REPORT

Project Title:IoT-based Home Automation using ESP8266 **

1. Introduction:**

The IoT-based Home Automation system using ESP8266 is designed to control and monitor various devices and appliances in a home environment. The system utilizes the ESP8266 microcontroller, which provides Wi-Fi connectivity and acts as a central hub for communication with different sensors and actuators. Through a web or mobile application, users can remotely control and monitor the devices in their homes, enhancing convenience, energy efficiency, and security.

2. Components Used:

- ESP8266 board: Serves as the main microcontroller and provides Wi-Fi connectivity.
- Sensors: Various sensors such as temperature, humidity, motion, and light sensors to gather environmental data.
- Actuators: Relays, motors, and LED lights to control devices and appliances.
- Jumper wires: To establish connections between the components.
- Breadboard or PCB: Provides a platform for mounting and connecting the components.

3. Schematics:

The following schematic diagram illustrates the connection of different sensors and actuators to the ESP8266 board:

```
+------------------+
| ESP8266 |
| |
| D1 (GPIO5)---|----Sensor 1
| D2 (GPIO4)---|----Sensor 2
| D3 (GPIO0)---|----Sensor 3
| D5 (GPIO14)-|----Actuator 1
| D6 (GPIO12)-|----Actuator 2
| D7 (GPIO13)-|----Actuator 3
+------------------+
```

```
```

4. Procedure of Connection:

- Follow the steps below to establish the connections between the components:
- Connect the sensors and actuators to the appropriate GPIO pins on the ESP8266 board.
- Ensure that power and ground connections are properly made for all components.
- Use jumper wires to establish the connections securely.
- Double-check for any loose connections or short circuits.

5. Software Implementation:

To implement the software part of the project, follow these steps:

- Set up the Arduino IDE and install the required libraries: ESP8266WiFi, Adafruit MQTT, and any specific libraries for the sensors and actuators used.
- Open the Arduino IDE and create a new sketch.
- Copy the provided code into the sketch.
- Modify the code to include the necessary configurations, such as Wi-Fi credentials and MQTT server details.
- Customize the code to control and monitor specific devices based on your home automation requirements.
- Upload the code to the ESP8266 board.
- Open the Serial Monitor to view the status and debug information.

6. System Operation:

Once the system is powered on and the code is uploaded successfully, the ESP8266 board starts executing the program. The sensors continuously gather environmental data, which can include temperature, humidity, motion, and light levels. The actuators receive commands from the user interface and control devices such as lights, fans, and motors accordingly.

Users can access the web or mobile application to remotely control and monitor the devices in their homes. They can turn on/off lights, adjust room temperature, receive notifications on motion detection, and view real-time sensor data. The system provides convenience, energy efficiency, and improved security for home automation.

7. Conclusion:

The IoT-based Home Automation system using ESP8266 offers an efficient and convenient way to control and monitor devices in a home environment. It allows users to customize their home automation setup and provides remote access to enhance comfort, energy efficiency, and security.

8. Code:

```
#include <ESP8266WiFi.h>
#include <Adafruit_Sensor.h>
#include <DHT.h>
```

```
#define WIFI_SSID "YourWiFiSSID"
#define WIFI_PASSWORD "YourWiFiPassword"

#define DHT_PIN D1     // Pin connected to the DHT sensor
#define DHT_TYPE DHT11 // Type of the DHT sensor (DHT11 or DHT22)

#define LIGHT_PIN D2    // Pin connected to the light sensor
#define MOTION_PIN D3   // Pin connected to the motion sensor
#define FAN_PIN D5      // Pin connected to the fan relay
#define LIGHT_PIN D6    // Pin connected to the light relay
#define AC_PIN D7       // Pin connected to the AC relay

#define TEMPERATURE_THRESHOLD 25    // Temperature threshold for turning on/off the AC (in degrees
Celsius)
#define HUMIDITY_THRESHOLD 60       // Humidity threshold for turning on/off the fan (in percentage)
#define MOTION_DELAY 10000          // Delay after motion is detected (in milliseconds)
#define LIGHT_THRESHOLD 500         // Light threshold for turning on/off the light (adjust based on your
environment)

WiFiClient client;

DHT dht(DHT_PIN, DHT_TYPE);

void setup() {
  Serial.begin(115200);
  delay(10);

  connectToWiFi();

  dht.begin();
  pinMode(LIGHT_PIN, INPUT);
  pinMode(MOTION_PIN, INPUT);
  pinMode(FAN_PIN, OUTPUT);
  pinMode(LIGHT_PIN, OUTPUT);
  pinMode(AC_PIN, OUTPUT);
}

void loop() {
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();
  int lightLevel = analogRead(LIGHT_PIN);
  bool motionDetected = digitalRead(MOTION_PIN);

  // Control AC based on temperature
  if (temperature > TEMPERATURE_THRESHOLD) {
    digitalWrite(AC_PIN, HIGH); // Turn on the AC
  } else {
    digitalWrite(AC_PIN, LOW); // Turn off the AC
  }

  // Control fan based on humidity
  if (humidity > HUMIDITY_THRESHOLD) {
    digitalWrite(FAN_PIN, HIGH); // Turn on the fan
```

```
  } else {
    digitalWrite(FAN_PIN, LOW); // Turn off the fan
  }

  // Control light based on light level
  if (lightLevel < LIGHT_THRESHOLD) {
    digitalWrite(LIGHT_PIN, HIGH); // Turn on the light
  } else {
    digitalWrite(LIGHT_PIN, LOW); // Turn off the light
  }

  // Motion detection
  if (motionDetected) {
    delay(MOTION_DELAY); // Wait for the defined delay after motion is detected
    if (digitalRead(MOTION_PIN)) {
      // If motion is still detected after delay, perform desired action (e.g., send notification)
      Serial.println("Motion detected!");
    }
  }

  delay(2000); // Delay between sensor readings
}

void connectToWiFi() {
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
  Serial.println("IP address: " + WiFi.localIP().toString());
}
```