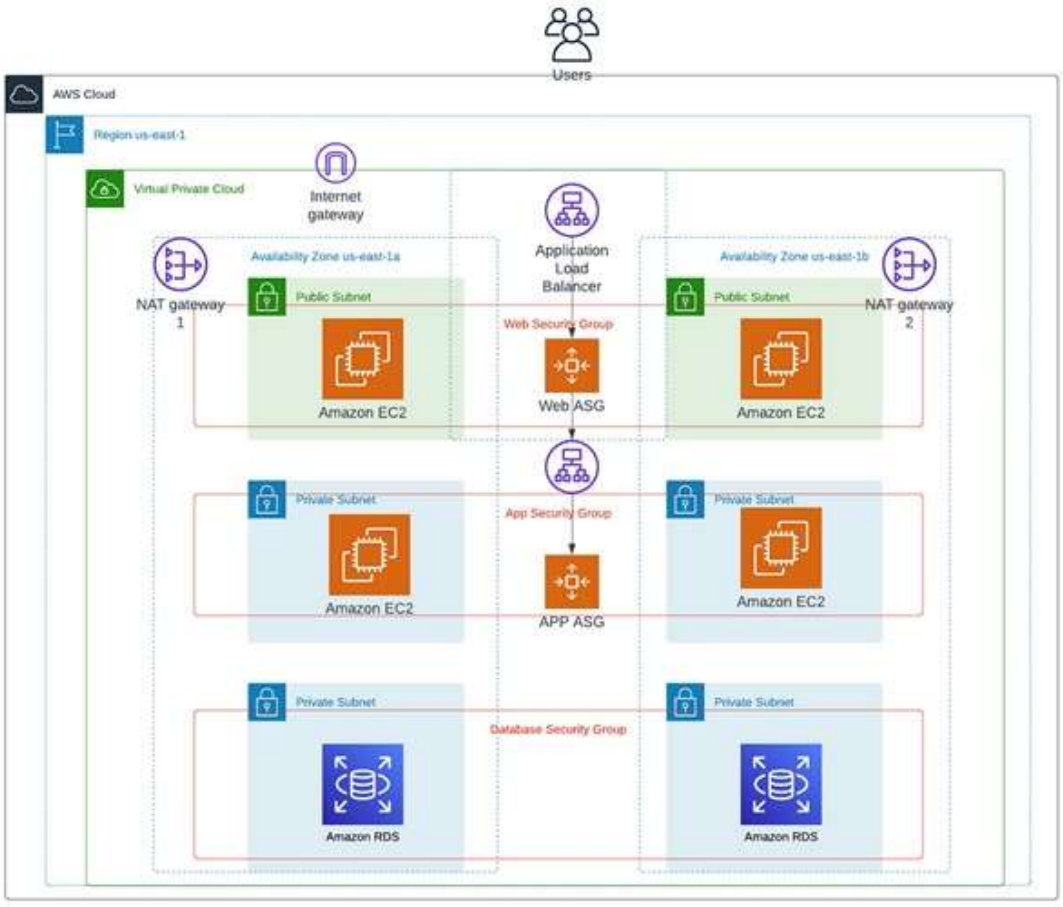
AWS DEVSECOPS-139

PROJECT- 01
3-TIER ARCHITECTURE

P. CHARAN KUMAR REDDY
Mail Id: charankumarreddyperam@gmail.com

3-TIER ARCHITECHTURE



What Are the 3 Tiers?

1) Web Tier

- Amazon EC2 in Public Subnet: Handles incoming user traffic.
- Connected to an Application Load Balancer.
- Protected by Web Security Group.

2) App Tier

- Amazon EC2 in Private Subnet: Processes the logic (e.g., Java, Spring Boot, Node.js).
- Not directly accessible from the internet.
- Protected by App Security Group.

3) Database Tier

- Amazon RDS in Private Subnet: Stores app data.
- Protected by Database Security Group.
- Only accessible from App tier.

Process:

Step 1: Setup VPC and Networking.

Creating a Virtual Private Cloud (VPC) in AWS involves several steps to establish a logically isolated network environment for your resources.

1. Create the VPC:

- Navigate to the Amazon VPC console.
- Choose "Create VPC."
- Select "VPC and more" for a comprehensive setup.
- Provide a name tag for auto-generation, for example, Project-1.
- Specify an IPv4 CIDR block (e.g., 10.0.0.0/16). Optionally, add an IPv6 CIDR block.

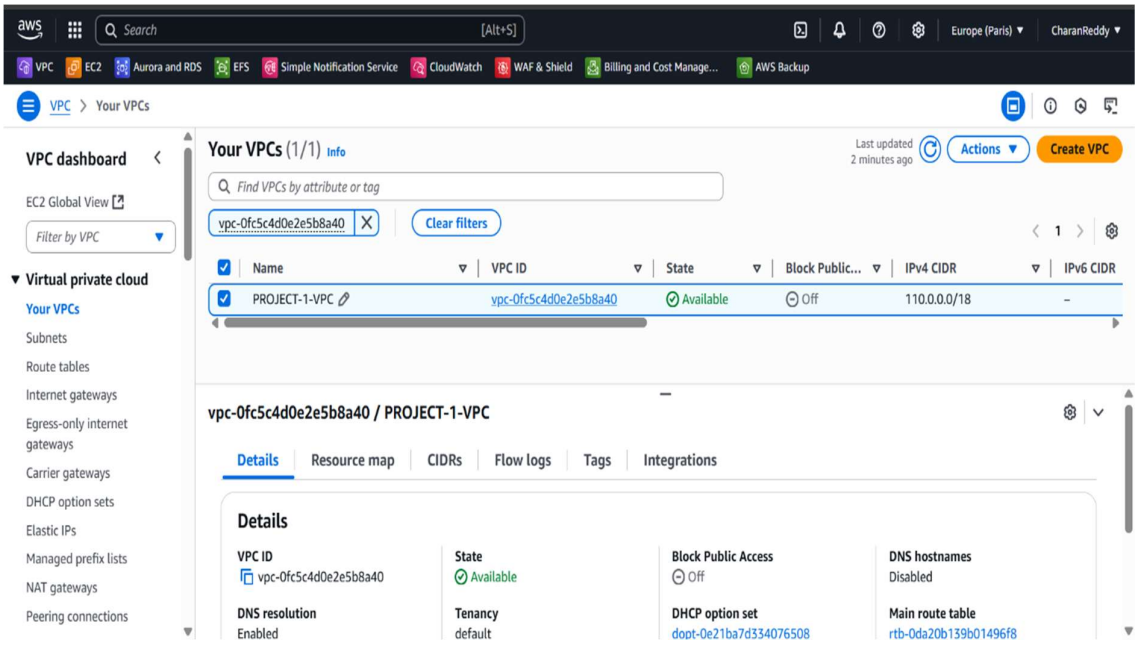


Fig 1.1: VPC

2. Configure Subnets:

- Determine the number of Availability Zones to use (e.g., 1 or more).
- Specify the number of public subnets(2) and private subnets(4) needed.

