

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

Aim :- Run containers on a managed service.

Lab overview and objectives

In a previous lab, you migrated an application that ran on Amazon Elastic Compute Cloud (Amazon EC2) instances to run on Docker containers. In this lab, you will deploy the application using two managed cloud services. You will deploy the database tier using Amazon Aurora Serverless and the web tier using AWS Elastic Beanstalk.

After completing this lab, you should be able to:

- Create a new Amazon Relational Database Service (Amazon RDS) instance using the AWS Management Console
- Launch a Docker container on an IDE using an image pulled from Amazon Elastic Container Registry (Amazon ECR)
- Configure and test the containerized application connection to Aurora Serverless
- Use the Amazon RDS query editor to create database objects and load data
- Launch the default Elastic Beanstalk application
- Update the Elastic Beanstalk application to run your node application and communicate with Amazon RDS
- Configure an Amazon API Gateway endpoint to forward calls to the Elastic Beanstalk URL

AWS service restrictions

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond the ones that are described in this lab.

Scenario

Sofía has containerized the coffee suppliers application. Now the café has asked if she can reduce the required manual application maintenance and plan for the future scalability of the environment. She knows from her studies that Amazon Web Services (AWS) provides several managed services. Managed services can help to reduce the burden of deploying and managing applications. After more research, she has decided to deploy the suppliers application website using Elastic Beanstalk.

However, Sofia has discovered that scaling a relational database using containers is not recommended because relational databases are stateful. Relational databases require reliable communication between database hosts and storage. This is difficult to accomplish using dynamic containers. Therefore, Sofia has decided to use Aurora Serverless as the data platform. Sofia will retire the container-based MySQL database and load the required user, tables, and data into an Aurora Serverless database. Aurora Serverless was designed to seamlessly and safely scale databases as transaction loads increase. As a bonus, when the database isn't being used, Aurora Serverless automatically scales down, which will save money for the café.

In this lab, you will again play the role of Sofía, and you will deploy the suppliers application using managed services.

Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

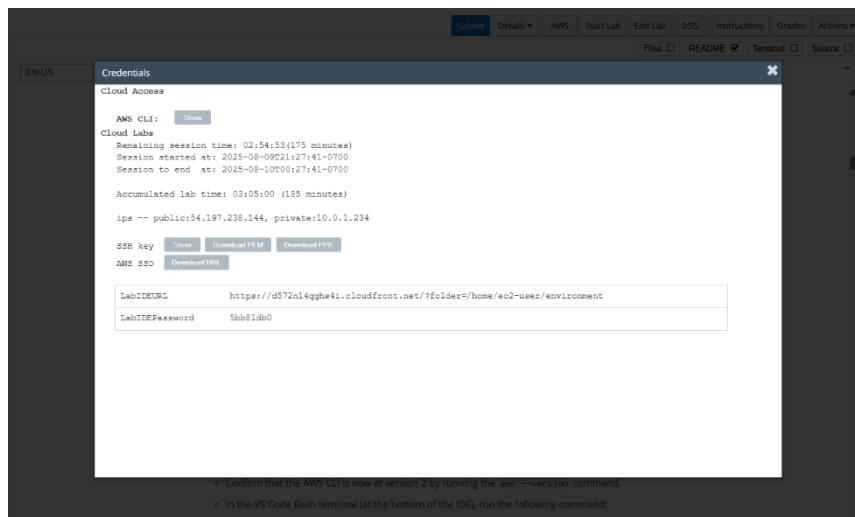
Date:

Enrolment No: 92200133030

Task 1: Preparing the lab

Connect to the VS Code IDE.

1. At the top of these instructions, choose **Details** followed by **AWS: Show**
2. Copy values from the table **similar** to the following and paste it into an editor of your choice for use later.
 - a. **LabIDEURL**
 - b. **LabIDEPASSWORD**



3. In a new browser tab, paste the value for **LabIDEURL** to open the VS Code IDE.
4. On the prompt window **Welcome to code-server**, enter the value for **LabIDEPASSWORD** you copied to the editor earlier, choose **Submit** to open the VS Code IDE.

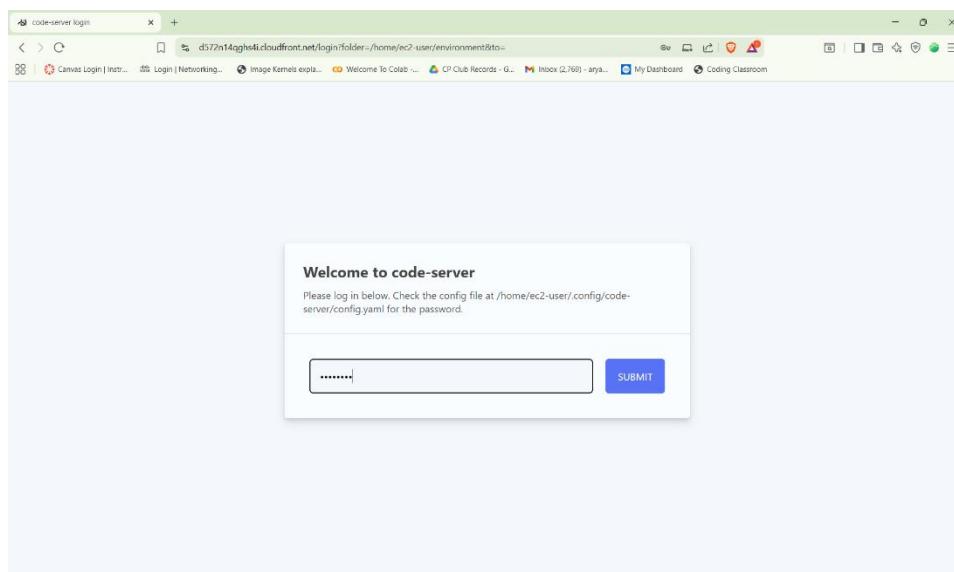
Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

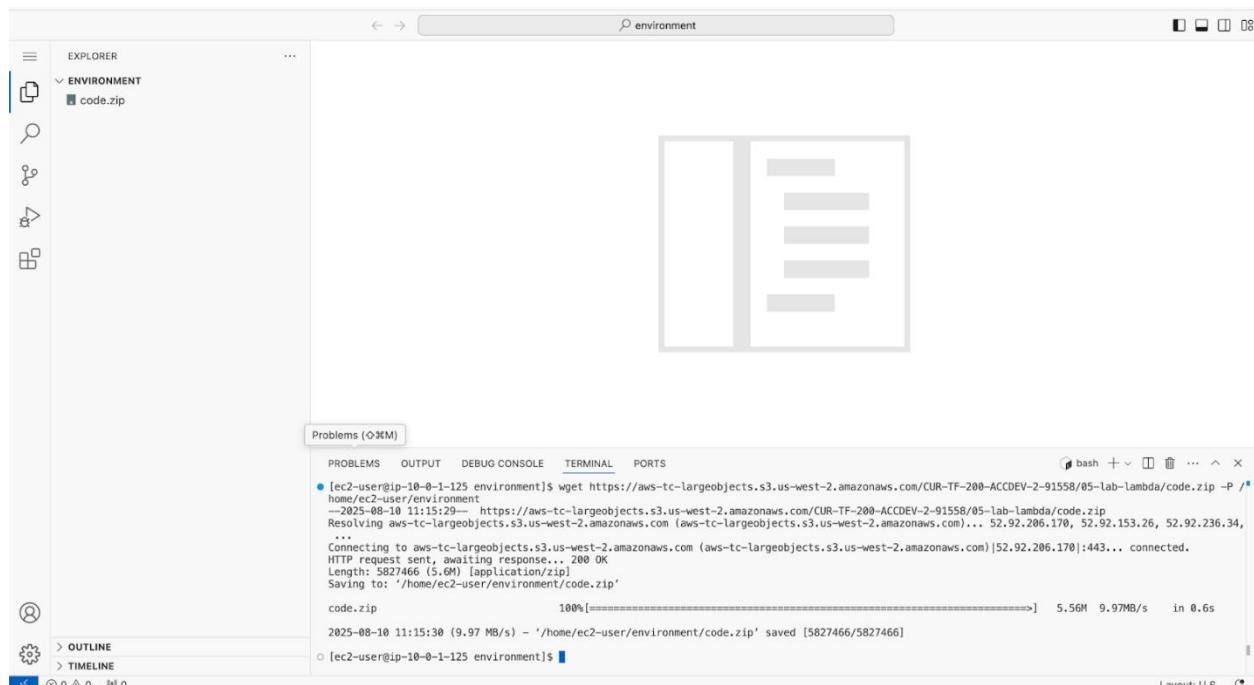
Date:

Enrolment No: 92200133030



- Download and extract the files that you need for this lab.
 - In the VS Code bash terminal (located at the bottom of the IDE), run the following commands:

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCDEV-2-91558/07-lab-deploy/code.zip -P /home/ec2-user/environment
```

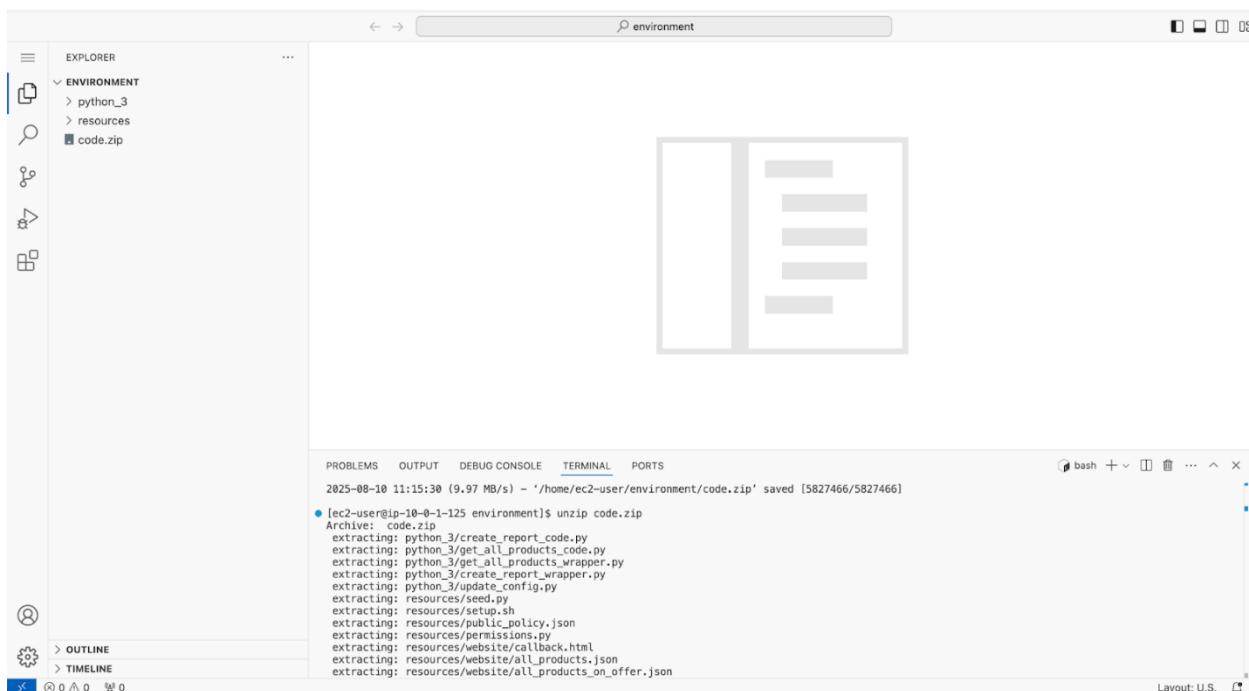


- You should see that the **code.zip** file was downloaded to the VS Code IDE and is now in the left navigation pane.



Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

- Extract the file by running the following command:
`unzip code.zip`



The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with a tree view showing 'ENVIRONMENT' expanded, containing 'python_3', 'resources', and 'code.zip'. The main area is the Terminal tab, which displays the command 'unzip code.zip' being run and its output. The output shows the extraction of several Python files and JSON files from the archive. The status bar at the bottom indicates 'Layout: U.S.'

```
2025-08-10 11:15:30 (9.97 MB/s) - '/home/ec2-user/environment/code.zip' saved [5827466/5827466]
[ec2-user@ip-10-0-1-125 environment]$ unzip code.zip
Archive:  code.zip
  extracting: python_3/create_report_code.py
  extracting: python_3/get_all_products_code.py
  extracting: python_3/get_all_products_wrapper.py
  extracting: python_3/create_report_wrapper.py
  extracting: python_3/update_config.py
  extracting: resources/seed.py
  extracting: resources/setup.sh
  extracting: resources/public_policy.json
  extracting: resources/permissions.py
  extracting: resources/website/callback.html
  extracting: resources/website/all_products.json
  extracting: resources/website/all_products_on_offer.json
```

- Run a script that upgrades the version of the AWS CLI installed on the VS Code IDE.
 - To set permissions on the script and then run it, run the following commands in the Bash terminal:

```
chmod +x ./resources/setup.sh && ./resources/setup.sh
```

The script will prompt you for the **IP address** by which your computer is known to the internet. Use www.whatismyip.com to discover this address and then paste the IPv4 address into the command prompt and finish running the script.



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

```
● [ec2-user@ip-10-0-1-150 environment]$ chmod +x resources/setup.sh && resources/setup.sh
Please enter a valid IP address:
152.58.63.192
IP address:152.58.63.192
upload: resources/website/all_products_on_offer.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/all_products_on_offer.json
upload: resources/website/callback.html to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/callback.html
upload: resources/website/all_products.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/all_products.json
upload: resources/website/beans.json to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/beans.json
upload: resources/website/images/beans/excelsa.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/beans/excelsa.png
upload: resources/website/config.js to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/config.js
upload: resources/website/images/items/blueberry_bagel.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/blueberry_bagel.png
upload: resources/website/images/items/apple_pie.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/apple_pie.jpeg
upload: resources/website/images/items/blueberry_jelly_doughnut.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/blueberry_jelly_doughnut.jpeg
upload: resources/website/images/beans/robusta.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/beans/robusta.png
upload: resources/website/images/items/boston_cream_doughnut.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/boston_cream_doughnut.jpeg
upload: resources/website/images/expanded.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/expanded.png
upload: resources/website/images/beans/liberica.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/beans/liberica.png
upload: resources/website/images/items/apple_pie_slice.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/apple_pie_slice.png
upload: resources/website/images/items/apple_pie.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/apple_pie.png
upload: resources/website/images/beans/arabica.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/beans/arabica.png
upload: resources/website/images/items/boston_cream_doughnut.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/boston_cream_doughnut.png
upload: resources/website/favicon.ico to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/favicon.ico
upload: resources/website/images/items/apple_pie_slice.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/apple_pie_slice.jpeg
upload: resources/website/images/items/cherry_pie.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/cherry_pie.png
upload: resources/website/images/items/blueberry_bagel.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/blueberry_bagel.jpeg
upload: resources/website/images/items/blueberry_jelly_doughnut.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/blueberry_jelly_doughnut.png
upload: resources/website/images/items/cherry_pie_slice.png to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/cherry_pie_slice.png
upload: resources/website/images/items/cherry_pie.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/cherry_pie.jpeg
upload: resources/website/images/items/cherry_pie_slice.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/cherry_pie_slice.jpeg
upload: resources/website/images/items/chocolate_chip_cupcake.jpeg to s3://c168617a4340248l11142234t1w184333714729-s3bucket-1wvvevyh51/images/items/chocolate_chip_cupcake.jpeg
```

Layout: US

8. Verify the AWS CLI version and also verify that the SDK for Python is installed.

- Confirm that the AWS CLI is now at version 2 by running the **aws --version** command.
- In the VS Code Bash terminal (at the bottom of the IDE), run the following command:

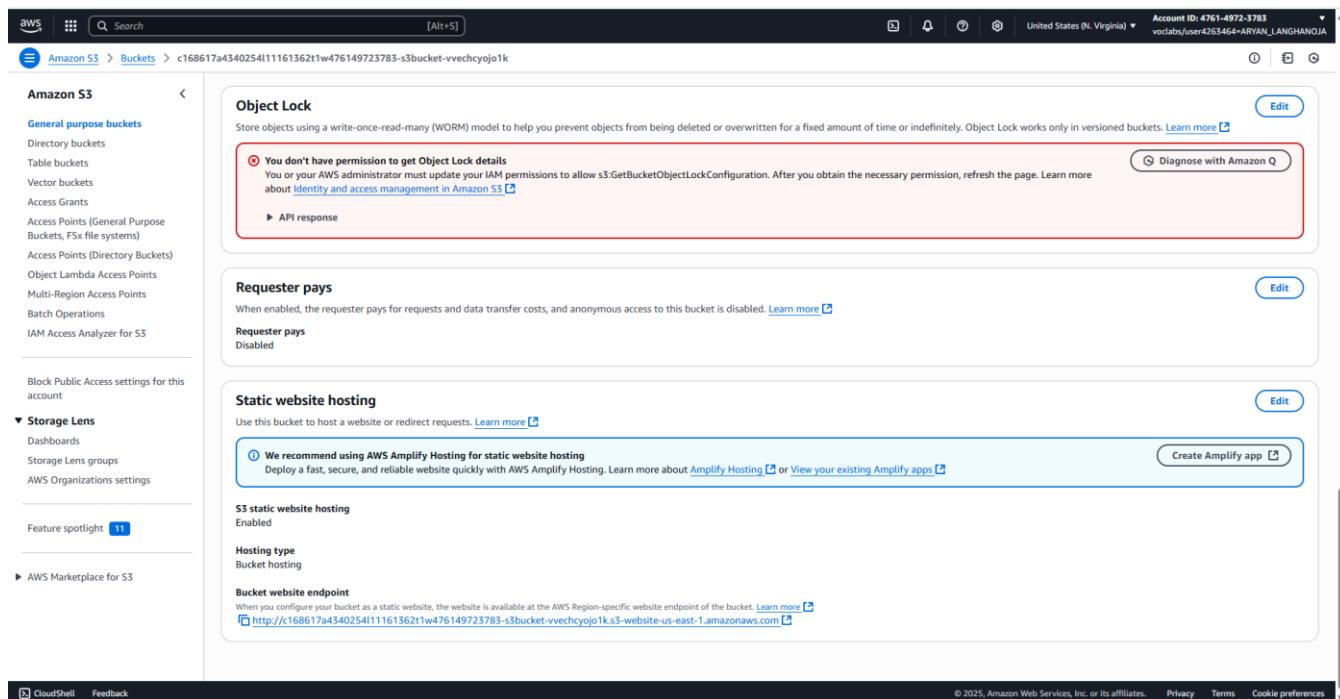
pip3 show boto3

```
● [ec2-user@ip-10-0-1-234 environment]$ aws --version
aws-cli/2.28.6 Python/3.13.4 Linux/6.1.147-172.266.amzn2023.x86_64 exe/x86_64.amzn.2023
● [ec2-user@ip-10-0-1-234 environment]$ pip3 show boto3
Name: boto3
Version: 1.40.6
Summary: The AWS SDK for Python
Home-page: https://github.com/boto/boto3
Author: Amazon Web Services
Author-email:
License: Apache License 2.0
Location: /usr/local/lib/python3.11/site-packages
Requires: botocore, jmespath, s3transfer
Required-by:
○ [ec2-user@ip-10-0-1-234 environment]$ █
```

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

9. Confirm access to the café website.

- Navigate to the Amazon S3 console.
- Choose the link for the bucket that has *-s3bucket* in the name.
- Choose the **Properties** tab, then scroll down to the Static website hosting section.
- Choose the URL that appears under Bucket website endpoint.
- The café website displays in a new browser tab. If it doesn't, see the following troubleshooting tip.



The screenshot shows the AWS S3 Bucket Properties page for a bucket named 'c168617a4340254111161362t1w476149723783-s3bucket-vvechcyojo1k'. The 'Static website hosting' section is highlighted with a red box. It shows that 'Requester pays' is disabled and recommends using AWS Amplify Hosting for static website hosting. The 'Bucket website endpoint' field contains the URL <http://c168617a4340254111161362t1w476149723783-s3bucket-vvechcyojo1k.s3-website-us-east-1.amazonaws.com>.

10. Open your preferred text editor, and copy and paste the following text into a new file.

