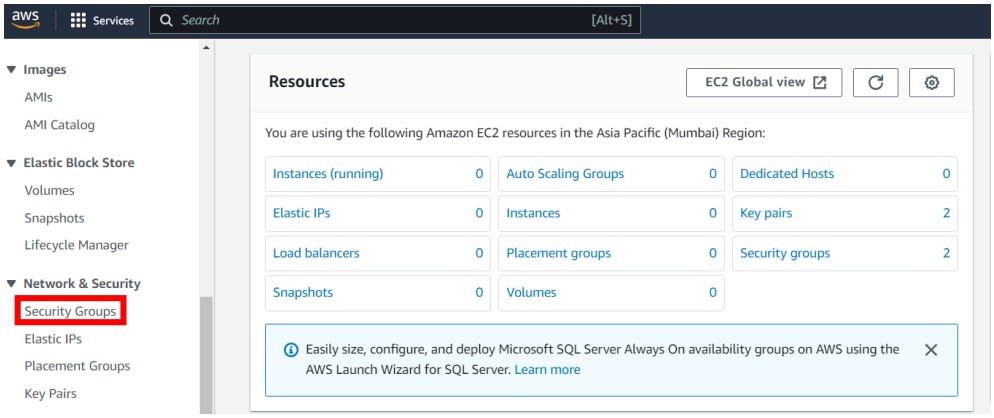
ASSIGNMENT – 10

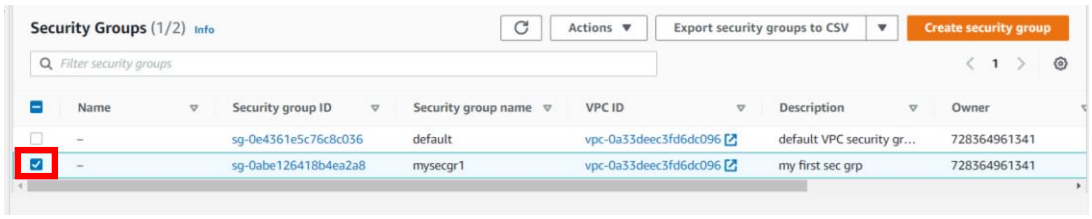
Problem Statement: Deploy project from GitHub to EC2 by creating new security group and user data.

Procedure:

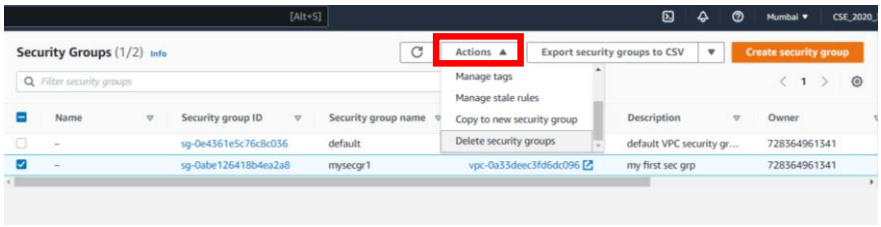
1. **Sign in** to your **AWS** account.
2. **Go to** your **EC2** dashboard
3. **Scroll down** and **Click** on **Security Groups** option on the left side nav bar under Network & Security option.



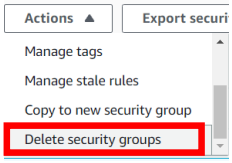
4. **Select** all **the Security Groups other than** the **one** named "**default**".



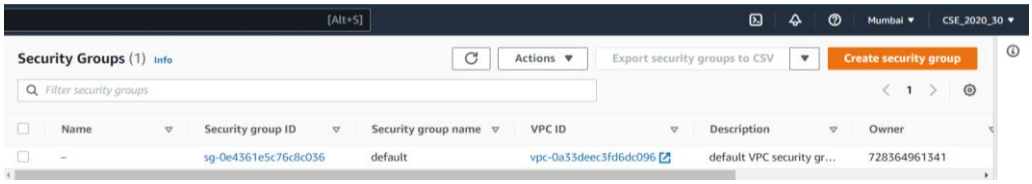
5. Then **Click** on the **Actions** button.



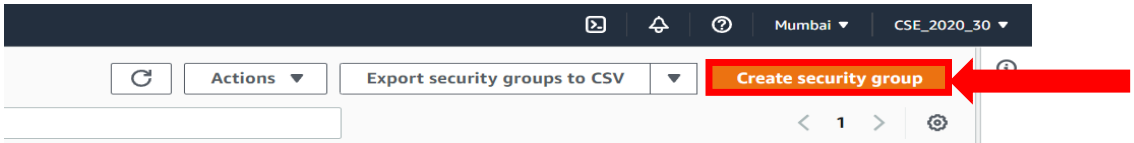
6. **Scroll-Down** the **dropdown** list until you find the "**delete all security groups**" option. **Click** on it.



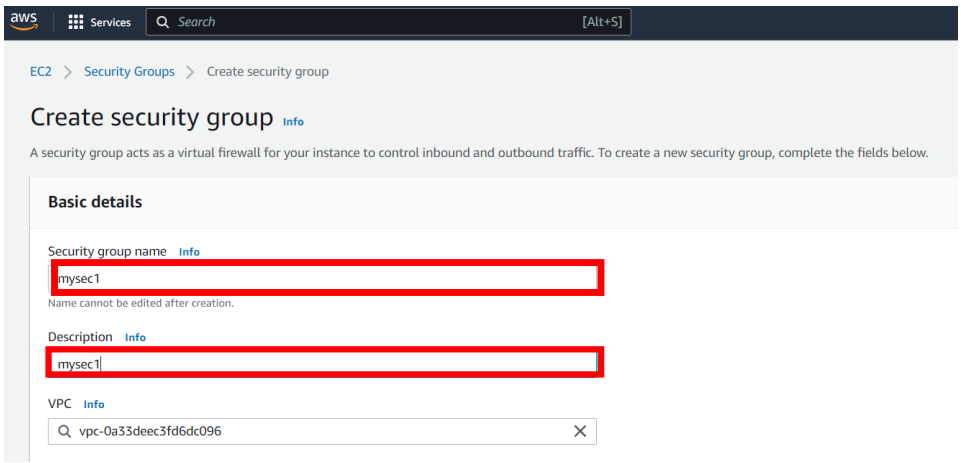
7. Now only the "**default**" **security group** remains and we keep it that way.



8. Now click on the "**Create Security Group**" button.

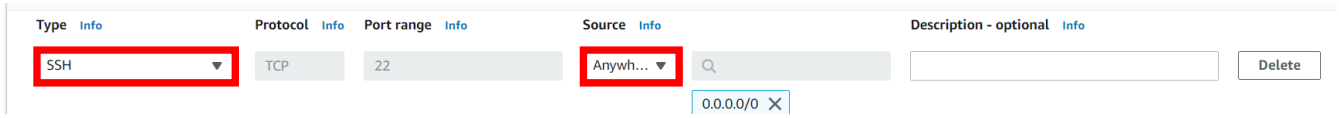


9. Now start by giving a **name** to the **security group** and **giving its description** (anything).
   Let the **VPC remain unchanged**.

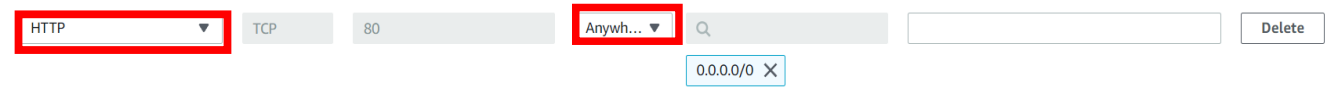


10. Next, we will **add Inbound Rules.** Start **adding** by **clicking the <u>Add rule</u> button**. These include:

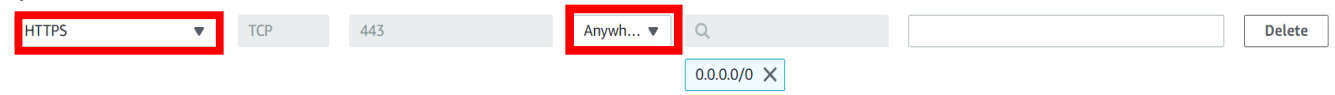
a) SSH



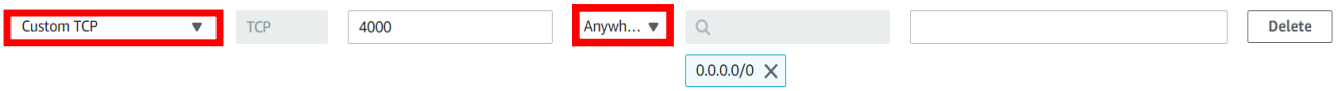
b) HTTP



c) HTTPS

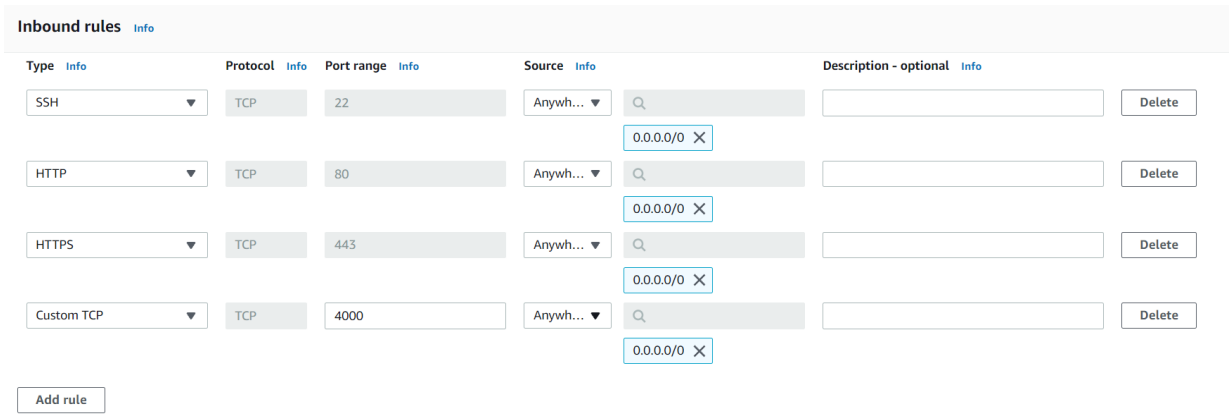


d) Custom TCP

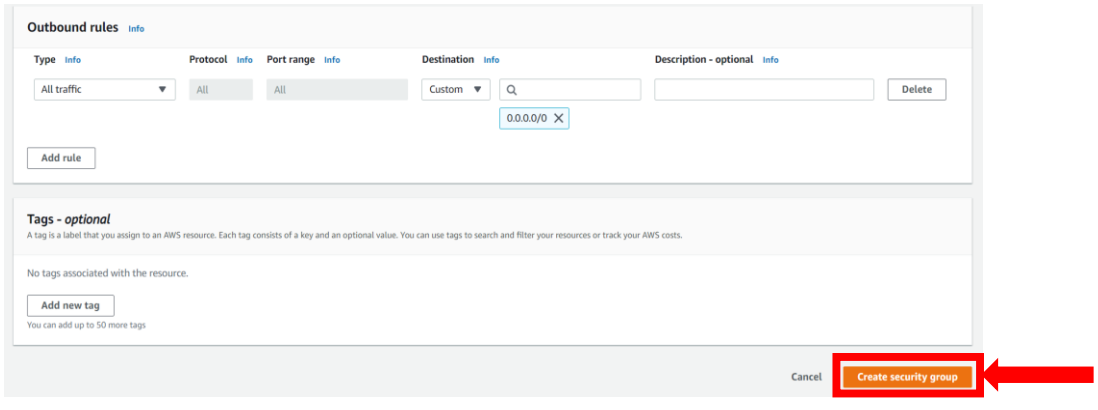


The **last one with custom TCP** has a **specific port range** that we require to connect to our project. It has been specified in our index.js file (refer Ass9).

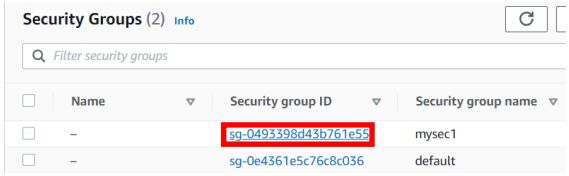
Now the final Inbound Rules section should look like this.

