Develop a Python Script

Develop a Python Script

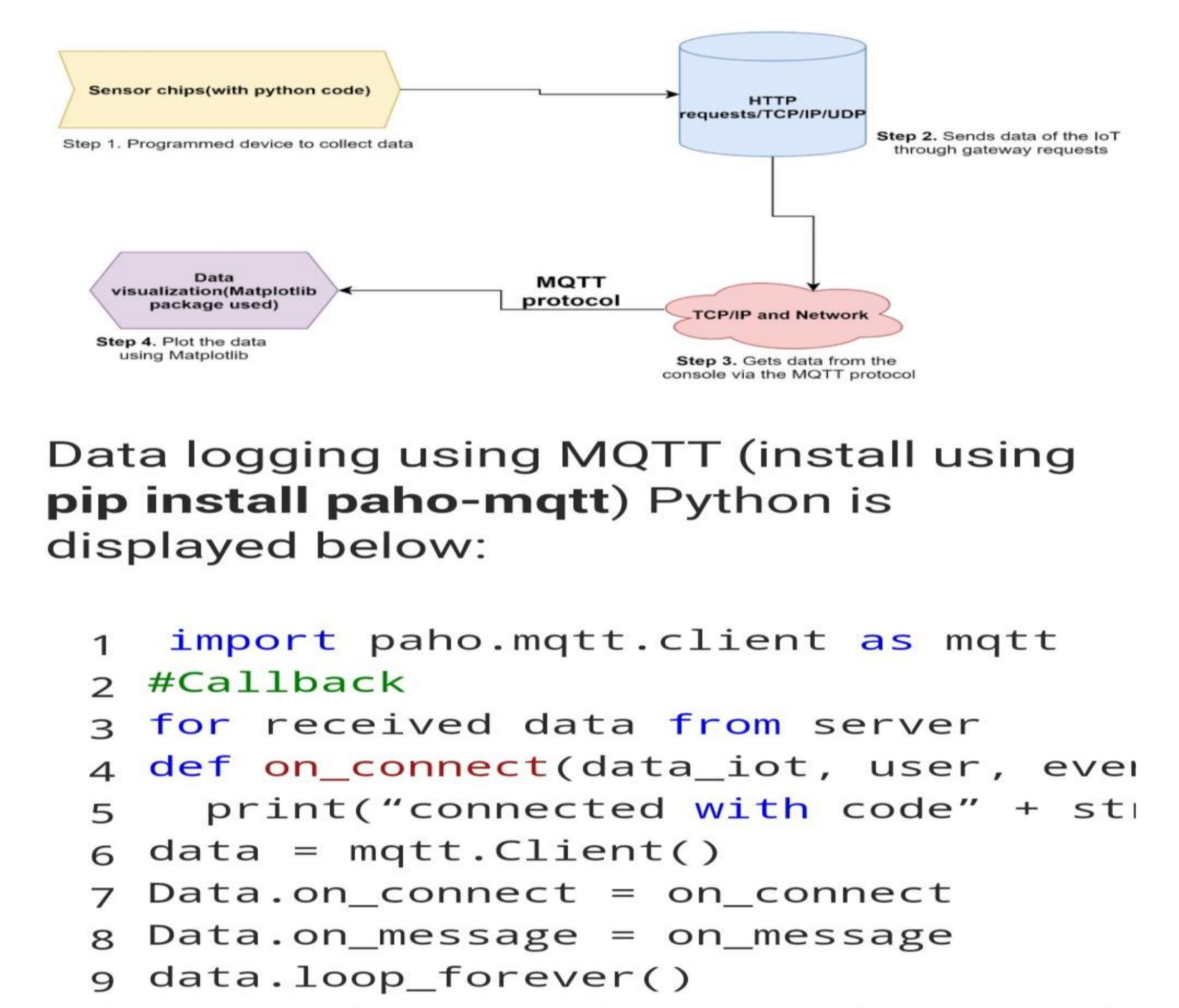
- Project Name:iot based safety gadget for child safety monitoring and notification
- Team ID:PNT2022TMID08341
- Date:8-November-2022

- NUMPY
- Numpy is a scientific computing package that helps to create datasets to test with the time series data in IoT. Numpy features are used in IoT to read sensor bulk data from the database inbuilt functions in the system
- SOCKETS AND MYSQLDB
- Sockets that facilitate networking in IoT devices include TCP/IP and UDP, which are compatible to work with Python packages. TCP/IP and UDP act as transport layer protocols for communication

- MATPLOTLIB
- To get data insights, matplotlib visualizes the most paramount operations by giving a variety of graphs to represent the data.

- REQUESTS, TKINTER AND TENSORFLOW
- To make HTTP calls and parse responses in Python, the request package acts as a major protocol for data exchanges. Tkinter GUI puts the aspects of Python script in a controlled distribution, which enables functional testing and repeated executions in IoT Python devices.

- IOT DEVICES USED TO DEVELOP APPLICATIONS IN IOT
- Raspberry Pi Model 3
- Intel Edison
- Arduino
- IOT SENSORS SIMULATORS USED IN PYTHON PROGRAMMING INCLUDE:
- MQ TELEMETRY TRANSPORT (MQTT) SENSOR SIMULATOR
- MQTT protocol for the IoT in Python enables high-speed data exchange with low payload communication between the devices. User-friendly requests of MQTT are made directly in Python. Data is collected in real-time and easily analyzed in mathematical computation libraries like matplotlib.



Sensor chips(with python code)

Step 1. Programmed device to collect data

HTTP requests/TCP/IP/UDP

Step 2. Sends data of the IoT through gateway requests

TCP/IP and Network

Step 3. Gets data from the console via the MQTT protocol

MQTT protocol

Data visualization(Matplotlib package used)

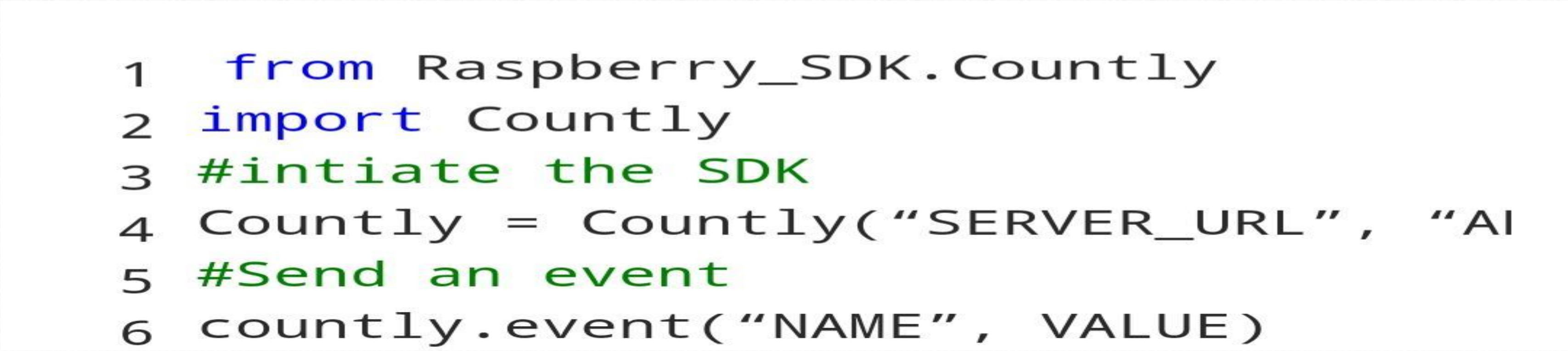
Step 4. Plot the data using Matplotlib

Data logging using MQTT (install using **pip install paho-mqtt**) Python is displayed below:

```
 import paho.mqtt.client as mqtt
#Callback
for received data from server
def on_connect(data_iot, user, eve
  print(“connected with code” + st
data = mqtt.Client()
Data.on_connect = on_connect
Data.on_message = on_message
data.loop_forever()
```

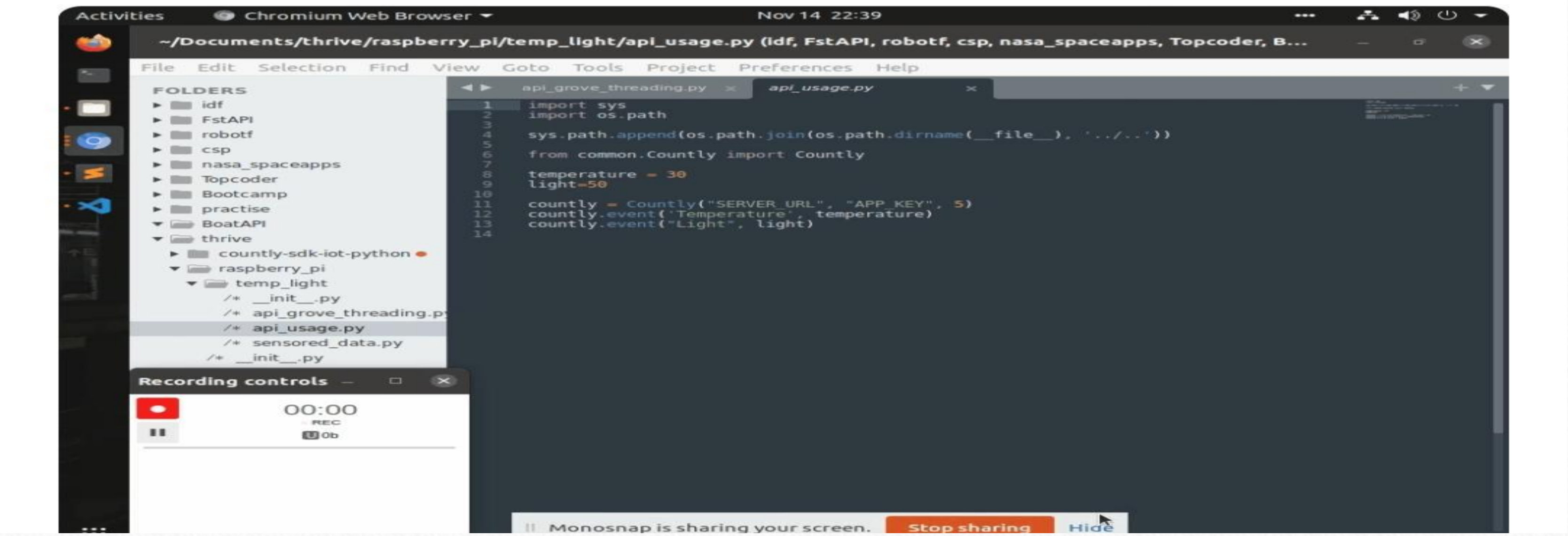
- **AZURE IOT SDK IN PYTHON**
- **Azure IoT hub offers a variety of features for IoT SDK usage which provides the ability to connect devices and services. The IoT SDK is supported by the MQTT protocol which facilitates the data exchange processes. The device requirements to be used along with Python include:**
- **Python version 3.7+: helps in both asynchronous and synchronous API**
- **Azure-iot-device library**

- COUNTLY IOT RASPBERRY PI SDK
- Sending data and visualizing data on a dashboard is simplified by involving the Countly IoT Pi SDK, which relies on internet connectivity for efficient and effective data insights from the device.
  The code below is used to start the process of collecting data using Countly IoT Pi SDK in Python. Install by running: pip install Raspberry_SDK:



```
 from Raspberry_SDK.Countly
import Countly
#intiate the SDK
Countly = Countly(“SERVER_URL”, “AI
#Send an event
countly.event(“NAME”, VALUE)
```

- Countly SDK also helps to retrieve data events for both analog and digital circuits. Use case of Countly IoT Raspberry Pi SDK is applicable in temperature room measuring and Bulb light. For instance, the server gets to pass the application key to collect the data and data is being manipulated by GroveAPI for raspberry IoT as displayed below:



```
import sys
import os.path

sys.path.append(os.path.join(os.path.dirname(__file__), '../..'))

from common.Countly import Countly

temperature = 30
light=50

countly = Countly("SERVER_URL", "APP_KEY", 5)
countly.event('Temperature', temperature)
countly.event("Light", light)
```