

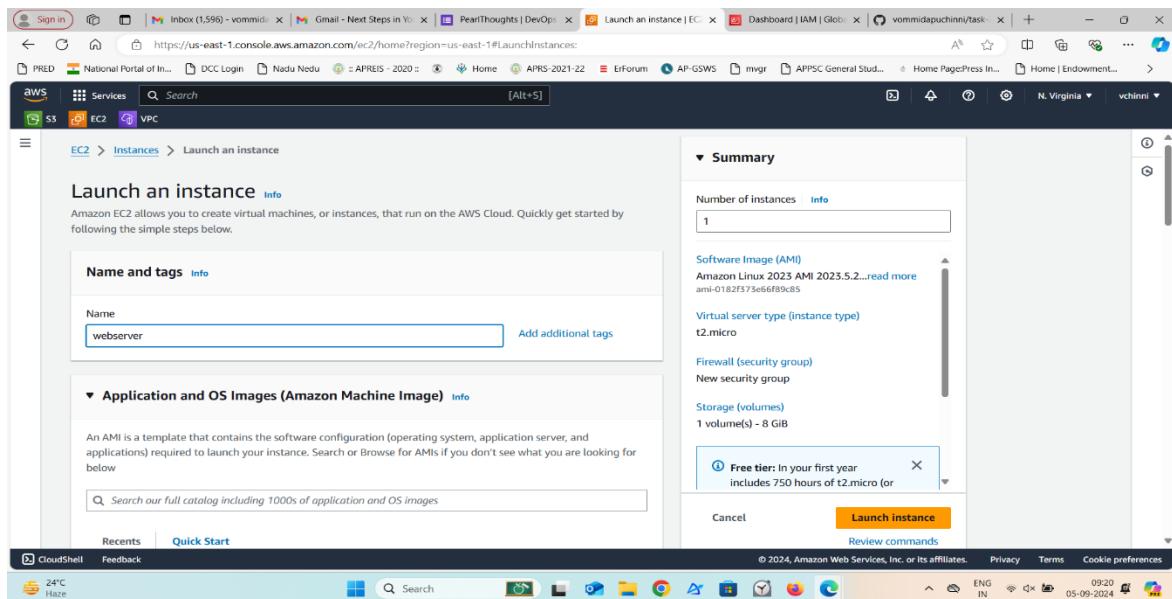
# Medusa Backend Deployment on AWS Using Terraform, ECS, and GitHub Actions

Git repo: <https://github.com/vommidapuchinni/pearl-task.git>

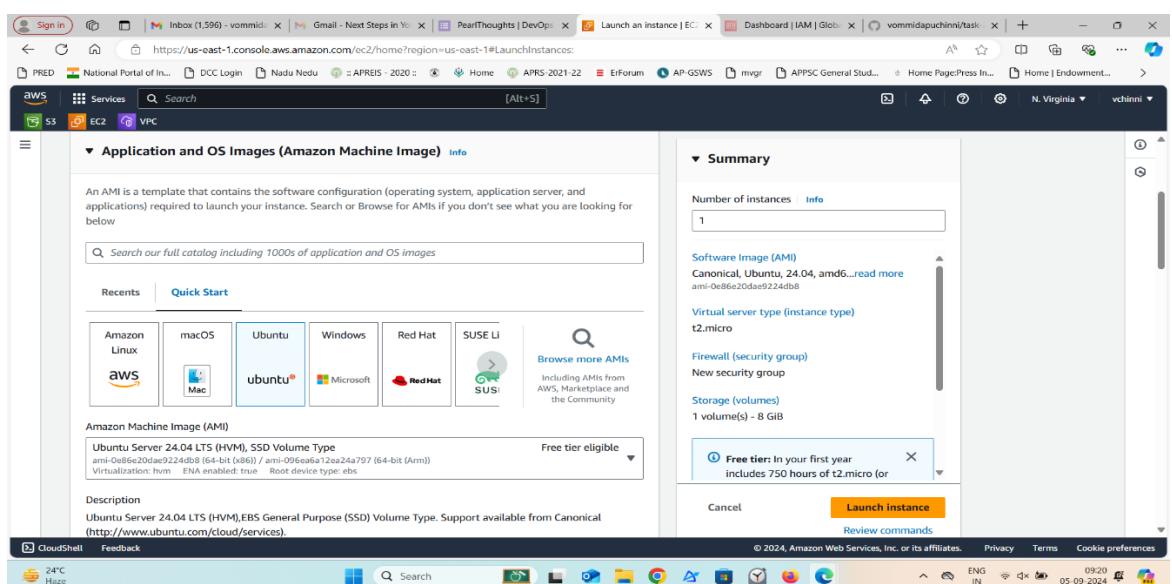
## Step 1: Launch EC2 Instances

Launch the Instance:

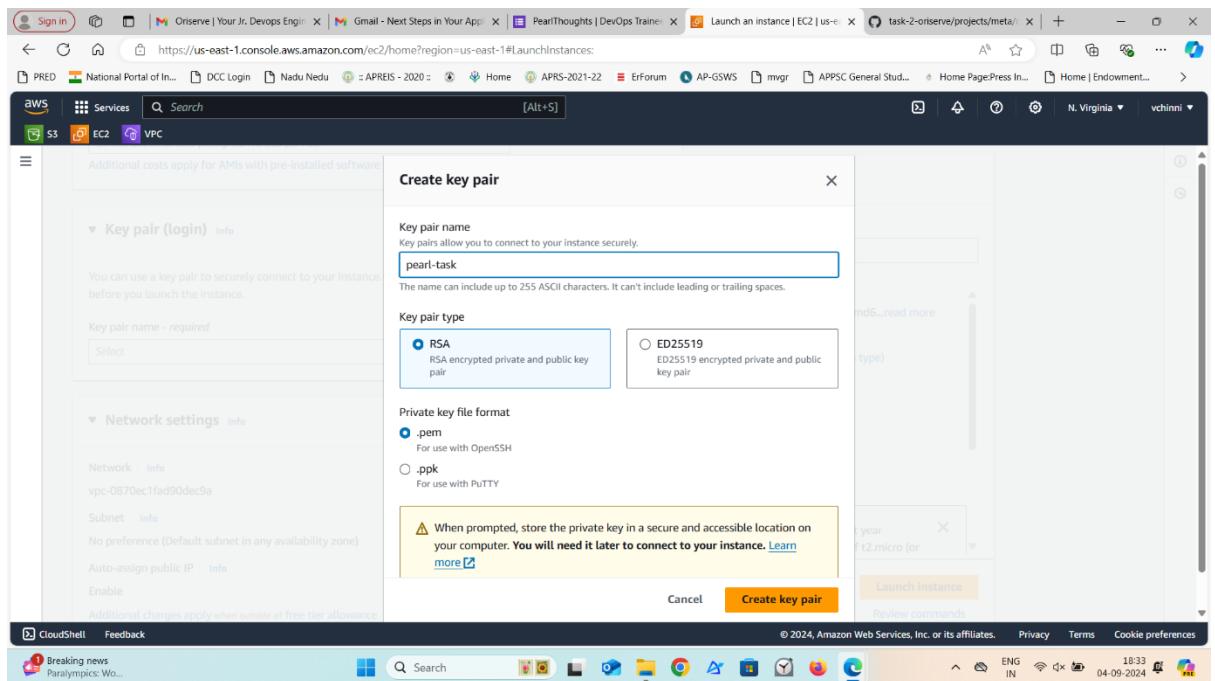
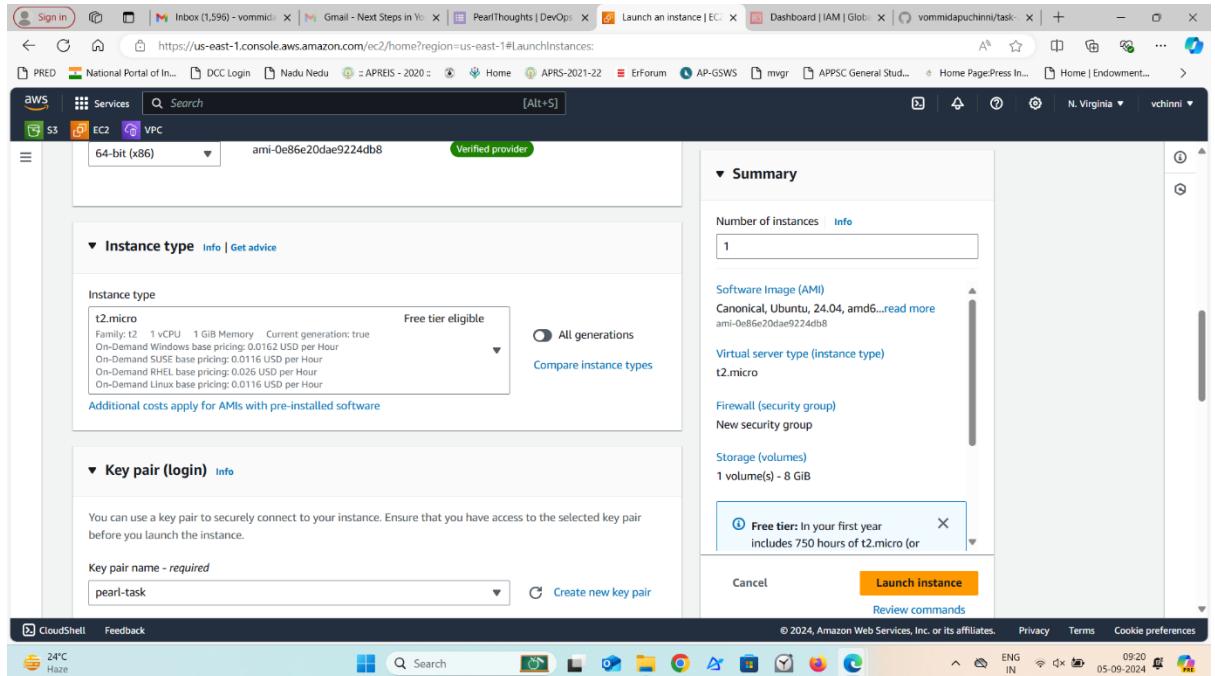
- Go to the AWS EC2 Console.
- Click Launch Instance
- Name and Tags: Name the instance webserver.



- AMI: Choose Ubuntu Server 20.04 LTS (or any preferred version).

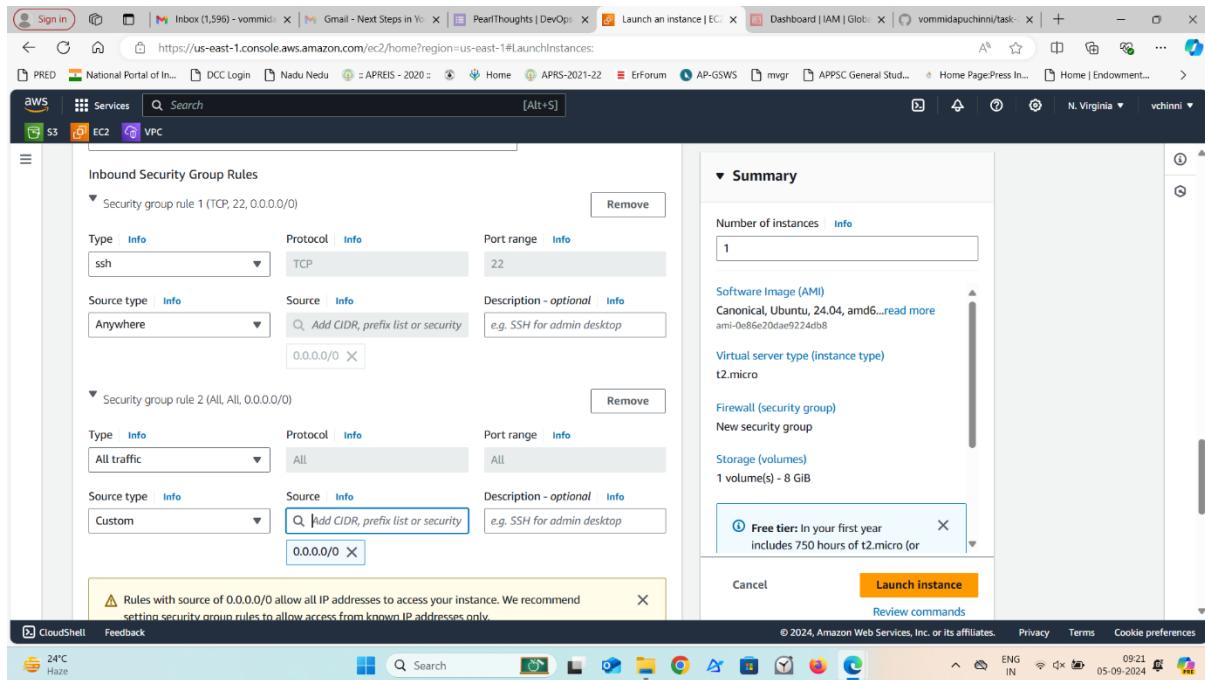


- Instance Type: Choose t2.micro for testing (eligible for free tier).
- Key Pair: create a new one.



