Task3

AWS EC2 Deployment

Launch EC2 (cost-optimized: t2.micro / t3.micro)

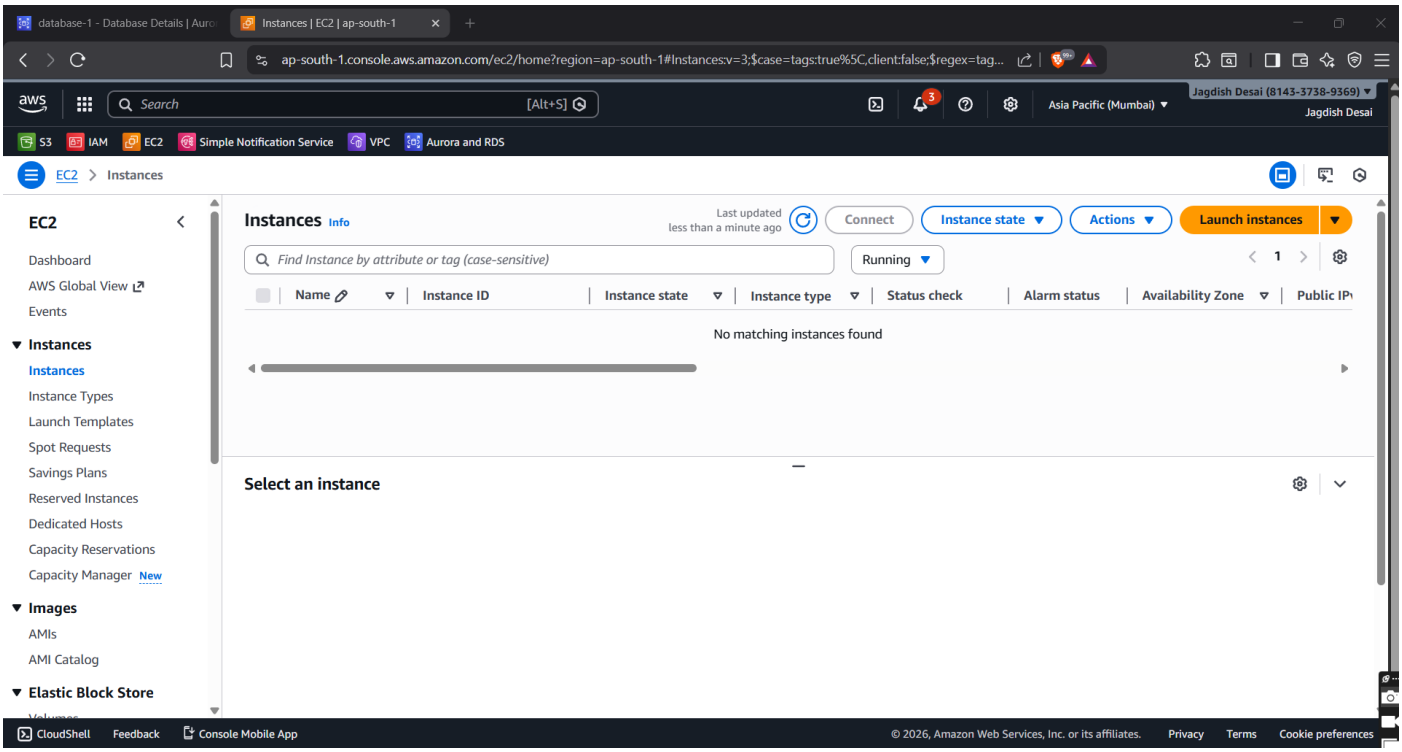
Install Docker

Run containers on EC2

Steps

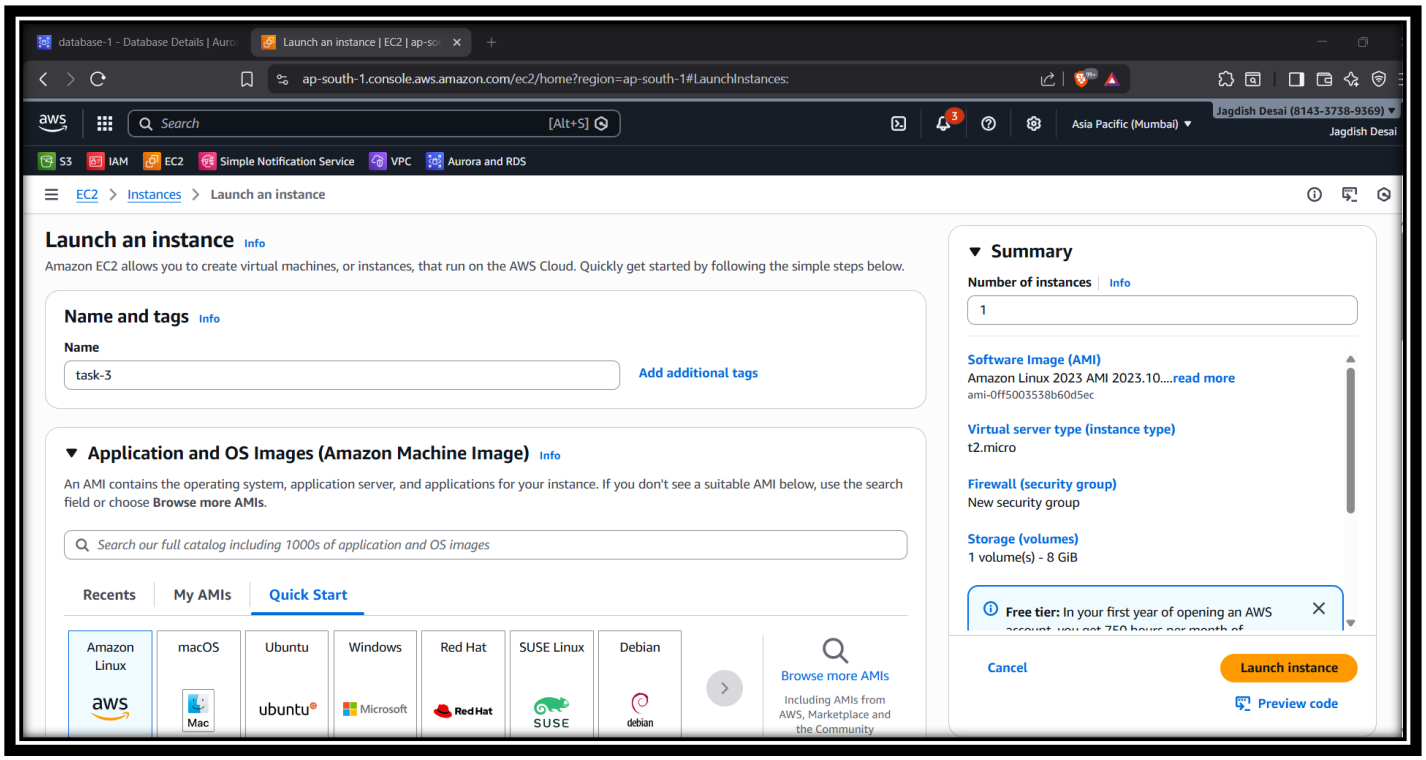
Step 1: Launch EC2 (t2.micro / t3.micro)

