رواد مصر الرقمية

وزارة الاتصالات وتكنولوجيا المعلومات

DEPI GRADUATION PROJECT

AUTOMATED DEPLOYMENT PIPELINE WITH JENKINS AND DOCKER

Track: DevOps Engineer

Instructor: Ahmed Nasr

Training Center: YAT Learning Solutions

Group Code: CAI1_SWD1_S6d

Team Number: 3

Team Members:

- Noor Elhoda Mohamed Abdo Eisa
- Ola Youssef Mohammed
- Maram Hassan Mohamed
- Basant Ehab Sayed
- Saif Mohammed Dawood

GitHub Repository of our Project:

https://github.com/DEPI-DevOps-tasks/DevOps-Challenge-fork

1. Introduction

The goal of this project was to develop a fully automated CI/CD pipeline using Jenkins, Docker, Terraform, and Ansible. This pipeline supports the build, testing, and deployment of a sample Dockerized application, facilitating seamless integration and continuous deployment to cloud infrastructure.

2. Technologies used

Jenkins: For automating continuous integration and deployment.

Docker: For containerizing the application.

Ansible: For automating configuration management and deployment.

Terraform: For provisioning cloud infrastructure (AWS).

AWS: For hosting the application on cloud infrastructure (EC2).

GitHub: Version control for source code.

3. Project Phases

Phase 1: Initial Setup & Dockerization

This phase focused on setting up the local environment, building the Dockerized application, and testing locally.

1. **Forking the Repository**: We forked the DevOps challenge repo and worked with the app to build a Dockerfile and docker-compose.yml.
2. **Creating the Docker Image**: A Dockerfile was written to containerize the application. Redis was added as a service using Docker Compose.
3. **Creating the Docker-compose file:** To orchestrate the containers (Python app and Redis), we created a docker-compose.yml file that sets up the necessary services.

Dockerfile Code:

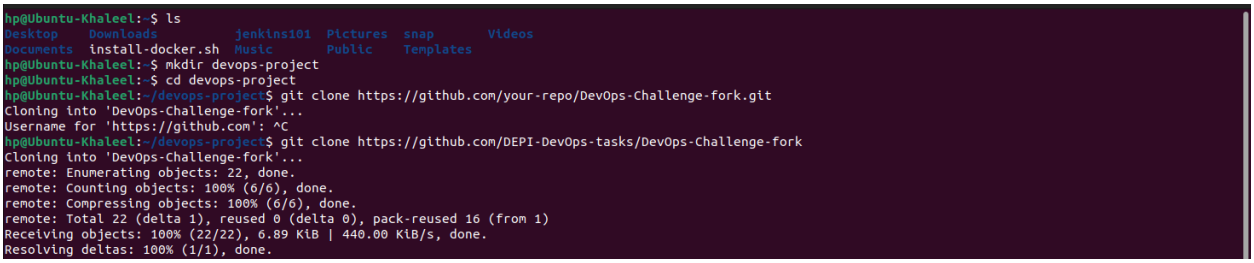
```
FROM python:3.8-slim

WORKDIR /app

COPY requirements.txt ./
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

EXPOSE 5000

CMD ["python", "hello.py"]
```

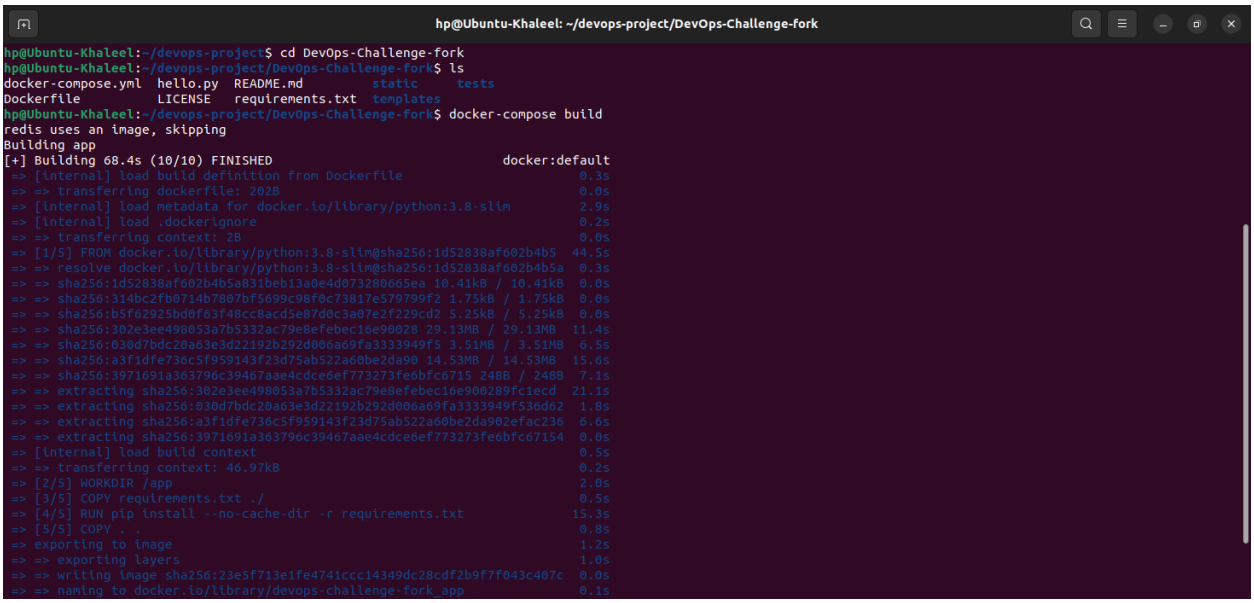
Docker-Compose.yml code:

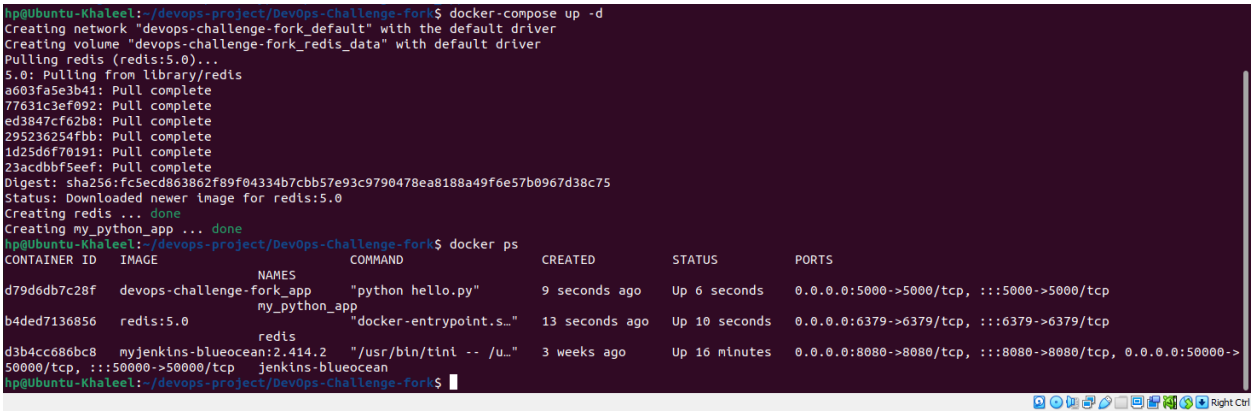
```
version: '3'
services:
  app:
    build: .
    container_name: my_python_app
    ports:
      - "5000:5000"
    environment:
      - ENVIRONMENT=DEV
      - HOST=0.0.0.0
      - PORT=5000
      - REDIS_HOST=redis
      - REDIS_PORT=6379
      - REDIS_DB=0
    depends_on:
      - redis
    command: sh -c "sleep 5 && python hello.py" # Wait for Redis to start

  redis:
    image: "redis:5.0"
    container_name: redis
    ports:
      - "6379:6379"
    volumes:
      - redis_data:/data

volumes:
  redis_data:
```

