SentinAI-Ops

AI-Powered Anomaly Detection for Critical Infrastructure Defense

Submitted by: Ben Tito - University of Nairobi
Emmanuel Nyabicha - Kenyatta University
Felix Robinson - Kenyatta University
Godfrey Ayuka - Kenyatta University
Shammah Sitati - JKUAT

Team Name: CyberSentinel

Hackathon: AI for National Prosperity

Category: Research Track – AI for Cybersecurity

January 29, 2026

1 Project Overview

Duration	43 Days (6 Weeks)
Team Size	5 Members
Total Tasks	18 Tasks across 4 Phases
Development Model	Parallel Track Architecture
Roadmap Diagrams	https://sentin-gantt.vercel.app/

Core Deliverables

1. Self-hosted Docker Compose deployment
2. Three ML models (Isolation Forest, NLP Logs, Behavioral Profiling)
3. Laravel Orchestrator with Blade + Livewire UI
4. FastAPI AI Microservice for heavy computation
5. Real-time threat visualisations (Livewire)
6. Laravel Sanctum Authentication & RBAC

2 Team Structure and Roles

The team will be divided into two tracks (A and B) to enable parallel development and maximise efficiency. This separation allows both teams to work independently for the majority of the project timeline, converging only at critical integration points.

Track A focuses on backend infrastructure and machine learning models, while Track B concentrates on user interface and experience. This eliminates blocking dependencies, reduces development time by approximately 50%, and ensures that security-critical backend components remain isolated within the Docker internal network.

2.1 Track A: Systems & Intelligence (3 Members)

Primary Responsibilities:

1. FastAPI AI Microservice development
2. Machine Learning model implementation
3. API Contract definition for Laravel consumption

4. Prometheus and Loki configuration
5. LLM context aggregation service
6. Security hardening of internal ML endpoints

Required Skills:

1. Python (FastAPI, scikit-learn, psutil, spaCy)
2. Machine Learning and NLP
3. DevOps (Docker, Prometheus, Loki)
4. System security and Linux administration
5. Database Management

Key Tasks:

- T1.2: Backend API Foundation
- T1.4: Monitoring Stack Configuration
- T2.1: Model 1 - Isolation Forest
- T2.2: Model 2 - NLP Log Analyzer
- T2.3: Model 3 - Behavioral Profiling
- T3.1: LLM Conversational Assistant
- T3.3: Multi-Channel Alerting
- T3.5: Security Hardening

2.2 Track B: Orchestration & Experience (2 Members)

Primary Responsibilities:

- Laravel Application development (Orchestrator)
- Blade + Livewire real-time dashboard components
- Integration of FastAPI microservice endpoints
- User Management & RBAC (Sanctum)
- Alerting System (Laravel Queues)
- Demo video production

Required Skills:

- PHP (Laravel 10/11), Livewire, Alpine.js
- UI/UX (Blade, Tailwind CSS)
- Database Design (MySQL/PostgreSQL)
- System Integration (Guzzle/HTTP Client)

Key Tasks:

- T1.3: Frontend Dashboard Foundation
- T2.4: Dashboard UI Components
- T3.2: Chat Interface & Remediation UI
- T4.3: Demo Environment & Video

Shared Responsibilities (Both Teams)

