📘 Auto Scaling Group (ASG) Web Application Deployment Documentation

👨‍💻 **Author: EC2 GROUP**

🗓️ **Date: April 2025**

📌 Objective

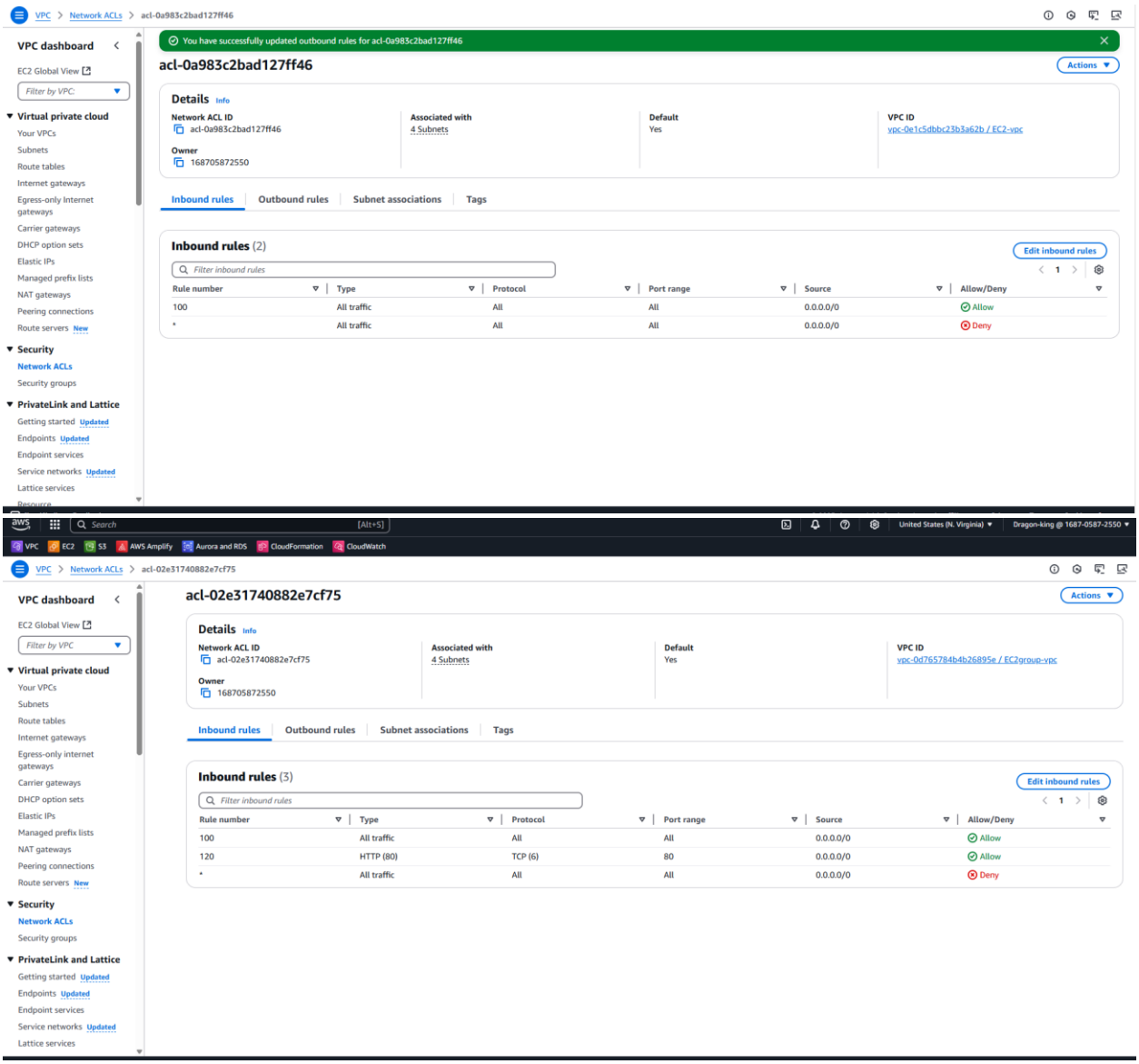
To document the step-by-step process of setting up an EC2 instance, configuring a launch template, Auto Scaling Group (ASG), and ensuring every instance launched by the ASG runs the same web application.

🔲Initial Setup

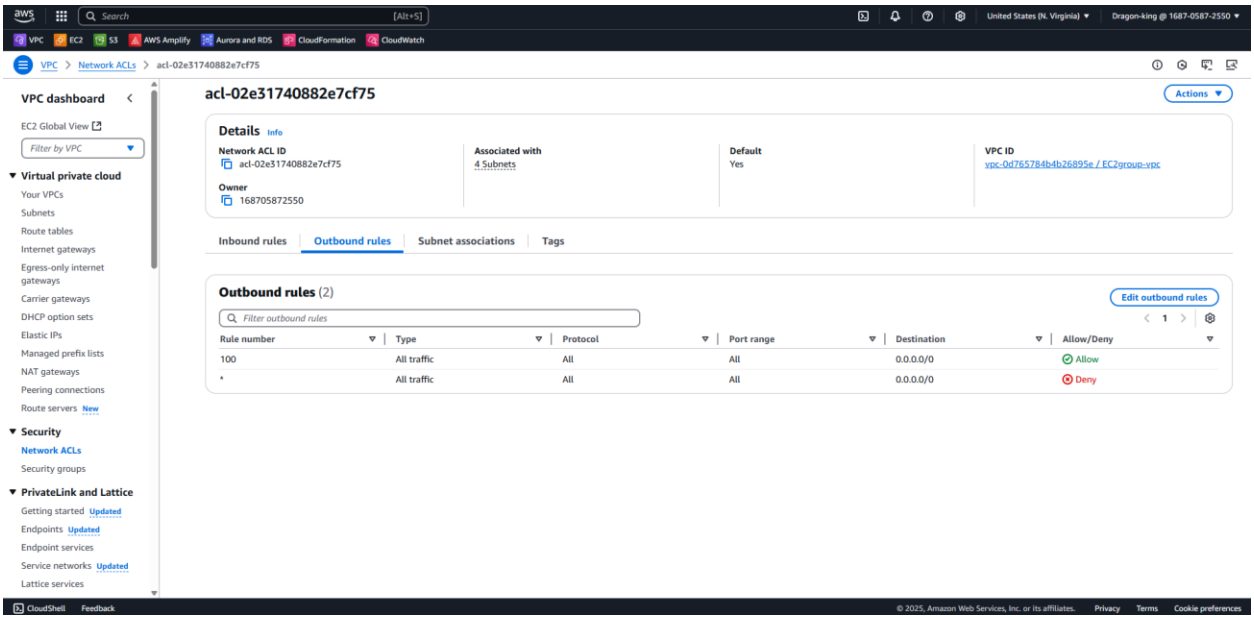
VPC and Network Setup

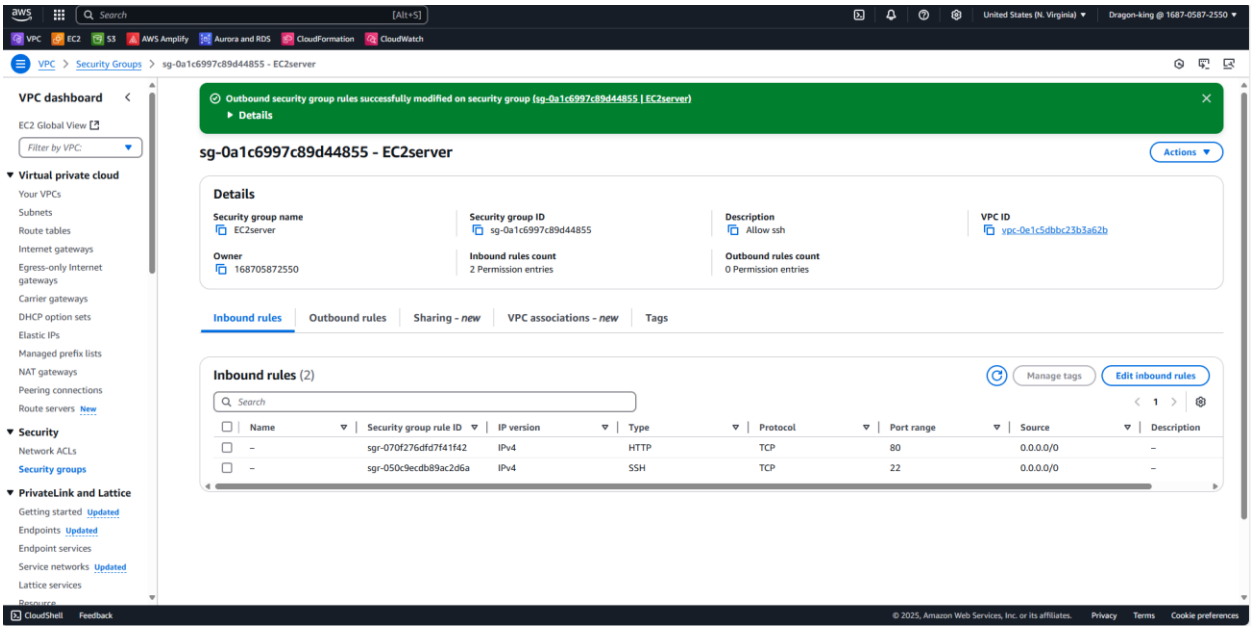
1. **VPC Creation**:
   A Virtual Private Cloud (VPC) was created to logically isolate the network for the application infrastructure.
2. **Subnets**:
   a. **Public Subnets**: Intended use, Load Balancer and ASG.
3. **Private Subnets**: Intended for use with EC2 and database.