1. Open **AWS Console**
2. **Go to EC2**
3. **Click on Launch instance**



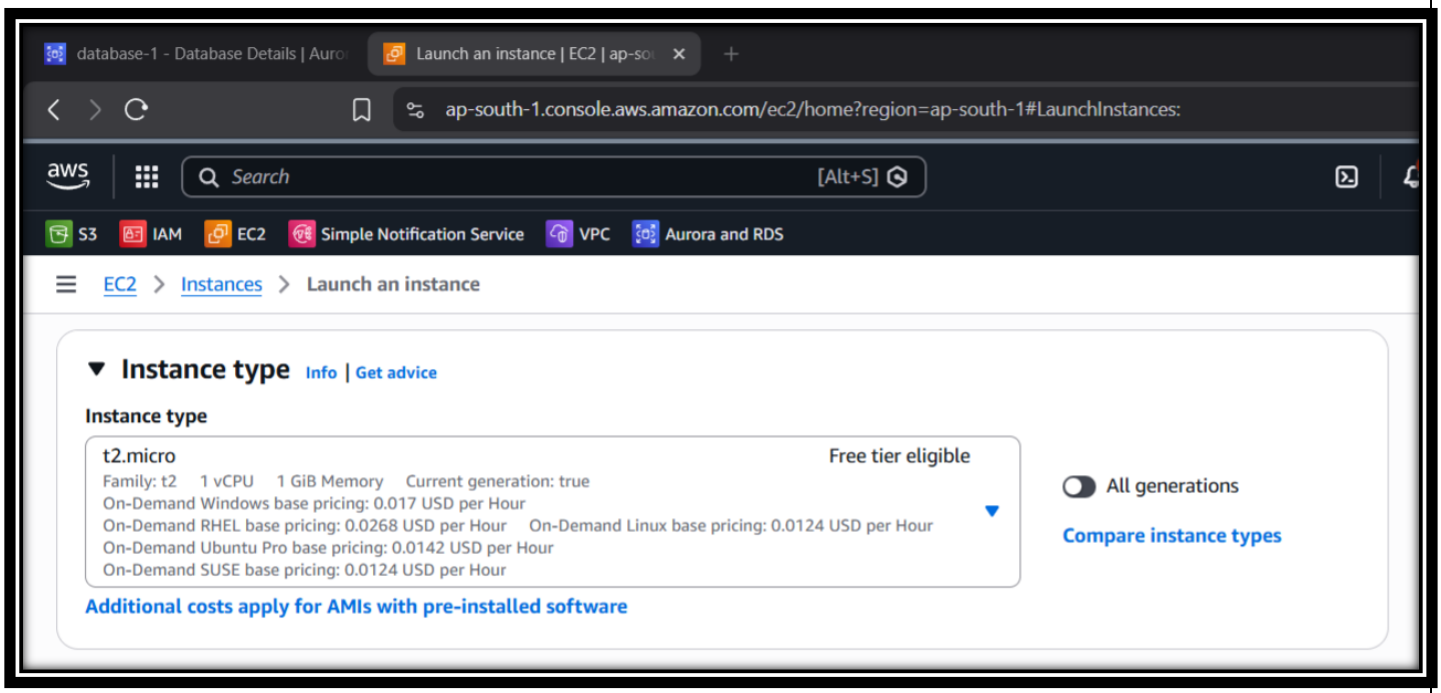
Step 2: Launching instance with task-3

Selecting operating system Amazon Linux



Step 3: Selecting instance type t2.micro

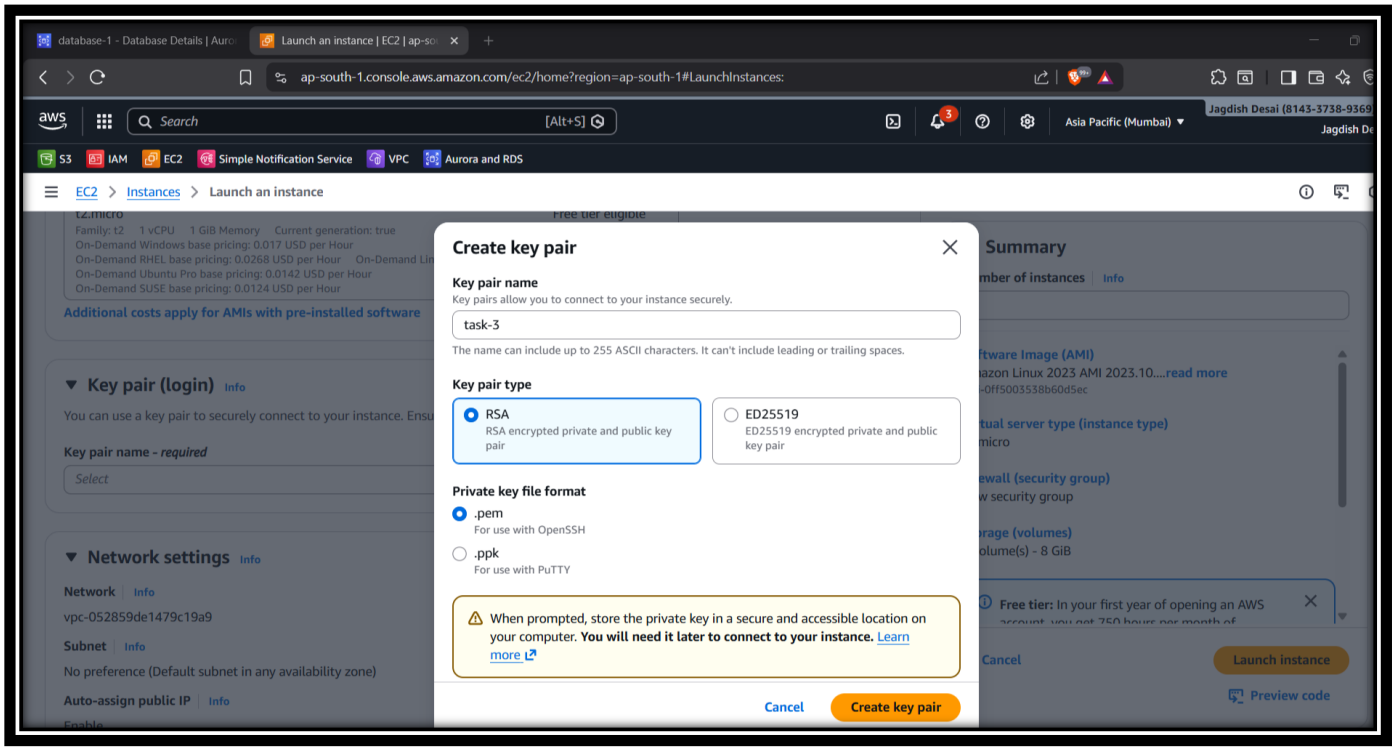
t2.micro is best when we want to learn or run a small app at low cost.



Step 4: Create a key pair or select existing key pair

We can use a key pair to securely connect to our instance.

Select key-pair type and create and download task3.pem file



After Creating key-pair select it:

Step 5: Go to network settings

Select vpc- I choose default vpc **// Virtual network**

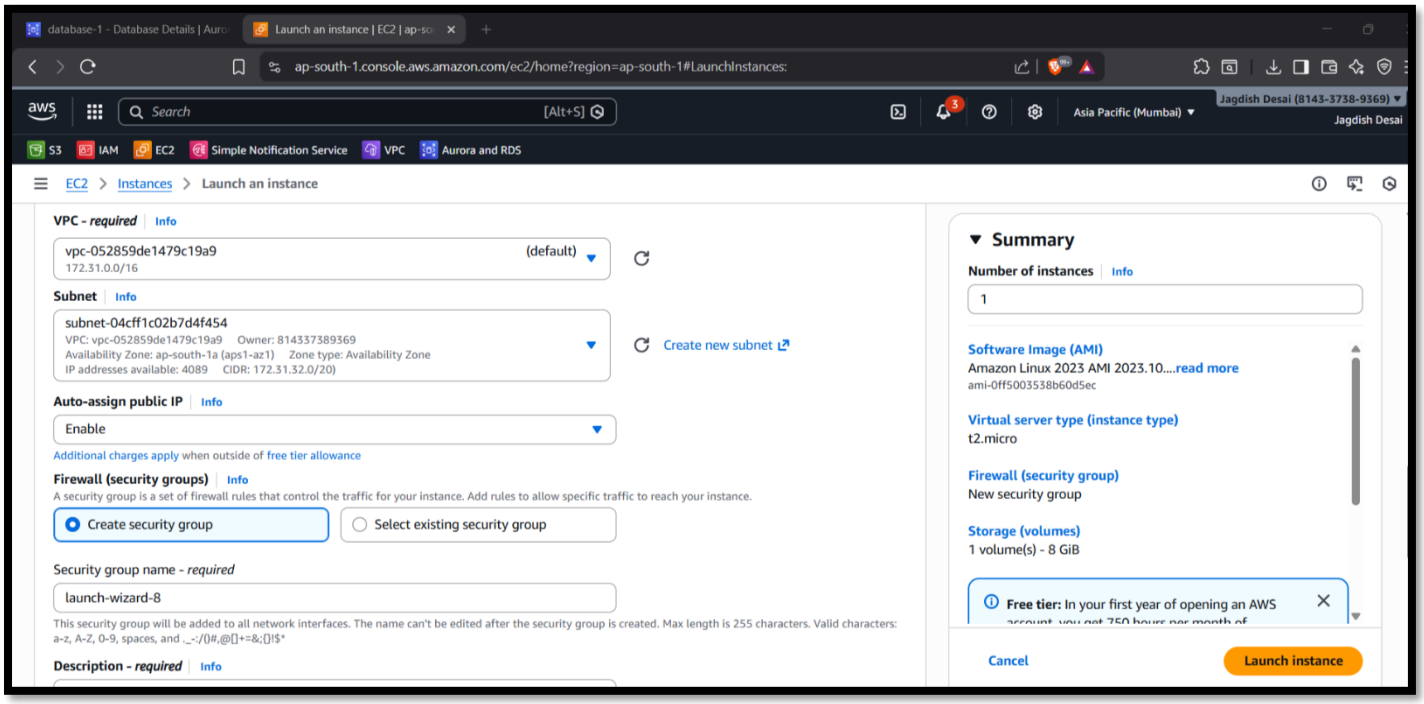
select subnet- I choose subnet of ap-south-1a **//Network segment**

