High-Level Design (HLD)

Cloud Migration and Monitoring Project

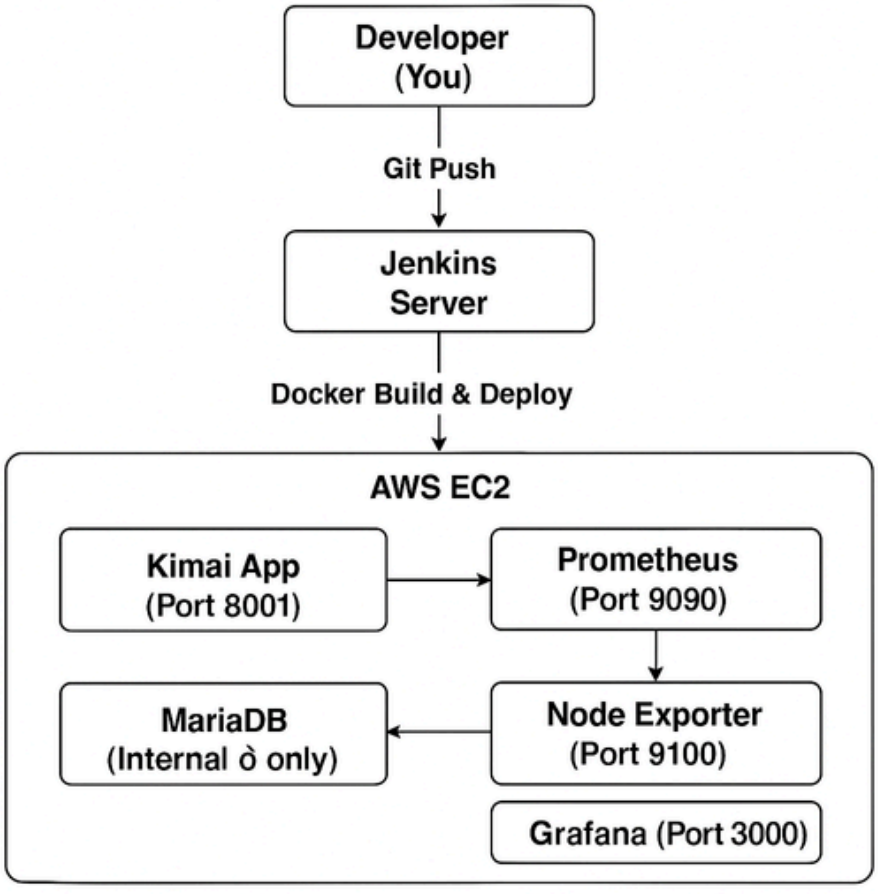
Project Overview

This project involves migrating the open-source Kimai time-tracking application to the AWS cloud infrastructure using Infrastructure as Code (Terraform), Docker-based deployment, Jenkins CI/CD automation, and cloud monitoring with Prometheus and Grafana. Security measures and scalability were also incorporated.

Objectives

- Host Kimai app on a secure and scalable AWS infrastructure.
- Automate provisioning using Terraform.
- Deploy using Docker containers.
- Enable CI/CD pipeline using Jenkins.
- Integrate monitoring and alerting using Prometheus and Grafana.

Architecture Diagram



Components

1. AWS Infrastructure

- EC2 instance with Amazon Linux 2023.
- Security Groups (Inbound: 22, 8001, 9090, 3000, 9100 from Bastion or Developer IP).
- IAM Role for EC2 (CloudWatch access).

2. Terraform (IAC)

- Used to create EC2, IAM, Security Groups, Key Pair.
- Used main.tf and variables.tf for modular configuration.

3. Docker Deployment

- Docker Compose runs Kimai and MariaDB.
- Monitoring stack runs Prometheus, Grafana, and Node Exporter.

4. CI/CD with Jenkins

- Jenkins job pulls GitHub repo.
- Jenkins triggers Docker Compose deployment.
- Optional: Jenkins runs inside Docker.

