# SMART LIGHT SYSTEM

#### Group Members

1. Sanjai S - 126018042
2. Vishunu P Menon - 126018058
3. Nithyabalan M - 126003186

#### Introduction

The smart light system project integrates IoT and Bluetooth technology to create a more efficient and automated lighting system. The system allows users to control lights remotely via a mobile application built using Java. By receiving Bluetooth commands, the Arduino Uno manages the LED status, which is then logged into a database through an ESP8266 NodeMCU module for further analysis and tracking.

#### Objective

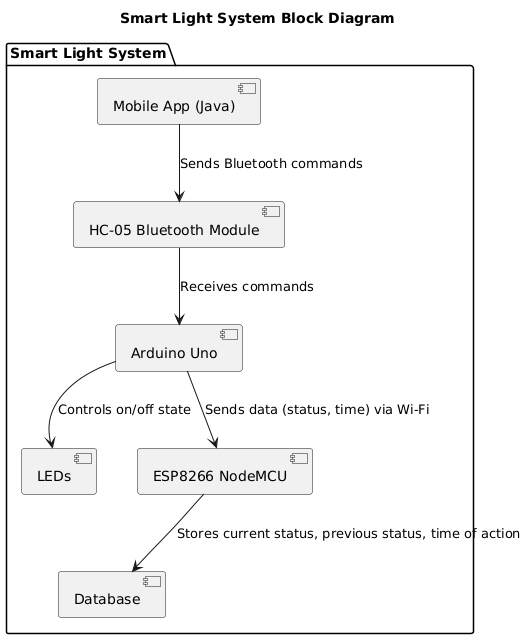
The objective of the project is to develop a user-friendly, low-cost smart lighting solution that can be controlled wirelessly using Bluetooth commands. Additionally, the system should be capable of storing and tracking the LED's current status, previous status, and the timestamp of actions taken. This will enable users to remotely monitor and manage their lighting systems efficiently.

#### Scope

This project is designed to improve the automation and management of home or industrial lighting systems. The Bluetooth-based control system allows for easy deployment in small-scale environments like homes or offices, while the integration of cloud storage through Wi-Fi enables potential scalability for larger environments.

### METHODS

#### BLOCK DIAGRAM



#### COMPONENTS SELECTION

**Arduino Uno**

* **Why**: Easy to program, versatile, and suitable for controlling LEDs and interfacing with communication modules.

**HC-05 Bluetooth Module**

* **Why**: Low-cost, reliable for Bluetooth communication between the mobile app and the Arduino.

**ESP8266 NodeMCU Wi-Fi Module**

* **Why**: Provides internet connectivity for storing data in a database, ideal for IoT projects.

**LEDs**

* **Why**: Simple, power-efficient indicators for light control in the system.

**Breadboard**

* **Why**: Flexible platform for prototyping without soldering.

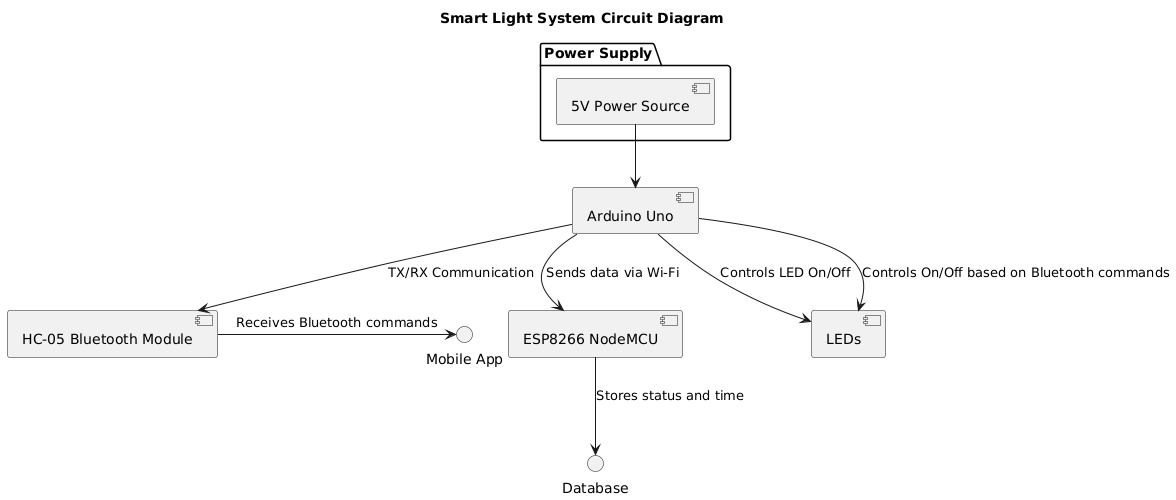
**Connecting Wires**

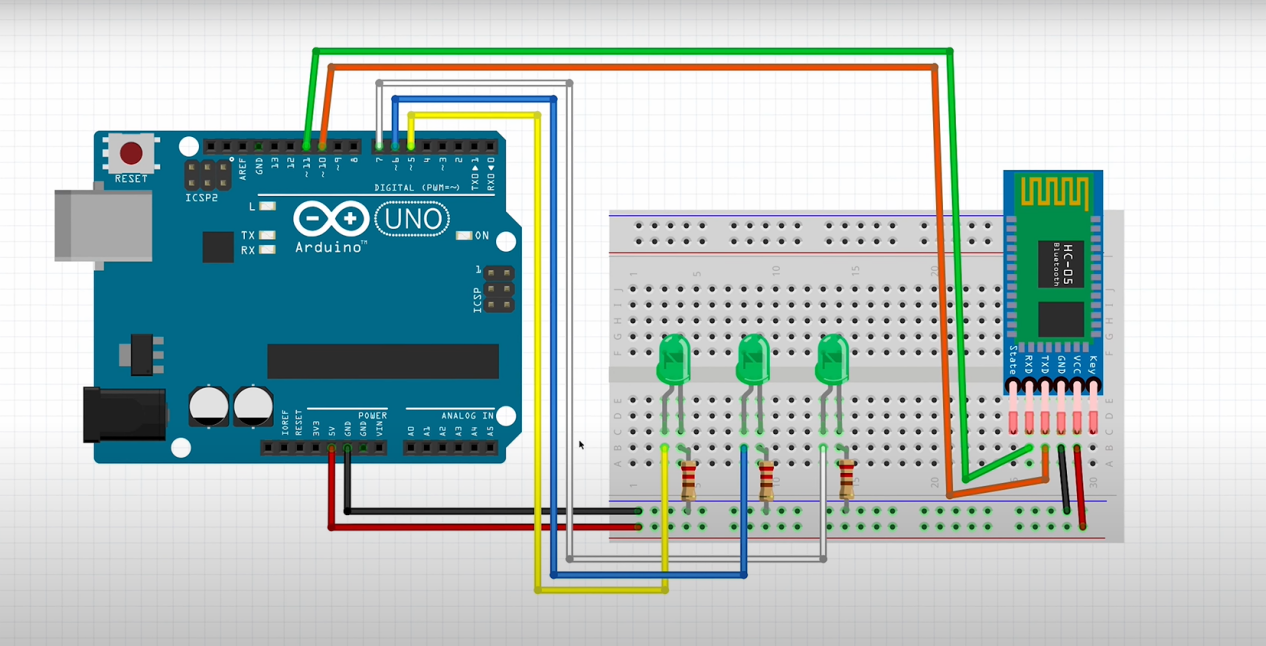
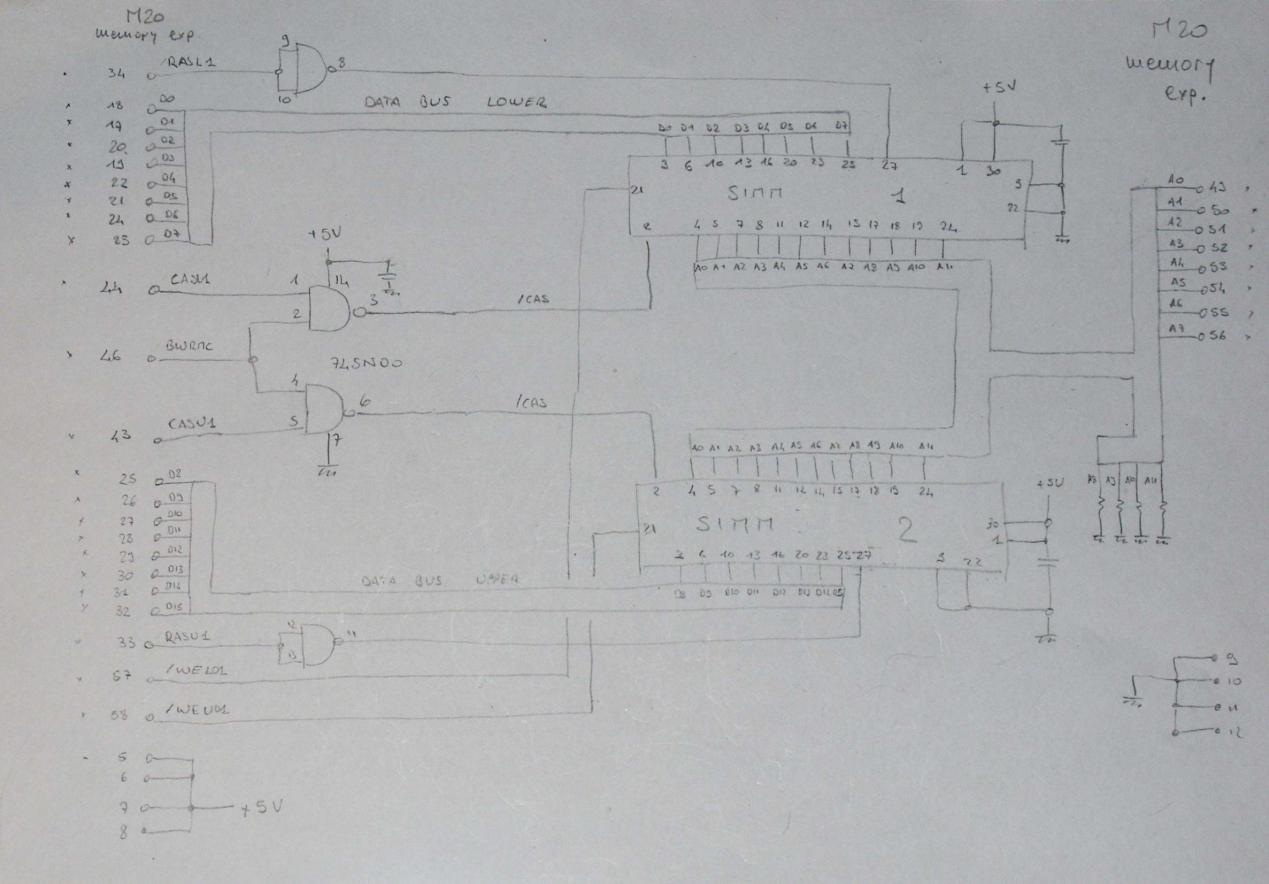
* **Why**: Essential for creating circuit connections between components.

