Final Project

Course: DevOps

Student Name: Muhammad Ghulam Abbas – 29417

Instructor: Sir Hafeez Khawaja

Project Overview

This project showcases the design and deployment of a scalable, secure, and containerized infrastructure on AWS using **Terraform**. The primary objective was to automate the provisioning of cloud resources and orchestrate the deployment of a full-stack Dockerized web application, complete with integrated database services, load balancing, custom domain setup, SSL encryption, and real-time business intelligence (BI) visualization.

By leveraging **Infrastructure as Code (IaC)** principles with Terraform, modular and reusable templates were developed to provision and manage critical AWS services. These include:

- **EC2 instances** deployed through **Auto Scaling Groups**
- **RDS instances** (PostgreSQL and MySQL) hosted in private subnets
- **Application Load Balancer (ALB)** configured with HTTPS and domain routing

The frontend and backend applications were containerized using **multi-stage Docker builds** and deployed across EC2 instances behind the ALB to ensure high availability and efficient load distribution.

Objective

The primary objective of this project is to build a **production-ready, scalable, and secure cloud infrastructure** on AWS using Infrastructure as Code (IaC) principles. The key goals include:

- Provisioning **Auto Scaling EC2 instances** pre-configured with **Nginx**, **Docker**, and **Node.js 20**
- Deploying **RDS databases** (MySQL and PostgreSQL) within **private subnets** to ensure data security
- Setting up an **Application Load Balancer (ALB)** with **HTTPS support** for secure traffic routing
- Deploying a **Business Intelligence (BI) tool – Metabase** on a dedicated EC2 instance
- Configuring a **custom domain** with **SSL encryption** using **AWS Certificate Manager (ACM)**
- Enabling **SSH tunneling** for secure access to the private RDS databases

Resources & References

- **Terraform Infrastructure (IaC) Repository:**
  https://github.com/MuhammadGhulamAbbas/Devops-Project
- **Frontend React Application Repository:**
  https://github.com/MuhammadGhulamAbbas/reactapp
- **Demonstration Video (Loom):**
  A Loom video walkthrough of the project implementation is provided in a text file located within the Devops GitHub repository.

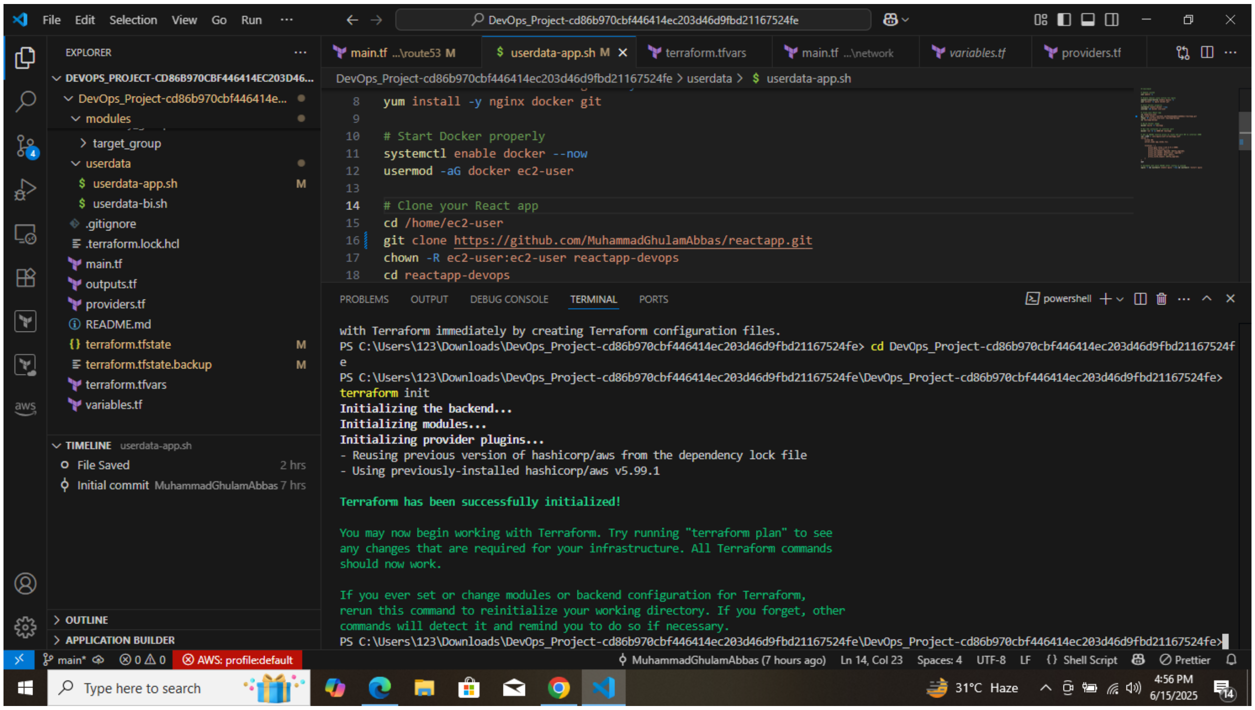
Project Structure (Modular Approach)

The project follows a modular Terraform structure to promote reusability, maintainability, and scalability. Below is the directory layout:

```
DevOps-Project/
├── main.tf
├── outputs.tf
├── providers.tf
├── terraform.tfvars
├── variables.tf
├── README.md

├── modules/
|   ├── network/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── security_groups/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── target_group/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── ec2/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── ec2-bi/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── rds/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── alb/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
```

```
|
|   ├── alb-bi/
|   |   ├── main.tf
|   |   ├── outputs.tf
|   |   └── variables.tf
|
|   ├── route53/
|       ├── main.tf
|       ├── outputs.tf
|       └── variables.tf

├── userdata/
|   ├── userdata-app.sh
|   └── userdata-bi.sh

├── docker/
|   └── Dockerfile
```

1.Terraform init

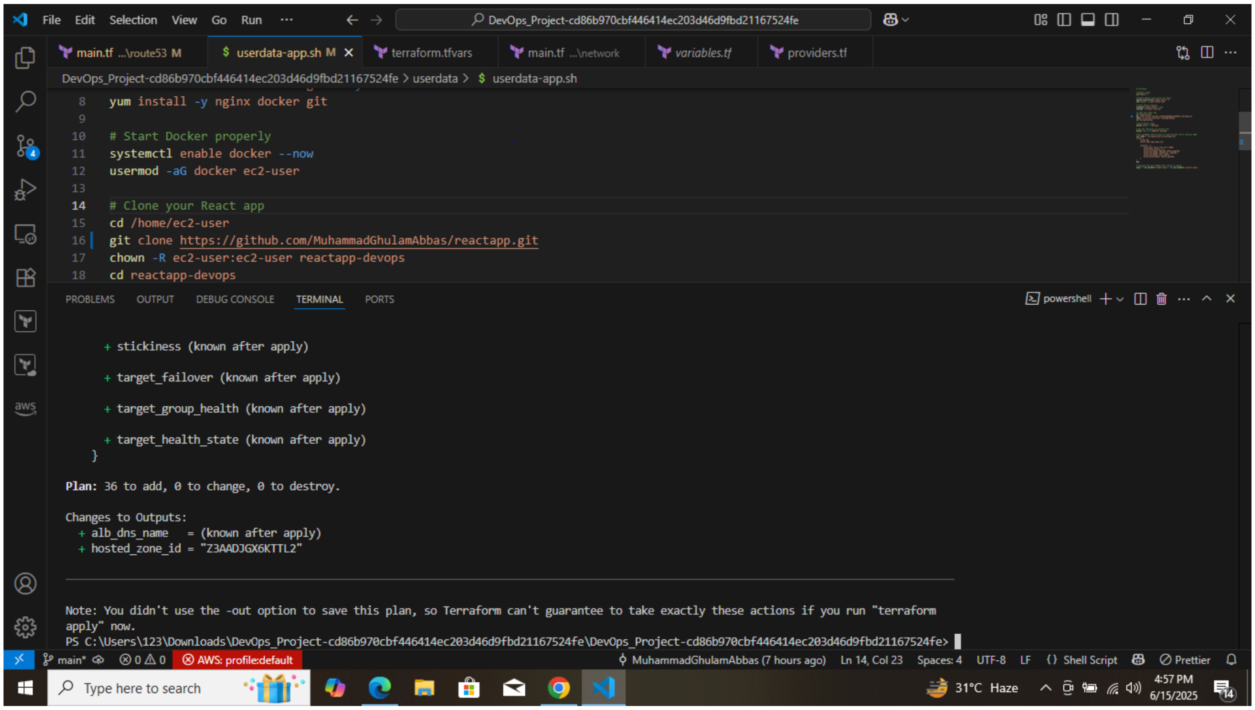


DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe > userdata > $ userdata-app.sh

```
yum install -y nginx docker git

# Start Docker properly
systemctl enable docker --now
usermod -aG docker ec2-user

# Clone your React app
cd /home/ec2-user
git clone https://github.com/MuhammadGhulamAbbas/reactapp.git
chown -R ec2-user:ec2-user reactapp-devops
cd reactapp-devops
```

```
with Terraform immediately by creating Terraform configuration files.
PS C:\Users\123\Downloads\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe> cd DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524f
e
PS C:\Users\123\Downloads\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe>
terraform init
Initializing the backend...
Initializing modules...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.99.1

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\123\Downloads\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe>
```

2.Terraform Plan

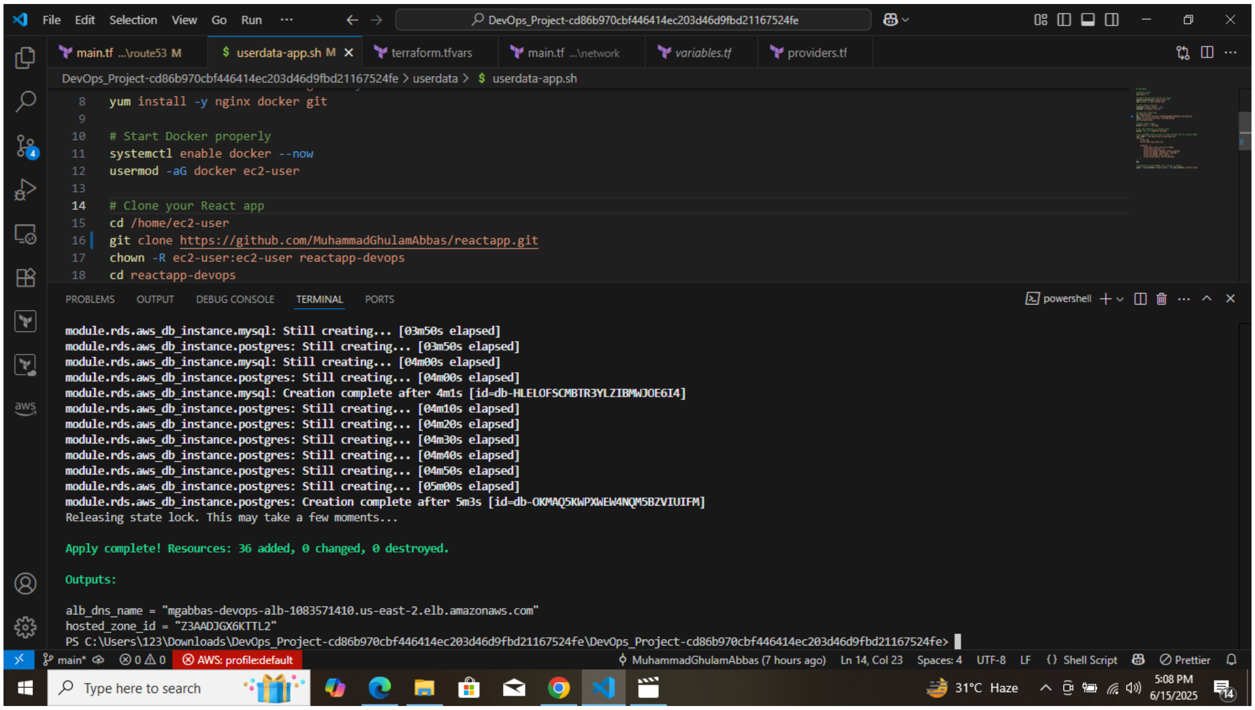


DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe > userdata > $ userdata-app.sh

```
yum install -y nginx docker git

# Start Docker properly
systemctl enable docker --now
usermod -aG docker ec2-user

# Clone your React app
cd /home/ec2-user
git clone https://github.com/MuhammadGhulamAbbas/reactapp.git
chown -R ec2-user:ec2-user reactapp-devops
cd reactapp-devops
```

```
      + stickiness (known after apply)

      + target_failover (known after apply)

      + target_group_health (known after apply)

      + target_health_state (known after apply)
    }

Plan: 36 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + alb_dns_name   = (known after apply)
  + hosted_zone_id = "Z3AADJGX6KTTL2"

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform
apply" now.
PS C:\Users\123\Downloads\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe>
```

