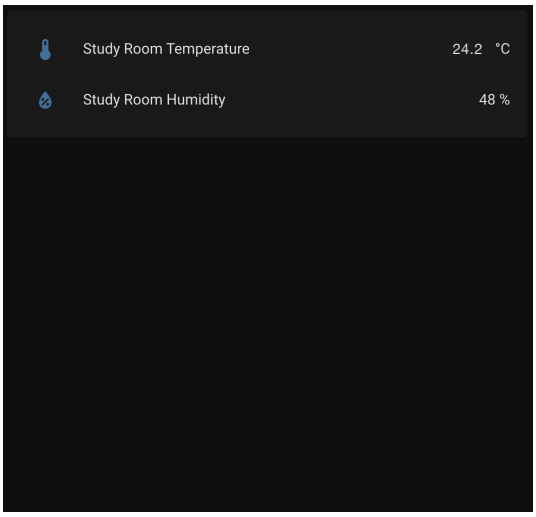
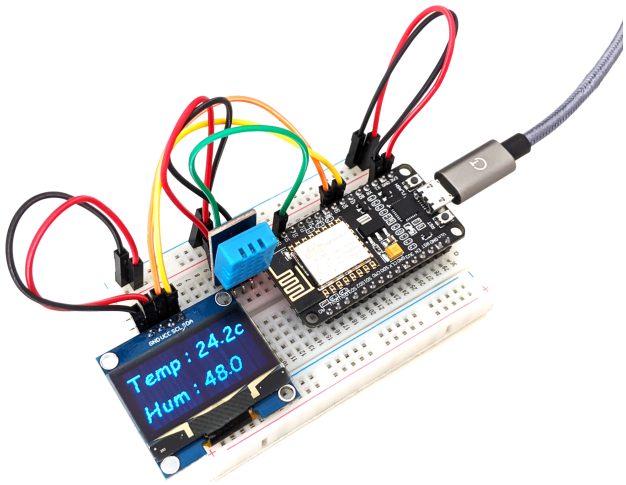
Home Assistant

Temperature and Humidity Sensor

Project description:

In this project, you will create a simple **temperature station** to monitor the temperature and humidity level of your study room. You will integrate your project with **Home Assistant**, an open-source home automation software which you can use as a central control system. Imagine having a single place where all of your house data is stored and monitored in real time, you can turn your place into a smart and dynamic environment.



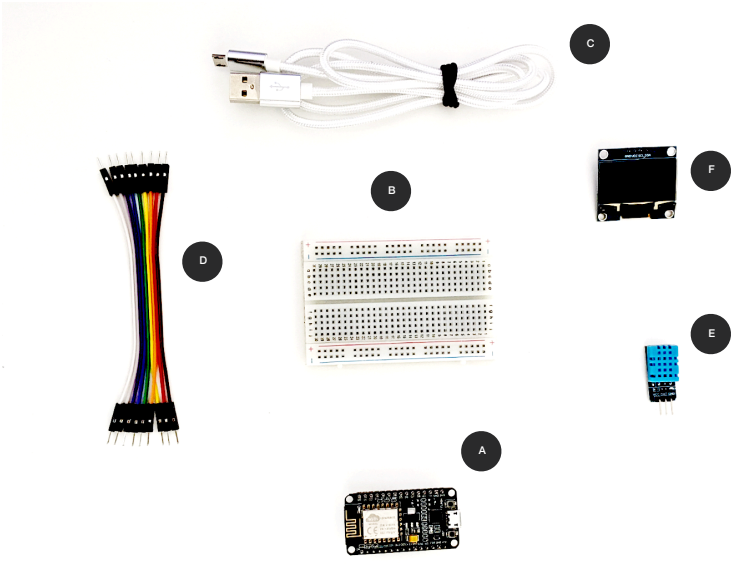


Project objectives:

- Install, configure, and use Home Assistant
- Create a circuit to monitor the temperature and humidity of your room
- Push live temperature data to Home Assistant
- Monitor your data through the Home Assistant interface

Project components:

Component Reference	Component Quantity	Component Name	Component Description
A	1	ESP8266 WiFi Module	A low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability
B	1	Breadboard	A rectangular plastic board with conductive rails for fast circuit prototyping
C	1	Micro USB Cable	A USB cable to power and upload instructions to a microcontroller
D	9	Jumper Wires	Conductive cables frequently used with a breadboard to connect two points in a circuit
E	1	DHT11 Temperature and Humidity Sensor	An electronic device that measures the temperature and humidity of an environment and converts the input data into electronic data
F	1	1.3” OLED IIC LCD Display	A low power consumption LCD display for visual data output



Step One
Installing Home Assistant

Home Assistant is among the most powerful home automation platforms which allows you to easily connect your microcontrollers via wireless protocols and with strong focus on local control and privacy. It is a free and open-source software powered by a community of tinkers which easily runs on a Raspberry Pi microcontroller or a local server.

For the purpose of this project, you will run Home Assistant on a local server trough **VirtualBox**, a free open-source virtualisation tool for x86 and x86-64 hardware. VirtualBox will make the Home Assistant installation and setup independent from your device operating system.

VirtualBox and Home Assistant Download

First thing first, let's download both VirtualBox and Home Assistant:

- VirtualBox: https://www.virtualbox.org/wiki/Downloads

Visit the above URL and download the VirtualBox package compatible with your operative system (e.g. OS X):

- Home Assistant : https://www.home-assistant.io/installation/windows#install-home-assistant-core

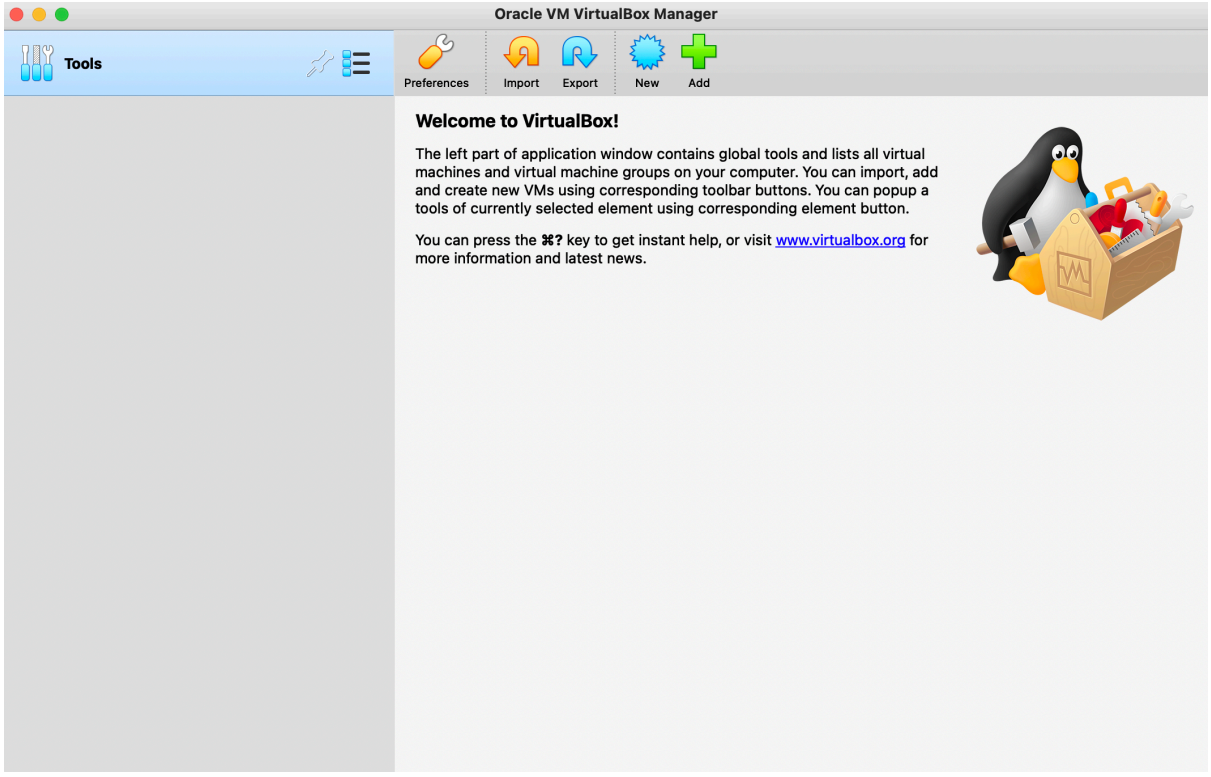
Visit the above URL and download the Home Assistant image compatible with VirtualBox:

At this point you should have two files sitting in your download folder, one which contains the VirtualBox installer and one which contains the Home Assistant image. Now, it is time to install VirtualBox and run Home Assistant from a local server through a virtual machine.

VirtualBox installation

The next step is to install and configure VirtualBox to use the Home Assistant as an image on startup. This means that VirtualBox will act as your operative system running Home Assistant on a private local server that you can access from your browser.

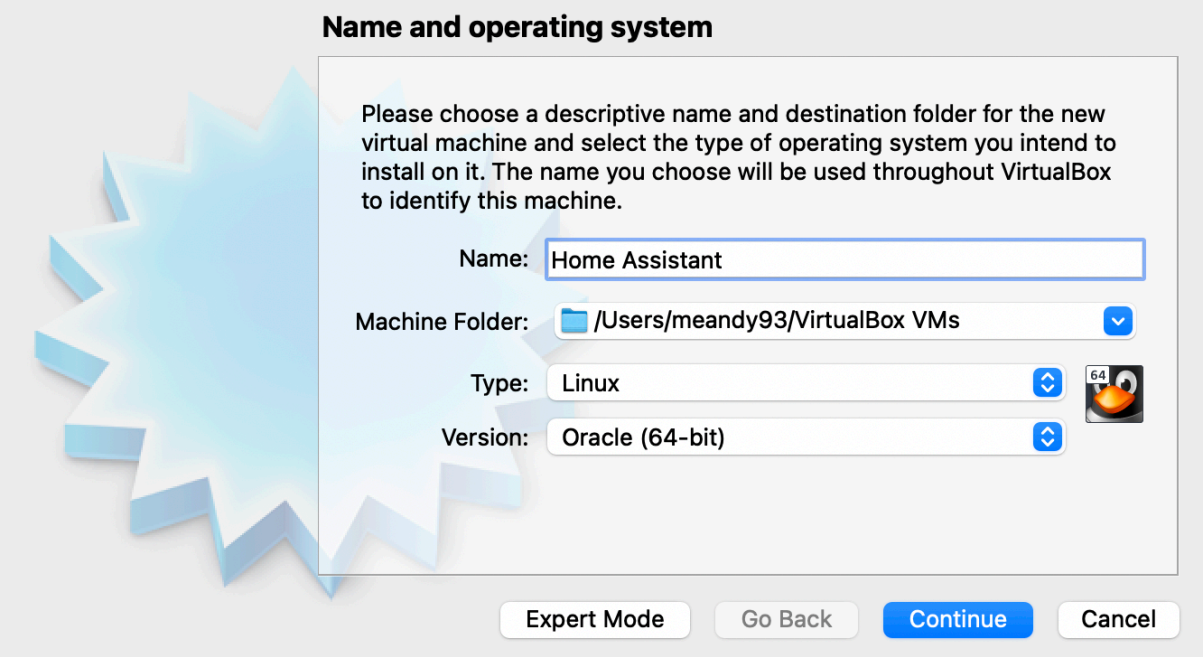
Double click on the dowloaded VirtualBox package and follow the installation instructions. Once the installation has finished, open the VirtualBox application. It should look like the image below:



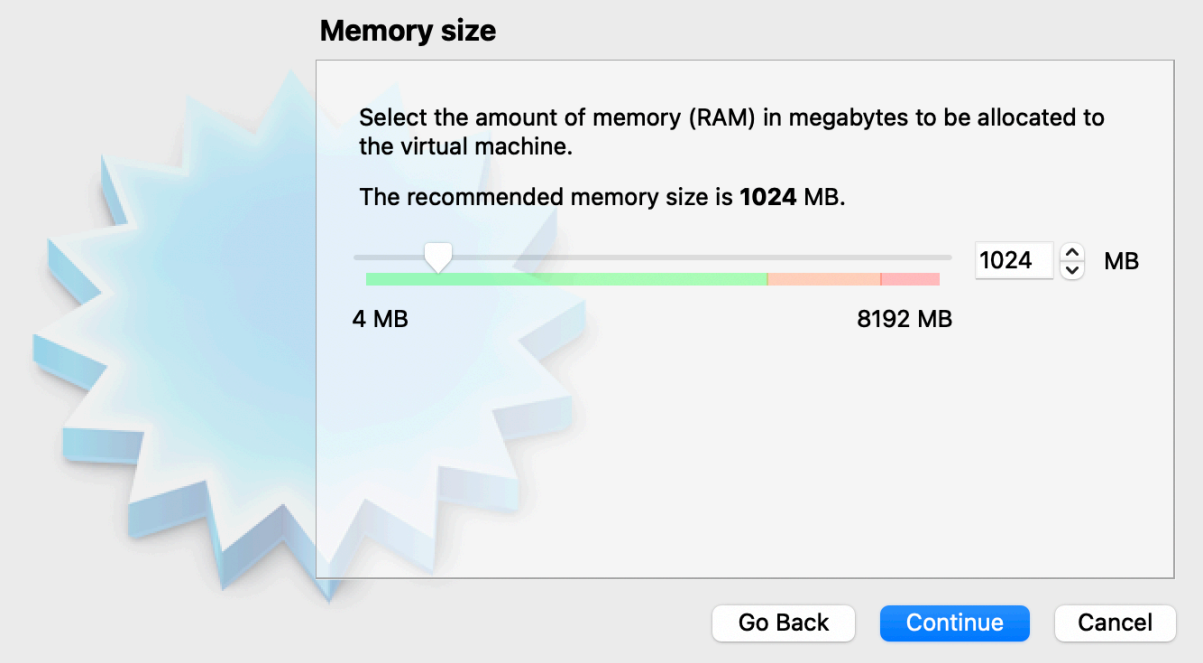
You will not have any virtual machines yet. These are listed on the left hand side column of the application.

Now, let's create a virtual machine which contains the Home Assistant image. Click on the **New** button located almost at the end of the main navigation bar to create a new virtual machine.

