

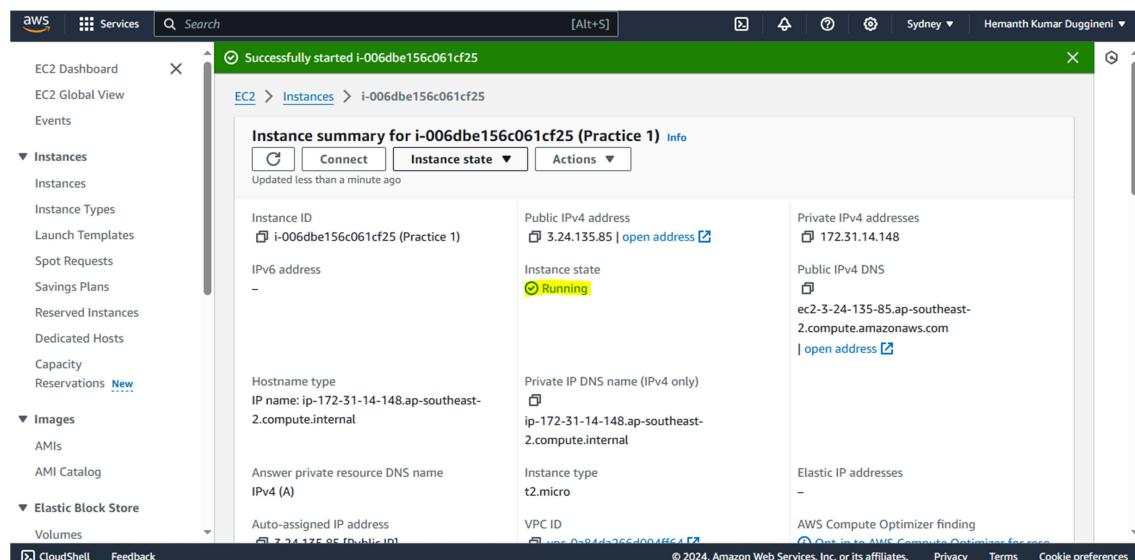
Deployment of Applications into a Server

Prerequisites:

- First we have to know the Application deployment region to decrease the latency.
- Need a Server with an OS to run the application.
- We have to know the Application size and usage size to make decision on volume and size of the Server respectively.
- We have to know the Properties of the application like in which language the application has built (Node JS, Python, Java, HTML, CSS, PHP, etc...) and with which port it has been associated.
- We have to check the status of the tools in the Server to deploy the applications.

Setup of a Server with an OS before starting the Deployment of Applications:

Step1: Check the status of the server which should be in running state.

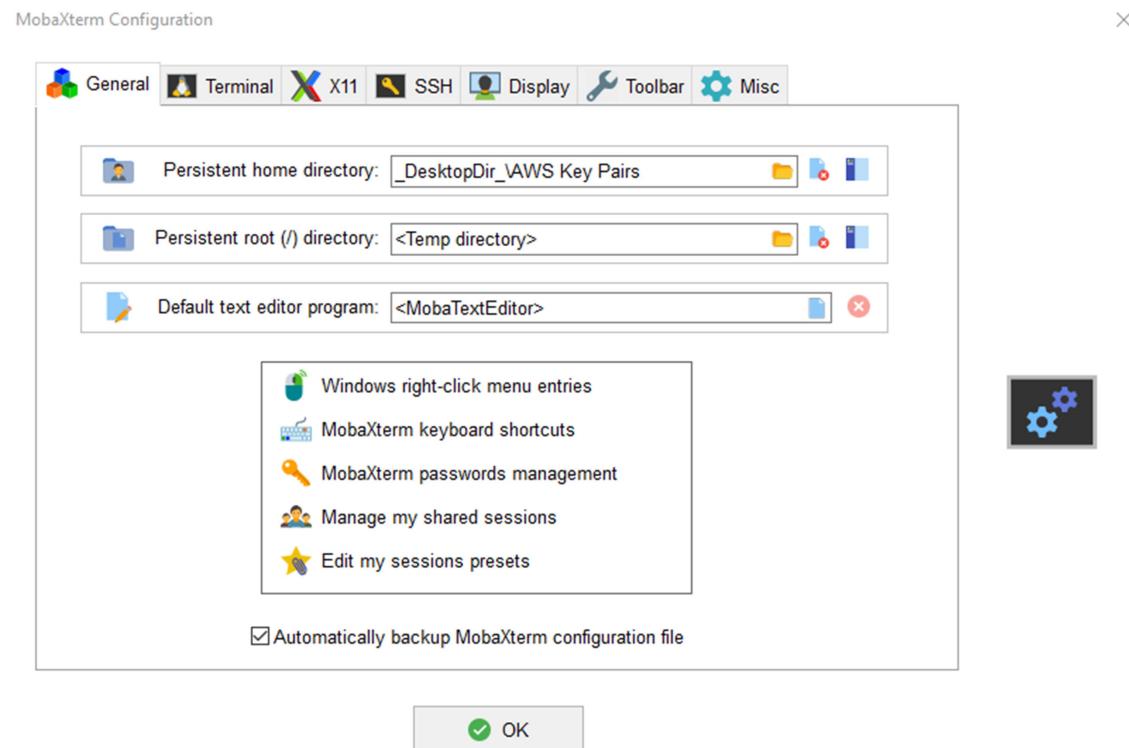


The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with navigation links like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, Reservations (New), Images (AMIs, AMI Catalog), and Elastic Block Store (Volumes). The main content area displays a single instance summary for 'i-006dbe156c061cf25 (Practice 1)'. The instance is in the 'Running' state. Key details shown include its Public IPv4 address (52.24.135.85), Private IP addresses (172.31.14.148), Public IPv4 DNS name (ec2-3-24-135-85.ap-southeast-2.compute.amazonaws.com), and its instance type (t2.micro). Other fields like Hostname type, IP name, Answer private resource DNS name (IPv4 A), Auto-assigned IP address, VPC ID, and AWS Compute Optimizer finding are also listed. A success message at the top says 'Successfully started i-006dbe156c061cf25'.

Step2: Connect through SSH client by locating Key pair.

AWS Key Pairs					
	Name	Date modified	Type	Size	
Quick access	HemanthInstanceLogin.pem	16-05-2024 13:45	PEM File	2 KB	
Desktop	Practice2InstanceKey.ppk	18-05-2024 10:27	PPK File	2 KB	
Downloads	PracticeInstanceKey.pem	16-05-2024 14:00	PEM File	2 KB	
Documents					
Pictures					
AWS Key Pairs					
Music					
Videos					
OneDrive - Personal					
This PC					
3D Objects					
Desktop					
Documents					
Downloads					
Music					
Pictures					
Videos					
Local Disk (C:)					
Local Disk (E:)					
Network					

Step3: Here I am using SSH client as MobaXterm and setting my Key pair path as default location in MobaXterm.



```

Terminal Sessions View Xserver Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
Quick connect...
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
2. ubuntu@Hemanth:~ 25/05/2024 14:19:32 /home/mobaxterm ssh -i "Prctice1InstanceKey.pem" ubuntu@ec2-3-24-135-85.ap-southeast-2.compute.amazonaws.com (3.24.135.85)
The authenticity of host 'ec2-3-24-135-85.ap-southeast-2.compute.amazonaws.com (3.24.135.85)' can't be established.
The key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-24-135-85.ap-southeast-2.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1000-aws x86_64)

System information as of Sat May 25 08:51:29 UTC 2024

System load: 0.08 Processes: 111
Usage of /: 29.9% of 6.71GB Users logged in: 1
Memory usage: 20% IPv4 address for enX0: 172.31.14.148
Swap usage: 0%

* Ubuntu Pro delivers the most comprehensive open source security and
compliance features.

https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

5 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Sat May 25 08:33:41 2024 from 106.221.191.13
/usr/bin/xauth: file /home/ubuntu/.Xauthority does not exist
ubuntu@Hemanth:~$ 

```

Follow terminal folder

Step4: Switch to Root access and here I am using a newly created server with Ubuntu Linux, so update the server with ‘`apt update`’ command.

```

root@Hemanth:~ 2. root@Hemanth:~ 25/05/2024 14:19:32 /home/mobaxterm ssh -i "Prctice1InstanceKey.pem" root@ec2-3-24-135-85.ap-southeast-2.compute.amazonaws.com (3.24.135.85)
Usage of /: 29.9% of 6.71GB Users logged in: 1
Memory usage: 20% IPv4 address for enX0: 172.31.14.148
Swap usage: 0%

* Ubuntu Pro delivers the most comprehensive open source security and
compliance features.

https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

5 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Sat May 25 08:51:33 2024 from 106.221.185.145
ubuntu@Hemanth:~$ sudo -i
root@Hemanth:~# apt update
Hit:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [89.7 kB]
Hit:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [77.1 kB]
Get:5 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [20.9 kB]
Get:6 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [35.7 kB]
Get:7 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [12.6 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security InRelease [89.7 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [32.2 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [10.3 kB]
Fetched 368 kB in 2s (198 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
15 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@Hemanth:~# 

```

mobaxterm has a donation to the non-free software here: <http://mobaxterm.mobatek.net>

Setups in a Server before Deploying Static Applications (HTML, CSS) into a Server:

Step1: Install the Apache2 web server by using ‘`apt install apache2 -y`’ command to create an access point of the CSS template to users and check the active status of the Apache2 by using ‘`systemctl status apache2`’ command.

```
root@Hemanth:~# apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 liblua5.4-0 ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 liblua5.4-0
  ssl-cert
0 upgraded, 10 newly installed, 0 to remove and 10 not upgraded.
Need to get 2080 kB of archives.
After this operation, 8091 kB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libapr1t64 amd64 1.7.2-3.1build2 [107 kB]
Get:2 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1t64 amd64 1.6.3-1.1ubuntu7 [91.2 kB]
Get:3 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.3-1.1ubuntu7 [11.2 kB]
Get:4 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-ldap amd64 1.6.3-1.1ubuntu7 [9116 kB]
Get:5 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 liblua5.4-0 amd64 5.4.6-3build2 [166 kB]
Get:6 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-bin amd64 2.4.58-1ubuntu8.1 [1327 kB]
Get:7 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-data all 2.4.58-1ubuntu8.1 [163 kB]
Get:8 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2-utils amd64 2.4.58-1ubuntu8.1 [96.2 kB]
Get:9 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 apache2 amd64 2.4.58-1ubuntu8.1 [90.2 kB]
Get:10 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble/main amd64 ssl-cert all 1.1.2ubuntu1 [17.8 kB]
Fetched 2080 kB in 0s (4529 kB/s)
Preconfiguring packages ...
Selecting previously unselected package libapr1t64:amd64.
(Reading database ... 71841 files and directories currently installed.)
Preparing to unpack .../0-libapr1t64_1.7.2-3.1build2_amd64.deb ...
Unpacking libapr1t64:amd64 (1.7.2-3.1build2) ...
Selecting previously unselected package libaprutil1t64:amd64.
Preparing to unpack .../1-libaprutil1t64_1.6.3-1.1ubuntu7_amd64.deb ...
Unpacking libaprutil1t64:amd64 (1.6.3-1.1ubuntu7) ...
Selecting previously unselected package libaprutil1-dbd-sqlite3:amd64.
Preparing to unpack .../2-libaprutil1-dbd-sqlite3_1.6.3-1.1ubuntu7_amd64.deb ...
Unpacking libaprutil1-dbd-sqlite3:amd64 (1.6.3-1.1ubuntu7) ...
Selecting previously unselected package libaprutil1-ldap:amd64.
Preparing to unpack .../3-libaprutil1-ldap_1.6.3-1.1ubuntu7_amd64.deb ...
```

```
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...

Service restarts being deferred:
systemctl restart systemd-logind.service

No containers need to be restarted.

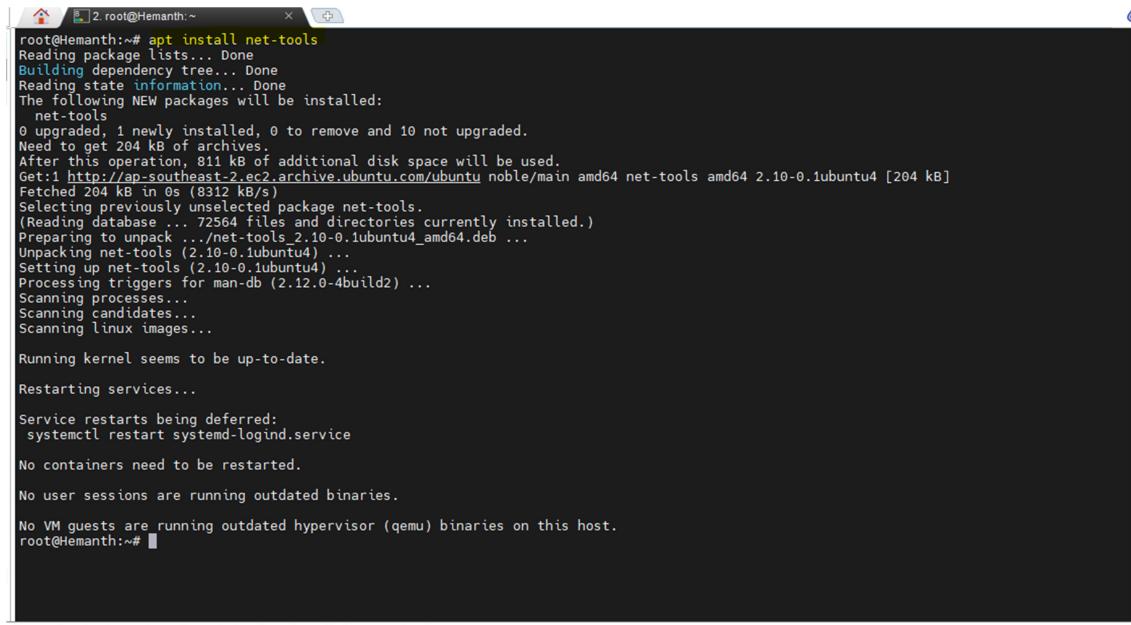
User sessions running outdated binaries:
ubuntu @ session #1: sshd[821]
ubuntu @ session #12: sshd[1950]
ubuntu @ session #13: sshd[2006]
ubuntu @ session #9: sshd[1519]
ubuntu @ user manager service: systemd[826]

No VM guests are running outdated hypervisor (qemu) binaries on this host.

root@Hemanth:~# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-05-25 09:12:38 UTC; 3min 4s ago
       Docs: https://httpd.apache.org/docs/2.4/
 Main PID: 7399 (apache2)
   Tasks: 55 (limit: 1130)
   Memory: 5.4M (peak: 5.6M)
      CPU: 49ms
     CGroup: /system.slice/apache2.service
             ├─7399 /usr/sbin/apache2 -k start
             ├─7402 /usr/sbin/apache2 -k start
             ├─7403 /usr/sbin/apache2 -k start

May 25 09:12:38 Hemanth systemd[1]: Starting apache2.service - The Apache HTTP Server...
May 25 09:12:38 Hemanth systemd[1]: Started apache2.service - The Apache HTTP Server.
root@Hemanth:~#
```

Step2: Since I am using the newly created server we have to install net-tools too with '**apt install net-tools**' command, to know the listen status of the ports.



```
root@Hemanth:~# apt install net-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 10 not upgraded.
Need to get 204 kB of additional disk space.
After this operation, 811 kB of additional disk space will be used.
Get:1 http://ap-southeast-2.ec2.archive.ubuntu.com/ubuntu noble amd64 net-tools amd64 2.10-0.1ubuntu4 [204 kB]
Fetched 204 kB in 0s (8312 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 72564 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...

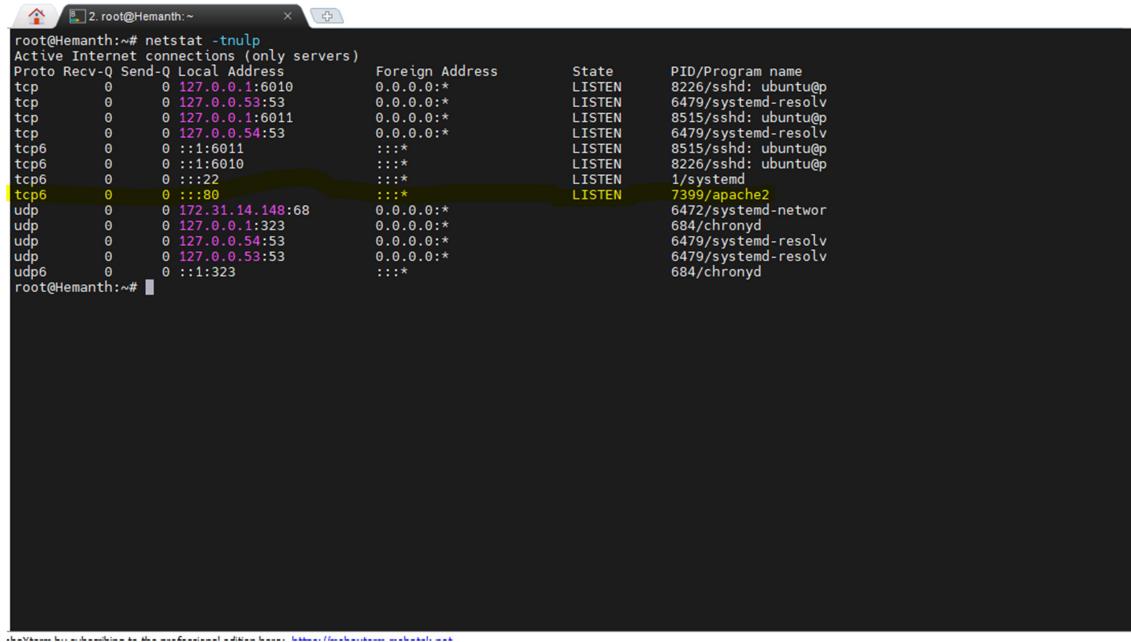
Service restarts being deferred:
  systemctl restart systemd-logind.service

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.
root@Hemanth:~#
```

Step3: By entering ‘**netstat –tnulp**’ command we can see the Listen status of the ports.



```
root@Hemanth:~# netstat -tnulp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State      PID/Program name
tcp     0      0 127.0.0.1:6010            0.0.0.0:*
                                         LISTEN
tcp     0      0 127.0.0.53:53             0.0.0.0:*
                                         LISTEN
tcp     0      0 127.0.0.1:6011            0.0.0.0:*
                                         LISTEN
tcp6    0      0 127.0.0.54:53             0.0.0.0:*
                                         LISTEN
tcp6    0      0 ::1:6011                  :::*
                                         LISTEN
tcp6    0      0 ::1:6010                  :::*
                                         LISTEN
tcp6    0      0 ::1:22                   :::*
                                         LISTEN
tcp6    0      0 ::1:80                   :::*
                                         LISTEN
                                         7399/apache2
udp     0      0 172.31.14.148:68           0.0.0.0:*
                                         LISTEN
                                         6472/systemd-network
                                         684/chrony
udp     0      0 127.0.0.1:323             0.0.0.0:*
                                         LISTEN
                                         6479/systemd-resolv
                                         6479/systemd-resolv
                                         684/chrony
udp6    0      0 127.0.0.53:53             0.0.0.0:*
                                         LISTEN
                                         6479/systemd-resolv
                                         684/chrony
root@Hemanth:~#
```

Step4: Check the port is connecting or not by entering ‘**telnet <Public IP> <Port>**’ command.

```

root@Hemanth:~# netstat -tnulp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State      PID/Program name
tcp        0      0 127.0.0.1:6010          0.0.0.0:*
                                         LISTEN
tcp        0      0 127.0.0.53:53          0.0.0.0:*
                                         LISTEN
tcp        0      0 127.0.0.1:6011          0.0.0.0:*
                                         LISTEN
tcp        0      0 127.0.0.54:53          0.0.0.0:*
                                         LISTEN
tcp6       0      0 ::1:6011              ::*:*
                                         LISTEN
tcp6       0      0 ::1:6010              ::*:*
                                         LISTEN
tcp6       0      0 ::1:22                ::*:*
                                         LISTEN
tcp6       0      0 ::1:80                ::*:*
                                         LISTEN
                                         1/systemd
                                         7399/apache2
                                         6472/systemd-networ
                                         684/chrony
                                         6479/systemd-resolv
                                         6479/systemd-resolv
                                         684/chrony
                                         684/chrony
root@Hemanth:~# curl ifconfig.io
3.24.135.85
root@Hemanth:#
root@Hemanth:~# telnet 3.24.135.85 80
Trying 3.24.135.85...

```

Step5: Here we can see that trying to connect but not connecting via required port because we didn't configure the port in security groups of the server in AWS console. So we have to configure required port in security groups of the server in AWS console.

Go to Instances page and select your server, under Security section click on the link under Security groups.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
Practice 1	i-006dbe156c061cf25	Running	t2.micro	2/2 checks passed	View alarms	ap-
Hemanth	i-0788726e119741745	Stopped	t2.micro	-	View alarms	ap-

i-006dbe156c061cf25 (Practice 1)

Security details

IAM Role	Owner ID	Launch time
-	851725491699	Sat May 25 2024 13:59:05 GMT+0530 (India Standard Time)
Security groups	sg-00c3c9865c90bcb63 (launch-wizard-2)	

Under Inbound rules section click on edit inbound rules.

The screenshot shows the AWS EC2 Security Groups page. The left sidebar has sections for Instances, Images, and Elastic Block Store. The main content shows the 'sg-00c3c9865c90bcb63 - launch-wizard-2' security group. The 'Details' section includes fields for Security group name (launch-wizard-2), Security group ID (sg-00c3c9865c90bcb63), Description (launch-wizard-2 created 2024-05-16T08:26:41.911Z), Owner (851725491699), VPC ID (vpc-0a84da266d094ff64), Inbound rules count (1 Permission entry), and Outbound rules count (1 Permission entry). Below this are tabs for 'Inbound rules', 'Outbound rules', and 'Tags'. The 'Inbound rules' tab shows one rule: 'Inbound rules (1)'. The rule table has columns for Security group rule ID, Type, Protocol, Port range, Source, and Description - optional. The first rule is for SSH (TCP port 22) from 'Cus...' (0.0.0.0/0). The second rule is for HTTP (TCP port 80) from 'Any...' (0.0.0.0/0). A yellow box highlights the 'Edit inbound rules' button. The bottom of the page includes a search bar, navigation links, and a footer with copyright information.