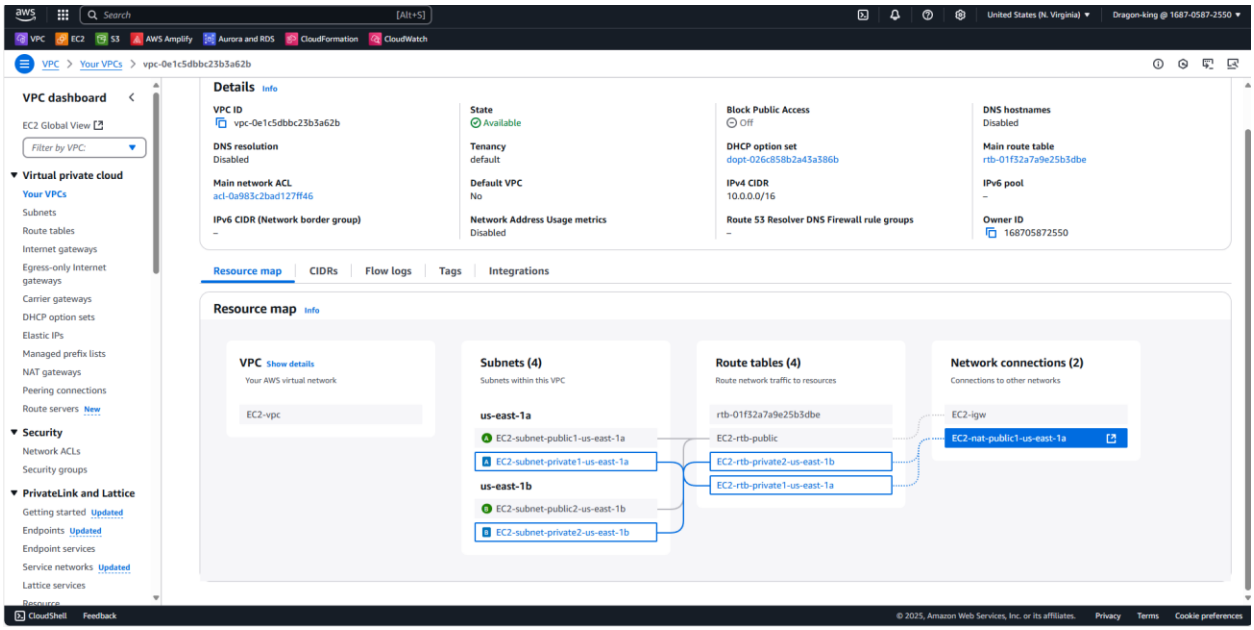
4. **Route Tables**:
   a. Public subnets are associated with a route table that includes a route to the Internet Gateway for outbound internet access.
   b. (Optional) Private subnets would typically use a NAT Gateway for outbound access, though not used here.
5. **Internet Gateway (IGW)**:
   Attached to the VPC to allow public subnet traffic to access the internet.
6. **Security Groups**:
   Configured to allow inbound HTTP (port 80), HTTPS (443), and SSH (22) access.
7. **Network Access Control Lists (NACLs)**:
   Set up to permit the same ports as security groups for both inbound and outbound traffic on public subnets.

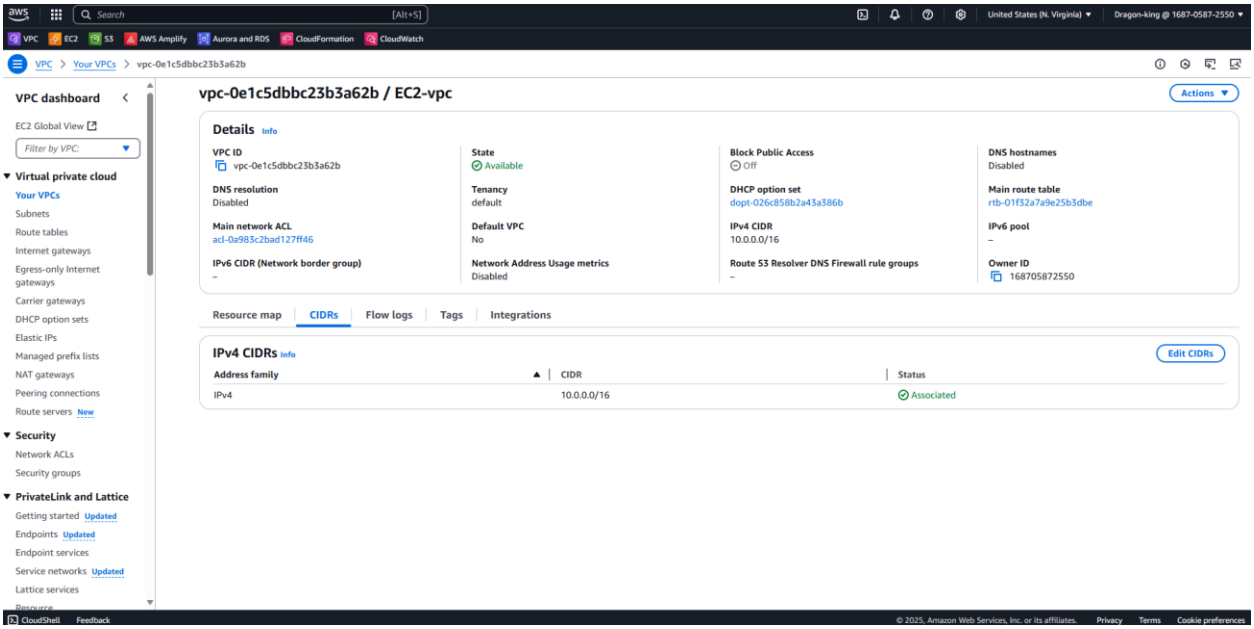

VPC > Network ACLs > acl-0a983c2bad127ff46
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started Updated
Endpoints Updated
Endpoint services
Service networks Updated
Lattice services
You have successfully updated outbound rules for acl-0a983c2bad127ff46
acl-0a983c2bad127ff46
Actions
Details Info
Network ACL ID
acl-0a983c2bad127ff46
Associated with
4 Subnets
Default
Yes
VPC ID
vpc-0e1c5dbbc23b3a62b / EC2-vpc
Owner
168705872550
Inbound rules
Outbound rules
Subnet associations
Tags
Inbound rules (2)
Edit inbound rules
Filter inbound rules
Rule number
Type
Protocol
Port range
Source
Allow/Deny
100
All traffic
All
All
0.0.0.0/0
Allow
*
All traffic
All
All
0.0.0.0/0
Deny

Search
[Alt+S]
United States (N. Virginia)
Dragon-king @ 1687-0587-2550
VPC
EC2
S3
AWS Amplify
Aurora and RDS
CloudFormation
CloudWatch
VPC > Network ACLs > acl-02e31740882e7cf75
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started Updated
Endpoints Updated
Endpoint services
Service networks Updated
Lattice services
acl-02e31740882e7cf75
Actions
Details Info
Network ACL ID
acl-02e31740882e7cf75
Associated with
4 Subnets
Default
Yes
VPC ID
vpc-0d765784b4b26895e / EC2group-vpc
Owner
168705872550
Inbound rules
Outbound rules
Subnet associations
Tags
Inbound rules (3)
Edit inbound rules
Filter inbound rules
Rule number
Type
Protocol
Port range
Source
Allow/Deny
100
All traffic
All
All
0.0.0.0/0
Allow
120
HTTP (80)
TCP (6)
80
0.0.0.0/0
Allow
*
All traffic
All
All
0.0.0.0/0
Deny


