

AWS Project

Made by: Demond Wael, Mariam Tarek, Mariam Gamal

Overview

This documentation guides the process of creating a web application environment on AWS with the following features:

- 1- Infrastructure as Code (IaC) using AWS CloudFormation.
- 2- Hosting the web app on EC2 or via microservices (like ECS, EKS, or Lambda).
- 3- External storage for static content using Amazon S3.
- 4- Monitoring and notifications for metrics exceeding specific limits.
- 5- Automatic remediation in case of web service failure.
- 6- The environment must be secure, highly available, scalable, and have disaster recovery capabilities.
- 7- HTTP to HTTPS redirection via Load Balancer.
- 8- Mounting S3 on EC2.
- 9- Using API Gateway to fetch an image from S3.

Procedure

1. Create Network Environment with Infrastructure as a Code.

We have used AWS CloudFormation to define our infrastructure as code.

- we created a VPC with 2 public and 2 private subnets across multiple availability zones for high availability.
- Set up Internet Gateway (IGW) and NAT Gateways for internet connectivity.
- For IAM Roles and Policies, we have used LabRole for access control.

So we made a YAML template to be used on CloudFormation which is as follows:

Resources:

VPC:

Type: AWS::EC2::VPC

Properties:

CidrBlock: 10.0.0.0/16

EnableDnsSupport: true

EnableDnsHostnames: true

Tags:

- Key: Name

Value: VPC1

IGW:

Type: AWS::EC2::InternetGateway

Properties:

Tags:

- Key: Name

Value: VPC1 IG

VPCtoIGWConnection:

Type: AWS::EC2::VPCEGatewayAttachment

DependsOn:

- IGW

- VPC

Properties:

InternetGatewayId: !Ref IGW

VpcId: !Ref VPC

PublicRouteTable:

Type: AWS::EC2::RouteTable

DependsOn: VPC

Properties:

VpcId: !Ref VPC

Tags:

- Key: Name

Value: Public Route Table

PublicRoute:

Type: AWS::EC2::Route

DependsOn:

- PublicRouteTable*
- VPCtoIGWConnection*

Properties:

DestinationCidrBlock: 0.0.0.0/0

GatewayId: !Ref IGW

RouteTableId: !Ref PublicRouteTable

PrivateRouteTable:

Type: AWS::EC2::RouteTable

DependsOn: VPC

Properties:

VpcId: !Ref VPC

Tags:

- Key: Name*

Value: Private Route Table 1

PublicSubnet1:

Type: AWS::EC2::Subnet

DependsOn: VPC

Properties:

VpcId: !Ref VPC

MapPublicIpOnLaunch: true

CidrBlock: 10.0.0.0/24

AvailabilityZone: !Select

- 0*

- !GetAZs*

Ref: AWS::Region

Tags:

- Key: Name*

Value: Public Subnet 1

PublicSubnet2:

Type: AWS::EC2::Subnet

DependsOn: VPC

Properties:

VpcId: !Ref VPC

MapPublicIpOnLaunch: true

CidrBlock: 10.0.1.0/24

AvailabilityZone: !Select

- 1

- !GetAZs

Ref: AWS::Region

Tags:

- Key: Name

Value: Public Subnet 2

PublicRouteTableAssociation1:

Type: AWS::EC2::SubnetRouteTableAssociation

DependsOn:

- PublicRouteTable

- PublicSubnet1

Properties:

RouteTableId: !Ref PublicRouteTable

SubnetId: !Ref PublicSubnet1

PublicRouteTableAssociation2:

Type: AWS::EC2::SubnetRouteTableAssociation

DependsOn:

- PublicRouteTable

- PublicSubnet2

Properties:

RouteTableId: !Ref PublicRouteTable

SubnetId: !Ref PublicSubnet2

PrivateSubnet1:

Type: AWS::EC2::Subnet

DependsOn: VPC

Properties:

VpcId: !Ref VPC

CidrBlock: 10.0.2.0/23

AvailabilityZone: !Select

- 0

- !GetAZs

Ref: AWS::Region

Tags:

- Key: Name

Value: Private Subnet 1

PrivateSubnet2:

Type: AWS::EC2::Subnet

DependsOn: VPC

Properties:

VpcId: !Ref VPC

CidrBlock: 10.0.4.0/23

AvailabilityZone: !Select

- 1

- !GetAZs

Ref: AWS::Region

Tags:

- Key: Name

Value: Private Subnet 2

PrivateRouteTableAssociation1:

Type: AWS::EC2::SubnetRouteTableAssociation

DependsOn:

- PrivateRouteTable

- PrivateSubnet1

Properties:

RouteTableId: !Ref PrivateRouteTable

SubnetId: !Ref PrivateSubnet1

PrivateRouteTableAssociation2:

Type: AWS::EC2::SubnetRouteTableAssociation

DependsOn:

- PrivateRouteTable

- PrivateSubnet2

Properties:

RouteTableId: !Ref PrivateRouteTable

SubnetId: !Ref PrivateSubnet2

NATGateway:

DependsOn: PrivateSubnet1

Type: AWS::EC2::NatGateway

Properties:

SubnetId: !Ref PublicSubnet1

AllocationId: !GetAtt

- NATGatewayEIP

- AllocationId

NATGatewayEIP:

Type: AWS::EC2::EIP

Properties:

Domain: vpc

NATGatewayRoute:

Type: AWS::EC2::Route

Properties:

RouteTableId:

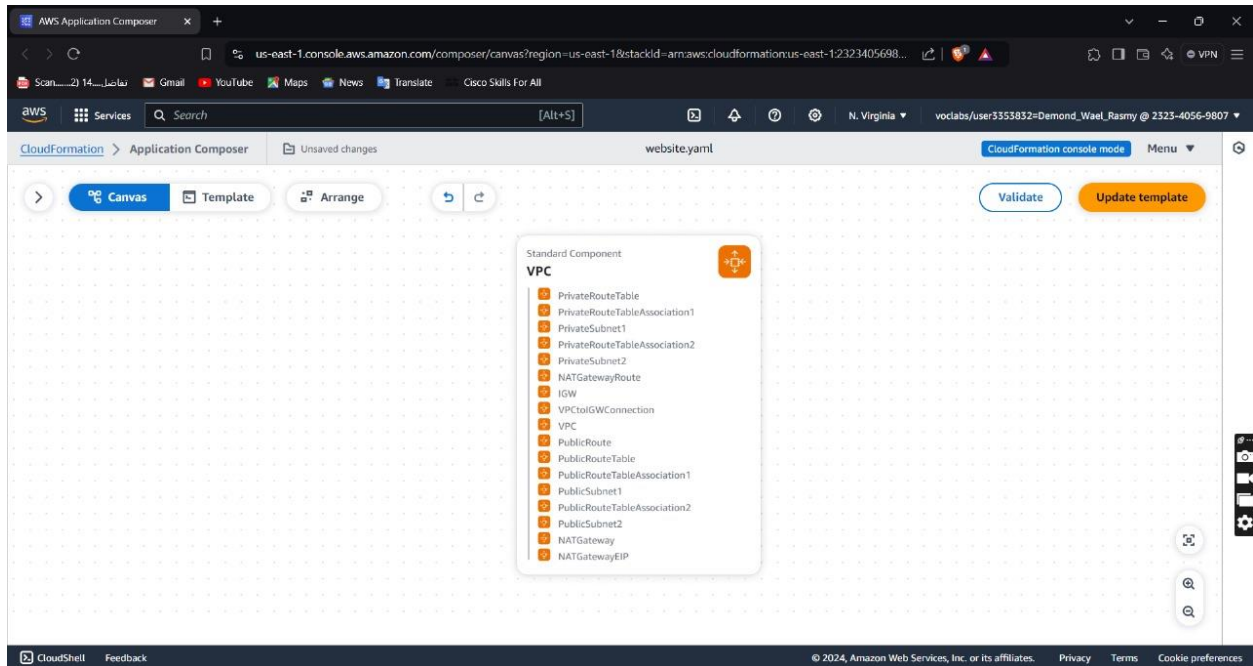
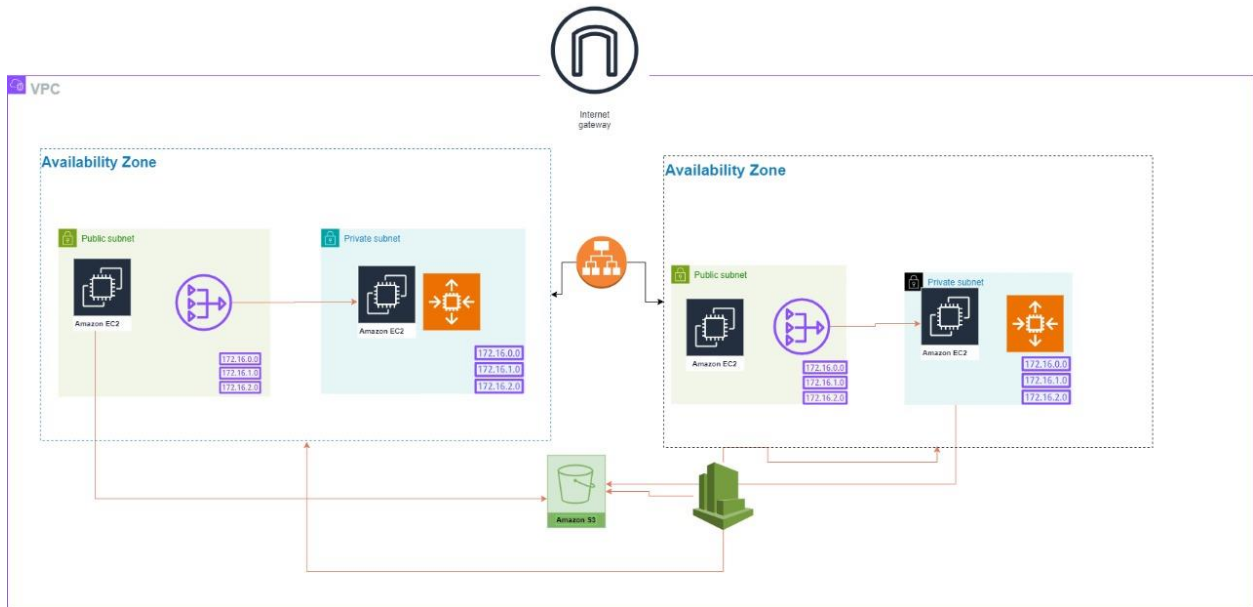
Ref: PrivateRouteTable

DestinationCidrBlock: 0.0.0.0/0

NatGatewayId:

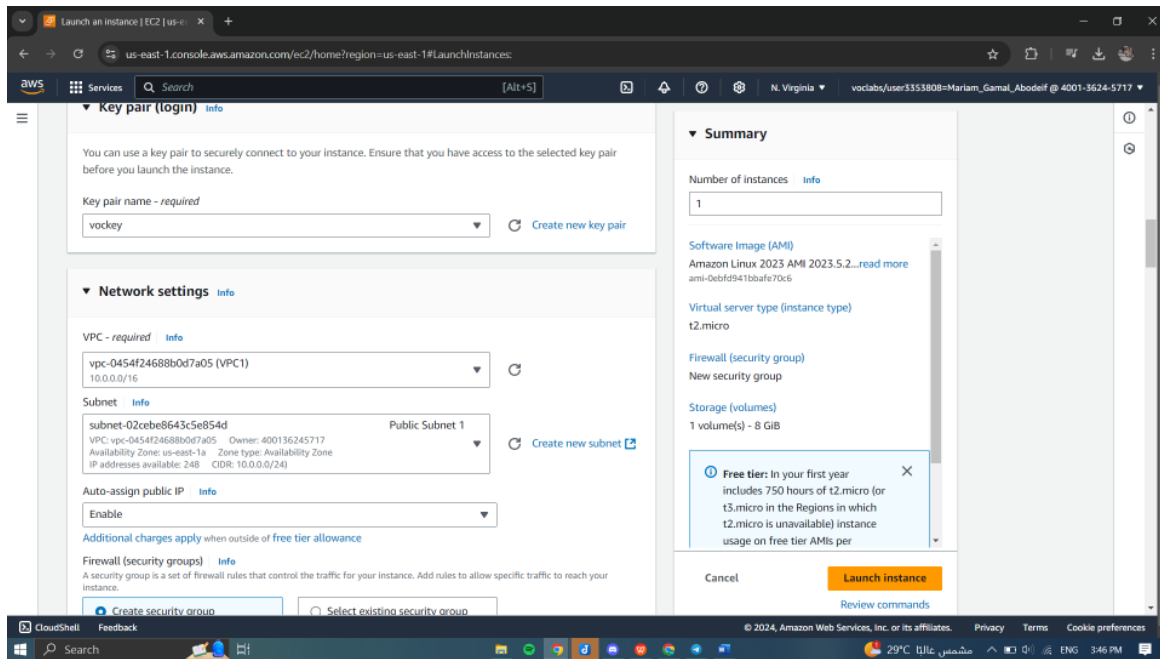
Ref: NATGateway

So our Environment Architecture will be like this:

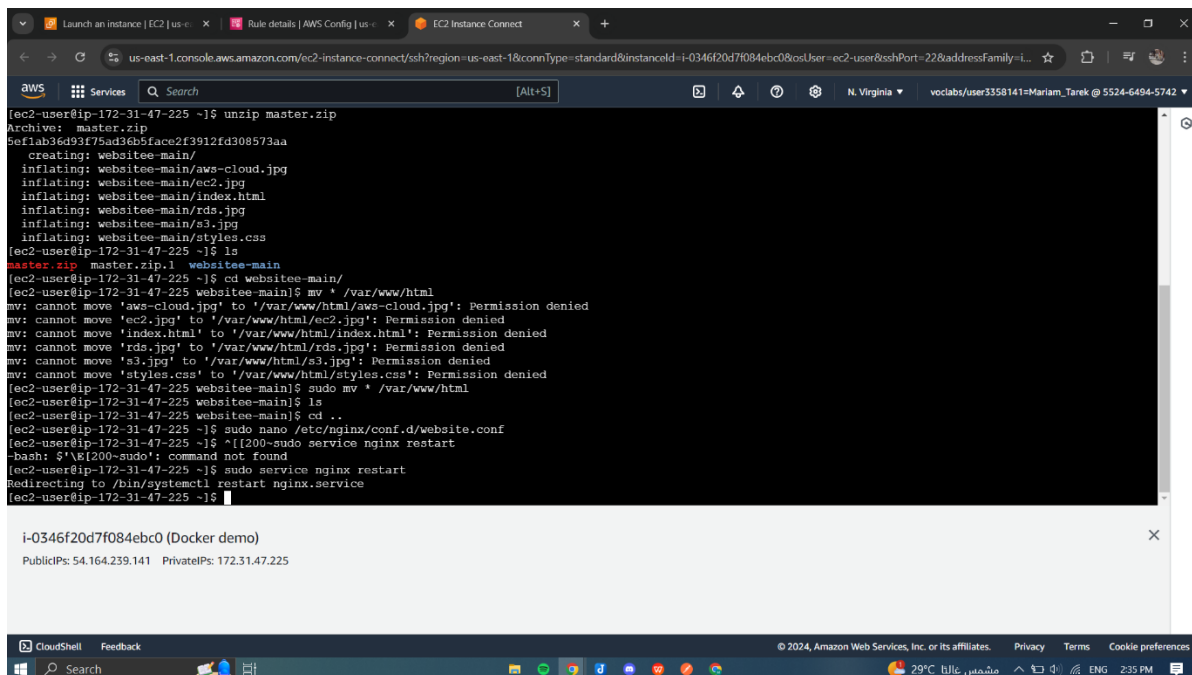


2. Web App Hosting on EC2

- Firstly, we have launched an EC2 instance to host the web server.



