**Configure and deploy below application  on linux server (3 tier application)**

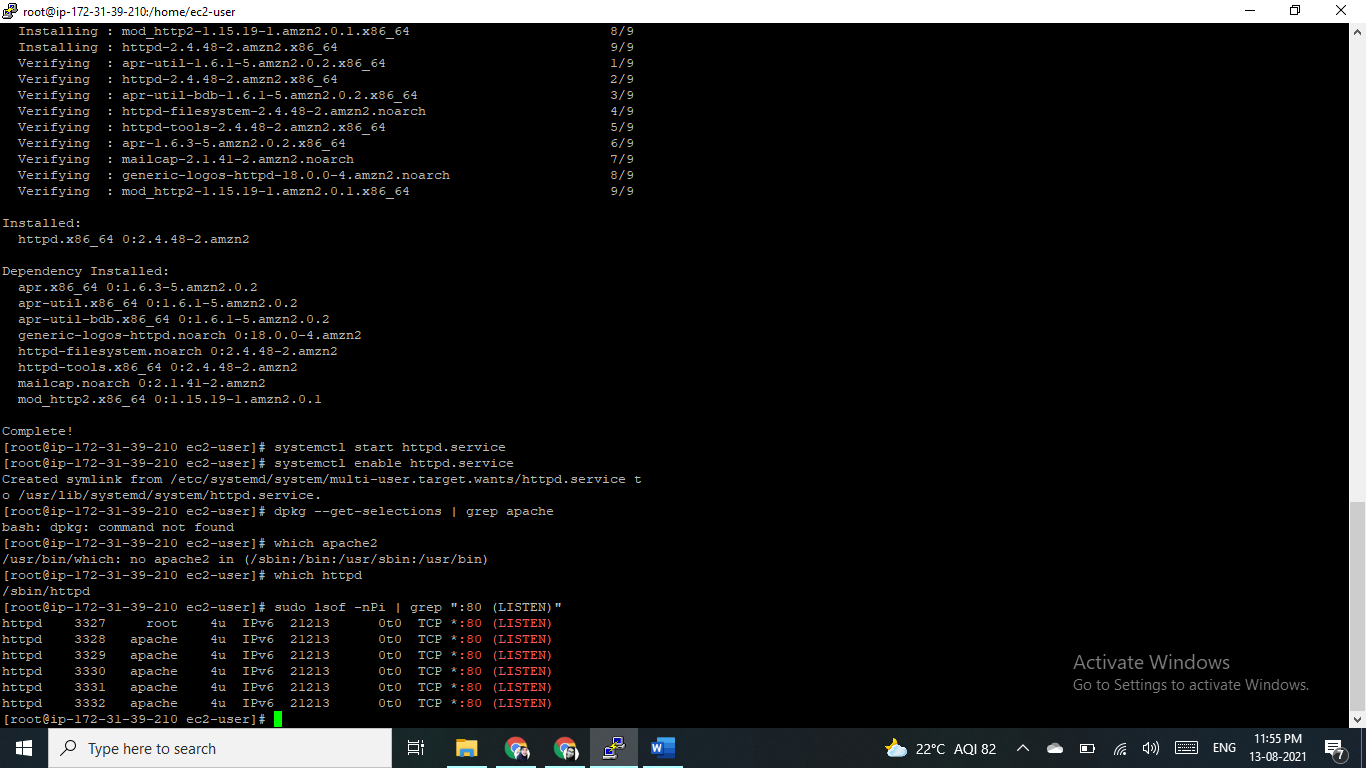
**Frontend server( apache webserver)**

Yum install -y httpd.x86\_64

[root@ip-172-31-39-210 ec2-user]# systemctl start httpd.service

[root@ip-172-31-39-210 ec2-user]# systemctl enable httpd.service

Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service t o /usr/lib/systemd/system/httpd.service.



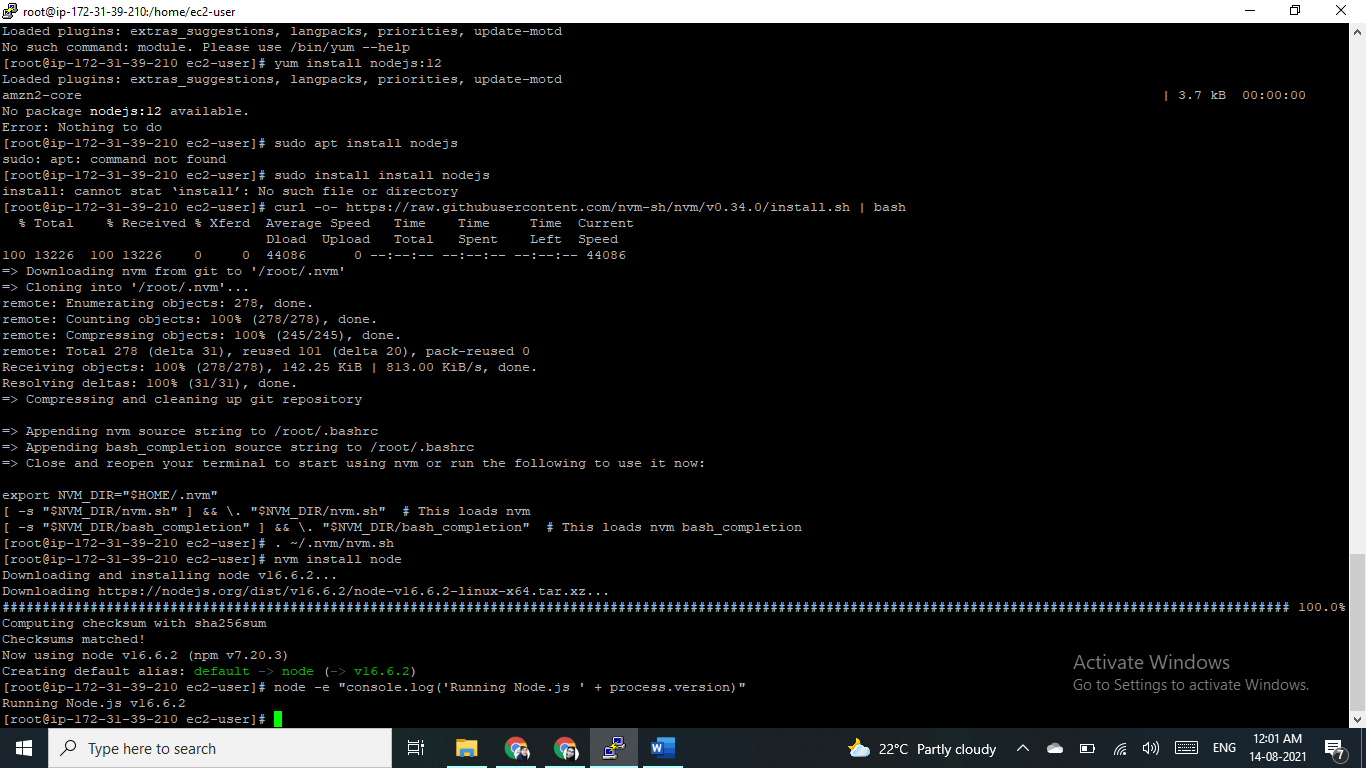
**Backend application (Node app)**

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash

. ~/.nvm/nvm.sh

nvm install node

node -e "console.log('Running Node.js ' + process.version)"



**Database (mongo)**

To check which machine I used.

[root@ip-172-31-39-210 ec2-user]# grep ^NAME /etc/\*release

/etc/os-release:NAME="Amazon Linux"

Install MongoDB Community Edition on Amazon Linux

Verify Linux Distribution

You can verify which Linux distribution you are running by running the following command on the command-line:

grep ^NAME /etc/\*release

### Configure the package management system (yum).

Create a /etc/yum.repos.d/mongodb-org-5.0.repo file so that you can install MongoDB directly using yum:

Nano /etc/yum.repos.d/mongodb-org-5.0.repo  (open with nano editor)

And the paste some line of code there, which is…

[mongodb-org-5.0]

name=MongoDB Repository

baseurl=https://repo.mongodb.org/yum/amazon/2/mongodb-org/5.0/x86\_64/

gpgcheck=1

enabled=1

gpgkey=https://www.mongodb.org/static/pgp/server-5.0.asc

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Install the MongoDB packages.

sudo yum install -y mongodb-org

Alternatively, to install a specific release of MongoDB, specify each component package individually and append the version number to the package name, as in the following example:

sudo yum install -y mongodb-org-5.0.2 mongodb-org-database-5.0.2 mongodb-org-server-5.0.2 mongodb-org-shell-5.0.2 mongodb-org-mongos-5.0.2 mongodb-org-tools-5.0.2

You can specify any available version of MongoDB. However yum upgrades the packages when a newer version becomes available. To prevent unintended upgrades, pin the package. To pin a package, add the following exclude directive to your /etc/yum.conf file:

exclude=mongodb-org,mongodb-org-database,mongodb-org-server,mongodb-org-shell,mongodb-org-mongos,mongodb-org-tools

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **Run MongoDB Community Edition**

If you are unsure which init system your platform uses, run the following command:

ps --no-headers -o comm 1

I am using : **systemd (systemctl)**

Then select the appropriate tab below based on the result:

* systemd - select the **systemd (systemctl)** tab below.
* init - select the **System V Init (service)** tab below.

You can start the [mongod](https://docs.mongodb.com/manual/reference/program/mongod/" \l "mongodb-binary-bin.mongod) process by issuing the following command

sudo systemctl start mongod

#### Verify that MongoDB has started successfully

sudo systemctl status mongod

You can optionally ensure that MongoDB will start following a system reboot by issuing the following command

sudo systemctl enable mongod

#### Stop MongoDB

As needed, you can stop the [mongod](https://docs.mongodb.com/manual/reference/program/mongod/" \l "mongodb-binary-bin.mongod) process by issuing the following command:

sudo systemctl stop mongod

#### Restart MongoDB

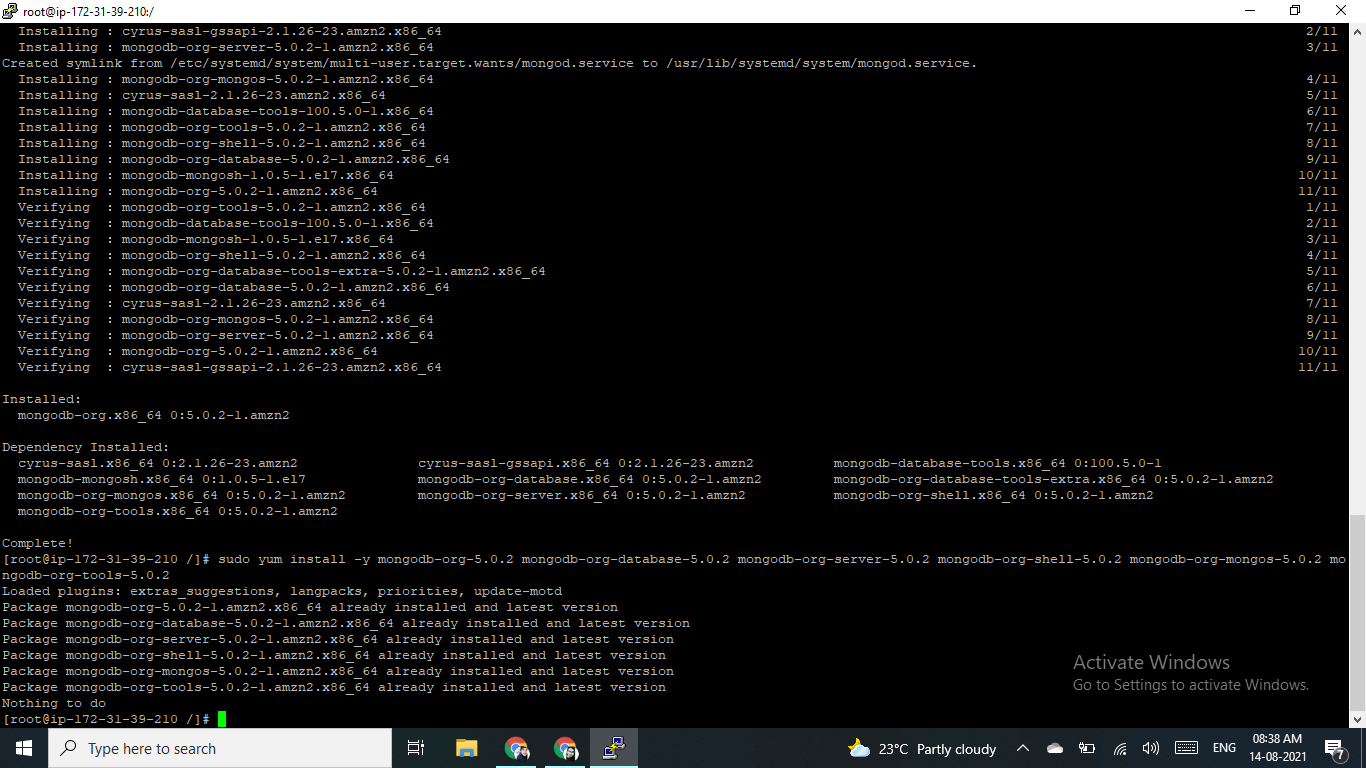
You can restart the [mongod](https://docs.mongodb.com/manual/reference/program/mongod/" \l "mongodb-binary-bin.mongod) process by issuing the following command

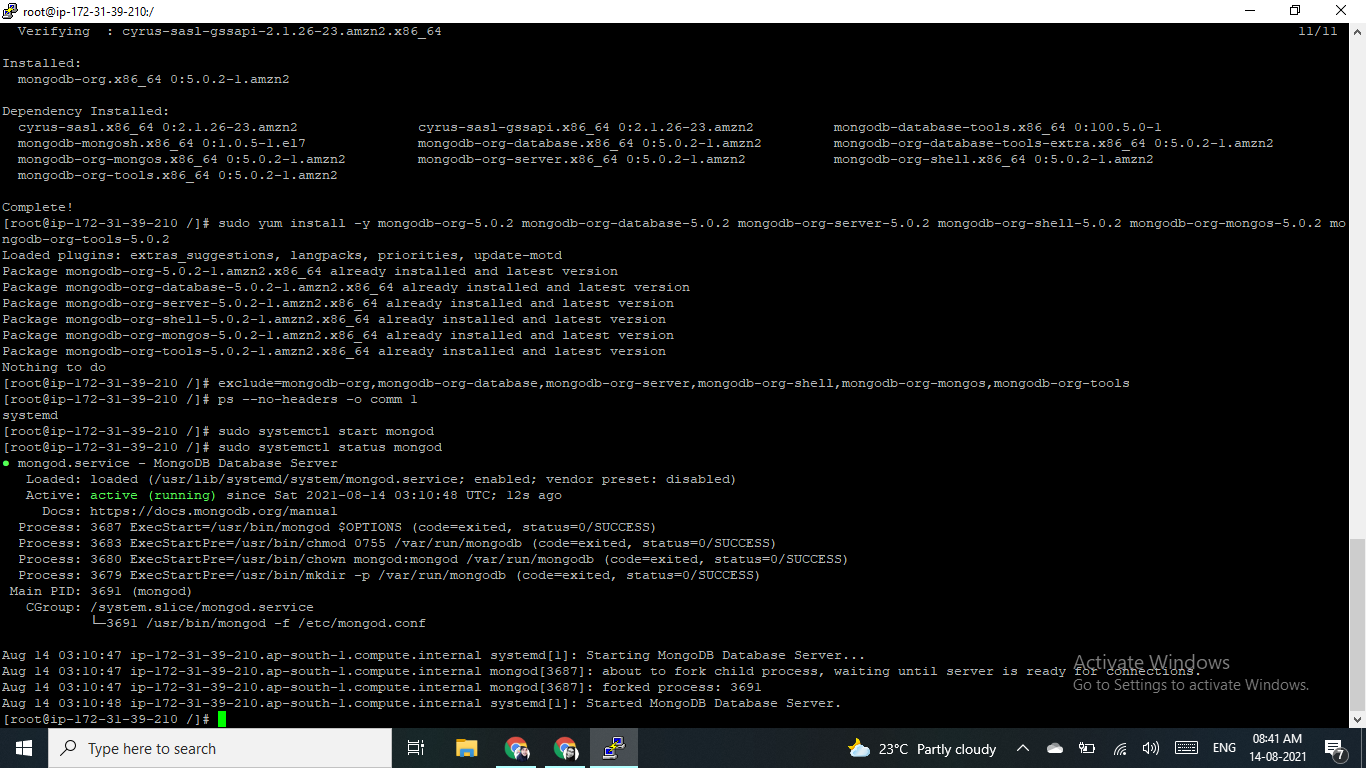
sudo systemctl restart mongod

#### Begin using MongoDB

Start a [mongosh](https://docs.mongodb.com/mongodb-shell/" \l "mongodb-binary-bin.mongosh) session on the same host machine as the [mongod](https://docs.mongodb.com/manual/reference/program/mongod/" \l "mongodb-binary-bin.mongod). You can run [mongosh](https://docs.mongodb.com/mongodb-shell/" \l "mongodb-binary-bin.mongosh) without any command-line options to connect to a [mongod](https://docs.mongodb.com/manual/reference/program/mongod/" \l "mongodb-binary-bin.mongod) that is running on your localhost with default port 27017.

Mongosh





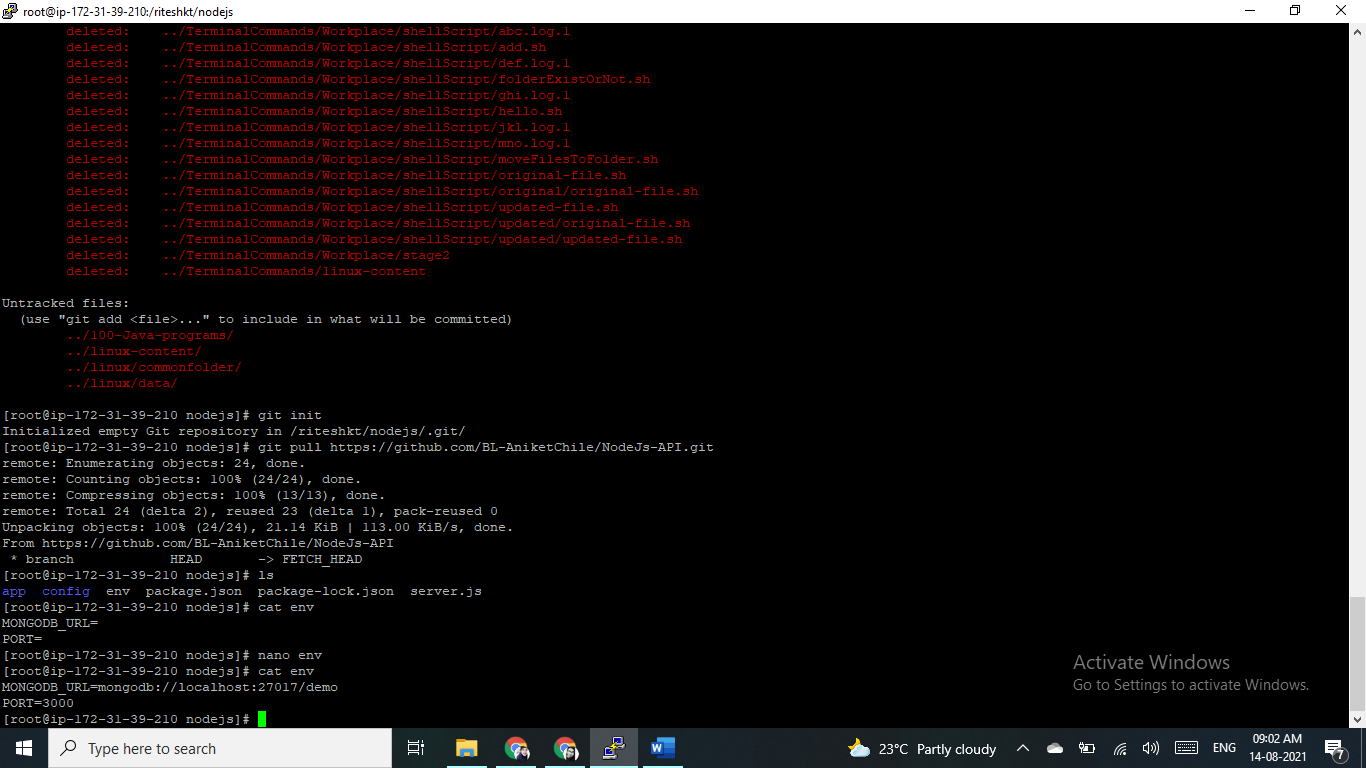
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Create .env file in you project to pass environment =>

MONGODB\_URL=mongodb://localhost:27017/demo

PORT=3000

Application source - <https://github.com/BL-AniketChile/NodeJs-API>



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

To start node server.js

[root@ip-172-31-39-210 nodejs]# node server.js

Server is running on port: 3000

MongoDB connection established successfully!!!

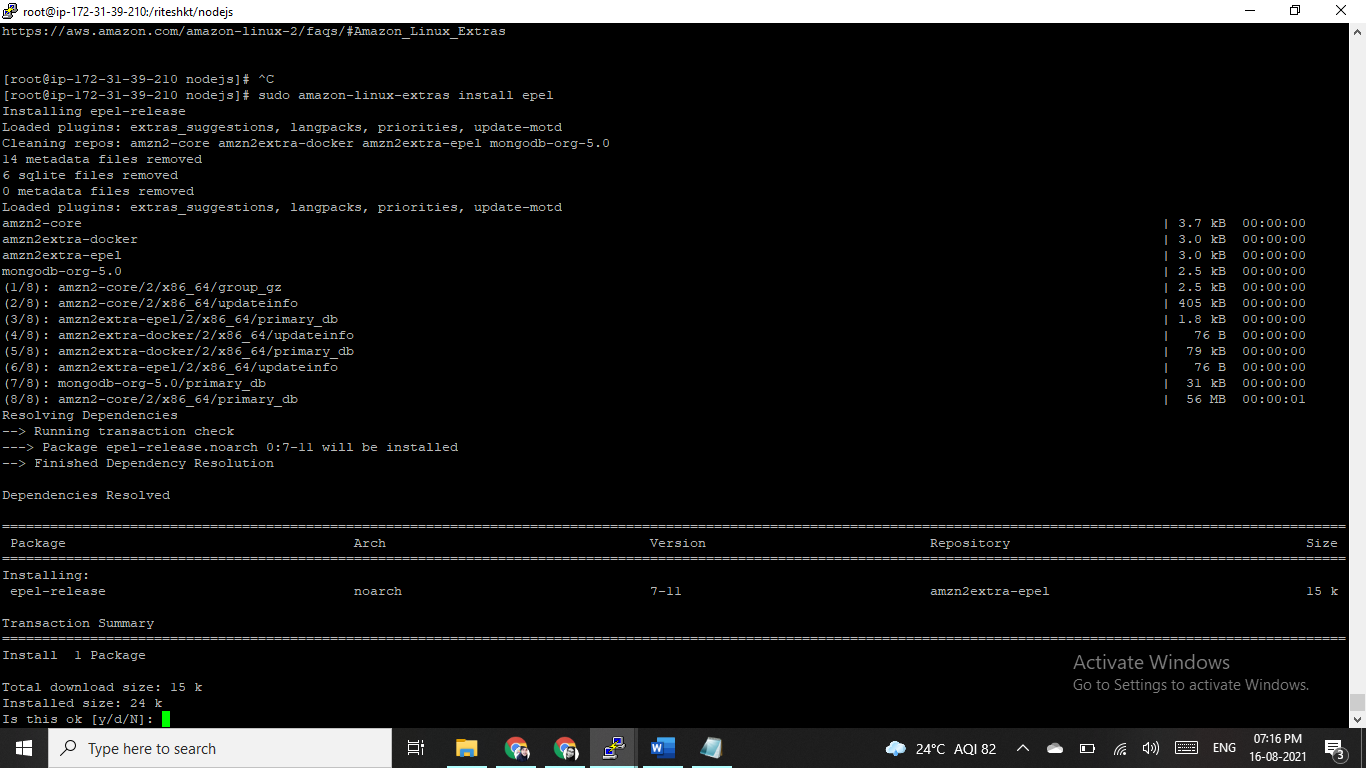
**How to Install and Use UFW Firewall on Linux**

UFW also known as Uncomplicated Firewall is an interface to iptables and is particularly well-suited for host-based firewalls. UFW provide an easy to use interface for beginner user who is unfamiliar with firewall concepts. It is most popular firewall tool originating from Ubuntu. It supports both IPv4 and IPv6.

**CentOS**

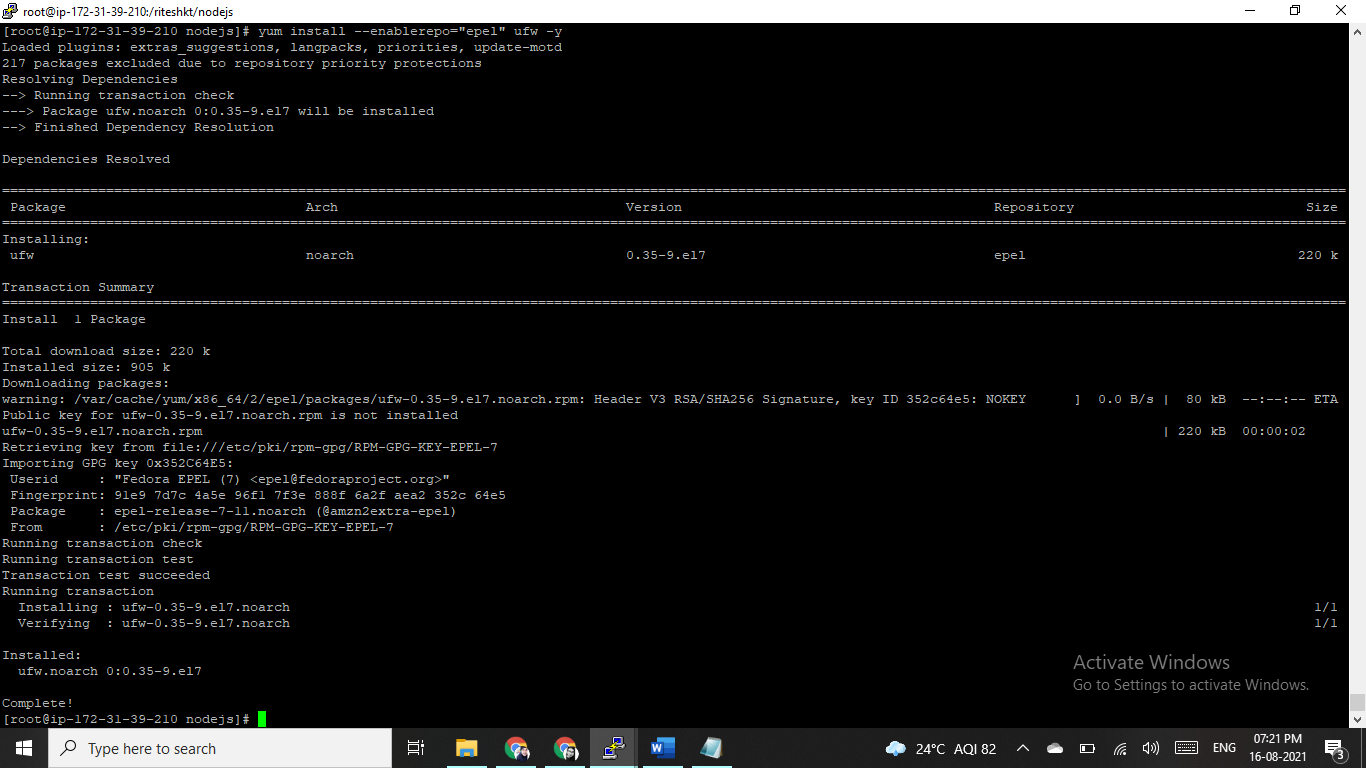
By default, UFW is not available in CentOS repository. So you will need to install the EPEL repository to your system. You can do this by running the following [linux command](https://linuxconfig.org/linux-commands" \t "_blank):

# sudo amazon-linux-extras install epel



Once the EPEL repository is installed, you can install UFW by just running the following linux command:

# yum install --enablerepo="epel" ufw -y



After installing UFW, start UFW service and enable it to start on boot time by running the following [linux command](https://linuxconfig.org/linux-commands" \t "_blank).