3.Terraform apply

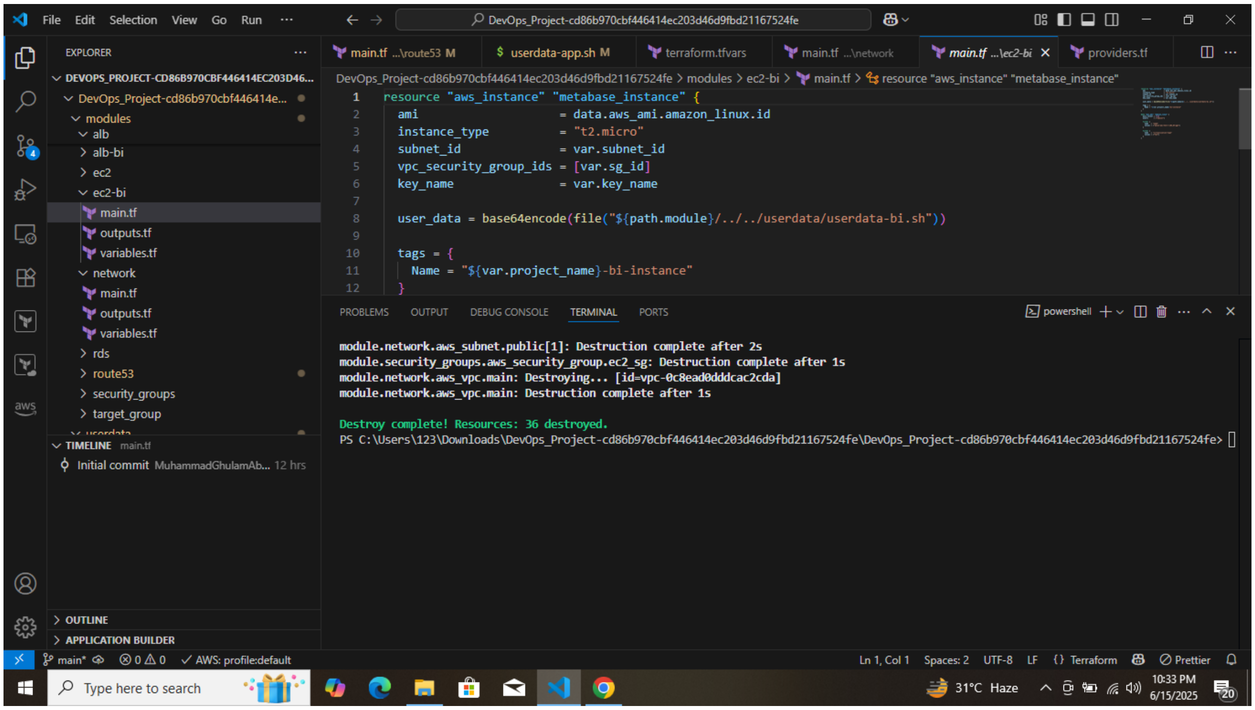


```
yum install -y nginx docker git

# Start Docker properly
systemctl enable docker --now
usermod -aG docker ec2-user

# Clone your React app
cd /home/ec2-user
git clone https://github.com/MuhammadGhulamAbbas/reactapp.git
chown -R ec2-user:ec2-user reactapp-devops
cd reactapp-devops
```

```
module.rds.aws_db_instance.mysql: Still creating... [03m50s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [03m50s elapsed]
module.rds.aws_db_instance.mysql: Still creating... [04m00s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [04m00s elapsed]
module.rds.aws_db_instance.mysql: Creation complete after 4m1s [id=db-HLELOFSCMBTR3YLZIBMWJOE6I4]
module.rds.aws_db_instance.postgres: Still creating... [04m10s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [04m20s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [04m30s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [04m40s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [04m50s elapsed]
module.rds.aws_db_instance.postgres: Still creating... [05m00s elapsed]
module.rds.aws_db_instance.postgres: Creation complete after 5m3s [id=db-OKMAQ5KWPXWEW4NQM5BZVIUIFM]
Releasing state lock. This may take a few moments...

Apply complete! Resources: 36 added, 0 changed, 0 destroyed.

Outputs:

alb_dns_name = "mgabbas-devops-alb-1083571410.us-east-2.elb.amazonaws.com"
hosted_zone_id = "Z3AADJGX6KTTL2"
PS C:\Users\123\Downloads\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe>
```

File Edit Selection View Go Run ···
DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe
main.tf ...\route53 M
userdata-app.sh M
terraform.tfvars
main.tf ...\network
variables.tf
providers.tf
DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe > userdata > userdata-app.sh
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
main* ⊗ 0 ⚠ 0 AWS: profile:default
MuhammadGhulamAbbas (7 hours ago) Ln 14, Col 23 Spaces: 4 UTF-8 LF {} Shell Script Prettier
Type here to search
31°C Haze
5:08 PM
6/15/2025

4.Terraform destroy



```
resource "aws_instance" "metabase_instance" {
  ami                    = data.aws_ami.amazon_linux.id
  instance_type          = "t2.micro"
  subnet_id              = var.subnet_id
  vpc_security_group_ids = [var.sg_id]
  key_name               = var.key_name

  user_data = base64encode(file("${path.module}/../../userdata/userdata-bi.sh"))

  tags = {
    Name = "${var.project_name}-bi-instance"
  }
```

```
module.network.aws_subnet.public[1]: Destruction complete after 2s
module.security_groups.aws_security_group.ec2_sg: Destruction complete after 1s
module.network.aws_vpc.main: Destroying... [id=vpc-0c8ead0dddcac2cda]
module.network.aws_vpc.main: Destruction complete after 1s

Destroy complete! Resources: 36 destroyed.
PS C:\Users\123\Downloads\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe\DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe>
```

File Edit Selection View Go Run ···
DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe
EXPLORER
DEVOPS_PROJECT-CD86B970CBF446414EC203D46...
DevOps_Project-cd86b970cbf446414e...
modules
alb
alb-bi
ec2
ec2-bi
main.tf
outputs.tf
variables.tf
network
main.tf
outputs.tf
variables.tf
rds
route53
security_groups
target_group
TIMELINE main.tf
Initial commit MuhammadGhulamAb... 12 hrs
OUTLINE
APPLICATION BUILDER
main.tf ...\route53 M
userdata-app.sh M
terraform.tfvars
main.tf ...\network
main.tf ...\ec2-bi
providers.tf
DevOps_Project-cd86b970cbf446414ec203d46d9fbd21167524fe > modules > ec2-bi > main.tf > resource "aws_instance" "metabase_instance"
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
main* ⊗ 0 ⚠ 0 ✓ AWS: profile:default
Ln 1, Col 1 Spaces: 2 UTF-8 LF {} Terraform Prettier
Type here to search
31°C Haze
10:33 PM
6/15/2025

Project Overview

1. EC2 Auto Scaling Group:

 A total of **3 EC2 instances** were launched using a **Launch Template** with Auto Scaling configuration for high availability and scalability.

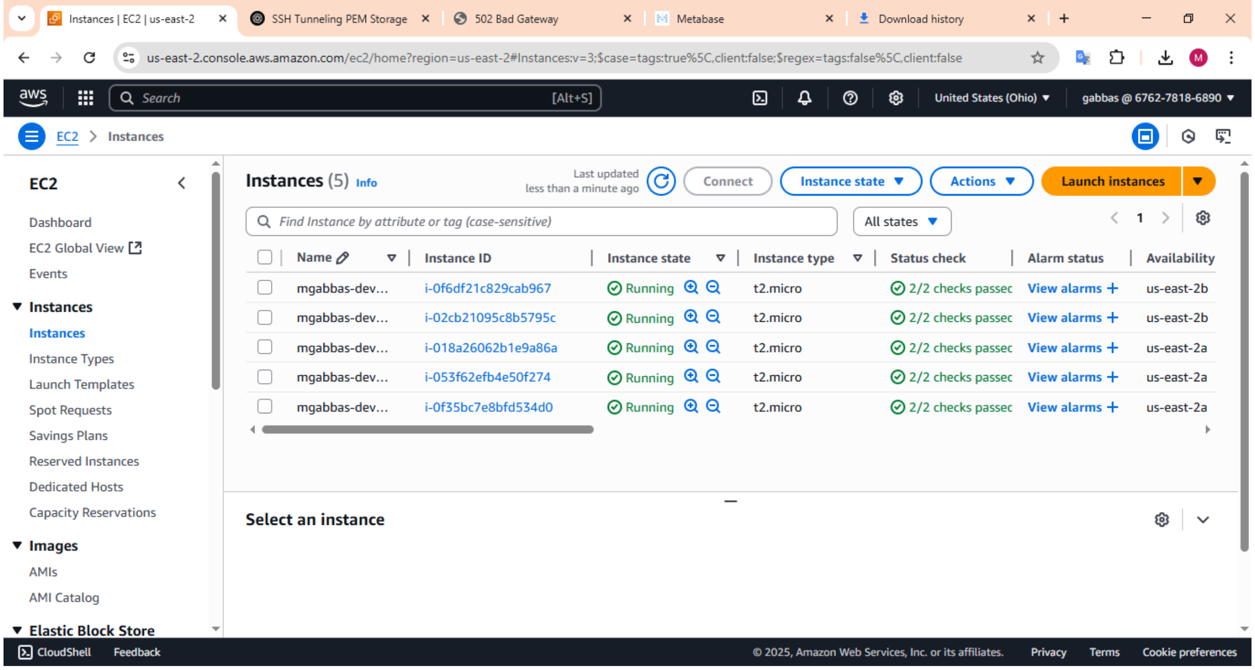
 User data scripts (userdata-app.sh and userdata-bi.sh) were used to automate the installation of required components on instance boot:

- **Nginx**
- **Docker**
- **Node.js 20**

 Instance Roles:

- **Two EC2 instances** were configured to host the Dockerized applications.
- **One EC2 instance** was dedicated to running **Metabase**, a Business Intelligence (BI) tool, containerized via Docker.

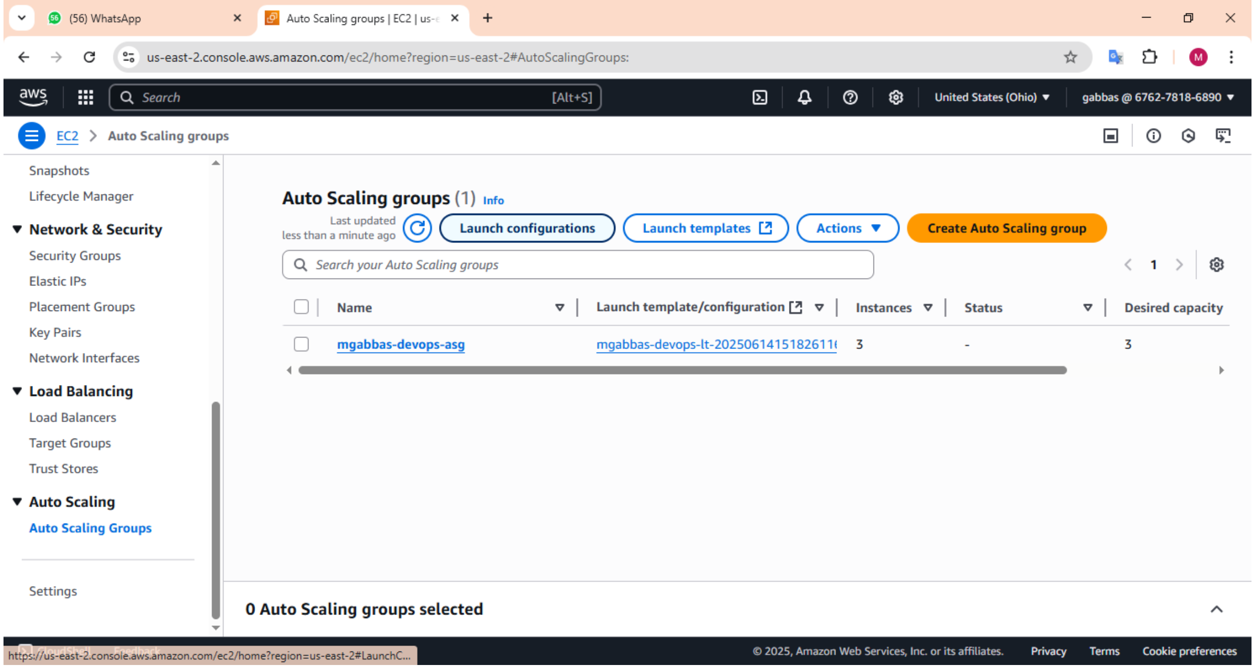
This image is instance in aws with their autoscaling group and template.



