**Experiment: - 10**

**AIM: To Study MQTT protocol T**

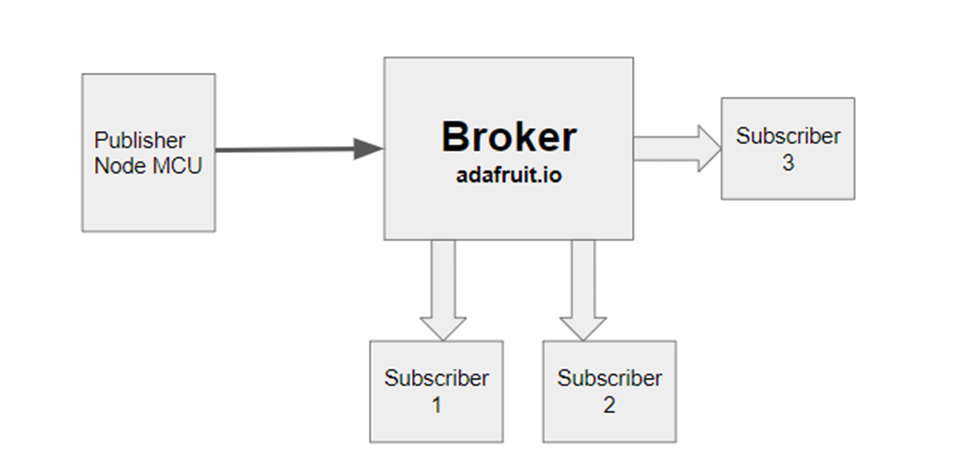
**Objective: To measure Humidity, Temperature and control a LED through the internet via MQTT.**

**Theory:**

MQTT is a Client Server publish/subscribe messaging transport protocol. This protocol is widely used in the field of IoT for communication between Machine to Machine because of its following features: light weight, open, and designed so as to be easy to implement.

This protocol is easy to implement and also very easy to understand. It basically comprises one Broker and multiple clients where clients can be treated as our smart phone, sensors, etc. and they all communicate with the server which is known as Broker.

In this protocol, every client needs to connect to any address of the broker which is known as the topic to be subscribed to in MQTT. In a single broker there can be multiple topics and clients can also subscribe to multiple topics of the same broker.



**Figure 1: Example of MQTT protocol**

**Mobile Application Link:**

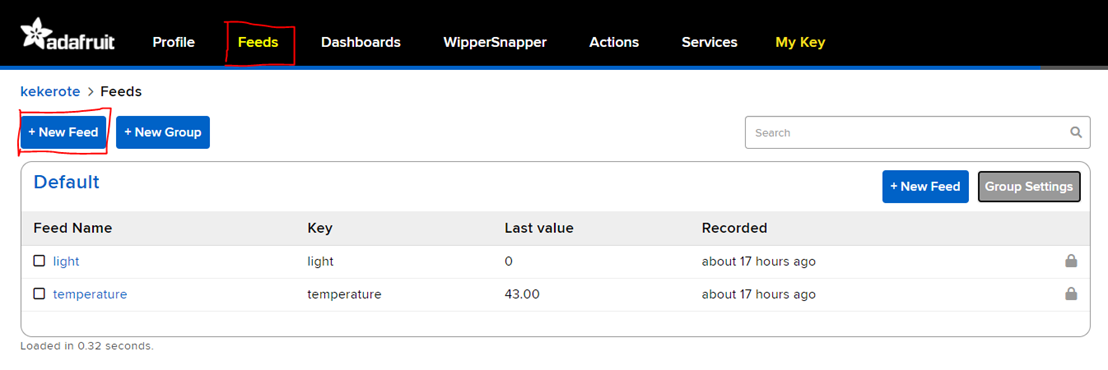
<https://play.google.com/store/apps/details?id=at.tripwire.mqtt.client&hl=en_IN>

