Assignment 4 - CI/CD Pipeline with Jenkins on Kubernetes

Course: DevOps - Container Orchestration
Assignment: Jenkins CI/CD Pipeline Integration
Date: November 29, 2025

Team Members

- **Nikhil Shankar Chirakkal Sivasankaran** - 9026254
- **Richard Andrey Biscazzi** - 8903530

Overview

This assignment demonstrates a complete CI/CD pipeline using Jenkins deployed on Kubernetes, integrated with both GitHub and self-hosted Gitea for automated build triggers.

Architecture

Infrastructure Setup

- **Jenkins Server:** Deployed on Kubernetes cluster
- **Version Control:** GitHub + Self-hosted Gitea (from Assignment 3)
- **CI/CD Runner:** Jenkins on K8s
- **Trigger Method:** Webhooks (GitHub & Gitea)

Repositories

Repository	Purpose	Branch	URL
Jenkins Infrastructure	Jenkins deployment on K8s	`assignment4`	container-asssignment3
Python Test App	Sample Flask application with CI/CD	`master`	PythonTest-CICD-Assignment4

Deployment URLs

- **Jenkins Server:** http://jenkins.devsecmindset.dev/
- **Gitea Instance:** Running locally from Assignment 3

Implementation Steps

1. Jenkins Setup on Kubernetes

Richard deployed Jenkins on the Kubernetes cluster using configurations from the `container-asssignment3` repository (branch: `assignment4`).


Initial Pipeline Setup

2. Python Application Repository

Created a Flask-based Python application with the following structure:

- `app.py` - Simple Flask web application
- `requirements.txt` - Python dependencies
- `Jenkinsfile` - Pipeline configuration

3. GitHub Integration

Webhook Configuration:

- URL: `https://jenkins.devsecmindset.dev/github-webhook/`
- Trigger: Push events
- Content-Type: `application/json`


GitHub Webhook Setup

Pipeline Trigger Test:


GitHub Push Trigger

4. Gitea Integration

Repository Clone:

Cloned the GitHub repository to self-hosted Gitea instance.


Gitea Repository

Generic Webhook Configuration:

Installed Generic Webhook Trigger Plugin in Jenkins and configured Gitea webhook.


Gitea Webhook Setup

Code Change Test:

Modified `app.py` in Gitea to test pipeline trigger.


Gitea Code Change

Successful Pipeline Execution:


Gitea Trigger Success

Jenkinsfile Pipeline

The pipeline consists of three stages:

```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                echo 'Code checked out from repository'
                sh 'ls -la'
            }
        }

        stage('Validate') {
            steps {
                echo 'Validating project structure...'
                sh 'test -f app.py && test -f requirements.txt'
            }
        }

        stage('Deploy') {
            steps {
                echo 'Pipeline completed successfully!'
            }
        }
    }
}
```

Key Features Implemented

- ☑ Jenkins deployed on Kubernetes cluster
- ☑ Automated GitHub webhook integration
- ☑ Self-hosted Gitea integration using Generic Webhook Plugin
- ☑ Multi-source repository support (GitHub + Gitea)
- ☑ Automated pipeline triggers on code push
- ☑ Successful validation and build processes

Testing Results

Test Case	Source	Result	Screenshot
Manual Trigger	Jenkins UI	☑ Success	Screenshot 1
GitHub Push	GitHub Webhook	☑ Success	Screenshot 4
Gitea Push	Gitea Webhook	☑ Success	Screenshot 7

Collaboration

- **Richard:** Set up Jenkins infrastructure on Kubernetes, configured cluster access
- **Nikhil:** Created Python application, configured webhooks (GitHub & Gitea), tested pipeline triggers

---

Conclusion

Successfully implemented a complete CI/CD pipeline using Jenkins on Kubernetes with dual repository integration (GitHub and self-hosted Gitea). The pipeline automatically triggers on code changes from both sources, demonstrating enterprise-grade DevOps practices.

Assignment Deliverables

- [x] Jenkins running on Kubernetes cluster
- [x] GitHub repository with Jenkinsfile
- [x] Gitea repository integration
- [x] Automated webhook triggers from both sources
- [x] Documentation with screenshots