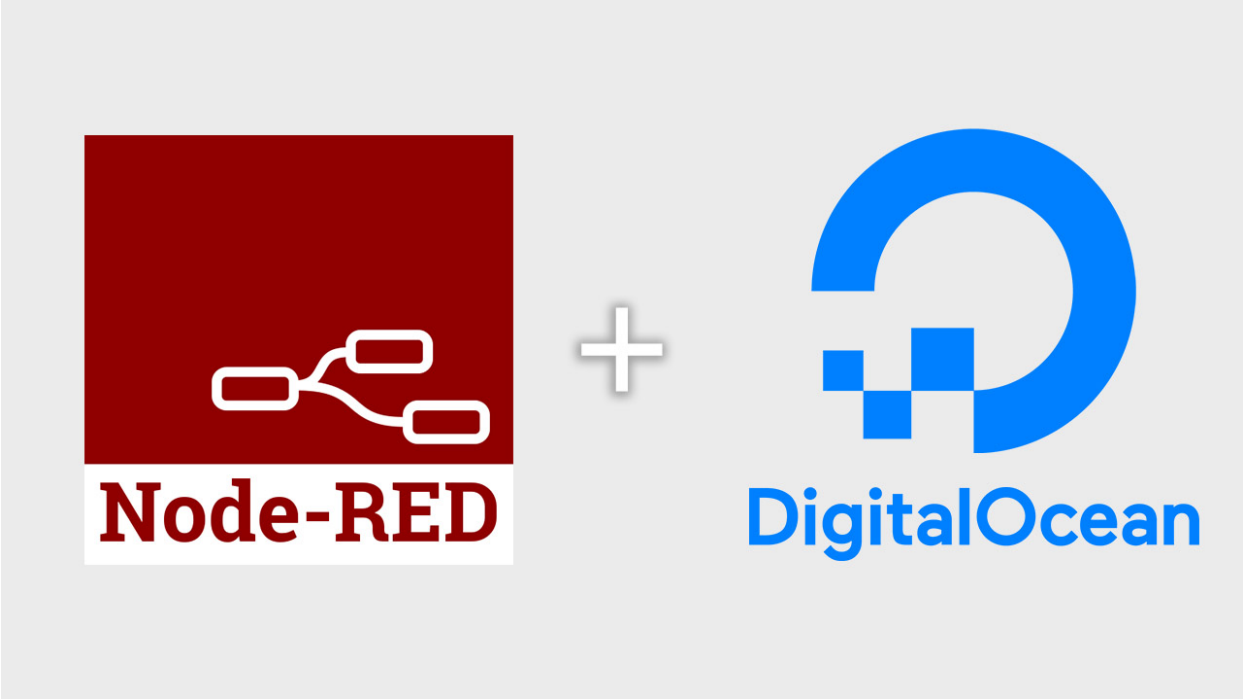
Access Node-RED Dashboard from Anywhere using Digital Ocean

This guide explains how to install Node-RED software on a Linux Ubuntu VM (Virtual Machine) using Digital Ocean. Running Node-RED in the cloud allows you to access your Node-RED Dashboard from anywhere.



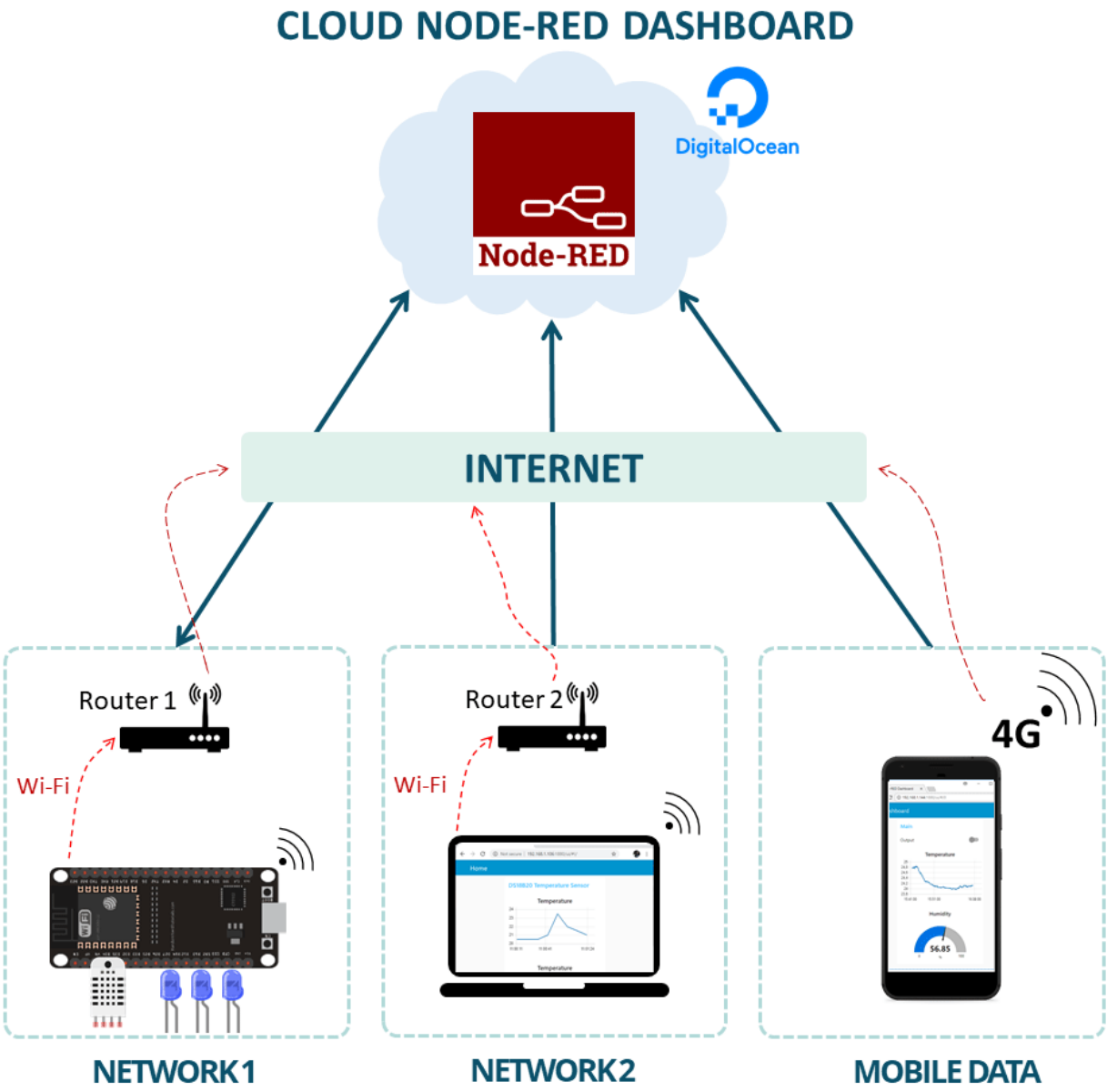
You can also install the MQTT Mosquitto Broker in your VM to connect several ESP32/ESP8266 boards and other IoT devices from anywhere using different networks with an Internet connection.

Introducing Node-RED

Node-RED is a powerful tool for building Internet of Things (IoT) applications using visual programming: connect code blocks (nodes) together to perform a task. Node-RED provides a dashboard (Node-RED dashboard) that can be used as an Home Automation platform to interact with IoT devices to control outputs and monitor sensors.

You can easily establish a communication with Node-RED and your ESP32/ESP8266 boards using HTTP requests or MQTT, for example.

What's the advantage of installing Node-RED on the cloud (Digital Ocean hosting service) and how it works? Here's an example.



- You have Node-RED software and Node-RED dashboard installed on Digital Ocean.
- The ESP32 can communicate with Node-RED as long as it is connected to a router with access to the internet.
- The ESP32 can send sensor readings to Node-RED and/or you can control its outputs by accessing the dashboard.
- You can access Node-RED dashboard using your computer or your smartphone from anywhere in the world.

- This allows you to control and monitor one or multiple ESP32/ESP8266 boards from anywhere.

Hosting Service – Digital Ocean

To run your Cloud MQTT Mosquitto Broker, you need to use a hosting service that allows you to have access to the command line and install any software that you need. I recommend using Digital Ocean that offers an Ubuntu server that you can manage through a command line.

I've been using it since 2015 and I personally recommend it, but you can use any other hosting service. Any hosting service that offers a Linux Ubuntu VM with full console access should work.

If you don't have a hosting account, I recommend signing up for Digital Ocean. When you sign up for Digital Ocean, you can try it for 60 days (they give you free credits to test the platform).

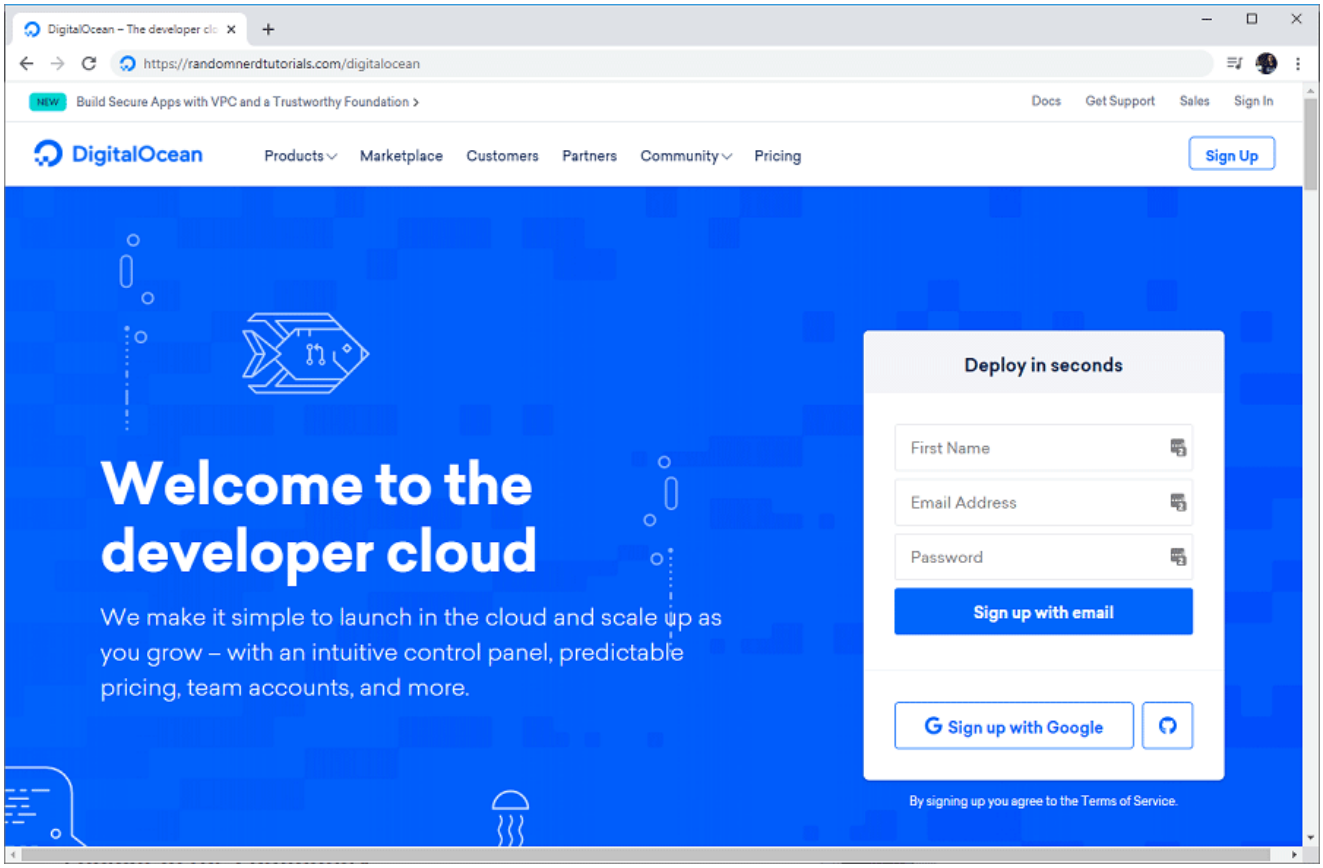
Grab Linux Ubuntu VM on Digital Ocean »

If you like our projects, you might consider signing up to the recommended hosting service, because you'll be supporting our work.

> **Note:** you can also run Node-RED in your local network using a Raspberry Pi board. However, the purpose of this tutorial is to run Node-RED in the cloud to communicate with boards (or other IoT devices) across different networks.

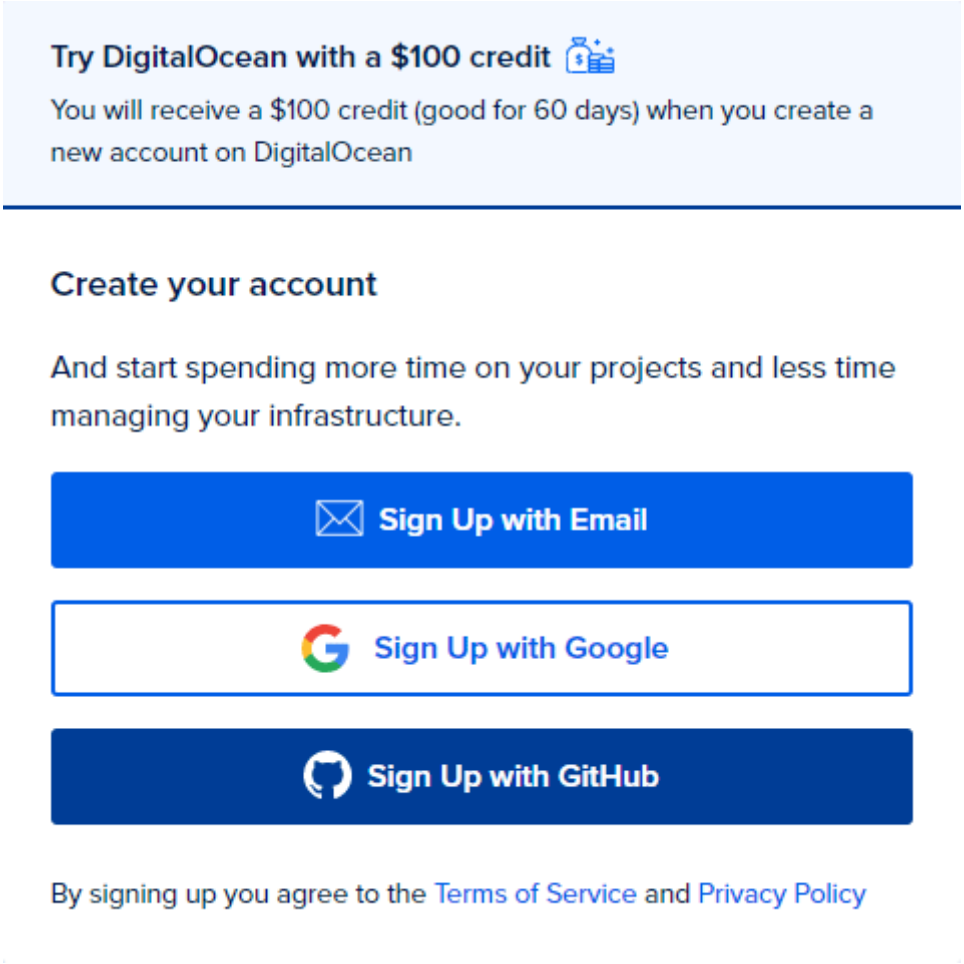
Creating Digital Ocean Account

To create a Digital Ocean Account, go to Digital Ocean and press the "**Sign Up**" button.

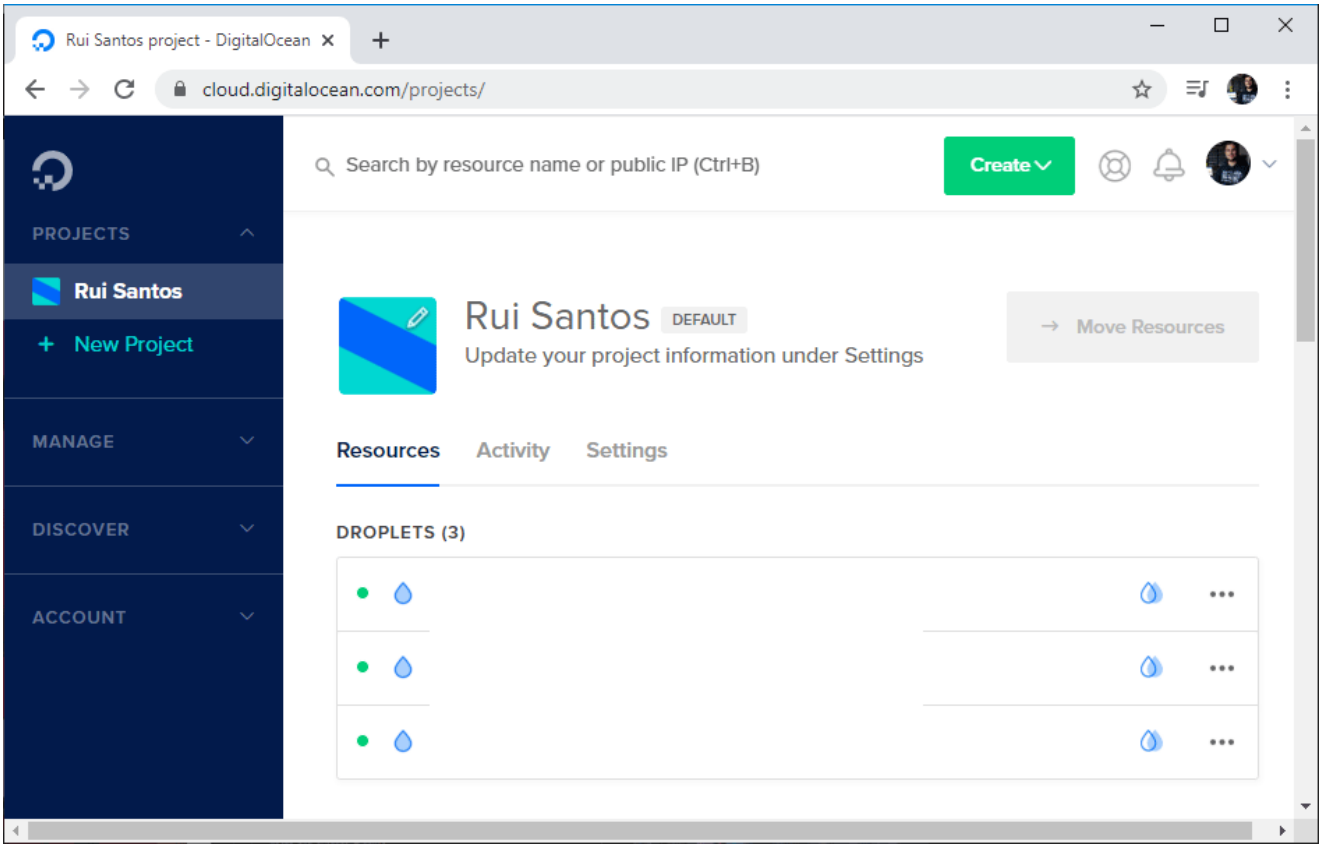


Create your account, and you'll receive a $100 credit that you can use for 60 days to test the platform. You might need to enter a valid credit card, but you can cancel your account anytime if you're no longer interested in using the service after the free 60 days trial.

Complete the account creation using your preferred method (I always use the Email option).



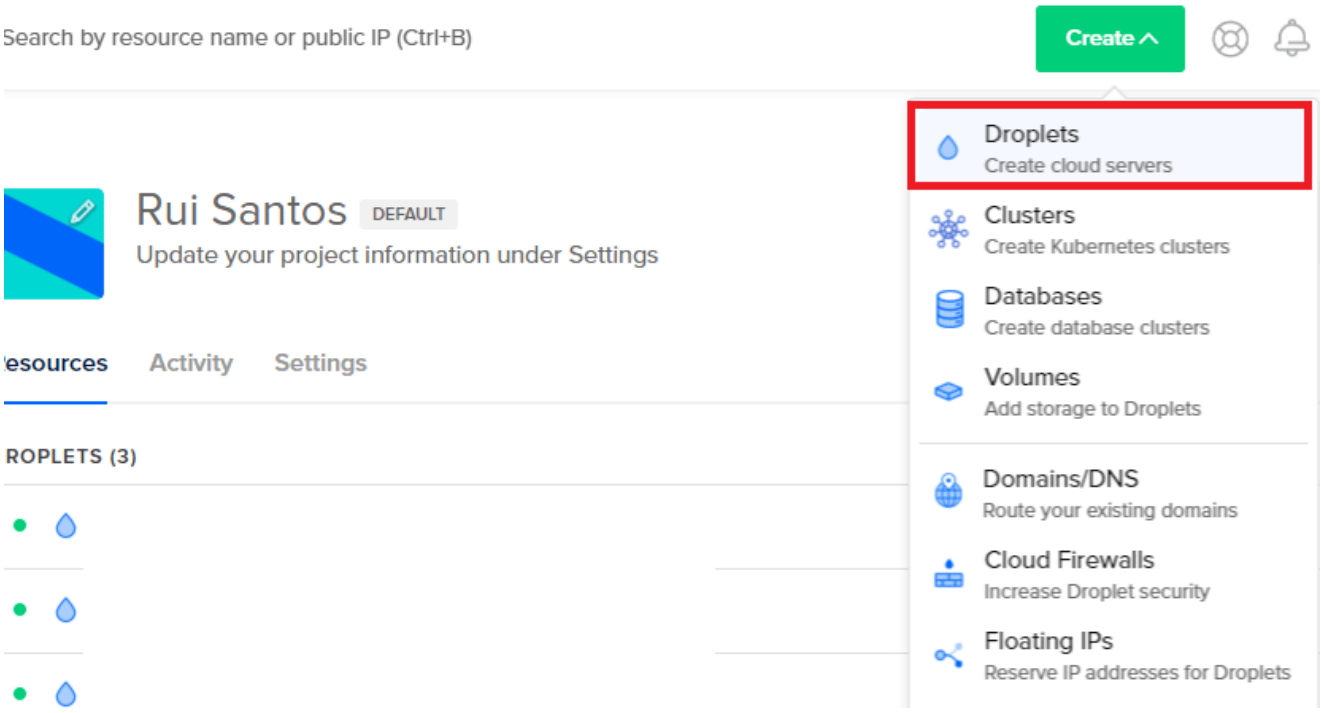
Confirm your account and login, you should see a similar Dashboard.



