Step by Step INTERNET OF THINGS TUTORIAL USING ESP8266

Nin Hayati Mohd Yusoff
Mohamad Ariffin Zulkifli
Noremy Che Azemi

POLITEKNIK MALAYSIA
Port Dickson

STEP BY STEP INTERNET OF THING TUTORIAL USING ESP8266

Nin Hayati Mohd Yusoff
Mohamad Ariffin Zulkifli
Noremy Che Azemi

Perpustakaan Negara Malaysia Cataloguing-in-Publication Data
Nin Hayati Mohd. Yusoff, 1981-
STEP BY STEP INTERNET OF THING TUTORIAL USING ESP8266 /
Nin Hayati Mohd Yusoff, Mohamad Ariffin Zulkifli, Noremy Che Azemi.
Mode of access: Internet
eISBN 978-967-2897-46-0
1. Internet of things
2. Embedded Internet devices
3. Government publications--Malaysia
4. Electronic books
I. Mohamad Ariffin Zulkifli, 1991-. II. Noremy Che Azemi, 1979-.
III. Title.
004.678

e ISBN 978-967-2897-46-0

9 7 8 9 6 7 2 8 9 7 4 6 0

Step-by-Step Internet of
Things Tutorial Using ESP8266

PUBLISHED BY:
Politeknik Port Dickson
KM14, Jalan Pantai, 71050 Si Rusa
Port Dickson, Negeri Sembilan
FEBRUARY 2022

PREFACE

The Internet of Things (IoT) is everything can communicate with each other via internet at anywhere and anytime. A thing in IoT can be person, sensor or other object able to exchange the data through network. Moreover, an IoT ecosystem consists embedded system that categories with low power, low bandwidth, low cost and etc. The embedded devices also call smart devices used to collect and share the data from the sensor by connecting to an IoT gateway. For this condition, IoT deployment rapidly year by year in multiple fields like smart home, smart farming and healthcare to improve quality of life.

NodeMCU ESP8266 is a microcontroller that can access Wi-Fi network capable to integrated with sensor and other application with minimal development up-front and loading during runtime. Therefore, this module has a powerful to bring to life a wide range of application scenarios using features like cloud connectivity, Wi-Fi security support and low power operation.

This book focuses on step-by-step IoT Tutorial using ESP8266 for beginner. It starts with the Tutorial 1: introduction to NodeMCU ESP8266, sensor and actuator in Tutorial 2, continue with Tutorial 3: NodeMCU ESP8266 using HTTP Protocol, Tutorial 4: NodeMCU ESP8266 using MQTT Protocol, Tutorial 5: Node-RED Flow & Node-RED. Finally, Tutorial 6: IoT Application Control Via Smartphone.

Hopefully, this book can benefit to teacher, student or anyone who is interesting in learning IoT ecosystem.

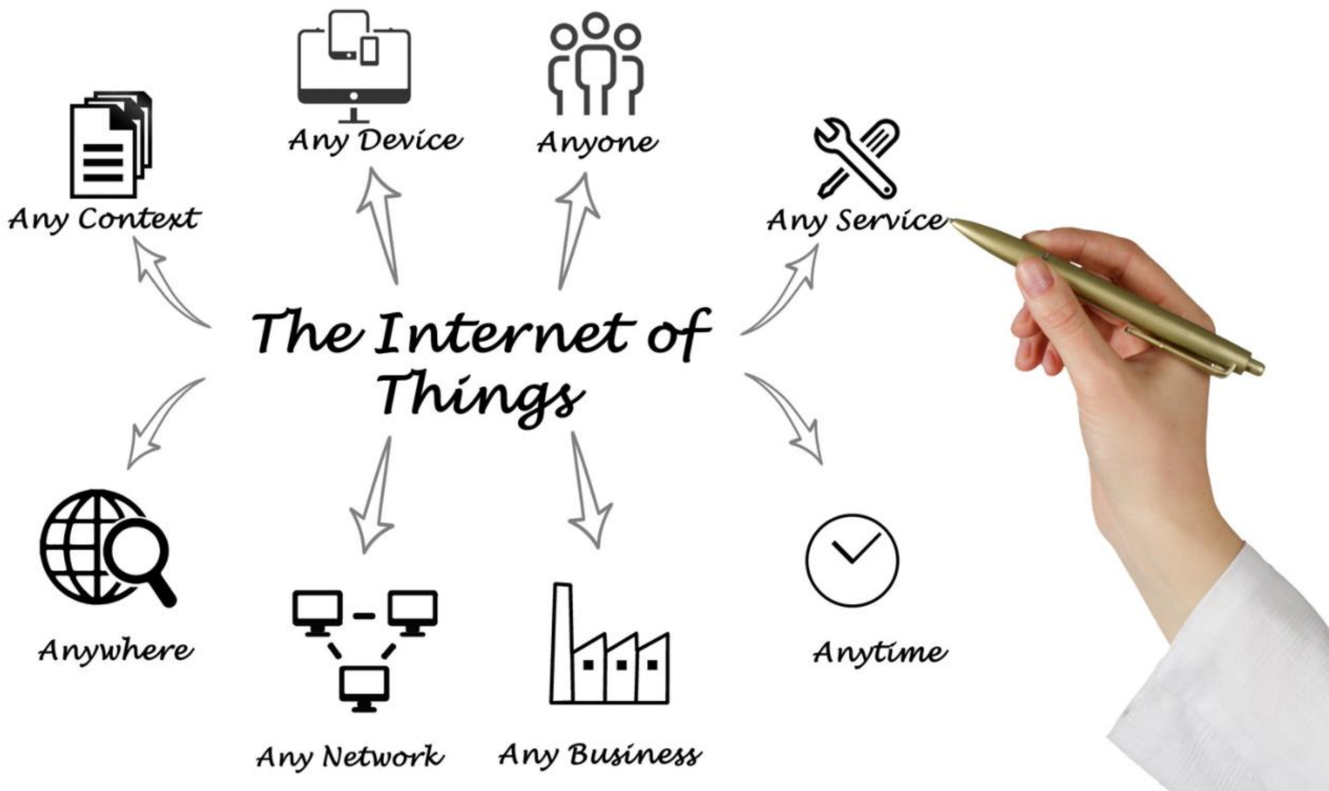
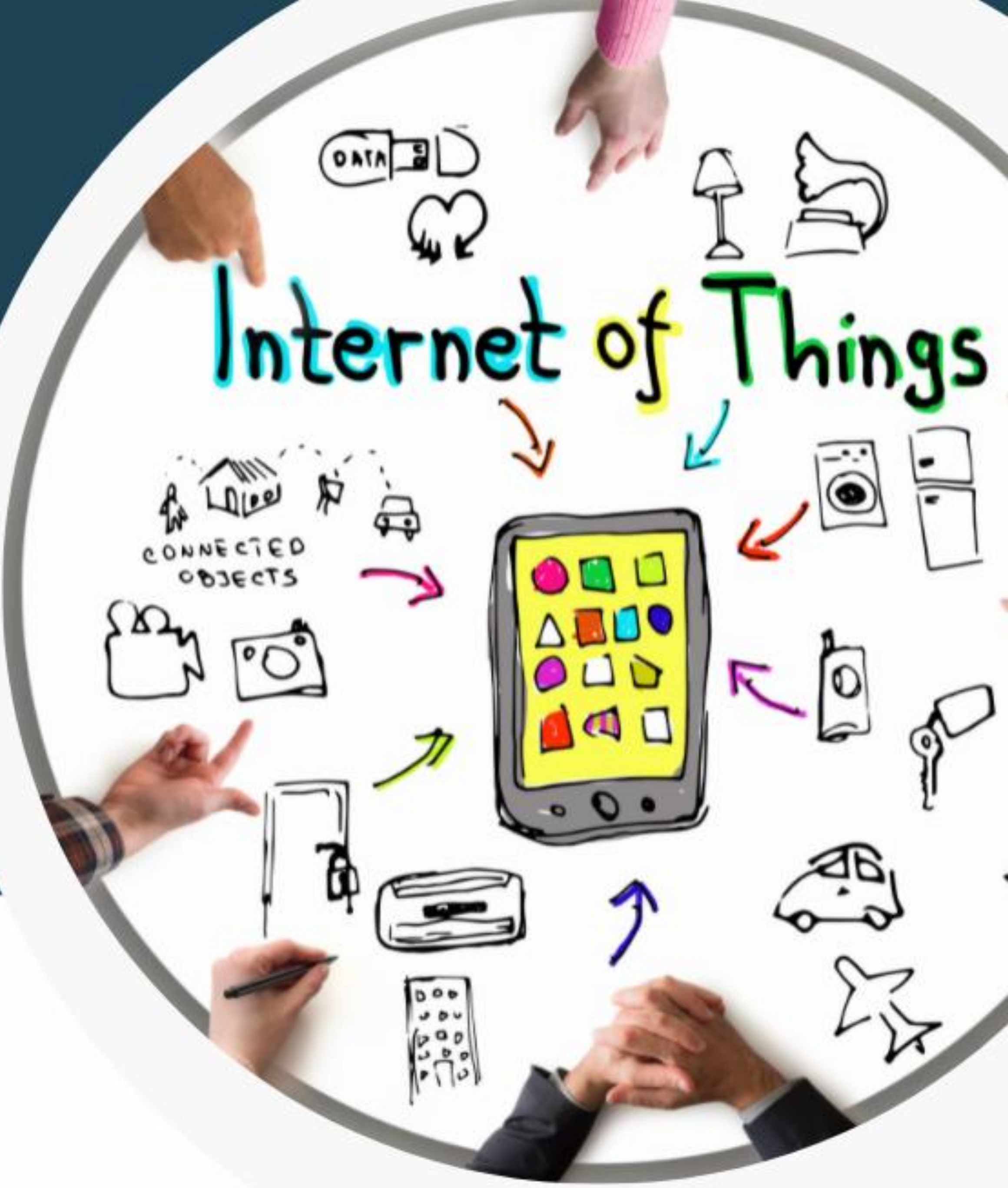


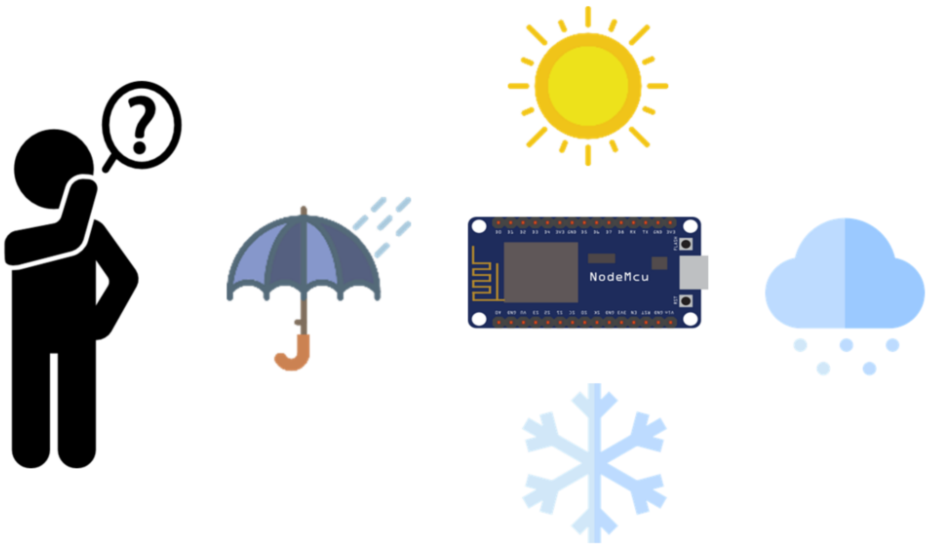
TABLE OF CONTENT



TUTORIAL 01

TUTORIAL: 1

Title: Introduction to NodeMCU ESP8266



OBJECTIVES

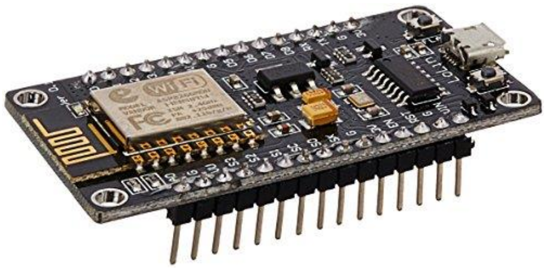
Upon completion of this tutorial, you should be able to:

- Install and configure Arduino IDE for ESP8266 board correctly.
- create a project file and write a simple code using Node MCU truly.
- Construct a simple hardware connection and perform compiling, uploading and testing successfully.

EQUIPMENTS

- NodeMCU ESP8266
- Breadboard
- Jumper cables
- Arduino IDE
- Internet connection
- Resistor 220 Ω (Red, Red, Brown, Gold)
- LED

THEORY

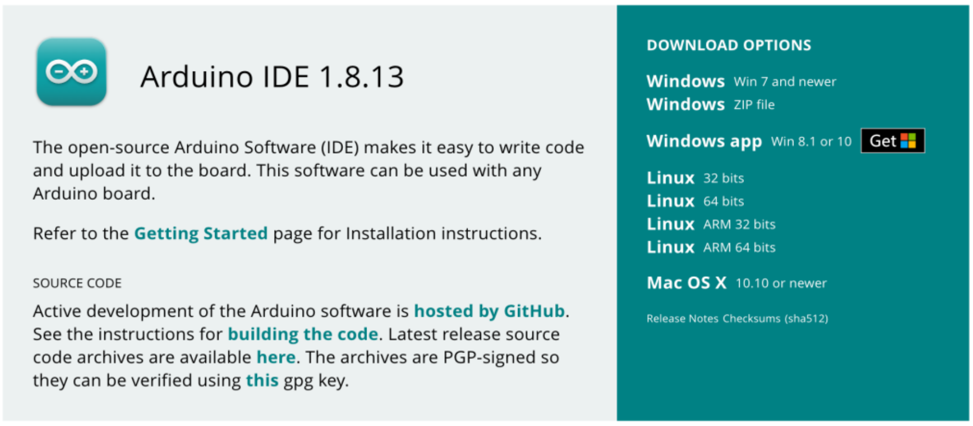


Node MCU (MicroController Unit) is an open-source software and ESP8266 is a embedded devices (hardware) that build in System-on-a Chip (SoC) to run the firmware. The ESP8266 was designed contain the elements of computer: Central Processing Unit (CPU), Random Access Memory (RAM), Wi-Fi, modern operating system and a software development kit (SDK). Meanwhile the Arduino Integrated Development Environment (Arduino IDE) is an open-source software used to write and upload code to the NodeMCU development board. Therefore, it become a cost effective and highly integrated MCU for Internet of Thing (IoT) application.

PROCEDURE

PART A: SETTING UP THE ARDUINO IDE

1. Go to https://www.arduino.cc/en/software and download the latest Arduino IDE.



Arduino IDE 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the **Getting Started** page for Installation instructions.

SOURCE CODE

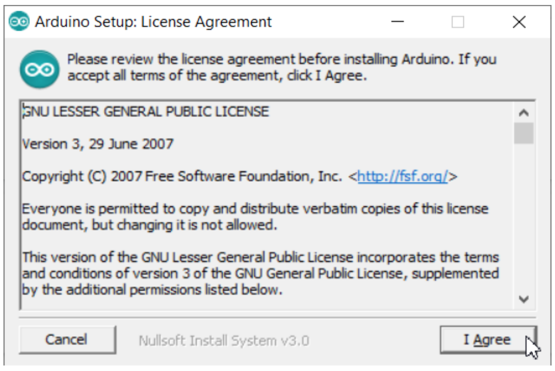
Active development of the Arduino software is **hosted by GitHub**. See the instructions for **building the code**. Latest release source code archives are available **here**. The archives are PGP-signed so they can be verified using **this** gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file
Windows app Win 8.1 or 10 Get
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits
Mac OS X 10.10 or newer

Release Notes Checksums (sha512)

2. Once downloaded, double-click the installer and follow the installation steps, until the installation is completed and close the installation wizard.



Arduino Setup: License Agreement

Please review the license agreement before installing Arduino. If you accept all terms of the agreement, click I Agree.

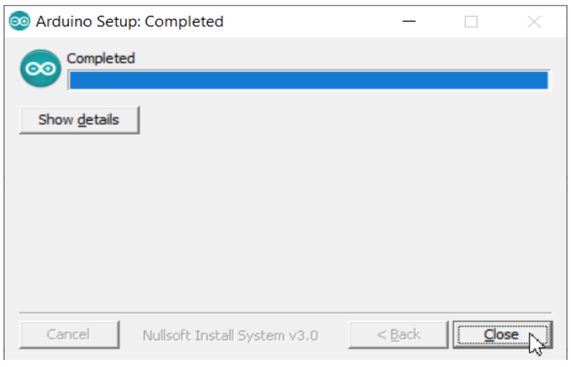
GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

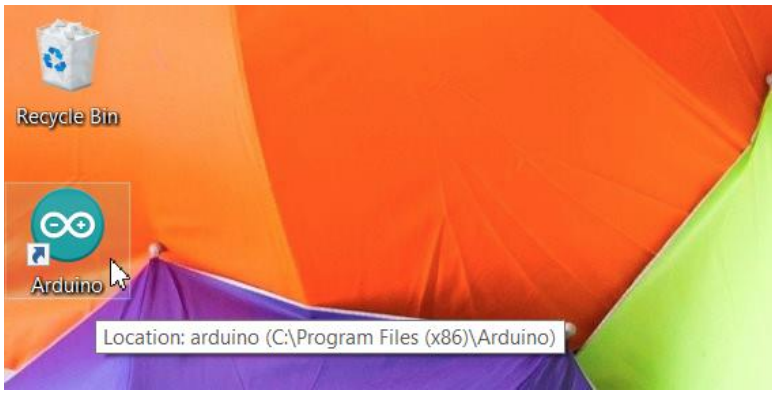
This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.



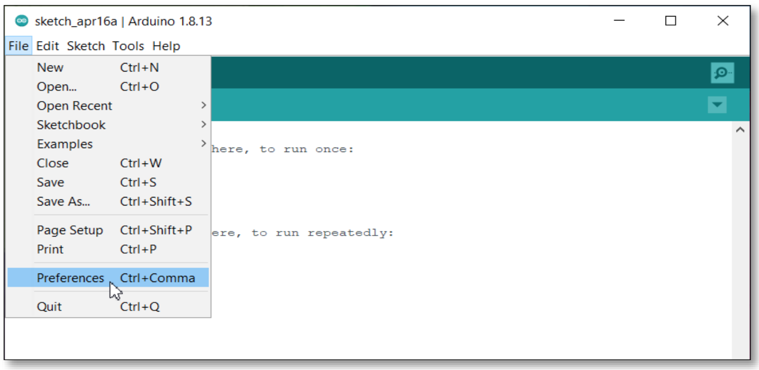
Arduino Setup: Completed

Completed

3. Open the Arduino IDE by double-clicking the shortcut icon, which the shortcut icon is available on the Windows Menu or on the Desktop.

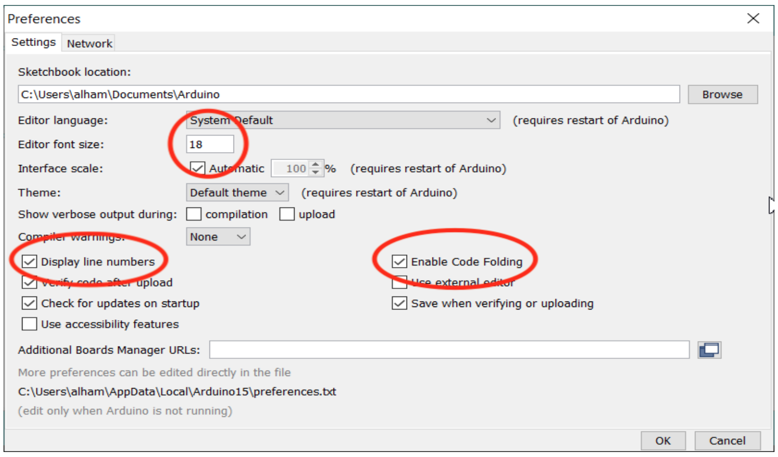


4. Once Arduino IDE is launched, open the Preferences window. From menu, ***File >> Preferences***.



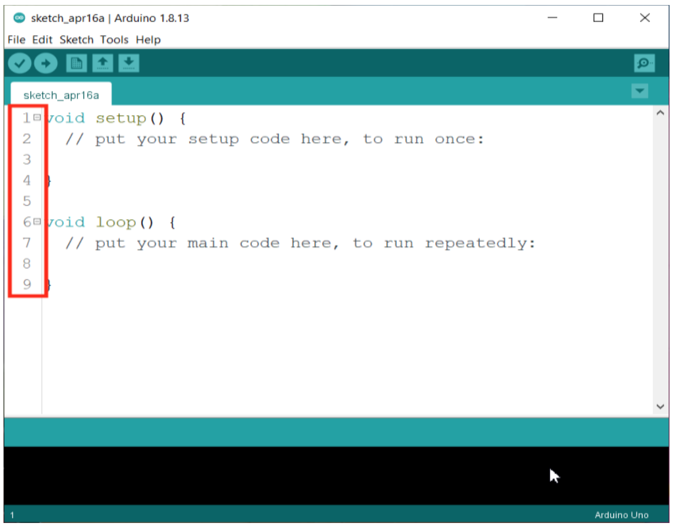
```
here, to run once:

ere, to run repeatedly:
```

5. On the Preferences window, increase the **Editor font size** by 18 or best views, check **Display line numbers** and **Enable sketch Folding**. Then click OK.



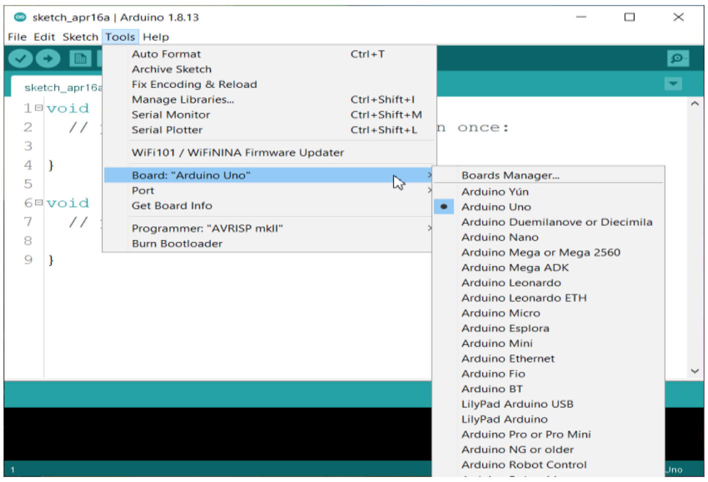
Sketchbook location:
C:\Users\alham\Documents\Arduino
Editor language: System Default (requires restart of Arduino)
Editor font size: 18
Interface scale: ☑ Automatic 100 % (requires restart of Arduino)
Theme: Default theme (requires restart of Arduino)
Show verbose output during: ☐ compilation ☐ upload
Compiler warnings: None
☑ Display line numbers ☑ Enable Code Folding
☑ Verify code after upload ☐ Use external editor
☑ Check for updates on startup ☑ Save when verifying or uploading
☐ Use accessibility features
Additional Boards Manager URLs:
More preferences can be edited directly in the file
C:\Users\alham\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

6. Arduino IDE text editor is now bigger, with line numbers and sketch folding option.

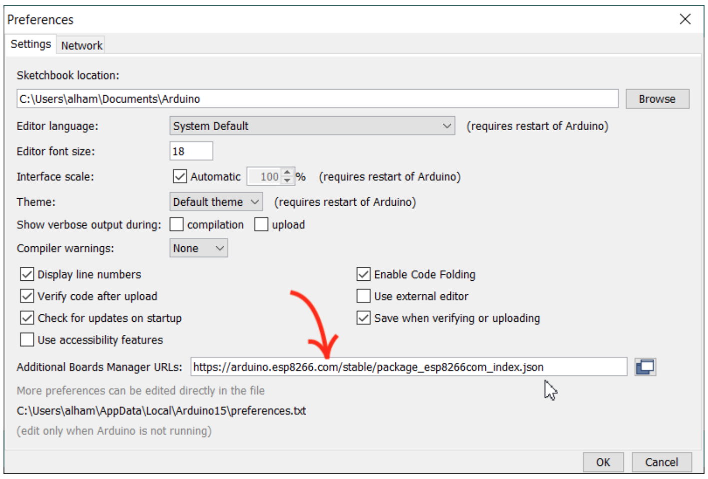


```
1 void setup() {
2   // put your setup code here, to run once:
3 
4 }
5 
6 void loop() {
7   // put your main code here, to run repeatedly:
8 
9 }
```

7. Check the available development board options on Arduino IDE. From menu, ***Tools >> Board: "name of development board" > "board selection"***. From the list of development board selection, only official Arduino boards are available.



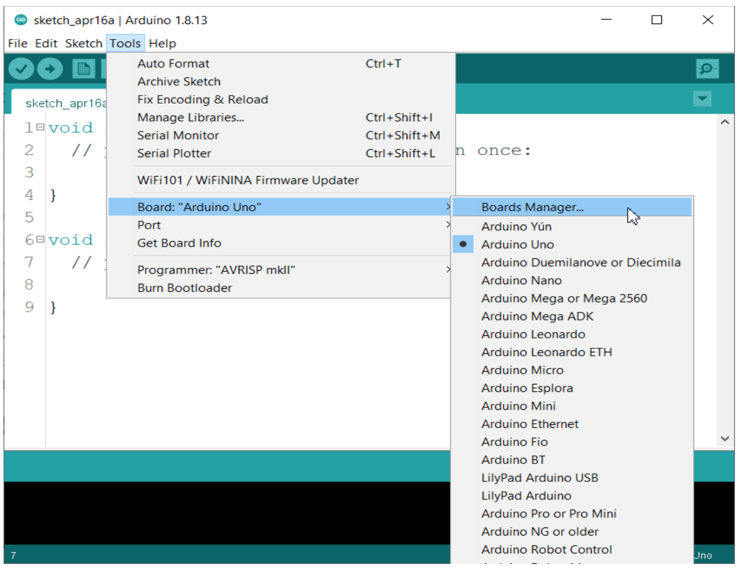
8. Add ESP8266 development boards into Arduino IDE. Open the Preferences Window and insert ESP8266 hardware library link below into **Additional Boards Manager URL** field. Then click OK. https://arduino.esp8266.com/stable/package_esp8266com_index.json


Preferences

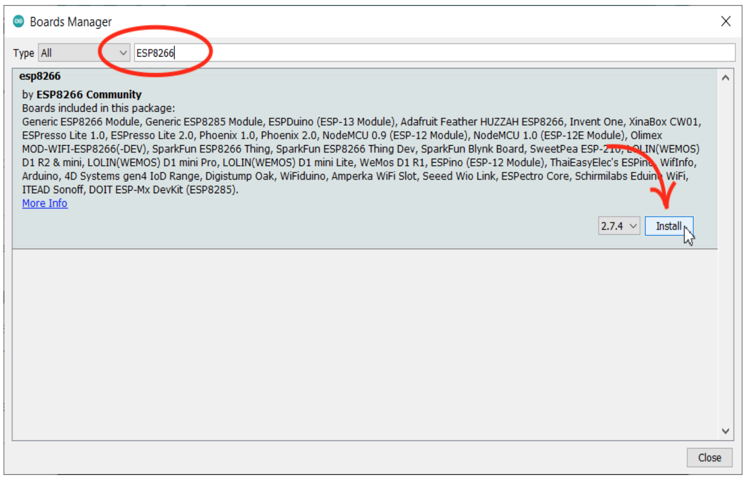
Settings | Network

Sketchbook location:
C:\Users\alham\Documents\Arduino [Browse]
Editor language: System Default (requires restart of Arduino)
Editor font size: 18
Interface scale: ☑ Automatic 100 % (requires restart of Arduino)
Theme: Default theme (requires restart of Arduino)
Show verbose output during: ☐ compilation ☐ upload
Compiler warnings: None
☑ Display line numbers ☑ Enable Code Folding
☑ Verify code after upload ☐ Use external editor
☑ Check for updates on startup ☑ Save when verifying or uploading
☐ Use accessibility features
Additional Boards Manager URLs: https://arduino.esp8266.com/stable/package_esp8266com_index.json
More preferences can be edited directly in the file
C:\Users\alham\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)
[OK] [Cancel]

9. Open Boards Manager window. From menu, ***Tools >> Board >> Boards Manager …***

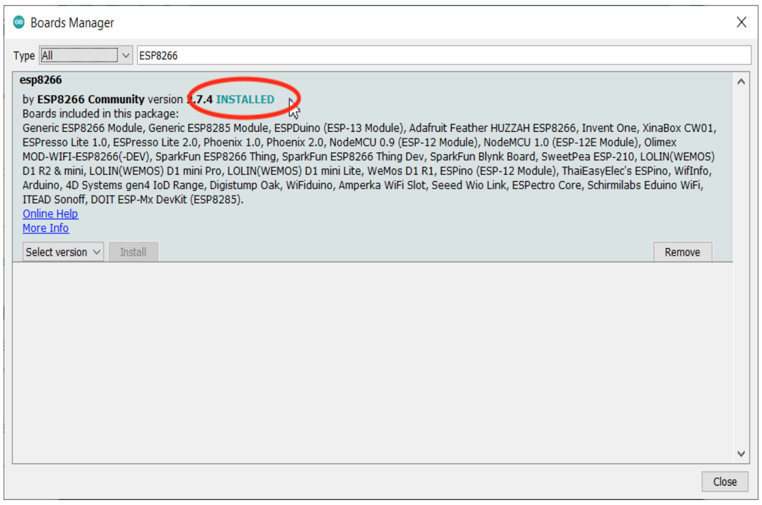


10. On the Boards Manager window, filter the list of development boards by typing ***ESP8266*** into the field and click the **Install** button to install the ESP8266 development boards hardware library on Arduino IDE.



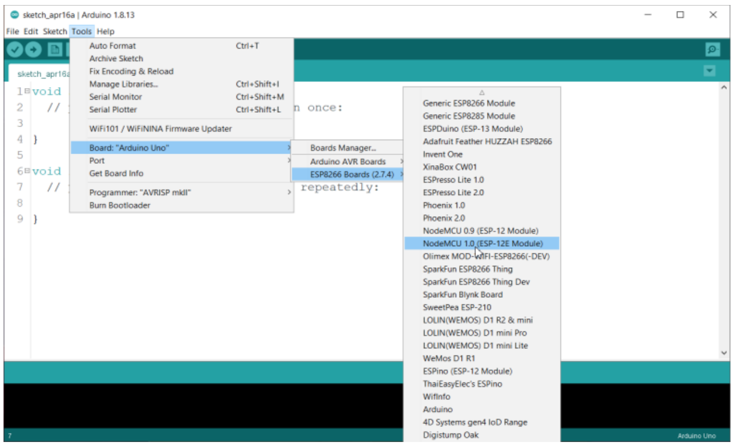
esp8266
by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).
More Info

11. Wait for the installation, which might take several minutes depending on the Internet connection speed. Once successfully **INSTALLED**, click the **Close** button.

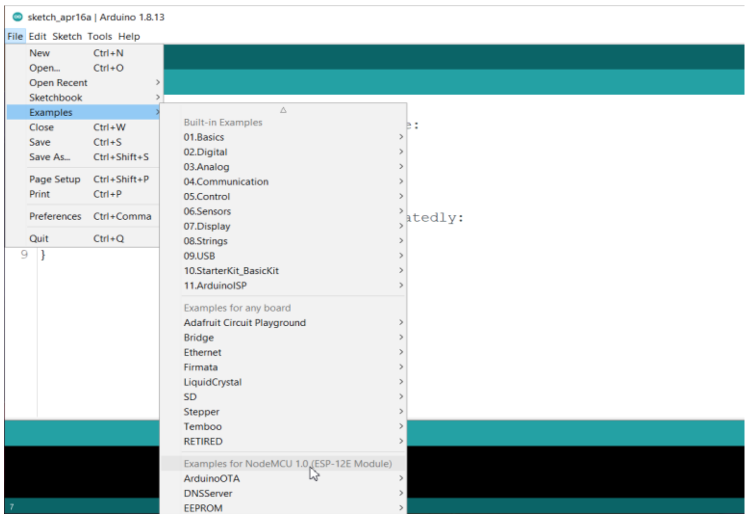


esp8266
by **ESP8266 Community** version **2.7.4** **INSTALLED**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, WeMos D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).
Online Help
More Info

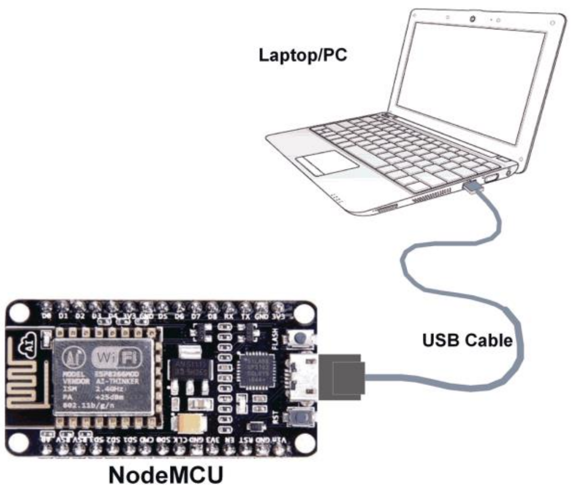
12. The development board is NodeMCU 1.0, thus selecting the right board for development. From menu, ***Tools >> Board: "name of development board" >> ESP8266 Boards (version) >> NodeMCU 1.0 (ESP-12E Module)***



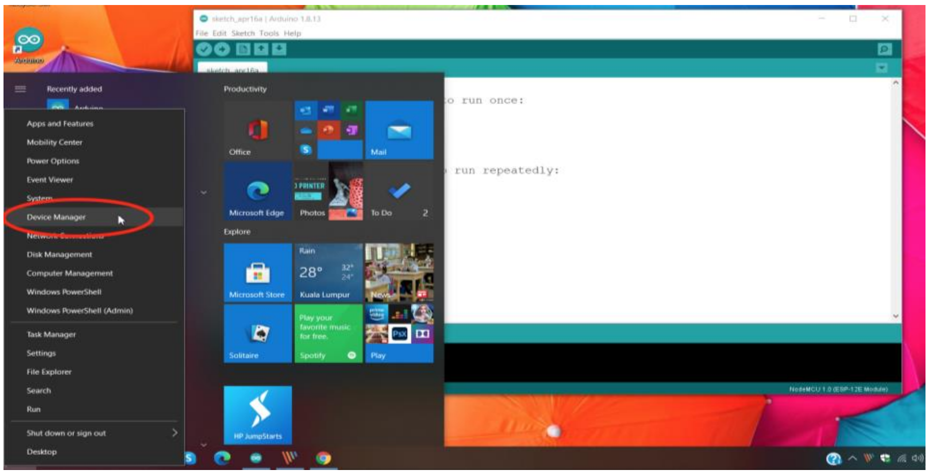
13. Now, we can see there are several available example sketches for ESP8266 on Arduino IDE. From menu, ***File >> Examples >> Examples for NodeMCU 1.0 (ESP-12E Module)***



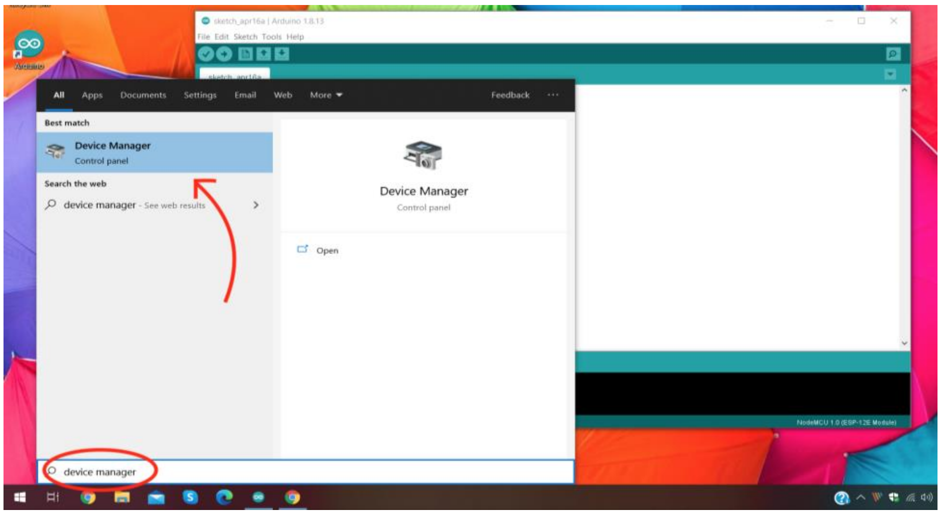
14. Connect the NodeMCU 1.0 to the computer using USB Cable Type A to Micro.



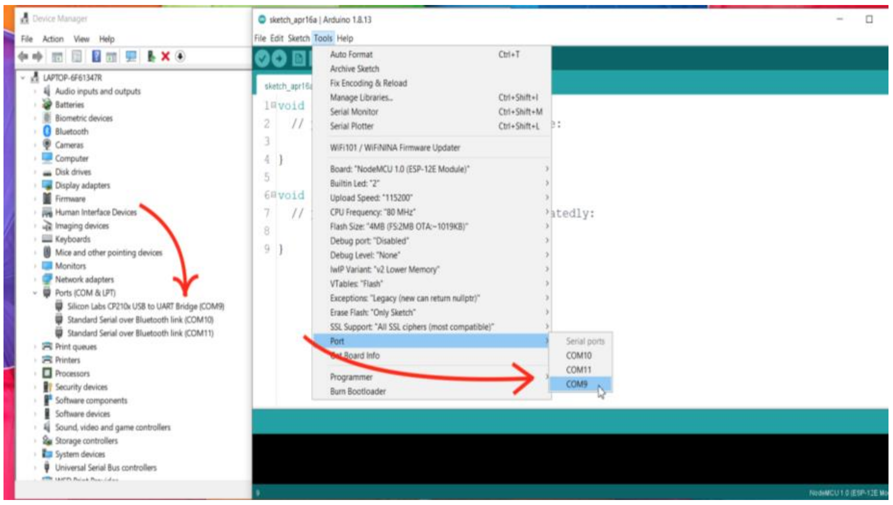
15. Now check on the Device Manager which COM port number, the NodeMCU is assigned on the computer. There are two ways to access Device Manager:
    a. Right-click Windows icon and select **Device Manager**



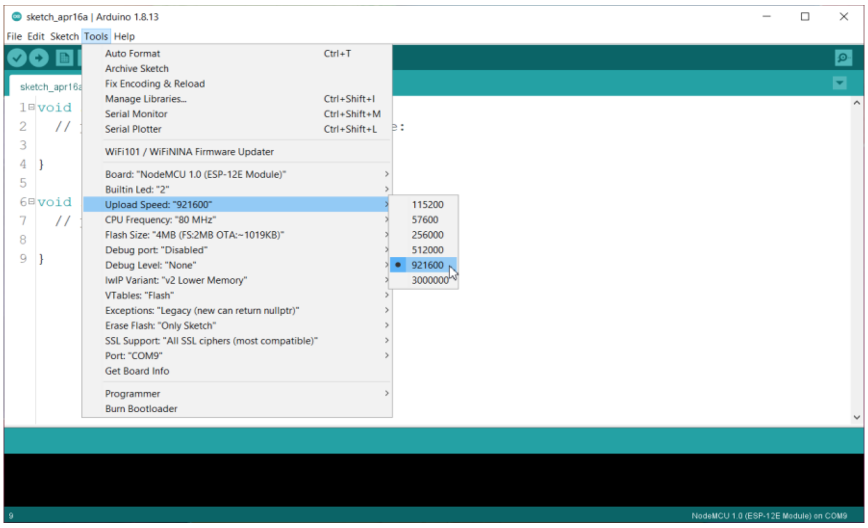
    b. Click on the Windows icon and type **Device Manager** to filter the list of applications.



