Self-Sustained Smart House

Hamad Almesmari, Electrical Engineering

Project Advisor: Dr. Mohsen Lotfalian

April 26, 2019

Evansville, Indiana

Table of content

List of Figures

List of Tables

Costs

1- Project cost

Items	Price per unit ($)
House model	35
Raspberry Pi 3 B+	38.10
Sensors	45.74
Solar panels * 16	3.398
Rechargeable batteries * 3	22.99
Charge controller	14.99
LED lights	15.99
INA 219 current/voltage sensor*3	12.55
relay module	11.99
ESP8266 WiFi module * 3	14.99
ADS 1115	6.99
Total	374.67

2- Real house automation cost

Items	Price per unit ($)
Raspberry Pi 3 B+	38.10
Sensors	28.97
Soil moisture sensor	7.99
Water level sensor	10.00
INA 219 current/voltage sensor	12.55
relay module	11.99
ESP8266 WiFi module * 3	14.99
ADS 1115	6.99
Total	139.075

3- Real house PV system cost

a. Average household power consumption = 32KW/day
b. Daily load: 3189.61 Ah/day
c. PV cell: 105 cells 100w/12v each
d. Battery size: 9764 Ah for 3 days
e. Invertor size: 2000 Watt

Items	Price ($)
Solar panels	3892.95
Battery	3490.00
Charge controller MPPT	158.87
Power invertor	294.54
Total	7836.36

I. Introduction

In an over populated planet, we are in an urgent need for solutions to preserve the environment and enhance our way of living. This project revolves around two main parts that help preserve the environment and enhance our lives. It combines clean energy with home automation to maximize efficiency and make life easier for individuals while keeping the cost to minimum. It provides the three main needs: food, water, and electricity. It uses sensors to monitor the environment and use the feedback to increase productivity and lower people's consumption of electricity and water. The ultimate purpose is lowering the cost of smart houses and make it accessible to the average consumer.

II. Things to consider

The main problem is the high cost of the materials used in the most accessible clean energy systems and smart gadgets, which makes using it seems counter intuitive. However, if we lower the cost and maximize the outcome it will be worth it in the short term and the long term as well. The secondary problem is connection and communication between gadgets and the control center. Whether we are using wireless or wired communication and the pros and cons of each type. On one hand, wireless communication is easier to establish, it doesn't require any modification to infrastructure, and is faster to install. On the other hand, it can be affected by the environment that is electromagnetic fields, walls, and other nearby devices and can be hacked. In contrast, wired connection is reliable, does not get interference from nearby devices, and harder to hack from the outside. However, it does require some serious infrastructure modification.

III. Client requirements

The client requested a solution to automate the house that will be easy to install and modify in the future. It must be cost effective and applicable to new houses and old houses alike. Finally, it must maximize efficiency and minimize power consumption.

IV. Design approach

a. Sustainability

Used solar power and battery storage to generate and store electricity. 16 UXcell solar panels 5.5V 180mA (amazon.com /UXcell) to generate the power needed for the design (figure 1). It is connected in parallel and series to meet the power requirements for the charge controller and battery. For the charge controller, a 10A Solar Charge Controller from “PowMr” 12V/24V Auto Parameter (figure 2). A 12 Volt “eDRENALINE” by “Wild Game Innovations” (amazon.com/ Wild Game Innovations) is used to store power for later use and allow the model to run in low light environment (figure 3). Battery type “12 Volt 9 Amp 20 Hour Sealed Lead Acid Battery with F2 Style Terminals” (amazon.com). Adafruit INA219 current and voltage sensor (adafruit.com/product/904) is used to monitor power generation, consumptions and battery level. (figure 4). A power inverter is used to convert power from DC to AC (amazon.com) (figure 5) By BESTEK 300W DC 12V to 110V (amazon.com). Figure 6 shows the setup for the photovoltaic system.

Figure 1: UXCell 5.5v 180m (amazon.com /UXcell)

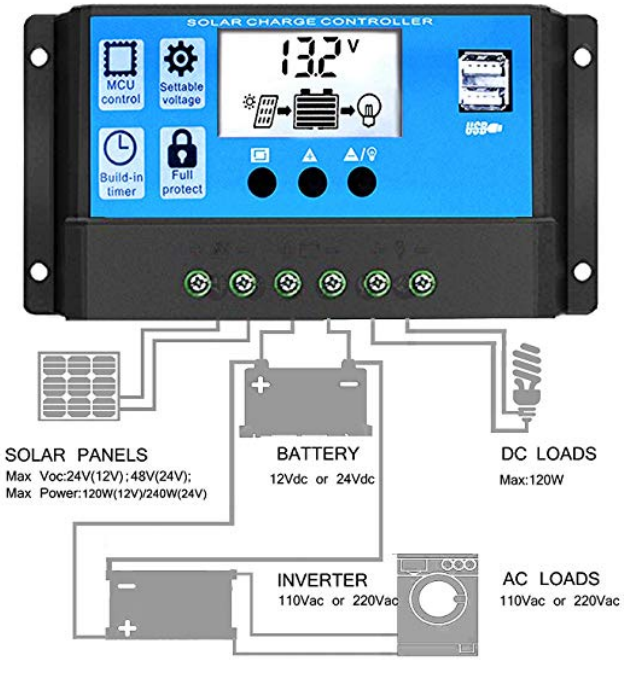

Figure 2: charge controller by PowMr. 10A 12v/24v (amazon.com)

Figure 3: 12v/24v 9Ah battery (amazon.com/ Wild Game Innovations)

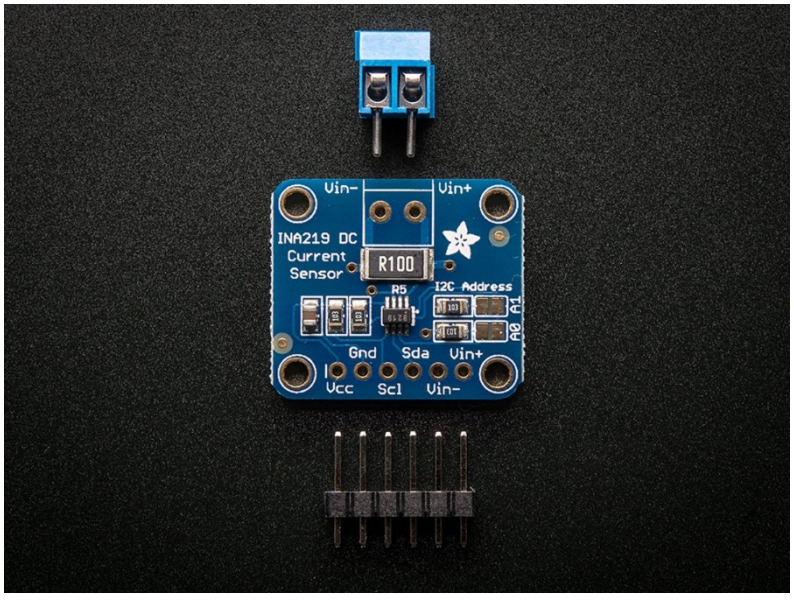

Figure 4: INA219 by Adafruit (adafruit.com/product/904)

Figure 5: power inverter By BESTEK 300W DC 12V to 110V (Amazom.com)

