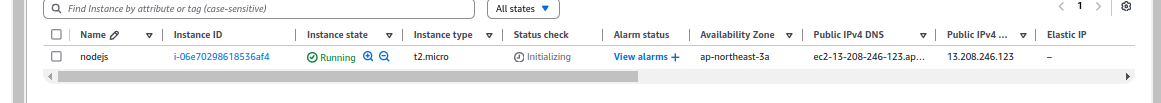
**Create a docker image for any application using Docker file and push it to Docker hub**

1. **Create a Instance**

**—-----------------------------------**

****

1. **Install nginx, Nodejs, Npm, Pm2**

**—-----------------------------------------------------------**

→ Login in EC2

**apt update -y**

→ Install Nginx webserver

**apt install nginx -y**



→ Install Nodejs

**apt install nodejs -y**



→ Install NPM (node package manager)

**apt install npm -y**

****

→ Install Pm2 (process manager)

**npm install -g pm2**

****

1. **Creating a Nodejs Application**

**—--------------------------------------------------------**

**→** First, using nano or your favorite text editor, create a sample application called hello.js inside the home directory

→ Go to the home directory

**cd /home**

→open a hello.js file and paste the code

**nano hello.js**

**const http = require('http');**

**const hostname = '0.0.0.0';**

**const port = 3000;**

**const server = http.createServer((req, res) => {**

**res.statusCode = 200;**

**res.setHeader('Content-Type', 'text/plain');**

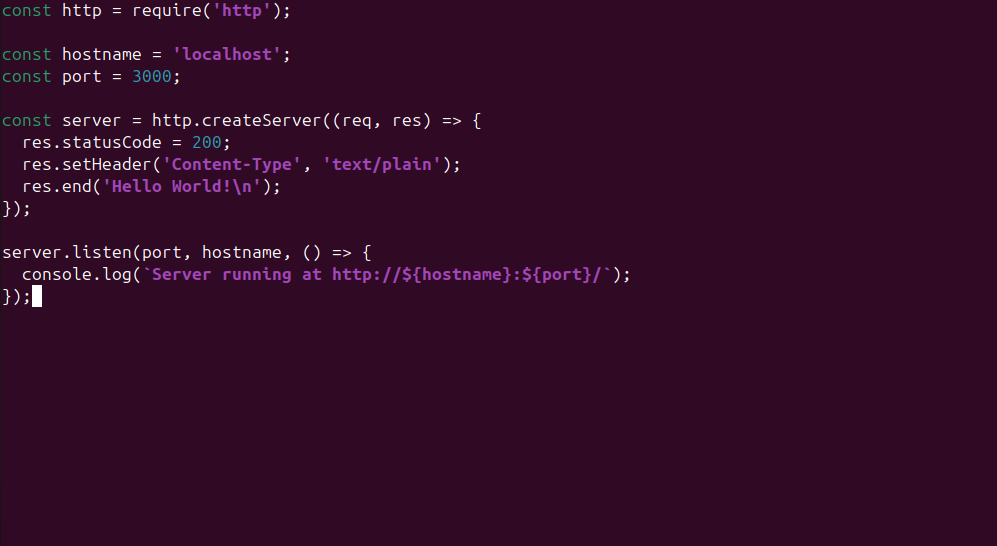
**res.end('Hello World!\n');**

**});**

**server.listen(port, hostname, () => {**

**console.log(`Server running at http://${hostname}:${port}/`);**

**});**

****

Save the file and exit from the editor

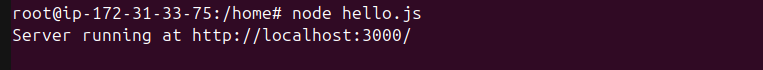
Ctrl + o = saving

Ctrl + x = exit

→ This Node.js application listens on the specified address **(0.0.0.0)** and **port (3000)**, and returns **“Hello World!”** with a **200 HTTP** success code. Since we’re listening on localhost, remote clients won’t be able to connect to our application.

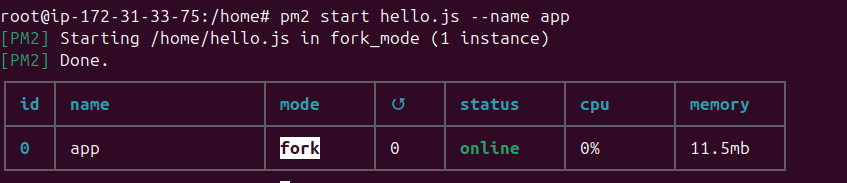
To test your application, type:

**node hello.js**



→ Going to start node application in pm2

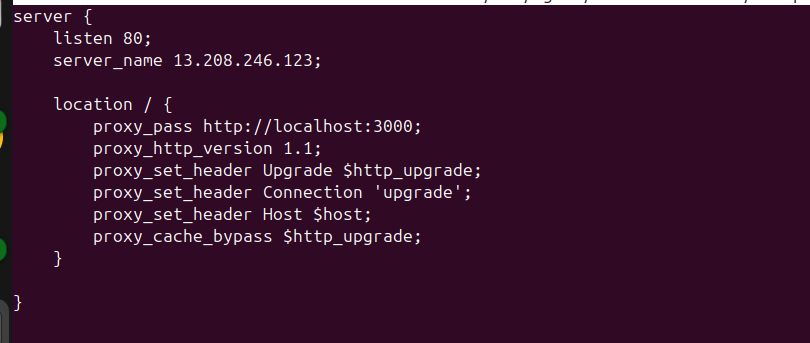
**pm2 start hello.js --name app**



→ Setting up node application in nginx as a reverse proxy

To let Your application is **running and listening on localhost**, but you need to set up a way for your users to access it. We will set up the **Nginx web server as a reverse proxy** for this purpose.

In the prerequisite tutorial, you set up your **Nginx** configuration in the **/etc/nginx/sites-available/example.com** file. Open this file for editing



**server {**

**listen 80;**

**server\_name 13.208.246.123;**

**location / {**

**proxy\_pass http://localhost:3000;**

**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;**

**}**

**}**

(replace with your server ip)

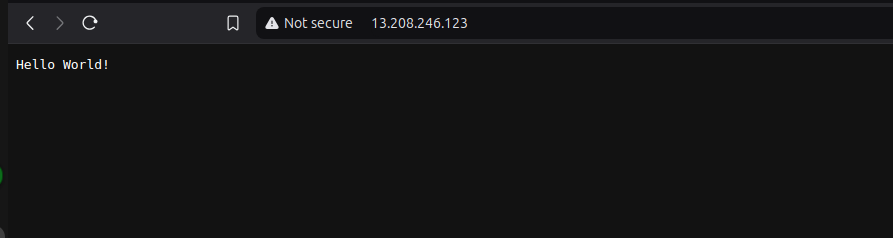
→ Create a symink for this example.com

**ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabled/**

→ Restart your nginx webserver

**systemctl restart nginx**

→ Check your application in browser using IP



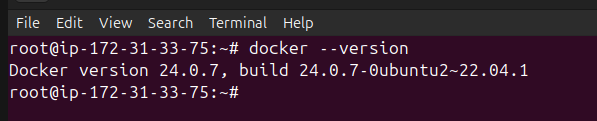
Nodejs application is Running on Ec2 instance

