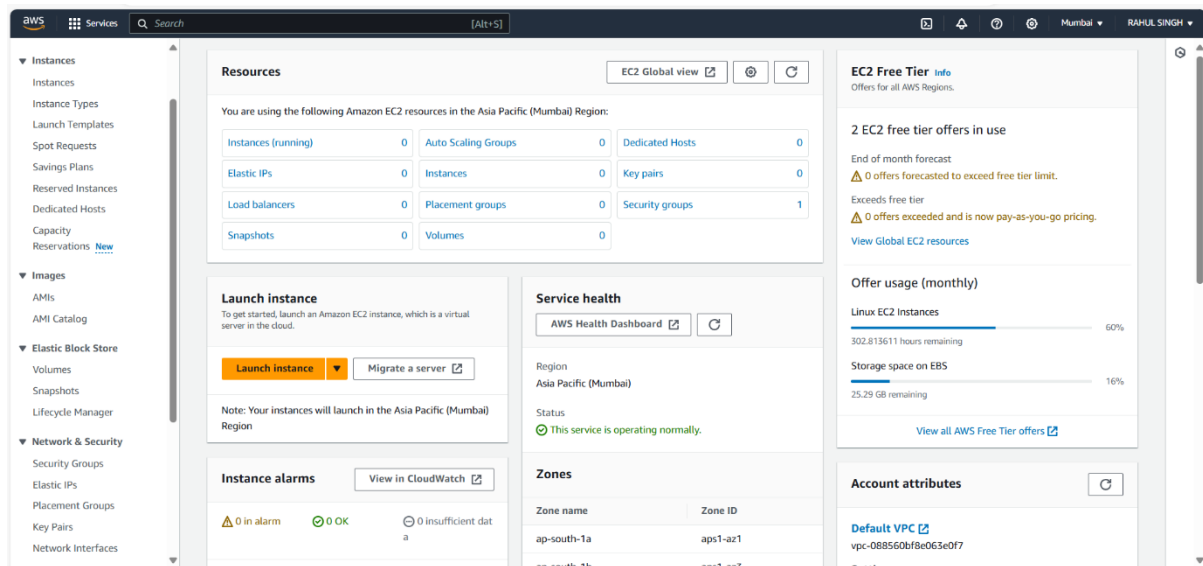


ASSIGNMENT – 10

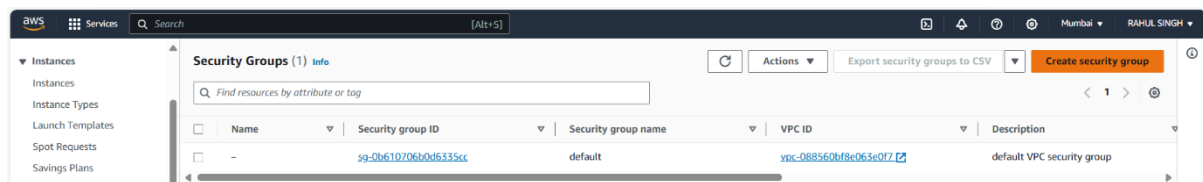
Problem Statement: Deploy project from GitHub to EC2 by creating new security group and user data.

Procedure:

1. Sign in to your AWS account.
2. Go to your EC2 dashboard
3. Scroll down and Click on Security Groups option on the left side nav bar under Network & Security option.



4. Select all the Security Groups other than the one named “default”.
5. Then Click on the Actions button.
6. Scroll-Down the dropdown list until you find the “delete all security groups” option. Click on it.
7. Now only the “default” security group remains and we keep it that way.
8. Now click on the “Create Security Group” button.



9. Now start by giving a name to the security group and giving its description (anything).

Let the VPC remain unchanged.

Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)

Name cannot be edited after creation.

Description [Info](#)

VPC [Info](#)

10. Next, we will add Inbound Rules. Start adding by clicking the Add rule button. These include:

- a) SSH
- b) HTTP
- c) HTTPS
- d) Custom TCP

Inbound rules [Info](#)

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
SSH	TCP	22	Anywhere... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>
HTTP	TCP	80	Anywhere... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>
HTTPS	TCP	443	Anywhere... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>
Custom TCP	TCP	4000	Anywhere... <input type="text" value="0.0.0.0/0"/>	<input type="text"/>

The last one with custom TCP has a specific port range that we require to connect to our project. It has been

specified in our index.js file (refer Ass9).

Now the final Inbound Rules section should look like this.

11. Next outbound rules and all other sections remain unchanged. Now Click on the create security group button.

12. Now go back to the security groups list and click on the security group ID of the newly created Security Group.

After clicking we can view the inbound rules that we added during its creation.

Outbound rules [Info](#)

Type [Info](#) Protocol [Info](#) Port range [Info](#) Destination [Info](#) Description - optional [Info](#)

All traffic All All Anywhere... 0.0.0.0/0 0.0.0.0/0 X Delete

Add rule

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags

Cancel Create security group

Security group (sg-0e573ffa9740e5c1d | mysec1) was created successfully

Security Groups (2) [Info](#) Actions Export security groups to CSV Create security group

Find resources by attribute or tag

Name	Security group ID	Security group name	VPC ID	Description
-	sg-0b610706b0d6335cc	default	ypc-088560bf8e063e0f7	default VPC security group
-	sg-0e573ffa9740e5c1d	mysec1	ypc-088560bf8e063e0f7	mysec1

Inbound rules (4) Manage tags Edit inbound rules

Search

Name	Security group rule...	IP version	Type	Protocol	Port range	Source
-	sgr-0fa0766ed75456f70	IPv4	Custom TCP	TCP	4000	0.0.0.0/0
-	sgr-065642b5afe0b875b	IPv4	HTTP	TCP	80	0.0.0.0/0
-	sgr-026b9e59cfb03e536	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-09558f9a81aa3b337	IPv4	HTTPS	TCP	443	0.0.0.0/0

13. Now we go to the instances section from the left side nav bar.

14. Now we Create a new EC2 instance. Click on the Launch Instance button.

Instances [Info](#) Connect Instance state Actions Launch instances

Find instance by attribute or tag (case-sensitive) All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
No instances							

You do not have any instances in this region

Launch instances

Now,

a) Give the name

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name rahulserver1 Add additional tags

Summary

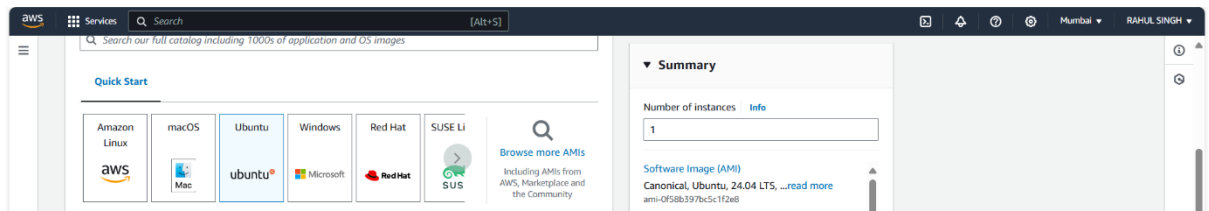
Number of instances 1

Software Image (AMI)
Canonical, Ubuntu, 24.04 LTS, ...read more
ami-0f58b3978c5c1f2e8

Virtual server type (instance type)
t2.micro

Firewall (security group)