**Power Supply**

* **Why**: Powers all the modules and the Arduino for system operation.

#### CIRCUIT DIAGRAM





#### ALGORITHM

· **Set up Wi-Fi Connection on NodeMCU**:

* Define the SSID and password for the Wi-Fi network.
* Attempt to connect to the Wi-Fi network.

· **Set up MySQL Database**:

* Create a table in the MySQL database to store the LED status and the corresponding timestamps.

· **Create a PHP Script**:

* Create a PHP script that receives data from the NodeMCU (via HTTP requests), connects to the MySQL database, and inserts the data into the database.

· **Control LEDs and Send Data to PHP**:

* Initialize the LED pins on the NodeMCU.
* Define the LED control logic based on conditions (e.g., turn the LED on or off).
* Capture the current status of each LED (ON/OFF).
* Send the data (LED ID, status) to the PHP server using HTTP GET requests.

· **Send HTTP Request to PHP**:

* Format the URL for the HTTP request, including the LED ID and status as query parameters.
* Send the HTTP GET request from the NodeMCU to the PHP script.
* Wait for a response from the server.

· **PHP Script Stores Data in MySQL**:

* The PHP script receives the request.
* It parses the query parameters (LED ID and status).
* Connect to the MySQL database.
* Insert the received data into the table.
* Return a success or error response.

· **Close Connection**:

* End the HTTP connection on the NodeMCU.
* Close the database connection in the PHP script.

#### PSEUDOCODE

##### FOR NodeMCU

BEGIN

// Wi-Fi credentials

SET ssid = "sanjai"

SET password = "12345678"

// Server URL

SET serverURL = “http://172.30.11.223:5173/save\_data.php"

// Define LED pins

SET led1 = pin 5

SET led2 = pin 6

SET led3 = pin 7

// Setup function

FUNCTION setup()

INITIALIZE serial communication at 115200 baud rate

CONNECT to Wi-Fi using ssid and password

WHILE Wi-Fi is not connected

WAIT and retry connection

END WHILE

INITIALIZE LED pins as output

END FUNCTION

// Loop function

FUNCTION loop()

IF Wi-Fi is connected

// Turn on LED1 and send status

TURN\_ON led1

CALL sendDataToServer(led\_id = 1, status = "ON")

WAIT for 5 seconds

// Turn off LED1 and send status

TURN\_OFF led1

CALL sendDataToServer(led\_id = 1, status = "OFF")

WAIT for 5 seconds

END IF

END FUNCTION

// Send data to the server

FUNCTION sendDataToServer(led\_id, status)

