CLOUD WORKSHOP

2102544 IoT Fundamentals

10 April 2024

Workshop outcomes

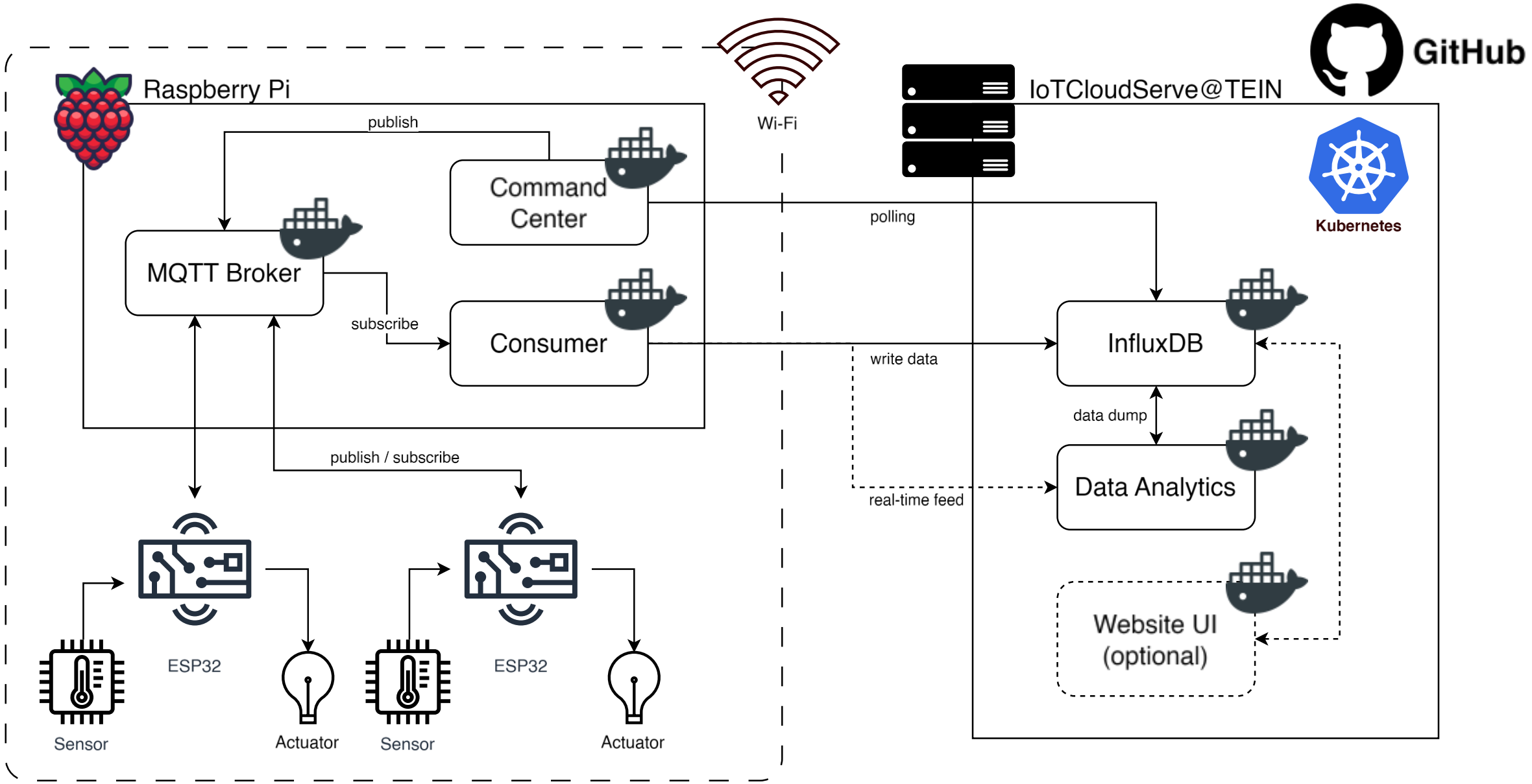
- Basic understanding of fundamental **Linux command line** operations.
- Capability to **build and deploy** applications using **Docker**.
- Ability to establish connections to a Raspberry Pi via **SSH** and **VNC Viewer**.
- Ability to **access services** running on the Raspberry Pi from a laptop.
- Familiarity with basic **Kubernetes** concepts.
- Ability to configure deployment parameters and deploy services on the **Cloud** using **Rancher**.
- Competence in developing a simple **REST API** server.
- Understanding of how to **integrate various service components** for cohesive functionality.

Before the workshop!
Please prepare the following:

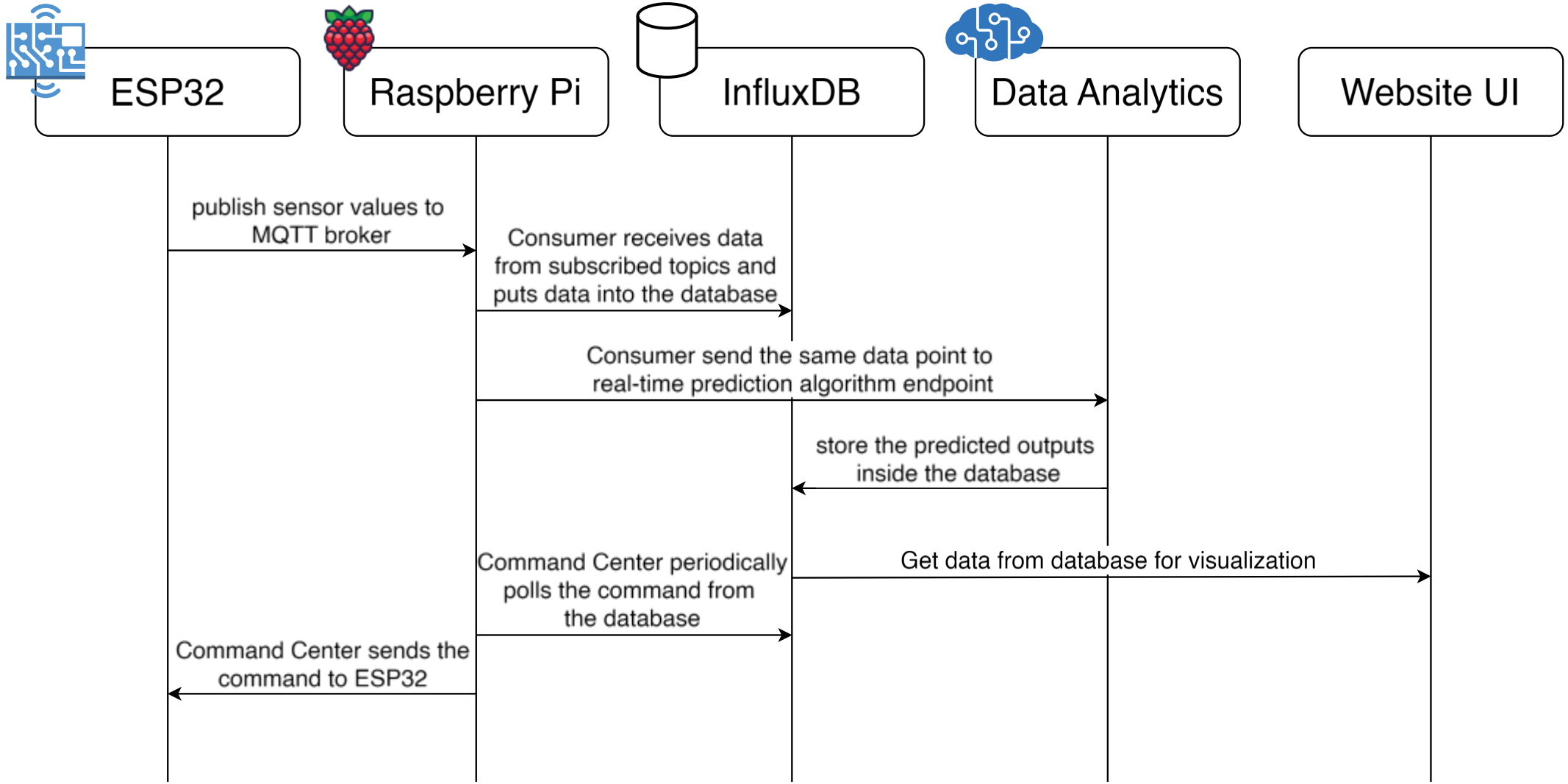
- Install Git: https://github.com/git-guides/install-git
- Install VS code: https://code.visualstudio.com/download
- Install Docker Desktop: https://docs.docker.com/get-docker/
- Register an account for the Docker Hub: https://hub.docker.com/
- Install VNC Viewer: https://www.realvnc.com/en/connect/download/viewer/
- Install Postman: https://www.postman.com/downloads/

You're free to use other software variants if you prefer, but please note that during the workshop, I'll be demonstrating based on the software list provided.

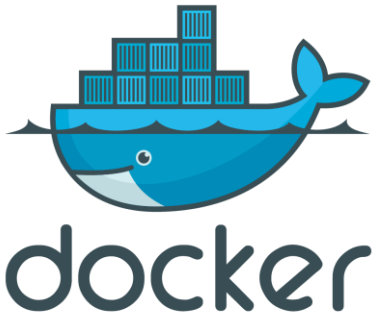
Therefore, I recommend installing the software listed to ensure you can follow along seamlessly.


GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Data Flow


ESP32
Raspberry Pi
InfluxDB
Data Analytics
Website UI
publish sensor values to MQTT broker
Consumer receives data from subscribed topics and puts data into the database
Consumer send the same data point to real-time prediction algorithm endpoint
store the predicted outputs inside the database
Command Center periodically polls the command from the database
Get data from database for visualization
Command Center sends the command to ESP32

Docker



- An open platform for developing, shipping, and running applications.
- Docker provides the ability to package and run an application in a loosely isolated environment called a **container.**
- **Containers** are lightweight and contain everything needed to run the application

https://docs.docker.com/get-started/overview/

What is a Container and how to run it? Hands-on Guide

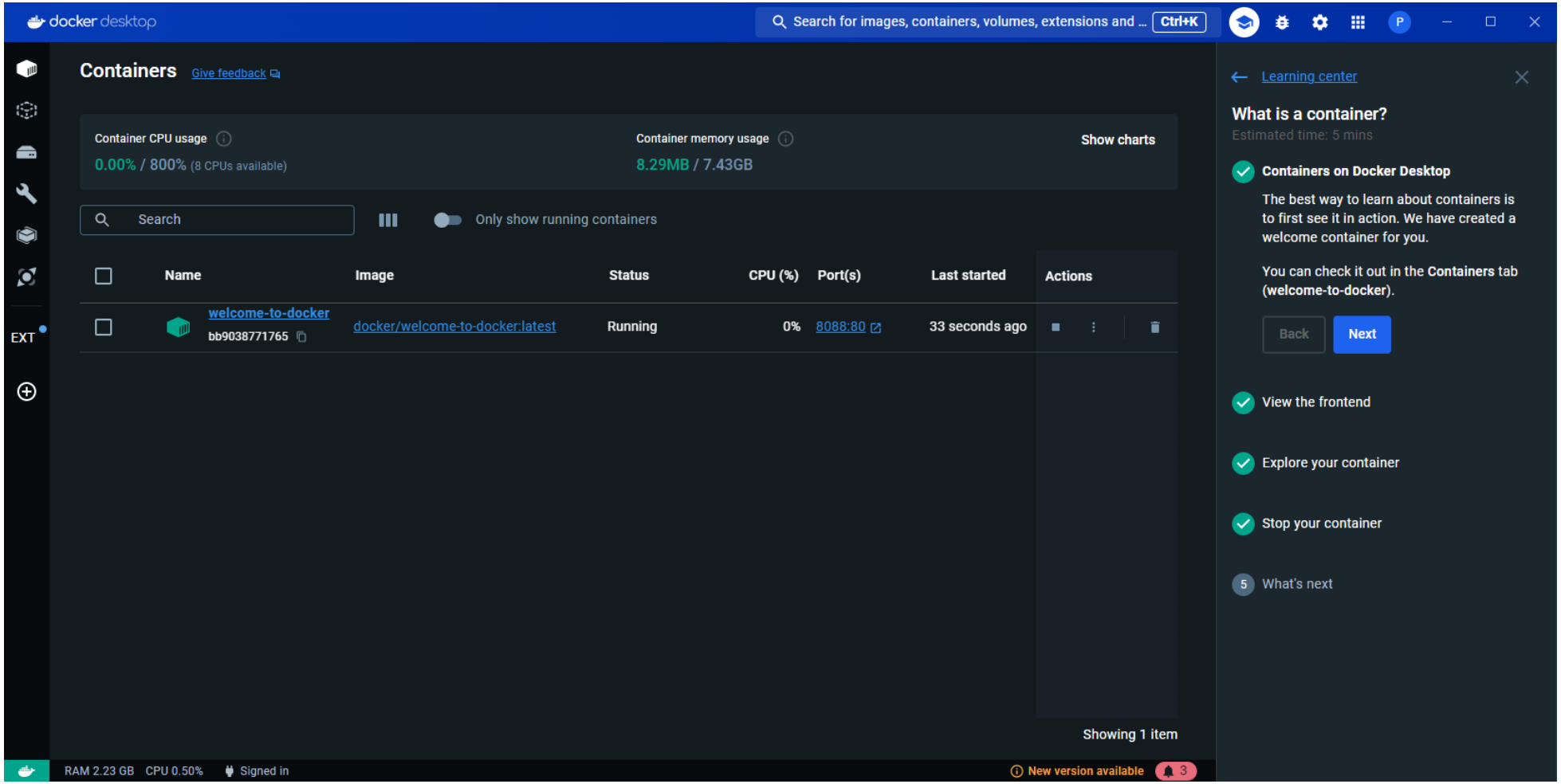
Let’s follow the guide from
https://docs.docker.com/guides/walkthroughs/what-is-a-container/

https://docs.docker.com/guides/walkthroughs/run-a-container/

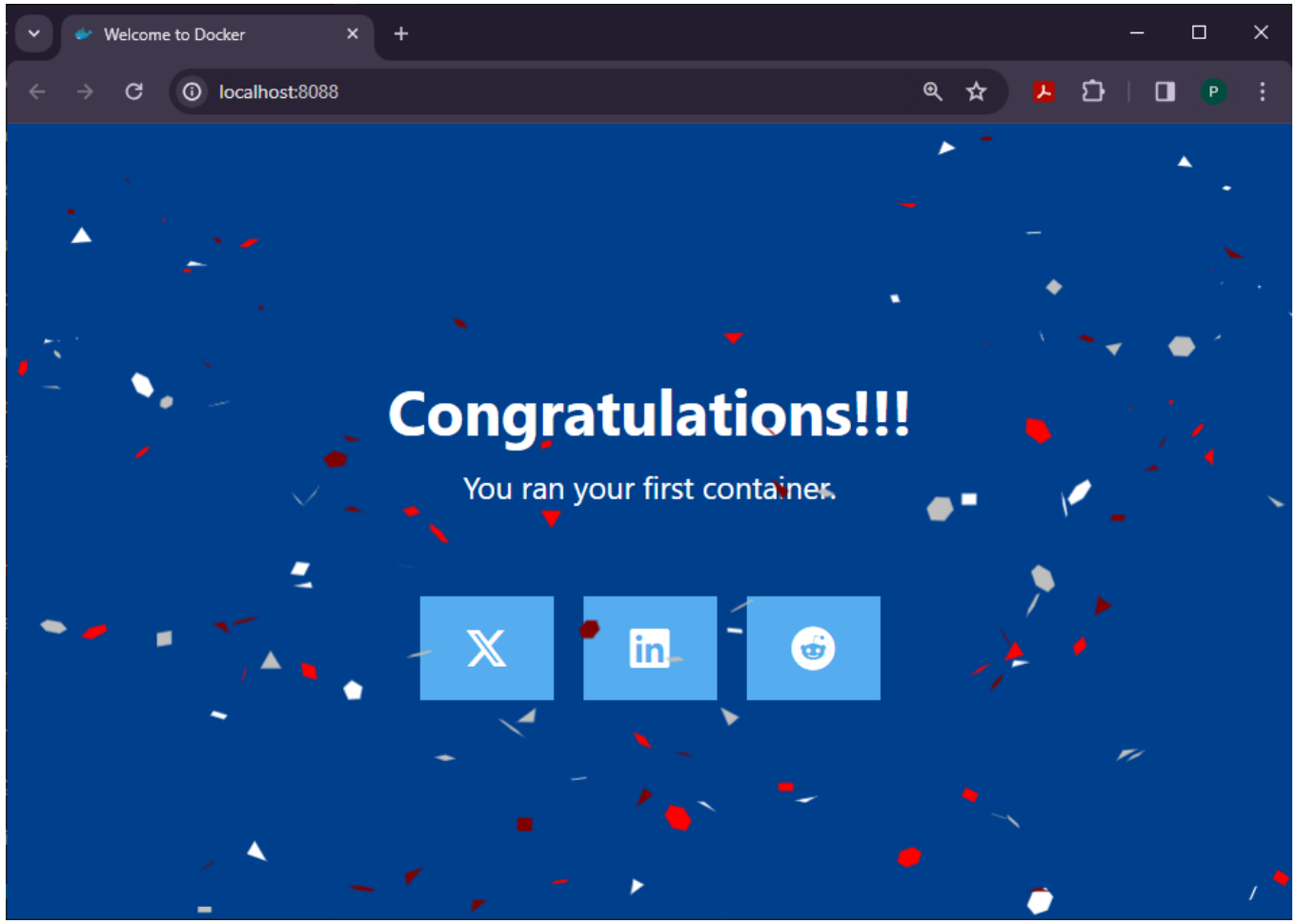
https://docs.docker.com/guides/walkthroughs/run-hub-images/

https://docs.docker.com/guides/walkthroughs/publish-your-image/

What is a Container and how to run it?



What is a Container and how to run it?



What is a Container and how to run it?

Open VS code > Add folder to workspace

> git clone https://github.com/docker/welcome-to-docker

View the details inside a *Dockerfile*

What is a Container and how to run it?

Build your image

> Open the command prompt at the cloned directory

> docker build -t welcome-to-docker .

> docker images

Run your image

> docker run -d -p 8088:3000 --name welcome-to-docker welcome-to-docker

>docker ps

What is a Container and how to run it?

Publish your image

1) Log in to Docker Hub with Docker Desktop
2) Change image name to be <Username>/<Image_name>:<tag>

> docker tag welcome-to-docker USERNAME/welcome-to-docker:v1

> docker push

3) Go to Docker Hub and find your image

What is a Container and how to run it?

Remove images

> docker images

> docker rmi <Image_name>

Pull image from Docker Hub

> docker pull USERNAME/welcome-to-docker:v1

> docker images

Docker Command Summary

Docker CLI Cheat sheet: https://www.docker.com/resources/cli-cheat-sheet/

Build an Image from a Dockerfile
docker build -t <image_name>

List local images
docker images

Run a container with and publish a container's port(s) to the host.
docker run -p <host_port>:<container_port> <image_name>

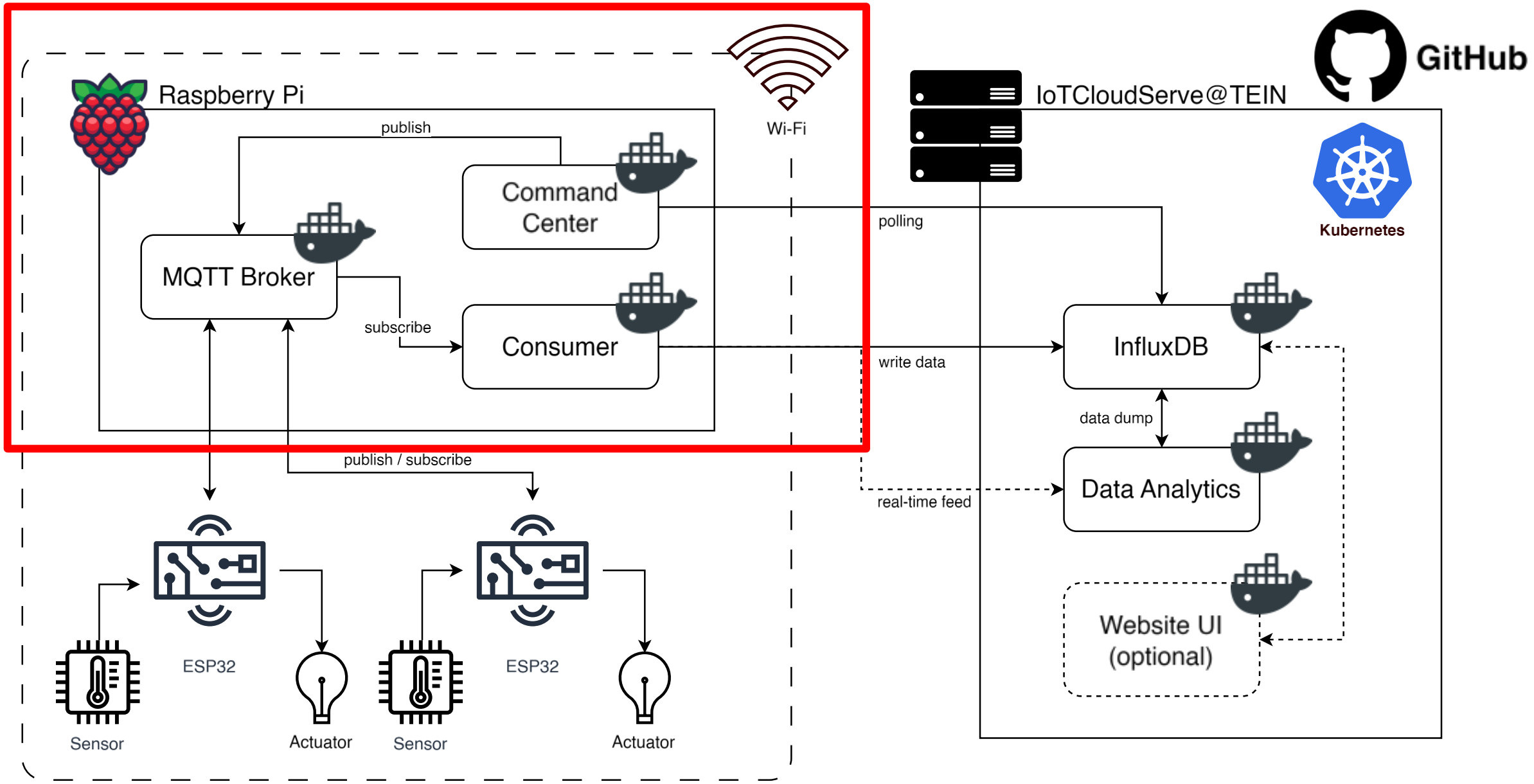
To list currently running containers:
docker ps

List all docker containers (running and stopped):
docker ps --all

Login into Docker
docker login -u <username>

Publish an image to Docker Hub
docker push <username>/<image_name>

Pull an image from a Docker Hub
docker pull <image_name>

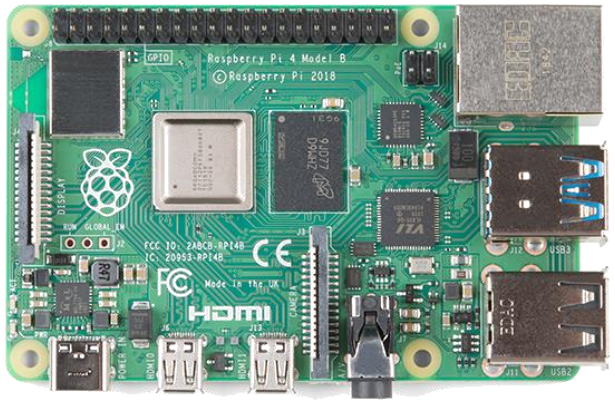

GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Establishing LAN Connection

*We need **LAN** for using SSH and VNC Viewer
You'll have several options for connecting network for Raspberry Pi

1. Connect Pi and Laptop with the same Wi-Fi
   A. Use monitor to connect Wi-Fi on Raspberry Pi
   B. In today's session, connect Raspberry Pi to your mobile hotspot so that Raspberry Pi remembers the SSID and Password.
2. Use LAN Cable to connect with the Ethernet port

Connect Wireless LAN Network



Group 1-3

TP-LINK_E0BE

14061653

Group 4-6

TP-Link_735E

15412904

Group 7-9

TP-Link_8A28

42695318

Connect Raspberry Pi via SSH

*We need **LAN** for connecting via SSH

** Change the number according to your group number

> ssh pi-group-**1**@raspberrypi**1**.local

*Note: pi-group-1 is your **username** of pi and rasberrypi1 is your **hostname***