16. On the Device Manager window, expand **Ports (COM & LPT)**, and look for **Silicon Labs CP210x USB to UART Bridge**. Note the COM port number next to it and inside the bracket.
17. Select the port on Arduino IDE. From menu, ***Tools >> Port >> COMx***

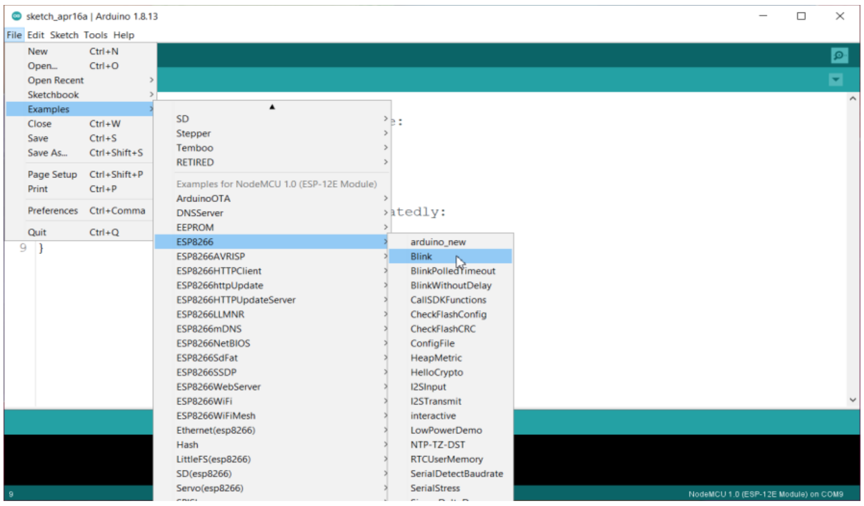

Ports (COM & LPT)
Silicon Labs CP210x USB to UART Bridge (COM9)
Standard Serial over Bluetooth link (COM10)
Standard Serial over Bluetooth link (COM11)
Board: "NodeMCU 1.0 (ESP-12E Module)"
Port
Serial ports
COM10
COM11
COM9

18. Set the **Upload Speed** to **921600**. From menu, ***Tools >> Upload Speed >> 921600***

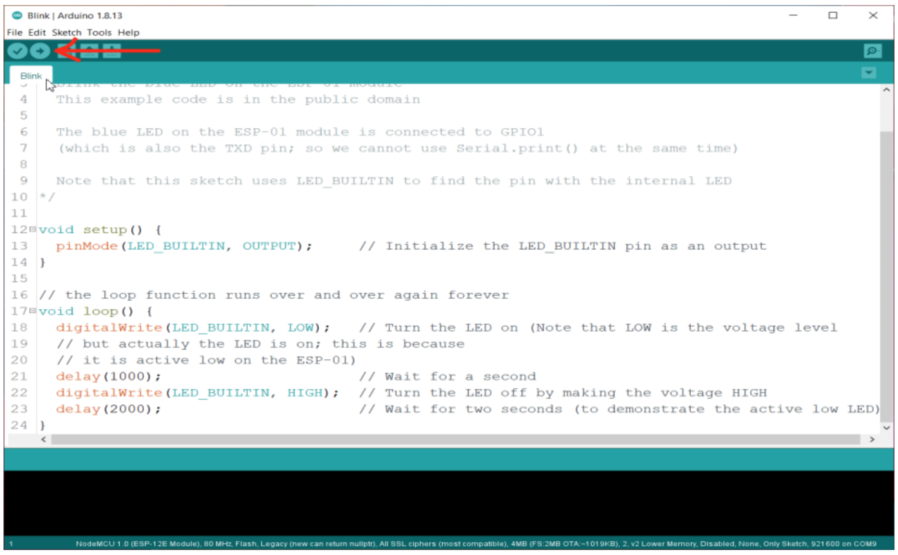

sketch_apr16a | Arduino 1.8.13
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Get Board Info
Programmer
Burn Bootloader
115200
57600
256000
512000
921600
3000000
NodeMCU 1.0 (ESP-12E Module) on COM9

```
Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "2"
Upload Speed: "921600"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
IwIP Variant: "v2 Lower Memory"
VTables: "Flash"
Exceptions: "Legacy (new can return nullptr)"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
Port: "COM9"
```

19. Now, test to program the NodeMCU 1.0 by uploading Blink example sketch. From menu, ***File >> Examples >> ESP8266 >> Blink***. Then Arduino IDE will open up a new window.



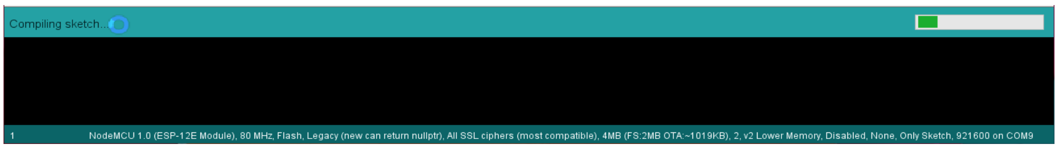
20. New window with Blink example sketch. Without changing anything on sketch, click the **Upload** button to upload the sketch to NodeMCU 1.0.



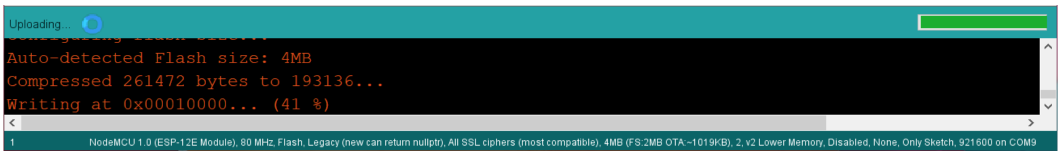
```
  This example code is in the public domain

  The blue LED on the ESP-01 module is connected to GPIO1
  (which is also the TXD pin; so we cannot use Serial.print() at the same time)

  Note that this sketch uses LED_BUILTIN to find the pin with the internal LED
*/

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);     // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, LOW);   // Turn the LED on (Note that LOW is the voltage level
  // but actually the LED is on; this is because
  // it is active low on the ESP-01)
  delay(1000);                      // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH);  // Turn the LED off by making the voltage HIGH
  delay(2000);                      // Wait for two seconds (to demonstrate the active low LED)
}
```

```
NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 921600 on COM9
```

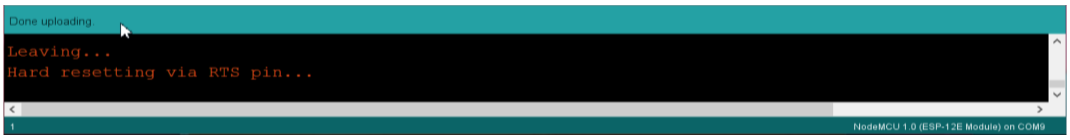
21. Wait for Arduino IDE to compile the sketch, upload the sketch and done uploading the sketch to NodeMCU 1.0.



```
Compiling sketch...
1    NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 921600 on COM9
```

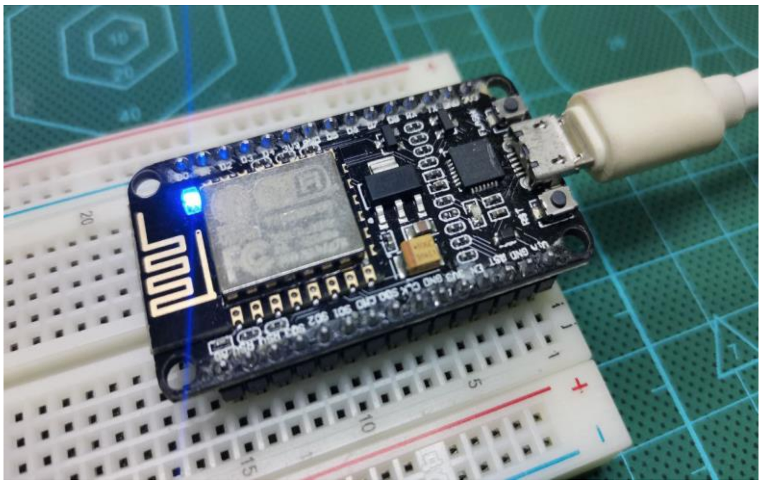


```
Uploading...
Auto-detected Flash size: 4MB
Compressed 261472 bytes to 193136...
Writing at 0x00010000... (41 %)
1    NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullptr), All SSL ciphers (most compatible), 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 921600 on COM9
```

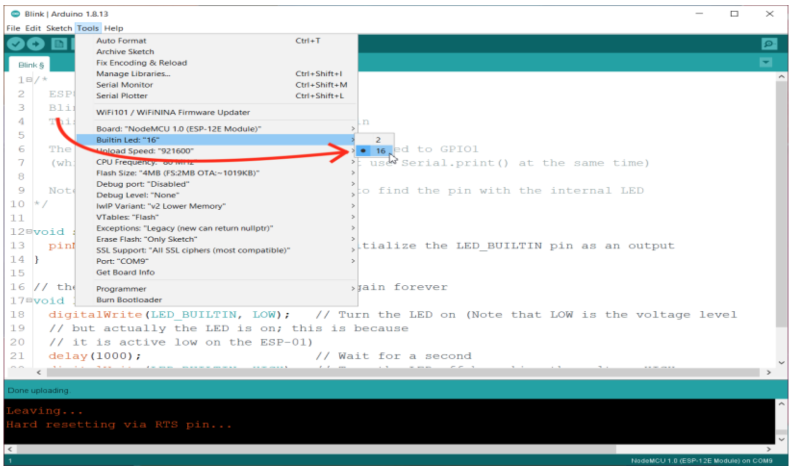


```
Done uploading.
Leaving...
Hard resetting via RTS pin...
1    NodeMCU 1.0 (ESP-12E Module) on COM9
```

22. Now we should see the on-board LED of NodeMCU 1.0 is blinking. The LED_BUILTIN LED is interfaced to GPIO2 of ESP8266.



23. Now, change the Builtin LED to GPIO16 on the Arduino IDE. From menu, **Tools >> Builtin Led >> 16**

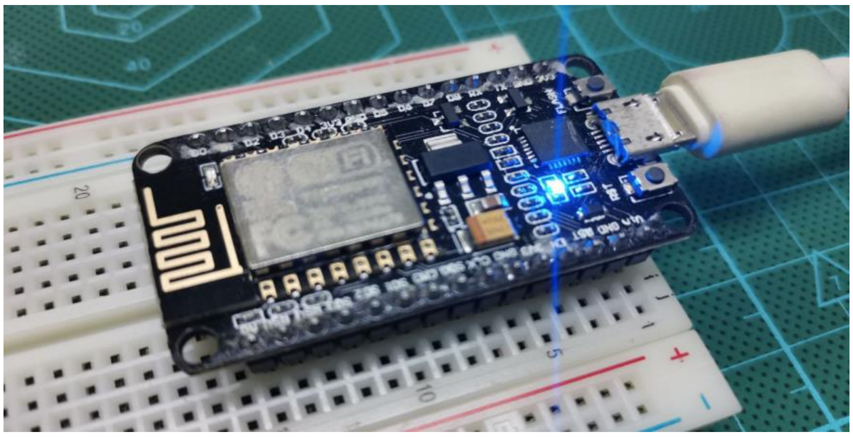


```
Blink | Arduino 1.8.13
File Edit Sketch Tools Help
Auto Format                     Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries...             Ctrl+Shift+I
Serial Monitor                  Ctrl+Shift+M
Serial Plotter                  Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "16"              > 2
                                 ● 16
Upload Speed: "921600"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
Exceptions: "Legacy (new can return nullptr)"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
Port: "COM9"
Get Board Info
Programmer
Burn Bootloader
```

```
  digitalWrite(LED_BUILTIN, LOW);   // Turn the LED on (Note that LOW is the voltage level
  // but actually the LED is on; this is because
  // it is active low on the ESP-01)
  delay(1000);                      // Wait for a second
```

```
Done uploading.
Leaving...
Hard resetting via RTS pin...
1    NodeMCU 1.0 (ESP-12E Module) on COM9
```

24. Upload the sketch and the different LED on-board LED on NodeMCU 1.0 is blinking, which interfaced to GPIO16 of ESP8266.



PART B: WRITING A PROGRAM (DELAY)

25. Following is the sketch for LED blinking.

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

26. Change the delay() function value from 1000 to 2000 and upload the sketch. Observe, what happens to the LED? Write the result in **Table 1**.

PART C: WRITING A PROGRAM (PIN MODE & DIGITAL WRITE)

27. Construct the following circuit using the given kit as **Figure C1**.

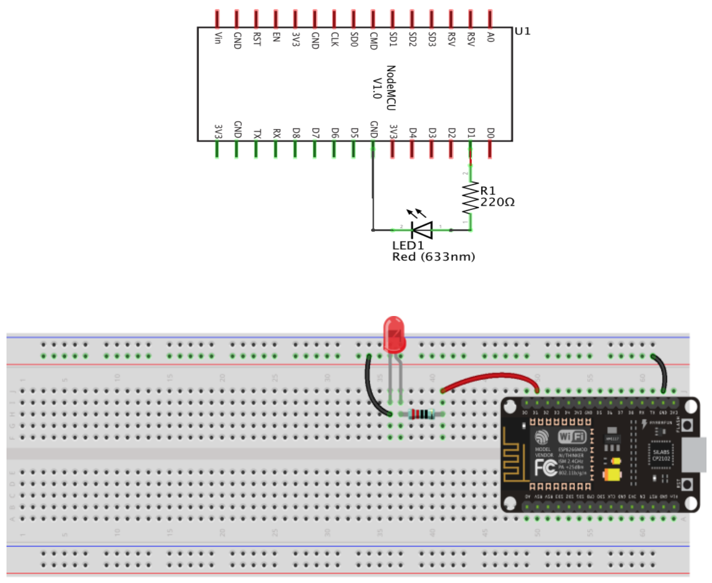
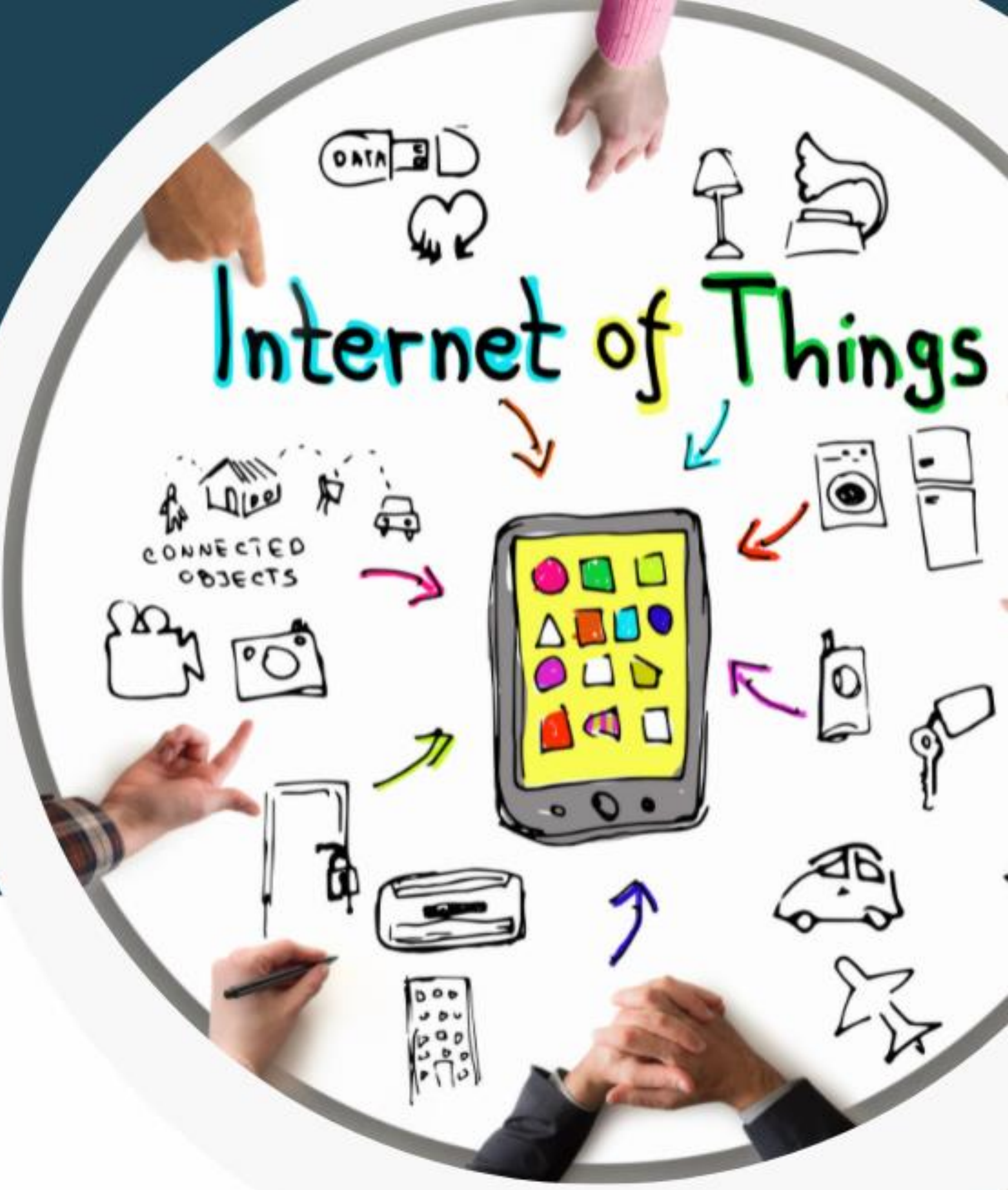


Figure C1

28. The red colour LED and 220Ω resistor (Red, Red, Black, Black, Brown) is connected to the Pin D1 of NodeMCU.
29. Change the word **LED_BUILTIN** to **D1** in the sketch.

```
void setup() {
  pinMode(D1, OUTPUT);
}

void loop() {
  digitalWrite(D1, HIGH);
  delay(1000);
  digitalWrite(D1, LOW);
  delay(1000);
}
```

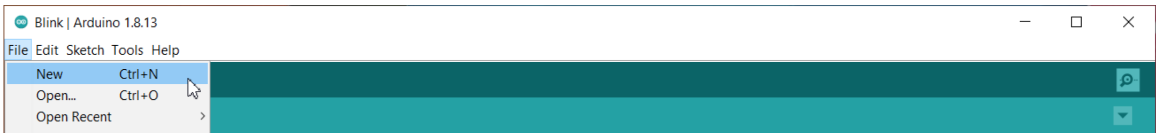
30. Upload the sketch.
31. Observe and explain the result in **Table 1**.



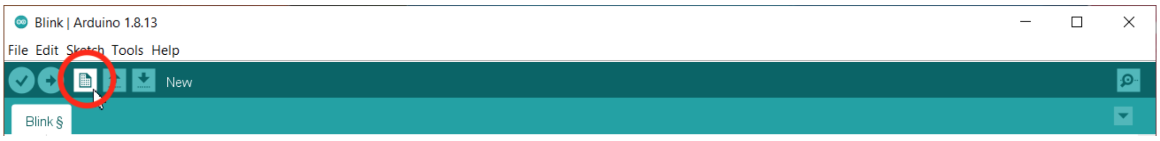
TUTORIAL 02

PART D: WRITING A PROGRAM (SERIAL MONITOR)

32. Create a new sketch file. There are 3 ways to create new file:
    a. Keyboard shortcut key, **Ctrl + N** or
    b. From menu, ***File >> New*** or



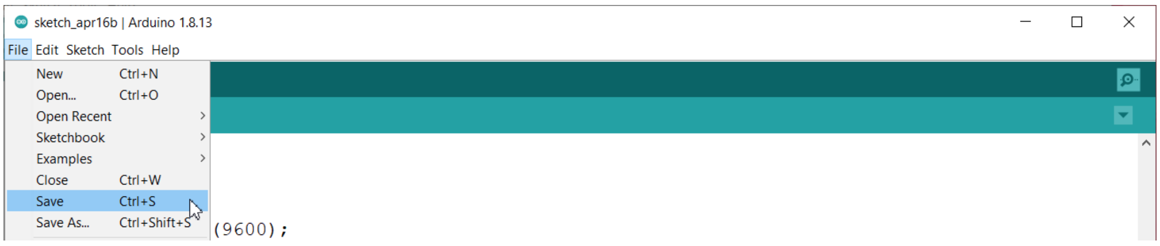
    c. Click New file icon.



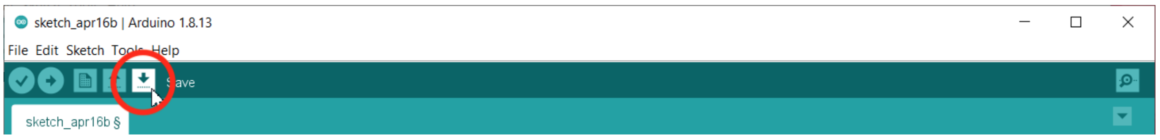
33. Write the following sketch and upload to NodeMCU.

```
int i = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.print("Time: ");
  Serial.print(i);
  Serial.println(" sec.");

  delay(1000);
  i = i + 1;
}
```

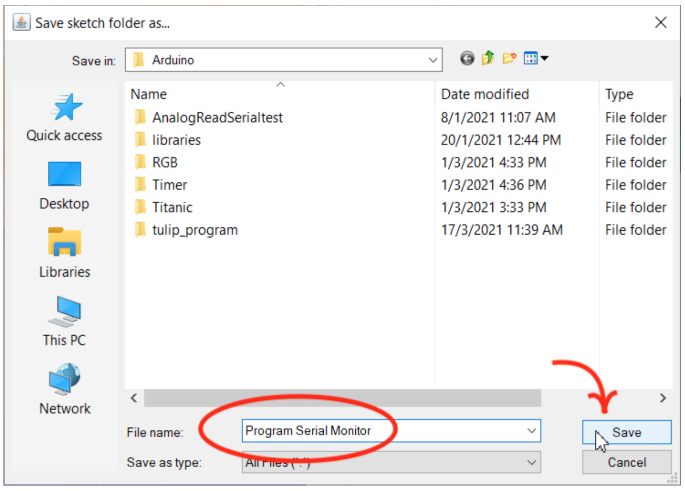
34. Save the sketch. There are 3 ways to save the sketch:
    a. Keyboard shortcut key, Ctrl + S or
    b. From menu, ***File >> Save*** or



    c. Click New file icon.



35. The default folder to save an Arduino sketch is in ***Documents >> Arduino*** folder. Type the name of the sketch and click the Save button.



Save sketch folder as...

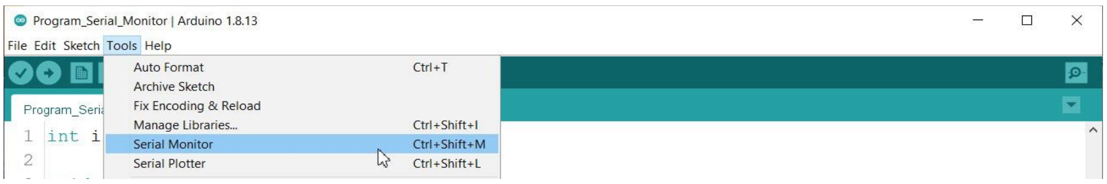
Save in: Arduino

Name	Date modified	Type
AnalogReadSerialtest	8/1/2021 11:07 AM	File folder
libraries	20/1/2021 12:44 PM	File folder
RGB	1/3/2021 4:33 PM	File folder
Timer	1/3/2021 4:36 PM	File folder
Titanic	1/3/2021 3:33 PM	File folder
tulip_program	17/3/2021 11:39 AM	File folder

File name: Program Serial Monitor

Save as type: All Files (*.*)

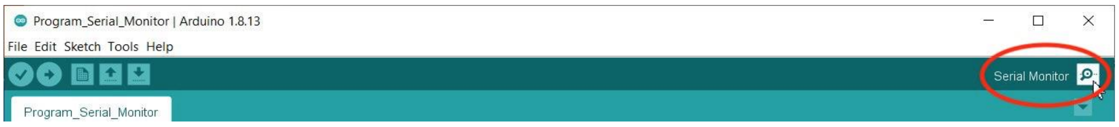
36. Once the file is saved, upload the sketch and open the Serial Monitor. There 3 ways to open the Serial Monitor.
    a. Keyboard shortcut key, **Ctrl + Shift + M** or
    b. From menu, ***File >> Save*** or



Program_Serial_Monitor | Arduino 1.8.13

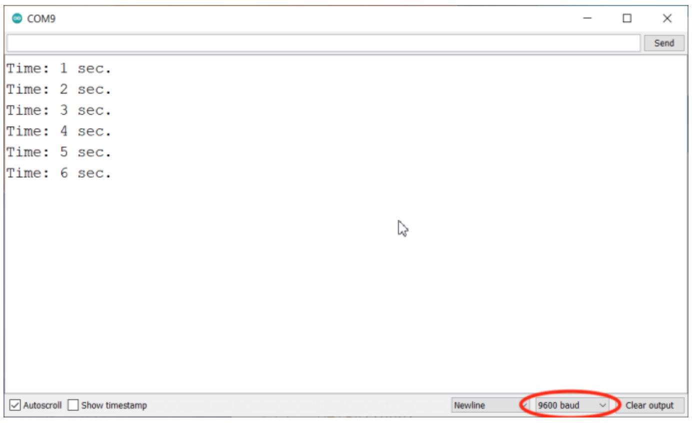
Auto Format	Ctrl+T
Archive Sketch	
Fix Encoding & Reload	
Manage Libraries...	Ctrl+Shift+I
Serial Monitor	Ctrl+Shift+M
Serial Plotter	Ctrl+Shift+L

    c. Click the Serial Monitor icon.



Program_Serial_Monitor | Arduino 1.8.13

37. Make sure the Serial Monitor baud-rate is set to 9600 baud, the same as the program.



COM9

```
Time: 1 sec.
Time: 2 sec.
Time: 3 sec.
Time: 4 sec.
Time: 5 sec.
Time: 6 sec.
```

Autoscroll, Show timestamp, Newline, 9600 baud

38. Press the **RST** (reset) button on the NodeMCU once. Observe the results on the Serial Monitor. Print screen the result and paste it in **Table 1**.

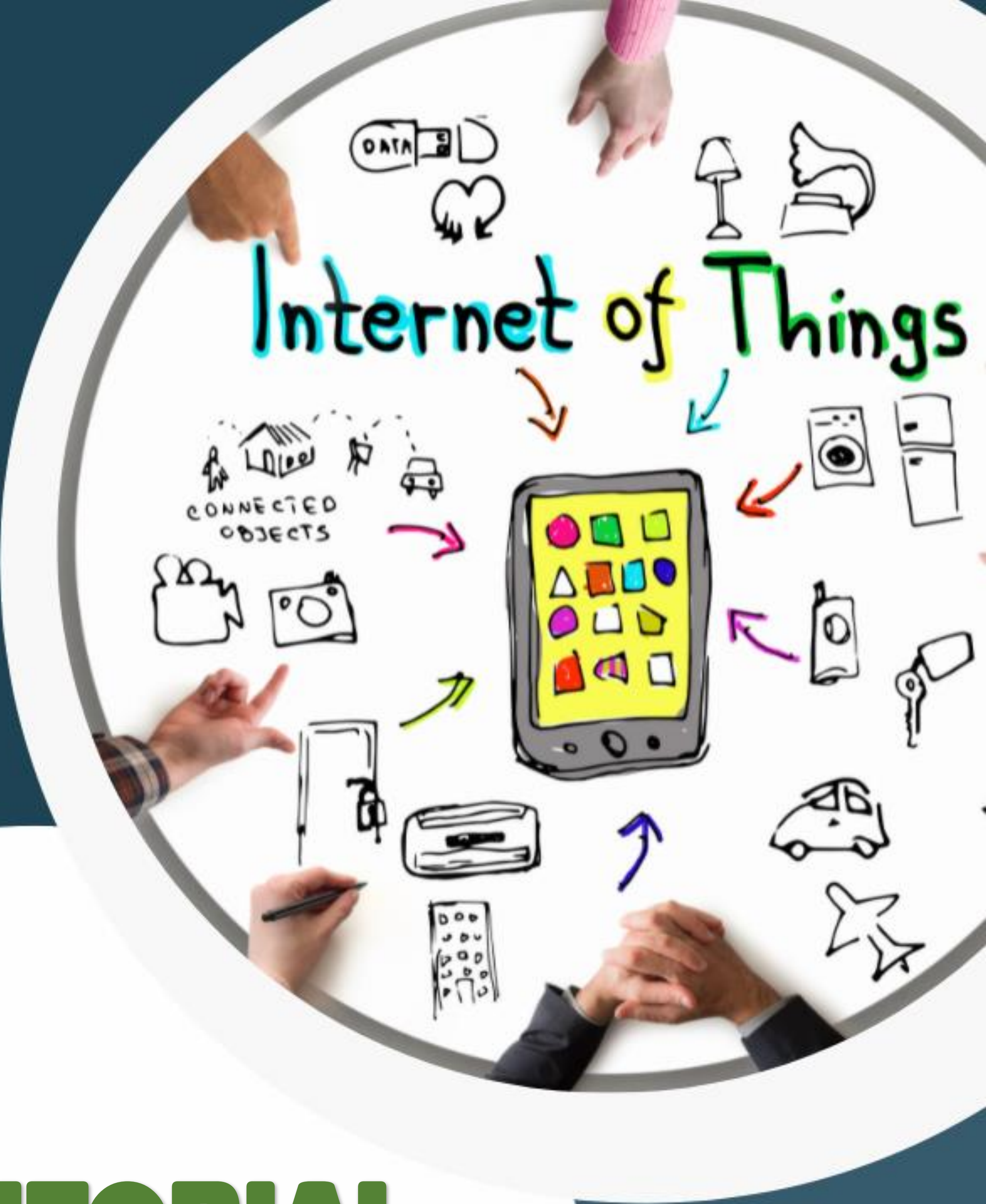
RESULT

Part B: Writing A Program (Delay) Delay change from 1000 to 2000.What happen to the LED?
Part C: Writing A Program (Pin Mode & Digital Write) LED_BUILTIN change to D1. What is the effect of changes?
Part C : Writing A Program (Serial Monitor) Print screen the result here. What is the observation of the display at serial monitor?

DISCUSSION

1. Explain the function of the following coding

pinMode (pin, MODE)	
delay ( )	
Serial.begin (baudrate)	
digitalWrite ( )	
Serial.print ( );	
Serial.println ( );	



TUTORIAL 02

TUTORIAL: 2

Title: Sensor and Actuator

OBJECTIVES

Upon completion of this tutorial, you should be able to:

- Write code to read data from digital sensor
- Write code to read data from analogue sensor
- Write code to control actuator
- Construct hardware connection

EQUIPMENTS

1. Node MCU ESP8266
2. LED
3. Breadboard
4. Push Button
5. Jumper cables
6. Transistor 2N2222
7. Resistor 220 Ohm
8. DC Motor
9. LDR
10. Arduino IDE
11. Resistor 10 K Ohm (brown, black, orange, gold)

THEORY

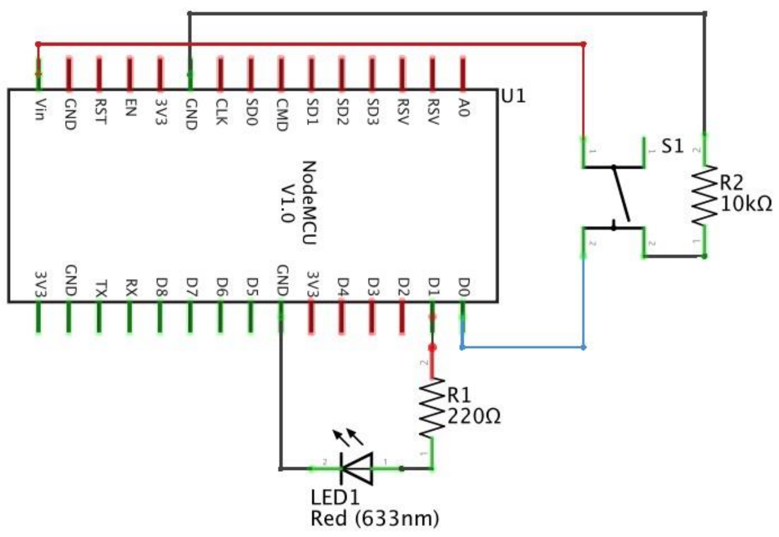
Sensor are electronic devices that detect and respond to changes in an environment while an actuator is a device that produces a motion by converting energy and signals going into the system. The motion it produces can be either rotary or linear. An **IoT** system consists of sensors which “talk” to the cloud via some kind of connectivity. Once the data gets to the cloud, software processes it and then might decide to perform an action, such as sending an alert or automatically adjusting the **actuator** without the need for the user.

Two types input of sensor there are analog input and digital input. In this practical work, the pushbutton is use as a digital input where LED and DC Motor use as a digital output. Sensor LDR use as an analog input and the serial monitors as an analog output.

PROCEDURE

PART A: DIGITAL INPUT (PUSHBUTTON) & DIGITAL OUTPUT (LED)

1. Construct the following circuit using the given kit as **Figure A1**.



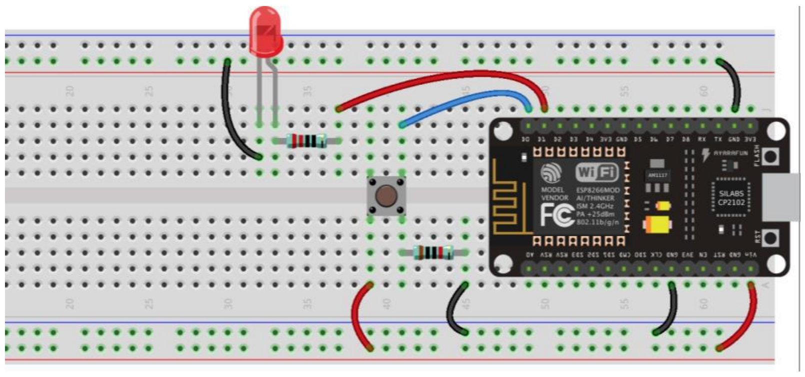


Figure A1

2. Write the following sketch in Arduino IDE and upload the sketch to NodeMCU.

```
int led = 5; // LED Pin (D1)
int button = 16; // Pushbutton is connected to D0
int temp = 0; // Temporary variable for reading the
button pin status

void setup() {
  Serial.begin(9600);
  pinMode(led,
  OUTPUT);
  pinMode(button,
  INPUT);


void loop() {
  // declare LED as output
  // declare push button as input
  temp = digitalRead(button);

  if (temp == HIGH) {
    digitalWrite(led, HIGH);
    Serial.println("LED Turned ON");
    delay(1000);
  }

  else {
    digitalWrite(led, LOW);
    Serial.println("LED Turned OFF");
    delay(1000);
  }
}
```

3. Press and release the pushbutton and write the result in **Table A1**.

PART B: NODEMCU CONNECTED TO ACTUATOR (DC MOTOR)

4. Now change the digital output (LED) to a DC Motor interface to a Transistor 2N2222. The circuit is shown in **Figure B1**.

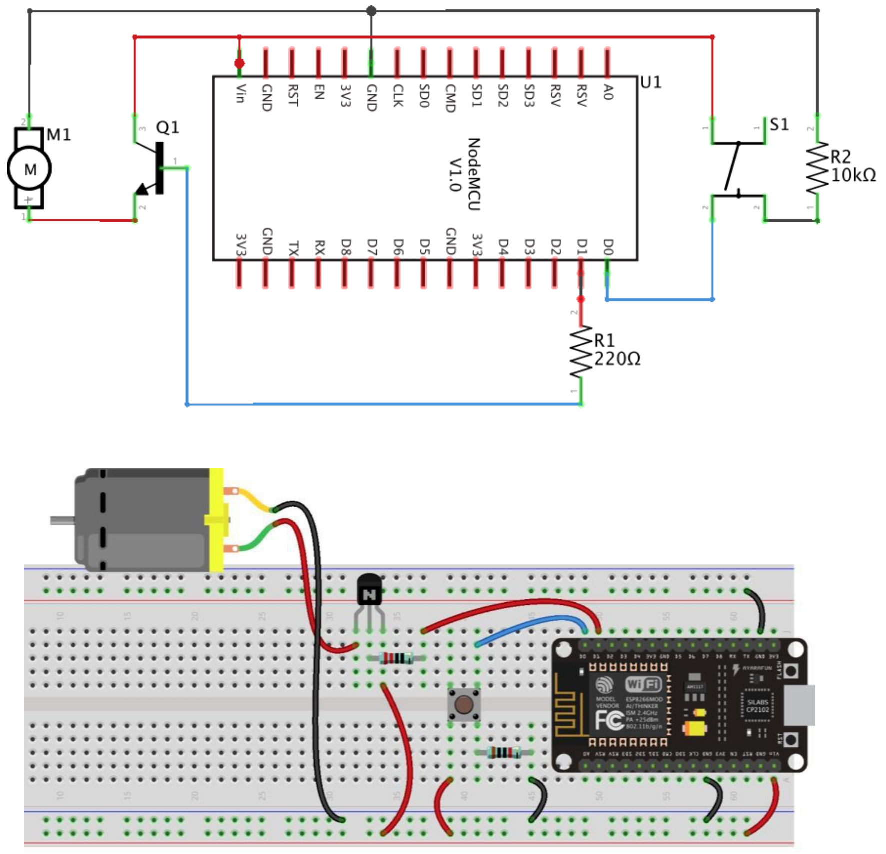


Figure B1

5. The connection of the Transistor 2N2222 which replaced the LED as follows:
   a) Transistor 2N2222 Base is connected to 220Ω resistor which interfaced to D1 NodeMCU.
   b) The Transistor 2N2222 Collector is connected to the power source VIN of NodeMCU.
   c) The Transistor 2N2222 Emitter is connected to +ve terminal of DC motor, and -ve terminal of DC motor is connected to GND.

6. Press & release the pushbutton and observe the result and write it in Table B1.
7. Observe that the motor will rotate if you press the pushbutton and it will stop if you release the push button. This condition is known as Active High.
8. Now, modify the sketch so that the motor will rotate if you release the button and it will stop if you push the button. Write the new sketch in discussion.

PART C: ANALOG INPUT – LIGHT DEPENDENT RESISTOR (LDR)

9. Construct the following circuit using the given kit as **Figure C1**.

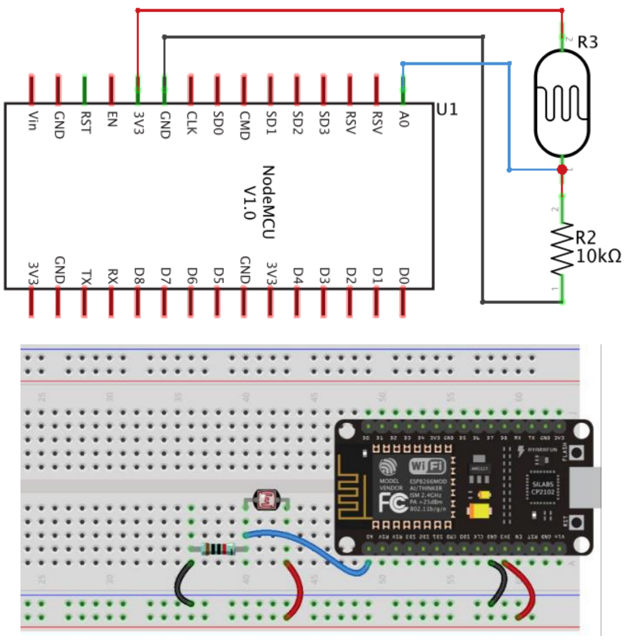


Figure C1

10. Write the following sketch that will monitor the output voltage of the LDR through the Serial Monitor

```
void setup() {
   Serial.begin(9600);
}

void loop() {
   int sensorValue = analogRead(A0);
   float sensorVoltage = sensorValue * (3.3 / 1024.0);

   Serial.print("ADC Value:
   ");
   Serial.print(sensorValue)
   ;
   Serial.print("\tVoltage:
   ");
   Serial.print(sensorVoltag
   e); Serial.println(" V");

   delay(1000);
}
```

11. Upload the sketch to the NodeMCU and open the Serial Monitor.
12. Observe the voltage reading on the Serial Monitor and write it down in Table C1 based on the given scenarios.