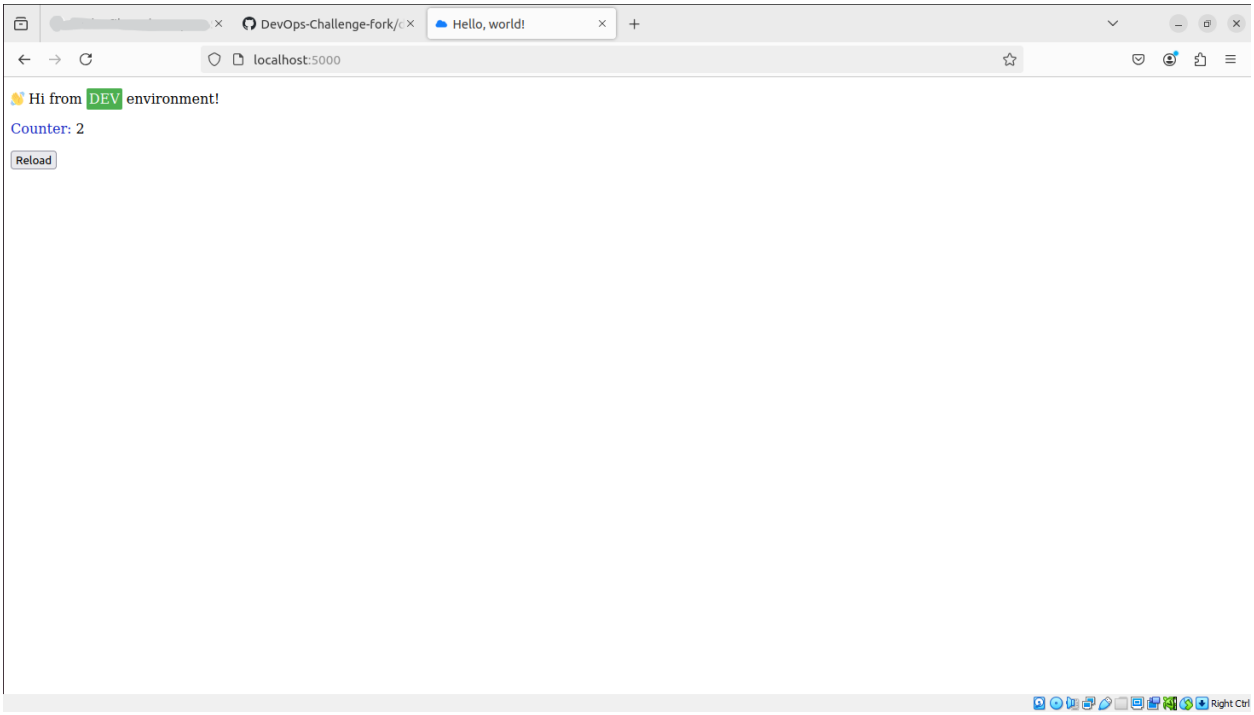
4. **Testing the Application**: The application was tested using Docker Compose, confirming that both the Python app and Redis are up and running locally.

Result: Both services were successfully running on local ports (Python on 5000, Redis on 6379).









Phase 2: Jenkins & CI Integration

This phase established the CI process, integrated Git, and enabled automated testing.

1. **Jenkins Pipeline**: A Jenkinsfile was created to clone the repo, build the Docker image, and run the application using Docker Compose. Unit tests were added to verify the build.
2. **Automated Notifications**: Configured Jenkins to send emails for build successes or failures using a webhook.

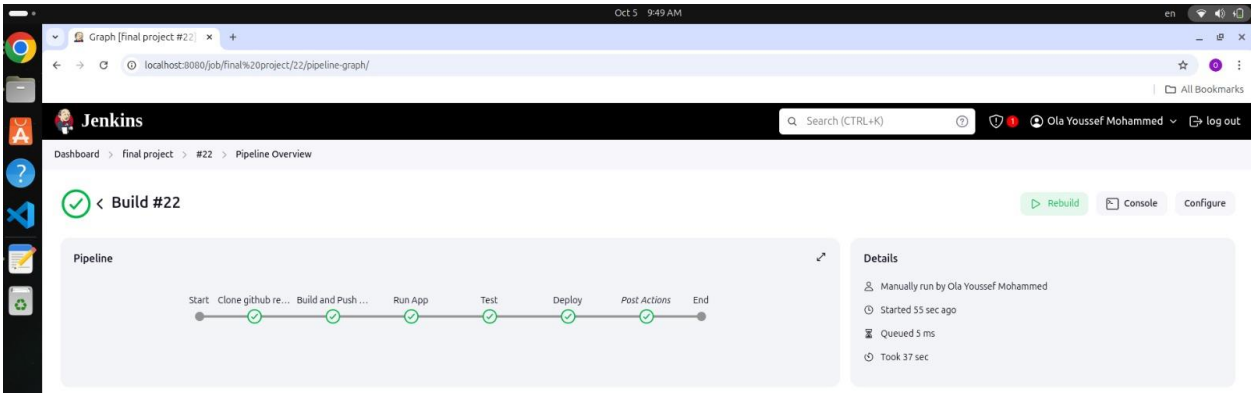
Part of the Pipeline Code:

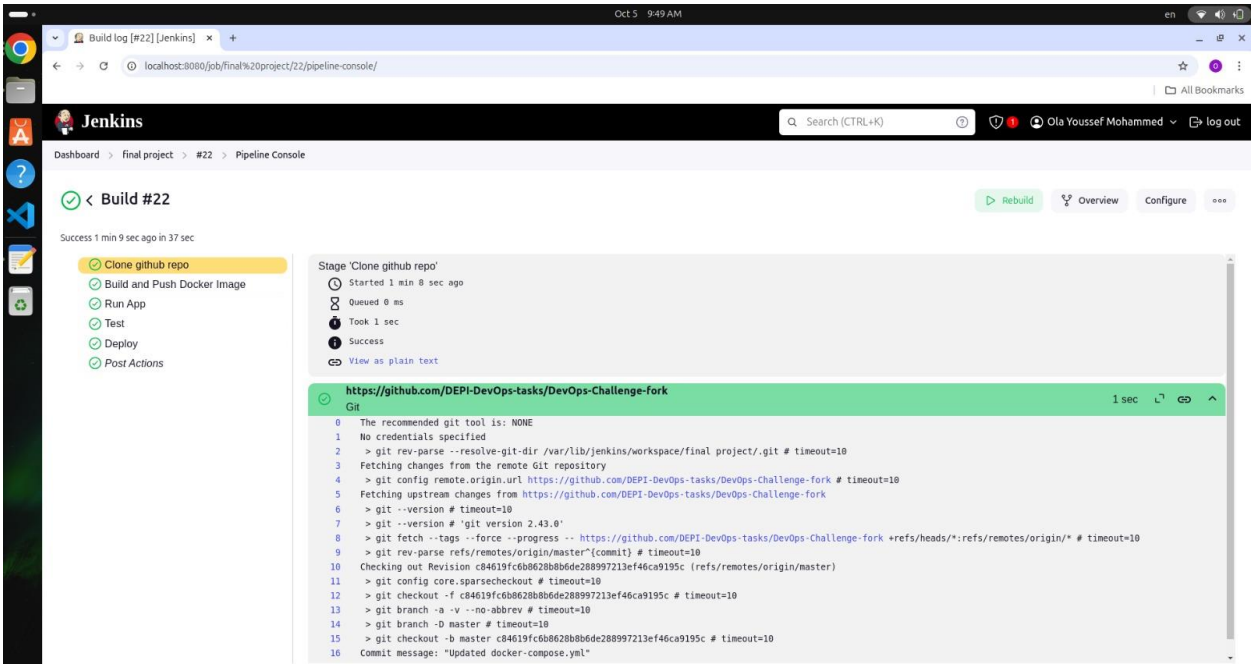
```
pipeline {
    agent any

    environment {
        ENVIRONMENT = 'DEV'
        HOST = '0.0.0.0'
        PORT = '5000'
        REDIS_HOST = 'redis'
        REDIS_PORT = '6379'
        REDIS_DB = '0'
        DOCKER_IMAGE_NAME = 'olayoussef/my_python_app'
    }

    triggers {
        githubPush()
    }

    stages {
        stage('Clone github repo') {
            steps {
                git 'https://github.com/DEPI-DevOps-tasks/DevOps-Challenge-fork'
            }
        }
```

