

# Automating Infrastructure Deployment

## Problem Statement

Modern cloud environments require automated, repeatable, and version-controlled infrastructure deployment. Managing networking and application resources manually across multiple AWS services and Regions is time-consuming and prone to configuration errors. There is a need for a centralized, automated approach to deploy, update, and replicate infrastructure using Infrastructure-as-Code (IaC) mechanisms.

## Objectives

After completing this lab, learners will be able to:

1. Deploy a VPC networking layer using AWS CloudFormation templates.
2. Deploy an application layer through CloudFormation to create and manage application resources.
3. Use Git and AWS CodeCommit to version-control CloudFormation templates and trigger stack creation and updates.
4. Automate stack deployments and updates using AWS CodePipeline.
5. Replicate networking and application infrastructure across multiple AWS Regions using CloudFormation for consistent multi-Region deployments.

## Business Goal

Automate the deployment and management of the cafe's static and dynamic website using AWS CloudFormation, while adopting version control (CodeCommit) and continuous delivery (CodePipeline). Later, duplicate the same infrastructure in another AWS Region.

## Phase 1 - Static Website Hosting with CloudFormation

## **Task 1 : Connect to VS Code IDE**

1. Copy LabIDEURL and LabIDEPASSWORD from AWS Details.
2. Open the VS Code IDE in a new browser tab and log in.

## **Task 2 : Create CloudFormation Template for S3 Bucket**

1. Create file S3.yaml.
2. Add the following yaml code snippet

```
AWSTemplateFormatVersion: "2010-09-09"
```

```
Description: cafe S3 template
```

```
Resources:
```

```
S3Bucket:
```

```
Type: AWS::S3::Bucket
```

3. Deploy stack using AWS CLI:

```
aws cloudformation create-stack --stack-name CreateBucket --template-body file://S3.yaml
```

4. Verify stack and bucket creation in CloudFormation + S3 console

The screenshot shows the AWS CloudFormation template editor in Visual Studio Code. The left sidebar has an 'EXPLORER' section with a 'ENVIRONMENT' folder containing 'S3.yaml'. The main area shows the content of 'S3.yaml':

```
! S3.yaml
1 AWSTemplateFormatVersion: "2010-09-09"
2 Description: "cafe S3 template"
3 Resources:
4   S3Bucket:
5     Type: AWS::S3::Bucket
```

Below the code editor is a terminal window showing the command-line interface:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● [ec2-user@ip-10-0-1-89 environment]$ aws configure get region
us-east-1
● [ec2-user@ip-10-0-1-89 environment]$ aws cloudformation create-stack --stack-name CreateBucket --template-body file://S3.yaml
{
  "StackId": "arn:aws:cloudformation:us-east-1:248631246916:stack/CreateBucket/320e35a0-c7b7-11f0-ac8b-0affc9765659"
}
● [ec2-user@ip-10-0-1-89 environment]$
```

The status bar at the bottom indicates: Ln 5, Col 26 Spaces: 2 UTF-8 LF YAML Layout: U.S.

## Task 3 : Configure Bucket as Static Website

1. Download website assets → set S3 ownership & access → upload files.
2. Modify template to:
  - a. Add DeletionPolicy: Retain
  - b. Enable static website hosting (index.html)
  - c. Add Output for WebsiteURL
3. Validate and update the stack:

```
aws cloudformation validate-template --template-body file://S3.yaml
```

```
aws cloudformation update-stack --stack-name CreateBucket --template-body file://S3.yaml
```

4. Test website URL from Outputs tab

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with navigation links like 'Amazon S3', 'General purpose buckets', 'Storage Lens', and 'Feature spotlight'. The main area displays a bucket named 'createbucket-s3bucket-t2dpn8nqvjuf'. Inside the bucket, there are three objects listed:

Name	Type	Last modified	Size	Storage class
S3.yaml	yaml	November 22, 2025, 21:03:18 (UTC+05:30)	119.0 B	Standard
static-website.zip	zip	November 22, 2025, 21:03:18 (UTC+05:30)	21.7 MB	Standard
static/	Folder	-	-	-

The screenshot shows the AWS CloudFormation console. On the left, the 'EXPLORER' panel shows files like 'S3.yaml' and 'static-website.zip'. The main area displays the 'S3.yaml' template content:

```

! S3.yaml x
! S3.yaml
1 AWSTemplateFormatVersion: "2010-09-09"
2 Description: "cafe S3 template"
3
4 Resources:
5   S3Bucket:
6     Type: AWS::S3::Bucket
7     Properties:
8       WebsiteConfiguration:
9         IndexDocument: index.html
10        DeletionPolicy: Retain
11
12 Outputs:
13   WebsiteURL:
14     Description: "URL for static website hosted in S3"
15     Value: !Sub "http://${S3Bucket}.s3-website-${AWS::Region}.amazonaws.com"
16

```

Below the template, the 'TERMINAL' tab shows the command-line output of the stack update and validation processes:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
① [ec2-user@ip-10-0-1-89 environment]$ aws cloudformation update-stack --stack-name CreateBucket --template-body file://S3.yaml
An error occurred (ValidationError) when calling the UpdateStack operation: No updates are to be performed.
② [ec2-user@ip-10-0-1-89 environment]$ aws cloudformation validate-template --template-body file://S3.yaml
{
  "Parameters": [],
  "Description": "cafe S3 template"
}
③ [ec2-user@ip-10-0-1-89 environment]$ aws cloudformation update-stack --stack-name CreateBucket --template-body file://S3.yaml
{
  "StackId": "arn:aws:cloudformation:us-east-1:248631246916:stack/CreateBucket/320e35a0-c7b7-11f0-ac8b-0affc9765659"
}
④ [ec2-user@ip-10-0-1-89 environment]$ 

```

## Phase 2 - Version Control with CodeCommit

## **Task 4 : Clone CodeCommit Repository**

1. Identify repository: CFTemplatesRepo

2. Clone using HTTPS (GRC) URL:

```
git clone <CodeCommit-URL>
```

3. Check repo state: git status

## **Phase 3 - CI/CD Deployment of Network & Application**

### **Task 5 : Create Network Layer via CloudFormation + CodePipeline**

1. Duplicate template → rename cafe-network.yaml

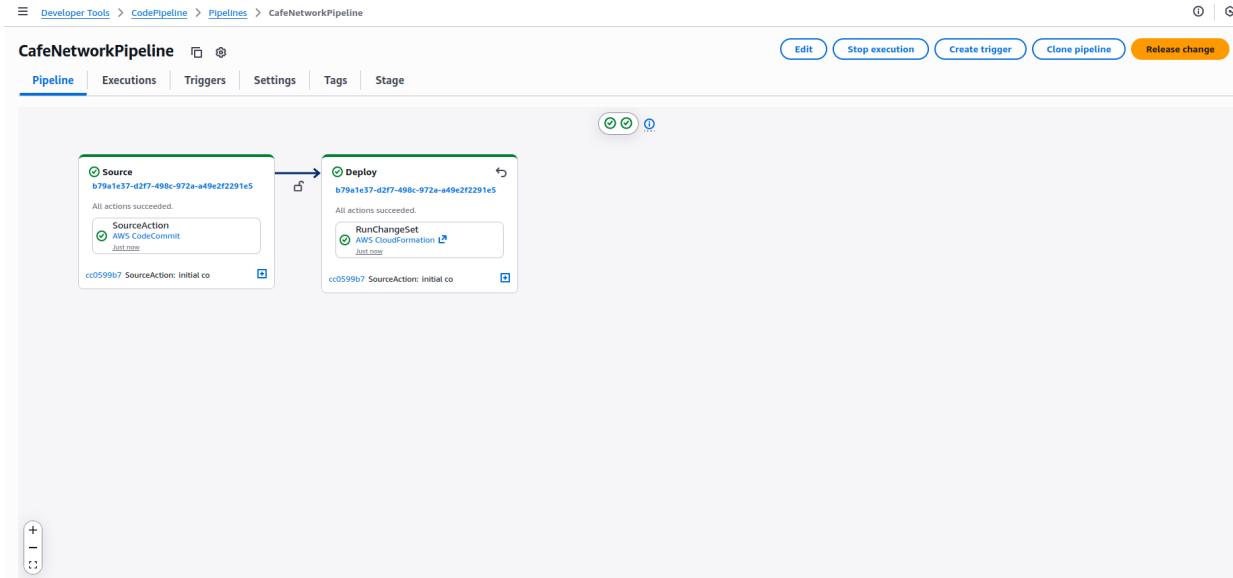
2. Edit description: *Network layer for the cafe*

3. Commit and push:

```
git add templates/cafe-network.yaml  
git commit -m "initial commit of network template"  
git push
```

4. CafeNetworkPipeline triggers automatically → creates update-cafe-network stack

5. Verify VPC & Subnet in VPC console



## Task 6 : Add Outputs to Network Template

1. Add VpcId and PublicSubnet Outputs including Export Names.
2. Commit and push → Pipeline updates stack.
3. Verify Outputs in CloudFormation.

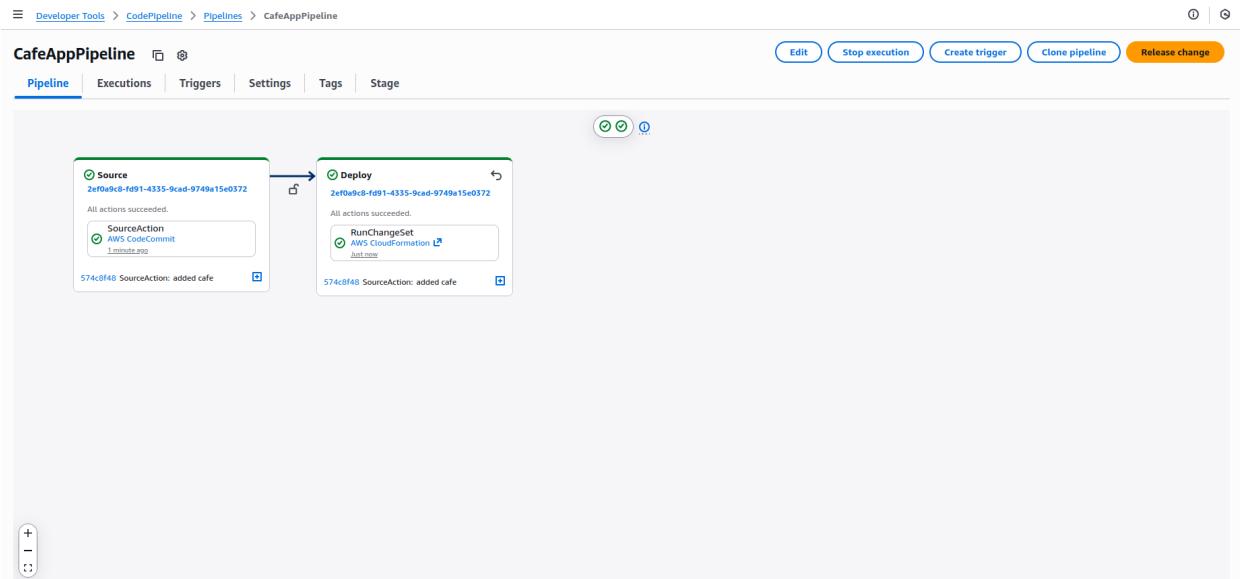
## Task 7 : Create Application EC2 Stack via CI/CD

1. Duplicate template2.yaml → rename cafe-app.yaml
2. Add new parameter for InstanceType
3. Add EC2 instance resource using:
  - a. LatestAmiId
  - b. InstanceType parameter
  - c. IAM Role: CafeRole
  - d. Mapping-based KeyName
  - e. Tag: Name: Cafe Web Server
  - f. Complete UserData installation script
4. Validate → git add → commit → push

5. CafeAppPipeline runs → creates update-cafe-app stack

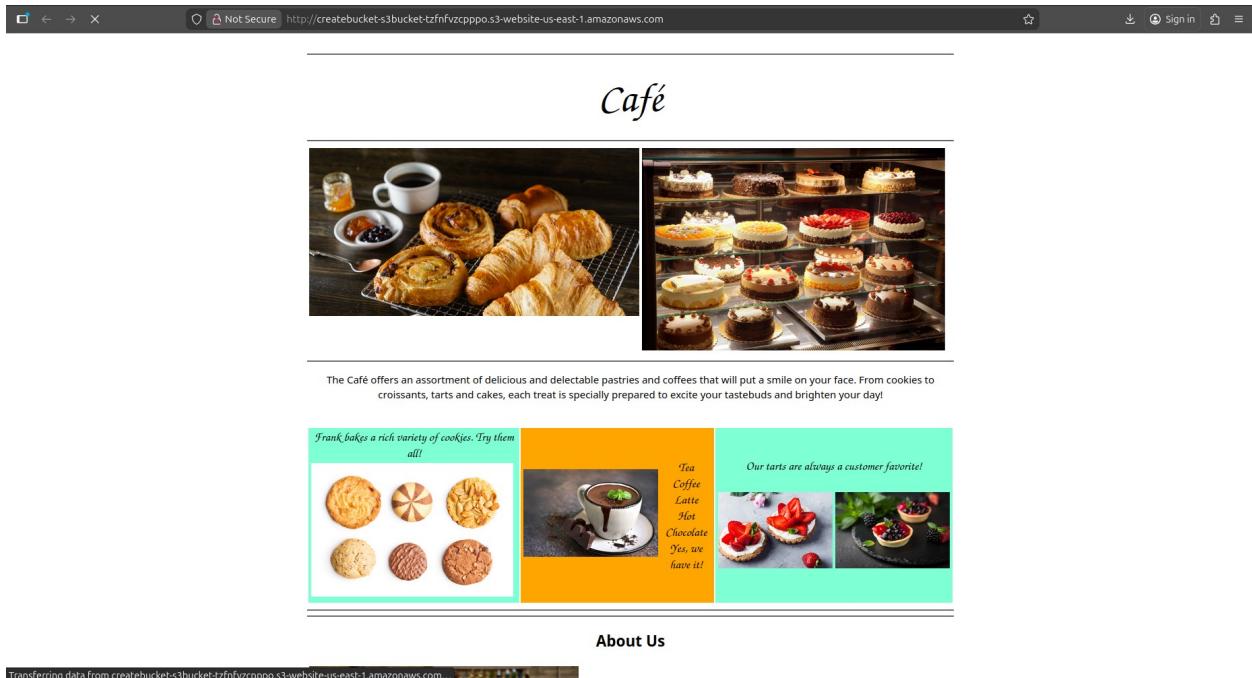
6. Verify EC2 instance and test site:

<http://<public-ip-address>/cafe>



The screenshot shows the AWS CodePipeline console with the sidebar expanded to show navigation options like 'Source', 'Artifacts', 'Build', 'Deploy', 'Pipeline', 'Getting started', 'Pipelines', 'Account metrics', and 'Settings'. The main area displays a table of existing pipelines. Two pipelines are listed: 'CafeAppPipeline' and 'CafeNetworkPipeline'. Both pipelines are shown as 'Succeeded' with their latest execution details. A 'Create pipeline' button is visible at the top right of the table area.

Name	Latest execution status	Latest source revisions	Latest execution started	Most recent executions
CafeAppPipeline	Succeeded	SourceAction - 574cbf48: added cafe app	1 minute ago	<span>Green</span> <span>Red</span> <span>Yellow</span> <span>Green</span> <span>Green</span> <span>Green</span> View details
CafeNetworkPipeline	Succeeded	SourceAction - 574cbf48: added cafe app	1 minute ago	<span>Green</span> <span>Green</span> <span>Green</span> <span>Red</span> <span>Green</span> <span>Green</span> View details



## Phase 4 - Replicate Infrastructure in a Second Region

### Task 8 : Duplicate Network + Application in us-west-2

1. Create network stack in new region:

```
aws cloudformation create-stack --stack-name update-cafe-network --template-body
file://.../cafe-network.yaml --region us-west-2
```

2. Create key pair: cafe-oregon.
3. Upload cafe-app.yaml to S3 and copy Object URL.
4. In CloudFormation (us-west-2) → create stack using S3 URL.
5. Choose InstanceType: t3.micro (via parameter)
6. Verify EC2 instance, wait for bootstrap → Access website

<http://<public-ip-address>/cafe>

**EC2 > Security Groups**

**Instances**

- Instances
- Instance Types
- Launch Templates
- Spot Requests
- Savings Plans
- Reserved Instances
- Dedicated Hosts
- Capacity Reservations
- Capacity Manager [New](#)

**Images**

- AMIs
- AMI Catalog

**Elastic Block Store**

- Volumes
- Snapshots
- Lifecycle Manager

**Network & Security**

- Security Groups**
- Elastic IPs
- Placement Groups
- Key Pairs
- Network Interfaces

**Load Balancing**

- Load Balancers
- Target Groups
- Trust Stores

**Events**

**Instances**

**Images**

**Elastic Block Store**

**Network & Security**

**Load Balancing**

**Events**

**Instances**

**Images**

**Elastic Block Store**

**Network & Security**

**Load Balancing**

**Security Groups (2) Info**

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-0969eaae75f1ea2	default	vpc-04d2690634a4a0ad1	default VPC security group	200259598426
-	sg-0bb7c9844fe990d3	default	vpc-0b28d59cb5be94bac	default VPC security group	200259598426

**Select a security group**

**Snapshots (1) Info**

Name	Snapshot ID	Full snapshot size	Volume size	Description	Storage tier	Snapshot status	Started	Progress
Web Data	snap-08113e871db190d03	0 B	100 GB	-	Standard	Completed	2025/11/23 20:34 GMT+5:...	100%

**Select a snapshot above.**

☰ CloudFormation > Stacks > Create stack

Step 1 **Create stack**

Step 2

Step 3

Step 4

Step 5

## Create stack

**Prerequisite - Prepare template**

You can also create a template by scanning your existing resources in the [IaC generator](#).

**Prepare template**

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

Choose an existing template  
Upload or choose an existing template.

Build from Infrastructure Composer  
Create a template using a visual builder.

**Specify template** Info

This [GitHub repository](#) contains sample CloudFormation templates that can help you get started on new infrastructure projects. [Learn more](#)

**Template source**

Selecting a template generates an Amazon S3 URL where it will be stored. A template is a JSON or YAML file that describes your stack's resources and properties.

Amazon S3 URL  
Provide an Amazon S3 URL to your template.

Upload a template file  
Provide your template directly to the console.

Sync from Git  
Sync a template from your Git repository.

**Amazon S3 URL**

`https://c174142a4508396124630421w008199202797-repobucket-ntvl2z4x6v5r.s3.us-east-1.amazonaws.com/cafe-app.yaml`

Amazon S3 template URL

S3 URL: `https://c174142a4508396124630421w008199202797-repobucket-ntvl2z4x6v5r.s3.us-east-1.amazonaws.com/cafe-app.yaml`

[View in Infrastructure Composer](#)

[Cancel](#) [Next](#)

# Automating Infrastructure Deployment with AWS CloudFormation

## Objectives:

Deploying infrastructure in a consistent, reliable manner is difficult. It requires people to follow documented procedures without taking any undocumented shortcuts. It can also be difficult to deploy infrastructure after hours when fewer staff are available. AWS CloudFormation changes this situation by defining infrastructure in a template that can be automatically deployed—even on an automated schedule.

## Task 1: Deploying a networking layer

At the top of the AWS Management Console, in the search box, search for and choose CloudFormation.

Choose Create stack > With new resources (standard) and configure these settings:

### Step 1: Create stack

Prepare template: Choose Template is ready.

Template source: Choose Upload a template file > Choose file, and then choose the lab-network.yaml file that you downloaded.

Choose Next.

### Step 2: Specify stack details

Stack name: lab-network

Choose Next.

### Step 3: Configure stack options

In the Tags section, choose Add new tag and configure the following:

Key: application

Value: inventory

Choose Next.

### Step 4: Review and create

Choose Submit.

Choose the Stack info tab.

Wait for the Status to change to CREATE\_COMPLETE.

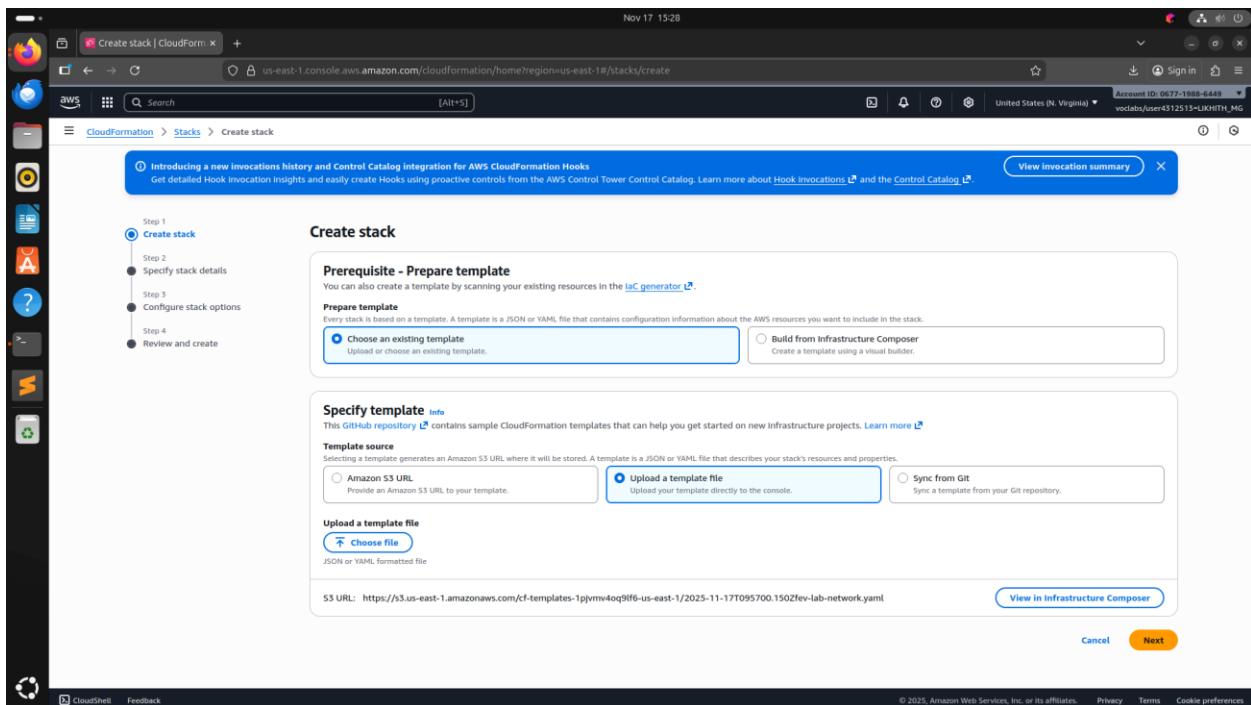
Choose Refresh every 15 seconds to update the display, if necessary.

You can now examine the resources that were created.

Choose the Resources tab.

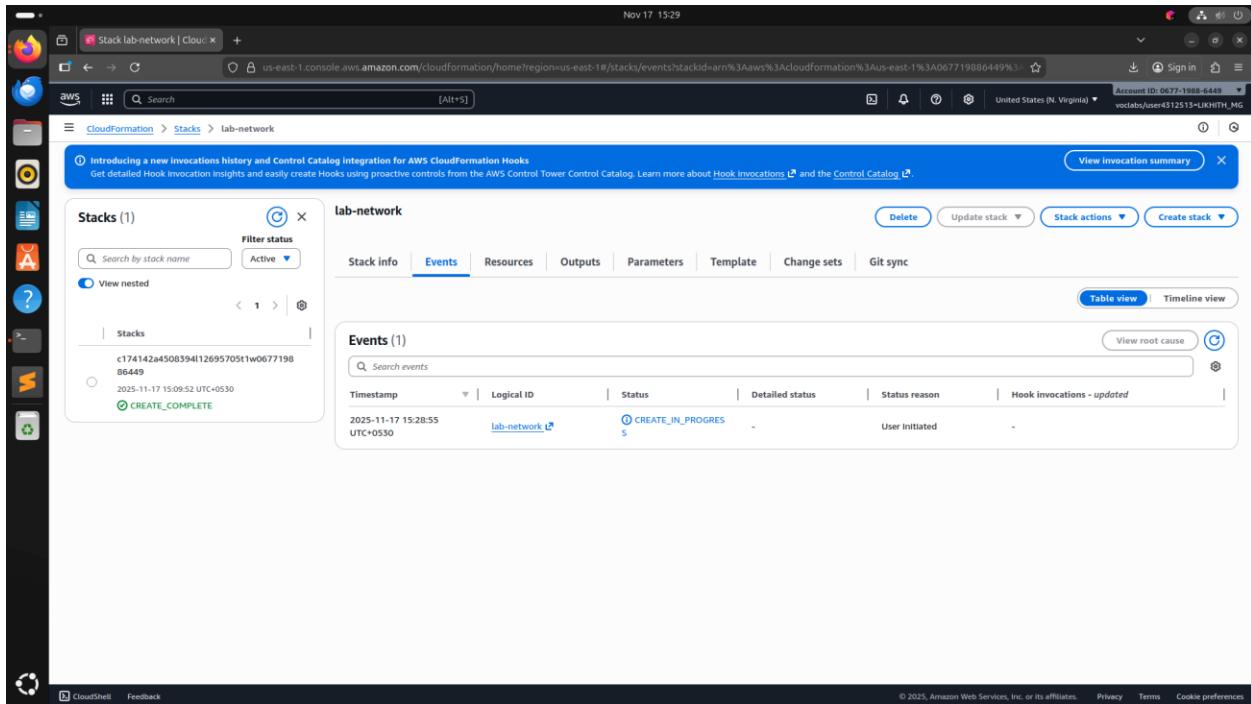
You see a list of the resources that were created by the template.

If the list is empty, update the list by choosing Refresh .



The screenshot shows the 'Specify stack details' step of the CloudFormation stack creation wizard. The left sidebar lists steps: Step 1 (Create stack), Step 2 (Specify stack details, currently selected), Step 3 (Configure stack options), and Step 4 (Review and create). The main area has two sections: 'Provide a stack name' (stack name: lab-network) and 'Parameters' (no parameters defined). At the bottom are 'Cancel', 'Previous', and 'Next' buttons.

The screenshot shows the 'Configure stack options' step of the CloudFormation stack creation wizard. The left sidebar lists steps: Step 1 (Create stack), Step 2 (Specify stack details), Step 3 (Configure stack options, currently selected), and Step 4 (Review and create). The main area includes sections for 'Tags - optional' (adding tags like application:inventory), 'Permissions - optional' (choosing an IAM role like Sample-role-name), and 'Stack failure options' (selecting 'Roll back all stack resources'). A screenshot capture window is visible at the top right.



## Task 2: Deploying an application layer

Create stack > With new resources (standard), and then configure these settings:

### Step 1: Create Stack

Prepare template: Choose Template is ready.

Template source: Choose Upload a template file > Choose file, and then choose the lab-application.yaml file that you downloaded.

Choose Next.

### Step 2: Specify stack details

Stack name: lab-application

Notice the NetworkStackName: lab-network

Choose Next.

The Network Stack Name parameter tells the template the name of the first stack that you created (lab-network), so it can retrieve values from the outputs.

### Step 3: Configure stack options

In the Tags section, choose Add new tag and configure the following:

Key: application

Value: inventory

Choose Next.

#### Step 4: Review and create

Choose Submit.

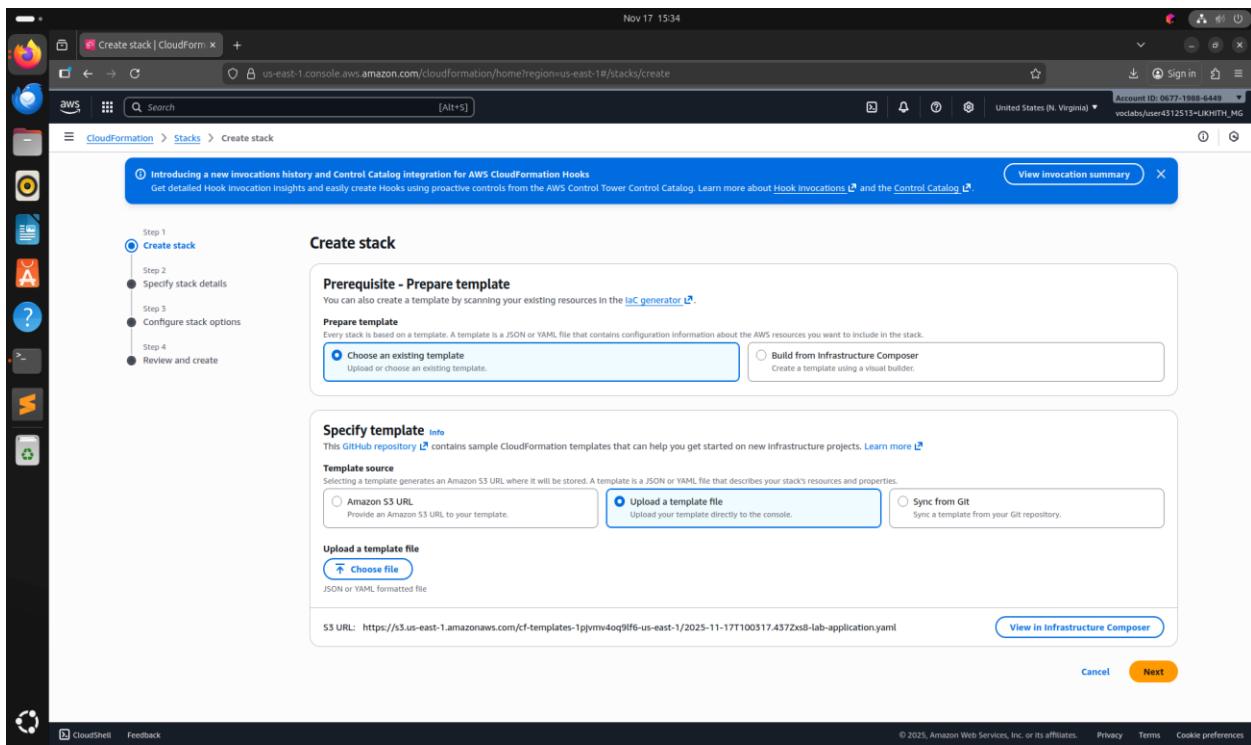
While the stack is being created, examine the details in the Events tab and the Resources tab. You can monitor the progress of the resource-creation process and the resource status.

In the Stack info tab, wait for the Status to change to CREATE\_COMPLETE.

Your application is now ready!

Choose the Outputs tab.

Copy the URL that is displayed, open a new web browser tab, paste the URL, and press ENTER.



Introducing a new invocations history and Control Catalog integration for AWS CloudFormation Hooks  
Get detailed Hook invocation insights and easily create Hooks using proactive controls from the AWS Control Tower Control Catalog. Learn more about Hook Invocations and the Control Catalog.

**lab-application**

**Outputs (1)**

Key	Value	Description	Export name
URL	<a href="http://ec2-98-88-71-111.compute-1.amazonaws.com">http://ec2-98-88-71-111.compute-1.amazonaws.com</a>	URL of the sample website	-

**Congratulations, you have successfully launched the AWS CloudFormation sample.**

## Task 3: Updating a Stack

At the top of the AWS Management Console, in the search box, search for and choose EC2.

In the left navigation pane, in the Network & Security section, choose Security Groups.

Select the check box for lab-application-WebServerSecurityGroup.

Choose the Inbound rules tab.

Currently, only one rule is in the security group. The rule permits HTTP traffic.

You now return to AWS CloudFormation to update the stack.

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
sgr-091bf52c4bf447537	IPv4	HTTP	TCP	80	0.0.0.0/0	-	

From the Services menu at the top, choose CloudFormation.

Open the context (right-click) menu for the following link and download the updated template to your computer: [lab-application2.yaml](#)

From the Stacks list of the AWS CloudFormation console, choose lab-application.

Choose Update and configure these settings:

Prepare template: Choose Replace current template.

Template source: Choose Upload a template file.

Upload a template file: Choose file, and then choose the lab-application2.yaml file that you downloaded.

Choose Next on each of the next three screens to go to the Review lab-application page.

In the Change set preview section at the bottom of the page, AWS CloudFormation displays the following resources that will be updated:

Choose Submit.

In the Stack info tab, wait for the Status to change to UPDATE\_COMPLETE.

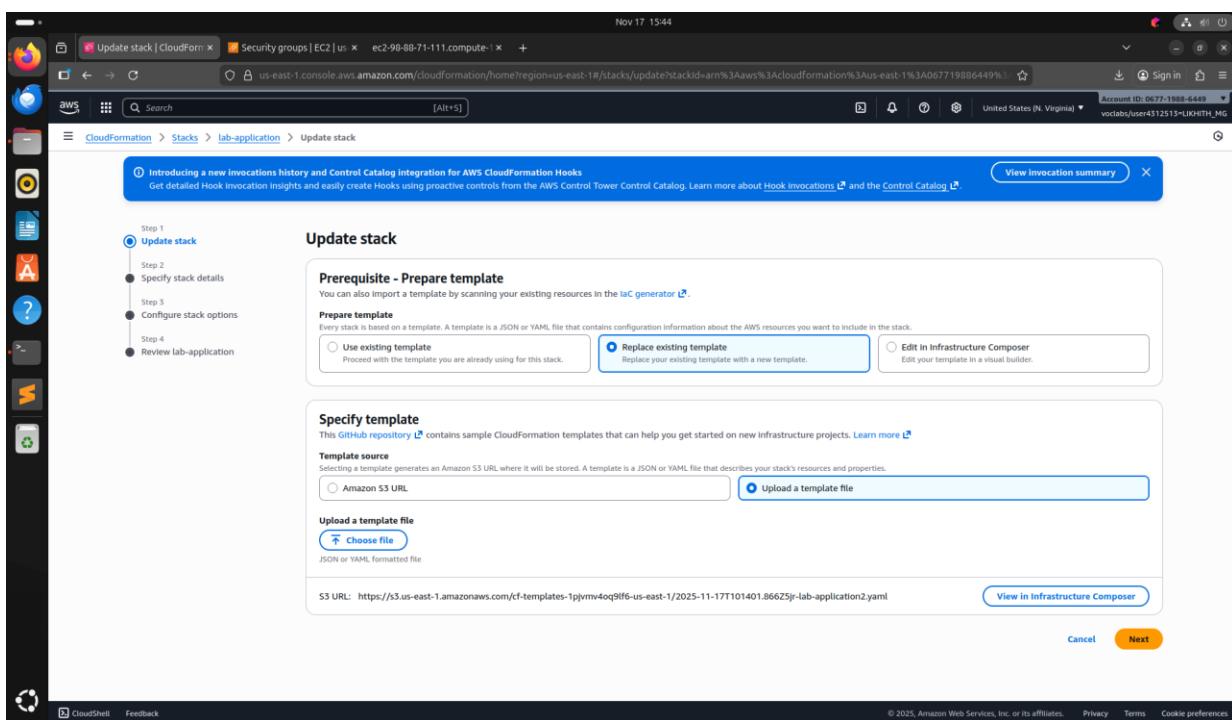
Update the status by choosing Refresh every 15 seconds, if necessary.

You can now verify the change.

Return to the Amazon EC2 console, and from the left navigation pane, choose Security Groups.

In the Security Groups list, choose lab-application-WebServerSecurityGroup.

The Inbound rules tab should display an additional rule that allows HTTPS traffic over TCP port 443.



**Stacks (3)**

**lab-application**

**Events (19)**

Timestamp	Logical ID	Status	Detailed status	Status reason
2025-11-17 15:35:15 UTC+0530	lab-application	UPDATE_IN_PROGRESS	-	User Initiated
2025-11-17 15:36:52 UTC+0530	lab-application	CREATE_COMPLETE	-	-
2025-11-17 15:36:49 UTC+0530	DiskMountPoint	CREATE_COMPLETE	-	-
2025-11-17 15:36:44 UTC+0530	lab-application	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated
2025-11-17 15:36:44 UTC+0530	DiskMountPoint	CREATE_IN_PROGRESS	CONFIGURATION_COMPLETE	Eventual consistency check initiated
2025-11-17 15:36:43 UTC+0530	DiskMountPoint	CREATE_IN_PROGRESS	-	Resource creation initiated
2025-11-17 15:36:41 UTC+0530	DiskMountPoint	CREATE_IN_PROGRESS	-	-

**Security Groups (1/3)**

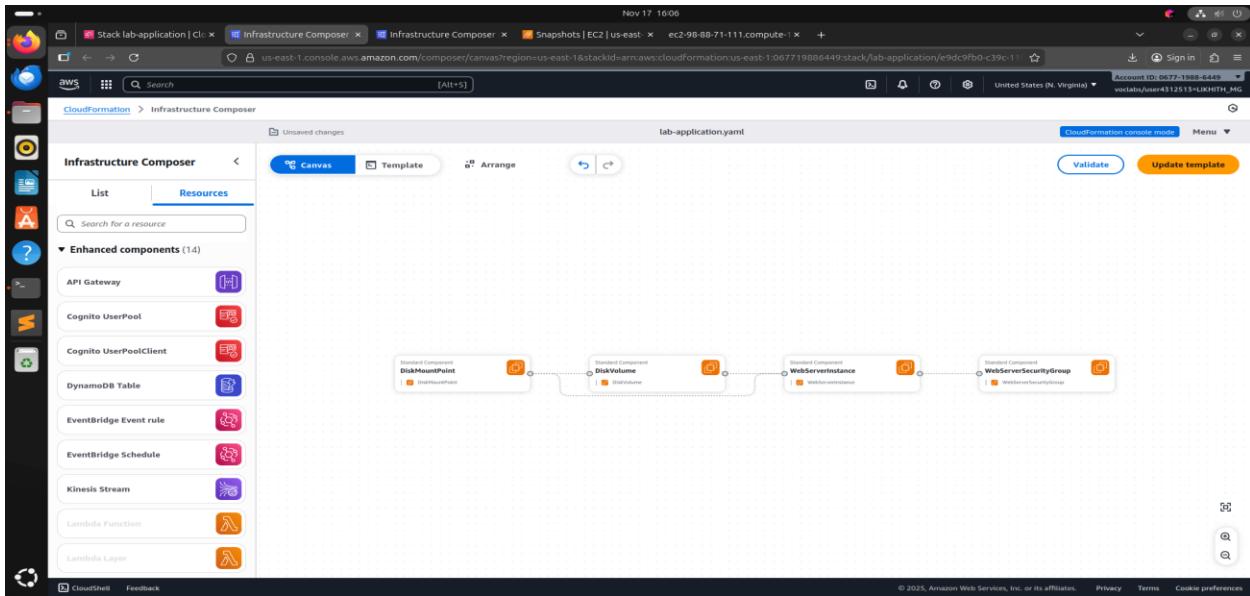
Name	Security group ID	Security group name	VPC ID	Description	Owner
default	sg-0fa5a6862f5207d60	default	vpc-0f2c4f1d5810f2b51	default VPC security group	067719886449
default VPC security group	sg-0fa02bd4771030548	default	vpc-06eedf3e3047a2e31	default VPC security group	067719886449
Web Server Security ...	sg-0335e2c0fc308c976	lab-application-WebServerSecurityGroup-WcKZEiY71nzV	vpc-0f2c4f1d5810f2b51	Enable HTTP Ingress	067719886449

**Inbound rules (2)**

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
-	sgr-091bf52c4bfd47537	IPv4	HTTP	TCP	80	0.0.0.0/0	-
-	sgr-059b0fb0d442b27c5	IPv4	SSH	TCP	22	0.0.0.0/0	-

## Task 4: Exploring templates with AWS CloudFormation Designer

Cloud formation -> SELECT Stack -> Template -> SELECT View Infrastructure Composer



## Task 5: Deleting the stack

AWS CloudFormation console by choosing Close at the top of the Designer page (choose Leave page if prompted).

In the list of stacks, choose the lab-application link.

Choose Delete.

On the Delete stack? dialog box, choose Delete.