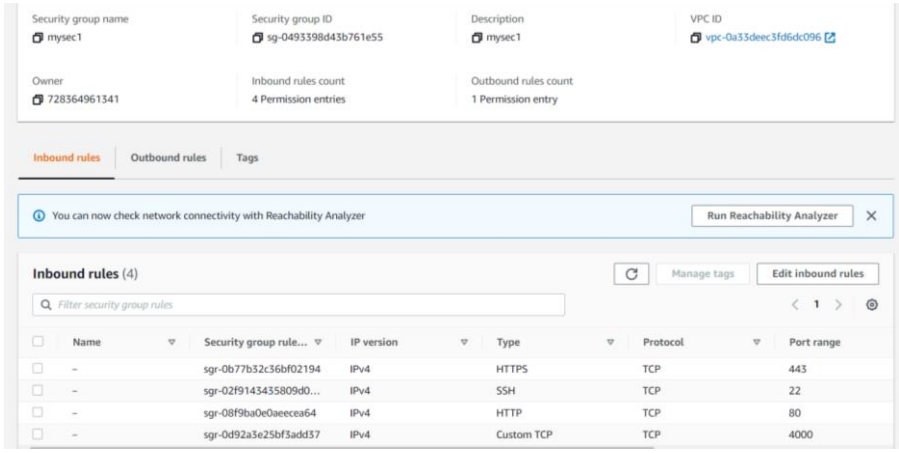


11. **Next outbound rules and all other sections remain unchanged. Now Click on the create security group button**.



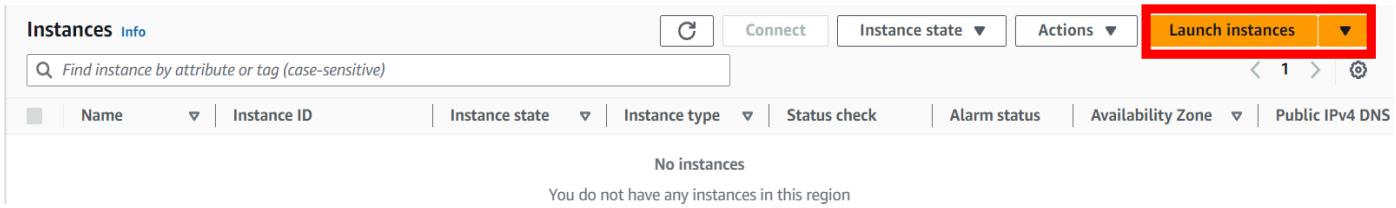
12. Now **go back** to the **security groups list** and **click** on the **security group ID** of the newly created Security Group.





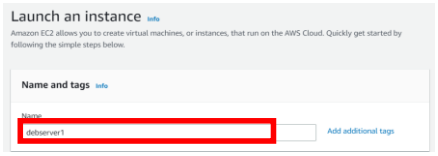
After clicking we can view the inbound rules that we added during its creation.

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**.

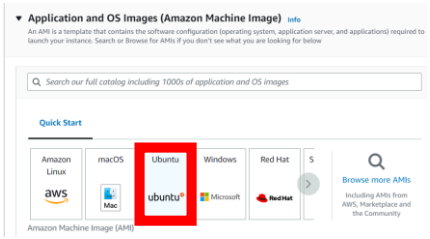


Now,

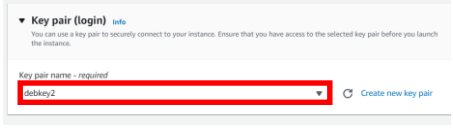
a) Give the name



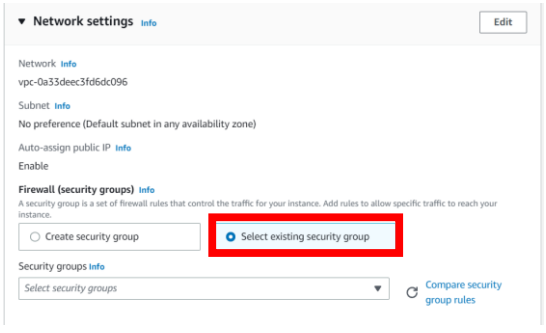
b) Select Ubuntu as OS.