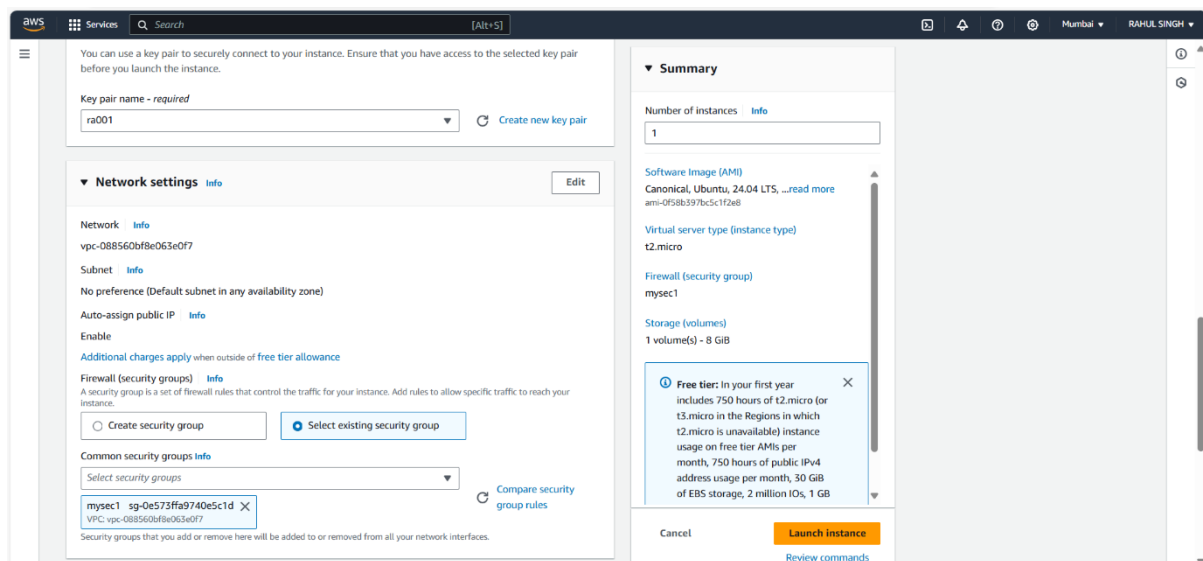
b) Select Ubuntu as OS.



- c) Select a keypair or generate a new one if none is available.
- d) Then under Network settings select the Select Existing Security Group option.

e) Now under the security groups dropdown menu select the one we just created.

It should look like this.....



f) Now scroll down and click on the Advanced Details option.

g) Now again scroll-down to the newly appeared sub-sections until you find User Data section.

h) Write the following commands in the given box. Remember this user data is given to execute the given

commands once the server starts. So essentially, we can provide all commands that we entered in our

Assignment 9 previously and execute them without connecting to our server itself!! They will be executed

sequentially.

requests. Applications or agents that use V1 for instance metadata access will break.

Metadata response hop limit [Info](#)

2

Allow tags in metadata [Info](#)

Select

User data - optional [Info](#)

Upload a file with your user data or enter it in the field.

[Choose file](#)

```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -
apt-get install -y nodejs
https://github.com/Rahulsinghrajp/ass8.git
cd ass8
npm install
node index.js
```

☐ User data has already been base64 encoded

Summary

Number of instances [Info](#)

1

Software Image (AMI)

Canonical, Ubuntu, 24.04 LTS, ...[read more](#)

ami-0f58b397bc5c1f2e8

Virtual server type (instance type)

t2.micro

Firewall (security group)

mysec1

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB

[Cancel](#) [Launch Instance](#) [Review commands](#)

Now, here is a caveat. We have created a private repository in GitHub. So, whenever we run the git clone

command it asks for our username and password. Hence this cannot be executed directly through our User

Data instructions. We have to connect manually and enter all commands starting from the git clone command.

i)

Now we click on the launch instance button.