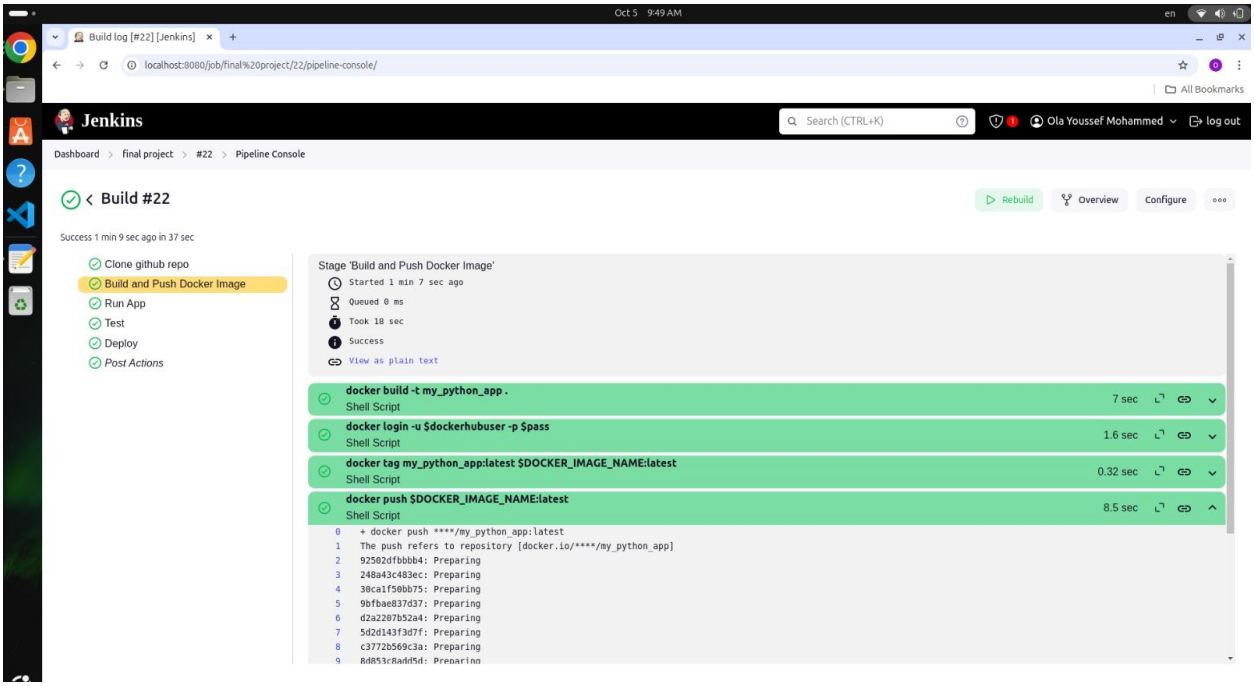
```
        stage('Build and Push Docker Image') {
            steps {
                script {

                    sh 'docker build -t my_python_app .'


                    withCredentials([usernamePassword(credentialsId: 'dockerhub-credentials', passwordVariable:
'pass', usernameVariable: 'dockerhubuser')]) {
                        sh 'docker login -u $dockerhubuser -p $pass'
                        sh 'docker tag my_python_app:latest $DOCKER_IMAGE_NAME:latest'
                        sh 'docker push $DOCKER_IMAGE_NAME:latest'
                    }
                }
            }
        }
```

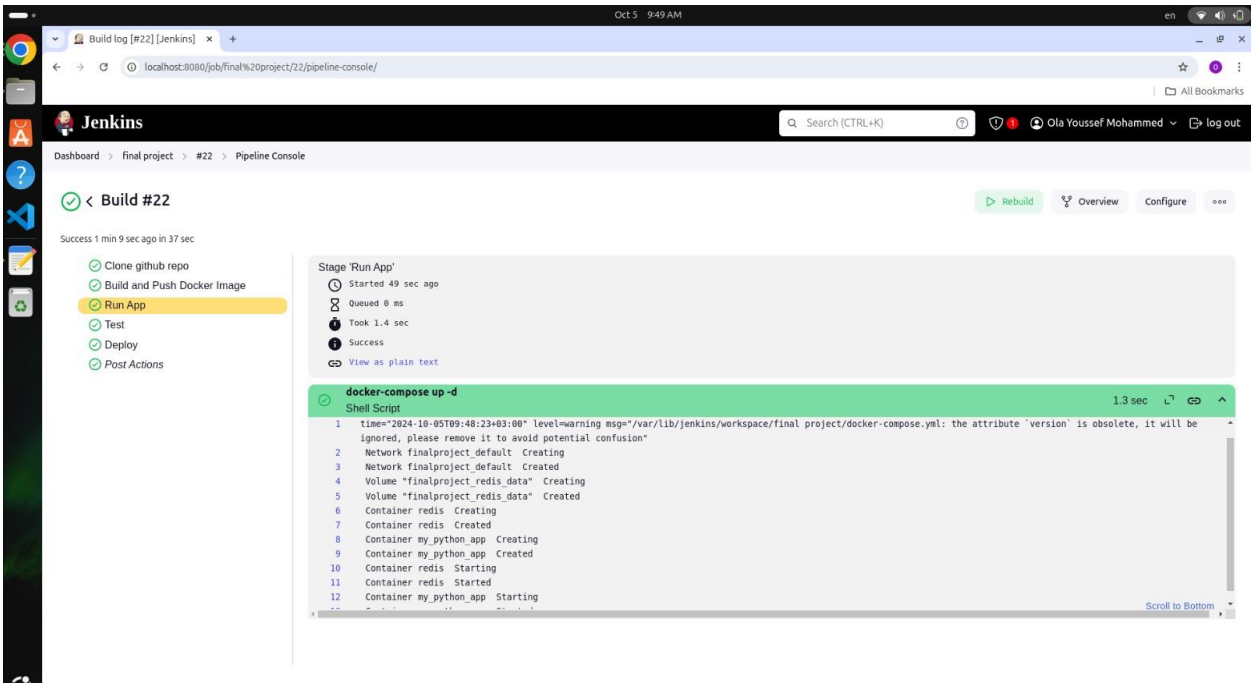
```
stage('Run App') {
            steps {
                script {
                    sh '''
                    # Clean up any existing containers
                    docker rm -f my_python_app || true
                    docker rm -f redis || true
                    # Run the application
                    docker-compose up -d
                    '''
                }
            }
        }

        stage('Test') {
            steps {
                sh 'docker exec my_python_app python tests/test.py'  // Replace with your actual test command
            }
        }

        stage('Deploy') {
            when {
                expression {
                    return currentBuild.result == null || currentBuild.result == 'SUCCESS'
                }
            }
            steps {
                sh 'docker-compose up -d'
            }
        }
    }
```

