PHASE 2 DESIGN

PROJECT TITLE :- Setting up a CI/CD pipeline for automated deployment

COLLEGE NAME:-Dr. SMCE

GROUP MEMBERS:-

1. Amith C Y [USN:- 1CC21CS004]
2. Charan K M [USN:- 1CC21CS017]
3. Nitin Teja N[USN:-1CC21CS040]
4. Sunil C M [USN:-1CC21CS057]

---

1. Plan of the Project

This project aims to create a **containerized Python application (llama.py)** that is tested, built, and deployed automatically using a **CI/CD pipeline**. The application is deployed to a Kubernetes cluster, and infrastructure provisioning is handled via **Terraform**.

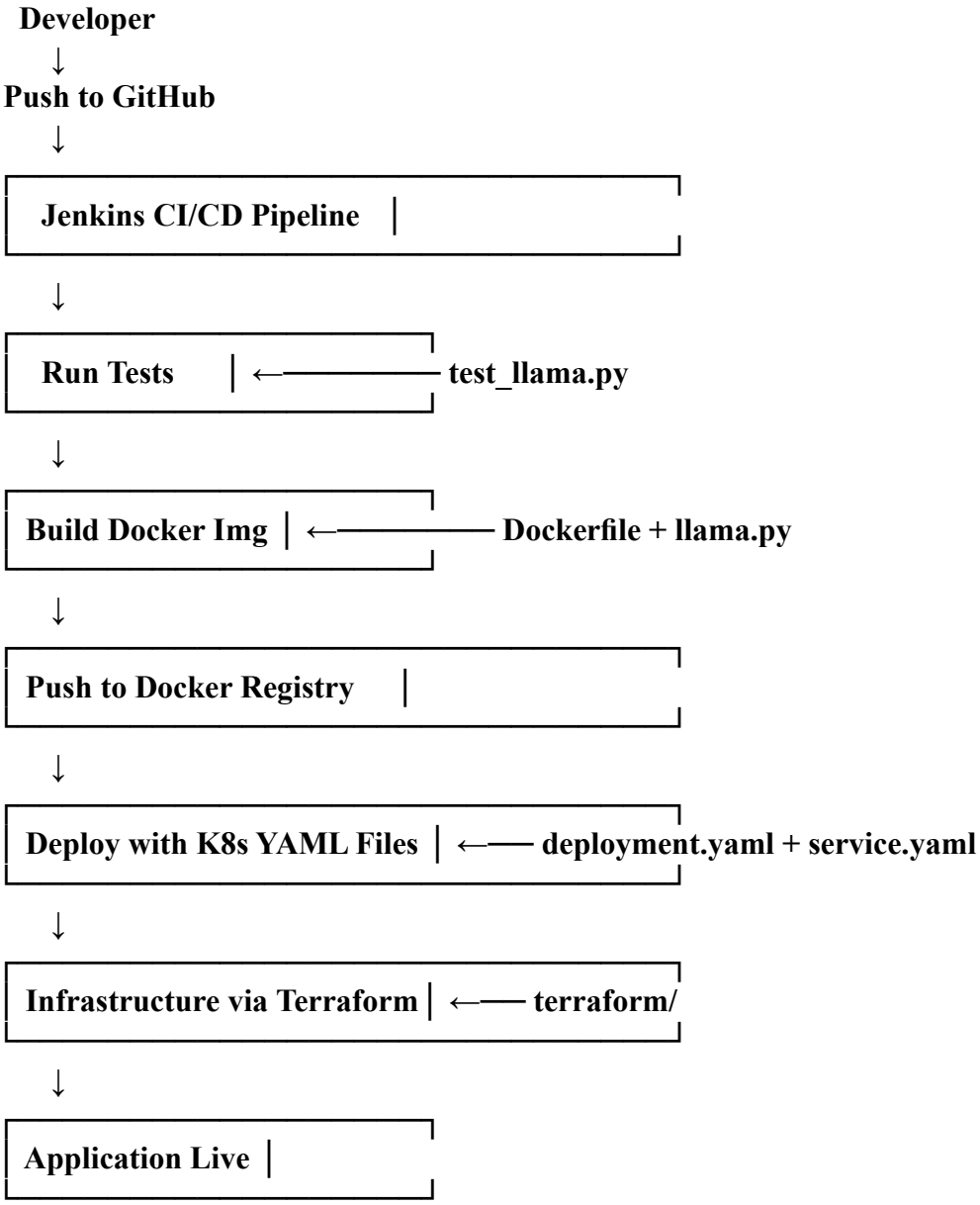
Key Goals:

- Containerize the app using Docker.
- Automate the build, test, and deploy process using Jenkins.
- Manage infrastructure with Terraform.
- Deploy the app using Kubernetes manifests.
- Ensure reliability and repeatability using version control and automation.

2. Blueprint of the Project

Component	File/Tool	Purpose
Source Code	llama.py	Core Python application
Dependency Mgmt	requirements.txt	Python dependencies for the app
Unit Testing	test_llama.py	Automated test cases
Containerization	Dockerfile	Create Docker image of the application
CI/CD Orchestration	Jenkinsfile	Jenkins pipeline for build/test/deploy
Infra as Code (IaC)	terraform/*.tf	Provision infrastructure (e.g., Kubernetes, cloud resources)
Kubernetes Deployment	deployment.yaml	Deploy container to a Kubernetes cluster
Kubernetes Service	service.yaml	Expose the application inside/outside the cluster

3. Flow Diagram of the CI/CD Plan



4. Why the Services Are Used and What is the Aim

Tool/Service	Purpose	Aim
Dockerfile	Containerizes the app for consistency across environments	Create an isolated, portable runtime
Jenkinsfile	Automates testing, building, and deployment processes	Enable repeatable, fast, and error-free deployments
deployment.yaml	Describes how the app should run inside Kubernetes	Automate app deployment in a scalable infrastructure
service.yaml	Exposes the application within the cluster or externally	Provide access to the app
llama.py	Core Python logic for the application	Business logic of the app
requirements.txt	Lists dependencies for building the environment	Reproducibility and dependency control
test_llama.py	Validates functionality before deployment	Ensure code reliability
Terraform	Provisions infrastructure (cloud, Kubernetes clusters, etc.)	Automate infra provisioning; reduce manual setup and drift

5. Step-by-Step Process of Execution of the Project

Step-by-Step Process of Execution of the Project

Step 1: Clone Repository and Configure Jenkins

- Set up a Jenkins instance (locally or on a server).
- Create a Jenkins job and connect it to the GitHub repository.
- Install required Jenkins plugins (Docker, Git, Pipeline, Kubernetes, Terraform, etc.).

Step 2: Dockerize the Application

- Create a Dockerfile:
- Build and test the image locally

Step 3: Create Jenkinsfile for CI/CD Pipeline

- Open-source CI server installed and managed manually.

Step 4: Infrastructure as Code with Terraform

- Purpose: Infrastructure as Code (IaC).
- Use: Automate infrastructure provisioning (e.g., setting up cloud servers).

Step 5: Kubernetes Deployment

- Purpose: Container orchestration.
- Use: Deploy and manage multiple containers at scale

Step 6: Validate & Monitor

- Verify the application is accessible via the load balancer.
- Use kubectl get all to check the status of pods, services.
- Integrate monitoring tools like Prometheus and Grafana if needed.