Auto-assign public IP Info- Enable

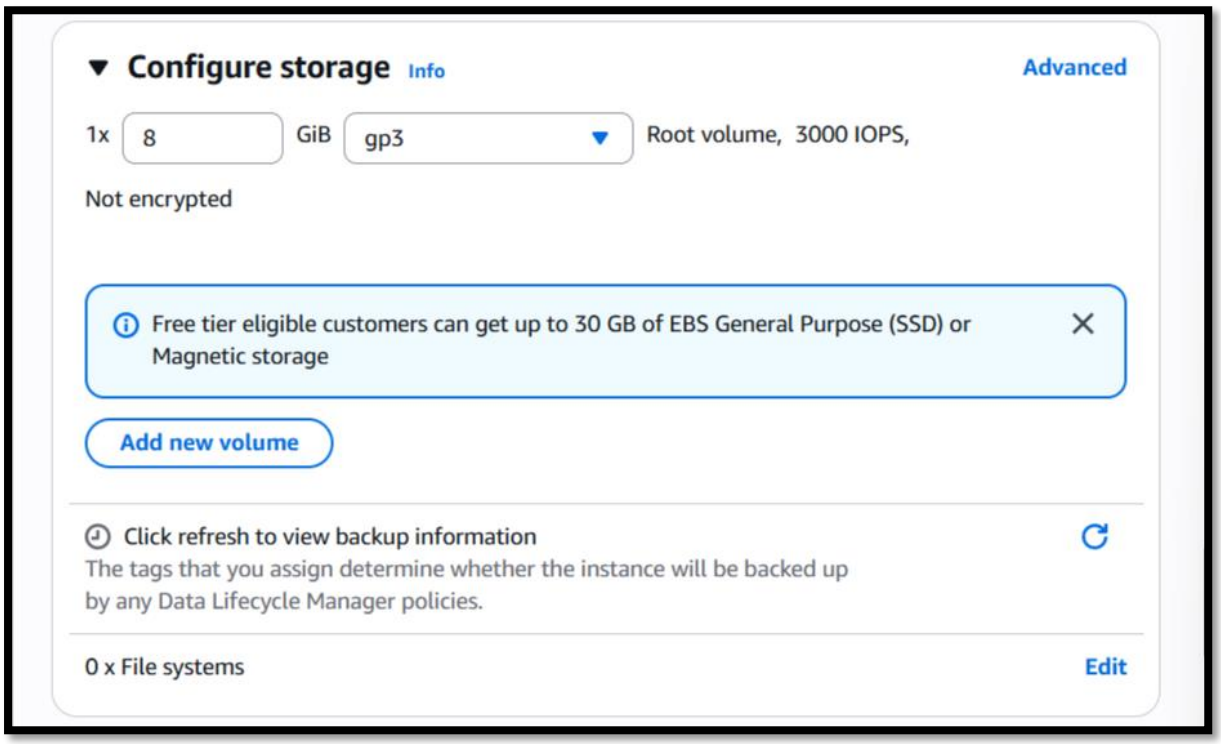
Security group- launch-wizard-8 **//firewall rules**

Allow SSH – 22 (My IP)

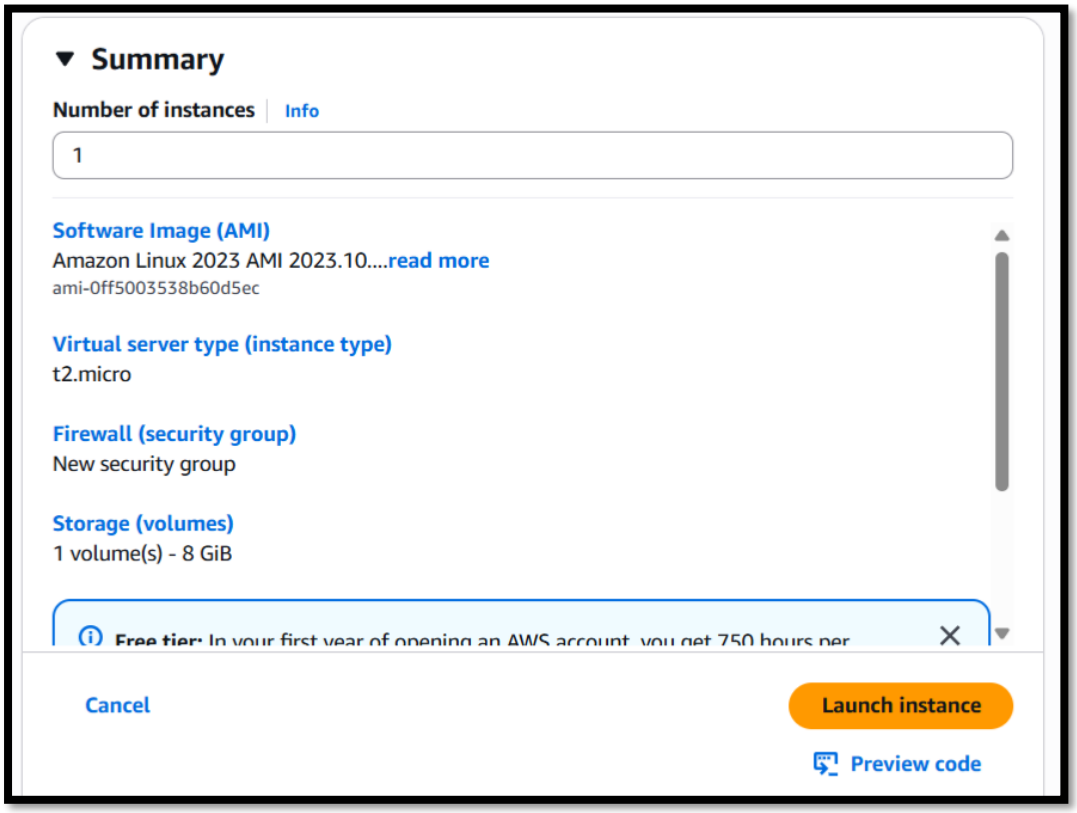
Allow HTTP – 80 (Anywhere)



Step 6: Configure storage according to reqirements I keep default storage.

