# PhantomBand: AI Cyber Warfare Simulator

**Author:** Ritvik Indupuri

**Date:** September 25, 2025

**Version:** 1.0

---

# Introduction

- **Purpose**

This document provides a definitive technical overview of the PhantomBand application. It is intended for developers, system architects, and technical analysts, offering a deep understanding of the system's architecture, data flow, AI integration, component-level implementation, and operational workflows.

- **Scope**

This documentation covers the application's as-built, mission-ready feature set. This includes its dual capabilities for AI-driven electronic warfare (EW) scenario generation and the analysis of real-world radio frequency (RF) data sets.

---

# System Architecture

---

- **Architectural Overview**

PhantomBand is a client-side **Single Page Application (SPA)** engineered with the React framework. It operates on an **"AI as an Analytical Service"** model, wherein the client communicates directly with the Google Gemini API. This serverless architecture offloads computationally intensive simulation and analysis tasks to the AI model, permitting the frontend to remain lightweight, responsive, and focused on delivering a professional user experience.

## Architectural Principles

**Client-Centric Processing:** All application logic, state management, and data pre-processing are executed entirely within the end-user's browser. This enhances security by ensuring user-provided data is never transmitted to an intermediary server before analysis and minimizes infrastructure dependencies.

- **AI as an Analytical Engine:** The Gemini API is leveraged not as a conversational language model but as a sophisticated, on-demand analytical service. It is tasked with either generating a complete simulation from parameters or interpreting a data summary to construct a grounded, fact-based narrative.

- **Schema-Driven Reliability:** Communication with the AI is strictly governed by a predefined JSON schema. This schema acts as a data contract, ensuring that the AI's response is always a predictable, machine-readable data structure, which is the cornerstone of the application's operational reliability.

---

- **System Components & Lifelines**

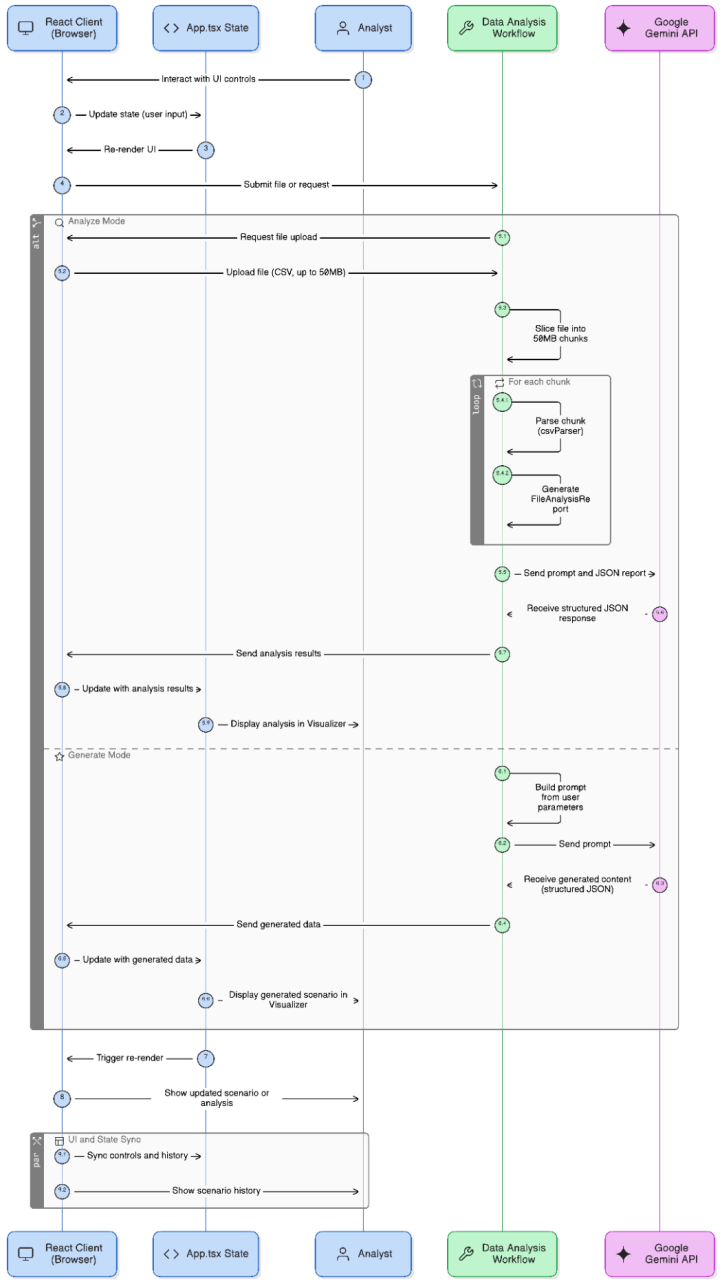
The architecture is composed of five principal components, which function as lifelines in the system sequence diagram:

- **Analyst:** The human operator.
- **React Client (Browser):** The user-facing interface and logic container.
- **App.tsx State:** The central, in-memory state management entity and single source of truth.
- **Data Analysis Workflow:** A logical, client-side service for data ingestion and pre-processing.
- **Google Gemini API:** The external AI analytical engine.

---

# System Interaction Diagram

*Figure 1:* **System Sequence Diagram for PhantomBand Operational Workflows**



The architecture diagram illustrates the temporal sequence of interactions among five core system components, detailing control and data flow. It highlights two primary operational modes:

- **Analyze Mode (top half):**
    - Initiates with Analyst input.
    - Progresses through client-side processing and state management.
    - Extends to external communication with the AI API.
    - Culminates in UI updates for the Analyst.
    - Includes a specific loop fragment that underscores the resilient, iterative methodology for processing substantial file chunks.