```

3  IDE VPC ID:
4  IDE Availability Zone:
5  IDE subnet ID:
6  extraSubnetForRds subnet ID:
7  IDE security group ID:
8  Database endpoint:
9  Repository URI:
10 Elastic Beanstalk URL:

```



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

Task 2: Configuring the subnets for Amazon RDS and Elastic Beanstalk to use

11. Navigate to the Amazon VPC console.

- Return to the AWS Management Console browser tab.
- From the **Services** menu, choose **VPC**.
- In the left navigation pane, choose **Your VPCs**.
- Select the checkbox for *IDE VPC*, and then copy the **VPC ID** into your text editor.

12. Review the Availability Zone for the *IDE Subnet*.

- In the left navigation pane, choose **Subnets**.
- Select the checkbox for *IDE Public Subnet One*.
- In the bottom pane, locate the **Availability Zone** and copy the value into your text editor.

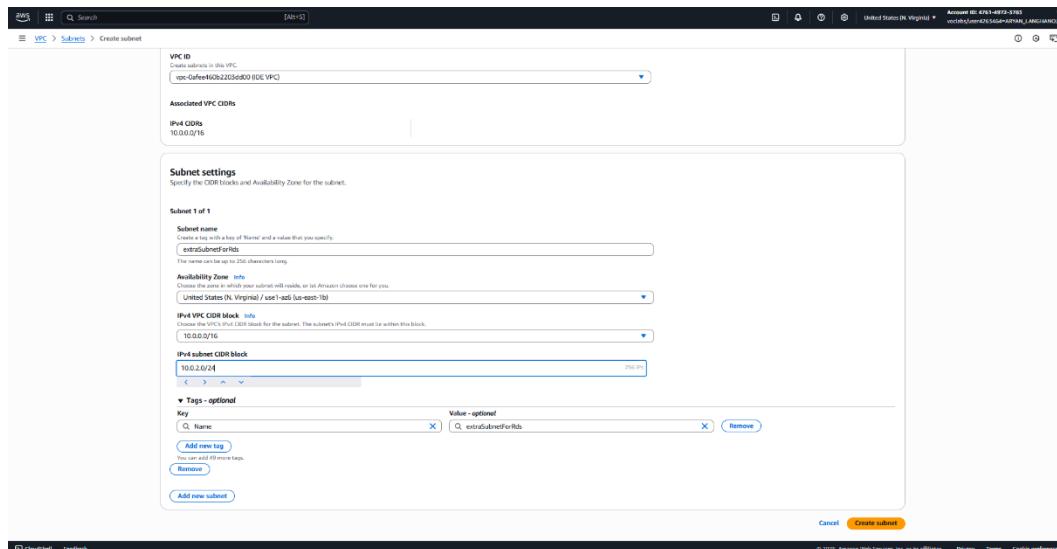
The Availability Zone looks similar to the following: *us-east-1a*

- Copy the **Subnet ID** value into your text editor as the **IDE Subnet ID**.

 Marwadi University <small>Marwadi Chandarana Group</small>	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

13. Create a second subnet to provide high availability for the database and application in the future:

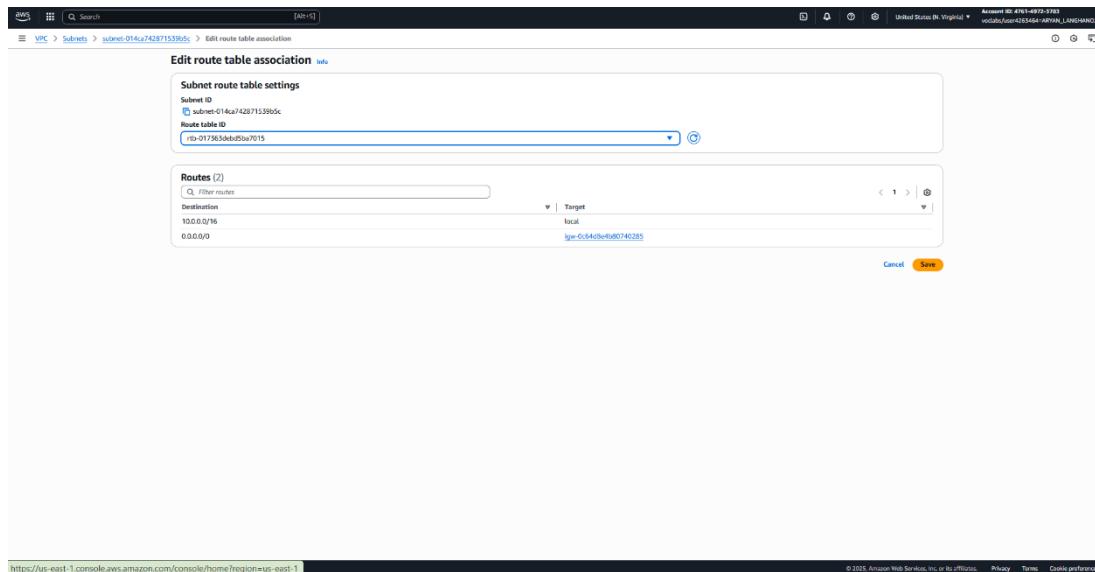
- Choose **Create subnet** and configure the following:
 - **VPC ID:** Choose **IDE VPC**
 - **Subnet name:** Enter **extraSubnetForRds**
 - **Availability Zone:** Choose a different Availability Zone than the one that the *IDE Subnet* is using
 - **IPv4 CIDR block:** Enter **10.0.2.0/24**
 - Choose **Create subnet**.
 - Configure the subnet to automatically assign a public IP address:
 - Select the checkbox for *extraSubnetForRds*.
 - From the **Actions** menu, select **Edit subnet settings**.
 - Select the checkbox for **Enable auto-assign public IPv4 address**.
 - Choose **Save**.
 - In a text editor, record the **Subnet ID** as the **extraSubnetForRds subnet ID**.



14. Update the route table for the *extraSubnetForRds* subnet.

- Select the checkbox for *extraSubnetForRds*.
- In the bottom pane, choose the **Route table** tab, and review the route table.
 - Notice that this route table does not include a path to the public internet (0.0.0.0/0). For this configuration to work, you need to update the route table association for this subnet.
- Choose **Edit route table association**.
 - The subnet is currently configured to use the *Main route table*.
- For **Route table ID**, choose the other route table.
- Choose **Save** then choose the **Route table** tab.

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030



Task 3: Setting up an Aurora Serverless database

Instead of running the database on an EC2 instance or in a Docker container, the application will use the Aurora Serverless managed service as the data platform.

In this task, you will create a new Aurora Serverless instance.

15. Create an Aurora Serverless database instance.

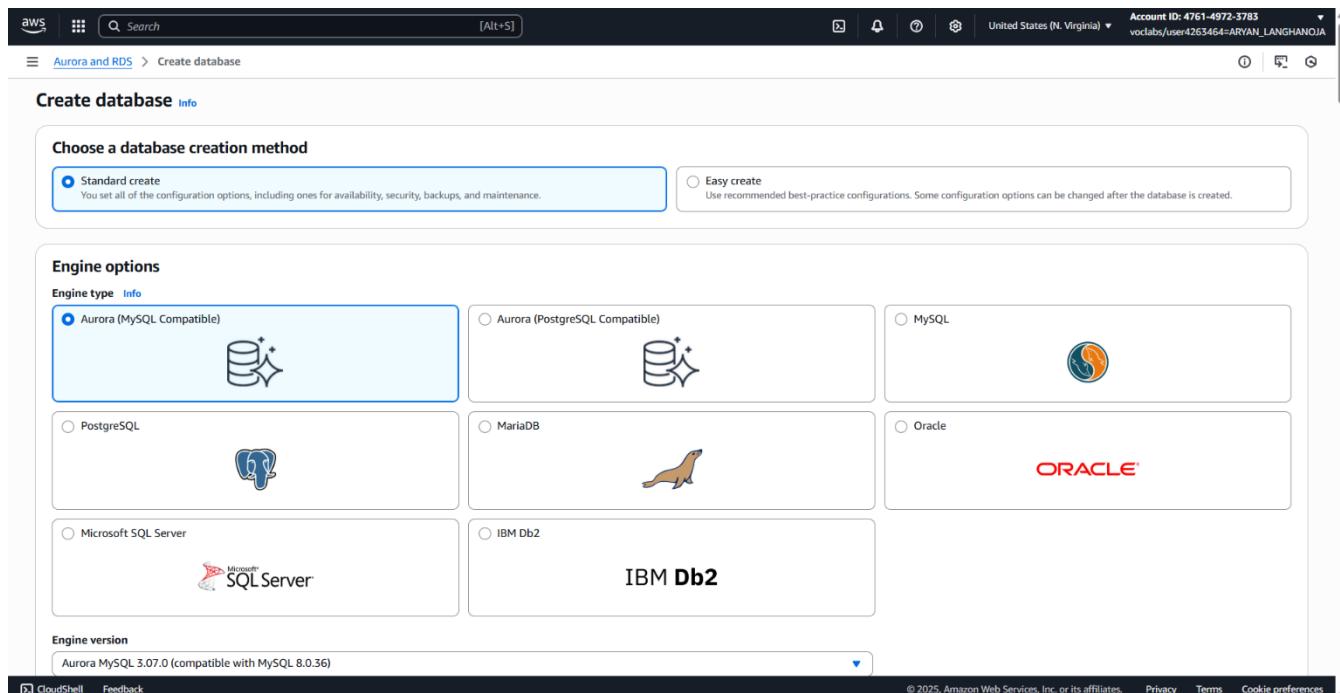
- In the search box to the right of **Services**, search for and choose **RDS**.
- In the Create database panel under the Resources panel, choose **Create database**.
- In the **Choose a database creation method** section, choose **Standard create**.
- In the **Engine options** section, configure the following:
 - **Engine type:** Choose **Aurora(MySQL-Compatible)**
 - **Version:** Choose **Aurora MySQL 3.07.0 (compatible with MySQL 8.0.36)**
- In the **Templates** section, choose **Dev/Test**.
- In the **Settings** section, configure the following:
 - **DB cluster identifier:** Enter `supplierdb`
- Under **Credentials management**, choose **Self managed**.
- Ensure that **Auto generate a password** is NOT checked
- **Master password:** Enter `coffee_beans_for_all`

 **Note:** This is the password for the database super user that is named *admin*. You will use this password later in this assignment.

- **Confirm password:** Re-enter the password
- In the **Instance configuration** section, configure the following:

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

- **DB instance class:** Choose **Serverless v2**
- Also note the default **Capacity range** settings (Minimum 2 ACU, Maximum 16 ACU) but do not change them.
- In the **Connectivity** section, configure the following:
 - **Virtual Private Cloud (VPC):** Choose **IDE VPC**
 - Under **VPC security group (firewall)**, choose **Existing VPC security groups**
 - Remove the **default** security group
 - From the Dropdown, add the security group with **Lab IDE** in the name
- Under **RDS Data API**, Select Checkbox for *Enable the RDS Data API*.
- Under **Monitoring > Performance Insights**
 - **UnCheck/DeSelect** *Enable Performance Insights (cluster level)*
 - **Expand Additional configuration (Enhanced Monitoring)**
 1. **Uncheck/DeSelect** *Enable Enhanced Monitoring*
- **Expand Additional configuration**
- **For Initial database name**, enter **suppliers**
- Keep rest of the parameters at their default values and Choose **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 

Engine version

Aurora MySQL 3.07.0 (compatible with MySQL 8.0.36)

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

16. Find the endpoint for your database.

- Choose the **supplierdb** hyperlink displayed at the top.
- From the **DB identifier**, select **supplierdb-instance-1**.
 - **Note:** Wait for the Status to become *Available*.
- On the In the bottom pane, in the **Connectivity & security** section
 - Copy the **Endpoint** value to your text editor.

The endpoint is similar to *supplierdb.cluster-xxxxxxxxxx.us-east-1.rds.amazonaws.com*.



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

The screenshot shows the AWS Aurora RDS console. On the left, there's a sidebar with navigation links like Dashboard, Databases, Query editor, etc. The main area displays the connectivity & security details for a database instance named 'supplierdb-instance-1'. It shows the endpoint (supplierdb-instance-1.cbz7vvte2w.us-east-1.rds.amazonaws.com), port (3306), VPC (IDE VPC (vpc-04f32ae033e769f30)), and subnet group (default-vpc-04f32ae033e769f30). Under networking, it lists availability zone (us-east-1b) and network type (IPv4). In the security section, it shows VPC security groups (Lab IDE SG (sg-0e796045c57522e09)) and certificate authority (rds-ca-rsa2048-g1). The certificate authority date is May 26, 2061, and the DB instance certificate expiration date is August 15, 2026, 08:40 (UTC+05:30).

17. Find the ID for the database security group.

- In the **Security** section, choose the **VPC security groups** hyperlink.
- On the **Security Groups** page, in the bottom pane, on the **Details** tab, copy the **Security group ID** value to your text editor.
- The security group ID is similar to *sg-123456acdbe*.

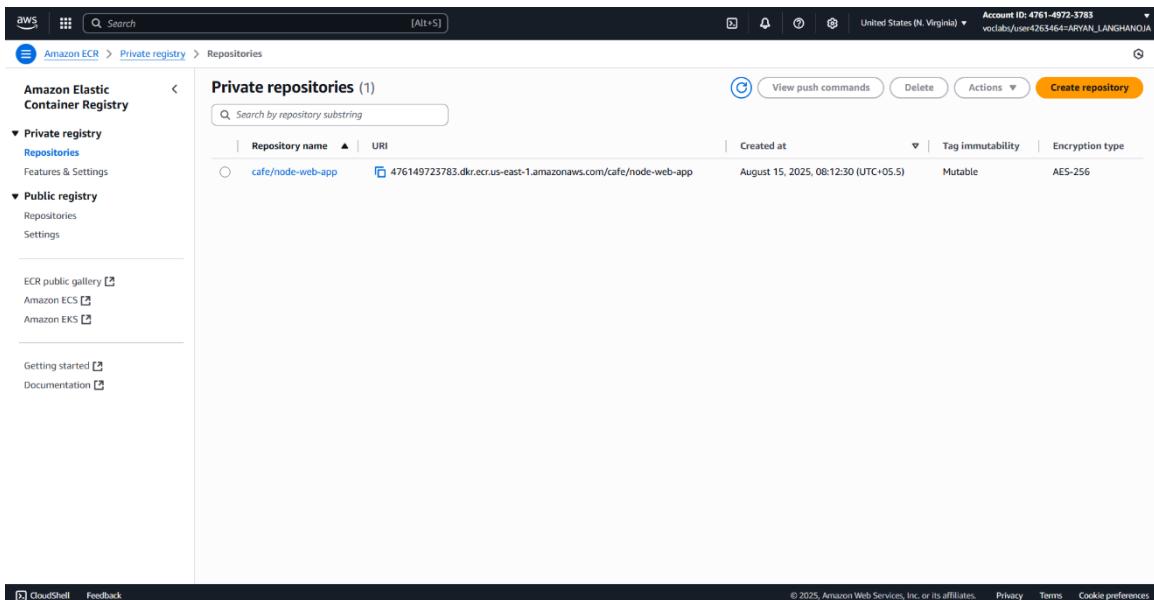
The screenshot shows the AWS EC2 Security Groups console. The left sidebar includes links for EC2 Global View, Instances, Images, Elastic Block Store, and Network & Security. The main area displays the details for a security group named 'sg-0e796045c57522e09 - Lab IDE SG'. It shows the security group name (Lab IDE SG), owner (476149723783), security group ID (sg-0e796045c57522e09), and description (SG for Developer Machine - only allow CloudFront ingress). The inbound rules count is 1, and the outbound rules count is 1. Below this, the inbound rules table shows one rule: sgr-0f73bf2011fb9bf0d, which is an HTTP rule on port 80 from source pl-3b927c52 (com.amazonaws.com).

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

Task 4: Reviewing the container image

18. Review the Docker image in the console.

- In the search box to the right of Services, search for and choose Elastic Container Registry.
- For the cafe/node-web-app repository, copy the URI value to your text editor as the Repository URI.
- Choose the cafe/node-web-app hyperlink, and review the details



The screenshot shows the AWS ECR console interface. On the left, there's a sidebar with navigation options like 'Amazon Elastic Container Registry', 'Private registry Repositories', 'Public registry Repositories', and 'Getting started'. The main area is titled 'Private repositories (1)' and lists a single repository: 'cafe/node-web-app'. The repository details are as follows:

Repository name	URI	Created at	Tag immutability	Encryption type
cafe/node-web-app	476149723783.dkr.ecr.us-east-1.amazonaws.com/cafe/node-web-app	August 15, 2025, 08:12:30 (UTC+05:5)	Mutable	AES-256

At the top right, there are buttons for 'View push commands', 'Delete', 'Actions', and 'Create repository'. The top bar also shows the account ID (4761-4972-3783) and the user's name (vordabs/user4263464-ARYAN_LANGHANOJA).

19. Review this same Docker image information in the VS Code bash terminal.

- Return to the VS Code IDE browser tab.
- Run the following command:

```
aws ecr describe-repositories
```

```
[ec2-user@ip-10-0-1-151 environment]$ aws ecr describe-repositories
{
  "repositories": [
    {
      "repositoryArn": "arn:aws:ecr:us-east-1:476149723783:repository/cafe/node-web-app",
      "registryId": "476149723783",
      "repositoryName": "cafe/node-web-app",
      "repositoryUri": "476149723783.dkr.ecr.us-east-1.amazonaws.com/cafe/node-web-app",
      "createdAt": "2025-08-15T02:42:30.172000+00:00",
      "imageTagMutability": "MUTABLE",
      "imageScanningConfiguration": {
        "scanOnPush": false
      },
      "encryptionConfiguration": {
        "encryptionType": "AES256"
      }
    }
  ]
}
[ec2-user@ip-10-0-1-151 environment]$
```



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

20. Next, to inspect the *cafe/node-web-app* image, run the following command in the VS Code bash terminal:

```
aws ecr describe-images --repository-name cafe/node-web-app
```

```
● [ec2-user@ip-10-0-1-151 environment]$ aws ecr describe-images --repository-name cafe/node-web-app
{
  "imageDetails": [
    {
      "registryId": "476149723783",
      "repositoryName": "cafe/node-web-app",
      "imageDigest": "sha256:5216e80aea58d10f7321e9613eee703633ef6c7943d4579bcff953d65d7ec82a",
      "imageTags": [
        "latest"
      ],
      "imageSizeInBytes": 29443630,
      "imagePushedAt": "2025-08-15T02:51:34.184000+00:00",
      "imageManifestMediaType": "application/vnd.docker.distribution.manifest.v2+json",
      "artifactMediaType": "application/vnd.docker.container.image.v1+json"
    }
  ]
}
○ [ec2-user@ip-10-0-1-151 environment]$
```

Ln 9, Col 23 Spaces: 4 UTF-8 LF Plain Text Layout: US

Task 5: Configuring communication between the container and the database

21. Start your container.

- In the VS Code bash terminal, enter the following command. Replace <db-endpoint> with the **Database endpoint** value from your text editor.

