

Automaton Auditor - Final Report

Executive Summary

This report walks through the design, implementation, and self-evaluation of my Automaton Auditor a multi-agent “Digital Courtroom” built with LangGraph to evaluate Week 2 submissions against the official rubric. At a high level, the system takes in a GitHub repository and a PDF report, spins up a swarm of Detective agents to gather structured evidence, and then routes that evidence through three Judge personas. A deterministic Chief Justice node synthesizes their competing views into a final, structured Markdown audit report.

Overall, the current implementation lands solidly at a “Competent Orchestrator” level on the Tenx rubric. The core StateGraph is correctly wired to support parallel Detectives and Judges, state is strictly typed using Pydantic and TypedDict, and the Chief Justice applies explicit, rule-based synthesis (such as security overrides and functionality weighting) instead of delegating judgment to an LLM. That said, there is still clear headroom to reach a “Master Thinker” level. The VisionInspector remains a stub, the fact-supremacy rule could rely more directly on concrete Evidence objects, and some remediation guidance could be more file-specific and operational.

Architecture Deep Dive

Dialectical Synthesis

Dialectical Synthesis is implemented by design, not by prompt rhetoric. The system enforces a three-persona judicial bench Prosecutor, Defense, and Tech Lead that independently evaluate each rubric dimension using the same shared evidence. Each Judge node produces structured output bound to a JudicialOpinion schema, which guarantees that every criterion is examined from three distinct and often conflicting perspectives: a strict security-focused view, an effort- and intent-oriented view, and a pragmatic architectural view.

These opinions are not averaged or loosely summarized. Instead, they are passed to the Chief Justice node, which applies explicit, named rules such as the Rule of Security (capping scores when security flaws are present) and the Rule of Functionality (giving extra weight to the Tech

Lead on architectural criteria). The resulting CriterionResult objects retain the original judge opinions, the final score, and a dissent summary, making the reasoning process visible rather than opaque.

Fan-In / Fan-Out Design

The StateGraph deliberately applies fan-out and fan-in patterns at two distinct layers, directly reflecting the Digital Courtroom specification.

Detective Fan-Out / Fan-In

From the start node, the graph fans out to three Detectives running in parallel: RepoInvestigator, DocAnalyst, and VisionInspector. Each reads from the shared AgentState (which includes the repository URL, PDF path, and rubric dimensions) and contributes structured Evidence objects into the evidences field using LangGraph reducers. The evidence_aggregator node then acts as a strict synchronization point, ensuring that all forensic findings are fully merged before any judicial reasoning begins.

Judge Fan-Out / Fan-In

Once evidence is aggregated, the graph fans out again to the three Judge nodes. Each Judge evaluates the same evidence against the same rubric dimensions, returning structured JudicialOpinion objects that are merged into state via an additive reducer. The Chief Justice then serves as the final fan-in point, consuming all opinions, resolving conflicts deterministically, and producing both an in-memory AuditReport and a persisted Markdown report.

This structure ensures that no single LLM call dominates the outcome. Every layer is forced to confront the same facts, reason independently, and then converge through explicit rules rather than generative consensus.

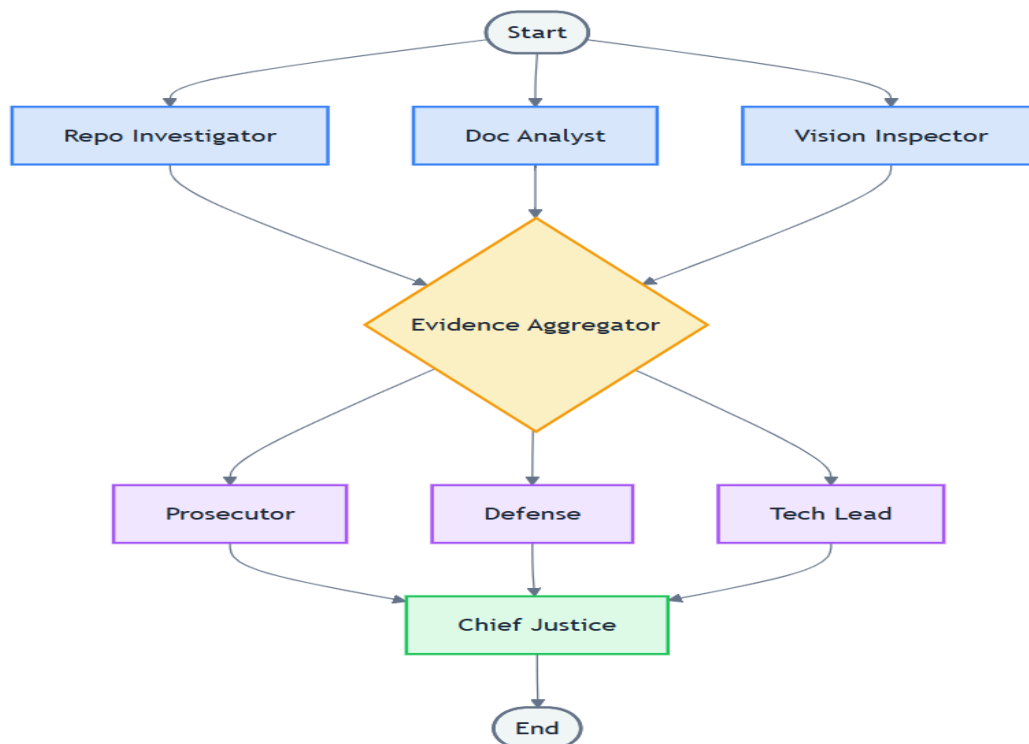
Metacognition

The system demonstrates metacognition in two ways that align closely with the assignment brief.

First, it enforces a strict separation between facts and judgments. Detective outputs are treated as forensic evidence, while Judge outputs are treated as subjective interpretations. The Chief Justice encodes a principle of fact supremacy: if a Judge's narrative claims deep metacognition or architectural sophistication but the evidence shows missing files or shallow explanations, that narrative is overridden and the final score is constrained.

Second, the system is capable of auditing itself. Running the Automaton Auditor against its own repository exposes weaknesses such as the stubbed VisionInspector or limited conditional edges. These findings feed directly into the remediation plan and future iterations. In this sense, the agent doesn't just evaluate others, it reasons about the quality and limits of its own reasoning loop.

Architectural Flow (Narrative Overview)



Discovery Phase: Investigators scan repositories, documents, and visual assets for raw data.

Argument Phase: Counsel teams interpret aggregated evidence from technical and legal perspectives.

Decision Phase: The Chief Justice renders a final verdict based on all competing inputs.

Criterion-by-Criterion Self-Audit (Summary)

At a glance, the system performs strongest in state management, structured output enforcement, and parallel orchestration, reflecting a strong architectural foundation. Areas with the most room for improvement are visual reasoning, documentation depth, and tighter coupling between evidence objects and synthesis rules. These gaps are well understood and already reflected in the remediation roadmap.