Technical Report: Threat Monitoring & Alert Management System

1. Project Objective

This project implements a high-performance backend API system for a simplified Threat Monitoring & Alert Management Platform. Designed for "Cyethack Solutions", the platform specializes in ingesting security events from diverse sources (surveillance systems, SIEM tools, etc.) and providing an intelligent, role-based management layer for security analysts.

2. Technical Architecture & Tech Stack

The system is built on a robust, scalable foundation:

- Core Framework: Python 3.11 with Django 5.2.
- API Engine: Django REST Framework (DRF) following RESTful best practices.
- Database: PostgreSQL (Production) with optimized schema and indexed lookups.
- Deployment: containerized via Docker for seamless environment parity.
- Gateways: Gunicorn + WhiteNoise for high-concurrency production handling.

3. Comprehensive Feature Implementation

3.1 Advanced Authentication & RBAC

The system implements secure JWT-based authentication, fulfilling both standard and bonus requirements.

- Admin: Full system access (Create, Read, Update, Delete).
- Analyst: Focused read-only access to alerts and events, ensuring zero unauthorized mutations.

3.2 Intelligent Threat/Event Ingestion

A high-throughput API endpoint (`POST /events/ingest/`) handles incoming data streams.

- Data Capture: source name, event type, severity (Low to Critical), and descriptions.
- Geo-Enrichment: The system automatically resolves the source IP address into City and Country coordinates, providing critical intelligence on attack origins.

3.3 Automated Alert Generation & Management

The system features an autonomous alert engine using Django Signals.

- Automation Logic: Any event with High or Critical severity is instantly converted into an 'Open' Alert.
- Lifecycle Management: Analysts can track alerts through Open, Acknowledged, and Resolved statuses.
- Filtering: Native support for filtering by severity and status via query parameters.

3.4 Governance & Auditing

A dedicated User Activity Log tracks every critical action (auth, registration, status shifts) with IP-level tracking and User-Agent fingerprints, ensuring full accountability.

---

4. Security & Performance (Compliance Checklist)

- Rate Limiting: Implemented global throttling (1000/hr per user) to prevent brute-force and DDoS attempts.
- Input Validation: Strict serializer-level validation protects against SQL injection and malformed payloads.
- Query Optimization: Leveraged `selectrelated` and `prefetchrelated` to eliminate N+1 database problems.
- Clean Code: Adhered to strict separation of concerns (Serializers for logic, Views for control, Models for data).

5. Deployment & Containerization

The system is fully containerized using Docker and Docker Compose, ensuring seamless local development and production-ready deployments. The production environment is hosted on Render.com utilizing Gunicorn and WhiteNoise for static file management and process handling.

Developer: Vinayagam

GitHub: https://github.com/Zyrvix/threat-monitor-system

Live Site: https://threat-monitor-system.onrender.com/