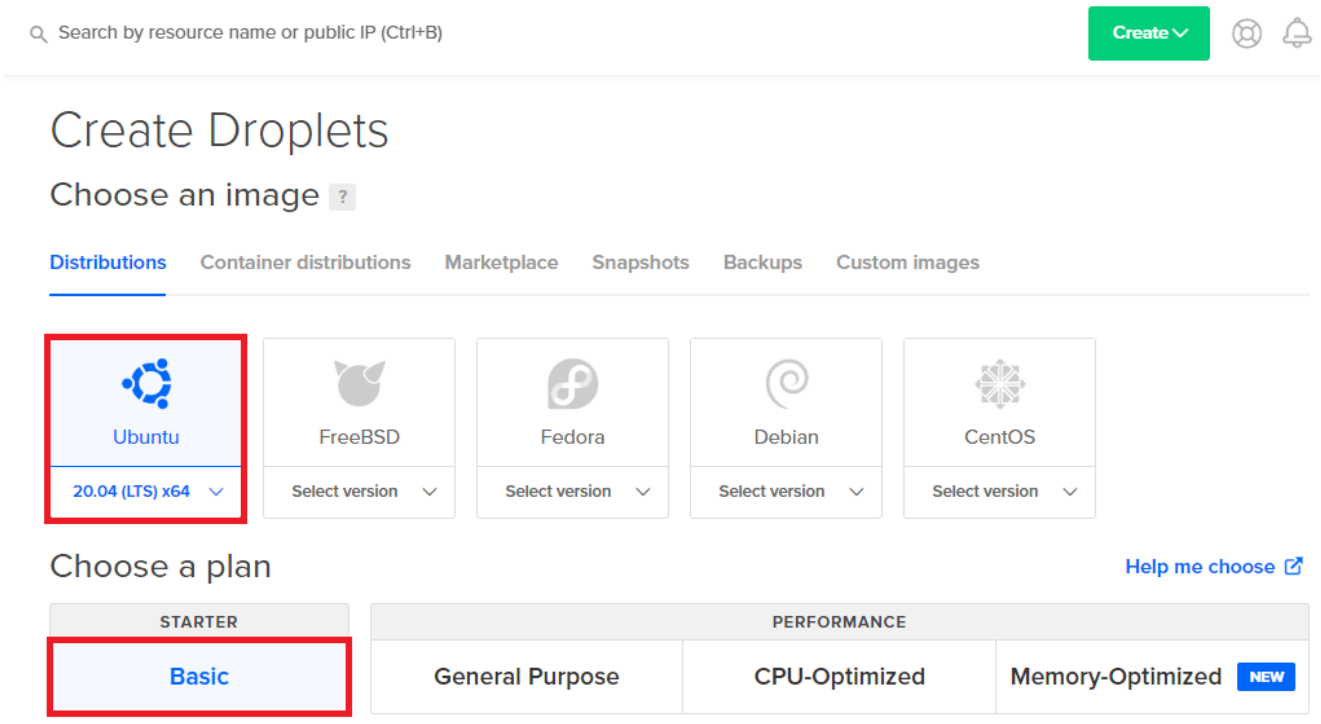
Create a Droplet (Linux Ubuntu VM)

To create a new VM, press the "**Create**" button on the top right corner and select the "**Droplets**" option. Digital Ocean calls **Droplets** to its VMs.

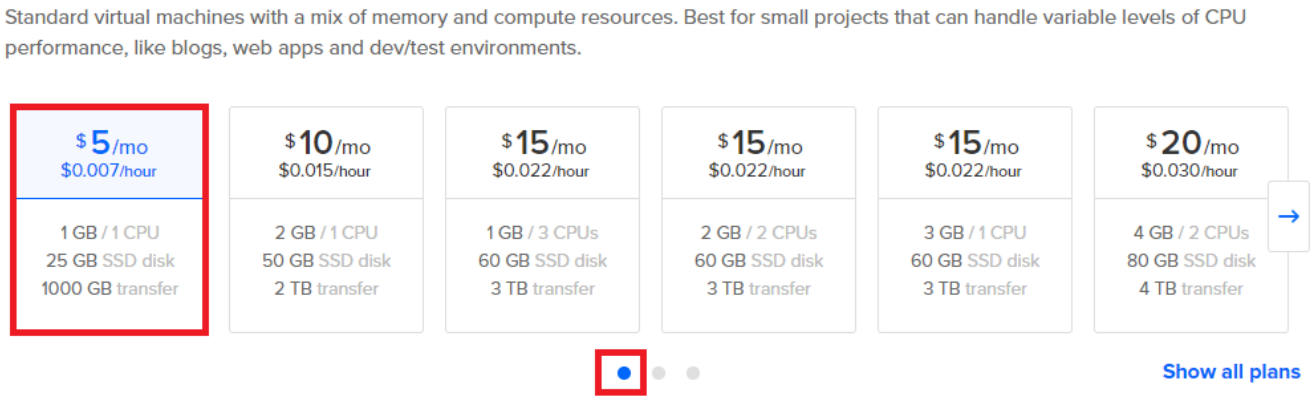
> **Important:** if you're already running a Droplet with MQTT Mosquitto Broker, you can skip these next steps (creating a Droplet). You can run both Node-RED and Mosquitto MQTT broker in the same server.



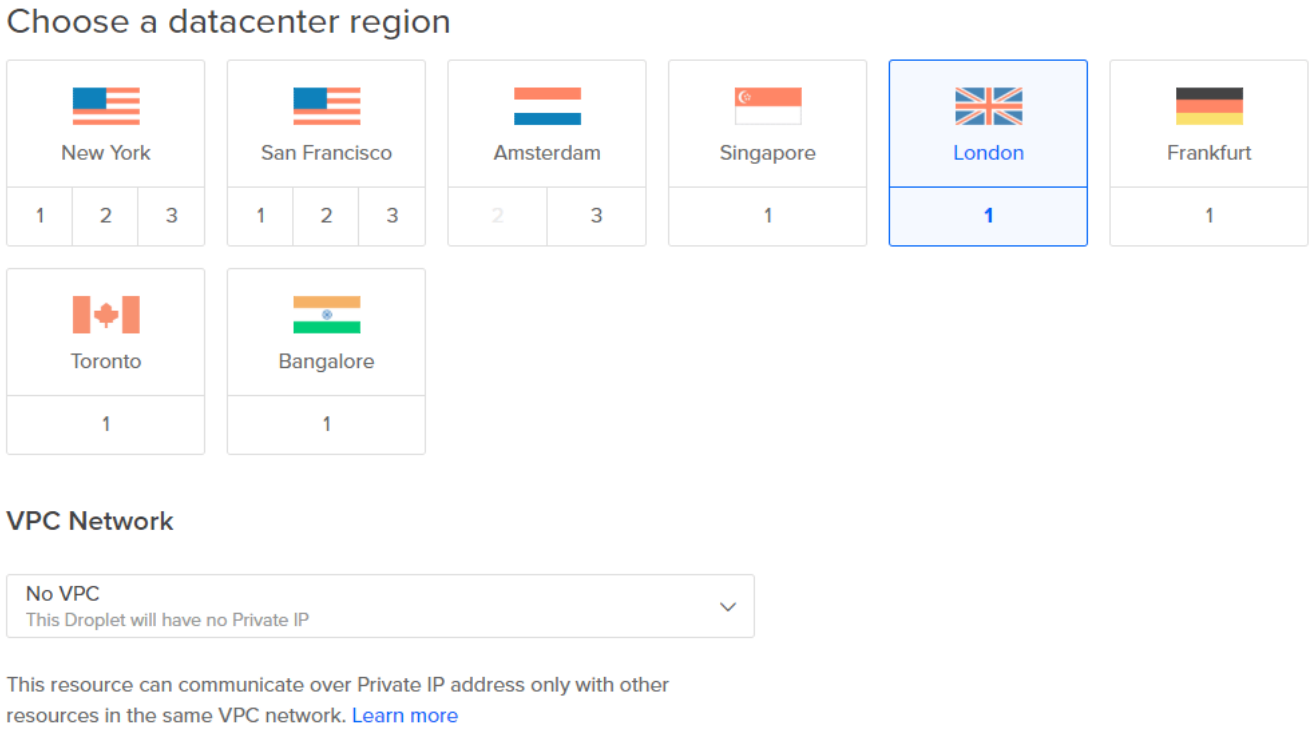
For this guide I'll be using Ubuntu 20.04 (LTS) x64, and I recommend choosing the same option. You can also use the "**Basic**" starter plan.



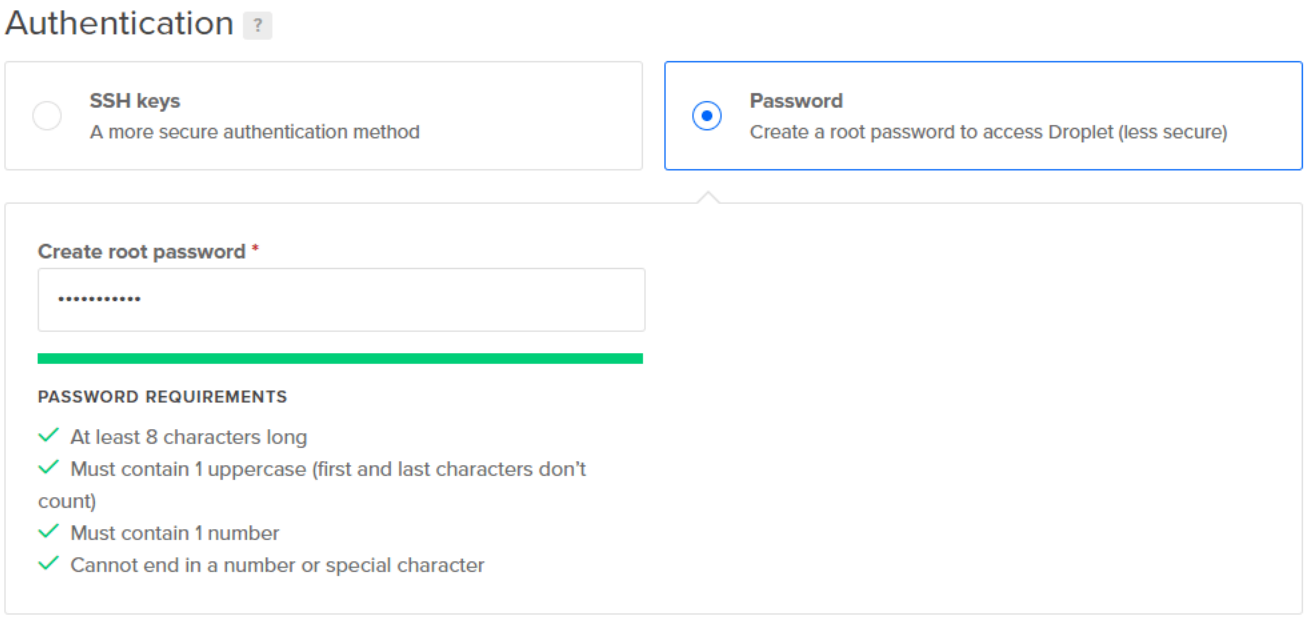
In the VM resources menu, you can choose the cheapest plan for $5/month. Even with the lowest plan, it will run the MQTT Broker and Node-RED smoothly .