RESULT

PART A: DIGITAL INPUT (PUSHBUTTON) & DIGITAL OUTPUT (LED)

Action	LED Condition
Press Push Button	
Release Push Button	

Table A1

PART B: NODEMCU CONNECTED TO ACTUATOR (DC MOTOR)

Action	Motor Condition
Press Push Button	
Release Push Button	

Table B1

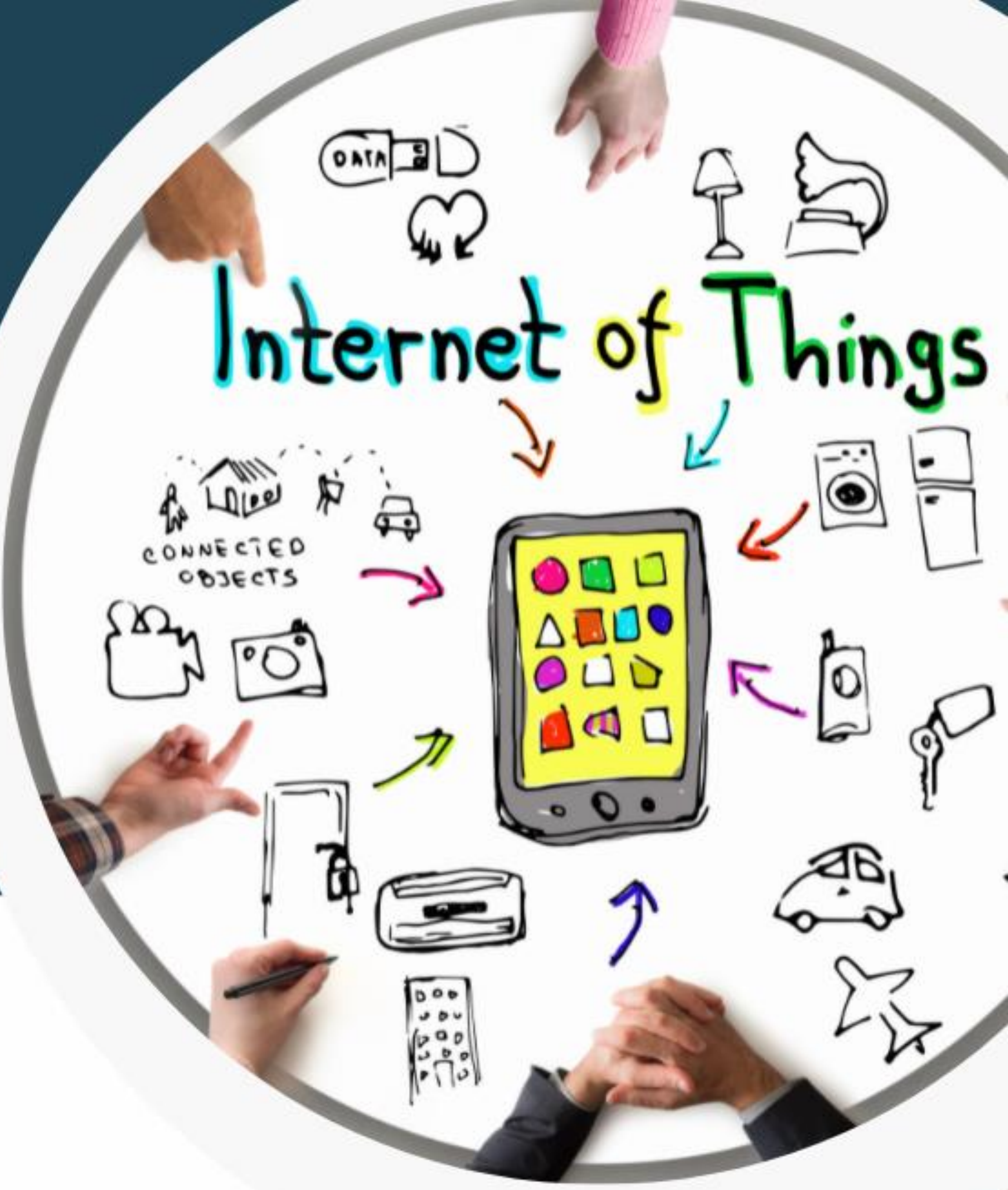
PART C: ANALOG INPUT – LIGHT DEPENDENT RESISTOR (LDR)

Action	Vout Value
Exposed LDR to room light	
Close the top surface of the LDR with your finger	
Torch a light from your handphone to LDR	
Turn Off the light of the room	

Table C1

DISCUSSION

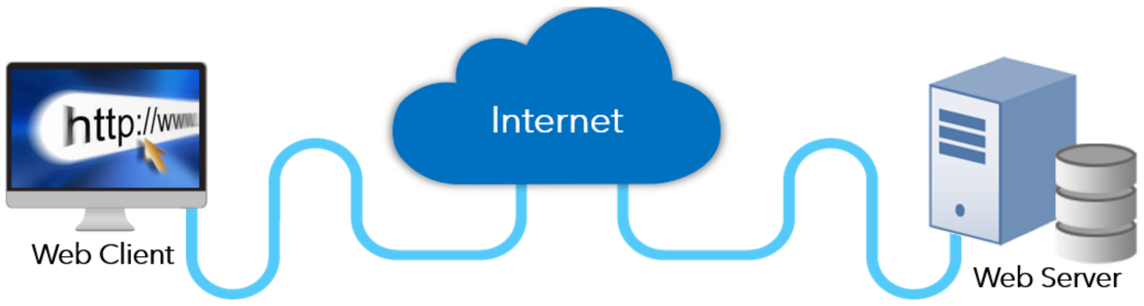
1. What is the different between analog and digital input/output?
2. Based on tutorial in Part B, if we want to change the direction of motor rotation, what should we do? Discuss the answer.
3. Write the new coding for procedure Part B (5).



TUTORIAL 03

TUTORIAL: 3

Title: NodeMCU ESP8266 using HTTP Protocol



OBJECTIVES

Upon completion of this tutorial, you should be able to:

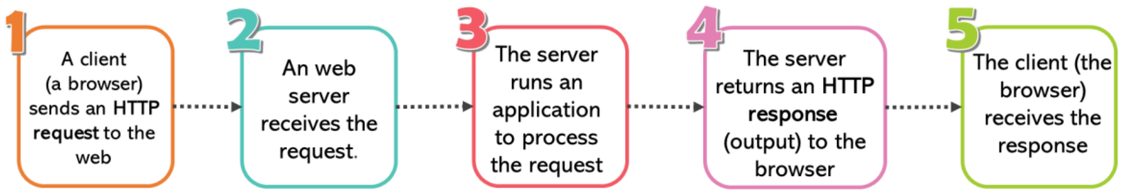
- Write codes to establish connection to internet
- To controls output from web browser
- Display data from sensor to web browser
- Display data from sensor to mobile application

EQUIPMENTS

- NodeMCU ESP8266
- Breadboard
- Jumper cables
- Resistor 220 Ω (Red, Red, Brown, Gold)
- Arduino IDE
- Resistor 10 K Ω (brown, black, orange, gold)
- LED
- LDR
- Internet Connection

THEORY

HTTP stands for Hyper Text Transfer Protocol. This protocol is the fundamental client-server model protocol used for the Web, and the one most compatible with existing network infrastructure, used by the web developers on a daily basis. **WWW** is about communication between web **clients** and **servers** Communication between **client computers and web servers** is done by sending **HTTP Requests** and receiving **HTTP Responses.** Figure below show the protocol step for request-response communication of machines in server-client mode.


1
A client (a browser) sends an HTTP **request** to the web
2
An web server receives the request.
3
The server runs an application to process the request
4
The server returns an HTTP **response** (output) to the browser
5
The client (the browser) receives the response

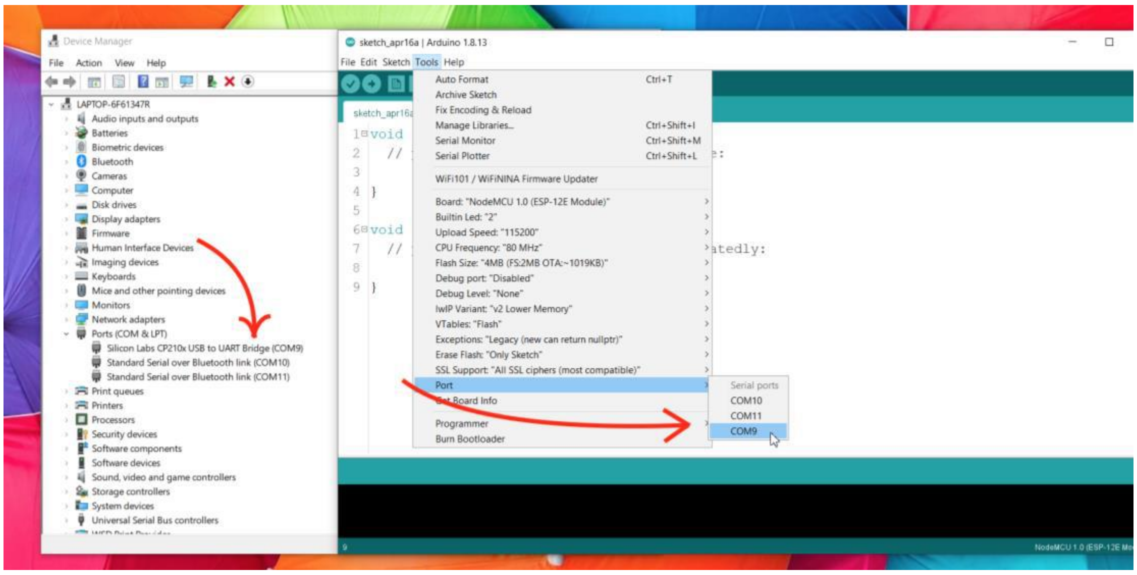
HTTP is widely used in IOT application like smart farming, healthcare, smart grid, smart home and etc that can access to control the sensor over the internet.

PROCEDURE

PART A: CONTROL OUTPUT FROM WEB BROWSER

ESP8266 Board Configuration

1. Open Arduino IDE platform.
2. From menu, ***Tools >> Board: "name of development board" >> ESP8266 Boards (version) >> NodeMCU 1.0 (ESP-12E Module)*** and from menu, ***Tools >> Port >> COMx*** select the correct Port depends on Ports (COM & LPT) as shown in Device Manager (Figure A1)

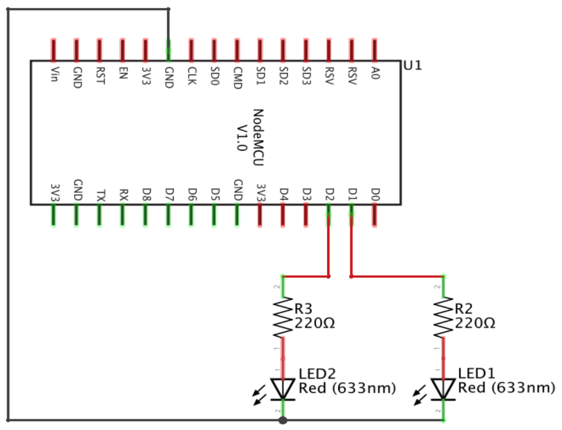


```
Ports (COM & LPT)
    Silicon Labs CP210x USB to UART Bridge (COM9)
    Standard Serial over Bluetooth link (COM10)
    Standard Serial over Bluetooth link (COM11)

sketch_apr16a | Arduino 1.8.13
Tools
Board: "NodeMCU 1.0 (ESP-12E Module)"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Flash Size: "4MB (FS:2MB OTA:~1019KB)"
Debug port: "Disabled"
Debug Level: "None"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
Exceptions: "Legacy (new can return nullptr)"
Erase Flash: "Only Sketch"
SSL Support: "All SSL ciphers (most compatible)"
Port > Serial ports: COM10, COM11, COM9
```

Figure A1

Hardware Connection

3. Construct the following circuit with 2x LEDs, 2x 220Ω resistors which interfaced to NodeMCU 1.0 as shown in Figure A2.



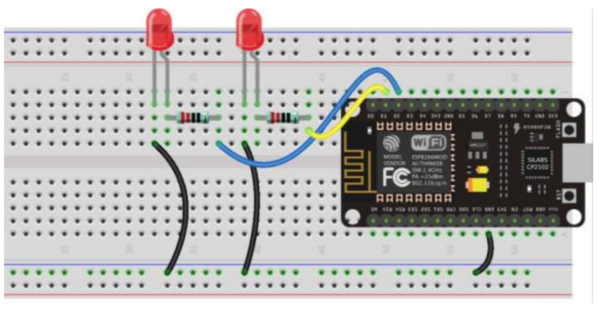


Figure A2

Writing Arduino Sketch

4. Write a new sketch as in Appendix 1.
5. From the sketch replace the variable of ssid and password respectively, YOUR_SSID and YOUR_PASSWORD with your network credentials as shown in Figure A3.

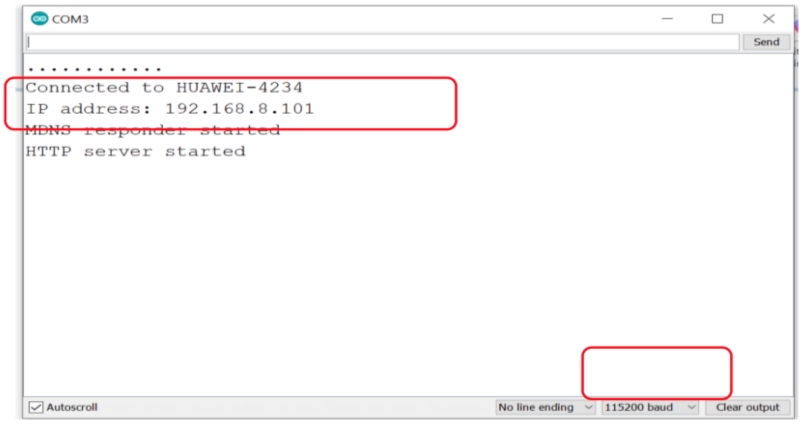
```
// Replace with your network credentials
const char* ssid = "YOUR_SSID";
const char* password = "YOUR_PASSWORD";
```

Figure A3

6. Click the "Upload Button" on the Arduino IDE and wait a few seconds until the "Done uploading" message is shown in the bottom left corner.]\
7. Observe if there are any errors while compiling and uploading.

Getting ESP8266 IP Address

8. Open Arduino IDESerial Monitor and set the baud rate to 115200 baud as shown in Figure A4.



```
............
Connected to HUAWEI-4234
IP address: 192.168.8.101
MDNS responder started
HTTP server started
```

No line ending
115200 baud

Figure A4

9. After a few seconds, your IP address should appear.
10. Write down the IP Address in Table 1 as a result.

Connect to Web Server

11. Open any browser from your computer.
12. Type the IP address of NodeMCU 1.0 as in the Serial Monitor and click Enter.
13. Print screen the layout from the browser and paste in Table 1 as a result.

Control LEDs from Web Server

14. Click at Socket #1 ON/OFF to turn on or off LED1 and click Socket #2 ON/OFF to turn on or off LED2.
15. Observe what happens to the LED when you click ON and OFF button at browser and write down your observation as result in Table 1.

PART B: DISPLAY DATA FROM SENSOR TO WEB BROWSER

Setting Up ThingSpeak Account

16. Go to https://www.thingspeak.com and Sign Up / Sign In.
17. Under My Channels, click New Channel as in Figure B1.

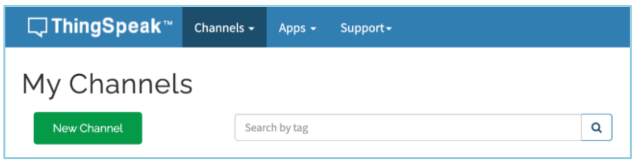
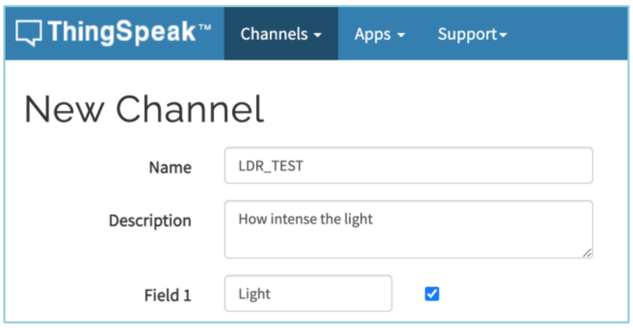


Figure B1

18. Give your channel a name, and enable one field(s) a Figure B2.



New Channel

Name: LDR_TEST

Description: How intense the light

Field 1: Light ☑

Figure B2

19. Click Save at the bottom of the page.

20. Click the API Keys tab as shown in Figure B3.

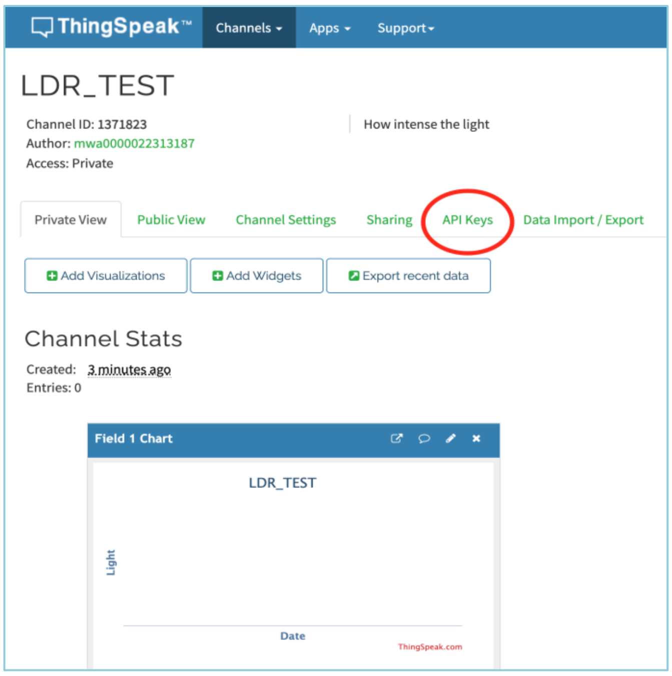


Figure B3

21. API keys will appear as in Figure B4.

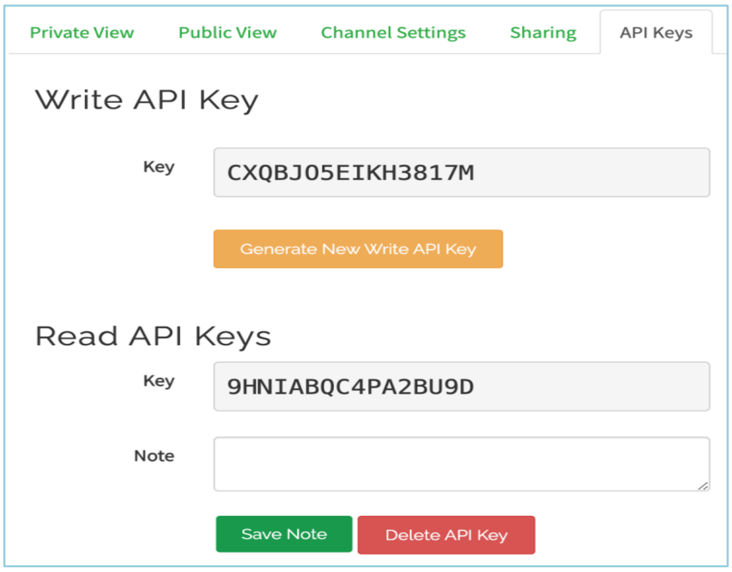


Figure B4

22. Write down your Channel ID and Write API Key at Table 2 as a result.

Hardware Installation

23. Construct the following circuit of light sensor (LDR) as shown in Figure B5.

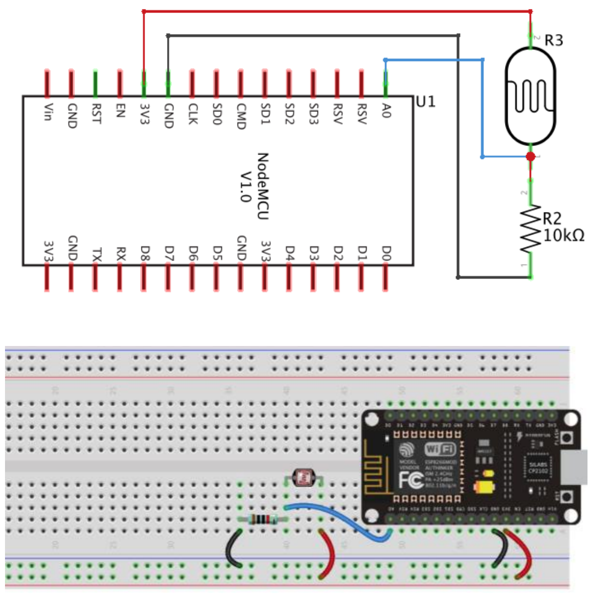
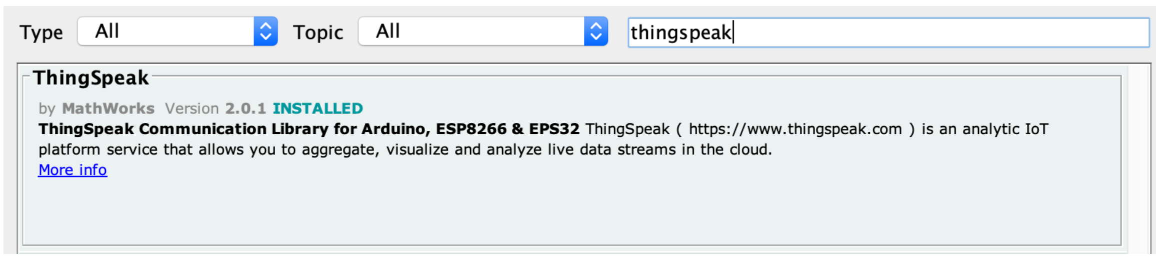


Figure B5

Install Thingspeak Library

24. From menu, ***Tools >> Manage Libraries***...
25. Type **ThingSpeak** to filter the list of available libraries, then click Install as in Figure B6.



Type All Topic All thingspeak

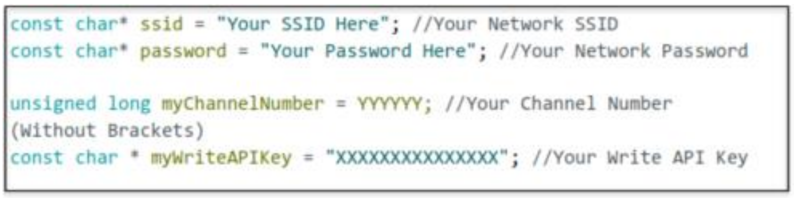
ThingSpeak

by **MathWorks** Version **2.0.1** **INSTALLED**
ThingSpeak Communication Library for Arduino, ESP8266 & EPS32 ThingSpeak (https://www.thingspeak.com) is an analytic IoT platform service that allows you to aggregate, visualize and analyze live data streams in the cloud.
More info

Figure B6

Upload Sketch

26. Write a new sketch as in Appendix 2.
27. Replace the following variable in the sketch with your network information and Thingspeak API Key and channel number.



```
const char* ssid = "Your SSID Here"; //Your Network SSID
const char* password = "Your Password Here"; //Your Network Password

unsigned long myChannelNumber = YYYYYY; //Your Channel Number
(Without Brackets)
const char * myWriteAPIKey = "XXXXXXXXXXXXXX"; //Your Write API Key
```

28. Then verify and upload the sketch.
29. Now we can display the data from sensor to web browser.
30. Print screen the display as a result in Table 2.

PART C: DISPLAY DATA FROM SENSOR TO MOBILE APPS

Install ThingSpeak App

31. Go to Google Play or App Store on your smartphone and install ThingView – ThingSpeak viewer and open ThingView application.

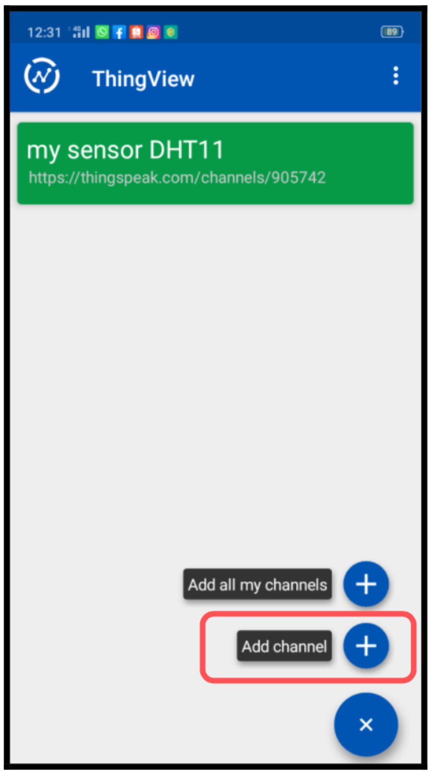


Figure C1

32. Click + icon to Add channel as shown in Figure C1.
33. Insert the Channel ID and API Key. Then click the Search button.

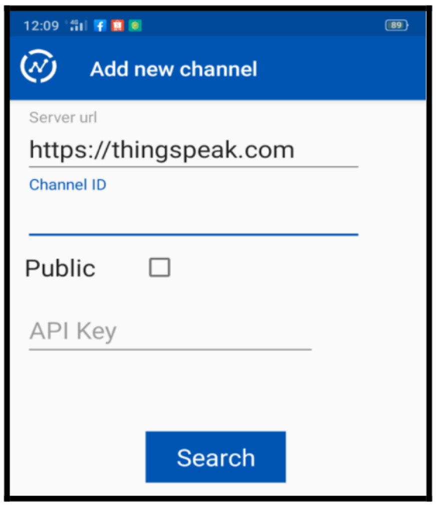


Figure C2

34. Print screen the output and paste in Table 3 as result.

RESULT

PART A: CONTROL OUTPUT FROM WEB BROWSER

Step 10	IP Address:
Step 13	Paste the layout from the browser
Step 15	Explain what happen when ON and OFF button in web browser is clicked.

Table 1

PART B: DISPLAY DATA FROM SENSOR TO WEB BROWSER

Step 22	Channel ID: Write API key:
Step 30	Print screen of displayed graph in Thingspeak web.

Table 2

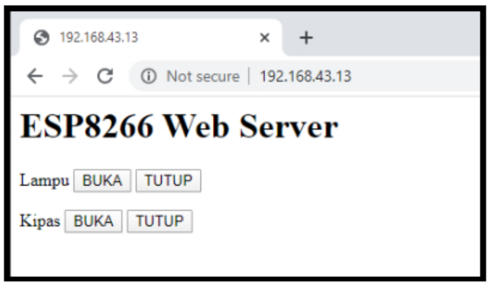
PART C: DISPLAY DATA FROM SENSOR TO MOBILE APPS

Step 34	Print screen of displayed graph in ThingView App.

Table 3

DISCUSSION

If you want to change the web page interface as following, which part of the coding should be change? Paste the new code below.

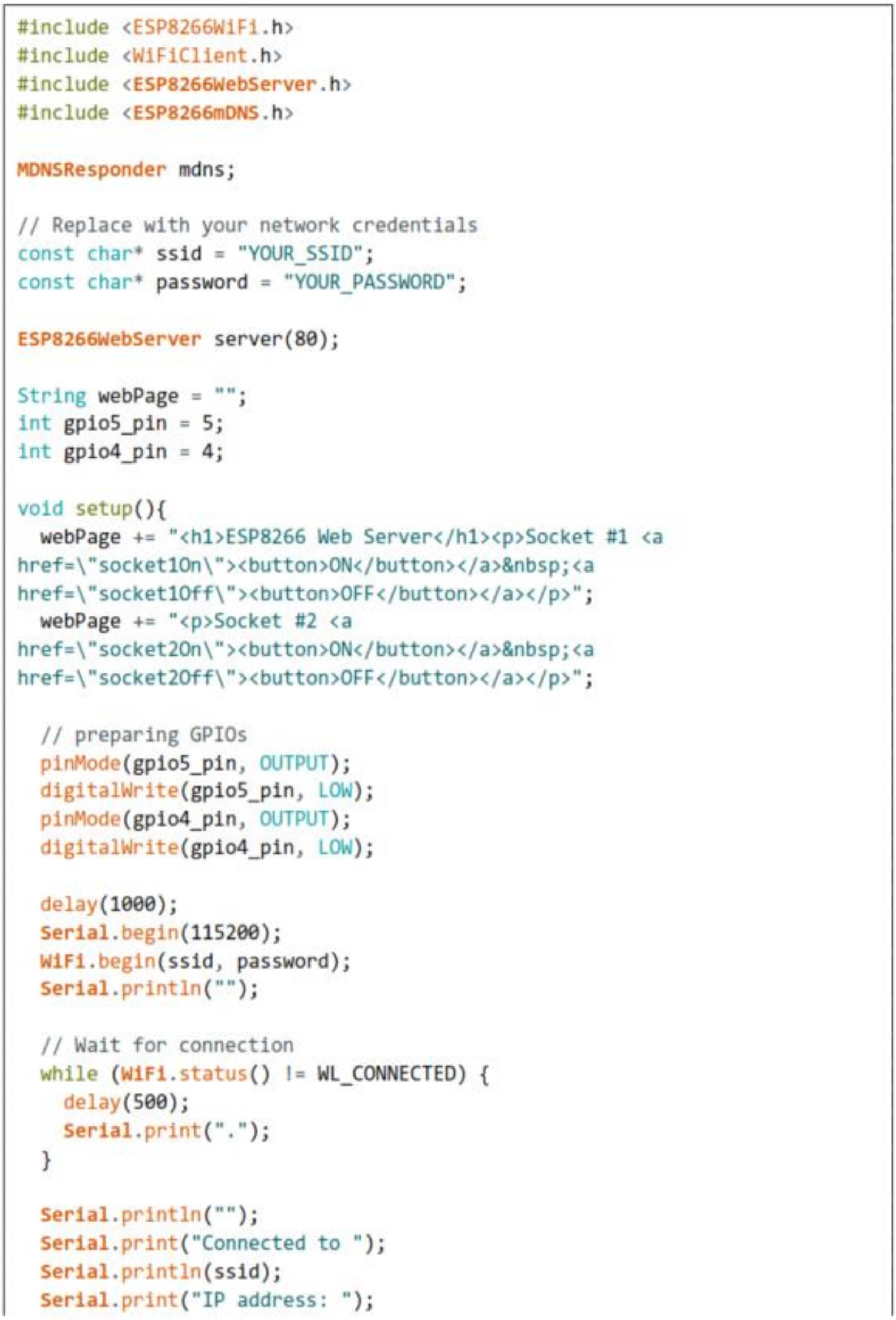


ESP8266 Web Server

Lampu BUKA TUTUP

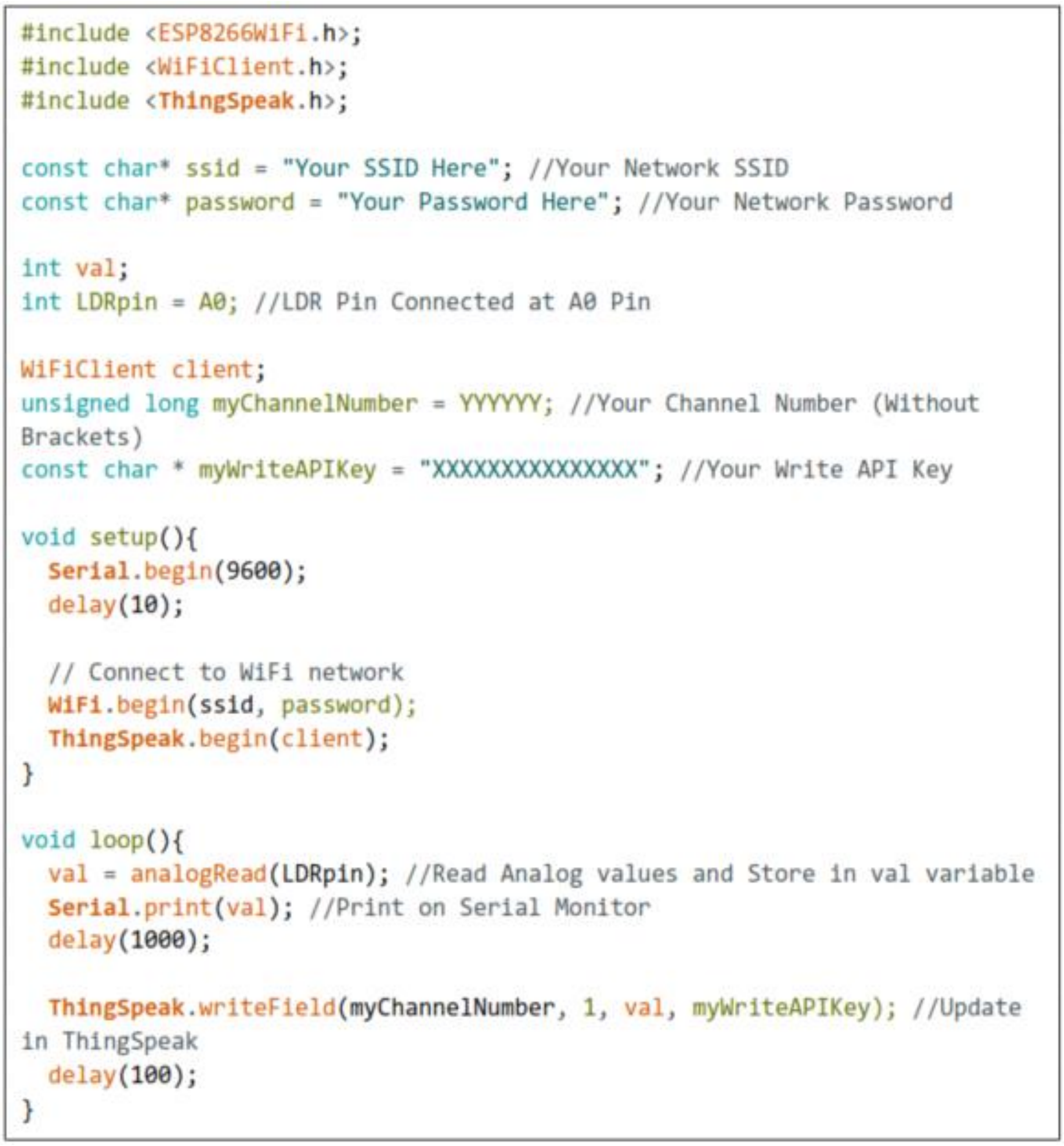
Kipas BUKA TUTUP

APPENDIX 1

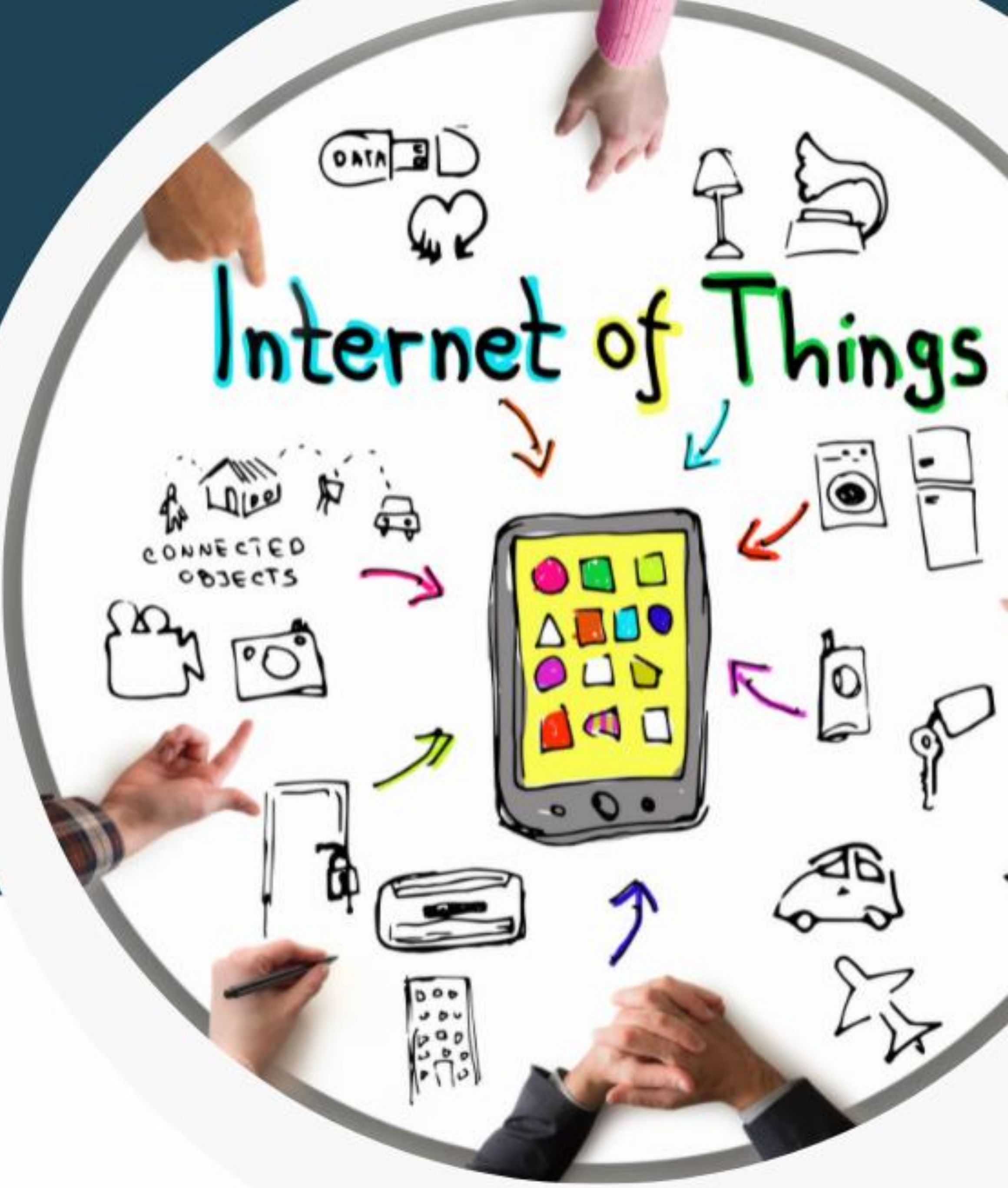


```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>

MDNSResponder mdns;

// Replace with your network credentials
const char* ssid = "YOUR_SSID";
const char* password = "YOUR_PASSWORD";

ESP8266WebServer server(80);

String webPage = "";
int gpio5_pin = 5;
int gpio4_pin = 4;

void setup(){
  webPage += "<h1>ESP8266 Web Server</h1><p>Socket #1 <a
href=\"socket1On\"><button>ON</button></a>&nbsp;<a
href=\"socket1Off\"><button>OFF</button></a></p>";
  webPage += "<p>Socket #2 <a
href=\"socket2On\"><button>ON</button></a>&nbsp;<a
href=\"socket2Off\"><button>OFF</button></a></p>";

  // preparing GPIOs
  pinMode(gpio5_pin, OUTPUT);
  digitalWrite(gpio5_pin, LOW);
  pinMode(gpio4_pin, OUTPUT);
  digitalWrite(gpio4_pin, LOW);

  delay(1000);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  if (mdns.begin("esp8266", WiFi.localIP())) {
    Serial.println("MDNS responder started");
  }

  server.on("/", [](){
    server.send(200, "text/html", webPage);
  });

  server.on("/socket1On", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio5_pin, HIGH);
    delay(1000);
  });

  server.on("/socket1Off", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio5_pin, LOW);
    delay(1000);
  });

  server.on("/socket2On", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio4_pin, HIGH);
    delay(1000);
  });

  server.on("/socket2Off", [](){
    server.send(200, "text/html", webPage);
    digitalWrite(gpio4_pin, LOW);
    delay(1000);
  });

  server.begin();
  Serial.println("HTTP server started");
}

void loop(){
  server.handleClient();
}
```

APPENDIX 2



```
#include <ESP8266WiFi.h>;
#include <WiFiClient.h>;
#include <ThingSpeak.h>;

const char* ssid = "Your SSID Here"; //Your Network SSID
const char* password = "Your Password Here"; //Your Network Password

int val;
int LDRpin = A0; //LDR Pin Connected at A0 Pin

WiFiClient client;
unsigned long myChannelNumber = YYYYYY; //Your Channel Number (Without
Brackets)
const char * myWriteAPIKey = "XXXXXXXXXXXXXXX"; //Your Write API Key

void setup(){
  Serial.begin(9600);
  delay(10);

  // Connect to WiFi network
  WiFi.begin(ssid, password);
  ThingSpeak.begin(client);
}

void loop(){
  val = analogRead(LDRpin); //Read Analog values and Store in val variable
  Serial.print(val); //Print on Serial Monitor
  delay(1000);

  ThingSpeak.writeField(myChannelNumber, 1, val, myWriteAPIKey); //Update
in ThingSpeak
  delay(100);
}
```



TUTORIAL 04

TUTORIAL: 4

Title: NodeMCU ESP8266 using MQTT Protocol

OBJECTIVES

Upon completion of this tutorial, you should be able to:

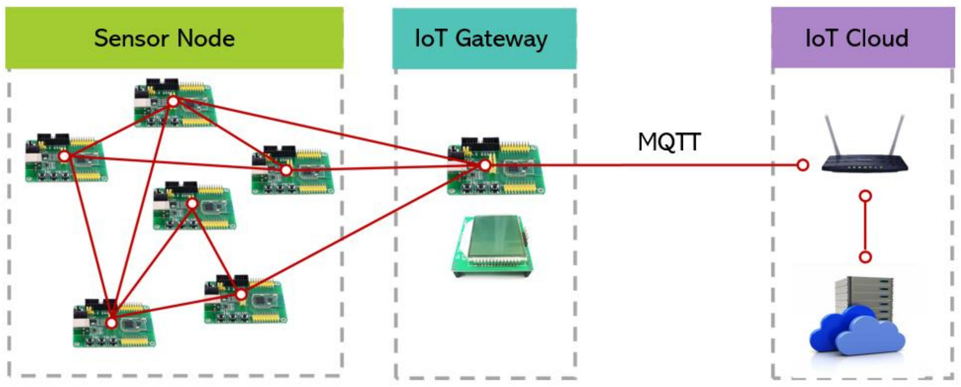
- Install Node-RED (Flow-Based IoT programming tools) locally
- Run Node-RED dashboard
- MQTT connectivity using HiveMQ MQTT broker
- MQTT connectivity using IoT MQTT Panel android application

EQUIPMENTS

- Internet Connection
- Laptop/Personal Computer
- Android Smartphone

THEORY

Message Queuing Telemetry Transport (MQTT) is **used for** data exchange between constrained **devices and server applications**. Ideal for machine-to-machine (M2M) communication – embedded devices. Furthermore, its low bandwidth, low memory, and low power.



