



## BÁO CÁO THỰC HÀNH

**Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins**

**Môn học:** Công nghệ DevOps và Ứng dụng

**Lớp:** NT548.P11.MMCL

**GVHD:** ThS. Lê Anh Tuấn

### THÀNH VIÊN THỰC HIỆN (Nhóm 14):

STT	Họ và tên	MSSV	Điểm
1	Lâm Bảo Duy	20521231	
2	Hồ Hải Dương	21520202	10

### ĐÁNH GIÁ KHÁC:

Tổng thời gian thực hiện	3 tuần
Link GitHub	
Phân chia công việc	
Ý kiến (nếu có) + Khó khăn + Đề xuất, kiến nghị	

*Phần bên dưới của báo cáo này là báo cáo chi tiết của nhóm thực hiện.*

## MỤC LỤC

1. Quản lý và triển khai hạ tầng AWS sử dụng Terraform và tự động hóa quy trình với Github Actions .....	3
1.1 Kết nối với AWS .....	3
1.2 Cấu hình Access Key và Secret Key cho dự án .....	3
1.3 Tạo workflows cho Terraform.....	3
1.4 Cấu hình trigger cho GitHub Actions.....	4
1.5 Định nghĩa các stage trong pipeline .....	5
1.5.1 Checkout Repository .....	5
1.5.2 Install Checkov .....	6
1.5.3 Run Checkov .....	6
1.5.4 Terraform Init .....	6
1.5.5 Terraform Format .....	6
1.5.6 Terraform Plan.....	7
1.5.7 Terraform Apply.....	7
1.6 Kiểm tra pipeline trên GitHub Actions .....	7
1.6.1 Truy cập GitHub Actions và xem kết quả pipeline .....	7
1.6.2 Kiểm tra log và kết quả của từng bước .....	8
1.6.3 Xác minh trạng thái Terraform Apply .....	8
1.6.4 Kiểm tra lại trên AWS .....	9
1.7 Kiểm tra hạ tầng AWS đã triển khai tự động lên AWS .....	9
1.7.1 Kiểm tra VPC .....	9
1.7.2 Kiểm tra các Subnet.....	10
1.7.3 Kiểm tra Internet Gateway và NAT Gateway .....	10
1.7.4 Kiểm tra Route Tables .....	10
1.7.5 Kiểm tra Security Groups .....	10
1.7.6 Kiểm tra EC2 Instances .....	11
2. Triển khai hạ tầng AWS với CloudFormation và tự động hóa quy trình build và deploy với AWS CodePipeline .....	11
2.1 Dùng CloudFormation để triển khai các dịch vụ AWS.....	11
2.2 Sử dụng AWS CodeBuild, tích hợp cfn-lint và Taskcat .....	12
2.2.1 CodeCommit.....	12
2.2.2 CodeBuild .....	12
2.3 Sử dụng AWS CodePipeline để tự động hóa quy trình build và deploy từ mã nguồn .....	17
3/ Sử dụng Jenkins để quản lý quy trình CI/CD cho ứng dụng microservices.....	25



## BÁO CÁO CHI TIẾT

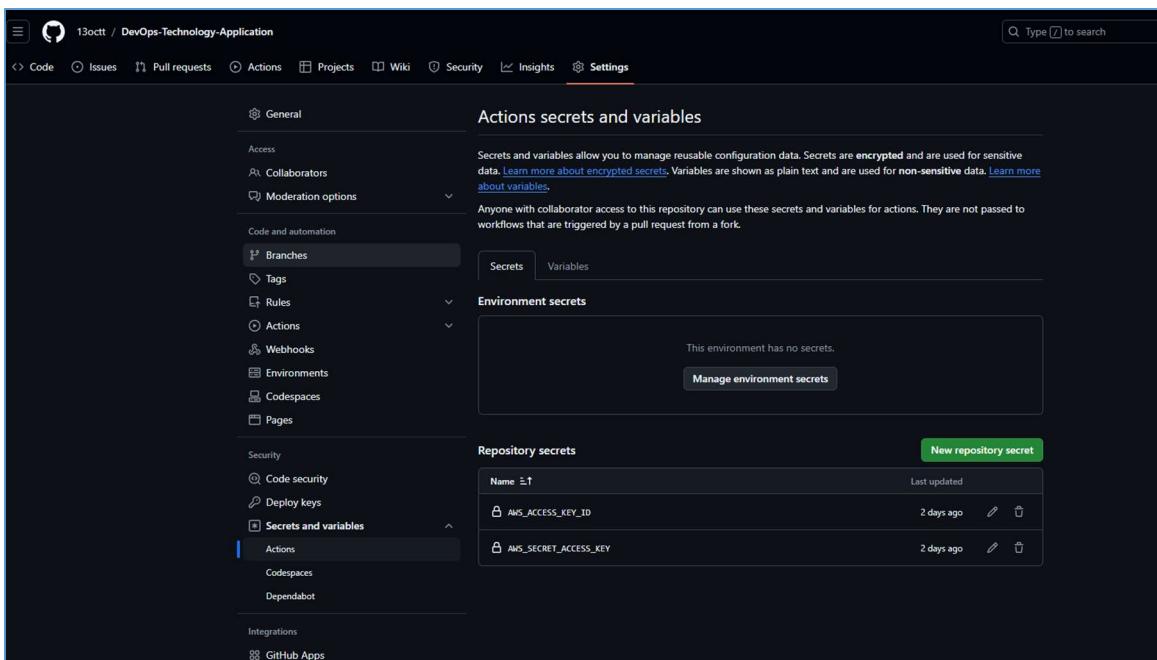
### 1. Quản lý và triển khai hạ tầng AWS sử dụng Terraform và tự động hóa quy trình với Github Actions

#### 1.1 Kết nối với AWS

Sử dụng lệnh “*aws configure*” để nhập Access Key và Secret Key của tài khoản AWS nhằm cho phép cấu hình mặc định AWS CLI, giúp Terraform có thể sử dụng các thông tin xác thực để truy cập vào các dịch vụ AWS.

#### 1.2 Cấu hình Access Key và Secret Key cho dự án

Truy cập GitHub > Settings > Secrets and variables > Actions > New repository secret để tạo Access Key và Secret Key thay vì lưu trực tiếp trong code.



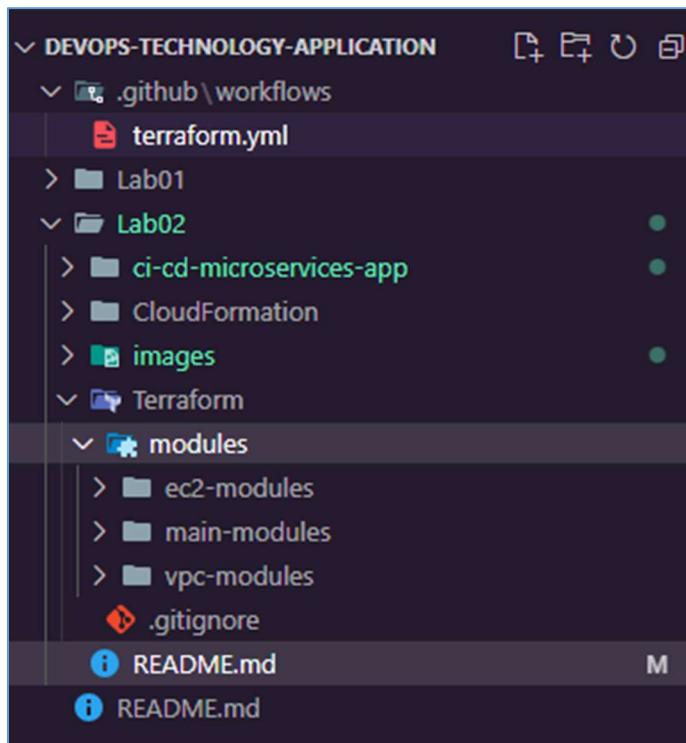
Hình 1 – Tạo repository secret

Tạo hai secrets mới là AWS\_ACCESS\_KEY\_ID và AWS\_SECRET\_ACCESS\_KEY. Các biến môi trường này sẽ được GitHub Actions sử dụng trong quá trình chạy pipeline, giúp bảo mật các thông tin nhạy cảm của tài khoản AWS:

- AWS\_ACCESS\_KEY\_ID: Hoạt động như một username, cho phép các dịch vụ hoặc ứng dụng xác định người dùng khi truy cập AWS.
- AWS\_SECRET\_ACCESS\_KEY: Là khóa bí mật liên kết với AWS\_ACCESS\_KEY\_ID, hoạt động như một mật khẩu. Khóa này cung cấp quyền truy cập bảo mật để xác thực các yêu cầu đến AWS, đảm bảo rằng người dùng có quyền truy cập vào các tài nguyên AWS.

#### 1.3 Tạo workflows cho Terraform

Sử dụng workflows giúp giảm thiểu việc thủ công và bảo đảm rằng mọi thay đổi đều được triển khai tự động, giúp tăng tính linh hoạt và nhanh chóng.