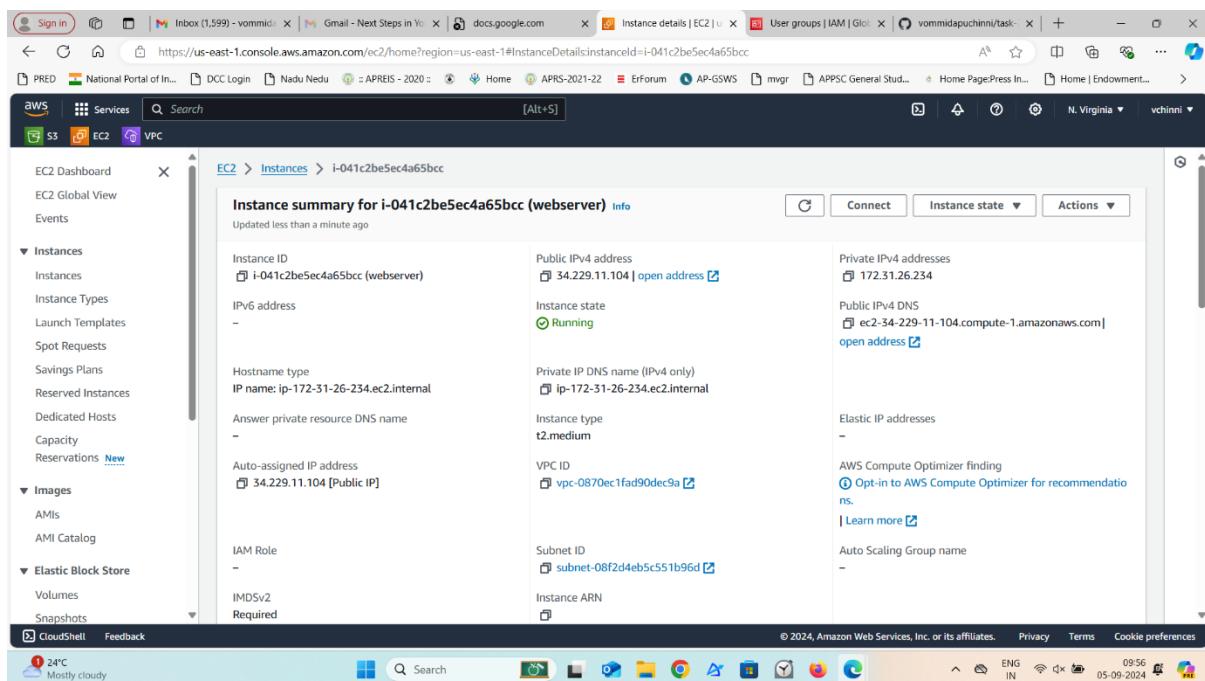
- Network Settings:

- Create a new security group allowing:
  - SSH (port 22) from your IP for secure access.
  - I gave all traffic.



- Click Launch Instance.

- Connect to the Instance



## Step 2: Set Up PostgreSQL

1. Install PostgreSQL: Ensure PostgreSQL is installed on your server. If not, install it using:
  - sudo apt update

```

Sign in  Inbox (1,599) - vommidapuchinni  Gmail - Next Steps  PearlThoughts | Dev  Instance details | EC2 Instance Connect  User groups | IAM  vommidapuchinni  N. Virginia  vchinni
https://us-east-1.console.aws.amazon.com/ec2-instance-connect/shell?region=us-east-1&connType=standard&instanceId=i-041c2be5ec4a65bcc&osUser=ubuntu&s...  A  Star  Print  Copy  Close  Help  More  ...
PREDE National Portal of In... DCC Login Nadu Nedu APREIS - 2020 Home APRS-2021-22 ErForum AP-GSWS mvgr APPSC General Stud... Home Page|Press In... Home | Endowment...
aws Services Search [Alt+S]
S3 EC2 VPC
Memory usage: 5% IPv4 address for enX0: 172.31.26.234
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

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-26-234:~$ sudo -i
root@ip-172-31-26-234:~# apt update -y
i-041c2be5ec4a65bcc (webserver)
PublicIP: 34.229.11.104 PrivateIP: 172.31.26.234

```

- sudo apt install postgresql postgresql-contrib

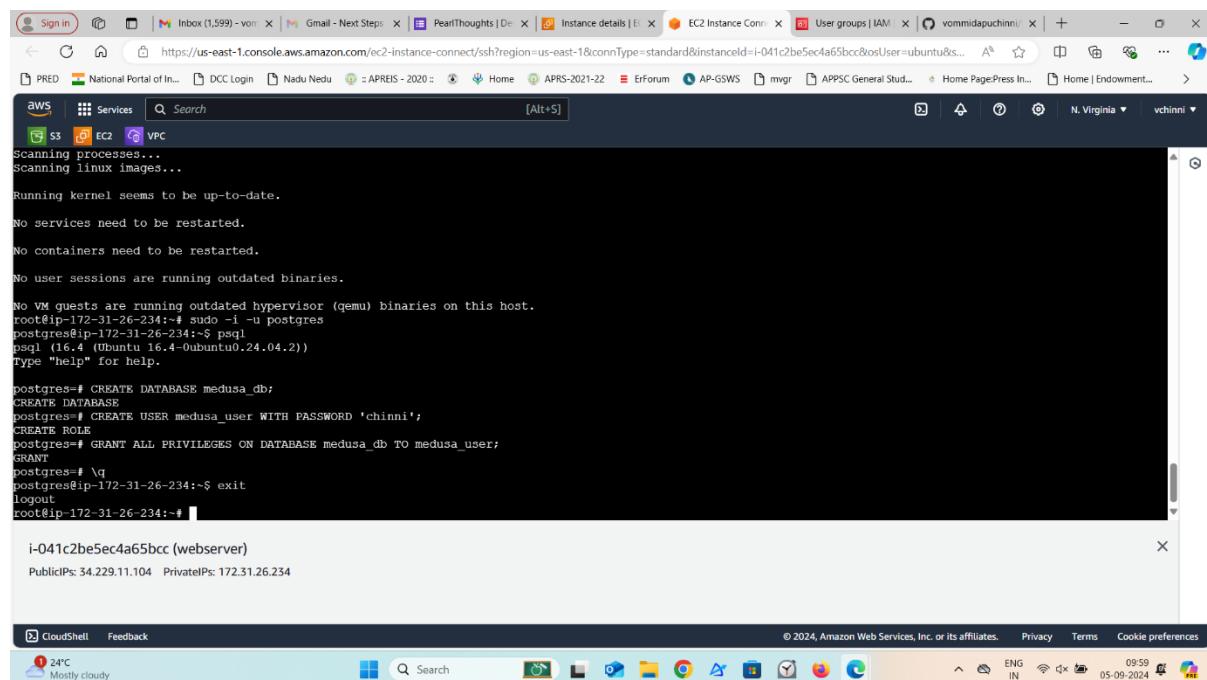
```

Sign in  Inbox (1,599) - vommidapuchinni  Gmail - Next Steps  PearlThoughts | Dev  Instance details | EC2 Instance Connect  User groups | IAM  vommidapuchinni  N. Virginia  vchinni
https://us-east-1.console.aws.amazon.com/ec2-instance-connect/shell?region=us-east-1&connType=standard&instanceId=i-041c2be5ec4a65bcc&osUser=ubuntu&s...  A  Star  Print  Copy  Close  Help  More  ...
PREDE National Portal of In... DCC Login Nadu Nedu APREIS - 2020 Home APRS-2021-22 ErForum AP-GSWS mvgr APPSC General Stud... Home Page|Press In... Home | Endowment...
aws Services Search [Alt+S]
S3 EC2 VPC
root@ip-172-31-26-234:~# sudo apt install postgresql postgresql-contrib
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libl10n-i18n-t64 libpq5 libtypes-serialiser-perl postgresql-16 postgresql-client-16 postgresql-client-common
  postgresql-common ssl-cert
Suggested packages:
  postgresql-doc postgresql-doc-16
The following NEW packages will be installed:
  libcommon-sense-perl libjson-perl libjson-xs-perl libl10n-i18n-t64 libpq5 libtypes-serialiser-perl postgresql postgresql-16 postgresql-client-16
  postgresql-client-common postgresql-common postgresql-contrib ssl-cert
0 upgraded, 13 newly installed, 0 to remove and 111 not upgraded.
Need to get 43.5 MB of archives.
After this operation, 175 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libjson-perl all 4.10000-1 [81.9 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 postgresql-client-common all 257build1.1 [36.4 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 ssl-cert all 1.1.2ubuntu1 [17.8 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 postgresql-common all 257build1.1 [161 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libcommon-sense-perl amd64 3.75-3build3 [20.4 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libtypes-serialiser-perl all 1.01-1 [11.6 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libjson-xs-perl amd64 4.030-2build3 [83.6 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libl10n-i18n-t64 amd64 1:17.0.6-9ubuntu1 [26.2 MB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libpq5 amd64 16.4-0ubuntu0.24.04.2 [141 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 postgresql-client-16 amd64 16.4-0ubuntu0.24.04.2 [1271 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 postgresql-16 amd64 16.4-0ubuntu0.24.04.2 [15.5 MB]
i-041c2be5ec4a65bcc (webserver)
PublicIP: 34.229.11.104 PrivateIP: 172.31.26.234

```

2. Start and Enable PostgreSQL: Start and enable PostgreSQL to run on boot:
  - sudo systemctl start postgresql
  - sudo systemctl enable postgresql
3. Switch to the PostgreSQL User: Switch to the postgres user:
  - sudo -i -u postgres
4. Access PostgreSQL CLI: Launch the psql command-line interface:

- psql
5. Create a Database and User: Execute the following commands in the psql prompt:
- CREATE DATABASE medusa\_db;
  - CREATE USER medusa\_user WITH ENCRYPTED PASSWORD 'your\_password';
  - GRANT ALL PRIVILEGES ON DATABASE medusa\_db TO medusa\_user;
  - \q
  - Replace 'your\_password' with a strong password of your choice.
6. Exit PostgreSQL User: Exit from the postgres user:
- Exit



```

Sign in | Gmail - Next Steps | PearThoughts | De... | Instance details | EC2 Instance Conn... | User groups | IAM | vommidaapuchinni/ | + | - | X
https://us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-041c2be5ec4a65bcc&osUser=ubuntu&s...
PRED National Portal of In... DCC Login Nadu Nedu : APREIS - 2020 : Home APRS-2021-22 ErForum AP-GSW5 mygr APPSC General Stud... Home Page|Press In... Home | Endowment... >
aws Services Search [Alt+S]
S3 EC2 VPC
Scanning processes...
Scanning linux images...
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-26-234:~# sudo -i -u postgres
postgres@ip-172-31-26-234:~$ psql
psql (16.4 (Ubuntu 16.4-0ubuntu0.24.04.2))
Type "help" for help.

postgres=# CREATE DATABASE medusa_db;
CREATE DATABASE
postgres# CREATE USER medusa_user WITH PASSWORD 'chinny';
CREATE ROLE
postgres# GRANT ALL PRIVILEGES ON DATABASE medusa_db TO medusa_user;
GRANT
postgres# \q
postgres@ip-172-31-26-234:~$ exit
logout
root@ip-172-31-26-234:~# 

```

i-041c2be5ec4a65bcc (webserver)  
PublicIPs: 34.229.11.104 PrivateIPs: 172.31.26.234

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences  
24C Mostly cloudy Search ENG IN 09:59 05-09-2024

### Step 3: Set Up Medusa Backend

- Install Node.js and npm: Ensure Node.js and npm are installed. If not, install them using:
  - sudo apt update
  - sudo apt install nodejs npm

- Install Medusa CLI: Install the Medusa CLI globally:
    - `npm install -g @medusajs/medusa-cli`

```
Sign in | Inbox (1,600) - von | Gmail - Next Steps | PearThoughts | Dev | Instance details | EC2 Instance Conn | User groups | IAM | vommidapuchini/x | + | CloudShell | Feedback | © 2024, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences | ENG | 1001 | 05.06.2024 | 24°C | Meteoblue

https://us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-041c2be5ec4a65bcc&osUser=ubuntu&s... A Star | DCC Login | Nadu Nedu | APREIS - 2020 | Home | APRS-2021-22 | ErForum | AP-GSWS | mvgr | APPSC General Stud... | Home Page | Press In... | Home | Endowment... | N. Virginia | vchinni

aws Services Search [Alt+S]
S3 EC2 VPC
ubuntu @ user manager service: systemd[1397]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-26-234:~# npm install -g medusajs/medusa-cli
npm WARN deprecated @oclif/screen@1.0.4: Deprecated in favor of @oclif/core
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm WARN deprecated @oclif/command@1.8.36: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @glob@8.1.0: Glob versions prior to v9 are no longer supported
npm WARN deprecated @oclif/help@1.0.15: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated @oclif/errors@1.3.6: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @oclif/errors@1.3.5: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @oclif/config@1.18.16: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @oclif/parser@3.8.17: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @oclif/config@1.18.2: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated @oclif/config@1.18.17: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.
npm WARN deprecated cli-ux@5.6.7: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.

added 337 packages in 13s

50 packages are looking for funding
  run 'npm fund' for details
root@ip-172-31-26-234:~# npm fund
root

root@ip-172-31-26-234:~# i-041c2be5ec4a65bcc (webserver)
PublicIP: 34.229.11.104 PrivateIP: 172.31.26.234
```

