SCHEME :K

Name :_____
Roll No.:_____ Year : 20 ___ 20 __
Exam Seat No. :_____

LABORATORY MANUAL FOR

IOT APPLICATIONS (315341)

ELECTRONICS ENGINEERING GROUP

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI
(Autonomous)(ISO21001:2018)(ISO/IEC27001:2013)

VISION

To ensure that the Diploma Level Technical Education constantly matches the latest requirements of technology and industry and includes the all-round personal development of students including social concerns and to become globally competitive, technology led organization.

MISSION

To provide high quality technical and managerial manpower, information and consultancy services to the industry and community to enable the industry and community to face the changing technological and environmental challenges.

QUALITY POLICY

We, at MSBTE, are committed to offer the best-in-class academic services to the students and institutes to enhance the delight of industry and society. This will be achieved through continual improvement in management practices adopted in the process of curriculum design, development, implementation evaluation and monitoring system along with adequate faculty development Programs.

CORE VALUES

MSBTE believes in the followings:

- Skill development in line with industry requirements.
- Industry readiness and improved employability of Diploma holders.
- Synergistic relationship with industry.
- Collective and cooperative development of all stake holders.
- Technological interventions in societal development.
- Access to uniform quality technical education.

A Laboratory manual
for

IOT Applications

(315341)

Semester – V

(DE/EJ/ET/EX/IE/TE)

Maharashtra State
Board of Technical Education, Mumbai

(Autonomous) (ISO 21001:2018) (ISO/IEC 27001:2013)

Maharashtra State
Board of Technical Education, Mumbai
(Autonomous) (ISO 21001:2018) (ISO/IEC 27001:2013)

4th Floor, Government Polytechnic Building, 49, Kherwadi,
Bandra (East), Mumbai – 400051.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This is to certify that Mr./Ms. ……. Roll No. ……. Of fifth Semester of Diploma in …… of Institute ……... (Code: ……) has attained pre-defined practical outcomes (PROs) satisfactorily in course **IOT Applications (315341)** for the academic year 20…… to 20…… as prescribed In the curriculum.

Place: …… Enrollment No.: ……

Date: …… Exam Seat No.: ……

Course Teacher **Head of Department** **Principal**

Seal Of Institute

Preface

The primary focus of any engineering laboratory/field work in the technical education system is to develop the much-needed industry relevant competencies and skills. With this in view, MSBTE embarked on this innovative 'K' Scheme curricula for engineering diploma programs with outcome- based education as the focus and accordingly, a relatively large amount of time is allotted for the practical work. This displays the great importance of laboratory work, making each teacher, instructor and student realize that every minute of the laboratory time needs to be effectively utilized to develop these outcomes, rather than doing other mundane activities. Therefore, for the successful implementation of this outcome-based curriculum, every practical has been designed to serve as a ***'vehicle'*** to develop this industry identified competency in every student. The practical skills are difficult to develop through 'chalk and duster' activity in the classroom situation. Accordingly, the 'K' scheme laboratory manual development team designed the practicals to ***focus*** on the ***outcomes***, rather than the traditional age-old practice of conducting practical's to 'verify the theory' (which may become a byproduct along the way).

This laboratory manual is designed to help all stakeholders, especially the students, teachers and instructors to develop in the student the predetermined outcomes. It is expected from each student that at least a day in advance, they have to thoroughly read through the concerned practical procedure that they will do the next day and understand the minimum theoretical background associated with the practical. Every practical in this manual begins by identifying the competency, industry relevant skills, course outcomes and practical outcomes which serve as a key focal point for doing the practical. The students will then become aware about the skills they will achieve through the procedure shown there and necessary precautions to be taken, which will help them to apply in solving real-world problems in their professional life.

This manual also provides guidelines to teachers and instructors to effectively facilitate student- centered lab activities through each practical exercise by arranging and managing necessary resources in order that the students follow the procedures and precautions systematically ensuring the achievement of outcomes in the students.

The Internet of Things (IoT) plays a vital role in modern technology by connecting devices, sensors, and systems through the internet. It helps in automating processes, saving time, and improving efficiency in industries like healthcare, agriculture, and manufacturing. IoT enables real-time data collection and monitoring, leading to smarter decision-making. For polytechnic students, learning IoT opens opportunities in innovative projects and future-ready careers in the tech world.

Although best possible care has been taken to check for errors (if any) in this laboratory manual, perfection may elude us as this is the first edition of this manual. Any errors and suggestions for improvement are solicited and highly welcome.

Program Outcomes (POs)

Following program outcomes are expected to be achieved through the practical of the course.

PO1: Basic and Discipline specific knowledge: Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the broad-based Electronics Engineering group program problems.

PO2: Problem analysis: Identify and analyze well-defined Electronics Engineering group program problems using codified standard methods.

PO3: Design/ development of solutions: Design solutions for well-defined technical problems and assist with the design of Electronics Engineering group program systems components or processes to meet specified needs.

PO4: Engineering Tools, Experimentation and Testing: Apply modern Electronic Engineering group program tools and appropriate technique to conduct standard tests and measurements.

PO5: Engineering practices for society, sustainability and environment: Apply appropriate Electronics Engineering group program technology in context of society, sustainability, environment and ethical practices.

PO6: Project Management: Use Electronics Engineering group program management principles individually, as a team member or a leader to manage projects and effectively communicate about well- defined engineering activities.

PO7: Life-long learning: Ability to analyze individual needs and engage in updating in the context of Electronics Engineering group program technological changes.

List of Industry Relevant Skills

The following industry relevant skills of the competency “Maintain system based on Internet of Things (IoT)." are expected to be developed in the student by undertaking the practical of this laboratory manual

1. Learn to use boards like Arduino and ESP32, and write basic programs in C/C++.
2. Know how to connect and use sensors like temperature, motion, and light sensors.
3. Send data to apps like Blynk or websites like ThingSpeak to monitor devices online.
4. Make simple electronic circuits and fix problems using basic tools
5. Make small IoT projects and write simple reports with diagrams and explanations.

Practical - Course Outcome matrix

Course Level Learning Outcomes (COs)

CO1 – Interpret the architecture of Internet of Things (IoT).

CO2 – Select IoT system for given application development.

CO3 – Integrate sensors and actuators in IoT based systems.

CO4 – Manage IoT communication for data handling.

CO5- Develop IoT based applications.

Sr. No.	Title of the Practical	CO 1	CO 2	CO 3	CO 4	CO 5
1.	Installation and configuration of Arduino IDE for NodeMCU		√			
2.	Interfacing LED and Switch with NodeMCU		√			
3.	Interfacing relay and IR sensor with NodeMCU			√		
4.	Interfacing Humidity sensor with NodeMCU			√		
5.	Interfacing PIR Sensor with NodeMCU			√		
6.	Connecting NodeMCU to a Wi-Fi network.				√	
7.	Data Transmission from NodeMCU to Webserver				√	
8.	Implementation of MQTT protocol with NodeMCU				√	
9.	Monitoring and Controlling Light intensity using NodeMCU				√	
10.	Implementation of IOT enabled smart home application					√

Guidelines to Teachers

1. Teacher should provide the guideline with demonstration of practical to the students with all features.
2. Teacher shall explain prior concepts to the students before starting of each practical.
3. Involve students in the performance of each experiment.
4. Teacher should ensure that the respective skills and competencies are developed in the students after the completion of the practical exercise.
5. Teachers should give opportunities to students for hands-on experience after the demonstration.
6. Teacher is expected to share the skills and competencies to be developed in the students.
7. Teacher may provide additional knowledge and skills to the students even though not covered in the manual but are expected of the students by the industry.
8. Finally give practical assignments and assess the performance of students based on tasks assigned to check whether it is as per the instructions.
9. Teacher is expected to refer complete curriculum document and follow guidelines for implementation
10. At the beginning of the practical which is based on the simulation, teacher should make the students acquainted with any simulation software environment.

Instructions for Students

1. Listen carefully to the lecture given by the teacher about course, curriculum, learning structure, skills to be developed.
2. Organize the work in the group and make a record of all observations.
3. Do the calculations and plot the graph wherever it is required in the practical
4. Students shall develop maintenance skills as expected by industries.
5. Student shall attempt to develop related hand-on skills and gain confidence.
6. Student shall develop the habits of evolving more ideas, innovations, skills etc. those included in scope of manual
7. Student should develop the habit to submit the practical on date and time.
8. Student should prepare well while submitting a write-up of exercise.

Content Page

List of Practical's and Progressive Assessment Sheet

Sr. No.	Title of the practical	Page No.	Date of performance	Date of submission	Assessment marks (25)	Dated sign. Of teacher	Remarks (if any)
1.	Installation and configuration of Arduino IDE for NodeMCU	1					
2.	Interfacing LED and Switch with NodeMCU	10					
3.	Interfacing relay and IR sensor with NodeMCU	16					
4.	Interfacing Humidity sensor with NodeMCU	23					
5.	Interfacing PIR Sensor with NodeMCU	29					
6.	Connecting NodeMCU to a Wi-Fi network.	35					
7.	Data Transmission from NodeMCU to Webserver	43					
8.	Implementation of MQTT protocol with NodeMCU	56					
9.	Monitoring and Controlling Light intensity using NodeMCU	64					
10.	Implementation of IOT enabled smart home application	72					
Total							

Note: Out of above suggestive LLOs -

- '*' Marked Practical's (LLOs) Are mandatory.
- Minimum 80% of above list of lab experiment are to be performed.
- Judicial mix of LLOs are to be performed to achieve desired outcomes.

Practical No. 1: Installation and configuration of Arduino IDE for NodeMCU

I Practical Significance:

Installing and configuring the Arduino IDE for NodeMCU is a crucial first step in learning to develop and implement Internet of Things (IoT) applications. This practical exercise provides students with the fundamental skills needed to program, interface with, and control the NodeMCU (ESP8266) microcontroller using the Arduino development platform.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'.

III Course Level Learning outcome(s)

- Select IoT system for given application development.

IV Laboratory Learning outcome(s)

- LLO 1.1 Establish a connection between the NodeMCU-ESP8266 and a computer using appropriate cables and drivers.
- LLO 1.2 Install and configure Arduino IDE for NodeMCU programming

V Relevant Affective Domain related outcome(s)

- The measurement process can not only provide technical knowledge but also foster essential skills and attitudes for success in future engineering endeavors.

VI Relevant Theoretical Background.

Arduino in IoT: In IoT applications the Arduino is used to collect the data from the sensors/devices to send it to the internet and receives data for purpose of control of actuators

Arduino IDE: The Arduino Software (IDE) is easy-to-use and is based on the Processing programming environment. The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. The open-source Arduino Software (IDE) makes it easy to write code and upload it to the NodeMCU.

VII Practical setup in Laboratory.

a. Sample:

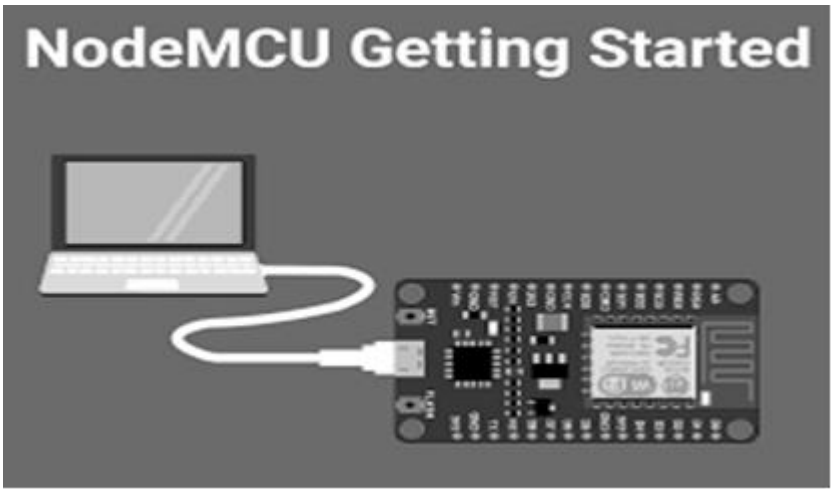


Figure 1.1: NodeMCU Programming through serial cable

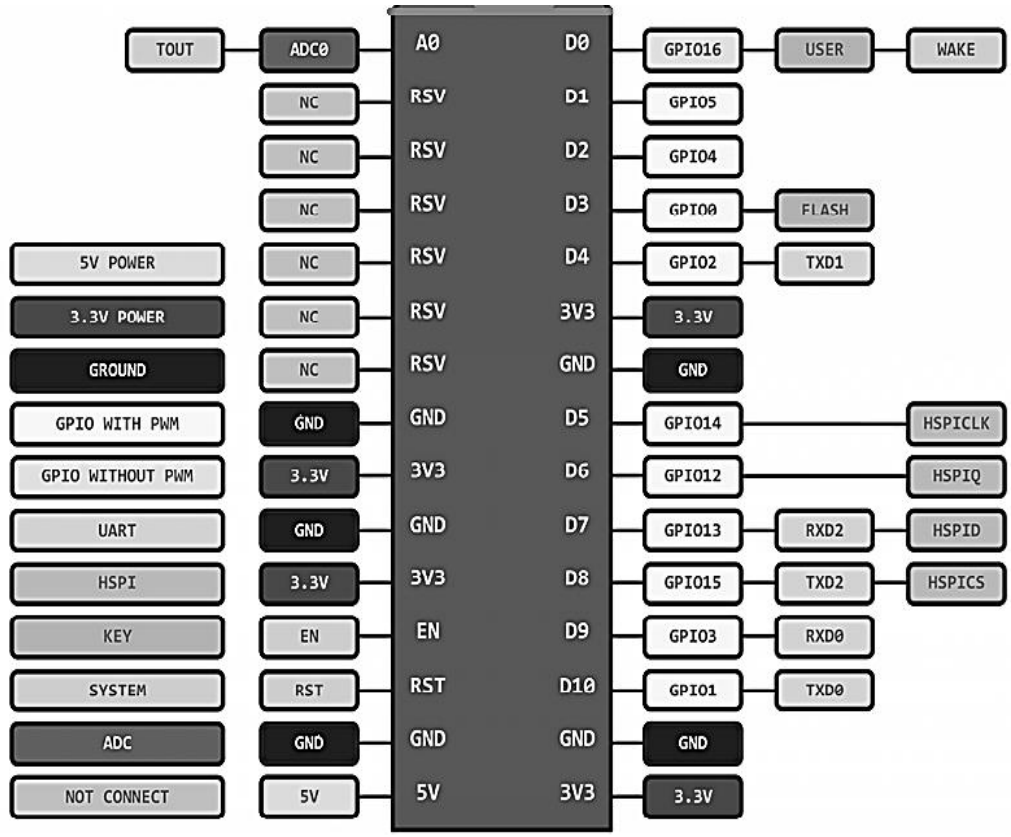


Figure 1.2: Pin diagram of NodeMCU (ESP 8266)

b. Actual setup used in Laboratory:

VIII Required Resources/ apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x), TypeScript/Electron (v2.x) Operating Systems Supported: Windows 10 or later (64-bit), macOS 10.15 (Catalina) or later, Linux (64-bit Debian/Ubuntu-based distributions)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) PWM Pins: All digital pins support PWM (10-bit resolution) Analog Input: 1 (10-bit ADC, 0–1V input range) Flash Memory: Typically, 4MB (varies by board) SRAM: ~50 KB available for application usage EEPROM: Emulated in flash	1

IX Precaution to be followed (Safety instructions / Rules / Standards):

1 Follows all step of while installing Arduino IDE software

X Procedure:

1. Connect your NodeMCU to the Computer/ Laptop.
2. Install the COM/Serial port driver: Download diver CH340G Driver using following link-
3. nodemcu-devkit/Drivers at master · nodemcu/nodemcu-devkit · GitHub
4. Install the Arduino IDE 1.8.3 or higher version

5. Install the ESP8266 Board Package

6. Enter http://arduino.esp8266.com/stable/package_esp8266com_index.json into Additional Board Manager URLs field in the Arduino v1.6.4+ preferences (Open Arduino IDE–>File–>Perferences–>Settings).