- Review and customize the CIDR blocks for each subnet if required.

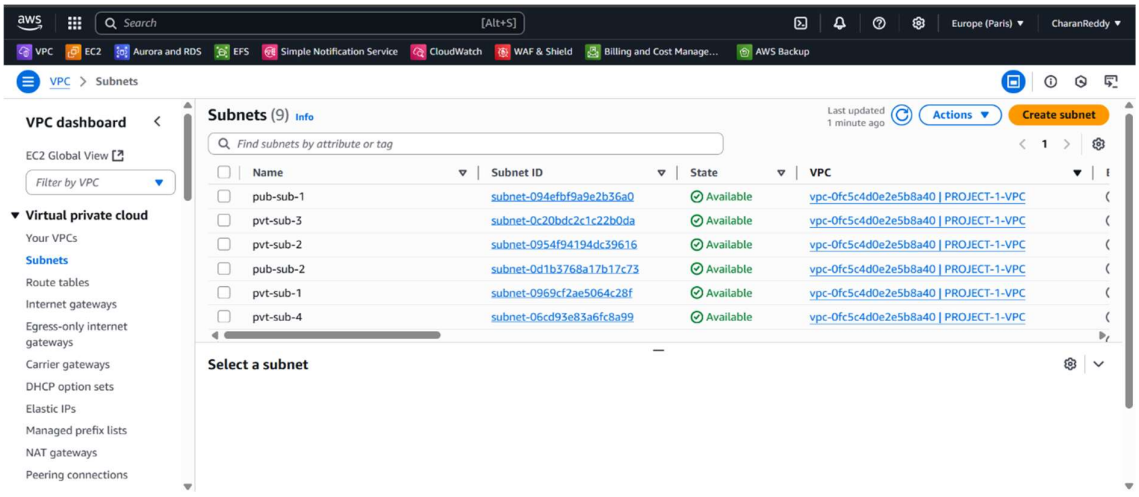


Fig 2.1: Subnets

3. Configure Internet Connectivity:

- For public subnets to access the internet, an Internet Gateway is required.

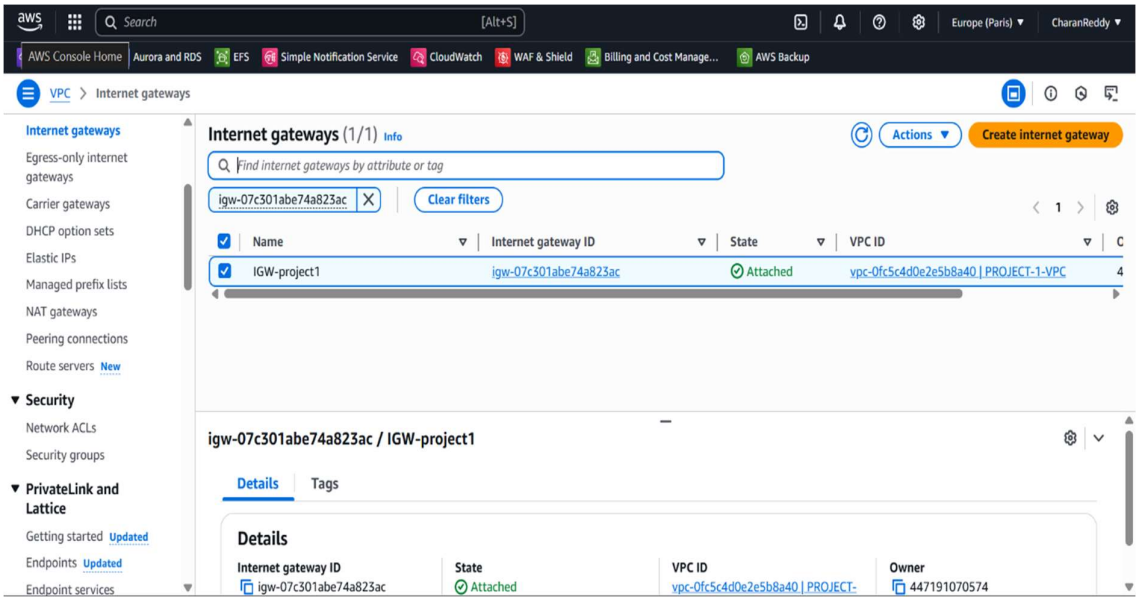


Fig 3.1: Internet Gateway

- For private subnets need outbound internet access, a NAT Gateway (for IPv6) is necessary.

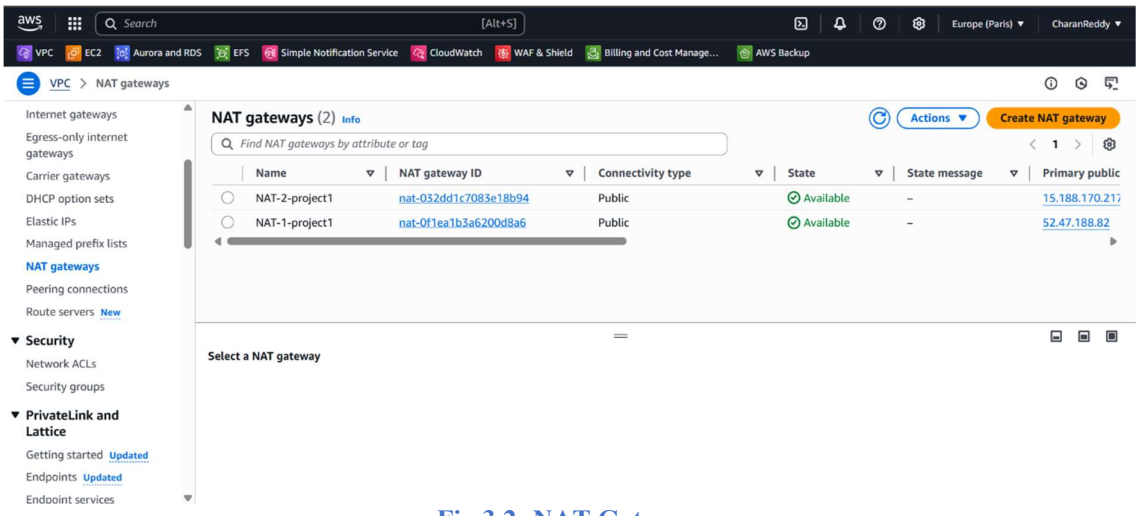


Fig 3.2: NAT Gateway

4. Configure Routing:

- AWS automatically creates route tables for your subnets.
- For public subnets, ensure a route to the Internet Gateway (0.0.0.0/0 pointing to the IGW) is present.
- For private subnets, using a NAT Gateway, ensure a route to the NAT Gateway (IPV4 CIDR of private subnets pointing to the NAT Gateway) is present.