- Create a New Medusa Project: Create a new Medusa project:
    - `medusa new medusa-backend`

```

root@ip-172-31-26-234:~# medusa new medusa-backend
Medusa collects anonymous usage analytics
to help improve Medusa for all users.

If you'd like to opt-out, you can use `medusa telemetry --disable`


✓ created starter directory layout - 296ms
info:  Installing packages...

info:  Preferred package manager set to "npm"
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm WARN deprecated @oclif/screen@1.0.4: deprecated in favor of @oclif/core
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated glob@8.1.0: Glob versions prior to v9 are no longer supported
npm WARN deprecated glob@8.1.0: Glob versions prior to v9 are no longer supported
npm WARN deprecated glob@8.1.0: Glob versions prior to v9 are no longer supported
npm WARN deprecated @babel/plugin-proposal-class-properties@7.10.6: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-class-properties instead.
npm WARN deprecated abab@2.0.6: Use your platform's native atob() and btoa() methods instead
npm WARN deprecated @babel/plugin-proposal-optional-chaining@7.21.0: This proposal has been merged to the ECMAScript standard and thus this plugin is no longer maintained. Please use @babel/plugin-transform-optional-chaining instead.
npm WARN deprecated @oclif/command@1.0.36: Package no longer supported. Contact Support at https://www.npmjs.com/support for more info.

i-041c2be5ec4a65bcc (webserver)
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234

```

- cd medusa-backend
- Install Project Dependencies: Install the required dependencies for the project:
  - npm install

```

root@ip-172-31-26-234:~# cd medusa-backend
root@ip-172-31-26-234:~/medusa-backend# npm install
up to date, audited 2021 packages in 5s
330 packages are looking for funding
  run 'npm fund' for details
17 vulnerabilities (8 moderate, 6 high, 3 critical)

To address issues that do not require attention, run:
  npm audit fix

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

Run 'npm audit' for details.
root@ip-172-31-26-234:~/medusa-backend# ls -a
.babelrc.js  .github  .vscode  README.md  index.js  node_modules  package.json  tsconfig.admin.json  tsconfig.server.json
..  .env        .gitignore  .yarnrc.yml  data       medusa-config.js  package-lock.json  src          tsconfig.json    tsconfig.spec.json
root@ip-172-31-26-234:~/medusa-backend# vi .env
root@ip-172-31-26-234:~/medusa-backend# sudo apt install redis-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libjemalloc2  liblzf1  redis-tools

i-041c2be5ec4a65bcc (webserver)
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234

```

## 5. Configure Medusa

Update .env File: Create or update the .env file in your project root with PostgreSQL credentials:

- DATABASE\_TYPE=postgres
  - DATABASE\_URL=postgres://medusa\_user:your\_password@localhost:5432/medusa\_db
  - JWT\_SECRET=your\_jwt\_secret
  - COOKIE\_SECRET=your\_cookie\_secret
  - NODE\_ENV=production
  - NPM\_CONFIG\_PRODUCTION=false
- **Update medusa-config.js:** Ensure your medusa-config.js is configured to use the PostgreSQL database

**Step 4:** setup all thing

### **Basic .env File Content**

# Database Configuration

```
DATABASE_URL=postgres://medusa_user:your_password@localhost:5432/medusa_db
```

# Redis Configuration (if using Redis)

```
REDIS_URL=redis://localhost:6379
```

# Other Medusa Configuration (if needed)

# Example: PORT=9000

# PORT=9000

### **Ensure Redis is installed and running:**

```
sudo apt update && sudo apt install redis-server && sudo systemctl start redis-server
```

### **Connect to PostgreSQL as a Superuser**

First, connect to your PostgreSQL database as a superuser (like postgres):

```
sudo -u postgres psql
```

### **2. Grant Necessary Permissions**

Once you're in the PostgreSQL shell, run the following commands to grant the medusa\_user the required permissions on the public schema:

```
GRANT ALL PRIVILEGES ON SCHEMA public TO medusa_user;
```

```
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL ON TABLES TO medusa_user;
```

### 3. Verify Permissions

You can verify that the user now has the correct permissions by running:

```
\dn+
```

This will list the schemas and the privileges granted to each role.

### 4. Exit the PostgreSQL Shell

Type `\q` to exit the PostgreSQL shell.

**Ensure that the redis\_url property** in the projectConfig is properly set. And update package.json and medusa-config.js

**Run Migrations:** If you haven't done so already, run the migrations to ensure the database schema is up to date:

```
medusa migrations run
```

### Start the Medusa Backend

Ensure the Medusa backend is running. Depending on how you are managing services, you might start it with a command like:

```
medusa develop
```

```
aws S3 Services Search [Alt+S] https://us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-041c2be5ec4a65bcc&osUser=ubuntu&s... Aa ... Sign in | DCC Login | Nadu Nedu | APREIS - 2020 | Home | APRS-2021-22 | EForum | AP-GSW | mvgr | APPSC General Stud... | Home Page | Press In... | Home | Endowment... | N. Virginia vchinni

Query: INSERT INTO "public"."migrations"("timestamp", "name") VALUES ($1, $2) -- PARAMETERS: [1709888477798, "AlterCustomerUniqueConstraint1709888477798"]
Migration AlterCustomerUniqueConstraint1709888477798 has been executed successfully.
Query: COMMIT
Info: Connection to Redis in module 'event-bus-redis' established
Info: Connection to Redis in module 'cache-redis' established
root@ip-172-31-26-234:~/medusa-backend# medusa develop
Successfully compiled 0 files with Babel (6ms).
10:05:15 AM [medusa@/admin] Started development server on http://localhost:7001/
<1> [Webpack-dev-server] Project is running at:
<1> [Webpack-dev-server] Loopback: http://localhost:7001/, http://127.0.0.1:7001/
<1> [Webpack-dev-server] Content not from webpack is served from '/root/medusa-backend/public' directory
<1> [Webpack-dev-server] 404s will fallback to '/'
Info: Connection to Redis established
✓ Models initialized - 34ms
✓ Plugin models initialized - 136ms
✓ Strategies initialized - 43ms
✓ Database initialized - 247ms
✓ Repositories initialized - 140ms
✓ Services initialized - 61ms
` Initializing modules
Info: Connection to Redis in module 'event-bus-redis' established
✓ Modules initialized - 266ms
✓ Express initialized - 4ms
` Initializing plugins
✓ Plugins initialized - 793ms
✓ Subscribers initialized - 43ms
✓ API initialized - 250ms

i-041c2be5ec4a65bcc (webserver)
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
24C Mostly cloudy ENG IN 10:35 05-09-2024
```

**Rebuild the Project:** Make sure to rebuild the project to include any changes made to the package.json scripts and medusa-config.js file.

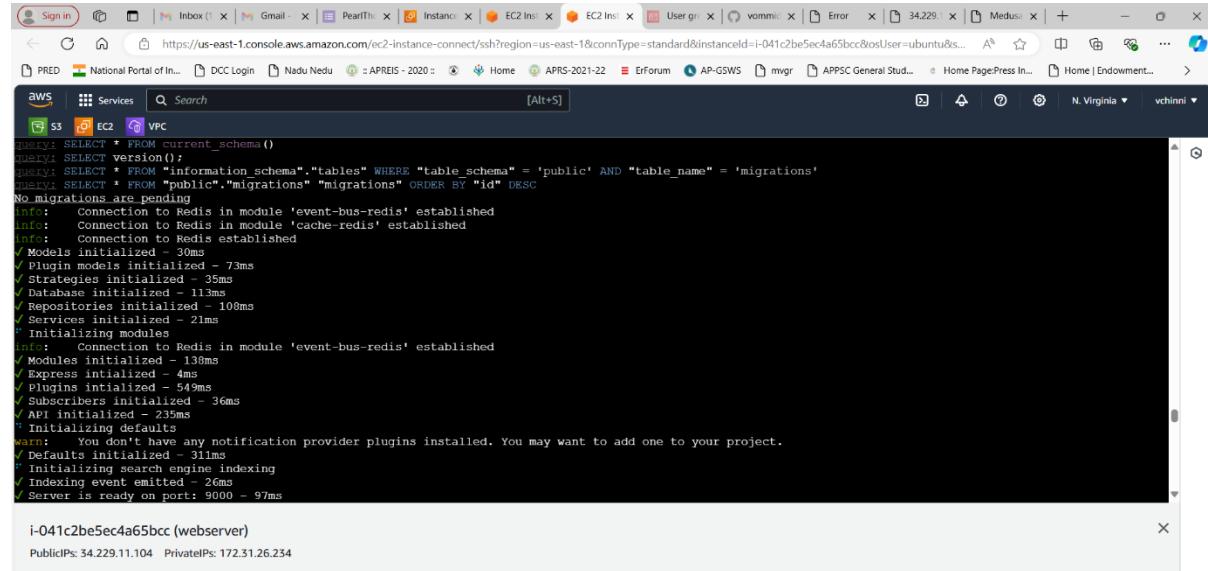
```
npm run build
```

**Run Migrations:** Apply any pending database migrations. This ensures that your database schema is up to date.

```
npx medusa migrations run
```

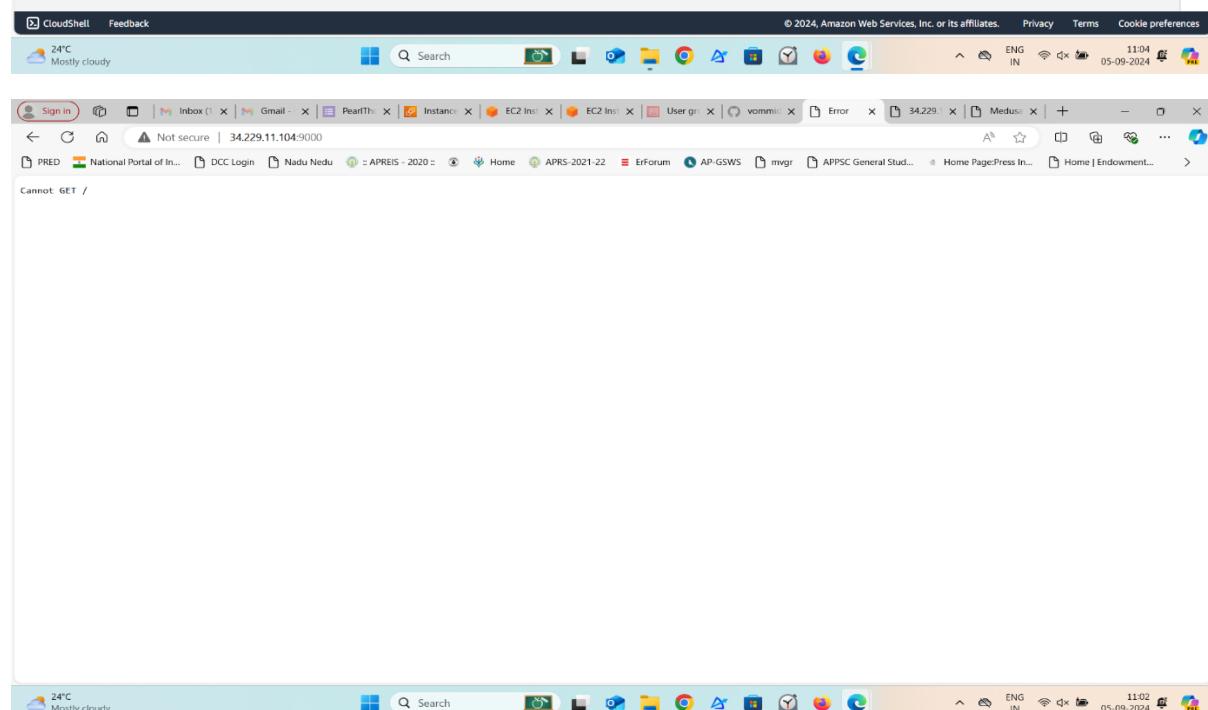
**Start the Backend:** Start your Medusa backend. This will use the updated configuration and ensure that the server is running with the latest changes.

```
npm start
```



```
aws services search [Alt+S]
query: SELECT * FROM current_schema()
query: SELECT version();
query: SELECT * FROM "information_schema"."tables" WHERE "table_schema" = 'public' AND "table_name" = 'migrations'
query: SELECT * FROM "public"."migrations" ORDER BY "id" DESC
No migrations are pending
info: Connection to Redis in module 'event-bus-redis' established
info: Connection to Redis in module 'cache-redis' established
info: Connection to Redis established
✓ Models initialized - 30ms
✓ Plugin models initialized - 73ms
✓ Strategies initialized - 35ms
✓ Database initialized - 113ms
✓ Repositories initialized - 108ms
✓ Services initialized - 21ms
+ Initializing modules
    ✓ Connection to Redis in module 'event-bus-redis' established
    ✓ Modules initialized - 138ms
    ✓ Express initialized - 4ms
    ✓ Plugins initialized - 549ms
    ✓ Subscribers initialized - 36ms
    ✓ API initialized - 235ms
    + Initializing defaults
warn: You don't have any notification provider plugins installed. You may want to add one to your project.
✓ Defaults initialized - 31ms
+ Initializing search engine indexing
✓ Indexing event emitted - 26ms
✓ Server is ready on port: 9000 - 97ms

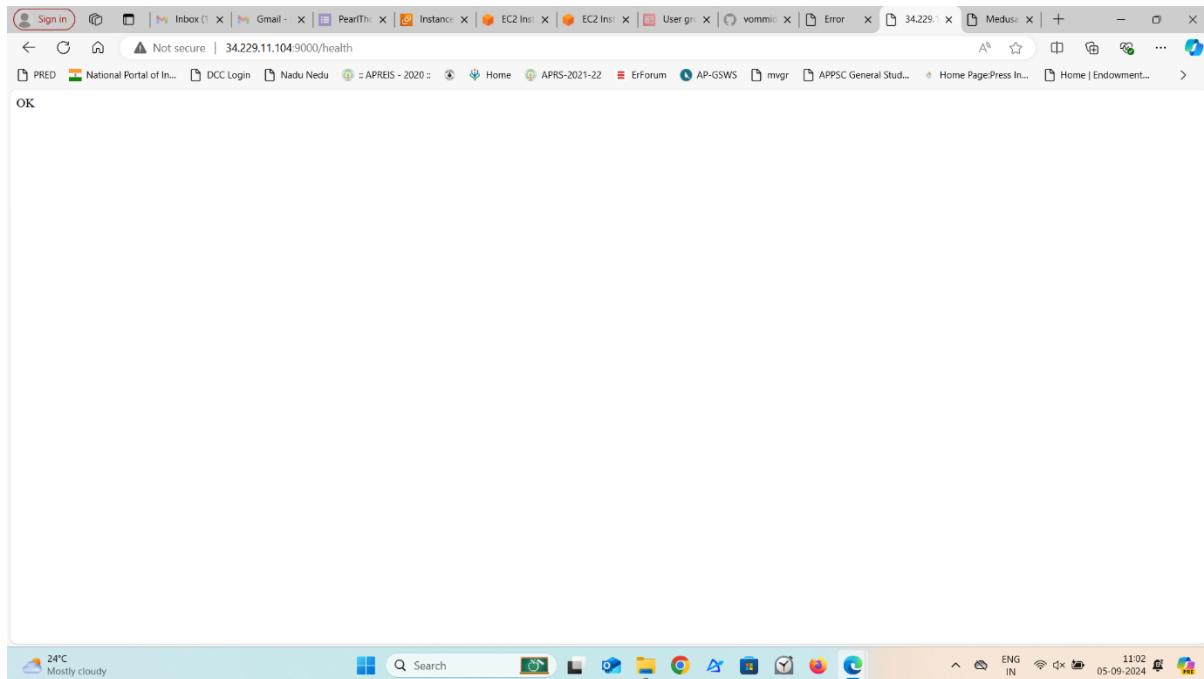
i-041c2be5ec4a65bcc (webserver)
PublicIPs: 34.229.11.104 PrivateIPs: 172.31.26.234
```