You can monitor the deletion process in the Events tab and update the screen by choosing Refresh occasionally. You might also see an events log entry that indicates that the EBS snapshot is being created.

Wait for the stack to be deleted. It will disappear from the stacks list.

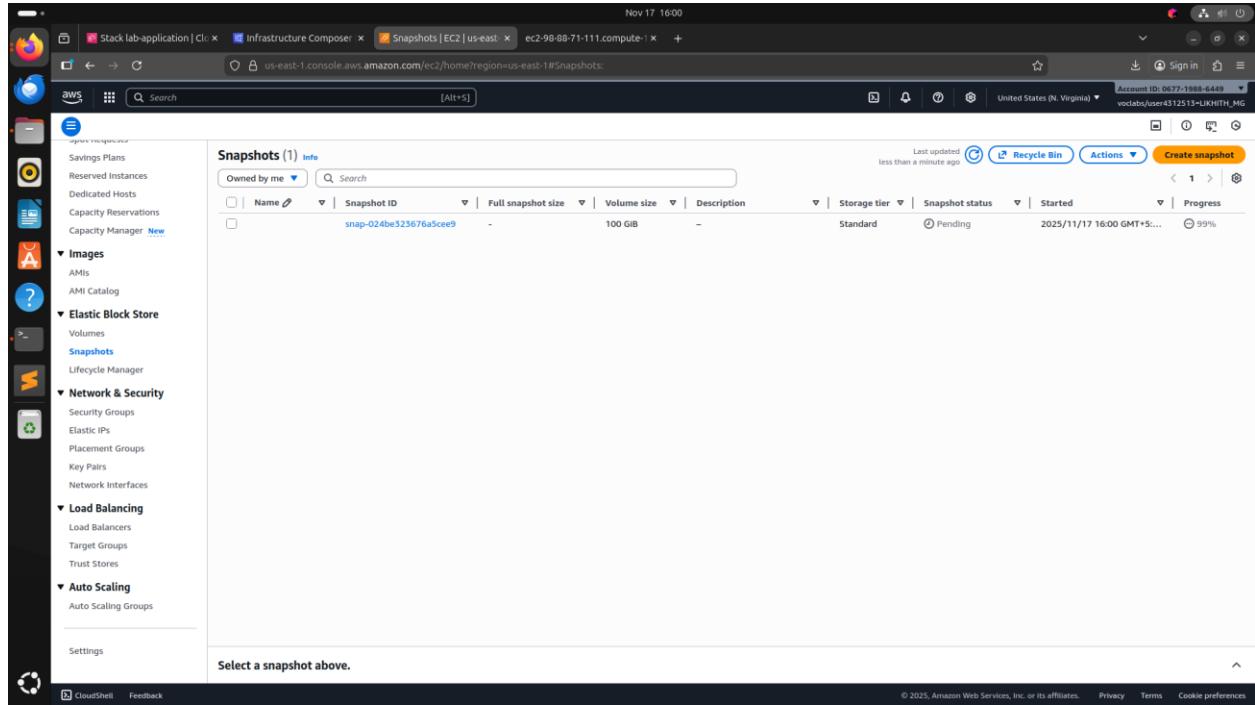
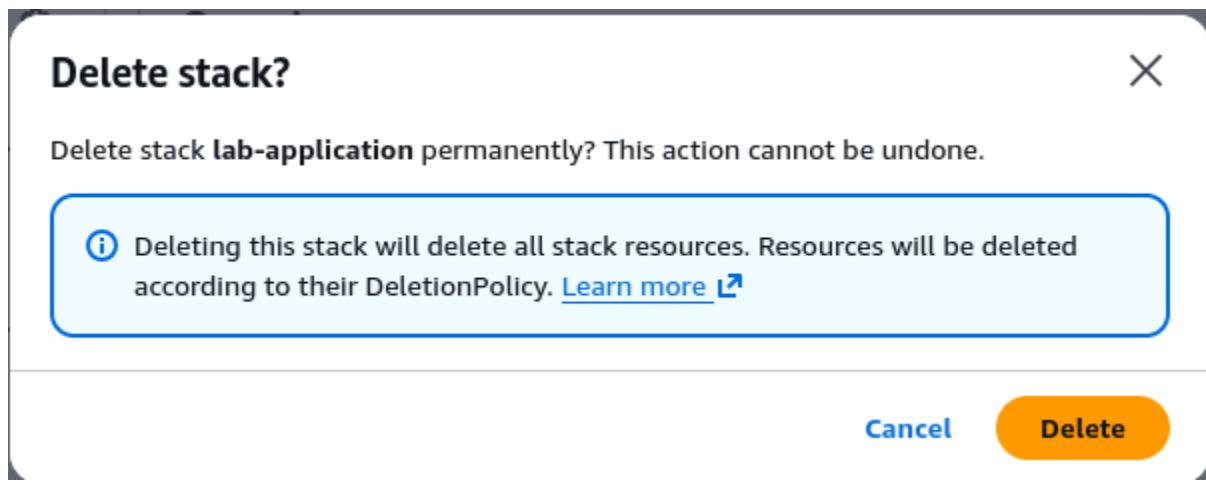
The application stack was removed, but the network stack remained untouched. This scenario reinforces the idea that different teams (for example, the network team or the application team) can manage their own stacks.

You will now verify that a snapshot of the EBS volume was created before the EBS volume was deleted.

From the Services menu, choose EC2.

In the left navigation pane, in the Elastic Block Store section, choose Snapshots.

You see a snapshot Web Data with a Started time in the last few minutes, and it changes to Completed soon.



# Amazon Elastic Compute Cloud (Amazon EC2) Report

**Lab:** Introducing Amazon Elastic Compute Cloud(EC2)

## Objectives

- Launch a web server with termination protection enabled
- Monitor Your EC2 instance
- Modify the security group that your web server is using to allow HTTP access
- Resize your Amazon EC2 instance to scale and enable stop protection
- Explore EC2 limits
- Test stop protection
- Stop your EC2 instance

## Lab Environment Setup

- This lab provides you with a basic overview of launching, resizing, managing, and monitoring an Amazon EC2 instance.
- **Amazon Elastic Compute Cloud (Amazon EC2)** is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.
- Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change.
- Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate themselves from common failure scenarios.

# Lab 3: Introduction to Amazon EC2

## Lab overview and objectives



## Task 1: Launch Your Amazon EC2 Instance

- In the **AWS Management Console** choose **Services**, choose **Compute** and then choose **EC2**.
- Choose the Launch instance menu and select **Launch instance**.

### Step 1: Name and tags

- Name the instance **Web Server**, which creates a tag with key **Name** and value **Web Server**.

### Step 2: AMI selection

- Keep the default **Amazon Linux 2023** AMI selected from the Quick Start list.

### Step 3: Instance type

- Keep the default **t2.micro** instance type with 1 vCPU and 1 GiB memory.

### Step 4: Key pair

- Select the **vockey** key pair for secure login authentication.

### Step 5: Network settings

- Select **Lab VPC**, use **PublicSubnet1**, create a security group named **Web Server security group**, and remove the default inbound rule.

## Step 6: Configure storage

- Keep the default **8 GiB EBS root volume** unchanged.

## Step 7: Advanced details

- Enable **termination protection** and add the provided user-data script to install and start the Apache web server.

## Step 8: Launch the instance

- Choose **Launch instance**, then view your instance and wait until it reaches **Running** with **2/2 status checks passed**.

### Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

#### Name and tags Info

Name

MyEC2-cafestatic

Add additional tags

#### ▼ Application and OS Images (Amazon Machine Image) Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Search our full catalog Including 1000s of application and OS Images

Recents

Quick Start



Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

#### Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type  
ami-0ecb62995f68bb549 (64-bit (x86)) / ami-01bf9fe1e7dc427266e (64-bit (Arm))  
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

#### Description

Ubuntu Server 24.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble Image

Architecture

AMI ID

Publish Date

Username

ubuntu

64-bit (x86)

ami-0ecb62995f68bb549

2025-10-22

Verified provider

## ▼ Instance type [Info](#) | [Get advice](#)

### Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand Windows base pricing: 0.0162 USD per Hour  
On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour On-Demand SUSE base pricing: 0.0116 USD per Hour  
On-Demand RHEL base pricing: 0.026 USD per Hour On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

[Compare instance types](#)

**Additional costs apply for AMIs with pre-installed software**

## ▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

### Key pair name - required

vockey

[Create new key pair](#)

## ▼ Network settings [Info](#)

### VPC - required [Info](#)

vpc-0146cdfc86b20f546 (Lab VPC)  
10.0.0.0/16

[Create](#)

### Subnet [Info](#)

subnet-00ca9a38f9763dc47 PublicSubnet1  
VPC: vpc-0146cdfc86b20f546 Owner: 975050223293 Availability Zone: us-east-1a (use1-az1)  
Zone type: Availability Zone IP addresses available: 1 CIDR: 10.0.1.0/28

[Create new subnet](#)

### Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

### Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

[Create security group](#)

[Select existing security group](#)

### Security group name - required

websg1

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and \_-:/()#,@[]+=&;{}!\$\*

### Description - required [Info](#)

launch-wizard-1 created 2025-11-18T07:12:50.161Z

**Inbound Security Group Rules**

▼ Security group rule 1 (TCP, 22, 45.112.148.130/32)

Type | Info      Protocol | Info      Port range | Info  
 ssh      TCP      22

Source type | Info      Name | Info      Description - optional | Info  
 My IP      Add CIDR, prefix list or security group      e.g. SSH for admin desktop  
 45.112.148.130/32 X

▼ Security group rule 2 (TCP, 80, 0.0.0.0/0)

Type | Info      Protocol | Info      Port range | Info  
 HTTP      TCP      80

Source type | Info      Source | Info      Description - optional | Info  
 Anywhere      Add CIDR, prefix list or security group      e.g. SSH for admin desktop  
 0.0.0.0/0 X

**⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.**

Add security group rule      Advanced network configuration

## In Terminal Execution:

```
ubuntu@ip-10-0-1-8:~$ git clone https://github.com/sreepathysois/Cafe_Dynamic_Website.git
```

```
ubuntu@ip-10-0-1-8:~$ sudo apt-get install apache2 php php-mysql mysql-server mysql-client libapache2-mod-php
```

```
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopcafe$ sudo apt-get update
```

```
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopcafe$ sudo cp -rf * /var/www/html/
```

```
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopcafe$ sudo rm -rf /var/www/html/index.html
```

```
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopcafe$ sudo systemctl restart apache2
```

```
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopcafe$ sudo mysql -u root -p
```

```
mysql> create database cafedb;
```

```
mysql> create user 'msis'@'localhost' identified by 'msis@123';
```

```
mysql> grant all privileges on cafedb .* to 'msis'@'localhost';
```

```
mysql> exit
```

```
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website$ cd mompopdb/
ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopdb$ mysql -u msis -p
mysql> use cafedb;
mysql> source create-db.sql
mysql> exit

ubuntu@ip-10-0-1-8:~/Cafe_Dynamic_Website/mompopdb$ cd
```

```
ubuntu@ip-10-0-1-8:~$ sudo nano /var/www/html/getAppParameters.php
```

```
<?php
    // Get the application environment parameters from the Parameter Store.
    //include ('getAppParameters.php');
    $showServerInfo = "false";
    $timeZone = "America/New_York";
    $currency = "$";
    $db_url = "localhost";
    $db_name = "cafedb";
    $db_user = "msis";
    $db_password = "msis@123";

    // Display the server metadata information if the showServerInfo parameter is
    //include('serverInfo.php');
?>
```

```
ubuntu@ip-10-0-1-8:~$ sudo systemctl restart apache2
```

```
ubuntu@ip-10-0-1-8:~$ sudo systemctl restart mysq
```

## In Web Browser using EC2 Public ip

---

# Mom & Pop Café

---

Home   About Us   Contact Us   Menu   Order History



Mom & Pop Café offers an assortment of delicious and delectable pastries and coffees that will put a smile on your face. From cookies to croissants, tarts and cakes, each treat is especially prepared to excite your tastebuds and brighten your day!

*Pop bakes a rich variety of cookies. Try them all!*



*Tea  
Coffee  
Latte  
Hot Chocolate  
Yes, we have it!*



*Our tarts are always a customer favorite!*



---

---

# Mom & Pop Café

---

Home   **Menu**   Order History

*Pastries*



**Croissant**  
\$1.50  
Fresh, buttery and fluffy... Simply delicious!  
Quantity:



**Donut**  
\$1.00  
We have more than half-a-dozen flavors!  
Quantity:



**Chocolate Chip Cookie**  
\$2.50  
Made with Swiss chocolate with a touch of Madagascar vanilla  
Quantity:



**Muffin**  
\$3.00  
Banana bread, blueberry, cranberry or apple  
Quantity:



**Strawberry Blueberry Tart**  
\$3.50  
Bursting with the taste and aroma of fresh fruit  
Quantity:



**Strawberry Tart**  
\$3.50  
Made with fresh ripe strawberries and a delicious whipped cream  
Quantity:

# AWS Identity and Access Management (IAM)

## 1. Introduction

AWS Identity and Access Management (IAM) is a critical security service in AWS that allows organizations to control access to AWS resources securely. With IAM, administrators can create users, groups, policies, and roles, assigning permissions based on the principle of least privilege, ensuring users can only access what is necessary for their job.

In this lab, we explored pre-created IAM users and groups, examined the attached managed and inline policies, assigned users to groups according to job roles, and tested their access through the IAM sign-in URL. This provides hands-on understanding of how IAM manages secure access in real AWS environments.

## 2. Lab Objectives

- To understand IAM user and group management.
- To explore managed and inline IAM policies.
- To assign users to appropriate groups based on business roles.
- To verify access restrictions through real-time testing using login URLs.
- To observe how permission boundaries affect AWS service access.

## 3. AWS IAM Key Concepts (Explanation)

**a) IAM Users:** IAM Users represent individual identities that can authenticate into AWS using passwords or access keys.

**b) IAM Groups:** IAM Groups are collections of users, allowing permissions to be assigned collectively instead of individually.

**c) IAM Policies:** Policies contain permission rules in JSON format which define:

- **Effect** – Allow or Deny
- **Action** – What operations are permitted (example: s3>ListBucket)
- **Resource** – Which AWS resource(s) the policy applies to (\* or ARN)

Policies are of two types:

- **Managed Policy** – Prebuilt by AWS or admins, reusable across users and groups.
- **Inline Policy** – Attached directly to a single user/group for unique permission use cases.

**d) IAM Roles:** A temporary access identity used by services, applications, or federated users (not used in this lab but important conceptually).

## 4. Business Scenario Summary

User	Assigned Group	Permissions Access
user-1	S3-Support	Read-Only access to Amazon S3
user-2	EC2-Support	Read-Only access to Amazon EC2
user-3	EC2-Admin	View, Start, Stop Amazon EC2 instances

## 5. Task-Wise Lab Explanation

### Task 1: Explore Users and Groups

- Open IAM from AWS console.
- Observed three pre-created users: user-1, user-2, user-3.

User name	Path	Group	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active key age
user-1	/spl66/	0	-	-	15 minutes	-	Active - AKIAZUCNU5Z...	15 minutes
user-2	/spl66/	0	-	-	15 minutes	-	Active - AKIAZUCNU5Z...	15 minutes
user-3	/spl66/	0	-	-	15 minutes	-	Active - AKIAZUCNU5Z...	15 minutes

- Verified that user-1 initially had no permissions and no group membership.
- Observed pre-created groups: EC2-Admin, EC2-Support, S3-Support.

Group name	Users	Permissions	Creation time
EC2-Admin	0	Defined	16 minutes ago
EC2-Support	0	Defined	16 minutes ago
S3-Support	0	Defined	16 minutes ago

Checked permissions for each group:

- EC2-Support → AmazonEC2ReadOnlyAccess (Managed Policy)
- S3-Support → AmazonS3ReadOnlyAccess (Managed Policy)
- EC2-Admin → Inline Policy allowing Describe, Start, Stop EC2 instances

### Task 2: Add Users to Groups

Based on business requirements:

- Added user-1 to S3-Support

The screenshot shows the AWS IAM User Groups page for the 'S3-Support' group. The left sidebar includes sections for Identity and Access Management (IAM), Access management (User groups, Roles, Policies, Identity providers, Account settings, Root access management), Access reports (Access Analyzer, Resource analysis, Unused access, Analyzer settings, Credential report, Organization activity, Service control policies, Resource control policies), IAM Identity Center, and AWS Organizations.

**S3-Support**

**Summary**

- User group name: S3-Support
- Creation time: November 18, 2025, 13:51 (UTC+05:30)
- ARN: arn:aws:iam::661587816049:group/spl66/S3-Support

**Users** | Permissions | Access Advisor

**Users in this group (0)**

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

User name
No resources to display

**Add users**

**Groups** | Last activity | Creation time

**1 user added to this group.**

**S3-Support**

**Summary**

- User group name: S3-Support
- Creation time: November 18, 2025, 13:51 (UTC+05:30)
- ARN: arn:aws:iam::661587816049:group/spl66/S3-Support

**Users (1)** | Permissions | Access Advisor

**Users in this group (1)**

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

User name
user-1

**Remove** | **Add users**

**Groups** | Last activity | Creation time

- Added user-2 to EC2-Support

The screenshot shows the AWS IAM User Groups page for the 'EC2-Support' group. The left sidebar includes sections for Identity and Access Management (IAM), Access management (User groups, Roles, Policies, Identity providers, Account settings, Root access management), Access reports (Access Analyzer, Resource analysis, Unused access, Analyzer settings, Credential report, Organization activity, Service control policies, Resource control policies), IAM Identity Center, and AWS Organizations.

**EC2-Support**

**Summary**

- User group name: EC2-Support
- Creation time: November 18, 2025, 13:51 (UTC+05:30)
- ARN: arn:aws:iam::661587816049:group/spl66/EC2-Support

**Users** | Permissions | Access Advisor

**Users in this group (0)**

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

User name
No resources to display

The screenshot shows the AWS IAM User Groups page for the 'EC2-Support' group. The summary section indicates that one user has been added to the group. The 'Users' tab shows a single user named 'user-2'. The ARN of the group is listed as arn:aws:iam::661587816049:group/spl66/EC2-Support.

- Added user-3 to EC2-Admin

The screenshot shows the AWS IAM User Groups page for the 'EC2-Admin' group. The summary section indicates that no users have been added to the group. The 'Users' tab shows a search bar and a note stating 'No resources to display'. The ARN of the group is listed as arn:aws:iam::661587816049:group/spl66/EC2-Admin.

The screenshot shows the AWS IAM User Groups page for the 'EC2-Admin' group again. This time, it displays a message indicating that one user has been added to the group. The 'Users' tab shows a single user named 'user-3'. The ARN of the group is listed as arn:aws:iam::661587816049:group/spl66/EC2-Admin.

Finally verified that each group displayed 1 assigned user.

### Task 3: Sign-In and Permissions Testing

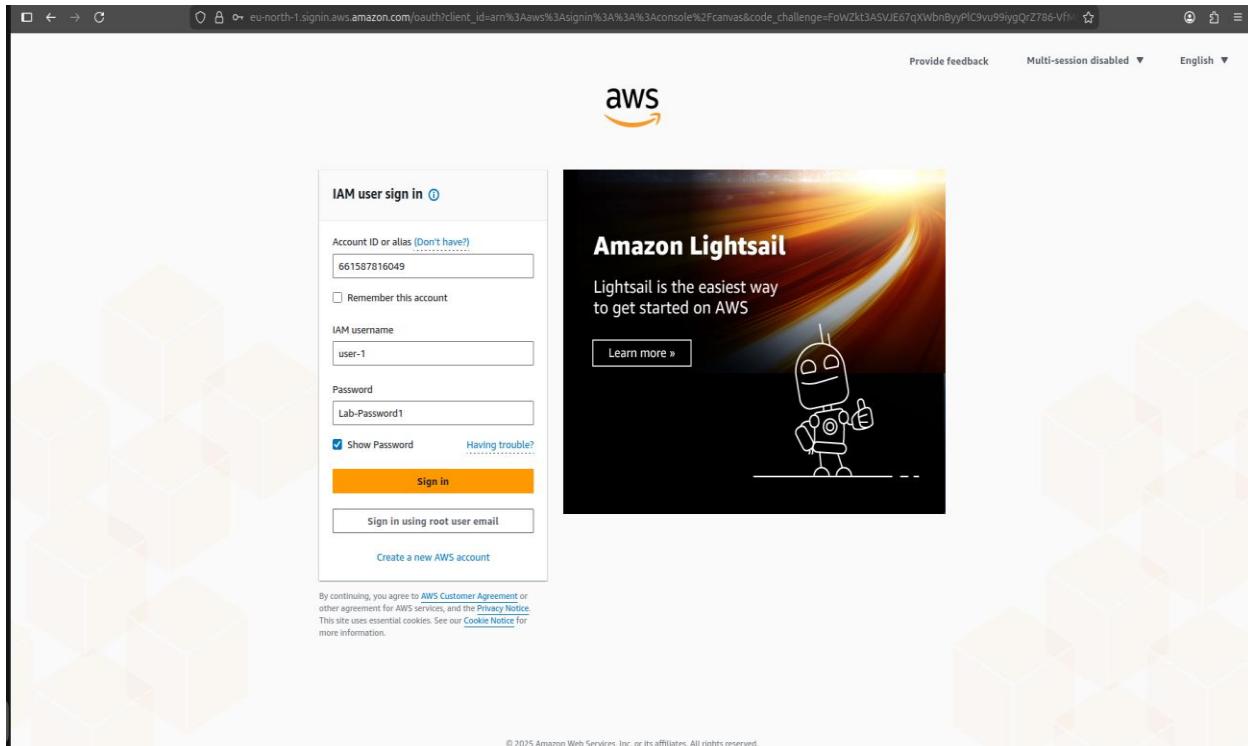
- In the navigation pane on the left, choose the Dashboard. On the right side, a Sign-in URL for IAM users in this account is displayed. The sign-in link shown in the lab environment was: <https://661587816049.signin.aws.amazon.com/console>

The screenshot shows the AWS IAM Dashboard. On the left, there's a sidebar with navigation links like 'Identity and Access Management (IAM)', 'Access management', 'Access reports', and 'AWS Organizations'. The main area has a 'IAM Dashboard' card with statistics: 3 User groups, 4 Users, 13 Roles, 1 Policies, and 0 Identity providers. Below this is a 'What's new' section with a list of recent changes. To the right, there's an 'AWS Account' summary with the Account ID (661587816049) and a 'Sign-in URL for IAM users in this account' link: <https://661587816049.signin.aws.amazon.com/console>. There are also sections for 'Tools' (Policy simulator) and 'Additional information' (Security best practices in IAM).

- This link can be used to sign-in to the AWS Account you are currently using. Copy the Sign-in URL for IAM users in this account to a text editor. Open a private (Incognito) window.

The screenshot shows a Firefox browser window with a dark theme. The address bar shows the URL: <https://661587816049.signin.aws.amazon.com/console>. A modal window titled 'Next-level privacy on mobile' is open, featuring an image of a smartphone and the text: 'Firefox Focus clears your history every time while blocking ads and trackers.' It includes a 'Download Firefox Focus' button. Below the modal, there's a note: 'Firefox clears your search and browsing history when you close all private windows, but this doesn't make you anonymous.' with a 'Learn more' link.

Logged in as user-1



Successfully accessed S3 and viewed bucket list (read-only)

Amazon S3

General purpose buckets [All AWS Regions](#) | Directory buckets

**General purpose buckets (1/1) [Info](#)**

Buckets are containers for data stored in S3.

Name	AWS Region	Creation date
samplebucket--7a82d9c0	US East (N. Virginia) us-east-1	November 18, 2025, 13:50:51 (UTC+05:50)

[Copy ARN](#) [Empty](#) [Delete](#) [Create bucket](#)

**Account snapshot [Info](#)** [View dashboard](#)

Updated daily

Storage Lens provides visibility into storage usage and activity trends.

**External access summary - new [Info](#)**

Updated daily

External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

**Amazon S3**

- General purpose buckets
- Directory buckets
- Table buckets
- Vector buckets
- Access Grants
- Access Points (General Purpose Buckets, FSx file systems)
- Access Points (Directory Buckets)
- Object Lambda Access Points
- Multi-Region Access Points
- Batch Operations
- IAM Access Analyzer for S3

Block Public Access settings for this account

**Storage Lens**

- Dashboards
- Storage Lens groups
- AWS Organizations settings

Feature spotlight: [11](#)

AWS Marketplace for S3

Tried accessing EC2 but received "not authorized" error

Screenshot of the AWS EC2 Instances page. The URL is [us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances). The account ID is 6615-8781-6049, and the region is United States (N. Virginia). The user is 'user-1'. A red box highlights an error message: 'You are not authorized to perform this operation. User: arn:aws:iam::661587816049:user/spl66/user-1 is not authorized to perform: ec2:DescribeInstances because no identity-based policy allows the ec2:DescribeInstances action'. The left sidebar shows navigation links for EC2, Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AM Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, and Key Pairs.

Logged in as user-2

Screenshot of the AWS IAM user sign-in page. The URL is [eu-north-1.signin.aws.amazon.com/auth/client\\_id-arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code\\_challenge=F6mdsXbbNXJmbdjWDaSoKSGQkdxgHzmo-ydellA](https://eu-north-1.signin.aws.amazon.com/auth/client_id-arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=F6mdsXbbNXJmbdjWDaSoKSGQkdxgHzmo-ydellA). The page includes a 'Provide feedback' link, 'Multi-session disabled' indicator, and an 'English' language dropdown. The main form fields are: Account ID or alias (661587816049), Remember this account (unchecked), IAM username (user-2), Password (Lab-Password2), Show Password (checked), Having trouble? (link), Sign in (button), and Sign in using root user email (link). Below the form is a note about AWS Customer Agreement and Privacy Notice, and a link to the Cookie Notice. To the right is an advertisement for Amazon Lightsail with the text 'Lightsail is the easiest way to get started on AWS' and a 'Learn more' button. The bottom of the page shows a copyright notice for 2025 Amazon Web Services, Inc. and a transfer status bar: 'Transferring data from eu-north-1.signin.aws.amazon.com...'.

Successfully viewed EC2 instance (read-only)

**Instances (2) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
LabHost	i-0678558a9257a3d57	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	ec2-100-24-119-163.co...	100.24.119.163	-
Bastion Host	i-071650efde8907a8a	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	ec2-44-200-160-192.co...	44.200.160.192	-

**Select an instance**

## Attempted to stop EC2 instance but got permission denied

**Instances (1/2) Info**

Name	Instance ID	Public IPv4 address	Private IPv4 addresses
i-0678558a9257a3d57 (LabHost)	i-0678558a9257a3d57	100.24.119.163   open address	10.1.11.152

**Details** **Status and alarms** **Monitoring** **Security** **Networking** **Storage** **Tags**

**Instance summary**

Instance ID	Public IPv4 address
i-0678558a9257a3d57	100.24.119.163   open address
IPv6 address	Instance state
-	Running
Hostname type	Private IP DNS name (IPv4 only)
IP name: ip-10-1-11-152.ec2.internal	ip-10-1-11-152.ec2.internal
Answer private resource DNS name	Instance type
-	t2.micro
	Private IPv4 addresses
	Public DNS
	EC2 instance
	Elastic IP addresses
	-

Tried accessing S3 but was denied

Screenshot of the AWS S3 Bucket creation page:

The URL is [us-east-1.console.aws.amazon.com/s3/bucket/create?region=us-east-1](https://us-east-1.console.aws.amazon.com/s3/bucket/create?region=us-east-1)

**Default encryption** (Info): Server-side encryption is automatically applied to new objects stored in this bucket.

**Encryption type** (Info): Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the Storage tab of the [Amazon S3 pricing page](#).

- Server-side encryption with Amazon S3 managed keys (SSE-S3)
- Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)

**Bucket Key**: Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

- Disable
- Enable

**Advanced settings**

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

**Failed to create bucket**: To create a bucket, the `s3:CreateBucket` permission is required. [View your permissions](#) in the [IAM console](#). [Identity and Access Management in Amazon S3](#)

[Diagnose with Amazon Q](#)

[Cancel](#) [Create bucket](#)

Logged in as user-3

Screenshot of the AWS IAM User Sign In page:

The URL is [eu-north-1.sigin.aws.amazon.com/auth/client\\_id=arn%3aws%3asignin%3A%3A%3Aconsole%2Fcanvas&code\\_challenge=J0t65dxdmU9gfEhaZ3wsdVjDYLZGe8OHQsglm](https://eu-north-1.sigin.aws.amazon.com/auth/client_id=arn%3aws%3asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=J0t65dxdmU9gfEhaZ3wsdVjDYLZGe8OHQsglm)

**IAM user sign in**

Account ID or alias (Don't have?):

Remember this account

IAM username:

Password:

Show Password [Having trouble?](#)

[Sign in](#)

[Sign in using root user email](#)

[Create a new AWS account](#)

By continuing, you agree to [AWS Customer Agreement](#) or other agreement for AWS services, and the [Privacy Notice](#). This site uses essential cookies. See our [Cookie Notice](#) for more information.

Provide feedback Multi-session disabled English

**Amazon Lightsail**: Lightsail is the easiest way to get started on AWS. [Learn more](#)

© 2025 Amazon Web Services, Inc. or its affiliates. All rights reserved.

Accessed EC2 and successfully stopped the LabHost instance

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like EC2, Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and Key Pairs. The main area displays two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 IP	Elastic IP
LabHost	i-0678558a9257a3d57	Stopping	t2.micro	2/2 checks passed	User: arn:aws:	us-east-1a	ec2-100-24-119-163.co...	100.24.119.163	-
Bastion Host	i-071650eefed8907a8a	Running	t2.micro	2/2 checks passed	User: arn:aws:	us-east-1a	ec2-44-200-160-192.co...	44.200.160.192	-

A green banner at the top says "Successfully initiated stopping of i-0678558a9257a3d57". Below the table, there's a detailed view for the LabHost instance:

**i-0678558a9257a3d57 (LabHost)**

**Details** (selected) | Status and alarms | Monitoring | Security | Networking | Storage | Tags

**Instance summary**

Instance ID	I-0678558a9257a3d57	Public IPv4 address	100.24.119.163   open address
IPv6 address	-	Instance state	Stopping
Hostname type	IP name: ip-10-1-11-152.ec2.internal	Private IP DNS name (IPv4 only)	ip-10-1-11-152.ec2.internal
Answer private resource DNS name	-	Instance type	t2.micro
		Private IPv4 addresses	10.1.11.152
		Public DNS	ec2-100-24-119-163.compute-1.amazonaws.com   open address
		Elastic IP addresses	-

Permissions worked as per admin role design

## 6. Conclusion

Through this lab, we successfully learned how IAM enables secure access control in AWS environments. We explored and analyzed IAM users, groups, and policies, assigned users to groups based on job roles, and verified permission restrictions by testing actions in AWS Management Console.

This demonstrates how IAM enforces secure, role-based access, ensuring every individual has only the required level of access, maintaining security and operational integrity.

# AWS Lambda Stopinator Lab Report

**Lab:** AWS Lambda – Scheduled EC2 Stopinator

## Objective

The objective of this lab was to create an AWS Lambda function that automatically stops a specified EC2 instance at a scheduled interval using Amazon EventBridge (CloudWatch Events) as the trigger. This demonstrates serverless automation in AWS.

## Lab Environment Setup

- AWS Management Console was used for this lab.
- A timer-based lab session was started using the “Start Lab” button.
- Pop-up windows were allowed in the browser to open the AWS Management Console in a new tab.
- Resources were named exactly as specified in the instructions to ensure the lab scoring script works properly.

The screenshot shows a web browser window with the URL [labs.vocareum.com/main/main.php?m=clabide&mode=s&asnid=4500351&stepid=4500352&hideNavBar=1](https://labs.vocareum.com/main/main.php?m=clabide&mode=s&asnid=4500351&stepid=4500352&hideNavBar=1). The page title is "Lab overview". It contains a flow diagram with three nodes: "EventBridge event" (represented by a sun icon), "Lambda function triggered" (represented by a Lambda icon with a checkmark), and "EC2 instance stopped" (represented by a red octagonal stop sign). Below the diagram is a descriptive text block: "In this hands-on activity, you will create an AWS Lambda function. You will also create an Amazon EventBridge event to trigger the function every minute. The function uses an AWS Identity and Access Management (IAM) role. This IAM role allows the function to stop an Amazon Elastic Compute Cloud (Amazon EC2) instance that is running in the Amazon Web Services (AWS) account." There are sections for "Duration" (30 minutes), "AWS service restrictions", and "Accessing the AWS Management Console".

## Task 1: Create a Lambda Function

### Steps:

1. Navigated to **AWS Lambda** in the AWS Console.
2. Chose **Create a function → Author from scratch**.

3. Configured the function:

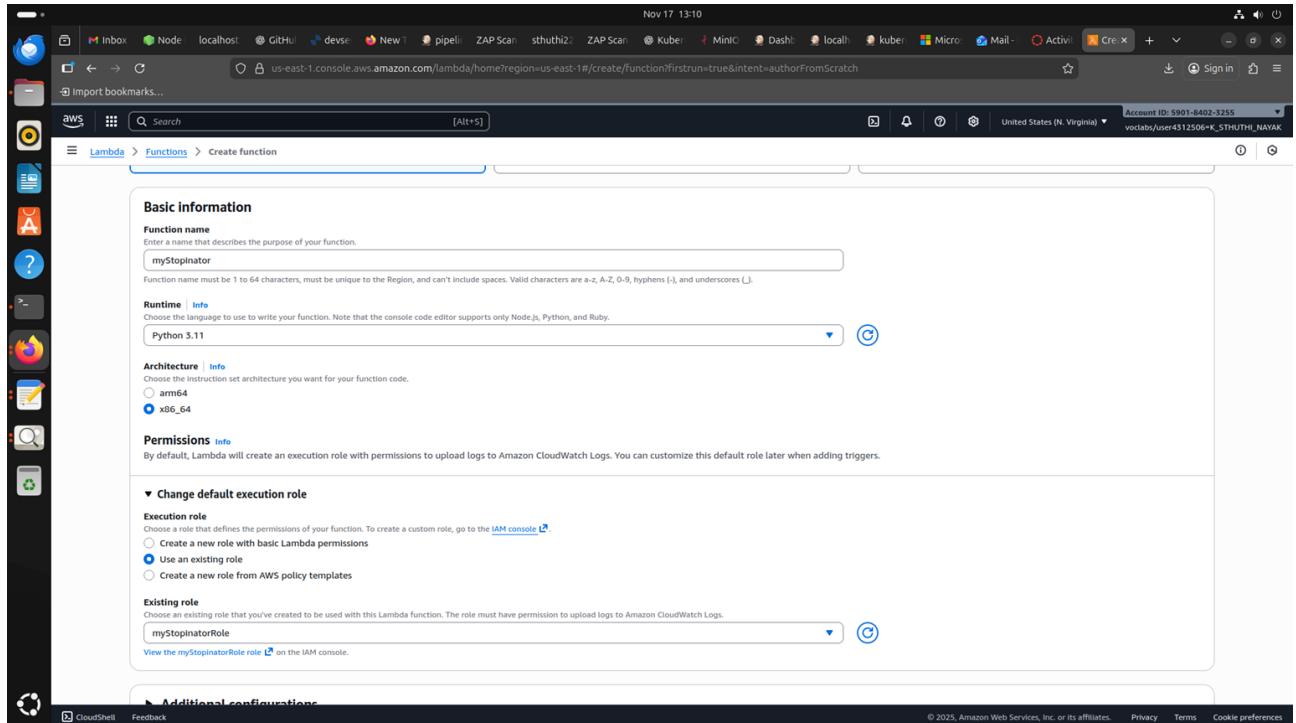
- **Function name:** myStopinator
- **Runtime:** Python 3.11

4. Configured the execution role:

- **Execution role:** Use an existing role
- **Existing role:** myStopinatorRole

5. Clicked **Create function**.

**Outcome:** Lambda function myStopinator successfully created.

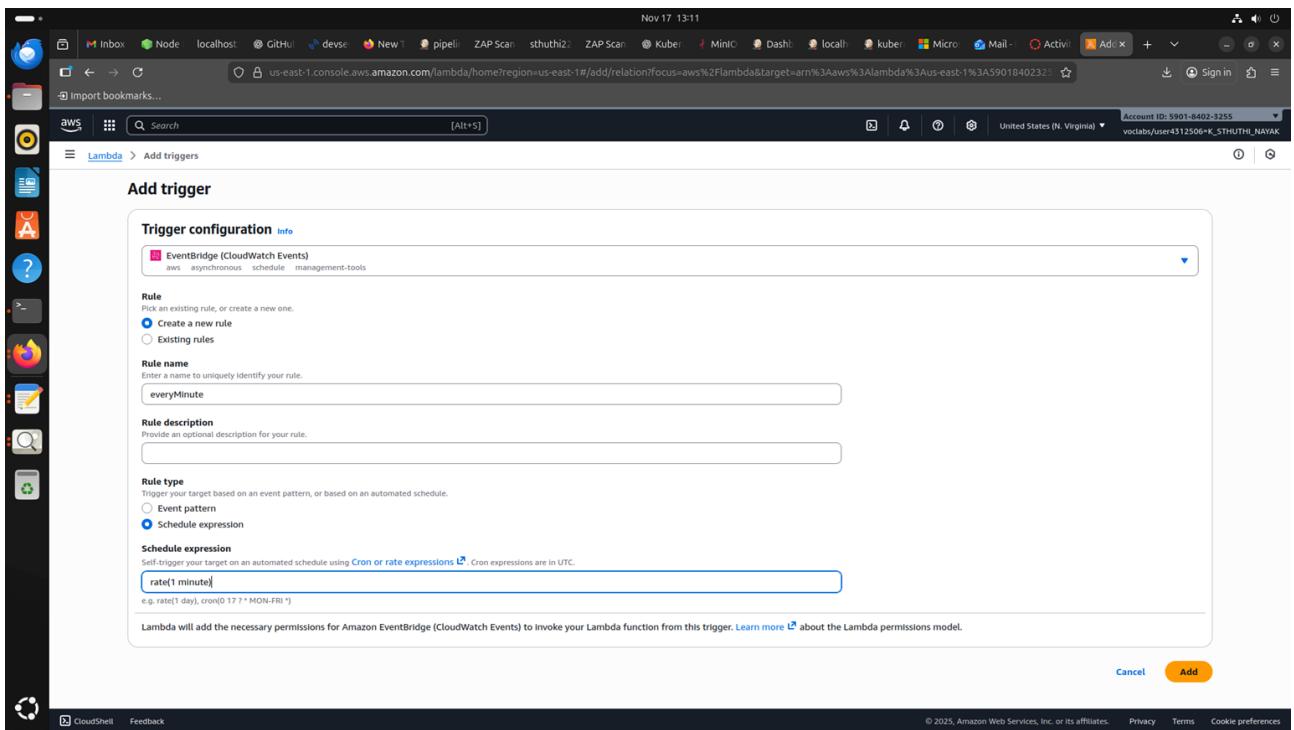


## Task 2: Configure the Trigger

**Steps:**

1. In the Lambda console, clicked **Add trigger**.
2. Selected **EventBridge (CloudWatch Events)**.
3. Created a new rule:
  - **Rule name:** everyMinute
  - **Rule type:** Schedule expression
  - **Schedule expression:** rate(1 minute)
4. Clicked **Add** to attach the trigger to the Lambda function.

**Outcome:** Lambda function is now scheduled to run every minute.



## Task 3: Configure the Lambda Function Code

### Steps:

1. Opened `lambda_function.py` under the **Code** tab.
2. Replaced the default code with the following Python script:

```
import boto3

region = '<REPLACE_WITH_REGION>'
instances = ['<REPLACE_WITH_INSTANCE_ID>']

ec2 = boto3.client('ec2', region_name=region)

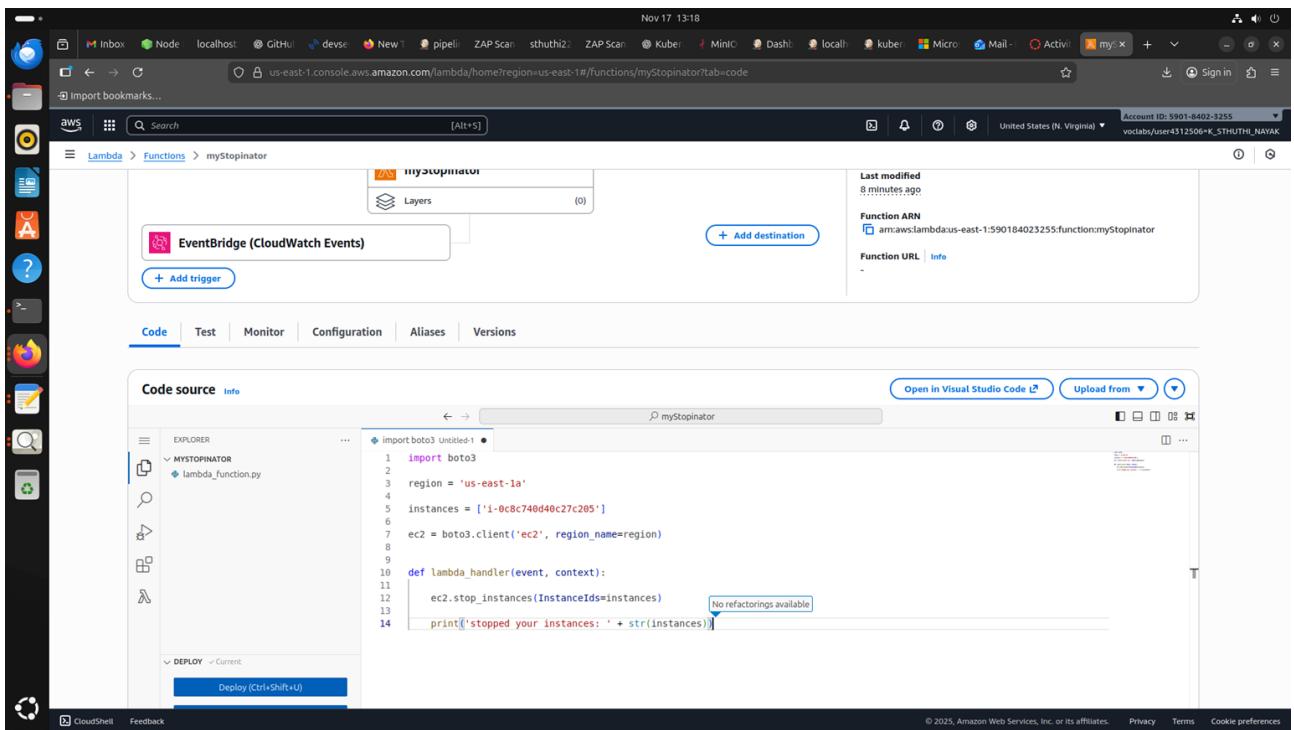
def lambda_handler(event, context):
    ec2.stop_instances(InstanceIds=instances)
    print('stopped your instances: ' + str(instances))
```

3. Replaced placeholders:

- <REPLACE\_WITH\_REGION> → e.g., 'us-east-1'
- <REPLACE\_WITH\_INSTANCE\_ID> → Actual EC2 instance ID of instance1

4. Saved the changes and clicked **Deploy**.

**Outcome:** Lambda function code is configured to stop the specified EC2 instance every minute.



## Task 4: Verify Lambda Function Execution

### Steps:

1. Navigated to **Monitor** tab in Lambda to check invocation metrics.
2. Observed the function invocation count and success rate.
3. Checked the EC2 console to verify the instance was stopped.
4. Attempted to restart the instance.

### Observation:

- The instance stops automatically again within 1 minute due to the scheduled Lambda trigger.

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed, and the main area displays a table of instances. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4 IP, and Elastic IP. There are two rows:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 IP	Elastic IP
Bastion Host	I-0404d5406c32498bd	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-54-235-6-84.comp...	54.235.6.84	-
Instance1	I-08c740d40c27c205	Stopped	t2.micro	-	View alarms +	us-east-1a	-	98.81.89.141	-

Below the table, there is a section titled "Select an instance" with a dropdown menu.

## Conclusion

- The lab successfully demonstrated automation of EC2 instance management using AWS Lambda and EventBridge.
- Serverless functions eliminate the need for a dedicated server to run scheduled tasks.
- Proper naming conventions and role permissions are essential for automation scripts to function correctly.

# IMPLEMENTING SERVERLESS ARCHITECTURE ON AWS

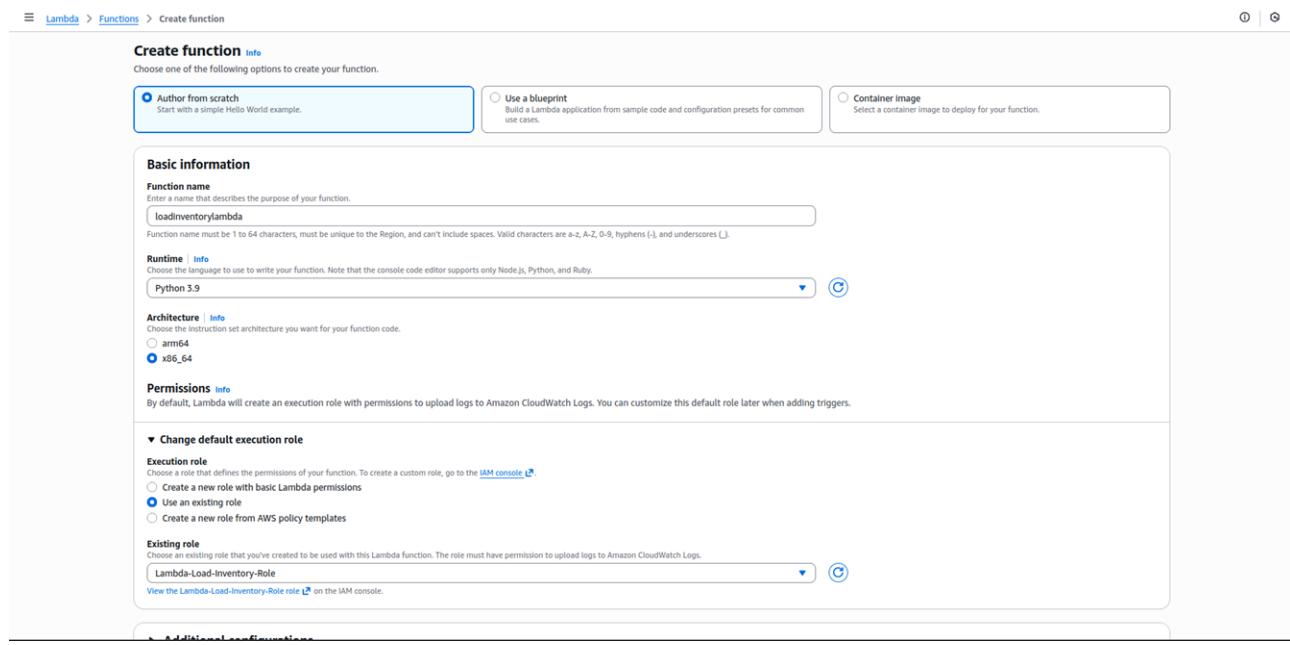
## STEP 1: CREATE A LAMBDA FUNCTION

Function name: loadinventorylamda

Runtime: python 3.9

Change execution role: use an existing role

Choose: lambda\_load\_inventory\_role



In the **Code source** section, in the **Environment** pane, choose **lambda\_function.py**.

In the code editor for the **lambda\_function.py** file, delete all the default code.

In the **Code source** editor, copy and paste the following code:

```
Load-Inventory Lambda function

# This function is invoked by an object being created in an Amazon S3 bucket.

# The file is downloaded and each line is inserted into a DynamoDB table.

import json, urllib, boto3, csv

# Connect to S3 and DynamoDB

s3 = boto3.resource('s3')

dynamodb = boto3.resource('dynamodb')

# Connect to the DynamoDB tables

inventoryTable = dynamodb.Table('Inventory');

# This handler is run every time the Lambda function is invoked

def lambda_handler(event, context):
```

```

# Show the incoming event in the debug log
print("Event received by Lambda function: " + json.dumps(event, indent=2))

# Get the bucket and object key from the Event
bucket = event['Records'][0]['s3']['bucket']['name']
key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])

localFilename = '/tmp/inventory.txt'

# Download the file from S3 to the local filesystem
try:
    s3.meta.client.download_file(bucket, key, localFilename)
except Exception as e:
    print(e)

print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the same region as this function.'.format(key, bucket))

raise e

# Read the Inventory CSV file
with open(localFilename) as csvfile:
    reader = csv.DictReader(csvfile, delimiter=',')
    # Read each row in the file
    rowCount = 0
    for row in reader:
        rowCount += 1
        # Show the row in the debug log
        print(row['store'], row['item'], row['count'])
        try:
            # Insert Store, Item and Count into the Inventory table
            inventoryTable.put_item(
                Item={
                    'Store': row['store'],
                    'Item': row['item'],
                    'Count': int(row['count'])})
        except Exception as e:
            print(e)
            print("Unable to insert data into DynamoDB table".format(e))
    # Finished!
    return "%d counts inserted" % rowCount

```

## STEP 2: CREATE S3 BUCKET

Name: myinventory

Navigate to Properties→Event notification→Create event notification

Event Name:Inventory load

Event Types:Object creation

Destination:Lambda Function

## STEP 3:UPLOAD THE CSV FILES INTO S3

The screenshot shows the AWS S3 'Upload' interface for the 'myinventory' bucket. It includes sections for 'Upload' (info), 'Destination' (info), 'Permissions', and 'Properties'. The 'Files and folders' section lists six CSV files: 'Inventory-berlin.csv', 'Inventory-calcutta.csv', 'Inventory-karachi.csv', 'Inventory-pusan.csv', 'Inventory-shanghai.csv', and 'Inventory-springfield.csv', each with its type (text/csv) and size (e.g., 140.0 B, 150.0 B, etc.). The 'Destination' section shows the target as 's3://myinventory12321'. The 'Upload' button is at the bottom right.

## STEP 4: DYNAMO DB

-Explore the items.

-Items will be displayed.

Inventory

Code

Logs

Inventory-2025-11-16T23:48:06-0800

## STEP 5: DASHBOARD

Go to AWS I details

Cloud Access

AWS CLI: Show

Cloud Labs

Remaining session time: 02:19:16(140 minutes)

Session started at: 2025-11-16T23:48:06-0800

Session to end at: 2025-11-17T02:48:06-0800

Accumulated lab time: 11:41:00 (701 minutes)

No running instance

SSH key Show Download PEM Download PPK

AWS SSO Download URL

Dashboard https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-3-89090/18-lab-mod14-guided-Lambda/s3/web/inventory.htm?region=us-east-1&poolId=us-east-1:df383338-2b3c-4379-9cd3-7bafb21c5307

IdentityPoolId us-east-1:df383338-2b3c-4379-9cd3-7bafb21c5307

Access the dashboard link.

## STEP 6: CREATE ANOTHER LAMBDA FUNCTION

Create function: Storenotification

Runtime:python 3.9

use existing role: lambda\_check\_stock\_role

create function

:tion

**Create function** Info

Choose one of the following options to create your function.

Author from scratch  
Start with a simple Hello World example.

Use a blueprint  
Build a Lambda application from sample code and configuration presets for common use cases.

Container image  
Select a container image to deploy for your function.

**Basic information**

**Function name**  
Enter a name that describes the purpose of your function.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).

**Runtime** Info  
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.  
 Python 3.9 ↻

**Architecture** Info  
Choose the instruction set architecture you want for your function code.  
 arm64  
 x86\_64

**Permissions** Info  
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

**▼ Change default execution role**

**Execution role**  
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).  
 Create a new role with basic Lambda permissions  
 Use an existing role  
 Create a new role from AWS policy templates

**Existing role**  
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.  
 ↻

View the Lambda-Check-Stock-Role role [on the IAM console](#).

**► Additional configurations**  
Use additional configurations to set up networking, security, and governance for your function. These settings help secure and customize your Lambda function deployment.

Cancel Create function

In the **Code source** section, in the **Environment** pane, choose **lambda\_function.py**.

In the code editor for the **lambda\_function.py** file, delete all the default code.

In the **Code source** editor, copy and paste the following code:

```

# Stock Check Lambda function
#
# This function is invoked when values are inserted into the Inventory DynamoDB table.
# Inventory counts are checked and if an item is out of stock, a notification is sent to an SNS Topic.
import json, boto3
# This handler is run every time the Lambda function is invoked
def lambda_handler(event, context):
    # Show the incoming event in the debug log
    print("Event received by Lambda function: " + json.dumps(event, indent=2))
    # For each inventory item added, check if the count is zero
    for record in event['Records']:
        newImage = record['dynamodb'].get('NewImage', None)
        if newImage:
            count = int(record['dynamodb']['NewImage']['Count'][0])
            if count == 0:
                store = record['dynamodb']['NewImage']['Store'][0]
                item = record['dynamodb']['NewImage']['Item'][0]
                # Construct message to be sent
                message = store + ' is out of stock of ' + item
                print(message)
                # Connect to SNS
                sns = boto3.client('sns')
                alertTopic = 'NoStock'
                snsTopicArn = [t['TopicArn'] for t in sns.list_topics()['Topics']
                               if t['TopicArn'].lower().endswith(':'+alertTopic.lower())][0]
                # Send message to SNS
                sns.publish(
                    TopicArn=snsTopicArn,
                    Message=message,
                    Subject='Inventory Alert!',
                    MessageStructure='raw'
                )
            # Finished!
    return 'Successfully processed {} records.'.format(len(event['Records']))

```

## STEP 7: SIMPLE NOTIFICATION SERVICE

Create topic: standard

Name: stock

Create topic.

Create Subscription

**Create subscription**

**Details**

**Topic ARN**  
arn:aws:sns:us-east-1:767397929468:no\_stock

**Protocol**  
The type of endpoint to subscribe  
Email

**Endpoint**  
An email address that can receive notifications from Amazon SNS.  
sudheerkumar04g@gmail.com

After your subscription is created, you must confirm it. [Info](#)

**Subscription filter policy - optional**  
This policy filters the messages that a subscriber receives.

**Redrive policy (dead-letter queue) - optional**  
Send undeliverable messages to a dead-letter queue.

[Cancel](#) [Create subscription](#)

Go mail and confirm subscription.

## Step 8: Trigger function

Go to lambda

Add trigger :dynamodb

Dynamo table:inventory

**Add trigger**

**Trigger configuration** [Info](#)

**DynamoDB** [aws](#) [database](#) [event-source-mapping](#) [nosql](#) [polling](#)

**DynamoDB table**  
Choose or enter the ARN of a DynamoDB table.  
 [X](#) [@](#)

**Event poller configuration**

**Activate trigger**  
Select to activate the trigger now. Keep unchecked to create the trigger in a deactivated state for testing (recommended).

**Enable EventCount metrics**  
Track the number of events polled, filtered, invoked, and dropped by your event source mapping. CloudWatch charges apply.

**Batch size** [Info](#)  
The maximum number of records in each batch to send to the function.

**Starting position** [Info](#)  
The position in the stream to start reading from.

**Batch window - optional**  
The maximum amount of time to gather records before invoking the function, in seconds.

**► Additional settings**

In order to read from the DynamoDB trigger, your execution role must have proper permissions.

[Cancel](#) [Add](#)

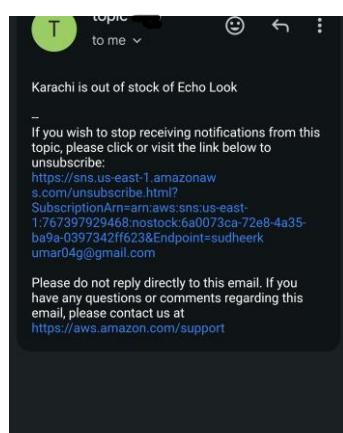
## STEP 9: upload files

Go to S3 bucket and upload the csv files.

## STEP 10: Notification

Go and check the email.

Email will report should be received.



## AWS Elastic Beanstalk

**Elastic Beanstalk** is a fully managed service provided by AWS that makes it easy to deploy and manage applications in the cloud without worrying about the underlying infrastructure. It is a **Platform as a Service (PaaS)** that simplifies deploying, managing, and scaling web applications and services.

## How it works

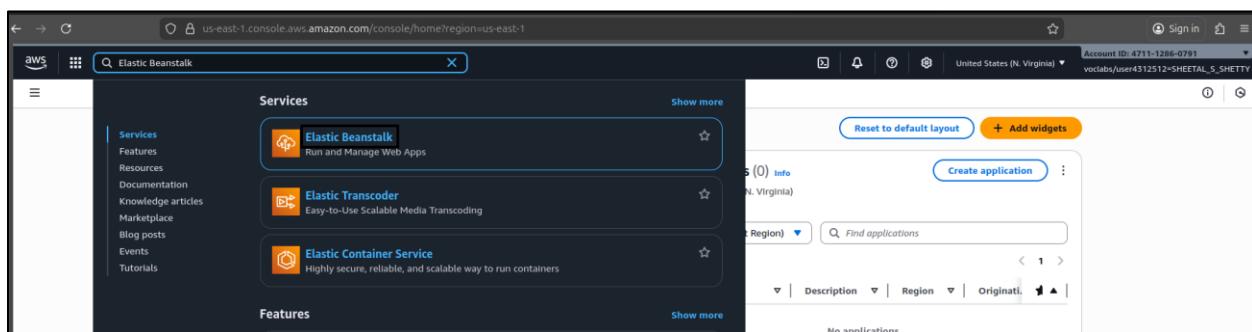
1. You upload your application code (Java, .NET, Node.js, Python, etc.).
2. Elastic Beanstalk automatically handles:
  - a. Provisioning and configuring servers (EC2 instances)
  - b. Setting up load balancers and Auto Scaling groups
  - c. Managing the application environment
  - d. Monitoring, logging, and health checks

## Who is it for

- Developers who want to focus on writing code rather than managing infrastructure
- Small to medium-sized applications that need quick and easy deployment
- Teams that want a managed environment but still need some level of control when required

Lab:

Step1: In the console, in the search box, search and choose **Elastic Beanstalk**.



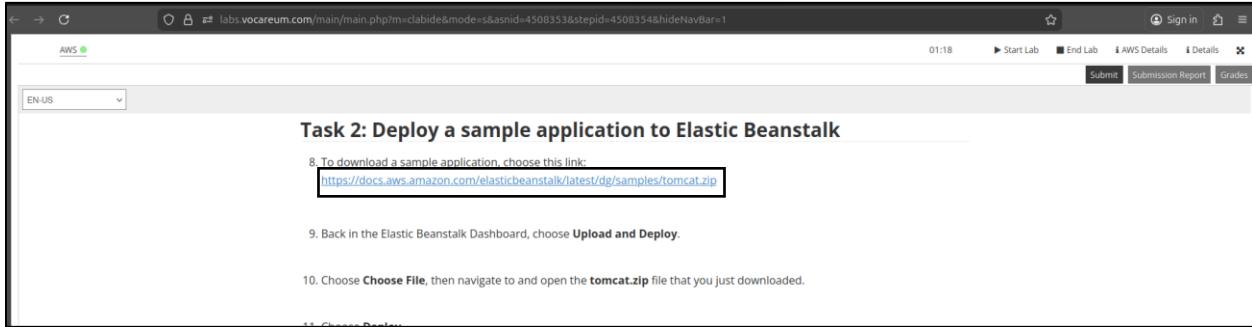
Under the **Environment**, click on the name of the environment. This opens the Dashboard page for the Elastic Beanstalk environment.

Go to EC2-> Click Load Balancer from the left navigation menu and access the Domain Name.

When you open the domain link in browser, AWS launches the EC2 instance with an application server. However, since no application code has been deployed to the Beanstalk environment yet, the page will not display any content. Instead, it will show an **HTTP Status 404 – Not Found** message. This is expected behavior until the application is deployed.

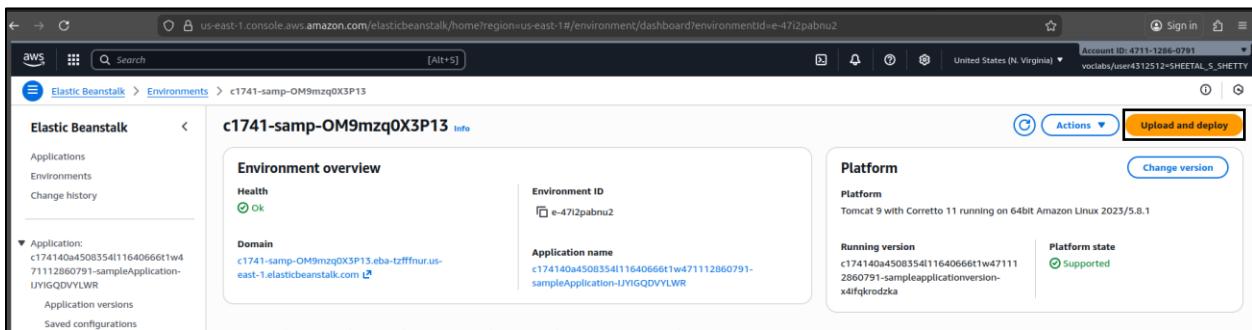
Step2: Deploying a sample application to Elastic Beanstalk.

Go to the AWS current lab instruction page. There, you will find a link to download the sample **tomcat.zip** file. Click on the link, and the **tomcat.zip** file will be downloaded to your system.

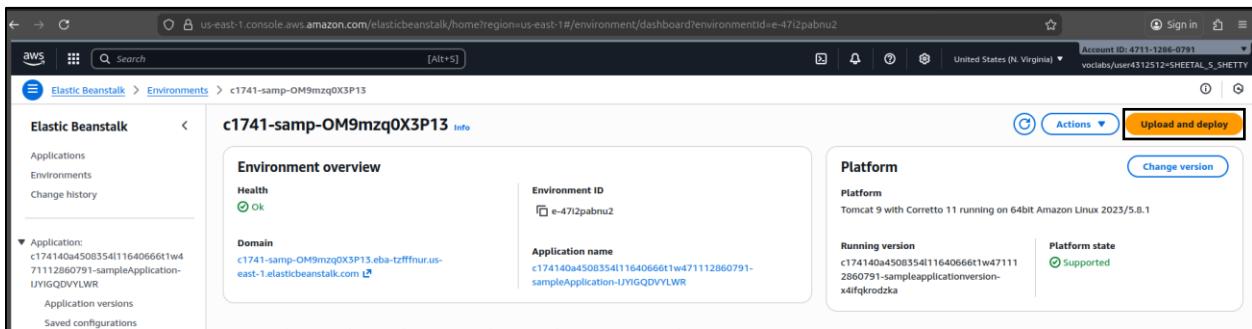


The screenshot shows a browser window with the URL <https://labs.vocareum.com/main/main.php?m=clabide&mode=s&asnid=4508353&stepid=4508354&hideNavBar=1>. The page title is "Task 2: Deploy a sample application to Elastic Beanstalk". Step 8 instructs to "To download a sample application, choose this link: <https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/tomcat.zip>". Step 9 says "Back in the Elastic Beanstalk Dashboard, choose **Upload and Deploy**". Step 10 says "Choose File, then navigate to and open the **tomcat.zip** file that you just downloaded." Step 11 says "Choose Deploy".

After downloading, return to the Elastic Beanstalk dashboard and continue with the **Upload and Deploy** steps.

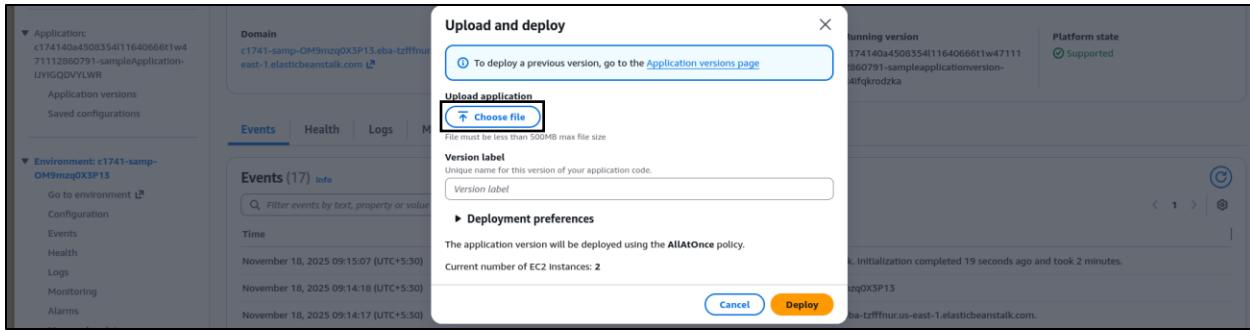


The screenshot shows the AWS Elastic Beanstalk console at <https://us-east-1.console.aws.amazon.com/elasticbeanstalk/home?region=us-east-1#/environment/dashboard?environmentId=e-4712pabnu2>. The environment name is "c1741-samp-OM9mzq0X3P13". The "Actions" button is highlighted in yellow. Other visible details include the Environment ID "e-4712pabnu2", Application name "c174140a4508354l11640666t1w471112860791-sampleapplication-UYIGQDVYLWR", and Platform "Tomcat 9 with Corretto 11 running on 64bit Amazon Linux 2023/5.8.1".

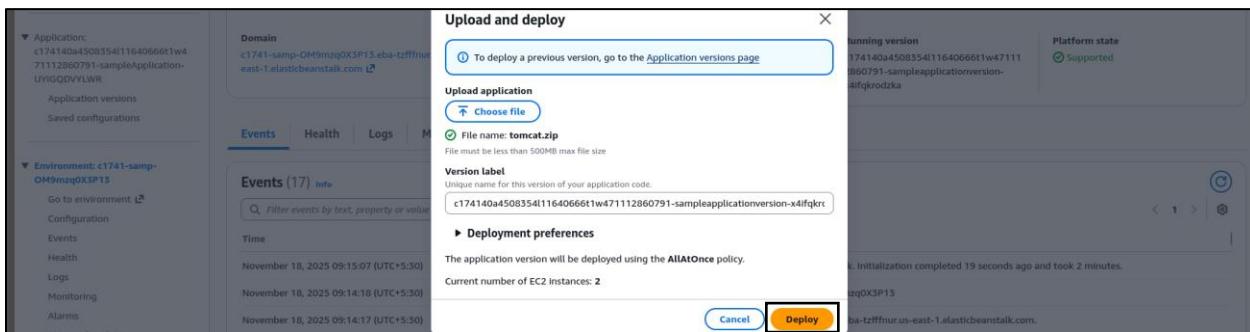


The screenshot shows the same AWS Elastic Beanstalk environment overview page as the previous one, but the "Upload and deploy" button is highlighted in yellow. The rest of the interface remains the same, showing the environment details and the "Actions" button.

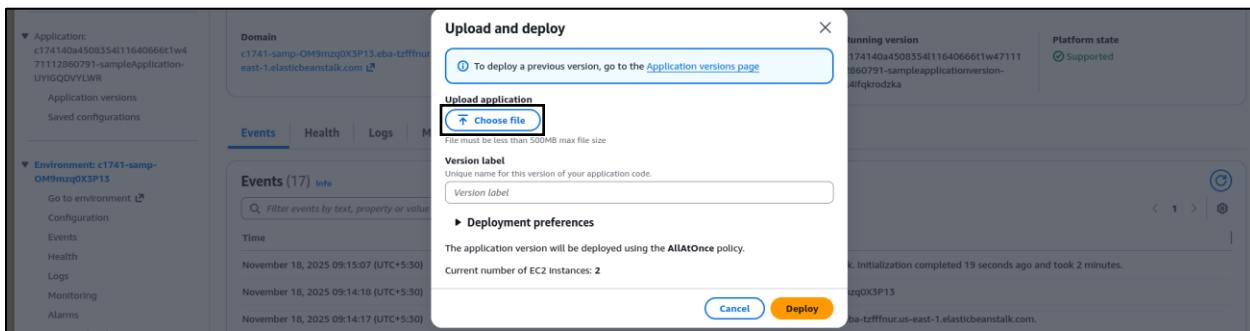
Select **Choose File**, then navigate to the **tomcat.zip** file you downloaded and open it.



Next, choose **Deploy**.



Elastic Beanstalk will take a minute or two to update your environment and deploy the application.



**Step3:** Once the deployment is complete, go to EC2-> Click **Load Balancer** from the left navigation menu and access the **Domain Name**.

**Load balancers (1/1) What's new?**

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones	Security groups
awseb-e-4-AWSEBLoa-1...	-	classic	-	-	vpc-0cc3737f79056cdea	3 Availability Zones	sg-0f80544d2e50db4b...

**Load balancer: awseb-e-4-AWSEBLoa-168FSLHRKYV7**

DNS name info: awseb-e-4-AWSEBLoa-168FSLHRKYV7-1100723373.us-east-1.elb.amazonaws.com (A Record)

This Classic Load Balancer can be migrated to a next generation load balancer. Migration wizard uses your load balancer's current configurations to create a new load balancer. [Learn more](#) [Launch migration wizard](#)

Distribution of targets by Availability Zone (AZ)

For each enabled Availability Zone, you can view the number of registered instances and their current health states. Selecting any values here will apply the corresponding filter to the Target instances table.

The deployed **web application** will now be displayed.

Not Secure | http://awseb-e-4-awsebbla-168fslhrkyv7-1100723373.us-east-1.elb.amazonaws.com

# Congratulations

Your first AWS Elastic Beanstalk Application is now running on your own dedicated environment in the AWS Cloud

**What's Next?**

- [Learn how to build, deploy and manage your own applications using AWS Elastic Beanstalk](#)
- [AWS Elastic Beanstalk concepts](#)
- [Learn how to create new application versions](#)
- [Learn how to manage your application environments](#)

**Download the AWS Reference Application**

- [Explore a fully-featured reference application using the AWS SDK for Java](#)

**AWS Toolkit for Eclipse**

- [Developers may build and deploy AWS Elastic Beanstalk applications directly from Eclipse](#)
- [Get started with Eclipse and AWS Elastic Beanstalk by watching this video](#)
- [View all AWS Elastic Beanstalk documentation](#)

# Amazon Elastic File System (Amazon EFS) Report

**Lab:** Introducing Amazon Elastic File System (Amazon EFS)

## Objectives

- ⑩ Access the AWS console, create an EFS file system, and launch an Amazon Linux EC2 instance.
- ⑩ Connect to the EC2 instance and mount the EFS file system.
- ⑩ Review and monitor the file system's performance.

## Lab Environment Setup

- ⑩ This lab introduces you to Amazon Elastic File System (Amazon EFS) by using the AWS Management Console.
- ⑩ A timer-based lab session was started using the “Start Lab” button.
- ⑩ Pop-up windows were allowed in the browser to open the AWS Management Console in a new tab.
- ⑩ Resources were named exactly as specified in the instructions to ensure the lab scoring script works properly.

The screenshot shows a web browser window for the AWS Academy platform. The URL is [awsacademy.instructure.com/courses/131613/assignments/1511958/module\\_item\\_id=12598718](https://awsacademy.instructure.com/courses/131613/assignments/1511958/module_item_id=12598718). The page title is "ACAv3EN-US-LT13-131613 > Assignments > Guided lab: Introducing Amazon Elastic File System (Amazon EFS) > Guided lab: Introducing Amazon Elastic File System (Amazon EFS)".

The main content area displays the assignment details:

- Due: No Due Date
- Points: 15
- Status: Submitting an external tool

A progress bar indicates "AWS 0%".

On the right side, there is a "Submission" section with the following information:

- Date: Oct 28 at 1:41pm
- Grade: 5 (15 pts possible)
- Graded Anonymously: no
- Comments: No Comments

Below the submission section, there is a "Guided Lab: Introducing Amazon Elastic File System (Amazon EFS)" section with the following content:

### Guided Lab: Introducing Amazon Elastic File System (Amazon EFS)

#### Lab overview and objectives

This lab introduces you to Amazon Elastic File System (Amazon EFS) by using the AWS Management Console.

After completing this lab, you should be able to:

- Log in to the AWS Management Console
- Create an Amazon EFS file system
- Log in to an Amazon Elastic Compute Cloud (Amazon EC2) instance that runs Amazon Linux
- Mount your file system to your EC2 instance
- Examine and monitor the performance of your file system

At the bottom of the page are navigation links: "Previous" and "Next".

# Task 1: Creating a security group to access your EFS file system

## Steps:

1. At the top of the AWS Management Console, in the search box, search for and choose EC2.
2. In the navigation pane on the left, choose **Security Groups**.
3. Copy the **Security group ID** of the *EFSClient* security group to your text editor. The Group ID should look similar to *sg-03727965651b6659b*.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation pane with sections like Dashboard, EC2 Global View, Events, Instances (with sub-options like Instances, Instance Types, Launch Templates, etc.), Images, Elastic Block Store, and Network & Security. The main area shows a table of instances. A single row is selected for an instance named "EFS Client" with the ID "i-09dfd0f7bcbba0af3". Below the table, a modal window is open for this specific instance. The modal has tabs for Details, Status and alarms, Monitoring, Security (which is selected), Networking, and Storage. Under the Security tab, there's a section for "Security details" which includes an IAM Role (c174142a4508368l12159964t1w53851797-Ec2InstanceRole-) and a "Security group ID copied" message with a tooltip pointing to the ID "sg-0a9e53857194be273 (EFSClient)". To the right of the modal, there's an "Owner ID" field showing "538517972279".

4. Choose **Create security group** then configure:

- ⑩ **Security group name:** EFS Mount Target
- ⑩ **Description:** Inbound NFS access from EFS clients
- ⑩ **VPC:** Lab VPC

5. Under the **Inbound rules** section, choose **Add rule** then configure:

⑩ Type: NFS

⑩ Source:

⑩ Custom

⑩ In the *Custom* box, paste the security group's **Security group ID** that you copied to your text editor

⑩ Choose **Create security group**.

Security Groups (3) <small>Info</small>					
<small>Find security groups by attribute or tag</small>					
	Name	Security group ID	Security group name	VPC ID	Description
	EFSClient	<a href="#">sg-0a9e53857194be273</a>	EFSClient	<a href="#">vpc-096da48e30394258d</a>	EFS Client
	-	<a href="#">sg-0cb4204eabb6fc7</a>	default	<a href="#">vpc-075b774dec577f345</a>	default VPC security group
	-	<a href="#">sg-0c3377404fb85b35b</a>	default	<a href="#">vpc-096da48e30394258d</a>	default VPC security group

EC2 > Security Groups > Create security group

### Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

**Basic details**

**Security group name** Info  
EFS Mount Target  
Name cannot be edited after creation.

**Description** Info  
EFS Mount Targe

**VPC** Info  
vpc-096da48e30394258d (Lab VPC)

**Inbound rules** Info

Type	Protocol	Port range	Source	Description - optional
NFS	TCP	2049	Custom	<input type="text" value="sg-0a9e53857194be273"/> <small>X</small> <small>Delete</small>
<small>Add rule</small>				

**Outbound rules** Info

Type	Protocol	Port range	Destination	Description - optional
All traffic	All	All	Custom	<input type="text" value="0.0.0.0/0"/> <small>X</small> <small>Delete</small>
<small>Add rule</small>				

## Task 2: Creating an EFS file system

## Steps:

1. At the top of the AWS Management Console, in the search box, search for and choose EFS.

2. Choose **Create file system**.

3. In the **Create file system** window, choose **Customize**.

4. On **Step 1**:

⑩ Uncheck Enable Automatic backups.

⑩ **Lifecycle management**:

⑩ for **Transition into IA** Select *None*.

⑩ In the **Tags optional** section, configure:

⑩ **Key:** Name

⑩ **Value:** myefs

5. Choose **Next**.

6. For **VPC**, select *Lab VPC*.

7. Detach the default security group from each *Availability Zone* mount target by choosing the check box on each default security group.

8. Attach the **EFS Mount Target** security group to each *Availability Zone* mount target by choosing **EFS Mount Target** for each Availability Zone.

9. Choose **Next**.

10. On **Step 3**, choose Next.

11. On **Step 4**:

Review your configuration.

Choose Create.

- Step 2  
Network access
- Step 3 - optional  
File system policy
- Step 4  
Review and create

### General

**Name - optional**  
Name your file system.

Name can include letters, numbers, and +-.~/ symbols, up to 256 characters.

**File system type**  
Choose to either store data across multiple Availability Zones or within a single Availability Zone. [Learn more](#)

**Regional**  
Offers the highest levels of availability and durability by storing file system data across multiple Availability Zones within an AWS Region.

**One Zone**  
Provides continuous availability to data within a single Availability Zone within an AWS Region.

**Automatic backups**  
Automatically backup your file system data with AWS Backup using recommended settings. Additional pricing applies. [Learn more](#)

Enable automatic backups

⚠ We recommend that you create a backup policy for your file system

**Lifecycle management**  
Automatically save money as access patterns change by moving files into the Infrequent Access (IA) or Archive storage class. [Learn more](#)

Transition into Infrequent Access (IA)	Transition into Archive	Transition into Standard
Transition files to IA based on the time since they were last accessed in Standard storage.	Transition files to Archive based on the time since they were last accessed in Standard storage.	Transition files back to Standard storage based on when they are first accessed in IA or Archive storage.
<input style="border: 1px solid #ccc; padding: 2px; width: 100%;" type="button" value="30 day(s) since last access"/>	<input style="border: 1px solid #ccc; padding: 2px; width: 100%;" type="button" value="90 day(s) since last access"/>	<input style="border: 1px solid #ccc; padding: 2px; width: 100%;" type="button" value="None"/>

**Encryption**  
Choose to enable encryption of your file system's data at rest. Uses the AWS KMS service key (aws/elasticfilesystem) by default. [Learn more](#)

Enable encryption of data at rest

Enable encryption of data at rest

### Performance settings

**Throughput mode**  
Choose a method for your file system's throughput limits. [Learn more](#)

**Enhanced**  
Provides more flexibility and higher throughput levels for workloads with a range of performance requirements.

**Bursting**  
Provides throughput that scales with the amount of storage for workloads with basic performance requirements.

**Elastic (Recommended)**  
Use this mode for workloads with unpredictable I/O. With Elastic Throughput, performance automatically scales with your workload activity and you only pay for the throughput you use (data transferred for your file systems per month). [Learn more](#)

**Provisioned**  
Use this mode if you can estimate your workload's throughput requirements. With Provisioned mode, you configure your file system's throughput and pay for throughput provisioned.

▶ **Additional settings**

▼ **Tags optional**

Add tags to associate key-value pairs to your resource. [Learn more](#)

<b>Tag key</b>	<input type="text" value="Name"/>	<small>Use: "myefs"</small>
<input type="text" value="Q_"/>	<input type="text" value="myefs"/>	<small>X Remove tag</small>

Add tag

You can add 49 more tag(s)

Next
Cancel

Amazon EFS > File systems > Create

**Step 1** File system settings  
**Step 2** Network access  
 Step 3 - optional  
 File system policy  
 Step 4  
 Review and create

### Network access

**Network**  
Virtual Private Cloud (VPC) | Learn more

Choose the VPC where you want EC2 instances to connect to your file system.

vpc-096da48e30394258d  
Lab VPC

**Mount targets**  
A mount target provides an NFSv4 endpoint at which you can mount an Amazon EFS file system. We recommend creating one mount target per Availability Zone. Learn more

Availability zone	Subnet ID	IP address type	IPv4 address	IPv6 address	Security groups
us-east-1a	subnet-015...	IPv4 only	Optional	-	<input type="button" value="Choose secur..."/> <input type="button" value="Remove"/> <span style="border: 1px solid #0072bc; padding: 2px;">sg-0b3c3b0df</span> X 0110b9b0 EFS Mount Target
us-east-1b	subnet-00ff...	IPv4 only	Optional	-	<input type="button" value="Choose secur..."/> <input type="button" value="Remove"/> <span style="border: 1px solid #0072bc; padding: 2px;">sg-0b3c3b0df</span> X 0110b9b0 EFS Mount Target

Amazon EFS > File systems

**Elastic File System**

**Success**  
File system (fs-0570cbaecd7532b4f) is available.

**File systems (1)**

Name	File system ID	Encrypted	Total size	Size in Standard	Size in IA	Size in Archive	Provisioned Throughput (MiB/s)	File system state	Creation time	Availability Zone
myefs	fs-0570cbaecd7532b4f	Unencrypted	6.00 KIB	6.00 KIB	0 Bytes	0 Bytes	-	Available	Tue, 18 Nov 2025 11:06:49 GMT	Regional

## Task 3: Connecting to your EC2 instance

1. To connect to the **EC2 instance**, from the top of this page, choose **i AWS Details** and copy the value for *InstanceSessionURL*.
2. Paste it into the new browser tab or window to connect to the EC2 instance using AWS Systems Manager Session Manager.

You should now be connected to the instance.

The screenshot shows the AWS Cloud Access interface. At the top, there are buttons for 'Start Lab', 'End Lab', 'AWS Details', 'Details', 'Submit', 'Submission Report', and 'Grades'. A 'Close' button is also present. The main area is titled 'Cloud Access' and contains the following information:

- AWS CLI:** A 'Show' button.
- Cloud Labs:** Remaining session time: 01:50:33(111 minutes), Session started at: 2025-11-18T06:07:59-0800, Session to end at: 2025-11-18T08:07:59-0800.
- Accumulated lab time: 05:24:00 (324 minutes).
- IPs -- public:52.1.203.163, private:10.0.1.118
- SSH key:** Buttons for 'Show', 'Download PEM', and 'Download PPK'.
- AWS SSO:** A 'Download URL' button.
- InstanceSessionURL:** A redacted URL starting with <https://us-east-1.console.aws.amazon.com/>.

On the left side of the interface, there is a vertical sidebar with the following text:

- ht of the command prompt
- efs.
- r and choose EFS.
- Attach** to open the Amazon
- sing the NFS client section.
- =600,retrans=2,noresvport fs-

## Task 4: Creating a new directory and mounting the EFS file system

1. In your EC2 terminal session, run the following command to install the required utilities:

```
sudo su -l ec2-user  
sudo yum install -y amazon-efs-utils
```

2. Run the following command to create directory for mount: `sudo mkdir efs`.

3. At the top of the AWS Management Console, in the search box, search for and choose EFS.

4. Choose **myefs**.

5. In the **Amazon EFS Console**, on the top right corner of the page, choose **Attach** to open the Amazon EC2 mount instructions.

6. In your EC2 terminal session, Copy and run the entire command in the **Using the NFS client section**.

```
sudo mount -t nfs4 -o  
nfsvers=4.1,rsize=1048576,wszie=1048576,hard,timeo=600,retrans=2,noresvport fs-  
bce57914.efs.us-west-2.amazonaws.com:/ efs
```

7. Get a full summary of the available and used disk space usage by entering:

```
sudo df -hT
```

The screenshot shows a terminal session in a CloudShell window. The session ID is user4312501-SNEHA\_S\_SHETTY-4z2gxap4747vbvtvaygbileuu. The instance ID is i-09dfdf0f7bcbba0af3. The terminal window has tabs for 'Shortcuts' and 'Terminate'. The command run was 'sudo yum install -y amazon-efs-utils'. The output shows the transaction summary for installing 'amazon-efs-utils' and 'stunnel'. It includes details like package name, architecture, version, repository, and size. The transaction summary also shows the total download size, installed size, and the progress of the download.

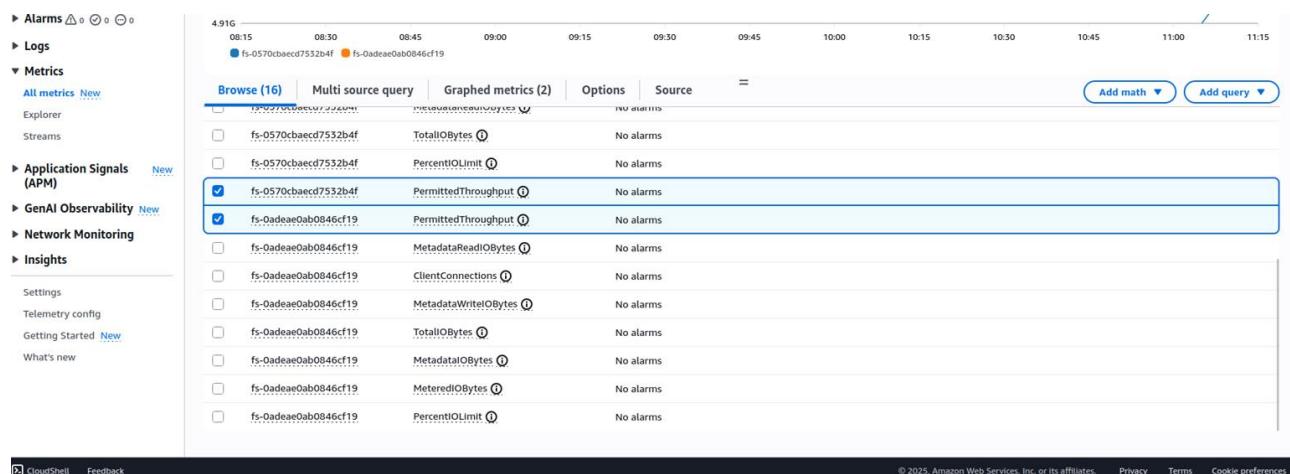
```
Session ID: user4312501-SNEHA_S_SHETTY-4z2gxap4747vbvtvaygbileuu [ Shortcuts ] Instance ID: i-09dfdf0f7bcbba0af3 [ Terminate ]  
  
sh-5.2$ sudo su -l ec2-user  
[ec2-user@ip-10-0-1-52 ~]$ sudo yum install -y amazon-efs-utils  
Last metadata expiration check: 0:28:41 ago on Tue Nov 18 10:47:13 2025.  
Dependencies resolved.  
=====  
Package           Architecture      Version       Repository      Size  
=====  
Installing:  
amazon-efs-utils          x86_64        2.4.0-1.amzn2023      amazonlinux    4.7 M  
Installing dependencies:  
stunnel                  x86_64        5.58-1.amzn2023.0.2  amazonlinux    156 k  
  
Transaction Summary  
=====  
Install 2 Packages  
  
Total download size: 4.9 M  
Installed size: 9.9 M  
Downloading Packages:  
(1/2): stunnel-5.58-1.amzn2023.0.2.x86_64.rpm          4.1 MB/s | 156 kB     00:00  
(2/2): amazon-efs-utils-2.4.0-1.amzn2023.x86_64.rpm      46 MB/s | 4.7 MB     00:00  
=====  
36 MB/s | 4.9 MB     00:00  
  
Total  
Running transaction check  
Transaction check succeeded.  
Running transaction test  
Transaction test succeeded.  
Running transaction  
  Preparing :  
  Installing : stunnel-5.58-1.amzn2023.0.2.x86_64          1/1  
  Running scriptlet: stunnel-5.58-1.amzn2023.0.2.x86_64      1/2  
  Installing : amazon-efs-utils-2.4.0-1.amzn2023.x86_64      1/2  
  Running scriptlet: amazon-efs-utils-2.4.0-1.amzn2023.x86_64  2/2  
  Verifying   : amazon-efs-utils-2.4.0-1.amzn2023.x86_64      2/2  
  Verifying   : stunnel-5.58-1.amzn2023.0.2.x86_64          2/2  
  
Installed:  
amazon-efs-utils-2.4.0-1.amzn2023.x86_64                      stunnel-5.58-1.amzn2023.0.2.x86_64  
  
Complete!  
[ec2-user@ip-10-0-1-52 ~]$
```

```
[ec2-user@ip-10-0-1-52 ~]$ sudo mkdir efs  
[ec2-user@ip-10-0-1-52 ~]$ sudo mount -t efs -o tls fs-0570cbaecd7532b4f:/ efs  
[ec2-user@ip-10-0-1-52 ~]$ sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wszie=1048576,hard,timeo=600,retrans=2,noresvport fs-0570cbaecd7532b4f.efs.us-east-1.amazonaws.com:/ efs  
[ec2-user@ip-10-0-1-52 ~]$
```

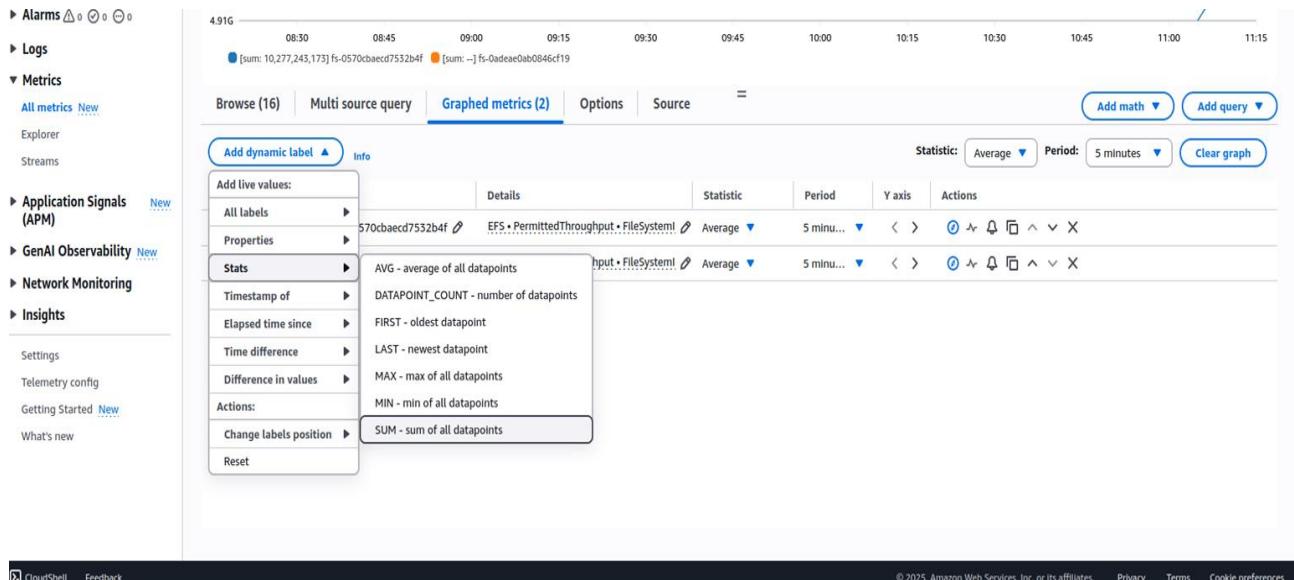
## Task 5: Examining the performance behavior of your new EFS file system

## Monitoring performance by using Amazon CloudWatch

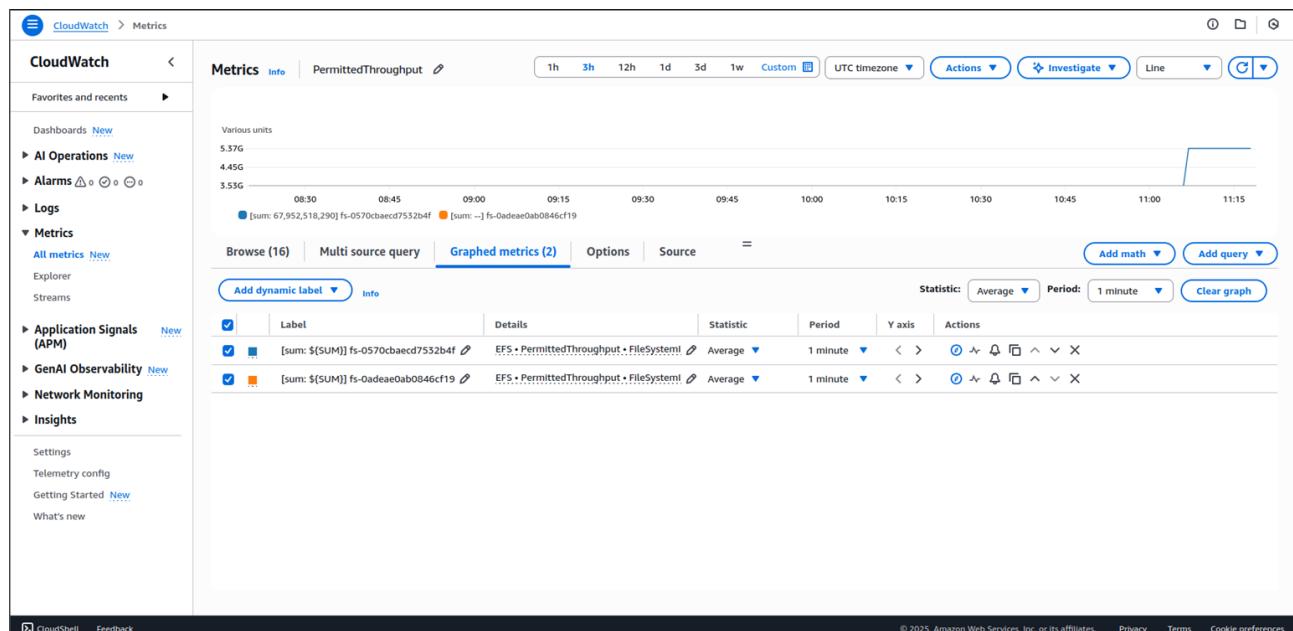
1. At the top of the AWS Management Console, in the search box, search for and choose **CloudWatch**.
2. In the navigation pane on the left, choose **All Metrics**.
3. In the **All metrics** tab, choose **EFS**.
4. Choose **File System Metrics**.
5. Select all the options that has the **PermittedThroughput** Metric Name.



6. Choose the **Graphed metrics** tab.
7. On the **Statistics** column, select **Sum**.
8. On the **Period** column, select **1 Minute**.



9. Note the the peak value, which is around 7.6G. Take this number (in bytes) and divide it by the duration in seconds (60 seconds). The result gives you the write throughput (B/s) of your file system during your test.



# Elastic ( user data) Introduction to EC2

Cloud foundations Module 6 Lab3

## Task 1: Launch Your Amazon EC2 Instance

Step 1: Name and tags - Web Server

Step 2: Application and OS Images (Amazon Machine Image) -Amazon Linux AMI

Step 3: Instance type - t2.micro

Step 4: Key pair (login) - vockey

Step 5: Network settings - Lab VPC - PublicSubnet1

Firewall (security groups) - Create security group

Security group name: Web Server security group

Description: Security group for my web server

Inbound security group rules- remove existing rules.

Step 6: Configure storage

Step 7: Advanced details

- Termination protection, select Enable

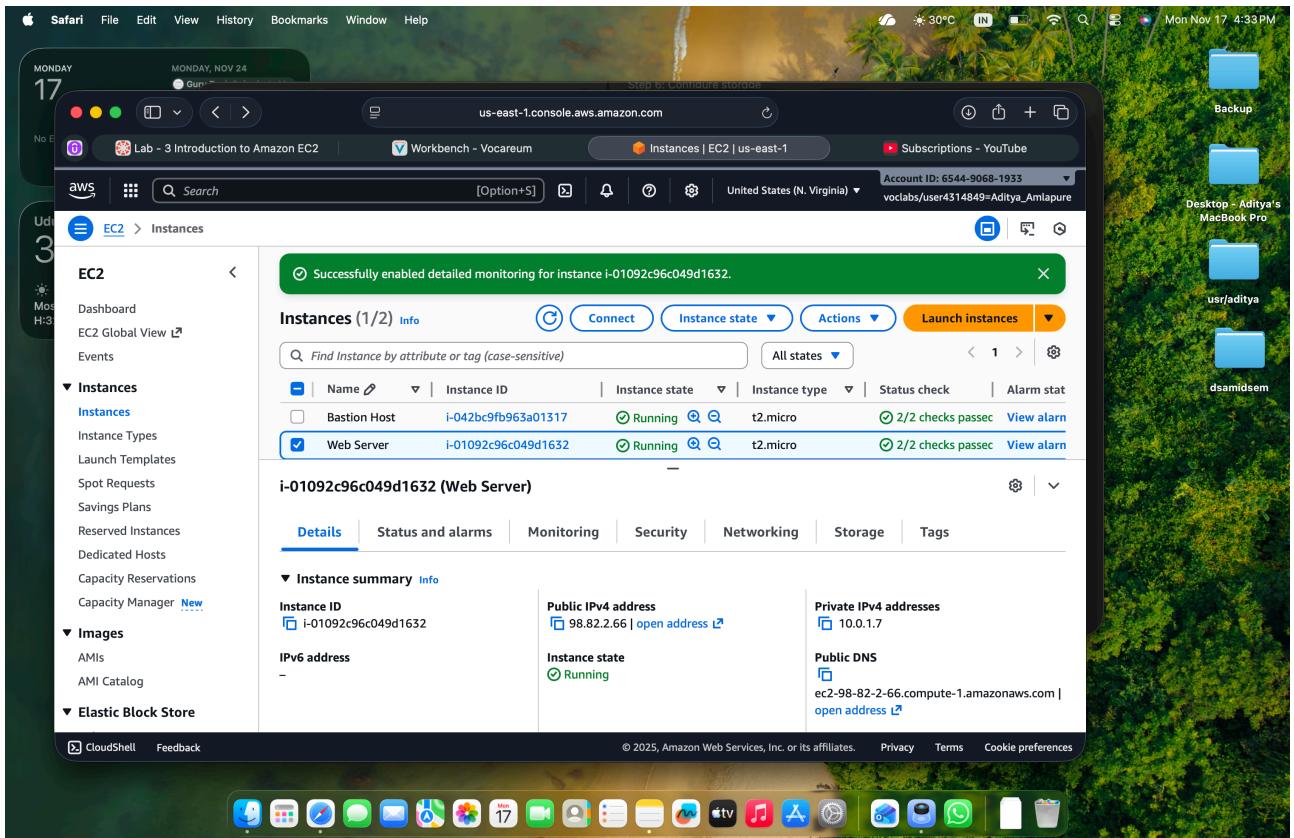
- user data box

```
#!/bin/bash
dnf install -y httpd
systemctl enable httpd
systemctl start httpd
echo '<html><h1>Hello From Your Web Server!</h1></html>' > /var/www/html/index.html
```

The script will:

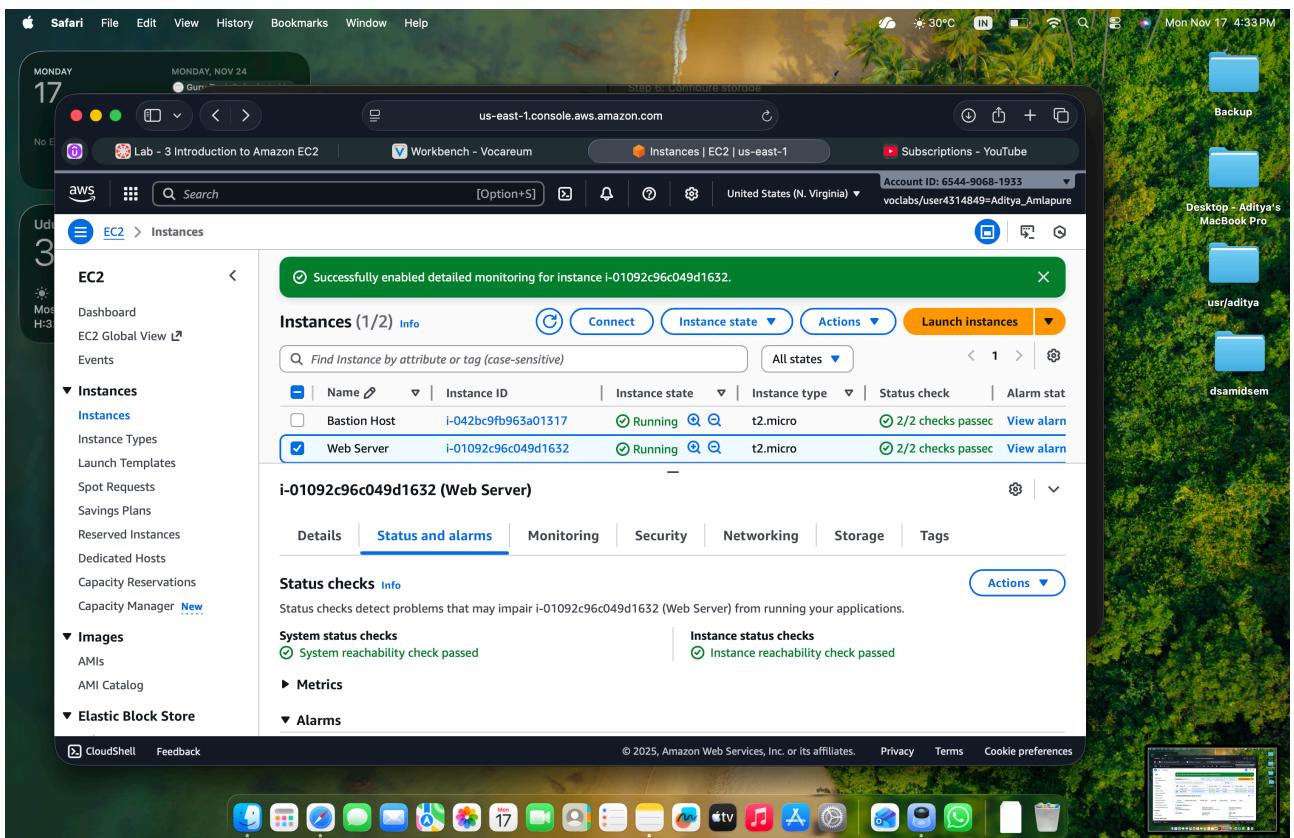
- Install an Apache web server (httpd)
- Configure the web server to automatically start on boot
- Run the Web server once it has finished installing
- Create a simple web page

Step 8: Launch the instance



## Task 2: Monitoring Your Instance

Status checks tab  
System reachability and Instance reachability



## Monitor and troubleshoot Get system log.

The screenshot shows the AWS CloudWatch Logs interface for an EC2 instance. The top navigation bar includes tabs for 'CloudTrail events', 'SSM command history', 'Reachability Analyzer - new', 'Instance events', 'Instance screenshot', and 'System log'. The 'System log' tab is selected. A large text area displays the system log output, which includes logs from cloud-init and SSH key fingerprints. The log ends with the message: 'Cloud-init v. 22.2.2 finished at Mon, 17 Nov 2025 10:56:32 +0000. Datasource DataSourceEc2. Up 31.33 seconds'.

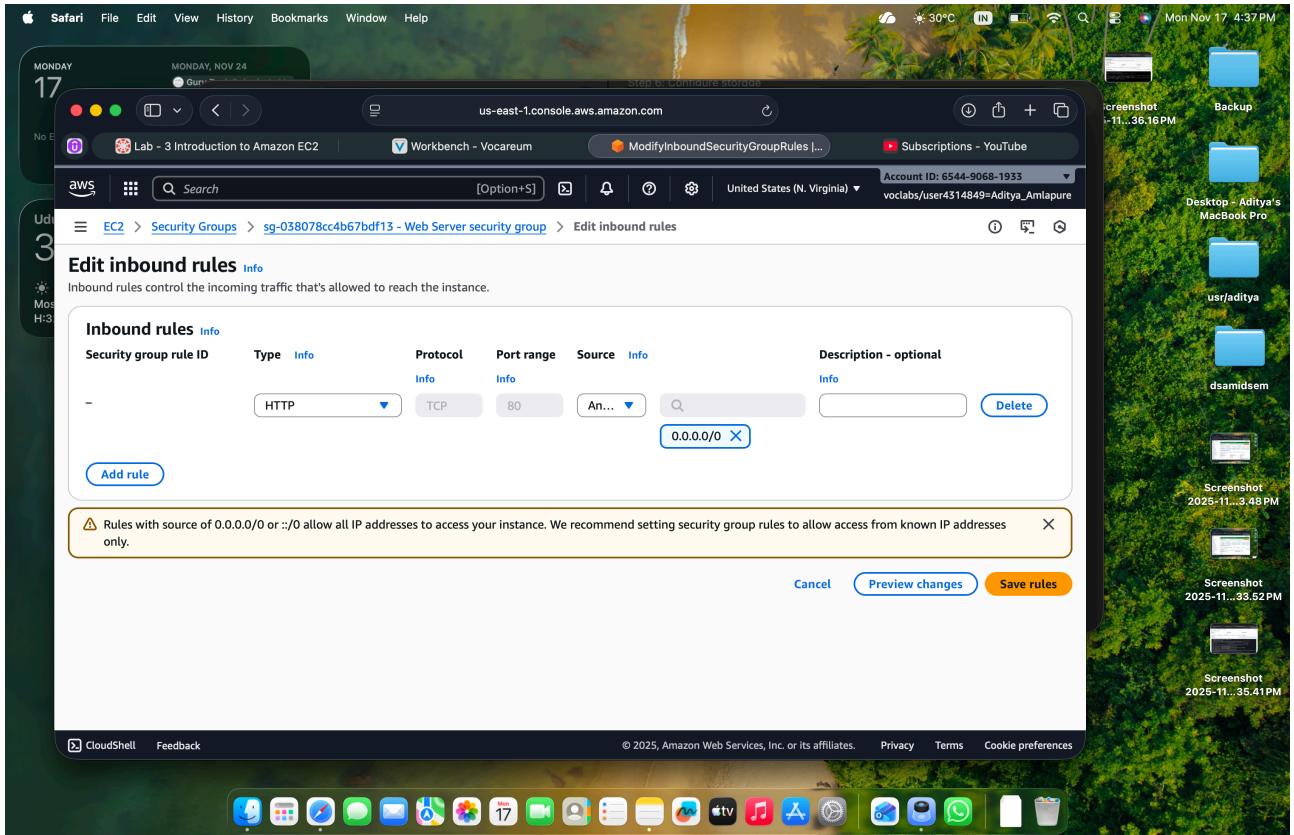
## Monitor and troubleshoot Get instance screenshot.

The screenshot shows the AWS Instance Diagnostics interface for an EC2 instance. The top navigation bar includes tabs for 'CloudTrail events', 'SSM command history', 'Reachability Analyzer - new', 'Instance events', 'Instance screenshot', and 'System log'. The 'Instance screenshot' tab is selected. A large text area displays the instance screenshot, which is a terminal session showing the boot process of an Amazon Linux 2023.9.20251110 instance. The terminal output includes kernel version information and messages related to memory limits and zram generator configuration.

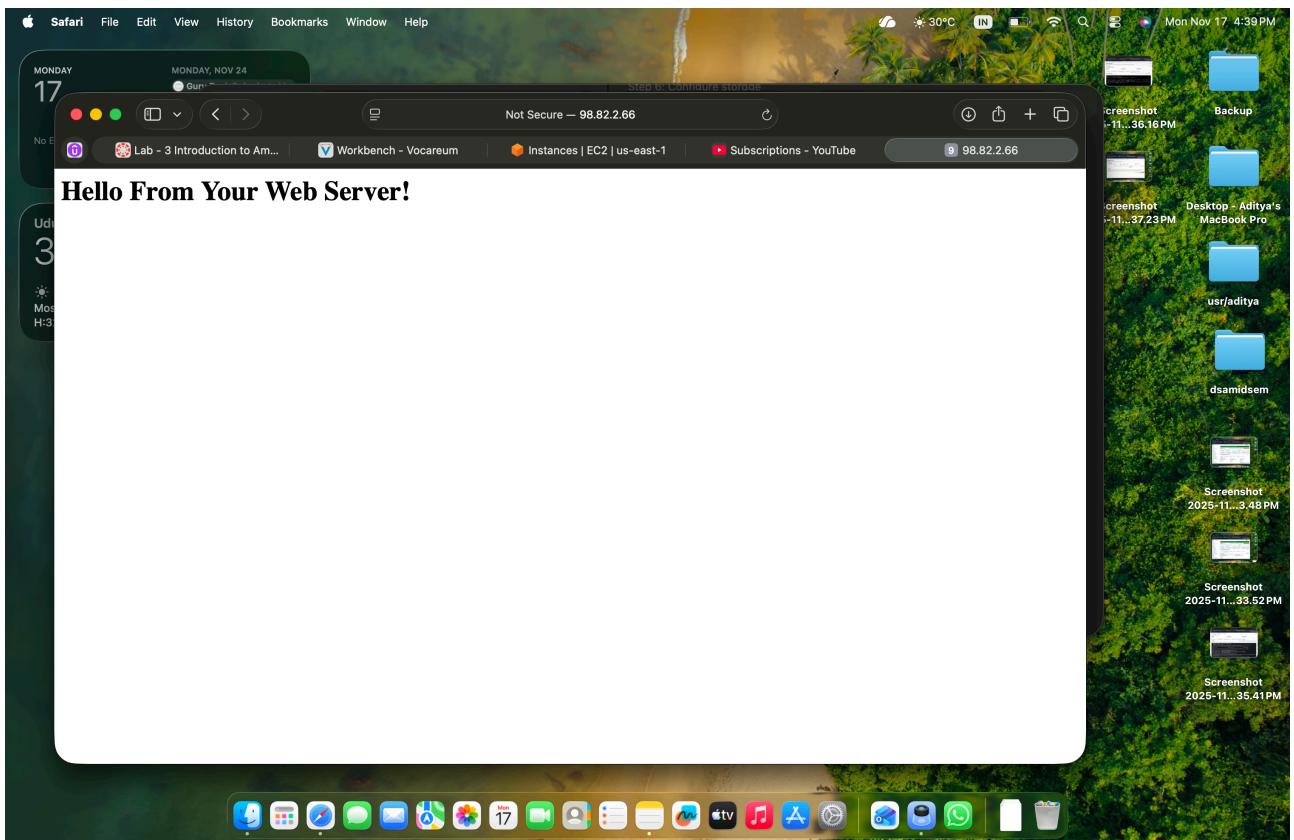
## Task 3: Update Your Security Group and Access the Web Server

## Web Server security group - Inbound rules - edit

- Type: HTTP
- Source: Anywhere-IPv4



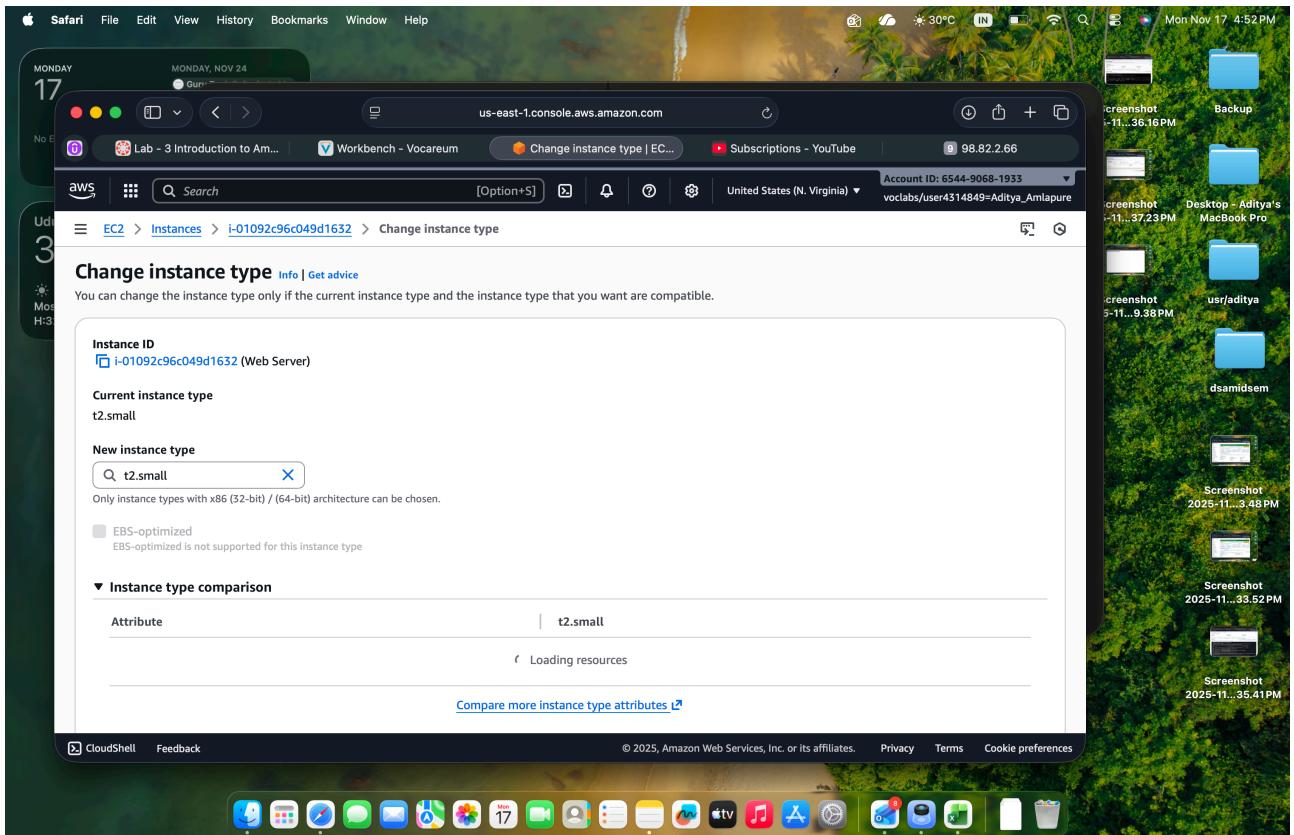
Return to the web server tab that you previously opened and refresh the page.  
You should see the message Hello From Your Web Server!

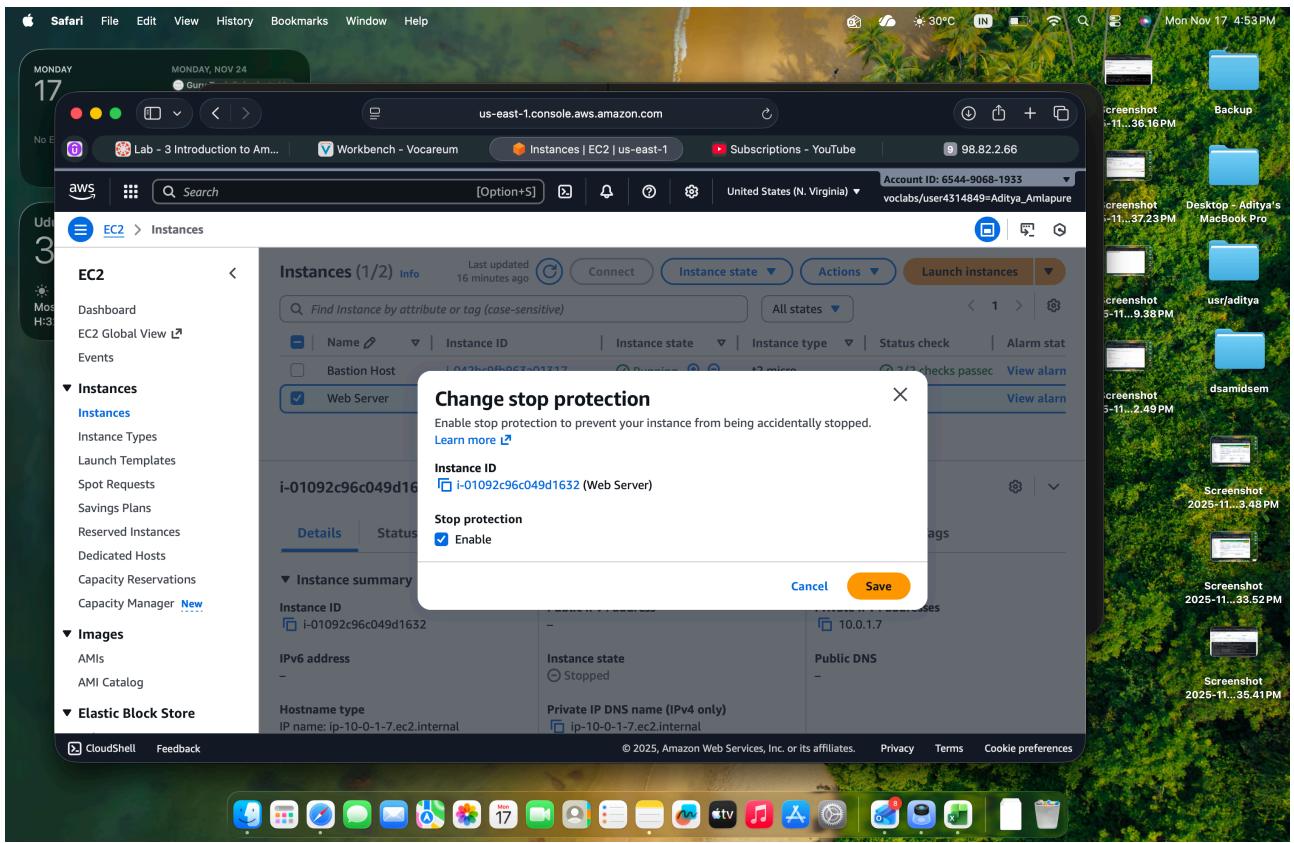


## Task 4: Resize Your Instance: Instance Type and EBS Volume

### Stop Your Instance

Change The Instance Type ( t2.small) and enable stop protection



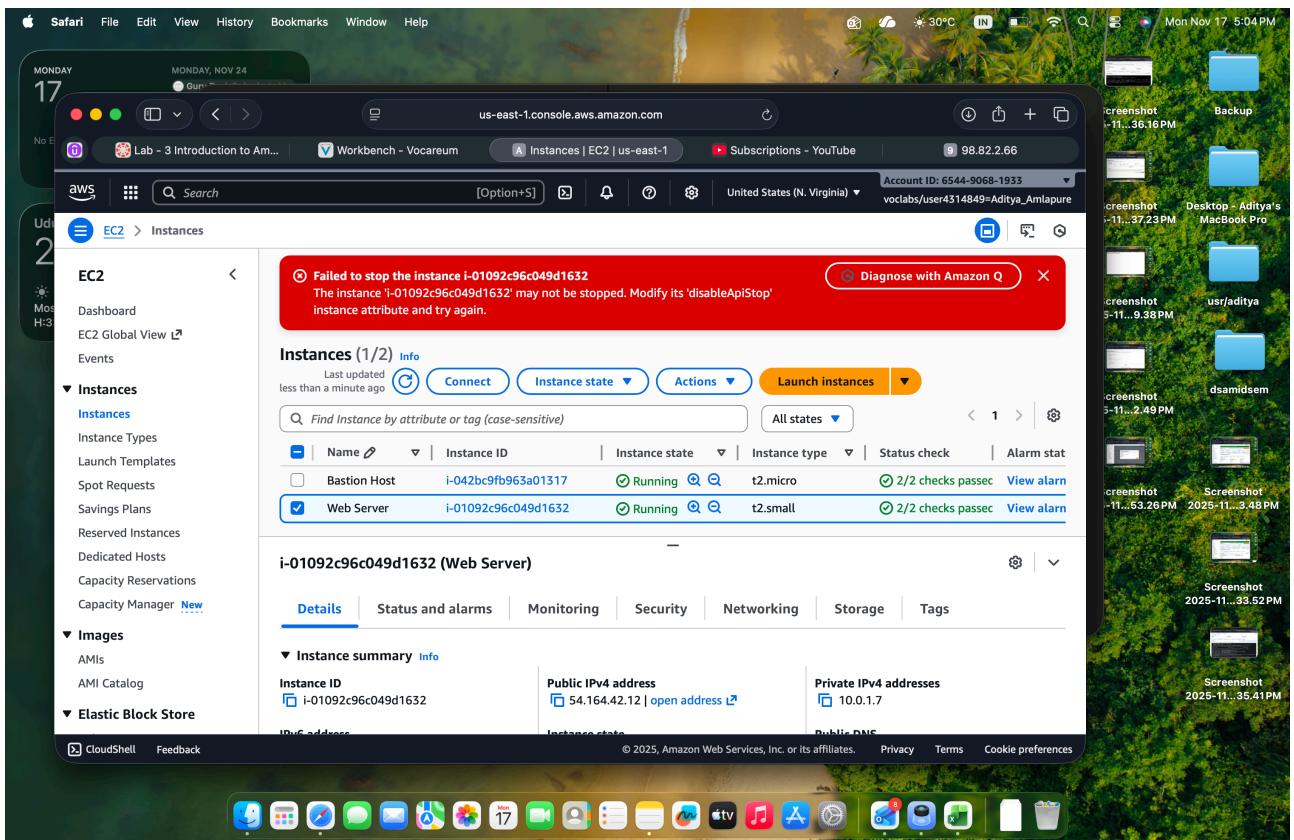


Resize the EBS Volume. 10

Start the stopped instance

Task 5: Explore EC2 Limits  
Service Quotas

Task 6: Test Stop Protection



# Creating a Scalable and Highly Available Environment for the Café

## TASK 1 — Update Network for Multi-AZ High Availability

### Task 1.1 — Create NAT Gateway in Public Subnet 2

#### Steps

1. Open **VPC Console**.
2. Left menu → **NAT Gateways**.
3. Click **Create NAT Gateway**.
4. Configure:
  - **Subnet:** Public Subnet 2
  - **Elastic IP:** Allocate Elastic IP
  - **Name:** NAT-GW-AZ2
5. Click **Create NAT Gateway**.