1. **Settting up in Docker**

**—------------------------------------**

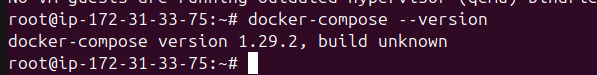
→ Install docker

**apt install -y docker.io**

****

→ Install Docker Compose

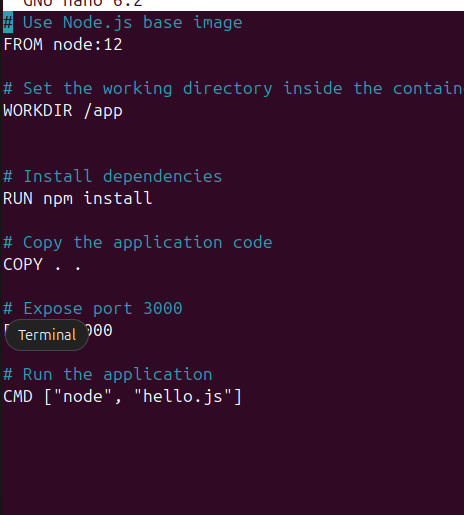
**apt install -y docker-compose**

****

→ Navigate to the application directory

**cd /home/node**

→ Create a Dockerfile in the /home/node directory



**# Use Node.js base image**

**FROM node:12**

**# Set the working directory inside the container**

**WORKDIR /app**

**# Install dependencies**

**RUN npm install**

**# Copy the application code**

**COPY . .**

**# Expose port 3000**

**EXPOSE 3000**

**# Run the application**

**CMD ["node", "hello.js"]**

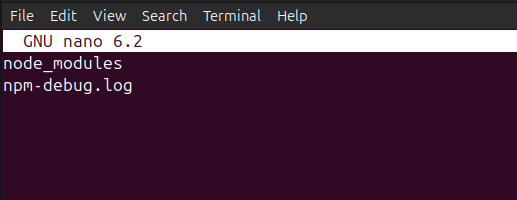
Save it and exit from editor

→ Create a .dockerignore File

Create a .dockerignore file to exclude unnecessary files from the image

**node\_modules**

**npm-debug.log**

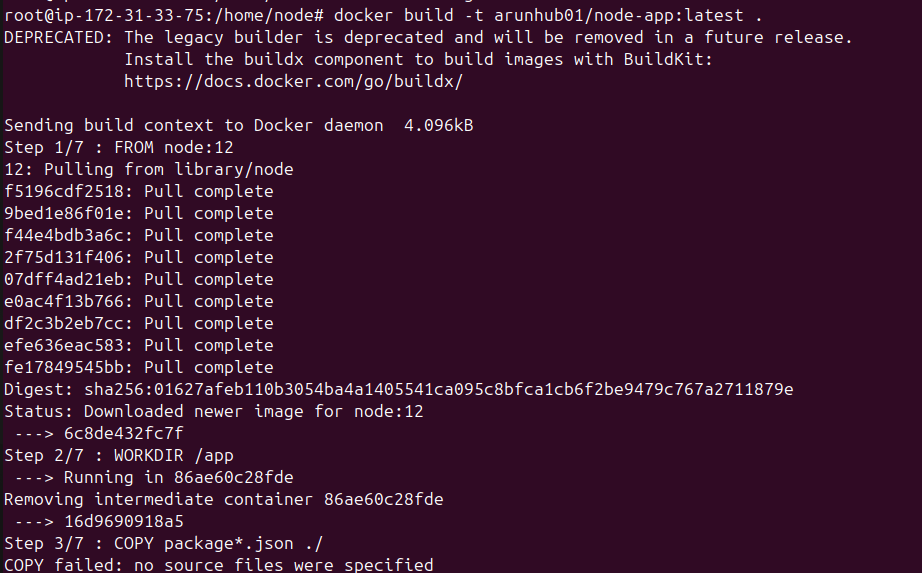


Save it and exit from editor

→ Build the docker image

**docker build -t arunhub01/node-app:latest .**

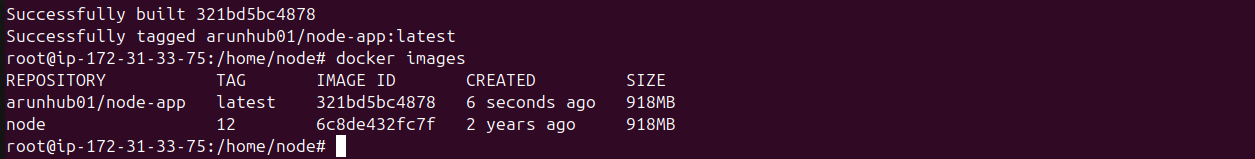
(Here arunhub01 is my dockerhub username. Replace with your dockerhub username)