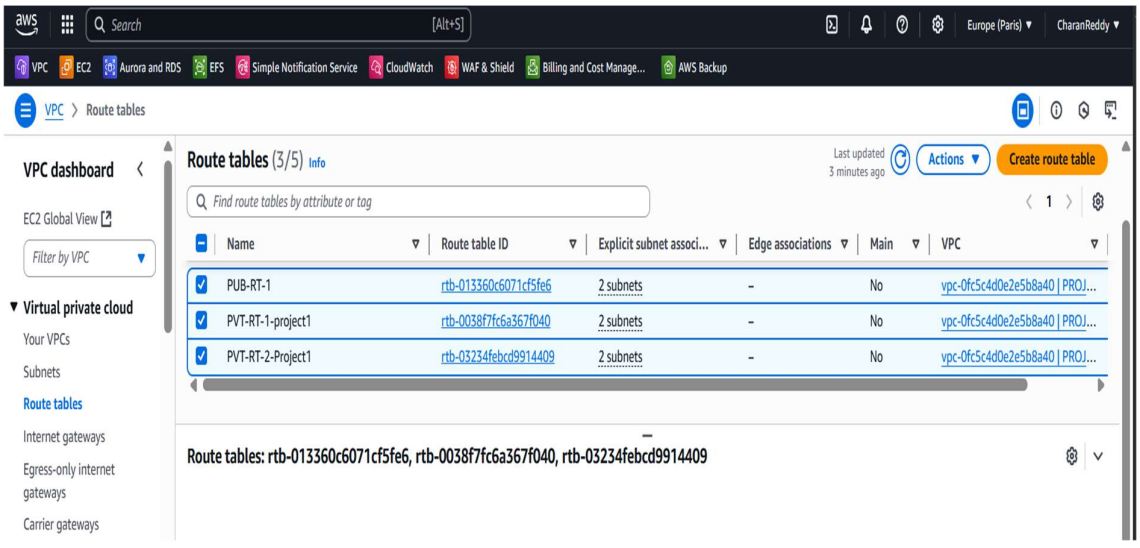


Fig 4.1: Route Tables

5. Configure Security:

- Create Security Groups to control inbound and outbound traffic for instances within your subnets **(SG-1-Project1).**
- Create Security Groups to allow traffic for MySQL server **(SG-2-Project-1)**.

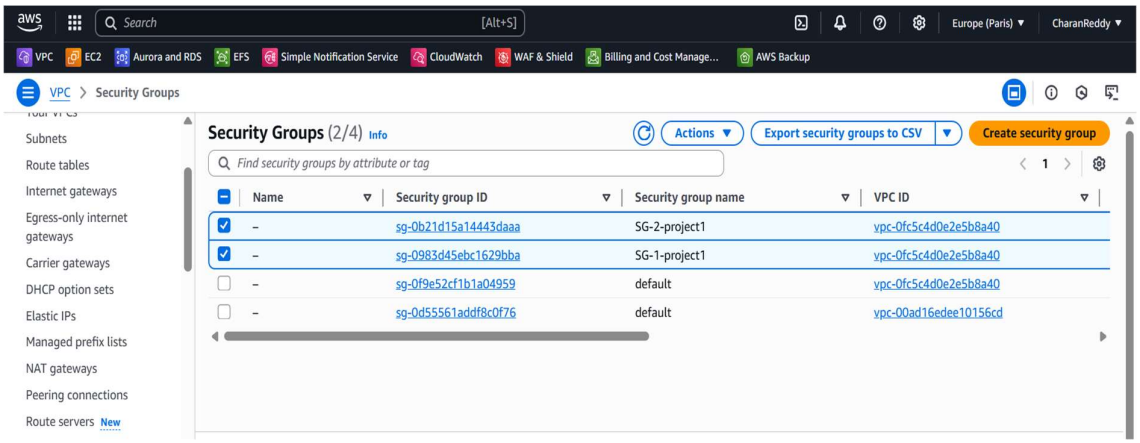


Fig 5.1: Security Groups

6. Enable DNS Hostnames:

- To enable automatic assignment of DNS hostnames to EC2 instances in your VPC, navigate to "Your VPCs," select your VPC, and edit the VPC settings to enable DNS hostnames.

Step 2:

1. Launch EC2 Instances

- Launch instances by using public and private subnets to verify network connectivity and internet access as intended.
- Here we launch 2 public EC2 instances for web tier & 2 private instances app tier.
- We use only SG-1(security group) for all the instances.

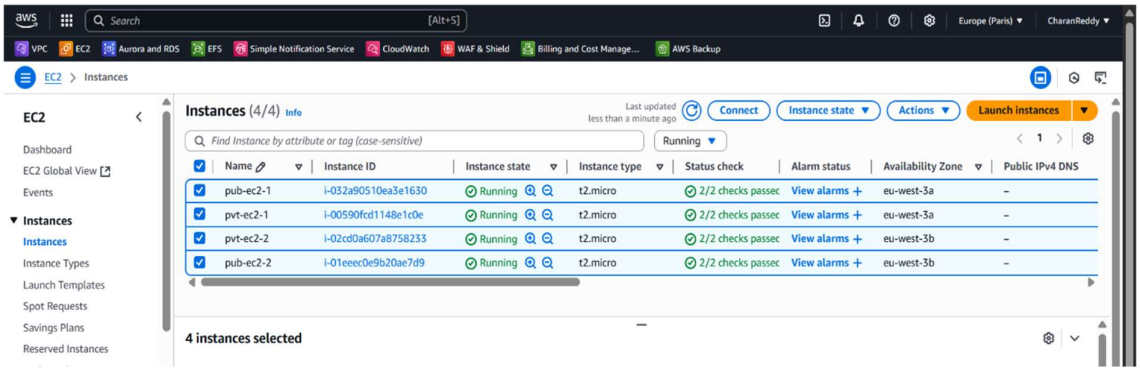


Fig 2.1.1: EC2 Instances

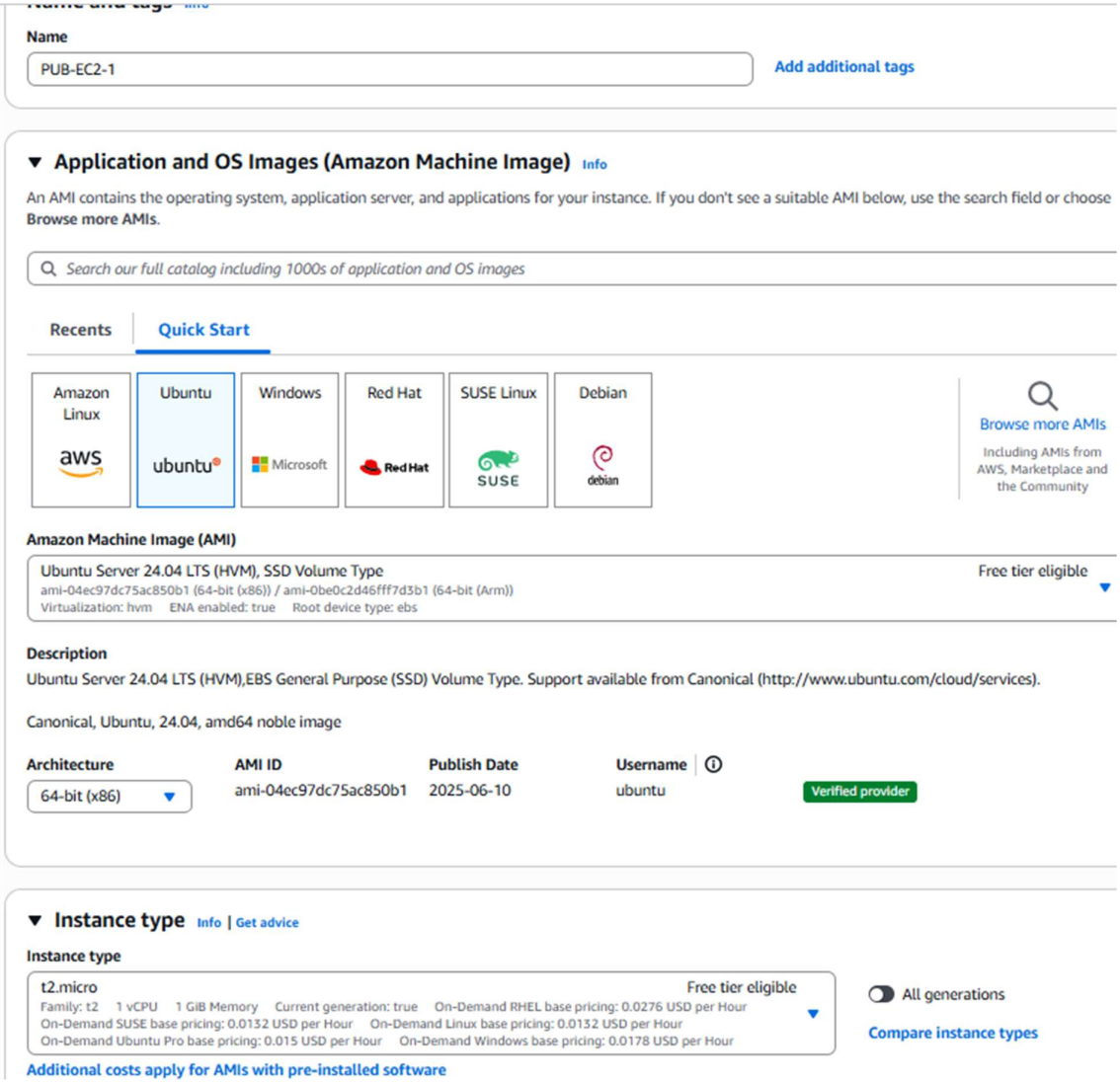


Fig 2.1.2: EC2

- for instances I use ubuntu OS, t2, micro instance type and key pair.

2. To create an image (Amazon Machine Image - AMI) of an EC2 instance in AWS, follow these steps

- Access the EC2 Dashboard.
- Select the Instance: In the navigation pane, choose "Instances" and then select the specific EC2 instance from which you want to create an AMI.
- Initiate Image Creation.
- Configure AMI Details.

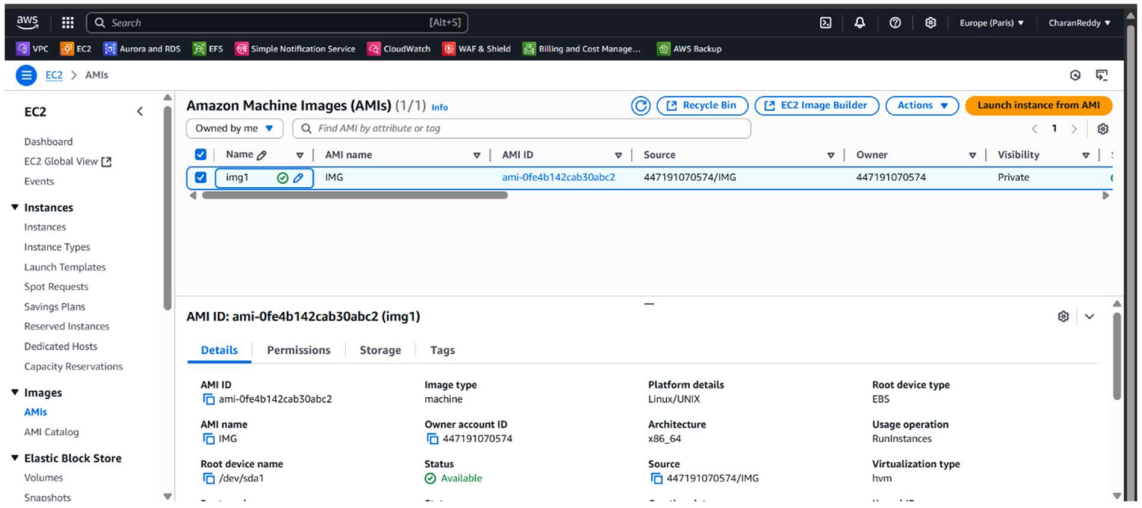


Fig 2.2.1: AMI (Image)

3. Launch Template

- Navigate to the EC2 Dashboard.
- Select Launch Templates.
- Choose Launch Template.
- Create Template **(TMP)**.

