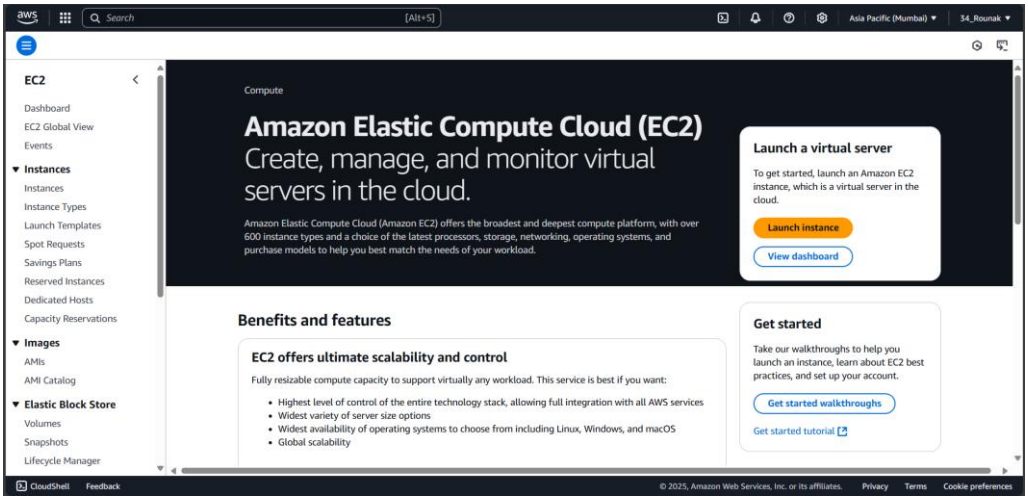
Assignment number: 10

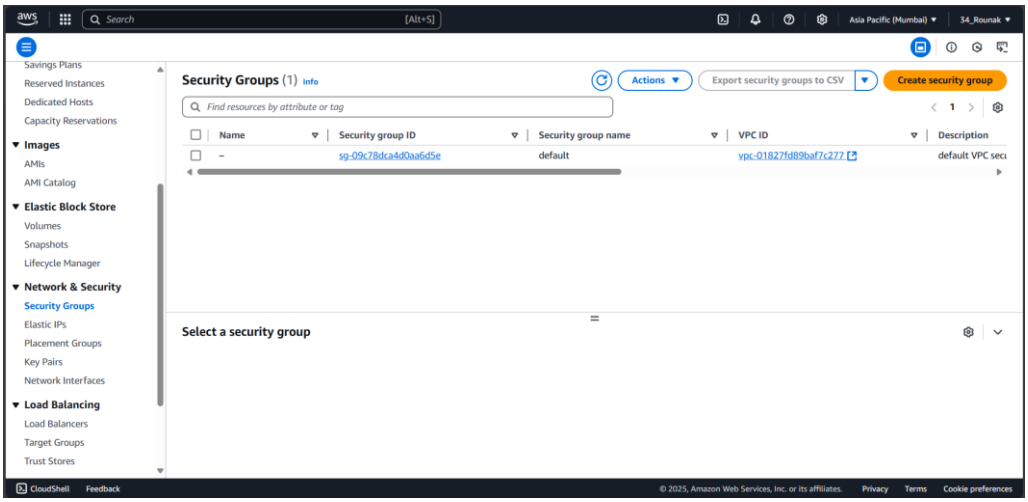
Problem definition: Deploy a project from GitHub to EC2 by creating a new security group and user data.

Step 1: Sign in to your AWS account as the root user and into GitHub.

