

Smart Home energy management using esp32

# 1. Abstract

## 1.Overview of the project objectives and outcomes:

This project aims to develop a Smart Home Energy Management System using an ESP32 and a temperature sensor. The objective is to monitor and control home appliances' energy usage, enhancing convenience and efficiency. The system will automatically turn appliances on or off based on temperature readings, providing a practical solution for energy conservation.

# 2. Introduction

## 1.Background and motivation for the project:

With the increasing demand for energy efficiency and smart home solutions, there is a need for an automated system that can manage home energy usage effectively. This project leverages IoT technology to create a cost-effective and user-friendly solution.

## 2.Importance of smart home energy management:

Efficient energy management reduces electricity bills and minimizes environmental impact. Implementing smart energy solutions helps households optimize energy consumption and contribute to sustainability.

# 3. Methodology

## 1.Detailed explanation of the methods and processes used in the project:

The project uses an ESP32 microcontroller to collect data from a DHT11 temperature sensor. Based on predefined temperature thresholds, the ESP32 controls the relay module connected to home appliances, automating their operation. The system also includes data transmission to a cloud platform for remote monitoring.

# 4. Existing Work

## 1.Review of current energy management systems:

Current energy management systems often require manual intervention or expensive proprietary solutions. They may lack real-time monitoring and control capabilities.

## 2.Comparison with the proposed system:

The proposed system is cost-effective, easy to implement, and offers real-time monitoring and control through IoT technology, making it more efficient and user-friendly.

# 5. Proposed Work

## 1.Description of the proposed IoT Smart Home Energy Management System:

The system consists of an ESP32 microcontroller, a DHT11 temperature sensor, and a relay module. It automates home appliances' operation based on real-time temperature data, providing an efficient energy management solution.

## 2.Innovations and improvements over existing systems:

The proposed system offers real-time data monitoring, remote control capabilities, and automated operation, which are significant improvements over traditional energy management systems.

# 6. System Requirements

## 1.Hardware Components:

***- ESP32****: Microcontroller with WiFi capabilities.*

***- DHT11****: Temperature sensor.*

***- Relay Module****: For controlling home appliances.*

***- Home Appliances****: Devices like lights and air conditioner.*

*-* ***Connecting Wires and Breadboard****: For circuit connections.*

## 2.Software Components:

***- Arduino IDE****: For programming the ESP32.*

***- Libraries****: DHT library for temperature sensor, WiFi library for network connectivity.*

# 7. Implementation Details

Step-by-step guide on how the system was implemented:

1. **Assemble the hardware:** Connect the DHT11 sensor and relay module to the ESP32 as per the circuit diagram.

2. **Configure the software:** Install the necessary libraries in the Arduino IDE and write the program code.

3. **Upload the code:** Upload the program code to the ESP32.

4. **Test the system:** Verify the system's functionality by observing the relay module's response to temperature changes.

## Configuration and setup of hardware and software:

- Hardware setup involves wiring the components correctly on a breadboard.

- Software setup involves configuring the Arduino IDE with the necessary libraries and uploading the code to the ESP32.

# 8. Data Collection and Processing

## 1.Methods used to collect and process data from sensors:

The DHT11 sensor collects temperature data, which is processed by the ESP32 to make decisions about controlling the relay module.

## 2.Data transmission to blynk:

The ESP32 can be programmed to send temperature data to the blynk cloud platform for remote monitoring.

# 9. Communication Protocols

## 1.Explanation of the communication protocols used:

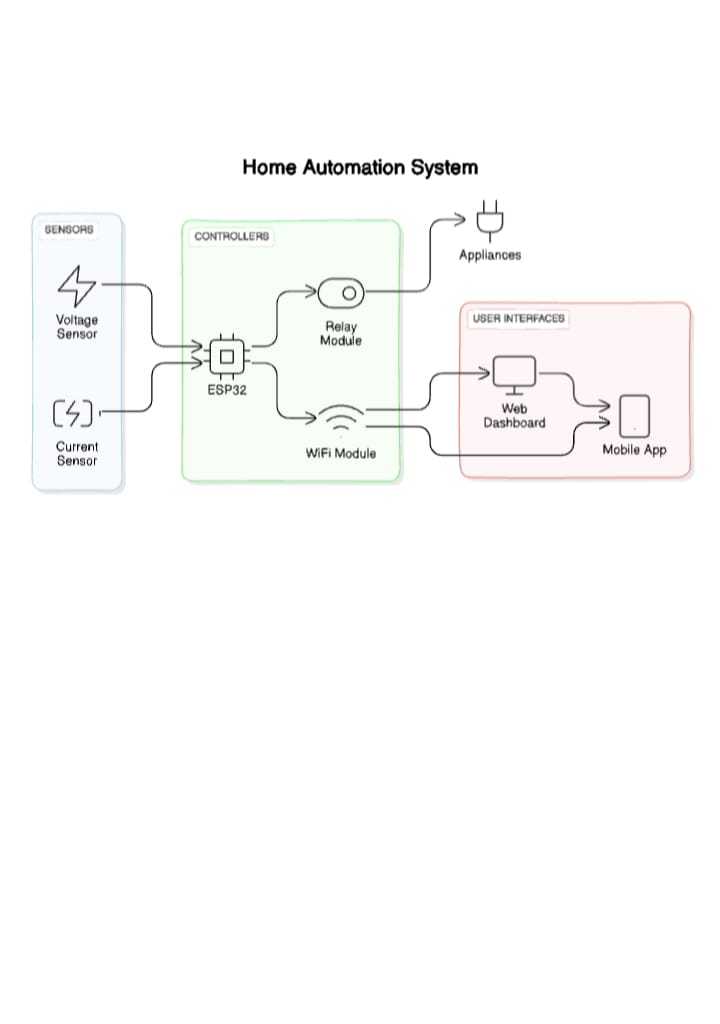
- **WiFi:** Used by the ESP32 to connect to the internet.

- **I2C**: Communication protocol between sensors and the microcontroller.

- **blynk API:** For transmitting data to the cloud platform.(AUTH TOKENS:XXX)

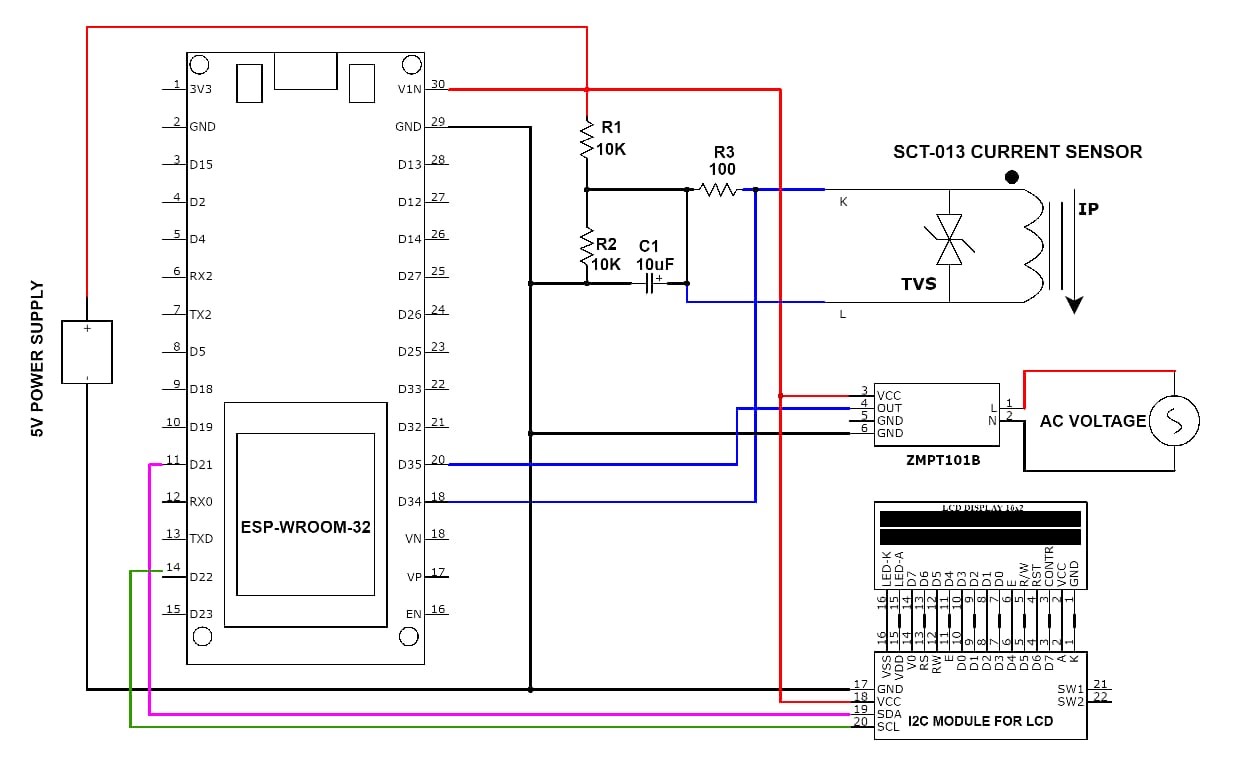
# 10. Block Diagram

## 1.Visual representation of the system architecture:

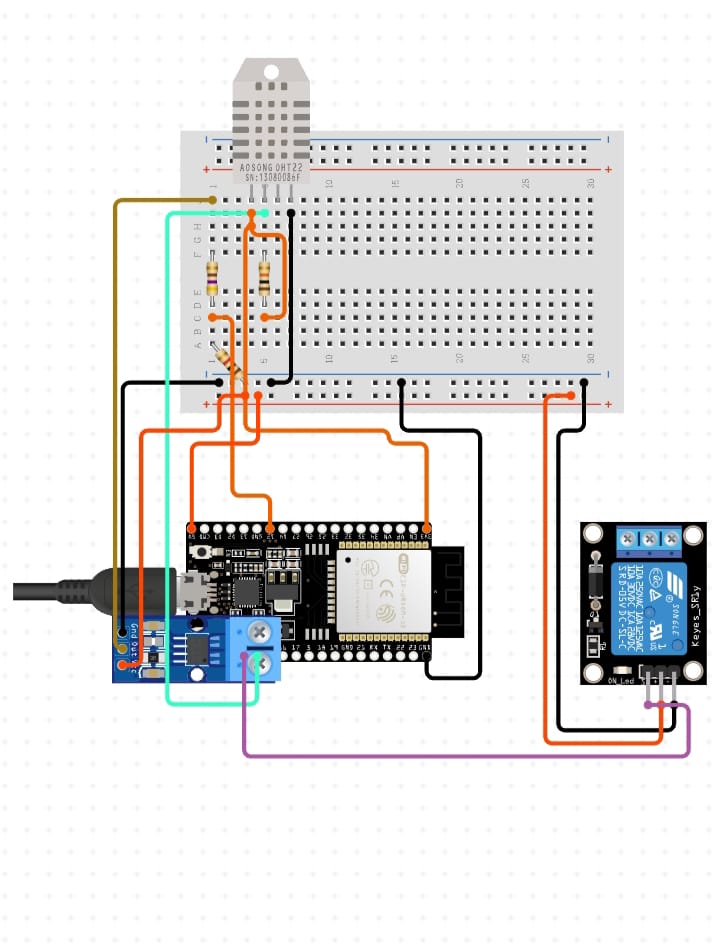


# 11. Circuit Diagram

## 1.Schematic diagram of the circuit connections:



11.1 2. **Simulation diagram**



# 12. Program Code (C++)

## 1.Complete source code of the project: using cpp(c++)

|  |
| --- |
| #include <WiFi.h>  #include "DHT.h"  #define DHTPIN 4  #define DHTTYPE DHT11  #define RELAY\_PIN 2  const char\* ssid = "your\_SSID";  const char\* password = "your\_PASSWORD";  DHT dht(DHTPIN, DHTTYPE);  void setup() {  Serial.begin(115200);  pinMode(RELAY\_PIN, OUTPUT);  digitalWrite(RELAY\_PIN, LOW);  dht.begin();  WiFi.begin(ssid, password);  while (WiFi.status() != WL\_CONNECTED) {  delay(1000);  Serial.println("Connecting to WiFi...");  }  Serial.println("Connected to WiFi");  }  void loop() {  float temperature = dht.readTemperature();    if (isnan(temperature)) {  Serial.println("Failed to read from DHT sensor!");  return;  }    Serial.print("Temperature: ");  Serial.print(temperature);  Serial.println(" \*C");  if (temperature > 25) {  digitalWrite(RELAY\_PIN, HIGH); // Turn on the appliance  Serial.println("Appliance turned ON");  } else {  digitalWrite(RELAY\_PIN, LOW); // Turn off the appliance  Serial.println("Appliance turned OFF");  }  delay(2000);  } |

## 1.Explanation of key code segments:

- **WiFi connection**: Establishes connection to the WiFi network.

- **Temperature reading**: Reads data from the DHT11 sensor.

- **Relay control**: Turns the relay on or off based on the temperature reading.

# 13. Simulation Output Link

## 1.Link to the simulation results and output:

|  |
| --- |
| https://wokwi.com/projects/398558632512308225 |

Simulation results can be viewed on platforms like wowki and integrate with tinkercad editor by setting up the project as described and running the simulation.

# 14. Conclusion

## 1.Summary of the project achievements:

The Smart Home Energy Management System successfully automates the control of home appliances based on real-time temperature data, offering an efficient and user-friendly solution for energy management.

## 2.Potential future improvements and applications:

Future improvements could include integrating additional sensors, enhancing remote monitoring capabilities, and developing a mobile application for better user control and interaction.