Search [Alt+S]
United States (N. Virginia)
Dragon-king @ 1687-0587-2550
VPC EC2 S3 AWS Amplify Aurora and RDS CloudFormation CloudWatch
VPC > Network ACLs > acl-02e31740882e7cf75
VPC dashboard
EC2 Global View
Filter by VPC
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started Updated
Endpoints Updated
Endpoint services
Service networks Updated
Lattice services
acl-02e31740882e7cf75
Actions
Details Info
Network ACL ID
acl-02e31740882e7cf75
Associated with
4 Subnets
Default
Yes
VPC ID
vpc-0d765784b4b26895e / EC2group-vpc
Owner
168705872550
Inbound rules
Outbound rules
Subnet associations
Tags
Outbound rules (2)
Edit outbound rules
Filter outbound rules
Rule number
Type
Protocol
Port range
Destination
Allow/Deny
100
All traffic
All
All
0.0.0.0/0
Allow
*
All traffic
All
All
0.0.0.0/0
Deny
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Search [Alt+S]
United States (N. Virginia)
Dragon-king @ 1687-0587-2550
VPC EC2 S3 AWS Amplify Aurora and RDS CloudFormation CloudWatch
VPC > Security Groups > sg-0a1c6997c89d44855 - EC2server
VPC dashboard
EC2 Global View
Filter by VPC:
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only Internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways
Peering connections
Route servers New
Security
Network ACLs
Security groups
PrivateLink and Lattice
Getting started Updated
Endpoints Updated
Endpoint services
Service networks Updated
Lattice services
Outbound security group rules successfully modified on security group (sg-0a1c6997c89d44855 | EC2server)
Details
sg-0a1c6997c89d44855 - EC2server
Actions
Details
Security group name
EC2server
Security group ID
sg-0a1c6997c89d44855
Description
Allow ssh
VPC ID
vpc-0e1c5dbbc23b3a62b
Owner
168705872550
Inbound rules count
2 Permission entries
Outbound rules count
0 Permission entries
Inbound rules
Outbound rules
Sharing - new
VPC associations - new
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
Description
–
sgr-070f276dfd7f41f42
IPv4
HTTP
TCP
80
0.0.0.0/0
–
–
sgr-050c9ecdb89ac2d6a
IPv4
SSH
TCP
22
0.0.0.0/0
–
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences





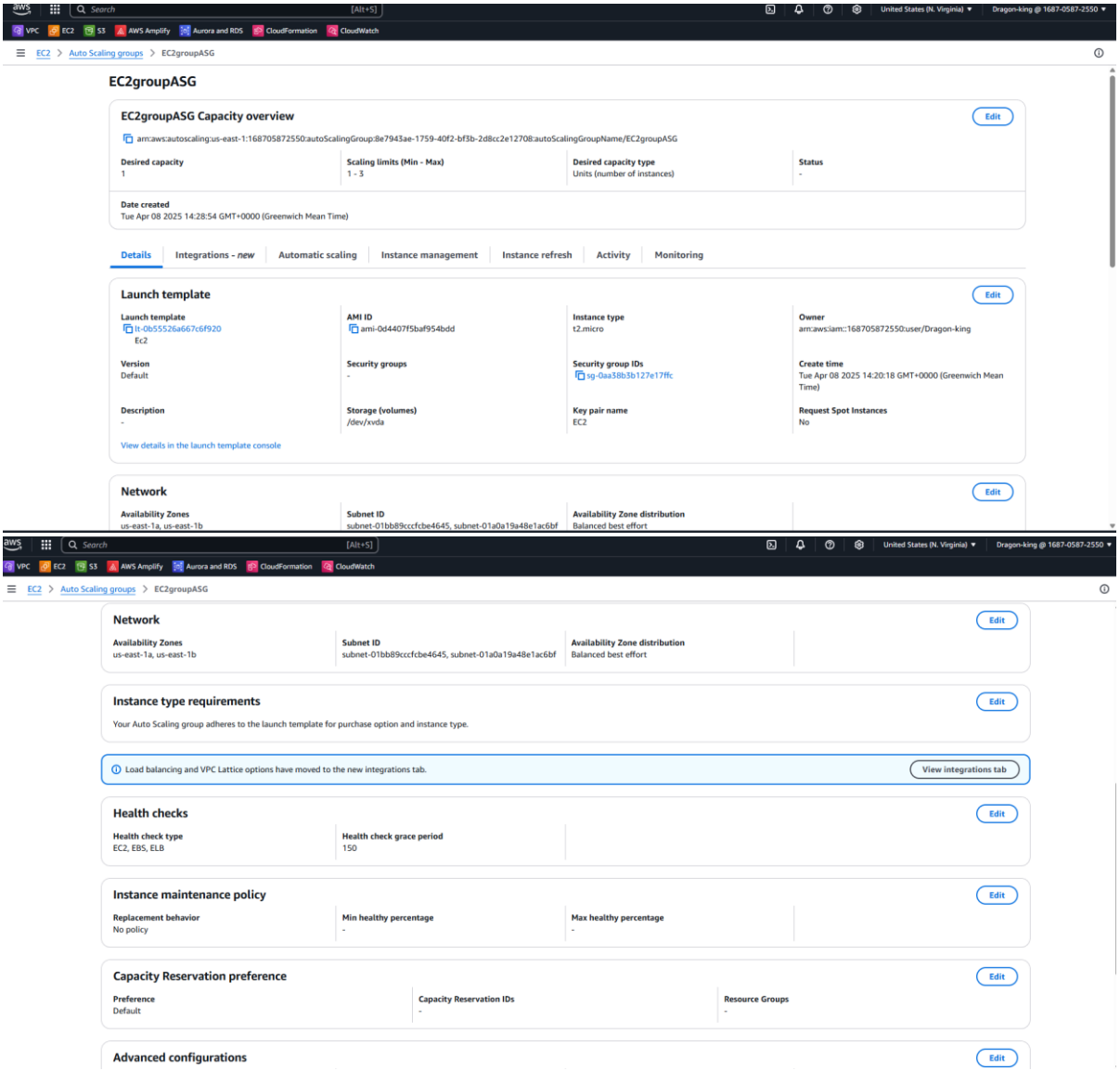
.

Auto Scaling Group Setup in the private subnets.

🔲Launch Template Creation

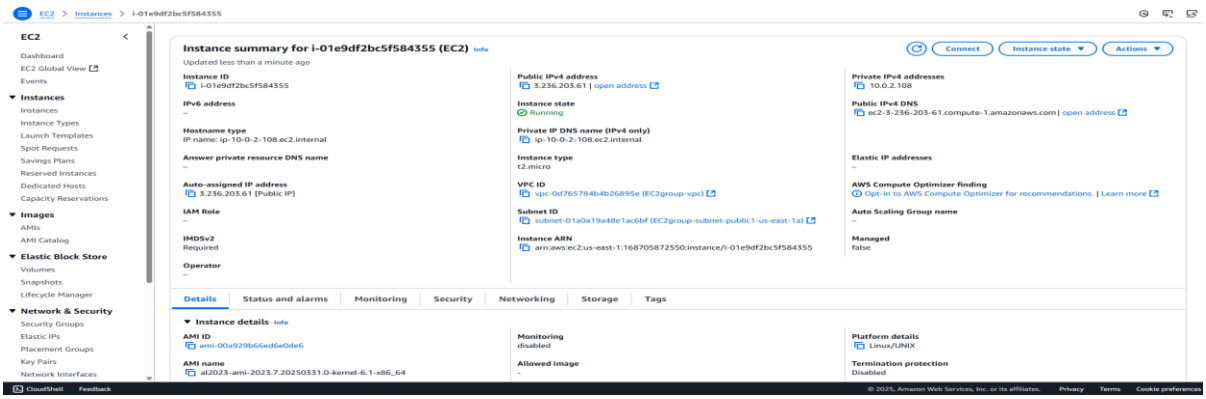
- A launch template was created based on the original EC2 instance.
- Includes instance type, AMI, security groups, and optionally User Data.