Instances (5) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
mgabbas-dev...	i-0f6df21c829cab967	Running	t2.micro	2/2 checks passec	View alarms +	us-east-2b
mgabbas-dev...	i-02cb21095c8b5795c	Running	t2.micro	2/2 checks passec	View alarms +	us-east-2b
mgabbas-dev...	i-018a26062b1e9a86a	Running	t2.micro	2/2 checks passec	View alarms +	us-east-2a
mgabbas-dev...	i-053f62efb4e50f274	Running	t2.micro	2/2 checks passec	View alarms +	us-east-2a
mgabbas-dev...	i-0f35bc7e8bfd534d0	Running	t2.micro	2/2 checks passec	View alarms +	us-east-2a

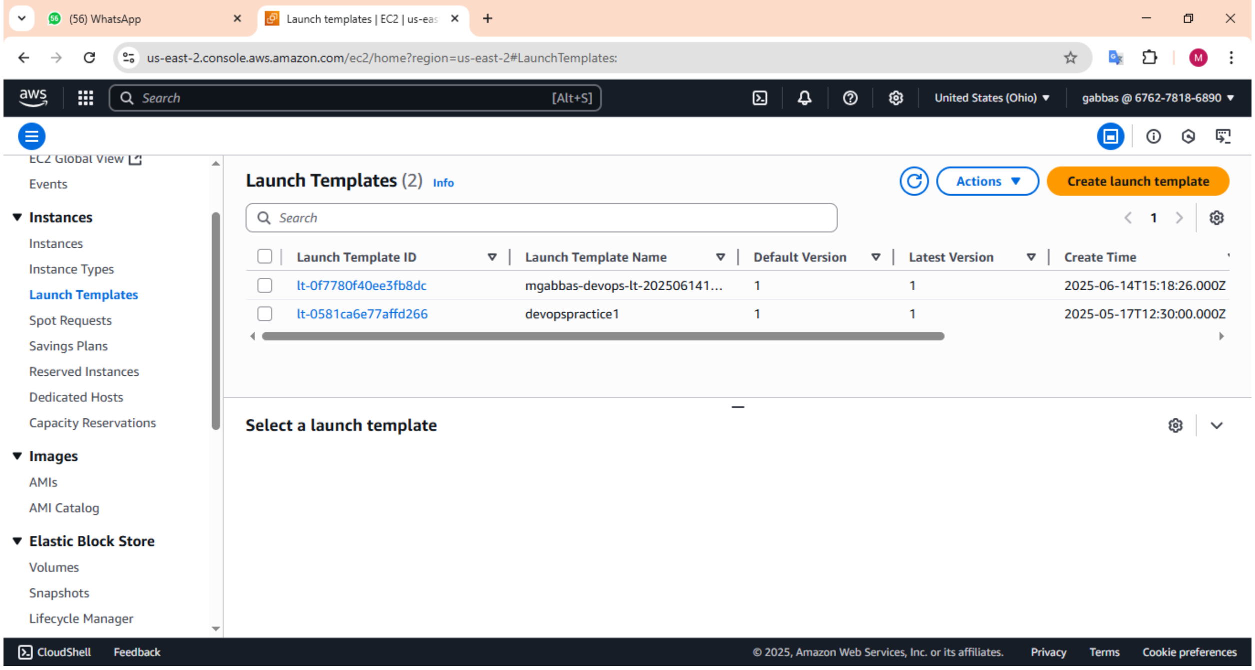
Select an instance



Auto Scaling groups (1) Info

Name	Launch template/configuration	Instances	Status	Desired capacity
mgabbas-devops-asg	mgabbas-devops-lt-2025061415182611(	3	-	3

0 Auto Scaling groups selected



Launch Templates (2) Info

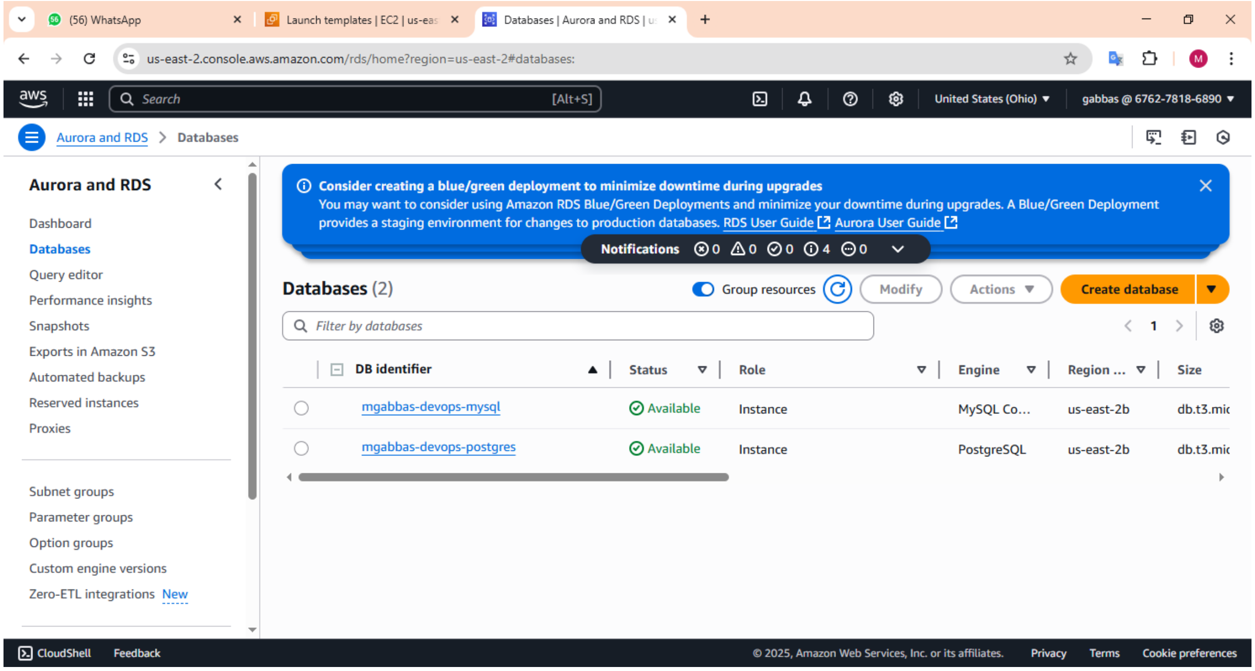
□	Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time
□	lt-0f7780f40ee3fb8dc	mgabbas-devops-lt-202506141…	1	1	2025-06-14T15:18:26.000Z
□	lt-0581ca6e77affd266	devopspractice1	1	1	2025-05-17T12:30:00.000Z

Select a launch template

2. RDS Instances

- **Two RDS instances** were provisioned:
  - **1 MySQL**
  - **1 PostgreSQL**
- Both databases were deployed in **private subnets** to ensure network-level security and prevent direct public access.
- **Secure connectivity** was established using **SSH tunneling** through the application EC2 instance, allowing access for development and monitoring without exposing the databases to the internet.
- **Terraform configurations** were used to define:
  - **DB Subnet Groups** to control the network placement of the RDS instances
  - **Parameter Groups** to customize engine-level settings for performance and compatibility

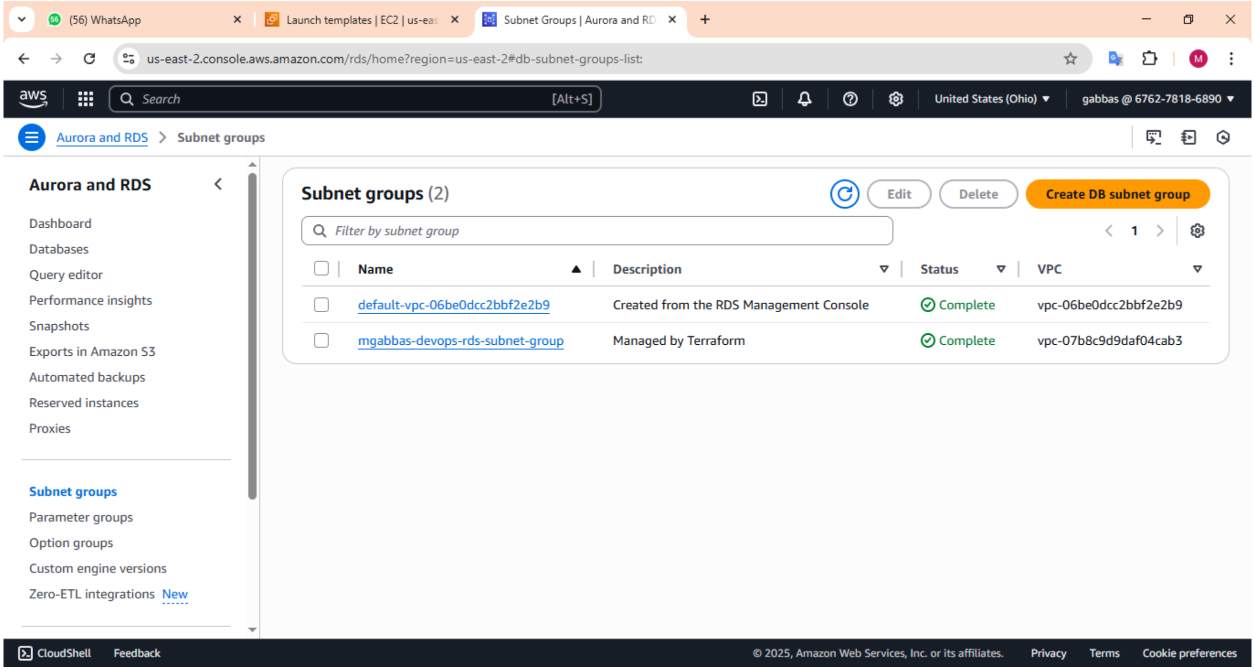
This images is RDS instance in aws with their subnet group.



Consider creating a blue/green deployment to minimize downtime during upgrades
You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. RDS User Guide Aurora User Guide

Databases (2)

	DB identifier	Status	Role	Engine	Region ...	Size
○	mgabbas-devops-mysql	✓ Available	Instance	MySQL Co...	us-east-2b	db.t3.mic
○	mgabbas-devops-postgres	✓ Available	Instance	PostgreSQL	us-east-2b	db.t3.mic



Subnet groups (2)

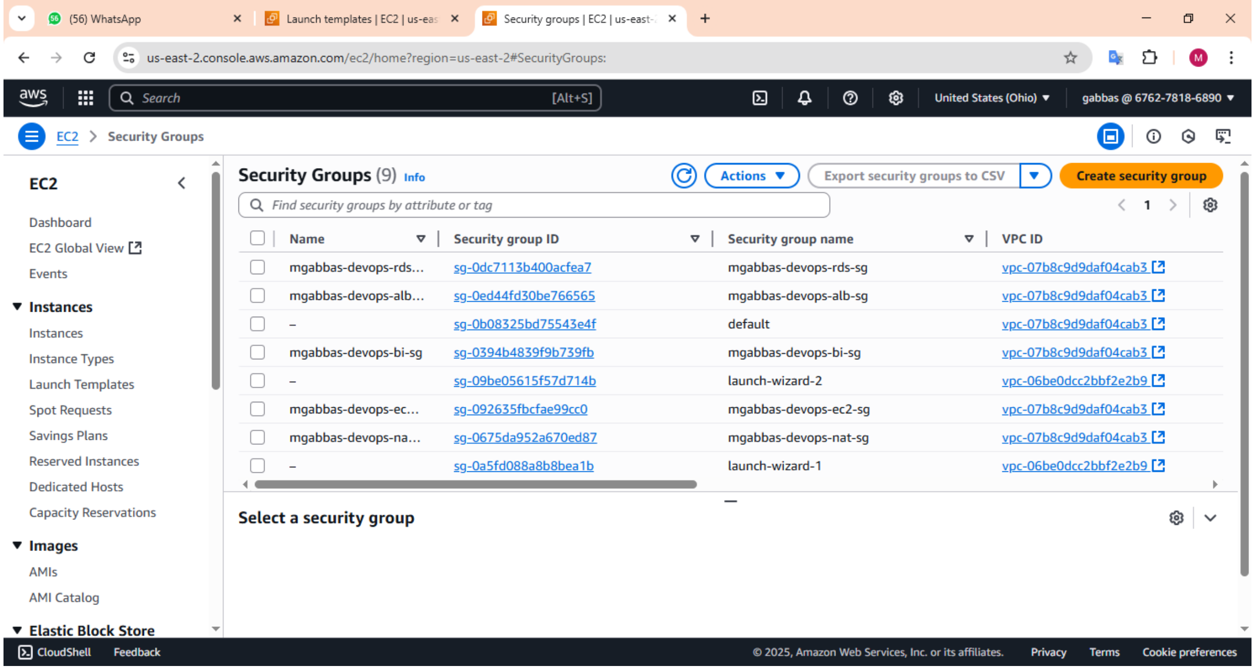
Name	Description	Status	VPC
default-vpc-06be0dcc2bbf2e2b9	Created from the RDS Management Console	Complete	vpc-06be0dcc2bbf2e2b9
mgabbas-devops-rds-subnet-group	Managed by Terraform	Complete	vpc-07b8c9d9daf04cab3

3. Security Groups

Proper security group configurations were implemented to control traffic flow between resources while maintaining security best practices:

- **EC2 Security Group**
  - **Ingress Rules:**
    - Port **22** – SSH access (restricted to developer IP)
    - Port **80** – HTTP traffic for web access
    - Port **443** – HTTPS traffic for secure web access
- **RDS Security Group**
  - **Ingress Rules:**
    - Port **3306** – MySQL access
    - Port **5432** – PostgreSQL access
  - **Access Scope:**
    - Inbound connections allowed **only from EC2 instances** (using security group referencing)
- **Application Load Balancer (ALB) Security Group**
  - **Ingress Rules:**
    - Port **80** – HTTP open to the public
    - Port **443** – HTTPS open to the public
- **Egress Rules (All Security Groups):**
  - **All traffic (0.0.0.0/0)** allowed outbound to enable internet connectivity and service dependencies.