```
docker run -d --name node-web-app-1 -p 8000:3000 -e APP_DB_HOST="supplierdb-instance-1.czb71vvete2w.us-east-1.rds.amazonaws.com" cafe/node-web-app
```

```
● [ec2-user@ip-10-0-1-151 environment]$ docker run -d --name node-web-app-1 -p 8000:3000 -e APP_DB_HOST="supplierdb-instance-1.czb71vvete2w.us-east-1.rds.amazonaws.com" cafe/node-web-app
5dcff3c736e387d6ccf50a5b20b265704acd78b4486ae09f1807fafef93dbf597
○ [ec2-user@ip-10-0-1-151 environment]$
```

22. To test the container application from the VS Code bash terminal, run the following command:

```
curl http://localhost:8000
```



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

```
● [ec2-user@ip-10-0-1-151 environment]$ curl http://localhost:8000
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="/css/bootstrap.min.css">
  <link rel="stylesheet" href="/css/base.css">
  <title>Coffee suppliers</title>
</head>
<body>

<div class="container">
  <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
    
    <div><a class="navbar-brand page-title" href="/supplier">Coffee suppliers</a></div>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav mr-auto">
        <li class="nav-item active">
          <a class="nav-link" href="/">Home</a>
          <a class="nav-link" href="/suppliers">Suppliers list</a>
        </li>
      </ul>
    </div>
  </nav>  <div class="container">
    <h1>Welcome</h1>
    <p>Use this app to keep track of your coffee suppliers</p>
    <p><a href="/suppliers">List of suppliers</a></p>
  </div>
</div>
<script src="/js/jquery-3.6.0.min.js"></script>
<script src="/js/bootstrap.min.js"></script>
</body>
○ [ec2-user@ip-10-0-1-151 environment]$
```

Ln 7, Col 82 (62 selected) Spaces: 4 UTF-8 LF Plain Text Layout: US

23. Configure the security group of the VS Code IDE EC2 instance.

As in a previous lab, you need to add a rule so that you can view the website on port 80:

- Return to the AWS Management Console browser tab.
- In the search box to the right of **Services**, search for and choose **EC2**.
- In the left navigation pane, choose **Security Groups**.
- Select the checkbox for the security group that has *Lab IDE SG* in the name.
- In the lower pane, choose the **Inbound rules** tab.
- Choose **Edit inbound rules**.
- Add an entry for the application port:
 - Choose **Add rule**
 - **Type:** Choose **Custom TCP**
 - **Port range:** **8000**
 - **Source:** Choose **My IP**

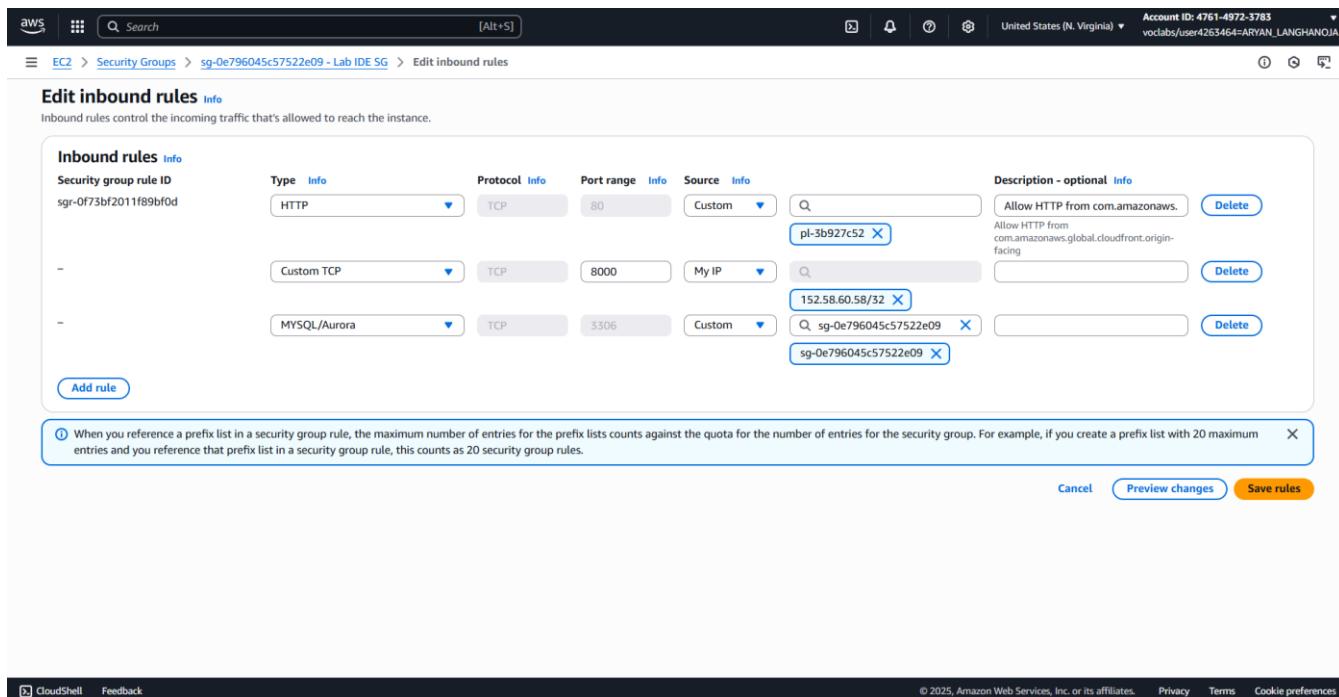
Add an entry for the database port:

- Choose **Add rule**
- **Type:** Choose **MYSQL/Aurora**
- **Source:** Choose **Custom**
- From the **Source** search box, choose the *Lab IDE SG* security group that you are currently editing.

 Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology <small>Marwadi Chandarana Group</small>		
Subject: Cloud Developing (01CT0720)		Aim: Run containers on a managed service.
Experiment No: 08	Date:	Enrolment No: 92200133030

This rule is self referencing to allow any call that originates from this security group to communicate with the database.

- Choose **Save rules**.



Inbound rules Info

Security group rule ID	Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>
sgr-0f73bf2011f89bf0d	HTTP	TCP	80	Custom	Allow HTTP from com.amazonaws. Delete
-	Custom TCP	TCP	8000	My IP	Allow HTTP from com.amazonaws.global.cloudfront.origin-facing Delete
-	MySQL/Aurora	TCP	3306	Custom	Allow MySQL/Aurora traffic from sg-0e796045c57522e09 Delete

[Add rule](#)

When you reference a prefix list in a security group rule, the maximum number of entries for the prefix lists counts against the quota for the number of entries for the security group. For example, if you create a prefix list with 20 maximum entries and you reference that prefix list in a security group rule, this counts as 20 security group rules.

[Cancel](#) [Preview changes](#) [Save rules](#)

24. Locate the public IP address of your VS Code IDE.
- In the left navigation pane, choose Instances.
 - Choose the Lab IDE.
 - In the bottom pane, choose the Details tab, and copy the Public IPv4 address value to your clipboard.

25. In a new browser tab, paste the `\http://<<Public IPv4 address:8000>>` in the address bar.

Note: The port being used here is 8000. Ensure that the protocol is http (and not https).
The coffee suppliers web application displays.

Task 6: Creating the application database objects

26. Connect to the Amazon RDS query editor.
- Return to the AWS Management Console browser tab.
 - In the search box to the right of **Services**, search for and choose **RDS**.
 - In the left navigation pane, choose **Query Editor**, and configure the following:
 - Database instance or cluster:** Choose **supplierdb**
 - Database username:** Choose **Add new database credentials**
 - Enter database username:** Enter admin
 - Enter database password:** Enter coffee_beans_for_all



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

- **Enter the name of the database or schema - (optional):** Enter suppliers
- Choose **Connect to database**.

Note: If you receive an error message, choose **Connect to database** again and the connection should be successful.

Connect to database



You need to choose a database and enter the database credentials to use the query editor. We will be storing your credentials and the connection in the AWS Secrets Manager service. [Learn more](#)

Database instance or cluster

supplierdb



Database username

Add new database credentials



Enter database username

admin

Enter database password

.....

Enter the name of the database or schema - (optional)

suppliers

Query statement terminator

;



[Cancel](#)

Connect to database



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

27. Create the database objects that support the coffee suppliers application.

- On the Editor tab, delete the contents of the text area, and paste in the following code:

```
CREATE USER "nodeapp" IDENTIFIED WITH mysql_native_password BY
"coffee";
CREATE DATABASE COFFEE;
USE COFFEE;
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, PROCESS,
REFERENCES, INDEX, ALTER, SHOW DATABASES, CREATE TEMPORARY TABLES, LOCK
TABLES, EXECUTE, REPLICATION SLAVE, REPLICATION CLIENT, CREATE VIEW,
SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, CREATE USER, EVENT, TRIGGER
ON *.* TO 'nodeapp'@'%' WITH GRANT OPTION;
CREATE TABLE suppliers(
    id INT NOT NULL AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    address VARCHAR(255) NOT NULL,
    city VARCHAR(255) NOT NULL,
    state VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    phone VARCHAR(100) NOT NULL,
    PRIMARY KEY ( id ));
```

- Choose Run.

Output

Statements (5)

[Export to csv](#)

Search rows

< 1 > |

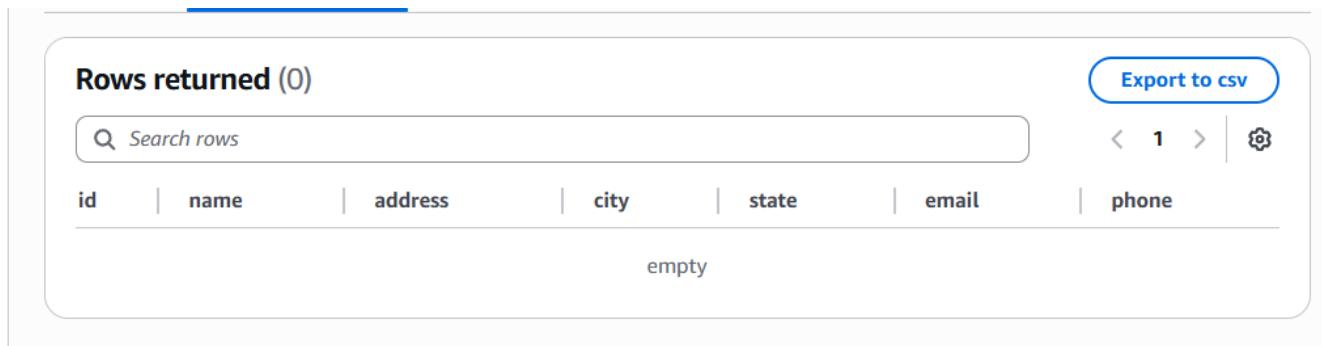
ID	Start	Statement
1	9:19:48	CREATE USER "nodeapp" IDENTIFIED WITH mysql_native_password BY "coffee"
2	9:19:49	CREATE DATABASE COFFEE
3	9:19:49	USE COFFEE
4	9:19:50	GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, RELOAD, PROCESS, REFERENCES, INDEX, ALTE
5	9:19:50	CREATE TABLE suppliers(id INT NOT NULL AUTO_INCREMENT, name VARCHAR(255) NOT NULL, address

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

28. Verify that the suppliers table is empty.

- Replace the contents in the text area with the following code:

```
use COFFEE;
select * from suppliers
```
- Choose Run.



The screenshot shows a table titled "Rows returned (0)" with a search bar and export options. The table has columns: id, name, address, city, state, email, and phone. A message "empty" is displayed below the table.

Task 7: Seeding the database with supplier data

29. Review the supplier data.

- Return to the VS Code IDE browser tab.
- To change the directory to resources, run the following command:
`cd ~/environment/resources`
- From the Environment pane in IDE, open the resources/coffee_db_dump.sql file and review its contents.

30. Connect to the database.

- Run the following command in the terminal window. Replace <db-endpoint> with the **Database endpoint** value from your text editor.
`mysql -h <db-endpoint> -P 3306 -u admin -p`
- When prompted for the password, enter coffee_beans_for_all
- The output looks similar to the following:

```
[ec2-user@ip-10-0-1-151 environment]$ cd ~/environment/resources
[ec2-user@ip-10-0-1-151 resources]$ mysql -h supplierdb-instance-1.czb7lvvete2w.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 270
Server version: 8.0.36 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)]>
```

31. To verify that you are connected to the correct database, run the following command:

```
use COFFEE; select * from suppliers;
```

 <p>Marwadi University Marwadi Chandarana Group</p>	<p>Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology</p>	
<p>Subject: Cloud Developing (01CT0720)</p>	<p>Aim: Run containers on a managed service.</p>	
<p>Experiment No: 08</p>	<p>Date:</p>	<p>Enrolment No: 92200133030</p>

```
MySQL [(none)]> use COFFEE; select * from suppliers;
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
Empty set (0.001 sec)

MySQL [COFFEE] >

Ln 10, Col 1 Spaces: 4 UTF-8 LF Plain Text Layout: US

32. To use the SQL dump to update your database with the information from the supplier, run the following command at the mysql prompt:

```
source coffee_db_dump.sql
```

```
MySQL [COFFEE]> source coffee_db.dump.sql
Query OK, 0 rows affected (0.004 sec)

Query OK, 0 rows affected (0.001 sec)

Database changed
Query OK, 0 rows affected (0.108 sec)

Query OK, 0 rows affected (0.001 sec)

Query OK, 0 rows affected (0.001 sec)

Query OK, 0 rows affected (0.180 sec)

Query OK, 0 rows affected (0.001 sec)

Query OK, 0 rows affected (0.013 sec)
```

Ln 10, Col 1 Spaces: 4 UTF-8 LF Plain Text Layout: US

33. Review the data that was loaded into the database.

- To query the suppliers table, run the following command at the mysql prompt:

```
use COFFEE; select * from suppliers;
```



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

```
MySQL [COFFEE]> use COFFEE; select * from suppliers;
Database changed
```

id name	address	city	state	email	phone
1 AnyCompany coffee suppliers	123 Any Street	Any Town	WA	info@example.com	555-555-0100
2 Central Example Corp. coffee	100 Main Street	Nowhere	CO	info@example.net	555-555-0101
3 North East AnyCompany coffee suppliers	1001 Main Street	Any Town	NY	info@example.co	555-555-0102
4 SE Example corp coffee suppliers	200 1st street	None city	GA	info@example.org	555-555-0103
5 SW Example Corp. coffee	333 Main st	Anytown	AZ	info@example.me	555-555-0104
6 Northern Example Corp. coffee	444 Main st	Not town	MN	coffee@example.com	555-555-0106
7 West coast example Corp. coffee	1212 Se 30th Ave	Any beach	CA	coffee@example.coffee	555-555-0107
8 Southern AnyCompany coffee suppliers	555 Main st	Anytown	TX	coffee@example.biz	555-555-0108

8 rows in set (0.004 sec)

```
MySQL [COFFEE]>
```

- To query the *beans* table, run the following command at the mysql prompt:
select * from beans;

```
MySQL [COFFEE]> select * from beans;
```

id supplier_id type	product_name	price	description	quantity
1 1 Arabica	Best bean	18.00	Delicious, smooth coffee.	1000
2 1 Robusta	Great bean	12.00	Full bodied, good to the last drop.	800
3 2 Robusta	Top bean	10.00	Great all around bean.	500
4 2 Liberica	Better bean	14.00	This bean stands above the rest.	600
5 3 Excelsa	Premiere bean	18.00	The best bean in all the land	200
6 4 Arabica	House bean	11.00	A solid performer.	900
7 4 Robusta	Quality bean	13.00	A great bean for daily use.	350
8 5 Robusta	Superb bean	16.00	No bean is better	700
9 5 Liberica	Top tier bean	15.00	The bean that impresses.	300
10 6 Arabica	Stellar bean	13.00	The top star of beans	300
11 7 Robusta	Terrific bean	12.00	This is a great bean	800
12 7 Liberica	Supreme bean	17.00	Solid performing bean. Light roast for smooth taste.	700
13 8 Liberica	Ace bean	10.00	Medium roast bean. Good for brewed coffee.	1000
14 8 Excelsa	Unrivaled bean	16.00	Dark roast bean. Best for espresso.	300

14 rows in set (0.023 sec)

```
MySQL [COFFEE]>
```

Task 8: Review the IAM policy and role for Elastic Beanstalk

34. Review the Elastic Beanstalk IAM policy that will be used with your Docker environment.

- Return to the AWS Management Console browser tab.
- From the **Services** menu, choose **IAM**.
- In the left navigation pane, choose **Policies**.
- In the search text box, enter **aws-elasticbeanstalk-ec2-instance-policy** then select the hyperlink of the policy.

 Marwadi University <small>Marwadi Chandarana Group</small>	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

Task 9: Creating an Elastic Beanstalk application

35. Create a sample Elastic Beanstalk application.

- Return to the VS Code IDE browser tab.
- To change to the *environment* directory, run the following command:

```
cd ~/environment
```

- To create a new folder called *bean*, run the following command:

```
mkdir bean
```

- To change to the *bean* directory, run the following command:

```
cd bean
```

- To create a sample application named *MyNodeApp*, run the following command:

```
aws elasticbeanstalk create-application --application-name MyNodeApp
```

```
[ec2-user@ip-10-0-1-151 bean]$ aws elasticbeanstalk create-application --application-name MyNodeApp
{
  "Application": {
    "ApplicationArn": "arn:aws:elasticbeanstalk:us-east-1:476149723783:application/MyNodeApp",
    "ApplicationName": "MyNodeApp",
    "DateCreated": "2025-08-15T04:06:47.645000+00:00",
    "DateUpdated": "2025-08-15T04:06:47.645000+00:00",
    "ConfigurationTemplates": [],
    "ResourceLifecycleConfig": {
      "VersionLifecycleConfig": {
        "MaxCountRule": {
          "Enabled": false,
          "MaxCount": 200,
          "DeleteSourceFromS3": false
        },
        "MaxAgeRule": {
          "Enabled": false,
          "MaxAgeInDays": 180,
          "DeleteSourceFromS3": false
        }
      }
    }
  }
}
[ec2-user@ip-10-0-1-151 bean]$
```

36. Create an Elastic Beanstalk environment.

- In VS Code IDE, in the *bean* folder, create a new file named *options.txt* and open it.
- Paste the following text into the *options.txt* file:



Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

37. To identify the currently available solution stack for Amazon Linux 2, run the following command in the VS Code bash terminal:

```
aws elasticbeanstalk list-available-solution-stacks|grep 'running Docker'
```

```
● [ec2-user@ip-10-0-1-151 bean]$ aws elasticbeanstalk list-available-solution-stacks|grep 'running Docker'  
"64bit Amazon Linux 2023 v4.6.3 running Docker",  
"64bit Amazon Linux 2 v4.2.3 running Docker",  
"SolutionStackName": "64bit Amazon Linux 2023 v4.6.3 running Docker",  
"SolutionStackName": "64bit Amazon Linux 2 v4.2.3 running Docker",
```

- Run the following command. Replace the <solution-stack-name> placeholder with the name of the available solution stack:

```
aws elasticbeanstalk create-environment --application-name MyNodeApp --environment-name MyEnv --solution-stack-name "<solution-stack-name>" --region us-east-1 --option-settings file://options.txt
```

```
● [ec2-user@ip-10-0-1-151 bean]$ aws elasticbeanstalk create-environment --application-name MyNodeApp --environment-name MyEnv --solution-stack-name "64bit Amazon Linux 2 v4.2.3 running Docker" --region us-east-1 --option-settings file://options.txt  
{  
    "EnvironmentName": "MyEnv",  
    "EnvironmentId": "e-mvyszgppi3",  
    "ApplicationName": "MyNodeApp",  
    "SolutionStackName": "64bit Amazon Linux 2 v4.2.3 running Docker",  
    "PlatformArn": "arn:aws:elasticbeanstalk:us-east-1::platform/Docker running on 64bit Amazon Linux 2/4.2.3",  
    "DateCreated": "2025-08-15T04:30:31.321000+00:00",  
    "DateUpdated": "2025-08-15T04:30:31.321000+00:00",  
    "Status": "Launching",  
    "...skipping..."  
    {  
        "EnvironmentName": "MyEnv",  
        "EnvironmentId": "e-mvyszgppi3",  
        "ApplicationName": "MyNodeApp",  
        "SolutionStackName": "64bit Amazon Linux 2 v4.2.3 running Docker",  
        "PlatformArn": "arn:aws:elasticbeanstalk:us-east-1::platform/Docker running on 64bit Amazon Linux 2/4.2.3",  
        "DateCreated": "2025-08-15T04:30:31.321000+00:00",  
        "DateUpdated": "2025-08-15T04:30:31.321000+00:00",  
        "Status": "Launching",  
        "Health": "Grey",  
        "Tier": {  
            "Name": "WebServer",  
            "Type": "Standard",  
            "Version": "1.0"  
        },  
        "EnvironmentArn": "arn:aws:elasticbeanstalk:us-east-1:476149723783:environment/MyNodeApp/MyEnv"  
    }  
~  
~
```

42. To stop the Docker container that is running on the VS Code IDE, run the following command:

```
docker stop node-web-app-1 && docker rm node-web-app-1
```

```
● [ec2-user@ip-10-0-1-151 bean]$ docker stop node-web-app-1  
node-web-app-1  
node-web-app-1  
○ fec2-user@ip-10-0-1-151 bean$ █
```

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

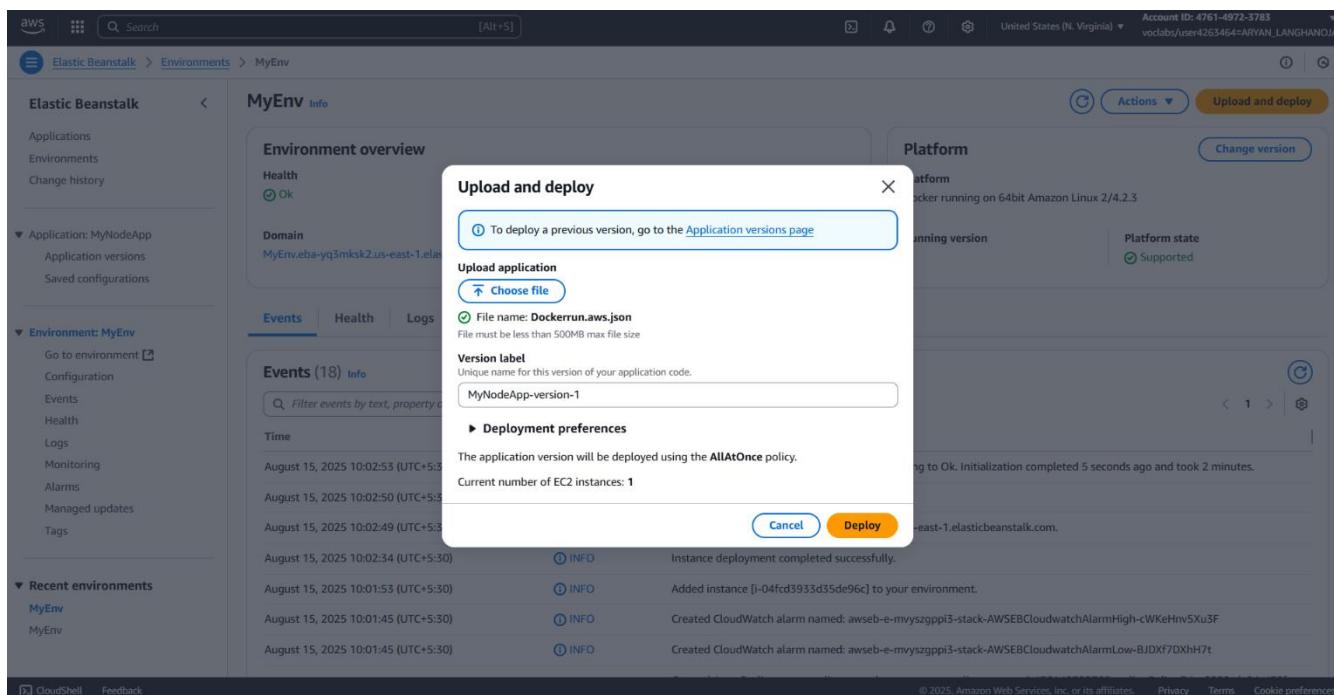
43. Review your Elastic Beanstalk environment in the AWS Management Console.

- Return to the AWS Management Console tab.
- From the **Services** menu, choose **Elastic Beanstalk**.
- For the **MyEnv** environment, the **Health** displays *Pending*.
- At the bottom under the **Events** tab.

44. Update the application with the code for the coffee suppliers application.

- The code is in the image that you uploaded to Amazon ECR. For now, you will deploy the application to Elastic Beanstalk manually. In a later lab, you will learn to automate deployments.
- On your local computer, create a new text file called *Dockerrun.aws.json*.
- Paste the following text into the file:

```
{
  "AWSEBDockerrunVersion": "1",
  "Image": {
    "Name": "<FMI_1>",
    "Update": "true"
  },
  "Ports": [ { "ContainerPort" : 3000 } ]
```

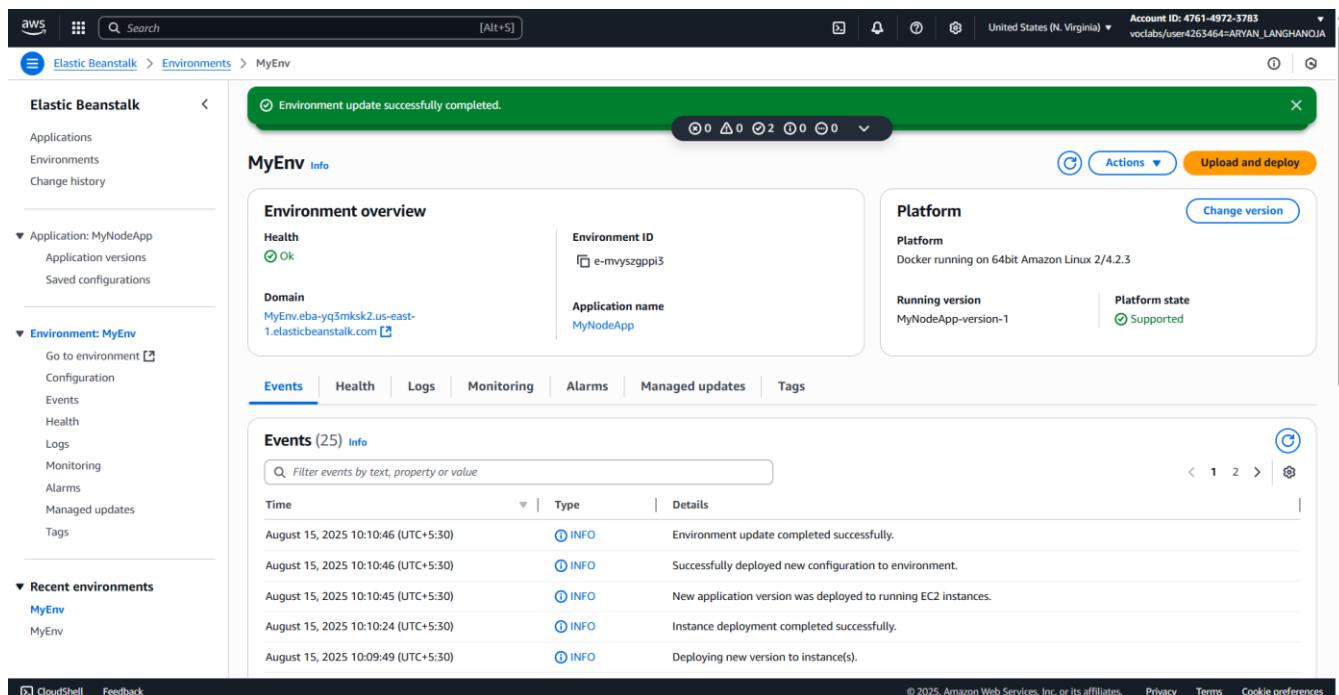


The screenshot shows the AWS Elastic Beanstalk console with the 'MyEnv' environment selected. The left sidebar shows the environment configuration, including applications, environments, and recent environments. The main area displays the 'Environment overview' with a 'Health' status of 'Ok'. A central modal window titled 'Upload and deploy' is open, prompting the user to upload an application version. The file 'Dockerrun.aws.json' has been selected. The 'Version label' field contains 'MyNodeApp-version-1'. Under 'Deployment preferences', the policy is set to 'AllAtOnce'. The background shows the deployment history with one instance currently running.

 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

45. In the file, replace the <FMI_1> placeholder with the **Repository URI** value from your text editor.

- Save the changes to the file.
- Return to the AWS Management Console browser tab.
- On the **MyEnv** environment page within the Elastic Beanstalk console, choose **Upload and deploy**.
- Navigate to and choose the *Dockerrun.aws.json* file.
- For **Version label**, append the letter a to the default MyNodeApp-version-1
- Keep the current deployment preferences.
- Choose **Deploy** and observe the deployment process.
- If you want to follow what is happening, observe the **Recent events** section.



The screenshot shows the AWS Elastic Beanstalk Environment Overview page for 'MyEnv'. At the top, a green banner displays 'Environment update successfully completed.' Below it, the 'MyEnv' environment info card shows the environment ID (e-mvyszgppi3), application name (MyNodeApp), and domain (MyEnv.eba-yq3mksk2.us-east-1.elasticbeanstalk.com). The 'Platform' section indicates Docker running on 64bit Amazon Linux 2/4.2.3. The 'Running version' is MyNodeApp-version-1, and the 'Platform state' is Supported. The 'Events' tab is selected, showing a list of 25 recent events. The first event is 'Environment update completed successfully' on August 15, 2025, at 10:10:46 UTC+5:30. Other events include 'Successfully deployed new configuration to environment.', 'New application version was deployed to running EC2 instances.', 'Instance deployment completed successfully.', and 'Deploying new version to instance(s.)'. The left sidebar shows navigation links for Applications, Environments, Change history, Application: MyNodeApp, Configuration, Saved configurations, Environment: MyEnv, Go to environment, Configuration, Events, Health, Logs, Monitoring, Alarms, Managed updates, Tags, and Recent environments (MyEnv).

