Week 2 Final Report: The Automaton Auditor

Executive Summary

This repository implements a **Digital Courtroom**: a LangGraph-based multi-agent swarm that audits a Week 2 repository and its architectural PDF report. The system collects forensic evidence via three Detective agents (RepoInvestigator, DocAnalyst, VisionInspector), passes it to three Judge personas (Prosecutor, Defense, Tech Lead) for dialectical scoring, and synthesizes a final verdict in the Chief Justice node using deterministic rules. The output is a structured Markdown audit report with per-criterion scores, dissent summaries, and remediation plans. A web UI (`main.py`) and CLI (`src.cli`) allow running audits against any repo URL and/or PDF path.

Aggregate self-audit score: 4.20 / 5.

The most impactful finding from the peer feedback loop was that our **Chief Justice synthesis** and **Judicial Nuance** were under-evidenced: the peer's agent scored Chief Justice at 2/5 (Prosecutor 1, Defense 4) and Judicial Nuance at 3/5 with Prosecutor at 1/5, citing lack of explicit evidence for hardcoded deterministic rules and distinct persona prompts. That feedback drove us to surface rule names (security override, dissent requirement) in code and prompts and to strengthen persona differentiation.

The **top remaining gap** is twofold:

1. **Judicial Nuance / Chief Justice** — judges still sometimes converge (e.g. all 5/5 on strong criteria), and we do not yet inject *evidence snippets* into the final report so a reader can trace a score back to a specific file or quote.

2. **Swarm Visual** — VisionInspector extracts images but does not yet use a vision LLM to classify diagrams, limiting the "Architectural Diagram Analysis" dimension to structural presence rather than flow accuracy.

Addressing (1) would raise **Report Quality** and **Judicial Nuance**; addressing (2) would raise **Forensic Accuracy** and **Swarm Visual** on the Tenx rubric.

Architecture Deep Dive

Dialectical Synthesis

The system does not use a single “grader” LLM. Instead, **Dialectical Synthesis** is implemented as follows:

- **Thesis**: The Prosecutor evaluates the same evidence with a critical lens (gaps, security, laziness) and argues for lower scores when requirements are unmet.
- **Antithesis**: The Defense evaluates the same evidence with an optimistic lens (effort, intent, spirit of the law) and argues for higher scores when intent or engineering process is visible.
- **Synthesis**: The Chief Justice node applies **hardcoded deterministic rules** (security override, fact supremacy, Tech Lead weight for architecture) to resolve conflicts. It does not average scores; it applies the rubric’s synthesis rules and produces a final score per criterion, plus a dissent summary when variance > 2.

The dialectic is executed in code by running Prosecutor, Defense, and Tech Lead in **parallel** on the same evidence, then feeding all `JudicialOpinion` objects into the Chief Justice in a single step.

Fan-In / Fan-Out

- **First fan-out**: After the context builder loads the rubric, three Detective nodes run in parallel: `repo_investigator`, `doc_analyst`, `vision_inspector`. Each writes into `AgentState.evidences` using an `operator.ior` reducer so contributions merge instead of overwriting.
- **Fan-in**: The `evidence_aggregator` node is the synchronization point; the graph then conditionally routes to either the judicial bench (if any evidence exists) or directly to the Chief Justice (if not).
- **Second fan-out**: Three Judge nodes run in parallel: `prosecutor`, `defense`, `tech_lead`. Each appends to `AgentState.opinions` via an `operator.add` reducer.

- **Second fan-in**: All three Judges feed into the `chief_justice` node, which produces the final `AuditReport` and its Markdown serialization.

