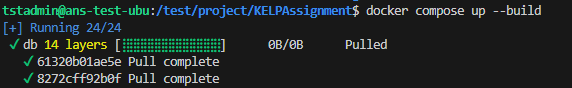
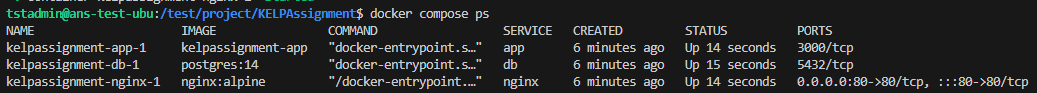
KELP Assignment  
  
  
Part1 - Installation  
Step1 : Create an EC2 instance  
Step2 : Install Docker  
Step3: Install Docker Compose  
  
Part2 – Containerize the Application  
Step1: Files used for the containerzing the Application

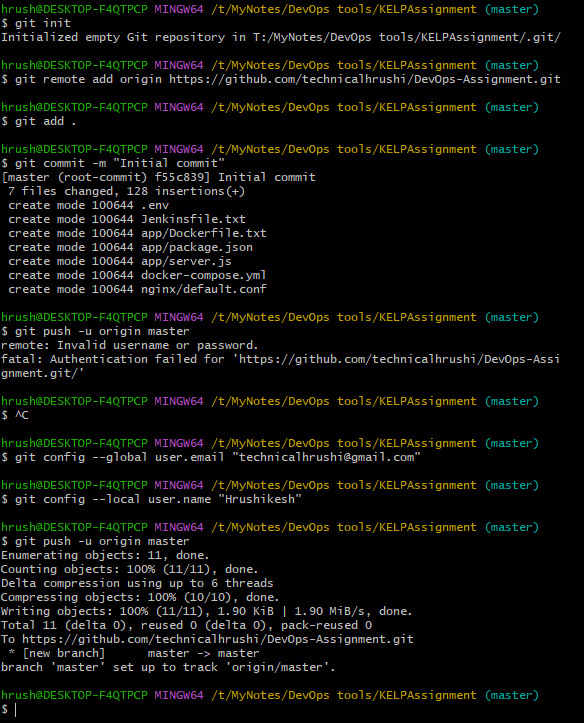
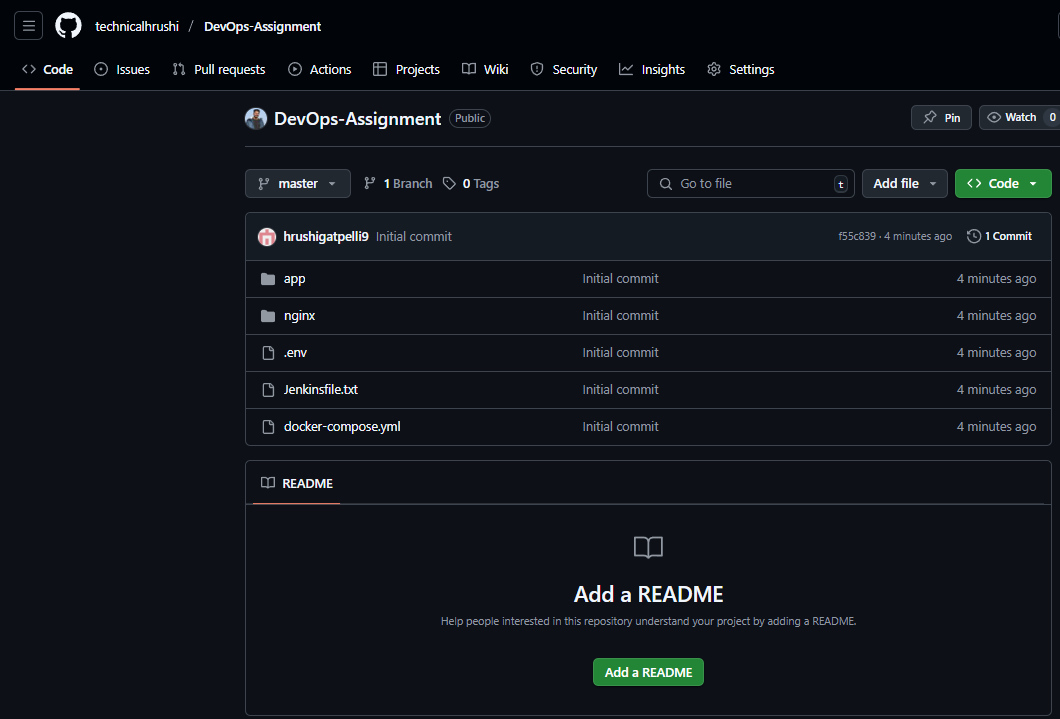
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | --------------------------------------------------------server.js------------------------------------------------------------ const express = require('express');  const { Pool } = require('pg');  const app = express();  const port = 3000;  const pool = new Pool({  user: process.env.POSTGRES\_USER,  host: 'db',  database: process.env.POSTGRES\_DB,  password: process.env.POSTGRES\_PASSWORD,  port: 5432,  });  app.get('/', async (req, res) => {  const result = await pool.query('SELECT NOW()');  res.send(`Hello from Node.js! DB time: ${result.rows[0].now}`);  });  app.listen(port, () => {  console.log(`App running on http://localhost:${port}`);  }); |      |  | | --- | | --------------------------------------------------------Dockerfile--------------------------------------------------------- FROM node:18  WORKDIR /usr/src/app  COPY package\*.json ./  RUN npm install  COPY . .  EXPOSE 3000  CMD ["node", "server.js"] |      |  | | --- | | ------------------------------------------------------package.json-------------------------------------------------------- {  "name": "simple-node-app",  "version": "1.0.0",  "main": "server.js",  "scripts": {  "start": "node server.js"  },  "dependencies": {  "express": "^4.18.2",  "pg": "^8.10.0"  }  } |      |  | | --- | | -----------------------------------------------------default.conf -------------------------------------------------------- server {  listen 80;  location / {  proxy\_pass http://app:3000;  }  } |      |  | | --- | | ----------------------------------------------------docker-compose.yml----------------------------------------------- version: '3.8'  services:  app:  build: ./app  environment:  - POSTGRES\_USER=${POSTGRES\_USER}  - POSTGRES\_PASSWORD=${POSTGRES\_PASSWORD}  - POSTGRES\_DB=${POSTGRES\_DB}  depends\_on:  - db  db:  image: postgres:14  environment:  POSTGRES\_USER: ${POSTGRES\_USER}  POSTGRES\_PASSWORD: ${POSTGRES\_PASSWORD}  POSTGRES\_DB: ${POSTGRES\_DB}  volumes:  - db\_data:/var/lib/postgresql/data  nginx:  image: nginx:alpine  ports:  - "80:80"  volumes:  - ./nginx/default.conf:/etc/nginx/conf.d/default.conf  depends\_on:  - app  volumes:  db\_data: |  |  | | --- | | ------------------------------------------------------------.env------------------------------------------------------------- POSTGRES\_USER=postgres  POSTGRES\_PASSWORD=secret  POSTGRES\_DB=mydb |      |  | | --- | | ----------------------------------------------------------Jenkinsfile------------------------------------------------------ pipeline {  agent any  environment {  DOCKER\_IMAGE = 'your-dockerhub-username/devops-app'  }  stages {  stage('Build') {  steps {  sh 'docker-compose build'  }  }  stage('Test') {  steps {  sh 'docker-compose up -d'  sh 'sleep 10 && curl -f http://localhost || exit 1'  }  }  stage('Push') {  steps {  withCredentials([usernamePassword(credentialsId: 'dockerhub', usernameVariable: 'USER', passwordVariable: 'PASS')]) {  sh """  echo $PASS | docker login -u $USER --password-stdin  docker tag devops-assignment\_app $DOCKER\_IMAGE  docker push $DOCKER\_IMAGE  """  }  }  }  stage('Deploy to AWS') {  steps {  // Use ECS CLI, Helm for EKS, or SSH commands to deploy to EC2  echo 'Deploy step here...'  }  }  }  } | |

Step2 : Create A Directory and place the below files inside it

|  |  |
| --- | --- |
| devops-assignment/  │  ├── app/  │ ├── server.js  │ ├── package.json  │ └── Dockerfile  │  ├── nginx/  │ └── default.conf  │  ├── docker-compose.yml  ├── .env  └── Jenkinsfile |  |

Step3: Execute the docker compose up --build  
  
  
Step4: check the present state  
  
  
Step5: check if the application is Running  


Part3  
Step1: Git Setup

Step2: Setup Jenkins Container  
