🚀 Auto-Healing Monitoring System

Prometheus | Alertmanager | Ansible | Grafana | Flask Webhook

---

✅ 1. Introduction

In modern DevOps environments, ensuring high availability and system resilience is critical. This project implements an **automated self-healing monitoring solution** designed to detect service failures (such as NGINX or MySQL outages) and recover them automatically without manual intervention.

✅ 2. Abstract

This project builds a robust monitoring and automation system by integrating **Prometheus for metrics collection, Alertmanager for alerting, and Ansible for automated remediation**, complemented by **Grafana for visualization**.
On detecting a failure condition (e.g., service down), Prometheus triggers an alert → Alertmanager forwards it to a **Flask Webhook** → The webhook invokes Ansible playbooks to restart the failed service, restoring normal operations automatically.

✅ 3. Tools Used

- **Prometheus** – Metrics collection and alerting
- **Alertmanager** – Manages and forwards alerts
- **Grafana** – Visualization of system metrics and alerts
- **Node Exporter / MySQL Exporter / Blackbox Exporter** – Collect server and service health metrics
- **Flask** – Webhook application to trigger Ansible

- **Ansible** – Automated service remediation playbooks
- **Ubuntu Linux** – Operating system environment
- **Nginx & MySQL** – Monitored critical services

✅ 4. Steps Involved in Building the Project

1️⃣ Set Up Monitoring with Prometheus

- Configured `prometheus.yml` to scrape metrics from Node Exporter, Blackbox Exporter, and MySQL Exporter.
- Created custom alert rules to detect NGINX/MySQL downtime.

2️⃣ Configure Alertmanager & Webhook Integration

- Alertmanager configured to send alerts to `http://127.0.0.1:5001/alert`.
- Flask Webhook receives alerts and triggers the appropriate Ansible playbook based on alert type.

3️⃣ Build Self-Healing Mechanism with Ansible

- Created separate playbooks:
    - `nginx_autoheal.yml` → Restart NGINX
    - `mysql_autoheal.yml` → Restart MySQL
- Webhook executes these playbooks automatically when corresponding alerts are fired.

4️⃣ Visualize Metrics & Alerts in Grafana

- Integrated Prometheus as a data source in Grafana.
- Imported dashboards for Node Exporter, MySQL metrics, and Blackbox HTTP probes.

- Enabled real-time visualization of system health and healing events.

5️⃣ Test Self-Healing Workflow

- Simulated service failures by manually stopping NGINX or MySQL.
- Verified Prometheus detected failure → Alertmanager forwarded alert → Webhook triggered Ansible → Service recovered automatically.

✅ 5. Conclusion

This project demonstrates an effective self-healing monitoring solution that drastically reduces downtime and manual intervention in system administration.
It improves system **reliability, availability, and resilience**, while providing clear visualization and automation.
The modular architecture allows easy future expansion to additional services (Docker, Redis, etc.).