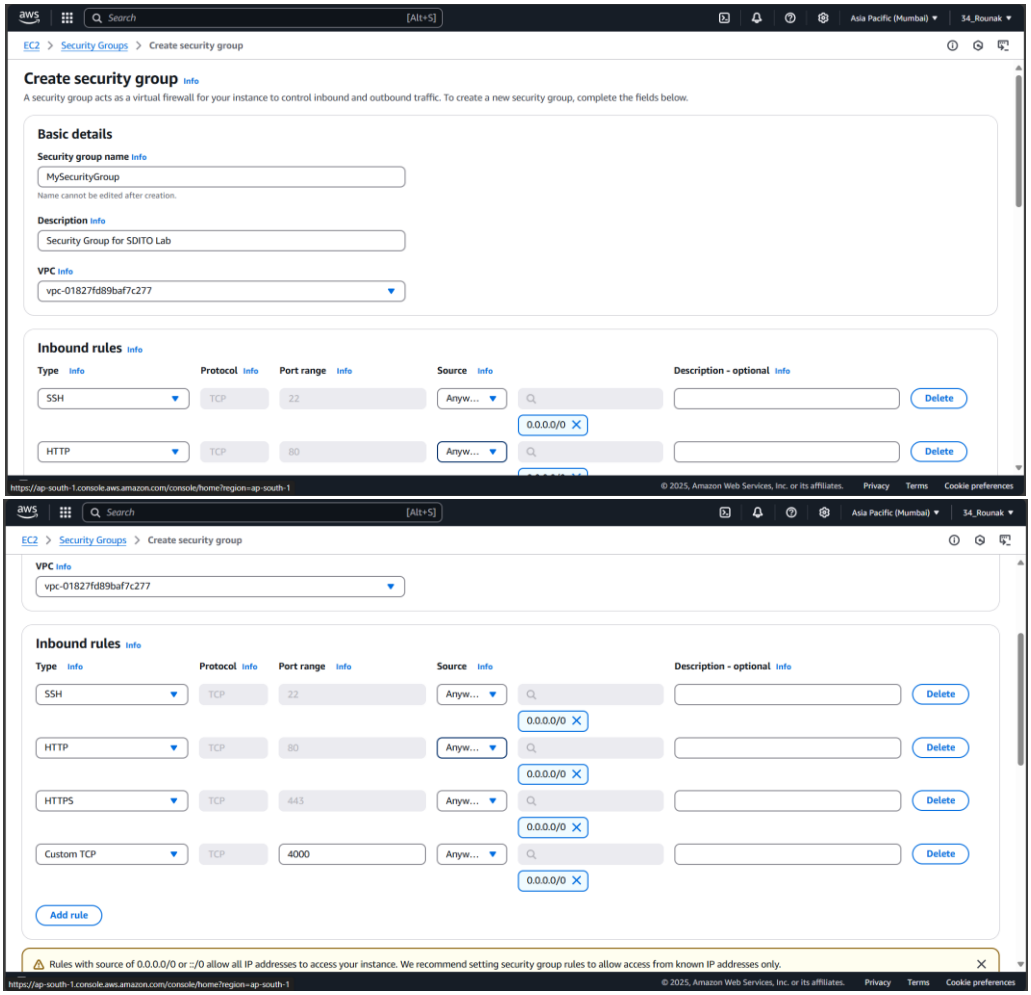
Step 2: Log in to the AWS Management Console, use the search bar to search for “EC2,” and click on the EC2 service.



Step 3: On the left-hand navigation bar, under the **"Network & Security"** section, click on **"Security Groups"**. Select all the security groups listed except for **"default"**. Click on the **"Actions"** button and click on **"Delete all security groups"**.