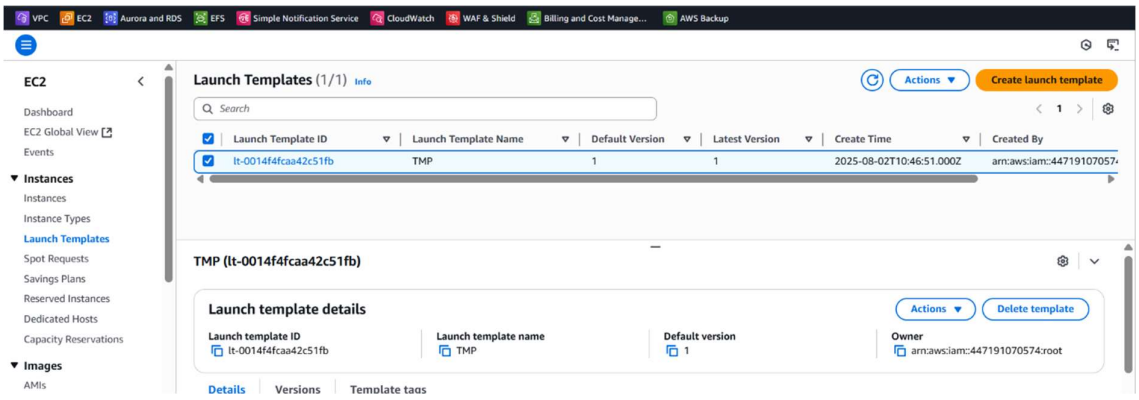


Fig 2.3.1: Template

4. **Configure a Target Group:** I created two target groups

   - PUB-TG-1(for public load balancer)
   - PVT-TG-2(for private load balancer)

- **Create Target Group:** In the navigation pane, under "Load Balancing," choose "Target Groups," then "Create target group."
- **Basic Configuration:**
  - **Choose a target type:** Select "Instances" to register EC2 instances or "IP addresses" to register targets by their IP.
  - **Protocol and Port:** Define the protocol (e.g., HTTP, HTTPS) and port on which the targets will receive traffic.
  - **Health checks:** Configure health check settings to ensure the ALB only routes traffic to healthy targets.
- **Register targets:** After creating the target group, navigate to its details and register the EC2 instances or IP addresses that will serve as targets for the ALB. Ensure these targets are in the same VPC as your planned ALB.

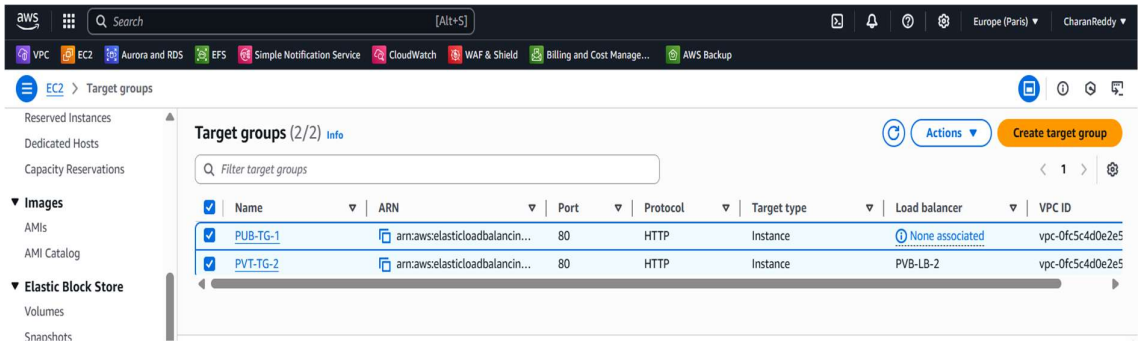


Fig 2.4.1: Target Groups

5. **Configure the Application Load Balancer and a Listener:**

- **Create Load Balancer:** In the EC2 console, under "Load Balancing," choose "Load Balancers," then "Create load balancer." Select "Application Load Balancer" and choose "Create."
- Create 2 load balancers one for web tier **(PVB-LB-1)**, another for app tier **(PVT-LB-2)**.

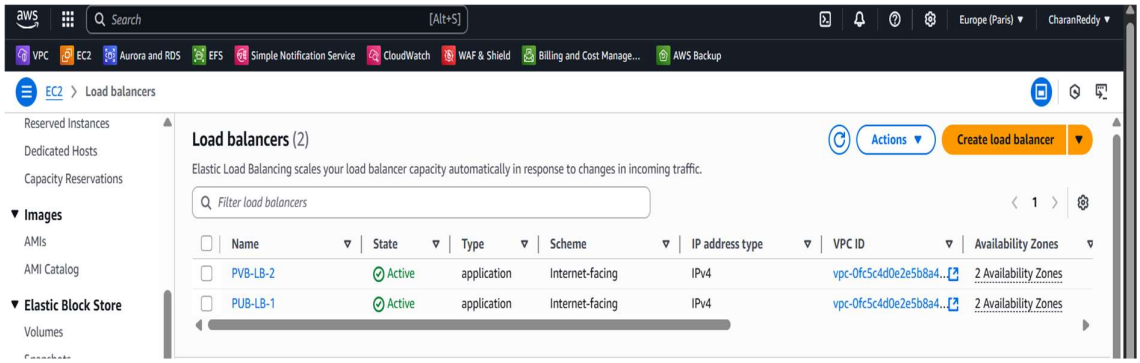


Fig 2.5.1: Load Balancers

6. Create an AWS Auto Scaling:

- Launch Template.
- Auto Scaling Group.
- Scaling Policies.
- Desired Capacity.
- Minimum/Maximum Size.
- Attach Load Balancers.

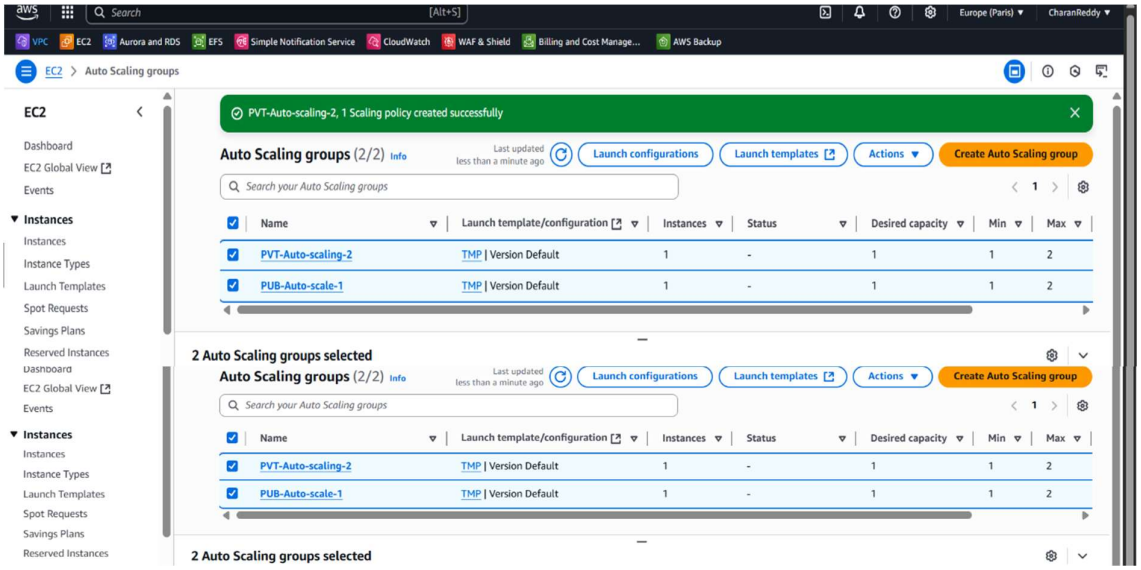


Fig 2.6.1: Auto Scaling

- After creating auto scaling we get extra ec2 instances which we gave the desired capacity of auto scaling.
- Auto scale manages the servers for increasing and decreasing the server and also performance by adding extra CPU's when needed **(AS-1, AS-2)**.

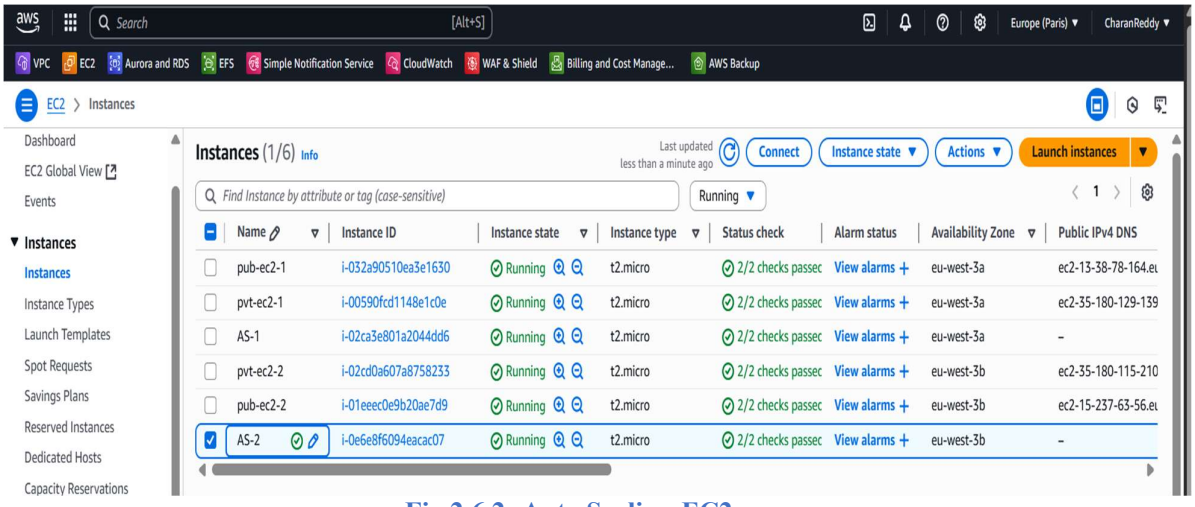


Fig 2.6.2: Auto Scaling-EC2

Step 3:

1. Subnet Groups:

We need to create one subnet group for Database tier to create RDS database within the subnet group.

- Sub-grp-1

Steps to create an RDS DB subnet group:

- Open the Amazon RDS console
- Navigate to Subnet groups.
- Initiate creation.
- Provide details.
- Select subnets.
- Create the subnet group.