Click on Add rule and select

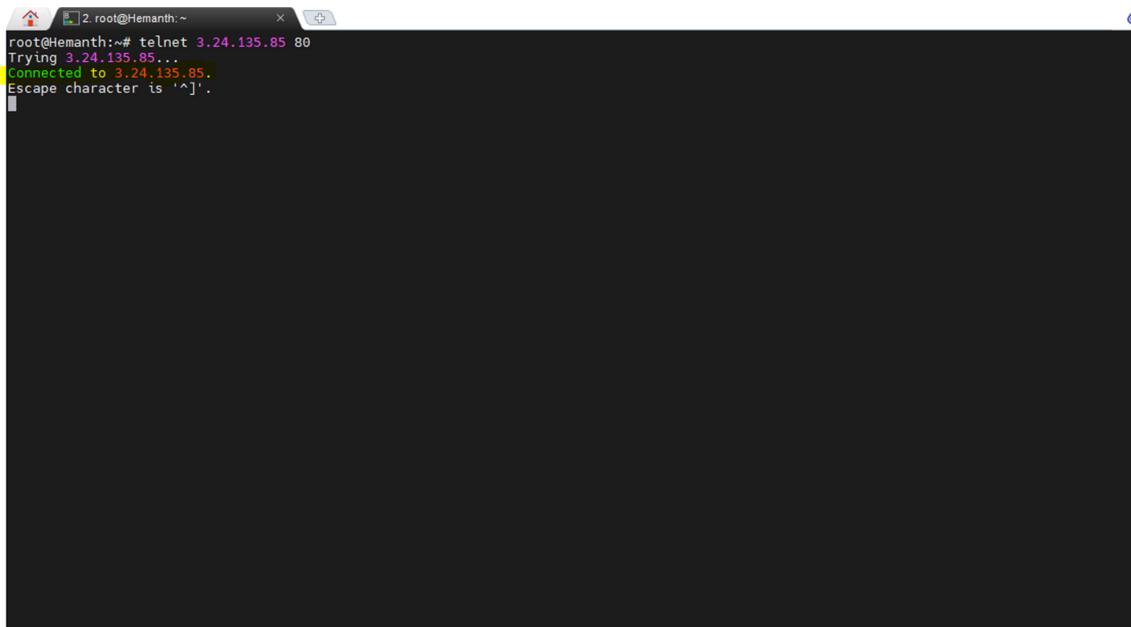
Type: HTTP (required protocol type)

Source: Anywhere-IPv4 (required access network IP addresses)

And click on save rules.

The screenshot shows the 'Inbound rules' configuration page. It lists two rules: one for SSH (TCP port 22) and one for HTTP (TCP port 80). A yellow box highlights the 'Add rule' button. A warning message at the bottom states: '⚠ 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.' The bottom right features 'Cancel', 'Preview changes', and a yellow 'Save rules' button. The footer includes standard AWS links and a cookie preferences link.

Step6: Again check the port is connecting or not by entering '**telnet <Public IP> <Port>**' command.



```
root@Hemanth:~# telnet 3.24.135.85 80
Trying 3.24.135.85...
Connected to 3.24.135.85.
Escape character is '^]'.
```

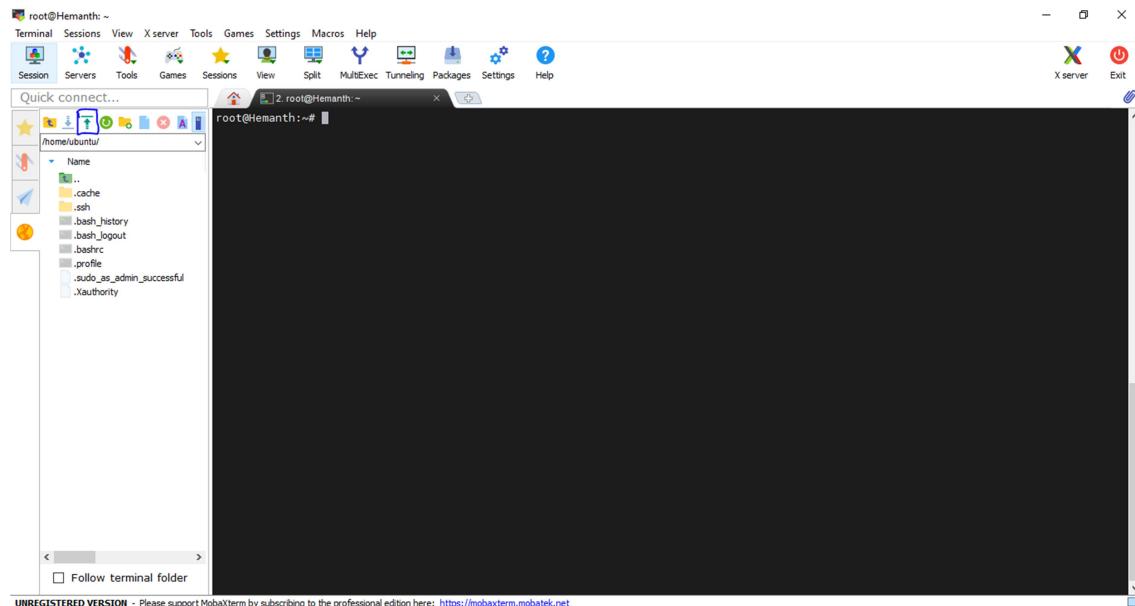
Here we can see that server is responding via required port, as we can confirm that access point has been created via web server to access the applications in the server.

Deployment of the Static Applications (HTML,CSS) into a Server:

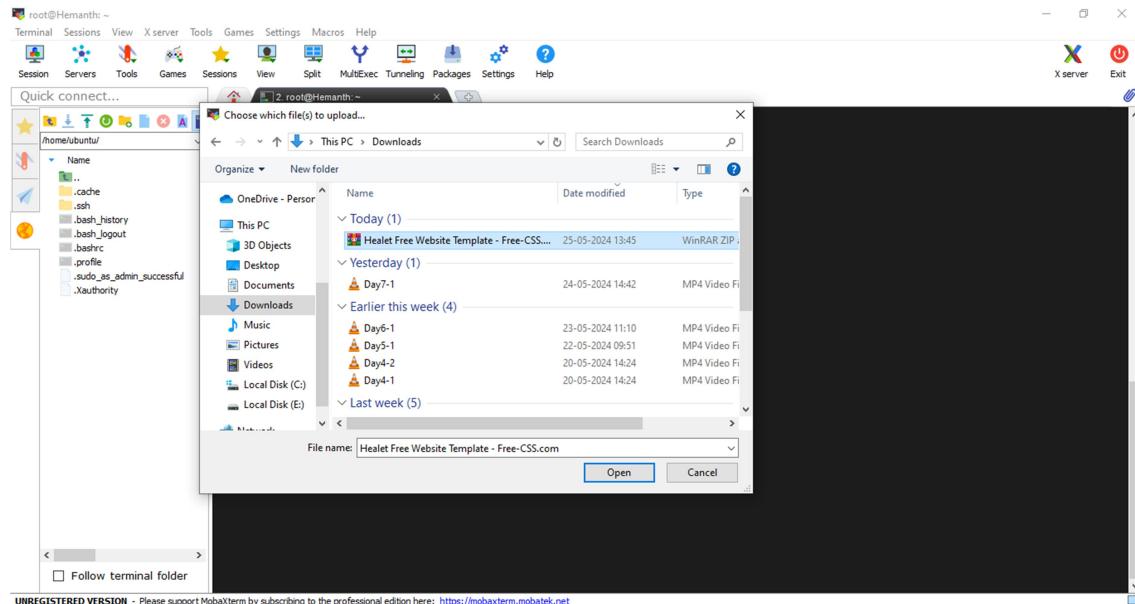
Step1: Here I am using MobaXterm to transfer our application files to our server.

Note: Before transferring the files to the server it is better to zip the files to prevent the loss of data of the application in the server.

Click on Upload button.



Select our Application files to upload.

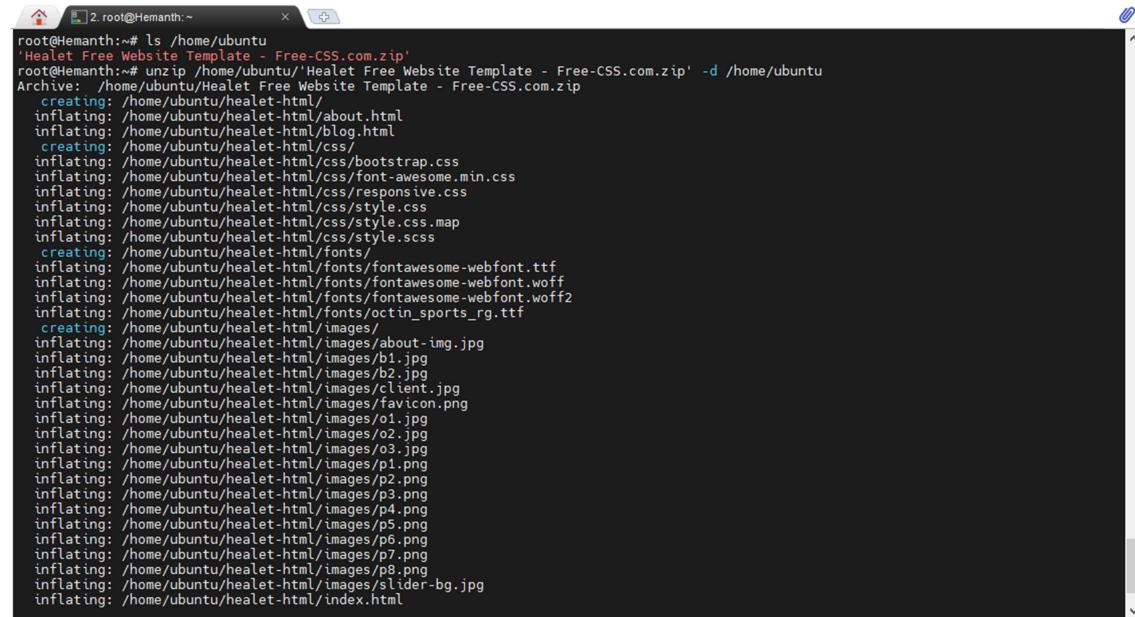


UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

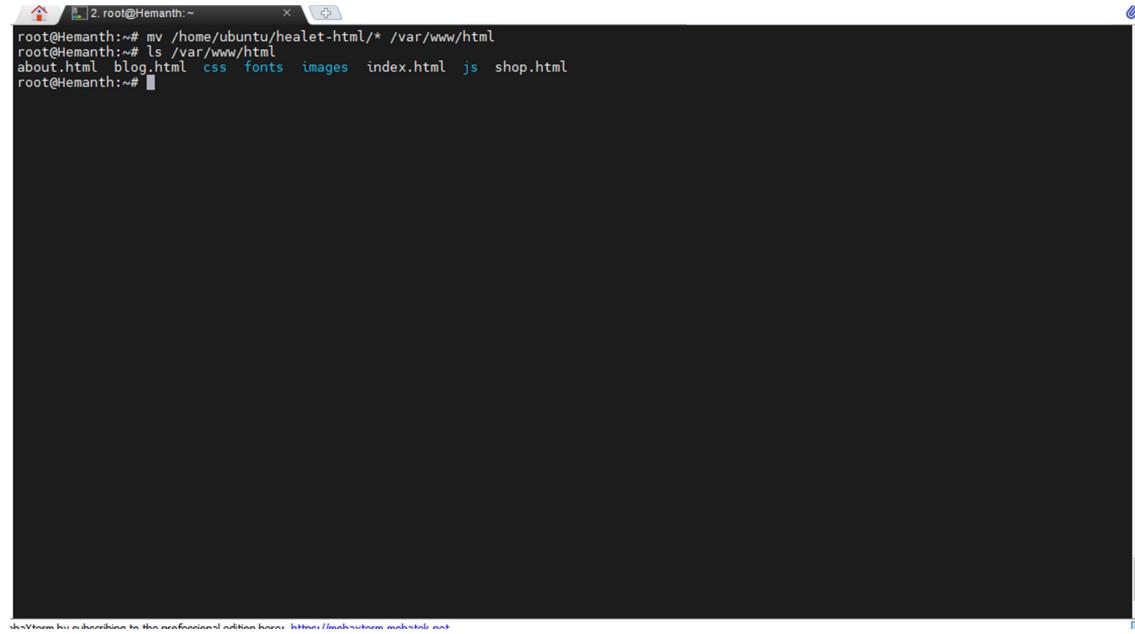
Note: We can use '**git clone <Github URL>**' command to transfer our application files from our Github account and we can proceed with the deployment process.

Locate the transferred files, unzip them with the command '**unzip <File path> -d <Destination path>**' and move to Web server default location (/var/www/html) with command '**mv <Source path> <Destination path>**'.

A screenshot of the MobaXterm terminal window. The terminal session is titled '2. root@Hemanth: ~'. The user has run the command 'ls /home/ubuntu' and the output shows a single file named 'Healet Free Website Template - Free-CSS.com.zip'. The user then runs 'mv /home/ubuntu/Healet Free Website Template - Free-CSS.com.zip' /var/www/html'. Finally, 'ls /var/www/html' is run, showing the contents of the directory, which include 'Healet Free Website Template - Free-CSS.com.zip' and 'index.html'. The terminal window has a dark background with light-colored text. The bottom of the window shows the MobaXterm copyright notice: 'MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>'.



```
root@Hemanth:~# ls /home/ubuntu
'Healet Free Website Template - Free-CSS.com.zip'
root@Hemanth:~# unzip '/home/ubuntu/Healet Free Website Template - Free-CSS.com.zip' -d /home/ubuntu
Archive: /home/ubuntu/Healet Free Website Template - Free-CSS.com.zip
  creating: /home/ubuntu/healet-html/
  inflating: /home/ubuntu/healet-html/about.html
  inflating: /home/ubuntu/healet-html/blog.html
  creating: /home/ubuntu/healet-html/css/
  inflating: /home/ubuntu/healet-html/css/bootstrap.css
  inflating: /home/ubuntu/healet-html/css/font-awesome.min.css
  inflating: /home/ubuntu/healet-html/css/responsive.css
  inflating: /home/ubuntu/healet-html/css/style.css
  inflating: /home/ubuntu/healet-html/css/style.css.map
  inflating: /home/ubuntu/healet-html/css/style.scss
  creating: /home/ubuntu/healet-html/fonts/
  inflating: /home/ubuntu/healet-html/fonts/fontawesome-webfont.ttf
  inflating: /home/ubuntu/healet-html/fonts/fontawesome-webfont.woff
  inflating: /home/ubuntu/healet-html/fonts/fontawesome-webfont.woff2
  inflating: /home/ubuntu/healet-html/fonts/octin_sports_rg.ttf
  creating: /home/ubuntu/healet-html/images/
  inflating: /home/ubuntu/healet-html/images/about-img.jpg
  inflating: /home/ubuntu/healet-html/images/b1.jpg
  inflating: /home/ubuntu/healet-html/images/b2.jpg
  inflating: /home/ubuntu/healet-html/images/client.jpg
  inflating: /home/ubuntu/healet-html/images/favicon.png
  inflating: /home/ubuntu/healet-html/images/o1.jpg
  inflating: /home/ubuntu/healet-html/images/o2.jpg
  inflating: /home/ubuntu/healet-html/images/o3.jpg
  inflating: /home/ubuntu/healet-html/images/p1.png
  inflating: /home/ubuntu/healet-html/images/p2.png
  inflating: /home/ubuntu/healet-html/images/p3.png
  inflating: /home/ubuntu/healet-html/images/p4.png
  inflating: /home/ubuntu/healet-html/images/p5.png
  inflating: /home/ubuntu/healet-html/images/p6.png
  inflating: /home/ubuntu/healet-html/images/p7.png
  inflating: /home/ubuntu/healet-html/images/p8.png
  inflating: /home/ubuntu/healet-html/images/slider-bg.jpg
  inflating: /home/ubuntu/healet-html/index.html
```

```
root@Hemanth:~# mv /home/ubuntu/healet-html/* /var/www/html
root@Hemanth:~# ls /var/www/html
about.html blog.html css fonts images index.html js shop.html
root@Hemanth:~#
```

Step2: Now try to open the access point (<Public IP>:<Port>) in a browser.

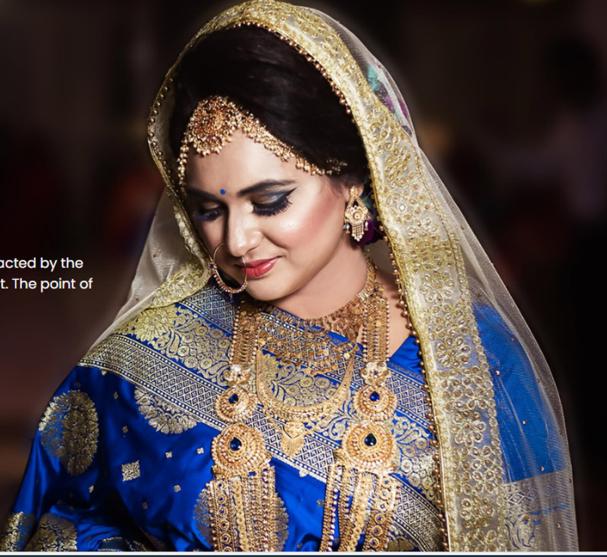
Not secure | 3.24.135.85

HEALET

Best Jewellery Collection

It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem

Shop Now

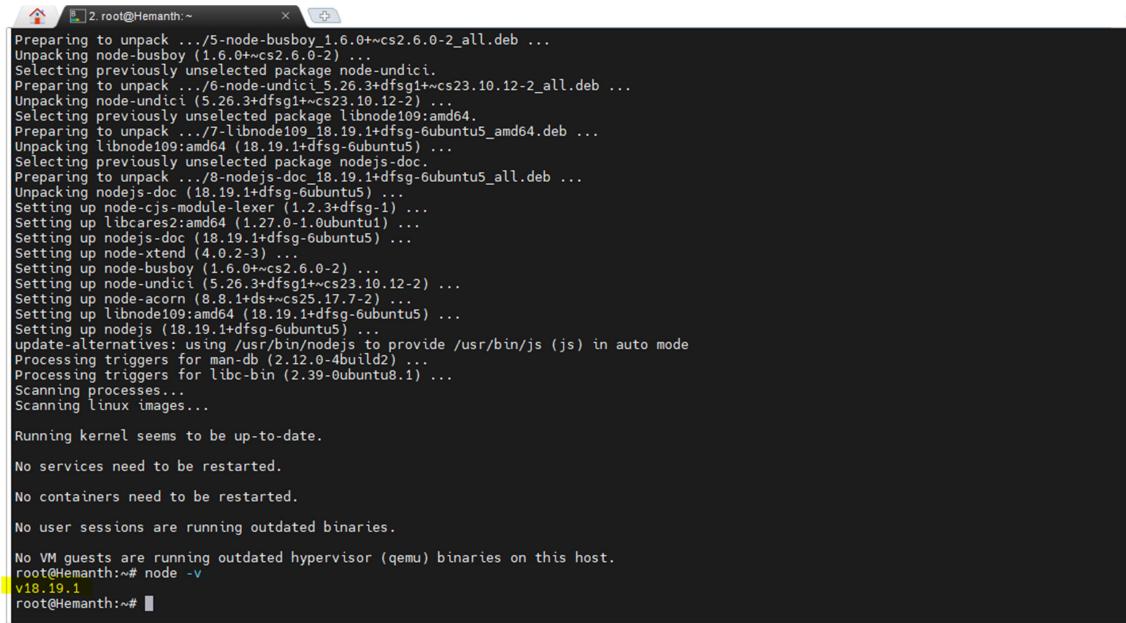


Deployment of a NodeJS Application into a Server:

- In NodeJS applications we have Frontend and Backend applications.
- Below I will show how to deploy both applications step by step.

Setups in a Server before Deploying NodeJS Applications into a Server:

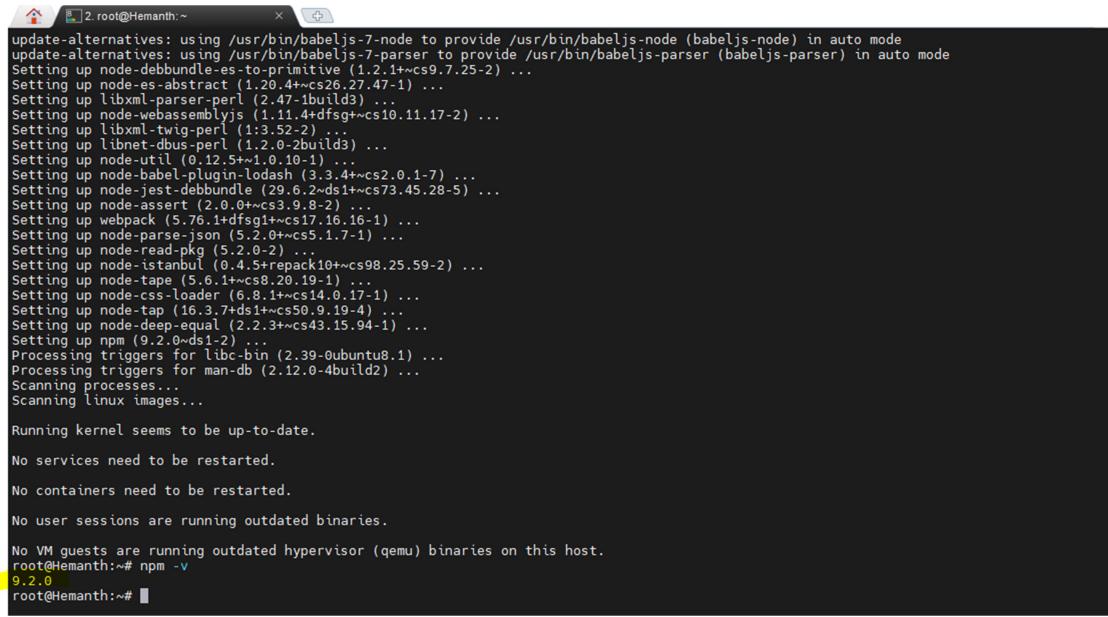
Step1: By using ‘`apt install nodejs -y`’ command we have to install nodejs tools to resolve the compatibility setting of the nodejs applications with the server and by using ‘`node -v`’ command we have to check the version and confirm that nodejs tools have been installed.



```
root@Hemanth:~# Preparing to unpack .../5-node-busboy_1.6.0+~cs2.6.0-2_all.deb ...
Unpacking node-busboy (1.6.0+~cs2.6.0-2) ...
Selecting previously unselected package node-undici.
Preparing to unpack .../6-node-undici_5.26.3+dfsg1+~cs23.10.12-2_all.deb ...
Unpacking node-undici (5.26.3+dfsg1+~cs23.10.12-2) ...
Selecting previously unselected package libnode109:amd64.
Preparing to unpack .../7-libnode109_18.19.1+dfsg-6ubuntu5_amd64.deb ...
Unpacking libnode109:amd64 (18.19.1+dfsg-6ubuntu5) ...
Selecting previously unselected package nodejs-doc.
Preparing to unpack .../8-nodejs-doc_18.19.1+dfsg-6ubuntu5_all.deb ...
Unpacking nodejs-doc (18.19.1+dfsg-6ubuntu5) ...
Setting up node-cjs-module-lexer (1.2.3+dfsg-1) ...
Setting up libcares2:amd64 (1.27.0-1.0ubuntu1) ...
Setting up nodejs-doc (18.19.1+dfsg-6ubuntu5) ...
Setting up node-xtend (4.0.2-3) ...
Setting up node-busboy (1.6.0+~cs2.6.0-2) ...
Setting up node-undici (5.26.3+dfsg1+~cs23.10.12-2) ...
Setting up node-acorn (8.8.1+dfsg+~cs25.17.7-2) ...
Setting up libnode109:amd64 (18.19.1+dfsg-6ubuntu5) ...
Setting up nodejs (18.19.1+dfsg-6ubuntu5) ...
update-alternatives: using /usr/bin/nodejs to provide /usr/bin/js (js) in auto mode
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.1) ...
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.
root@Hemanth:~# node -v
v18.19.1
root@Hemanth:~#
```

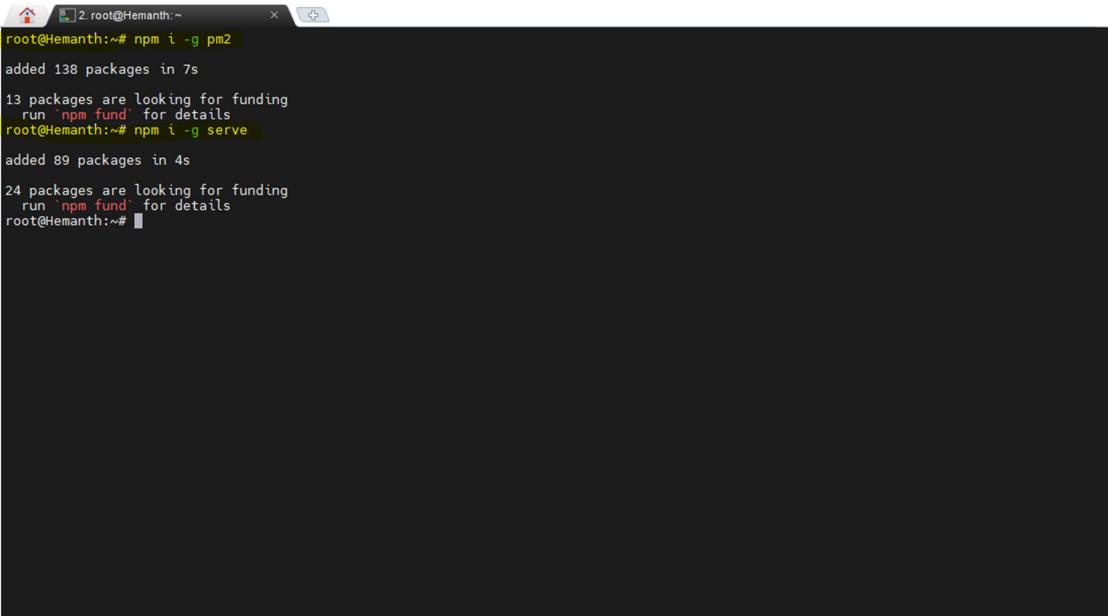
Step2: By using ‘`apt install npm -y`’ command we have to install Node Package Manager Tools to resolve the compatibility issue of the applications while running and by using ‘`npm -v`’ command we have to check the version and confirm that npm tools have been installed.



```
root@Hemanth:~# dpkg -l
update-alternatives: using /usr/bin/babeljs-7-node to provide /usr/bin/babeljs-node (babeljs-node) in auto mode
update-alternatives: using /usr/bin/babeljs-7-parser to provide /usr/bin/babeljs-parser (babeljs-parser) in auto mode
Setting up node-debbundle-es-to-primitive (1.2.1+~cs26.27.47-1) ...
Setting up libxml-parser-perl (2.47-1build3) ...
Setting up node-wabassemblyjs (1.11.4+dfsg+~cs10.11.17-2) ...
Setting up libxml-twig-perl (1:3.52-2) ...
Setting up libnet-dbus-perl (1.2.0-2build3) ...
Setting up node-util (0.12.5+~1.0.10-1) ...
Setting up node-babel-plugin-lodash (3.3.4+~cs2.0.1-7) ...
Setting up node-jest-debbundle (29.6.2~ds1+~cs73.45.28-5) ...
Setting up node-assert (2.0.0+~cs3.9.8-2) ...
Setting up webpack (5.76.1+dfsg1+~cs17.16.16-1) ...
Setting up node-parse-json (5.2.0+~cs5.1.7-1) ...
Setting up node-read-pkg (5.2.0-2) ...
Setting up node-istanbul (0.4.5+repack10+~cs98.25.59-2) ...
Setting up node-tape (5.6.1+~cs8.20.19-1) ...
Setting up node-css-loader (6.8.1+~cs14.0.17-1) ...
Setting up node-tap (16.3.7~ds1+~cs50.9.19-4) ...
Setting up node-deep-equal (2.2.3+~cs43.15.94-1) ...
Setting up npm (9.2.0~ds1-2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
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.
root@Hemanth:~# npm -v
9.2.0
root@Hemanth:~#
```

Step3: By using ‘**npm i pm2**’ and ‘**npm i serve**’ commands we have to install ‘Package Manager Monitor’ and ‘Serve’ tools to run the NodeJS applications.



```
root@Hemanth:~# npm i -g pm2
added 138 packages in 7s
13 packages are looking for funding
  run `npm fund` for details
root@Hemanth:~# npm i -g serve
added 89 packages in 4s
24 packages are looking for funding
  run `npm fund` for details
root@Hemanth:~#
```

Deployment of Frontend NodeJS Applications:

Step1: Here I am using ‘**git clone <Github URL>**’ to transfer our files into the server from Github account and saved as the Repository name which is in the Github account.

The screenshot shows a GitHub repository page for 'simple-node-js-react-npm-app'. The 'Code' tab is selected. A context menu is open over the repository details, with the 'Clone' option highlighted. The menu also includes options for 'HTTPS' and 'GitHub CLI', along with links to 'Clone using the web URL', 'Open with GitHub Desktop', and 'Download ZIP'. The repository details on the right include the branch 'master', 1 branch, 0 tags, and an 'About' section describing the repository as an introductory tutorial on how to use Jenkins to build a simple Node.js and React application with npm. The terminal window below shows the command 'git clone https://github.com/koteswararao73/simple-node-js-react-npm-app.git' being run, and the output shows the repository being cloned successfully.

Step2: Enter into the transferred file with ‘`cd <Directory Name>`’ command and by using ‘`npm i`’ command install the compatibilities for the application files and the compatibilities can be stored in a directory named as ‘`node_modules`’.

Note: Before using any npm, nodejs, pm2, serve tools your cursor location should be in the project or application directory.

```
root@Hemanth:~/simple-node-# npm audit
npm WARN deprecated browserslist@1.7.7: Browserslist 2 could fail on reading Browserslist >3.0 config used in other tools.
npm WARN deprecated browserslist@1.7.7: Browserslist 2 could fail on reading Browserslist >3.0 config used in other tools.
npm WARN deprecated html-webpack-plugin@2.29.0: out of support
npm WARN deprecated chokidar@1.7.0: Chokidar 2 will break on node v14+. Upgrade to chokidar 3 with 15x less dependencies.
npm WARN deprecated sw-precache@5.2.1: Please migrate to Workbox: https://developers.google.com/web/tools/workbox/guides/migrations/migrate-from-sw
npm WARN deprecated uid@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.
npm WARN deprecated chokidar@2.1.8: Chokidar 2 does not receive security updates since 2019. Upgrade to chokidar 3 with 15x fewer dependencies.
npm WARN deprecated extract-text-webpack-plugin@3.0.0: Deprecated. Please use https://github.com/webpack-contrib/mini-css-extract-plugin
npm WARN deprecated uid@2.0.3: 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.
npm WARN deprecated sw-toolbox@3.6.0: Please migrate to Workbox: https://developers.google.com/web/tools/workbox/guides/migrations/migrate-from-sw
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated svgo@0.7.2: This SVGO version is no longer supported. Upgrade to v2.x.x.
npm WARN deprecated core-js@2.6.12: core-js@3.23.3 is no longer maintained and not recommended for usage due to the number of issues. Because of the V8 engine whisms, feature detection in old core-js versions could cause a slowdown up to 100x even if nothing is polyfilled. Some versions have web compatibility issues. Please, upgrade your dependencies to the actual version of core-js.

added 1359 packages, and audited 1360 packages in 2m

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

146 vulnerabilities (11 low, 61 moderate, 29 high, 45 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.
root@Hemanth:~/simple-node-js-react-npm-app# ls
Dockerfile Jenkinsfile README.md deployment.yaml jenkins node_modules package-lock.json package.json public src
root@Hemanth:~/simple-node-js-react-npm-app#
```

Step3: By using ‘**npm run build**’ command we can build the application with the existing files and which can be stored in a directory named as ‘build’.

```
root@Hemanth:~/simple-node-js-react-npm-app# npm run build
> my-app@0.1.0 build
> react-scripts build

Creating an optimized production build...
Compiled successfully.

File sizes after gzip:
 45.03 KB  build/static/js/main.9940296e.js
 299 B     build/static/css/main.c17080f1.css

The project was built assuming it is hosted at the server root.
To override this, specify the homepage in your package.json.
For example, add this to build for GitHub Pages:

  "homepage" : "http://myname.github.io/myapp",

The build folder is ready to be deployed.
You may serve it with a static server:
  serve -s build

root@Hemanth:~/simple-node-js-react-npm-app# ls
Dockerfile Jenkinsfile README.md build deployment.yaml jenkins node_modules package-lock.json package.json public src
root@Hemanth:~/simple-node-js-react-npm-app#
```

Step4: Now we have to allow the application port in Security Groups.

Go to Instances page and select your server, under Security section click on the link under Security groups.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instances Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, Reservations (New), Images, AMIs, AMI Catalog, and Elastic Block Store (Volumes). The main area displays 'Instances (1/2) Info' with a search bar and filters for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. Two instances are listed: 'Practice 1' (i-006dbe156c061cf25) is Running, t2.micro, and has 2/2 checks passed. 'Hemanth' (i-0788726e119741745) is Stopped, t2.micro, and has 0 checks passed. Below the table, a detailed view for 'i-006dbe156c061cf25 (Practice 1)' is shown, including Security details (IAM Role: -, Owner ID: 851725491699), Launch time (Sat May 25 2024 13:59:05 GMT+0530 (India Standard Time)), and Security groups (sg-00c3c9865c90bcb63 (launch-wizard-2)).

Under Inbound rules section click on edit inbound rules.

The screenshot shows the AWS Security Groups page for the security group 'sg-00c3c9865c90bcb63 - launch-wizard-2'. The left sidebar is identical to the previous screenshot. The main area shows the 'Details' section with fields: Security group name (launch-wizard-2), Security group ID (sg-00c3c9865c90bcb63), Description (launch-wizard-2 created 2024-05-16T08:26:41.911Z), Owner (851725491699), Inbound rules count (1 Permission entry), and Outbound rules count (1 Permission entry). Below this, the 'Inbound rules (1)' section is visible, containing a single rule with a yellow border around the 'Edit inbound rules' button. A search bar and a 'Manage tags' button are also present in this section.

