# Week 2 Interim Report - Automaton Auditor

## 1. Introduction

This report documents the architecture decisions and implementation strategy for Week 2 of the FDE Challenge, 'The Automaton Auditor.' It focuses on the design of the Detective layer and the StateGraph orchestration of evidence aggregation, preparing the groundwork for the Judicial Layer and Synthesis Engine.

## 2. Architecture Decisions Made So Far

### 2.1 Pydantic over dicts

Typed, validated state ensures consistency across multi-agent nodes. Benefits include automatic type checking, serialization between nodes, and support for reducers (operator.add, operator.ior) for fan-in merging of evidence.

### 2.2 AST Parsing Structure

Tree-Sitter parses ASTs to verify tool registration and code structure. Avoids brittle regex-only parsing. Evidence is precise and auditable.

### 2.3 Sandboxing Strategy

Clone repos into temporary directories using tempfile. Run git commands in isolated subprocesses. Restrict file operations and handle errors gracefully. Ensures safe and reproducible forensic analysis.

## 3. Known Gaps & Plan for Next Phases

### 3.1 Judicial Layer

Three personas: Prosecutor, Defense, Tech Lead.
Analyze rubric criteria in parallel.
Output JudicialOpinion objects with score, reasoning, citations.
Independent prompts to avoid collusion.

### 3.2 Synthesis Engine

ChiefJusticeNode aggregates JudicialOpinions.
Apply synthesis rules: Security Override, Fact Supremacy, Functionality Weight.
Generate final scores, dissent summaries, and actionable remediation plans.

## 4. StateGraph Flow Diagram

Diagram shows Detective layer fan-out, EvidenceAggregator fan-in, parallel Judges, ChiefJustice synthesis, and final report generation.

## 5. Conclusion

Detective layer implemented, typed state, sandboxed repo ingestion, RAG-lite PDF querying. Next steps: Judges, ChiefJusticeNode, final report generation.