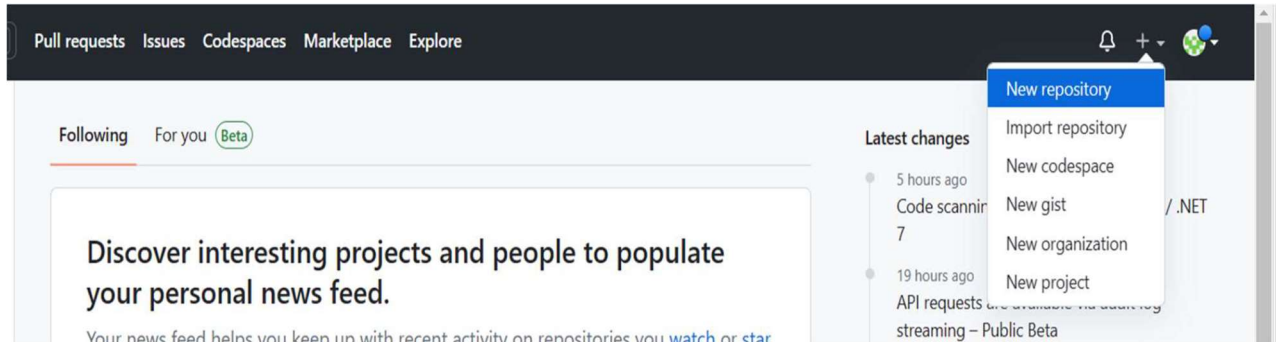


ASSIGNMENT 9

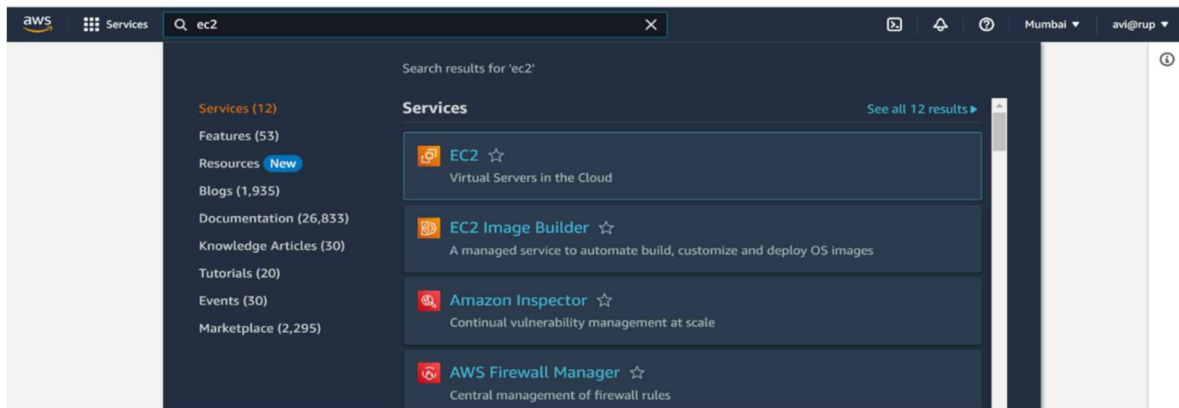
Problem Statement: Deploy a project from Github to EC2.

1. Sign in to your Github account.

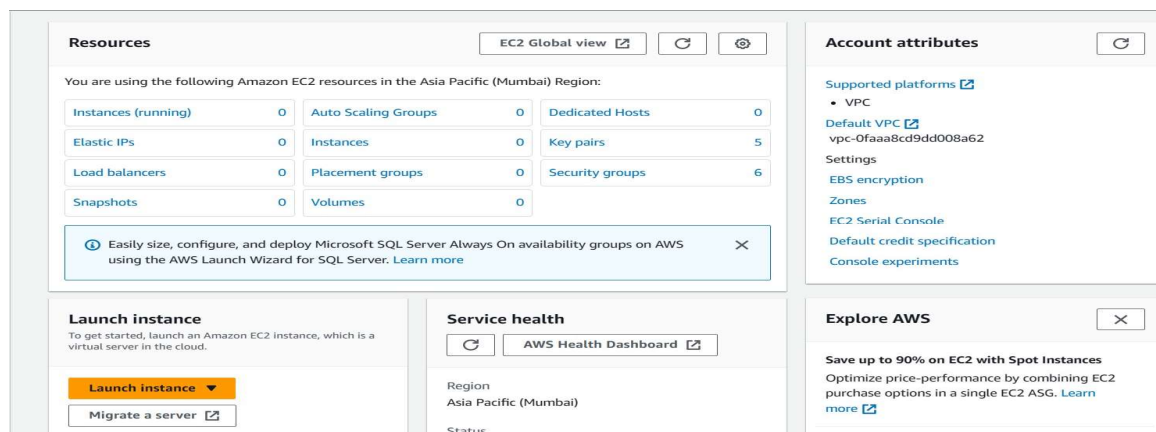


2. Create an EC2 instance in AWS.

a) Sign in to your AWS account and then go to EC2.



b) Next, Click on **Launch Instance**.



c) Write the instance name. Then, Select an OS. Here, we have selected **Ubuntu**.

EC2 > Instances > Launch an instance

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

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

▼ Summary

Number of instances [Info](#)

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...[read more](#)

ami-02eb7a4783e7e9317

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Cancel **Launch instance**

[Review commands](#)

d) We can see the Instance type set here is t2.micro

▼ Instance type [Info](#)

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory

On-Demand Linux pricing: 0.0124 USD per Hour

On-Demand Windows pricing: 0.017 USD per Hour

On-Demand RHEL pricing: 0.0724 USD per Hour

On-Demand SUSE pricing: 0.0124 USD per Hour

Free tier eligible

☒ All generations

[Compare instance types](#)

e) We have given a Key pair.

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

[Create new key pair](#)

f) Allow the SSH, HTTPS, HTTP traffic from the internet.

▼ Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-6' with the following rules:

- ☒ Allow SSH traffic from **Anywhere** (0.0.0.0/0)
- ☒ Allow HTTPS traffic from the internet
- ☒ Allow HTTP traffic from the internet

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

g) Now, Click on **Launch Instance**.

▼ Summary

Number of instances [Info](#)

Software Image (AMI)

Canonical, Ubuntu, 22.04 LTS, ...[read more](#)

ami-02eb7a4783e7e9317

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

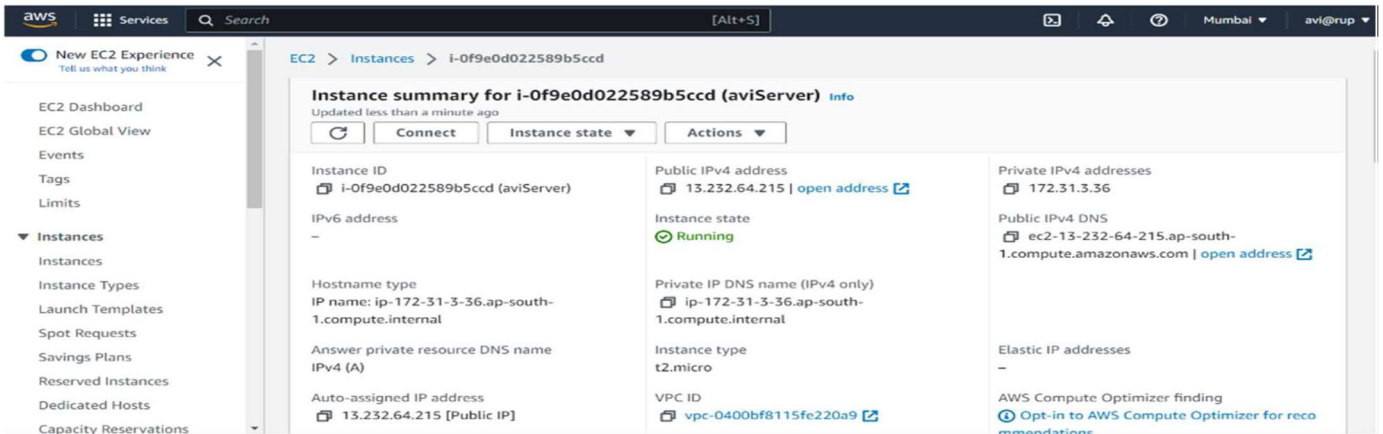
1 volume(s) - 8 GiB

☒ Free tier: In your first year includes 750

Cancel **Launch instance**

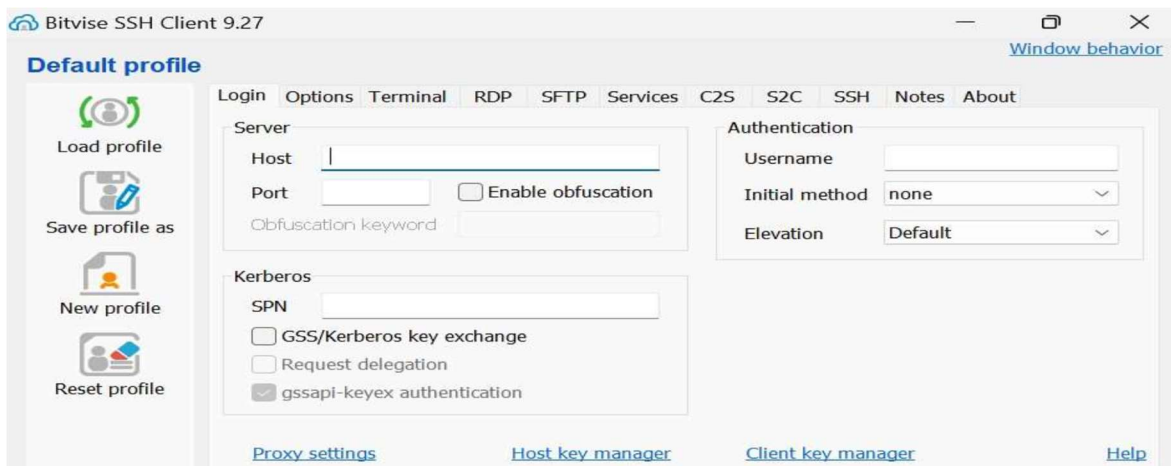
[Review commands](#)

h) Next, copy the *Public IPv4 address*.



3. Now open Bitvise SSH.

a) Within the Login tab, In the *host* field paste the Public IPv4 address which you have copied earlier. Set the Username as *ubuntu*, Initial method as *public key* and Client key as *Global 1*.



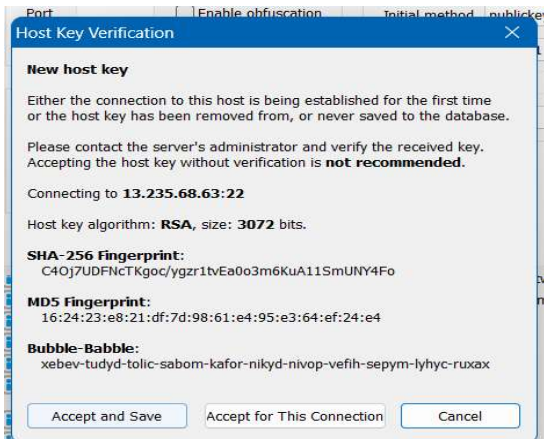
b) Now, Import the key pair.



c) Now, Click on **Log in**.



d) Click on *Accept and Save*.



e) Next, Open the **Terminal**.

4. In the *Bitvise SSH Terminal*, Type the following commands in the terminal one by one.

a) `pwd`

```
ubuntu@ip-172-31-0-51:~$ pwd
/home/ubuntu
```

b) `sudo apt-get update && sudo apt-get upgrade`

```
ubuntu@ip-172-31-0-51:~$ sudo apt-get update && sudo apt-get upgrade
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [108 kB]
Get:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [731 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [147 kB]
```

Continuing...

```
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.
ubuntu@ip-172-31-0-51:~$
```

c) `sudo apt-get install nginx`

```
ubuntu@ip-172-31-0-51:~$ sudo apt-get install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Continuing...