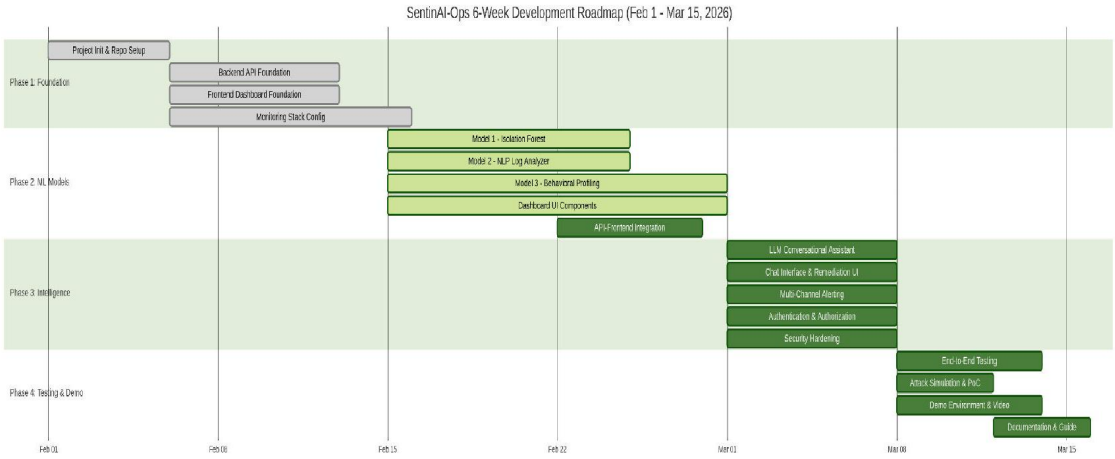
- T1.1: Project Initialization (Day 1)
- T2.5: API-Frontend Integration (Days 6-7)
- T3.4: Authentication & Authorization (Days 10-11)
- T4.1: End-to-End Testing (Days 12-14)
- T4.4: Documentation & Guide (Days 13-14)

3 Technology Stack

Layer	Technology	Purpose
Frontend/UI	Laravel Blade + Tailwind	Server-side rendered, responsive dashboard
Reactivity	Laravel Livewire	Real-time chart updates and live threat feeds
Orchestrator	Laravel (PHP 8.x)	Core logic: Auth, RBAC, Eloquent DB, Alert Queues
ML Engine	FastAPI (Python)	High-performance API for AI models (Isolation Forest, NLP)
Auth System	Laravel Sanctum	Secure session and token-based authentication
Alerting	Laravel Queues	Asynchronous multi-channel notifications (Telegram/SMS)
Metrics	Prometheus	Time-series database with PromQL
Logs	Loki + Promtail	Log aggregation for ML analysis

4 Development Timeline

Gantt Chart Visualization



Roadmap Diagrams: https://sentin-gantt.vercel.app/

Phase 1: Foundation & Infrastructure (Weeks 1-2: Feb 1-14)

Objective: Establish project infrastructure and enable parallel development

Task	Name	Track	Duration	Team	Dependencies	Deliv
T1.1	Project Init	DevOps	5 days	Full Team	None	GitHub repo, docker-compose skele-ton, README
T1.2	FastAPI AI Microser-vice	Backend	7 days	Track A	T1.1	FastAPI struc-ture, ML end-points, Swag-ger docs
T1.3	Laravel Setup (Or-chestrator)	Frontend	7 days	Track B	T1.1	Laravel 11, Livewire, Tail-wind init, Database
T1.4	Monitoring Stack Config	DevOps	10 days	Track A	T1.1	Prometheus, Loki, ex-porters con-fig-ured

Milestones:

- Both teams working independently with clear interfaces
- docker-compose up runs all containers successfully
- Frontend uses mock data, backend connects to Prometheus/Loki

Phase 2: ML Models & Core Features (Weeks 3-4: Feb 15-28)

Objective: Implement all ML models in parallel with dashboard components

Task	Name	Track	Duration	Team	Dependencies	Deliverables
T2.1	Model 1: Isolation Forest	Backend	10 days	Track A	T1.2, T1.4	Trained model, anomaly detection API
T2.2	Model 2: NLP Log Analyzer	Backend	10 days	Track A	T1.2, T1.4	Log parsing engine, regex patterns
T2.3	Model 3: Behavioral Profiling	Backend	14 days	Track A	T1.2	psutil agent, process whitelist
T2.4	Blade/Livewire Components	Frontend	14 days	Track B	T1.3	System vitals, alert timeline, metrics viz
T2.5	Orchestrator-Engine Handshake	Both	6 days	Both	T2.1, T2.4	Laravel consuming FastAPI endpoints

Milestones:

- All three ML models operational with test data
- Dashboard displays real backend data
- End-to-end anomaly detection functional

