Home Automation System

CS314: Wireless Sensor Networks Project Report

Rachit Singh Chahar
202151123

Rohit Kumar Varshney
202151135

Vatsal Srivastava
202151176

Yash Kumar Singh
202151181

***Abstract*—This project addresses the critical aspects of data transmission and system integrity in Wireless Sensor Networks (WSNs) by introducing a comprehensive home automation system using wireless technology. With components like NodeMCU, DHT temperature sensor, Relay Module and Servo Motor. The system employs multiple layers of security measures. It includes a temperature sensor for monitoring environmental conditions, and a fan which operates when temperature is low and a bulb which we can operate from blynk. Additionally, integration of a Telegram bot enables real-time notifications. Through the integration of these features, the system significantly enhances home automation, offering users a dependable way to remotely monitor and manage their homes, ensuring both safety and convenience.**

I. Introduction

The aim of the project is to develop an advanced home automation system using a regular Wi-Fi technology. In the realm of home automation, ensuring convenience and efficiency is crucial. This project emphasizes the creation of a reliable home automation system that utilizes wireless technology, NodeMCU, DHT temperature sensor, Relay Module, and Servo Motor. The system provides various features geared towards improving user interaction.

The control device for automation in the project is an NodeMCU. Data sent from the end device via WiFi is received by the WiFi module connected to NodeMCU . The NodeMCU reads the data and determines the switching behaviour of electrical devices connected through relays.

The primary highlight is the Temperature Alert System, made possible by integrating NodeMCU with a DHT11 temperature sensor. This system enables continuous monitoring of indoor temperature levels. If the temperature surpasses a predetermined limit, the setup activates the Servo Motor, functioning as a fan.

It's crucial to emphasize that while this project lays down a solid groundwork, there's immense potential for customization. Users have the flexibility to adapt the system according to their unique preferences and needs, effectively turning a basic home automation setup into a sophisticated smart home.

II. Components Used

A. Nodemcu ESP8266:

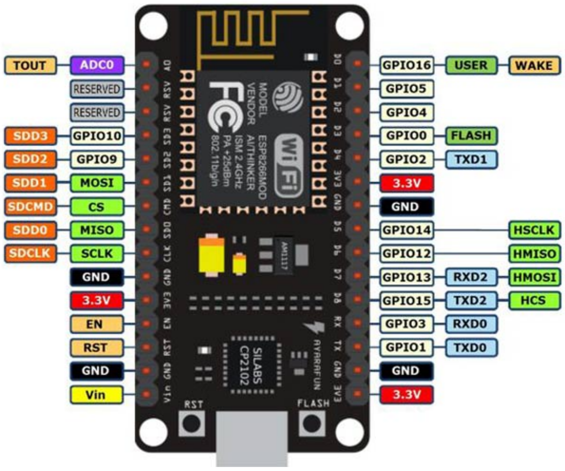


Fig. 1. NodeMCU ESP8266

The NodeMCU ESP8266 is an affordable Wi-Fi development board built around the ESP8266 chip. It is equipped with a Tensilica L106 32-bit microcontroller running at 80MHz and has 4MB of flash memory for storing programs. This board supports 802.11 b/g/n Wi-Fi connectivity and can be programmed using the Arduino IDE with ease. With multiple GPIO pins, it allows for various functionalities like digital input/output, analog input, and PWM. Widely utilized in IoT projects, its popularity stems from its low cost, integrated Wi-Fi capabilities, and user-friendly interface. It can be powered through USB or an external power source and communicates with other devices via Wi-Fi or serial communication.

B. DHT11 Sensor:

The DHT11 sensor serves as a fundamental digital sensor for measuring both temperature and humidity. Found in various applications such as weather stations, climate control systems, and home automation projects, it plays a crucial role in monitoring ambient conditions. Additionally, its simple design and affordability make it a popular choice for hobbyists and professionals alike seeking to incorporate temperature and humidity sensing into their projects.

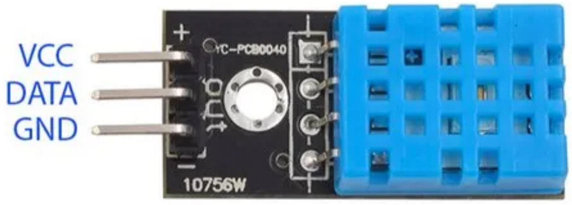


Fig. 2. DHT11 Sensor

C. Relay Module:

A relay module functions as an electronic switch, facilitating the control of high-voltage circuits with low-voltage signals, often originating from microcontrollers. Typically comprising a relay

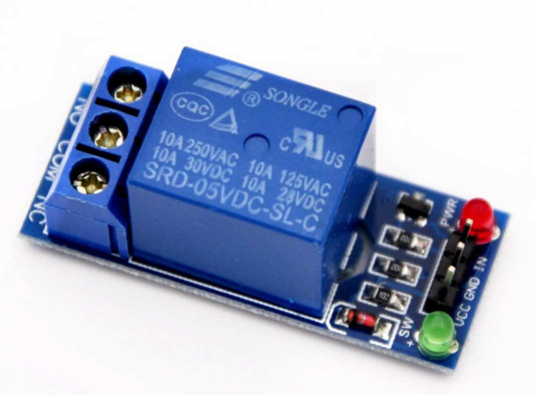


Fig. 3. Relay Module

(an electromagnetic switch) along with supporting components like transistors and diodes arranged on a printed circuit board (PCB), it enables the management of high-power devices such as lights and motors through a low-power input like that from an Arduino. This module provides isolation between the control and load circuits, thereby enhancing safety measures. Available in various configurations, including single or multiple relays on a single board, these modules are versatile in switching different voltage and current levels based on the relay's specifications. Commonly operating with 5V or 12V DC control signals, relay modules find extensive applications in domains such as home automation, industrial automation, and robotics due to their efficient control over high-power devices.

D. Breadboard:

A breadboard is a plastic board, typically rectangular, containing a grid of holes organized in rows and columns. These holes are internally connected in a specific arrangement, enabling components to be inserted and interconnected without the need for soldering. Widely utilized in electronics, breadboards facilitate the rapid assembly and testing of circuits before their finalization on a printed circuit board (PCB). Their reusability and flexibility allow for easy modifications to circuits by simply rearranging or replacing components. Available in various sizes, larger breadboards offer more space for intricate circuits. They serve as indispensable tools for both novice and seasoned electronics enthusiasts, providing a user-friendly means to prototype circuits without requiring specialized equipment.

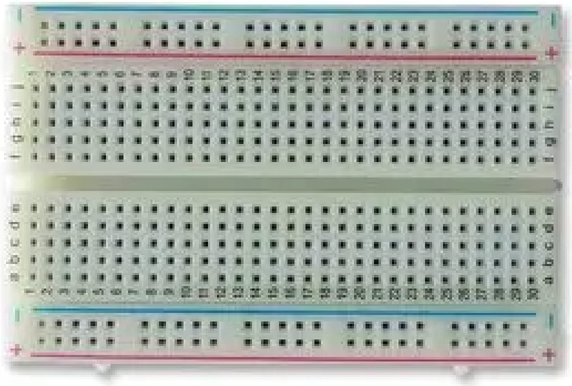


Fig. 4. Breadboard

E. Servo Motor:

A servo motor is a type of rotary actuator or motor that allows for precise control of angular position, acceleration, and velocity. It consists of a motor coupled with a feedback system that continuously monitors and adjusts the position of the motor shaft. Servo motors are widely used in various applications where precise control of angular position is required, such as robotics, RC vehicles, industrial automation, and aerospace.

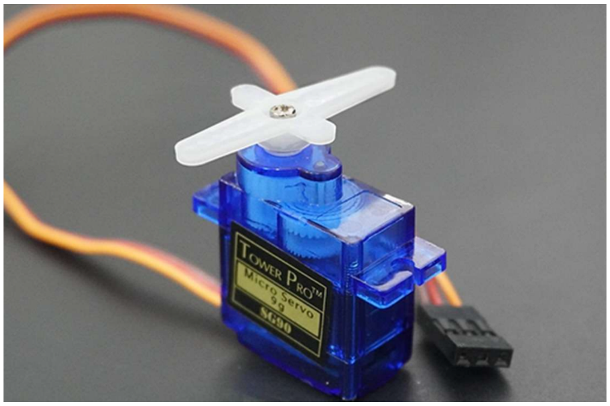


Fig. 5. Servo Moter

F. Blynk:

Blynk is a platform that allows developers to build IoT (Internet of Things) applications for controlling hardware remotely using smartphones or tablets. It provides a drag-and-drop interface for creating customizable mobile apps that can interact with a variety of microcontrollers, such as Arduino, Raspberry Pi, ESP8266, and others.

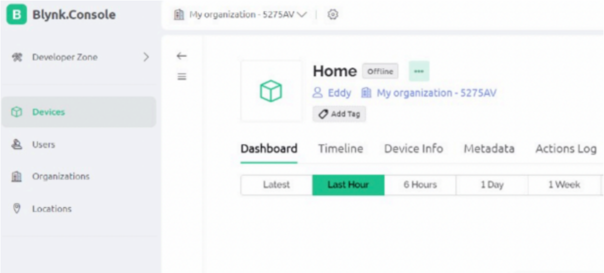
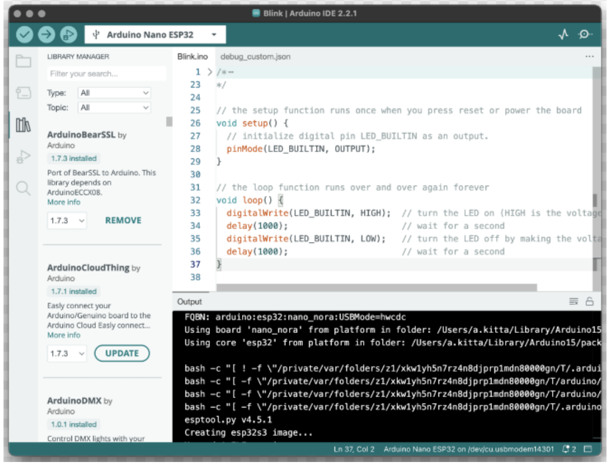


Fig. 6. Blynk Interface

G. Arduino IDE:

The Arduino Integrated Development Environment (IDE) is an open-source software application used to program and upload code to Arduino and compatible microcontroller boards. It provides a user-friendly interface for writing, compiling, and uploading code to the board, making it accessible to beginners and experienced developers alike.

Overall, the Arduino IDE provides a comprehensive development environment for writing, compiling, and uploading code to Arduino and compatible microcontroller boards, making it an essential tool for electronics enthusiasts, hobbyists, and professionals alike.


Blink | Arduino IDE 2.2.1
Arduino Nano ESP32
LIBRARY MANAGER
Filter your search...
Type: All
Topic: All
ArduinoBearSSL by Arduino
1.7.3 installed
Port of BearSSL to Arduino. This library depends on ArduinoECCX08.
More info
1.7.3
REMOVE
ArduinoCloudThing by Arduino
1.7.1 installed
Easly connect your Arduino/Genuino board to the Arduino Cloud Easly connect...
More info
1.7.3
UPDATE
ArduinoDMX by Arduino
1.0.1 installed
Control DMX lights with your
Blink.ino
debug_custom.json
Output
Ln 37, Col 2 Arduino Nano ESP32 on /dev/cu.usbmodem14301 2

```
/*…
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off by making the volta
  delay(1000);                      // wait for a second
}

```

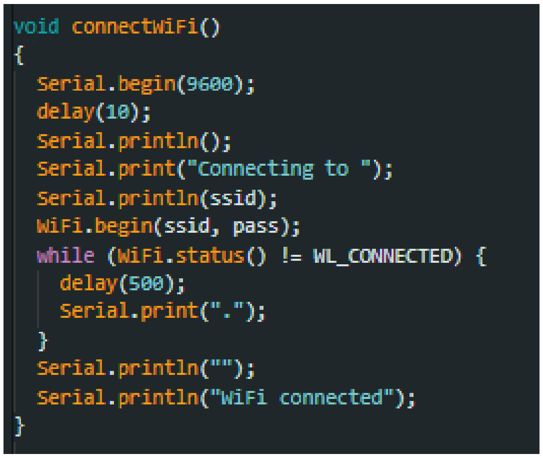
```
FQBN: arduino:esp32:nano_nora:USBMode=hwcdc
Using board 'nano_nora' from platform in folder: /Users/a.kitta/Library/Arduino15
Using core 'esp32' from platform in folder: /Users/a.kitta/Library/Arduino15/pack

bash -c "[ ! -f \"/private/var/folders/z1/xkw1yh5n7rz4n8djprp1mdn80000gn/T/.ardui
bash -c "[ -f \"/private/var/folders/z1/xkw1yh5n7rz4n8djprp1mdn80000gn/T/arduino/
bash -c "[ -f \"/private/var/folders/z1/xkw1yh5n7rz4n8djprp1mdn80000gn/T/arduino/
bash -c "[ -f \"/private/var/folders/z1/xkw1yh5n7rz4n8djprp1mdn80000gn/T/.arduino
esptool.py v4.5.1
Creating esp32s3 image...
```

Fig. 7. Arduino IDE

III. Hardware Model & Implementation

A. WiFi Connection:

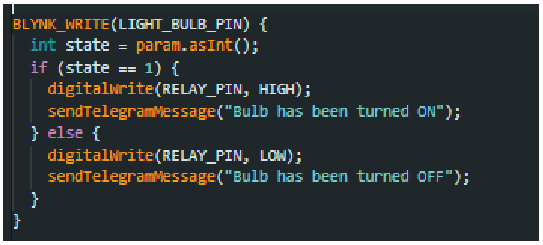
This algorithm initiates a connection to a specified WiFi network and manages re-connection attempts in case the initial connection is unsuccessful. It persists in trying to establish a connection until it successfully connects, ensuring reliable network connectivity management.



```
void connectWiFi()
{
  Serial.begin(9600);
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
}
```

Fig. 8. Code for WiFi Connection

B. Bulb Control using Blynk:



```
BLYNK_WRITE(LIGHT_BULB_PIN) {
  int state = param.asInt();
  if (state == 1) {
    digitalWrite(RELAY_PIN, HIGH);
    sendTelegramMessage("Bulb has been turned ON");
  } else {
    digitalWrite(RELAY_PIN, LOW);
    sendTelegramMessage("Bulb has been turned OFF");
  }
}
```

Fig. 9. Code for Bulb Control

We have connected a bulb using Relay Module which operates on AC current and this we are operating from our Blynk Interface which we can control from anywhere.

The Blynk Interface is used for monitoring the temperature and humidity and can also control bulb.

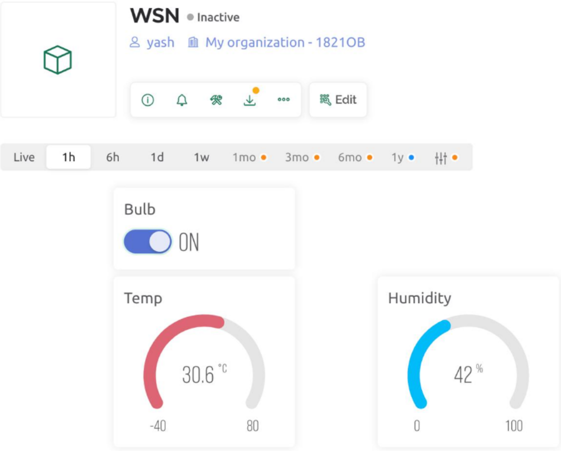
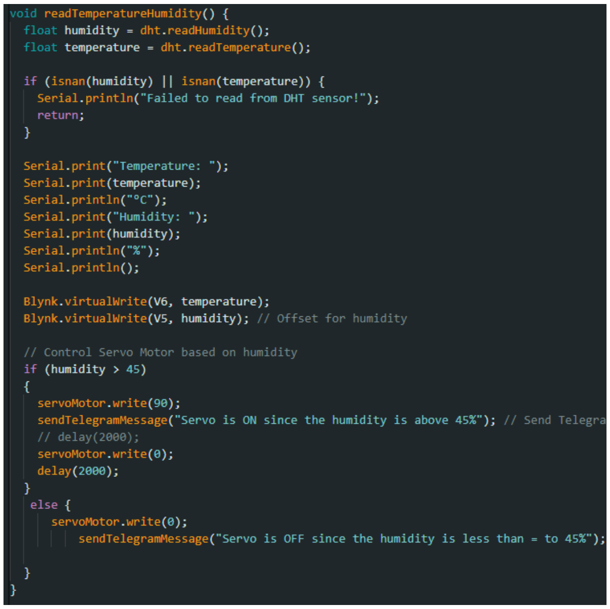


Fig. 10. Blynk Interface

C. Temperature & Humidity Monitoring:

The system continuously reads temperature & humidity from a DHT11 sensor. If the temperature or humidity exceeds a preset threshold, it triggers the servo motor which is acting as fan and it starts working and stops when the reading gets normal.

