# Automated Docker Image Building with Jenkins

## Technical Implementation Guide

### Prerequisites

* AWS Account
* GitHub Account
* Docker Hub Account
* Basic understanding of Docker and Jenkins

### 1. AWS Infrastructure Setup

1. Login to AWS Console
2. Create EC2 Instance:
   * Name: [Your instance name]
   * AMI: Ubuntu
   * Use default key pair
   * Configure Security Group:
     + Allow HTTP (80)
     + Open ports: 8080, 8081, 3000, 5000, 50000, 1234
     + Source: 0.0.0.0/0

### 2. Jenkins Installation and Configuration

1. Connect to EC2 instance
2. Install Docker:

sudo apt update && sudo apt install docker.io -y

sudo systemctl start docker

sudo systemctl enable docker

sudo usermod -aG docker $USER

newgrp docker

1. Set up Jenkins with Docker:

mkdir jenkins-docker

cd jenkins-docker

1. Create Dockerfile for Jenkins:

FROM jenkins/jenkins:lts

USER root

RUN apt-get update && \

apt-get install -y \

python3 \

python3-pip \

docker.io \

&& rm -rf /var/lib/apt/lists/\*

RUN python3 --version && \

pip3 --version

USER jenkins

1. Build and Run Jenkins Container:

docker build -t jenkins-docker .

docker volume create jenkins\_home

docker run -d \

--name jenkins \

--user root \

--restart=unless-stopped \

-p 8080:8080 \

-p 50000:50000 \

-v jenkins\_home:/var/jenkins\_home \

-v /var/run/docker.sock:/var/run/docker.sock \

jenkins-docker

1. Access Jenkins:
   * Get initial admin password:

docker exec -it jenkins cat /var/jenkins\_home/secrets/initialAdminPassword

* + Access Jenkins UI: http://<your-ec2-public-ip>:8080
  + Complete initial setup and install recommended plugins

### 3. GitHub Repository Setup

1. Create a new repository
2. Configure webhook:
   * URL: http://<your-ec2-public-ip>:8080/github-webhook/
   * Content type: application/json
   * Enable webhook

### 4. Docker Hub Configuration

1. Sign in to Docker Hub
2. Create a repository with the same name used in pipeline script
3. Note down credentials for Jenkins configuration

### 5. Jenkins Pipeline Configuration

1. Install required plugins:
   * Navigate to "Manage Jenkins" → "Plugins"
   * Install "Docker Pipeline" plugin
2. Configure Docker Hub credentials:
   * Go to "Manage Jenkins" → "Credentials"
   * Add new credentials
   * Kind: Docker Registry
   * ID: dockerhub
   * Description: Docker Hub Password
   * Enter Docker Hub username and password
3. Create Pipeline:
   * Click "New Item"
   * Name: test-pipeline
   * Select "Pipeline"
   * Configure pipeline script (see below)

### 6. Pipeline Script

pipeline {

agent any

triggers {

githubPush() // Trigger pipeline on GitHub push events

}

stages {

stage('Clone Repository') {

steps {

git url: 'https://github.com/MANOVIGNESH-S/sample-app.git', branch: 'main'

}

}

stage('Build and Push Docker Image') {

steps {

script {

withDockerRegistry(credentialsId: 'dockerhub') { // Docker Hub credentials ID

sh '''

# Build the Docker image

docker build -t manovignesh7575/sample-app .

# Push the Docker image to Docker Hub

docker push manovignesh7575/sample-app

'''

}

}

}

}

stage('Build') {

steps {

script {

withDockerRegistry(credentialsId: 'dockerhub') {

sh '''

# Build the Docker image with a different tag

docker build -t manovignesh7575/myrepo:latest .

# Push the Docker image to Docker Hub

docker push manovignesh7575/myrepo

'''

}

}

}

}

stage('Run Docker Container') {

steps {

script {

sh '''

# Stop and remove the existing container if it exists

if [ $(docker ps -aq -f name=sample-app) ]; then

docker stop sample-app

docker rm sample-app

fi

# Run the new container

docker run -d --name sample-app -p 8081:8080 manovignesh7575/sample-app

'''

}

}

}

stage('Deploy') {

steps {

script {

sh '''

# Stop and remove the existing container if it exists

if [ $(docker ps -aq -f name=cont-1) ]; then

docker stop cont-1

docker rm cont-1

fi

# Run the new container

docker run -d --name cont-1 -p 1234:80 manovignesh7575/myrepo

'''

}

}

}

}

post {

always {

echo "Pipeline execution completed!"

}

failure {

echo "Pipeline failed. Check logs for details."

}

}

}

### 7.Verification

1. Build the pipeline
2. Monitor build stages in Jenkins
3. Verify image in Docker Hub
4. Check application: http://<your-ec2-public-ip>:8081

### Notes

* Replace placeholder values (URLs, usernames) with your actual values
* Ensure proper security group settings in AWS
* Monitor Jenkins logs for troubleshooting
* Regular maintenance of Docker images and containers is recommended