Setting up a Raspberry Pi

so you can connect to the network headlessly

Scott Murphy scott.murphy@arrow-eye.com

Introduction

Why am I talking about this?

Two or three meetings ago, there was a discussion prior to the start regarding configuring Raspberry Pi units that would DHCP an address from one of multiple access points on different LANs.

Question: How do you know what address your Raspberry Pi device obtained from the local WiFi if you are bringing it online for the first time?

More specifically, how to determine the address to connect to if you couldn’t use the zeroconf/mDNS name to get to it, as you may be on a different network/broadcast domain. There were several proposals on how to do this.

I had a proposal

It was untested, so I did some testing.

Assuming I am remembering the conversation correctly, the unit was a Pi400, and I happen to have one of them, so testing my proposal was easy.

This method only requires two additional items:

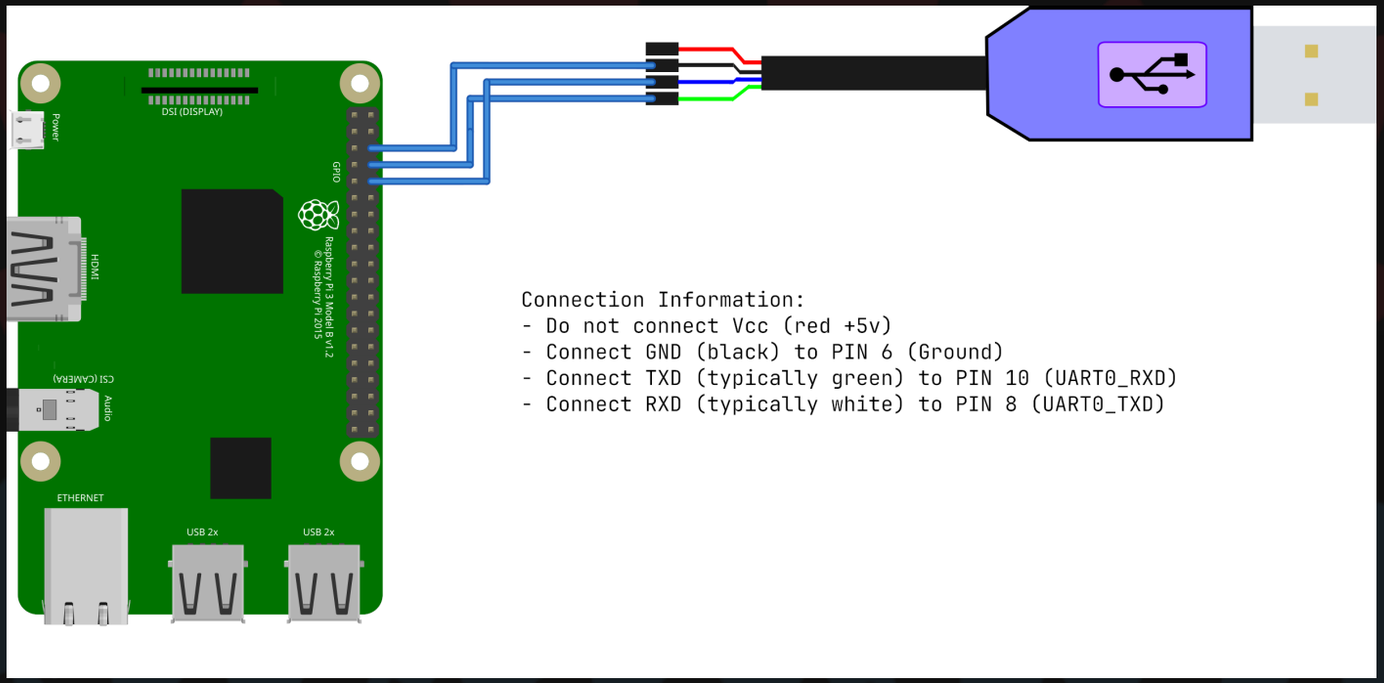
- a USB to TTL device to connect to the exposed serial port pins on the 40 pin GPIO header.
- The necessary wiring to connect it.

Other items are assumed since you have successfully created a microSD card to boot from.

This is somewhat over engineered, but I wanted to create a full HOWTO that included verification steps and extras.

Overview of the setup

Depending on what device you have, the wiring is going to be different, but in all cases, you are doing this:



Connecting the red wire (Vcc) will attempt to power the Pi from this 5v port and we are not doing that here.

Requirements

Assumptions

Most of this should be obvious, but just in case:

- You already know how to do a lot of this
  - You know how to install the necessary software
  - You have some minimal command line experience
- You have the Raspberry Pi
- You already have the hardware to run your Raspberry Pi
- You have some sort of serial to TTL adapter

My setup

- System to do the work on (my Fedora 42 mini PC)
- Raspberry Pi 400 (the target) and related power supply
- Console port stub (pishop.ca)
- Micro-USB cable (pishop.ca)
- Target WiFi Access Port to test connectivity (I created one for the demo)
- Raspberry Pi Imager software (Raspberry Pi Foundation)
- A microSD writer for the imager software to use (whatever you have)
- A communications program. I'll be using picocom. (freely installable)

