Set up Home Assistant with a Raspberry Pi

Created by Richard Albritton

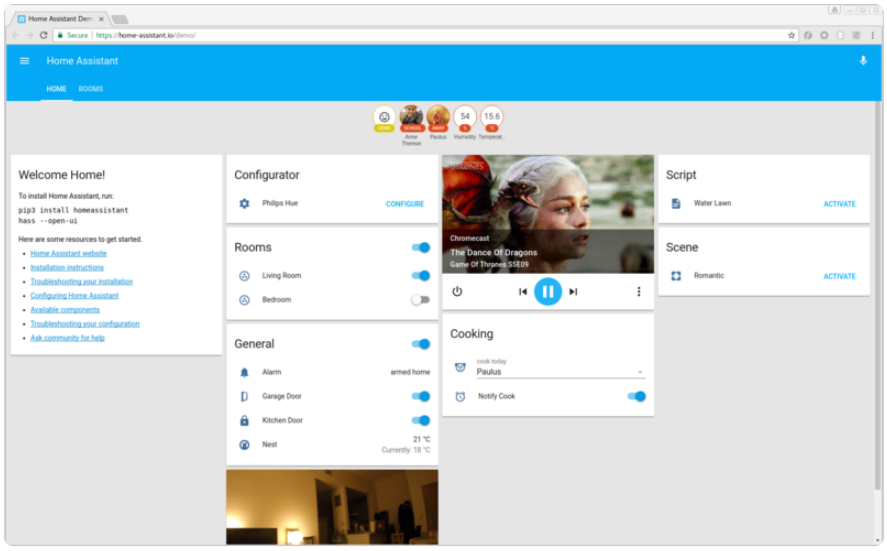
https://learn.adafruit.com/set-up-home-assistant-with-a-raspberry-pi

Last updated on 2025-12-02 02:54:07 PM EST

Table of Contents

Intro

Home Assistant is an open source operating system for a localized Smart Home Hub. Basically it works like IFTTT or Samsung Smart Things, but without having to send your data out onto the internet. This means that you have total control over your data, limit the amount of internet traffic from your smart devices, and tighten up security. Along with all of those things Home Assistant also allows you to create complex Scripts and Automations that blow other such Smart Home systems out of the water.



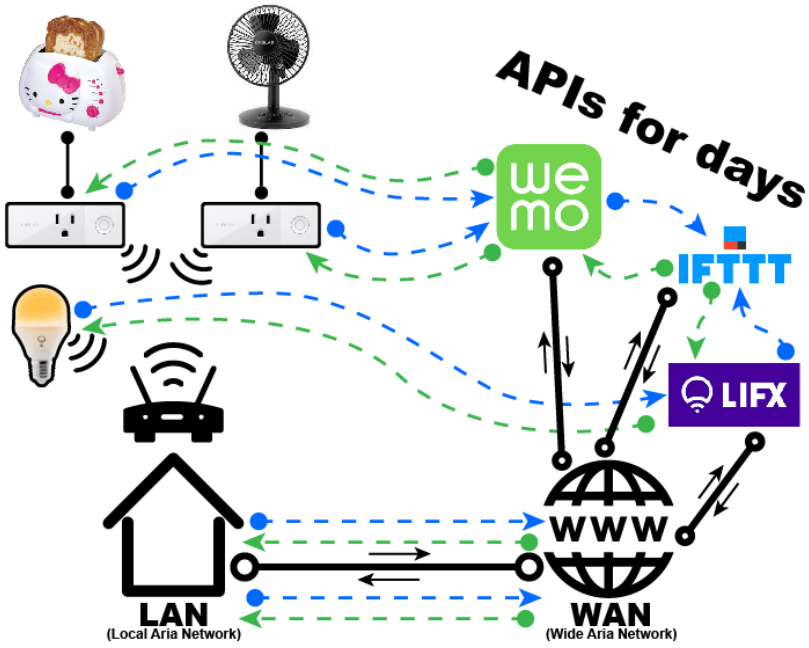
Support for over 1400 devices means that just about every type of Smart Home device will likely work from consumer products like Hue, Lifx, Google Home, Alexa, Ecobee, Z-Wave, WeMo, IKEA Trådfri, and so many DIY devices that use things like Python or MQTT. Most IOT projects that use the ESP8266 or ESP32 can be tied into this system as easily as connecting to Adafruit.io.

> This guide will walk you through setting up the Home Assistant server onto a Raspberry Pi using Hass.io.
>
> Next we will configure the server using the web interface.
>
> Then it is time to install goodies like MQTT and Node Red
>
> Last we will go over some things to look for in devices that work best with this kind of system.

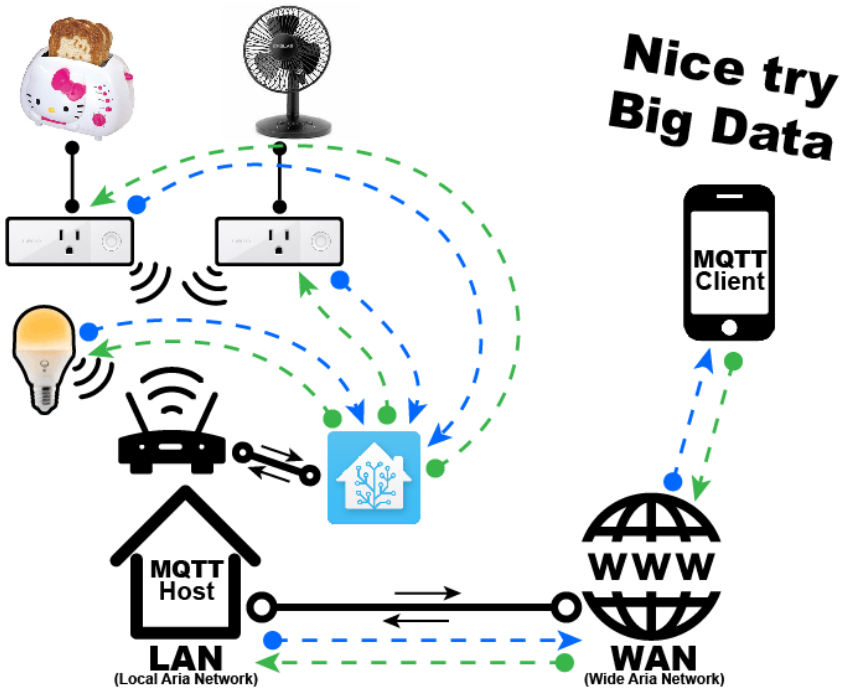
But Why?

There are many reasons why having a Smart Home Hub makes sense, but the big ones have to do with security, data storage, and network traffic.

Let's start by showing the network map of a typical Smart Home with no Hub.



Now here is an example of the same network using a local Hub that still allows remote access:



A Smart Home Hub will not only cut down on the internet data usage, but it also eliminates the need for installing multiple apps to control your devices. Don't even get me started on APIs that are meant to get APIs to talk to other APIs.

Choosing devices that support local control and allow the option of an API will make things work smoothly down the road while API dependent devices will limit your options. The real magic happens when you get communications standards involved like Z-Wave, MQTT, and Zigbee.

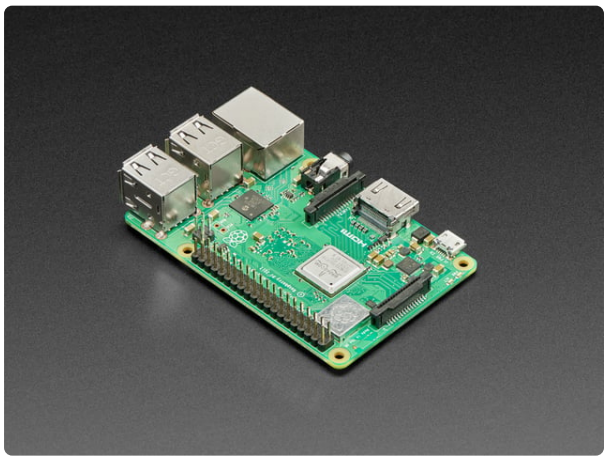
Raspberry Pi Server Setup

This guide will show installing Home Assistant onto a Raspberry Pi 3B+ , 4, or 5 and connecting it to a home network using Ethernet.

Basically this is the easiest and best way to set up a Smart Home Hub like this as the Disk Image used automatically configures the server when connected to the internet through Ethernet. By using a wired connection for the server, you can be sure that you get the fastest and most reliable connection possible for your Smart Home.

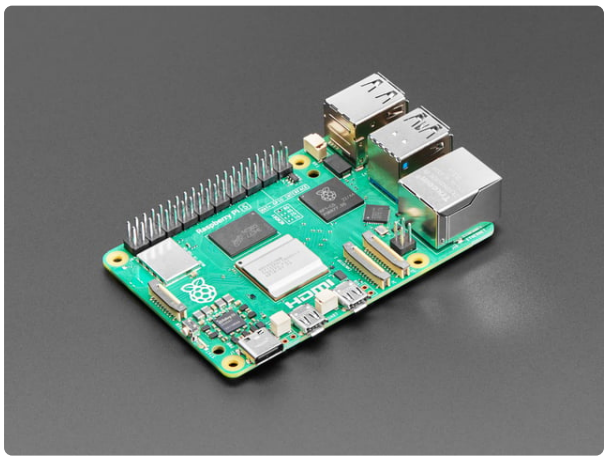
To get this going, you will create a boot disk using a microSD card. This microSD card will then go into the Raspberry Pi, which is then connected to your home Router via Ethernet. After about 20 minutes, you should be able to access the user interface from any device connected to your home WiFi by simply going to http://homeassistant.local:8123/

Here are the things we will need to get your server up and running:



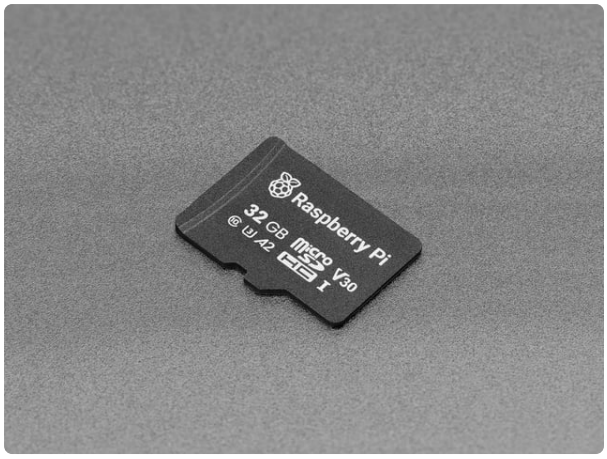
Raspberry Pi 3 - Model B+ - 1.4GHz Cortex-A53 with 1GB RAM
The Raspberry Pi 3 Model B is the most popular Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 3...
https://www.adafruit.com/product/3775

Raspberry Pi 4 Model B - 4 GB RAM
The Raspberry Pi 4 Model B is the newest Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 4 better...
https://www.adafruit.com/product/4296



Raspberry Pi 5 - 4 GB RAM
The Raspberry Pi 5 is the newest Raspberry Pi computer, and the Pi Foundation knows you can always make a good thing better! And what could make the Pi 5 better than the...
https://www.adafruit.com/product/5812

setup will also work with Raspberry Pi versions 4 and 5 as they are now ɔorted by Home Assistant. Always check at https://www.home-assistant.io/allation/raspberrypi/ to check if your hardware is currently supported.

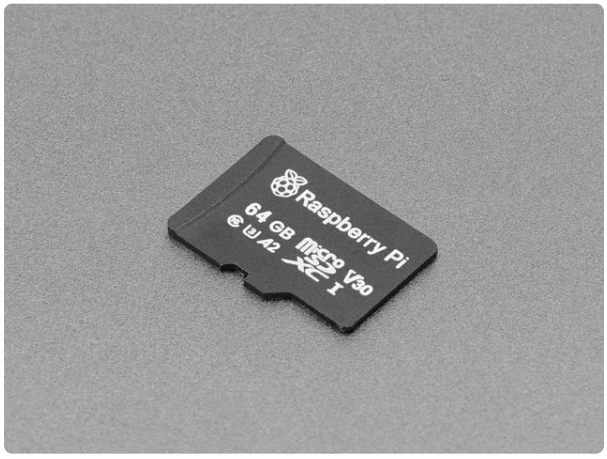


Official Raspberry Pi A2-Class microSD Card - 32GB Blank

Optimise data transfer speeds on your Raspberry Pi computer with an official Raspberry Pi 32GB microSD card. Rigorously tested to ensure optimal performance on...

https://www.adafruit.com/product/6010

or



Official Raspberry Pi A2-Class microSD Card - 64GB Blank

Optimise data transfer speeds on your Raspberry Pi computer with an official Raspberry Pi 64GB microSD card. Rigorously tested to ensure optimal performance on...

https://www.adafruit.com/product/6011

er MicroSD cards will work as well, but 32GB is a good baseline microSD . For better operation it is recommended that you use a 64GB Application s 2 microSD card.

Ethernet Cable - 3 ft long

We have so many Internet-connected goodies in the shop, we figured it's time to carry a cable so you can easily connect them up! This cable is 3 feet long, comes in Adafruit black...

https://www.adafruit.com/product/995

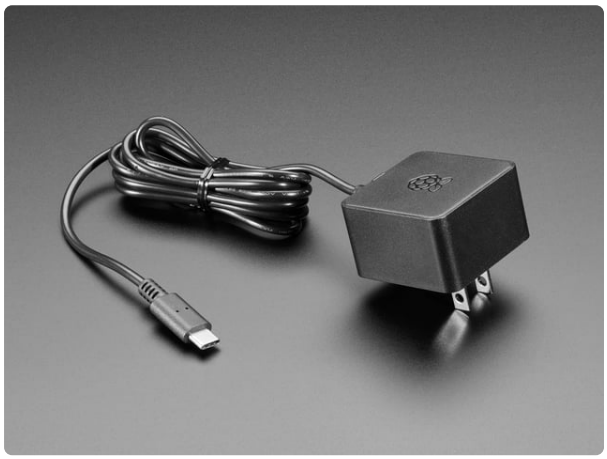
For Raspberry Pi 3, use this power supply:

5V 2.5A Switching Power Supply with 20AWG MicroUSB Cable

Our all-in-one 5V 2.5 Amp + MicroUSB cable power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that's...

https://www.adafruit.com/product/1995

For Raspberry 4, use this power supply:



Official Raspberry Pi Power Supply 5.1V 3A with USB C

The official Raspberry Pi USB-C power supply is here! And of course, we have 'em in classic Adafruit black! Superfast with just the right amount of cable length to get your Pi 4...

https://www.adafruit.com/product/4298

For a Raspberry Pi 5, use this power supply:

Official Raspberry Pi 27W PD Power Supply 5.1V 5A with USB C

The official Raspberry Pi PD USB-C power supply is here! Superfast with just the right amount of cable length to get your Pi 5 projects up and running! With true Power Delivery...

https://www.adafruit.com/product/5814

As for the Software tools, you will want the following:

SD Card Formatting tool:

https://adafru.it/FKd

SD Card Boot Disk tool:

https://adafru.it/EMc

Installing Home Assistant

Download the latest disk image for Hass.io from the Home Assistant page. We will be using the Raspberry Pi 3B+ for this project so that is the one you want to get, but if you want to use another Pi version, pick that one from the downloads page.

https://adafru.it/1axk

If needed, use the SD Card Formatter to format the MicroSD Card and remove any partitions.

Use Etcher to install the boot disk image onto your microSD card.

Once finished, eject the microSD card and put it into the microSD slot for the Raspberry Pi.

Use the Ethernet cable to connect your Raspberry Pi into an available network port on the back of your WiFi Router.

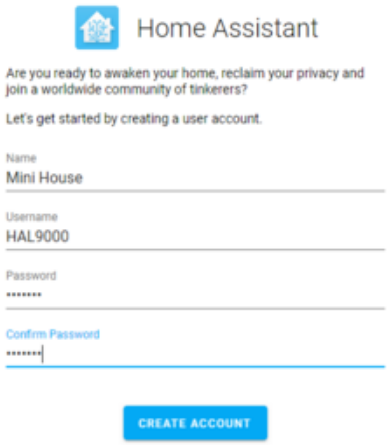
Plug the Raspberry into power using the Micro USB connector (Pi 3) or USB C connector (Pi 4 and 5).

It can take up to 20 min for Home Assistant to install. After that time you should be able to access the user interface by going to http://homeassistant.local:8123/ from and web browser on your local network WiFi or Ethernet connection.

can connect the Raspberry Pi to your internet using WiFi if you do not
e access to an Ethernet connection. However this process is much more
plex and should be avoided if possible.

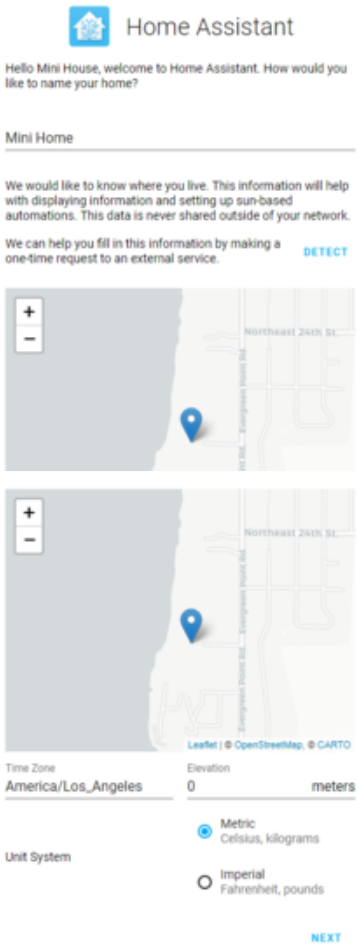
https://adafru.it/FKf

Home Assistant Setup



Once connected, you will be prompted to create a user account. This will serve as the administrative account for your Smart Home Hub so be sure to remember the information that you use for the login. There is no password recovery system in place.

Once finished, click **CREATE ACCOUNT**.

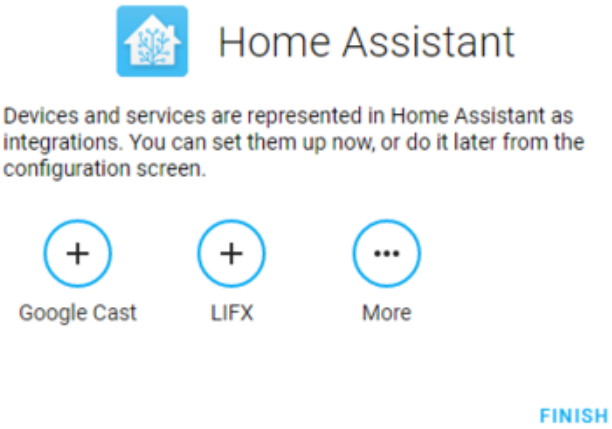


Now you will be asked to name your home and set it's location.

The Location part helps with things like weather reports, sun position data, and other things that can be added latter.

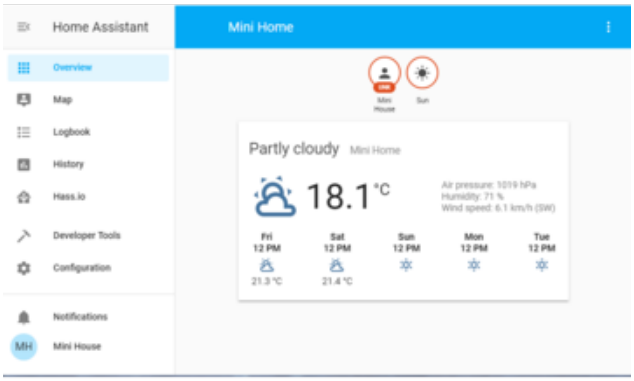
You can also set your **Time Zone**, **Elevation**, and unit of measurements.

Click **NEXT** when finished.



If you have any smart devices already connected to your network, Home Assistant may detect them and let you set them up. Devices like Philips Hue, Lifx, WeMo, Google Cast, and a few others are quite easy to set up using this user interface. You may choose to set these up now or do so later using the Integrations tool in the Configurations menu.

Click **FINISH** when done.



Now you are all set up and any smart devices that you added from the previous screen will now be available and ready to control using Home Assistant.

On the left you should see the Menu where you can see a Map view, Logbook, History, Hass.io, Configuration, and some Developer tools.

The Configuration menu is where you can add many supported devices and services using Integrations. You can also add new user accounts including non Admin users that can not access the full menu structure.

The General menu is where you can change location info along with reloading and restarting the core services if needed. Area Registry lets you define rooms for better use with Google Home, Alexa, or Home Kit. Automation allows you to create complex automations using a fairly easy form based GUI. Script is similar to Automation, but there is no event trigger. You use Script mostly to create complex combinations of tasks that you want to call repeatedly with an automation or with a button in the user interface.

MQTT Setup

What it MQTT?

MQTT is a kind of secure machine-to-machine message protocol that is made for the internet of things (IoT). For MQTT to work you need to have a computer running as an MQTT Broker and all other devices connect to that computer as an MQTT Client.

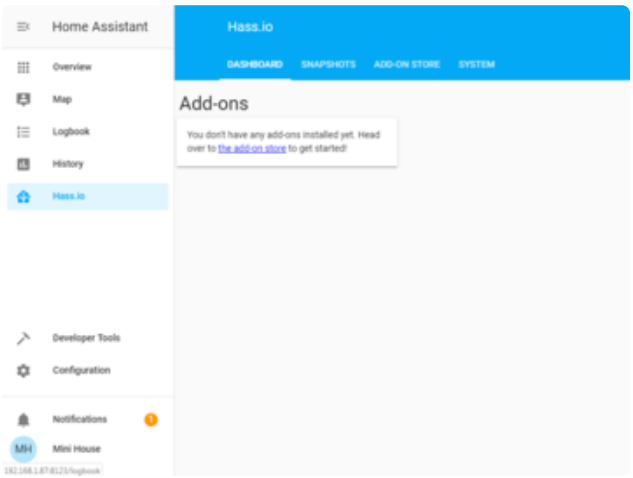
MQTT Clients can Subscribe to a Feed on the MQTT Broker and will receive an update if the content of the Feed it is Subscribed to changes. Publishing to a Feed is also allowed by the MQTT Client and the MQTT Broker will pass that published data along to any device that is Subscribed to that particular Feed. Any MQTT Client can Subscribe and Publish to a Feed, but they can also just Subscribe or Publish depending on what your device needs to do.

Adafruit.io hosts an MQTT server, as do many other companies, and they work very well for testing out devices or creating global networks of devices. However, these

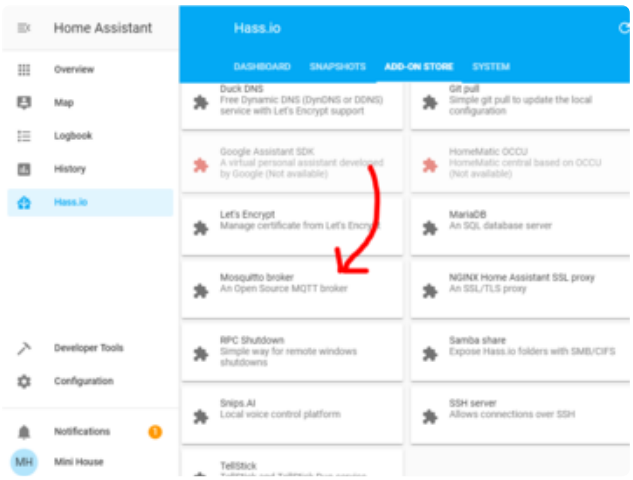
servers typically have Feed limits or subscription fees that may not always jell with what you want to do.

So why not run your own MQTT Broker using Mosquitto? It is not quite as powerful as Adafruit.io, but it works just as well for most non-industrial projects. Plus it is free and secure.

Install Mosquitto Broker

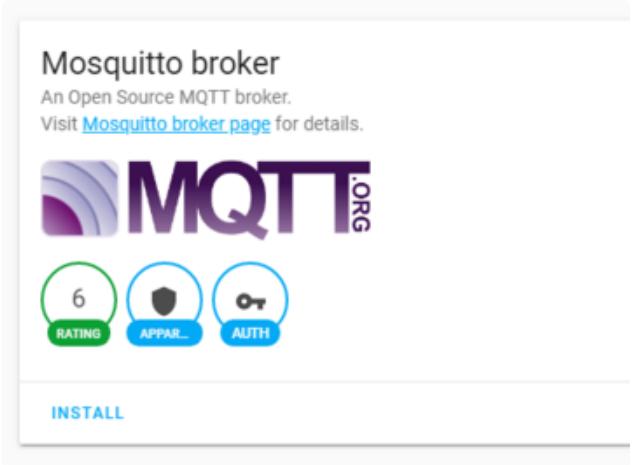


From the **Menu**, click **Hass.io**



Go to the **ADD-ON STORE**

From the **ADD-ON STORE**, click on **Mosquitto broker**.



Click on **INSTALL.**

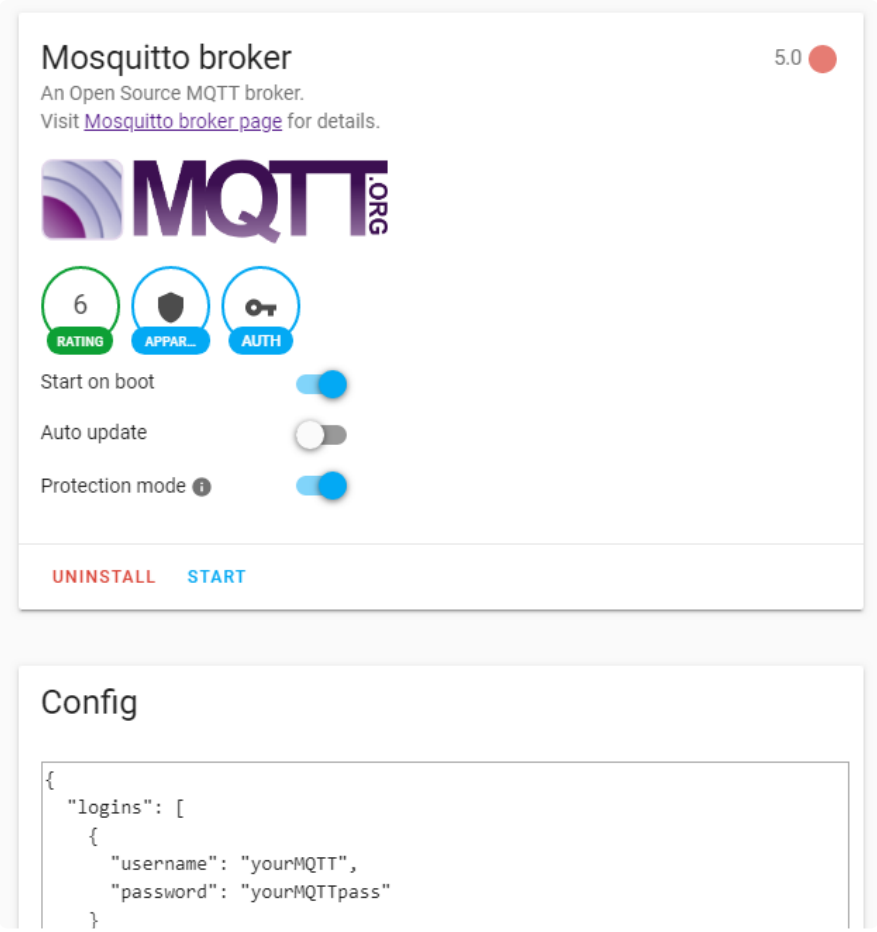
Add the following text to the **Config** text box.

```
        {
  "logins": [
    {
      "username": "yourMQTT",
      "password": "yourMQTTpass"
    }
  ],
  "anonymous": false,
  "customize": {
    "active": false,
    "folder": "mosquitto"
  },
  "certfile": "fullchain.pem",
  "keyfile": "privkey.pem",
  "quiet_logs": true
}
```

Change the **username** and **password** to whatever you want to use for connecting to the MQTT Broker.

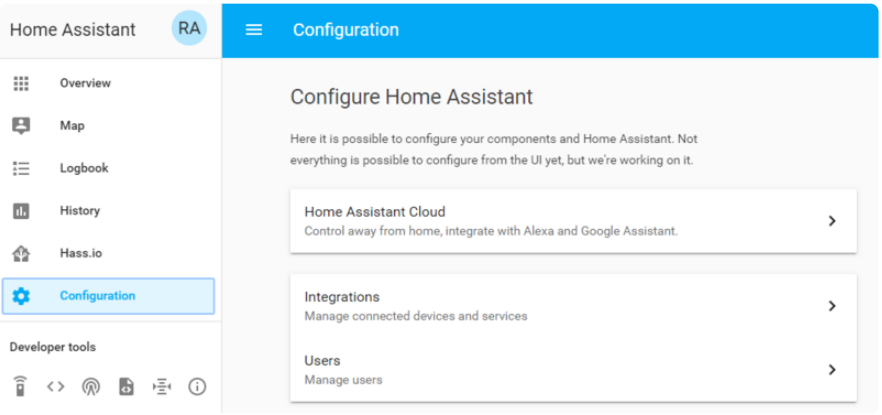
Click on **SAVE** to save your new settings.

Now press **START** to start your new MQTT Broker.



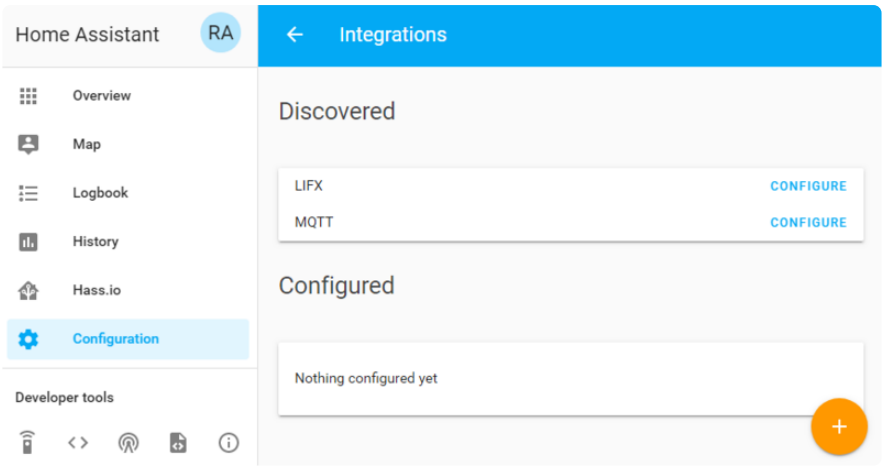
Click on the **Configuration** menu.

Select **Integrations** from this menu.

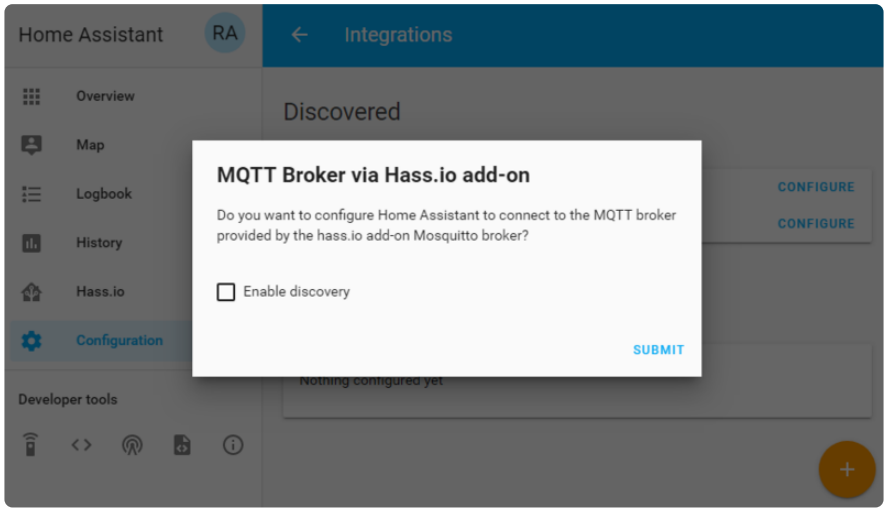


You should see a list of devices waiting to be set up.

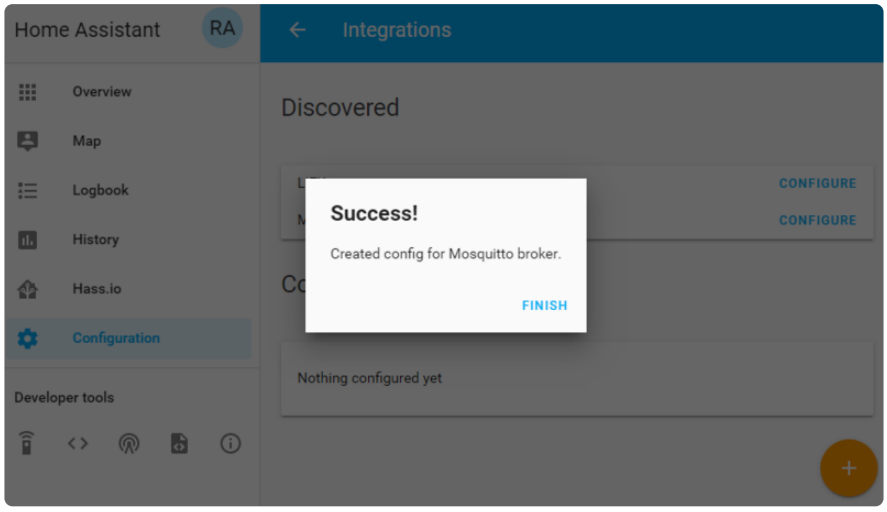
If you see **MQTT** in this list, click **CONFIGURE**. If **MQTT** is not in the list, go to **Configuration** -> **General** -> **RESTART**. Once reconnected, go back to Integrations and **MQTT** should be there.



Click **SUBMIT**

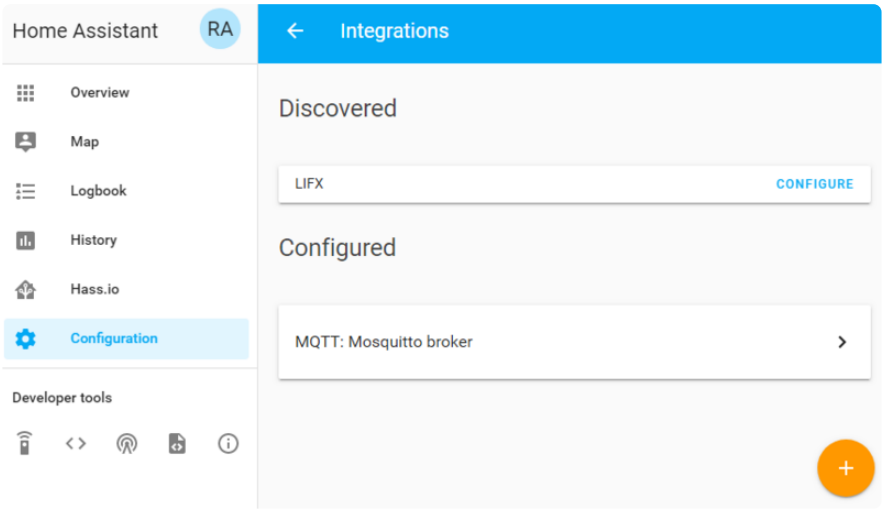


Click **FINISH**



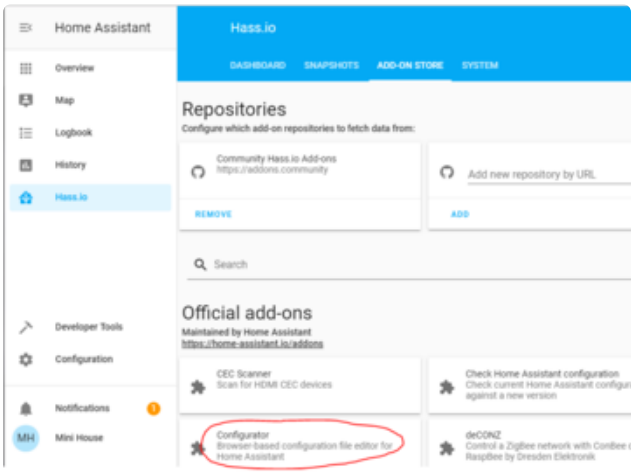
You now have your own MQTT server linked to the Home Assistant server. From any MQTT client, you can interact with feeds using the Home Assistant IP address as the Host and the MQTT Username and Password that you set up.

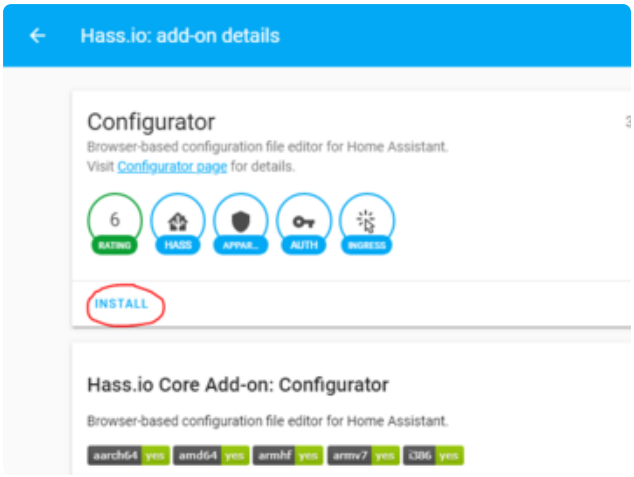
This will only work within your local network unless you use Port forwarding using the MQTT port to point internet traffic to your Home Assistant IP address. I do not recommend this unless you have some experience with network administration as it could pose a security risk if not set up properly.

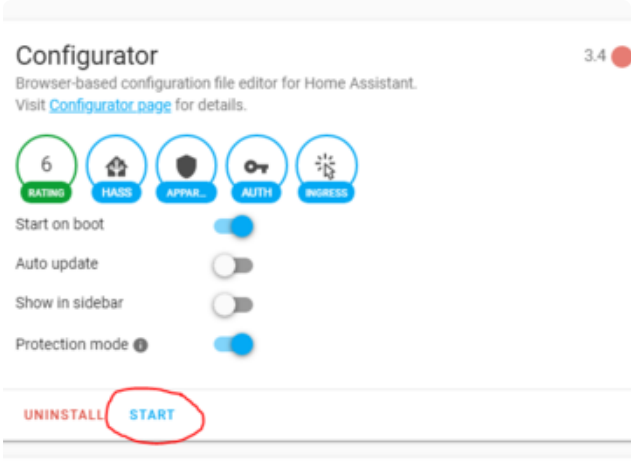


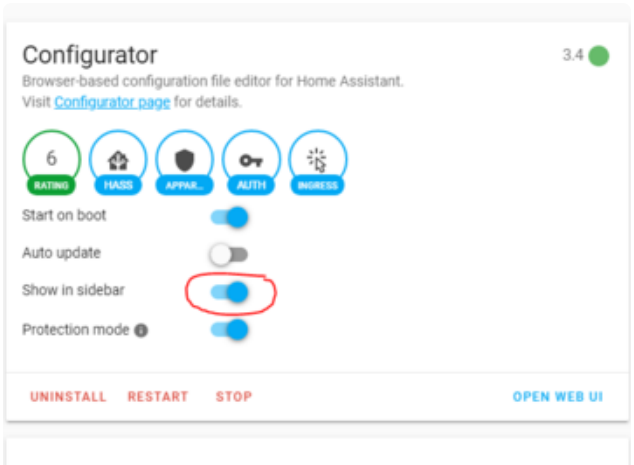
Configurator

While many consumer products can be set up using the Interactions tool, other devices like those that use MQTT will need to be set up in the Configuration.YAML file. The code that goes in there will depend on what you are setting up and we will get to that after setting up the Configurator add-on.









1. From the ADD-ON STORE, find Configurator and click it.
2. Click the INSTALL link.
3. Once installed click START.
4. Click to turn on the Show in Sidebar option.

You can now access the file system by clicking on the OPEN WEB UI link or through the Configurator menu option on the left.

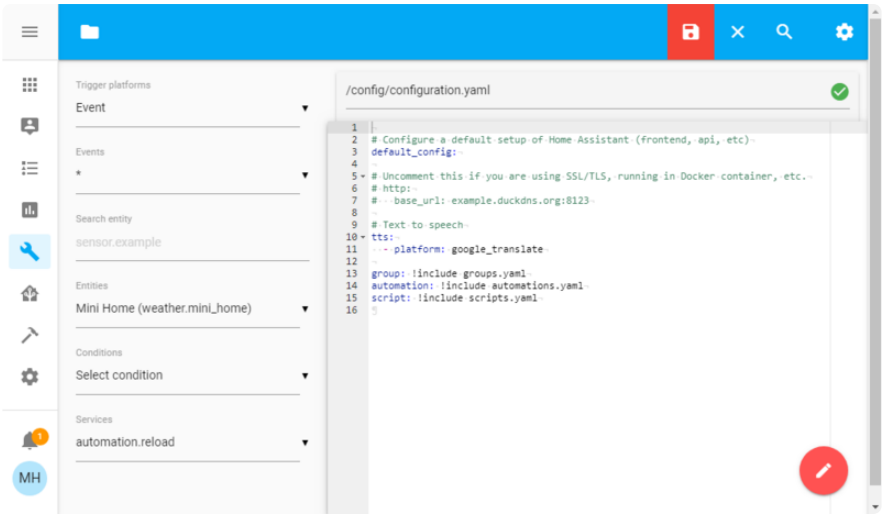
Now you are ready to edit files for Home Assistant. Next you will find and edit the Configuration.YAML file so that you can add all sorts of fun custom Arduino and MQTT devices to the user interface.

Using the Configurator

Now let us go and have a look at the Configuration file that you will likely be spending most of your setup time in.

1. From the **Configurator** panel, click on the folder Icon on the top left.
2. This should show a list of files from the **config/** directory.
3. Click the **configuration.yaml** file to open it.
    - Note: If you get a popup titled **Unsaved Changes**, click the **CLOSE FILE** option.

You should now see the configuration.yaml.



There is not much here but you can add things like lights, switches, and sensors that use MQTT or other communication methods to send data. You can find some examples of code to add to this in most of the documentation for Home Assistants many available Components.

https://adafru.it/FKg

Here is an example of what the code looks like with a few MQTT devices added to it:

```
# Configure a default setup of Home Assistant (frontend, api, etc)
default_config:

# Uncomment this if you are using SSL/TLS, running in Docker container, etc.
# http:
#   base_url: example.duckdns.org:8123

# Text to speech
tts:
  - platform: google_translate

sensor:
  # Weather prediction
  - platform: yr
  - platform: mqtt
    name: "Light Sensor"
    state_topic: "house/lux"
    unit_of_measurement: 'Lux'
    icon: mdi:brightness-6
  - platform: mqtt
    name: "Door Sensor"
    state_topic: "house/door"
    icon: mdi:door

fan:
  - platform: mqtt
    name: "Fan"
    state_topic: "house/fan"
    command_topic: "house/fan"
    speed_state_topic: "house/fan/speed"
    speed_command_topic: "house/fan/speed"
    qos: 0
    payload_on: "ON"
    payload_off: "OFF"
    payload_low_speed: "low"
    payload_medium_speed: "medium"
    payload_high_speed: "high"
    speeds:
      - low
      - medium
      - high

light:
  - platform: mqtt
    name: "Light 1"
    state_topic: "house/led/one"
    command_topic: "house/led/one"
    brightness_state_topic: "house/led/one/brightness"
    brightness_command_topic: "house/led/one/brightness"
    rgb_state_topic: "house/led/one/color"
    rgb_command_topic: "house/led/one/color"
    on_command_type: first
    state_value_template: "{{ value_json.state }}"
    brightness_value_template: "{{ value_json.brightness }}"
    rgb_value_template: "{{ value_json.rgb | join(',') }}"
    qos: 0
    payload_on: "ON"
    payload_off: "OFF"
    optimistic: false

lock:
  - platform: mqtt
    name: Frontdoor
    state_topic: "house/lock"
    command_topic: "house/lock"
    payload_lock: "LOCK"
```

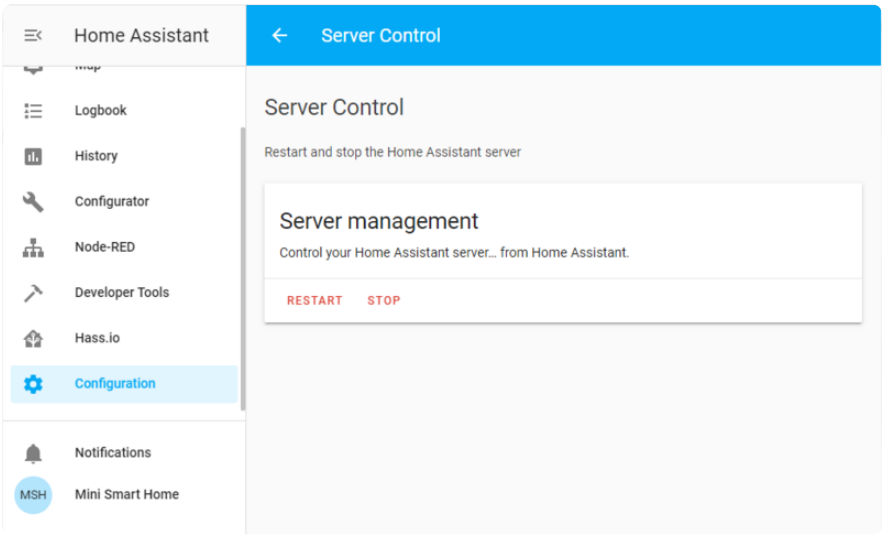
```
    payload_unlock: "UNLOCK"
    optimistic: true
    qos: 1
    #retain: true
    value_template: '{{ value.x }}'

group: !include groups.yaml
automation: !include automations.yaml
script: !include scripts.yaml
```

This tool will verify your YAML code is formatted correctly by displaying a green check mark near the top right corner of the screen. A red ! in the top right corner will tell you that something is wrong with your code and you can click on it to see what will need to be fixed before you are finished.

Do not restart Home Assistant if you have any errors in your YAML files. Home stant will not restart if it finds errors because these errors will prevent gs from starting up properly. Do not try to force a restart unless you are that your YAML files are error free because they are hard to fix without Home Assistance UI unless you have added and configured Samba or SSH. ALWAYS VERIFY YOUR CODE BEFORE SAVING OR RESTART!

When you have checked your YAML code and saved, go to Server Control from the Configuration menu and click RESTART.

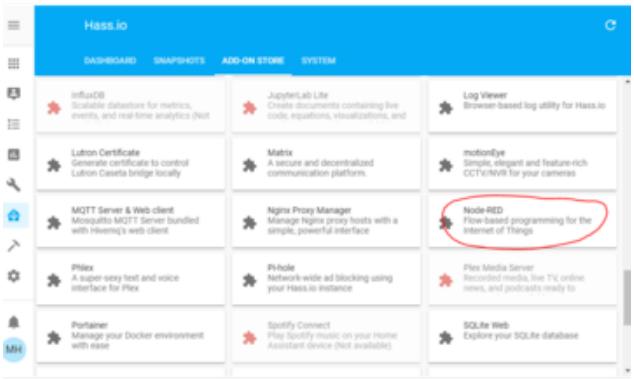


After a few seconds, you will see some text appear that will say **Connection lost. Reconnecting...** at the bottom left of the window. When that text disappears, the server is back online and your changes have been loaded.

Node Red

Node RED is a Flow based programming interface that is somewhere between block and text based coding. It can be an easy way to program automations for Home Assistant that are a bit more advanced than what can be done with the Automations menu.

Add Node-RED

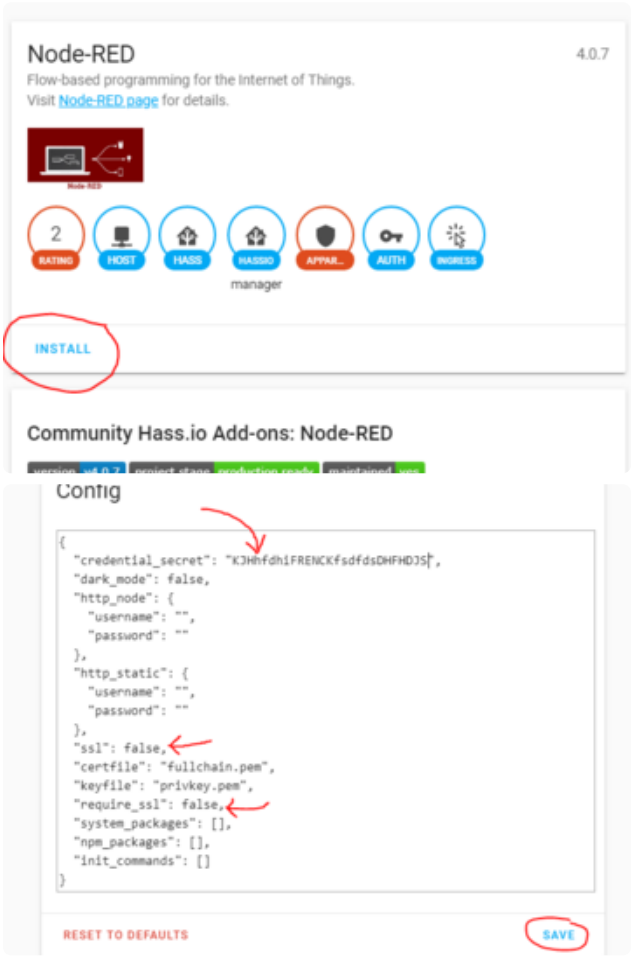


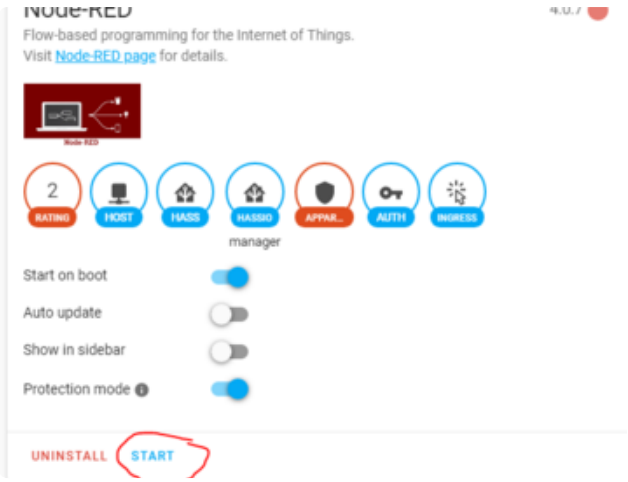
From the Hass.io **ADD-ON STORE**, scroll down to find **Node-Red** and click on it.

1. Click the **INSTALL** link.
2. Once Installed, scroll down to the **Config** section.
3. Set a **credential_secret**, which is used to encrypt sensitive data. This is just a password, which you should save in a secondary location.
4. Next set **ssl** and **require_ssl** to **false**.

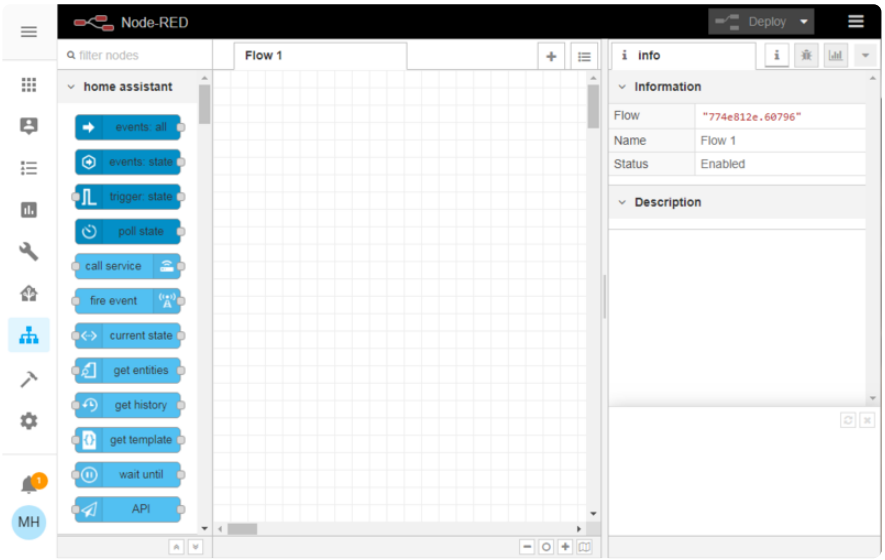
   Note: SSL can be added, more information in the documentation (https://adafru.it/FKh).

5. Click the **SAVE** link
6. Now scroll up and click the **START** link.
7. This can take a few moments to start up, but you can check the log at the bottom of the page to see when Node-RED is up and running.
8. If desired, turn on the **Show in sidebar**
9. Once finished, you can now click on the **OPEN WEB UI** link or the **Node-RED** menu link to open Node-RED.



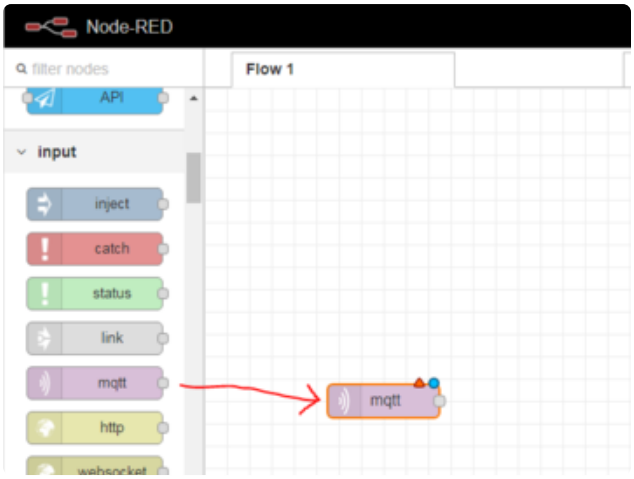


u get an error "502: Bad Gateway" this could indicate that Node-RED is fully loaded. Check the Log to see Node-RED is still loading or to read any rs while loading.



Setup Node-RED MQTT Client

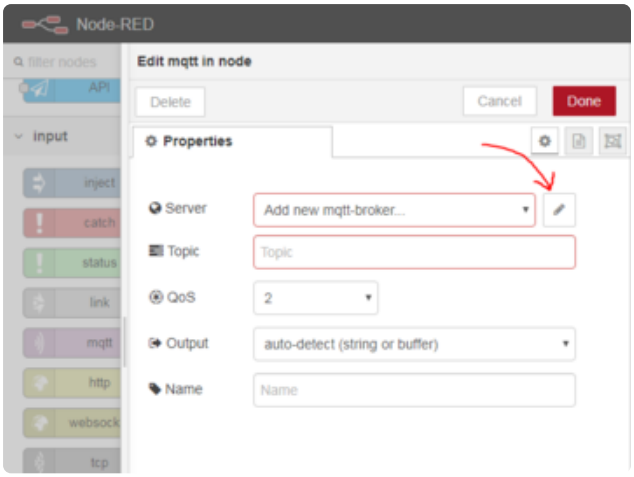
Since we have already installed the MQTT Broker we may as well set up Node-RED to use that for sending and receiving data from connected devices right?



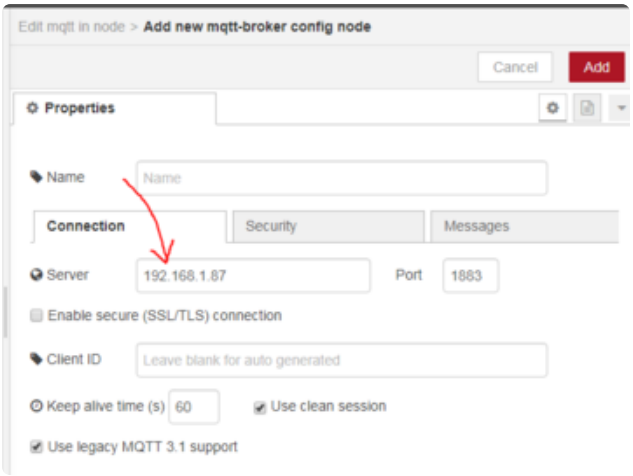
Scroll down to find the **mqtt** node under **input** or **output**.

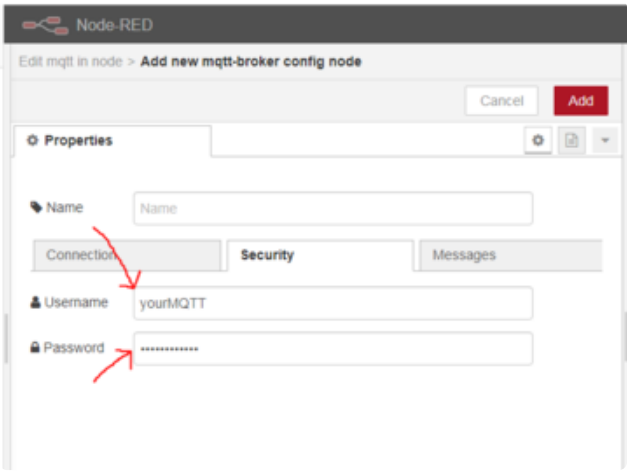
Move the **mqtt** node to **Flow 1**

Double click on the **mqtt** node to open it's **Properties** window.



Click the **Edit** icon to add a new MQTT broker.





Under the **Connection** tab, add your MQTT **Server** address.

This will be the same IP address as your Raspberry Pi unless you are using another MQTT server like Adafruit.IO

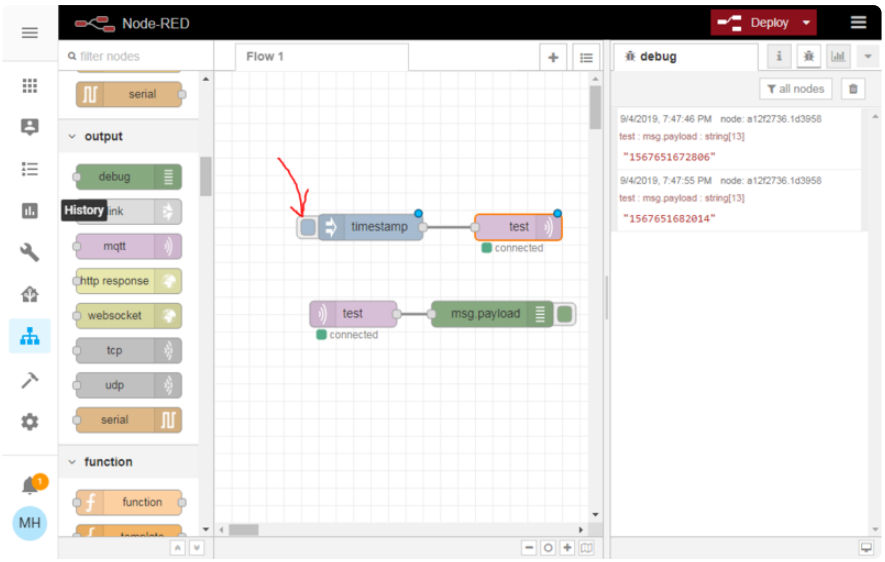
Next click the **Security** tab.

Add the **Username** and **Password** that you set for your MQTT server.

Now click **Add**

u misplaced your MQTT credentials, they can be found by going to the s.io menu and click on Mosquitto broker. The Password and Username uld be in the Config field.

That will add your connection and your can now set a topic to interact with. As a test, I will usually add and **inject** node to send a time stamp to the **mqtt** node under **output** with a **Topic** of "**test**". Then I will get another **mqtt** node from the **input** section, set it's **Topic** to "**test**" and connect it to a **debug** output.



When you click the tab next to the timestamp **inject** node it will send the Unix time to the topic "test" via MQTT. Once that happens, the **mqtt** node set as **input** will receive that update and send it to be printed in the **debug** log.

Doing more with Home Assistant

There are so many things that you can do with Home Assistant and the project keeps expanding. There is talk about machine learning integration and new user account control in the development pipeline. Here are some links to documentation on the Home Assistant web site.

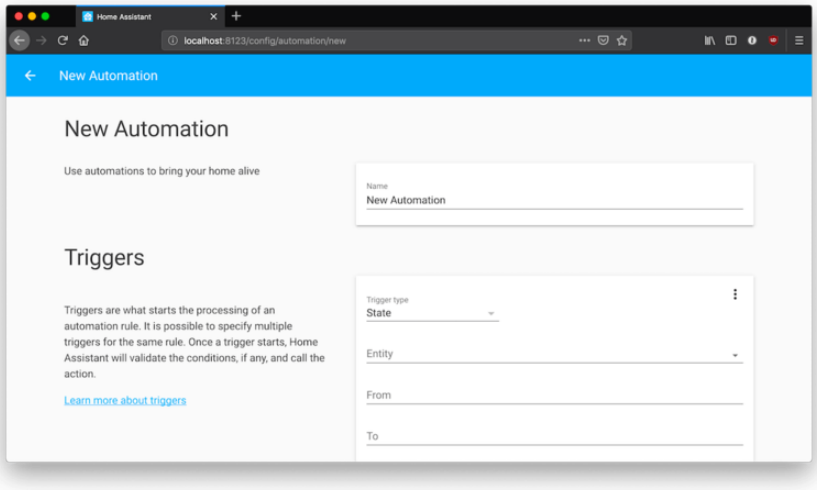
Lovelace UI

The user interface that is used for Home Assistant is called Lovelace and not only is it nice looking, but it easy easy to customize through any computer web browser.

https://adafru.it/FPG

Automations UI

There is a tool that can be used to create some rather complex Automations in the Configuration menu. This is the easiest way to create some rather complex automations without using Node-RED or editing YAML files in the automations.yaml file.



https://adafru.it/FPH

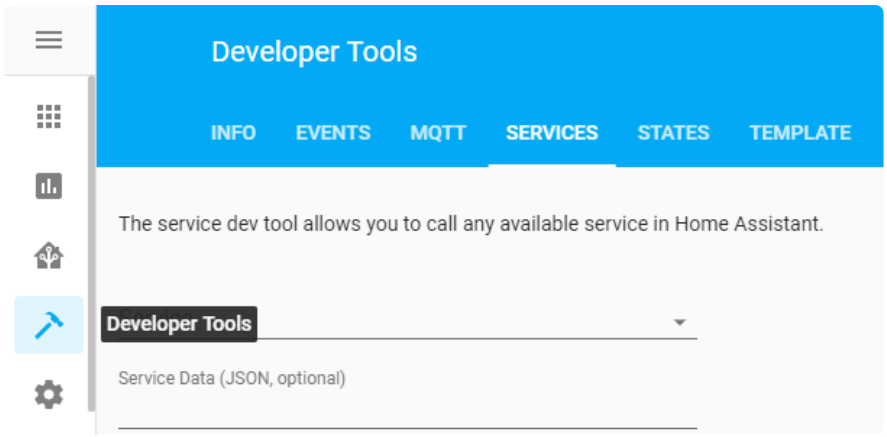
Scripts UI

Scripts will allow you to use one command to do a bunch of things at once. This can be handy if you have a series of actions that will be reused in automations.

https://adafru.it/FPI

Development Tools

These tools are very useful for testing and troubleshooting devices, but also for seeing what data is needed for automations and scripts. The Services tool is very helpful for creating and testing the JSON formatted Service Data used with more complex devices like lights and media players.



https://adafru.it/FPJ

Even More Info

I do highly recommend keeping up to date with all of the cool stuff happening with Home Assistant by checking out their official website.

https://adafru.it/DQw