EchoPiBox 2.0

Detect.Classify.Exhibit

BUILD YOUR OWN

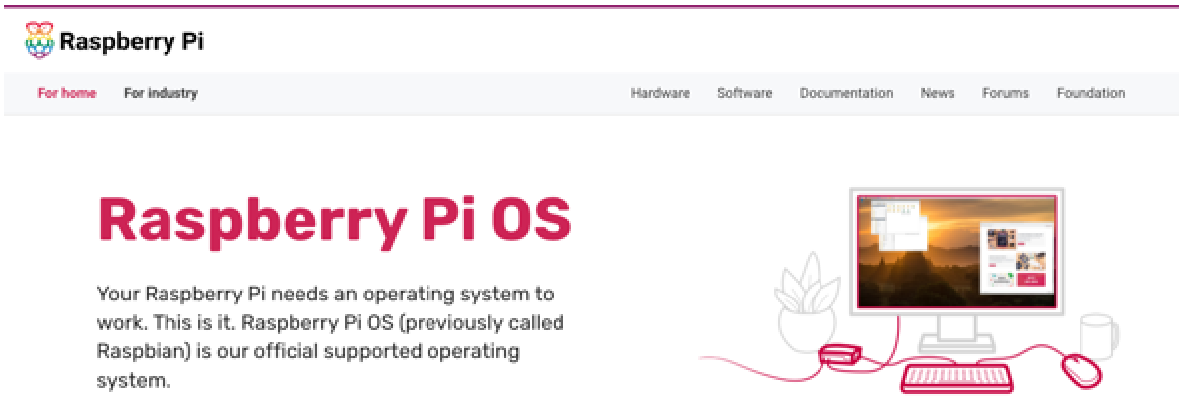
Detailed Instructions *(for IOS users only)*

Margarita Smoldareva

University College London, MSc Connected Environments, 2023

Raspberry Pi Imager is free to install from the official Raspberry Pi website. It is under the download page section of the website. It's available for MacOS, Windows, Linux and Ubuntu systems.

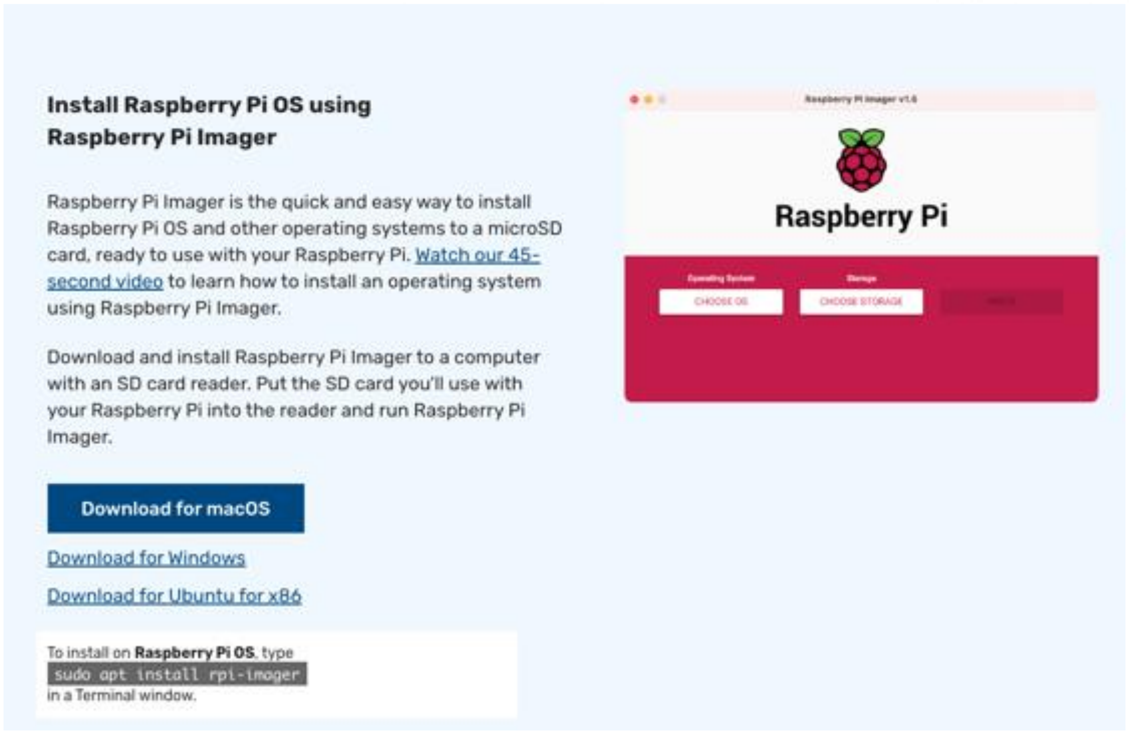
Visit Raspberry Pi Software section (https://www.raspberrypi.com/software/)



Raspberry Pi OS

Your Raspberry Pi needs an operating system to work. This is it. Raspberry Pi OS (previously called Raspbian) is our official supported operating system.

Download the latest version of Raspberry Pi Imager installer for your Operating System.



Install Raspberry Pi OS using Raspberry Pi Imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi. Watch our 45-second video to learn how to install an operating system using Raspberry Pi Imager.

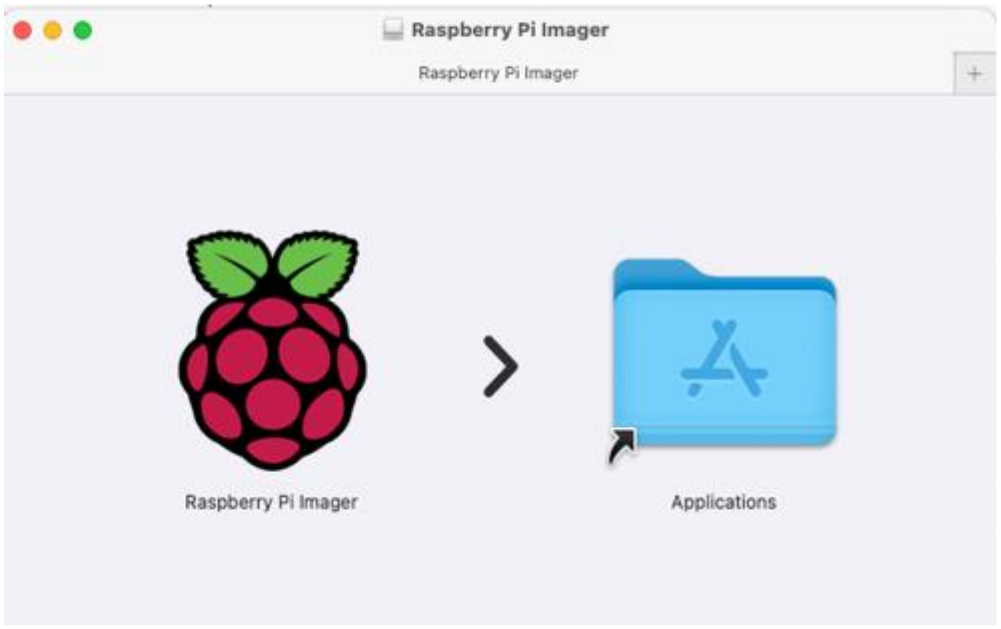
Download and install Raspberry Pi Imager to a computer with an SD card reader. Put the SD card you'll use with your Raspberry Pi into the reader and run Raspberry Pi Imager.

To install on **Raspberry Pi OS**, type

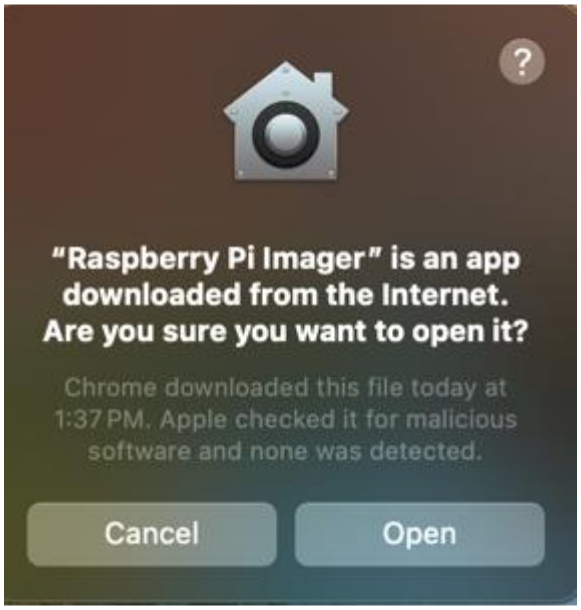
```
sudo apt install rpi-imager
```

in a Terminal window.

Click on Download for macOS and it will download the imager.dmg file. Open the file.

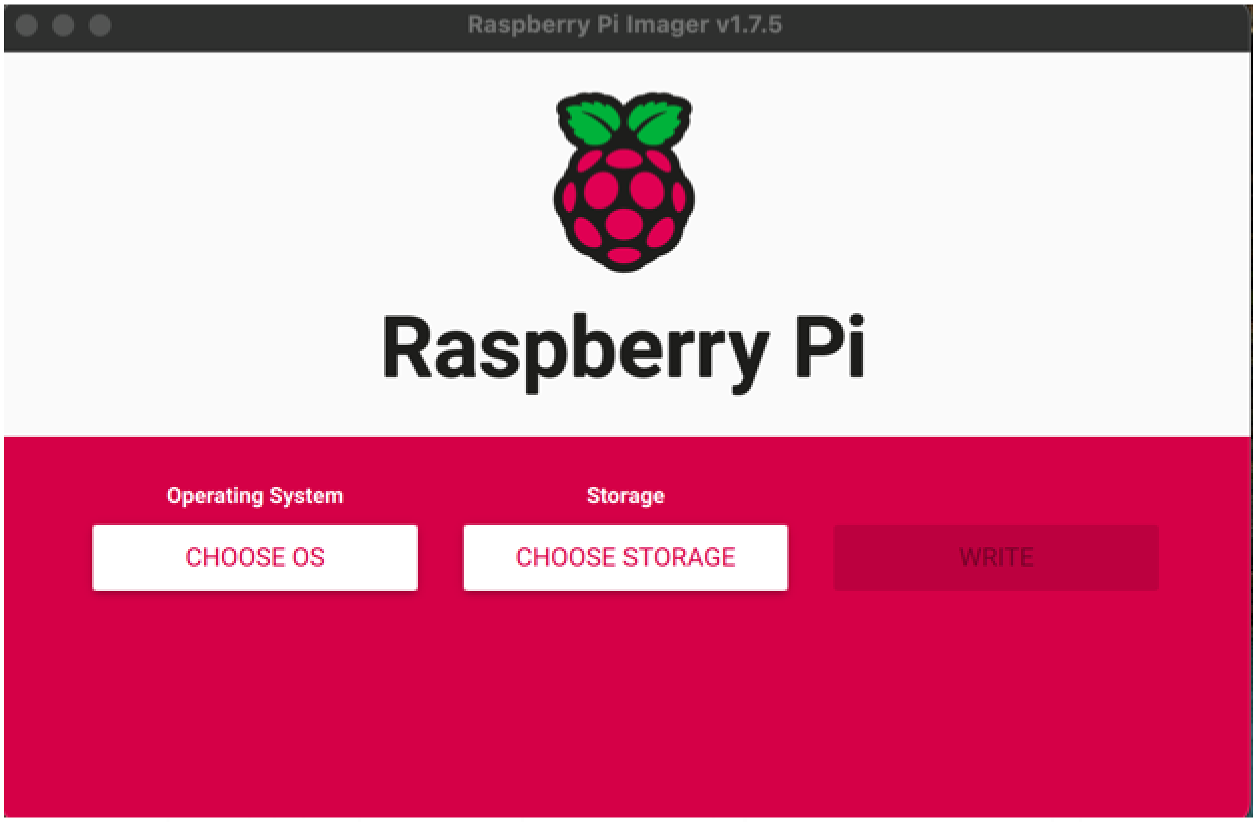


Once the file is open simply drag and drop Raspberry Pi Imager into the Applications folder and the program should be installed.

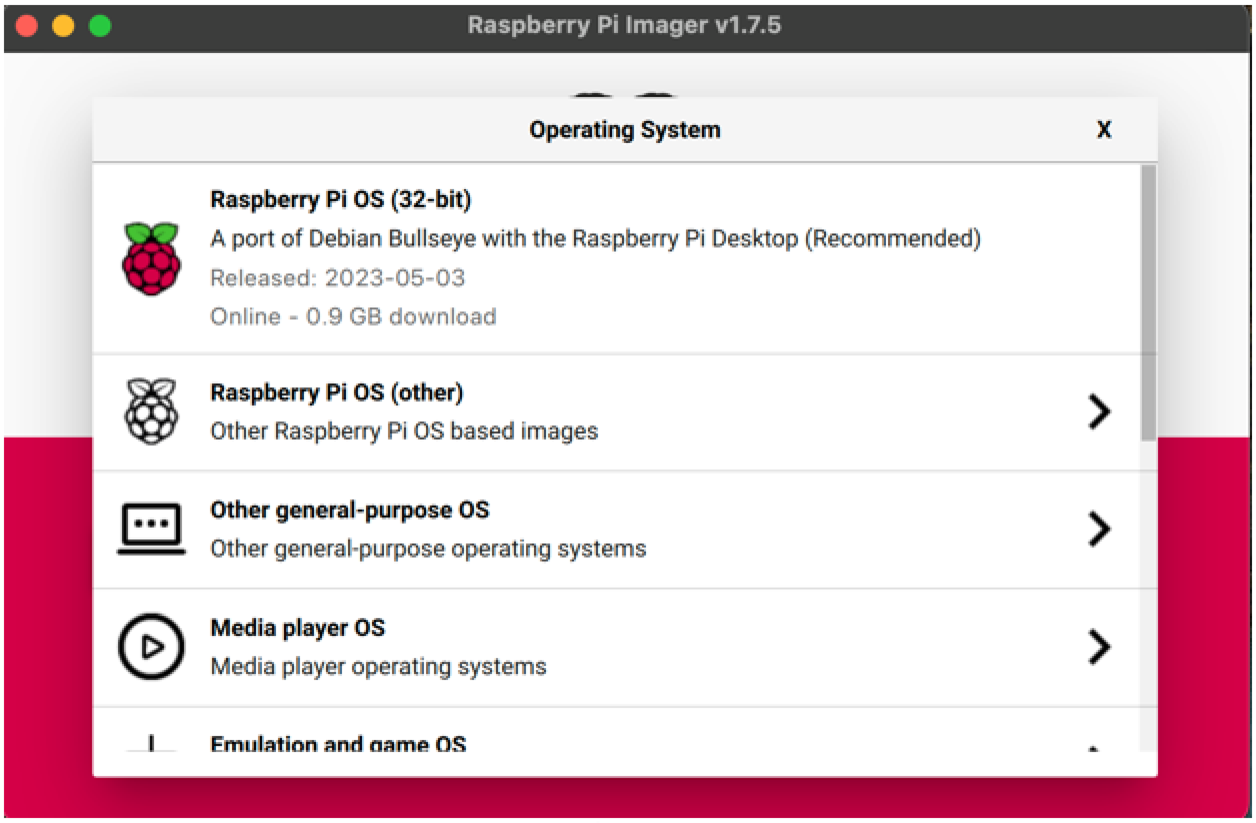

"Raspberry Pi Imager" is an app downloaded from the Internet. Are you sure you want to open it?

Chrome downloaded this file today at 1:37 PM. Apple checked it for malicious software and none was detected.

Now open Raspberry Pi Imager from the launchpad and the above prompt will appear. Click on Open. The program should open.



Now click on CHOOSE OS.


Raspberry Pi Imager v1.7.5

Operating System X

Raspberry Pi OS (32-bit)
A port of Debian Bullseye with the Raspberry Pi Desktop (Recommended)
Released: 2023-05-03
Online - 0.9 GB download

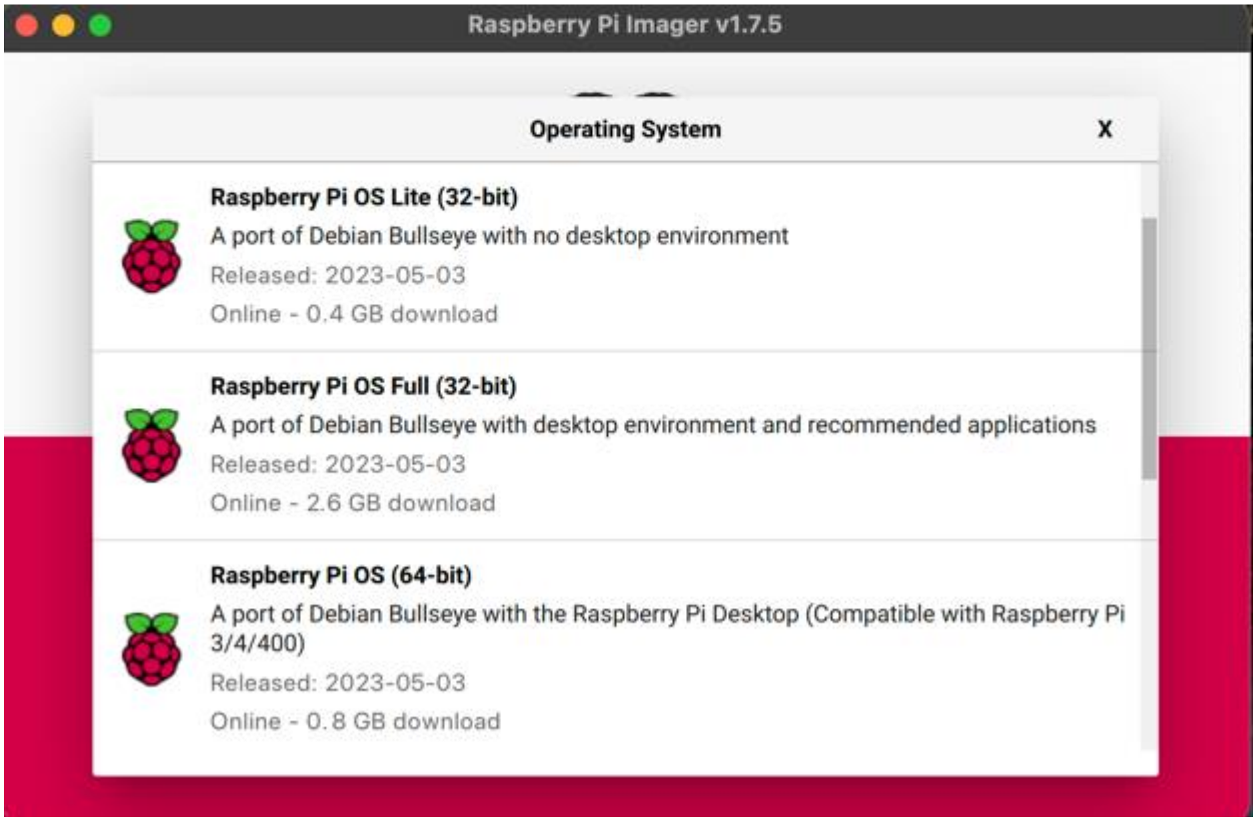
Raspberry Pi OS (other)
Other Raspberry Pi OS based images

Other general-purpose OS
Other general-purpose operating systems

Media player OS
Media player operating systems

Emulation and game OS

Operating System pop up will open. Click on Raspberry Pi OS (other). This project needs


Raspberry Pi Imager v1.7.5

Operating System X

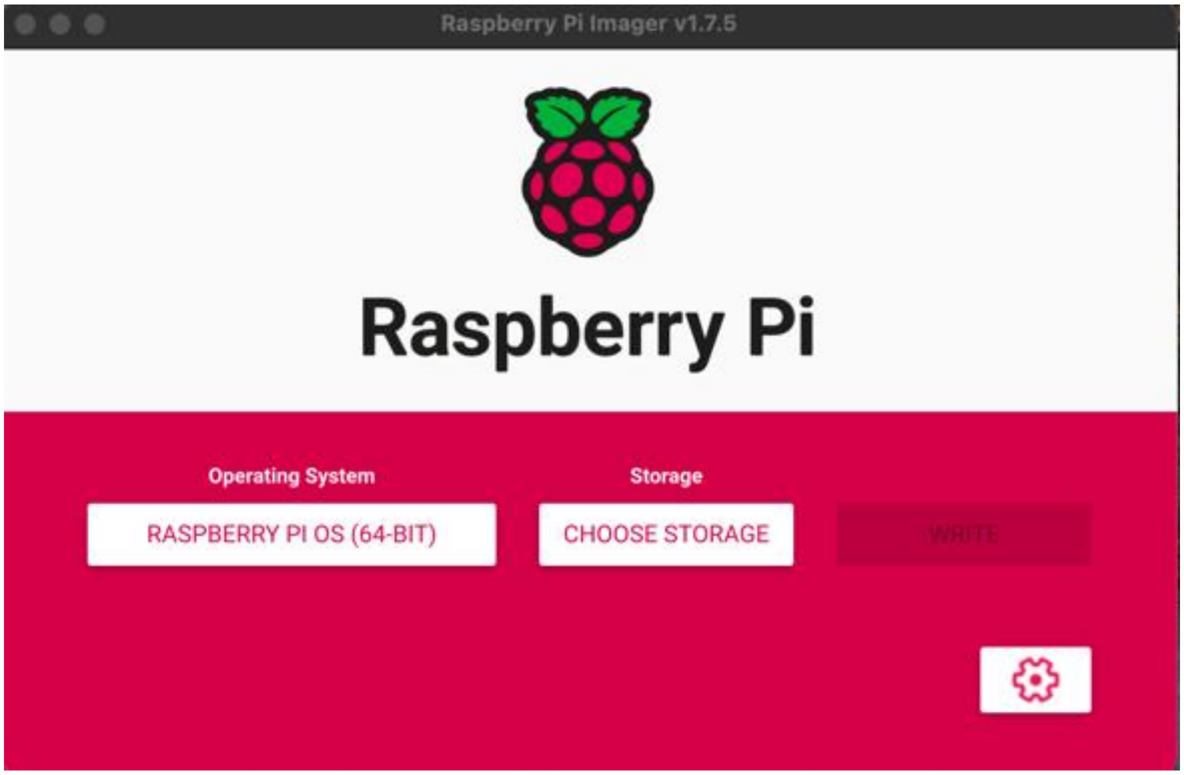
Raspberry Pi OS Lite (32-bit)
A port of Debian Bullseye with no desktop environment
Released: 2023-05-03
Online - 0.4 GB download

Raspberry Pi OS Full (32-bit)
A port of Debian Bullseye with desktop environment and recommended applications
Released: 2023-05-03
Online - 2.6 GB download

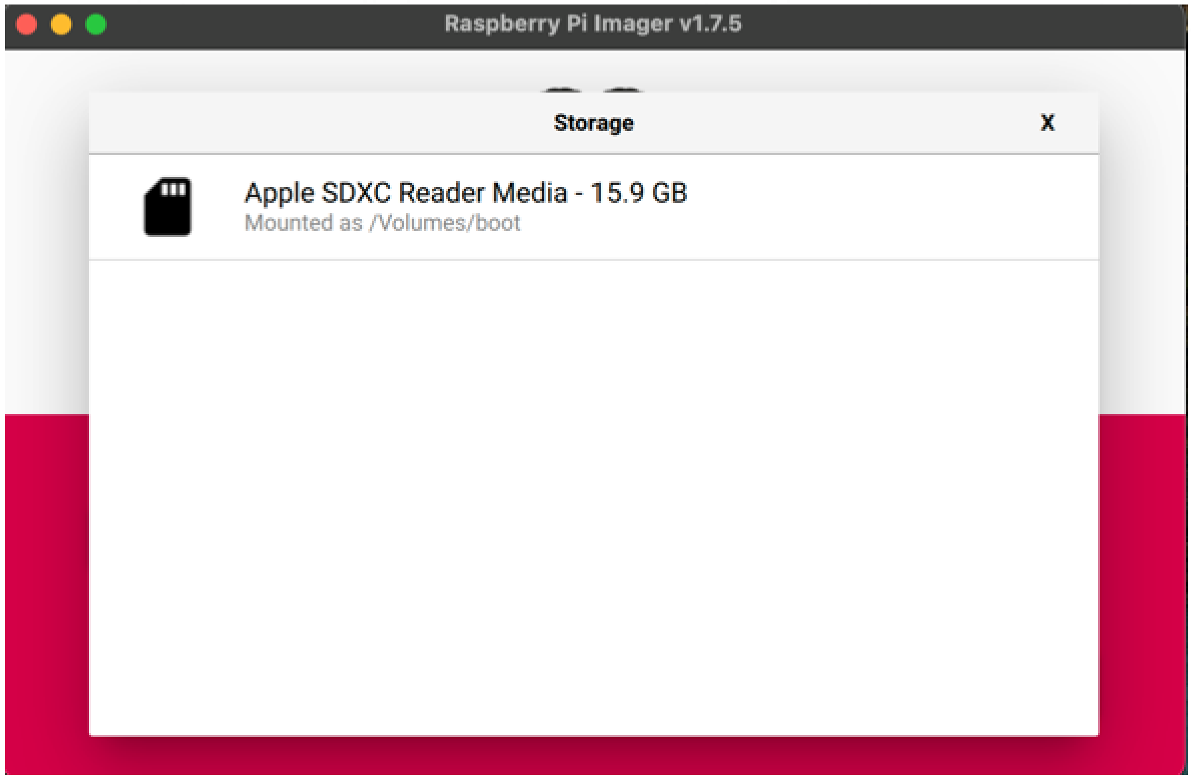
Raspberry Pi OS (64-bit)
A port of Debian Bullseye with the Raspberry Pi Desktop (Compatible with Raspberry Pi 3/4/400)
Released: 2023-05-03
Online - 0.8 GB download

64-bit OS for this development.

Scroll a bit down and choose Raspberry Pi OS (64-bit). Keep an eye on the size, this should be 0.8 GB as we need Desktop version of Debian Bullseye.

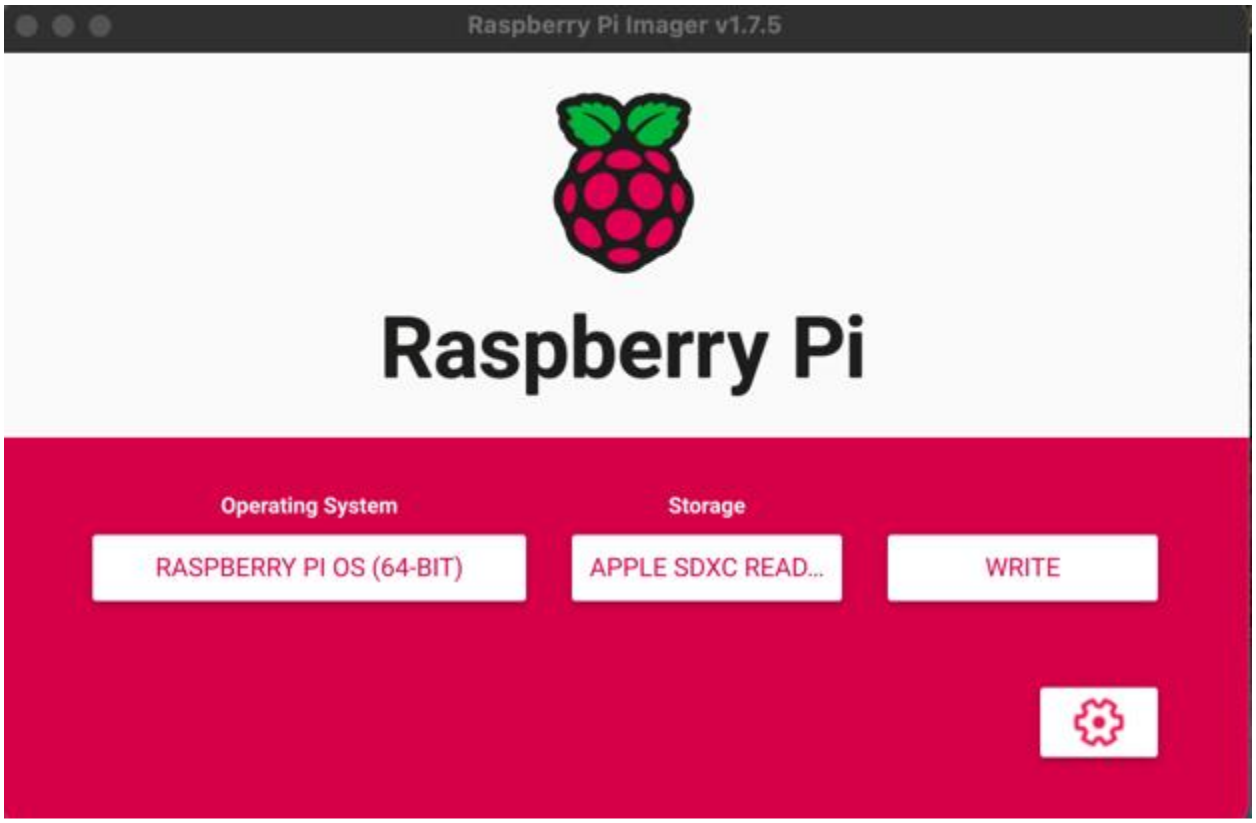


It will take you back once you select the OS. Insert the SD card into the mac. Now click on CHOOSE STORAGE.

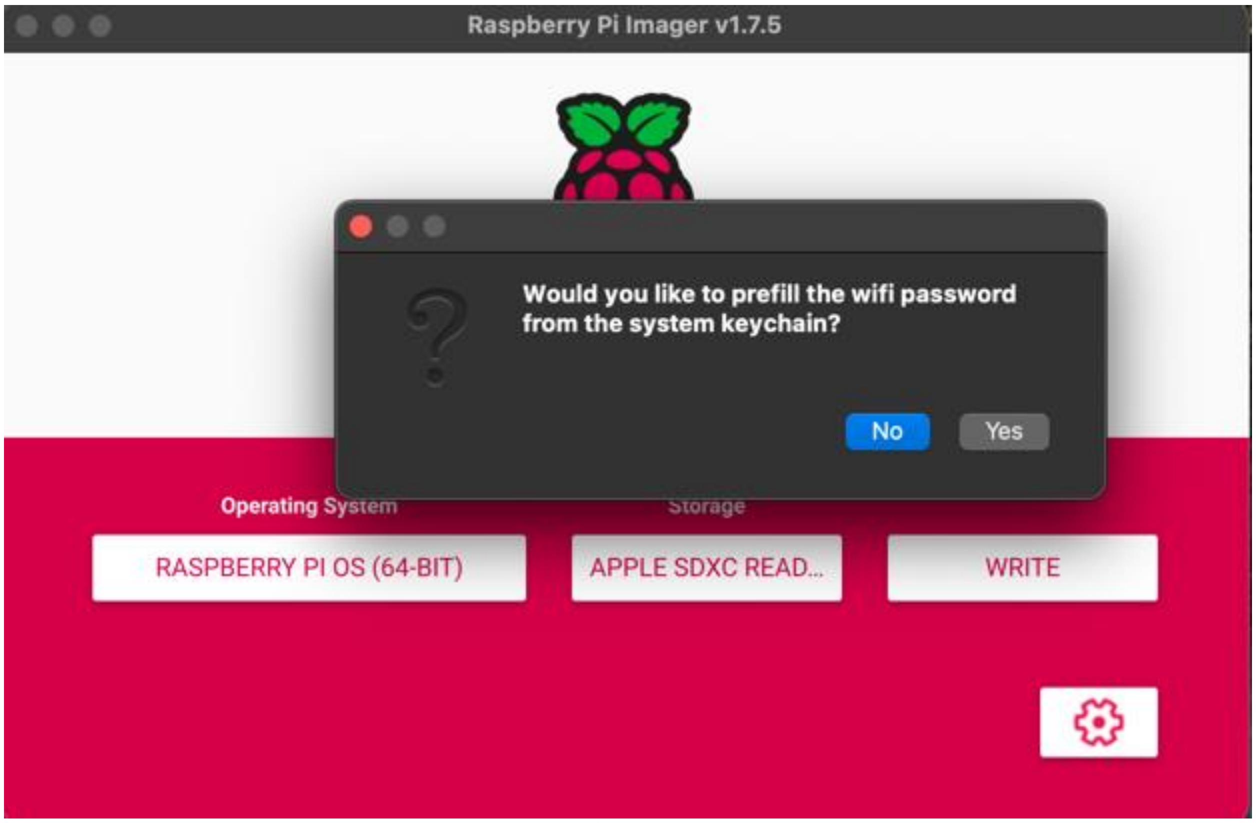


Apple SDXC Reader Media - 15.9 GB
Mounted as /Volumes/boot

Select the SD card.



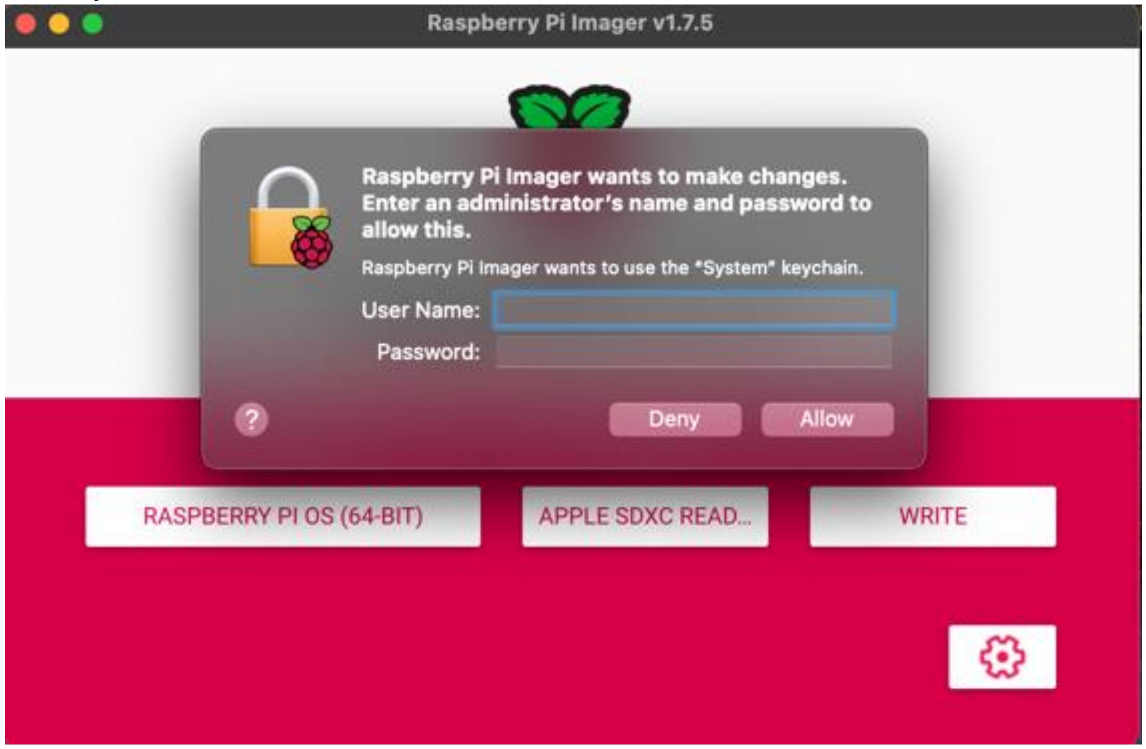
We can use Raspberry Pi Imager to setup a raspberry pi in the headless mode without the need for additional peripherals such as monitor, keyboard and mouse. By configuring our Wi-fi credentials and locale settings into the OS we can use this imaging tool to write the OS onto the SD card. So, we click on the setting logo.



Raspberry Pi Imager v1.7.5

Would you like to prefill the wifi password from the system keychain?

A prompt would ask saying Would you like to prefill the Wi-Fi password from the system keychain? For easier access we can select Yes.



Raspberry Pi Imager v1.7.5

Raspberry Pi Imager wants to make changes. Enter an administrator's name and password to allow this.

Raspberry Pi Imager wants to use the "System" keychain.

User Name:

Password:

It will ask for system credentials. Please provide the credentials and click on Allow.

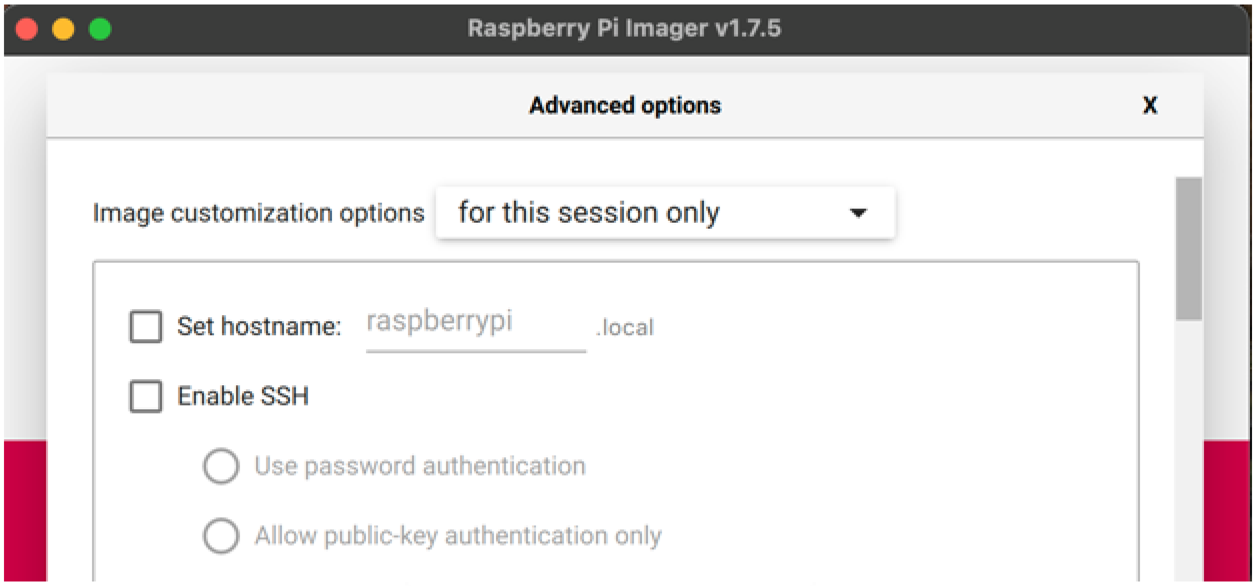


Image customization options: for this session only

☐ Set hostname: raspberrypi .local

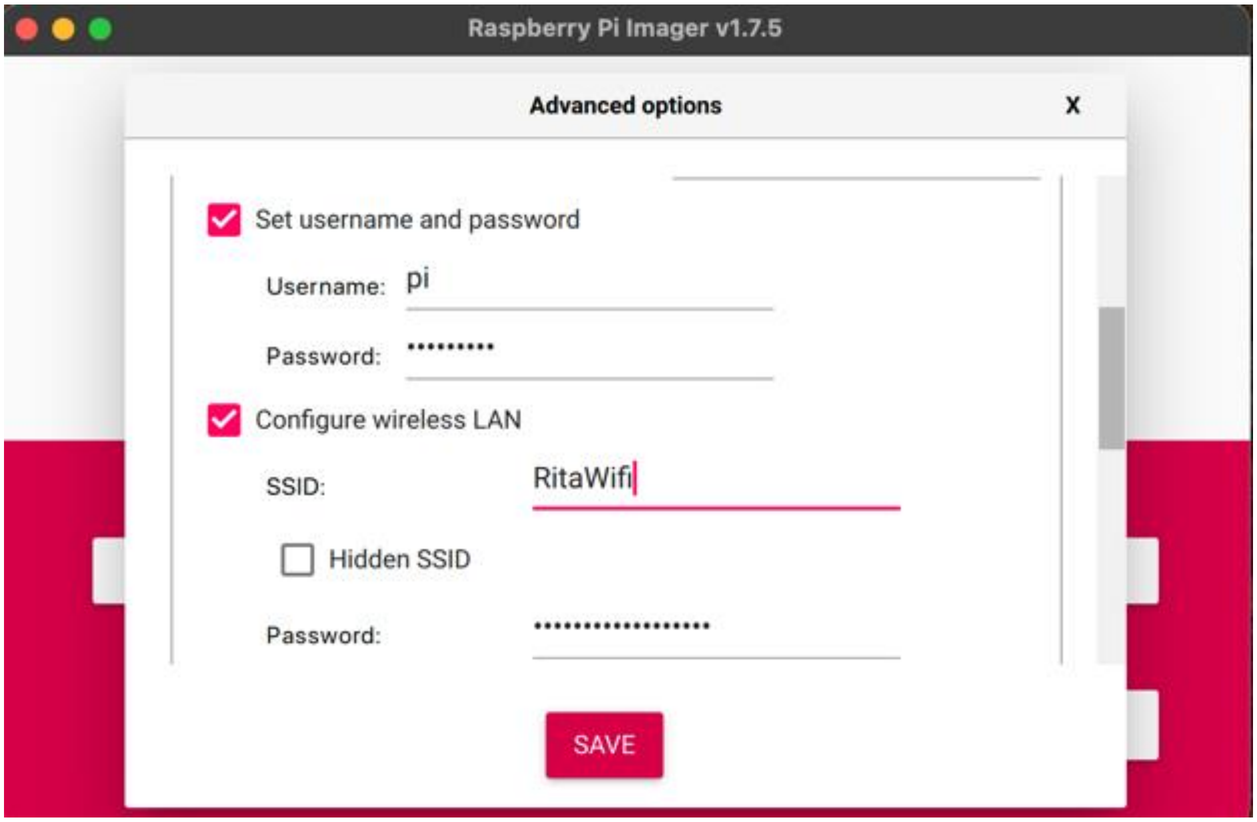
☐ Enable SSH

○ Use password authentication

○ Allow public-key authentication only

You can select Image customization options to “to always use”. Select Enable SSH.

Then scroll down a bit.


Raspberry Pi Imager v1.7.5

Advanced options

[✓] Set username and password

Username: pi

Password: ••••••••••

[✓] Configure wireless LAN

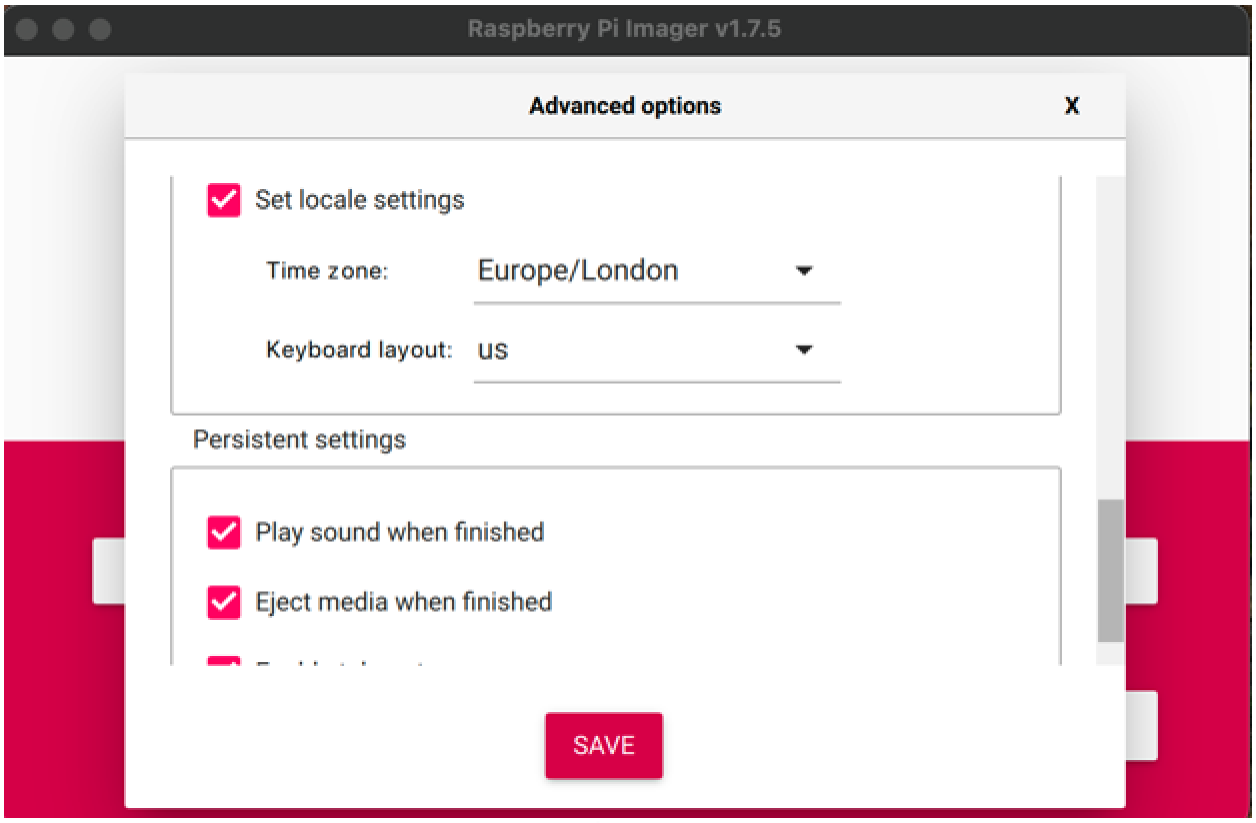
SSID: RitaWifi

[ ] Hidden SSID

Password: ••••••••••••••••••

The set Username as "pi" and password as "raspberry". And also provide your Wi-Fi credentials.

Then scroll again.



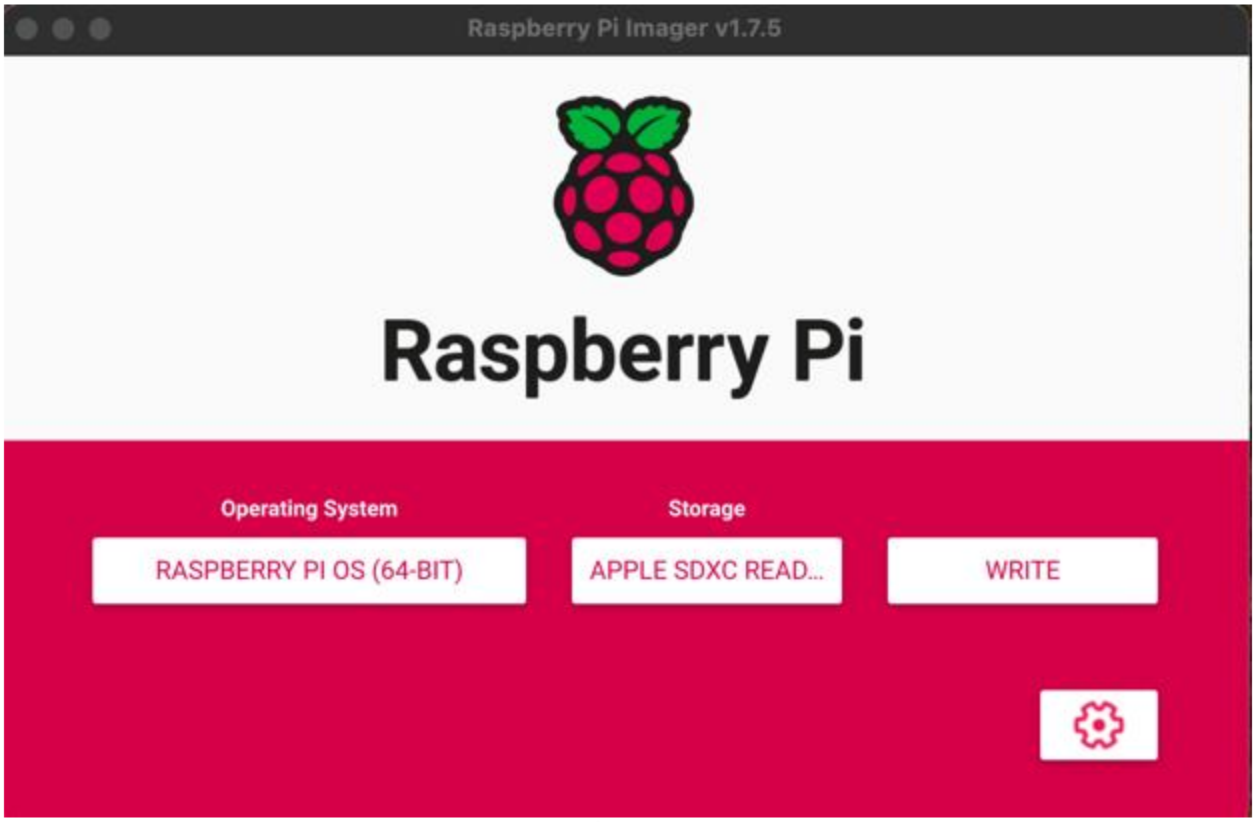
Advanced options

- [✓] Set locale settings
  - Time zone: Europe/London
  - Keyboard layout: us

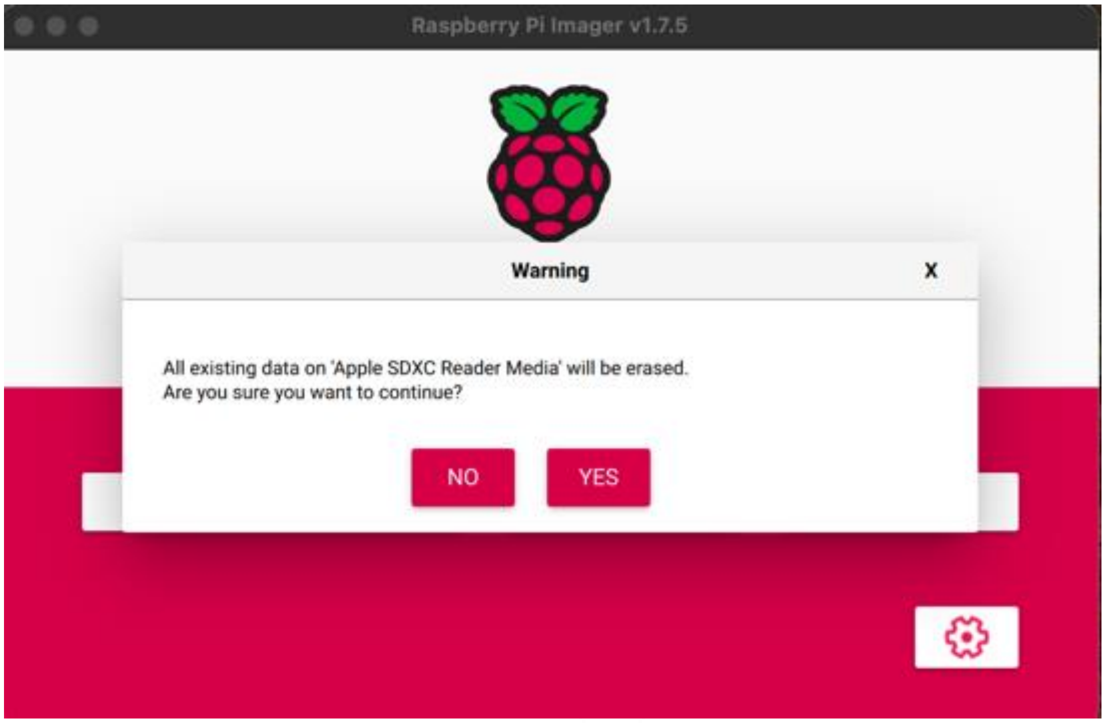
Persistent settings

- [✓] Play sound when finished
- [✓] Eject media when finished

and select on set locale settings and choose your desired Time zone and Keyboard layout. Click on SAVE.

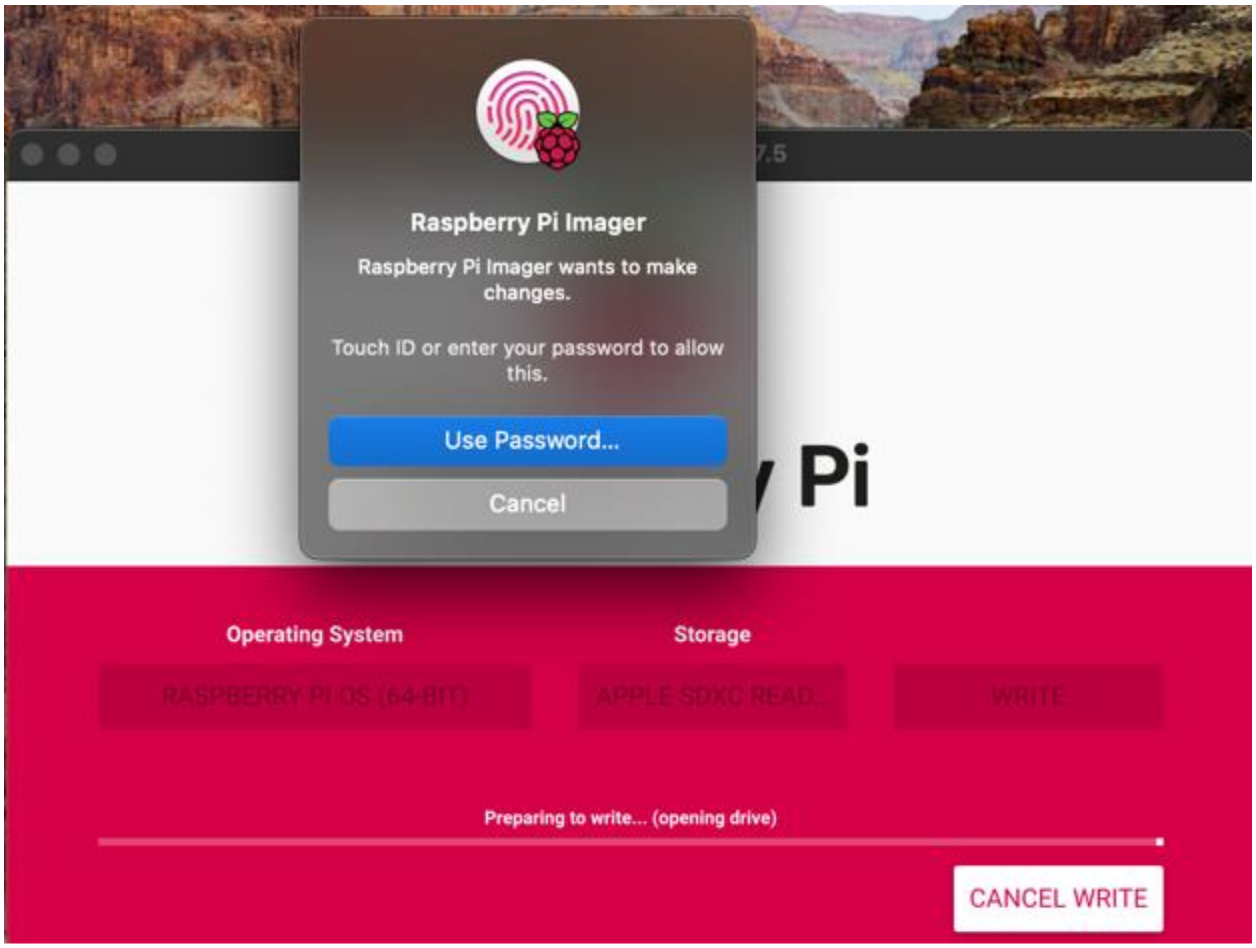


It will take you back to the home screen. Now simply press WRITE button. The process will start.



All existing data on 'Apple SDXC Reader Media' will be erased.
Are you sure you want to continue?

Click on YES.

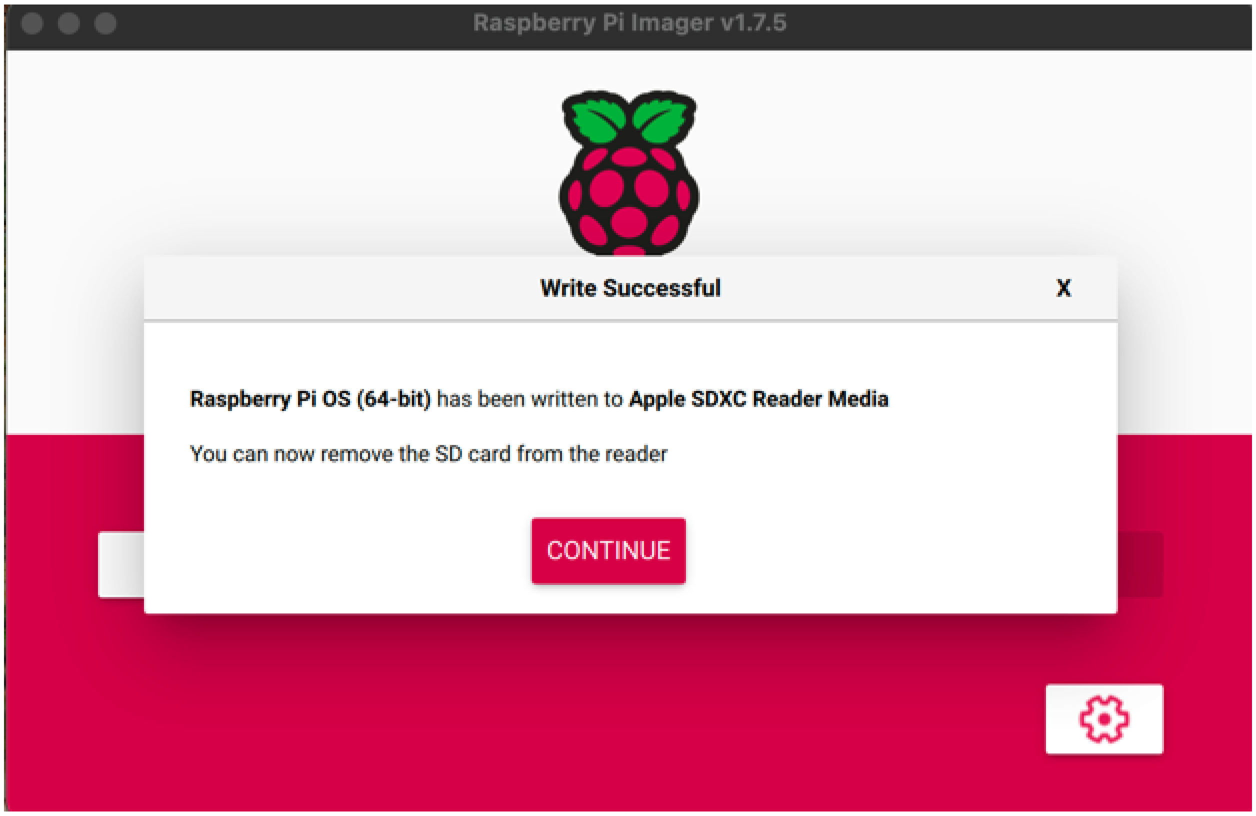

Raspberry Pi Imager

Raspberry Pi Imager wants to make changes.

Touch ID or enter your password to allow this.

Preparing to write... (opening drive)

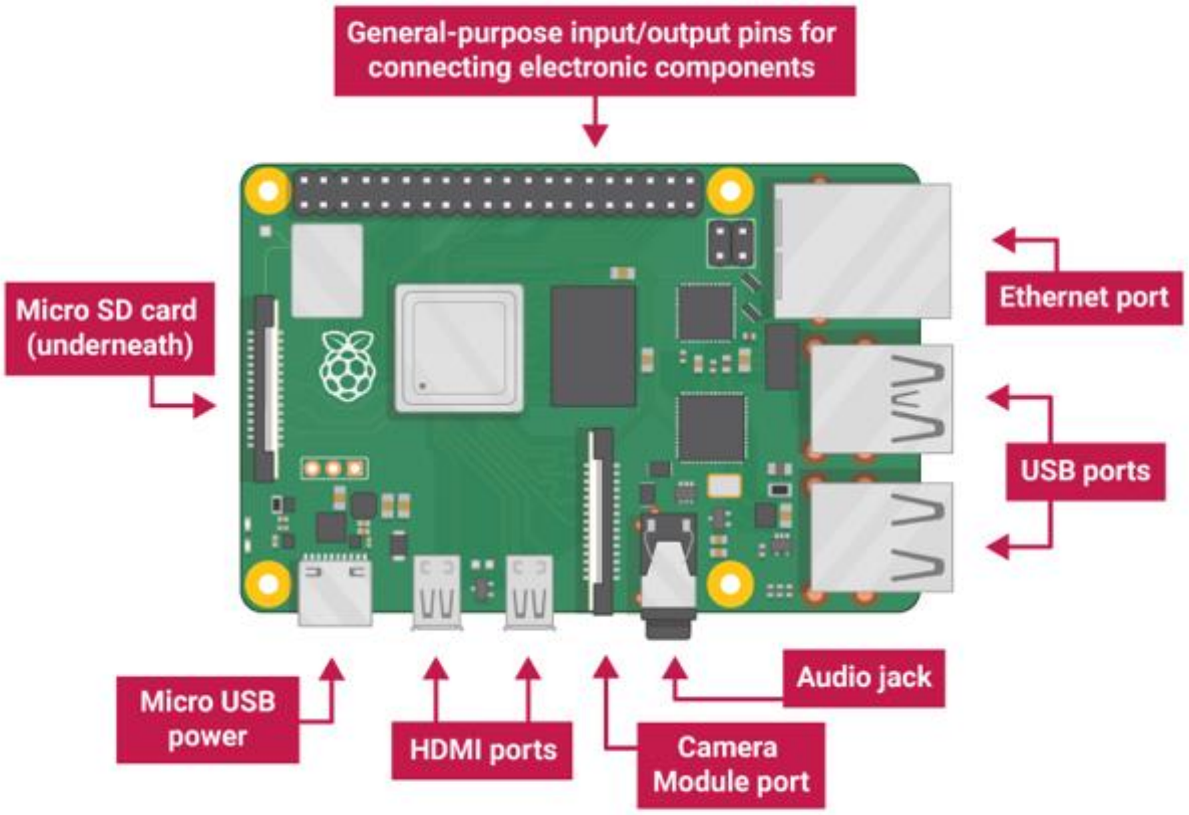
Please verify either using fingerprint or password and the process will start writing the OS into the SD card.



Raspberry Pi OS (64-bit) has been written to **Apple SDXC Reader Media**

You can now remove the SD card from the reader

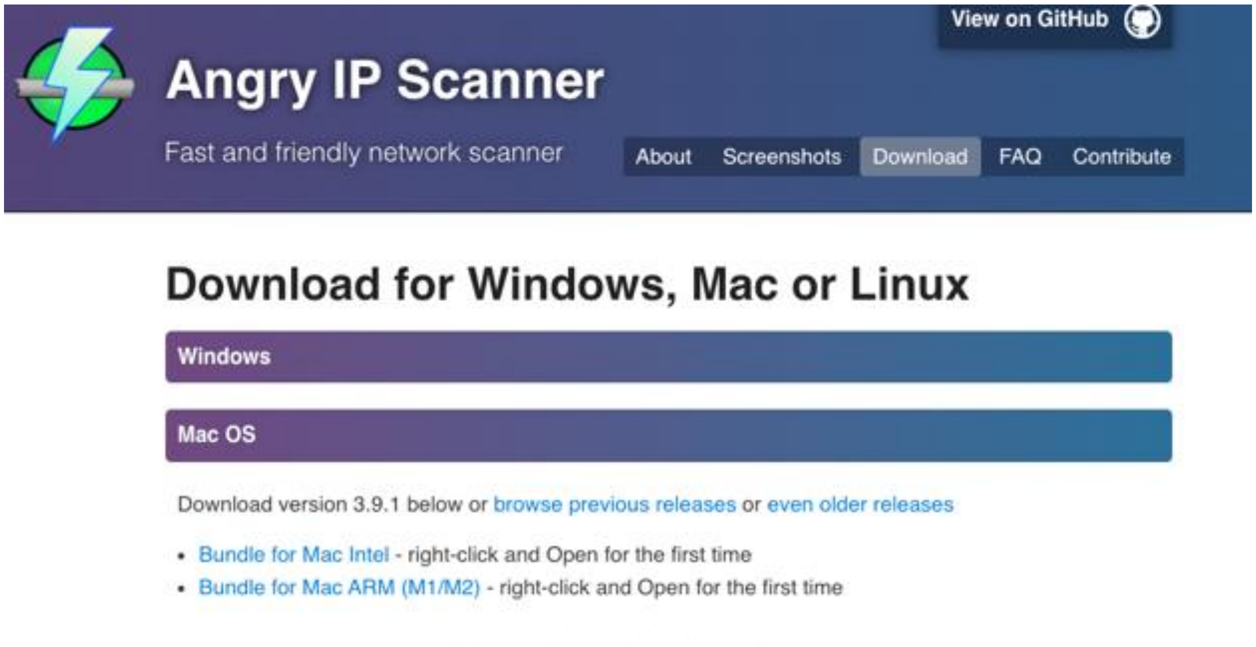
Once the writing process is complete it will prompt a message with Write Successful message. Click on CONTINUE. Remove the SD card from the laptop.


General-purpose input/output pins for connecting electronic components
Micro SD card (underneath)
Ethernet port
USB ports
Micro USB power
HDMI ports
Camera Module port
Audio jack

Insert the SD card in the raspberry pi. Connect the Micro USB power using raspberry pi 4 official power supply. The red and green status LED on the side of the Micro USB power will start to blink.

Now once the pi boots up it will connect to the Wi-Fi; we provided credentials in the setup section. In order to connect to the Pi, we need to know the IP address of the raspberry pi. For that purpose, we can use a software called angry IP scanner. It's available for MacOS, Windows, Linux and Ubuntu systems.

Visit (https://angryip.org/download/#mac)



Download for Windows, Mac or Linux

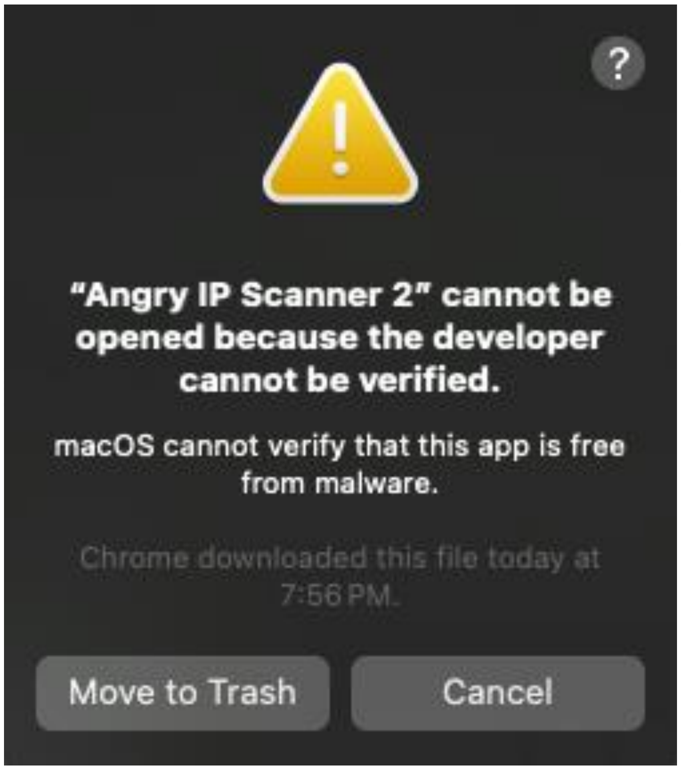
Windows

Mac OS

Download version 3.9.1 below or browse previous releases or even older releases

- Bundle for Mac Intel - right-click and Open for the first time
- Bundle for Mac ARM (M1/M2) - right-click and Open for the first time

Select the mac type and download the file. Once the file is downloaded.

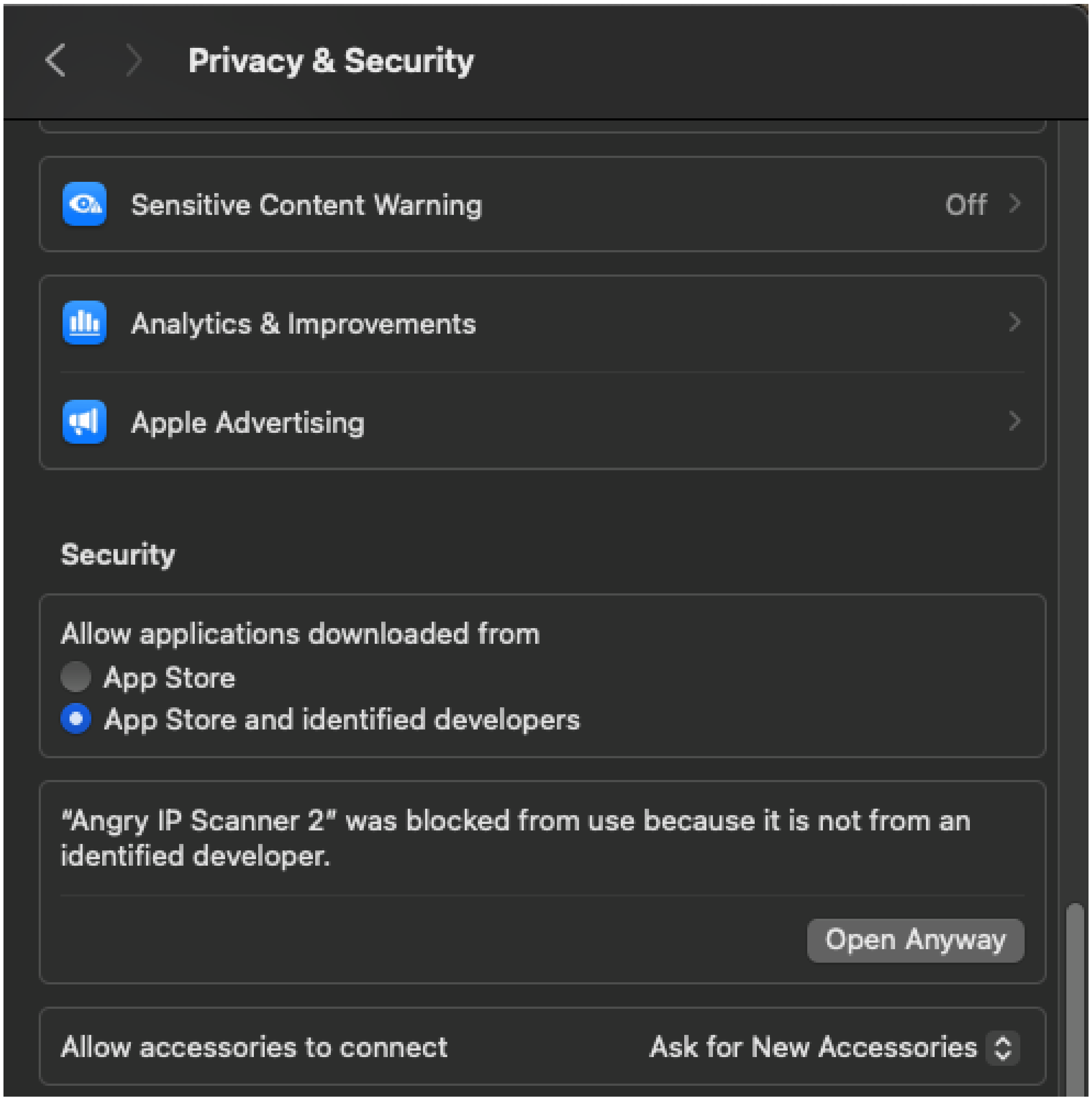


"Angry IP Scanner 2" cannot be opened because the developer cannot be verified.

macOS cannot verify that this app is free from malware.

Chrome downloaded this file today at 7:56 PM.

MacOS will prompt with this message. You will have to Cancel this. Developer mode has to be enabled in the system. After that go to mac’s Settings, then Privacy and Security then scroll until you see this.



Privacy & Security

Sensitive Content Warning: Off

Analytics & Improvements

Apple Advertising

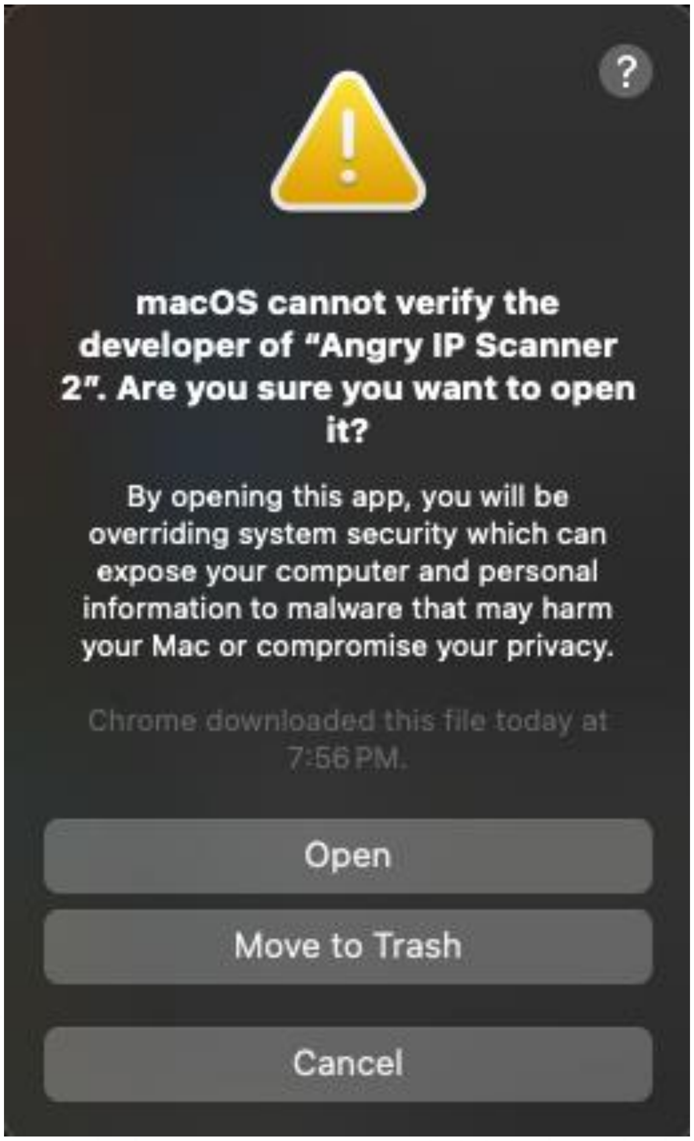
Security

Allow applications downloaded from
App Store
[✓ App Store and identified developers]

"Angry IP Scanner 2" was blocked from use because it is not from an identified developer.

Allow accessories to connect: Ask for New Accessories

Click on Open Anyway. It will prompt for password provide the password.

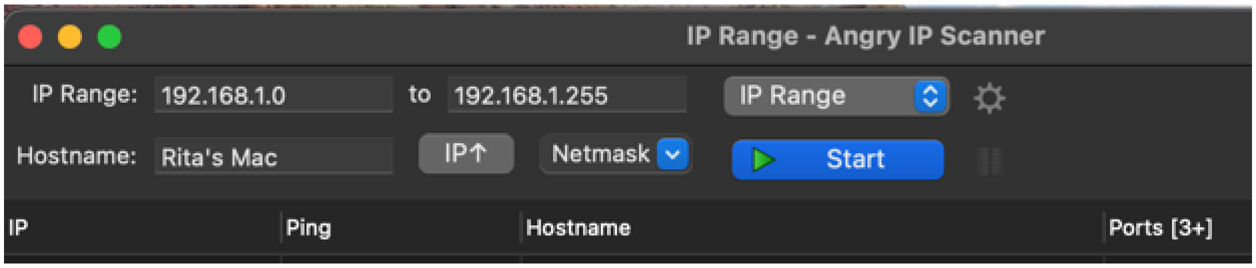


macOS cannot verify the developer of "Angry IP Scanner 2". Are you sure you want to open it?

By opening this app, you will be overriding system security which can expose your computer and personal information to malware that may harm your Mac or compromise your privacy.

Chrome downloaded this file today at 7:56 PM.

Click on Open. The application will open. Click on Start.



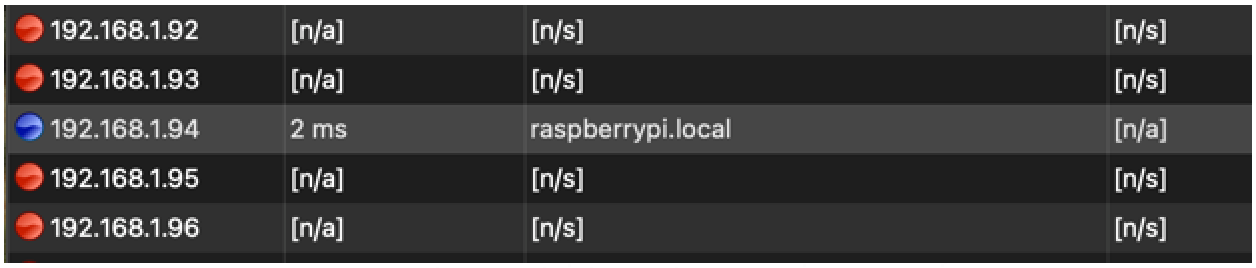
IP Range - Angry IP Scanner

IP Range: 192.168.1.0 to 192.168.1.255 IP Range

Hostname: Rita's Mac IP↑ Netmask

IP	Ping	Hostname	Ports [3+]

Once the process is complete. Close the prompt. Select anyone of the IP address. Press CMD+H until you see raspberrypi.local on the host name. Sometimes raspberry pi will not be able to connect to the Wi-Fi. In that case an ethernet cable can be connected directly from router to the pi. And also keep in mind that the Mac has to be on the same network.

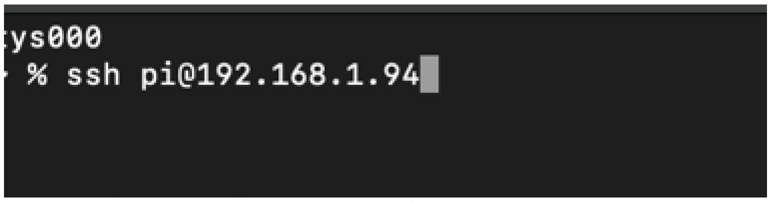


🔴 192.168.1.92	[n/a]	[n/s]	[n/s]
🔴 192.168.1.93	[n/a]	[n/s]	[n/s]
🔵 192.168.1.94	2 ms	raspberrypi.local	[n/a]
🔴 192.168.1.95	[n/a]	[n/s]	[n/s]
🔴 192.168.1.96	[n/a]	[n/s]	[n/s]

Now we have our IP address of the Pi. We can use mac's terminal to SSH into the PI.

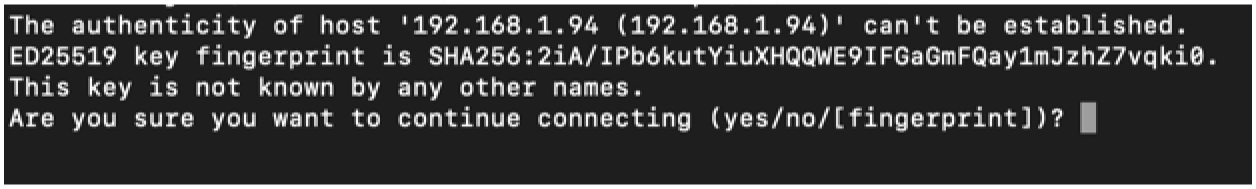
Open terminal and enter the command

ssh pi@192.168.94



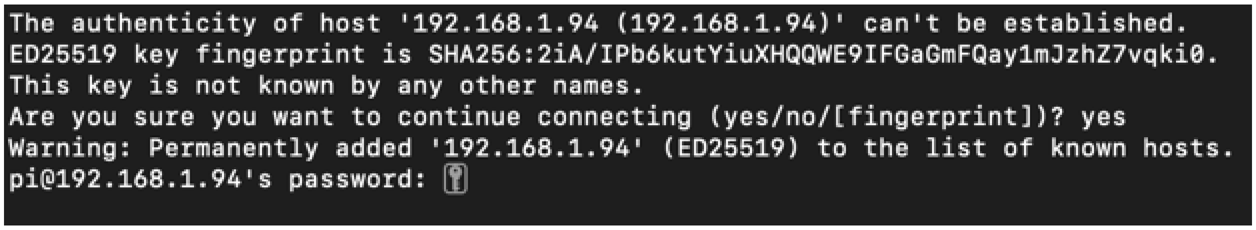
```
ys000
 % ssh pi@192.168.1.94
```

Now press enter.



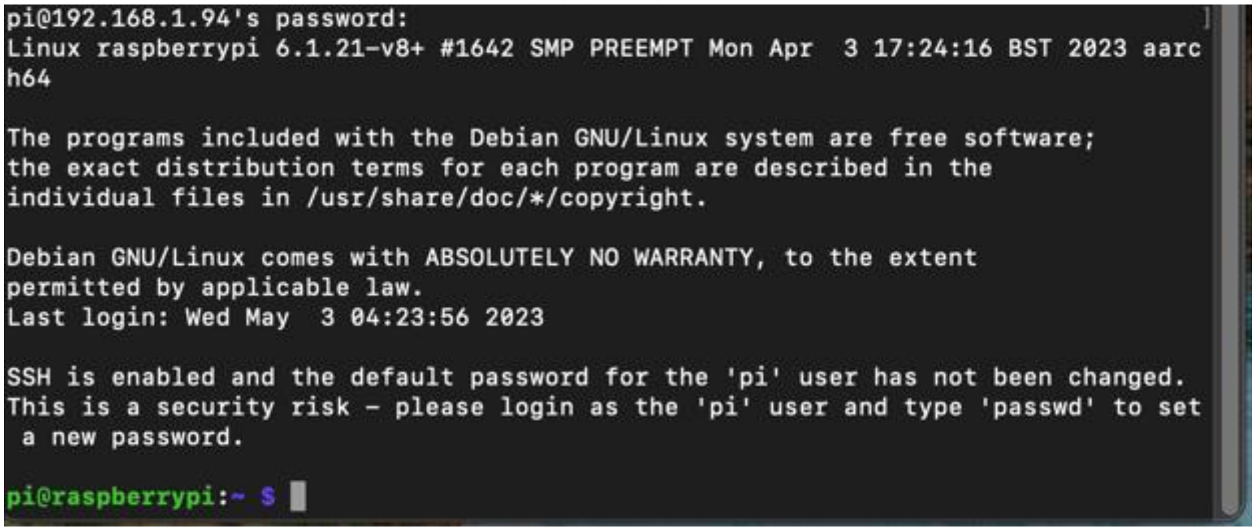
```
The authenticity of host '192.168.1.94 (192.168.1.94)' can't be established.
ED25519 key fingerprint is SHA256:2iA/IPb6kutYiuXHQQWE9IFGaGmFQay1mJzhZ7vqki0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Types yes and press enter.



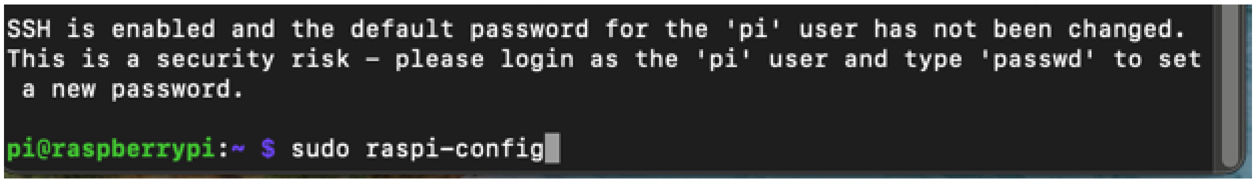
```
The authenticity of host '192.168.1.94 (192.168.1.94)' can't be established.
ED25519 key fingerprint is SHA256:2iA/IPb6kutYiuXHQQWE9IFGaGmFQay1mJzhZ7vqki0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.94' (ED25519) to the list of known hosts.
pi@192.168.1.94's password:
```

The password was set to “raspberry”. As you type in the password you will see no characters. Once the password is completed press enter and you will be logged into raspberry pi console.



```
pi@192.168.1.94's password:
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarc
h64

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 May  3 04:23:56 2023

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $
```

Now we can enable VNC viewer to have Desktop GUI access.

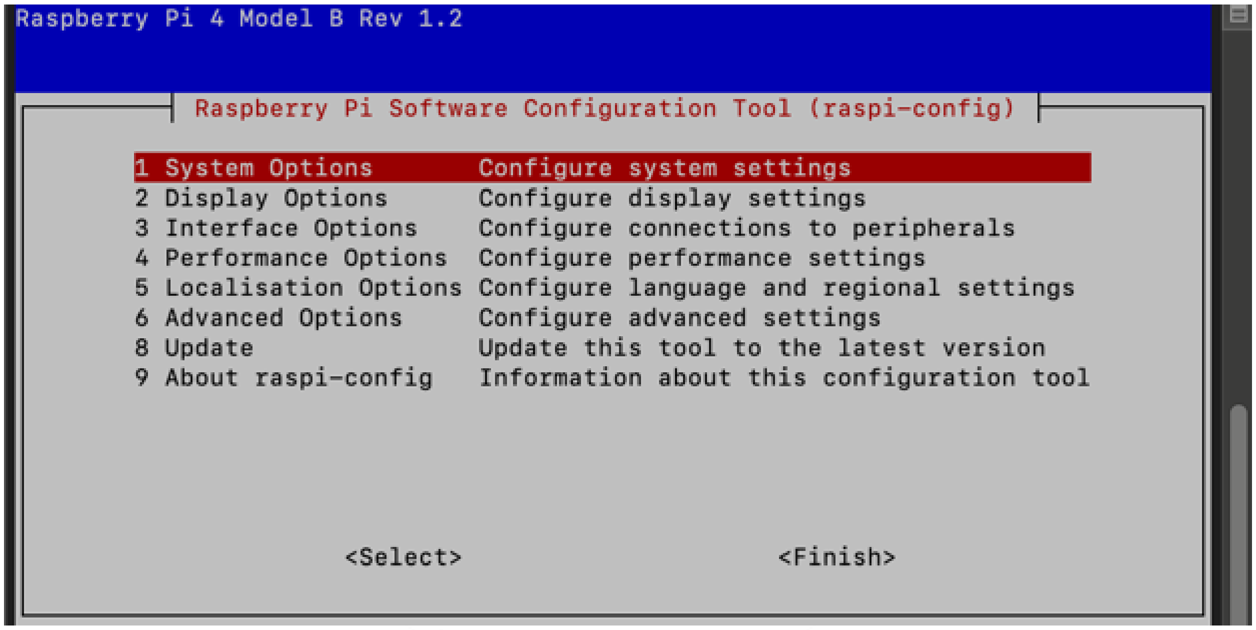


```
SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $ sudo raspi-config
```

Type the command

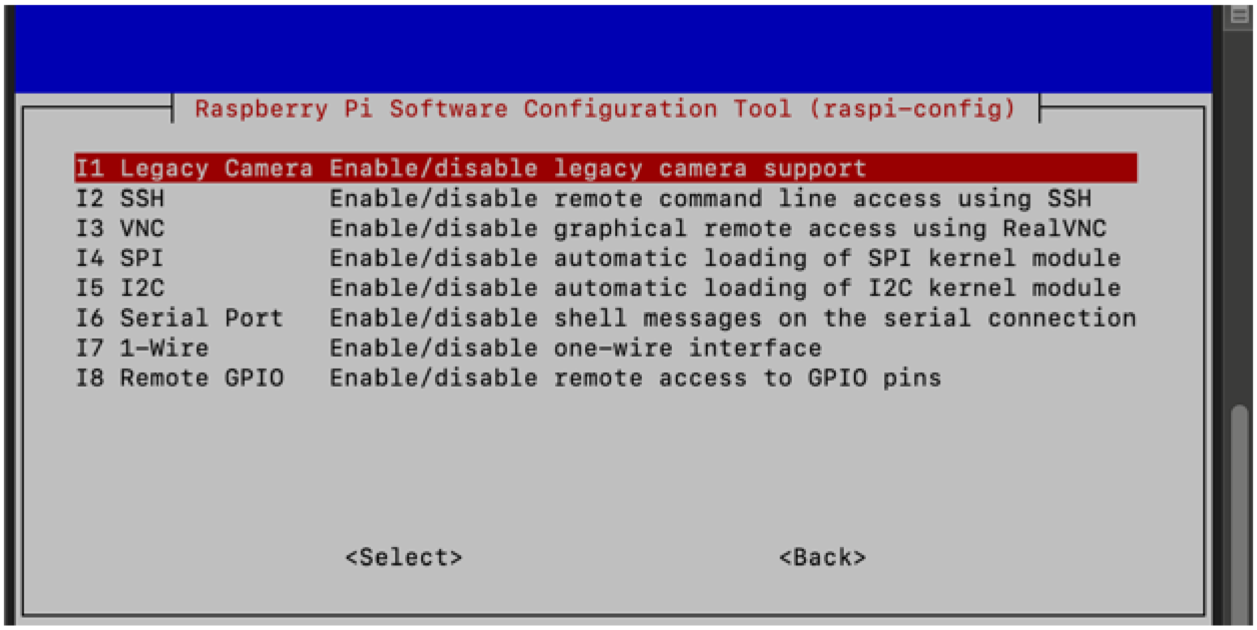
sudo raspi-config

And press enter this will direct to raspberry pi configuration.



```
Raspberry Pi 4 Model B Rev 1.2

        Raspberry Pi Software Configuration Tool (raspi-config)

    1 System Options       Configure system settings
    2 Display Options      Configure display settings
    3 Interface Options    Configure connections to peripherals
    4 Performance Options  Configure performance settings
    5 Localisation Options Configure language and regional settings
    6 Advanced Options     Configure advanced settings
    8 Update               Update this tool to the latest version
    9 About raspi-config   Information about this configuration tool

                <Select>                    <Finish>
```

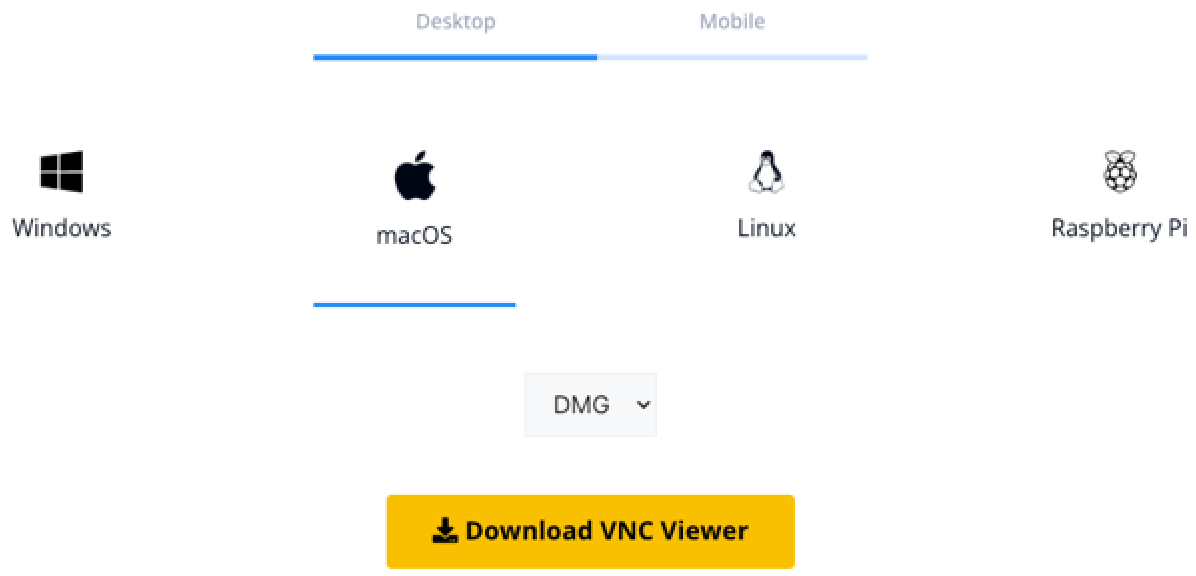
Go to interface options



```
Raspberry Pi Software Configuration Tool (raspi-config)

I1 Legacy Camera Enable/disable legacy camera support
I2 SSH           Enable/disable remote command line access using SSH
I3 VNC           Enable/disable graphical remote access using RealVNC
I4 SPI           Enable/disable automatic loading of SPI kernel module
I5 I2C           Enable/disable automatic loading of I2C kernel module
I6 Serial Port   Enable/disable shell messages on the serial connection
I7 1-Wire        Enable/disable one-wire interface
I8 Remote GPIO   Enable/disable remote access to GPIO pins

                <Select>                        <Back>
```

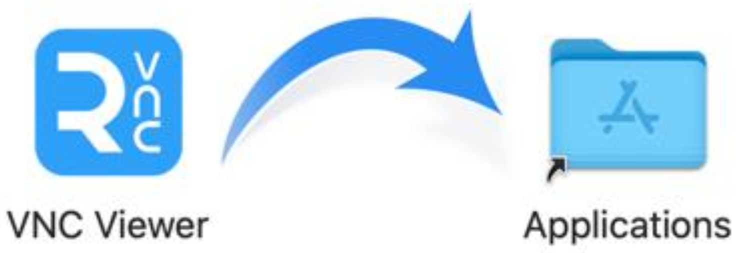
Go to VNC, enable the VNC Server by choosing Yes. Then go to Display options. Then VNC Resolution and select 1280x720. Once that is done, click on tab and finish it, when asked for reboot select yes.

After this the SSH session will end on Terminal. Now let's install VNC viewer. Go to (https://www.realvnc.com/en/connect/download/viewer/)

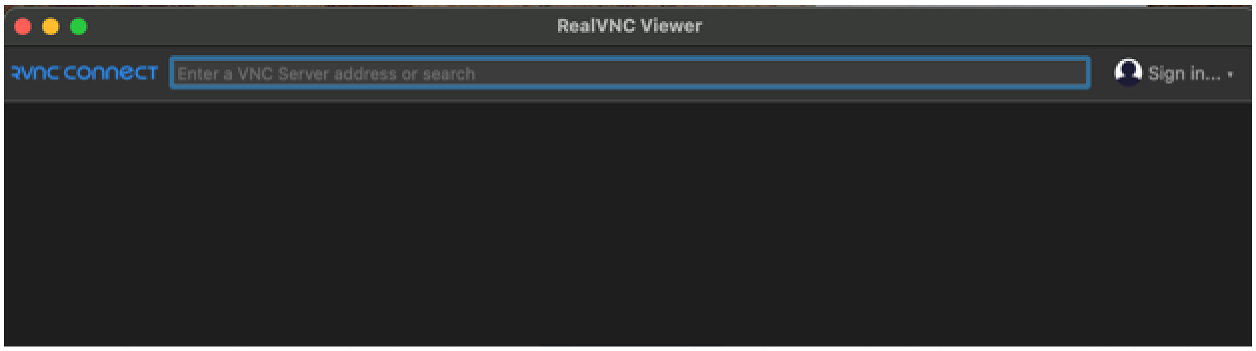


Click on Download VNC Viewer. Once the DMG file is downloaded open it.

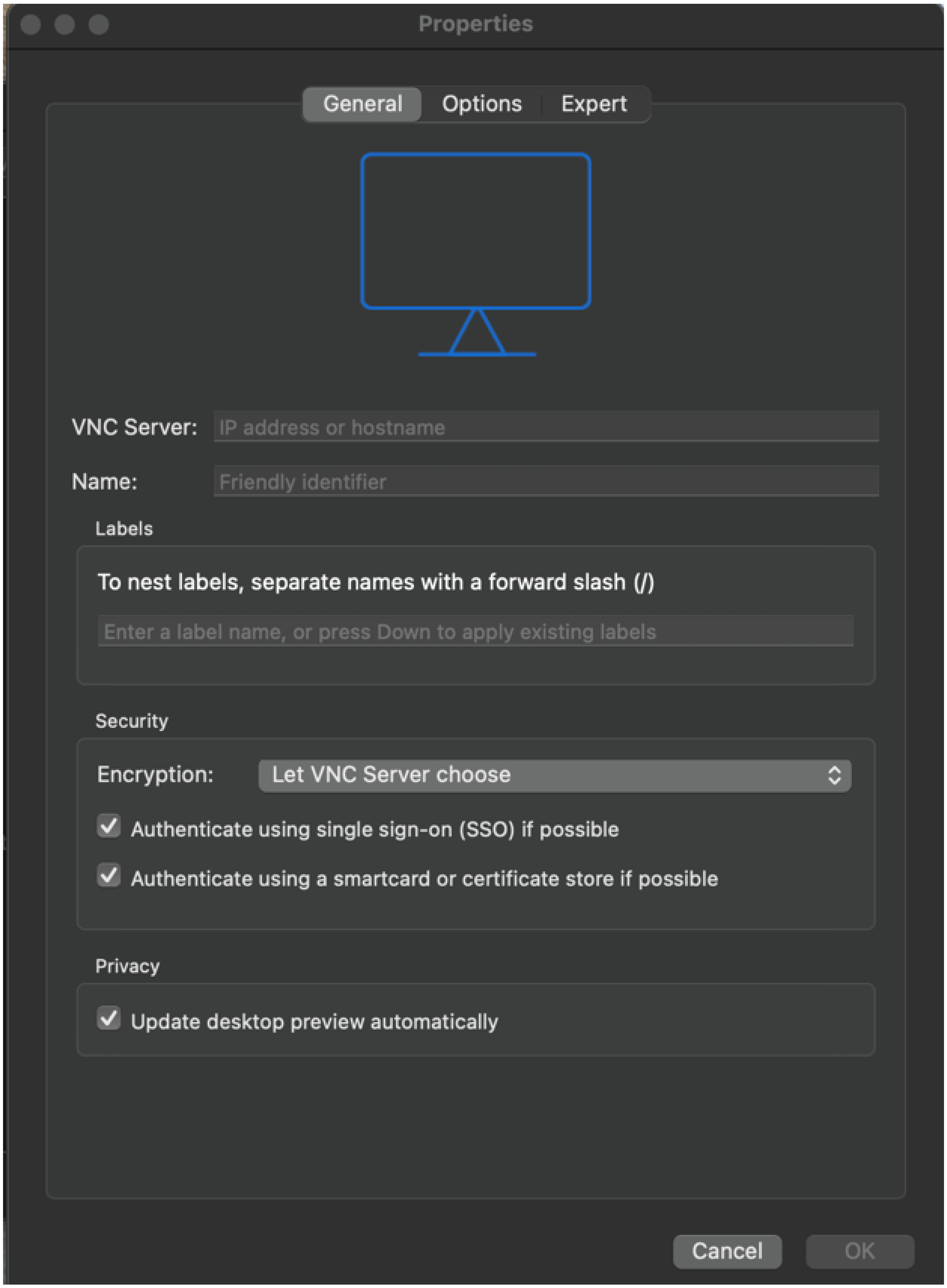
VNC-Viewer-7.6.0-MacOSX-universal
VNC-Viewer-7.6.0-MacOSX-universal



Just drag and drop to install. Once the installation is complete, open the VNC Viewer from the launchpad and click on open when prompt appears.



You will see the application like this. Right click on the body section and select new connection.



VNC Server: IP address or hostname

Name: Friendly identifier

Labels

To nest labels, separate names with a forward slash (/)

Enter a label name, or press Down to apply existing labels

Security

Encryption: Let VNC Server choose

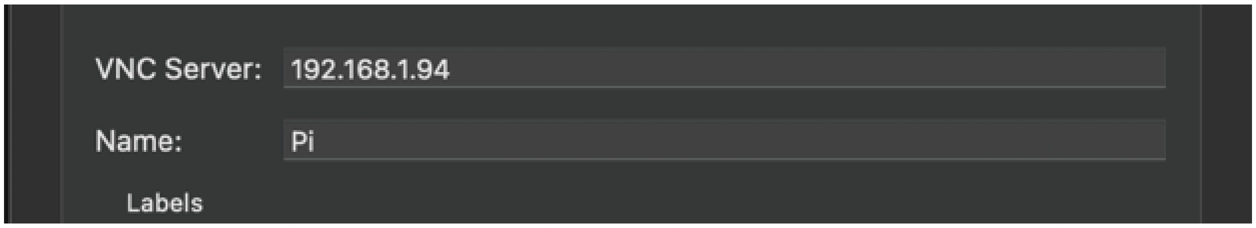
✓ Authenticate using single sign-on (SSO) if possible

✓ Authenticate using a smartcard or certificate store if possible

Privacy

✓ Update desktop preview automatically

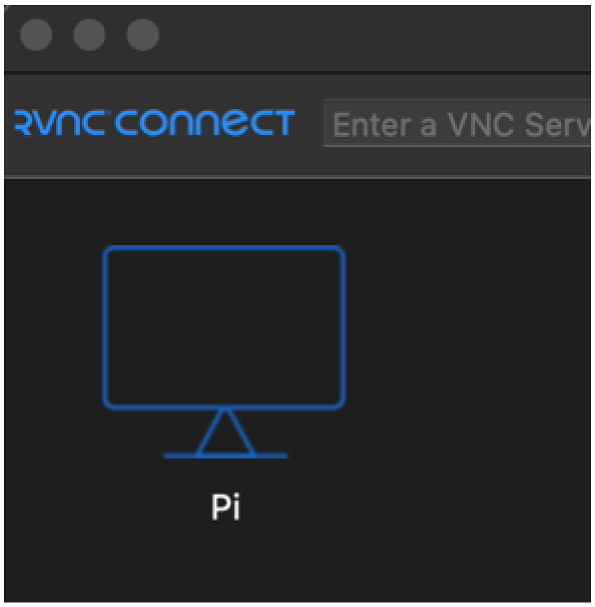
In VNC Server we have to input the IP address of the Pi. (192.168.1.94) and the name can be anything (Pi).



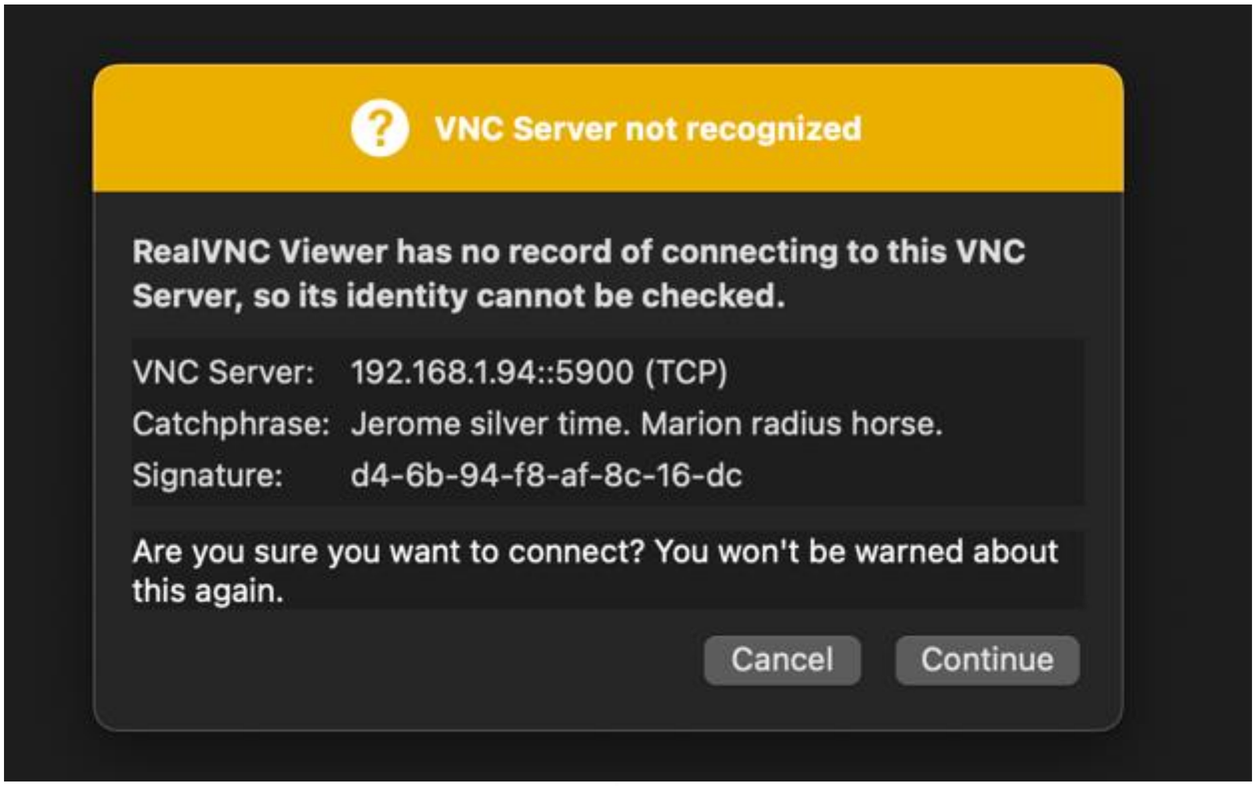
VNC Server: 192.168.1.94

Name: Pi

Click on Ok.



This icon will appear, double click on it.

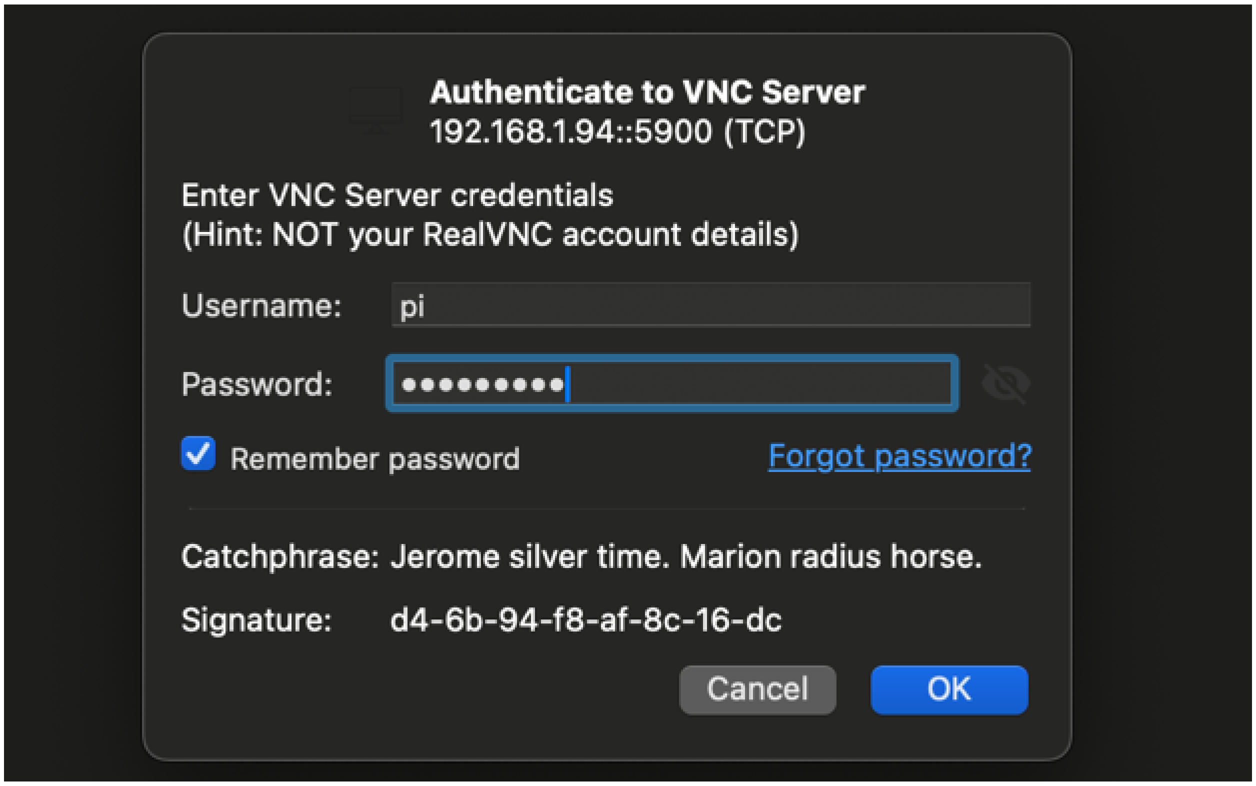

VNC Server not recognized

RealVNC Viewer has no record of connecting to this VNC Server, so its identity cannot be checked.

VNC Server: 192.168.1.94::5900 (TCP)
Catchphrase: Jerome silver time. Marion radius horse.
Signature: d4-6b-94-f8-af-8c-16-dc

Are you sure you want to connect? You won't be warned about this again.

A prompt will appear, click on Continue.


Authenticate to VNC Server
192.168.1.94::5900 (TCP)

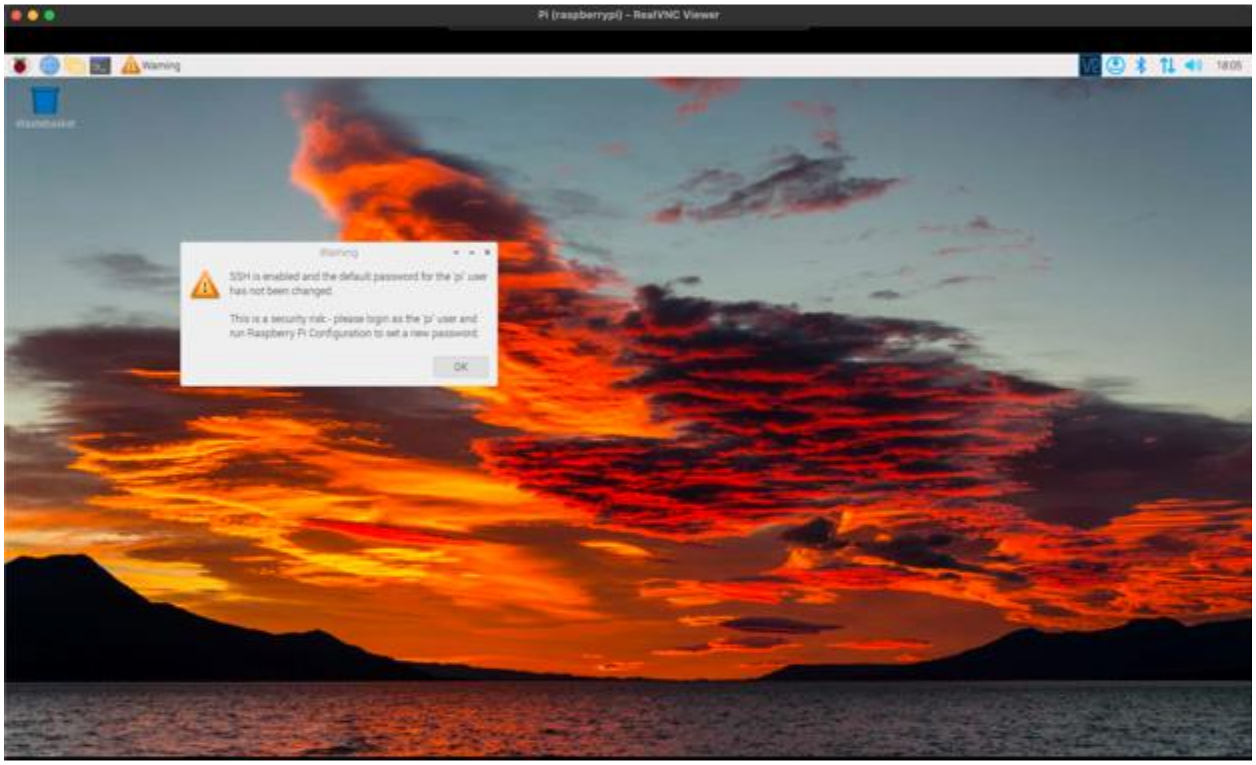
Enter VNC Server credentials
(Hint: NOT your RealVNC account details)

Username: pi
Password: ●●●●●●●●●●

Remember password

Catchphrase: Jerome silver time. Marion radius horse.
Signature: d4-6b-94-f8-af-8c-16-dc

Username is “pi” and the password is “raspberry”. Check the remember password box and click ok.



SSH is enabled and the default password for the 'pi' user has not been changed.

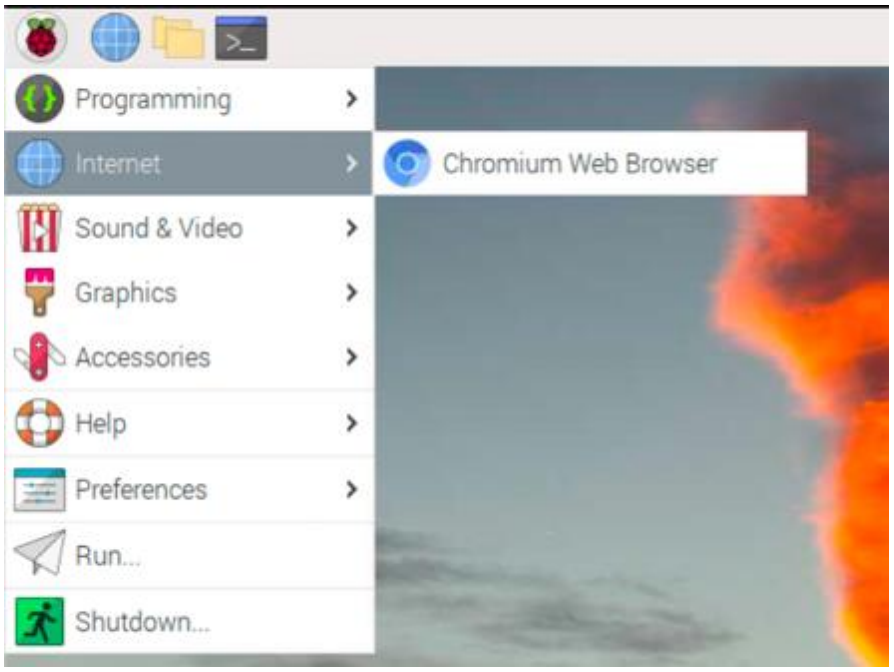
This is a security risk - please login as the 'pi' user and run Raspberry Pi Configuration to set a new password.

The Raspberry pi desktop will appear and we can use raspberry pi now. The warning can be closed by clicking on ok.

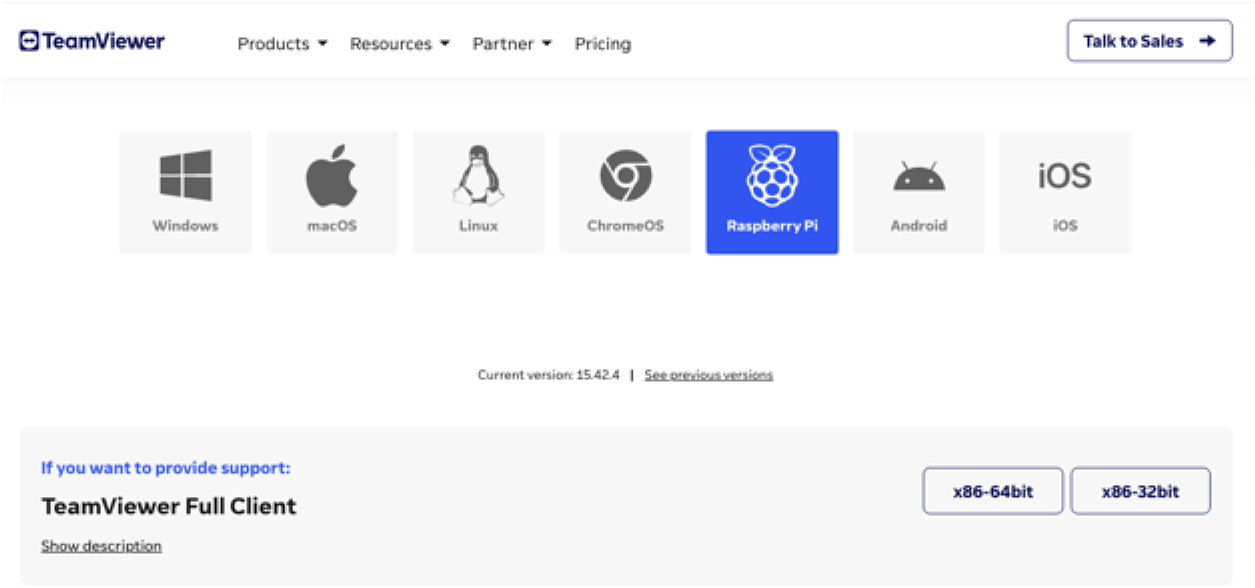
SSH and VNC only works locally and are often time consuming because whenever the IP changes then the whole thing has to be reconfigured. So, we can use third party application like any desk, TeamViewer for easy access. Whenever the device is connected to the internet, no matter from where, they can be configured once to be logged in using the same credentials.

Since this OS is 64 bits, any desk on 64-bit Raspbian OS is not supported yet. Let's install TeamViewer and set it up.

Open up chromium web browser on raspberry pi.



now go to (https://www.teamviewer.com/apac/download/raspberry-pi/)



Current version: 15.42.4 | See previous versions

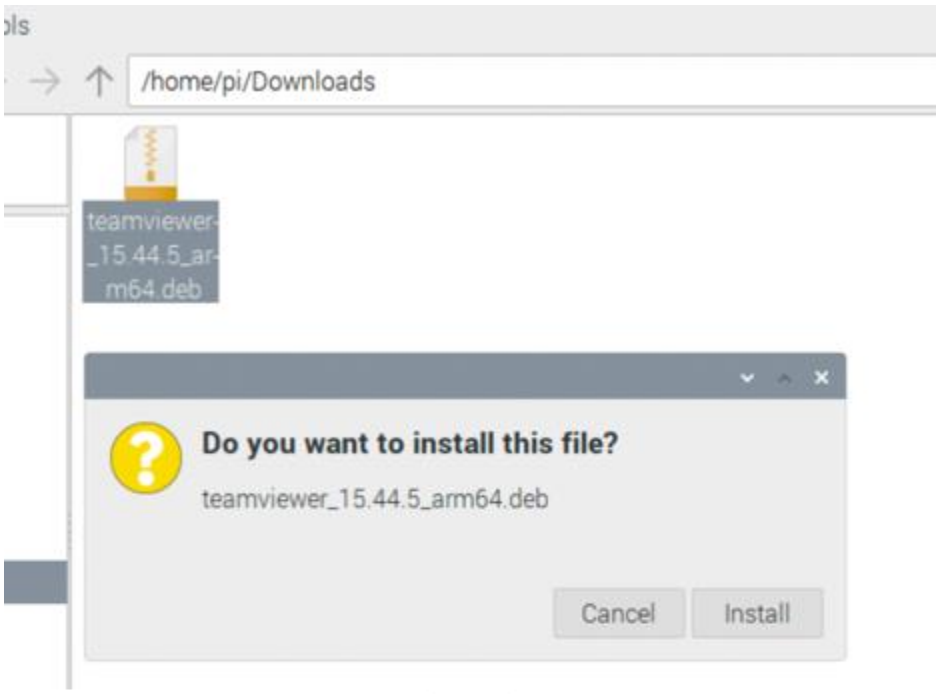
If you want to provide support:

TeamViewer Full Client

Show description

x86-64bit x86-32bit

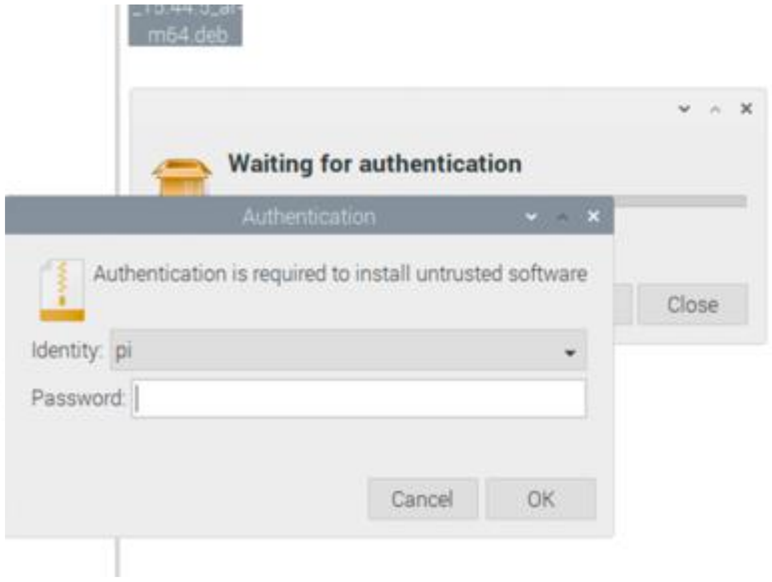
click on x86-64 bit. It will download the Debian package. Now close the browser. Go to Downloads folder and double click on the TeamViewer Debian package.



Do you want to install this file?

teamviewer_15.44.5_arm64.deb

Click on install. A prompt will appear for password.



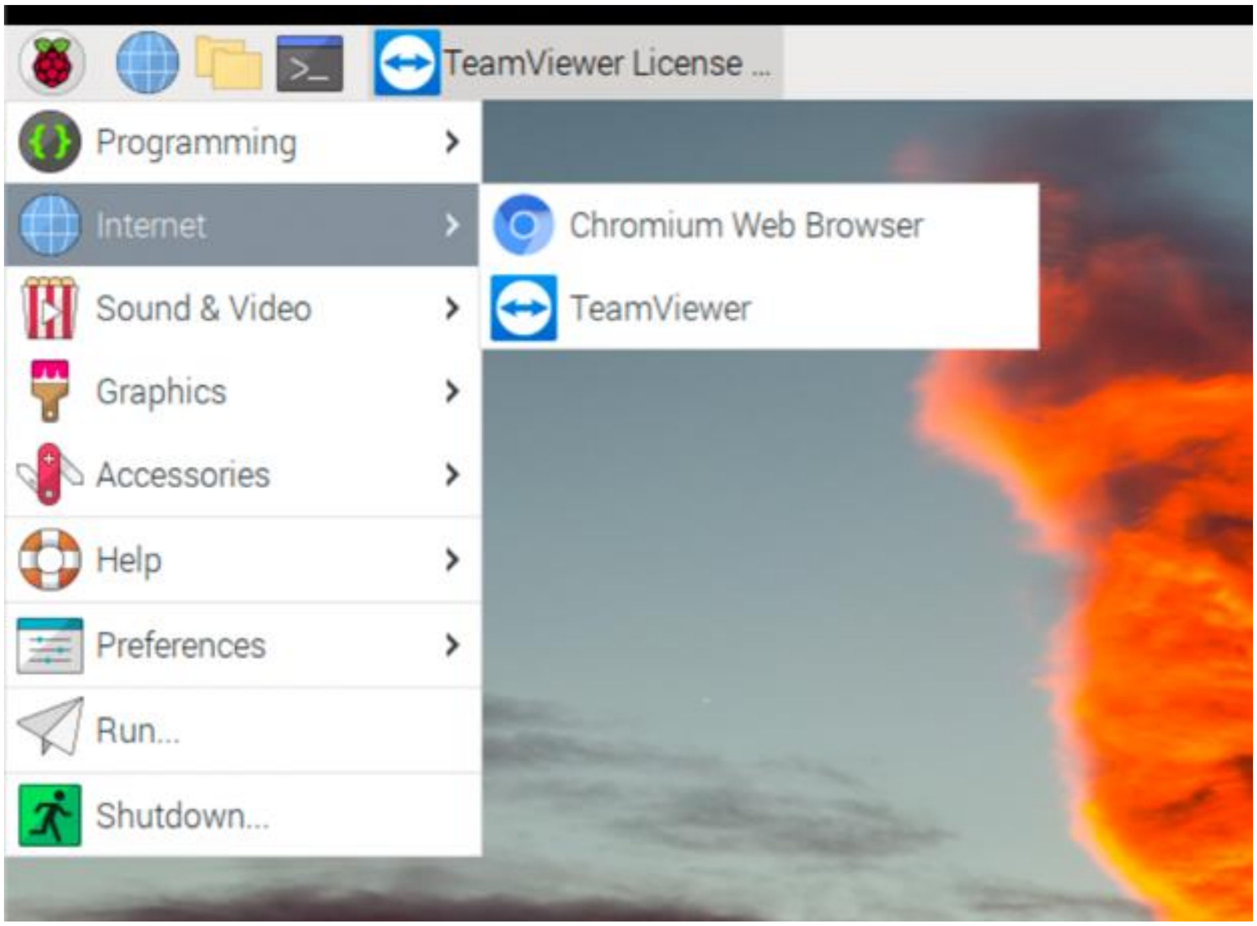
Waiting for authentication

Authentication is required to install untrusted software

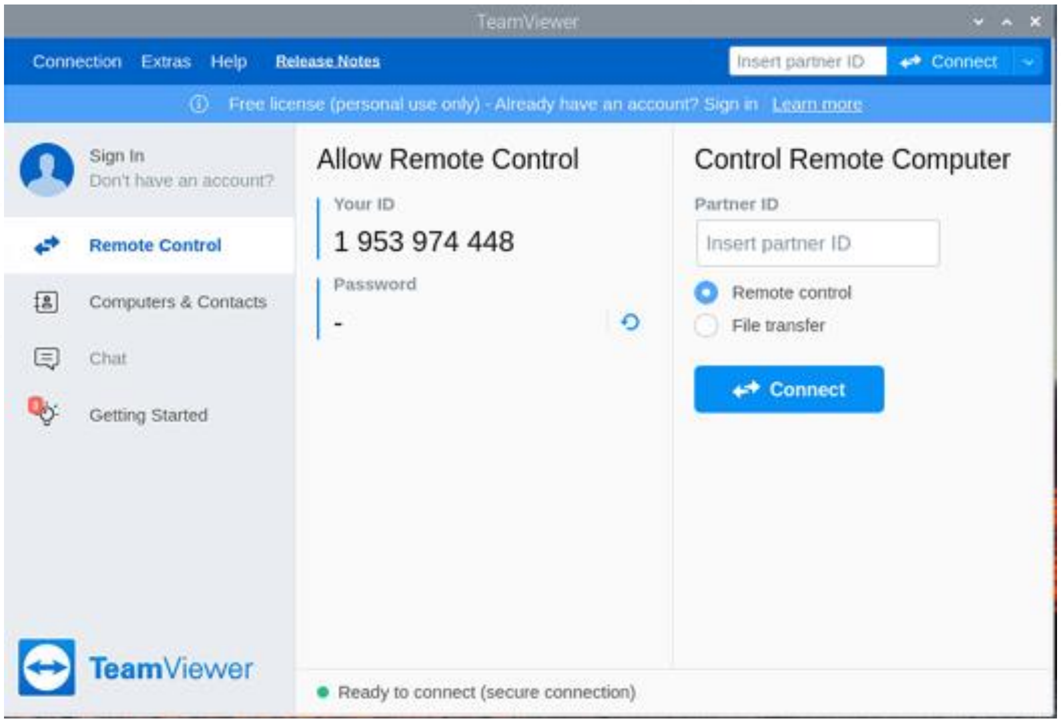
Identity: pi

Password:

The default password is “raspberry”. Then click on OK and the packages will install.



Open team viewer. If prompt appears then check I accept and click on continue.



Free license (personal use only) - Already have an account? Sign in Learn more

Your ID
1 953 974 448

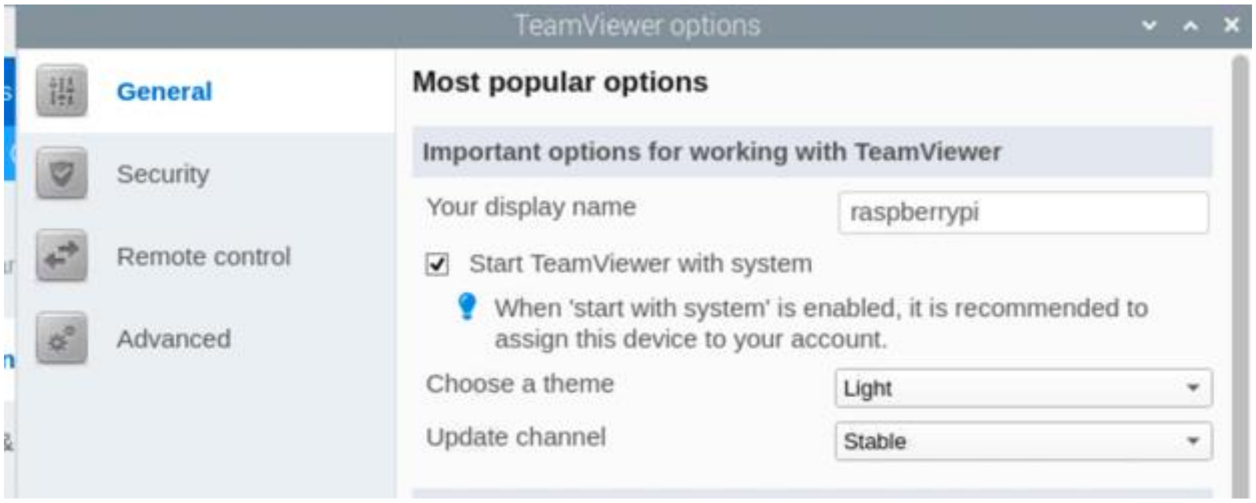
Password
-

Ready to connect (secure connection)

This screen will appear.

Let’s setup it in such a way that the password remains the same and TeamViewer always runs at boot.

Go to extras then options. In general check the start TeamViewer with system.



Most popular options

Important options for working with TeamViewer

Your display name: raspberrypi

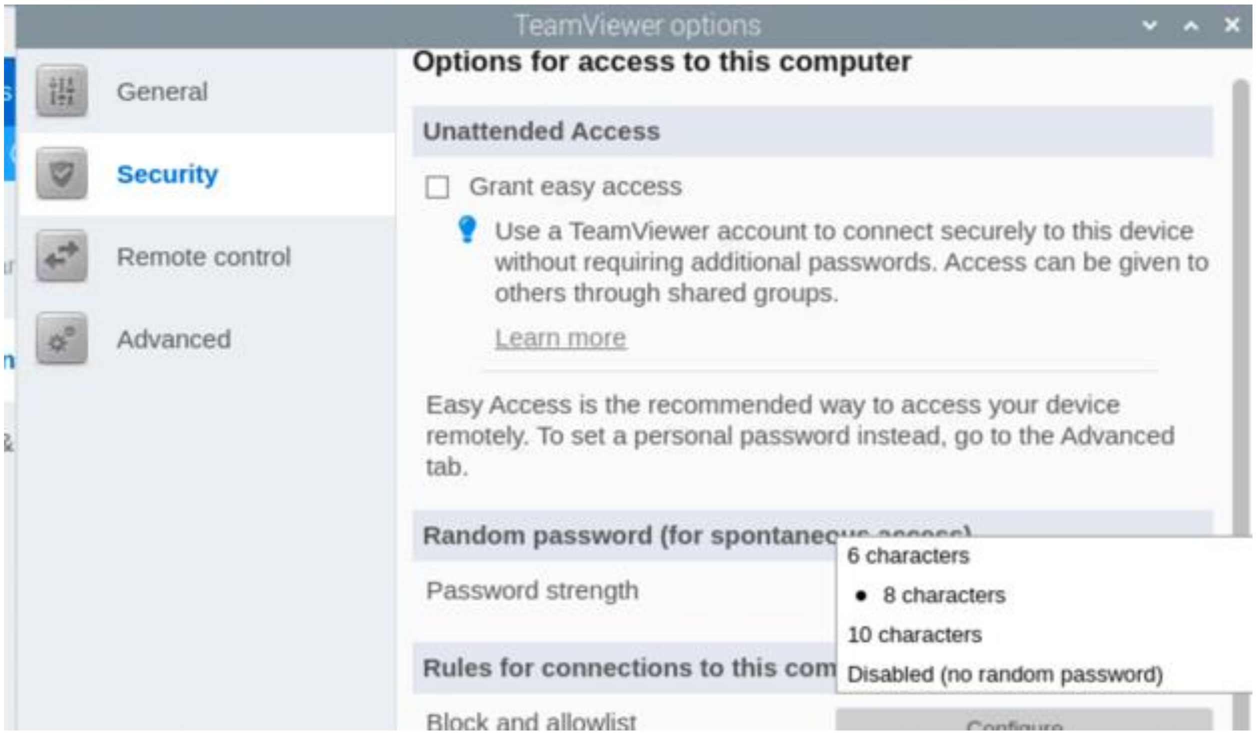
[✓] Start TeamViewer with system

When 'start with system' is enabled, it is recommended to assign this device to your account.

Choose a theme: Light

Update channel: Stable

Go to Security tab and disable random password.



Options for access to this computer

Unattended Access

☐ Grant easy access

Use a TeamViewer account to connect securely to this device without requiring additional passwords. Access can be given to others through shared groups.

Learn more

Easy Access is the recommended way to access your device remotely. To set a personal password instead, go to the Advanced tab.

Random password (for spontaneous access)

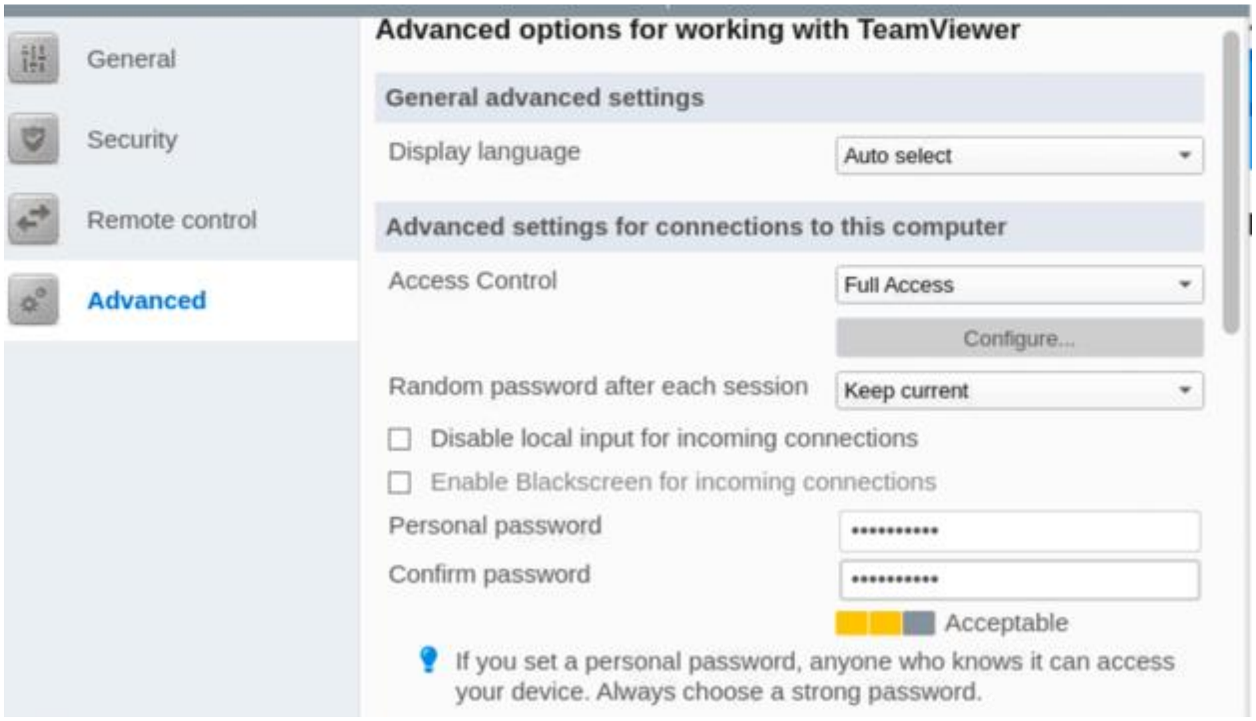
Password strength

6 characters
● 8 characters
10 characters
Disabled (no random password)

Rules for connections to this com

Block and allowlist

Go to advanced and then under Personal password, keep a password of your choice.



Advanced options for working with TeamViewer

General advanced settings

Display language: Auto select

Advanced settings for connections to this computer

Access Control: Full Access

Random password after each session: Keep current

☐ Disable local input for incoming connections

☐ Enable Blackscreen for incoming connections

Personal password: •••••••••••

Confirm password: •••••••••••

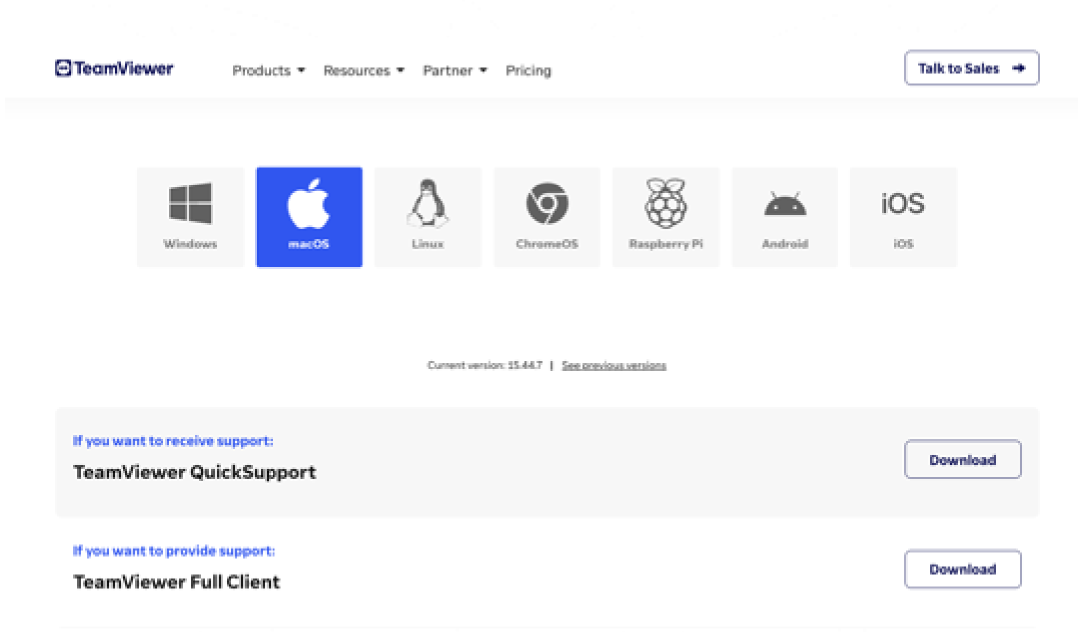
Acceptable

If you set a personal password, anyone who knows it can access your device. Always choose a strong password.

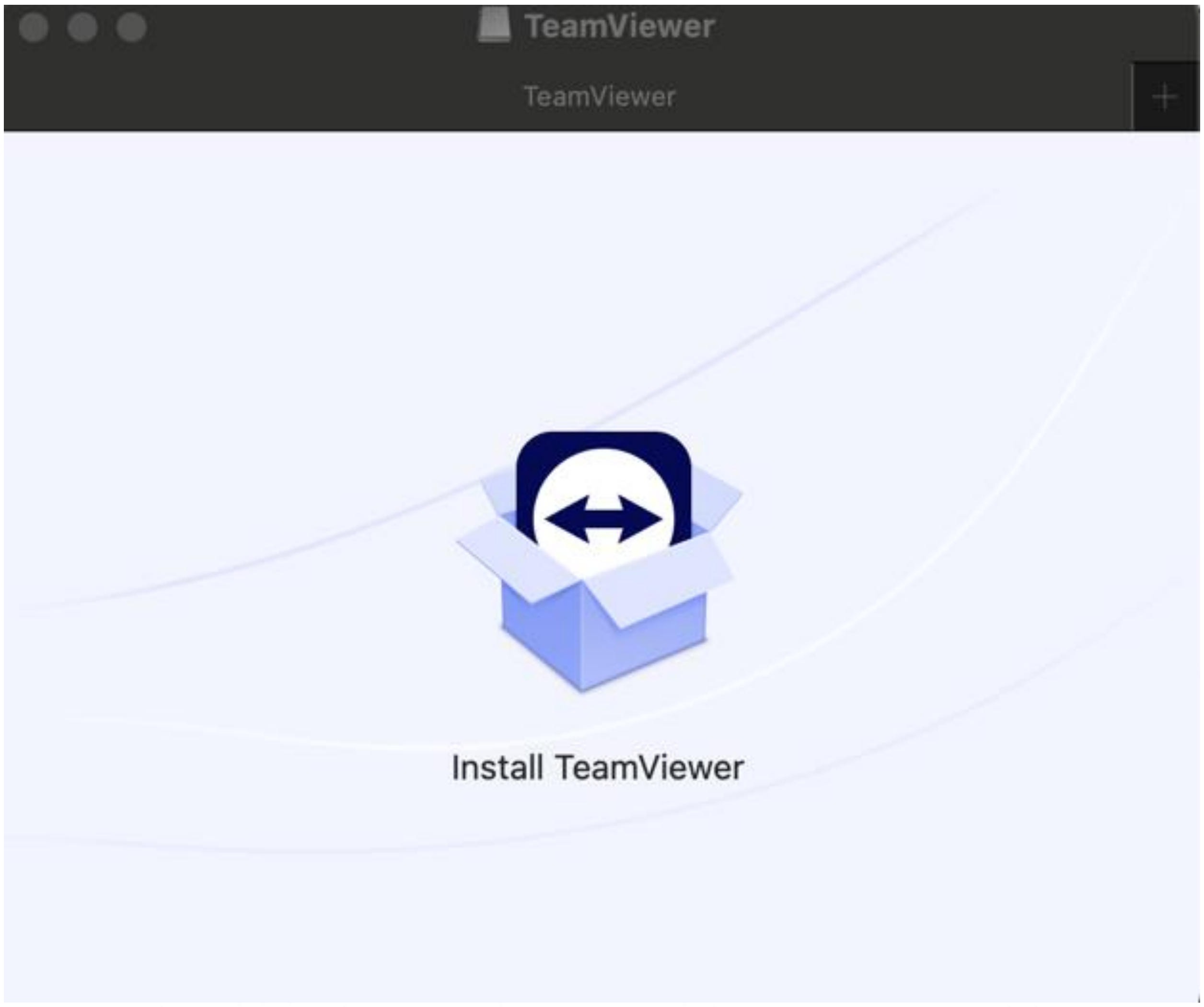
Click on Apply and Ok.

Now we can install team viewer in the mac and try to access the pi using the id and password we got from above steps.

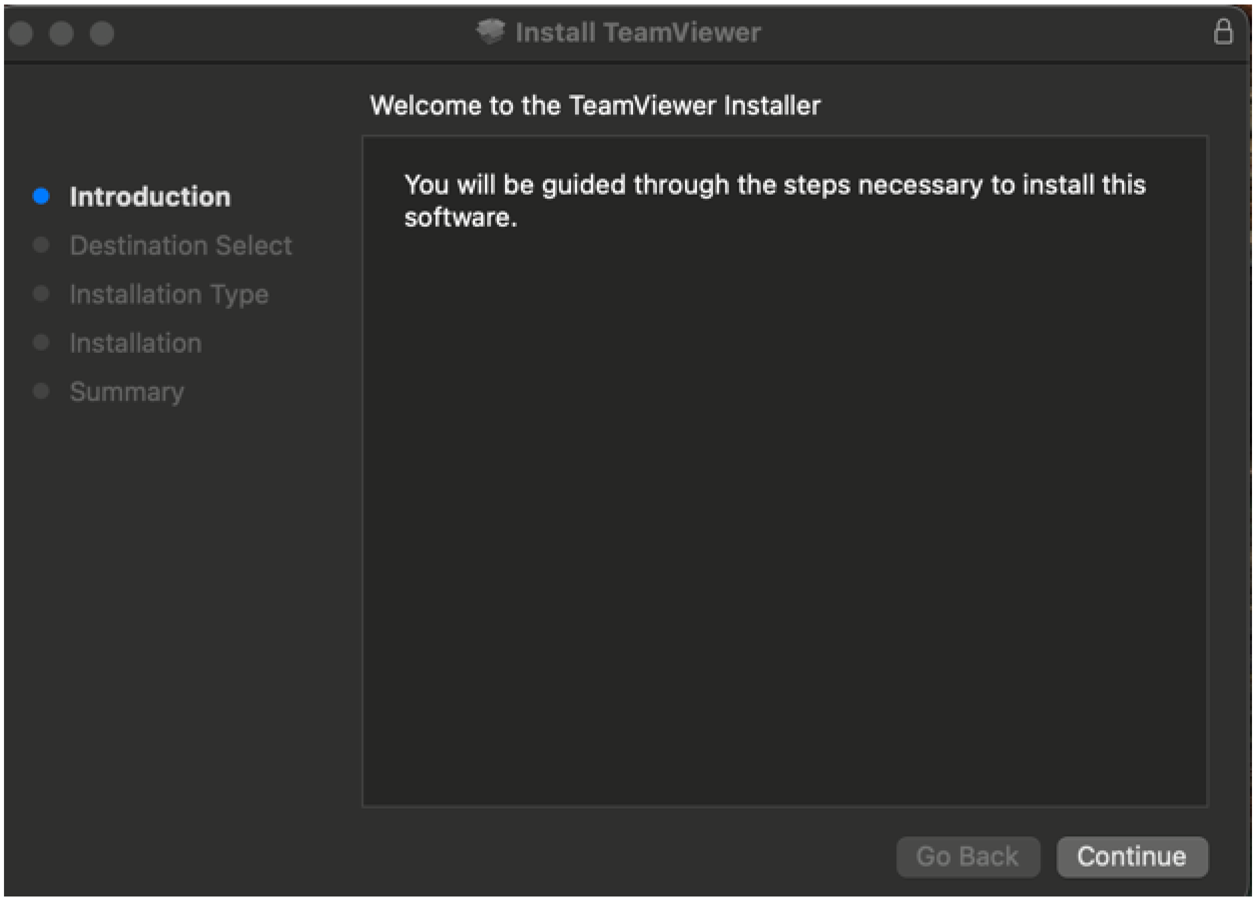
Go to (https://www.teamviewer.com/apac/download/macos/) from browser in mac.



Click on Download in TeamViewer full client. Once download is complete, open the file.



Double click on the icon and a prompt will appear click on Open. Another prompt will appear click on I accept and continue.



You will be guided through the steps necessary to install this software.

The installation process will begin. Follow the instruction and install team viewer. Once the installation is complete, open TeamViewer from launchpad.

Then you can use the Id and password we set from above steps and we can access our pi easily.

Hardware needed:

https://thepihut.com/products/raspberry-pi-4-model-b?variant=20064052740158&src=raspberrypi

https://thepihut.com/products/adafruit-aht20-temperature-humidity-sensor-breakout-board (optional)

https://thepihut.com/products/adafruit-tsl2591-high-dynamic-range-digital-light-sensor (optional)

https://thepihut.com/products/adafruit-mini-gps-pa1010d-uart-and-i2c-stemma-qt (optional)

(NHBS, 2023)

The Ultramic is a fully digital USB ultrasound microphone with integrated digital to analog converter and an option of 192 kHz, 200kHz or 250kHz sampling rate. The USB 2.0 port allows for easy connection to your PC or MAC and no driver installation is required.

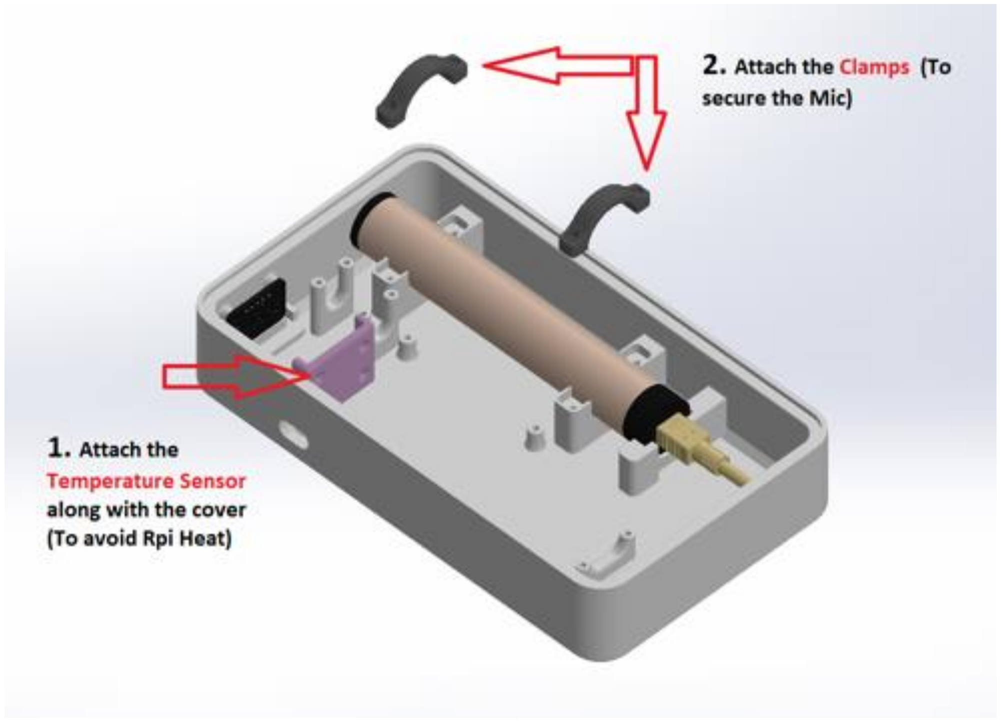
Compact and versatile the Ultramic is a great option for use in the field or laboratory. Whilst most often used to record the ultrasonic signals of both bats and rodents it is equally useful for detection of high-frequency noises emitted by LCD screens, turbines and power adaptors.

Specification

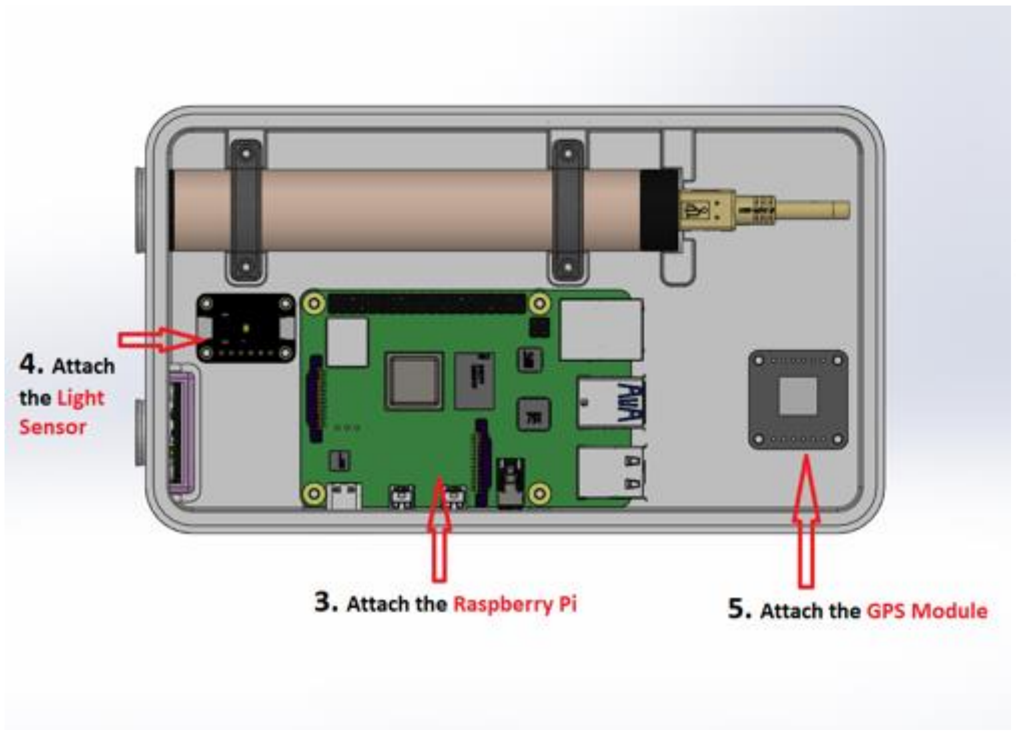
* 192K, 200K or 250K sampling per second.
* True 16 bit resolution.
* Frequency range up to 100kHz - 125kHz.
* MEMS high sensitivity Surface Mount Wide-band Ultrasonic Acoustic Sensor.
* High quality and low noise analogue amplification.
* USB full speed port with a mini B USB connector.
* 32 bit 80MHz integrated microcontroller.
* Dimensions: 130mm length x 20mm diameter.

Available on https://www.nhbs.com/ultramic-usb-ultrasound-microphone

Now let's connect all hardware together.

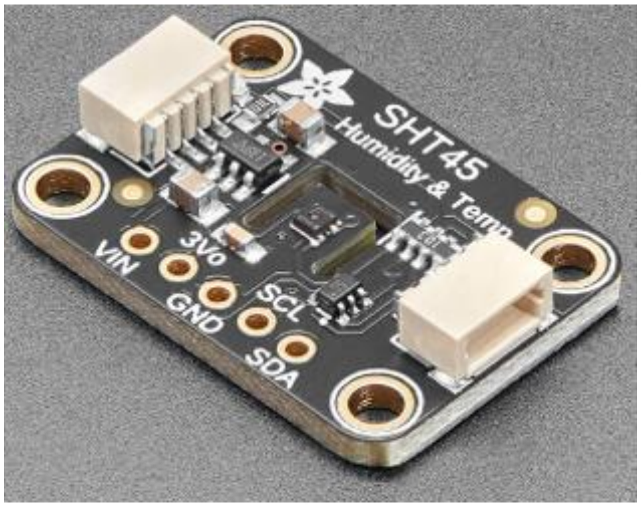

2. Attach the Clamps (To secure the Mic)
1. Attach the Temperature Sensor along with the cover (To avoid Rpi Heat)

please install temperature sensor and the Microphone as depicted on the image above.



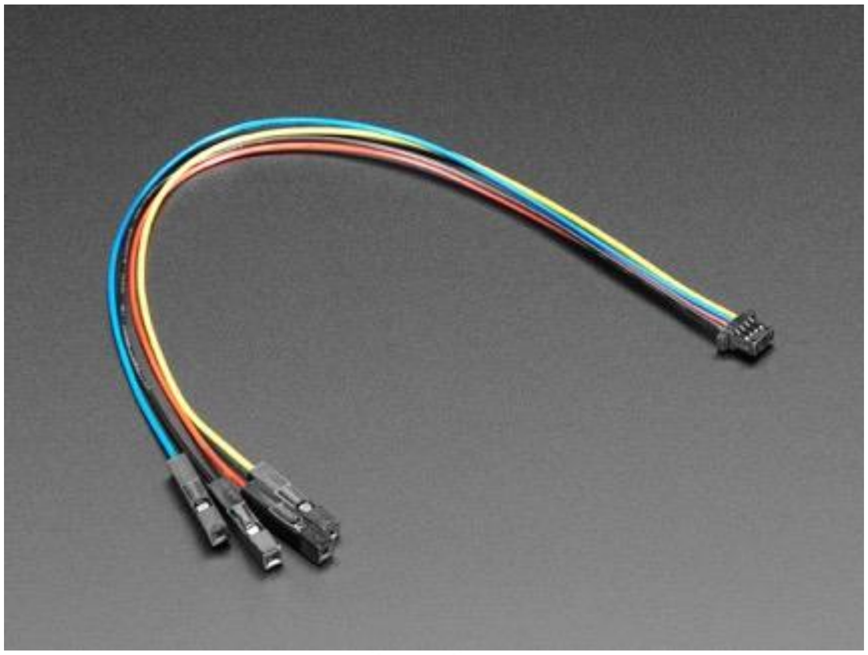
Then we can place light sensor, GPS module and Raspberry Pi inside the housing. Power supply is connected to Pi via a cut out from the side. The microphone is connected on the USB port.

All three sensors are STEMMA QT / Qwiic compatible. The sensor can be wired from one to another in a daisy chain.



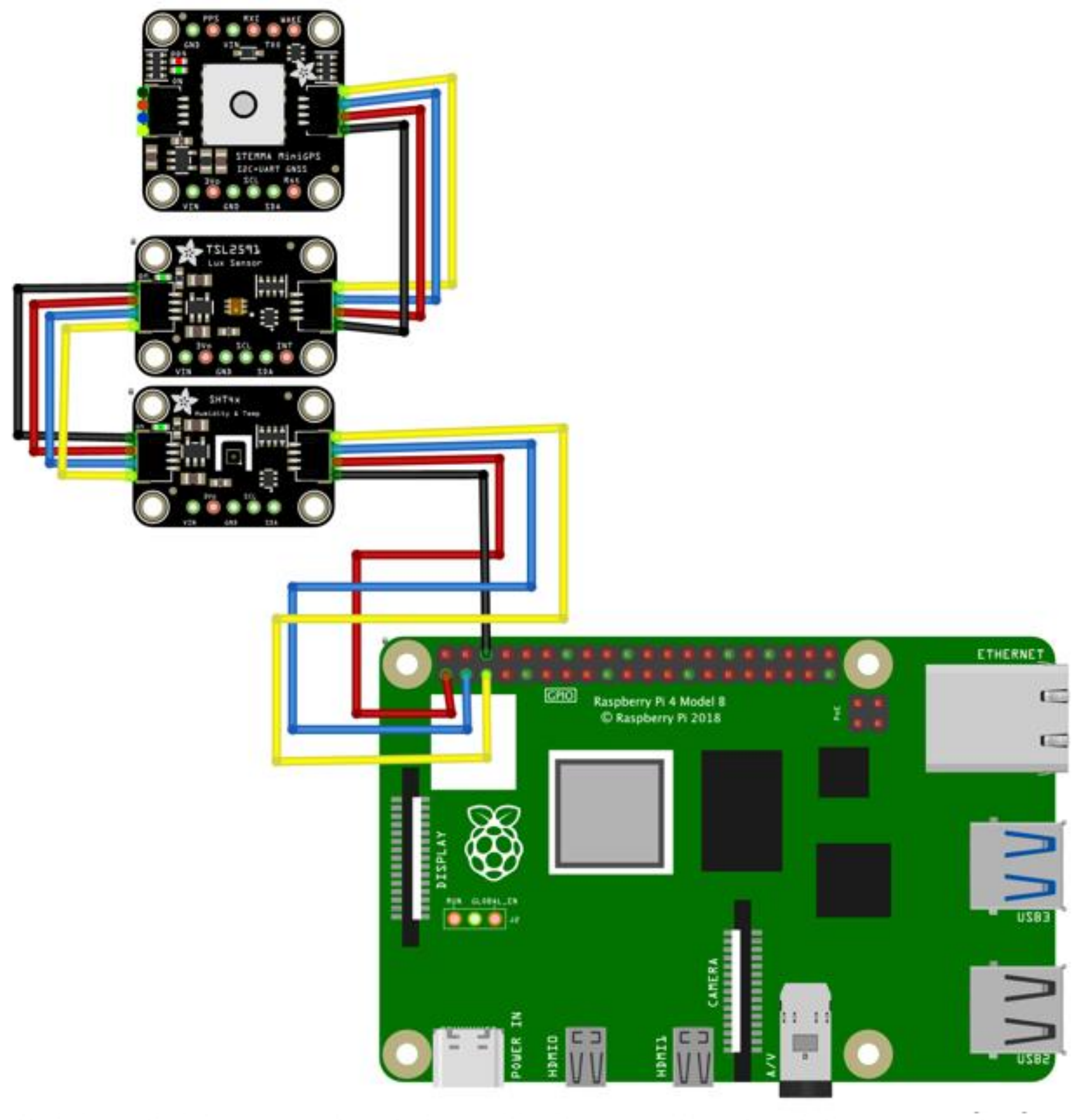
The white connectors on the side in the above SHT45 temperature and humidity sensors are STEMMA QT / Qwiic connectors. Qt cable can be directly connected on those ones. Please use

STEMMA QT / Qwiic JST SH 4-pin Cable with Premium Female Sockets as in the pictures



below.

The connector side is connected to the sensor and the female sockets are connected to the raspberry pi's GPIO pin.



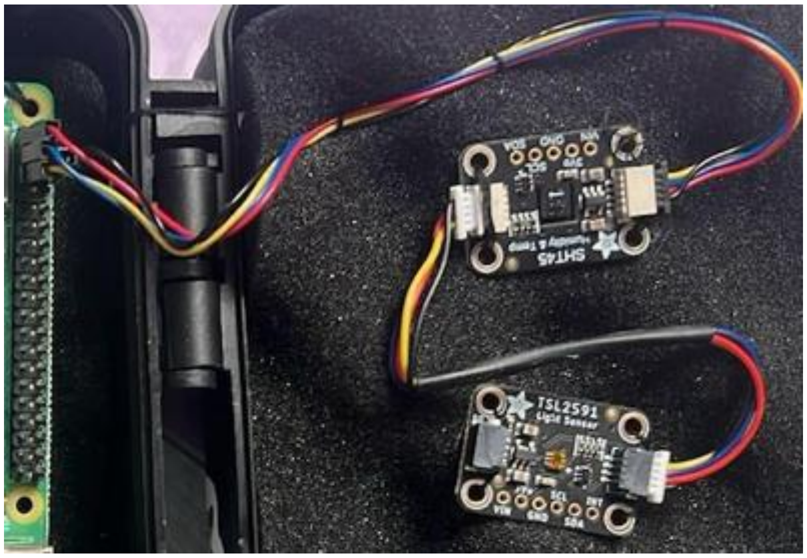
The female socket cables are color coded so they can be easily used. The red-wire is VCC, black wire is GND, blue is SDA and yellow is SCL.

Red wire:	pin 1 of raspberry pi (3v3)
Black wire:	pin 6 of raspberry pi (GND)
Blue wire:	pin 3 of raspberry pi (GPIO 2) (SDA)

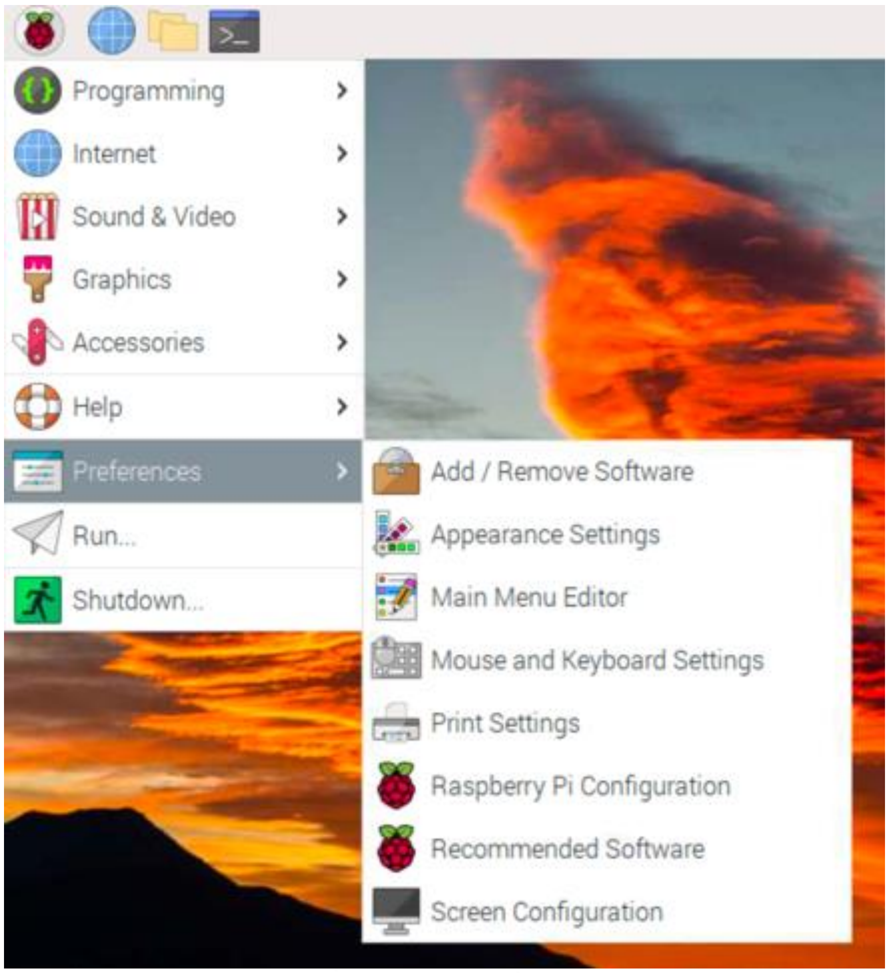
Yellow wire: pin 5 of raspberry pi (GPIO 3) (SCL)

After that we use QT to QT cable (STEMMA QT / Qwiic JST SH 4-Pin Cable - 200mm Long) to connect between sensors as shown in figure below.

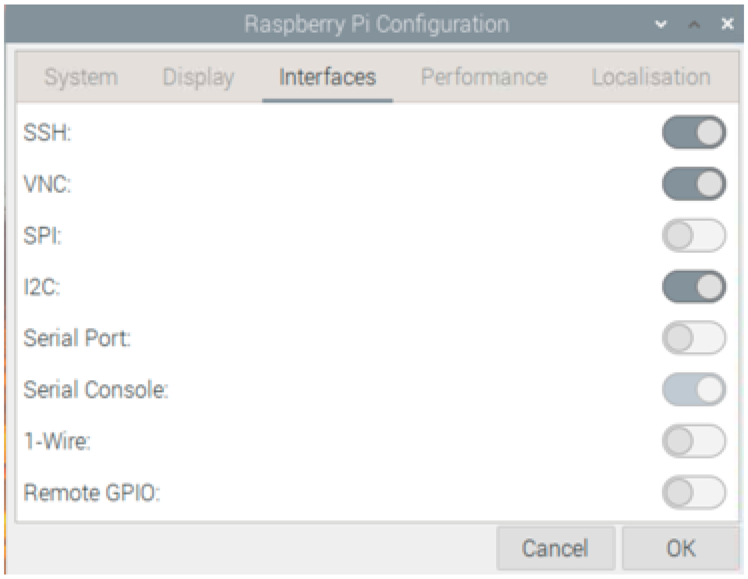
The final result should look like this. The pi



Likewise, GPS module is also connected in similar fashion. Now all the hardware is connected and assembled. Let’s start programming them to test and build it completely. All the sensors are I2C. Let’s enable I2C on raspberry pi.

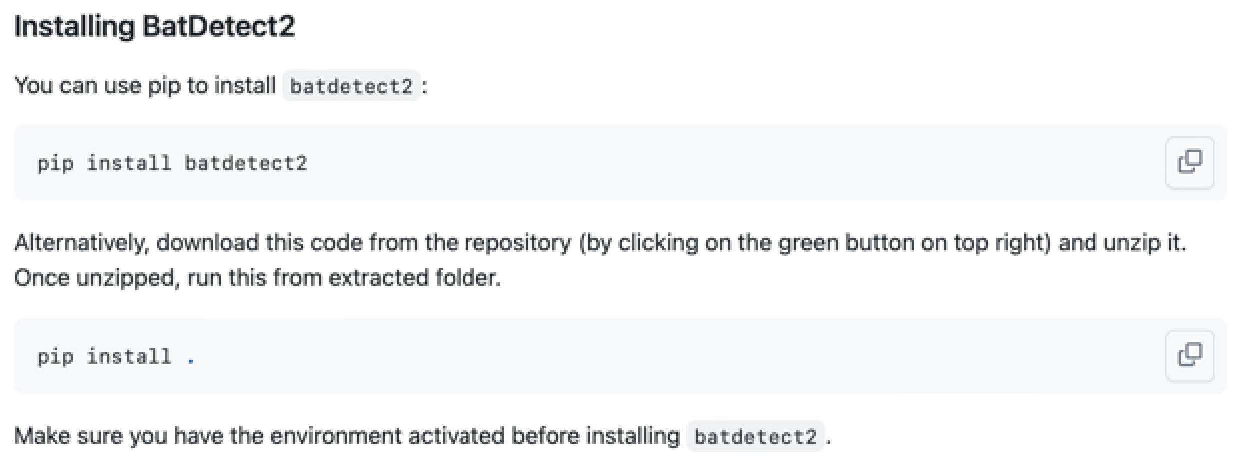


Go to application menu, then preferences then Raspberry Pi configuration.



SSH: on
VNC: on
SPI: off
I2C: on
Serial Port: off
Serial Console: on
1-Wire: off
Remote GPIO: off

Install BatDetect2 Model from https://github.com/macaodha/batdetect2



Installing BatDetect2

You can use pip to install `batdetect2`:

```
pip install batdetect2
```

Alternatively, download this code from the repository (by clicking on the green button on top right) and unzip it. Once unzipped, run this from extracted folder.

```
pip install .
```

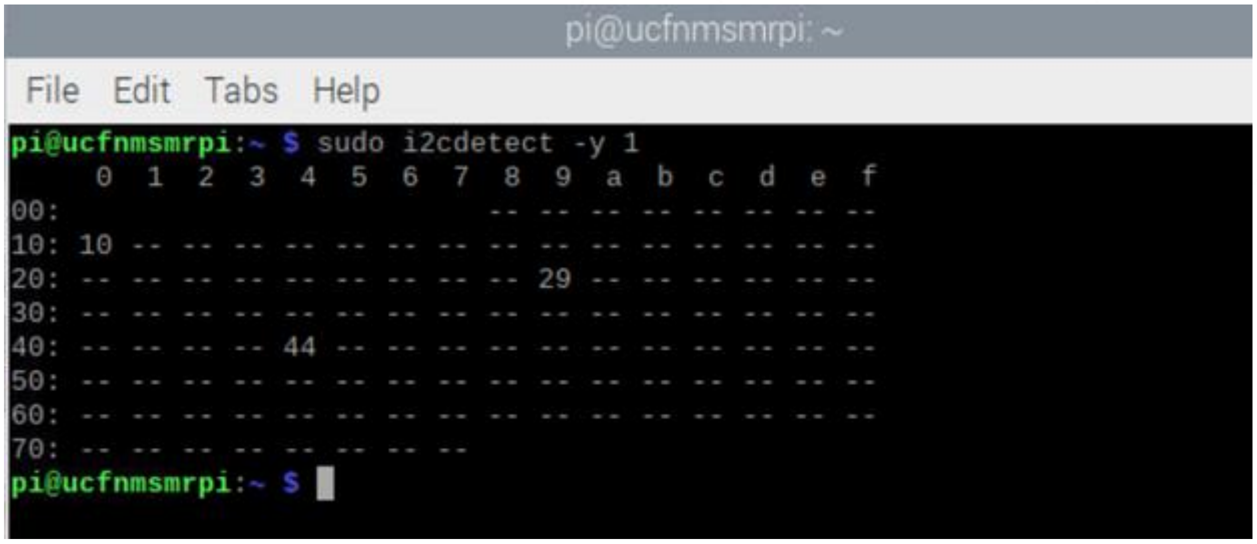
Make sure you have the environment activated before installing `batdetect2`.

Go to Interfaces and turn on I2C. Now click on OK.

Now in order to verify all the sensors are connected go to terminal and run this command

sudo i2cdetect -y 1

and press enter.



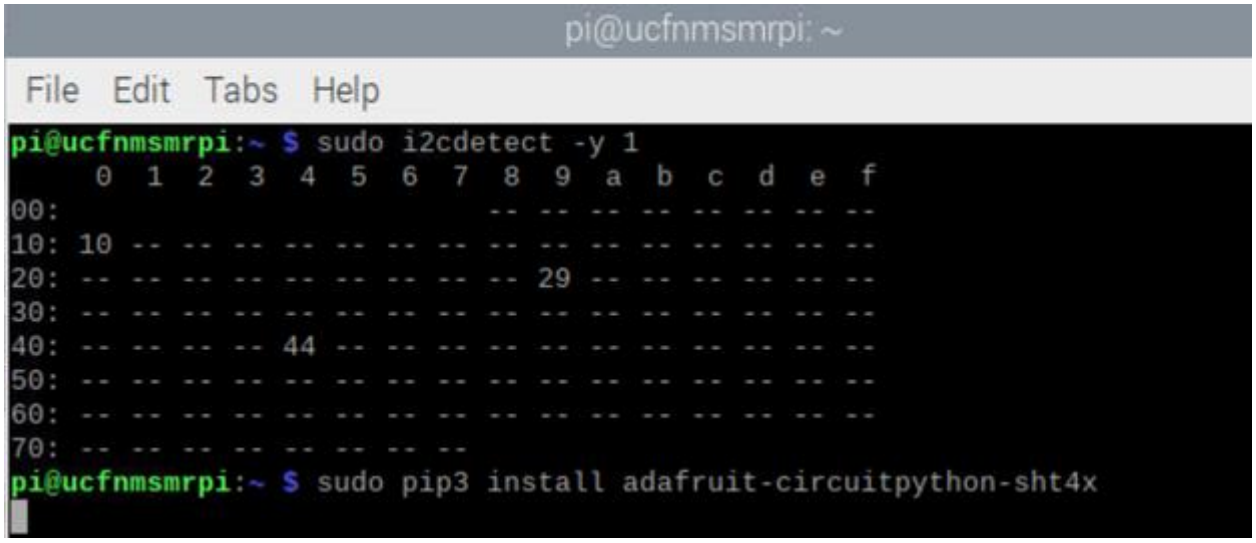
```
pi@ucfnmsmrpi:~ $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: 10 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- 29 -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- 44 -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@ucfnmsmrpi:~ $
```

So, we got three I2C address and that means 3 I2C devices are connected.

Now let's install libraries for all three sensors.

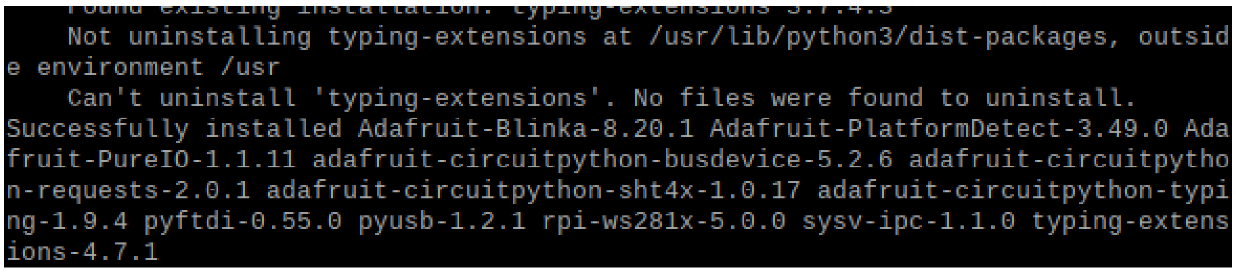
In order to install library for SHT45 sensor, run the following command.

sudo pip3 install adafruit-circuitpython-sht4x



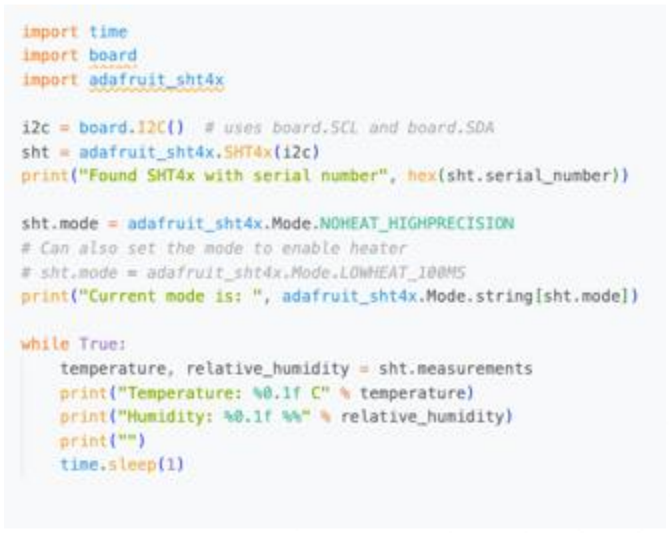
```
pi@ucfnmsmrpi:~ $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: 10 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- 29 -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- 44 -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@ucfnmsmrpi:~ $ sudo pip3 install adafruit-circuitpython-sht4x
```

once the library is installed you should be able to see this screen.



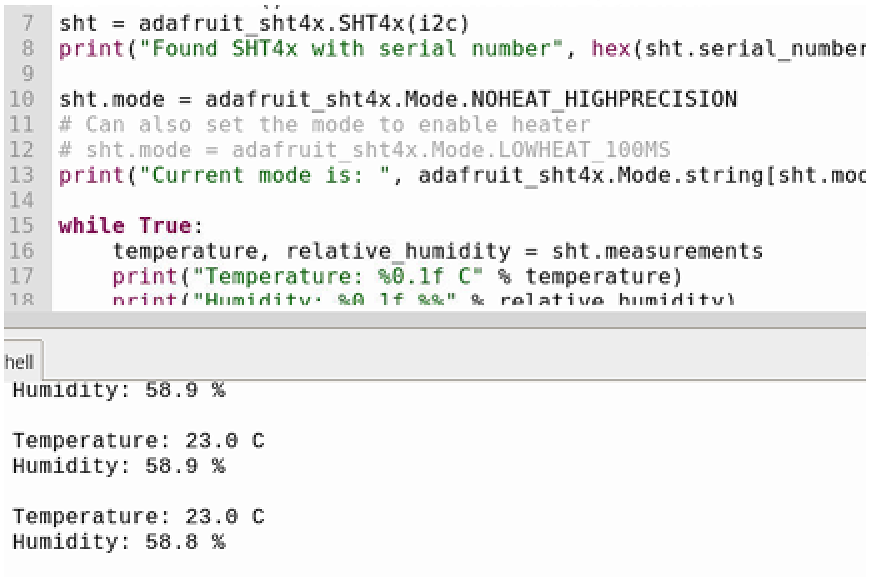
```
    Not uninstalling typing-extensions at /usr/lib/python3/dist-packages, outsid
e environment /usr
    Can't uninstall 'typing-extensions'. No files were found to uninstall.
Successfully installed Adafruit-Blinka-8.20.1 Adafruit-PlatformDetect-3.49.0 Ada
fruit-PureIO-1.1.11 adafruit-circuitpython-busdevice-5.2.6 adafruit-circuitpytho
n-requests-2.0.1 adafruit-circuitpython-sht4x-1.0.17 adafruit-circuitpython-typi
ng-1.9.4 pyftdi-0.55.0 pyusb-1.2.1 rpi-ws281x-5.0.0 sysv-ipc-1.1.0 typing-extens
ions-4.7.1
```

Now let's run the test code and check the output. Let's create a python file in desktop and paste this code there are run.



```
import time
import board
import adafruit_sht4x

i2c = board.I2C()  # uses board.SCL and board.SDA
sht = adafruit_sht4x.SHT4x(i2c)
print("Found SHT4x with serial number", hex(sht.serial_number))

sht.mode = adafruit_sht4x.Mode.NOHEAT_HIGHPRECISION
# Can also set the mode to enable heater
# sht.mode = adafruit_sht4x.Mode.LOWHEAT_100MS
print("Current mode is: ", adafruit_sht4x.Mode.string[sht.mode])

while True:
    temperature, relative_humidity = sht.measurements
    print("Temperature: %0.1f C" % temperature)
    print("Humidity: %0.1f %%" % relative_humidity)
    print("")
    time.sleep(1)
```

Let’s run the code and check the output.

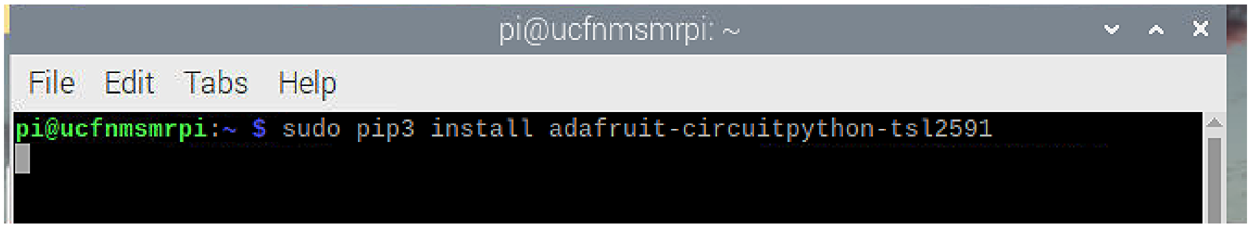


```
sht = adafruit_sht4x.SHT4x(i2c)
print("Found SHT4x with serial number", hex(sht.serial_number

sht.mode = adafruit_sht4x.Mode.NOHEAT_HIGHPRECISION
# Can also set the mode to enable heater
# sht.mode = adafruit_sht4x.Mode.LOWHEAT_100MS
print("Current mode is: ", adafruit_sht4x.Mode.string[sht.moc

while True:
    temperature, relative_humidity = sht.measurements
    print("Temperature: %0.1f C" % temperature)
    print("Humidity: %0.1f %%" % relative_humidity)
```

```
Humidity: 58.9 %

Temperature: 23.0 C
Humidity: 58.9 %

Temperature: 23.0 C
Humidity: 58.8 %
```

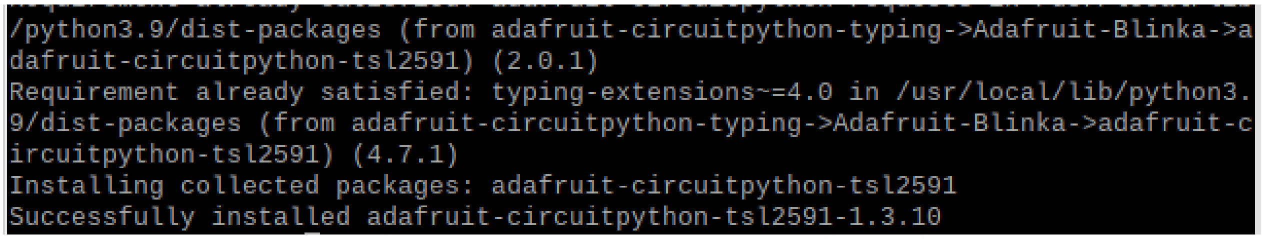
Temperature and humidity were obtained in the serial monitor.

Now let’s install library for light intensity sensor. In order to install library for TSL2591 sensor, run the following command sudo pip3 install adafruit-circuitpython-tsl2591



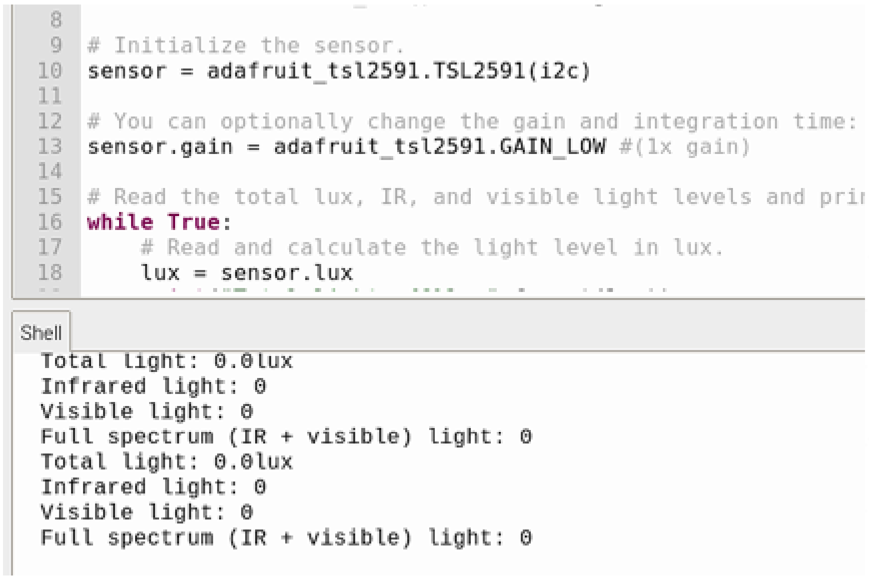
```
pi@ucfnmsmrpi:~ $ sudo pip3 install adafruit-circuitpython-tsl2591
```

Press enter and once the library is installed, you should see the following.



```
/python3.9/dist-packages (from adafruit-circuitpython-typing->Adafruit-Blinka->a
dafruit-circuitpython-tsl2591) (2.0.1)
Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/python3.
9/dist-packages (from adafruit-circuitpython-typing->Adafruit-Blinka->adafruit-c
ircuitpython-tsl2591) (4.7.1)
Installing collected packages: adafruit-circuitpython-tsl2591
Successfully installed adafruit-circuitpython-tsl2591-1.3.10
```

Now let's test the sensor's data.

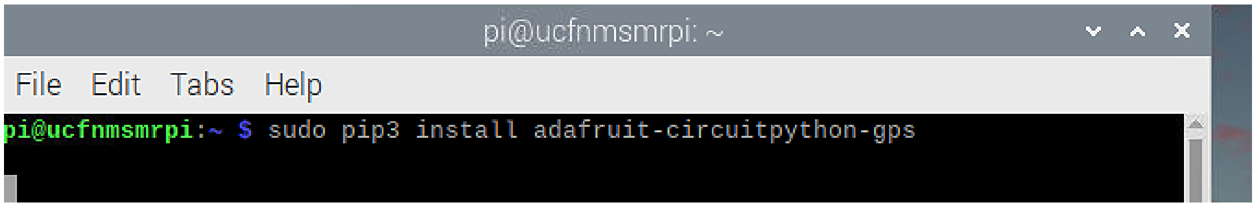


```python

# Initialize the sensor.
sensor = adafruit_tsl2591.TSL2591(i2c)

# You can optionally change the gain and integration time:
sensor.gain = adafruit_tsl2591.GAIN_LOW #(1x gain)

# Read the total lux, IR, and visible light levels and pri
while True:
    # Read and calculate the light level in lux.
    lux = sensor.lux
```

```
Total light: 0.0lux
Infrared light: 0
Visible light: 0
Full spectrum (IR + visible) light: 0
Total light: 0.0lux
Infrared light: 0
Visible light: 0
Full spectrum (IR + visible) light: 0
```

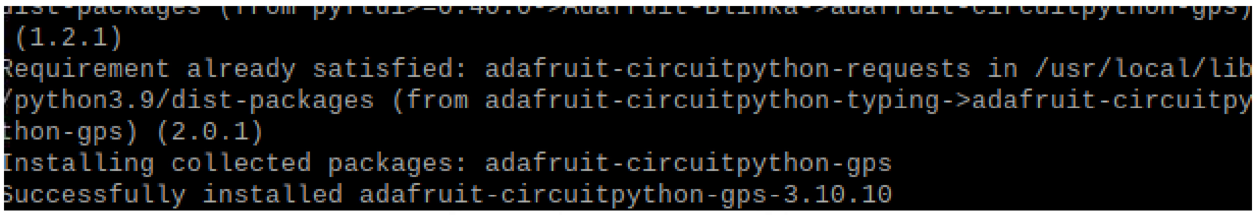
Sensor was kept at dark and the data was 0.

Now let's install library for GPS module. In order to install library for GPS module, run the following command sudo pip3 install adafruit-circuitpython-gps



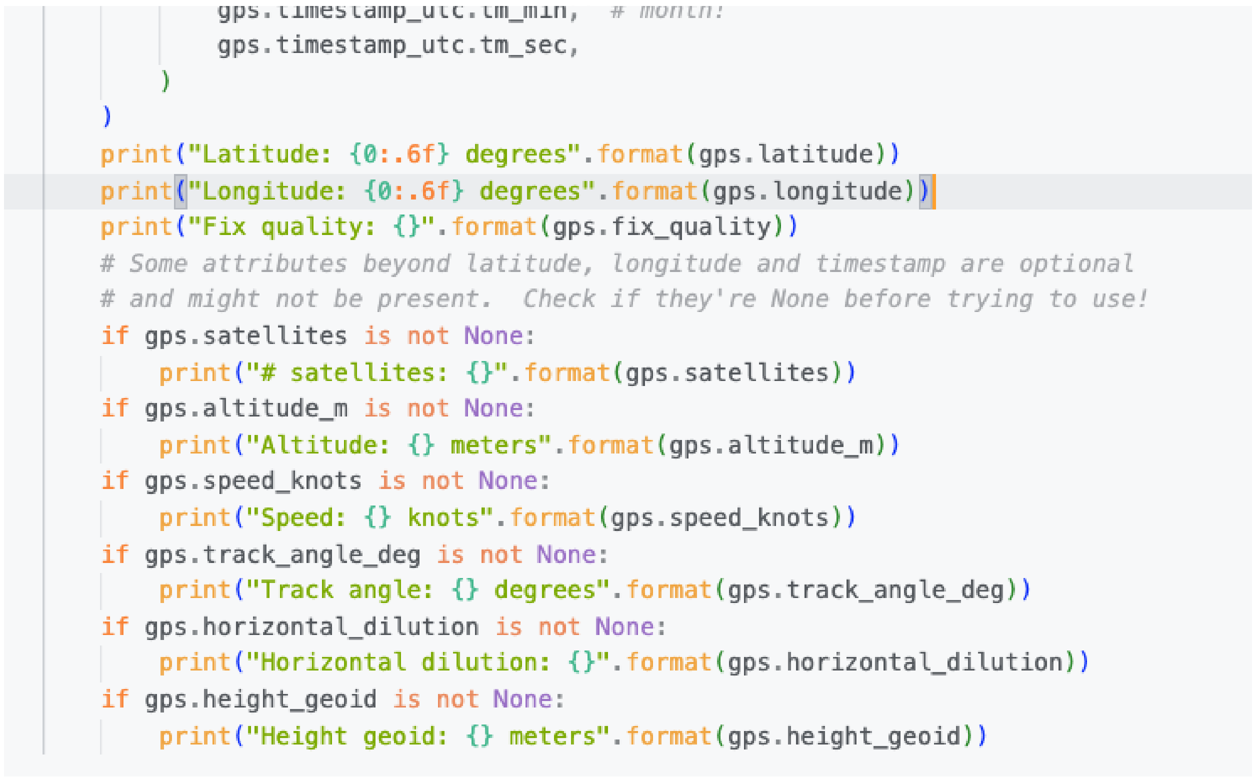
```
pi@ucfnmsmrpi:~ $ sudo pip3 install adafruit-circuitpython-gps
```

Once the command is completed you should see the screen below.



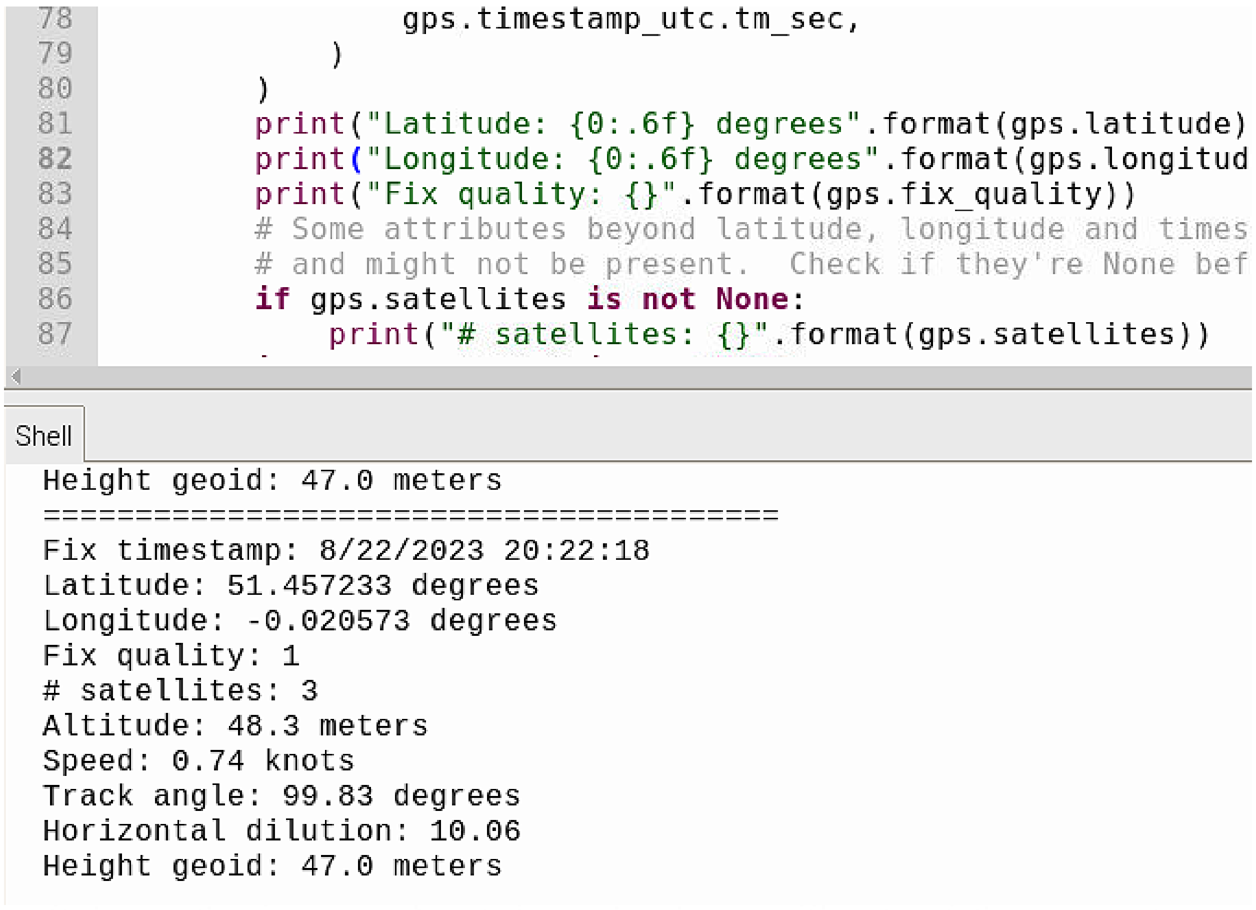
```
 (1.2.1)
Requirement already satisfied: adafruit-circuitpython-requests in /usr/local/lib
/python3.9/dist-packages (from adafruit-circuitpython-typing->adafruit-circuitpy
thon-gps) (2.0.1)
Installing collected packages: adafruit-circuitpython-gps
Successfully installed adafruit-circuitpython-gps-3.10.10
```

Now lets test the test code and see the data.



```
            gps.timestamp_utc.tm_sec,
        )
    )
    print("Latitude: {0:.6f} degrees".format(gps.latitude))
    print("Longitude: {0:.6f} degrees".format(gps.longitude))
    print("Fix quality: {}".format(gps.fix_quality))
    # Some attributes beyond latitude, longitude and timestamp are optional
    # and might not be present.  Check if they're None before trying to use!
    if gps.satellites is not None:
        print("# satellites: {}".format(gps.satellites))
    if gps.altitude_m is not None:
        print("Altitude: {} meters".format(gps.altitude_m))
    if gps.speed_knots is not None:
        print("Speed: {} knots".format(gps.speed_knots))
    if gps.track_angle_deg is not None:
        print("Track angle: {} degrees".format(gps.track_angle_deg))
    if gps.horizontal_dilution is not None:
        print("Horizontal dilution: {}".format(gps.horizontal_dilution))
    if gps.height_geoid is not None:
        print("Height geoid: {} meters".format(gps.height_geoid))
```

The above is just a snippet of the test script.



```
                gps.timestamp_utc.tm_sec,
            )
        )
        print("Latitude: {0:.6f} degrees".format(gps.latitude)
        print("Longitude: {0:.6f} degrees".format(gps.longitud
        print("Fix quality: {}".format(gps.fix_quality))
        # Some attributes beyond latitude, longitude and times
        # and might not be present.  Check if they're None bef
        if gps.satellites is not None:
            print("# satellites: {}".format(gps.satellites))
```

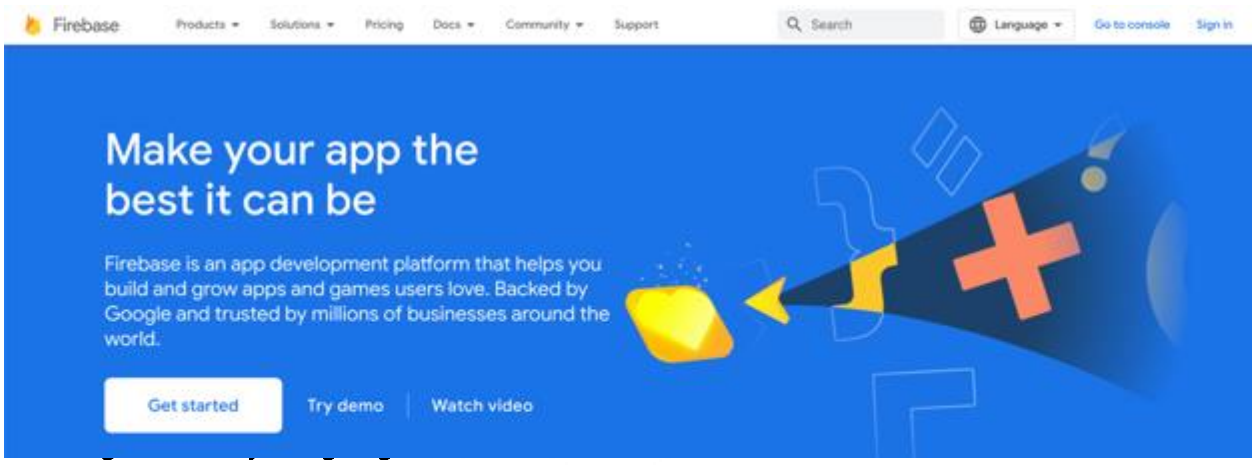
Shell

```
Height geoid: 47.0 meters
========================================
Fix timestamp: 8/22/2023 20:22:18
Latitude: 51.457233 degrees
Longitude: -0.020573 degrees
Fix quality: 1
# satellites: 3
Altitude: 48.3 meters
Speed: 0.74 knots
Track angle: 99.83 degrees
Horizontal dilution: 10.06
Height geoid: 47.0 meters
```

we could get the data of the GPS coordinates and all other data.

Now let’s test firebase.

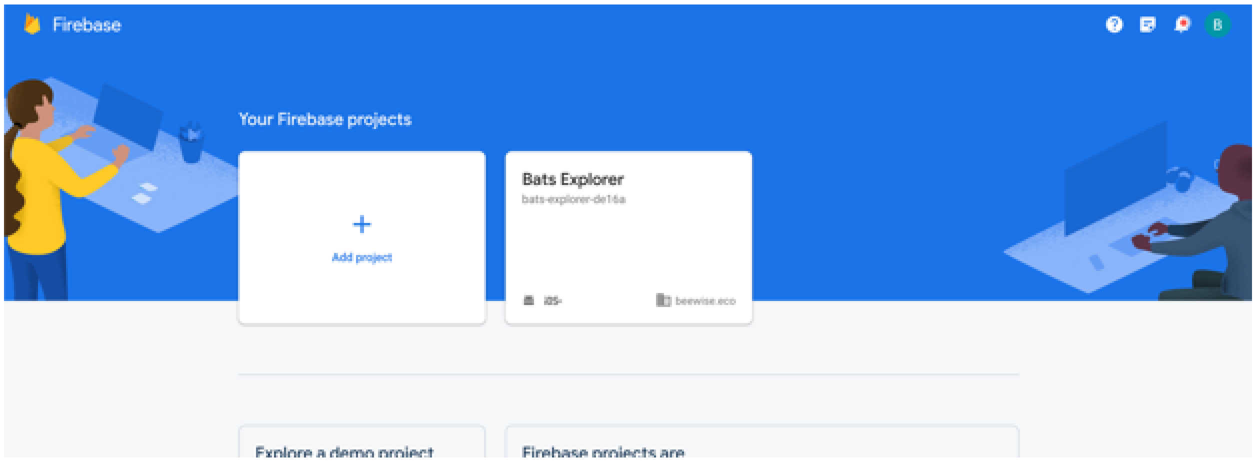
In order to run firebase in raspberry pi, we need to setup firebase account using our gmail account. For that go to(https://firebase.google.com/)



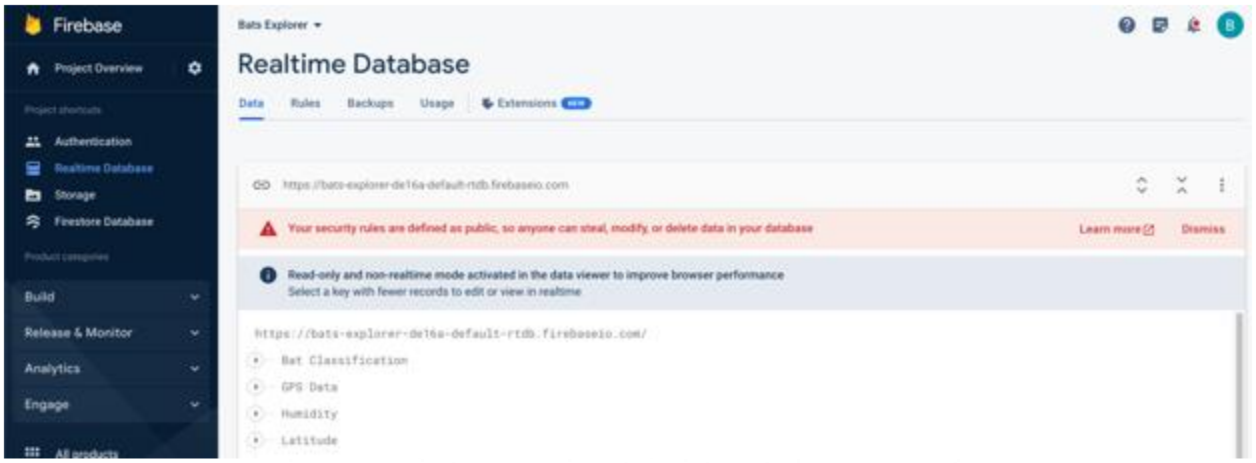
Make your app the best it can be

Firebase is an app development platform that helps you build and grow apps and games users love. Backed by Google and trusted by millions of businesses around the world.

Once you sign in press on go to console.



Add project to create a new one. Once that is done click on the project. And setup a Realtime database.



Realtime Database

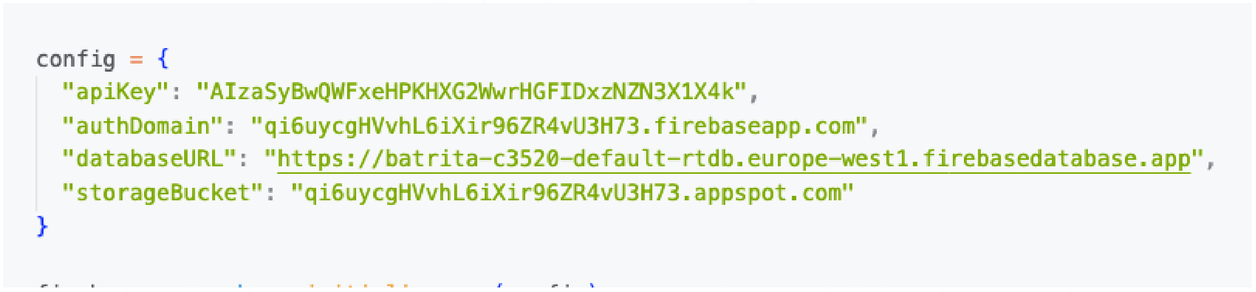
https://bats-explorer-de16a-default-rtdb.firebaseio.com

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database Learn more Dismiss

Read-only and non-realtime mode activated in the data viewer to improve browser performance
Select a key with fewer records to edit or view in realtime

```
https://bats-explorer-de16a-default-rtdb.firebaseio.com/
  Bat Classification
  GPS Data
  Humidity
  Latitude
```

Now we need config parameters from the fire, you can get databaseURL from realtime database, apiKey from Project settings and authDomain and storageDomain from user uid inside authentication.

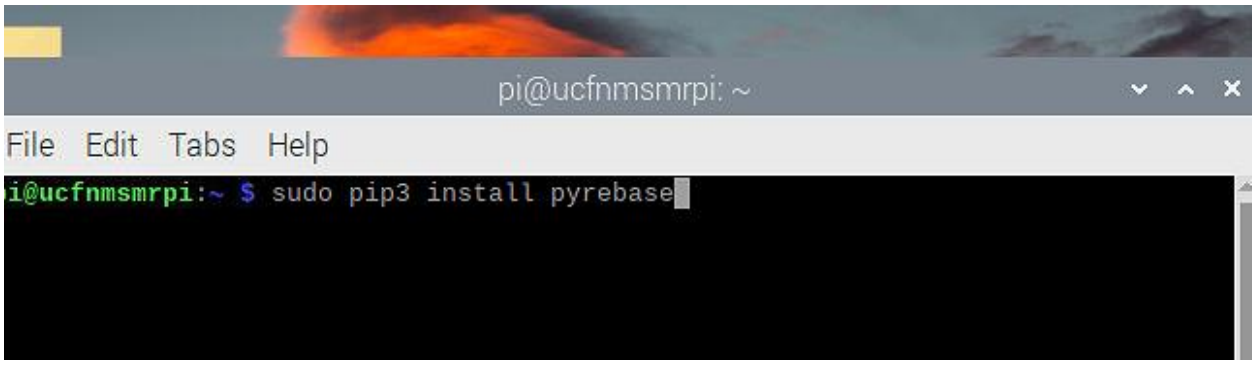


```
config = {
  "apiKey": "AIzaSyBwQWFxeHPKHXG2WwrHGFIDxzNZN3X1X4k",
  "authDomain": "qi6uycgHVvhL6iXir96ZR4vU3H73.firebaseapp.com",
  "databaseURL": "https://batrita-c3520-default-rtdb.europe-west1.firebasedatabase.app",
  "storageBucket": "qi6uycgHVvhL6iXir96ZR4vU3H73.appspot.com"
}
```

Now let’s use this info and send data from raspberry pi to the firebase.

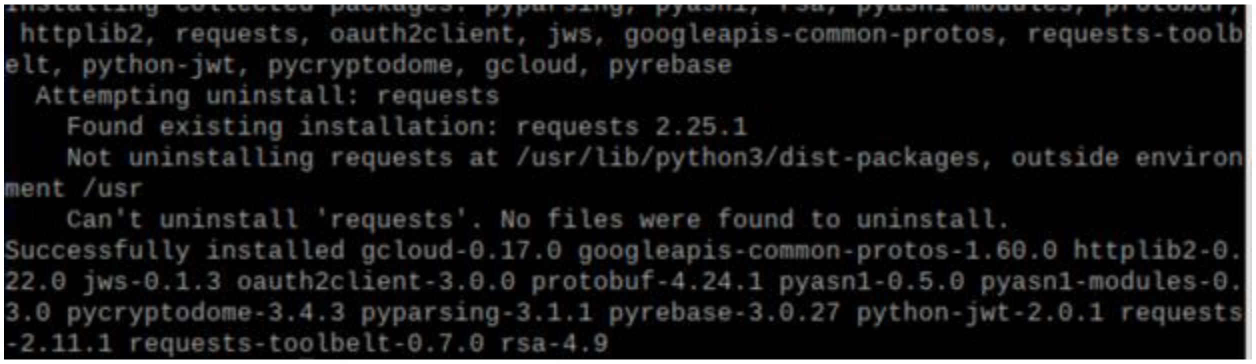
Let’s install the library we need to send data to firebase.

sudo pip3 install pyrebase



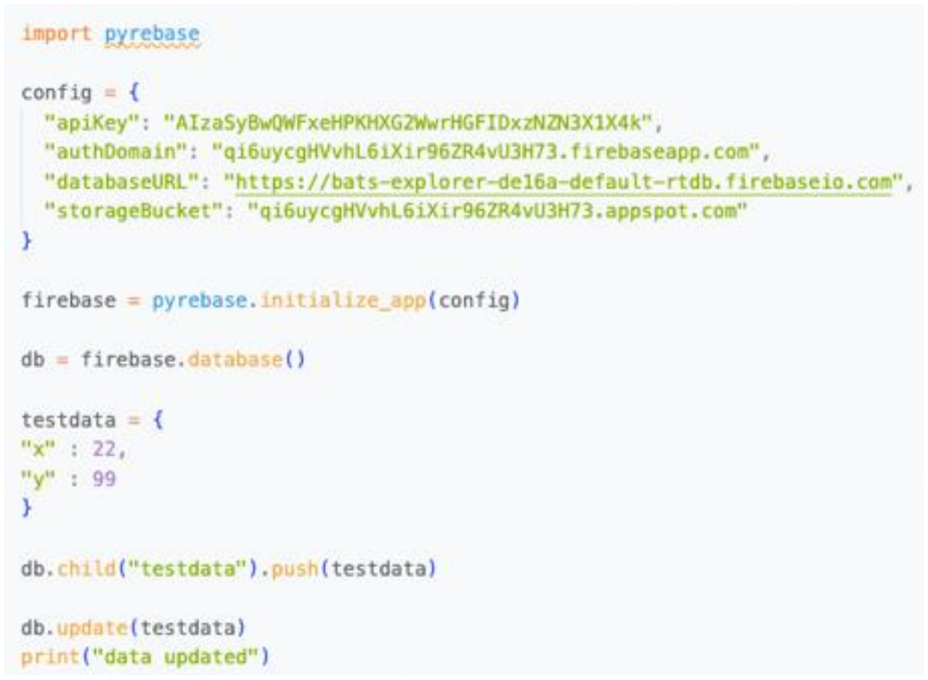
```
i@ucfnmsmrpi:~ $ sudo pip3 install pyrebase
```

Once the library is installed you will see this screen.



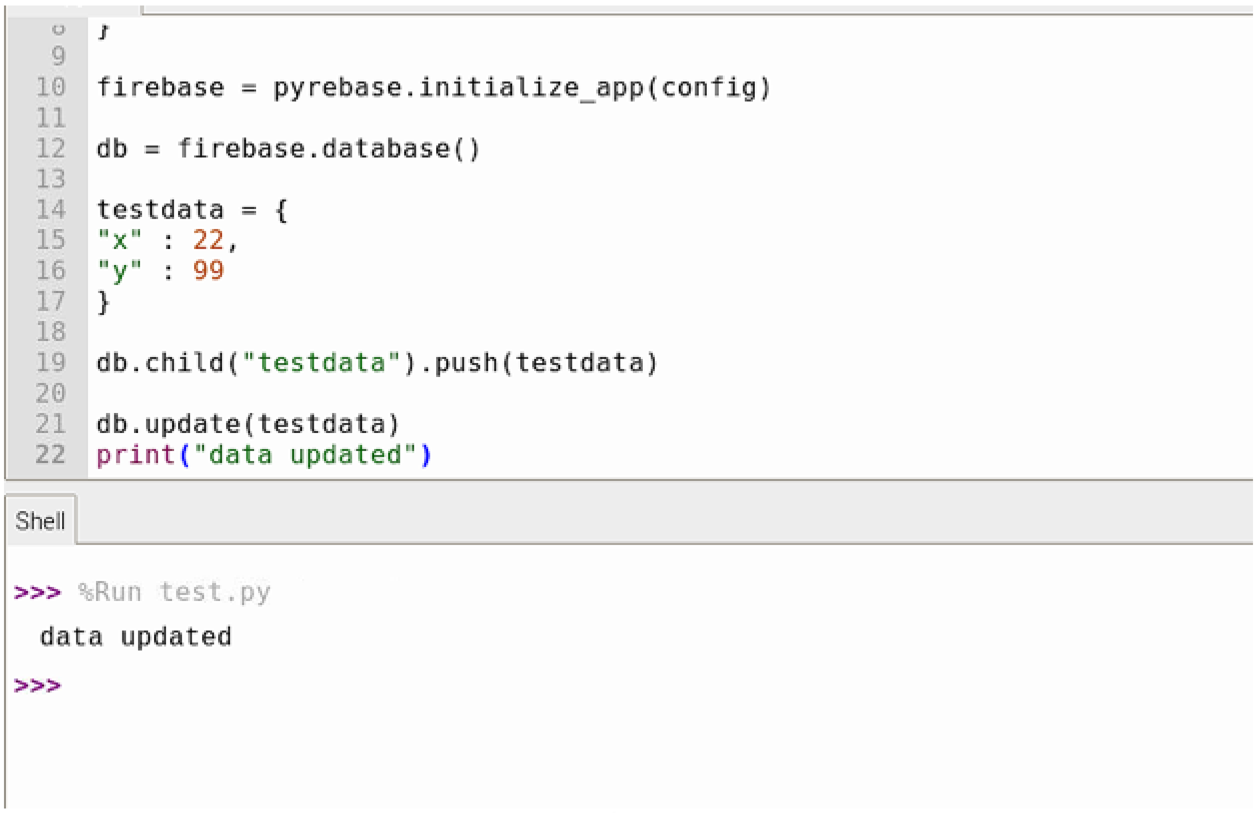
```
 httplib2, requests, oauth2client, jws, googleapis-common-protos, requests-toolb
elt, python-jwt, pycryptodome, gcloud, pyrebase
  Attempting uninstall: requests
    Found existing installation: requests 2.25.1
    Not uninstalling requests at /usr/lib/python3/dist-packages, outside environ
ment /usr
    Can't uninstall 'requests'. No files were found to uninstall.
Successfully installed gcloud-0.17.0 googleapis-common-protos-1.60.0 httplib2-0.
22.0 jws-0.1.3 oauth2client-3.0.0 protobuf-4.24.1 pyasn1-0.5.0 pyasn1-modules-0.
3.0 pycryptodome-3.4.3 pyparsing-3.1.1 pyrebase-3.0.27 python-jwt-2.0.1 requests
-2.11.1 requests-toolbelt-0.7.0 rsa-4.9
```

Now let's test the demo code to send data to our firebase.



```python
import pyrebase

config = {
  "apiKey": "AIzaSyBwQWFxeHPKHXG2WwrHGFIDxzNZN3X1X4k",
  "authDomain": "qi6uycgHVvhL6iXir96ZR4vU3H73.firebaseapp.com",
  "databaseURL": "https://bats-explorer-de16a-default-rtdb.firebaseio.com",
  "storageBucket": "qi6uycgHVvhL6iXir96ZR4vU3H73.appspot.com"
}

firebase = pyrebase.initialize_app(config)

db = firebase.database()

testdata = {
"x" : 22,
"y" : 99
}

db.child("testdata").push(testdata)

db.update(testdata)
print("data updated")
```

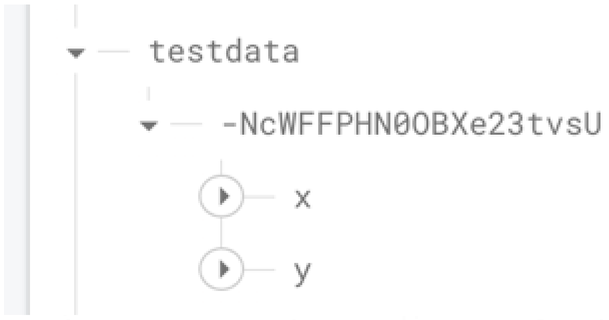
Let's run the test code.



```
}

firebase = pyrebase.initialize_app(config)

db = firebase.database()

testdata = {
"x" : 22,
"y" : 99
}

db.child("testdata").push(testdata)

db.update(testdata)
print("data updated")
```

```
>>> %Run test.py
  data updated
>>>
```

The data was updated on the firebase.

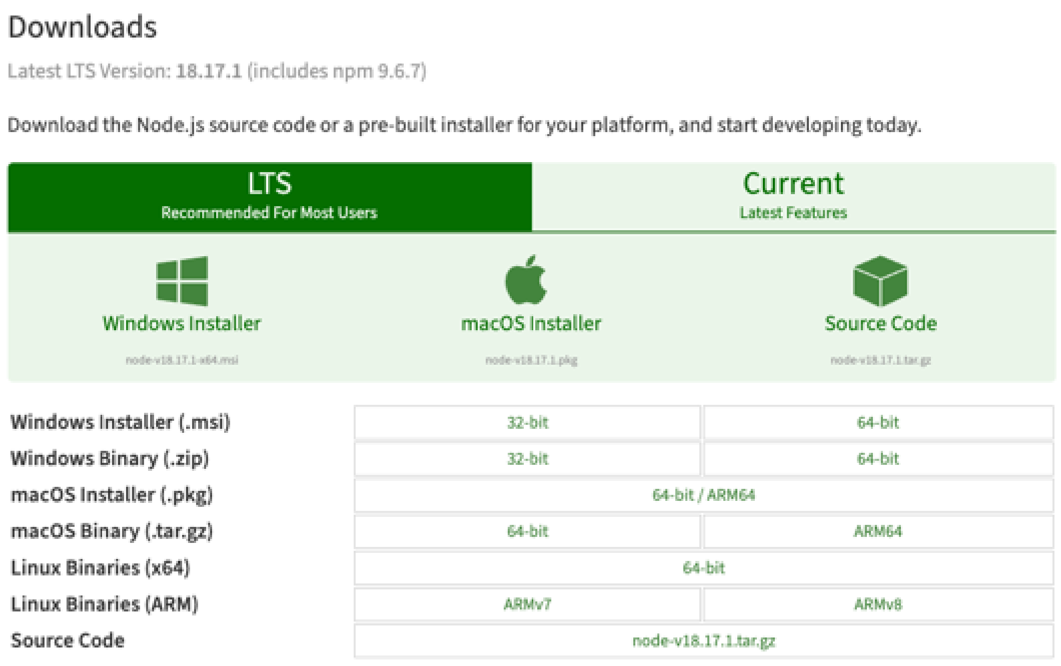


```
testdata
  -NcWFFPHN0OBXe23tvsU
    x
    y
```

Now let’s setup PM2 to run the code once raspberry pi boots up.

We need to install nodejs and npm in our system to install pm2.

Go to (https://nodejs.org/en/download)



Downloads

Latest LTS Version: 18.17.1 (includes npm 9.6.7)

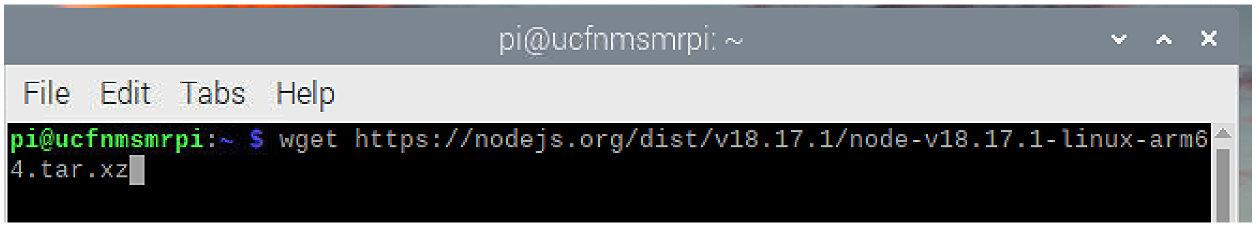
Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v18.17.1.tar.gz	

Now we copy link for Linux binary(arm) ARMv8 because our raspberry pi has 64-bit OS.

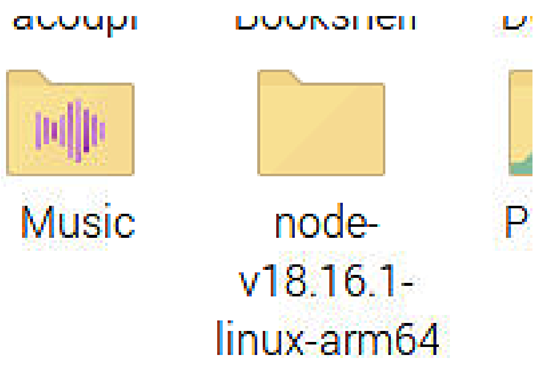
Then we run this command

wget https://nodejs.org/dist/v18.17.1/node-v18.17.1-linux-arm64.tar.xz



```
pi@ucfnmsmrpi:~ $ wget https://nodejs.org/dist/v18.17.1/node-v18.17.1-linux-arm6
4.tar.xz
```

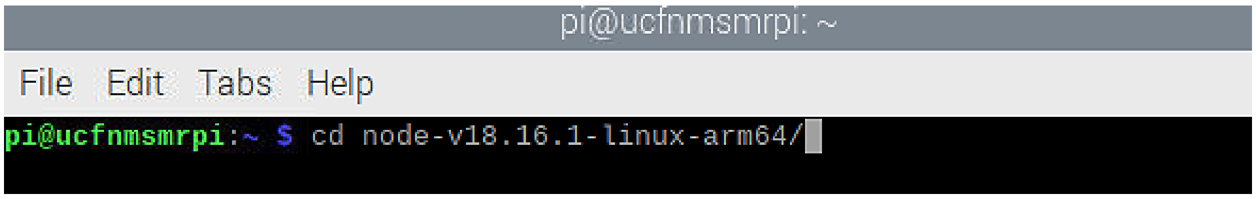
Once the package is downloaded extract that.



Copy the Extracted Files to Directory Path.

To do that go to terminal and type

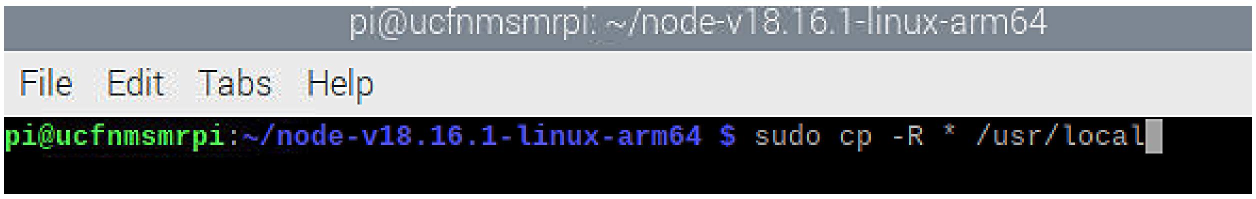
cd node-v18.16.1-linux-arm64



```
pi@ucfnmsmrpi:~ $ cd node-v18.16.1-linux-arm64/
```

now run

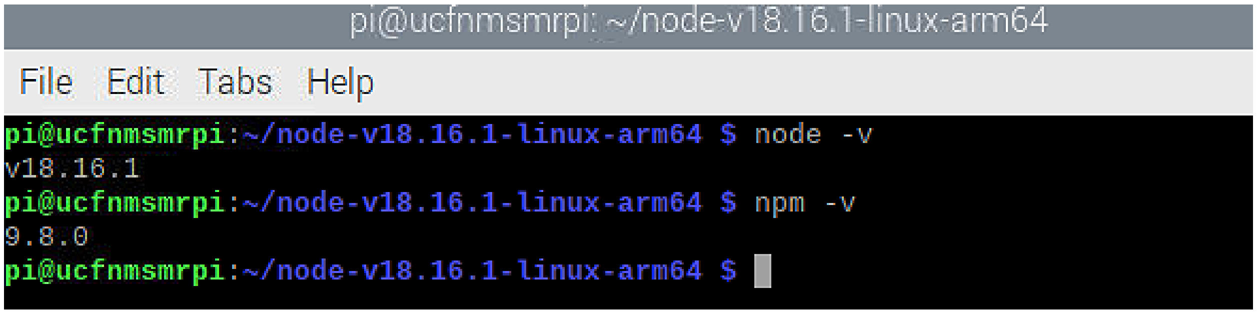
sudo cp -R * /usr/local



```
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $ sudo cp -R * /usr/local
```

Once the process completes it will go to next line on terminal.

Now let’s verify npm and node

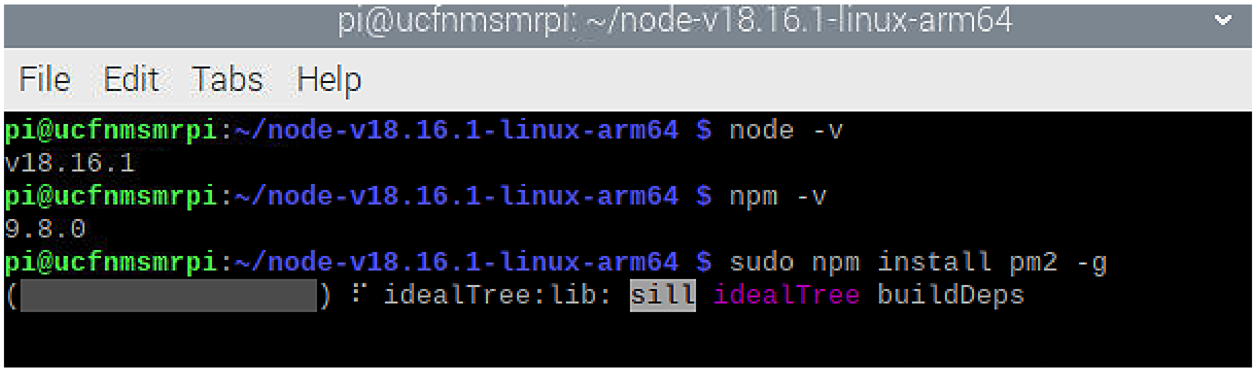


```
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $ node -v
v18.16.1
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $ npm -v
9.8.0
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $
```

Ok now let's install pm2

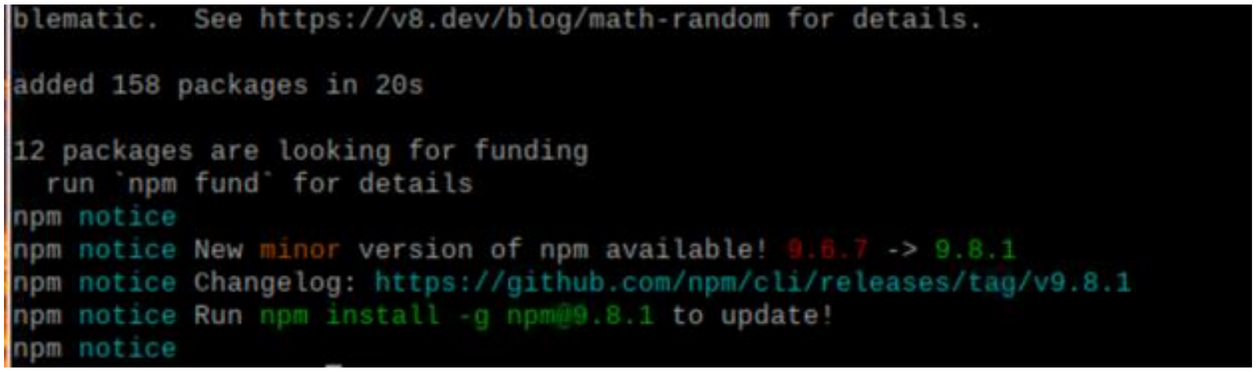
Enter this command in the terminal

sudo npm install pm2 -g



```
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $ node -v
v18.16.1
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $ npm -v
9.8.0
pi@ucfnmsmrpi:~/node-v18.16.1-linux-arm64 $ sudo npm install pm2 -g
(                  ) ⠋ idealTree:lib: sill idealTree buildDeps
```

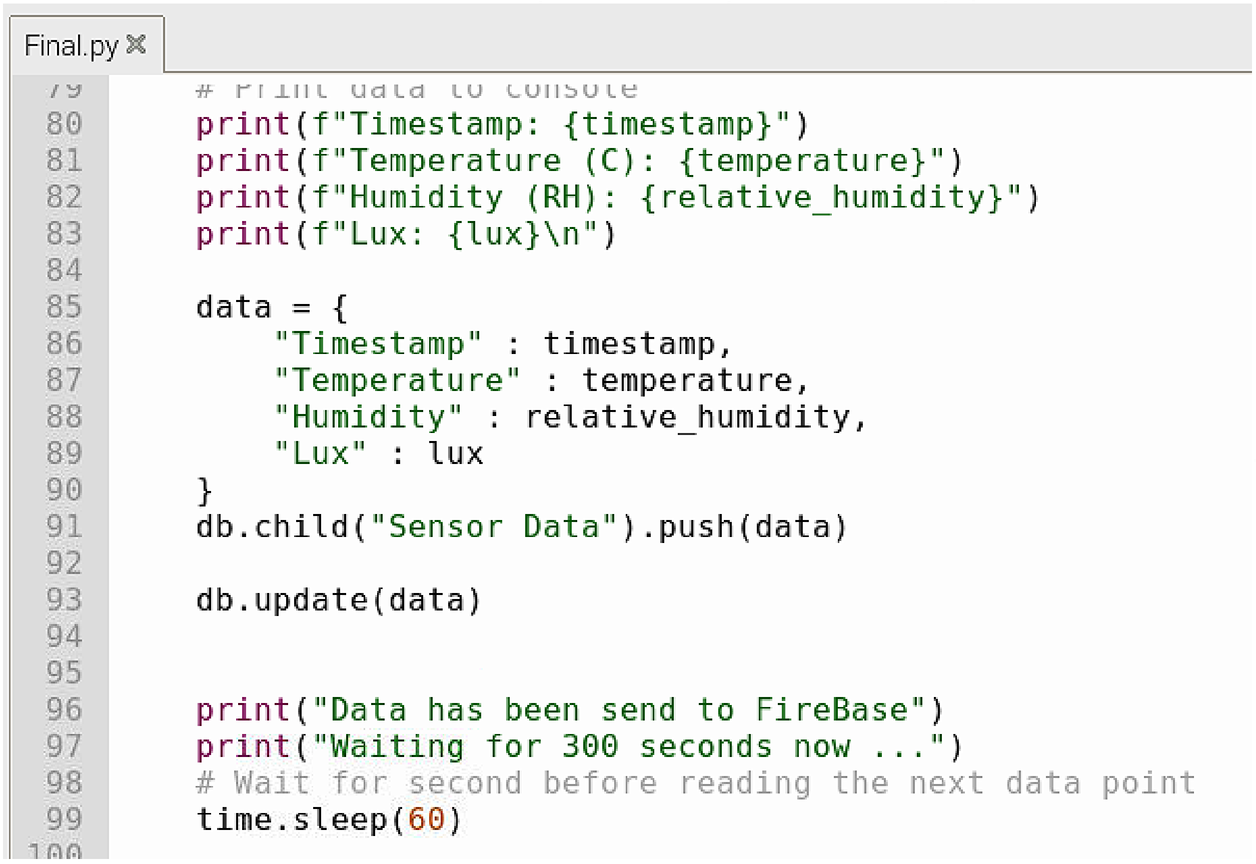
Once the process completes you will see this.



```
blematic.  See https://v8.dev/blog/math-random for details.

added 158 packages in 20s

12 packages are looking for funding
  run `npm fund` for details
npm notice
npm notice New minor version of npm available! 9.6.7 -> 9.8.1
npm notice Changelog: https://github.com/npm/cli/releases/tag/v9.8.1
npm notice Run npm install -g npm@9.8.1 to update!
npm notice
```

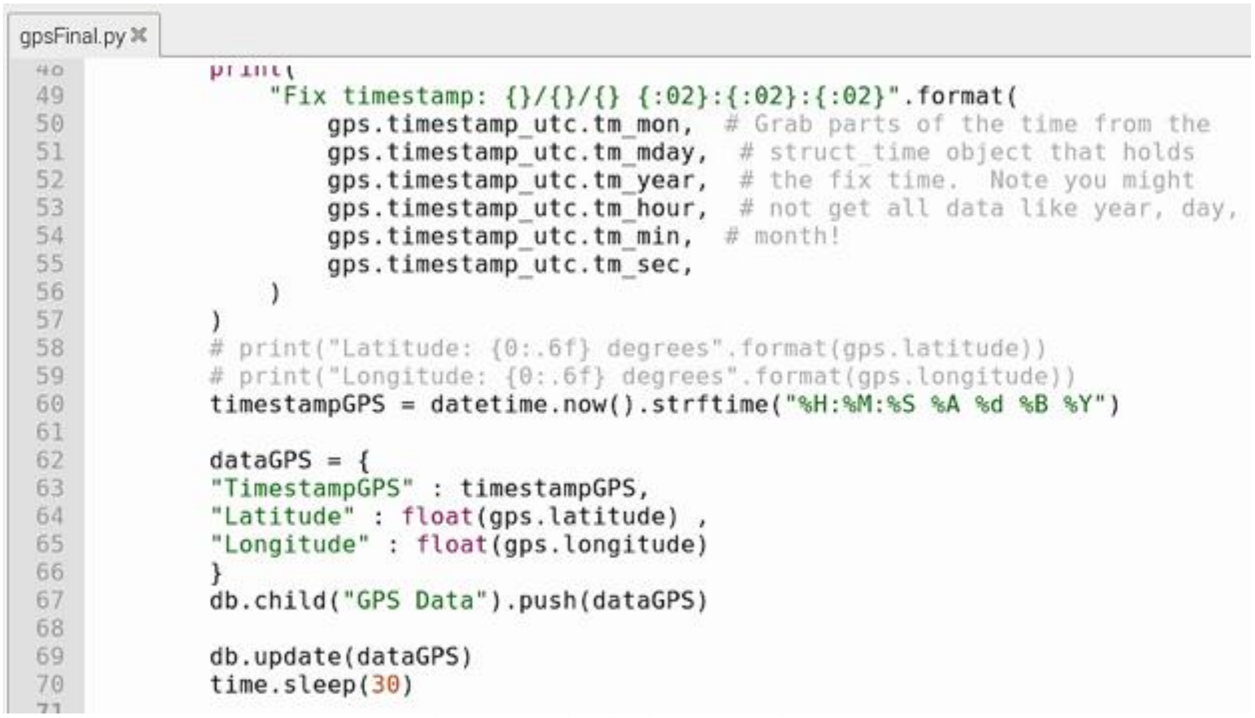
Now let's finalize our final codes and place them in Raspberry pi and run the automated scripts.

The temperature/humidity sensor's code, firebase's code was compiled together and saved in Final.py



```
    # Print data to console
    print(f"Timestamp: {timestamp}")
    print(f"Temperature (C): {temperature}")
    print(f"Humidity (RH): {relative_humidity}")
    print(f"Lux: {lux}\n")

    data = {
        "Timestamp" : timestamp,
        "Temperature" : temperature,
        "Humidity" : relative_humidity,
        "Lux" : lux
    }
    db.child("Sensor Data").push(data)

    db.update(data)


    print("Data has been send to FireBase")
    print("Waiting for 300 seconds now ...")
    # Wait for second before reading the next data point
    time.sleep(60)
```

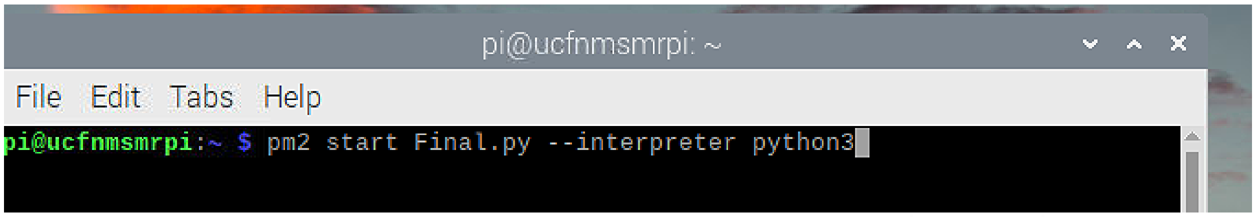
The GPS sensor's code, firebase's code was compiled together and saved in gpsFinal.py


gpsFinal.py

```
        print(
            "Fix timestamp: {}/{}/{} {:02}:{:02}:{:02}".format(
                gps.timestamp_utc.tm_mon,  # Grab parts of the time from the
                gps.timestamp_utc.tm_mday,  # struct_time object that holds
                gps.timestamp_utc.tm_year,  # the fix time.  Note you might
                gps.timestamp_utc.tm_hour,  # not get all data like year, day,
                gps.timestamp_utc.tm_min,  # month!
                gps.timestamp_utc.tm_sec,
            )
        )
        # print("Latitude: {0:.6f} degrees".format(gps.latitude))
        # print("Longitude: {0:.6f} degrees".format(gps.longitude))
        timestampGPS = datetime.now().strftime("%H:%M:%S %A %d %B %Y")

        dataGPS = {
        "TimestampGPS" : timestampGPS,
        "Latitude" : float(gps.latitude) ,
        "Longitude" : float(gps.longitude)
        }
        db.child("GPS Data").push(dataGPS)

        db.update(dataGPS)
        time.sleep(30)
```

To automate this scripts, we went to terminal and ran this command.

pm2 start Final.py –interpreter python3


pi@ucfnmsmrpi: ~

File Edit Tabs Help

```
pi@ucfnmsmrpi:~ $ pm2 start Final.py --interpreter python3
```

And the same for gpsFinal script

pm2 start gpsFinal.py –interpreter python3

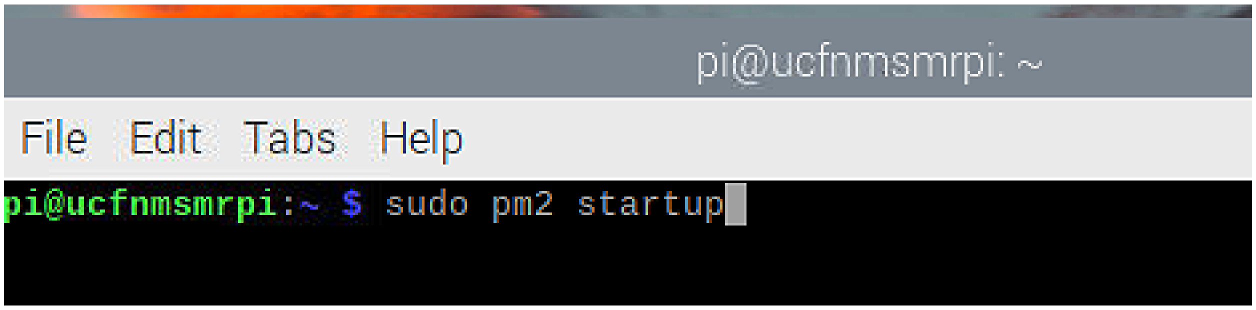
Then we ran

pm2 save

Now in order to run these scripts once Raspberry pi turns on. let's run the scripts in startup.

To do that run this command in terminal.

sudo pm2 startup

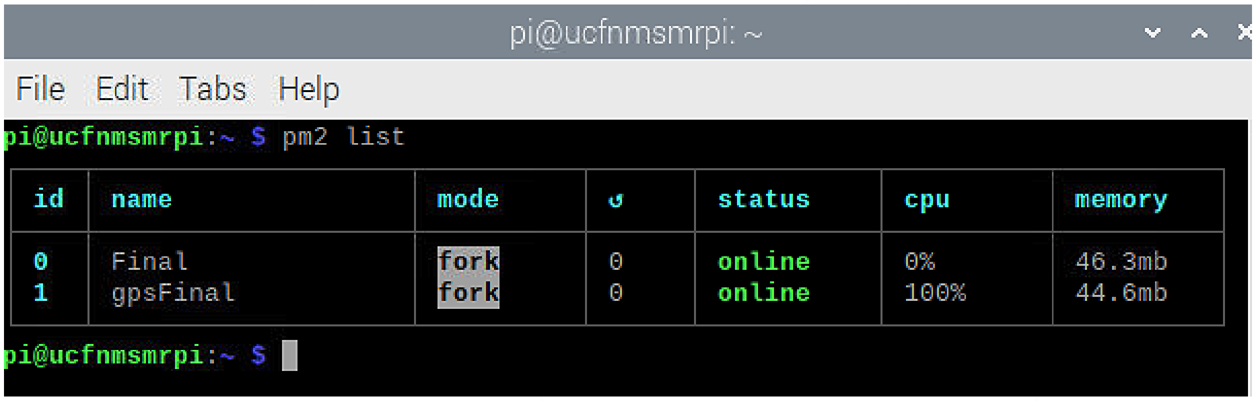


```
pi@ucfnmsmrpi:~ $ sudo pm2 startup
```

after this the code was able to run at boot. So, we rebooted.

To verify automatic starting of the code we ran

pm2 list



```
pi@ucfnmsmrpi:~ $ pm2 list
┌────┬──────────────────────┬──────────┬──────┬───────────┬──────────┬──────────┐
│ id │ name                 │ mode     │ ↺    │ status    │ cpu      │ memory   │
├────┼──────────────────────┼──────────┼──────┼───────────┼──────────┼──────────┤
│ 0  │ Final                │ fork     │ 0    │ online    │ 0%       │ 46.3mb   │
│ 1  │ gpsFinal             │ fork     │ 0    │ online    │ 100%     │ 44.6mb   │
└────┴──────────────────────┴──────────┴──────┴───────────┴──────────┴──────────┘
pi@ucfnmsmrpi:~ $
```

We could see that the scripts were running and the data was also being sent to the firebase.