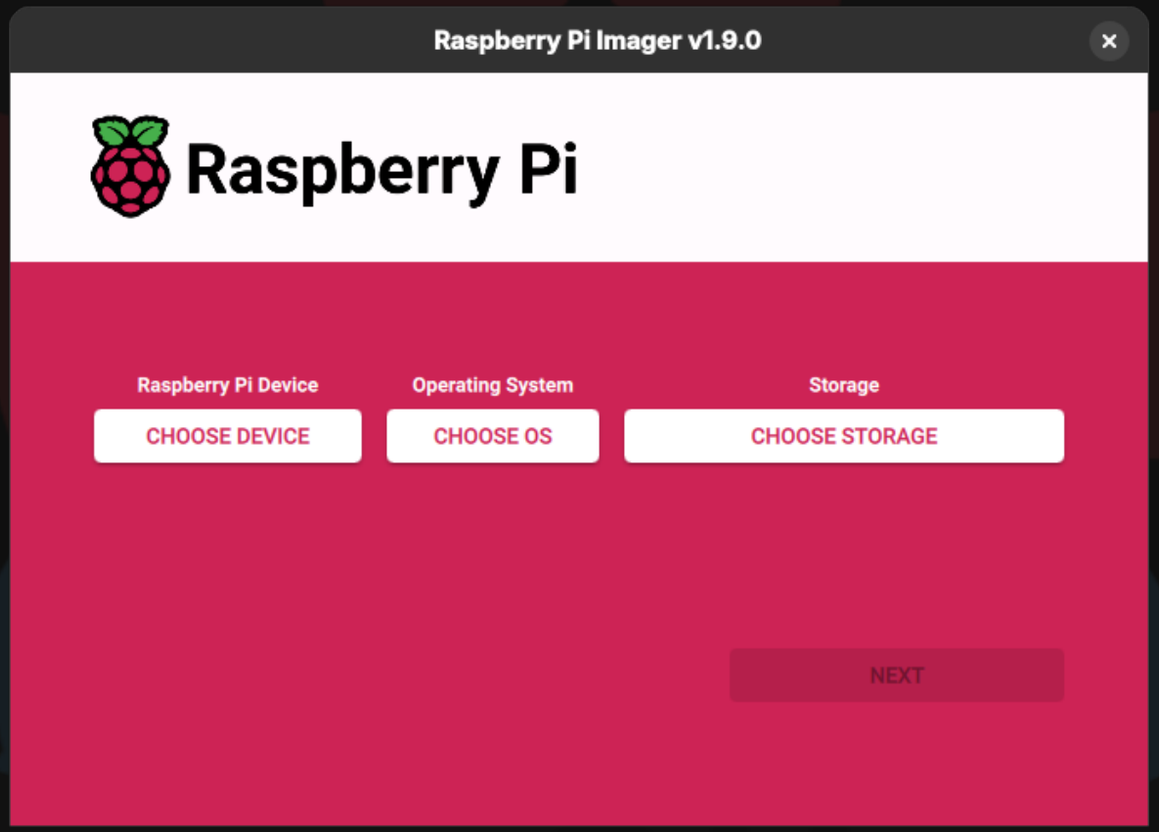
Using the Raspberry Pi Imager

Assumptions

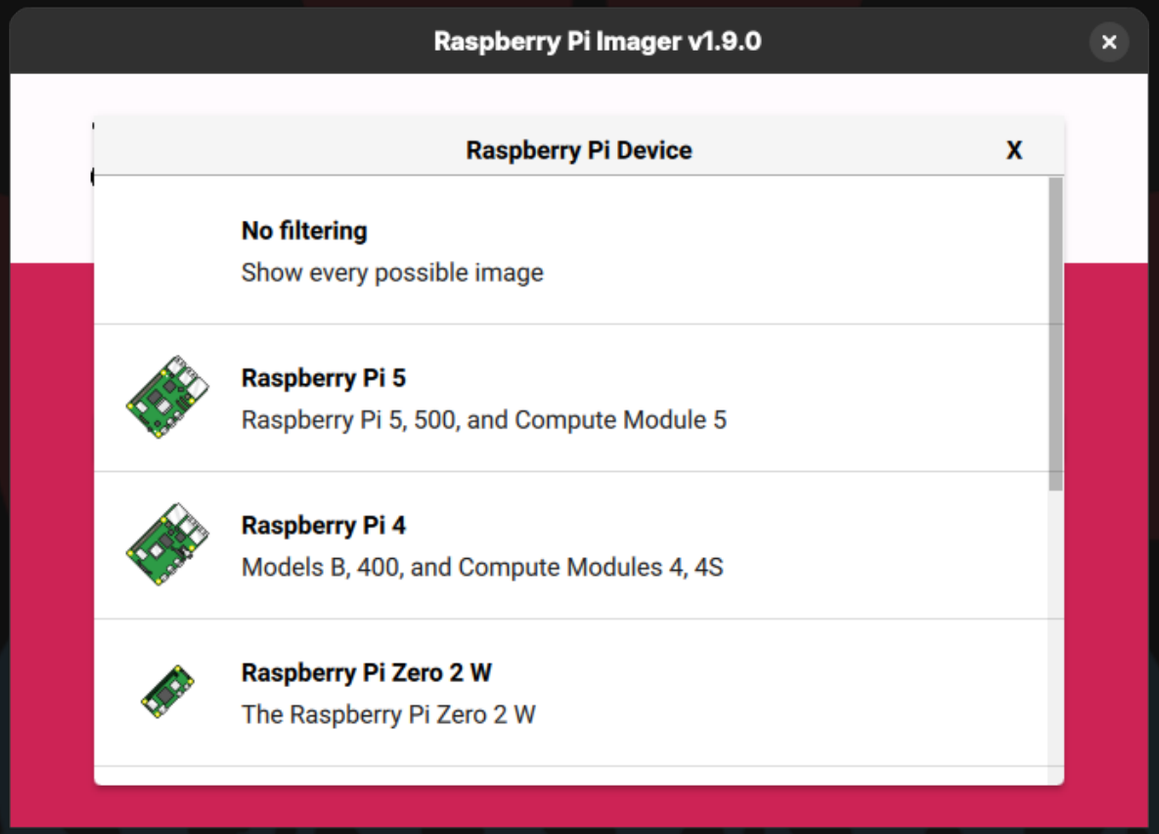
- You have installled the required software (see resources)
- You have an available microSD card at the appropriate size and speed

This is basically a walk-through with screenshots.

Launch the imager software



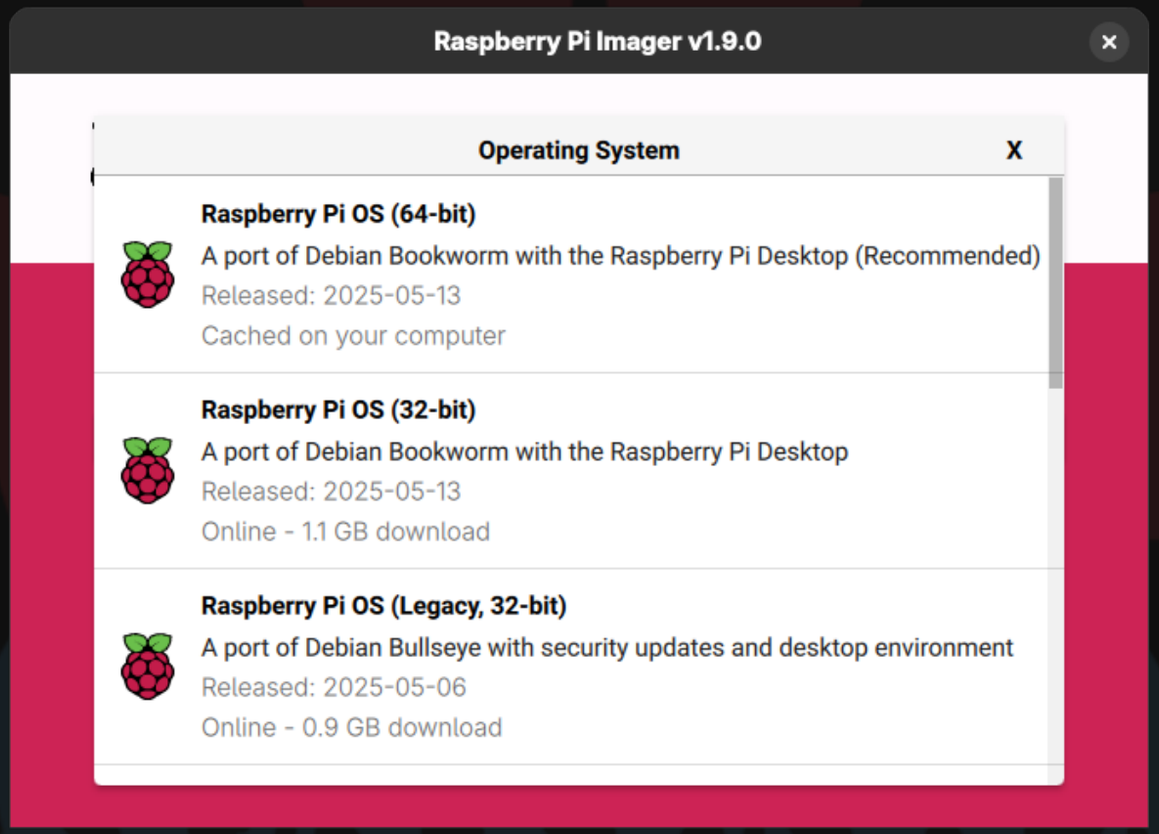
Choose the Pi device



I’m selecting the Pi 4 - it includes the 400.

Choose the OS version

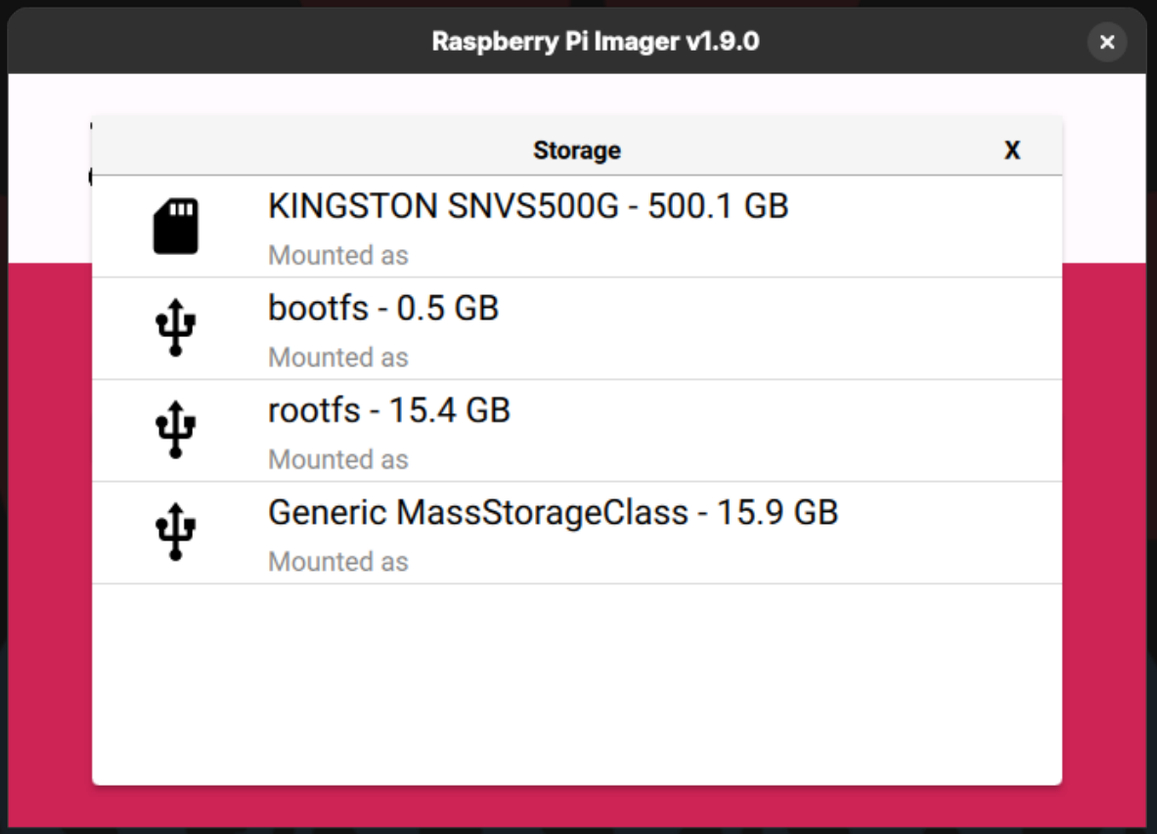
Pick from the presented selection.



I’m picking the first selection here, 64-bit Pi OS.

Time to select the storage device.

Selection



I’m selecting the bottom item.

