Project Sentinel: Developer Implementation Guide

This guide is designed to help your team architect the "Project Sentinel" system. Use this as a reference for how to integrate the disparate components (AI, ML, and Security) into a cohesive Full Stack application.

1. Suggested System Architecture

To succeed in this challenge, maintain a clear "Separation of Concerns."

- **Frontend (React/Vue/Next.js):** Handles the "HUD" (Dashboard) and the "Input" (Report Submission).
- **Backend (Node/Python/Go):** The central nervous system. It receives requests, interacts with the Database, and calls the AI/ML services.
- **Database (PostgreSQL/MongoDB):** Stores the 1,000 historical logs and any new reports processed by your system.
- **Services Layer:** Dedicated scripts or modules for ML (Forecasting) and AI (LLM Prompting).

2. Recommended Implementation Workflow

Phase 1: The "Vault" (Database & API)

1. Initialize your database.
2. Write a script to seed your database using historical_avengers_data.csv.
3. Build a basic GET endpoint to serve this data to your frontend.

Phase 2: The "HUD" (Frontend & Visualization)

1. Connect your frontend to your API.
2. Use a charting library (like Chart.js, Recharts, or D3) to visualize the historical stock levels.
3. Build a "Report Submission" form.

Phase 3: The "Jarvis" Layer (AI & ML Integration)

1. **ML Integration:** Create a function that takes the last 10-20 records of a specific resource and calculates a "Time-to-Zero."
2. **AI Integration:** Set up your LLM API (OpenAI, Gemini, etc.). Create a prompt that accepts the raw_text from the heroes and returns structured JSON.

Phase 4: The "Shield" (Security Middleware)

1. **Crucial:** Create a utility function redactPII(text).

2. Use Regular Expressions (Regex) or String Manipulation to replace names and phone numbers with [REDACTED].
3. **The Flow:** * User clicks "Submit" -> Backend receives data -> **Redaction Function runs** -> Redacted text sent to LLM.

3. Data Integration Map

Data Source	Logic	Output
CSV Data	Linear Regression / Trend Analysis	A "Predicted Exhaustion Date" line on your chart.
JSON Reports	LLM Entity Extraction	A new row in your "Live Intelligence" table.
Hero Metadata	Security Middleware	Redacted logs that prove PII was never leaked to the AI.

4. Technical Strategy Tips

- **Don't Block the UI:** AI and ML processing can be slow. Use loading states or "Pending" indicators in your UI while the backend works.
- **Handle the "Snap":** The historical data contains a "Thanos Snap" event (50% drop). Your ML model should be robust enough to handle this anomaly.
- **Error Handling:** What happens if the LLM returns bad JSON? Write a fallback or a "validation" step in your middleware.

Good luck, teams. Focus on the integration—the magic happens where the pieces connect.