- **This images is Aws security Groups.**



Security Groups (9)

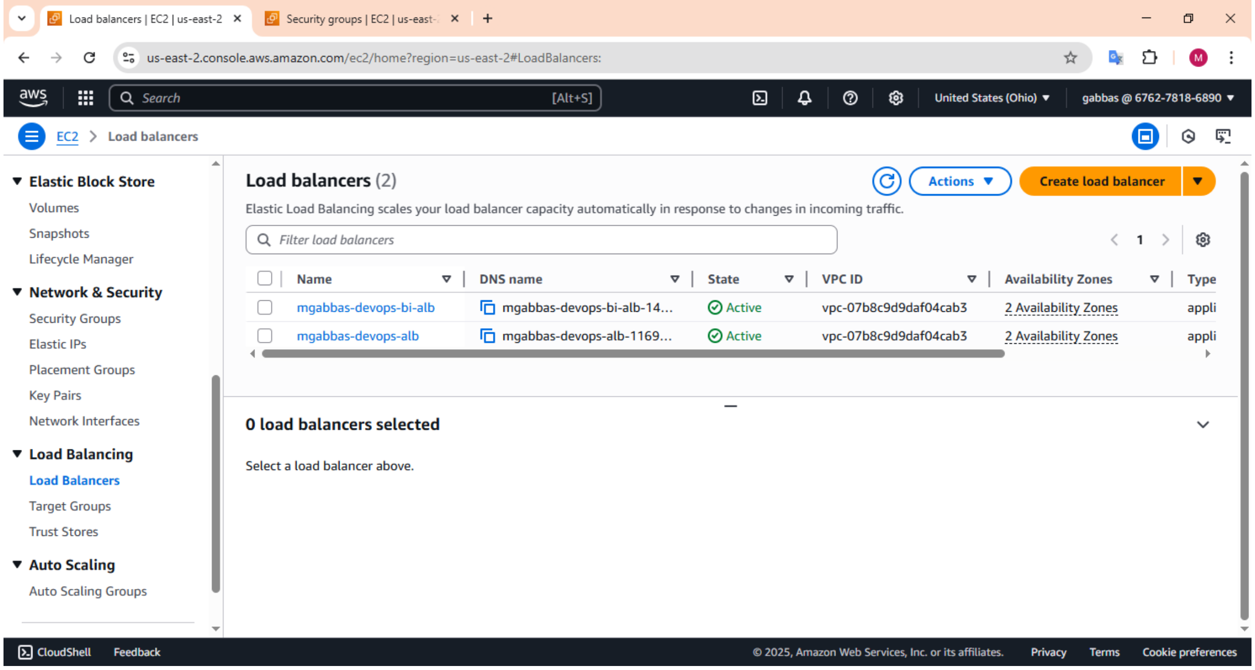
Name	Security group ID	Security group name	VPC ID
mgabbas-devops-rds...	sg-0dc7113b400acfea7	mgabbas-devops-rds-sg	vpc-07b8c9d9daf04cab3
mgabbas-devops-alb...	sg-0ed44fd30be766565	mgabbas-devops-alb-sg	vpc-07b8c9d9daf04cab3
–	sg-0b08325bd75543e4f	default	vpc-07b8c9d9daf04cab3
mgabbas-devops-bi-sg	sg-0394b4839f9b739fb	mgabbas-devops-bi-sg	vpc-07b8c9d9daf04cab3
–	sg-09be05615f57d714b	launch-wizard-2	vpc-06be0dcc2bbf2e2b9
mgabbas-devops-ec...	sg-092635fbcfae99cc0	mgabbas-devops-ec2-sg	vpc-07b8c9d9daf04cab3
mgabbas-devops-na...	sg-0675da952a670ed87	mgabbas-devops-nat-sg	vpc-07b8c9d9daf04cab3
–	sg-0a5fd088a8b8bea1b	launch-wizard-1	vpc-06be0dcc2bbf2e2b9

Select a security group

4. load Balancer (ALB)

A highly available and secure **Application Load Balancer (ALB)** was provisioned using Terraform to distribute incoming traffic across EC2 instances.

- **Deployment & Configuration:**
  - The ALB was deployed via Terraform within the public subnets.
  - Configured to handle both **HTTP (port 80)** and **HTTPS (port 443)** traffic.
- **HTTPS Enforcement:**
  - All HTTP requests on port 80 are **automatically redirected to HTTPS (port 443)** to enforce secure communication.
  - **SSL/TLS certificates** were provisioned using **AWS Certificate Manager (ACM)** for encrypted traffic.
- **Listener Rules & Routing:**
  - Listener rules were defined to **forward traffic to specific target groups** based on the **subdomain or path**.
  - Ensures that frontend, backend, and BI services can be routed independently and securely.
- **This images is load balancer Groups with their overview which tell health of specific load balancer for example bi alb with their target group which tells their configuration.**



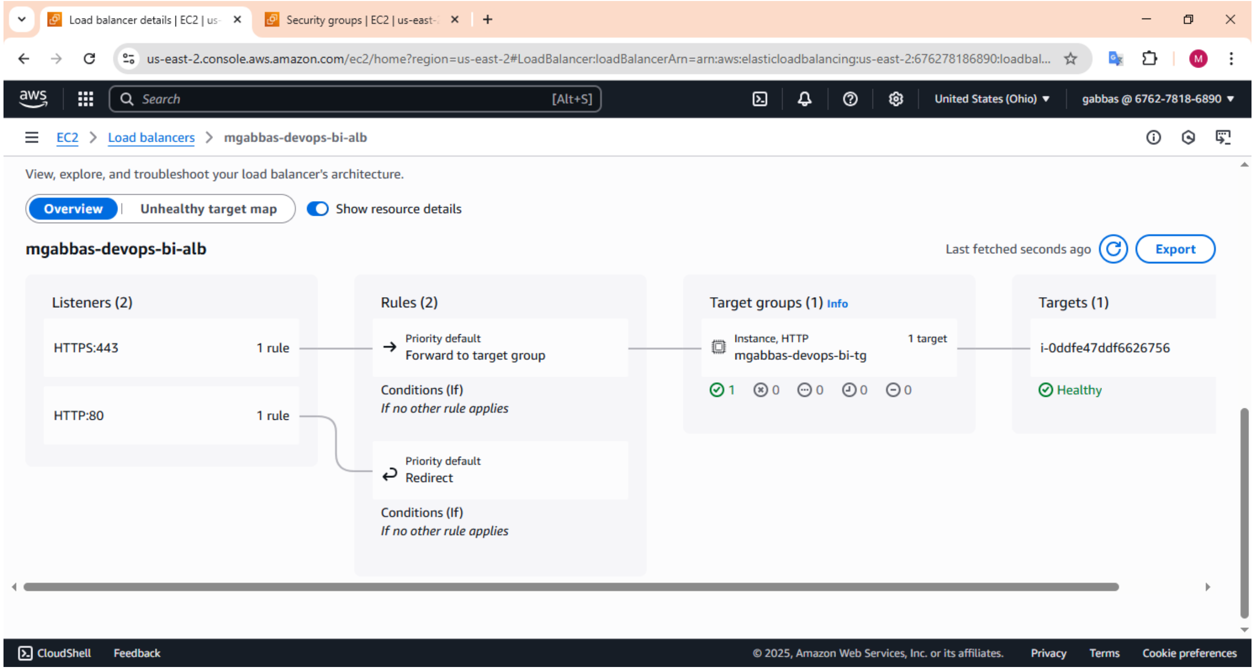
Load balancers (2)

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

□	Name	DNS name	State	VPC ID	Availability Zones	Type
□	mgabbas-devops-bi-alb	mgabbas-devops-bi-alb-14...	✓ Active	vpc-07b8c9d9daf04cab3	2 Availability Zones	appli
□	mgabbas-devops-alb	mgabbas-devops-alb-1169...	✓ Active	vpc-07b8c9d9daf04cab3	2 Availability Zones	appli

0 load balancers selected

Select a load balancer above.



Load balancer details | EC2 | us-
Security groups | EC2 | us-east-
us-east-2.console.aws.amazon.com/ec2/home?region=us-east-2#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:us-east-2:676278186890:loadbal...
Search
[Alt+S]
United States (Ohio) ▾
gabbas @ 6762-7818-6890 ▾
EC2 > Load balancers > mgabbas-devops-bi-alb
Overview
Unhealthy target map
Show resource details
Last fetched seconds ago
Export
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

View, explore, and troubleshoot your load balancer's architecture.

mgabbas-devops-bi-alb

Listeners (2)

- HTTPS:443 — 1 rule
- HTTP:80 — 1 rule

Rules (2)

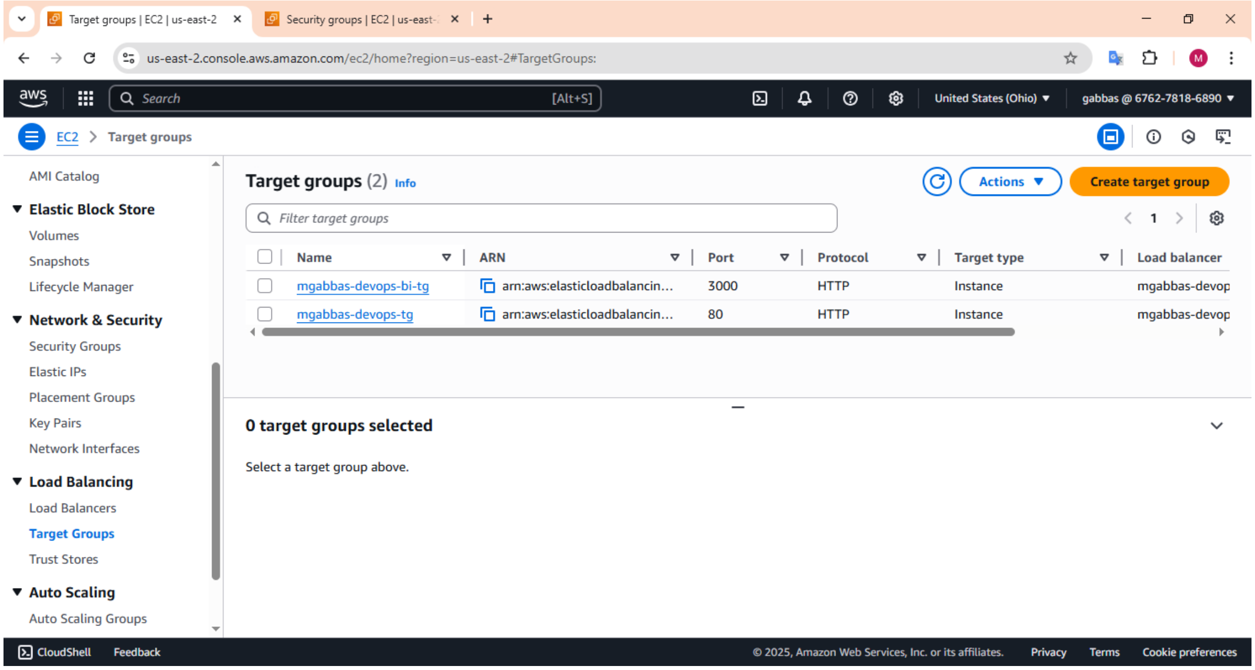
- Priority default
  Forward to target group
  Conditions (If)
  If no other rule applies
- Priority default
  Redirect
  Conditions (If)
  If no other rule applies

Target groups (1) Info

- Instance, HTTP — 1 target
  mgabbas-devops-bi-tg
  ✓ 1 ⊗ 0 ⊖ 0 ◷ 0 ⊖ 0

Targets (1)

- i-0ddfe47ddf6626756
  ✓ Healthy



Target groups (2) Info

□	Name	ARN	Port	Protocol	Target type	Load balancer
□	mgabbas-devops-bi-tg	arn:aws:elasticloadbalancin…	3000	HTTP	Instance	mgabbas-devop
□	mgabbas-devops-tg	arn:aws:elasticloadbalancin…	80	HTTP	Instance	mgabbas-devop

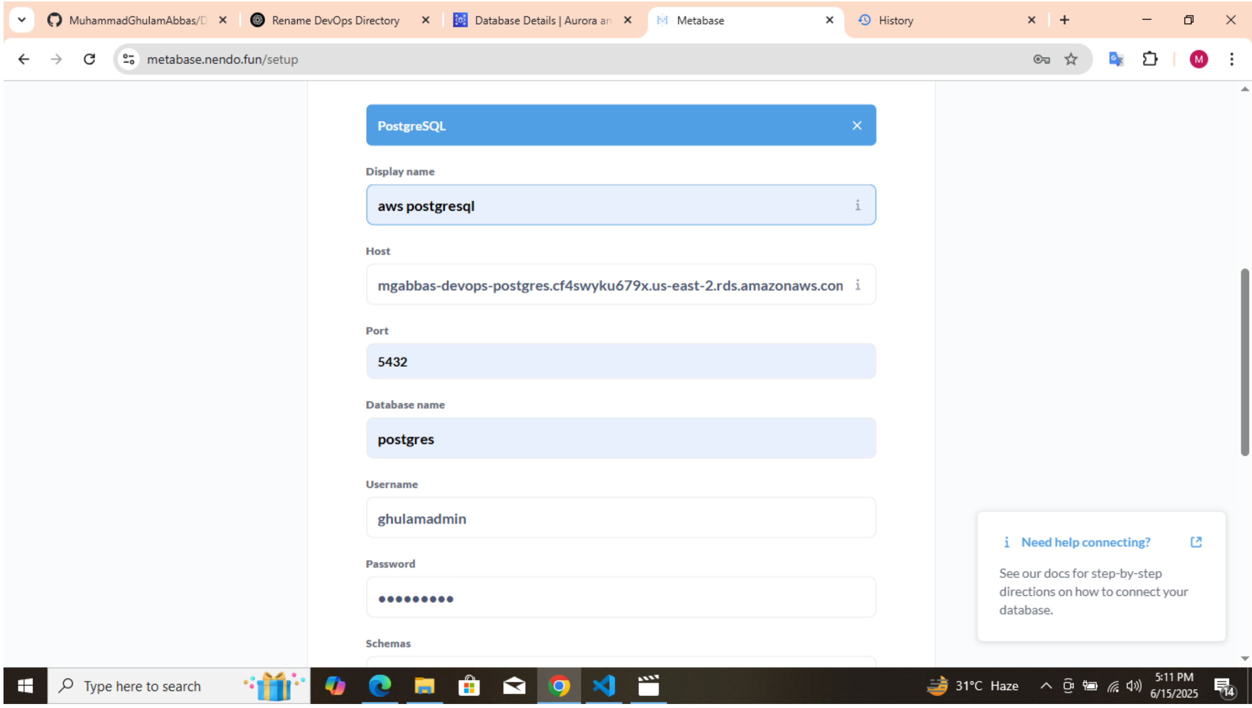
0 target groups selected

Select a target group above.