- Then, we connected to it with EC2 instance connect, and hosted the web app through nginx-service.



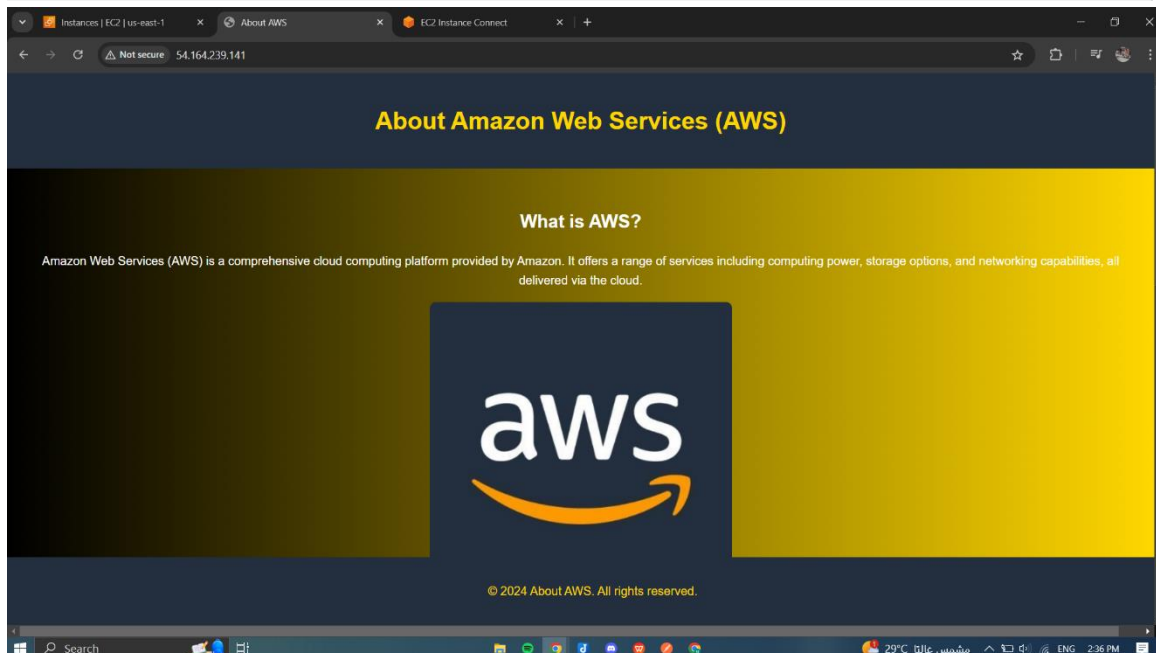

```
server {
    listen 80;
    server_name your-domain.com;

    root /var/www/html;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

```
mv: cannot move 'styles.css' to '/var/www/html/styles.css': Permission denied
[ec2-user@ip-172-31-47-225 website-main]$ sudo mv * /var/www/html
[ec2-user@ip-172-31-47-225 website-main]$ ls
[ec2-user@ip-172-31-47-225 website-main]$ cd ..
[ec2-user@ip-172-31-47-225 ~]$ sudo nano /etc/nginx/conf.d/website.conf
[ec2-user@ip-172-31-47-225 ~]$ sudo service nginx restart
-bash: $'\n[200-sudo': command not found
[ec2-user@ip-172-31-47-225 ~]$ sudo service nginx restart
Redirecting to /bin/systemctl restart nginx.service
[ec2-user@ip-172-31-47-225 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
   Active: active (running) since Tue 2024-10-01 11:35:29 UTC; 26s ago
     Process: 29341 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
     Process: 29349 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 29361 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
    Main PID: 29369 (nginx)
      Tasks: 2 (limit: 1112)
     Memory: 2.2M
        CPU: 59ms
    CGroup: /system.slice/nginx.service
            └─29369 "nginx: master process /usr/sbin/nginx"
              └─29372 "nginx: worker process"