Hình 2 – Code pipeline Terraform cho Lab02 trong file terraform.yml

Tạo file terraform.yml trong thư mục .github/workflows để định nghĩa quy trình triển khai hạ tầng bằng Terraform.

#### 1.4 Cấu hình trigger cho GitHub Actions

Thực hiện cấu hình trigger cho workflows GitHub Actions trong file terraform.yml để các thay đổi về mã nguồn sẽ được áp dụng tự động.

Trigger sẽ xác định khi nào workflows được kích hoạt tự động, giúp kiểm soát việc triển khai hạ tầng AWS bằng Terraform dựa trên các sự kiện trong repository.

```
.github > workflows > terraform.yml > name
GitHub Workflow - YAML GitHub Workflow (github-workflow.j
1   name: 'Terraform Deploy with Github Actions'
2
3   on:
4     push:
5       branches: [ "lab02/terra" ]
6     pull_request:
7       branches: [ "main" ]
8
```

Hình 3 – Trigger pipeline

Cấu hình sự kiện “on”:

- push: Workflows sẽ tự động kích hoạt khi có lệnh push vào nhánh lab02/terra, đảm bảo rằng mọi thay đổi trên nhánh này sẽ được triển khai tự động lên hạ tầng AWS.

- pull\_request: Workflows cũng kích hoạt khi có yêu cầu pull request vào nhánh main, giúp kiểm tra và xác thực các thay đổi trước khi hợp nhất vào nhánh chính, đảm bảo tính an toàn cho quá trình triển khai.

## 1.5 Định nghĩa các stage trong pipeline

Tại file terraform.yml, các stage của pipeline nhằm bảo đảm quy trình triển khai hạ tầng AWS bằng Terraform diễn ra tuần tự, chính xác và bảo mật.

```
12   jobs:
13     terraform:
14       name: 'Terraform'
15       runs-on: ubuntu-latest
16
17     steps:
18       - name: Checkout Repository
19         uses: actions/checkout@v4
20
21       - name: "Install Checkov"
22         run: pip install checkov
23
24       - name: "Run Checkov"
25         run: checkov -f main.tf
26         working-directory: Lab02/Terraform/modules/main-modules
27
28       - name: Terraform Init
29         run: terraform init
30         working-directory: Lab02/Terraform/modules/main-modules
31
32       - name: Terraform Format
33         run: terraform fmt
34         working-directory: Lab02/Terraform/modules/
35
36       - name: Terraform Plan
37         env:
38           AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
39           AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
40         run: terraform plan -input=false
41         working-directory: Lab02/Terraform/modules/main-modules
42
43       - name: Terraform Apply
44         if: (github.ref == 'refs/heads/lab02/terra' && github.event_name == 'push')
45         env:
46           AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
47           AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
48         run: terraform apply -auto-approve -input=false
49         working-directory: Lab02/Terraform/modules/main-modules
```

Hình 4 – Định nghĩa các stage

### 1.5.1 Checkout Repository

Sử dụng actions/checkout@v4 để tải mã nguồn từ repository về môi trường GitHub Actions.

Bảo đảm rằng các file Terraform và các file cấu hình cần thiết đều có sẵn để sử dụng trong các bước tiếp theo. Việc tải mã nguồn là bước cơ bản giúp mọi thay đổi từ repository được cập nhật trong pipeline.

```
18   - name: Checkout Repository
19     uses: actions/checkout@v4
```

Hình 5 – Checkout Repository

### 1.5.2 Install Checkov

Cài đặt Checkov – công cụ IaC. Checkov giúp phát hiện các lỗ hổng bảo mật và đảm bảo rằng mã Terraform tuân thủ các quy định bảo mật trước khi được triển khai lên AWS.

```
21      - name: "Install Checkov"  
22      run: pip install checkov
```

Hình 6 – Install Checkov

“**pip install checkov**” dùng để cài đặt Checkov thông qua pip.

### 1.5.3 Run Checkov

Chạy Checkov để quét code Terraform trong thư mục Lab02/Terraform/modules/main-modules nhằm kiểm tra tính tuân thủ và bảo mật. Bảo đảm rằng các cấu hình trong code Terraform đáp ứng các yêu cầu bảo mật, giúp giảm thiểu rủi ro bảo mật cho hạ tầng triển khai.

```
24      - name: "Run Checkov"  
25      run: checkov -f main.tf  
26      working-directory: Lab02/Terraform/modules/main-modules
```

Hình 7 – Run Checkov

“**checkov -f main.tf**” để quét mã trong file chính main.tf

### 1.5.4 Terraform Init

Khởi tạo Terraform bằng cách tải các plugin và thiết lập cấu hình trong thư mục Lab02/Terraform/modules/main-modules.

Chuẩn bị môi trường cho Terraform, bảo đảm rằng tất cả các plugin và module cần thiết đều sẵn sàng để sử dụng.

```
28      - name: Terraform Init  
29      run: terraform init  
30      working-directory: Lab02/Terraform/modules/main-modules
```

Hình 8 – Terraform Init

“**terraform init**” khởi tạo Terraform trong thư mục chính.

### 1.5.5 Terraform Format

Định dạng mã Terraform để bảo đảm tính nhất quán về mặt cú pháp.

```
32      - name: Terraform Format  
33      run: terraform fmt  
34      working-directory: Lab02/Terraform/modules/
```

Hình 9 – Terraform Format

“**terraform fmt**” giúp code Terraform dễ đọc, tuân thủ theo chuẩn định dạng, và làm giảm khả năng lỗi do định dạng không chuẩn.

### 1.5.6 Terraform Plan

Tạo một kế hoạch triển khai (plan) để xác định những thay đổi sẽ được áp dụng lên hạ tầng AWS. “**terraform plan**” cho phép xem trước các thay đổi trước khi thực sự áp dụng, giúp tránh các lỗi không mong muốn.

```
36      - name: Terraform Plan
37      env:
38          AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
39          AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
40      run: terraform plan -input=false
41      working-directory: Lab02/Terraform/modules/main-modules
```

Hình 10 – Terraform Plan

“**terraform plan -input=false**” kiểm tra kế hoạch triển khai mà không cần đầu vào từ người dùng. Các thông tin xác thực AWS được gọi từ secrets trong GitHub Actions.

### 1.5.7 Terraform Apply

Áp dụng các thay đổi đã xác nhận lên hạ tầng AWS. Triển khai hạ tầng như đã được định nghĩa trong mã Terraform.

“**-auto-approve**”: Việc áp dụng được tự động phê duyệt để không làm thủ công, với điều kiện push vào nhánh lab02/terra được thỏa.

“**terraform apply -auto-approve -input=false**” thực hiện áp dụng tự động với input mặc định.

```
43      - name: Terraform Apply
44          if: (github.ref == 'refs/heads/lab02/terra' && github.event_name == 'push')
45          env:
46              AWS_ACCESS_KEY_ID: ${{ secrets.AWS_ACCESS_KEY_ID }}
47              AWS_SECRET_ACCESS_KEY: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
48          run: terraform apply -auto-approve -input=false
49          working-directory: Lab02/Terraform/modules/main-modules
```

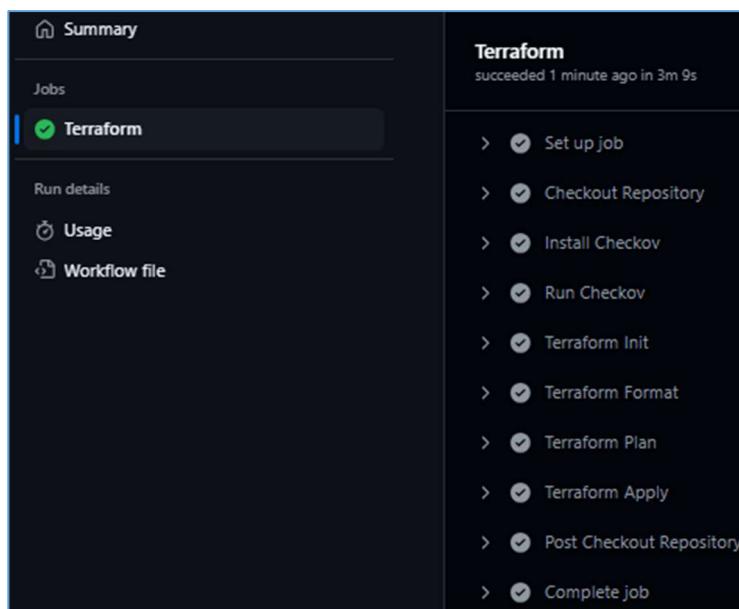
Hình 11 – Terraform Apply

## 1.6 Kiểm tra pipeline trên GitHub Actions

Sau khi các stage trong pipeline đã được cấu hình và kích hoạt trên GitHub Actions, thực hiện kiểm tra quá trình thực thi pipeline để xác minh rằng các tài nguyên trên AWS đã được triển khai thành công theo cấu hình Terraform, giúp phát hiện sớm vấn đề xảy ra trong quy trình tự động hóa.

### 1.6.1 Truy cập GitHub Actions và xem kết quả pipeline

Pipeline tự động kích hoạt khi có thay đổi mã nguồn trên nhánh lab02/terra. Truy cập vào Actions để xem trạng thái và chi tiết của mỗi workflow (thành công sẽ có màu xanh, lỗi sẽ là màu đỏ).



