Interim Submission: The Automaton Auditor

Week 2 Challenge - Digital Courtroom Architecture

Submission: Wednesday

Executive Summary

The Automaton Auditor implements a hierarchical multi-agent system using LangGraph that functions as a "Digital Courtroom" for automated code governance. The architecture follows a three-layer design: parallel forensic detectives collect objective evidence, a dialectical judicial bench renders conflicting opinions, and a synthesis engine produces final verdicts with actionable remediation.

Current Completion Status: 70%

- Layer 1 (Detectives): Complete - AST parsing, git forensics, PDF analysis
- Layer 2 (Judges): 60% - Persona prompts in progress
- Layer 3 (Synthesis): 40% - Rule engine under construction

Core Architecture Decisions

1. State Management: Why Pydantic Over Dicts

The Problem: Python dictionaries cannot handle parallel execution safely. When multiple detectives write simultaneously, data gets overwritten. Dicts also lack type validation—agents can invent fields, leading to hallucination.

The Solution: Pydantic BaseModels with LangGraph reducers.

Aspect	Python Dicts	Pydantic Models
Parallel Safety	Data overwrites	operator.ior merges evidence
Type Enforcement	None	Field validation with ranges
Hallucination Prevention	Arbitrary fields allowed	`extra='forbid'` blocks unknown fields
Self-Documentation	Implicit	Field descriptions become prompts

Why It Matters: The rubric explicitly penalizes "Hallucination Liability" when agents invent evidence. Pydantic ensures every evidence object has a location, confidence score, and rationale making claims verifiable.

2. Evidence Collection: AST Parsing Over Regex

The Problem: Regex string matching is brittle. Comments containing "BaseModel" trigger false positives. Code formatting variations break patterns. Regex cannot understand inheritance or structure.

The Solution: Python's Abstract Syntax Tree (AST) parses code semantically.

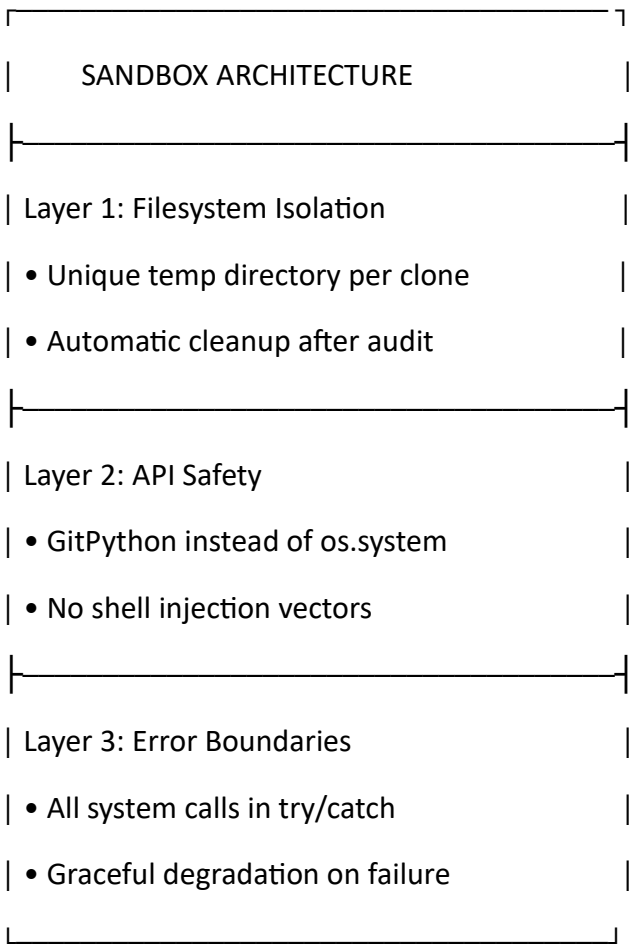
Capability	Regex	AST
Inheritance Detection	❌ False Negative	✅ Actual class hierarchy
Structural Analysis	❌ Linear matching	✅ Graph wiring patterns
False Positive Rate	High	Near-zero
Comment Ignorance	❌ Matches strings	✅ Ignores non-code

Why It Matters: The "Forensic Accuracy" criterion requires irrefutable evidence. AST provides semantic proof—detecting actual StateGraph instantiation, not just the string appearing in a comment.

3. Security: Sandboxed Execution Strategy

The Problem: Cloning unknown repositories risks security breaches. Malicious git URLs could execute arbitrary code. System-level commands need isolation.