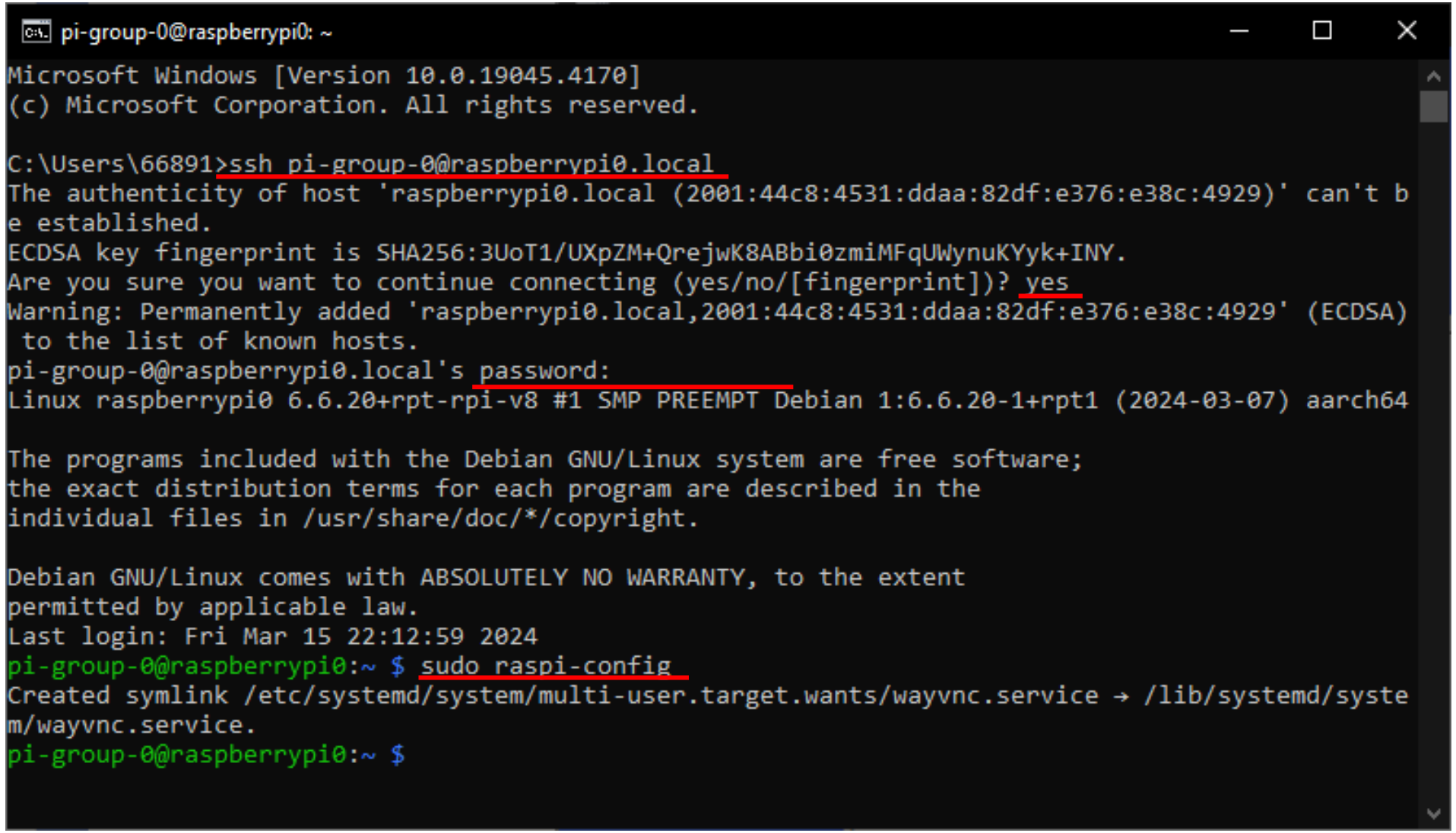
Enter the password: iotfun2023

Now you can access Raspberry Pi using the command line

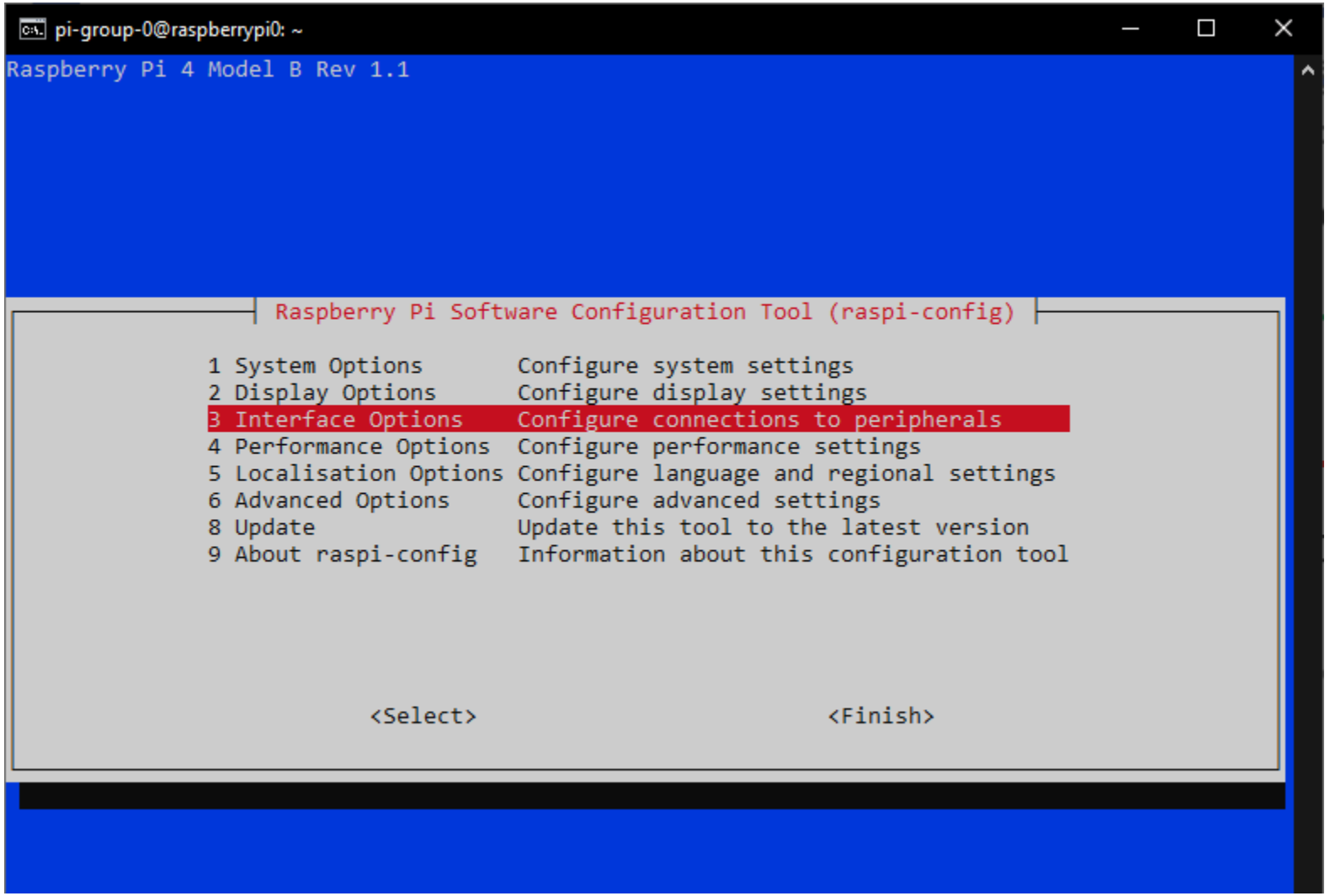
> sudo raspi-config

Go to Interface Options > Enable VNC

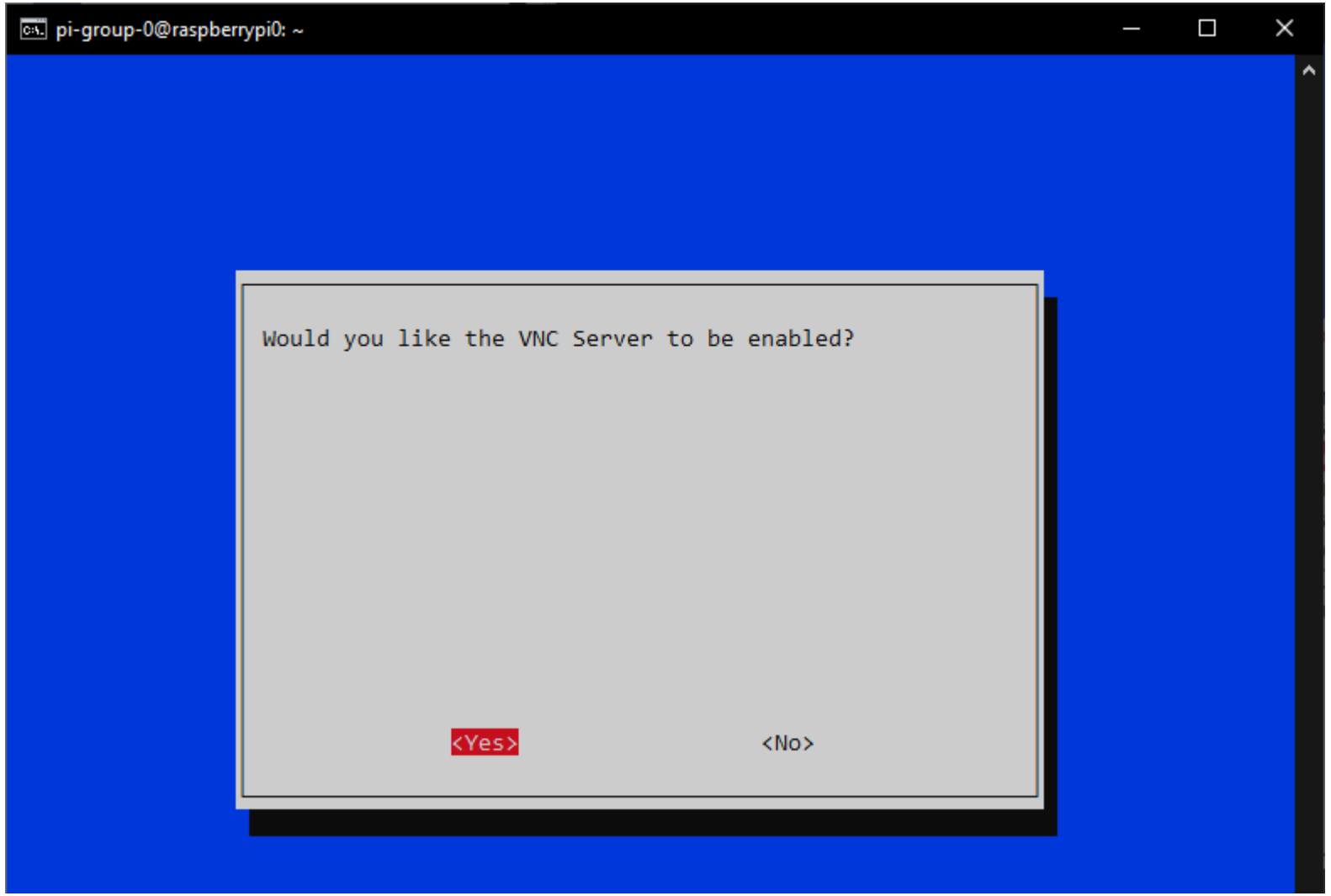
Connect Raspberry Pi via SSH



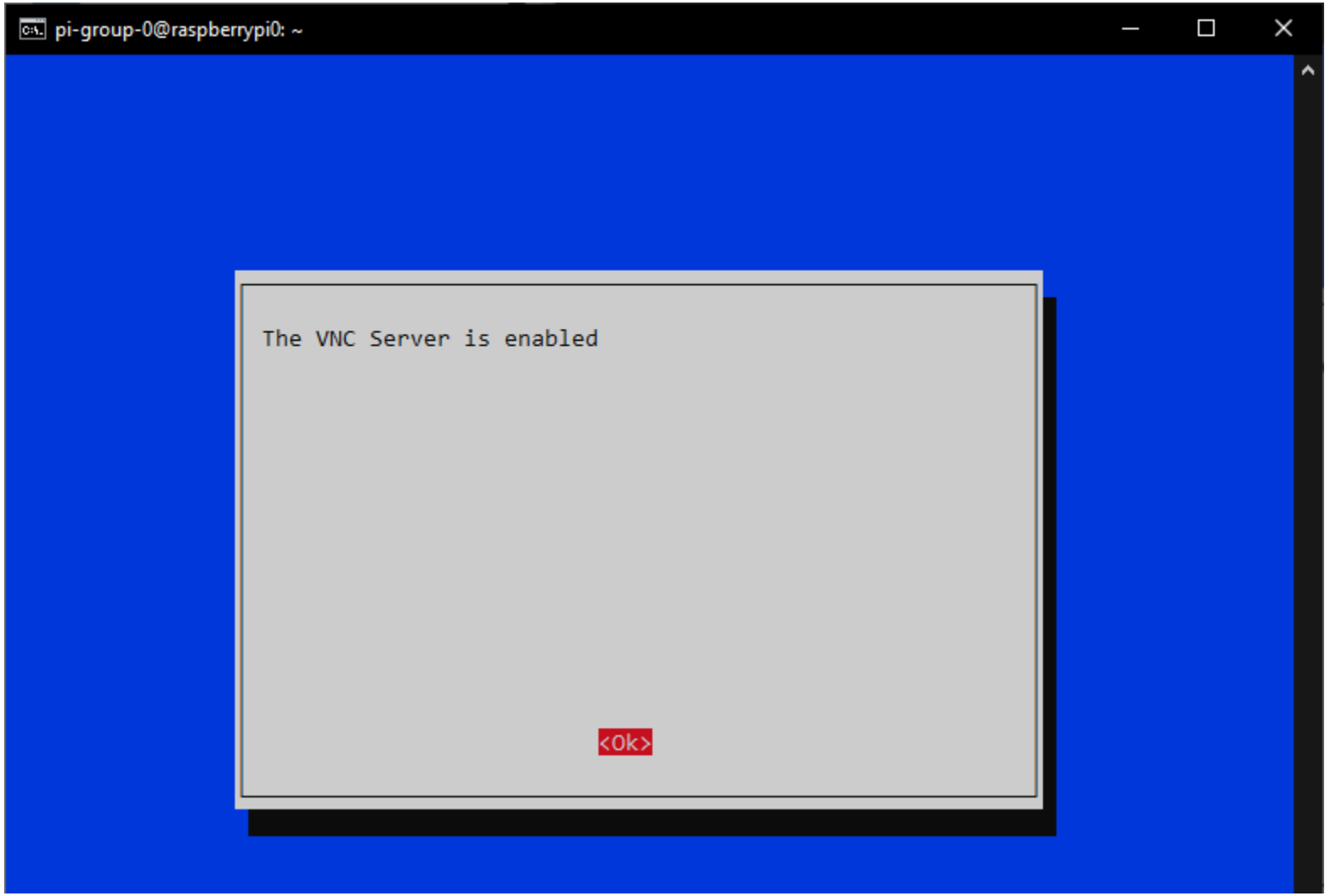
Connect Raspberry Pi via SSH



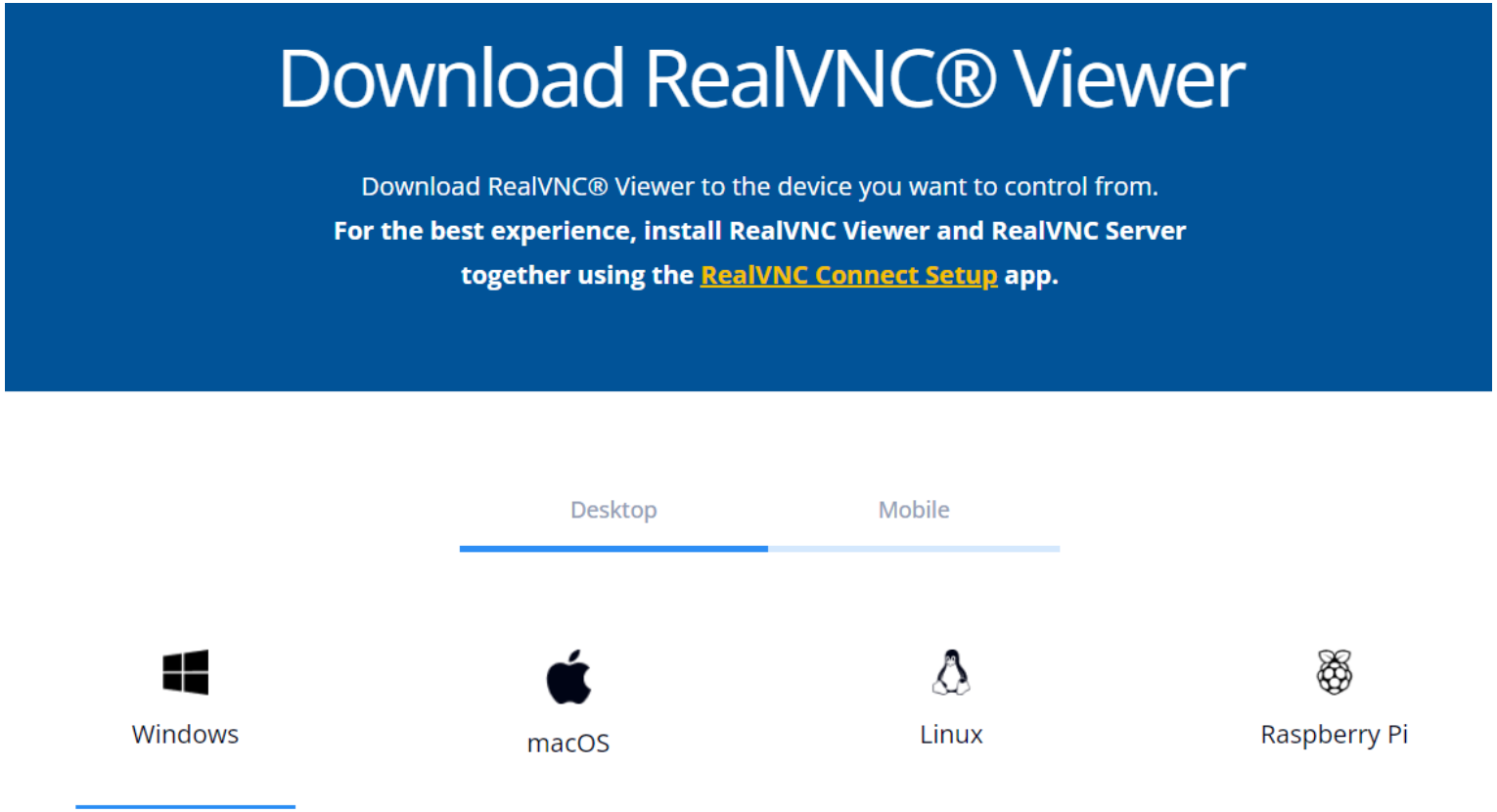
Connect Raspberry Pi via SSH



Connect Raspberry Pi via SSH



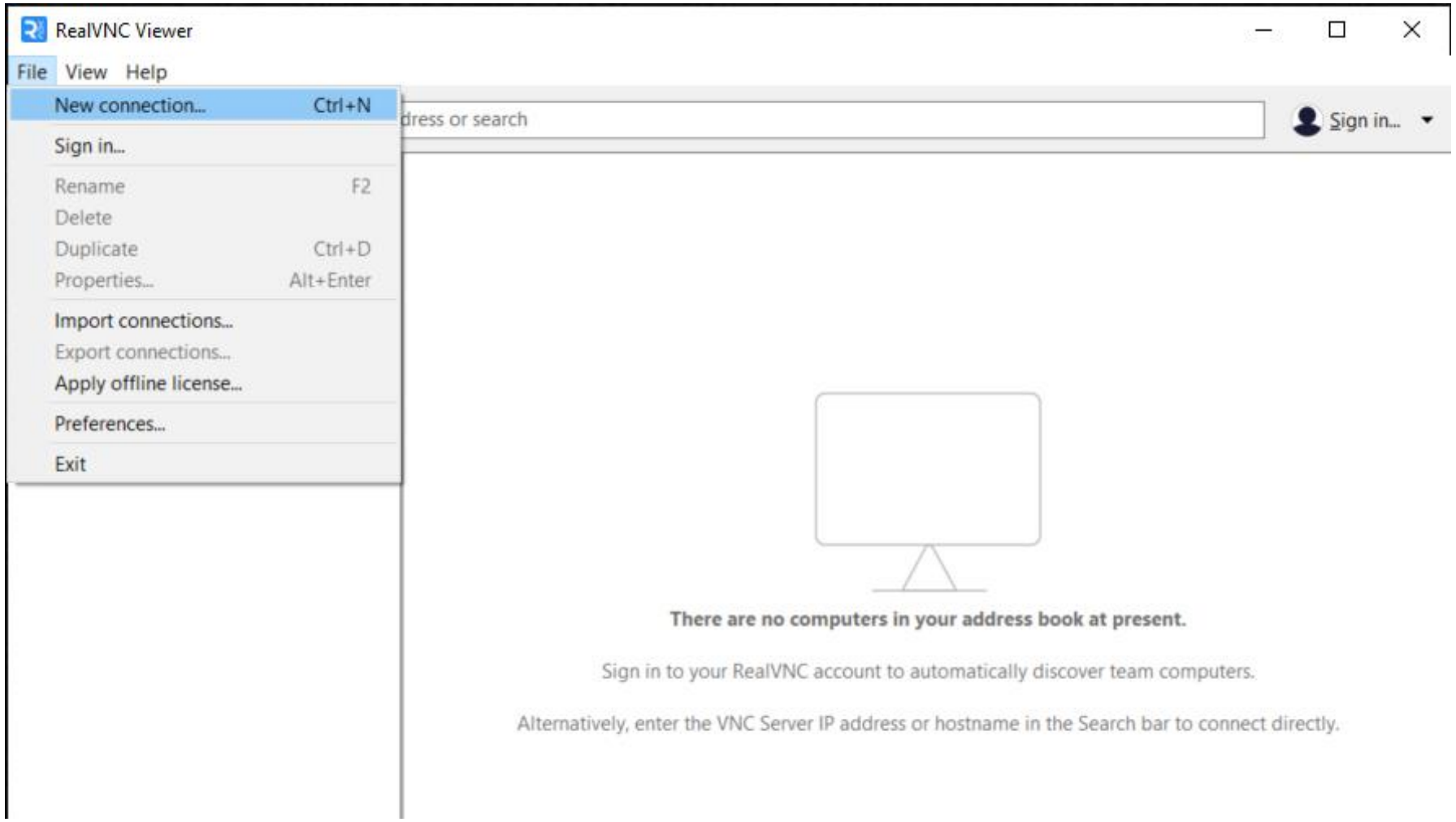
Download VNC Viewer



*We need LAN for using VNC Viewer

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

VNC Viewer

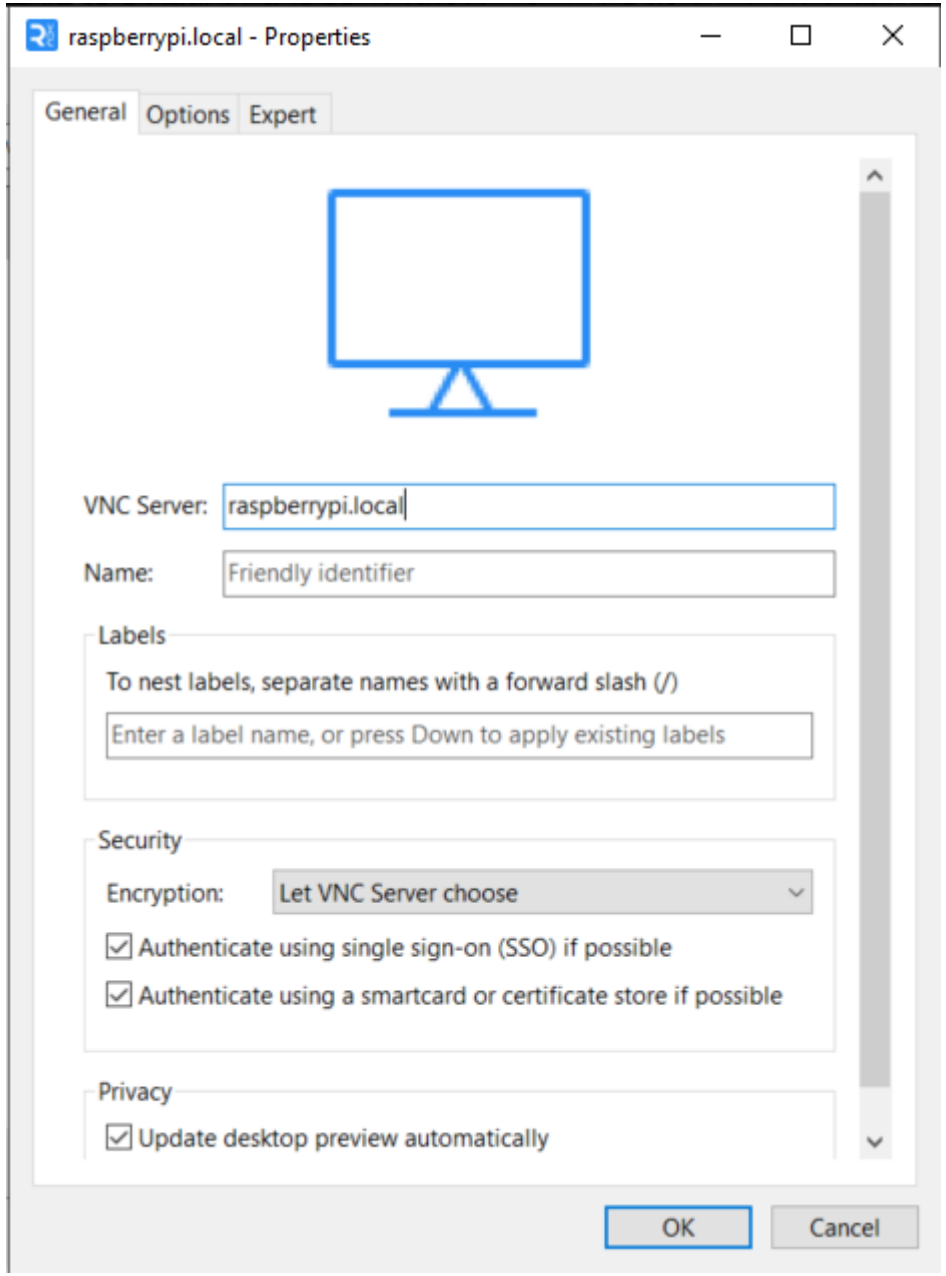

RealVNC Viewer
File
View
Help
New connection... Ctrl+N
Sign in...
Rename F2
Delete
Duplicate Ctrl+D
Properties... Alt+Enter
Import connections...
Export connections...
Apply offline license...
Preferences...
Exit
dress or search
Sign in...
There are no computers in your address book at present.
Sign in to your RealVNC account to automatically discover team computers.
Alternatively, enter the VNC Server IP address or hostname in the Search bar to connect directly.

VNC Viewer

VNC Server:

- [Hostname].local
- or IP Address

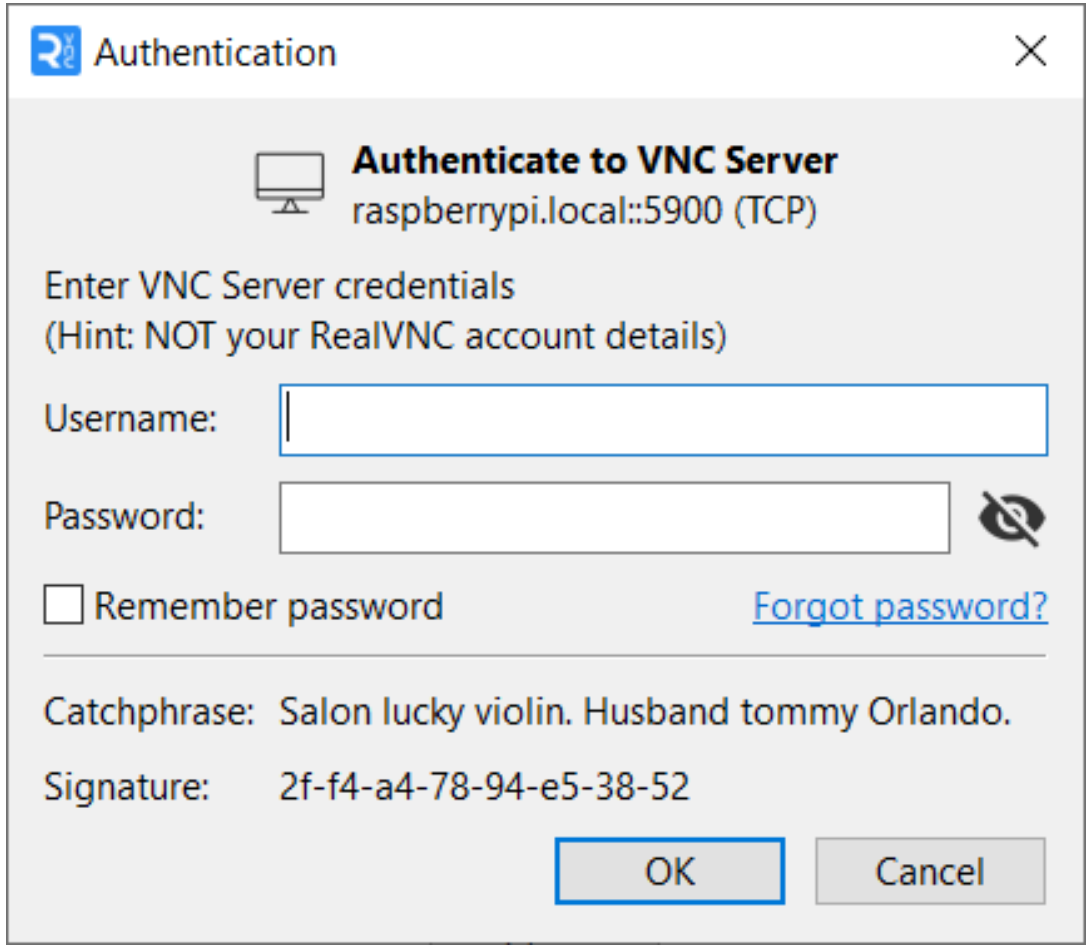
*Note: pi-group-**1** is your **username** and rasberrypi**1** is your **hostname***



VNC Viewer

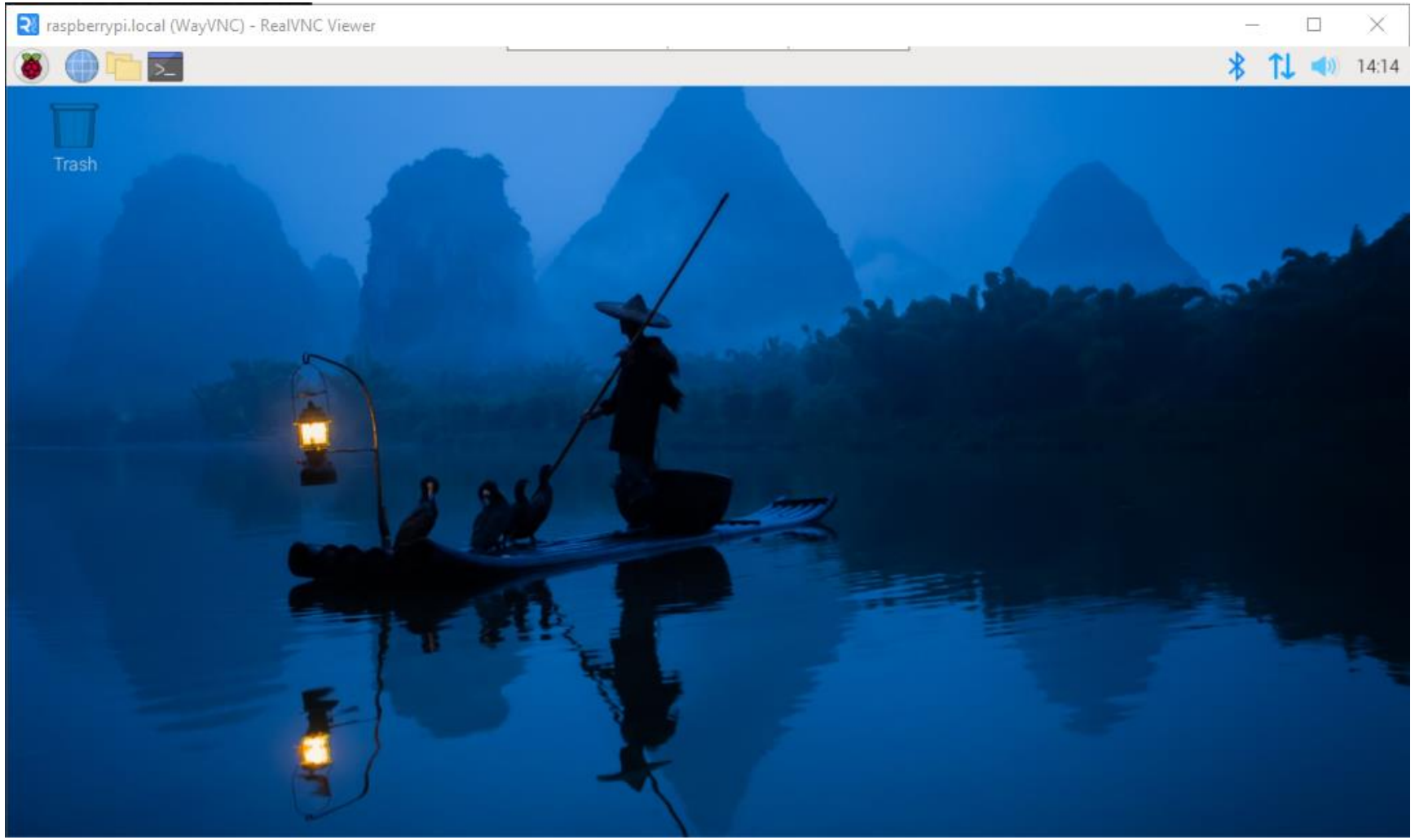
Username: [Pi username]
Password: iotfun2023

*Note: pi-group-**1** is your **username** and rasberrypi**1** is your **hostname***

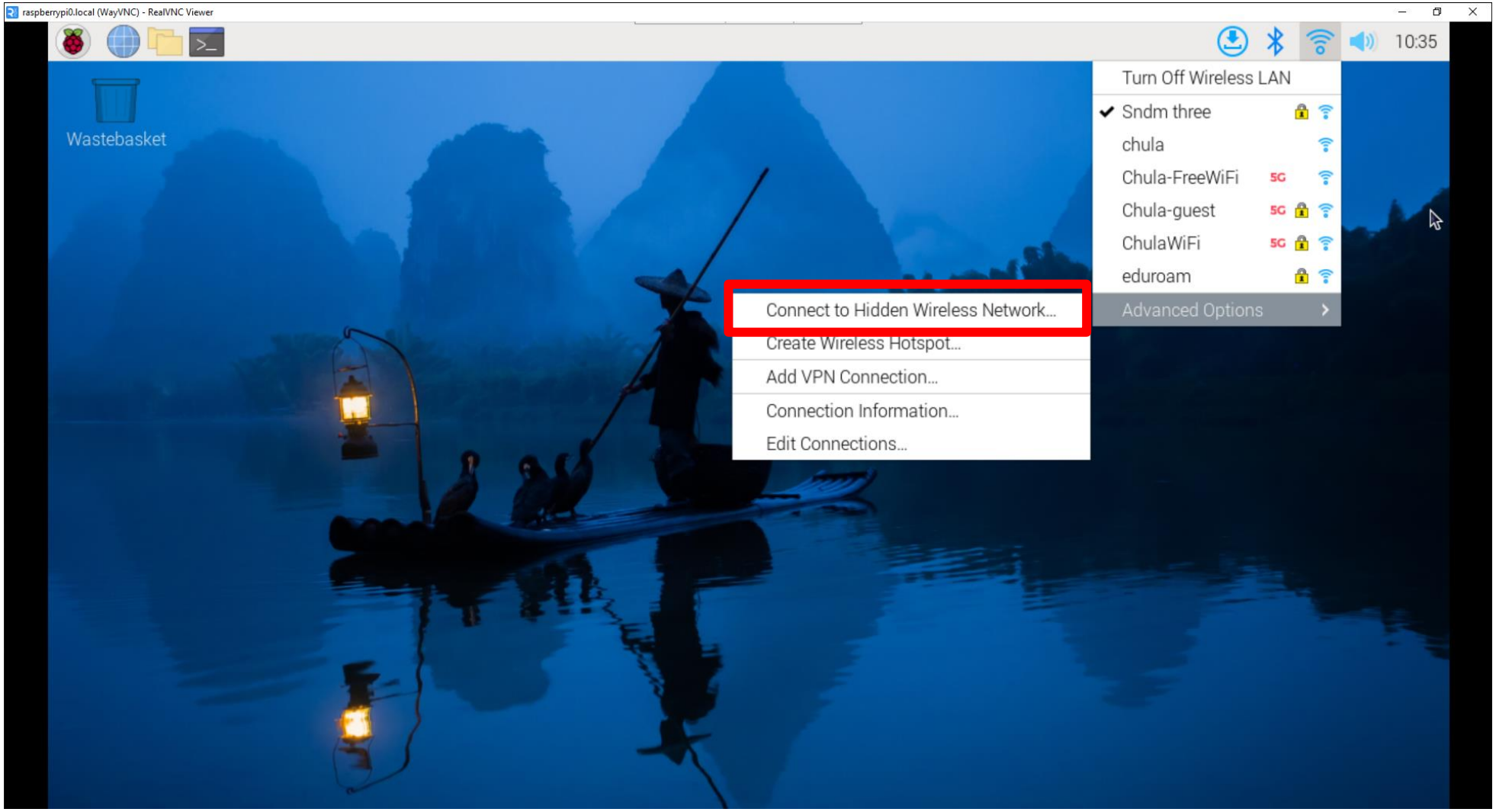


VNC Viewer

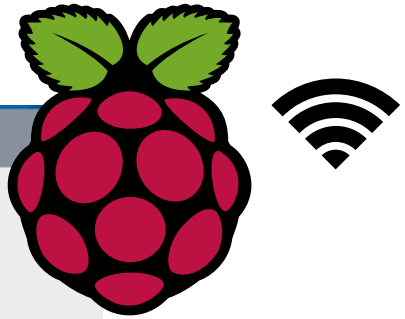
You can now access to Pi screen



Connect to your Mobile Hotspot

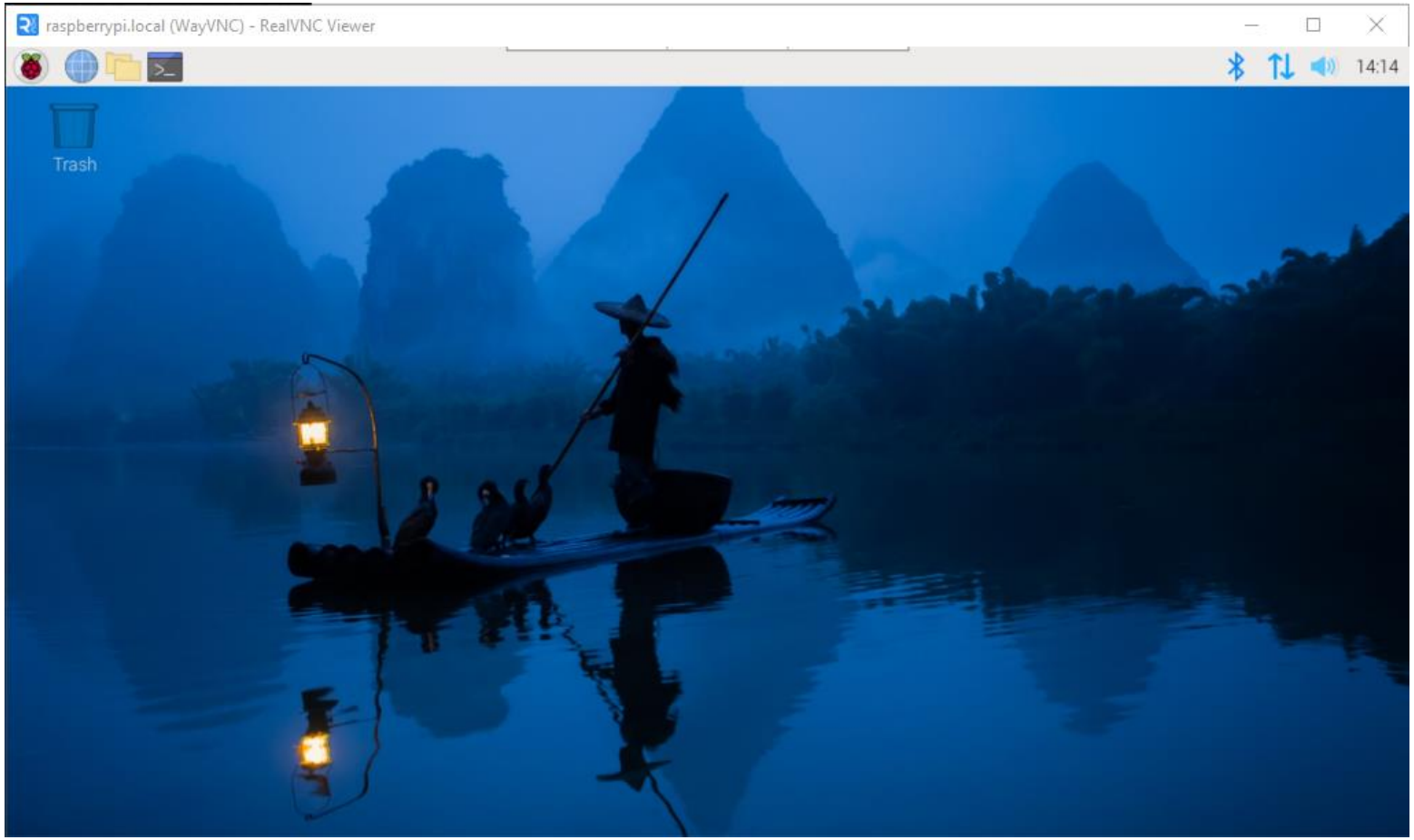


Connect to your Mobile Hotspot



VNC Viewer

You can now access to Pi screen with your hotspot



VNC Viewer

You can switch back to the network from the given Wi-Fi

Please ensure to disable your hotspot, allowing the Raspberry Pi to attempt an automatic reconnection to the default Wi-Fi network.

Note: For those who don't need to connect to the Raspberry Pi, please use **ChulaWiFi** because the router capacity is limited.

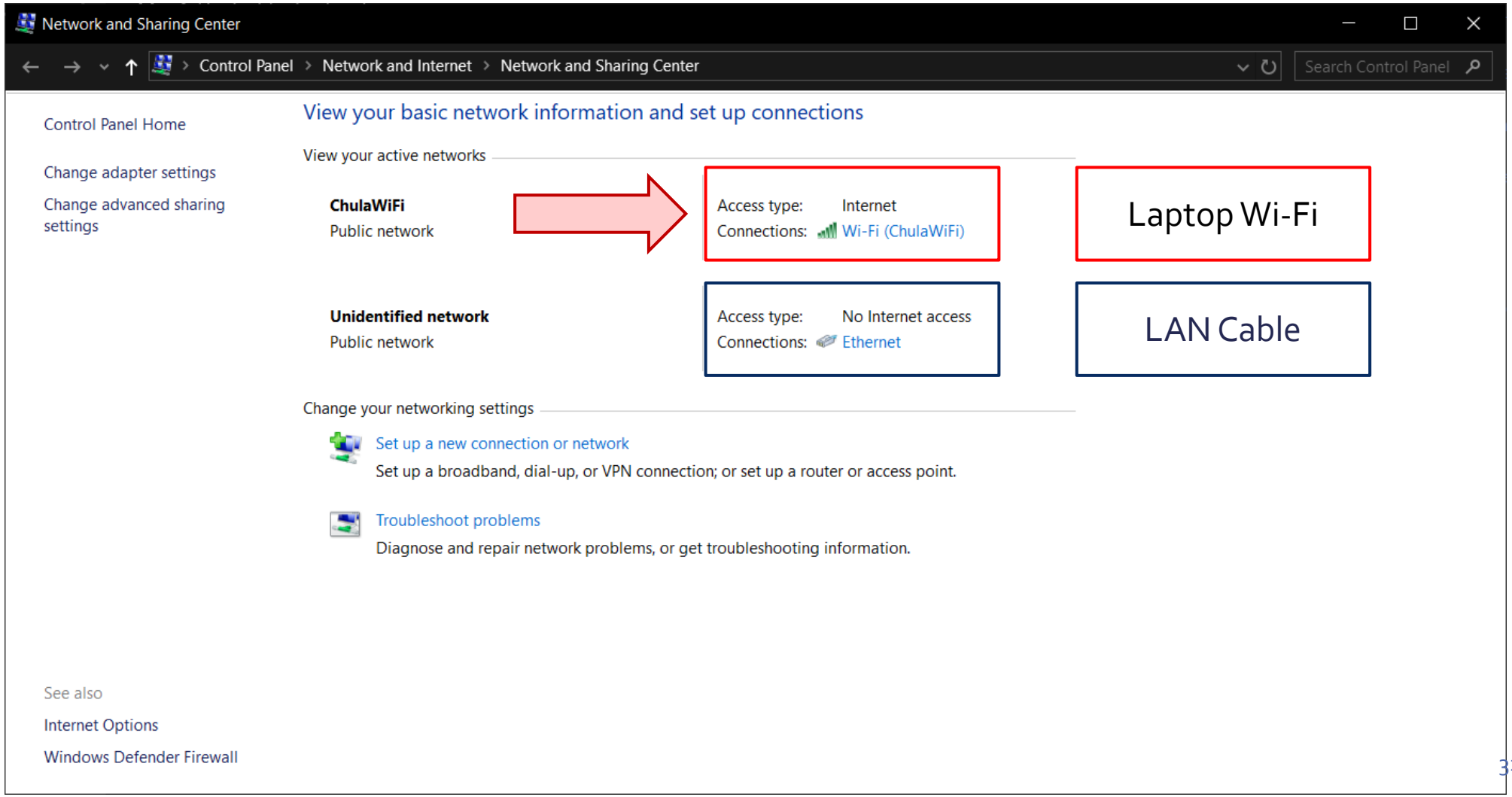
Group 1-3
TP-LINK_E0BE

Group 4-6
TP-Link_735E

Group 7-9
TP-Link_8A28

(Optional) Setup Network for Raspberry Pi

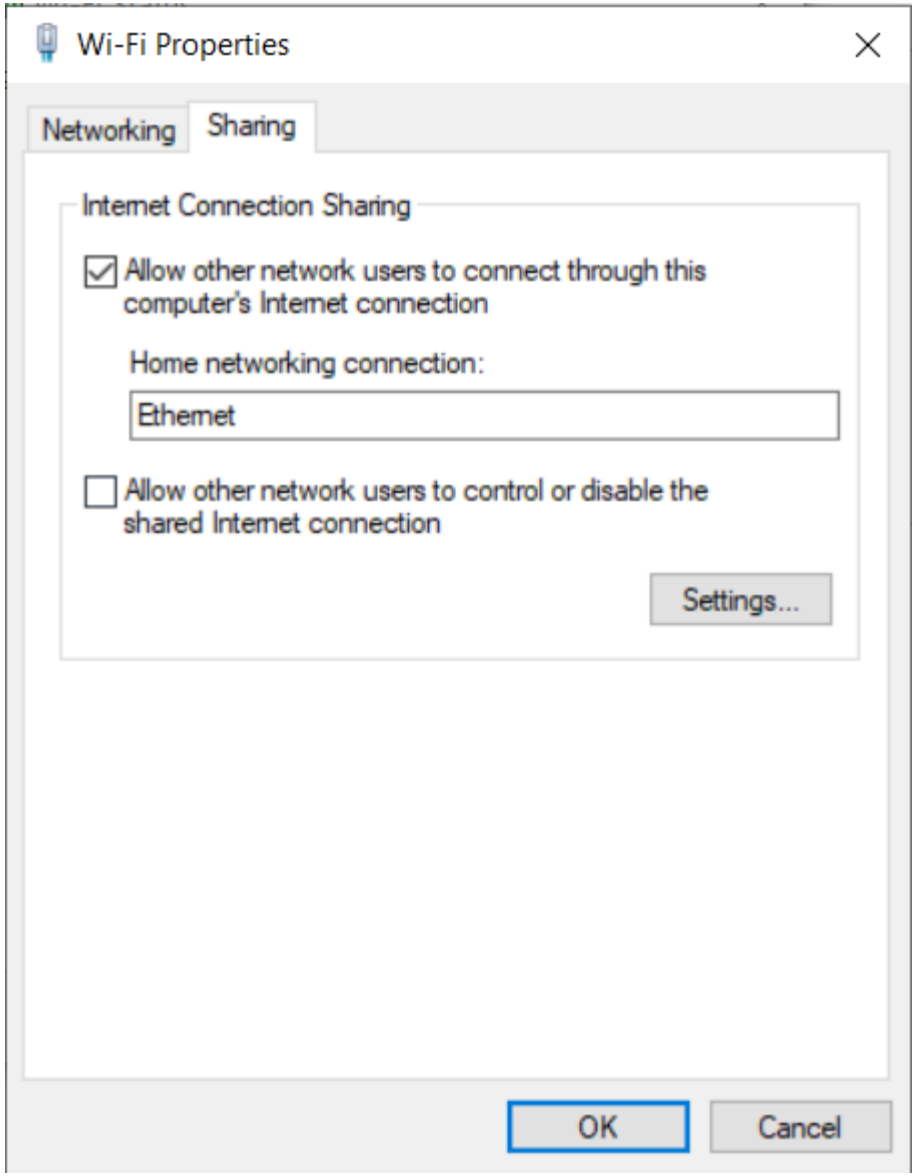
For those who connect LAN using LAN Cable only!



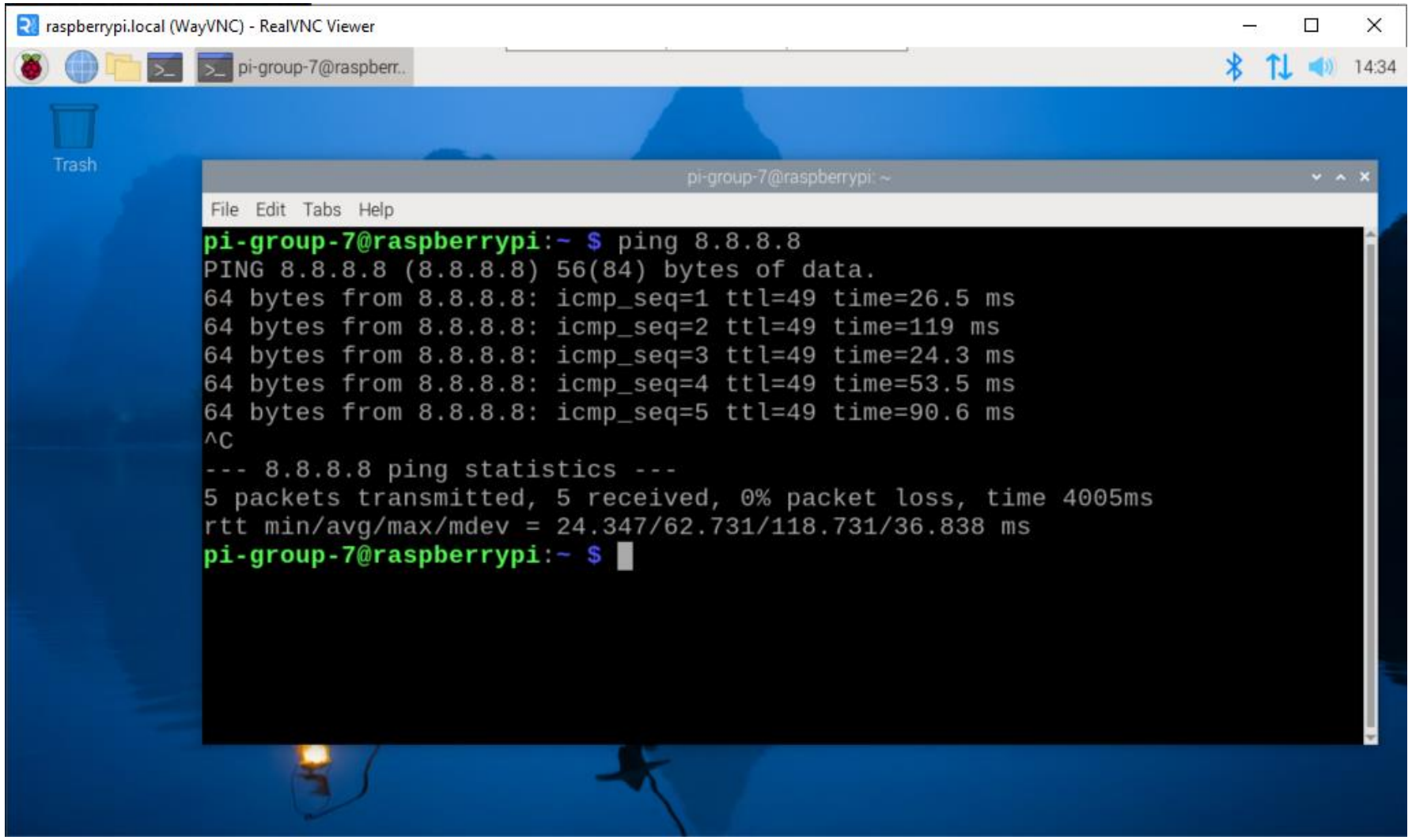
(Optional) Setup Network for Raspberry Pi

For those who connect LAN using LAN Cable only!

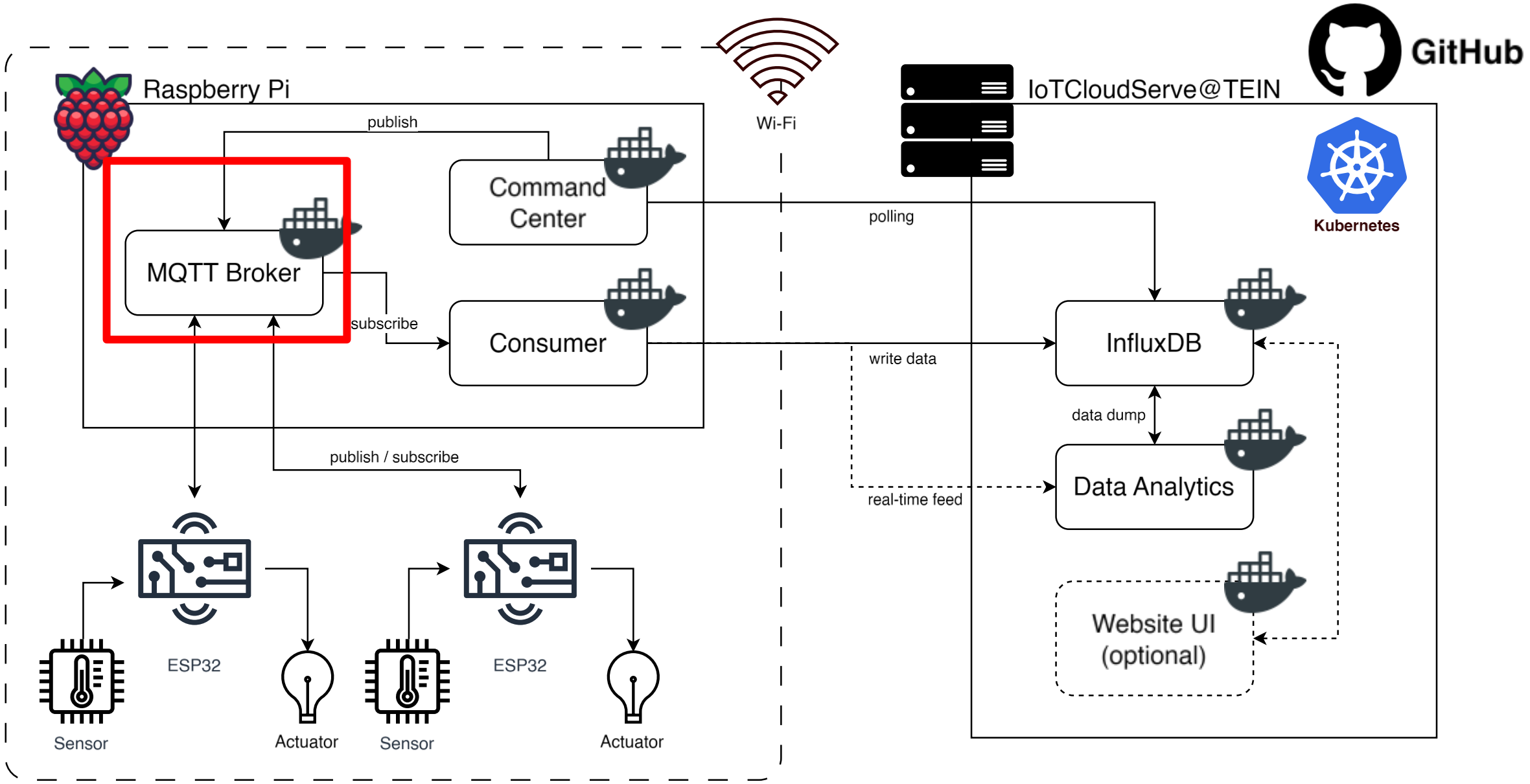
You allow an Ethernet Port to connect through Laptop Wi-Fi



Setup Network for Raspberry Pi



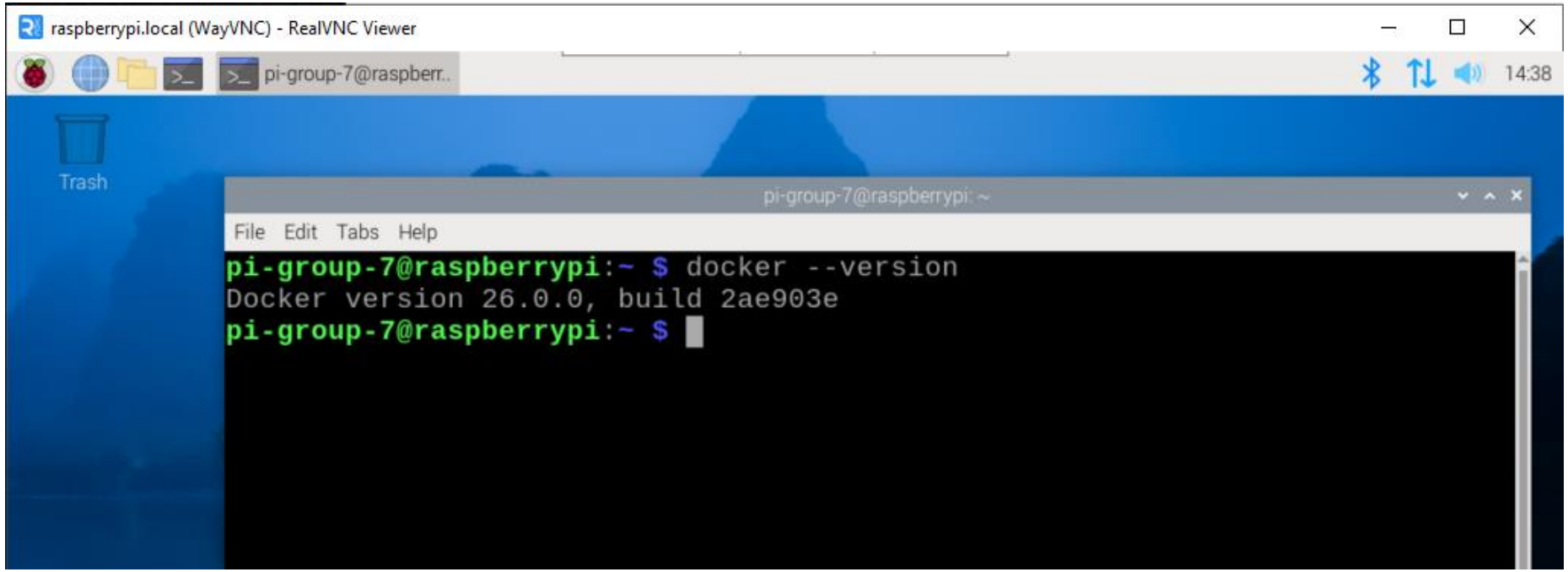
Pi now can connect to the internet


GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

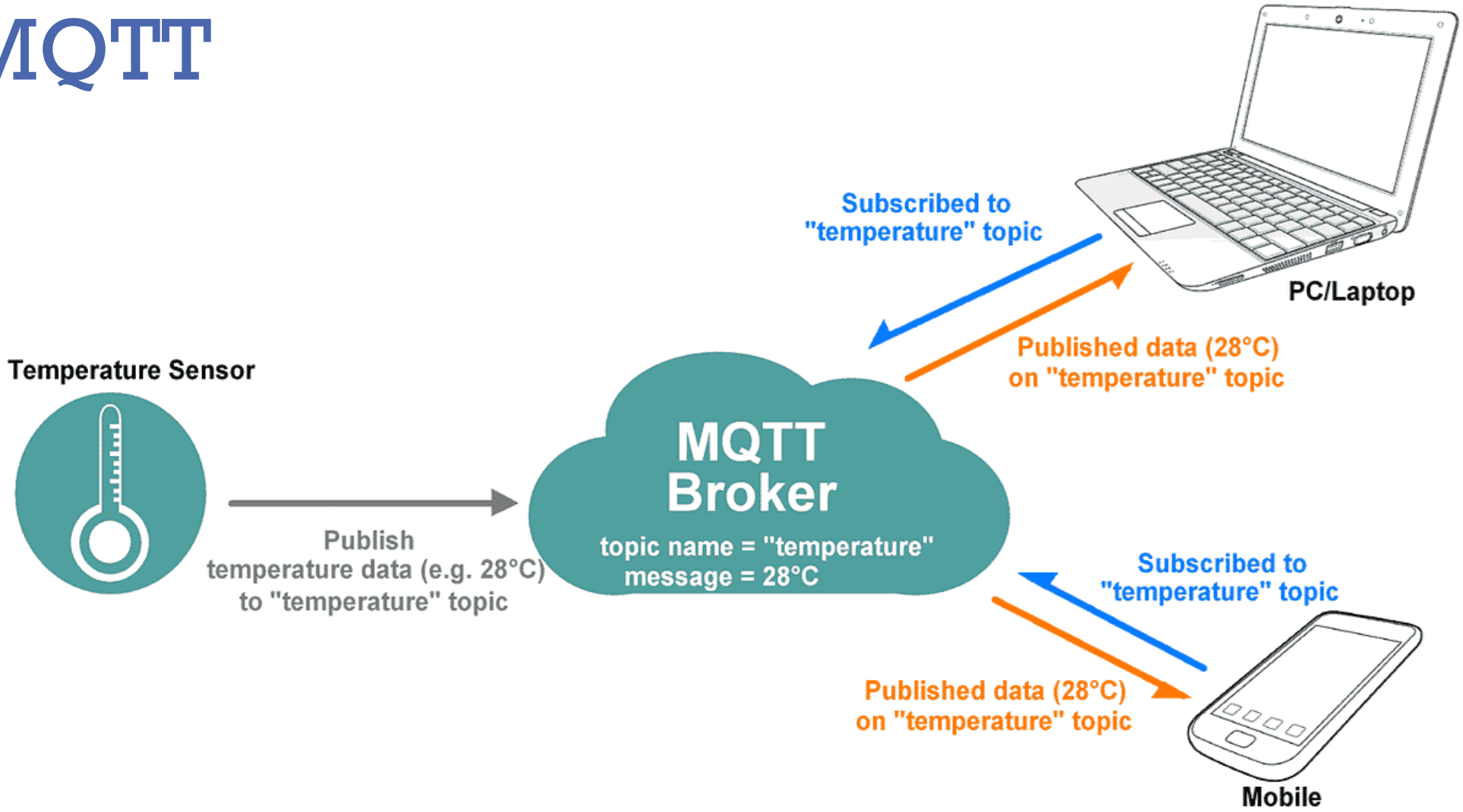
Install Docker on Raspberry Pi

https://raspberrytips.com/docker-on-raspberry-pi/

curl -sSL https://get.docker.com | sh
sudo usermod -aG docker $USER



MQTT



https://medium.com/@jaydev.dave93/what-is-mqtt-protocol-c6aocafffa8c

Deploy MQTT Broker

https://www.emqx.com/en

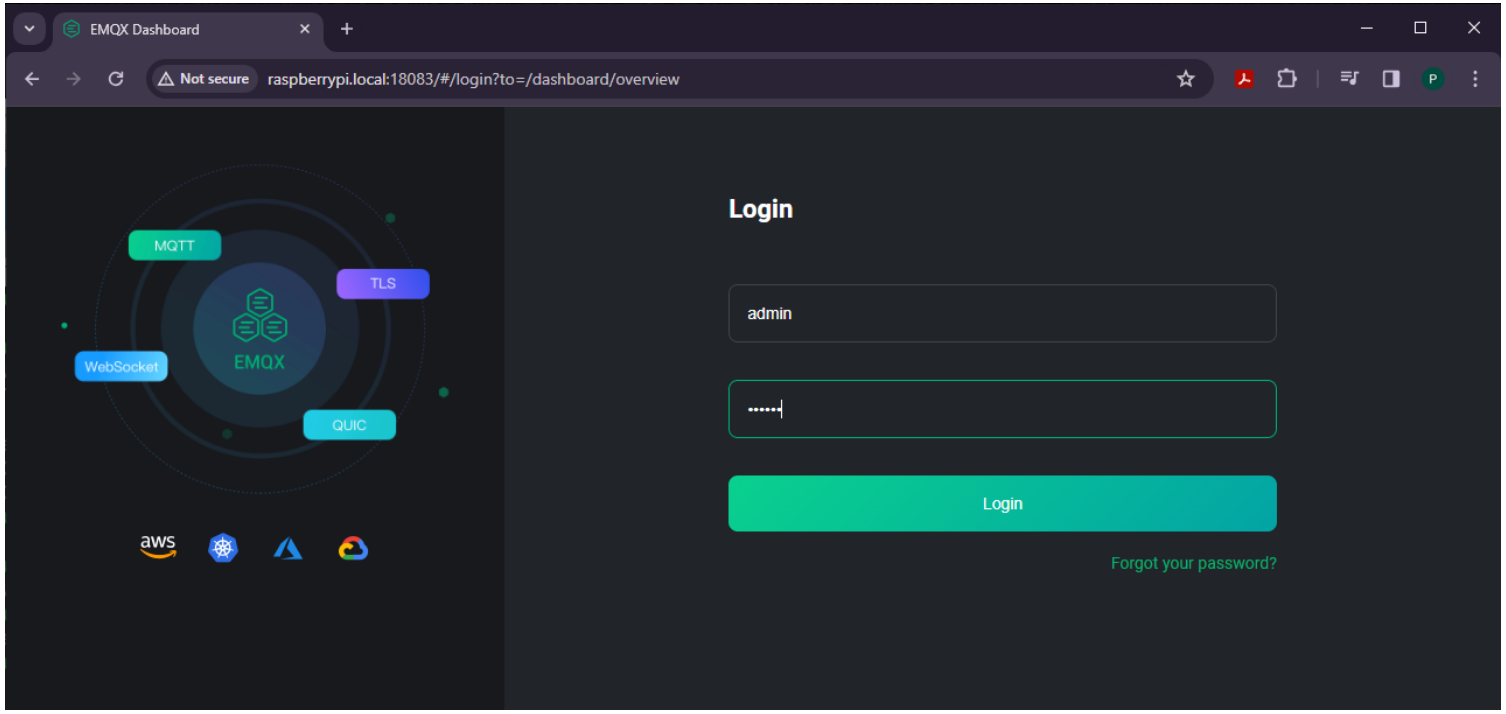
> sudo docker run –p 1883:1883 –p 18083:18083 emqx

We will use port 1883 for the MQTT connection and port 18083 for the MQTT Broker management

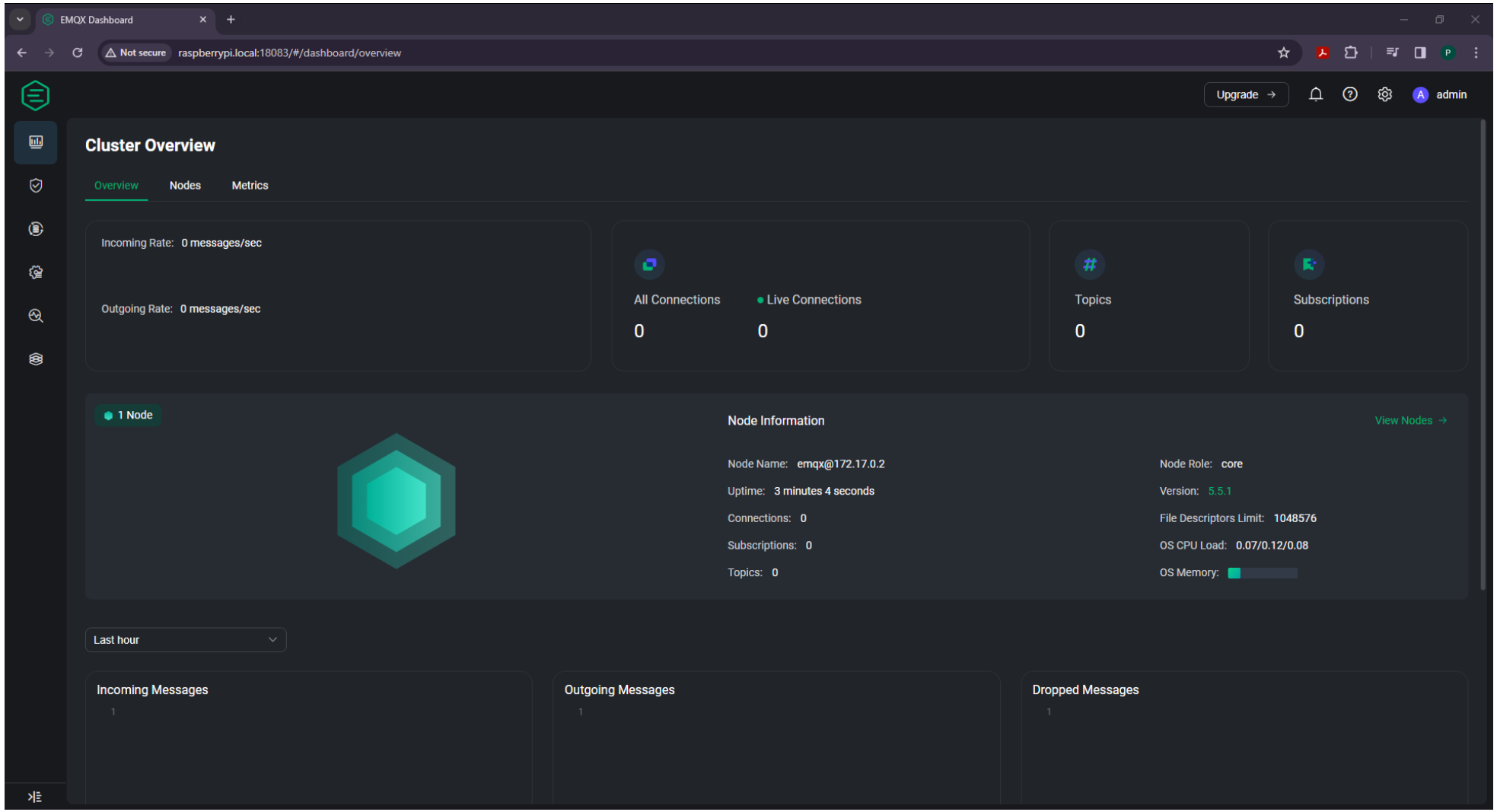
Deploy MQTT Broker

Go to “raspberrypi**1**.local:18083” and login

Username: admin Password: public



Deploy MQTT Broker

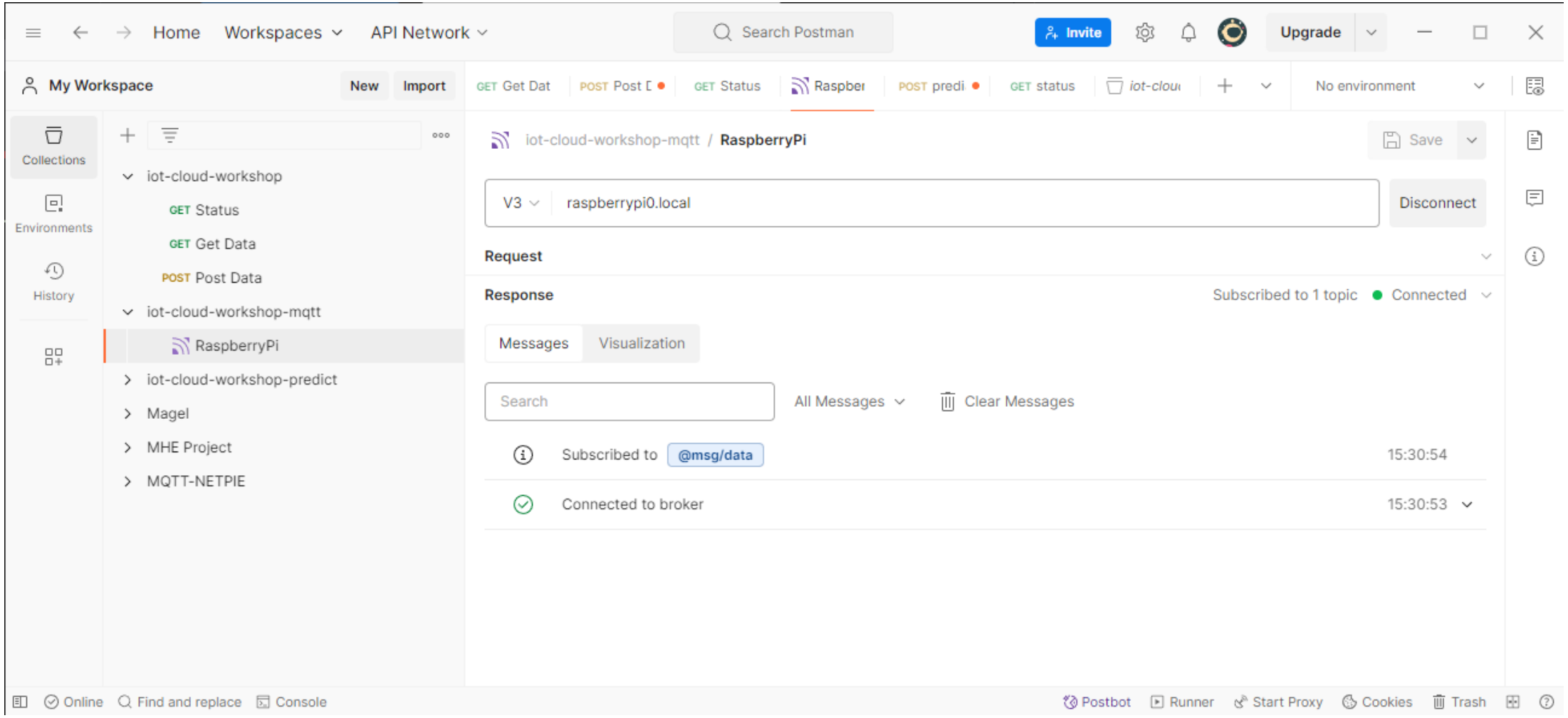


Test MQTT Broker using Postman

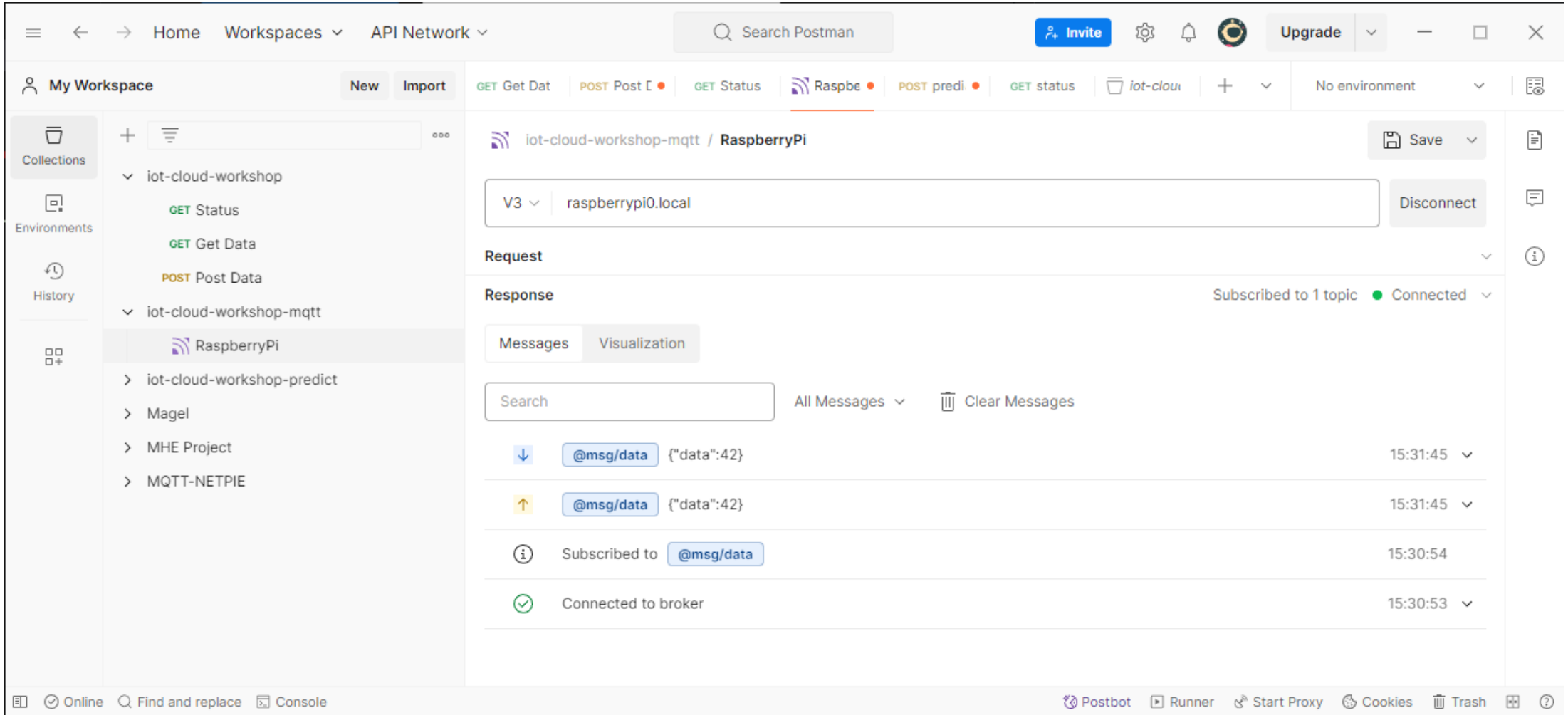
To verify the functionality of your broker, you can use Postman for testing.

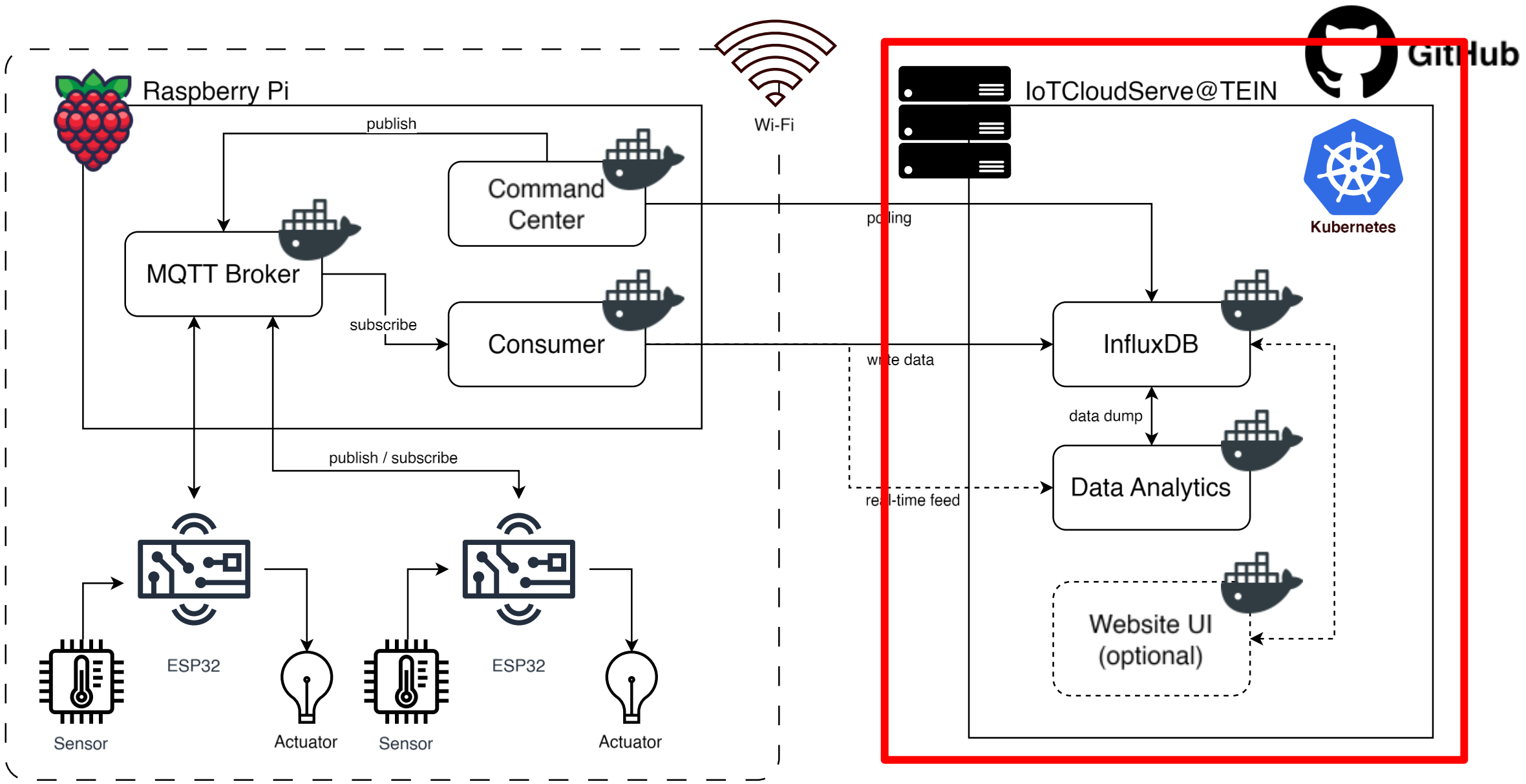
1. Connect to "raspberrypi**1**.local" at port 1883
2. Subscribe to the topic "@msg/data"
3. Try publishing a message on that topic and see the result

Test MQTT Broker using Postman



Test MQTT Broker using Postman




Raspberry Pi
publish
Command Center
MQTT Broker
subscribe
Consumer
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator
Wi-Fi
GitHub
IoTCloudServe@TEIN
Kubernetes
polling
write data
real-time feed
InfluxDB
data dump
Data Analytics
Website UI
(optional)

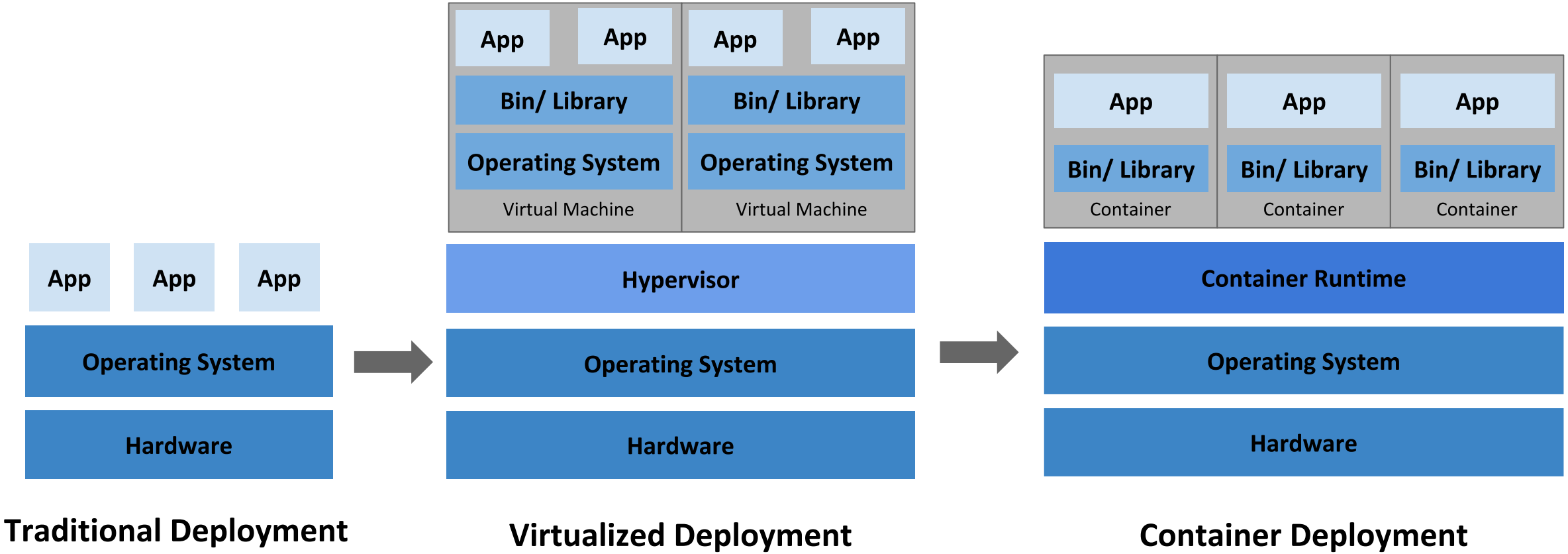
Kubernetes

https://kubernetes.io/docs/concepts/overview/

Kubernetes is a portable, extensible, open-source **platform for managing containerized workloads and services**, that facilitates both declarative configuration and automation.

Kubernetes

https://kubernetes.io/docs/concepts/overview/



Why you need Kubernetes?

https://kubernetes.io/docs/concepts/overview/

- **Service discovery and load balancing**
  - expose a container using the DNS name or using their IP address
  - distribute the network traffic so that the deployment is stable
- **Automated rollouts and rollbacks**
  - create new containers for your deployment, remove existing containers and adopt all their resources to the new container

Why you need Kubernetes?

https://kubernetes.io/docs/concepts/overview/

- **Self-healing**
  - restarts containers that fail
  - replaces containers
  - kills containers that don't respond to the health check
- **Secret and configuration management**
  - store and manage sensitive information, such as passwords or keys
  - You can deploy and update secrets and application configuration without rebuilding your container images

Rancher

https://rancher manager.docs.rancher.com/v2.0-v2.4/getting-started/introduction/overview

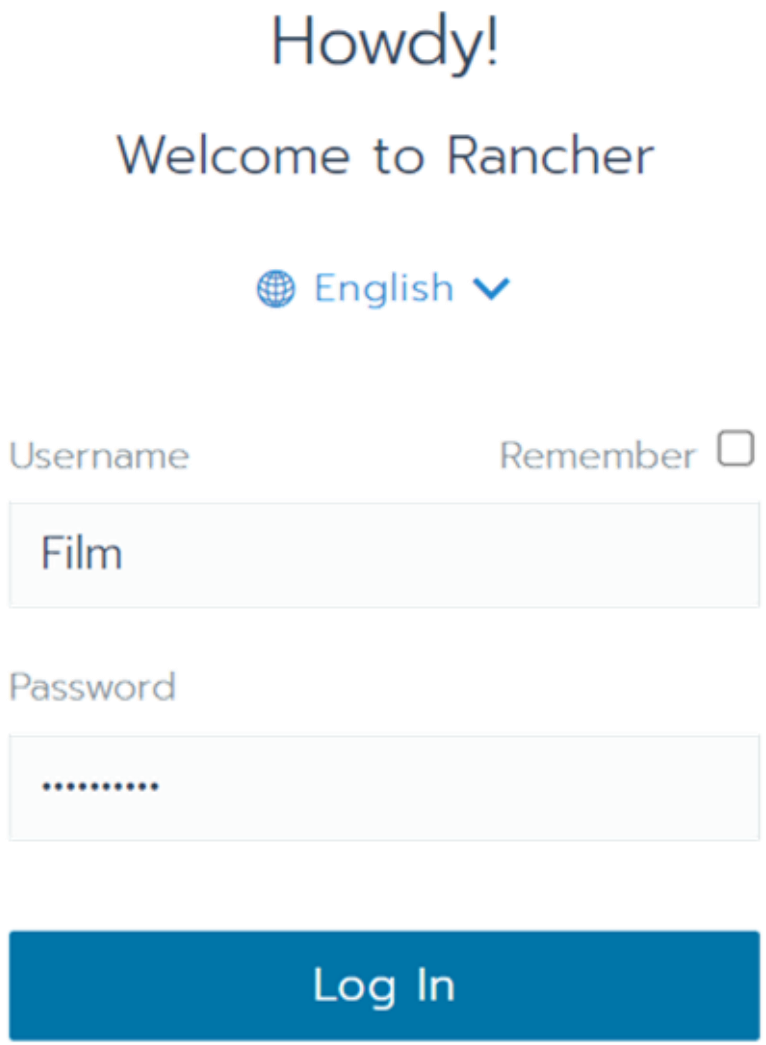
Rancher is a container management platform built for organizations that deploy containers in production. Rancher makes it easy to run Kubernetes everywhere

Rancher

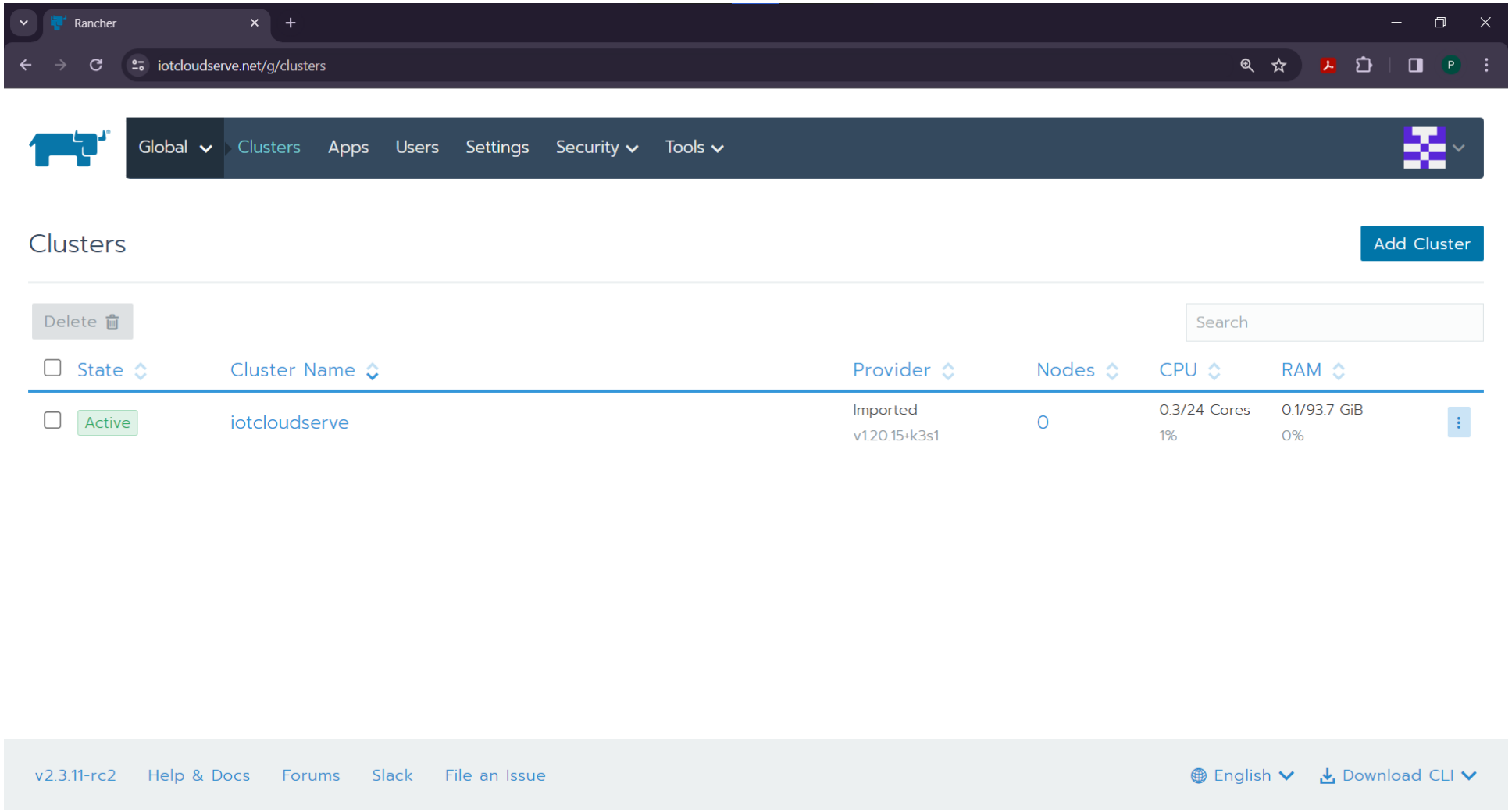
Go to https://www.iotcloudserve.net/login

Username: iot-group-**1**

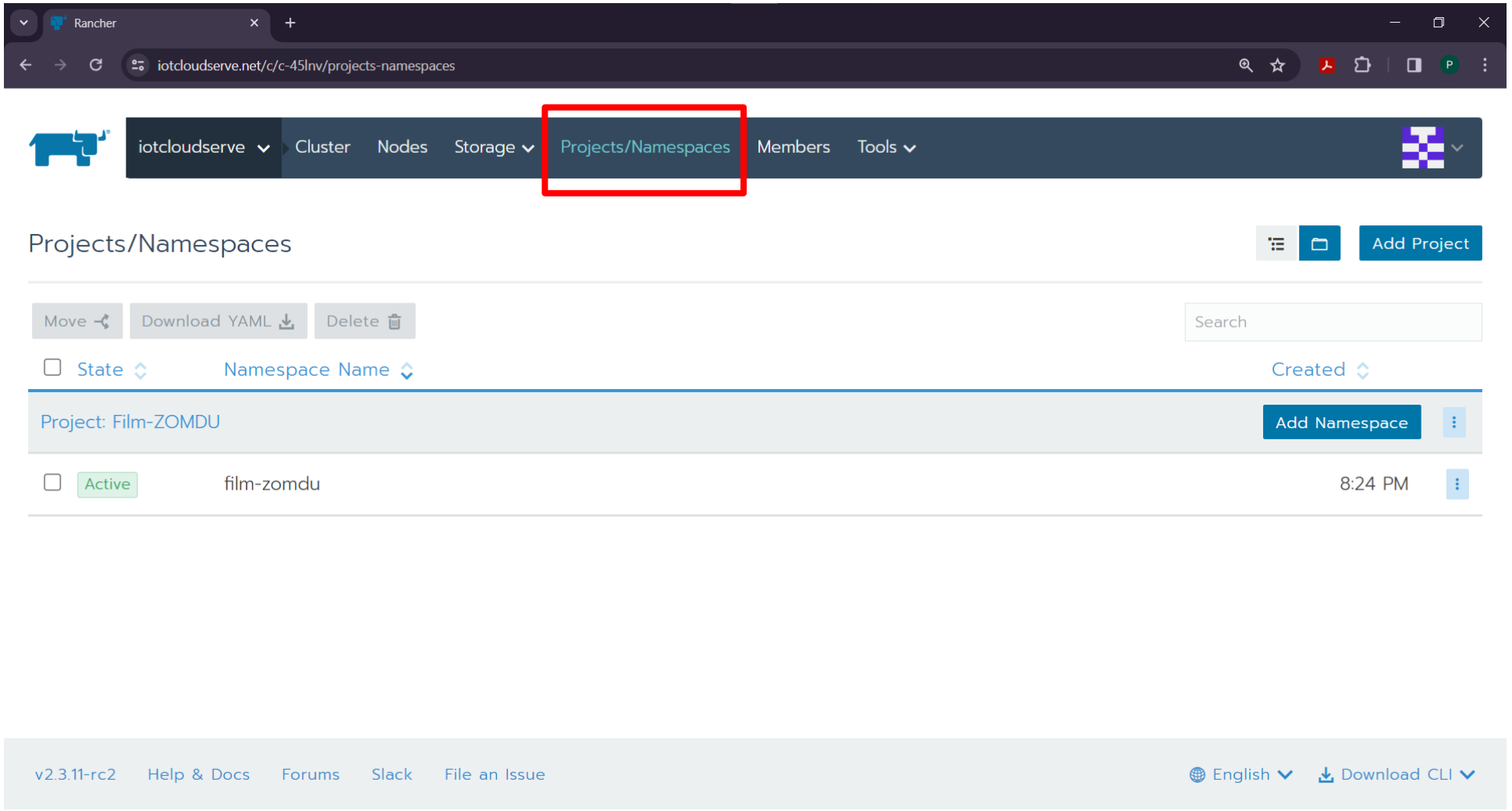
Password: [Student ID of Edge & Cloud operators]
e.g. 6230361921



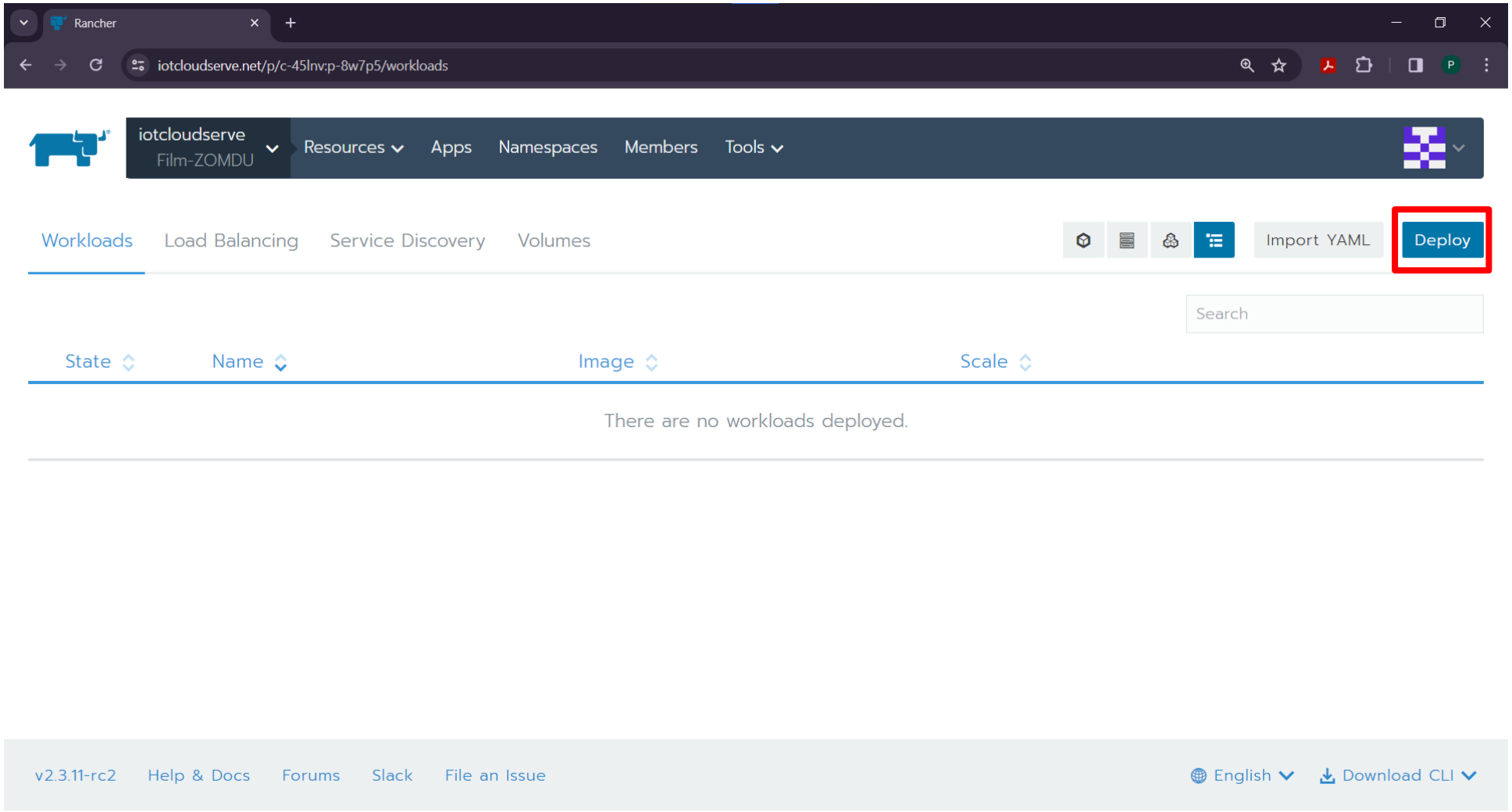
Rancher


Rancher
iotcloudserve.net/g/clusters
Global
Clusters
Apps
Users
Settings
Security
Tools
Clusters
Add Cluster
Delete
Search
State
Cluster Name
Provider
Nodes
CPU
RAM
Active
iotcloudserve
Imported
v1.20.15+k3s1
0
0.3/24 Cores
1%
0.1/93.7 GiB
0%
v2.3.11-rc2
Help & Docs
Forums
Slack
File an Issue
English
Download CLI

Rancher

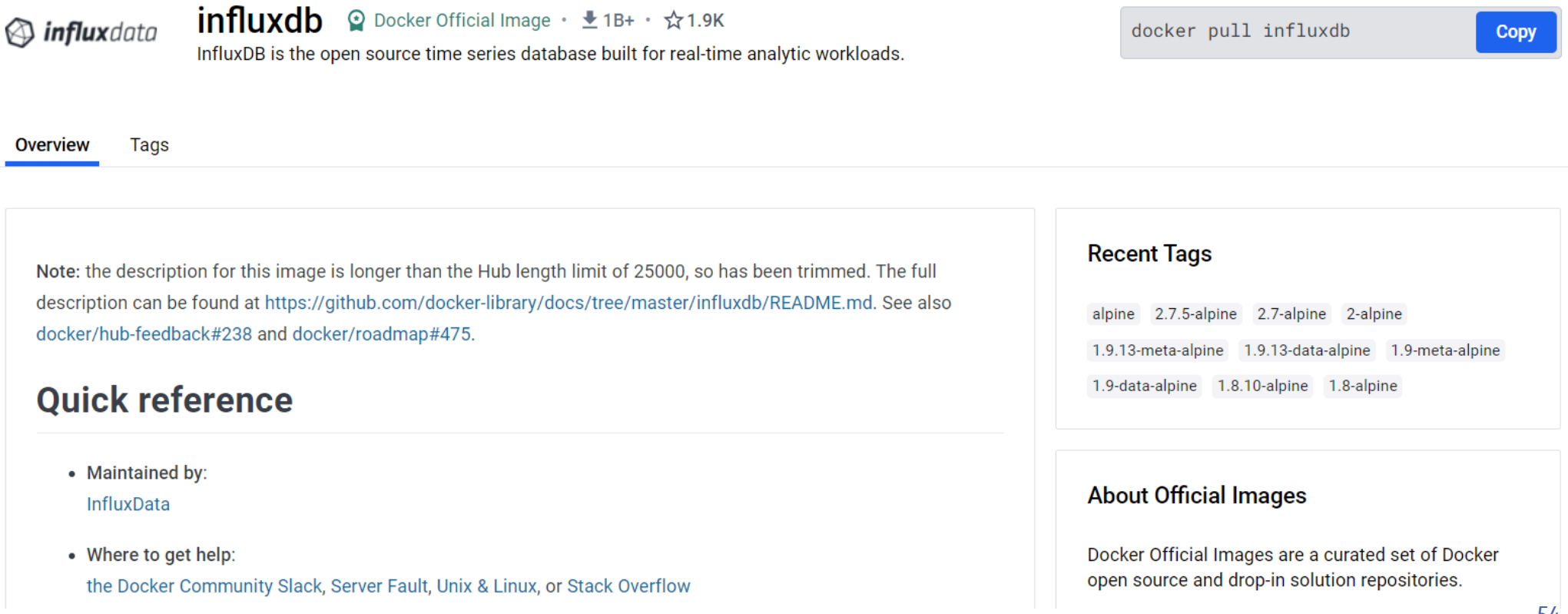

Rancher
iotcloudserve.net/c/c-45lnv/projects-namespaces
iotcloudserve
Cluster
Nodes
Storage
Projects/Namespaces
Members
Tools
Projects/Namespaces
Add Project
Move
Download YAML
Delete
Search
State
Namespace Name
Created
Project: Film-ZOMDU
Add Namespace
Active
film-zomdu
8:24 PM
v2.3.11-rc2
Help & Docs
Forums
Slack
File an Issue
English
Download CLI

Rancher – Deploy InfluxDB

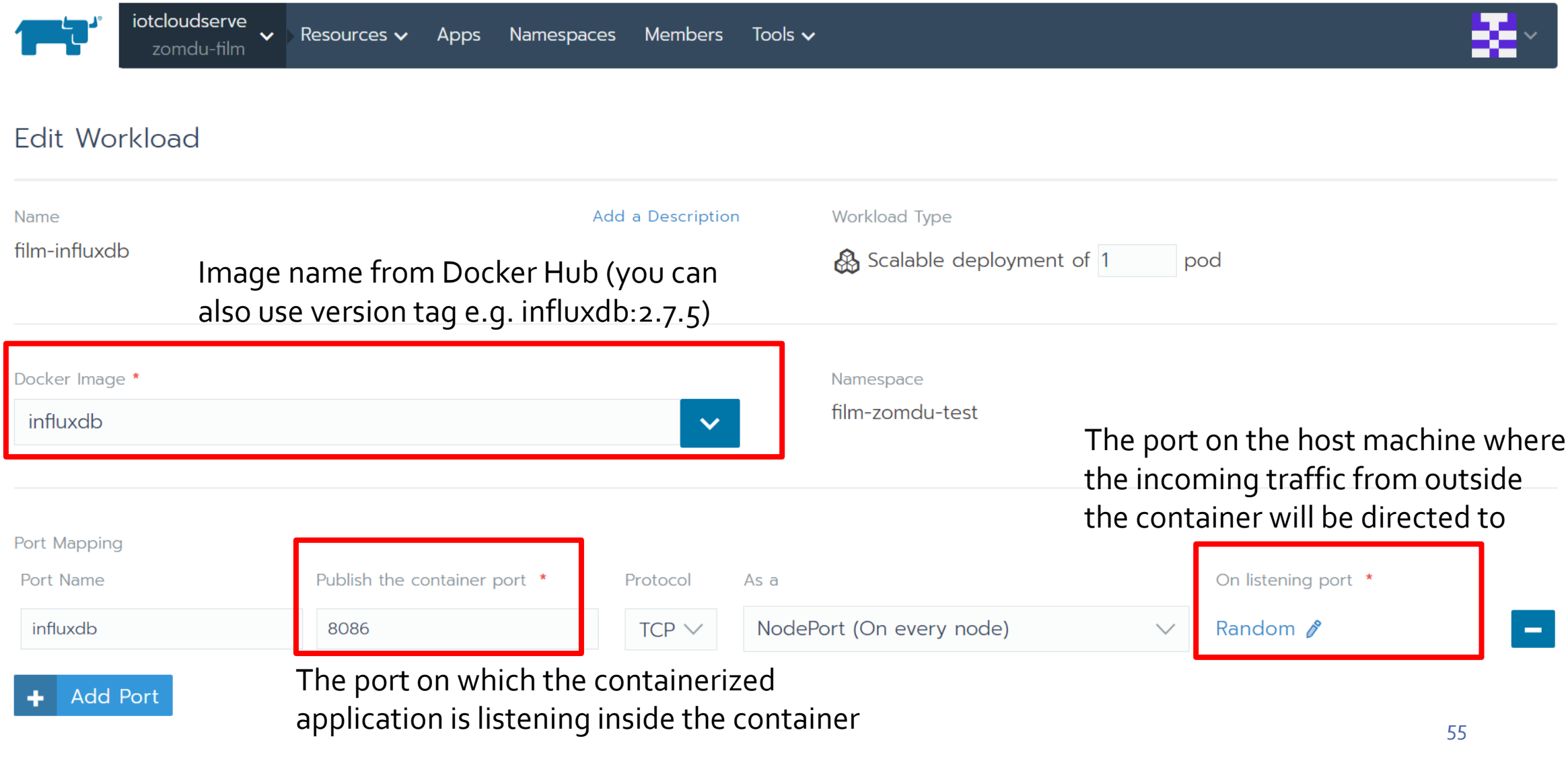


Rancher – Deploy InfluxDB

Docker image: https://hub.docker.com/_/influxdb



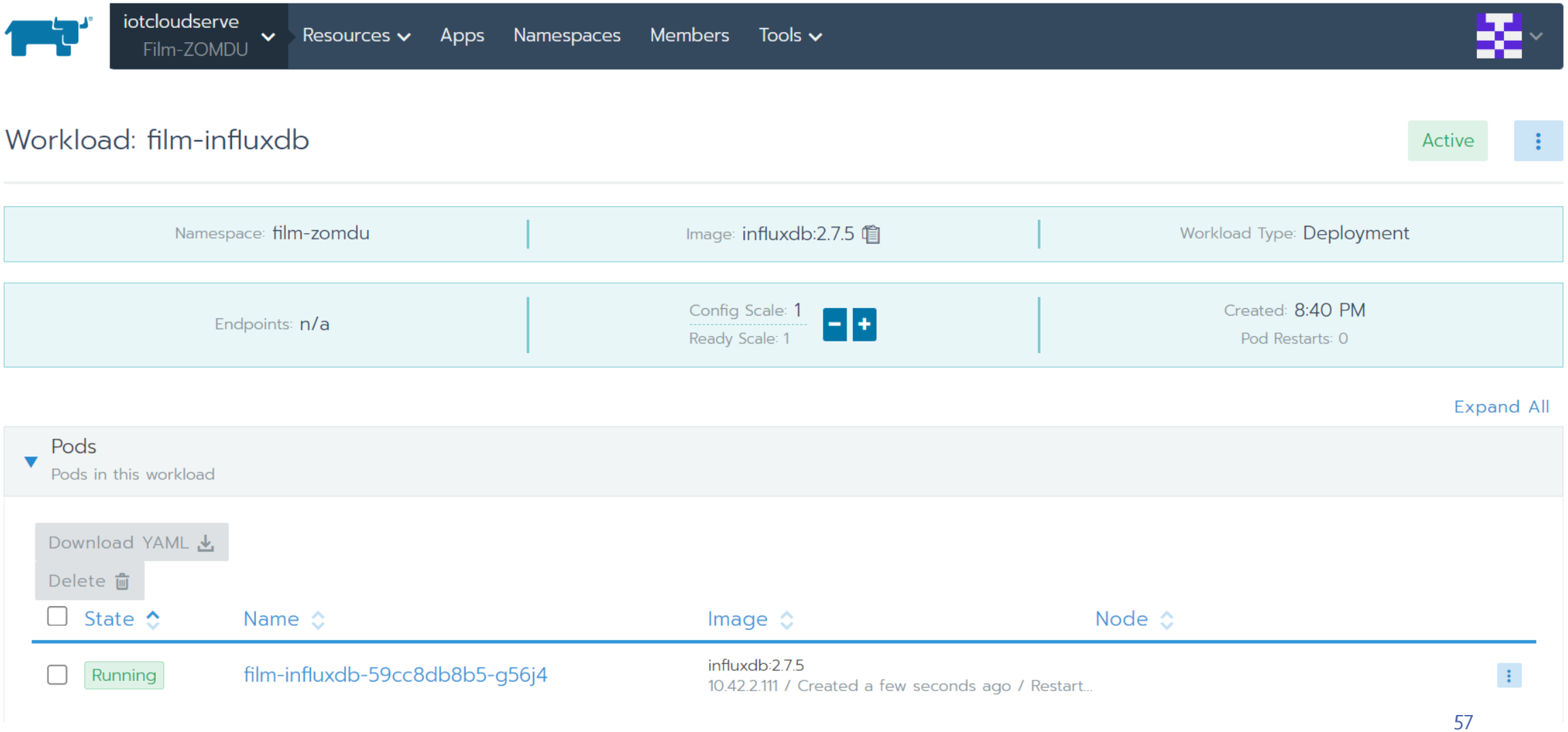
Rancher – Deploy InfluxDB



Rancher – Deploy InfluxDB

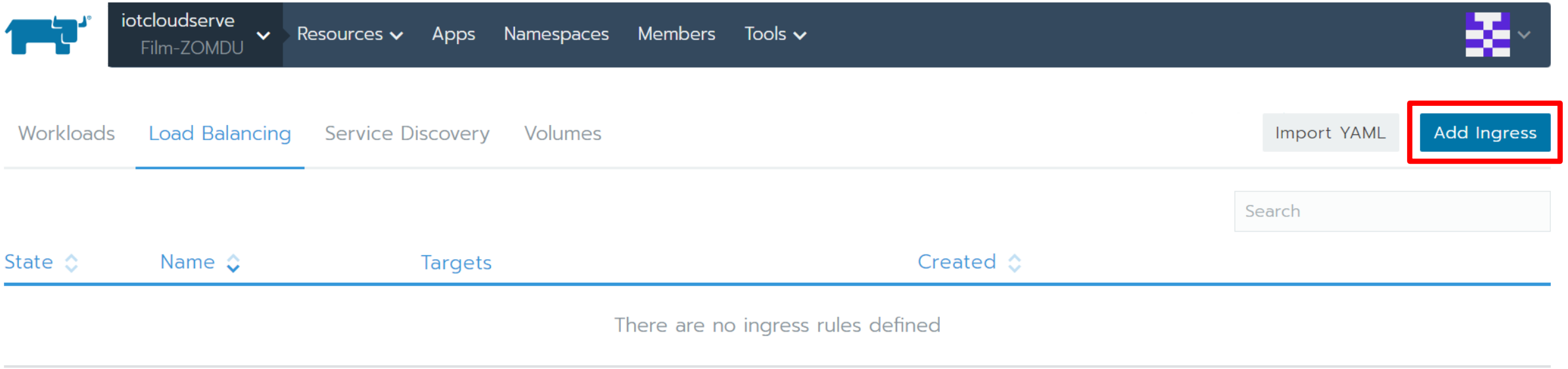
Each user is allocated 5 CPUs and 5 GB of memory. The reservation and limit for memory and CPU of each workload can be adjusted based on your application's requirements, but the total allocation must not exceed this quota.

Rancher – Deploy InfluxDB



Rancher – Deploy InfluxDB

You need to assign a domain name for the workload to access it.



Rancher – Deploy InfluxDB

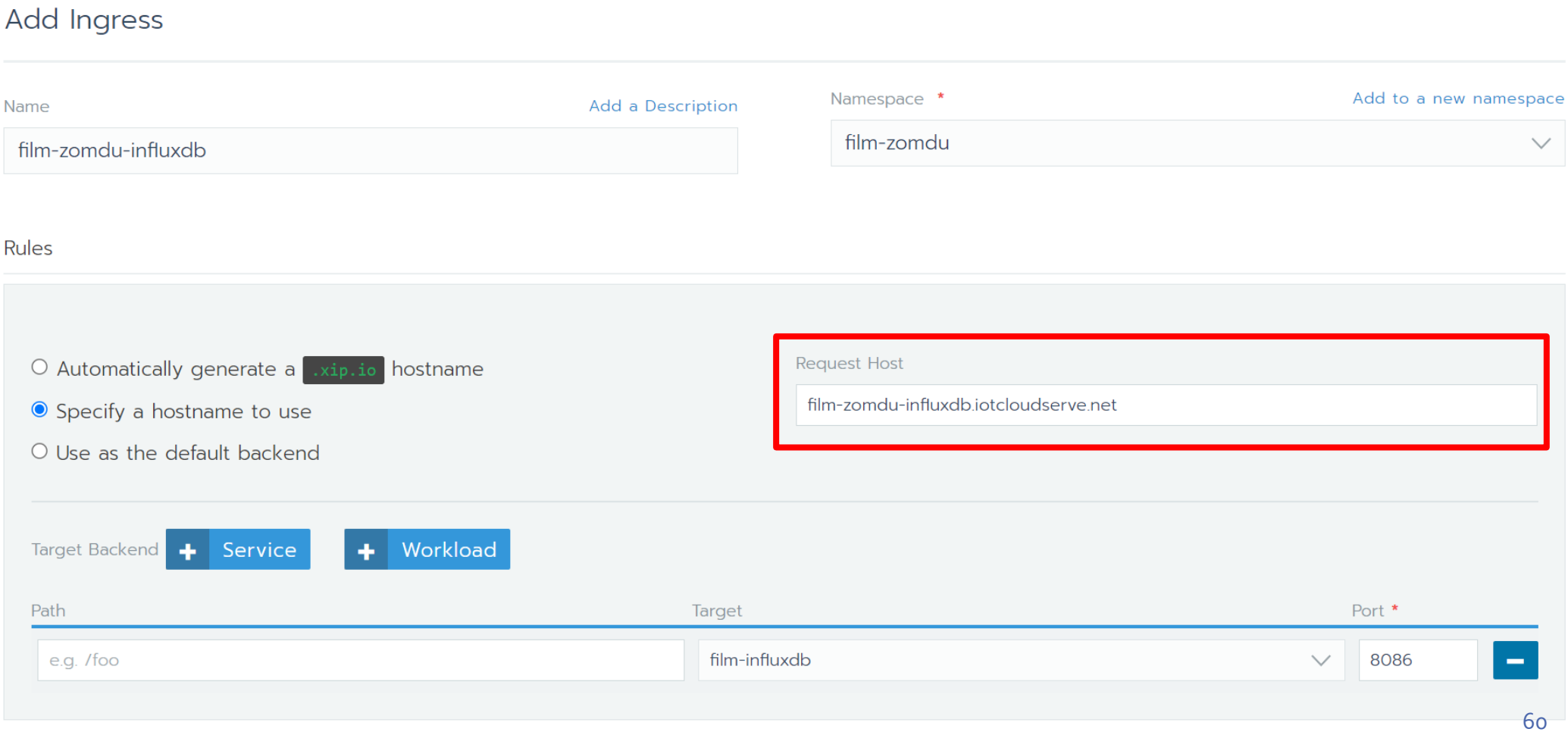
Domain Name list for each group

- iot-group**1**-service1.iotcloudserve.net
- iot-group**1**-service2.iotcloudserve.net
- iot-group**1**-service3.iotcloudserve.net

You will get three names for each group.
If extra names are required, please contact Film.

Rancher – Deploy InfluxDB

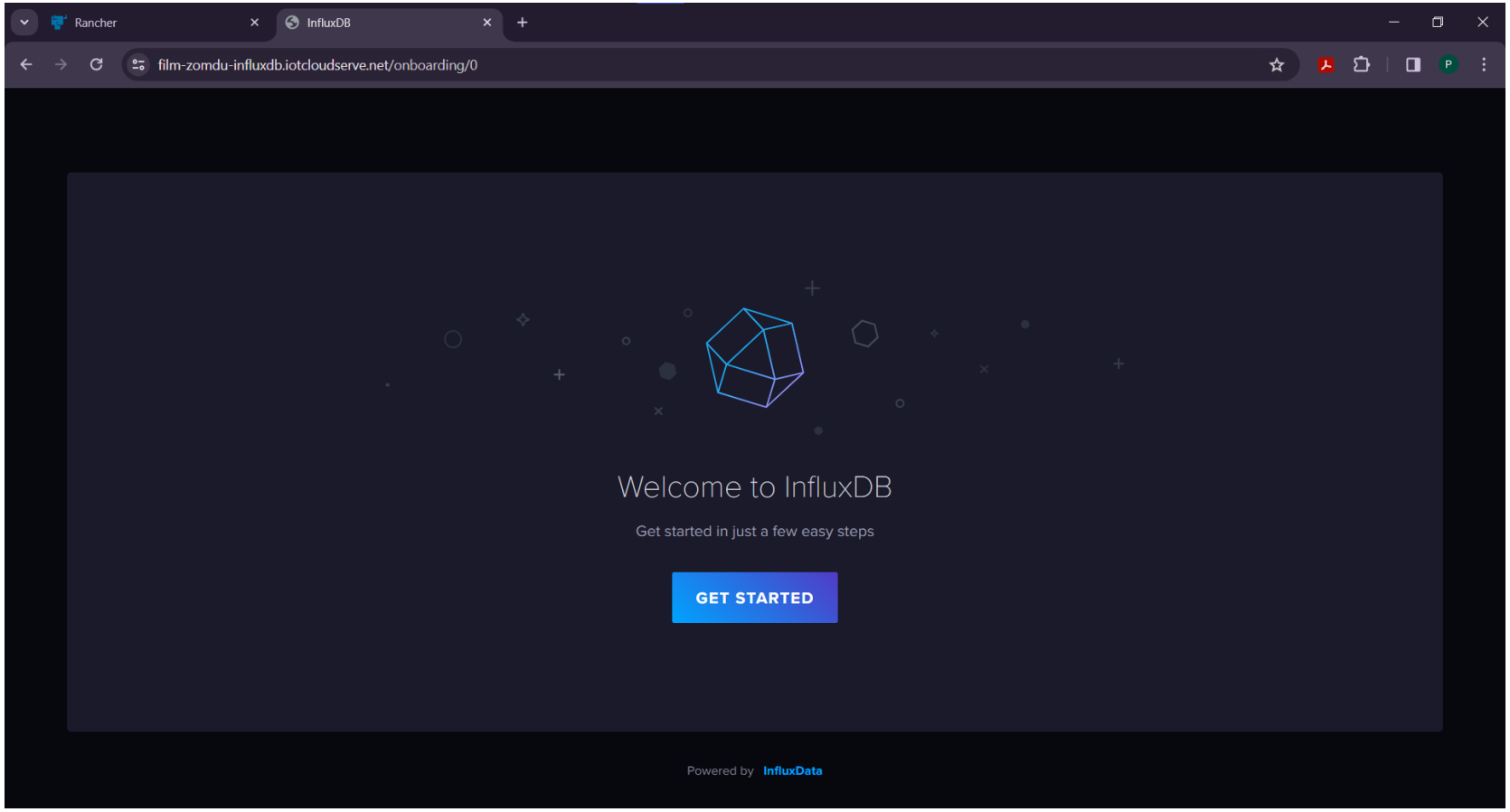
Hostname for InfluxDB: iot-group**1**-service1.iotcloudserve.net



InfluxDB

Go to iot-group**1**-service1.iotcloudserve.net

You should see the InfluxDB UI



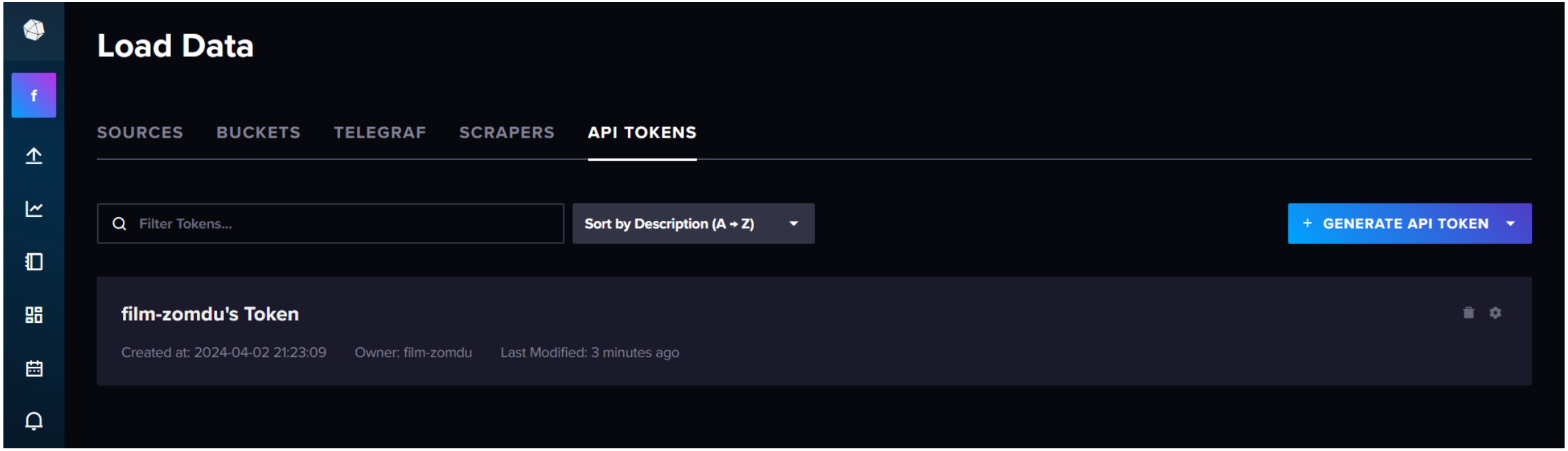
InfluxDB

Configure a user account for InfluxDB. Ensure to securely store the credentials as they will be required for accessing InfluxDB.

INFLUXDB_USERNAME
INFLUXDB_PASSWORD
INFLUXDB_ORG
INFLUXDB_BUCKET
INFLUXDB_TOKEN
INFLUXDB_URL

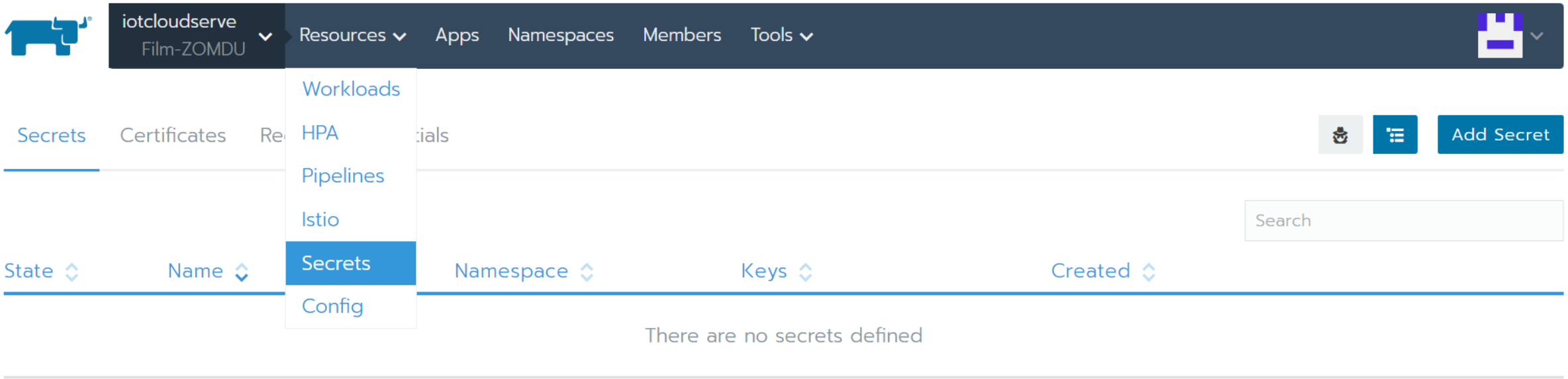
Rancher – Deploy InfluxDB

If you forget to copy the token, you can generate the token here

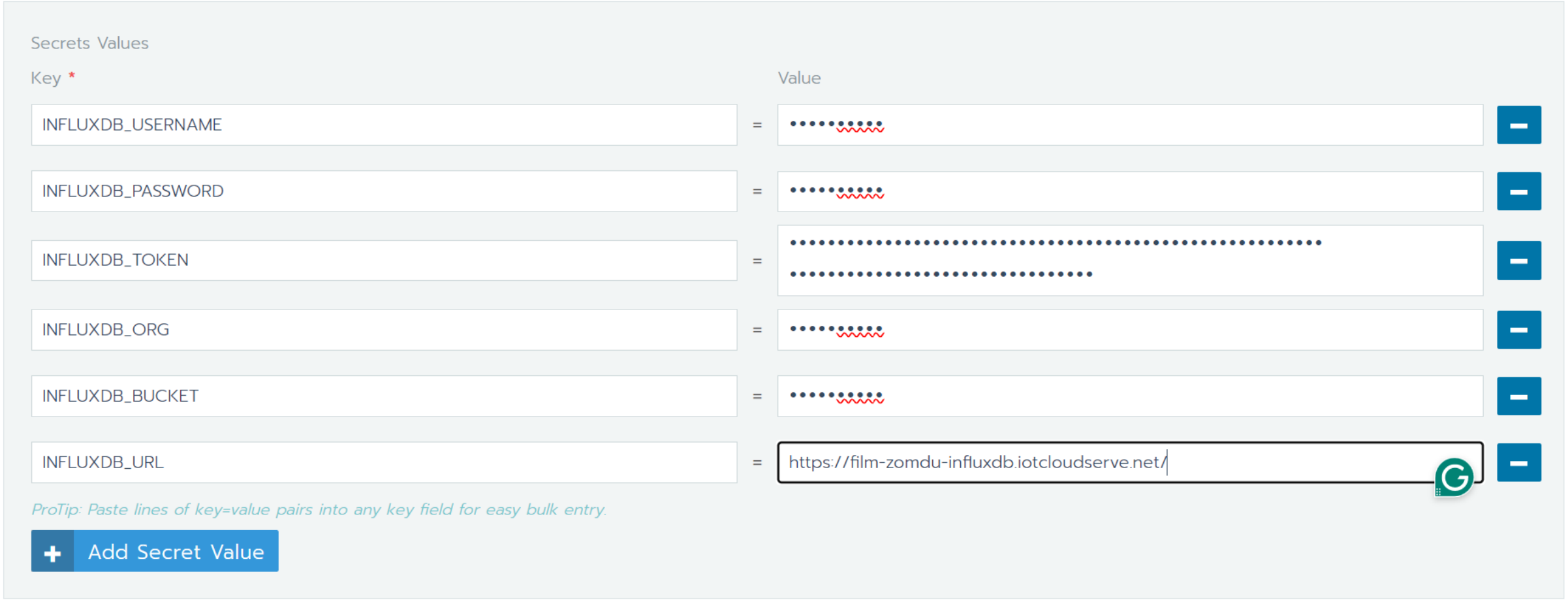


Rancher – Deploy InfluxDB

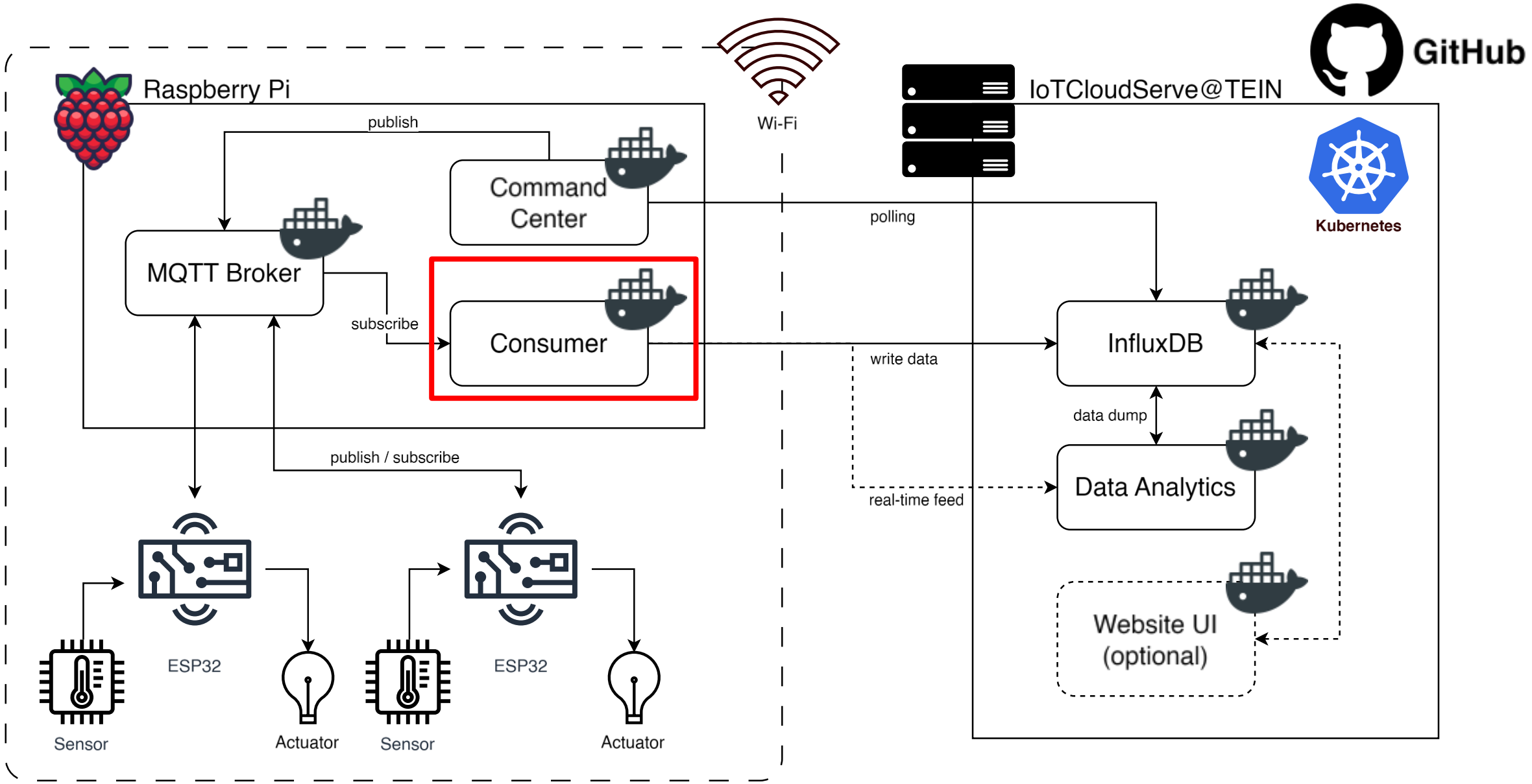
Return to Rancher and store the credentials for InfluxDB in the "secrets" feature for secure access management.



Rancher – Deploy InfluxDB



Your InfluxDB credentials are now stored in the Rancher.


GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Run Consumer on Raspberry Pi

The **consumer** receives the published data from the subscribed topic and puts the data into InfluxDB

Inside Raspberry Pi

git clone https://github.com/PatchapongKul/IoT-Final-Project.git

Look for Reference_Code > RaspberryPi > Consumer

Build your docker image

sudo docker build -t [image-name]:[tag] .

Run Consumer on Raspberry Pi

If you want to test the Python script inside Raspberry Pi, you can use the following commands to activate a virtual environment

> cd ~/IoT-Final-Project/Training-code/RaspberryPi
> source pienv/bin/activate

> python [file name].py

Run Consumer on Raspberry Pi

Make sure you change the ".env" file before running the image
> sudo nano .env

Kindly include all your credentials there

To run the docker image

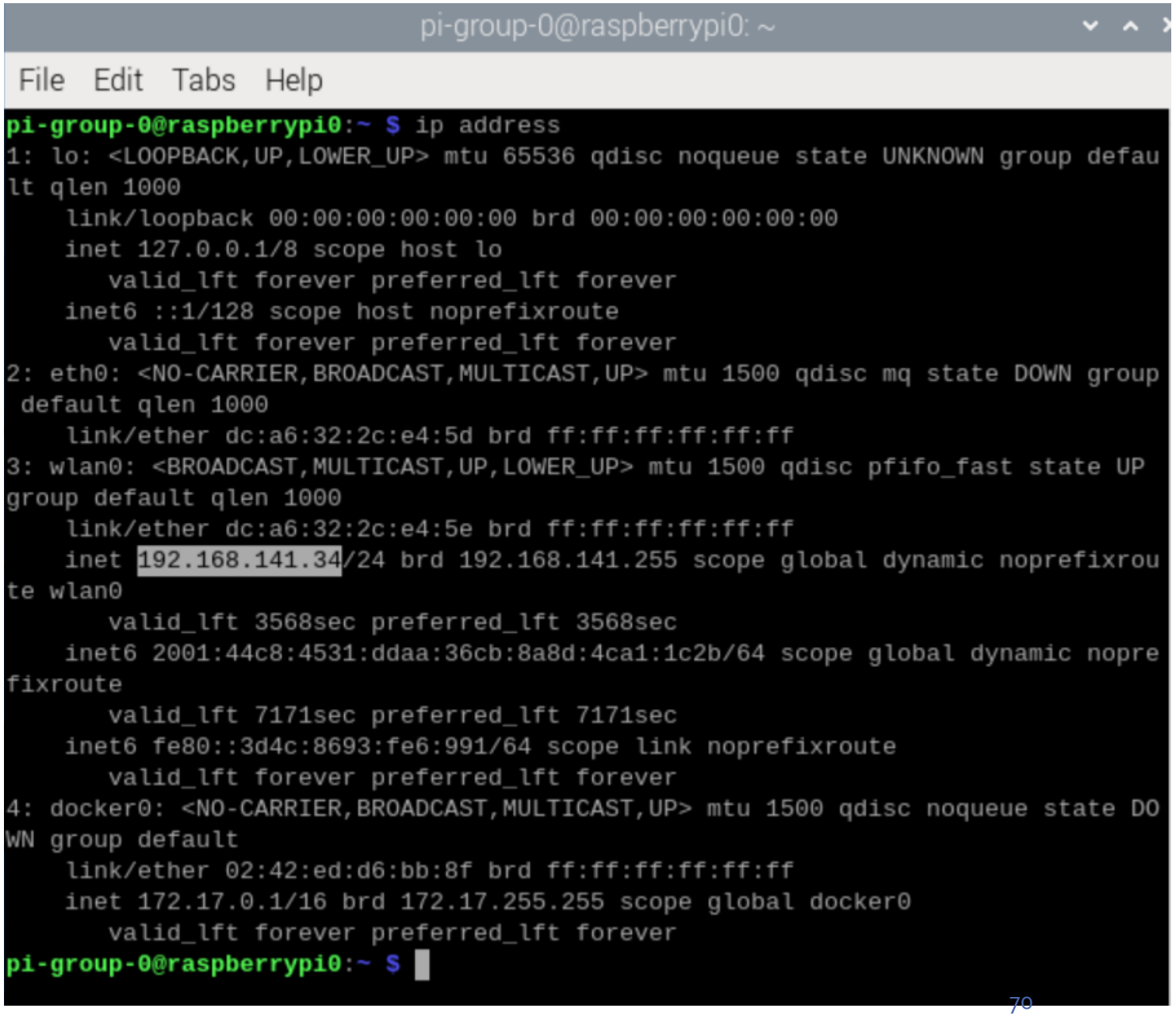
> docker images – to see list of images
> docker run --env-file=.env [image name]

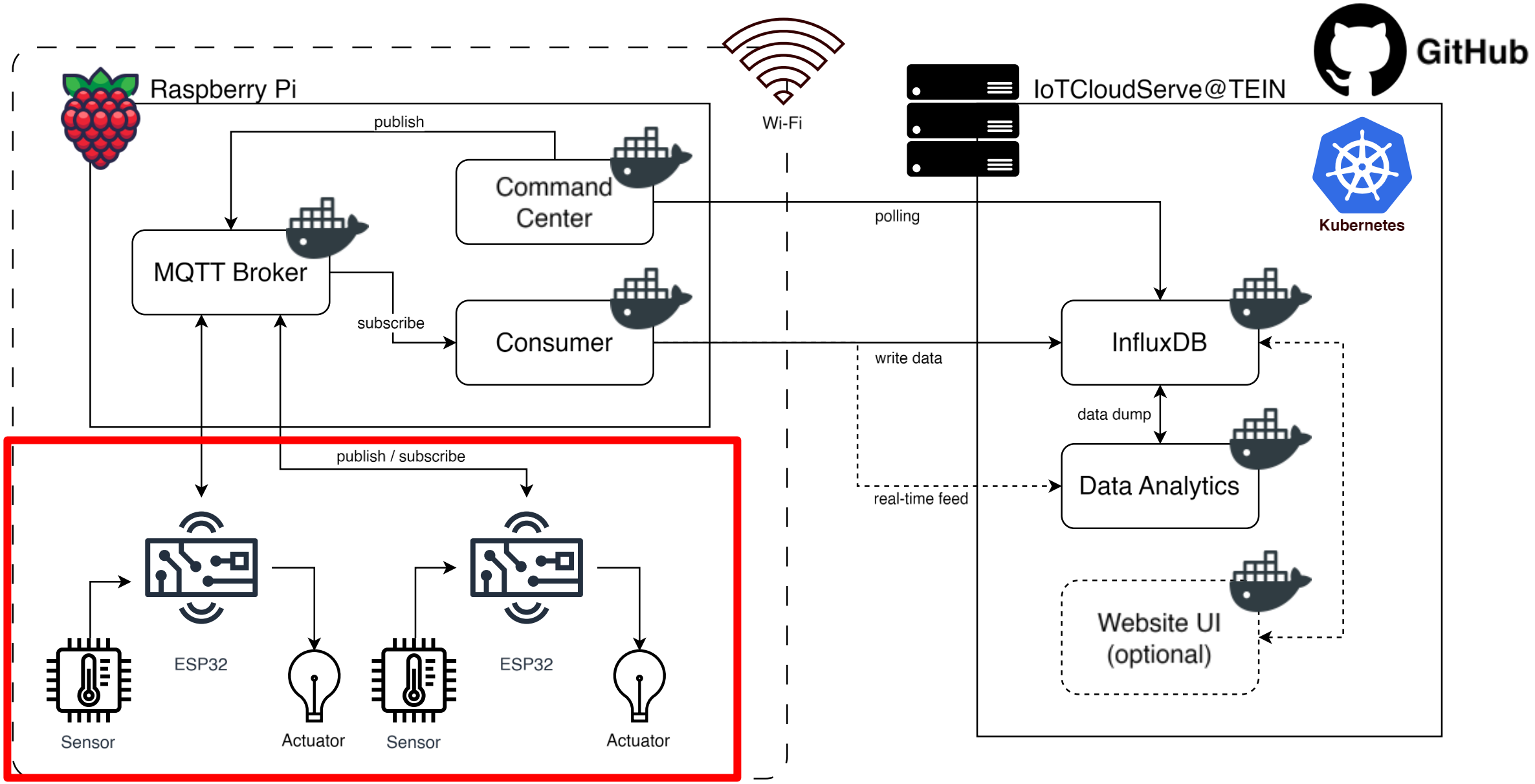
Run Consumer on Raspberry Pi

To find your IP address

> ip address
or
> ifconfig

Find network interface
eth: Ethernet
wlan: Wireless LAN




GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Run Producer on LAPTOP

The **producer** publishes the data from the dataset available at https://www.kaggle.com/datasets/ananthr1/room-occupancy-estimation-data-set/data

The dataset contains ambient room data, including temperature, sound, light, and PIR sensors. We aim to predict the number of people inside the room (0 to 3 people) using this data.

On your Laptop

git clone https://github.com/PatchapongKul/IoT-Final-Project.git

Run Producer on LAPTOP

The **producer** publishes the data from the dataset available at https://www.kaggle.com/datasets/ananthr1/room-occupancy-estimation-data-set/data

Activate virtual environment

> cd Training-code/

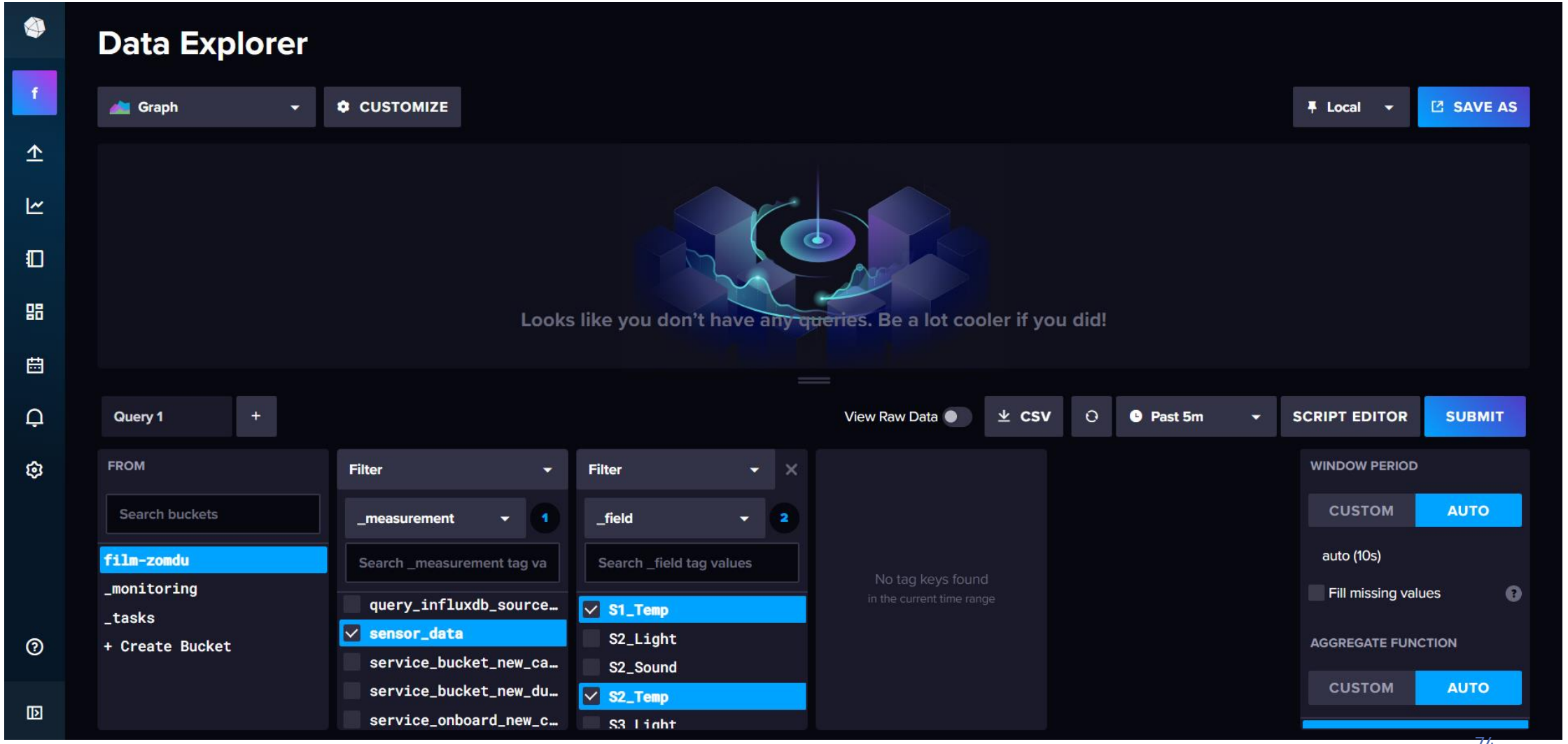
> .\venv\Script\activate

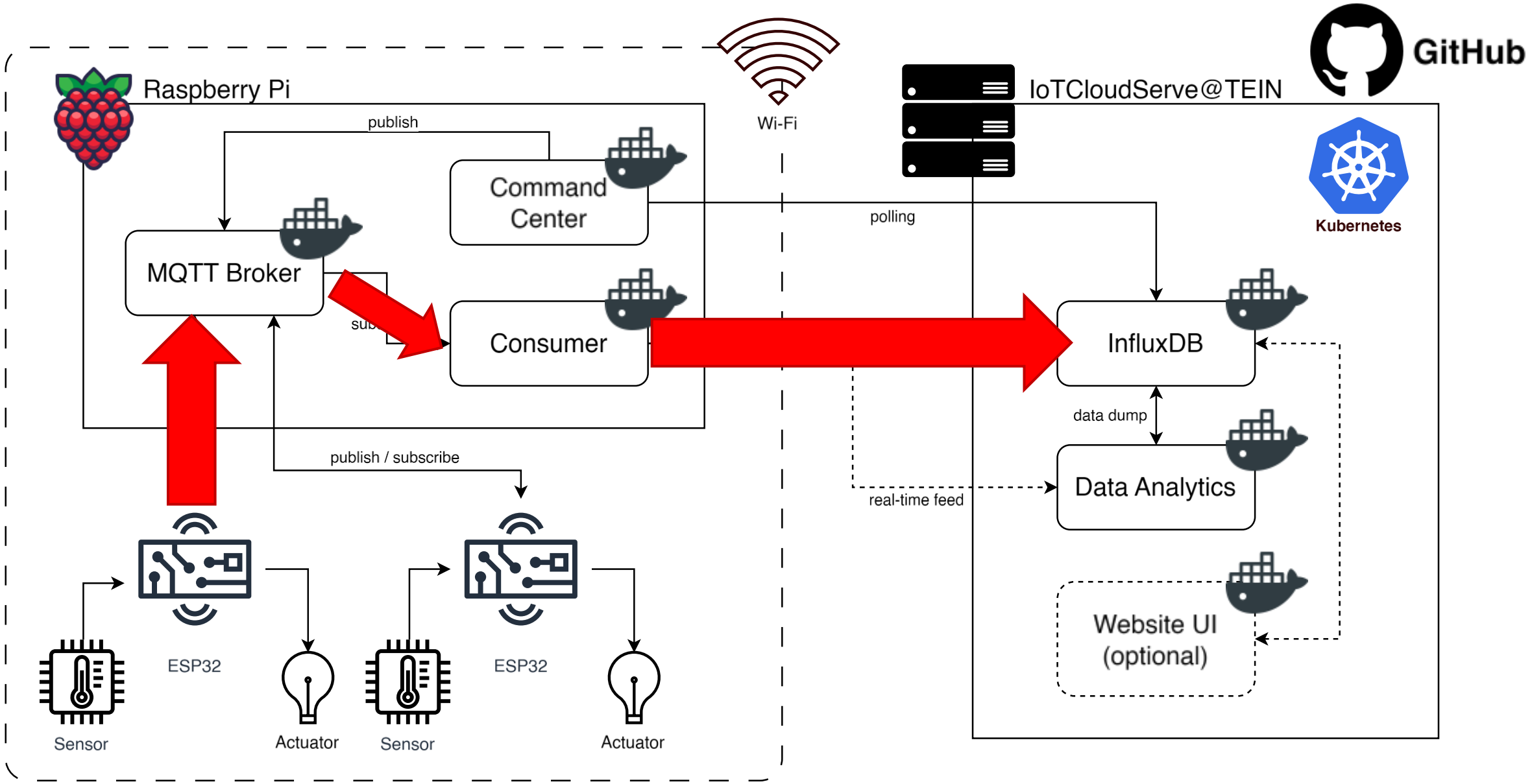
Run Python Script

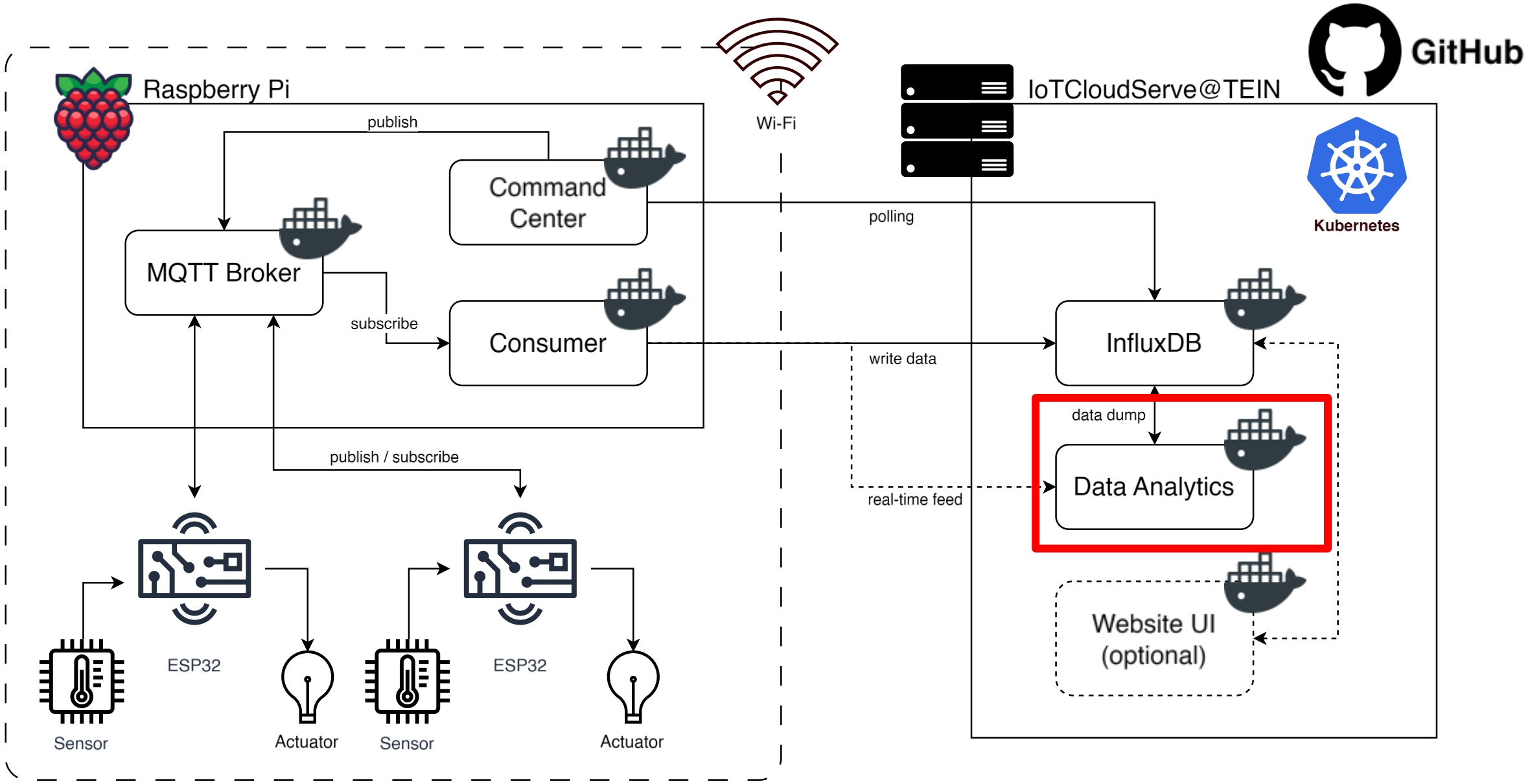
> cd [your directory]

> python Producer.py

Check the result in InfluxDB




GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
polling
MQTT Broker
Consumer
InfluxDB
data dump
publish / subscribe
Data Analytics
real-time feed
Website UI (optional)
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator


GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Train KNN Model on LAPTOP

Within the limited time, we will apply the analysis from the Kaggle to our work.
Ref: https://www.kaggle.com/code/vivekaryan/room-occupancy-estimation-with-variable-selection

Look inside Training-code> IoTCloudServe@TEIN > ModelTraining

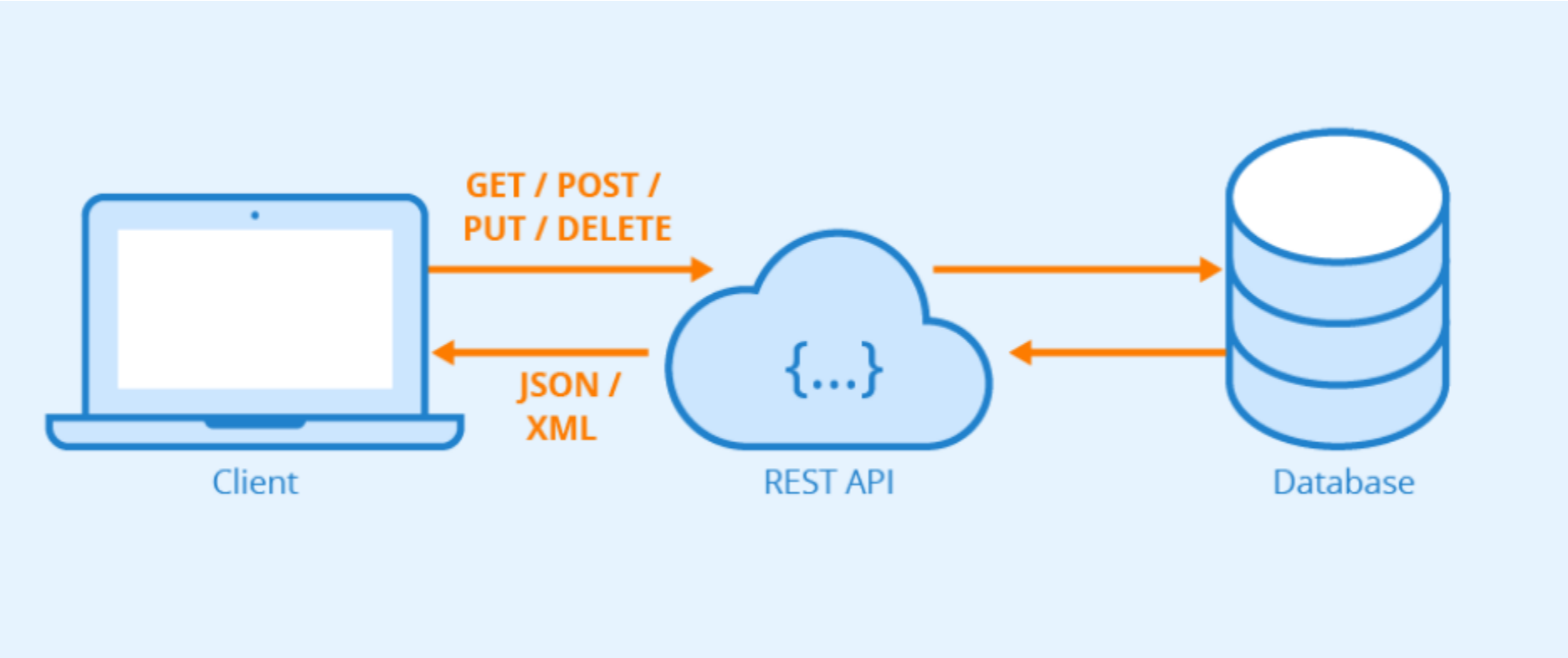
Deploy Model on IoTCloudServe@TEIN

We will develop a REST API server to serve as the endpoint for consumers to send data, enabling our server to predict the occupancy of a room based on the received information.

Look inside Training-code > IoTCloudServe@TEIN > RealTimePrediction

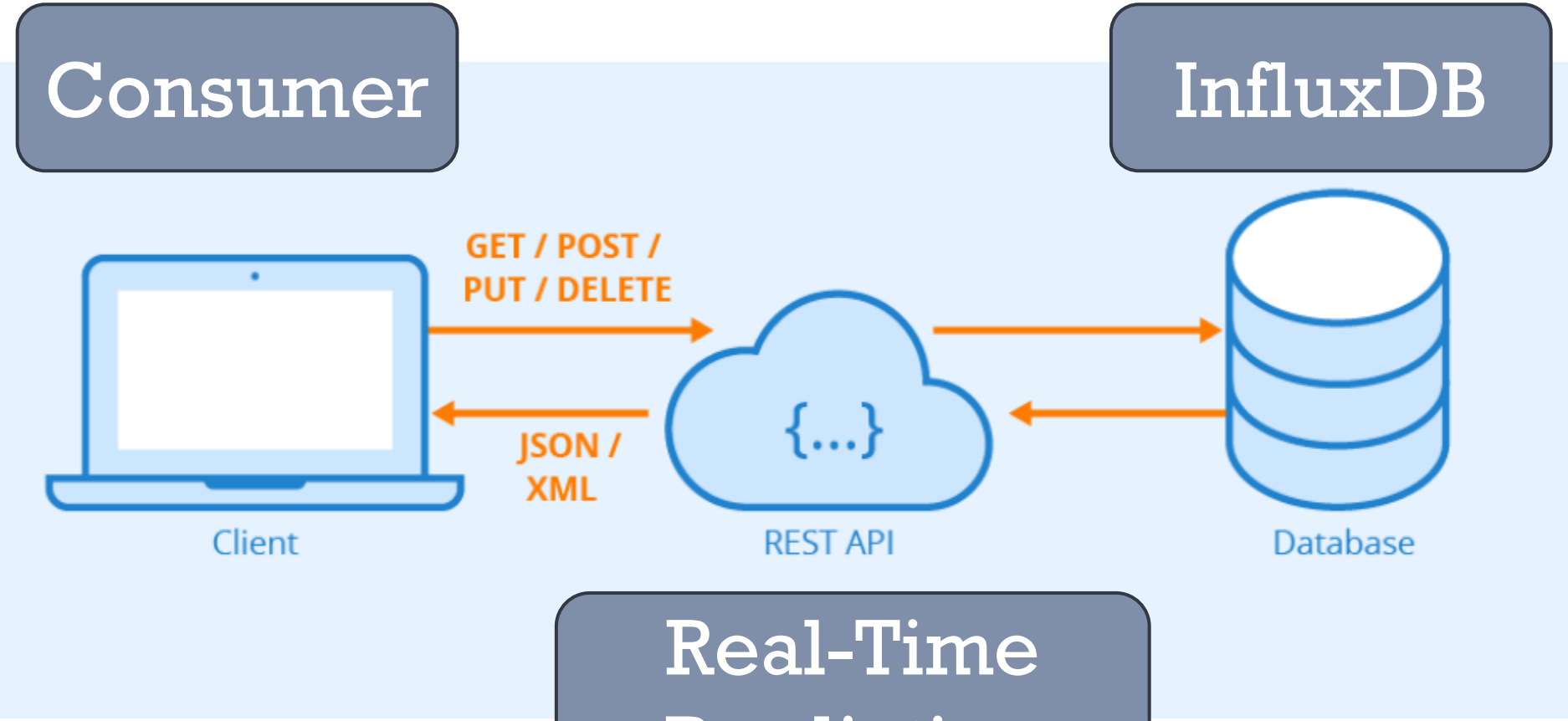
Develop REST API Server using Flask

https://flask.palletsprojects.com/en/3.0.x/



Develop REST API Server using Flask

https://flask.palletsprojects.com/en/3.0.x/



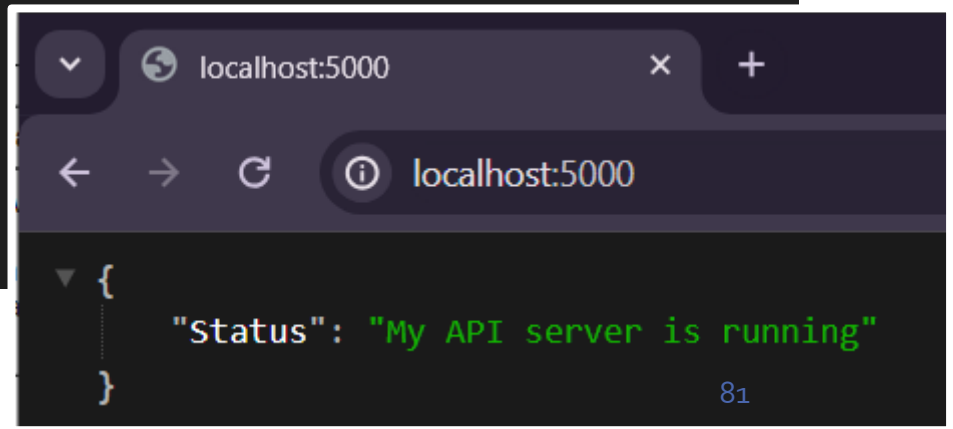
Develop REST API Server using Flask

```python
from flask import Flask, request, jsonify

app = Flask(__name__)

# Default Route
@app.route('/')
def hello():
    return jsonify({"Status": "My API server is running"})

if __name__ == '__main__':
    app.run()
```



Develop REST API Server using Flask

Look inside Training-code > IoTCloudServe@TEIN > SimpleAPI

```python
# Route for handling GET requests
@app.route('/get_data', methods=['GET'])
def get_data():
    return jsonify(data)

# Route for handling POST requests
@app.route('/add_data', methods=['POST'])
def add_data():
    req_data = request.get_json()
    if req_data:
        data.append(req_data)
        return jsonify({"message": "Data added successfully"})
    else:
        return jsonify({"error": "No data provided"}), 400
```

Develop REST API Server using Flask

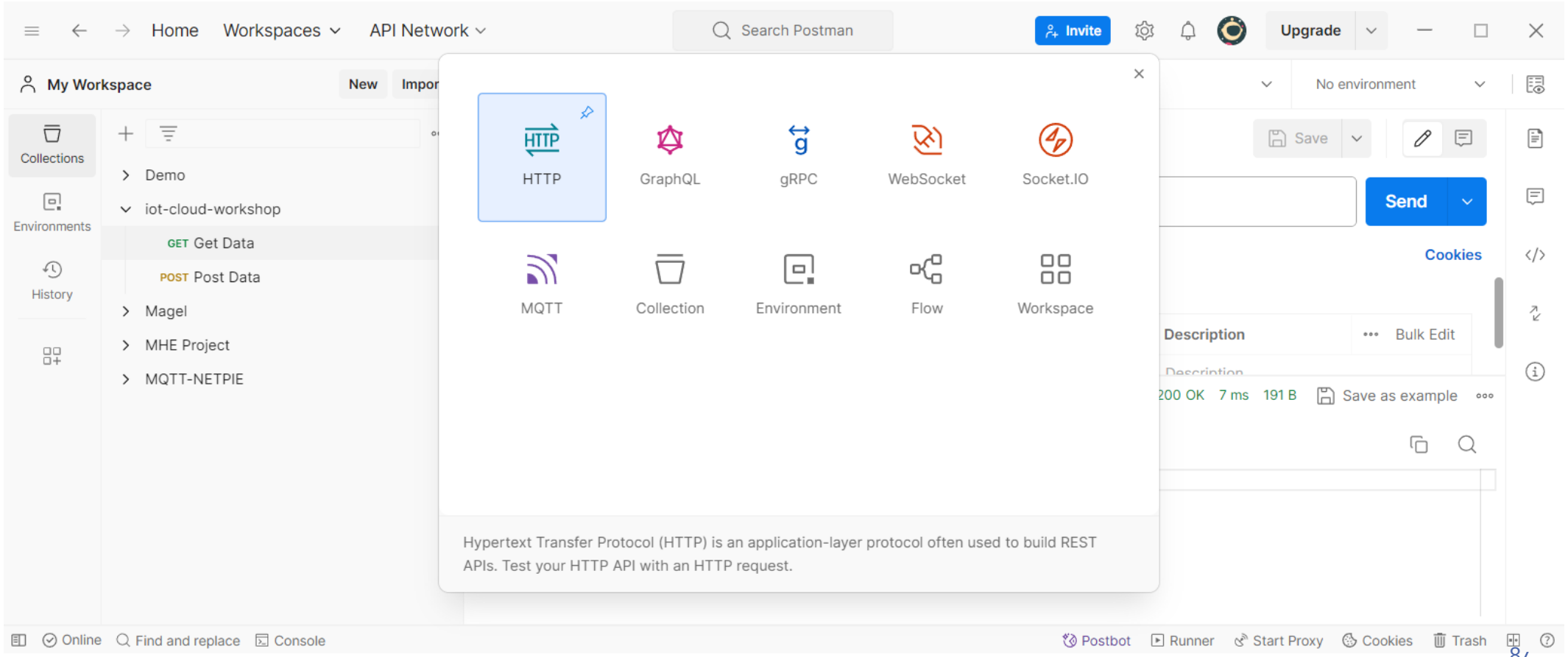
> cd [your directory]
> python app.py

By default, Flask will run on port 5000

You can test the API server using **Postman**

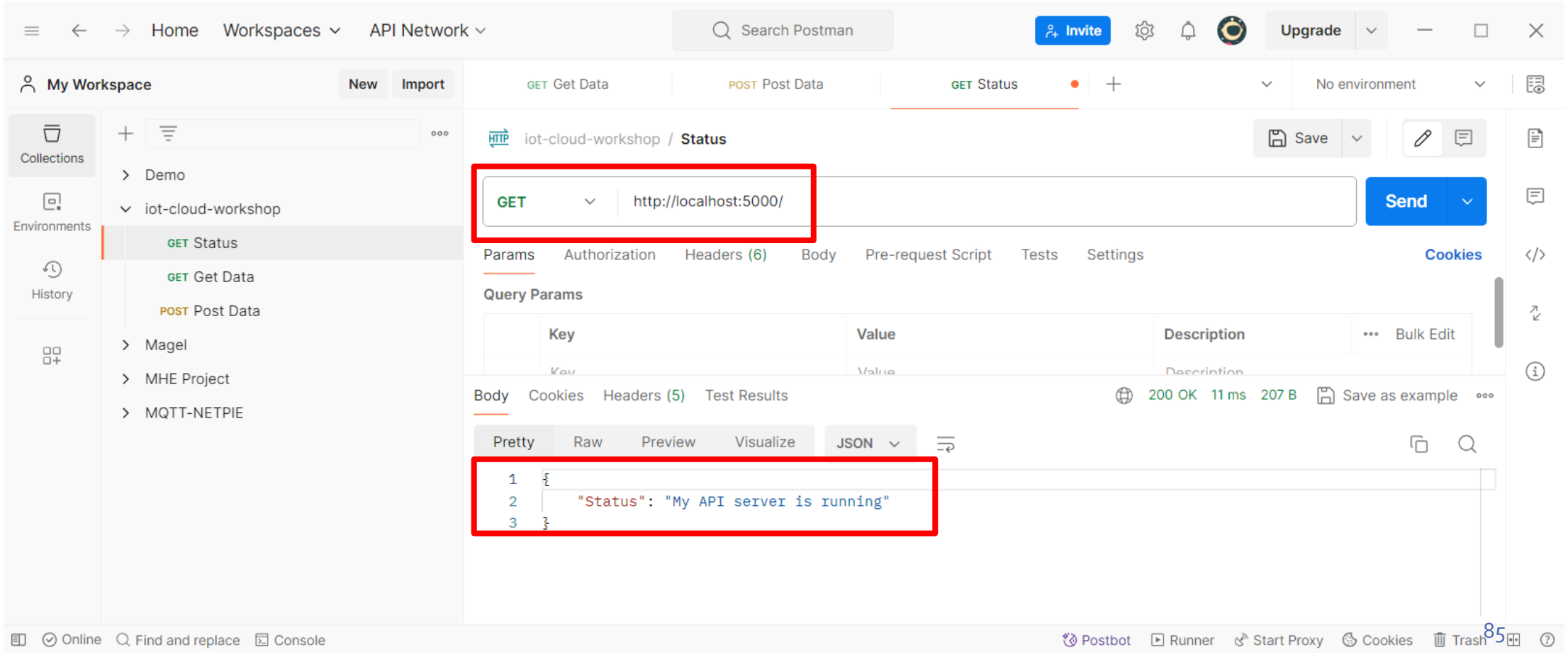
Test REST API Server using Postman

Create three new HTTP request



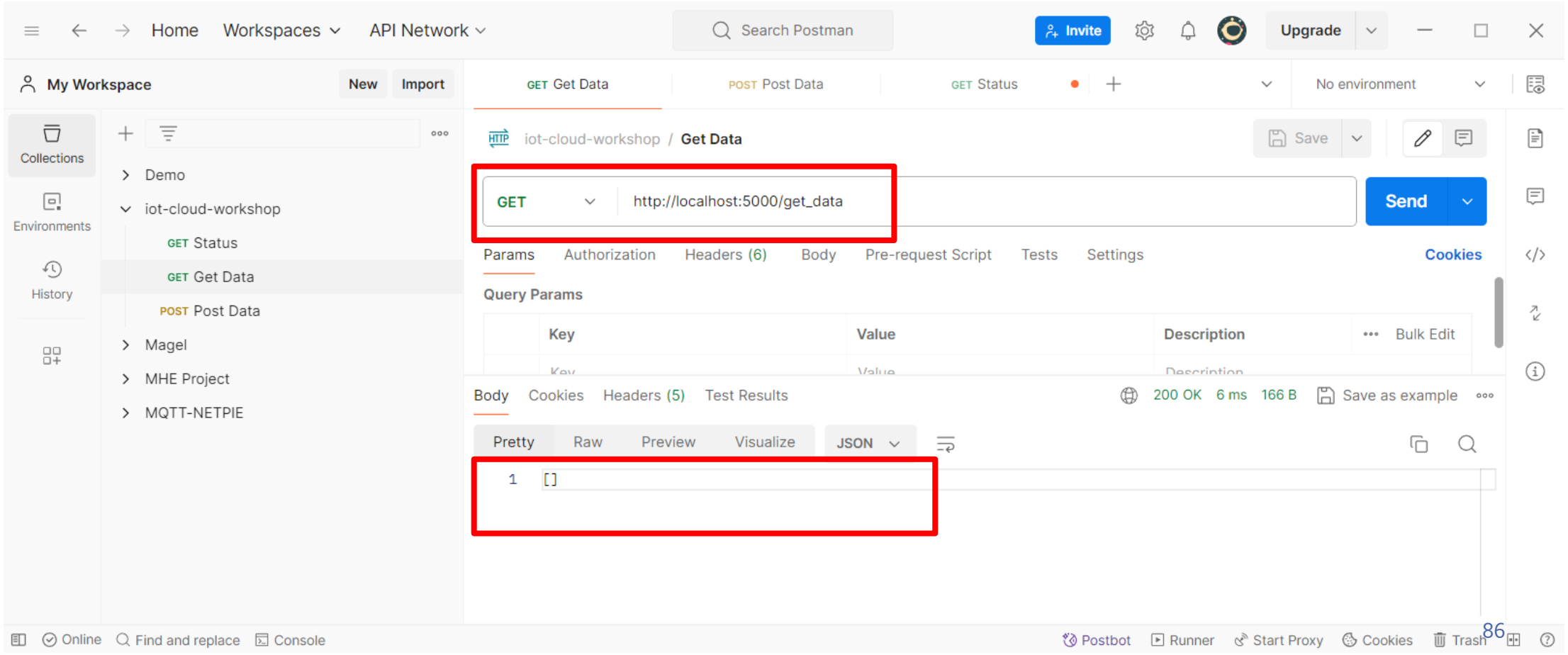
Test REST API Server using Postman

Default Route shows the Status



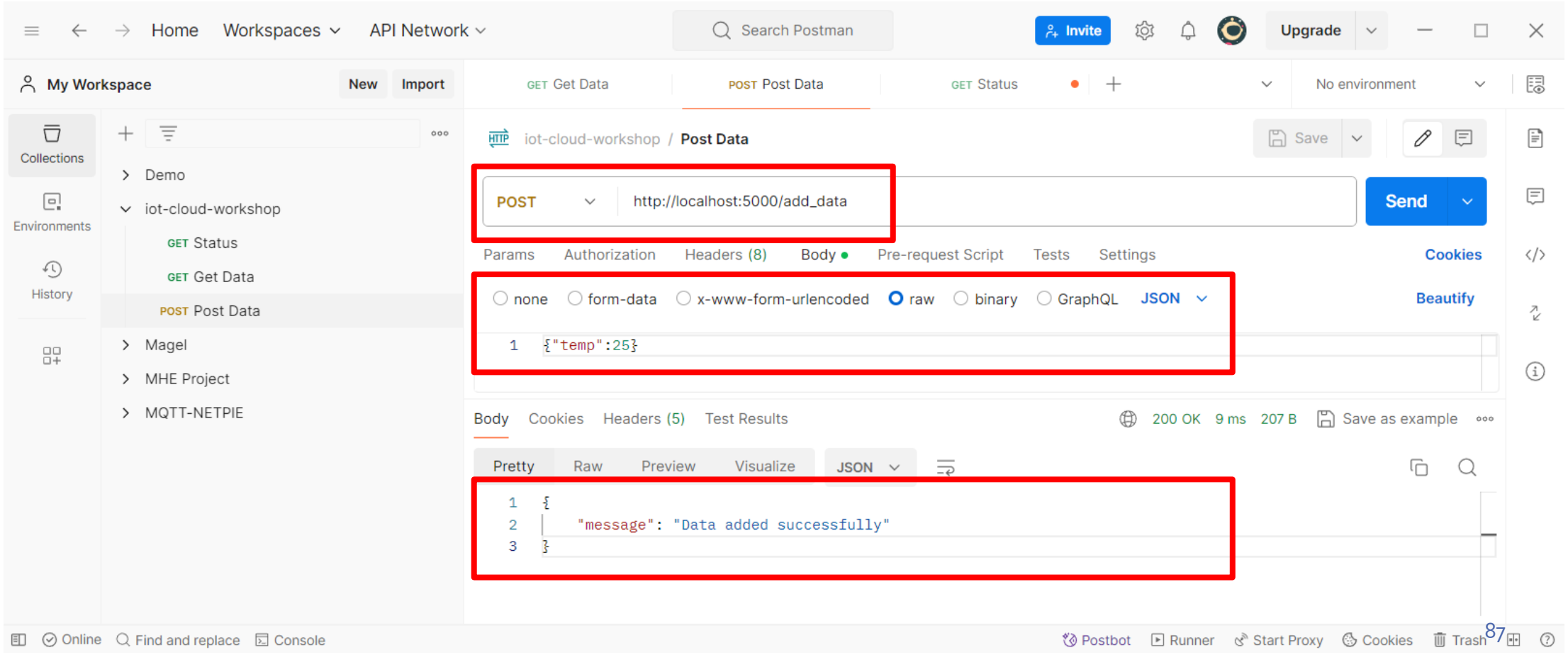
Test REST API Server using Postman

Get Data Route shows list of all data



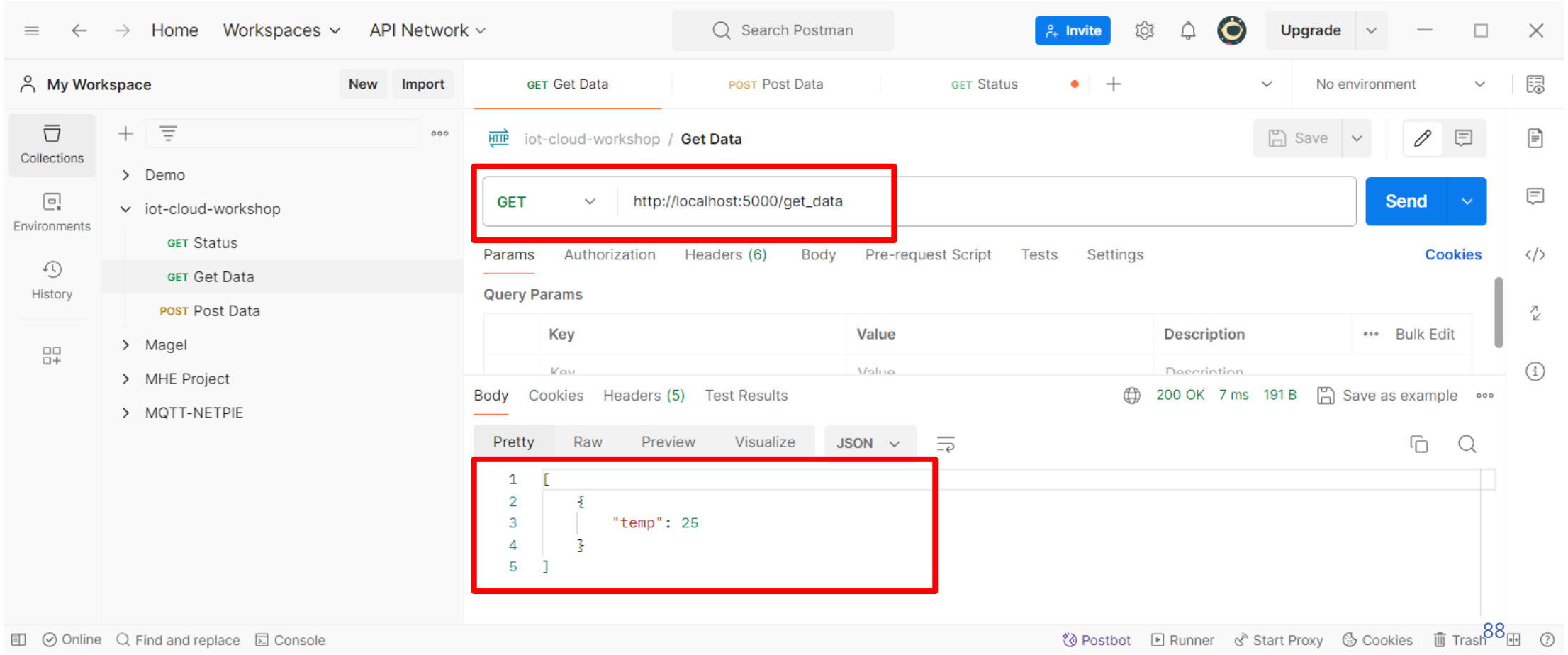
Test REST API Server using Postman

Add Data Route shows the response of adding JSON data



Test REST API Server using Postman

Get Data Route shows list of all data



Deploy Model on IoTCloudServe@TEIN

We will develop a REST API server to serve as the endpoint for consumers to send data, enabling our server to predict the occupancy of a room based on the received information.

Look inside Training-code > IoTCloudServe@TEIN > RealTimePrediction

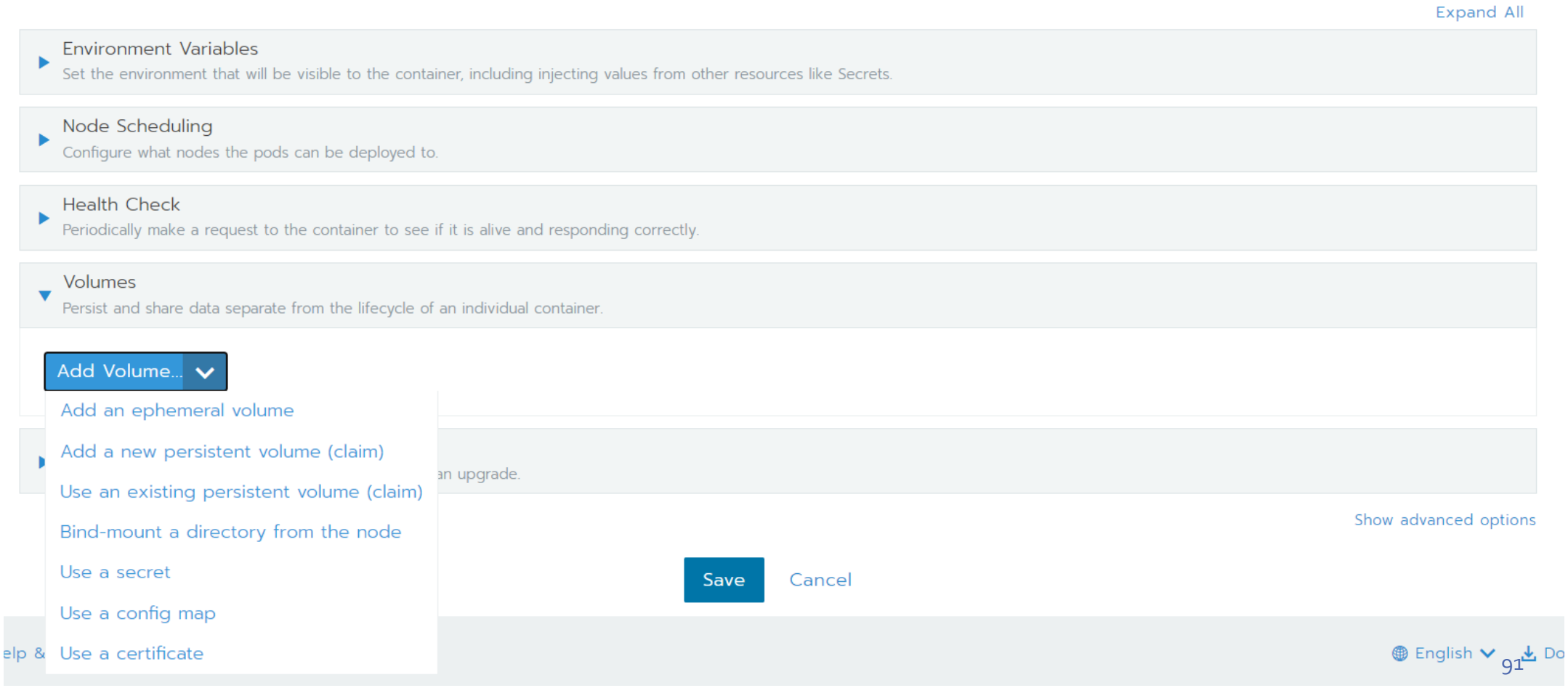
Deploy Model on IoTCloudServe@TEIN

Your Turn!!!

> Add your ML model to the folder
> Build image
> Push to Docker Hub
> Deploy on Rancher (map port, add secrets)
> Assign Domain Name

Deploy Model on IoTCloudServe@TEIN

You will need to include your InfluxDB secrets as a ".env"

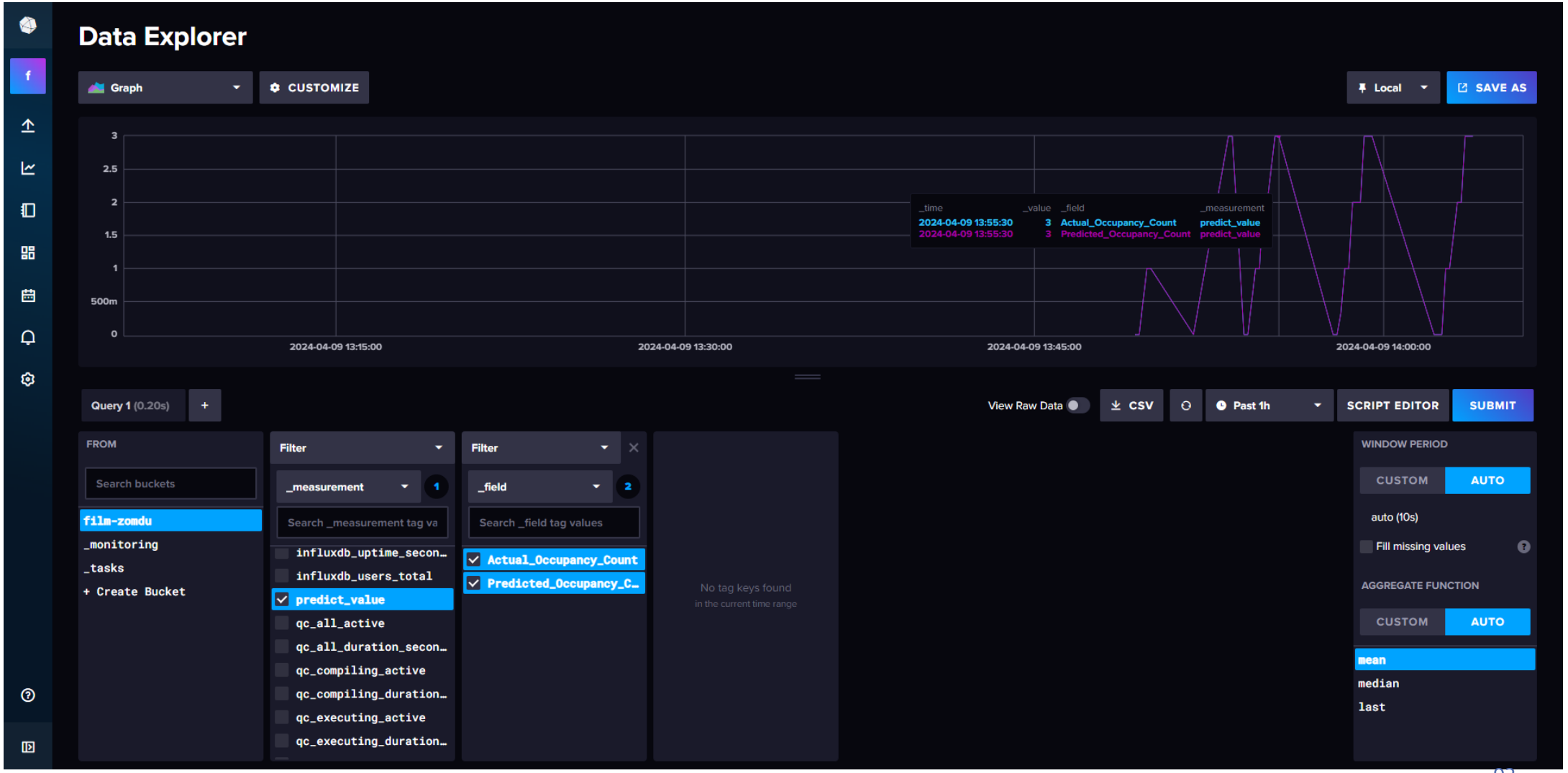


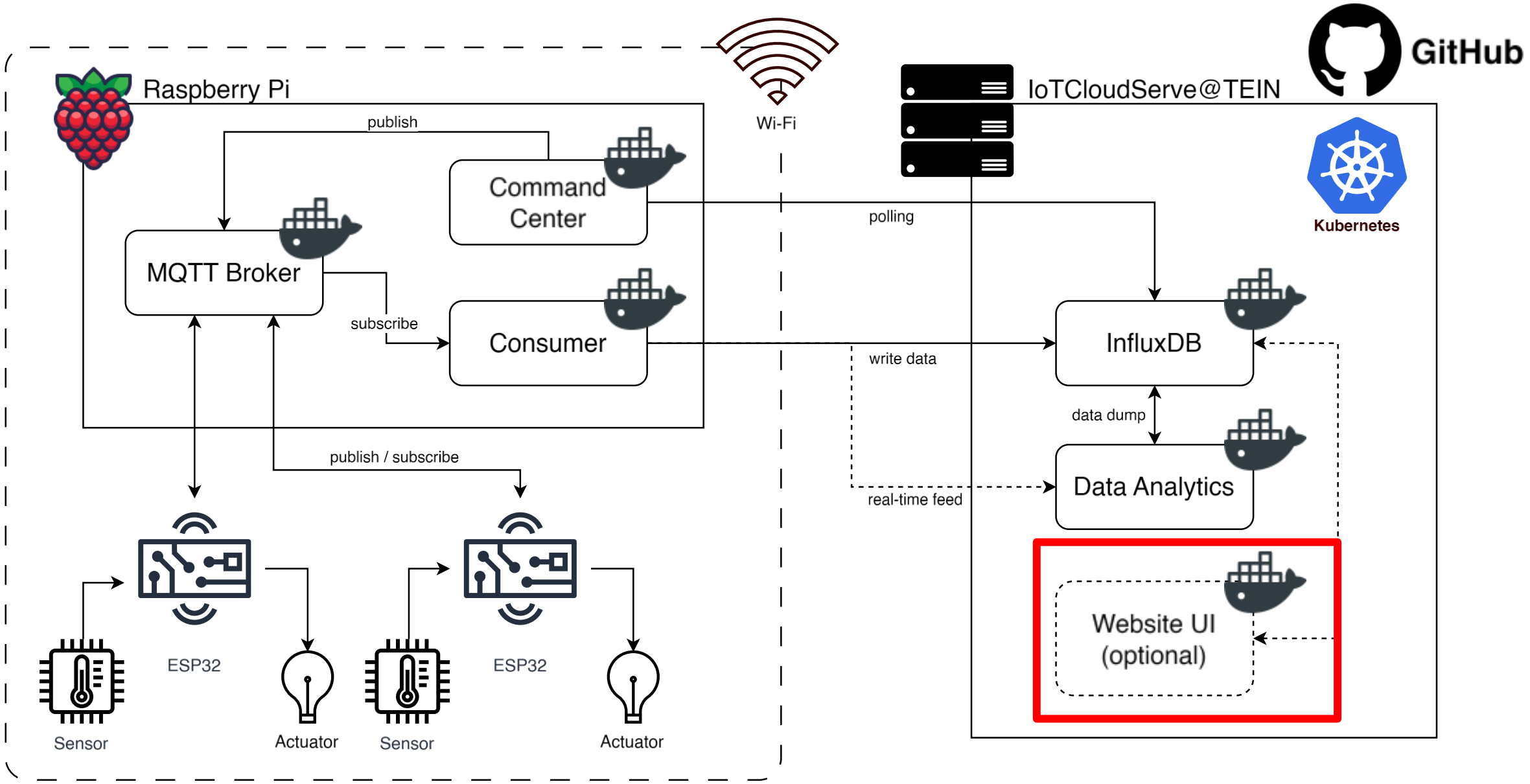
Deploy Model on IoTCloudServe@TEIN

Your Turn!!!

> Go back to Consumer in Raspberry Pi
> Change the PREDICT_URL inside ".env"
> Run the Consumer again with the new ".env"
> See the result in InfluxDB
measurement: "predict_value"

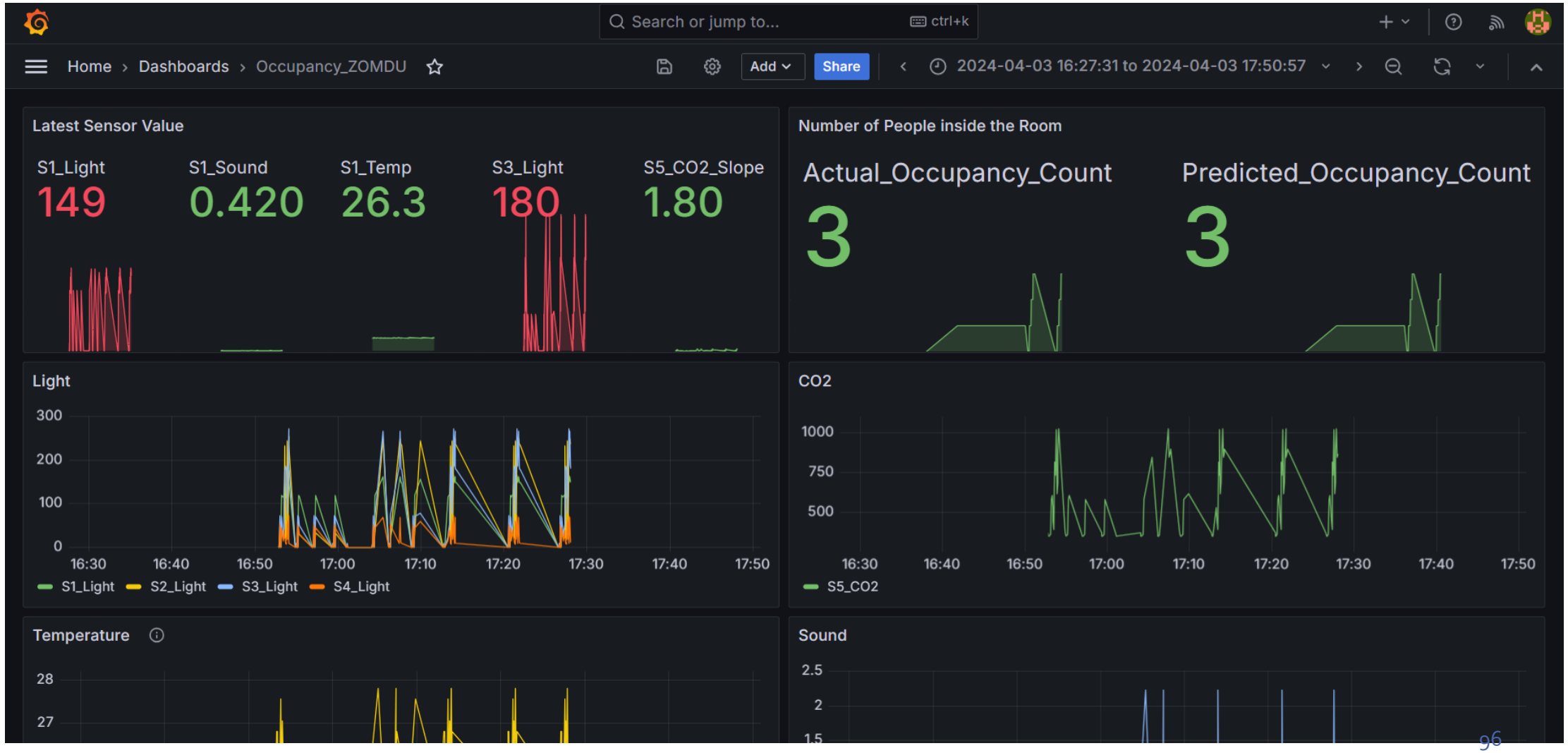
Check the result in InfluxDB




GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Website UI - InfluxDB Dashboard

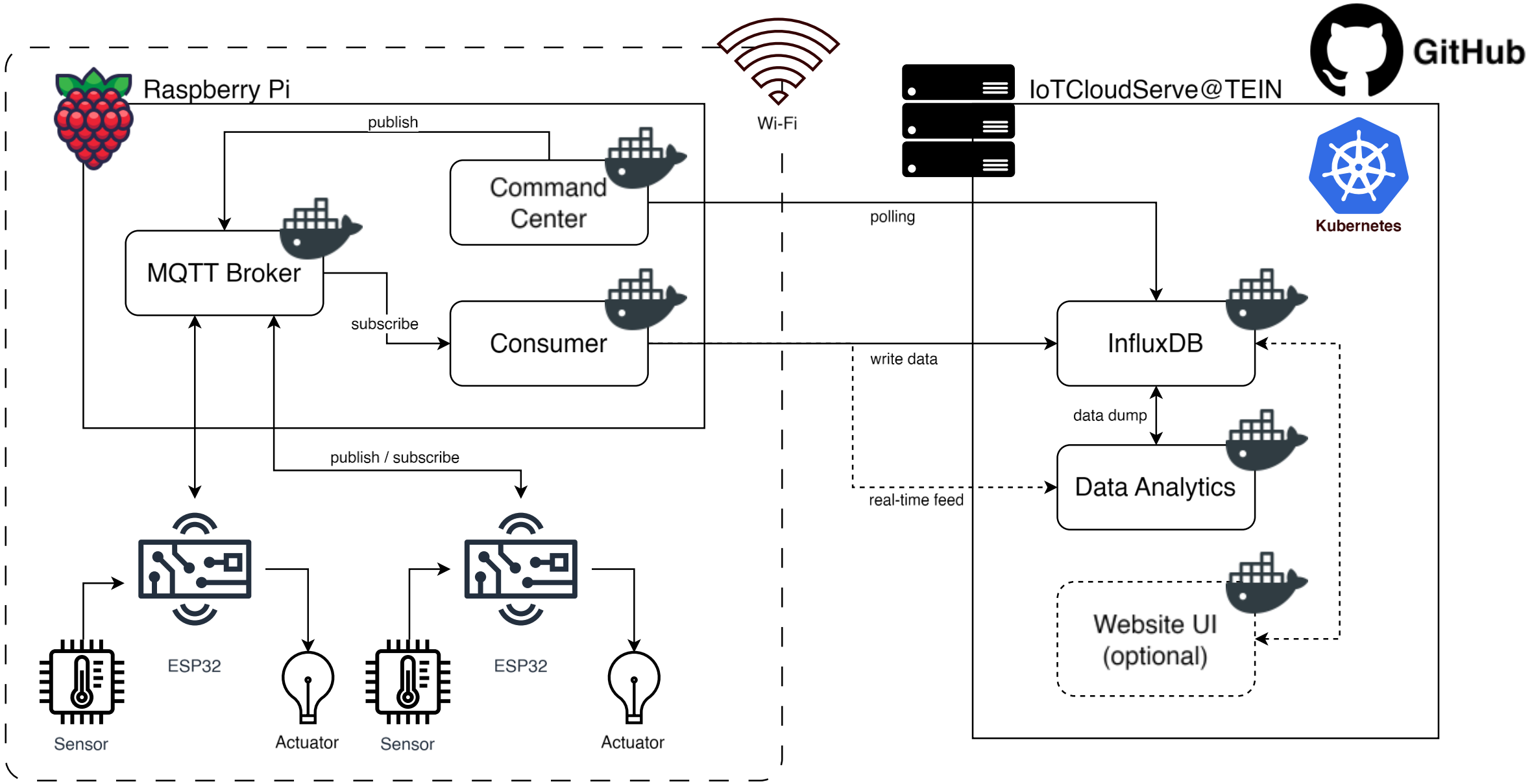
Deploy Grafana on IoTCloudServe@TEIN



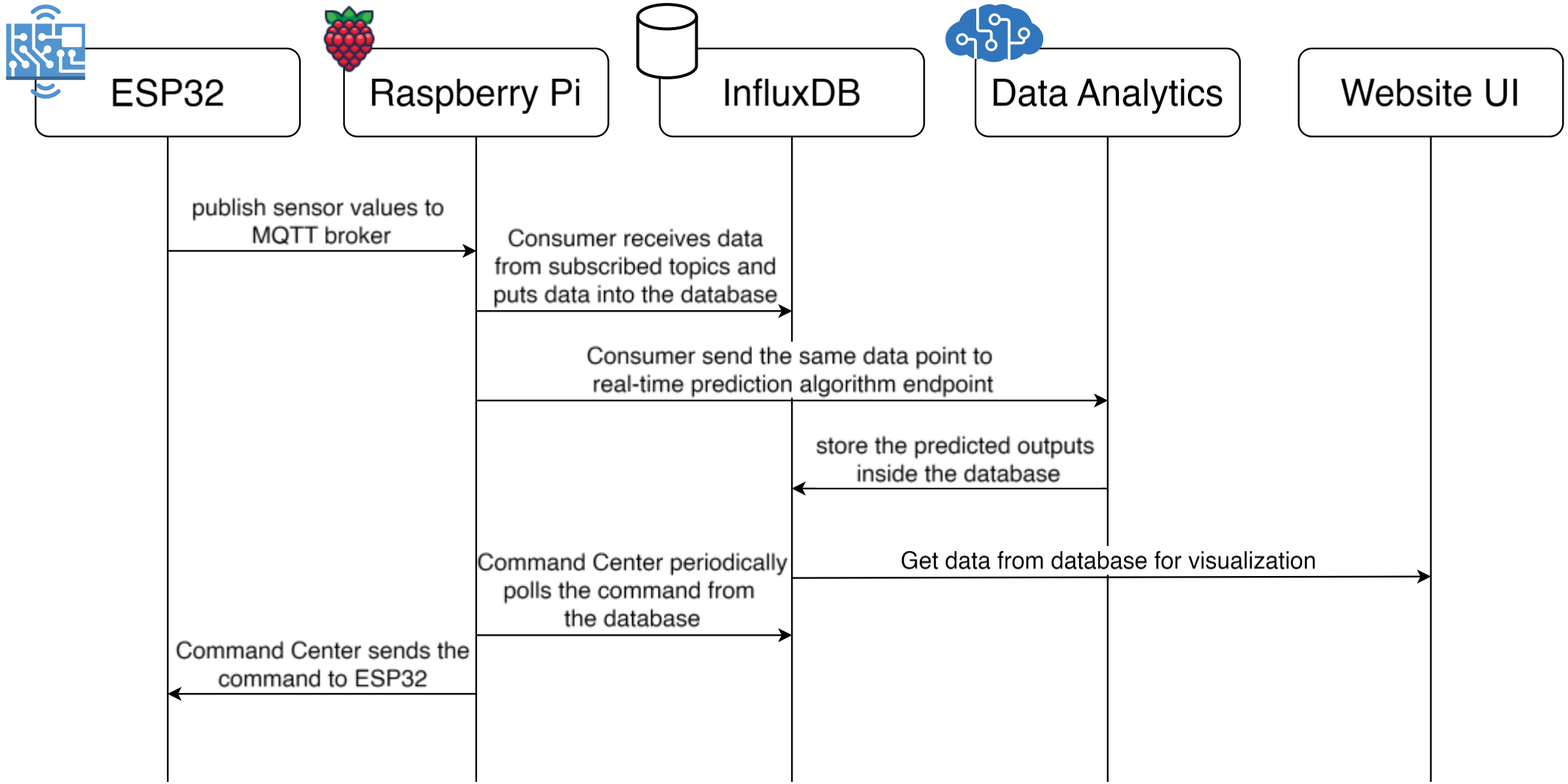
Deploy Grafana on IoTCloudServe@TEIN

Docker Image:
https://hub.docker.com/r/grafana/grafana-oss

Grafana is running on Port 3000


GitHub
Raspberry Pi
Wi-Fi
IoTCloudServe@TEIN
Kubernetes
publish
Command Center
MQTT Broker
subscribe
Consumer
polling
write data
InfluxDB
data dump
Data Analytics
real-time feed
Website UI (optional)
publish / subscribe
ESP32
ESP32
Sensor
Actuator
Sensor
Actuator

Data Flow


ESP32
Raspberry Pi
InfluxDB
Data Analytics
Website UI
publish sensor values to MQTT broker
Consumer receives data from subscribed topics and puts data into the database
Consumer send the same data point to real-time prediction algorithm endpoint
store the predicted outputs inside the database
Command Center periodically polls the command from the database
Get data from database for visualization
Command Center sends the command to ESP32

HOPE YOU ALL ENJOY THE WORKSHOP!