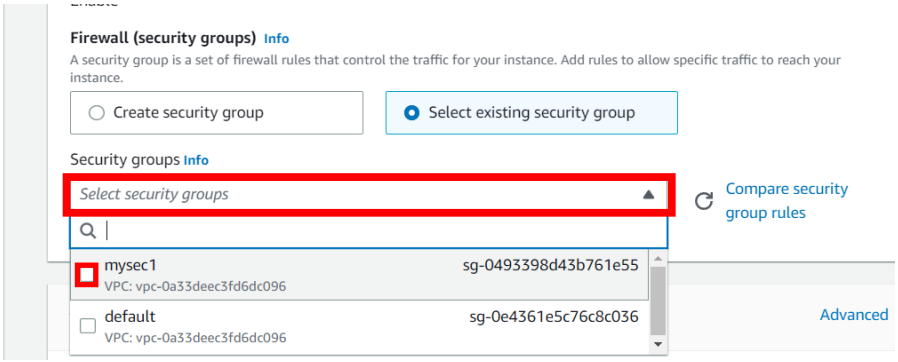
c) Select a keypair or generate a new one if none is available.



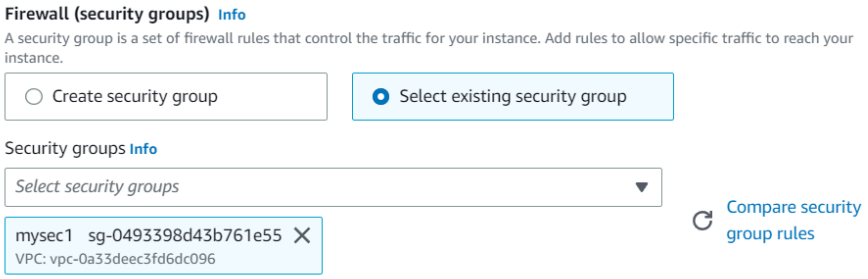
d) Then under Network settings select the Select Existing Security Group option.



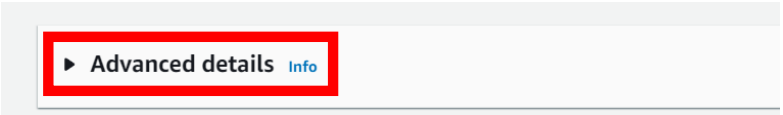
e) **Now under the security groups dropdown menu select the one we just created.**



It should look like this…..



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



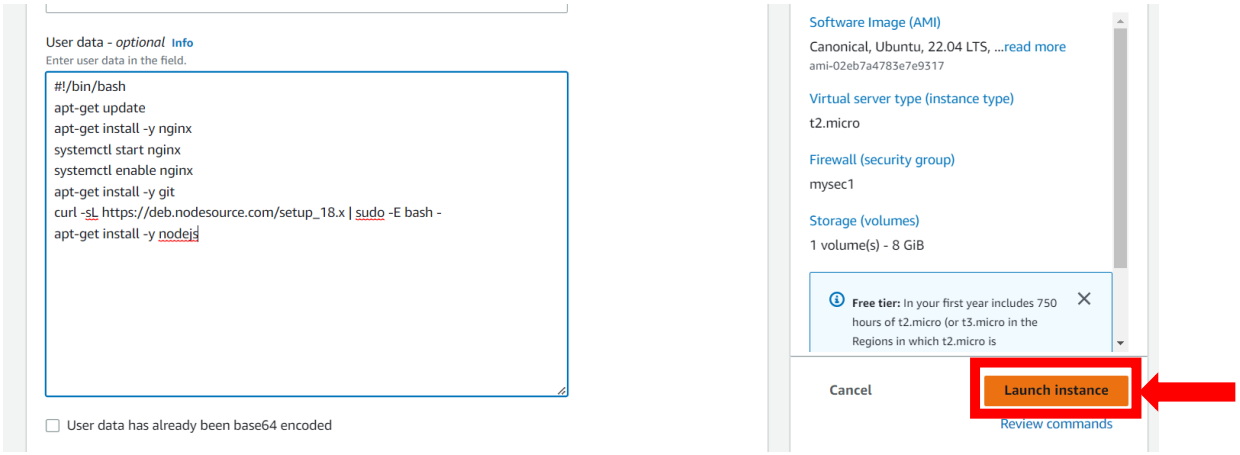
g) **Now again scroll-down to the newly appeared sub-sections until you find User Data section.**