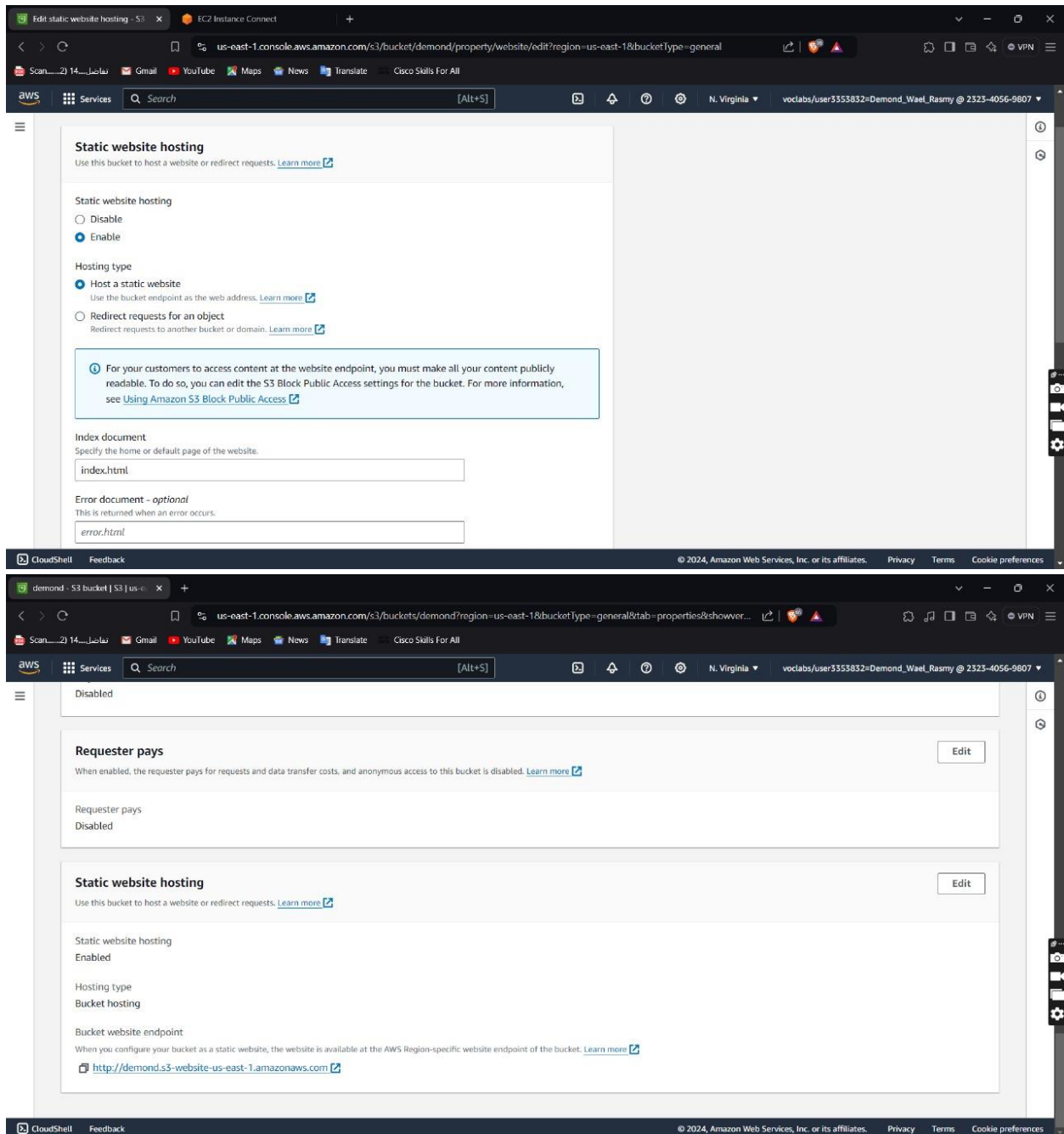
Oct 01 11:35:29 ip-172-31-47-225.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Oct 01 11:35:29 ip-172-31-47-225.ec2.internal nginx[29349]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Oct 01 11:35:29 ip-172-31-47-225.ec2.internal nginx[29349]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Oct 01 11:35:29 ip-172-31-47-225.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-31-47-225 ~]$
```



Finally, we managed to host the web app on EC2.

3. External Storage for Static Content through S3 bucket

- We created an S3 bucket to store the website content on it



The screenshot shows the Amazon S3 console interface. On the left is a navigation sidebar with options like Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Storage Lens, Dashboards, Storage Lens groups, AWS Organizations settings, Feature spotlight, and AWS Marketplace for S3. The main area displays the 'Objects (6)' view for a bucket named 'voclabs'. At the top of the main area are tabs for Objects, Properties, Permissions, Metrics, Management, and Access Points. Below the tabs is a toolbar with buttons for Refresh, Copy S3 URI, Copy URL, Download, Open, Delete, Actions, Create folder, and Upload. A descriptive paragraph explains that objects are fundamental entities stored in Amazon S3 and provides a link to learn more about permissions. Below this is a search bar labeled 'Find objects by prefix'. The core of the page is a table listing six objects:

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	aws-cloud.jpeg	jpeg	September 26, 2024, 02:21:29 (UTC+03:00)	11.5 KB	Standard
<input type="checkbox"/>	ec2.jpg	jpg	September 25, 2024, 03:18:47 (UTC+03:00)	33.0 KB	Standard
<input type="checkbox"/>	index.html	html	September 25, 2024, 03:18:47 (UTC+03:00)	2.8 KB	Standard
<input type="checkbox"/>	rds.jpg	jpg	September 25, 2024, 03:18:48 (UTC+03:00)	34.6 KB	Standard
<input type="checkbox"/>	s3.jpg	jpg	September 25, 2024, 03:18:49 (UTC+03:00)	30.9 KB	Standard
<input type="checkbox"/>	styles.css	css	September 25, 2024, 03:18:50 (UTC+03:00)	1.3 KB	Standard

At the bottom of the console, there is a footer bar containing CloudShell, Feedback, copyright information (© 2024, Amazon Web Services, Inc. or its affiliates), Privacy, Terms, and Cookie preferences links.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::EC2-INSTANCE-ROLE-ARN"
      },
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::your-bucket-name"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::EC2-INSTANCE-ROLE-ARN"
      },
```

```

    "Action": [

        "s3:GetObject",

        "s3:PutObject"

    ],

    "Resource": "arn:aws:s3:::your-bucket-name/*"

}

]

}

```

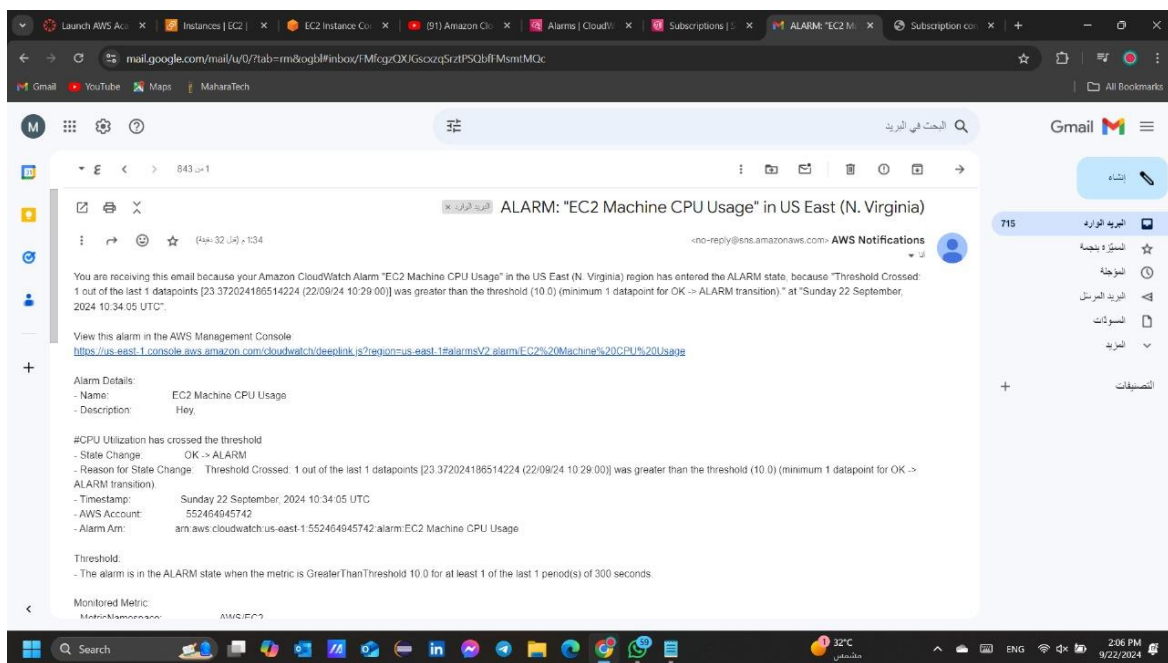
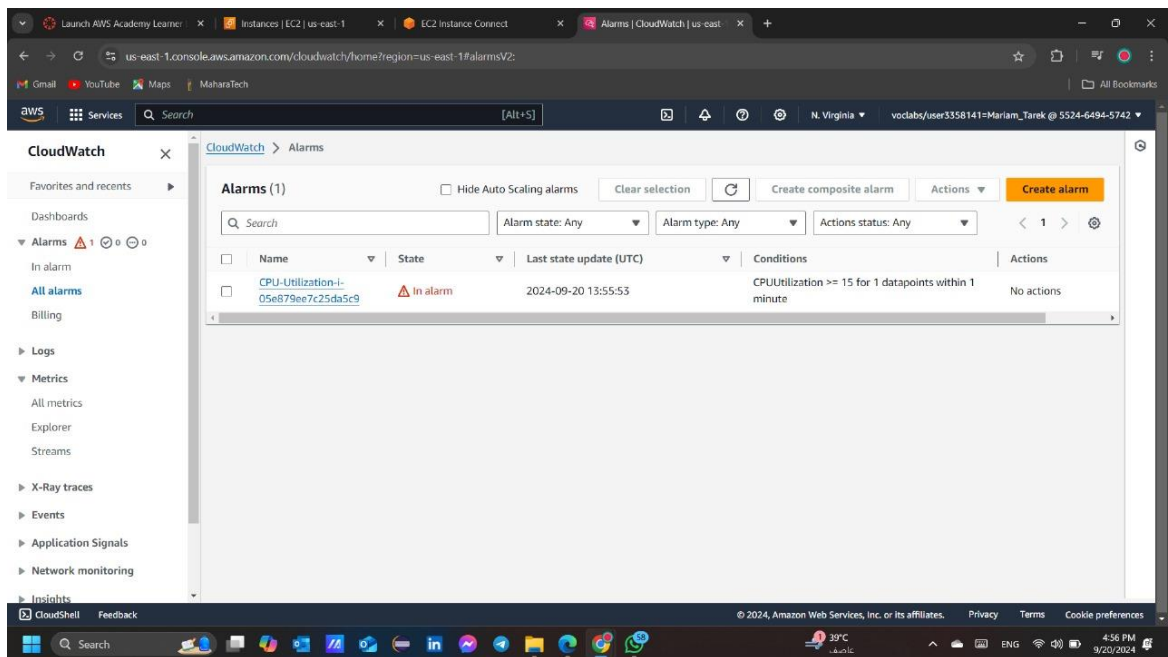
4. Monitoring your servers with integrated notifications while metrics exceed specific limit.

- We have set up Amazon CloudWatch to monitor CPU usage for our EC2 instances.
- Then, we set CloudWatch Alarms to notify via SNS (Simple Notification Service) if a metric exceeds a defined threshold (CPUUtilization > 10 for 1 datapoints within 5 minutes).

The screenshot displays the Amazon CloudWatch console interface. On the left, a navigation sidebar includes links for Dashboards, Alarms, Logs, Metrics, X-Ray traces, Events, Application Signals, Network monitoring, and Insights. The main content area is titled 'CloudWatch Alarms > EC2 Machine CPU Usage'. It shows a list of alarms with one active alarm, 'EC2 Machine CPU Usage', which is in an 'OK' state. Below this, a 'Details' panel provides comprehensive information about the alarm:

Details			
Name EC2 Machine CPU Usage	State OK	Namespace AWS/EC2	Datapoints to alarm 1 out of 1
Type Metric alarm	Threshold CPUUtilization > 10 for 1 datapoints within 5 minutes	Metric name CPUUtilization	Missing data treatment Treat missing data as missing
Description Hey, #CPU Utilization has crossed the threshold	Last state update 2024-10-01 10:28:05 (UTC)	Instanceid i-05e879ee7c25da5c9	Percentiles with low samples evaluate
Actions Actions enabled		Instance name CloudWatch-Test	ARN arn:aws:cloudwatch:us-east-1:552464945742:alarm:EC2 Machine CPU Usage
		Statistic Average	
		Period 5 minutes	

At the bottom of the details panel, there is a link to 'View EventBridge rule'.



5. Automatic Remediation

- We used System Manager and AWS Config for automatic remediation.
- The event here is to check if S3 bucket have default encryption enabled or bucket policies, and if it doesn't have, then the remediation action is to enable the encryption for S3 bucket.

This screenshot shows the AWS Config console interface. The left sidebar contains navigation links for Dashboard, Conformance packs, Rules, Resources, Aggregators, Compliance Dashboard, Inventory Dashboard, Authorizations, Advanced queries, Settings, and What's new. The main content area displays the details for a remediation action. At the top, it shows the rule name 's3-bucket-server-side-encryption-enabled2' and its description: 'using AES-256 or AWS KMS. The rule is NON_COMPLIANT if an S3 bucket is not encrypted by default.' Below this, the 'Remediation action' is listed as 'AWS-EnableS3BucketEncryption'. The 'Parameters' section is a table with the following data:

Key	Value	Description
AutomationAssumeRole	arn:aws:iam::552464945742:role/LabRole	(Optional) The ARN of the role that allows Automation to perform the actions on your behalf.
BucketName	RESOURCE_ID	(Required) The name of the S3 Bucket whose content will be encrypted.
SSEAlgorithm	AES256	(Optional) Server-side encryption algorithm to use for the default encryption.

This screenshot shows the AWS Config console interface, displaying the details of the rule 's3-bucket-server-side-encryption-enabled2'. The left sidebar is the same as in the previous screenshot. The main content area shows the rule details, including the description: 'Checks if S3 buckets have default encryption enabled or have bucket policies that explicitly deny put-object requests without server side encryption using AES-256 or AWS KMS. The rule is NON_COMPLIANT if an S3 bucket is not encrypted by default.' Below this, the 'Rule details' section shows the rule name, description, and the last successful evaluation time: 'September 26, 2024 2:32 PM'. The 'Remediation action' section is also visible, showing the action 'AWS-EnableS3BucketEncryption' and its description: 'Enables Encryption on S3 Bucket'.

6. Check the environment

Secure: We attached the Lab IAM Role for our environment that Securely grant permissions to services and users.

Highly Available – Scalable: We have made our environment highly available and scalable through making an application load balancer and auto scaling group that's attached with a launch template.

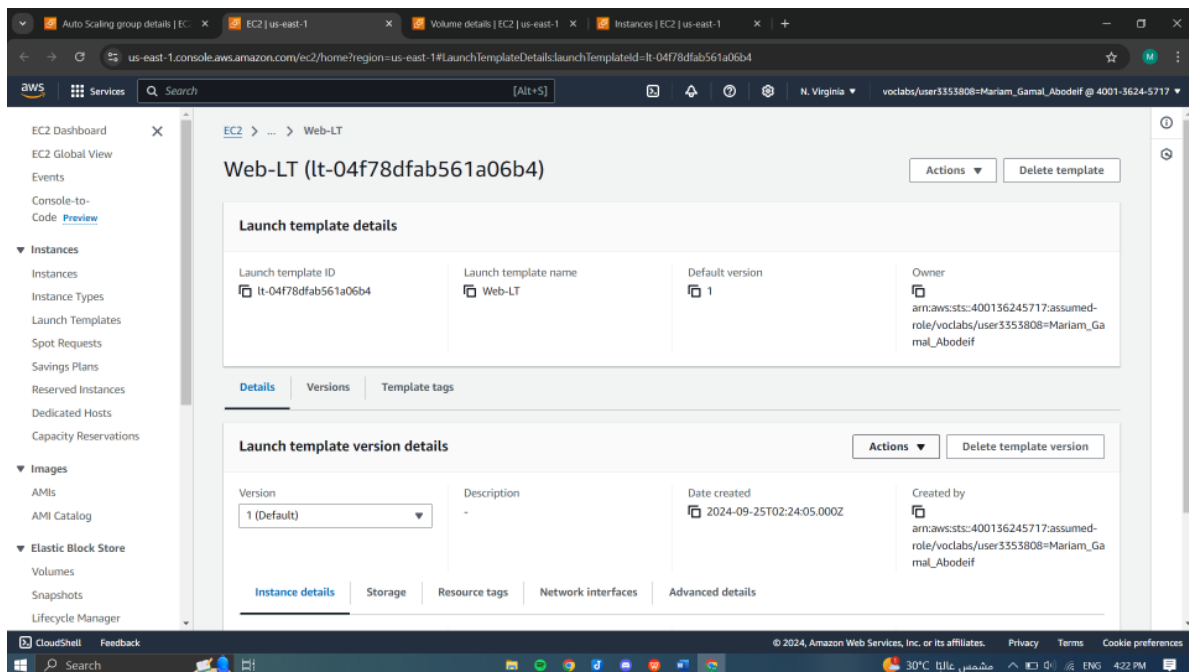
The image displays two screenshots of the AWS Management Console. The top screenshot shows the 'Web-ASG' (Auto Scaling Group) details page. The bottom screenshot shows the 'VPC1-LB' (Load Balancer) details page.

Web-ASG Details:

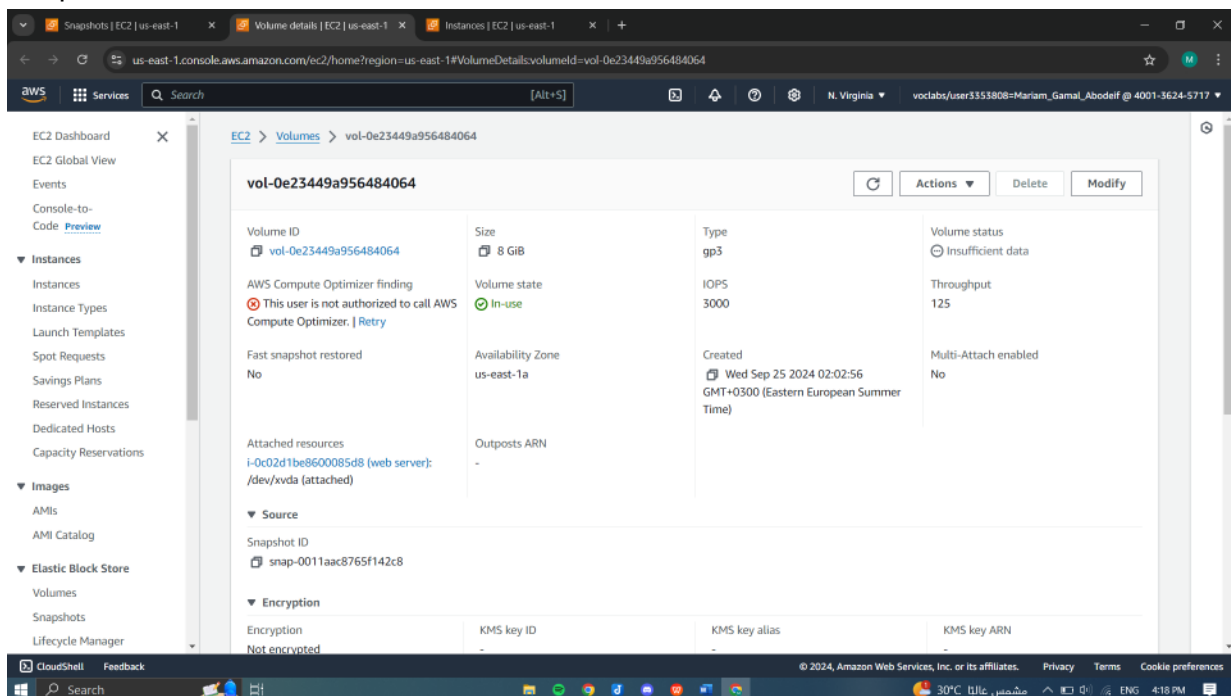
Group details			
Auto Scaling group name	Web-ASG	Desired capacity	2
Date created	Wed Sep 25 2024 05:26:19 GMT+0300 (Eastern European Summer Time)	Minimum capacity	2
		Maximum capacity	2
Launch template		AMI ID	Instance type
Web-LT		ami-0c4b23591f29f79c	t2.micro
		Owner	arn:aws:sts::400136245717:assumed-role/voclabs/user3353808-Mariam_Gamal_Abodeif

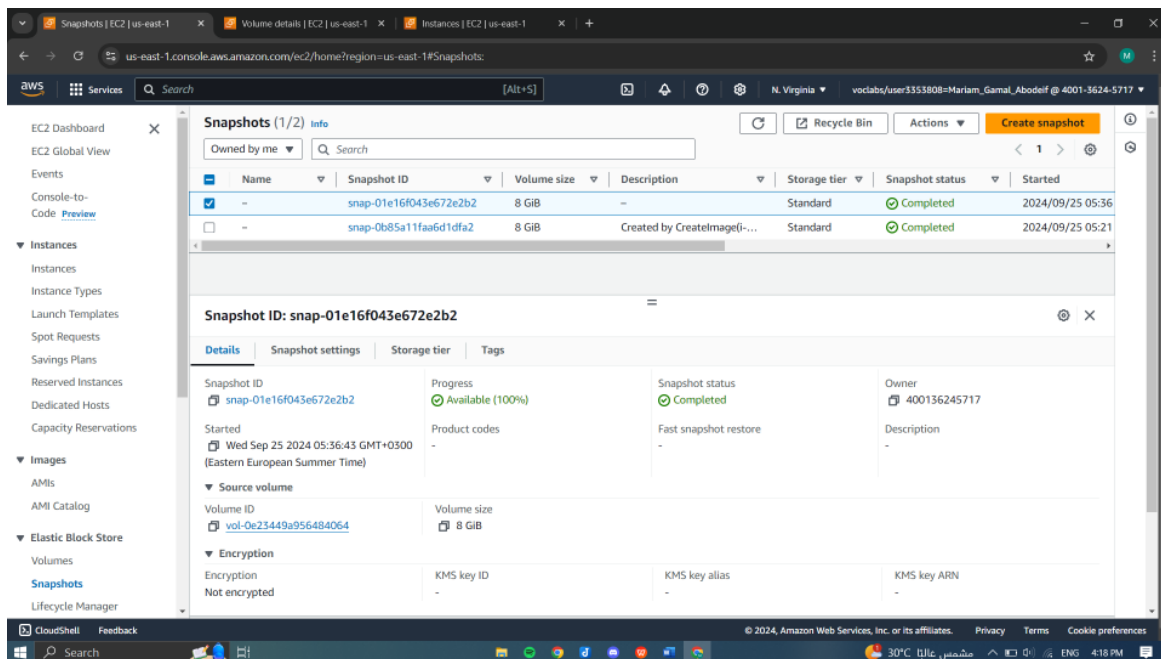
VPC1-LB Details:

Details			
Load balancer type	Application	Status	Active
Scheme	Internet-facing	Hosted zone	Z35SXDOTRQ7X7K
Load balancer ARN		DNS name	
arn:aws:elasticloadbalancing:us-east-1:400136245717:loadbalancer/app/VPC1-LB/54ff685578c9d6e9		VPC1-LB-2107200022.us-east-1.elb.amazonaws.com (A Record)	



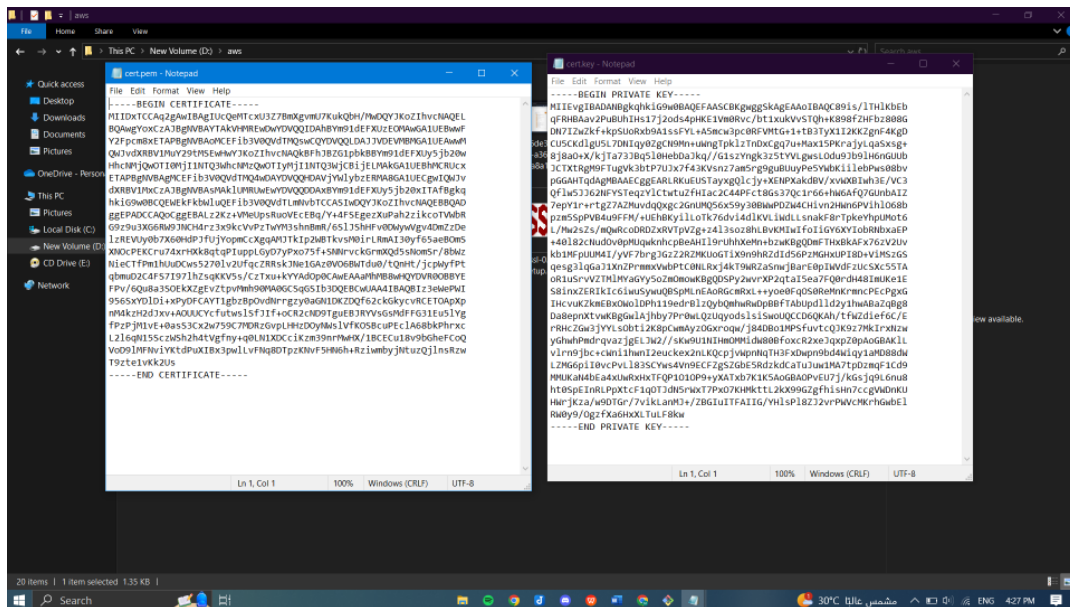
Disaster Recovery: We have created a volume attached to our EC2 instance and made a snapshot for it.



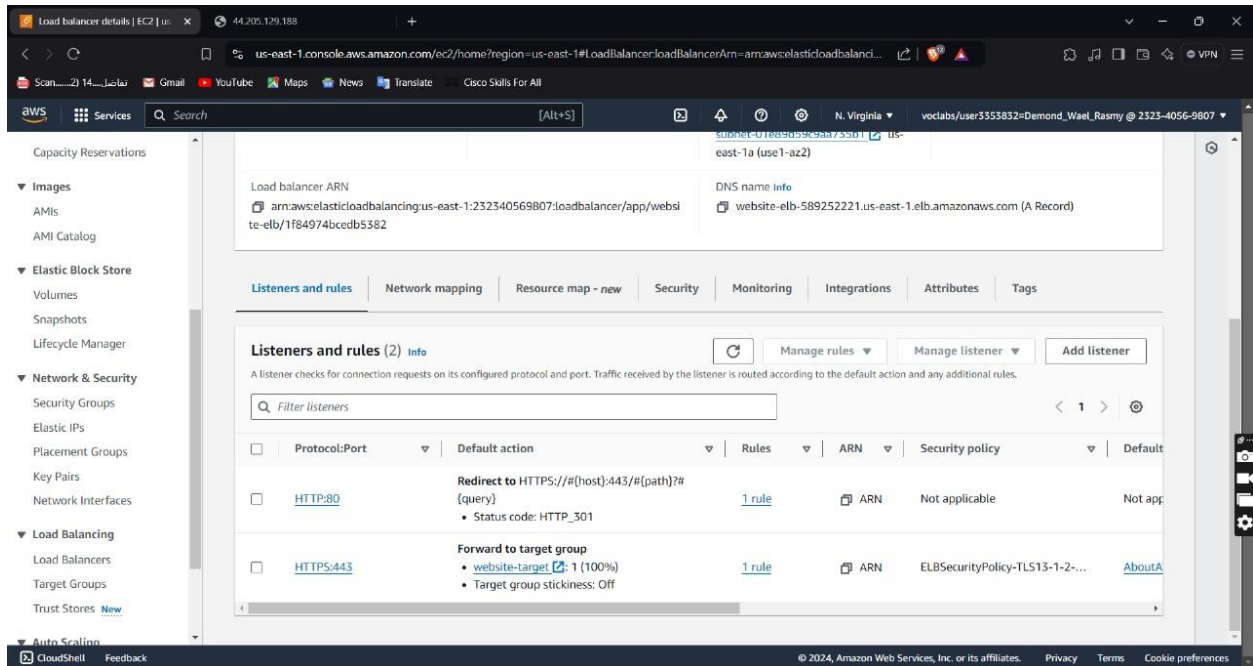


7. Enable HTTP to HTTPS Redirection on Load Balancer

1. First, we have created an SSL/TLS certificate through OpenSSL.



Then, we configured the Load balancer listeners to redirect all HTTP traffic (port 80) to HTTPS (port 443).