CREATE HTTP client

SET url = serverURL + "?led\_id=" + led\_id + "&status=" + status

SEND HTTP GET request to url

IF request is successful

PRINT server response

ELSE

PRINT error message

END IF

CLOSE HTTP client

END FUNCTION

END

##### FOR PHP SERVER

BEGIN

// Database connection credentials

SET servername = "localhost"

SET username = "root"

SET password = "Knight@1982"

SET dbname = "iot\_system"

// Create MySQL connection

FUNCTION connectToDatabase()

CONNECT to MySQL server using servername, username, password, and dbname

IF connection fails

RETURN "Connection failed" error

END IF

END FUNCTION

// Insert data into database

FUNCTION insertData(led\_id, status)

PREPARE SQL query to insert (led\_id, status) into led\_status table

EXECUTE SQL query

IF query is successful

PRINT "Record inserted successfully"

ELSE

PRINT "Error in inserting data"

END IF

CLOSE database connection

END FUNCTION

// Main function

FUNCTION main()

CALL connectToDatabase()

// Retrieve data from URL parameters

GET led\_id from HTTP request

GET status from HTTP request

// Insert data into the database

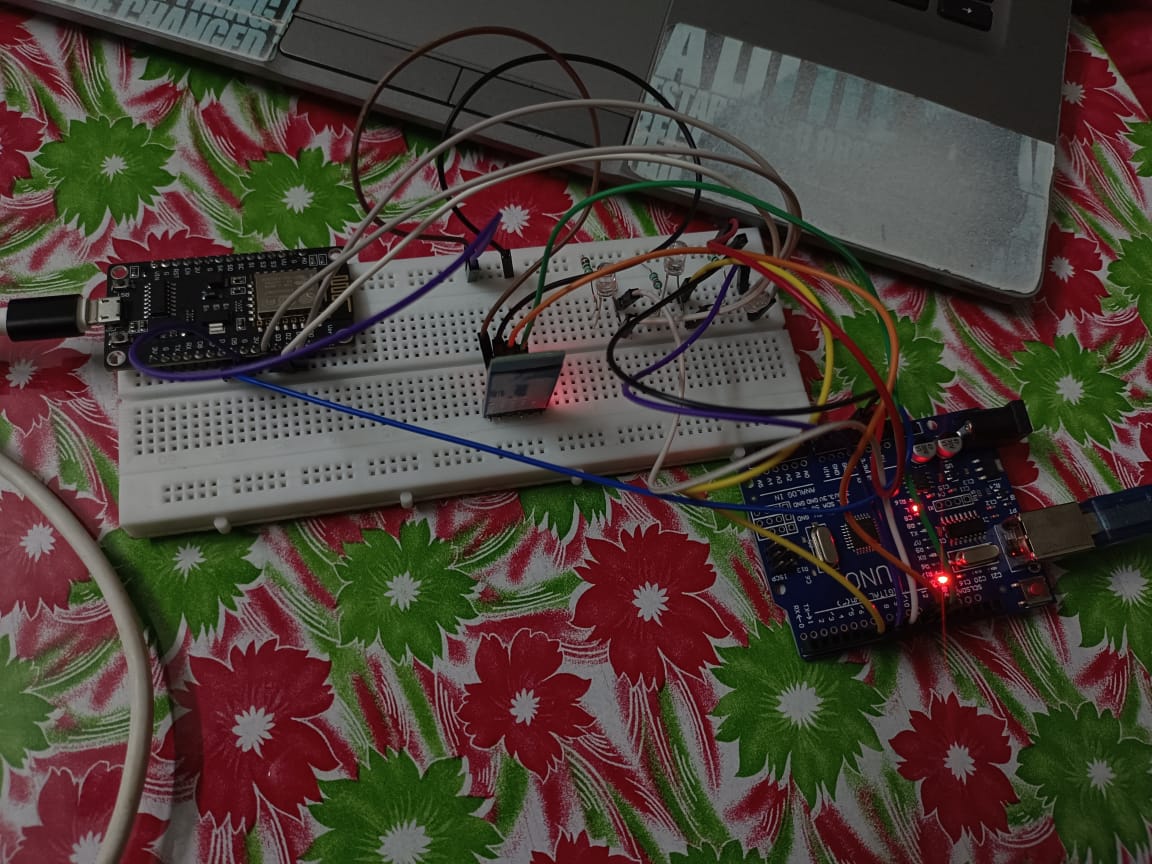
CALL insertData(led\_id, status)

END FUNCTION

END

## RESULTS

#### OVERALL DIAGRAM



#### CODE

#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>

const char\* ssid = "sanjai"; // Your WiFi SSID

const char\* password = "12345678"; // Your WiFi Password

const char\* server = "http://localhostL8080/save\_data.php"; // Your PHP script URL