5. Database Access and Initialization

To ensure secure and controlled access to the RDS databases, an **SSH tunneling approach** was used for connectivity and data initialization:

- **Secure Access via SSH Tunnel:**
    - Direct access to RDS instances is **blocked from the public internet**.
    - An **SSH tunnel** is created through a publicly accessible **EC2 instance** (bastion or app server) to connect to the **MySQL** and **PostgreSQL** databases located in private subnets.
- **Database Client:**
    - Tools such as **DBeaver** or **pgAdmin** were used to connect to the databases through the SSH tunnel.
    - This setup supports secure management and monitoring without compromising security best practices.
- **Data Initialization:**
    - Both RDS instances were **populated with dummy data** to simulate application behavior and enable integration testing.
    - This data was later visualized through **Metabase** to demonstrate real-time BI insights.



PostgreSQL

Display name: aws postgresql

Host: mgabbas-devops-postgres.cf4swyku679x.us-east-2.rds.amazonaws.con

Port: 5432

Database name: postgres

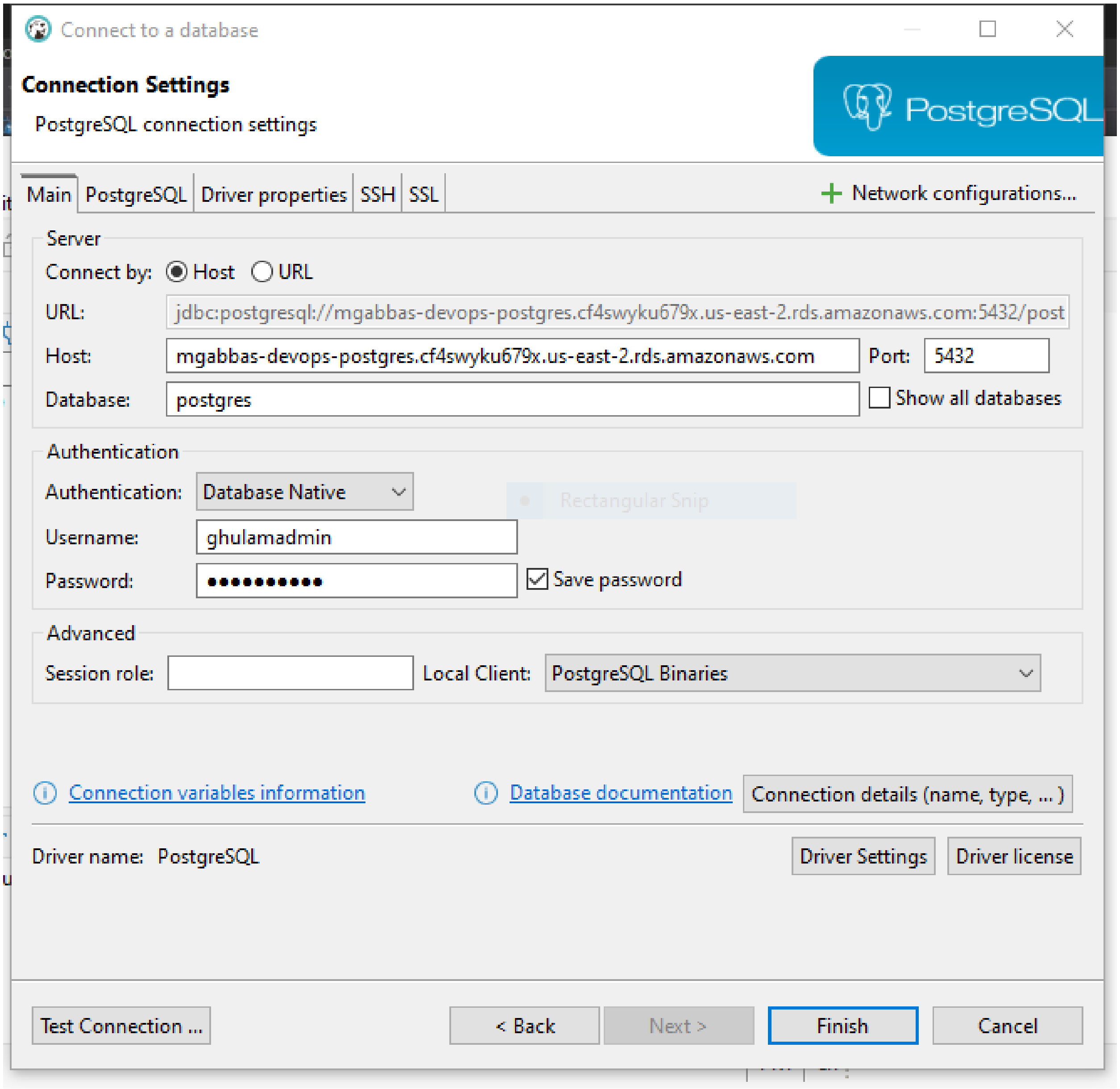
Username: ghulamadmin

Password: ●●●●●●●●●

Schemas

Need help connecting?

See our docs for step-by-step directions on how to connect your database.



Connection Settings

PostgreSQL connection settings

Server

Connect by: ● Host ○ URL

URL: jdbc:postgresql://mgabbas-devops-postgres.cf4swyku679x.us-east-2.rds.amazonaws.com:5432/post

Host: mgabbas-devops-postgres.cf4swyku679x.us-east-2.rds.amazonaws.com Port: 5432

Database: postgres □ Show all databases

Authentication

Authentication: Database Native

Username: ghulamadmin

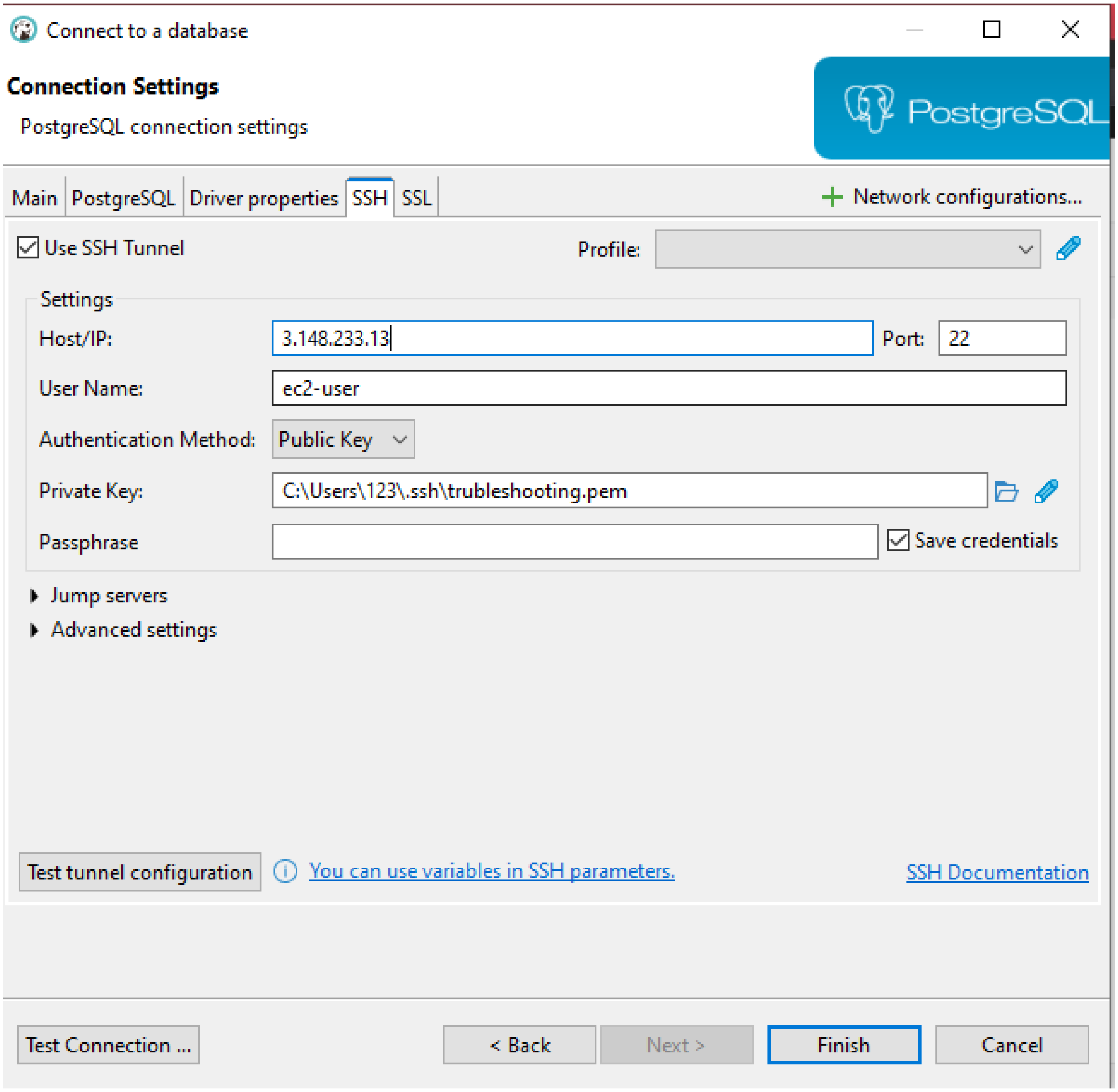
Password: ●●●●●●●●●●● ☑ Save password

Advanced

Session role: Local Client: PostgreSQL Binaries

Driver name: PostgreSQL

Host name is RDS database Username and Database is in tf variables file



☑ Use SSH Tunnel

Profile:

Settings

Host/IP: 3.148.233.13

Port: 22

User Name: ec2-user

Authentication Method: Public Key

Private Key: C:\Users\123\.ssh\trubleshooting.pem

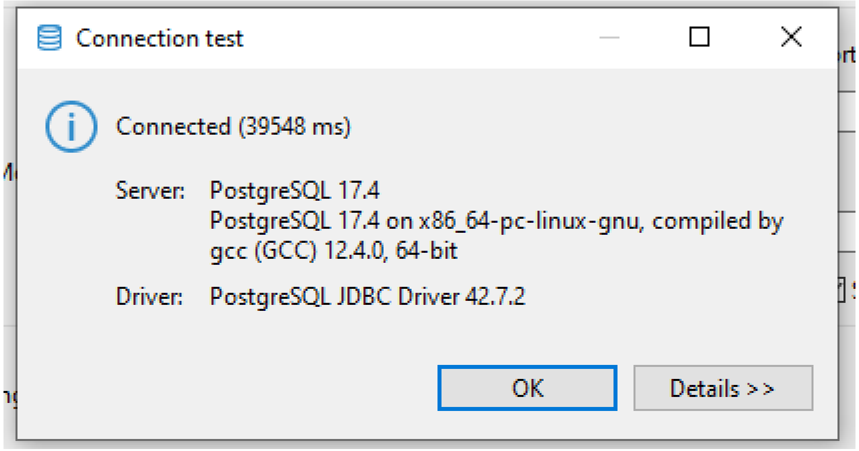
Passphrase

☑ Save credentials

SSH Tunneling Configuration

To access RDS instances securely, **SSH tunneling** was configured as follows:

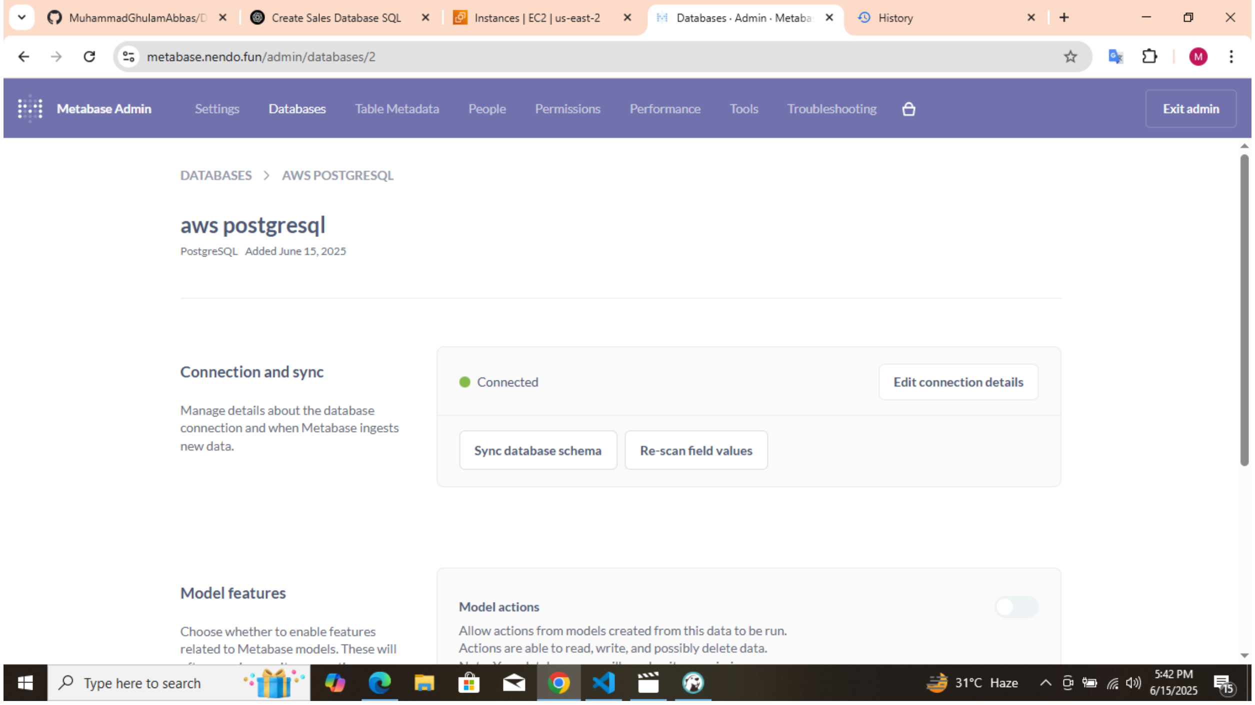
- The **PEM (private key) file** required for SSH authentication was selected from the **user's local drive (C:)**, specifically from the `.ssh` directory.
- SSH tunneling was established through the **relevant EC2 instance** that resides in the **public subnet** and has access to the private RDS instances.
- This allows developers and administrators to securely connect to the RDS databases using tools like **DBeaver**, without exposing the databases to public IPs



Connected (39548 ms)

Server: PostgreSQL 17.4
PostgreSQL 17.4 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 12.4.0, 64-bit

Driver: PostgreSQL JDBC Driver 42.7.2



aws postgresql

PostgreSQL Added June 15, 2025

Connection and sync

Manage details about the database connection and when Metabase ingests new data.

● Connected

Model features

Choose whether to enable features related to Metabase models. These will

Model actions

Allow actions from models created from this data to be run. Actions are able to read, write, and possibly delete data.

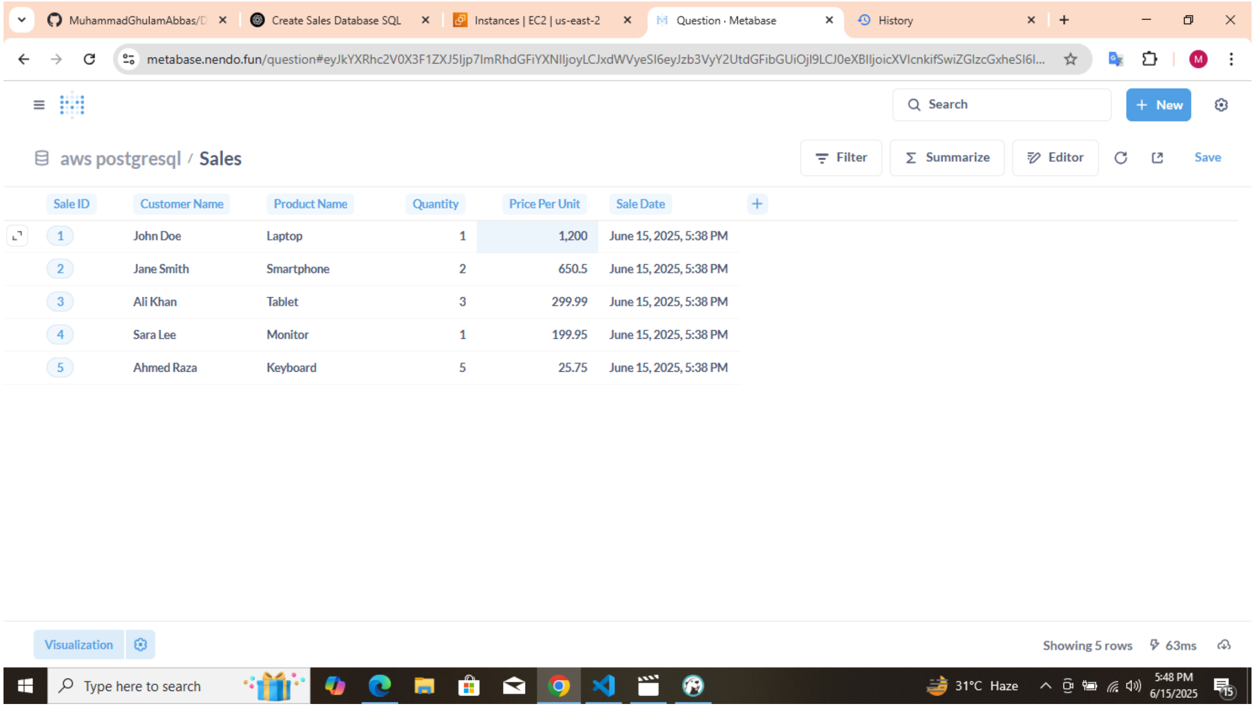
This is aws postgresql database

6. BI Tool Deployment – Metabase

A **Business Intelligence (BI) tool**, **Metabase**, was deployed on a **dedicated third EC2 instance** to enable real-time analytics and data visualization.

- **Tool Selection:**

  - **Metabase** was chosen over alternatives like Redash due to its ease of deployment, intuitive interface, and strong PostgreSQL support.
- **Deployment Method:**
  - The Metabase server was containerized using **Docker** and launched via a user data script (`userdata-bi.sh`) during EC2 instance provisioning.
  - The Docker image was pulled and executed automatically as part of the instance initialization process.
- **Database Connectivity:**
  - Metabase was securely connected to the **PostgreSQL RDS instance** hosted in a private subnet.
  - SSH tunneling or internal security group communication enabled private access from Metabase to the database.

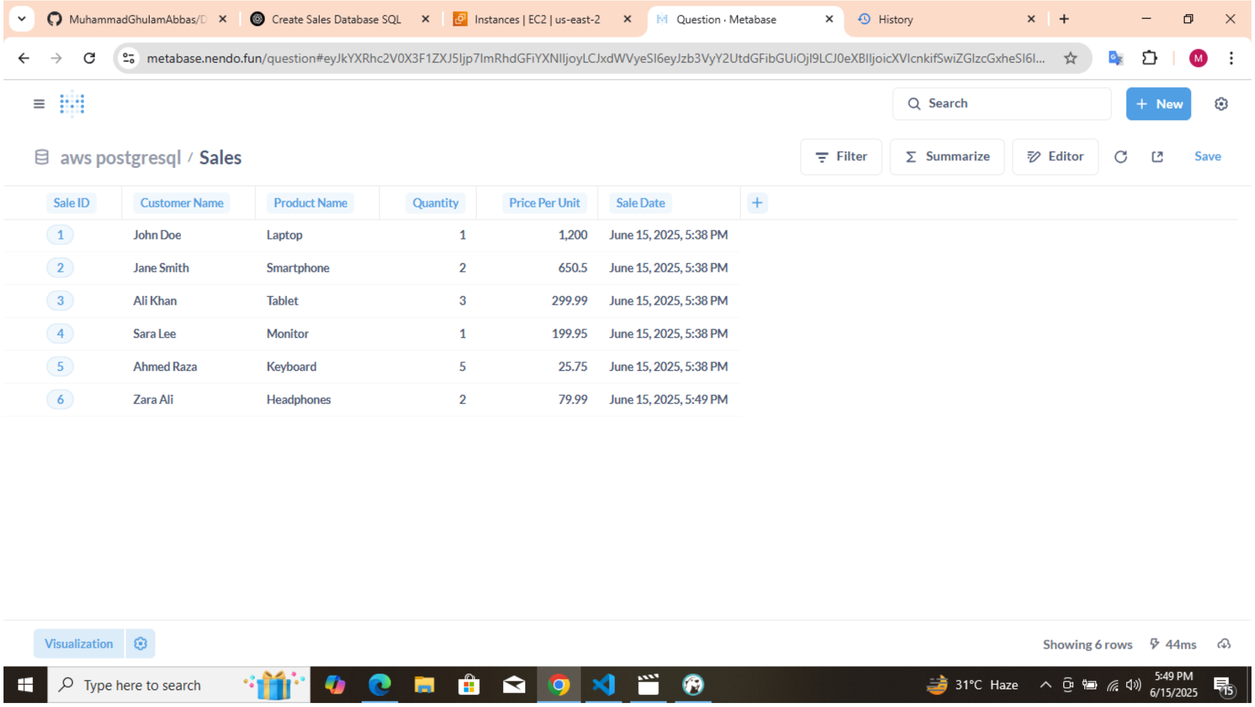


metabase.nendo.fun/question#eyJkYXRhc2V0X3F1ZXJ5Ijp7ImRhdGFiYXNlIjoyLCJxdWVyeSI6eyJzb3VyY2UtdGFibGUiOjI9LCJ0eXBlIjoicXVlcnkifSwiZGlzcGxheSI6I...

aws postgresql / Sales

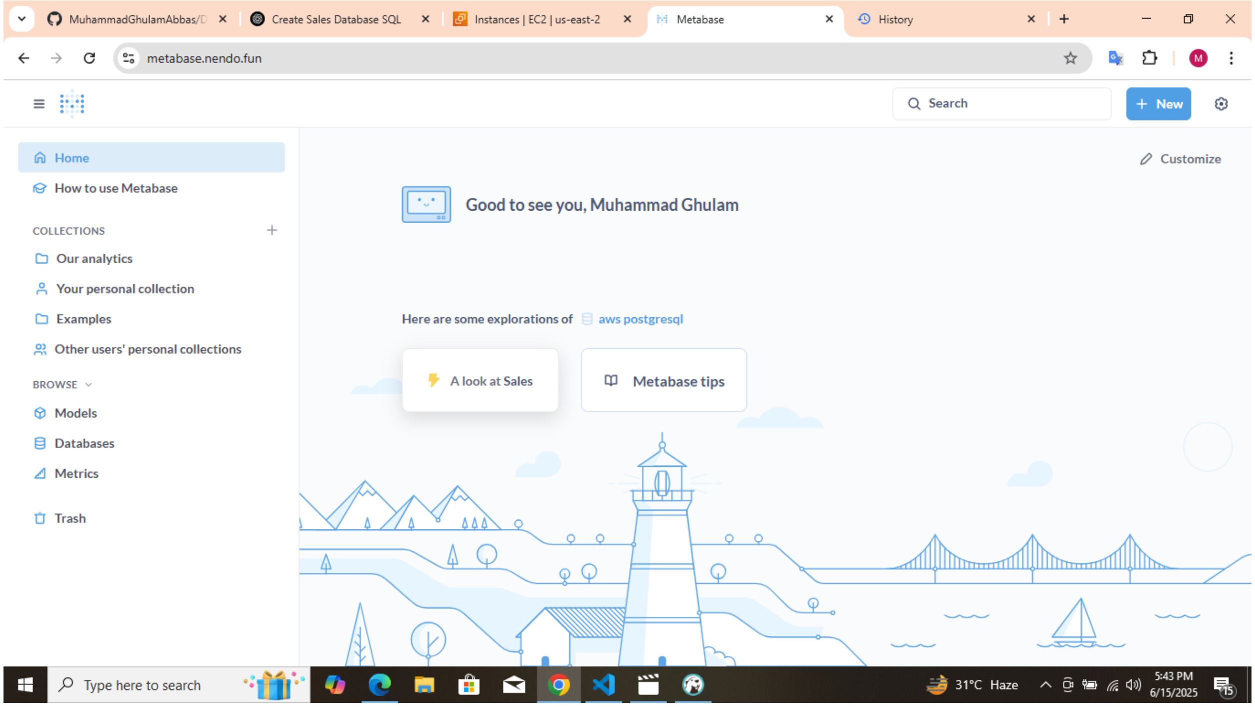
Sale ID	Customer Name	Product Name	Quantity	Price Per Unit	Sale Date
1	John Doe	Laptop	1	1,200	June 15, 2025, 5:38 PM
2	Jane Smith	Smartphone	2	650.5	June 15, 2025, 5:38 PM
3	Ali Khan	Tablet	3	299.99	June 15, 2025, 5:38 PM
4	Sara Lee	Monitor	1	199.95	June 15, 2025, 5:38 PM
5	Ahmed Raza	Keyboard	5	25.75	June 15, 2025, 5:38 PM