✅ ASG Linked to Launch Template

- **Screenshot:** EC2ASG.png, EC2ASG-2.png
- **Description:** Shows the Auto Scaling Group with its configuration attached to the launch template.
- 

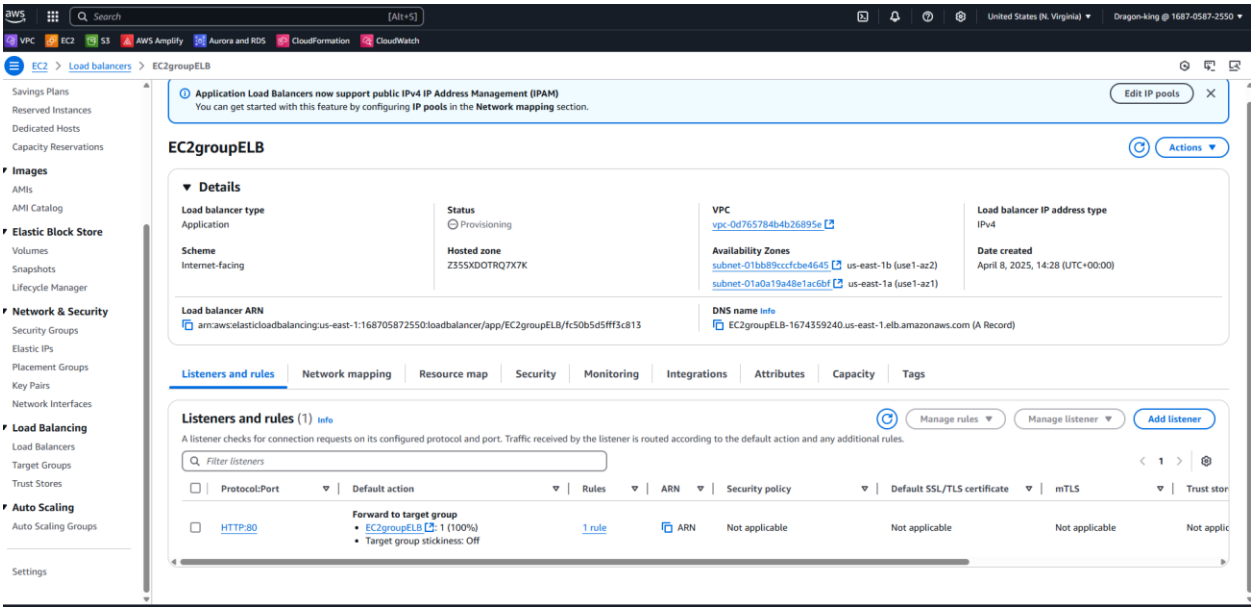
-

EC2 launched or spanned from ASG



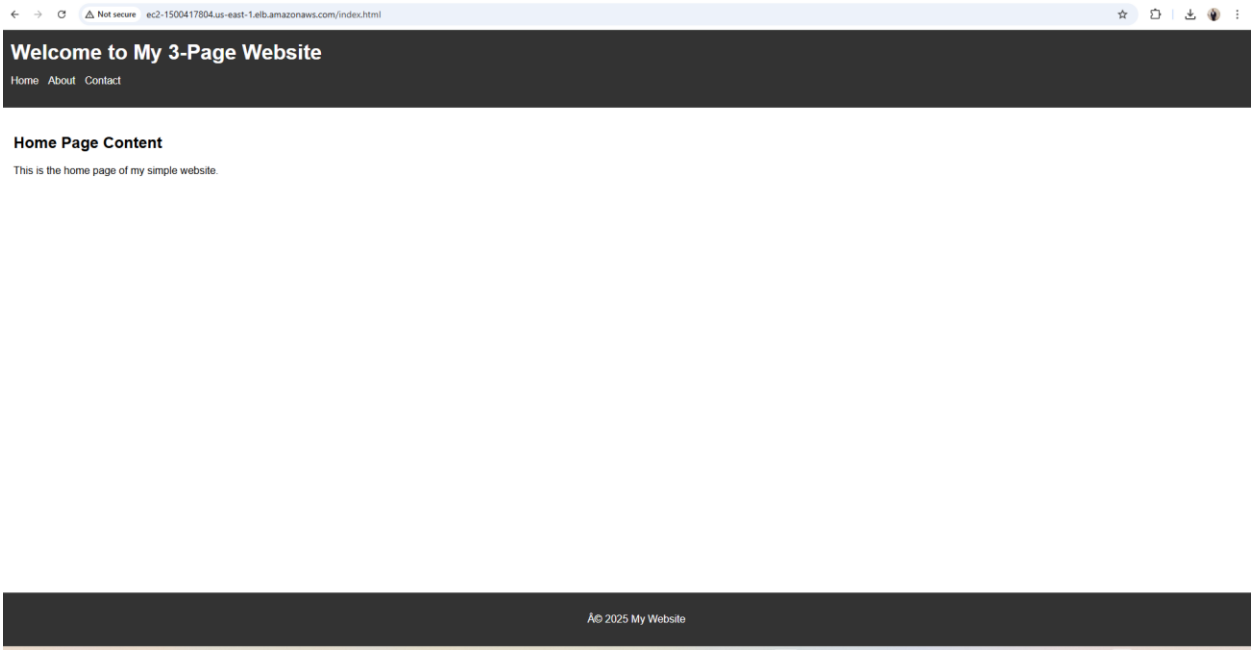
☑ Load Balancer Setup

- **Screenshot:** EC2GROUP-ELB-SETUP.png
- **Description:** Displays the Load Balancer forwarding requests to instances within the ASG.



Web Application Accessible via Elastic Load Balancer DNS

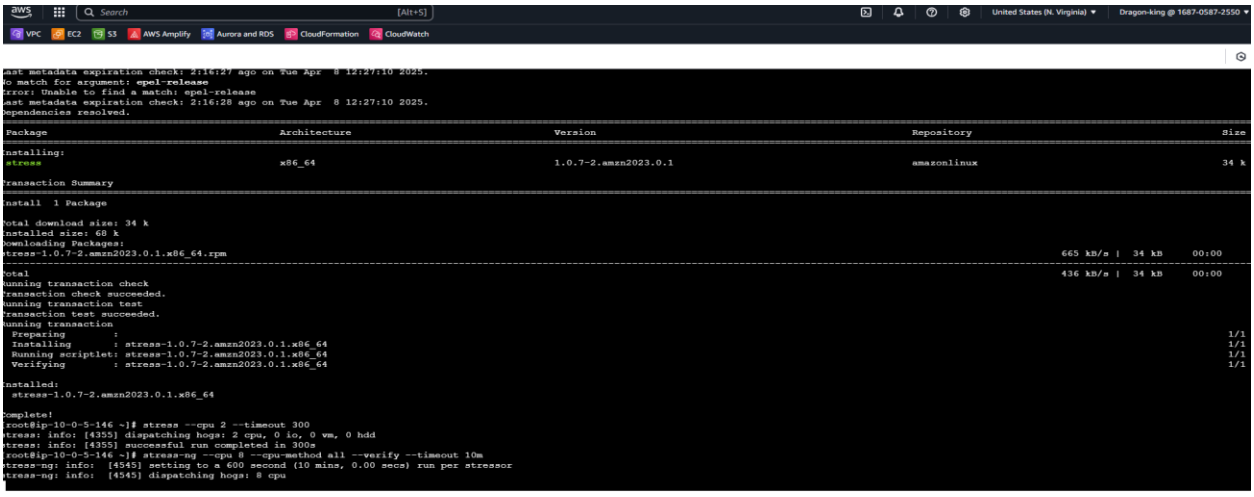
The app was uploaded through user date during Launch template setup or configuration.