// LED Pins

int led1 = 5;

int led2 = 6;

int led3 = 7;

void setup() {

// Initialize Serial Monitor

Serial.begin(115200);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

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

delay(1000);

Serial.println("Connecting to WiFi...");

}

Serial.println("Connected to WiFi");

// Initialize LED pins

pinMode(led1, OUTPUT);

pinMode(led2, OUTPUT);

pinMode(led3, OUTPUT);

// Initial LED states

digitalWrite(led1, LOW);

digitalWrite(led2, LOW);

digitalWrite(led3, LOW);

}

void loop() {

if (WiFi.status() == WL\_CONNECTED) {

HTTPClient http;

// Turn LED1 on and send data to PHP

digitalWrite(led1, HIGH);

sendDataToServer(1, "ON");

delay(5000); // Wait 5 seconds

// Turn LED1 off and send data to PHP

digitalWrite(led1, LOW);

sendDataToServer(1, "OFF");

delay(5000); // Wait 5 seconds

}

}

// Function to send data to the server

void sendDataToServer(int led\_id, const char\* status) {

HTTPClient http;

String url = String(server) + "?led\_id=" + String(led\_id) + "&status=" + String(status);

// Start connection and send HTTP GET request

http.begin(url);

int httpCode = http.GET();

// Check for the returning code

if (httpCode > 0) {

String payload = http.getString();

Serial.println(payload); // Print server response

} else {

Serial.println("Error in HTTP request");

}

http.end(); // Close connection

}

SAMPLE OUTPUTS In JSON FORMAT FROM THE DATABASE

{

      "id": "6654",

      "led": "led1",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:35:37 PM"

    },

    {

      "id": "a599",

      "led": "led1",

      "previousStatus": "on",

      "currentStatus": "off",

      "time": "10/24/2024, 3:35:38 PM"

    },

    {

      "id": "2735",

      "led": "led2",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:35:39 PM"

    },

    {

      "id": "4eb7",

      "led": "led3",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:35:40 PM"

    },

    {

      "id": "e395",

      "led": "led1",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:38:52 PM"

    },

    {

      "id": "75ff",

      "led": "led2",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:38:53 PM"

    },

    {

      "id": "9c08",

      "led": "led3",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:38:54 PM"

    },

    {

      "id": "99d8",

      "led": "led1",

      "previousStatus": "on",

      "currentStatus": "off",

      "time": "10/24/2024, 3:38:55 PM"

    },

    {

      "id": "100e",

      "led": "led2",

      "previousStatus": "on",

      "currentStatus": "off",

      "time": "10/24/2024, 3:38:56 PM"

    },

    {

      "id": "f3ee",

      "led": "led3",

      "previousStatus": "on",

      "currentStatus": "off",

      "time": "10/24/2024, 3:38:56 PM"

    },

    {

      "id": "bb73",

      "led": "led1",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:38:57 PM"

    },

    {

      "id": "273b",

      "led": "led2",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:39:01 PM"

    },

    {

      "id": "d275",

      "led": "led3",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:39:01 PM"

    },

    {

      "id": "31c9",

      "led": "led1",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:39:49 PM"

    },

    {

      "id": "6191",

      "led": "led2",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:39:50 PM"

    },

    {

      "id": "b9b2",

      "led": "led3",

      "previousStatus": "off",

      "currentStatus": "on",

      "time": "10/24/2024, 3:39:51 PM"

    },

    {

      "id": "12ae",

      "led": "led1",

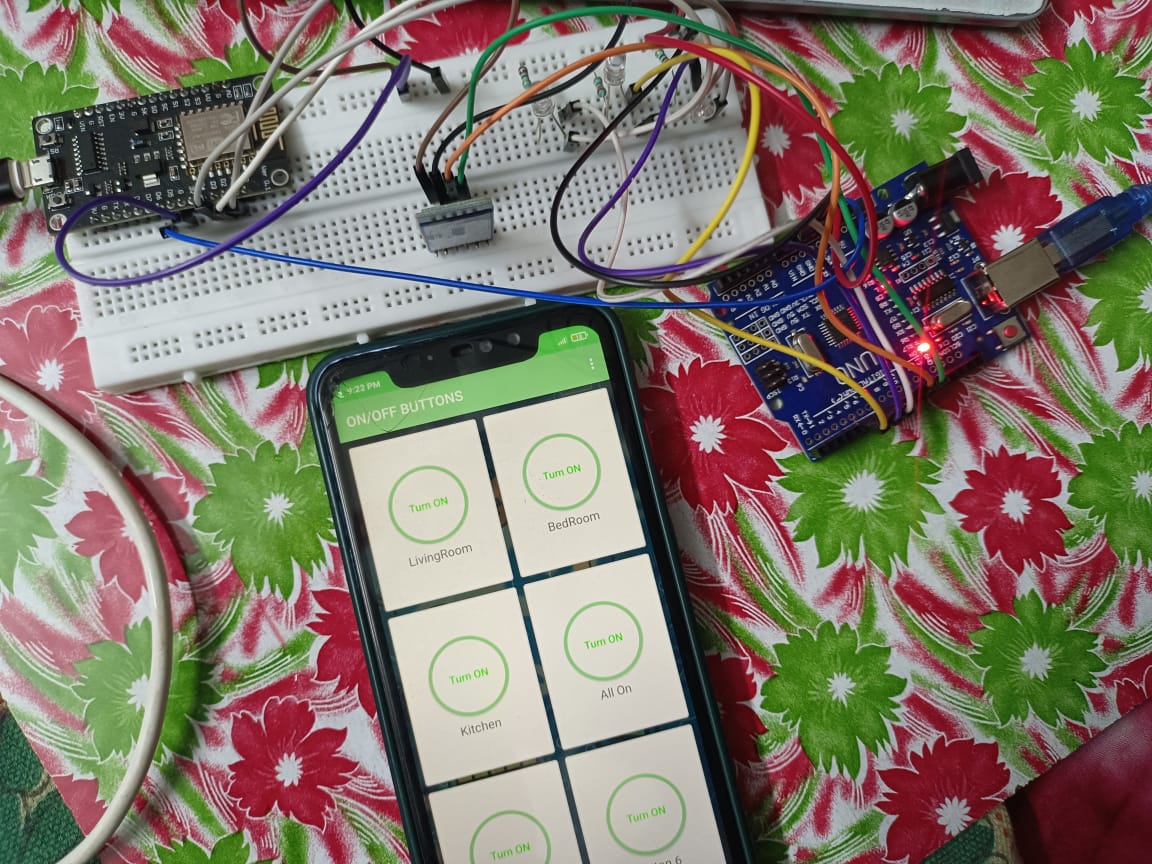
      "previousStatus": "off",