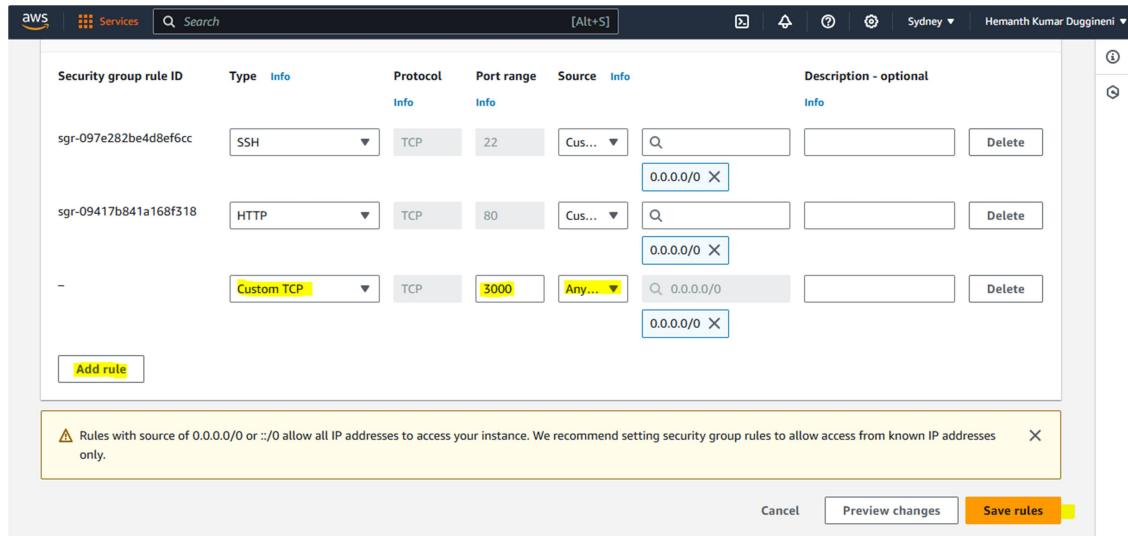
Click on Add rule and select

Type: Custom TCP (required protocol type)

Port Range: Required Port number

Source: Anywhere-IPv4 (required access network IP addresses)

And click on save rules.



The screenshot shows the AWS Management Console interface for managing security group rules. The top navigation bar includes the AWS logo, Services, a search bar, and account information for 'Hemanth Kumar Duggineni' in 'Sydney'. The main content area displays a table of security group rules:

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-097e282be4d8ef6cc	SSH	TCP	22	Cus... ▾	Info Delete 0.0.0.0/0 X
sgr-09417b841a168f318	HTTP	TCP	80	Cus... ▾	Info Delete 0.0.0.0/0 X
-	Custom TCP	TCP	3000	Any... ▾	Info Delete 0.0.0.0/0 X

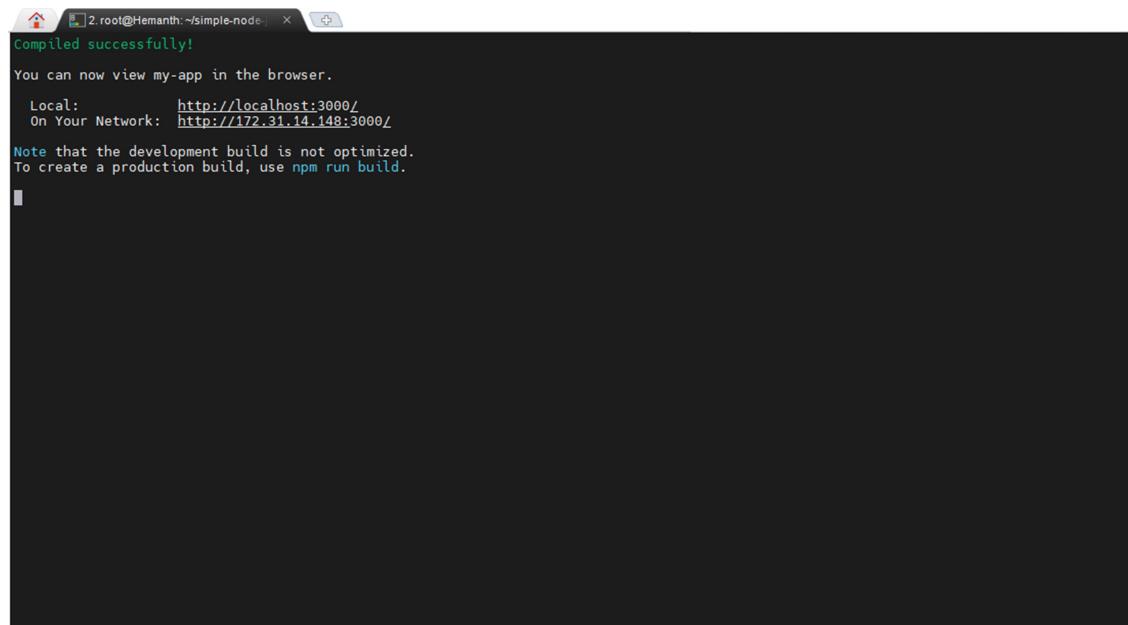
At the bottom left is a yellow 'Add rule' button. A warning message box states: '⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.' with a close button 'X'. At the bottom right are 'Cancel', 'Preview changes', and a large orange 'Save rules' button.

Up to here deployment of application is done.

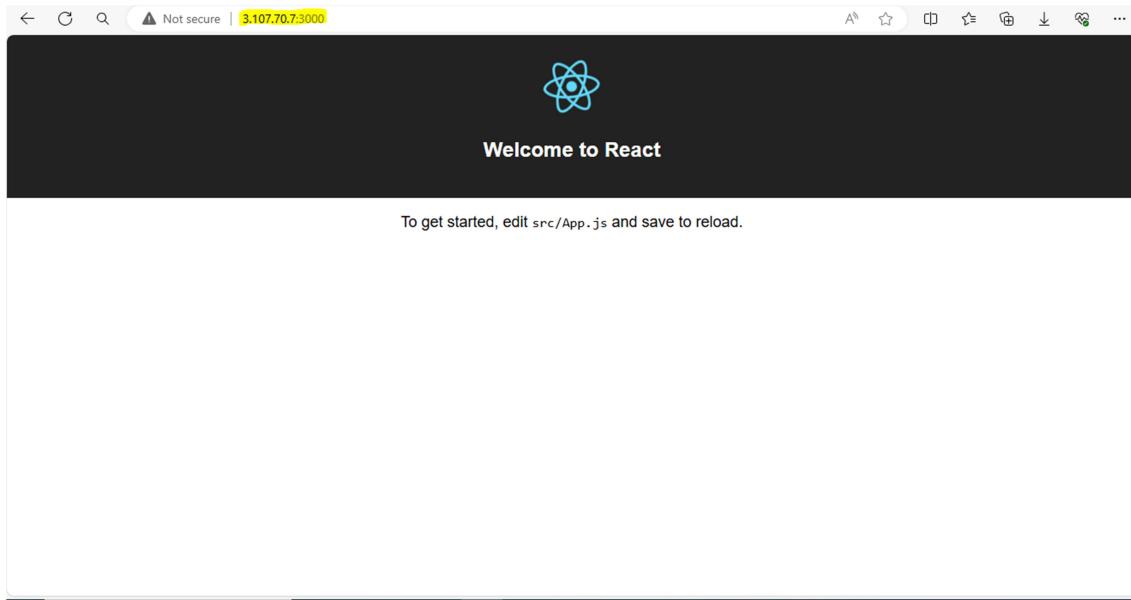
To run the deployed application there are two ways:

Temporary Run: By using '**npm start**' we can run our application temporarily up to we click on 'ctrl+c' in the server.

After passing the command '**npm start**' the application will start running with the access point.



```
2 root@Hemanth:~/simple-node ~ +  
Compiled successfully!  
You can now view my-app in the browser.  
Local: http://localhost:3000/  
On Your Network: http://172.31.14.148:3000/  
Note that the development build is not optimized.  
To create a production build, use npm run build.
```



After entering 'ctrl+c', the application will stop running with access point.

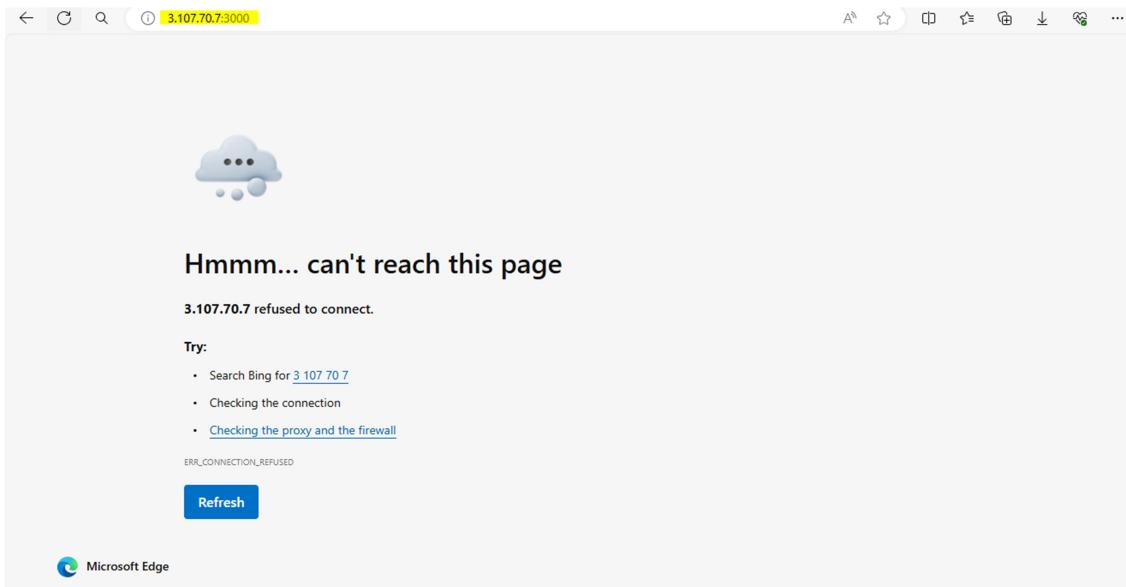
```
2.root@Hemanth:~/simple-nodejs-react-npm-app# 
Compiled successfully!

You can now view my-app in the browser.

Local:      http://localhost:3000/
On Your Network:  http://172.31.14.148:3000/

Note that the development build is not optimized.
To create a production build, use npm run build.

root@Hemanth:~/simple-nodejs-react-npm-app#
```



Permanent Run: By using '**pm2 serve <Application's Build Directory> <Port> --spa**' command run the application in the server.

A screenshot of a terminal window titled "root@Hemanth:~/simple-node-javascript-npm-app#". The window displays the following command and its output:

```
root@Hemanth:~/simple-node-javascript-npm-app# ls
Dockerfile Jenkinsfile README.md build deployment.yaml jenkins node_modules package-lock.json package.json public src
root@Hemanth:~/simple-node-javascript-npm-app# pm2 serve build 3000 --spa
[PM2] Starting /usr/local/lib/node_modules/pm2/lib/API/Serve.js in fork_mode (1 instance)
[PM2] Done
[PM2] Serving /root/simple-node-javascript-npm-app/build on port 3000
[PM2] 0 static-page-server-3000 default 5.4.0 fork 1860 0s 0 online 0% 11.3mb root
root@Hemanth:~/simple-node-javascript-npm-app#
```

The terminal shows the directory structure, then runs the pm2 serve command to start the application on port 3000 in SPA mode. It then lists the active process with ID 0, name "static-page-server-3000", and other details like namespace, version, and status.

By using '**pm2 startup**' command start the application.

```

[2 root@Hemanth:~/simple-node-] x + 
Description=PM2 process manager
Documentation=https://pm2.keymetrics.io/
After=network.target

[Service]
Type=forking
User=root
LimitNOFILE=infinity
LimitNPROC=infinity
LimitCORE=infinity
Environment=PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/snap/bin:/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
r/bin
Environment=PM2_HOME=/root/.pm2
PIDFile=/root/.pm2/pid
Restart=on-failure

ExecStart=/usr/local/lib/node_modules/pm2/bin/pm2 resurrect
ExecReload=/usr/local/lib/node_modules/pm2/bin/pm2 reload all
ExecStop=/usr/local/lib/node_modules/pm2/bin/pm2 kill

[Install]
WantedBy=multi-user.target

Target path
/etc/systemd/system/pm2-root.service
Command list
[ 'systemctl enable pm2-root' ]
[PM2] Writing init configuration in /etc/systemd/system/pm2-root.service
[PM2] Making script booting at startup...
[PM2] [--] Executing: systemctl enable pm2-root...
[PM2] [v] Command successfully executed.
+-----+
[PM2] Freeze a process list on reboot via:
$ pm2 save

[PM2] Remove init script via:
$ pm2 unstartup systemd
root@Hemanth:~/simple-node-js-react-npm-app# 

```

By using '**pm2 save -f**' command save the running programs forcefully to run the application even if user got logged out from the server and by using '**pm2 ls**' command we can check the application running status.

```

[2 root@Hemanth:~/simple-node-] x + 
Restart:on-failure

ExecStart=/usr/local/lib/node_modules/pm2/bin/pm2 resurrect
ExecReload=/usr/local/lib/node_modules/pm2/bin/pm2 reload all
ExecStop=/usr/local/lib/node_modules/pm2/bin/pm2 kill

[Install]
WantedBy=multi-user.target

Target path
/etc/systemd/system/pm2-root.service
Command list
[ 'systemctl enable pm2-root' ]
[PM2] Writing init configuration in /etc/systemd/system/pm2-root.service
[PM2] Making script booting at startup...
[PM2] [--] Executing: systemctl enable pm2-root...
[PM2] [v] Command successfully executed.
+-----+
[PM2] Freeze a process list on reboot via:
$ pm2 save

[PM2] Remove init script via:
$ pm2 unstartup systemd
root@Hemanth:~/simple-node-js-react-npm-app# pm2 save -f
[PM2] Saving current process list...
[PM2] Successfully saved in /root/.pm2/dump.pm2
root@Hemanth:~/simple-node-js-react-npm-app# pm2 ls
+-----+
| id | name | namespace | version | mode | pid | uptime | ⚡ | status | cpu | mem | user |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | static-page-server-3000 | default | 5.4.0 | fork | 1860 | 7m | 0 | online | 0% | 57.0mb | root |
+-----+
root@Hemanth:~/simple-node-js-react-npm-app# 

```

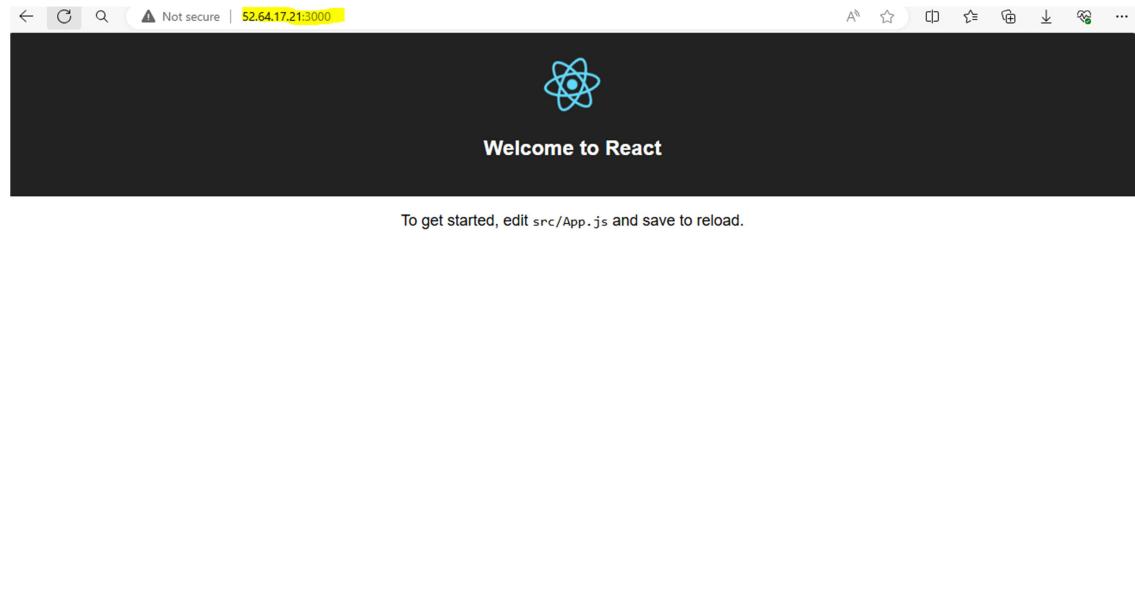
Now I am logged out from the server and even though the access point is working fine.

```
2 ./home/mobaxterm
WantedBy=multi-user.target

Target path
/etc/systemd/system/pm2-root.service
Command list
[ `systemctl enable pm2-root` ]
[PM2] Writing init configuration in /etc/systemd/system/pm2-root.service
[PM2] Making script booting at startup...
[PM2] [-] Executing: systemctl enable pm2-root...
[PM2] [v] Command successfully executed.
+-----+
[PM2] Freeze a process list on reboot via:
$ pm2 save

[PM2] Remove init script via:
$ pm2 unstartup systemd
root@Hemanth:~/simple-node-js-react-npm-app# pm2 save -f
[PM2] Saving current process list...
[PM2] Successfully saved in /root/.pm2/dump.pm2
root@Hemanth:~/simple-node-js-react-npm-app# pm2 ls
| id | name           | namespace | version | mode   | pid    | uptime | v | status | cpu   | mem   | user
| --- | ---             | ---       | ---     | ---   | ---   | ---   | --- | ---   | ---  | ---  | --- |
| 0  | static-page-server-3000 | default | 5.4.0  | fork  | 1860  | 7m    | 0 | online | 0%   | 57.0mb | root
| --- | ---             | ---       | ---     | ---   | ---   | ---   | --- | ---   | ---  | ---  | --- |
| 1  | disabled        |           |          |         |        |        |        |        |        |        |        |        |
root@Hemanth:~/simple-node-js-react-npm-app# exit
logout
ubuntu@Hemanth:~$ exit
logout
Connection to ec2-52-64-17-21.ap-southeast-2.compute.amazonaws.com closed.

29/05/2024 12:36:34 ./home/mobaxterm
```



Deployment of Backend NodeJS Applications:

Step1: Here I am using '**git clone <Github URL>**' to transfer our files into the server from Github account and saved as the Repository name which is in the Github account.

This branch is up to date with [basir/node-javascript-commerce:master](#).

Code

master 1 Branch 0 Tags

Clone

HTTPS GitHub CLI

<https://github.com/koteswararao73/node.js-application>

Clone using the web URL

Open with GitHub Desktop

Download ZIP

About

Build ECommerce Like Amazon Using Vanilla JS

[jsamazona.herokuapp.com/](#)

Readme

Activity

0 stars

0 watching

1 fork

Report repository

Releases

No releases published

```
root@Hemanth:~# git clone https://github.com/koteswararao73/node.js-application.git
Cloning into 'node.js-application'...
remote: Enumerating objects: 560, done.
remote: Counting objects: 100% (34/34), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 560 (delta 29), reused 25 (delta 25), pack-reused 526
Receiving objects: 100% (560/560), 564.29 KiB | 1.87 MiB/s, done.
Resolving deltas: 100% (364/364), done.
root@Hemanth:~# ls
Hemanth a b f1 f2 f3 node.js-application node_modules simple-node-js-react-npm-app snap
root@Hemanth:~#
```

Step2: Enter into the transferred file with ‘`cd <Directory Name>`’ command and by using ‘`npm i`’ command install the compatibilities for the application files and the compatibilities can be stored in a directory named as ‘`node_modules`’.

Note: Before using any npm, nodejs, pm2, serve tools your cursor location should be in the project or application directory.

```

root@Hemanth:~/node.js-application# cd node.js-application
root@Hemanth:~/node.js-application# npm i
npm WARN EBADENGINE Unsupported engine {
npm WARN   package: 'jsamazona@1.0.0',
npm WARN   required: { node: '12.4.0', npm: '6.9.0' },
npm WARN   current: { node: 'v18.19.1', npm: '9.2.0' }
npm WARN EBADENGINE }
npm WARN deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good
and tested way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm WARN deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm WARN deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm WARN deprecated multer@1.4.4: Multer 1.x is affected by CVE-2022-24434. This is fixed in v1.4.4-lts.1 which drops support for version
s of Node.js before 6. Please upgrade to at least Node.js 6 and version 1.4.4-lts.1 of Multer. If you need support for older versions of
Node.js, we are open to accepting patches that would fix the CVE on the main 1.x release line, whilst maintaining compatibility with Node
.js 0.10.
added 534 packages, and audited 535 packages in 33s

93 packages are looking for funding
  run npm fund for details

7 vulnerabilities (4 moderate, 3 high)

To address issues that do not require attention, run:
  npm audit fix

To address all issues possible (including breaking changes), run:
  npm audit fix --force

Some issues need review, and may require choosing
a different dependency.

Run `npm audit` for details.
root@Hemanth:~/node.js-application# ls
Procfile README.md backend frontend node_modules package-lock.json package.json uploads
root@Hemanth:~/node.js-application#

```

Step3: By using ‘**npm run build**’ command we can build the application with the existing files, which can be stored in a directory named as ‘dist’ and by using ‘**cd**’ command move your cursor to ‘dist’ directory.

```

root@Hemanth:~/node.js-application# ./node_modules/.bin/npm run build
| ./src/src eens/DashboardsScreen.js 2.09 KiB [built]
| ./src/src eens/ProductListScreen.js 2.55 KiB [built]
| + 10 hidden modules
+ 28 hidden modules

ERROR in main.js from Terser
Error: error:0308010C:digital envelope routines::unsupported
  at new Hash (node:internal/crypto/hash:69:19)
  at Object.createHash (node:crypto:183:10)
  at /root/node.js-application/frontend/node_modules/terser-webpack-plugin/dist/index.js:217:37
  at Array.forEach (<anonymous>)
  at TerserPlugin.optimizeFn (/root/node.js-application/frontend/node_modules/terser-webpack-plugin/dist/index.js:160:259)
  at AsyncSeriesHook.eval [as callAsync] (eval at create (/root/node.js-application/frontend/node_modules/tapable/lib/HookCodeFactory.j
s:33:10), <anonymous>:7:1)
  at AsyncSeriesHook.lazyCompileHook (/root/node.js-application/frontend/node_modules/tapable/lib/Hook.js:154:20)
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1409:36
  at AsyncSeriesHook.eval [as callAsync] (eval at create (/root/node.js-application/frontend/node_modules/tapable/lib/HookCodeFactory.j
s:33:10), <anonymous>:6:1)
  at AsyncSeriesHook.lazyCompileHook (/root/node.js-application/frontend/node_modules/tapable/lib/Hook.js:154:20)
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1405:32
  at AsyncSeriesHook.eval [as callAsync] (eval at create (/root/node.js-application/frontend/node_modules/tapable/lib/HookCodeFactory.j
s:33:10), <anonymous>:6:1)
  at AsyncSeriesHook.lazyCompileHook (/root/node.js-application/frontend/node_modules/tapable/lib/Hook.js:154:20)
  at Compilation.seal (/root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1342:27)
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compiler.js:675:18
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1261:4
  at AsyncSeriesHook.eval [as callAsync] (eval at create (/root/node.js-application/frontend/node_modules/tapable/lib/HookCodeFactory.j
s:33:10), <anonymous>:24:1)
  at AsyncSeriesHook.lazyCompileHook (/root/node.js-application/frontend/node_modules/tapable/lib/Hook.js:154:20)
  at Compilation.finish (/root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1253:28)
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compiler.js:672:17
  at eval (eval at create (/root/node.js-application/frontend/node_modules/tapable/lib/HookCodeFactory.js:33:10), <anonymous>:11:1)
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1185:12
  at /root/node.js-application/frontend/node_modules/webpack/lib/Compilation.js:1097:9
  at process.processTicksAndRejections (node:internal/process/task_queues:77:11)
root@Hemanth:~/node.js-application# ls
Procfile README.md backend dist frontend node_modules package-lock.json package.json uploads
root@Hemanth:~/node.js-application#

```

Step4: Now we have to allow the application port in Security Groups.