This will prompt a pop up window asking you to enter some configuration details for the new virtual machine:



Click on the **Continue** button. This will prompt another window asking you to configure the amount of memory to be allocated to the virtual machine. Select the recommended memory size of 1024MB for now, you can always adjust it later:

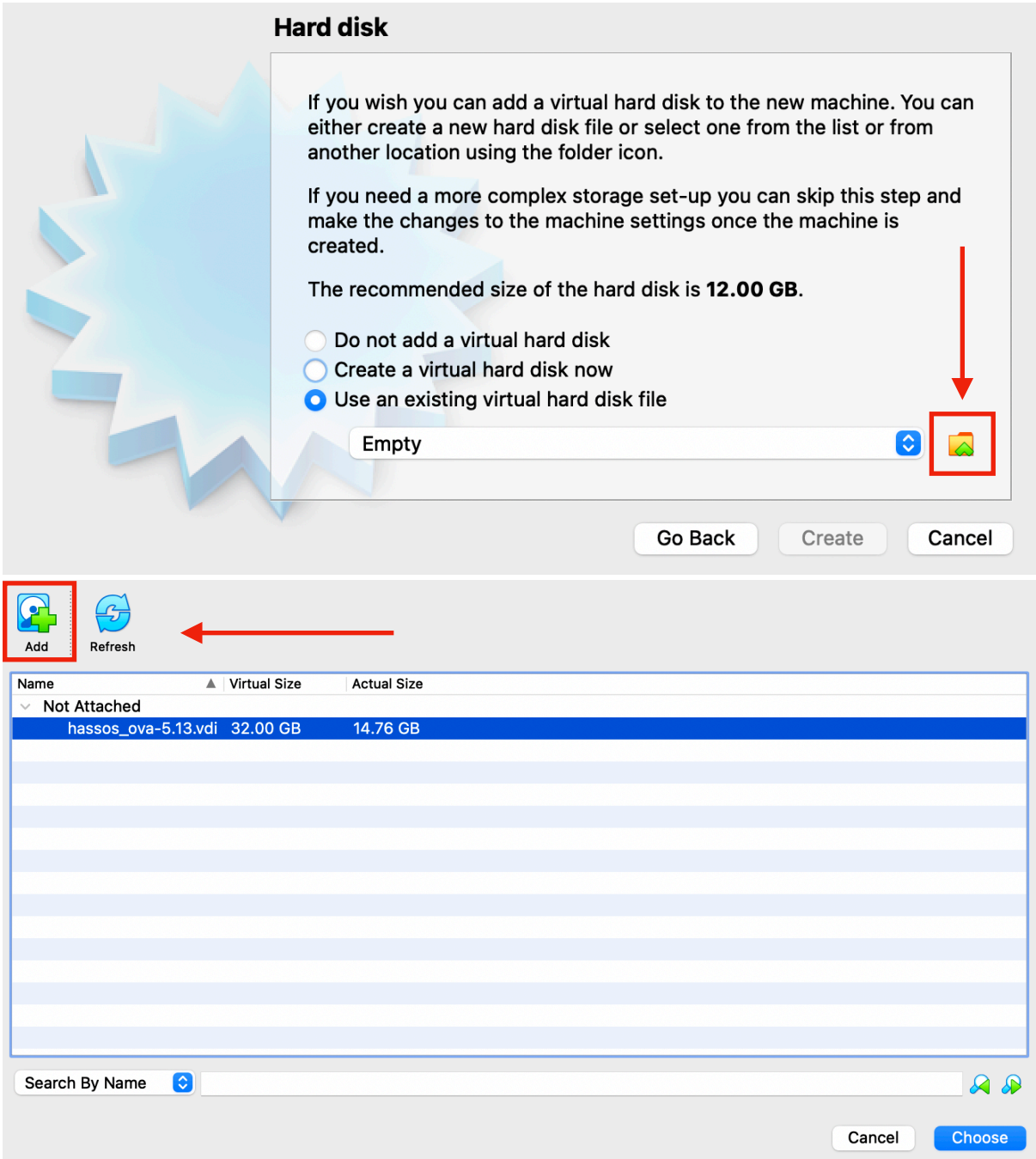


Click on the **Continue** button.

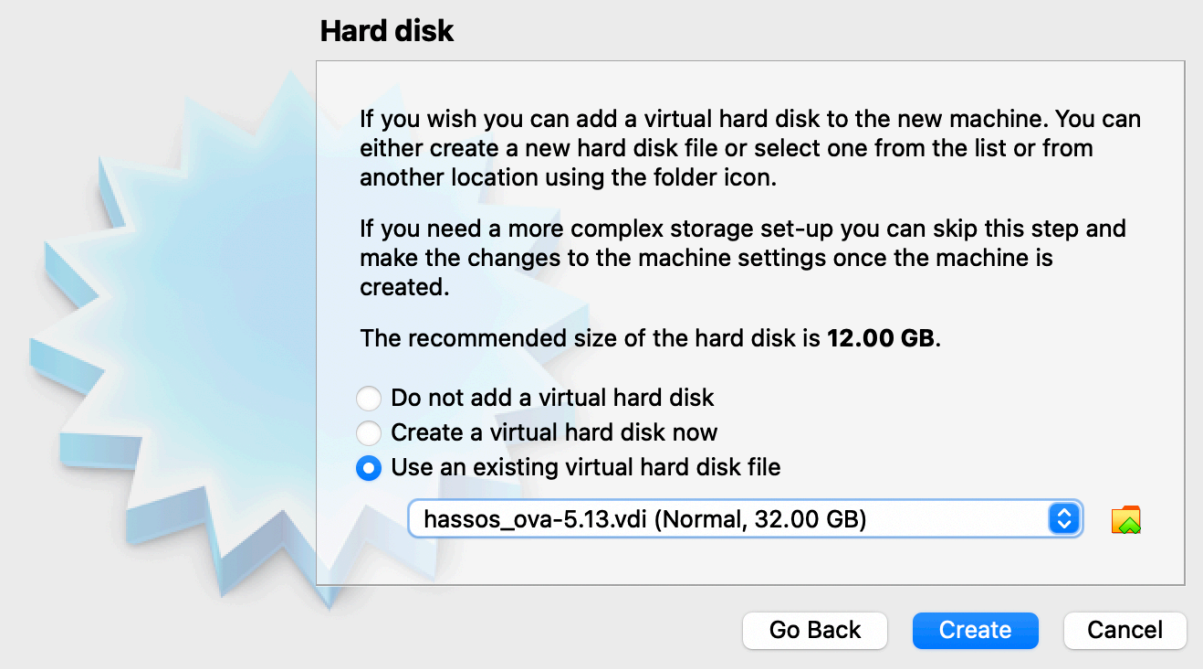
The next pop up window will ask you to select which virtual hard disk you want to add to the virtual machine. This is where we can assign the Home Assistant image that we dowloaded previously.

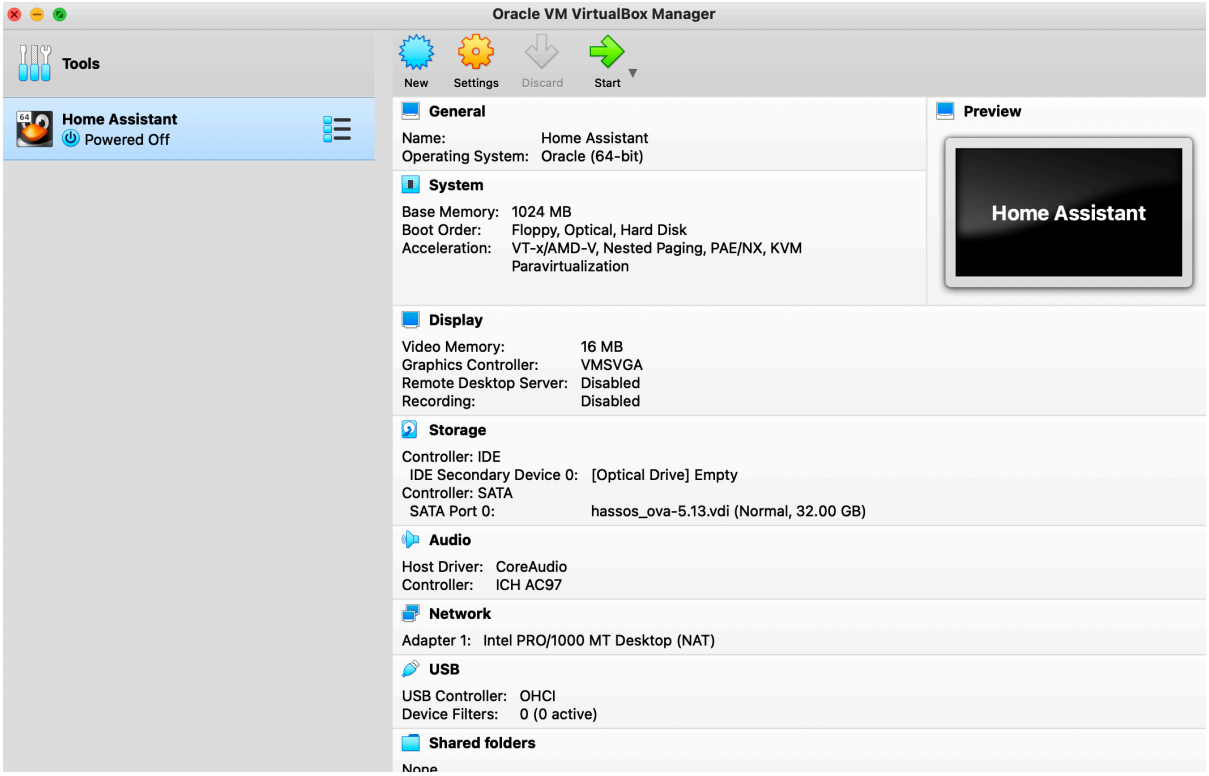
Note: go to your download folder and unpack (simply double click on the file) the **hassos_ova-5.13.vdi.xz**, this will extract the Home Assistant file that you can link to your virtual machine.

Select the "**Use an existing virtual hard disk file**" option from the list, click on the **folder** icon, and successively on the **Add** button. Select the Home Assistant **hassos_ova-5.13.vdi** extracted file and click on the **Choose** button as shown below:



One last step to create your Home Assistant virtual image is to click on the **Create** button. This will create an initial configuration of your virtual machine with the Home Assistant image installed:





Now that you have your Home Assistant virtual machine installed, it is time to build the temperature station circuit. Then, we will come back and configure Home Assistant to program with your circuit and read the data.

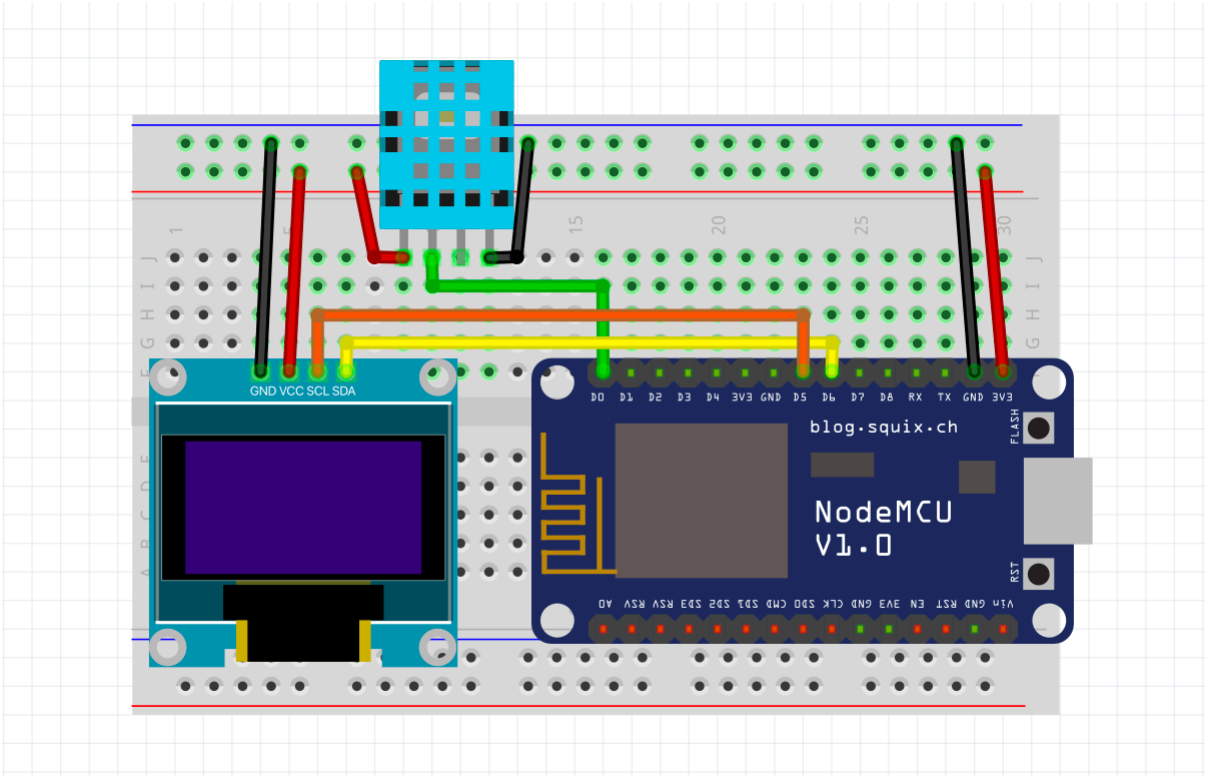
Step Two
Building the Temperature Station Circuit

The circuit assembly is pretty straight forward as it only involves the temperature/humidity sensor component and the LCD display component. The idea is to use the LCD display to show your study room temperature and then push the live data to Home Assistant via WiFi.

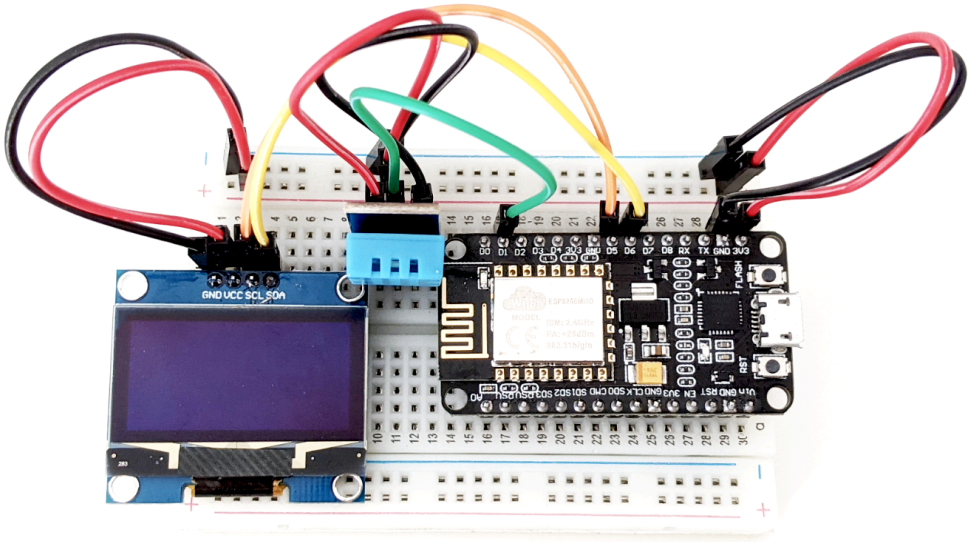
The temperature/humidity sensor: this sensor has four pins. It requires a power and a ground connection, as well as a third connection to read the temperature and humidity data. The latter refers to the pin next to the power pin of the temperature/humidity sensor which you will connect to your ESP8286 board **D1** pin.

The LCD display: this sensor has four pins. It requires a power and a ground connection, as well as two additional connections to display the temperature data. These additional connections are **SCL** (clock line) and **SDA** (data line) which are used to synchronise the data transfer. You will connect them respectively to your ESP8286 board **D5** and **D6** pins.

It is now your turn to assembly the circuit.
The diagram below shows you how the circuit should be assembled:



Furthermore, see below a picture of the circuit assembled:



At this point, you should have your temperature station circuit assembled. This means that your ESP8266 board is connected to both the LCD display component and the temperature/humidity sensor component.

