰 Fluffy: Where System Monitoring meets Personal Intelligence.

Thank You!

Questions? | documentation/agent.md