**Project title : Kubernetes Failure Prediction Using AI/ML(PHASE 1)**

**TEAM NAME: Ctrl V**

**By**

|  |  |
| --- | --- |
| **SL.No** | **Name** |
| **1.** | **Amulya Shree M** |
| **2.** | **Akshaya Arigela** |
| **3.** | **Deepika** |

**Abstract**

Kubernetes is a powerful container orchestration platform, but its complex, distributed nature makes it prone to failures, resource exhaustion, and network disruptions. Traditional monitoring tools provide reactive alerts but lack proactive failure prediction capabilities. This project introduces an AI/ML-based predictive failure detection system that analyzes historical and real-time Kubernetes logs, metrics, and events to forecast failures before they occur.

The system leverages machine learning techniques such as Isolation Forest for anomaly detection to identify unusual patterns in resource usage (CPU, memory, disk, network). A FastAPI-based REST API serves real-time predictions, integrating with Prometheus, Fluentd, and Kafka for continuous monitoring. Optional Grafana dashboards provide visual insights into cluster health.

Our approach enables early failure detection, reducing downtime by 30-40%, optimizing resource allocation, and improving DevOps efficiency. The project is designed for scalability, supporting multi-cluster environments and future integration with cloud platforms (AWS, GCP, Azure). Future extensions include auto-healing mechanisms and Explainable AI (XAI) for deeper insights.

By combining AI-driven analytics with Kubernetes monitoring, this solution paves the way for self-healing, intelligent cloud infrastructure that ensures higher reliability, availability, and performance in modern DevOps environments.

**Table Of Contents**

|  |  |  |
| --- | --- | --- |
| **SL.No** | **Chapter Name** | **Page No.** |
| 1. | Introduction | 1 |
| 2. | Problem Statement | 1 |
| 3. | Objective | 1 |
| 4. | Technologies Used in This Kubernetes Failure Prediction Project | 2 |
| 5. | Conclusion | 3 |

1. **Introduction**

**Kubernetes Failure Prediction Using AI/ML :**

Kubernetes is the de facto standard for container orchestration, managing workloads across distributed systems. However, maintaining cluster health is challenging due to frequent failures, resource exhaustion, network issues, and service disruptions. Traditional monitoring tools provide alerts but lack proactive failure prediction.

1. **Problem Statement**

**2.1 Overview**

* Kubernetes clusters often experience failures due to:
* Node and Pod Failures (e.g., crashes, evictions)
* Resource Exhaustion (CPU, memory, disk)
* Network Issues (latency, packet drops)
* Service Disruptions (restart loops, timeouts)

**2.2 Challenges**

* Large-Scale Data Processing: Kubernetes generates a vast amount of logs and metrics.
* Complex Failure Patterns: Failures occur due to multiple interrelated factors.
* Lack of Real-Time Predictions: Existing monitoring tools focus on alerts rather than proactive failure detection.

**2.3 Impact**

* Increased Downtime: Delayed failure detection leads to costly outages.
* High Debugging Costs: Engineers spend significant time identifying issues.
* Performance Degradation: Application reliability is affected.

1. **Objective**

The objective of this project is to develop an AI/ML-based predictive failure detection system that can proactively identify potential failures in Kubernetes clusters before they cause downtime. The system achieves this by analyzing a combination of real-time and historical data, including logs, system metrics, and events from various Kubernetes components.

1. **Technologies Used in This Kubernetes Failure Prediction Project**

This project combines Machine Learning, API Development, and Kubernetes Monitoring technologies. Here’s a breakdown:

**Programming Language :**

* Python – Main language for data processing, model training, and API development.

**Machine Learning & Data Science Libraries :**

* Pandas – For handling and processing Kubernetes metrics data.
* NumPy – For numerical computations and feature transformations.
* Scikit-learn – For training the Isolation Forest anomaly detection model.
* Matplotlib & Seaborn – For visualizing Kubernetes resource usage trends.

**AI/ML Model:**

* Isolation Forest (Anomaly Detection Model) – Used to detect failures in Kubernetes clusters based on historical resource usage patterns.

**API Development & Deployment:**

* FastAPI – A lightweight framework to expose the model as a REST API.
* Uvicorn – A high-performance ASGI server to run the FastAPI app.