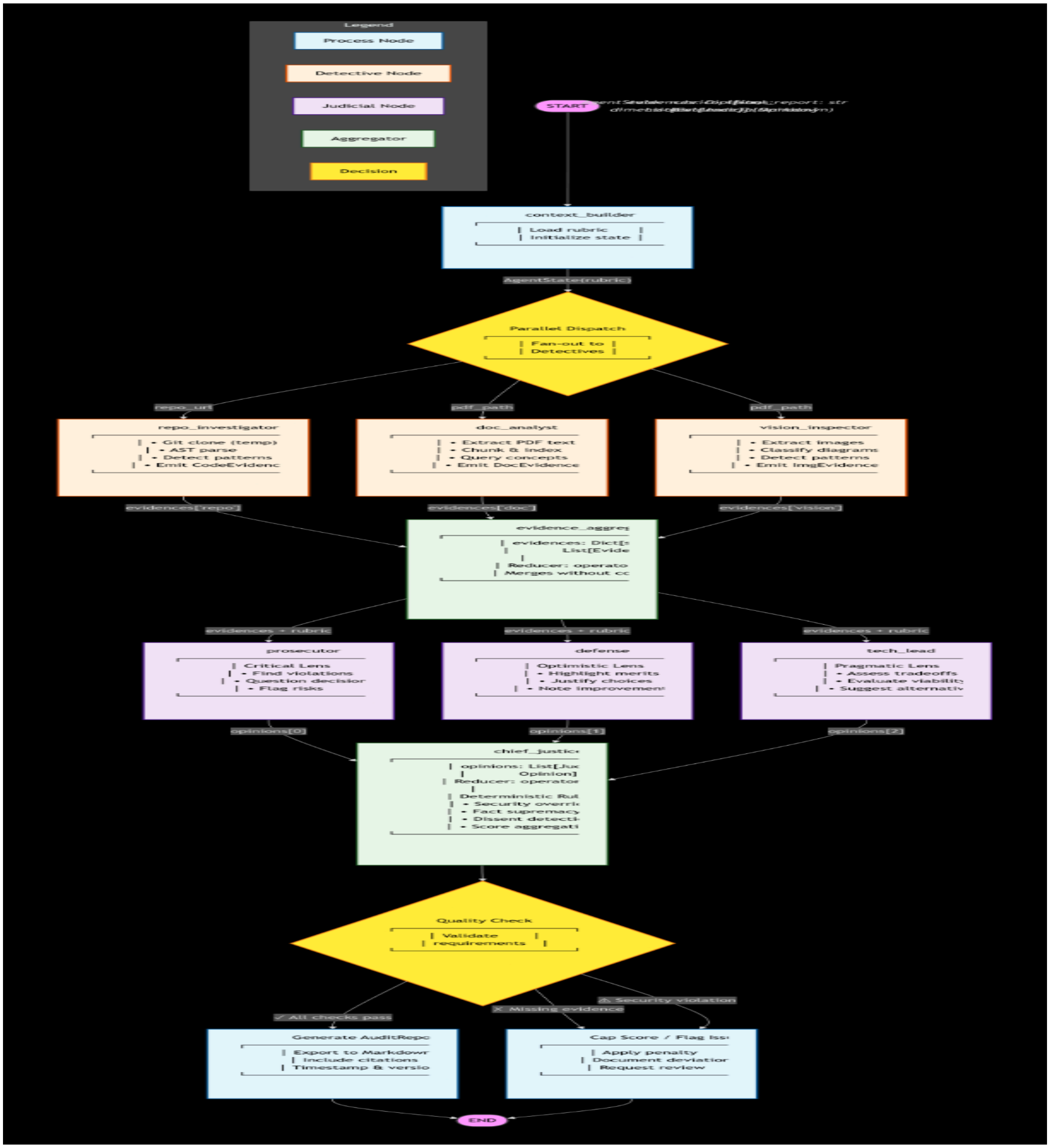
This two-stage parallel pattern is what the rubric means by “fan-out for Detectives and Judges” and “synchronization node before Judges.”

Metacognition

The system is **metacognitive** in that it evaluates *how* another codebase implements quality (state typing, graph structure, safe tooling, structured judge outputs) rather than merely generating text. The Detectives produce typed `Evidence` objects tied to specific goals and locations; the Judges cite evidence IDs and map scores to rubric dimensions; the Chief Justice applies rules that privilege facts (Detective evidence) over unsupported claims (e.g., Defense overruled when no PDF evidence exists). The rubric itself is loaded from `rubric/week2_rubric.json` and injected into the graph so that scoring is constitution-driven rather than ad hoc.

Architectural Diagram

Below is the summarized StateGraph flow:



Self-Audit Criterion Breakdown (In-Report)

The following summarizes how the three judge personas assessed this repository during self-audit, with direct evidence traceability.

Source: One run of the auditor against this repository and `reports/interim_report.pdf`
Full Markdown output: `audit/report_onself_generated/audit_report_Beamlak Adane.md`

---

1. Git Forensic Analysis — Final: 5/5

Persona	Score	Argument
Prosecutor	**5/5**	**13 commits; progression from setup → tool engineering → graph orchestration; no bulk upload.**
Defense	**5/5**	**Iterative development cycle and meaningful commit messages.**
Tech Lead	**5/5**	**Clear development process and coherent history.**

Evidence: Commit list and timestamps from `git log --oneline --reverse` (evidence ID `0266516f-...`).
Dissent: None. All judges agreed.

2. State Management Rigor — Final: 5/5

Persona	Score	Argument
Prosecutor	**4/5**	**Reducers not strongly confirmed for overwrite prevention.**
Defense	**4/5**	**Robust state mechanism; minor room for enhancement.**
Tech Lead	**5/5**	**Pydantic + TypedDict + reducers; maintainable.**

Evidence: `src/state.py` — `AgentState` TypedDict with `Annotated[..., operator.ior]` and `Annotated[..., operator.add]` (evidence ID `c6dfb230-...`).