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**.**

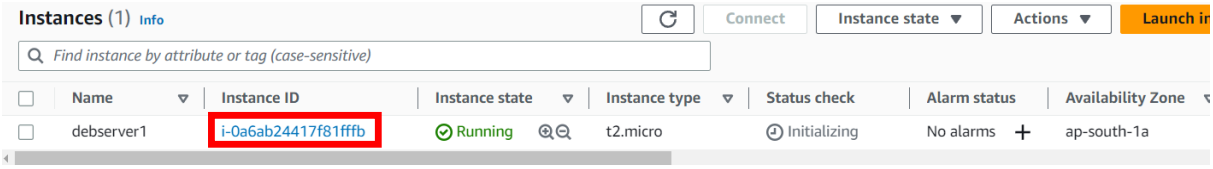
```
#!/bin/bash
apt-get update
apt-get install -y nginx
systemctl start nginx
systemctl enable nginx
apt-get install -y git
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
apt-get install -y nodejs
```

Now, here is a caveat. **We have 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.**

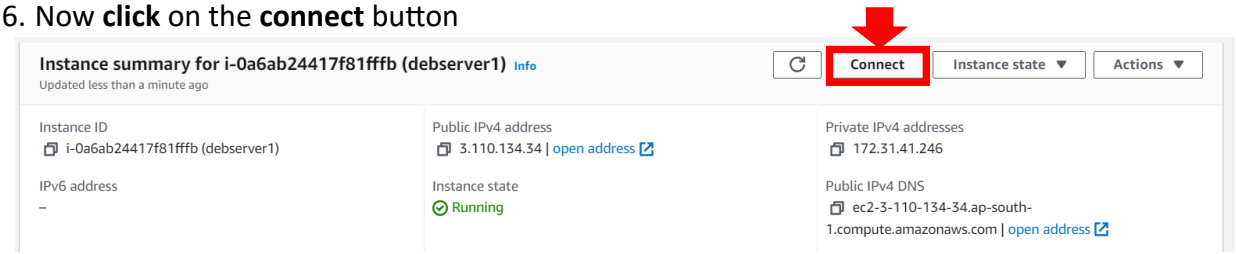
i) Now we **click** on the **launch instance button**.



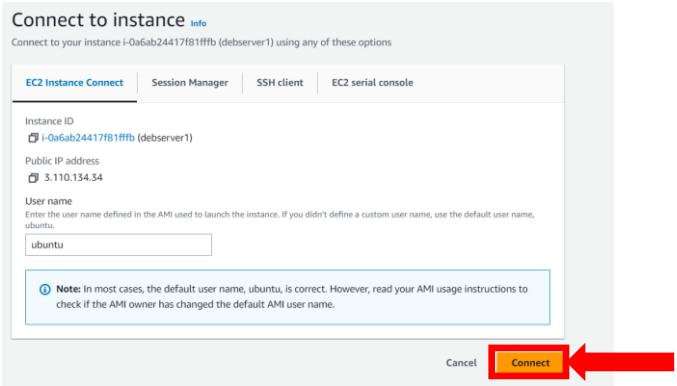
15. Now we **Click** on the **'Instance Id' link** of our newly created server in our Instances list.



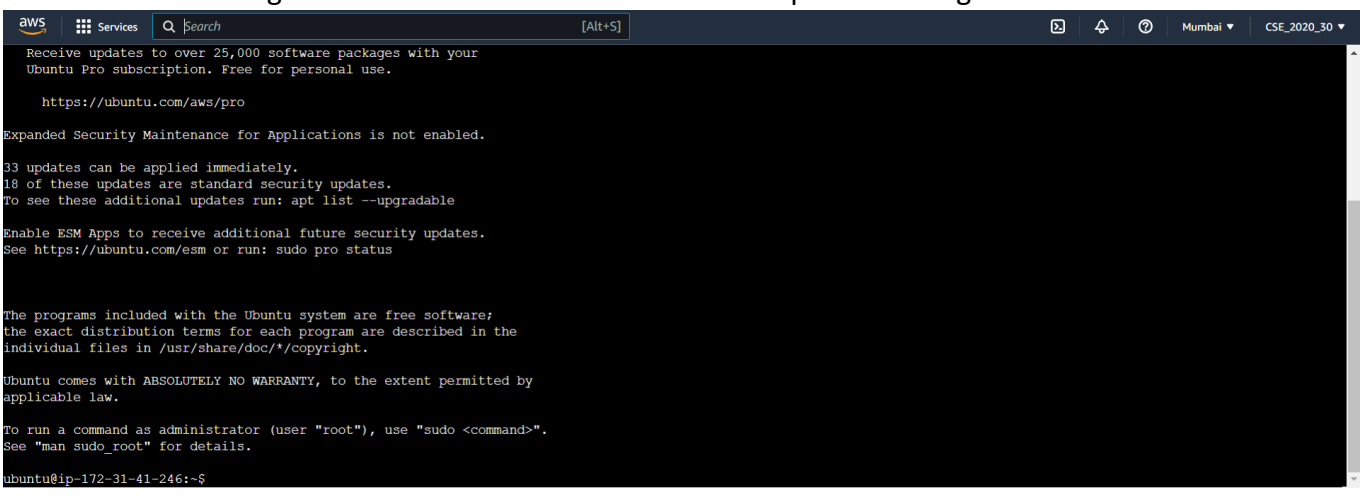
16. Now **click** on the **connect** button



17. **Again**, **click** on the **connect** button



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 Bitvise 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.)

```
ubuntu@ip-172-31-41-246:~$ git clone https://github.com/DebrupPramanik/myRepoV1.git
Cloning into 'myRepoV1'...
Username for 'https://github.com': DebrupPramanik
Password for 'https://DebrupPramanik@github.com':
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 15 (delta 6), reused 4 (delta 0), pack-reused 0
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (6/6), done.
```

➔ **cd YourRepositoryname/**

```
ubuntu@ip-172-31-41-246:~$ cd myRepoV1/
ubuntu@ip-172-31-41-246:~/myRepoV1$
```

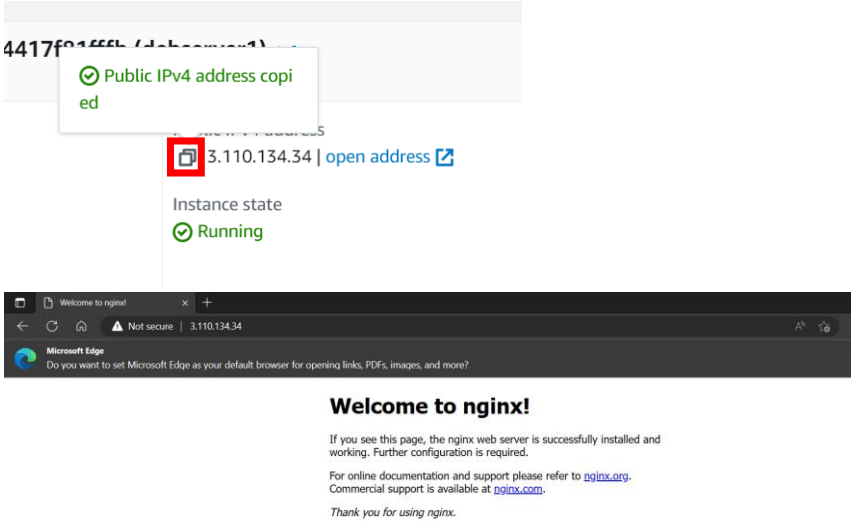
➔ **npm install**

```
ubuntu@ip-172-31-41-246:~/myRepoV1$ npm install
npm WARN deprecated uuid@3.4.0: Please upgrade  to version 7 or higher.  Se
ee https://v8.dev/blog/math-random for details.

added 258 packages, and audited 259 packages in 15s

18 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
npm notice
npm notice New minor version of npm available! 9.5.1 -> 9.6.5
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.6.5
npm notice Run npm install -g npm@9.6.5 to update!
npm notice
```

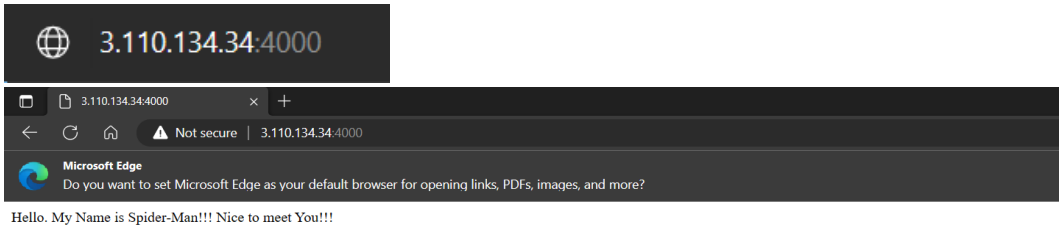
➔ **node index.js**

```
ubuntu@ip-172-31-41-246:~/myRepoV1$ 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.