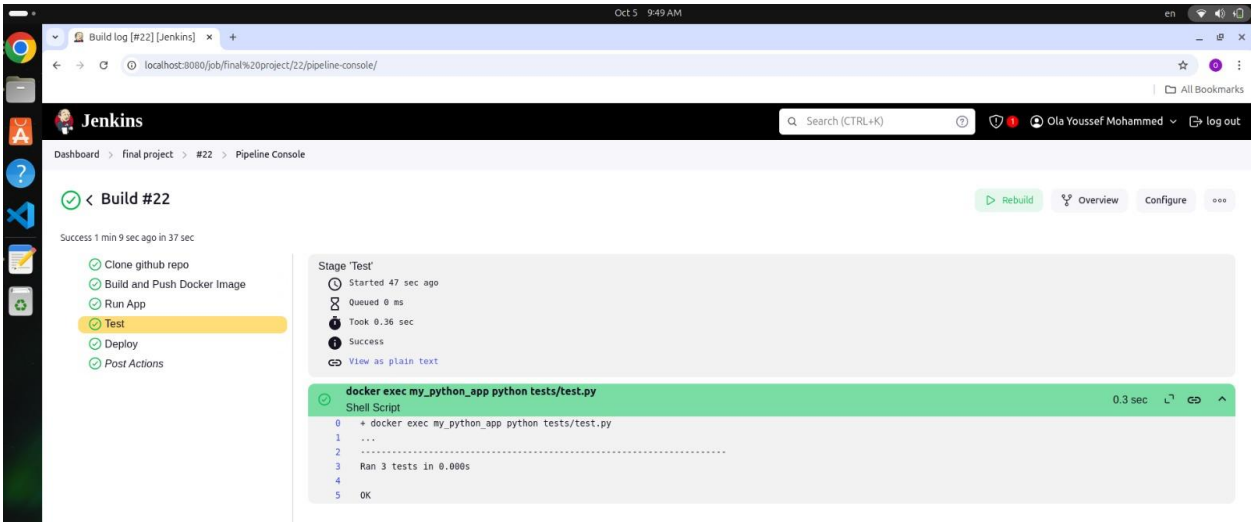
```
    post {
        always {

            sh 'docker-compose down --volumes'
        }
        success {
            mail to: 'olayoubadr@gmail.com, maram.hassan95@gmail.com, basantehab83@gmail.com,
saifdawoodcs@gmail.com, noor.mohamed.eisa@gmail.com',
                 subject: "Build Succeeded: ${env.BUILD_TAG}",
                 body: "The build was successful. Check the logs for details."
        }
        failure {
            mail to: 'olayoubadr@gmail.com, maram.hassan95@gmail.com, basantehab83@gmail.com,
saifdawoodcs@gmail.com, noor.mohamed.eisa@gmail.com',
                 subject: "Build Failed: ${env.BUILD_TAG}",
                 body: "The build failed. Please check the logs."
        }
    }
}
```

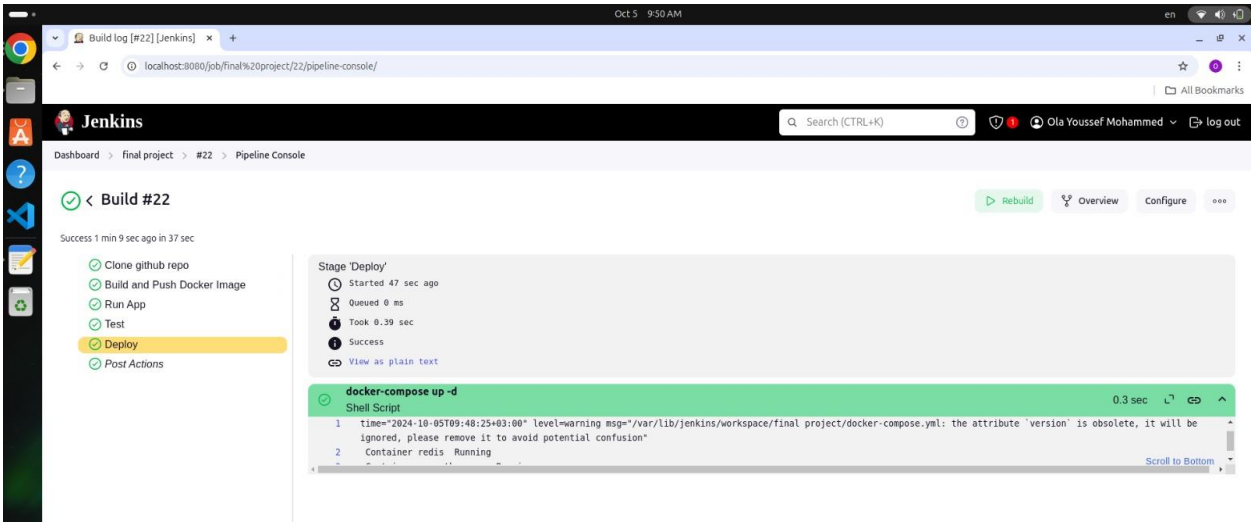

Oct 5 9:49 AM
Graph [final project #22]
localhost:8080/job/final%20project/22/pipeline-graph/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Overview
Build #22
Rebuild
Console
Configure
Pipeline
Start
Clone github re...
Build and Push ...
Run App
Test
Deploy
Post Actions
End
Details
Manually run by Ola Youssef Mohammed
Started 55 sec ago
Queued 5 ms
Took 37 sec

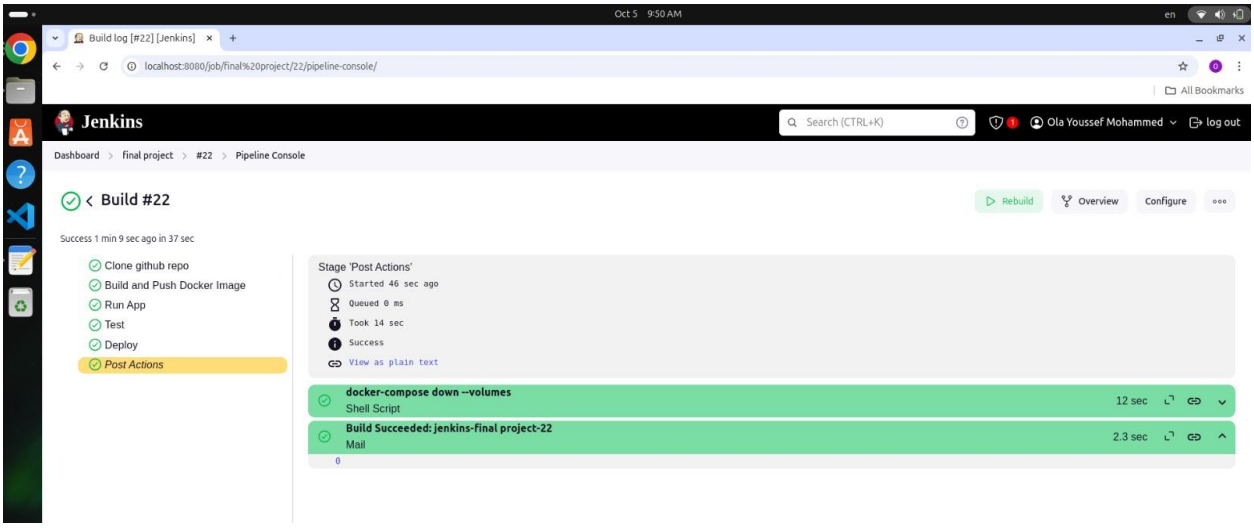

Oct 5 9:49 AM
Build log [#22] [Jenkins]
localhost:8080/job/final%20project/22/pipeline-console/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Console
Build #22
Rebuild
Overview
Configure
Success 1 min 9 sec ago in 37 sec
Clone github repo
Build and Push Docker Image
Run App
Test
Deploy
Post Actions
Stage 'Clone github repo'
Started 1 min 8 sec ago
Queued 0 ms
Took 1 sec
Success
View as plain text
https://github.com/DEPI-DevOps-tasks/DevOps-Challenge-fork
Git
1 sec
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/final project/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/DEPI-DevOps-tasks/DevOps-Challenge-fork # timeout=10
Fetching upstream changes from https://github.com/DEPI-DevOps-tasks/DevOps-Challenge-fork
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/DEPI-DevOps-tasks/DevOps-Challenge-fork +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision c84619fc6b8628b8b6de288997213ef46ca9195c (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f c84619fc6b8628b8b6de288997213ef46ca9195c # timeout=10
> git branch -a -v --no-abbrev # timeout=10
> git branch -D master # timeout=10
> git checkout -b master c84619fc6b8628b8b6de288997213ef46ca9195c # timeout=10
Commit message: "Updated docker-compose.yml"

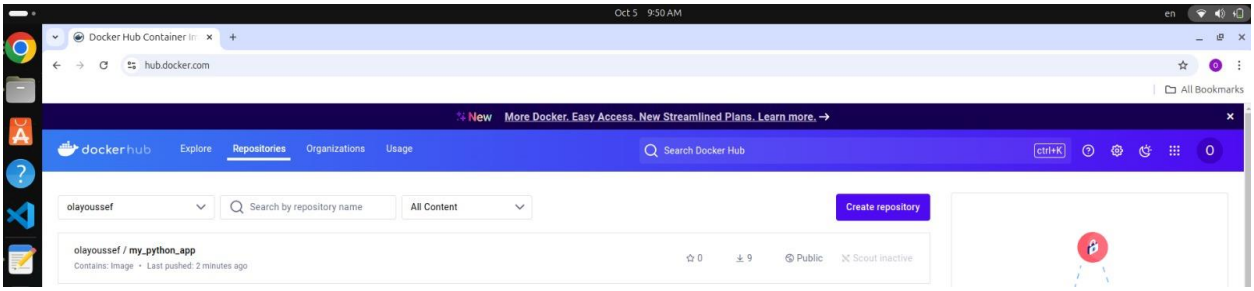

Oct 5 9:49 AM
Build log [#22] [Jenkins]
localhost:8080/job/final%20project/22/pipeline-console/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Console
Build #22
Rebuild
Overview
Configure
Success 1 min 9 sec ago in 37 sec
Clone github repo
Build and Push Docker Image
Run App
Test
Deploy
Post Actions
Stage 'Build and Push Docker Image'
Started 1 min 7 sec ago
Queued 0 ms
Took 18 sec
Success
View as plain text
docker build -t my_python_app .
Shell Script
7 sec
docker login -u $dockerhubuser -p $pass
Shell Script
1.6 sec
docker tag my_python_app:latest $DOCKER_IMAGE_NAME:latest
Shell Script
0.32 sec
docker push $DOCKER_IMAGE_NAME:latest
Shell Script
8.5 sec
+ docker push ****/my_python_app:latest
The push refers to repository [docker.io/****/my_python_app]
92502dfbbbb4: Preparing
248a43c483ec: Preparing
30ca1f50bb75: Preparing
9bfbae837d37: Preparing
d2a2207b52a4: Preparing
5d2d143f3d7f: Preparing
c3772b569c3a: Preparing

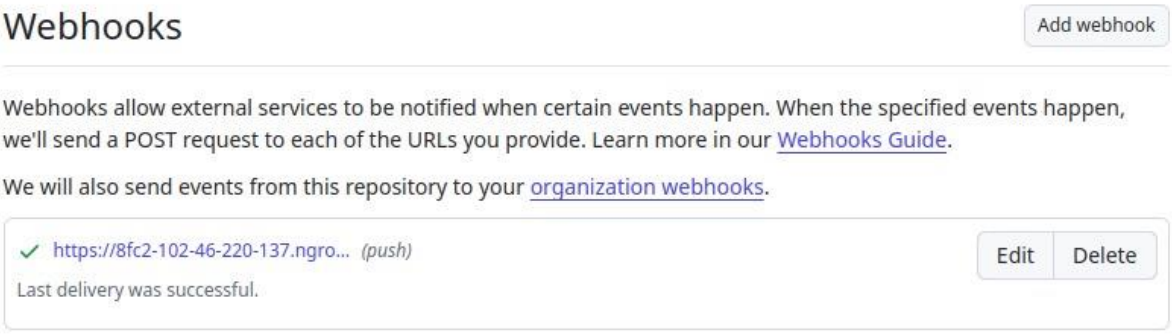

Oct 5 9:49 AM
Build log [#22] [Jenkins]
localhost:8080/job/final%20project/22/pipeline-console/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Console
Build #22
Rebuild
Overview
Configure
Success 1 min 9 sec ago in 37 sec
Clone github repo
Build and Push Docker Image
Run App
Test
Deploy
Post Actions
Stage 'Run App'
Started 49 sec ago
Queued 0 ms
Took 1.4 sec
Success
View as plain text
docker-compose up -d
Shell Script
1.3 sec
time="2024-10-05T09:48:23+03:00" level=warning msg="/var/lib/jenkins/workspace/final project/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
Network finalproject_default Creating
Network finalproject_default Created
Volume "finalproject_redis_data" Creating
Volume "finalproject_redis_data" Created
Container redis Creating
Container redis Created
Container my_python_app Creating
Container my_python_app Created
Container redis Starting
Container redis Started
Container my_python_app Starting
Scroll to Bottom


Oct 5 9:49 AM
Build log [#22] [Jenkins]
localhost:8080/job/final%20project/22/pipeline-console/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Console
Build #22
Rebuild
Overview
Configure
Success 1 min 9 sec ago in 37 sec
Clone github repo
Build and Push Docker Image
Run App
Test
Deploy
Post Actions
Stage 'Test'
Started 47 sec ago
Queued 0 ms
Took 0.36 sec
Success
View as plain text
docker exec my_python_app python tests/test.py
Shell Script
0.3 sec
+ docker exec my_python_app python tests/test.py
...
Ran 3 tests in 0.000s
OK


Oct 5 9:50 AM
Build log [#22] [Jenkins]
localhost:8080/job/final%20project/22/pipeline-console/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Console
Build #22
Rebuild
Overview
Configure
Success 1 min 9 sec ago in 37 sec
Clone github repo
Build and Push Docker Image
Run App
Test
Deploy
Post Actions
Stage 'Deploy'
Started 47 sec ago
Queued 0 ms
Took 0.39 sec
Success
View as plain text
docker-compose up -d
Shell Script
0.3 sec
time="2024-10-05T09:48:25+03:00" level=warning msg="/var/lib/jenkins/workspace/final project/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"
Container redis Running
Scroll to Bottom


Oct 5 9:50 AM
Build log [#22] [Jenkins]
localhost:8080/job/final%20project/22/pipeline-console/
Jenkins
Search (CTRL+K)
Ola Youssef Mohammed
log out
Dashboard > final project > #22 > Pipeline Console
Build #22
Rebuild
Overview
Configure
Success 1 min 9 sec ago in 37 sec
Clone github repo
Build and Push Docker Image
Run App
Test
Deploy
Post Actions
Stage 'Post Actions'
Started 46 sec ago
Queued 0 ms
Took 14 sec
Success
View as plain text
docker-compose down --volumes
Shell Script
12 sec
Build Succeeded: jenkins-final project-22
Mail
2.3 sec



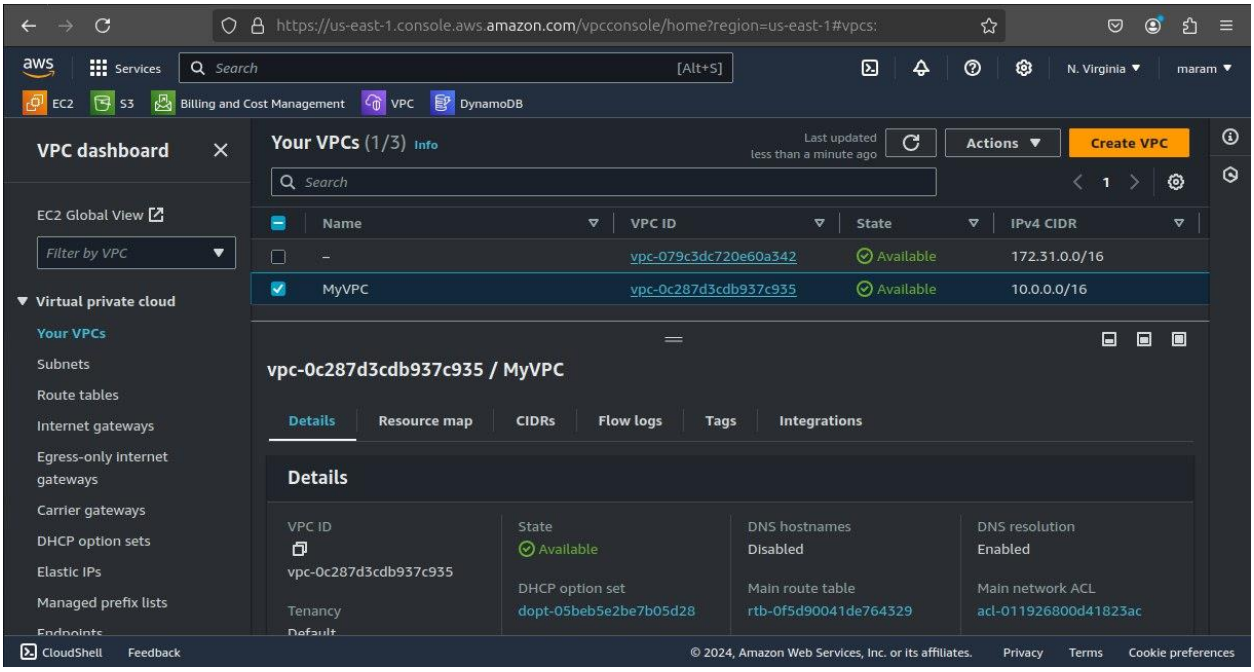


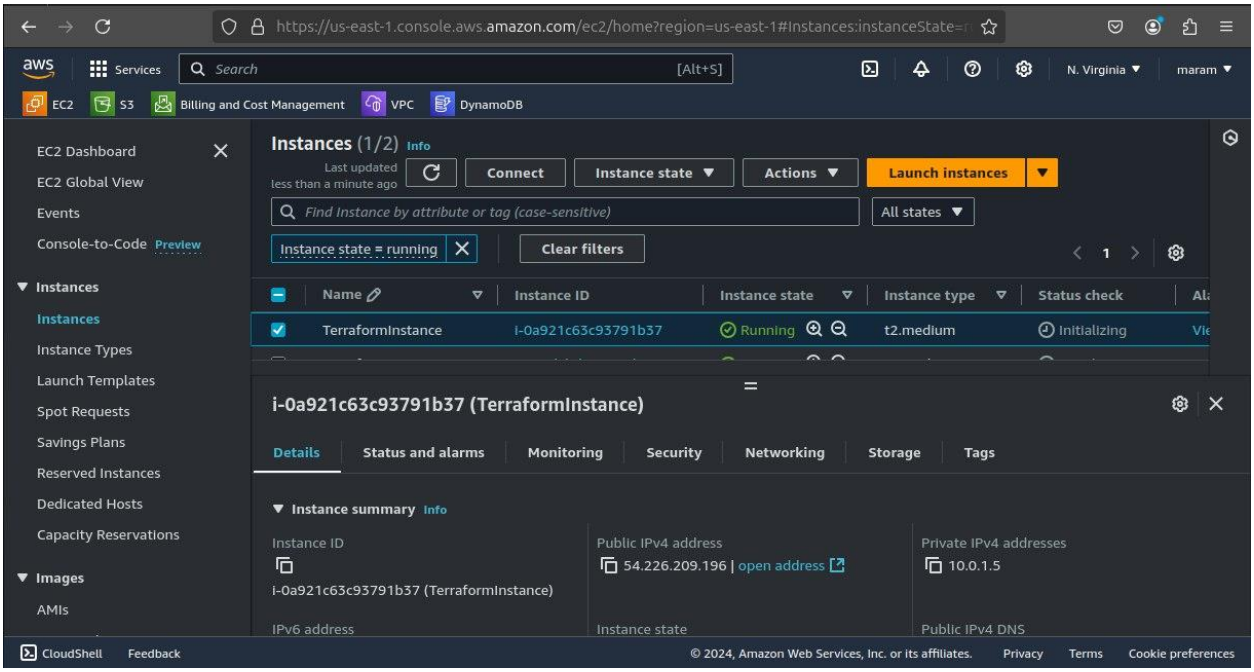
Phase 3: Infrastructure Provisioning with Terraform

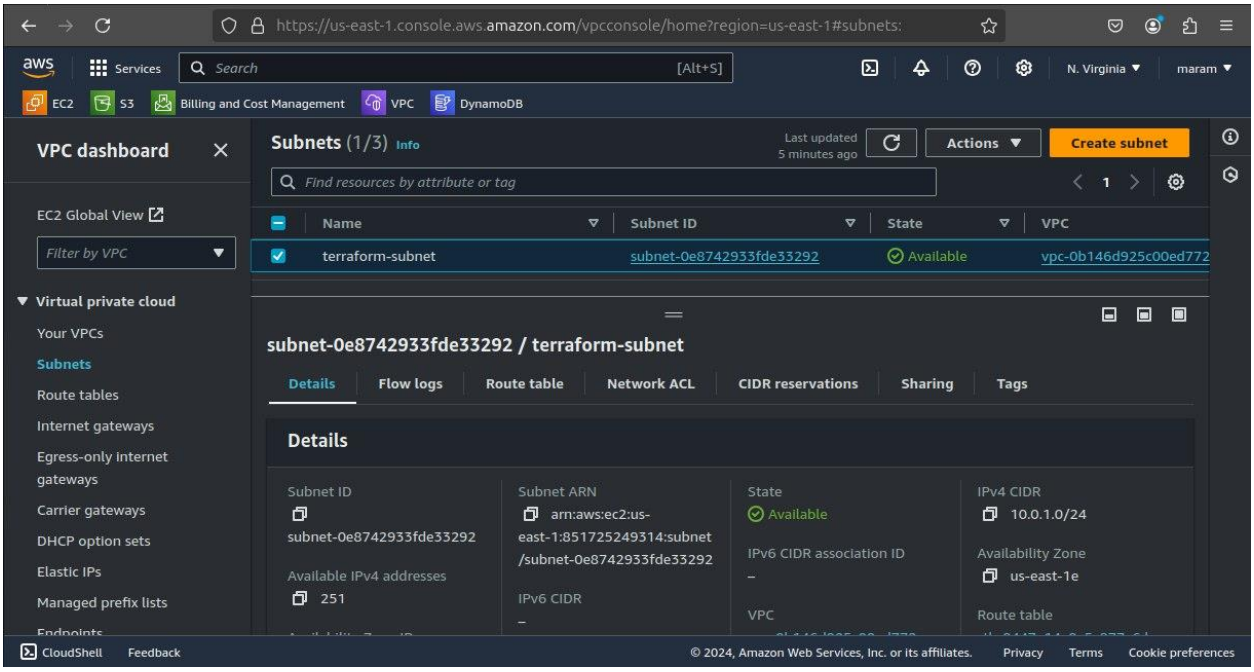
This phase used Terraform to provision AWS resources for the application's deployment.

