**Jenkins-Docker Project Documentation**

1. Introduction

The Jenkins-Docker project demonstrates how to integrate Jenkins with Docker for automated deployment. It includes a simple Flask application containerized using Docker and managed with a Jenkins CI/CD pipeline.

2. Project Structure

The project consists of the following files:

* app.py – The main Python application (Flask).
* Dockerfile – Instructions to build the Docker image.
* docker-compose.yml – Defines services for Docker Compose.
* Jenkinsfile – The Jenkins CI/CD pipeline script.
* requirements.txt – Lists dependencies for the Flask app.
* README.md – Project documentation.

3. Setup & Installation

Prerequisites

Ensure the following tools are installed:

* Docker
* Docker Compose
* Jenkins
* Git

Cloning the Repository

git clone <repository-url>

cd Jenkins-Docker

Running the Application Locally

pip install -r requirements.txt

python app.py

By default, the app runs on http://localhost:5000.

4. Dockerization

Dockerfile (Build the Image)

# Use an official Python runtime as a parent image

FROM python:3.9

# Set the working directory

WORKDIR /app

# Copy files to the container

COPY . /app

# Install dependencies

RUN pip install flask

# Expose port 5000

EXPOSE 5000

# Run the Flask application

CMD ["python", "app.py"]

Building and Running the Docker Container

docker build -t flask-app .

docker run -p 5000:5000 flask-app

5. Docker Compose Setup

docker-compose.yml

version: '3.8'

services:

web:

build: .

ports:

- "5000:5000"

volumes:

- .:/app

restart: always

Running with Docker Compose

docker-compose up -d

6. CI/CD Pipeline with Jenkins

Jenkinsfile

pipeline {

agent any

environment {

DOCKER\_IMAGE = "viratpk18/docker-app:latest"

CONTAINER\_NAME = "docker-running-app"

REGISTRY\_CREDENTIALS = "docker\_praveen"

}

stages {

stage('Checkout Code') {

steps {

withCredentials([usernamePassword(credentialsId: 'git\_viratpk18', usernameVariable: 'GIT\_USER', passwordVariable: 'GIT\_TOKEN')]) {

sh 'git clone https://github.com/viratpk18/Jenkins-Docker.git'

}

}

}

stage('Build Docker Image') {

steps {

sh 'docker build -t $DOCKER\_IMAGE .'

}

}

stage('Push to Docker Hub') {

steps {

withDockerRegistry([credentialsId: 'docker\_praveen', url: '']) {

sh 'docker push $DOCKER\_IMAGE'

}

}

}

stage('Deploy Container') {

steps {

sh 'docker run -d --name $CONTAINER\_NAME -p 5000:5000 $DOCKER\_IMAGE'

}

}

}

}

Setting Up Jenkins Pipeline

1. Open Jenkins and create a new Pipeline Project.
2. Configure Source Code Management (GitHub Repository).
3. Add Docker Hub Credentials in Jenkins.
4. Define the pipeline using the Jenkinsfile.
5. Build and deploy automatically with each commit.

7. Testing the Deployment

Checking Running Containers

docker ps

Stopping and Removing the Container

docker stop docker-running-app

docker rm docker-running-app

Accessing the Application

After deployment, open:

http://localhost:5000

8. Conclusion

This project demonstrates: ✅ Dockerizing a Flask app ✅ Automating CI/CD with Jenkins ✅ Deploying using Docker Compose

Future Improvements

* Add a database service (PostgreSQL/MySQL).
* Implement unit tests in the pipeline.
* Deploy to AWS/GCP.