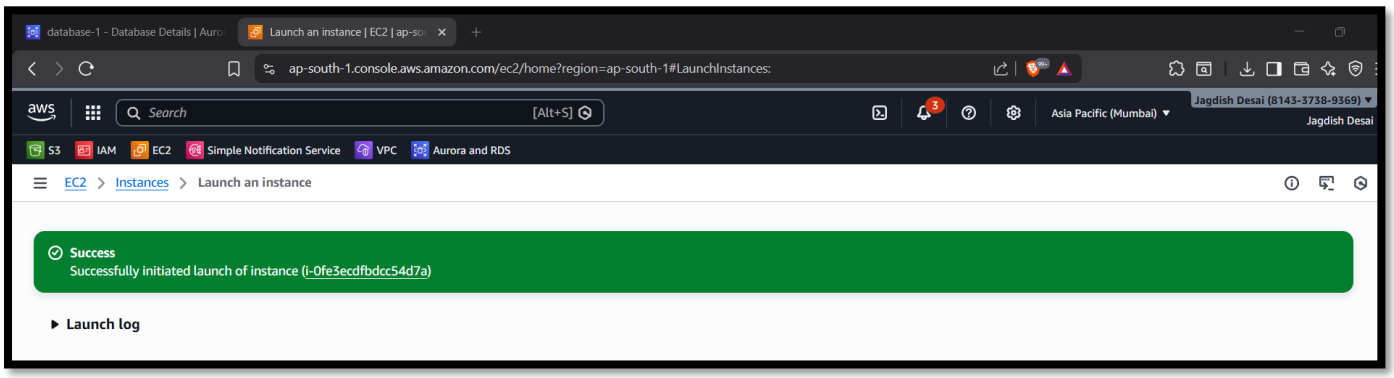


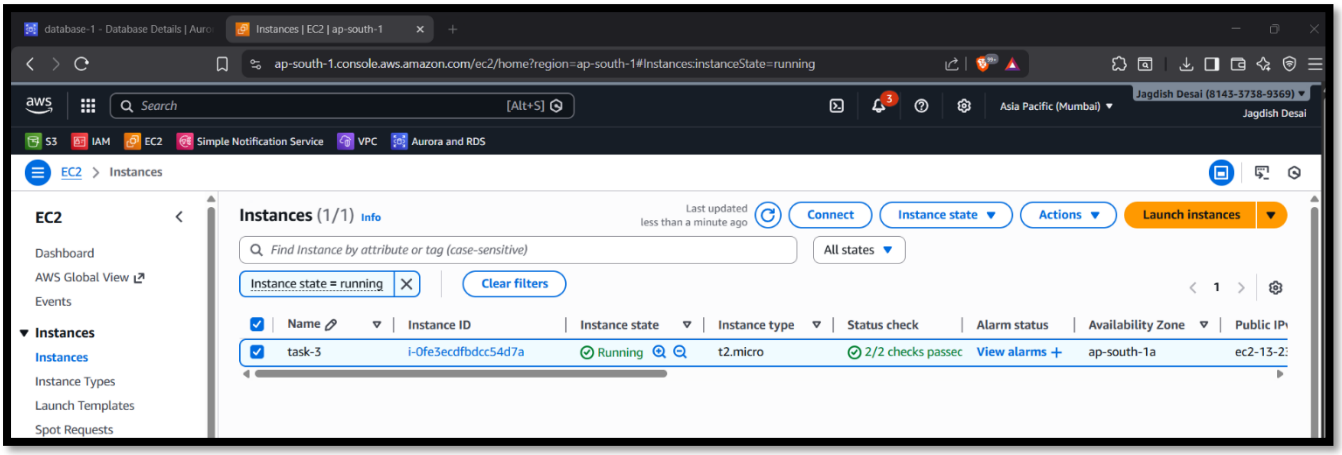
Step 7: See the overview of our task 3 instance



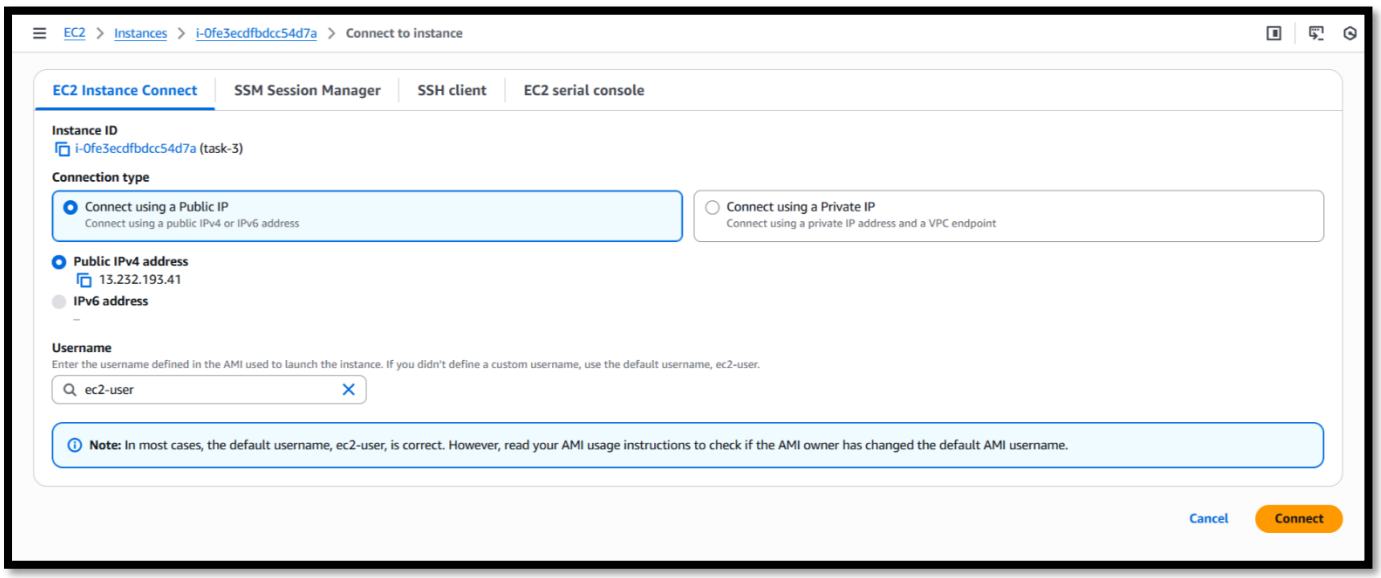
Here we can create number instances according to our requirements.