□□CPU Stress Testing

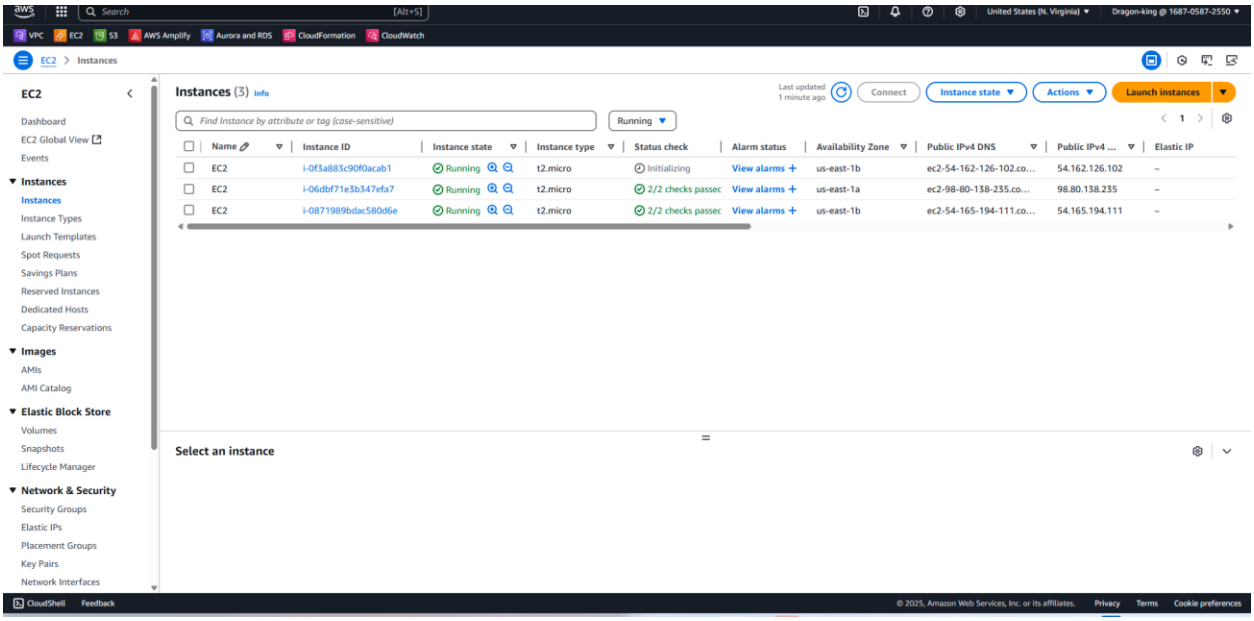
✅ Test CPU Load

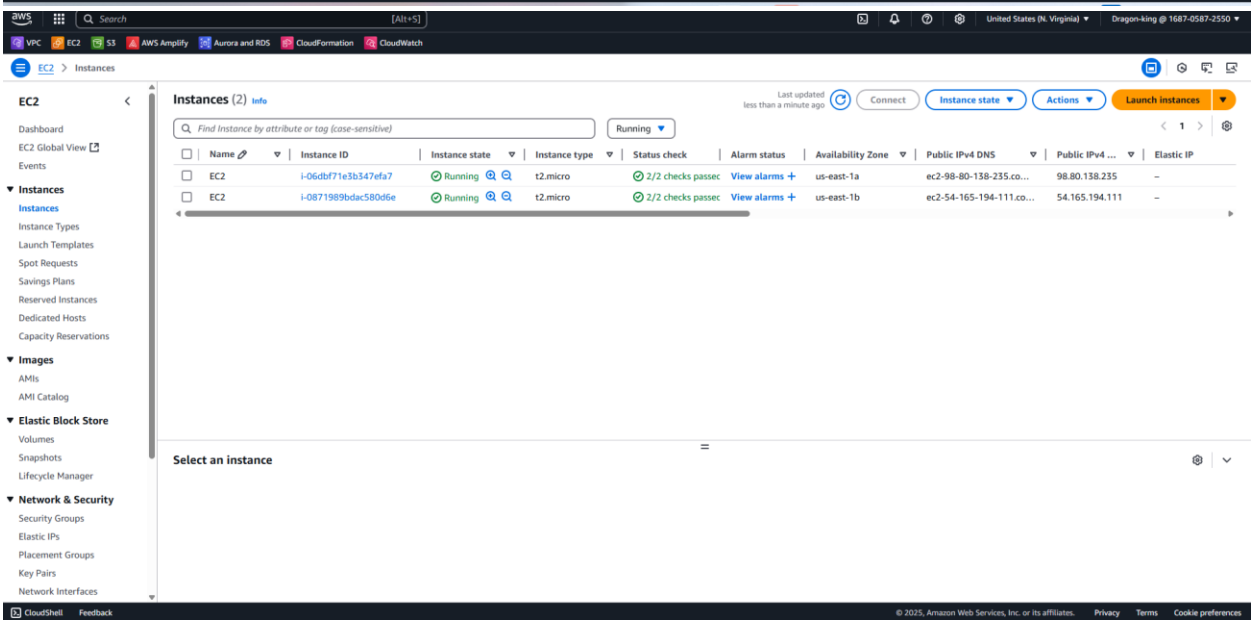
- **Screenshot:** CPU-STRESSED-TO-90%.png, ASG-CPU-STREESED-72%.png
- **Description:** Load generated to test auto-scaling response.



✅ Instances Scaled

- **Screenshot:** ASG-SPANNED-NEW-INSTANCE-TO COPE-WITH-THE-STRESS.png, 3RD-INSTANCE-SPANNED-TO-ACCOMODATE-THE-LOAD.png
- **Description:** New instances launched in response to high CPU usage.





Database

Prerequisites

- Your **EC2** instance and **RDS** are in the **same VPC**
- Your EC2 has **network access** (via route tables and security groups) to the RDS
- You have the **RDS endpoint** and port
- Your EC2 instance has the necessary **DB client tools** installed (e.g., mysql, psql)

🔍 Step 1: Verify RDS Details

Make sure you have:

- ✅ RDS **endpoint**: ✅ (database-1.cu5emkwymw55.us-east-1.rds.amazonaws.com)
- ✅ **Port**: e.g., 3306 (MySQL) or 5432 (PostgreSQL)
- ✅ **DB username & password**
- ✅ **DB engine** (MySQL, PostgreSQL, etc.)
- ✅ **VPC** and subnet setup for RDS

🔒 Step 2: Security Group Settings

To allow EC2 to reach RDS:

Go to: VPC > Security Groups

- Find the **security group attached to the RDS**
- Add an **Inbound Rule**:
  - Type: MySQL/Aurora or PostgreSQL (depending on your DB engine)
  - Port: 3306 or 5432
  - Source: Select **Custom** and type the **EC2's security group ID**

🌐 Step 3: Network Routing

- ✅ Make sure **EC2** and **RDS** are in the **same VPC**
- ✅ Their **subnets can route** to each other
- ✅ DNS resolution is enabled for EC2:
  - Go to **VPC > Your VPC > Edit DNS resolution** → Ensure both **DNS hostnames** and **DNS resolution** are enabled

✅ This ensures only your EC2 instance can access the DB

SSH INTO EC2 USING ENDPOINTS

: Install MySQL Client on EC2

If you don't have internet access yet, set up a NAT Gateway or use a bastion EC2 in public subnet to proxy installs.

To install:

Copysudo yum install mysql -y

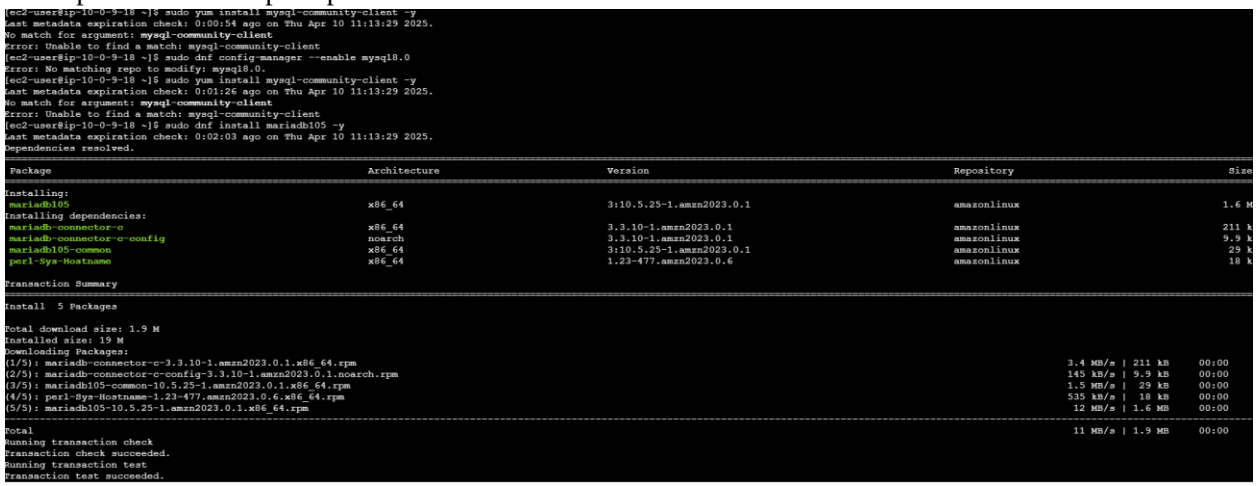
Connect to RDS from EC2

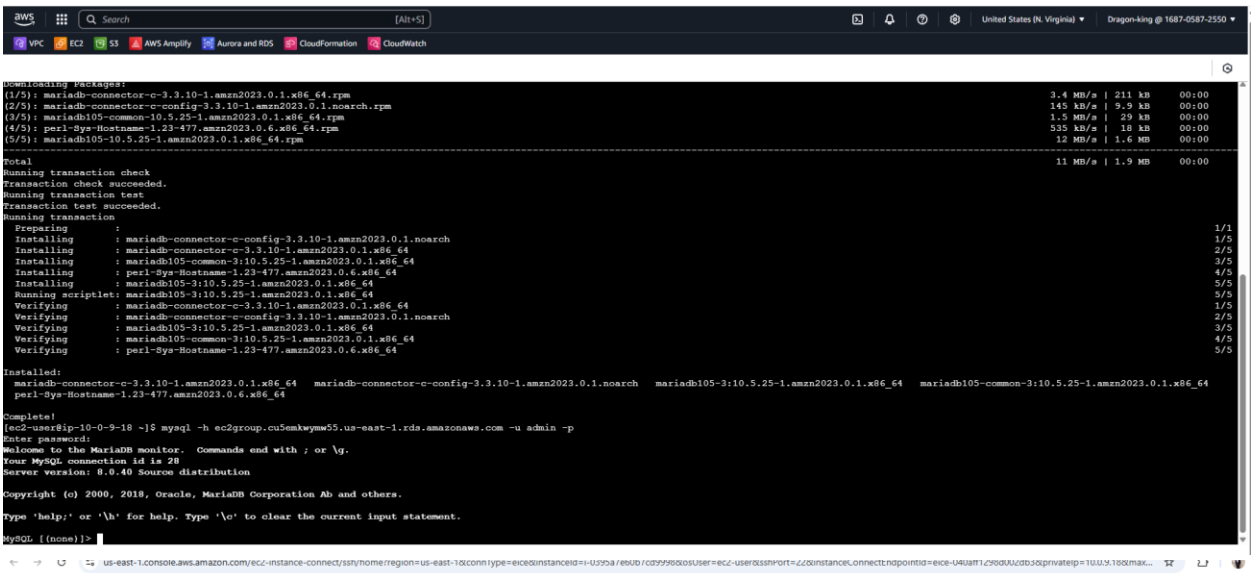
Once the security group rule is set and MySQL client is installed:

mysql -h <your-db-endpoint> -u <username> -p

mysql -h database-1.cu5emkwymw55.us-east-1.rds.amazonaws.com -u admin -p

Enter password when prompted.





```
Downloading Packages:
(1/5): mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64.rpm                                3.4 MB/s | 211 kB     00:00
(2/5): mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch.rpm                         145 kB/s | 9.9 kB     00:00
(3/5): mariadb105-common-10.5.25-1.amzn2023.0.1.x86_64.rpm                                 1.5 MB/s |  29 kB     00:00
(4/5): perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64.rpm                                  535 kB/s |  18 kB     00:00
(5/5): mariadb105-10.5.25-1.amzn2023.0.1.x86_64.rpm                                         12 MB/s | 1.6 MB     00:00
--------------------------------------------------------------------------------------------------------------------
Total                                                                                       11 MB/s | 1.9 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                            1/1
  Installing       : mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch                                    1/5
  Installing       : mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64                                           2/5
  Installing       : mariadb105-common-3:10.5.25-1.amzn2023.0.1.x86_64                                          3/5
  Installing       : perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64                                             4/5
  Installing       : mariadb105-3:10.5.25-1.amzn2023.0.1.x86_64                                                 5/5
  Running scriptlet: mariadb105-3:10.5.25-1.amzn2023.0.1.x86_64                                                 5/5
  Verifying        : mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64                                           1/5
  Verifying        : mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch                                    2/5
  Verifying        : mariadb105-3:10.5.25-1.amzn2023.0.1.x86_64                                                 3/5
  Verifying        : mariadb105-common-3:10.5.25-1.amzn2023.0.1.x86_64                                          4/5
  Verifying        : perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64                                             5/5

Installed:
  mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64   mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch   mariadb105-3:10.5.25-1.amzn2023.0.1.x86_64   mariadb105-common-3:10.5.25-1.amzn2023.0.1.x86_64
  perl-Sys-Hostname-1.23-477.amzn2023.0.6.x86_64

Complete!
[ec2-user@ip-10-0-9-18 ~]$ mysql -h ec2group.cu5emkwymw55.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: 8.0.40 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)]>
```


```
    ->   name VARCHAR(100),
    ->   email VARCHAR(100),
    ->   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.046 sec)

MySQL [sampledb]> CREATE TABLE users (
    ->   id INT AUTO_INCREMENT PRIMARY KEY,
    ->   name VARCHAR(100),
    ->   email VARCHAR(100),
    ->   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
ERROR 1050 (42S01): Table 'users' already exists
MySQL [sampledb]> SELECT * FROM users;
Empty set (0.000 sec)

MySQL [sampledb]> INSERT INTO users (name, email) VALUES ('Emmanuel', 'emmanuel@example.com');
Query OK, 1 row affected (0.004 sec)

MySQL [sampledb]> SELECT * FROM users;
+----+----------+---------------------+---------------------+
| id | name     | email               | created_at          |
+----+----------+---------------------+---------------------+
|  1 | Emmanuel | emmanuel@example.com | 2025-04-10 11:25:46 |
+----+----------+---------------------+---------------------+
1 row in set (0.001 sec)

MySQL [sampledb]> INSERT INTO users (name, email) VALUES
    -> ('Mary Mensah', 'maray@example.com'),
    -> ('Jeffery', 'jeffery@example.com'),
    -> ('Priscilla', 'priscilla@example.com'),
    -> ('Richard', 'richard@example.com');
Query OK, 4 rows affected (0.005 sec)
Records: 4  Duplicates: 0  Warnings: 0

MySQL [sampledb]> SELECT * FROM users;
+----+-------------+----------------------+---------------------+
| id | name        | email                | created_at          |
+----+-------------+----------------------+---------------------+
|  1 | Emmanuel    | emmanuel@example.com | 2025-04-10 11:25:46 |
|  2 | Mary Mensah | maray@example.com    | 2025-04-10 11:28:58 |
|  3 | Jeffery     | jeffery@example.com  | 2025-04-10 11:28:58 |
```


```
MySQL [sampledb]> CREATE TABLE users (
    ->   id INT AUTO_INCREMENT PRIMARY KEY,
    ->   name VARCHAR(100),
    ->   email VARCHAR(100),
    ->   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
ERROR 1050 (42S01): Table 'users' already exists
MySQL [sampledb]> SELECT * FROM users;
Empty set (0.000 sec)

MySQL [sampledb]> INSERT INTO users (name, email) VALUES ('Emmanuel', 'emmanuel@example.com');
Query OK, 1 row affected (0.004 sec)

MySQL [sampledb]> SELECT * FROM users;
+----+----------+---------------------+---------------------+
| id | name     | email               | created_at          |
+----+----------+---------------------+---------------------+
|  1 | Emmanuel | emmanuel@example.com | 2025-04-10 11:25:46 |
+----+----------+---------------------+---------------------+
1 row in set (0.001 sec)

MySQL [sampledb]> INSERT INTO users (name, email) VALUES
    -> ('Mary Mensah', 'maray@example.com'),
    -> ('Jeffery', 'jeffery@example.com'),
    -> ('Priscilla', 'priscilla@example.com'),
    -> ('Richard', 'richard@example.com');
Query OK, 4 rows affected (0.005 sec)
Records: 4  Duplicates: 0  Warnings: 0

MySQL [sampledb]> SELECT * FROM users;
+----+-------------+----------------------+---------------------+
| id | name        | email                | created_at          |
+----+-------------+----------------------+---------------------+
|  1 | Emmanuel    | emmanuel@example.com | 2025-04-10 11:25:46 |
|  2 | Mary Mensah | maray@example.com    | 2025-04-10 11:28:58 |
|  3 | Jeffery     | jeffery@example.com  | 2025-04-10 11:28:58 |
|  4 | Priscilla   | priscilla@example.com | 2025-04-10 11:28:58 |
|  5 | Richard     | richard@example.com  | 2025-04-10 11:28:58 |
+----+-------------+----------------------+---------------------+
5 rows in set (0.001 sec)

MySQL [sampledb]>
```