```
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.
ubuntu@ip-172-31-0-51:~$
```

d) `nginx -v`

```
ubuntu@ip-172-31-0-51:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
```


e) `curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -`

```
ubuntu@ip-172-31-0-51:~$ curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -  
## Installing the NodeSource Node.js 18.x repo...
```

Continuing...

f) `sudo apt install nodejs`

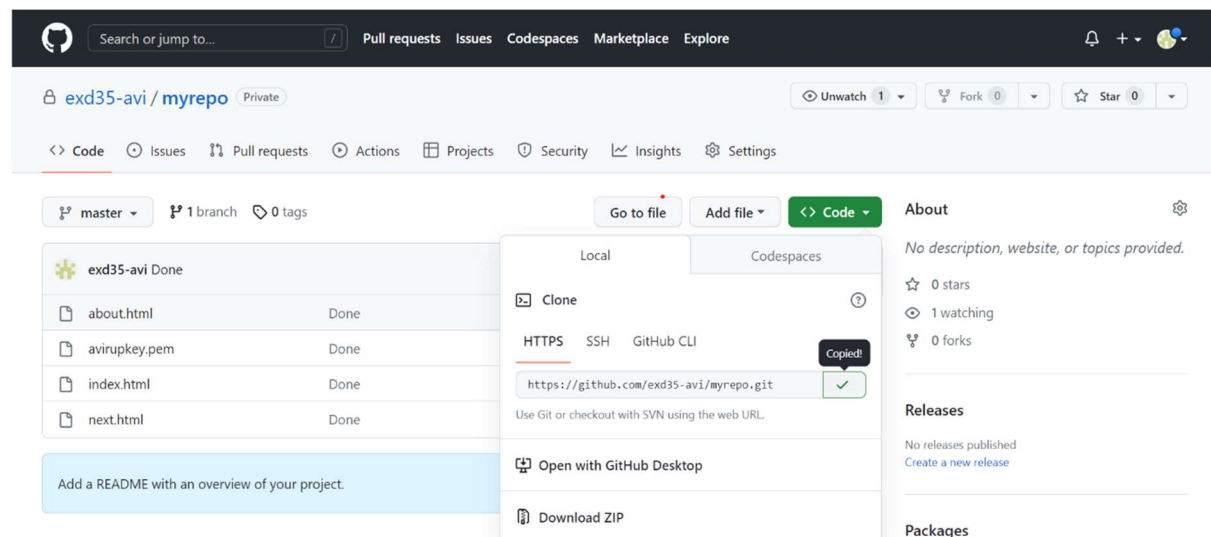
```
ubuntu@ip-172-31-0-51:~$ sudo apt install nodejs  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following NEW packages will be installed:  
  nodejs
```

Continuing...

g) `node -v`

```
ubuntu@ip-172-31-0-51:~$ node -v  
v18.15.0
```

h) Now, copy the HTTPS link of the Github Repository.



Again, go to Bitwise SSH terminal and type the following command to clone the repository in the EC2 server.

`git clone https://github.com/exd35-avi/myrepo.git`

Now, give the Username and Password(Token)

```
ubuntu@ip-172-31-42-96:~$ git clone https://github.com/exd35-avi/myrepo.git  
Cloning into 'myrepo'...  
Username for 'https://github.com': exd35-avi  
Password for 'https://exd35-avi@github.com':  
remote: Enumerating objects: 6, done.  
remote: Counting objects: 100% (6/6), done.  
remote: Compressing objects: 100% (5/5), done.  
remote: Total 6 (delta 1), reused 6 (delta 1), pack-reused 0  
Receiving objects: 100% (6/6), done.  
Resolving deltas: 100% (1/1), done.
```

i) `dir`

```
ubuntu@ip-172-31-42-96:~$ dir  
myrepo
```

j) `cd Awsproject2`

```
ubuntu@ip-172-31-42-96:~$ cd myrepo
```

k) `dir`

```
ubuntu@ip-172-31-42-96:~/myrepo$ ls
about.html  avirupkey.pem  index.html  next.html
```

l) npm install

```
ubuntu@ip-172-31-0-51:~/Awsproject2$ 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 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! 9.5.0 -> 9.6.4
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.6.4
npm notice Run npm install -g npm@9.6.4 to update!
npm notice
```

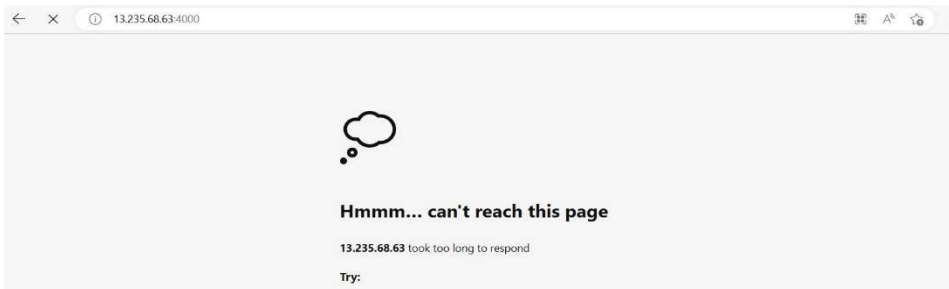
m) node index.js

```
ubuntu@ip-172-31-0-51:~/Awsproject2$ node index.js
Started server
```

n) Copy the Public IPv4 address and paste it in a browser.

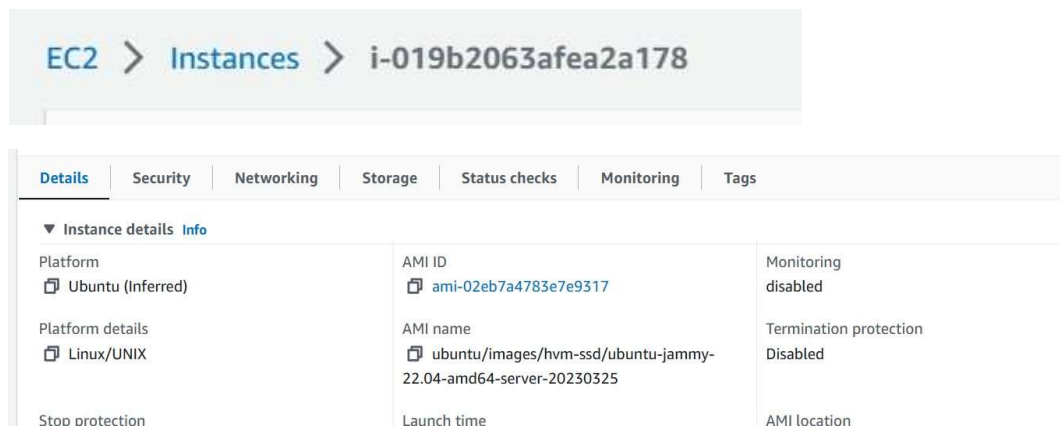


o) We cannot see our website in any port.

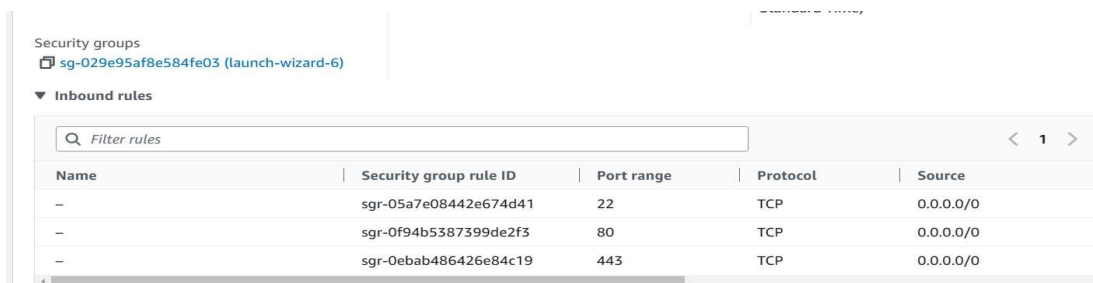


5. Now, to see our website in port 4000, we have to follow the steps.

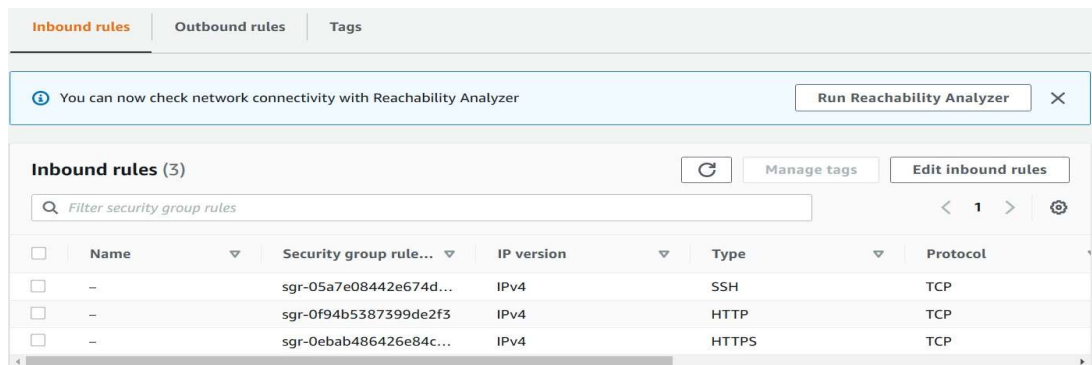
a) Within the instance you create, got to the Security tab.



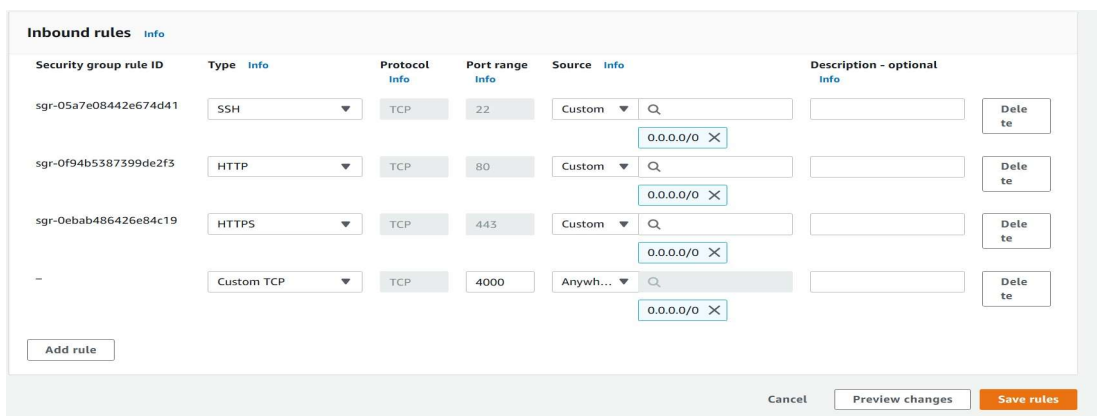
b) Next, Go to the Security groups.



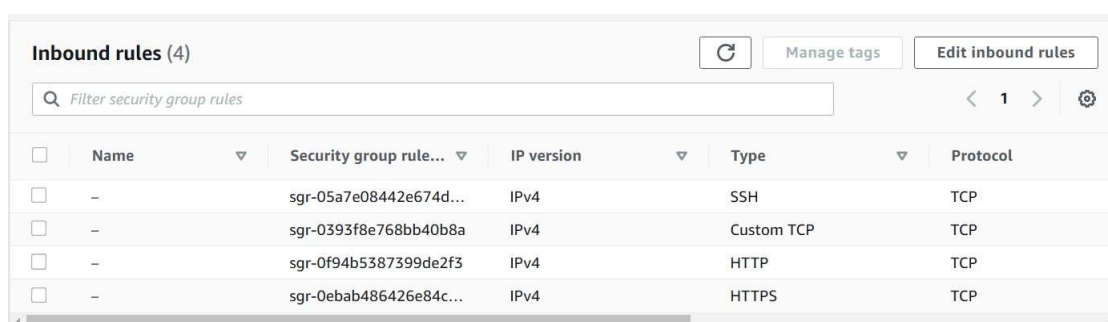
c) Next, Click on Edit inbound rules.



d) Now, add a custom TCP with port 4000 and source 0.0.0.0/0 and Save rules.



e) We can see that the new rule is added.



f) Now, add the port number 4000 after the Public IPv4 address you have in the browser URL.



We can see the Text “Hello”. Thus, it is working.

6. Making changes in the webpage.

a) Go to index.js file in your Github Repository. Then, Click on the pen icon.

exd35-avi Update index.js Latest commit e9df28e now History

2 contributors

11 lines (9 sloc) 179 Bytes

