# IoT-Based Smart Home Automation System

**Project Components & Working :**

**1. Hardware Components:**

* **Microcontroller (ESP8266/ESP32/Raspberry Pi) → Acts as the brain of the system, connecting sensors and appliances to the internet.**
* **Relays (for switching devices) → Controls high-voltage appliances like lights and fans.**
* **Motion Sensor (PIR Sensor) → Detects human presence to turn lights on/off automatically.**
* **Temperature Sensor (DHT11/DHT22) → Measures room temperature for automatic AC control.**
* **Smart Plug/Relay Module → Enables remote control of non-smart appliances.**

**2. Software & Communication:**

* **Mobile App (Android/iOS) → Acts as the user interface to control appliances remotely.**
* **Blynk/Google Firebase/MQTT (for communication) → Cloud services that enable remote communication between the mobile app and microcontroller.**
* **Arduino IDE/Micropython (for coding ESP8266/ESP32) → Used to program the microcontroller.**
* **Wi-Fi/Bluetooth → Enables connectivity between hardware and mobile app.**

**3. Features & Functionalities:**

**✅ Remote Control via Mobile App**

* **Users can turn ON/OFF appliances from anywhere.**
* **Control fan speed or AC temperature.**

**✅ Automation using Sensors**

* **PIR Sensor: Detects motion and turns lights ON when someone enters; turns OFF when no motion is detected for a set time.**
* **Temperature Sensor: If room temperature exceeds a set value (e.g., 30°C), the AC turns ON automatically.**

**✅ Scheduling & Voice Control**

* **Set timers to turn appliances ON/OFF at specific times.**
* **Integrate with Google Assistant/Alexa for voice commands.**

**✅ Energy Monitoring (Optional)**

* **Use current sensors (ACS712) to monitor power usage and send reports via the mobile app.**

**How It Works (Workflow)**

1. **User opens the mobile app and selects an appliance to control.**
2. **App sends a command to the microcontroller via Wi-Fi (through Firebase/MQTT).**
3. **Microcontroller receives the command and activates/deactivates the relay connected to the appliance.**
4. **Sensor-based automation happens in parallel (lights turn off if no motion is detected).**
5. **Status updates (ON/OFF) are sent back to the mobile app for user feedback.**

**Components Required**

| **Component** | **Quantity** | **Purpose** |
| --- | --- | --- |
| **ESP32 (or ESP8266)** | **1** | **Main microcontroller to handle Wi-Fi and control appliances** |
| **Relay Module (5V)** | **2** | **To switch lights and fan ON/OFF remotely** |
| **PIR Sensor (HC-SR501)** | **1** | **Detects human presence for light automation** |
| **DHT11 Temperature Sensor** | **1** | **Measures temperature for AC automation** |
| **LED (Simulating Light)** | **1** | **Represents a light bulb** |
| **DC Motor (Simulating Fan)** | **1** | **Represents a ceiling fan** |
| **Resistor (220Ω)** | **1** | **For LED protection** |
| **Push Button (optional)** | **1** | **For manual control of appliances** |
| **Wires (Jumpers)** | **As needed** | **To connect components** |

**Circuit Connections (ESP32 with Components)**

| **ESP32 Pin** | **Component** | **Component Pin** | **Purpose** |
| --- | --- | --- | --- |
| **GPIO 5** | **Relay Module 1** | **IN1** | **Controls Light** |
| **GPIO 18** | **Relay Module 2** | **IN2** | **Controls Fan** |
| **GPIO 4** | **PIR Sensor** | **OUT** | **Detects motion** |
| **GPIO 15** | **DHT11 Sensor** | **DATA** | **Reads temperature** |
| **GPIO 2** | **LED (Light)** | **Anode (+)** | **Represents a controlled light** |
| **GND** | **LED Resistor (220Ω)** | **Cathode (-)** | **Current limiting resistor** |
| **3.3V** | **PIR Sensor** | **VCC** | **Power supply for PIR** |
| **GND** | **PIR Sensor** | **GND** | **Ground connection** |
| **5V** | **Relay Module** | **VCC** | **Power for relays** |
| **GND** | **Relay Module** | **GND** | **Ground for relays** |
| **5V** | **DHT11 Sensor** | **VCC** | **Power for temperature sensor** |
| **GND** | **DHT11 Sensor** | **GND** | **Ground for DHT11** |