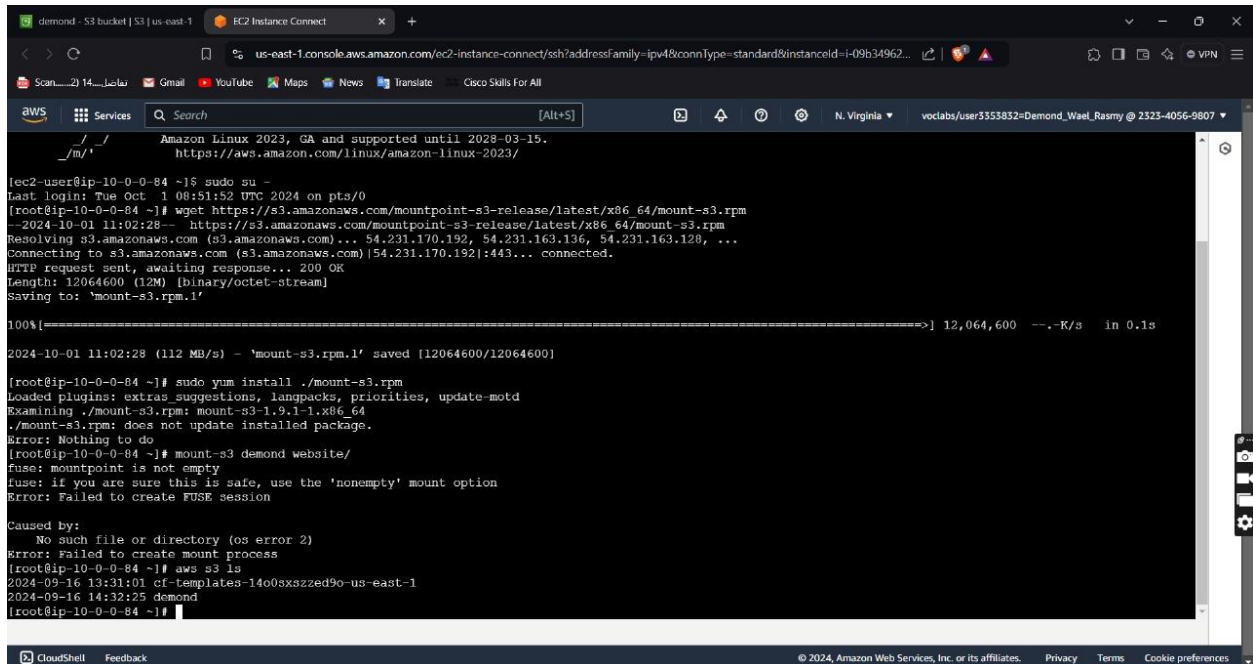
8. Mount S3 on EC2

- we connect our EC2 instance.
- Then enter the following commands:

```
$ wget https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm
```

```
$ sudo yum install ./mount-s3.rpm
```

```
$ sudo mount-s3 your-s3-bucket s3mount/
```



```
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/

[ec2-user@ip-10-0-0-84 ~]$ sudo su -
Last login: Tue Oct 1 08:51:52 UTC 2024 on pts/0
[root@ip-10-0-0-84 ~]# wget https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm
--2024-10-01 11:02:28-- https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm
Resolving s3.amazonaws.com (s3.amazonaws.com)... 54.231.170.192, 54.231.163.136, 54.231.163.128, ...
Connecting to s3.amazonaws.com (s3.amazonaws.com)|54.231.170.192|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12064600 (12M) [binary/octet-stream]
Saving to: 'mount-s3.rpm.1'

100%[=====>] 12,064,600 --.-K/s in 0.1s

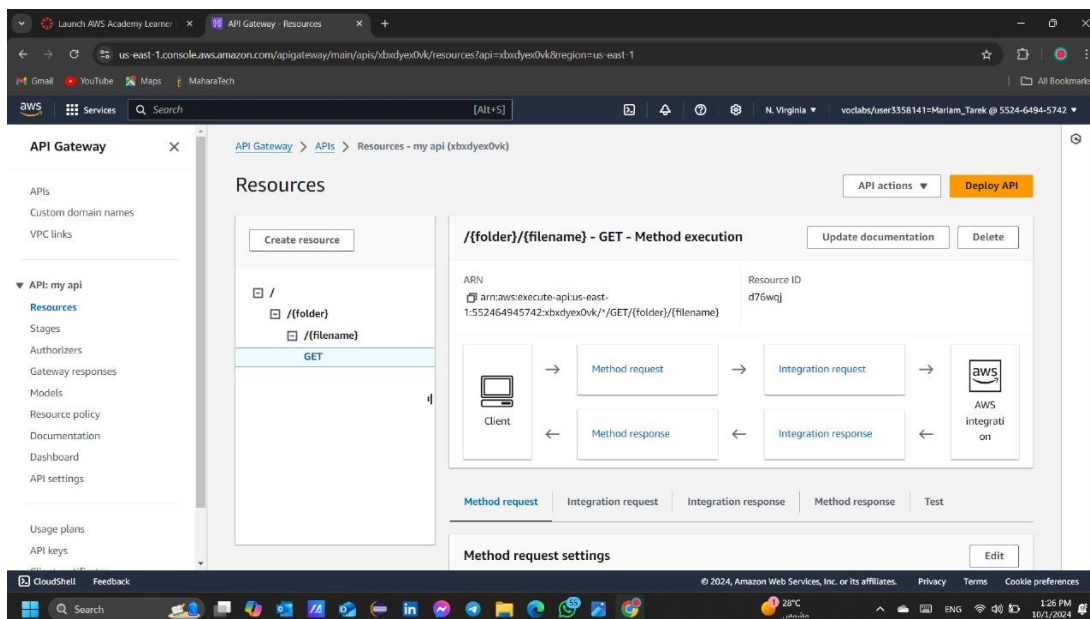
2024-10-01 11:02:28 (112 MB/s) - 'mount-s3.rpm.1' saved [12064600/12064600]

[root@ip-10-0-0-84 ~]# sudo yum install ./mount-s3.rpm
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Examining ./mount-s3.rpm: mount-s3-1.9.1-1.x86_64
./mount-s3.rpm: does not update installed package.
Error: Nothing to do
[root@ip-10-0-0-84 ~]# mount-s3 demond website/
fuse: mountpoint is not empty
fuse: if you are sure this is safe, use the 'nonempty' mount option
Error: Failed to create FUSE session

Caused by:
  No such file or directory (os error 2)
Error: Failed to create mount process
[root@ip-10-0-0-84 ~]# aws s3 ls
2024-09-16 13:31:01 cf-templates-14o0sxszzed9o-us-east-1
2024-09-16 14:32:25 demond
[root@ip-10-0-0-84 ~]#
```

9. Using API Gateway to Fetch Images from S3

1. First, we create an API Gateway with an endpoint that fetches images stored in S3 bucket.



Services

Search


[Alt+S]

Method details

Integration type


☐ Lambda function

Integrate your API with a Lambda function.




☐ HTTP

Integrate with an existing HTTP endpoint.




☐ Mock

Generate a response based on API Gateway mappings and transformations.




☒ AWS service

Integrate with an AWS Service.



☐ VPC link

Integrate with a resource that isn't accessible over the public internet.



AWS Region

us-east-1

AWS service

Simple Storage Service (S3)

AWS subdomain

CloudShell

Feedback

Services

Search

[Alt+S]

HTTP method

GET

Action type

☐ Use action name

☒ Use path override

Path override

{folder}/{filename}

Execution role

arn:aws:iam::552464945742:role/LabRole

Credential cache

Do not add caller credentials to cache key

Content handling

Learn more

Passthrough

Integration timeout

Info

By default, you can enter an integration timeout of 50 - 29,000 milliseconds. You can use Service Quotas to raise the integration timeout to greater than 29,000 ms.

29000

CloudShell

Feedback

Services

Search

[Alt+S]

transform any matches for the incoming content type. To secure your integration, select **When there are no templates defined (recommended)**.

URL path parameters

Name

Mapped from

Caching

filename

method.request.path.file

☐

Remove

folder

method.request.path.fol

☐

Remove

Add path parameter

URL query string parameters

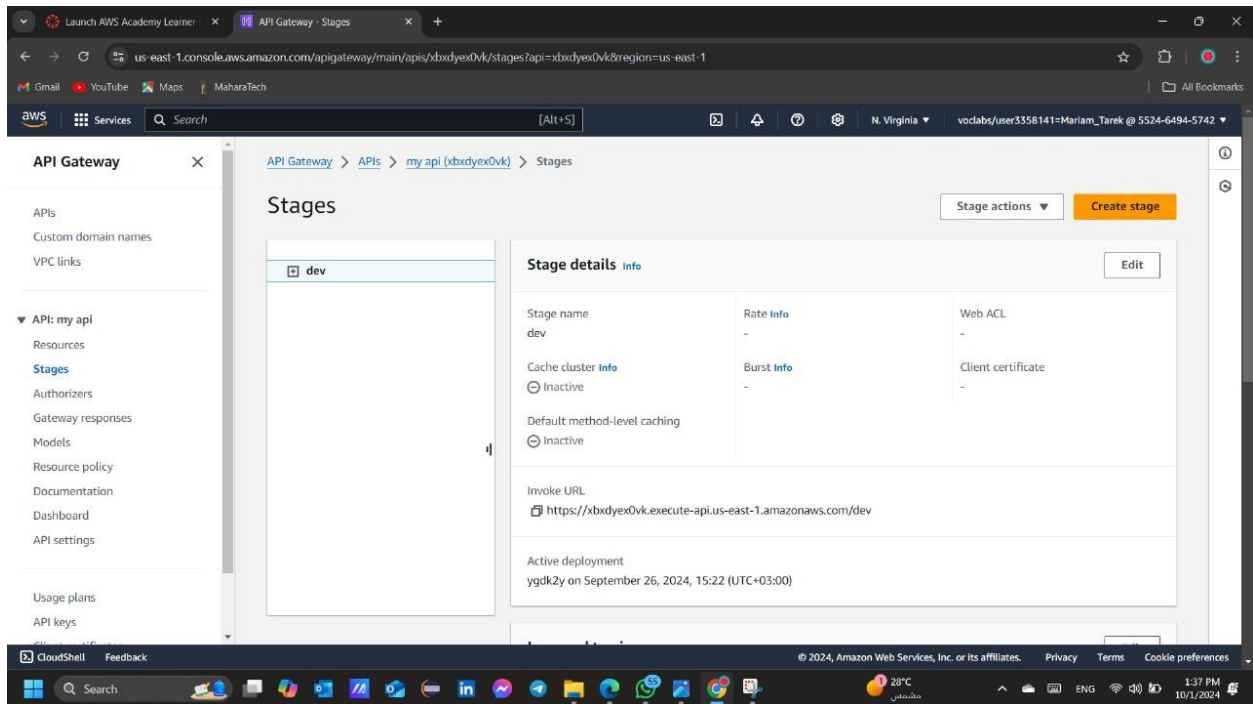
URL request headers parameters

Mapping templates

CloudShell

Feedback

- Then, we create a stage to deploy the API



- Finally, we test it through Postman to view any object from S3 bucket