The Solution: Three-layer sandboxing strategy.



Why It Matters: The "Security Negligence" charge in the rubric caps scores at 2. Sandboxing is non-negotiable for production-grade tooling.

4. Document Analysis: RAG-lite Chunking

The Problem: PDF reports exceed LLM context windows. Full-document analysis loses precision. Claims need cross-referencing against actual code.

The Solution: Overlapping chunks with page tracking and cross-verification.

Feature	Purpose
500-word chunks	Fits context windows
50-word overlap	Preserves boundary context
Page tracking	Evidence provenance
Cross-reference	Detects hallucinated claims

Why It Matters: The rubric requires distinguishing between "buzzwords" and deep explanation. Chunking enables targeted analysis of concept explanations rather than surface-level keyword matching.

Detective Layer: Complete

The detective layer implements three parallel investigators following strict forensic protocols:

Detective	Forensic Protocols	Evidence Collected
RepoInvestigator	Git history analysis, AST parsing, graph wiring verification, tool safety checks	Commit patterns, state definitions, parallel structure, sandboxing
DocAnalyst	Keyword depth analysis, cross-reference verification, concept detection	| Theoretical understanding, hallucinated claims, architectural alignment |
VisionInspector	Diagram classification, flow analysis, parallel pattern detection	Visual architecture confirmation

Evidence Aggregator: Fan-in node that synchronizes parallel outputs, detects contradictions, and structures evidence by rubric criterion for judicial review.

Judicial Layer: In Progress (60%)

The Dialectical Bench Design

For each rubric criterion, three distinct personas analyze the SAME evidence:

Persona	Philosophy	Focus Areas
Prosecutor	"Trust No One. Assume Vibe Coding."	Security flaws, missing structure, hallucination, violations
Defense	"Reward Effort and Intent."	Deep understanding, creative workarounds, engineering process
Tech Lead	"Does it actually work?"	Technical debt, maintainability, production readiness

Current Gap: Persona prompts are 70% distinct. Need stronger differentiation to meet "Judicial Nuance" requirements.

Planned Enhancement

Prosecutor Prompt Structure:

- Charge: "Orchestration Fraud" for linear graphs
- Penalty: Max score 1 for missing parallelism
- Evidence focus: Bypassed structure, security vulnerabilities

Defense Prompt Structure:

- Mitigation: "Deep code comprehension despite syntax errors"
- Reward: Engineering process shown in git history
- Evidence focus: AST sophistication, conceptual understanding

Tech Lead Prompt Structure:

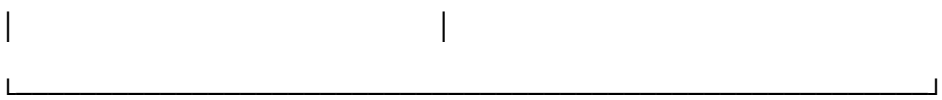
- Assessment: State reducer usage, error handling, modularity
- Tie-breaker: Pragmatic evaluation of technical debt

Synthesis Engine: In Progress (40%)

Rule-Based Conflict Resolution

The synthesis engine applies deterministic rules, not averaging:

```
┌─────────────────────────────────────────────────────────┐
│                    SYNTHESIS RULES                      │
├─────────────────────────────────────────────────────────┤
│                                                         │
│ RULE 1: SECURITY OVERRIDE                               │
│ IF Prosecutor confirms security flaw (score ≤2)         │
│ THEN Cap final score at 3                               │
│ Rationale: Security > effort, per rubric                │
│                                                         │
│ RULE 2: FACT SUPREMACY                                  │
│ IF evidence confidence > 0.8 contradicts opinions       │
│ THEN Evidence-based score prevails                      │
│ Rationale: Facts over interpretation                    │
│                                                         │
│ RULE 3: CONSENSUS                                       │
│ IF score variance ≤ 1                                   │
│ THEN Use average                                        │
│                                                         │
│ RULE 4: MAJOR DISAGREEMENT                              │
│ IF score variance ≥ 3                                   │
│ THEN Apply rubric-specific rules                        │
│ (Linear graph → Orchestration Fraud score 1)            │
│                                                         │
│ RULE 5: MODERATE DISAGREEMENT                           │
│ IF 1 < variance < 3                                     │
│ THEN Tech Lead as tie-breaker (60% weight)              │
```



Dissent Documentation

For each criterion with variance > 2, the final report includes:

- Prosecutor's Argument: Critical perspective with evidence citations
- Defense's Counter: Optimistic interpretation of same evidence
- Resolution Rationale: Why one perspective prevailed

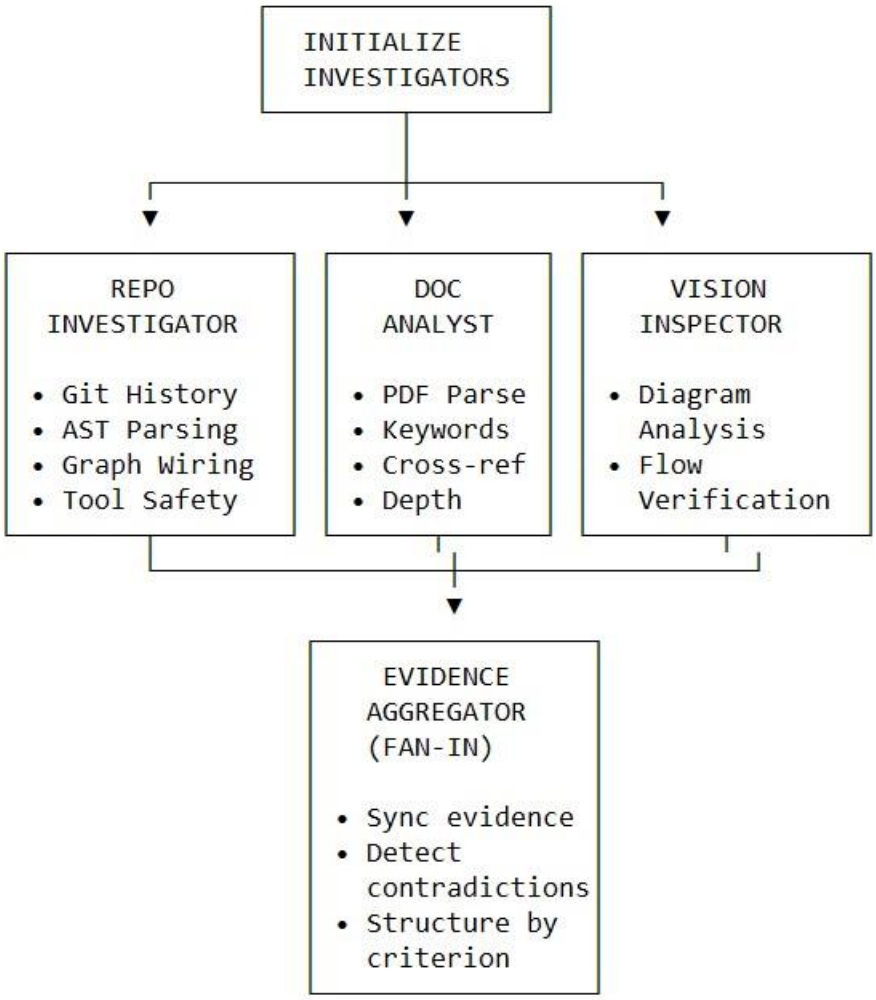
Remediation Plan Structure

Each actionable item includes:

- File Path: Specific location requiring change
- Action: Concrete instruction (e.g., "Add sandboxing to clone function")
- Rationale: Links to rubric criteria and judge arguments