Step 8: Launch instance





Step 9: Connect Ec2 instance

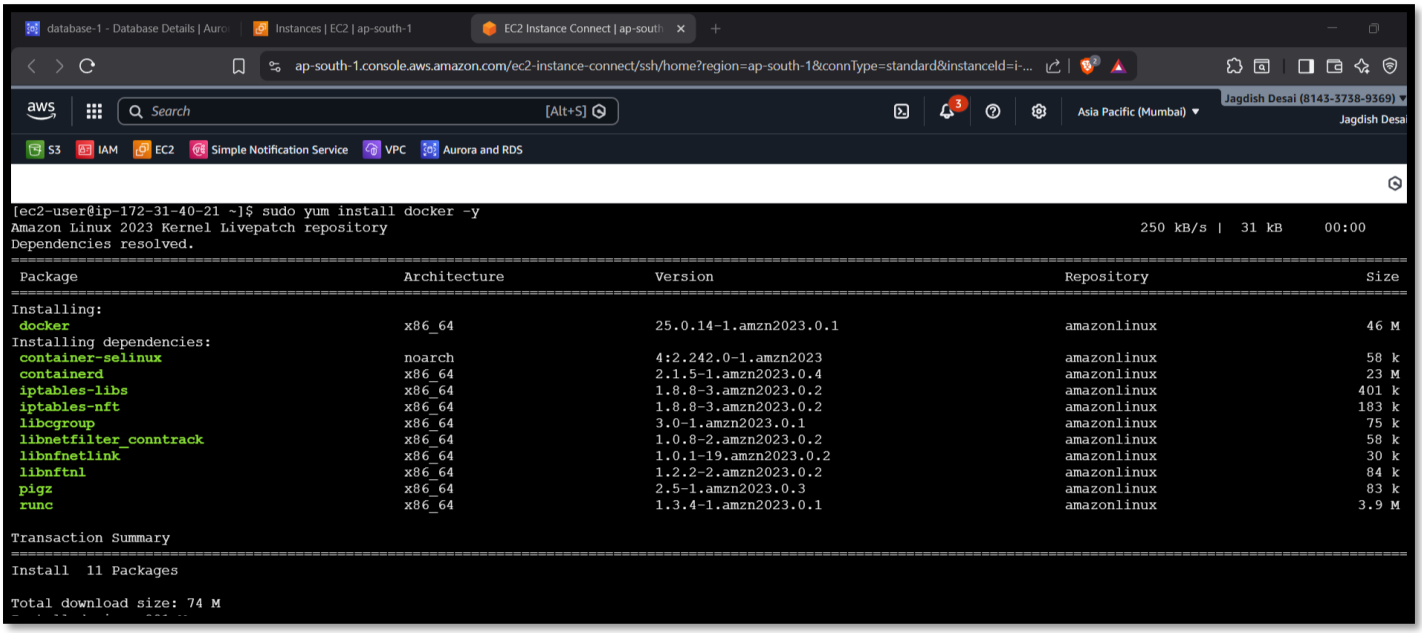


Installing docker –

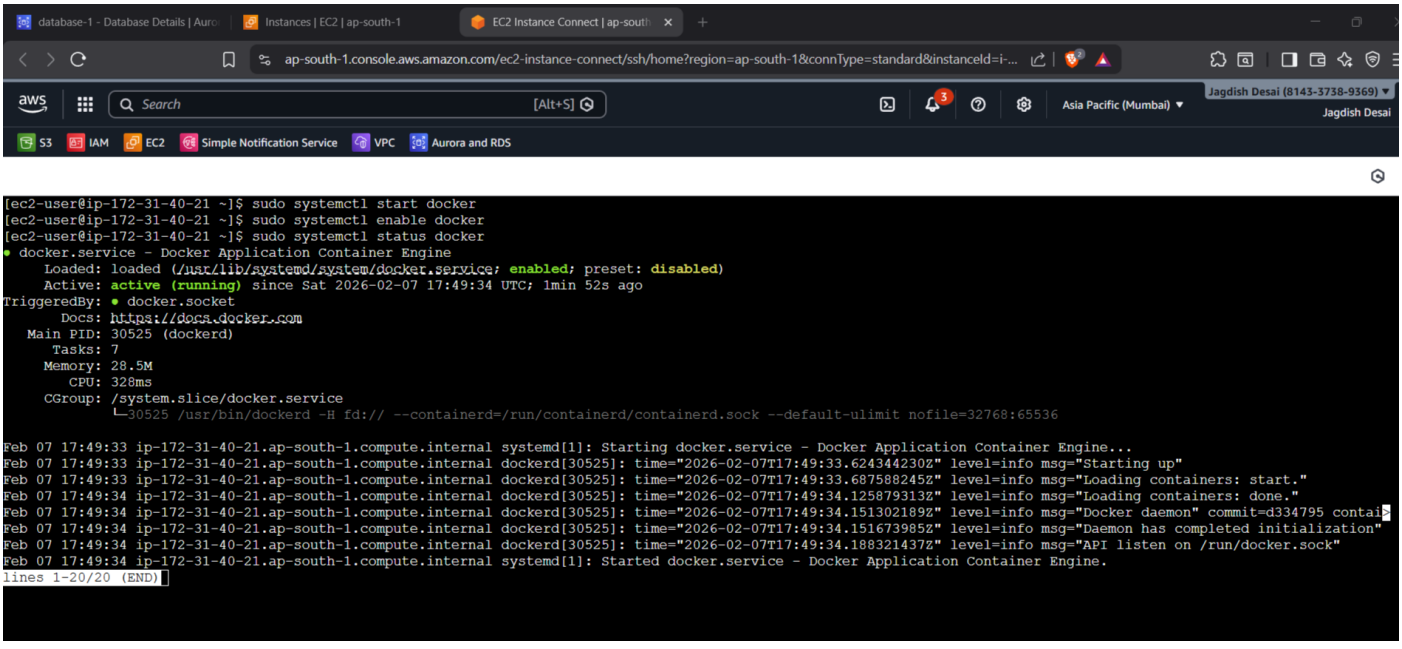
Step 1: Install Docker on EC2

sudo yum install docker -y

sudo systemctl start docker



Status: Running

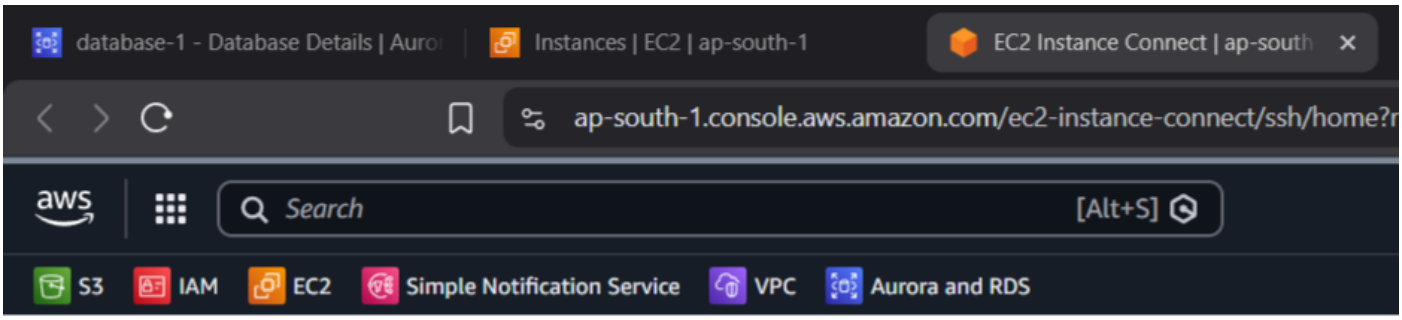


Give permission:

sudo usermod -aG docker ec2-user

allows ec2-user to run Docker commands without sudo

Step 2: Check Docker version




```
[ec2-user@ip-172-31-40-21 ~]$ docker --version
Docker version 25.0.14, build 0bab007
[ec2-user@ip-172-31-40-21 ~]$
```

Step 3: What is happening?

we:

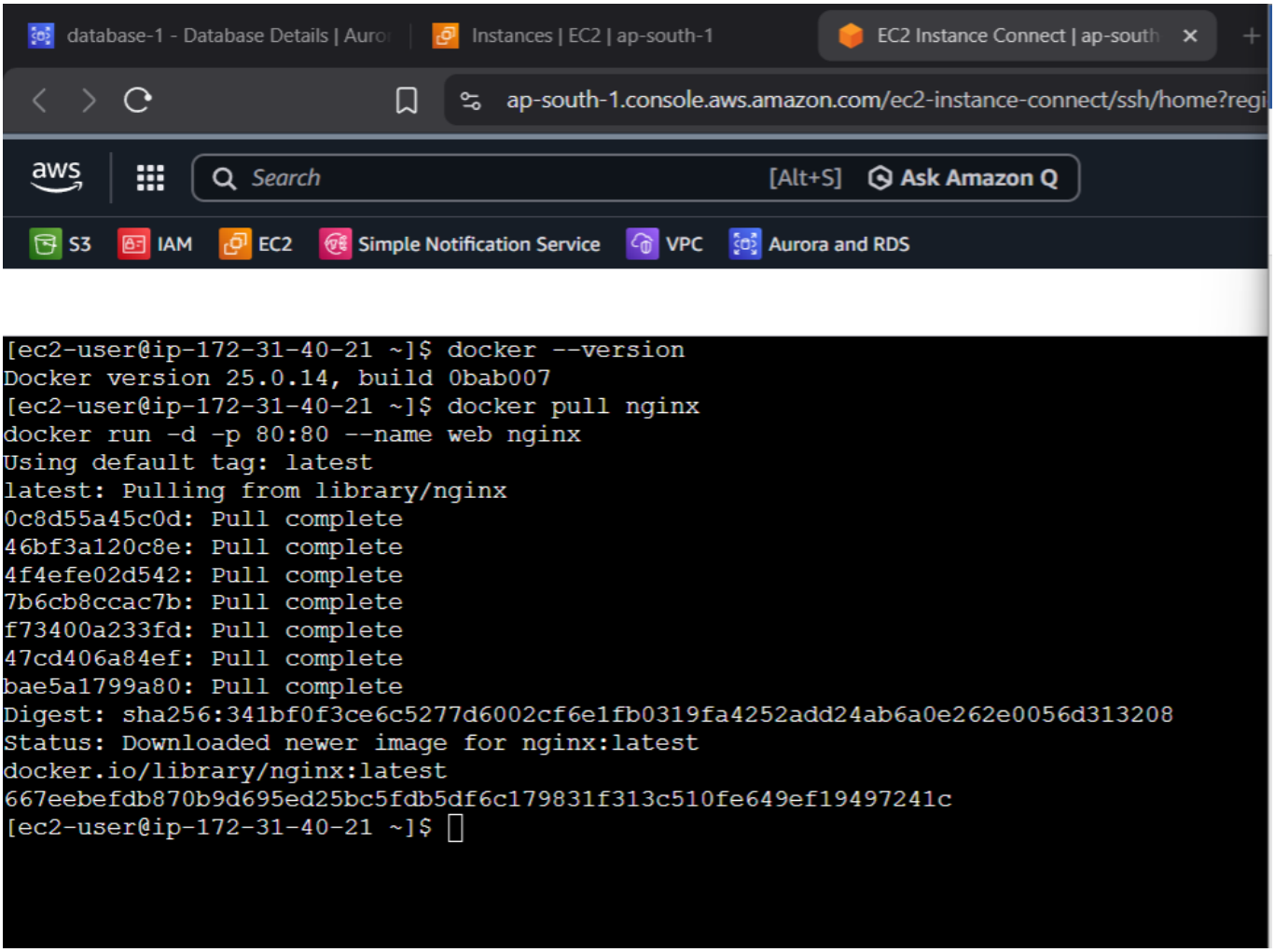
- Using **EC2**
- Running **Docker**
- Pulling **Nginx** from **Docker Hub**
- Hosting a **web server**
- Accessing it using **EC2 Public IP**
-