# ufw enable

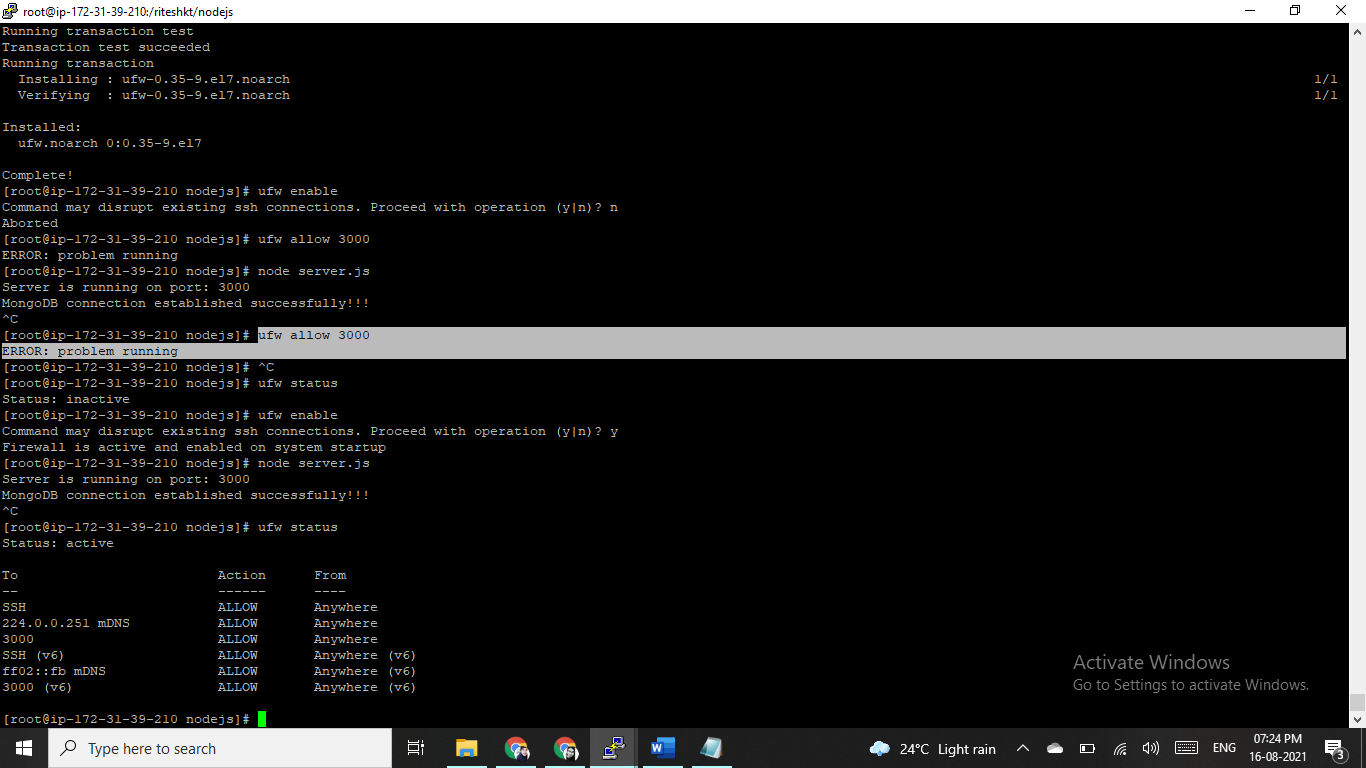
Next, check the status of UFW with the following linux command. You should see the following output:

# ufw status

Status: active

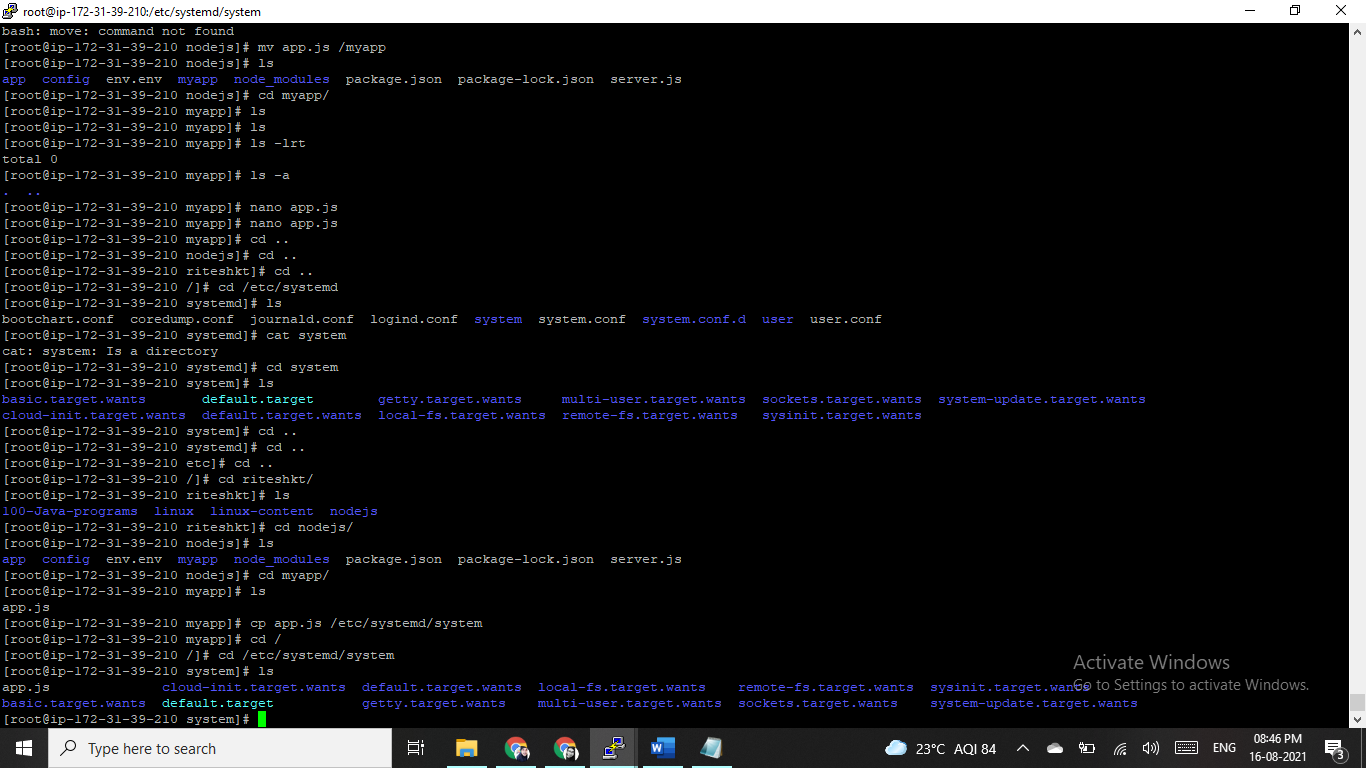
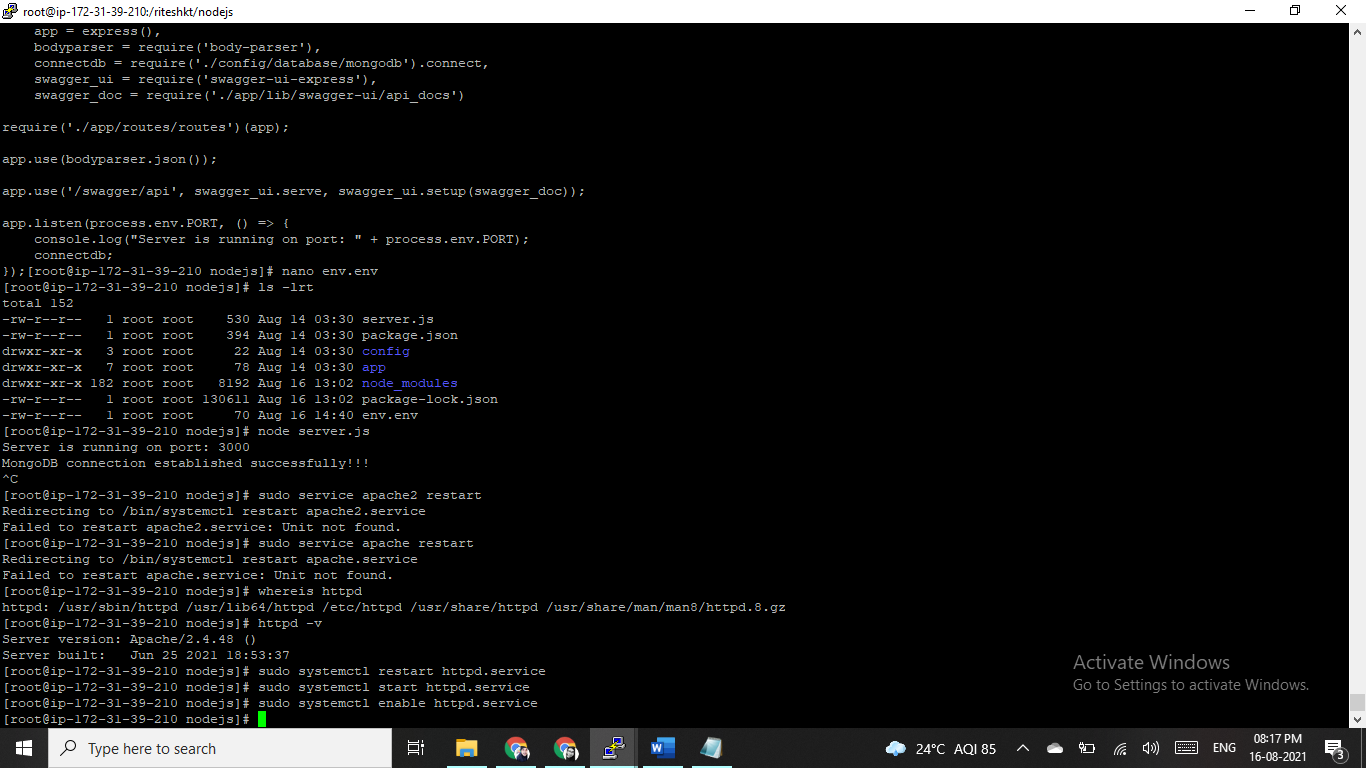
We can also disable UFW firewall by running the following linux command:

# ufw disable



create background process for node applicatoin using systemctl configuration

#systemctl start myapp instead of "node server.js"



Make a myapp.service file

[Unit]

Description=My app

[Service]

ExecStart=/var/www/myapp/app.js

Restart=always

User=nobody

# Note Debian/Ubuntu uses 'nogroup', RHEL/Fedora uses 'nobody'

Group=nogroup

Environment=PATH=/usr/bin:/usr/local/bin

Environment=NODE\_ENV=production

WorkingDirectory=/var/www/myapp

[Install]

WantedBy=multi-user.target

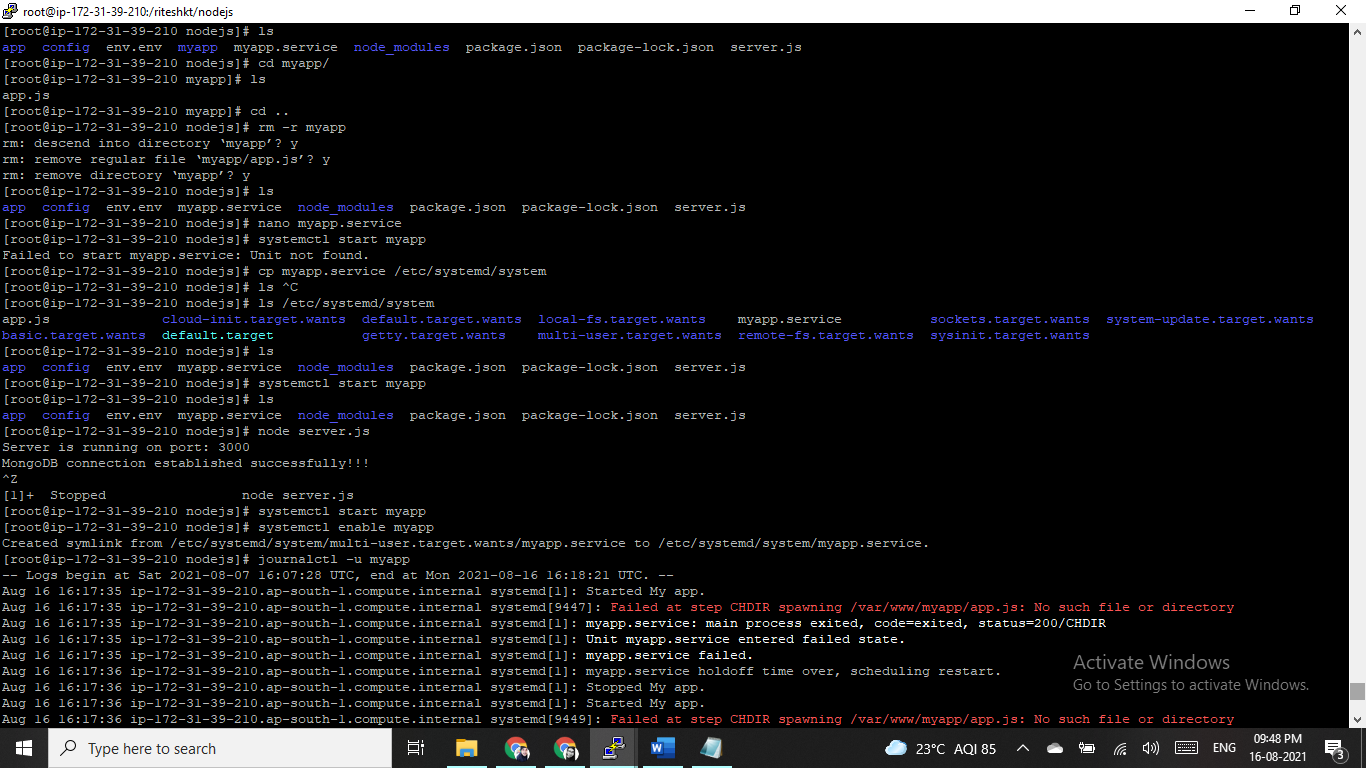
**Note if you're new to Unix:** /var/www/myapp/app.js should have #!/usr/bin/env node on the very first line and have the executable mode turned on chmod +x myapp.js.

Copy your service file into the /etc/systemd/system.

Start it with systemctl start myapp.

Enable it to run on boot with systemctl enable myapp.

See logs with journalctl -u myapp



Apache Reverse Proxy configuration: Step by Step

configure apache - add proxypass for your application.

A reverse proxy is a kind of proxy server that takes HTTP or HTTPS requests & transfers/distributes them to one or more backend servers.

A reverse proxy is a type of proxy server that takes HTTP(S) requests and transparently distributes them to one or more backend servers. Reverse proxies are useful because many modern web applications process incoming HTTP requests using backend application servers which aren’t meant to be accessed by users directly and often only support rudimentary HTTP features.

All four modules are enabled by default on a fresh CentOS 7 installation. You can verify that they are enabled by running:

* httpd -M

In case the modules are not enabled, you can enable them by opening the and uncommenting lines with necessary modules by removing # sign from the line beginnings so the file looks as follows:

sudo nano /etc/httpd/conf.modules.d/00-proxy.conf

## **Step 3 — Modifying the Default Configuration to Enable Reverse Proxy**

Create the new default virtual host by creating a new empty Apache configuration file in /etc/httpd/conf.d directory using nano or your favorite text editor.

sudo nano /etc/httpd/conf.d/default-site.conf

### **Example 1 — Reverse Proxying a Single Backend Server**

Paste the following contents into the default-site.conf file, so your configuration file looks like this:

<VirtualHost \*:80>

ProxyPreserveHost On

ProxyPass / http://127.0.0.1:8080/

ProxyPassReverse / http://127.0.0.1:8080/

</VirtualHost>

ProxyPreserveHost makes Apache pass the original Host header to the backend server. This is useful, as it makes the backend server aware of the address used to access the application.

ProxyPass is the main proxy configuration directive. In this case, it specifies that everything under the root URL (/) should be mapped to the backend server at the given address. For example, if Apache gets a request for /example, it will connect to http://your\_backend\_server/example and return the response to the original client.

ProxyPassReverse should have the same configuration as ProxyPass. It tells Apache to modify the response headers from backend server. This makes sure that if the backend server returns a location redirect header, the client’s browser will be redirected to the proxy address and not the backend server address, which would not work as intended.