**Create NAT gateway** Info

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

**NAT gateway settings**

**Name** - optional  
Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

**Subnet**  
Select a subnet in which to create the NAT gateway.

**Connectivity type**  
Select a connectivity type for the NAT gateway.  
 Public  
 Private

**Elastic IP allocation ID** Info  
Assign an Elastic IP address to the NAT gateway.

### Update Private Subnet 2 Route Table

1. VPC console → **Route Tables**.
2. Find the route table for **Private Subnet 2**
3. Select it → **Routes** tab → **Edit routes**.
4. Add route:
  - **Destination:** 0.0.0.0/0
  - **Target:** NAT Gateway → NAT-GW-AZ2

5. Click **Save changes**.

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
	Q_ local	X		
	NAT Gateway	Active	No	CreateRoute
	Q_ nat-00018f6648f2b7446	X		

Add route      Cancel      Preview      Save changes

## TASK 2 — Create a Launch Template

### 1. Open Launch Templates

EC2 Console → left menu → **Launch Templates** → **Create launch template**

### 2. Template Information

- Name: CafeWebServerTemplate

### 3. AMI

- Application & OS Images → **My AMIs**
- Select: **Cafe WebServer Image**

### 4. Instance Type

- Choose: **t2.micro**

### 5. Key Pair

- vokey

### 6. Network Settings

- Security Group: **CafeSG**

### 7. Tags

Add tag:

- Key:** Name
- Value:** webserver
- Resource:** Instances

### 8. IAM Role

- Under Advanced details:
  - IAM Instance Profile:** CafeRole

### 9. Create

Click **Create launch template**.

The screenshot shows the AWS EC2 Launch Templates page. On the left, there's a navigation sidebar with links like Dashboard, EC2 Global View, Instances, Launch Templates (selected), Images, and Elastic Block Store. The main content area has a title 'Launch Templates (1/1)' and a search bar. A table lists one launch template: 'Launch Template ID' (lt-0fd12de1a72db7cc9), 'Launch Template Name' (CafeWebServerTemplate), 'Default Version' (1), 'Latest Version' (1), 'Create Time' (2025-11-18T07:10:38.000Z), and 'Created By' (arnawssts:6374232). Below the table, a detailed view for 'CafeWebServerTemplate' is shown with tabs for 'Details', 'Versions', and 'Template tags'. The 'Details' tab shows the launch template ID, name, default version (1), and owner information (arnawssts:6374232 assumed role/vclabs/user4312508=VIKRAM\_TIM\_MANNA\_MADHYASTA).

## TASK 3 — Create Application Load Balancer

### 1. Open Load Balancer Console

EC2 → Load Balancers → **Create Load Balancer** → Application Load Balancer

### 2. ALB Configuration

- Name: CafeALB
- Scheme: **Internet-facing**
- IP type: IPv4
- VPC: *lab VPC*
- Subnets:
  - Public Subnet 1
  - Public Subnet 2
- Select security group

### 3. Create Target Group

- Target type: **Instances**
- Name: CafeTargetGroup
- Protocol: **HTTP**
- Health check path: /cafe
- Click **Create target group**

### 4.Create

#### Create Load Balancer

The screenshot shows the AWS EC2 Load Balancers console. On the left, there's a navigation sidebar with options like Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main area displays a table titled 'Load balancers (1/1)'. The table has columns for Name, State, Type, Scheme, IP address type, VPC ID, and Availability Zones. One row is selected, showing 'CafeALB' as the name, 'Active' as the state, 'application' as the type, 'Internet-facing' as the scheme, 'IPv4' as the IP address type, 'vpc-091f9a3f317680236' as the VPC ID, and '2 Availability Zones'. Below the table, a detailed view for 'Load balancer: CafeALB' is shown with tabs for Details, Listeners and rules, Network mapping, Resource map, Security, Monitoring, Integrations, Attributes, and Capacity. The 'Details' tab is selected, displaying information such as Load balancer type (Application), Status (Active), VPC (vpc-091f9a3f317680236), and Load balancer IP address type (IPv4).

## TASK 4 — Create Auto Scaling Group

### 1. Open ASG Console

EC2 → Auto Scaling Groups → Create Auto Scaling group

### 2. Basic Details

- ASG name: CafeWebServerASG
- Launch template: CafeWebServerTemplate

### 3. Network

- VPC: *your lab VPC*
- Subnets (select both):
  - Private Subnet 1
  - Private Subnet 2

### 4. Group Size

- Desired:** 2
- Min:** 2
- Max:** 6

### 5. Scaling Policy

- Select **Target tracking scaling policy**
  - Metric: **Average CPU Utilization**
  - Target: **25%**
  - Instance warmup: **60 sec**

### 6. Create ASG

Click **Create Auto Scaling group**.

Auto Scaling groups (1/1) [Info](#)

Last updated less than a minute ago [Edit](#) Launch configurations Launch templates Actions [Create Auto Scaling group](#)

Search your Auto Scaling groups

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
<a href="#">CafeWebServerASG</a>	<a href="#">CafeWebServerTemplate</a>   Version Default	2	-	2	2	6	2 Availability Zones

Auto Scaling group: **CafeWebServerASG**

Details Integrations Automatic scaling Instance management Instance refresh Activity Monitoring Tags - moved

**CafeWebServerASG Capacity overview**

arn:aws:autoscaling:us-east-1:637423272929:autoScalingGroup:cc5fc03b-b996-40b9-8630-139cd5d30f21:autoScalingGroupName/CafeWebServerASG

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
2	2 - 6	Units (number of instances)	-

## Task 5 — Testing and Validation

### 5.1 Application Reachability Test

The ALB DNS name was used to verify that the café application was accessible:

<http://cafealb-641114072.us-east-1.elb.amazonaws.com/cafe/>

This confirmed:

- Instances were healthy
- Target group was functioning
- ALB routing was operational

AWS Academy CloudWatch Metrics Challenge (Café) Instances | EC2 | us-east-1 | Auto Scaling group: Cafe! RouteTables | VPC Configuration | +

Not secure cafealb-641114072.us-east-1.elb.amazonaws.com/cafe/

Home About Us Contact Us Menu Order History

Café

Frank bakes a rich variety of cookies. Try them all!

Tea, Coffee, Lattes.

Our tarts are always a customer favorite!

13:09 18-11-2025 ENG IN

## 5.2 Auto Scaling Stress Test

To validate autoscaling behavior, a stress test was performed using **AWS Systems Manager Session Manager**.

### Commands to Run:

```
sudo amazon-linux-extras install epel
```

```
sudo yum install stress -y
```

```
stress --cpu 1 --timeout 600
```

This generated artificial CPU load.

The ASG detected CPU >25% and initiated **scale-out**, launching additional EC2 instances.

Monitoring the ASG dashboard confirmed:

- Scaling policies were applied correctly
- New instances launched in both private subnets
- Load gradually balanced across instances

```
Installing:
 stress                               x86_64                         1.0.4-16.el7                    epel                           39 k
Transaction Summary
Install 1 Package
Total download size: 39 k
Installed size: 94 k
Downloading packages:
warning: /var/cache/yum/x86_64/2/epel/packages/stress-1.0.4-16.el7.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID 352c64e5: NOKEY
Public key for stress-1.0.4-16.el7.x86_64.rpm is not installed
stress-1.0.4-16.el7.x86_64.rpm
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7
Importing GPG key 0x352C64E5:
 Userid   : "Fedora EPEL (7) <epel@fedoraproject.org>"
 Fingerprint: 91e9 7d7c 4a5e 96f1 7f3e 888f 6a2f aea2 352c 64e5
 Package   : epel-release-7-11.noarch (@amzn2extra-epel)
 From     : /etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7
Running transaction check
Running transaction test
transaction test succeeded
Running transaction
  Installing : stress-1.0.4-16.el7.x86_64
  Verifying   : stress-1.0.4-16.el7.x86_64
                                                               1/1
                                                               1/1
Installed:
  stress.x86_64 0:1.0.4-16.el7
Complete!
rh-4.25 stress --cpu 1 --timeout 600
stress: info: [3367] dispatching hogs: 1 cpu, 0 io, 0 vm, 0 hdd
```

The screenshot shows the AWS CloudWatch Metrics Insights interface. A search bar at the top contains the query: `stress`. Below the search bar, a table displays metric data over time. The columns include Metric Name, Value, and Time. The data shows a sharp peak in stress values starting around 2023-06-20T10:00:00Z, with values fluctuating between 0.0 and 1.0. A legend indicates that the orange line represents the maximum value and the blue line represents the average value.

Metric Name	Value	Time
max	1.0	2023-06-20T10:00:00Z
average	0.5	2023-06-20T10:00:00Z
min	0.0	2023-06-20T10:00:00Z

**Instances (4) Info**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
webserver	i-043869f27f8393563	Running	t2.micro	2/2 checks passed	<a href="#">View alarms +</a>	us-east-1a	-
webserver	i-0cdc809a4a99aa173	Shutting-down	t2.micro	-	<a href="#">View alarms +</a>	us-east-1a	-
webserver	i-0f94678aeba34169a	Running	t2.micro	2/2 checks passed	<a href="#">View alarms +</a>	us-east-1b	-
CafeWebAppS...	i-002df20ccf01c176d	Running	t2.micro	2/2 checks passed	<a href="#">View alarms +</a>	us-east-1a	ec2-54-196-11-149

**Auto Scaling groups (1) Info**

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
CafeWebServerASG	CafeWebServerTemplate   Version Default	6	Updating capacity...	3	2	6

**Instances (9) Info**

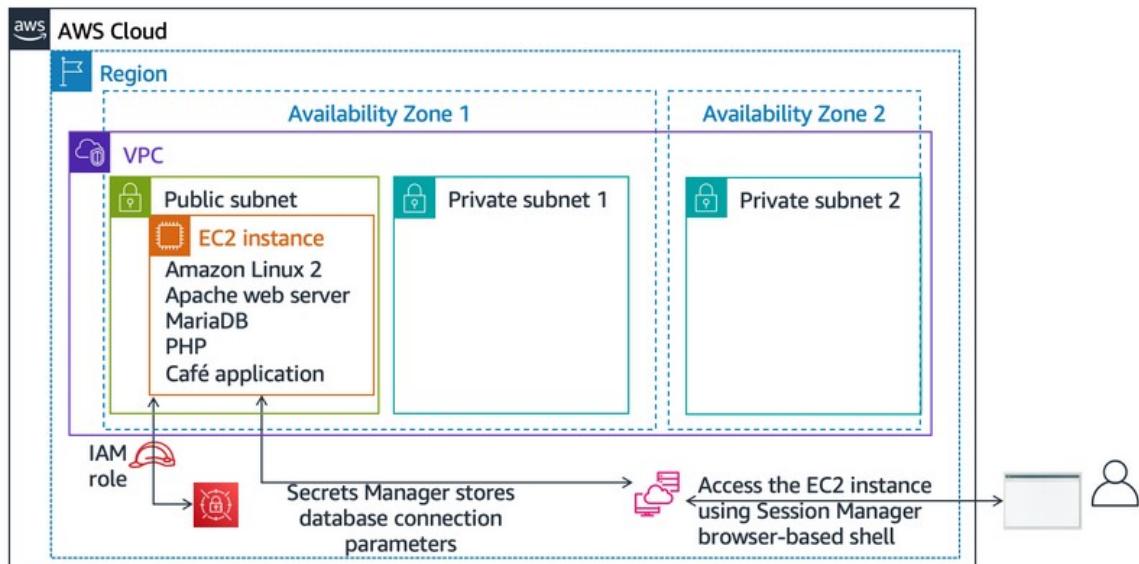
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
webserver	i-043869f27f8393563	Running	t2.micro	2/2 checks passed	<a href="#">View alarms +</a>	us-east-1a	-
webserver	i-0cdc809a4a99aa173	Terminated	t2.micro	-	<a href="#">View alarms +</a>	us-east-1a	-
webserver	i-0f94678aeba34169a	Terminated	t2.micro	-	<a href="#">View alarms +</a>	us-east-1b	-
webserver	i-0763a15c42aedd3f4	Running	t2.micro	2/2 checks passed	<a href="#">View alarms +</a>	us-east-1b	-
webserver	i-0b36ce2b70e23b2d2	Running	t2.micro	Initializing	<a href="#">View alarms +</a>	us-east-1b	-
webserver	i-0ac8d48636c050975	Running	t2.micro	Initializing	<a href="#">View alarms +</a>	us-east-1b	-
CafeWebAppS...	i-002df20ccf01c176d	Running	t2.micro	2/2 checks passed	<a href="#">View alarms +</a>	us-east-1a	ec2-54-196-11-149
webserver	i-0f6bb9080fb56fa	Running	t2.micro	Initializing	<a href="#">View alarms +</a>	us-east-1a	-
webserver	i-078dd64e1b4e6ae0	Running	t2.micro	Initializing	<a href="#">View alarms +</a>	us-east-1a	-

## Conclusion

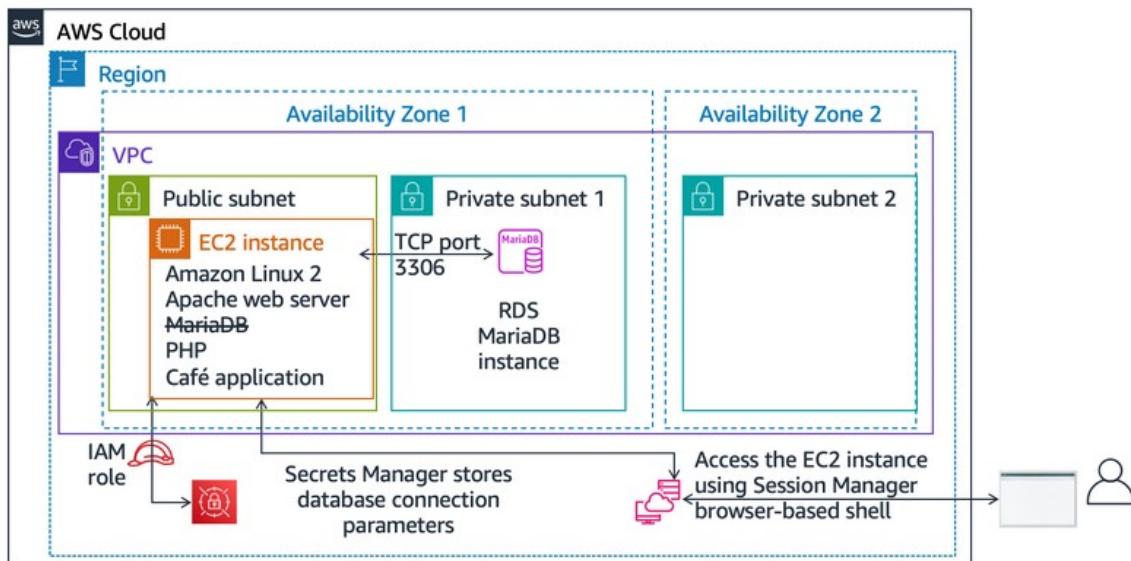
This lab successfully demonstrated the deployment of a highly available, fault tolerant, and scalable web application architecture using core AWS services. A multi-AZ design combined with NAT Gateways, private subnets, an ALB, and an Auto Scaling Group ensures robust performance and resiliency.

# CA, Mod4; Challenge (Cafe) lab: Migrating a Database to Amazon RDS

Initially :



Final output :



## Step 1: Creating an RDS Instance

1. Go to the **Amazon RDS console** → **Create database**.

The screenshot shows the Aurora and RDS Dashboard. On the left sidebar, there are several sections: **Aurora and RDS**, **Dashboard** (selected), **Databases**, **Query editor**, **Performance insights**, **Snapshots**, **Exports in Amazon S3**, **Automated backups**, **Reserved instances**, **Proxies**, **Subnet groups**, **Parameter groups**, **Option groups**, **Custom engine versions**, **Zero-ETL integrations**, **Events**, **Event subscriptions**, **Recommendations** (0), and **Certificate update**.

**Resources** section (top right):
 

- You are using the following Amazon RDS resources in the US East (N. Virginia) region (used/quota)
- DB Instances (0/40)**: Allocated storage (0 TB/100 TB). Instances and storage include Neptune and DocumentDB. [Increase DB instances limit ↗](#)
- DB Clusters (0/40)**: Reserved instances (0/40)
- Snapshots (0)**: Manual (DB Cluster (0/100), DB Instance (0/100))
- Automated**: DB Cluster (0), DB Instance (0)
- Recent events: (0)**: Event subscriptions (0/20)
- Parameter groups (1)**: Default (1), Custom (0/100)
- Option groups (1)**: Default (1), Custom (0/20)
- Subnet groups (1/50)**: VPC
- Supported platforms ↗**: Default network vpc-0b7ef037d9d053d25

**Create a database** section (bottom right):
 

- Amazon Relational Database Service (RDS) makes it easy to set up, operate, and scale a relational database in the cloud.
- Create a database** button
- Note: your DB instances will launch in the **US East (N. Virginia)** region
- Restore from S3** button
- You can use a backup from Amazon S3 to restore and create a new Aurora MySQL and MySQL database.

2. Choose a database creation method: **Standard create**
3. Engine type: **MariaDB**.
4. Template: **Free tier**.
5. Availability & durability: **Single-AZ DB instance deployment (1 instance)**
6. Settings:
  - DB instance identifier: **Cafedb**
  - Master username: **admin**
  - Credentials management: **self managed**
  - Master password: **Msis1234**
7. Instance configuration:
  - DB instance class: **db.t3.micro** (Burstable classes)
8. Storage:
  - Storage type: **General Purpose SSD (gp3)**

- Allocated storage: **20 GiB**

## 9. Connectivity:

- Virtual private cloud (VPC): select **Lab VPC**
- DB subnet group: **lab-db-subnet-group**
- Existing VPC security groups: **dbSG** (uncheck default)

## 10. Monitoring: Uncheck **Enable Enhanced monitoring**

## 11. Click **Create database.**

Aurora and RDS > Databases > Create database

**Create database** Info

**Choose a database creation method**

Standard create You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

**Engine options**

Engine type Info

Aurora (MySQL Compatible) 

Aurora (PostgreSQL Compatible) 

MySQL 

PostgreSQL 

MariaDB 

Oracle 

Microsoft SQL Server 

IBM Db2 

**Edition**

MySQL Community

**Engine version** Info

View the engine versions that support the following database features.

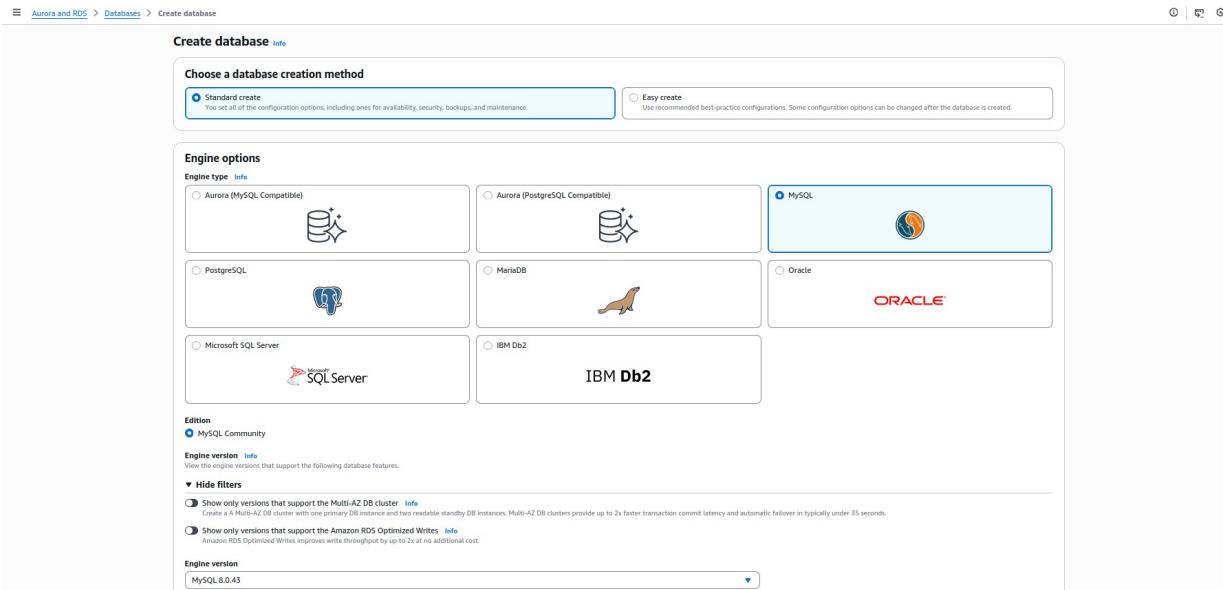
Hide filters

Show only versions that support the Multi-AZ DB cluster Info  
Create a Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

Show only versions that support the Amazon RDS Optimized Writes Info  
Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

**Engine version**

MySQL 8.0.43



**Engine version**

MySQL 8.0.43

Enable RDS Extended Support Info  
Amazon RDS Extended Support is a paid offering. By selecting this option, you consent to being charged for this offering if you are running your database major version past the RDS end of standard support date for that version. Check the end of standard support date for your major version in the [RDS for MySQL documentation](#).

**Templates**

Choose a sample template to meet your use case.

Production Use defaults for high availability and fast, consistent performance.

Dev/Test This instance is intended for development use outside of a production environment.

Free tier Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.

**Availability and durability**

Deployment options Info

Choose the deployment option that provides the availability and durability needed for your use case. AWS is committed to a certain level of uptime depending on the deployment option you choose. Learn more in the [Amazon RDS service level agreement \(SLA\)](#).

Single-AZ DB instance deployment (1 instance) Creates a single DB instance without standby instances. This setup provides:

- 99.9% uptime
- No data redundancy

Multi-AZ DB instance deployment (2 instances) Creates a primary DB instance with a non-readable standby instance in a separate Availability Zone. This setup provides:

- 99.99% uptime
- Redundancy across Availability Zones

Multi-AZ DB cluster deployment (3 instances) Creates a primary DB instance with two readable standbys in separate Availability Zones. This setup provides:

- 99.99% uptime
- Redundancy across Availability Zones
- Increased read capacity
- Reduced write latency

**Settings**

**DB instance identifier** Info

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

cafef0

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 63 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

**Credentials Settings**

Master username Info

Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - most secure RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Auto generate password Amazon RDS can generate a password for you, or you can specify your own password.

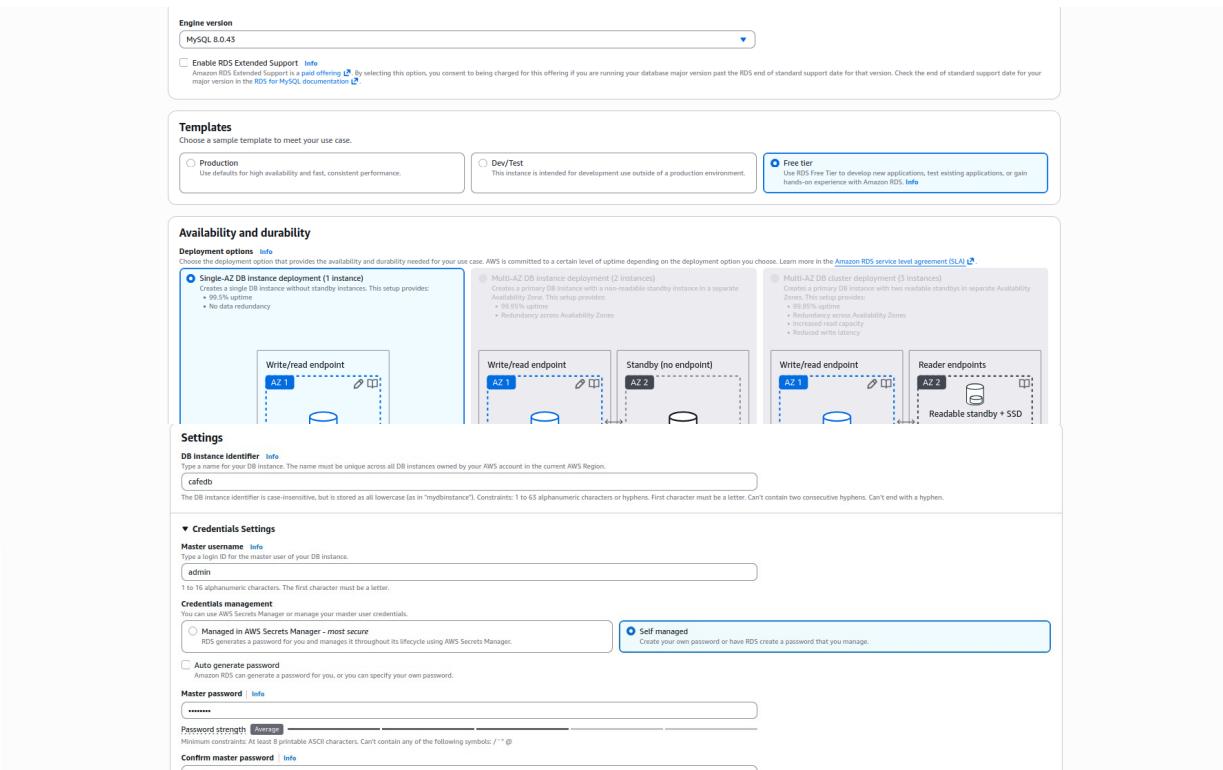
**Master password** Info

\*\*\*\*\*

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / \ \* ?

**Confirm master password** Info

\*\*\*\*\*



### Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

#### DB instance class Info

Hide filters

Show instance classes that support Amazon RDS Optimized Writes Info

Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Include previous generation classes

Standard classes (includes m classes)

Memory optimized classes (includes t and x classes)

Burstable classes (includes t classes)

db.t3.micro

**Storage**

**Storage type** [Info](#)  
Provisioned IOPS SSD (io2) storage volumes are now available.

**General Purpose SSD (gp3)**  
Performance scales independently from storage

**Allocated storage** [Info](#)  
20  GiB  
Minimum: 20 GiB; Maximum: 6,144 GiB

**Provisioned IOPS** [Info](#)  
3000  IOPS  
Baseline IOPS of 3,000 IOPS is included for allocated storage less than 400 GiB.

**Storage throughput** [Info](#)  
125  MiBps  
Baseline storage throughput of 125 MiBps is included for allocated storage less than 400 GiB.

**To provision additional IOPS and throughput, increase the allocated storage to 400 GiB or greater.**

**Additional storage configuration**

**Connectivity** [Info](#)

**Compute resource**  
Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

**Don't connect to an EC2 compute resource**  
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

**Connect to an EC2 compute resource**  
Set up a connection to an EC2 compute resource for this database.

**Virtual private cloud (VPC)** [Info](#)  
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

**Lab VPC (vpc-03190cd5d2f18e199)**  
5 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

**After a database is created, you can't change its VPC.**

**DB subnet group** [Info](#)  
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

**lab-db-subnet-group**  
2 Subnets, 2 Availability Zones

**Public access** [Info](#)

**Yes**  
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

**No**  
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

**VPC security group (firewall)** [Info](#)  
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

**Choose existing**  
Choose existing VPC security groups

**Create new**  
Create new VPC security group

**Existing VPC security groups**  
Choose one or more options  
**dbSG**

**Availability Zone** [Info](#)  
No preference

**RDS Proxy**  
RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

**Create an RDS Proxy** [Info](#)  
RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

**Certificate authority - optional** [Info](#)  
Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

**rds-ca-rsa2048-1 (default)**  
Expires May 26, 2021

If you don't select a certificate authority, RDS chooses one for you.

**Additional configuration**

**Tags - optional**  
A tag consists of a case-sensitive key-value pair.  
No tags associated with the resource.

**Add new tag**  
You can add up to 50 more tags.

**Database authentication**

**Database authentication options** [Info](#)

**Password authentication**  
Authenticates using database passwords.

**Password and IAM database authentication**  
Authenticates using the database password and user credentials through AWS IAM users and roles.

**Password and Kerberos authentication**  
Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.

**Monitoring** [Info](#)  
Choose monitoring tools for this database. Database Insights provides a combined view of Performance Insights and Enhanced Monitoring for your fleet of databases. Database Insights pricing is separate from RDS monthly estimates. See [Amazon CloudWatch pricing](#).

**Database Insights - Advanced**  
• Retains 15 months of performance history  
• Fleet-level monitoring  
• Integration with CloudWatch Application Signals

**Database Insights - Standard**

**Additional monitoring settings**  
Enhanced Monitoring, CloudWatch Logs and DevOps Guru

**Enhanced Monitoring**  
 **Enable Enhanced monitoring**  
Enabling Enhanced Monitoring metrics are useful when you want to see how different processes or threads use the CPU.

**Log exports**  
Select the log types to publish to Amazon CloudWatch Logs

Audit log  
 Error log  
 General log  
 iam-db-auth-error log  
 Slow query log

**IAM role**  
The following service-linked role is used for publishing logs to CloudWatch Logs.

RDS service-linked role

**Additional configuration**  
Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

**Estimated monthly costs**  
The Amazon RDS Free Tier is available to you for 12 months. Each calendar month, the free tier will allow you to use the Amazon RDS resources listed below for free:

- 750 hrs of Amazon RDS in a Single-AZ db.t2.micro, db.t3.micro or db.t4g.micro instance.
- 20 GB of General Purpose Storage (SSD).
- 20 GB for automated backup storage and any user-initiated DB Snapshots.

[Learn more about AWS Free Tier](#)

When your free usage expires or if your application use exceeds the free usage tiers, you simply pay standard, pay-as-you-go service rates as described in the [Amazon RDS Pricing page](#).

**You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.**

**Cancel** **Create database**

## Step 3: In Session Manager

1. Create session → select cafe server → Start session.

2. Run the following commands:

```
sudo su
```

```
su ec2-user
```

```
cd /var/www/html/cafe/
```

```
sudo mysql -u root -p //paste the secretes password from Secretes Manager
```

The screenshot shows the AWS Systems Manager Session Manager landing page. The left sidebar contains navigation links for Review node insights, Explore nodes, Diagnose and remediate, Just-in-time node access, Settings, Node Tools (Compliance, Distributor, Fleet Manager, Hybrid Activations, Inventory, Patch Manager, Run Command, Session Manager, State Manager), Change Management Tools (Automation, Change Calendar, Change Manager, Documents, Maintenance Windows, Quick Setup), and Application Tools (AppConfig). The main content area features a "Session Manager" title and sub-sections: "How it works" (3 steps: Configure instances, Assign IAM policies, Start session), "Why use Session Manager?" (Improved security posture, Centralized access control), and "Getting started" (links to What Is Session Manager, Set up Session Manager, Set up session logging, Set up session notifications, Create and manage sessions, Monitor session activity). A "More resources" section includes Documentation, FAQs, and the Systems Manager forum.

The screenshot shows the "Specify target" step of the Session Manager wizard. The left sidebar is identical to the previous screenshot. The main content area shows the "Reason" section with a "Reason for session – optional" input field containing "Enter reason". Below it is the "Target instances" section, which lists a single instance: "CafeServer" (Instance ID: i-0d4c030609ed0f47d, Agent version: 3.3.3050.0, Instance state: running, Availability zone: us-east-1a, Platform: Amazon Linux). Buttons at the bottom right include "Start session" (disabled), "Cancel", and "Next".

Session ID: user4312509=Aman\_Arun\_Sanil-ag8lg4prretjjcfvrsfdele0    Shortcuts    Instance ID: i-0d4c030609ed0f47d

sh-4.2\$ sudo su  
[root@cafeserver bin]# su ec2-user  
[ec2-user@cafeserver bin]# sudo mysql -u root -p  
Enter password: [REDACTED]

## To retrieve password from Secrets Manager :

AWS Secrets Manager > Secrets

**Secrets**

Filter secrets by name, description, tag key, tag value, owning service or primary Region

Secret name	Description	Last retrieved (UTC)
rds-db-credentials/db-PQTIRJLMZWFH75JCPEVM2BB4/admin/1763376778787	RDS database admin credentials for database-1	-
/cafe/dbPassword	-	-
/cafe/dbUser	-	-
/cafe/dbName	-	-
/cafe/dbUrl	-	-
/cafe/currency	-	-
/cafe/timeZone	-	-
/cafe/showServerInfo	-	-

**/cafe/dbPassword**

**Secret details**

Encryption key: aws/secretsmanager  
Secret name: /cafe/dbPassword  
Secret ARN: arn:aws:secretsmanager:us-east-1:992382621222:secret:/cafe/dbPassword-a2tbGZ

Secret description: -  
Secret type: -

**Actions**

**Overview** | **Rotation** | **Versions** | **Replication** | **Tags**

**Secret value** Info  
Retrieve and view the secret value.

Key/value  **Plaintext**

Re:Start!

**Resource permissions - optional** Info  
Add or edit a resource policy to access secrets across AWS accounts.

**Edit permissions**

**Sample code**

## In Session Manager after entering into mysql :

- > show Databases;
- > use cafe\_db;
- > show tables;

```
Session ID: user4312509-Aman_Arun_Sanil-ag8lg4prrretjcfvsrfeloe [Shortcuts] Instance ID: i-0d4c030609ed0f47d [Terminate]
sh-4.2$ sudo su
[ec2-user@cafeserver bin]$ su ec2-user
[ec2-user@cafeserver bin]$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 11
Server version: 10.2.38-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\e' to clear the current input statement.

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| cafe_db |
| information_schema |
| mysql |
| performance_schema |
| test |
+-----+
5 rows in set (0.00 sec)

MariaDB [(none)]> use cafe_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [cafe_db]> show tables;
+-----+
| Tables_in_cafe_db |
+-----+
| order |
| order_item |
| product |
| product_group |
+-----+
4 rows in set (0.00 sec)

MariaDB [cafe_db]> exit
Bye
[ec2-user@cafeserver bin]$
```

## In EC2, security groups > Edit inbound Rules of DB security group:

The screenshot shows the AWS EC2 Security Groups page. The left sidebar navigation includes EC2, Instances, Images, Elastic Block Store, Network & Security, and other services. The main content area displays the details for the security group 'sg-03dc4490dd9dad09e - dbSG'. The 'Details' section shows the security group name 'dbSG', owner '992382621222', security group ID 'sg-03dc4490dd9dad09e', and VPC ID 'vpc-0bae965cd93305db5'. The 'Inbound rules' tab is selected, showing a table with no results found. The table columns include Name, Security group rule ID, IP version, Type, Protocol, Port range, Source, and Description.

select MYSQL/AURORA → CUSTOM → CAFESG

The screenshot shows the 'Edit inbound rules' configuration page for the security group 'sg-03dc4490dd9dad09e - dbSG'. The top navigation bar includes 'Edit inbound rules' and 'Info' tabs. The 'Info' tab is selected, stating that inbound rules control incoming traffic. The main form allows setting the protocol (TCP), port range (3306), source (Custom), and description. A dropdown menu for 'Security group rule ID' is set to 'MySQL/Aurora'. A table lists one existing rule: 'sg-00f2fd5c6bd4825d1'. At the bottom, there are 'Cancel', 'Preview changes', and 'Save rules' buttons.

