Smart Home: Adaptive Thermostat System

Project Type

IoT-Based Smart Home Automation

---

1. Project Overview

The **Smart Home Adaptive Thermostat System** is an IoT solution designed to **analyze user behavior and optimize energy consumption**. Traditional thermostats operate with fixed settings, leading to unnecessary energy usage. This project introduces an adaptive system that automatically adjusts based on environmental conditions, reducing energy costs and enhancing user comfort.

---

2. Problem Statement

- **Existing Issue:** Traditional thermostats lack adaptability and fail to optimize energy usage, leading to wasted electricity and increased costs.

---

3. Proposed Solution

The system uses **temperature and humidity sensors** to monitor environmental conditions.

- If temperature & humidity exceed defined thresholds, the system **activates cooling appliances** (represented by an LED in this prototype).
- When the environment reaches comfortable levels, the system **turns them off automatically**.

---

4. Objectives

- Minimize energy consumption.
- Increase comfort levels automatically without manual intervention.
- Provide a low-cost solution that can be scaled to real-world applications.

---

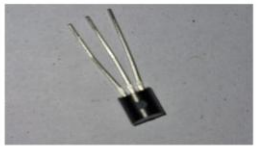
5. Hardware Components

Component	Description
Microcontroller	Arduino (controls entire system)
Temperature Sensor	Monitors ambient room temperature

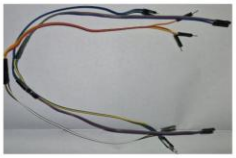
Component	Description
Humidity Sensor	Measures air moisture levels
LED Bulb	Simulates activation of cooling devices
Jumper Wires	Connect components in the circuit

Arduino Uno

LM35(Temperature sensor)



Jumper wires

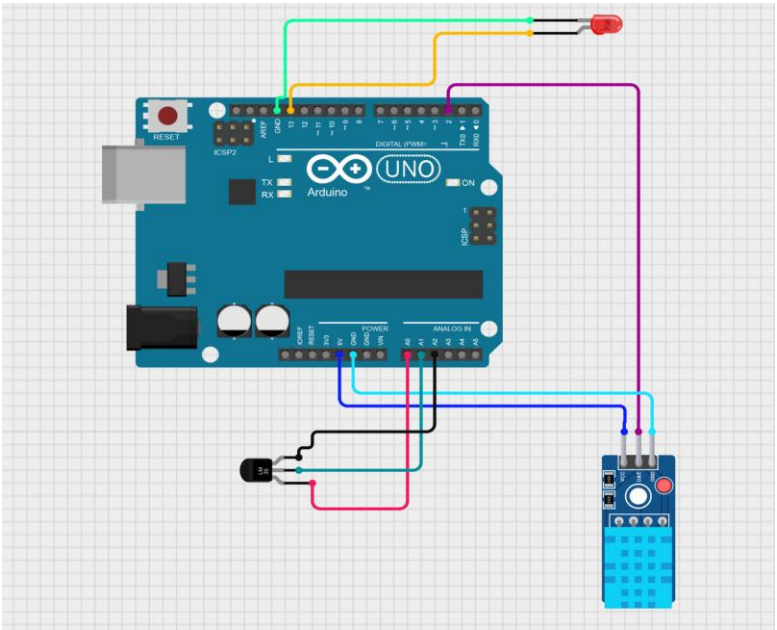


DH11(humidity sensor)

LED bulb

---

6. Circuit Diagram



---

7. System Implementation

Software Development

- Arduino IDE used for programming.
- Code processes sensor data, compares with thresholds, and controls LED.

Hardware Integration

- Sensors connected to Arduino; LED output simulates cooling appliance activation.

User Interface

- Serial monitor displays readings during testing. Future versions may include IoT dashboards.

---

8. Working Principle

1. Sensors read environmental parameters (temperature, humidity).
2. Microcontroller compares readings with thresholds.
3. If values exceed limits → LED ON (cooling simulated).
4. When normal → LED OFF (cooling deactivated).

---

9. Applications

- **Residential Homes:** Automated energy-efficient cooling/heating.
- **Commercial Buildings:** Smart temperature control for offices, hotels, retail spaces.
- **Smart Cities:** Contributes to sustainable urban energy management.

---

10. Advantages

- Energy efficiency
- Cost savings
- User comfort through automation
- Scalable to real appliances

---

11. Limitations

- Prototype uses LED instead of real appliances.
- Lacks advanced machine learning for behavior prediction (can be added in future).

---

12. Conclusion

The **Smart Home Adaptive Thermostat System** demonstrates how IoT can optimize energy use in homes. By monitoring and adjusting environmental conditions automatically, it reduces energy waste and improves user comfort. This project serves as a foundation for fully automated smart home systems.

---

13. Future Scope

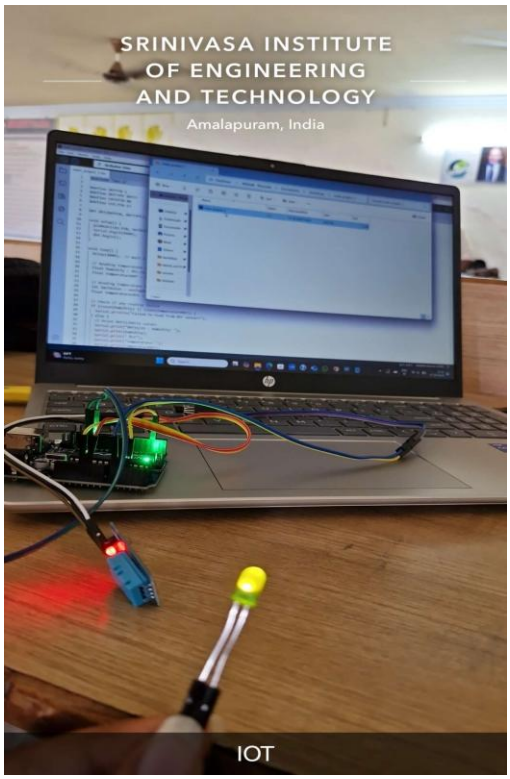
- Integration with **cloud-based IoT platforms**.
- Implementing **machine learning** for user behavior prediction.
- Controlling **multiple appliances** in real time.

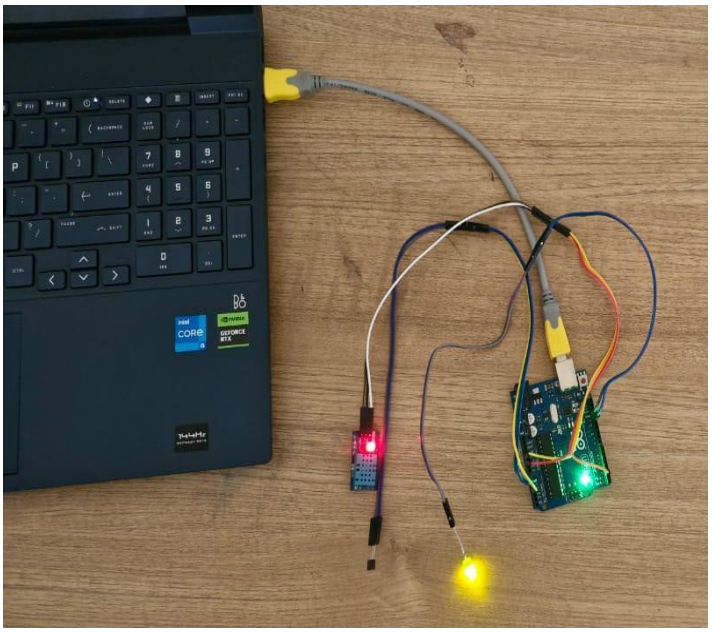
---

14. Arduino Code

```
#include <DHT.h>
#define DHTPIN 2        // Sensor connected to pin 2
#define DHTTYPE DHT11   // Using DHT11 sensor
DHT dht(DHTPIN, DHTTYPE);

int ledPin = 13;        // LED connected to pin 13
float tempThreshold = 30.0;  // Temperature threshold (°C)
float humidityThreshold = 70.0; // Humidity threshold (%)

void setup() {
  Serial.begin(9600);
  dht.begin();
  pinMode(ledPin, OUTPUT);
}

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" °C  Humidity: ");
  Serial.print(h);
  Serial.println(" %");

  if (t > tempThreshold || h > humidityThreshold) {
    digitalWrite(ledPin, HIGH); // Turn on LED (cooling)
  } else {
    digitalWrite(ledPin, LOW);  // Turn off LED
  }
  delay(2000);
}
```

Output:





---

15. References

- Arduino Documentation
- IoT Applications in Smart Homes
- DHT11 Sensor Datasheet