Task 10: Configuring the API Gateway proxy

46. Create an API Gateway resource.

- Return to the AWS Management Console browser tab.
- From the **Services** menu, choose **API Gateway**.
- Choose the **ProductsApi** hyperlink.
- Choose **Create resource**, and configure the following:
 1. **Resource Path:** Keep the default / selection
 2. **Resource Name:** Enter bean_products

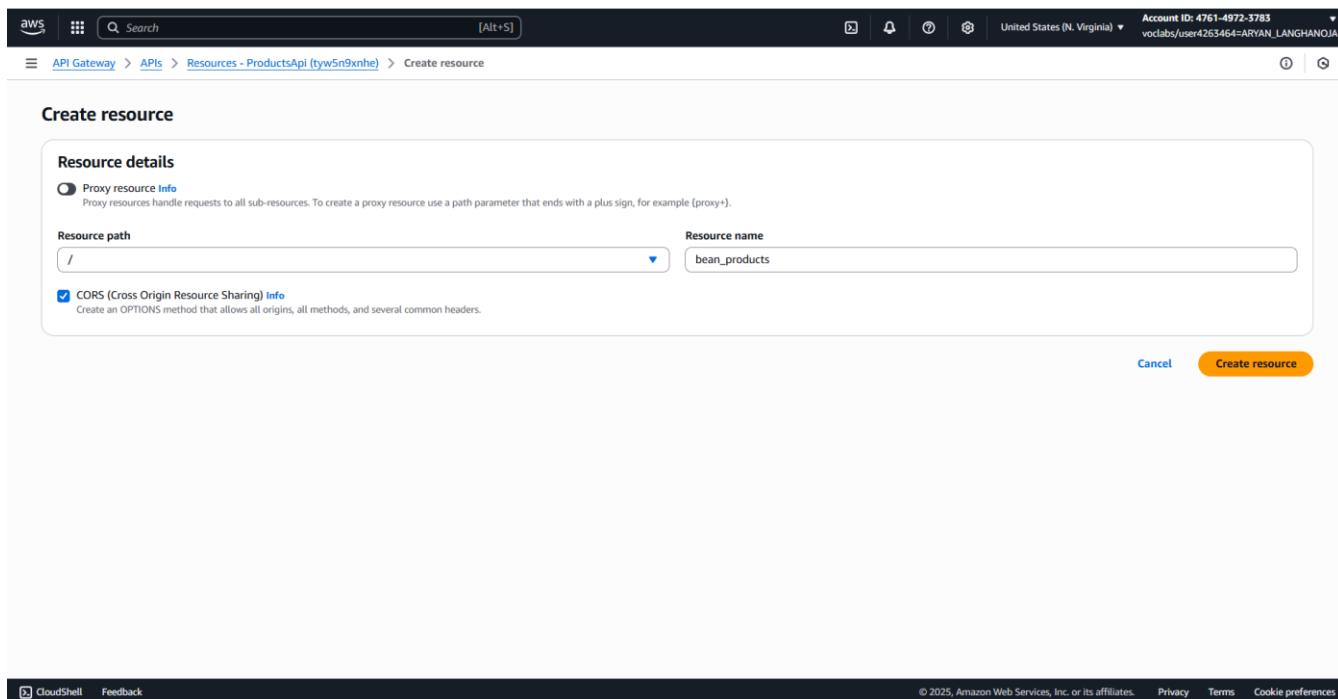
 Marwadi University Marwadi Chandarana Group	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

 **Note:** Ensure the resource name value has an underscore.

3. Select CORS (Cross Origin Resource Sharing)

 **Note:** CORS is required to enable communication between your website and the endpoint.

4. Choose Create resource.



The screenshot shows the AWS API Gateway 'Create resource' interface. The 'Resource details' section is open, showing the following configuration:

- Proxy resource** is selected.
- Resource path** is set to '/'.
- Resource name** is set to 'bean_products'.
- CORS (Cross Origin Resource Sharing)** is checked.

At the bottom right of the dialog, there are 'Cancel' and 'Create resource' buttons. The 'Create resource' button is highlighted with a yellow background.

47. Create an API Gateway method.

- In the Methods panel, choose **Create method**.
- For **Method type** choose **GET**.
- Configure the following:
 - Integration type:** Choose **HTTP**
 - Toggle **HTTP proxy integration** to **on**
 - For **HTTP method** choose **GET**
 - For **Endpoint URL**, enter the following. Replace the <FMI_1> placeholder with the Elastic Beanstalk URL from your text editor:

URL :- http://myenv.eba-yq3mksk2.us-east-1.elasticbeanstalk.com/beans.json



Marwadi University
Faculty of Engineering and Technology
Department of Information and Communication Technology

Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

- Choose Create method.
- Choose the Test tab.

Request	Latency ms	Status
/bean_products	111	200

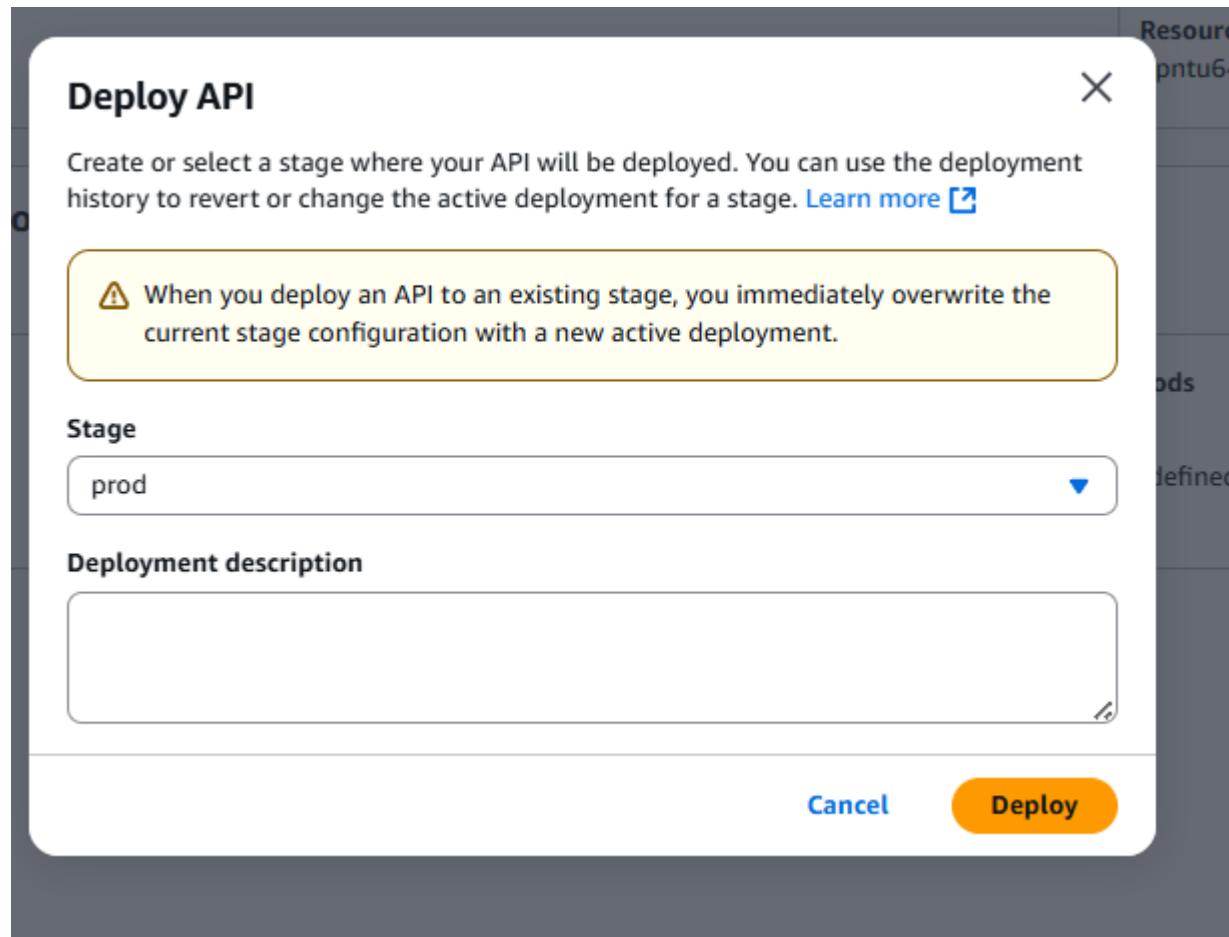
```

Response body
[
  {
    "id": 1,
    "supplier_id": 1,
    "type": "Arabica",
    "product_name": "Best bean",
    "price": "18.00",
    "description": "Delicious, smooth coffee.",
    "quantity": 1000
  },
  {
    "id": 2,
    "supplier_id": 1,
    "type": "Robusta",
    "product_name": "Great bean",
    "price": "12.00",
    "description": "Full bodied, good to the last drop.",
    "quantity": 800
  },
  {
    "id": 3,
    "supplier_id": 2,
    "type": "Robusta",
    "product_name": "Top bean",
    "price": "10.00",
    "description": "Great all around bean.",
    "quantity": 500
  },
  {
    "id": 4,
    "supplier_id": 2,
    "type": "Liberica",
    "product_name": "Better bean",
    "price": "14.00",
    "description": "This bean stands above the rest.",
    "quantity": 600
  }
]
  
```

 Marwadi University <small>Marwadi Chandarana Group</small>	Marwadi University Faculty of Engineering and Technology Department of Information and Communication Technology	
Subject: Cloud Developing (01CT0720)	Aim: Run containers on a managed service.	
Experiment No: 08	Date:	Enrolment No: 92200133030

48. Deploy the API changes.

- In the **Resources** pane, choose the top-level "/" resource.
- Choose **Deploy API**.
- For **Stage**, choose **prod**.
- Choose **Deploy**.





Subject: Cloud Developing (01CT0720)

Aim: Run containers on a managed service.

Experiment No: 08

Date:

Enrolment No: 92200133030

Conclusion:-

In this lab,

- I deployed the coffee suppliers app on AWS using Aurora Serverless for the database and Elastic Beanstalk for the web app.
- I downloaded the code,
- upgraded AWS CLI,
- set up VPC subnets,
- created the Aurora DB, and
- connected it to the Docker container.
- I created database tables, loaded sample data, and reviewed the ECR image.
- Then, I deployed the app to Elastic Beanstalk using Dockerrun.aws.json and linked it with API Gateway.
- The app now runs successfully with managed services, scalable, and accessible via an API endpoint.

Result :-

Total score	45/45
[TASK 2] Configured new subnet	5/5
[TASK 3] Created database	5/5
[TASK 5] Configured Lab IDE security group	5/5
[TASK 6] Created suppliers table	5/5
[TASK 7] Seeded supplier data	5/5
[TASK 9A] Created Elastic Beanstalk application	5/5
[TASK 9B] Deployed Elastic Beanstalk environment	5/5
[TASK 9C] Updated Elastic Beanstalk application	5/5
[TASK 10] Configured bean_products API resource	5/5