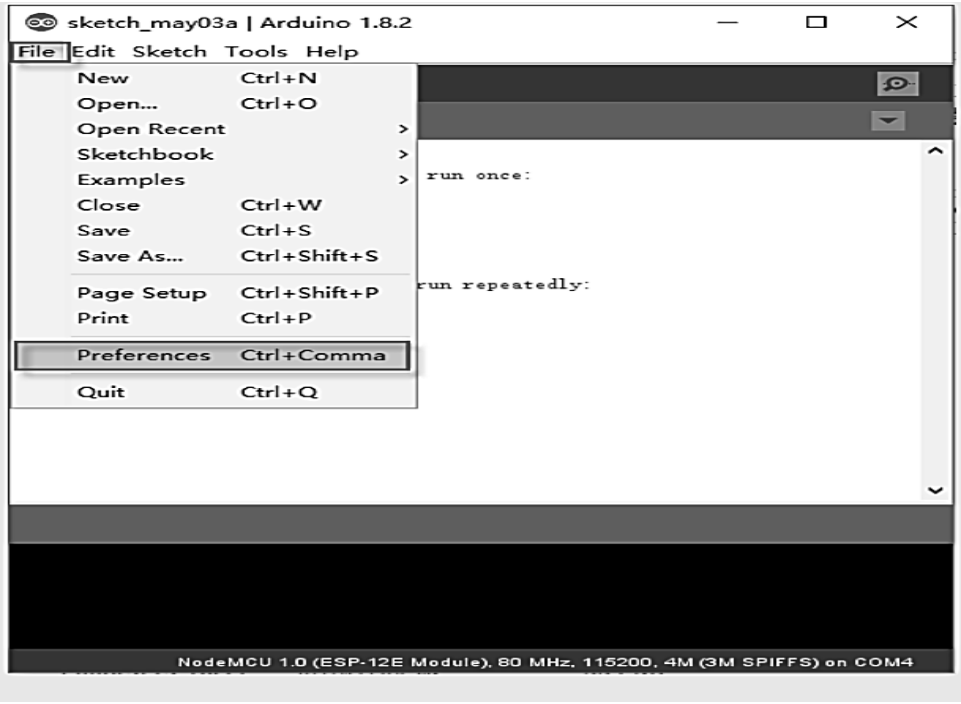


Figure 1.3: Installation of ESP8266 Board Package

7. Enter the link and click “OK” to save your changes.

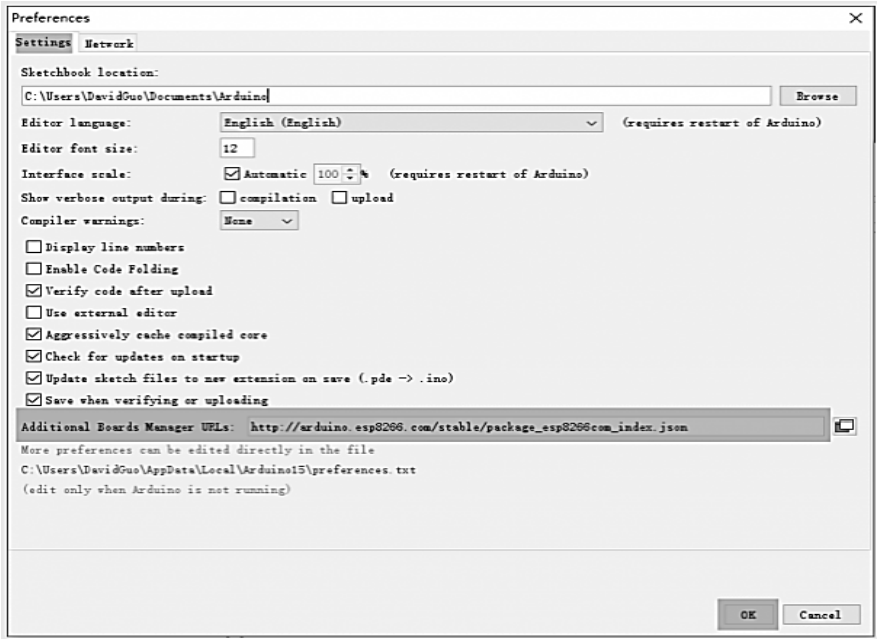


Figure 1.4: Setting of ESP8266 Board Package

8. Next, use the Board Manager to install the ESP8266 package

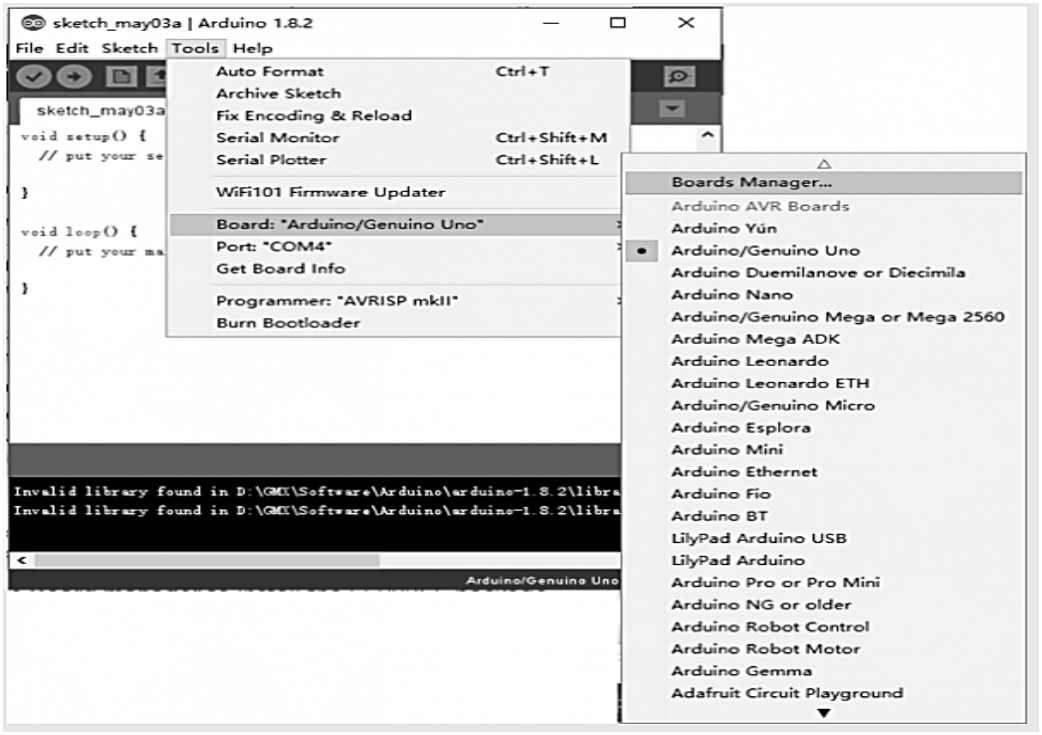


Figure 1.5: Tools of ESP8266 Board Package

9. Enter the Boards Manager and find the board type as below:

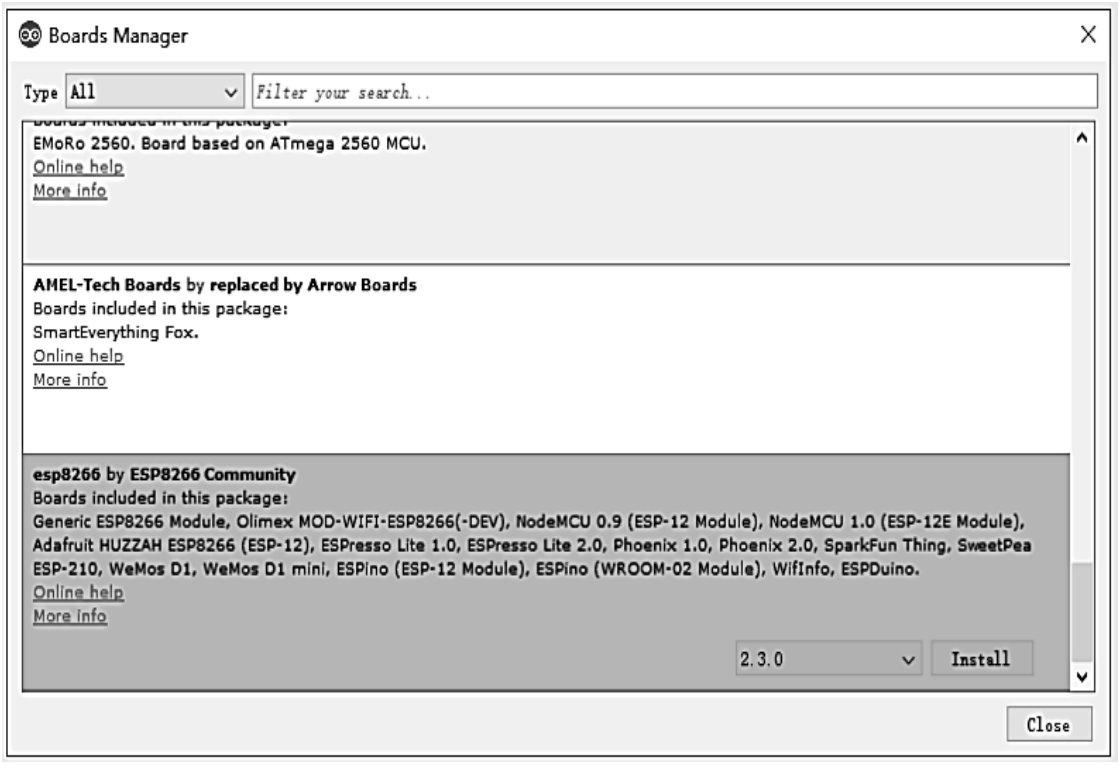


Figure 1.6: Various types of ESP8266 Board

10. Scroll the Boards Manager screen down to the bottom, you will see, A module called “esp8266 by esp8266 Community” (see following picture), select the latest version and click “Install”.

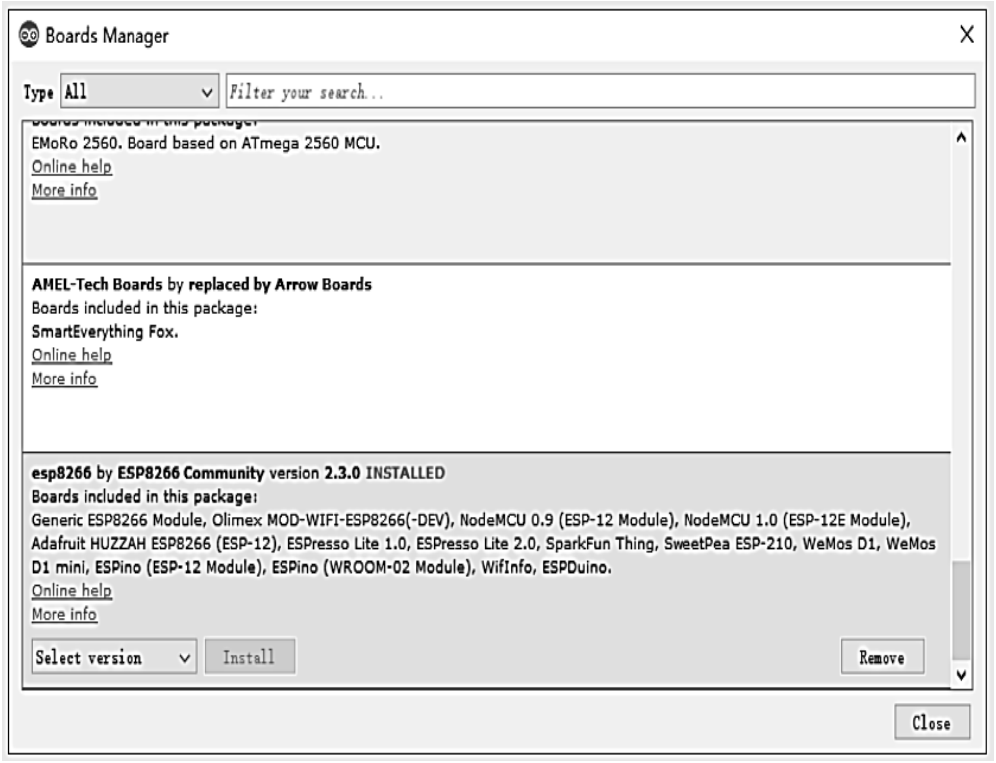


Figure 1.7: Selection of ESP8266 Board

11. Setup ESP8266 Support

When you’ve restarted, select NodeMCU 1.0 from the Tools->Board dropdown.

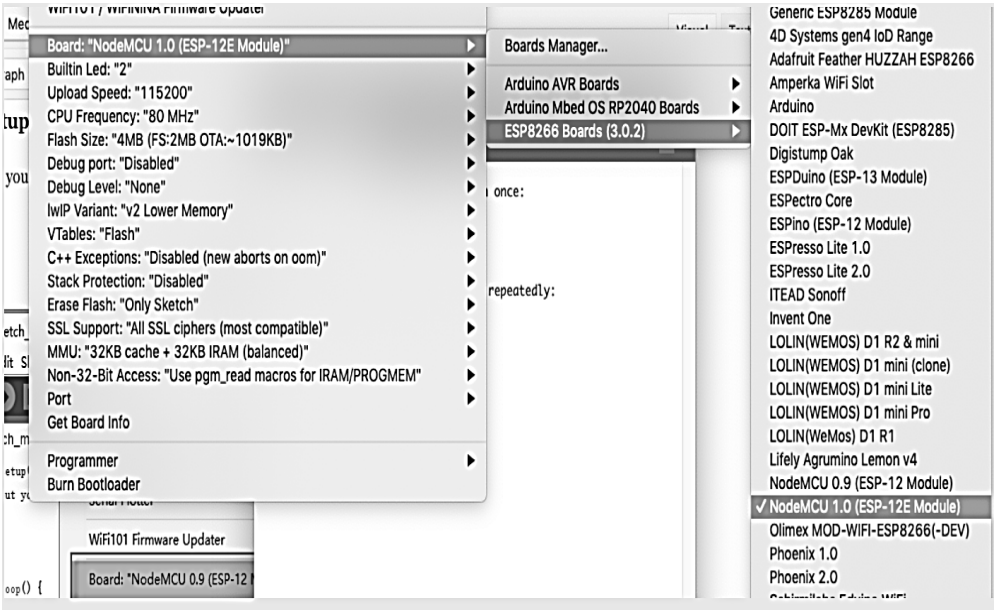


Figure 1.8: Setup of ESP8266 Board

12. You also need to select a Port which matches your NodeMCU port.

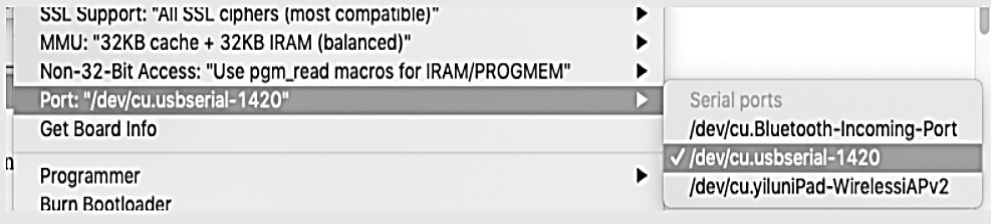


Figure 1.8: Select port of ESP8266 Board

XI Result(s):

……

……

……

……

XII Interpretation of Result(s):

……

……

……

……

……

XIII Conclusion and Recommendations:

……

……

……

……

XIV Practical Related Questions:

Note: Below given are few sample questions. Teachers must design such questions to ensure achievement of identified CO.

1 State applications of IoT.
2 What are the common libraries needed for programming NodeMCU?
3 What to do if NodeMCU is not detected by Arduino IDE (no COM port)?

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

XV References/Suggestions for further reading: include websites/links/ Virtual Lab Link.

1 https://www.youtube.com/watch?v=Ugf9pXhw-4E
2 https://www.youtube.com/watch?v=njAC1u8vs8Q

XVI Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Use of IDE tools for programming	20%
02	Making connections of hardware	20%
03	Follow ethical practices	20%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	30%
02	Timely Submission of report	10%
Total (25 Marks)		**100%**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 02: Interfacing LED and Switch with NodeMCU

I Practical Significance:

Interfacing LEDs and switches with NodeMCU has practical significance in various applications, enabling remote control and automation. This experiment will help students to control LEDs and other devices by using a simple switch, or remotely via a network connection using the NodeMCU's Wi-Fi capabilities.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'

III Course Level Learning outcome(s)

- Select IoT system for given application development.

IV Laboratory Learning outcome(s)

- LLO 2.1- Interface LED and switch with NodeMCU to turn ON and OFF LED.

V Relevant Affective Domain Related outcome(s)

1 Handle equipment carefully
2 Follow safety practices

VI Relevant Theoretical Background.

A light-emitting diode (LED) is a semiconductor device that emits light when an electric current flows through it. When current passes through an LED, the electrons recombine with holes emitting light in the process. LEDs allow the current to flow in the forward direction and blocks the current in the reverse direction.
When the diode is forward biased, the minority electrons are sent from p to n while the minority holes are sent from n to p. At the junction boundary, the concentration of minority carriers increases. The excess minority carriers at the junction recombine with the majority charges carriers

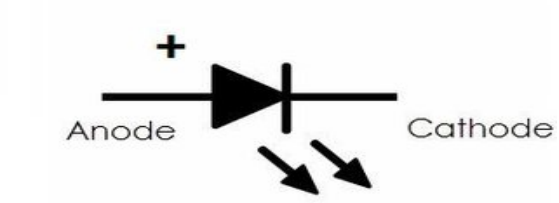


Figure 2.1: Symbol of LED

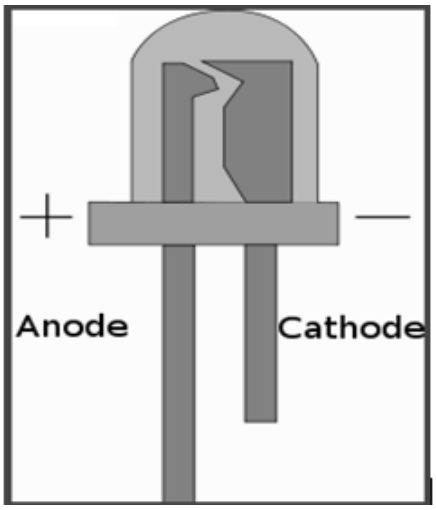


Figure 2.2: Structure of Light Emitting Diode (LED)

VII Practical setup in Laboratory.

a) Simple Circuit/Experimental Setup:

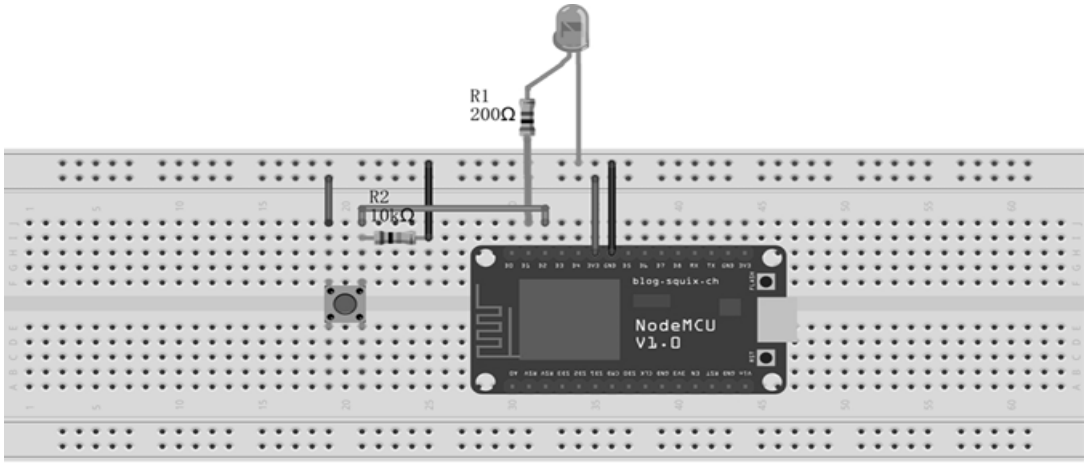


Figure 2.3: Circuit connection of NodeMCU to LED.

b) Actual Circuit/Experimental Setup used in laboratory with related equipment rating:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) Analog Input: 1 (10-bit ADC, 0–1V input range)	1
4	Push Button	Power Rating: MAX 50mA 24V DC	1
3	LED	Red/Green	1
4	Resistor	10 kΩ, 200Ω	1 each
5	Breadboard	5.5cm X 17 cm	1
6.	Micro-USB	-	1
7.	Connecting wires	-	As per Requirement

IX Precautions to be followed

1. Use always current limiting resistor before LED connected to Arduino.

X Procedure:

1. Interface LED to NodeMCU as per circuit diagram shown in fig. 2.3.
2. Develop program using Arduino IDE software or any other relevant software tool
3. Compile program on IDE
4. Upload the program
5. Observe the LED on/ off status

XI Sample Program:

```
int ledpin = 5;
int button = 4;
int buttonState=0;
void setup()
{
```

```
pinMode(ledpin, OUTPUT);
pinMode(button, INPUT);
}
void loop()
{
buttonState=digitalRead(button); // put your main code here, to run repeatedly:
if (buttonState == 1)
{
digitalWrite(ledpin, HIGH);
delay(200);
}
if (buttonState==0)
{
digitalWrite(ledpin, LOW);
delay(200);
}
}
```

XII Result(s)

……..

……..

……..

……..

XIII Conclusions:

……..

…….....

……..

XIV Practical Related Questions:

Note: Below given are few sample questions. Teachers must design such questions to ensure achievement of identified CO.

1. Write a program to turn the LED on when the switch is pressed?
2. Write a program to change the color of multicolor LED?
3. What is debounce, and why is it important in switch interfacing?

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

XV References/Suggestions for further reading: include websites/links/Virtual lab Link:

1. https://www.youtube.com/watch?v=omsLiyrrUbc
2. NodeMCU Lesson 4—Button Control LED « osoyoo.com
3. https://www.youtube.com/watch?v=Hs-HO5ko3Sc

XVI Assessment Scheme:

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Coding and Debugging ability	25%
02	Making connections of hardware	25%
03	Follow ethical practices.	10%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	20%
02	Timely Submission of report	205%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 03: Interfacing relay and IR sensor with NodeMCU

I Practical Significance

Interfacing a relay and an IR sensor with a NodeMCU has significant practical applications in the field of home automation, where the remote control and automation of electrical devices are increasingly important. This practical helps students understand how NodeMCU can safely control high-voltage systems through relays, ensuring electrical isolation and safety. The integration of an IR sensor enables proximity detection, allowing the system to respond dynamically to changes in the environment. Such capabilities are crucial in the development of smart systems for applications such as security, energy efficiency, and automated control.

II Industry / Employer Expected outcome(s):

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'.

III Course Level Learning outcome(s)

- Integrate sensors and actuators in IoT based system.

IV Laboratory Learning outcome(s)

- LLO 3.1 Control relay operation using NodeMCU and IR sensor.

V Relevant Affective Domain related outcome(s)

1. Handle equipment carefully
2. Follow safety practices.

VI Relevant Theoretical Background.

Relay: Relay modules are simply circuit boards that house one or more relays. They come in a variety of shapes and sizes.

Relay modules contain other components than the relay unit. These include indicator LEDs, protection diodes, transistors, resistors, and other parts. A relay is an electrical switch that can be used to control devices and systems that use higher voltages. The relay module input voltage is usually DC. However, the electrical load that a relay will control can be either AC or DC, but essentially within the limit levels that the relay is designed for.

Figure 3.1: Relay Module

IR Sensor: IR sensors work by detecting infrared radiation emitted by objects in the environment. They typically use a photodiode or a phototransistor to detect the IR radiation and convert it into an electrical signal that can be processed and analyzed. The IR transmitter continuously emits the IR light and the IR receiver keeps on checking for the reflected light. If the light gets reflected back by hitting any object in front it, the IR receiver receives this light. This way the object is detected in the case of the IR sensor.

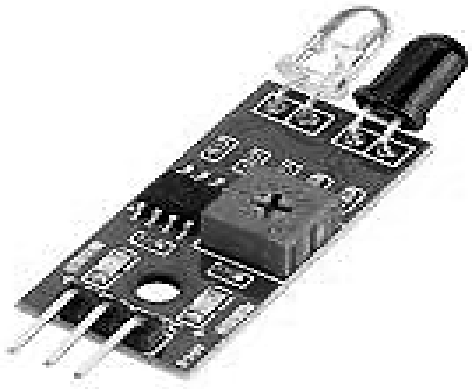


Figure 3.2: IR Sensor Module

VII Practical setup in Laboratory

a. Simple Circuit/Experimental setup:

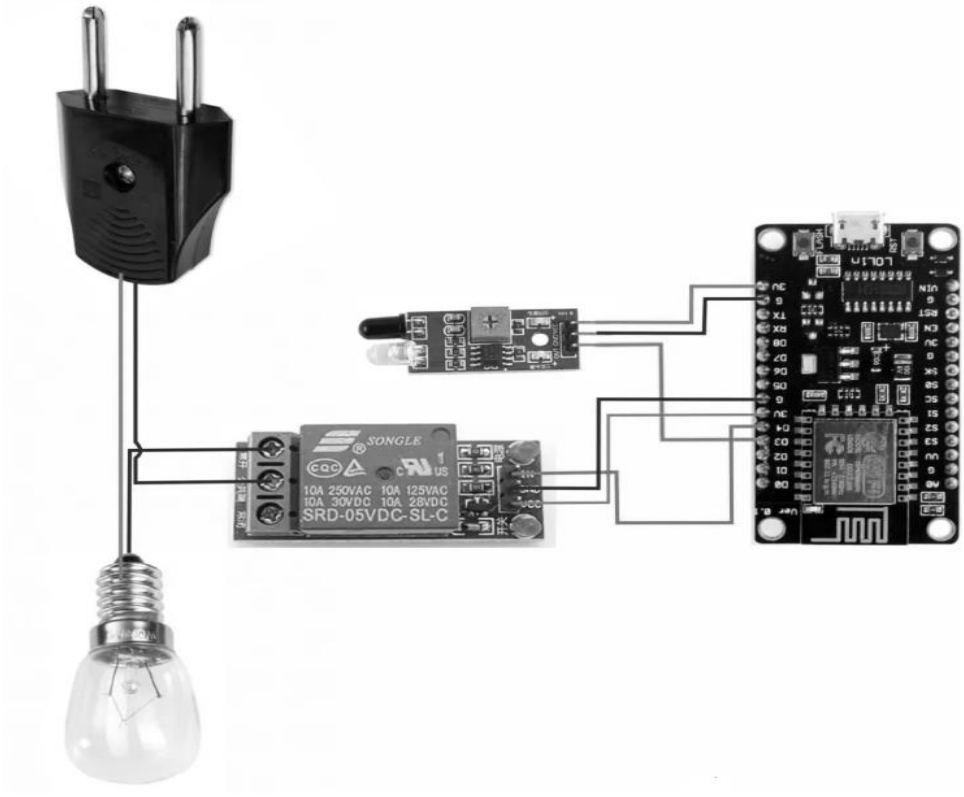


Figure 3.3: Circuit diagram of interfacing NodeMCU with IR sensor for controlling relay

b. Actual practical set up used in Laboratory:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1.	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2.	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x)	1
3.	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) Analog Input: 1 (10-bit ADC, 0–1V input range)	1
4.	IR sensor Module	Range: 2-30 cm	1
5.	Relay Module	5V, 1 Channel	1
6.	Micro-USB	-	1
7.	Connecting wires	-	As per Requirement

IX Precautions to be followed

1 Connect pins of IR sensor in accurate manner.

X Procedure

- **Part A: General setup:**

1 Interface LED to NodeMCU as per circuit diagram shown in fig. 3.3.

2 Develop program using Arduino IDE software or any other relevant software tool.

3 Compile program on IDE.

4 Upload the program.

5 Observe the relay ON/OFF status.

XI Sample Program:

```
int IR = D3;
int relay = D4;
void setup()
{
Serial.begin(9600);
pinMode(IR, INPUT);
pinMode(relay, OUTPUT);
digitalWrite(relay, HIGH);
}
void loop()
{
int IRState = digitalRead(IR);
Serial.println(IRState);
if (IRState==0)
{
digitalWrite(relay, LOW);
}
else
{
digitalWrite(relay, HIGH);
}
delay(200);
}
```

XII Result(s):

…….....

…….....

…….....

…….....

…….....

……..

XIII Conclusions and Recommendations:

…… ……

…….

…….

XIV Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1. What is distance at which the IR sensor can detect an object which is used in your laboratory.
2. Discuss methods to prevent relay "chattering" when the IR sensor rapidly changes state?
3. Give the reason: why relay not directly connected to the IR sensor output.

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

XV References/Suggestions for further reading: include websites/links/Virtual lab Link

1 https://www.youtube.com/watch?v=MY8tMPbzW-k
2 https://www.youtube.com/watch?v=zq51oZMzyP0

XVI Assessment Scheme:

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Coding and Debugging ability	25%
02	Making connections of hardware	25%
03	Follow ethical practices.	10%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	20%
02	Timely Submission of report	20%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 04: Interfacing Humidity sensor with NodeMCU

I Practical Significance:

Temperature and Humidity monitoring play a vital role in our day-to-day life. Temperature and Humidity monitoring and control is very important part in home automation, automatic drier incubator and so on. In essence, measuring and displaying temperature and humidity using the DHT11 and NodeMCU offers a cost-effective and efficient way to collect real-time environmental data, which can be used to improve various aspects of life, from home comfort to industrial processes and scientific research.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'.

III Course Level Learning outcome(s)

- Integrate sensors and actuators in IoT based system.

IV Laboratory Learning outcome(s)

- LLO 4.1 Measure and display humidity and temperature using DHT 11 and NodeMCU.

V Relevant Affective Domain related outcome(s)

1. Handle equipment carefully.
2. Follow safety practices.

VI Minimum Theoretical Background.

DHT11: The DHT11 is a commonly used Temperature and humidity sensor that comes with a dedicated NTC to measure temperature and Humidity.DHT11 sensor measures and provides humidity and temperature values serially over a single wire. It can measure the relative humidity in percentage (20 to 90% RH) and temperature in degree Celsius in the range of 0 to 50°C. It has 4 pins; one of which is used for data communication in serial form. Pulses of different TON and TOFF are decoded as logic 1 or logic 0 or start pulse or end of the frame.

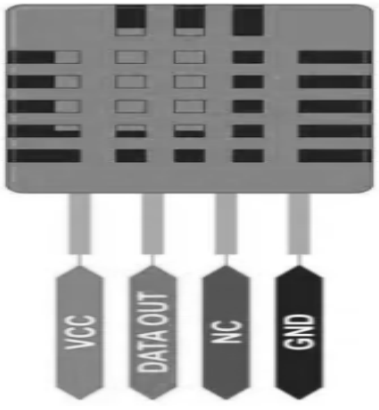


Figure 4.1: DHT11 Humidity sensor

VII Practical setup in Laboratory.

a. Sample:

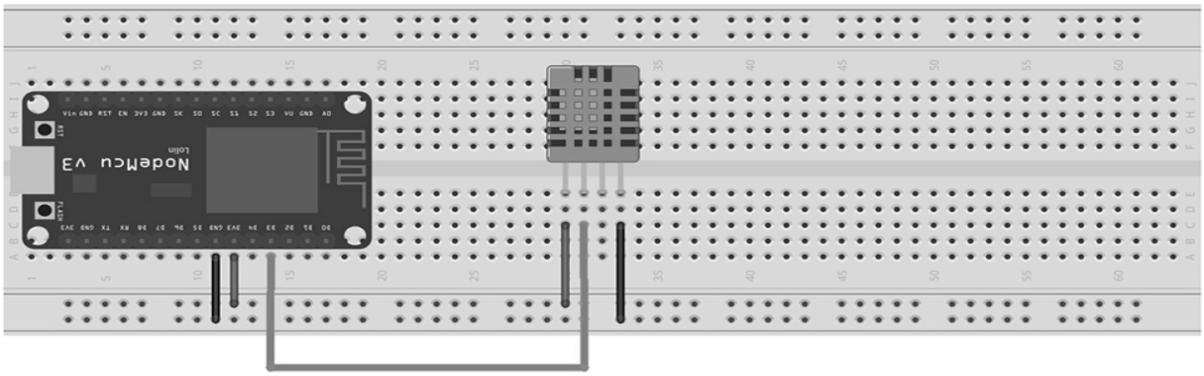


Figure 4.2: Circuit diagram of interfacing NodeMCU with DHT 11

b. Actual practical set up used in Laboratory:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1.	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2.	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x)	1
3.	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V	1

		Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) Analog Input: 1 (10-bit ADC, 0–1V input range)	
4.	Temperature Humidity Sensor DHT11	Temperature Range: 0°C to 50°C (±2°C accuracy) Humidity Range: 20% to 90% RH (±5% RH accuracy) Operating Voltage: 3.3V to 5.5V Output: Digital (1-wire protocol) Sampling Rate: 1 reading per second (1 Hz)	1
5.	Micro-USB	-	1
6.	Connecting wires	-	As per Requirement

IX Precautions to be followed

1 Connect pins of DHT11 sensor in accurate manner.

X Procedure:

1. Interface DHT11 to NodeMCU as per circuit diagram shown in fig. 4.2.
2. Develop program using Arduino IDE software or any other relevant software tool.
3. Compile program on IDE.
4. Upload the program.
5. Observe the output.

XI Sample Program:

```
#include <dht.h>
#define DHT11_PIN 2
void setup()
{
Serial.begin(9600);
}
void loop()
{
int chk = DHT.read11(DHT11_PIN);
Serial.print("Temperature = ");
Serial.println(DHT.temperature);
Serial.print("Humidity = ");
Serial.println(DHT.humidity);
delay(1000);
}
```

XII Result(s)

……

……

……

……..

XIII Conclusions

……

……

……

……

XIV Practical Related Questions

Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1 How do environmental factors influence the accuracy of humidity sensor readings?
2 Give the difference between DHT11 and DHT22 sensors?
3 Write steps to calibrate humidity sensor if readings seem off?

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

XV References/Suggestions for further reading: include websites/ links/Virtual lab Link

1 https://www.youtube.com/watch?v=I3FOa2UW7LI

2 https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet- Translated-Version-1143054.pdf?srsltid=AfmBOorgT_nyX- Ln5Bkzw0cT-q-wV_T3pEolcITAe1wiC9-GU7MlCQXH

XVI Assessment Scheme:

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Coding and Debugging ability	25%
02	Making connections of hardware	25%
03	Follow ethical practices.	10%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	20%
02	Timely Submission of report	20%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 5: Interfacing PIR Sensor with NodeMCU.

I Practical Significance:

The PIR sensor identifies motion by detecting variations in infrared radiation, usually caused by the movement of humans or animals. When connected to the Wi-Fi-enabled NodeMCU, the motion data it captures can be used to initiate automated responses such as switching on lights, sending instant notifications, or triggering alarms. This combination is commonly implemented in smart home systems for intrusion detection and in elderly care applications.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'.

III Course Level Learning outcome(s)

- Integrate sensors and actuators in IoT based system.

IV Laboratory Learning outcome(s)

- LLO 5.1 Motion detection using PIR sensor and NodeMCU.

V Relevant Affective Domain related outcome(s)

- Handle equipment carefully
- Follow safety practices

VI Relevant Theoretical Background.

PIR: PIR sensors allow you to sense motion. They are small, inexpensive, low- power, easy to use so commonly used at home, medical, factories etc. A passive infrared sensor (PIR Sensor) is an electronic sensor that measures infrared light radiating from objects. PIR sensors mostly used in PIR-based motion detectors. PIR sensor i.e. Passive Infrared Sensor, passive word indicates PIR Sensor does not generate or radiate any energy for detection purposes. PIR Sensors don't detect or measure "HEAT"; they detect the infrared radiation emitted or reflected from objects.

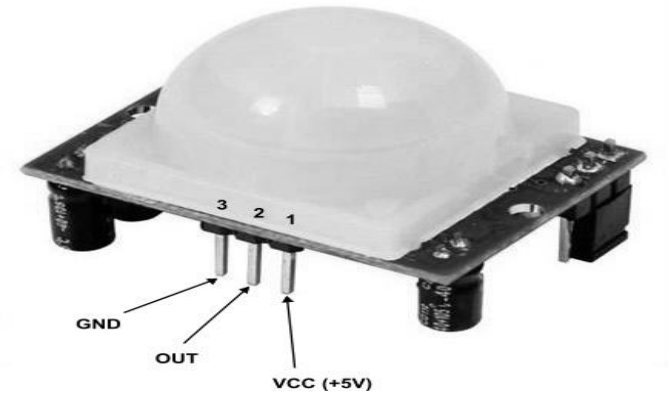


Figure 5.1: PIR sensor

- **Pin1:** It corresponds to the drain terminal of the device, which connected to the positive supply 5V DC.
- **Pin2:** It corresponds to the source terminal of the device, which connects to the ground terminal via a 100K or 47K resistor. The Pin2 is the output pin of the sensor. The pin 2 of the sensor carries the detected IR signal to an amplifier.
- **Pin3:** It of the sensor connected to the ground.

VII Practical setup in Laboratory

a. Sample:

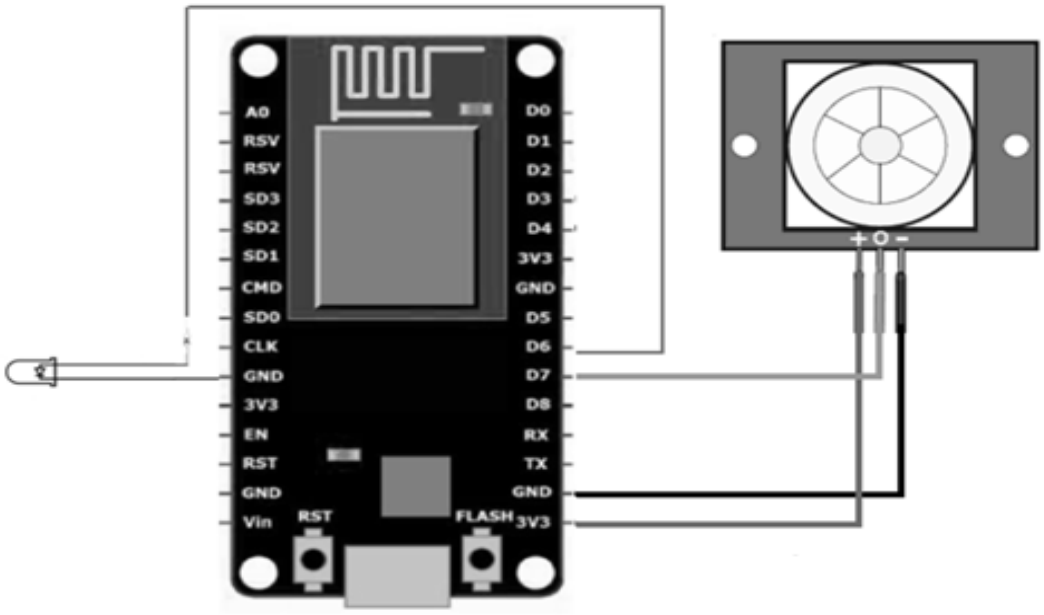


Figure 5.2: Circuit diagram of interfacing NodeMCU with PIR sensor

b. Actual practical set up used in Laboratory:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1.	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2.	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x)	1
3.	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) Analog Input: 1 (10-bit ADC, 0–1V input range)	1
4.	PIR Sensor HC-SR501	Sensing Method: Passive Infrared (PIR) Detection Range: 3 to 7 meters (adjustable) Detection Angle: ~120° cone Operating Voltage: 4.5V to 20V DC Output Type: Digital (High = motion detected, Low = no motion) Delay Time: 3 to 300 seconds (adjustable) Trigger Modes: Repeatable & Non-repeatable Power Consumption: Very low	1
5.	Resistor	100Ω	1
6.	LED	Red/Green	1
7.	Micro-USB	-	1
8.	Connecting wires	-	As per requirement

IX Precautions to be followed

1. Connect PIR sensor in accurate manner.

X Procedure:

1. Interface PIR sensor to NodeMCU as per circuit diagram shown in fig. 5.2.
2. Develop program using Arduino IDE software or any other relevant software tool.
3. Compile program on IDE.
4. Upload the program.
5. Observe the output.

XI Sample Program

```
int Status = 12;
int sensor = 13;
void setup()
{
pinMode(sensor, INPUT);
pinMode(Status, OUTPUT);
}
void loop()
{
long state = digitalRead(sensor);
delay(1000);
if(state == HIGH)
{
digitalWrite (Status, HIGH);
Serial.println("Motion detected!");
}
else
{
digitalWrite (Status, LOW);
Serial.println("Motion absent!");
}
}
```

XII Result(s):

……

……

……

……

XIII Conclusions:

……

……

……

……

XIV Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1. Implement smart security system using a PIR sensor and NodeMCU that sends message on a mobile when motion is detected.
2. List the advantages and limitations of using PIR sensors in motion detection systems.
3. Can the PIR sensor detect motion behind glass or walls? Why or why not?

[Space for Answers]

……

……

……

……

……

……

……

……

……

XV References/Suggestions for further reading:

1 https:// www.instructables.com/Interface-PIR-Sensor-With-NodeMCU/

2 https:// www.youtube.com/watch?v=Bjga_UqT-Do

XVI Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Coding and Debugging ability	25%
02	Making connections of hardware	25%
03	Follow ethical practices	10%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	20%
02	Timely Submission of report	20%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 6: Connecting NodeMCU to a Wi-Fi network

I Practical Significance:

Configuring a NodeMCU to connect to a Wi-Fi network enables it to access the internet, allowing it to communicate with other devices, servers, and cloud platforms. This capability is crucial for various Internet of Things (IoT) applications, enabling remote monitoring, control, and data logging.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'

III Course Level Learning outcome(s)

- Manage IoT communication for data handling.

IV Laboratory Learning outcome(s)

- LLO 6.1- Configure NodeMCU to connect to a Wi-Fi network and troubleshoot connectivity issue.

V Relevant Affective Domain related outcome(s)

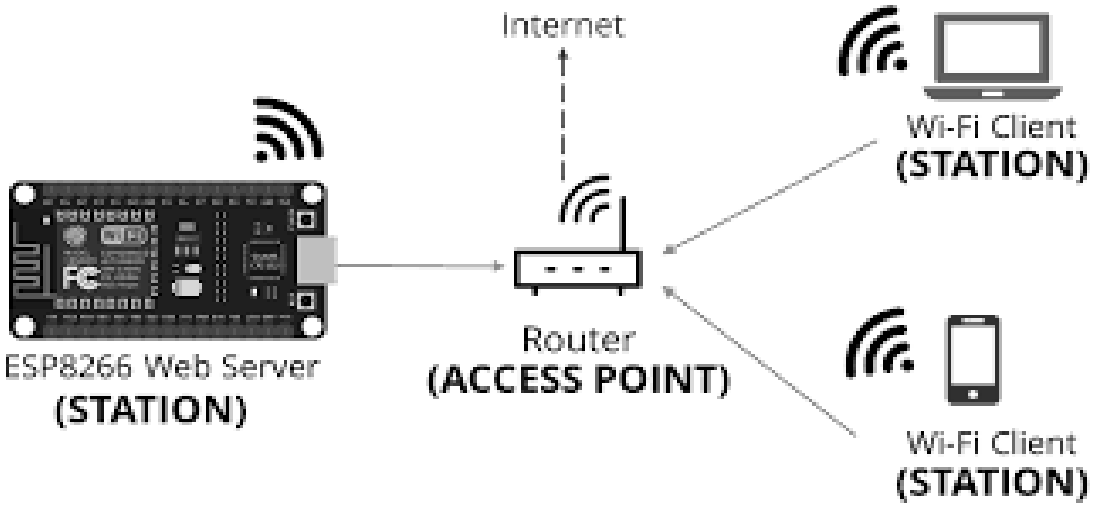
- Follow safe practices.
- Maintain tools and equipment.
- Follow ethical practices.

VI Relevant Theoretical Background:

NodeMCU can run in one of four Wi-Fi modes:

- Station mode (STA), where the NodeMCU device joins an existing network.
- Access point (AP) mode, where it creates its own network that others can join.
- Station + AP mode, where it both creates its own network while at the same time being joined to another existing network.
- Wi-Fi off, where NodeMCU can also have its Wi-Fi functionality completely disable.

Station (STA) Mode: STA mode is used to connect the ESP8266 to a pre-existing Wi-Fi network. This connection is made from an Access Point, which will be responsible for managing information traffic.
In this mode, ESP8266 will just act like smart phone or laptop. It will get connected to existing Wi-Fi channel or in most cases the Wi-Fi Advertised by your router. Once you have programmed it ESP8266 in STA mode, you are successfully connected to stable Wi-Fi. You can access any webpage on the internet Configuring at NodeMCU in station mode, we need to assign our access point or state mobile hotspot's, Service Set Identifier (SSID) which is name of the Wi-Fi and password in ESP8266 code. When uploaded with the code will search for the SSID and password of the hotspot, and if connected will show the status as connected.



For configuration and use on the Arduino platform need to include the ESP8266WiFi.h library in the code. Simple to use and extremely powerful, this library offers us all the tools to configure the Wi-Fi module, without overloading with flags and registers.

VII Practical setup in Laboratory

a) Sample:

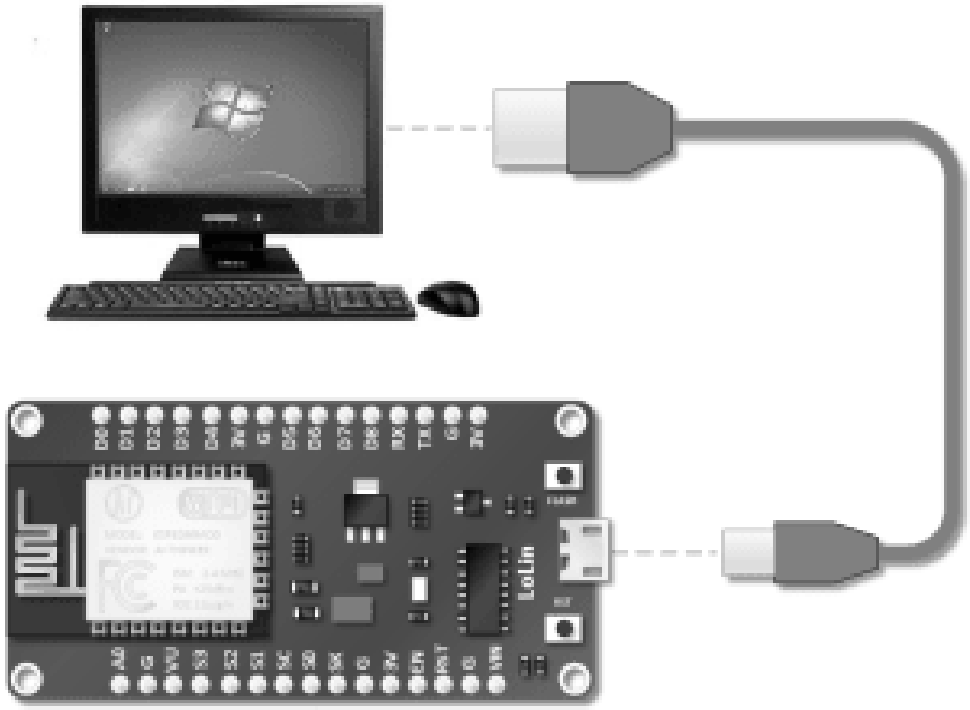


Fig. 6.1: NodeMCU Programming through serial cable

b) Actual Circuit/Experimental Setup used in laboratory with related equipment rating:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x), TypeScript/Electron (v2.x) Operating Systems Supported: Windows 10 or later (64-bit), macOS 10.15 (Catalina) or later, Linux (64-bit Debian/Ubuntu-based distributions)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) PWM Pins: All digital pins support PWM (10-bit resolution) Analog Input: 1 (10-bit ADC, 0–1V	1

		input range) Flash Memory: Typically, 4MB (varies by board) SRAM: ~50 KB available for application usage EEPROM: Emulated in flash	
4	USB To Micro USB cable for connection	Cable length: 1 Meter connector type: USB to USB Male to Male	1

IX Precautions to be followed

1. Follows all step of while installing Arduino IDE software.

X Procedure:

A. Square wave modulation operation:

1. Connect your NodeMCU to the Computer/ Laptop.
2. Install the COM/Serial port driver: Download diver CH340G Driver using following link-
   nodemcu-devkit/Drivers at master · nodemcu/nodemcu-devkit · GitHub
3. Install the Arduino IDE 1.8.3 or higher version.
4. Install the ESP8266 Board Package
   Enter http://arduino.esp8266.com/stable/package_esp8266com_index.json into Additional Board Manager URLs field in the Arduino v1.6.4+ preferences (Open Arduino IDE–>File–>Preferences–>Settings).
   Enter the link and click “OK” to save your changes.
5. Next, use the Board Manager to install the ESP8266 package.
6. Enter the Boards Manager and find the board type.
7. Scroll the Boards Manager screen down to the bottom, A module called “esp8266 by esp8266 Community”, select the latest version and click “Install”.
8. Setup ESP8266, select NodeMCU 1.0 from the Tools->Board dropdown.
9. You also need to select a Port which matches your NodeMCU port
10. Verify and upload the program

XI Program/Code

```
#include <ESP8266WiFi.h>
void setup()
{
Serial.begin(9600);
WiFi.begin("iot", "project1234");
while(WiFi.status() != WL_CONNECTED)
{
Serial.print("..");
delay(200);
```

```
}
Serial.println();

Serial.println("NodeMCU is Connected!");
Serial.println(WiFi.localIP());
}
void loop()
{
}
```

XII Output

(Attach Screen shots of Output generated and code in Sketch)

XIII Result(s):

……

……

……

……

……

XIV Interpretation of Results:

……

……

……

……

……

XV Conclusions and Recommendations:

……

……

……

……

……

XVI Practical Related Questions:

Note: Below given are few sample questions for references. Teachers must design such questions to ensure achievement of identified CO.

1. Write the code for ESP 8266 to check status of Wi-Fi connection and connect it to network?
2. Write the code for ESP 8266 as Access point mode.
3. Explain the use of conditional statement:

```
while(WiFi.status() != WL_CONNECTED)
{
        }
```

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

XVII References/Suggestions for further reading: include websites/links/Virtual lab Link

1 https://www.youtube.com/watch?v=uwjWyMFwvQw&t=23s
2 https://github.com/himanshus2847/Connect-NodeMCU-to-WiFi/blob/master/Connect_to_WiFi_Network/Connect_to_WiFi_Network.ino
3 https://github.com/nodemcu/nodemcu-firmware
4 https://tttapa.github.io/ESP8266/Chap07%20-%20Wi-Fi%20Connections.html

XVIII Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Use of IDE tools for programming	20%
02	Making connections of hardware	20%
03	Follow ethical practices.	20%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	30%
02	Timely Submission of report	10%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 07: Data Transmission from NodeMCU to Webserver

I Practical Significance:

The data transmission from NodeMCU to a webserver enables remote monitoring and control of devices and environments. This allows for real-time data visualization, data logging, and remote management of connected systems. The transmission of data from a NodeMCU to a web server allows for the creation of connected, intelligent systems that can be monitored, controlled, and analyzed remotely, making it a crucial component of modern IoT solutions.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'

III Course Level Learning outcome(s)

- Manage IoT communication for data handling.

IV Laboratory Learning outcome(s)

- LLO 7.1- Use HTTP protocol to send sensor data from NodeMCU to a web server (use any cloud).

V Relevant Affective Domain related outcome(s)

1. Follow safe practices.
2. Maintain tools and equipment.
3. Follow ethical practices.

VI Relevant Theoretical Background.

To connect a NodeMCU (ESP8266) to a web server, can use various cloud platforms like AWS, Google Cloud, Arduino cloud, BLYNK cloud or others. The process typically involves configuring the NodeMCU to connect to your Wi-Fi network, then using an appropriate library in the Arduino IDE to send data to or receive data from the cloud server.

Steps to connect to webserver:

1. Choose a Cloud Platform:

- **AWS IoT Core:** A popular choice for IoT devices, offering MQTT and HTTP communication
- **Google Cloud IoT Core:** Another robust platform with similar features to AWS.
- Arduino Cloud supports secure connections with boards via Wi-Fi®, LoRa®-enabled devices, Ethernet and Cellular (GSM/NB-IoT) and REST API and command line tools for larger scale automations

- **Thingspeak:** A simpler platform, often used for basic data logging and visualization
- **Blynk:** A user-friendly platform with mobile app integration for controlling and monitoring devices.
- **Azure IoT Hub:** Microsoft's cloud platform for IoT solutions
- **Other Platforms:** Consider platforms like Adafruit IO, Particle, or building your own server using Node-RED or similar technologies

2. **Setup Cloud Platform Account and Credentials:**
   - Create an account on your chosen cloud platform.
   - Follow the platform's instructions to create a new project or device
   - Obtain necessary credentials like API keys, device IDs, or MQTT broker details
3. **Install Necessary Libraries in Arduino IDE:**
   - **ESP8266WiFi:** For connecting the NodeMCU to your Wi-Fi network
   - **WiFiClientSecure (if using HTTPS):** For secure communication
   - **Libraries specific to your chosen cloud platform:** For example, AWS IoT Device SDK for AWS, or libraries for MQTT communication like PubSubClient)
4. **Configure NodeMCU for Wi-Fi Connection:**
   - Include the ESP8266WiFi library in your Arduino code
   - Use WiFi.begin(ssid, password) to connect to your Wi-Fi network
   - Obtain the NodeMCU's IP address using WiFi.localIP()
5. **Implement Communication with the Cloud Server:**
   - **MQTT (Message Queuing Telemetry Transport):** A lightweight protocol often used in IoT
     - Use the PubSubClient library
     - Connect to the MQTT broker using client.connect(clientId, username, password)
     - Subscribe to topics using client.subscribe(topic)
     - Publish data to topics using client.publish(topic, payload)
   - **HTTP (Hypertext Transfer Protocol):**
     - Use the WiFiClient or WiFiClientSecure class
     - Establish a connection to the server using client.connect(host, port)
     - Send HTTP requests using client.print(), client.println()
     - Read the response using client.read()

Blynk- is a complete IoT software platform where you can prototype, deploy, and remotely manage connected electronic devices at any scale

Components of BLYNK:

1. **Blynk.Console** is a feature-rich web application catering to different types of users.
2. **Blynk.Apps** is a versatile native iOS and Android mobile application that provides Remote monitoring and control of connected devices, Configuration of mobile UI during prototyping and production stages, Automation of connected device operations.
3. **Blynk.Edgent** is a packaged solution designed to simplify the connection of supported devices to the Blynk platform, providing access to all its advanced

features without extensive coding

4. **Blynk.Library** is a user-friendly and portable C++ library, that comes pre-configured to work with hundreds of development boards
5. **Blynk.Cloud** is a server infrastructure acting as the heart of Blynk IoT platform binding all the components together.
6. Blynk provides micro-services, which are software modules that work across products and perform specific functionalities. -Blynk.R, Blynk.air, A micro- service focused on Firmware Over-the-Air (OTA) Updates

VII Practical setup in Laboratory.

a) Sample:

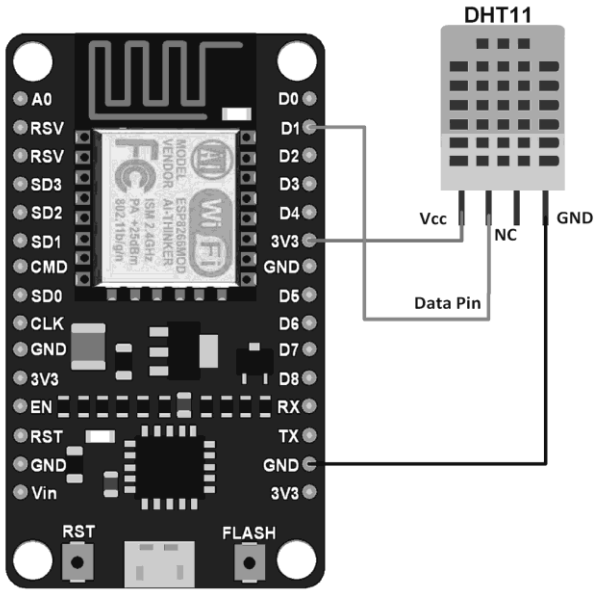


Fig. 7.3: NodeMCU and DHT11 connection to send data to Server

b) Actual setup used in Laboratory:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x), TypeScript/Electron (v2.x) Operating Systems Supported: Windows 10 or later (64-bit), macOS 10.15 (Catalina) or later, Linux (64-bit Debian/Ubuntu-based distributions)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) PWM Pins: All digital pins support PWM (10-bit resolution) Analog Input: 1 (10-bit ADC, 0–1V input range) Flash Memory: Typically, 4MB (varies by board) SRAM: ~50 KB available for application usage EEPROM: Emulated in flash	1
4	USB To Micro USB cable for connection	Cable length: 1 Meter connector type: USB to USB Male to Male	1
5	DHT11	Operating Voltage: 3.5V to 5.5V Operating current: 0.3mA (measuring) 60uA (standby) Output: Serial data Temperature Range: 0°C to 50°C Humidity Range: 20% to 90% Resolution: Temperature and Humidity both are 16-bit	1

IX Precautions to be followed

1. Follow all step of while installing Arduino IDE software.
2. Connect Arduino board to Wi-Fi and cloud properly.

X Procedure:

1. **Set up Blynk IoT:**
   - Download and install the Blynk IoT app on your smartphone
   - Create a new device in the app and connect it to your Wi-Fi network
   - Create a new template for your device, which includes adding DataStream
   - Set up a web dashboard canvas for your device, including widgets like buttons, sliders, or displays
   - Create a device within the Blynk console, linking it to the template you just created
2. **Set up the Arduino IDE:**
   - Download and install the Arduino IDE.
   - Install the ESP8266 core for Arduino IDE
     - Go to File > Preferences (Arduino > Preferences for Mac)
     - Click on the "Additional Boards Manager URLs" button and add this link: http://arduino.esp8266.com/stable/package_esp8266com_index.json
     - In the Arduino IDE open: Tools > Board > Boards Manager
     - Search for esp8266 and install it.
   - **Install the Blynk library:**
     - Download the latest Blynk library from GitHub
     - Unpack it
     - Move the libraries to your Arduino libraries folder (e.g., C:/User//Documents/Arduino/libraries)
3. **Connect the ESP8266 to your Wi-Fi and Blynk:**
   - Open the Blynk example sketch for ESP8266 (File > Examples > Blynk > Boards_Ethernet > Arduino_Ethernet or similar).
   - Replace the placeholder Wi-Fi credentials and Auth Token with your own.
   - You can also find example sketches on the Blynk website and in the Blynk library examples.
   - Upload the code to your ESP8266.
4. **Interact with your device:**

   Set Up Blynk Cloud (HTTP REST API method):

   1. Go to https://blynk.cloud and create/log in to your account.

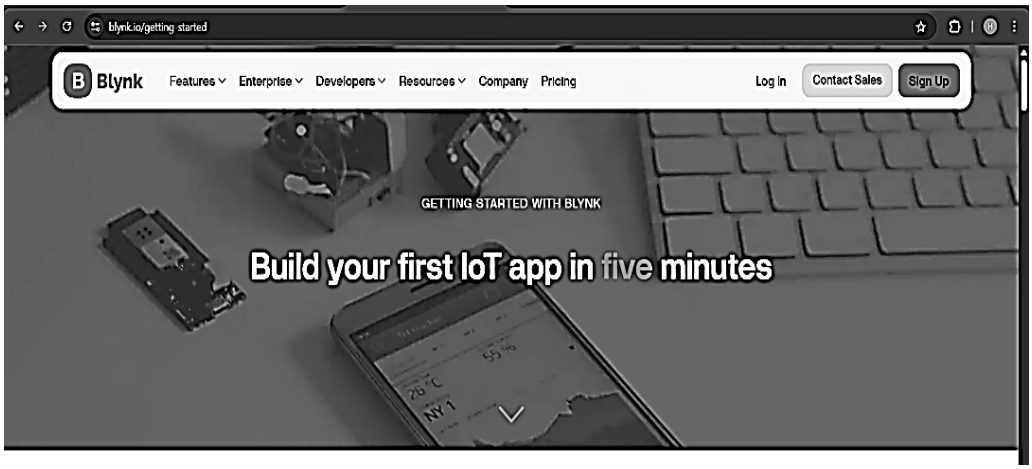


Figure 7.1: Blynk server log in page

2. Create a new Template (e.g., “Sensor Monitor”)

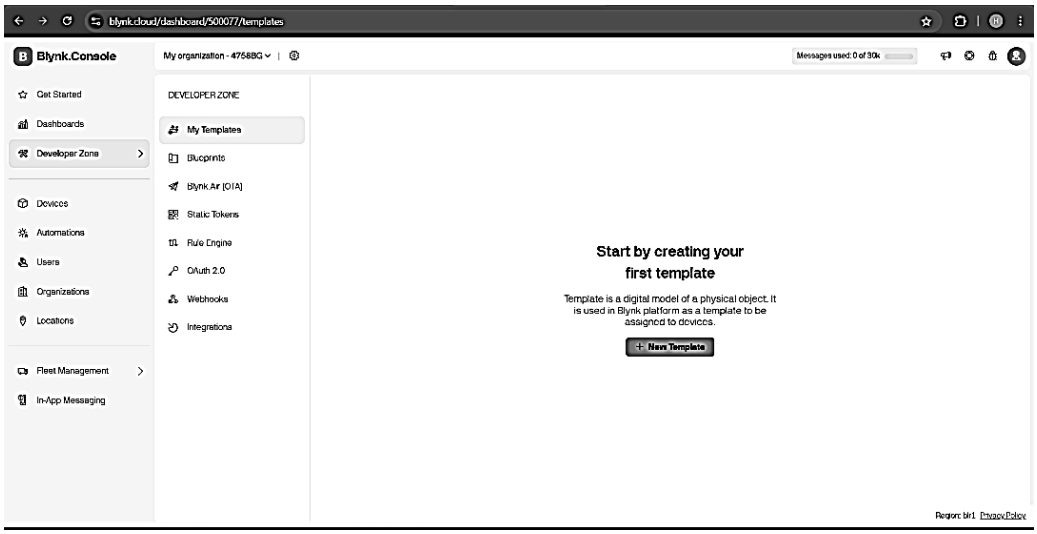


Figure 7.2: Blynk server template creation page

- Add 2 Data streams (Virtual Pins):
  - V0 → Temperature
  - V1 → Humidity

3. Create a new Device based on that Template

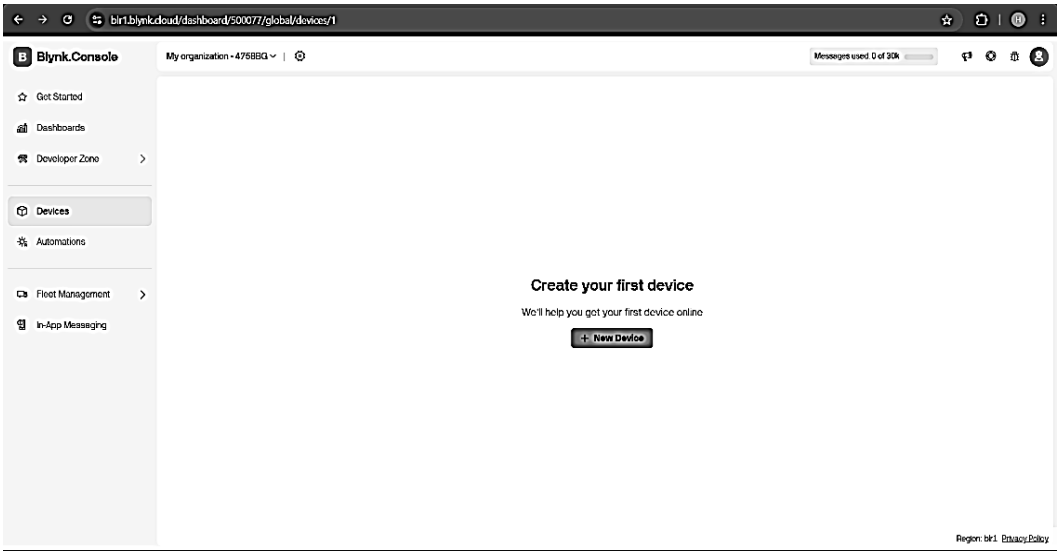


Figure 7.3: Blynk server Device creation page

4. Copy your Device Info which is in the Developer zone:
   a. Device Name
   b. Device ID
   c. Auth Token (IMPORTANT!)

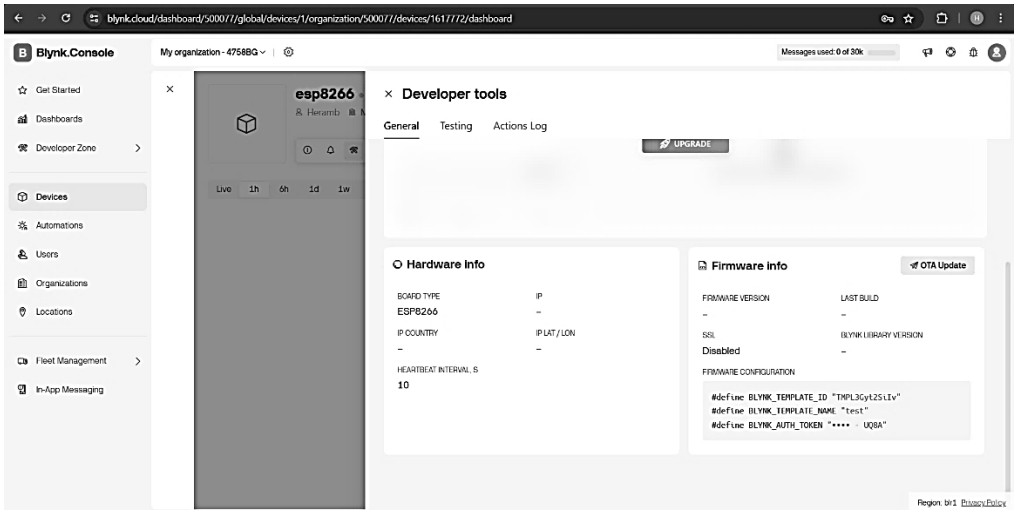


Figure 7.4: Blynk server Developer zone page.

5. To create the dashboard on mobile app follow the following steps in the app of Blynk

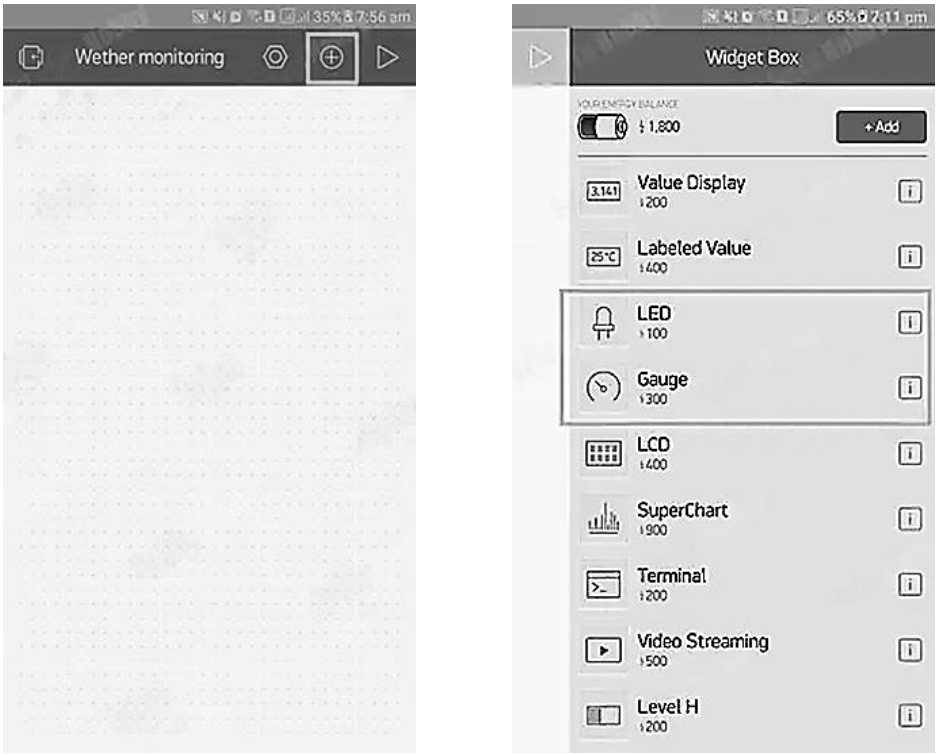


Figure 7.5: Blynk app mobile dashboard

6. Add the widgets as per your need

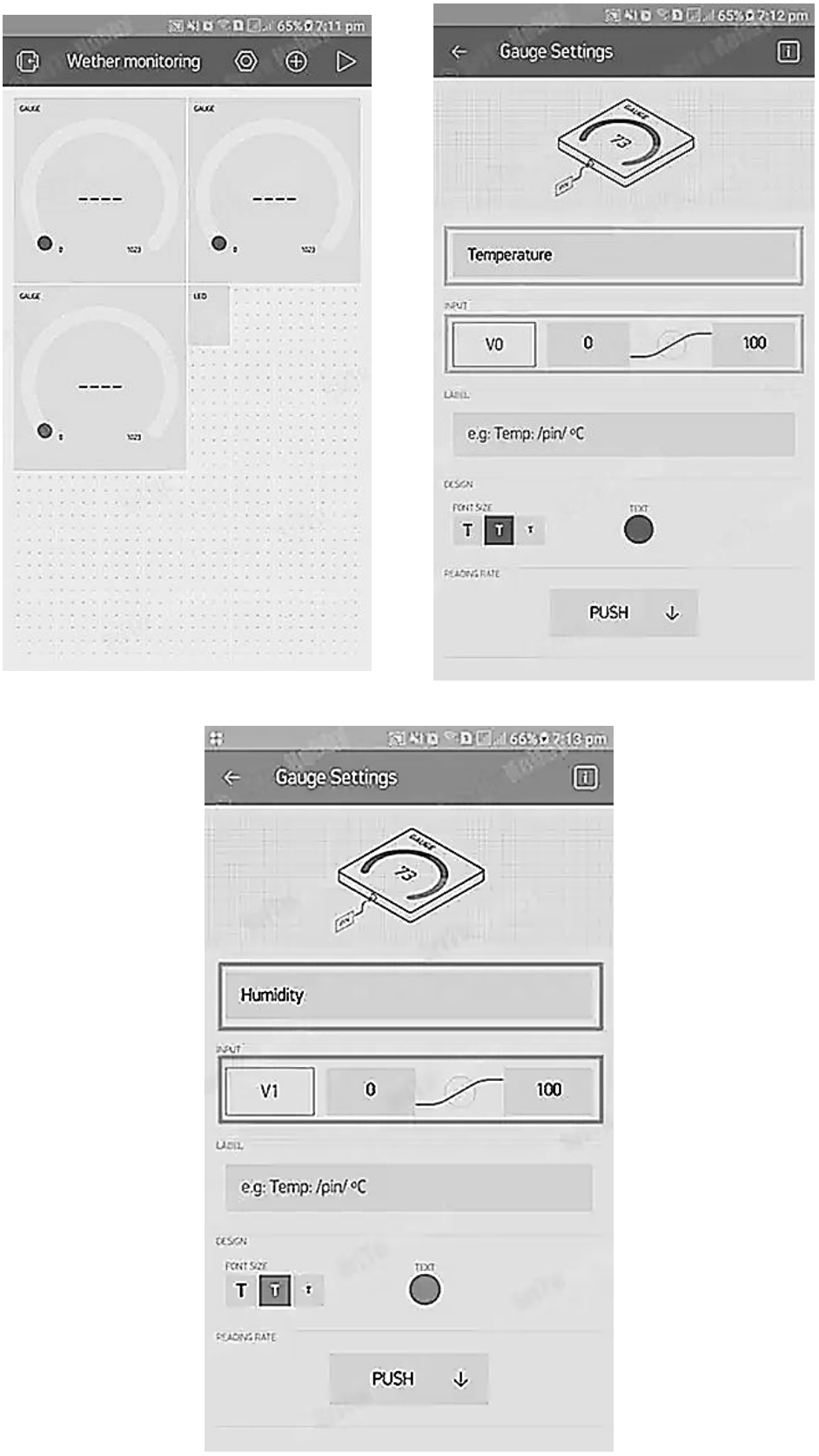


Figure 7.6: Blynk app widget window on mobile

XI Program / Code

```
#include <ESP8266WiFi.h>
#include <DHT.h>
#define DHTPIN D1
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Replace with your Wi-Fi credentials
const char* ssid = "YOUR_WIFI_SSID";
const char* password = "YOUR_WIFI_PASSWORD";

// Blynk HTTP API info
String authToken = "YOUR_BLYNK_AUTH_TOKEN";

const char* host = "blynk.cloud";

void setup() {
Serial.begin(115200);
delay(10);
dht.begin();

WiFi.begin(ssid, password);
Serial.print("Connecting to WiFi");

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

Serial.println("\nWiFi Connected");
}

void loop() {
float h = dht.readHumidity();
float t = dht.readTemperature(); // Celsius

if (isnan(h) || isnan(t)) {
Serial.println("Failed to read from DHT sensor!");
return;
}
WiFiClient client;
if (client.connect(host, 80)) {
// Send temperature to V0
String tempURL = "/external/api/update?token=" + authToken + "&V0=" + String(t);
client.print(String("GET ") + tempURL + " HTTP/1.1\r\n" +
"Host: " + host + "\r\n" +
"Connection: close\r\n\r\n");
delay(1000);
// Send humidity to V1
String humURL = "/external/api/update?token=" + authToken + "&V1=" + String(h);
```

```
client.print(String("GET ") + humURL + " HTTP/1.1\r\n" +
"Host: " + host + "\r\n" +
"Connection: close\r\n\r\n");
Serial.println("Data sent to Blynk:");
Serial.print("Temperature: "); Serial.println(t);
Serial.print("Humidity: "); Serial.println(h);
}
delay(15000);  // Send every 15 seconds
}
```

XII Output

(Attach Screen shots of Output generated and code in Sketch)

XIII Result(s):

……

……

……

……

XIV Interpretation of Results:

……

……

……

……

XV Conclusions and Recommendations:

……

……

……

……

……

XVI Practical Related Questions:
Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1 Write and execute the code for ESP 8266 to connect to Arduino cloud and attach output screen shot on your system.
2 Explain the process of creating a basic web server on the NodeMCU.
3 How to implement data logging and visualization on a local web server (e.g., using Node-RED) instead of relying solely on the Blynk cloud server?
4 What are some potential applications of this project in real-world scenarios?

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

XVII References/Suggestions for further reading: include websites/links/Virtual lab Link

1. https://www.youtube.com/watch?v=BIkBKGVzSsY&t=47s
2. https://docs.blynk.io/en
3. https://iotcircuithub.com/esp8266-iot-project-with-blynk-automation/
4. https://docs.arduino.cc/arduino-cloud/guides/overview/
5. https://www.youtube.com/watch?v=IH45WOAhLtU

XVIII Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Use of IDE tools for programming	20%
02	Making connections of hardware	20%
03	Follow ethical practices	20%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	35%
02	Timely Submission of report	10%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 08: Implementation of MQTT protocol with NodeMCU

I Practical Significance:

Implementing the MQTT protocol with NodeMCU enables efficient, lightweight, and reliable communication for Internet of Things (IoT) and other connected systems. MQTT's publish-subscribe model, combined with NodeMCU's Wi-Fi capabilities, allows for seamless data exchange between devices and servers, making it suitable for various applications like smart homes, industrial automation, and remote monitoring.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)':

III Course Level Learning outcome(s)

- Manage IoT communication for data handling.

IV Laboratory Learning outcome(s)

- LLO 8.1- Set up MQTT communication to publish and subscribe to topics using NodeMCU.

V Relevant Affective Domain related outcome(s)

1. Follow safe practices.
2. Maintain tools and equipment.
3. Follow ethical practices

VI Minimum Theoretical Background.

MQTT is a lightweight publish-subscribe-based messaging protocol.

- It is quicker (faster) than other request-response based APIs like HTTP.
- It is developed on the base of the TCP/IP protocol.
- It allows remote location devices to connect, subscribe, publish, to a specific topic on the server with the help of a message broker.
- MQTT Broker/Message broker is a module in between the sender and the receiver. It is an element for message validation, transformation, and routing.
- The broker is responsible for distributing messages to the interested clients (subscribed clients) of their interested topic

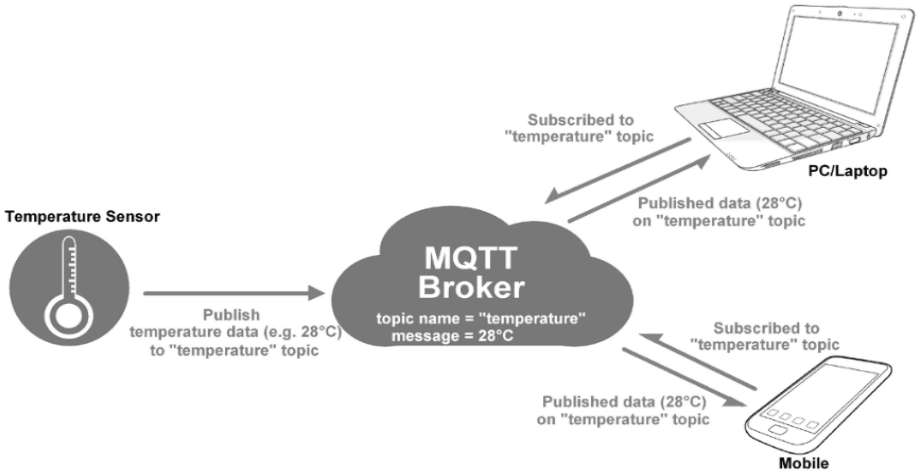


Figure 8.1: MQTT communication

For example, if the temperature sensor publishes the temperature data (message) on the topic "temperature" then interested clients who have subscribed to the "temperature" topic get that published temperature data as shown in the above Figure 8.1.

MQTT is widely used in IoT (Internet of Things) embedded applications, where every sensor is connected to a server and have access to control them over the internet.

NodeMCU is an open-source IoT platform. It is a firmware which runs on ESP8266 Wi-Fi SoC from Espressif Systems. It has onboard wi-fi available through which IoT applications become easy to build.

The MQTT Client module of NodeMCU is according to version 3.1.1 of the MQTT protocol. Make sure that your broker supports and is correctly configured for version 3.1.1. let's see the functions used for MQTT on NodeMCU.

To set up MQTT communication on a NodeMCU (ESP8266), you'll need to install the PubSubClient library, connect to your Wi-Fi network, and configure the MQTT client to connect to your broker. Then, you can publish and subscribe to topics.

VII Practical setup in Laboratory

a) Sample:

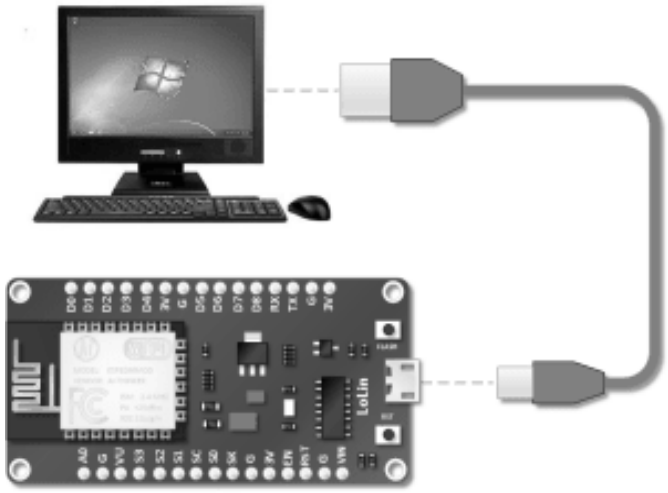


Figure 8.1: NodeMCU connection to send data

b) Actual simulation code used in laboratory:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x), TypeScript/Electron (v2.x) Operating Systems Supported: Windows 10 or later (64-bit), macOS 10.15 (Catalina) or later, Linux (64-bit Debian/Ubuntu-based distributions)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) PWM Pins: All digital pins support PWM (10-bit resolution) Analog Input: 1 (10-bit ADC, 0–1V input range) Flash Memory: Typically, 4MB (varies by board) SRAM: ~50 KB available for application usage EEPROM: Emulated in flash	1
4	USB To Micro USB cable for connection	Cable length: 1 Meter connector type: USB to USB Male to Male	1

IX Precautions to be followed

1 Follows all step of while installing Arduino IDE software.
2 Connect Arduino board to Wi-Fi and cloud properly.

X Procedure:

a. Install Arduino IDE
b. Install ESP8266 Board Support
   - Go to File > Preferences
   - Add this URL to Board Manager: http://arduino.esp8266.com/stable/package_esp8266com_index.json
   - Go to Tools > Board > Board Manager → Search and install ESP8266
c. Install Libraries
   - Install PubSubClient
   - Install ESP8266Wi-Fi
d. Connect NodeMCU to PC via USB
e. Setup Wi-Fi and MQTT connection.
f. User a free MQTT Broker (e.g., broker.hivemq.com)
   - Use a phone app (MQTT Dashboard) or PC software (MQTT.fx) to test MQTT
g. Write and Upload Code via Arduino IDE

XI Program Code

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
// WiFi credentials
const char* ssid = "YOUR_SSID";
const char* password = "YOUR_PASSWORD";
// MQTT Broker
const char* mqtt_server = "broker.hivemq.com";
WiFiClient espClient;
PubSubClient client(espClient);
void setup_wifi()
{
   delay(10);
   Serial.println();
   Serial.print("Connecting to ");
   Serial.println(ssid);
   WiFi.begin(ssid, password);
   while (WiFi.status() != WL_CONNECTED)
 {
    delay(500);
    Serial.print(".");
 }
    Serial.println("\nWiFi connected");
 }
```

```
void callback(char* topic, byte* payload, unsigned int length)
{
   Serial.print("Message arrived [");
   Serial.print(topic);
   Serial.print("]: ");
   for (unsigned int i = 0; i < length; i++)
{
   Serial.print((char)payload[i]);
}
   Serial.println();
}
void reconnect()
{
   while (!client.connected())
{
   Serial.print("Attempting MQTT connection...");
   if (client.connect("ESP8266Client"))
{
     Serial.println("connected");
     client.subscribe("esp8266/test");
}
else
{
     Serial.print("failed, rc=");
     Serial.print(client.state());
     delay(5000);
  }
 }
}

void setup()
{
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop()
{
  if (!client.connected())
{
   reconnect();
 }
   client.loop();

  // Publish message every 5 seconds
  static unsigned long lastMsg = 0;
  if (millis() - lastMsg > 5000) {
  lastMsg = millis();
  String msg = "Hello from ESP8266";
```

```
  client.publish("esp8266/test", msg.c_str());
  Serial.println("Published: " + msg);
 }
}
```

XII Output

(Observe output on Serial monitor of Arduino IDE and Attach Screen shots of Output generated and code in Sketch)

XIII Result(s):

……

……

……

……

XIV Interpretation of Results:

……

……

……

……

XV Conclusions and Recommendations:

……

……

……

……

XVI Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1. How do you install and configure an MQTT broker (e.g., Mosquitto)? Discuss local and cloud-based options.
2. How do you choose a suitable MQTT client library for the NodeMCU (e.g.PubSubClient)
3. State how to implement secure communication between NodeMCU and the web server (MQTT)?
4. If the RCS of the target in Q1 drops from 1 m^2 to 0.01 m^2 (e.g., stealth aircraft), what will be the new maximum range?

[Space for Answers]

……

……

……

……

……

……

……

……

……

……

……

……

……

XVII References/Suggestions for further reading: include websites/links/Virtual lab Link

1 https://youtu.be/kOs2XetCWII
2 https://www.electronicwings.com/nodemcu/nodemcu-mqtt-client-with-arduino-ide
3 https://www.youtube.com/watch?v=UArq-UfC-d8

XVIII Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Use of IDE tools for programming	20%
02	Making connections of hardware	20%
03	Follow ethical practices	20%
Product Related (10 Marks)		**40%**
01	Correctness of output	30%
02	Interpretation of Result	10%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 09: Monitoring and Controlling Light intensity using NodeMCU

I Practical Significance:

The data transmission from NodeMCU to a webserver enables remote monitoring and control of devices and environments. This allows for real-time data visualization, data logging, and remote management of connected systems. Monitoring and controlling light intensity using NodeMCU has practical significance in various applications, primarily for energy conservation and improved user experience. By adjusting light levels based on ambient conditions or user preferences, it's possible to significantly reduce energy consumption, particularly in smart lighting systems for homes and streets. Additionally, this technology can enhance user comfort and productivity in environments like greenhouses, where precise light control is crucial for plant growth.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'

III Course Level Learning outcome(s)

- Manage IoT communication for data handling.

IV Laboratory Learning outcome(s)

- LLO 9.1- Measure data from LDR to monitor light intensity and transmit it to cloud.
- LLO 9.2 Control intensity of LED according to the data received from the cloud

V Relevant Affective Domain related outcome(s)

1. Follow safe practices.
2. Maintain tools and equipment.
3. Follow ethical practices.

VI Minimum Theoretical Background.

To monitor and control light intensity using a NodeMCU and Blynk, need to connect a light-dependent resistor (LDR) to the NodeMCU's analog pin and an LED to a digital pin. The Blynk app will be used to display the light intensity reading from the LDR and control the LED's brightness using PWM.
To connect a NodeMCU (ESP8266) to a web server, can use various cloud platforms like AWS, Google Cloud, Arduino cloud, BLYNK cloud or others. The process typically involves configuring the NodeMCU to connect to your Wi-Fi network, then using an appropriate library in the Arduino IDE to send data to or receive data from the cloud server.

Steps to connect to Webserver:

1) Choose a Cloud Platform
2) Setup Cloud Platform Account and Credentials
3) Install Necessary Libraries in Arduino IDE

4) Configure NodeMCU for Wi-Fi Connection
5) Implement Communication with the Cloud Server: MQTT or HTTP

Blynk- is a complete IoT software platform where you can prototype, deploy, and remotely manage connected electronic devices at any scale. Blynk App is a versatile native iOS and Android mobile application that provides Remote monitoring and control of connected devices, Configuration of mobile UI during prototyping and production stages, Automation of connected devices

VII Practical setup in Laboratory

a) Simple:

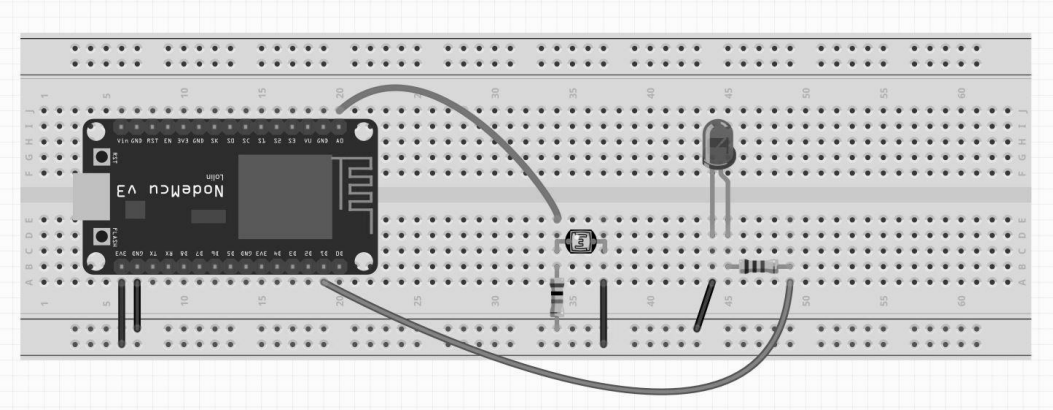


Fig 9.1: NodeMCU and LDR connection to send data to Server

b) Actual Setup used in Laboratory:

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x), TypeScript/Electron (v2.x) Operating Systems Supported: Windows 10 or later (64-bit), macOS 10.15 (Catalina) or later, Linux (64-bit Debian/Ubuntu-based distributions)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) PWM Pins: All digital pins support PWM (10-bit resolution) Analog Input: 1 (10-bit ADC, 0–1V input range) Flash Memory: Typically, 4MB (varies by board) SRAM: ~50 KB available for application usage EEPROM: Emulated in flash	1
4	USB To Micro USB cable for connection	Cable length: 1 Meter connector type: USB to USB Male to Male	1
5	LDR	Operating Voltage: 3.3V to 5V DC. Operating Current: 15mA.	1
6	LED	Forward voltage- 2 v Current – 20 mA	

IX Precautions to be followed:

1. Follows all step of while installing Arduino IDE software.
2. Connect Arduino board to Wi-Fi and cloud properly.

X Procedure:

1. **Set up Blynk IoT:**
   - Download and install the Blynk IoT app on your smartphone.
   - Create a new device in the app and connect it to your Wi-Fi network.

- Create a new template for your device, which includes adding DataStream.
- Set up a web dashboard canvas for your device, including widgets like buttons, sliders, or displays.
- Create a device within the Blynk console, linking it to the template you just created.

2. **Set up the Arduino IDE:**
   - Download and install the Arduino IDE.
   - Install the ESP8266 core for Arduino IDE.
     - Go to File > Preferences (Arduino > Preferences for Mac).
     - Click on the "Additional Boards Manager URLs" button and add this link: http://arduino.esp8266.com/stable/package_esp8266com_index.json.
     - In the Arduino IDE open: Tools > Board > Boards Manager.
     - Search for esp8266 and install it.
   - Install the Blynk library:
     - Download the latest Blynk library from GitHub.
     - Unpack it.
     - Move the libraries to your Arduino libraries folder (e.g., C:/User//Documents/Arduino/libraries).
3. **Connect the ESP8266 to your Wi-Fi and Blynk:**
   - Open the Blynk example sketch for ESP8266 (File > Examples > Blynk > Boards_Ethernet > Arduino_Ethernet or similar).
   - Replace the placeholder Wi-Fi credentials and Auth Token with your own.
   - Upload the code to your ESP8266.
4. **Interact with your device:**
   - Set Up Blynk Cloud (HTTP REST API method)
5. Go to https://blynk.cloud and create/log in to your account

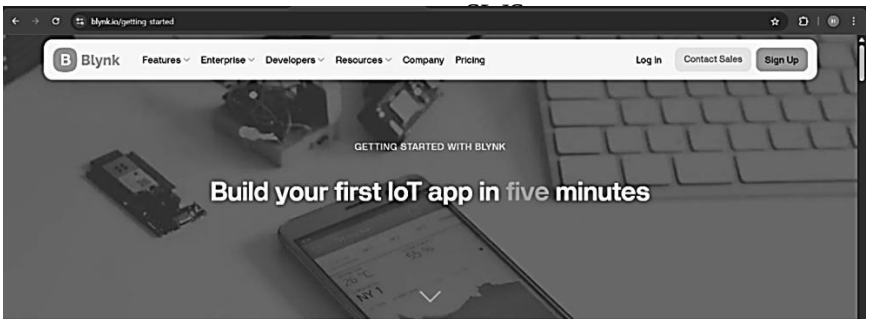


Fig 9.2: Login window if BLYNK app to send data to Server

1. Create a new Template (e.g., “Sensor Monitor”).
2. Add the widgets as per your need
3. Write and Upload Code
4. Code will read analog value from LDR and send it to cloud at regular intervals
5. View Data on Mobile App or Dashboard
6. Cloud (Blynk) sends value from the slider to the ESP8266
7. ESP8266 uses that value to control LED brightness (PWM)
8. Use Blynk App to Control Brightness in Real Time

XI Program / Code

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#define led D1
#define sensor A0

// Auth Token from Blynk
char auth[] = "YourAuthToken ";

// WiFi credentials
char ssid[] = "YourSSID ";
char pass[] = "YourPassword ";

BlynkTimer timer;

void setup()
{

  // Debug console
  pinMode(led, OUTPUT);
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(500L, sendSensor);

}
void loop()
{

  Blynk.run();
  timer.run();

}
void sendSensor()
{

  int LDR = analogRead(sensor);
  if(LDR <150)

{

  digitalWrite(led, HIGH);
  Blynk.notify("Light ON");

}
else
{

  digitalWrite(led, LOW);
}
```

```
Blynk.virtualWrite(V5, LDR);

}
```

XII Output

(Attach Screen shots of Output generated and code in Sketch)

XIII Result(s):

……

……

……

……

XIV Interpretation of Results:

……

……

……

……

XV Conclusions and Recommendations:

……

……

……

……

XVI Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1. Explain the role of the Blynk authentication token in connecting the NodeMCU to the Blynk app
2. How are the widgets on the Blynk dashboard connected to the corresponding virtual pins in the code
3. What are some common problems that might arise during the project (e.g., connectivity issues, incorrect readings, LED not responding) and how can they be resolved?
4. Write and execute same code using Arduino cloud

[Space for Answers]

……

……

……

……

……

……

XVII References/Suggestions for further reading: include websites/links/Virtual lab Link

1. https://www.youtube.com/watch?v=ej06oRWQ9j4
2. https://eldontronics.wordpress.com/2017/09/07/iot-with-nodemcu-and-blynk/#:~:text=Our%20goal%20in%20this%20activity%20is%20to,start%20with%2C%20we%20need%20the%20following%20components:
3. https://www.youtube.com/watch?v=Z701jeS98eo
4. https://www.youtube.com/watch?v=BIkBKGVzSsY&t=47s
5. https://docs.blynk.io/en
6. https://iotcircuithub.com/esp8266-iot-project-with-blynk-automation/

XVIII Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Use of IDE tools for programming	20%
02	Making connections of hardware	20%
03	Follow ethical practices.	20%
Product Related (10 Marks)		**40%**
01	Relevance of output of the problem definition	30%
02	Timely Submission of report	10%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	

Practical No. 10: Implementation of IOT enabled smart home application.

I Practical Significance:

An IoT-enabled smart home has enhanced convenience, energy efficiency, and security. Smart homes allow for remote control and automation of various devices, leading to a more comfortable and streamlined living experience. Furthermore, they can optimize energy consumption, potentially lowering utility bills and promoting sustainable living. Security is also enhanced through features like remote monitoring and automated alerts. It helps in monitoring environmental changes providing insights into potential health risks and monitor.

II Industry / Employer Expected outcome(s)

- This practical is expected to develop the following skill:
  'Maintain system based on Internet of Things (IoT)'

III Course Level Learning outcome(s)

- Develop IoT based application

IV Laboratory Learning outcome(s)

- LLO 10.1- Design a smart home system using NodeMCU to control lights, fans and locking system

V Relevant Affective Domain related outcome(s)

1. Follow safe practices.
2. Maintain tools and equipment
3. Follow ethical practices.

VI Relevant Theoretical Background.

An IoT-enabled smart home application allows users to control and automate various devices and systems within their homes through internet connectivity. This can range from simple tasks like turning lights on and off to more complex functions like remotely adjusting the thermostat or monitoring security cameras. The implementation involves integrating smart devices, a central hub or application, and potentially voice assistants to create a connected and automated living space.

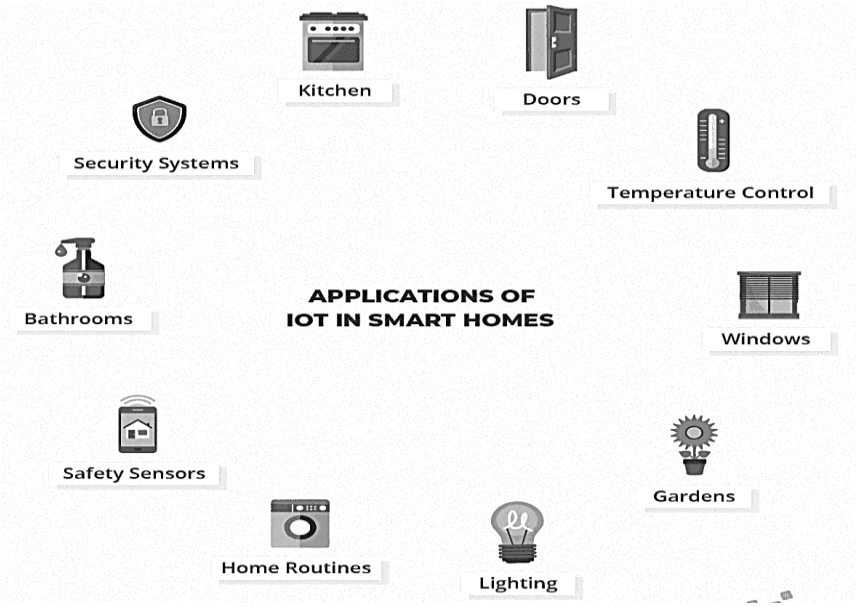


Fig 10.1: Applications of IoT in Smart Homes

An IoT automation system has a complex architecture that includes remote servers with sensors. The servers are located on the cloud and can manage many sensors at the same time. IoT sensors can use Bluetooth, ZigBee, Wi-Fi, and Z-Wave for communication. The main device in the system is a controller, often referred to as a hub or gateway.
It is connected to the home router via Ethernet. The sensors send and receive commands via this centralized gateway. The gateway then takes this communication to the cloud. This means that all the devices are interconnected, and it is possible to set up a desired sequence of actions. This interconnectedness also makes it possible to monitor IoT devices remotely via an app.

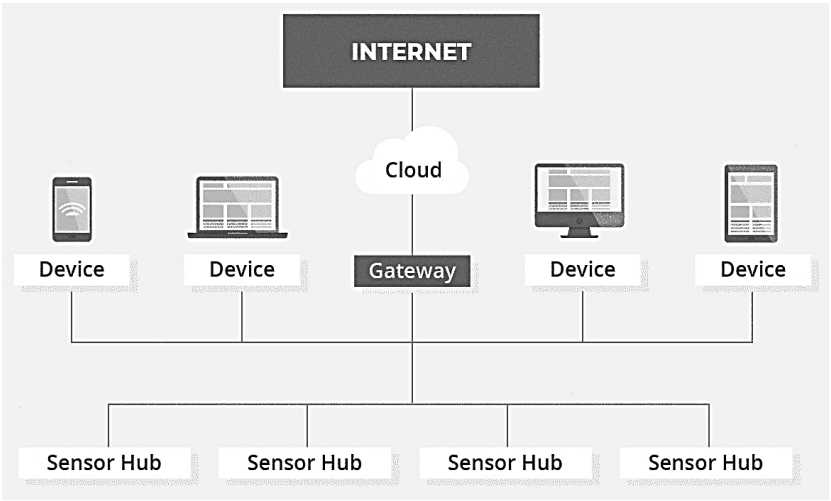


Fig. 10.2: Architecture of Smart home system

List of components used in system:

Gateway- NodeMCU, ESP 8266 is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) with inbuilt Wi-Fi capability.

Cloud -Blynk is a complete IoT software platform where you can prototype, deploy, and remotely manage connected electronic devices at any scale. Blynk. Apps is a versatile

native iOS and Android mobile application that provides Remote monitoring and control of connected devices, Configuration of mobile UI during prototyping and production stages, Automation of connected devices.

Devices - Internet enabled Smart phones and Laptop.

Actuators – Relays to control fan and light, Servo motor to control door opening and closing.

VII Practical setup in Laboratory

a) Sample:

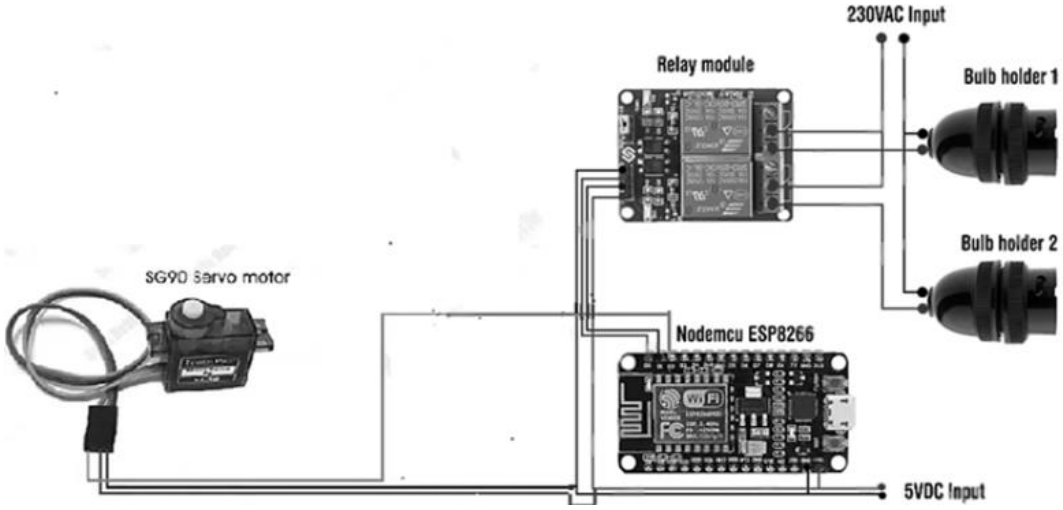


Figure 10.3: NodeMCU connection for home automation

b) Actual setup used in Laboratory

VIII Required Resources/apparatus/equipment with specifications:

Sr. No	Instruments/Components	Specification	Quantity
1	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1
2	Arduino IDE software	Latest Version: Arduino IDE 2.x (Current major release) Arduino IDE 1.8.x (Legacy, still widely used) Programming Language: Java (v1.x), TypeScript/Electron (v2.x) Operating Systems Supported: Windows 10 or later (64-bit), macOS 10.15 (Catalina) or later, Linux (64-bit Debian/Ubuntu-based distributions)	1
3	NodeMCU (ESP8266)	Tensilica L106 32-bit RISC processor Clock Speed: 80 MHz (default), up to 160 MHz Operating Voltage: 3.3V Input Voltage (VIN): 4.5V to 10V (via VIN pin) or 5V via micro-USB Digital I/O Pins: 11 (D0–D10) PWM Pins: All digital pins support PWM (10-bit resolution) Analog Input: 1 (10-bit ADC, 0–1V input range) Flash Memory: Typically, 4MB (varies by board) SRAM: ~50 KB available for application usage EEPROM: Emulated in flash	1
4	USB To Micro USB cable for connection	Cable length: 1 Meter connector type: USB to USB Male to Male	1
5	Relay Module	Supply voltage – 3.75V to 6V Quiescent current: 2mA Current when the relay is active: ~70mA Relay maximum contact voltage – 250VAC or 30VDC Relay maximum current – 10A	2
6	Servo motor	Operating Voltage is +5V typically Torque: 2.5kg/cm Operating speed is 0.1s/60° Gear Type: Plastic Rotation : 0°-180° Weight of motor : 9gm Package includes gear horns and screws	1
7	Connecting wires	Single strand wire with connectors	

IX Precautions to be followed (Safety instructions / Rules / Standards)

1. Follows all step of while installing Arduino IDE software.
2. Connect Arduino board to Wi-Fi and cloud properly.

X Procedure:

1. Set up Blynk IoT:

- Download and install the Blynk IoT app on your smartphone.
- Create a new device in the app and connect it to your Wi-Fi network.
- Create a new template for your device, which includes adding DataStream.
- Set up a web dashboard canvas for your device, including widgets like buttons, sliders, or displays.
- Create a device within the Blynk console, linking it to the template you just created.

2. Set up the Arduino IDE:

- Download and install the Arduino IDE.
- Install the ESP8266 core for Arduino IDE.
  - Go to File > Preferences (Arduino > Preferences for Mac).
  - Click on the "Additional Boards Manager URLs" button and add this link: http://arduino.esp8266.com/stable/package_esp8266com_index.json.
  - In the Arduino IDE open: Tools > Board > Boards Manager.
  - Search for esp8266 and install it.
- Install the Blynk library:
  - Download the latest Blynk library from GitHub.
  - Unpack it.
  - Move the libraries to your Arduino libraries folder (e.g., C:/User//Documents/Arduino/libraries).

3. Connect the ESP8266 to your Wi-Fi and Blynk:

- Open the Blynk example sketch for ESP8266 (File > Examples > Blynk > Boards_Ethernet > Arduino_Ethernet or similar).
- Replace the placeholder Wi-Fi credentials and Auth Token with your own.
- Upload the code to your ESP8266.

4. Connect Hardware properly as per circuit diagram

- Connect Relay for Light to D0
- Connect Relay for Light to D1
- Connect Servo motor for door locking and unlocking to D2

5. Interact with your device:

- **Set Up Blynk Cloud (HTTP REST API method):**
  a. Go to https://blynk.cloud and create/log in to your account
  b. Create a new Template
  c. Add the widgets as per your need.
     - Button widgets for Light (V1), Fan (V2), Lock (V3)
     - Get Auth Token from Blynk

6. Write and Upload Code

7. **View Data on Mobile App or Dashboard**
8. **Test System**
9. **Use Blynk app to turn devices ON/OFF**
10. **Observe feedback via serial monitor**

XI Program / Code

/*New Blynk app with Home Automation*/

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Servo.h>
char auth[] = "YourAuthToken";
char ssid[] = "YourSSID";
char pass[] = "YourPassword";

//Define the relay pins
#define relay1 D0
#define relay2 D1
Servo lockServo;

//Get the button values
BLYNK_WRITE(V1)
{
   bool value1 = param.asInt();
   // Check these values and turn the relay1 ON and OFF
 if (value1 = = 1)
 {
   digitalWrite(relay1, LOW);
   Serial.println("Light ON");
 }
else
{
    digitalWrite(relay1, HIGH);
    Serial.println("Light OFF");
 }
}
//Get the button values
BLYNK_WRITE(V2)
{
   bool value2 = param.asInt();
   // Check these values and turn the relay2 ON and OFF
   if (value2 = = 1)
 {
   digitalWrite(relay2, LOW);
```

```
    Serial.println("Fan ON");
  }
else
 {
    digitalWrite(relay2, HIGH);
    Serial.println("Fan OFF");
  }
}

BLYNK_WRITE(V3) {
 int value = param.asInt();
 // Lock control
 if (value = = 1)
 {
     lockServo.write(0);    // Lock
     Serial.println("Door Locked");
  }
else
 {
     lockServo.write(90); // Unlock
     Serial.println("Door Unlocked");
  }
}
void setup()
{
   Serial.begin(115200);
   //Set the relay pins as output pins
   pinMode(relay1, OUTPUT);
   pinMode(relay2, OUTPUT);

// Turn OFF the relay

   digitalWrite(relay1, HIGH);
   digitalWrite(relay2, HIGH);

   lockServo.attach(D2);
   Blynk.begin(auth, ssid, pass);
}
void loop()
{
//Run the Blynk library
  Blynk.run();
}
```

XII Output

(Attach Screen shots of Output generated and code in Sketch)

XIII Result(s):

……

……

……

……

XIV Interpretation of Results:

……

……

……

……

XV Conclusions and Recommendations:

……

……

……

……

XVI Practical Related Questions:

Note: Below given are few sample questions for reference. Teachers must design such questions to ensure achievement of identified CO.

1. Write and execute the code for ESP 8266 to implement smart home system such that:

a) lights should be ON/OFF depending on intensity of light
b) Fans should be ON/OFF depending on presence of person
c) Water pump should ON/OFF depend on water level in tank

2. State how to resolve security issues raised in smart home system?
3. List the challenges in developing and deploying IoT-based smart home application faced?
4. Expand the code to control multiple LEDs or other appliances using multiple relays and Blynk widgets.

[Space for Answers]

XVII References/Suggestions for further reading: include websites/links/Virtual lab Link

1 https://www.youtube.com/watch?v=JgaKrPK4H-M
2 https://srituhobby.com/how-to-make-a-home-automation-system-using-the-nodemcu- esp8266-board-and-the-new-blynk-app/?srsltid=AfmBOors4_fAg_eeoGXnJCvo9JjTnpIM8kNX_Astr7DzZmVv_WMZCe8S
3 https://www.youtube.com/watch?v=sOYK_kq8Jkg
4 https://srituhobby.com/servo-motor-control-using-esp8266-and-blynk-app-step-by-step-instructions/#:~:text=Step%201,by%20the%20SriTu%20Hobby%20team

XVIII Assessment Scheme

Performance Indicators		Weightage
Process Related (15 Marks)		**60%**
01	Use of IDE tools for programming	20%
02	Making connections of hardware	20%
03	Follow ethical practices.	20%
Product Related (10 Marks)		**40%**
05	Relevance of output of the problem definition	30%
06	Timely Submission of report	10%
Total (25 Marks)		**100 %**

Marks Obtained			Dated signature of Teacher
Process Related (15)	**Product Related (10)**	**Total(25)**	