|  |  |
| --- | --- |
|  |  |
|  |  |

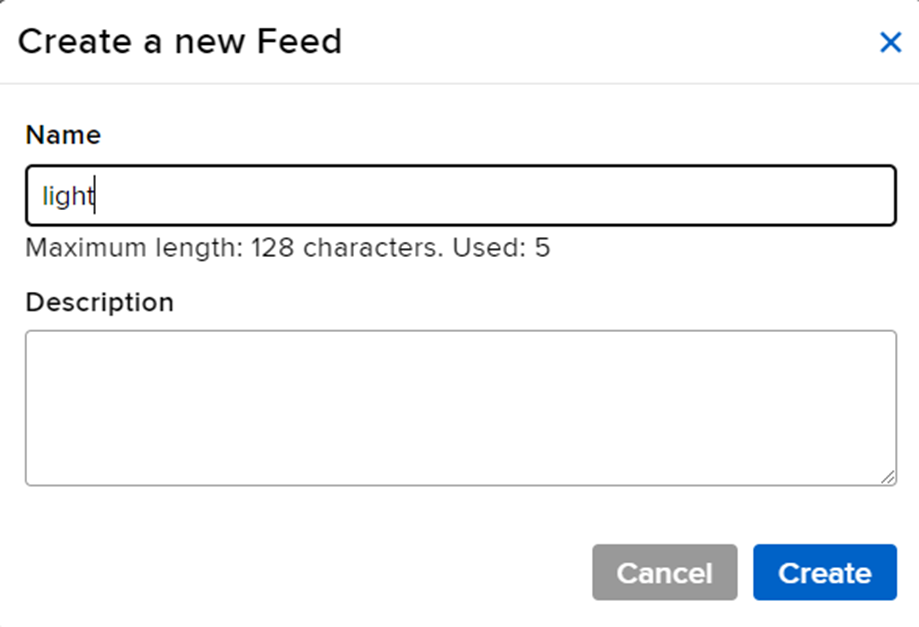
**Figure 2: Mobile Application Settings**



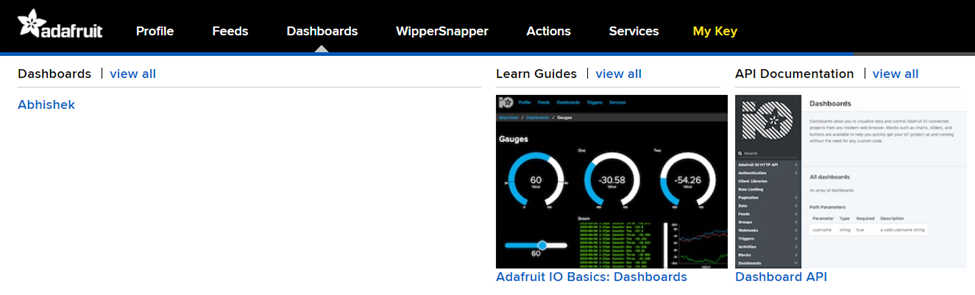
**Figure 3: Adafruit.io platform user name and key**



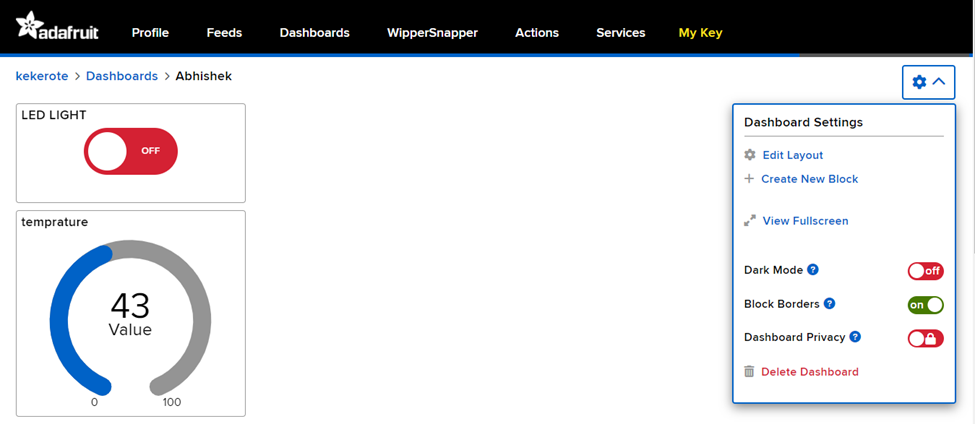
**Figure 4: Feed Window**



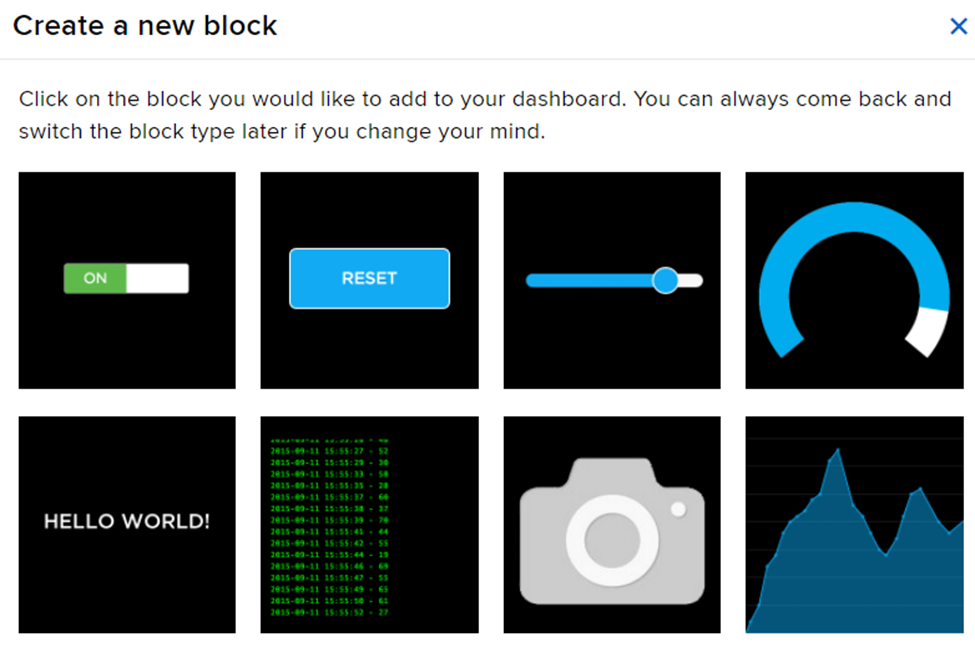
**Figure 5: Creating a new feed**



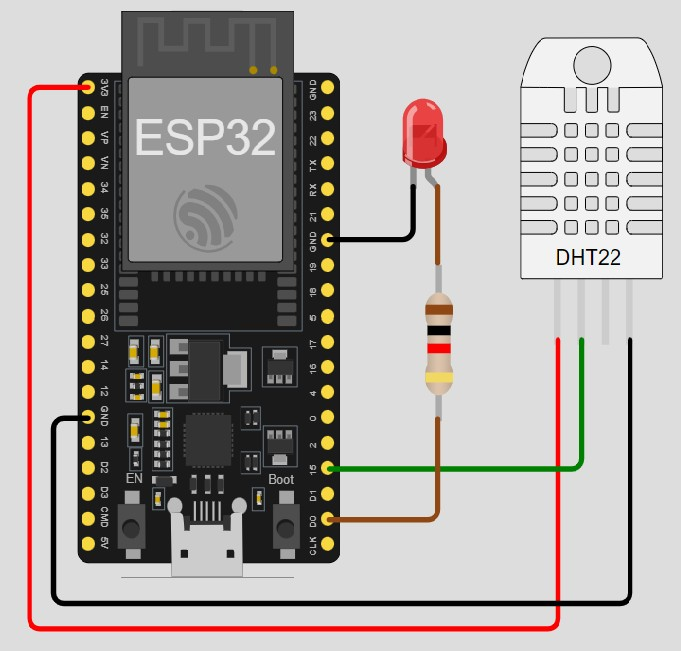
**Figure 6: Dashboard Window**



**Figure 7: Dashboard Block**



**Figure 8: Creating a block in dashboard**

**Connection Diagram:**

**Figure 9: Connection diagram**

**Program**

#include <ESP8266WiFi.h>

#include "Adafruit\_MQTT.h"

#include "Adafruit\_MQTT\_Client.h"

#include <Wire.h>

#include "DHT.h"

#define DHTTYPE DHT22

const int LED = 16;

uint8\_t DHTPin = 5; //D1

DHT dht(DHTPin, DHTTYPE);

float \_Temperature;

float \_Humidity;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* **WiFi Access Point** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define WLAN\_SSID "Hare Krishn"

#define WLAN\_PASS "Iot@1234"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* **Adafruit.io Setup** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define AIO\_SERVER "io.adafruit.com"

#define AIO\_SERVERPORT 1883 // use 8883 for SSL

#define AIO\_USERNAME "IOT\_LAB\_ADA"

#define AIO\_KEY "aio\_dAEH39JkzHCdypNLK3ZLgNPw4lGp"

/\*\*\*\*\*\*\*\*\*\*\*\* **Global State (you don't need to change this!)** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.

WiFiClient client;

// or... use WiFiFlientSecure for SSL

//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.

Adafruit\_MQTT\_Client mqtt( & client, AIO\_SERVER, AIO\_SERVERPORT, AIO\_USERNAME, AIO\_KEY);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* **Feeds** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Setup a feed called 'onoff' for subscribing to changes.

Adafruit\_MQTT\_Subscribe LIGHT = Adafruit\_MQTT\_Subscribe( & mqtt, AIO\_USERNAME "/feeds/LIGHT");

Adafruit\_MQTT\_Publish Humidity = Adafruit\_MQTT\_Publish( & mqtt, AIO\_USERNAME "/feeds/Humidity");

Adafruit\_MQTT\_Publish Temperature = Adafruit\_MQTT\_Publish( & mqtt, AIO\_USERNAME "/feeds/Temperature");

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Sketch Code \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Bug workaround for Arduino 1.6.6, it seems to need a function declaration

// for some reason (only affects ESP8266, likely an arduino-builder bug).

void MQTT\_connect();

void setup() {

Serial.begin(115200);

delay(10);

pinMode(LED, OUTPUT);

pinMode(DHTPin, INPUT);

dht.begin();

Serial.println(F("Adafruit MQTT demo"));

// Connect to WiFi access point.

Serial.println();

Serial.println();

Serial.print("Connecting to ");

Serial.println(WLAN\_SSID);

WiFi.begin(WLAN\_SSID, WLAN\_PASS);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println();

Serial.println("WiFi connected");

// Serial.println("IP address: "); Serial.println(WiFi.localIP());

// Setup MQTT subscription for onoff feed.

mqtt.subscribe( & LIGHT);

}

void loop() {

// Ensure the connection to the MQTT server is alive (this will make the first

// connection and automatically reconnect when disconnected). See the MQTT\_connect

// function definition further below.

MQTT\_connect();

// this is our 'wait for incoming subscription packets' busy subloop

// try to spend your time here

Adafruit\_MQTT\_Subscribe \* subscription;

while ((subscription = mqtt.readSubscription(5000))) {

if (subscription == & LIGHT) {

Serial.print(F("Got\_light: "));

Serial.println((char \* ) LIGHT.lastread);

uint16\_t num = atoi((char \* ) LIGHT.lastread);

digitalWrite(LED, num);

}

}

Serial.print(F("\nSending Humidity and Temperature value value "));

\_Humidity = dht.readHumidity();

// Read temperature as Celsius (the default)

\_Temperature = dht.readTemperature();

// Read temperature as Fahrenheit (isFahrenheit = true)

// Check if any reads failed and exit early (to try again).

if (isnan(\_Humidity) || isnan(\_Temperature)) {

Serial.println(F("Failed to read from DHT sensor!"));

return;

}

Serial.print(F("Humidity: "));

Serial.println(\_Humidity);

Serial.print(F("% Temperature: "));

Serial.print(\_Temperature);

Serial.println(F(" °C "));

if (!Humidity.publish(\_Humidity)) {

Serial.println(F("Failed"));

}

delay(100);

if (!Temperature.publish(\_Temperature)) {

Serial.println(F("Failed"));

}

delay(100);

}

// Function to connect and reconnect as necessary to the MQTT server.

// Should be called in the loop function and it will take care if connecting.

void MQTT\_connect() {

int8\_t ret;

// Stop if already connected.

if (mqtt.connected()) {

return;

}

Serial.print("Connecting to MQTT... ");

uint8\_t retries = 3;

while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected

Serial.println(mqtt.connectErrorString(ret));

Serial.println("Retrying MQTT connection in 5 seconds...");

mqtt.disconnect();

delay(5000); // wait 5 seconds

retries--;

if (retries == 0) {

// basically die and wait for WDT to reset me

while (1);

}

}

Serial.println("MQTT Connected!");

}

**Code Description:**

**ESP8266 with MQTT for Sensor Data and LED Control**

**This code connects an ESP8266 device to the internet via WiFi and uses MQTT (Message Queuing Telemetry Transport) to:**

**Measure temperature and humidity:** It uses a DHT22 sensor connected to pin 5 and reads values using the Adafruit\_MQTT library**.**

**Control an LED:** The LED is connected to pin 16 and can be turned on/off based on messages received over MQTT**.**

**Publish sensor data:** It publishes temperature and humidity readings to MQTT topics named "Humidity" and "Temperature" under the user's Adafruit.io account.

**Subscribe to LED control:** It subscribes to an MQTT topic named "LIGHT" to receive commands for turning the LED on/off.

**Conclusion:** We have studied and understood MQTT protocol by successfully performing the experiment of measuring temperature and humidity and controlling an LED via the internet using MQTT.