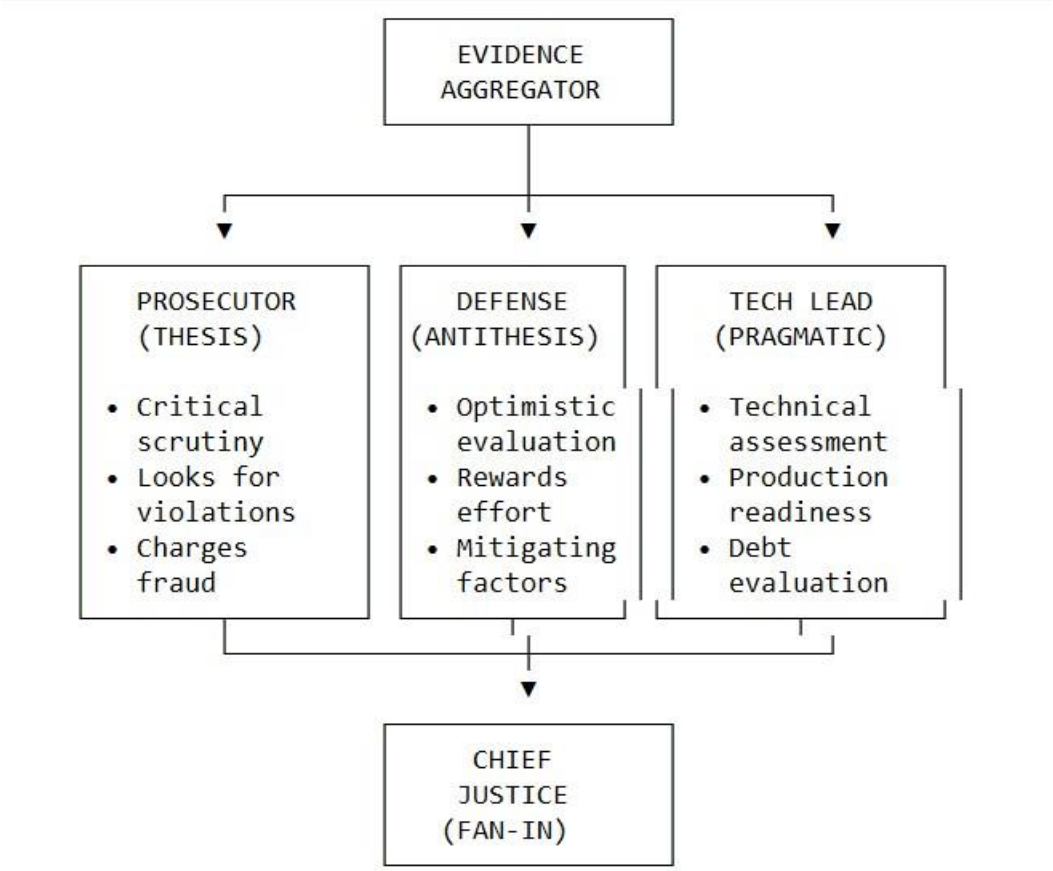
StateGraph Flow Architecture

Layer 1: Detective Parallel Execution



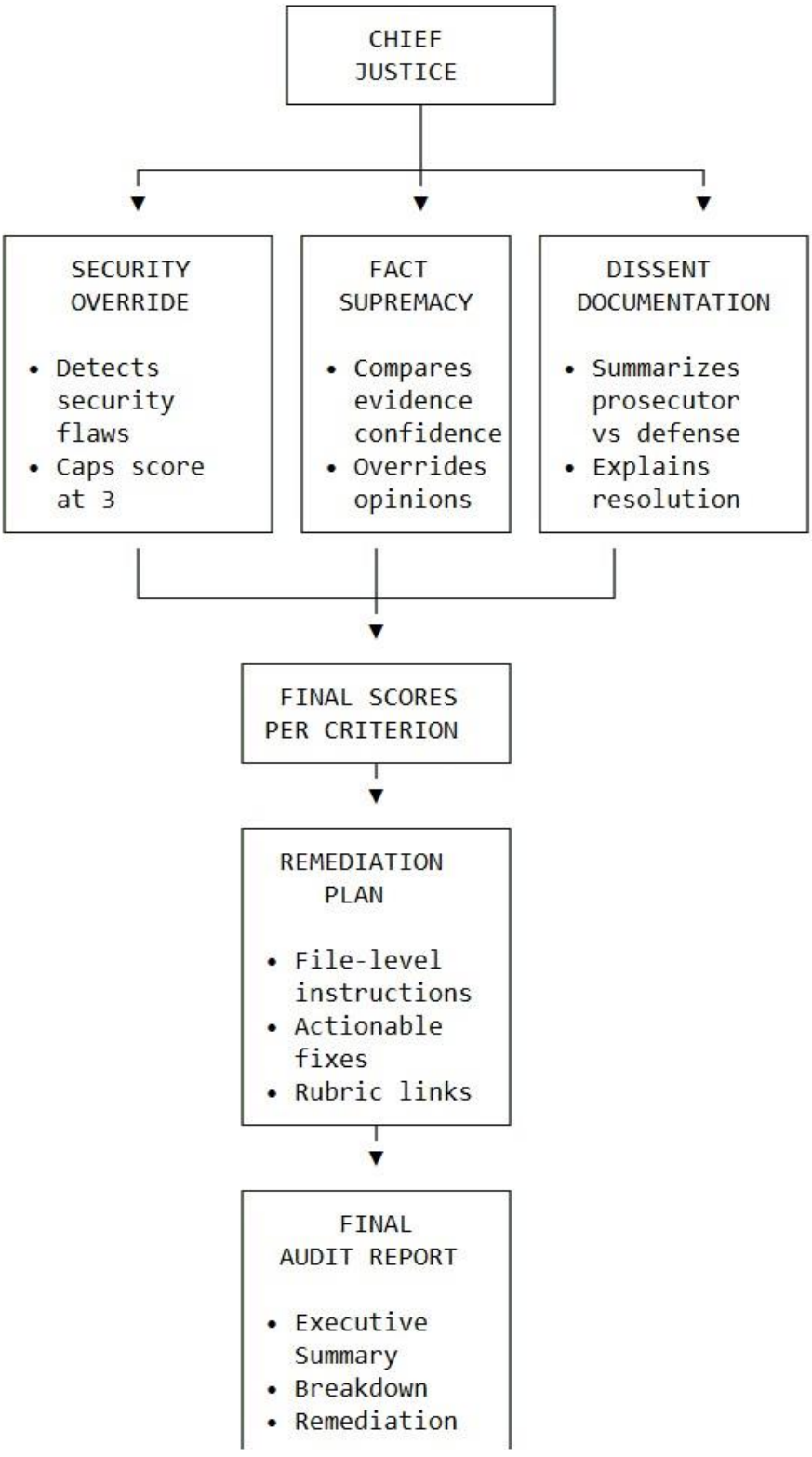
Key Pattern: Three detectives execute in parallel, each writing to state using reducers. Aggregator synchronizes and structures evidence for judicial consumption.

