

# Nodejs application using docker file

Introduction :

This document provides a guide for implementing a Docker container for the Node.js Chat App using the provided Dockerfile. The Dockerfile defines the environment and dependencies required to run the application within a Docker container.

Prerequisites:

- Docker installed on the host machine.
- Internet connectivity to download dependencies during the Docker build process.

## Step-1

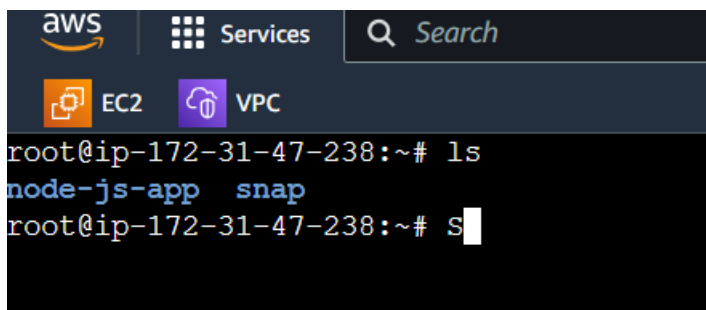
Login through aws console , start your ubuntu instance & install docker in it

Connect ec2 instance

## Step-2

Create directory as nodejsapp

#mkdir nodejsapp

A screenshot of the AWS Management Console. The top navigation bar shows the AWS logo, 'Services' menu, and a search bar. Below the navigation bar, there are icons for 'EC2' and 'VPC'. The main content area shows a terminal window for an EC2 instance. The terminal prompt is 'root@ip-172-31-47-238:~#'. The user has entered 'ls' and the output is 'node-js-app snap'. The user has then entered 'S' and the cursor is at the end of the line.

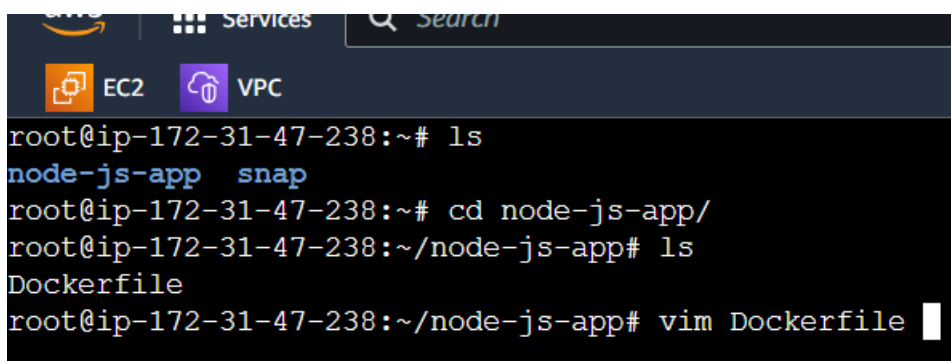
```
aws | Services | Search
EC2 | VPC
root@ip-172-31-47-238:~# ls
node-js-app snap
root@ip-172-31-47-238:~# S
```

## Step-3

Create Dockerfile in that directory with vim command

#cd nodejsapp

#vim Dockerfile

A screenshot of the AWS Management Console, similar to the previous one. The terminal window shows the user navigating to the 'node-js-app' directory and creating a 'Dockerfile'. The terminal prompt is 'root@ip-172-31-47-238:~#'. The user enters 'ls' and the output is 'node-js-app snap'. The user then enters 'cd node-js-app/' and the prompt changes to 'root@ip-172-31-47-238:~/node-js-app#'. The user enters 'ls' and the output is 'Dockerfile'. Finally, the user enters 'vim Dockerfile' and the cursor is at the end of the line.

```
aws | Services | Search
EC2 | VPC
root@ip-172-31-47-238:~# ls
node-js-app snap
root@ip-172-31-47-238:~# cd node-js-app/
root@ip-172-31-47-238:~/node-js-app# ls
Dockerfile
root@ip-172-31-47-238:~/node-js-app# vim Dockerfile
```

### Step-4

Paste following content inside the Docker file

FROM ubuntu:latest

LABEL app="nodejs"

**Label** Author="divya"

RUN apt update

RUN apt install nodejs npm -y

RUN `git clone https://github.com/owanhunte/nodejs-chat-app.git`

WORKDIR nodejs-chat-app

RUN npm install

EXPOSE 3000

```
CMD [ "npm", "start" ]
```

[illegible]

## Step-5

Enter command “docker build .” to create an docker image

```
root@ip-172-31-47-238:~/node-js-app# docker build .
[+] Building 1.4s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 289B
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/6] FROM docker.io/library/ubuntu:latest@sha256:77906da86b60585ce12215807090eb327e7386c8fafb5402369e421f44eff17e
=> CACHED [2/6] RUN apt update
=> CACHED [3/6] RUN apt install nodejs npm -y
=> CACHED [4/6] RUN git clone https://github.com/owanhunte/nodejs-chat-app.git
=> CACHED [5/6] WORKDIR nodejs-chat-app
=> CACHED [6/6] RUN npm install
=> exporting to image
=> => exporting layers
=> => writing image sha256:9e7b533eff082ba03b4f9d924fe4e8031fb9876c5ba5558fdc9365b4605905b6
root@ip-172-31-47-238:~/node-js-app#
```

## Step-6

List the images by using command “docker images”

```
root@ip-172-31-47-238:~/node-js-app# docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
<none>          <none>       9e7b533eff08  5 hours ago   964MB
root@ip-172-31-47-238:~/node-js-app#
```

## Step-7

Create container with the help of docker image

#docker run -d -p 3000:3000

```
root@ip-172-31-47-238:~/node-js-app# docker run -d -p 3000:3000 9e7b
7630ced847dc8108bf688ac056e1130b604542e69ae7efede6d25074c2a12bd6
root@ip-172-31-47-238:~/node-js-app# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
7630ced847dc   9e7b     "npm start"              9 seconds ago Up 8 seconds  0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   competent_mclaren
root@ip-172-31-47-238:~/node-js-app#
```

## Step-8

Check the application in browser

http//<IP of instance>:3000