**Verify the Deployment:** Check if the backend is running properly by accessing the health check endpoint.

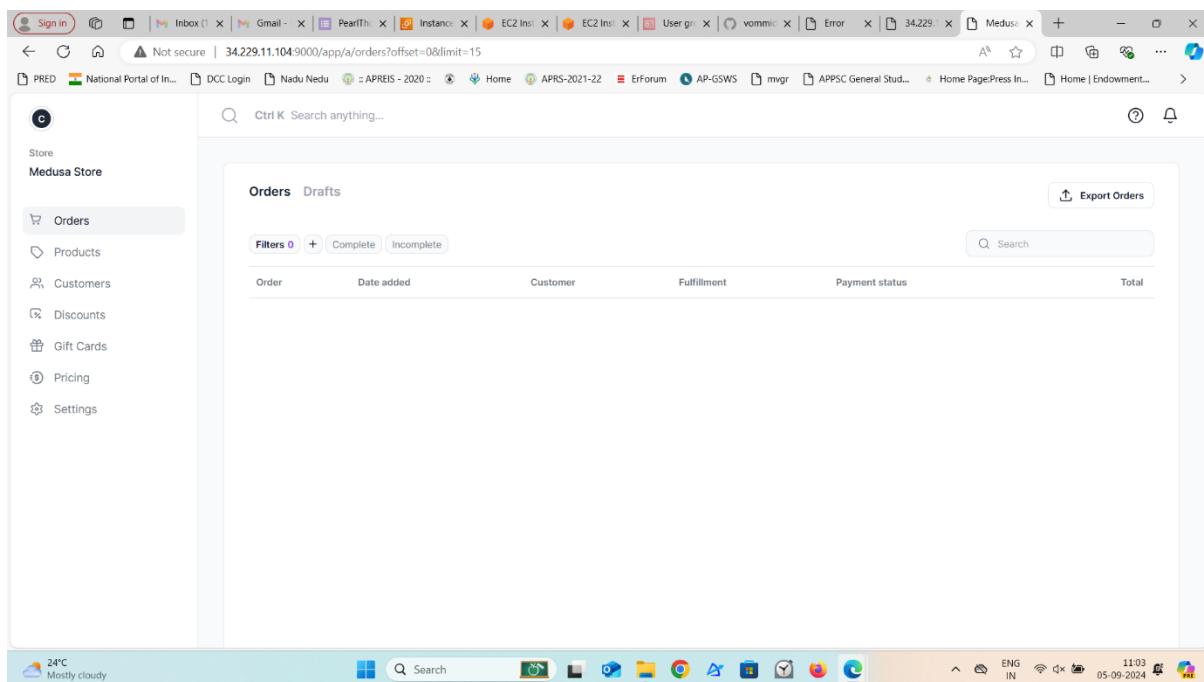
```
curl http://localhost:9000/health
```

Replace localhost:9000 with the appropriate URL if your backend is running on a different host or port.



**Check the Admin Dashboard (if deployed):** If you deployed the admin dashboard with the backend, check if it's available at:

<http://localhost:9000/app>



**Create Admin User (if needed):** If you need to create an admin user, you can do so by running:

```
npx medusa user --email admin@medusa-test.com --password supersecret
```

```

*** System restart required ***
Last login: Thu Sep 5 04:26:56 2024 from 18.206.107.29
ubuntu@ip-172-31-26-234:~$ sudo -i
root@ip-172-31-26-234:~# cd medusa-backend/
root@ip-172-31-26-234:~/medusa-backend# npx medusa user --email chinni@medusa-test.com --password chinni
Info: Connection to Redis established
✓ Models initialized - 38ms
✓ Plugin models initialized - 69ms
✓ Strategies initialized - 39ms
✓ Database initialized - 11ms
✓ Repositories initialized - 35ms
✓ Services initialized - 13ms
` Initializing modules
Info: connection to Redis in module 'event-bus-redis' established
✓ Modules initialized - 145ms
✓ Express initialized - 3ms
✓ Plugins initialized - 466ms
✓ Subscribers initialized - 86ms
✓ API initialized - 145ms
` Initializing defaults
warn: You don't have any notification provider plugins installed. You may want to add one to your project.
✓ Defaults initialized - 277ms
` Initializing search engine indexing
✓ Indexing event emitted - 29ms
root@ip-172-31-26-234:~/medusa-backend# 

i-041c2be5ec4a65bcc (webserver)
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234

```

## Create user and from that create access keys

**Access key best practices & alternatives**

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

**Use case**

- Command Line Interface (CLI)**  
You plan to use this access key to enable the AWS CLI to access your AWS account.
- Local code**  
You plan to use this access key to enable application code in a local development environment to access your AWS account.
- Application running on an AWS compute service**  
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- Third-party service**  
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- Application running outside AWS**  
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

## Step 5: Install needed packages

1. Install Docker: Docker is essential for building and running containerized applications, including the Medusa backend.
  1. Install Docker: `sudo apt update && sudo apt install -y docker.io`
  2. Start Docker and Enable on Boot: `sudo systemctl start docker && sudo systemctl enable docker`

3. Add Your User to Docker Group (Optional): sudo usermod -aG docker \$USER
  4. Verify Docker Installation: docker --version
2. Install Docker Compose

Docker Compose is used to define and run multi-container Docker applications.

1. Download and Install Docker Compose: sudo apt install -y unzip  

```
curl -L "https://github.com/docker/compose/releases/download/$(curl -s https://api.github.com/repos/docker/compose/releases/latest | jq -r .tag_name)/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```
2. Set Permissions: sudo chmod +x /usr/local/bin/docker-compose
3. Verify Docker Compose Installation: docker-compose --version
4. Install AWS CLI: The AWS CLI is already configured with your credentials.
5. Install Terraform:  
Terraform is used for managing infrastructure as code.  
Download and Install Terraform: sudo apt install -y unzip  