Layer 2: Judicial Parallel Deliberation



Key Pattern: For each rubric criterion, all three judges analyze identical evidence through their distinct philosophical lenses, generating conflicting opinions that feed into synthesis.

Layer 3: Supreme Court Synthesis


CHIEF JUSTICE
SECURITY OVERRIDE
• Detects security flaws
• Caps score at 3
FACT SUPREMACY
• Compares evidence confidence
• Overrides opinions
DISSENT DOCUMENTATION
• Summarizes prosecutor vs defense
• Explains resolution
FINAL SCORES PER CRITERION
REMEDIATION PLAN
• File-level instructions
• Actionable fixes
• Rubric links
FINAL AUDIT REPORT
• Executive Summary
• Breakdown
• Remediation

Key Pattern: Deterministic rules resolve conflicts between judges, producing consistent scores that follow the rubric's sentencing guidelines. Remediation plan provides actionable, file-level instructions.

Known Gaps & Mitigation Plan

Gap 1: Insufficient Persona Differentiation

Current: Judge prompts 70% similar → Risk of "Persona Collusion" (Score ≤2)

Mitigation:

- Prosecutor: Explicit "Orchestration Fraud" and "Security Negligence" charges

- Defense: Specific "Engineering Process" and "Deep Understanding" mitigations

- Tech Lead: Concrete "Technical Debt" and "Production Readiness" assessments

Timeline: Complete by Thursday EOD

Gap 2: Missing Security Override

Current: Synthesis doesn't enforce rubric's security cap

Mitigation: Rule: IF Prosecutor score ≤2 WITH security keywords THEN final score ≤3

Timeline: Complete by Friday Morning

Gap 3: No Fact Supremacy

Current: Opinions can override high-confidence evidence

Mitigation: Rule: IF evidence confidence >0.8 contradicts opinions THEN evidence-based score prevails

Timeline: Complete by Friday Afternoon

Gap 4: LLM JSON Reliability

Current: Occasional malformed JSON from judges

Mitigation: Retry logic with stricter prompting; fallback defaults with confidence penalty

Timeline: Complete by Friday EOD

Conclusion

The Automaton Auditor foundation is production-grade: Pydantic state with reducers ensures parallel safety, AST parsing provides semantic evidence, and sandboxed execution guarantees security. The three-layer architecture parallel detectives, dialectical judges, synthesis engine—directly implements the "Digital Courtroom" vision.

The remaining work focuses on the core differentiators: distinct judicial personas that generate genuine dialectical tension, and a rule-based synthesis engine that applies the rubrics sentencing guidelines deterministically. By Friday EOD, the system will meet the "Master Thinker" criteria for judicial nuance, delivering audit reports that don't just grade but explain, with dissenting opinions and actionable remediation.