Go to Instances page and select your server, under Security section click on the link under Security groups.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instances Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity, Reservations (New), Images, AMIs, AMI Catalog, and Elastic Block Store (Volumes). The main area displays 'Instances (1/2) Info' with a search bar and filters for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. Two instances are listed: 'Practice 1' (i-006dbe156c061cf25, Running, t2.micro, 2/2 checks passed) and 'Hemanth' (i-0788726e119741745, Stopped, t2.micro). Below the table, a detailed view for 'i-006dbe156c061cf25 (Practice 1)' is shown, including Security details (IAM Role: -, Owner ID: 851725491699), Launch time (Sat May 25 2024 13:59:05 GMT+0530 (India Standard Time)), and Security groups (sg-00c3c9865c90bcb63 (launch-wizard-2)).

Under Inbound rules section click on edit inbound rules.

The screenshot shows the AWS Security Groups page for the security group 'sg-00c3c9865c90bcb63 - launch-wizard-2'. The left sidebar is identical to the previous screenshot. The main area shows the 'Details' section with fields: Security group name (launch-wizard-2), Security group ID (sg-00c3c9865c90bcb63), Description (launch-wizard-2 created 2024-05-16T08:26:41.911Z), Owner (851725491699), Inbound rules count (1 Permission entry), and Outbound rules count (1 Permission entry). Below this, the 'Inbound rules (1)' section is visible, featuring a search bar and a button labeled 'Edit inbound rules'.

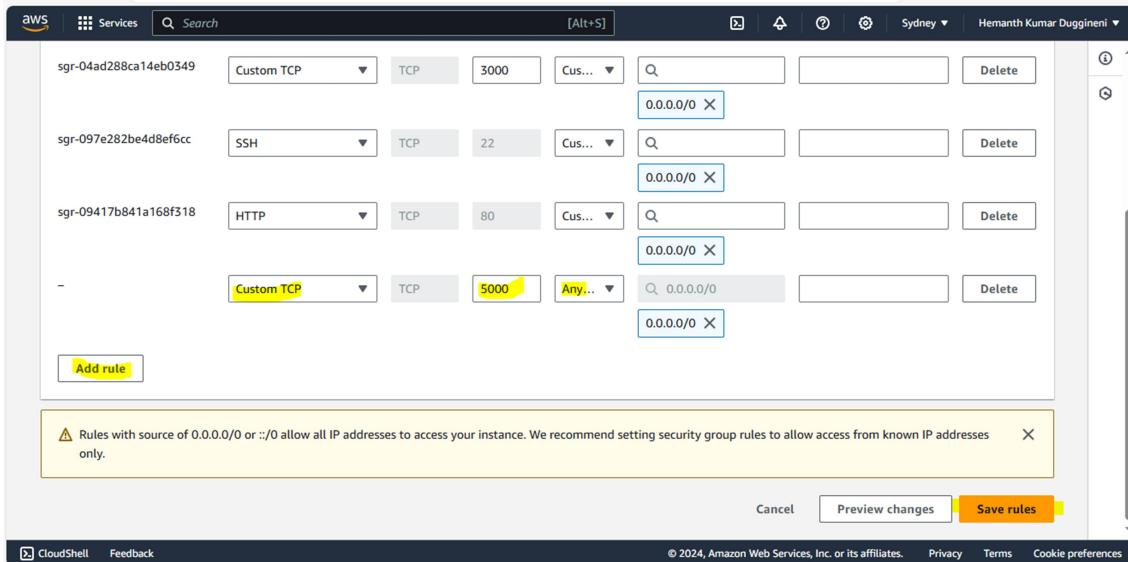
Click on Add rule and select

Type: Custom TCP (required protocol type)

Port Range: Required Port number

Source: Anywhere-IPv4 (required access network IP addresses)

And click on save rules.



Up to here deployment of application is done.

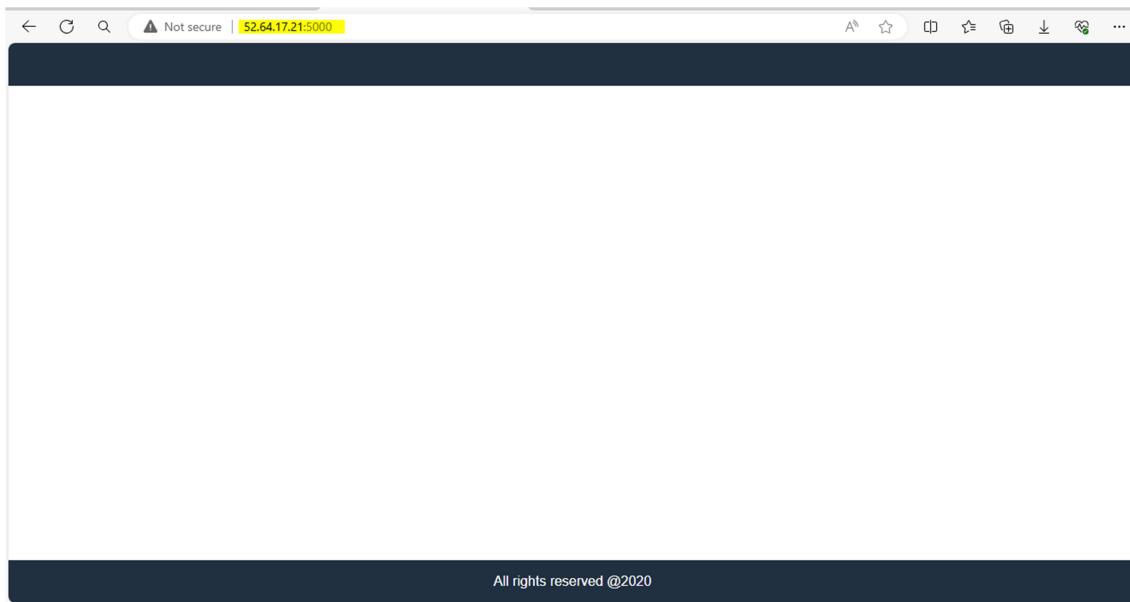
To run the deployed application there are two ways:

Temporary Run: By using '**npm start**' we can run our application temporarily up to we click on 'ctrl+c' in the server.

After passing the command '**npm start**' the application will start running with the access point.

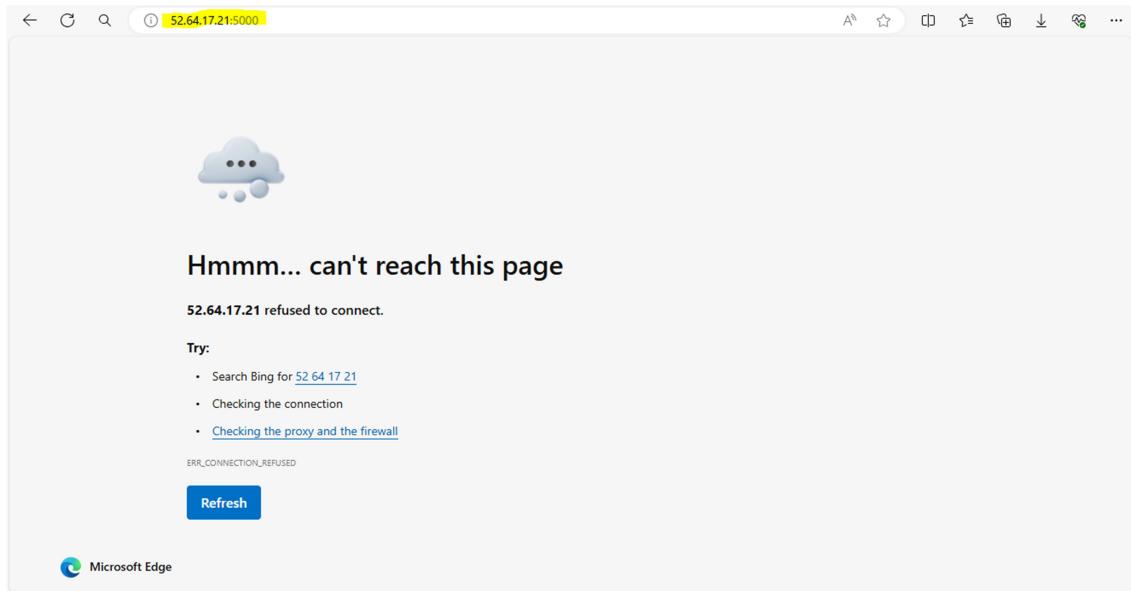
A screenshot of a terminal window titled 'root@Hemanth:~/node.js-application#'. The command 'npm start' is entered and executed. The output shows the nodemon process starting, watching files in the 'backend' directory, and serving the application at 'http://localhost:5000'.

```
root@Hemanth:~/node.js-application# npm start
> jsamazona@1.0.0 start
> nodemon --watch backend --exec babel-node backend/server.js
[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): backend/**/*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `babel-node backend/server.js`
serve at http://localhost:5000
undefined
```



After entering 'ctrl+c', the application will stop running with access point.

```
root@Hemanth:~/node.js-application/dist# npm start
> jsamazona@1.0.0 start
> nodemon --watch backend --exec babel-node backend/server.js
[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): backend/**/*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `babel-node backend/server.js`
serve at http://localhost:5000
undefined
root@Hemanth:~/node.js-application/dist#
```



Permanent Run: By using '**pm2 start <Application's server.js file>**' command start and run the application in the server. By using '**pm2 save -f**' command save the running programs forcefully to run the application even if user got logged out from the server.

```
root@Hemanth:~/node.js-application/dist# pm2 start server.js
[PM2] Starting /root/node.js-application/dist/server.js in fork_mode (1 instance)
[PM2] Done.

| id | name      | namespace | version | mode   | pid    | uptime | ↻ | status  | cpu   | mem   | user
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | server    | default   | 1.0.0   | fork   | 3166  | 0s    | 0 | online | 0%   | 12.1mb | root
| 0 | static-page-server-3000 | default   | 5.4.0   | fork   | 2870  | 30m   | 0 | online | 0%   | 60.4mb | root
|---|---|---|---|---|---|---|---|---|---|---|---|
```

[PM2][WARN] Current process list is not synchronized with saved list. Type 'pm2 save' to synchronize.
root@Hemanth:~/node.js-application/dist# pm2 save -f
[PM2] Saving current process list...
[PM2] Successfully saved in /root/.pm2/dump.pm2
root@Hemanth:~/node.js-application/dist#

Now I am logged out from the server and even though the access point is working fine.

```
root@Hemanth:~/node.js-application/dist# pm2 start server.js
[PM2] Starting /root/node.js-application/dist/server.js in fork_mode (1 instance)
[PM2] Done.

| id | name           | namespace | version | mode   | pid    | uptime | ⚡ | status  | cpu   | mem   | user
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | server         | default   | 1.0.0   | fork   | 3166  | 0s    | 0 | online | 0%   | 12.1mb | root
| 0 | static-page-server-3000 | default   | 5.4.0   | fork   | 2870  | 30m   | 0 | online | 0%   | 60.4mb | root
|---|---|---|---|---|---|---|---|---|---|---|---|
[PM2][WARN] Current process list is not synchronized with saved list. Type 'pm2 save' to synchronize.
root@Hemanth:~/node.js-application/dist# pm2 save -f
[PM2] Saving current process list...
[PM2] Successfully saved in /root/.pm2/dump.pm2
root@Hemanth:~/node.js-application/dist# exit
logout
ubuntu@Hemanth:~$ exit
logout
Connection to ec2-52-64-17-21.ap-southeast-2.compute.amazonaws.com closed.
```