- **Generate Mode (Bottom half):**
    - Initiates with the **Analyst** defining a set of scenario **parameters** within the user interface.
    - Progresses by having the client-side workflow construct a structured **prompt** directly from these user-defined parameters.
    - Extends to external communication by transmitting the completed prompt directly to the **Google Gemini API**.
    - Culminates when the AI's structured JSON response is received, integrated into the application state, and displayed as a new scenario in the **visualizer**.
    - Represents a direct, parameter-driven workflow that bypasses the file ingestion and parsing stages entirely.

---

## AI Integration: Digital Adversary & Analyst

The core innovation of PhantomBand is its deployment of the Gemini API as a **non-deterministic analytical engine**. This is achieved through advanced prompt engineering and strict output schema enforcement.

- **Role Priming:** The system instruction configures the AI with the operational persona of "PhantomBand, a specialized AI for advanced RF signal analysis and electronic warfare simulation." This establishes the necessary technical context for its output.

- **Dual-Prompt System:** A buildUserPrompt function dynamically constructs either a **generation prompt** from user parameters or an **analysis prompt** that includes a JSON summary of uploaded data, instructing the AI to function as a Signals Intelligence (SIGINT) analyst.

- **Mandating Actionable Intelligence:** By including classification and countermeasure as mandatory fields in the required JSON schema, the system mandates that the model move beyond simple data reporting to perform analysis, classify anomalies, and propose viable countermeasures.

---

## Core Operational Workflows

- **File Analysis & Pre-Processing Workflow (analyze mode)**

This workflow is engineered for the secure and efficient analysis of large, local data files.

1. **Ingestion:** The Analyst uploads a .csv or .txt file via the FileUpload component.
2. **Large File Handling:** A critical safeguard engages if the file exceeds 50 MB. The **Data Analysis Workflow** partitions the file into manageable chunks using the browser's slice method without loading the entire file into memory.
3. **Client-Side Parsing:** A robust utility parses each chunk, handling multiple delimiters and using a score-based system to identify key data columns. It performs a full statistical analysis and generates a FileAnalysisReport.
4. **AI Tasking:** The compact FileAnalysisReport (a JSON summary) is sent to the Gemini API.
5. **Response and Visualization:** The AI returns a structured JSON analysis, which updates the application state and is rendered in the DataVisualizer for the Analyst.

- **Scenario Generation Workflow (generate mode)**

This workflow creates a synthetic scenario based on user-defined parameters.

1. **Initiation:** The Analyst configures mission parameters in the SimulationControls.
2. **Prompt Construction:** The client assembles these parameters into a structured generation prompt.
3. **AI Tasking & Response:** The prompt is sent to the Gemini API, which returns a complete AnalysisResult JSON object.
4. **State Update & Ingestion:** The application's central state is updated with the AnalysisResult, triggering a full re-render of the dashboard.

---

## Key Component Implementation

---

- **App.tsx**
    - **Role:** The root component and single source of truth for all application state. Manages parameters, analysis results, loading states, history, timesteps, and the primary operational mode.

- **FileUpload.tsx**
    - **Role:** Manages the entire file selection and pre-analysis UI.
    - **Key Logic:** Contains the 50 MB file size check and renders controls for large file segmentation. It is responsible for triggering the analysis callback with either the complete file or a sliced blob.

- **DataVisualizer.tsx**
    - **Role:** The primary data visualization component, built using the Recharts library.
    - **Functionality:** Creates a visual display that shows changes in sound frequencies over time (like a "waterfall" of sound) and an analysis that breaks down sound into its different frequency components. This analysis uses advanced mathematical techniques, including selectable filters (like a "Hamming" filter) and a specialized, efficient algorithm called the Cooley-Tukey Fast Fourier Transform.

- **services/geminiService.ts**
    - **Role:** The exclusive interface for all communication with the Google Gemini API.
    - **Critical Asset:** Contains the responseSchema, the detailed data contract that enforces the structure of the AI's output, and the buildUserPrompt function that enables the application's dual-mode functionality.

---

## Architectural Merits

---

- **Zero-Trust, Serverless Architecture:** Operating entirely on the client reduces the system's attack surface and ensures sensitive data remains within the Analyst's control.

- **Performance and Robustness:** The client-side file chunking mechanism ensures application stability and responsiveness when handling mission-scale data files.

- **Modularity and Decoupling:** The high degree of separation between the UI, state, data processing, and AI components allows for independent maintenance and upgrades with minimal system impact.

- **Data Integrity:** The explicit flow of "structured JSON" to and from the Gemini API guarantees that API responses are always predictable and machine-readable, ensuring operational reliability.

---

## Source Code & Repository

The complete source code for the PhantomBand application is maintained in a private Git repository and is available for review by authorized personnel.

- **Link to GitHub:** https://github.com/ritvikindupuri/PhantomBand

---

## Conclusion

PhantomBand represents a paradigm shift in tactical RF analysis, leveraging a secure, client-centric architecture to transform a large language model into a potent analytical engine. By offloading complex processing to the AI and enforcing rigid data contracts through schema validation, the application delivers reliable, actionable intelligence directly within the analyst's browser. Its sophisticated handling of large datasets, modular design, and inherently secure serverless posture make it a robust and versatile tool for electronic warfare operations, training, and signals intelligence analysis. This innovative approach ensures high performance and data security, establishing PhantomBand as a powerful asset for the modern digital battlespace.