Hình 12 – Run pipeline thành công

### 1.6.2 Kiểm tra log và kết quả của từng bước

Mỗi bước trong pipeline có log chi tiết hiển thị các lệnh đã thực thi và thông báo lỗi. Log của Run Checkov giúp xác minh tính bảo mật của mã Terraform, log của Terraform Plan hiển thị các thay đổi dự kiến sẽ áp dụng lên AWS.

### 1.6.3 Xác minh trạng thái Terraform Apply

Terraform Apply là bước cuối để triển khai hạ tầng lên AWS. Kiểm tra log để đảm bảo tất cả tài nguyên đã được tạo hoặc cập nhật đúng theo cấu hình. Nếu thành công, tất cả tài nguyên như VPC, subnets, EC2 instances,... sẽ được triển khai.

```
Plan: 15 to add, 0 to change, 0 to destroy.  
module.vpc.aws_eip.nat: Creating...  
module.vpc.aws_vpc.vpc: Creating...  
module.vpc.aws_eip.nat: Creation complete after 1s [id=eipalloc-0275304c54564cb62]  
module.vpc.aws_vpc.vpc: Still creating... [10s elapsed]  
module.vpc.aws_vpc.vpc: Creation complete after 12s [id=vpc-02b8d3a48a78374d7]  
module.vpc.aws_internet_gateway.igw: Creating...  
module.vpc.aws_subnet.public_subnet: Creating...  
module.vpc.aws_subnet.private_subnet: Creating...  
module.vpc.aws_security_group.default_sg: Creating...  
module.vpc.aws_security_group.public_ec2_sg: Creating...  
module.vpc.aws_internet_gateway.igw: Creation complete after 0s [id=igw-0f0006d4920f6e8d5]  
module.vpc.aws_route_table.rtb_public: Creating...  
module.vpc.aws_subnet.private_subnet: Creation complete after 1s [id=subnet-01b91e95f78f64963]  
module.vpc.aws_route_table.rtb_public: Creation complete after 2s [id=rtb-07e665e28a661b24f]  
module.vpc.aws_security_group.default_sg: Creation complete after 2s [id=sg-0139208809c2f73ff3]  
module.vpc.aws_security_group.public_ec2_sg: Creation complete after 3s [id=sg-018480d74c91b144d]  
module.vpc.aws_security_group.private_ec2_sg: Creating...  
module.vpc.aws_security_group.private_ec2_sg: Creation complete after 2s [id=sg-008e33e838661bcd8]  
module.ec2.aws_instance.private_ec2_instace: Creating...  
module.vpc.aws_subnet.public_subnet: Still creating... [10s elapsed]  
module.vpc.aws_subnet.public_subnet: Creation complete after 11s [id=subnet-08dd52ab1d2f24140]  
module.vpc.aws_route_table_association.public-association: Creating...  
module.vpc.aws_nat_gateway.nat_gw: Creating...  
module.ec2.aws_instance.public_ec2_instace: Creating...  
module.vpc.aws_route_table_association.public-association: Creation complete after 1s [id=rtbassoc-062c038f6a6e34942]  
module.ec2.aws_instance.private_ec2_instace: Still creating... [10s elapsed]  
module.ec2.aws_instance.private_ec2_instace: Creation complete after 13s [id=i-0928806734ee3e98a]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [10s elapsed]  
module.ec2.aws_instance.public_ec2_instace: Still creating... [10s elapsed]  
module.ec2.aws_instance.public_ec2_instace: Creation complete after 13s [id=i-0c295e99df12f481b]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [20s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [30s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [40s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [50s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [1m0s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [1m10s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [1m20s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [1m30s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [1m40s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [1m50s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Still creating... [2m0s elapsed]  
module.vpc.aws_nat_gateway.nat_gw: Creation complete after 2m5s [id=nat-03cf0a3ee8124523b]  
module.vpc.aws_route_table.rtb_private: Creating...  
module.vpc.aws_route_table.rtb_private: Creation complete after 1s [id=rtb-0832edb67a014d77e]  
module.vpc.aws_route_table_association.private-association: Creating...  
module.vpc.aws_route_table_association.private-association: Creation complete after 0s [id=rtbassoc-085299b75bfh3998c]
```

Apply complete! Resources: 15 added, 0 changed, 0 destroyed.

Hình 13 – Thông tin chi tiết của hạ tầng

#### 1.6.4 Kiểm tra lại trên AWS

Thực hiện kiểm tra thủ công trên AWS Management Console, trực tiếp xác minh trên AWS để bảo đảm tất cả tài nguyên được triển khai đúng và hoạt động bình thường sau khi quá trình Terraform Apply hoàn tất.

#### 1.7 Kiểm tra hạ tầng AWS đã triển khai tự động lên AWS

Sau khi Terraform Apply hoàn tất và pipeline báo thành công, thực hiện kiểm tra thủ công trên AWS Management Console nhằm xác minh rằng các tài nguyên đã được tạo đúng theo cấu hình và hoạt động như mong đợi

##### 1.7.1 Kiểm tra VPC

Xác nhận rằng VPC chính đã được tạo với CIDR block và các thuộc tính phù hợp.

Your VPCs (5) Info						
<input type="text"/> Search						
<input type="checkbox"/>	Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	
<input type="checkbox"/>	nhom14-lab02-vpc	vpc-02b8d3a48a78374d7	<span>Available</span>	10.0.0.0/16	-	

Hình 14 – VPC (nhom4-lab02-vpc) và CIDR đã được tạo và ở trạng thái Available

### 1.7.2 Kiểm tra các Subnet

Bảo đảm rằng các subnet công khai và riêng tư đã được tạo trong VPC và nằm trong dải CIDR đúng.

Subnets (2) Info						
<input type="text"/> Find resources by attribute or tag						
<input type="checkbox"/>	Name	Subnet ID	State	VPC	IPv4 CIDR	
<input type="checkbox"/>	nhom14-lab02-private-subnet	subnet-01b91e95f78f64963	<span>Available</span>	vpc-02b8d3a48a78374d7   nhom14-lab02-vpc	10.0.2.0/24	
<input type="checkbox"/>	nhom14-lab02-public-subnet	subnet-08dd52ab1d2f24140	<span>Available</span>	vpc-02b8d3a48a78374d7   nhom14-lab02-vpc	10.0.1.0/24	

Hình 15 – Hai subnets và CIDR đều thuộc cùng một VPC và đang Available

### 1.7.3 Kiểm tra Internet Gateway và NAT Gateway

Bảo đảm rằng Internet Gateway và NAT Gateway đã được tạo và liên kết đúng với VPC.

Internet gateways (1) Info						
<input type="text"/> Find resources by attribute or tag						
<input type="checkbox"/>	Name	Internet gateway ID	State	VPC ID	Owner	
<input type="checkbox"/>	nhom14-lab02-igw	igw-0f0006d4920f6e8d5	<span>Attached</span>	vpc-02b8d3a48a78374d7   nhom14-la...	381492048687	

Hình 16 – Internet Gateway đã được liên kết với VPC

NAT gateways (1) Info						
<input type="text"/> Find resources by attribute or tag						
<input type="checkbox"/>	NAT gateway ID	Con...	State	Stat...	Primary pu...	VPC
<input type="checkbox"/>	nat-03cf0a3ee8124523b	Public	<span>Available.</span>	-	100.28.158.15	10.0.1.168 eni-0fd... vpc-02b8d3a48a78374d7 / nhom14-lab02-vpc subnet-08dd52ab1d2f24140 / nhom14-lab02-public-su...

Hình 17 – NAT Gateway đã được liên kết với VPC và gán với subnet “nhom4-lab02-public-subnet”

### 1.7.4 Kiểm tra Route Tables

Bảo đảm rằng các Route Tables đã được thiết lập đúng và gán vào các subnet thích hợp, cho phép kết nối giữa public và private subnet, cũng như truy cập internet từ subnet công khai.

<input type="checkbox"/>	nhom14-lab02-public-rtb	<a href="#">rtb-07e665e28a661b24f</a>	subnet-08dd52ab1d2f24140 / nhom14-lab02-public-subnet	-	No	vpc-02b8d3a48a78374d7   nhom14-lab02-vpc	381492048687
<input type="checkbox"/>	nhom14-lab02-private-rtb	<a href="#">rtb-0832edb67a014d77e</a>	subnet-01b91e95f78f64963 / nhom14-lab02-private-subnet	-	No	vpc-02b8d3a48a78374d7   nhom14-lab02-vpc	381492048687

Hình 18 – Hai Route Tables đã được gán với các subnet tương ứng trong VPC

### 1.7.5 Kiểm tra Security Groups

Xác nhận rằng các Security Groups được cấu hình đúng để bảo đảm an ninh cho các tài nguyên như EC2 instances.

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

Name	Security group ID	Security group name	VPC ID	Description	Owner	Inbound rules count	Outbound rules count
nhom14-lab02-private-sg	sg-008e33e838661bcd8	private_ec2_sg	vpc-02b8d3a48a78374d7	Private Security Group for EC2	381492048687	1 Permission entry	1 Permission entry
nhom14-lab02-default-sg	sg-013920809c2f73ff3	default_ec2_sg	vpc-02b8d3a48a78374d7	Default Security Group for EC2	381492048687	0 Permission entries	1 Permission entry
nhom14-lab02-public-sg	sg-018480d74c91b144d	public_ec2_sg	vpc-02b8d3a48a78374d7	Public Security Group for EC2	381492048687	1 Permission entry	1 Permission entry

Hình 19 – Các Security Groups với các description và inbound/outbound phù hợp

### 1.7.6 Kiểm tra EC2 Instances

Xác nhận rằng các EC2 instances public và private đã được tạo và chạy trong các subnet đúng.

Name	Instance ID	Instance state	Instanc...	Status check	Alarm status	Availabil...	Public IPv4 DNS	Public IPv4 ...
nhom14-lab02-public-ec2	i-0c295e99df12f481b	Running	t2.micro	2/2 checks passed..	View alarms +	us-east-1a	ec2-50-17-37-166.com...	50.17.37.166
nhom14-lab02-private-ec2	i-0928806734ee3e98a	Running	t2.micro	2/2 checks passed..	View alarms +	us-east-1a	-	-

Hình 20 – Hai EC2 instances đang chạy trong các subnet

## 2. Triển khai hạ tầng AWS với CloudFormation và tự động hóa quy trình build và deploy với AWS CodePipeline

### 2.1 Dùng CloudFormation để triển khai các dịch vụ AWS

**main.yml:** Là file CloudFormation tổng hợp, định nghĩa cấu trúc tổng thể của hạ tầng AWS, sử dụng nested task gọi vpc và ec2 từ file main để tạo tài nguyên:

- **VPCStack:** Stack con này tạo ra cấu trúc VPC bao gồm các subnet và route tables, sử dụng các thông tin từ vpc.yaml để cấu hình mạng.
- **EC2Stack:** Stack này bảo đảm các instance EC2 được triển khai trong các subnet thích hợp, với các thiết lập bảo mật phù hợp từ vpc.yaml.

```

1 AWSTemplateFormatVersion: "2010-09-09"
2 Description: Main Template
3
4 Parameters:
5   VpcCIDR:
6     Type: String
7     Default: "192.168.0.0/16"
8   PublicSubnetCIDR:
9     Type: String
10    Default: "192.168.1.0/24"
11   PrivateSubnetCIDR:
12     Type: String
13     Default: "192.168.2.0/24"
14   AvailabilityZone:
15     Type: String
16     Default: "us-east-1a"
17   BucketName:
18     Type: String
19     Default: "nhom14lab02s3bucket"
20
21 Resources:
22   VPCStack:
23     Type: AWS::CloudFormation::Stack
24     Properties:
25       TemplateURL: !Sub "https://${BucketName}.s3.us-east-1.amazonaws.com/nhom14/vpc.yaml"
26       Parameters:
27         VpcCIDR: !Ref VpcCIDR
28         PublicSubnetCIDR: !Ref PublicSubnetCIDR
29         PrivateSubnetCIDR: !Ref PrivateSubnetCIDR
30         AvailabilityZone: !Ref AvailabilityZone
31
32   EC2Stack:
33     Type: AWS::CloudFormation::Stack
34     Properties:
35       TemplateURL: !Sub "https://${BucketName}.s3.us-east-1.amazonaws.com/nhom14/ec2.yaml"
36       Parameters:
37         PublicSubnetId: !GetAtt VPCStack.Outputs.PublicSubnetId
38         PrivateSubnetId: !GetAtt VPCStack.Outputs.PrivateSubnetId
39         PublicEC2SecurityGroup: !GetAtt VPCStack.Outputs.PublicSgId
40         PrivateEC2SecurityGroup: !GetAtt VPCStack.Outputs.PrivateSgId
41
42     Outputs:
43       VPCId:
44         Value: !GetAtt VPCStack.Outputs.VPCId
45       PublicSubnetId:
46         Value: !GetAtt VPCStack.Outputs.PublicSubnetId
47       PrivateSubnetId:
48         Value: !GetAtt VPCStack.Outputs.PrivateSubnetId
49       PublicEC2InstanceId:
50         Value: !GetAtt EC2Stack.Outputs.PublicEC2InstanceId
51       PrivateEC2InstanceId:
52         Value: !GetAtt EC2Stack.Outputs.PrivateEC2InstanceId

```

Hình 21 – main.yaml

**ec2.yml:** Là file CloudFormation để định nghĩa các instance EC2, bao gồm một instance public và private nhằm cung cấp tài nguyên tính toán cho hạ tầng AWS, bao gồm:

- **PublicEC2Instance** Định nghĩa instance EC2 public, được triển khai trong subnet public với security group phù hợp. Cho phép truy cập từ bên ngoài và được gắn thẻ tên để dễ dàng quản lý.
- **PrivateEC2Instance** Định nghĩa instance EC2 private, triển khai trong subnet private với security group nội bộ, chỉ cho phép truy cập từ các tài nguyên đáng tin cậy bên trong VPC.

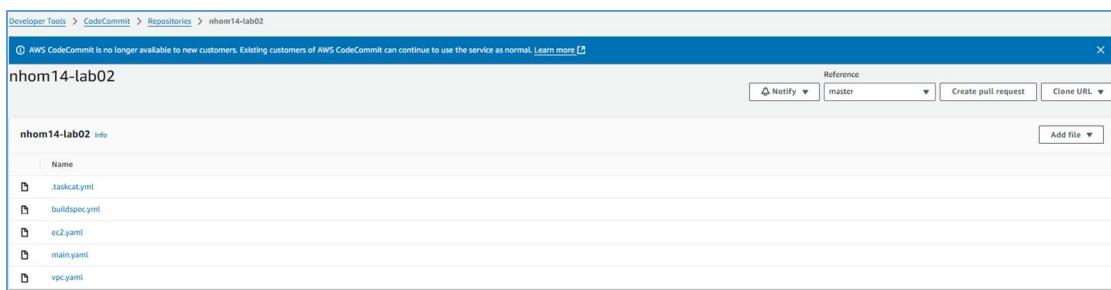
**vpc.yml:** File này định nghĩa cấu hình các thành phần mạng liên quan, giúp xây dựng kiến trúc mạng cơ bản cho hạ tầng AWS, bao gồm VPC, Subnet, Route Tables, Internet Gateway, NAT Gateway, Security Group.

Cách cấu hình chi tiết cho từng file đã thực hiện ở Lab01.

## 2.2 Sử dụng AWS CodeBuild, tích hợp cfn-lint và Taskcat

### 2.2.1 CodeCommit

Sử dụng CodeCommit nhằm quản lý mã nguồn của AWS, nơi lưu trữ các file CloudFormation để dễ dàng kiểm soát các thay đổi.



Hình 22 – Repository của nhóm trên AWS CodeCommit chứa các file cấu hình

### 2.2.2 CodeBuild

- **Cấu hình CodeBuild tích hợp Taskcat và cfn-lint**

**taskcat.yaml:** Chứa cấu hình cho Taskcat, công cụ kiểm tra tính đúng đắn của các file CloudFormation trong môi trường giả lập.

```

1   project:
2       name: nhom14
3       regions:
4           - us-east-1
5       s3_bucket: nhom14lab02s3bucket
6
7       tests:
8           my-test:
9               template: ./main.yaml

```

Hình 23 – taskcat.yaml

Định nghĩa các regions (vùng) để kiểm thử cấu hình trên nhiều vùng AWS khác nhau, đảm bảo tính tương thích và sẵn sàng của các cấu hình trong mọi khu vực.

**buildspec.yml:** Là file cấu hình dành cho CodeBuild, xác định các lệnh và quy trình mà CodeBuild sẽ thực hiện khi xây dựng và kiểm tra code.

```
1  version: 0.2
2
3  phases:
4      install:
5          runtime-versions:
6              python: 3.x
7          commands:
8              - pip install cfn-lint taskcat
9      pre_build:
10         commands:
11             - echo "Running cfn-lint..."
12             - cfn-lint main.yaml
13             - cfn-lint ec2.yaml
14             - cfn-lint vpc.yaml
15      build:
16         commands:
17             - echo "Running Taskcat..."
18             - taskcat test run
```

Hình 24 – buildspec.yml

**“phases”:** Các giai đoạn của quá trình build, bao gồm cài đặt “*install*”, “*pre\_build*” và “*build*”.

“*install*”: Cài đặt các công cụ cfn-lint và taskcat cần thiết.

“*pre\_build*”: Chạy cfn-lint với để kiểm tra cú pháp các file main.yaml, ec2.yaml, và vpc.yaml.

“*build*”: Chạy taskcat test run để thực hiện kiểm thử các tệp.

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

```

Showing the last 561 lines of the build log. View entire log
No previous logs

1 [Container] 2024/10/29 13:46:51.437803 Running on CodeBuild On-demand
2 [Container] 2024/10/29 13:46:51.437865 Waiting for agent ping
3 [Container] 2024/10/29 13:46:51.639354 Waiting for DOWNLOAD_SOURCE
4 [Container] 2024/10/29 13:46:51.810685 Phase is DOWNLOAD_SOURCE
5 [Container] 2024/10/29 13:46:51.142560 CODEBUILD_SRC_DIR=/codebuild/output/src41292860/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/nhom14-lab02
6 [Container] 2024/10/29 13:46:51.143097 YAML location is /codebuild/output/src41292860/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/nhom14-lab02/buildspec.yml
7 [Container] 2024/10/29 13:46:51.144679 Not setting HTTP client timeout for source type codecommit
8 [Container] 2024/10/29 13:46:51.144881 Processing environment variables
9 [Container] 2024/10/29 13:46:51.377415 Resolved 'python' runtime alias '3.x' to '3.12'.
10 [Container] 2024/10/29 13:46:51.377445 Selecting 'python' runtime version '3.12' based on manual selections...
11 [Container] 2024/10/29 13:46:51.377992 Running command echo "Installing Python version 3.12 ..."
12 Installing Python version 3.12 ...
13
14 [Container] 2024/10/29 13:46:51.385143 Running command pyenv global $PYTHON_312_VERSION
15
16 [Container] 2024/10/29 13:46:51.029163 Moving to directory /codebuild/output/src41292860/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/nhom14-lab02
17 [Container] 2024/10/29 13:46:51.036862 Unable to initialize cache download: no paths specified to be cached
18 [Container] 2024/10/29 13:46:51.178148 Configuring ssh agent with target id: codebuild:d7fc72b2-7002-4c32-8dfa-64b9483adcd4
19 [Container] 2024/10/29 13:46:51.217916 Successfully updated ssh agent configuration
20 [Container] 2024/10/29 13:46:51.218284 Registering with agent
21 [Container] 2024/10/29 13:46:51.251475 Phase found in YAML: 3
22 [Container] 2024/10/29 13:46:51.251497 BUILD: 2 commands
23 [Container] 2024/10/29 13:46:51.251502 INSTALL: 1 commands
24 [Container] 2024/10/29 13:46:51.251508 PRE_BUILD: 4 commands
25 [Container] 2024/10/29 13:46:51.251786 Phase complete: DOWNLOAD_SOURCE State: SUCCEEDED
26 [Container] 2024/10/29 13:46:51.251799 Phase context status code: Message:
27 [Container] 2024/10/29 13:46:51.321439 Entering phase INSTALL
28 [Container] 2024/10/29 13:46:51.355462 Running command pip install cfn-lint taskcat
29 Collecting cfn-lint
30   Downloading cfn_lint-1.18.2-py3-none-any.whl.metadata (22 kB)
31 Collecting taskcat
32   Downloading taskcat-0.9.54-py3-none-any.whl.metadata (1.1 kB)
33 Requirement already satisfied: pyyaml<5.4 in /root/.pyenv/versions/3.12.4/lib/python3.12/site-packages (from cfn-lint) (5.4.1)
34 Collecting aws-sam-translator>=1.91.0 (from cfn-lint)
35   Downloading aws_sam_translator-1.91.0-py3-none-any.whl.metadata (8.3 kB)
36 Collecting jsonpatch
37   Downloading jsonpatch-1.33-py2.py3-none-any.whl.metadata (3.0 kB)
38 Collecting networkx>4.0
39   Downloading networkx-3.4.2-py3-none-any.whl.metadata (6.3 kB)
40 Collecting sympy>=1.0.0 (from cfn-lint)
41   Downloading sympy-1.13.3-py3-none-any.whl.metadata (12 kB)
42 Collecting regex (from cfn-lint)
43   Downloading regex-2024.9.11-cp312-cp312-manylinux2_17_x86_64_manylinux2014_x86_64.whl.metadata (40 kB)
44   _____ 48.5/40.5 kB 6.8 MB/s eta 0:00:00
45 Collecting typing_extensions (from cfn-lint)
46   Downloading typing_extensions-4.12.2-py3-none-any.whl.metadata (3.0 kB)
47 Collecting GitPython>=3.1.43 (from taskcat)
48   Downloading GitPython-3.1.43-py3-none-any.whl.metadata (13 kB)
49 Collecting pyyaml>5.4 (from cfn-lint)
50   Downloading PyYAML-6.0.2-cp312-cp312-manylinux2_17_x86_64_manylinux2014_x86_64.whl.metadata (2.1 kB)
51 Requirement already satisfied: boto<3.2.0,>=1.9.21 in /root/.pyenv/versions/3.12.4/lib/python3.12/site-packages (from taskcat) (1.35.46)
52 Requirement already satisfied: botocore<2.0,>=1.12.21 in /root/.pyenv/versions/3.12.4/lib/python3.12/site-packages (from taskcat) (1.35.46)
53 Collecting cfn-lint
54   Downloading cfn_lint-0.87.11-py3-none-any.whl.metadata (16 kB)

```

Hình 25 – Run CodeBuild

- CodeBuild tải code từ repository trên CodeCommit, sau đó cài đặt môi trường và các công cụ cần thiết như Python 3.12, cfn-lint, và Taskcat. Quá trình cài đặt Python phiên bản 3.12 và các gói cần thiết được cấu hình qua buildspec.yml.

```

195
196 [Container] 2024/11/02 13:17:28.883632 Phase complete: INSTALL State: SUCCEEDED
197 [Container] 2024/11/02 13:17:28.883653 Phase context status code: Message:
198 [Container] 2024/11/02 13:17:28.922426 Entering phase PRE_BUILD
199 [Container] 2024/11/02 13:17:28.923541 Running command echo "Running cfn-lint..."
200 Running cfn-lint...
201
202 [Container] 2024/11/02 13:17:28.930867 Running command cfn-lint main.yaml
203
204 [Container] 2024/11/02 13:17:30.184643 Running command cfn-lint ec2.yaml
205
206 [Container] 2024/11/02 13:17:31.434841 Running command cfn-lint vpc.yaml
207
208 [Container] 2024/11/02 13:17:32.787414 Phase complete: PRE_BUILD State: SUCCEEDED
209 [Container] 2024/11/02 13:17:32.787436 Phase context status code: Message:
210 [Container] 2024/11/02 13:17:32.824177 Entering phase BUILD
211 [Container] 2024/11/02 13:17:32.825137 Running command echo "Running Taskcat..."
212 Running Taskcat...
213

```

Hình 26 – Check cfn-lint

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

```

215 [Container] 2024/10/29 13:47:42.671304 Phase complete: INSTALL State: SUCCEEDED
216 [Container] 2024/10/29 13:47:42.671319 Phase context status code: Message:
217 [Container] 2024/10/29 13:47:42.706581 Entering phase PRE_BUILD
218 [Container] 2024/10/29 13:47:42.707435 Running command echo "Running cfn-lint..."
219 Running cfn-lint...
220
221 [Container] 2024/10/29 13:47:42.712634 Running command cfn-lint main.yaml
222
223 [Container] 2024/10/29 13:47:45.354528 Running command cfn-lint ec2.yaml
224
225 [Container] 2024/10/29 13:47:46.962938 Running command cfn-lint vpc.yaml
226
227 [Container] 2024/10/29 13:47:48.477043 Phase complete: PRE_BUILD State: SUCCEEDED
228 [Container] 2024/10/29 13:47:48.477066 Phase context status code: Message:
229 [Container] 2024/10/29 13:47:48.509628 Entering phase BUILD
230 [Container] 2024/10/29 13:47:48.510471 Running command echo "Running Taskcat..."
231 Running Taskcat...
232
233 [Container] 2024/10/29 13:47:48.516089 Running command taskcat test run
234
235 [Container] 2024/10/29 13:47:48.516100 Taskcat test run completed successfully.
236 |   |
237 |   \--> Taskcat test run completed successfully.
238 |
239
240
241
242 version 0.9.54
243 Not in terminal, reprint now using normal build-in print function.
244
245 [INFO ] : Linting passed for file: /codebuild/output/src41292860/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/nhom14-lab02/ec2.yaml
246 [INFO ] : Linting passed for file: /codebuild/output/src41292860/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/nhom14-lab02/main.yaml
247 [INFO ] : Linting passed for file: /codebuild/output/src41292860/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/nhom14-lab02/vpc.yaml
248 [S3: -> ] s3://nhom14lab02s3bucket/nhom14/buildspec.yml
249 [S3: -> ] s3://nhom14lab02s3bucket/nhom14/ec2.yaml
250 [S3: -> ] s3://nhom14lab02s3bucket/nhom14/main.yaml
251 [S3: -> ] s3://nhom14lab02s3bucket/nhom14/vpc.yaml
252     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539
253     region: us-east-1
254     status: CREATE_IN_PROGRESS
255     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539
256     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539-VPCStack-108HB8B2RDSCN
257     region: us-east-1
258     status: CREATE_IN_PROGRESS
259     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539
260     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539-VPCStack-108HB8B2RDSCN
261     region: us-east-1
262     status: CREATE_IN_PROGRESS
263     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539
264     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539-VPCStack-108HB8B2RDSCN
265     region: us-east-1
266     status: CREATE_IN_PROGRESS
267     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539
268     stack @ tCat-nhom14-my-test-6ade6e67c1c04acc82880d3c0ed2f539-VPCStack-108HB8B2RDSCN
269     region: us-east-1
270     status: CREATE_IN_PROGRESS