15. Now we Click on the 'Instance Id' link of our newly created server in our Instances list.

16. Now click on the connect button

17. Again, click on the connect button

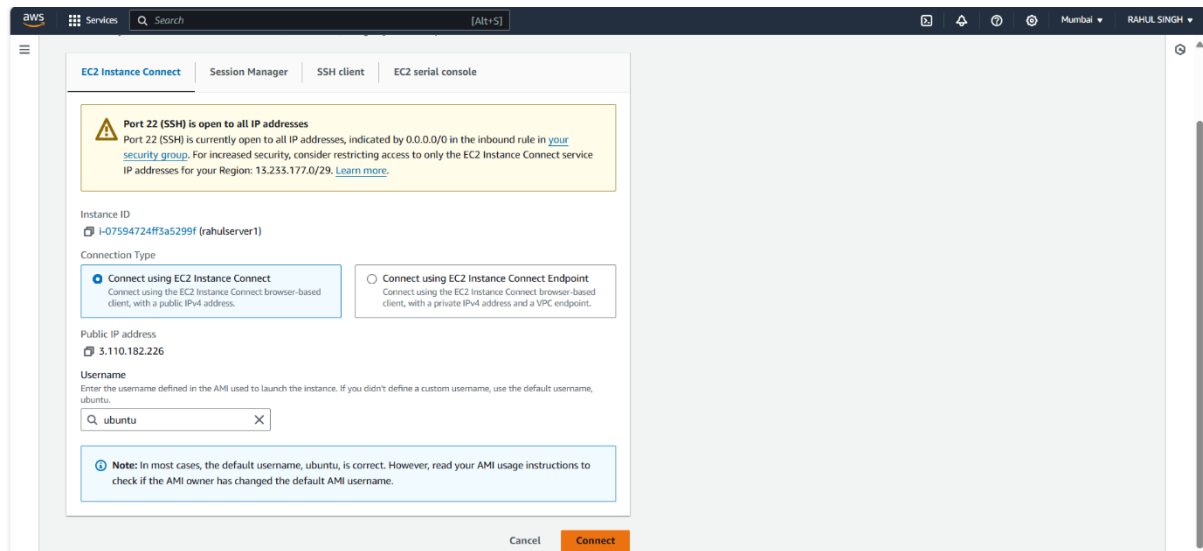
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 address
<input type="checkbox"/> rahuserver1	i-07594724ff3a5299f	Running	t2.micro	Initializing	View alarms	ap-south-1a	ec2-3-110-182-226.ap...	3.110.182.226

Instance summary for i-07594724ff3a5299f (rahuserver1) [Info](#)

Updated less than a minute ago

[Connect](#) [Instance state](#) [Actions](#)

Instance ID	Public IPv4 address	Private IPv4 addresses
i-07594724ff3a5299f (rahuserver1)	3.110.182.226 open address	172.31.40.47



18. After this a new Tab will open with a Bash Terminal that is of our remote EC2 server!

Here we can type all our required commands that we used to type in a similar terminal by connecting to our

remote server through our Bitwise SSH client software in our previous assignments.

19. Now type the following commands in the terminal:-

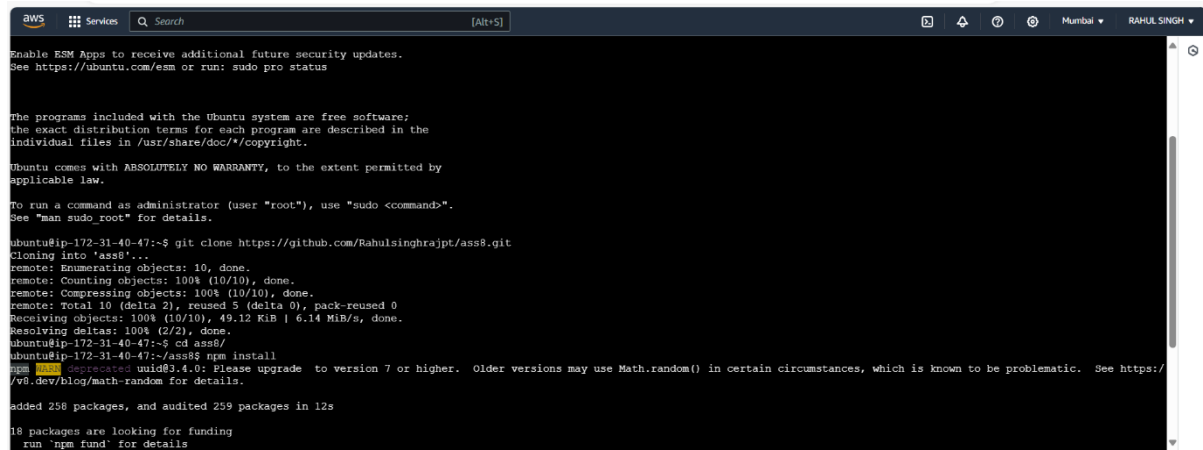
git clone https://github.com/..... //Your GitHub Repository URL

Give your Username of GitHub when asked.

