**Jenkins Pipeline Deployment and Troubleshooting Documentation**

**Introduction**

**This document outlines the process of deploying a Java application using Jenkins, Docker, and AWS, along with detailed troubleshooting steps encountered during the deployment.**

**Pipeline Overview**

**The deployment pipeline consists of the following steps:**

1. **Clone Repository - Fetches the latest code from GitHub.**
2. **Build Docker Image - Creates a Docker image of the application.**
3. **Login to AWS ECR - Authenticates with AWS Elastic Container Registry.**
4. **Push Image to ECR - Uploads the built image to ECR.**
5. **Deploy to EC2 - Pulls the image from ECR and runs it on an AWS EC2 instance.**

**Jenkins Pipeline Code**

**pipeline {**

**agent any**

**environment {**

**AWS\_ACCOUNT\_ID = '302263074599'**

**AWS\_REGION = 'ap-south-1'**

**IMAGE\_NAME = 'spring-boot-app'**

**ECR\_REPO = 'myapp/1'**

**// Fetch AWS credentials from Jenkins**

**AWS\_ACCESS\_KEY\_ID = credentials('AWS\_ACCESS\_KEY\_ID')**

**AWS\_SECRET\_ACCESS\_KEY = credentials('AWS\_SECRET\_ACCESS\_KEY')**

**}**

**stages {**

**stage('Clone Repository') {**

**steps {**

**git branch: 'main', url: 'https://github.com/fardeenkhan6295/devpro1.git'**

**}**

**}**

**stage('Build Docker Image') {**

**steps {**

**sh 'docker build -t $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_REGION.amazonaws.com/$ECR\_REPO:$BUILD\_NUMBER .'**

**}**

**}**

**stage('Login to AWS ECR') {**

**steps {**

**sh '''**

**aws configure set aws\_access\_key\_id $AWS\_ACCESS\_KEY\_ID**

**aws configure set aws\_secret\_access\_key $AWS\_SECRET\_ACCESS\_KEY**

**aws configure set region $AWS\_REGION**

**aws ecr get-login-password --region $AWS\_REGION | \**

**docker login --username AWS --password-stdin $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_REGION.amazonaws.com**

**'''**

**}**

**}**

**stage('Push Image to ECR') {**

**steps {**

**sh 'docker push $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_REGION.amazonaws.com/$ECR\_REPO:$BUILD\_NUMBER'**

**}**

**}**

**stage('Deploy to EC2') {**

**steps {**

**sshagent(['ec2-ssh-key']) {**

**sh '''**

**ssh -o StrictHostKeyChecking=no ubuntu@43.204.236.28 <<EOF**

**docker pull $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_REGION.amazonaws.com/$ECR\_REPO:$BUILD\_NUMBER**

**docker stop myapp || true**

**docker rm myapp || true**

**docker run -d --name myapp -p 80:80 $AWS\_ACCOUNT\_ID.dkr.ecr.$AWS\_REGION.amazonaws.com/$ECR\_REPO:$BUILD\_NUMBER**

**'''**

**}**

**}**

**}**

**}**

**}**

**Troubleshooting Steps**

**During the deployment process, multiple issues were encountered and resolved. Below is a chronological list of problems and their solutions:**

**Issue 1: Git Not Recognized in Jenkins**

**Error Message:**

**Selected Git installation does not exist. Using Default**

**The recommended git tool is: NONE**

**Solution:**

* **Installed Git on the Jenkins server using:**

**sudo apt update && sudo apt install git -y**

* **Configured Git in Jenkins under Manage Jenkins > Global Tool Configuration.**

**Issue 2: AWS Credentials Not Set Correctly**

**Error Message:**

**aws: command not found**

**Solution:**

* **Installed AWS CLI on the Jenkins server:**

**sudo apt install awscli -y**

* **Ensured AWS credentials were properly configured in Jenkins credentials manager.**

**Issue 3: Docker Build Failing**

**Error Message:**

**COPY failed: file not found in build context or excluded by .dockerignore**

**Solution:**

* **Checked that all required files were present in the repository.**
* **Verified the .dockerignore file to ensure essential files were not excluded.**

**Issue 4: Docker Login to ECR Failing**

**Error Message:**

**Cannot perform an interactive login from a non TTY device**

**Solution:**

* **Used --password-stdin method for logging into AWS ECR:**

**aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 302263074599.dkr.ecr.ap-south-1.amazonaws.com**

**Issue 5: EC2 Deployment Failing**

**Error Message:**

**ssh: connect to host 43.204.236.28 port 22: Connection refused**

**Solution:**

* **Verified the security group settings in AWS to allow SSH access.**
* **Ensured the SSH key was correctly added to Jenkins.**

**Issue 6: Container Not Running After Deployment**

**Error Message:**

**unknown directive "deamon" in command line**

**Solution:**

* **Found a typo in the Dockerfile where daemon was incorrectly spelled as deamon.**
* **Corrected it and rebuilt the Docker image.**

**Issue 7: Nginx Configuration Issue**

**Error Message:**

**nginx: [emerg] host not found in upstream**

**Solution:**

* **Ensured the correct server\_name was used in the Nginx configuration.**
* **Restarted the Nginx container:**

**docker restart myapp**

**Final Successful Deployment**

**After resolving the above issues, the deployment pipeline ran successfully, and the application was up and running on the EC2 instance.**

**Conclusion**

**This document provides a complete record of setting up a CI/CD pipeline using Jenkins, AWS, and Docker, along with troubleshooting steps. By referring to this, future deployments can be done more efficiently, and issues can be resolved quickly.**