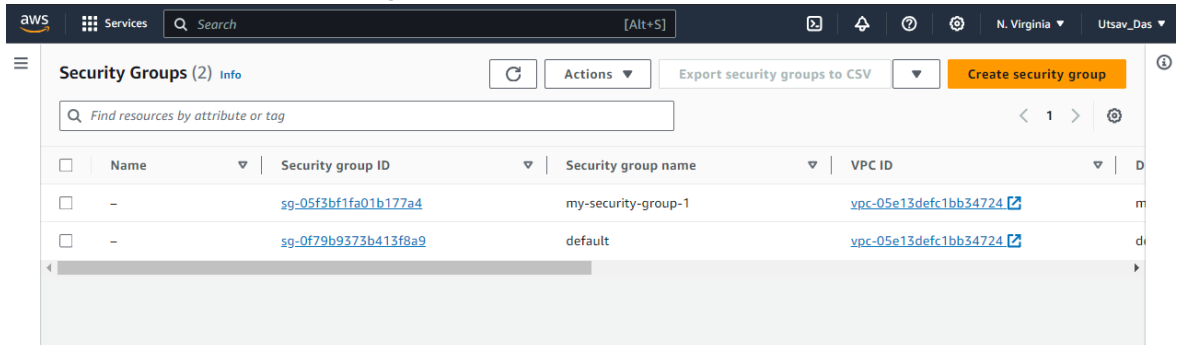


Assignment – 12

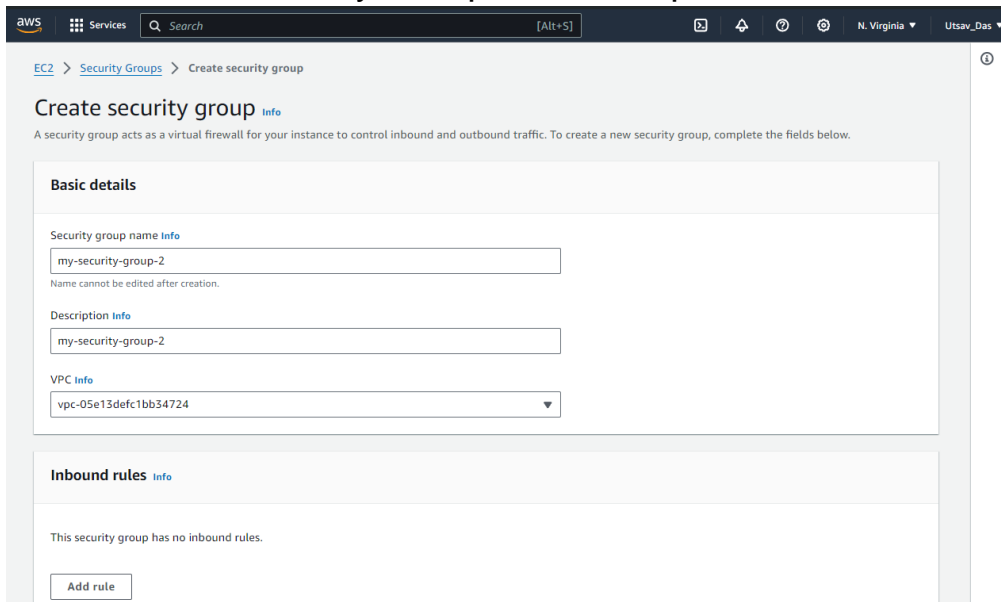
Problem Statement:

Deploy and run the project in AWS without using port.

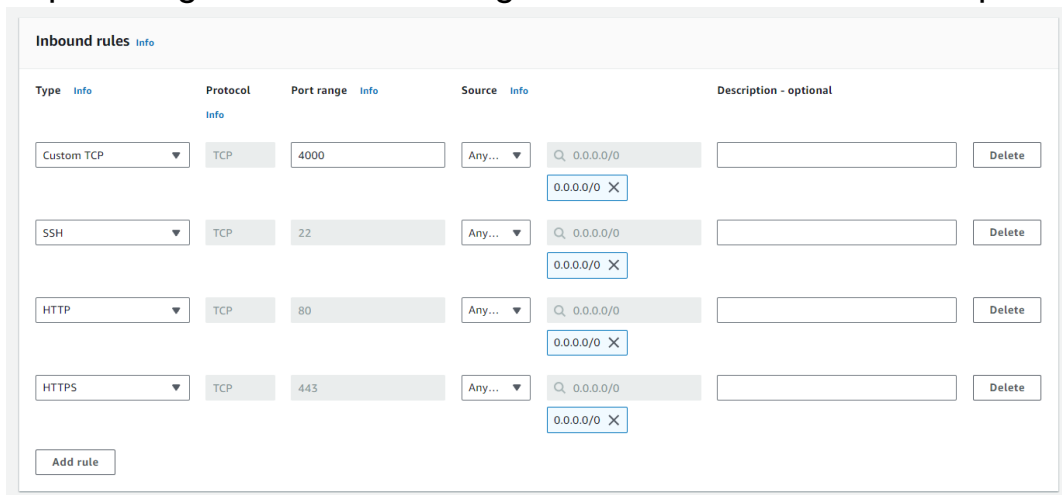
- 1) Go to EC2 -> Security groups and click on Create Security Group option.



- 2) Give name of Security Group and description.



- 3) In Inbound rules click on Add rule. Here, we add all 4 rules: Custom TCP, SSH, HTTP, HTTPS and in Source select 0.0.0.0/0
In port range of Custom TCP give 4000. Rest have default port number.



4) Go back to instance and click on Launch instance.

The screenshot shows the 'Resources' section of the AWS Management Console for the US East (N. Virginia) Region. It displays a grid of resource counts: Instances (running) 0, Elastic IPs 0, Load balancers 0, Snapshots 0, Auto Scaling Groups 0, Instances 0, Placement groups 0, Volumes 0, Dedicated Hosts 0, Key pairs 5, and Security groups 3. Below this, the 'Launch instance' section is visible, featuring a prominent orange 'Launch instance' button and a 'Migrate a server' link. A note states: 'Note: Your instances will launch in the US East (N. Virginia) Region'. To the right, the 'Service health' section shows the AWS Health Dashboard and indicates that the service is operating normally in the US East (N. Virginia) Region.

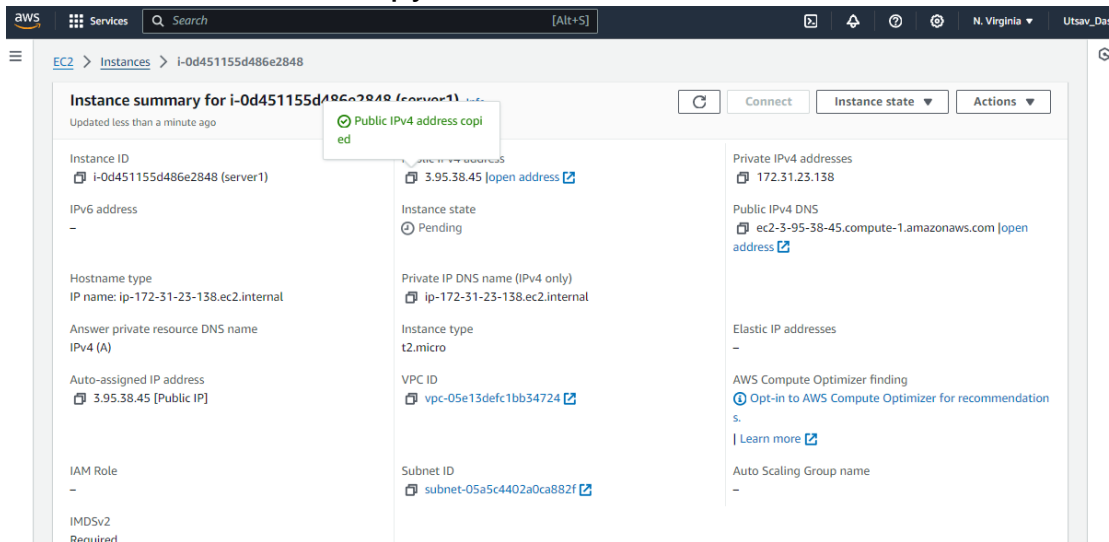
5) Give name of instance and in Application and OS Images select Ubuntu.

The screenshot shows the 'Launch an instance' wizard. The 'Name and tags' section has 'server1' entered in the Name field. The 'Application and OS Images (Amazon Machine Image)' section is active, showing a search bar and a 'Quick Start' section with various OS options. The 'Ubuntu' option is selected. Below the OS options, the 'Amazon Machine Image (AMI)' section shows 'Ubuntu Server 22.04 LTS (HVM), SSD Volume Type' as the selected AMI. The 'Free tier eligible' checkbox is checked.

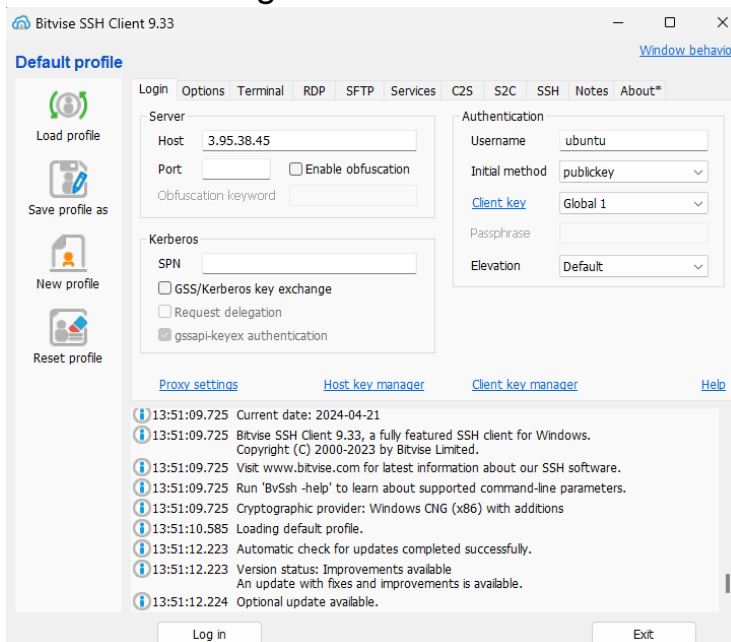
6) Select existing key pair and then click on Common Security group dropdown and select the created security group. Then click on Launch Instance.

The screenshot shows the 'Network settings' section of the 'Launch an instance' wizard. The 'Network' dropdown is set to 'vpc-05e13defc1bb34724'. The 'Subnet' dropdown is set to 'No preference (Default subnet in any availability zone)'. The 'Auto-assign public IP' dropdown is set to 'Enable'. The 'Security groups' dropdown is open, showing a list of security groups: 'my-security-group-1', 'my-security-group-2', and 'default'. 'my-security-group-2' is selected. A 'Compare security group rules' link is visible. On the right, the 'Virtual server type (instance type)' is set to 't2.micro', the 'Firewall (security group)' is set to 'my-security-group-2', and the 'Storage (volumes)' section shows '1 volume(s) - 8 GiB'. A 'Free tier' notification box is displayed, stating: '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 I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.' The 'Launch instance' button is visible at the bottom right.

7) Go to Instances and copy Public IPv4 address.



8) In Bitvise SSH Client, paste Copied IPv4 address, import the required key and click on Log In.



9) Open New Terminal Console and enter the following commands:

- `sudo apt-get update`
- `sudo apt upgrade`
- `sudo apt-get install nginx`
- `curl -SL https://deb.nodesource.com/setup_16.x | sudo -E bash -`
- `sudo apt install nodejs`
- `git clone https://github.com/UnderDevelopment10/new-repo1.git`
- `cd repo2`
- `npm install`
- `node index.js`

```
ubuntu@ip-172-31-23-138:~/new-repo1$ node index.js
Started server
^C
```

10) Paste the copied URL in browser. Then stop the server.

```
3.95.38.45
```

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

11) Now type the following commands:

- `cd /`
- `pwd`
- `cd etc/nginx/sites-available/`
- `sudo nano default`

```
ubuntu@ip-172-31-23-138:/etc/nginx/sites-available$ sudo nano default
```

12) In the “default” file, comment out all lines under location. Then type the following lines in place of location.

- `location / {`
 `proxy_pass http://localhost:4000;`
 `proxy_http_version 1.1;`
 `proxy_set_header Upgrade $http_upgrade;`
 `proxy_set_header Connection 'upgrade';`
 `proxy_set_header Host $host;`
 `proxy_cache_bypass $http_upgrade;`
 `}`

```
GNU nano 6.2                                     default *
# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name _;

# location / {
#     # First attempt to serve request as file, then
#     # as directory, then fall back to displaying a 404.
#     try_files $uri $uri/ =404;
# }

location / {
    proxy_pass http://localhost:4000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}

# pass PHP scripts to FastCGI server
#
#location ~ \.php$ {
#    include snippets/fastcgi-php.conf;
#
#    # With php-fpm (or other unix sockets):
#    fastcgi_pass unix:/run/php/php7.4-fpm.sock;
#    # With php-cgi (or other tcp sockets):
#    fastcgi_pass 127.0.0.1:9000;
#}

# deny access to .htaccess files, if Apache's document root
```

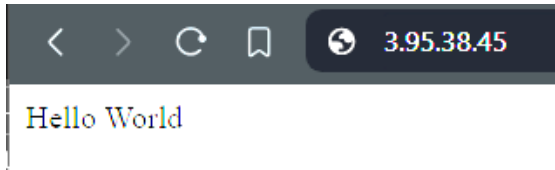
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^G Location M-U Undo
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line M-E Redo

To save and exit, press Ctrl+X, then Y and press Enter.

13) Open new server terminal and type

- `cd new-repo1`
- `sudo systemctl restart nginx`

14) Paste the copied IPv4 address in browser.



Here, the page has been accessed without using any port number with the IPv4 address.