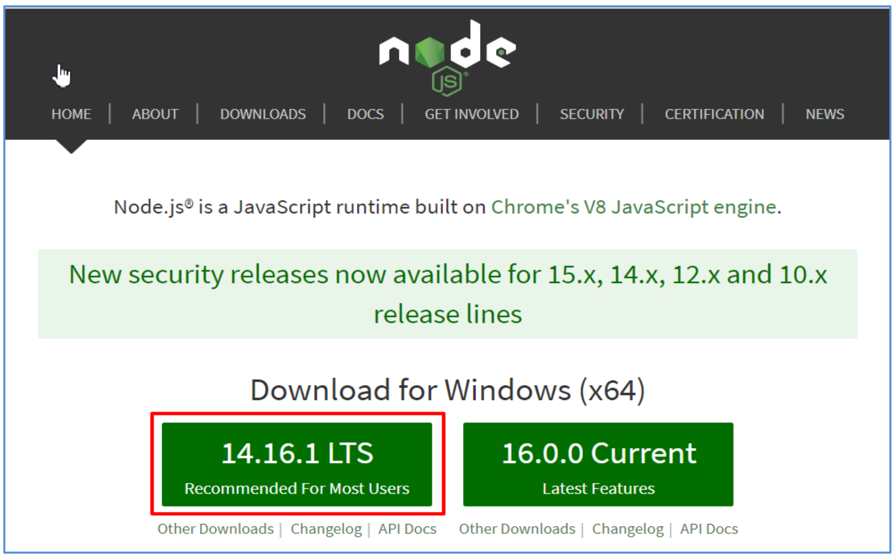
MQTT is widely used in IoT applications systems alongside cloud platform like AWS, Alibaba, Microsoft Azure etc.

PROCEDURE

PART A: INSTALL NODE-RED (FLOW-BASED IOT PROGRAMMING TOOLS)

Install Node-RED Locally

1. Node-RED requires NodeJS. You need to install NodeJS. Go to https://nodejs.org/en/ and install the latest LTS (Long Term Support) version of NodeJS as in Figure A1.


HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

New security releases now available for 15.x, 14.x, 12.x and 10.x release lines

Download for Windows (x64)

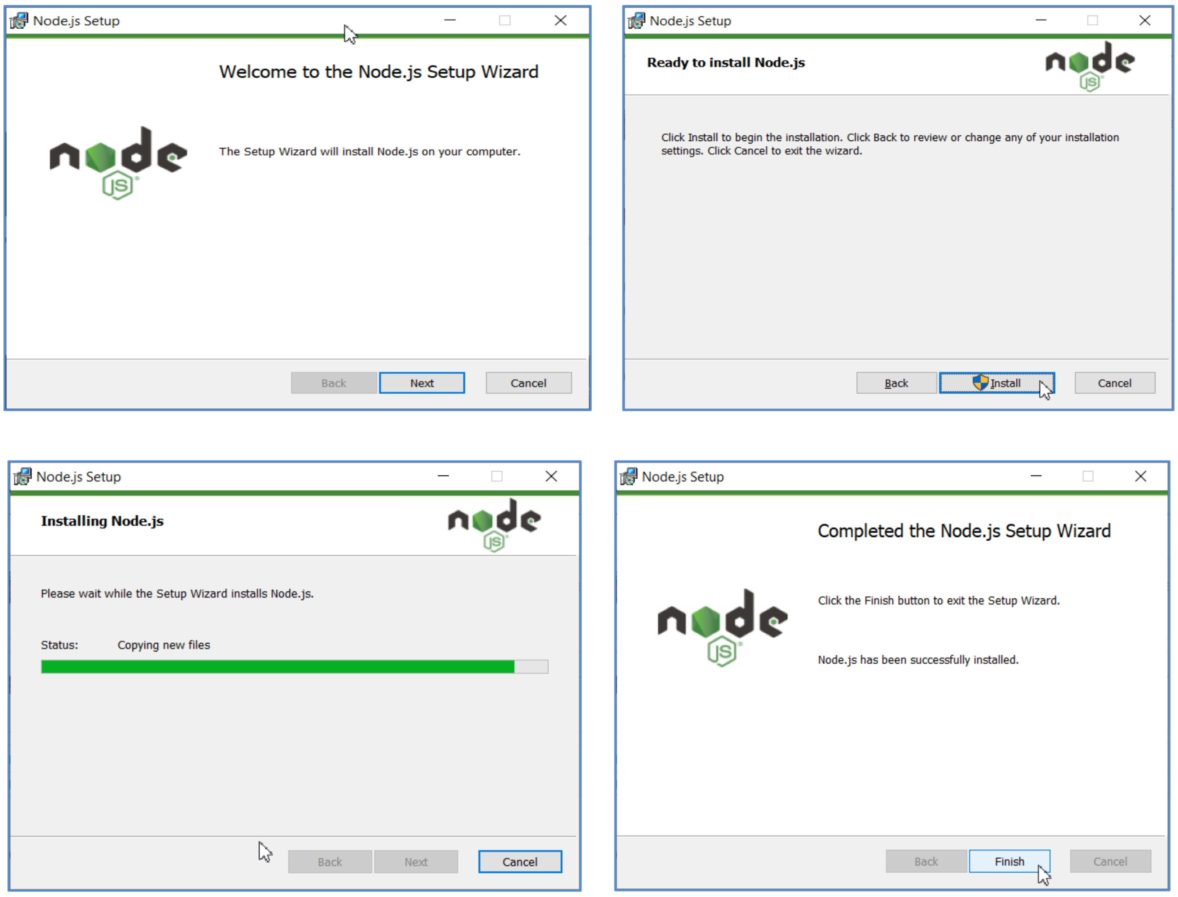
14.16.1 LTS Recommended For Most Users

16.0.0 Current Latest Features

Other Downloads | Changelog | API Docs Other Downloads | Changelog | API Docs

Figure A1

2. Once downloaded, double-click the installer and follow the installation steps by default options, until the installation is completed and click **Finish** button to close the installation wizard (Figure A2).


Node.js Setup

Welcome to the Node.js Setup Wizard

The Setup Wizard will install Node.js on your computer.

Node.js Setup

Ready to install Node.js

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Node.js Setup

Installing Node.js

Please wait while the Setup Wizard installs Node.js.

Status: Copying new files

Node.js Setup

Completed the Node.js Setup Wizard

Click the Finish button to exit the Setup Wizard.

Node.js has been successfully installed.

Figure A2

3. NodeJS runs as background processes in Windows OS. Therefore, to call NodeJS, you need to use Command Prompt as command line tools. Click the Windows icon and type **cmd** and click Command Prompt.

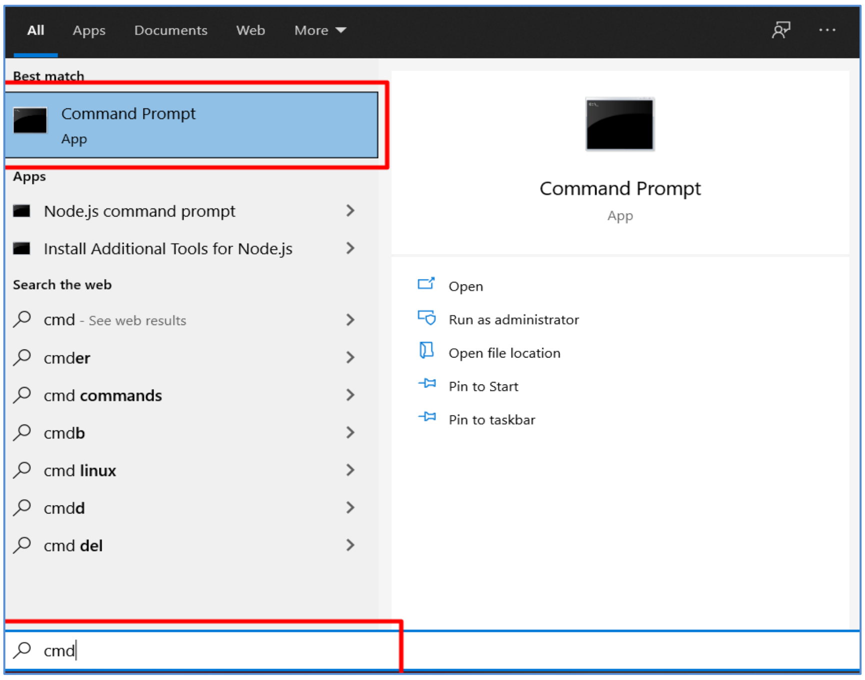
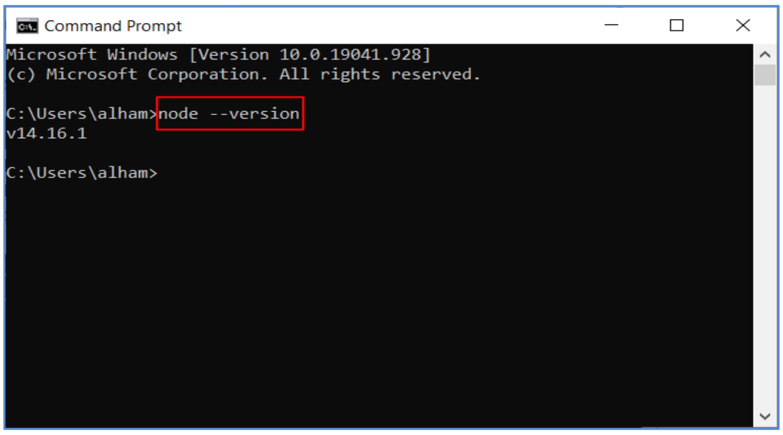


Figure A3

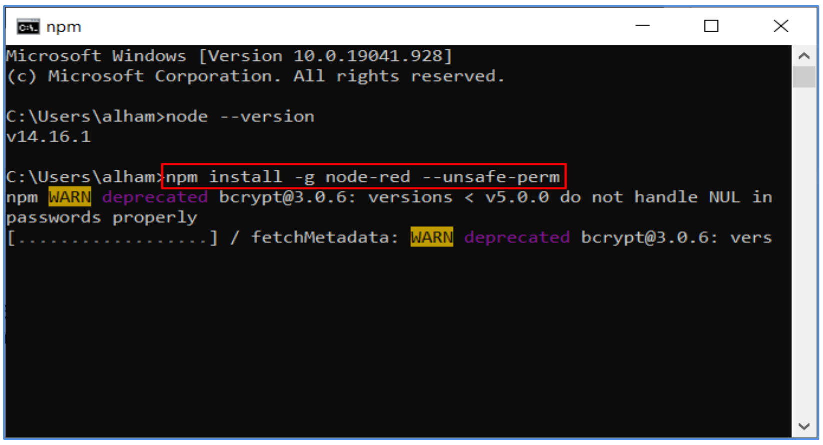
4. To make sure NodeJS is running on your computer. Type **node --version** and you should receive the version of current NodeJS running on your computer. *If Command Prompt does not return the version, try to Log Out from Windows OS and Log In again.*



```
Microsoft Windows [Version 10.0.19041.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\alham>node --version
v14.16.1

C:\Users\alham>
```

Figure A4

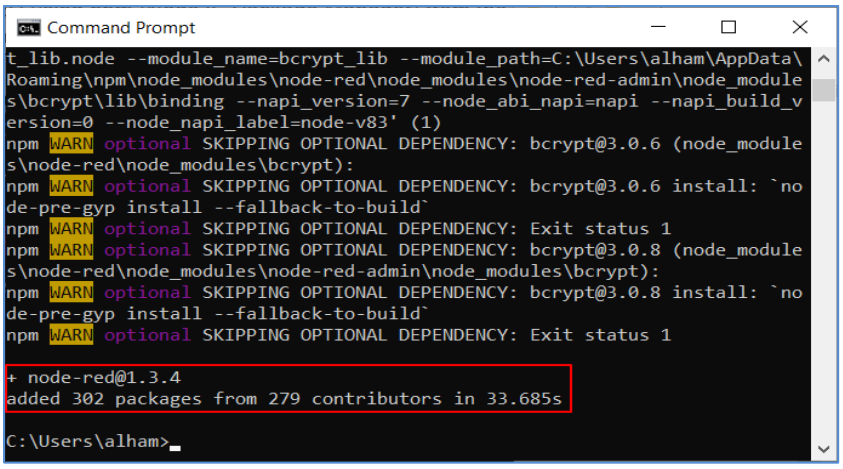
5. Install Node-RED using **npm** (NodeJS Package Manager) from the command line on Command Prompt. Type **npm install -g node-red --unsafe-perm** and Enter.



```
Microsoft Windows [Version 10.0.19041.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\alham>node --version
v14.16.1

C:\Users\alham>npm install -g node-red --unsafe-perm
npm WARN deprecated bcrypt@3.0.6: versions < v5.0.0 do not handle NUL in
passwords properly
[..................] / fetchMetadata: WARN deprecated bcrypt@3.0.6: vers
```

Figure A5

6. Wait until the install process is successful, where you will see the status of node-red packages is added. Print screen the layout from your command prompt and paste in Table 1 as a result.

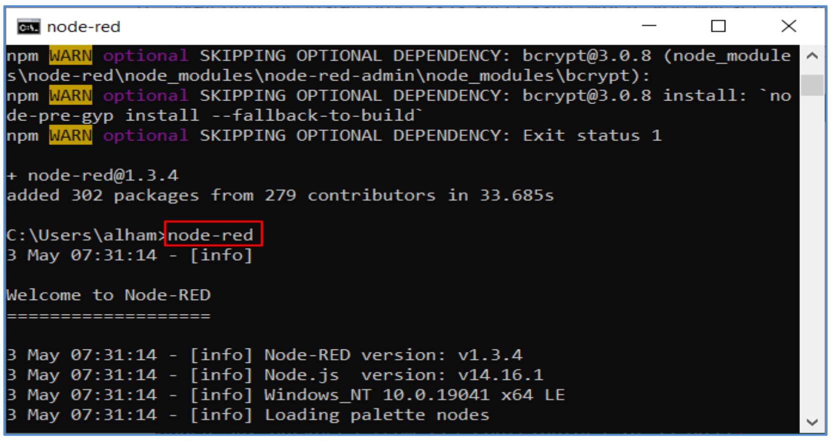


```
t_lib.node --module_name=bcrypt_lib --module_path=C:\Users\alham\AppData\
Roaming\npm\node_modules\node-red\node_modules\node-red-admin\node_module
s\bcrypt\lib\binding --napi_version=7 --node_abi_napi=napi --napi_build_v
ersion=0 --node_napi_label=node-v83' (1)
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: bcrypt@3.0.6 (node_module
s\node-red\node_modules\bcrypt):
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: bcrypt@3.0.6 install: `no
de-pre-gyp install --fallback-to-build`
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: Exit status 1
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: bcrypt@3.0.8 (node_module
s\node-red\node_modules\node-red-admin\node_modules\bcrypt):
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: bcrypt@3.0.8 install: `no
de-pre-gyp install --fallback-to-build`
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: Exit status 1

+ node-red@1.3.4
added 302 packages from 279 contributors in 33.685s

C:\Users\alham>_
```

Figure A6

Run Node-RED Locally

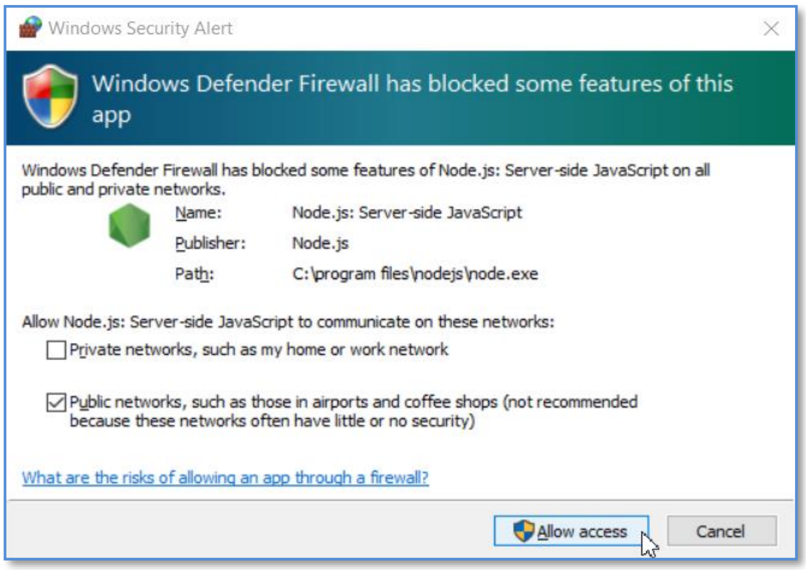
7. Type **node-red** and Enter to run Node-RED applications.



```
node-red
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: bcrypt@3.0.8 (node_module
s\node-red\node_modules\node-red-admin\node_modules\bcrypt):
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: bcrypt@3.0.8 install: `no
de-pre-gyp install --fallback-to-build`
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: Exit status 1

+ node-red@1.3.4
added 302 packages from 279 contributors in 33.685s

C:\Users\alham>node-red
3 May 07:31:14 - [info]

Welcome to Node-RED
===================

3 May 07:31:14 - [info] Node-RED version: v1.3.4
3 May 07:31:14 - [info] Node.js  version: v14.16.1
3 May 07:31:14 - [info] Windows_NT 10.0.19041 x64 LE
3 May 07:31:14 - [info] Loading palette nodes
```

Figure A7

8. Wait until the execution process is done, where you will see the status of **Server now running at** http://127.0.0.1:1880/. During execution for the first time, you will see Firewall Windows Security Alert, click **Allow access**.



Windows Security Alert

Windows Defender Firewall has blocked some features of this app

Windows Defender Firewall has blocked some features of Node.js: Server-side JavaScript on all public and private networks.

Name: Node.js: Server-side JavaScript

Publisher: Node.js

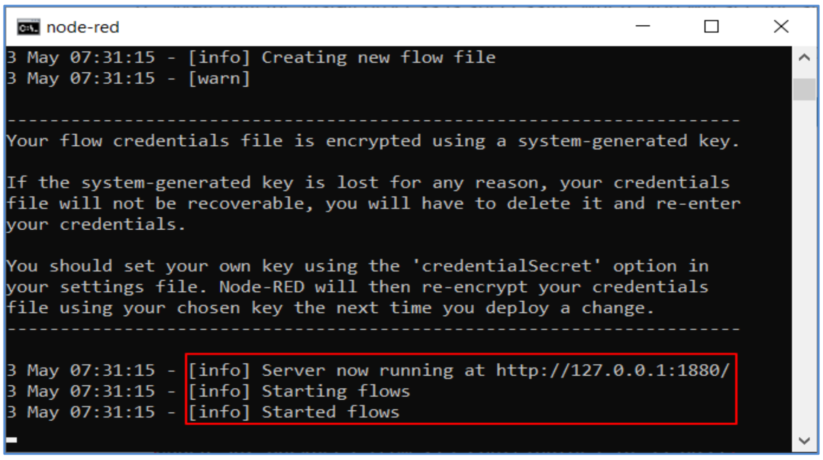
Path: C:\program files\nodejs\node.exe

Allow Node.js: Server-side JavaScript to communicate on these networks:

☐ Private networks, such as my home or work network

☑ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?



```
node-red
3 May 07:31:15 - [info] Creating new flow file
3 May 07:31:15 - [warn]

---------------------------------------------------------------------
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
---------------------------------------------------------------------

3 May 07:31:15 - [info] Server now running at http://127.0.0.1:1880/
3 May 07:31:15 - [info] Starting flows
3 May 07:31:15 - [info] Started flows
```

Figure A8

9. Access Node-RED. Open your browser and type http://localhost:1880 or http://127.0.0.1:1880/ and Enter.

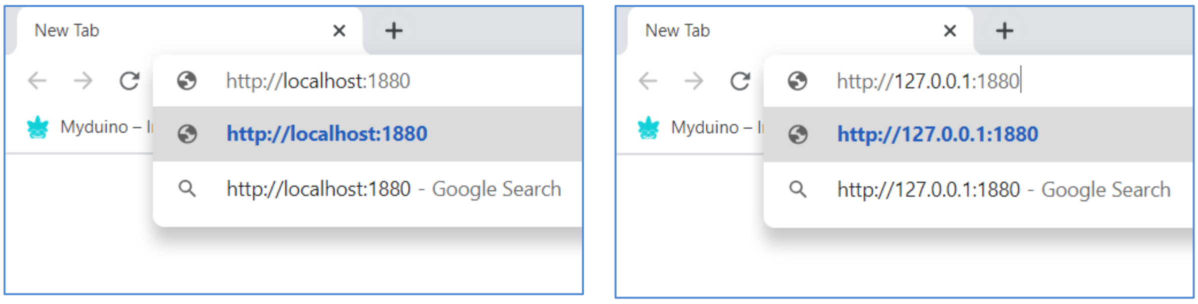


Figure A9

10. Node-RED interface as in your browser. The flow-based programming tools.

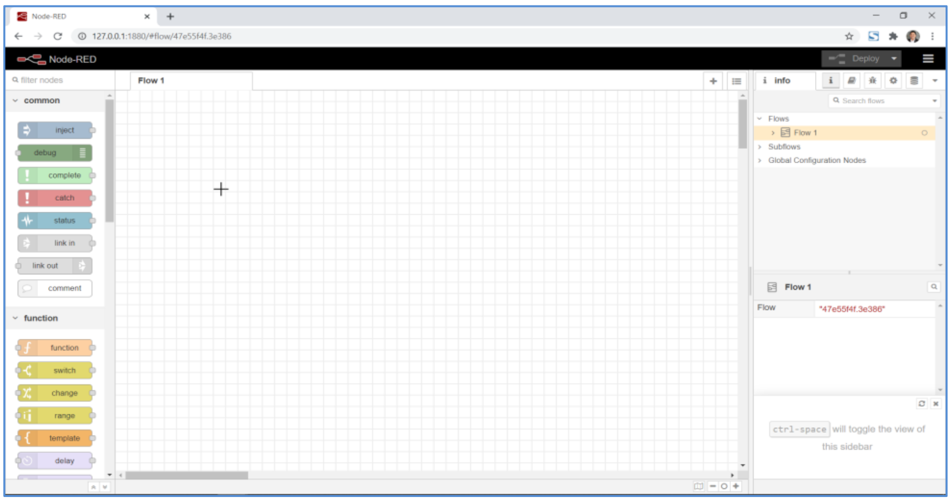


Figure A10

Install Node-RED Dashboard

11. Click the **Setting** menu (3 horizontal bar icon) and Click **Manage palette**.

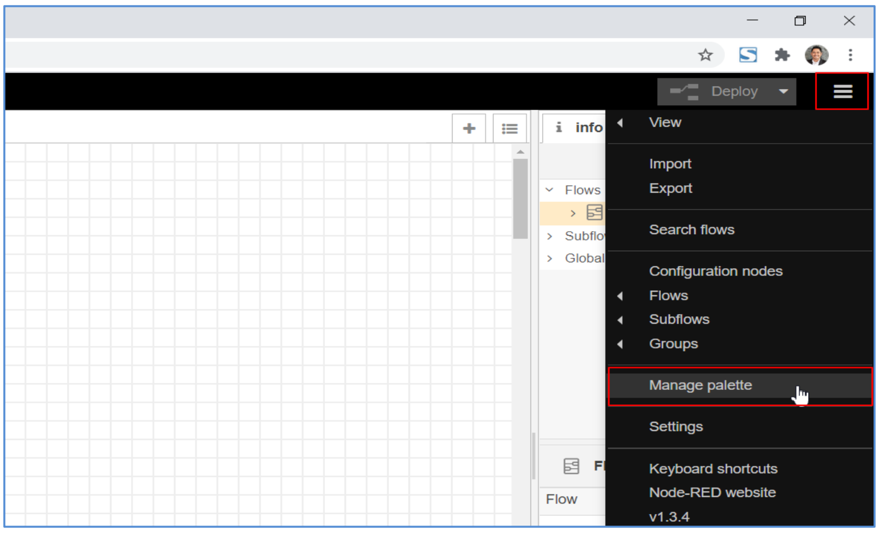
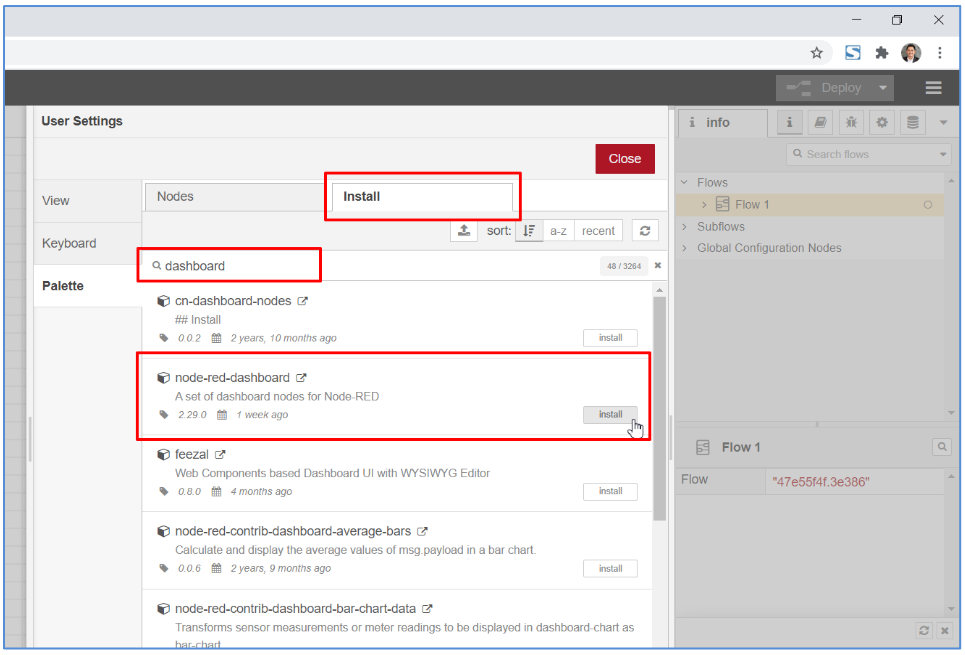


Figure A11

12. Click **Install** tab, type **dashboard** and click **install** button on **node-red-dashboard**.



cn-dashboard-nodes
Install
0.0.2 2 years, 10 months ago install

node-red-dashboard
A set of dashboard nodes for Node-RED
2.29.0 1 week ago install

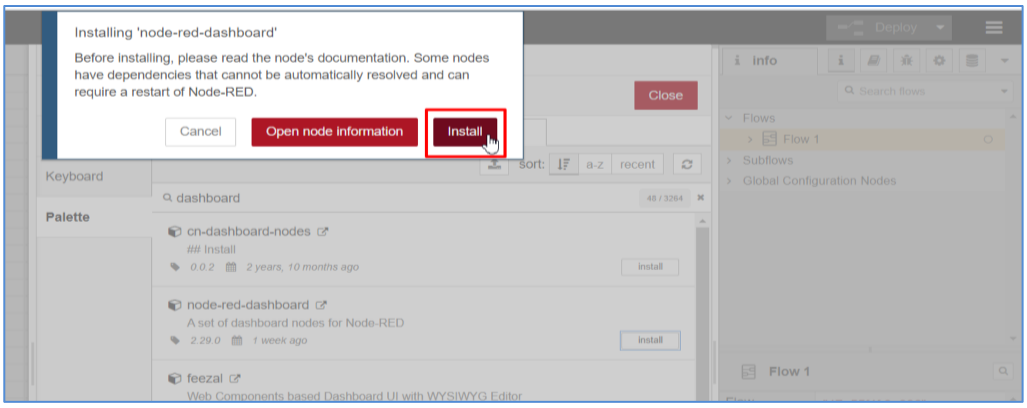
feezal
Web Components based Dashboard UI with WYSIWYG Editor
0.8.0 4 months ago install

node-red-contrib-dashboard-average-bars
Calculate and display the average values of msg.payload in a bar chart.
0.0.6 2 years, 9 months ago install

node-red-contrib-dashboard-bar-chart-data
Transforms sensor measurements or meter readings to be displayed in dashboard-chart as bar-chart

Figure A12

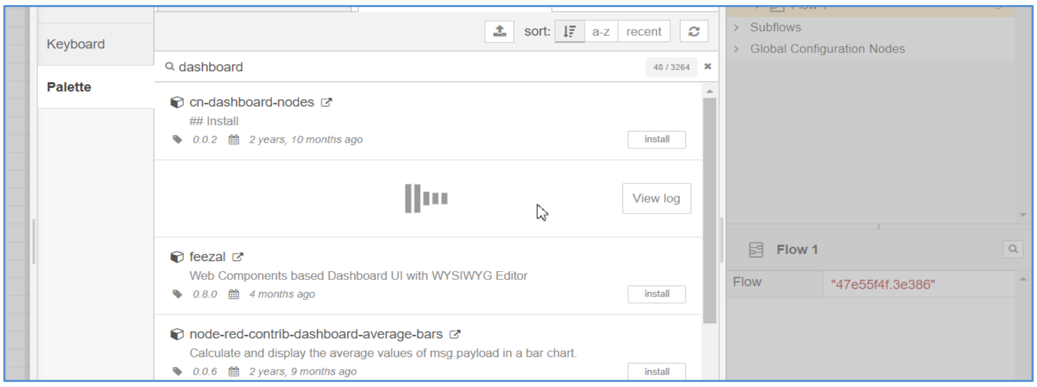
13. Click **Install** to confirm the installation of node-red-dashboard palette.


Installing 'node-red-dashboard'

Before installing, please read the node's documentation. Some nodes have dependencies that cannot be automatically resolved and can require a restart of Node-RED.

Figure A13

14. Wait until the installation process is done.



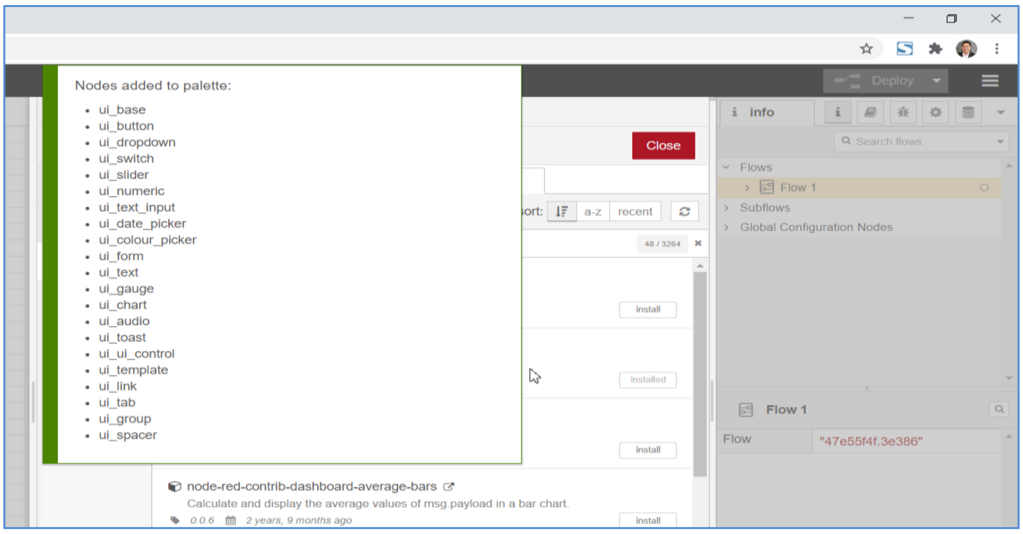
cn-dashboard-nodes
Install
0.0.2 2 years, 10 months ago install

feezal
Web Components based Dashboard UI with WYSIWYG Editor
0.8.0 4 months ago install

node-red-contrib-dashboard-average-bars
Calculate and display the average values of msg.payload in a bar chart.
0.0.6 2 years, 9 months ago install

Figure A14

15. Once the installation process is done, you will see added node-red-dashboard nodes added to the palette.

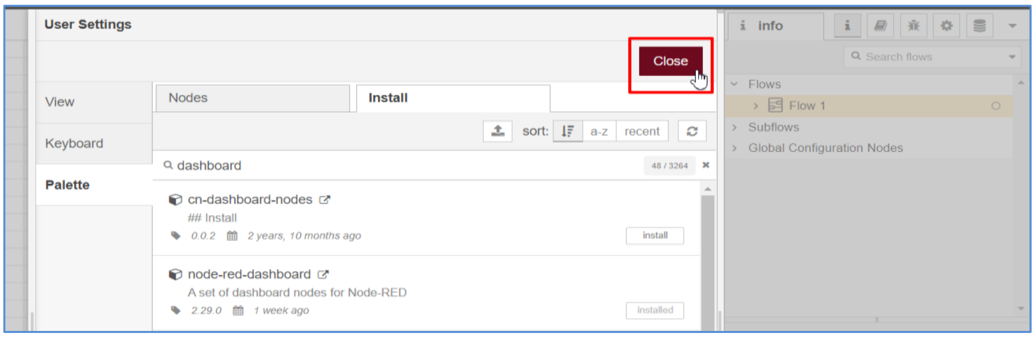

Nodes added to palette:

- ui_base
- ui_button
- ui_dropdown
- ui_switch
- ui_slider
- ui_numeric
- ui_text_input
- ui_date_picker
- ui_colour_picker
- ui_form
- ui_text
- ui_gauge
- ui_chart
- ui_audio
- ui_toast
- ui_ui_control
- ui_template
- ui_link
- ui_tab
- ui_group
- ui_spacer

node-red-contrib-dashboard-average-bars
Calculate and display the average values of msg.payload in a bar chart.
0.0.6 2 years, 9 months ago

Figure A15

16. Click the Close button.



cn-dashboard-nodes
Install
0.0.2 2 years, 10 months ago

node-red-dashboard
A set of dashboard nodes for Node-RED
2.29.0 1 week ago

Figure A16

17. Scroll on the palette list, you will see added node-red-dashboard nodes, including **button**, **dropdown**, **switch**, **slider**, etc.

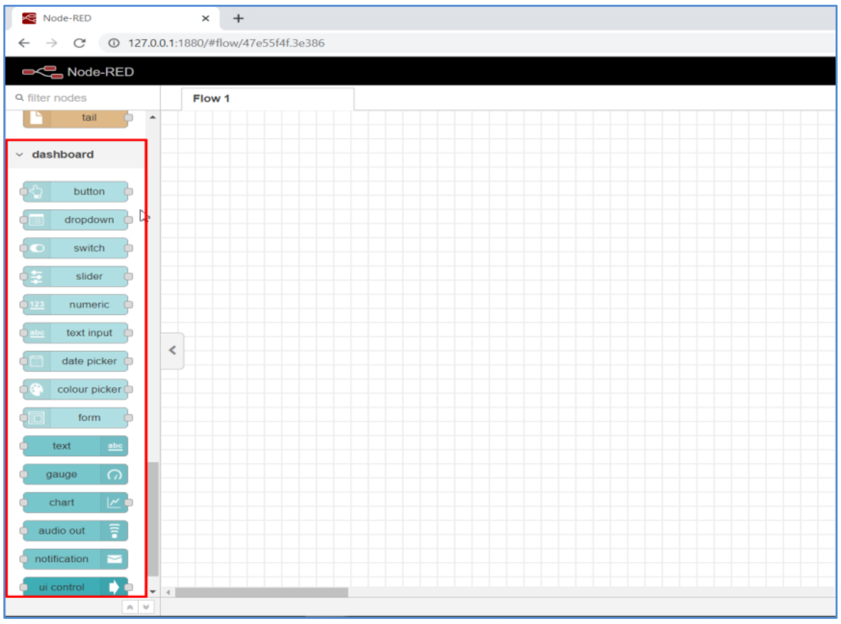
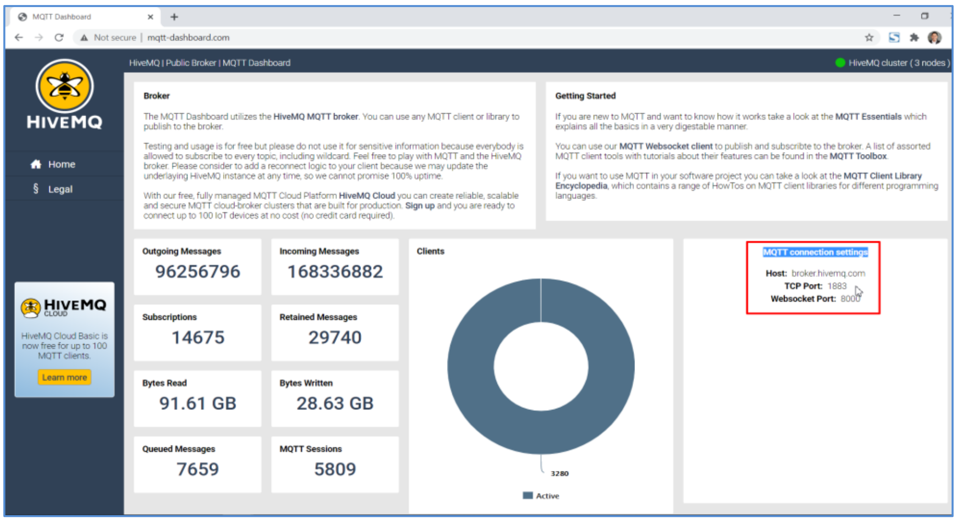


Figure A17

PART B: MQTT CONNECTIVITY USING HIVEMQ MQTT BROKER

Connect to HiveMQ MQTT Broker

1. Open your browser and go to http://www.mqtt-dashboard.com/ (official information site for HiveMQ MQTT broker).



MQTT Dashboard

Not secure | mqtt-dashboard.com

HiveMQ | Public Broker | MQTT Dashboard

HiveMQ cluster (3 nodes)

Broker

The MQTT Dashboard utilizes the **HiveMQ MQTT broker**. You can use any MQTT client or library to publish to the broker.

Testing and usage is for free but please do not use it for sensitive information because everybody is allowed to subscribe to every topic, including wildcard. Feel free to play with MQTT and the HiveMQ broker. Please consider to add a reconnect logic to your client because we may update the underlaying HiveMQ instance at any time, so we cannot promise 100% uptime.

With our free, fully managed MQTT Cloud Platform **HiveMQ Cloud** you can create reliable, scalable and secure MQTT cloud-broker clusters that are built for production. **Sign up** and you are ready to connect up to 100 IoT devices at no cost (no credit card required).

Getting Started

If you are new to MQTT and want to know how it works take a look at the **MQTT Essentials** which explains all the basics in a very digestable manner.

You can use our **MQTT Websocket client** to publish and subscribte to the broker. A list of assorted MQTT client tools with tutorials about their features can be found in the **MQTT Toolbox**.

If you want to use MQTT in your software project you can take a look at the **MQTT Client Library Encyclopedia**, which contains a range of HowTos on MQTT client libraries for different programming languages.

Outgoing Messages	96256796
Incoming Messages	168336882
Subscriptions	14675
Retained Messages	29740
Bytes Read	91.61 GB
Bytes Written	28.63 GB
Queued Messages	7659
MQTT Sessions	5809

Clients: 3280 Active

MQTT connection settings

Host: broker.hivemq.com
TCP Port: 1883
Websocket Port: 8000

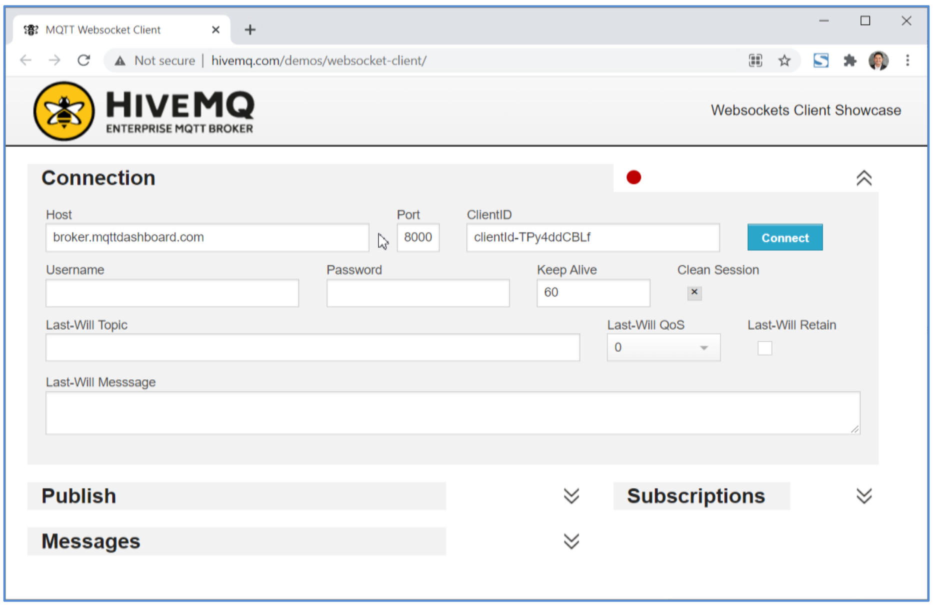
HiveMQ Cloud Basic is now free for up to 100 MQTT clients.

Figure B1

2. Look for MQTT connection settings:
   a. **Host**: broker.hivemq.com
   b. **TCP Port**: 1883
   c. **Websocket Port**: 8000

MQTT connectivity using HiveMQ Websocket Client

3. On the browser go to http://www.hivemq.com/demos/websocket-client/



MQTT Websocket Client

Not secure | hivemq.com/demos/websocket-client/

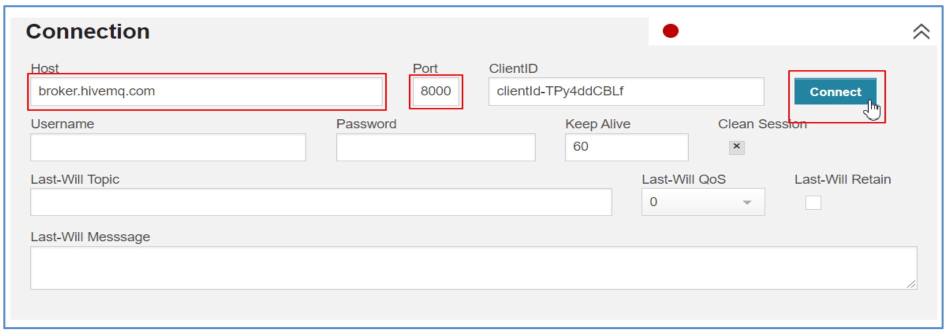
HIVEMQ ENTERPRISE MQTT BROKER — Websockets Client Showcase

Connection

Host	broker.mqttdashboard.com
Port	8000
ClientID	clientId-TPy4ddCBLf
Username	
Password	
Keep Alive	60
Clean Session	✓
Last-Will Topic	
Last-Will QoS	0
Last-Will Retain	
Last-Will Messsage	

Figure B2

4. Replace the Host and the Port 8000 (for Websocket connection) as follows on Step 2, where **ClientID** will be left auto-generated and click the **Connect** button.

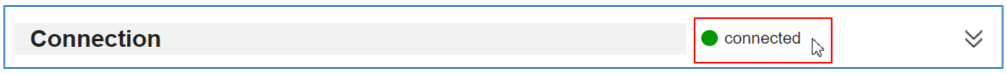


Connection

Host: broker.hivemq.com	Port: 8000	ClientID: clientId-TPy4ddCBLf	
Username:	Password:	Keep Alive: 60	Clean Session: ✓
Last-Will Topic:		Last-Will QoS: 0	Last-Will Retain:
Last-Will Messsage:			

Figure B3

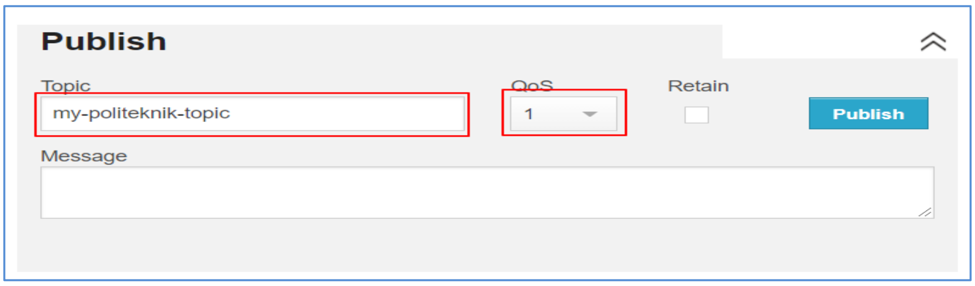
5. Wait until the connection is successful and you will see Connected status.



Connection ● connected

Figure B4

6. On the Publish section, replace **testtopic/1** to your own topic (for example: **my-politeknik-topic**) and change QoS (Quality of Service) to **1**.

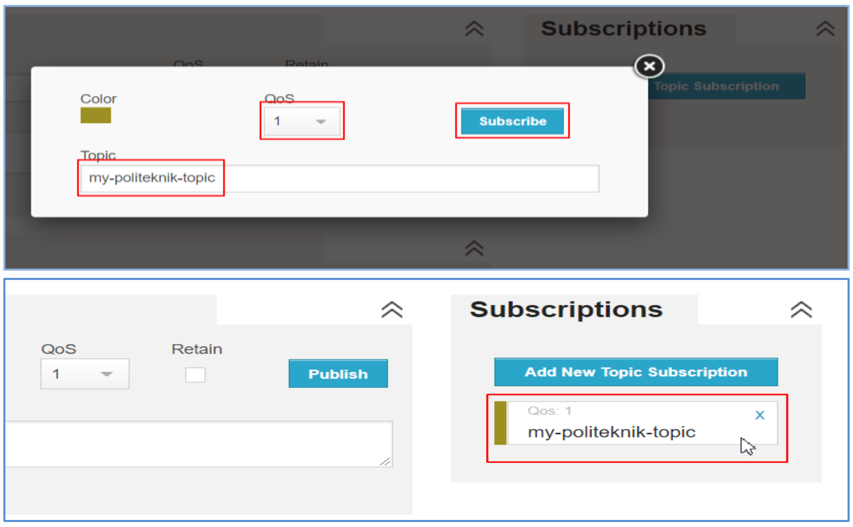


Publish

Topic: my-politeknik-topic
QoS: 1
Retain:
Message:

Figure B5

7. On the Subscriptions section, click on **Add New Topic Subscription**, replace **testtopic/1** to your own topic (same as in Step 6), change QoS to **1** and click **Subscribe** button. You will see the topic is listed as a subscribed topic.



Subscriptions

Color
QoS: 1
Topic: my-politeknik-topic

QoS: 1
Retain:

Subscriptions

Qos: 1
my-politeknik-topic

Figure B6

8. Publish a message, by adding a message (for example: Hello MQTT World!) into the Message area and click the **Publish** button. You will receive the message on the Messages section as in Figure B7. Print screen the layout from your WebSocket Client Showcase and paste in Table 2 as a result.

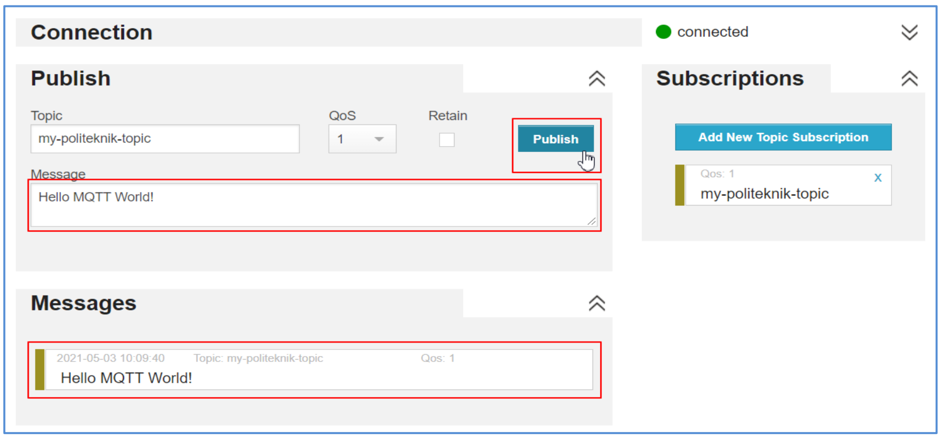


Figure B7

MQTT connectivity using Node-RED TCP Client

9. Under the **network** palette, click **mqtt in** node, drag and release it on the Node-RED workspace.

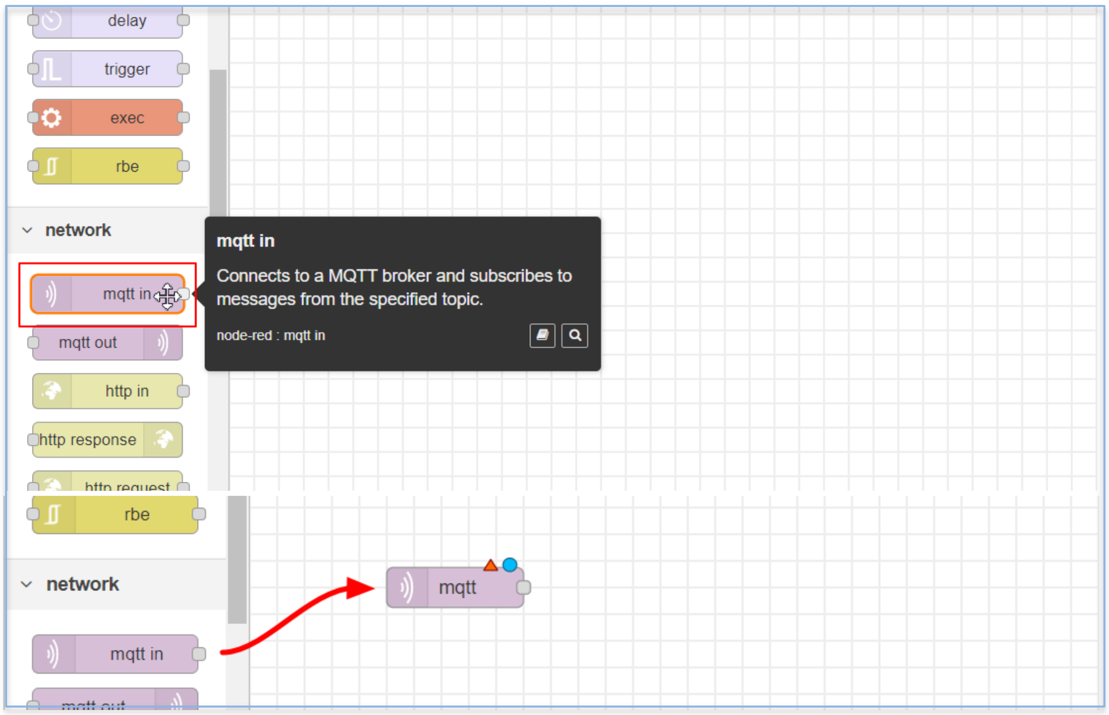
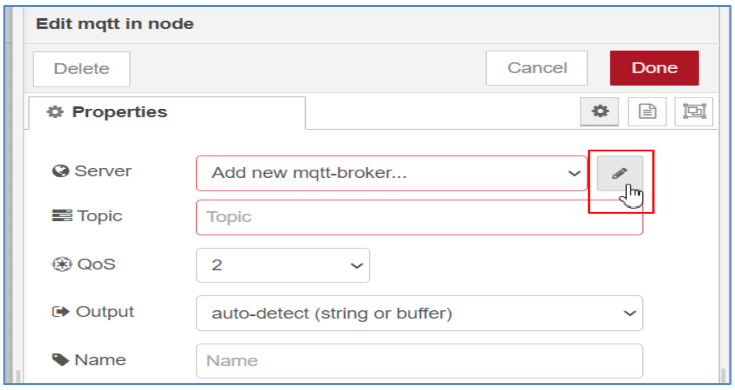


Figure B8

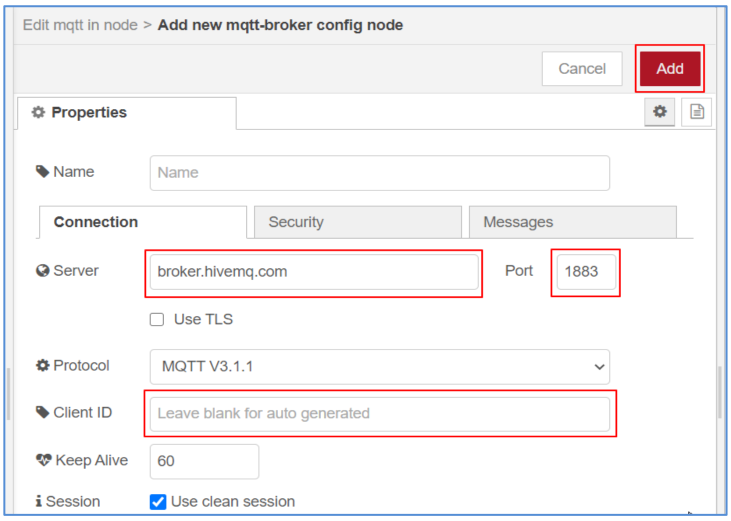
10. Double click the **mqtt in** node to change the settings and click the pencil icon to add HiveMQ MQTT broker.



Server: Add new mqtt-broker...
Topic: Topic
QoS: 2
Output: auto-detect (string or buffer)
Name: Name

Figure B9

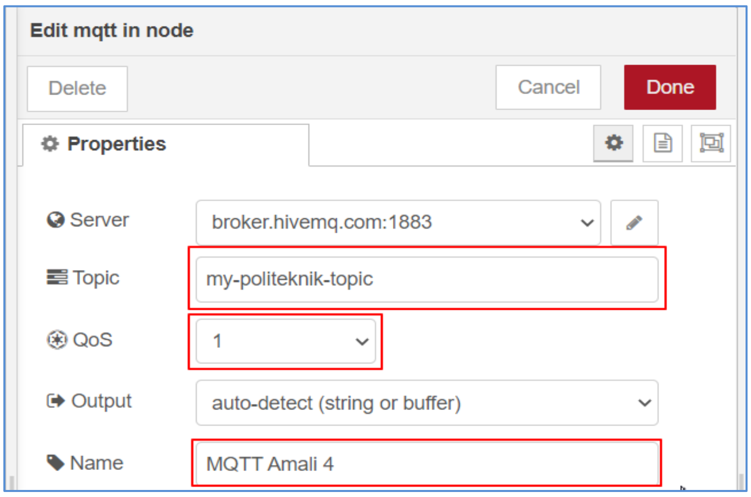
11. Add **broker.hivemq.com** on **Server** field, **Port** default 1883 (for TCP connection), leave the **Client ID** field for auto-generated client id and click the **Add** button.



Name: Name
Server: broker.hivemq.com Port: 1883
Use TLS
Protocol: MQTT V3.1.1
Client ID: Leave blank for auto generated
Keep Alive: 60
Session: ☑ Use clean session

Figure B10

12. Add your MQTT topic (same as in Step 6) on the **Topic** field, change the QoS to **1** and click the **Done** button.



Server: broker.hivemq.com:1883
Topic: my-politeknik-topic
QoS: 1
Output: auto-detect (string or buffer)
Name: MQTT Amali 4

Figure B11

13. Under the **dashboard** palette, click, drag and release **gauge** node into the Node-RED workspace.

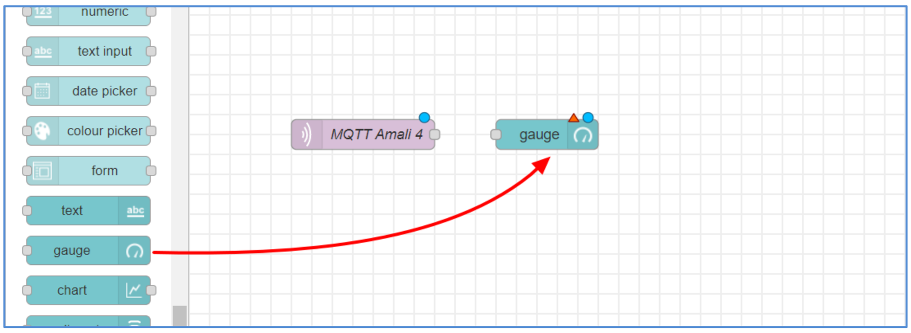
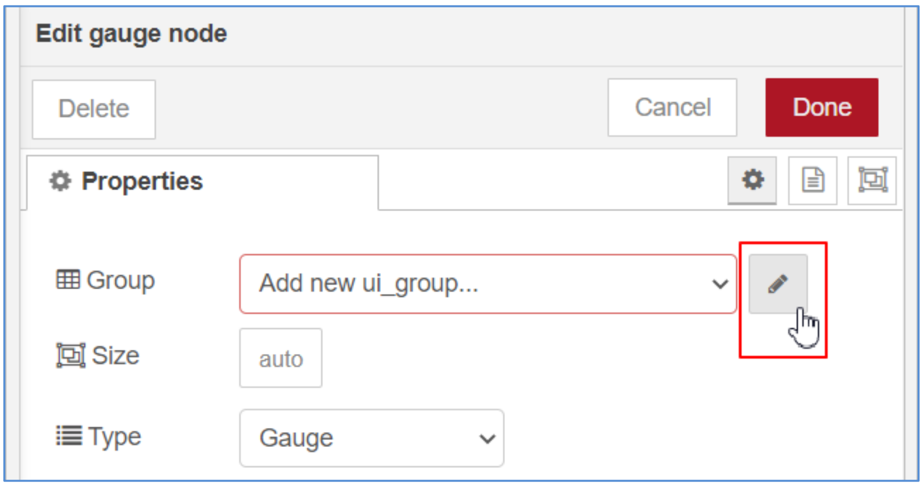


Figure B12

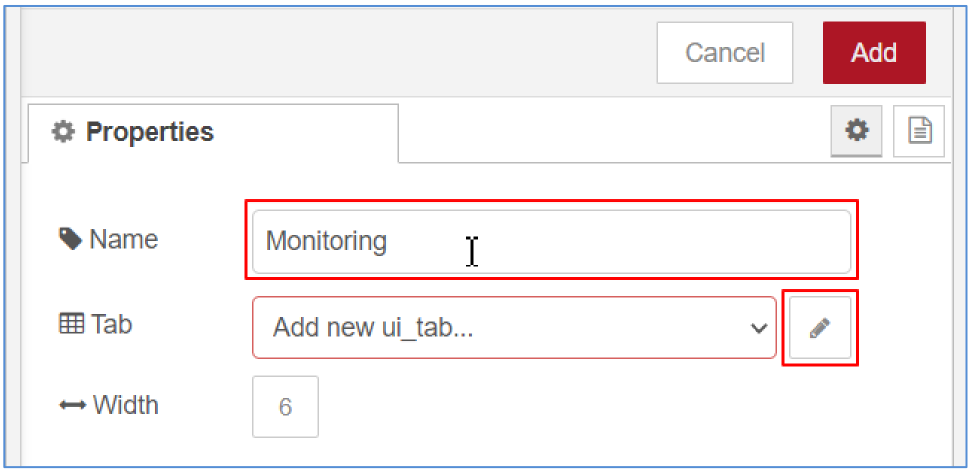
14. Double click the **gauge** node. Click the pencil icon to set up the name of the Group.



Group: Add new ui_group...
Size: auto
Type: Gauge

Figure B13

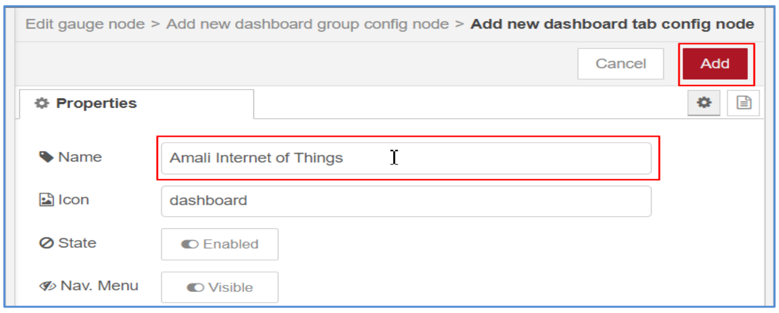
15. Change the **default** title of the **Name** field to **Monitoring** and click the pencil icon to set up the name of the Tab.



Name: Monitoring
Tab: Add new ui_tab...
Width: 6

Figure B14

16. Change the default title of the **Name** field to **Amali Internet of Things** and click the **Add** button.



Edit gauge node > Add new dashboard group config node > **Add new dashboard tab config node**

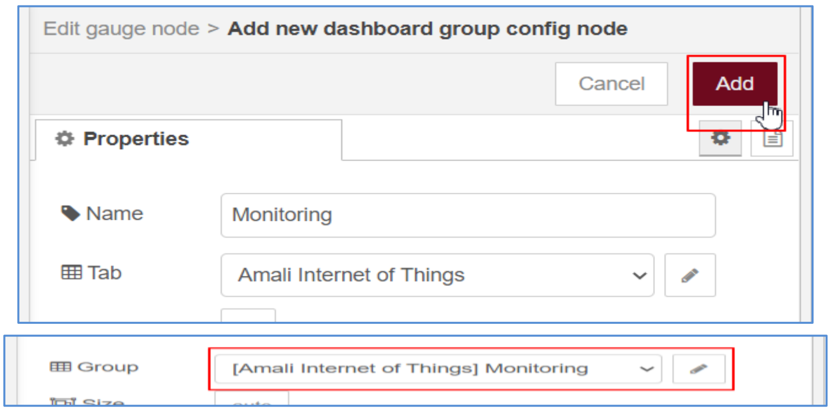
Cancel | Add

Properties

Name: Amali Internet of Things
Icon: dashboard
State: Enabled
Nav. Menu: Visible

Figure B15

17. Continue to click the **Add** button. Then you will see the Group as **[Amali Internet of Things] Monitoring**.



Edit gauge node > **Add new dashboard group config node**

Cancel | Add

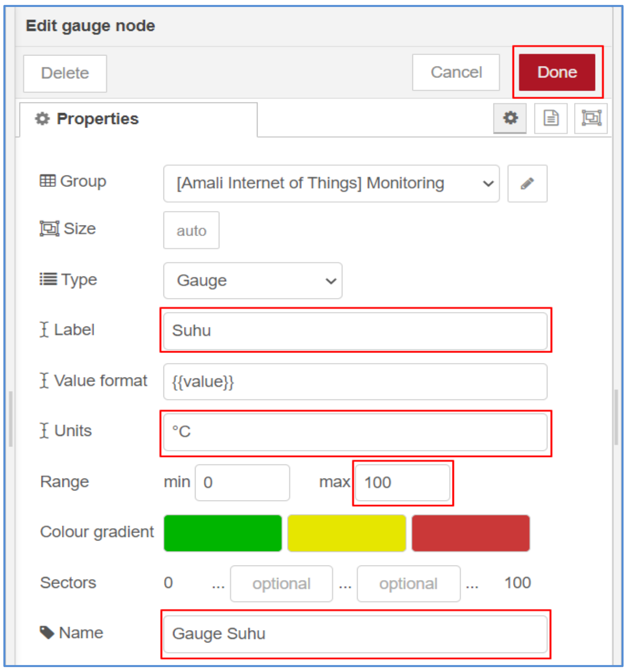
Properties

Name: Monitoring
Tab: Amali Internet of Things

Group: [Amali Internet of Things] Monitoring

Figure B16

18. Replace the **Label** field title as **Suhu**, the **Units** as **°C**, the max **Range** as **100** and the **Name** of the gauge node is **Gauge Suhu** and click the **Done** button.



Edit gauge node

Delete | Cancel | Done

Properties

Group: [Amali Internet of Things] Monitoring
Size: auto
Type: Gauge
Label: Suhu
Value format: {{value}}
Units: °C
Range: min 0 max 100
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 100
Name: Gauge Suhu

Figure B17

19. Connect the wires from **mqtt in** node [MQTT Amali 4] to **gauge** node [Gauge Suhu].

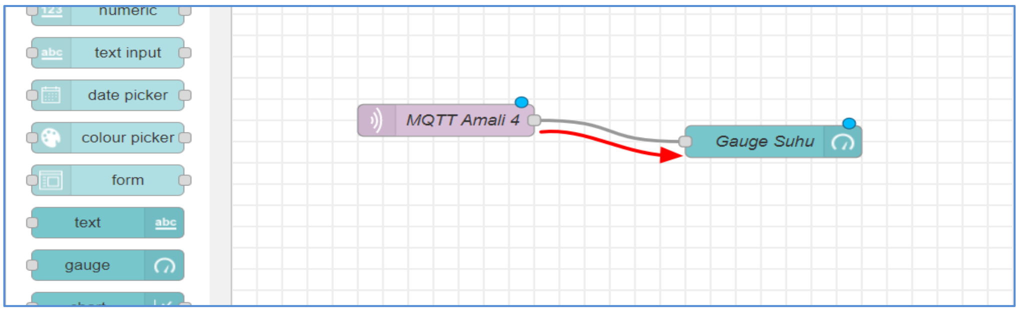


Figure B18

20. To execute the flows, click the **Deploy** button. Once the flow is successfully deployed, you will see the notification and **mqtt in** node will show connected status.

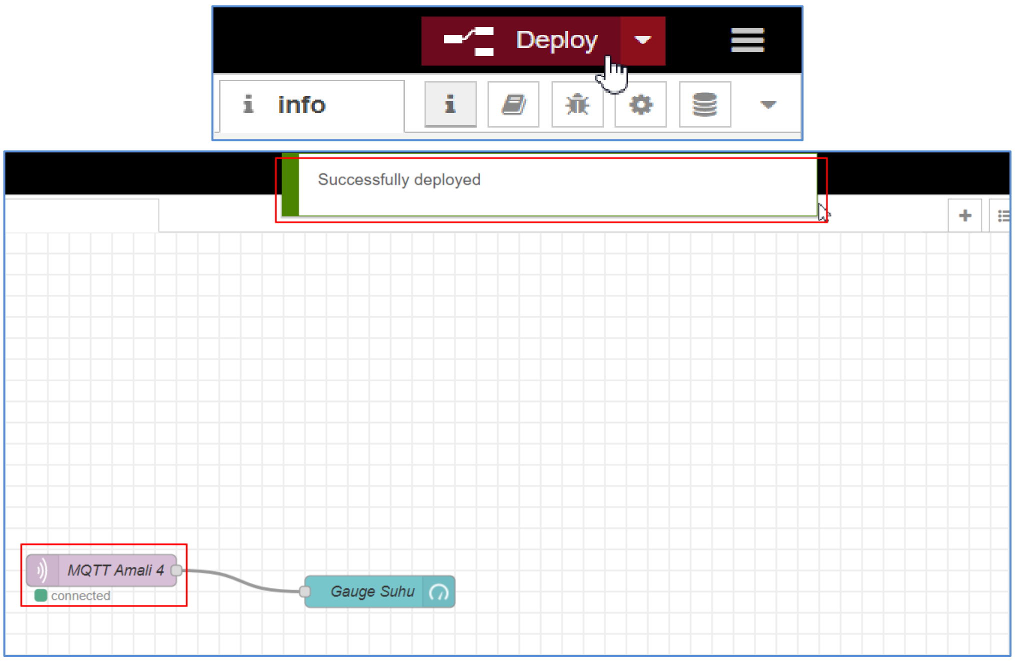


Figure B19

Accessing the Node-RED Dashboard

21. On the right sidebar, click the dropdown menu and click **Dashboard**.

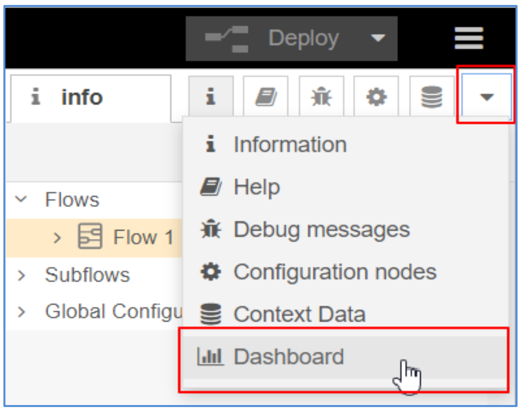


Figure B20

22. On the **dashboard** tab, click the icon to access the Node-RED Dashboard and you'll be forwarded to the Node-RED Dashboard page, where you will see the name of the Tab [Amali Internet of Things], the name of the Group [Monitoring] and the gauge.

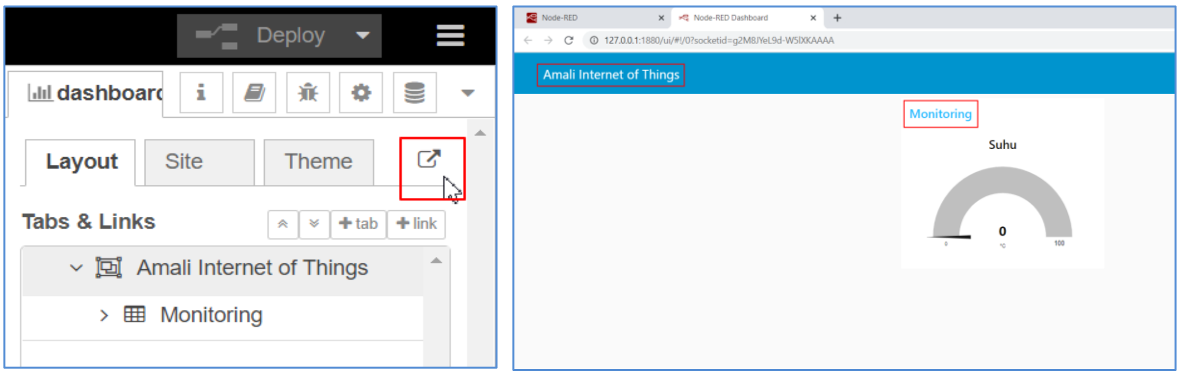


Figure B21

Publish Message/Payload using HiveMQ Websocket Client

23. On HiveMQ Websocket Client, type any value (for example: 60) on the **Message** field, then click the **Publish** button to publish the value.

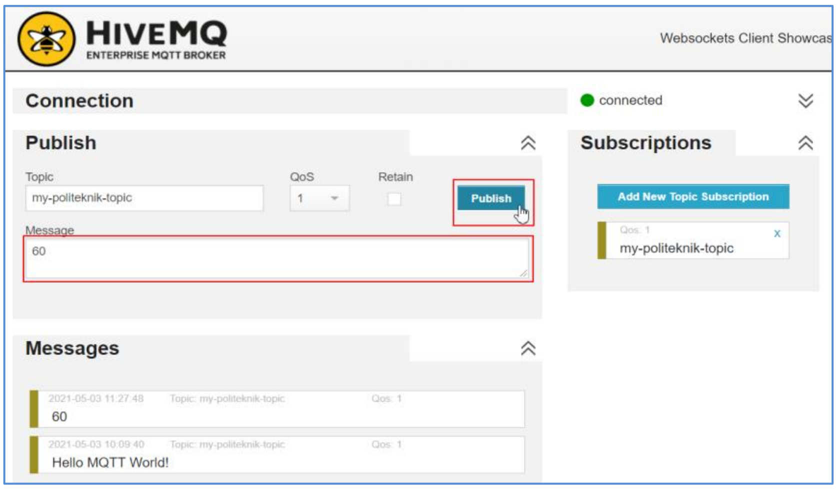


Figure B22

24. Once the message is successfully published the **Messages** section will show the value and on the Node-RED Dashboard the gauge [Suhu] animated the gauge meter based on the received value.

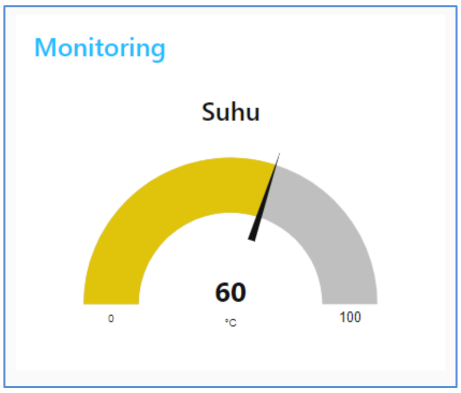


Figure B23

25. Try changing the publish value (for example: 89) and observe the changes on the gauge meter on the Node-RED Dashboard. Print screen the layout from your HiveMQ Websocket Client and Node-RED Dashboard as in Figure B24. Paste it in Table 2 as a result.

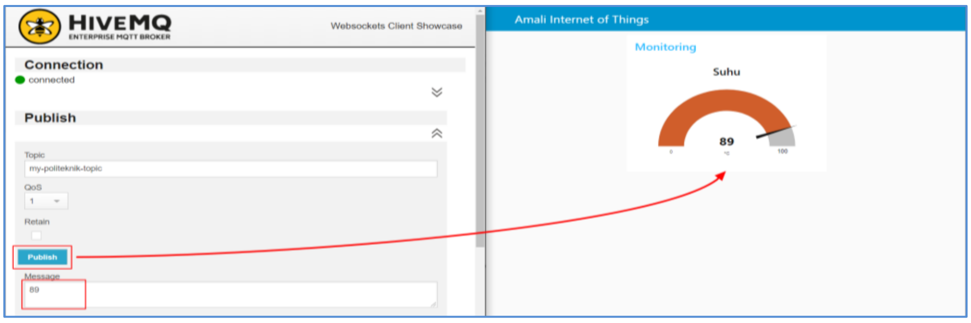


Figure B24

PART C: MQTT CONNECTIVITY USING IOT MQTT PANEL ANDROID APP

1. On your android smartphone, go to the Play Store, search "**mqtt panel**" and Install "**IoT MQTT Panel**" by Rahul Kundu.
2. Wait for the App installation to be done and open the App.

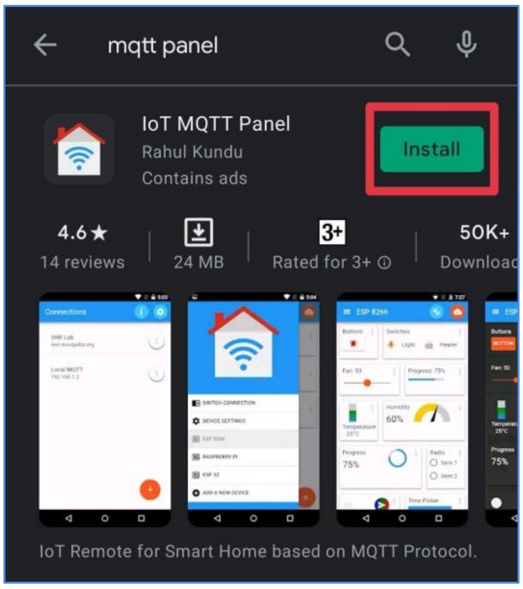


Figure C1

3. Tap on **SETUP A CONNECTION** button.

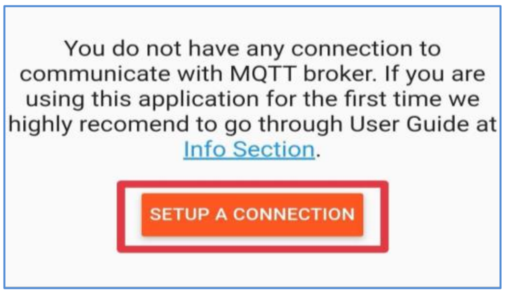
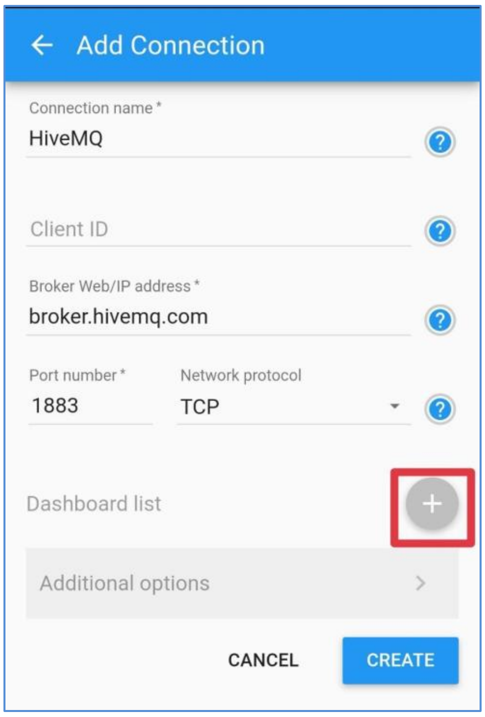

You do not have any connection to communicate with MQTT broker. If you are using this application for the first time we highly recomend to go through User Guide at Info Section.

Figure C2

4. Type any preferable title in the **Connection name** field (for example: **HiveMQ**), left blank **Client ID** field for auto-generated client id, Type broker host address (for example: broker.hivemq.com) in the **Broker Web/IP address** field, Type **1883** in the **Port number** field and **TCP** for **Network protocol**. Then, tap on **+** icon beside Dashboard list to set up the name of the Dashboard and click Add.



Add Connection

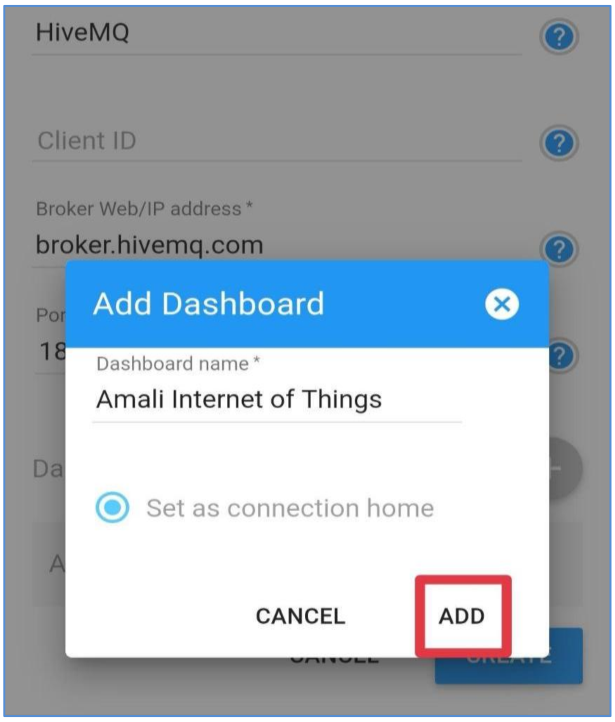
Connection name *: HiveMQ

Client ID

Broker Web/IP address *: broker.hivemq.com

Port number *: 1883

Network protocol: TCP



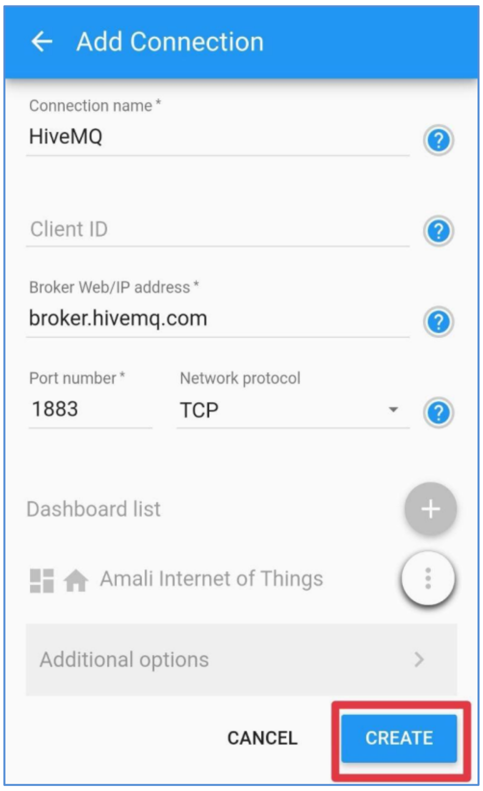
Add Dashboard

Dashboard name *: Amali Internet of Things

Set as connection home

Figure C3

5. Click the **Create** button.



Add Connection

Connection name *: HiveMQ

Client ID

Broker Web/IP address *: broker.hivemq.com

Port number *: 1883

Network protocol: TCP

Dashboard list: Amali Internet of Things

Figure C4

6. Tap on the **HiveMQ** (Figure C5). Then Click the **ADD PANEL** button (Figure C6).

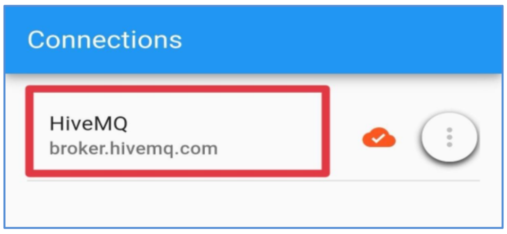


Figure C5

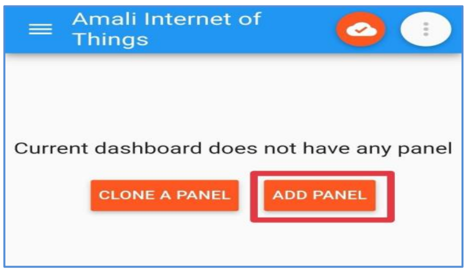


Figure C6

7. Scroll the list of available panels and tap on the **Gauge** panel.

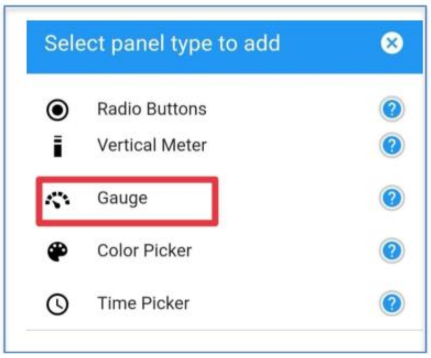
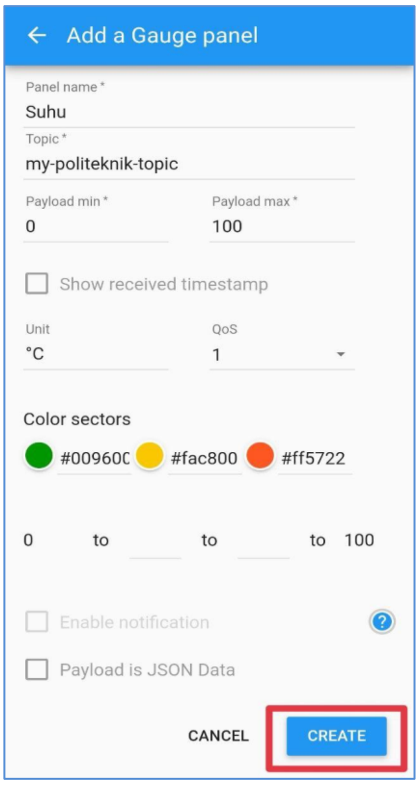


Figure C7

8. Insert the name of the panel (for example: **Suhu**) on **Panel name** field, insert the name of the topic (for example: **my-politeknik-topic**), minimum and maximum value of payload (for example: **0** and **100**) on **Payload min** and **Payload max** field, symbol of unit (for example: **°C**) on **Unit** field, QoS (for example: **1**) and tap the **Create** button.
9.



Panel name *
Suhu
Topic *
my-politeknik-topic
Payload min *
0
Payload max *
100
Show received timestamp
Unit
°C
QoS
1
Color sectors
#00960C #fac800 #ff5722
0 to to to 100
Enable notification
Payload is JSON Data
CANCEL CREATE

Figure C8

10. You will see the **Gauge** panel is available on the dashboard as below.

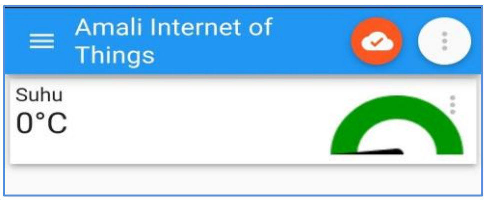


Figure C9

11. Click **+** icon at the bottom of the screen to add a **Text Input** panel.

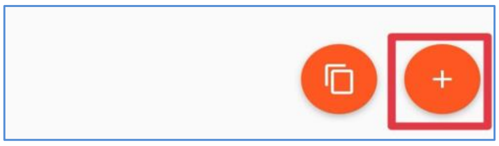


Figure C10

12. Scroll the list of available panels and tap on the **Text Input** panel.

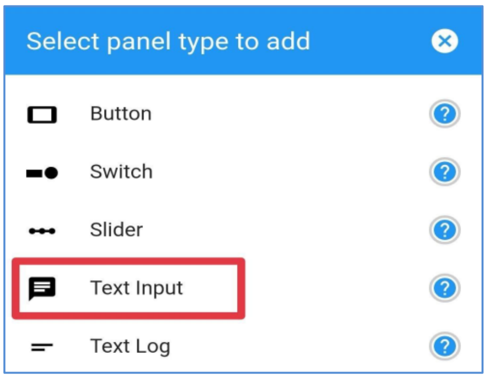
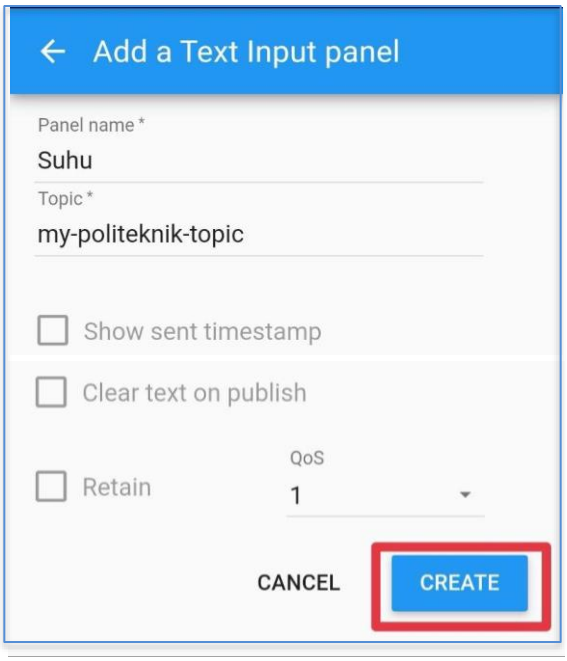


Figure C11

13. Insert the name of the panel **(Suhu)** and Topic (same as Step 34), QoS (1) and other options left default. Then, tap on the **Create** button.



Add a Text Input panel

Panel name *
Suhu

Topic *
my-politeknik-topic

☐ Show sent timestamp

☐ Clear text on publish

☐ Retain

QoS
1

Figure C12

14. You will see both **Gauge** and **Text Input** panel on the dashboard.

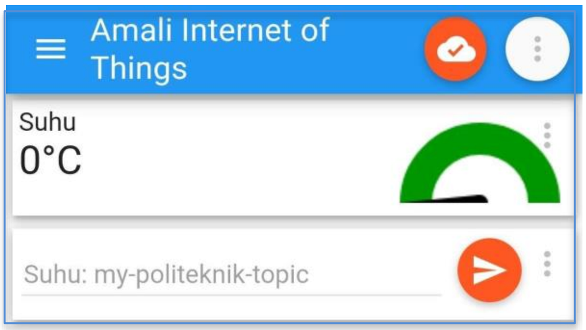


Figure C13

15. Try insert any value (for example: 67) on the **Text Input** panel and tap on the **Send** button. You will see the value will appear on the **Gauge** panel.

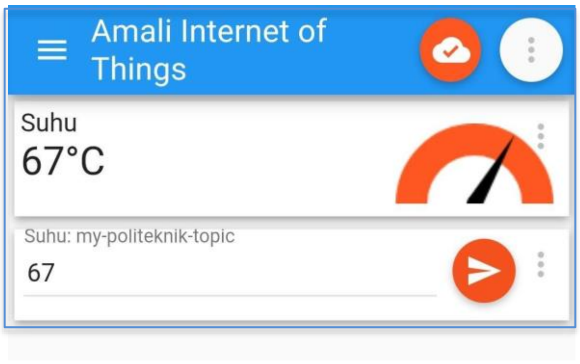


Figure C14

16. Go to Node-RED Dashboard, you will also see the **Gauge** value is based on publish value via the IoT MQTT Panel app.

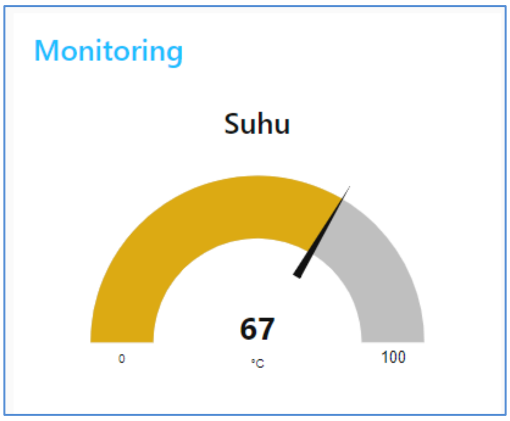


Figure C15

Add Graph Node to Node-RED Dashboard

17. Click, drag and release **graph** node into Node-RED workspace.

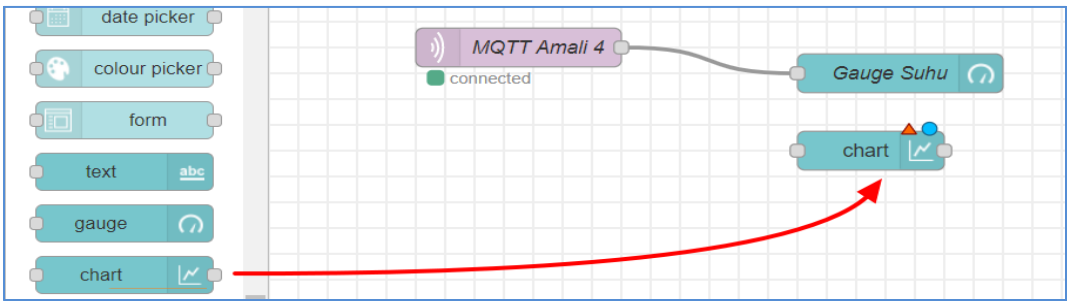
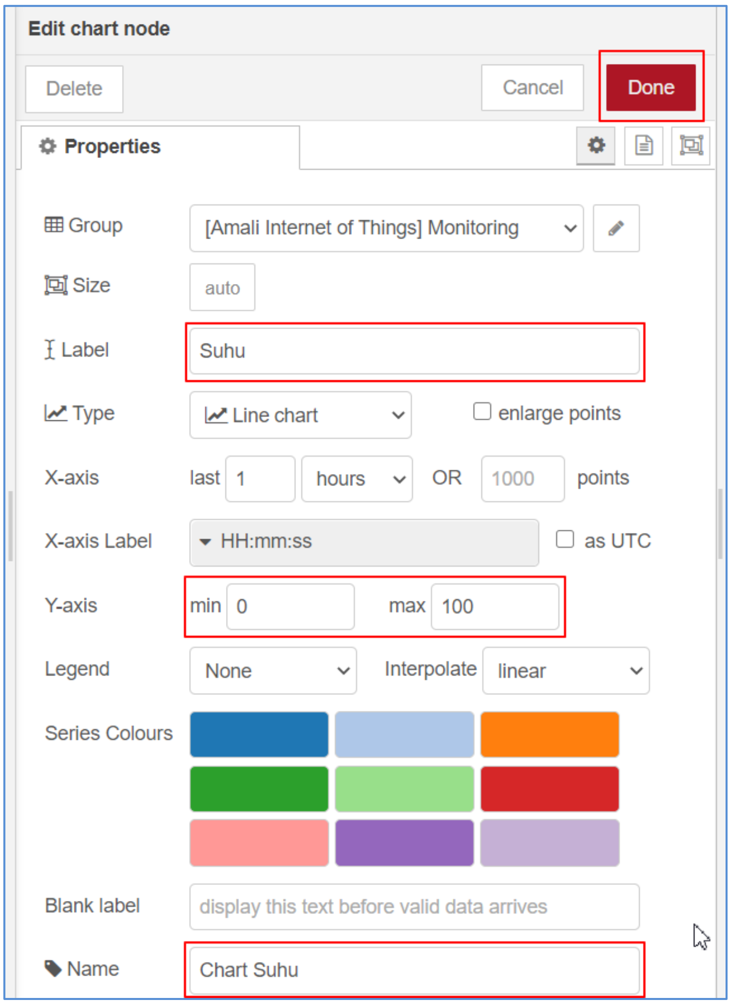


Figure C16

18. Double click the **chart** node. Type in the name of the **Label** (for example: **Suhu**), the minimum and maximum value of the chart (for example: **0** and **100**) and the **Name** of the gauge node (for example: **Chart Suhu**). Then, click the **Done** button.



Group: [Amali Internet of Things] Monitoring
Size: auto
Label: Suhu
Type: Line chart ☐ enlarge points
X-axis: last 1 hours OR 1000 points
X-axis Label: HH:mm:ss ☐ as UTC
Y-axis: min 0 max 100
Legend: None Interpolate linear
Series Colours: 🟦 🟦 🟧 🟩 🟩 🟥 🟥 🟪 🟪
Blank label: display this text before valid data arrives
Name: Chart Suhu

Figure C17

19. Connect the wires from **mqtt in** node [MQTT Amali 4] to **chart** node [Chart Suhu].

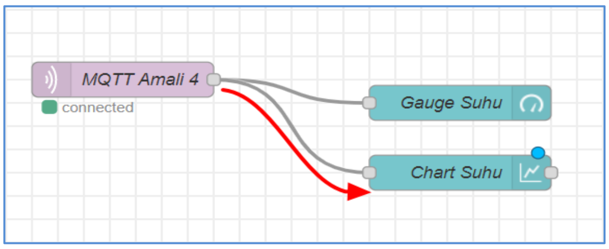


Figure C18

20. **Deploy** the flow and observe the difference on the Node-RED Dashboard, which now has the chart interface.

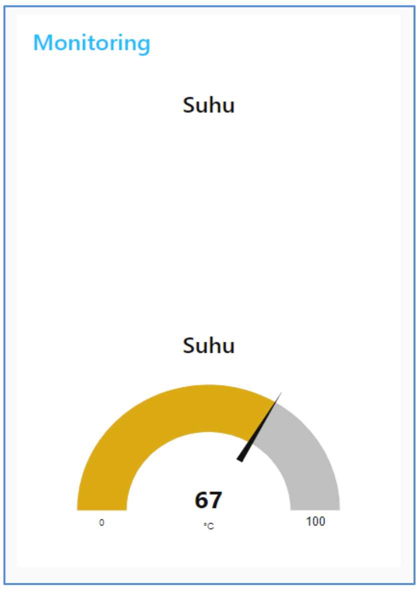


Figure C19

21. Try publishing several values from the **IoT MQTT Panel** app and observe the changes on the Node-RED Dashboard. Print screen the layout from your IoT MQTT Panel app and Node-RED Dashboard. Paste it in Table 3 as a result.

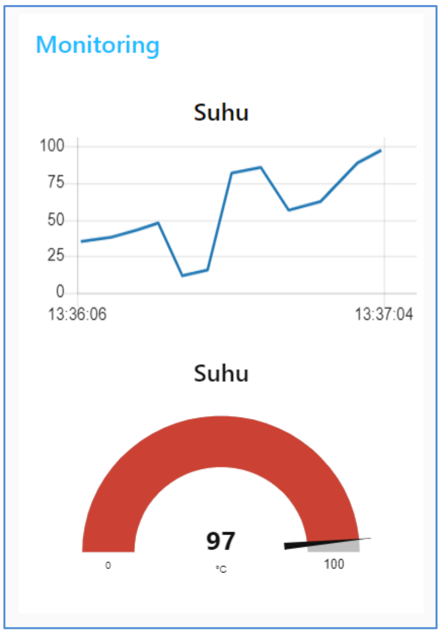


Figure C20

RESULT

Part A: Install Node-Red (Flow-Based IoT Programming Tools)

Step 6	Status of node-red packages added
Step 17	Layout from your node-red-dashboard palette

Table 1

Part B: MQTT Connectivity Using Hivemq MQTT Broker

Step 8	Layout from Websockets Client Showcase
Step 25	Print screen of HiveMQ Websocket Client and Node-RED Dashboard

Table 2

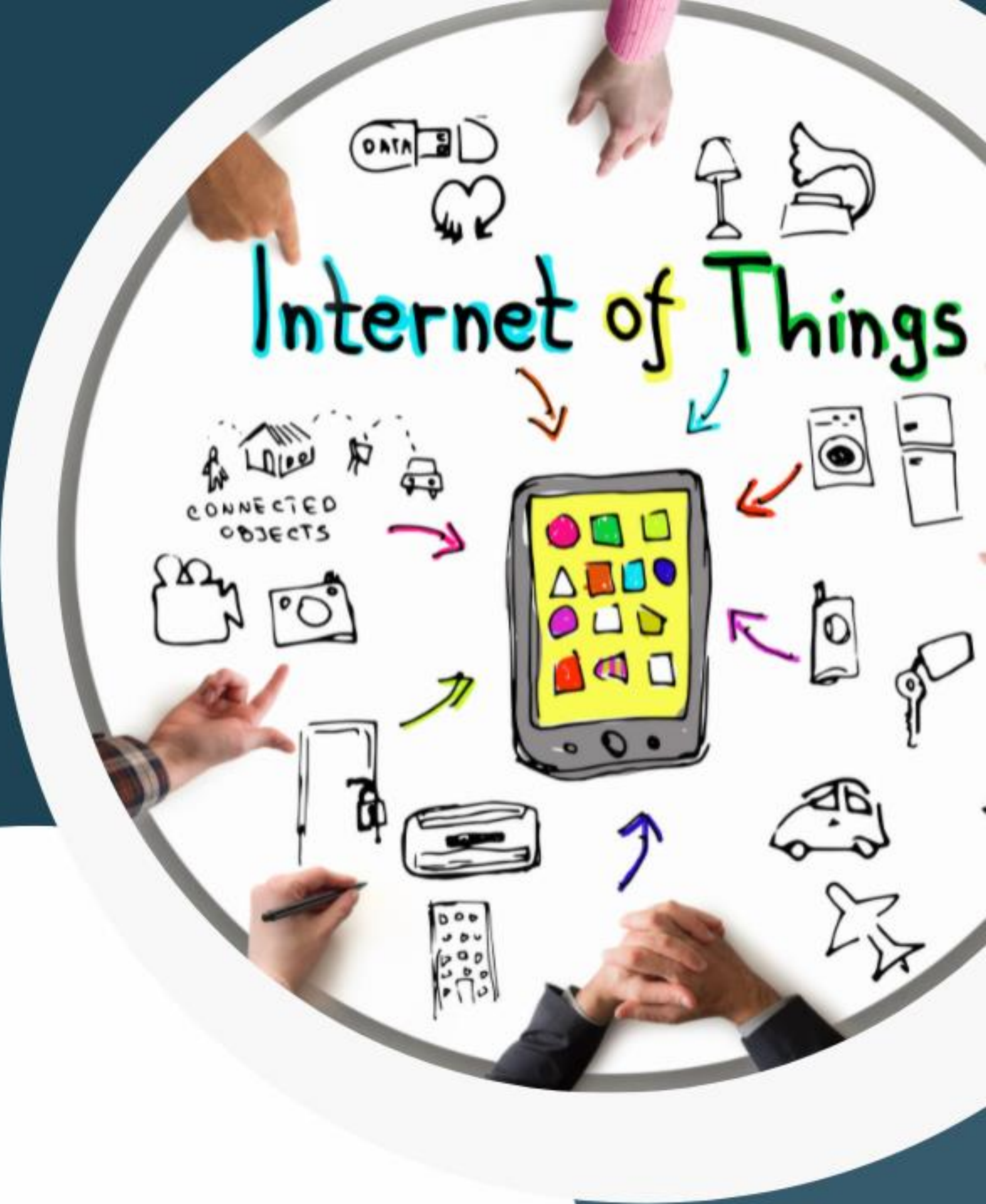
Part C: MQTT Connectivity Using IoT MQTT Panel Android App

Step 20	Print screen layout from your IoT MQTT Panel app and Node-RED Dashboard

Table 3

DISCUSSION

1. Discuss the connection between broker and client in MQTT protocol.
2. Explain Quality of Service (QoS) in MQTT. How does it work?



TUTORIAL 05

TUTORIAL: 5

Title: Node-RED Flow & Node-RED Dashboard

OBJECTIVES

Upon completion of this tutorial, you should be able to:

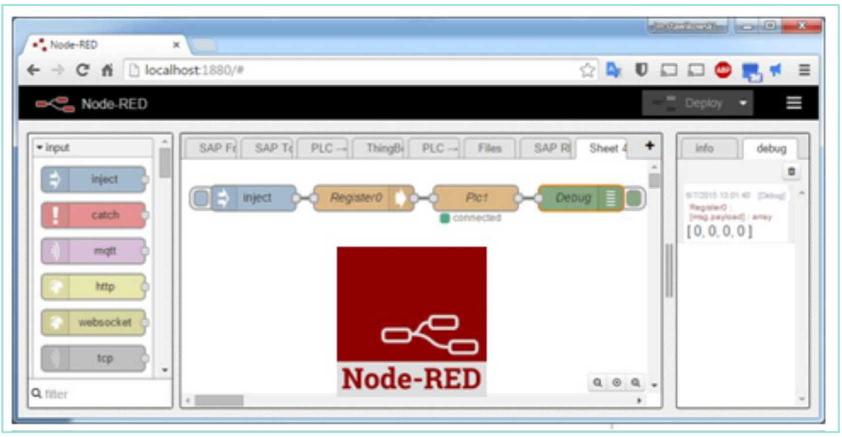
- install Node-RED, Node-RED Dashboard & IoT MQTT Panel
- construct, configure and deploy Node-RED flow and publish to Node-RED dashboard.
- construct and configure IoT MQTT Panel (mobile apps)
- connect Node-RED dashboard and mobile apps using HiveMQ MQTT broker

EQUIPMENTS

- Internet Connection
- Laptop/Personal Computer
- Android Smartphone

THEORY

Node RED is a visual programming for building workflows for IoT scenario. It offers an easiest way for wiring together hardware devices, APIs and online services. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.



PROCEDURE

PART A: NODE-RED DASHBOARD MOTOR CONTROL INTERFACE

Run Node-RED Locally

1. Open Command Prompt by clicking the Windows icon and type **cmd** and click Command Prompt.

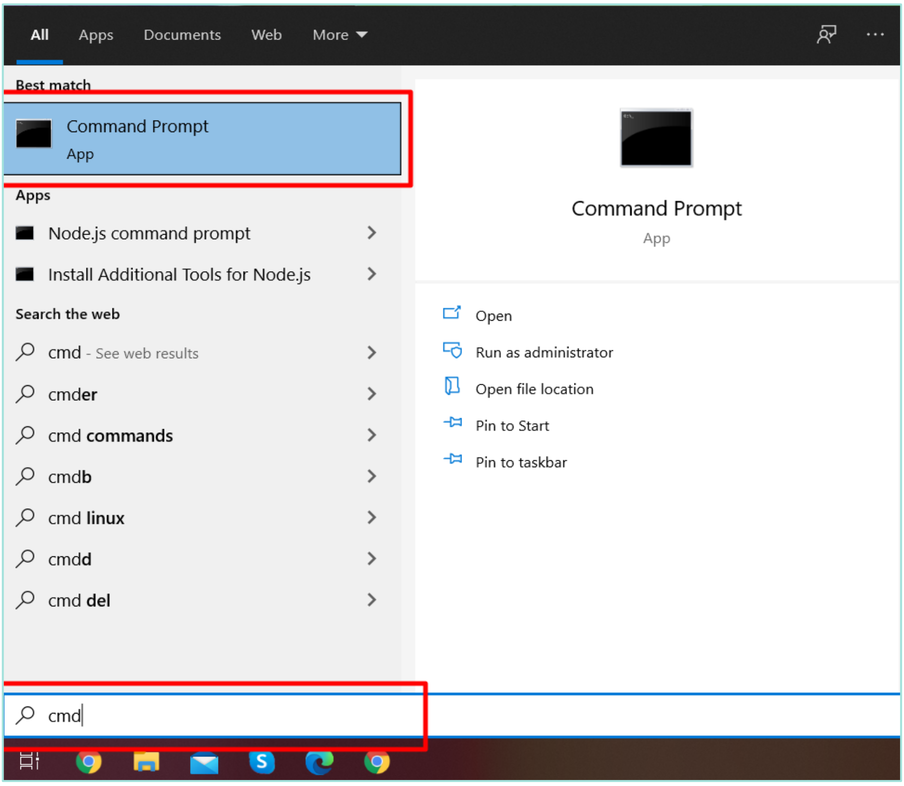
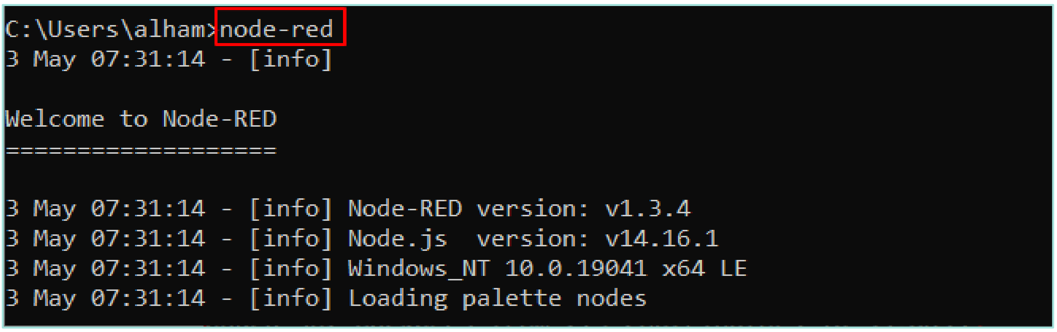


Figure A1

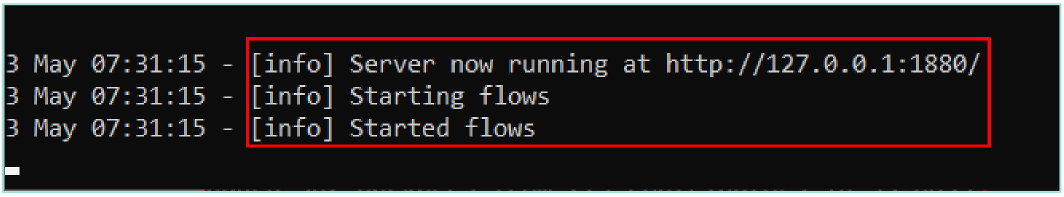
2. Type **node-red** and Enter to run Node-RED applications.



```
C:\Users\alham>node-red
3 May 07:31:14 - [info]

Welcome to Node-RED
===================

3 May 07:31:14 - [info] Node-RED version: v1.3.4
3 May 07:31:14 - [info] Node.js  version: v14.16.1
3 May 07:31:14 - [info] Windows_NT 10.0.19041 x64 LE
3 May 07:31:14 - [info] Loading palette nodes
```

Figure A2

3. Wait until the execution process is done, where you will see the status of **Server now running at** http://127.0.0.1:1880/.



```
3 May 07:31:15 - [info] Server now running at http://127.0.0.1:1880/
3 May 07:31:15 - [info] Starting flows
3 May 07:31:15 - [info] Started flows
```

Figure A3

4. Access Node-RED. Open your browser and type http://localhost:1880 or http://127.0.0.1:1880/ and Enter.

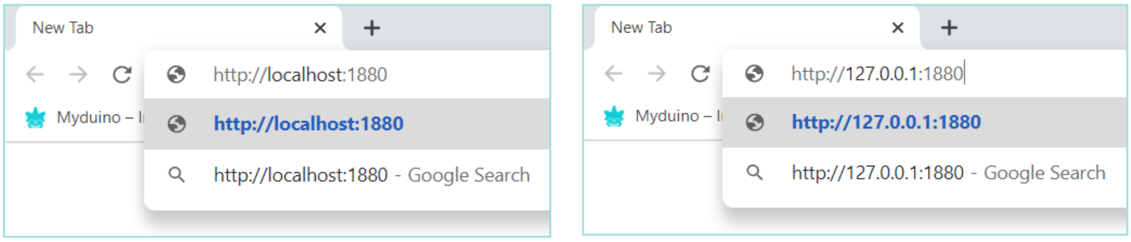


Figure A4

Button Node for Node-RED Dashboard

5. Under the **dashboard** palette, click **button** node, drag and release it on the Node-RED workspace.

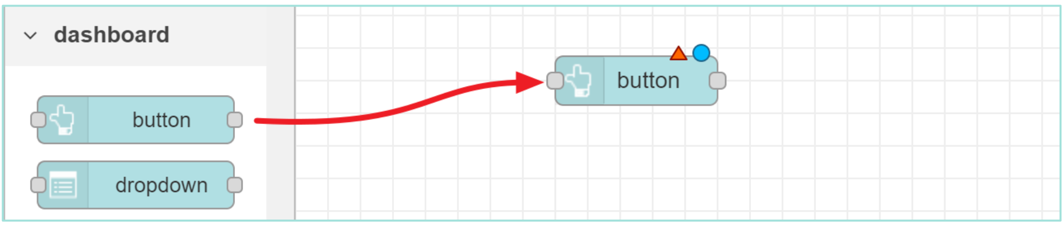
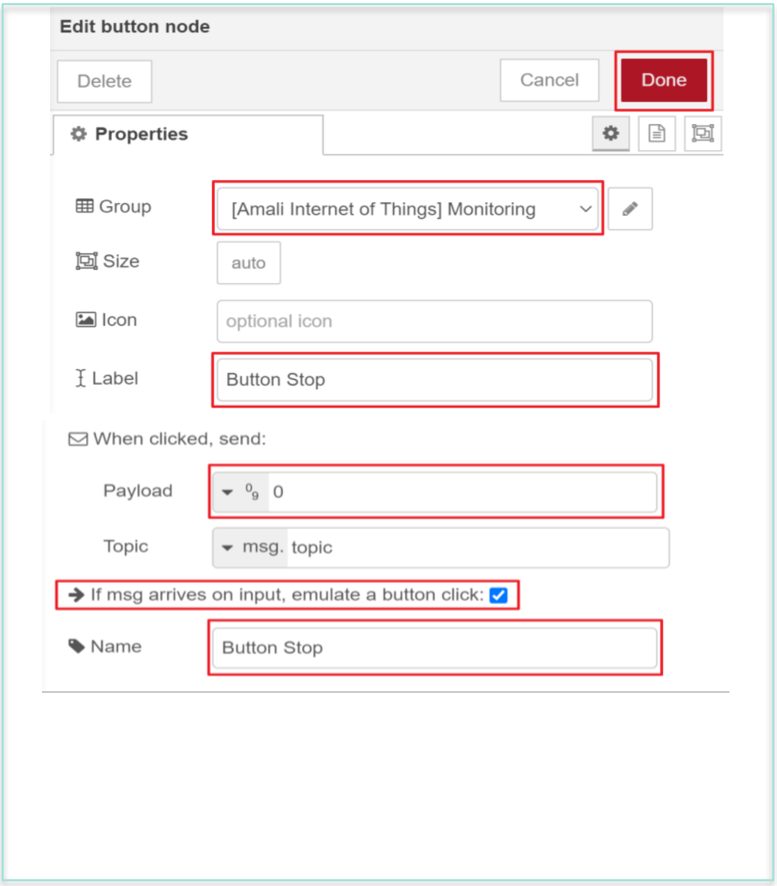


Figure A5

6. Set the **Group** name to **Monitoring** and **Tab** name to **Amali Internet of Things**. Replace the **Label** field title as **Button Stop**, the Payload as number **0**, **check** the emulate a button click and the **Name** of the gauge node is **Button Stop** and click the **Done** button.



Edit button node

Delete | Cancel | Done

Properties

Group: [Amali Internet of Things] Monitoring

Size: auto

Icon: optional icon

Label: Button Stop

When clicked, send:

Payload: 0

Topic: msg. topic

If msg arrives on input, emulate a button click: ✓

Name: Button Stop

Figure A6

7. Add 4 more buttons, respectively for 25%, 50%, 75% and 100% buttons as below.

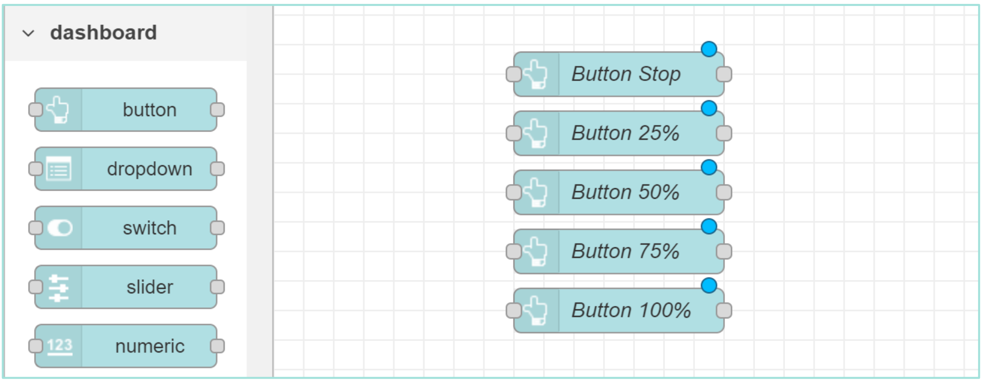
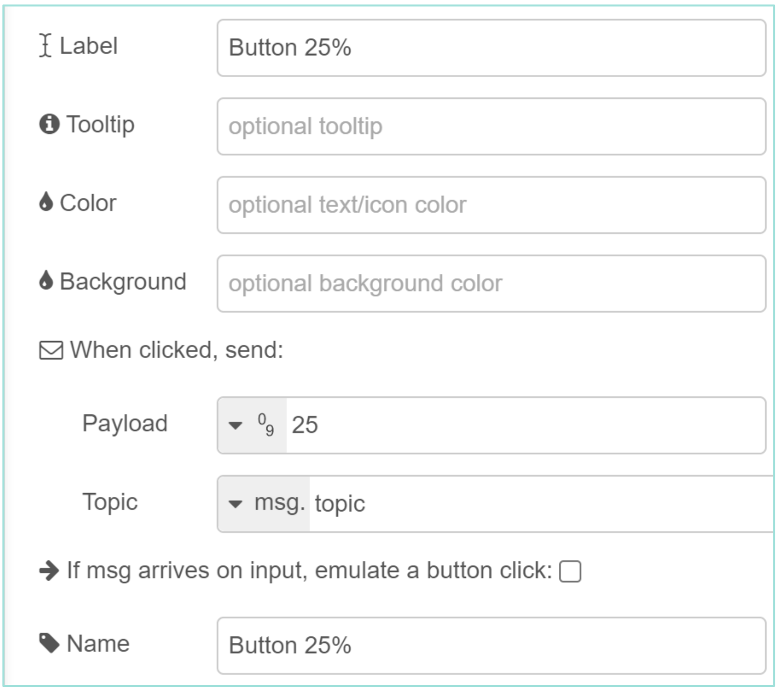


Figure A7

8. Each button's Payload number is set as the following value in Step 7. 25% button payload value is 25, same goes for 50% button payload value is 50, 75% button payload value is 75 and 100% button payload value is 100.



Label	Button 25%
Tooltip	optional tooltip
Color	optional text/icon color
Background	optional background color

When clicked, send:

Payload	25
Topic	msg. topic

If msg arrives on input, emulate a button click: ☐

Name	Button 25%

Figure A8

9. Deploy the flow and open the Node-RED dashboard.

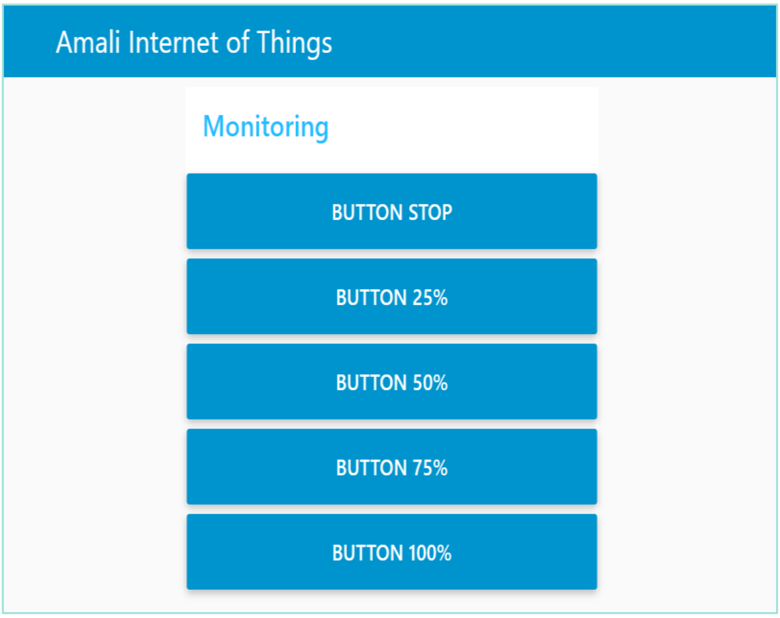
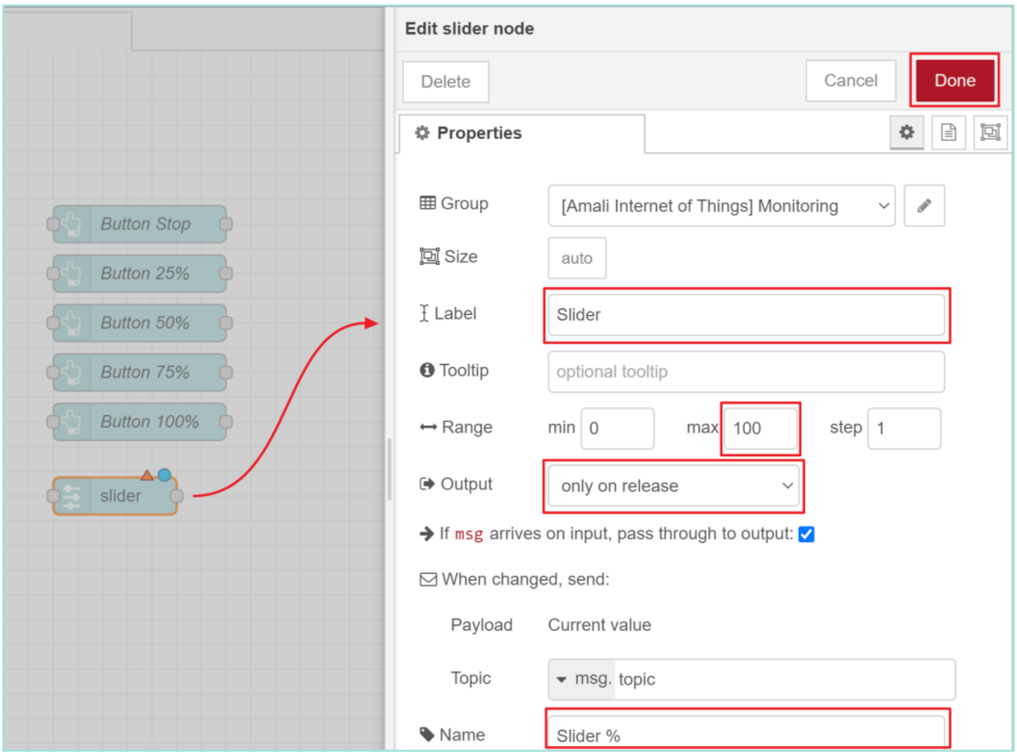


Figure A9

10. Add the slider node into the workspace and set the **Label** field title as **Slider**, the Range max value to **100**, the Output option as **only on release** and the **Name** of the slider node is **Slider %** and click the **Done** button.


Edit slider node

Delete Cancel Done

Properties

Group: [Amali Internet of Things] Monitoring
Size: auto
Label: Slider
Tooltip: optional tooltip
Range: min 0 max 100 step 1
Output: only on release
If msg arrives on input, pass through to output: ☑
When changed, send:
Payload: Current value
Topic: msg. topic
Name: Slider %

Figure A10

11. Deploy the flow and open the Node-RED dashboard.

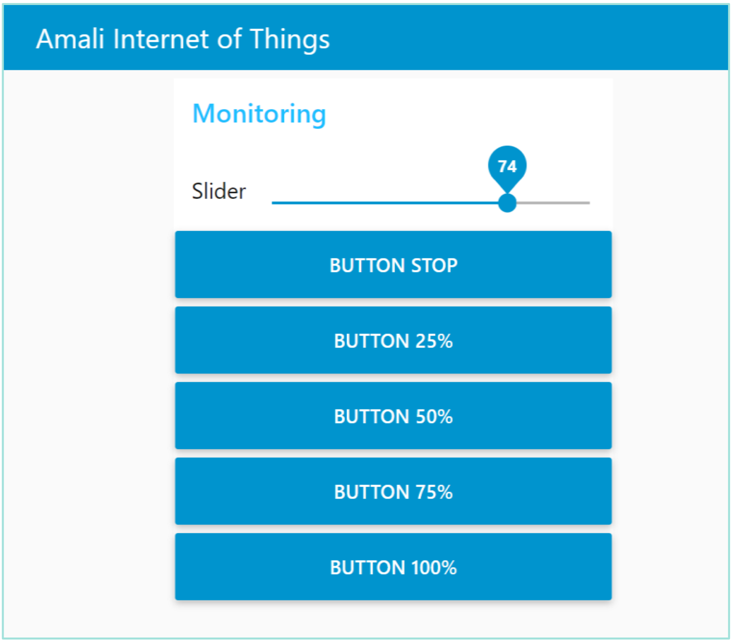
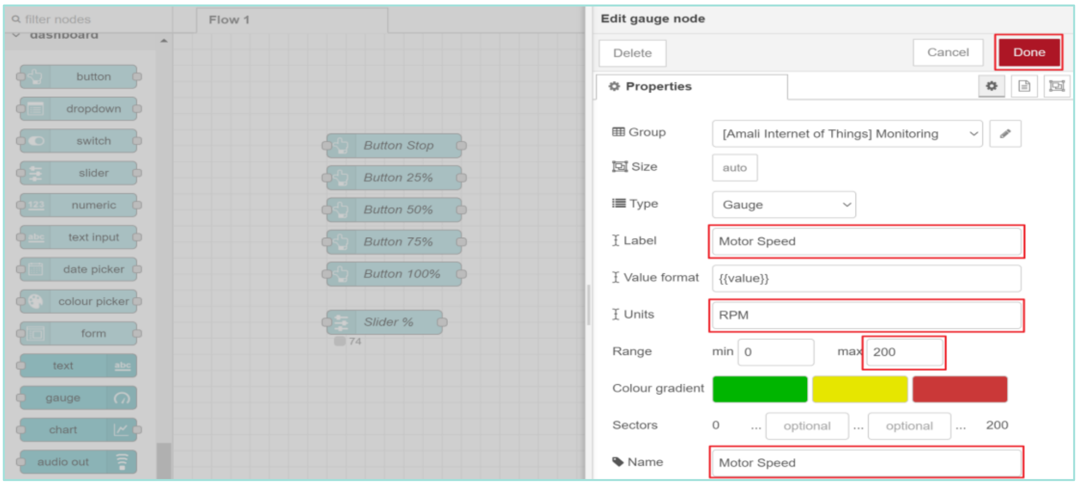


Figure A11

12. Add the gauge node into the Node-RED workspace and set the gauge node **Label** as **Motor Speed**, **Units** as **RPM**, **Range** max to **200** and **Name** of the gauge node as **Motor Speed** and click the Done button.



Group [Amali Internet of Things] Monitoring
Size auto
Type Gauge
Label Motor Speed
Value format {{value}}
Units RPM
Range min 0 max 200
Colour gradient 🟩 🟨 🟥
Sectors 0 ... optional ... optional ... 200
Name Motor Speed

Figure A12

13. Under the **function** palette, click the **range** node, drag and release it on the Node-RED workspace, between the button's node and the gauge node.

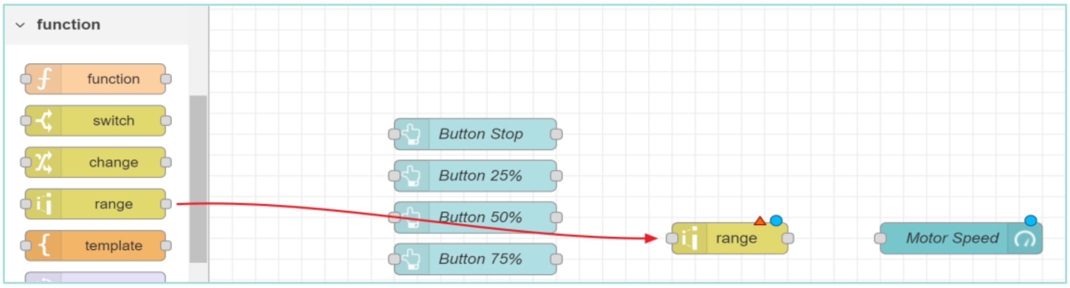
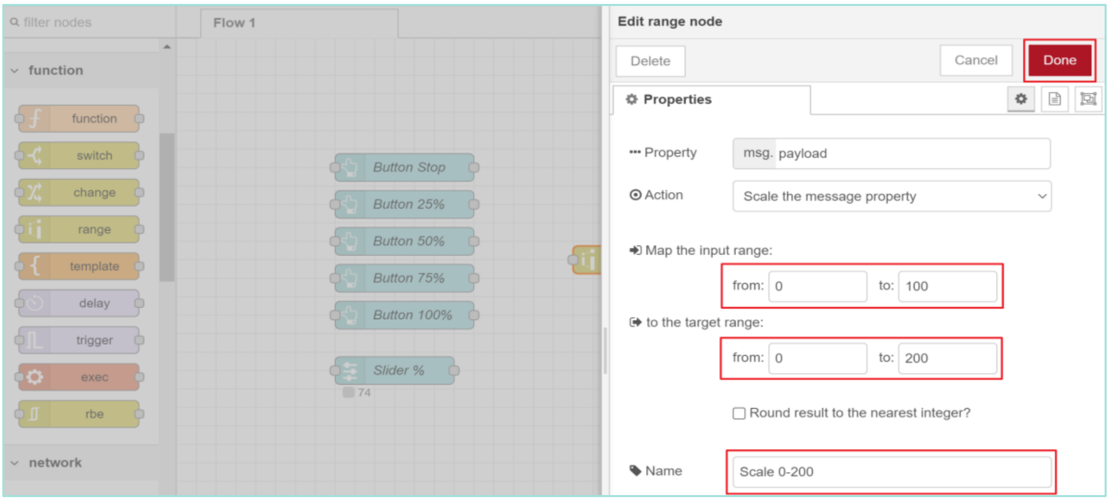


Figure A13

14. Double click the range node to set the **input range** from **0 to 100**, to the **target range** from **0 to 200** and the **Name** of the node is **Scale 0 to 200**. Click the Done button.



Property msg. payload
Action Scale the message property
Map the input range:
from: 0 to: 100
to the target range:
from: 0 to: 200
☐ Round result to the nearest integer?
Name Scale 0-200

Figure A14

15. Wires all the nodes as follows.

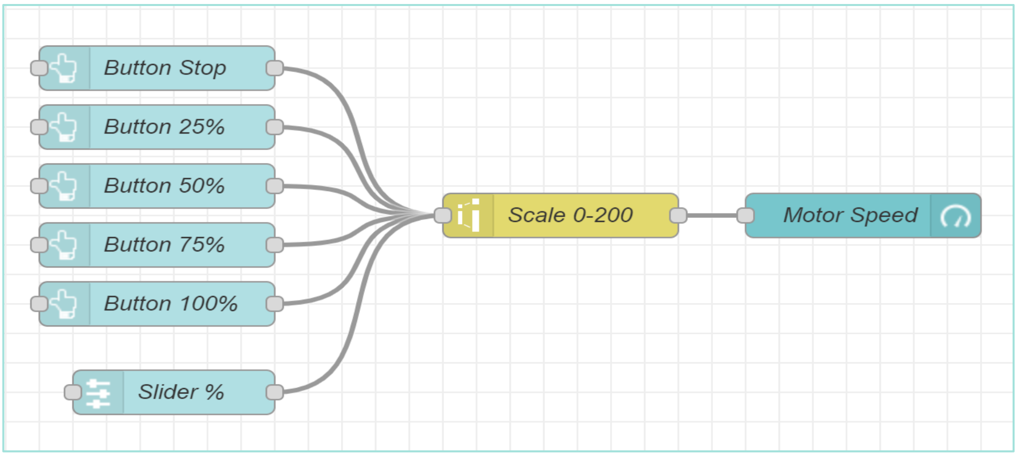


Figure A15

16. Deploy the flows and go to the Node-RED Dashboard and test the range slider and observe the value change on the gauge. Print screen the layout from your NodeRED Dashboard. Paste it in Table 1as a result.

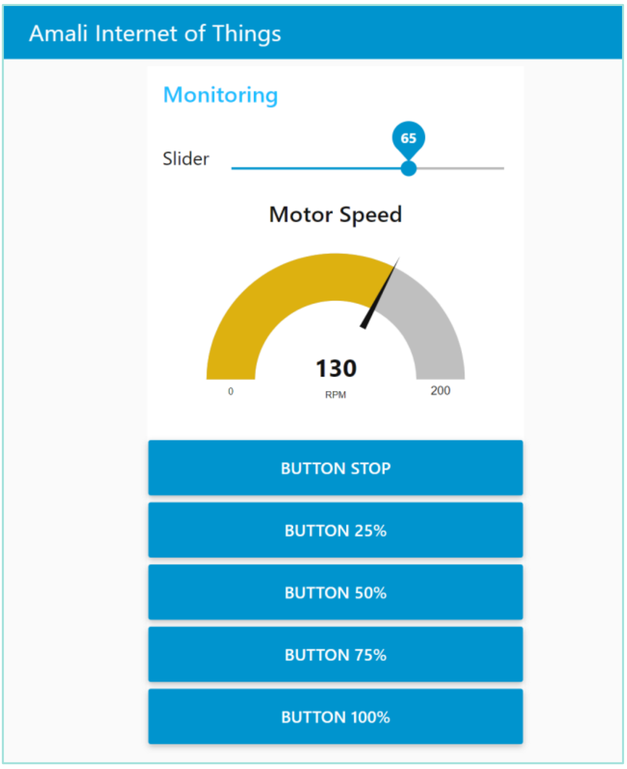


Figure A16

MQTT connectivity

17. Under the **network** palette, click **mqtt in** node, drag and release it on the Node-RED workspace. Set the **Server** as **broker.hivemq.com:1883**, the MQTT topic **my-politeknik-topic/bs** on the **Topic** field, change the QoS to **1**, **Name** of the node as **Button Stop** and click the **Done** button.

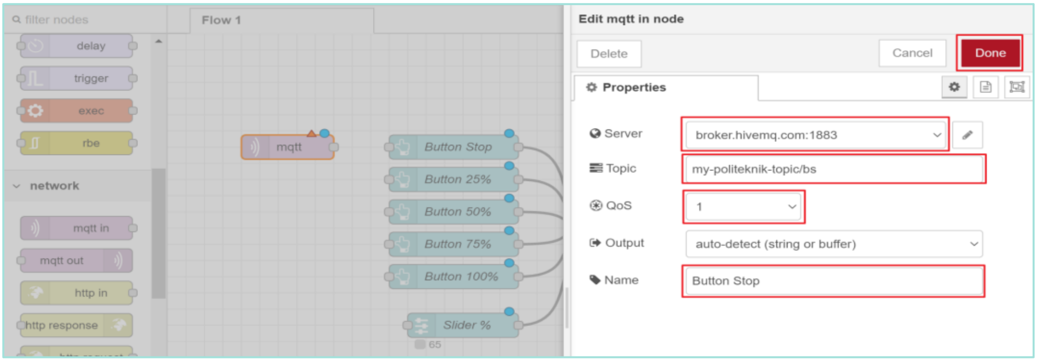

Edit mqtt in node
Server: broker.hivemq.com:1883
Topic: my-politeknik-topic/bs
QoS: 1
Output: auto-detect (string or buffer)
Name: Button Stop

Figure A17

18. Add 5 more **mqtt in** node into the Node-RED workspace, respectively for 25% topic (**my-politeknik-topic/bs25**), 50% (**my-politeknik-topic/bs50**), 75% (**my-politeknik-topic/bs75**), 100% (**my-politeknik-topic/bs100**) buttons and slider (**my-politeknik-topic/bslider**) as below.

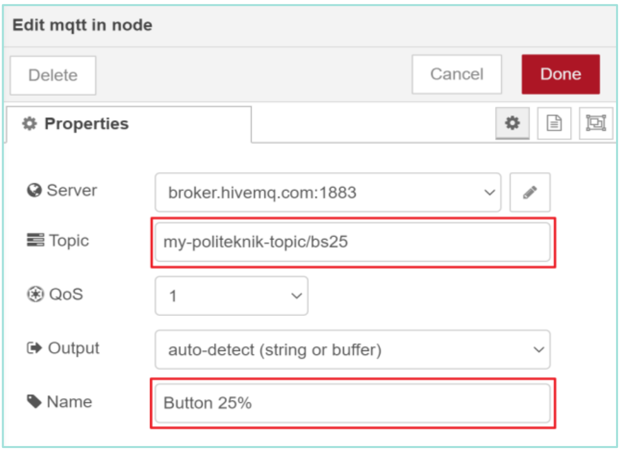

Edit mqtt in node
Server: broker.hivemq.com:1883
Topic: my-politeknik-topic/bs25
QoS: 1
Output: auto-detect (string or buffer)
Name: Button 25%

Figure A18

19. Connect each **mqtt in** the node to the respected button and slider node as below. Deploy the flows.

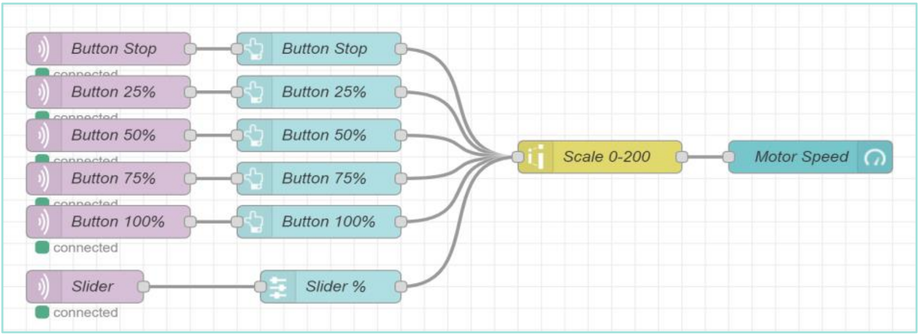


Figure A19

PART B: IOT MQTT PANEL MOTOR CONTROL INTERFACE

Connect to HiveMQ MQTT Broker

1. Open IOT MQTT Panel from your Android Smartphone.
2. Tap on **HiveMQ**.

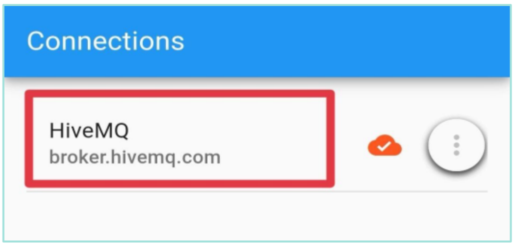


Figure B1

3. Click the **ADD PANEL** button.

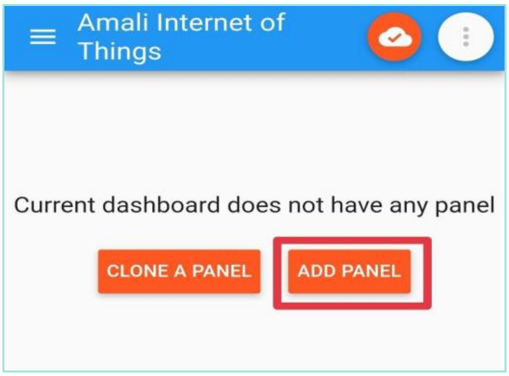


Figure B2

4. Scroll the list of available panels and tap on the **Button** panel.

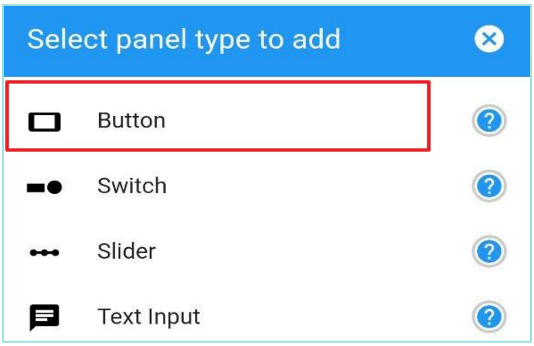
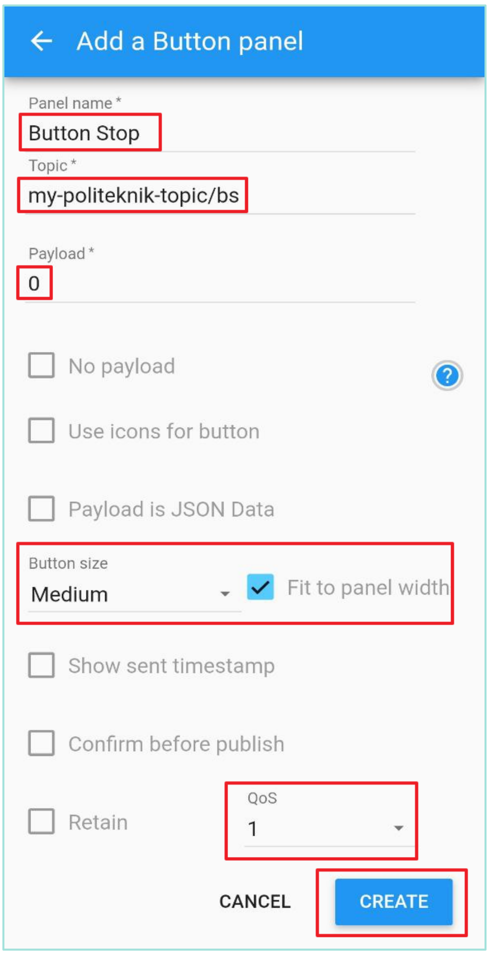


Figure B3

5. Insert the name of the panel (**Button Stop**) on **Panel name** field, insert the name of the topic (**my-politeknik-topic/bs**), payload (**0**) on **Payload** field, Medium button size and fit to panel width, QoS (**1**) and tap the **Create** button.



Add a Button panel

Panel name *
Button Stop

Topic *
my-politeknik-topic/bs

Payload *
0

☐ No payload
☐ Use icons for button
☐ Payload is JSON Data

Button size
Medium
☑ Fit to panel width

☐ Show sent timestamp
☐ Confirm before publish
☐ Retain

QoS
1

Figure B4

6. You will see the **Button** panel is available on the dashboard as below.

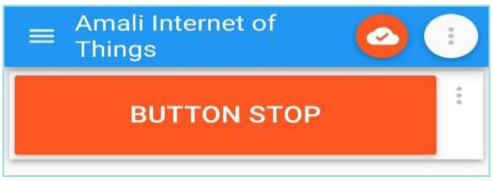


Figure B5

7. Add 4 more buttons, respectively for the topic: Button 25% topic (**my-politeknik-topic/bs25**), Button 50% (**my-politeknik-topic/bs50**), Button 75% (**my-politeknik-topic/bs75**) and Button 100% (**my-politeknik-topic/bs100**) buttons as below.

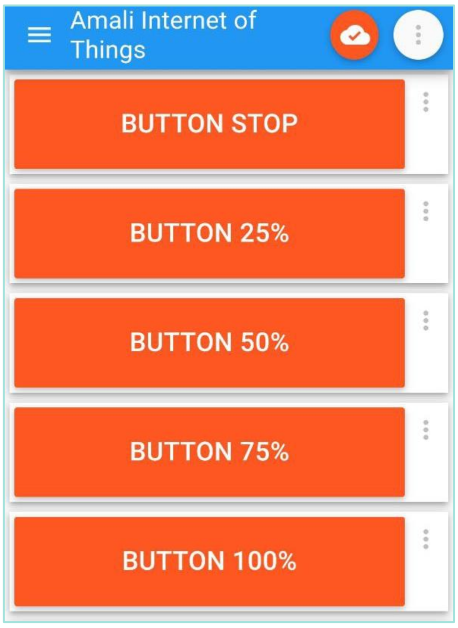
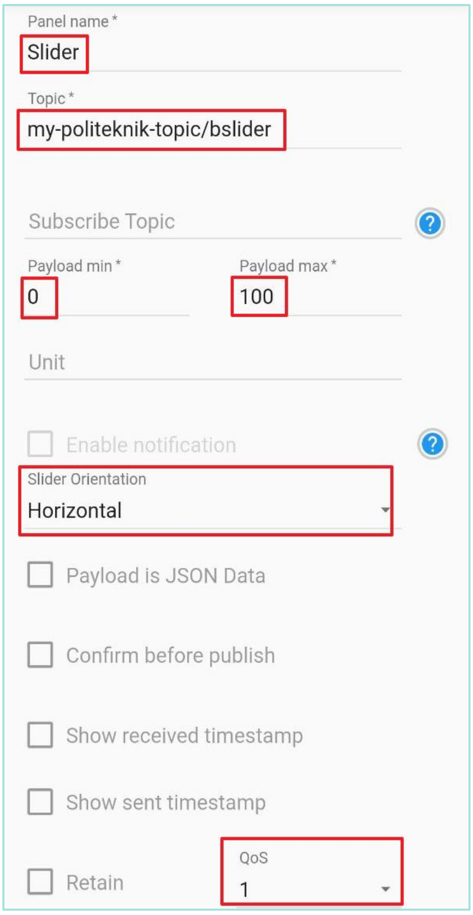


Figure B6

8. Add a slider panel into the dashboard and the setting as below, where the topic is (**my-politeknik-topic/bslider**) and tap the **Create** button.



Panel name *
Slider

Topic *
my-politeknik-topic/bslider

Subscribe Topic

Payload min *
0

Payload max *
100

Unit

☐ Enable notification

Slider Orientation
Horizontal

☐ Payload is JSON Data

☐ Confirm before publish

☐ Show received timestamp

☐ Show sent timestamp

☐ Retain

QoS
1

Figure B7

9. Try sliding the slider and tap the Button, you will see the result on the Node-RED dashboard gauge changes. Print screen the layout from your IoT MQTT Panel app and Node-RED Dashboard. Paste it in Table 2 as a result.

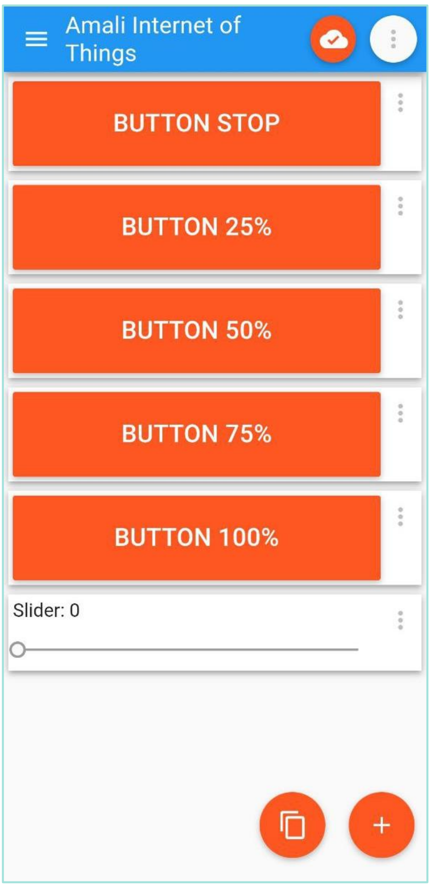


Figure B8

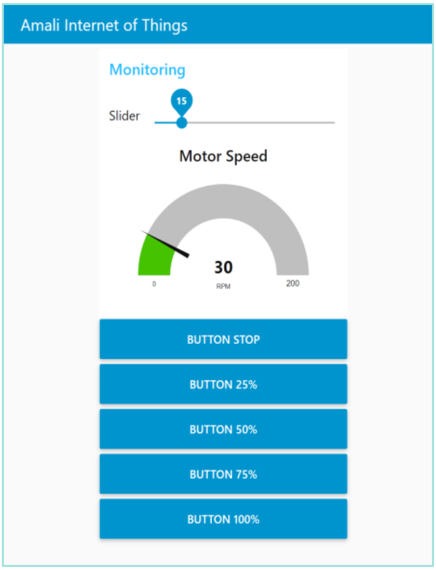


Figure B9

RESULT

PART A: Node-RED Dashboard Motor Control Interface

Step 16: Test the range slider and observe the value change on the gauge.

Button clicked/ Slider	Print screen the Node-RED Dashboard
Button STOP	
Button 25%	
Button 75%	
Button 100%	
Slider	

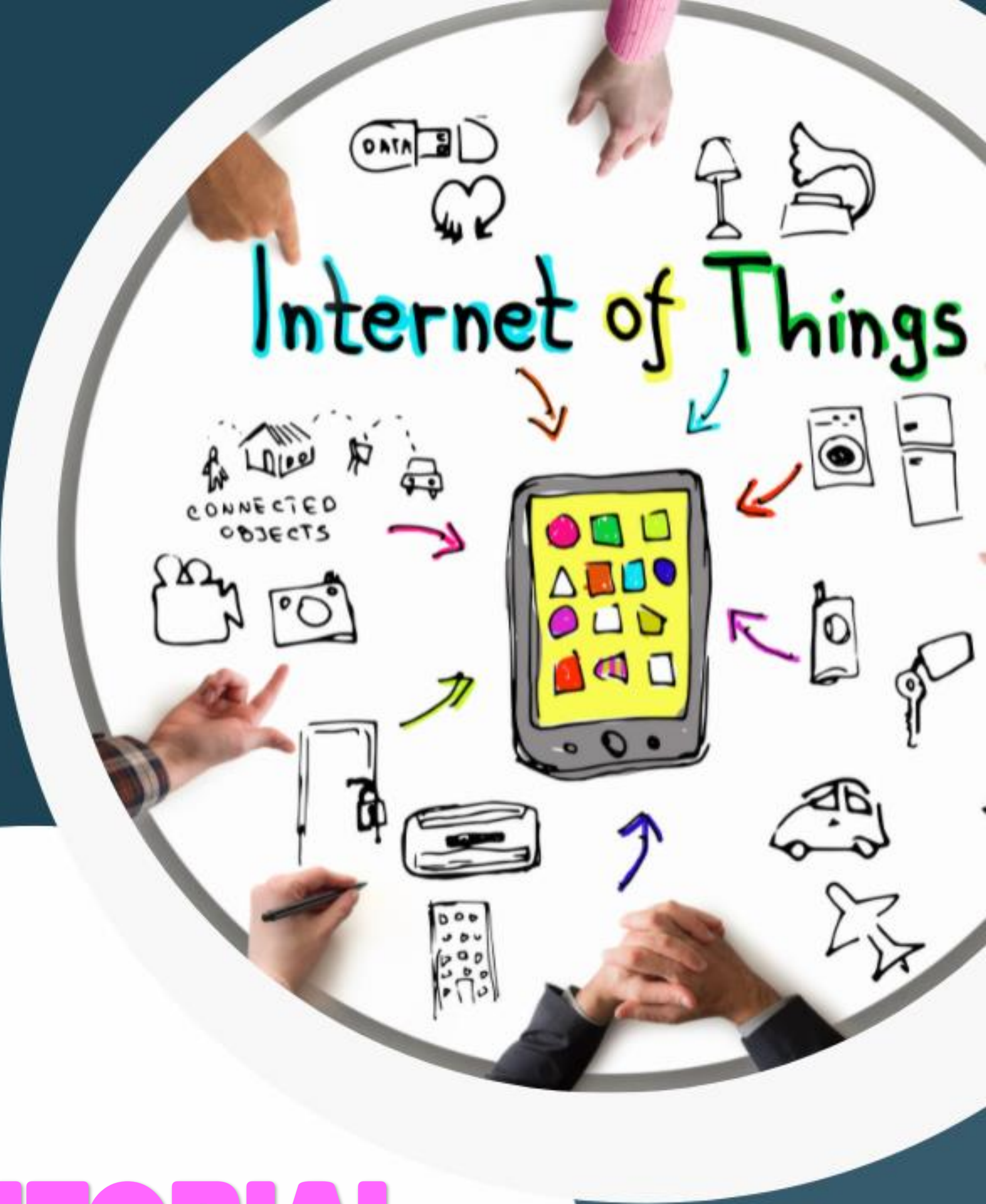
Table 1

PART B: IoT MQTT Panel Motor Control Interface

Step 9: Layout from your IoT MQTT Panel app and Node-RED Dashboard

DISCUSSION

1. Explain the process of publish and subscribe in MQTT protocol.
2. Explain function for publishing and subscribing to MQTT broker



TUTORIAL 06

TUTORIAL: 6

Title: IoT Application Control Via Smartphone

OBJECTIVES

Upon completion of this tutorial, you should be able to:

- Setup NodeMCU 8266 with circuit
- Write codes using Arduino.ide to established connection to internet
- Controlling DHT11 using Blynk App over internet

EQUIPMENTS

- NodeMCU ESp8266 Wi-Fi Dev. Board
- Micro USB cable to upload Code
- 330r resistor, DHT11 Temperature Sensor
- Android App (Blynk App)
- Arduino IDE
- Internet connection

THEORY

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many things. There are three major components in the platform:

- **Blynk App** - allows to you create amazing interfaces for your projects using various widgets we provide.
- **Blynk Server** - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private Blynk server locally.
- **Blynk Libraries** - for all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

WIDGET JUSTIFICATION:

- Buttons: There are a couple of buttons: one to activate light bulb, and the other one to activate the alarm buzzer, both are independent.
- LEDs: There are four LED widgets, indicating the status of each sensor, such as: movement sensor (as M.S.), doorbell button (D.B.B.), door sensor (D.S.) and window sensor (W.S.)
- Tabs: Enable optional tabs to organize the widgets better.
- Email: Enable email notifications
- Notifications: Enable smartphone notifications
- Slider: one slider to set the counter data value (0 to 248 on this case) of the CNT6/DLY6 in GP Design
- Gauge: One gauge to show the level of water of the home's tank
- Graph: One graph to show the level of water of the home's tank (same as gauge widget).
- History Graph: to show data statistics of the water tank level through the time (from hours to months of data storage)

PROCEDURE

PART A: ARDUINO LIBRARY INSTALLATION

Blynk

1. Open Arduino IDE.
2. Open Library Manager. From menu, select ***Tools > Manage Libraries...***

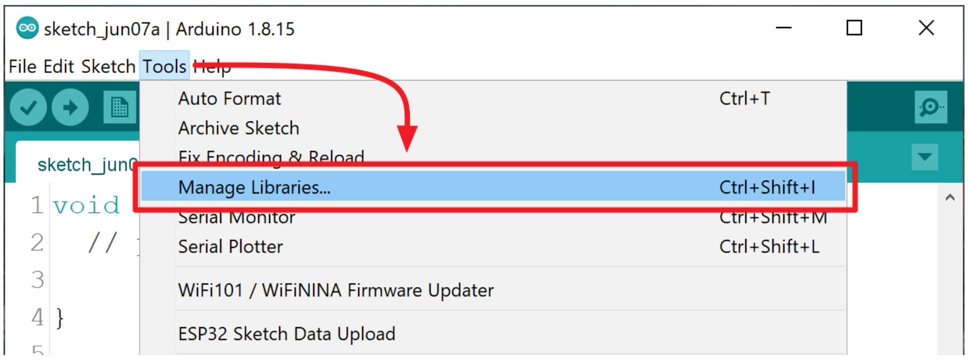
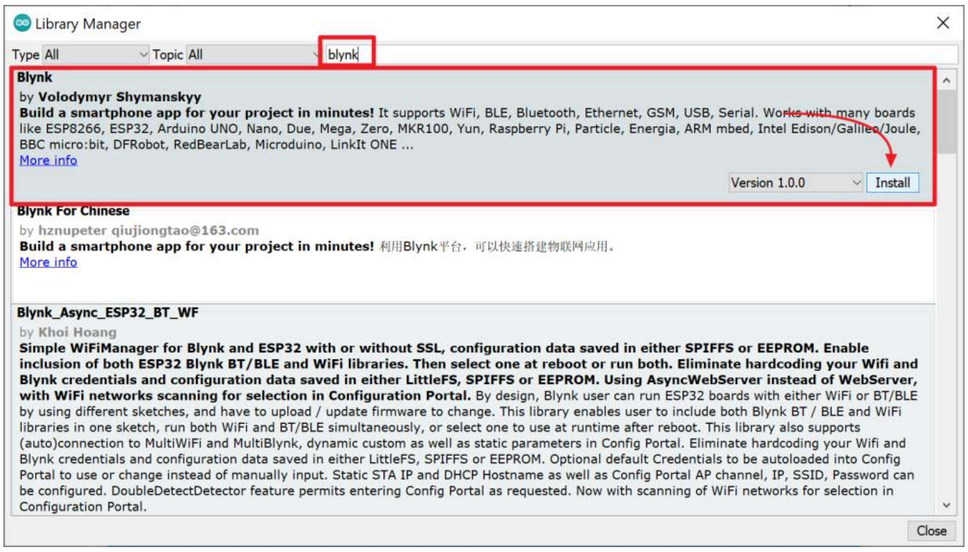

sketch_jun07a | Arduino 1.8.15
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
ESP32 Sketch Data Upload
sketch_jun0
1 void
2 //
3
4 }

Figure A1

3. On Library Manager, filter the list of libraries by typing **blynk** on *Filter your search* … field, then look for the name of library **Blynk *by Volodymyr Shymanskyy*** and click **Install** button to install the library. Wait for the installation to be completed.



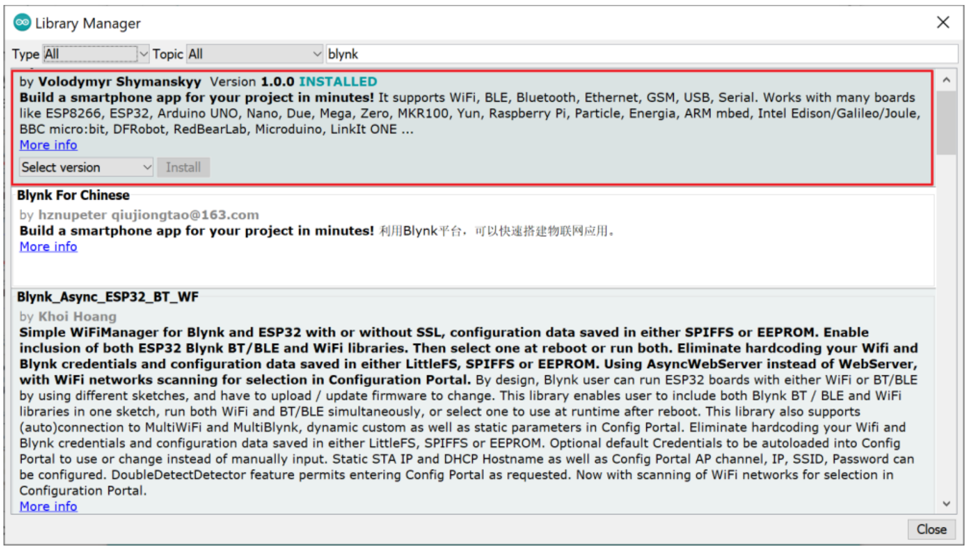
Blynk
by **Volodymyr Shymanskyy**
Build a smartphone app for your project in minutes! It supports WiFi, BLE, Bluetooth, Ethernet, GSM, USB, Serial. Works with many boards like ESP8266, ESP32, Arduino UNO, Nano, Due, Mega, Zero, MKR100, Yun, Raspberry Pi, Particle, Energia, ARM mbed, Intel Edison/Galileo/Joule, BBC micro:bit, DFRobot, RedBearLab, Microduino, LinkIt ONE ...
More info

Blynk For Chinese
by **hznupeter qiujiongtao@163.com**
Build a smartphone app for your project in minutes! 利用Blynk平台，可以快速搭建物联网应用。
More info

Blynk_Async_ESP32_BT_WF
by **Khoi Hoang**
Simple WiFiManager for Blynk and ESP32 with or without SSL, configuration data saved in either SPIFFS or EEPROM. Enable inclusion of both ESP32 Blynk BT/BLE and WiFi libraries. Then select one at reboot or run both. Eliminate hardcoding your Wifi and Blynk credentials and configuration data saved in either LittleFS, SPIFFS or EEPROM. Using AsyncWebServer instead of WebServer, with WiFi networks scanning for selection in Configuration Portal. By design, Blynk user can run ESP32 boards with either WiFi or BT/BLE by using different sketches, and have to upload / update firmware to change. This library enables user to include both Blynk BT / BLE and WiFi libraries in one sketch, run both WiFi and BT/BLE simultaneously, or select one to use at runtime after reboot. This library also supports (auto)connection to MultiWiFi and MultiBlynk, dynamic custom as well as static parameters in Config Portal. Eliminate hardcoding your Wifi and Blynk credentials and configuration data saved in either LittleFS, SPIFFS or EEPROM. Optional default Credentials to be autoloaded into Config Portal to use or change instead of manually input. Static STA IP and DHCP Hostname as well as Config Portal AP channel, IP, SSID, Password can be configured. DoubleDetectDetector feature permits entering Config Portal as requested. Now with scanning of WiFi networks for selection in Configuration Portal.

Figure A2

4. Library installed successfully with **INSTALLED** status on the list of the library.


Library Manager

Type All Topic All blynk

by **Volodymyr Shymanskyy** Version **1.0.0** **INSTALLED**
Build a smartphone app for your project in minutes! It supports WiFi, BLE, Bluetooth, Ethernet, GSM, USB, Serial. Works with many boards like ESP8266, ESP32, Arduino UNO, Nano, Due, Mega, Zero, MKR100, Yun, Raspberry Pi, Particle, Energia, ARM mbed, Intel Edison/Galileo/Joule, BBC micro:bit, DFRobot, RedBearLab, Microduino, LinkIt ONE ...
More info
Select version Install

Blynk For Chinese
by **hznupeter qiujiongtao@163.com**
Build a smartphone app for your project in minutes! 利用Blynk平台，可以快速搭建物联网应用。
More info

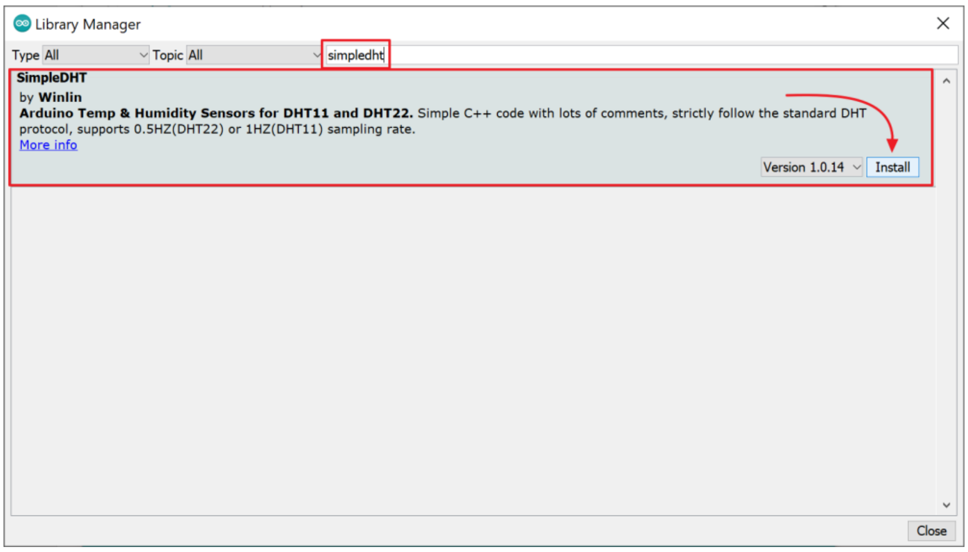
Blynk_Async_ESP32_BT_WF
by **Khoi Hoang**
Simple WiFiManager for Blynk and ESP32 with or without SSL, configuration data saved in either SPIFFS or EEPROM. Enable inclusion of both ESP32 Blynk BT/BLE and WiFi libraries. Then select one at reboot or run both. Eliminate hardcoding your Wifi and Blynk credentials and configuration data saved in either LittleFS, SPIFFS or EEPROM. Using AsyncWebServer instead of WebServer, with WiFi networks scanning for selection in Configuration Portal. By design, Blynk user can run ESP32 boards with either WiFi or BT/BLE by using different sketches, and have to upload / update firmware to change. This library enables user to include both Blynk BT / BLE and WiFi libraries in one sketch, run both WiFi and BT/BLE simultaneously, or select one to use at runtime after reboot. This library also supports (auto)connection to MultiWiFi and MultiBlynk, dynamic custom as well as static parameters in Config Portal. Eliminate hardcoding your Wifi and Blynk credentials and configuration data saved in either LittleFS, SPIFFS or EEPROM. Optional default Credentials to be autoloaded into Config Portal to use or change instead of manually input. Static STA IP and DHCP Hostname as well as Config Portal AP channel, IP, SSID, Password can be configured. DoubleDetectDetector feature permits entering Config Portal as requested. Now with scanning of WiFi networks for selection in Configuration Portal.
More info

Close

Figure A3

SimpleDHT

5. On Filter your search … field, replace **blynk** to **simpledht,** then look for the name of library **SimpleDHT *by Winlin*** and click **Install** button to install the library. Wait for the installation to be completed.


Library Manager

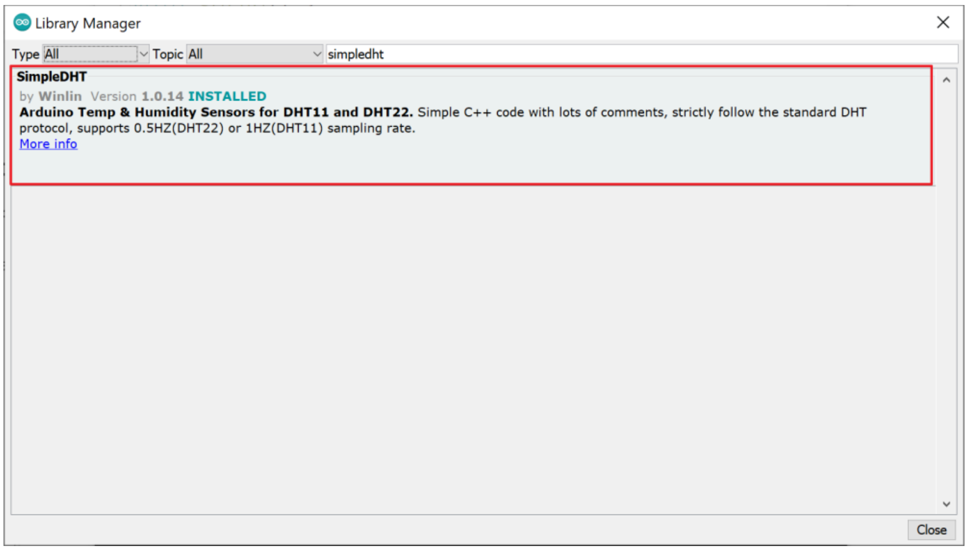
Type All Topic All simpledht

SimpleDHT
by **Winlin**
Arduino Temp & Humidity Sensors for DHT11 and DHT22. Simple C++ code with lots of comments, strictly follow the standard DHT protocol, supports 0.5HZ(DHT22) or 1HZ(DHT11) sampling rate.
More info
Version 1.0.14 Install

Close

Figure A4

6. Library installed successfully with **INSTALLED** status on the list of the library.



Library Manager

Type All | Topic All | simpledht

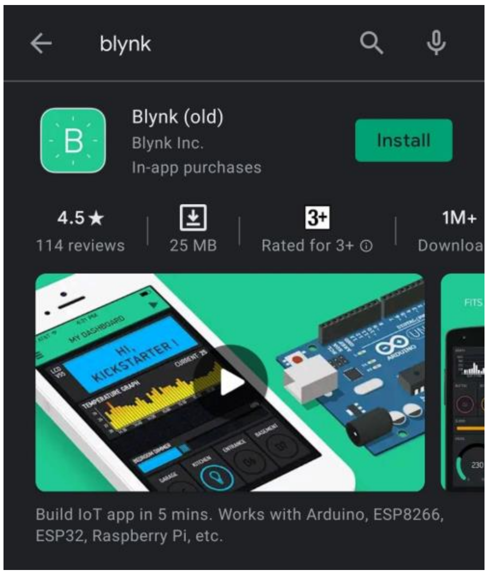
SimpleDHT
by **Winlin** Version **1.0.14** **INSTALLED**
Arduino Temp & Humidity Sensors for DHT11 and DHT22. Simple C++ code with lots of comments, strictly follow the standard DHT protocol, supports 0.5HZ(DHT22) or 1HZ(DHT11) sampling rate.
More info

Close

PART B: ARDUINO BLYNK INTEGRATION

Blynk App Installation and Setup

1. On your smartphone, go to Google Play or App Store. Search for **blynk** and install.
   Latest information: *Blynk has the newest version of the app, but this lab will follow previous version usage of the app.*



blynk

Blynk (old)
Blynk Inc.
In-app purchases
Install

4.5★ 114 reviews | 25 MB | Rated for 3+ | 1M+ Downloa

Build IoT app in 5 mins. Works with Arduino, ESP8266, ESP32, Raspberry Pi, etc.

Figure B1

2. Open the Blynk app on your smartphone.

3. Register a Blynk account using your email account, by tapping on **Create New Account**. Insert your email account the chosen password in the **Email** and **Password** field, for your Blynk account and tap the **Sign Up** button as below.

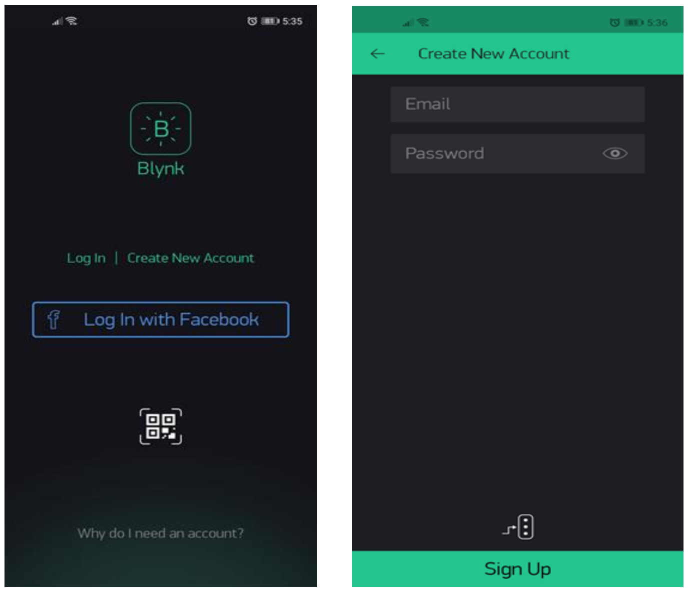


Figure B2

4. Once the registration is done. Tap on **+ New Project** to create a new project.
5. Type your project name on the Project **Name** field (example: **Practical Work 6**).
6. On the Choose Device section, tap on **ES8266** to select the right board, which is **NodeMCU**, then tap the OK button.

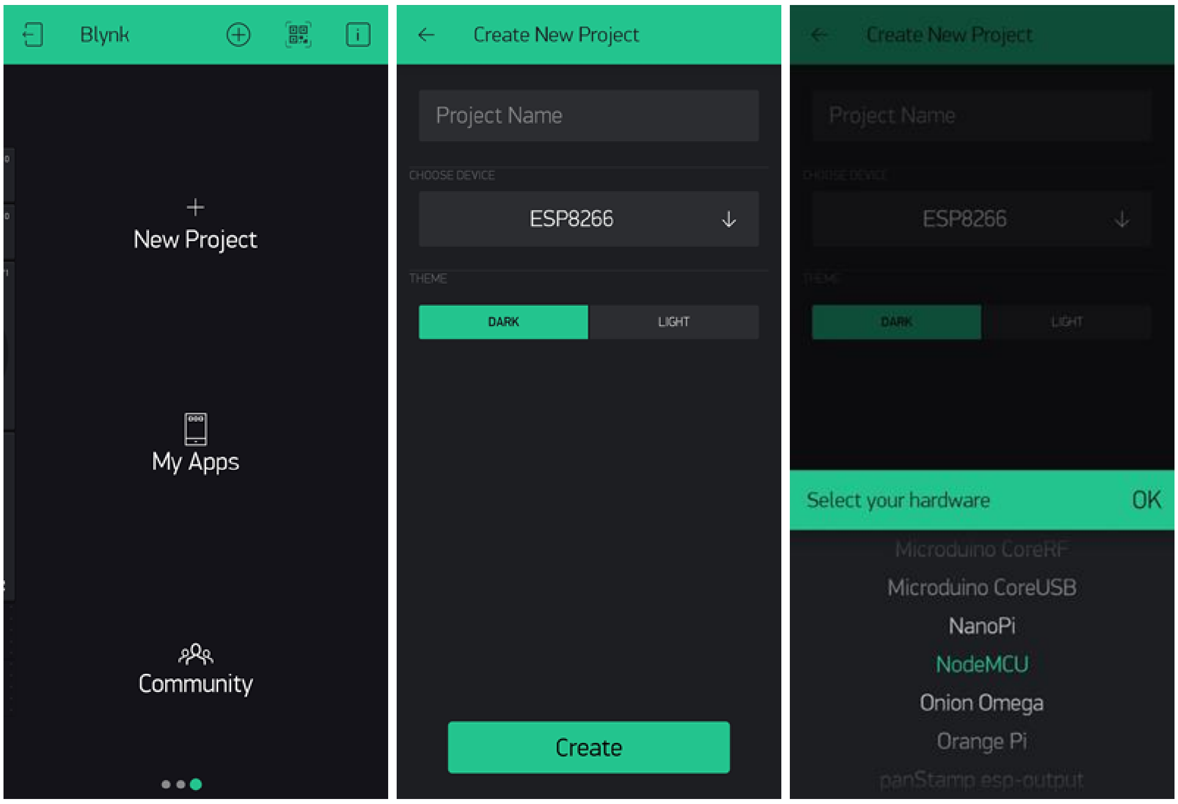
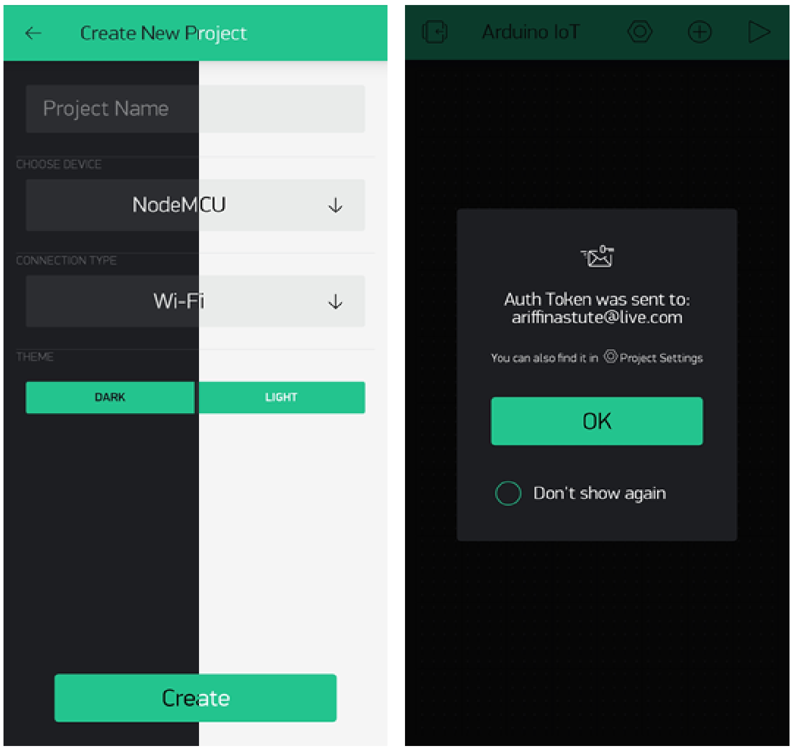


Figure B3

7. Optional to choose the theme colour of your Blynk project, either **Dark** or **Light**.
8. Tap the **Create** button and you will receive **Auth Token** on your email. Then, tap **OK**.



Auth Token was sent to: ariffinastute@live.com

You can also find it in Project Settings

Figure B4

9. The Blynk project canvas is now empty, add **Button** widget from **Widget Box**. Tap on **+** icon on the header, it will open the Widget Box.
10. Tap the Button from the Widget Box and the Button will appear on the project canvas.

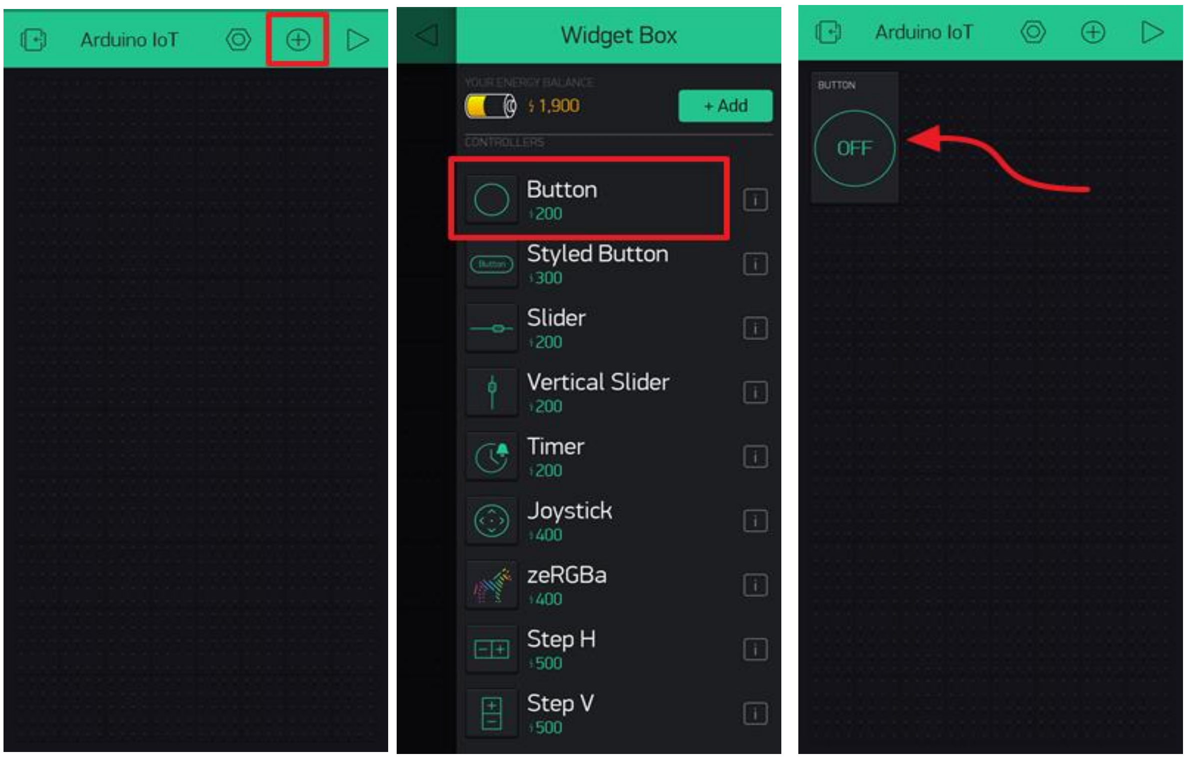


Figure B5

11. On the project canvas, tap on the Button widget.
12. Insert widget name, change the PIN to D1, mode of PUSH to SWITCH and tap ← icon.

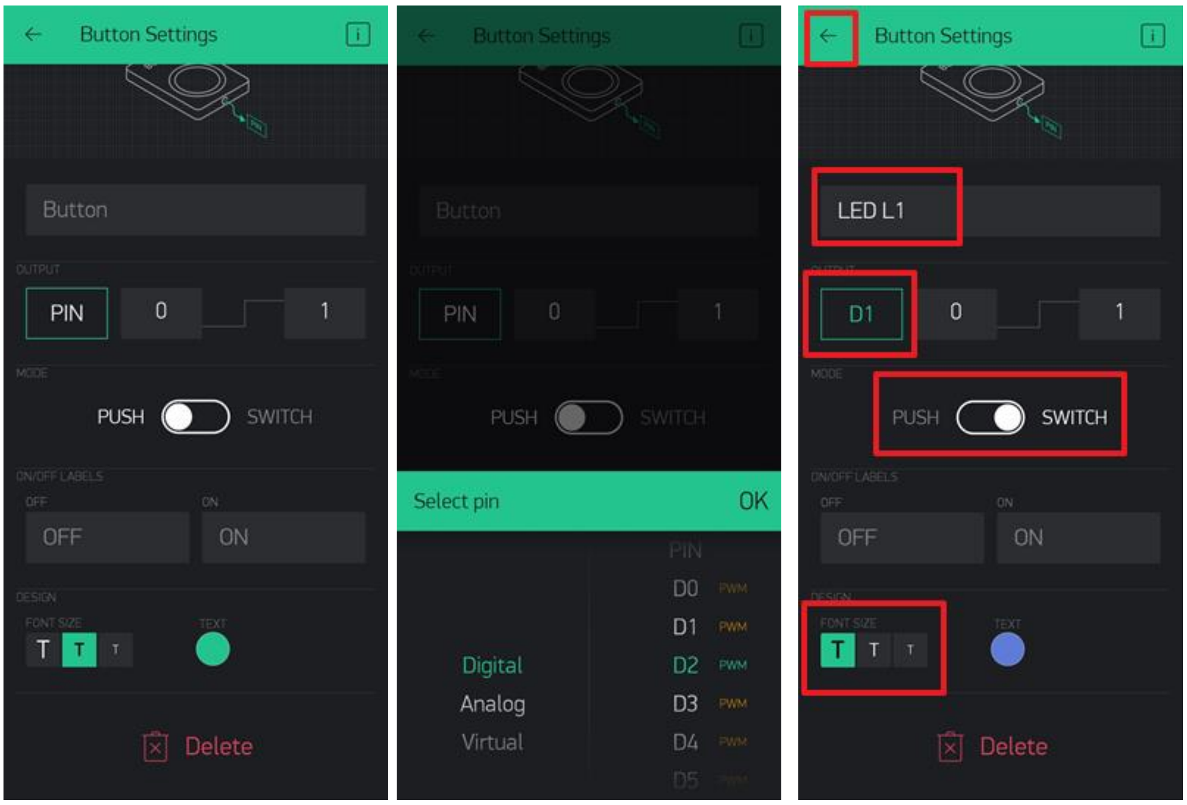


Figure B6

13. Once the setting is done, the Button widget on the project canvas will look like below and it is ready to control any LED interface to NodeMCU pin D1.

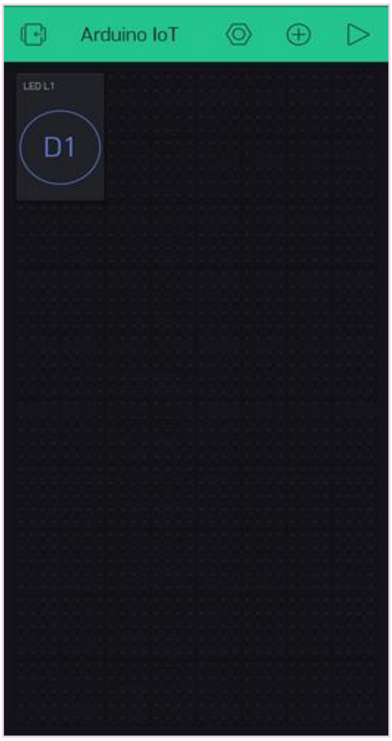
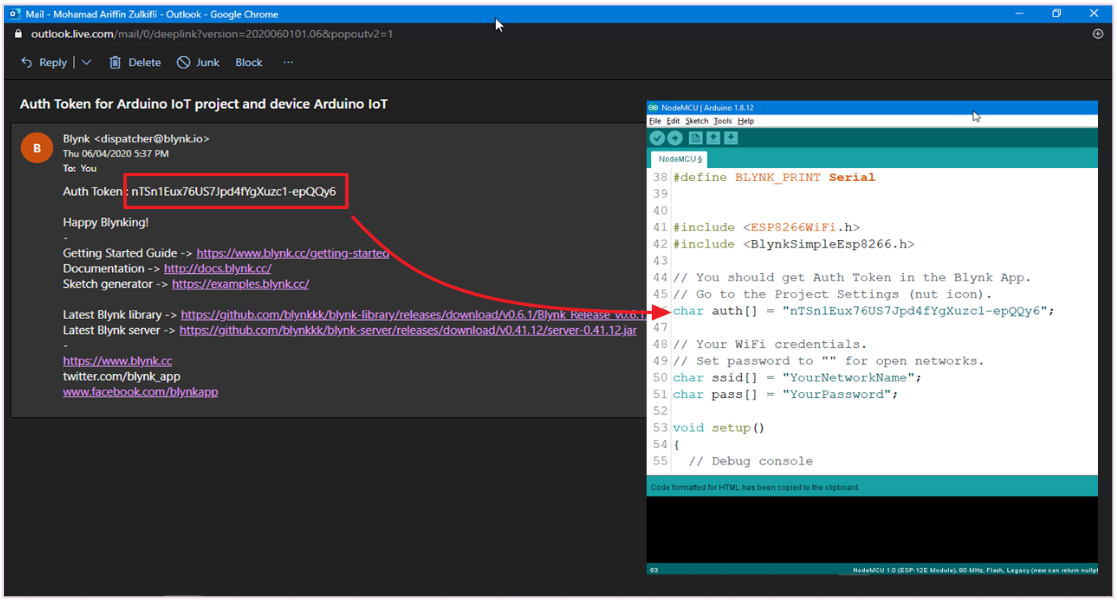


Figure B7

14. Open your email on your computer's browser, so we can copy the Auth Token to the Arduino sketch.
15. Open the email from Blynk, where the subject stated **Auth Token for …** as below and look for **Auth Token:** in the email and copy the Auth Token.


Mail - Mohamad Ariffin Zulkifli - Outlook - Google Chrome
outlook.live.com/mail/0/deeplink?version=2020060101.06&popoutv2=1
Reply
Delete
Junk
Block
NodeMCU | Arduino 1.8.12
File Edit Sketch Tools Help
NodeMCU §

Auth Token for Arduino IoT project and device Arduino IoT

Blynk <dispatcher@blynk.io>
Thu 06/04/2020 5:37 PM
To: You

Auth Token : nTSn1Eux76US7Jpd4fYgXuzc1-epQQy6

Happy Blynking!
-
Getting Started Guide -> https://www.blynk.cc/getting-started
Documentation -> http://docs.blynk.cc/
Sketch generator -> https://examples.blynk.cc/

Latest Blynk library -> https://github.com/blynkkk/blynk-library/releases/download/v0.6.1/Blynk_Release_v0.6.1
Latest Blynk server -> https://github.com/blynkkk/blynk-server/releases/download/v0.41.12/server-0.41.12.jar
-
https://www.blynk.cc
twitter.com/blynk_app
www.facebook.com/blynkapp

```
#define BLYNK_PRINT Serial


#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "nTSn1Eux76US7Jpd4fYgXuzc1-epQQy6";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

void setup()
{
  // Debug console
```

Code formatted for HTML has been copied to the clipboard.

NodeMCU 1.0 (ESP-12E Module), 80 MHz, Flash, Legacy (new can return nullpt

Figure B8

16. On the Arduino IDE, Open Blynk - NodeMCU example code. From menu select, **File > Examples > Blynk > Boards WiFi > NodeMCU**.

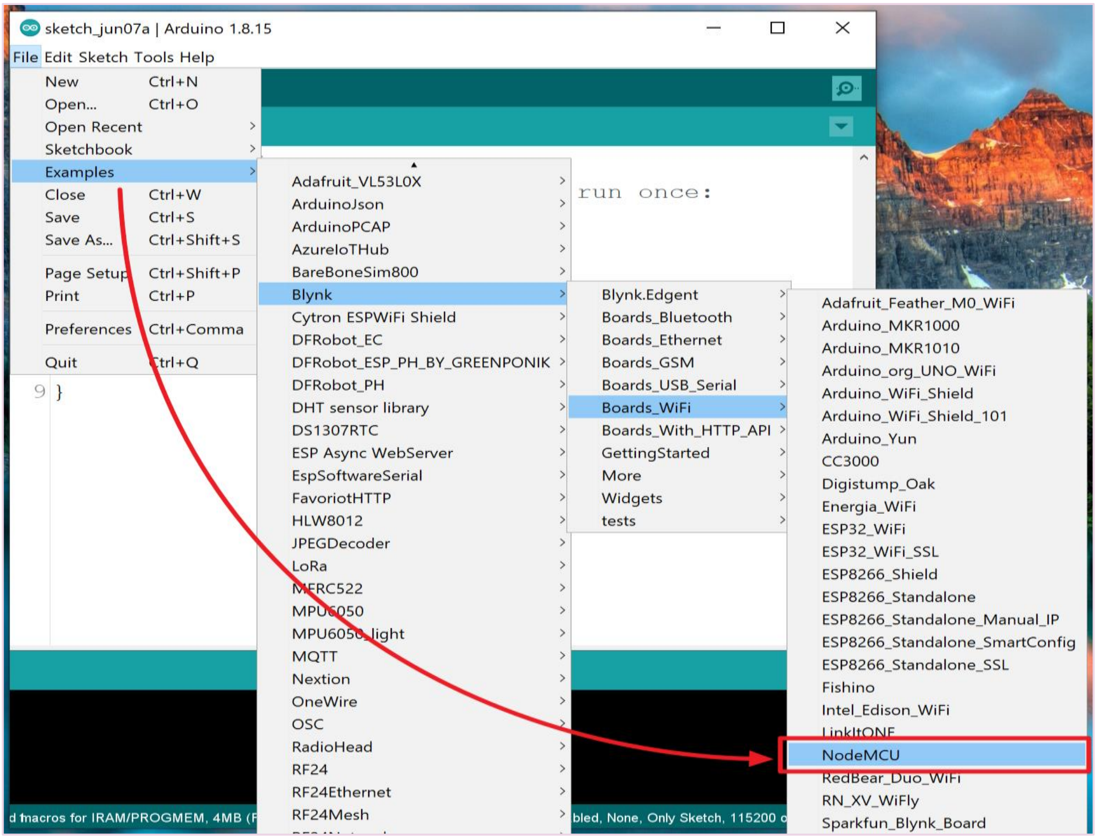
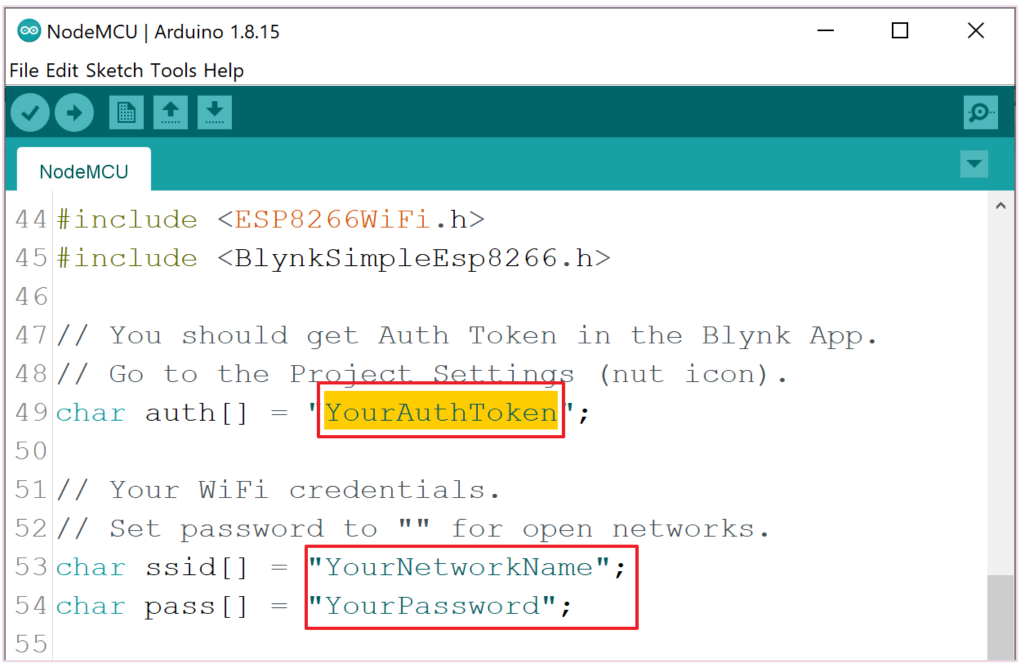


Figure B9

17. On the sketch, there are 3 variable you need to replace, to the right information:
    a. YourAuthToken
    b. YourNetworkName
    c. YourPassword



```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "YourAuthToken";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

```

Figure B10

18. Replace YourAuthToken by pasting the Auth Token which you have copied from the email.
19. Replace the network name and its password.

Hardware Installation

20. Construct the following circuit using the given kit as **Figure B11**.

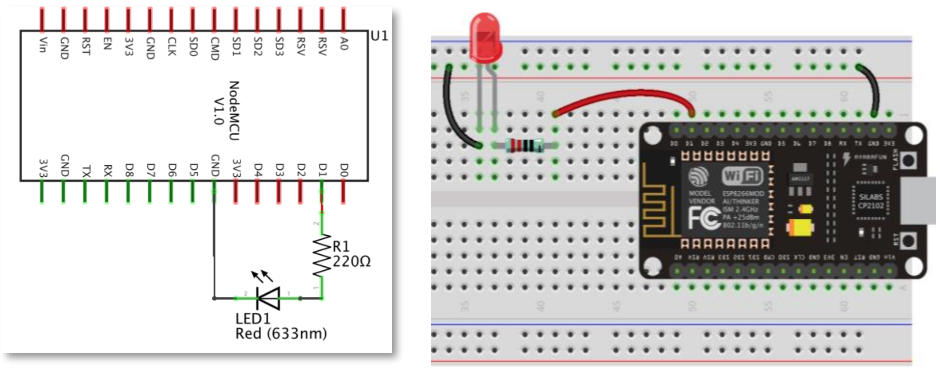
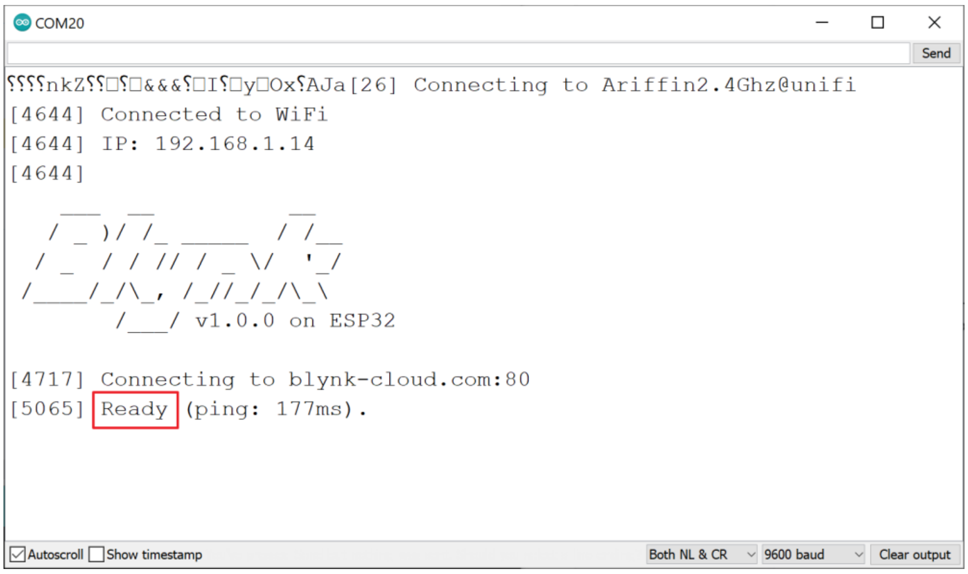


Figure B11

21. The red color LED and 220Ω resistor (Red, Red, Black, Black, Brown) is connected to the Pin D1 of NodeMCU.
22. Now, upload the sketch from Step 12.

23. Open the Serial Monitor and observe the action until you see the **Ready** status on the Serial Monitor, showing that the NodeMCU is now connected to the Blynk cloud and ready on the Blynk App.



```
ʢʢʢnkZʢʢ□ʢ□&&&ʢ□Iʢ□y□Oxʢ AJa[26] Connecting to Ariffin2.4Ghz@unifi
[4644] Connected to WiFi
[4644] IP: 192.168.1.14
[4644]
    ___  __          __
   / _ )/ /_ _____  / /__
  / _  / / // / _ \/  '_/
 /____/_/\_, /_//_/_/\_\
        /___/ v1.0.0 on ESP32

[4717] Connecting to blynk-cloud.com:80
[5065] Ready (ping: 177ms).
```

Figure B12

24. Now, on the Blynk App, tap the ▶ button on the header. If the connection is successfully established, now you can control the LED using the widget Button.

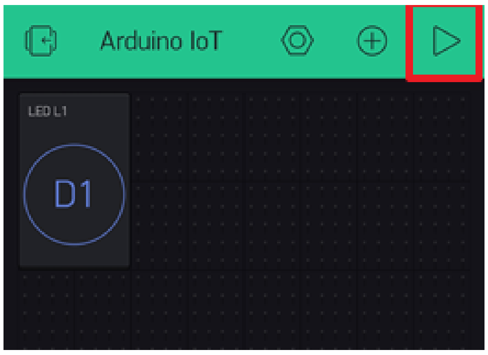


Figure B13

25. Observe on the header, now the ▶ button has changed to the ■ button and there is device icon on its left. ***If the device icon has a red dot as below, it means the device is still not connected to the cloud, please check your NodeMCU to WiFi connectivity.***

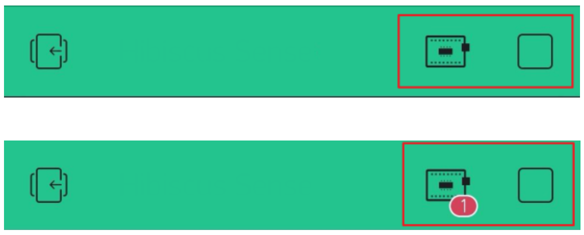


Figure B14

PART C: INCLUDE DHT SENSOR ON BLYNK

1. Construct the following circuit using the given kit as **Figure C1**.

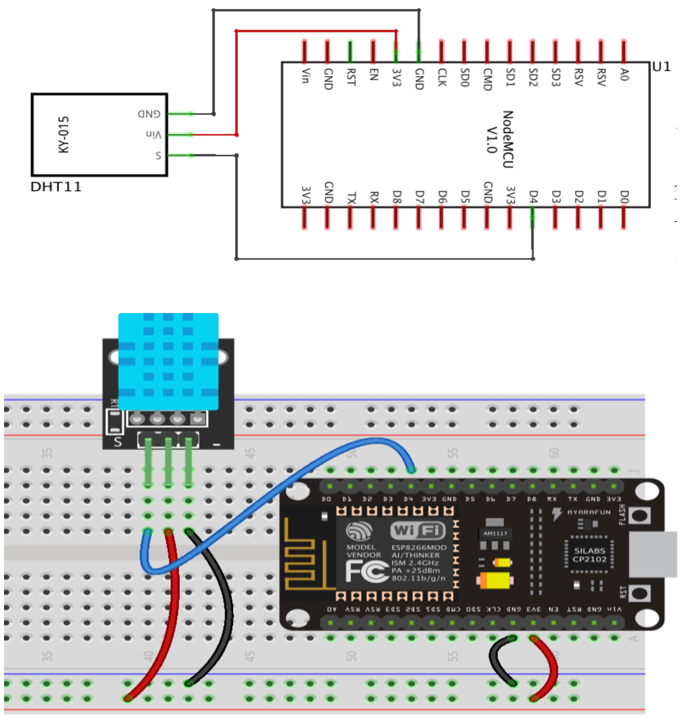
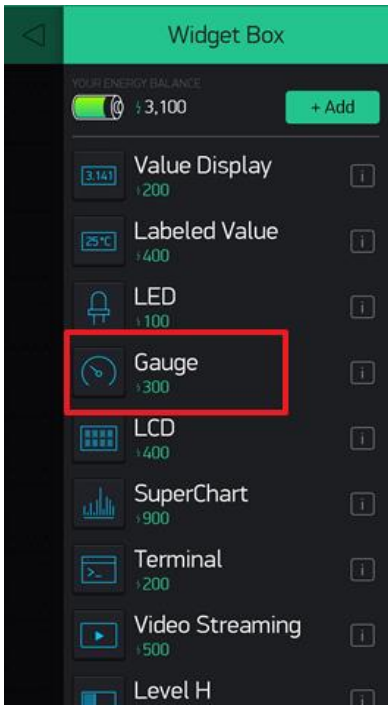


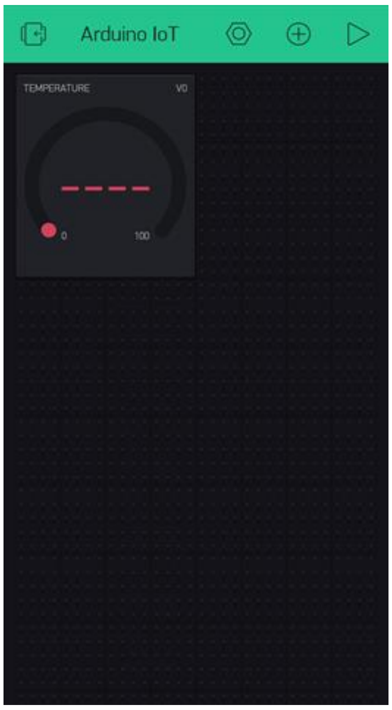
Figure C1

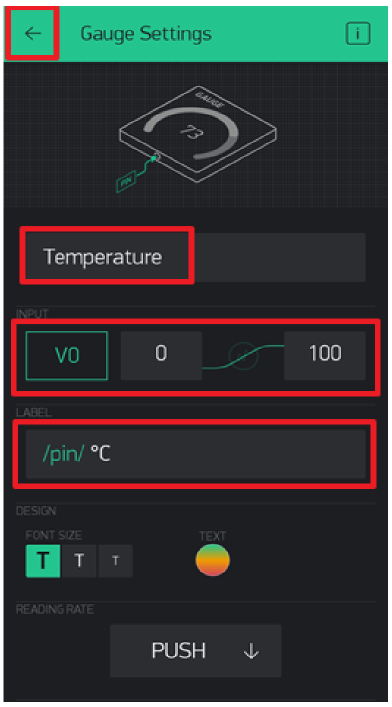
2. The DHT11 **S** label pin is connected to **D4** on NodeMCU. **+** label of DHT11 pin connected to **3V3** on NodeMCU and **-** label of DHT11 pin connected to **GND** on NodeMCU.
3. Write a sketch as in Appendix 1 below.
4. Replace the variable of auth[], ssid[] and pass[] below to the correct information.

```
char auth[] = "YourAuthToken";
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";
```

5. Upload the sketch.
6. Open the Serial Monitor and make sure the connection status is ready.
7. On the Blynk App, tap the ■ button (if there is not ▶ button)
8. Add 2 **Gauge** widget from the Widget Box.
9. Tap the 1st Gauge widget, insert widget name as **Temperature**, change the PIN to V0, maximum value replace from 1023 to 100, the label has the degree celcius (°C) symbol as below and tap ← icon.






Gauge Settings

Temperature

INPUT

V0 | 0 | 100

LABEL

/pin/ °C

DESIGN

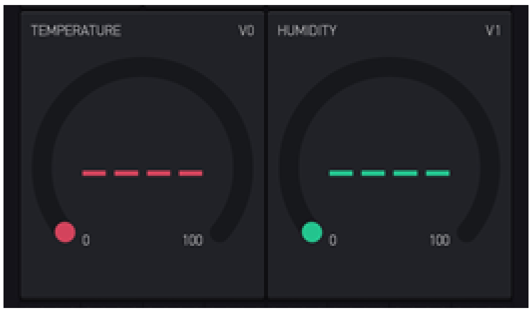
FONT SIZE T T T

TEXT

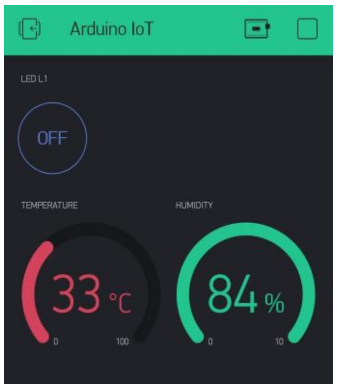
READING RATE

PUSH

10. Tap the 2nd Gauge widget and the setting is the same as above, but change the PIN to V1, and the label has a percentage (%) symbol.
11. Now, on the project canvas, you should have both Gauges as below.



12. Tap on the ▶ button on the header. If the connection is successfully established, now you can see the value of temperature and humidity are displayed real-time on the Gauges widget, as well as you control the LED using the widget Button.



DISCUSSION

1. Write your discussion about Home Automation using BLYNK App.
2. Find **FOUR (4)** examples from internet what Blynk app can do other than exercises an above.

APPENDIX 1

```
#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SimpleDHT.h>

char auth[] = "YourAuthToken";
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

SimpleDHT11 dht11(D4);

void setup() {
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
}

void loop() {
  Blynk.run();

  byte temperature = 0;
  byte humidity = 0;

  int err = SimpleDHTErrSuccess;
  if ((err = dht11.read(&temperature, &humidity, NULL)) !=
SimpleDHTErrSuccess) {
    return;
  }

  Blynk.virtualWrite(V0, temperature);
  Blynk.virtualWrite(V1, humidity);

}
```

e ISBN 978-967-2897-46-0

9 789672 897460

Step-by-Step Internet of Things Tutorial Using ESP8266

POLITEKNIK MALAYSIA
Port Dickson