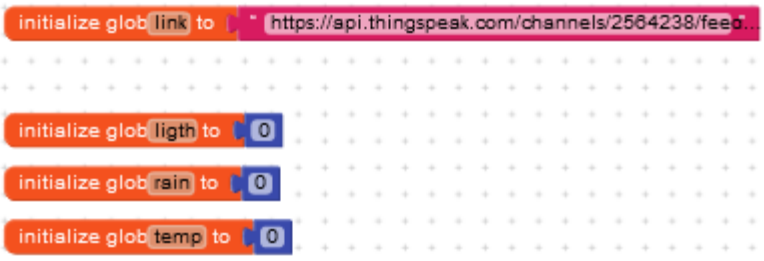
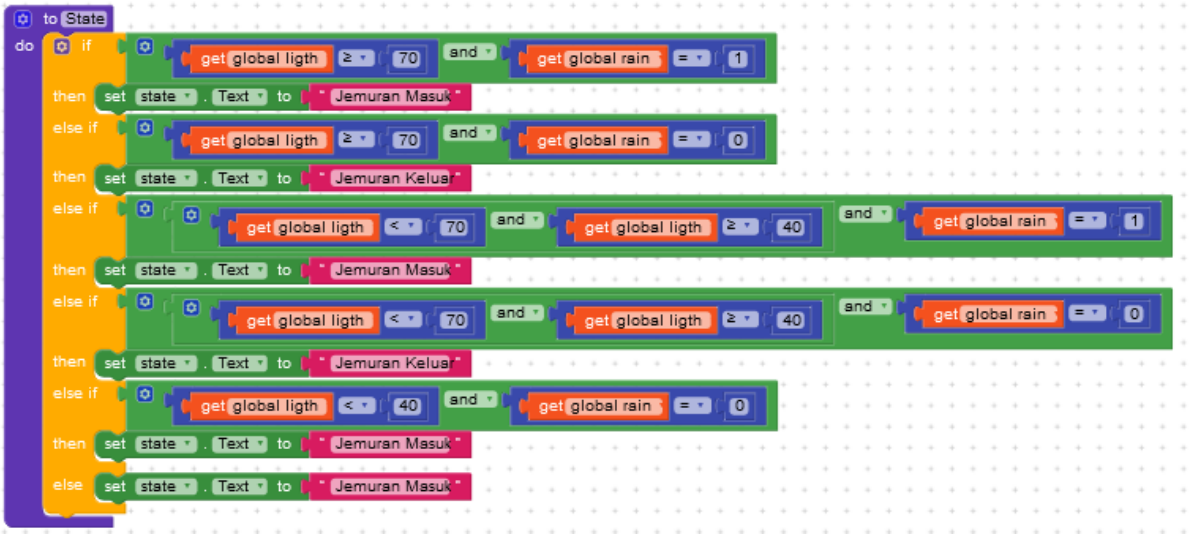
LAMPIRAN SCRIPT

1. Gambar Screen Monitoring

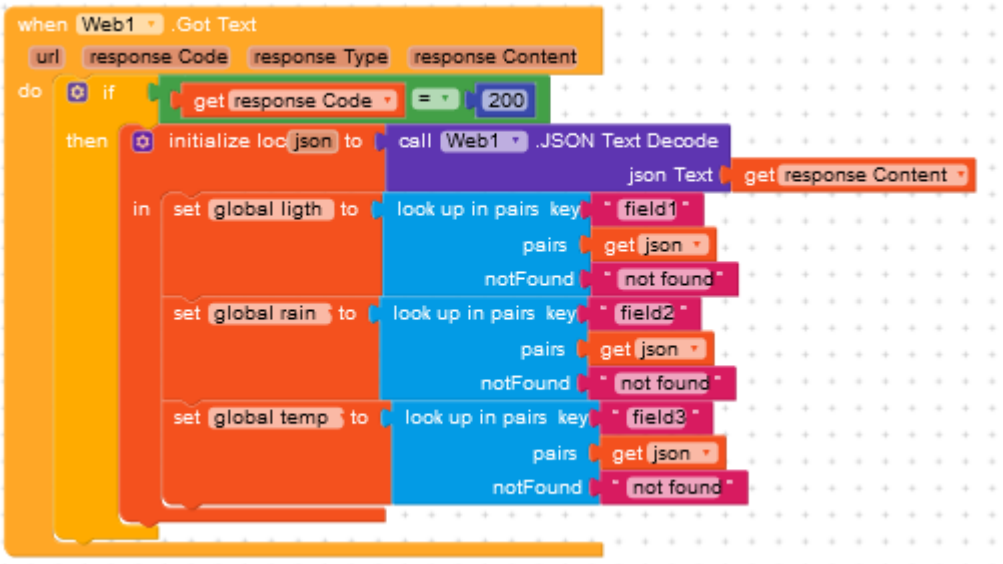
2. Code inisialisasi variabel link yang berisi url untuk read data ThingSpeak, variabel light, rain, dan temp untuk menampung nilai data dari setiap field ThingSpeak.



3. Code fungsi State untuk menampung logika yang akan menampilkan keterangan apakah jemuran masuk atau keluar dengan menggunakan 9 logika berikut.

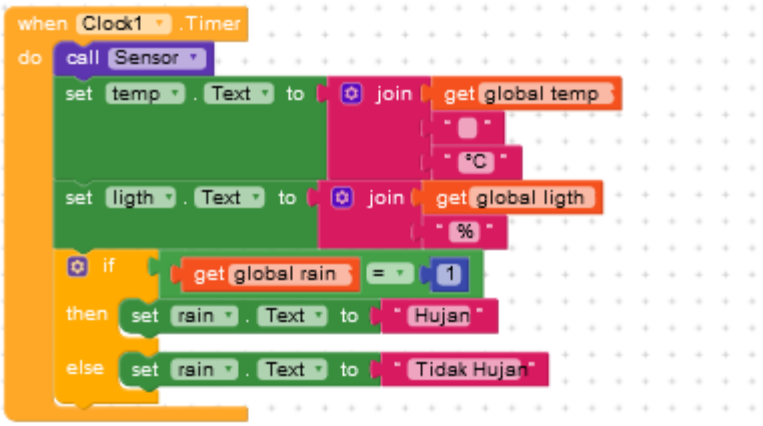


4. Code Web1.Got Text berfungsi untuk mengambil data dari setiap field ThingSpeak menggunakan Web1.Got Text, yang kemudian ditampung di variabel light, rain dan temp.



5. Code fungsi Sensor yang berisi pemanggilan fungsi Web1.Get, State, dan setup Web1.URL dengan url read data ThingSpeak.

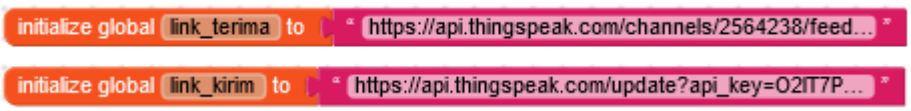
6. Code Clock1.Timer berisi pemanggilan fungsi Sensor, setup label nilai sensor seperti Temperatur (°C), Light (%) dan Rain (Hujan atau Tidak).



7. Code control.Click berfungsi untuk melakukan perpindahan dari screen Monitoring ke Screen Controlling Fan.

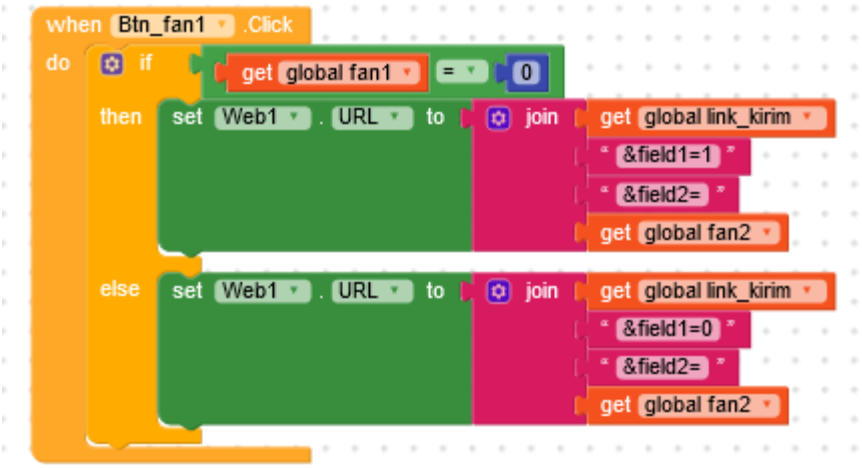
8. Gambar Screen Controlling Fan

9. Code inisialisasi variabel link_terima yang berisi url untuk read data dari ThingSpeak, dan variabel link_kirim berisi url untuk write data ke ThingSpeak.

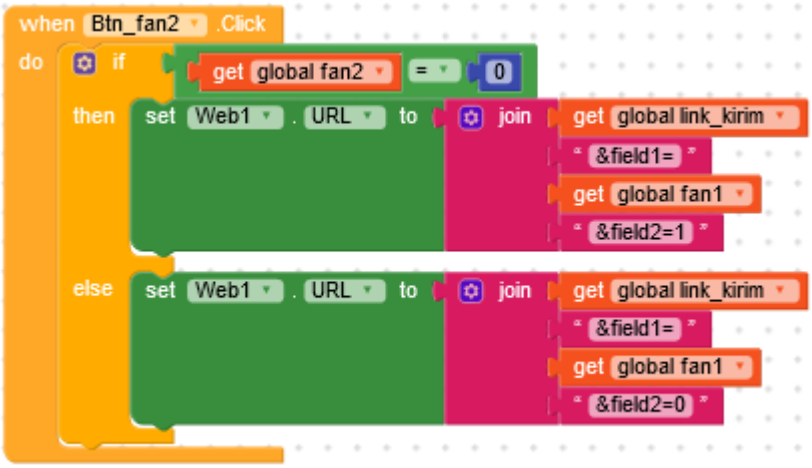


10. Code inisialisasi variabel fan1 dan fan2 untuk menamping nilai dari field ThingSpeak.

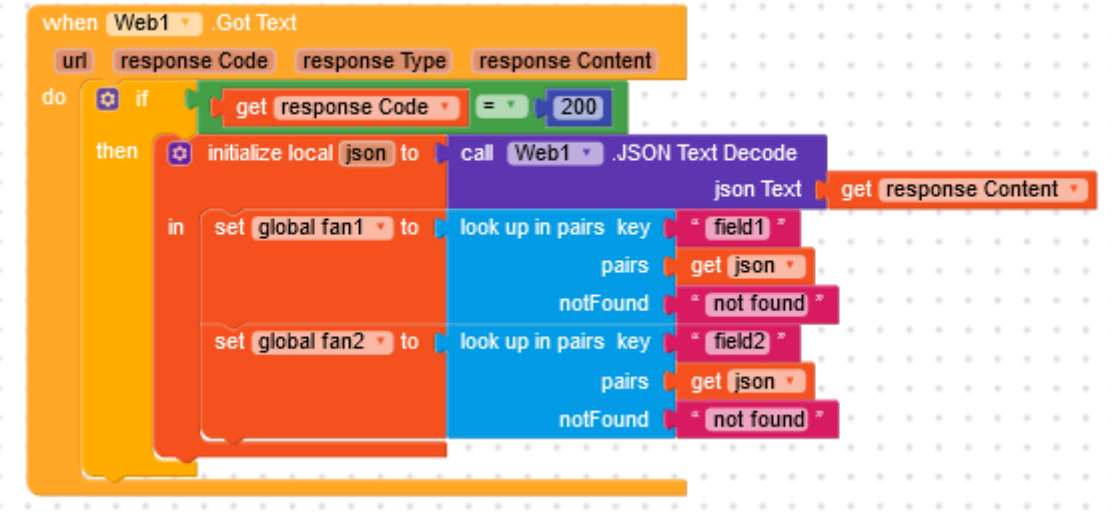
11. Code Btn_fan1 berfungsi untuk mengirimkan data ke ThingSpeak dengan ketentuan, jika nilai field fan1 = 0 maka field1 = 1 dan field2 = data terakhir, dan jika field fan1 = 1 maka field1 = 0 dan field2 = data terakhir.



12. Code Btn_fan1 berfungsi untuk mengirimkan data ke ThingSpeak dengan ketentuan, jika nilai field fan2 = 0 maka field1 = data terakhir dan field2 =1, dan jika field fan2 = 1 maka field1 = data terakhir dan field2 = 0.

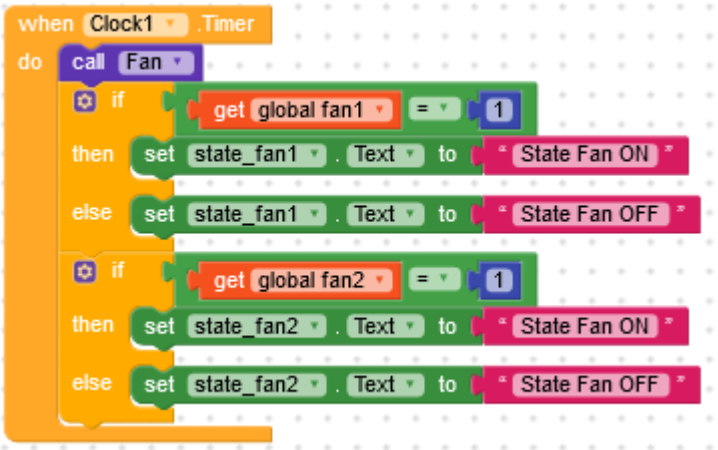


13. Code Web1.Got Text berfungsi untuk mengambil data dari setiap field ThingSpeak menggunakan Web1.Got Text, yang kemudian ditampung di variabel fan1 dan fan2.
14.



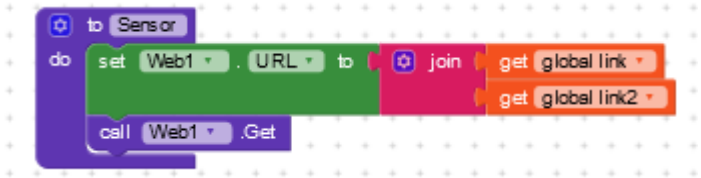
15. Code fungsi Fan berisi pemanggilan fungsi Web1.Get dan setup Deb1.URL dengan url read data dari ThingSpeak.

16. Code Clock1.Timer berisi pemanggilan fungsi Fan, setup label fan1 maupun fan2 yang kemudian jika fan1 atau fan2 memiliki nilai sama dengan 1 maka setup label state_fan menjadi State Fan ON, dan jika fan1 atau fan2 memiliki nilai sama dengan 0 maka setup label state_fan menjadi State Fan OFF.

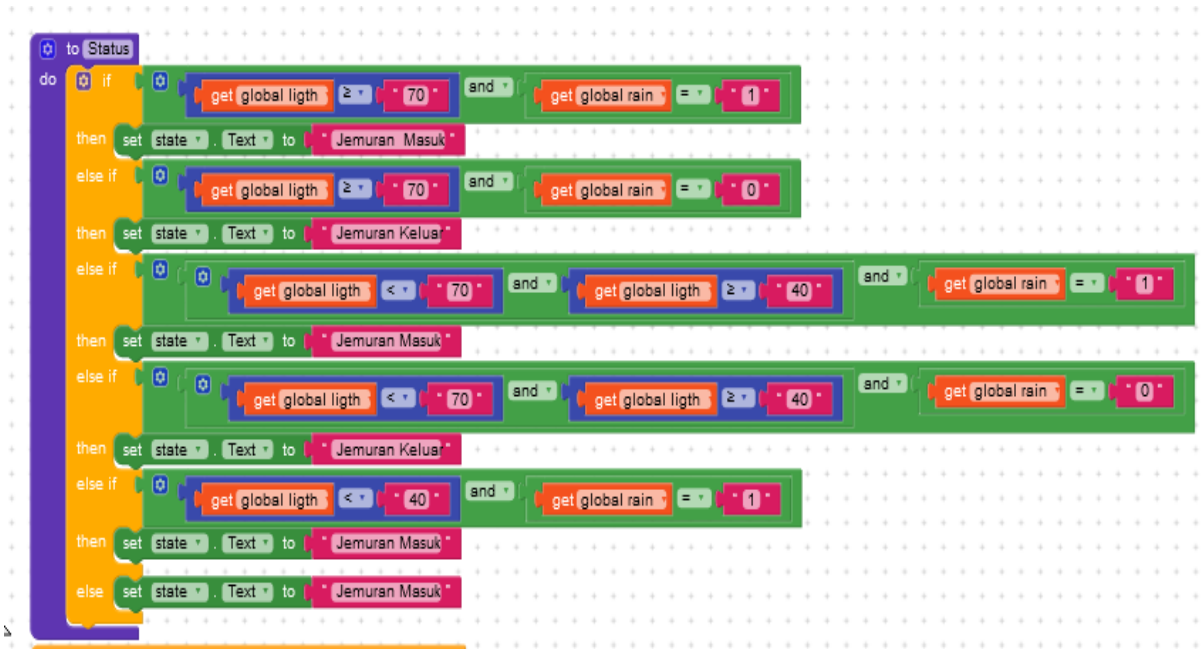


17. Code Btn_back.Click berfungsi untuk melakukan perpindahan dari screen Controlling Fan ke screen Monitoring (kembali ke tampilan awal).

Inisialisasi variabel untuk menampung nilai dari sensor suhu (DS18B20), sensor cahaya (LDR), dan sensor hujan.

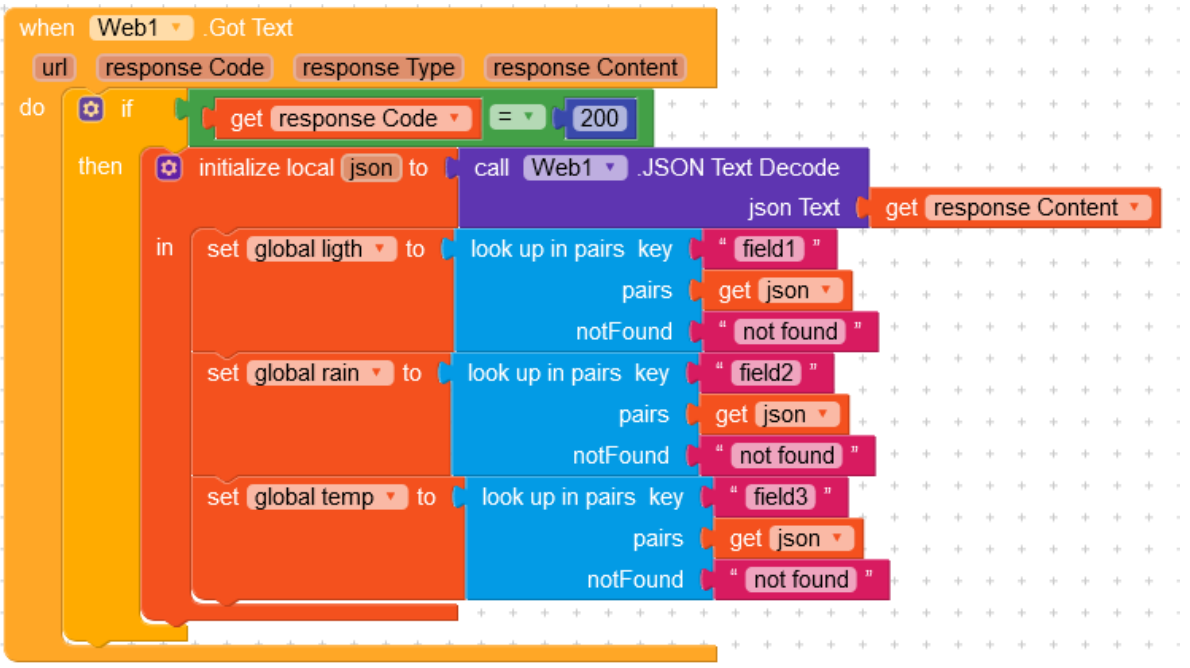


Membuat prosedur dengan nama Sensor untuk menampung kode sumber dari database ThingSpeak dan memanggil data dari method Web1.Get

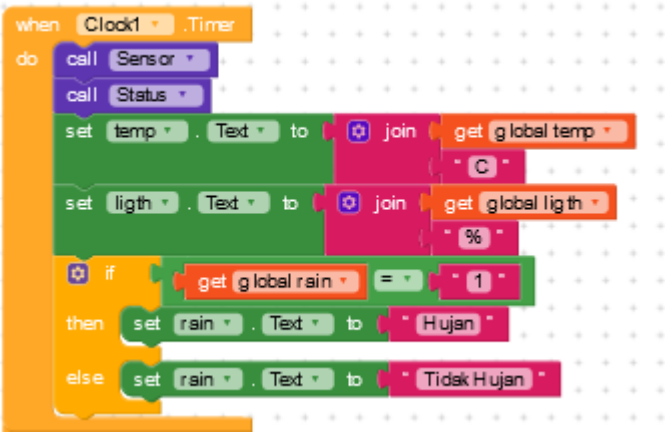


Prosedur Sensor berisi logika untuk menampilkan status yang di mana logika ke;

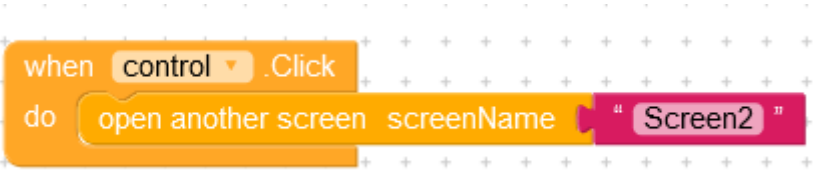
1. Jika nilai Cahaya ≥ 70 dan terjadi hujan, maka status “Jemuran Masuk” .
2. Jika nilai Cahaya ≥ 70 dan tidak terjadi hujan, maka status “Jemuran Keluar”.
3. Jika nilai Cahaya < 70 dan nilai Cahaya ≥ 40 serta terjadi hujan, maka status “Jemuran Masuk”.
4. Jika nilai Cahaya < 70 dan nilai Cahaya ≥ 40 serta tidak terjadi hujan, maka status “Jemuran Keluar”.
5. Jika nilai Cahaya < 40 dan terjadi hujan, maka status “Jemuran Masuk”.
6. Jika nilai Cahaya < 40 dan tidak terjadi hujan, maka status “Jemuran Masuk”.



Method Web1.GetText berisi sekumpulan perintah untuk mengambil data dari setiap field (variabel tampungan) yang sudah dibuat di dalam database ThingSpeak, seperti field1 untuk menampung nilai sensor Cahaya, field2 untuk menampung nilai sensor Hujan, dan field3 untuk menampung nilai sensor Suhu (DS18B20).



Method Clock.Timer berfungsi untuk menampilkan data secara realtime atau tepat waktu sesuai data terakhir yang ditampilkan dari database ThingSpeak. Dengan beberapa ketentuan yakni jika nilai hujan = 1 maka keterangannya adalah Hujan, dan jika hujan = 0 maka keterangannya adalah Tidak Hujan. Dan



Methode control.Cliick berfungsi untuk melakukan perpindahan tampilan layer dari *Screen Monitoring* ke *Screen Controlling*

CODE THINGSPEAK

```
#include <WiFi.h>
#include "secrets.h"
#include "ThingSpeak.h" // always include thingspeak header file after other header files
        and custom macros

char ssid[] = "speedy4G";   // your network SSID (name)
char pass[] = "Ghony@2010";   // your network password
int keyIndex = 0;            // your network key Index number (needed only for WEP)
WiFiClient  client;

// Channel RainyDay
unsigned long myChannel_1 = 2555364;
const char * myWriteAPIKey_1 = "PZ04L6PWHBXJVB6X";
// Channel Control Fan
unsigned long myChannel_2 = 2556077;
const char * myWriteAPIKey_2 = "KYFVLGQI863XG7DK";
const char * myReadAPIKey_2 = "3YWII4ST6AFQ4T3B";

#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 14
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature Sensor_suhu(&oneWire);

const int
Sensor_LDR    = 33,
Sensor_Hujan  = 12,
```

```
Relay_1       = 23,
Relay_2       = 22,
M1            = 32,
M2            = 13;

int fan1, fan2;
bool overrideFan1 = false,
overrideFan2 = false;
String myStatus   = "",
StatusBaju = "";

const int micro1 = 18,
micro2 = 19;

void setup() {
Serial.begin(115200);  //Initialize serial
while (!Serial) {
; // wait for serial port to connect. Needed for Leonardo native USB port only
}

// pinMode(Sensor_LDR, INPUT);     // VCC 3.3V
pinMode(Sensor_Hujan, INPUT);
Sensor_suhu.begin();

pinMode(Relay_1, OUTPUT);
pinMode(Relay_2, OUTPUT);

pinMode(M1, OUTPUT);
pinMode(M2, OUTPUT);
```

```
pinMode(micro1, INPUT_PULLUP);
pinMode(micro2, INPUT_PULLUP);

WiFi.mode(WIFI_STA);
WiFi.begin(ssid, pass);
ThingSpeak.begin(client);  // Initialize ThingSpeak
}
void Maju(){
bool micro = digitalRead(micro1);
Serial.print("Micro 1 : ");
Serial.println(micro);
if(micro == HIGH){
digitalWrite(M1, LOW);
digitalWrite(M2, LOW);
}else{
digitalWrite(M1, HIGH);
digitalWrite(M2, LOW);}
delay(500);
}
void Mundur(){
bool micro = digitalRead(micro2);
Serial.print("Micro 2 : ");
Serial.println(micro);
if(micro == HIGH){
digitalWrite(M1, LOW);
digitalWrite(M2, LOW;
}else{
digitalWrite(M1, LOW);
digitalWrite(M2, HIGH);
```

```
delay(500);}
}
void loop() {

// Connect or reconnect to WiFi
if(WiFi.status() != WL_CONNECTED){
Serial.print(".");
delay(500);
}else{
Serial.println("\nConnected.");

// Setup variabel untuk menampung nilai sensor
float hujan   = digitalRead(Sensor_Hujan);
int Suhu      = Sensor_suhu.getTempCByIndex(0);
int n_cahaya  = analogRead(Sensor_LDR);
int Cahaya      = map(n_cahaya, 0, 4095, 100, 0);
// 0-40 = Gelap, 41-70 = Teduh, 71-100 = Cerah
// 0-40 = Kering, 41-70 = Lembab, 71-100 = Basah
// hujan = 0, tidak hujan = 1

Serial.print("Hujan : ");
Serial.println(hujan);
Serial.print("Suhu : ");
Serial.println(Suhu);
Serial.print("Cahaya : ");
Serial.println(Cahaya);

// membaca field fan1 dan fan2
float field1 = ThingSpeak.readFloatField(myChannel_2, 1);
```

```
float field2 = ThingSpeak.readFloatField(myChannel_2, 2);
// Serial.print("Field 1: ");
// Serial.println(field1);
// Serial.print("Field 2: ");
// Serial.println(field2);

// Check override dari thingspeak
if(field1 == 0 && field2 == 0){
overrideFan1 = false;
overrideFan2 = false;
fan1 = 0;
fan2 = 0;
Serial.println("Override OFF");
}else if(field1 == 1 && field2 == 0){
overrideFan1 = true;
overrideFan2 = false;
fan1 = 1;
fan2 = 0;
Serial.println("Override Fan 1 : ON");
}else if(field1 == 0 && field2 == 1){
overrideFan1 = false;
overrideFan2 = true;
fan1 = 0;
fan2 = 1;
Serial.println("Override Fan 2 : ON");
}else if(field1 == 1 && field2 == 1){
overrideFan1 = true;
overrideFan2 = true;
fan1 = 1;
```

```
fan2 = 1;
Serial.println("Override Fan 1 & Fan 2 : ON");
}

if (!overrideFan1 || !overrideFan2){
if(hujan == 0 && Cahaya >= 71 ){
StatusBaju = "Jemuran Masuk";
Serial.println(StatusBaju);
Mundur();
}else if (hujan == 0 && (70 > Cahaya >=41)){
StatusBaju = "Jemuran Masuk";
Serial.println(StatusBaju);
Mundur();
}else if (hujan == 0 && Cahaya <=40){
StatusBaju = "Jemuran Masuk";
Serial.println(StatusBaju);
Mundur();
}else if(hujan == 1 && Cahaya >= 71 ){
StatusBaju = "Jemuran Keluar";
Serial.println(StatusBaju);
Maju();
}else if (hujan == 1 && (70 > Cahaya >=41)){
StatusBaju = "Jemuran Keluar";
Serial.println(StatusBaju);
Maju();
}else if (hujan == 1 && Cahaya <=40){
StatusBaju = "Jemuran Masuk";
Serial.println(StatusBaju);
Mundur();
```

```
}
}

digitalWrite(Relay_1, fan1);
Serial.print("Fan1 : ");
Serial.println(fan1);
digitalWrite(Relay_2, fan2);
Serial.print("Fan2 : ");
Serial.println(fan2);

ThingSpeak.setField(1, Cahaya);
ThingSpeak.setField(2, hujan);
ThingSpeak.setField(3, Suhu);
// write to the ThingSpeak channel
int x = ThingSpeak.writeFields(myChannel_1, myWriteAPIKey_1);
if(x == 200){
Serial.println("Channel update successful.");
}
else{
Serial.println("Problem updating channel. HTTP error code " + String(x));
}

ThingSpeak.setField(1, fan1);
ThingSpeak.setField(2, fan2);
// write to the ThingSpeak channel
int y = ThingSpeak.writeFields(myChannel_2, myWriteAPIKey_2);
if(y == 200){
Serial.println("Channel update successful.");
}
```

```
else{
Serial.println("Problem updating channel. HTTP error code " + String(y));
}

delay(500);  // Wait 20 seconds to update the channel again
}
}
```