IoT Based Smart Fan controlled by Temperature and Online Switch

Shrey Verma
Department of CSE
Indian Institute of Technology
Guwahati, India
v.shrey@iitg.ac.in

I. Introduction

Nowadays everything is moving towards automation. The temperature keeps varying throughout the day, there is a need of a fan which switches itself OFF when it is cold and turns ON when the room is hot. Also, what should one do if the fan is left ON before leaving the room? How can it be switched OFF? These problems are solved by this IoT device. There are major advantages such as Power Saving and increase in comfort by automating home electronic appliances.

II. Main objective

The objective of this project is to solve the problem of wastage of energy and hence reducing electricity bill amount and removing the need for physical switch which is a factor of comfort. In this project we will create a fan which has a threshold temperature i.e. if **temperature of room is higher than the threshold temperature the fan will start**. Also, we will implement an **IoT controller using cloud which can switch the fan ON / OFF** and also **change the threshold temperature.**

III. Implemented Attributes

The following is the list of attributes with brief definition of how they are implemented. How which all the attributes given in assignment: -

A. DHT11 sensor senses the Environment

The DHT11 sensor is used to sense the temperature.

B. DC motor (Actuator) turns ON/OFF based on temperature and controller switch

The DC motor corresponds to a fan in our project. It **turns on when the temperature sensed by DHT11 reaches above threshold** value and also be **controlled by switch given in the controller interface**. First, we have to disable the switching based on temperature Off by setting Automatic mode to 0. Now, the fan is in manual mode. Now one can use the interface's fan switch to control the fan.

C. Internet connectivity to access the Cloud

Used my WiFi router for providing access to internet. The ESP8266 is connected by providing ssid, password.

D. Data visualization is done using Cloud application

All the values are **sent to Cayenne cloud via virtual pins** connected to our cloud device. Also, the controlling buttons change the values of state or threshold temperature by sending data to ESP8266 via virtual pins.

Complete visualization of system state is done using **Cayenne Dashboard**

The data for the following variables is visualized on the dashboard: -

- Temperature: Visualized using Chart and a float value display box
- Fan State Bool: Visualized using 2 state 0/1 value display box.

E. Cayenne Cloud is used to control the system

Complete system is controlled by using buttons and sliders in Cayenne Dashboard

All the variables are provided with controllers in the dashboard, **one can change Threshold Temperature, Toggle Automatic Mode and Fan Switch** (when automatic mode is turned off).

The value of the following variables is overwritten to control the system: -

- Threshold Temperature: Using slider
- Automatic Mode: 0/1 Slider
- Fan Switch: 0/1 Slider

IV. Hardware & software components

Sensor: - DHT11 - To sense temperature

Actuator: - DC Motor – Corresponds to Fan,
5v Relay – Acts as an MCU controlled switch for powering DC motor

MCU: - NodeMCU with embedded ESP8266

Cloud: - Cayenne – Cloud platform designed to display, respond and interact with the project data

Router: - D- Link 2.4GHz WiFi Router

V. Data flows implemented

Out of the list given in assignment all the data flow paths have been implemented. The following are various data flows with brief explanation: -

a) Reading Sensor Data and Displayed in Serial Monitor

Done using the line in the code. `t=dht.readTemperature();println(t);`

b) Transmit the data to Cloud and display it on cloud

The CayenneMQTT library provides methods to connect to internet by providing credentials in

```
" char ssid[] = "ssid";
  char wifiPassword[] = "pass;  "
```

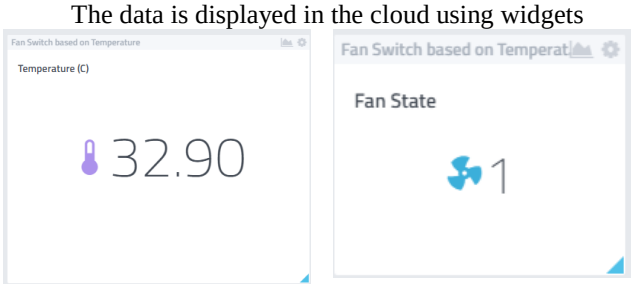
After connecting to internet connecting our project to Cayenne Cloud using the following code after entering credentials provided in cayenne.

```
"char username[] = "XXXXXXX";
 char password[] = "YYYYYY";
 char clientID[] = "ZZZZZZ";"
```

The data is sent to cloud using virtual pin

```
"CAYENNE_OUT(VIRTUAL_PIN) {
 Cayenne.virtualWrite(VIRTUAL_PIN,
VARIABLE NAME);}"
```

The data is displayed in the cloud using widgets



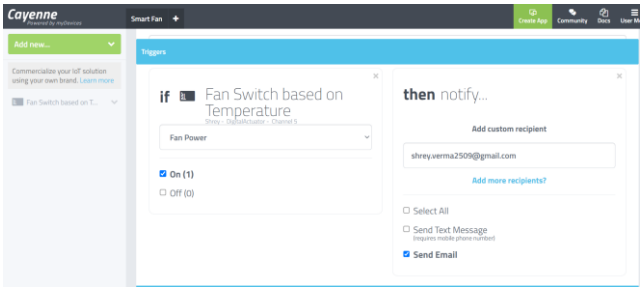
c) Controlling system using command from cloud..

The CayenneMQTT library also provides method to change the value of variables using cloud buttons/sliders.
The data is received from cloud using virtual channels. For example, threshold value can be changed using slider given in the project. The code for the same is as follows: -

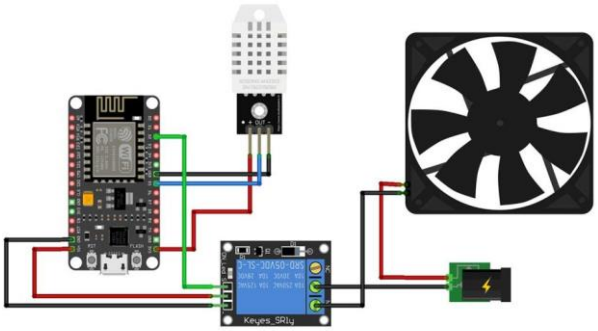
```
"CAYENNE_IN(2){
      threshold = getValue.asInt();}"
```

d) Sending alert to user's email is implemented

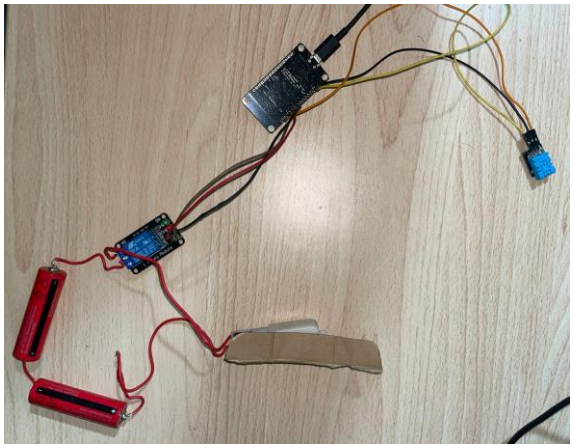
In Cayenne cloud we get an option to create triggers. In this project there is a trigger which sends an e-mail to the user when the fan is switched on.



VI. Configuration Diagram



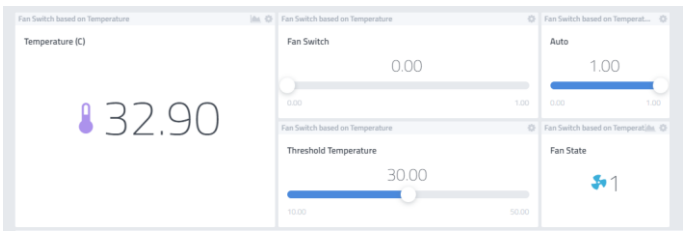
Concept



Realize

VII. Sample Output

The fan is set to auto mode and the room temperature i.e. 32.9 C is greater than threshold hence the fan state is 1 on.

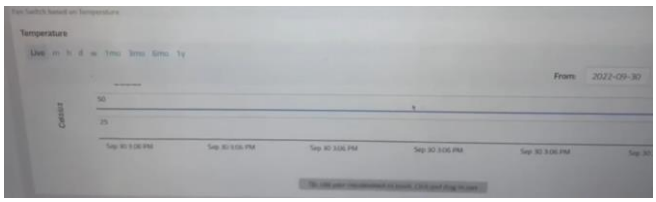


ESP8266 connecting to Cayenne and temperature reading output: -

VIII. User Manual

Turn on the NodeMCU and visit the following link to open the dashboard: -
https://cayenne.mydevices.com/shared/63360dcb321ecb5eafed39c7

Temperature Vizualization: -



Code Explanation

1) *Including libraries and intializing variables*

```
#define CAYENNE_PRINT Serial
#define DHTPIN 5
int relay = 0;
#define DHTTYPE DHT11

#include <CayenneMQTTESP8266.h>
#include <DHT.h>

// WiFi network info.
char ssid[] = "dlink-CC0D";
char wifiPassword[] = "chmtd93436";

// Cayenne authentication info. This should be obtained from the
char username[] = "24023180-3ffe-11ed-bf0a-bb4ba43bd3f6";
char password[] = "95f5f36e4f384b42baf34d1c809f30072ece7134";
char clientID[] = "492d1dd0-3ffe-11ed-baf6-35fab7fd0ac8";
```

- CayenneMQTT8266 for connecting to Cayenne cloud and accessing its features
- Also, we declare variables corresponding to our WiFi SSID (name discoverable on PC) and password
- Cayenne Cloud credentials to connect the device to our account is also added in the code
- Define pins for motor (relay) and DHT11 (DHTPIN)

2) *Creating global variables and setup()*

```
DHT dht(DHTPIN, DHTTYPE);

float h;
float t;
int ato;
int fan_state;
bool fstt;
int threshold;
void setup() {
  Serial.begin(9600);
  Serial.print("Setup");
  Cayenne.begin(username, password, clientID, ssid, wifiPassword);
  dht.begin();
  pinMode(relay, OUTPUT);
  digitalWrite(relay, HIGH);
  ato = 1;
  fan_state = 0;
  fstt = 0;
  threshold = 25;
}
```

Begin serial to get serial monitor output.
Next, we define variables for taking readings of DHT sensors, and for controlling relay motor.
Initializing values of variables.

3) *Creating input channels for variables*

The CayenneMQTT library also provides method to change the value of variables using cloud buttons/sliders.
The data is received from cloud using virtual channels.
For example, threshold value can be changed using slider given in the project.

```
CAYENNE_IN(2)
{
  threshold = getValue.asInt();
  Serial.print("Threshold Temperature changed to: ");
  Serial.println(threshold);
  if (ato == 1) {
    if (t > threshold) {digitalWrite(relay, LOW);fan_state = 1;fstt = fan_state;}
    else { digitalWrite(relay, HIGH); fan_state = 0;fstt = fan_state;}
  }
  CAYENNE_OUT_DEFAULT();
}

CAYENNE_OUT(5) {
  Cayenne.virtualWrite(5, fstt);
}
```

Cayenne cloud also provides out channel to display data through widgets an example of out code is given below. Here we are sending temperature to cloud using the following method.
"Cayenne.celciusWrite();"

```
CAYENNE_OUT(1) {
  do {
    t = dht.readTemperature();
    delay(1000);
  } while (isnan(t));
  if (ato == 1) {
    if (t > threshold) {digitalWrite(relay, LOW);fan_state = 1;fstt = fan_state;}
    else { digitalWrite(relay, HIGH); fan_state = 0;fstt = fan_state;}
  }else{
    if (fan_state == 1) {
      digitalWrite(relay, LOW);fan_state = 1;fstt = fan_state;
    } else {
      digitalWrite(relay, HIGH);fan_state = 0;fstt = fan_state;
    }
  }
  Serial.print("temperature: ");
  Serial.println(t);
  Cayenne.celsiusWrite(1, t);
  CAYENNE_OUT_DEFAULT();
}
```

Similarly, all other variables are sent out or in and in each out/in there is a part of code which switches fan according to current state.