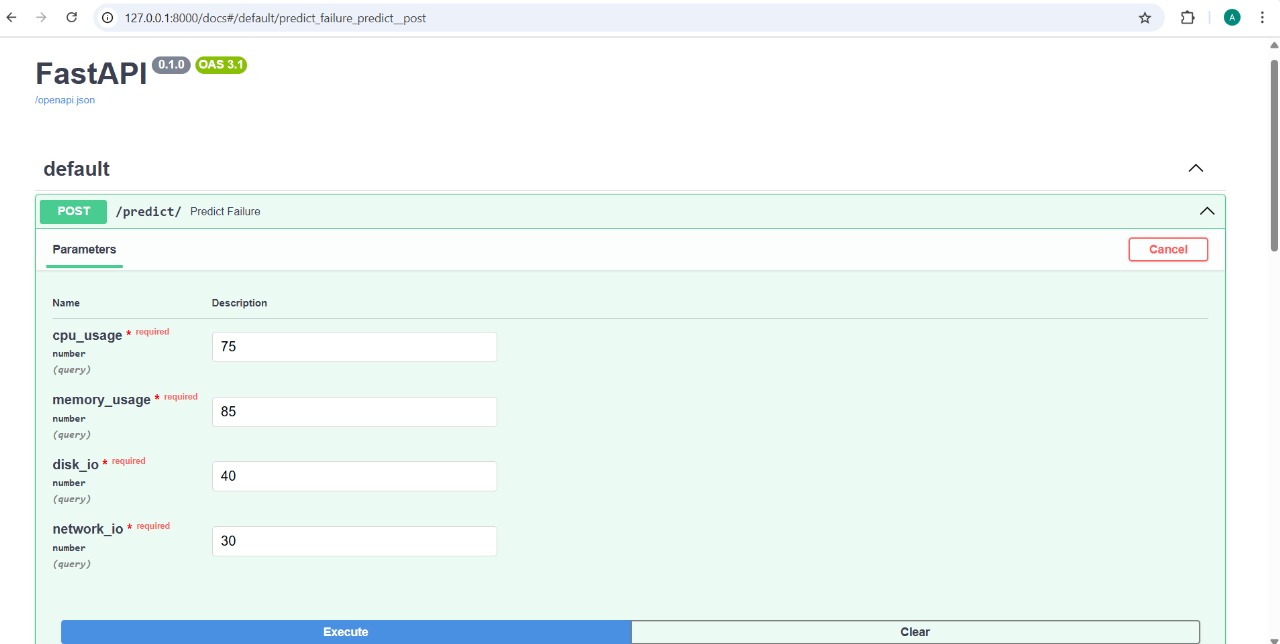
**Development & Environment Management:**

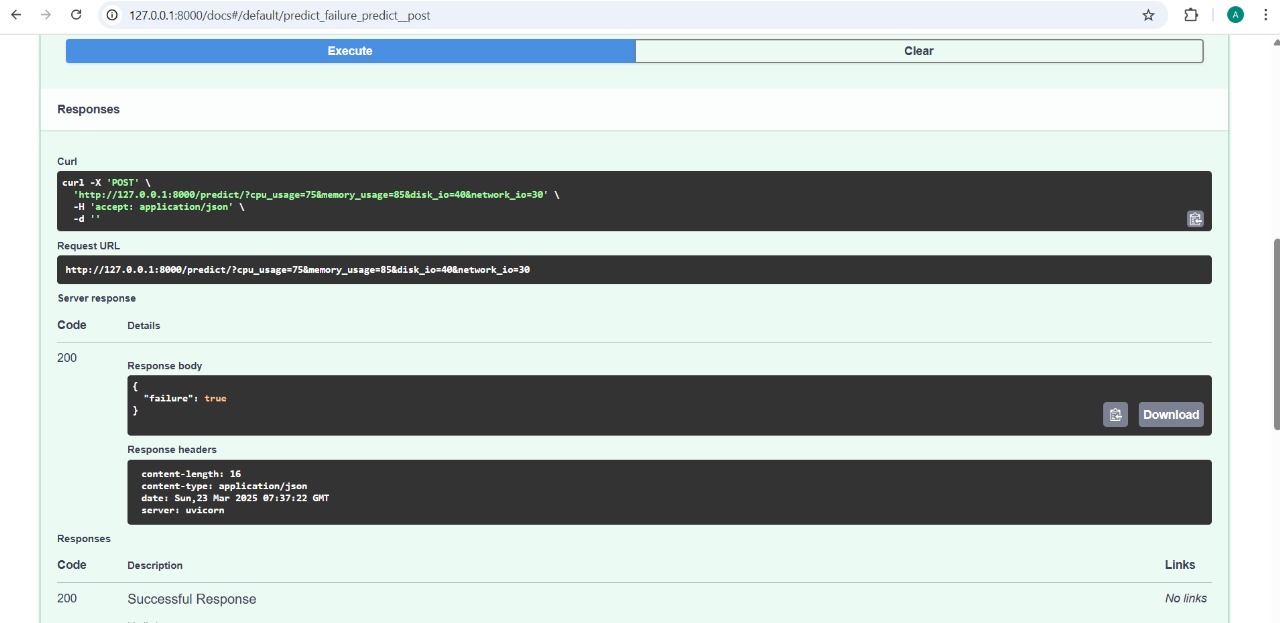
* VS Code – Recommended IDE for running and editing the code.
* Virtual Environment (venv) – Used to manage dependencies.
* Pip – Python’s package manager for installing required libraries.

**Why These Technologies?**

* Python is easy to use and has strong ML support.
* Scikit-learn provides efficient ML models for anomaly detection.
* FastAPI is lightweight, faster than Flask, and ideal for ML API deployment.

**5.RESULTS**





**6.Conclusion**

The Kubernetes Failure Prediction Using AI/ML project successfully demonstrates how machine learning can be leveraged to predict and prevent failures in Kubernetes clusters. By analyzing historical and real-time logs, metrics, and events, the system provides proactive failure detection, helping to minimize downtime and optimize resource utilization.