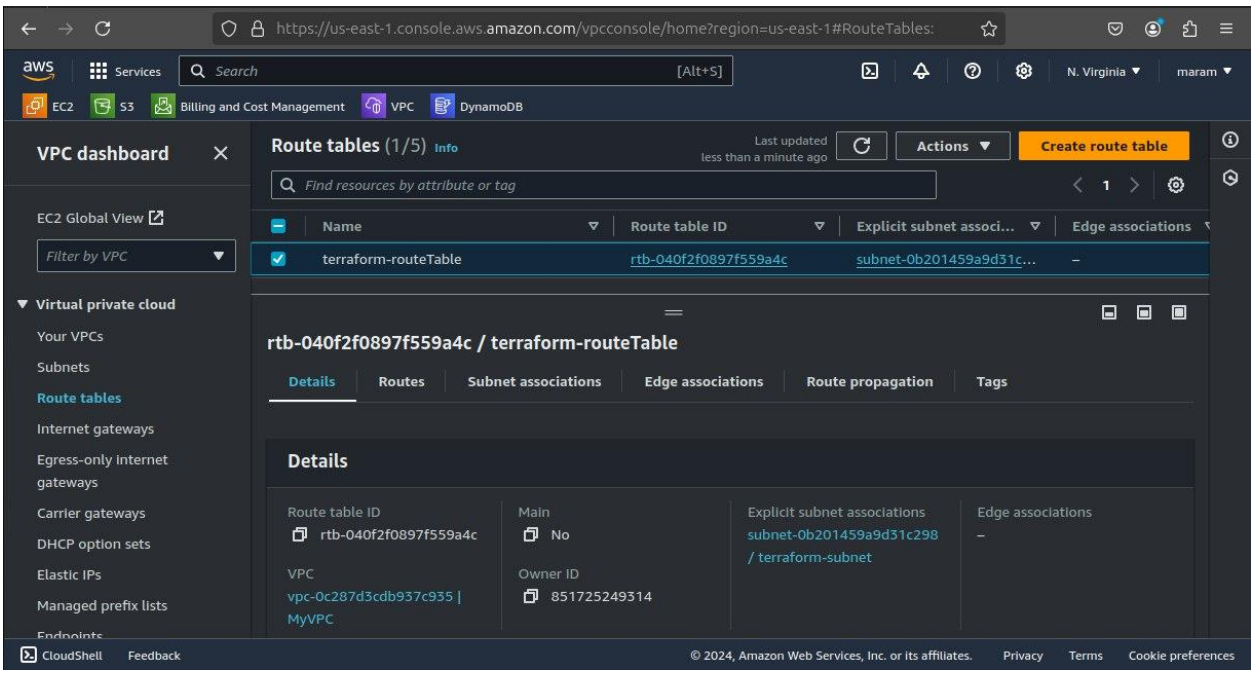
1. **Terraform Script**: A Terraform script was written to create an EC2 instance on AWS. It also used a local provisioner to initiate the Ansible playbook for configuration management.
2. **Jenkins Pipeline for Terraform**: A Jenkins pipeline was created to execute the Terraform script, setting up the infrastructure automatically.
3. **Code:** (See full code in our GitHub Repo)

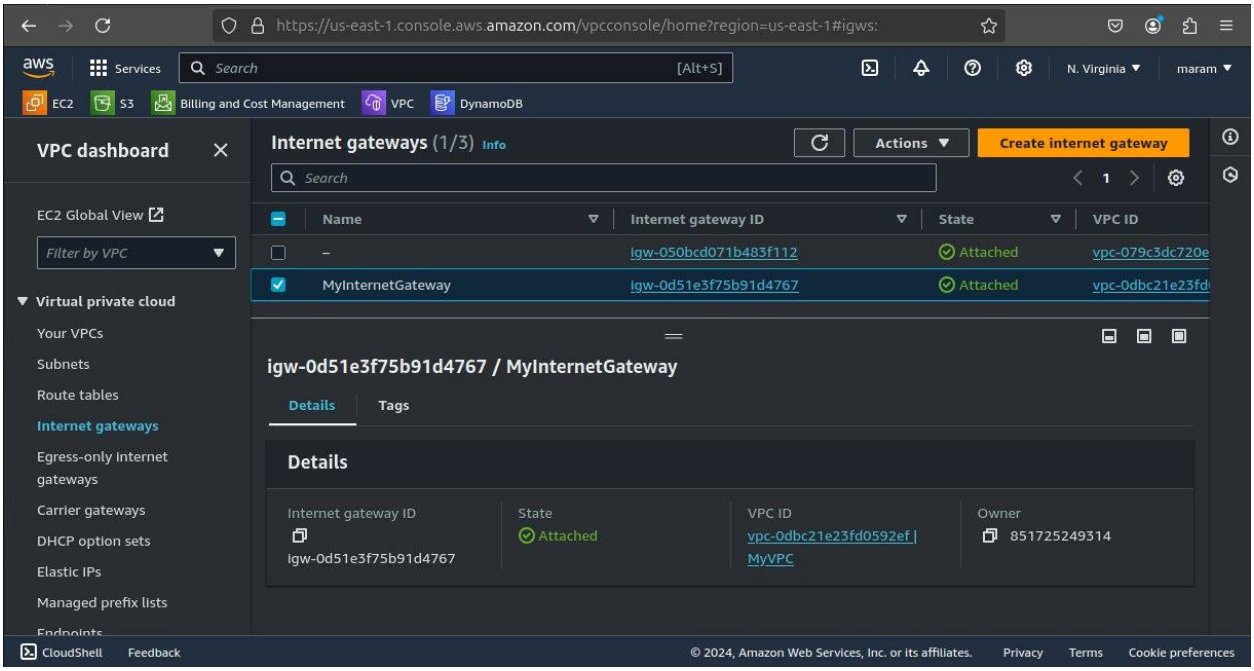
```
provider "aws" {
  region = "us-west-2"
}
resource "aws_instance" "my_ec2" { ... }
```

