

WISE-WARE installation guide and first run

You are advised to read through this guide at least once before following it.

Contents

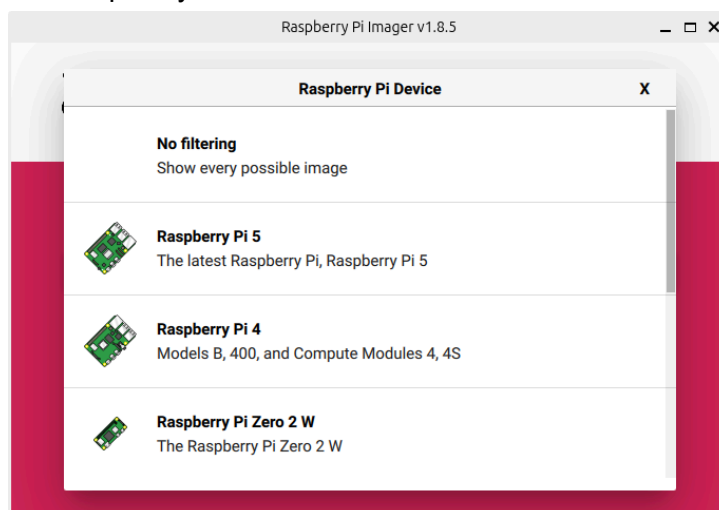
Making installation media	3
Accessing the PI	6
Get PI IP	6
SSH into PI	8
Getting WISE-WARE software	9
Installing WISE-WARE software	10
Go through Debian Docker install process	10
Run regenerator script	10
Running the Docker containers	11
Make an admin account in Portainer before it times out	12
Home Assistant	13
Node-RED install, long-term access token	14

Making installation media

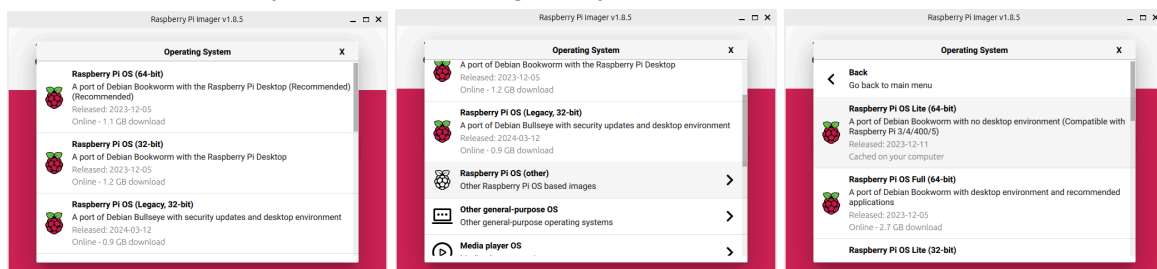
Download, install and run the Raspberry Pi Imager (<https://www.raspberrypi.com/software/>).



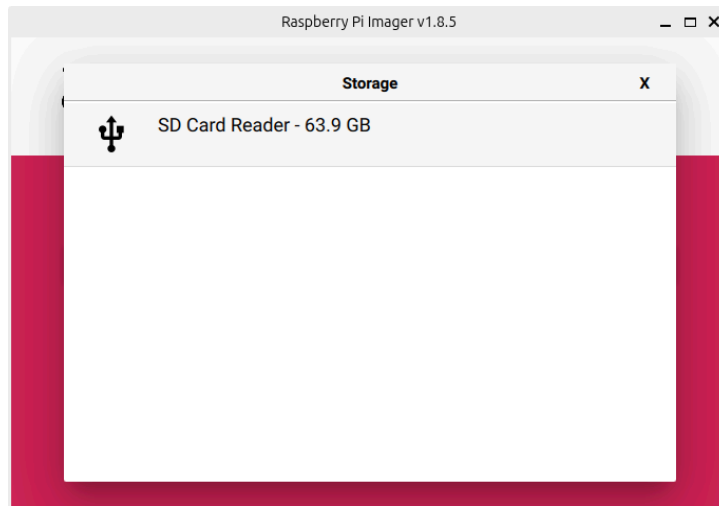
Choose device type. For this example, we will choose the Raspberry Pi 4, but WISE-WARE is compatible with the Raspberry Pi 5 as well.



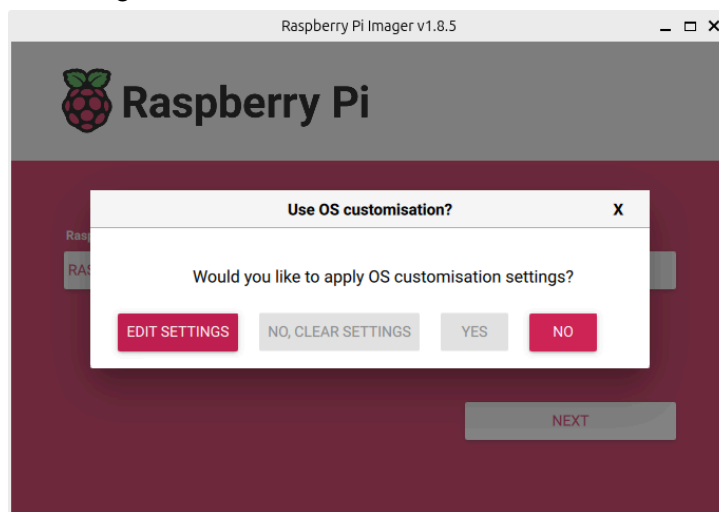
Choose the operating system (OS). Stock WISE-WARE is intended to run on Raspberry Pi OS Lite, but you could use the normal version if you wanted to. In our case, as this installation process has already been done with this laptop before, the OS is already cached on the computer. For your first time doing this, you will need to wait for it to download.