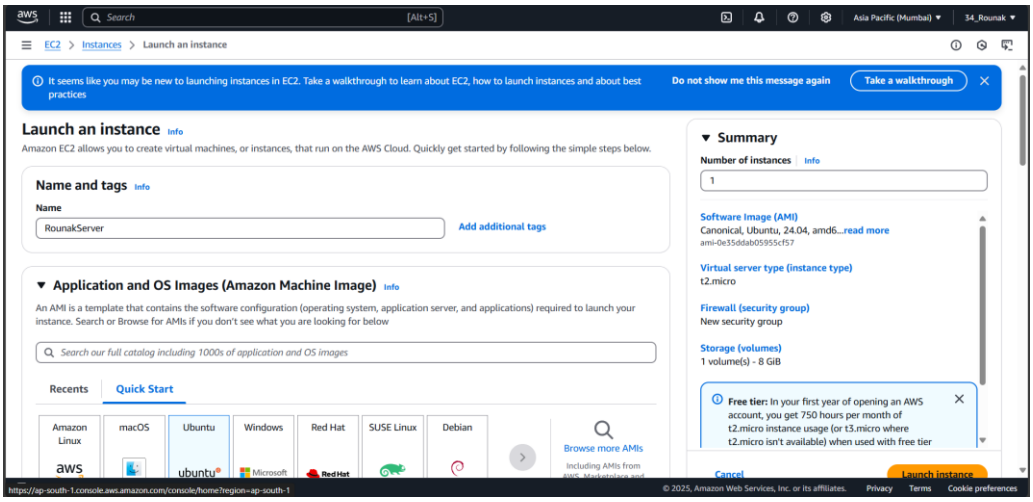


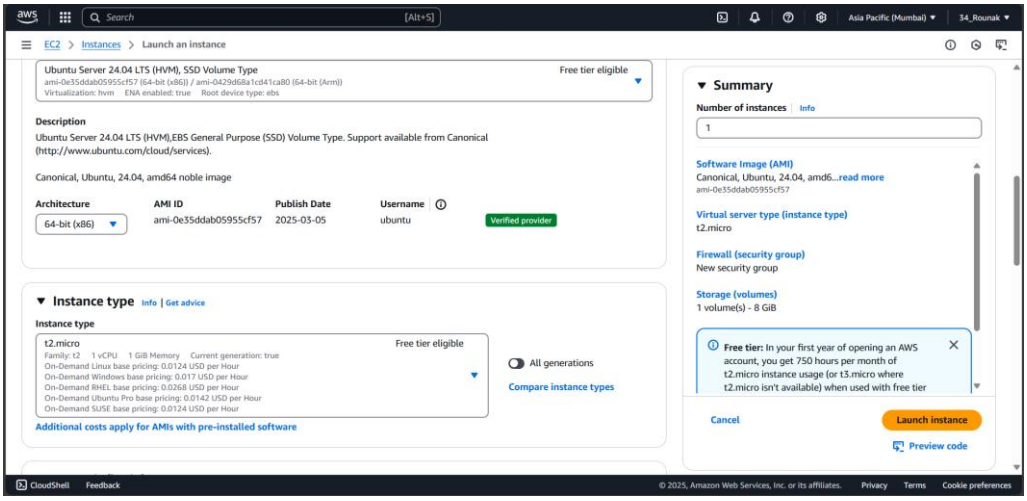
Step 4: Click the **"Create Security Group"** button. Add name and description. Under **“Inbound Rules”**, add the **“Add rule”** to add the following rules: SSH, HTTP, HTTPS and Custom TCP with port 4000. Select 0.0.0.0/0 under source for all rules. We leave the outbound rules and other sections unchanged. Finally click on **“Create security group”.** Pop up is shown stating our rules are created successfully.



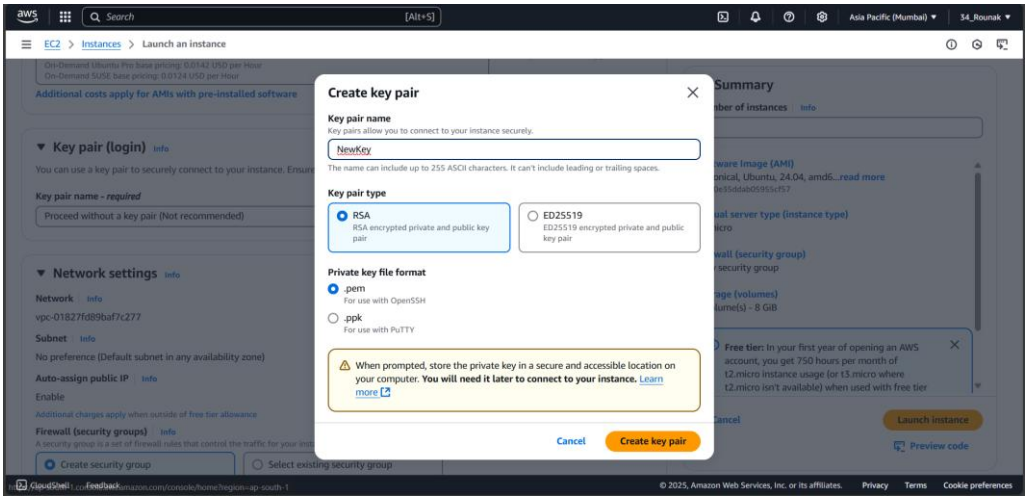
Step 5: On the left-hand navigation bar, under the **"Instances"** section, click on **"Instances"**. Click the **"Launch Instance"** button.

Step 6: In the “Name and Tags” section, enter a descriptive name for your instance. Under “Application and OS Images,” choose Ubuntu. Ensure that the selected instance type meets your requirements.

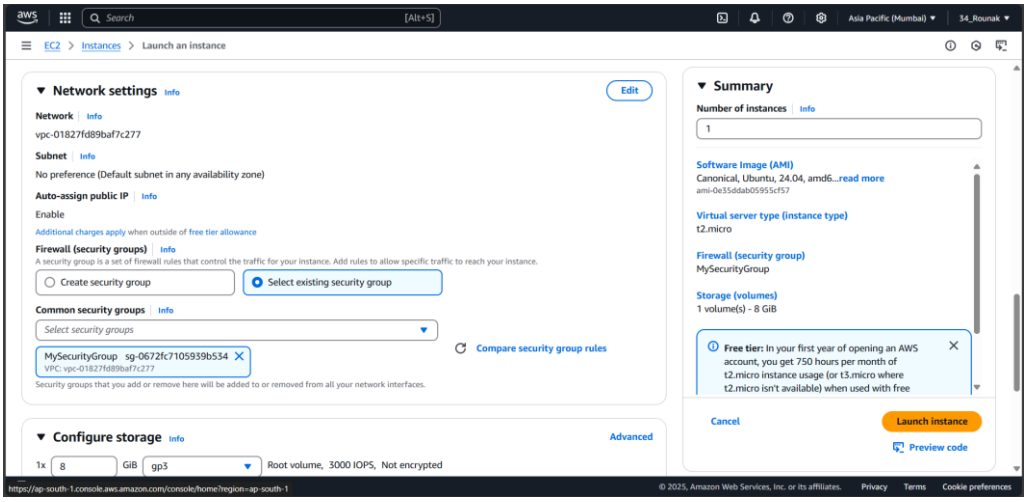




Step 4: In the “Key pair (login)” section, click on **Create new key pair**. Give a name, ensure that the key pair type is set to **RSA** and select **.pem** as the private key file format. Click on **“Create key pair”**.



Step 5: In the **“Network Settings”** section, ensure that the **"Select existing security group"** option is chosen. Click on “Launch instance”. Find and select the security group you just created in the previous steps. This will associate your new EC2 instance with the custom security rules you defined.

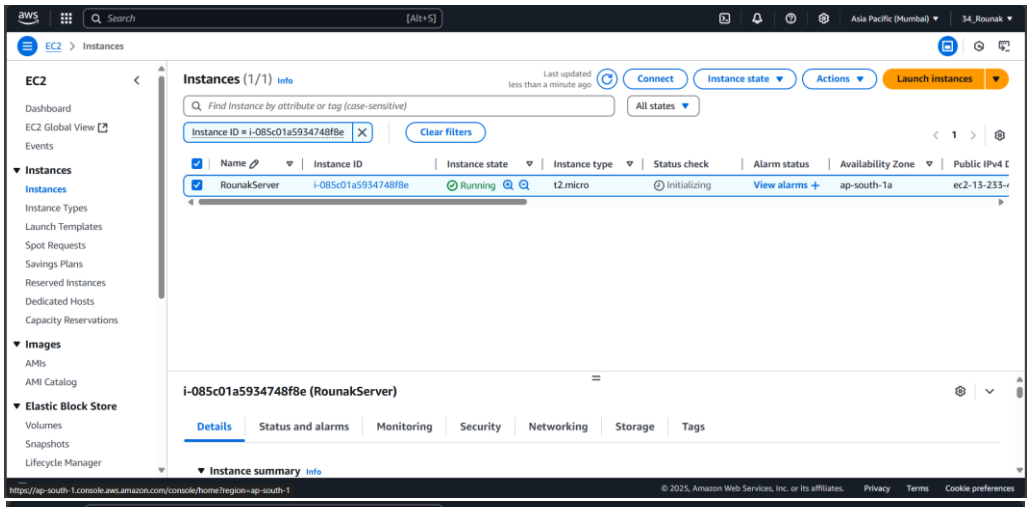


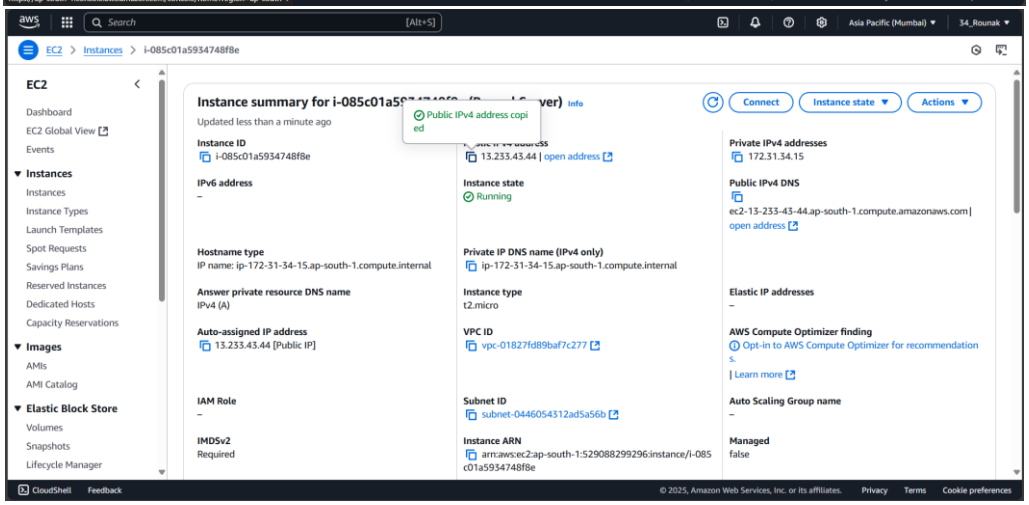
Step 6: Click on **"Advanced Details"**. Scroll Down to **"User Data"**. This is a text area where you can enter commands that will be executed automatically with root privileges when the instance is launched for the very first time. In the text box, enter the following commands sequentially:

```
#!/bin/bash
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
sudo apt-get install -y git
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs
git clone https://github.com/RounakS33/SDITO_Assignment.git
cd SDITO_Assignment
npm install
node index.js
```

Click on **“Launch Instance”**.

Step 7: Once the instance is launched, you should see it listed under **Instances**. Click on the **Instance ID** to view more details. In the EC2 details, click on the **Public IPv4 address** and copy it.





Step 8: Open a new tab in your web browser. In the address bar, enter the **Public IPv4 address** of your EC2 instance, followed by a colon (:) and the port number your application is listening on (in our case, http://13.233.43.44:4000). Press Enter. You should now see your deployed project running in the browser.

Conclusion:

In conclusion, this assignment successfully demonstrated the process of deploying a web project from GitHub to an EC2 instance on AWS by creating a custom security group and leveraging user data for automated setup. We first established a dedicated security group with specific inbound rules for SSH, HTTP, HTTPS, and our application's port (4000), ensuring controlled access to the instance. Subsequently, by utilizing user data during instance creation, we automated the installation of necessary software (Nginx, Node.js), cloned the project repository from GitHub, installed dependencies, and started the application server, all without manual intervention after instance launch. This approach highlights the efficiency and automation capabilities of AWS, streamlining the deployment workflow and reducing the time required to get a project up and running on the cloud. This exercise provides a valuable understanding of how to enhance security and automate deployment processes on AWS.