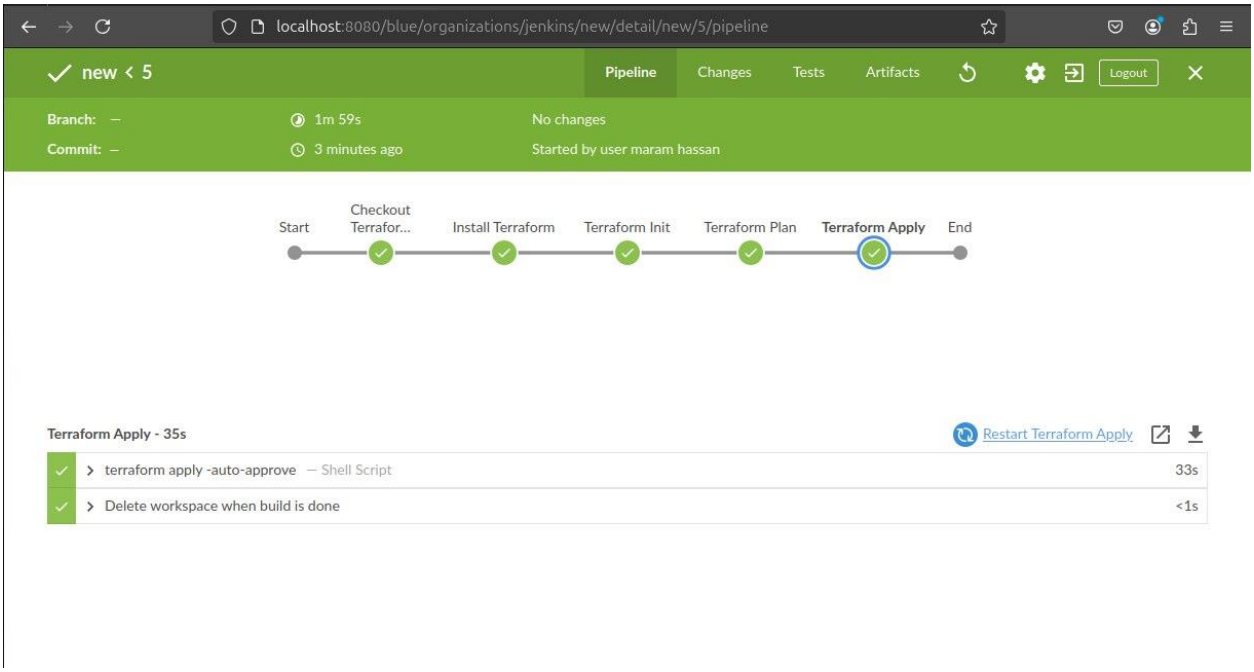

https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#vpcs:
Services
Search
[Alt+S]
N. Virginia
maram
EC2
S3
Billing and Cost Management
VPC
DynamoDB
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
Endpoints
Your VPCs (1/3) Info
Last updated less than a minute ago
Actions
Create VPC
Search
Name
VPC ID
State
IPv4 CIDR
– vpc-079c3dc720e60a342 Available 172.31.0.0/16
MyVPC vpc-0c287d3cdb937c935 Available 10.0.0.0/16
vpc-0c287d3cdb937c935 / MyVPC
Details
Resource map
CIDRs
Flow logs
Tags
Integrations
Details
VPC ID
vpc-0c287d3cdb937c935
State
Available
DNS hostnames
Disabled
DNS resolution
Enabled
Tenancy
Default
DHCP option set
dopt-05beb5e2be7b05d28
Main route table
rtb-0f5d90041de764329
Main network ACL
acl-011926800d41823ac
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:instanceState=
Services
Search
[Alt+S]
N. Virginia
maram
EC2
S3
Billing and Cost Management
VPC
DynamoDB
EC2 Dashboard
EC2 Global View
Events
Console-to-Code Preview
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
Instances (1/2) Info
Last updated less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Instance state = running
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
TerraformInstance i-0a921c63c93791b37 Running t2.medium Initializing
i-0a921c63c93791b37 (TerraformInstance)
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance summary Info
Instance ID
i-0a921c63c93791b37 (TerraformInstance)
Public IPv4 address
54.226.209.196 | open address
Private IPv4 addresses
10.0.1.5
IPv6 address
Instance state
Public IPv4 DNS
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#subnets:
Services
Search
[Alt+S]
N. Virginia
maram
EC2
S3
Billing and Cost Management
VPC
DynamoDB
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
Subnets (1/3) Info
Last updated 5 minutes ago
Actions
Create subnet
Find resources by attribute or tag
Name
Subnet ID
State
VPC
terraform-subnet
subnet-0e8742933fde33292
Available
vpc-0b146d925c00ed772
subnet-0e8742933fde33292 / terraform-subnet
Details
Flow logs
Route table
Network ACL
CIDR reservations
Sharing
Tags
Subnet ID
subnet-0e8742933fde33292
Subnet ARN
arn:aws:ec2:us-east-1:851725249314:subnet/subnet-0e8742933fde33292
State
Available
IPv4 CIDR
10.0.1.0/24
Available IPv4 addresses
251
IPv6 CIDR
IPv6 CIDR association ID
Availability Zone
us-east-1e
VPC
Route table
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#RouteTables:
Services
Search
[Alt+S]
N. Virginia
maram
EC2
S3
Billing and Cost Management
VPC
DynamoDB
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
Route tables (1/5) Info
Last updated less than a minute ago
Actions
Create route table
Find resources by attribute or tag
Name
Route table ID
Explicit subnet associ...
Edge associations
terraform-routeTable
rtb-040f2f0897f559a4c
subnet-0b201459a9d31c...
rtb-040f2f0897f559a4c / terraform-routeTable
Details
Routes
Subnet associations
Edge associations
Route propagation
Tags
Route table ID
rtb-040f2f0897f559a4c
Main
No
Explicit subnet associations
subnet-0b201459a9d31c298 / terraform-subnet
Edge associations
VPC
vpc-0c287d3cdb937c935 | MyVPC
Owner ID
851725249314
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#igws:
Services
Search
[Alt+S]
N. Virginia
maram
EC2
S3
Billing and Cost Management
VPC
DynamoDB
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
Endpoints
Internet gateways (1/3) Info
Actions
Create internet gateway
Search
Name
Internet gateway ID
State
VPC ID
igw-050bcd071b483f112
Attached
MyInternetGateway
igw-0d51e3f75b91d4767
Attached
igw-0d51e3f75b91d4767 / MyInternetGateway
Details
Tags
Internet gateway ID
igw-0d51e3f75b91d4767
State
Attached
VPC ID
vpc-0dbc21e23fd0592ef | MyVPC
Owner
851725249314
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


localhost:8080/blue/organizations/jenkins/new/detail/new/5/pipeline
new < 5
Pipeline
Changes
Tests
Artifacts
Logout
Branch: –
Commit: –
1m 59s
3 minutes ago
No changes
Started by user maram hassan
Start
Checkout Terrafor...
Install Terraform
Terraform Init
Terraform Plan
Terraform Apply
End
Terraform Apply - 35s
Restart Terraform Apply
terraform apply -auto-approve — Shell Script
33s
Delete workspace when build is done
<1s

Phase 4: Deployment with Ansible

Ansible was used to automate the deployment of the application to the AWS EC2 instance.

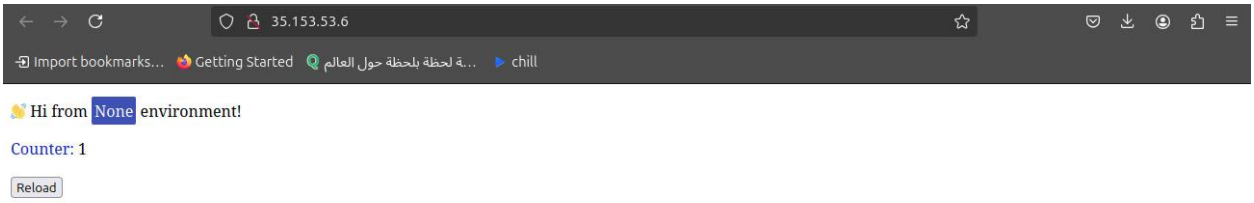
1. **Ansible Playbook**: The playbook installs Docker on the EC2 instance, pulls the application's Docker image, and runs the app and Redis containers.
2. **Testing Deployment**: Verified that the application was running on the EC2 instance.

Code:

```yaml
- name: Deploy Docker Application on AWS EC2
  hosts: all
  become: true

  tasks:
    - name: Gather facts
      setup:

    - name: Update apt and install required packages
      apt:
        name: "{{ item }}"
        state: present
      with_items:
        - docker.io
        - python3-pip
        - python3-venv
        - python3-apt
        - curl  # Ensure curl is installed
        - git  # Optional: Install git if you need version control
```

```
    - name: Start Docker service
      service:
        name: docker
        state: started
        enabled: true

    - name: Create Python virtual environment
      command: python3 -m venv /home/ubuntu/venv
      args:
        creates: /home/ubuntu/venv

    - name: Install Docker Python module in virtual environment
      command: /home/ubuntu/venv/bin/python -m pip install docker

    - name: Create a directory for the application
      file:
        path: /home/ubuntu/app
        state: directory
```

```
    - name: Copy docker-compose.yml to EC2 instance
      copy:
        src: ./docker-compose.yml
        dest: /home/ubuntu/app/docker-compose.yml

    - name: Install Docker Compose
      shell: >
        curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)"
-o /usr/local/bin/docker-compose
      args:
        creates: /usr/local/bin/docker-compose

    - name: Set permissions for Docker Compose
      command: chmod +x /usr/local/bin/docker-compose

    - name: Verify Docker Compose installation
      command: docker-compose --version
      register: docker_compose_version

    - debug:
        var: docker_compose_version.stdout

    - name: Pull Docker images
      command: docker-compose -f /home/ubuntu/app/docker-compose.yml pull
      args:
        chdir: /home/ubuntu/app

    - name: Run Docker containers
      command: docker-compose -f /home/ubuntu/app/docker-compose.yml up -d
      args:
        chdir: /home/ubuntu
```



CONCLUSION

This project implemented an automated CI/CD pipeline that allows for continuous integration, testing, and deployment of a Dockerized application to a cloud environment using Jenkins, Docker, Terraform, and Ansible. The pipeline is flexible, scalable, and enables the rapid deployment of applications to production environments.