Step 4: Pull Docker Image

docker pull nginx

explaination:

- Downloads the **nginx image** from **Docker Hub**
- Docker Hub = online image store
- Nginx = web server



Step 3: Run Docker Container

docker run -d -p 80:80 --name web nginx

docker run

Start a container

-d

Run in background (detached mode)

-p 80:80

Connect ports

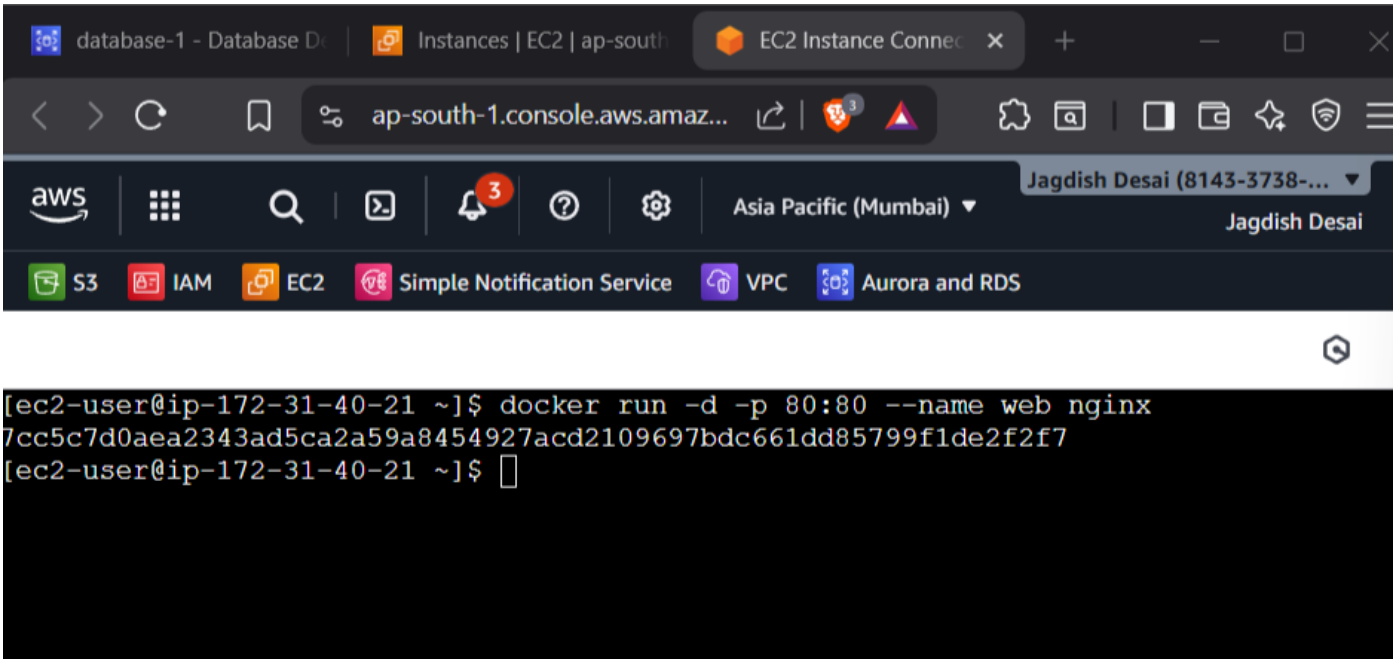
- Left 80 → EC2 port
- Right 80 → container port
- Means: browser traffic goes to container

--name web

Container name = web

□ nginx

👉 Image name



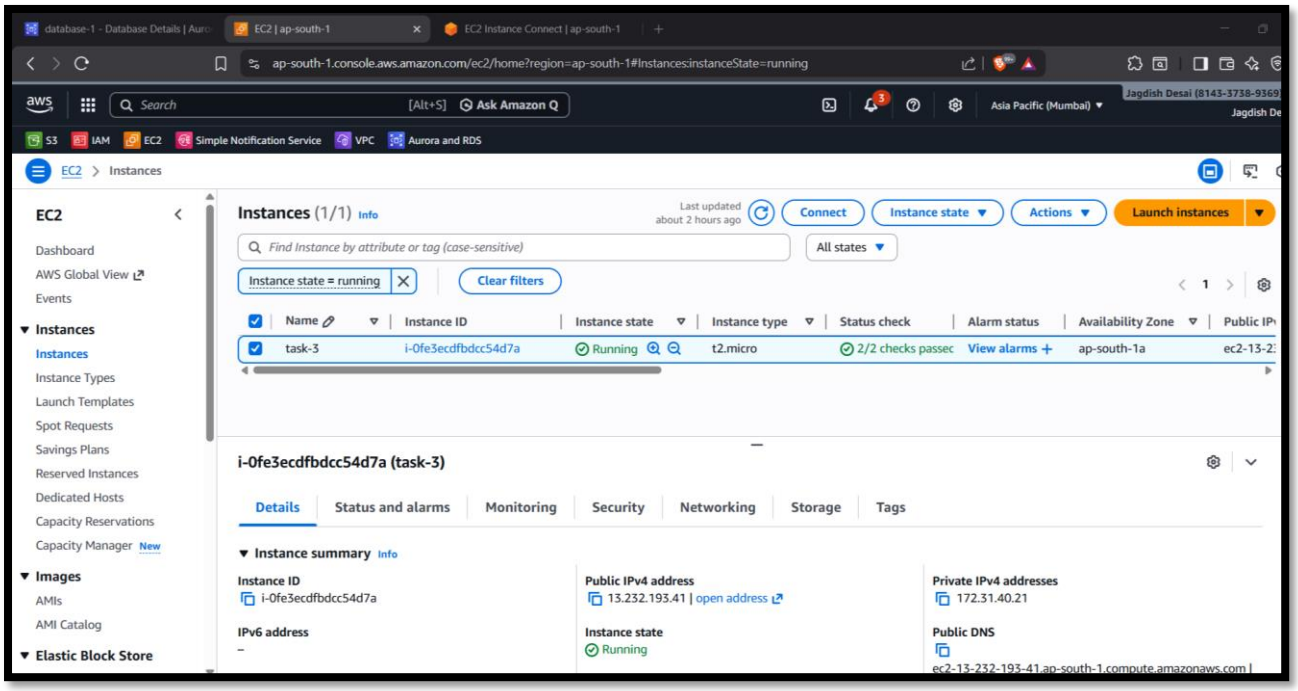
EC2 → Security Group → Inbound Rules

Add:

- Type: **HTTP**
- Port: **80**
- Source: 0.0.0.0/0

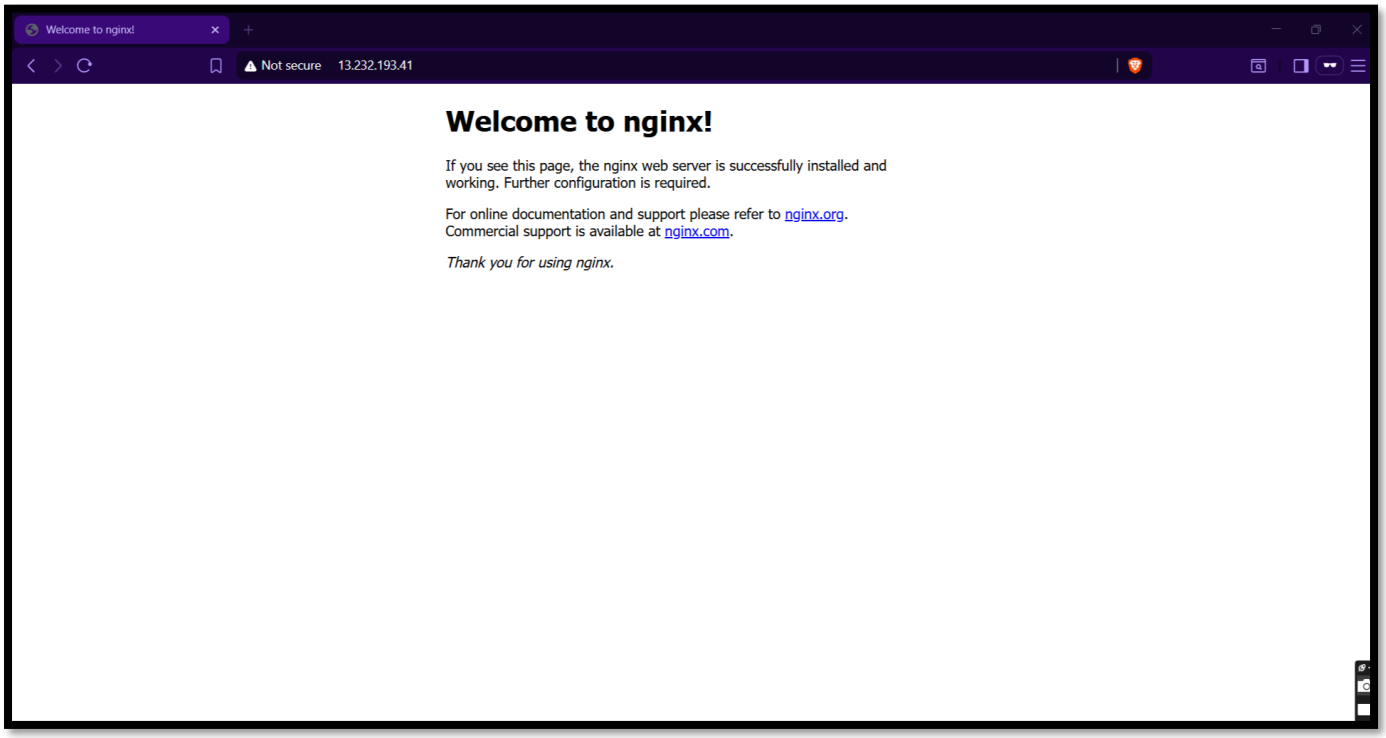
This allows internet users to access your web server

Step 4: Copy Public IP address.



Step 5:Going to browser and paste public ip.

http://13.232.193.41/



Output This confirms:

- EC2 is running
- Docker is working
- Nginx container is live

Creating Own Image

1.**Connect to EC2:**

Create directory – mkdir app

Create- index.html

Create- server.js

Create package.json - npm init -y

2. **Install Required Packages:**

npm install express mysql2 body-parser

3. Install Docker

sudo yum install docker -y
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker ec2-user
exit

4. Login again to EC2

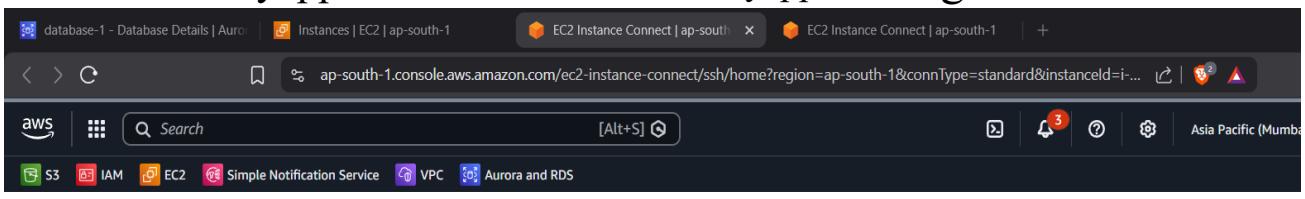
ssh -i yourkey.pem ec2-user@EC2-Public-IP

5. Going to app folder

cd app

6. Build Docker image

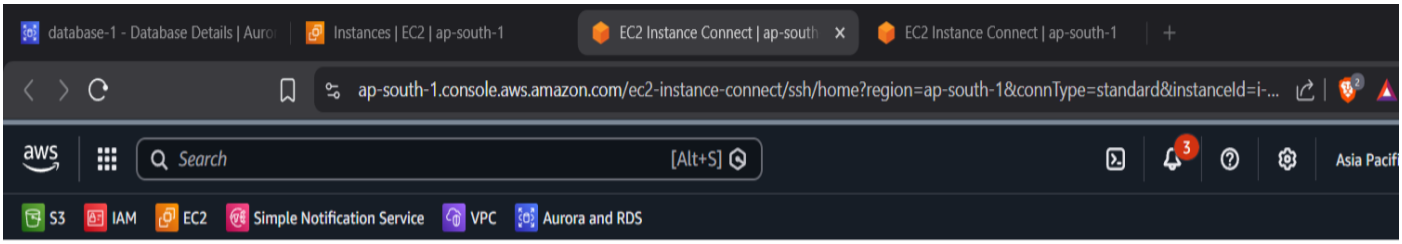
docker build -t myapp . //myapp = image name




```
[ec2-user@ip-172-31-1-172 ~]$ cd app
[ec2-user@ip-172-31-1-172 app]$ docker build -t myapp .
ERROR: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdoc
var/run/docker.sock: connect: permission denied
[ec2-user@ip-172-31-1-172 app]$ sudo docker build -t myapp .
[+] Building 1.7s (10/10) FINISHED                                  docker:default
 => [internal] load build definition from dockerfile                         0.0s
 => => transferring dockerfile: 255B                                         0.0s
 => [internal] load metadata for docker.io/library/node:18                   1.5s
 => [internal] load .dockerignore                                            0.0s
 => => transferring context: 2B                                              0.0s
 => [1/5] FROM docker.io/library/node:18@sha256:c6ae79e38498325db67193d391e6  0.0s
 => [internal] load build context                                            0.2s
 => => transferring context: 109.28kB                                        0.1s
 => CACHED [2/5] WORKDIR /app                                                0.0s
 => CACHED [3/5] COPY package*.json ./                                       0.0s
 => CACHED [4/5] RUN npm install                                             0.0s
 => CACHED [5/5] COPY . .                                                    0.0s
 => exporting to image                                                       0.0s
 => => exporting layers                                                      0.0s
```