```

Hình 27 – Run Taskcat

- Sau khi vượt qua các kiểm tra cú pháp, CodeBuild chạy Taskcat để kiểm tra tính đúng đắn của các cấu hình CloudFormation trong môi trường giả lập trên AWS. Lệnh “taskcat test run” được sử dụng để thực hiện các kiểm tra này.

```

546 [Container] 2024/10/29 13:58:05.315288 Phase complete: BUILD State: SUCCEEDED
547 [Container] 2024/10/29 13:58:05.315319 Phase context status code: Message:
548 [Container] 2024/10/29 13:58:05.363247 Entering phase POST_BUILD
549 [Container] 2024/10/29 13:58:05.365358 Phase complete: POST_BUILD State: SUCCEEDED
550 [Container] 2024/10/29 13:58:05.365376 Phase context status code: Message:
551 [Container] 2024/10/29 13:58:05.419636 Set report auto-discover timeout to 5 seconds
552 [Container] 2024/10/29 13:58:05.419703 Expanding base directory path: .
553 [Container] 2024/10/29 13:58:05.422820 Assembling file list
554 [Container] 2024/10/29 13:58:05.422835 Expanding .
555 [Container] 2024/10/29 13:58:05.426004 Expanding file paths for base directory .
556 [Container] 2024/10/29 13:58:05.426018 Assembling file list
557 [Container] 2024/10/29 13:58:05.426021 Expanding **/*
558 [Container] 2024/10/29 13:58:05.429480 No matching auto-discover report paths found
559 [Container] 2024/10/29 13:58:05.429586 Report auto-discover file discovery took 0.009949 seconds
560 [Container] 2024/10/29 13:58:05.429602 Phase complete: UPLOAD_ARTIFACTS State: SUCCEEDED
561 [Container] 2024/10/29 13:58:05.429609 Phase context status code: Message:
562

```

Hình 28 – Taskcat build thành công

- Tất cả các file cấu hình (ec2.yaml, vpc.yaml, main.yaml) đều vượt qua kiểm tra mà không có lỗi cấu hình. Các stack đã được tạo trong môi trường giả lập mà không gặp vấn đề gì.

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

The screenshot shows the AWS CodeBuild console. At the top, there's a navigation bar with 'Developer Tools > CodeBuild > Build projects > nhom14-lab02-codebuild'. Below the navigation is a toolbar with 'Actions', 'Create trigger', 'Edit', 'Clone', 'Debug build', 'Start build with overrides', and a prominent orange 'Start build' button. The main area is titled 'Configuration' and contains fields for 'Source provider' (AWS CodeCommit), 'Primary repository' (nhom14-lab02), 'Artifacts upload location' (empty), and 'Service role' (arn:aws:iam::767397858864:role/service-role/codebuild-nhom14-lab02-codebuild-service-role). Under 'Public builds', it says 'Disabled'. Below the configuration is a tab bar with 'Build history', 'Batch history', 'Project details', 'Build triggers', and 'Metrics'. The 'Build history' tab is selected, showing a table of build runs. One run is listed: 'nhom14-lab02-codebuild:0020143a-e53b-4602-893f-de67aae4c20e' with status 'Succeeded', build number 7, source version 'arn:aws:s3:::codepipeli ne-us-east-1-572358130979/nhom14-lab02-pipeline/SourceArti/60yKWNs', submitter 'codepipeline/nhom14-lab02-pipeline', duration '7 minutes 12 seconds', and completed '20 minutes ago'. There are also buttons for 'Stop build', 'View artifacts', 'View logs', 'Delete builds', and 'Retry build'.

Hình 29 – Kiểm tra CodeBuild trên AWS

The screenshot shows the AWS CodeBuild console with the URL 'nhom14-lab02-codebuild:b5d26a73-742a-4b31-816b-afed32935000'. It displays the 'Build status' section with details like status 'Succeeded', initiator 'cloud\_user', start time 'Nov 2, 2024 8:17 PM (UTC+7:00)', end time 'Nov 2, 2024 8:23 PM (UTC+7:00)', and build number '2'. Below this is a table of 'Phase details' with columns: Name, Status, Context, Duration, Start time, and End time. The phases listed are SUBMITTED, QUEUED, PROVISIONING, DOWNLOAD\_SOURCE, INSTALL, PRE\_BUILD, BUILD, POST\_BUILD, UPLOAD\_ARTIFACTS, FINALIZING, and COMPLETED, all showing a successful status and short durations.

Hình 30 – Phase details của CodeBuild

- Phase details cho thấy quy trình build gồm các bước:
  - SUBMITTED, QUEUED, PROVISIONING: Các bước chuẩn bị ban đầu.
  - DOWNLOAD\_SOURCE: Tải code từ CodeCommit.
  - INSTALL: Cài đặt môi trường và các công cụ kiểm tra.
  - PRE\_BUILD: Thực hiện kiểm tra cfn-lint.
  - BUILD: Chạy Taskcat để xác minh cấu hình.
  - POST\_BUILD và UPLOAD\_ARTIFACTS: Hoàn tất quá trình và tải lên các artifact.

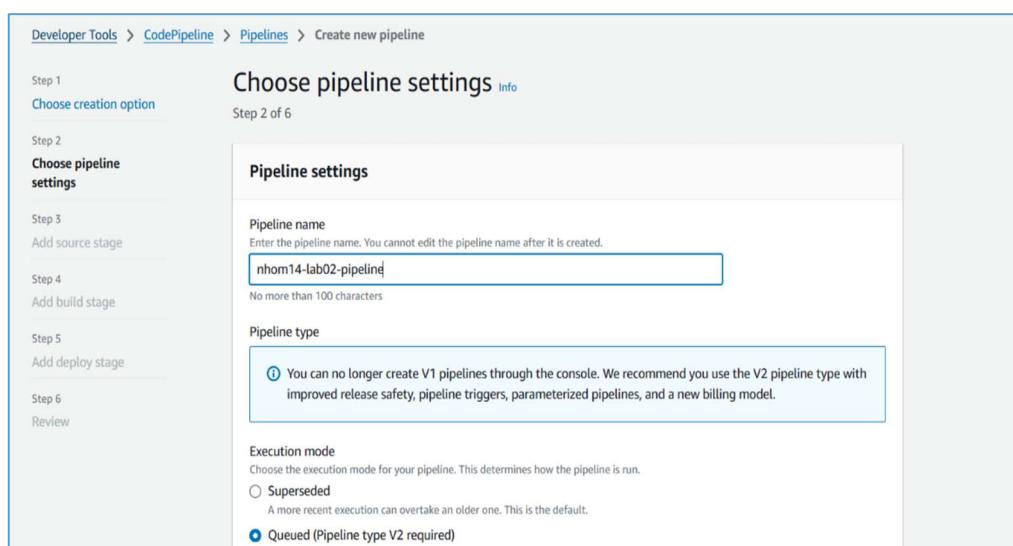
Mỗi giai đoạn đều có trạng thái "Succeeded", xác nhận rằng các bước trong quy trình build đã hoàn thành thành công.

### 2.3 Sử dụng AWS CodePipeline để tự động hóa quy trình build và deploy từ mã nguồn

Pipeline được thiết kế nhằm tự động hóa quy trình triển khai code từ source đến triển khai trên môi trường AWS. Mục tiêu là đảm bảo quá trình CI/CD diễn ra trơn tru, từ việc lấy mã nguồn, kiểm tra và build mã, cho đến việc triển khai hạ tầng.

- Các bước cấu hình:

- Pipeline này sử dụng phiên bản V2, kích hoạt pipeline, và các tùy chọn khác để kiểm soát việc thực thi tuần tự các tác vụ.



Hình 31 – Step 2: Choose Pipeline Settings

- AWS CodeCommit được chọn làm nguồn, và nhánh master trong repository nhom14-lab02 là nguồn cung cấp mã nguồn. Bất kỳ thay đổi nào trên nhánh này sẽ kích hoạt pipeline thông qua Amazon CloudWatch Events.

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1 Choose creation option

Step 2 Choose pipeline settings

Step 3 Add source stage

Step 4 Add build stage

Step 5 Add deploy stage

Step 6 Review

Add source stage Info Step 3 of 6

**Source**

Source provider This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

AWS CodeCommit

Repository name Choose a repository that you have already created where you have pushed your source code.

Q nhom14-lab02 X

Branch name Choose a branch of the repository

Q master X

Change detection options Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

Amazon CloudWatch Events (recommended) Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs

AWS CodePipeline Use AWS CodePipeline to check periodically for changes

Output artifact format Choose the output artifact format.

CodePipeline default AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.

Full clone AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild

Hình 32 – Step 3: Add Source Stage

- AWS CodeBuild làm công cụ build. Pipeline sẽ thực hiện một quá trình single build dựa trên các thiết lập đã cấu hình trước đó trong CodeBuild. Đây là nơi code sẽ được build và kiểm tra trước khi triển khai.