Select "NEXT"


Raspberry Pi Imager v1.9.0
Raspberry Pi
Raspberry Pi Device
RASPBERRY PI 4
Operating System
RASPBERRY PI OS (64-BIT)
Storage
GENERIC MASSSTORAGECLASS
NEXT

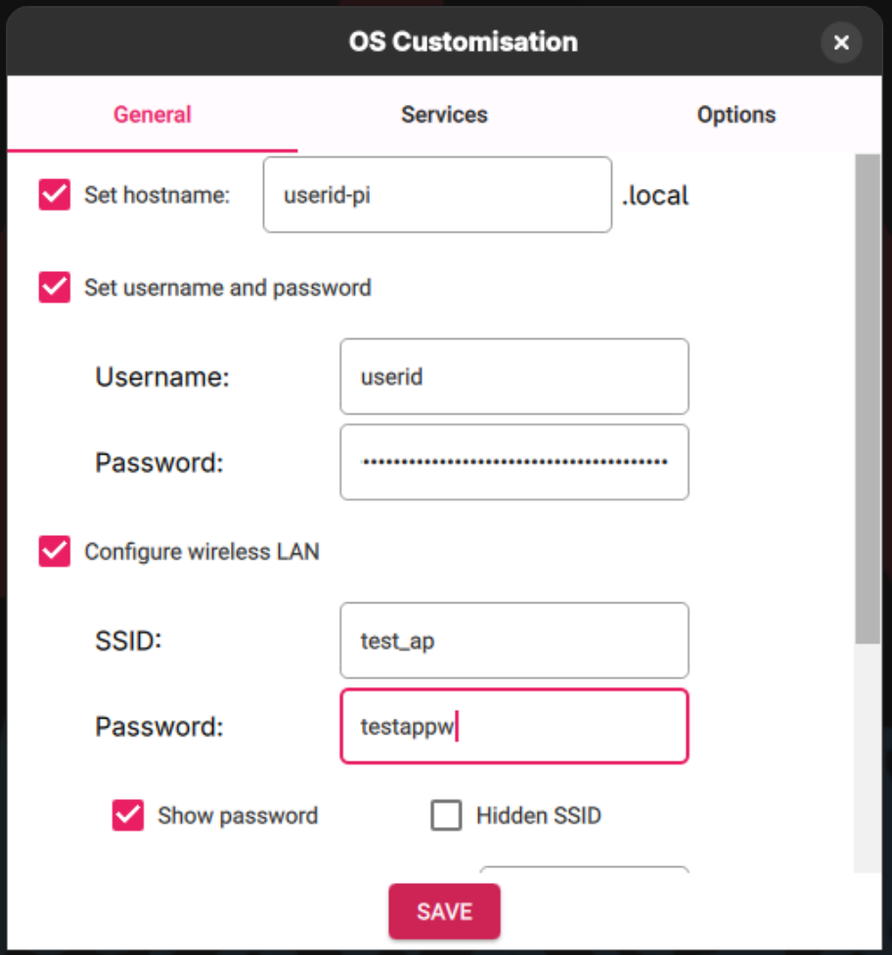
Customization

Assuming you will be doing different entries for each image, you probably want to make some changes here. Select "EDIT SETTINGS"

OS Customization

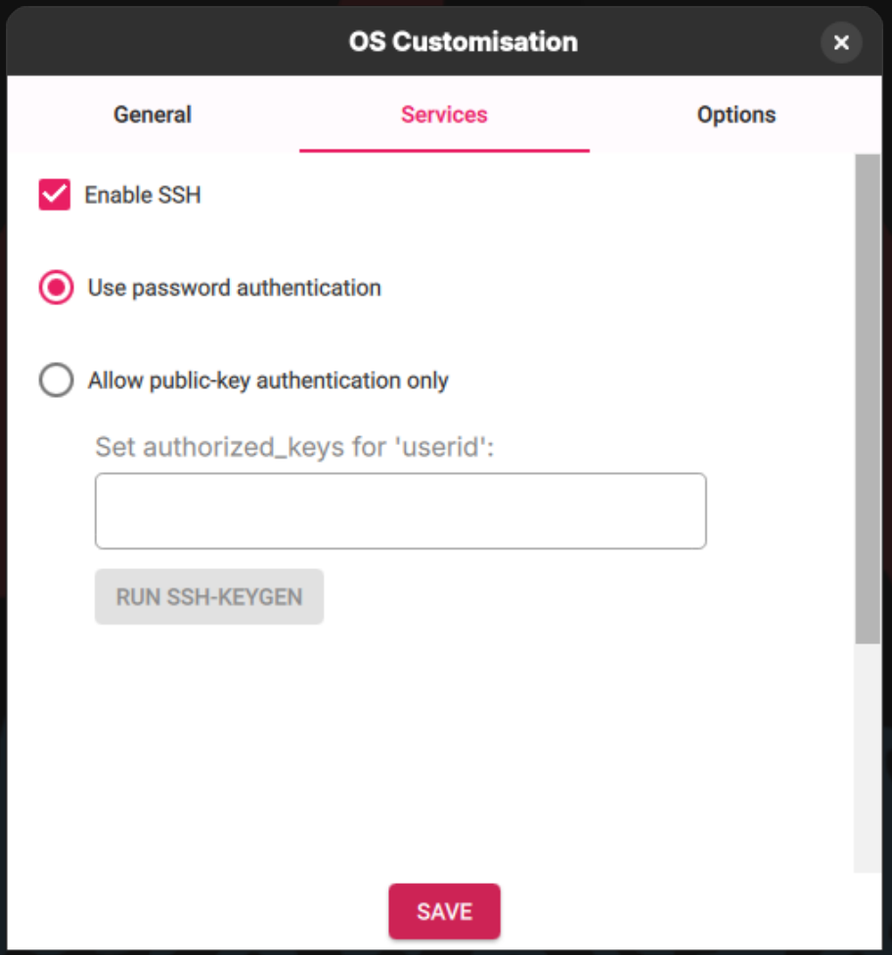
You will get three tables to edit

General



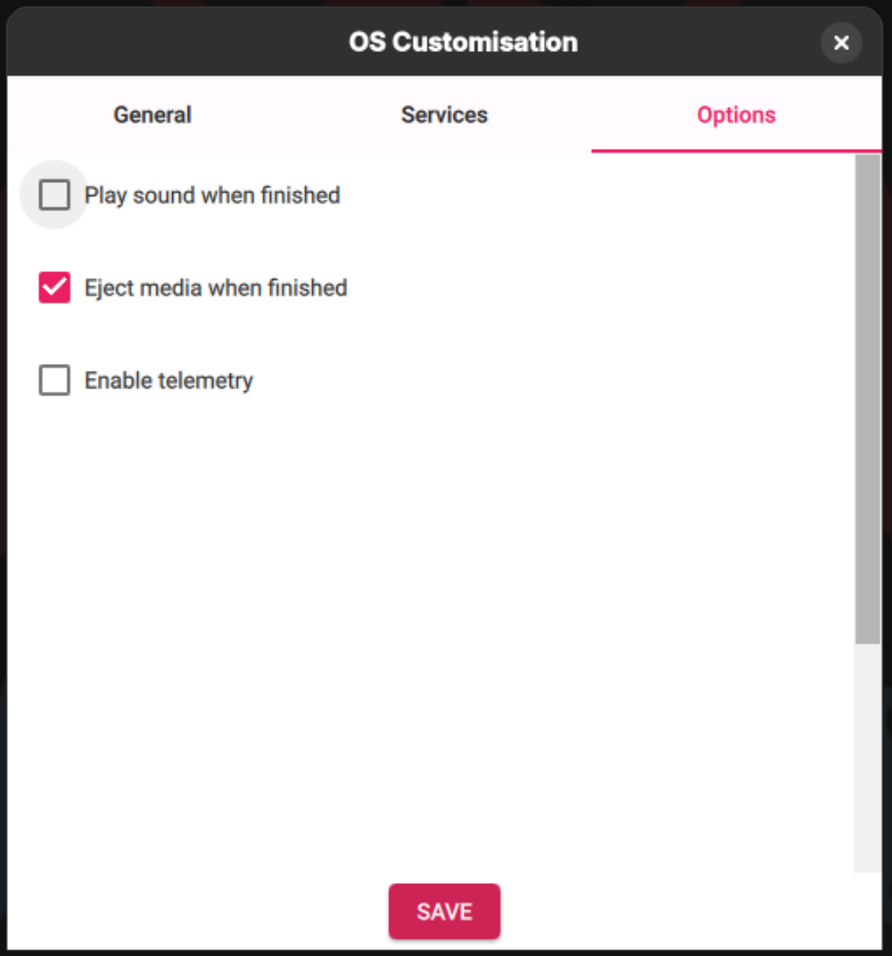
The first one will let you select the host name, the userid, the initial user password, and the WiFi settings.

Services



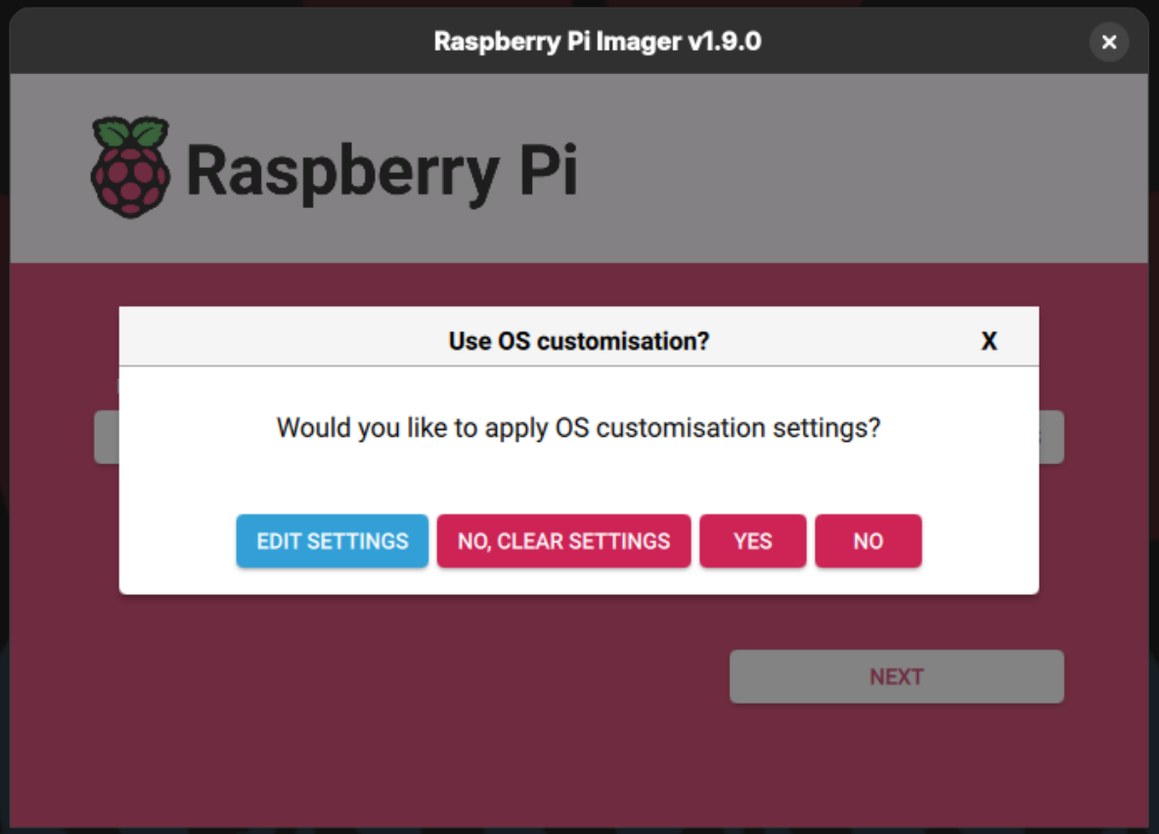
Enable ssh so you can connect and test after the system has come up.
Select password authentication for simplicity at this point.

Options



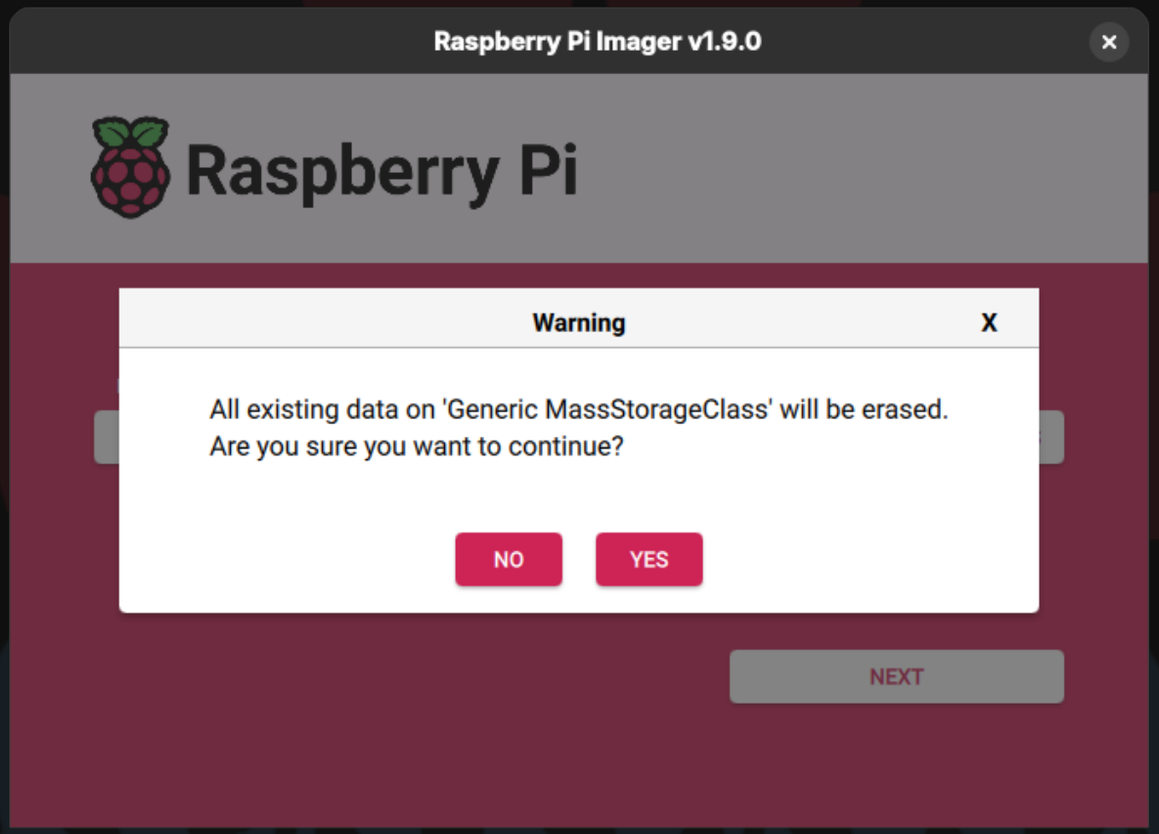
Make sure "Eject media when finished" is selected.

Apply your selections



Select "YES" to use these settings.

Time to write the image

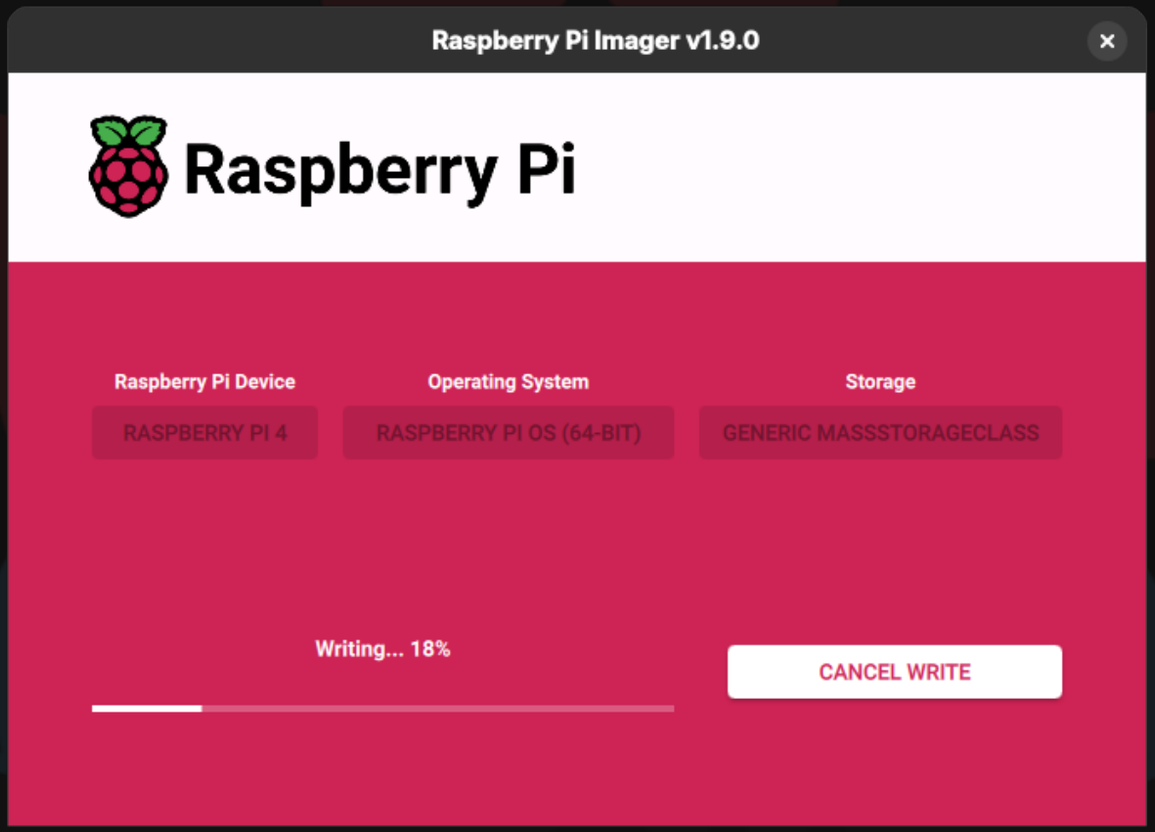


Confirm that you wish to overwrite the selected media. Last chance!

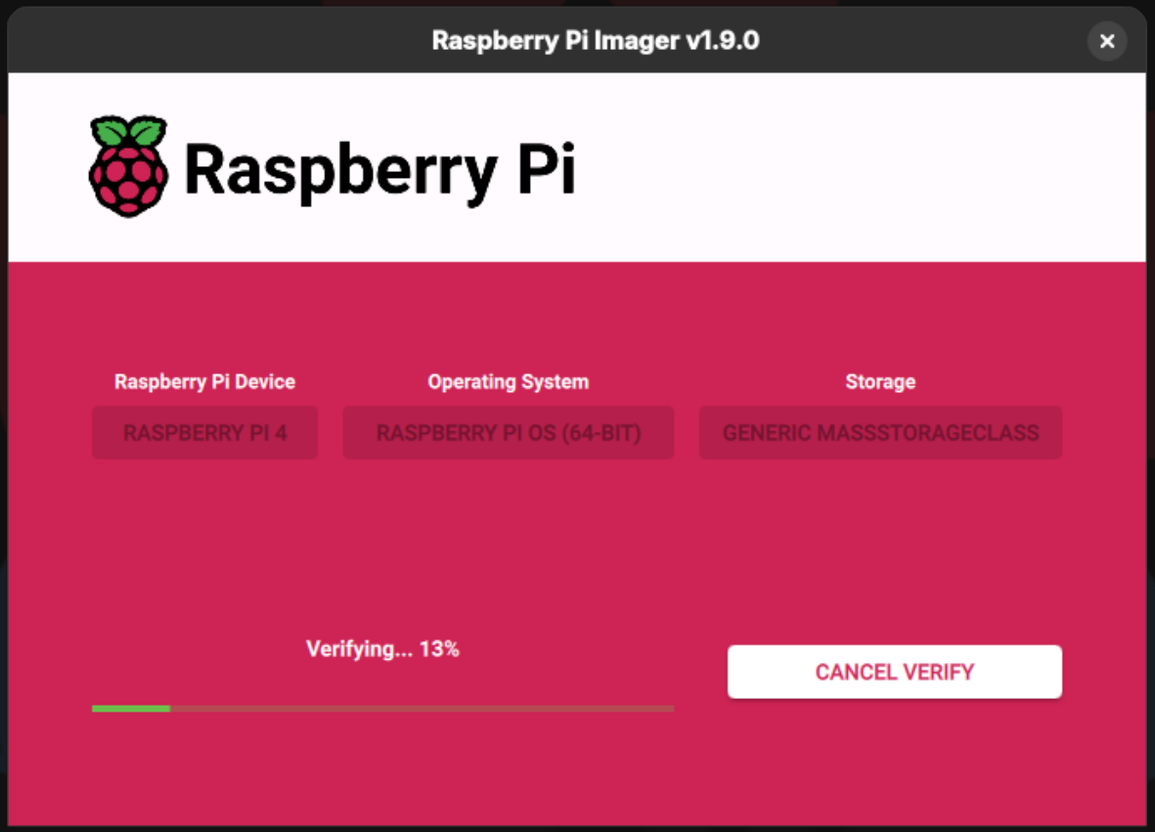
Preparation

Preparing to write. You will probably get a popup for a password in order to write to the device

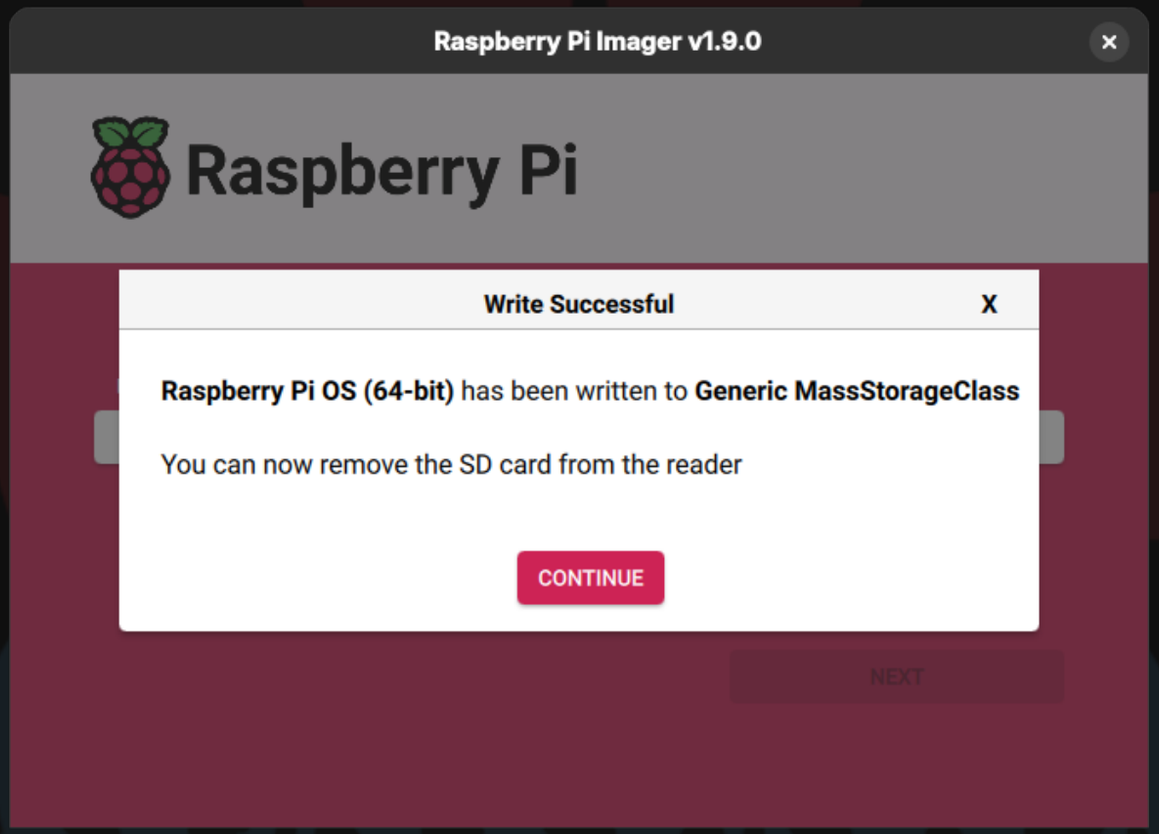
Writing


Raspberry Pi Imager v1.9.0
Raspberry Pi
Raspberry Pi Device
RASPBERRY PI 4
Operating System
RASPBERRY PI OS (64-BIT)
Storage
GENERIC MASSSTORAGECLASS
Writing... 18%
CANCEL WRITE

Verifying


Raspberry Pi Imager v1.9.0
Raspberry Pi
Raspberry Pi Device
RASPBERRY PI 4
Operating System
RASPBERRY PI OS (64-BIT)
Storage
GENERIC MASSSTORAGECLASS
Verifying... 13%
CANCEL VERIFY

Done



The image has now been written. Select Continue.

Card prep is complete

- You can close the imager software now.
- You should eject the card for the next step.
- Don’t lose it, you will be putting it back next

CLI Fun

Open a terminal

You are going to finish the SD card setup from the CLI.

This will be a walk-through to get the unit ready and get the initial network address to connect to.

Block Devices

type `lsblk` to see the block storage devices

```
user@demo: lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1     0B  0 disk
sdb           8:16   1     0B  0 disk
sdc           8:32   1     0B  0 disk
sdd           8:48   1     0B  0 disk
zram0       251:0    0     8G  0 disk [SWAP]
nvme0n1     259:0    0 465.8G  0 disk
├─nvme0n1p1 259:1    0   600M  0 part /boot/efi
├─nvme0n1p2 259:2    0     1G  0 part /boot
└─nvme0n1p3 259:3    0 464.2G  0 part /home
                                     /
```

Insert the storage card

Insert the microSD card back into your computer (not the Pi) and wait a couple of seconds, Enter the `lsblk` command again.

```
user@demo: lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1     0B  0 disk
sdb           8:16   1     0B  0 disk
sdc           8:32   1     0B  0 disk
sdd           8:48   1  14.8G  0 disk
├─sdd1        8:49   1   512M  0 part /run/media/user/bootfs
└─sdd2        8:50   1   5.2G  0 part
zram0       251:0    0     8G  0 disk [SWAP]
nvme0n1     259:0    0 465.8G  0 disk
├─nvme0n1p1 259:1    0   600M  0 part /boot/efi
├─nvme0n1p2 259:2    0     1G  0 part /boot
└─nvme0n1p3 259:3    0 464.2G  0 part /home
```

You can see the new storage has mounted. You care about the `bootfs` mount.

Enable the serial port

Issue the following command: `echo "enable_uart=1" >> /run/media/user/bootfs/config.txt`

This will enable the serial port on the Pi when it boots. This is where the "magic" happens.

Eject the storage card

Eject the microSD card with the command `sudo eject /dev/sdd` check to see that it is gone with `lsblk`.

Example:

```
user@demo: sudo eject /dev/sdd
[sudo] password for user:
user@demo: lsblk
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    1     0B  0 disk
sdb           8:16   1     0B  0 disk
sdc           8:32   1     0B  0 disk
sdd           8:48   1     0B  0 disk
zram0       251:0    0     8G  0 disk [SWAP]
nvme0n1     259:0    0 465.8G  0 disk
├─nvme0n1p1 259:1    0   600M  0 part /boot/efi
├─nvme0n1p2 259:2    0     1G  0 part /boot
└─nvme0n1p3 259:3    0 464.2G  0 part /home
                                     /
```

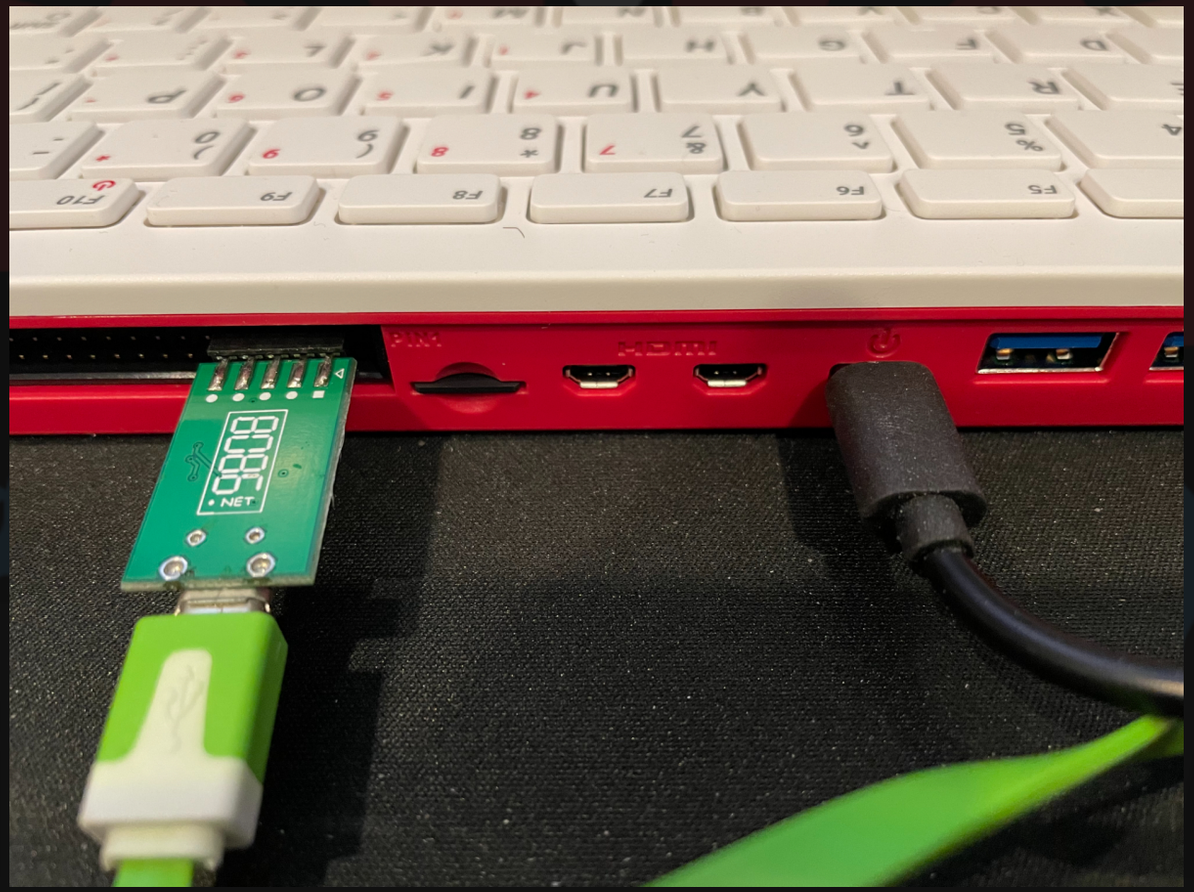
Hardware Prep

Steps

- Insert the microSD card into the Pi.
- Connect the console stub to the GPIO pins. It goes on Pin 1 with the little white triangle above pin one.
- Plug the micro-USB connector into the stub and the other end into your computer.
- Do not power it on yet.

Visual

It will probably not look exactly like this unless you have an inline power switch or the adapter is not plugged in. Remember that you DO NOT have the power applied yet. You want to see it booting.



Back to the command line

Where are we now?

The USB cable you plugged in is now acting as a serial device. You can find it in listed in `/dev`. It will have a name similar to `ttyACM0` or `ttyUSB0` on most systems. It could also show us as `cu.serial` or some other variation. If it is not obvious, unplug it, check the `/dev/*` listings, plug it back in, and check again. The device that appeared is your device.

Example listing

```
user@demo: ls /dev
autofs           i2c-0        nvme0n1    tty1   tty37  tty7     ttyS31
block            i2c-1        nvme0n1p1  tty10  tty38  tty8     ttyS4
bsg              i2c-2        nvme0n1p2  tty11  tty39  tty9     ttyS5
btrfs-control    i2c-3        nvme0n1p3  tty12  tty4   ttyACM0  ttyS6
bus              i2c-4        nvram      tty13  tty40  ttyACM1  ttyS7
char             i2c-5        port       tty14  tty41  ttyS0    ttyS8
console          i2c-6        ppp        tty15  tty42  ttyS1    ttyS9
hugepages        null         tty        tty35  tty62  ttyS3    vcsa1
hwrng            nvme0        tty0       tty36  tty63  ttyS30   vcsa2
```

My device in this instance is `/dev/ttyACM1`. Many devices were removed from the example for the sake of clarity.

Picocom

```
user@demo: picocom -b 115200 /dev/ttyACM1
picocom v2024-07

port is        : /dev/ttyACM1
flowcontrol    : none
baudrate is    : 115200
parity is      : none
databits are   : 8
stopbits are   : 1
...
minimal cmds is: no

Type [C-a] [C-h] to see available commands
Terminal ready
```

I deleted a few lines so it would fit on the slide.

Power on the Pi

Now you can plug in the Pi. This next part takes a moment, as you do not have a screen plugged in, so you have to wait for a few housekeeping tasks to run. After a short period, you will see something like this:

```
[   17.506728] reboot: Restarting system with command '1'
[   14.754996] reboot: Restarting system

Debian GNU/Linux 12 userid-pi ttyS0

My IP address is 127.0.1.1 fdbb:5bee:87ae:de4b:f71c:46c2:fe8:33b1

userid-pi login:
Debian GNU/Linux 12 userid-pi ttyS0

My IP address is 192.168.0.157 fdbb:5bee:87ae:de4b:f71c:46c2:fe8:33b1

userid-pi login:
```

What was all that?

Notice that the Pi rebooted, it came up with no networking info (127.0.0.1), then provided an IP address (this time 192.168.0.157). That is from the WiFi AP it connected to. Log in with the userid/password combination you set in the initial configuration with the Pi Imager software. You can see that you are connected to the Pi serial port ttyS0 (in the banner message).

Login

```
Debian GNU/Linux 12 userid-pi ttyS0

My IP address is 192.168.0.157 fdbb:5bee:87ae:de4b:f71c:46c2:fe8:33b1

userid-pi login: userid
Password:
Linux userid-pi 6.12.25+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.12.25-1+rpt1 (2025-04-30)

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: Mon May 12 20:18:06 EDT 2025 on tty1
userid@userid-pi:~$
```

Check the interfaces

```
userid@userid-pi:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group defaul
    link/ether dc:a6:32:d9:a0:30 brd ff:ff:ff:ff:ff:ff
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group de
    link/ether dc:a6:32:d9:a0:32 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.157/24 brd 10.9.15.255 scope global dynamic noprefixroute wlan0
       valid_lft 86377sec preferred_lft 86377sec
    inet6 fdbb:5bee:87ae:de4b:f71c:46c2:fe8:33b1/64 scope global dynamic noprefixroute
       valid_lft 1789sec preferred_lft 1789sec
    inet6 fe80::39c7:63c:6735:8dd9/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
userid@userid-pi:~$
```

Check your connectivity

Verify it has network connectivity, try pinging Google.

```
userid@userid-pi:~$ ping google.com
PING google.com (142.251.33.174) 56(84) bytes of data.
64 bytes from yyz10s17-in-f14.1e100.net (142.251.33.174): icmp_seq=1 ttl=113 time=19.5
64 bytes from yyz10s17-in-f14.1e100.net (142.251.33.174): icmp_seq=2 ttl=113 time=21.3
64 bytes from yyz10s17-in-f14.1e100.net (142.251.33.174): icmp_seq=3 ttl=113 time=13.6
64 bytes from yyz10s17-in-f14.1e100.net (142.251.33.174): icmp_seq=4 ttl=113 time=14.4
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 13.598/17.200/21.324/3.278 ms
userid@userid-pi:~$
```

Logout from the serial port

```
userid@userid-pi:~$ exit
logout

Debian GNU/Linux 12 userid-pi ttyS0

My IP address is 192.168.0.157 fdbb:5bee:87ae:de4b:f71c:46c2:fe8:33b1

userid-pi login:
```

Try the network now

That is all working, so now try connecting via the network.

Leave picocom with `[CTRL-a],[CTRL-x]` and `ssh` into the Pi via the IP address you saw on the banner or from the `ip` command output. You could try to connect by the name you gave it with a `.local` extension, but that will only work inside the same broadcast domain and you may be on a different network.

Example session

```
Terminating...
Thanks for using picocom
user@demo:
user@demo: ssh userid@192.168.0.157
userid@192.168.0.157's password:
Linux userid-pi 6.12.25+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.12.25-1+rpt1 (2025-04-30)

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: Mon May 12 20:18:27 2025
userid@userid-pi:~ $ uptime
 12:31:58 up 1 min,  3 users,  load average: 0.51, 0.27, 0.10
```

Next steps

- Update the system (`sudo apt update; sudo apt upgrade`).
- Do whatever you else need to do.

Getting a USB HDMI capture card would allow you to view and capture the Pi HDMI output on your desktop with appropriate software. Great for showing things and capturing to a video/streaming setup.

Additional Notes

Remote display of graphics

Even without a display, you can see the graphical output of most commands. If you are on windows, you need something like Cygwin-X installed. A simple way to accomplish this is to install MobaXterm, which gets you an encapsulated Debian like environment with X11 support running under windows. It is free for personal use.

Assuming you have that or a system that understands X-11 like environments, you can log in with X-11 forwarding through the ssh connection.

Tunneling X traffic

Connect with `ssh -Y <userid>@<ip-address>` and you will have the ability to run GUI apps from the Pi. There are probably a few that will not work properly, and you will see some errors since the full desktop environment is not available over the tunnel.

```
user@demo: ssh -Y userid@192.168.0.157
userid@10.9.8.156's password:
Linux userid-pi 6.12.25+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.12.25-1+rpt1 (2025-04-30)

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: Sun Sep 21 10:20:01 2025 from 10.9.8.63
userid@userid-pi:~ $
```

The 'Y' flag rather than 'X' flag helps with some trust issues. See the man page for a full explanation.

Some things to demo

From here, you can install a few things:

- cool-retro-term
- screenfetch (there are newer ones, but this is sufficient for our purposes)
- cpufetch

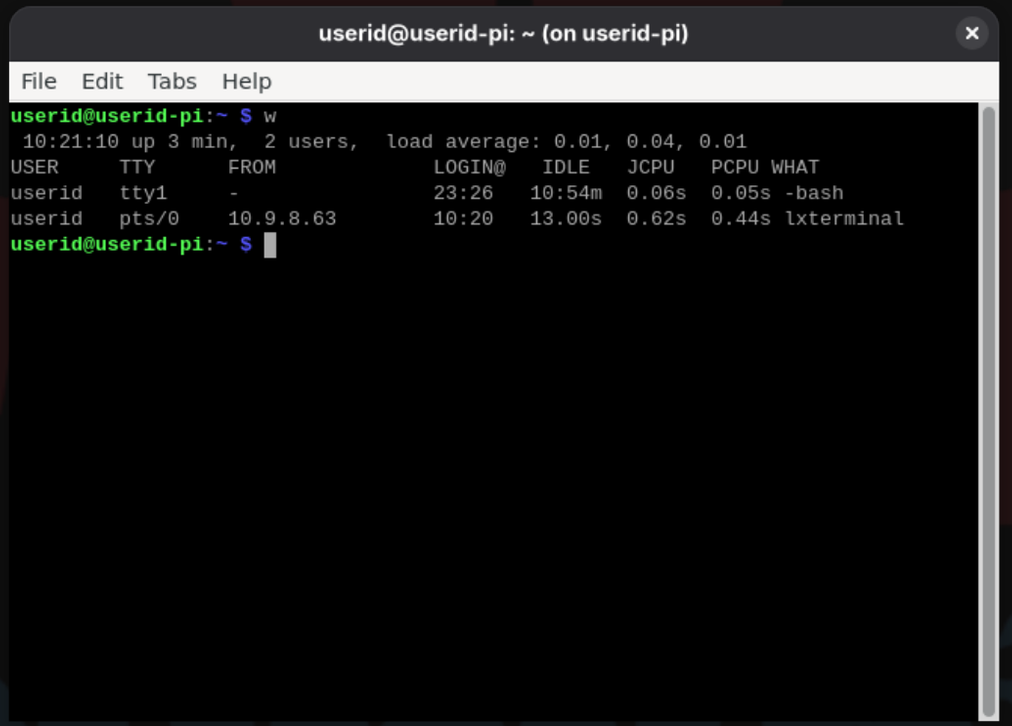
Install with: `sudo apt install cpufetch screenfetch cool-retro-term`

You are already logged in over the tunnel, so enter the following command (You will not get a prompt back unless you background it. We will not do that for this demo).

```
userid@userid-pi:~ $ lxterminal
Gtk-Message: 10:38:40.903: Failed to load module "canberra-gtk-module"
Gtk-Message: 10:38:40.903: Failed to load module "pk-gtk-module"
Gtk-Message: 10:38:40.911: Failed to load module "canberra-gtk-module"
Gtk-Message: 10:38:40.911: Failed to load module "pk-gtk-module"
```

lxterminal

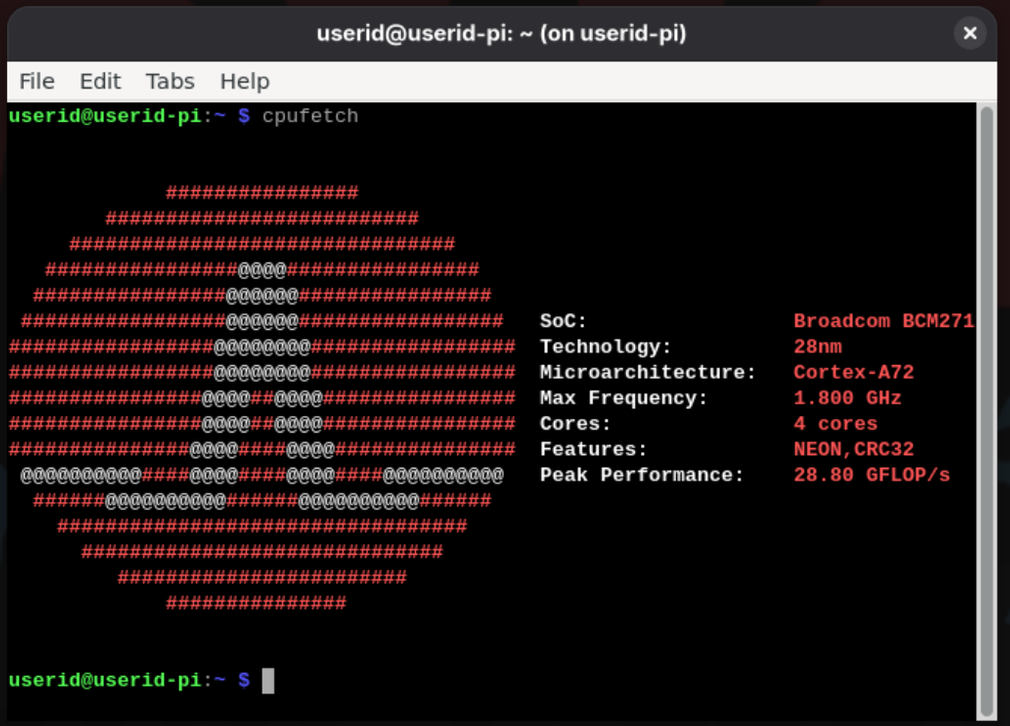
You should get a window that pops up on your device that looks like this:



I went ahead and typed the `w` command. That first login on tty1 is the console and this Pi is currently set to autologin on the console. That should be changed to requiring a login. You can fix that with the `raspi-config` program later.

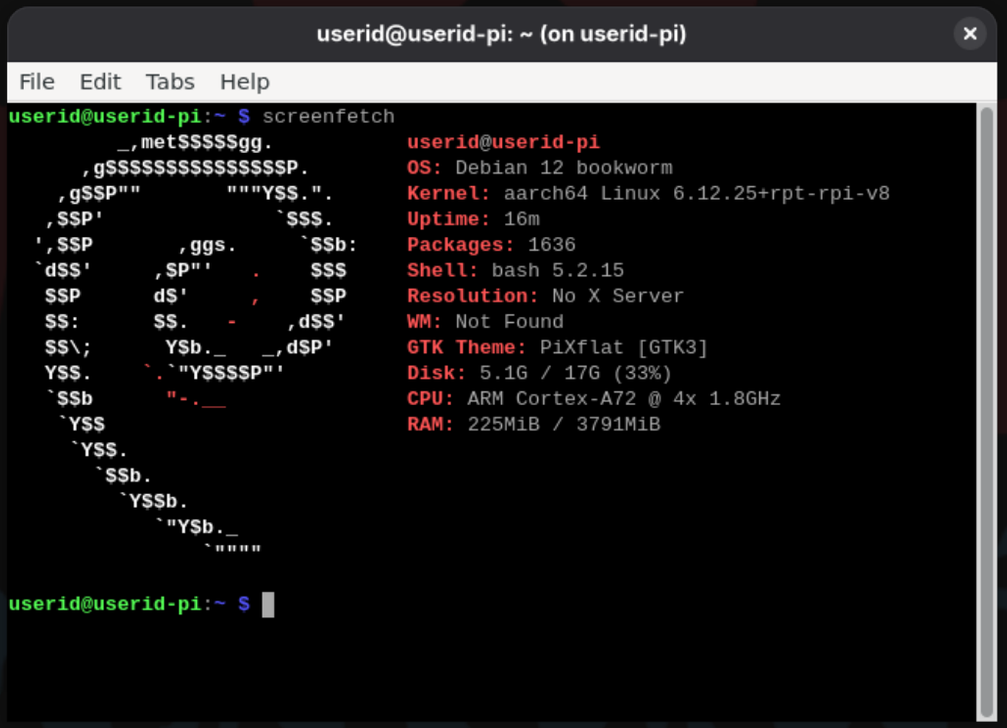
cpufetch

If you clear the screen with `clear` and then type `cpufetch`, you will get something like this:



screenfetch

If you clear the screen with `clear` and then type `screenfetch`, you will get something like this:



You can exit the new window by typing `exit` and your console session will go live again.

Cool retro term

Next try out `cool-retro-term` by typing the `cool-retro-term` command.

```
userid@userid-pi:~ $ cool-retro-term
Both point size and pixel size set. Using pixel size.
```

This pops up a legacy CRT themed terminal emulator if you like old school things. Looks great for vintage emulation.

Other options

You can run other things such as the libre-office suite, image editors, etc. It will work faster with an actual screen installed, but that is a little hard to carry around. There are small screens you can use, and small mice so you can have the actual desktop experience since this is basically a full computer embedded in a keyboard.

Wired connection

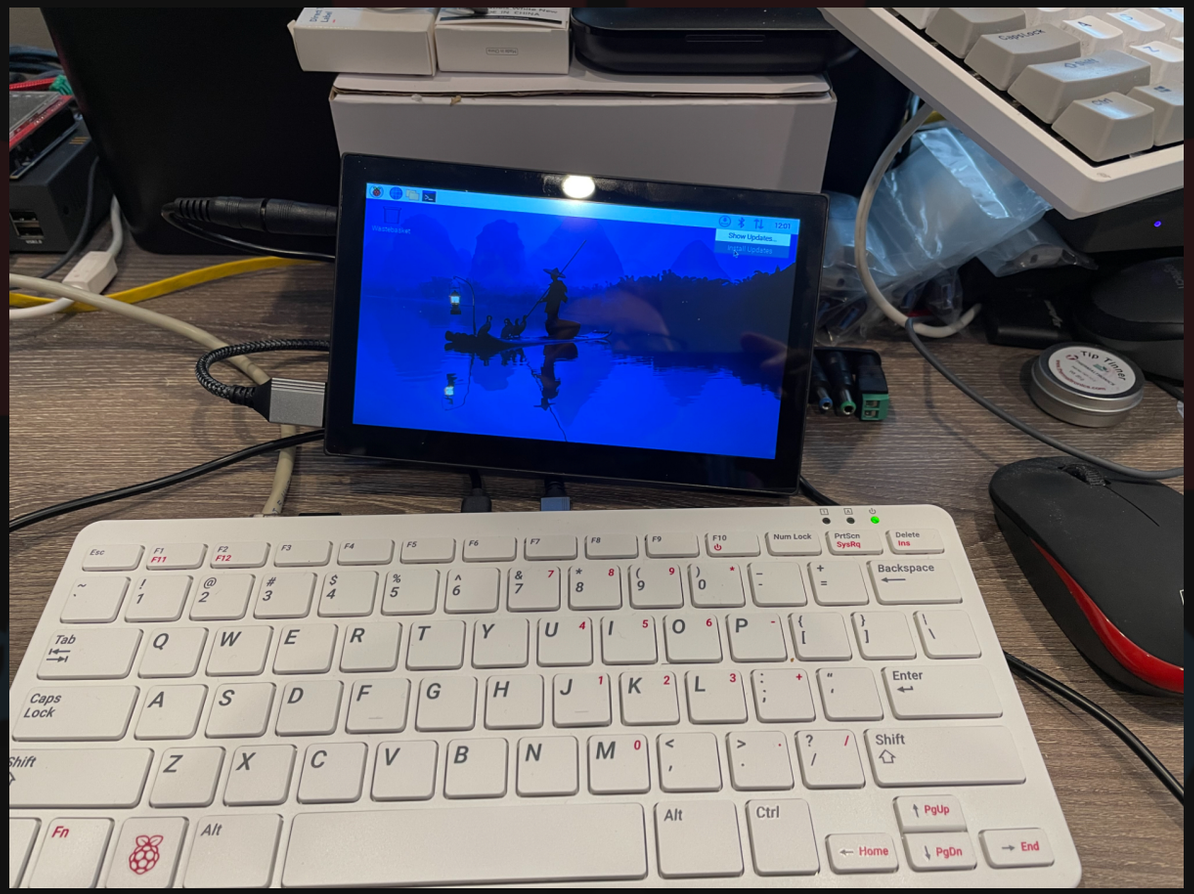
If you physically connect an interface and the network assigns a DHCP address, the IP address will also show up on the console when you log in, so you can also use that to ssh into the unit.

Physical screen

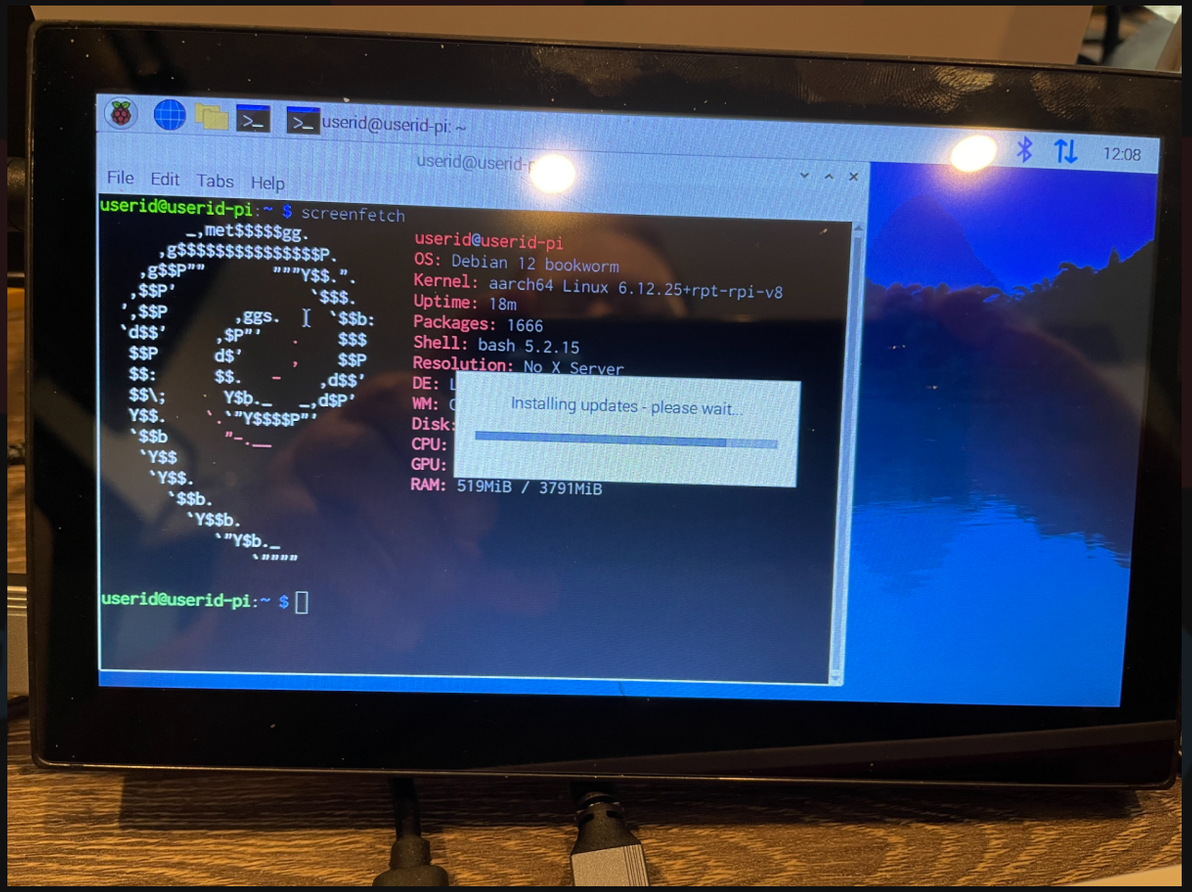
You can use a variety of them and all modern screens will get properly identified for size. There are many options, here are two small ones, both of which require external power. The physically smaller one is also a touch screen, so if you plug in a USB connector to it and the Pi, it will act as one.

I have a couple of pictures to show how this works

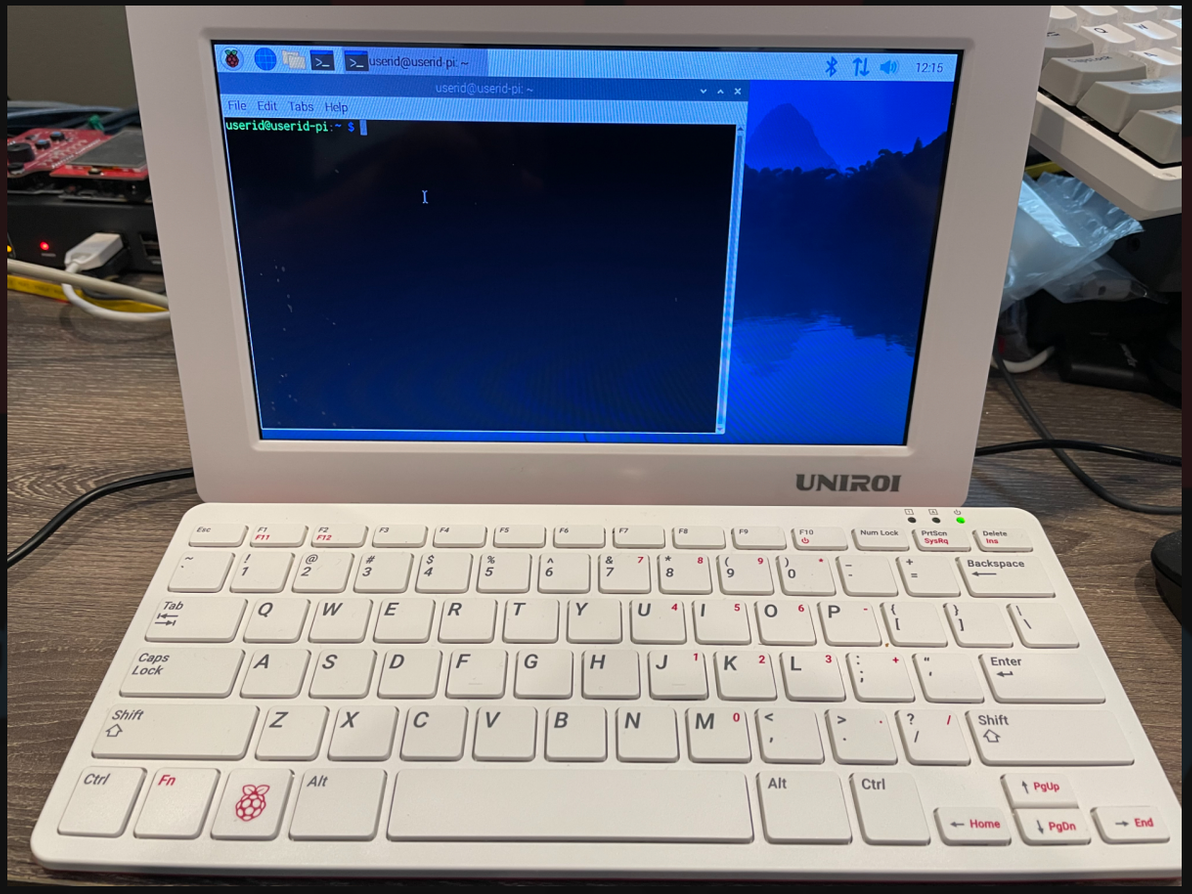
Physical screen 1



Physical screen 1 (closeup)



Physical screen 2



References

Links to the items discussed

- Raspberry Pi Imager https://www.raspberrypi.com/software/
- USB Console Stub https://www.pishop.ca/product/usb-console-stub-serial-adaptor-for-raspberry-pi/
- Micro-USB cable https://www.pishop.ca/product/usb-flat-cable-a-microb-white/
- MobaXterm https://mobaxterm.mobatek.net/
- Cygwin https://www.cygwin.com/
- Picocom https://github.com/npat-efault/picocom