Interim Forensic Audit Report: Swarm Architecture

Project: Automaton Auditor

Date: February 25, 2026

1. Architecture Decision Rationale

The architecture follows a **Digital Courtroom** model.For the interim phase, I have focused on establishing a typed, observable, and isolated runtime environment.

- **Pydantic State over Dicts:** I implemented AgentStateusing TypedDictand Pydanticmodels (eg, Evidenceclass) to enforce strict typing and prevent "Dict Soups"..
- **Parallel Reducers:** To enable a **Robust Swarm** , I used operator.iorfor the evidencesdictionary and operator.addfor lists.This ensures that when parallel detectives write to the state, they merge findings rather than overwriting each other.
- **AST Parsing over Regex:** The RepoInvestigatorutilizes Python's astmodule to verify the logical structure of code.This allows me to confirm if a StateGraphis actually instantiated and wired correctly, avoiding the brittleness of simple string matching..
- **Sandboxing Strategy:** All git operations are executed withintempfile.TemporaryDirectory().This provides security isolation, ensuring that third-party code being audited does not pollute my live working directory.

2. Gap Analysis and Forward Plan

The current implementation successfully completes the **Detective Layer** (Layer 1)..

Known Gaps:

- **Judicial Layer (Layer 2):** The Prosecutor, Defense, and Tech Lead nodes are drafted but not yet wired for dialectical reasoning..
- **Supreme Court (Layer 3):** The ChiefJusticeNodeand its deterministic conflict resolution rules (eg, Rule of Security, Fact Supremacy) are yet to be implemented.
- **Multimodal Analysis:** The VisionInspectoris planned but currently lacks the integration to parse architectural diagrams from PDFs.

Forward Plan:

- **Refine Judicial Prompts:** Develop persona-specific system prompts to ensure the Prosecutor is adversarial and the Defense is optimistic..
- **Logic-Based Synthesis:** Code the hardcoded Python logic for the Chief Justice to resolve score variances greater than 2.
- **Markdown Serialization:** Build the final output node to convert the AuditReportPydantic model into a formatted Markdown file.

3. StateGraph Architecture Diagram

The planned and partially implemented flow follows the mandated **Fan-Out / Fan-In** pattern to ensure non-linear execution..

- **Detective Fan-Out:** The STARTnode triggers both RepoInvestigatorand DocAnalystsimultaneously.
- **Evidence Fan-In:** An EvidenceAggregatornode (or State Reducer logic) synchronizes the collected facts before proceeding to the next layer.
- **Planned Judicial Fan-Out:** Once evidence is gathered, the three Judges will analyze the artifacts independently and in parallel..

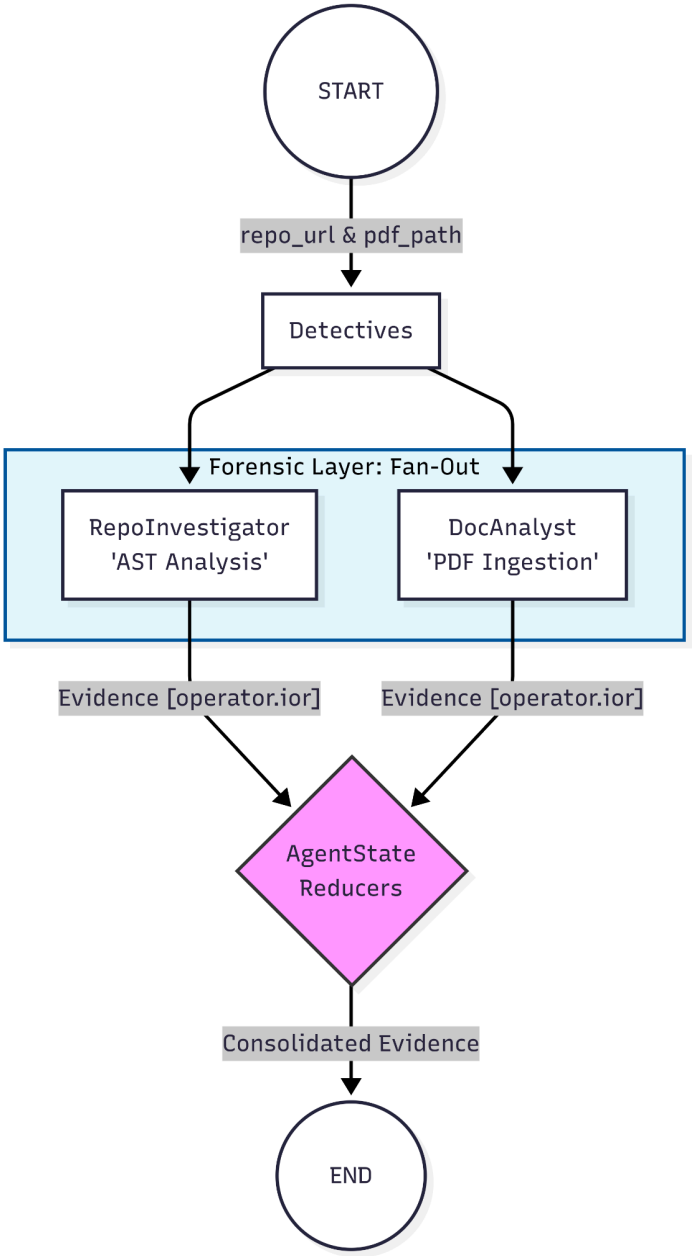

Figure 1: Parallel Forensic Fan-Out & State Reducer Architecture