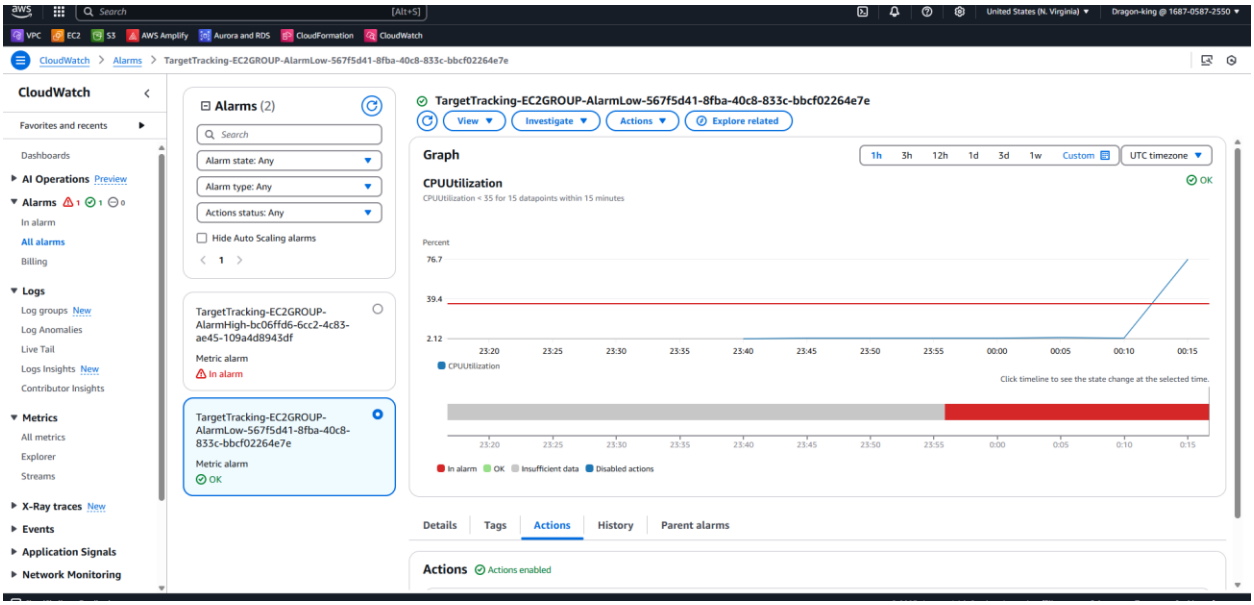
- **Monitoring & Logging**

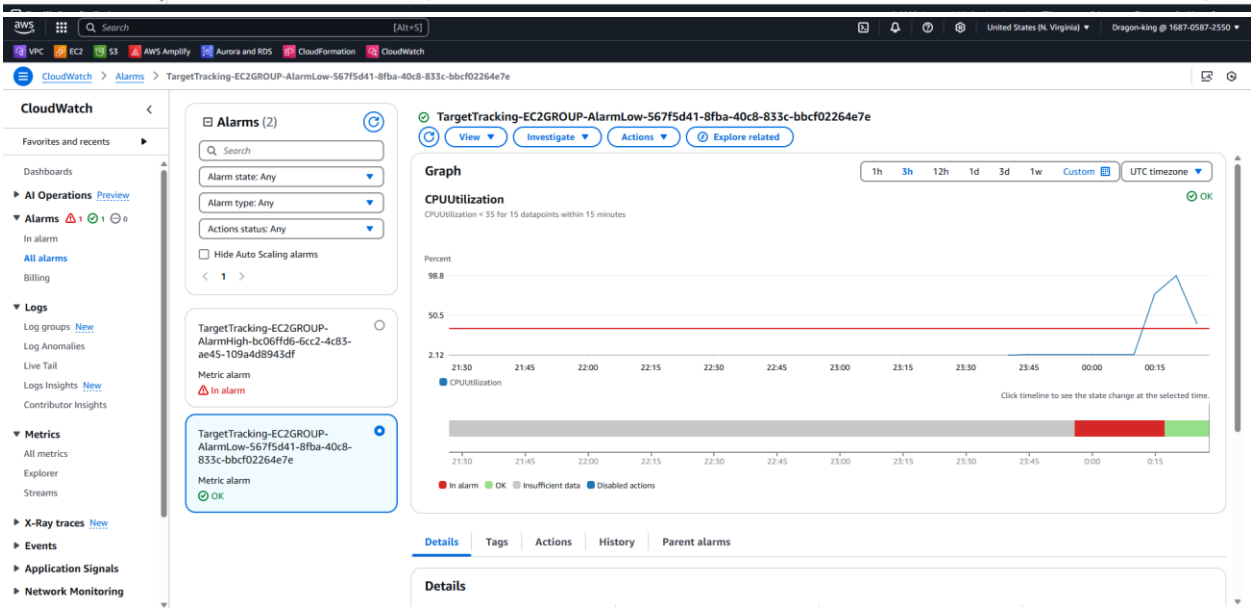
6.1 Enable CloudWatch Monitoring

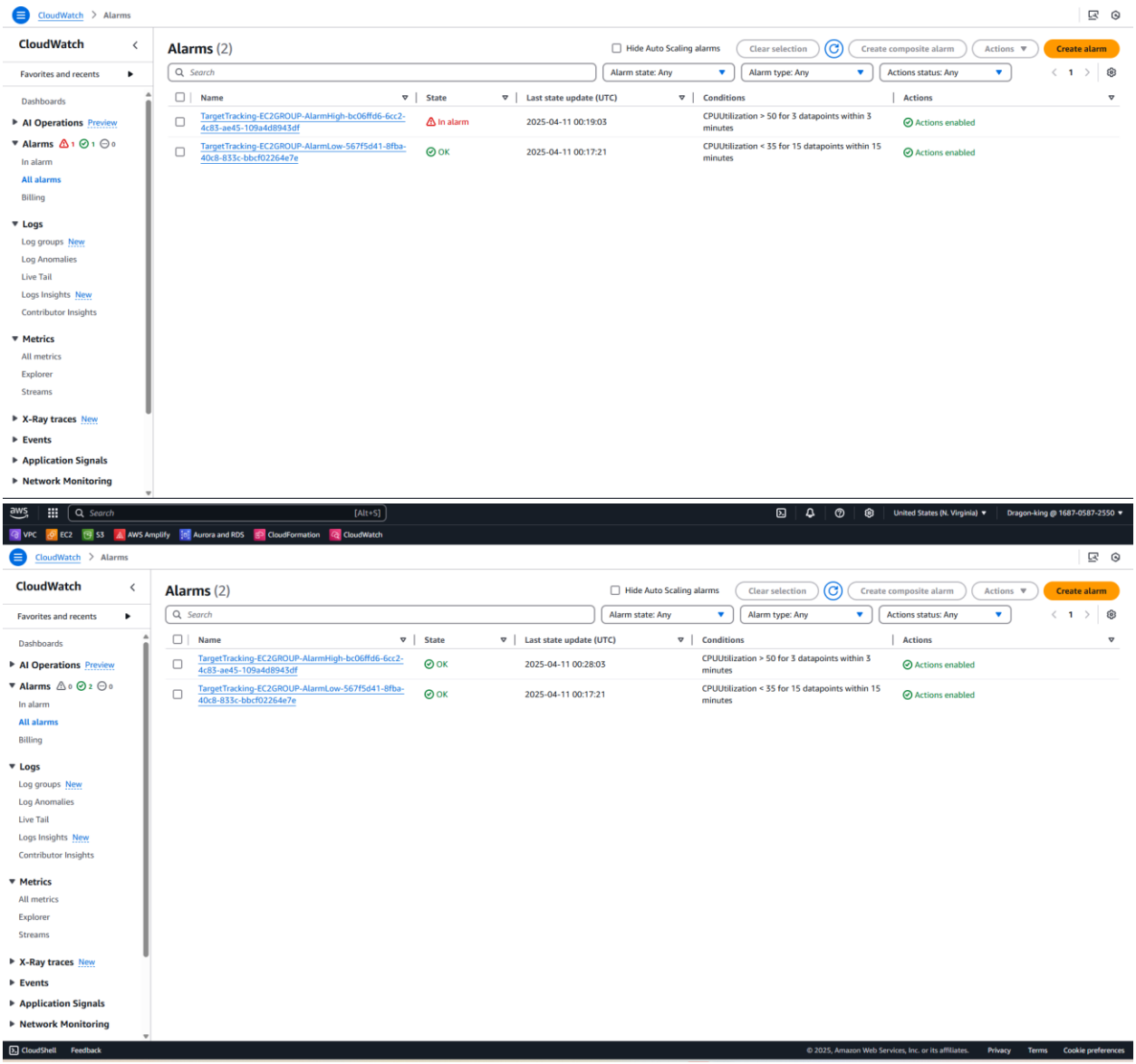
1. Go to **CloudWatch > Logs** > **Create Log Group**.
2. Name it: EC2group-Logs.
3. Go to **EC2 > CloudWatch Metrics** → Enable detailed monitoring.
4. Create **Alarms** for:
    a. High CPU Usage (> 80%)
    b. Low Available Memory