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1 Choose creation option

Step 2 Choose pipeline settings

Step 3 Add source stage

Step 4 Add build stage

Step 5 Add deploy stage

Step 6 Review

Add build stage Info Step 4 of 6

**Build - optional**

Build provider Choose the tool you want to use to run build commands and specify artifacts for your build action.

Commands

Other build providers

AWS CodeBuild

Project name Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.

Q nhom14-lab02-codebuild X or Create project □

Environment variables - optional Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. Learn more □

Add environment variable

Build type

Single build Triggers a single build.

Batch build Triggers multiple builds as a single execution.

Region US East (N. Virginia)

Hình 33 – Step 4: Add Build Stage

- AWS CloudFormation được chọn làm công cụ triển khai. File template main.yml được sử dụng để tạo hoặc cập nhật stack có tên nhom14-stack. Cho phép triển khai cơ sở hạ tầng dựa trên các cấu hình trong file main.yml, với quyền EC2CloudFormationRole được gán để quản lý các tài nguyên liên quan.

**Choose pipeline settings**

**Deploy - optional**

**Deploy provider**  
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.

AWS CloudFormation

**Region**  
US East (N. Virginia)

**Input artifacts**  
Choose an input artifact for this action. [Learn more](#)

BuildArtifact X  
Defined by: Build

No more than 100 characters

**Action mode**  
When you update an existing stack, the update is permanent. When you use a change set, the result provides a diff of the updated stack and the original stack before you choose to execute the change.

Create or update a stack

**Stack name**  
If you are updating an existing stack, choose the stack name.

nhom14-stack

**Template**  
Specify the template you uploaded to your source location.

Artifact name	File name	Template file path
BuildArtifact	main.yaml	BuildArtifact::main.yaml

**Template configuration - optional**  
Specify the configuration file you uploaded to your source location.

Use configuration file

Artifact name	File name	Template configuration file path

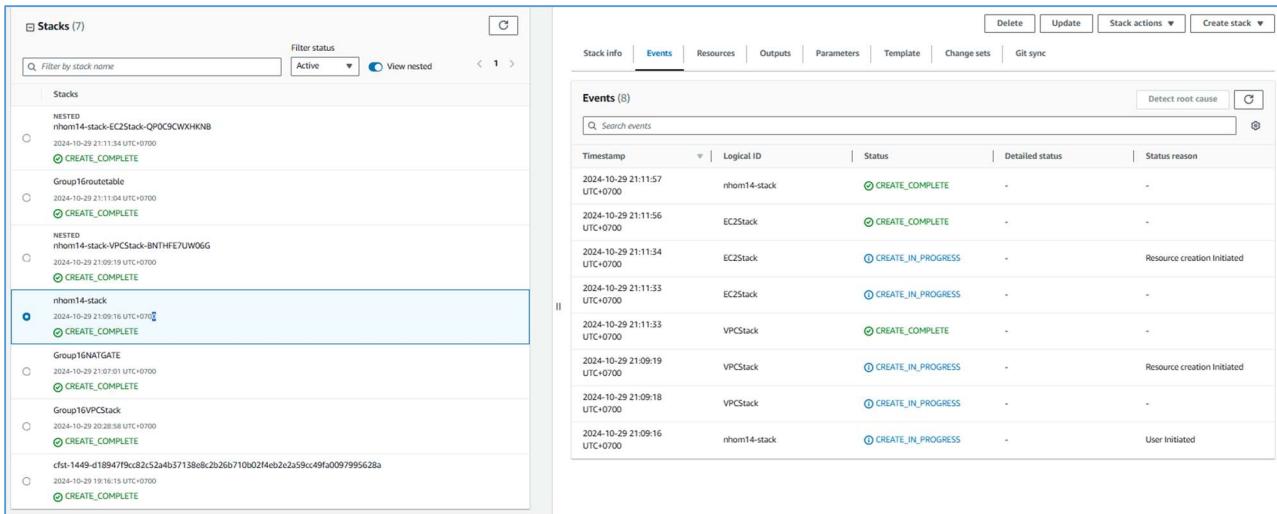
**Capabilities - optional**  
Specify whether you want to allow AWS CloudFormation to create IAM resources on your behalf.

**Role name**

arn:aws:iam::767397858864:role/EC2CloudFormationRole

Hình 34 – Step 5: Add Deploy Stage

- CloudFormation Stack được tạo tự động từ pipeline. Các stack con được tạo thành công, thể hiện qua trạng thái CREATE\_COMPLETE, xác nhận rằng tất cả các tài nguyên trong quy trình triển khai đã được thiết lập hoàn chỉnh và sẵn sàng sử dụng.



Hình 35 – Tạo Stack trong AWS CloudFormation

- Hình ảnh tổng quan sự kiện khởi tạo tài nguyên trong VPC Stack (NATGateway, PrivateRoute và các bảng định tuyến) và các instance (Public và Private) trong EC2 Stack. Trạng thái CREATE-COMPLETE cho thấy tất cả các thành phần đã hoàn tất quá trình khởi tạo thành công, bảo đảm rằng cơ sở hạ tầng được triển khai đúng.

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

21

nhom14-stack-VPCStack-BNTHFE7UW06G					
NESTED					
		Events		Resources	
Stack info	Events	Resources	Outputs	Parameters	Template
Change sets	Git sync				
<b>Events (52)</b>					
<input type="text"/> Search events					
Timestamp	Logical ID	Status	Detailed status	Status reason	
2024-10-29 21:11:23 UTC+0700	nhom14-stack-VPCStack-BNTHFE7UW06G	CREATE_COMPLETE	-	-	
2024-10-29 21:11:22 UTC+0700	PrivateRoute	CREATE_COMPLETE	-	-	
2024-10-29 21:11:21 UTC+0700	PrivateRoute	CREATE_IN_PROGRESS	-	Resource creation Initiated	
2024-10-29 21:11:20 UTC+0700	PrivateRoute	CREATE_IN_PROGRESS	-	-	
2024-10-29 21:11:20 UTC+0700	NatGateway	CREATE_COMPLETE	-	-	
2024-10-29 21:09:53 UTC+0700	PrivateEC2SecurityGroup	CREATE_COMPLETE	-	-	
2024-10-29 21:09:49 UTC+0700	NatGateway	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated	
2024-10-29 21:09:48 UTC+0700	PublicRoute	CREATE_COMPLETE	-	-	
2024-10-29 21:09:48 UTC+0700	PublicRoute	CREATE_IN_PROGRESS	-	Resource creation Initiated	
2024-10-29 21:09:47 UTC+0700	PublicRoute	CREATE_IN_PROGRESS	-	-	
2024-10-29 21:09:46 UTC+0700	PrivateEC2SecurityGroup	CREATE_IN_PROGRESS	-	Resource creation Initiated	
2024-10-29 21:09:46 UTC+0700	PublicRouteTable	CREATE_COMPLETE	-	-	
2024-10-29 21:09:46 UTC+0700	PrivateRouteTable	CREATE_COMPLETE	-	-	

Hình 36 – Chi tiết sự kiện VPC Stack

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

Events (9)					
Timestamp		Logical ID	Status	Detailed status	Status reason
2024-10-29 21:11:51 UTC+0700		nhom14-stack-EC2Stack-QP0C9CWXHKNB	CREATE_COMPLETE	-	-
2024-10-29 21:11:50 UTC+0700		PrivateEC2Instance	CREATE_COMPLETE	-	-
2024-10-29 21:11:49 UTC+0700		PublicEC2Instance	CREATE_COMPLETE	-	-
2024-10-29 21:11:40 UTC+0700		PublicEC2Instance	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated
2024-10-29 21:11:38 UTC+0700		PrivateEC2Instance	CREATE_IN_PROGRESS	-	Resource creation Initiated
2024-10-29 21:11:38 UTC+0700		PublicEC2Instance	CREATE_IN_PROGRESS	-	Resource creation Initiated
2024-10-29 21:11:36 UTC+0700		PrivateEC2Instance	CREATE_IN_PROGRESS	-	-
2024-10-29 21:11:36 UTC+0700		PublicEC2Instance	CREATE_IN_PROGRESS	-	-
2024-10-29 21:11:34 UTC+0700		nhom14-stack-EC2Stack-QP0C9CWXHKNB	CREATE_IN_PROGRESS	-	User Initiated

Hình 37 – Chi tiết sự kiện EC2 Stack

- Outputs của VPC Stack hiển thị các giá trị quan trọng đã được tạo và sẵn sàng để tham chiếu trong các phần khác của cấu trúc hạ tầng, bao gồm:
  - PrivateSgId và PublicSgId: ID của các Security Group tương ứng cho subnet private và public.
  - PrivateSubnetId và PublicSubnetId: ID của subnet private và public.
  - VPCId: ID của VPC, cung cấp cơ sở hạ tầng mạng cho toàn bộ stack.

