Smart Home Automation using ESP32 & Arduino

---

1. Project Overview

Objective

This project integrates an **ESP32 DevKit V1** and an **Arduino** to create a **WiFi-based smart home automation system**. The system monitors **temperature** and **motion detection**, controls a **fan (or LED)**, and allows remote control via a web interface.

Key Features

- **Temperature Monitoring** using **DHT11**
- **Motion Detection** using **IR Sensor**
- **Buzzer Alert** for motion detection
- **Automatic Fan Control** when temperature exceeds 30°C
- **Web Interface** to monitor sensor data and control the fan
- **ESP32 WiFi Connectivity** for remote access

---

2. Components Required

Component	**Quantity**
ESP32 DevKit V1	1
Arduino Uno (Optional)	1
DHT11 Temperature Sensor	1
IR Motion Sensor	1
Buzzer	1
Relay Module (for Fan)	1
LED (Optional instead of Fan)	1
Jumper Wires	As needed

---

3. Circuit Diagram & Wiring

Component	ESP32 Pin
DHT11 (Temp/Humidity)	GPIO 4
IR Sensor (Motion)	GPIO 5
Buzzer	GPIO 18

Fan (Relay / LED)	GPIO 19

Power: Use the **3.3V** pin of ESP32 for **DHT11** and **IR sensor**. The **fan or relay module** can be powered using an external power source (e.g., 5V adapter).

---

4. Arduino & ESP32 Code

ESP32 Code for Web Control

```
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
#include <DHT.h>
#define DHTPIN 4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
#define IR_SENSOR 5
#define BUZZER 18
#define FAN 19

const char* ssid = "Your_WiFi_Name";
const char* password = "Your_WiFi_Password";

AsyncWebServer server(80);

void setup() {
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("\nWiFi Connected! IP: " + WiFi.localIP().toString());

    dht.begin();
    pinMode(IR_SENSOR, INPUT);
    pinMode(BUZZER, OUTPUT);
    pinMode(FAN, OUTPUT);

    server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
        float temp = dht.readTemperature();
        int irValue = digitalRead(IR_SENSOR);
        String html = "<h1>ESP32 Smart System</h1>";
        html += "<p>Temperature: " + String(temp) + "°C</p>";
        html += "<p>Motion: " + String(irValue ? "No Motion" : "Motion Detected!") + "</p>";
        html += "<button onclick=\"fetch('/fan_on')\">Turn ON Fan</button>";
        html += "<button onclick=\"fetch('/fan_off')\">Turn OFF Fan</button>";
        request->send(200, "text/html", html);
    });

    server.on("/fan_on", HTTP_GET, [](AsyncWebServerRequest *request){
        digitalWrite(FAN, HIGH);
        request->send(200, "text/plain", "Fan Turned ON");
    });

    server.on("/fan_off", HTTP_GET, [](AsyncWebServerRequest *request){
        digitalWrite(FAN, LOW);
        request->send(200, "text/plain", "Fan Turned OFF");
    });

    server.begin();
}
void loop() {
    float temp = dht.readTemperature();
    int irValue = digitalRead(IR_SENSOR);

    if (irValue == LOW) {
        digitalWrite(BUZZER, HIGH);
    delay(500);
    digitalWrite(BUZZER, LOW);
  }
  if (temp > 30) {
    digitalWrite(FAN, HIGH);
  } else {
    digitalWrite(FAN, LOW);
  }

  delay(1000);
}
```

---

5. Setup & Execution

Step 1: Upload Code to ESP32

1. Install **ESP32 Board** in Arduino IDE.
2. Install required libraries:
   - ESPAsyncWebServer
   - AsyncTCP
   - DHT sensor library
3. Replace Your_WiFi_Name and Your_WiFi_Password with your WiFi details.
4. Select **"ESP32 Dev Module"** under **Tools > Board**.
5. Upload the code and open the **Serial Monitor** (115200 baud) to find the ESP32 **IP Address**.

Step 2: Open Web Interface

1. Enter the ESP32 **IP Address** (e.g., http://192.168.1.100) in a web browser.
2. Monitor **temperature** and **motion**.
3. Control the **fan** (or LED) using ON/OFF buttons.