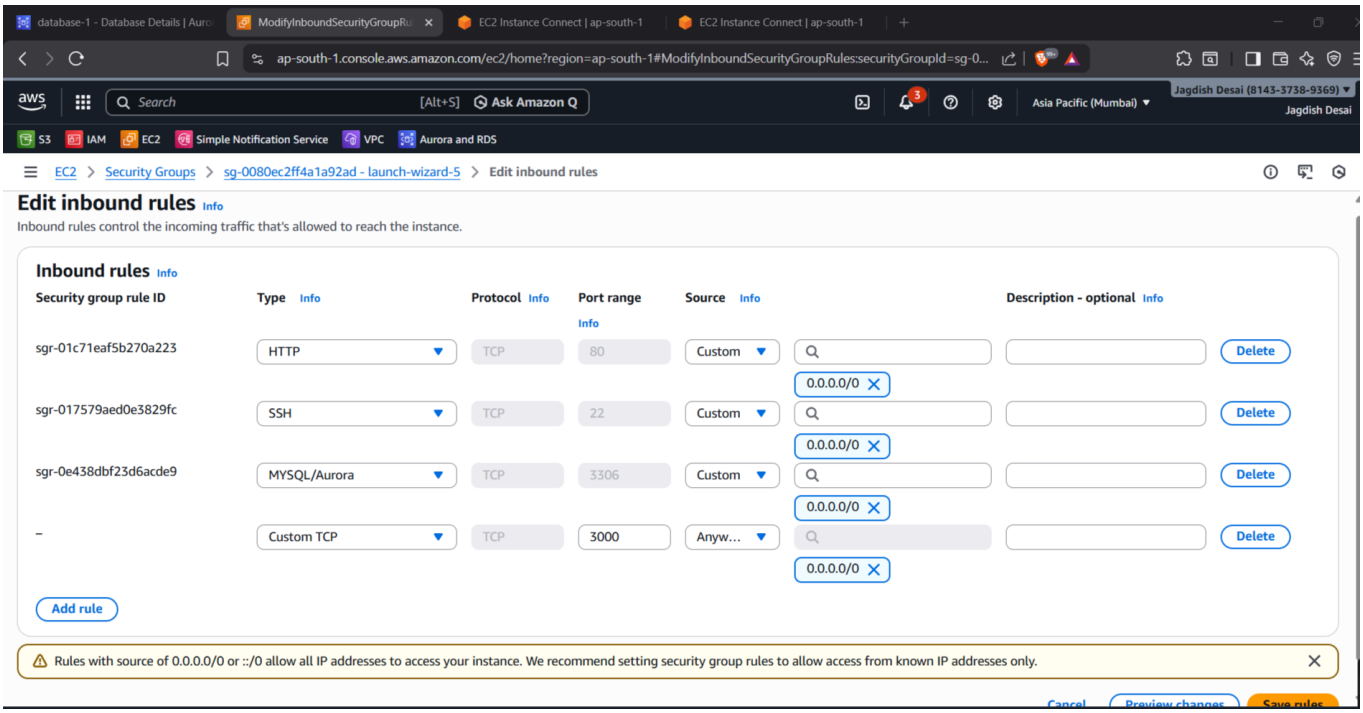
7. Run Docker container

docker run -d -p 3000:3000 --restart always myapp



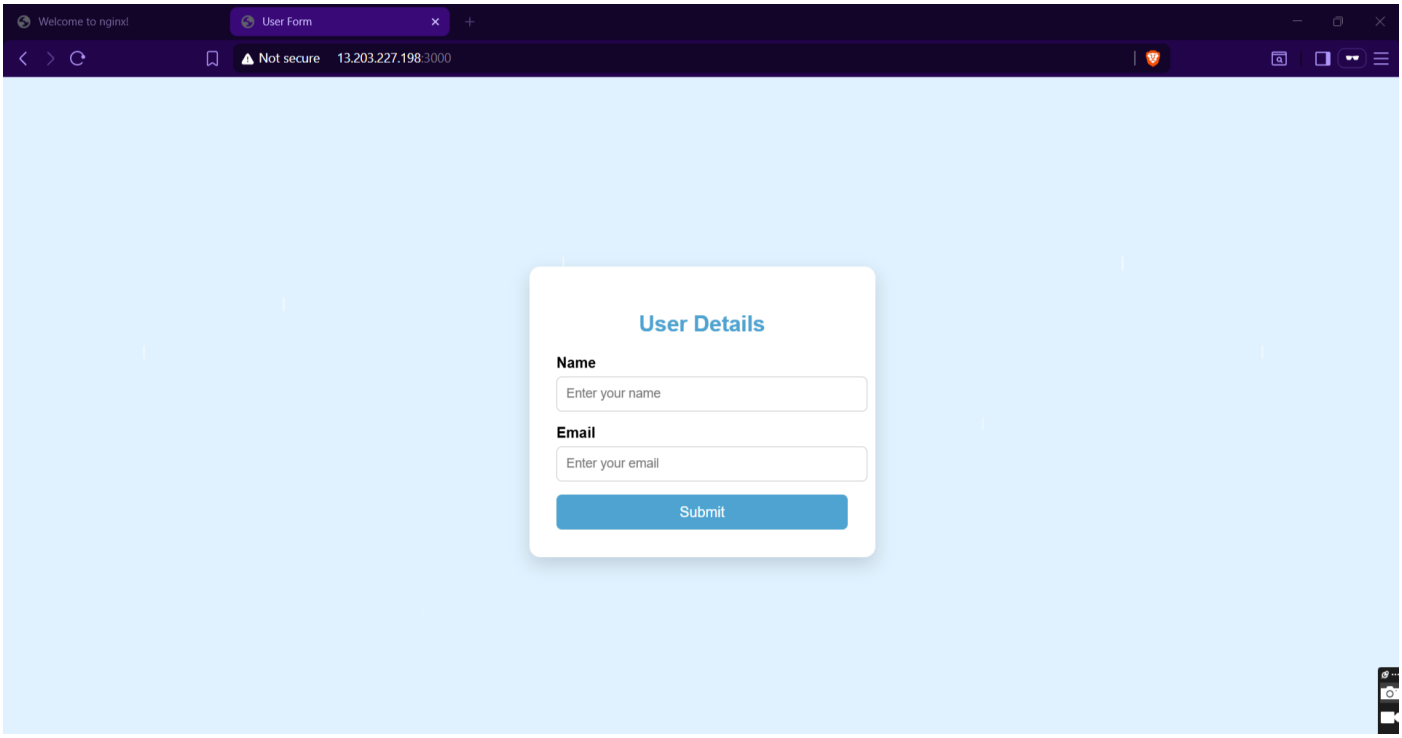

```
[ec2-user@ip-172-31-1-172 app]$ sudo docker run -d -p 3000:3000 --restart always myapp
1b2fcf642c14c7a4617e86582e76a15f122b1934d5b399e0ed1270960808cb83
[ec2-user@ip-172-31-1-172 app]$
```

8. Allow port 3000 in EC2 Security Group



Step 9: Using Public IP accessing app in browser

http://13.203.227.198:3000/



Final Output:

- EC2 running
- Docker installed
- Container running
- App accessible via **EC2 Public IP**

Index.html code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>User Form</title>
  <style>
    body {
      margin: 0;
      padding: 0;
      height: 100vh;
      display: flex;
      justify-content: center;
      align-items: center;
      background: #f8c8dc; /* ice pink background */
      font-family: Arial, sans-serif;
    }

    .form-container {
      background: white;
      padding: 30px;
      border-radius: 12px;
      width: 320px;
      box-shadow: 0 8px 20px rgba(0, 0, 0, 0.15);
    }
```

```
.form-container h2 {
  text-align: center;
  margin-bottom: 20px;
  color: #e75480;
}

label {
  font-weight: bold;
  display: block;
  margin-bottom: 5px;
}
input {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border-radius: 6px;
  border: 1px solid #ccc;
  font-size: 14px;
}
input:focus {
  outline: none;
  border-color: #e75480;
}
button {
  width: 100%;
  padding: 10px;
```

```
    background: #e75480;
    border: none;
    color: white;
    font-size: 16px;
    border-radius: 6px;
    cursor: pointer;
  }
  button:hover {
    background: #d64570;
  }
 </style>
</head>
<body>
<div class="form-container">
  <h2>User Details</h2>
  <form method="POST">
    <label>Name</label>
    <input type="text" name="name" placeholder="Enter your name" required>
    <label>Email</label>
    <input type="email" name="email" placeholder="Enter your email"
required>
      <button type="submit">Submit</button>
  </form>
 </div>
</body>
</html>
```

Server.js

```
const express = require("express");
const mysql = require("mysql2");
const bodyParser = require("body-parser");
const app = express();
app.use(bodyParser.urlencoded({ extended: true }));
const db = mysql.createConnection({
  host: "database-1.chq6mgeaipg2.ap-south-1.rds.amazonaws.com",
  user: "admin",
  password: "shankarrao71",
  database: "appdb"
});
db.connect(err => {
  if (err) throw err;
  console.log("Database Connected");
});
app.post("/", (req, res) => {
  const { name, email } = req.body;
  db.query(
    "INSERT INTO users (name, email) VALUES (?, ?)",
    [name, email],
    () => res.send("Data inserted successfully")
  );
});
app.get("/", (req, res) => {
  res.sendFile(__dirname + "/index.html");
```

```
});
app.listen(3000, () => {
  console.log("Server running on port 3000");
});
db.connect((err) => {
  if (err) {
    console.error("DB connection failed:", err.message);
    process.exit(1);
  }
  console.log("Database Connected");
});
```