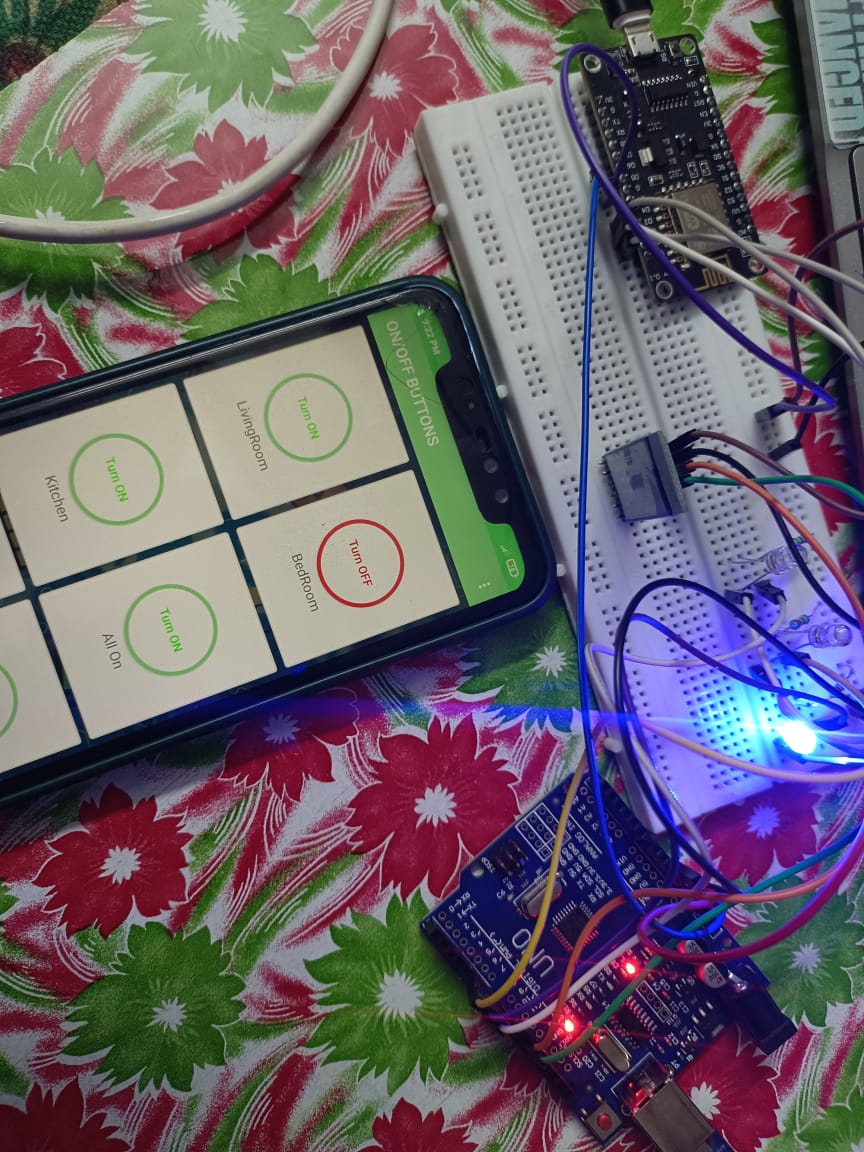
      "currentStatus": "on",

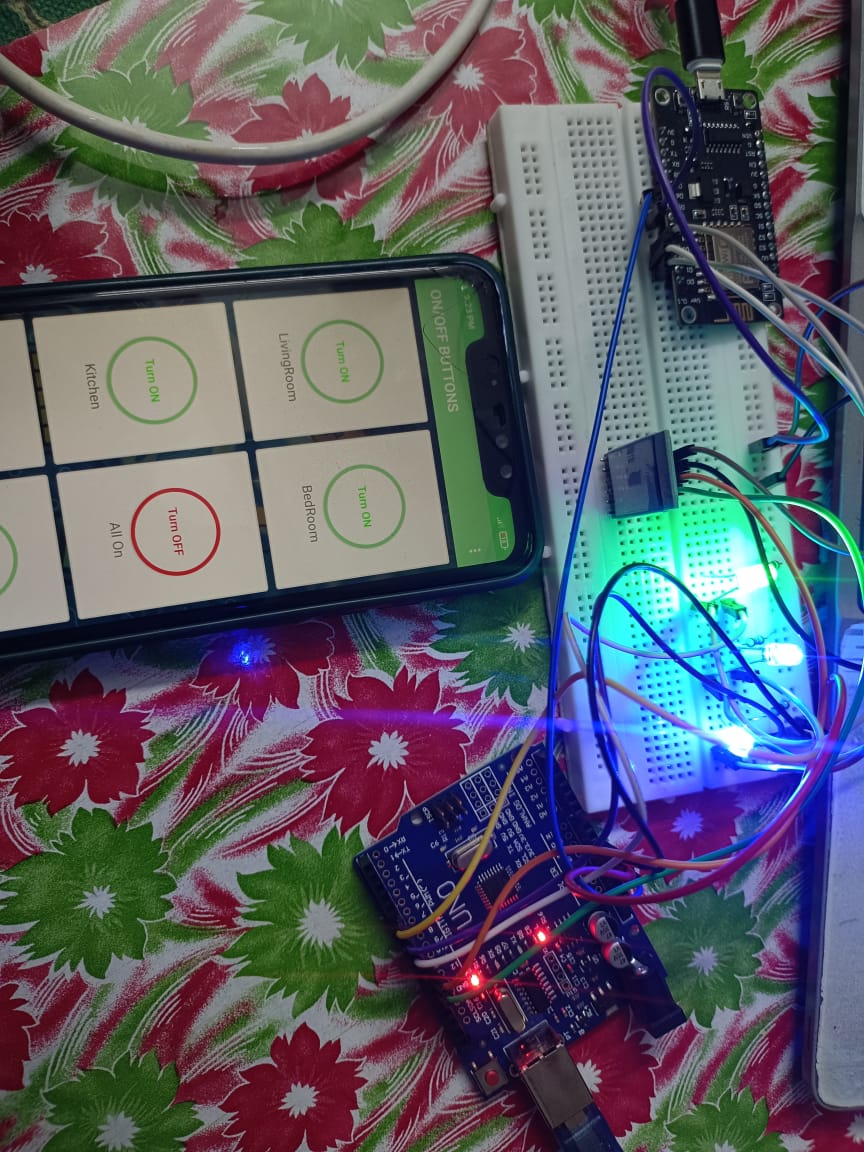
      "time": "10/24/2024, 3:42:41 PM"

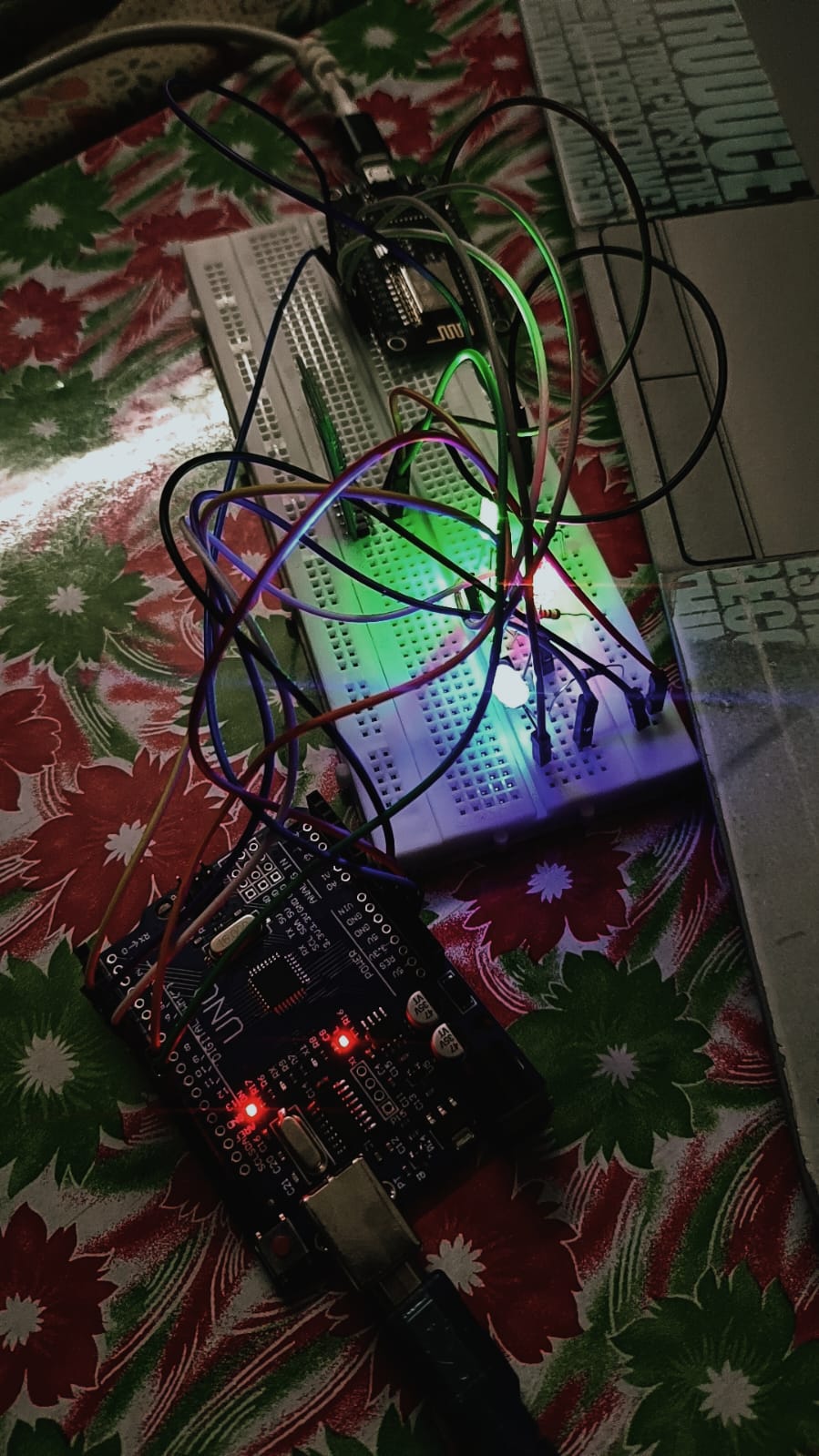
}

#### PICTURE DEPICTING THE WORK









#### Inference

Through the implementation of this smart light system, it was observed that wireless control and real-time monitoring of the lights provide significant convenience to users. The ability to store historical data such as LED status and action times allows for comprehensive tracking of the system's performance. The combination of Bluetooth commands and Wi-Fi connectivity proved to be effective for both local and remote operation.

#### Conclusion

The smart light system successfully achieved its goals of creating a flexible, wireless-controlled lighting solution with real-time data storage capabilities. It offers enhanced usability, efficiency, and the possibility of remote monitoring, making it a practical solution for various lighting applications.

#### Future Scope

* Integration with voice assistants like Alexa or Google Assistant for voice-controlled lighting.
* Expanding the system to control multiple lights and appliances, using advanced automation protocols.
* Implementing machine learning algorithms to predict usage patterns and automate lighting schedules.
* Enhancing security protocols, including encryption of data transmission between the app and the hardware.

### References

* Arduino Documentation: [https://www.arduino.cc/](https://www.arduino.cc/" \t "_new)
* ESP8266 NodeMCU Guide: [https://www.nodemcu.com/](https://www.nodemcu.com/" \t "_new)
* Bluetooth Module HC-05 Setup: [https://howtomechatronics.com/](https://howtomechatronics.com/" \t "_new)
* Firebase and IoT Projects: https://firebase.google.com/docs