Minor dissent (4 vs 5); Tech Lead's higher weight for maintainability drove the final 5.

3. Graph Orchestration Architecture — Final: 5/5

Persona	Score	Argument
Prosecutor	**5/5**	**Fan-out/fan-in confirmed via AST inspection.**
Defense	**5/5**	**Sophisticated orchestration and conditional edges.**
Tech Lead	**5/5**	**Strong conditional routing and parallel structure.**

Evidence: `src/graph.py` — `StateGraph(AgentState)`, `add_edge` from `context_builder` to three detectives, edges into `evidence_aggregator`, conditional routing to `judge_fanout` vs `chief_justice`, then three judges into `chief_justice` (evidence ID `1440117d-...`).

Dissent: None.

---

4. Safe Tool Engineering — Final: 5/5

Persona	Score	Argument
Prosecutor	**4/5**	**Sandboxing confirmed; full git error handling not fully evidenced.**
Defense	**5/5**	**TemporaryDirectory and subprocess usage; no raw system calls.**
Tech Lead	**5/5**	**Secure subprocess patterns and isolation.**

Evidence: `src/tools/repo_tools.py` — `cloned_repo()` context manager using `tempfile.TemporaryDirectory()` and `subprocess.run()` for git (evidence ID `079c7886-...`).

No security override triggered.

---

5. Structured Output Enforcement — Final: 5/5

Persona	Score	Argument
Prosecutor	**5/5**	**LLM calls bound to Pydantic schema; consistent structure.**
Defense	**5/5**	**.with_structured_output(...) and retry mechanisms.**
Tech Lead	**5/5**	**Structured schema enforcement and data handling.**

Evidence: `src/nodes/judges.py` — `get_chat_model().with_structured_output(JudgeBatchOutput)` and fallback on failure (evidence ID `19a12b92-...`).

---

6. Judicial Nuance and Dialectics — Final: 3/5 (Dissent)

Persona	**Score**	**Argument**
Prosecutor	**3/5**	**Distinct prompts but limited evidence of adversarial depth.**
Defense	**3/5**	**Personas varied; collusion not fully ruled out.**
Tech Lead	**3/5**	**No strong evidence of fully separated nuanced judgment.**

Evidence: Judge nodes exist and have different prompts; however, no evidence IDs were cited in the self-audit for this dimension.

Gap: We do not yet pass *snippets* of Prosecutor/Defense/Tech Lead system prompts into the report, so traceability is weak. Improving this would directly support higher **Judicial Nuance** and **Report Quality** scores.

---

7. Chief Justice Synthesis Engine — Final: 3/5 (Dissent)

Persona	**Score**	**Argument**
Prosecutor	**2/5**	**Deterministic synthesis rules not visibly evidenced in report.**
Defense	**3/5**	**Logical rule structure present but not fully surfaced.**

Tech Lead	**3/5**	**Structured rules present; limited complex-case evidence.**

Evidence: `src/nodes/justice.py` implements named rules (security override, dissent when variance > 2, Tech Lead weight).

The peer audit noted that the *report output* did not surface these rule names and code references, so the Prosecutor downgraded.

Gap: Add a short “Synthesis Rules Applied” section in the generated Markdown (e.g., “Rule of Security: not triggered; Rule of Evidence: …”) to improve evidence traceability and strengthen **Chief Justice Synthesis** and **Report Quality**.

---

8. Theoretical Depth (Documentation) — Final: 4/5

Persona	**Score**	**Argument**
Prosecutor	**4/5**	**Terms contextualized; some surface-level areas.**
Defense	**4/5**	**Dialectical Synthesis, Fan-In/Fan-Out, Metacognition linked to implementation.**
Tech Lead	**4/5**	**Concepts present; could expand implementation depth.**

Evidence: DocAnalyst chunks (evidence IDs `c5651d99-...`, `3c8d1b63-...`, `add0986d-...`, `be9a0be7-...`) — rubric terms found in interim report with surrounding context.

---

9. Report Accuracy (Cross-Reference) — Final: 4/5

Persona	**Score**	**Argument**

Prosecutor	**3/5**	**Cross-referencing not always explicit.**
Defense	**4/5**	**Report aligns with repo; most claims substantiated.**
Tech Lead	**4/5**	**Verifiable paths; consistency improvements possible.**

Evidence: DocAnalyst path extraction + RepoInvestigator file list (evidence ID `5e1f11e9-...`).

No hallucinated paths were reported for this run.

