Below is a comprehensive blueprint for your **DevOps Ghostwriter** project. This documentation is organized into technical segments to serve as your project's "Source of Truth" during the hackathon.

📄 Project Overview: "DevOps Ghostwriter"

Concept: An autonomous, multi-agent AI system that serves as a virtual DevOps engineer. It monitors GitHub Pull Requests (PRs), performs deep security audits, executes code in a sandbox to verify logic, and automatically drafts documentation—all while providing full observability via Weights & Biases.

🛠️ Technical Stack

- **Frontend:** React.js (Vite), Tailwind CSS, Socket.io-client (for real-time agent logs).
- **Backend (Orchestration):** Node.js, Express.js, MongoDB (Audit history & status tracking).
- **Agent Engine:** Python 3.10+, FastAPI, **Google Agent Development Kit (ADK)**.
- **AI Brain: Gemini 1.5 Pro** (Reasoning/Security) & **Gemini 1.5 Flash** (Tests/Linting).
- **Observability: Weights & Biases (W&B) Weave** (Tracing & Eval).

🗺️ Frontend: Routes & UI Structure

Path	Component	Description
/	Dashboard	Grid view of repositories; displays "Health Scores" for recent PRs.
/repo/:id	RepoDetails	Timeline of all AI-reviewed PRs and security trends over time.
/audit/:prId	AuditCenter	**The Main Demo Page:** Real-time terminal showing agent thoughts and final verdict.

/trace/:traceId	Observability	An embedded **W&B Weave iframe** showing the full reasoning trace.

⚙️ Backend: Logic & API Routes

The backend acts as a bridge between GitHub and your AI Agents.

Node.js Routes (The Bridge)

- POST /api/webhook/github: Entry point for GitHub PR events. Validates signatures and saves PR data to MongoDB.
- GET /api/audits/:prId: Fetches the status and results of a specific AI review.
- GET /api/stats: Aggregates "Bugs Caught" and "Tokens Saved" for the dashboard.

Python FastAPI (The Agent Engine)

- POST /analyze: Triggered by Node.js. Initializes the ADK session.
- GET /health: Checks connectivity to Gemini and W&B.

🤖 AI Agent Workflow (Google ADK)

Using the **Parallel Fan-Out/Gather Pattern**, your agents collaborate as follows:

1. **Orchestrator Agent (The Manager):** Receives the PR diff. It creates a shared session.state and spawns three sub-agents.
2. **Security Auditor (Specialist):** Scans the diff for hardcoded secrets, SQL injection, or vulnerable dependencies.
3. **Runtime Validator (Tool User):**
   - Uses Gemini's native **Code Execution** tool.
   - Writes a temporary test script based on the PR changes.
   - Executes it in the ADK sandbox and reports if it crashes.
4. **Ghostwriter (The Synthesizer):** * Gathers findings from the state.
   - Writes a professional GitHub comment: *"✅ Logic Passed | ⚠️ 1 Security Risk Found | 📝 Updated README."*

📊 W&B Weave Implementation

To win the "Innovative Use of W&B" category:

- **Instrumentation:** Decorate every agent function with @weave.op().

- **Live Tracing:** Capture the exact "Thought" before a tool is called.
- **Evaluations:** Create a dataset of "Known Buggy PRs." Run your agent against them and use W&B to visualize the **Success Rate vs. Latency**.

🚀 Execution Roadmap (Priority Tasks)

1. **P1 (Core):** Connect Node.js to a GitHub Webhook and forward the payload to a simple Python "Echo" agent.
2. **P2 (Intelligence):** Build the ADK hierarchy (Manager + 1 Specialist). Integrate **W&B Weave** for tracing.
3. **P3 (Polish):** Build the React "Live Log" UI using Socket.io and embed the W&B Trace dashboard.

[Build a multi-agent app with ADK and Gemini]

This video is essential because it demonstrates how to orchestrate multiple specialized agents using Google's ADK, which is the core framework for your DevOps Ghostwriter's "intelligence" layer.