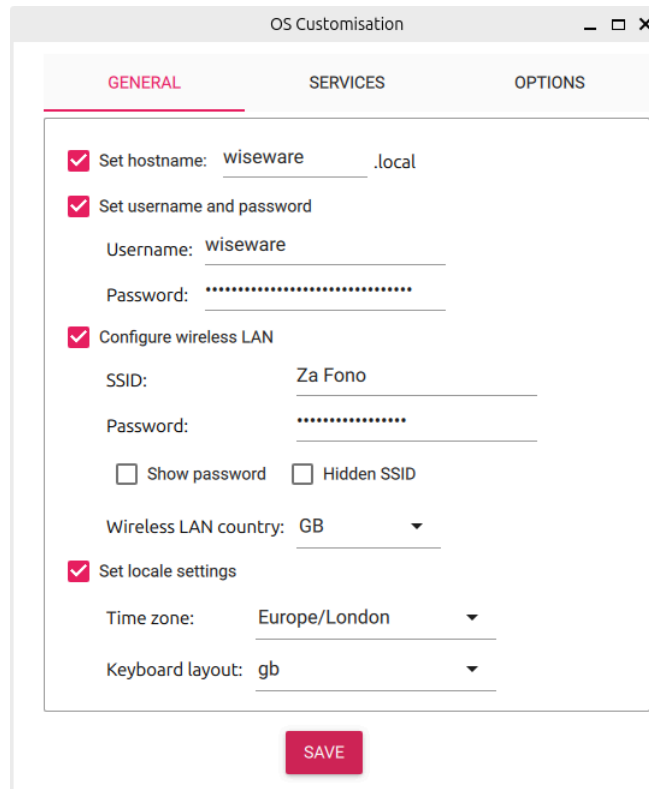
Select your Micro SD card; we are using an external card reader, but yours may be named differently.



Click next, then “Edit settings”.



Configure the settings as shown; it may have already entered the details necessary for connecting to your router, otherwise enter them yourself. This will allow your WISE-WARE to connect to your router on it's first run, so you can download the relevant installation files and remotely connect to it from another computer on the same network. Make sure your user account has a good password!



OS Customisation

GENERAL SERVICES OPTIONS

☒ Set hostname: wiseware.local

☒ Set username and password

Username: wiseware

Password:

☒ Configure wireless LAN

SSID: Za Fono

Password:

☐ Show password ☐ Hidden SSID

Wireless LAN country: GB

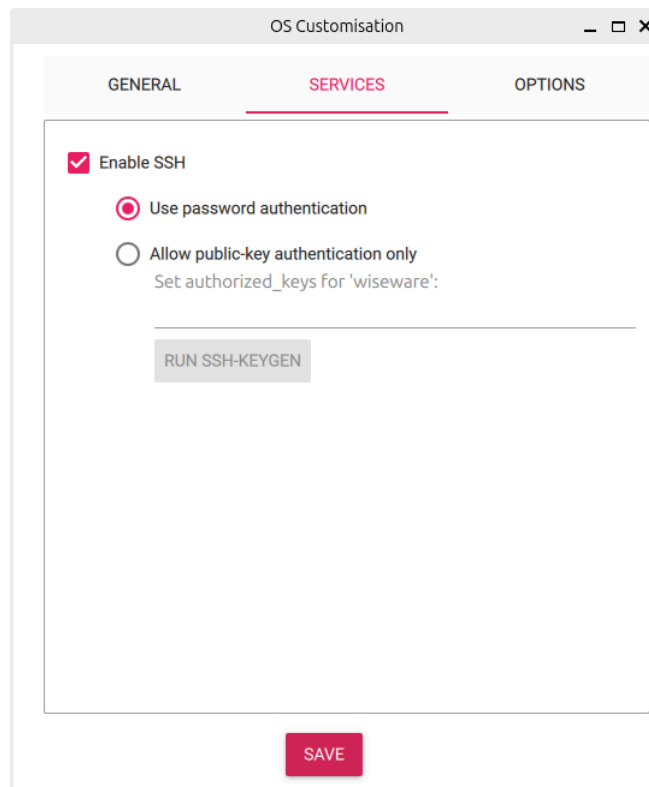
☒ Set locale settings

Time zone: Europe/London

Keyboard layout: gb

SAVE

Go into the “Services” tab, and enable SSH with password authentication. If you are familiar and comfortable with doing so, you can use public-key authentication instead. This step will allow you to connect to the WISE-WARE from another device on the network - such as the one you are doing this step on now - without needing to plug the device into a monitor or keyboard of it’s own.



OS Customisation

GENERAL SERVICES OPTIONS

☒ Enable SSH

☒ Use password authentication

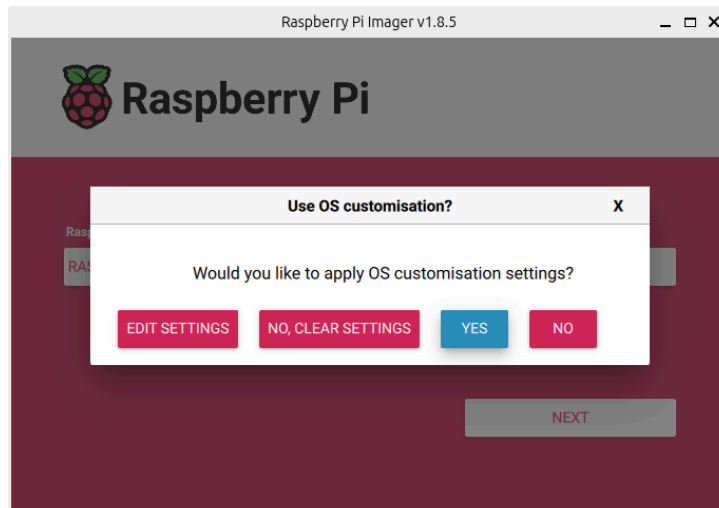
☐ Allow public-key authentication only

Set authorized_keys for 'wiseware':

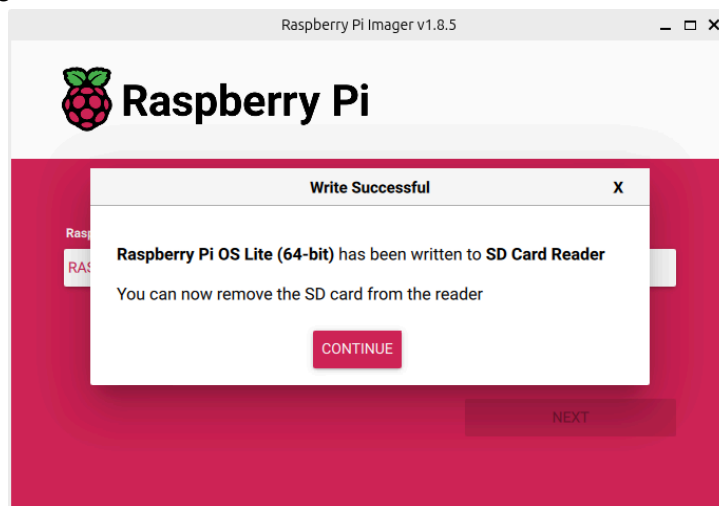
RUN SSH-KEYGEN

SAVE

Click “Save” at the bottom of the window, and “Yes” to go ahead with the installation.



From here, it's just a matter of waiting for the download and installation; when it's done you can close the imager and remove the Micro SD card.



Insert the Micro SD card into the Pi, connect it to the power, and wait for it to go through it's initial startup.

Accessing the PI

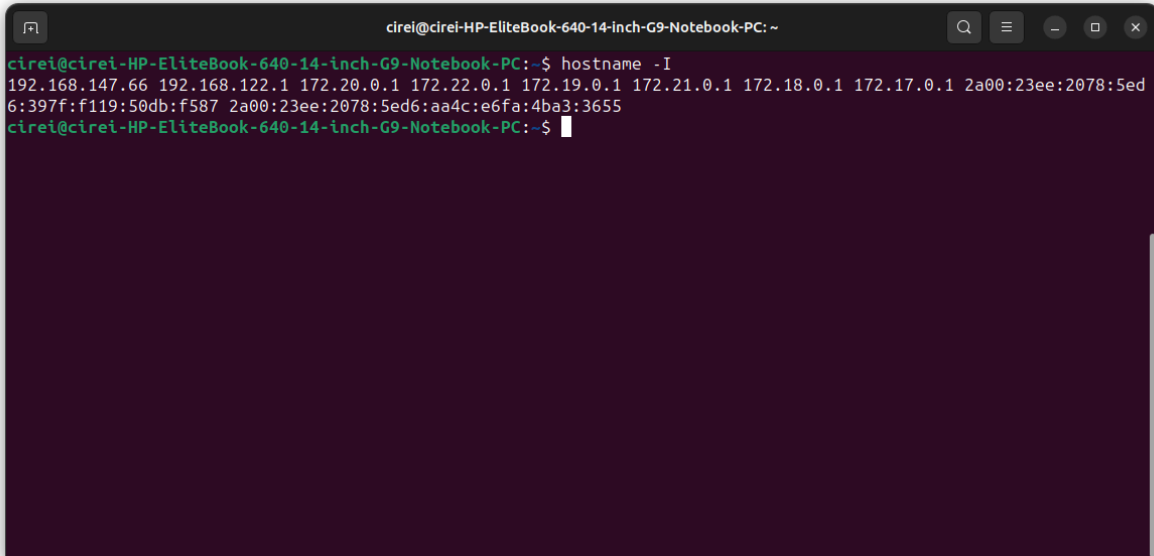
Get PI IP

If you set the host name as “wiseware” as was shown in the previous steps, you can access the Pi through “wiseware” in place of needing to use an IP address. If you did not, or there is some other reason the host name cannot resolve, you will need to access the Pi through it's IP address on the network; the following steps will assume that this is the case.

If you have access to your router's administrator panel, you may be able to get the IP address easily. If you do not have access to the administrator panel, you will need to find the address yourself.

First, we want to find the address of the machine we are working on - in this case, we are using a laptop - so we know what range of addresses the Pi is likely to be in. Make sure you are connected to the same network the Pi will be connecting to, open up a terminal window, and run:

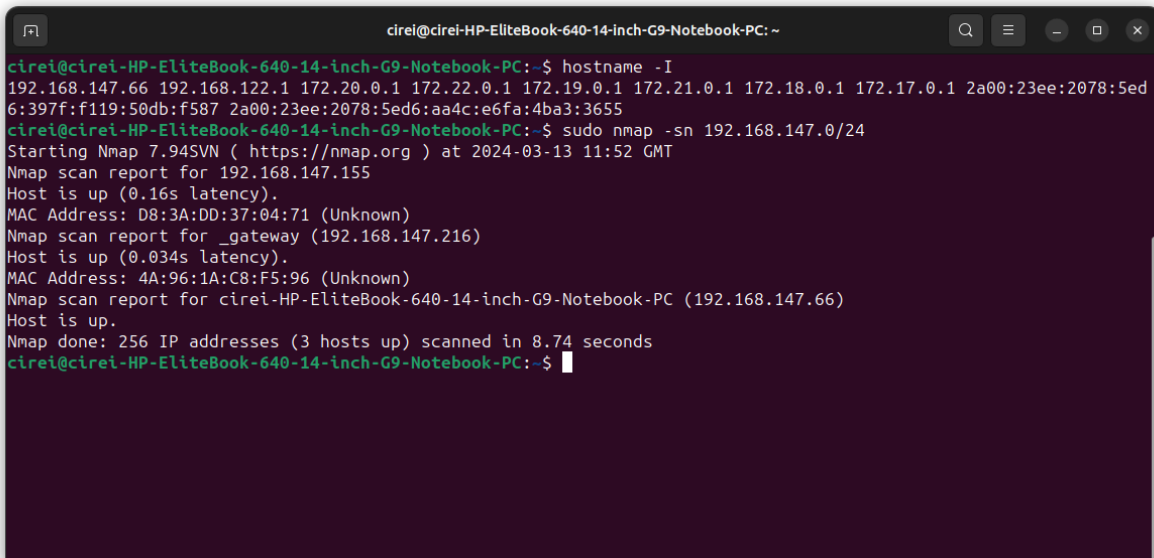
```
hostname -I
```



```
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC: ~  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$ hostname -I  
192.168.147.66 192.168.122.1 172.20.0.1 172.22.0.1 172.19.0.1 172.21.0.1 172.18.0.1 172.17.0.1 2a00:23ee:2078:5ed6:397f:f119:50db:f587 2a00:23ee:2078:5ed6:aa4c:e6fa:4ba3:3655  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$
```

The first address shown is that of our laptop, so we will use that as a basis; we will search for other addresses from 192.168.147.0 to 192.168.147.255 with:

```
sudo nmap -sn 192.168.147.0/24"
```



```
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC: ~  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$ hostname -I  
192.168.147.66 192.168.122.1 172.20.0.1 172.22.0.1 172.19.0.1 172.21.0.1 172.18.0.1 172.17.0.1 2a00:23ee:2078:5ed6:397f:f119:50db:f587 2a00:23ee:2078:5ed6:aa4c:e6fa:4ba3:3655  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$ sudo nmap -sn 192.168.147.0/24  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 11:52 GMT  
Nmap scan report for 192.168.147.155  
Host is up (0.16s latency).  
MAC Address: D8:3A:DD:37:04:71 (Unknown)  
Nmap scan report for _gateway (192.168.147.216)  
Host is up (0.034s latency).  
MAC Address: 4A:96:1A:C8:F5:96 (Unknown)  
Nmap scan report for cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC (192.168.147.66)  
Host is up.  
Nmap done: 256 IP addresses (3 hosts up) scanned in 8.74 seconds  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$
```

Of the 3 addresses here, the second and third are already known - `_gateway` is the phone being used as a router, and `cirei-HP-Elitebook...` is our laptop, so we will assume in this case that the third might be WISE-WARE. We can check by attempting to ssh into it, as shown in the following subsection; other, more involved methods for finding a particular device's IP address are beyond the scope of this brief introduction.

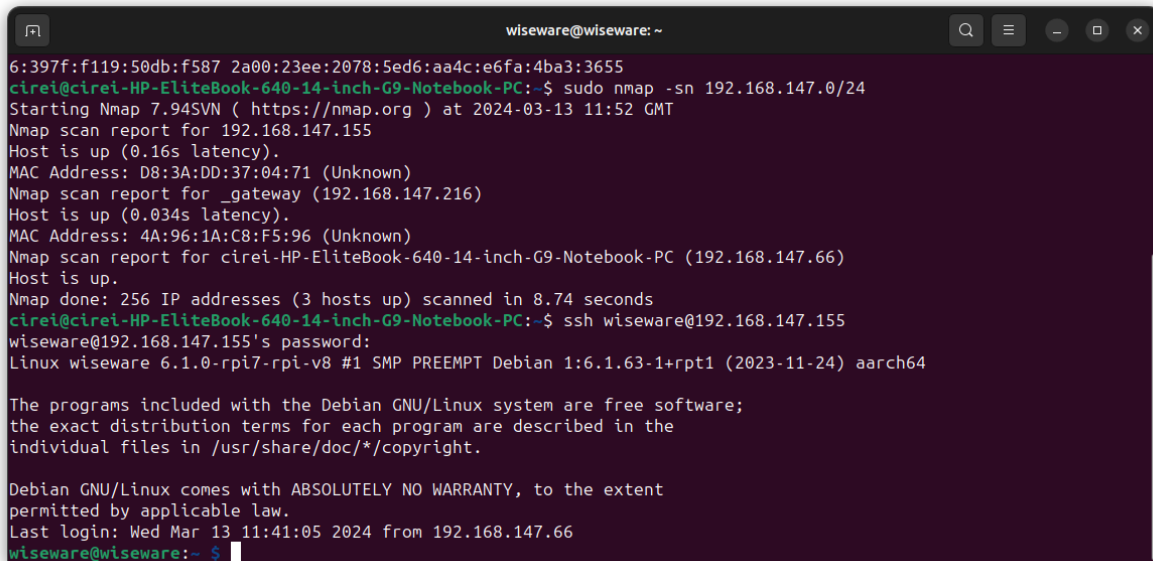
SSH into PI

SSH (Secure SHell) is a networking protocol for securely connecting to a remote device and running commands on it, from another computer such as our laptop. For an introduction to SSH, refer to: <https://www.baeldung.com/cs/ssh-intro>

We will be using SSH to connect to WISE-WARE, and issue commands to go through the installation process. First, the connection itself:

```
ssh wiseware@192.168.147.155
```

(your IP address will look different):



```
wiseware@wiseware: ~  
6:397f:f119:50db:f587 2a00:23ee:2078:5ed6:aa4c:e6fa:4ba3:3655  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$ sudo nmap -sn 192.168.147.0/24  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 11:52 GMT  
Nmap scan report for 192.168.147.155  
Host is up (0.16s latency).  
MAC Address: D8:3A:DD:37:04:71 (Unknown)  
Nmap scan report for _gateway (192.168.147.216)  
Host is up (0.034s latency).  
MAC Address: 4A:96:1A:C8:F5:96 (Unknown)  
Nmap scan report for cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC (192.168.147.66)  
Host is up.  
Nmap done: 256 IP addresses (3 hosts up) scanned in 8.74 seconds  
cirei@cirei-HP-EliteBook-640-14-inch-G9-Notebook-PC:~$ ssh wiseware@192.168.147.155  
wiseware@192.168.147.155's password:  
Linux wiseware 6.1.0-rpi7-rpi-v8 #1 SMP PREEMPT Debian 1:6.1.63-1+rpt1 (2023-11-24) aarch64  
  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Wed Mar 13 11:41:05 2024 from 192.168.147.66  
wiseware@wiseware:~$
```

This will start us in the /home/wiseware directory.

Getting WISE-WARE software

If you already have the files for WISE-WARE on another device, you can transfer them across using your preferred method: copying them to the SD card, using the “scp” command to copy over the network for example. If not, you will need to download them directly to the Pi. In the /home/wiseware directory:

```
curl  
https://github.com/CIREIEmergence/WISE-WARE-Release/releases/download/v1  
.0.0/WISE-WARE-v1.0.0.zip -OL
```

This will download a .zip file, which needs extracting; we will also delete the .zip file afterwards as it will be unneeded:

```
unzip WISE-WARE-v1.0.0.zip && rm WISE-WARE-v1.0.0.zip
```

This will produce a folder with the scripts and configuration files needed for installing WISE-WARE, using the subsequent sections.

Installing WISE-WARE software

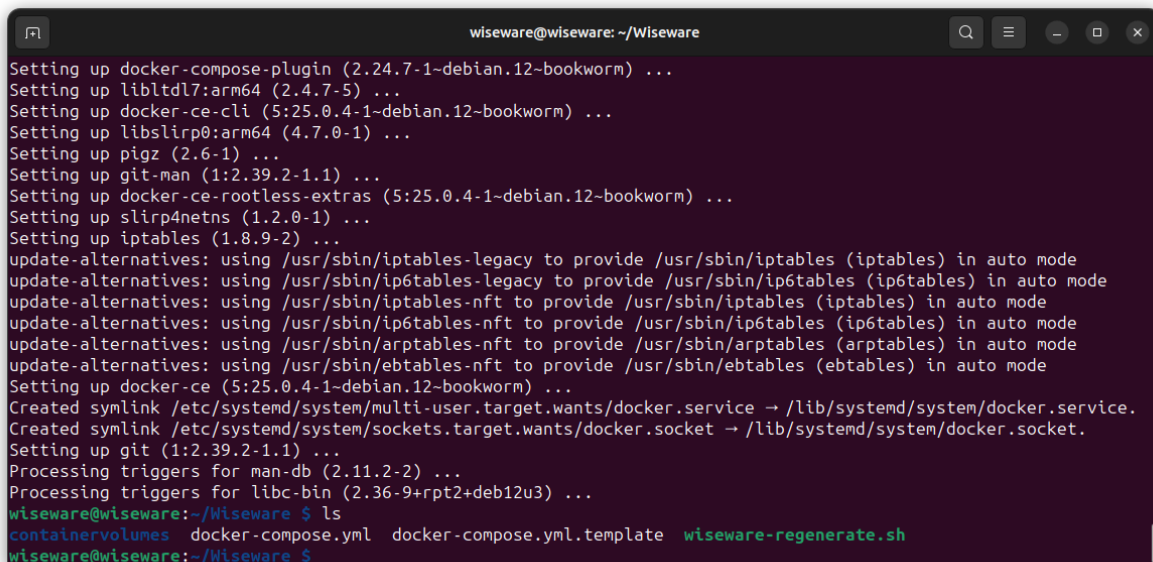
Go through Debian Docker install process

Once the software is downloaded, we need to install Docker, a key prerequisite for running WISE-WARE. There is a script included that will carry out the install process, but if you want to execute those steps yourself you can follow the linked Debian tutorial for installing Docker: <https://docs.docker.com/engine/install/debian/>. The installation script, using commands from the linked guide, is run as:

```
./docker-install-debian.sh
```

Run regenerator script

To confirm that you are in the WISE-WARE software directory, use “ls”:

A terminal window titled 'wiseware@wiseware: ~/Wiseware' showing the output of the 'wiseware-regenerate.sh' script. The script performs various system updates and installations, including docker-compose-plugin, libltdl7, docker-ce-cli, libslirp0, pigz, git-man, docker-ce-rootless-extras, slirp4netns, iptables, and docker-ce. It also creates symlinks for systemd targets and processes triggers for man-db and libc-bin. The terminal output is as follows:

```
wiseware@wiseware: ~/Wiseware
Setting up docker-compose-plugin (2.24.7-1-debian.12-bookworm) ...
Setting up libltdl7:arm64 (2.4.7-5) ...
Setting up docker-ce-cli (5:25.0.4-1-debian.12-bookworm) ...
Setting up libslirp0:arm64 (4.7.0-1) ...
Setting up pigz (2.6-1) ...
Setting up git-man (1:2.39.2-1.1) ...
Setting up docker-ce-rootless-extras (5:25.0.4-1-debian.12-bookworm) ...
Setting up slirp4netns (1.2.0-1) ...
Setting up iptables (1.8.9-2) ...
update-alternatives: using /usr/sbin/iptables-legacy to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/ip6tables-legacy to provide /usr/sbin/ip6tables (ip6tables) in auto mode
update-alternatives: using /usr/sbin/iptables-nft to provide /usr/sbin/iptables (iptables) in auto mode
update-alternatives: using /usr/sbin/ip6tables-nft to provide /usr/sbin/ip6tables (ip6tables) in auto mode
update-alternatives: using /usr/sbin/arptables-nft to provide /usr/sbin/arptables (arptables) in auto mode
update-alternatives: using /usr/sbin/ebtables-nft to provide /usr/sbin/ebtables (ebtables) in auto mode
Setting up docker-ce (5:25.0.4-1-debian.12-bookworm) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Setting up git (1:2.39.2-1.1) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for libc-bin (2.36-9+rpt2+deb12u3) ...
wiseware@wiseware:~/Wiseware $ ls
containervolumes  docker-compose.yml  docker-compose.yml.template  wiseware-regenerate.sh
wiseware@wiseware:~/Wiseware $
```

Run wiseware-regenerate.sh. This will populate the template docker compose file and produce a usable docker compose file. Messages will be printed to the console informing you of what needs to be done next, as well as the steps taken during the regeneration process. Depending on the version of the regenerator (and any extensions made), it may have different numbers and types of messages to those shown in this guide's screenshots:

```

wiseware@wiseware: ~/Wiseware
update-alternatives: using /usr/sbin/ip6tables-nft to provide /usr/sbin/ip6tables (ip6tables) in auto mode
update-alternatives: using /usr/sbin/arptables-nft to provide /usr/sbin/arptables (arptables) in auto mode
update-alternatives: using /usr/sbin/ebtables-nft to provide /usr/sbin/ebtables (ebtables) in auto mode
Setting up docker-ce (5:25.0.4-1~debian.12-bookworm) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Setting up git (1:2.39.2-1.1) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for libc-bin (2.36-9+rpt2+deb12u3) ...
wiseware@wiseware:~/Wiseware $ ls
containervolumes  docker-compose.yml  docker-compose.yml.template  wiseware-regenerate.sh
wiseware@wiseware:~/Wiseware $ ./wiseware-regenerate.sh
docker-compose.yml exists, deleting...
Cloning docker-compose.yml.template and renaming to docker-compose.yml...
Substituting templates in docker-compose.yml...
Substituting <<HOST_IP>>...

Template resolved!
Use 'docker compose down' in the same folder as the compose file to clear out any leftover containers.
Then use 'docker compose up -d' to run WISE-WARE with the regenerated compose file.
When configuring the Apache Kafka integration for Home Assistant, use the IP address 192.168.147.155 and the port
9092.

wiseware@wiseware:~/Wiseware $

```

Running the Docker containers

To do the first startup of WISE-WARE, use “sudo docker compose up -d”. The “-d” flag runs it in detached mode, so you can continue to use the terminal for other tasks while WISE-WARE is running. If this is the first time you are going through these steps on this Pi, you will need to wait for the container images to download, which can take 5-15 minutes depending on the speed and stability of your internet connection.

```

wiseware@wiseware: ~/Wiseware
Substituting templates in docker-compose.yml...
Substituting <<HOST_IP>>...

Template resolved!
Use 'docker compose down' in the same folder as the compose file to clear out any leftover containers.
Then use 'docker compose up -d' to run WISE-WARE with the regenerated compose file.
When configuring the Apache Kafka integration for Home Assistant, use the IP address 192.168.147.155 and the port
9092.

wiseware@wiseware:~/Wiseware $ sudo docker compose up -d
[+] Running 73/5
 ✓ zookeeper 2 layers [#####] 0B/0B Pulled 674.5s
 ✓ homeassistant 29 layers [#####] 0B/0B Pulled 736.7s
 ✓ nodered 16 layers [#####] 0B/0B Pulled 684.0s
 ✓ portainer 10 layers [#####] 0B/0B Pulled 603.0s
 ✓ kafka 11 layers [#####] 0B/0B Pulled 475.6s
[+] Running 5/6
   ⚙ Network wiseware_default Created 62.1s
   ✓ Container zookeeper Healthy 29.7s
   ✓ Container portainer Started 9.3s
   ✓ Container kafka Healthy 54.1s
   ✓ Container homeassistant Started 54.2s
   ✓ Container nodered Started 54.8s
wiseware@wiseware:~/Wiseware $

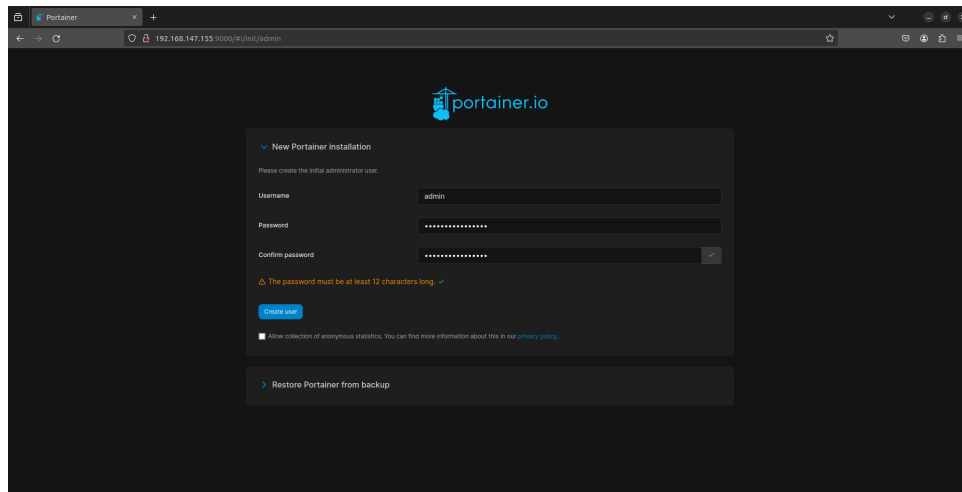
```

The downloads take place in parallel, so the whole process in the above screenshot took around 14 minutes (just over 12 for the downloads, the remainder for the container creation and startup).

Make an admin account in Portainer before it times out

Once the initial download has finished, all of WISE-WARE's services will be up and running. One of them - Portainer - will time out and close if you do not create an admin account within 5 minutes of it's initial startup, you will need to restart the container and try again. This is the only time-sensitive step of the installation process.

First, go to the IP address of the Pi at the port 9000, and follow the on-screen instructions to create your administrator account:



If you do need to restart Portainer, run:

```
sudo docker compose restart portainer
```

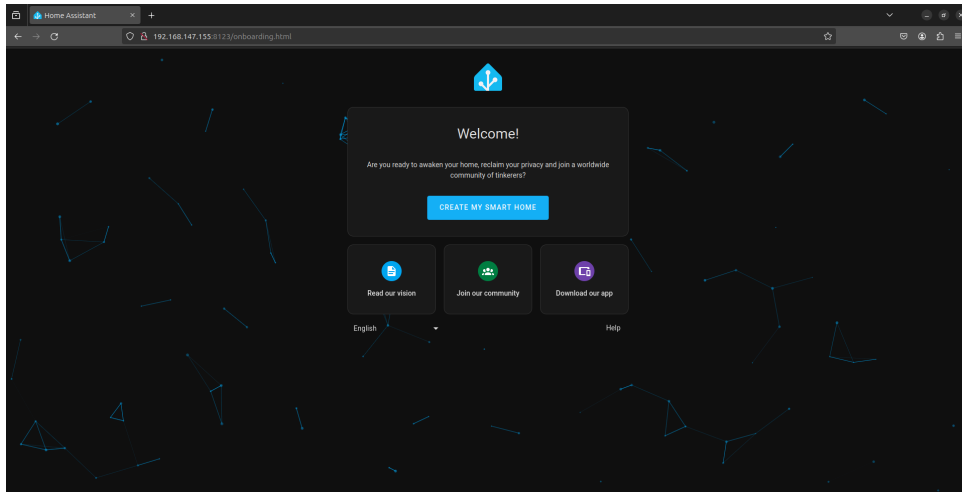
Once it has restarted, go through the admin account creation process above.

```
wiseware@wiseware: ~/Wiseware
Template resolved!
Use 'docker compose down' in the same folder as the compose file to clear out any leftover containers.
Then use 'docker compose up -d' to run WISE-WARE with the regenerated compose file.
When configuring the Apache Kafka integration for Home Assistant, use the IP address 192.168.147.155 and the port 9092.

wiseware@wiseware:~/Wiseware $ sudo docker compose up -d
[+] Running 73/5
 ✓ zookeeper 2 layers [ ] 0B/0B Pulled 674.5s
 ✓ homeassistant 29 layers [ ] 0B/0B Pulled 736.7s
 ✓ nodered 16 layers [ ] 0B/0B Pulled 684.0s
 ✓ portainer 10 layers [ ] 0B/0B Pulled 603.0s
 ✓ kafka 11 layers [ ] 0B/0B Pulled 475.6s
[+] Running 5/6
   Network wiseware_default Created 62.1s
 ✓ Container zookeeper Healthy 29.7s
 ✓ Container portainer Started 9.3s
 ✓ Container kafka Healthy 54.1s
 ✓ Container homeassistant Started 54.2s
 ✓ Container nodered Started 54.8s
wiseware@wiseware:~/Wiseware $ sudo docker compose restart portainer
[+] Restarting 1/1
 ✓ Container portainer Started 0.9s
wiseware@wiseware:~/Wiseware $
```

Home Assistant

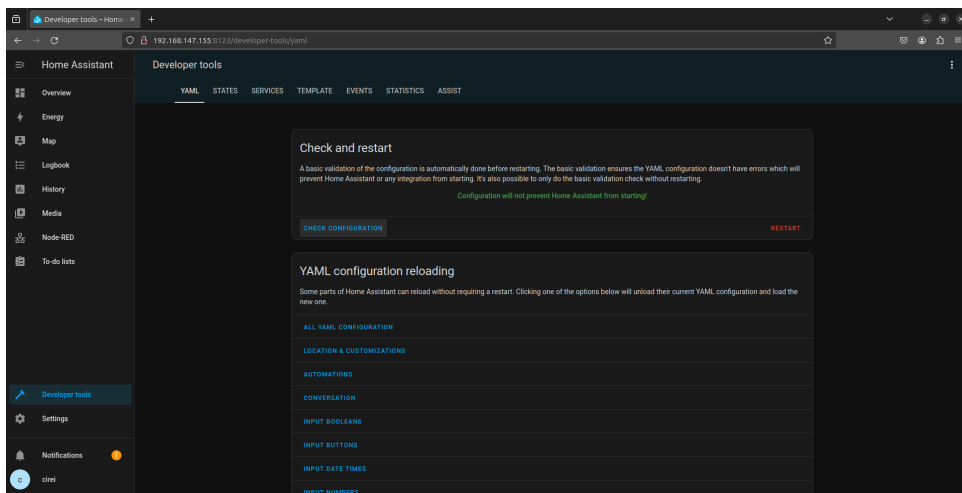
To access the Home Assistant interface, go to the IP address of the Pi at the port 8123 and follow the on-screen instructions.



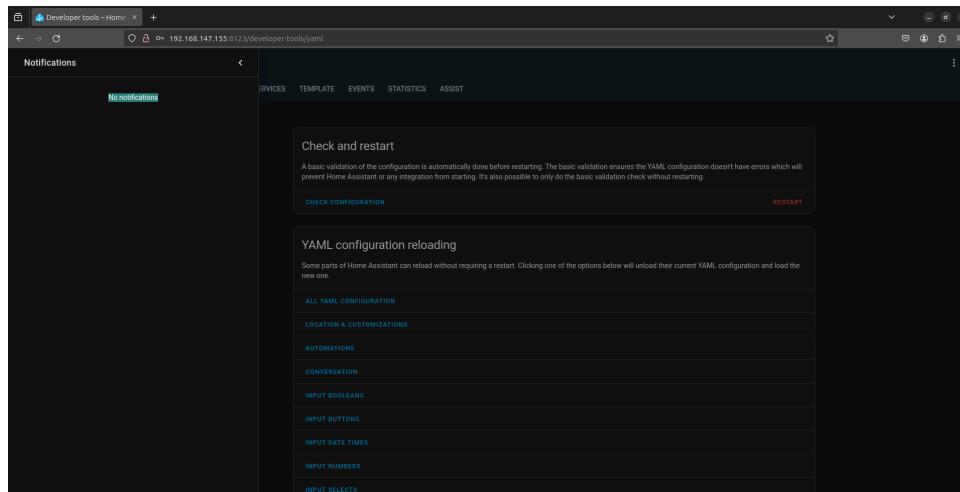
Once you're in, you will need to install the Kafka integration and Node-RED into the sidebar; to do this, you need to move the generated Home Assistant config file into place like so:

```
cp ./configuration.yaml ./containervolumes/homeassistant/config/
configuration.yaml
```

Then, go to the developer tools tab of the Home Assistant interface, click "Check configuration". If there are no problems (and there shouldn't be; closely follow previous steps if there are to diagnose the problem) you can click "Restart" to restart Home Assistant:



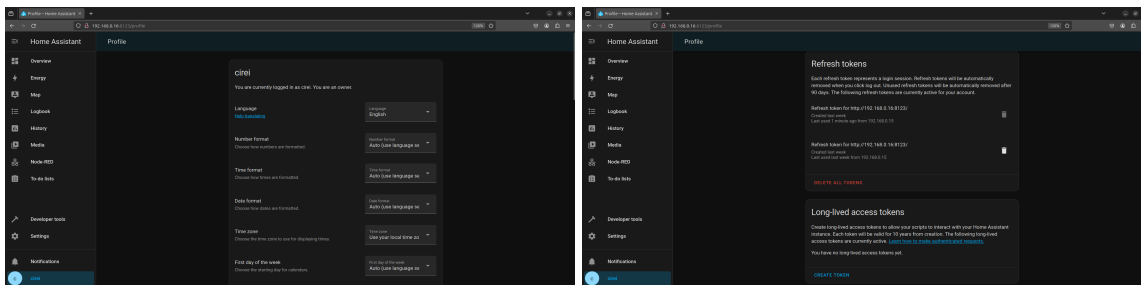
Log back into Home Assistant, and you should see no notifications telling you that there are errors.



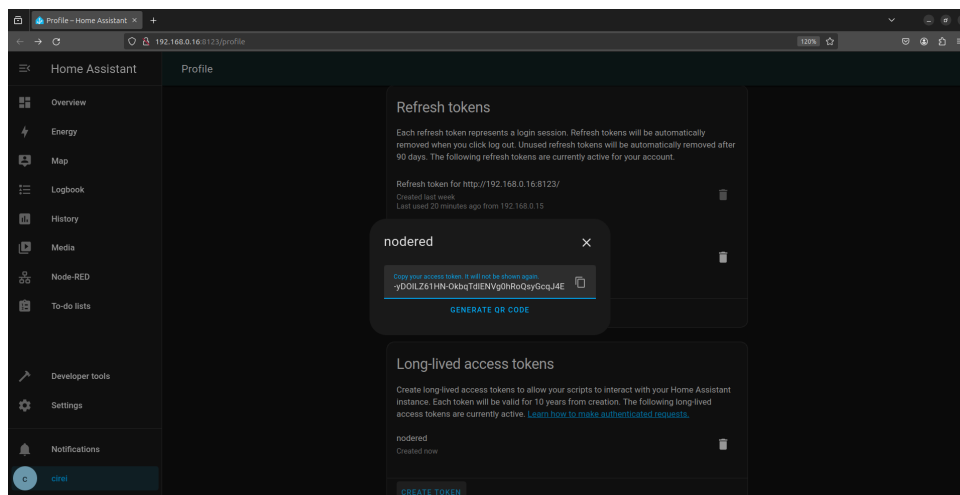
Node-RED install, long-term access token

If you are using stock WISE-WARE - and have installed the Home Assistant config file as earlier described - the panel to access Node-RED will already be in the Home Assistant interface. The remaining installation for Node-RED is getting a new long-lived access token. These are used to allow Node-RED to communicate with Home Assistant; the following instructions will allow Node-RED to react to any events that take place in Home Assistant.

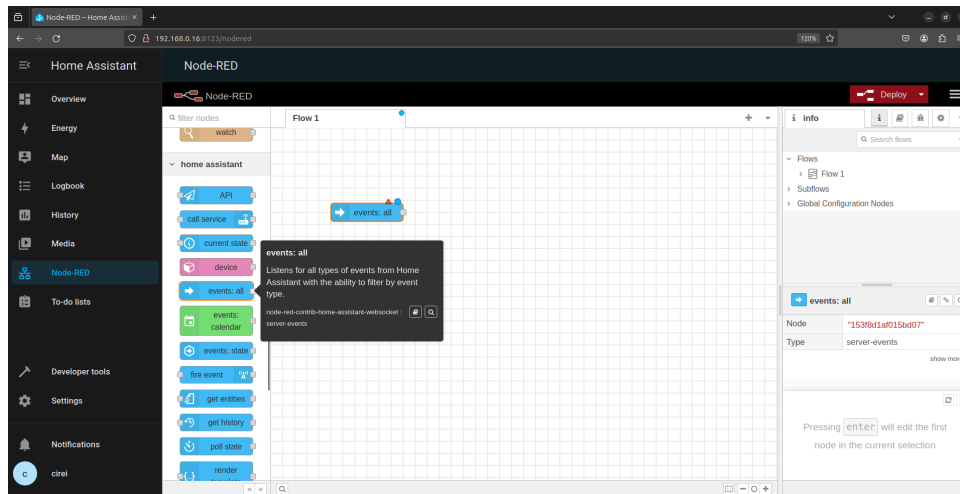
Go to the account's profile page for whichever administrator account you are using, and scroll to the bottom:



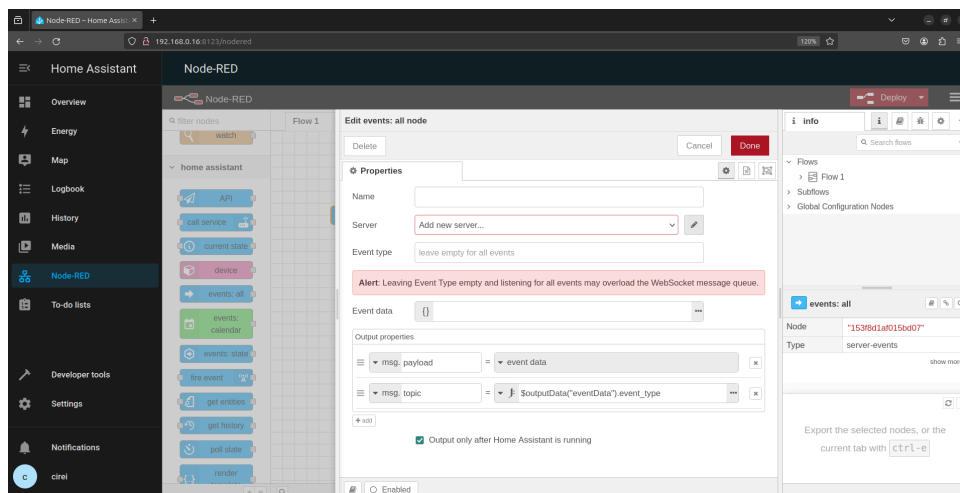
Click 'create token', choose a name, and save the token somewhere safe. You will not be able to retrieve it if it is lost after this step, and will need to make a new one in that case:



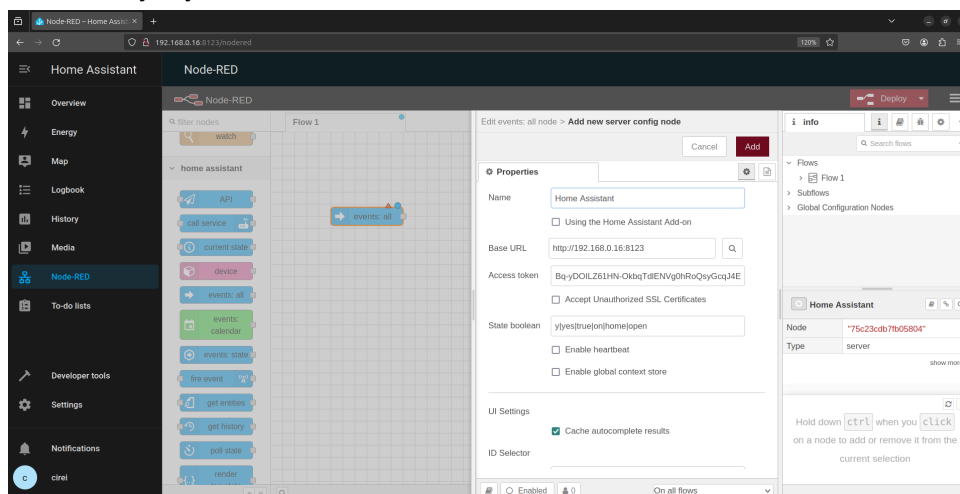
Go to the Node-RED sidebar panel (click through the tutorials if you wish) then make a new “events:all” node, by selecting it from the palette on the left. You will need to scroll down some distance to find the right section:



Double-click to open the configuration menu, then click the pencil button to add a new server configuration:



Configure the URL and access token, then click “Add”; note that we are not using the official add-on for Node-RED, so do not need to toggle that setting, and that the configuration can be named differently if you wish:



Click “Done” in the node’s configuration menu, then “Deploy” in the top right of the interface. You will see a notification like this:

