**Automated web Application Deployment:**

**Project Overview:**

**Objective:** Automate the deployment of a web application using Jenkins, Docker, and AWS. The pipeline will include stages for building, testing, and deploying the application.

**Tools & Technologies:**

* Version Control: Git
* CI/CD Tool: Jenkins
* Containerization: Docker
* Cloud Provider: AWS (Amazon Web Services)
* Web Application Framework: Node.js, Python Flask, or similar

**Step-by-Step Process:**

**Step-1:** **Set Up Version Control with Git**

1. **Create a Git Repository**
2. Sign Up/Log In:
3. Create a New Repository: web-app-deployment
4. Clone the Repository Locally:

Use Bash : git clone <https://github.com/your-username/web-app-deployment.git>

1. **Configure Git**
2. Set User Name and Email:

git config --global user.name "Your Name"

git config --global user.email [your.email@example.com](mailto:your.email@example.com)

1. Create a .gitignore File:

node\_modules/

.env

\*.log

1. Check Status and Add Files:

git status – command for git status

git add . – command for git add file

1. Commit Changes:

git commit -m "Initial commit with web application setup"

1. Push Changes:

git push origin main

**Step-2: Develop the Web Application**

1. **Choose a framework –** node.js or python flask
2. **Create Application Files**
3. Initialize Node.js Application:

npm init -y

1. Install Express:

npm install express

1. **Create Basic Application:** “app.js”- file name

const express = require('express');

const app = express();

const port = 3000;

app.get('/', (req, res) => {

res.send('Hello, World!');

});

app.listen(port, () => {

console.log(`App listening at http://localhost:${port}`);

});

1. **Add Unit Tests:**

test('Basic test', () => {

expect(1 + 1).toBe(2);

});

1. **Test Locally:**

Run application- node app.js

Execute tests- npm test

**Step-3:** **Containerize the Application with Docker**

1. **Install Docker**- from docker website
2. **Create a Dockerfile**

# Use an official Node.js runtime as a parent image

FROM node:14

# Set the working directory in the container

WORKDIR /usr/src/app

# Copy application dependency manifests to the container

COPY package\*.json ./

# Install application dependencies

RUN npm install

# Copy the application source code to the container

COPY . .

# Expose the application port

EXPOSE 3000

# Start the application

CMD ["node", "app.js"]

1. **Build and Test the Docker Image**
2. **Build the Docker Image:**

docker build -t myapp:latest . – use this command for build docker image

1. **Run a Docker Container:**

docker run -p 3000:3000 myapp:latest – use this command for run docker image

**Step-4:** **Set Up Jenkins for CI/CD**

1. **Install Jenkins**
2. **Launch EC2 Instance:**
3. **Install Jenkins:**

sudo yum update -y

sudo yum install java-1.8.0-openjdk-devel -y

wget -O /etc/yum.repos.d/jenkins.repo \

https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

sudo yum install jenkins -y

sudo systemctl start jenkins

sudo systemctl enable Jenkins

1. Access Jenkins:

Access Jenkins through your browser using “http://<EC2-Public-IP>:8080”.

1. Unlock Jenkins:

Retrieve the initial admin password using:

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

1. **Configure Jenkins**
2. **Install Plugins:**
3. **Create a New Pipeline Job:**
4. **Create a Jenkinsfile**
5. Create a “Jenkinsfile”:

pipeline {

agent any

stages {

stage('Build') {

steps {

script {

// Build Docker image

sh 'docker build -t myapp:latest .'

}

}

}

stage('Test') {

steps {

script {

// Run tests

sh 'docker run myapp:latest npm test'

}

}

}

stage('Deploy') {

steps {

script {

// Deploy to AWS ECS or EC2

sh 'docker tag myapp:latest <AWS\_ACCOUNT\_ID>.dkr.ecr.<REGION>.amazonaws.com/myapp:latest'

sh 'docker push <AWS\_ACCOUNT\_ID>.dkr.ecr.<REGION>.amazonaws.com/myapp:latest'

}

}

}

}

}

1. Configure Jenkins Job
2. Set Up the Pipeline Job:

Step-5: **Deploy to AWS**

1. **Set Up AWS Environment**
2. **Create AWS Account:**
3. **Configure AWS CLI:**

Install and configure the AWS CLI on your local machine:

aws configure

1. **Create an ECR Repository**
2. **Create an ECS Cluster**
3. Create an ECS Cluster:
4. Define a Task Definition:
5. Configure ECS Service:

4 . **Update Jenkins for Deployment**

1. Push Docker Image to ECR:
2. Deploy to ECS:

5 . Verify Deployment

1. Access Application:
2. Monitor with CloudWatch: