KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG, BELGAVI-590008

(An Autonomous Institution under Visvesvaraya Technological University Belagavi)

(APPROVED BY AICTE, NEW DELHI)

Open Book Assignment on

"HOME AUTOMATION USING RASPBERRY PI AND SPI PROTOCOL"

submitted in the partial fulfilment for the academic requirement of

6th Semester BE in

"ROBOTICS AND AUTOMATION"

Submitted by

NAME OF THE CANDIDATE	USN
Shruti Kambale	**2GI20EC135**
Vandana Binge	**2GI20EC156**
Anjana zare	**2GI20EC182**
Amruta Alagur	**2GI20EC179**

GUIDED BY

Prof. Praveen Kalkund

KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG, BELAGAVI – 590008

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

(APPROVED BY AICTE, NEW DELHI)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that Anusha Mathad of **6th Semester** bearing **USN:2GI20EC179,** has satisfactorily completed the course in *Open Book Assignment on Robotics and Automation*. It can be considered as a bonafide work carried out for partial fulfilment of the academic requirement of 6th Semester B.E. prescribed by KLS Gogte Institute of Technology, Belagavi during the academic year 2022-23.

The report has been approved as it satisfies the academic requirements prescribed for the said degree.

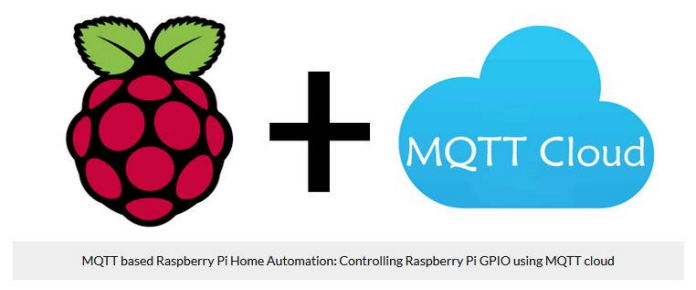
Signature of the Faculty Member Signature of the HOD

Date:

1.INDRODUCTION:

The IoT Controlled Home Appliances with MQTT and Raspberry Pi project offers a practical and accessible way to control home appliances remotely using the MQTT (Message Queuing Telemetry Transport) protocol and a Raspberry Pi. By leveraging the power of cloud-based MQTT brokers, such as Adafruit IO, this project enables users to control their appliances from anywhere in the world.Traditionally, local MQTT servers limited control to a specific location, but by hosting the MQTT server on a cloud platform, global control becomes possible. This opens up a wide range of possibilities for home automation and remote appliance management.

2.PROCEDURE:



MQTT based Raspberry Pi Home Automation: Controlling Raspberry Pi GPIO using MQTT cloud

2.1 Installing MQTT Cloud Libraries on Raspberry Pi

Here **Adafruit IO platform is used with Raspberry Pi as MQTT broker**. As we have used **Adafruit IO** platform many times in our previous tutorials similarly we can use it in Raspberry Pi. Make an account on Adafruit IO platform and make a feed, if you don't know how to make feed and customize Adafruit dashboard then follow the link.After making dashboard, install few libraries in Raspberry Pi to get start with MQTT.

1. First, update Your Pi and Python by issuing following commands

sudo apt-get update
sudo apt-get upgrade
sudo pip3 install --upgrade setuptools

2. Now, install Rpi.gpio and Adafruit blink libraries using below commands

sudo pip3 install RPI.GPIO
sudo pip3 install adafruit-blinka

3. Install Adafruit IO library using below command

sudo pip3 install adafruit-io

4. Clone the adafruit examples from github using below command

git clone https://github.com/adafruit/Adafruit_IO_Python.git

5. Then, navigate to the examples folder by entering the following command into the terminal:

cd Adafruit_IO_Python/examples/basics

6. For all examples in this folder,we need to set the ADAFRUIT_IO_KEY and ADAFRUIT_IO_USERNAME, which can be found from the Adafruit dashboard.

7. Now, open the **subscribe.py** file using *nano* editor. Type following command to open it

sudo nano subscribe.py

We have to modify this program to control any GPIO from dashboard.

Coding Explanation for controlling Raspberry Pi GPIO with MQTT

First, import all the required libraries to use GPIO pins and Adafruit MQTT client.

```
import RPi.GPIO as GPIO
import sys
from Adafruit_IO import MQTTClient
```

Now, set GPIO mode and define LED pin number and set as output.

```
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
ledPin = 12
GPIO.setup(ledPin, GPIO.OUT)
```

Next, we have to set AIO key and Username that we have found while creating the dashboard.

```
ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'
ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'
```

Enter the feed name that you have given to turn on and off the light. Here, it is *"light"*.

```
FEED_ID = 'light'
```

Now, define a function that will be called when there will be an event happen. So, we will subscribe the Feed using ***client.subscribe(FEED_ID)***

```
def connected(client):
    client.subscribe(FEED_ID)
    print('Waiting for feed data...')
```

After subscribing the feed, we have to check for the new value and store it into a *payload* variable. For this message function is called. So, whenever there is "1" in payload variable, make the led pin HIGH and for "0" make it LOW.

```
def message(client, feed_id, payload):
  print('Feed {0} received new value: {1}'.format(feed_id, payload))
  if payload == 1:
      GPIO.output(ledPin, GPIO.HIGH)
else:
     GPIO.output(ledPin, GPIO.LOW)
```

Now, create an MQTT client to connect with the Adafruit IO platform and send the messages to and fro.

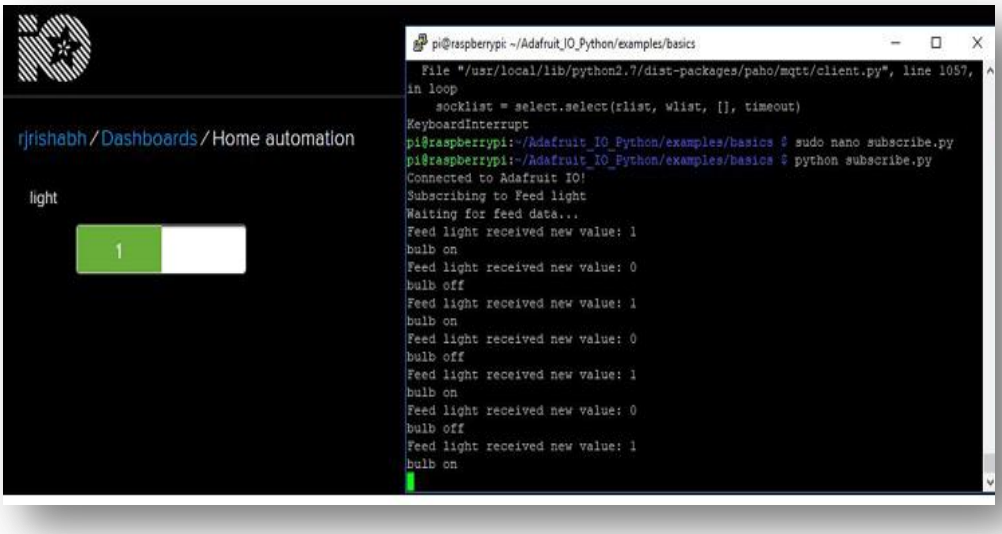
```
client = MQTTClient(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
client.on_connect = connected
client.on_disconnect = disconnected
```

Also, take care about proper indentation in the code else it will show an error. **Complete python code** is given at the end of the tutorial.

Finally, save the program using ctrl+x and hit the enter. Now, we have to run the script in order to subscribe the messages. So In the terminal type **python subscribe.py** and hit the enter.

python subscribe.py

We will see a message *Waiting For Feed Data*… as shown below snapshot.

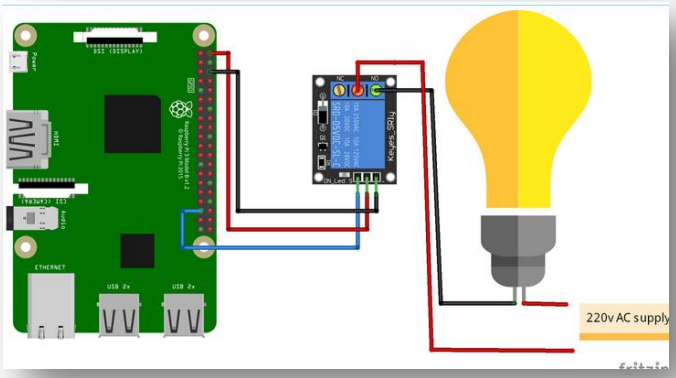


3.Code:

```
import RPi.GPIO as GPIO
import sys
from Adafruit_IO import MQTTClient
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
ledPin = 12
GPIO.setup(ledPin, GPIO.OUT)
ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'
ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'
FEED_ID = 'light'
def connected(client):
    # Subscribe to changes on a feed named Counter.
    print('Subscribing to Feed {0}'.format(FEED_ID))
    client.subscribe(FEED_ID)
    print('Waiting for feed data...')
def disconnected(client):
    sys.exit(1)
def message(client, feed_id, payload):
    print('Feed {0} received new value: {1}'.format(feed_id, payload))
  if payload == 1:
      GPIO.output(ledPin, GPIO.HIGH)
else:
     GPIO.output(ledPin, GPIO.LOW)
```

```
# Create an MQTT client instance.

client = MQTTClient(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)

# Setup the callback functions defined above.

client.on_connect = connected

client.on_disconnect = disconnected

client.on_message = message

# Connect to the Adafruit IO server.

client.connect()

client.loop_blocking()
```

Circuit Daigram:



4.CONCLUSION:

The project highlights the importance of cloud-based MQTT brokers in extending the reach of home automation systems. By hosting the MQTT server on a cloud platform like Adafruit IO, users can control their appliances globally, providing flexibility and convenience. This opens up possibilities for managing appliances while away from home, creating schedules, integrating with other IoT devices, and implementing advanced automation scenarios. Furthermore, the project underscores the significance of MQTT as a lightweight and efficient messaging protocol for IoT applications. MQTT's publish-subscribe model allows for reliable and scalable communication between devices and brokers, making it an ideal choice for home automation projects. Its low overhead and support for asynchronous communication contribute to its efficiency and suitability for resource-constrained devices like the Raspberry Pi.

5.APPLICATIONS:

1. Remote Appliance Control: The project enables users to control their home appliances remotely from anywhere in the world using MQTT messages. This feature is particularly useful for turning on/off lights, fans, or other electrical devices before arriving home or while being away.
2. Security and Surveillance: The Raspberry Pi can be connected to various security devices such as cameras, motion sensors, and door/window sensors. By using MQTT, users can remotely monitor their home's security, receive notifications on intrusions, and control security devices in real-time.
3. Smart Lighting: The project can be extended to create dynamic and personalized lighting solutions. By integrating with sensors or using time-based triggers, users can automate lighting patterns, adjust brightness levels, and create ambiance according to their preferences.
4. Home Theater Automation: The Raspberry Pi can be utilized to automate home theater systems. Users can control audio/video equipment, adjust volume levels, switch between devices, and create custom multimedia experiences.
5. Integration with IoT Ecosystems: The project can be integrated into broader IoT ecosystems by connecting with other IoT devices and platforms. This allows for seamless interoperability, data sharing, and coordination between different smart devices within the home.
6. Data Logging and Analytics: By leveraging MQTT and cloud-based platforms, users can collect and analyze data from various sensors and devices. This provides insights into energy consumption patterns, occupancy trends, and environmental conditions, helping users make informed decisions regarding energy efficiency and resource management.

6.RESULT: