AUTOMATIC AIR FRESHENER

PROJECT REPORT

ZEE BALOCH

Introduction

Our Project is automatic air freshener with IoT analytics with mobile app control. Our project aim is to improve air quality of our environment. We want our environment to be refreshing and fragrant. Our mobile app put the control directly in user hands allowing us to customize and manage air freshener with ease and flexibility. This system can use in houses and work place.

Detail of Project

For developing Air freshener control by mobile application we require following components. Here is the details we need to consider:

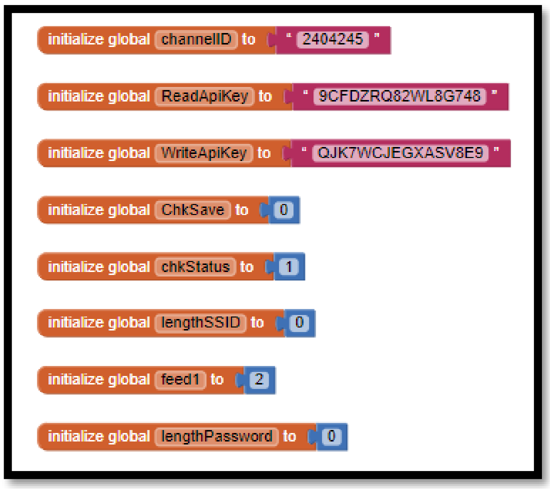
Hardware Component:

1. **Air Freshener:** We use liquid air freshener bottle as air freshener.
2. **Motor pump:** We have use motor pump for dispensing air freshener liquid. It is efficient in delivering air freshener. Incorporating motor pump will help in automation of air freshener. The amount of air freshener is dispensed more precisely leading to less waste compared to manual dispense where excessive liquid is may be poured or sprayed.
3. **Female and Male headers:** Female and male headers are used to provide convenient way to make temporary or semi-permanent connection on printed circuit or Vero board. Male header have pins that can easily insert corresponding socket of female header. It help in assemble and dissemble circuit much easier. We use header for mounting esp-32.
4. **Relay Module:** Relay module is used to control high power devices such as motor, light etc. In air freshener which may require high voltage or current than relay can handle it. Relay module can be easily interfaced with microcontroller or IoT devices. Relay module are available in various configuration to suite different voltage and currant
5. **Voltage divider:** It is used to create specific voltage from higher voltage. It uses two register connected the voltage across each register is proportional to its resistance value. It one of application is of voltage divider is it reduce from higher voltage.
6. **Esp-32:** It is the powerful microcontroller especially used for IoT applications. It is dual-core processor running on 240Hz. It integrate Wi-Fi and Bluetooth connectivity allowing device connect to the internet. It is low powered. It has wide range of peripheral such as SPI, I^{2}C, UART, GPIO, ADC, DAC and more. These peripheral enable sensors, actuator, displays, etc.

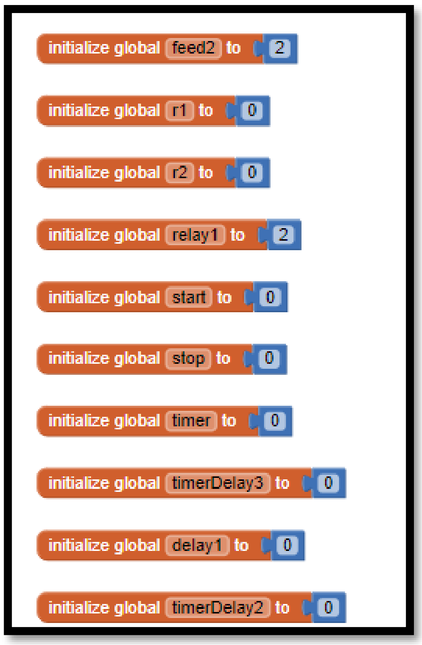
Software Component:

1. **Mobile App:** We have develop user friendly interface for air freshener on MIT app inverter. MIT app inverter code for our data is this:

Initialization of variables:

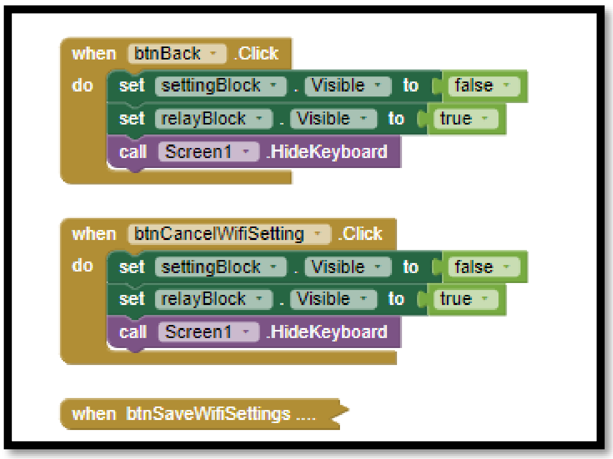


```
initialize global channelID to " 2404245 "
initialize global ReadApiKey to " 9CFDZRQ82WL8G748 "
initialize global WriteApiKey to " QJK7WCJEGXASV8E9 "
initialize global ChkSave to 0
initialize global chkStatus to 1
initialize global lengthSSID to 0
initialize global feed1 to 2
initialize global lengthPassword to 0
```

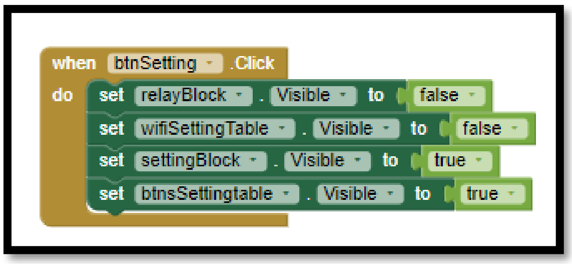


```
initialize global feed2 to 2
initialize global r1 to 0
initialize global r2 to 0
initialize global relay1 to 2
initialize global start to 0
initialize global stop to 0
initialize global timer to 0
initialize global timerDelay3 to 0
initialize global delay1 to 0
initialize global timerDelay2 to 0
```

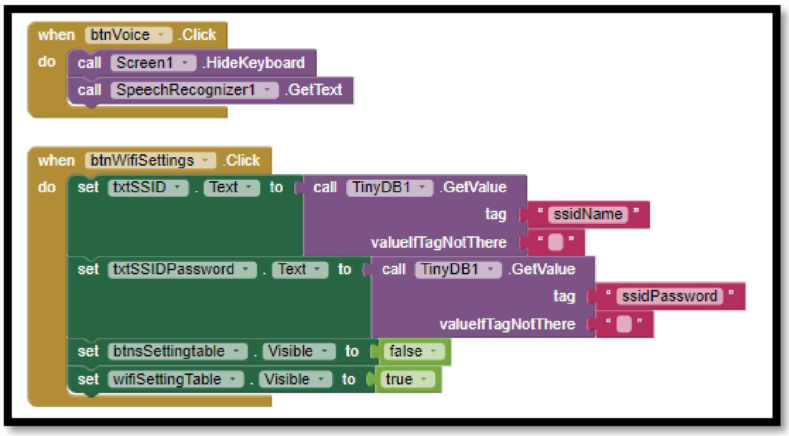
Wi-Fi Setting:



```
when btnBack .Click
do  set settingBlock . Visible to false
    set relayBlock . Visible to true
    call Screen1 .HideKeyboard

when btnCancelWifiSetting .Click
do  set settingBlock . Visible to false
    set relayBlock . Visible to true
    call Screen1 .HideKeyboard

when btnSaveWifiSettings ....
```

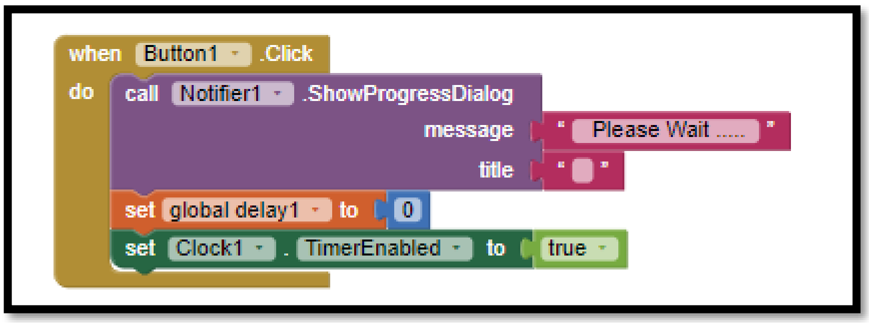


```
when btnSetting .Click
do  set relayBlock . Visible to false
    set wifiSettingTable . Visible to false
    set settingBlock . Visible to true
    set btnsSettingtable . Visible to true
```



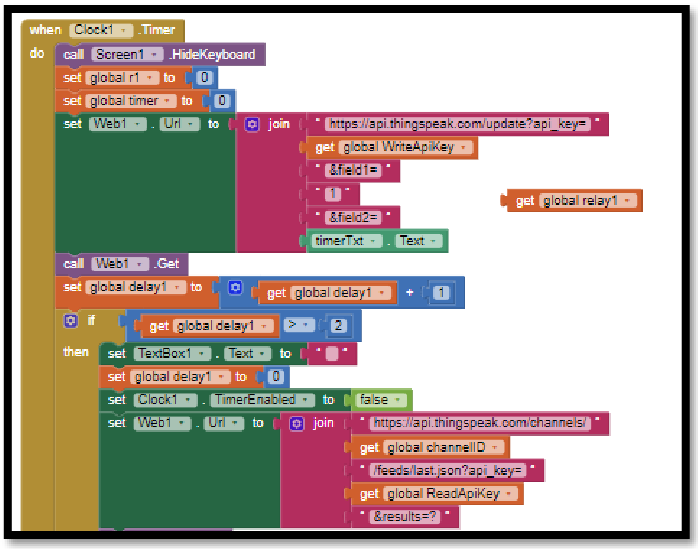
```
when btnVoice .Click
do  call Screen1 .HideKeyboard
    call SpeechRecognizer1 .GetText

when btnWifiSettings .Click
do  set txtSSID . Text to  call TinyDB1 .GetValue
                                       tag  " ssidName "
                         valueIfTagNotThere  "  "
    set txtSSIDPassword . Text to  call TinyDB1 .GetValue
                                              tag  " ssidPassword "
                                valueIfTagNotThere  "  "
    set btnsSettingtable . Visible to  false
    set wifiSettingTable . Visible to  true
```

ON/OFF button:

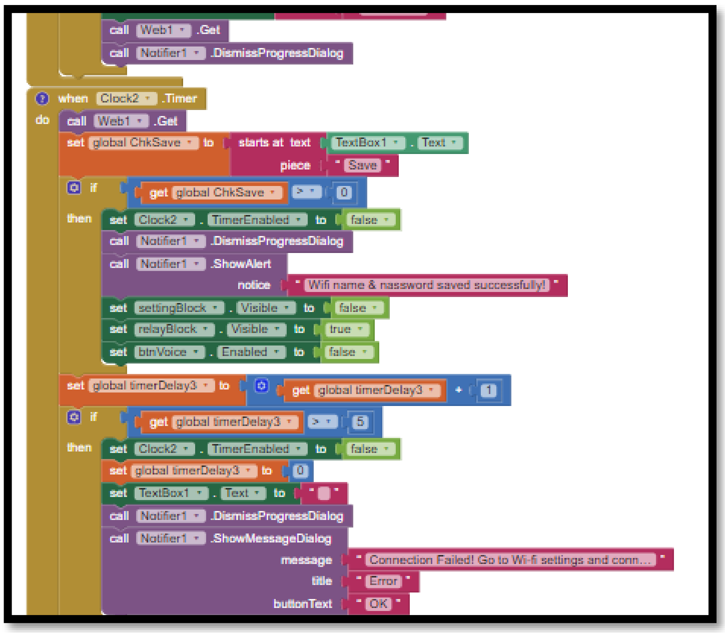


```
when Button1 .Click
do  call Notifier1 .ShowProgressDialog
                              message  " Please Wait ..... "
                                title  "  "
    set global delay1 to  0
    set Clock1 . TimerEnabled to  true
```

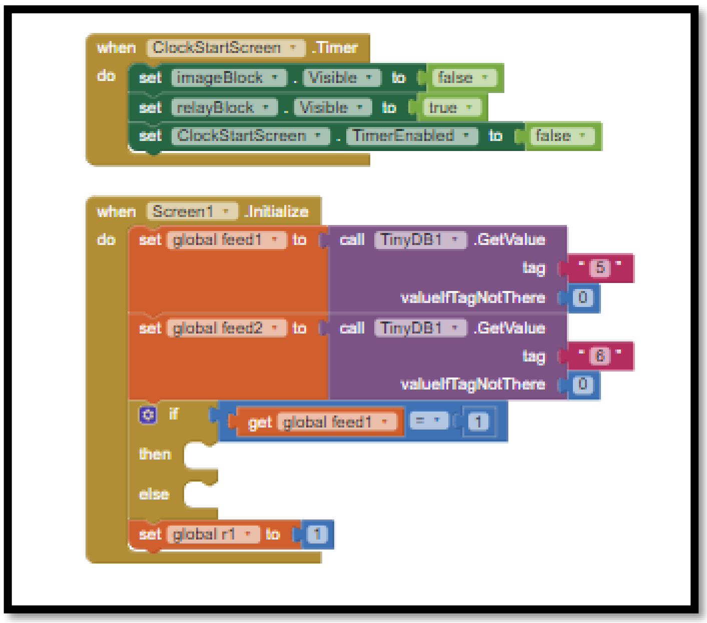
Timer Setting:



```
when Clock1 .Timer
do  call Screen1 .HideKeyboard
    set global r1 to  0
    set global timer to  0
    set Web1 . Url to  join  " https://api.thingspeak.com/update?api_key= "
                             get global WriteApiKey
                             " &field1= "
                             " 1 "                                get global relay1
                             " &field2= "
                             timerTxt . Text
    call Web1 .Get
    set global delay1 to  get global delay1 + 1
    if  get global delay1 > 2
    then  set TextBox1 . Text to  "  "
          set global delay1 to  0
          set Clock1 . TimerEnabled to  false
          set Web1 . Url to  join  " https://api.thingspeak.com/channels/ "
                                   get global channelID
                                   " /feeds/last.json?api_key= "
                                   get global ReadApiKey
                                   " &results=? "
```

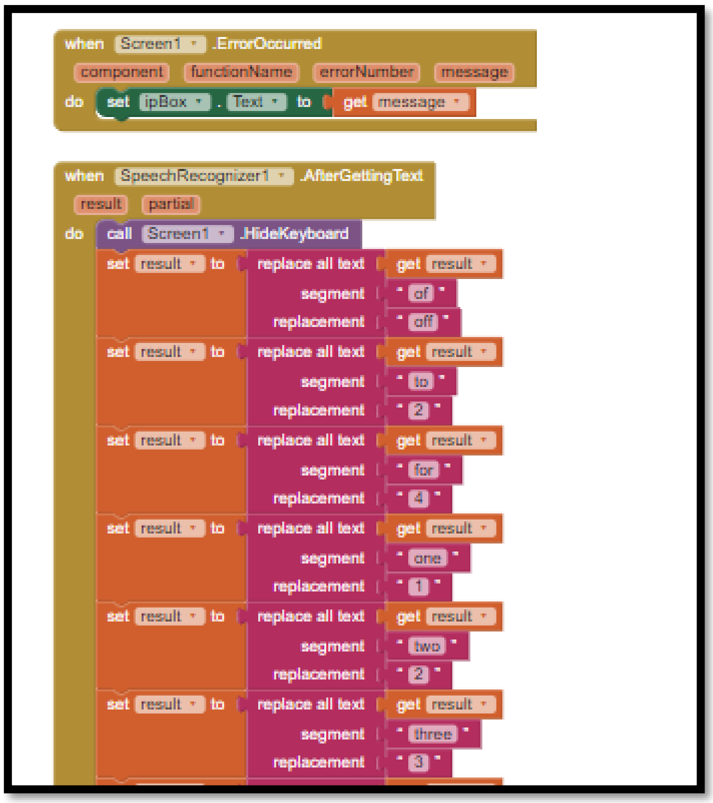


```
        call Web1 .Get
        call Notifier1 .DismissProgressDialog
when Clock2 .Timer
do  call Web1 .Get
    set global ChkSave to  starts at text  TextBox1 . Text
                                     piece  " Save "
    if  get global ChkSave > 0
    then  set Clock2 . TimerEnabled to false
          call Notifier1 .DismissProgressDialog
          call Notifier1 .ShowAlert
                          notice  " Wifi name & password saved successfully! "
          set settingBlock . Visible to false
          set relayBlock . Visible to true
          set btnVoice . Enabled to false
    set global timerDelay3 to  get global timerDelay3 + 1
    if  get global timerDelay3 > 5
    then  set Clock2 . TimerEnabled to false
          set global timerDelay3 to 0
          set TextBox1 . Text to " "
          call Notifier1 .DismissProgressDialog
          call Notifier1 .ShowMessageDialog
                          message  " Connection Failed! Go to Wi-fi settings and conn... "
                            title  " Error "
                       buttonText  " OK "
```

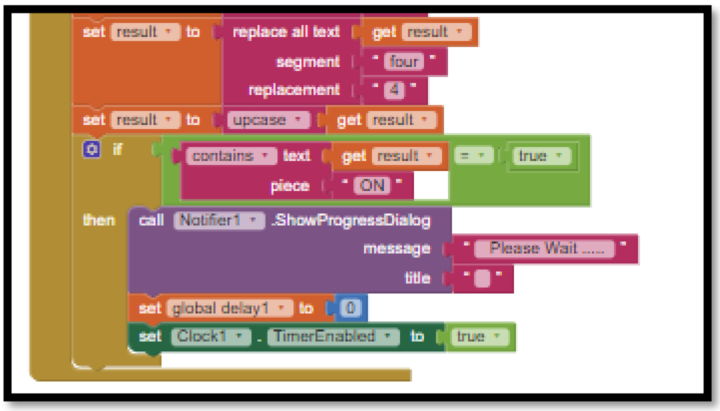


```
when ClockStartScreen .Timer
do  set imageBlock . Visible to false
    set relayBlock . Visible to true
    set ClockStartScreen . TimerEnabled to false

when Screen1 .Initialize
do  set global feed1 to  call TinyDB1 .GetValue
                                          tag  " 5 "
                           valueIfTagNotThere  0
    set global feed2 to  call TinyDB1 .GetValue
                                          tag  " 6 "
                           valueIfTagNotThere  0
    if  get global feed1 = 1
    then
    else
    set global r1 to 1
```

Error Notification:

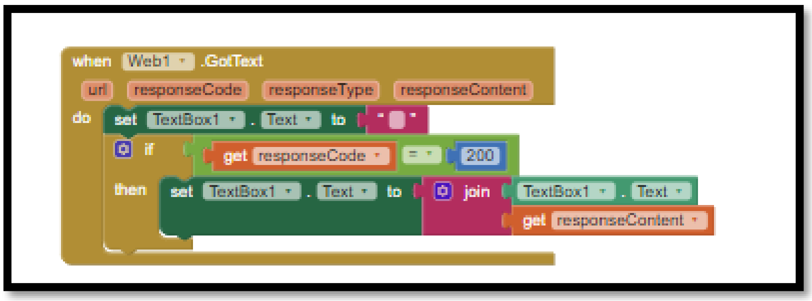


```
when Screen1 .ErrorOccurred
  component  functionName  errorNumber  message
do  set ipBox . Text to  get message

when SpeechRecognizer1 .AfterGettingText
  result  partial
do  call Screen1 .HideKeyboard
    set result to  replace all text  get result
                   segment  " of "
                   replacement  " off "
    set result to  replace all text  get result
                   segment  " to "
                   replacement  " 2 "
    set result to  replace all text  get result
                   segment  " for "
                   replacement  " 4 "
    set result to  replace all text  get result
                   segment  " one "
                   replacement  " 1 "
    set result to  replace all text  get result
                   segment  " two "
                   replacement  " 2 "
    set result to  replace all text  get result
                   segment  " three "
                   replacement  " 3 "
```



```
    set result to  replace all text  get result
                   segment  " four "
                   replacement  " 4 "
    set result to  upcase  get result
    if   contains text  get result  =  true
              piece  " ON "
    then  call Notifier1 .ShowProgressDialog
                        message  " Please Wait ..... "
                        title  "  "
          set global delay1 to  0
          set Clock1 . TimerEnabled to  true
```

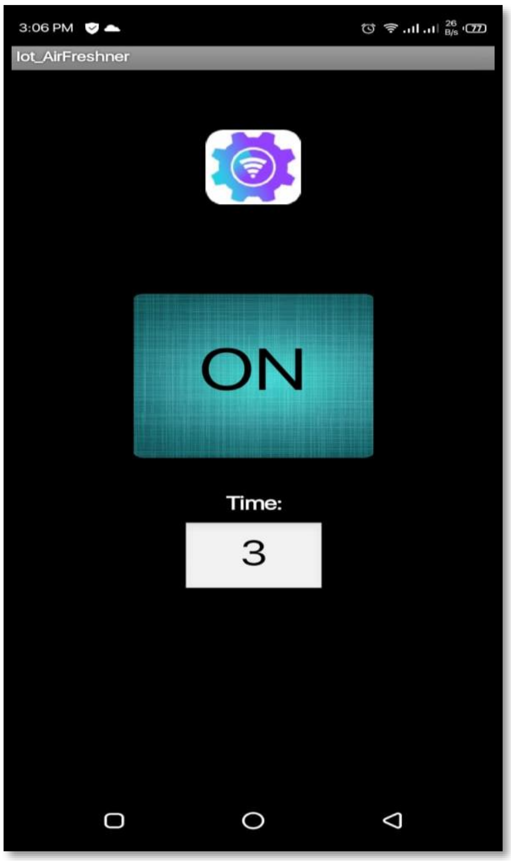
Response of Air freshener:

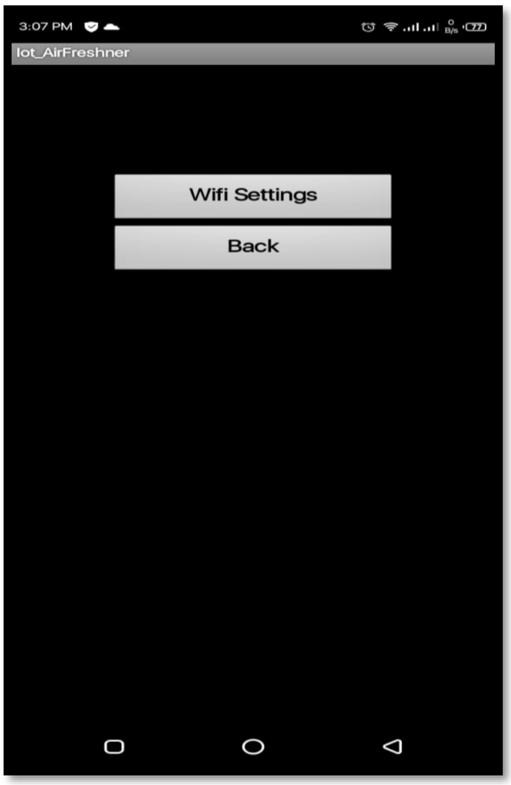


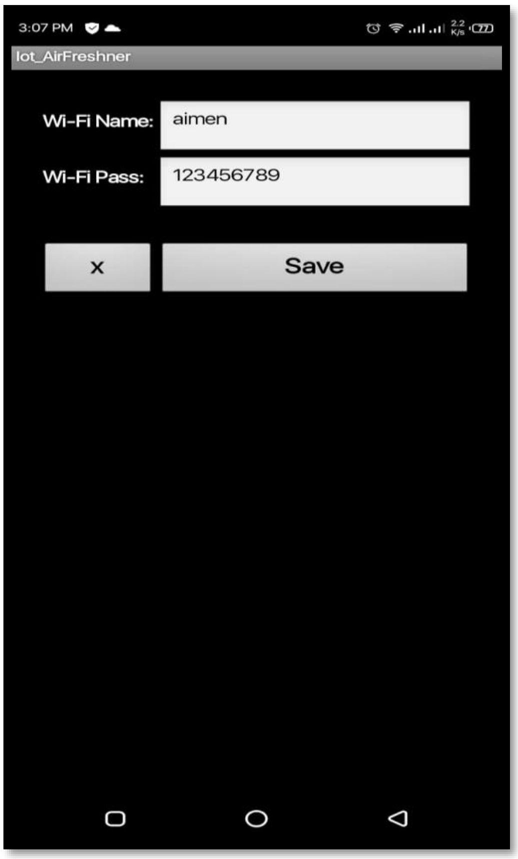
```
when Web1 .GotText
  url  responseCode  responseType  responseContent
do  set TextBox1 . Text to  "  "
    if   get responseCode  =  200
    then  set TextBox1 . Text to  join  TextBox1 . Text
                                        get responseContent
```

Mobile Application Splash Screen:

9:48
Iot_AirFreshner
IoT AirFreshner


3:06 PM
Iot_AirFreshner
ON
Time:
3


3:07 PM
Iot_AirFreshner
Wifi Settings
Back


3:07 PM

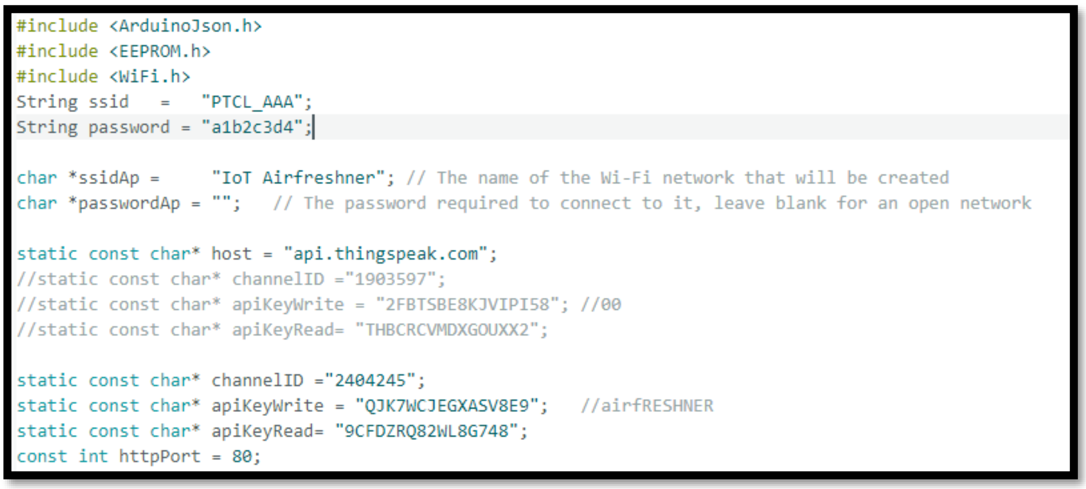
Iot_AirFreshner

Wi-Fi Name: aimen

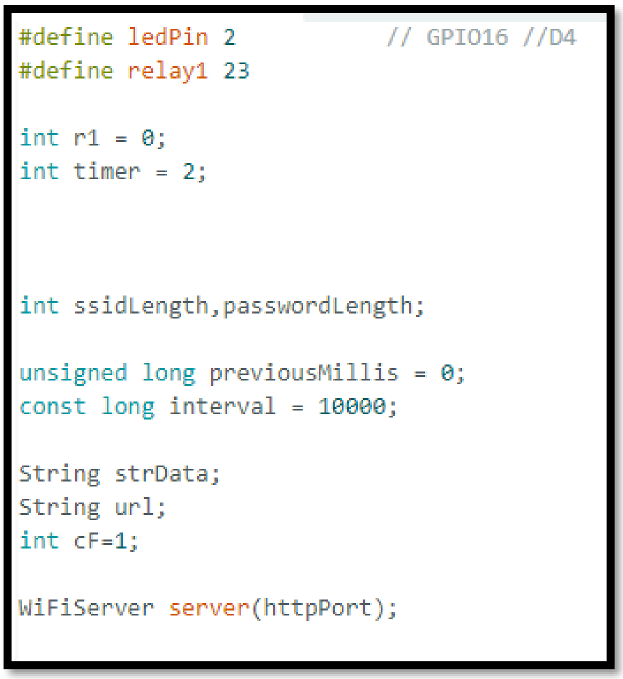
Wi-Fi Pass: 123456789

2. Esp-32 code on Arduino IDE:

Think Speak Connectivity:

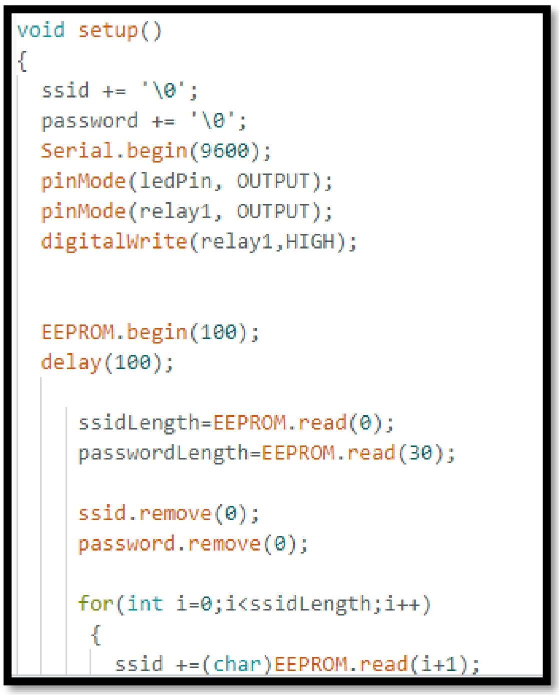

```
#include <ArduinoJson.h>
#include <EEPROM.h>
#include <WiFi.h>
String ssid    =   "PTCL_AAA";
String password = "a1b2c3d4";

char *ssidAp =      "IoT Airfreshner"; // The name of the Wi-Fi network that will be created
char *passwordAp = "";    // The password required to connect to it, leave blank for an open network

static const char* host = "api.thingspeak.com";
//static const char* channelID ="1903597";
//static const char* apiKeyWrite = "2FBTSBE8KJVIPI58"; //00
//static const char* apiKeyRead= "THBCRCVMDXGOUXX2";

static const char* channelID ="2404245";
static const char* apiKeyWrite = "QJK7WCJEGXASV8E9";   //airfRESHNER
static const char* apiKeyRead= "9CFDZRQ82WL8G748";
const int httpPort = 80;
```

Esp-32 port Connection:

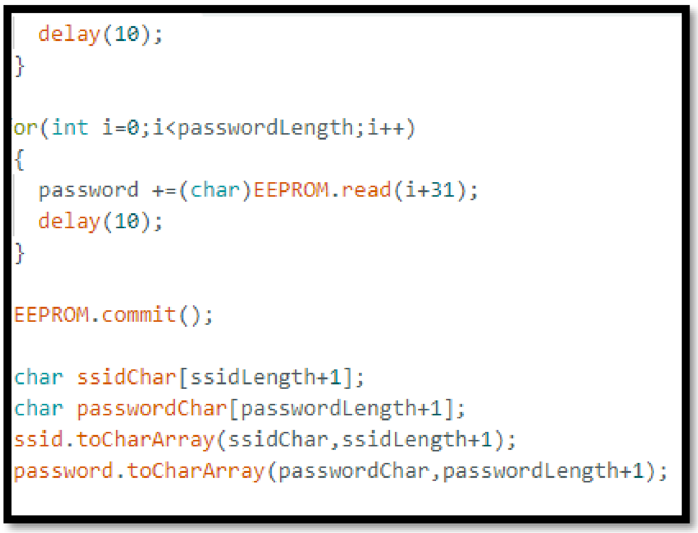


```
#define ledPin 2            // GPIO16 //D4
#define relay1 23

int r1 = 0;
int timer = 2;



int ssidLength,passwordLength;

unsigned long previousMillis = 0;
const long interval = 10000;

String strData;
String url;
int cF=1;

WiFiServer server(httpPort);
```

Setup EEPROM Connection:

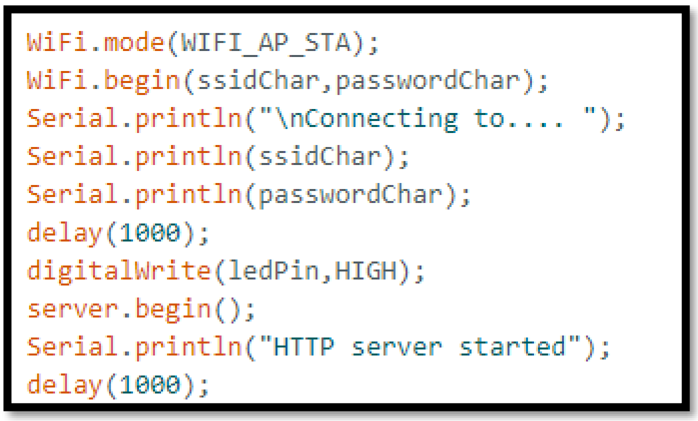


```
void setup()
{
  ssid += '\0';
  password += '\0';
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(relay1, OUTPUT);
  digitalWrite(relay1,HIGH);


  EEPROM.begin(100);
  delay(100);

     ssidLength=EEPROM.read(0);
     passwordLength=EEPROM.read(30);

     ssid.remove(0);
     password.remove(0);

     for(int i=0;i<ssidLength;i++)
      {
        ssid +=(char)EEPROM.read(i+1);
```



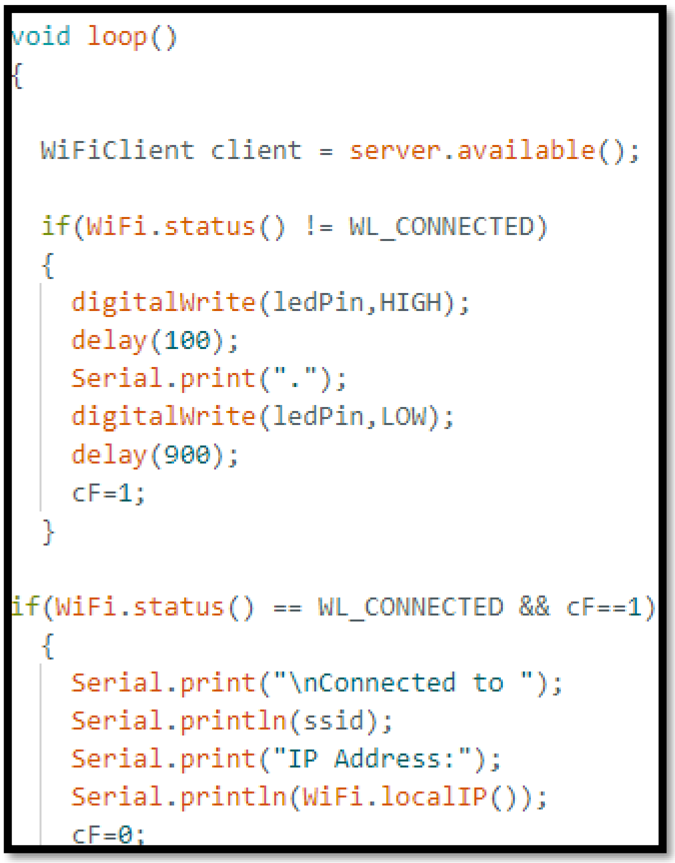
```
  delay(10);
}

or(int i=0;i<passwordLength;i++)
{
  password +=(char)EEPROM.read(i+31);
  delay(10);
}

EEPROM.commit();

char ssidChar[ssidLength+1];
char passwordChar[passwordLength+1];
ssid.toCharArray(ssidChar,ssidLength+1);
password.toCharArray(passwordChar,passwordLength+1);
```

Setup Wi-Fi Connection:

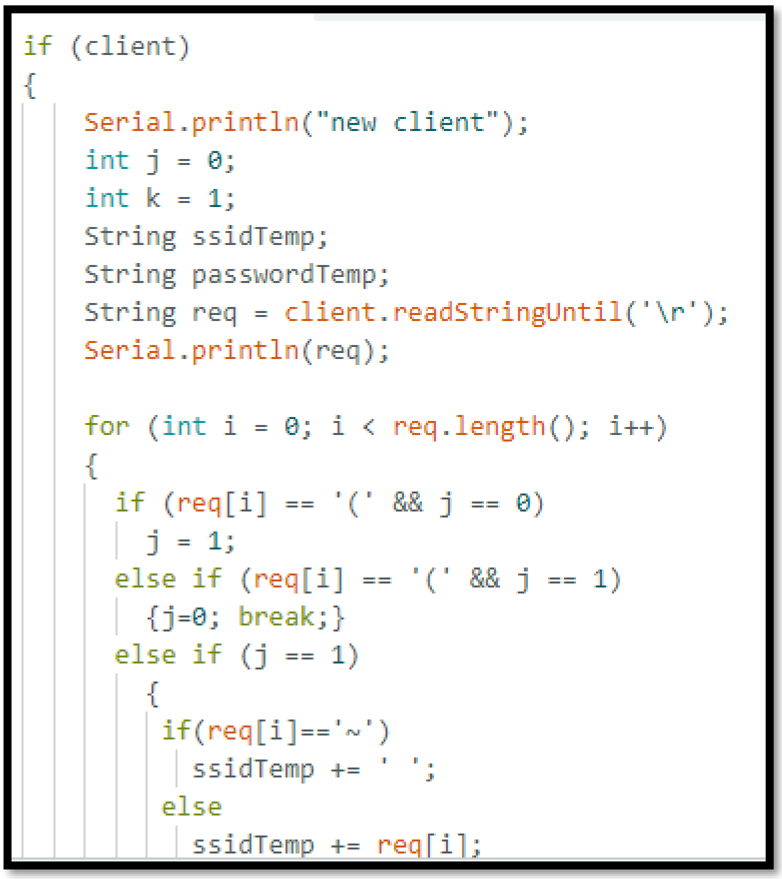


```
WiFi.mode(WIFI_AP_STA);
WiFi.begin(ssidChar,passwordChar);
Serial.println("\nConnecting to.... ");
Serial.println(ssidChar);
Serial.println(passwordChar);
delay(1000);
digitalWrite(ledPin,HIGH);
server.begin();
Serial.println("HTTP server started");
delay(1000);
```

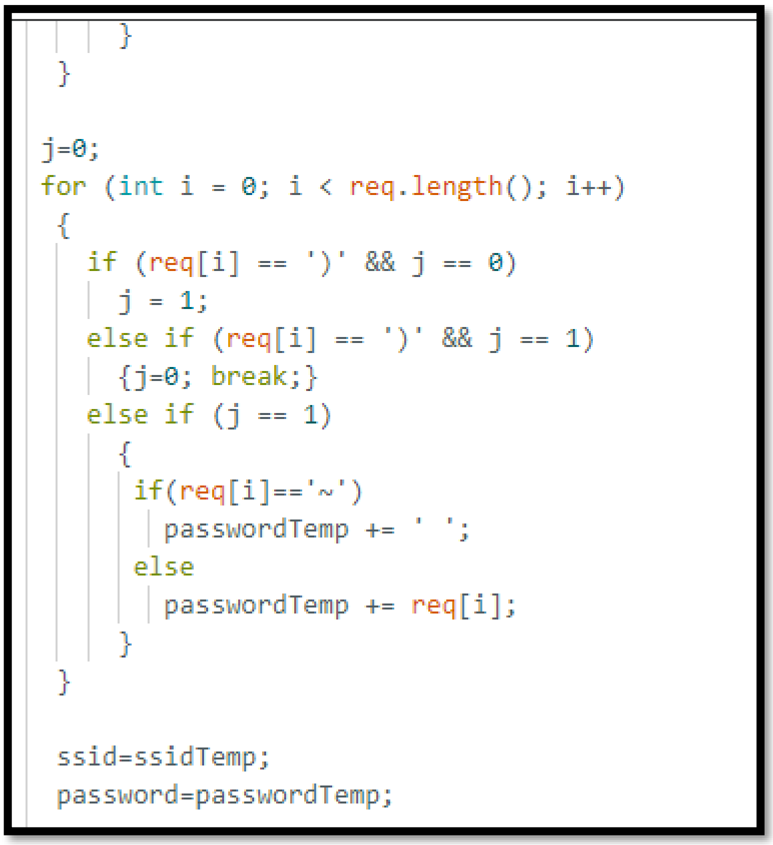
Wi-Fi Connection:



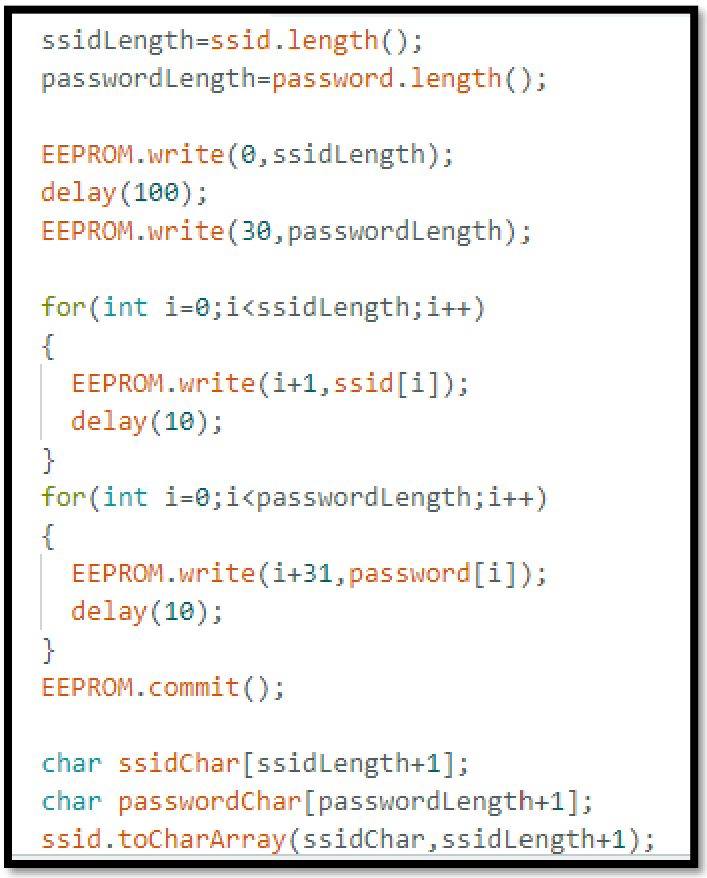
```
void loop()
{

  WiFiClient client = server.available();

  if(WiFi.status() != WL_CONNECTED)
  {
    digitalWrite(ledPin,HIGH);
    delay(100);
    Serial.print(".");
    digitalWrite(ledPin,LOW);
    delay(900);
    cF=1;
  }

if(WiFi.status() == WL_CONNECTED && cF==1)
  {
    Serial.print("\nConnected to ");
    Serial.println(ssid);
    Serial.print("IP Address:");
    Serial.println(WiFi.localIP());
    cF=0;
```



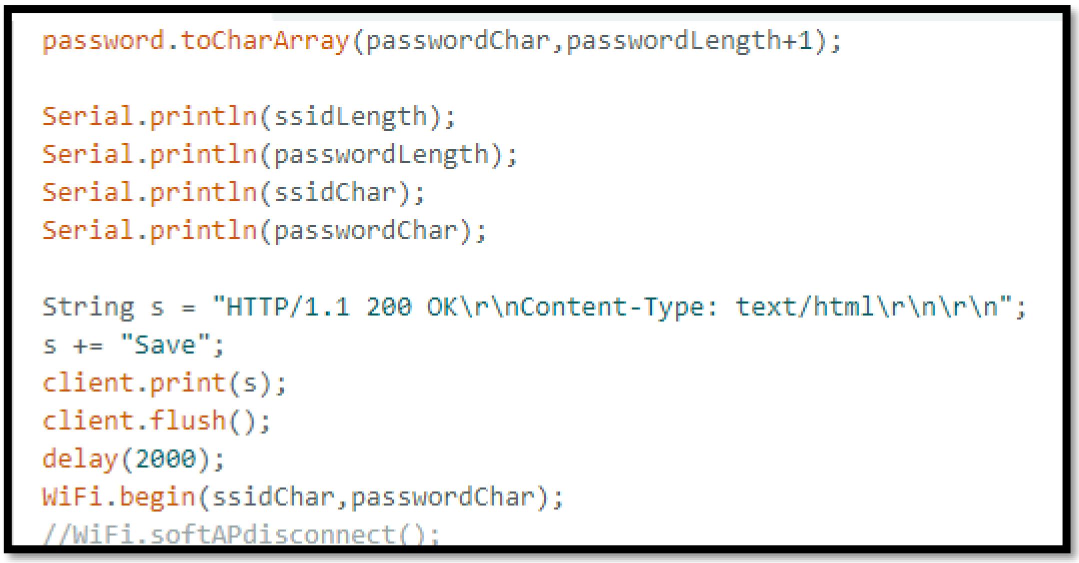
```
if (client)
{
    Serial.println("new client");
    int j = 0;
    int k = 1;
    String ssidTemp;
    String passwordTemp;
    String req = client.readStringUntil('\r');
    Serial.println(req);

    for (int i = 0; i < req.length(); i++)
    {
      if (req[i] == '(' && j == 0)
        j = 1;
      else if (req[i] == '(' && j == 1)
        {j=0; break;}
      else if (j == 1)
        {
         if(req[i]=='~')
           ssidTemp += ' ';
         else
           ssidTemp += req[i];
```



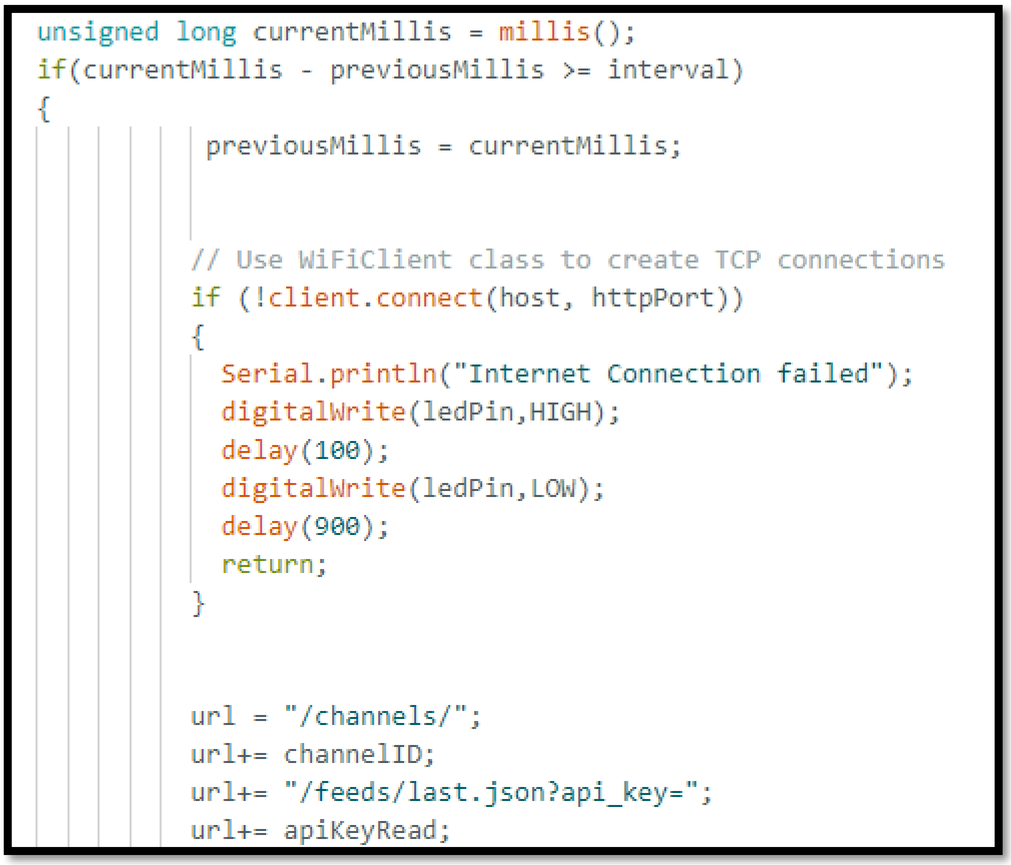
```
        }
  }

j=0;
for (int i = 0; i < req.length(); i++)
  {
    if (req[i] == ')' && j == 0)
      j = 1;
    else if (req[i] == ')' && j == 1)
      {j=0; break;}
    else if (j == 1)
      {
       if(req[i]=='~')
         passwordTemp += ' ';
       else
         passwordTemp += req[i];
      }
  }

  ssid=ssidTemp;
  password=passwordTemp;
```



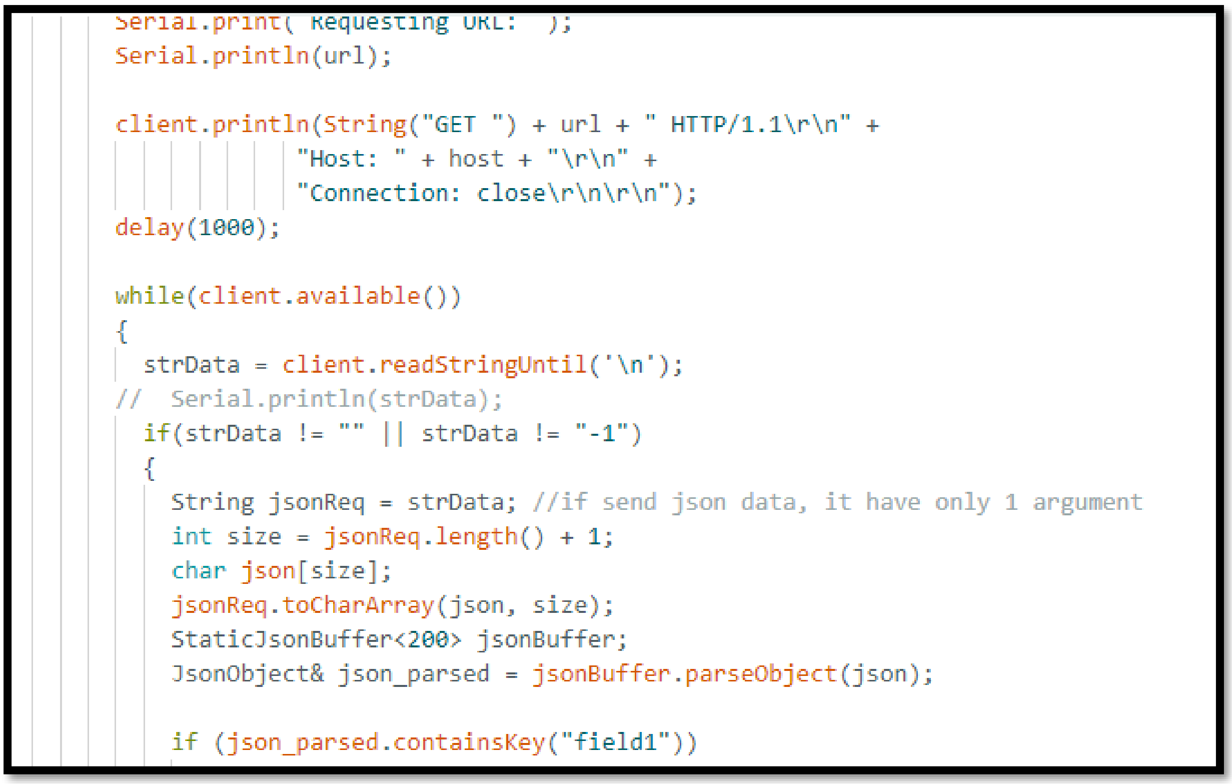
```
ssidLength=ssid.length();
passwordLength=password.length();

EEPROM.write(0,ssidLength);
delay(100);
EEPROM.write(30,passwordLength);

for(int i=0;i<ssidLength;i++)
{
  EEPROM.write(i+1,ssid[i]);
  delay(10);
}
for(int i=0;i<passwordLength;i++)
{
  EEPROM.write(i+31,password[i]);
  delay(10);
}
EEPROM.commit();

char ssidChar[ssidLength+1];
char passwordChar[passwordLength+1];
ssid.toCharArray(ssidChar,ssidLength+1);
```



```
password.toCharArray(passwordChar,passwordLength+1);

Serial.println(ssidLength);
Serial.println(passwordLength);
Serial.println(ssidChar);
Serial.println(passwordChar);

String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n";
s += "Save";
client.print(s);
client.flush();
delay(2000);
WiFi.begin(ssidChar,passwordChar);
//WiFi.softAPdisconnect();
```

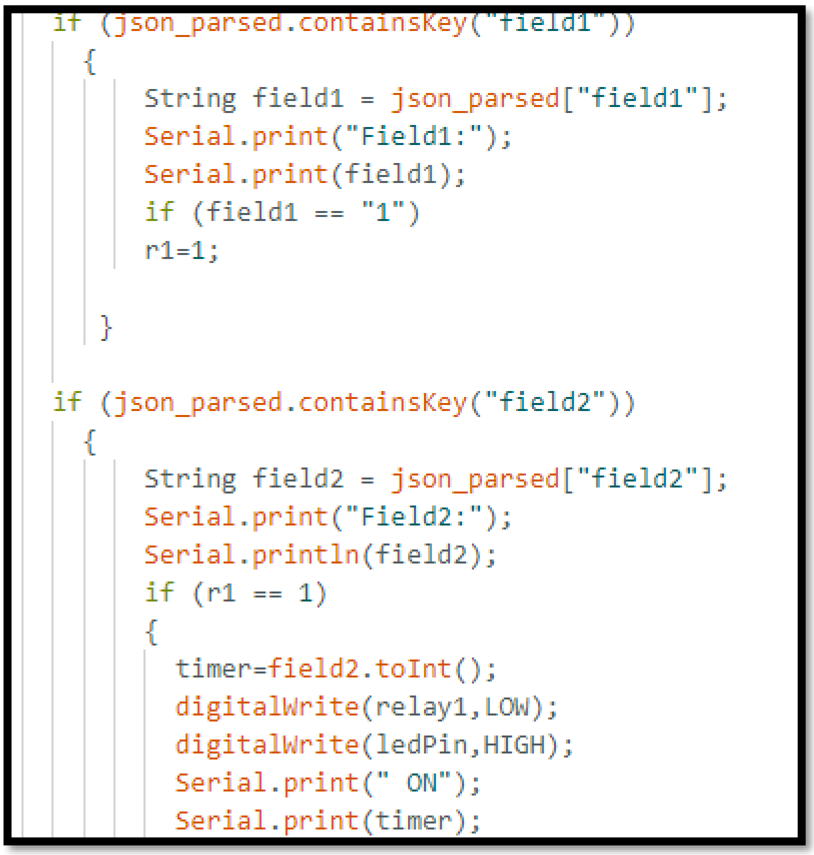


```
unsigned long currentMillis = millis();
if(currentMillis - previousMillis >= interval)
{
           previousMillis = currentMillis;


          // Use WiFiClient class to create TCP connections
          if (!client.connect(host, httpPort))
          {
            Serial.println("Internet Connection failed");
            digitalWrite(ledPin,HIGH);
            delay(100);
            digitalWrite(ledPin,LOW);
            delay(900);
            return;
          }


          url = "/channels/";
          url+= channelID;
          url+= "/feeds/last.json?api_key=";
          url+= apiKeyRead;
```

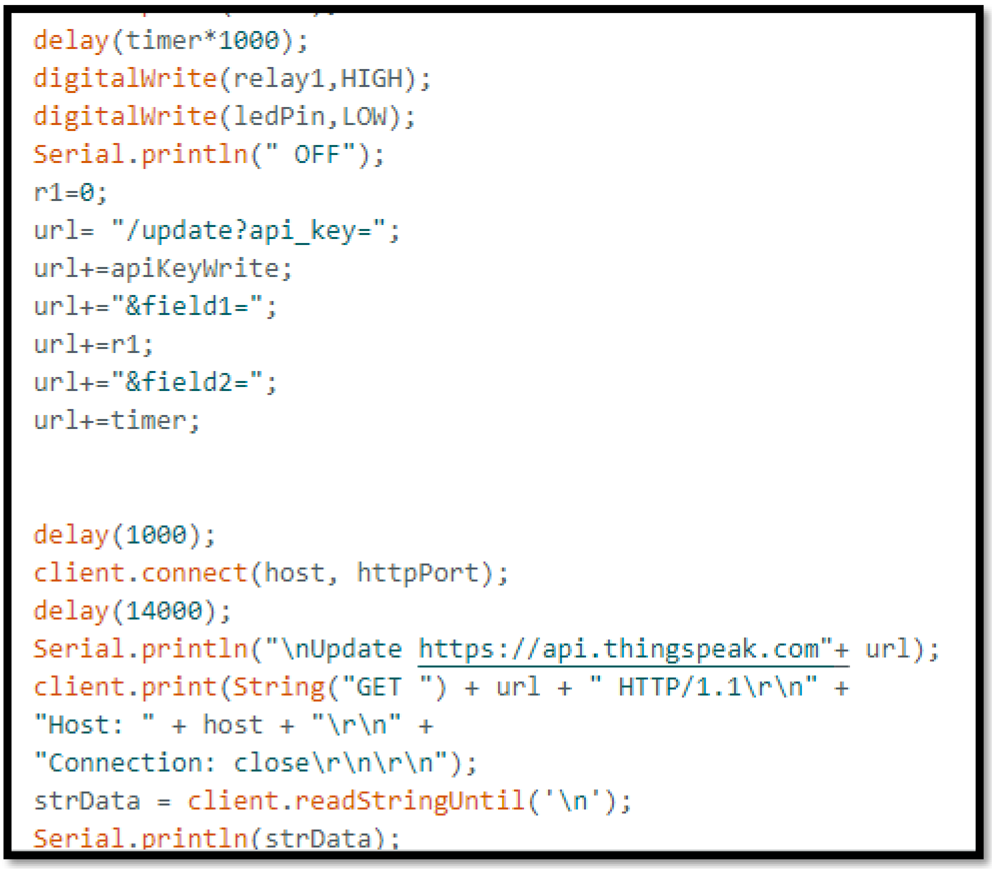


```
Serial.print("Requesting URL: ");
Serial.println(url);

client.println(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
delay(1000);

while(client.available())
{
  strData = client.readStringUntil('\n');
//  Serial.println(strData);
  if(strData != "" || strData != "-1")
  {
    String jsonReq = strData; //if send json data, it have only 1 argument
    int size = jsonReq.length() + 1;
    char json[size];
    jsonReq.toCharArray(json, size);
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject& json_parsed = jsonBuffer.parseObject(json);

    if (json_parsed.containsKey("field1"))
```

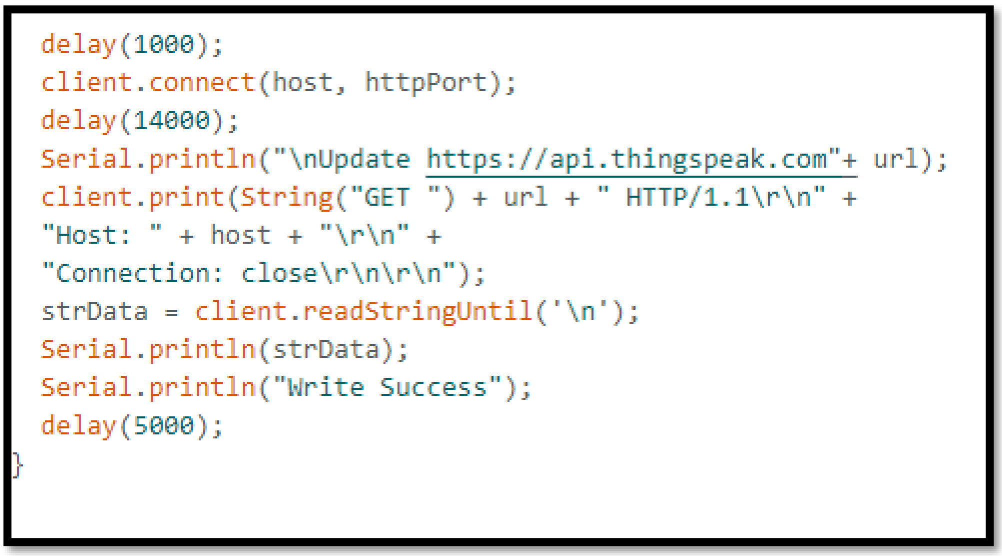
ON/OFF button and Timer button:



```
if (json_parsed.containsKey("field1"))
  {
      String field1 = json_parsed["field1"];
      Serial.print("Field1:");
      Serial.print(field1);
      if (field1 == "1")
      r1=1;

   }

if (json_parsed.containsKey("field2"))
  {
      String field2 = json_parsed["field2"];
      Serial.print("Field2:");
      Serial.println(field2);
      if (r1 == 1)
      {
        timer=field2.toInt();
        digitalWrite(relay1,LOW);
        digitalWrite(ledPin,HIGH);
        Serial.print(" ON");
        Serial.print(timer);
```

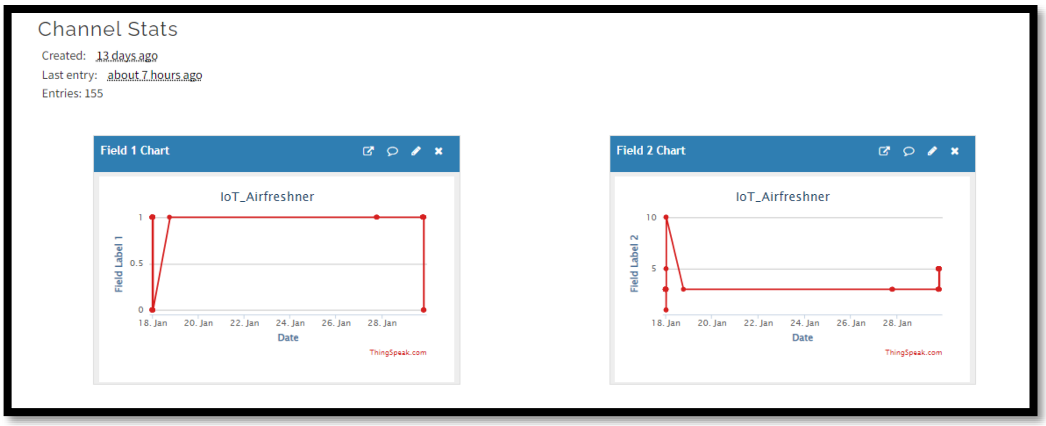


```
delay(timer*1000);
digitalWrite(relay1,HIGH);
digitalWrite(ledPin,LOW);
Serial.println(" OFF");
r1=0;
url= "/update?api_key=";
url+=apiKeyWrite;
url+="&field1=";
url+=r1;
url+="&field2=";
url+=timer;


delay(1000);
client.connect(host, httpPort);
delay(14000);
Serial.println("\nUpdate https://api.thingspeak.com"+ url);
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
"Host: " + host + "\r\n" +
"Connection: close\r\n\r\n");
strData = client.readStringUntil('\n');
Serial.println(strData);
```



```
 delay(1000);
 client.connect(host, httpPort);
 delay(14000);
 Serial.println("\nUpdate https://api.thingspeak.com"+ url);
 client.print(String("GET ") + url + " HTTP/1.1\r\n" +
 "Host: " + host + "\r\n" +
 "Connection: close\r\n\r\n");
 strData = client.readStringUntil('\n');
 Serial.println(strData);
 Serial.println("Write Success");
 delay(5000);
}
```

3. Think Speak for Data Analytics:



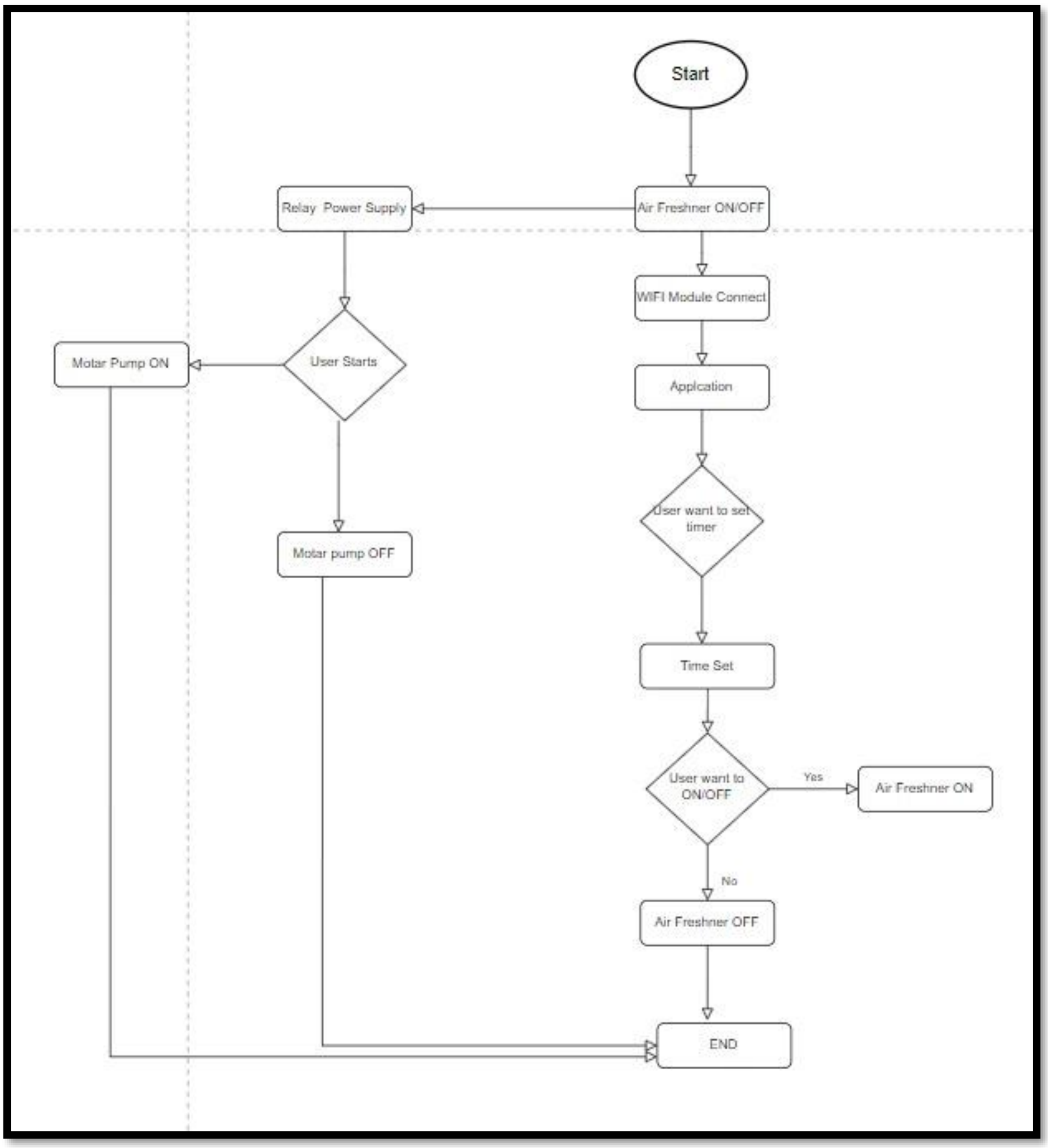
Channel Stats

Created: 13 days ago

Last entry: about 7 hours ago

Entries: 155

Block Diagram:



Cost of Development

ITEMS	**QUANTITY**	**COST (RS)**
Vero Board	1	80
Female/ Male Header	1	30
Relay Module	1	150
Adapter	1	200
Motor Pump	1	190
Esp-32	1	1000
Buck Converter	1	200
Total		**1850**