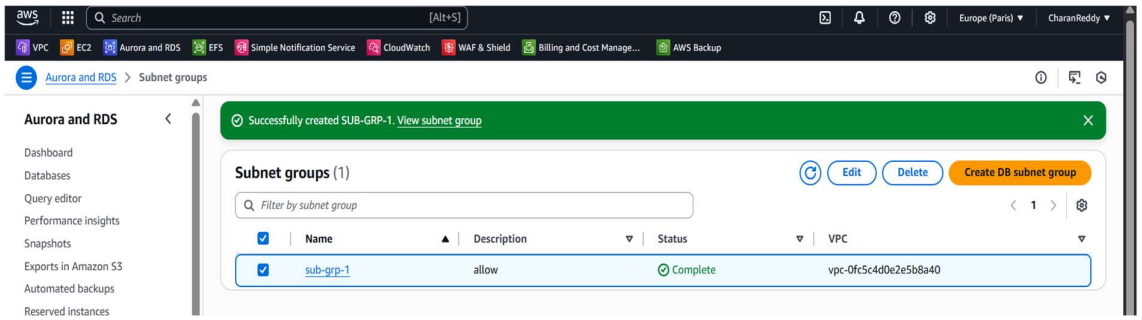


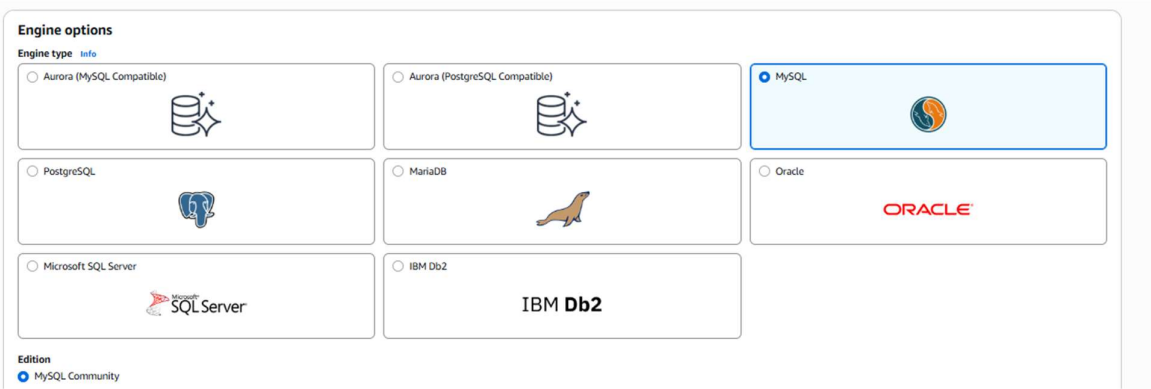
Fig 3.1.1: Subnets Group

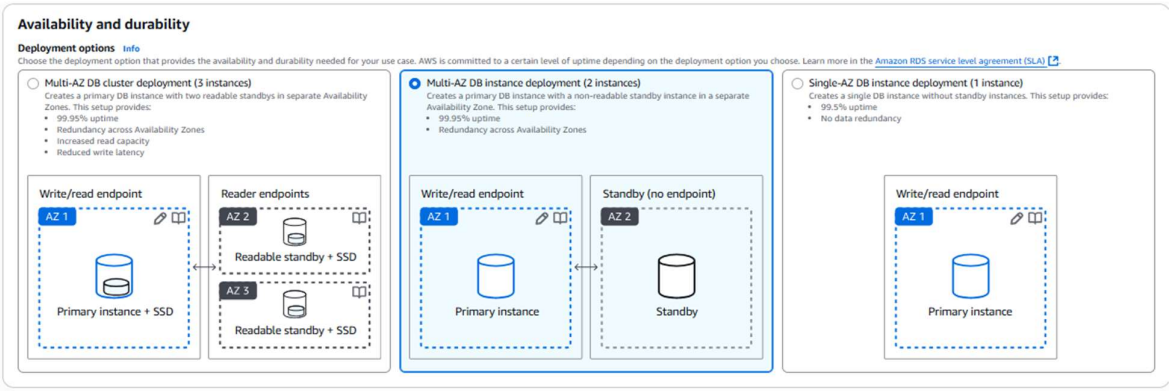
2. Create RDS (database-1):

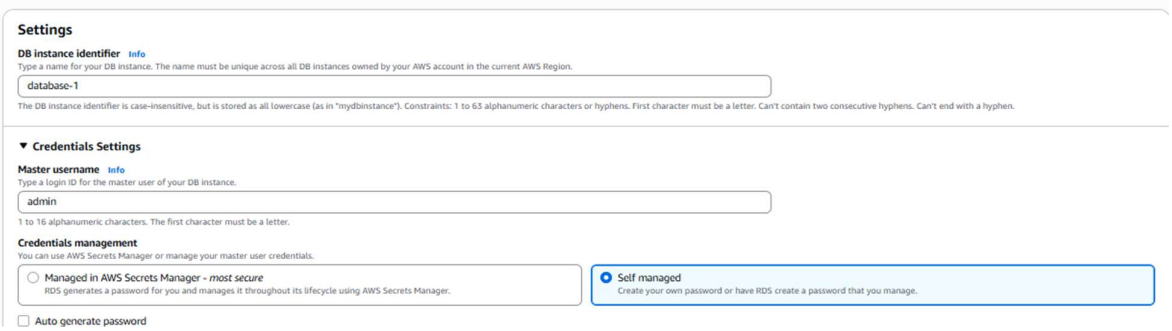
Creating an Amazon Relational Database Service (RDS) database involves several steps within the AWS Management Console.

- Access the RDS Console.
- Create Database.
- Choose Creation Method and Engine (MySQL).
- Configure Database Settings (Multi-AZ (2instances)).
- Configure Connectivity (sub-grp-1).
- Initial database name(database-1).
- Specify a name for the initial database within the instance.
- Backup retention, monitoring, and maintenance settings.

Review and Create: Review all your configurations and Click "Create database" to provision your RDS instance.







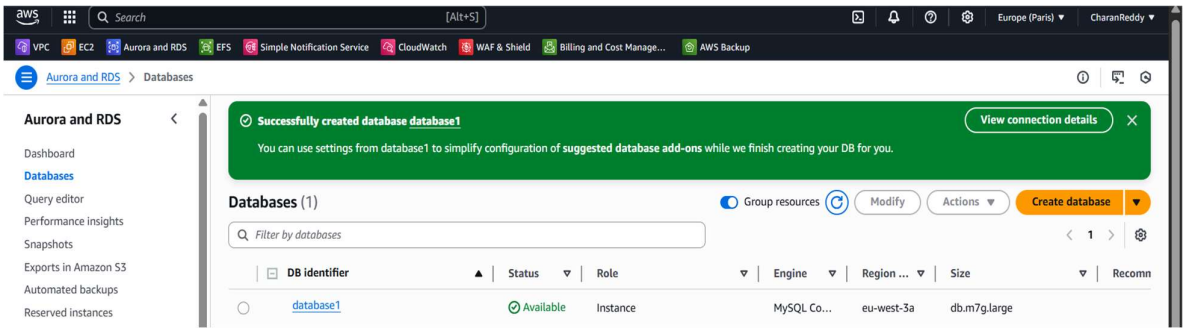


Fig 3.2.1: RDS Database

Commands List we used for this project :

- **sudo apt update –y**: Update the server.
- **yum install apache2**: Install the apache2 software in server.
- **cd /var/www/html**: Change directory and check the index file.
- **Ls Index.html:** List with Index file.
- **rm index.html:** Remove the existing index file.
- **vi index.html:** Then create index.html.
- **systemctl restart apache2**: Server Restart.
- **systemctl status apache2**: Server status check.
- **sudo apt install mysql-server**: Then install the mysql-server package.
- **sudo systemctl start mysql.server**: Ensure that the server is running.
- **MySQL -h** pj-rds.cjkyiegucdkd.eu-west-2.rds.amazonaws.com **-u admin -p**: used to connect to MySQL.

Connect and Manage:

- Once the database status changes to "Available," you can connect to it using a database client (e.g., MySQL Workbench) and the master user credentials.

Step 4:

1. Connect PUB-EC2-1 to PVT-EC2-1

- First connect web instance through Gitbash build a connection and make sure root user(sudo -i).
- Save the key pair inside the web instance with the same name used in the private instance after connecting through ssh.
- Change the permissions of user to read only (**chmod 400 Project-1.pem)**
- Copy the ssh client link of PVT-EC2-1 to replace the public IP with private IP(app instance) and then make connection successfully.

```
ubuntu@ip-110-0-23-71:~$ sudo -i
root@ip-110-0-23-71:~# chmod 400 project-1.pem
root@ip-110-0-23-71:~# ls -l
total 8
-r-------- 1 root root 1679 Aug  2 11:38 project-1.pem
drwx------ 3 root root 4096 Aug  2 10:39 snap
root@ip-110-0-23-71:~# ssh -i "project-1.pem" ubuntu@ec2-110-0-40-93.eu-west-3.compute.amazonaws.com
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Aug  2 11:48:21 UTC 2025

  System load:  0.0               Processes:             105
  Usage of /:   25.6% of 6.71GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for enX0: 110.0.40.93
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Sat Aug  2 11:46:21 2025 from 110.0.23.71
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-110-0-40-93:~$ |
```