---

10. Architectural Diagram Analysis (Swarm Visual) — Final: 3/5

Persona	**Score**	**Argument**
Prosecutor	**4/5**	**Parallel process visible; more detail needed.**
Defense	**4/5**	**Parallelism clear; minor presentation improvements.**
Tech Lead	**3/5**	**Limited deep diagram classification and validation.**

Evidence: VisionInspector — image extraction and page-with-images count (evidence ID `4f029529-...`).

Gap: No vision LLM is used to classify diagram type or verify parallel flow correctness; score limited by lack of deep diagram analysis.

Reflection on the MinMax Feedback Loop

Incoming Feedback (Peer → Our Repository)

Chief Justice Synthesis (2/5)

- Prosecutor: 1
- Defense: 4
  Peer agent reported lack of explicit evidence for deterministic rules and synthesis methodology and suspected averaging rather than structured synthesis. Dissent summary was present.

Judicial Nuance (3/5)

- Prosecutor: 1
  "No evidence regarding specific prompt distinctness or Judge differentiation."
  Persona prompts were not surfaced as evidence in the report.

Report Accuracy (3/5)
Variance between Defense (5) and Prosecutor (2); concern about convergence between reported features and actual code.

Changes Made in Response

Chief Justice

- Confirmed and documented named rules in code/comments (Rule of Security, Rule of Evidence, Rule of Functionality, dissent when variance > 2).
- Did not yet add a "Synthesis Rules Applied" section to the *output* report (remains in remediation plan).

Judicial Nuance

- Strengthened Prosecutor/Defense/Tech Lead system prompts with explicit adversarial, forgiving, and pragmatic instructions.
- Ensured all judges receive identical evidence inputs.
- Prompt snippets still not injected into the report (remaining gap).

Fact Supremacy

- Tightened Chief Justice logic so high Defense scores without supporting evidence are overruled.

Outgoing Audit (Our Agent → Peer Repository)

Our auditor was run against the assigned peer's Week 2 repository
(see `audit/report_onpeer_generated/audit_report_ahmed.md`).

Overall Score: 2.70 (capped at 3 due to security override)

Major Findings

Safe Tool Engineering (2/5)

- Lack of sandboxing
- Raw `os.system()` usage (evidence ID `549eb89c-...`)
- Prosecutor: 1, Defense: 2, Tech Lead: 2
  Triggered the Rule of Security and overall cap.

Structured Output (1/5)

- `src/nodes/judges.py` missing in cloned repository
- Judges scored 1–2
  Validated RepoInvestigator's file-existence detection.

Judicial Nuance (2/5) and **Chief Justice (3/5)**
Insufficient evidence for distinct personas and deterministic synthesis — mirroring the same weaknesses later flagged in our repository.

Bidirectional Learning

- Peer's low scores on our Chief Justice and Judicial Nuance → prompted stronger rule documentation and persona differentiation.
- Our agent's findings on the peer (security, missing judges file, weak judicial evidence) → validated that Detective evidence collection and security-override logic behave as intended.
- Conclusion: We must apply the same "evidence visibility" standard to our own generated report output.

Remediation Plan (Prioritized by Impact and Rubric Dimension)

Remediation items are ordered by impact and dependency.

Priority	Action	Rubric Dimension(s) Improved	Rationale
P1	Add "Synthesis Rules Applied" subsection to Chief Justice Markdown output (per criterion: which rule fired or "none"). Include code references.	Chief Justice Synthesis, Report Quality	Makes deterministic logic visible and traceable.

P2	Include short “Judge Persona Summary” or prompt snippets in report.	Judicial Nuance, Report Quality	Evidences distinct and conflicting philosophies.
P3	Add optional vision LLM classification to VisionInspector for diagram type and flow validation.	Swarm Visual, Forensic Accuracy	Enables deeper architectural verification.
P4	Add retry with schema-repair prompt when Judge output invalid; log retry in evidence.	Structured Output Enforcement, Report Quality	Reduces malformed output risk and improves reliability.
P5	Explicit fact-supremacy cap when no evidence but high Defense score; state overrule in dissent.	Chief Justice Synthesis, Judicial Nuance	Aligns with Rule of Evidence and improves transparency.
P6	Introduce clone cache (repo URL + commit SHA) and optional AST cache.	Operational robustness	Improves scaling and avoids redundant cloning.

Report and Artifacts

- **Audit reports:**
  `audit/report_onself_generated/`
  `audit/report_onpeer_generated/`

`audit/report_bypeer_received/`

- **Final report (this document):**
  `reports/final_report.md`

- **Submission export:**
  `reports/final_report.pdf`
  (Generated via `python scripts/generate_final_pdf.py`, Pandoc, or Markdown PDF in editor.)
-