EXP.No.4: LED BLINK

OBJECTIVE:

Interface LED with ESP8266 and control it through mobile using Blynk IOT app.

Materials required:

- ESP8266 NodeMCU board
- LED
- Breadboard
- Resistor (1K ohm)
- Jumper wires
- Usb cable
- Computer with Arduino IDE installed
- Blynk app installed on a smartphone or tablet

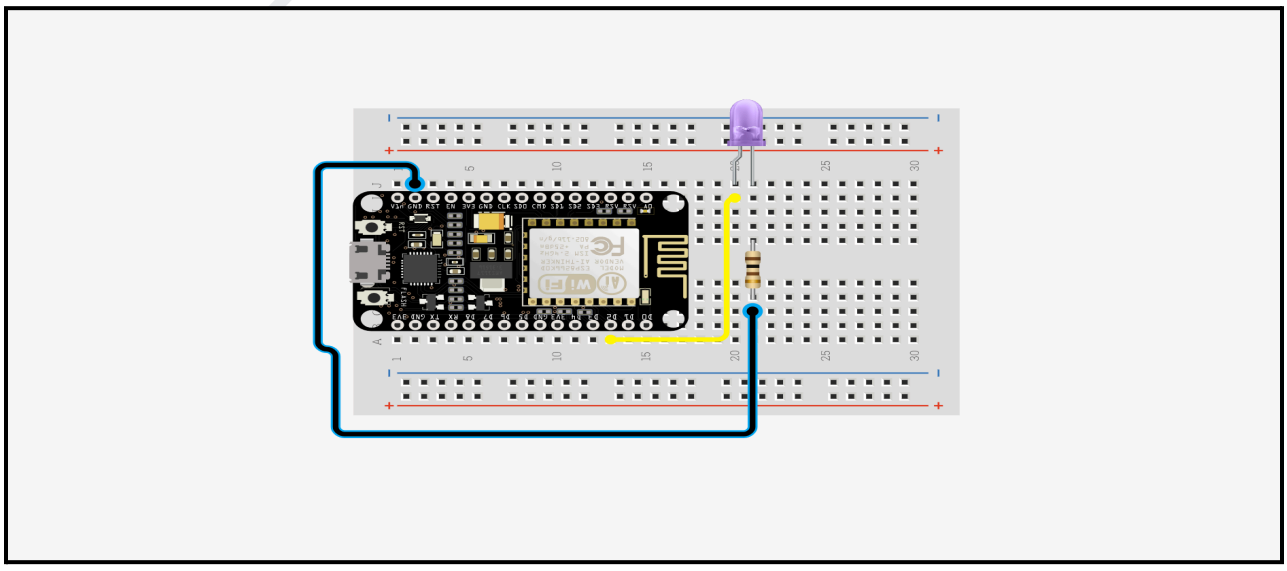
Theory:

The ESP8266 microcontroller, with its built-in Wi-Fi, is ideal for IoT projects. In this setup, the ESP8266 connects to a Wi-Fi network and communicates with the Blynk server, allowing an LED to be controlled remotely via the Blynk app on a smartphone.

When a user presses a button in the app, a command is sent to the ESP8266 to turn the LED on or off by toggling the GPIO pin connected to the LED.

This simple interaction demonstrates the core concept of IoT: remote control and monitoring of devices over the internet, making the ESP8266 and Blynk platform a powerful combination for such applications.

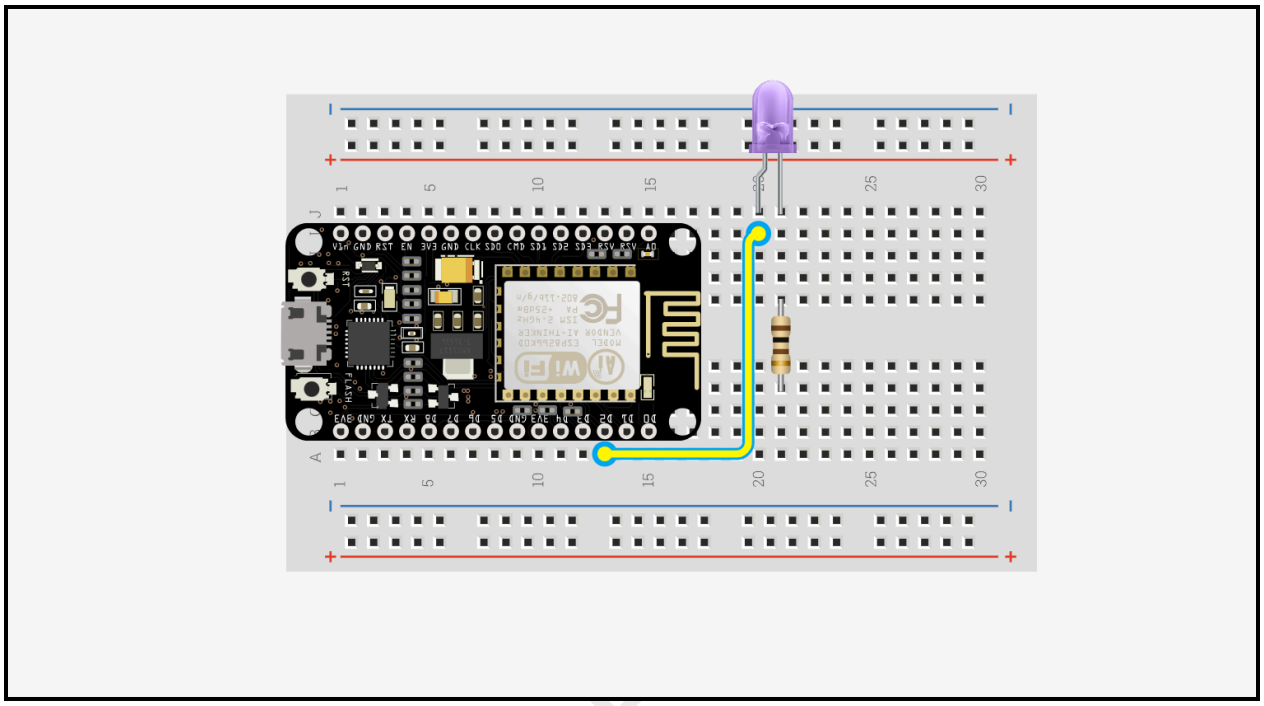
Circuit diagram:



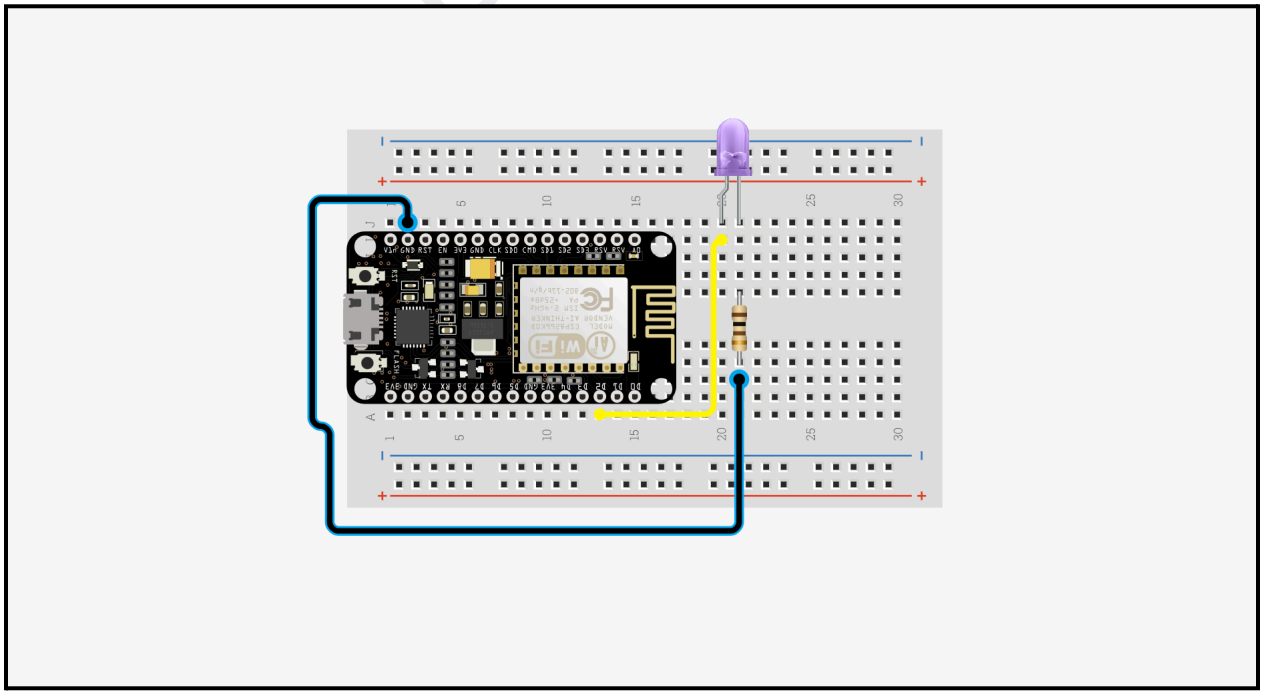
Procedure:

1. **Hardware Setup:**

- Connect the Data pin(GPIO2) of the ESP8266 to +ve of LED.



- Connect the GND pin of the esp8266 to -ve of LED through Resistor.



2. **Software Setup:**

- Open the Arduino IDE on your computer.
- Install the Blynk library by navigating to Sketch > Include Library > Manage Libraries and searching for “Blynk”.

3. **Blynk App Setup:**

- Open the Blynk app on your smartphone/tablet.
- Create a new project and note the Auth Token.
- Add Button and set them to Virtual Pin V0,respectively.

4. **Programming:**

- Connect the ESP8266 to your computer using a USB cable.
- In the Arduino IDE, write the following code:

Source code:

```
#define BLYNK_TEMPLATE_ID "---"
#define BLYNK_TEMPLATE_NAME "---"
#define BLYNK_AUTH_TOKEN "---"

#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char ssid[] = "wifi name";
char pass[] = "password";

void setup()
{
  // Debug console
  Serial.begin(9600);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  pinMode(D4, OUTPUT);
}
void loop()
{
  Blynk.run();
}
BLYNK_WRITE(V0)
{
  int pinValue = param.asInt(); // Get value as integer
```

```
 if (pinValue == 1)
 {
digitalWrite(D4, HIGH); // Turn LED on
 }
 else
 {
  digitalWrite(D4, LOW); // Turn LED off
 }
}
```

- Replace "YourAuthToken", "YourNetworkName", and "YourPassword"with the actual values.
- Upload the code to the ESP8266.

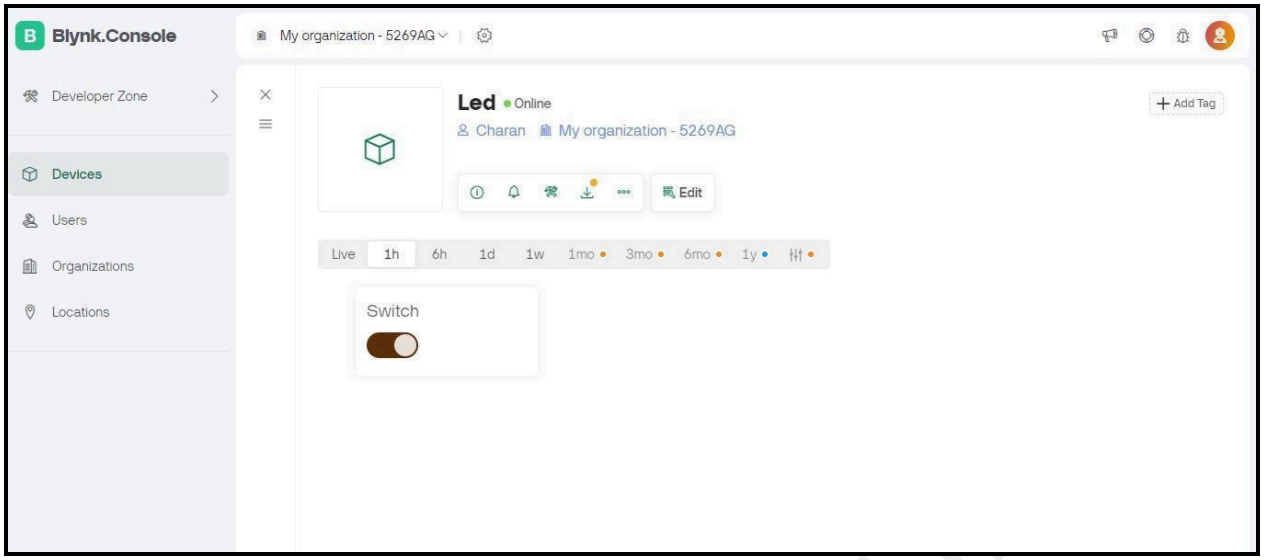
5. **Running the Experiment:**

- Open the Serial Monitor in the Arduino IDE to observe the temperature and humidity readings.
- Open the Blynk app to view the live temperature and humidity data on the Value Display widgets.

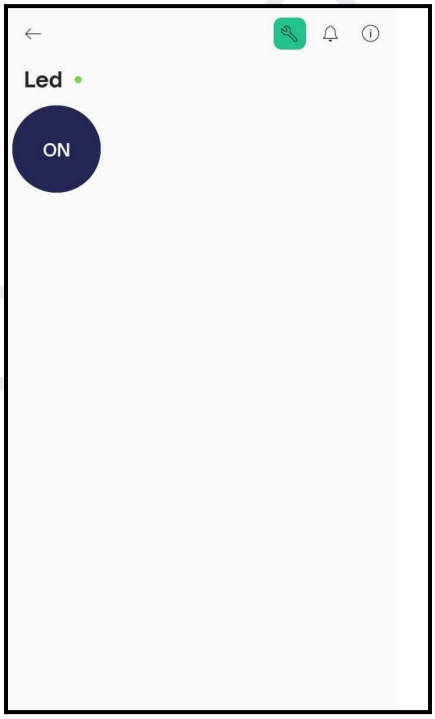
Observation:

S.No	Blynk website	Blynk IoT mobile app

Observation in Blynk website:

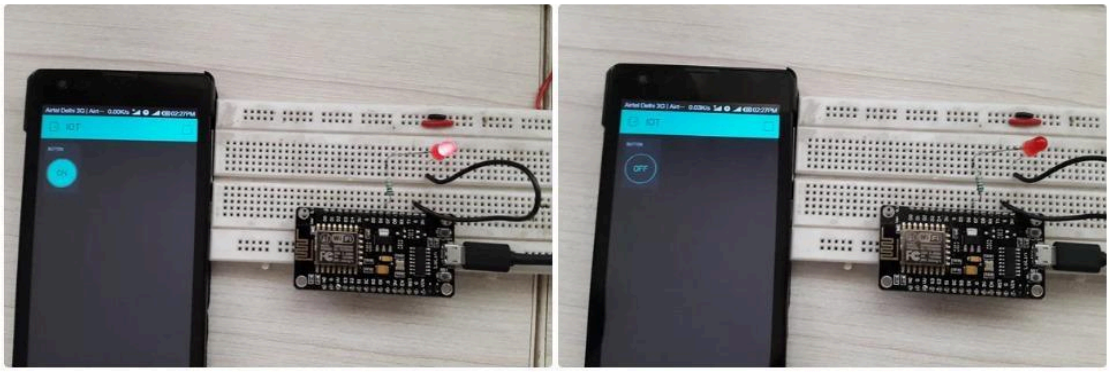


Observation in Blynk IoT mobile app:



Result:

Hence we observed LED Blinking with 8266 using with blynk IOT app.



Conclusion:

This experiment demonstrates how to interface the LED with the ESP8266 board and use the Blynk IoT platform to remotely monitor On and Off . The successful implementation confirms the practicality of using ESP8266 and Blynk for IoT applications.

Appendix:

A. Symbols and Abbreviations

- V_{CC}: Voltage Common Collector
- GND: Ground
- GPIO: General Purpose Input/Output

B. Tools Required

- ESP8266 NodeMCU board
- LED
- Breadboard
- Resistor (1K ohm)
- Jumper wires
- USB cable
- Computer with Arduino IDE installed
- Blynk app installed on a smartphone or tablet

B. Additional Resources

- ESP8266 Documentation
- Arduino IDE Installation Guide
- Blynk Documentation

D. Reference link with QR code

https://www.youtube.com/watch?v=Yc5zn2LQ3Mw

This format provides a clear and comprehensive guide for conducting the experiment, ensuring students can follow along and achieve the desired outcomes.