```
curl -LO "https://releases.hashicorp.com/terraform/$(curl -s https://checkpoint-api.hashicorp.com/v1/check/terraform | jq -r .current_version)/terraform_$(curl -s https://checkpoint-api.hashicorp.com/v1/check/terraform | jq -r .current_version)_linux_amd64.zip"
```

  
sudo unzip terraform\_\*.zip -d /usr/local/bin/

Verify Terraform Installation: terraform --version

You should create your Dockerfile in the root directory of your project. This is typically where your main application files and configuration are located, and it makes it easier to manage and build your Docker image.

## Step 6: Steps to Create and Use the Dockerfile

1. Create the Dockerfile:
  - o Navigate to the root of your project directory.
  - o Create a file named Dockerfile using a text editor.

## 2. Add the content to the file.

### Step 7: Design the IaC (Terraform, Aws ECS) for Medusa open source

#### 1. provider.tf

This file sets up the AWS provider.

#### 2. networking.tf

This file sets up the VPC, subnet, and security group.

The screenshot shows the AWS VPC dashboard with two VPCs listed:

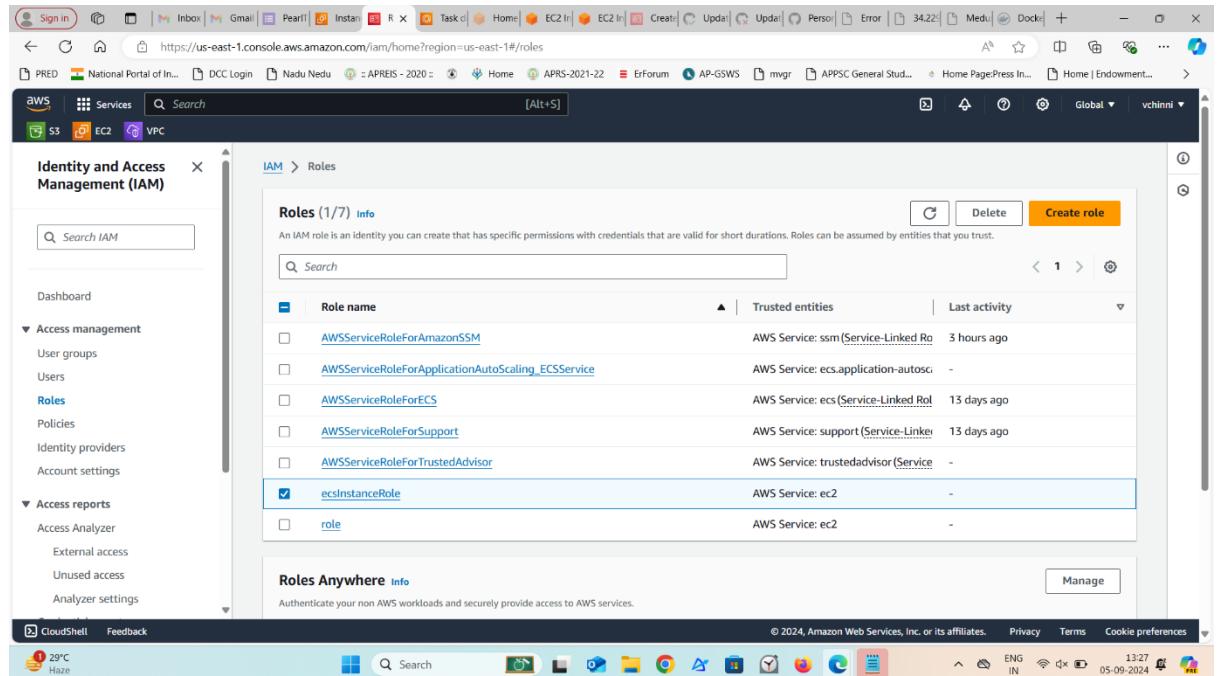
Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP options
main-vpc	vpc-04bcd4ee10df13e2f	Available	10.0.0.0/16	-	dopt-Daf
-	vpc-0870ec1fad90dec9a	Available	172.31.0.0/16	-	dopt-Daf

The screenshot shows the AWS Subnets dashboard with multiple subnets listed under the "public-subnet" VPC:

Name	Subnet ID	State	VPC	IPv4 CIDR
public-subnet	subnet-0d4a69042565117dc	Available	vpc-04bcd4ee10df13e2f   main...	10.0.1.0/24
-	subnet-02267fa7452f80027	Available	vpc-0870ec1fad90dec9a	172.31.64.0/20
-	subnet-0475bcd89ab2559c4	Available	vpc-0870ec1fad90dec9a	172.31.32.0/20
-	subnet-0062220f92f4639e3	Available	vpc-0870ec1fad90dec9a	172.31.0.0/20
-	subnet-08fd2d4eb5c551b96d	Available	vpc-0870ec1fad90dec9a	172.31.16.0/20
-	subnet-0b1de4b5aaef814fea	Available	vpc-0870ec1fad90dec9a	172.31.48.0/20

### 3. iam.tf

This file sets up the IAM roles and instance profile.

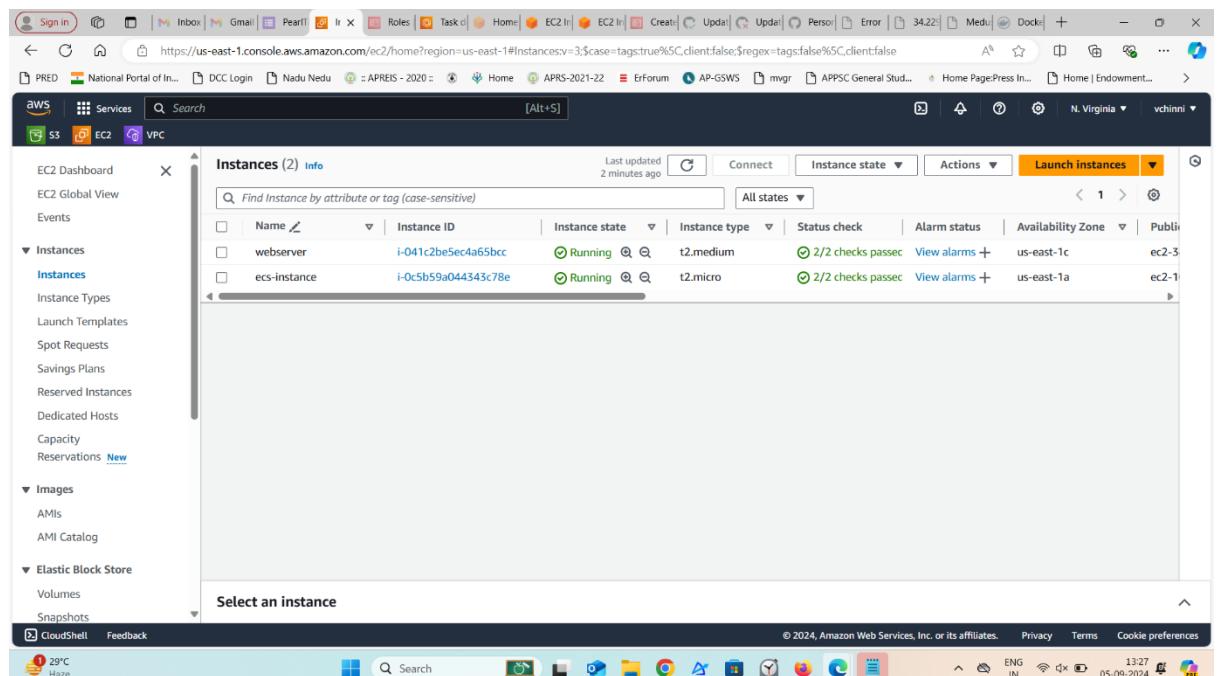


The screenshot shows the AWS IAM Roles page. The left sidebar has 'Identity and Access Management (IAM)' selected. The main area shows a table of roles:

Role name	Trusted entities	Last activity
AWSServiceRoleForAmazonSSM	AWS Service: ssm (Service-Linked Role)	3 hours ago
AWSServiceRoleForApplicationAutoScaling_ECSERVICE	AWS Service: ecs.application-autoscaling	-
AWSServiceRoleForECS	AWS Service: ecs (Service-Linked Role)	13 days ago
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)	13 days ago
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service)	-
<b>ecsInstanceRole</b>	AWS Service: ec2	-
role	AWS Service: ec2	-

### 4. ec2.tf

This file sets up the EC2 instance.

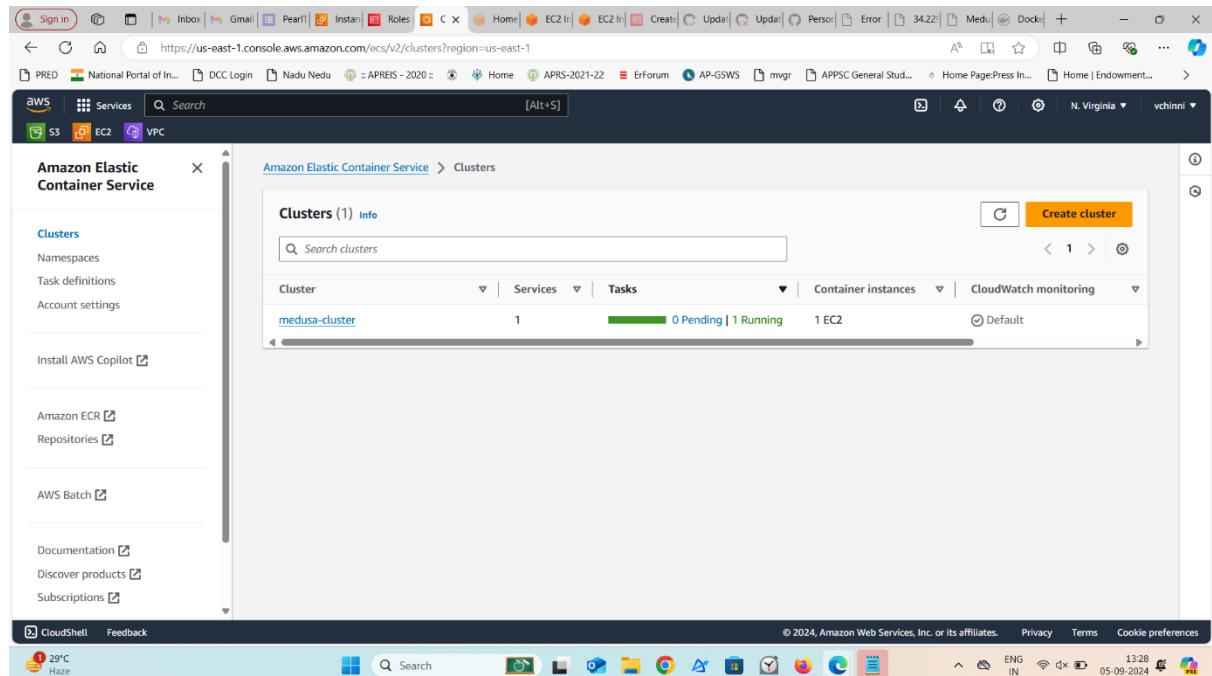


The screenshot shows the AWS EC2 Instances page. The left sidebar has 'Instances' selected. The main area shows a table of instances:

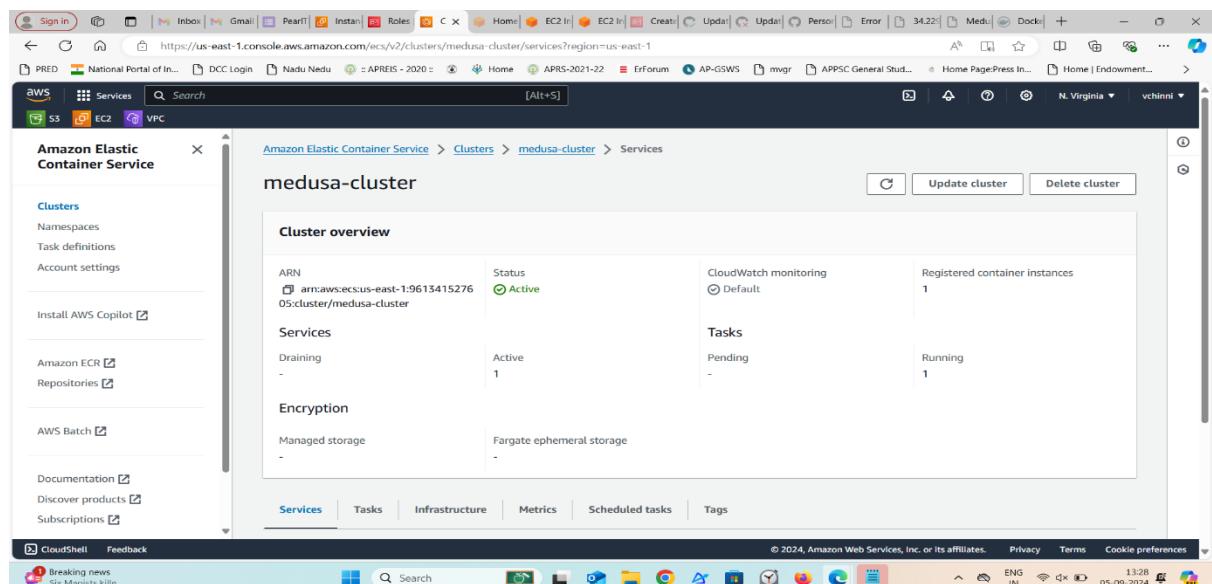
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
webserver	i-041c2be5ec4a65bcc	Running	t2.medium	2/2 checks passed	View alarms +	us-east-1c	ec2-3
ecs-instance	i-0c5b59a044343c78e	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-1

## 5. ecs.tf

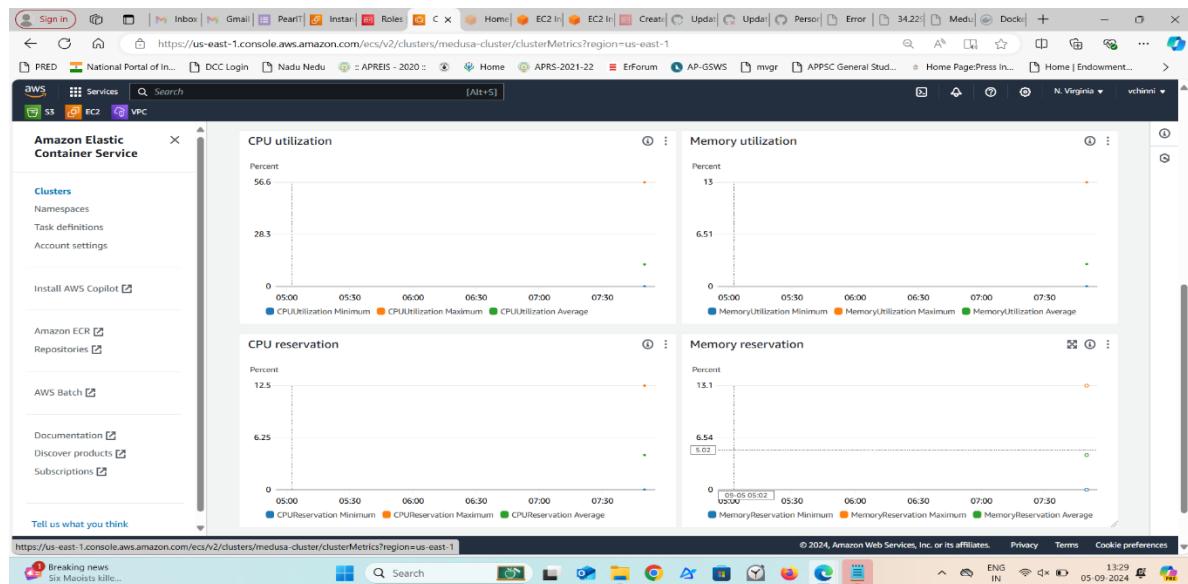
This file sets up the ECS cluster, task definition, and service.



The screenshot shows the AWS Cloud Console interface for the Amazon Elastic Container Service (ECS). The left sidebar is collapsed, and the main area displays the 'Clusters' section. A single cluster, 'medusa-cluster', is listed. The cluster details show 1 EC2 instance and 1 pending task. The status bar at the bottom indicates the region is N. Virginia and the date is 05-09-2024.



The screenshot shows the AWS Cloud Console interface for the Amazon Elastic Container Service (ECS). The left sidebar is collapsed, and the main area displays the 'Services' section for the 'medusa-cluster'. The 'Services' tab is selected. The 'Cluster overview' section shows the ARN (arn:aws:ecs:us-east-1:9613415276:cluster/medusa-cluster), Status (Active), CloudWatch monitoring (Default), and Registered container instances (1). The 'Tasks' section shows 1 Pending task and 1 Running task. The status bar at the bottom indicates the region is N. Virginia and the date is 05-09-2024.



**Task definitions (1) info**

Task definition	Status of last revision
medusa-task	ACTIVE

**medusa-task (1) info**

Filter status
Active

**Task definition: revision**

Status	
medusa-task:3	ACTIVE

The screenshot shows the AWS CloudWatch Metrics interface. At the top, there's a search bar and a navigation bar with links like 'Sign in', 'Inbox', 'Gmail', 'Pearl', 'Insta...', 'Roles', 'Task', 'Create...', 'Update...', 'Person...', 'Error', and '34.22'. Below the search bar, there are tabs for 'Metrics' and 'Logs'. The main area displays a chart titled 'CPU Usage (%)' for the last 1 hour. The chart shows a single data series with a value of approximately 0.68%. The x-axis represents time from 13:32 to 13:33, and the y-axis represents CPU usage from 0% to 100%. The chart includes a legend indicating 'CPU Usage (%)' and a data table with the same information.

## 6. Configure GitHub Actions for CD

Create a GitHub Actions workflow in `.github/workflows/deploy.yml`

The screenshot shows the GitHub Actions workflow status page for the file `mycicd.yaml`. The page has a header with links for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. Below the header, it says 'Deploy to ECS' and shows a green checkmark next to 'Update mycicd.yaml #4'. The main area is titled 'Summary' and shows a single job named 'build-and-deploy'. The job status is 'Succeeded 26 minutes ago in 2m 26s'. The job steps are listed as follows:

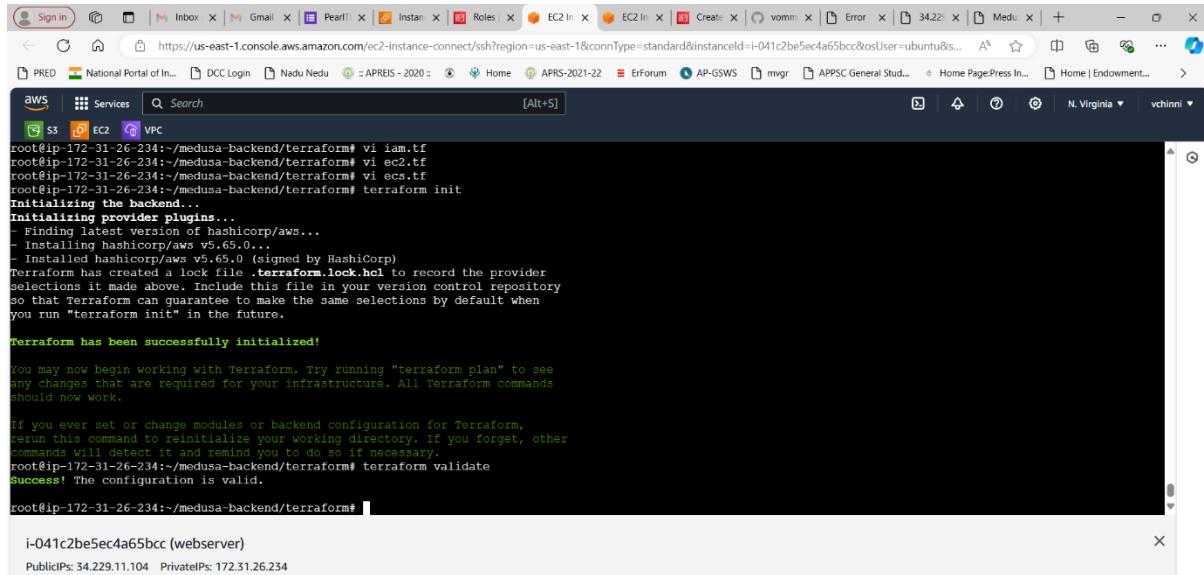
- > Set up job
- > Checkout code
- > Set up Docker Buildx
- > Log in to Docker Hub
- > Build Docker image
- > Push Docker image to Docker Hub
- > Deploy to ECS
- > Post Log in to Docker Hub
- > Post Set up Docker Buildx
- > Post Checkout code

On the right side, there's a 'Search logs' input field and a 'Re-run all jobs' button. At the bottom, there's a link to the workflow YAML file: <https://github.com/vommida-puchinni/pearl-task/actions/runs/1071593540/job/2971236501>.

## Deploy the Configuration

Initialize and apply the Terraform configuration:

- terraform init
- terraform validate
- terraform apply –auto-approve



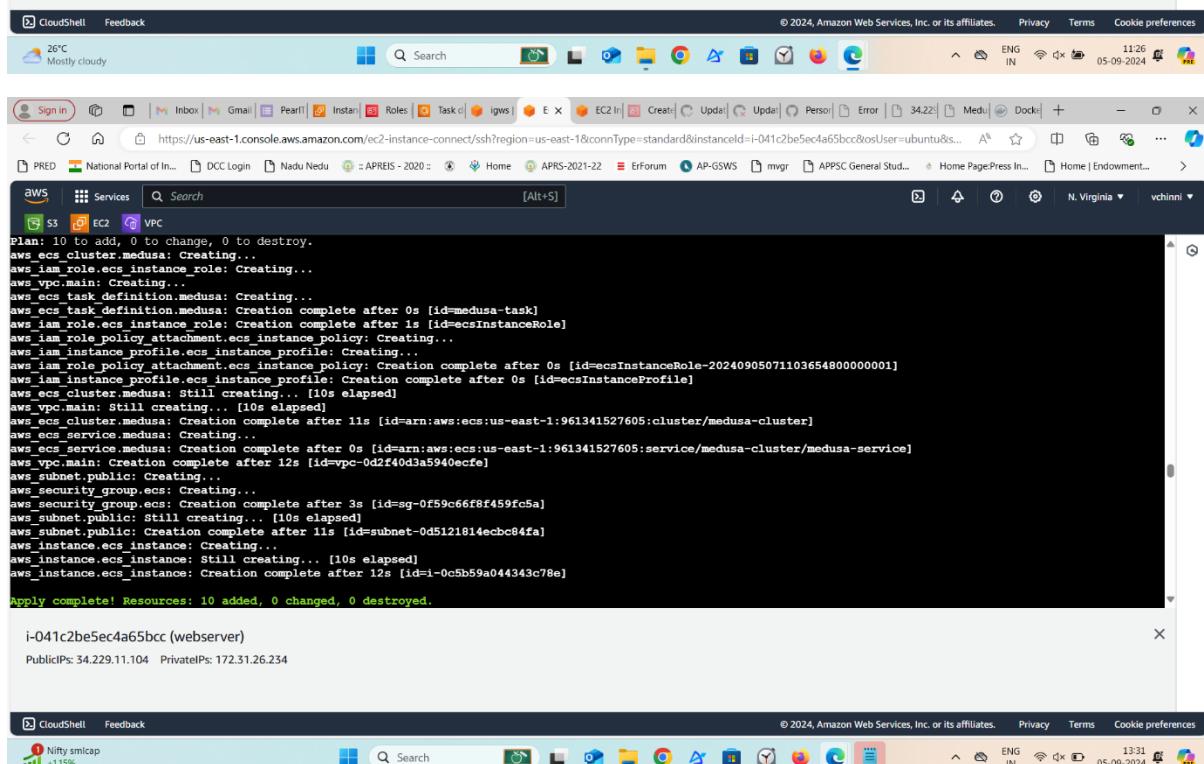
```
root@ip-172-31-26-234:~/medusa-backend/terraform# vi lam.tf
root@ip-172-31-26-234:~/medusa-backend/terraform# vi ec2.tf
root@ip-172-31-26-234:~/medusa-backend/terraform# vi ecs.tf
root@ip-172-31-26-234:~/medusa-backend/terraform# terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.65.0...
- Installed hashicorp/aws v5.65.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

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.
root@ip-172-31-26-234:~/medusa-backend/terraform# terraform validate
Success! The configuration is valid.

root@ip-172-31-26-234:~/medusa-backend/terraform# 
```



```
i-041c2be5ec4a6bcc (webserver)
PublicIPs: 34.229.11.104 PrivateIPs: 172.31.26.234

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
26°C Mostly cloudy Search ENG IN 11:26 05-09-2024

Sign in https://us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-041c2be5ec4a6bcc&osUser=ubuntu&s... A N. Virginia vchinni

AWS Services Search [Alt+S]
S3 EC2 VPC
root@ip-172-31-26-234:~/medusa-backend/terraform# terraform apply
Plan: 10 to add, 0 to change, 0 to destroy.
aws_ecs_cluster.medusa: Creating...
aws_iam_role.ecs_instance_role: Creating...
aws_vpc.main: Creating...
aws_ecs_task_definition.medusa: Creating complete after 0s [id=medusa-task]
aws_iam_role.ecs_instance_role: Creation complete after 0s [id=ecsInstanceRole]
aws_iam_role_policy_attachment.ecs_instance_policy: Creating...
aws_iam_instance_profile.ecs_instance_profile: Creating...
aws_iam_role_policy_attachment.ecs_instance_policy: Creation complete after 0s [id=ecsInstanceRole-20240905071103654800000001]
aws_iam_instance_profile.ecs_instance_profile: Creation complete after 0s [id=ecsInstanceProfile]
aws_ecs_cluster.medusa: Still creating... [10s elapsed]
aws_vpc.main: Still creating... [10s elapsed]
aws_ecs_cluster.medusa: Creation complete after 11s [id=arn:aws:ecs:us-east-1:961341527605:cluster/medusa-cluster]
aws_ecs_service.medusa: Creating...
aws_ecs_service.medusa: Creation complete after 0s [id=arn:aws:ecs:us-east-1:961341527605:service/medusa-cluster/medusa-service]
aws_vpc.main: Creation complete after 12s [id=vpc-0d2f40d3a5940cef]
aws_subnet.public: Creating...
aws_security_group.ecs: Creating...
aws_security_group.ecs: Creation complete after 3s [id=sg-0f59c66f8f459fc5a]
aws_subnet.public: Still creating... [10s elapsed]
aws_subnet.public: Creation complete after 11s [id=subnet-0d5121814ecbc84fa]
aws_instance.ecs_instance: Creating...
aws_instance.ecs_instance: Still creating... [10s elapsed]
aws_instance.ecs_instance: Creation complete after 12s [id=i-0c5b59a044343c78e]

Apply complete! Resources: 10 added, 0 changed, 0 destroyed.

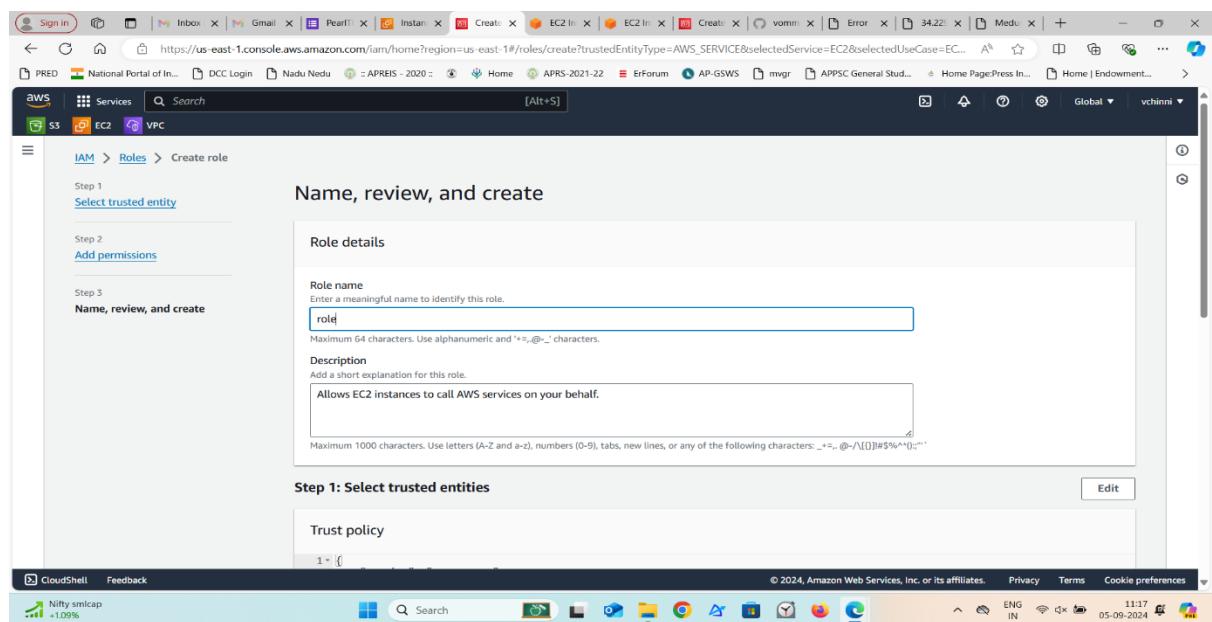
i-041c2be5ec4a6bcc (webserver)
PublicIPs: 34.229.11.104 PrivateIPs: 172.31.26.234

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
Nifty smcap +115% Search ENG IN 13:31 05-09-2024 
```

## Step 8: Install the ECS Agent

To install the ECS agent on your EC2 instance, follow these steps:

1. Update the Package List: sudo apt-get update
2. Install Docker (if not already installed): sudo apt-get install -y docker.io
3. Start Docker Service: sudo systemctl start docker
4. Install the ECS Agent: sudo apt-get install -y ecs-init
5. Start and Enable the ECS Agent Service: sudo systemctl enable ecs && sudo systemctl start ecs
6. Verify ECS Agent Installation. After installing the ECS agent, verify that it's running: sudo systemctl status ecs
7. Verify ECS Configuration on EC2 Instance: Confirm that the ECS agent is correctly configured on your EC2 instance. The configuration file /etc/ecs/ecs.config should have the correct ECS cluster name:  
cat /etc/ecs/ecs.config
8. Ensure it contains: ECS\_CLUSTER=medusa-cluster
9. Confirm EC2 Instance Role and Permissions: Ensure the IAM role attached to the EC2 instance has the required permissions: IAM Role: Should include the **AmazonEC2ContainerServiceforEC2Role** policy.



Instance summary for i-041c2be5ec4a65bcc (webserver) [Info](#)

Updated: 18 minutes ago

Instance ID	<a href="#">i-041c2be5ec4a65bcc (webserver)</a>	Public IPv4 address	<a href="#">34.229.11.104   Open address</a>
IPv6 address	-	Instance state	<a href="#">Running</a>
Hostname type	IP name: ip-172-31-26-234.ec2.internal	Private IP DNS name (IPv4 only)	<a href="#">ip-172-31-26-234.ec2.internal</a>
Answer private resource DNS name	-	Instance type	t2.medium
Auto-assigned IP address	<a href="#">34.229.11.104 [Public IP]</a>	VPC ID	<a href="#">vpc-0870ec1fad90dec9a</a>
IAM Role	-	Subnet ID	<a href="#">subnet-08f2d4eb5c551b96d</a>
IMDSv2 Required	-	Instance ARN	<a href="#">i-041c2be5ec4a65bcc</a>
Elastic IP addresses	-	AWS Compute Optimizer finding	<a href="#">Opt-in to AWS Compute Optimizer for recommendations.</a>
Auto Scaling Group name	-	Learn more	<a href="#">Learn more</a>

Connect  
Manage instance state  
Instance settings  
Networking  
**Security**  
Change security groups  
Get Windows password  
**Modify IAM role**  
Image and templates  
Monitor and troubleshoot

CloudShell Feedback

Nifty smicap +10%

CloudShell Feedback

Nifty smicap +10%

10.Policies: Check that the role has appropriate permissions to interact with ECS and other AWS services.

11.Restart ECS Agent: Restart the ECS agent after verifying configurations:  
docker restart ecs-agent

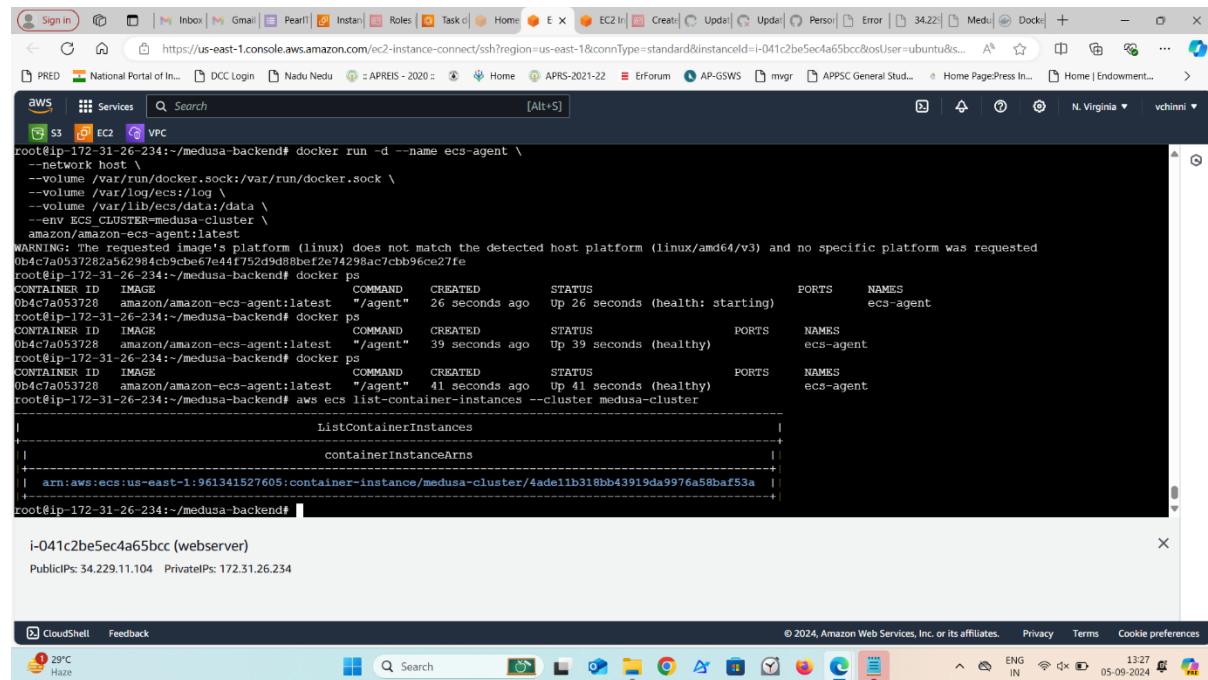
12.Inspect Container Instance Status: After ensuring configurations and permissions, check if the instance appears in the ECS console or via CLI:  
aws ecs list-container-instances --cluster medusa-cluster

Run the ECS agent container with the updated command:

```
docker run -d --name ecs-agent \ --network host \ --volume
/var/run/docker.sock:/var/run/docker.sock \ --volume /var/log/ecs:/log \ --
```

```
volume /var/lib/ecs/data:/data \ --env ECS_CLUSTER=medusa-cluster \
amazon/amazon-ecs-agent:latest
```

Check Cluster Registration: Verify if the ECS instance is now registered with the cluster: aws ecs list-container-instances --cluster medusa-cluster



```
root@ip-172-31-26-234:~/medusa-backend# docker run -d --name ecs-agent \
--network host \
--volume /var/run/docker.sock:/var/run/docker.sock \
--volume /var/log/ecs:/log \
--volume /var/lib/ecs/data:/data \
--env ECS_CLUSTER=medusa-cluster \
amazon/amazon-ecs-agent:latest
WARNING: The requested image's platform (linux) does not match the detected host platform (linux/amd64/v3) and no specific platform was requested
0b4c7a0537282a562984cb9cbe67e4f752d9d8bfe2e74298ac7ccb96ce27fe
root@ip-172-31-26-234:~/medusa-backend# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0b4c7a053728 amazon/amazon-ecs-agent:latest "/agent" 26 seconds ago Up 26 seconds (health: starting) ecs-agent
root@ip-172-31-26-234:~/medusa-backend# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0b4c7a053728 amazon/amazon-ecs-agent:latest "/agent" 39 seconds ago Up 39 seconds (healthy) ecs-agent
root@ip-172-31-26-234:~/medusa-backend# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0b4c7a053728 amazon/amazon-ecs-agent:latest "/agent" 41 seconds ago Up 41 seconds (healthy) ecs-agent
root@ip-172-31-26-234:~/medusa-backend# aws ecs list-container-instances --cluster medusa-cluster
|          ListContainerInstances           |
|          +-----+                         |
|          | containerInstanceArns           |
|          +-----+                         |
|          | arn:aws:ecs:us-east-1:961341527605:container-instance/medusa-cluster/4ade11b31bb43919da9976a58baf53a |
|          +-----+                         |
root@ip-172-31-26-234:~/medusa-backend#
```

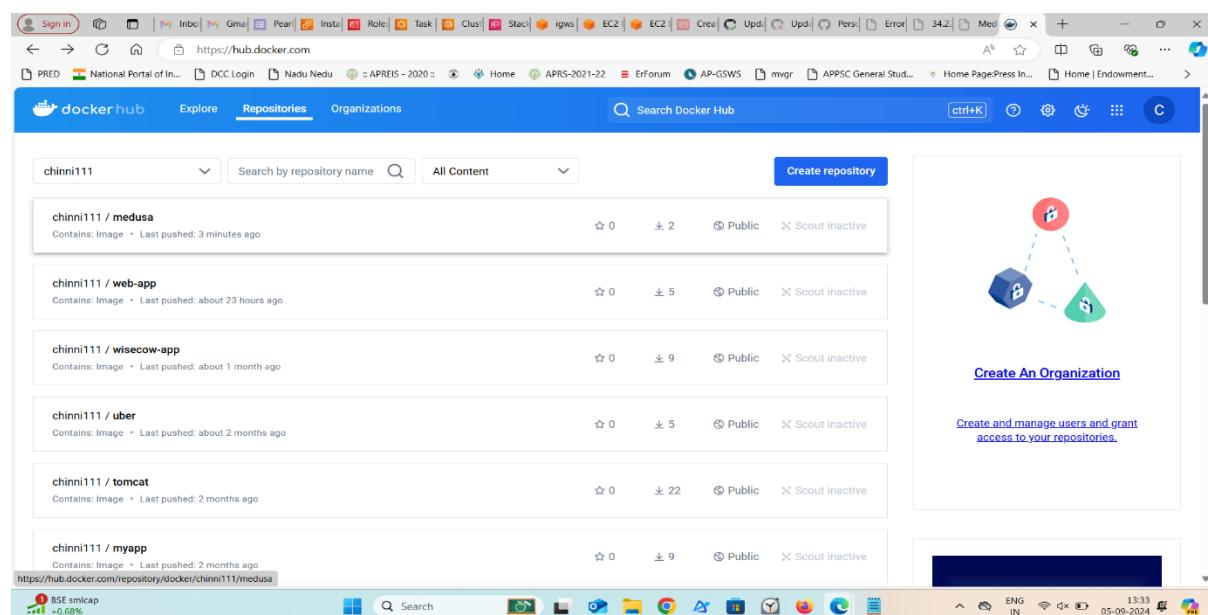
i-041c2be5ec4a65bcc (webserver)  
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences 29°C Haze ENG IN 13:27 05-09-2024

Verify Container Status: Check the status of the ECS agent container: docker ps

We see our image is deployed to ECS service.

Our image is pushed to docker hub



# Docker Scout is a tool for scanning container images for vulnerabilities.

The screenshot shows the Docker Scout Overview page for organization 'chinni111'. The left sidebar includes sections like Overview, Policies (with 1 policy), Images, Base images, Packages, Vulnerabilities, Exceptions (NEW), Integrations, and Settings. A 'Guided setup' section indicates 1 of 4 complete. The main area displays four cards:

- DOCKER SCOUT POLICY**  
**Default non-root user**  
0%  
0/1 Images comply  
Violation: 1 (+1 in last 7 days)
- DOCKER SCOUT POLICY**  
**No AGPL v3 licenses**  
100%  
1/1 Images comply  
Packages: 0 (no change in last 7 days)
- DOCKER SCOUT POLICY**  
**No fixable critical or high vulnerabilities**  
0%  
0/1 Images comply  
Vulnerabilities: 14 (+14 in last 7 days)
- DOCKER SCOUT POLICY**  
**No high-profile vulnerabilities**  
0%  
0/1 Images comply  
Vulnerability: 1 (+1 in last 7 days)

At the bottom, there's a link to 'https://scout.docker.com/reports/org/chinni111/policies/non-root-user/config/non-root-user?stream=latest-indexed'.

The screenshot shows the Docker Scout Vulnerabilities trends page. It features a chart titled 'Vulnerabilities trends' showing the count of vulnerabilities over time. The chart shows a significant spike in critical vulnerabilities starting around August 25, 2024. The legend indicates:

- Critical: 3 (+1)
- High: 13 (+12)
- Medium: 12 (+8)
- Low: 3 (+2)

A note at the bottom states: "No recently disclosed CVEs. We didn't detect any recently disclosed CVEs in the context of this environment and organization."

The screenshot shows the Docker Scout Images page for organization 'chinni111'. The left sidebar includes sections like Overview, Policies (with 1 policy), Images (selected), Base images, Packages, Vulnerabilities, Exceptions (NEW), Integrations, and Settings. A 'Guided setup' section indicates 1 of 4 complete. The main area lists repositories and images:

Repository	Most Recent Image	OS/Arch	Last pushed	Vulnerabilities	Policies status
chinni111/medusa (hub.docker.com)	latest	amd64	7 minutes ago	3 Critical, 13 High, 12 Medium, 3 Low	Compliance, Improved, Worsened

At the bottom, there are links for 'Rows per page: 15' and '1-1 of 1'.

Scout Dashboard - chinni111

## Vulnerabilities

An in-depth look at the vulnerabilities affecting your images.

Severity	Vulnerability	Package	CVSS score	Detected in	Fix available	Fix Version
Critical	<a href="#">CVE-2023-24540</a>	pkg:golang/stdlib@v1.20.3	9.8	<a href="#">1 image</a>	Yes	1.20.4
Critical	<a href="#">CVE-2024-24790</a>	pkg:golang/stdlib@v1.20.3	9.8	<a href="#">1 image</a>	Yes	1.21.11
Critical	<a href="#">CVE-2023-45133</a>	pkg:npm/%40babel/traverse@v7.22.5	9.3	<a href="#">1 image</a>	Yes	7.23.2

Rows per page: 25 | 1-3 of 3

Scout Dashboard - chinni111

## Vulnerabilities

An in-depth look at the vulnerabilities affecting your images.

Severity	Vulnerability	Package	CVSS score	Detected in	Fix available	Fix Version
High	<a href="#">CVE-2022-0235</a>	pkg:npm/node-fetch@v1.7.3	8.8	<a href="#">1 image</a>	Yes	2.6.7
High	<a href="#">CVE-2024-29415</a>	pkg:npm/v@2.0.0	8.1	<a href="#">1 image</a>	No	-
High	<a href="#">CVE-2018-20225</a>	pkg:ppi/pip@v24.0	7.8	<a href="#">1 image</a>	No	-
High	<a href="#">CVE-2023-29403</a>	pkg:golang/stdlib@v1.20.3	7.8	<a href="#">1 image</a>	Yes	1.20.5
High	<a href="#">CVE-2022-25883</a>	pkg:npm/semver@v7.3.7	7.5	<a href="#">1 image</a>	Yes	7.5.2
High	<a href="#">CVE-2023-39325</a>	pkg:golang/stdlib@v1.20.3	7.5	<a href="#">1 image</a>	Yes	1.20.10
High	<a href="#">CVE-2023-44487</a>	pkg:golang/stdlib@v1.20.3	7.5	<a href="#">1 image</a>	Yes	1.20.10
High	<a href="#">CVE-2023-45283</a>	pkg:golang/stdlib@v1.20.3	7.5	<a href="#">1 image</a>	Yes	1.20.11
High	<a href="#">CVE-2023-45288</a>	pkg:golang/stdlib@v1.20.3	7.5	<a href="#">1 image</a>	Yes	1.21.9
High	<a href="#">CVE-2024-24784</a>	pkg:golang/stdlib@v1.20.3	7.5	<a href="#">1 image</a>	Yes	1.21.8
High	<a href="#">CVE-2024-24791</a>	pkg:golang/stdlib@v1.20.3	7.5	<a href="#">1 image</a>	Yes	1.21.12
High	<a href="#">CVE-2023-24539</a>	pkg:golang/stdlib@v1.20.3	7.3	<a href="#">1 image</a>	Yes	1.20.4
High	<a href="#">CVE-2023-29400</a>	pkg:golang/stdlib@v1.20.3	7.3	<a href="#">1 image</a>	Yes	1.20.4

Scout Dashboard - chinni111

## Vulnerabilities

An in-depth look at the vulnerabilities affecting your images.

Severity	Vulnerability	Package	CVSS score	Detected in	Fix available	Fix Version
Medium	<a href="#">CVE-2023-29406</a>	pkg:golang/stdlib@v1.20.3	6.5	<a href="#">1 image</a>	Yes	1.20.6
Medium	<a href="#">CVE-2023-45857</a>	pkg:npm/axios@v24.0 <a href="#">1 more</a>	6.5	<a href="#">1 image</a>	Yes	1.6.0
Medium	<a href="#">CVE-2024-28863</a>	pkg:npm/tar@v6.1.11	6.5	<a href="#">1 image</a>	Yes	6.2.1
Medium	<a href="#">CVE-2023-39318</a>	pkg:golang/stdlib@v1.20.3	6.1	<a href="#">1 image</a>	Yes	1.20.8
Medium	<a href="#">CVE-2023-39319</a>	pkg:golang/stdlib@v1.20.3	6.1	<a href="#">1 image</a>	Yes	1.20.8
Medium	<a href="#">CVE-2024-29041</a>	pkg:npm/express@v4.18.2	6.1	<a href="#">1 image</a>	Yes	4.19.2
Medium	<a href="#">CVE-2019-3902</a>	pkg:pypi/mercurial@v4.8.2	5.9	<a href="#">1 image</a>	Yes	4.9
Medium	<a href="#">CVE-2024-24789</a>	pkg:golang/stdlib@v1.20.3	5.5	<a href="#">1 image</a>	Yes	1.21.11
Medium	<a href="#">CVE-2020-15168</a>	pkg:npm/node-fetch@v1.7.3	5.3	<a href="#">1 image</a>	Yes	2.6.1
Medium	<a href="#">CVE-2023-29409</a>	pkg:golang/stdlib@v1.20.3	5.3	<a href="#">1 image</a>	Yes	1.20.7
Medium	<a href="#">CVE-2023-39326</a>	pkg:golang/stdlib@v1.20.3	5.3	<a href="#">1 image</a>	Yes	1.20.12
Medium	<a href="#">CVE-2023-45284</a>	pkg:golang/stdlib@v1.20.3	5.3	<a href="#">1 image</a>	Yes	1.20.11

The screenshot shows the Scout Docker interface. On the left, a sidebar for user 'chinni111' includes 'Overview', 'Policies', 'Images', 'Base images', 'Packages', 'Vulnerabilities' (which is selected), 'Exceptions', 'Integrations', 'Settings', 'Repository settings', 'Notifications', and 'Billing'. A 'Guided setup' section indicates 1 of 4 complete. The main area is titled 'Vulnerabilities' and contains a table of findings:

Severity	Vulnerability	Package	CVSS score	Detected in	Fix available	Fix Version
Low	CVE-2010-0928	pkg.deb/debian/openssl@1.1.1n-0%2Bdeb10u7os..._...	1 image	No	-	-
Low	CVE-2021-3487	pkg.deb/debian/binutils@2.31.1-16os_distro=buster...	1 image	No	-	-
Low	CVE-2023-42282	pkg.npm/tp@2.0.0	1 image	Yes	2.0.1	-

At the bottom, there are buttons for 'Rows per page' (25) and '1-3 of 3'.

To access your Docker image using a Kubernetes Load Balancer, you need to follow these steps:

## First create EKS cluster

The screenshot shows the AWS EKS console. The left sidebar includes 'Clusters' (selected), 'Amazon EKS Anywhere', 'Enterprise Subscriptions', 'Related services' (with 'Amazon ECR' and 'AWS Batch'), and 'Console settings', 'Documentation', and 'Submit feedback'. The main area shows 'Clusters (1) info' with a table:

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created
jawan	Active	1.24 <a href="#">Upgrade now</a>	Extended support until January 31, 2025	Extended	7 minutes ago

At the bottom, there are buttons for 'CloudShell' and 'Feedback'.

## Create node group

The screenshot shows the AWS CloudFormation console with the URL <https://us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks?filteringText=&filteringStatus=active&viewNested=true>. The left sidebar includes sections for Stacks, StackSets, Exports, Application Composer (IaC generator), Registry (Public extensions, Activated extensions, Publisher), Spotlight, and Feedback. The main area displays a table titled 'Stacks (2)' with columns for Stack name, Status, Created time, and Description. There are two entries:

Stack name	Status	Created time	Description
<a href="#">eksctl-jawan-nodegroup-jawan-nginx-public1</a>	CREATE_COMPLETE	2024-09-05 13:45:03 UTC+0530	EKS Managed Nodes (SSH access: true) [created by eksctl]
<a href="#">eksctl-jawan-cluster</a>	CREATE_COMPLETE	2024-09-05 13:32:22 UTC+0530	EKS cluster (dedicated VPC: true, dedicated IAM: true) [created and managed by eksctl]

## Config kubectl

### 1. Deploy Your Docker Image to a Kubernetes Cluster

Assuming you already have a Kubernetes cluster running, you need to deploy your Docker image to it.

1. Create a Kubernetes Deployment: Create a deployment.yaml file
2. Apply the Deployment: `kubectl apply -f deployment.yaml`
3. Verify Deployment: `kubectl get deployments && kubectl get pods`

### 2. Expose the Deployment with a Service

Create a service.yaml file to expose your deployment:

This configuration will create a LoadBalancer service that routes traffic to your pods.

### 3. Apply the Service: `kubectl apply -f service.yaml`

### 4. Check the Load Balancer

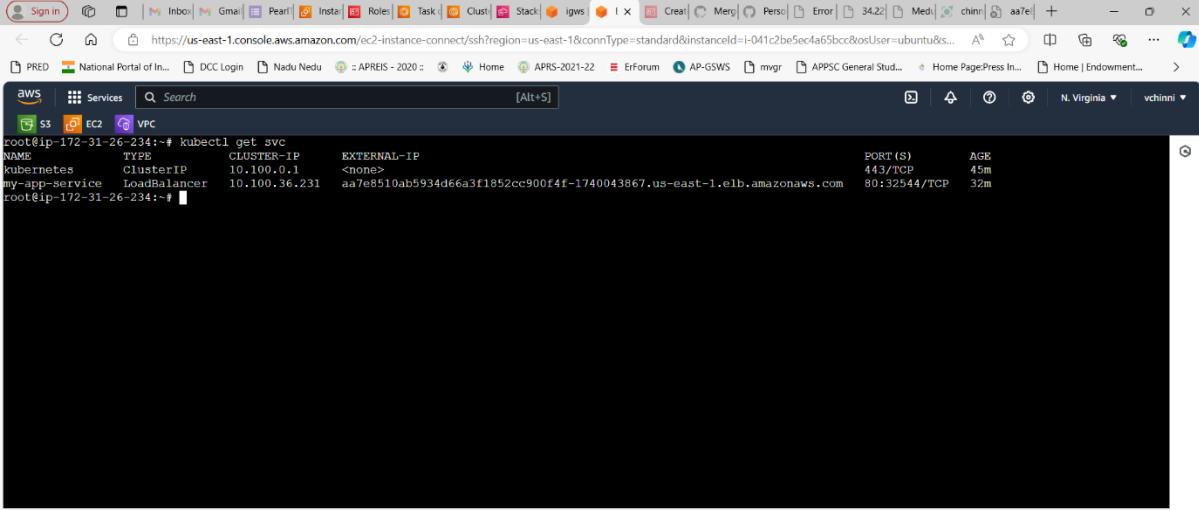
After applying the service, Kubernetes will provision a cloud load balancer and associate it with your service.

1. Get the Service Details: `kubectl get services`

Look for the EXTERNAL-IP of my-app-service. This is the public IP or DNS name of your Load Balancer.

## 2. Access Your Application

Open your web browser and navigate to the EXTERNAL-IP or DNS name.

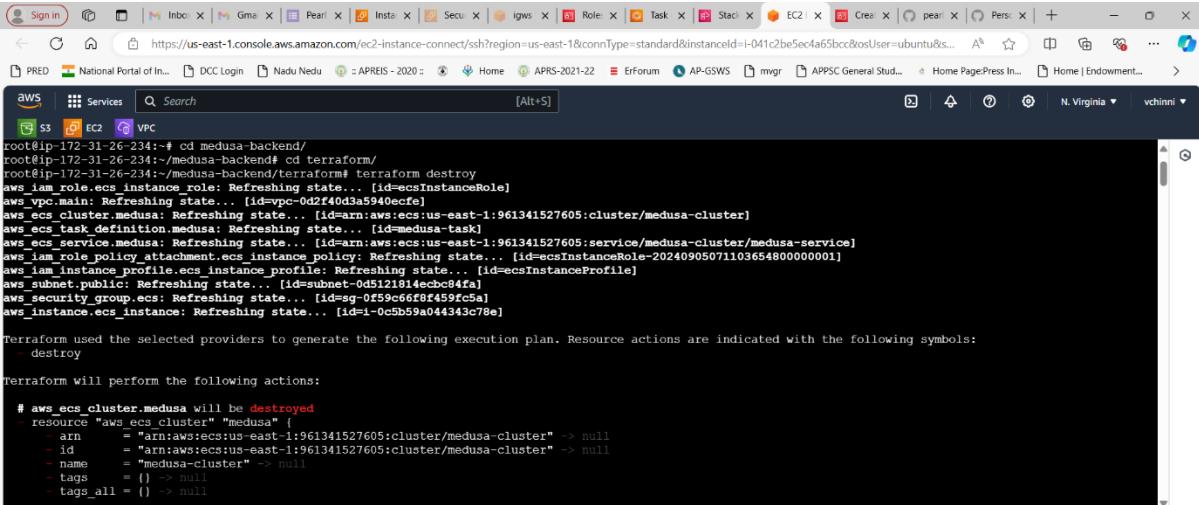


```
root@ip-172-31-26-234:~# kubectl get svc
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP
kubernetes     ClusterIP 10.100.0.1   <none>
my-app-service LoadBalancer 10.100.36.231 aa7e8510ab5934d66a3f1852cc900f4f-1740043867.us-east-1.elb.amazonaws.com 80:32544/TCP
root@ip-172-31-26-234:~#
```

i-041c2be5ec4a65bcc (webserver)  
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234



Destroy the configuration



```
root@ip-172-31-26-234:~# cd medusa-backend/
root@ip-172-31-26-234:~/medusa-backend# cd terraform/
root@ip-172-31-26-234:~/medusa-backend/terraform# terraform destroy
aws iam role.ecs_instance_role: Refreshing state... [id=ecsInstanceRole]
aws vpc main: Refreshing state... [id=vpc-0d2f40d3a5940ecfe]
aws ecs cluster medusa: Refreshing state... [id=arn:aws:ecs:us-east-1:961341527605:cluster/medusa-cluster]
aws ecs task_definition medusa: Refreshing state... [id=medusa-task]
aws ecs service medusa: Refreshing state... [id=arn:aws:ecs:us-east-1:961341527605:service/medusa-cluster/medusa-service]
aws iam role_policy_attachment.ecs_instance_policy: Refreshing state... [id=ecsInstanceRole-20240905071103654800000001]
aws iam instance_profile.ecs_instance_profile: Refreshing state... [id=ecsInstanceProfile]
aws subnet_public.ecs_instance_profile: Refreshing state... [id=subnet-0d5121814ecbc84fa]
aws security_group.ecs: Refreshing state... [id=sg-0f59c66f0f459fc5a]
aws instance.ecs_instance: Refreshing state... [id=i-0c5b59a044343c78e]
```

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:  
- destroy

Terraform will perform the following actions:

```
# aws ecs cluster.medusa will be destroyed
- resource "aws ecs cluster" "medusa" {
  - arn      = "arn:aws:ecs:us-east-1:961341527605:cluster/medusa-cluster" -> null
  - id       = "arn:aws:ecs:us-east-1:961341527605:cluster/medusa-cluster" -> null
  - name     = "medusa-cluster" -> null
  - tags     = {} -> null
  - tags_all = {} -> null}
```

i-041c2be5ec4a65bcc (webserver)  
Public IPs: 34.229.11.104 Private IPs: 172.31.26.234



## Terraform and ECS Deployment

1. ECS Configuration:
  - o The ecs.tf file defines ECS resources with the following configurations:
    - network\_mode set to bridge
    - launch\_type set to EC2
  - o The ECS service does not include network configuration as it is not required for EC2 launch type with bridge mode.
2. Disk Space Issues:
  - o You encountered pod eviction issues in your Kubernetes cluster due to disk space. Here's how disk space was checked:
    - o df -h /var/lib/docker
3. Error Handling in ECS Cluster Deletion:
  - o You encountered the error:  
ClusterContainsContainerInstancesException: The Cluster cannot be deleted while Container Instances are active or draining.
  - o To resolve this, you deregistered the ECS container instances:
  - o aws ecs deregister-container-instance --cluster medusa-cluster --container-instance <container-instance-arn> --force
    - However, the correct container-instance-arn had to be fetched using:
    - aws ecs list-container-instances --cluster medusa-cluster

**Conclusion:** In this project, you set up and deployed the Medusa backend on an AWS Ubuntu server using PostgreSQL, Redis, and ECS. You automated infrastructure provisioning and deployment using Terraform and GitHub Actions, ensuring a streamlined CI/CD pipeline. Despite facing challenges like ECS cluster deletion and resource management, you resolved issues efficiently, demonstrating your cloud expertise and troubleshooting skills.