Give your account Token when your Password is asked.

cd YourRepositoryname/

npm install

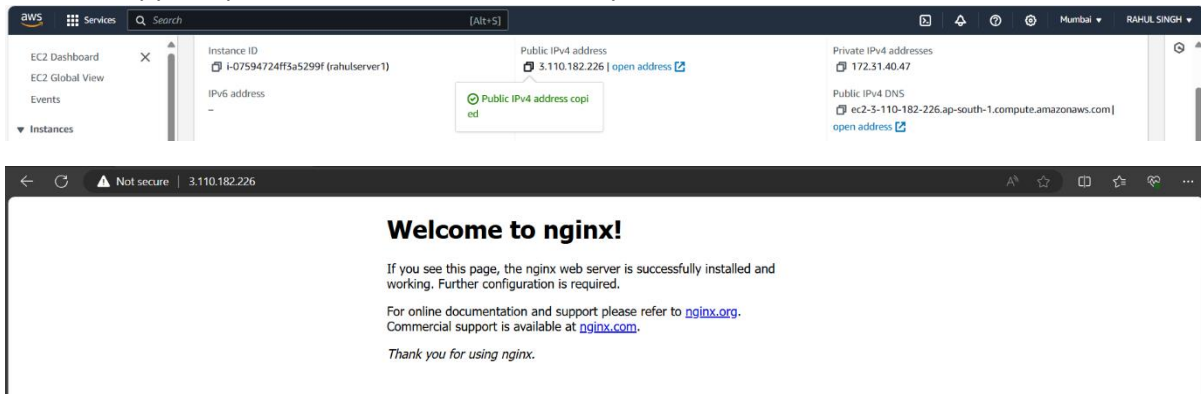


node index.js

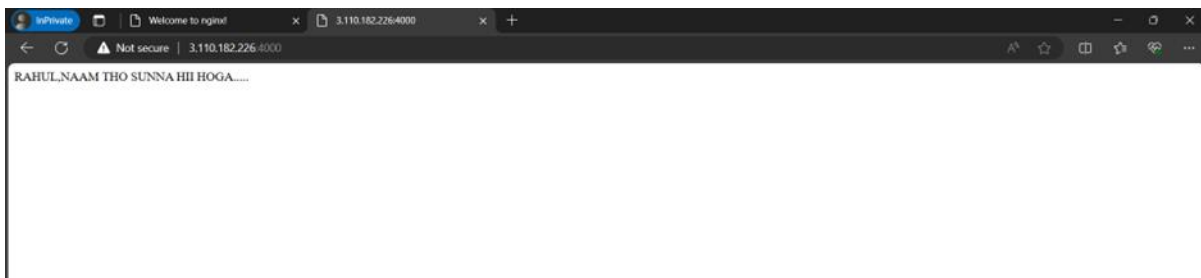
```
aws
Services
Search
[Alt+S]
Mumbai
RAHUL SINGH

ubuntu@ip-172-31-40-47:~$ git clone https://github.com/Rahulsinghrajpt/ass8.git
Cloning into 'ass8'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 10 (delta 2), reused 5 (delta 0), pack-reused 0
Receiving objects: 100% (10/10), 49.12 KiB | 6.14 MiB/s, done.
Resolving deltas: 100% (2/2), done.
ubuntu@ip-172-31-40-47:~$ cd ass8/
ubuntu@ip-172-31-40-47:~/ass8$ npm install
npm WARN deprecated uuid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
added 258 packages, and audited 259 packages in 12s
18 packages are looking for funding
  run `npm fund` for details
12 vulnerabilities (10 moderate, 2 critical)
To address all issues, run:
  npm audit fix
Run `npm audit` for details.
npm notice New major version of npm available! 8.19.4 -> 10.8.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.8.0
npm notice Run `npm install -g npm@10.8.0` to update!
npm notice
ubuntu@ip-172-31-40-47:~/ass8$ node index.js
Started server
```

20. Now copy and paste the Public IPv4 address of your EC2 instance in another browser.



21. Now append the port no. 4000 (for our case) to the IP address in the browser with a “.” sign.



We have successfully Deployed a project from GitHub to EC2 by creating a new Security group and User Data.