Outputs (5)					
Key	Value	Description	Export name		
PrivateSgId	sg-00044d0769d7b44a	-	-		
PrivateSubnetId	subnet-05a4d02051a6b5c1e	-	-		
PublicSgId	sg-08e84b279edd8d0c0	-	-		
PublicSubnetId	subnet-073606b4929d6d3db	-	-		
VPCId	vpc-0576b4cefc931f401	-	-		

Hình 38 – Outputs của VPC Stack

- Outputs của EC2 Stack hiển thị thông tin về các instance EC2 đã được tạo ra:

- PrivateEC2InstanceId: ID của instance EC2 trong subnet private.
- PublicEC2InstanceId: ID của instance EC2 trong subnet public.

Key	Value	Description	Export name
PrivateEC2InstanceId	i-08d02074116adec51	-	-
PublicEC2InstanceId	i-00908dd9317fe8daf	-	-

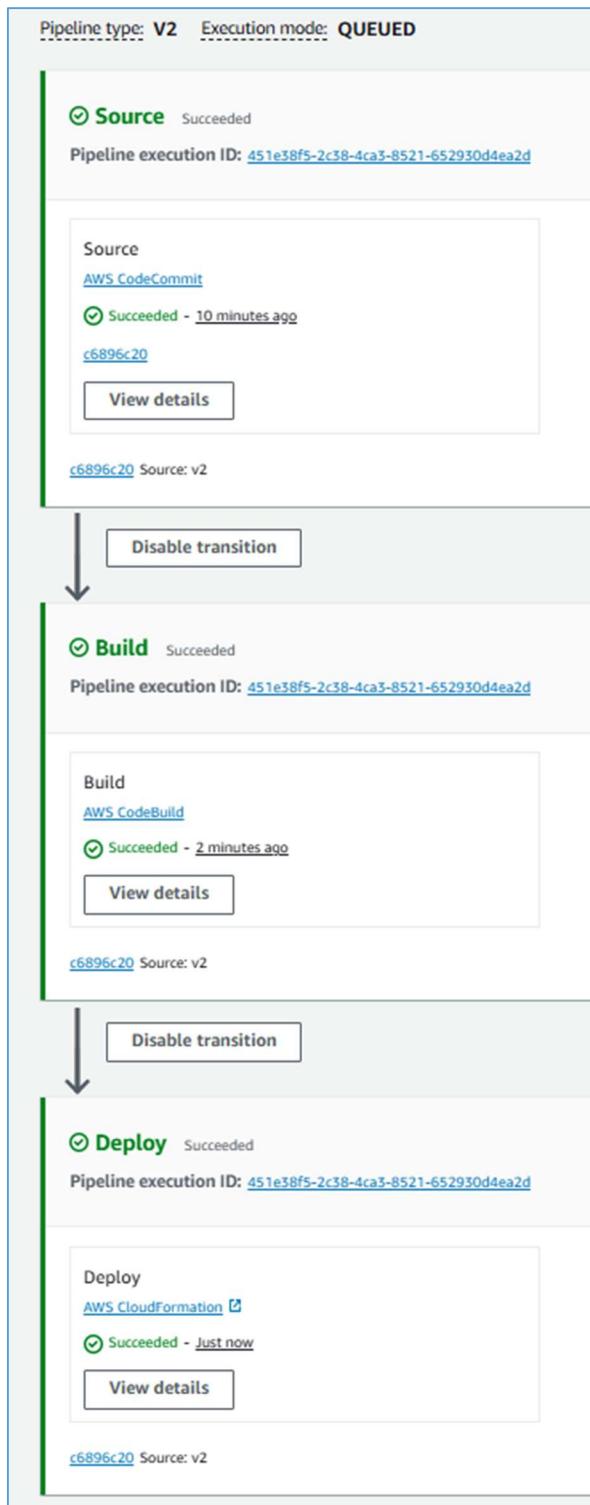
Hình 39 – Outputs của EC2 Stack

- Các log của quá trình thực thi trong bước triển khai của pipeline. Các tài nguyên như nhom14-stack, VPCStack, và EC2Stack được tạo tuân tự. Status hiển thị trạng thái Create in progress khi các tài nguyên đang được khởi tạo và Create complete khi chúng hoàn tất.

Timestamp	Logical ID	Status	Status reason
Tue Oct 29 2024 21:11:57...	nhom14-stack	>Create complete	-
Tue Oct 29 2024 21:11:56...	EC2Stack	>Create complete	-
Tue Oct 29 2024 21:11:54...	EC2Stack	>Create in progress	-
Tue Oct 29 2024 21:11:53...	EC2Stack	Create in progress	Resource creation initiated
Tue Oct 29 2024 21:11:53...	VPCStack	>Create complete	-
Tue Oct 29 2024 21:09:19...	VPCStack	>Create in progress	Resource creation initiated
Tue Oct 29 2024 21:09:18...	VPCStack	Create in progress	-
Tue Oct 29 2024 21:09:16...	nhom14-stack	Create in progress	User Initiated

Hình 40 – Logs của quá trình triển khai các stack trong AWS CodePipeline

- Các giai đoạn trong quy trình tự động hóa AWS CodePipeline ở trạng thái Succeeded:
  - Source: Sử dụng AWS CodeCommit để lấy mã nguồn từ kho lưu trữ. Và mã nguồn đã được lấy thành công.
  - Build: Sử dụng AWS CodeBuild để xây dựng và kiểm tra mã nguồn. Tất cả các kiểm thử và xây dựng mã nguồn đã hoàn tất thành công, đảm bảo mã nguồn sẵn sàng cho giai đoạn triển khai tiếp theo.
  - Deploy: Sử dụng AWS CloudFormation trong giai đoạn triển khai. Mã nguồn đã được triển khai thành công lên môi trường đích, xác nhận rằng tất cả cấu trúc hạ tầng được tạo và cấu hình đúng cách.



Hình 41 – Trạng thái thành công của các giai đoạn trong AWS CodePipeline

- Kiểm tra tài nguyên trên AWS
  - Public EC2 instance: Các thông tin chi tiết về “**Lab1\_Nhom14\_PublicEC2Instance**” (như Instance ID, IP Public, VPC, Subnet ID,...). Đây là một máy chủ public, cho phép truy cập trực tiếp từ internet qua IP public đã chỉ định.
  - Private EC2 instance: Các thông tin chi tiết về “**Lab1\_Nhom14\_PrivateEC2Instance**” (như Instance ID, IP Private, VPC, Subnet

## Bài thực hành 02: Quản lý và triển khai hạ tầng AWS và ứng dụng microservices với Terraform, CloudFormation, GitHub Actions, AWS CodePipeline và Jenkins

ID,...). Đây là một máy chủ private, nó không có địa chỉ IP public và không thể truy cập trực tiếp từ internet.

The screenshot shows the AWS EC2 Instances page for a specific instance. The instance summary for 'i-00908dd9317fe8daf (Lab1\_Nhom14\_PublicEC2Instance)' is displayed. Key details include:

- Public IPv4 address:** 54.88.10.125 [open address]
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-192-168-1-88.ec2.internal
- Instance type:** t2.micro
- VPC ID:** vpc-0576b4cef931f401 (Lab1\_Nhom14\_VPC)
- Subnet ID:** subnet-073606b4929d6d3db (Lab1\_Nhom14\_PublicSubnet)
- Instance ARN:** arn:aws:ec2:us-east-1:851725517215:instance/i-00908dd9317fe8daf
- Tags:**
  - aws:cloudformation:logical-id: PublicEC2Instance
  - aws:cloudformation:stack-id: arn:aws:cloudformation:us-east-1:851725517215:stack/nhom14-stack-EC2Stack-QP0C9CWXHKNB/b42c5ca0-95ff-11ef-9f71-0affc957c93d
  - aws:cloudformation:stack-name: nhom14-stack-EC2Stack-QP0C9CWXHKNB
  - Name: Lab1\_Nhom14\_PublicEC2Instance

Hình 42 – Thông tin chi tiết Public EC2 Instance

The screenshot shows the AWS EC2 Instances page for a specific instance. The instance summary for 'i-08d02074116adec51 (Lab1\_Nhom14\_PrivateEC2Instance)' is displayed. Key details include:

- Public IPv4 address:** –
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-192-168-2-17.ec2.internal
- Instance type:** t2.micro
- VPC ID:** vpc-0576b4cef931f401 (Lab1\_Nhom14\_VPC)
- Subnet ID:** subnet-05a4d02051a6b5c1e (Lab1\_Nhom14\_PrivateSubnet)
- Instance ARN:** arn:aws:ec2:us-east-1:851725517215:instance/i-08d02074116adec51
- Tags:**
  - aws:cloudformation:logical-id: PrivateEC2Instance
  - aws:cloudformation:stack-id: arn:aws:cloudformation:us-east-1:851725517215:stack/nhom14-stack-EC2Stack-QP0C9CWXHKNB/b42c5ca0-95ff-11ef-9f71-0affc957c93d
  - aws:cloudformation:stack-name: nhom14-stack-EC2Stack-QP0C9CWXHKNB
  - Name: Lab1\_Nhom14\_PrivateEC2Instance

Hình 43 – Thông tin chi tiết Private EC2 Instance

### 3/ Sử dụng Jenkins để quản lý quy trình CI/CD cho ứng dụng microservices