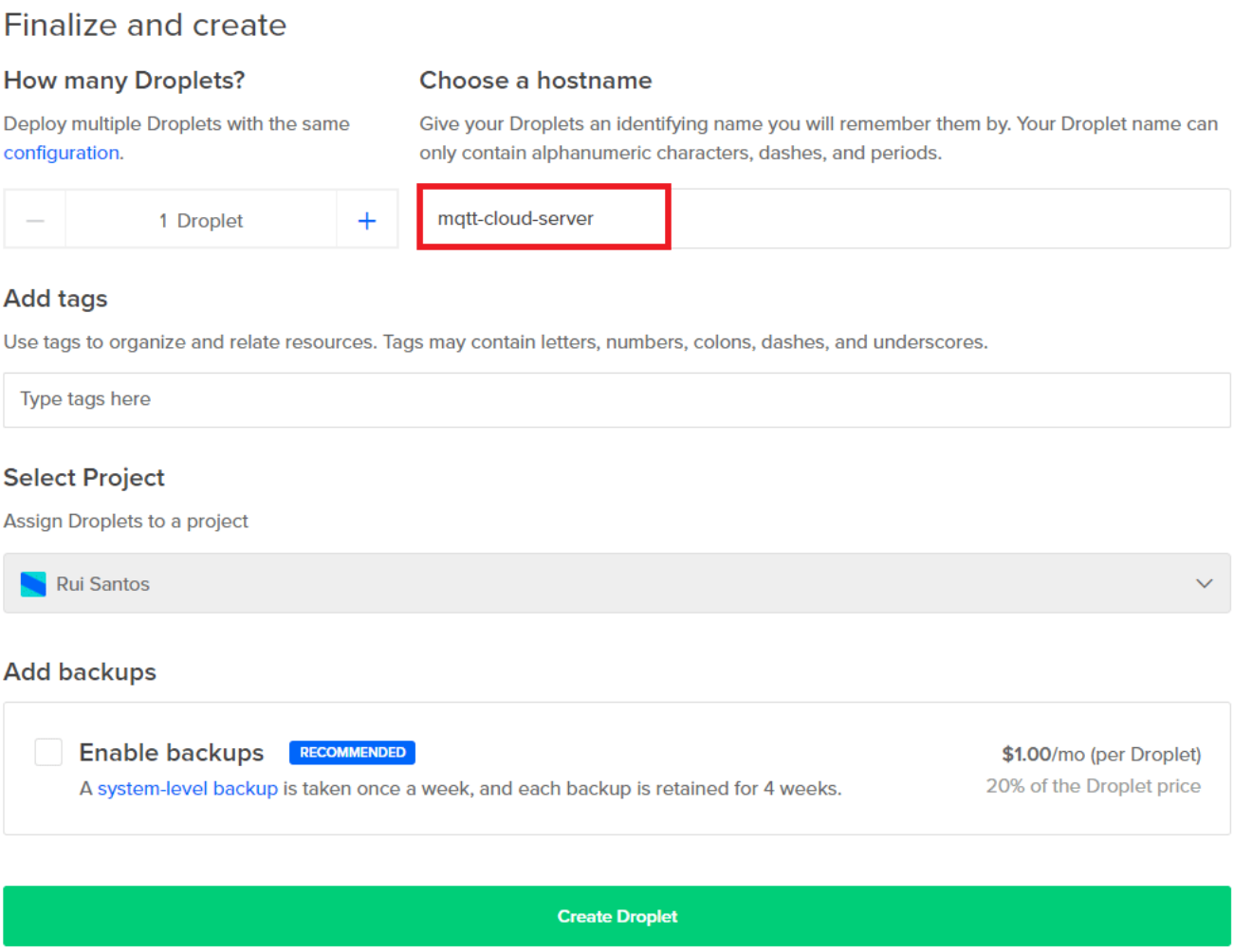
Choose the Virtual Machine datacenter location closer to you, in my case I've used "**London**" region.



Create the root password that allows you to access your Droplet (save this password, because you'll need it to access your server).



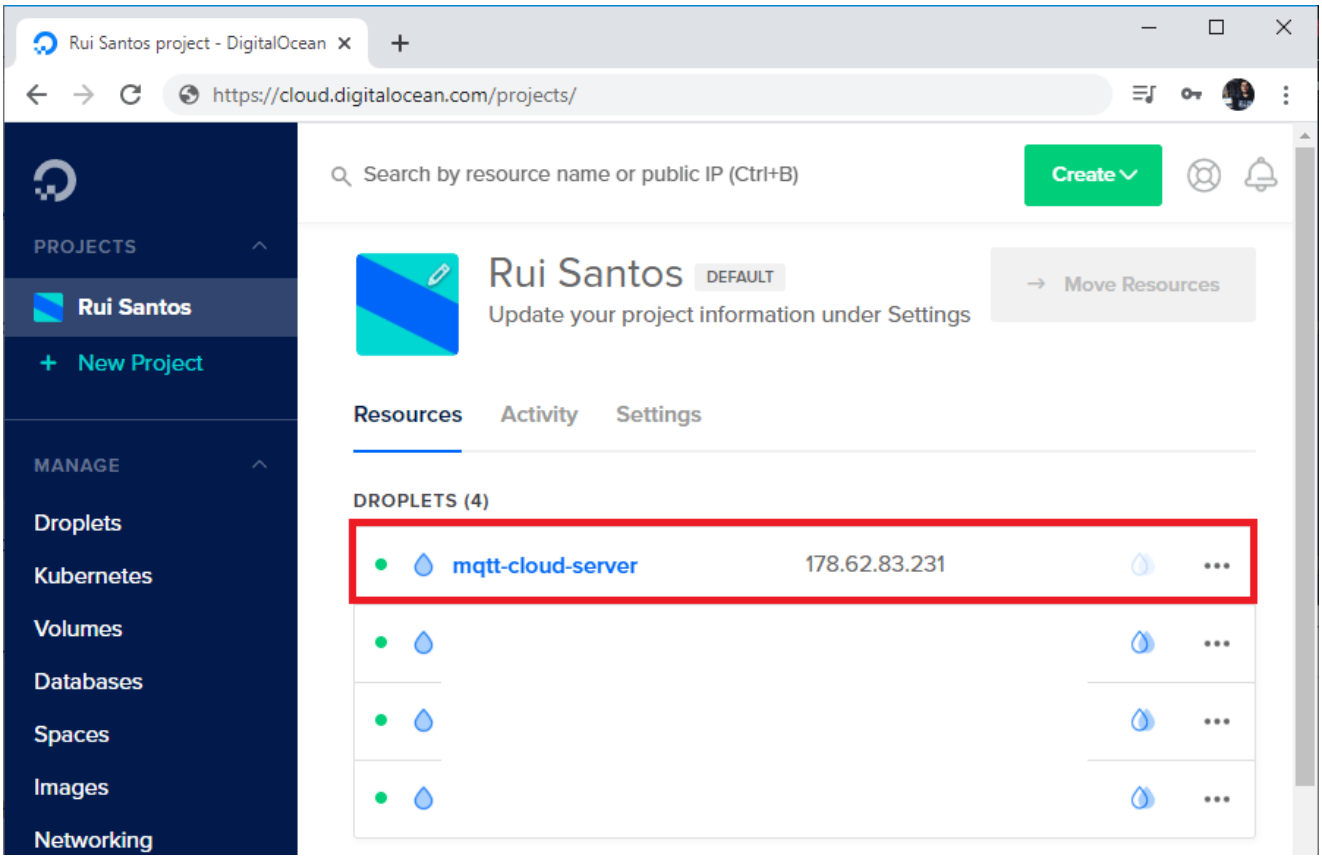
Finally, choose a **hostname** to easily identify which Virtual Machine you are working with. I've named my Droplet as "mqtt-cloud-server". That's it, you just need to press the big green button "Create Droplet" to finish the process.



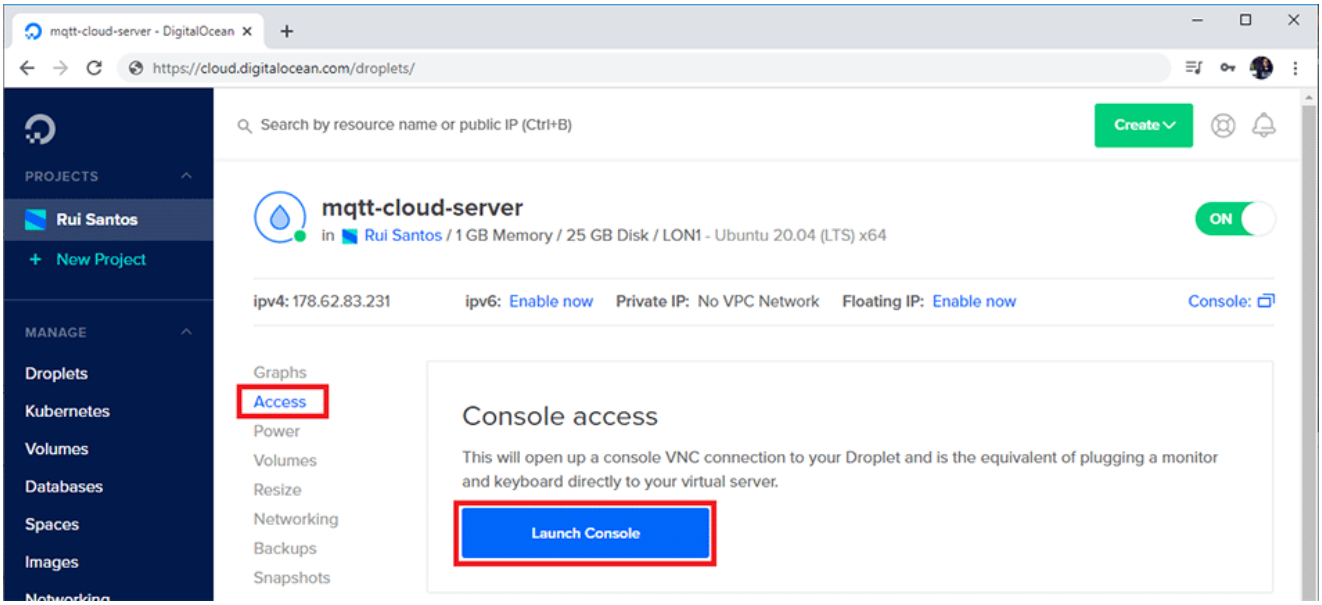
Wait a few minutes and when the progress bar ends, your Droplet is ready.

Accessing Your Linux Ubuntu VM Console

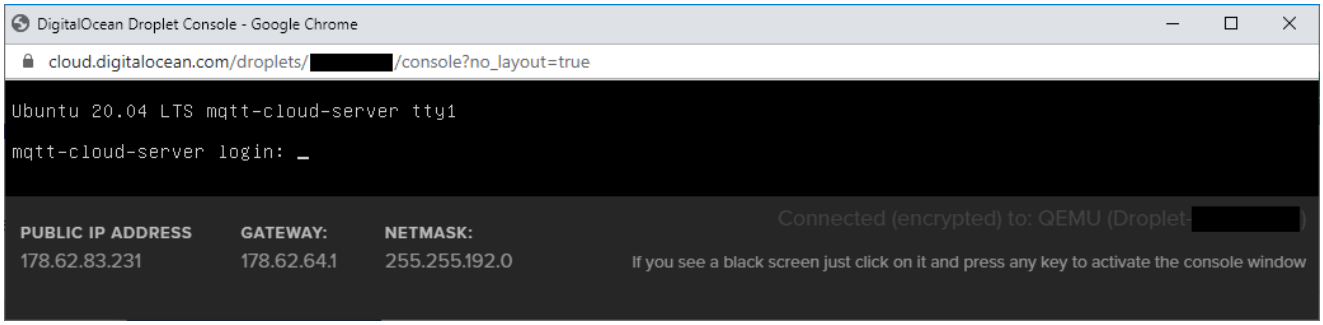
When your Droplet is prepared, open your newly created server (in my case, it's called "**mqtt-cloud-server**").



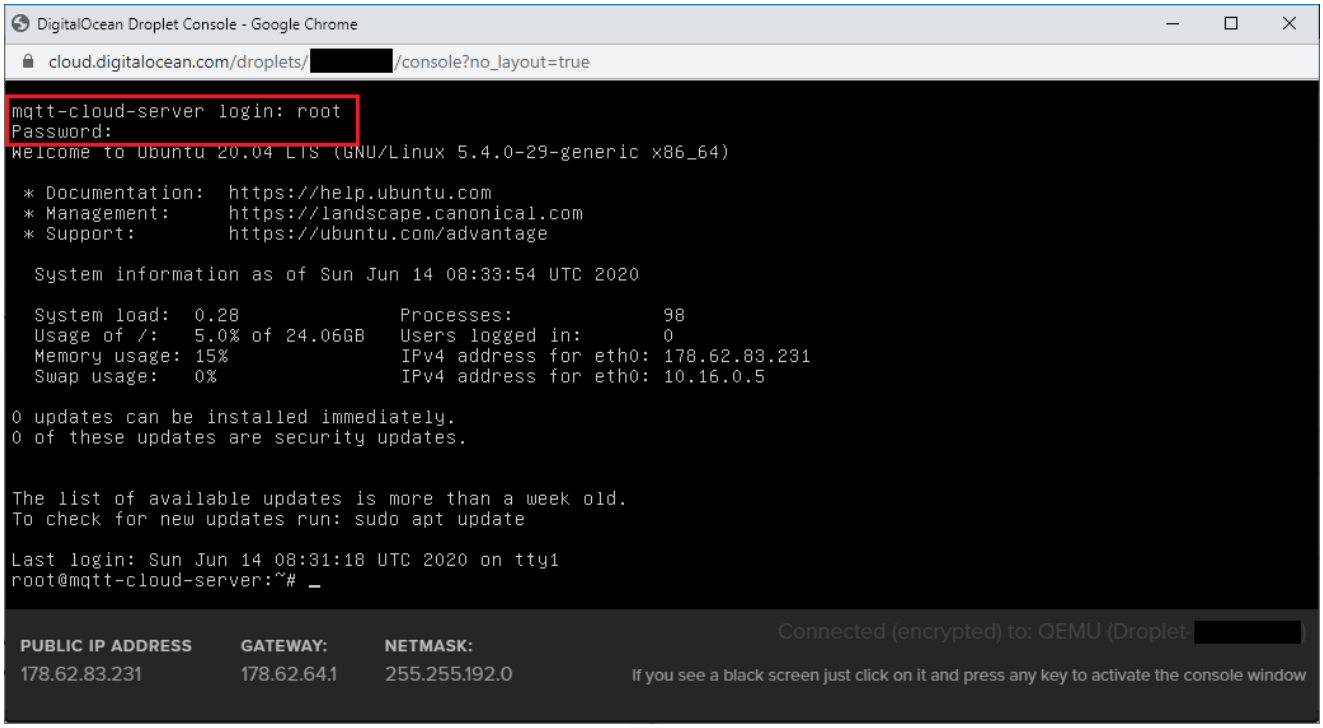
Select the "**Access**" menu and press the "**Launch Console**" button.



A new browser window opens up in your computer.



Type your login (root) and the password defined earlier, press Enter key to access your server.



There's an ***optional*** step, but it goes beyond the scope of this tutorial. It's ***not*** required to make this project work: prepare your server with non-root, sudo-enabled user and basic firewall with this Initial Server Setup with Ubuntu 20.04.

Installing Node-RED

Let's install Node-RED software.

Before installing the software, update and upgrade your server (this will take a few minutes to complete).

```
sudo apt update && sudo apt upgrade -y
```

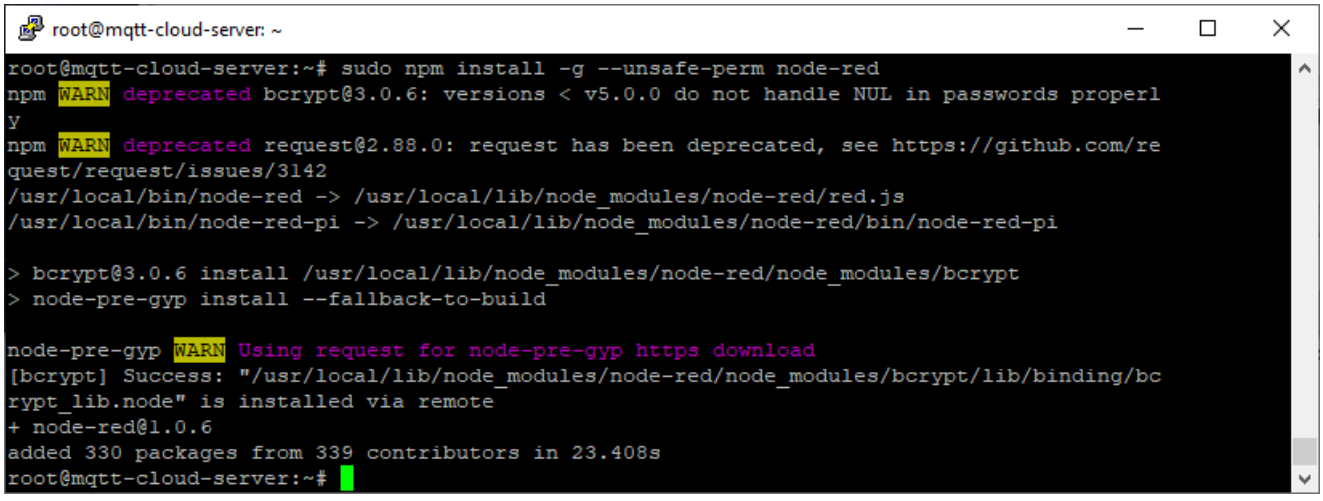
To install Node-RED, you'll need npm:

```
sudo apt install npm -y
```

This next command installs Node-RED as a global module along with all its dependencies.

```
sudo npm install -g --unsafe-perm node-red
```

When the installation process is completed, you should see a similar message:

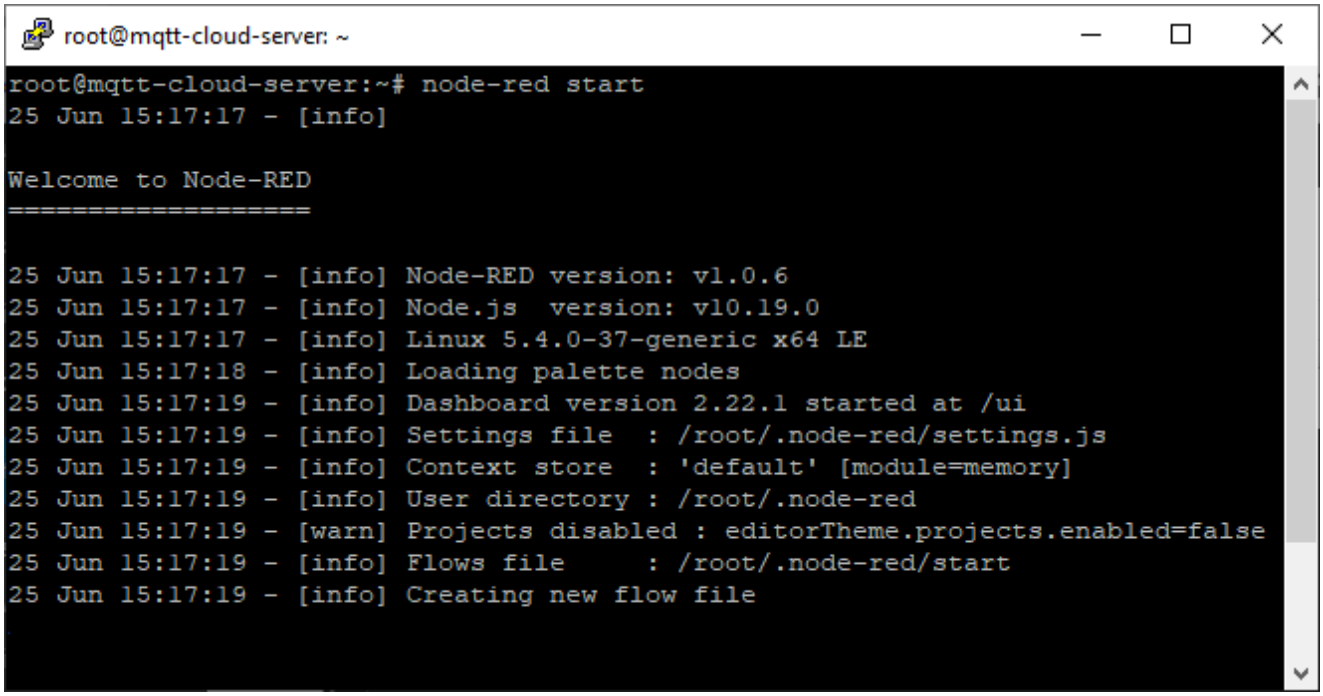


Open up a port on your server firewall. Node-RED defaults to using port 1880, so run this next command:

```
sudo ufw allow 1880
```

You can confirm it has succeeded if the end of the command output looks similar to:

```
node-red start
```



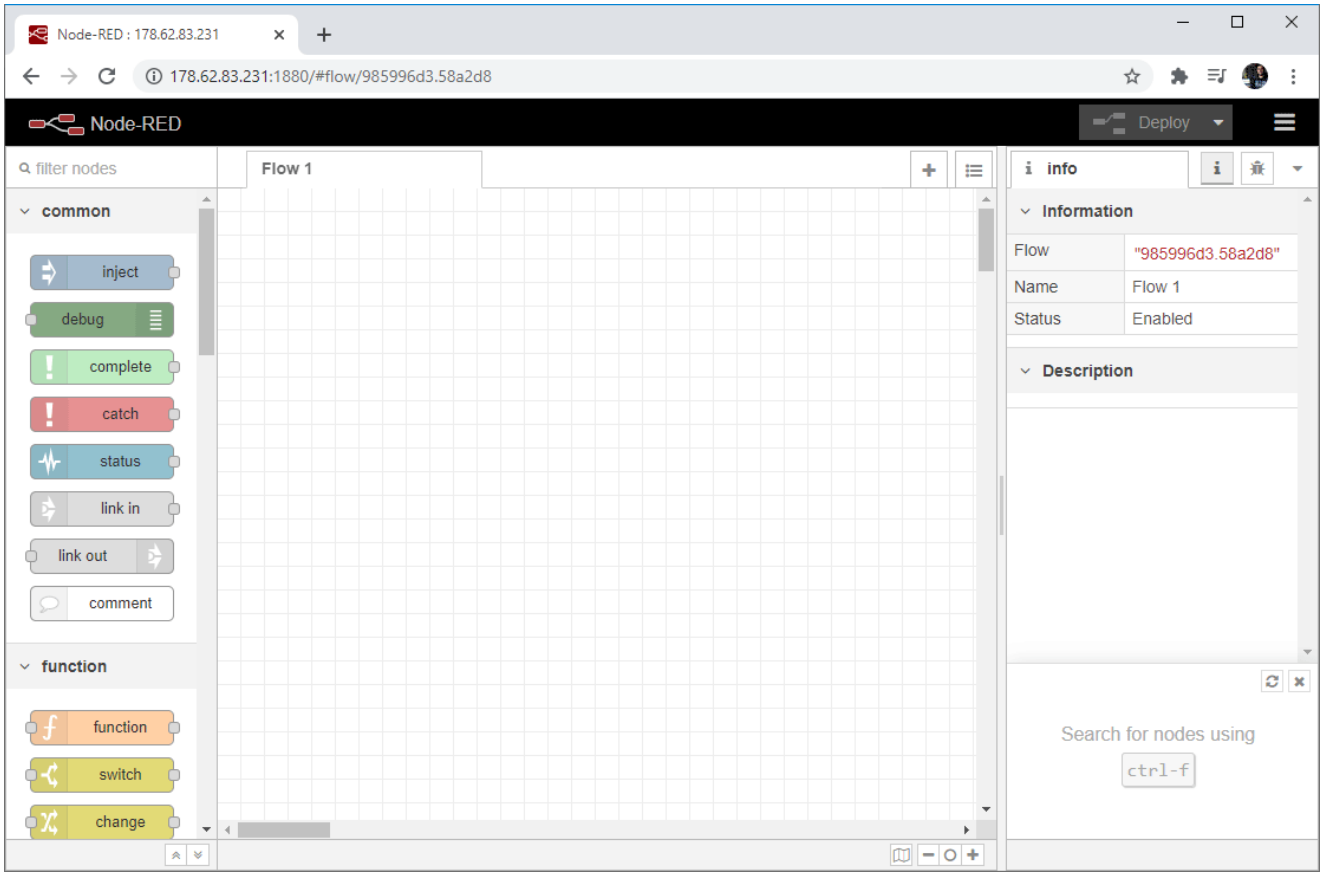
Wait a few seconds while everything loads, if you go to your Digital Ocean Droplet IP address followed by the port number 1880:

```
http://YOUR-Digital-Ocean-IP-Address:1880
```

For example:

```
http://178.62.83.231:1880
```

Node-RED software should load, here's how it should look like:



In your console window, press the Ctrl+C key to stop Node-RED.

Autostart Node-RED on Boot

To make Node-RED autostart when you boot or restart your Virtual Machine, you need to create a new *systemd* file for the `nodered.service`:

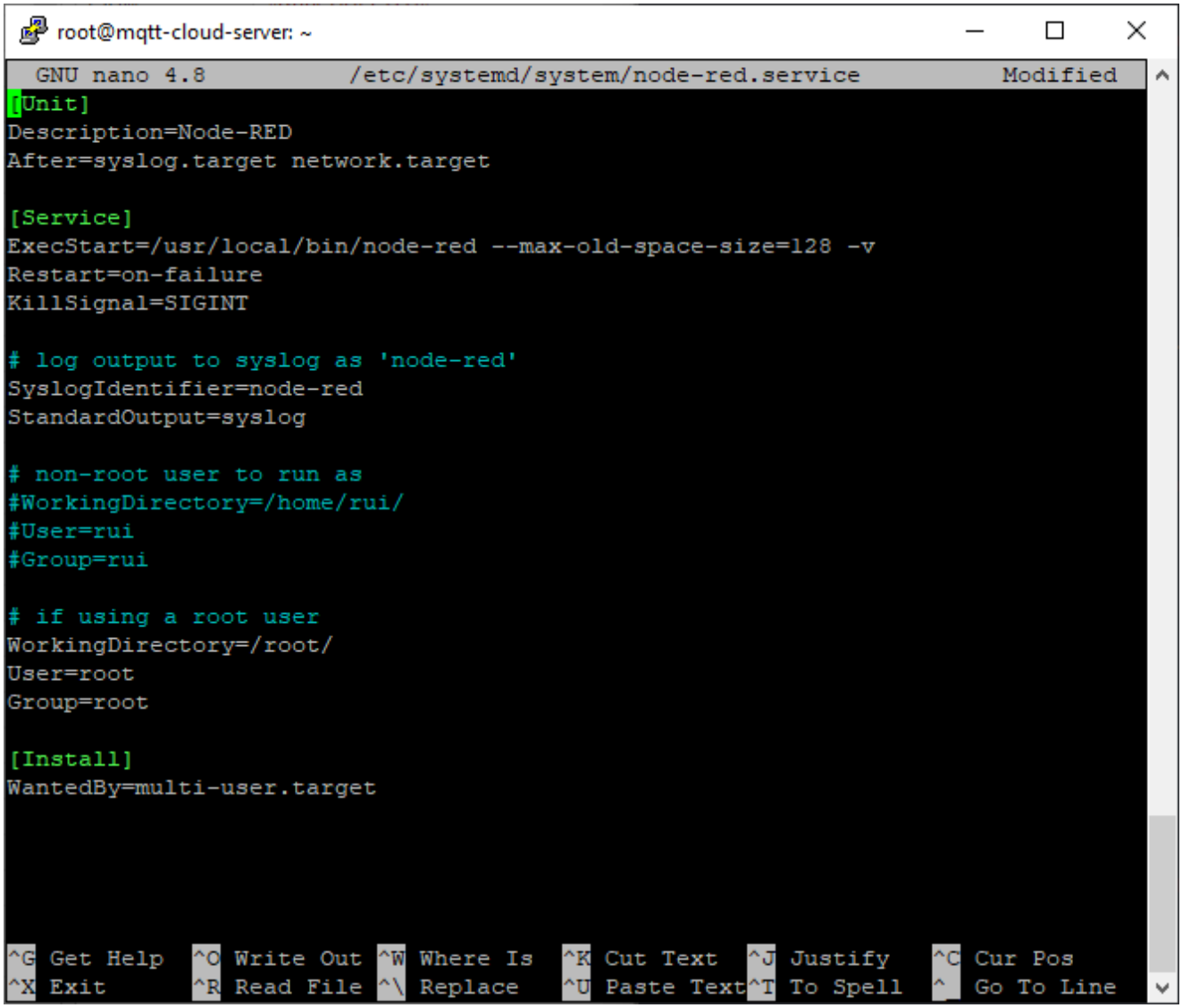
```
sudo nano /etc/systemd/system/nodered.service
```

Then, add the following (if you're using root, you can leave the file as it is – otherwise comment the root and configure the non-root user option):

```
[Unit]
Description=Node-RED
After=syslog.target network.target

[Service]
ExecStart=/usr/local/bin/node-red --max-old-space-size=128 -v
Restart=on-failure
KillSignal=SIGINT

# log output to syslog as 'node-red'
SyslogIdentifier=node-red
StandardOutput=syslog
```

```
# non-root user to run as
#WorkingDirectory=/home/rui/
#User=rui
#Group=rui

# if using a root user
WorkingDirectory=/root/
User=root
Group=root

[Install]
WantedBy=multi-user.target
```

Your file should look like this, save it and exit (Ctrl+X, Y, Enter key).



To enable the Node-RED service and run this file on boot, enter the command:

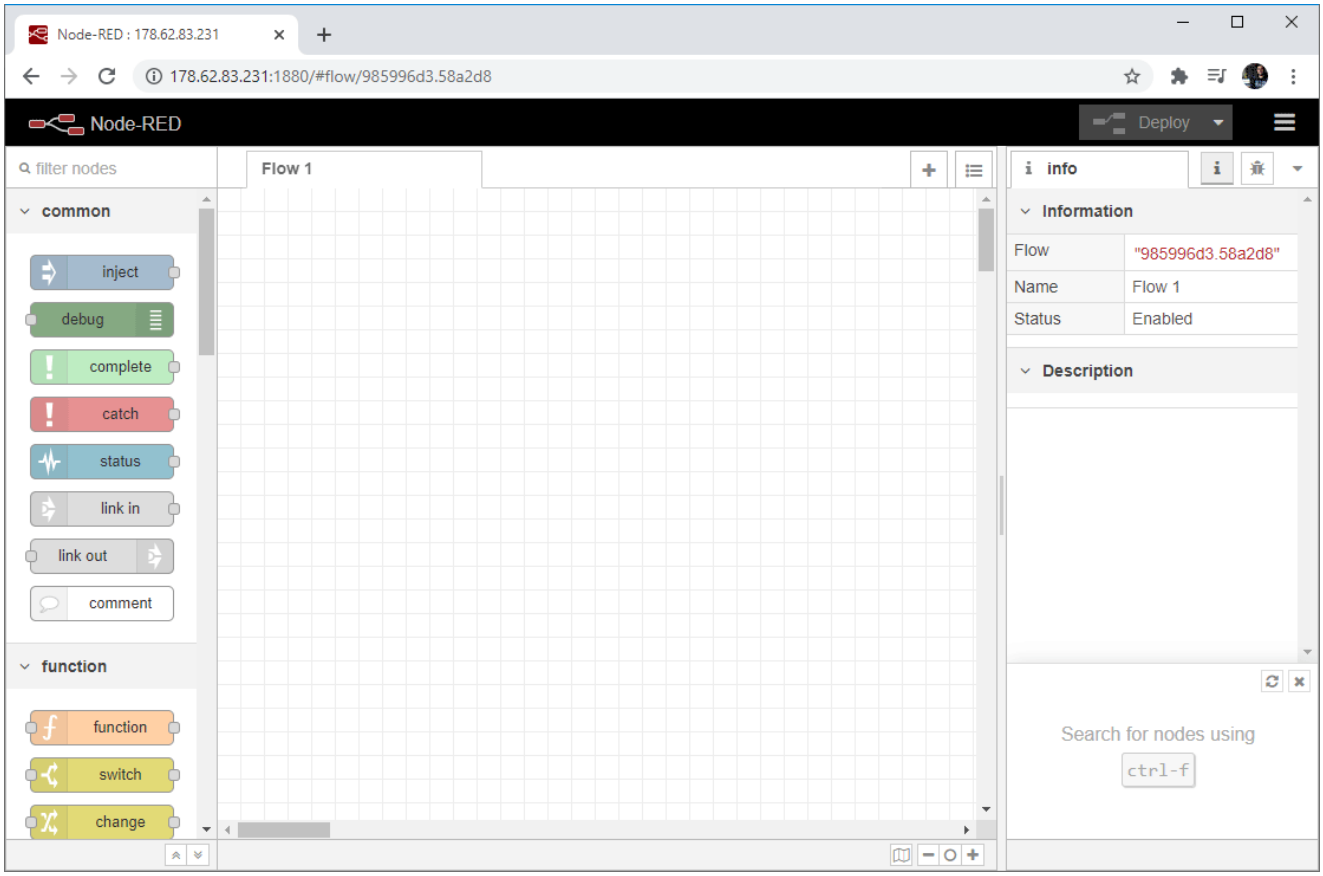
```
sudo systemctl enable nodered.service
```

Restart your Virtual Machine to test if Node-RED is automatically starting on boot:

```
sudo reboot
```

Wait approximately 2 to 3 minutes for your server to fully restart, when you open your server IP address Node-RED software should load automatically:

```
http://YOUR-Digital-Ocean-IP-Address:1880
```



That's it! Your Node-RED software is prepared.

Securing Node-RED Software with Username and Password

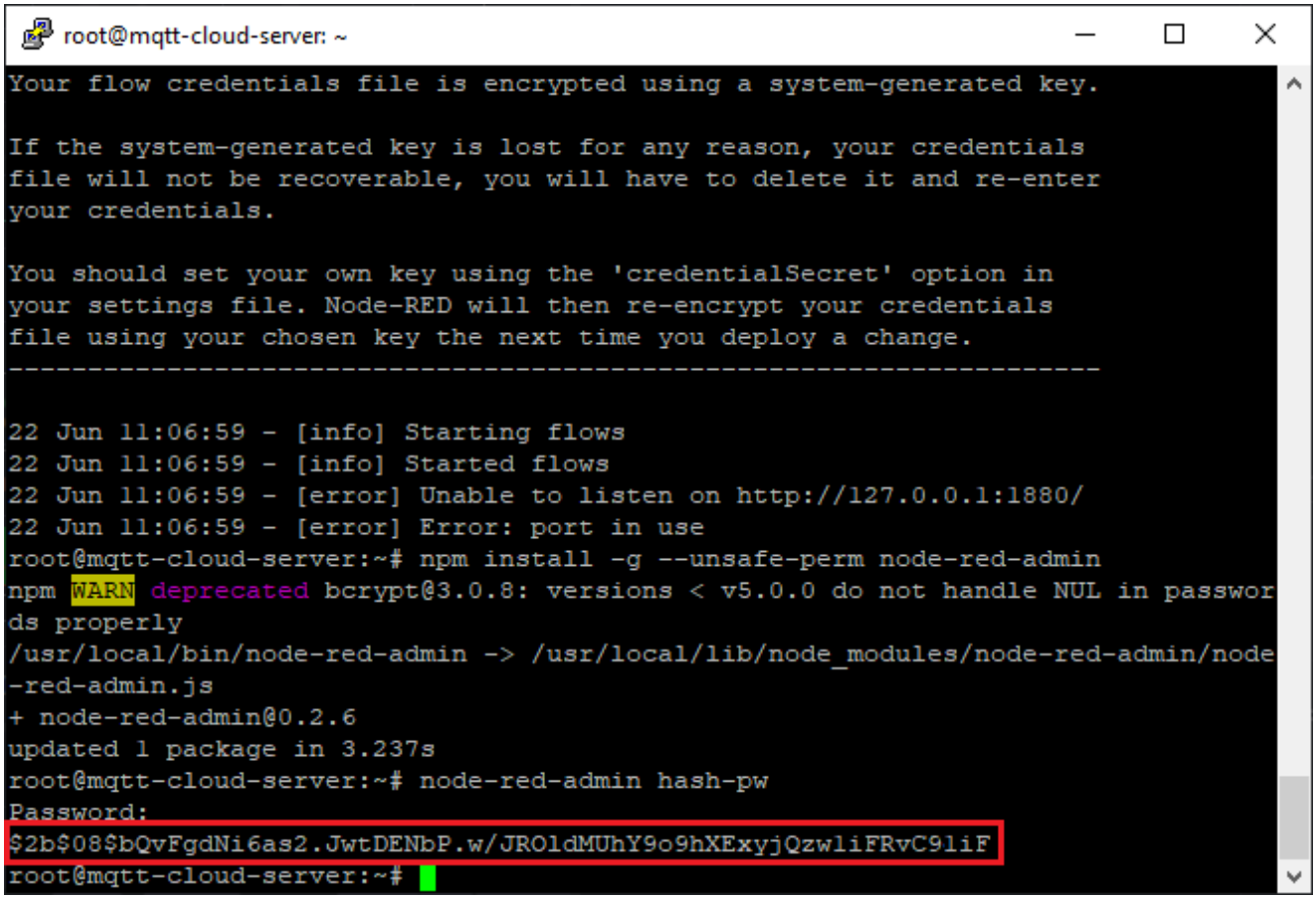
Install node-red-admin package to make it available globally:

```
npm install -g --unsafe-perm node-red-admin
```

Run the next command to create a password hash:

```
node-red-admin hash-pw
```

You will be prompted for a password. Type your desired password, press Enter key, and a hash will be printed on screen.