6.2 Enable VPC Flow Logs

1. Go to **VPC > Flow Logs** > **Create Flow Log**.
2. Choose:
    a. **VPC**: EC2
    b. **Destination**: CloudWatch Logs
    c. **Filter**: Accept-only traffic
3. Click **Create**.


CloudWatch > Alarms > TargetTracking-EC2GROUP-AlarmLow-567f5d41-8fba-40c8-833c-bbcf02264e7e
CloudWatch
Alarms (2)
Alarm state: Any
Alarm type: Any
Actions status: Any
Hide Auto Scaling alarms
TargetTracking-EC2GROUP-AlarmHigh-bc06ffd6-6cc2-4c83-ae45-109a4d8943df
Metric alarm
In alarm
TargetTracking-EC2GROUP-AlarmLow-567f5d41-8fba-40c8-833c-bbcf02264e7e
Metric alarm
OK
TargetTracking-EC2GROUP-AlarmLow-567f5d41-8fba-40c8-833c-bbcf02264e7e
View
Investigate
Actions
Explore related
Graph
1h 3h 12h 1d 3d 1w Custom
UTC timezone
CPUUtilization
CPUUtilization < 35 for 15 datapoints within 15 minutes
OK
Percent
76.7
39.4
2.12
CPUUtilization
Click timeline to see the state change at the selected time.
In alarm
OK
Insufficient data
Disabled actions
Details
Tags
Actions
History
Parent alarms
Actions
Actions enabled


CloudWatch > Alarms > TargetTracking-EC2GROUP-AlarmLow-567f5d41-8fba-40c8-833c-bbcf02264e7e
CloudWatch
Alarms (2)
Alarm state: Any
Alarm type: Any
Actions status: Any
Hide Auto Scaling alarms
TargetTracking-EC2GROUP-AlarmHigh-bc06ffd6-6cc2-4c83-ae45-109a4d8943df
Metric alarm
In alarm
TargetTracking-EC2GROUP-AlarmLow-567f5d41-8fba-40c8-833c-bbcf02264e7e
Metric alarm
OK
TargetTracking-EC2GROUP-AlarmLow-567f5d41-8fba-40c8-833c-bbcf02264e7e
View
Investigate
Actions
Explore related
Graph
1h 3h 12h 1d 3d 1w Custom
UTC timezone
CPUUtilization
CPUUtilization < 35 for 15 datapoints within 15 minutes
OK
Percent
98.8
50.5
2.12
CPUUtilization
Click timeline to see the state change at the selected time.
In alarm
OK
Insufficient data
Disabled actions
Details
Tags
Actions
History
Parent alarms
Details



❌ Issue Identified

Problem

Instances launched by the Auto Scaling Group **did not run** the web application.

Root Cause

The launch template did **not include** application deployment instructions.

✅ Solutions

Option 1: Create a Custom AMI

1. Launch & configure EC2.
2. Install the application.
3. Go to EC2 > Actions > Image > Create Image.
4. Use the AMI in your launch template.

Option 2: Use User Data Script

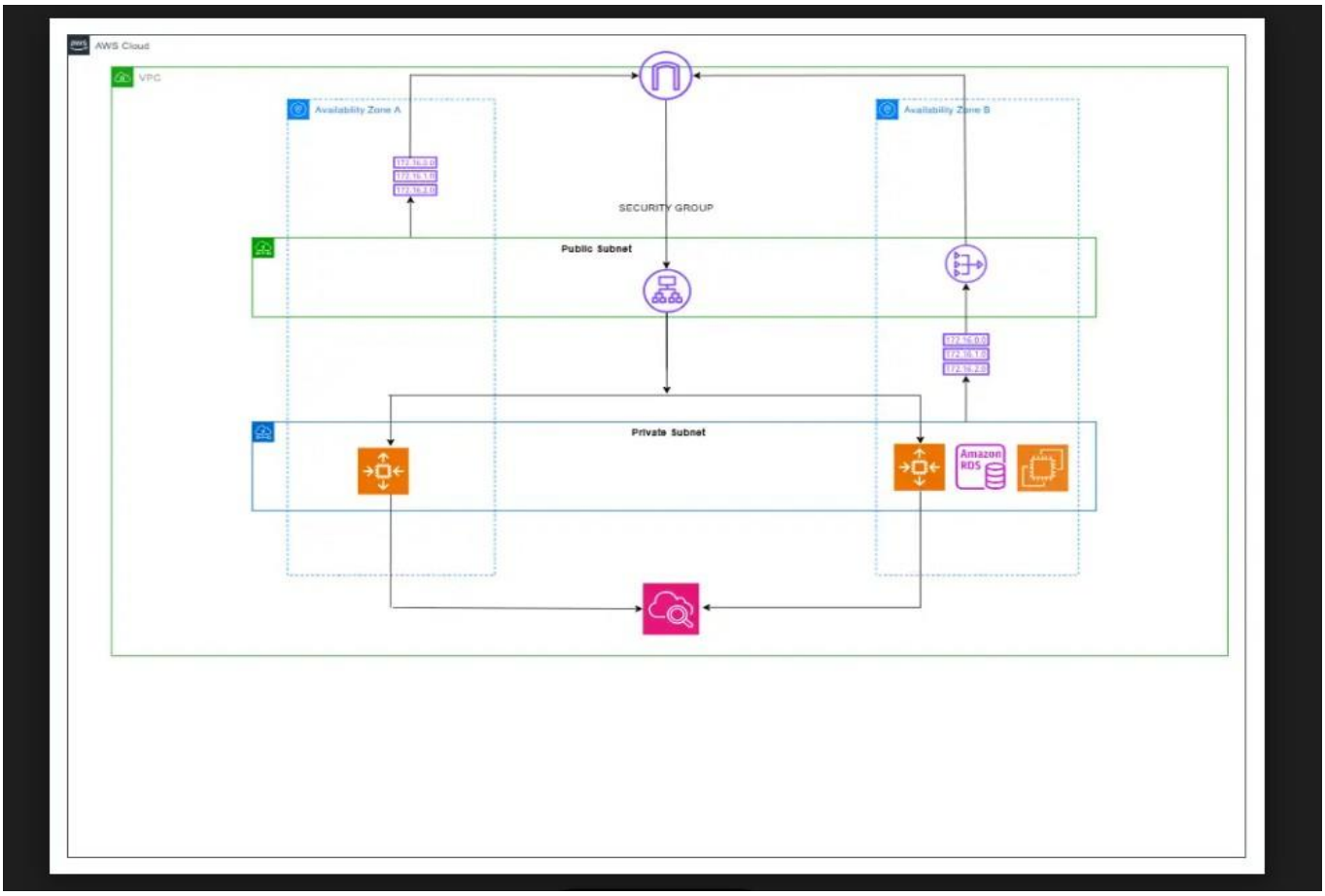
Add the following script in the launch template:

```
#!/bin/bash
yum update -y
yum install -y nginx unzip
cd /usr/share/nginx/html
aws s3 cp s3://ec2/livestock-management-1.zip .
unzip livestock-management-1.zip
systemctl restart nginx

npm build

npm run
```

🔁 Architecture Overview

🧠 Flowchart

- **Screenshot:** flowchat.png
- **Description:** High-level visual representation of the AWS architecture.



Summary

To ensure all ASG instances are provisioned with your app:

- Use a pre-baked AMI, or
- Use a proper user data script