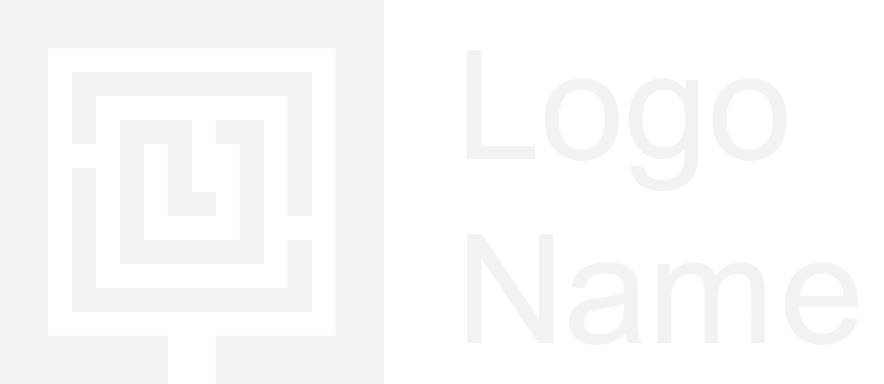


|  |
| --- |
| Opensource Smart Home Assistant with privacy by design |
|  |
| July 6  Research and developed by: Billy Ng, Alex Tay, Sook, Zul Fadil |



Contents

[Product Objective 4](#_Toc76222212)

[Problem Statement 6](#_Toc76222213)

[Solution 7](#_Toc76222214)

[Architecture and Design 8](#_Toc76222215)

[Key consideration: 8](#_Toc76222216)

[Hardware considerations: 8](#_Toc76222217)

[Opensource Software considerations: 8](#_Toc76222218)

[Hardware evaluation and finalization 9](#_Toc76222219)

[Opensource Voice Assistant software evaluation and finalization: 9](#_Toc76222220)

[Home Automation software evaluation and finalization: 9](#_Toc76222221)

[Opensource Security software evaluation and finalization: 9](#_Toc76222222)

[Rhasspy architecture 10](#_Toc76222223)

[Home Assistant architecture 11](#_Toc76222224)

[Initial basic setup cost 12](#_Toc76222225)

[What’s next? 15](#_Toc76222226)

[Building Blocks as per below simple steps: 15](#_Toc76222227)

[Hardware preparation 16](#_Toc76222228)

[Getting Started 17](#_Toc76222229)

[Overview of overall process 17](#_Toc76222230)

[Install, Secure, Configure Home Assistant, Rhasspy and others Add-On 18](#_Toc76222231)

[Write image file to your installation media 18](#_Toc76222232)

[Start up your Raspberry Pi 4 B and Home Assistant 20](#_Toc76222233)

[Onboarding Home Assistant 21](#_Toc76222234)

[Key Add-On needed 26](#_Toc76222235)

[Securing Home Assistant 27](#_Toc76222236)

[Setup Home Assistant security 28](#_Toc76222237)

[Secured connection to Home Assistant using DuckDNS 30](#_Toc76222238)

[Setting Raspberry Pi 4 B Static IP running Home Assistant 31](#_Toc76222239)

[DuckDNS Port Forwarding 32](#_Toc76222240)

[Setting Up DuckDNS Domain 33](#_Toc76222241)

[Installing DuckDNS Add-on 35](#_Toc76222242)

[DuckDNS Home Assistant Configuration 36](#_Toc76222243)

[DuckDNS entry on Configuration.yaml 37](#_Toc76222244)

[Setup Maria DB follow by Nginx Proxy Manager addon 39](#_Toc76222245)

[Install and configure WireGuard VPN 45](#_Toc76222246)

[Install, configure and test MQTT broker and Voice Assistant 46](#_Toc76222247)

[Install and configure Mosquitto MQTT broker 46](#_Toc76222248)

[Install and configure Rhasspy the Voice Assistant 54](#_Toc76222249)

[Quick tutorial of Rhasspy Web Interface 55](#_Toc76222250)

[Automation 63](#_Toc76222251)

[Create new Event Automation in Home Assistant 63](#_Toc76222252)

[Wrap-up 70](#_Toc76222253)

[Acknowledgement 72](#_Toc76222254)

[About me 73](#_Toc76222255)

[References 74](#_Toc76222256)

# Product Objective

|  |
| --- |
| Key objective is to develop a “**Opensource Smart Home Assistant with privacy by design**” guide to benefit our community who wish to own a Opensource Smart Home Assistant device that can protect their privacy yet help with their daily home automation needs.  How secure are commercial Smart Home Assistant devices like Alexa Echo and Google Home?  Much of the data that these smart home assistant devices collect and store your information including personal, potentially identifiable and possibly sensitive information. |
| *“Can other smart home assistant devices violate the privacy and security of our data?”* |
| Real life examples:   1. A couple had to unplug their Personal Assistant device, as their private conversation was recorded and sent to one of their friends on their contact list since all data is stored in the cloud thus their Personal Assistant has to be connected to Internet Service. 2. A man accidentally received access to 1,700 Personal Assistant audio files belonging to a complete stranger. The files revealed the person’s name, habits, jobs and other sensitive information.   Problem Statement   |  | | --- | | To develop an alternative Opensource smart home assistant device by design whereby users have full control over their privacy to help with their daily automation needs. | |

Solution

|  |
| --- |
| 1. To build a device that may or may not be connected to the internet but live up to functionalities of a smart home assistant device within the home network 2. To only connect to the internet with built-in security protocol ring fencing the device should there be a need for any software update 3. Have an option for device owners to access securely over the internet via mobile devices for monitoring and receive notification purpose |

Architecture and Design

|  |  |
| --- | --- |
| *“The platforms are chosen based on the communities behind these platforms, and why it's so important to have a strong foundation when choosing an open-source solution.”* | |
| **Key consideration:**  1. Low-cost device 2. Simple enough for all ages 3. Integrate any communication protocols used in smart home devices 4. Early childhood learning of tech and programming education 5. Very simple guide for DIY and home automation enthusiast 6. Mature open-source products of huge community support 7. Open-source products especially security related are widely available  **Hardware considerations:**  1. Portability and Non-portability 2. Size matter 3. GUI or None GUI 4. Aesthetic 5. Affordability  **Opensource Software considerations:**  1. Simple operating system 2. Personal Assistant software 3. Security Software (i.e. VPN, Firewall, Ad-Blocker & etc.)  **Hardware evaluation and finalization**  1. Desktop/Laptop computer 2. Arduino 3. Raspberry Pi 4 B (Min. 4Gb ram) |

## **Opensource Voice Assistant software evaluation and finalization:**

1. Mycroft
2. Rhasspy
3. Almond

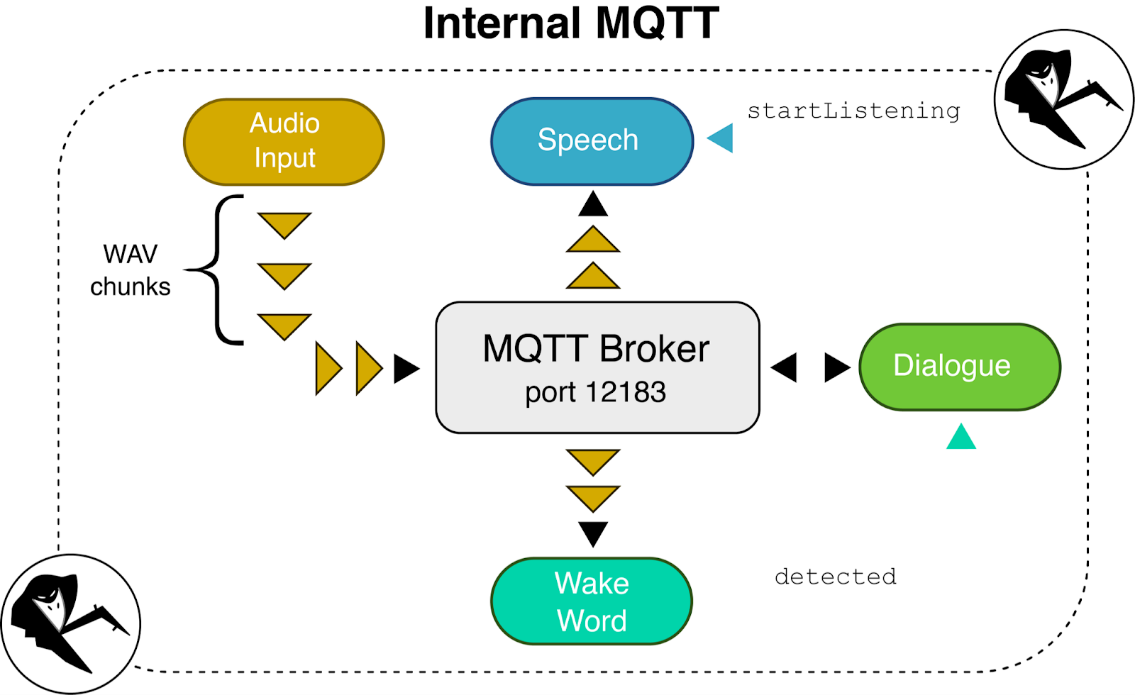
## **Home Automation software evaluation and finalization:**

1. **Home Assistant**
2. openHAB
3. OpenMotics

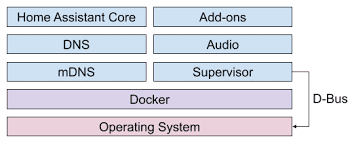
## **Opensource Security software evaluation and finalization:**

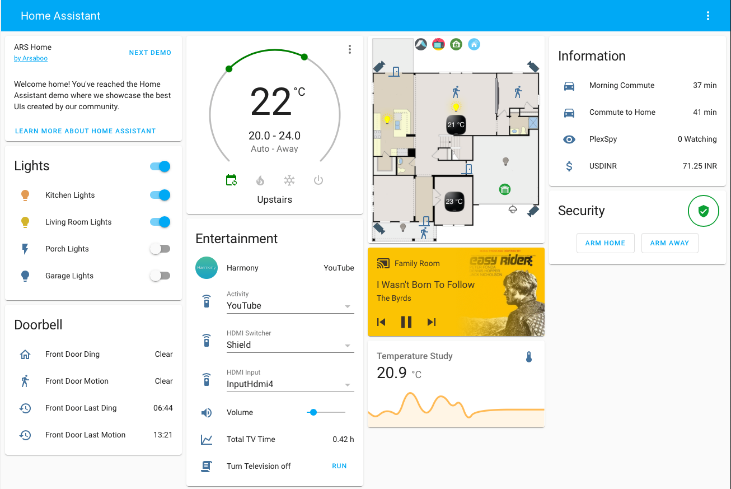
1. WireGuard VPN
2. Ufw Firewall
3. Pi-Hole Ad-blocker
4. Duckdns HTTPs enabler
5. Let’s Encrypt
6. Nginx reverse proxy

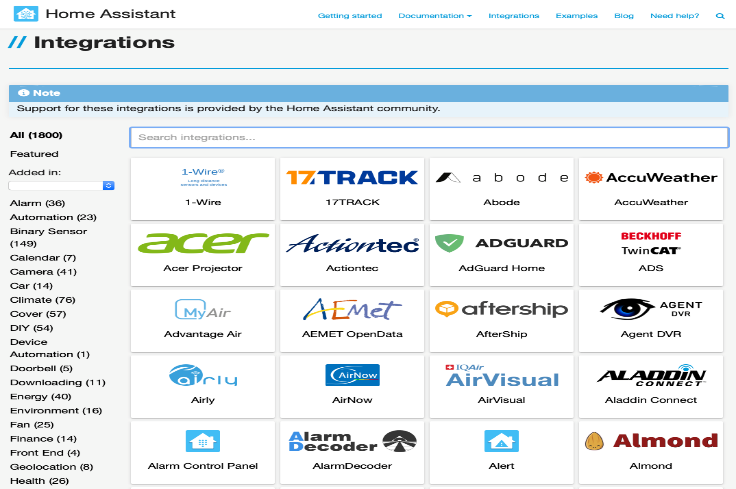
Rhasspy architecture



Home Assistant architecture







Initial basic setup cost

|  |
| --- |
| Total Hardware cost = US$137.75 +/-:   1. Raspberry Pi 4 B 4Gb Starter pro kit – US$99.99 (Bundle 32Gb micro-SD) 2. Mini USB 2.0 microphone – US$7.98 3. Anker Soundcore Mini with 3.5mm jack output – US$23.99   *(Note: Reuse any old smart speaker with 3.5mm jack output laying at home)*   1. Cat 6 0.5m Ethernet cable – US$5.79 |

Rhasspy opensource Voice Assistant software cost = Zero

Home Assistant opensource Automation software cost = Zero

Opensource Security software cost = Zero

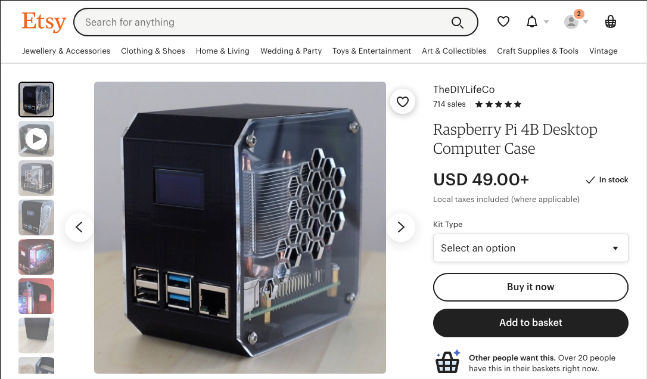
1. DuckDNS HTTPs enabler
2. Let’s Encrypt
3. Nginx reverse proxy
4. WireGuard VPN

Total initial basic setup cost = US$137.75 +/- plus free time spent to build this smart home assistant device.

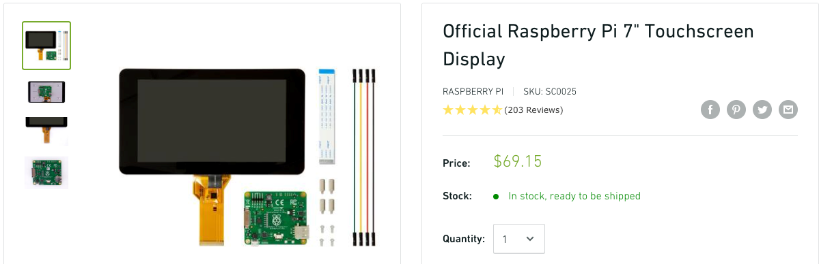
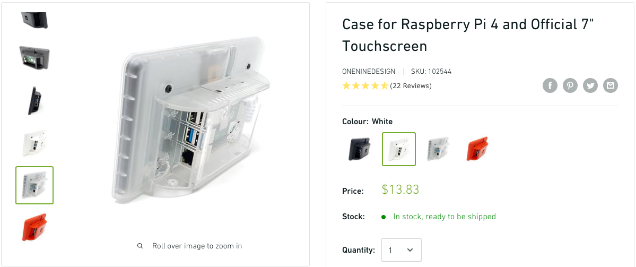
For better looking Home Automation console, you may consider below optional hardware add-ons:

1. 3D printed casing
2. GeekPi Raspberry Pi Cooling Fan ICE Tower cooler with RGB
3. Raspberry Pi 7” Touchscreen panel display with cooling fan
4. Case for Raspberry Pi 7” Touchscreen – Case for Official Raspberry Pi 7” Touchscreen display with cooling fan

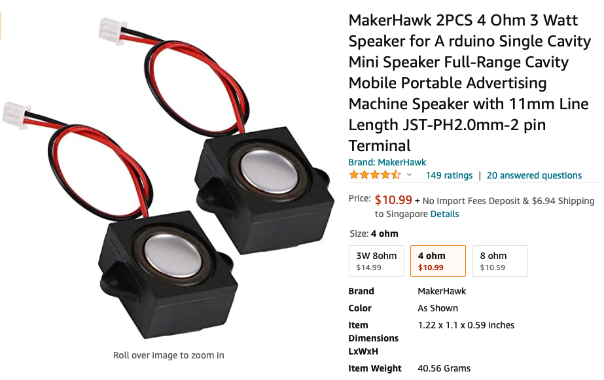
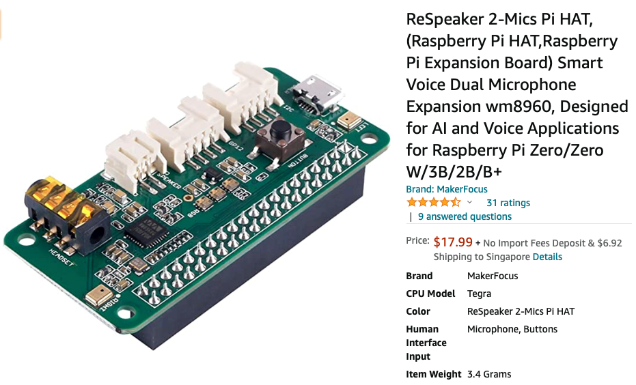
Add-On #1



Add-On #2



Add-On #3



Add-on #1 – Total cost for a sleek casing with a RGB cooling fan and ICE Tower cooler estimated at US$67.99

Add-on #2 – Total cost for an interactive touch screen of your Home Assistant dashboard estimated at US$82.98

Add-on #3 – Total cost for a Respeaker 2-mics Pi Hat with RGB and built-in speakers to conceal your mic and speaker estimated at US$28.92

What’s next?

## **Building Blocks as per below simple steps:**

1. Hardware assembly
2. Flash Operating System (OS) of your choice on to micro-SD card
   1. Raspbian OS
   2. Home Assistant OS
3. Insert micro-SD card into Raspberry Pi
4. For Home Assistant OS, follow below step and proceed to step 6:
   1. Hardening of Home Assistant OS, start-up and configure Home Assistant thru HTTP://localhost:8123
5. Home Assistant will now scan, discover and notify you of all the available smart devices connected within the same private network
6. Either click on the notification to jump to Integration tab or manually go to Configuration>Integration tab to configure any devices you need to integrate and automate via Home Assistant
7. Next step is to discover Rhasspy + OpenTTS from the Add-On > Repository tab. Cut and paste the Rhasspy GitHub link to download 2 Add-On to Home Assistant
8. Configure both Rhassy and OpenTTS as per Home Assistant and Rhasspy guide

## **Hardware preparation**

We will need to prepare a few things to get started with installing Home Assistant.

1. Raspberry Pi 4
2. Power Supply for Raspberry Pi 4
3. Micro SD Card. Ideally get one that is Application Class 2 as they handle small I/O much more consistently than cards not optimized to host applications. A 32 GB or bigger card is recommended.
4. SD Card reader. This is already part of most laptops, but you can purchase a standalone USB adapter if you don’t have one. The brand doesn’t matter, just pick the cheapest.
5. Ethernet cable. Home Assistant can work with Wi-Fi, but an Ethernet connection is more reliable and highly recommended.

Getting Started

## **Overview of overall process**

* + 1. Install Home Assistant Operating System build on SD card
    2. Start up Raspberry Pi 4 B with Home Assistant on SD card
    3. Setup Home Assistant with 2 Factor Authentication
    4. Install and configure security add-ons
    5. Install, configure and test Mosquitto broker, Rhasspy Voice Assistant
    6. Setup Event commands, train and test Rhasspy on Speech To Text (STT). Wake Rhasspy up using the Wake Work you configured. Issue a voice command to Rhasspy (i.e. turn on the lights). Rhasspy will convert this voice command into text using Speech To Text (STT) function and send text only via external MQTT to Home Assistant
    7. Setup Event automation in Home Assistant with MQTT message response as text message to Rhasspy. Rhasspy to read out the response (i.e. the light has been turned on) after converting it using Text To Speech (TTS) function
    8. Test and re-train Rhasspy

Install, Secure, Configure Home Assistant, Rhasspy and others Add-On

The first step is to install Home Assistant Operating System build. We recommend a dedicated system to run Home Assistant.

Now follow the below steps to download and flash Home Assistant Operating System into your micro-SD card:

## **Write image file to your installation media**

1. Download and install Raspberry Pi Imager for an easy way to install Home Assistant operating systems to an SD card ready to use with your Raspberry Pi:
   1. [Raspberry Pi Imager for Windows](https://downloads.raspberrypi.org/imager/imager_1.4.exe)
   2. [Raspberry Pi Imager for macOS](https://downloads.raspberrypi.org/imager/imager_1.4.dmg)
   3. [Raspberry Pi Imager for Ubuntu](https://downloads.raspberrypi.org/imager/imager_1.4_amd64.deb)
2. Attach the installation media (SD card / SD Card reader) to your computer
3. Double click to launch Raspberry Pi Imager
4. Select “Choose OS” and select



1. Go to Home Assistant Installation page. At “Install Home Assistant Operating System section, scroll down to below section:



1. Copy below GitHub URL of Raspberry Pi 4 64-bit link below into your web browser. Hit enter to download the imager onto your computer “Download” folder:
2. Home Assistant Operating System for Raspberry Pi 4 64-bit link: [*https://github.com/home-assistant/operating-system/releases/download/6.1/haos\_rpi4-64-6.1.img.xz*](https://github.com/home-assistant/operating-system/releases/download/6.1/haos_rpi4-64-6.1.img.xz)or the latest link
3. Now launch Raspberry Pi Imager software
4. Click on “Choose OS” button and select “haos\_rpi4-64-6.1.img.xz” from your “Download” folder
5. Click on “Choose Storage” button and select the “Generic STORAGE DEVICE Media” your SD card drive
6. Then click on the “Write” button and wait for it to Write => Verify => Finish. Eject the SD card if it is not automatically done so



## **Start up your Raspberry Pi 4 B and Home Assistant**

1. Insert the installation media (SD card) you just created
2. Attach an ethernet cable for the network.
3. Attach USB mini microphone, Smart Speaker, mini-HDMI cable and last but not least the power cable. *(Note: Remember to connect the other end of your full HDMI cable end to your computer monitor HDMI input.)*
4. Power on the Raspberry and within a few minutes you will be able to reach Home Assistant on homeassistant.local:8123. If you are running an older Windows version or have a stricter network configuration, you might need to access Home Assistant at homeassistant:8123 or http://X.X.X.X:8123 (replace X.X.X.X with your Raspberry Pi’s Static IP address. If DHCP is turned on at the router, ensure you place this Raspberry Pi IP address as reserved)

With the Home Assistant Operating System installed and accessible you can continue with onboarding.

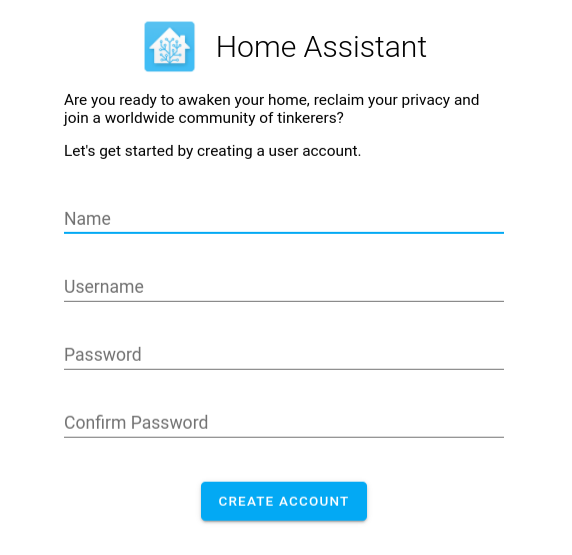
## **Onboarding Home Assistant**

Alright, you made it so far. The tough part is done. Here is where the adventure begins.

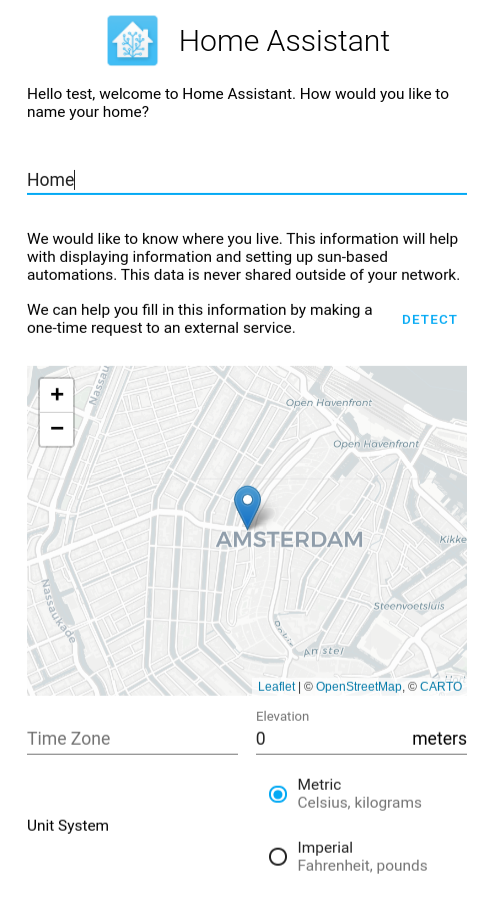
With Home Assistant installed, it’s time to configure it. Once you connect to Home Assistant thru your computer browser, Home Assistant will need about 20 minutes to prepare the environment before you get to see the log-on screen.



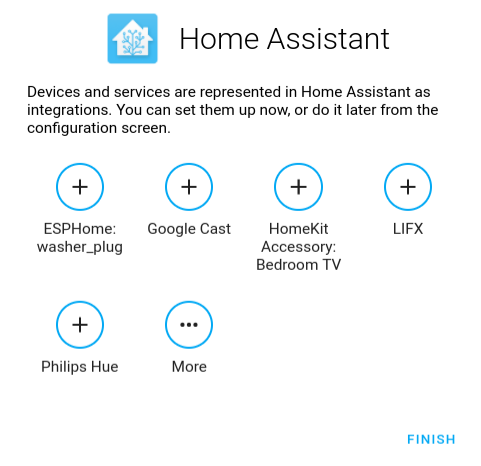
Here you will create the owner account of Home Assistant. This account will be an administrator and will always be able to change all configurations. Enter a name, username, password and click on “create account”.



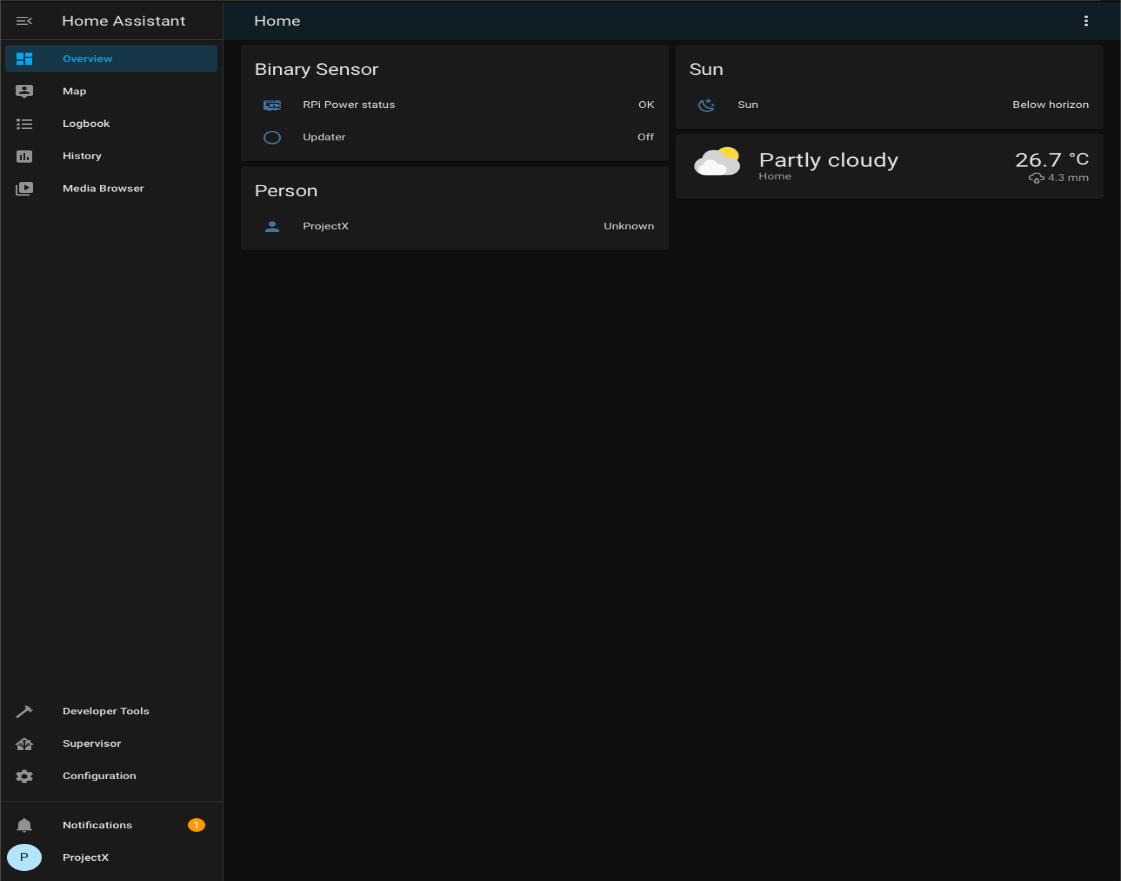
Next, you can enter a name for your home and set your location and unit system. Click “DETECT” to find your location and set your time zone and unit system based on that location. If you’d rather not send your location, you can set these values manually.



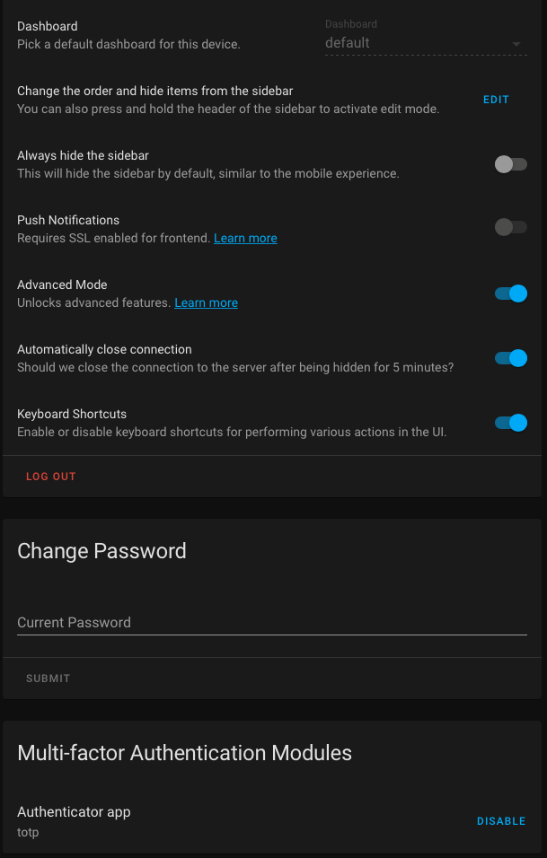
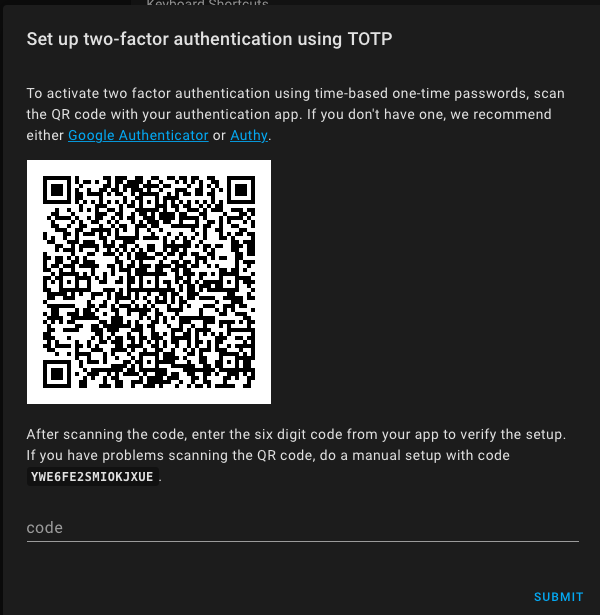
Once you are done, click Next. In this screen, Home Assistant will show any devices that it has discovered on your network. Don’t be alarmed if you see fewer items than what is shown below; you can always manually add devices later.



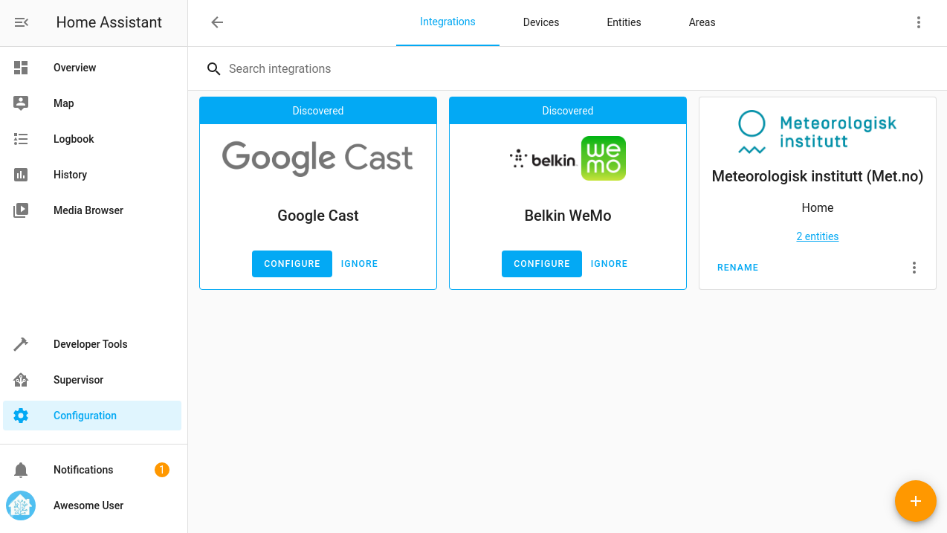
Finally, click Finish. Now you’re brought to the Home Assistant web interface “Overview” which is an underlying dashboard tool called “Lovelace”. This screen will show all of your devices. So, let’s get that screen filled up!



Now go to your profile to enable both “Advanced Mode” and “Multi-factor Authentication Modules” using any recommended Authenticator apps for your mobile phone i.e. “Google Authenticator” or “Authy”. Install your choice of Authenticator apps and scan the QR code and enter the passcode into Home Assistant.

Click on Configuration in the sidebar on the left. On the next screen, click on Integrations. At this screen you will be able to set up integrations with Home Assistant. You might notice a “discovered” section. This section contains integrations that were found on your network and can easily be added with a few clicks. If your integrations are not discovered, click the + button in the lower right and search for your integration in that list.

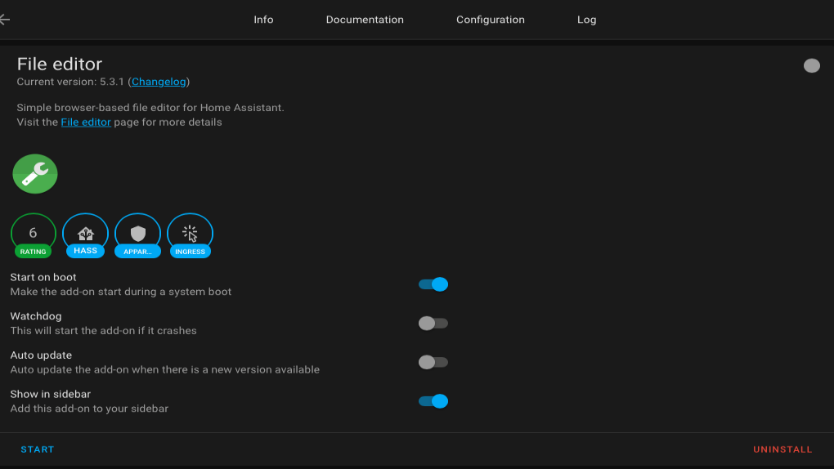


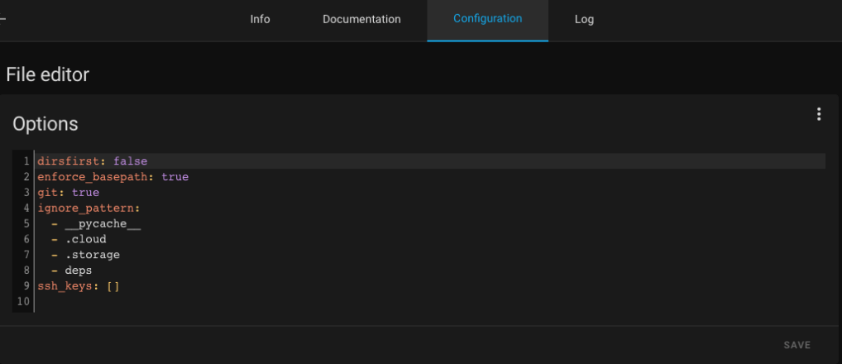
When each integration is done setting up, it will ask you to put the new devices in areas. Areas allow you to organize all the devices in your home.

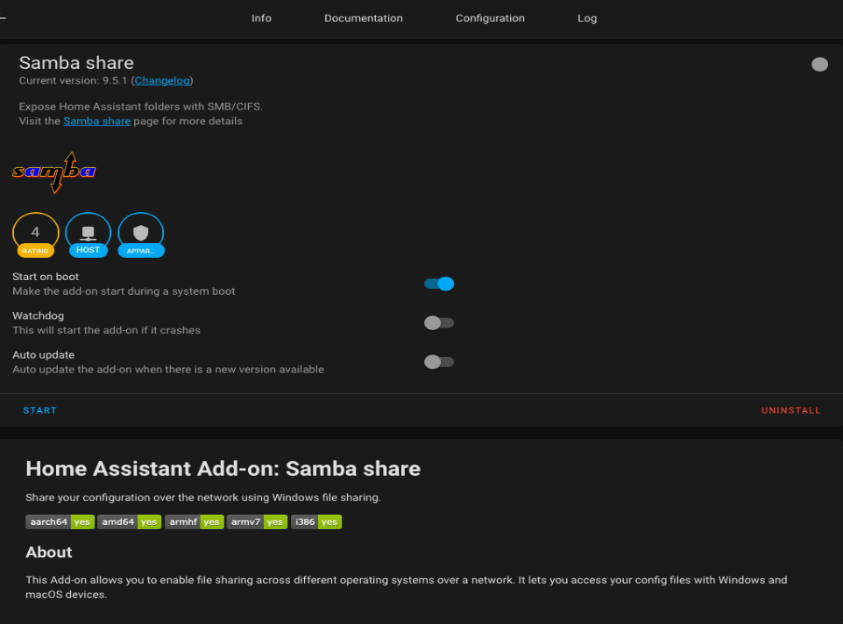
When you’re done, navigate back to the web interface and voila, your devices are ready for you to control. Next is to install a few key Add-Ons that you will use at all times.

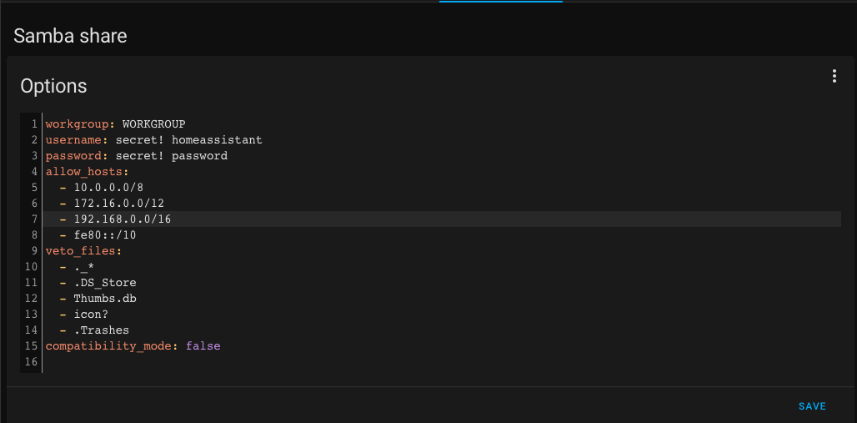
## **Key Add-On needed**

1. File Editor – Terminal emulator to allow you to enable all Home Assistant configurations files with extension “.yaml”.
2. Samba Windows – Allow you to media sharing between your computer and your Raspberry Pi 4 B Home Assistant and Add-On directories to edit any files on your Raspberry Pi 4 B. Do take note that it requires your Home Assistant log-on ID and Password. To hide the ID and password, Home Assistant has a “Secret.yaml” configuration file to keep all ID and password. By using the command “secret! *tag of username* and secret! *tag of Password”* on your Add-On configuration setting. Home Assistant will do a look-up at “Secret.yaml” for it. This helps keep your credential safe in case you need to share screenshots with others.









Next is to secure your Home Assistant operating system.

## **Securing Home Assistant**

A few Add-Ons and configurations required in order to secure your Raspberry Pi 4 B and Home Assistant with:

1. DuckDNS – Setup a secured HTTPS URL for external connections with external to internal network port forwarding. It also provides dynamic DNS service that allows you to point a subdomain under duckdns.org at your computer.
2. Let’sEncrypt – To manage SSL certificate of DuckDNS SSL certificate
3. Nginx – Setup a reverse proxy server serving as a traffic manager to direct packets from clients and direct them to Home Assistant
4. Home Assistant security
   1. Enable 2 Factor Authentication
   2. Setup IP Ban – To permanently ban any IP address attempting to login X number of times based on setting
   3. Secret – To protect your ID and Password

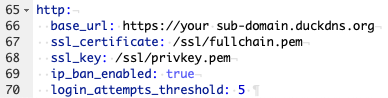
### Setup Home Assistant security

1. During initial setup of Home Assistant, you have already enabled Multi-factor Authentication
2. Setup IP Ban
   1. Use File Editor to open your configuration.yaml file. Copy paste this command line into your HTTP section below your

ip\_ban\_enabled: true

login\_attempts\_threshold: X

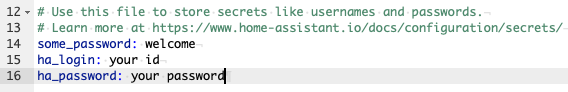
*where X is the number of attempted logins from 1 specific IP address. Once the threshold has exceeded, IP Ban will permanently ban this IP address*



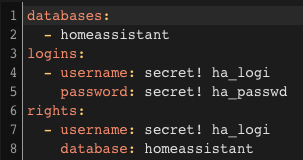
* 1. Now this setup is completed

1. Create a Secret.yaml file to store your login credential in configuration.yaml using command to make reference back to secret.yaml.

*“Do take note that secret.yaml will not be encrypted. The objective of secret.yaml is to protect and prevent exposure of your login credential in configuration.yaml when you are sharing the contents or screenshots with someone.”*



Then process to define your defined login and password tag from your secret.yaml to all your \*.yaml files or Add-On configuration that requires your Home Assistant or other ID and Password as per below command line instead of clear text of ID and Password.



### Secured connection to Home Assistant using DuckDNS

Setting up Duck DNS allows you to access your Home Assistant from outside of your home securely.

Without this service, you are limited to only accessing it with HTTP by going to http://homeassistant.local:8123

This is okay if you really only want local control, but most of us want to be able to control our smart homes from outside of the house as well, and that’s where Duck DNS comes in.

There are a couple of prerequisites necessary to getting this up and running. Firstly, you need to have Home Assistant up and running.

Secondly you need to be able to access your router / modem’s settings, and that’s pretty much it.

Now this portion will be slightly different for everyone, depending on the model of your router. I personally use a Linksys router, so my settings will look a little different than yours.

We are essentially going to do two things:

1. Set a static IP address for the Home Assistant. If your router is configured to auto assign IP address thru DHCP, you will need to reserve the IP address that your Raspberry Pi 4 B was assigned with.
2. Set port forwarding for the Home Assistant
3. If you are using a different brand router, you’ll have to consult Google to help you along the way. These are both very basic functions, so it shouldn’t be too difficult.

### Setting Raspberry Pi 4 B Static IP running Home Assistant

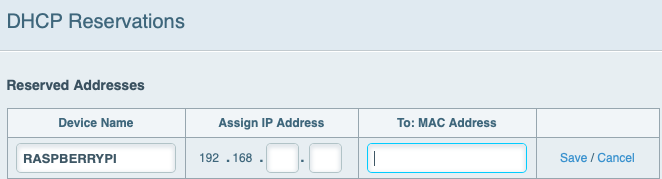
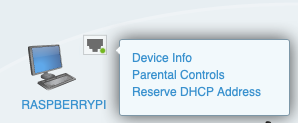
In Linksys settings I can set a static IP address by going to Advanced – Setup – LAN (or WAN if your Home Assistant is connected via Wi-Fi) – Address Reservation.

Linksys Static IP assignment

You need to check your router setting whether DHCP turns on/off. It is turned off, then your Raspberry Pi would have acquired a Static IP Address from your router. This means that will be the Static IP address of the Home Assistant on your network thus nothing needs to be done.

If your router is on DHCP, then you need to login your router and look for your Raspberry Pi 4 B icon and do a right click. You will see the option to reserve an IP address for your Raspberry Pi 4 B.

Hit Save / Apply once you have done this. Your router may have to restart for this to take effect.



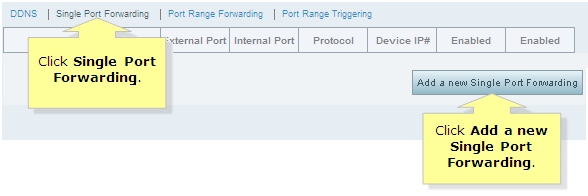
Now we’re going to set up the Port Forwarding aspect. This should also be somewhere in your advanced settings.

### DuckDNS Port Forwarding

For Linksys router it is under Security – Apps and Gaming – Single Port Forwarding.



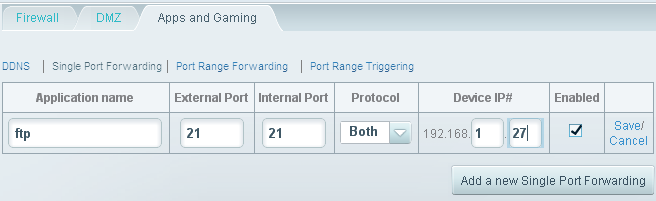


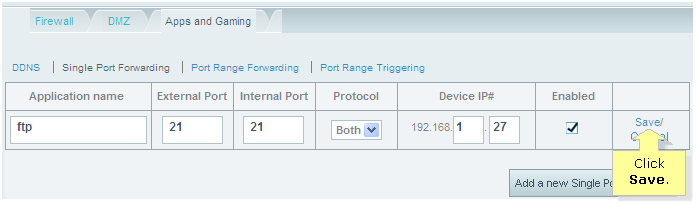


You’re going to set up three instances of port forwarding; one for port 443, one for port 80, and one for port 8123.

Select “Add a new Single Port Forwarding” under Single Port Forwarding. Type in Home Assistant in Application Name, External Port as 443 and Internal Port as 8123, Protocol is Both (TCP/UDP) followed by your Home Assistant IP address (192.168.XX.XXX). Make sure you check the Enable check box.

Save your first rule and set up 2 more rules which are 80 => 8123 and 8123 => 8123. Once all 3 has been set up, click Apply to complete your port forwarding setup on your router.

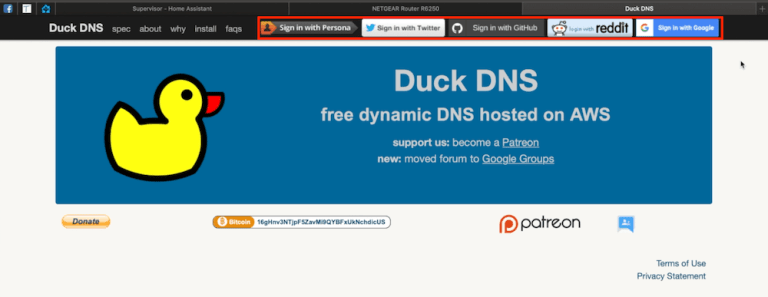




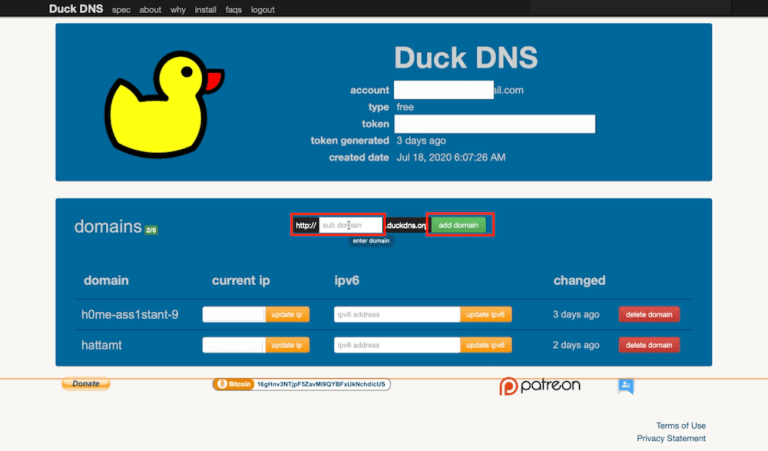
You may need to restart your router for this to go into effect, depending on your router. This can usually be done from the home screen of the router settings. Next is to setup your sub-domain in DuckDNS and get the token into Home Assistant configuration.yaml.

### Setting Up DuckDNS Domain

Next step is to get our DuckDNS account up and running by going to DuckDNS.org. Thankfully you don’t actually need to set up a new account, but you can actually just log in with a variety of social accounts (including Gmail) that you most likely already have.



After you’ve selected your account; go ahead and type in a unique URL that you want as your Home Assistant URL. Try to pick something unique that hasn’t been used before, but is easy enough to type into your URL bar.

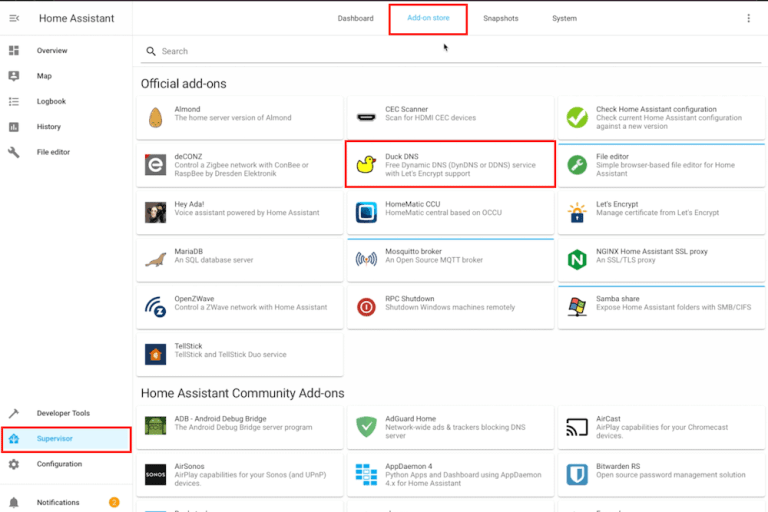


When you’ve decided on one, hit “add domain”.

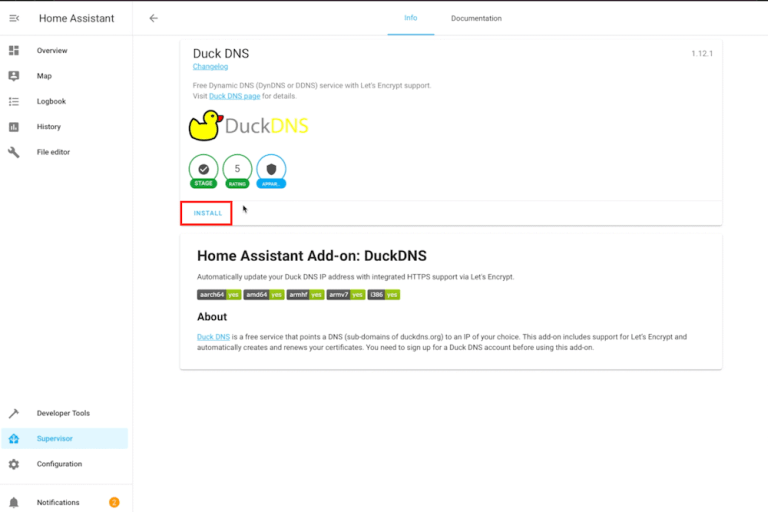
At the top of the screen, you’ll see a “Token“, which is a long alphanumeric string of characters; this is what we’ll need for Duck DNS add-on to work in Home Assistant. Go back to Home Assistant and install DuckDNS.

### Installing DuckDNS Add-on

First thing we’re going to do is login to our Home Assistant and make our way to the Add-on Store.



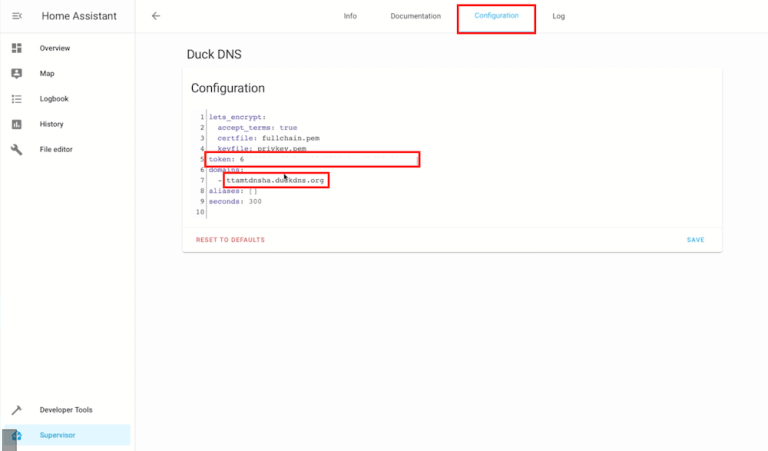
The Duck DNS add-on is towards the top of the screen, select it, and click install. It will take a minute or two, once it is installed, we’re going to configure DuckDNS.



### DuckDNS Home Assistant Configuration

The DuckDNS add-on page should have a “configuration” tab at the top.

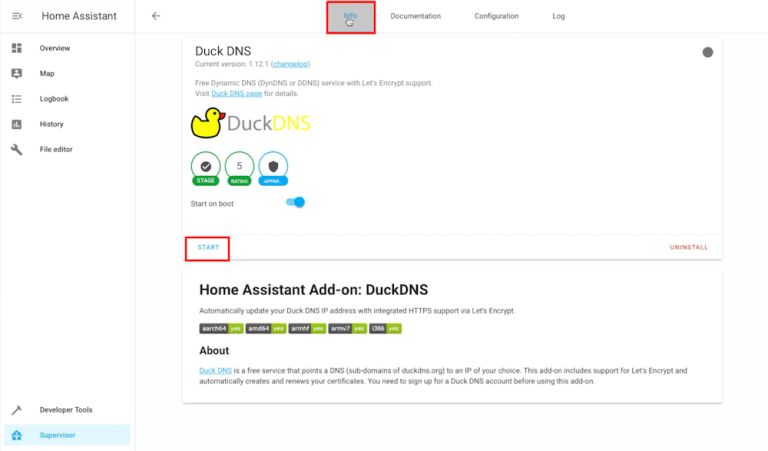
You will fill out three fields in the configurator, the “Accept\_Term”, “Token” and the “Domain”.



Paste your “Token” into the token field, and paste your private domain under where it says “domains”. It should look like the above screenshots provided.

*Also, don’t share your full Tokens, as that is what people would need to access your Home Assistant.*

The last thing we will do in the Configuration tab is to set the Let’s Encrypt “accept\_terms” from “False” to “true”. This is the equivalent of checking the “I agree” box on a form, and allows us to use Let’s Encrypt to securely access your Home Assistant using SSL.



Go ahead and start the add-on, and proceed to the “configuration.yaml” file in your File Editor.

### DuckDNS entry on Configuration.yaml

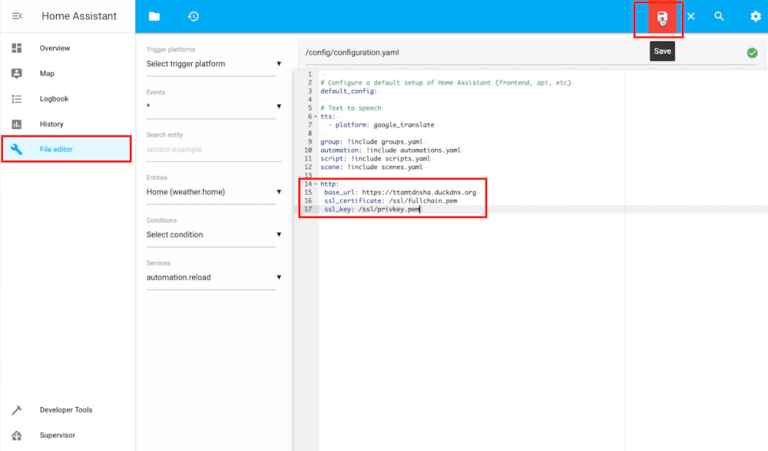
In the configuration.yaml file you will paste the following text:

*http:*

*base URL: https://YOUR DOMAIN HERE.duckdns.org*

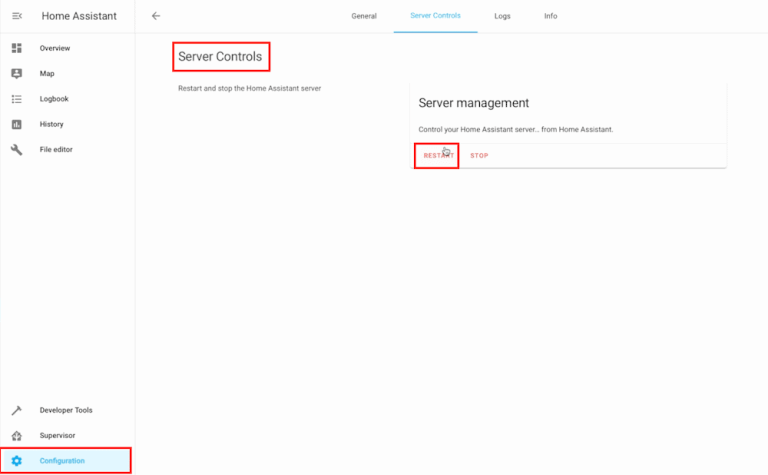
*ssl\_certificate: /ssl/fullchain.pem*

*ssl\_key: /ssl/privkey.pem*



Hit save, and go to Configuration – Server Controls to validate the configuration file and make sure there’s no errors.

Once you get the Configuration Valid response, go ahead and Restart Home Assistant.



Now you proceed to install “Let’s Encrypt” from Add-on Store. No setup on configuration required.

To ensure your port 8123 and 443 has been opened, go to this website called <https://CanYouSeeMe.org> and you will see your public IP. Enter port number 8123 and 443 to ensure that these 2 ports have been opened.

You’ll notice that it won’t be able to reconnect on the same page that you had it open previously. This is because you currently have it open on homeassistant.local:8123, but we need to access it “externally” now.

Give it a minute or two to restart, and then type in your FULL DOMAIN (your “sub-domain”.duckdns.org) into the URL bar, including the https://. It will not connect otherwise.

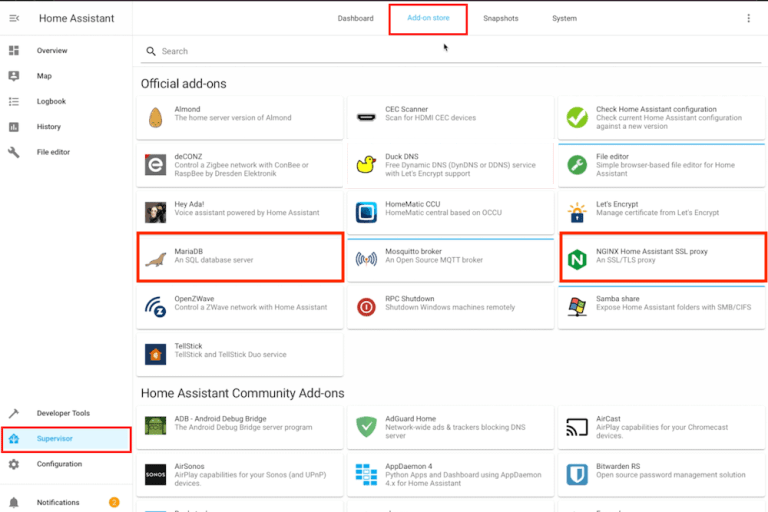
If you did everything like was described you should be greeted with the standard Home Assistant User login screen, where you’ll login with your username and password. If you have 2 Factor Authentication turned on, you will be prompted to input your passcode.

To test out if it works, you can get the iOS or Android app and login using your cellular connection, and you should be able to access your Home Assistant without any issues.

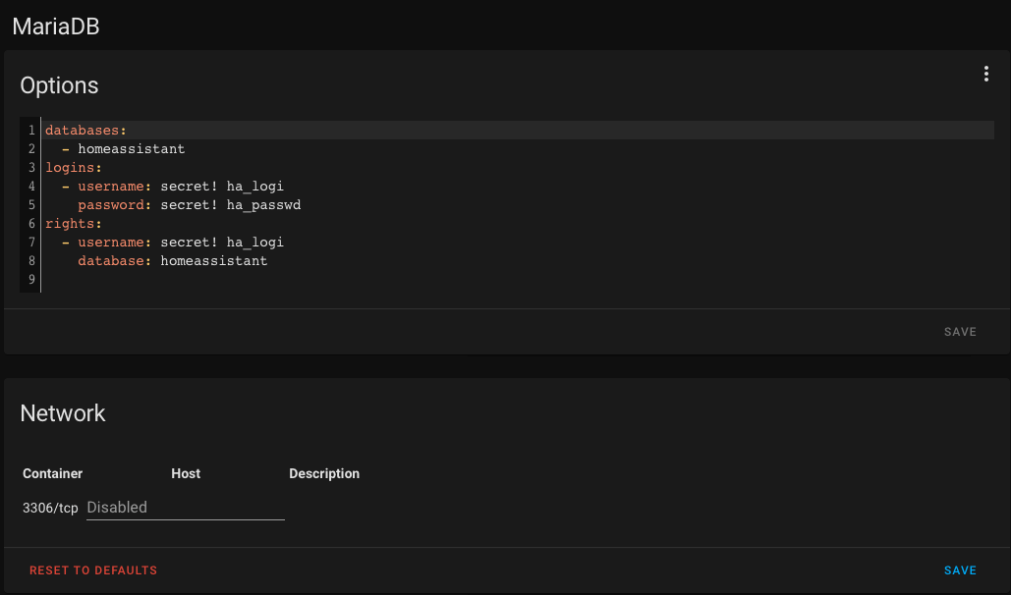
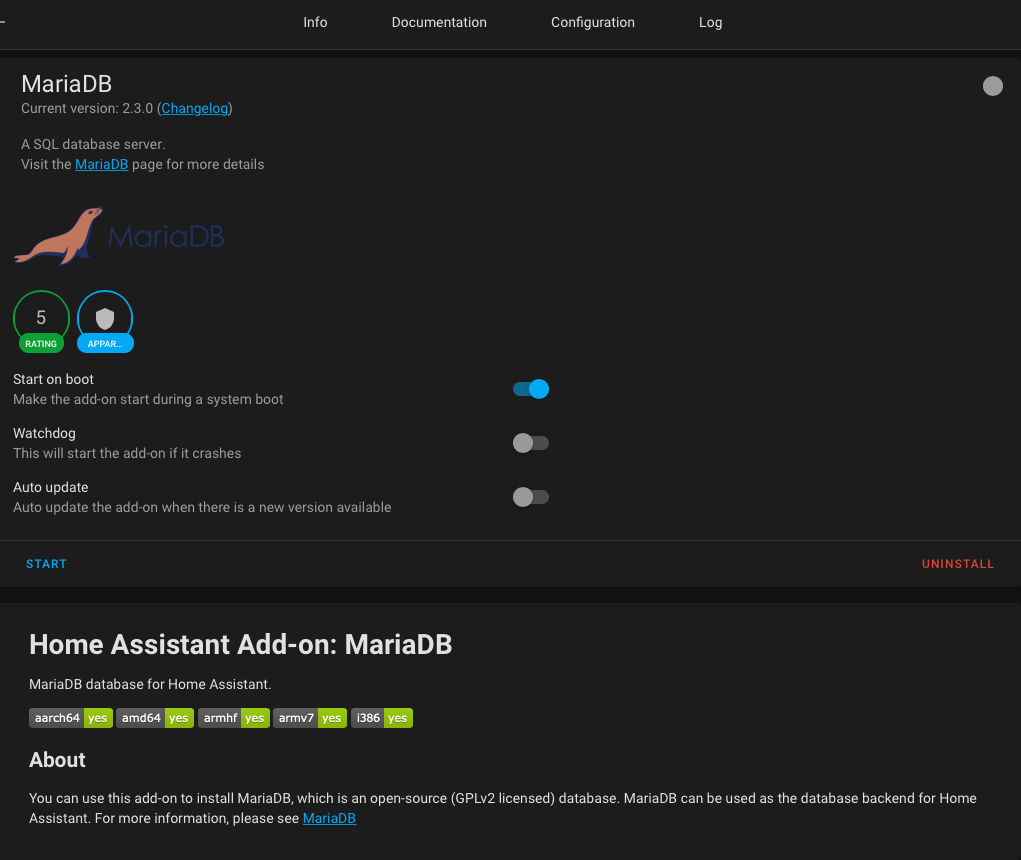
There you have it, Duck DNS set up for free giving you access to your Home Assistant securely without the need to be on your home network. Now you will proceed to the next step to install and set up Nginx, the reverse proxy add-on.

### Setup Maria DB follow by Nginx Proxy Manager addon

Open your Home Assistant, go to Supervisor-> Add-On Store. You want two addons: Nginx Proxy Manager and MariaDB.

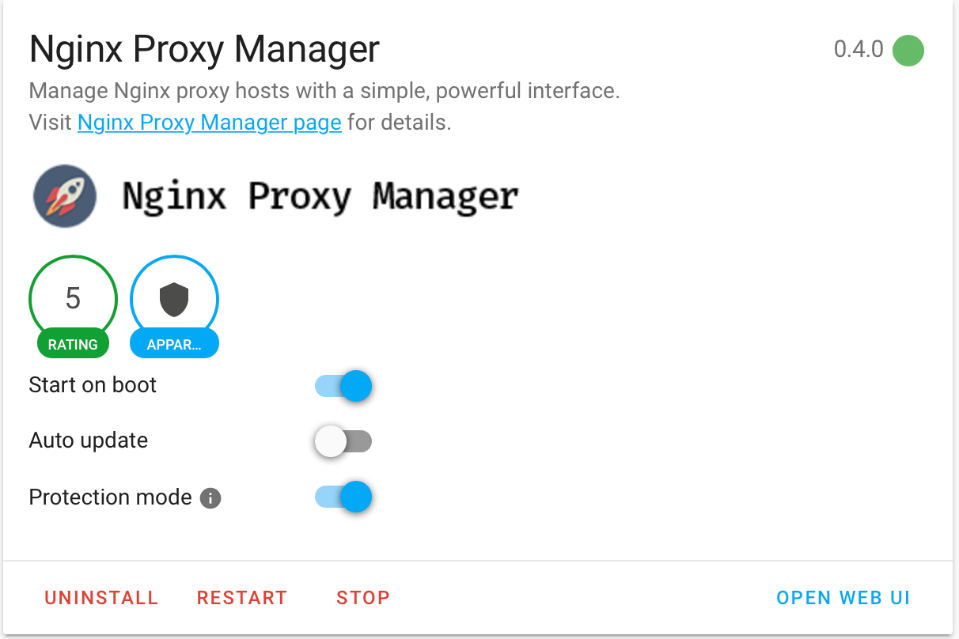


A minimum configuration setting for MariaDB addon is a password for home assistant user, which is not necessary for Nginx Proxy Manager but is essential if you decide to switch from SQLite database to something more productive. This is anyway a good improvement which speeds-up Logbook and History pages in Home Assistant.

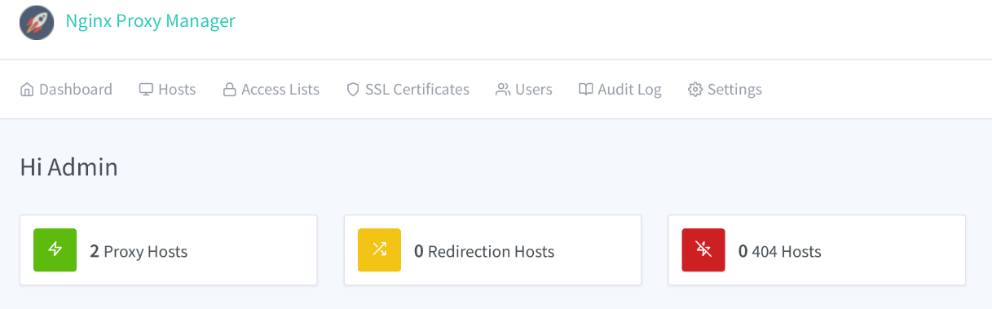


Once the password is set, MariaDB should start without errors.

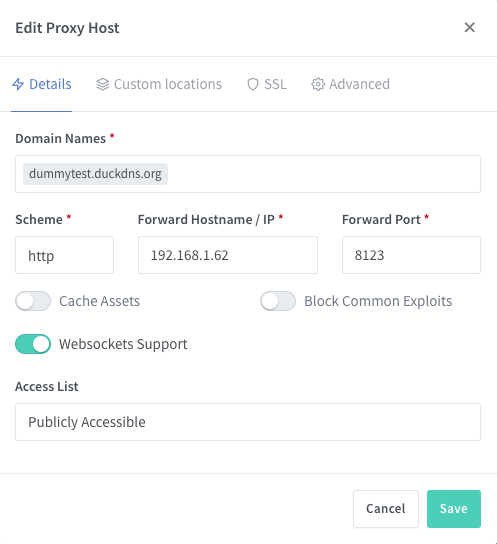
Now we need to install Nginx Proxy Manager. Once started, it will automatically find and connect to MariaDB addon without user intervention. No additional settings are required, we should open its page via Open Web UI link:



Log in using “admin@example.com” as username and password “changeme” and immediately change password.

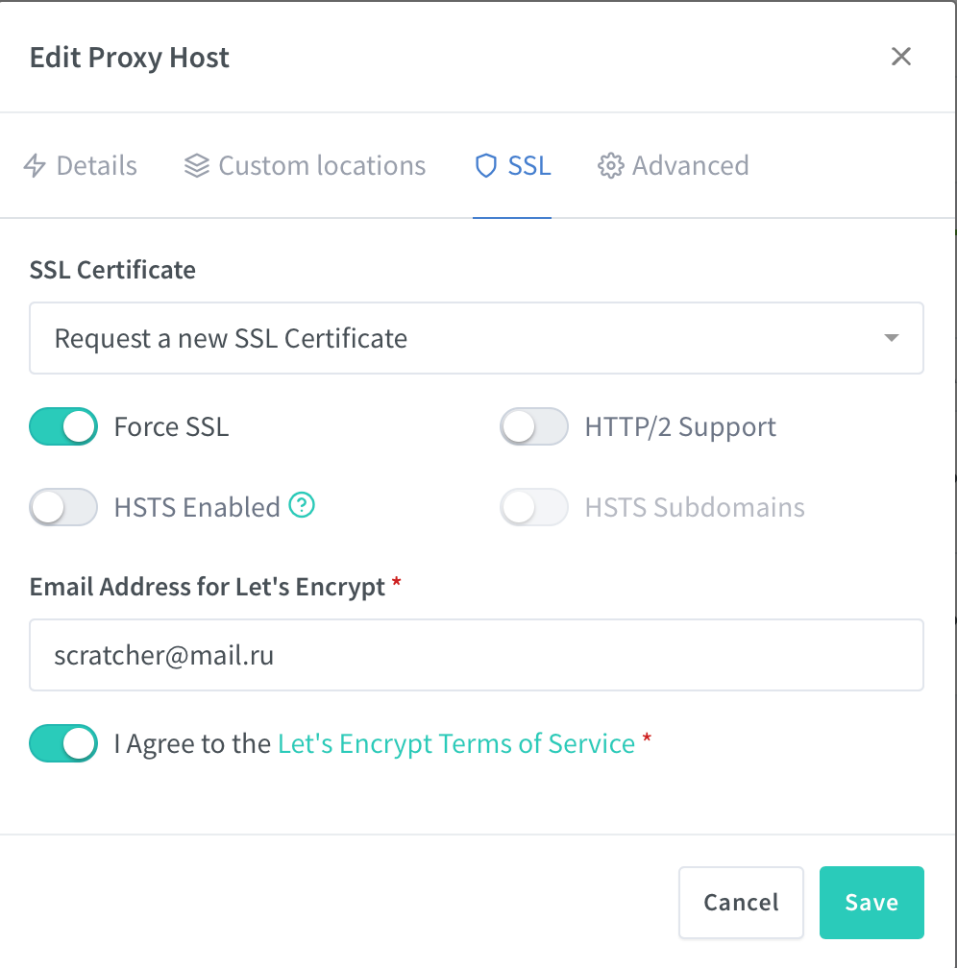


Go to “Proxy Hosts”, add your full domain (your sub-domain.duckdns.org), your Home Assistant IP address and Port number and then enable “Websockets Support” and hit Save:



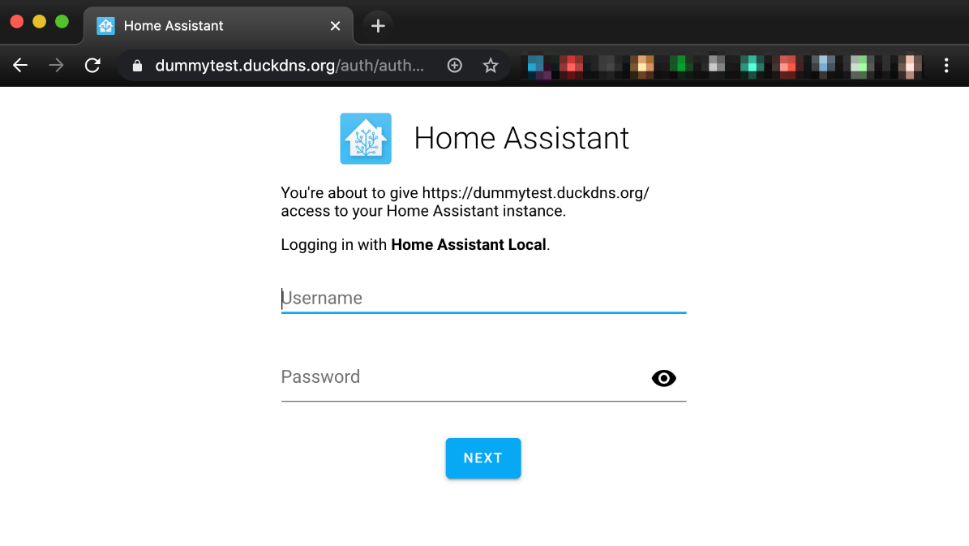
At this point, it is recommended to open up the Home Assistant URL http://dummytest.duckdns.org and ensure that basic HTTP access is working. it will be necessary to obtain a Let’s Encrypt certificate at the next step. If you don’t see your HA prompt, most probably port forwarding was not set up properly. You can also check your router’s firewall settings.

If Home Assistant is accessible (via HTTP), go back to the Nginx Proxy Manager addon page and edit the previously created connection. Go to the SSL tab and select Request a new SSL Certificate, the switches Force SSL and I Agree to… should also be turned on. Save your settings:



That’s it! Remember to restart Home Assistant at Configuration > Server Control > Restart.

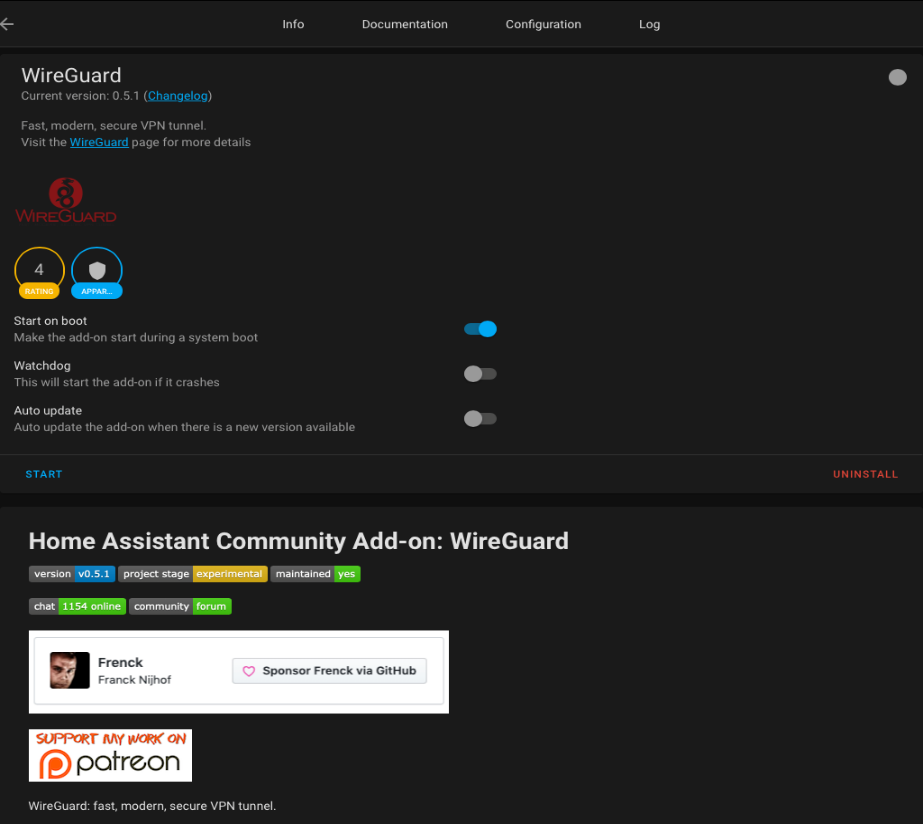
You can now open https://dummytest.duckdns.org in your browser and ensure that your connection is secure:

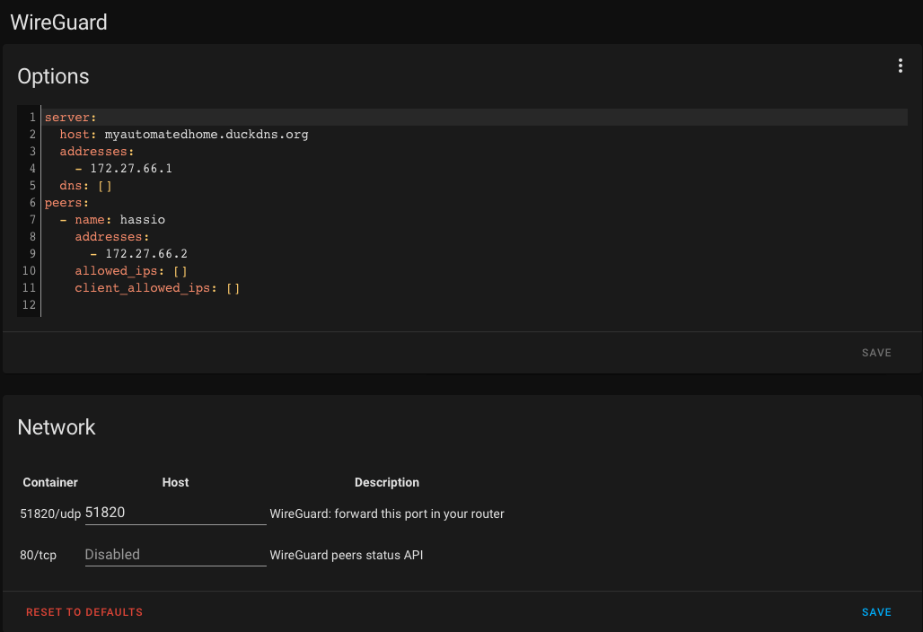


Addon will take care of automatic Let’s Encrypt certificate renewal.

### Install and configure WireGuard VPN

WireGuard is a secure network tunnel, operating at layer 3, implemented as a kernel virtual network interface. It will perform an encrypted network tunnel to further enhance the security of your connections when you remotely access Home Assistant at home via your phone or computer. *Note: Only install WireGuard if you already have subscribed to multi-devices VPN service provided or intent to subscribe.*





## **Install, configure and test MQTT broker and Voice Assistant**

### Install and configure Mosquitto MQTT broker

Home Assistant and Rhasspy voice assistant communicate through an external MQTT broker.

Before installing Mosquitto, do these 2 pre-configurations in Home Assistant

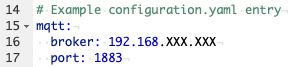
1. Use File Editor to add below into configuration.yaml:

# Example configuration.yaml entry

*mqtt:*

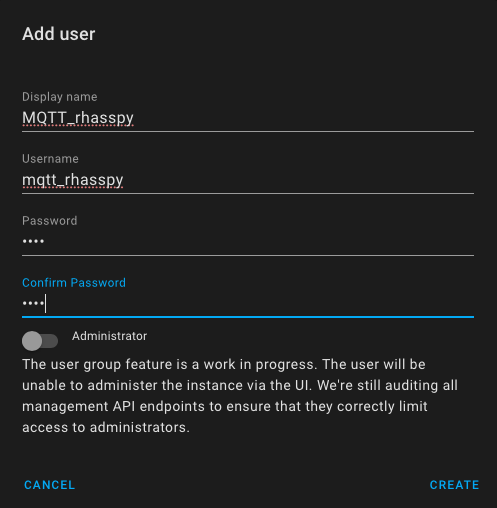
*broker: 192.168.XXX.XXX (Your IP address)*

*port: 1883*

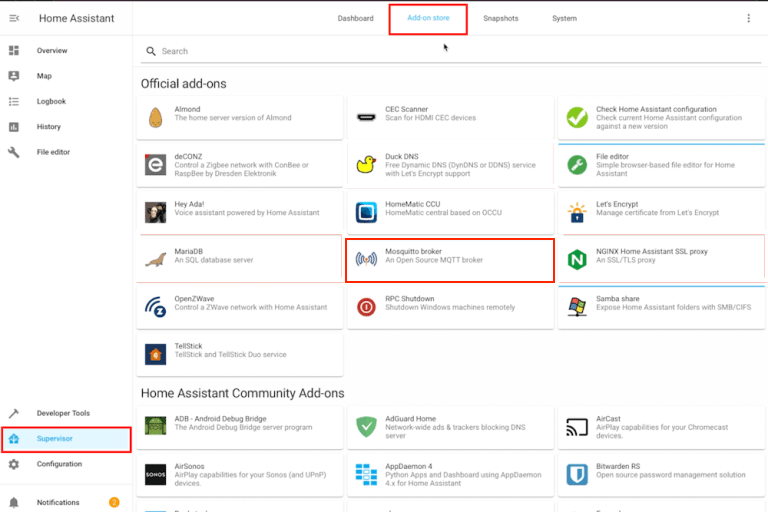


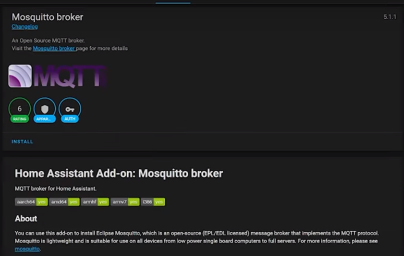
2. Now go to Home Assistant main page => Configuration => Users (Turn on Advance Mode at your profile page if Users function not show up)

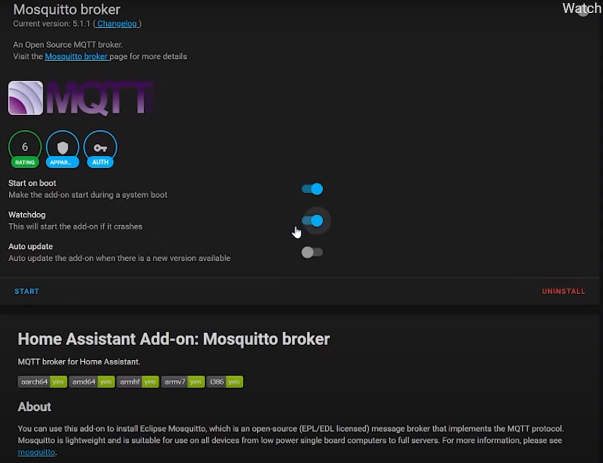
Create a new MQTT user and password. No need to turn on Administrator for this new ID



Proceed to install and configure Mosquitto MQTT broker from Home Assistant Supervisor => Add-On.

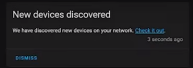






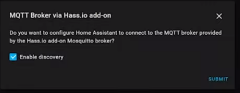
Once installed, turn on Watch Dog and restart Home Assistant => Configuration => Server Control => Restart.

Once Home Assistant restarted, Home Assistant will auto discover Mosquitto on Notification. Go to Notification and click on the Discovery link to redirect you to the Discovery page.



Click on Configure Mosquitto broker => Submit => Finish.

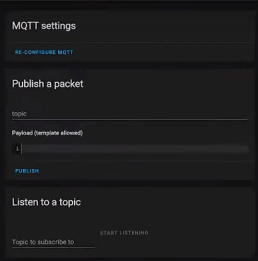


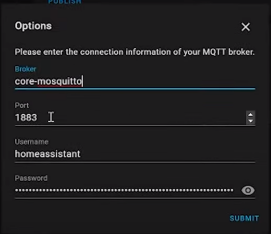




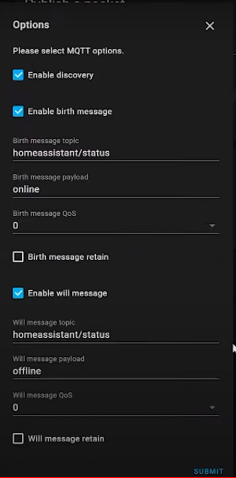
And then click Re-Configure MQTT and enter user name and password then submit.

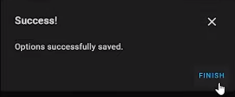
* + 1. Username – Enter the new MQTT username you just created
    2. Password – Enter the password of new MQTT created





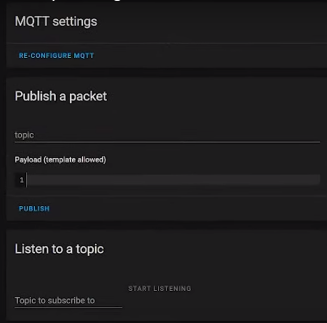
Leave the next screen as default and click Submit followed by Finish on the last screen.

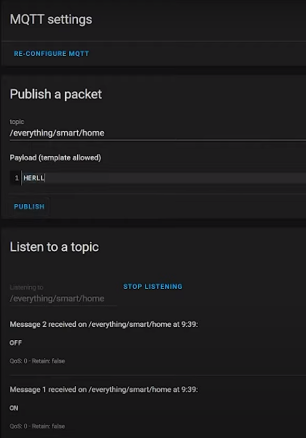


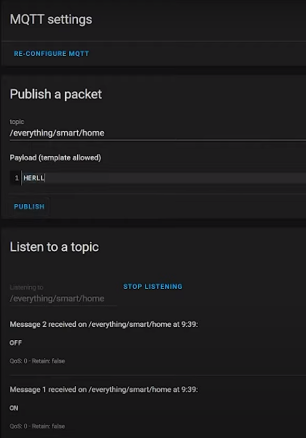


Now Mosquitto broker setup has been completed and ready to do a simple testing to ensure it is working.

Input a topic (i.e. /homeassistant/automation) into Publish a packet and “ON” or “OFF” on Payload. Type in the same topic into Listen to a topic and click on Start Listening. Once done, click on Publish to ensure your Payload value is displayed at the bottom of this screen.



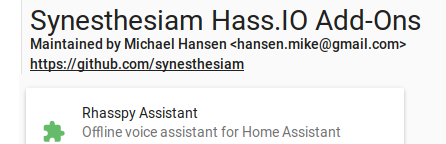




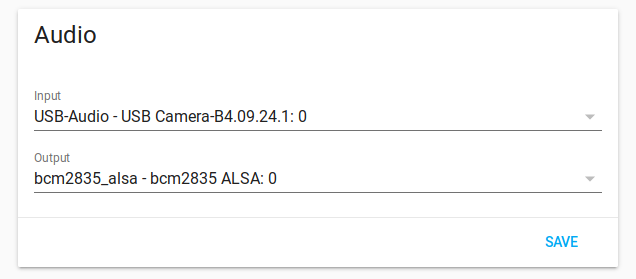
At this point you have completed your Home Assistant securely and ready for Voice Assistant integration and testing. It is time to move on to next step of installing and configure Rhasspy your Voice Assistant.

### Install and configure Rhasspy the Voice Assistant

To install the add-on, add this GitHub hassio-addons [link](https://github.com/synesthesiam/hassio-addons) on to your Home Assistant Add-On Repository (3 vertical dot on the top right corner) in the Add-On Store, refresh, then install the "Rhasspy Assistant 2.5" under “Synesthesiam Hass.IO Add-Ons” (all the way at the bottom of the Add-On Store screen).



Before starting the add-on, make sure to give it access to your microphone and speakers:



Updating Home Assistant OS Add-On

You should receive notifications when a new version of Rhasspy is available for Home Assistant OS. Follow the instructions from Home Assistant OS on how to update the add-on.

Once Rhasspy 2.5 service starts in Supervisor Dashboard of Home Assistant, launch the Rhasspy 2.5 Web UI by clicking “Open Web UI” and follow the below Rhasspy configuration steps.

### Quick tutorial of Rhasspy Web Interface

A browser-based interface for Rhasspy is available on port 12101 by default (http://localhost:12101 if running locally). From this interface, you can test voice commands, add new voice commands, re-train, and edit your profile.

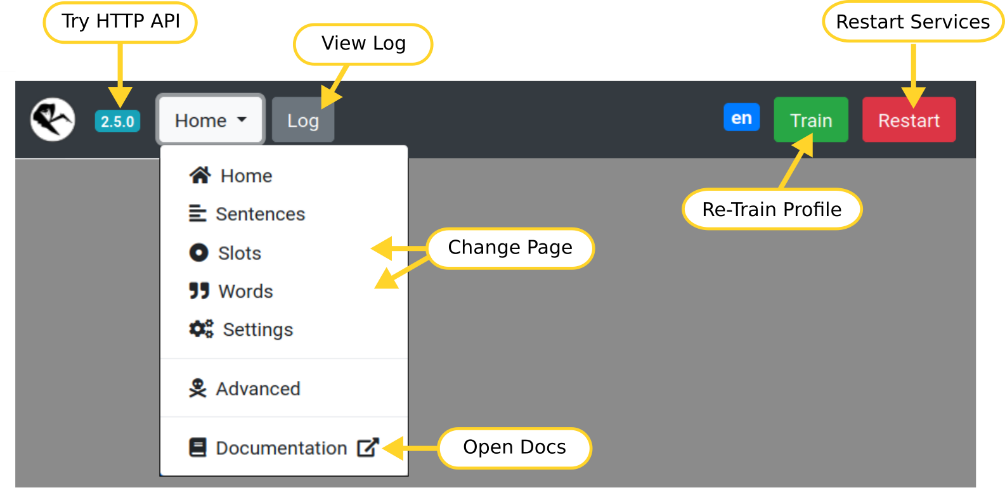
By default, Rhasspy does not start any services (highlighted in green on this bar). From left to right, these services are:

|  | Service | Description |
| --- | --- | --- |
| MQTT icon | MQTT | Indicates if Rhasspy is connected to an internal or external MQTT broker (green) |
| Microphone icon | Audio Input | Records audio from a microphone |
| Handle icon | Intent Handle | Sends recognized intents to other software |
| Wake icon | Wake Word | Listens to live audio and detects a hot/wake word |
| Speech recognition icon | Speech Recognition | Converts voice commands into text |
| Intent recognition icon | Intent Recognition | Recognizes intents and slots from text |
| Text to speech icon | Text to Speech | Speaks text through audio output system |
| Speaker icon | Audio Output | Plays audio through a speaker |
| Dialogue icon | Dialogue Management | Coordinates wake/speech/intent systems and external skills |

Clicking on an individual icon on the service bar will take you to its settings. You can also choose "Settings" from the dropdown menu at the top of the web page.

Top Bar

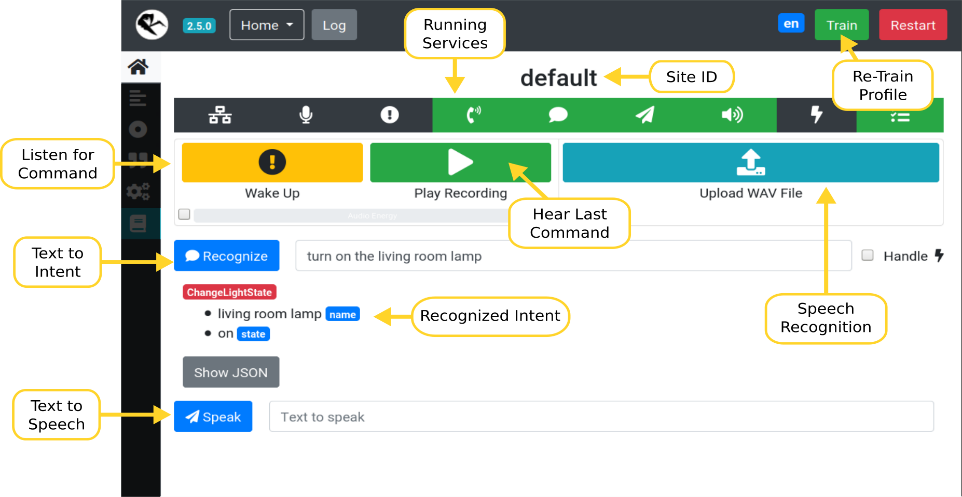
The top bar of the web interface lets you perform some global actions on Rhasspy, regardless of which page you have selected.



1. Click the Rhasspy logo to reload the page
2. Click the version number to test the HTTP API
3. Use the dropdown menu to change pages or open the documentation
4. The gray “Log” button opens a dialog with log messages
5. The green “Train” button will re-train your profile
6. The red “Restart” button forces Rhasspy to restart

Home Page

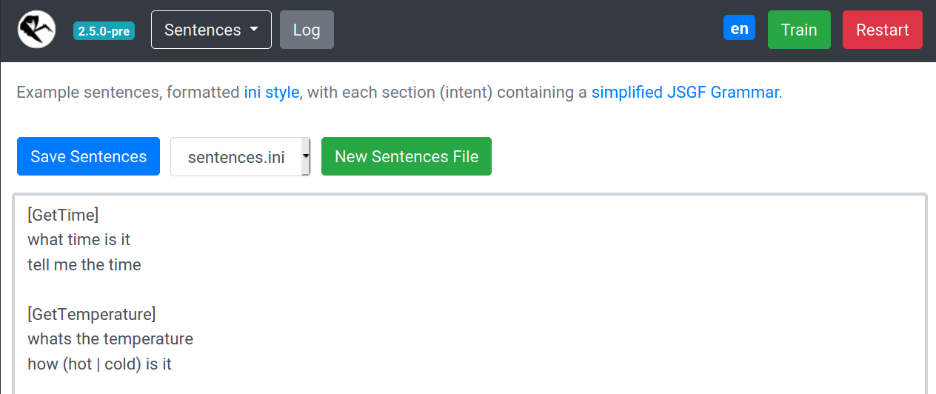
Test voice and text commands.



1. Wake up Rhasspy and have it listened for a voice command
2. Upload a WAV file with a voice command
3. Enter a text command and Rhasspy recognize the intent
4. Speak a sentence using the text to speech system

Sentences Page

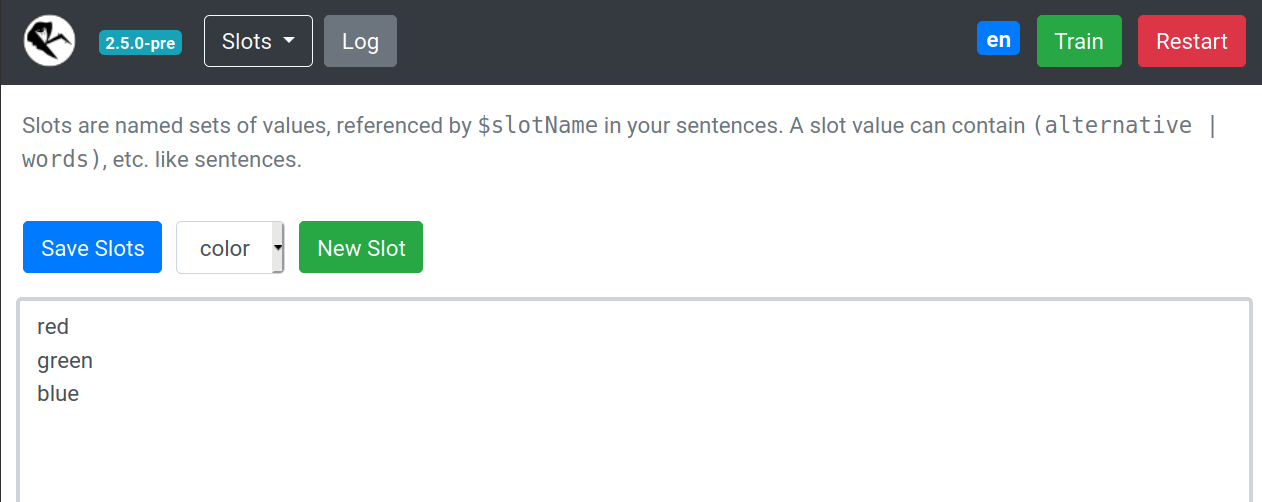
Add new voice commands to Rhasspy using the template syntax.



1. Edits “sentences.ini” by default
2. Create additional template files
3. These should be prefixed by the “sentences\_dir” in your profile. For example, “intents/more-commands.ini”
4. The drop down can be used to switch editing between different template files

Slots Page

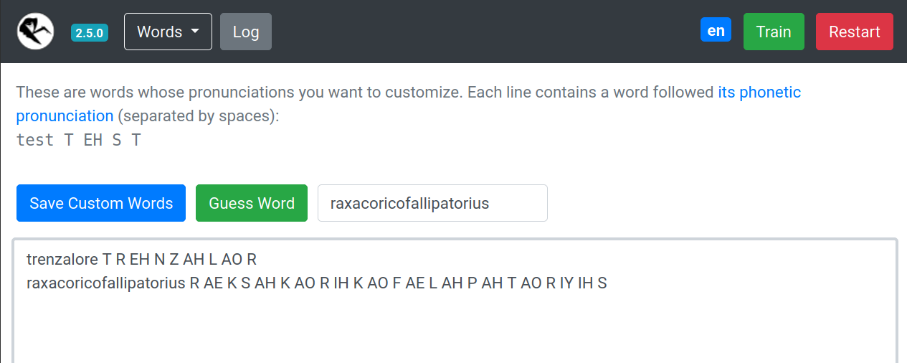
Edit your slots lists.



1. Slot values will overwrite previous ones
2. Create new slots (files in your “slots” directory)
3. Delete a slot by deleting all values and saving

Words Page

Teach Rhasspy how to pronounce new words.



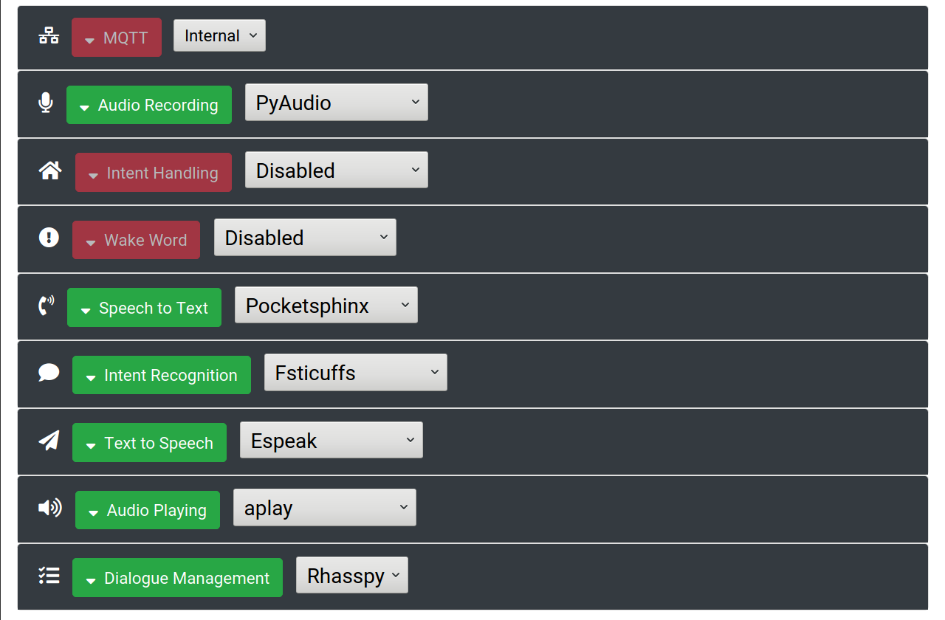
Look up pronunciation(s) for known words (in your profile's “base\_dictionary.txt” file)

Have Rhasspy guess how to pronounce a new (unknown) word

Add new words to your “custom\_words.txt” file

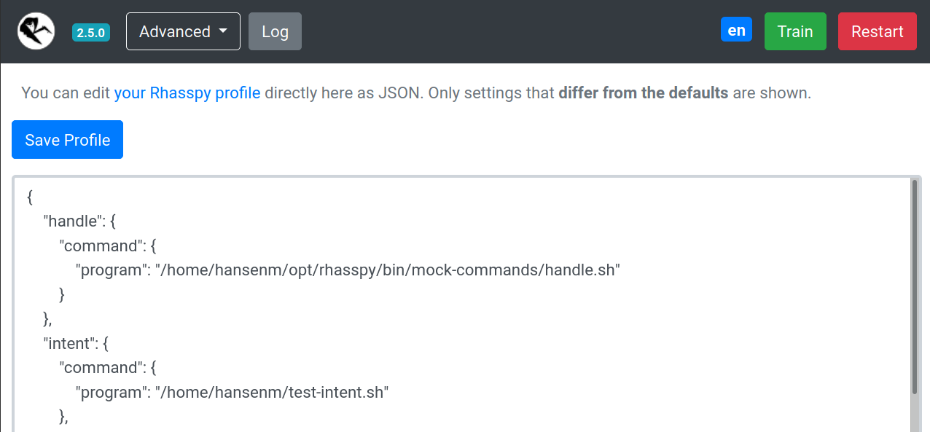
Settings Page

Simplified interface for editing your profile. Rhasspy will automatically restart after saving changes.



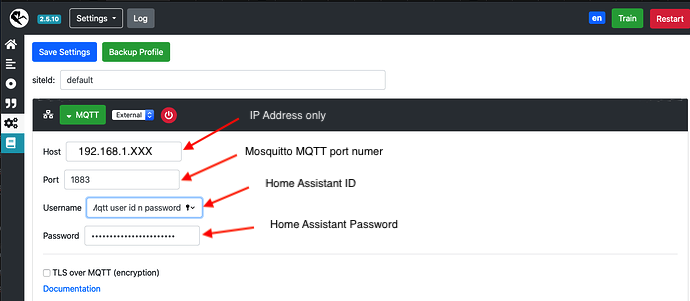
Advanced Page

Direct interface for editing your profile. Be careful! Entering invalid settings here can cause Rhasspy to not start.

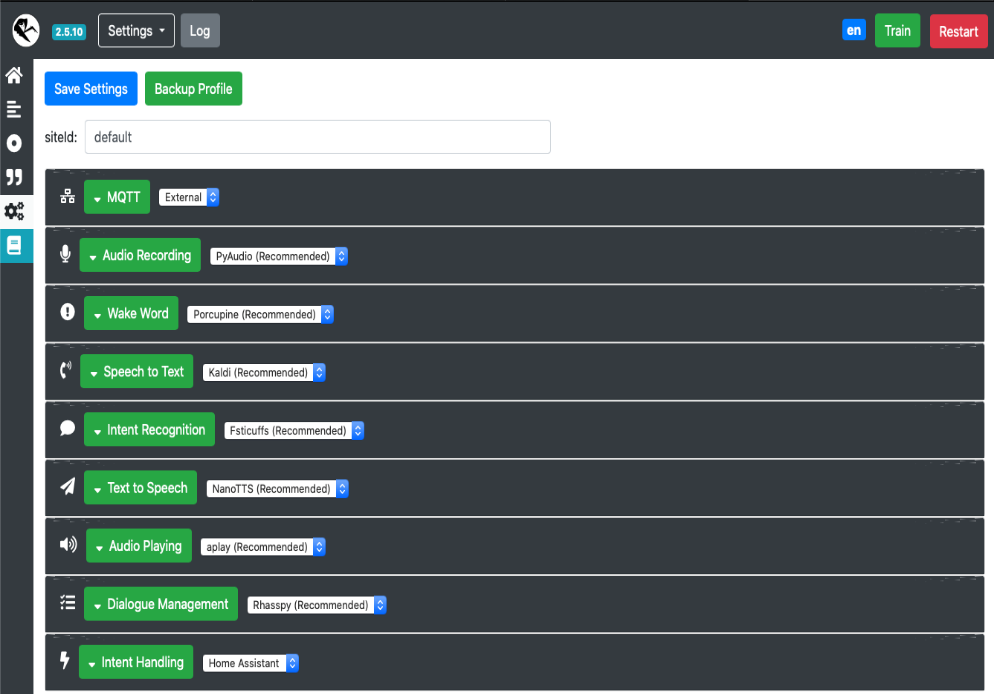


Configure, create intent and test Rhasspy

Key configuration is External MQTT in order to exchange text messages with Home Assistant.



Follow by remaining setting as per below





After completing all configurations, click on “Save Settings” and Rhasspy will save settings and prompt to restart Rhasspy. Click “Ok” to restart and Rhasspy will then prompt to download 20+ files.

These are mainly Speech to Text and Text to Speech files that will take 3-5 minutes to download. Once completed, Rhasspy will initiate “Training Profile” for another 15 minutes to ensure all the Speech profiles have been trained.

There are sample sentences and slots predefined in the Sentence Slots option.

On the home page, you can test your audio in into your USB mini microphone and audio out thru your speaker by clicking on the yellow “Wake Up” button.

Do some voice command testing in Rhasspy by calling the Wake Word you have set.

Once you hear a response beep from Rhasspy, say your command (i.e. Turn on the Lights). If Rhasspy has heard your command, there will be an acknowledgement beep.

Now click on JSON to see the processing logic. And also click on the Log to check if your command was processed and the text message has been sent to Home Assistant with no errors.

As you have yet to create any automation in Home Assistant, your voice command will not have any action or response from Home Assistant yet.

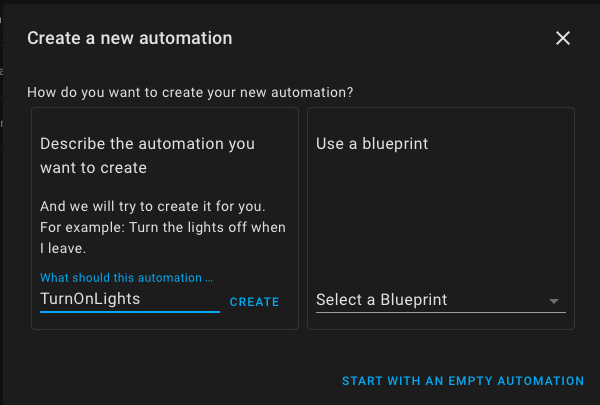
So, it is time to create Event Automation in Home Assistant to match up the Events defined in Rhasspy to be able to perform an end-to-end test.

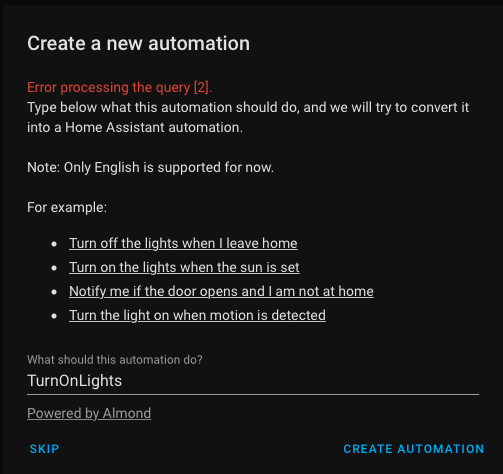
Automation

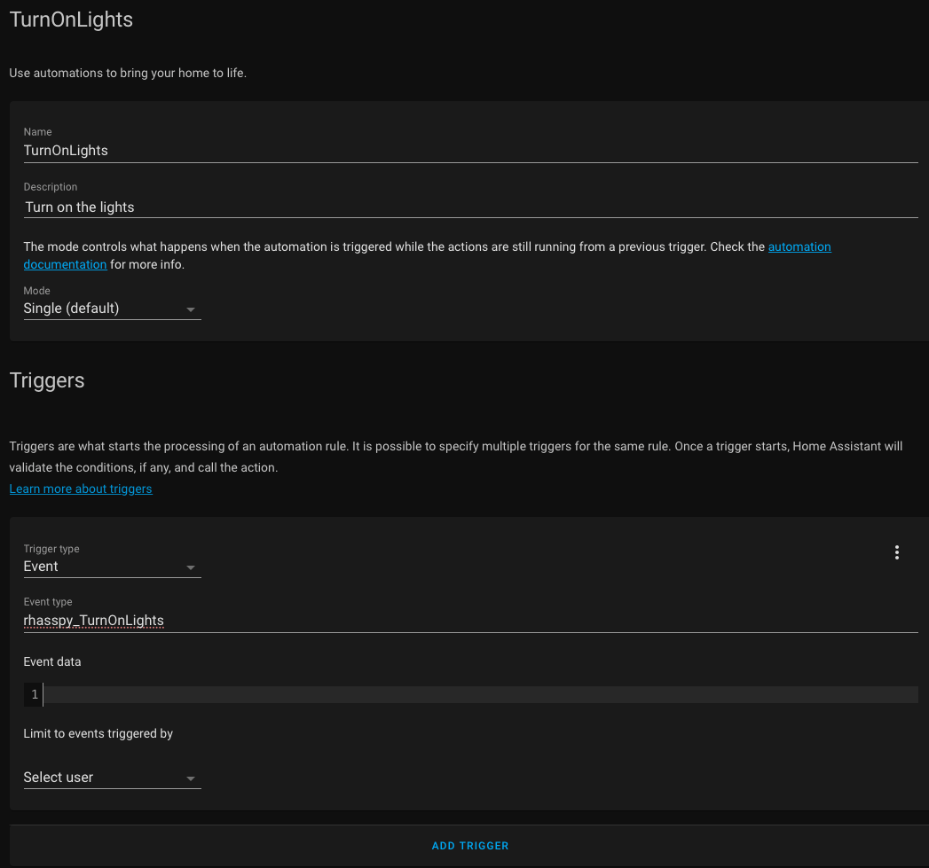
## **Create new Event Automation in Home Assistant**

At Home Assistant home page go to Configure => Automation => Add Automation.

Give this new automation a Name and click Create => Skip. You now enter the Name and Description. At the Trigger section, enter Event ID (to start with rhasspy\_Event Name defined in Rhasspy) and move on to the Action section to add 2 actions. First is to have Home Assistant to trigger (i.e. turn on the light) via Entity ID Turn on the light service. Second action is to send a message back to Rhasspy via Publish MQTT service.







Select Service Call option, select Entity ID “Turn On Light” and select the location of the light, Device ID and Integration ID to complete the first action and move to second action.

Select Service Call option, select Entity ID “Publish\_MQTT). Copy paste the below in BOLD into topic, payload\_template. Note that this section will change to yaml instead of Visual Studio Code which is normal. Also do take note that this is just a model. You can amend the payload\_template to tailor suite the event at Home Assistant and Sentences at Rhasspy in order to establish end to end communication.

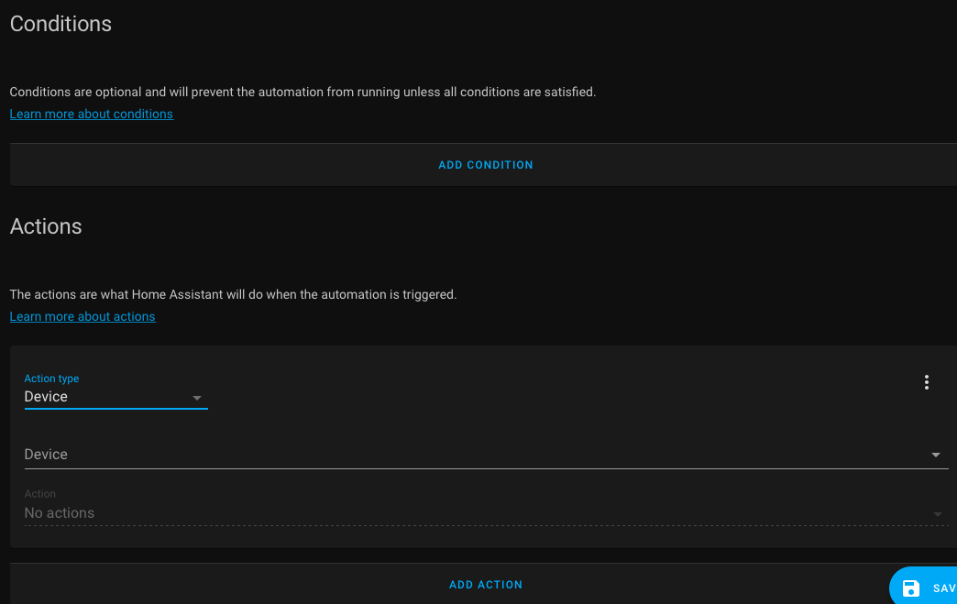
*- service: mqtt.publish*

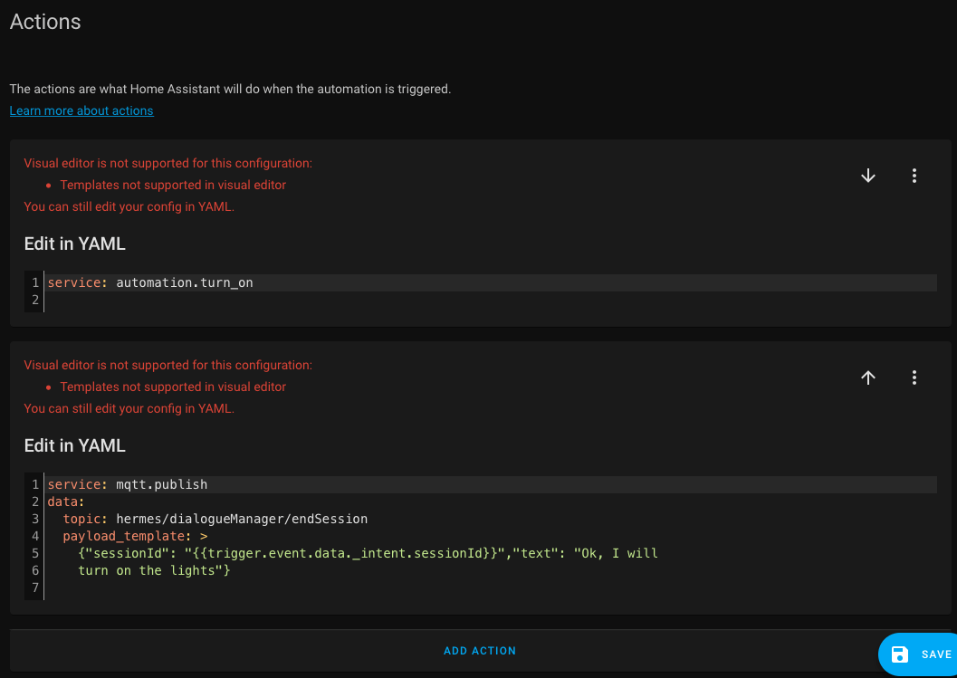
*data:*

*topic: hermes/dialogueManager/endSession*

*payload\_template: '{"sessionId": "{{trigger.event.data.\_intent.sessionId}}",*

*"text": "Ok, the weather today is cloudy day"}*





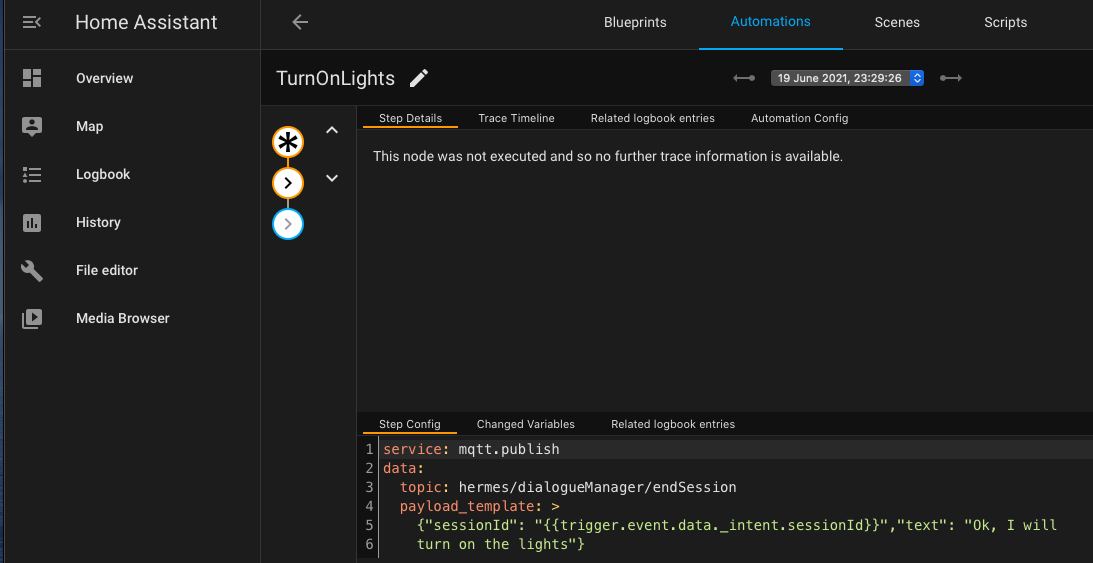
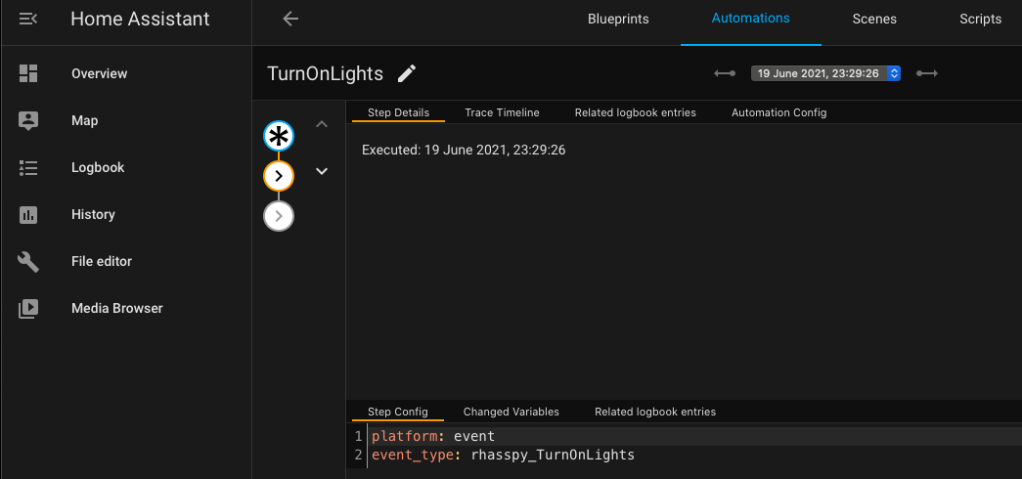
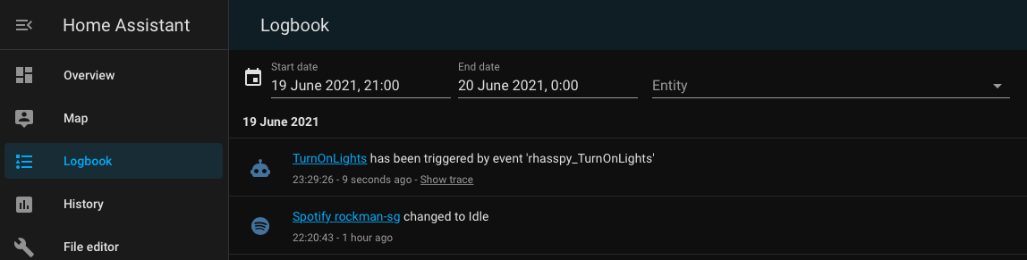
Click Save and your automation is ready for testing. Final step is to perform a Home Assistant restart to ensure all automation has been loaded with all the add-on restarted as well.

To do an end-to-end test, go to Rhasspy and Open Web UI using <HTTP://your> IP address: 12101 (12101 is Rhasspy port number)

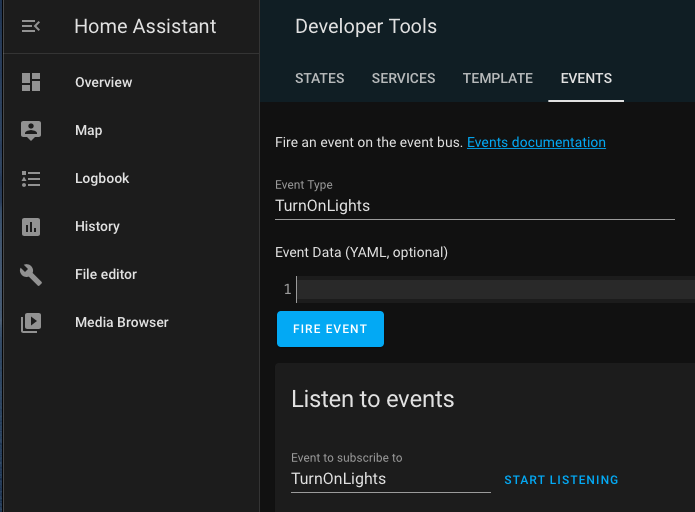
Use Wake Word (i.e Hey Porcupine) to wake Rhasspy up. Once you hear a response beep, say your voice command of “Turn on the lights”. Rhasspy will have an acknowledgement beep and look for matching Sentences. They will process your command, send “rhasspy\_TurnOnLights” event to Home Assistant. Home Assistant will match “rhasspy\_TurnOnLights” event with the defined Automation event ID of “rhasspy\_TurnOnLights” and process the actions defined which is Turn On Lights entity follow by Publish MQTT response back to Rhasspy. Rhasspy will then read out the response you have configure in the Payload\_Template in Home Assistant (i.e. I have turned on the lights) to complete the end-to-end process of 1 voice command. You can verify the Rhasspy Log to confirm success.

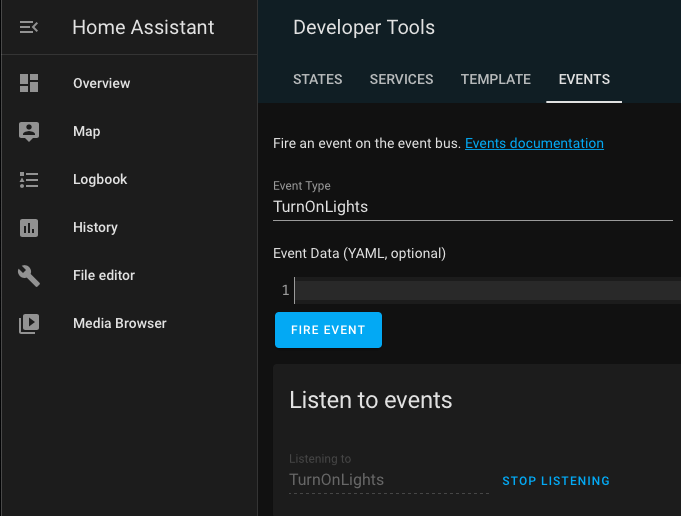
Now go back to Home Assistant home page to do 2 step verifications of this event.

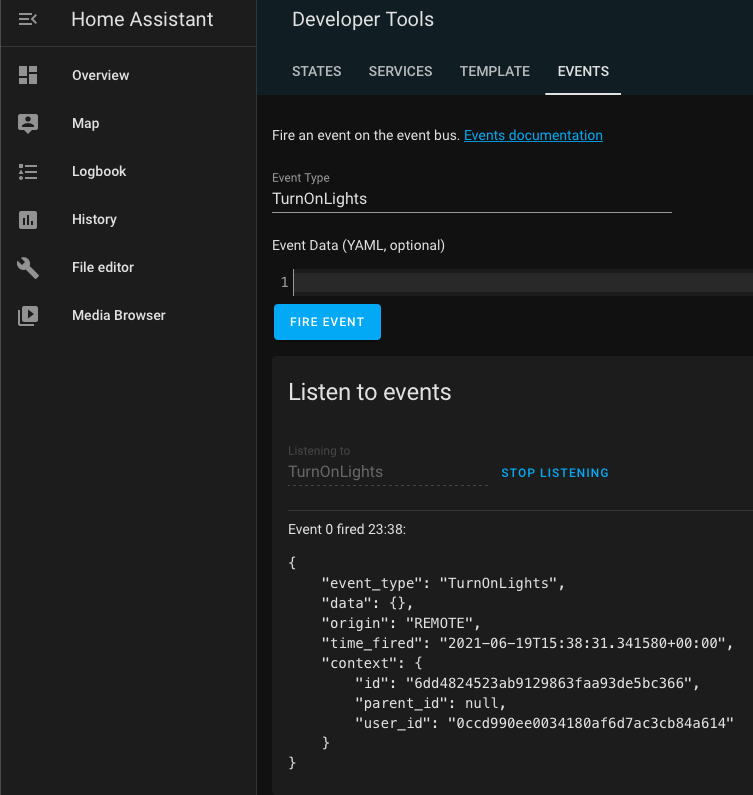
First step is to click on LogBook and Refresh. Check if the Automation you just invoked has been recorded. If it has, click on the Trace right below to check that Event and Actions has been triggered and processed successfully with no errors.



If you didn’t get any Home Assistant response back in Rhasspy, you can use the above method to troubleshoot the problem. There will be more advanced techniques described in the Home Assistant guide using Developer Tools if needed.







Once your first automation end to end test is successful, you now repeat the same Automation steps to define the remaining sample Intent predefined in Rhasspy.

Once all remaining automations have been defined in Home Assistant, you can proceed to test it all.

And voila, you build a safe and secured “Open-Source Smart Home Assistant with privacy” device within your home local network.

Wrap-up

For a first timer like me in this DIY project, I have been overwhelmed with information overload in the internet world to find a simple and straightforward guide for this project. Frankly speaking, there are none out there where you can just take 1 guidebook to cook up one.

This is the first simple guide that I have compiled after spending hours, days and weeks of reading up, watching tons of YouTube, trying and testing out my many prototypes and builds to successfully document this guide.

My intent is to provide a simple guide to newbies like yourself out there in our community to benefit from this guide to have a better head start of this project till the last mile of satisfaction of a working model of Home Automation console.

I hope your entire journey in this DIY project is fun, exciting and with lots of learning too.

By the time you finish reading this guide and completed your first console, you by then have achieved 1 single Dashboard of Home Assistant to a huge number of home automation of any device (you have or to acquire) and integrate a vast number of communication protocol for devices (i.e. Wi-Fi, ZigBee, ZWave, Bluetooth, Thread). No more many different standalone applications on your computer and mobile devices that couldn’t communicate with each other.

So do enjoy your very first successful DIY project with many more automation that can be built and enhanced in the future.

Last but not least, do look out for my future publication about how to build more complex automation with new devices integrated into Home Assistant like Spotify, XiaoMi Roborock Vacuum cleaner, XiaoMi Honeywell Fire Alarm Smoke Detector, XiaoMi Thermostat, Philip Hue lights, Apple TV and many more.

Alternatively, there are many online articles and YouTube that you can learn from after this head start of your DIY project.

Acknowledgement

I would like to express my special thanks to my project mate Alex Tay, Jacelyn Kuan and Zul Fadil for ideas, comments and encouragement in developing this guide as well as a working prototype as well as our Cybersecurity Bootcamp facilitator Jonathan Erez from Cybint.

Romkabouter who is an experience user in Rhasspy community has helped many newbies like me in the forum. It was him/her who has guide and supported me to overcome the challenge specific to Rhasspy External MQTT communication with Home Assistant. Beside solving my issue, he/she was kind enough to also share Home Assistant Automation for MQTT Publish to get Home Assistant to send Text to Speech back to Rhasspy.

Last but not least, I would also like to express my gratitude to Home Assistant Guide, Home Assistant community, Rhasspy 2.5 Guide and Rhasspy community, Everything Smart Home, Tech Tech and More Tech website and YouTube, Dummy Labs, Awesome Open Source, Smart Home Junkie, Lee Tech Channel and Mostly Chris, and KPeyanski for all the contents (User Guide, YouTube and Publications) that I have read and used to develop this guide.

About me

I am a techie who has vast knowledge and experience in regional complex and critical banking application of large installation of sensitive data to protect and secure.

Very passionate about philanthropic activities and strongly believes in giving back to the community in many ways.

I rode cross countries on my 2 wheeler big bike to help give back to the needy. Leverage on my project management skills, I easily get into the inner circle of large charitable event organizing committees to help enhance and improve on project planning, management and execution remotely from my home country and while riding to such events too.

My past time achievement and satisfaction is thru creative ideas and innovation of my DIY such as this project.

The initial idea of this project came about when my team and I were tasked to complete a final project for a Cybersecurity Bootcamp I attended.

As we chosen the “Securing Smart Home” topic where we have bounced off several ideas with our facilitator Jonathan Erez.

Given my background and passion, I have volunteered to take on this project and that is how this project was initiated.

References

1. Home Assistant Operating System guide - <https://www.home-assistant.io/getting-started/>
2. 5 Essential Tips for Security on Home Assistant - <https://youtu.be/AXKPIF3aJ6U>
3. Tech Tech and More Tech setup DuckDNS on Home Assistant guide - <https://techtechandmoretech.com/guides/hass-duckdns/>
4. Setup Nginx Proxy Manager with Maria DB - <https://dummylabs.com/posts/2021-03-13-the-easiest-way-to-https-home-assistant/>
5. Remote Access to Your Home Assistant Using DuckDNS - <https://youtu.be/fNzngoXfxzw>
6. Verify port forwarding is working – [www.CanYouSeeMe.org](http://www.CanYouSeeMe.org)
7. Securing external access to your Home Assistant install - <https://youtu.be/OduLIs1d0gk>
8. How to set up Automations in Home Assistant tutorial - <https://youtu.be/KXTs5_x_T5c>
9. Home Assistant WiFi Setup - <https://youtu.be/j35ytqa_Pfc>
10. Access your LOCAL NETWORK with WireGuard Add-On in Home Assistant - <https://youtu.be/biU5l0T4e8w>
11. Home Assistant WireGuard VPN Install and Setup! - <https://youtu.be/aGIg6N9HzSg>
12. Home Assistant MQTT Install and Setup - <https://youtu.be/dqTn-Gk4Qeo>
13. Rhasspy Voice Assistant 2.5 guide - <https://rhasspy.readthedocs.io/en/latest/>
14. Opensource Smart Home Assistant with privacy by design guide - <https://github.com/Roborock5/DIY-Smart-Home-Assistant>
15. Hacking out of the box device - <https://youtu.be/vsHHiHTJqzc>
16. Opensource Smart Home Assistant - <https://youtu.be/NMS7ndeqOL4>