**Summary**

DB identifier	Status	Role	Engine
database-1	Backing-up	Instance	MySQL Community
CPU	Class	Current activity	Region & AZ
-	db.t4g.micro	-	us-east-1b

**Connectivity & security**

Endpoint & port	Networking	Security
Endpoint: database-1.ck2wa68m7pz.us-east-1.rds.amazonaws.com	Availability Zone: us-east-1b	VPC security groups: dbSG (sg-03dc4490dd9dad09e) Active
Port: 3306	VPC: Lab VPC (vpc-0bae965cd93305db5)	Publicly accessible: No
	Subnet group: lab-db-subnet-group	Certificate authority: Info rds-ca-rsa2048-g1
	Subnets: subnet-0d1d2c64097a99494, subnet-07bdb5df8f1f3a5d9	Certificate authority date: May 26, 2061, 05:04 (UTC+05:30)
	Network type: IPv4	DB instance certificate expiration date: November 17, 2026, 16:25 (UTC+05:30)

**Connected compute resources (0)**

```
| ec2-user@cafeserver:~$ cd
| ec2-user@cafeserver:~$ mysqldump --database cafe_db -u root -p > cafe1.sql
Info: Using unique option prefix 'database' is error-prone and can break in the future. Please use the full name 'databases' instead.
Enter password: [REDACTED]
```

## In RDS, Copy the RDS Endpoint :

```
| ec2-user@cafeserver:~$ ls | grep cafe1.sql
cafe1.sql
| ec2-user@cafeserver:~$ mysql -h database-1.ck2wa68m7pz.us-east-1.rds.amazonaws.com -u admin -p
```

// here paste the rds endpoint and password as Msis1234

**Summary**

DB identifier	Status	Role	Engine
database-1	Available	Instance	MySQL Community
CPU	Class	Current activity	Region & AZ
5.77%	db.t4g.micro	0 Connections	us-east-1b

**Connectivity & security**

Endpoint & port	Networking	Security
Endpoint copied: database-1.ck2wa68m7pz.us-east-1.rds.amazonaws.com	Availability Zone: us-east-1b	VPC security groups: dbSG (sg-03dc4490dd9dad09e) Active
Port: 3306	VPC: Lab VPC (vpc-0bae965cd93305db5)	Publicly accessible: No
	Subnet group: lab-db-subnet-group	Certificate authority: Info rds-ca-rsa2048-g1
	Subnets: subnet-0d1d2c64097a99494, subnet-07bdb5df8f1f3a5d9	Certificate authority date: May 26, 2061, 05:04 (UTC+05:30)
	Network type: IPv4	DB instance certificate expiration date: November 17, 2026, 16:25 (UTC+05:30)

**Connected compute resources (0)**

```

Session ID: user4312509=Aman_Arun_Sanil-ag8lg4prretjcfvsrfdeoe [Shortcuts] Instance ID: i-0d4c030609ed0f47d
[ec2-user@cafeserver ~]$ ls | grep cafe1.sql
cafe1.sql
[ec2-user@cafeserver ~]$ mysql -h database-1.clk2wa68m7pz.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 8.0.43 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> create database cafe_db;
Query OK, 1 row affected (0.01 sec)

MySQL [(none)]> use cafe_db;
Database changed
MySQL [cafe_db]> source cafe1.sql;
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected, 1 warning (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 1 row affected (0.01 sec)

Database changed
Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected, 1 warning (0.00 sec)

[ec2-user@cafeserver ~]$
```

```

MySQL [cafe_db]> show tables;
+-----+
| Tables_in_cafe_db |
+-----+
| order |
| order_item |
| product |
| product_group |
+-----+
4 rows in set (0.00 sec)

MySQL [cafe_db]> exit;
Bye
[ec2-user@cafeserver ~]$
```

in mysql run the follwing commands;

```

> create database cafe_db;
> use cafe_db;
> show tables;
> source cafe1.sql;
> show tables;
> exit;
```

```

sudo systemctl stop mariadb.service
sudo systemctl status mariadb.service // shows inactive, dead
sudo systemctl restart httpd
```

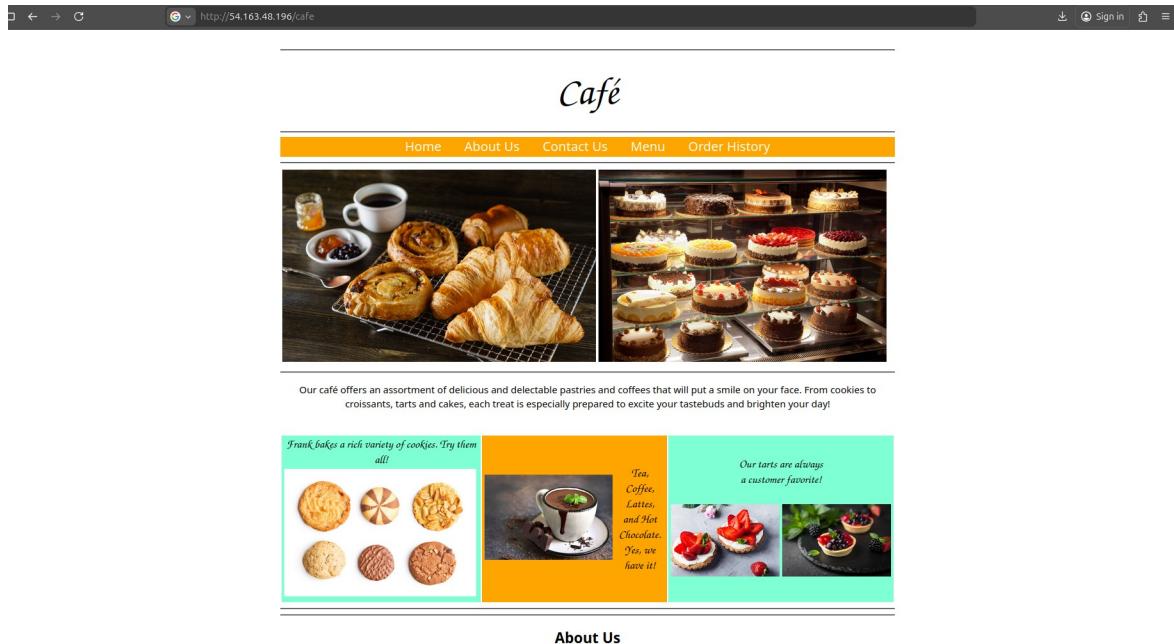
```

Session ID: user4312509=Aman_Arun_Sanil-e982gootsuxbpv98139gcvxu5e [Shortcuts] Instance ID: i-0d4c030609ed0f47d
[ec2-user@cafeserver ~]$ sudo systemctl stop mariadb.service
[ec2-user@cafeserver ~]$ sudo systemctl restart httpd
[ec2-user@cafeserver ~]$
```

## Test the Cafe Application on the EC2 Instance

1. Go to **EC2 console** → **Instances** → **CafeServer**.
2. Copy the public address.

3. Access the application:  
<http://<public-ip>/cafe>



The screenshot shows a web application for a cafe. The main content area includes a header with the word "Café" in a stylized font, a navigation menu with links to Home, About Us, Contact Us, Menu, and Order History, and a main content area featuring images of baked goods and a display of various cakes. Below the images is a paragraph describing the cafe's offerings. Three separate sections below the images provide more details: one about cookies, one about coffee, and one about tarts. The "About Us" section is located at the bottom of the main content area.

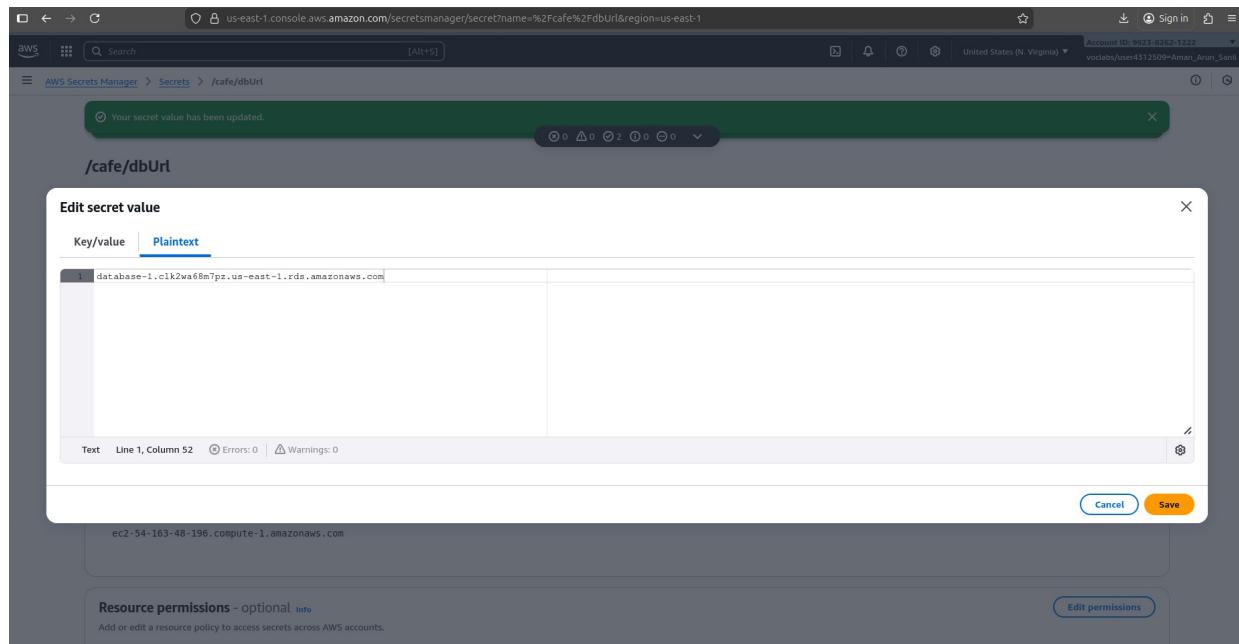
For the Menu and Order to work do the below changes in **Secrets Manager**

#### In Secrets Manager change:

dbUrl → paste Endpoint

dbPassword → Msis1234

dbUser → admin



The screenshot shows the AWS Secrets Manager console with a modal dialog titled "Edit secret value". The "Key/value" tab is selected, and the value field contains the URL "database-1.clk2wa68m7px.us-east-1.rds.amazonaws.com". The "Save" button is highlighted in orange at the bottom right of the modal. The background shows the AWS navigation bar and some other secret details.

AWS Secrets Manager > Secrets > /cafe/dbPassword

### Secret details

Edit secret value

Key/value | **Plaintext**

```
1 |
```

Text Line 1, Column 1 | Errors: 0 | Warnings: 0

Cancel Save

Resource permissions - optional Info

Add or edit a resource policy to access secrets across AWS accounts.

Edit permissions

Sample code

aws [Alt+S] United States (N. Virginia) vociabs/user4312509=Aman\_Arun\_Sam

AWS Secrets Manager > Secrets > /cafe/dbUser

Your secret value has been updated.

### /cafe/dbUser

Edit secret value

Key/value | **Plaintext**

```
1 | admin|
```

Text Line 1, Column 6 | Errors: 0 | Warnings: 0

Cancel Save

Resource permissions - optional Info

Add or edit a resource policy to access secrets across AWS accounts.

Edit permissions

aws [Alt+S] United States (N. Virginia) vociabs/user4312509=Aman\_Arun\_Sam

<http://<public-ip>/cafe>

//check the Menu and Order up and running

*Café*

[Home](#) [Menu](#) [Order History](#)

*Order History*

Order Number: 24 Date: 2020-07-28 Time: 13:14:07 Total Amount: \$35.00			
Item	Price	Quantity	Amount
Strawberry Blueberry Tart	\$3.50	4	\$14.00
Strawberry Tart	\$3.50	3	\$10.50
Latte	\$3.50	3	\$10.50

Order Number: 23 Date: 2020-07-28 Time: 13:13:54 Total Amount: \$6.00			
Item	Price	Quantity	Amount
Coffee	\$3.00	2	\$6.00

Order Number: 22 Date: 2020-07-21 Time: 13:13:47 Total Amount: \$33.50			
Item	Price	Quantity	Amount
Chocolate Chip Cookie	\$2.50	3	\$7.50
Strawberry Blueberry Tart	\$3.50	4	\$14.00
Coffee	\$3.00	4	\$12.00

Order Number: 21 Date: 2020-07-20 Time: 13:13:36 Total Amount: \$17.50			
Item	Price	Quantity	Amount
Latte	\$3.50	5	\$17.50

Order Number: 20 Date: 2020-07-18 Time: 13:13:27 Total Amount: \$14.00			
Item	Price	Quantity	Amount
Donut	\$1.00	5	\$5.00
Hot Chocolate	\$3.00	3	\$9.00



**Croissant**  
\$1.50  
Fresh, buttery and fluffy... Simply delicious!  
Quantity:



**Donut**  
\$1.00  
We have more than half-a-dozen flavors!  
Quantity:



**Chocolate Chip Cookie**  
\$2.50  
Made with Swiss chocolate with a touch of Madagascar vanilla  
Quantity:



**Muffin**  
\$3.00  
Banana bread, blueberry, cranberry or apple  
Quantity:



**Strawberry Blueberry Tart**  
\$3.50  
Bursting with the taste and aroma of fresh fruit  
Quantity:



**Strawberry Tart**  
\$3.50  
Made with fresh ripe strawberries and a delicious whipped cream  
Quantity:

# CFA LAB

## Challenge (Cafe) lab: Creating a Static Website for the Cafe

1. At the top of these instructions, choose **Start Lab**.
  - a. The lab session starts.
  - b. A timer displays at the top of the page and shows the time remaining in the session.

**Tip:** To refresh the session length at any time, choose **Start Lab** again before the timer reaches 0:00.

- c. Before you continue, wait until the circle icon to the right of the [AWS](#) link in the upper-left corner turns green. When the lab environment is ready, the **AWS Details** panel displays.
2. To connect to the AWS Management Console, choose the [AWS](#) link in the upper-left corner above the terminal window.
  - a. A new browser tab opens and connects you to the console.

**Tip:** If a new browser tab does not open, a banner or icon is usually at the top of your browser with a message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and then choose **Allow pop-ups**.

3. Arrange the AWS Management Console tab so that it displays alongside these instructions. Ideally, you have both browser tabs open at the same time so that you can follow the lab steps.

Nov 17 15:00

Challenge (Café) lab: Creating a Static Website for the Café

awsacademy.instructure.com/courses/131613/assignments/1511928?module\_item\_id=12598703

ACAv3EN-US-LT13-131613 > Assignments > Challenge (Café) lab: Creating a Static Website for the Café > Challenge (Café) lab: Creating a Static Website for the Café

Home      Challenge (Café) lab: Creating a Static Website for the Café

Modules      Due No Due Date Points 29 Submitting an external tool

Discussions      Grades 1      Lucid (Whiteboard)

AWS

EN\_US

Submit      Submission Report      Grades

# Challenge Lab: Creating a Static Website for the Café

## Scenario

Frank and Martha are a husband-and-wife team who own and operate a small café business that sells desserts and coffee. Their daughter, Sofia, and their other employee, Nikhil (who is a secondary school student), also work at the café. The café has a single location in a large city.

The café currently doesn't have a marketing strategy. It gains new customers mostly when someone walks by, notices the café, and decides to try it. The café has a reputation for high-quality desserts and coffees, but the café's reputation is limited to people who have visited or who have heard about it from other café

◀ Previous      Next ▶

The screenshot shows a web browser window titled "Challenge (Café) lab: Creating a Static Website for the Café". The URL is [awsacademy.instructure.com/courses/131613/assignments/1511928?module\\_item\\_id=12598703](https://awsacademy.instructure.com/courses/131613/assignments/1511928?module_item_id=12598703). The page displays the assignment details: "Challenge (Café) lab: Creating a Static Website for the Café", "Due No Due Date", "Points 29", and "Submitting an external tool". A sidebar on the left contains links for Home, Modules, Discussions, Grades, Courses (selected), Calendar, Inbox, History, and Help. The main content area shows the challenge scenario:

## Challenge Lab: Creating a Static Website for the Café

### Scenario

Frank and Martha are a husband-and-wife team who own and operate a small café business that sells desserts and coffee. Their daughter, Sofia, and their other employee, Nikhil (who is a secondary school student), also work at the café. The café has a single location in a large city.

The café currently doesn't have a marketing strategy. It gains new customers mostly when someone walks by, notices the café, and decides to try it. The café has a reputation for high-quality desserts and coffees, *but the café's reputation is limited to people who have visited or who have heard about it from other café*

Navigation buttons at the bottom include "◀ Previous" and "Next ▶".

## Task 2: Creating an S3 bucket to host your static website

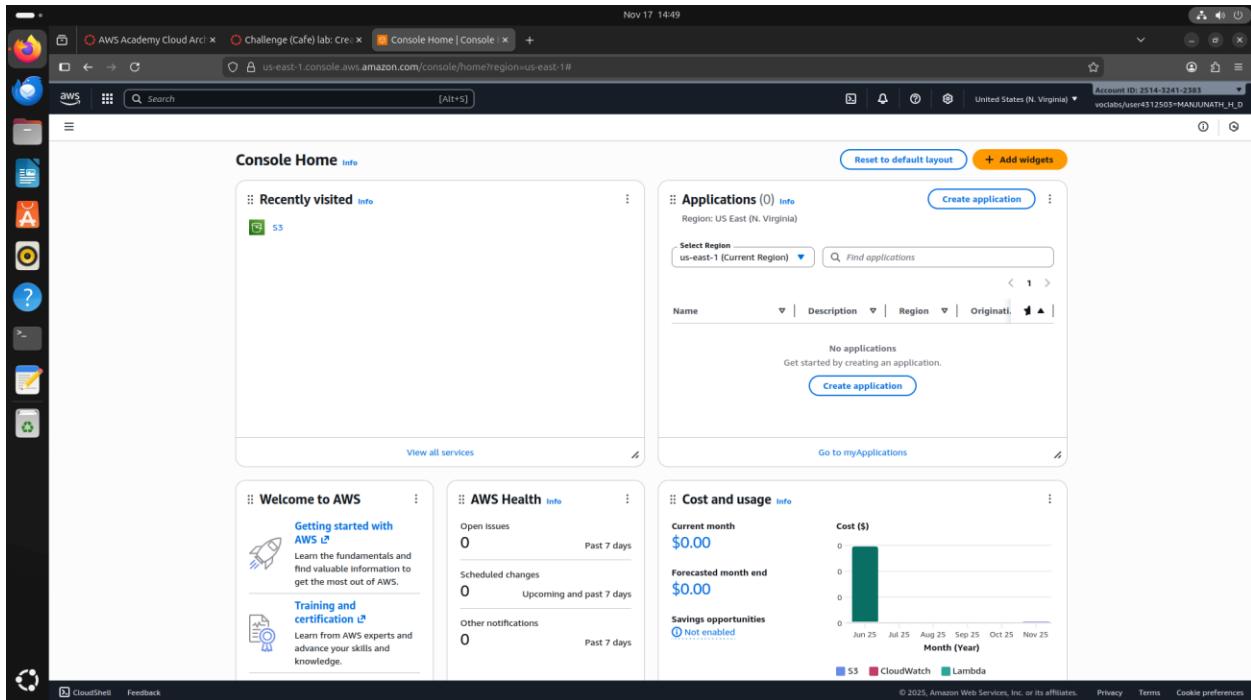
In this task, you create an S3 bucket and configure it to host your static website.

6. Open the Amazon S3 console.
7. Create a bucket in the **US East (N. Virginia) us-east-1** AWS Region to host your static website.

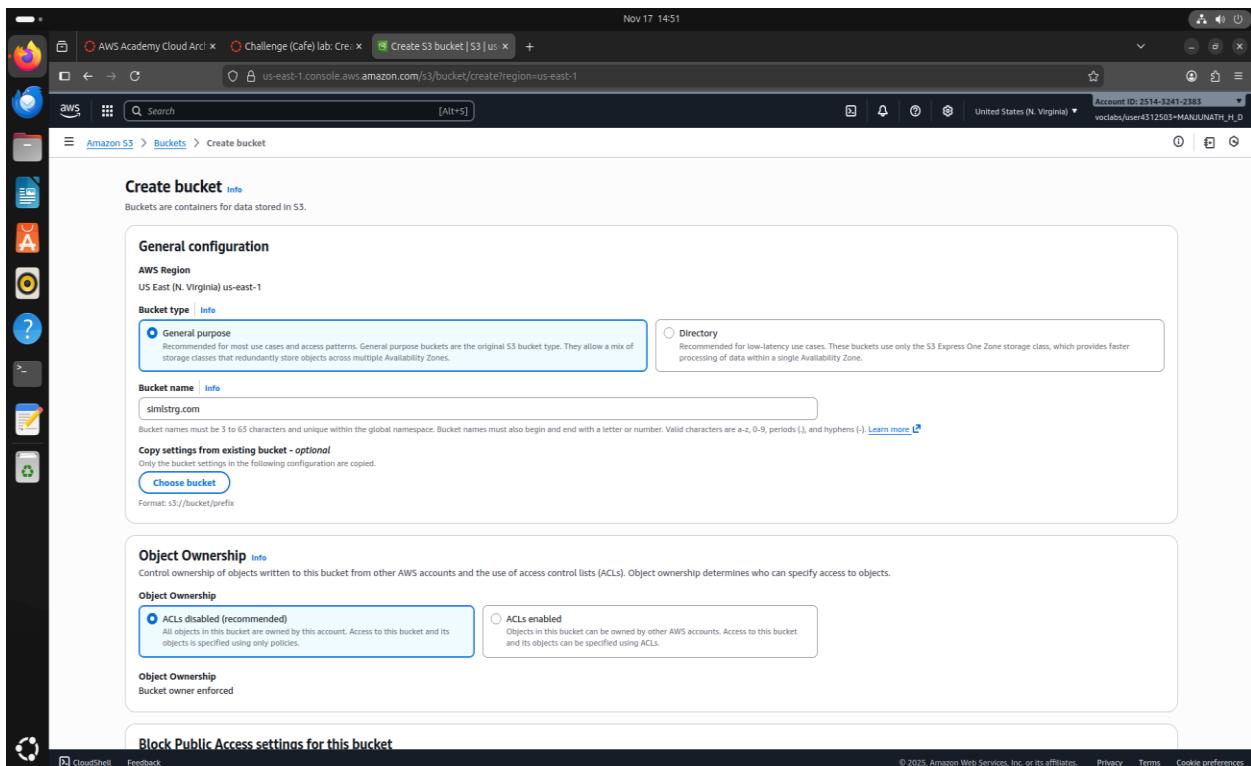
**Tip:** You must clear **Block all public access** and enable **ACLs**.

8. Enable static website hosting on your bucket.

**Tip:** You use the index.html file for your index document.



STEP 3 : Select the S3 bucket in the AWS Management Console.



STEP 4 : I created an S3 bucket called *simlstrg*, and its ACL is not enabled yet.

The screenshot shows the AWS S3 'Create S3 bucket' page. In the 'Block all public access' section, there is a warning message: 'Turning off block all public access might result in this bucket and the objects within becoming public'. Below this, a checkbox is checked with the label 'I acknowledge that the current settings might result in this bucket and the objects within becoming public.' Other sections visible include 'Bucket Versioning' (disabled) and 'Tags - optional' (no tags added).

Step 5: Uncheck Block all public access and check the box for 'I acknowledge that I want to block all public access.'

The screenshot shows the AWS S3 'Buckets' page. A green success message at the top states 'Successfully created bucket "simlstrg.com"'. The 'General purpose buckets' section lists two buckets: 'c174142a45083661120840711w2514324123' and 'simlstrg.com'. An 'Account snapshot' sidebar provides storage usage details, and an 'External access summary' sidebar shows external access findings.

> Now the S3 bucket with the name simlstrg has sucessfully created.

The screenshot shows the AWS CloudShell interface with a browser window open to the AWS S3 console. The user is uploading files to the 'simlstrg.com' bucket. The upload interface displays 10 files and folders with their details:

Name	Folder	Type	Size
styles.css	css/	text/css	541.0 B
Mom-&-Pop-Coffee-Shop.png	Images/	image/png	726.8 KB
Mom-&-Pop.png	Images/	image/png	2.7 MB
Cake-Vitrine.png	Images/	image/png	3.8 MB
Cup-of-Hot-Chocolate.png	Images/	image/png	3.6 MB
Cookies.png	Images/	image/png	1.4 MB
Strawberry-Tarts.png	Images/	image/png	3.4 MB
Coffee-and-Pastries.png	Images/	image/png	3.1 MB
Strawberry-&-Blueberry-Tarts.png	Images/	image/png	2.9 MB
index.html	-	text/html	2.9 KB

The 'Destination' section shows the target bucket as 's3://simlstrg.com'. The 'Destination details' section indicates that bucket settings will impact new objects stored in the specified destination. The 'Permissions' section notes that public access and access to other AWS accounts are granted.

STEP 6 : Upload the index.html file and the folders css and images.

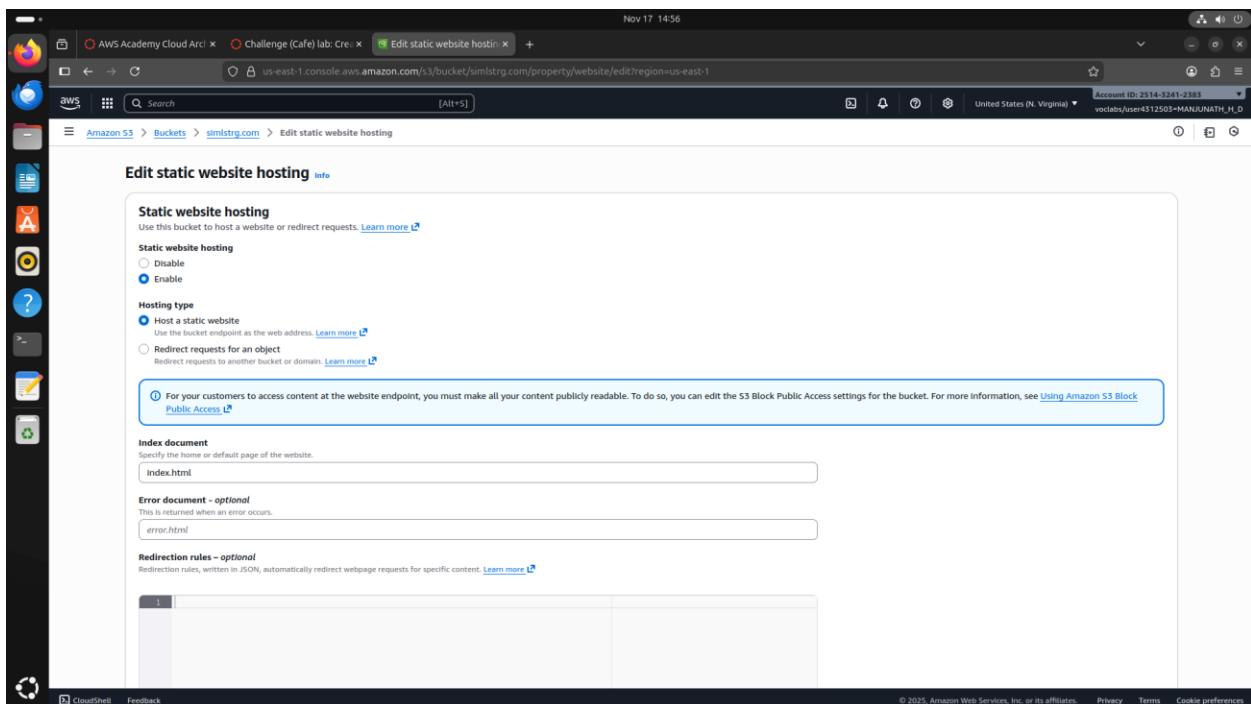
The screenshot shows the AWS S3 console interface. At the top, there's a green success message: "Upload succeeded" with a link to "Files and folders table". Below this, the "Summary" section shows "Succeeded" (10 files, 21.7 MB) and "Failed" (0 files, 0 B). The "Files and folders" tab is selected, displaying a table of uploaded files. The table includes columns for Name, Folder, Type, Size, Status, and Error. All files listed are marked as "Succeeded".

Name	Folder	Type	Size	Status	Error
styles.css	css/	text/css	541.0 B	Succeeded	-
Morn-&-Pop-Coffee-Shop.png	Images/	image/png	726.8 KB	Succeeded	-
Morn-&-Pop.png	Images/	image/png	2.7 MB	Succeeded	-
Cake-Vitrine.png	Images/	image/png	3.0 MB	Succeeded	-
Cup-of-Hot-Chocolate.png	Images/	image/png	3.6 MB	Succeeded	-
Cookies.png	Images/	image/png	1.4 MB	Succeeded	-
Strawberry-Tarts.png	Images/	image/png	3.4 MB	Succeeded	-
Coffee-and-Pastries.png	Images/	image/png	3.1 MB	Succeeded	-
Strawberry-&-Blueberry-Tarts.png	Images/	image/png	2.9 MB	Succeeded	-
index.html	-	text/html	2.9 KB	Succeeded	-

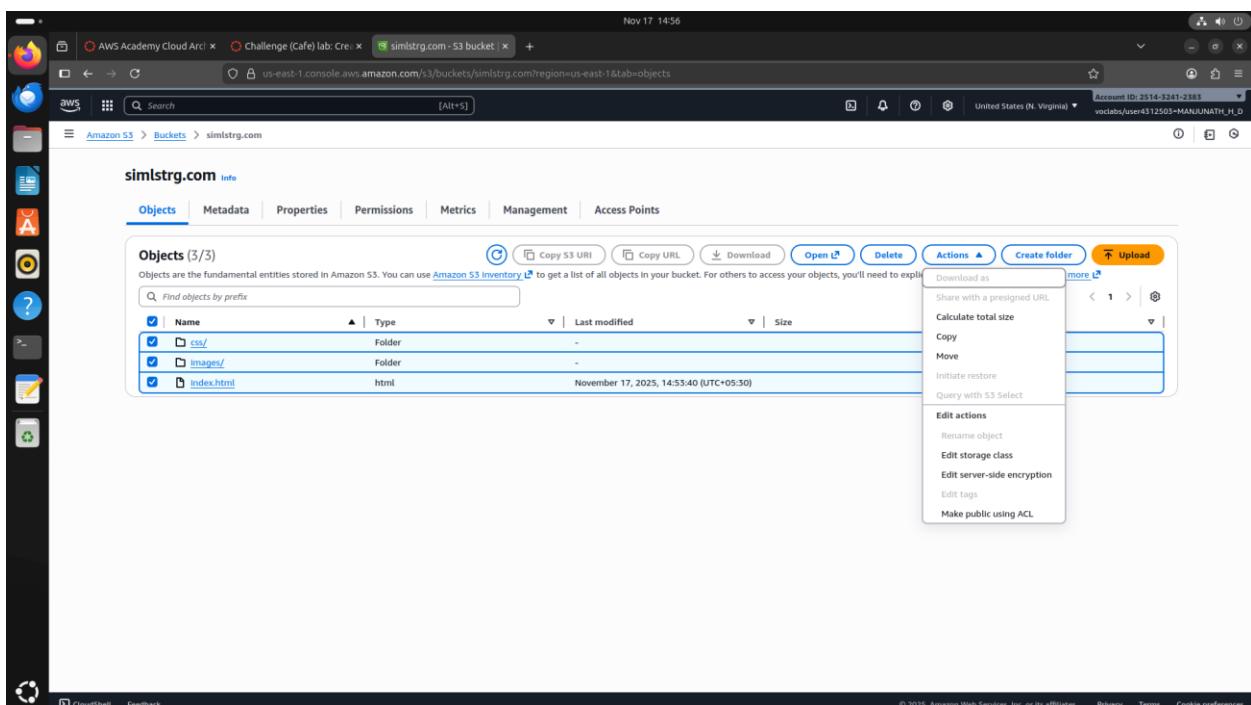
> Uploaded sucessfully

The screenshot shows the "Edit Object Ownership" page in the AWS S3 console. The "Object Ownership" section has two options: "ACLs disabled (recommended)" and "ACLs enabled". The "ACLs enabled" option is selected, indicated by a blue border around its radio button. A note states: "Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs." Below this, a warning message says: "We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing." Another note below says: "Enabling ACLs turns off the bucket owner enforced setting for Object Ownership. Once the bucket owner enforced setting is turned off, access control lists (ACLs) and their associated permissions are restored. Access to objects that you do not own will be based on ACLs and not the bucket policy." A checkbox is checked with the label "I acknowledge that ACLs will be restored." The "Object Ownership" section also includes "Bucket owner preferred" and "Object writer" options. A note at the bottom says: "If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more](#)". At the bottom right are "Cancel" and "Save changes" buttons.

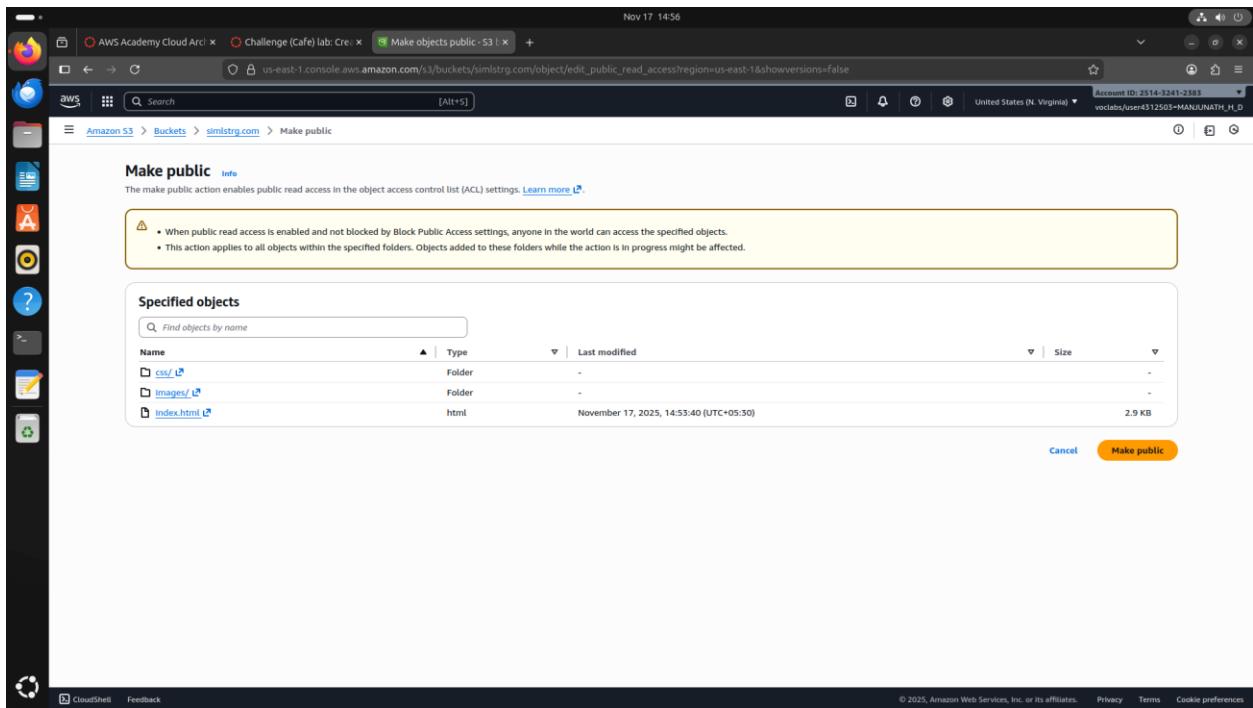
STEP 7 : Go to permissions and enable the ACL and give check mark to the i acknowledge to enable ACL.



STEP 8: Go to properties and enable static website hosting and give the name index.html in the place of index document.



STEP 9 : Go to objects and check all the boxes which contains the uploaded files.



>and go to actions->make public using ACL and then press make public.

The screenshot shows a browser window with the AWS S3 console. The URL is `us-east-1.console.aws.amazon.com/s3/buckets/simlstrg.com/object/edit_public_read_access?region=us-east-1&showversions=false`. The page title is "Make objects public - S3". A green success message at the top says "Successfully edited public access". Below it, a section titled "Make public: status" shows a summary: "Source" is `s3://simlstrg.com`, "Successfully edited public access" is 10 objects (21.7 MB), and "Failed to edit public access" is 0 objects. There are tabs for "Failed to edit public access" (selected) and "Configuration". Under "Failed to edit public access", there is a table header with columns: Name, Folder, Type, Last modified, Size, and Error. The table body below says "No objects failed to edit". The browser's sidebar on the left shows various AWS services like CloudWatch, Lambda, and CloudFront.

> Now the static website is ready for the public access.

Screenshot of the AWS CloudFront console showing the properties of the 'simlstrg.com' bucket. The 'Static website hosting' section is highlighted, showing that it is enabled and configured for 'Bucket hosting'. A note recommends using AWS Amplify Hosting for static website hosting.

**Static website hosting**

We recommend using AWS Amplify Hosting for static website hosting

[Create Amplify app](#)

**Bucket website endpoint**

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)

<http://simlstrg.com.s3-website-us-east-1.amazonaws.com>

Screenshot of the browser displaying the static website for 'Mom & Pop Café'. The page features a header with the cafe's name, two images of pastries and cakes, and a descriptive paragraph about the assortment of treats. Below this are three callout boxes: one for cookies, one for coffee, and one for tarts. At the bottom, there is an 'About Us' section.

**Mom & Pop Café**

Mom & Pop Café offers an assortment of delicious and delectable pastries and coffees that will put a smile on your face. From cookies to croissants, tarts and cakes, each treat is especially prepared to excite your tastebuds and brighten your day!

Pop bakes a rich variety of cookies. Try them all!



Tea  
Coffee  
Latte  
Hot  
Chocolate  
Yes, we have it!

Our tarts are always a customer favorite!

**About Us**

> Now go the properties and scroll down, there you get an link-> then the cafe-static-website will open on a new window.

# Scale and Load Balance Your Architecture

## 1. Create an Amazon Machine Image (AMI) from a running instance

The screenshot shows the AWS EC2 console with the 'AMIs' section selected. A new AMI named 'WebServerAMI' is being created from a running instance. The details page for the AMI shows the following information:

AMI ID	Image type	Platform details	Root device type
ami-02e71aa5520e3382c	machine	Linux/UNIX	EBS
AMI name	Owner account ID	Architecture	Usage operation
WebServerAMI	730335452349	x86_64	RunInstances
Root device name	Status	Source	Virtualization type
/dev/xvda	Pending	730335452349/WebServerAMI	hvm
Boot mode	State reason	Creation date	Kernel ID
uefi-preferred	-	2025-11-17T17:15:22.000Z	-
Description	Product codes	RAM disk ID	Deprecation time
Lab AMI for Web Server	-	-	-
Last launched time	Block devices	Registration protection	Allowed image
Source AMI ID	/dev/xvda1@tnarugn3	Disabled	-
Source AMI Region			

## 2. Create a Load Balancer

The screenshot shows the AWS ELB console with the 'Target groups' section selected. A new target group named 'LabGroup' is being created. The 'Targets' tab shows the following status summary:

Total targets	Healthy	Unhealthy	Unused	Initial	Draining
0	0	0	0	0	0
	Anomalous				

The 'Registered targets' table is currently empty.

Created target group

Screenshot of the AWS CloudWatch Metrics console showing a log stream for a Lambda function named 'HelloWorld'. The log message indicates a successful execution with a duration of 10ms.

## Created Load Balancer

### 3 Create a Launch Template and an Auto Scaling Group

Screenshot of the AWS EC2 Launch Templates page showing a table of existing launch templates. A new launch template named 'LabConfig' is being created, with its details and version information visible.

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By	Managed	Operator
lt-0ab028945a71a202f	LabConfig	1	1	2025-11-17T17:43:34.000Z	arn:aws:sts::730355452349:assumed-role/vocabs/user4512517:SHAZIL_HAMEED	false	-

## Created launch template

The screenshot shows the AWS Auto Scaling Groups page. At the top, there are tabs for 'Launch configurations', 'Launch templates', 'Actions', and 'Create Auto Scaling group'. A search bar is present. The main table has columns for Name, Launch template/configuration, Instances, Status, Desired capacity, Min, Max, Availability Zones, and Creation time. One row is visible for 'Lab Auto Scaling Group'.

## Created auto scaling group

The screenshot shows the AWS Instances page. The left sidebar includes sections for Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, Load Balancing, Load Balancers, Target Groups, and Trust Stores. The main table lists four instances: 'Lab Instance' (running, t2.micro, us-east-1a, 44.211.87.53), 'Web Server 1' (running, t2.micro, us-east-1a, 44.211.90.235), 'Bastion Host' (running, t2.micro, us-east-1a, -), and another 'Lab Instance' (running, t2.micro, us-east-1b, -).

Verified Load Balancing is Working

# VPC Environment Creation and Configuration Lab Report

## Objective

The objective of this lab was to create and configure a custom Amazon Virtual Private Cloud (VPC) named `lab-vpc` in the `us-east-1` region, complete with public and private subnets, and corresponding route tables.

## Lab Environment Setup

- The **AWS Management Console** was used for this lab.
- The lab focused on creating and configuring core VPC components: VPC, Subnets, and Route Tables.

## Task 1: Create the Custom VPC and Subnets

The VPC creation process involved the following steps:

1. **Initiation:** The VPC dashboard was accessed, and the "Create VPC" workflow was started.
2. **Configuration:**
  - The IPv4 CIDR block was set to `10.0.0.0/16`.
  - The VPC was configured to span a single Availability Zone (AZ).
  - One public and one private subnet were specified for creation within the VPC.

## Result

The operation successfully created the VPC named `lab-vpc` (ID: `vpc-04c508510705089f1`). The resulting infrastructure included two subnets (`lab-subnet-public1-us-east-1a` and `lab-subnet-private1-us-east-1a`) and an associated Internet Gateway (IGW).

Nov 18 14:24

Import bookmarks... Account ID: 7828-4076-9740  
vocabs/user4512511=KARTHIK\_UDAYA\_POOJARY

VPC | us-east-1 take screenshot in window

us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#CreateVpc:createMode=vpcWithResources

Search [Alt+S]

VPC > Your VPCs > Create VPC

Tags for all resources in the VPC.  
 Auto-generate  
**Lab**

**IPv4 CIDR block** Info  
Determine the starting IP and the size of your VPC using CIDR notation.  
 65,536 IPs  
CIDR block size must be between /16 and /28.

**IPv6 CIDR block** Info  
 No IPv6 CIDR block  
 Amazon-provided IPv6 CIDR block

**Tenancy** Info  
Default

**Number of Availability Zones (AZs)** Info  
Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.  
 1  2  3

**Customize AZs**

**Number of public subnets** Info  
The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the Internet.  
 0  1  2

**Number of private subnets** Info  
The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.  
 0  1  2

**Customize subnets CIDR blocks**  
Public subnet CIDR block in us-east-1a

**Preview**

**VPC** Show details  
Your AWS virtual network  
lab-vpc

**Subnets (2)**  
Subnets within this VPC  
us-east-1a  
● lab-subnet-public1-us-east-1a  
■ lab-subnet-private1-us-east-1a

**Route tables (2)**  
Route network traffic to resources  
lab-rtb-public  
lab-rtb-private1-us-east-1a

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Nov 18 14:26

Import bookmarks... Account ID: 7828-4076-9740  
vocabs/user4512511=KARTHIK\_UDAYA\_POOJARY

Lab - 2 Build your VPC an... Workbench - Vocareum VPC | us-east-1 +

us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#VpcDetails:vpcId=vpc-04c508510705089f1

Search [Alt+S]

VPC > Your VPCs > vpc-04c508510705089f1 / lab-vpc Actions

**VPC dashboard**

AWS 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
  - NAT gateways
  - Peering connections
  - Route servers
- Security
  - Network ACLs
  - Security groups
- PrivateLink and Lattice
  - Getting started
  - Endpoints
  - Endpoint services
  - Service networks
  - Lattice services

Details Info

VPC ID: vpc-04c508510705089f1  
State: Available  
DNS resolution: Enabled  
Main network ACL: acl-0bf2145ea74053469  
IPv6 CIDR (Network border group): -

Block Public Access: Off  
DHCP option set: dopt-0537e328c5c3634c3  
IPv4 CIDR: 10.0.0.0/16  
Route 53 Resolver DNS Firewall rule groups: -

DNS hostnames: Enabled  
Main route table: rtb-0b544f28d77d5e657  
IPv6 pool: -  
Owner ID: 782840769740

Resource map

Resource map Info

VPC  
Your AWS virtual network  
lab-vpc

Subnets (2)  
Subnets within this VPC  
us-east-1a  
● lab-subnet-public1-us-east-1a  
■ lab-subnet-private1-us-east-1a

Route tables (3)  
Route network traffic to resources  
rtb-0b544f28d77d5e657  
lab-rtb-public  
lab-rtb-private1-us-east-1a

Network Connections (2)  
Connections to other networks  
lab-igw  
lab-nat-public1-us-east-1a

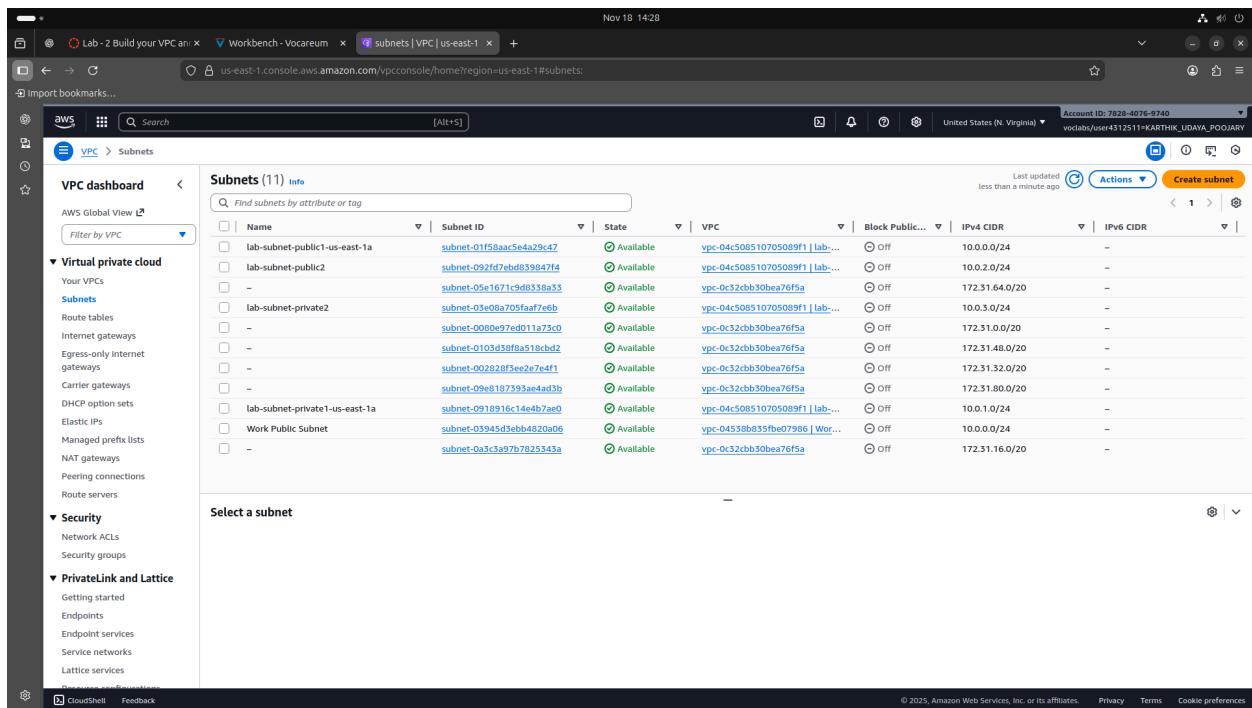
CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## Task 2: Verify Subnet Details

Steps:

1. Navigated to the Subnets section in the VPC console.
2. Verified the creation and status of the new subnets:lab-subnet-public1-us-east-1a and lab-subnet-private1-us-east-1a.

Outcome: The creation of the two specific subnets within the lab-vpc was confirmed, and their availability status was noted as "Available".



Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR
lab-subnet-public1-us-east-1a	subnet-01158aac5e4a29c47	Available	vpc-04c508510705089f1   lab...	Off	10.0.0.0/24	-
lab-subnet-public2	subnet-092fd7ebd039047f4	Available	vpc-04c508510705089f1   lab...	Off	10.0.2.0/24	-
-	subnet-05e1671cd9d338a35	Available	vpc-0c32ccb30bea76f5a	Off	172.31.64.0/20	-
lab-subnet-private2	subnet-03e0ba705faa7e6b	Available	vpc-04c508510705089f1   lab...	Off	10.0.3.0/24	-
-	subnet-0090097e0d011a73c0	Available	vpc-0c32ccb30bea76f5a	Off	172.31.0.0/20	-
-	subnet-0103d3ff8a518cb2	Available	vpc-0c32ccb30bea76f5a	Off	172.31.48.0/20	-
-	subnet-00282ff3ee2e7edf1	Available	vpc-0c32ccb30bea76f5a	Off	172.31.32.0/20	-
-	subnet-09e8107593ea4d5b	Available	vpc-0c32ccb30bea76f5a	Off	172.31.80.0/20	-
lab-subnet-private1-us-east-1a	subnet-0918916c14e4b7ae0	Available	vpc-04c508510705089f1   lab...	Off	10.0.1.0/24	-
Work Public Subnet	subnet-03945d3ebb4820a0	Available	vpc-0453bb035fbef07986   Wor...	Off	10.0.0.0/24	-
-	subnet-0a3c97b7025343a	Available	vpc-0c32ccb30bea76f5a	Off	172.31.16.0/20	-

## Task 3: Configure and Verify Route Tables

Steps:

1. Navigated to the Route tables section.
2. Selected the public route table lab-rtb-public and verified the default route http://0.0.0.0/0 pointed to the Internet Gateway.
3. Selected the private route table ( and verified the default route http://0.0.0.0/0 pointed to the NAT Gateway.

Outcome: Both the public and private route tables were correctly configured and associated with their respective subnets, ensuring proper routing for external and internal traffic.

Nov 18 14:29

Lab - 2 Build your VPC and... Workbench - Vocareum VPC | us-east-1 +

us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#RouteTables:

Import bookmarks...

AWS Global View Filter by VPC

VPC dashboard < VPC > Route tables

You have successfully updated subnet associations for rtb-06c6423ccb10a88b4 / lab-rtb-private1-us-east-1a.

Route tables (1/6) Info Last updated less than a minute ago Actions Create route table

Find route tables by attribute or tag

Name	Route table ID	Explicit subnet assoc...	Edge associations	Main	VPC	Owner ID
-	rtb-0278fc9d96a03f5750	-	-	Yes	vpc-0c32ccb30bea76f5a	782840769740
-	rtb-0f817ba4413f7cf5	-	-	Yes	vpc-0453bb835fbe07986   Wor...	782840769740
Work Public Route Table	rtb-0d21c5ba96a6306ff	subnet-03945d3ebb4820...	-	No	vpc-0453bb835fbe07986   Wor...	782840769740
-	rtb-0b344f28d77d5e657	-	-	Yes	vpc-04c508510705089f1   lab...	782840769740
<b>lab-rtb-public</b>	<b>rtb-0ea5bdcc10b3c8b21</b>	<b>subnet-01f58aac5e4a29c...</b>	-	No	<b>vpc-04c508510705089f1   lab...</b>	<b>782840769740</b>
lab-rtb-private1-us-east-1a	rtb-06c6423ccb10a88b4	2 subnets	-	No	vpc-04c508510705089f1   lab...	782840769740

rtb-0ea5bdcc10b3c8b21 / lab-rtb-public

Details Routes Subnet associations Edge associations Route propagation Tags

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0a2092d7f55624ce	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Nov 18 14:28

Lab - 2 Build your VPC and... Workbench - Vocareum VPC | us-east-1 +

us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#RouteTables:

Import bookmarks...

AWS Global View Filter by VPC

VPC dashboard < VPC > Route tables

Route tables (1/6) Info Last updated 1 minute ago Actions Create route table

Find route tables by attribute or tag

Name	Route table ID	Explicit subnet assoc...	Edge associations	Main	VPC	Owner ID
-	rtb-0278fc9d96a03f5750	-	-	Yes	vpc-0c32ccb30bea76f5a	782840769740
-	rtb-0f817ba4413f7cf5	-	-	Yes	vpc-0453bb835fbe07986   Wor...	782840769740
Work Public Route Table	rtb-0d21c5ba96a6306ff	subnet-03945d3ebb4820...	-	No	vpc-0453bb835fbe07986   Wor...	782840769740
-	rtb-0b344f28d77d5e657	-	-	Yes	vpc-04c508510705089f1   lab...	782840769740
<b>lab-rtb-public</b>	<b>rtb-0ea5bdcc10b3c8b21</b>	<b>subnet-01f58aac5e4a29c...</b>	-	No	<b>vpc-04c508510705089f1   lab...</b>	<b>782840769740</b>
<b>lab-rtb-private1-us-east-1a</b>	<b>rtb-06c6423ccb10a88b4</b>	<b>subnet-0918916c14e4b2...</b>	-	No	<b>vpc-04c508510705089f1   lab...</b>	<b>782840769740</b>

rtb-06c6423ccb10a88b4 / lab-rtb-private1-us-east-1a

Details Routes Subnet associations Edge associations Route propagation Tags

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-0cf72fe24a90e6860	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## **Conclusion**

The lab successfully demonstrated the creation of a custom VPC environment with a CIDR block of 10.0.0.0/16, properly segregated into one public and one private subnet. The correct routing rules were applied to the associated route tables, establishing the foundational network for hosting applications with both public and private tiers on AWS.

# Lab 4: Working with EBS

## Task 1: Create a New EBS Volume

The screenshot shows the AWS EBS Volumes page. A success message at the top states "Successfully created volume vol-089daec5c5ce48031." Below this is a table listing three volumes:

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot ID	Source volume ID	Created
My Volume	vol-089daec5c5ce48031	gp2	1 GiB	100	-	-	-	2025/11/20
	vol-09bf1205113368745	gp3	9 GiB	3000	125	snap-0881c81...	-	2025/11/20
	vol-0ede54c865b16ff94	gp3	8 GiB	3000	125	snap-0881c81...	-	2025/11/20

Below the table, a section titled "Fault tolerance for all volumes in this Region" shows "0 / 2" volumes.

## Task 2: Attach the Volume to an Instance

The screenshot shows the "Attach volume" dialog for volume `vol-089daec5c5ce48031`. The "Basic details" section includes:

- Volume ID: `vol-089daec5c5ce48031` (My Volume)
- Availability Zone: `use1-a22 (us-east-1a)`
- Instance: `i-0c9e45266c84198ce` (Lab) (running)
- Device name: `/dev/sdf`

A note at the bottom states: "Newer Linux kernels may rename your devices to `/dev/xvdf` through `/dev/xvdः` internally, even when the device name entered here (and shown in the details) is `/dev/sdf` through `/dev/sdp`."

At the bottom right are "Cancel" and "Attach volume" buttons.

## Task 3: Connect to Your Amazon EC2 Instance

The screenshot shows the AWS EC2 Instance Connect interface. At the top, it displays the instance ID: i-0c9e45266c84198ce (Lab). Under "Connection type", the "Public IPv4 address" option is selected, showing the IP 34.227.13.110. The "Username" field is set to "ec2-user". A note at the bottom states: "Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username." At the bottom right are "Cancel" and "Connect" buttons.

## Task 4: Create and Configure Your File System

The screenshot shows an AWS CloudShell terminal window. The user runs the command `df -h` to show disk usage, which includes entries for devtmpfs, tmpfs, /dev/xvda1, /tmp, /boot/efi, /run/user/1000, and /dev/xvda128. The user then runs `sudo mkfs -t ext3 /dev/sdf` to create a new ext3 filesystem on the device. The process involves creating filegroup tables, writing inode tables, creating a journal, and writing superblocks. Finally, the user runs `sudo mkdir /mnt/data-store` to create a directory for the new filesystem.

```
'  #  
~\ #####      Amazon Linux 2023  
~~ \#####!  
~~ \###!  
~~ \#/ ____ https://aws.amazon.com/linux/amazon-linux-2023  
~~ V-^ r->  
~~ .- /  
~~ .- /  
~/m/  
[ec2-user@ip-10-1-11-236 ~]$ df -h  
Filesystem      Size  Used Avail Use% Mounted on  
devtmpfs        4.0M   0    4.0M  0% /dev  
tmpfs          475M   0   475M  0% /dev/shm  
tmpfs          190M  452K 190M  1% /run  
/dev/xvda1      8.0G  1.6G  6.4G  21% /  
tmpfs          475M   0   475M  0% /tmp  
/dev/xvda128    10M  1.3M  8.7M  13% /boot/efi  
tmpfs          95M   0   95M  0% /run/user/1000  
[ec2-user@ip-10-1-11-236 ~]$ sudo mkfs -t ext3 /dev/sdf  
mke2fs 1.46.5 (30-Dec-2021)  
Creating filesystem with 262144 4k blocks and 65536 inodes  
Filesystem UUID: b719f563-4cda-43da-a549-3d6faea2a6a2  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (8192 blocks): done  
Writing superblocks and filesystem accounting information: done  
  
[ec2-user@ip-10-1-11-236 ~]$ sudo mkdir /mnt/data-store
```

## Task 5: Create an Amazon EBS Snapshot

The screenshot shows two screenshots of the AWS EC2 console. The top screenshot is the 'Create snapshot' dialog for a volume named 'vol-089daec5c5ce48031'. It includes fields for 'Description' (with a placeholder 'Add a description for your snapshot'), 'Encryption Info' (set to 'Not encrypted'), and 'Tags' (with one tag 'Name: My Snapshot'). The bottom screenshot shows the 'Snapshots' list, which contains a single entry: 'My Snapshot' (Snapshot ID: snap-0c1671a8c83d1d4a0, Full snapshot size: -, Volume size: 1 GiB, Description: -, Storage tier: Standard, Snapshot status: Pending). The left sidebar shows navigation links for Launch templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, and Elastic Block Store (selected), with sub-links for Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces.

**Screenshot 1: Snapshot details**

Description  
Add a description for your snapshot  
255 characters maximum.

Encryption Info  
Not encrypted

**Tags** Info  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional
Name	My Snapshot

Add tag  
You can add 49 more tags.

**Create snapshot**

**Screenshot 2: Snapshots (1) Info**

Name	Snapshot ID	Full snapshot size	Volume size	Description	Storage tier	Snapshot status
My Snapshot	snap-0c1671a8c83d1d4a0	-	1 GiB	-	Standard	Pending

Last updated less than a minute ago

Recycle Bin Actions Create snapshot

Select a snapshot above.

The screenshot shows the AWS Cloud9 interface. On the left, there's a sidebar with navigation links like Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces. The 'Snapshots' link under 'Elastic Block Store' is highlighted.

The main content area displays a table titled 'Snapshots (1/1)'. It shows one entry: 'My Snapshot' with Snapshot ID 'snap-0c1671a8c83d1d4a0', Full snapshot size '53 MiB', Volume size '1 GiB', and a status of 'Completed'. A context menu is open over this row, with 'Create volume from snapshot' highlighted.

At the bottom of the page, there's a footer with links for Privacy, Terms, and Cookie preferences.

## Create a Volume Using Your Snapshot

This screenshot shows the same AWS Cloud9 interface after creating a volume from the snapshot. A green success message at the top says 'Successfully created volume vol-0f4415c04689e9d79.' Below it, the 'Snapshots (1)' table remains the same as in the previous screenshot.

At the bottom of the page, there's a footer with links for Privacy, Terms, and Cookie preferences.

## Task 6: Restore the Amazon EBS Snapshot

### Create a Volume Using Your Snapshot

The screenshot shows the AWS Volumes page with a list of volumes. A context menu is open over the 'Restored Volu...' volume, specifically over the 'Actions' dropdown. The 'Attach volume' option is highlighted. The volume details are shown below:

Volume ID	Size	Type	Status check
vol-0f4415c04689e9d79 (Restored Volume)	1 GiB	gp2	Okay
AWS Compute Optimizer finding	Volume state	IOPS	Throughput
Opt-in to AWS Compute Optimizer for recommendations.   Learn more	Available	100	-
Fast snapshot restored	Availability Zone	Created	Multi-Attach enabled
N/A	us-east-1a	2025-12-16 10:51:30 UTC	N/A

### Attach the Restored Volume to Your EC2 Instance

The screenshot shows the 'Attach volume' dialog for an EC2 instance. The 'Basic details' section is visible, showing the selected volume and instance.

**Basic details**

Volume ID: vol-0f4415c04689e9d79 (Restored Volume)

Availability Zone: us-east-1a

Instance: i-0c9e45266c84198ce (Lab) (running)

Device name: /dev/sdg

Only instances in the same Availability Zone as the selected volume are displayed.

Recommended device names for Linux: /dev/xvda for root volume. /dev/sdf-f-p for data volumes.

Info: Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

Buttons: Cancel, Attach volume

## Mount the Restored Volume

Superblock backups stored on blocks:  
32768, 98304, 163840, 229376  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (8192 blocks): done  
Writing superblocks and filesystem accounting information: done  
  
[ec2-user@ip-10-1-11-236 ~]\$ sudo mkdir /mnt/data-store  
[ec2-user@ip-10-1-11-236 ~]\$ sudo mount /dev/sdf /mnt/data-store  
[ec2-user@ip-10-1-11-236 ~]\$ echo "/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee -a /etc/fstab  
/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2  
[ec2-user@ip-10-1-11-236 ~]\$ cat /etc/fstab  
#  
UUID=9a57ac2f-bfe7-4e29-bf89-56caddc22c97 / xfs defaults,noatime 1 1  
UUID=9682-52BE /boot/efi vfat defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2  
/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2  
[ec2-user@ip-10-1-11-236 ~]\$ df -h  
Filesystem Size Used Avail Use% Mounted on  
devtmpfs 4.0M 0 4.0M 0% /dev  
tmpfs 475M 0 475M 0% /dev/shm  
tmpfs 190M 448K 190M 1% /run  
/dev/xvda1 8.0G 1.6G 6.4G 21% /  
tmpfs 475M 0 475M 0% /tmp  
/dev/xvda128 10M 1.3M 8.7M 13% /boot/efi  
tmpfs 95M 0 95M 0% /run/user/1000  
/dev/xvdf 975M 60K 924M 1% /mnt/data-store  
[ec2-user@ip-10-1-11-236 ~]\$ sudo sh -c "echo some text has been written > /mnt/data-store/file.txt"  
[ec2-user@ip-10-1-11-236 ~]\$ cat /mnt/data-store/file.txt  
some text has been written  
[ec2-user@ip-10-1-11-236 ~]\$ sudo rm /mnt/data-store/file.txt  
[ec2-user@ip-10-1-11-236 ~]\$ ls /mnt/data-store/  
  
CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates.

Writing superblocks and filesystem accounting information: done  
  
[ec2-user@ip-10-1-11-236 ~]\$ sudo mkdir /mnt/data-store  
[ec2-user@ip-10-1-11-236 ~]\$ sudo mount /dev/sdf /mnt/data-store  
[ec2-user@ip-10-1-11-236 ~]\$ echo "/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2" | sudo tee -a /etc/fstab  
/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2  
[ec2-user@ip-10-1-11-236 ~]\$ cat /etc/fstab  
#  
UUID=9a57ac2f-bfe7-4e29-bf89-56caddc22c97 / xfs defaults,noatime 1 1  
UUID=9682-52BE /boot/efi vfat defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2  
/dev/sdf /mnt/data-store ext3 defaults,noatime 1 2  
[ec2-user@ip-10-1-11-236 ~]\$ df -h  
Filesystem Size Used Avail Use% Mounted on  
devtmpfs 4.0M 0 4.0M 0% /dev  
tmpfs 475M 0 475M 0% /dev/shm  
tmpfs 190M 448K 190M 1% /run  
/dev/xvda1 8.0G 1.6G 6.4G 21% /  
tmpfs 475M 0 475M 0% /tmp  
/dev/xvda128 10M 1.3M 8.7M 13% /boot/efi  
tmpfs 95M 0 95M 0% /run/user/1000  
/dev/xvdf 975M 60K 924M 1% /mnt/data-store  
[ec2-user@ip-10-1-11-236 ~]\$ sudo sh -c "echo some text has been written > /mnt/data-store/file.txt"  
[ec2-user@ip-10-1-11-236 ~]\$ cat /mnt/data-store/file.txt  
some text has been written  
[ec2-user@ip-10-1-11-236 ~]\$ sudo rm /mnt/data-store/file.txt  
[ec2-user@ip-10-1-11-236 ~]\$ ls /mnt/data-store/  
lost+found  
[ec2-user@ip-10-1-11-236 ~]\$ sudo mkdir /mnt/data-store2  
[ec2-user@ip-10-1-11-236 ~]\$ sudo mount /dev/sdg /mnt/data-store2  
[ec2-user@ip-10-1-11-236 ~]\$ ls /mnt/data-store2/  
file.txt lost+found  
[ec2-user@ip-10-1-11-236 ~]\$ ss  
  
CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates.

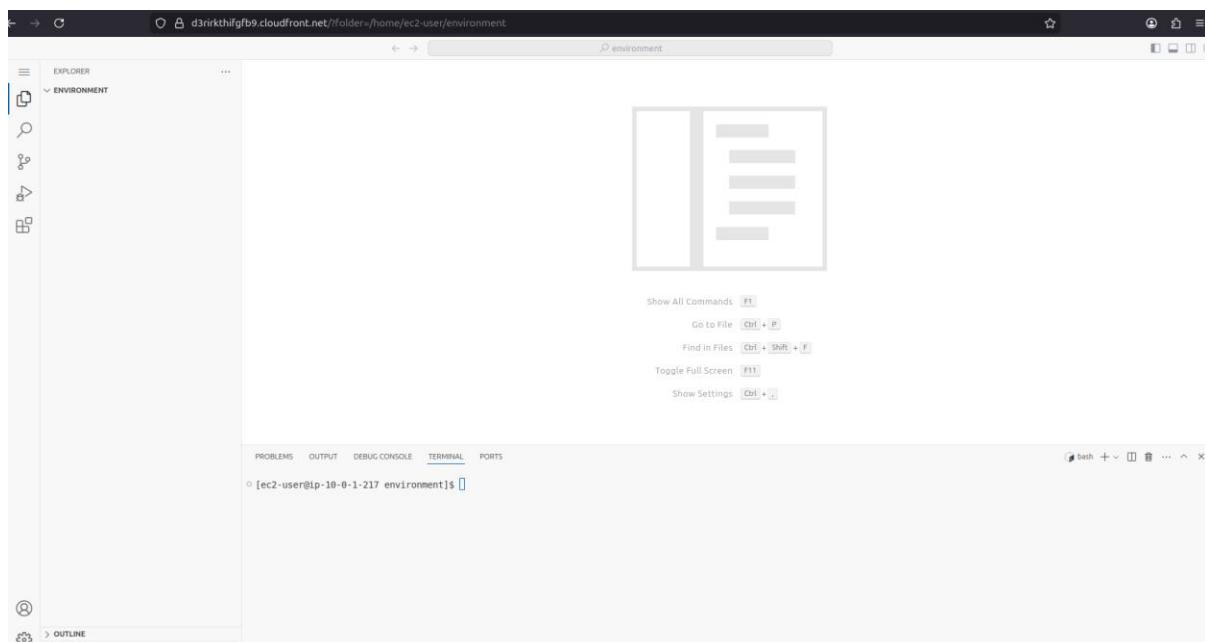
# Breaking a Monolithic Node.js Application into Microservices

## Objective:

- Migrate a monolithic Node.js application to run in a Docker container.
- Refactor a Node.js application from a monolithic design to a microservices architecture.
- Deploy a containerized Node.js microservices application to Amazon ECS

**Task 1:** Connecting to the IDE and preparing the development environment on the EC2 instance.

1. At top of the instructions choose AWS Details
2. Copy values from the table
  - a.LabIDEURL
  - b.LabIDEPASSWORD
3. In a new browser tab, paste the LabIDEURL On the prompt window Welcome to code-server: Enter the value for LabIDEPASSWORD.
4. Choose submit



5. In the bottom pane of the IDE, in the terminal tab, run the following command:

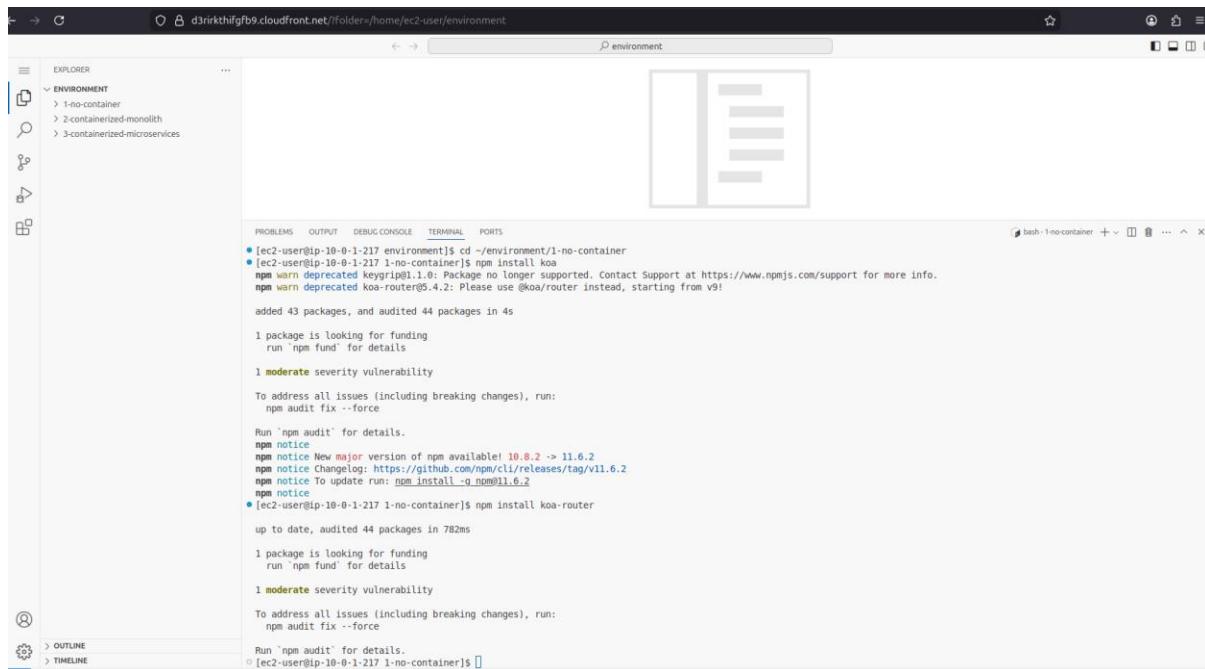
```
curl -s https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-3-113230/19-lab-mod14-guided-ECS/s3/lab-files-ms-node-js.tar.gz | tar -zv
```

## Task 2: Running the application on a basic Node.js server

### Step 1: Installing the required Node.js modules

1.In the Lab IDE terminal, to install the koa and koa-router modules, enter the following commands

- `cd ~/environment/1-no-container`
- `npm install koa`
- `npm install koa-router`



```
[ec2-user@ip-10-0-1-217 environment]$ cd ~/environment/1-no-container
[ec2-user@ip-10-0-1-217 1-no-container]$ npm install koa
npm warn deprecated keyv@0.1.0: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm warn deprecated koa-router@5.4.2: Please use @koa/router instead, starting from v9!
added 43 packages, and audited 44 packages in 4s
1 package is looking for funding
  run 'npm fund' for details
1 moderate severity vulnerability

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
npm notice
npm notice New major version of npm available! 10.8.2 -> 11.6.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.6.2
npm notice To update run: npm install -g npm@11.6.2
npm notice
npm notice
[ec2-user@ip-10-0-1-217 1-no-container]$ npm install koa-router
up to date, audited 44 packages in 782ms
1 package is looking for funding
  run 'npm fund' for details
1 moderate severity vulnerability

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
[ec2-user@ip-10-0-1-217 1-no-container]$
```

### Step 2: Running the application

1.In the terminal tab, to start the Node.js server and the application, enter the following command:

- `Npm start`

The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER** pane on the left showing a tree view of "ENVIRONMENT" with items: > 1-no-container, > 2-containerized-monolith, > 3-containerized-microservices.
- TERMINAL** pane on the right showing the output of an npm install command for a Koa application. It includes npm audit results and a start command output.
- PROBLEMS** pane at the bottom showing 1 moderate severity vulnerability.
- OUTPUT** and **DEBUG CONSOLE** tabs are also visible in the terminal header.

```

db.json index.js package.json server.js
• [ec2-user@ip-10-0-1-204 1-no-container]$ npm install koa

npm install koa-router
npm warn deprecated keygrip@1.1.0: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm warn deprecated koa-router@5.4.2: Please use @koa/router instead, starting from v9!

added 43 packages, audited 44 packages in 4s

1 package is looking for funding
  run 'npm fund' for details

1 moderate severity vulnerability

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
npm notice
npm notice New major version of npm available! 10.0.2 -> 11.6.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.6.2
npm notice To update run: npm install -g npm@11.6.2
npm notice

up to date, audited 44 packages in 821ms

1 package is looking for funding
  run 'npm fund' for details

1 moderate severity vulnerability

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.
○ [ec2-user@ip-10-0-1-204 1-no-container]$ npm start

> start
> node index.js

Leader 32663 is running
Worker 32671 started
Worker 32679 started

```

2. In the bottom pane, choose (+), and choose New Terminal to open a new terminal tab.

3. In the right terminal tab, to retrieve the /api/users resource, enter the following command:

```
curl localhost:3000/api/users
```

4. Retrieve information about 4th user: In the right terminal tab, enter the following command:

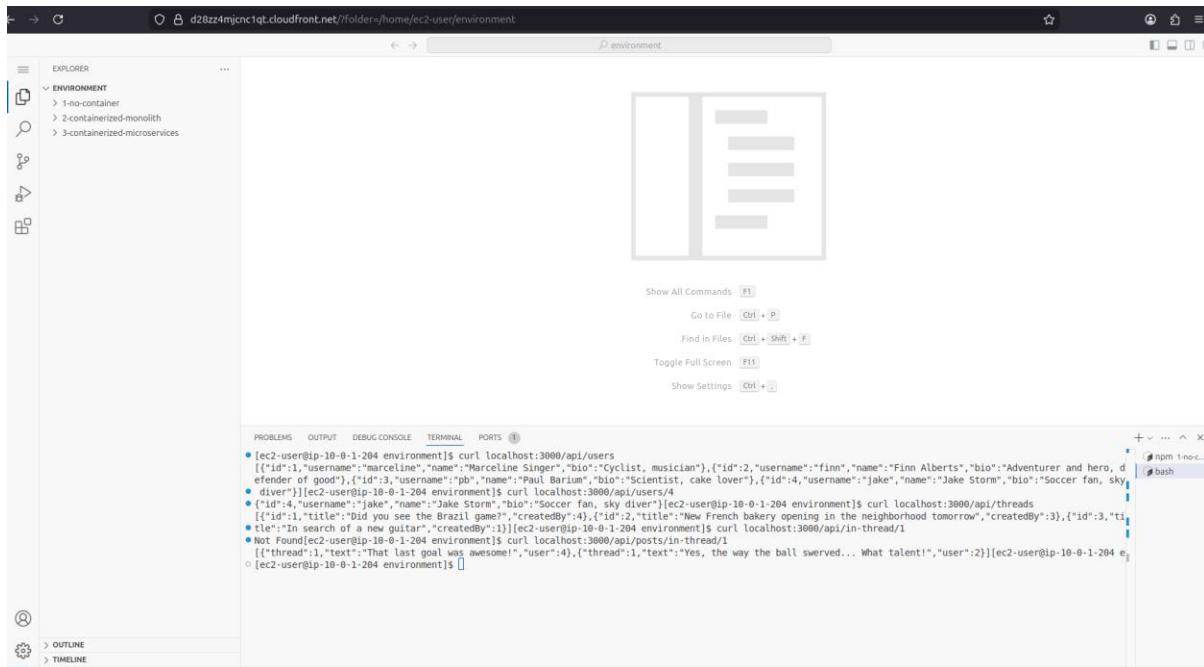
```
curl localhost:3000/api/users/4
```

5. Retrieve threads: In the right terminal tab, enter the following command:

```
curl localhost:3000/api/threads
```

6. Retrieve thread 1: In the right terminal tab, enter the following command:

```
curl localhost:3000/api/posts/in-thread/1
```



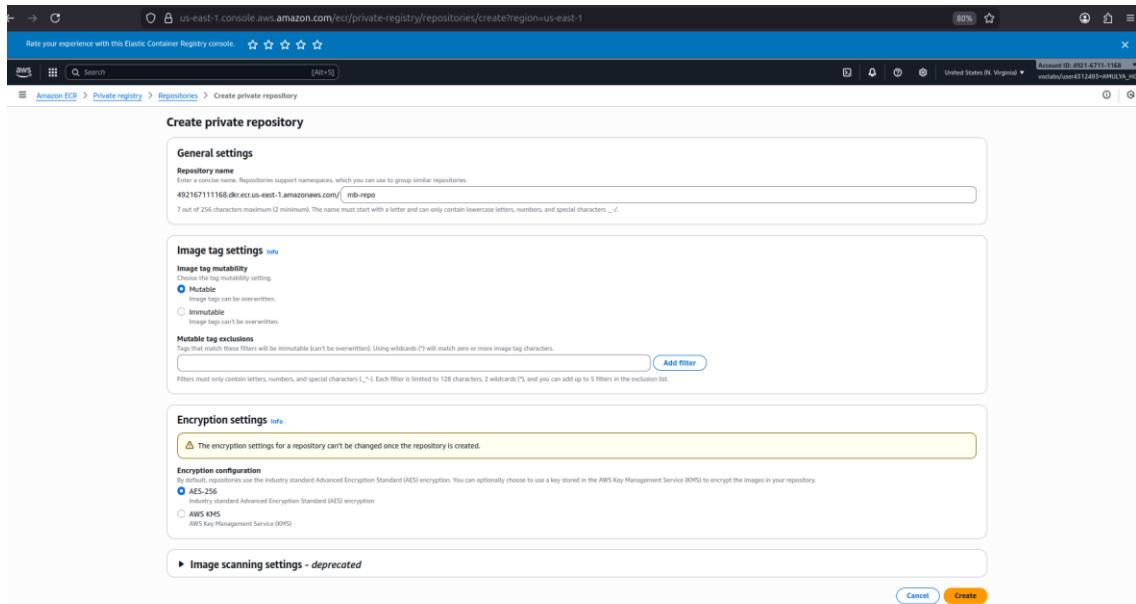
7. Now you stop the Node.js server.

8. In the left terminal tab, press Ctrl+C to shut down the server process.

## Task 3: Containerizing the monolith for Amazon ECS

### Step 1: Provisioning the repository

- On the AWS Management Console, in the search box, enter and choose Elastic Container Registry.



- In the Create a repository section, choose Create

3. Under Create Private repositories
4. For the Repository name enter mb-repo
5. Choose Create.

## Step 2: Building and pushing the Docker image

1. From the Private repositories listed, choose the mb-repo you created and then choose View push commands.
2. A pop-up window titled Push commands for mb-repo opens.

The screenshot shows the AWS Lambda function configuration interface. In the left sidebar, under 'FUNCTION CONFIGURATION', there is a section titled 'Push commands'. It contains a table with two rows:

Command	Description
<code>aws ecr get-login-password --region us-east-1   docker login --username AWS --password-stdin NNNNNNNNNNNN.dkr.ecr.us-east-1.amazonaws.com</code>	Get ECR password and log in to the specified repository.

3. Make sure that the macOS/Linux tab is selected.
4. In the pop-up window, for the first command, choose the Copy icon to copy it to the clipboard.

```
$ (aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin NNNNNNNNNNNN.dkr.ecr.us-east-1.amazonaws.com)
```

5. In the terminal tab, to change the directory to the 2-containerized-monolith folder, enter the following command:

```
cd ~/environment/2-containerized-monolith
```

6. Switch to the Elastic Container Registry browser tab.
7. Copy the following commands:

a. `docker build -t mb-repo`

- b. docker tag mb-repo:latest NNNNNNNNNNNN.dkr.ecr.us-east-1.amazonaws.com/mb-repo:latest
- c. docker push NNNNNNNNNNNN.dkr.ecr.us-east-1.amazonaws.com/mb-repo:latest
8. In the Repositories list, choose mb-repo
  9. In the Images list, next to Copy URI, choose the Copy icon. Paste the value in a text editor.

The screenshot shows the AWS ECR console interface. On the left, there's a sidebar with navigation links for 'Amazon Elastic Container Registry', 'Private registry', 'Public registry', and various documentation links. The main area is titled 'Images (1)' and lists a single image entry:

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Last recorded pull time
latest	Image	November 17, 2025, 19:56:11 (UTC+05:5)	20.46	<a href="#">Copy URI</a>	sha256:729740cc9887d7...	-

## Task 4: Deploying the monolith to Amazon ECS

### Step 1: Creating an Amazon ECS cluster

1. On the AWS Management Console, in the search box, enter and select Elastic Container Services
2. From the left menu, choose Clusters.
3. Choose Create cluster, and configure the following options:
4. In the Cluster configuration section, for the Cluster name, enter mb-ecs-cluster.
5. Uncheck *AWS Fargate (serverless)*
6. Check Amazon EC2 instances,
7. Leave the Auto Scaling group (ASG) settings at defaults.

8. Provisioning model: On-demand.
9. For EC2 instance type, choose t2.medium.
10. For Container instance Amazon Machine Image (AMI), choose Amazon Linux 2023.
11. EC2 instance role: *Create new role*.
12. For Desired capacity, for Minimum 2 and Maximum, enter 6
13. VPC: choose IDE VPC.
14. Subnets: Reconfirm that, all public subnets chosen.
15. Security group: Choose Use an existing security group.
16. Security group name dropdown list, select the security group that has ECSSG in the name.
17. Clear the default security group
18. Choose Create.
19. Choose mb-ecs-cluster

The screenshot shows the AWS CloudShell interface with the URL <https://us-east-1.console.aws.amazon.com/ecs/v2/create-cluster?region=us-east-1>. The page is titled 'Create cluster' under the 'Amazon Elastic Container Service' section. The configuration steps shown are:

- Subnets:** Three subnets are selected: 'subnet-096420f949b89c01c' (IDE Public Subnet Three, us-east-1c, 10.0.3.0/24), 'subnet-08dca405ca652f6b0' (IDE Public Subnet One, us-east-1a, 10.0.1.0/24), and 'subnet-00bd5d713be83dff8' (IDE Public Subnet Two, us-east-1a, 10.0.2.0/24).
- Security group:** 'Use an existing security group' is selected, and the security group 'sg-07f1fd51a2e1c7eb6' (c174142a450841412162d411w492167111168-ECSSG-4ne6wbhB2bnX) is chosen.
- Auto-assigning public IP:** 'Use subnet setting' is selected.
- Monitoring - optional:** Configure observability, encryption, and logging options to maintain compliance and operational visibility of your container environment.
- Encryption - optional:** Choose the KMS keys used by tasks running in this cluster to encrypt your storage.
- Tags - optional:** Tags help you to identify and organize your clusters.

At the bottom right, there are 'Cancel' and 'Create' buttons. The status bar at the bottom indicates 'CloudShell' and 'Feedback'.

20. Choose infrastructure tab
21. The Container instances pane shows that two EC2 instances for the cluster were created.

The screenshot shows the AWS ECS console interface. On the left, there's a navigation sidebar for the Amazon Elastic Container Service. The main content area is titled 'mb-eecs-cluster ASG' and shows the 'Infrastructure' tab selected. It provides an overview of the cluster, including its ARN (arn:aws:ecs:us-east-1:492167111168:cluster/mb-eecs-cluster), status (Active), and CloudWatch monitoring settings. It also lists registered container instances (2). Below this, the 'Services' section shows one service named 'Training' with an 'Active' status. The 'Tasks' section shows pending and running tasks. The 'Capacity providers' section lists three providers: Fargate, Fargate Spot, and Infra-ECS-Cluster. The 'Container instances' section lists two instances: one t2.medium and one t2.large, both active and running. The bottom of the page includes standard AWS navigation links like 'Feedback' and 'Tell us what you think'.

## Step 2: Creating a task definition for the application container image

1. On the Amazon ECS console, from the left menu, choose Task definitions.
2. Choose Create new task definition
3. Task definition configuration section, for Task definition family, enter mb-task
4. In the Infrastructure requirements, select Amazon EC2 instances, and clear *AWS Fargate*.
5. For the Task size, enter CPU: .5vCPU, Memory: 1GB

The screenshot shows the 'Create new task definition' wizard. Step 1: Task definition configuration, where the task definition family is set to 'mb-task'. Step 2: Infrastructure requirements, where the launch type is set to 'AWS Fargate'. Other options shown include 'Managed instances - new' and 'Amazon EC2 instances'. Step 3: Task size, where CPU is set to '.5 vCPU' and Memory to '1 GB'. Step 4: Task roles - conditional, which is currently collapsed. The bottom of the page includes standard AWS navigation links like 'Feedback' and 'Tell us what you think'.

6. For Container details, for Name, enter mb-container

7. For Image URI, paste the URI of the user's container image that you copied to a text editor earlier.

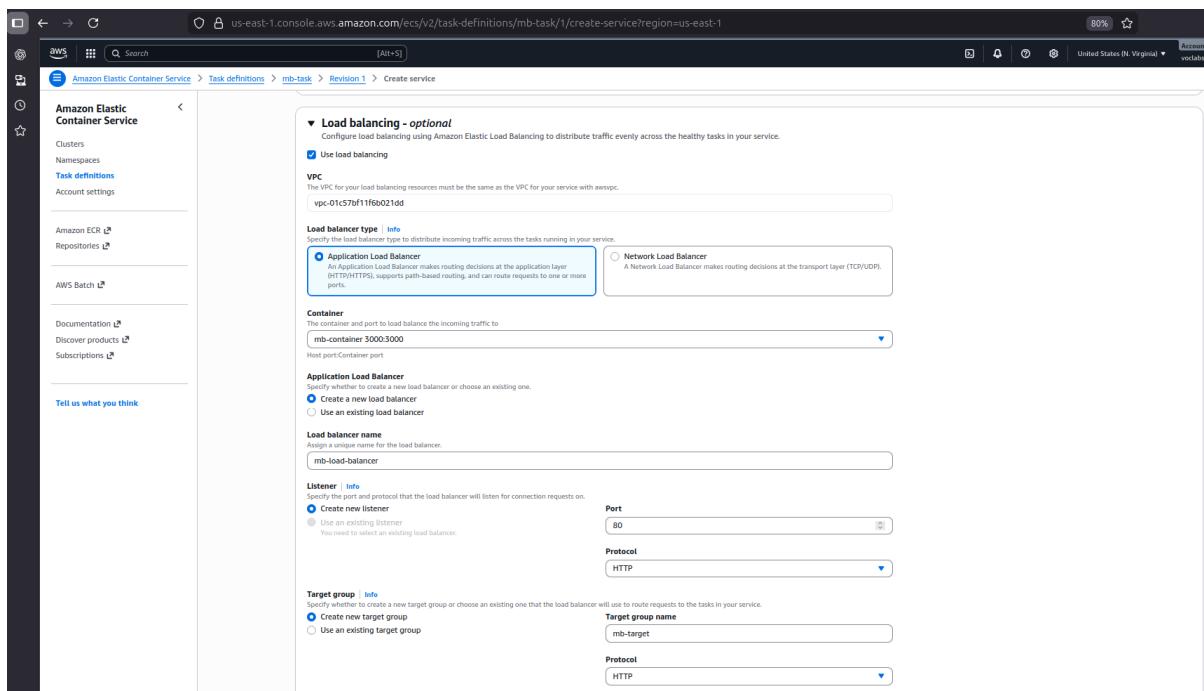
8. For Port mappings, for Container port, enter 3000

9. Choose Create.

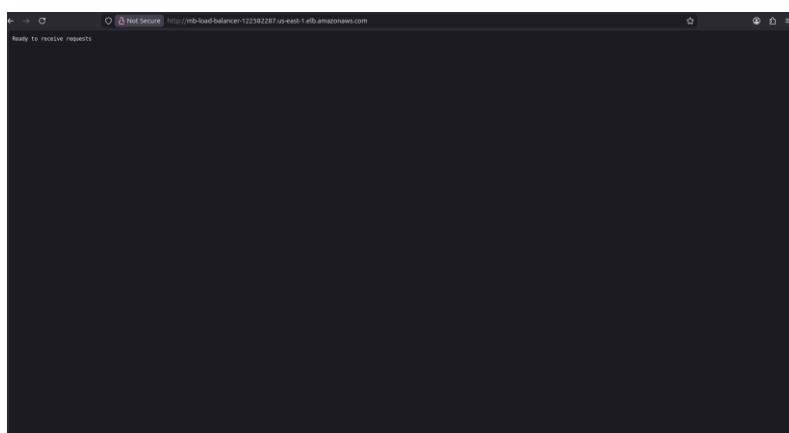
### Step 3: Creating and deploying the service

1. Select mb-task, choose Deploy then choose Create Service.
2. For Compute options, choose Launch type.
3. For Launch type, choose EC2.
4. For Service name choose mb-ecs-service.
5. For the Security group, choose Use an existing security group.

4. From the Security group name dropdown list, select the security group that has ECSSG in the name
5. Clear the default security group.
6. Check Use load balancing.
7. For Load balancer type, choose Application Load Balancer.
8. For Application Load Balancer, choose Create a new load balancer.
9. For Load balancer name, enter mb-load-balancer
10. For Target group name, enter mb-target.
11. Choose Create



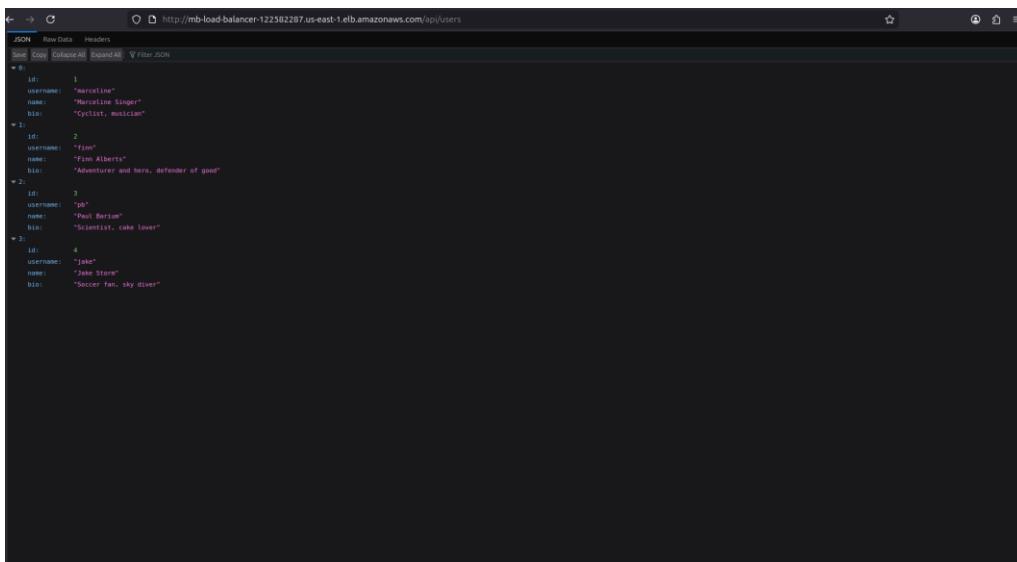
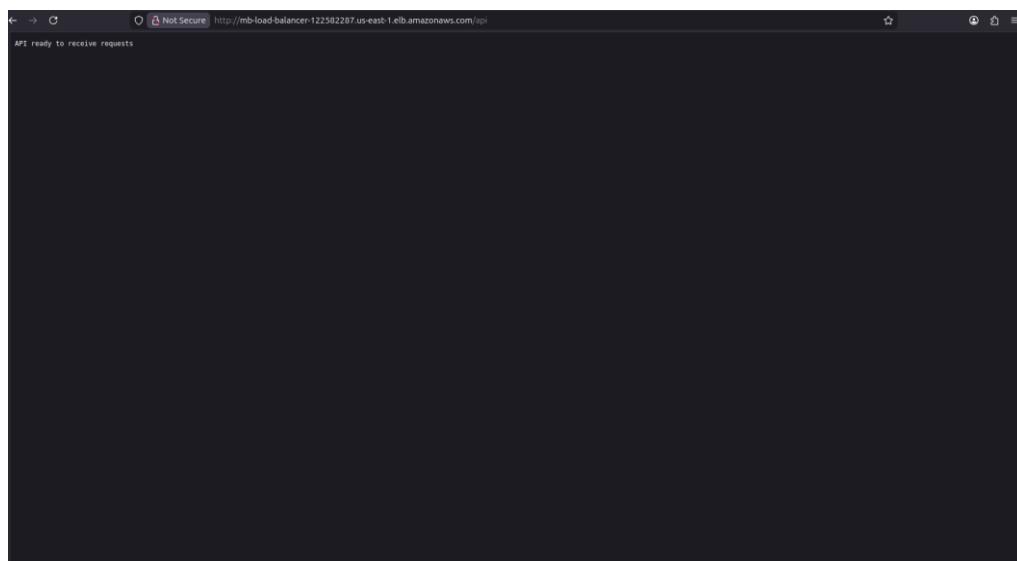
12. From the services list, choose mb-ecs-service
13. It should show a Status of *Active* with one Task that's Running
14. In the Load balancer health section choose load balancer.
15. Copy the DNS name for the load balancer, and paste it into a new browser tab

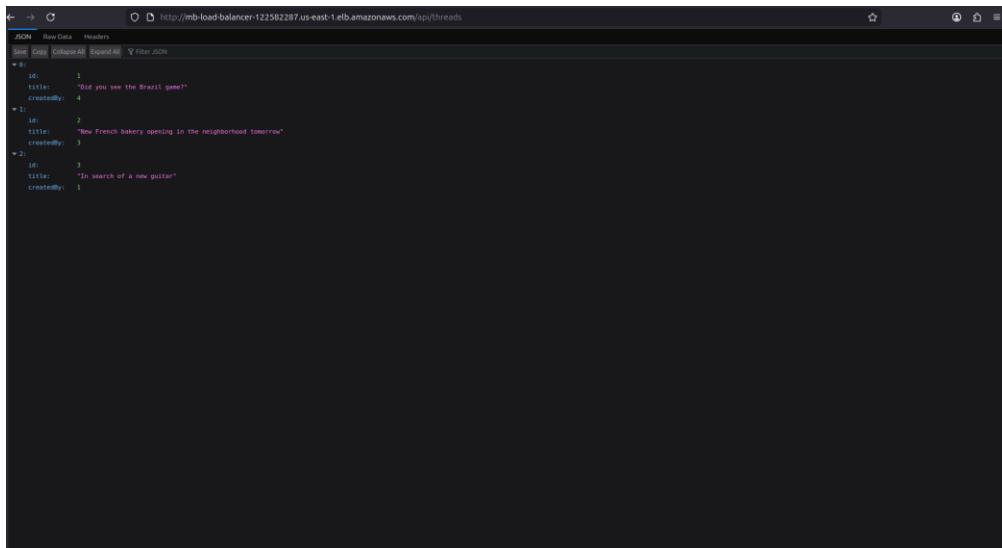


**Step 4:** Testing the containerized monolith

1. You need the DNS name of the load balancer that you used in the previous steps.
2. Open a new browser tab, paste the DNS name into the address field, and press Enter.
3. Enter the following addresses in the browser tab and examine the results. For each address, replace *DNS name* with the DNS name from the previous steps

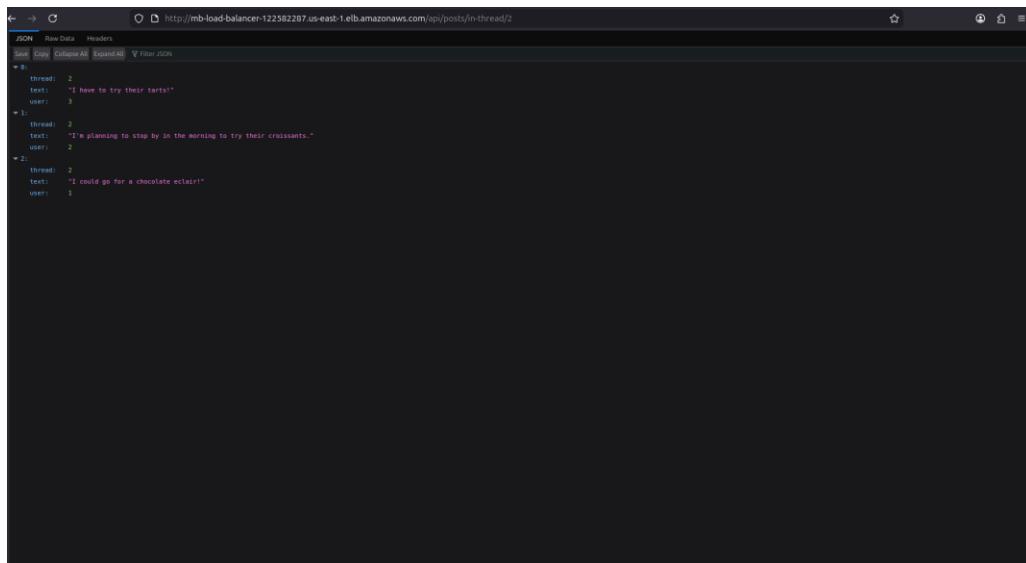
- *DNS name/api*
- *DNS name/api/users*
- *DNS name/api/threads*
- *DNS name/api/posts/in-thread/2*





A screenshot of a web browser displaying JSON data. The URL is <http://mb-load-balancer-122502287.us-east-1.elb.amazonaws.com/api/threads>. The JSON response contains three objects, each representing a thread:

```
[{"id": 1, "title": "Did you see the Brazil game?", "createdBy": 4}, {"id": 2, "title": "New French bakery opening in the neighborhood tomorrow", "createdBy": 3}, {"id": 3, "title": "In search of a new guitar", "createdBy": 1}]
```



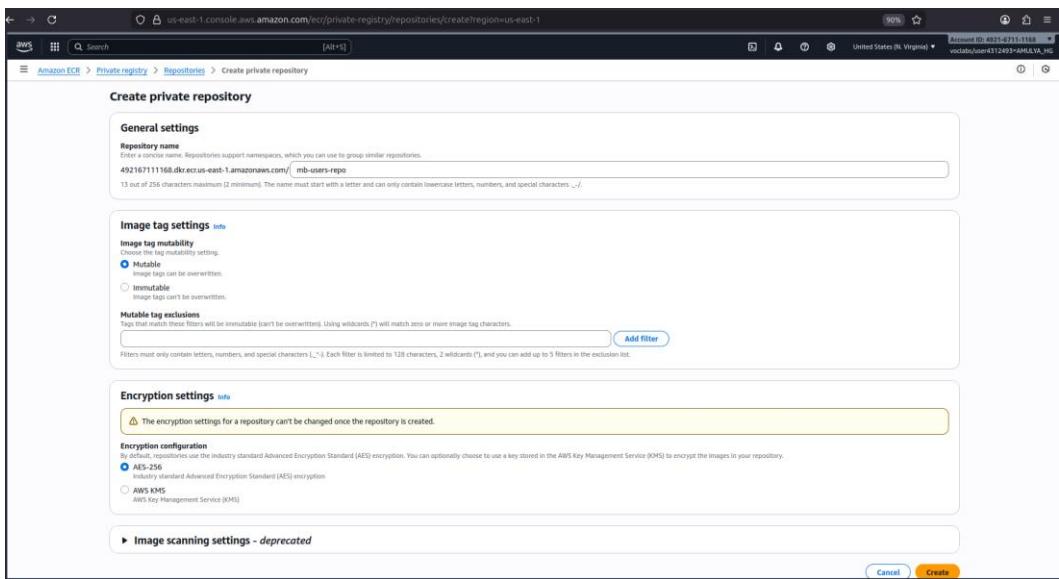
A screenshot of a web browser displaying JSON data. The URL is <http://mb-load-balancer-122502287.us-east-1.elb.amazonaws.com/api/posts/in-thread/2>. The JSON response contains three objects, each representing a post:

```
[{"post": 0, "thread": 2, "text": "I have to try their tarts!", "user": 3}, {"post": 1, "thread": 2, "text": "I'm planning to stop by in the morning to try their croissants.", "user": 2}, {"post": 2, "thread": 2, "text": "I could go for a chocolate eclair!", "user": 1}]
```

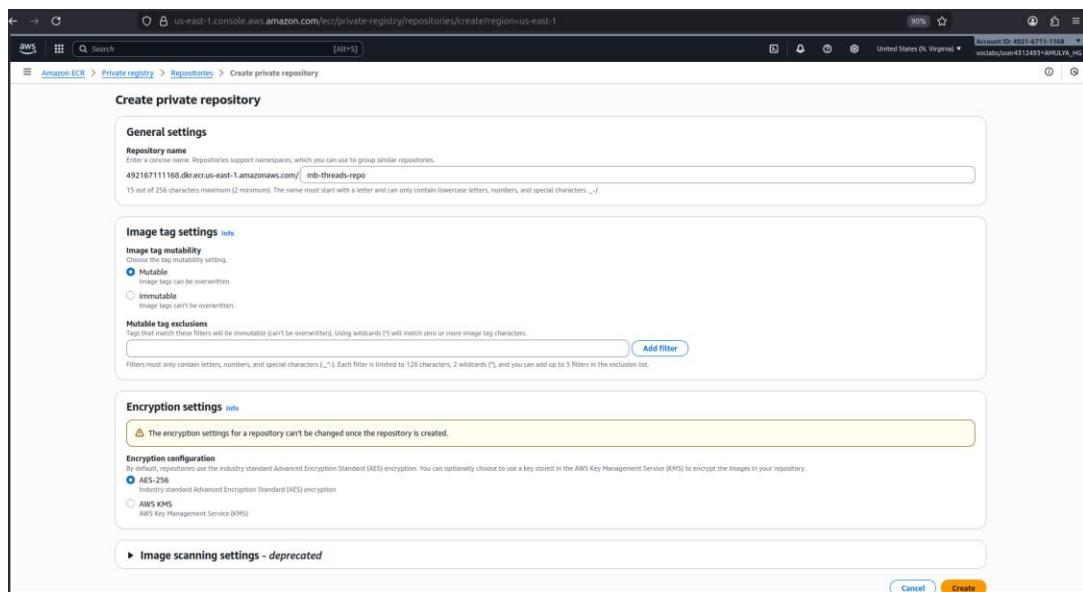
## Task 5: Refactoring the monolith

### Step1: Provisioning an ECR repository for each microservice

1. On the AWS Management Console, in the search box, enter and choose Elastic Container Registry
2. Choose Repositories, then choose Create repository.
3. For Repository name, enter mb-users-repo
4. Choose Create



## 5. Repeat the steps in this sub-task to create a repository named mb-threads-repo



## 6. Repeat the steps in this sub-task to create a repository named mb-posts-repo

**Create private repository**

**General settings**

Repository name  
Enter a concise name. Repositories support namespaces, which you can use to group similar repositories.  
492167111168.dkr.ecr.us-east-1.amazonaws.com/ **mb-posts-repo**  
13 out of 256 characters maximum (2 minimum). This name must start with a letter and can only contain lowercase letters, numbers, and special characters \_-./.

**Image tag settings** Info

Image tag mutability  
Choose the tag mutability setting.

**Mutable**  
Image tags can be overwritten.

**Immutable**  
Image tags can't be overwritten.

**Mutable tag exclusions**  
Tags that match these filters will be immutable (can't be overwritten). Using wildcards (\*) will match zero or more image tag characters.  
 **Add filter**

Filters must only contain letters, numbers, and special characters \_-./. Each filter is limited to 128 characters, 2 wildcards (\*), and you can add up to 5 filters in the exclusion list.

**Encryption settings** Info

**⚠️** The encryption settings for a repository can't be changed once the repository is created.

Encryption configuration  
By default, repositories use the industry standard Advanced Encryption Standard (AES) encryption. You can optionally choose to use a key stored in the AWS Key Management Service (KMS) to encrypt the images in your repository.

**AES-256**  
Industry standard Advanced Encryption Standard (AES) encryption

**AWS KMS**  
AWS Key Management Service (KMS)

**▶ Image scanning settings - deprecated**

**Create**

**Successfully created mb-posts-repo**

**Private repositories (4)**

Repository name	URI	Created at	Tag immutability	Encryption type
mb-posts-repo	492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-posts-repo	November 17, 2025, 21:24:35 (UTC+05:5)	Mutable	AES-256
mb-repo	492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-repo	November 17, 2025, 19:54:15 (UTC+05:5)	Mutable	AES-256
mb-threads-repo	492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-threads-repo	November 17, 2025, 21:23:56 (UTC+05:5)	Mutable	AES-256
mb-users-repo	492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo	November 17, 2025, 21:18:19 (UTC+05:5)	Mutable	AES-256

**Create repository**

## Step 2: Building and pushing the image for the users microservice

1. Switch to the Lab IDE browser tab
2. In the terminal tab, to change directory to the 3-containerized-microservices/users' folder

```
cd ~/environment/3-containerized-microservices/users
```

3. Switch to the Elastic Container Registry browser tab.

4. From the Private repositories list, choose mb-users-repo.

5. At the top of the page, choose View push commands

6. A pop-up window titled Push commands for mb-users-repo opens.

7. Enter the following commands

- docker build -t mb-users-repo
- docker tag mb-users-repo:latest [1234567890.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo:latest](https://1234567890.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo:latest)
- docker push [1234567890.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo:latest](https://1234567890.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo:latest)

```
[ec2-user@ip-10-0-1-264 ~]$ cd 3-containerized-microservices/
[ec2-user@ip-10-0-1-264 3-containerized-microservices]$ cd users
[ec2-user@ip-10-0-1-264 users]$ docker build -t mb-users-repo .
[+] Building 6.4s (9/9) FINISHED
   => [internal] load build definition from Dockerfile
   => [internal] transfer Dockerfile: 199B
   => [internal] load metadata for docker.io/nhart/alpine-node:7.10.1
   => [internal] load .dockerignore
   => [internal] transfer context: 2B
[+] 4/4 FROM docker.io/nhart/alpine-node:7.10.1@sha256:d334920c966d440676ce9d1e6162ab544349e4a4359c517380391c877bcff8c
   => [internal] load build context
   => [internal] transfer context: 1.9kB
   => [internal] CACHED [2/4] WORKDIR /srv
   => [internal] CACHED [3/4] RUN npm install
   => [internal] CACHED [4/4] RUN npm install
   => [internal] export image
   => [internal] export layers
   => [internal] naming to docker.io/library/mb-users-repo
[ec2-user@ip-10-0-1-264 users]$ docker tag mb-users-repo:latest 49210711168.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo:latest
[ec2-user@ip-10-0-1-264 users]$ docker push 49210711168.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo:latest
The push refers to repository [49210711168.dkr.ecr.us-east-1.amazonaws.com/mb-users-repo]
558709777908: Pushed
d1f706f0e3d3: Pushed
5f5f13345260: Pushed
3eb933345260: Pushed
046fd7841192: Pushed
latest: digest: sha256:611e3e2a5674afe3f0dd67b327b51d015648b2e3bfead05379fa1bd4ccb872c5 size: 1364
```

8. Switch to Elastic Container Registry select mb-users-repo

9. In the Image list you can see the container image that you pushed.

10. In the image list, choose copy URI for the image URI.

### Step 3: Building and pushing the image for the threads microservice

1. In the terminal tab, change directory to the 3-containerized-microservices/threads folder.

cd ~/environment/3-containerized-microservices/threads

2. Switch to the Elastic Container Registry.
3. Choose Repositories, and choose mb-threads-repo
4. At the top of the page, choose View push commands
5. A pop-up window titled Push commands for mb-threads-repo opens.
6. Repeat the steps from the previous task to do the following
  - Build the Docker image for the microservice
  - Tag the image with the repository URI so that it can be pushed to the repository
  - Push the container image to the microservice's repository

```

[ec2-user@ip-10-0-1-204 ~]$ cd threads
[ec2-user@ip-10-0-1-204 threads]$ docker build -t mb-threads-repo .
[+] Building 4.4s (9/9) FINISHED
--> [internal] load build definition from Dockerfile
--> [internal] transfer dockerfile: 199B
--> [internal] load metadata for docker.io/mhart/alpine-node:7.10.1
--> [internal] load .dockerignore
--> [internal] transfer context: 2B
--> [1/4] FROM docker.io/mhart/alpine-node:7.10.1@sha256:d334920c96dd440676ce9d1e162ab544349e4a4359c517300391c877bcff8c
--> [2/4] COPY index.js /app
--> [3/4] ADD .
--> [4/4] RUN npm install
--> exporting to image
--> exporting layers
--> [internal] load build context
--> [internal] transfer context: 1.73kB
--> [internal] naming to docker.io/library/mb-threads-repo
[ec2-user@ip-10-0-1-204 threads]$ docker tag mb-threads-repo:latest 492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-threads-repo:latest
[ec2-user@ip-10-0-1-204 threads]$ docker push 492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-threads-repo:latest
The push refers to repository [492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-threads-repo]
288d3f13f92c: Pushed
5170bf18a086: Pushed
3e893534526a: Pushed
040f7841192: Pushed
latest: digest: sha256:5c0e2fbdddf659a69cc01ff0b89e959cb78cc2d5b9483a9fee5ddde4a7cdma5 size: 1364
[ec2-user@ip-10-0-1-204 threads]$ 

```

7. Switch to Elastic Container Registry select mb-threads-repo
8. In the image list you can see the container image that you pushed
9. In the image list, choose copy URI for the image URI

### Step 3: Building and pushing the image for the posts microservice

1. In the terminal tab, change directory to the 3-containerized-microservices/posts folder.

`cd ~/environment/3-containerized-microservices/posts`

2. Switch to the Elastic Container Registry.
3. Choose Repositories, and choose mb-posts-repo
4. At the top of the page, choose View push commands
5. A pop-up window titled Push commands for mb-posts-repo opens.
6. Repeat the steps from the previous task to do the following
  - Build the Docker image for the microservice

- Tag the image with the repository URI so that it can be pushed to the repository
- Push the container image to the microservice's repository

```

[ec2-user@ip-10-0-1-204 3-containerized-microservices]$ cd posts
[ec2-user@ip-10-0-1-204 posts]$ docker build -t mb-posts-repo .
[+] Building 4.1s (9/9) FINISHED
   => [internal] load build definition from Dockerfile
   => transferring dockerfile: 199B
   => [internal] load metadata for docker.io/mhart/alpine-node:7.10.1
   => [internal] load .dockerignore
   => [internal] load index.json
   => [1/4] FROM docker.io/mhart/alpine-node:7.10.1@sha256:d334920c966d440876ce9d1e162ab544349e4a4359c517300391c877bcffbb0c
   => [internal] load context
   => transferring context: 2.15kB
   => CACHE [2/4] WORKDIR /usr/bin
   => [3/4] ADD .
   => [4/4] RUN npm install
   => [internal] load layers
   => exporting layers
   => writing image sha256:f3cf2413896fb6f045a29a369ee49724297eb2932c9534618990c2e3428348af
[ec2-user@ip-10-0-1-204 posts]$ docker tag mb-posts-repo:latest 492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-posts-repo:latest
[ec2-user@ip-10-0-1-204 posts]$ docker push 492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-posts-repo:latest
The push refers to a repository [492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-posts-repo]
492167111168.dkr.ecr.us-east-1.amazonaws.com/mb-posts-repo:latest
343bf9080ca2: Pushed
5f70bf18a886: Pushed
3e893534526a: Pushed
040fd7d841192: Pushed
latest: digest: sha256:8da8513a80dd890b1ddf362f0db6953ab1955812ce687f36802f60716f3511b size: 1364
[ec2-user@ip-10-0-1-204 posts]$

```

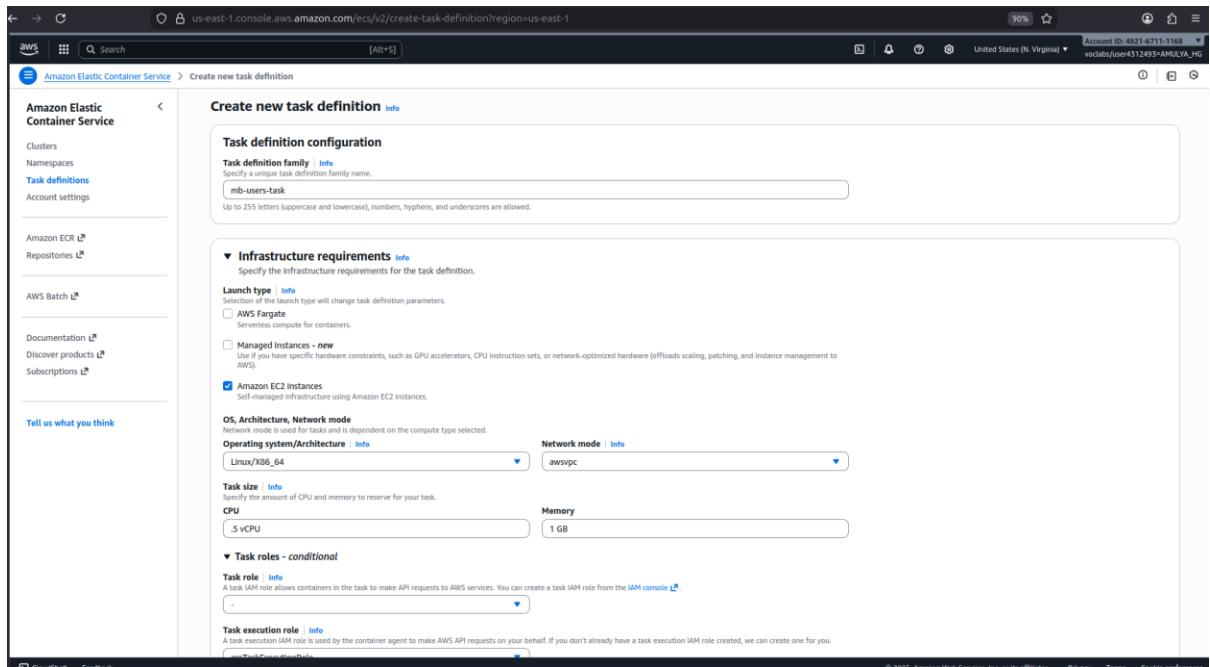
7. Switch to Elastic Container Registry select mb-posts-repo
8. In the image list you can see the container image that you pushed
9. In the image list, choose copy URI for the image URI

## Task 6: Deploying the containerized microservices

### Step 1: Creating a task definition for the users microservice

1. On the AWS Management Console, in the search box, enter and select Elastic Container Service.
2. Choose Task Definition
3. Choose Create new Task definition
4. In the Task definition configuration section, for Task definition family, enter mb-users-task
5. In the Infrastructure requirements, select Amazon EC2 instances, and clear AWS Fargate
6. For the Task size, choose CPU: .5 vCPU, Memory: 1GB
7. In the Container - 1 section, configure the following options:
8. For Container details, for Name, enter mb-users-container
9. For Image URI, paste the URI of the users container image that you copied to a text editor earlier.
10. For Port mappings, for Container port, enter 3000.

## 11. Choose Create.



### Step 2: Creating a task definition for the posts microservice

1. In the left navigation pane, choose Task definitions.
2. Choose Create new task definition, and configure the following options:
  3. In the Task definition configuration section, for Task definition family, enter mb-posts-task
    - a. In the Infrastructure requirements, select Amazon EC2 instances, and clear AWS Fargate.
    - b. For the Task size, choose CPU: .5 vCPU, Memory: 1GB
    - c. For Container - 1, configure the following options:
      - d. For Container details, for Name, enter mb-posts-container
      - e. For Image URI, paste the URI of the posts container image that you copied to a text editor earlier.
      - f. For Port mappings, for Container port, enter 3000. This option specifies the port on which the container receives requests.
  4. Choose Create.
  5. A message is displayed at the top that says, "Task definition successfully created."

### Step 3: Creating a task definition for the threads microservice

1. In the left navigation pane, choose Task definitions.
2. Choose Create new task definition, and configure the following options:

3. In the Task definition configuration section, for Task definition family, enter mb-threads-task

- In the Infrastructure requirements, select Amazon EC2 instances, and clear AWS Fargate.
- For the Task size, choose CPU: .5 vCPU, Memory: 1GB
- For Container - 1, configure the following options:
  - For Container details, for Name, enter mb-threads-container
  - For Image URI, paste the URI of the threads container image that you copied to a text editor earlier.
  - For Port mappings, for Container port, enter 3000.

4. Choose Create.

Task 7: Creating a task definition for each microservice

Step1: Creating a task definition for the users microservice

1. On the AWS Management Console, in the search box, enter and select Elastic Container Service
2. In the left navigation pane, choose Task definitions.
3. Choose Create new task definition, and configure the following options:
  - a. In the Task definition configuration section, for Task definition family, enter mb-users-task
  - b. In the Infrastructure requirements, select Amazon EC2 instances, and clear AWS Fargate.
  - c. For the Task size, choose CPU: .5 vCPU, Memory: 1GB
  - d. In the Container - 1 section, configure the following options:
    - e. For Container details, for Name, enter mb-users-container
    - f. For Image URI, paste the URI of the users container image that you copied to a text editor earlier.
    - g. For Port mappings, for Container port, enter 3000. This option specifies the port on which the container receives requests.

4. Choose Create.

Step 2: Creating a task definition for the posts microservice

1. In the left navigation pane, choose Task definitions.
2. Choose Create new task definition, and configure the following options:  
In the Task definition configuration section, for Task definition family, enter mb-posts-task
  - In the Infrastructure requirements, select Amazon EC2 instances, and clear AWS Fargate.
  - For the Task size, choose CPU: .5 vCPU, Memory: 1GB
  - For Container - 1, configure the following options:

- For Container details, for Name, enter mb-posts-container
  - For Image URI, paste the URI of the posts container image that you copied to a text editor earlier.
  - For Port mappings, for Container port, enter 3000. This option specifies the port on which the container receives requests.
3. Choose Create.

#### Step 3: Creating a task definition for the threads microservice

1. In the left navigation pane, choose Task definitions.
2. Choose Create new task definition, and configure the following options:  
In the Task definition configuration section, for Task definition family, enter mb-threads-task
  - In the Infrastructure requirements, select Amazon EC2 instances, and clear AWS Fargate.
  - For the Task size, choose CPU: .5 vCPU, Memory: 1GB
  - For Container - 1, configure the following options:
    - For Container details, for Name, enter mb-threads-container
    - For Image URI, paste the URI of the threads container image that you copied to a text editor earlier.
    - For Port mappings, for Container port, enter 3000. This option specifies the port on which the container receives requests.
3. Choose Create.

#### Task 8: Creating and deploying the services

##### Step 1:

1. In the left navigation pane, choose Clusters, and choose your mb-ecs-cluster cluster.
2. On the Services tab, choose Create, and configure the following options:
  - In the Environment section, configure the following options:
    - For Compute options, choose Launch type.
    - For Launch type, choose EC2.
  - In the Deployment configuration section, configure the following options:
    - For Application type, choose Service.
    - For Family, chose mb-users-task.
    - For Service name, enter mb-users-service
  - Expand the Networking section, and configure the following options:
    - For Security group, choose Use an existing security group.
    - From the Security group name dropdown list, select the security group that has ECSSG in the name.
    - Clear the default security group.
  - Expand the Load balancing - optional section, and configure the following options:

- Check Use load balancing.
  - For Load balancer type, choose Application Load Balancer.
  - For Application Load Balancer, choose Use an existing load balancer.
  - For Load balancer, choose mb-load-balancer.
  - For Listener, choose Use an existing listener, and then choose 80:HTTP from the dropdown list.
  - For Target group, choose Create new target group.
  - For Target group name, enter mb-users-target
  - For Path pattern, enter /api/users\*
  - For Evaluation order, enter 1
3. Choose Create.

Step 2:

1. Return to your mb-ecs-cluster cluster. On the Services tab, choose Create, and configure the following options:
    - In the Environment section, configure the following options:
      - For Compute options, choose Launch type.
      - For Launch type, choose EC2.
    - In the Deployment configuration section, configure the following options:
      - For Application type, choose Service.
      - For Family, choose mb-posts-task.
      - For Service Name, enter mb-posts-service
    - Expand the Networking section, and configure the following options:
      - For Security group, choose Use an existing security group.
      - From the Security group name dropdown list, select the security group that has ECSSG in the name.
      - Clear the default security group.
    - Expand the Load balancing section, and configure the following options:
      - Check Use load balancing.
      - For Load balancer type, choose Application Load Balancer.
      - For Application Load Balancer, choose Use an existing load balancer.
      - For Load balancer, choose mb-load-balancer.
      - For Listener, choose Use an existing listener, and then choose 80:HTTP from the dropdown list.
      - For Target group, choose Create new target group.
      - For Target group name, enter mb-posts-target
      - For Path pattern, enter /api/posts\*
      - For Evaluation order, enter 2
2. Choose Create.

Step 3:

1. Return to your mb-ecs-cluster cluster. On the Services tab, choose Create, and configure the following options:
  - In the Environment section, configure the following options:
    - For Compute options, choose Launch type.
    - For Launch type, choose EC2.
  - In the Deployment configuration section, configure the following options:
    - For Application type, choose Service.
    - For Family, choose mb-threads-task.
    - For Service Name, enter mb-threads-service
  - Expand the Networking section, and configure the following options:
    - For Security group, choose Use an existing security group.
    - From the Security group name dropdown list, select the security group that has ECSSG in the name.
    - Clear the default security group.
  - Expand the Load balancing section, and configure the following options:
    - Check Use load balancing.
    - For Load balancer type, choose Application Load Balancer.
    - For Application Load Balancer, choose Use an existing load balancer.
    - For Load balancer, choose mb-load-balancer.
    - For Listener, choose Use an existing listener, and then choose 80:HTTP from the dropdown list.
    - For Target group, choose Create new target group.
    - For Target group name, enter mb-threads-target
    - For Path pattern, enter /api/threads\*
    - For Evaluation order, enter 3
2. Choose Create.

#### Task 9: Validating the deployment

In the browser address bar, enter the following addresses in the browser tab, and examine the results.

- *DNS name/api/users*

Expected output: List of users

- *DNS name/api/users/2*

Expected output: Details of user 2

- *DNS name/api/threads*

Expected output: List of threads

- *DNS name/api/posts/in-thread/2*

Expected output: Details of thread 2

- *DNS name/xxx*

Expected output: Invalid request