- 5) For connecting to RDS we need to install MySQL-server in app instance. Follow the following commands to install MySQL and connect to database.

```
root@ip-110-0-23-71:~# sudo apt install mysql-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7t64 libfcgi-bin libfcgi-perl libfcgi0t64 libhtml-parser-perl
  libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libmecab2 libprotobuf-lite32t64 libtimedate-perl
  liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server-8.0 mysql-server-core-8.0
Suggested packages:
  libdata-dump-perl libipc-sharedcache-perl libio-compress-brotli-perl libbusiness-isbn-perl libregexp-ipv6-perl libwww-perl mailx tinyca
The following NEW packages will be installed:
  libcgi-fast-perl libcgi-pm-perl libclone-perl libencode-locale-perl libevent-pthreads-2.1-7t64 libfcgi-bin libfcgi-perl libfcgi0t64 libhtml-parser-perl
  libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libmecab2 libprotobuf-lite32t64 libtimedate-perl
  liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0 mysql-client-core-8.0 mysql-common mysql-server mysql-server-8.0 mysql-server-core-8.0
0 upgraded, 28 newly installed, 0 to remove and 98 not upgraded.
Need to get 29.6 MB of archives.
After this operation, 243 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble/main amd64 mysql-common all 5.8+1.1.0build1 [6746 B]
Get:2 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 mysql-client-core-8.0 amd64 8.0.42-0ubuntu0.24.04.2 [2728 kB]
Get:3 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 mysql-client-8.0 amd64 8.0.42-0ubuntu0.24.04.2 [22.4 kB]
Get:4 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libevent-pthreads-2.1-7t64 amd64 2.1.12-stable-9ubuntu2 [7982 B]
Get:5 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libmecab2 amd64 0.996-14ubuntu4 [201 kB]
Get:6 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libprotobuf-lite32t64 amd64 3.21.12-8.2ubuntu0.2 [238 kB]
Get:7 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 mysql-server-core-8.0 amd64 8.0.42-0ubuntu0.24.04.2 [17.5 MB]
Get:8 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 mysql-server-8.0 amd64 8.0.42-0ubuntu0.24.04.2 [1438 kB]
Get:9 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libhtml-tagset-perl all 3.20-6 [11.3 kB]
Get:10 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble/main amd64 liburi-perl all 5.27-1 [88.0 kB]
Get:11 http://eu-west-3.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libhtml-parser-perl amd64 3.81-1build3 [85.8 kB]
```

Fig 4.1.1: Installing MySQL

- After installing the MySQL server in APP tier we need to make connection to MySQL as shown in below

```
root@ip-110-0-23-71:~# mysql -h database1.cxkeaqaaq645.eu-west-3.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 50
Server version: 8.0.41 Source distribution

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql>
```

- Here we need to create one Database and connect it to create a table.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| charan             |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

```
mysql> use charan;
Database changed
mysql> CREATE TABLE Books (
    ->     BookID INT PRIMARY KEY,
    ->     Title VARCHAR(100),
    ->     Author VARCHAR(100),
    ->     PublishedYear INT,
    ->     Genre VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO Books (BookID, Title, Author, PublishedYear, Genre)
    -> VALUES
    -> (1, 'To Kill a Mockingbird', 'Harper Lee', 1960, 'Fiction'),
    -> (2, '1984', 'George Orwell', 1949, 'Dystopian'),
    -> (3, 'The Great Gatsby', 'F. Scott Fitzgerald', 1925, 'Classic'),
    -> (4, 'Sapiens', 'Yuval Noah Harari', 2011, 'Non-fiction');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> show tables;
+-----------------+
| Tables_in_charan |
+-----------------+
| Books           |
+-----------------+
1 row in set (0.00 sec)
```

- Table created (BOOKS)

```
mysql> select* from Books;
+--------+----------------------+--------------------+--------------+-------------+
| BookID | Title                | Author             | PublishedYear | Genre       |
+--------+----------------------+--------------------+--------------+-------------+
|      1 | To Kill a Mockingbird | Harper Lee         |         1960 | Fiction     |
|      2 | 1984                 | George Orwell      |         1949 | Dystopian   |
|      3 | The Great Gatsby     | F. Scott Fitzgerald |         1925 | Classic     |
|      4 | Sapiens              | Yuval Noah Harari  |         2011 | Non-fiction |
+--------+----------------------+--------------------+--------------+-------------+
4 rows in set (0.00 sec)

mysql>
```

- Connect another instance (app tier) with other availability zone and check the table we created is accessed from the other instance or not.

```
ubuntu@ip-110-0-23-71:~$ sudo -i
root@ip-110-0-23-71:~# chmod 400 project-1.pem
root@ip-110-0-23-71:~# ls -l
total 8
-r-------- 1 root root 1679 Aug  2 11:38 project-1.pem
drwx------ 3 root root 4096 Aug  2 10:39 snap
root@ip-110-0-23-71:~# ssh -i "project-1.pem" ubuntu@ec2-110-0-40-93.eu-west-3.compute.amazonaws.com
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat Aug  2 11:48:21 UTC 2025

  System load:  0.0               Processes:             105
  Usage of /:   25.6% of 6.71GB   Users logged in:       0
  Memory usage: 21%               IPv4 address for enX0: 110.0.40.93
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Sat Aug  2 11:46:21 2025 from 110.0.23.71
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-110-0-40-93:~$
```

- Install and restart MySQL server in instance PVT-EC2-2

```
root@ip-110-0-23-71:~# sudo systemctl start mysql.service
root@ip-110-0-23-71:~#
```

- Successfully we accessed the Books table from other instance

```
mysql> select* from Books;
+--------+---------------------+-------------------+-------------+------------+
| BookID | Title               | Author            | PublishedYear | Genre      |
+--------+---------------------+-------------------+-------------+------------+
|      1 | To Kill a Mockingbird | Harper Lee        |         1960 | Fiction    |
|      2 | 1984                | George Orwell     |         1949 | Dystopian  |
|      3 | The Great Gatsby    | F. Scott Fitzgerald |         1925 | Classic    |
|      4 | Sapiens             | Yuval Noah Harari |         2011 | Non-fiction |
+--------+---------------------+-------------------+-------------+------------+
4 rows in set (0.00 sec)

mysql>
```

Conclusion

The three-tier architecture in AWS provides a robust and well-established framework for deploying scalable, secure, and highly available applications. Its core strength lies in the logical separation of concerns into distinct web tier (presentation layer), app tier (application layer), and database tier (data layers), each optimized for its specific function. This modularity facilitates independent development, scaling, and maintenance, minimizing the impact of changes or failures in one tier on the others.

By leveraging AWS services like Amazon VPC for network isolation, Auto Scaling Groups and Elastic Load Balancers for high availability and scalability, and various database services for data persistence, the three-tier architecture offers a resilient and efficient solution for diverse application needs. This design promotes enhanced security through controlled access between tiers and allows for optimized resource utilization by enabling independent scaling of each layer based on demand. Ultimately, the three-tier architecture on AWS serves as a foundational pattern for building modern, cloud-native applications that can adapt to evolving business requirements and user traffic.