Phase 3: Intelligence & User Experience (Week 5: Mar 1-7)

Objective: Add AI assistant, authentication, alerting, and security

Task	Name	Track	Duration	Team	Dependencies	Deliverables
T3.1	LLM Conversational Assistant	Backend	7 days	Track A	T2.1, T2.2, T2.3	Flask aggregator, LLM integration
T3.2	Chat Interface	Frontend	7 days	Track B	T2.4, T3.1	AI chat component, remediation display
T3.3	Multi-Channel Alerting (Jobs)	Backend	7 days	Track B	T2.5	Laravel Queues for Telegram, Email
T3.4	Auth & RBAC (Sanctum)	Security	7 days	Track B	T1.3	User roles, protected routes
T3.5	Security Hardening	Security	7 days	Track A	T1.1	Network isolation, secret management

Milestones:

- AI assistant provides contextual threat explanations
- Multi-channel alerts operational
- Secure admin access implemented
- Backend isolated within Docker network

Phase 4: Integration, Testing & Demo (Week 6: Mar 8-15)

Objective: End-to-end testing, attack simulation, demo preparation

Task	Name	Track	Duration	Team	Dependencies	Deliverables
T4.1	End-to-End Testing	QA	6 days	Full Team	T3.1-T3.5	Test scenarios, bug fixes
T4.2	Attack Simulation	Security	4 days	Track A	T2.3	Reverse shell test, detection validation
T4.3	Demo Environment	Demo	6 days	Track B	T4.1	Demo script, video recording
T4.4	Documentatio	Docs	4 days	Full Team	T4.1	Installation guide, architecture diagrams

Milestones:

- Detection-to-alert latency under 5 seconds
- Professional demo video completed
- Complete installation documentation
- Zero critical bugs

5 Development Timeline & Weekly Routine

The project follows a 6-Week Agile Sprint structure. Instead of a rigid daily timetable, the team operates on a cyclical weekly routine that balances deep work with necessary synchronization points.

Sunday: Sprint Planning & Strategy

- Full Team Virtual Meeting (1 Hour): The week begins with a high-level sync to review the backlog and define the specific goals for the upcoming cycle.

- Role Assignment: Tasks are distributed based on the current priorities. Track A (Backend) defines the necessary API contracts and data models, while Track B (Frontend) confirms that the proposed data structures meet the UI visualization requirements.

Monday & Tuesday: Parallel Execution

- Mode of Work: Asynchronous development.
- Track A Focus (Backend): Deep technical implementation, including infrastructure setup, database schema management, and ML model training.
- Track B Focus (Frontend): UI/UX development, creating layout scaffolding, and building visual components using mock data until the backend is ready.

Wednesday: Mid-Week Checkpoints

- Track-Specific Virtual Meetings (**30** Minutes): Each track holds a short, focused call to address specific technical blockers.
- Technical Review: Track A reviews code for endpoint logic and model accuracy. Track B reviews the interface responsiveness and theme consistency.

Thursday & Friday: Deep Work & Integration

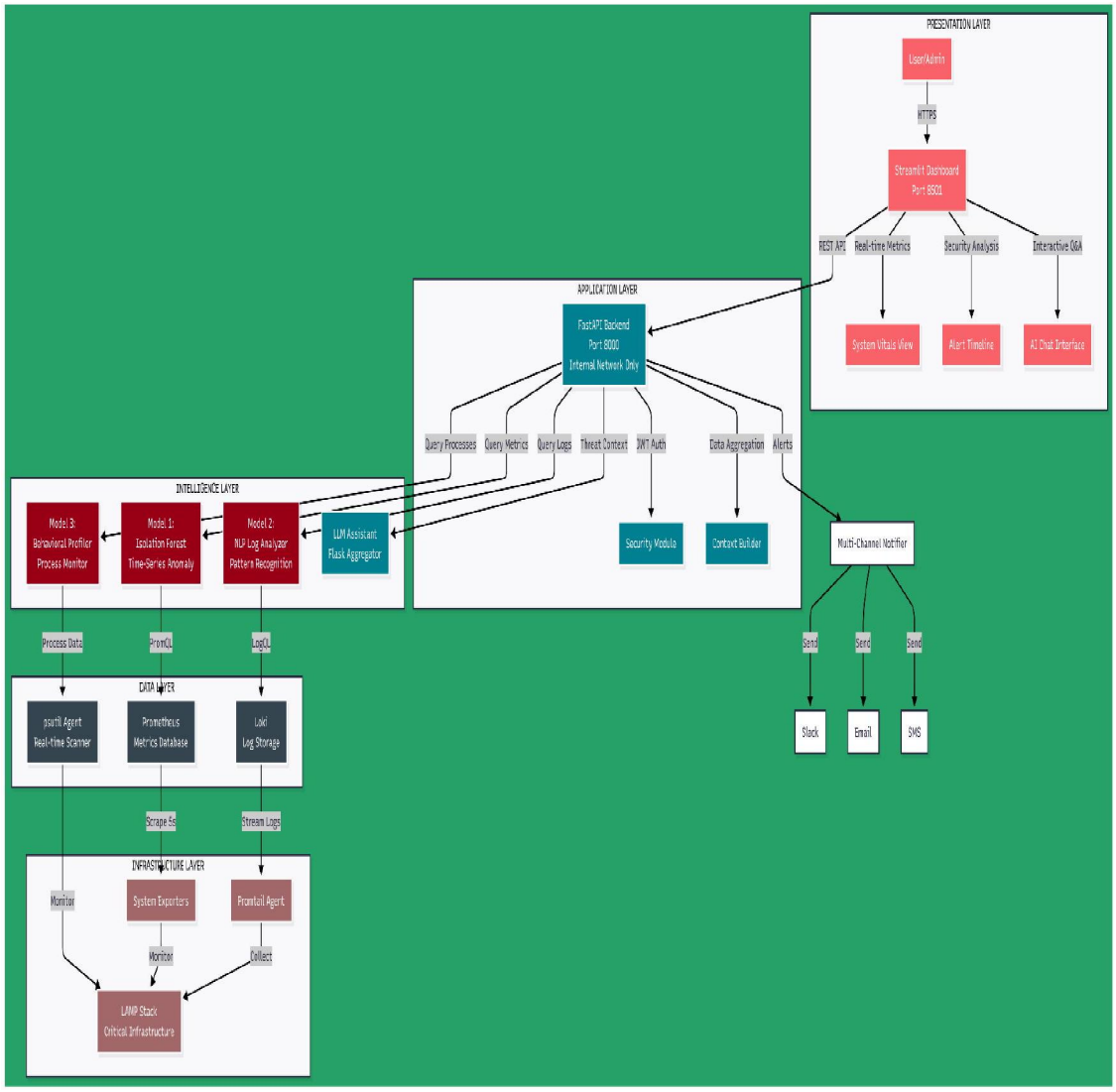
- Heavy Coding: Teams focus on complex logic implementation and connecting the separate components.
- Integration (Friday): Code branches are merged into the development environment. Track A focuses on testing Docker container stability, while Track B performs end-to-end flow testing to ensure the dashboard correctly displays real-time data from the API.

Saturday: Review & Deliverables

- Full Team Physical Meeting (3 Hours): The week concludes with a comprehensive review of the output.
- System Demo: The team walks through the completed features. Track A verifies system security and latency, while Track B verifies the user journey and aesthetics.
- Next Steps: Any incomplete items are moved to the backlog for the following Sunday's planning session.

