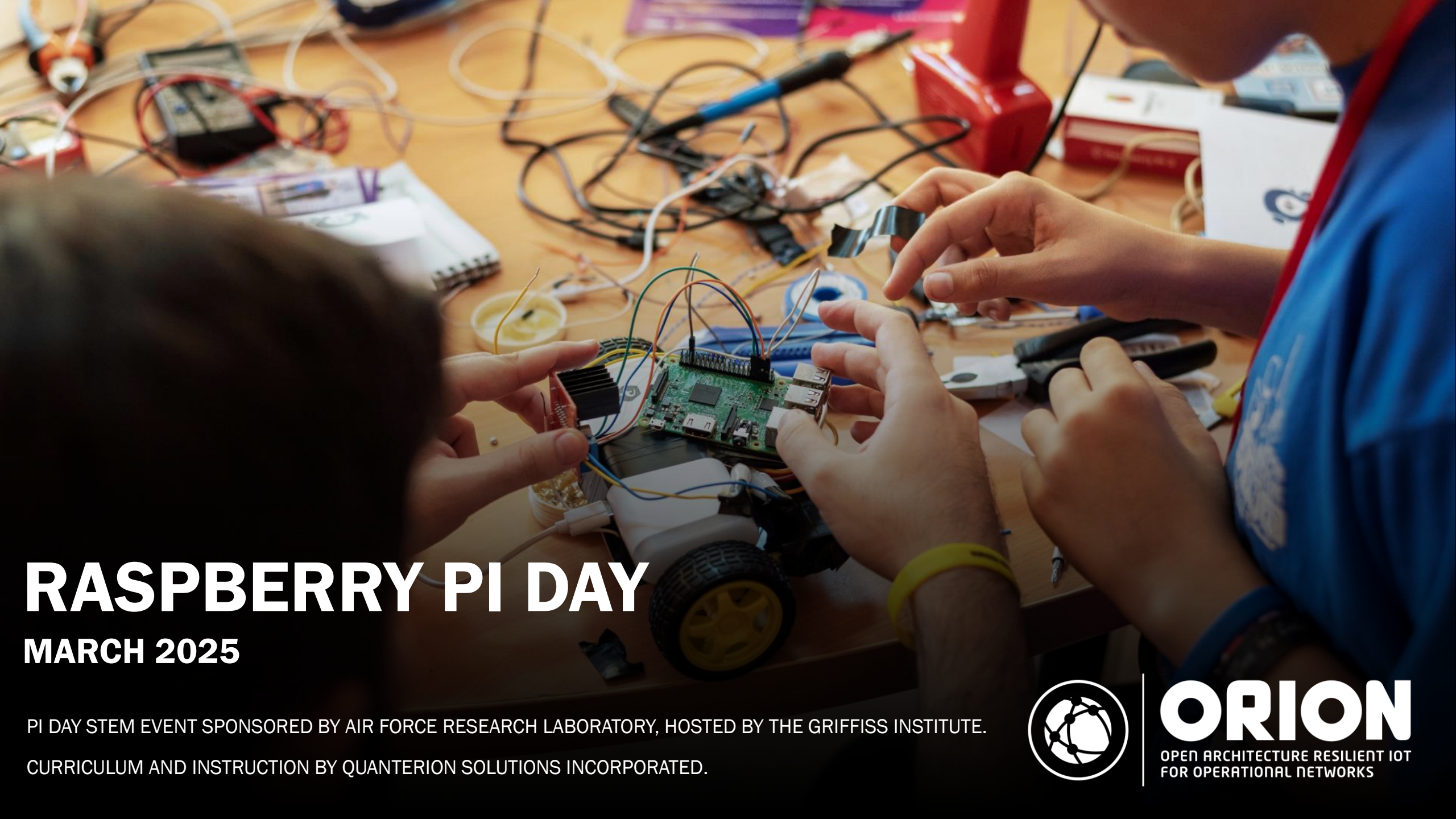


RASPBERRY PI DAY

MARCH 2025

PI DAY STEM EVENT SPONSORED BY AIR FORCE RESEARCH LABORATORY, HOSTED BY THE GRIFFISS INSTITUTE.

CURRICULUM AND INSTRUCTION BY QUANTERION SOLUTIONS INCORPORATED.

ORION

OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

STEM GOALS

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Today's Goals

- Introductions (instructors and mentors)
- Goals for Today…
  - Get excited, learn about single board computers (SBC) – namely, Raspberry Pi (RPi).
    - What is RPi?
      - How are they used?
      - How much do they cost $?
    - Demystify Arduino versus Raspberry Pi
  - Learn about enabling technologies…
    - Linux operating system, IoT (Internet-of-things), Basic electronics
    - Programming (Python and JavaScript)
      - Making hardware and software talk with network protocols: MQTT and WebSockets
- Complete a hands-on project
  - Construct a custom IoT device from scratch
    - Use Raspberry Pi 3b to build a "Smart" (IoT) lamp

Agenda and Schedule - Today (9AM – 3PM)

9:00-9:45

- Introductions
- Griffiss Institute Welcome and Safety Brief
- AFRL – Information Directorate Welcome
- Background and Enabling Technologies

9:45-12:00 : "Smart" IoT Light Bulb Prototype

- **9:45-10:45** : **Module 1: Provisioning the Raspberry Pi for Use**
- **10:45-11:00** : Break (restrooms, stretch, questions)
- **11:00-12:00** : **Module 2: Construct the Smart Lamp Device**

12:00-1:00PM : Lunch and Demo

- **12:30 – 1:00** : Special Demo

1:00-2:00 : **Module 3: Implementing the Software (Front-end and Back-end Code)**

2:00-2:20 : Challenge Questions

2:20-2:45: Team Demos : (5 min each)

2:45-3:00: Wrap up : Issue certificates, Raspberry 5 drawing

RASPBERRY PI DOOR PRIZE!

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Drawing at 2:45pm today

- All camper’s names entered in a drawing for chance to win a latest Raspberry Pi 5

Raspberry Pi 5 Starter Kit PRO - Turbine Black (128GB Edition) (8GB RAM)

2:45pm Survey… Please

If each person can please visit and complete the here before you leave today: https://dafstem.us/stem-camp-student-survey/

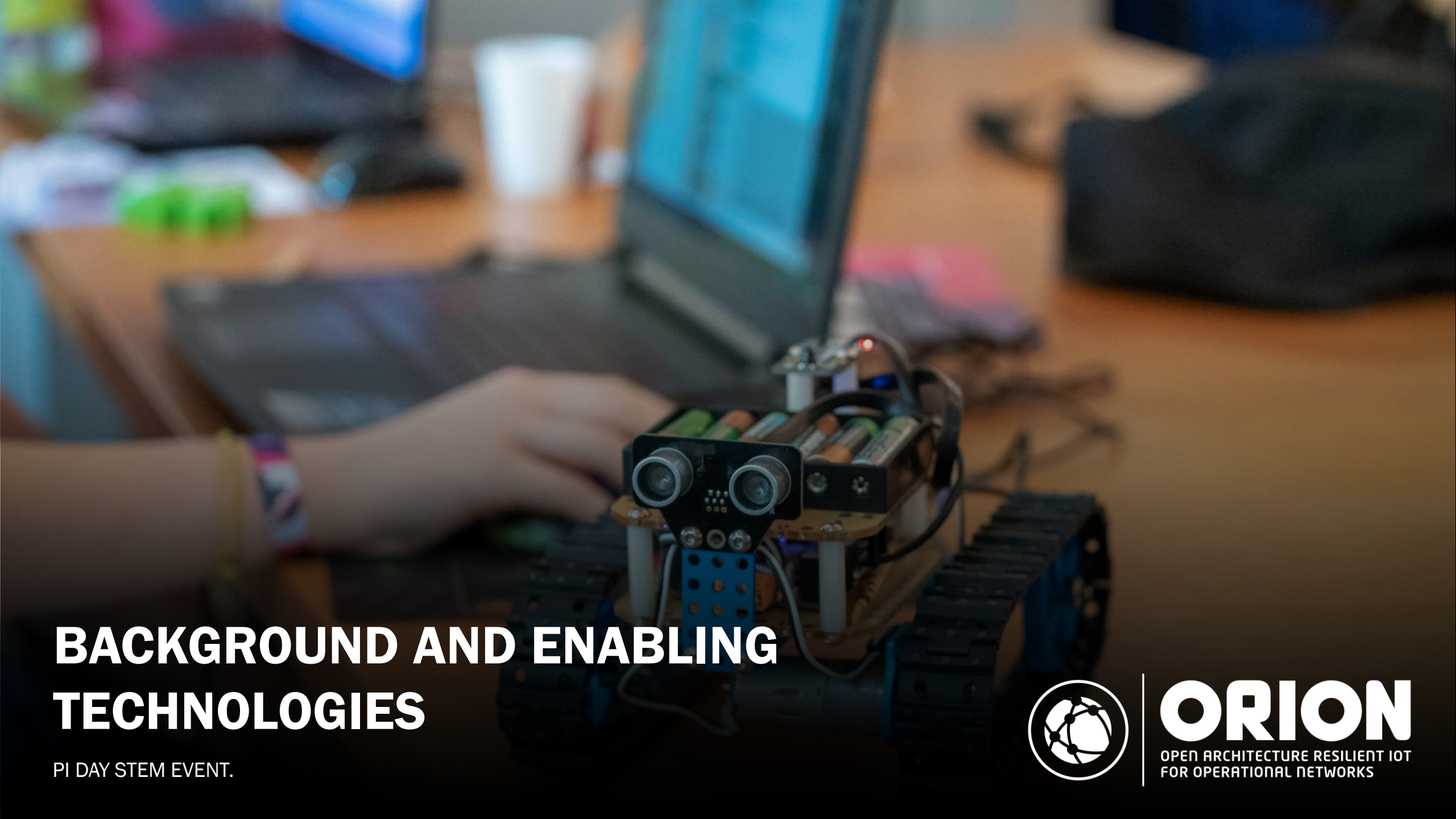
ICEBREAKER

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Icebreaker

- Pick a team name (Air Force Aircraft Theme)
  - F-15 Eagle,
  - F-16 Fighting Falcon,
  - F-18 Super Hornet,
  - F-22 Raptor,
  - F-35 Lightning,
  - A-10 Warthog,
  - B-21 Raider,
  - B-52 Stratofortress
- Pick a team representative
- Question: Nest vs. Ring doorbell?



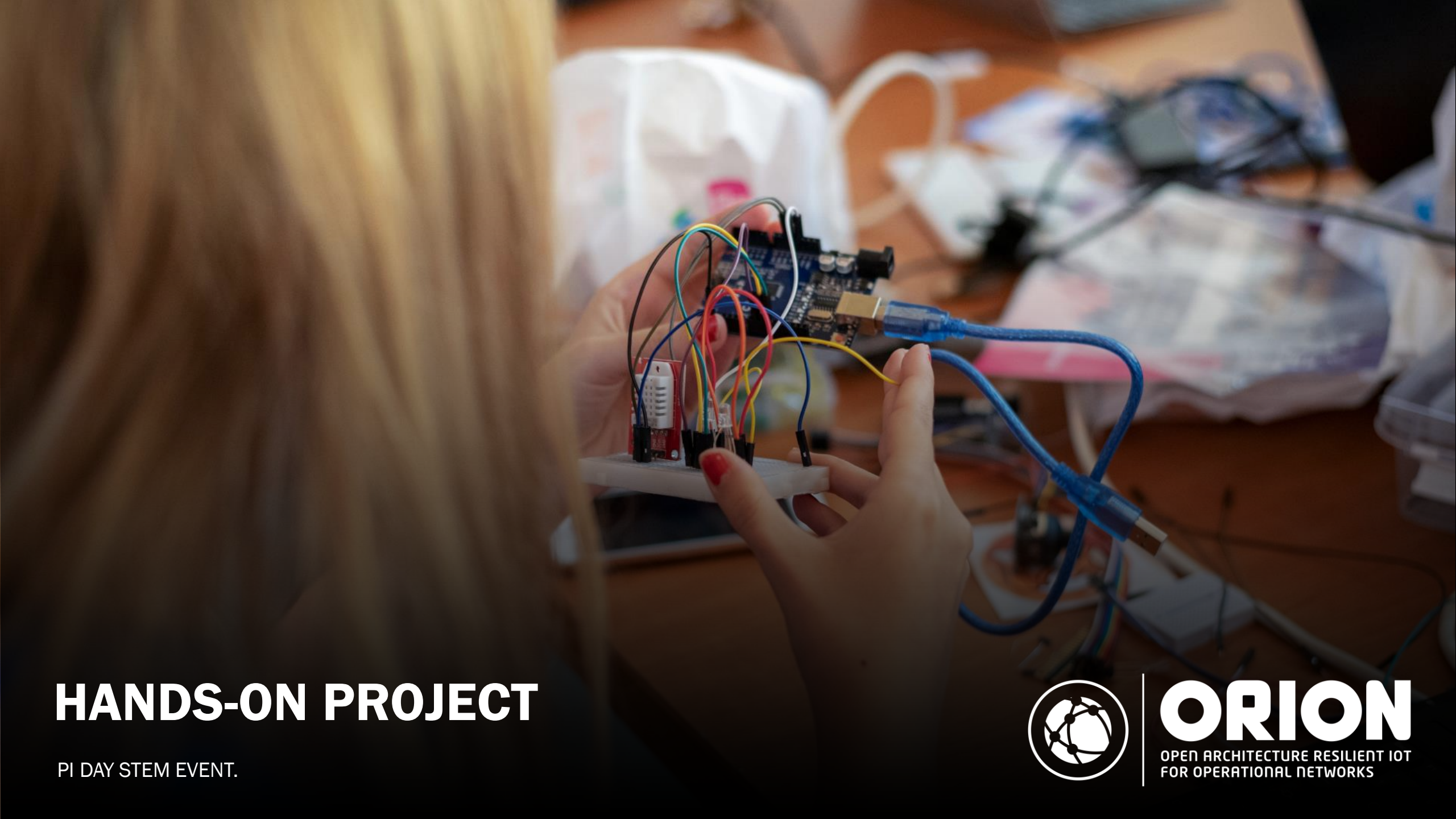
BACKGROUND AND ENABLING TECHNOLOGIES

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Background and Enabling Technologies

- What is a Raspberry PI?
  - https://www.youtube.com/watch?v=eZ74x6dVYes
    - Pi board comparison: https://socialcompare.com/en/comparison/raspberrypi-models-comparison
      - Uses?
        - industrial automation
          - applications for prototyping
          - embedded systems
          - low-cost process controller
        - Home automation
          - DIY, education
  - Raspberry Pi versus Arduino
    - https://www.youtube.com/watch?v=p40OetppIDg
      - https://webbylab.com/blog/arduino-vs-raspberry-pi-key-differences-comparison-table/
  - Raspberry Pi 5 (latest version)
    - https://www.raspberrypi.com/news/introducing-raspberry-pi-5/
  - What is Internet-of-things (IoT) all about anyway
    - https://www.youtube.com/watch?v=6mBO2vqLv38



HANDS-ON PROJECT

PI DAY STEM EVENT.

ORION

OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Hands-on Project: IoT “Smart” Lamp

Web app: (based on HTML 5 CSS and JavaScript)

MQTT over WebSockets

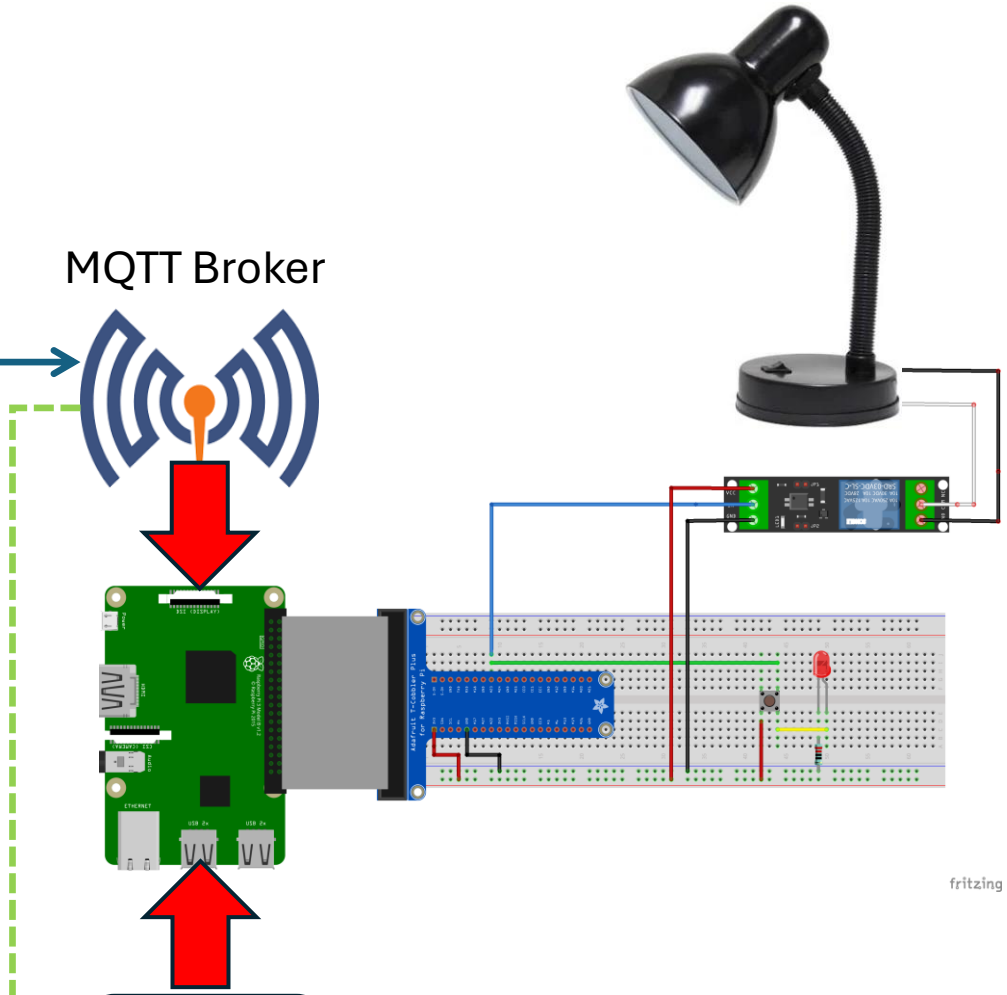
Publish in topic: *Bulb/*

Message: *“on”/”off”*

Subscribe to topic: *Bulb/*

Bulb/”on”

Bulb/”off”

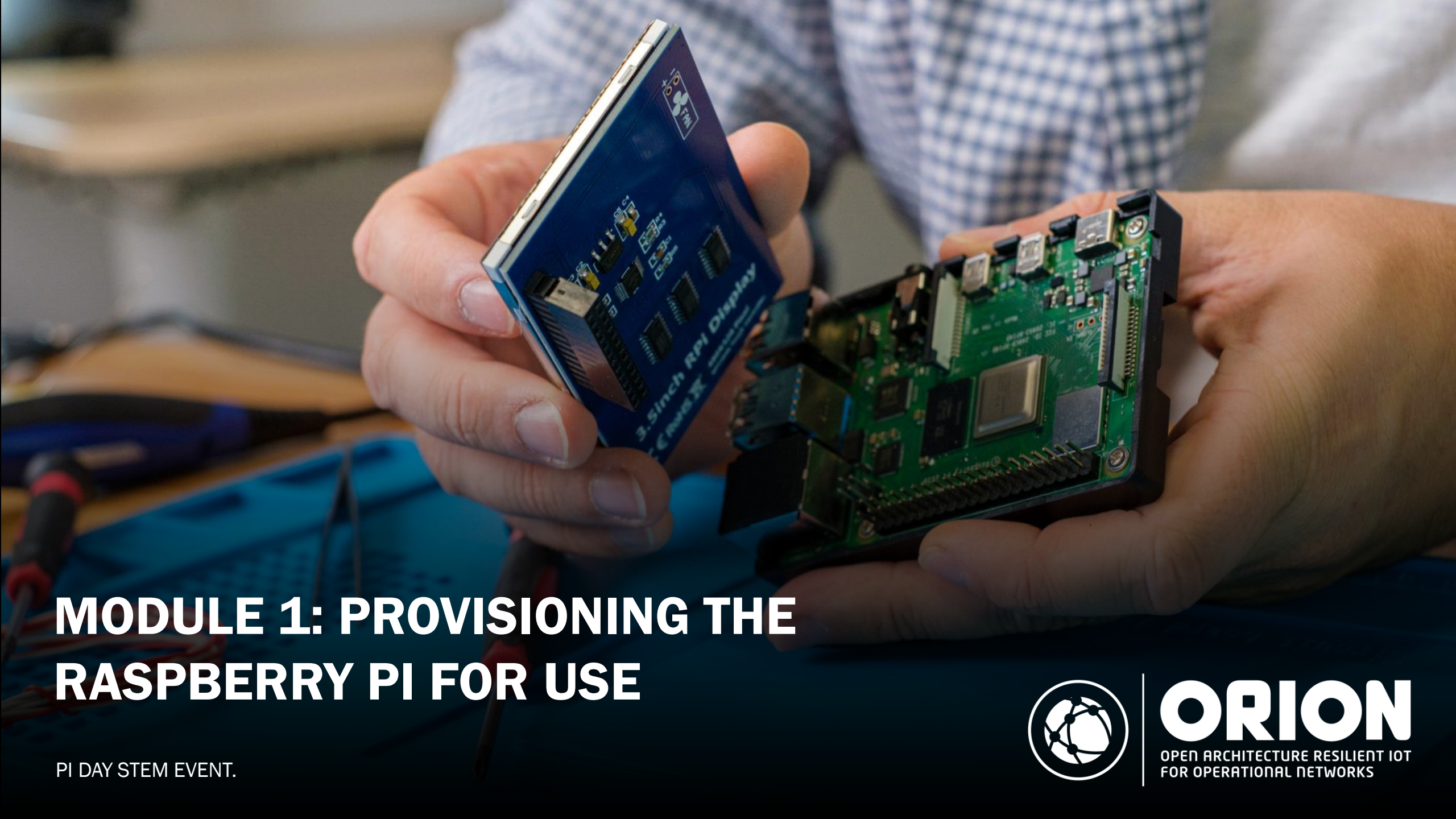


Python Backend

Python smart bulb logic running on RPi

Hands-On Project Breakdown

- Build an IoT “Smart” Lamp Prototype using a Raspberry Pi and an ordinary desk lamp
  - Module 1:
    - Provision The Raspberry Pi for Use
      - Write the Raspberry Pi OS (operating system) Image and configure the services
  - Module 2:
    - Build the device: (hardware): an electronic circuit to connect the desk lamp to the Raspberry Pi
  - Module 3:
    - Implement the software (back-end and front-end) to enable the user to control the lamp using a web application
      - Use the standard IoT communication protocol (MQTT) protocol to enable the parts communicate.
        - https://mqtt.org/



MODULE 1: PROVISIONING THE RASPBERRY PI FOR USE

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Module 1: Provisioning the Raspberry Pi for Use

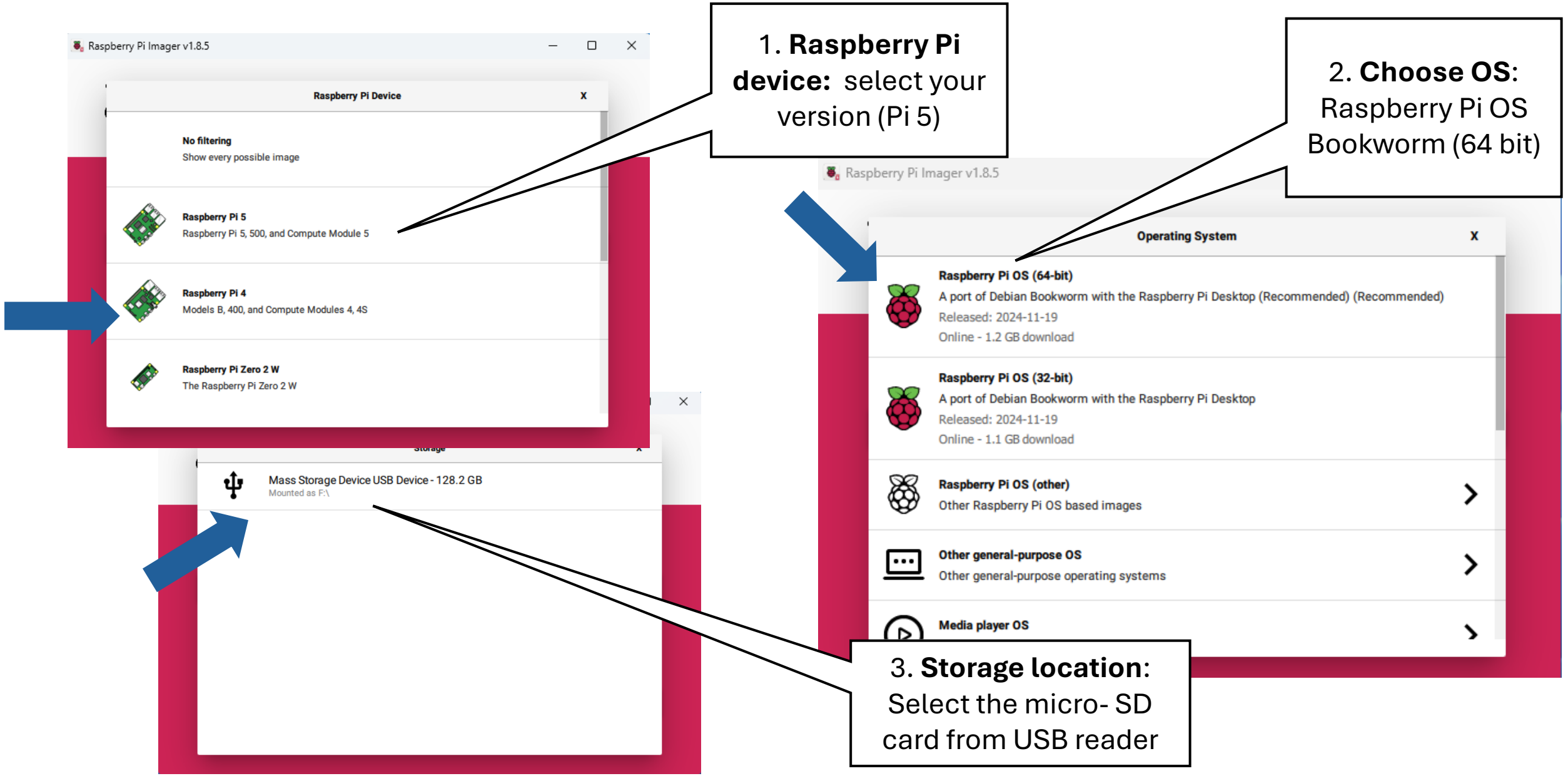
- Write the operating system image: Raspberry Pi OS
  A. Configure services
     - Secure Shell (SSH)
       - raspi-config
         - I2C
         - Virtual Network Computing (VNC)
  B. Install and Configure Mosquitto MQTT Broker
  C. Linux Primer

Install and Configure the Operating System (OS) and Services

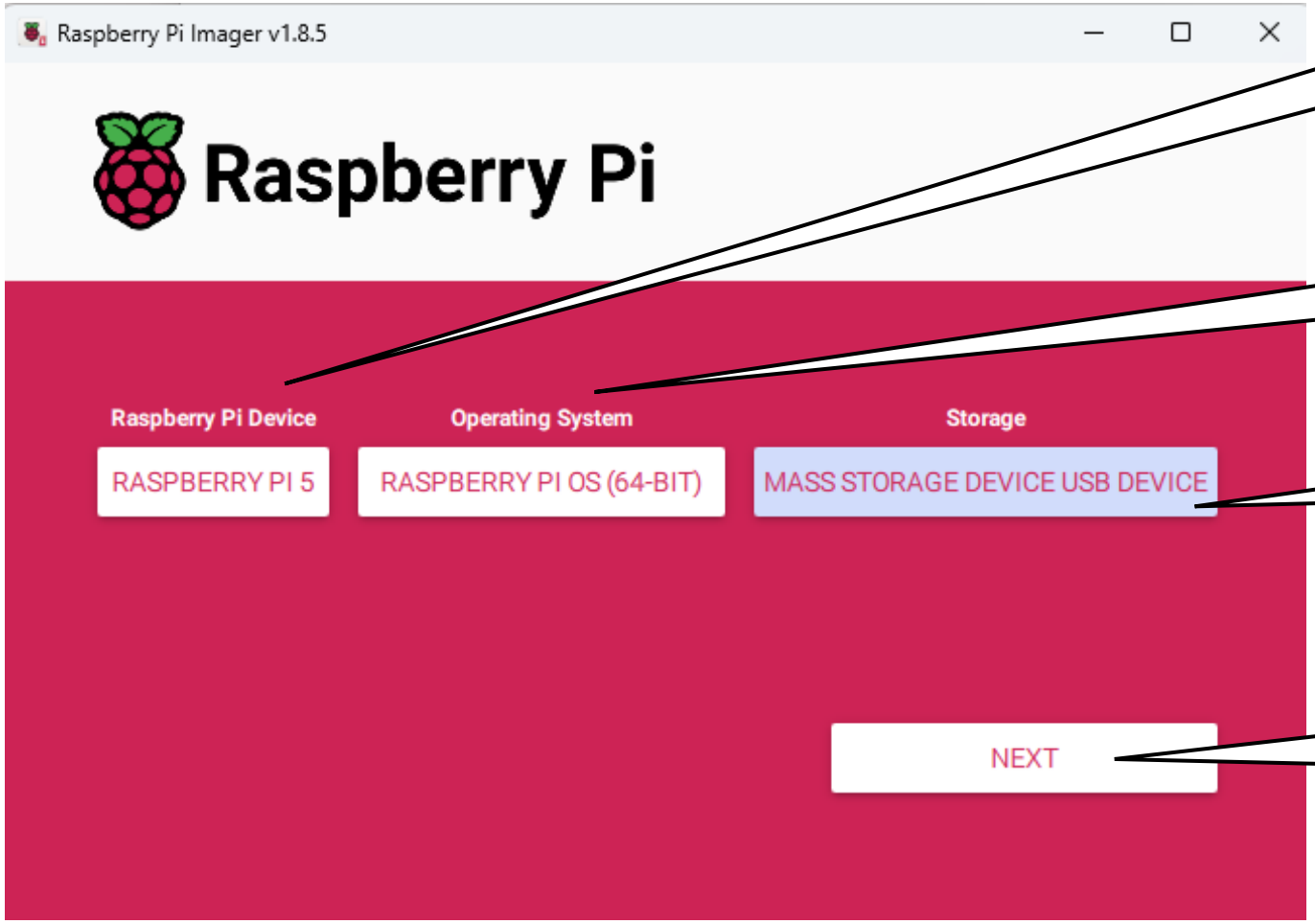
1. **Download and Install Raspberry Pi Imager on Laptop**
   - **Select “Download for Windows”**
     - https://www.raspberrypi.com/software/

2. Insert micro-SD card into laptop USB card reader and run the imager tool. See screen shots below

Select and Write the Operating System (OS) Image to the SD card



Before you proceed, Make sure....



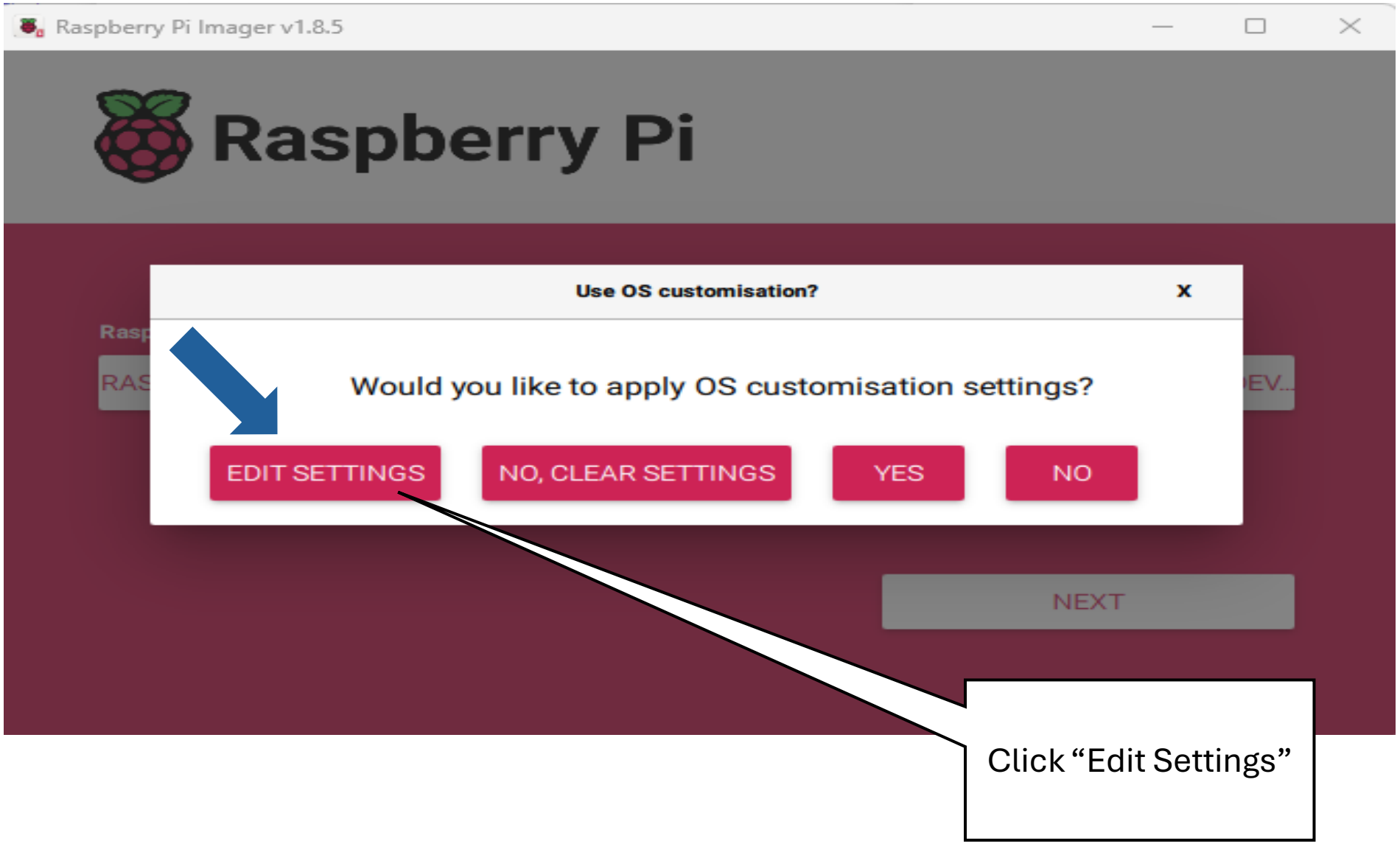
1. Raspberry Pi 5 is selected

Raspberry Pi OS Bookworm (64 bit)

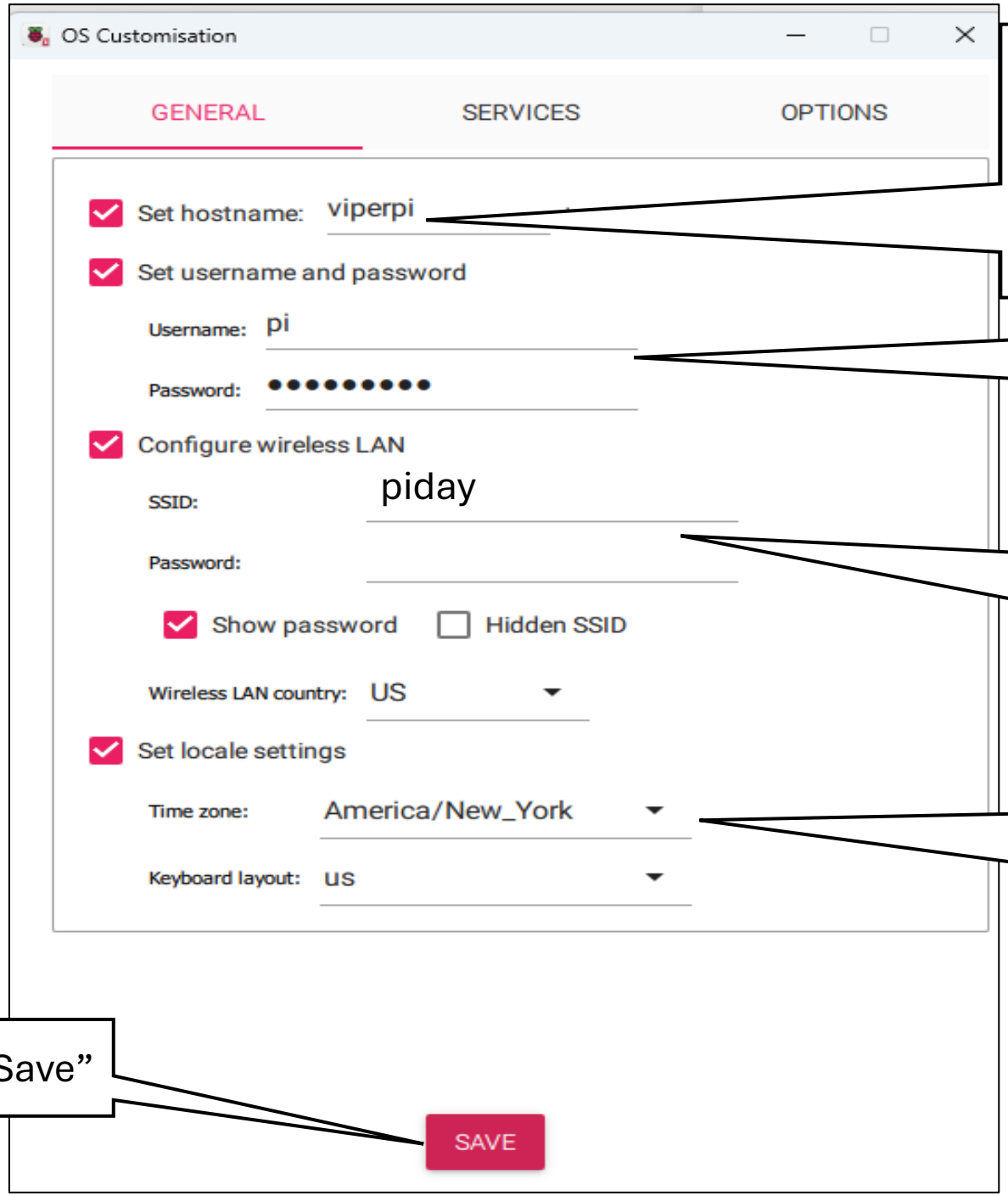
3. Make sure the micro-SD card is selected as the storage

* Only when 1,2, and 3 are verified, click “next”

Customize the Settings:



These settings must be as specified, or your Pi will not be properly configured for the project



Specify the hostname. Use your Team name with “pi” appended. For example, Team “Viper”, would use hostname “viperpi” as shown

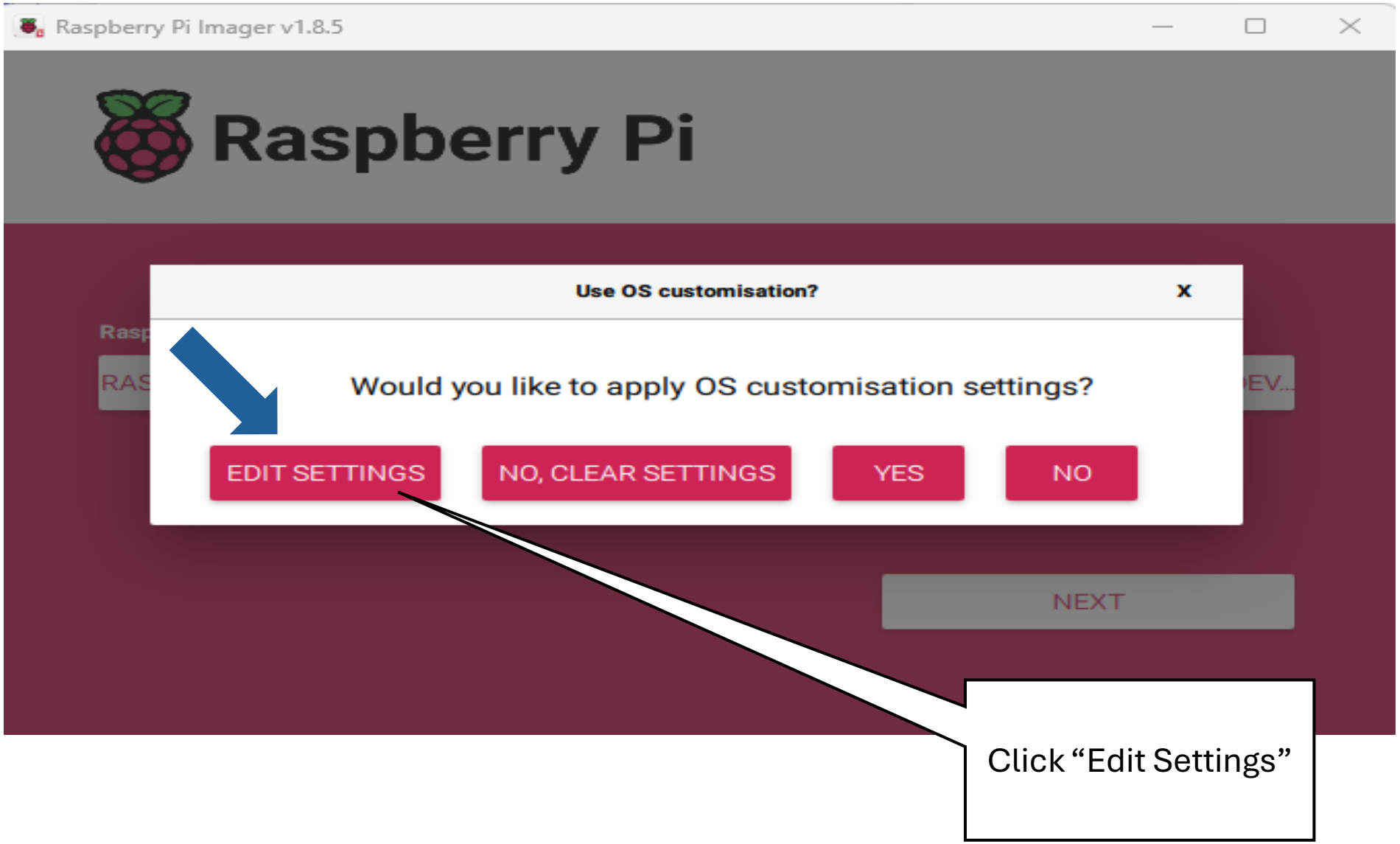
Set the username to “pi” and the Password to “team chooses” (*make it simple to remember*)

Configure the WIFI settings to use the Griffiss institute Guest network.

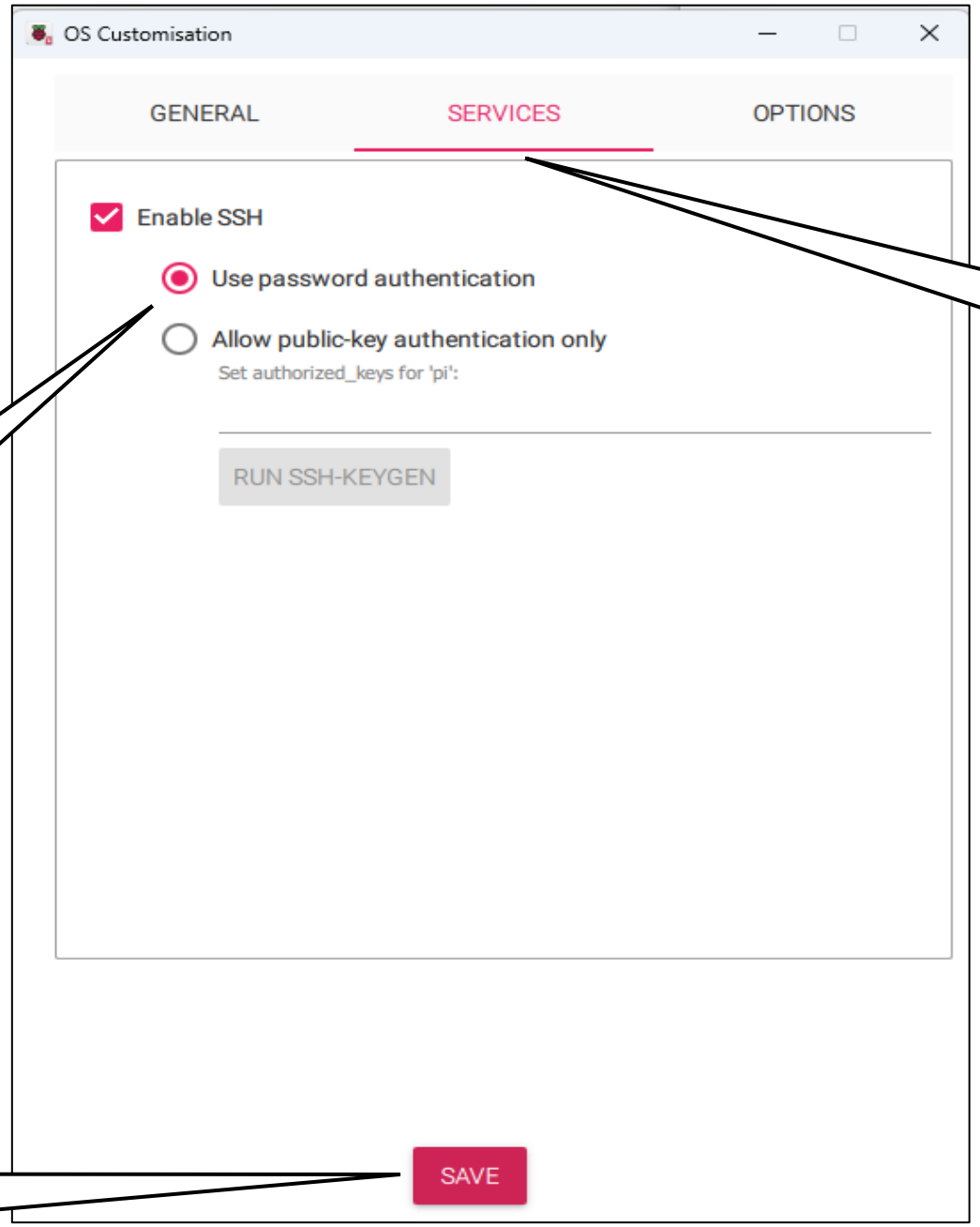
Select the country to ‘US’ and set the locale settings as illustrated

Click “Save”

Customize the Settings:



Enable Secure Shell

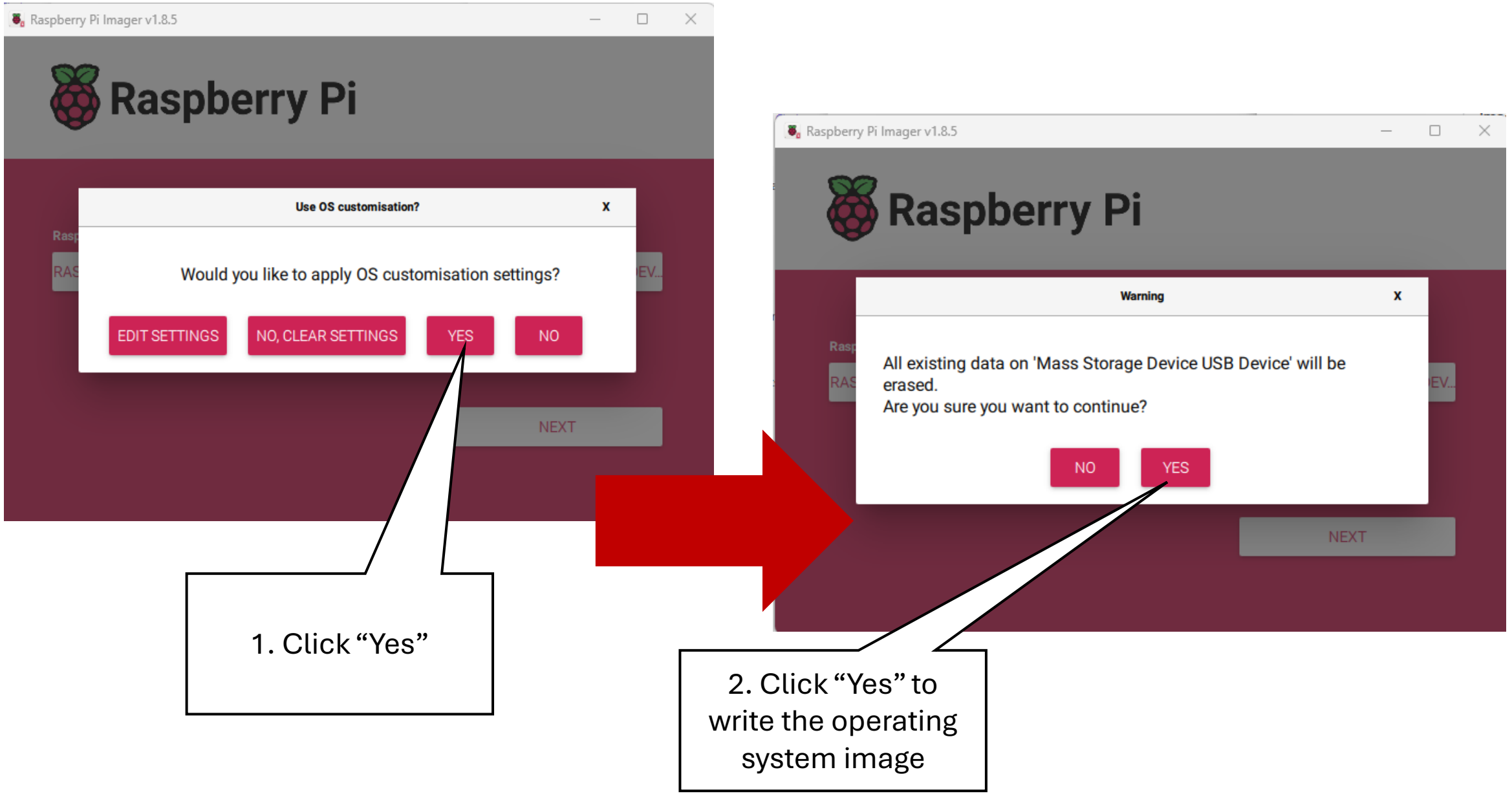


1. Click “Services” Tab

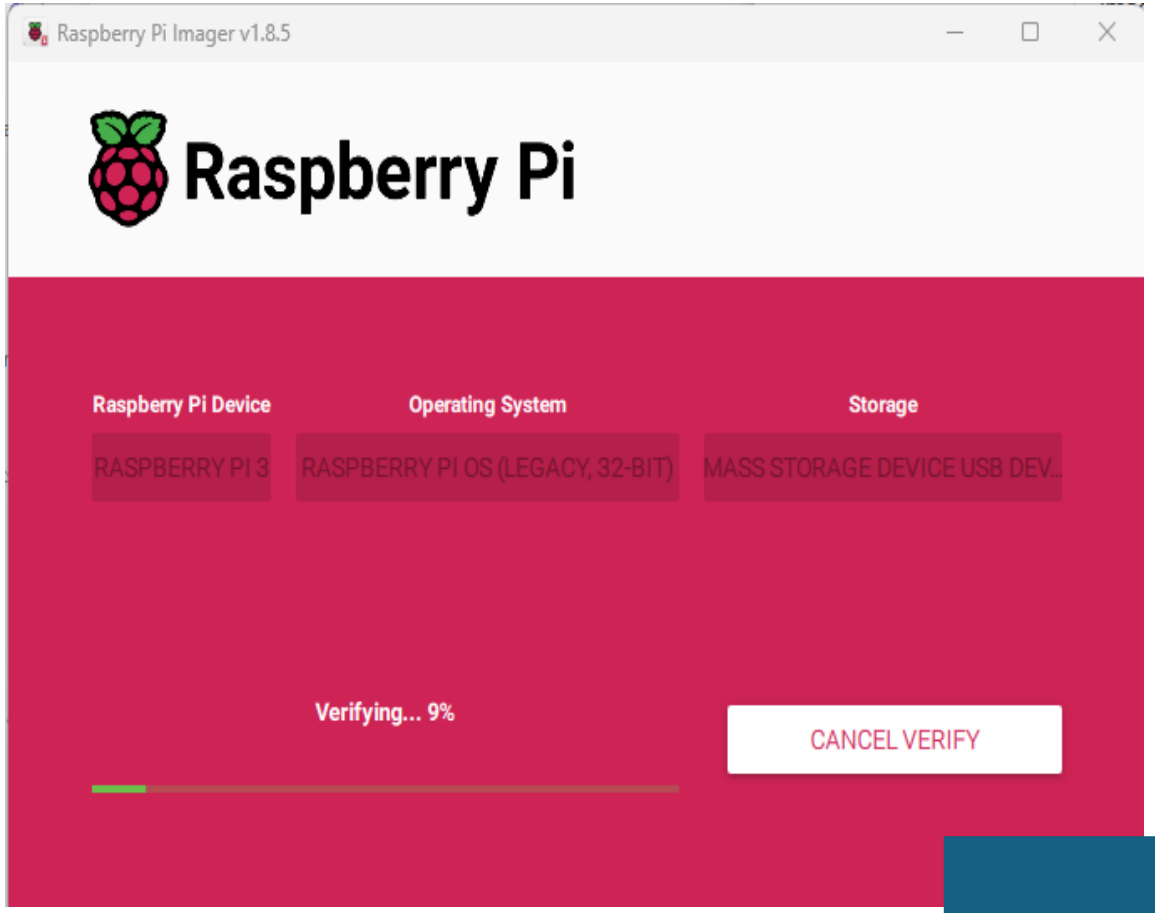
2. Choose password authentication

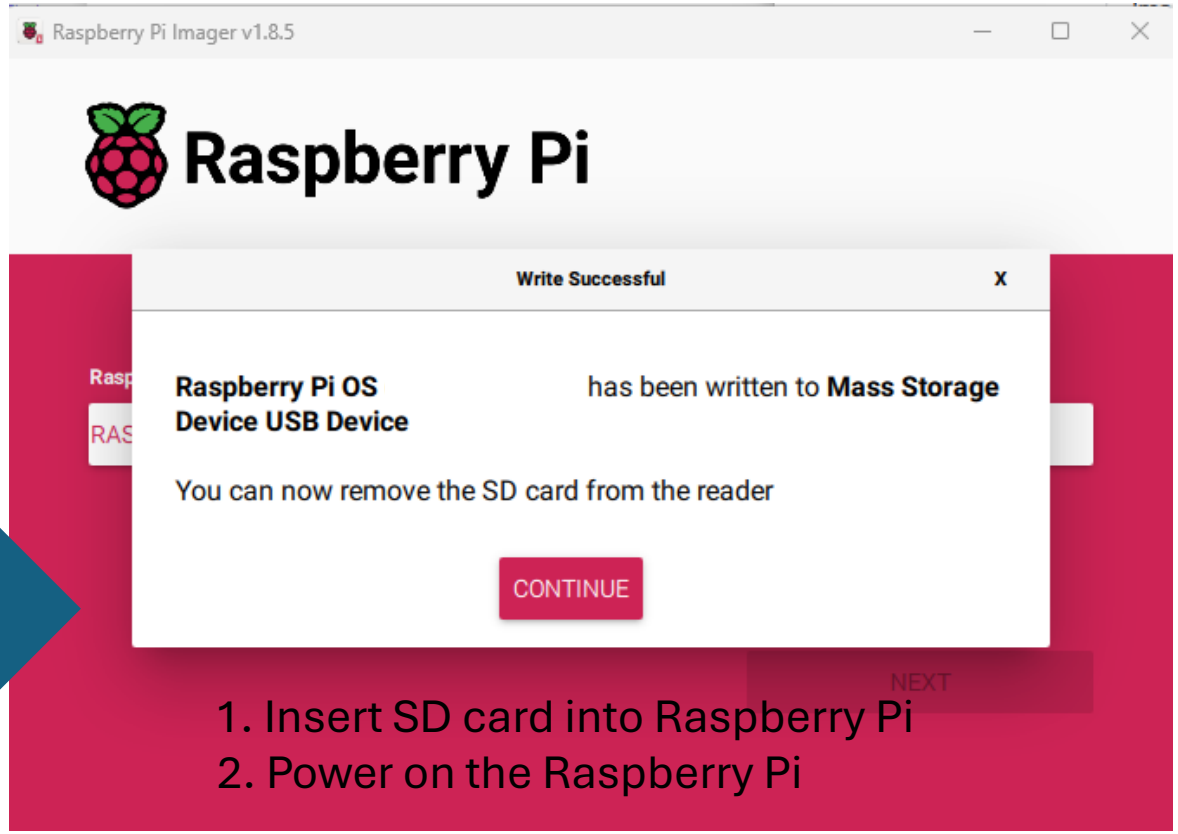
3. Save settings

Customize the Settings:

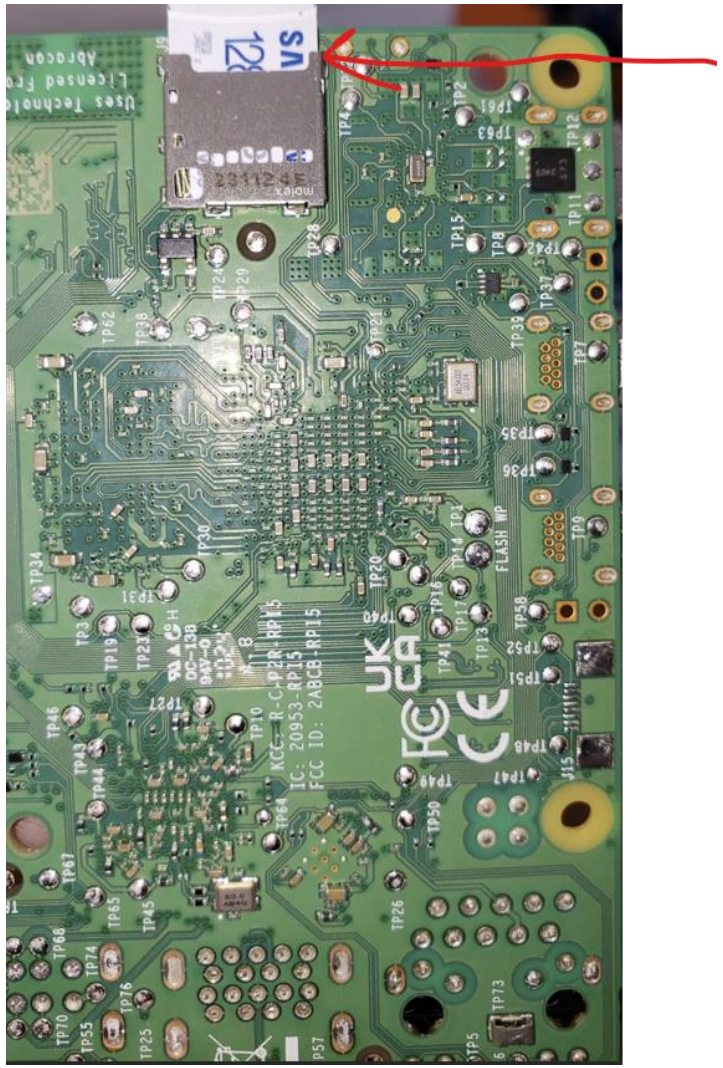


Wait for the OS image to be written to the SD-card





Insert the SD card into the Raspberry Pi



Note: you will need the IP address issues the raspberry Pi for the next steps

- There are number of ways determine the IP address. One command way is log into your home router and identify the IP in the list of the connect WIFI clients.
- A second (perhaps easier) way is if your home router defaults to multi-cast DNS on, you can use the hostname you specified on slide 20 in place of the IP on the next slide

Start an SSH session with the Raspberry Pi

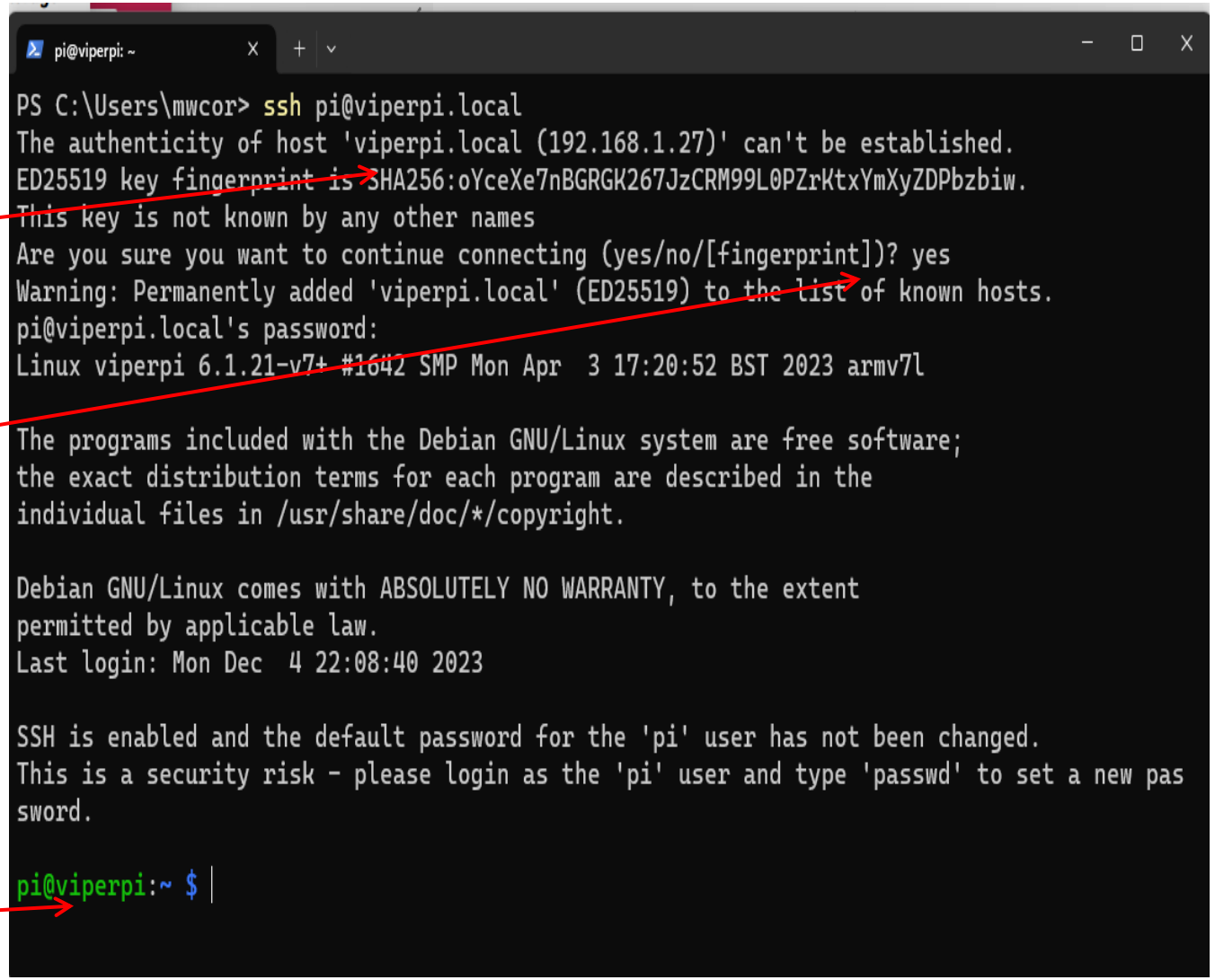
1. Open a new terminal window
2. Type command: *pi@ip-address (10.x.x.x)*

OR

pi@hostname.local (with multicast DNS)

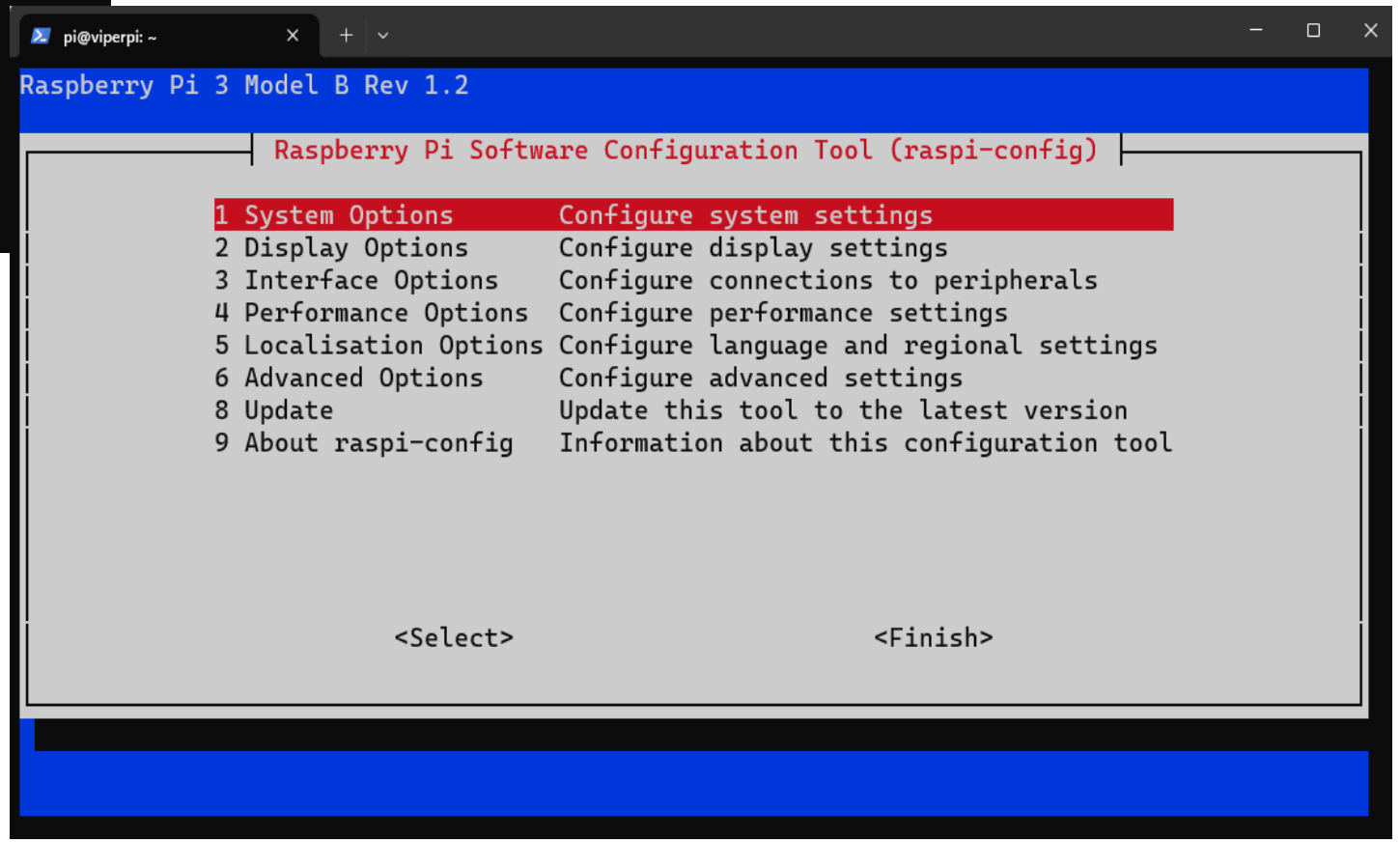
Type command: *yes*

Congratulations. You have a new (remote) connection to the Raspberry Pi



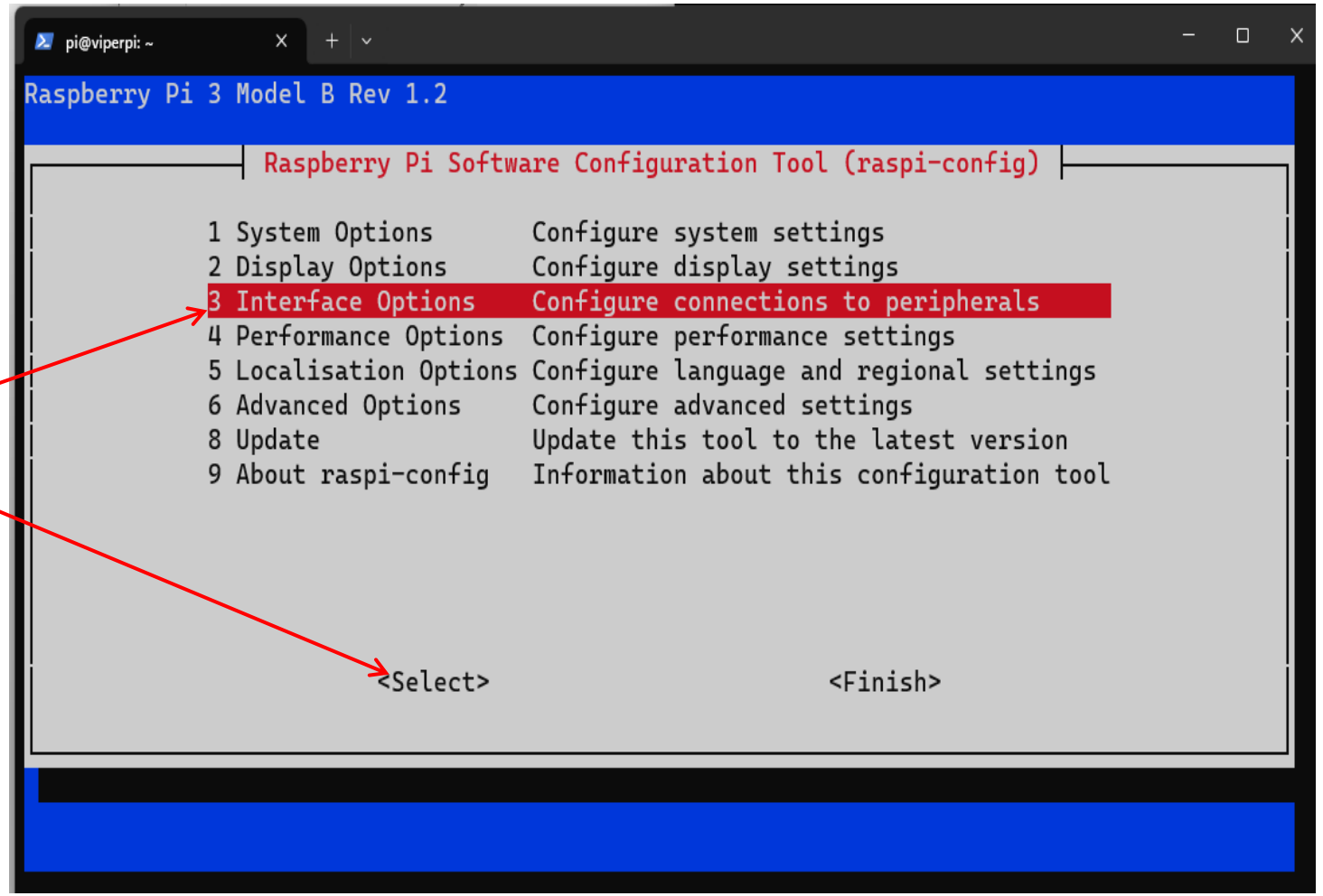
Run the Raspberry Pi configuration tool to configure the services

Type command: sudo raspi-config



Select Configure the Interfaces:

Select menu item 3:



Enable SSH, VNC, and I2C

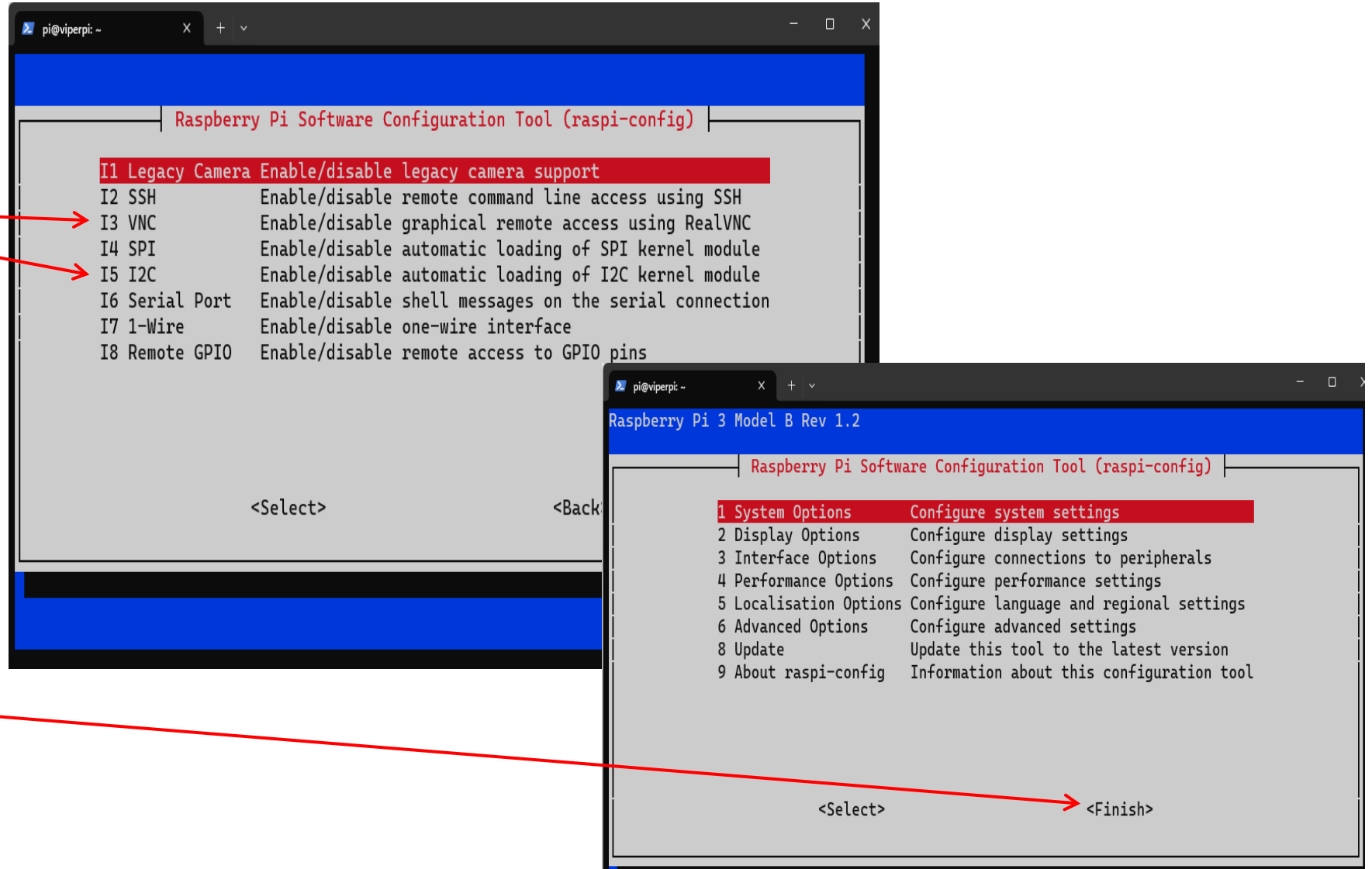
SSH is already enabled

1. Enable both VNC and I2C

I2C *(Inter-Integrated Circuit) is widely used a serial bus protocol for establish serial communication integrated circuits (ICs)*

1. Select Finish to exit

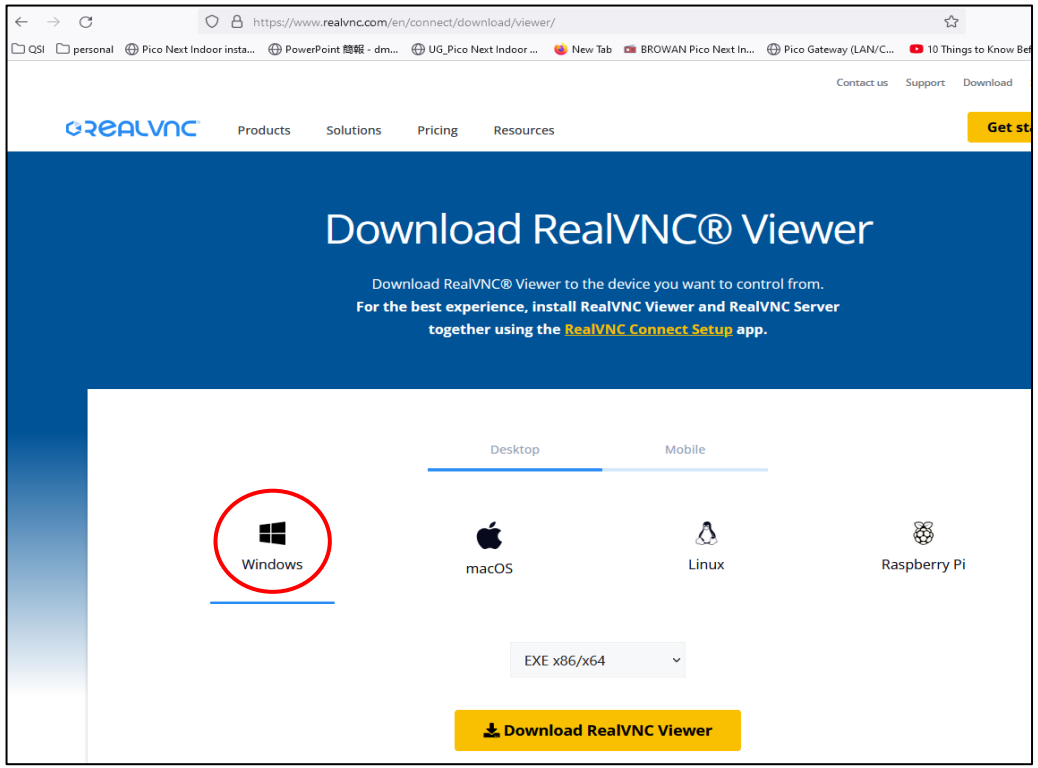
reboot the pi: *sudo reboot*



Setup VNC viewer laptop PC

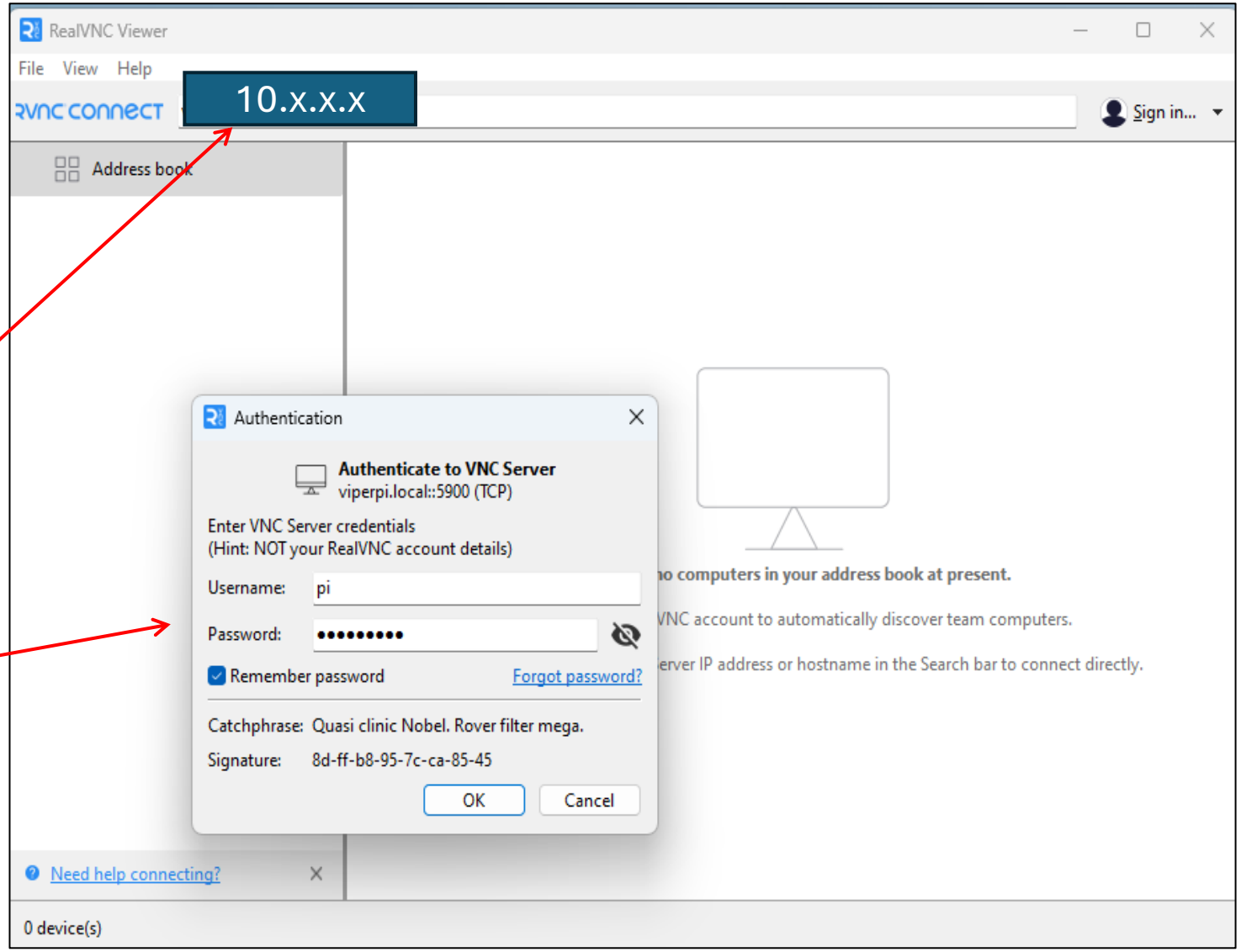
- VNC stands for Virtual Network Computing. It is a cross-platform screen sharing system that was created to **remotely control/view another computer**
  - Download the VNC viewer application to Windows (host) computer:

    https://www.realvnc.com/en/connect/download/viewer/



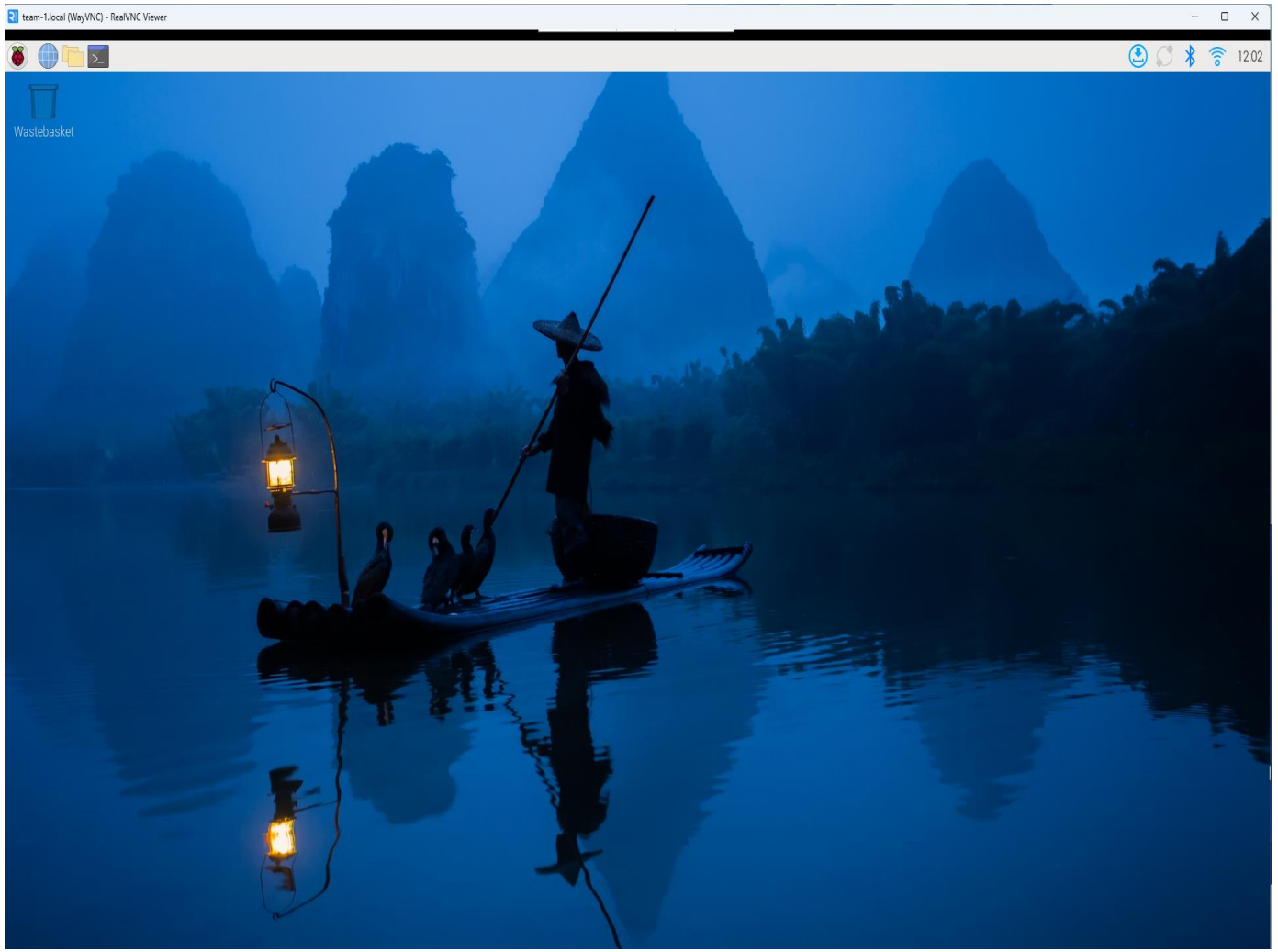
Establish a VNC session with Raspberry PI

- Open the Real VNC viewer and enter the Raspberry Pi host name in the connect box.
    - Enter the IP address (10.x.x.x) for your team's pi in the Real VNC viewer connect box as shown and press
    - Supply the username "pi", and the password your team specified.

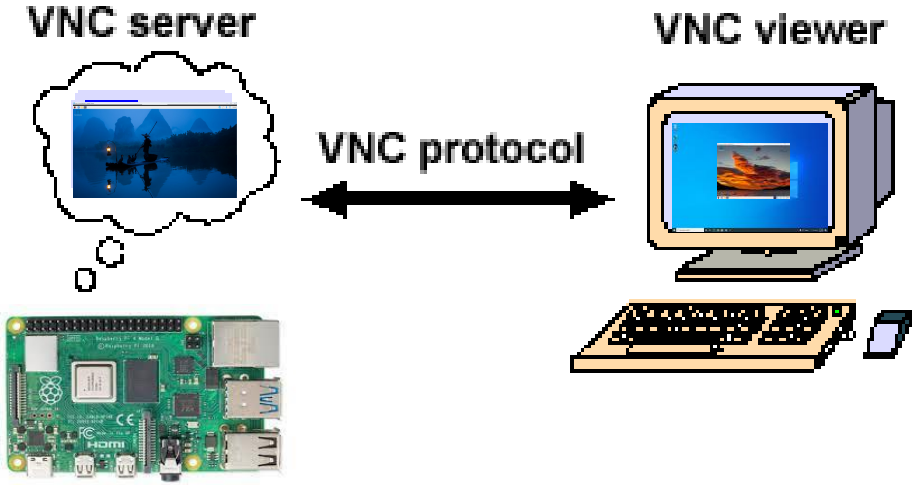


- Example established VNC session

Remote desktop experience, without the need for a physical monitor and peripherals



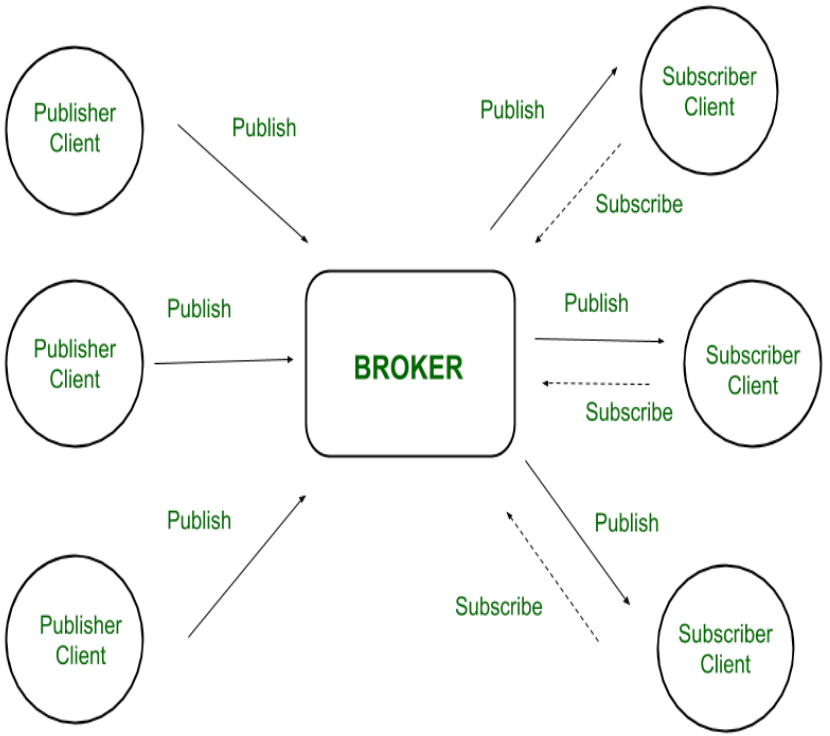
Questions / Recap ?



Source: http://web.mit.edu/cdsdev/src/howitworks.html

MQTT enables communication between the web (user) interface and the Raspberry PI device (the electronic circuit)

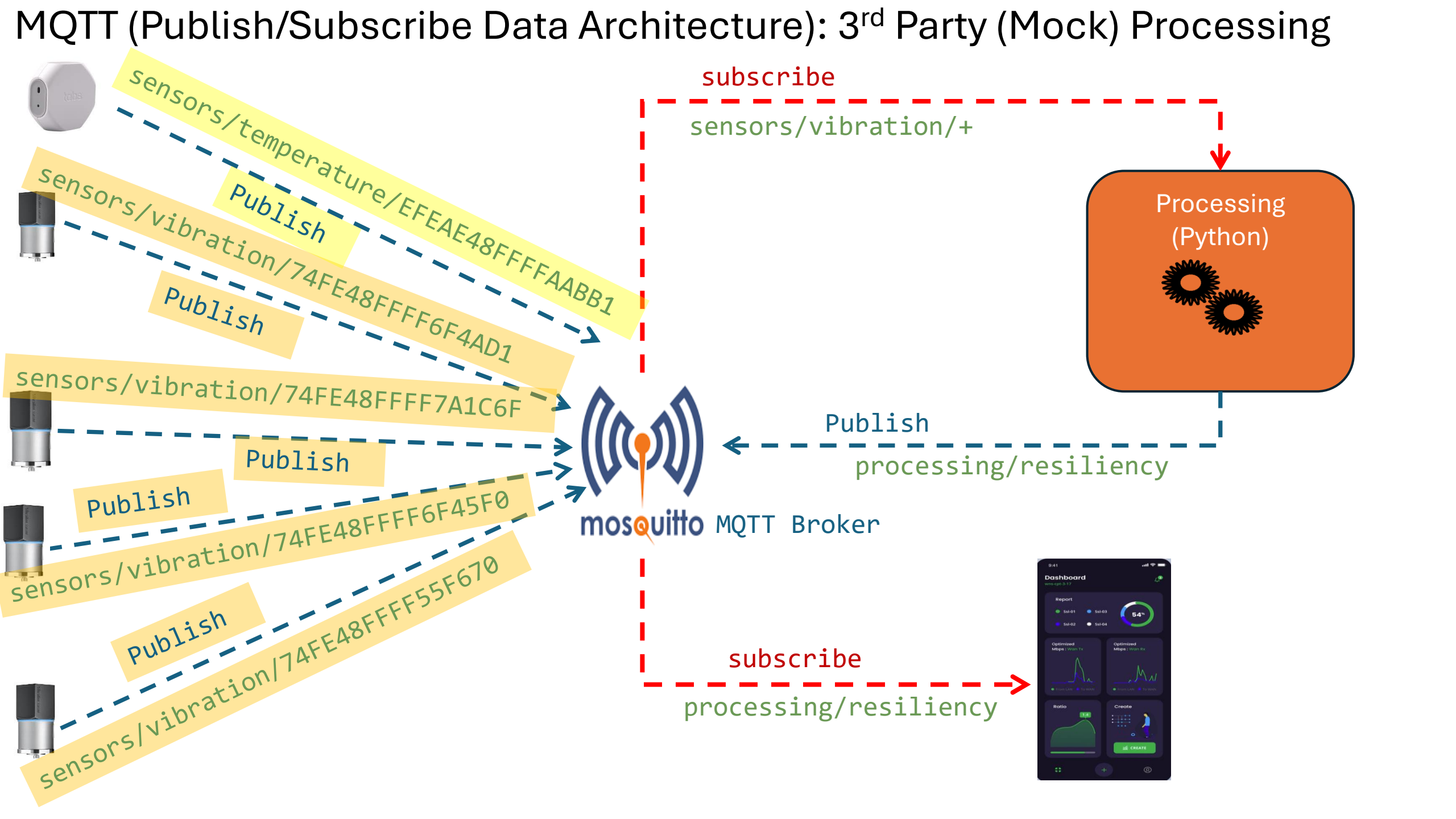
- MQTT stands for **M**essage **Q**ueuing **T**elemetry **T**ransport) -is a simple messaging protocol, designed for constrained devices with low bandwidth. – [1]
    - Perfect for exchanging data between IoT devices.
- https://mqtt.org/
- https://learn.sparkfun.com/tutorials/introduction-to-mqtt/all
- https://www.geeksforgeeks.org/introduction-of-message-queue-telemetry-transport-protocol-mqtt/ - [1]
- https://www.geeksforgeeks.org/fundamental-features-of-mqtt/
- https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/
- MQTT over web sockets (JavaScript) tutorial
    - http://www.steves-internet-guide.com/using-javascript-mqtt-client-websockets/



Source: https://www.geeksforgeeks.org/introduction-of-message-queue-telemetry-transport-protocol-mqtt/ - [1]

```
match individual (single) sensor subscriptions
# sensors/vibration/74FE48FFFF7A1C6F
# sensors/vibration/74FE48FFFF6F4AD1
# sensors/vibration/74FE48FFFF6F45F0
# sensors/vibration/74FE48FFFF55F670

# sensors/sound/E8E1E1000105034E
# sensors/sound/E8E1E1000105036D

# match all levels of hierarchy after #  (all sensors)
# sensors/#

# match a single level of hierarchy after + (all vibration sensors)
# sensors/vibration/+

# match a single level of hierarchy after + (all sound sensors)
# sensors/sound/+
```

MQTT (Publish/Subscribe Data Architecture): 3rd Party (Mock) Processing



IoT “Smart” Lamp

Bulb Web app: (based on HTML 5 CSS and JavaScript)

Publish in topic: *Bulb/*

Message: *“on”/”off”*

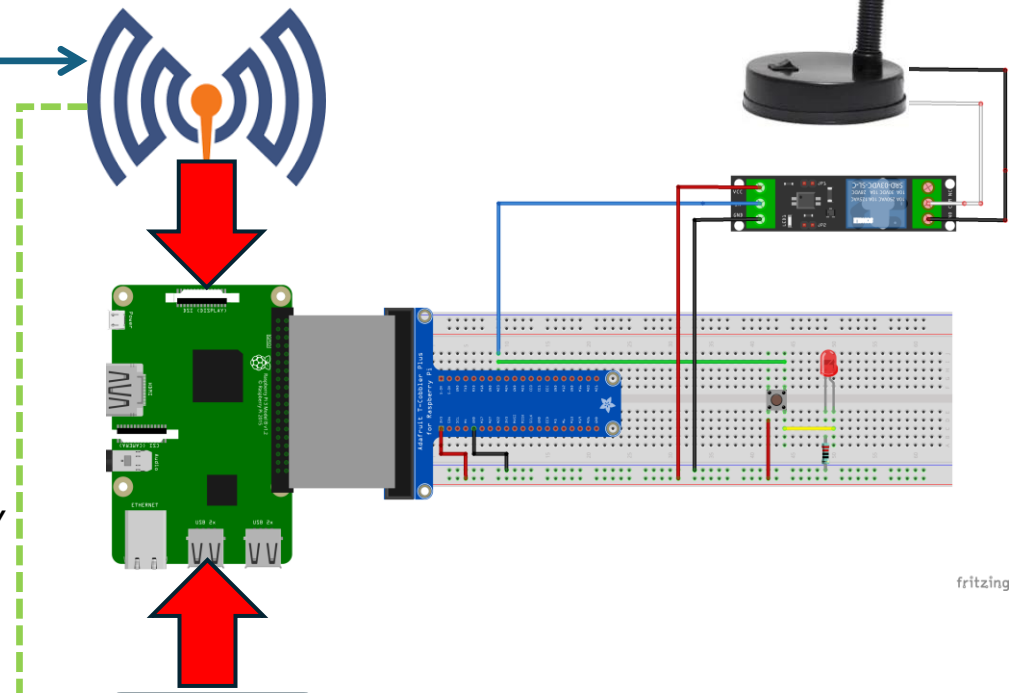
MQTT over WebSockets

Subscribe to topic: *Bulb/*

Bulb/”on”

Bulb/”off”

MQTT Broker on RPi



Python Backend

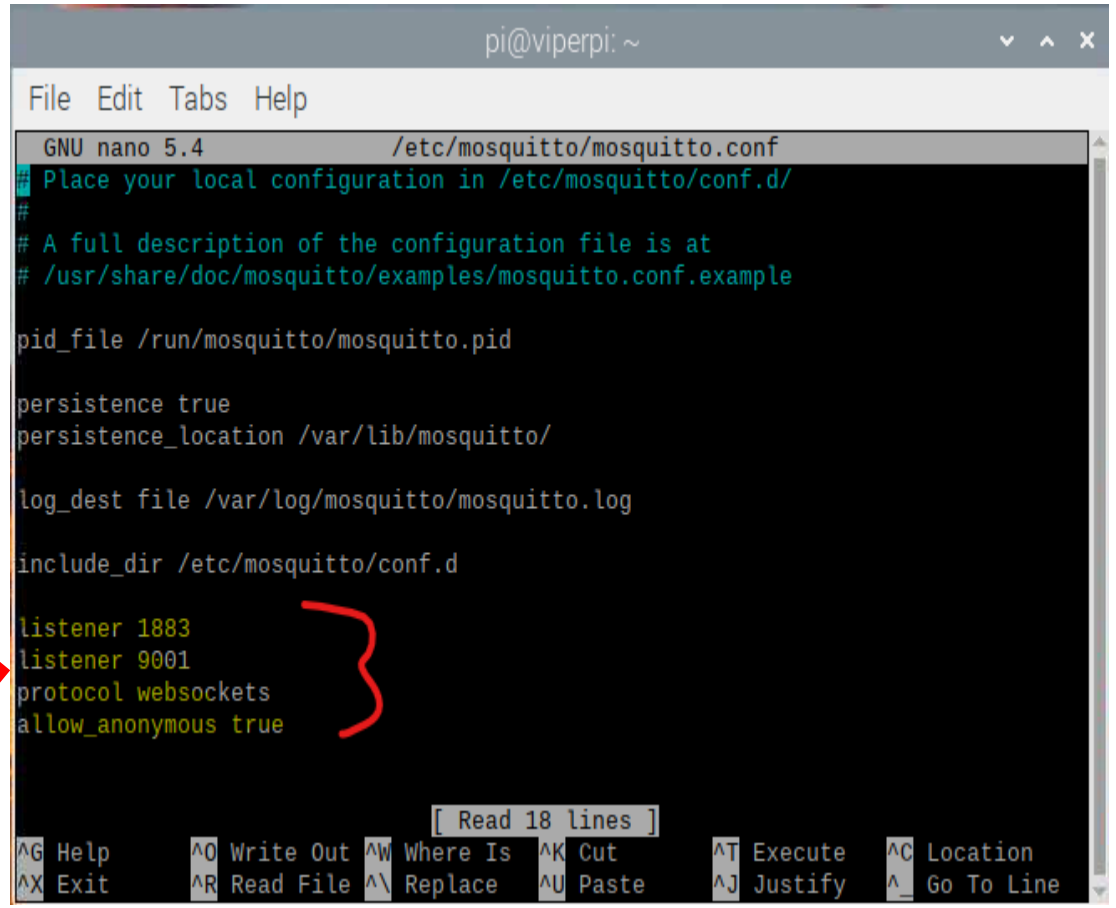
Python smart bulb logic running on RPi

MQTT Broker configuration: WebSockets

- Following this link to install and configure the Mosquitto MQTT Broker:
  - https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/
- Websockets reference:
  - http://www.steves-internet-guide.com/mqtt-websockets/

- Start (or use existing) SSH session with the Pi
  - Use *nano* or *vi* to open the configuration file: *sudo nano /etc/mosquitto/mosquitto.conf*

listener 1883
listener 9001
protocol websockets
allow_anonymous true



- Append the highlighted configuration and save
  - To accomplish a save operation, you need to press Ctrl+o (lowercase “o”) <enter>, which is not intuitive.

- Restart the service: *sudo systemctl restart mosquitto*

Interlude: Short Linux Primer

- Wait! Isn't this camp about Raspberry Pi? Why are we pausing to talk about Linux?
  - Ans: Linux is an operating system that runs almost everywhere! In almost all IT technical careers, you will be required to know and use Linux with proficiency.
    - Linux runs in more environments than you may realize:
      - Big enterprise datacenters, Cloud service backends, your Android phone, smart TVs, cameras, wearables (smart watches), etc.
        - And yes… it is the default OS that runs on your Raspberry PI.
          - Called Raspberry Pi OS (based on the Debian GNU/Linux distribution)
            - https://en.wikipedia.org/wiki/Raspberry_Pi_OS

- Reasonable (Funny) Video Intro to Linux
  - https://www.youtube.com/watch?v=zA3vmx0GaO8

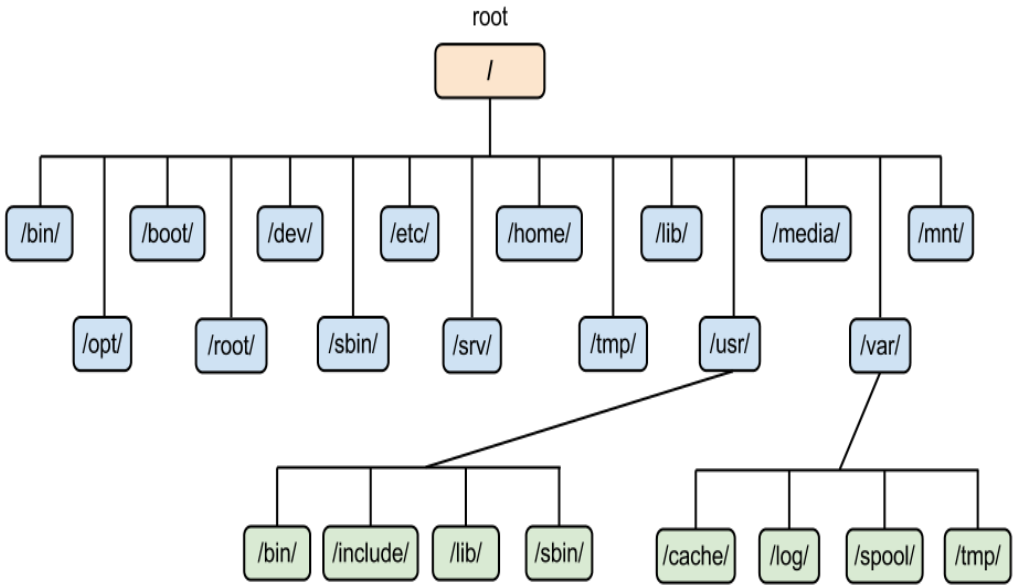
Linux Command Line

- Shell - CLI (command line interface) primer user interface to system. Its essentially a program that interprets user commands, and run other programs

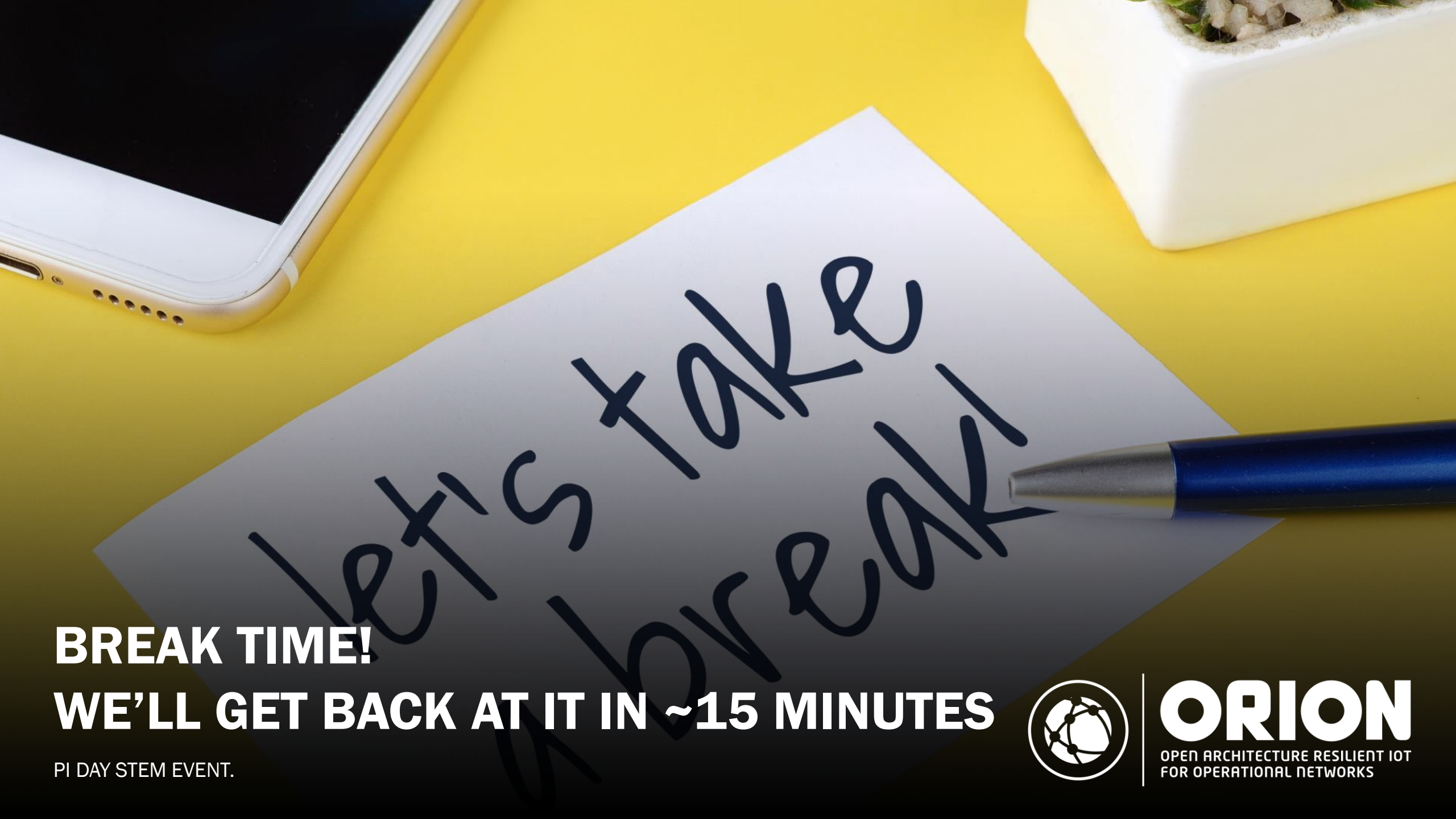
- Understanding the Filesystems
  - Files and folders (called directories), and are organized hierarchically (in a tree structure)

- Basic Linux Commands (video)
  - https://www.youtube.com/watch?v=bb_ApuH15YE

- Further study reference: “The Linux Command line” ebook
  - https://linuxcommand.org/tlcl.php



ls – list contents of folder
~ refers to home folder
cd – change to new directory (folder)



BREAK TIME!
WE'LL GET BACK AT IT IN ~15 MINUTES

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

MODULE 2: PROTOTYPING THE IOT “SMART” LAMP DEVICE

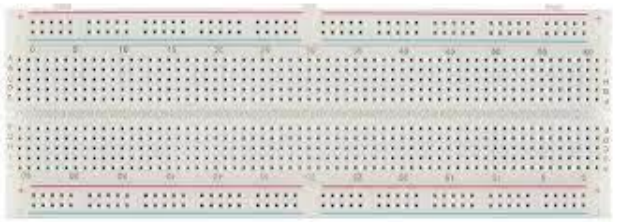
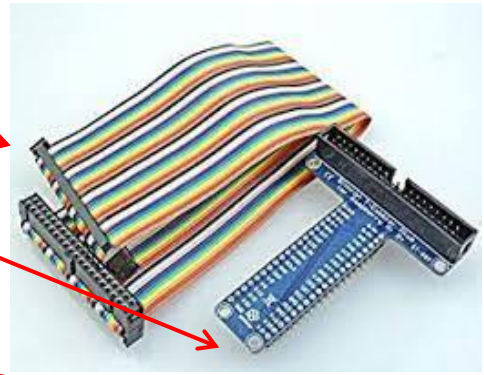
PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Module 2: Prototyping the IoT “Smart” lamp device

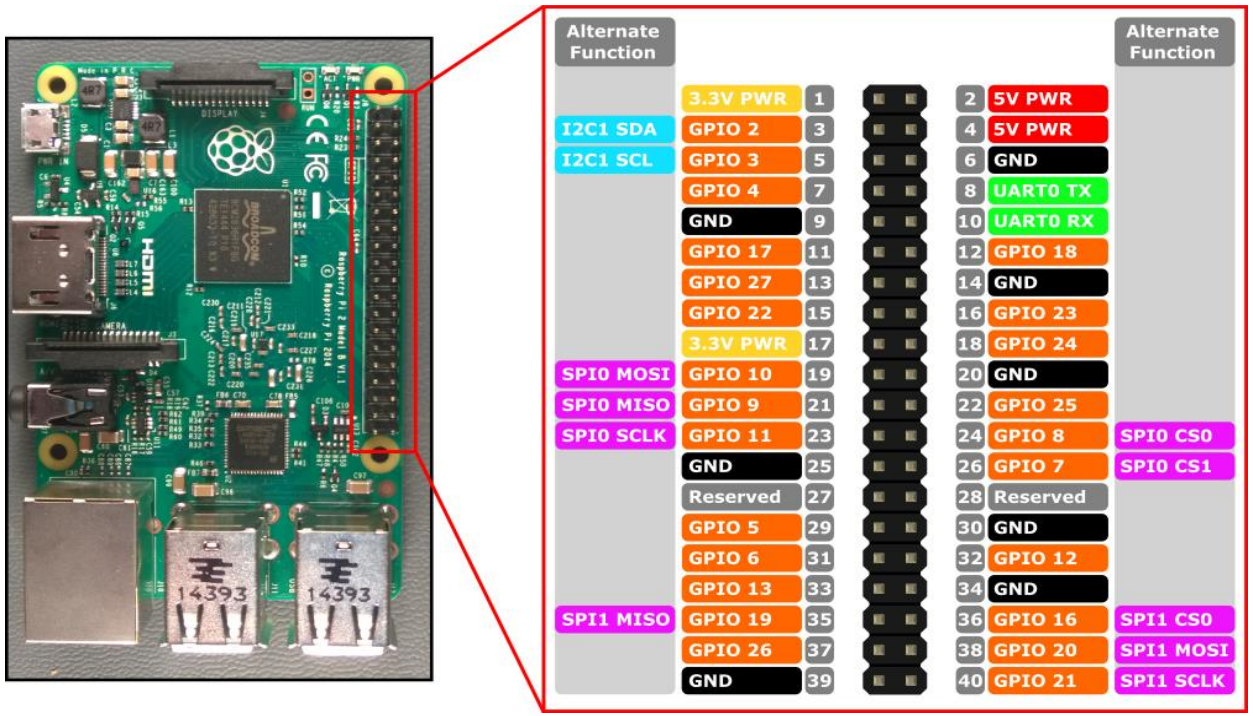
- The “smart” lamp hardware is implemented as a simple electronic circuit, consisting of the following components:

  - List of components:
    - 1 x Raspberry Pi 5
    - 1 x full size bread board
    - 1 x 40-pin ribbon cable
    - 1 x T-cobbler (GPIO extender)
    - 2 x 220 Ohm resistor
    - 2 X 5mm LED
    - 1 X 5v 1 channel relay
    - Jumper wires
    - Switch button




40-pin GPIO (General Purpose Input/Output) Header Overview



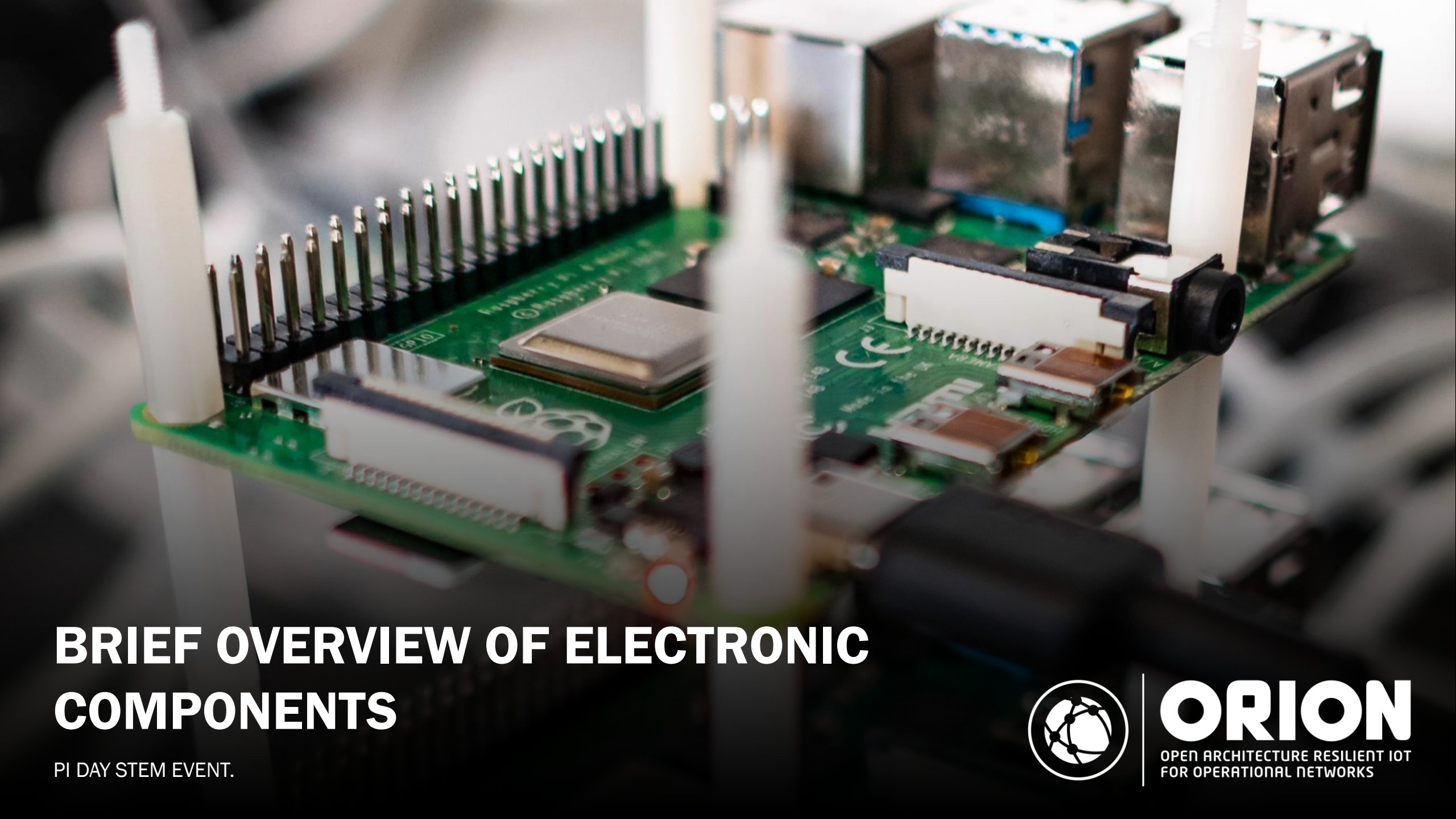
- Allows the code running on Raspberry PI to interface with the outside world…
  - Supports digital communication with external devices (the outside world) using serial bus protocols including: UART, SPI, i2c
    - Does NOT support analog inputs
      - We can easily fix that using an analog-to-digital converter (ADC)

- 40-pin GPIO header
- Pin numbering: Board, BCM/GPIO/WiringPi

https://learn.sparkfun.com/tutorials/introduction-to-the-raspberry-pi-gpio-and-physical-computing/gpio-pins-overview

https://www.raspberrypi-spy.co.uk/2012/06/simple-guide-to-the-rpi-gpio-header-and-pins/

https://projects.raspberrypi.org/en/projects/physical-computing/1

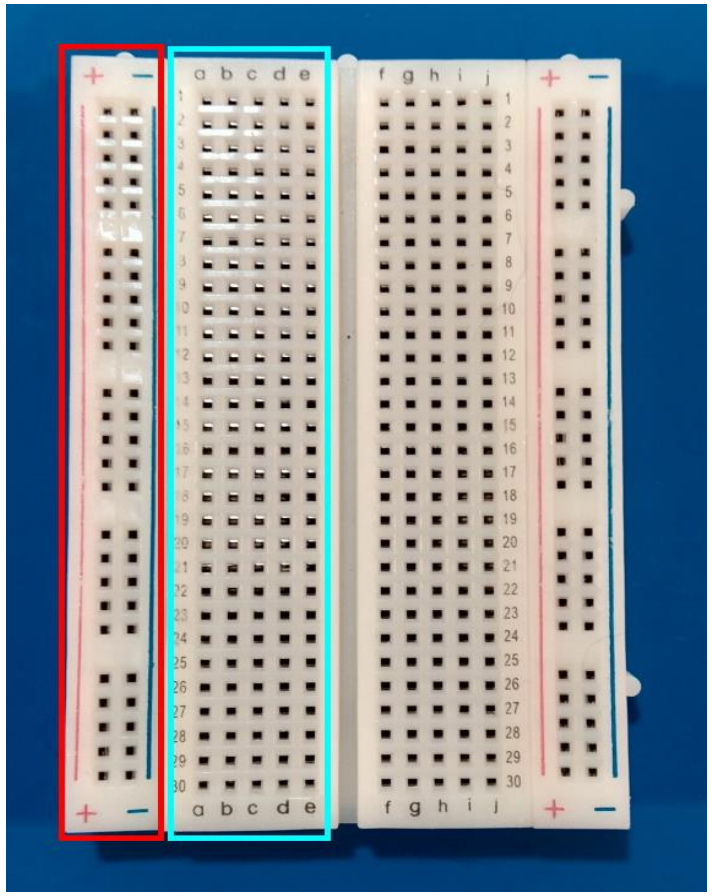


BRIEF OVERVIEW OF ELECTRONIC COMPONENTS

PI DAY STEM EVENT.

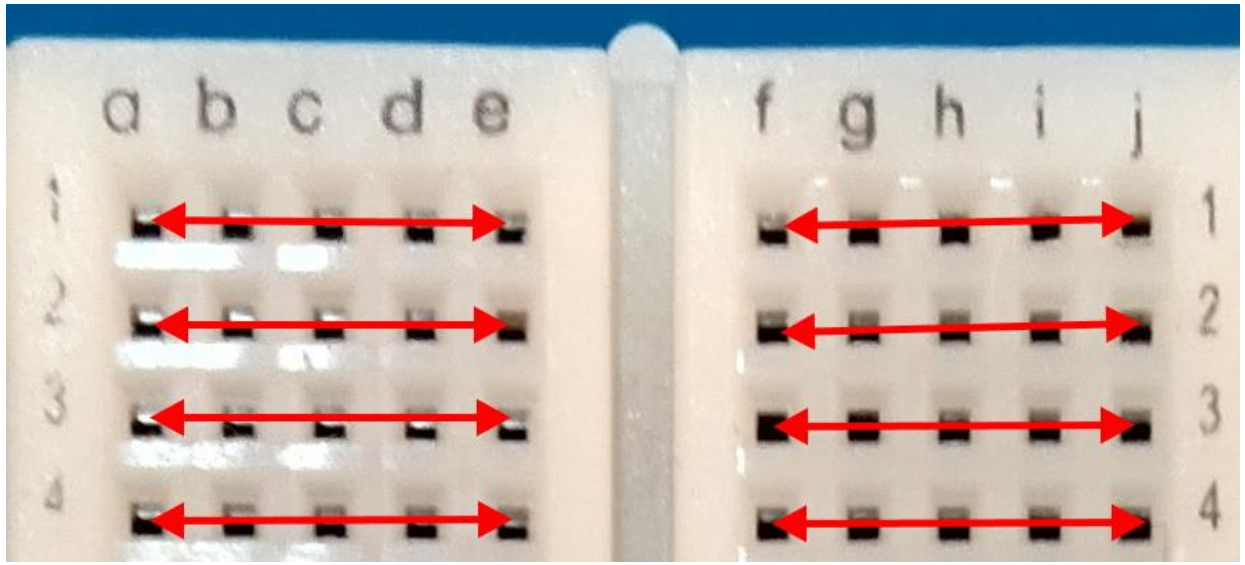
ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Breadboard



Source: https://learn.microsoft.com/en-us/training/modules/create-iot-device-dotnet/2-construct-iot-hardware - [2]

- The breadboard is organized in rows and columns called socket strips (shown in cyan)
  - *Socket strips are connected vertically*
- The bus strips on the edges (shown in red) are using connecting a power source (3.3v and 5v for the Pi)
  - *Edge strips connect horizontally over the length of the breadboard*
- Socket strips allow components to be connected without soldering or wires. [2]



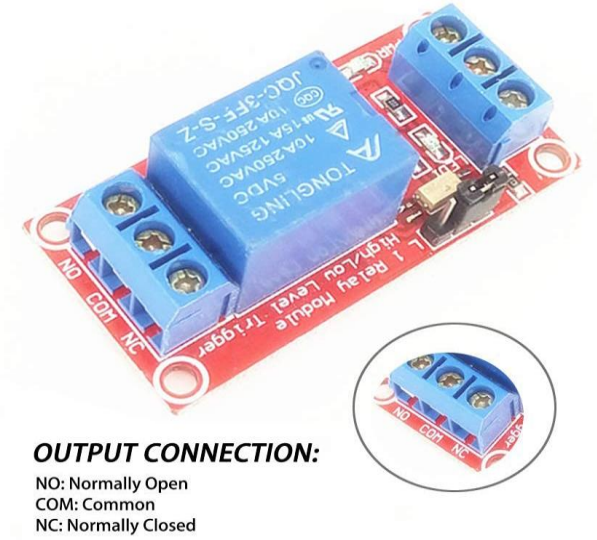
- Any pin plugged into row 1, column ‘a’ would be connected to any pin plugged into row 1, columns ‘b-e’.
- On the other side of the divider, row 1 columns f-j are similarly connected. [2]

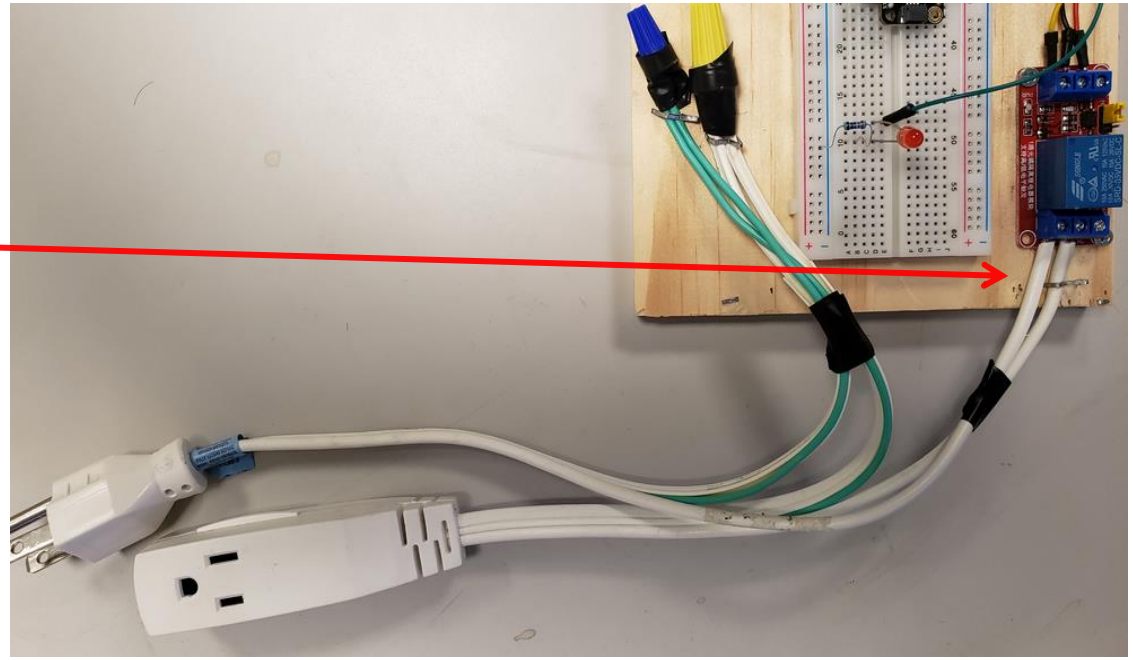
5v Relay Board Relay Module 1 Channel

- An electromagnetic switch used to programmatically activate/deactivate external devices (i.e., a desk lamp or fan).

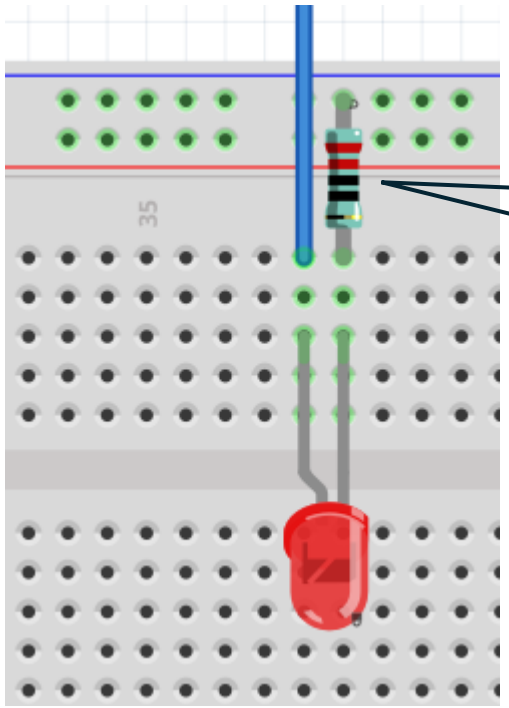
- Use a small current to control a separate (often larger) current.
  - When a small current is passed through the low-voltage (3.3v DC) input on the relay it activates the switch (using a magnetic coil) and completes the separate high voltage (120v AC) circuit (i.e.,the larger current).

    - This is one way how we can control external devices: i.e., the desk lamp





Resistor



Use of 220 Ohm resistor to limit the current through the LED and to prevent excess current that can burn out the LED

A resistor is **a passive two-terminal electrical component that implements electrical resistance as a circuit element**. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages – [3]

[3] - https://en.wikipedia.org/wiki/Resistor

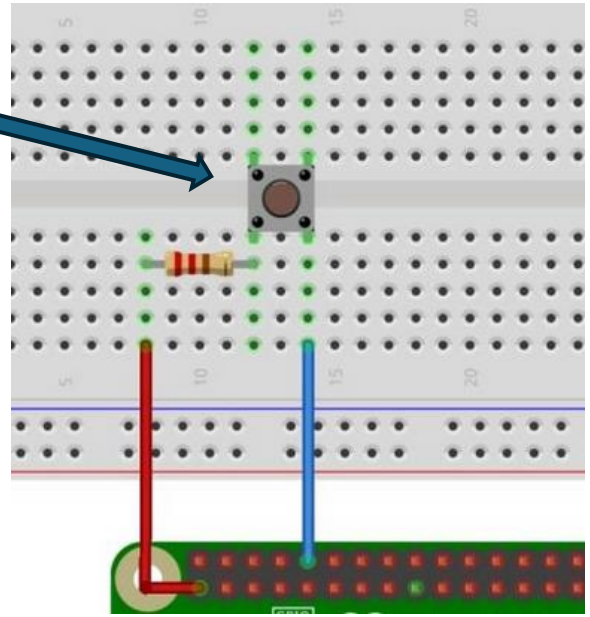
LED and Button Switch

- The longer leg (known as the '**anode**'), is always connected to the positive supply of the circuit. The shorter leg (known as the '**cathode**') is connected to the negative side of the power supply, known as 'ground'.

- **LED** stands for **L**ight **E**mitting **D**iode, and glows when electricity (current) is passed through it. - [2]

- Connect one side of the switch to an input pin on the Raspberry Pi, in this case we use pin 10.

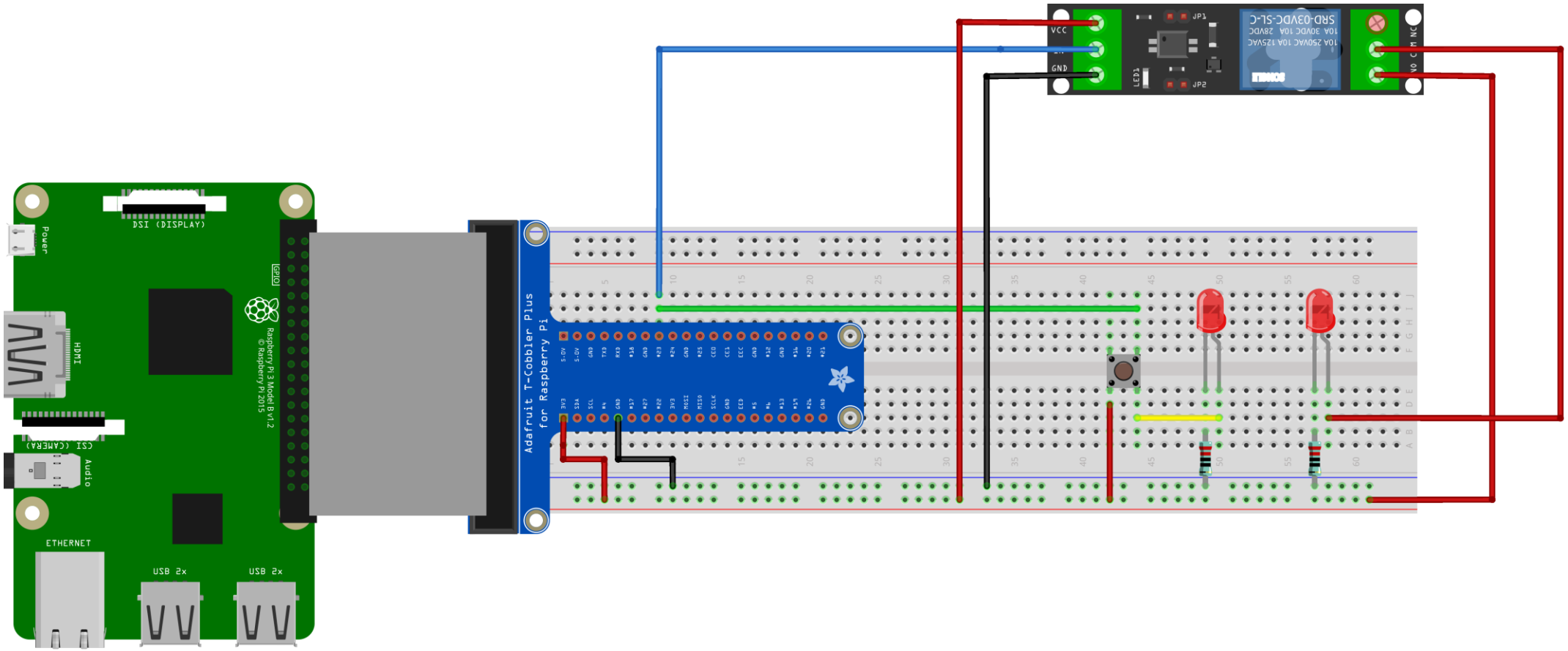
- The other side of the switch we connect to 3.3V on pin 1 using a resistor.
  - The resistor is used as a current limiting resistor to protect our input pin by limiting the amount of current that can flow. [3]



Source: https://thepihut.com/blogs/raspberry-pi-tutorials/27968772-turning-on-an-led-with-your-raspberry-pis-gpio-pins - [2]

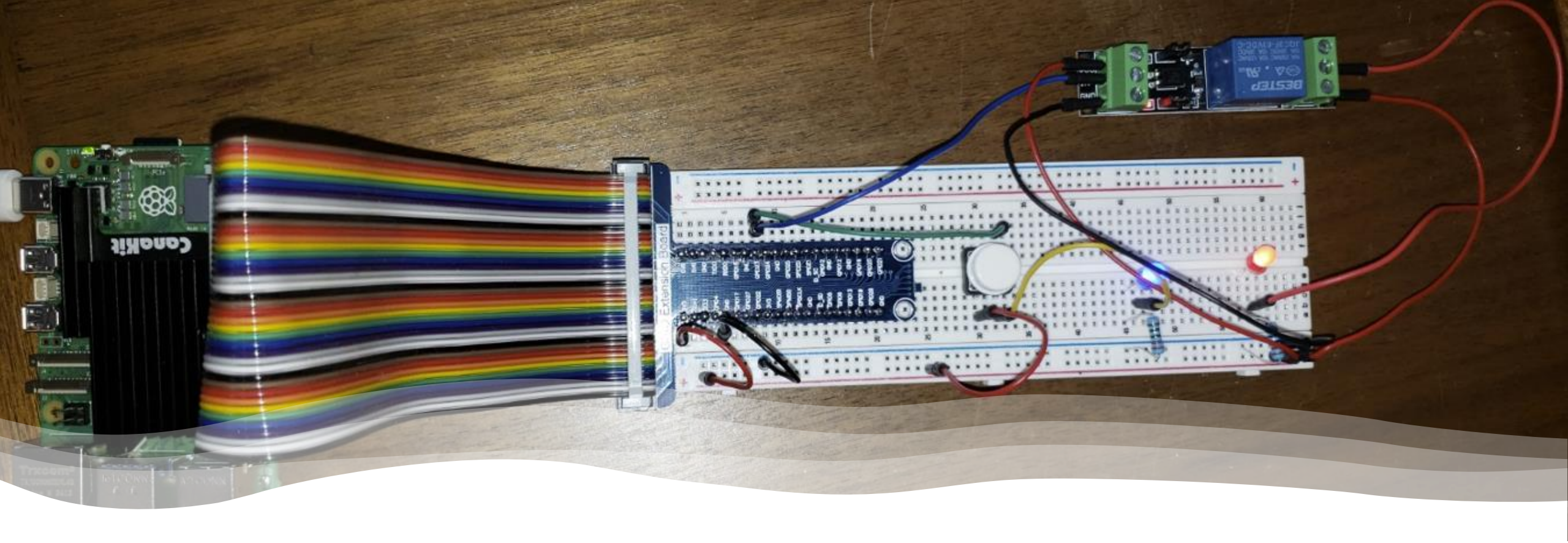
Source: https://raspberrypihq.com/use-a-push-button-with-raspberry-pi-gpio/ - [3]

“Smart” lamp device (Fritzing circuit design) concept



Fritzing is an [open-source hardware initiative](https://fritzing.org/) that makes electronics accessible as a creative material for anyone.

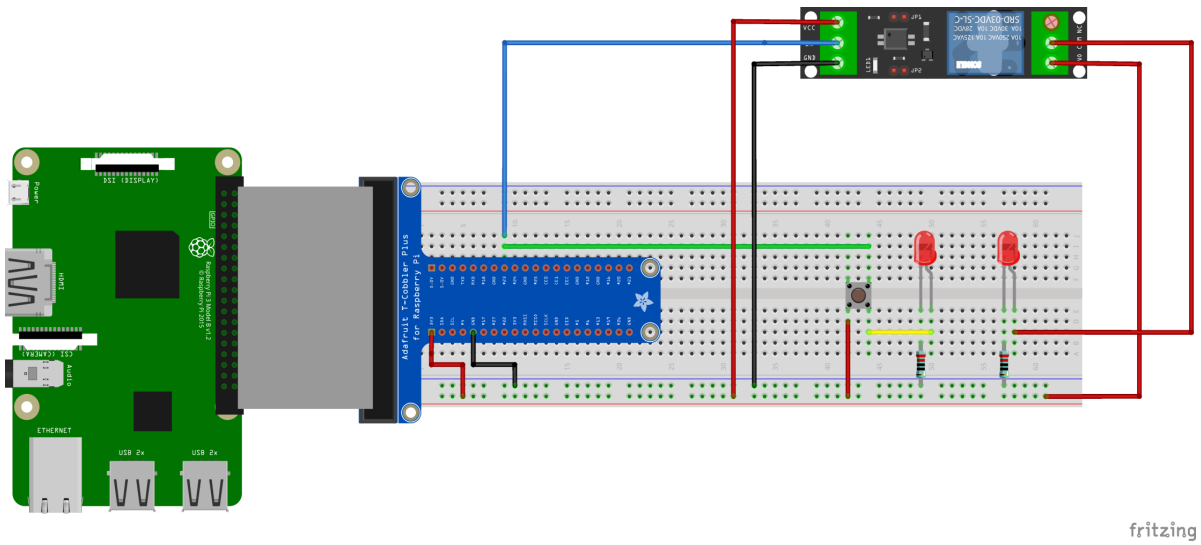
https://fritzing.org/

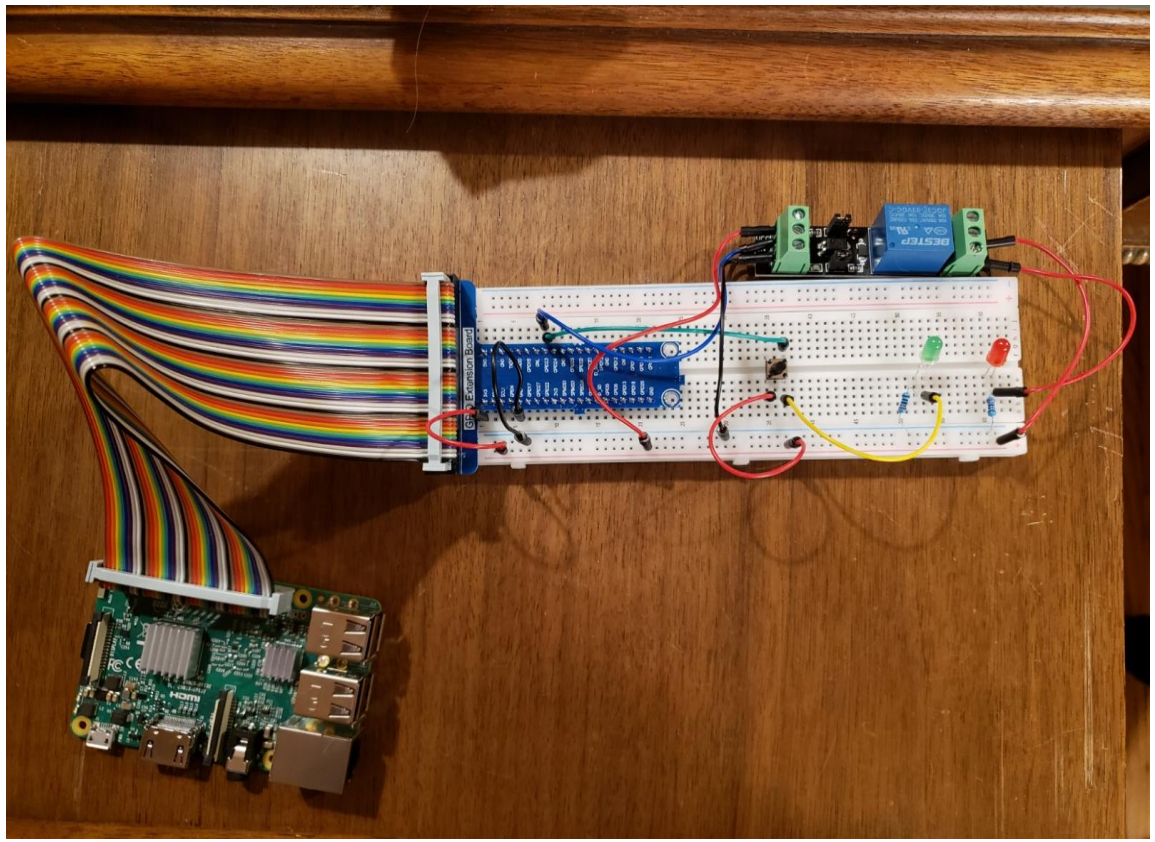


Completed “Smart” la device (completed prototype)

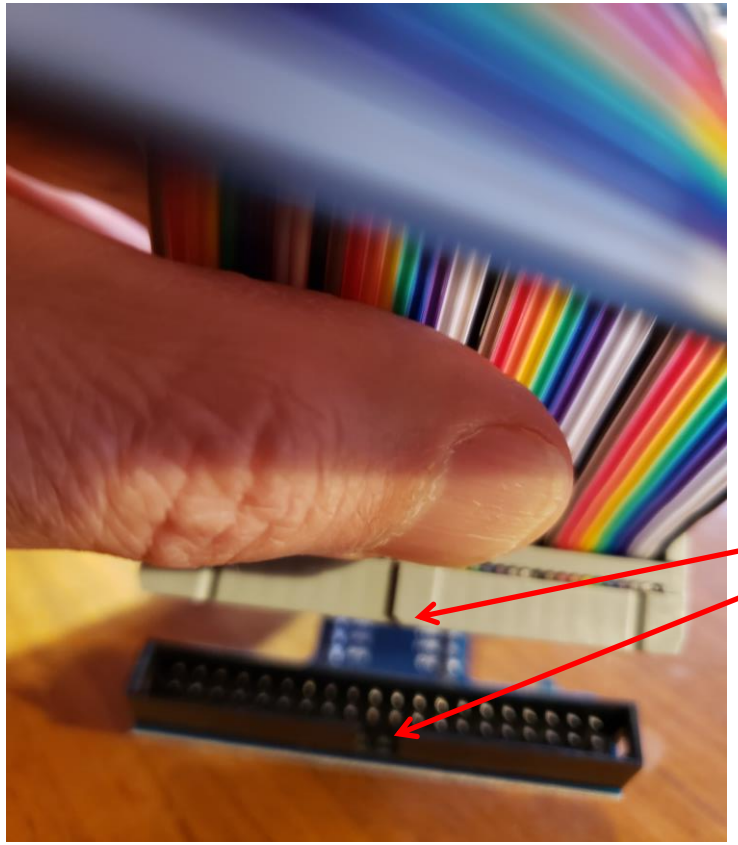
“Smart” lamp device (concept versus prototype)

versus





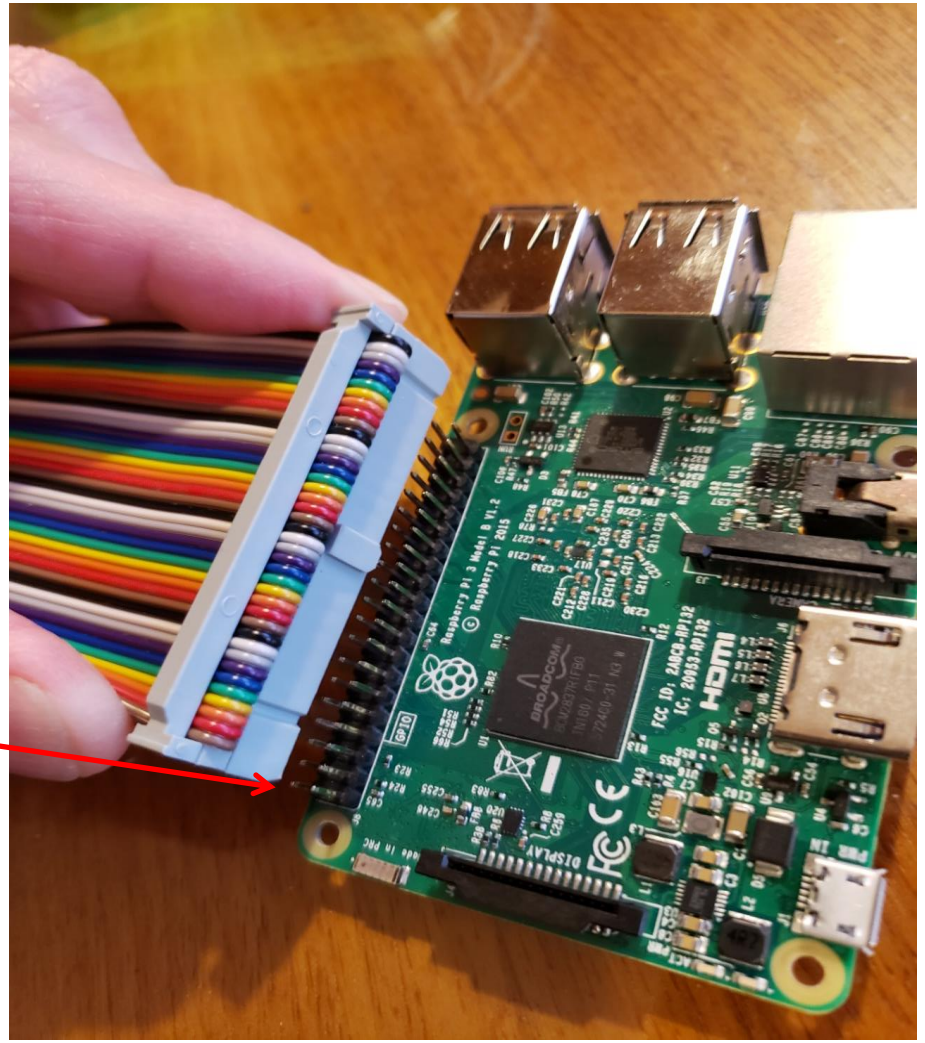
Properly Attaching the Ribbon Cable between the Raspberry Pi and the T-Cobbler



Align the notch on the ribbon cable with the t-cobbler

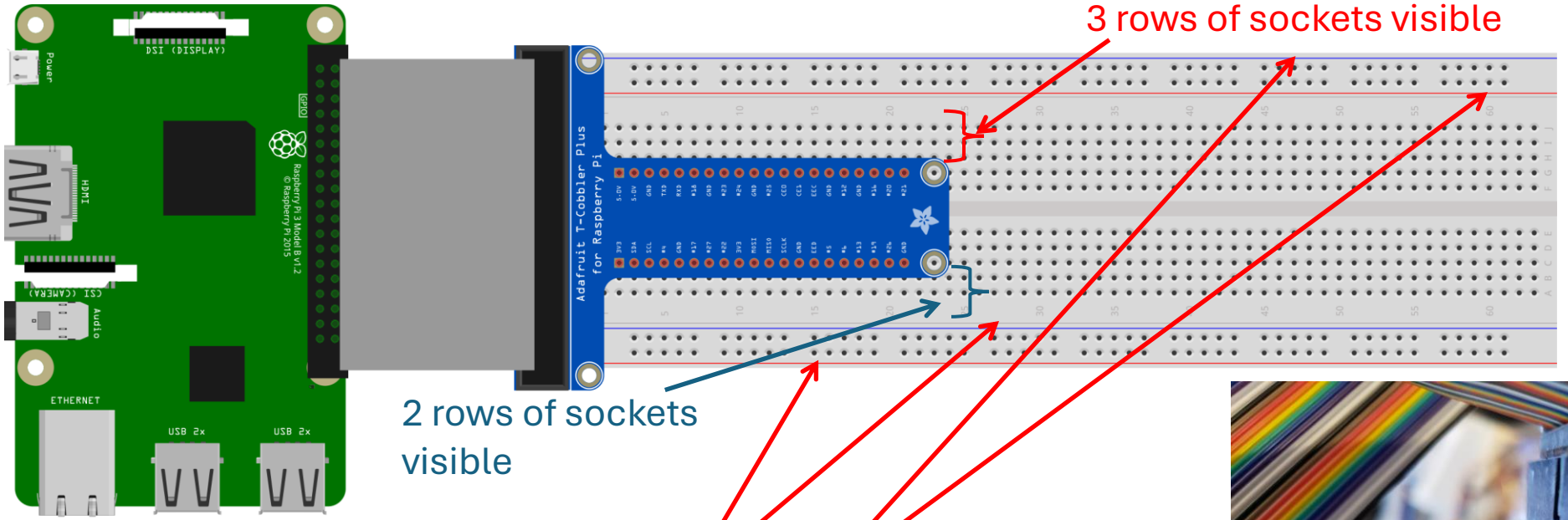
Connect the ribbon cable so it does not cross over the Raspberry Pi (as illustrated)

* Carefully align the ribbon cable to connect all 40 pins of the GPIO header.



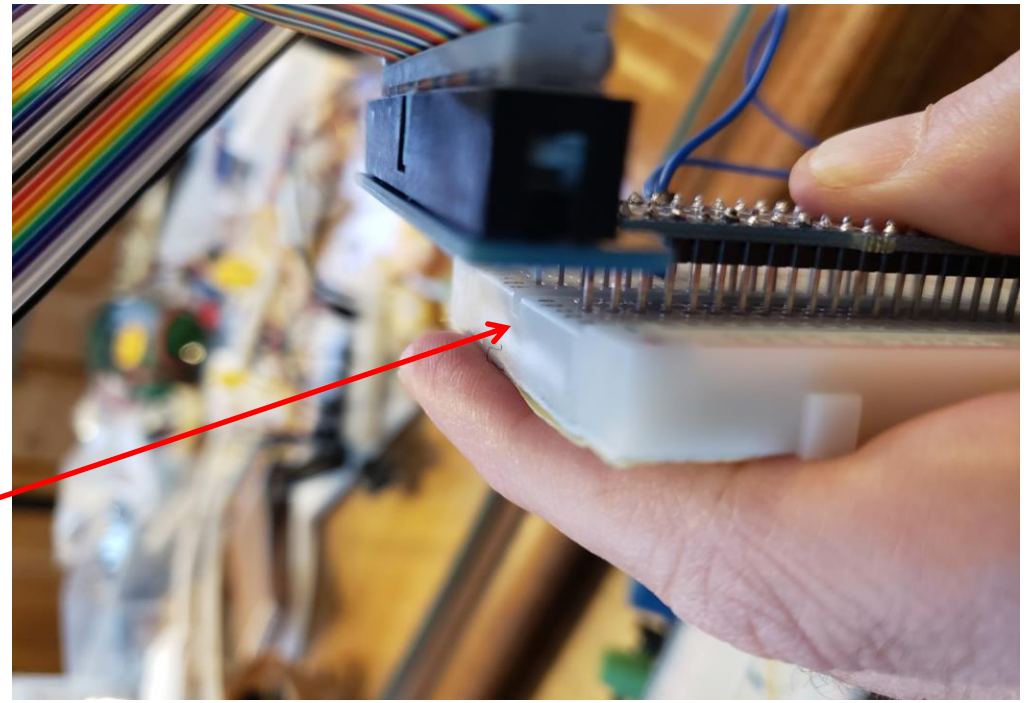
Attach the T-Cobbler to the breadboard

*** Note of the rows on exposed on each side of the t-cobbler (3 on one side, and 2 on the other. Your design should look like this too.**



**** Note the orientation to negative/positive (power) terminal strips**

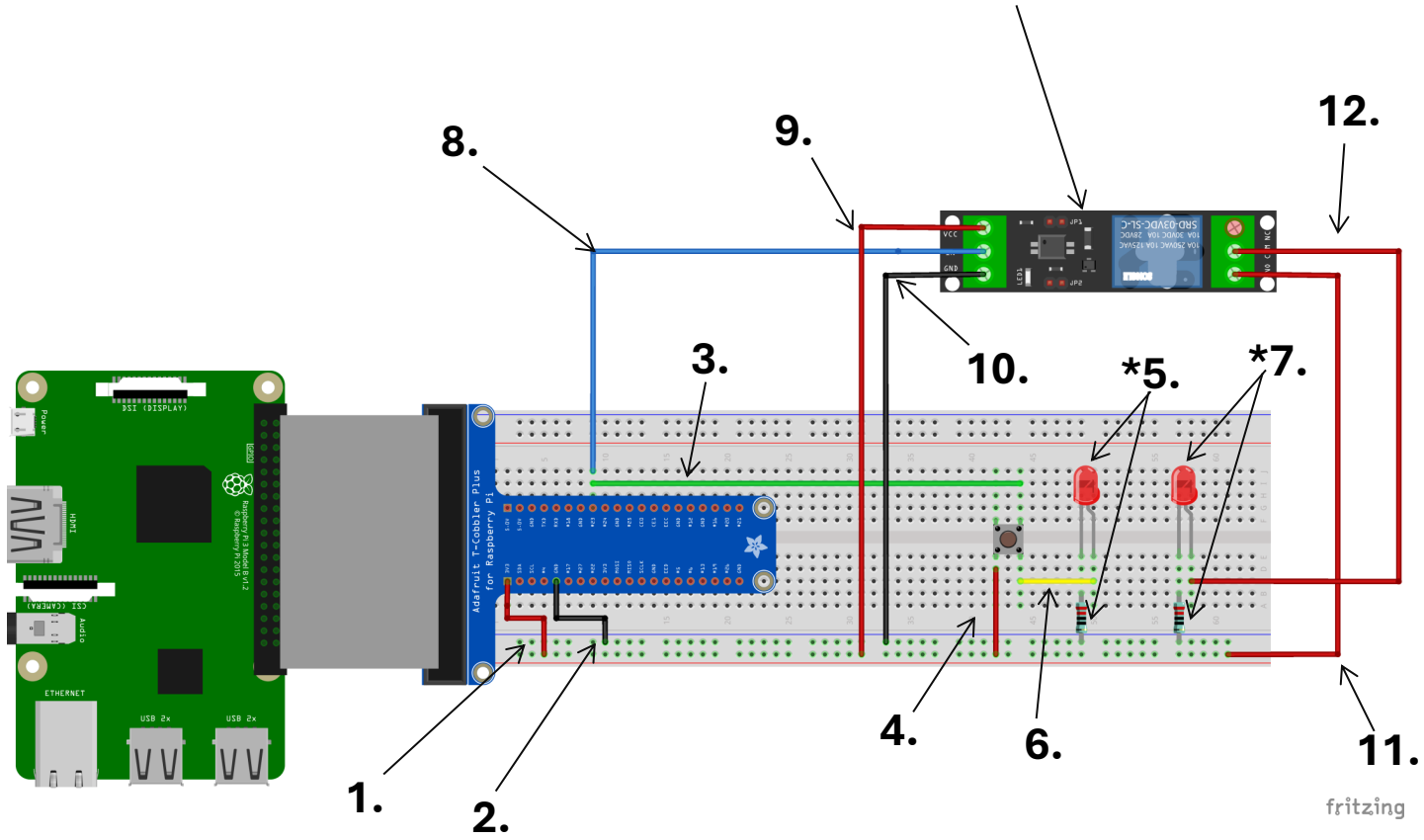
***** Insert the t-cobbler beginning in the first column of sockets in the breadboard**



Construct the device (electronic circuit):

Power off and disconnect the Raspberry Pi power source

1. Connect a **"Red"** jumper wire to the 3.3v pin of the T-Cobber and the positive (+) terminal strip of the breadboard (same side as the 3v pin). This provides 3.3v power to entire edge terminal socket strip designated by the red line
2. Connect a **"Black"** jumper wire to a "GND" pin on 3.3v side of the T-Cobbler, and the negative (-) terminal strip on the breadboard (same side 3v pin). **This provides a ground (-) to the entire edge terminal socket strip designated by the blue line**
3. Push a **switch button** onto the bread board (as shown). Connect an **"Green"** wire to GPIO 23 from the T-cobbler to the output side of the button
4. Connect **"Red"** wire from the positive (+) 3.3v terminal strip of the breadboard to the input side of the switch button
5. Push an LED into the breadboard (as shown). *Note: the bend in the LED terminal indicates the longer (+) LED anode.* Connect a **220-ohm resistor** between LED cathode (shorter negative -) pin and the negative (-) terminal strip on the breadboard
6. Connect a "Yellow" wire from the output side of the switch button to *the longer (+) LED anode* terminal of the LED
7. Push an LED into the breadboard (as shown). *Note: the bend in the LED terminal indicates the longer (+) LED anode.* Connect a **220-ohm resistor** between LED cathode (shorter negative -) pin and the negative (-) terminal strip on the breadboard
8. Connect a **"Blue"** jumper wire to the GPIO pin 23 from the T-cobbler, and the *IN* terminal of the 3.3v 1 channel relay
9. Connect a **"Red"** jumper wire from 3.3v positive (+) terminal strip of the breadboard to *VCC* terminal on of the relay
10. Connect a **"Black"** jumper wire from negative (+) terminal strip of the breadboard to *GND* terminal on of the relay
11. Connect a **"Red"** wire to the NO (Normally open) terminal of the relay to the 3.3v (positive +) terminal strip of the breadboard
12. Connect **"Red"** wire the COM (Common) terminal of the relay to the (positive +) terminal of the LED



* - indicates multiple steps

Connect and power on Raspberry Pi and test by press the button to activate the bulb

Module Questions?

1. Does your circuit operate? I.e., do both LEDs illuminate?
   - Can you summarize how the circuit is operating?
     - Temporarily disconnect the green wire connection to button
       - Explain what happened and why?

MODULE 3: IMPLEMENT THE “SMART” LAMP CONTROLLER SOFTWARE

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Module 3: Implement the “Smart” Lamp Controller Software

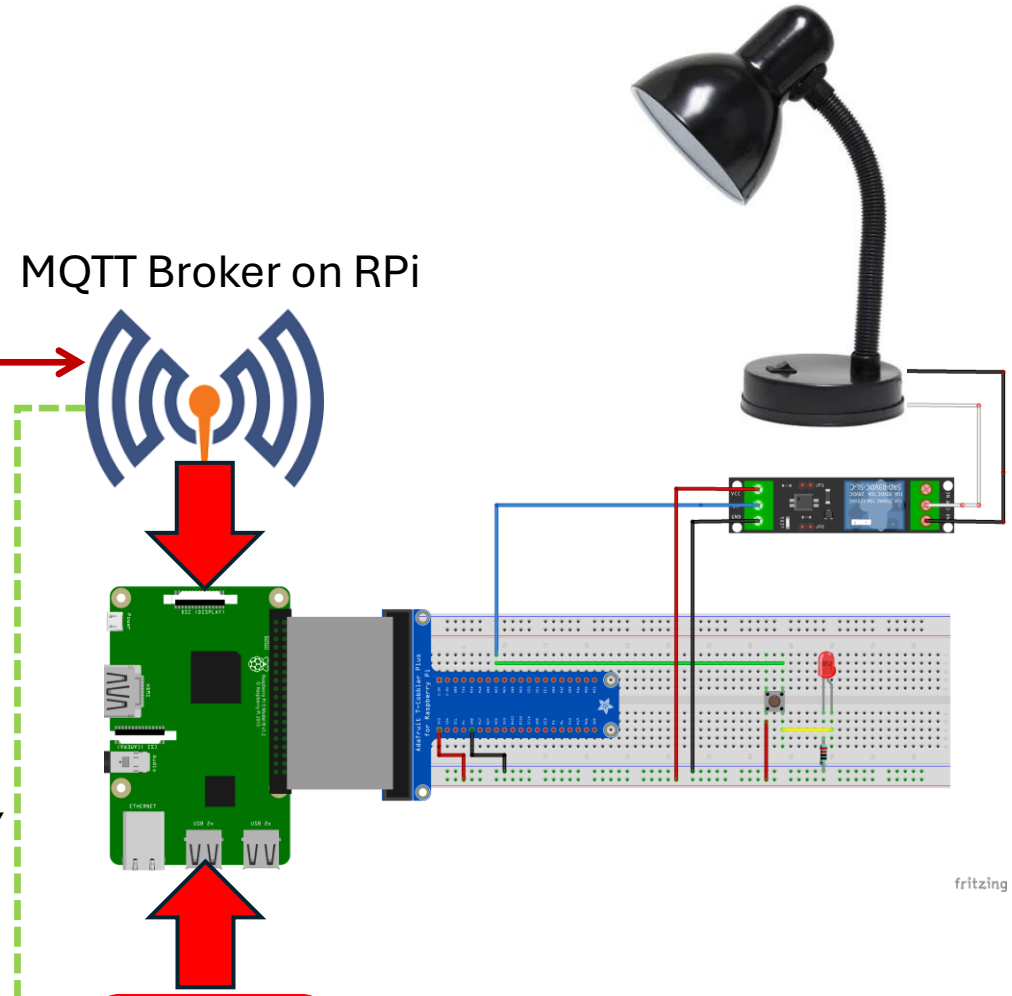
1. Implement the back-end to control the lamp
2. Implement the front-end user interface

MQTT / Web sockets

Publish in topic: *Bulb/*

Message: *“on”/”off”*

MQTT Broker on RPi



Subscribe to topic: *Bulb/*

Bulb/”on”

Bulb/”off”

Python Backend

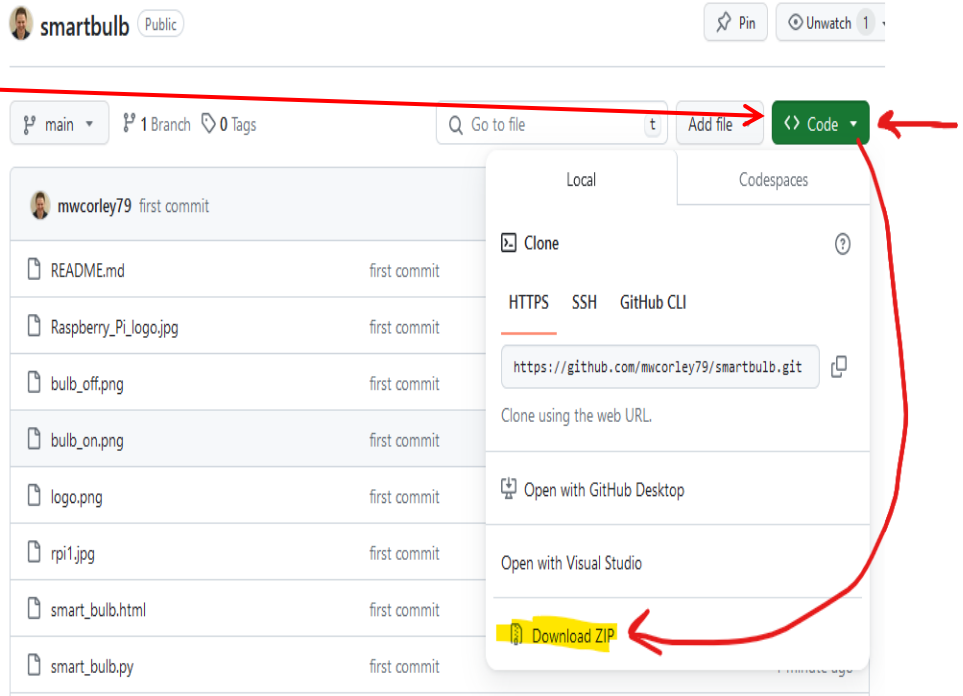
1. Smart Bulb Web app: (based on HTML 5 CSS and JavaScript)

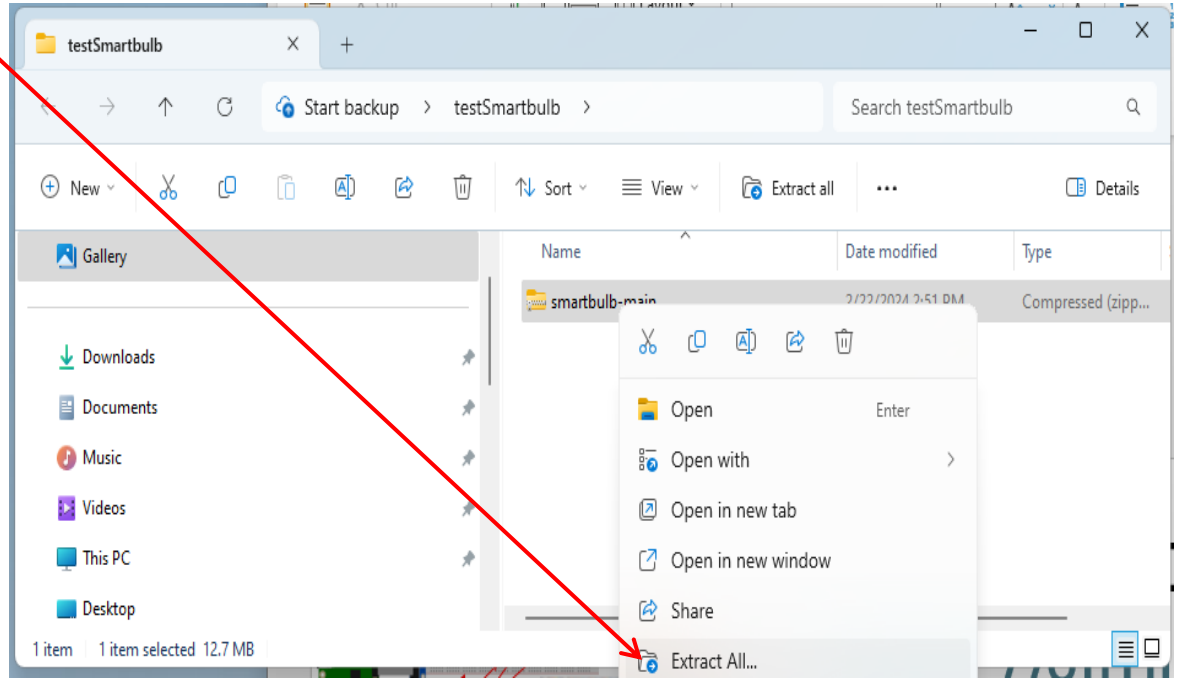
2. Python smart bulb logic running on RPi

Download the smart bulb source code from GitHub:

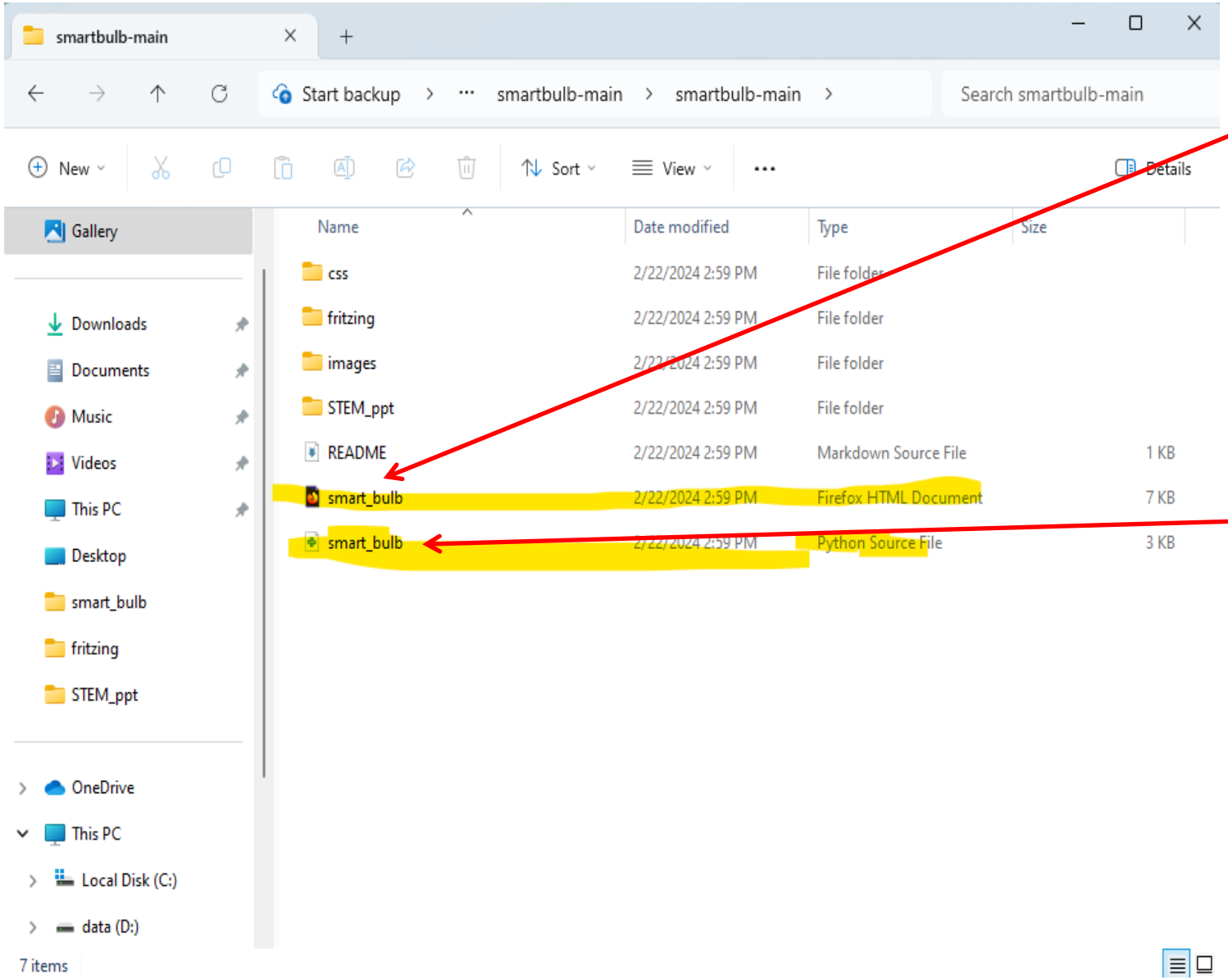
https://github.com/mwcorley79/smartbulb

- Choose "Code"→"Download Zip", and save the zip to the "Desktop" folder on the laptop
- Navigate to Desktop, and right-click the zip file, and choose extract all





Project structure: navigate to the extracted folder to view the project files

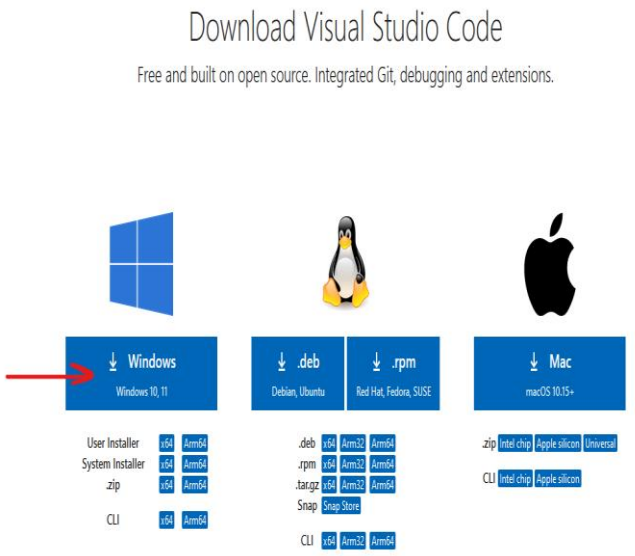


1. *smart_bulb.html* – This is front-end html user (web) interface app

2. *smart_bulb.py* – This is the back-end Python script that runs on the Raspberry Pi
   - interfaces with the GPIO header to control the desk lamp

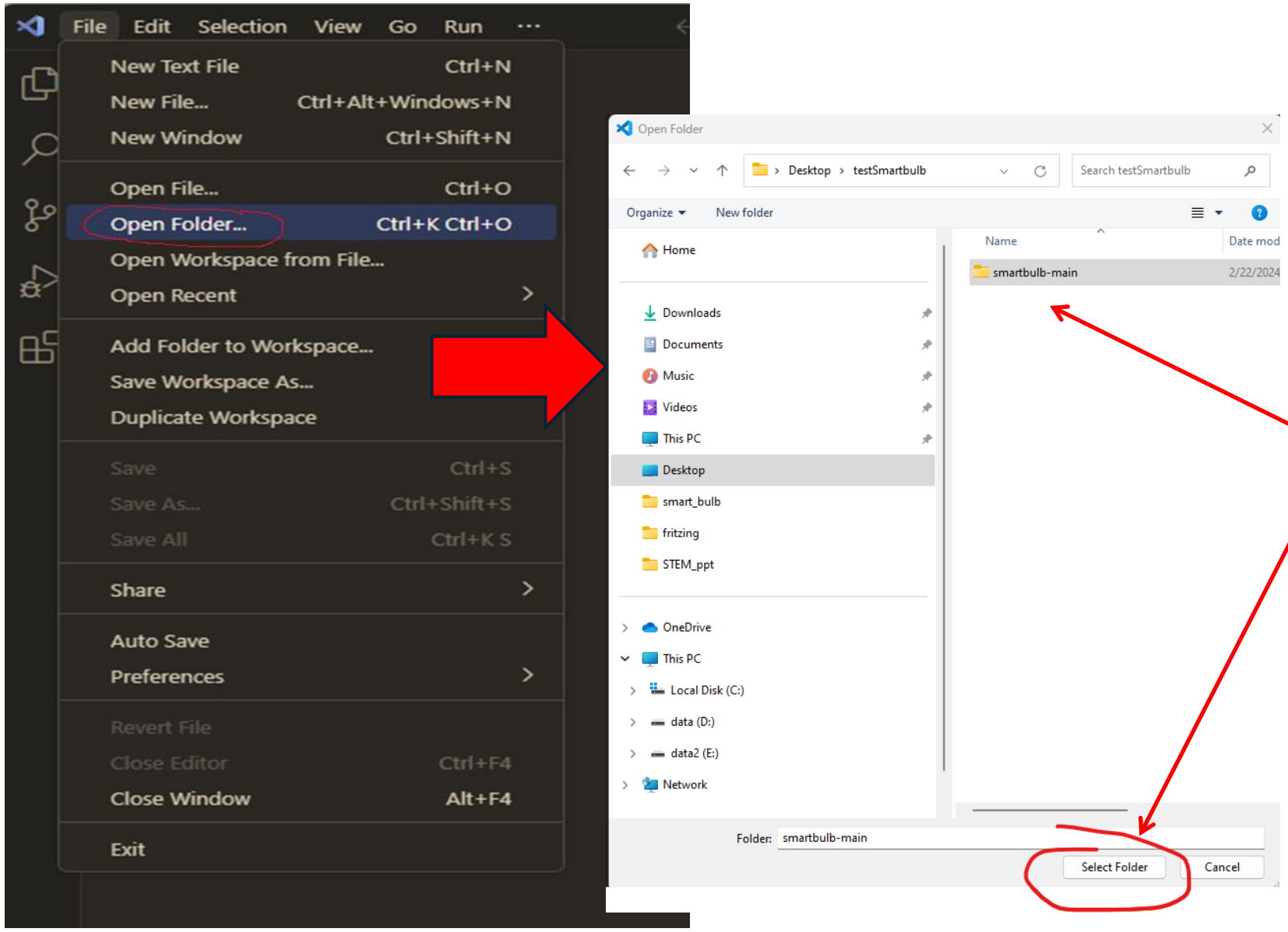
Install Visual Studio Code (VSCode) Editor

- VSCode is the most popular software development code editor available today.
  - Typically, the software for the Raspberry Pi is developed using a development host (your windows laptop), equipped with development tools including: VSCode, Python interpreter, debuggers, and potentially device simulators etc.)
    - Software is typically developed and tested incrementally
      - The resulting software artifacts are moved from the dev host to the Raspberry Pi (sometimes referred to as the *target* device).
        - Debuggers help with development
          - Note: although development can be performed directly on the Raspberry Pi, it is more often performe on the development host



- Download and install Visual studio code for Windows:
  - https://code.visualstudio.com/download

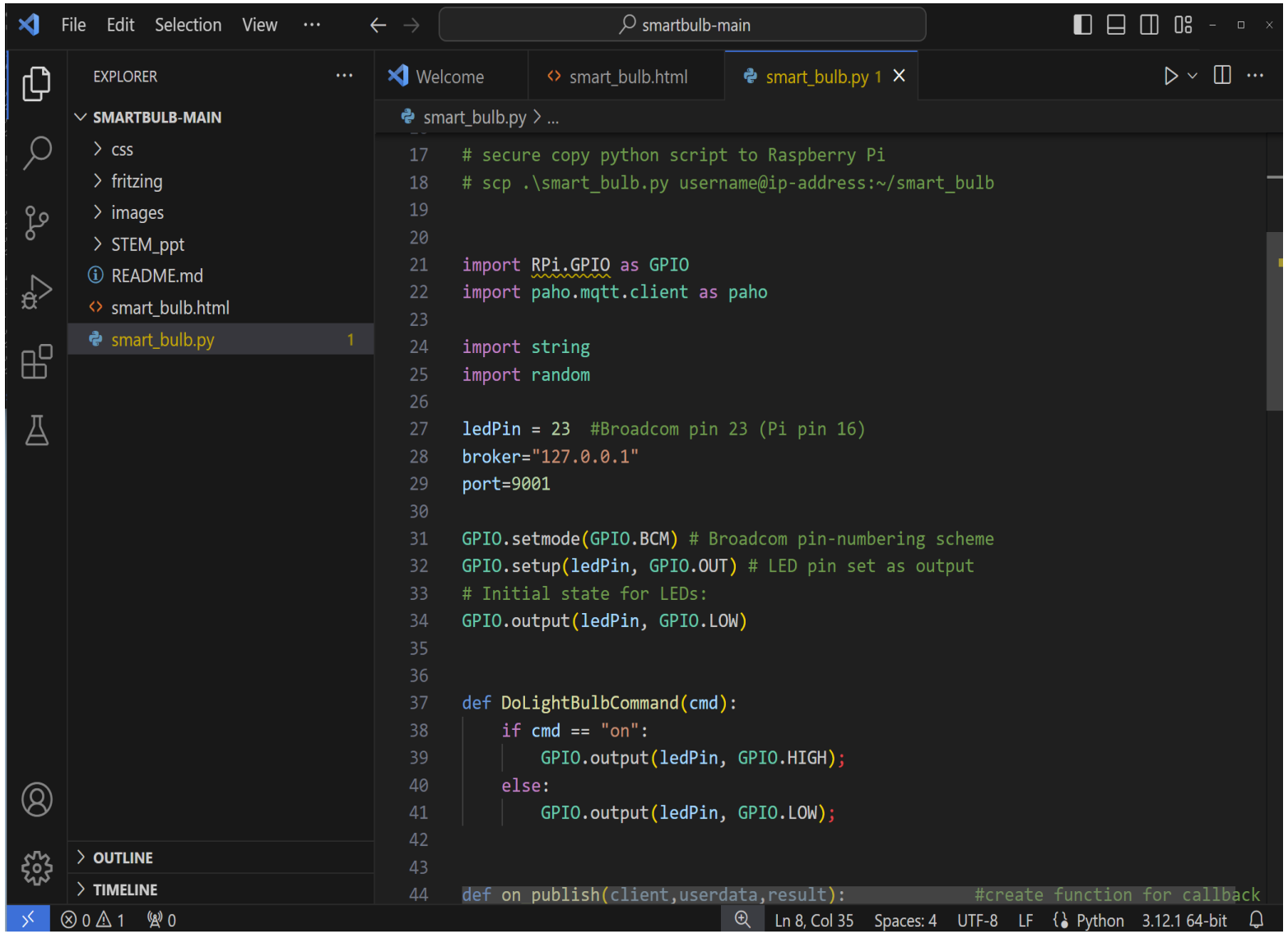
Opening the Project in VSCode



- Navigate to the Desktop
- Double click the VSCode icon to start the editor

- Choose “Open Folder”, select the project folder, and click “Select Folder”

The back-end (Python) code in VSCode: ***smart_bulb.py***



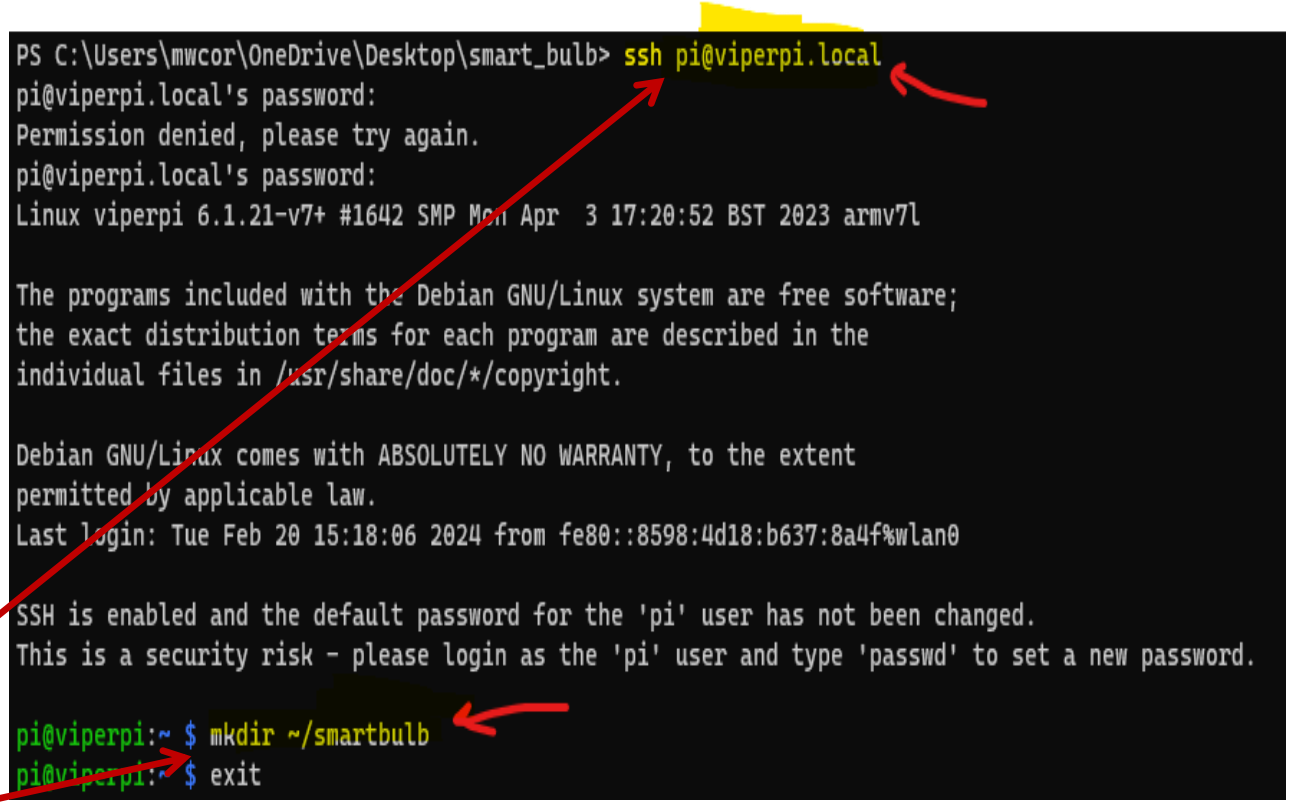
- The **back-end (Python 3) script runs on the Raspberry Pi** to control the desk lamp
  - We **need to copy *smart_bulb.py* to the Raspberry Pi** using secure copy (SCP)

- Controlling the lamp requires logic that uses
  1. *Paho MQTT library* to connect to the broker using MQTT over web sockets
     - subscribes to the “bulb” topic, and receives “on” / “off” messages
  2. Message processing logic uses a *GPIO access library* to interface with of the desk lamp by controlling the relay switch

- Take a few mins to review this code

Create folder on the Raspberry Pi to store ***smart_bulb.py***

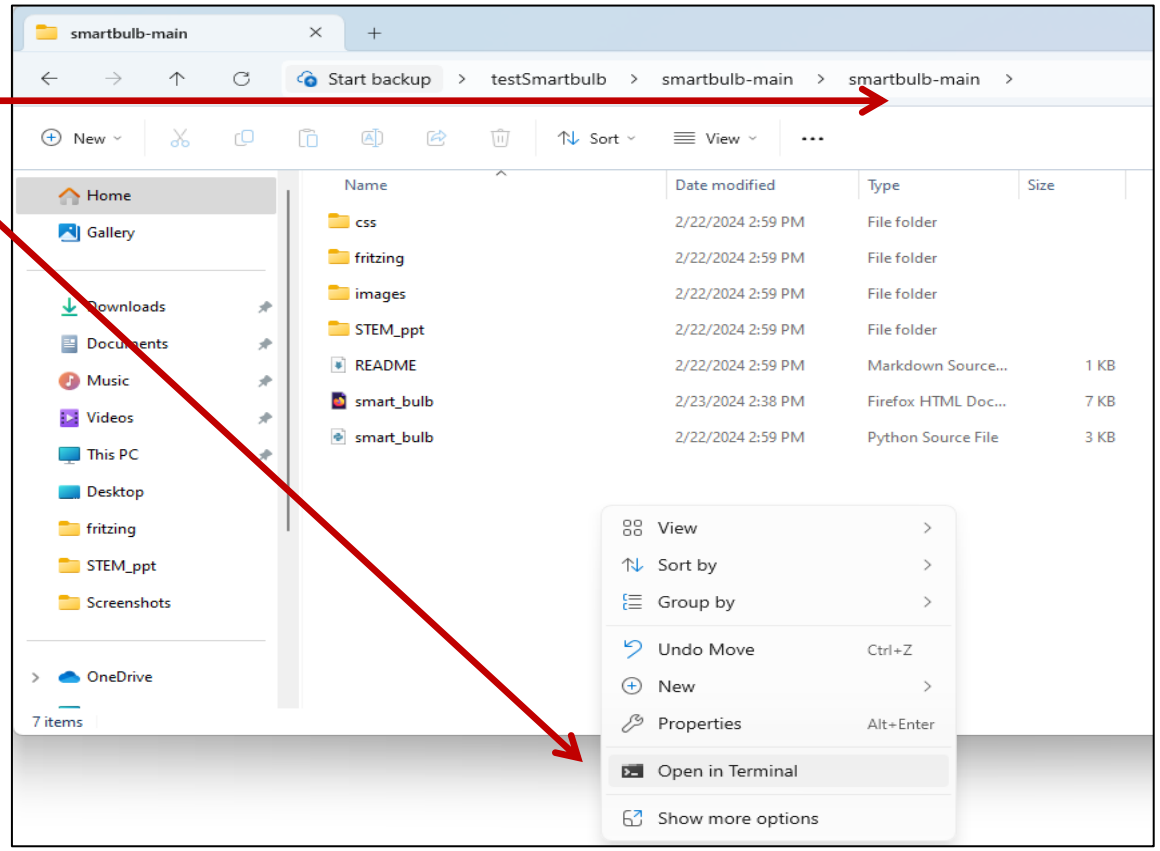
- Using the a VNC session, or open a windows terminal and start an SSH session with Raspberry Pi using the IP address:
  - *ssh pi@10.x.x.x*
  - Create a new destination folder to store the smart_bulb.py script
    - *mkdir ~/smartbulb*
  - You can type *exit* to terminate the SSH session, but leave it active, we will continue to use it in the next steps



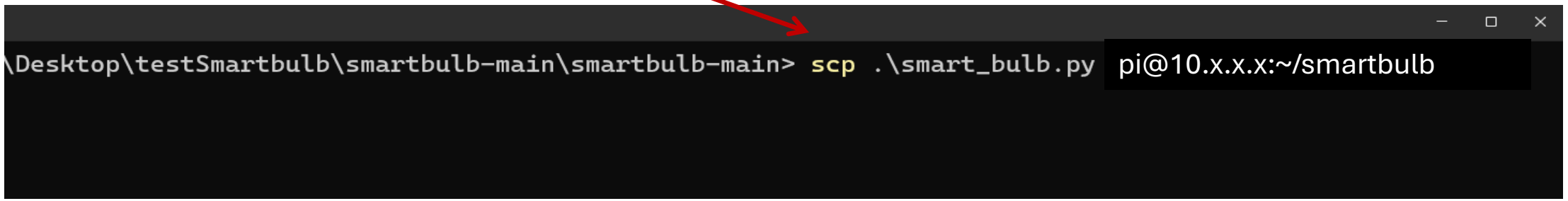
Use Secure (SCP) to copy ***smart_bulb.py*** to the Raspberry Pi

Navigate to the project folder, right-click, and choose “Open in terminal”

- Type the following command:
  - `scp .\smart_bulb.py pi@10.x.x.x:~/smartbulb`
    - This copies *smart-bulb.py* to the Raspberry Pi destination folder created in the previous step: *~/smartbulb*

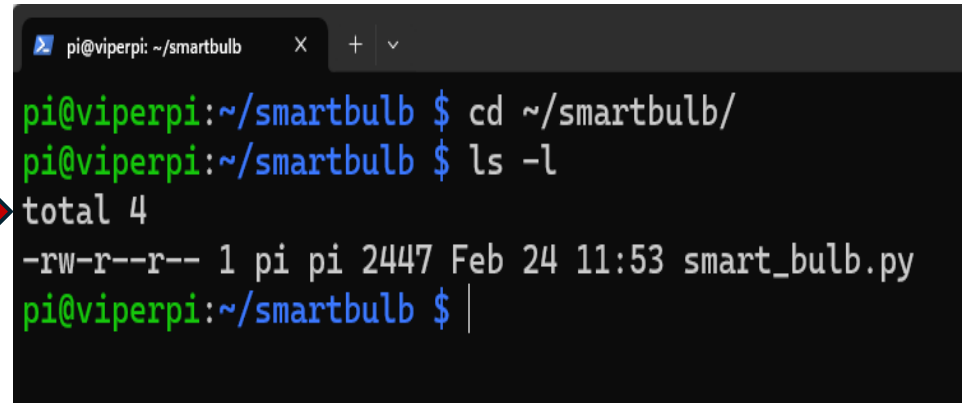


Example:

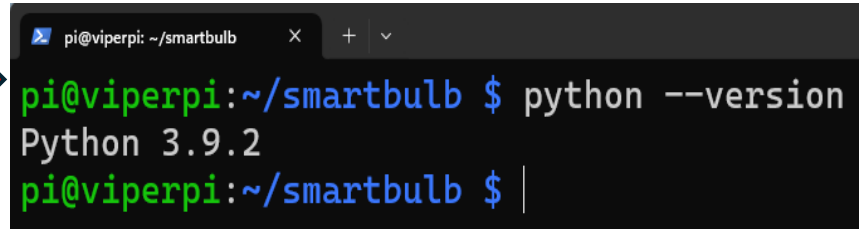


Install the Python dependencies required to run *smart_bulb.py*

- Return to active SSH session with the Raspberry Pi (or start a new session)
  - Navigate to the folder created previously for storing the smart_bulb.py
    - Enter command: *cd ~/smartbulb*
  - Verify the smart_bulb.py was successfully transferred by entering command: *ls –l*



- The IoT smartbulb project was developed to be compatible with Python 3.9 and later:



1. Verify Python 3 is preinstalled on the Raspberry Pi
   - ➢ *python --version*

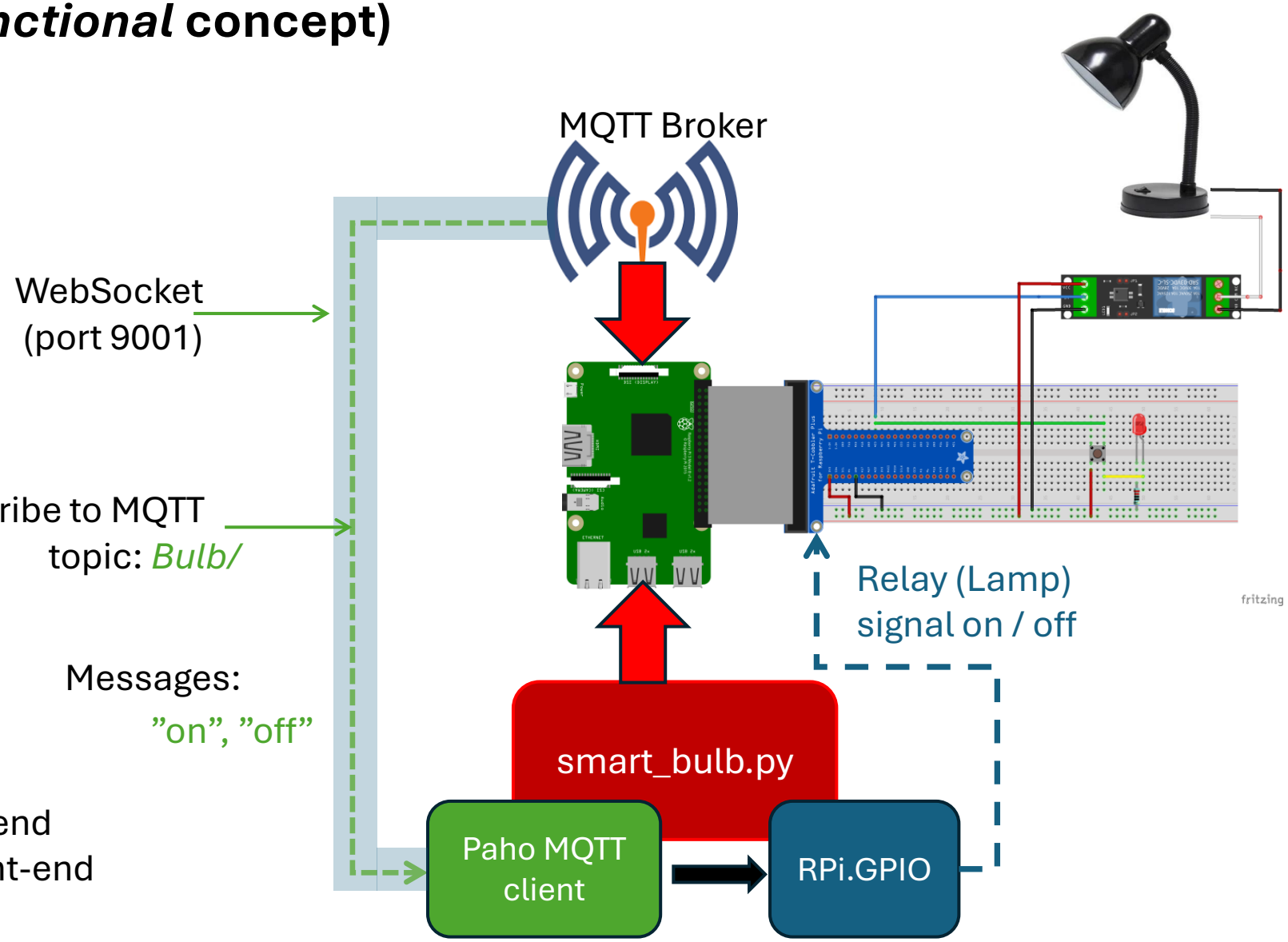
2. Install Paho MQTT Client package library
   - ➢ pip3 install "paho-mqtt<2.0.0“ --break-system-packages
     - Note: Version 2.0.0 of the library was released in February 2024; and contains breaking changes so the above installs v1

3. Install Python module to control the GPIO: RPi.GPIO
   - ➢ pip3 install rpi-lgpio --break-system-packages

Back-end: smart_bulb.py (functional concept)

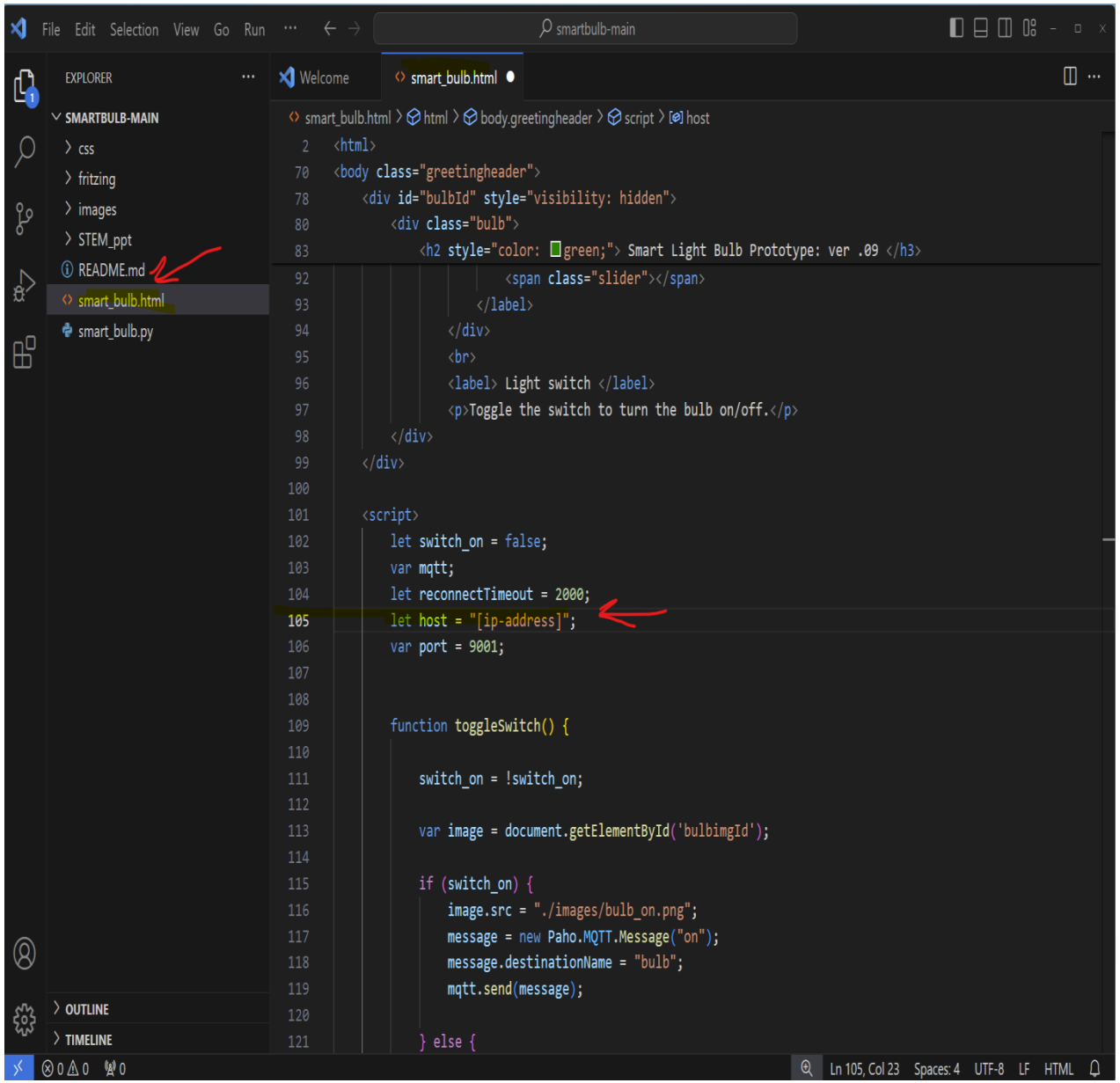
To Run the back-end, enter command:
python ./smart_bulb.py

```
pi@viperpi:~/smartbulb $ python ./smart_bulb.py
client igfru0bo65_client
Python Client connected over web sockets with result: 0
Subscribed to topic: bulb
```

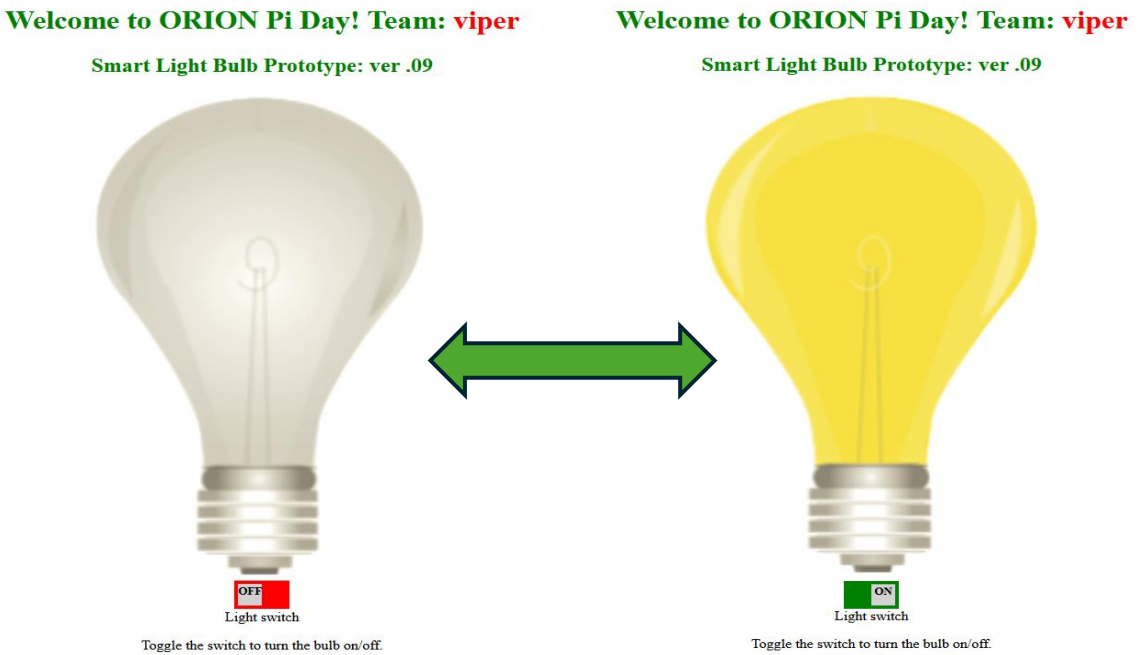


Note: It is important to start the back-end code and leave it running, else the front-end (user interface) will not function.

Front-end user interface (HTML/JS) code in VSCode: ***smartbulb.html***

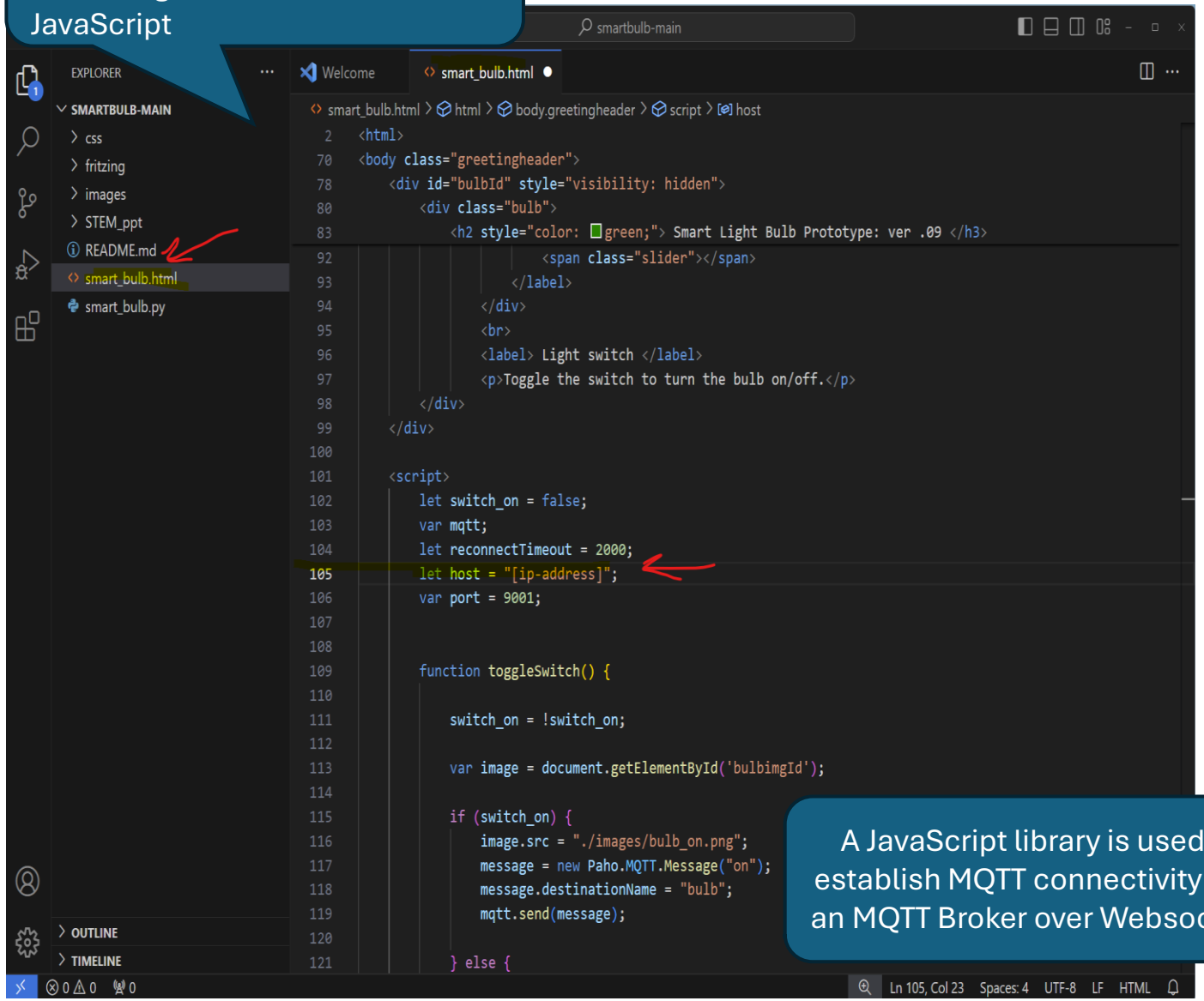


- The web interface allows the user to toggle a slider switch button that simulates turning a lamp on and off .
  - Uses MQTT over WebSockets to control the operation of a desk lamp



Front-end user interface (HTML/JS) code:
smartbulb.html

Smartbulb.html is developed using a combination web technologies: HTML 5, CSS , and JavaScript



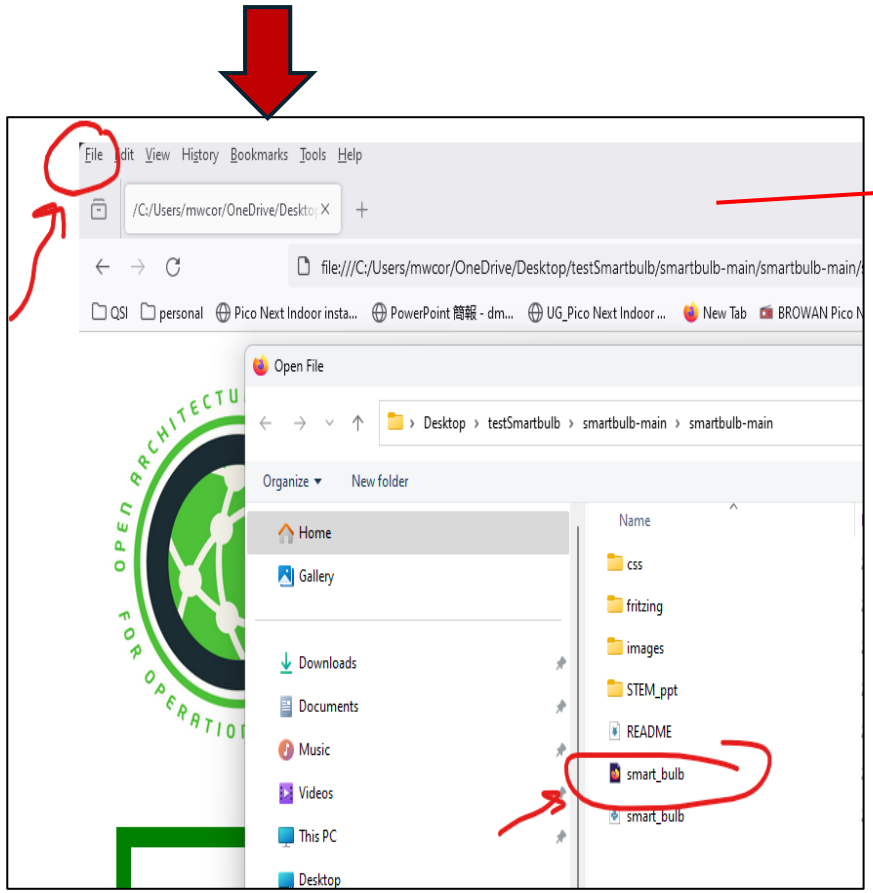
A JavaScript library is used to establish MQTT connectivity with an MQTT Broker over Websockets

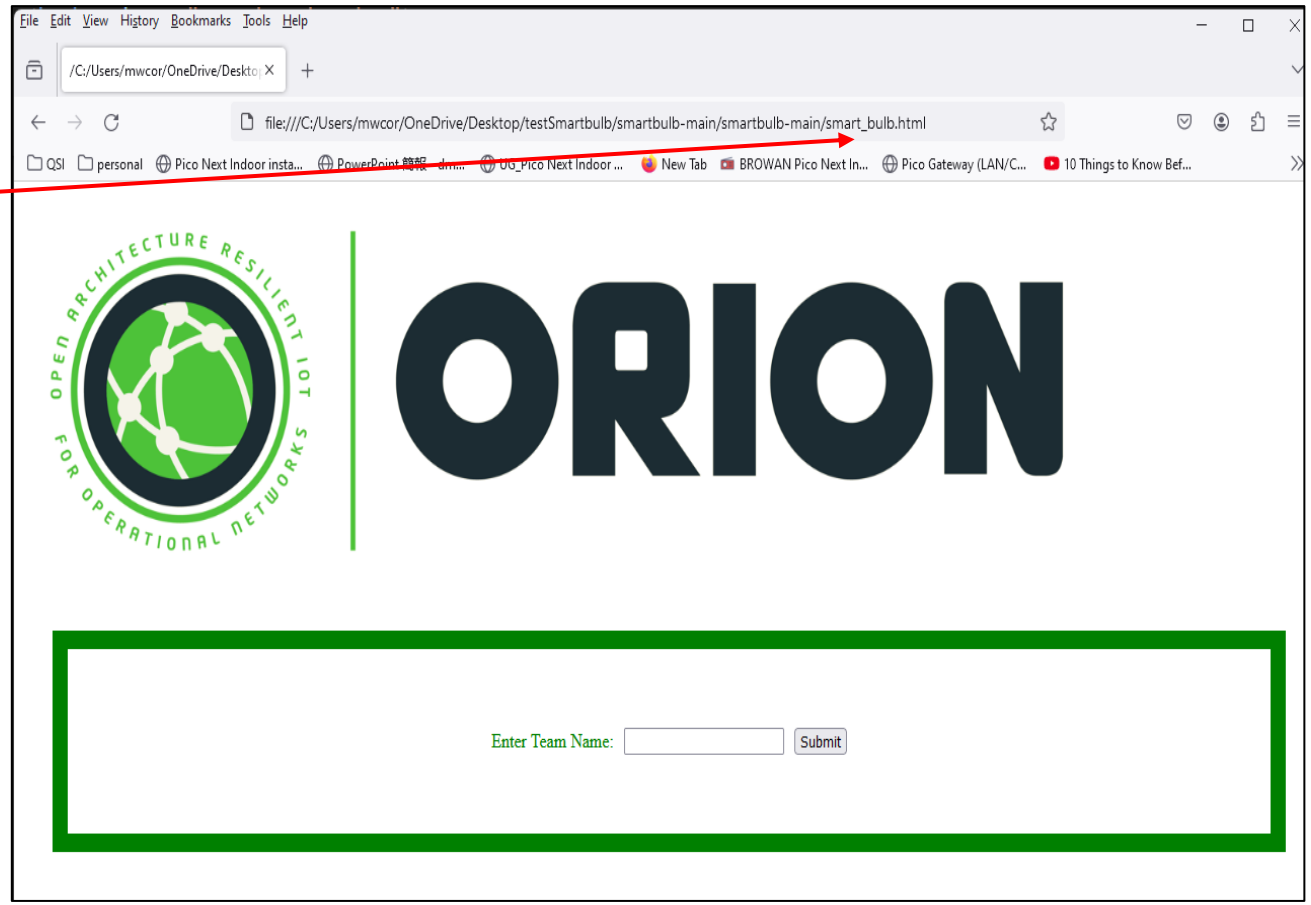
- Find the variable “host” on line 105
- Replace the value: “[ip-address]” with the IPv4 address of the Raspberry Pi and save the file.
- To find the IP address, open a terminal and SSH to the Raspberry Pi
  - run the following command: *ifconfig*
- *The ip address is shown under the wireless network interface (wlan0)*

```
pi@viperpi:~ $ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether b8:27:eb:44:09:2f  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 29  bytes 2882 (2.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 29  bytes 2882 (2.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::aa3b:5e35:6684:1555  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:11:5c:7a  txqueuelen 1000  (Ethernet)
        RX packets 4542  bytes 657989 (642.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 379  bytes 38149 (37.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

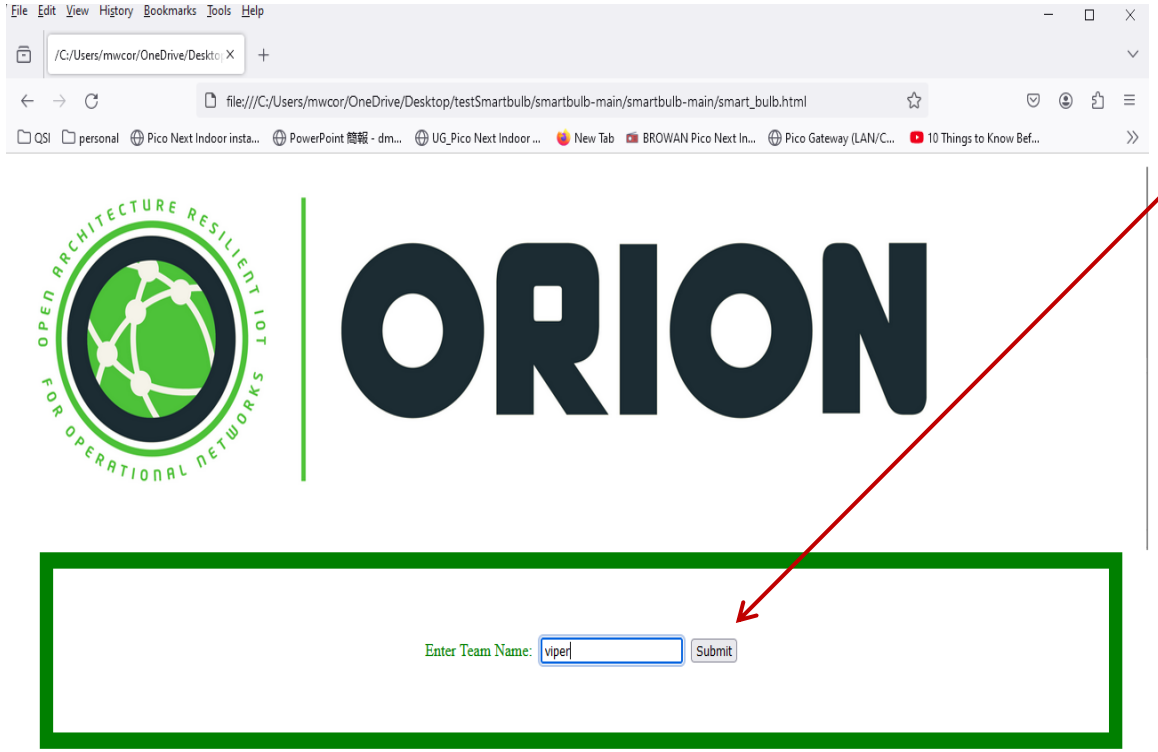
Serve up the user interface in a browser such Chrome or Firefox

Open a browser, navigate to the smartbulb project folder and open *smartbulb.html*



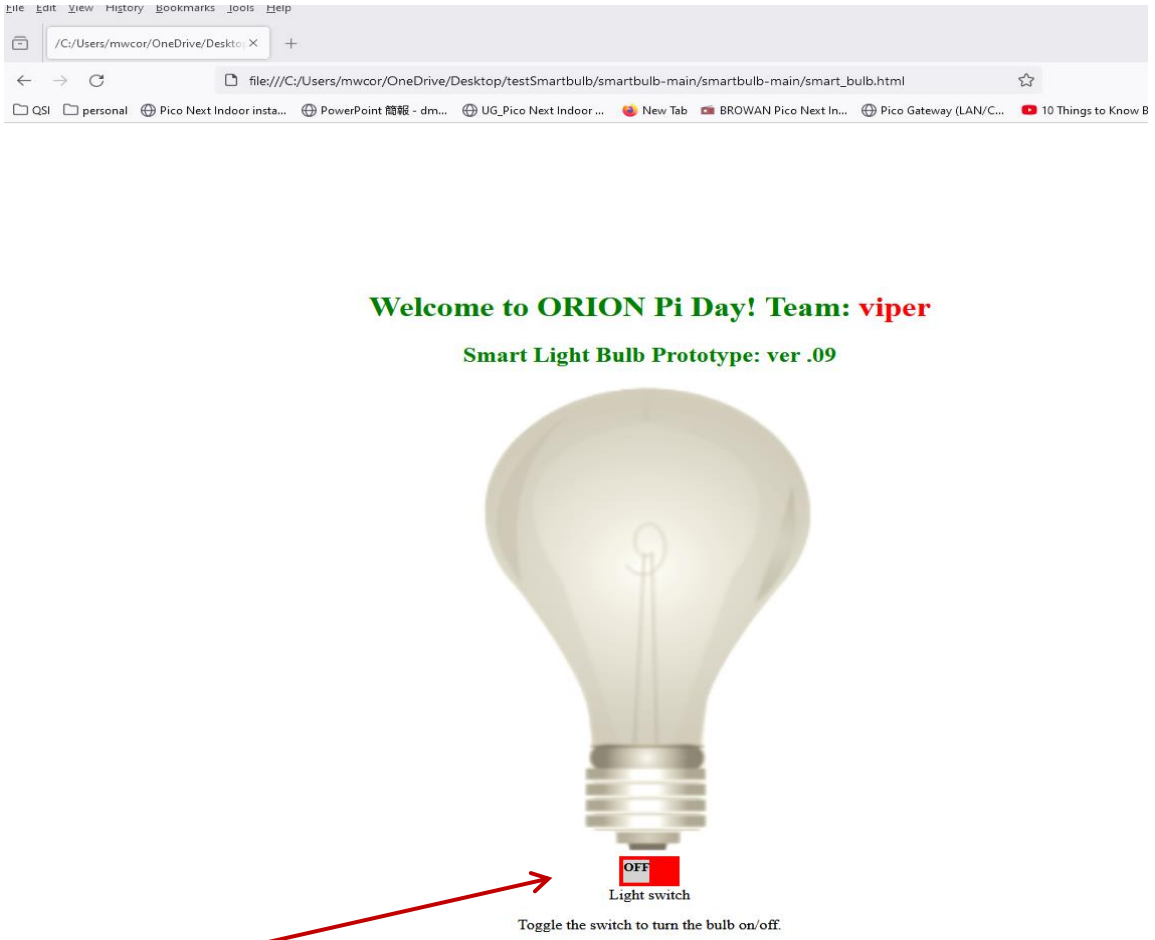


Test the Smart bulb: Make sure the back-end: smart-bulb.py is running



Front-end user interface (HTML/JS) code in VSCode:
smartbulb.html

Enter Team name and click “submit”



Toggle the light switch on/off to control the light bulb

CHALLENGE QUESTIONS FOR FUN

PI DAY STEM EVENT.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Challenge Questions, for fun…

1. Now that you have a functioning “smart” lamp.
2. How secure what you say it is?
   - Can an unauthorized 3rd party access and control your lamp?
     - Can you control another team’s lamp?
     - Try to control instructor’s desk lamp at the front of the room
3. Notice (on line 28) that your back-end code (smart_bulb.py) connects to the MQTT over local host (127.0.0.1) .
   1. Explain what localhost is (use the web) ?
   2. Why can your back-end code connect to MQTT over localhost?
4. Modify the Smart lamp to with a PIR (motion) sensor a means activate the lamp in addition to the button. You need to modify the circuit and the backend code.

5. The front-end (user) interface is developed using HTML 5/ CSS. Do you think you can change the background color (or other visual aspects) of the user interface?

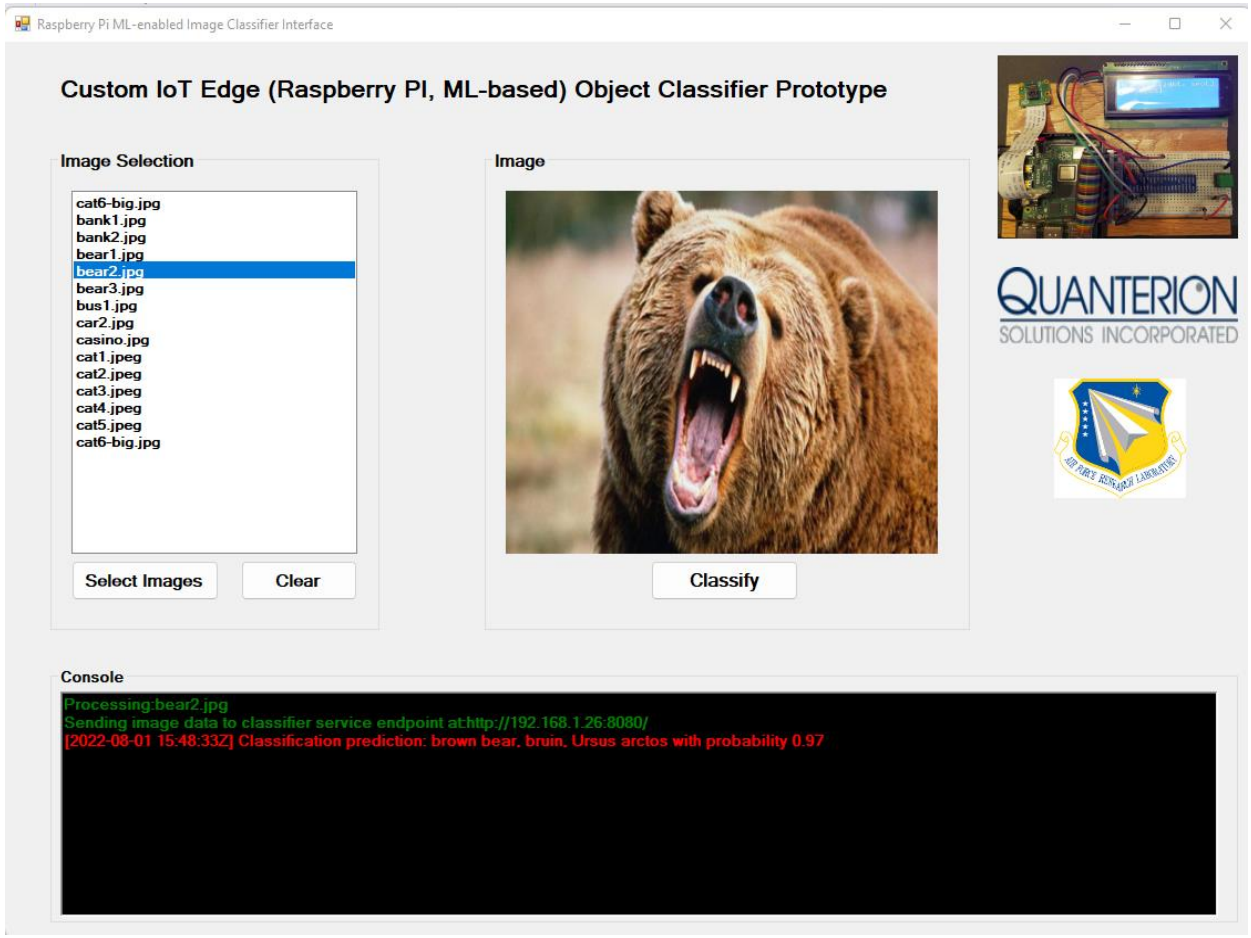
6. What was your most surprising/interesting discovery about the about Raspberry Pi Day after completing the smart lamp project?

TIME FOR A DEMO

PI DAY STEM EVENT.

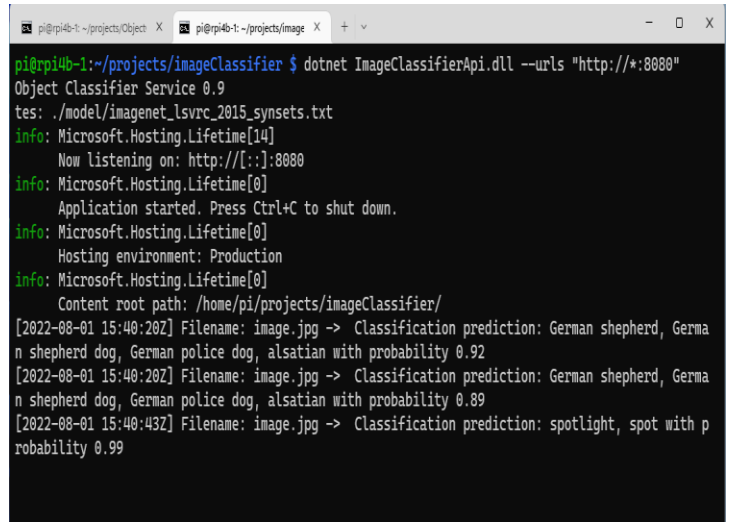
ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Raspberry Pi Edge Processing : (Artificial Intelligence Image Recognition Demo) (Time Permitting)

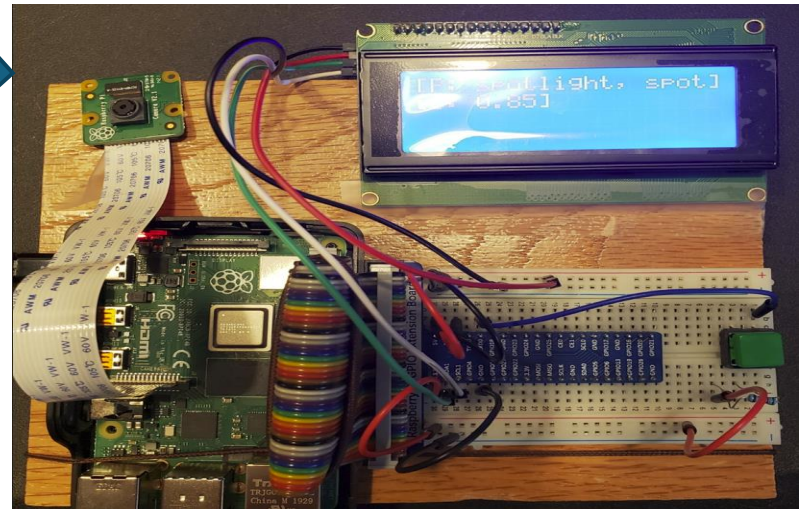


Object Classifier (External) Client – running on Windows 11

HTTP-REST



Object Classifier (REST) Service



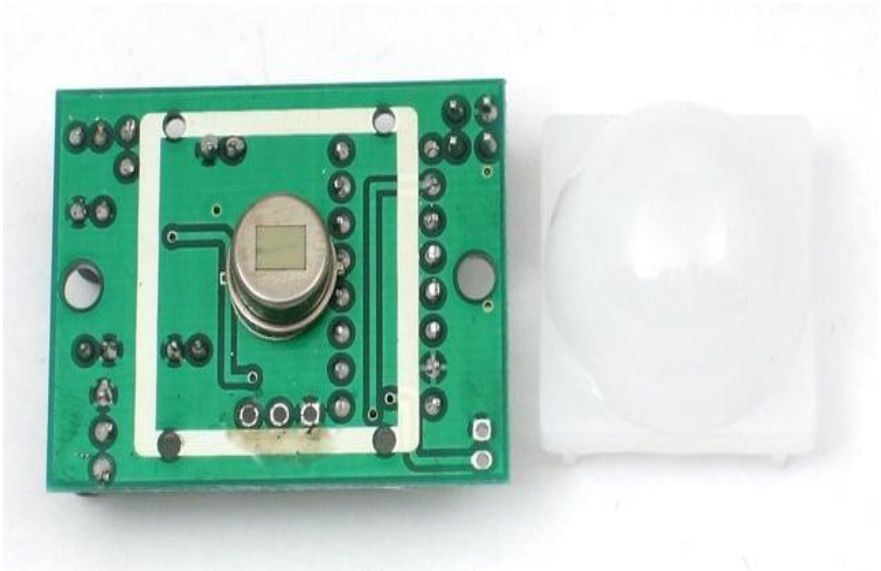
Where do we go from here?

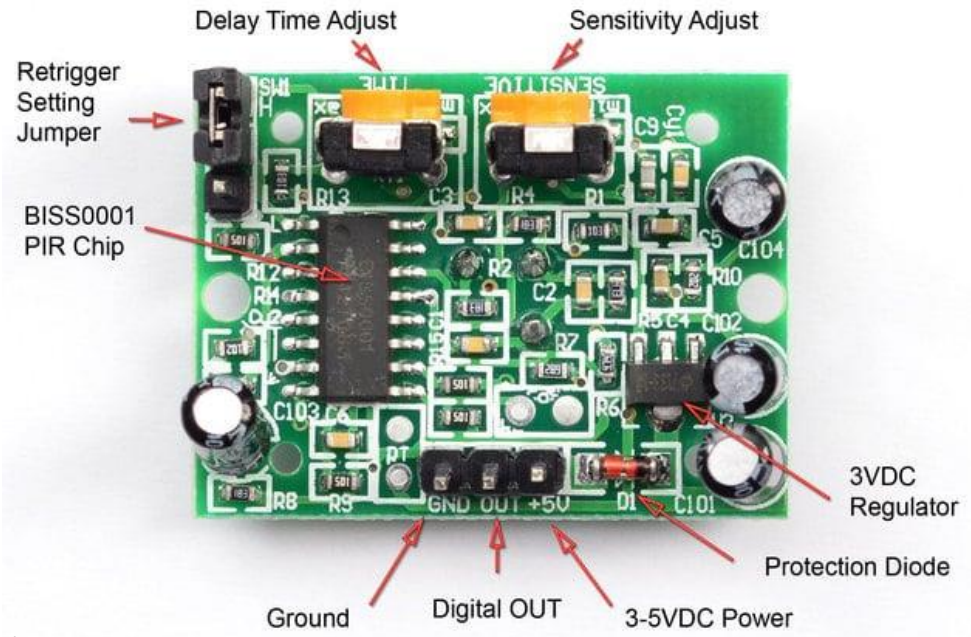
- Other STEM camps, competitions, etc.
  - https://www.griffissinstitute.org/who-we-work-with/afrl/stem
- Online based learning resources
  - https://randomnerdtutorials.com/getting-started-with-raspberry-pi/
  - Let's Learn .NET - IoT (Internet of Things) (youtube.com)
- High school course work
  - Computer Science/Programming/Cybersecurity offered
  - College careers
    - Computer Science, Computer/Electrical Engineering
      - Speak to your teachers/guidance counselors
      - Reach out to us, anytime!

PIR, "Passive Infrared", "Pyroelectric", or "IR motion" sensors

- sense motion, almost always used to detect whether a human has moved in or out of the sensors range.
- commonly found in appliances and gadgets used in homes or businesses.`
- pyroelectric sensor (which you can see below as the round metal can with a rectangular crystal in the center), which can detect levels of infrared radiation.

- Source: https://learn.adafruit.com/pir-passive-infraredproximity-motion-sensor/overview



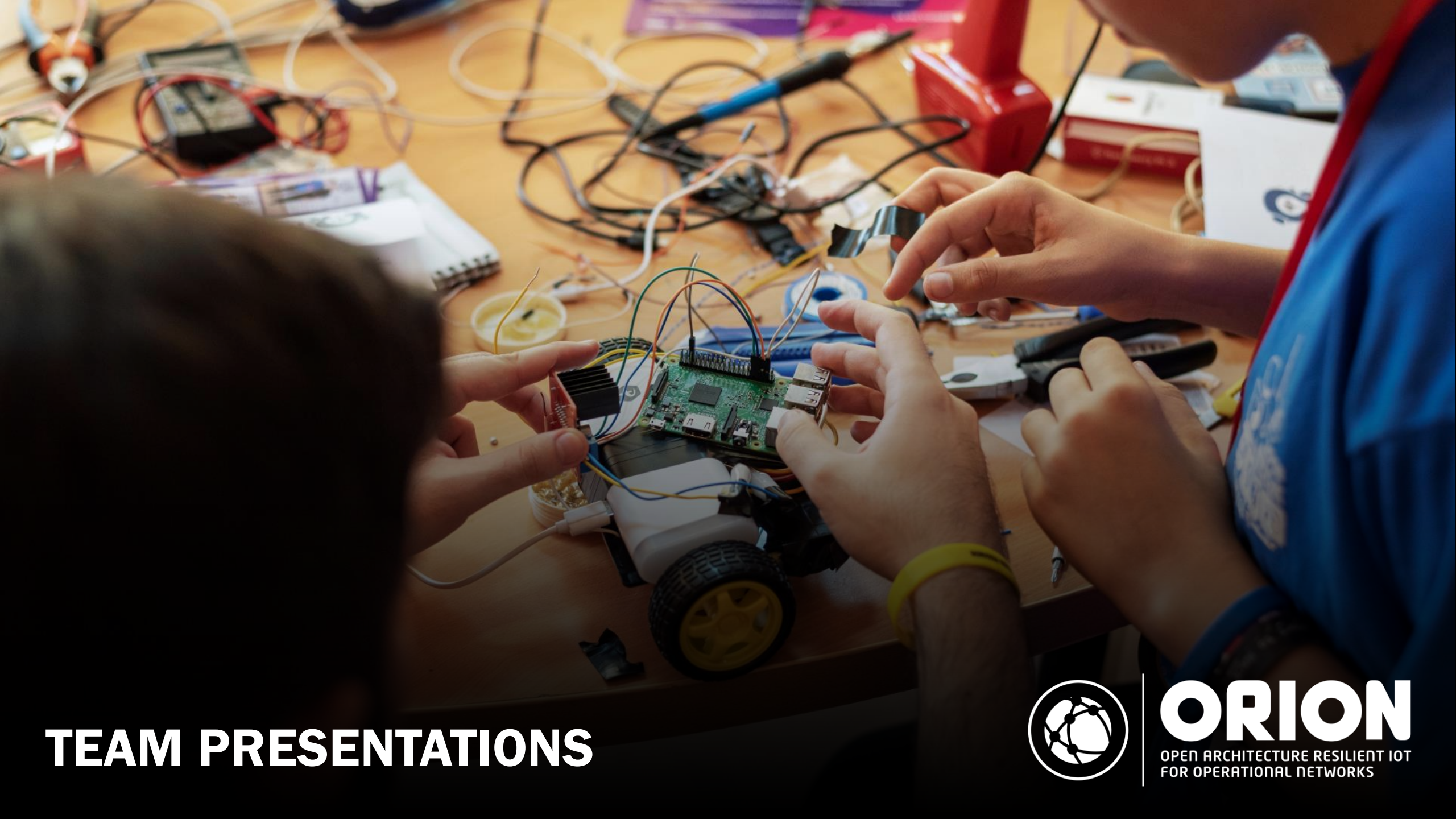


https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview

Add Motion (PIR) Detection to Smart Lamp

Add the following code snippet to the back end “smart_bulb.py”

```
pir = MotionSensor(24)
pir.when_motion = lambda: GPIO.output(ledPin, GPIO.HIGH);
pir.when_no_motion = lambda: GPIO.output(ledPin, GPIO.LOW);
```



TEAM PRESENTATIONS

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

THANK YOU FOR JOINING US!

PLEASE COMPLETE THE ONLINE SURVEY!

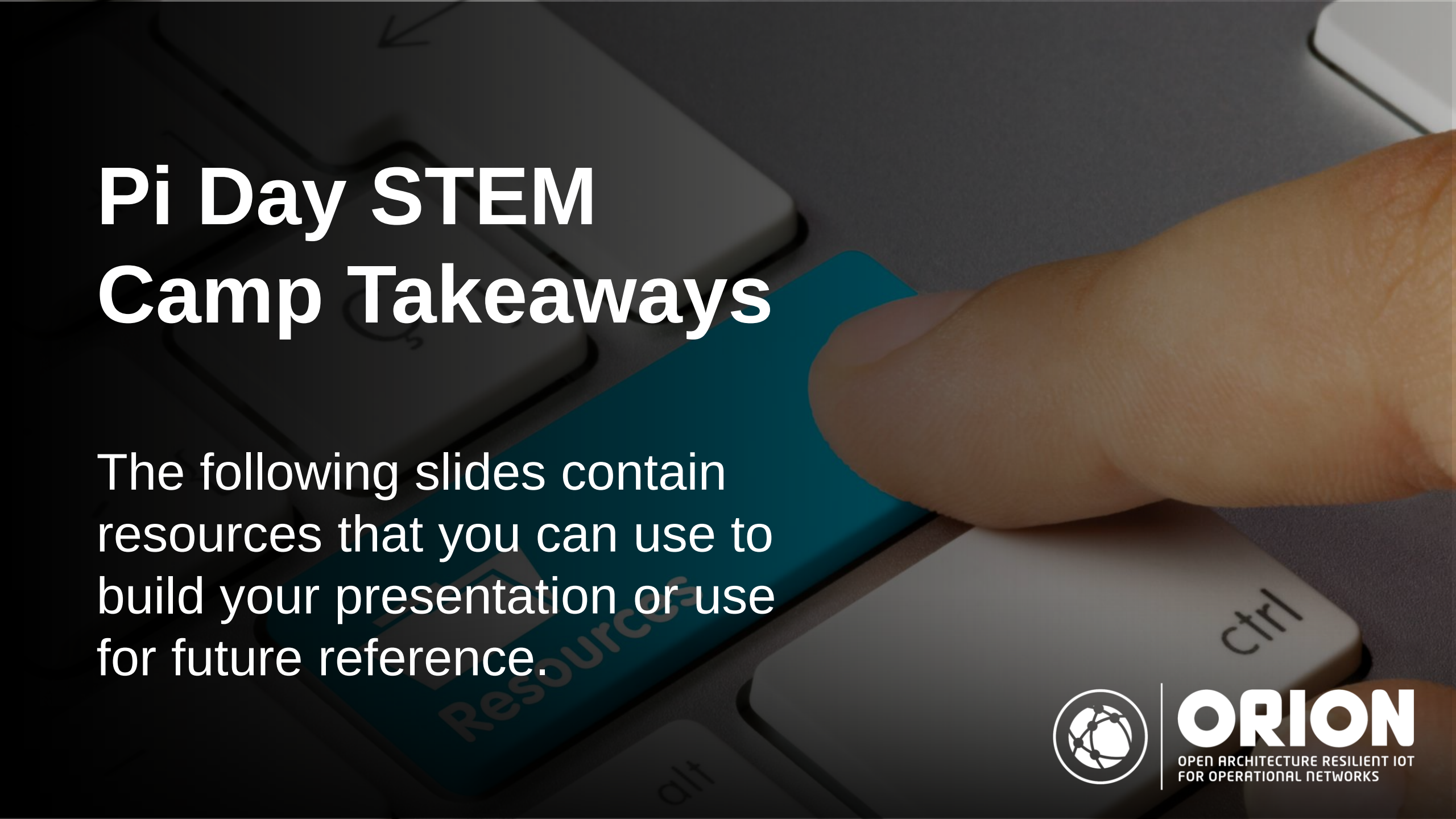
PI DAY STEM EVENT SPONSORED BY AIR FORCE RESEARCH LABORATORY, HOSTED BY THE GRIFFISS INSTITUTE.

CURRICULUM AND INSTRUCTION BY QUANTERION SOLUTIONS INCORPORATED.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Pi Day STEM Camp Takeaways

The following slides contain resources that you can use to build your presentation or use for future reference.



ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Pi Day STEM Camp Takeaways

ORION's Pi Day introduced students to microcomputers through a mentored, hands-on exercise to build an Internet of Things (IoT) smart lamp using a Raspberry Pi, while additionally discovering the enabling technologies (e.g., network and messaging protocols, programming languages, etc.).

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

IoT Resources

The proliferation of broadband Internet connectivity, increasing processing power of small devices, and the exponential growth of cloud computing capabilities have enabled an IoT revolution.

An IoT Overview is HERE.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Raspberry Pi Resources

This single board computer is an excellent resource for learning and experimenting. It provides a wealth of capability in a small form factor at a reasonable price.

A Raspberry Pi Overview HERE.

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Raspberry Pi Resources

Setting up a Raspberry Pi may require some help. Connecting to and interfacing with the Pi can involve the following:

Secure Shell (SSH)

Virtual Network Computing (VNC) viewer download HERE.

ORION

OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Message Protocol Resources

The IoT realm involves many disparate devices, services, and user applications. One messaging protocol that has emerged as a leader for IoT applications is the Message Queueing Telemetry Transport (MQTT).

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS

Electronic Components

IoT solutions typically interface with electronic components to sense some environmental condition and then perform some sort of action in response.

For your continued exploration:
Physically Interfacing with a Raspberry Pi IoT Prototypes Using a Breadboard

ORION
OPEN ARCHITECTURE RESILIENT IOT FOR OPERATIONAL NETWORKS