Project Title: Secure, Scalable AI Model Training Platform

Objective

To design and build a secure, containerized infrastructure capable of training reinforcement learning (RL) models with sensitive or proprietary data, implementing zero-trust security, observability, and cryptographic protection throughout the stack – from data ingestion to inference.

This project aims to showcase practical skills in secure architecture, AI infrastructure, DevSecOps practices, observability, containerization, and automation – aligning with real-world systems used at OpenAI, Tesla, Amazon, Nvidia, and SpaceX.

Scope Of Work

* Model Training Module
  + Build an RL agent (e.g. PPO or DQN) to learn from a simulated environment
* API Gateway + Auth
  + Flask/FastAPI backend secured by Oauth2/JWT; exposed via HTTPS
* Containerized Infrastructure
  + Use Docker to isolate the application, model, and supporting services
* Zero-Trust Architecture
  + Enforce least privilege access between microservices; simulate user roles
* Key Management System
  + Encrypt sensitive model files using AES and manage keys securely
* Logging and Observability
  + Use tools like Prometheus, Grafana, ELK for telemetry and security visibility
* CI/CD Pipeline
  + Automate tests, security checks, and deployments using GitHub Actions or similar

Business Requirements

FR1. The platform shall allow users to upload datasets for model training securely

FR2. The system shall encrypt model outputs and sensitive data at rest/in transit

FR3. All services must authenticate using OAuth2 tokens and enforce role-based access control

FR4. The training module shall support at least one RL algorithm implemented from scratch or using **stable-baselines3**

FR5. Logs must be collected from all containers and services, centralized, and visualized.

FR6. Alerts must be triggered on anomalous activity or unexpected system behavior

NFR1. System must support concurrent sessions with isolated container environment

NFR2. System shall be resilient to container crashes and support auto-recovery

NFR3. Latency of Training API must be 200 ms on average under normal load

NFR4. System must be modular and support plug-and-play reinforcement learning (RL) environments

NFR5. Infrastructure must be fully Dockerized and runnable on both local and cloud systems

Assumptions and Constraints

A1. I have root access to my deployment environment

A2. Open-source tools (e.g. Python, Docker, etc.) will be used

A3. CI/CD is implemented using GitHub Actions

C1. Development will occur on limited hardware (e.g. personal laptop, free-tier cloud, etc.)

C2. GPU acceleration or simulated depending on availability

C3. No access to proprietary cloud services unless opted for personally

Acceptance Criteria

AC1. The system successfully trains an RL model in a secure, containerized environment

AC2. All logs, metrics, and alerts are visible and working through the observability stack

AC3. All authentication and authorization flows are tested and validated

AC4. CI/CD successfully builds, tests, scans, and deploys containers to the chosen platform

AC5. Documentation is provided, and the project can be demoed end-to-end