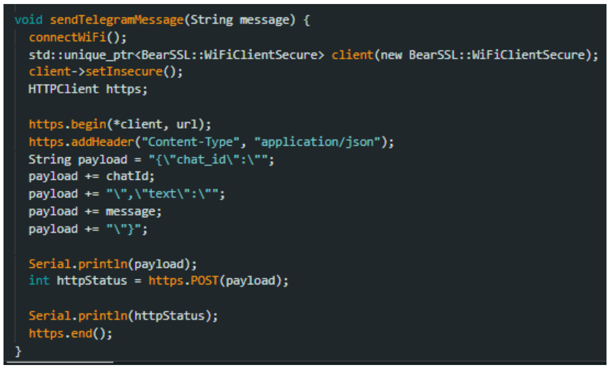


```
void readTemperatureHumidity() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature();

  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println("°C");
  Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println("%");
  Serial.println();

  Blynk.virtualWrite(V6, temperature);
  Blynk.virtualWrite(V5, humidity); // Offset for humidity

  // Control Servo Motor based on humidity
  if (humidity > 45)
  {
    servoMotor.write(90);
    sendTelegramMessage("Servo is ON since the humidity is above 45%"); // Send Telegra
    // delay(2000);
    servoMotor.write(0);
    delay(2000);
  }
   else {
      servoMotor.write(0);
          sendTelegramMessage("Servo is OFF since the humidity is less than = to 45%");

  }
}
```

Fig. 11. Code for Temperature & Humidity

D. Telegram Notifications:

It sends custom notifications via the Telegram API when certain events occur, when we turn on the bulb or turn off the bulb, also when the servo motor starts working and stops working. This feature enhances the remote monitoring capabilities of the system.



```
void sendTelegramMessage(String message) {
  connectWiFi();
  std::unique_ptr<BearSSL::WiFiClientSecure> client(new BearSSL::WiFiClientSecure);
  client->setInsecure();
  HTTPClient https;

  https.begin(*client, url);
  https.addHeader("Content-Type", "application/json");
  String payload = "{\"chat_id\":\"";
  payload += chatId;
  payload += "\",\"text\":\"";
  payload += message;
  payload += "\"}";

  Serial.println(payload);
  int httpStatus = https.POST(payload);

  Serial.println(httpStatus);
  https.end();
}
```

Fig. 12. Code for Telegram Connection

E. Circuit

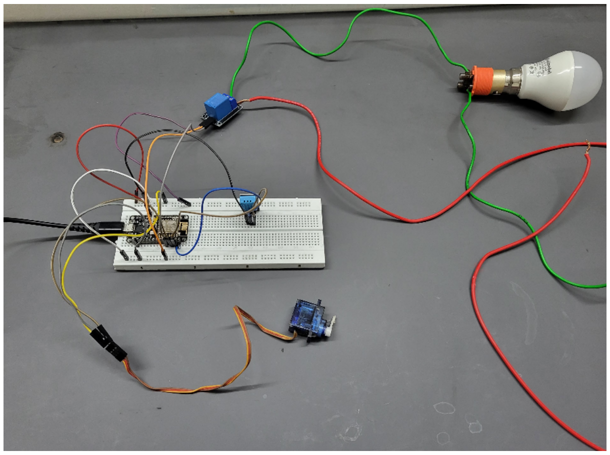


Fig. 13. Hardware Model

IV. Conclusion

In conclusion, our home automation project, seamlessly integrating NodeMCU ESP8266 and WiFi technologies, represents a significant stride in the realm of smart living. Informed by insights from researchers such as Hussein and Asadullah, our system offers a practical solution, utilizing NodeMCU ESP8266 for connectivity and Arduino for flexibility. This amalgamation allows for real-time communication, empowering users with remote control and monitoring capabilities. The project not only contributes to the existing knowledge on home automation but also underscores the efficacy of NodeMCU ESP8266 in enhancing WiFi-based applications. As we address challenges, the system stands as a robust foundation for future improvements in security, energy efficiency, and potential integration with broader IoT initiatives. Our project not only mirrors the current state of technology but also points to the exciting possibilities ahead, showcasing collaborative efforts propelling advancements in smart living.