Docker build the image is completed

→ Verify the image is build or not

**docker images**

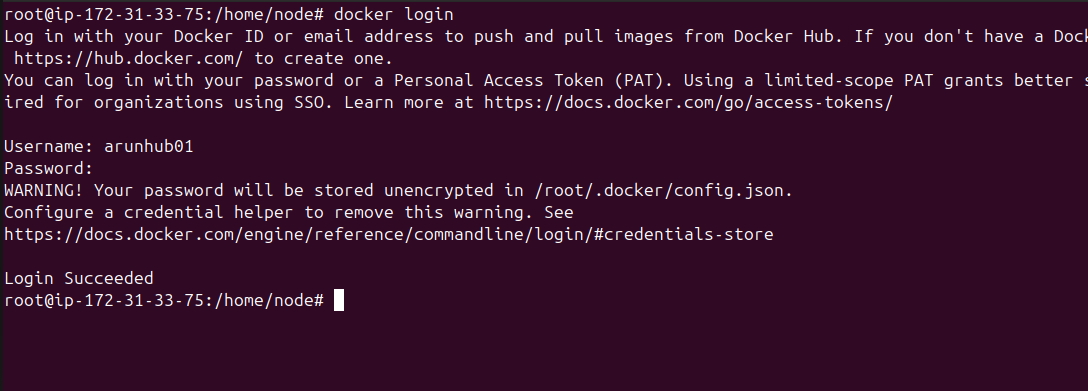


1. **Push the image to Docker Hub**

**—-------------------------------------------------**

→ Log in to Docker Hub

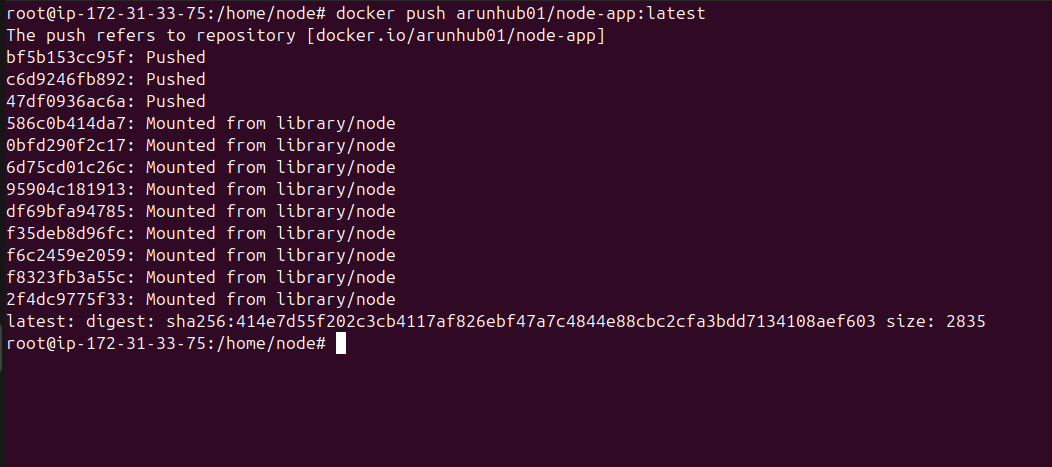
**docker login**

****

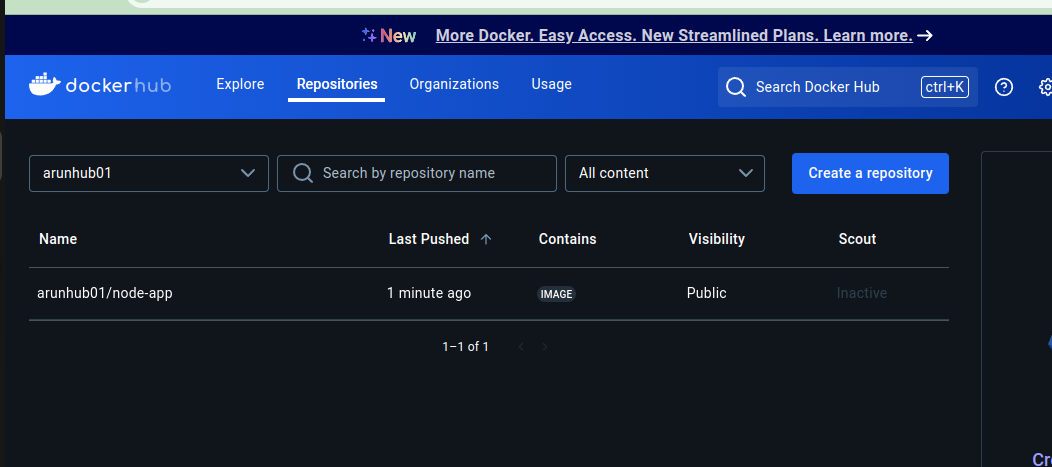
When you use docker login command, you were asked by username and password. Please enter correct username and password of your dockerhub.

→ Push the image to docker hub

**docker push arunhub01/node-app:latest**

****

→ Go to the Docker hub and check repo is there or not



Completed

—--------------------------------------------------