. This is sales database created in metabase with access dbeaver

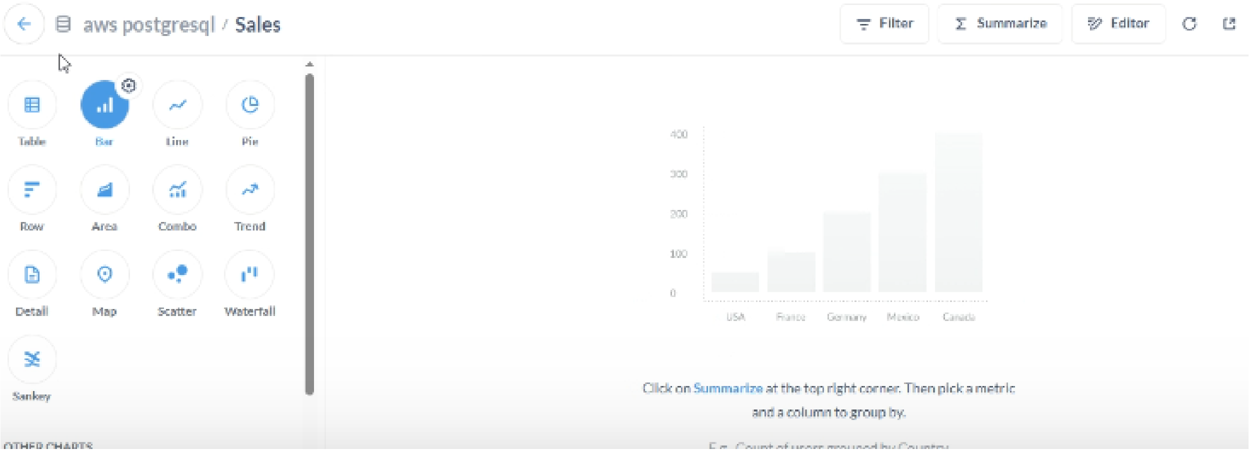

MuhammadGhulamAbbas/D
Create Sales Database SQL
Instances | EC2 | us-east-2
Question · Metabase
History
metabase.nendo.fun/question#eyJkYXRhc2V0X3F1ZXJ5Ijp7ImRhdGFiYXNlIjoyLCJxdWVyeSI6eyJzb3VyY2UtdGFibGUiOjI9LCJ0eXBlIjoicXVlcnkifSwiZGlzcGxheSI6I...
Search
New
aws postgresql / Sales
Filter
Summarize
Editor
Save
Visualization
Showing 6 rows
44ms
Type here to search
31°C Haze
5:49 PM
6/15/2025

Sale ID	Customer Name	Product Name	Quantity	Price Per Unit	Sale Date
1	John Doe	Laptop	1	1,200	June 15, 2025, 5:38 PM
2	Jane Smith	Smartphone	2	650.5	June 15, 2025, 5:38 PM
3	Ali Khan	Tablet	3	299.99	June 15, 2025, 5:38 PM
4	Sara Lee	Monitor	1	199.95	June 15, 2025, 5:38 PM
5	Ahmed Raza	Keyboard	5	25.75	June 15, 2025, 5:38 PM
6	Zara Ali	Headphones	2	79.99	June 15, 2025, 5:49 PM

After live update one row changed



This is metabase



Click on Summarize at the top right corner. Then pick a metric and a column to group by.

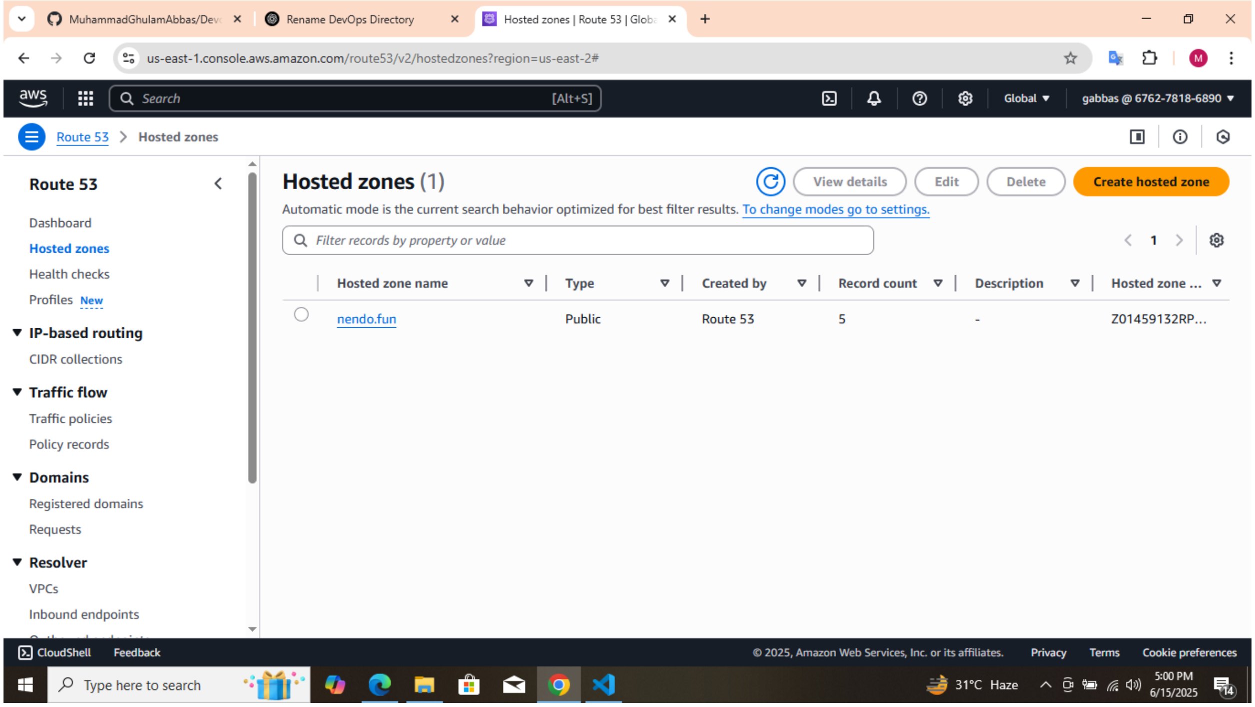
E.g., Count of users grouped by Country

Live analytics can be performed in metabase.

7. Domain & SSL Configuration

To ensure secure and professional access to the deployed services, **custom domain names** and **SSL encryption** were configured for both the application and the BI tool.

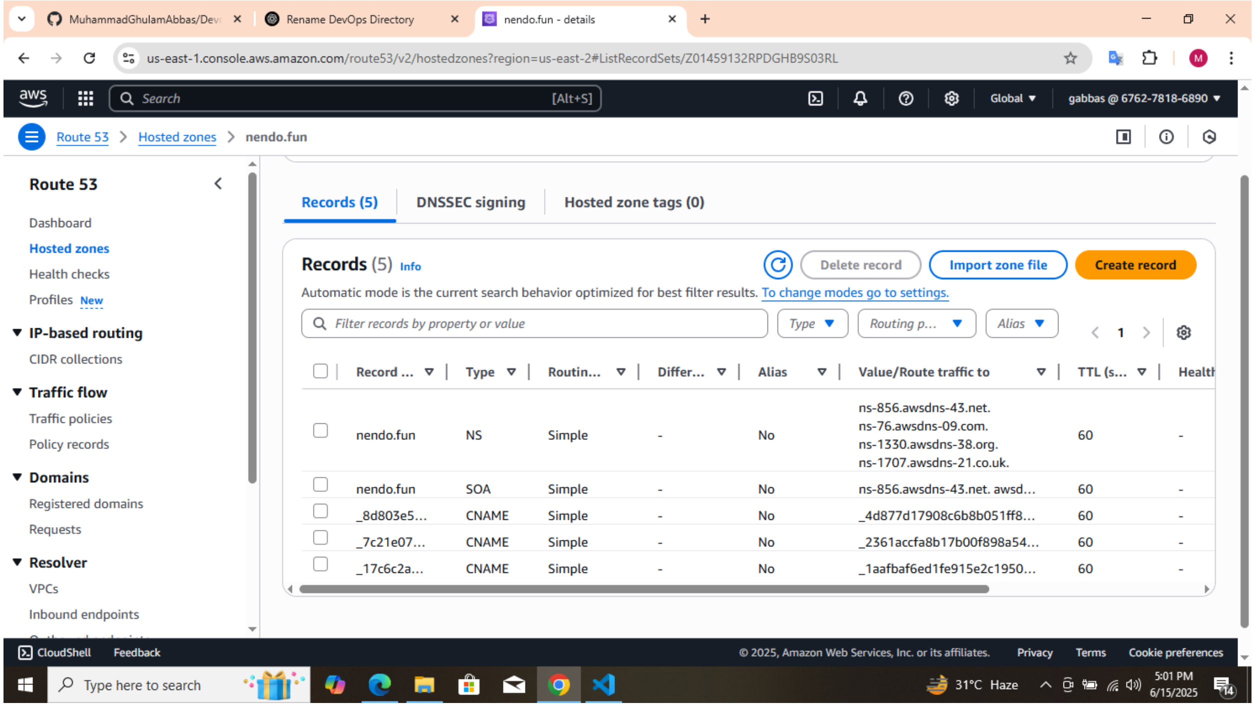
- **Custom Domain Integration:**
  - A domain was registered and managed via **Amazon Route 53**.
  - **DNS records (A and CNAME)** were created to point the custom domain and its subdomains to:
    - The **Application Load Balancer (ALB)** hosting the frontend/backend
    - The **EC2 instance** running the Metabase BI tool
- **SSL Configuration:**
  - **AWS Certificate Manager (ACM)** was used to provision **SSL/TLS certificates** for both domain endpoints.
  - Certificates were **attached to the ALB**, enabling **HTTPS support** for secure client communication.
  - Optionally, **Let's Encrypt** can also be used for EC2-hosted services like Metabase, if required.
- **HTTPS Enforcement:**
  - **Listener rules** on the ALB automatically **redirect HTTP (port 80) to HTTPS (port 443)**.
  - This ensures all traffic is encrypted, complying with security best practices and enhancing user trust.



Hosted zones (1)

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

Hosted zone name	Type	Created by	Record count	Description	Hosted zone ...
nendo.fun	Public	Route 53	5	-	Z01459132RP...



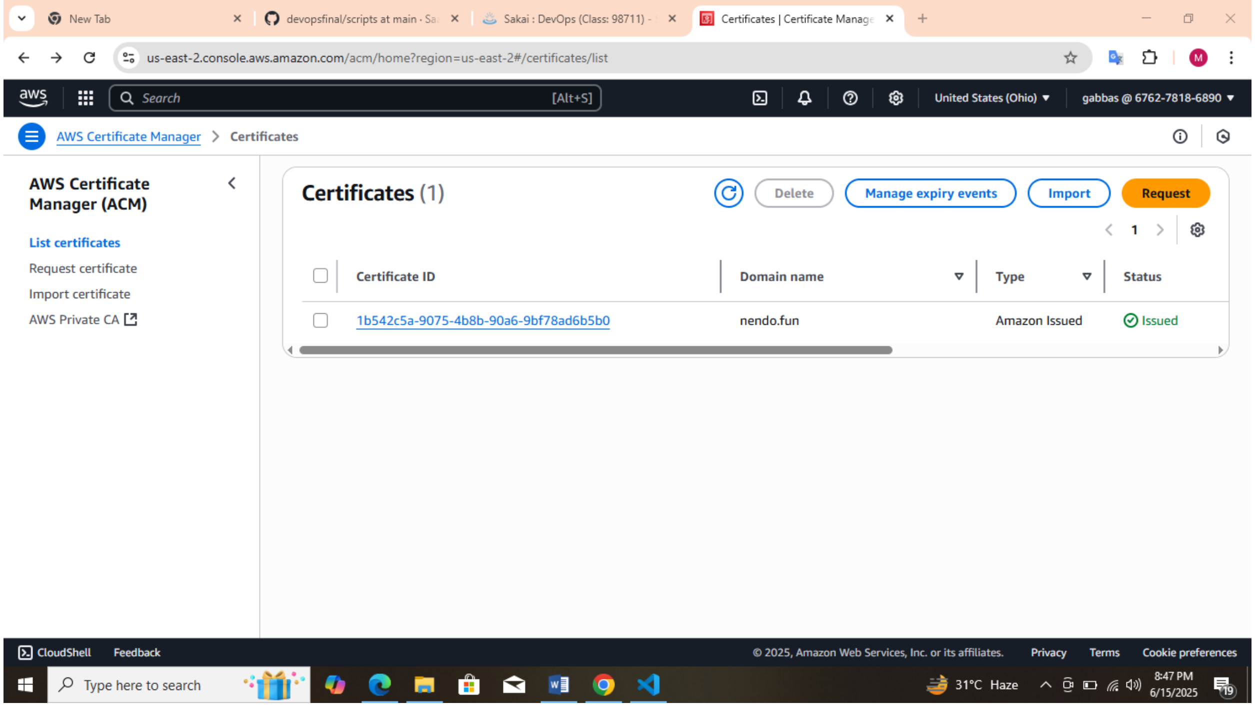
MuhammadGhulamAbbas/Dev
Rename DevOps Directory
nendo.fun - details
us-east-1.console.aws.amazon.com/route53/v2/hostedzones?region=us-east-2#ListRecordSets/Z01459132RPDGHB9S03RL
Search [Alt+S]
Global
gabbas @ 6762-7818-6890

Route 53 > Hosted zones > nendo.fun

Records (5) Info

Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.

□	Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to	TTL (s...	Healt
□	nendo.fun	NS	Simple	-	No	ns-856.awsdns-43.net. ns-76.awsdns-09.com. ns-1330.awsdns-38.org. ns-1707.awsdns-21.co.uk.	60	-
□	nendo.fun	SOA	Simple	-	No	ns-856.awsdns-43.net. awsd...	60	-
□	_8d803e5...	CNAME	Simple	-	No	_4d877d17908c6b8b051ff8...	60	-
□	_7c21e07...	CNAME	Simple	-	No	_2361accfa8b17b00f898a54...	60	-
□	_17c6c2a...	CNAME	Simple	-	No	_1aafbaf6ed1fe915e2c1950...	60	-

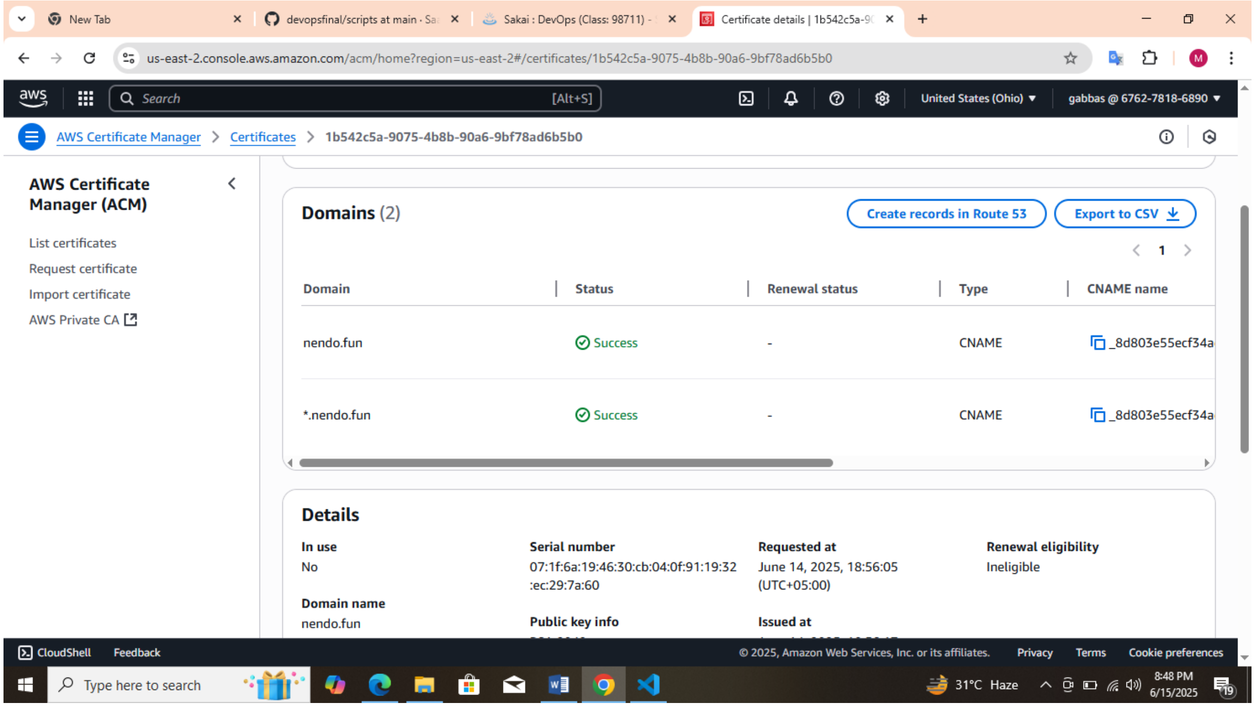


New Tab
devopsfinal/scripts at main - Sa
Sakai : DevOps (Class: 98711) -
Certificates | Certificate Manage
us-east-2.console.aws.amazon.com/acm/home?region=us-east-2#/certificates/list
Search [Alt+S]
United States (Ohio)
gabbas @ 6762-7818-6890

AWS Certificate Manager > Certificates

Certificates (1)

□	Certificate ID	Domain name	Type	Status
□	1b542c5a-9075-4b8b-90a6-9bf78ad6b5b0	nendo.fun	Amazon Issued	✓ Issued



New Tab
devopsfinal/scripts at main · Sa
Sakai : DevOps (Class: 98711) ·
Certificate details | 1b542c5a-9
us-east-2.console.aws.amazon.com/acm/home?region=us-east-2#/certificates/1b542c5a-9075-4b8b-90a6-9bf78ad6b5b0
Search [Alt+S]
United States (Ohio)
gabbas @ 6762-7818-6890
AWS Certificate Manager > Certificates > 1b542c5a-9075-4b8b-90a6-9bf78ad6b5b0
AWS Certificate Manager (ACM)
List certificates
Request certificate
Import certificate
AWS Private CA
Create records in Route 53
Export to CSV
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Type here to search
31°C Haze
8:48 PM
6/15/2025

Domains (2)

Domain	Status	Renewal status	Type	CNAME name
nendo.fun	✓ Success	-	CNAME	_8d803e55ecf34a
*.nendo.fun	✓ Success	-	CNAME	_8d803e55ecf34a

Details

In use
No

Serial number
07:1f:6a:19:46:30:cb:04:0f:91:19:32:ec:29:7a:60

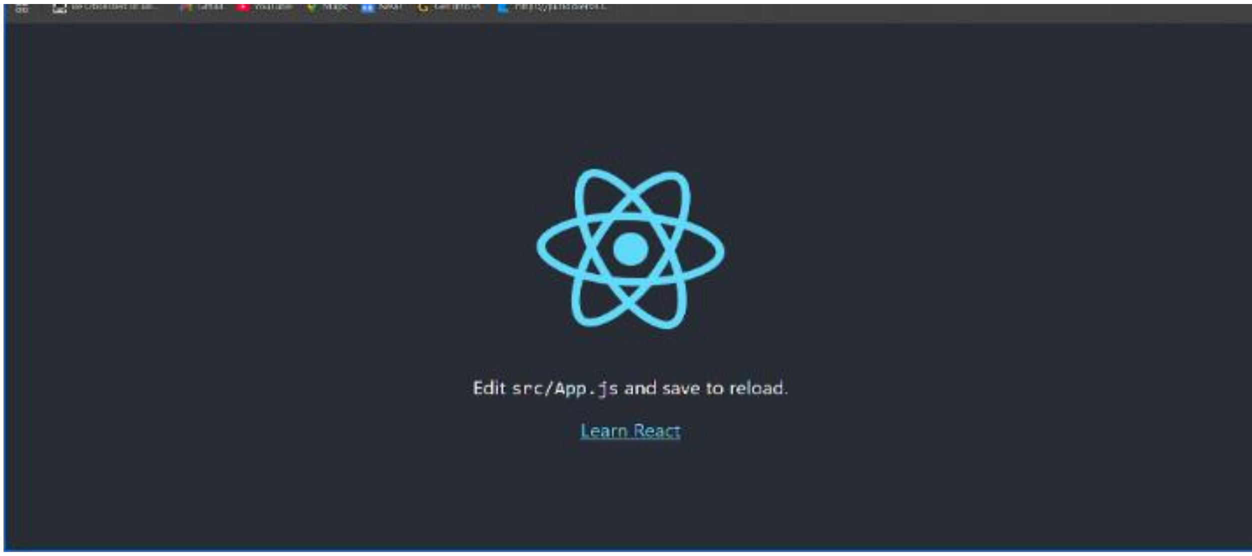
Requested at
June 14, 2025, 18:56:05 (UTC+05:00)

Renewal eligibility
Ineligible

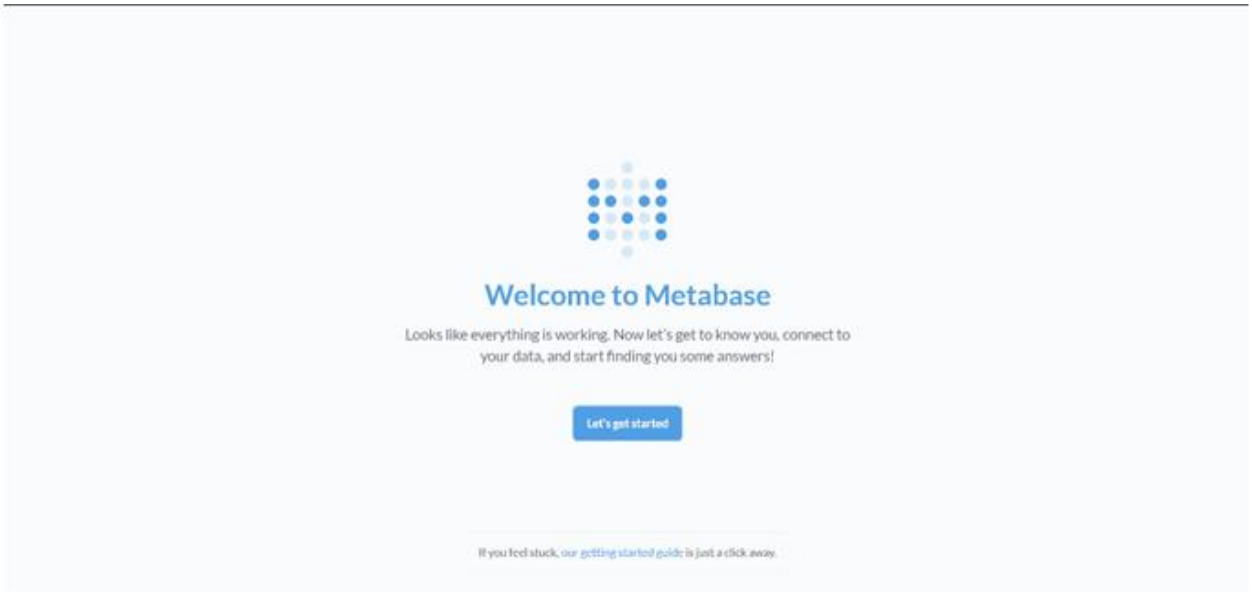
Domain name
nendo.fun

Public key info

Issued at



Edit src/App.js and save to reload.

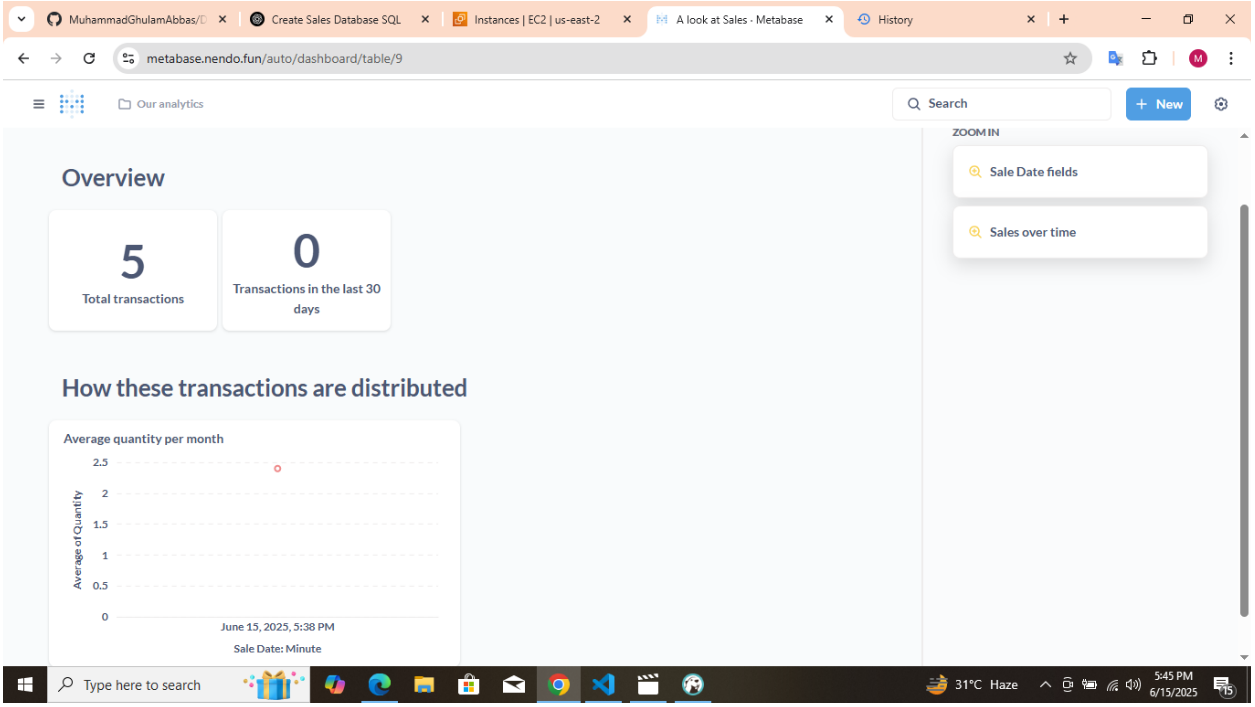


Welcome to Metabase

Looks like everything is working. Now let's get to know you, connect to your data, and start finding you some answers!

Let's get started

If you feel stuck, our getting started guide is just a click away.



MuhammadGhulamAbbas/D
Create Sales Database SQL
Instances | EC2 | us-east-2
A look at Sales · Metabase
History

metabase.nendo.fun/auto/dashboard/table/9

Overview

5 Total transactions

0 Transactions in the last 30 days

How these transactions are distributed

Average quantity per month