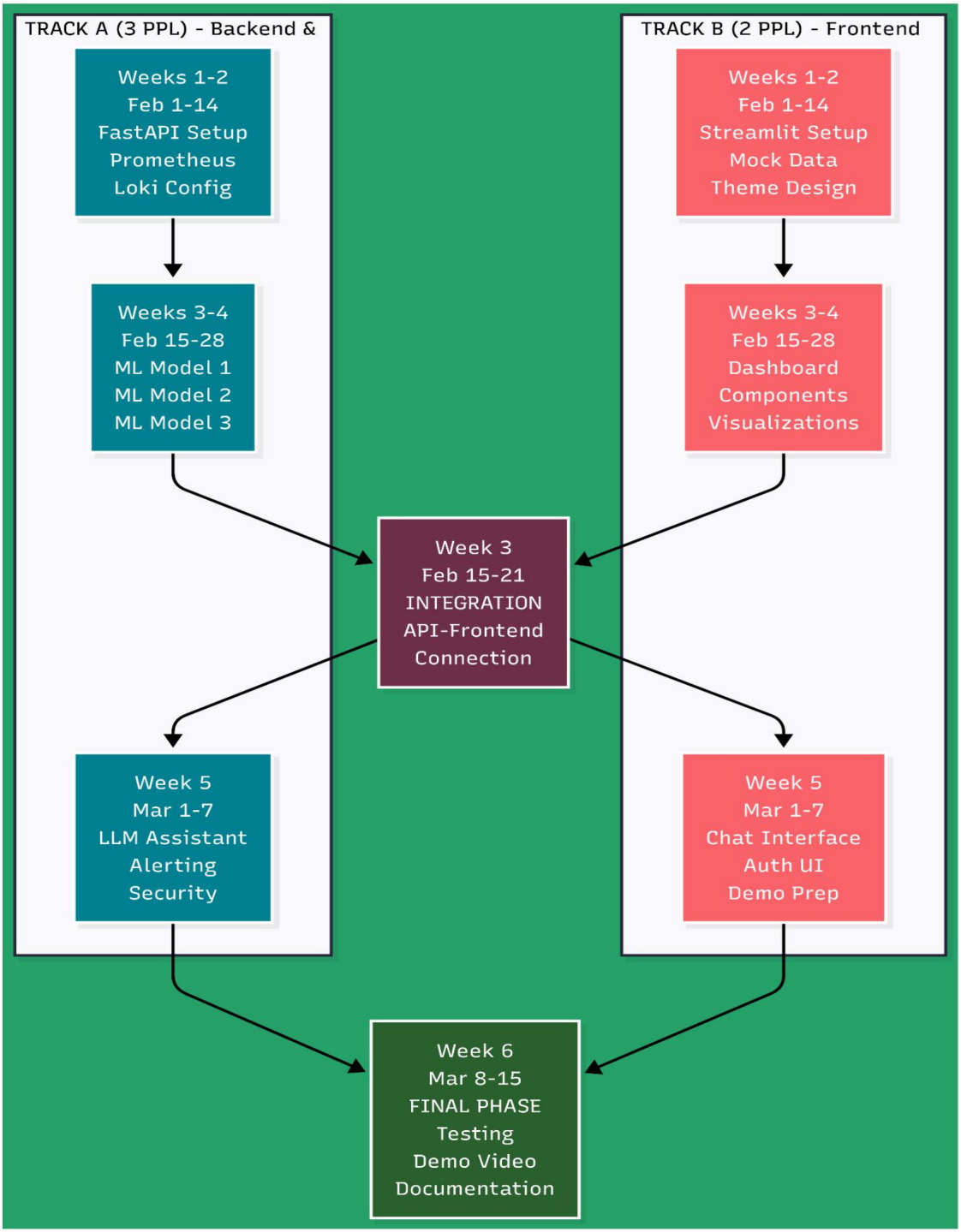
6 Architecture Diagrams

6.1 System Architecture Flowchart



Roadmap Diagrams: https://sentin-gantt.vercel.app/

6.2 Parallel Development Workflow



7 Quick Reference

7.1 Installation Commands

```
# Clone repository
git clone https://github.com/Nuelnyakyz/SentinAI-Ops.git
cd SentinAI-Ops

# Configure environment
cp .env.example .env

# Launch stack
docker-compose up -d

# Access dashboard
open http://localhost:8501
```

7.2 Key Endpoints

- Dashboard: http://localhost
- FastAPI Microservice: http://localhost:8000/docs
- Prometheus: http://localhost:9090
- Loki: http://localhost:3100

Hackathon: AI for National Prosperity 2026