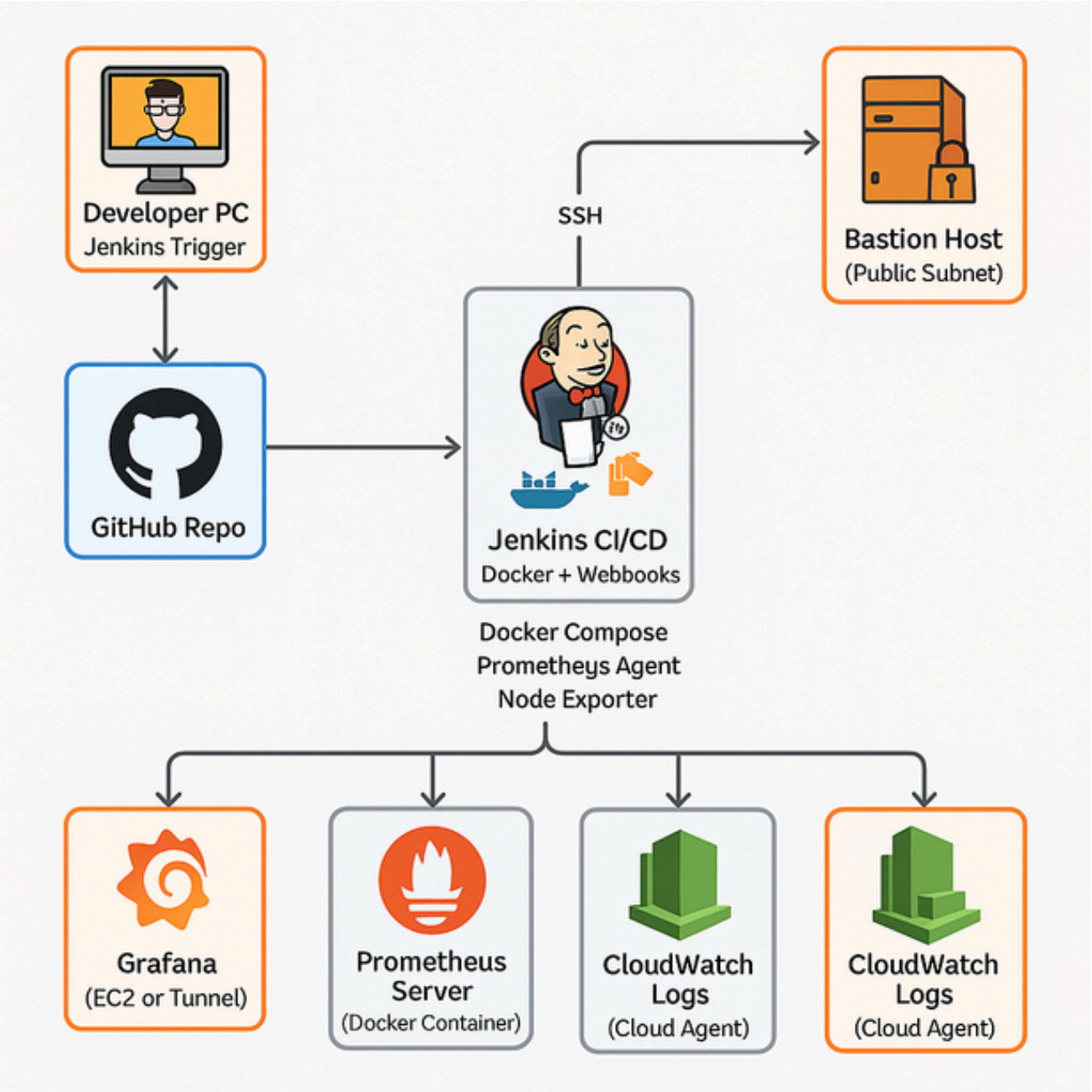
5. Monitoring

- Prometheus scrapes metrics.
- Node Exporter provides system metrics.
- Grafana visualizes metrics.
- Alerts configured for CPU usage.

Key Ports Used

Component	Port
Kimai	8001
Jenkins	8080
Prometheus	9090
Grafana	3000
Node Exporter	9100
SSH	22

Visual Representation of Architecture Diagram



Deployment Flow

1. **Terraform** provisions infrastructure.
2. **Jenkins** pulls code and builds Docker containers.
3. **Docker Compose** deploys:
   - Kimai + MariaDB
   - Prometheus + Node Exporter
   - Grafana
4. **Monitoring** via Prometheus scraping + Grafana dashboard.
5. **Alerts** raised for CPU usage or system failures.

Security Considerations

- SSH restricted to Bastion or your IP.
- IAM Role with minimum permissions.
- Internal traffic between services (MariaDB).
- Docker networks isolated.

Scalability and Maintenance

- Stateless components (Kimai, Prometheus) can be horizontally scaled.
- Docker Compose simplifies service restart/recovery.
- Logs streamed to CloudWatch.

Tools & Technologies

- **Cloud**: AWS EC2, IAM, Security Groups
- **IaC**: Terraform
- **Containerization**: Docker, Docker Compose
- **CI/CD**: Jenkins
- **Monitoring**: Prometheus, Grafana, Node Exporter

Outcome

The architecture ensures:

- Reliable and repeatable deployments.
- Real-time monitoring of application and infrastructure health.
- CI/CD automation.
- Easy-to-maintain MNC-style structure for long-term use.

Ankitha J

BE CSE (AI&ML)

Jerusalem college of Engineering