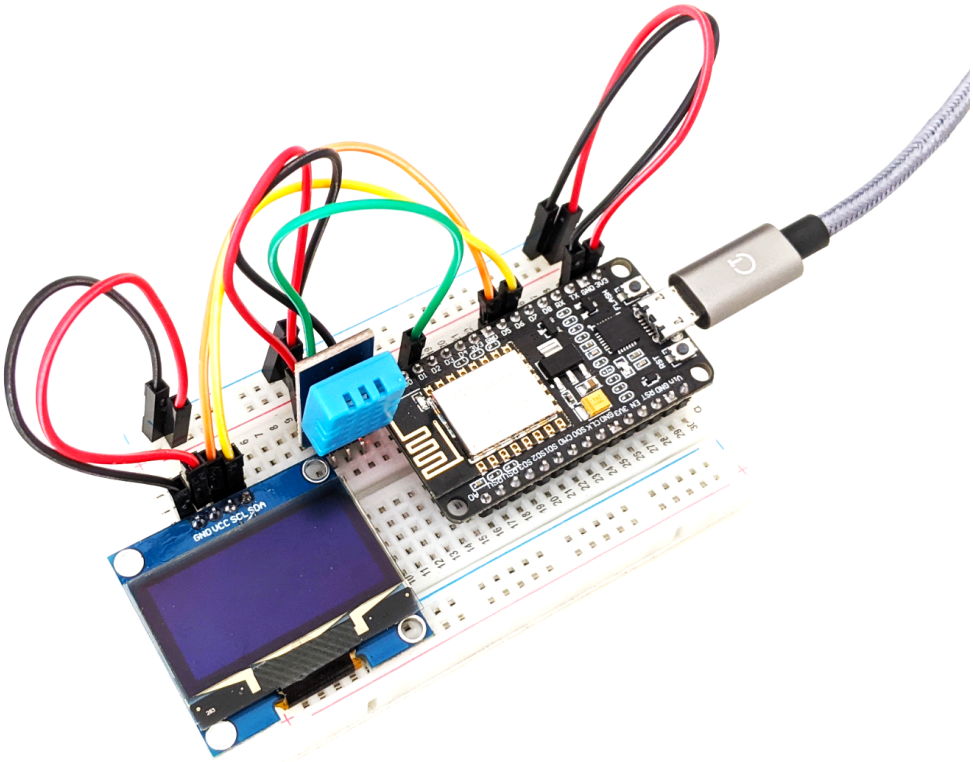
Next up is to configure Home Assistant so that you can program the circuit to read both the temperature and humidity of your room, display the temperature and humidity on the LCD screen, and push the data via WiFi to Home Assistant .

Step Three
Configuring Home Assistant

Now that the temperature station circuit is assembled, you are ready to program it to read the data, display the data, and push the data to Home Assistant.

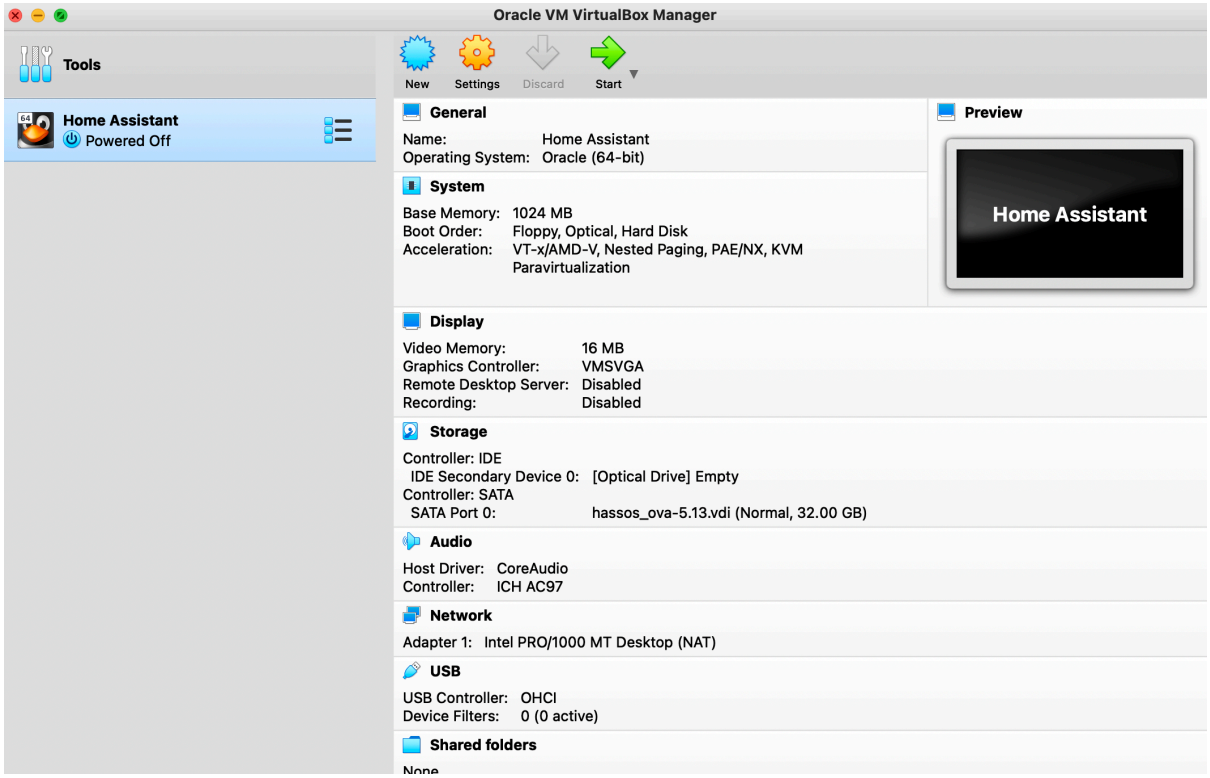
In order to do so, we will first have to configure and run the virtual machine with the Home Assistant interface.

First thing first, double check your circuit assembly and connect the ESP8266 board to your computer with the micro USB cable:

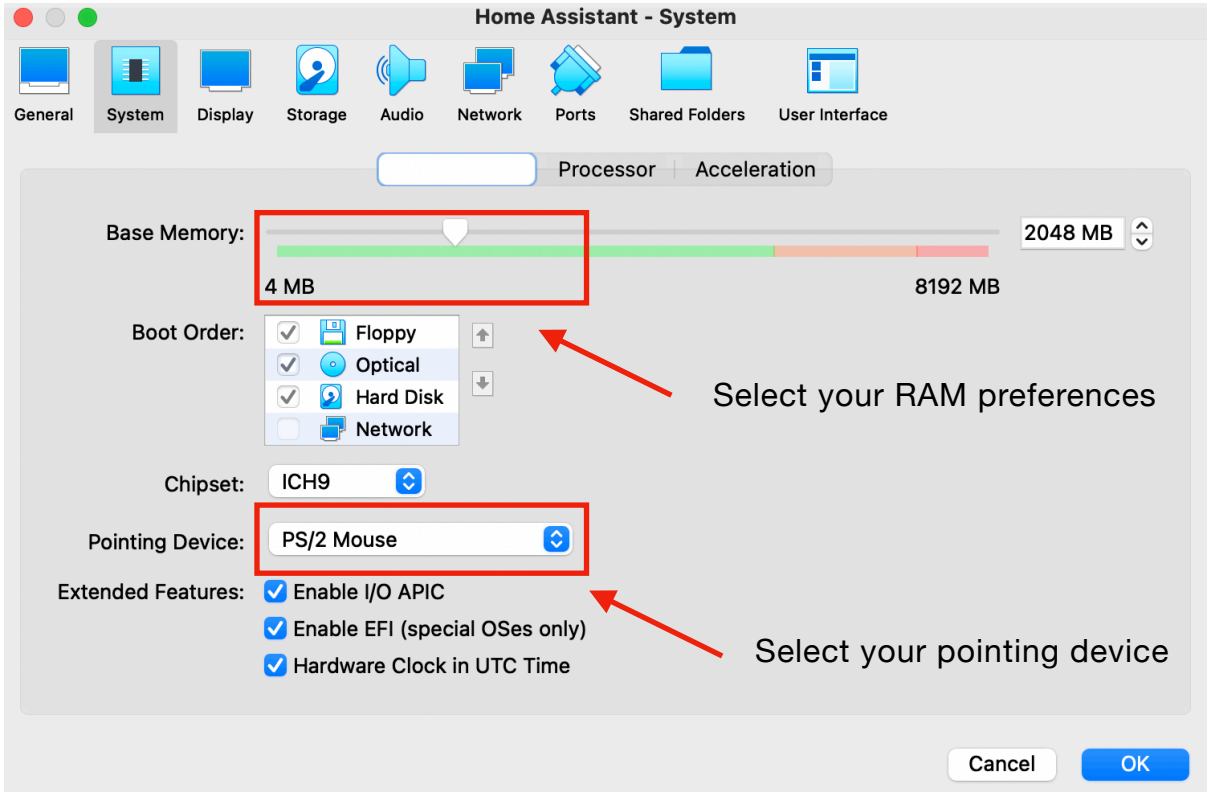


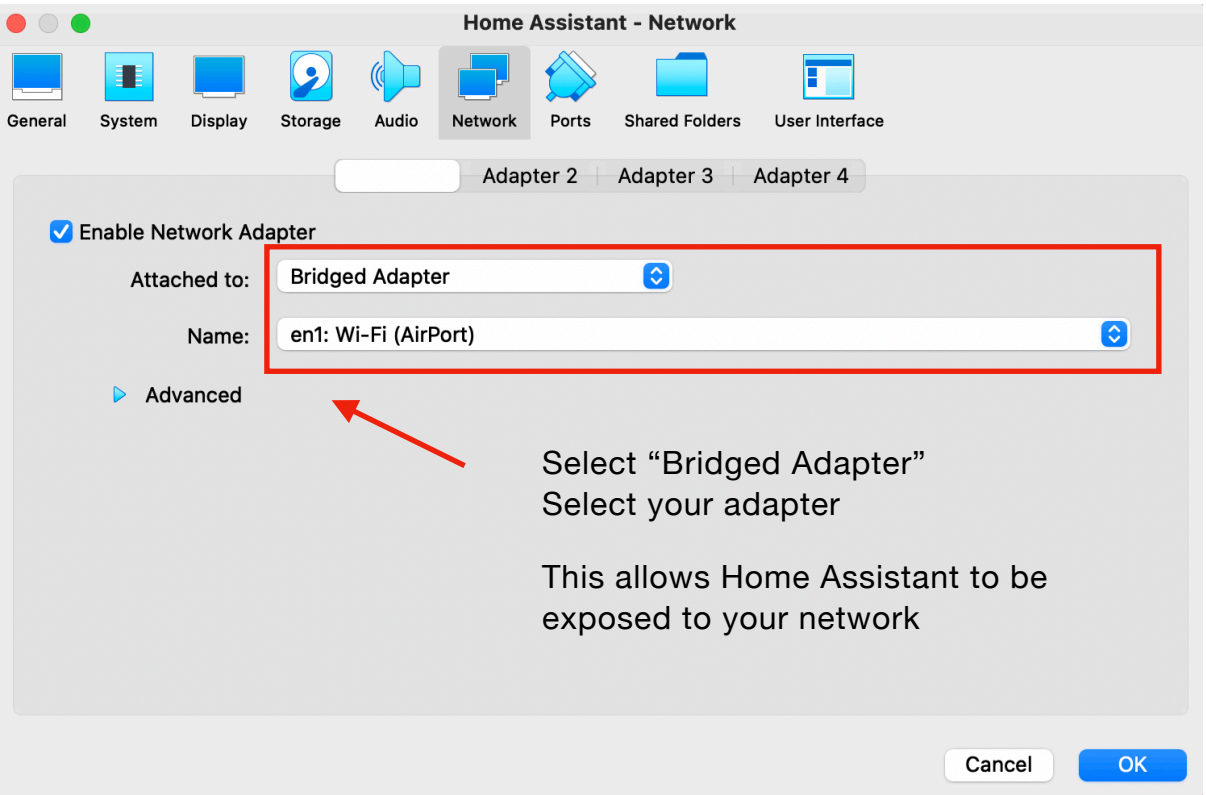
Let’s configure VirtualBox now so that you can access Home Assistant.

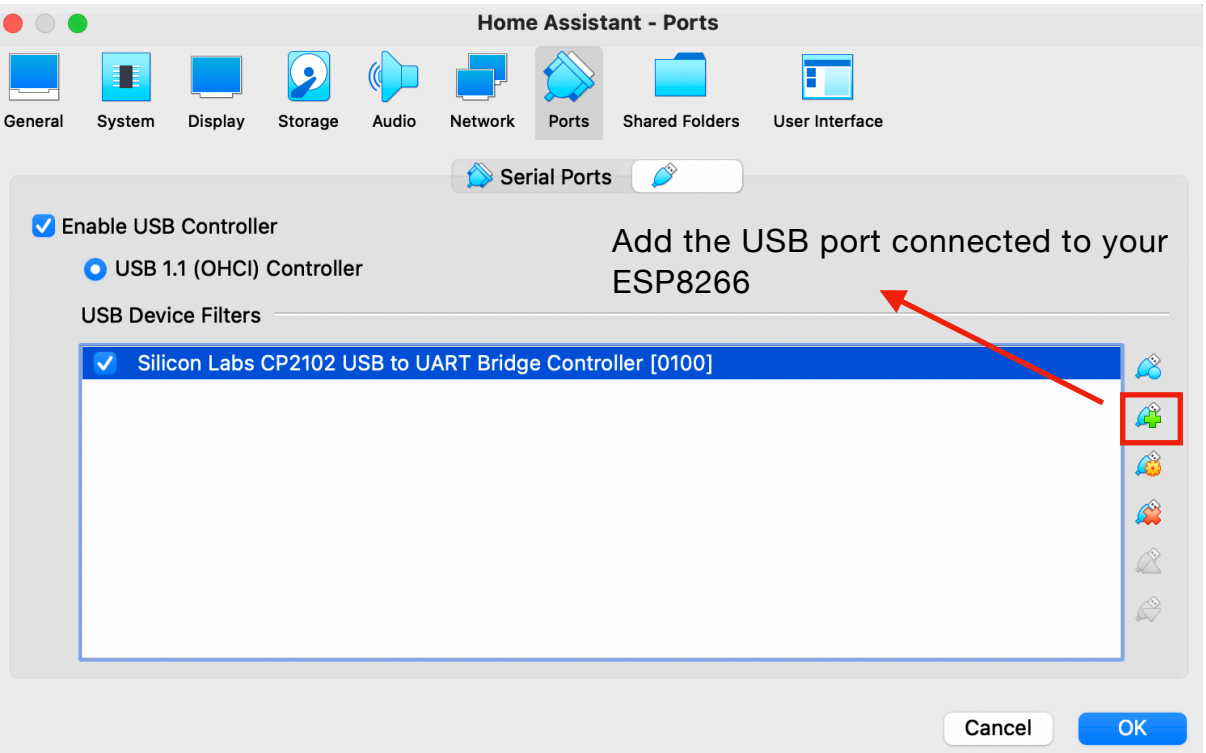
Open the VitualBox application, you should see the Home Assistant virtual machine that you created previously:



Click on the **Settings** button located on the main navigation bar and configure the Home Assistant virtual machine with the following options:

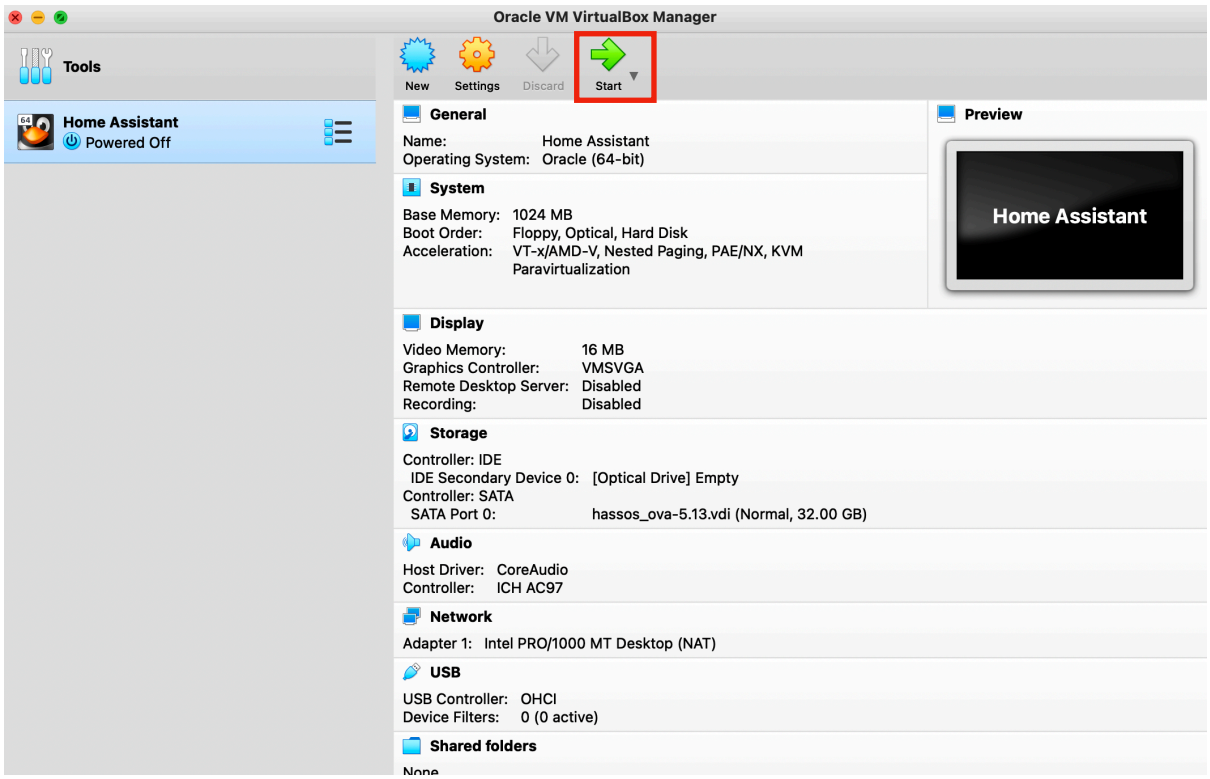




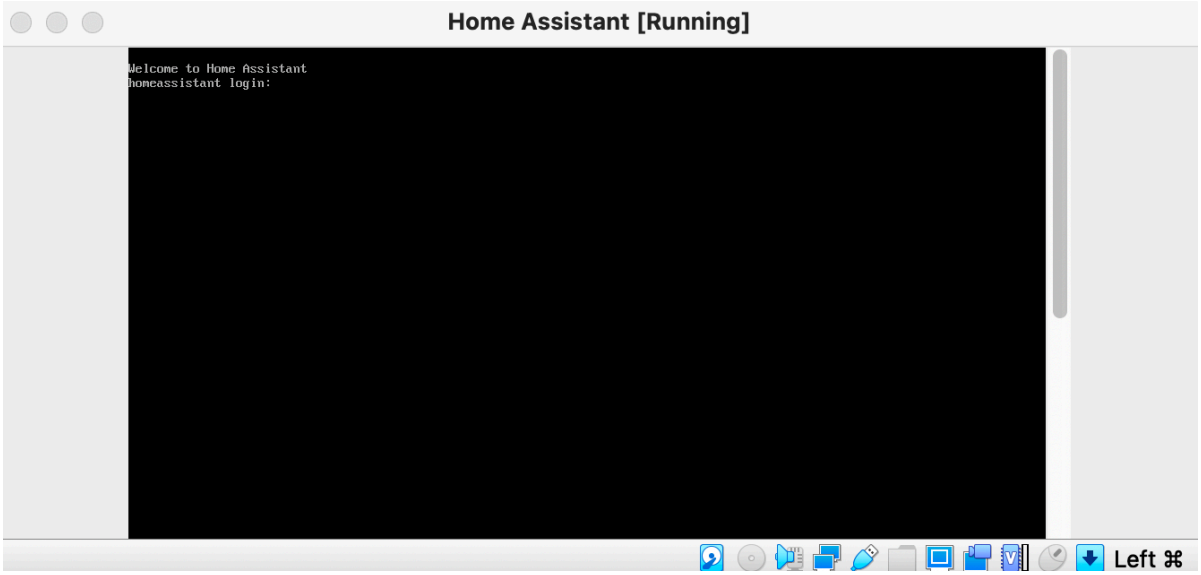


At this point, the VirtualBox configuration is ready and you can start the Home Assistant virtual machine.

Click on the **Start** button to launch the virtual machine:



This will start the virtual machine and run Home Assistant:

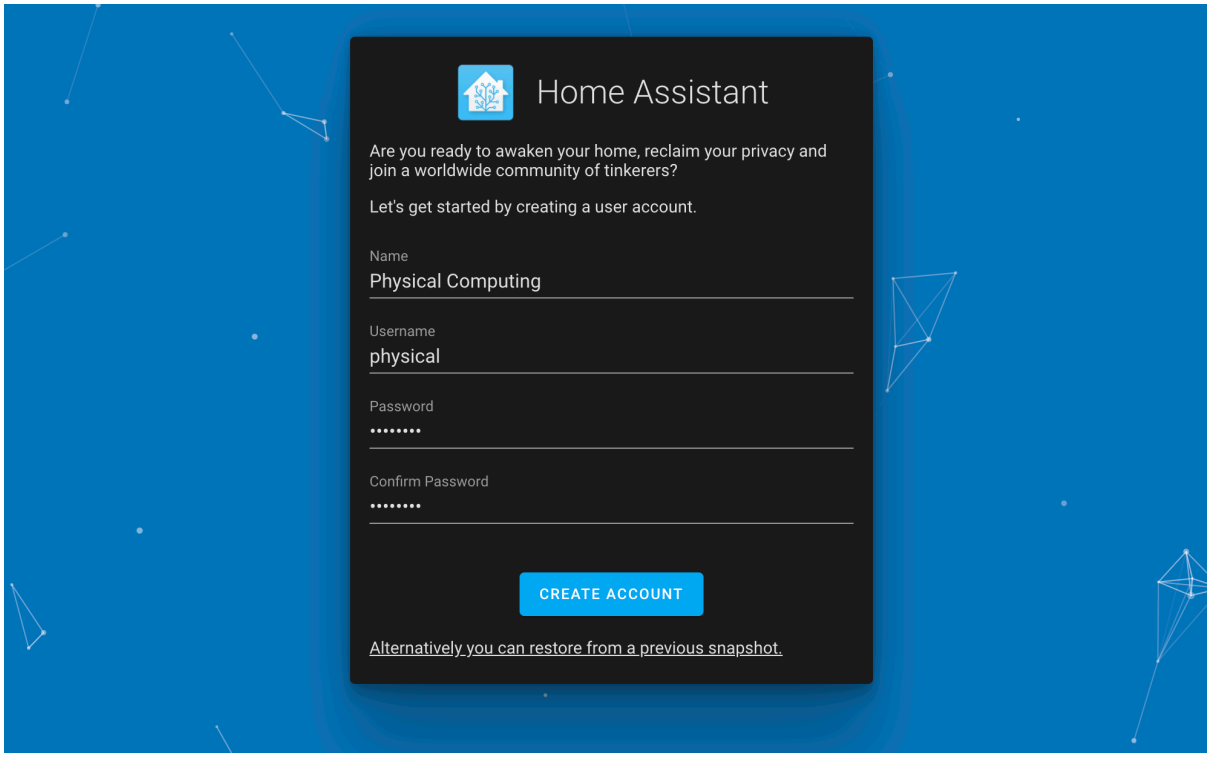


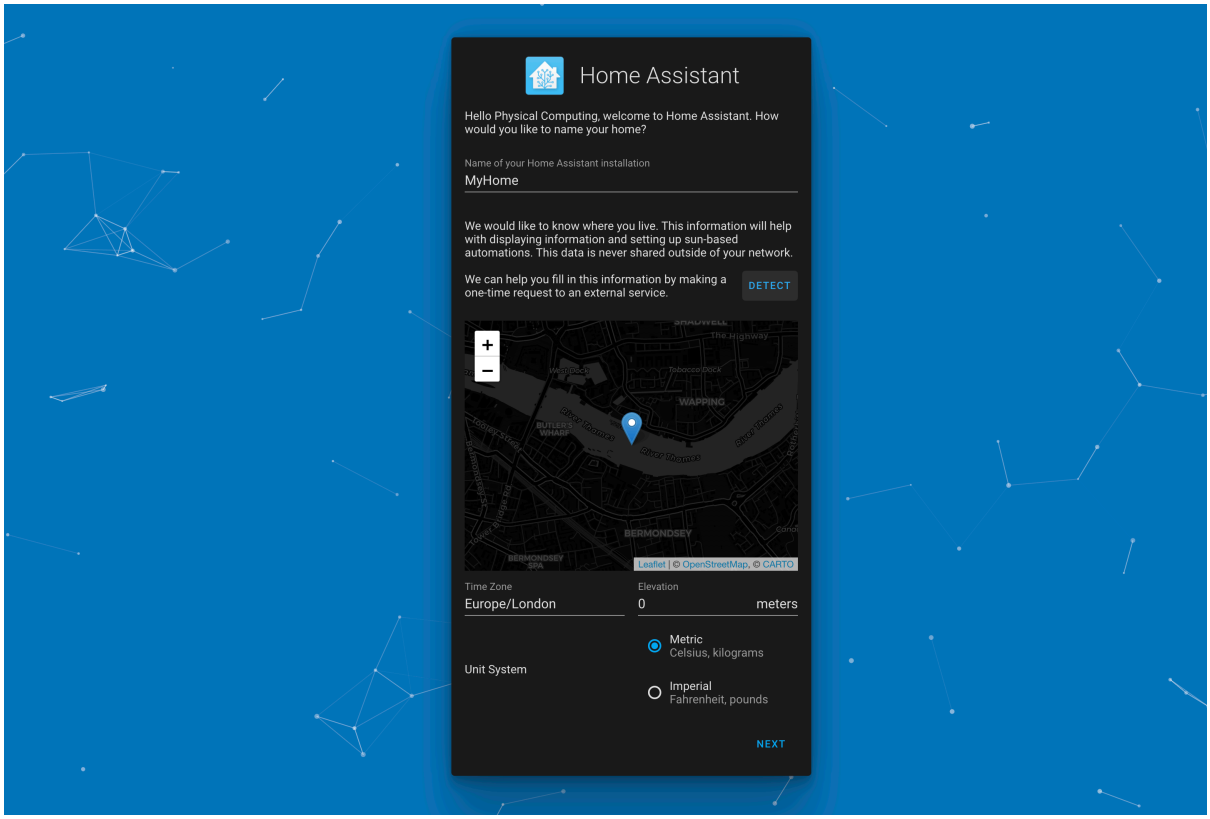
Open your favourite browser (Chrome recommended) and access Home Assistant at the following URL:

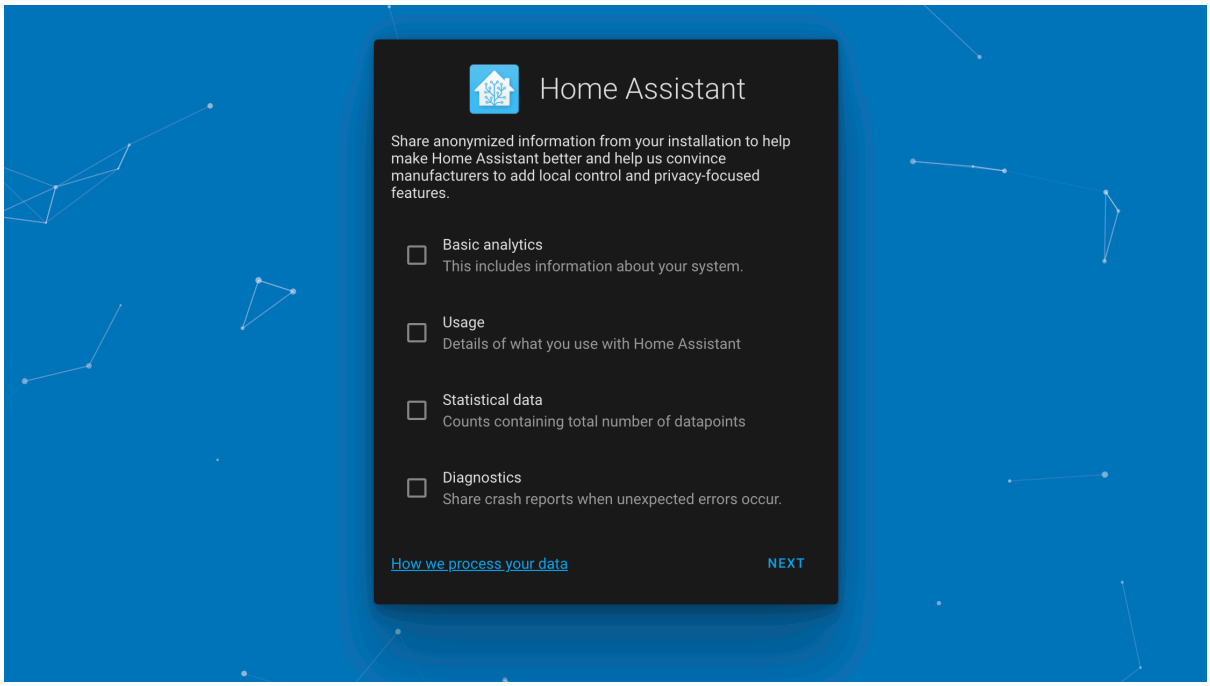
http://homeassistant.local:8123/

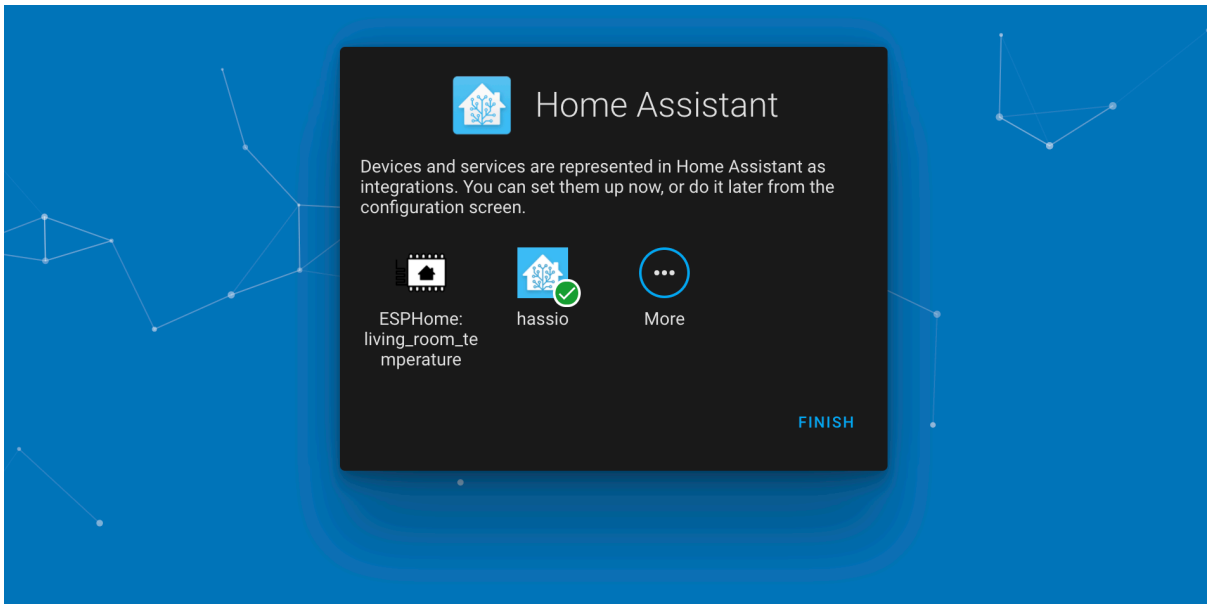
The above link will redirect you to the onboarding page of Home Assistant.

You are almost there, just follow the onboarding instructions:







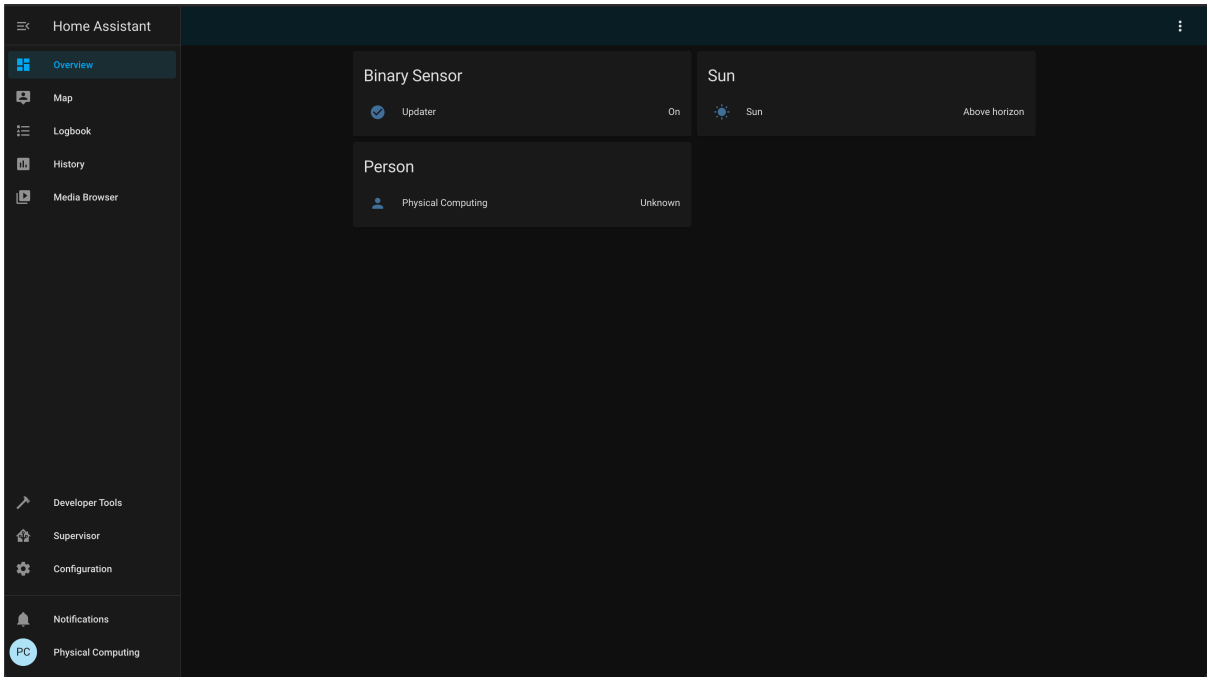


Click on the **FINISH** button at the end of the onboarding instructions and this should redirect you to the Home Assistant dashboard.

Congratulations for making it this far, next up is to actually create the temperature and humidity detection code and upload it to the ESP8266 board. You will do this with the help of ESPHome, a Home Assistant package which facilitates the communication with your ESP8266 board.

Step Four
Installing the ESPHome Add-on

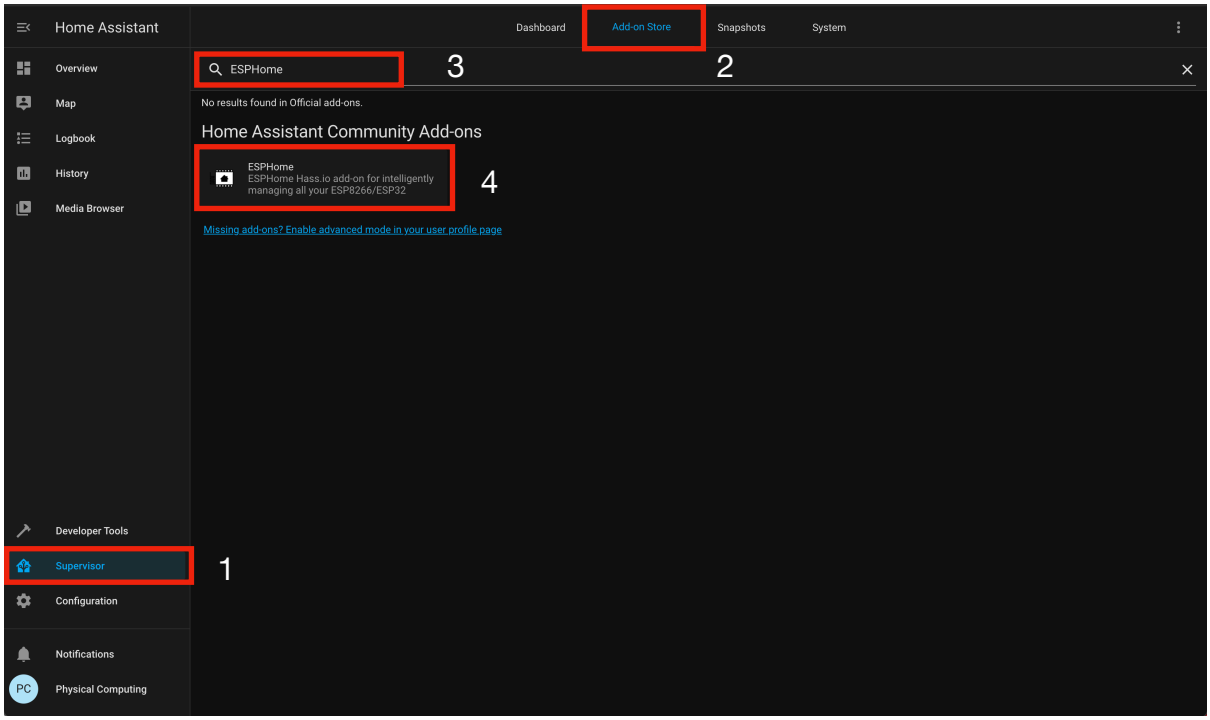
Once you have configured Home Assistant and completed the onboarding process, accessing **http://homeassistant.local:8123/** should direct you to the Home Assistant dashboard. On the dashboard, you will already have some data displayed, e.g. the weather card. These information were taken from the onboarding process data (location) that you input earlier:



What you want to do now is to add another card which displays the temperature and humidity of your study room. In order to do so we first need to install the ESPHome add-on.

The ESPHome is a system that allows you to control your ESP8266 board by simple, yet powerful, configuration files. These files are written in YAML, a human-readable data-serialisation language.

In order to install ESPHome click first on the **Supervisor** tab (1) and then on the **Add-on Store** tab (2). Now search for **ESPHome** (3) and click on the add-on (4):

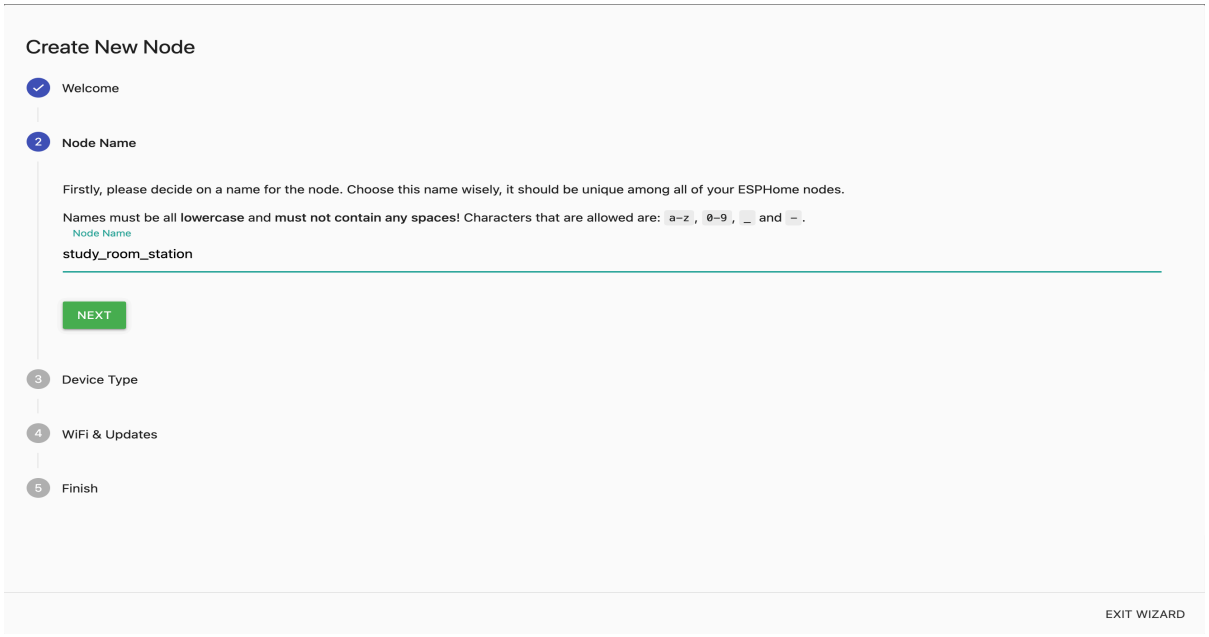


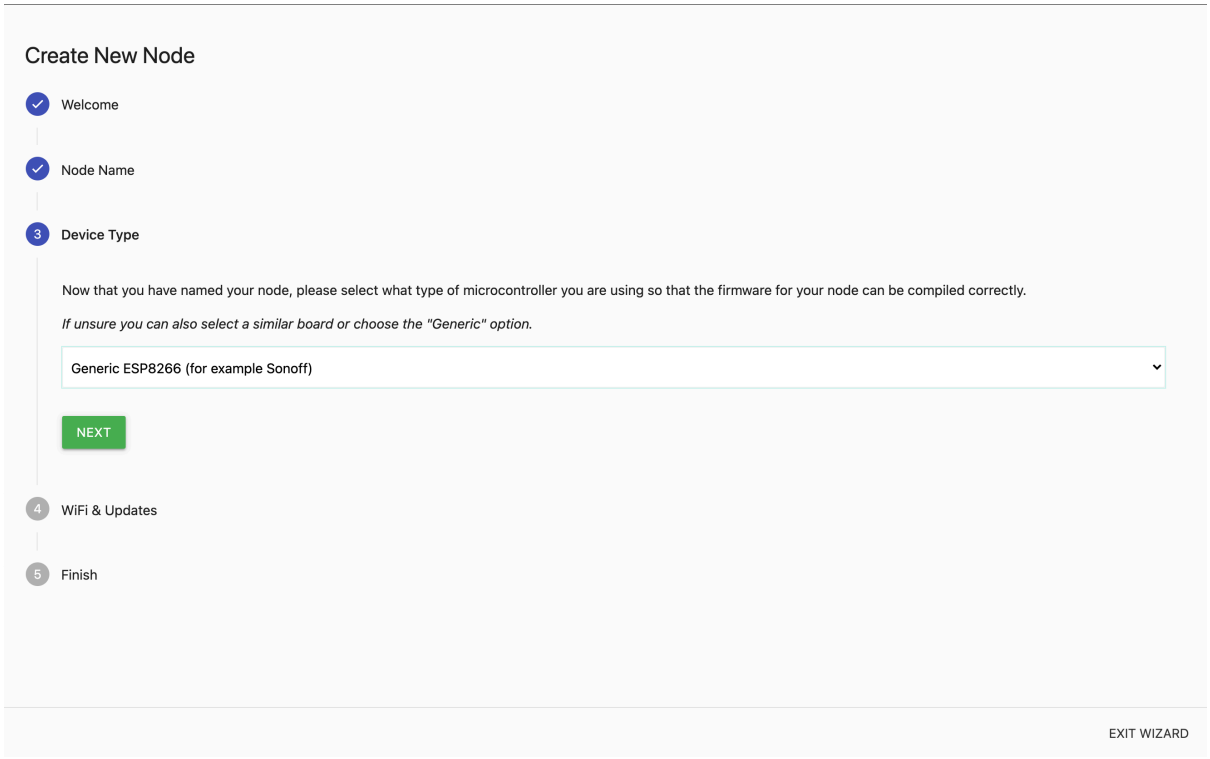
Click on **install,** thick all the options (Start on boot, Watchdog, and Show in sidebar), and click on **start**:

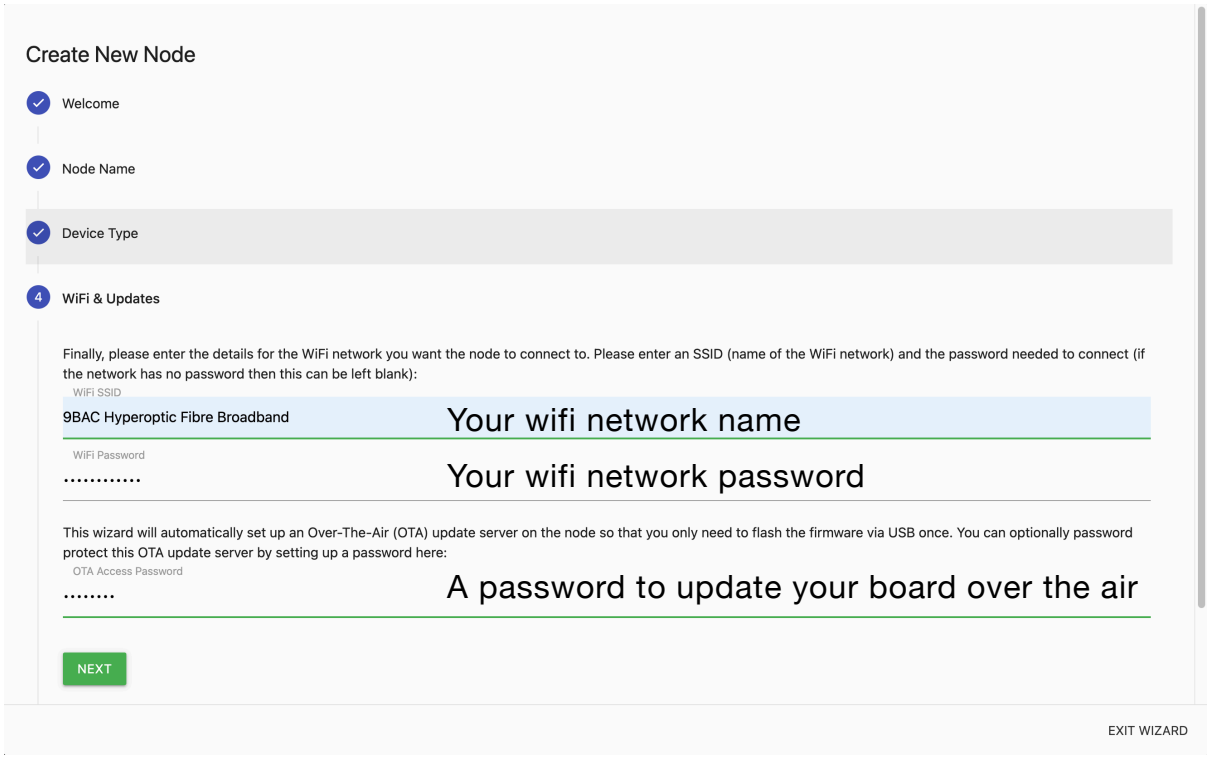
Step Five
Your first ESPHome node

Now that you have installed ESPHome, you can access it from the Home Assistant sidebar. Click on the **ESPHome** tab (1) and click on the **+** button (2) to add a new node. This will allow you to configure your first node (connection point):

Click on the **Begin** button and configure as shown below:

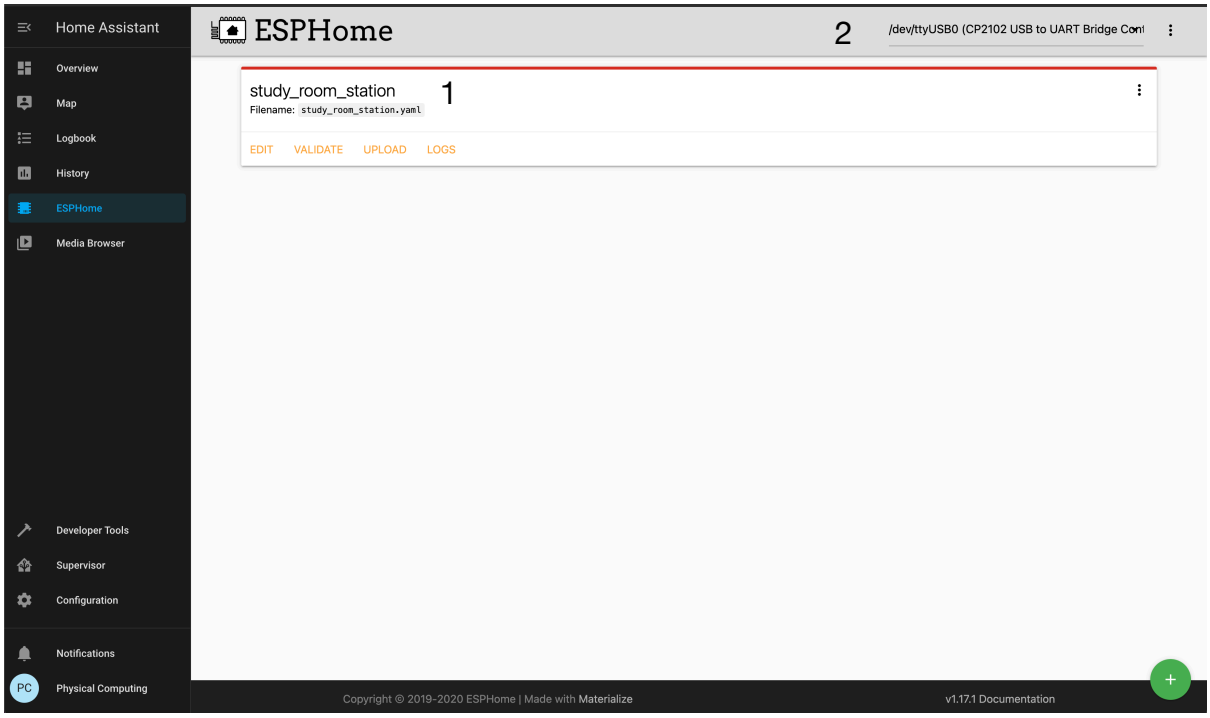






Finally, click on the **Submit** button to create your first node.

You can now see you first node inside the ESPHome dashboard with the name of study_room_station (1). Change the upload method to be via USB by selecting your USB port connected to the ESP8266 board from the dropdown menu (2). The first time you upload a firmware to a board it has to be via USB, then you can do it over the air for subsequent uploads:



Before you click on the **Upload** button, let's check the content of the study_room_station.yaml file. This file contains the firmware to enable the communication with the ESP8266 board on your wifi network. Click on the **Edit** button to view the file:

Editing: `study_room_station.yaml`

```
1  esphome:
2    name: study_room_station
3    platform: ESP8266
4    board: esp01_1m
5
6  wifi:
7    ssid: "[redacted]"
8    password: [redacted]
9
10   # Enable fallback hotspot (captive portal) in case wifi connection fails
11   ap:
12     ssid: "Study Room Station"
13     password: [redacted]
14
15 captive_portal:
16
17 # Enable logging
18 logger:
19
20 # Enable Home Assistant API
21 api:
22   password: [redacted]
23
24 ota:
25   password: [redacted]
```

Your node and board information

Your wifi information

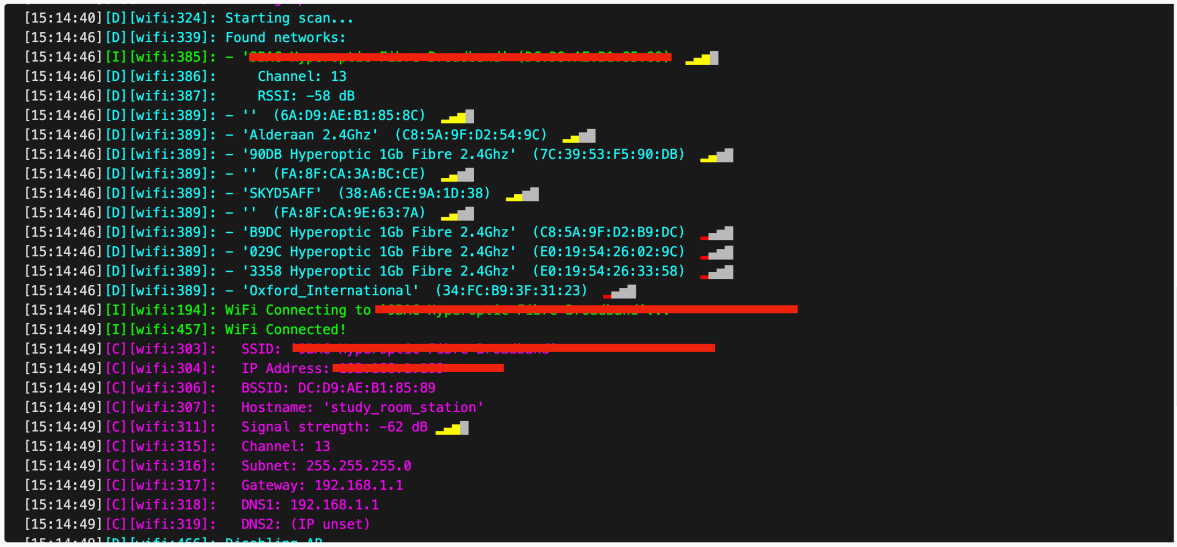
Fall back information if no wifi

Logger enabled for debugging

Your Home Assistant and Over the Air passwords

Click on the **Upload** button located at the bottom of the YAML file window to upload the firmware to your board. You can view the logs and verify that the upload was successful and that the board is connected to your wifi network:

Compiling & Uploading: `study_room_station.yaml`



The console allows you to see the logs of your ESP8266 board and debug any issues that you might have during uploads.

Now that the connection is established, it is time to write the code to read the temperature and humidity data of your study room from the DHT11 sensor.

You will have to edit the “study_room_station.yaml” file with additional code to do so. Let’s move to the next section and more code to read the data.

Step Six
The Temperature and Humidity Firmware

Open the study_room_station.yaml file and add the following code at the end of the document (make sure the code is indented properly):

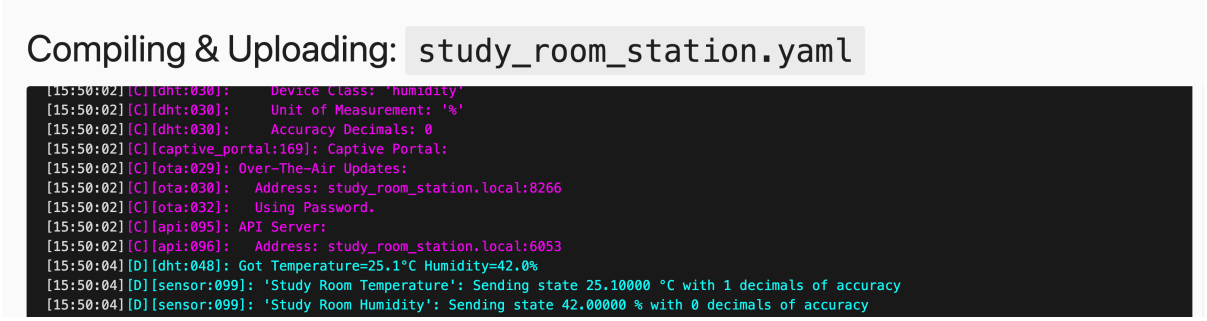
```
# Configuration entry for the temperature/humidity sensor:
sensor:
  - platform: dht
    pin: 5
    model: DHT11
    update_interval: 5s
    temperature:
      id: tmp
      name: "Study Room Temperature"
    humidity:
      id: hum
      name: "Study Room Humidity"
```

The above snippet of code configures the reading of the temperature and humidity from the DHT11 sensor. You can check more information about the DHT11 configuration here: https://esphome.io/components/sensor/dht.html

Pin: 5 is the equivalent of the D1 pin on the ESP8266 board, the pin to which the DHT11 data pin is connected to.

Update the code once again. You can now do this either via USB or Over the Air by selecting the appropriate option on the dropdown menu.

If the upload was successful, you should able to see live temperature and humidity data from the logs:

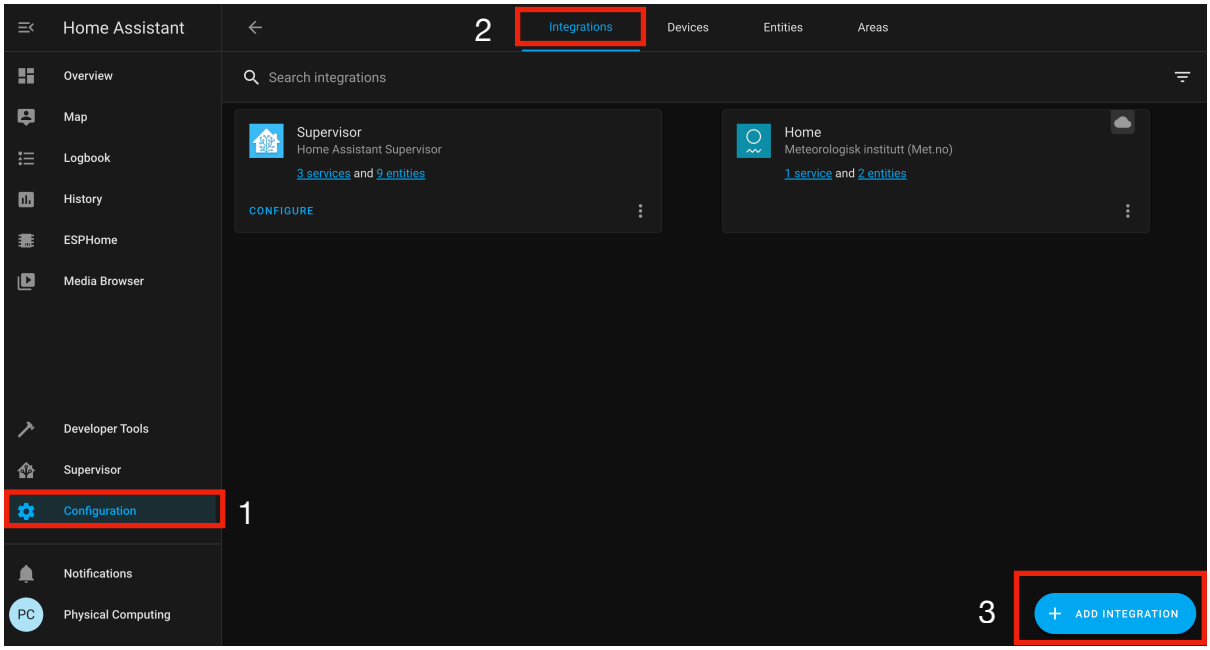


Hurray! It is now time to display the data on the Home Assistant dashboard.

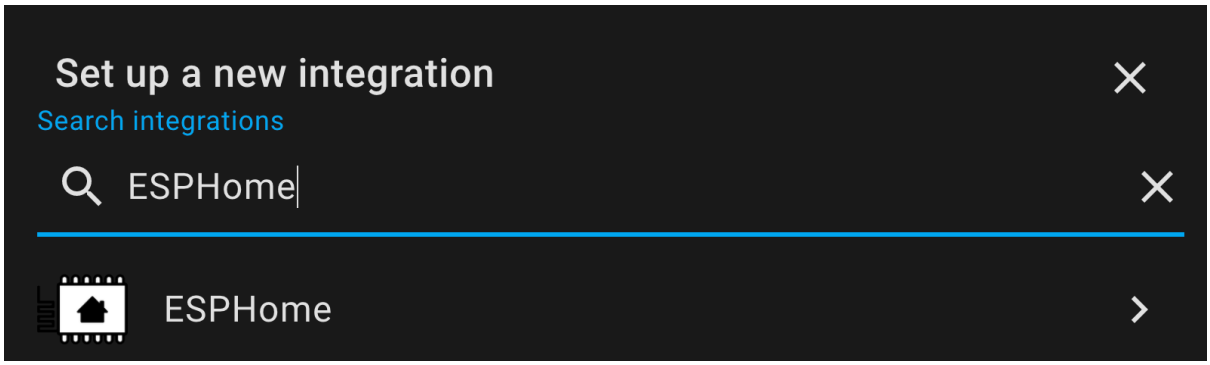
Step Seven
The Home Assistant Dashboard

Now that you have live temperature and humidity data coming into Home Assistant, you can use built-in GUI cards to display your data on the dashboard.

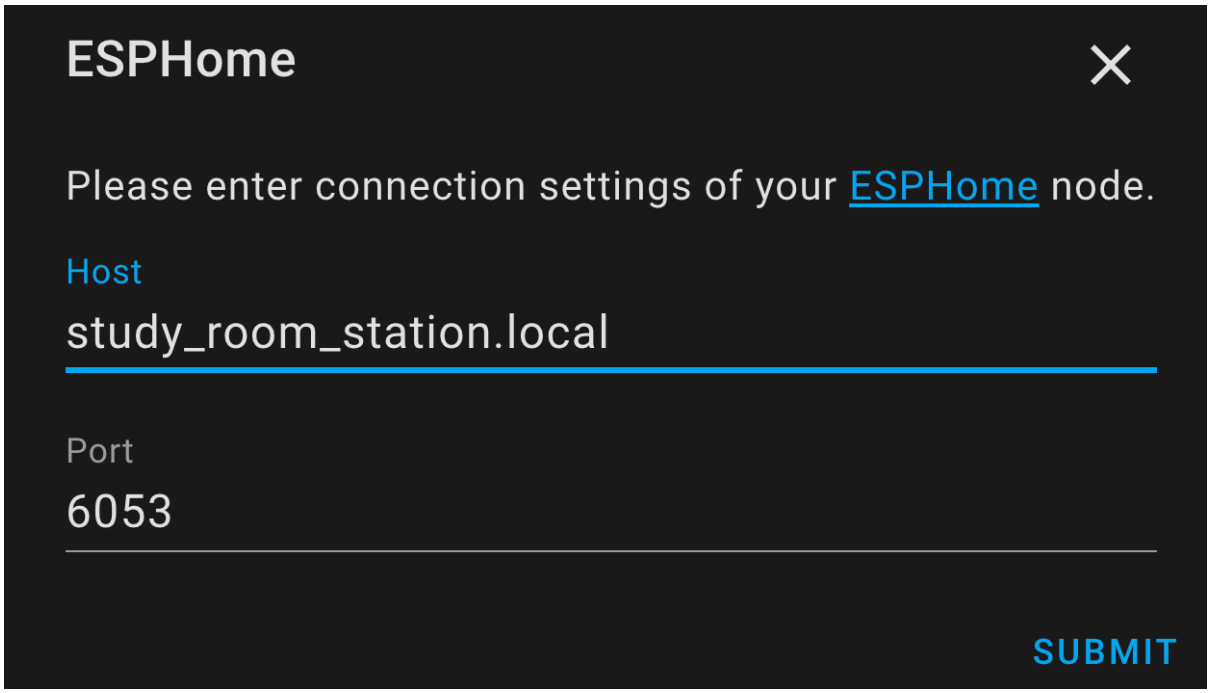
Click on the **Configuration** tab first (1), then on the **Integration** tab (2), and finally click on the **+ Add Integration** button to create your UI integration (3):



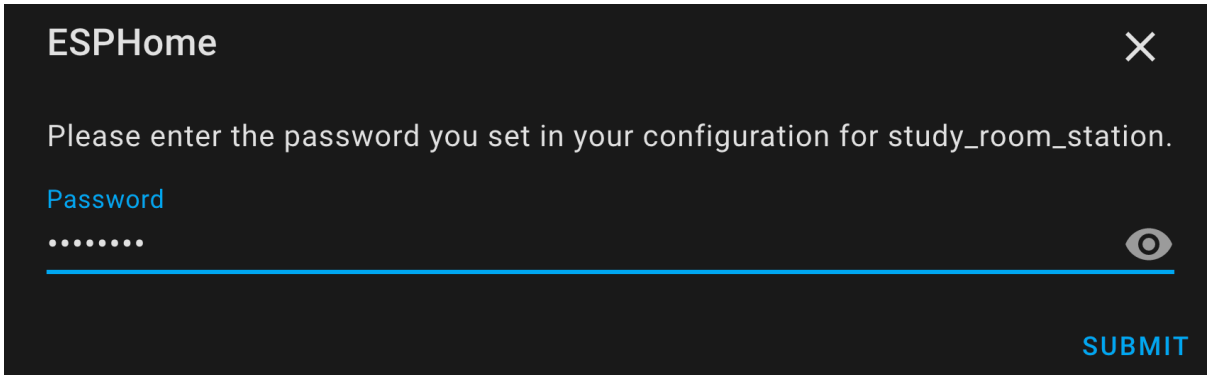
Search for **ESPHome** and select the integration:



Insert the name of your node and press **submit** to start the integration:
Note: add .local at the end of your node name and leave Port as suggested

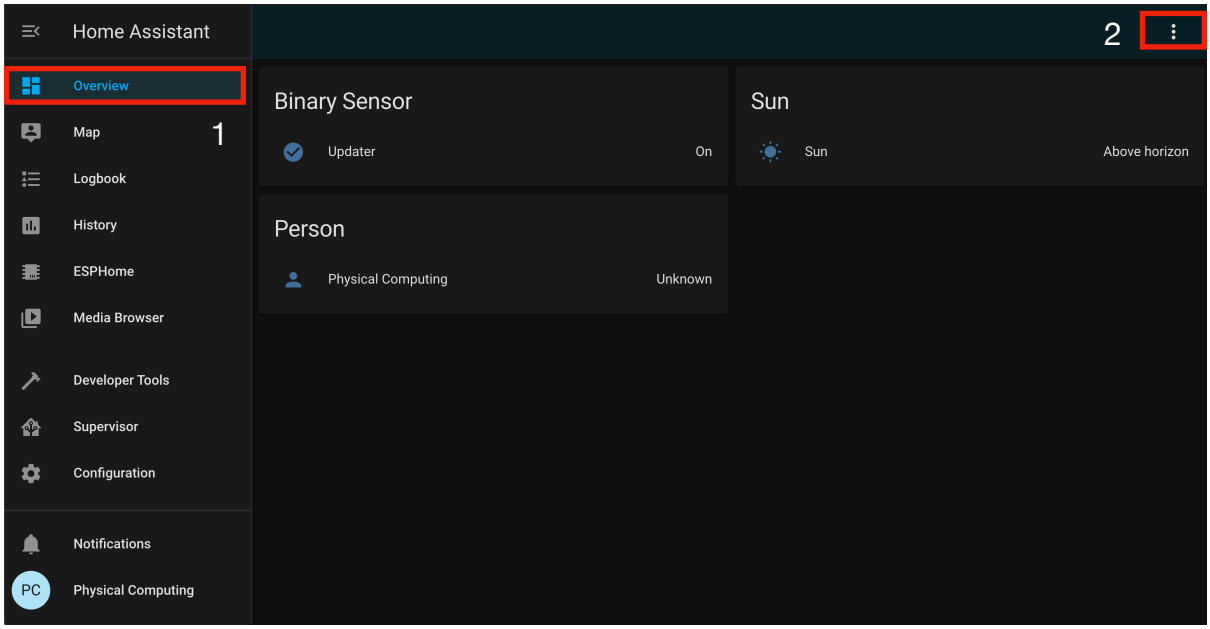


Enter the password that you configured for the study_room_station node and press **submit**:

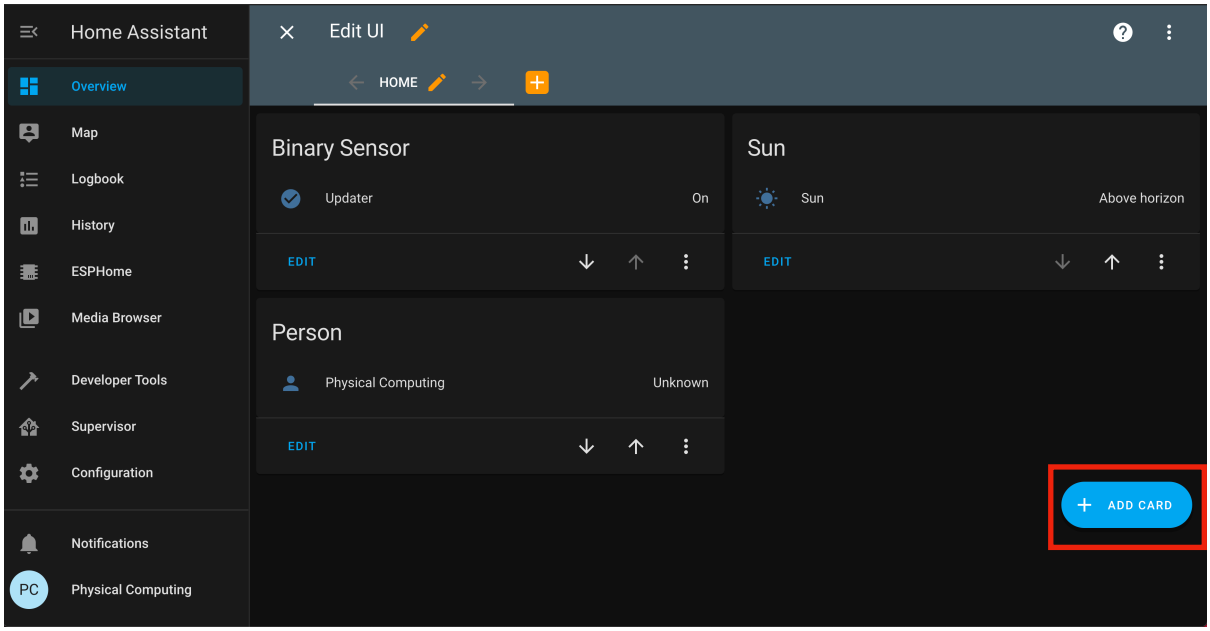


Finally, click on **Finish** to confirm the integration.

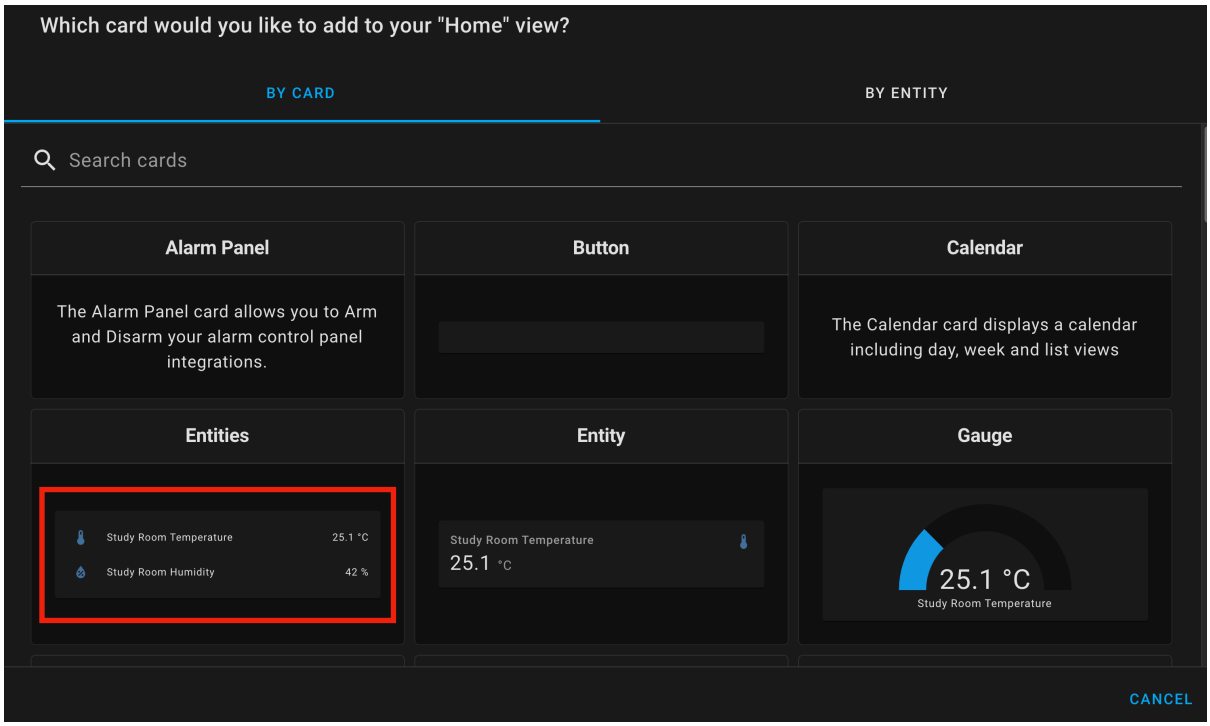
Head to the Home Assistant **dashboard** (1) and click on the **three dots** to add a card (2):



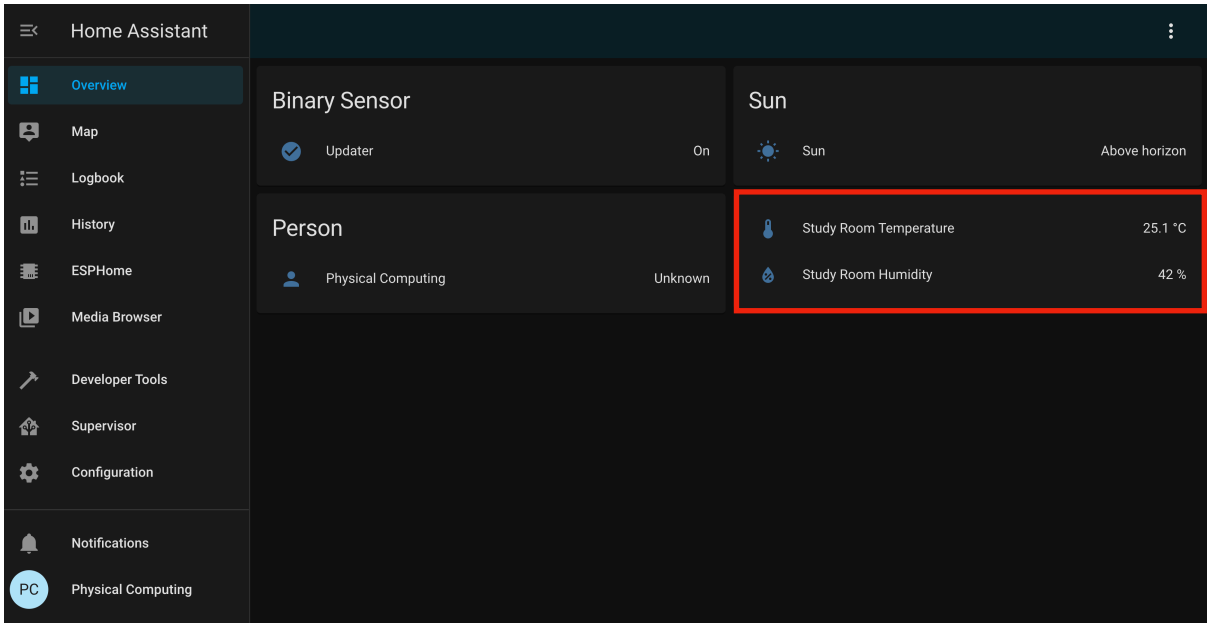
Click on the **+ Add Card** button to add a new card:



Select the card that contains both the temperature and the humidity data and click on the **Save** button:



Now, you can view your card on the Home Assistant dashboard:



Congratulations for setting up your first node and integration with Home Assistant. You can now monitor the temperature and humidity of your room via wifi and have your data displayed on the Home Assistant dashboard.

Extension Task
The LCD Display Integration

You have seen how easy it is to create an ESP integration with Home Assistant and the possibilities on creating your smart environment are endless here.
Let's explore one more functionality for your temperature station circuit.

Would it be handy to also have the temperature and humidity data showed on the LCD display component? Guess what, you can edit the YAML configuration file to accomplish this result.

Add the following code to the study_room_station.yaml file:

```
# Configuration entry for the temperature/humidity sensor:
i2c:
  sda: 12
  scl: 14

font:
  - file: "font.ttf"
    id: my_font
    size: 20

display:
  - platform: ssd1306_i2c
    model: "SH1106_128x64"
    reset_pin: 16
    address: 0x3C
    lambda: |-
      it.printf(0, 0, id(my_font), "Temp : %.1fc°", id(tmp).state);
      it.printf(0, 32, id(my_font), "Hum : %.1f", id(hum).state);
```

The above snippet of code configures the LCD display component to render the temperature and humidity values. You can check more information about the component setup here: https://esphome.io/components/display/ssd1306.html

Pin: 12 is the equivalent of the D6 pin on the ESP8266 board, the pin to which the display SDA data pin is connected to.

Pin: 14 is the equivalent of the D5 pin on the ESP8266 board, the pin to which the display SCL data pin is connected to.

Make sure that both the **model** and the **address** values are correct for your display component. You can find the model on the component data sheet and the address on the logs during upload.

The **lambda** key is used to print information on the LCD display:
printf(display matrix row, display matrix column, font id, text, reference variable)

Almost there, there is one more step before you can upload this new firmware to the ESP8266 board. Unfortunately, the ssd1306_i2c display component requires a valid font (.ttf file) to render text on the screen and there is no default or included fonts on the ESPHome integration.

The following block of code attempts to load a font called “font.ttf”:

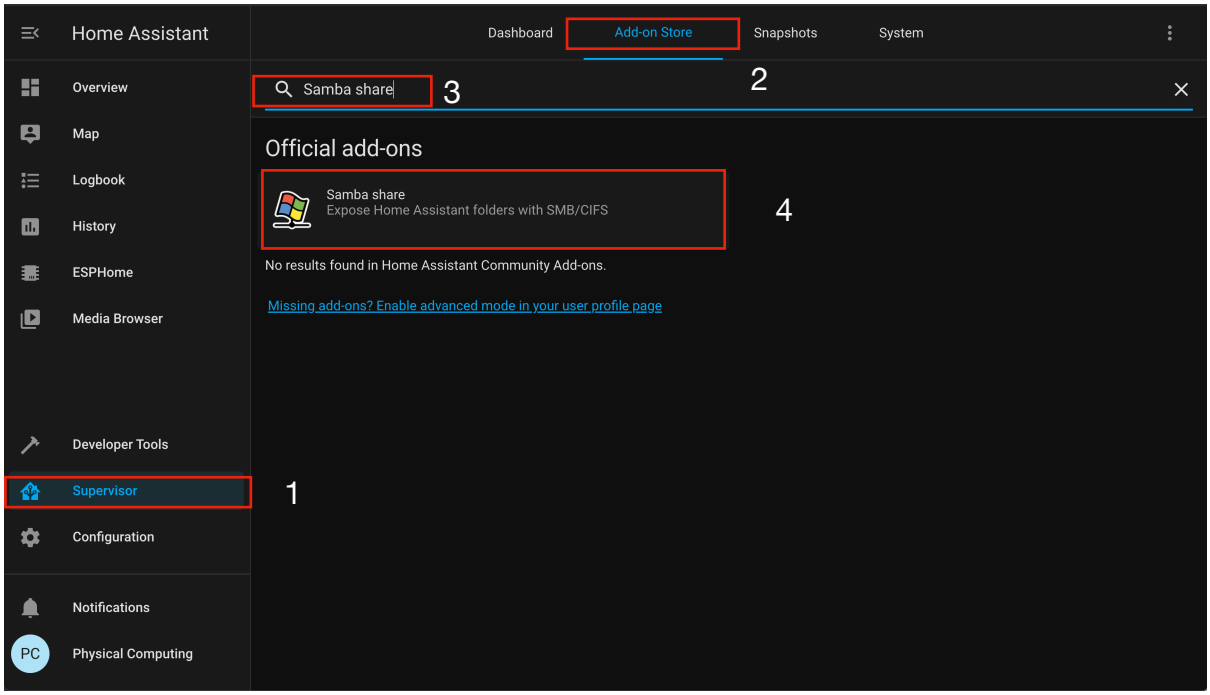
```
font:
  - file: "font.ttf"
    id: my_font
    size: 20
```

The file path is relative to where the .yaml file is located. In this case, the font is searched in the same directory as the .yaml file which is located at: **/config/esphome**

In order to access such font, you first need to place your favourite TrueType font file (.ttf) inside **/config/esphome** with the name of **font.ttf**.

Since you are running Home Assistant through a virtual machine, the Home Assistant file system is not available from your local computer. A quick workaround is to install another add-on which will enable you to navigate the file system of Home Assistant and place the font file in the desired directory.

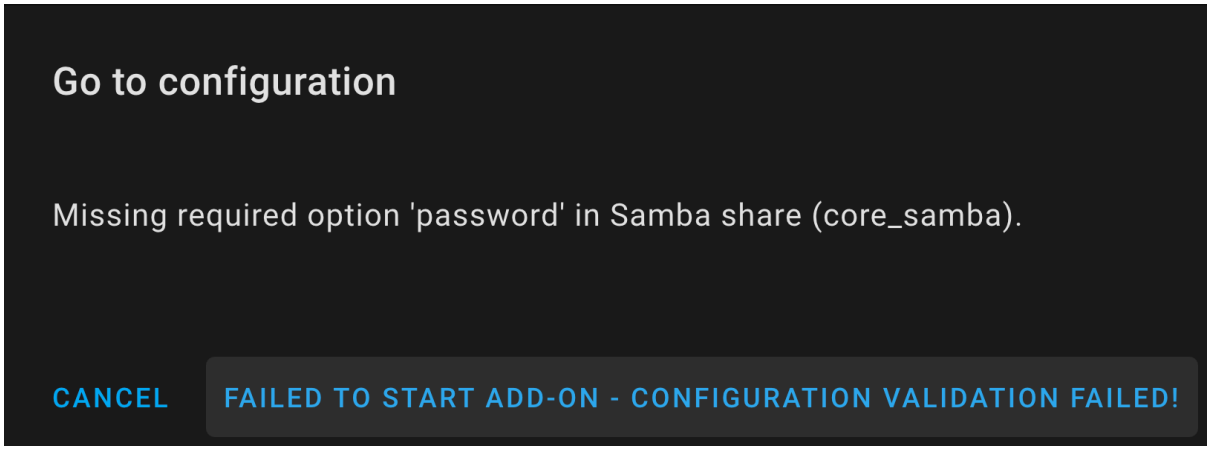
Click on the **Supervisor** tab (1), then on the **Add-on Store** tab (2), search for the **Samba share** extension (3), and **click** on the extension (4):



One you select the extension, click on the **install** button.

Enable both the **Start on boot** and the **Watchdog** (1) options and click on **Start** (2):

The first time you run this extension, you will get an error:

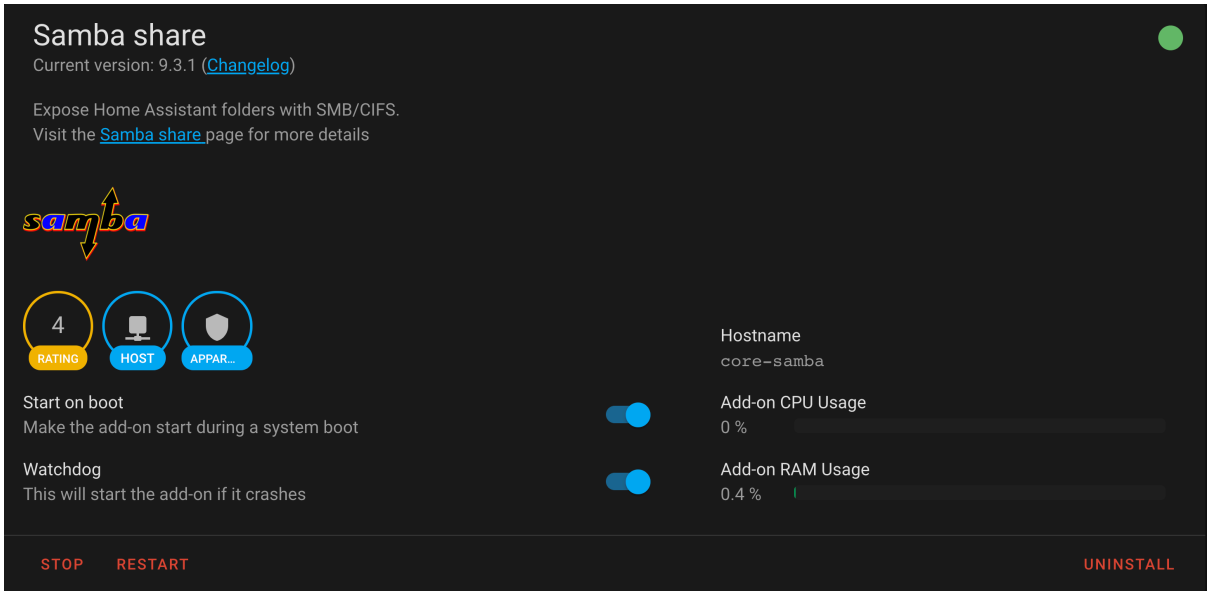


This is because you need to setup a password in the Samba share configuration file. Simply click on the **Failed to start Add-on** button, this will open the configuration file for you.

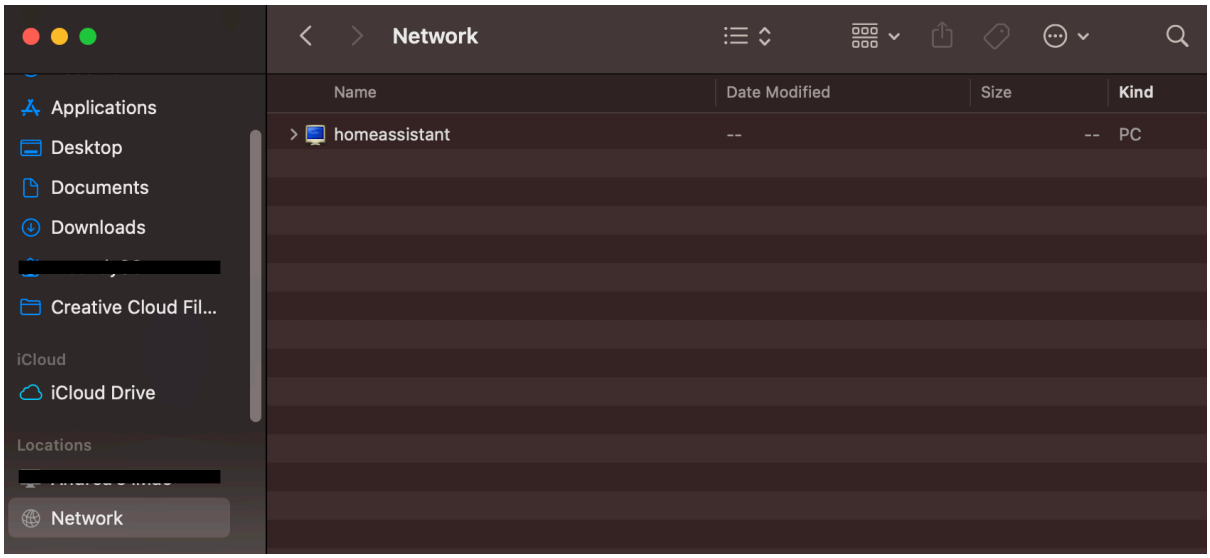
Edit the configuration file by adding a password and click on the **Save** button.

Now, restart the Samba share extension.

The Samba share integration should now be running as show below:



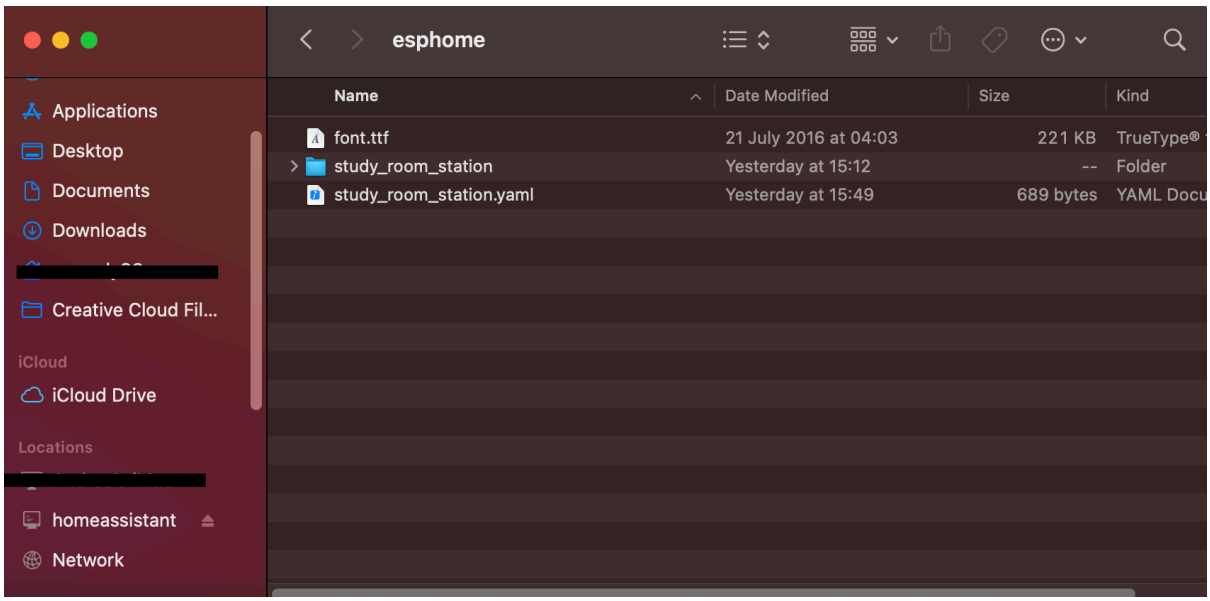
The integration will allow you to access the Home Assistant file system from your local computer Network section which, on MacOs, can be identified in the finder application:



Windows and Linux users can access the Network drive respectively in either the File Explorer or the File Manager.

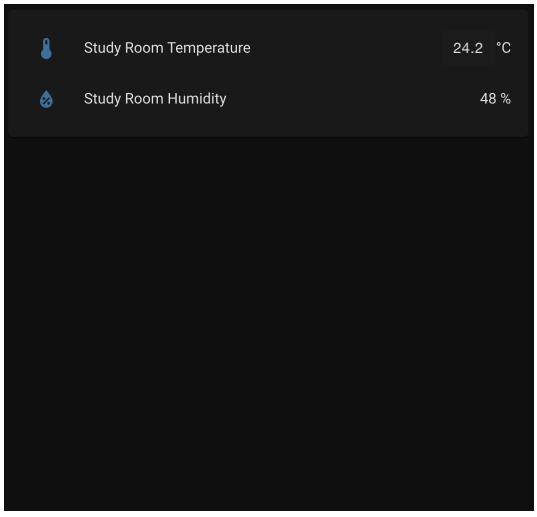
Click on the **homeassistant** network, enter the password that you configured inside the Samba share configuration file, and navigate to the desired location to insert the .ttf file: **/config/esphome**.

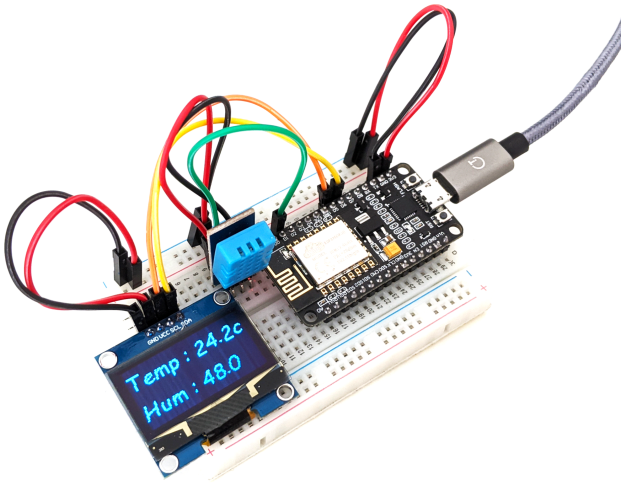
At this point, simply add your favourite font file as shown below:



Excellent, you are now ready to upload the latest firmware to your ESP8266 board. Head back to the Home Assistant ESPHome tab and upload the firmware to your board.

You should now be able to see your temperature and humidity data both on the Home Assistant dashboard and on the LCD display component:





Congratulations for completing this activity. You can now enjoy your personal temperature and humidity data station and monitor live data from the Home Assistant dashboard.

You are probably a little overwhelmed by the amount of new information encountered in this activity but I know that you might already be thinking about the next ESPHome integration. Check the full documentation and have fun:

https://esphome.io/