Copy the hash string to your clipboard and open the Node-RED settings file. In my case the hash is:

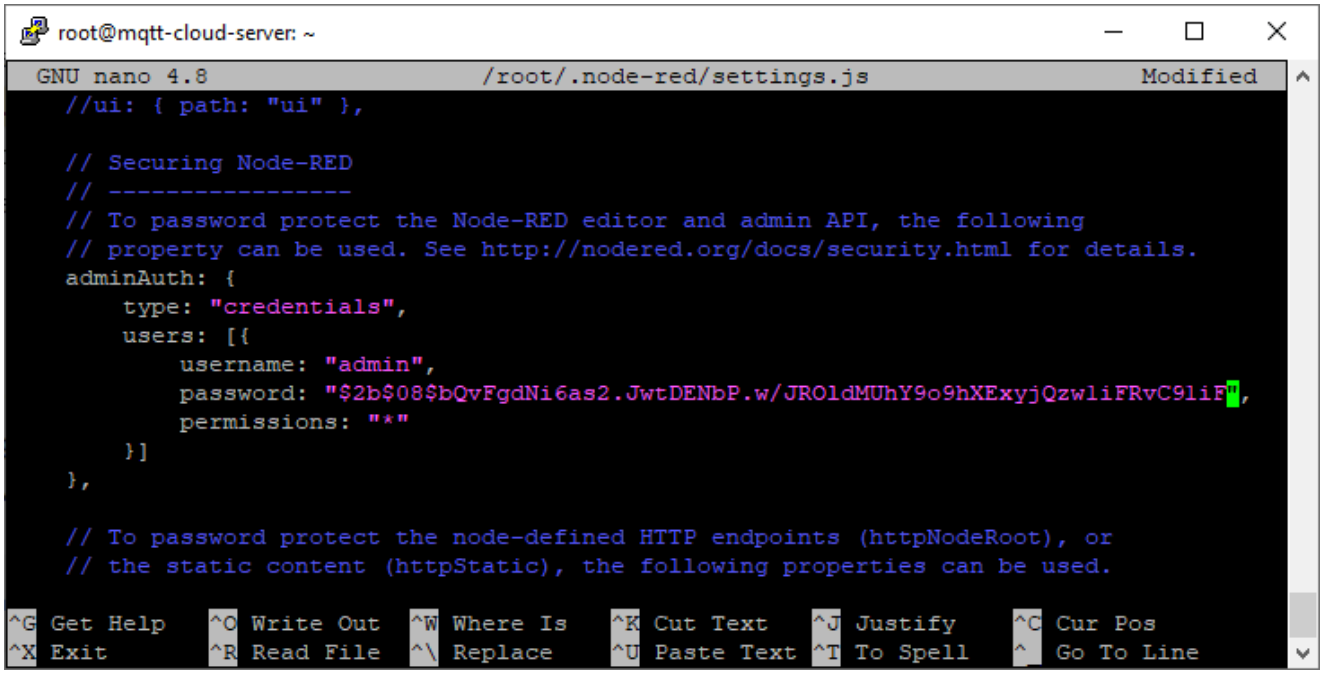
```
$2b$08$bQvFgdNi6as2.JwtDENbP.w/JROldMUhY9o9hXExyjQzw1iFRvC9liF
```

Open the *settings.js* file:

```
sudo nano ~/.node-red/settings.js
```

Scroll down and uncomment the adminAuth block (by removing the “// ” in front of each line). Change username to whatever you like, and paste the previously

generated hash into the password field.

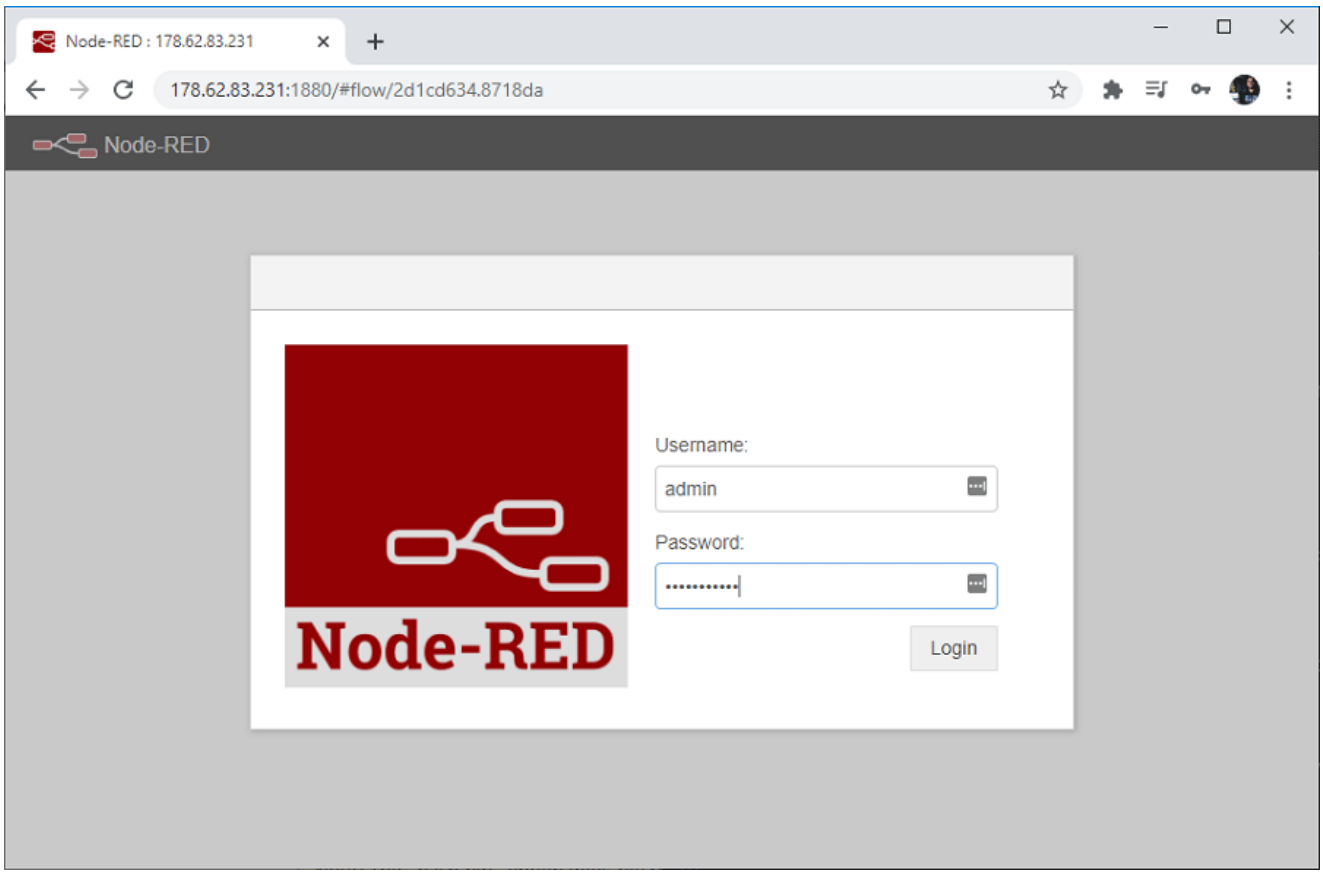


Make sure you uncomment and replace the password field with the hash exactly as illustrated above, failing to do it will make it impossible to login and you'll have to repeat this process.

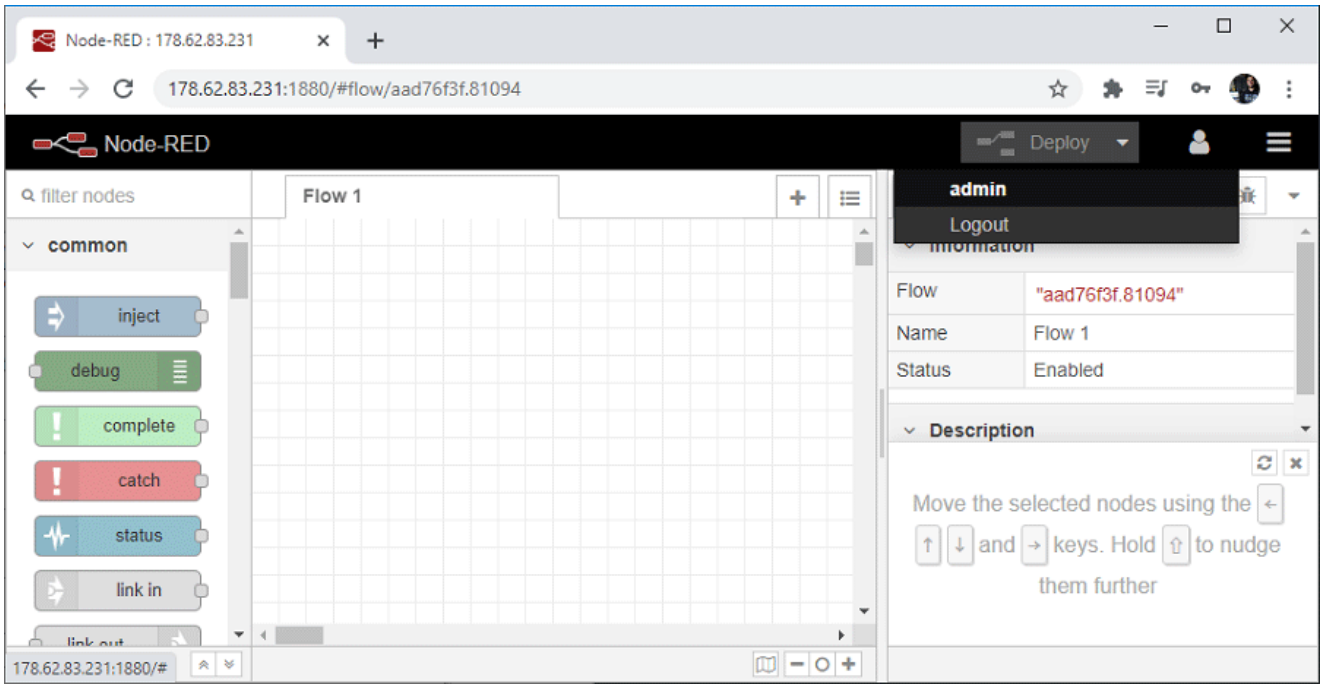
Exit the file and save it (Ctr+X, Y, Enter key). Restart your server to ensure that all changes take effect:

```
sudo reboot
```

Now, when you access your Node-RED software you'll be prompted to enter the username (default is admin) and the password defined earlier.



After logging in, you'll be redirected to Node-RED software where you can create and edit your flows. You also have a menu at the top right corner to logout.



That's it! Your server is ready and you can install Node-RED Dashboard or connect Node-RED to your cloud MQTT broker.

(Optional) Taking It Further – Node-RED SSL Certificate

The best method to add an SSL certificate to your server is by having a domain name pointed at your server and use Let's Encrypt SSL certificates.

- You'll have to buy a domain name and point it to Digital Ocean Name Servers (like example.com).
- The web server Nginx installed, with the firewall updated to allow traffic on ports 80 and 443, as describe in How To Install Nginx on Ubuntu (that guides also works for version 20.04)
- Let's Encrypt installed, and a certificate generated for the domain name (like example.com). How To Secure Nginx with Let's Encrypt on Ubuntu 20.04. You can ignore steps 3-5 regarding Nginx configuration, as we'll cover that next.

Having those prerequisites completed, open a new Nginx configuration for the site (replace the path with your domain name).

```
sudo nano /etc/nginx/sites-enabled/example.com
```

Copy and paste the following to the new file. You need to change the domain name and certificate paths:

```
server {
  listen 80;
  listen 443 ssl http2;
  server_name example.com;
  ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/example.com/privkey.pem;
  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
  ssl_ciphers
EECDH+AES128:RSA+AES128:EECDH+AES256:RSA+AES256:EECDH+3DES:RSA+3DES:!MD5;
  ssl_prefer_server_ciphers On;
  ssl_session_cache shared:SSL:128m;
  ssl_stapling on;
  ssl_stapling_verify on;
  resolver 8.8.8.8;

  location / {
    if ($scheme = http) {
      return 301 https://$server_name$request_uri;
    }
```

```
      proxy_pass http://localhost:1880;
      proxy_set_header Host $host;
      proxy_set_header X-Real-IP $remote_addr;
      proxy_http_version 1.1;
      proxy_set_header Upgrade $http_upgrade;
      proxy_set_header Connection "upgrade";
    }

    location '/.well-known/acme-challenge' {
      root /var/www/html;
    }
  }
```

Change the example.com text to your own domain name:

```
server_name example.com;
ssl_certificate /etc/letsencrypt/live/example.com/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/example.com/privkey.pem;
```

When you're done editing the file, save it and exit the editor (Ctrl+X, Y, Enter key). Reload Nginx to pick up the new configuration.

```
sudo systemctl reload nginx
```

Finally, restart your server:

```
sudo reboot
```

Wait a couple of minutes to restart. Finally, navigate to your server: `http://example.com`. You should be redirected to `https://example.com` (note the https) and see the Node-RED software login page. This means we're now proxying Node-RED through Nginx.

Wrapping Up

This complete guide was tested and it should work, however there are many steps and they must be followed exactly as we describe, otherwise something might not work properly.

In all our guides and projects we always try to help if anyone gets stuck. However, in this particular case, there are so many steps that it can be extremely hard to help you without having access to the server and test it (of course, we don't have the resources to personally help everyone).

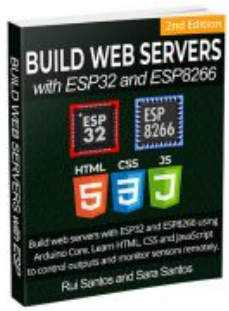
If you have any problem installing Mosquitto MQTT broker, preparing your Linux Ubuntu server, running Node-RED, installing an SSL certificate, contact Digital Ocean support and describe exactly what's happening. I've been using their service since 2015 and they always have an extremely helpful support team (or just use their Forum).

Now, if you want to install Mosquitto MQTT Broker on Digital Ocean, follow the next tutorial: Run Your Cloud MQTT Mosquitto Broker (access from anywhere using Digital Ocean).

Read the next guides to learn more about Node-RED:

- Node-RED Dashboard – Getting Started
- ESP32 MQTT – Publish and Subscribe with Arduino IDE
- ESP32 MQTT – Publish BME280 Sensor Readings (Arduino IDE)

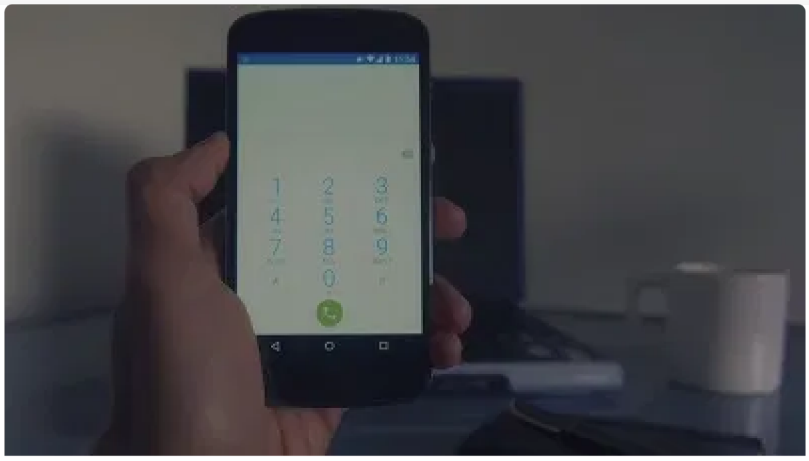
Thanks for reading.



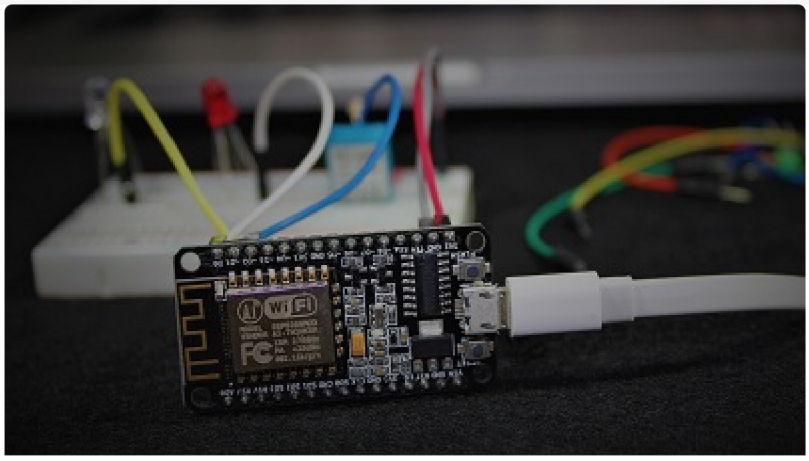
[eBook] Build Web Servers with ESP32 and ESP8266 (2nd Edition)

Build Web Server projects with the ESP32 and ESP8266 boards to control outputs and monitor sensors remotely. Learn HTML, CSS, JavaScript and client-server communication protocols **DOWNLOAD »**

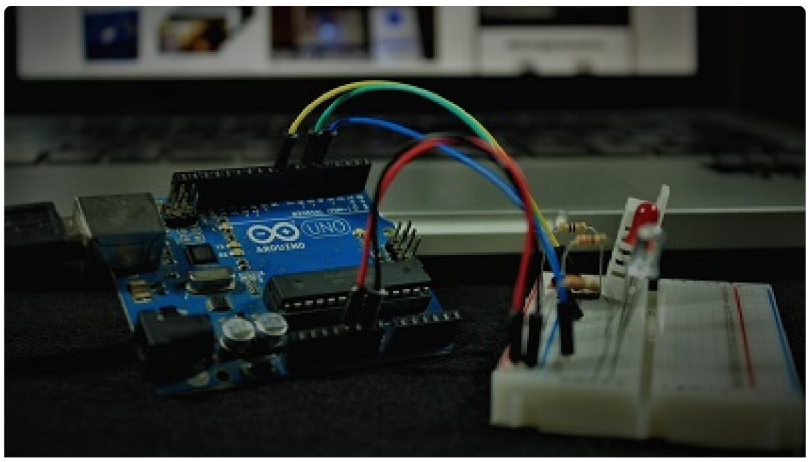
Recommended Resources



Build a Home Automation System from Scratch » With Raspberry Pi, ESP8266, Arduino, and Node-RED.



Home Automation using ESP8266 eBook and video course » Build IoT and home automation projects.



What to Read Next…

ESP8266 NodeMCU: Getting Started with Firebase (Realtime Database)

MicroPython: ESP32/ESP8266 Access Point (AP)

ESP32/ESP8266 with HTTPS and SSL/TLS Encryption: Basic Concepts

8 thoughts on “Access Node-RED Dashboard from Anywhere using Digital Ocean”

Otto Ziegler

February 26, 2021 at 11:29 pm

I have tried following this tutorial, but have failed miserably. This is my time trying to set up anything on Ubuntu and I find it absolutely unworkable. Console window does not allow Highlighting, Copying or Pasting, so password had to be painstakingly copied by hand & then re-entered. Now when I try to open the Node Red terminal I get nothing.

Time to try something else – this is ridiculous.

Reply

Roberto

October 20, 2022 at 1:56 pm

If you do the Initial Server Setup with Ubuntu 20.04. tutorial you can use putty to connect to your server.

Reply

Pradeep Nayak

September 25, 2021 at 4:42 am

This is very helpful, followed the steps and I have node-red up and running on VPS

Reply

Thorsten

October 30, 2021 at 7:11 am

Hi,

I released a Node/App/Service called Remote-RED. With this you can access your Dashboard also from anywhere, but without having your Node-RED installed in the cloud. So also all other IoT/Smarthome connections are possible, that requires a local Node-RED.

Greetings!

Reply

Harvey Liewin

November 22, 2021 at 4:39 am

Hi,

i have encountered a problem when I'm accessing the node-red software for the first time, it shows "Deceptive site ahead", is there any way to fix this ?

thank you

Reply

Taufiq

March 1, 2022 at 1:48 am

Hi Rui, Thankyou for your great tutorial. If I register to Digital Ocean for 1 month only, Should I charged ?

Reply

dave

March 3, 2022 at 5:21 am

great tutorial thanks for posting

Reply

Chuk Salmon

March 28, 2022 at 12:34 pm

Installing globally on a current Ubuntu 20.04 puts it in /usr/bin so this is incorrect a fails.

ExecStart=/usr/local/bin/node-red –max-old-space-size=128 -v

It’s incorrect in the Digital Ocean instructions also….

Regards,

Reply

Leave a Comment

Name *

Email *

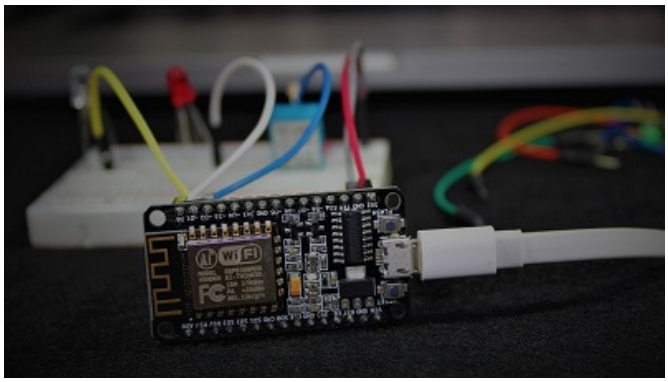
Website

☐ Notify me of follow-up comments by email.

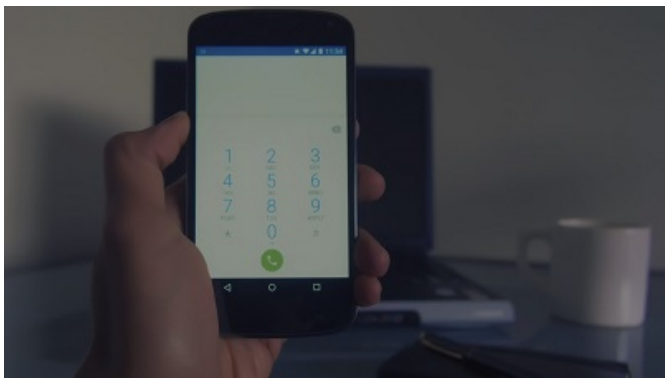
☐ Notify me of new posts by email.

Post Comment

Visit Maker Advisor – Tools and Gear for makers, hobbyists and DIYers »



Home Automation using ESP8266 eBook and video course » Build IoT and home automation projects.



Build Web Servers with ESP32 and ESP8266 » boards to control outputs and monitor sensors remotely.