**Assignment:**

Create a CICD pipeline in Jenkins or AWS CodePipeline or AWS CodeBuild / CodeDeploy or GitHub Actions to fetch code (any tech stack / code e.g Python, Java, DotNet, Node etc) from GitHub repository, Build it, dockerize / containerize and deploy to ECS / EKS or AWS Lambda, S3 so that users can access the application viz URL.

---------------------------------------------------------------------------------------------------------------------------------------

**Technology Used:**

Docker, ECR, EKS, GitHub Actions

---------------------------------------------------------------------------------------------------------------------------------------

**Folder structure:**

**|\_** src

**|\_** .git

**|\_**workflows

**|\_** deploy.yml

**|\_** Dockerfile

**|\_** pom.xml

|\_ k8s/

**|\_** deployment.yaml

**|\_** service.yaml

---------------------------------------------------------------------------------------------------------------------------------------

**Java Project**

* Install JDK & verify

$ java -version

* Install Apache Maven, Add path in environment variable (Environment variable > Path > C:\Program Files\apache-maven-3.9.9\bin) & verify

$ mvn -version

* Install Intel J
* Spring Initializr

Browse – <https://start.spring.io/>

Project – Maven

Language – Java

Spring Boot – 3.5.3

Group – com.example

Artifact – java-hello-world

Name – java-hello-world

Description – Demo project for Spring Boot

Package name – com.example.java-hello-world

Packaging – Jar

Java – 17

Dependencies – Spring Web

Generate

* Extract downloaded zip file and drag into Intel J
* Verify project java version in Intel J

Settings > Project structure > Project > SDK – 17 > Apply > OK

* Navigate into source folder "src/main/java/com.example.java\_hello\_world".
* Run " JavaHelloWorldApplication.java" & verify

Browse – http://localhost:8080/ > It will show "Whitelabel Error Page"

* Create "MyClass.java" file under "src/main/java/ com.example.java\_hello\_world".

package com.example.java\_hello\_world;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class MyClass {

@GetMapping("/")

public String sayHello(){

return "Hello World";

}

}

* Run "JavaHelloWorldApplication.java" file & verify

Browse – http://localhost:8080/ > "It will show Hello World

---------------------------------------------------------------------------------------------------------------------------------------

**Setup**

* Create new repository in GitHub.
* Git Configuration

1. Configure Git user info in local

$ git config --global user.name "shivam"

$ git config --global user.email [shivamthakur0567@gmail.com](mailto:shivamthakur0567@gmail.com)

1. Initialize Git Repository

$ git init

1. Add files to Git:

$ git add .

1. Commit changes:

$ git commit -m "Initial commit"

1. Check current local branch

$ git branch

Master

1. Rename local branch to main

$ git branch -m main

1. Add our remote repository

$ git remote add origin https://github.com/shivam-th/java-hello-world.git

1. Push the code from local to remote repository

$ git push -u origin main --force

* Create AWS User

IAM > Users > Create user > User-name – awscli-user > Next

Permission options – Attach policies directly > Permission Policies – AdministratorAccess > Next > Create user

* Create Access keys

IAM > Users > awscli-user > Create access key

Use case – Command Line Interface (CLI) > Confirmation – Check > Next > Create access key > Download .csv file > Done

* Store Secrets in GitHub

Store AWS secrets in GitHub secrets

Repository Settings > Secrets and variables > Actions > New repository secret

Name – AWS\_ACCESS\_KEY > Secret – <Enter Access key> > Add

Name – AWS\_SECRET\_ACCESS\_KEY > Secret – <Enter Access key> > Add

* AWS Configuration

$ aws configure

AWS Access Key ID [None]:

AWS Secret Access Key [None]:

Default region name [None]:

Default output format [None]:

* Install EKS

1. Browse – <https://eksctl.io/installation/> > download required zip file
2. Create "eksctl" folder in C drive, extract the downloaded zip file and place .exe file in a "C:\eksctl".
3. Add folder path to Environment Variable
4. Edit Environment Variables > Path
5. Verify

$ eksctl version

* Create EKScluster

$ eksctl create cluster \

--name my-cluster \

--region us-east-1 \

--nodegroup-name linux-nodes \

--node-type t3.medium \

--nodes 2 \

--nodes-min 1 \

--nodes-max 3 \

--managed

* Create ECR

$ aws ecr create-repository \

    --repository-name my-java-app \

    --region us-east-1