```
1 const express = require('express')
2 const app = express()
3
4 app.get('/', function (req, res) {
5   res.send('Hello')
6 })
7
8 app.listen(4000, () => {
9   console.log("Started server");
10 })
11 )
```

b) Edit the code.

exd35-avi / myrepo Private

Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Security Insights Settings

myrepo / index.js in main Cancel changes

Edit file Preview changes Spaces 2 No wrap

```
1 const express = require("express");
2 const app = express();
3
4 app.get("/", function (req, res) {
5   res.send("Hello world");
6 });
7
8 app.listen(4000, () => {
9   console.log("Started server");
10 });
11
```

c) Click **Commit changes**.

Commit changes

Update index.js

Add an optional extended description...

☒ Commit directly to the master branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

d) Now, go to the Bitwise terminal and run the following commands respectively.

- Stop the server first.

```
Started server
^C
ubuntu@ip-172-31-0-51:~/Awsproject2$
```

git pull and Give *username* and *password*.

```
ubuntu@ip-172-31-32-29:~/myrepo$ git pull
Username for 'https://github.com': exd35-avi
Password for 'https://exd35-avi@github.com':
```

You can see the change is done. i.e. 1 file changed, 1 insertion(+), 1 deletion(-)

```
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 662 bytes | 662.00 KiB/s, done.
From https://github.com/exd35-avi/myrepo
   f0abafe..4f19de6  main       -> origin/main
Updating f0abafe..4f19de6
Fast-forward
 index.js | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```


- Start the server again.

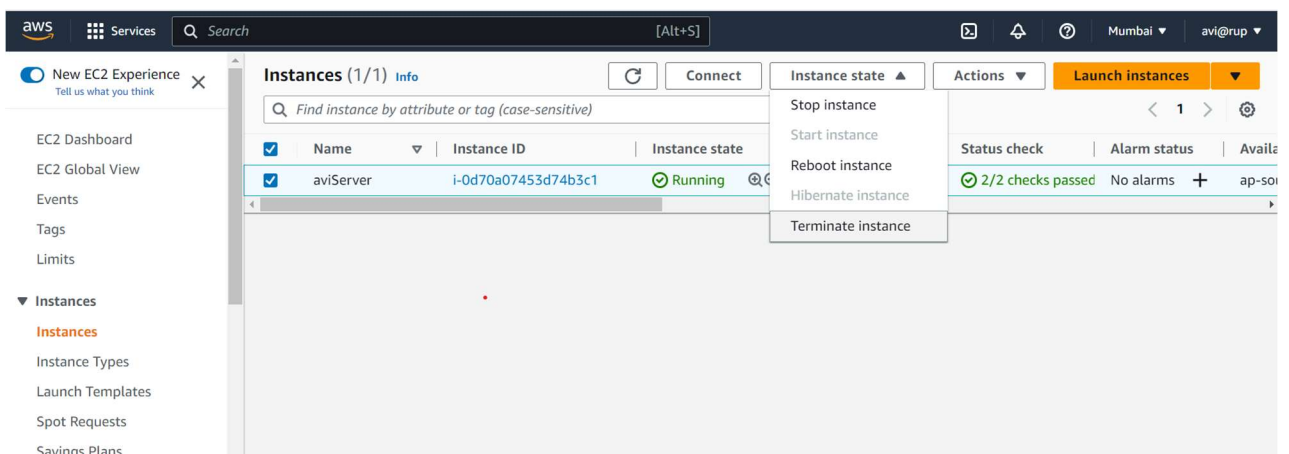
```
ubuntu@ip-172-31-32-29:~/myrepo$ node index.js
Started server
```

- Refresh the page and we can see the changes.

← → ↻ ⚠ Not secure | 43.205.233.253:4000

Hello world

7. Terminate the EC2 server.



The screenshot shows the AWS Management Console interface. On the left, there's a navigation menu with options like 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Tags', 'Limits', and 'Instances'. The main panel displays 'Instances (1/1) Info'. A table lists the instance 'aviServer' with ID 'i-0d70a07453d74b3c1' and state 'Running'. An 'Actions' dropdown menu is open, showing options: 'Stop instance', 'Start instance', 'Reboot instance', 'Hibernate instance', and 'Terminate instance'. On the right, there's a summary bar showing 'Status check: 2/2 checks passed', 'Alarm status: No alarms', and 'Availability: ap-southeast-1a'.

Next, Logout of **Bitwise SSH Client** by clicking *Abort*.