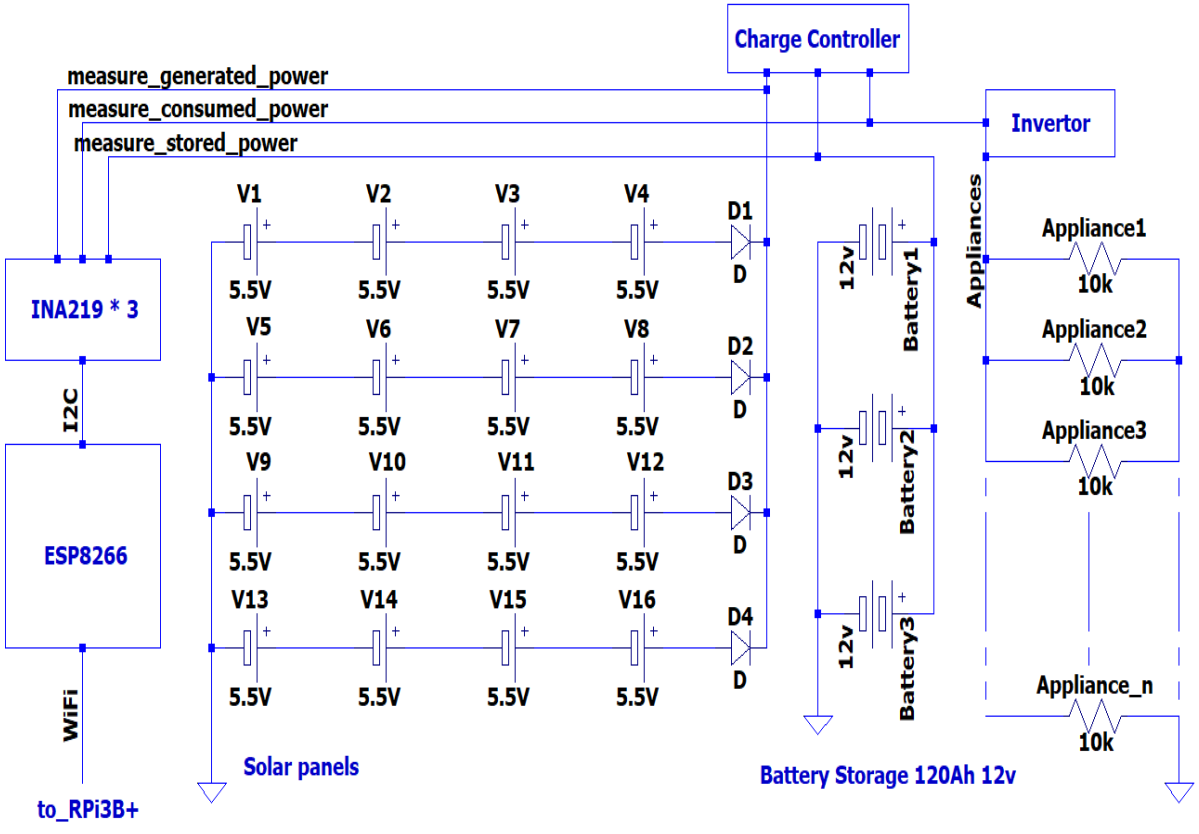

Figure 6: photovoltaic system setup

b. Automation

The model is using a computer raspberry pi 3b+(amazon.com) (figure 7) to control the house through real-time feedback from sensors. Also, it provides internet connection via ESP8266 Wi-Fi module (Amazon.com) (figure 8) and a local hub (Raspberry Pi 3b+) for secure communication and fast control. Through automation, appliances turn on when people are around and off when no one is around, conserving energy. The system alerts the users of excess usage of resources like showering for too long or keeping the lights on when no one is home. Safety wise, a local hub provides the ultimate protection because for anyone to hack it they need to enter the house and connect to the system wirily which eliminates the risk of phishing scheme and hackers. It provides control even if there is no connection to the internet. A combination of wired and wireless connection provides flexibility for the user to control everything close to the and far from them.

Figure 7: Raspberry Pi 3B+ (amazon.com)

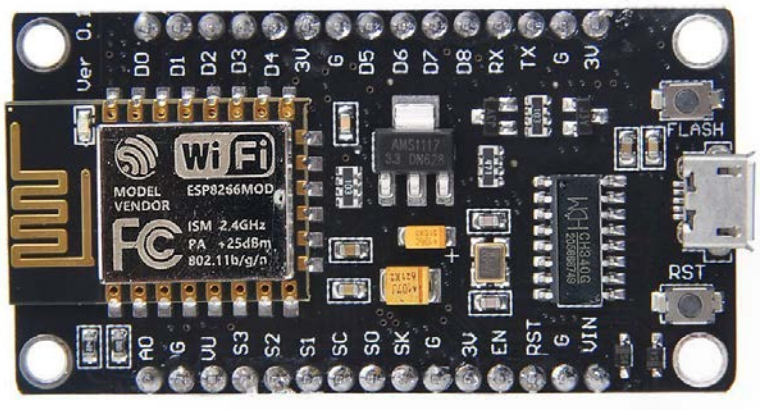

Figure 8: ESP8266 Wi-Fi model (amazon.com)

The design will have two main parts, software and hardware. For the software I used Python, C language and json.js to write the code. The user interface allows the user to control the environment from the raspberry pi touchscreen using Node-Red and TKinter. GUI and the code are in figures 9-15.

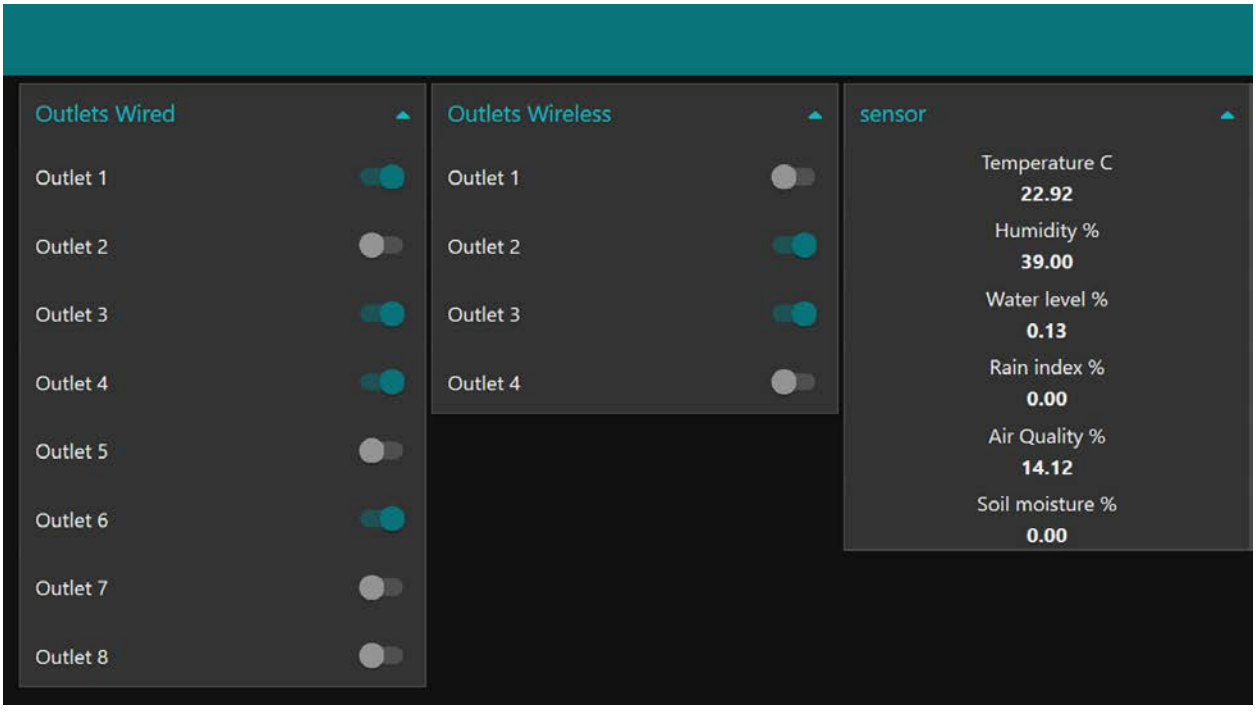

Figure 9: GUI design to control outlets

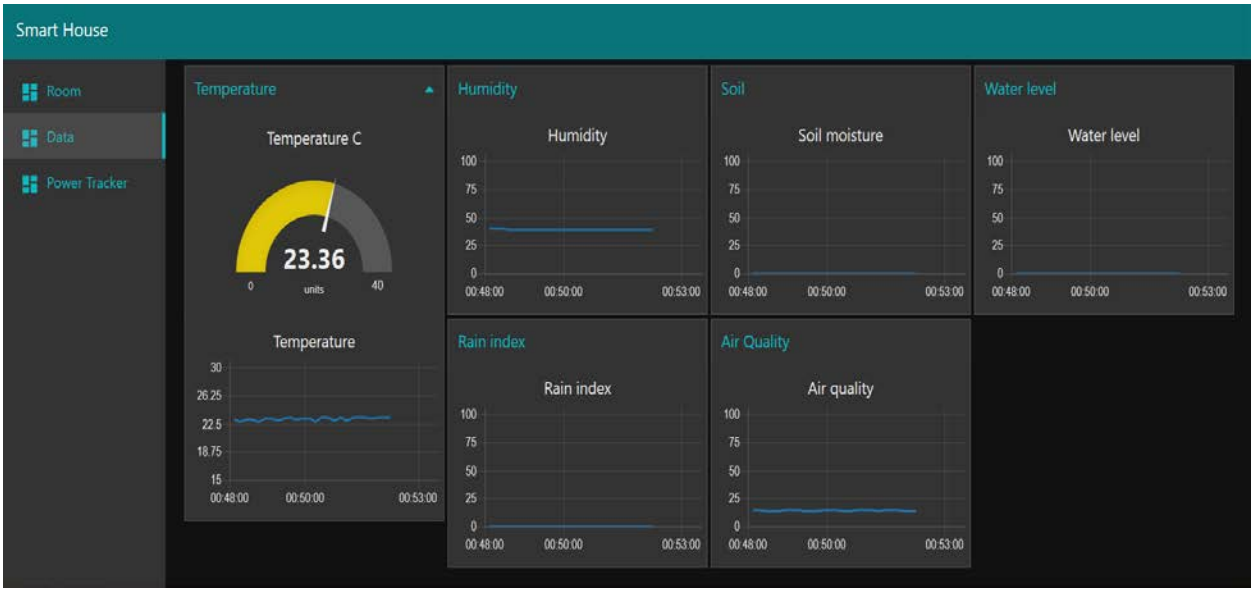

Figure 10 Sensors feedback over time

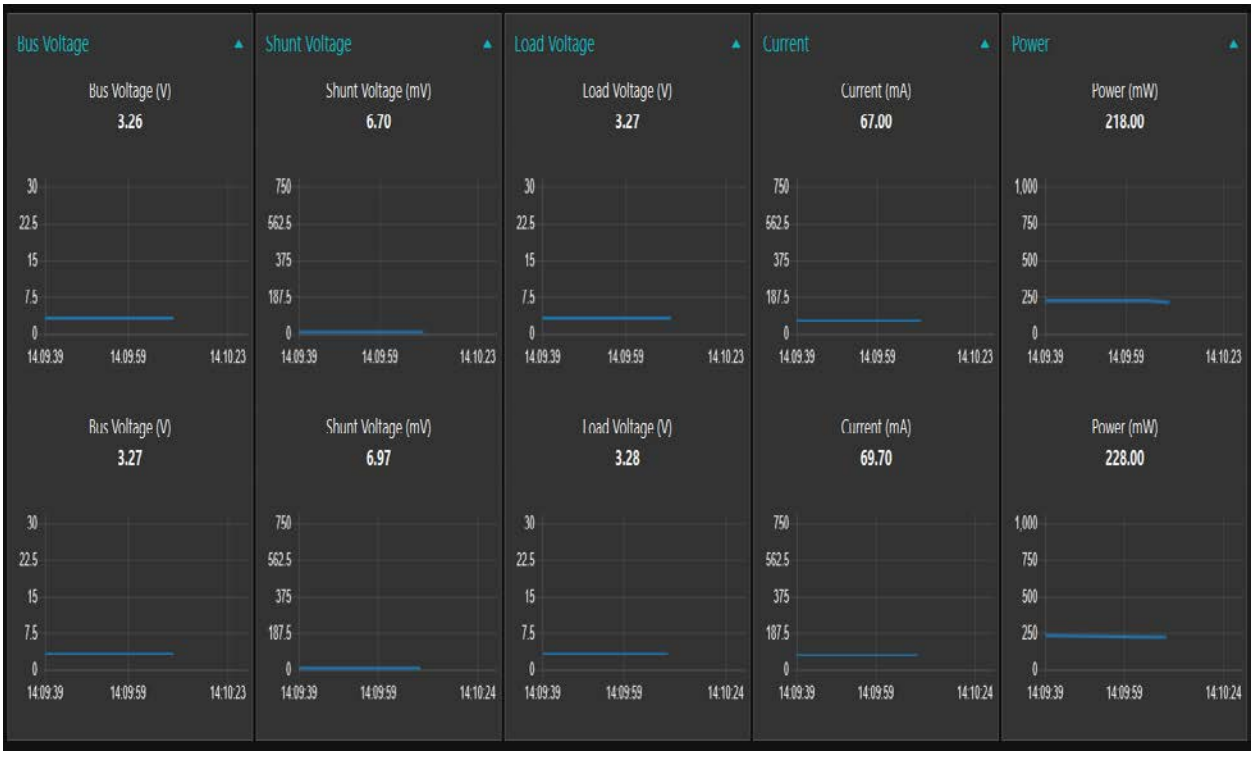

Figure 11: Power tracker from the PV system

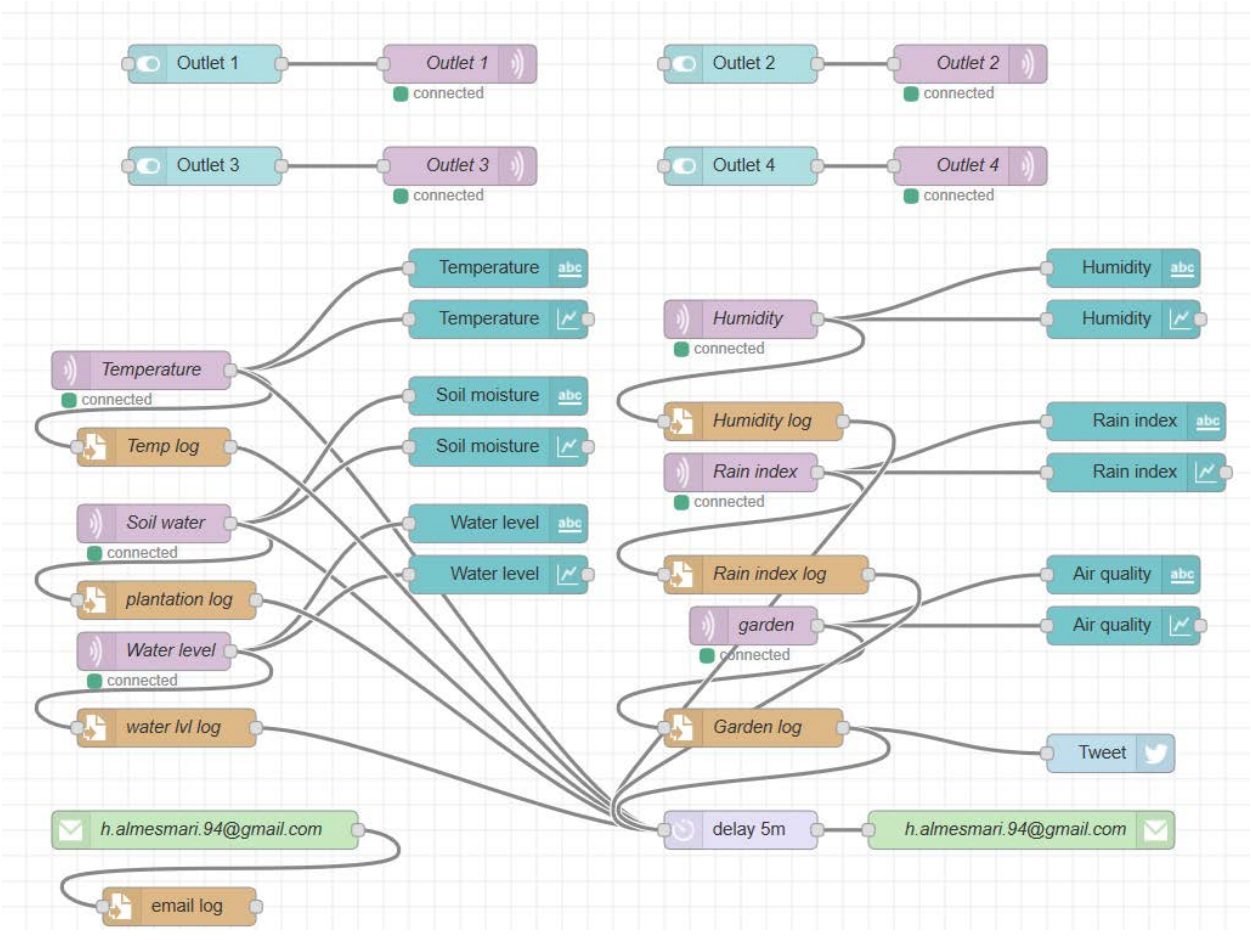

Figure 12 Node-Red flow design for wireless connection

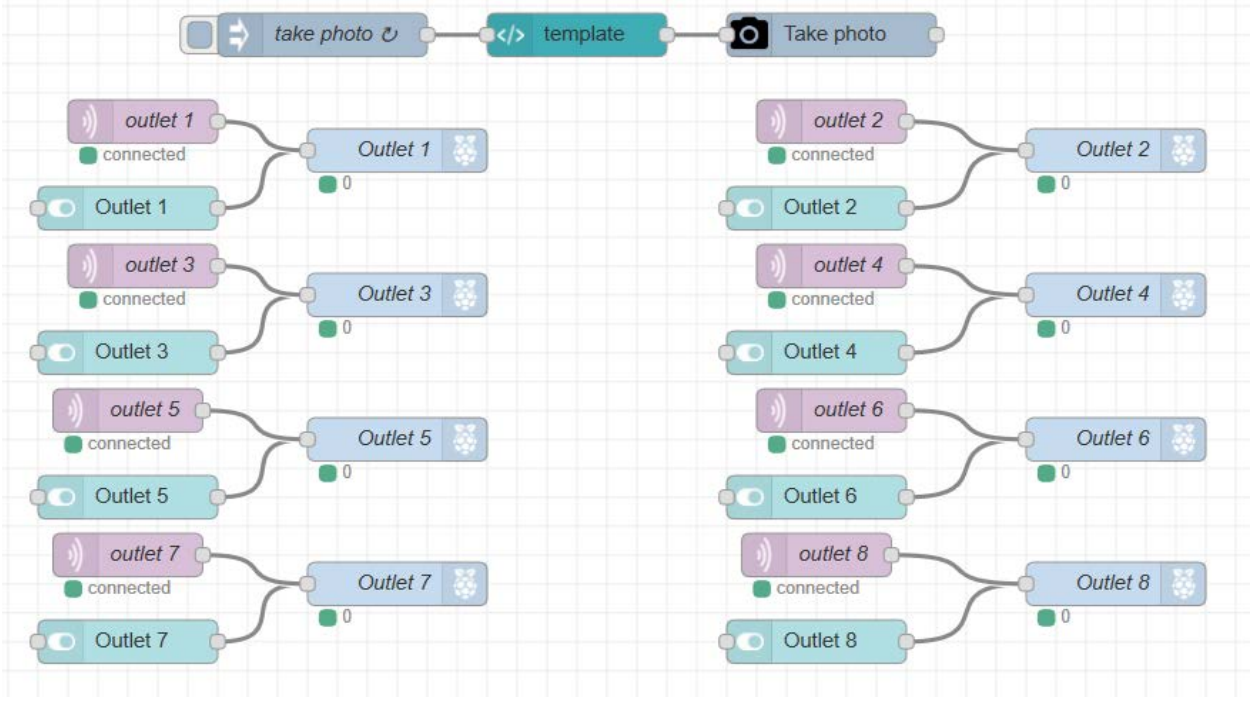

Figure 13 Node-Red Flow design for wired connection

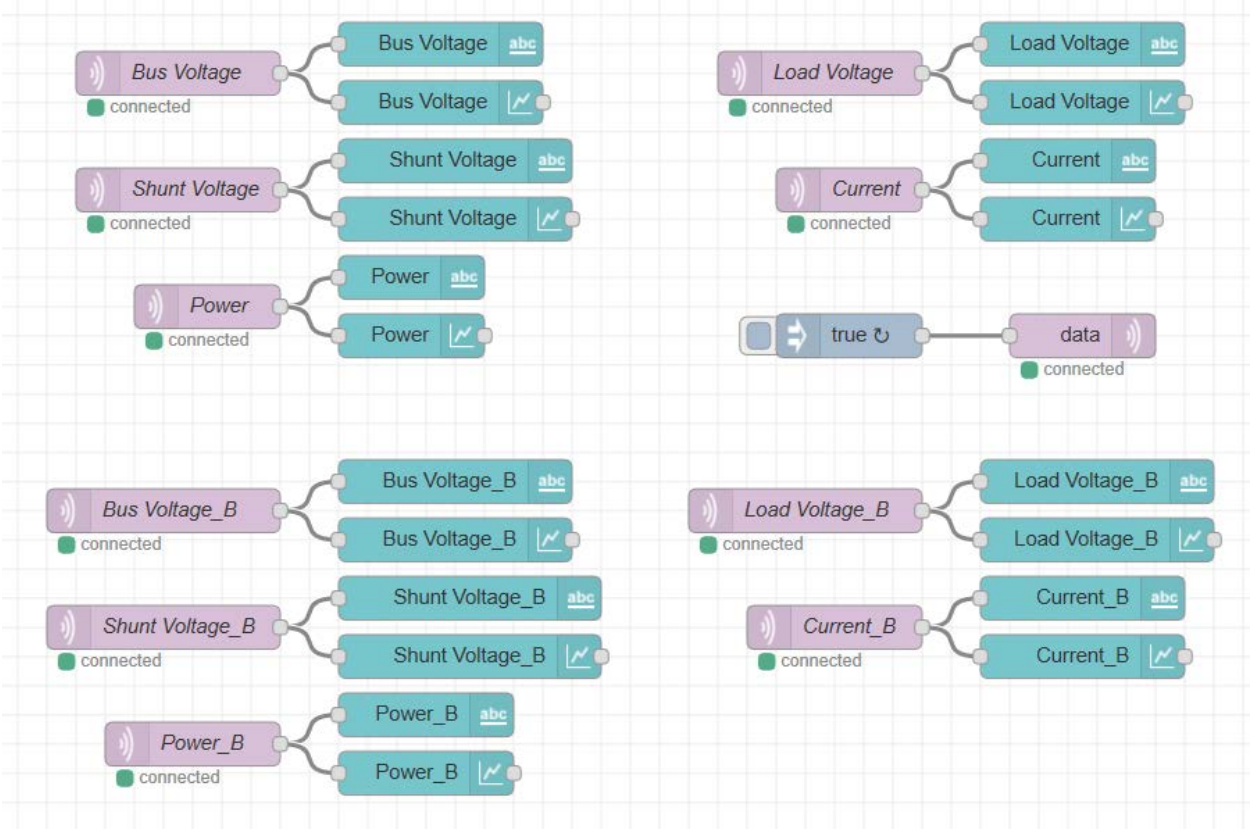

Figure 14: Node-Red flow for the Power tracker

To build the house model light wood and plastic was used in addition to nails, hot glue and wrap paper. The hardware includes several parts starting with the house model, SainSmart 8-Channel relay module (Amazon.com) and 4-channel relay module (Amazon.com) (figure 15,16) respectively, it allows the house to be automated without any serious modifications and when combined with ESP8266 Wi-Fi module (figure 8) it allows it to expand over large areas wirelessly.

Figure 15 SainSmart 8-Channel Relay Module

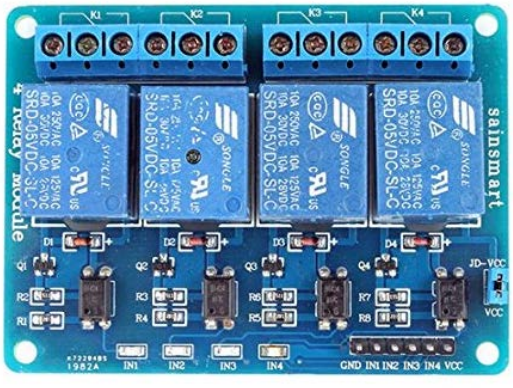

Figure 16 4-channel relay module

The channel relay and ESP8266 allows the user to control the house over the internet and allow for future expansion. To automate the house, sensors (figure 17) and feedback loop are used, but not all sensors report to the Raspberry Pi directly. The motion sensor (PIR HC) (Amazon.com), for instance, is connected directly to the channel relay to counter any communication delay between it and the Raspberry Pi and allow the lights to turn on and off even if the Raspberry Pi is offline. Also, the smoke detector (MQ-2 sensor)(Amazon.com) detects smoke and turn on the alarm, and Flame sensor (ARD0633) detects flame and turn on the sprinklers. Also, the temperature and humidity sensor (DHT11) (Amazon.com) controls the home climate system.

Figure 17: sensors (Amazon.com)

The Raspberry pi also receives information of the photovoltaic system operation from the INA219 power tracker via the ESP wi-fi module it allows the user to monitor power production and consumption. The dynamics of the automation and sustainability features are exhibit in (figure 18). The INA219 gathers the power consumption date and send it via ESP8266 to the Raspberry Pi to analyze it and store it. The sensors send feedback to the Raspberry Pi and control outlets at the same time. The raspberry pi also does control outlets like the Air Conditioner and webserver.

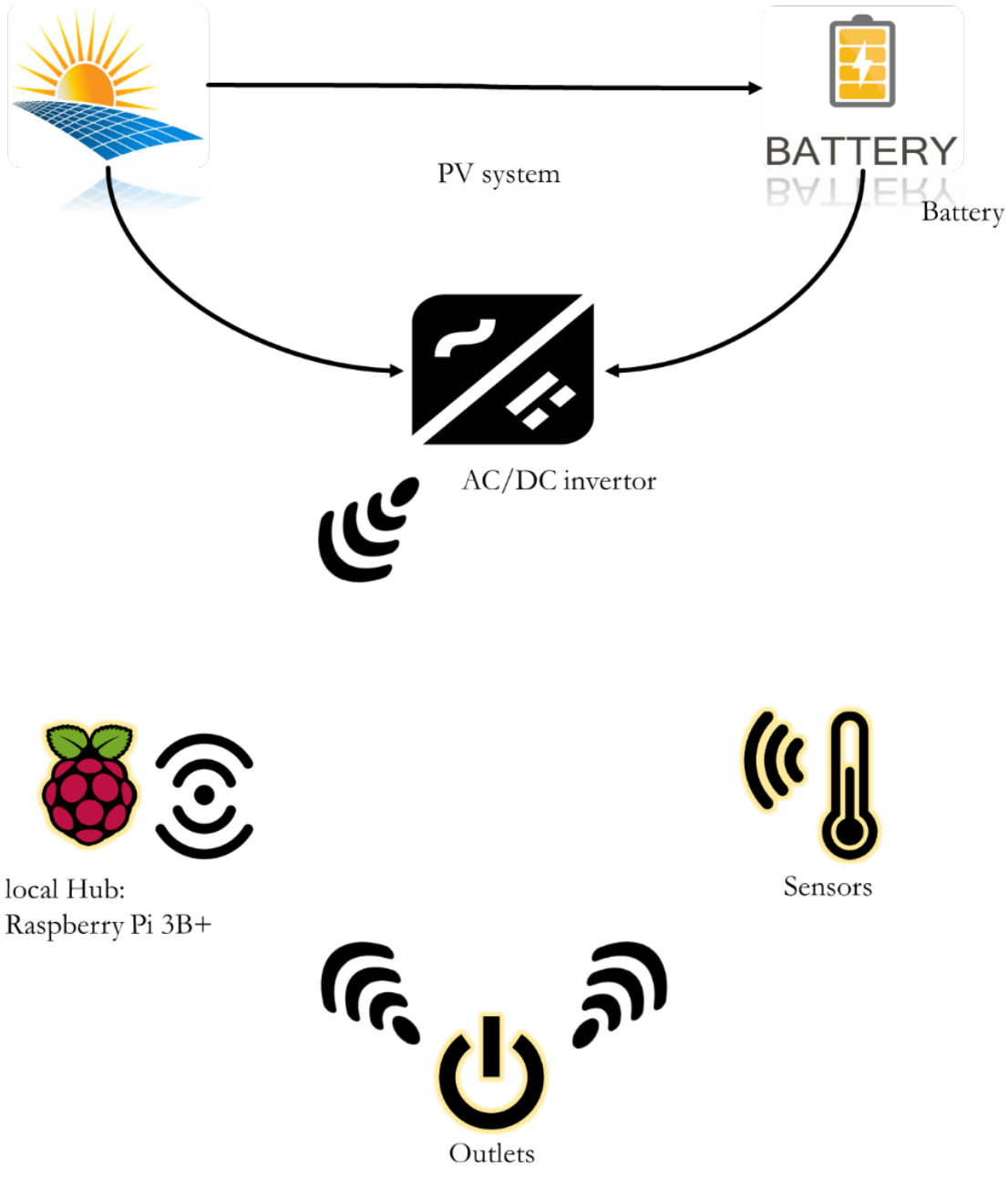

Figure 18: the project dynamic

V. constraints on the final design

Since the project is implemented on a house model, the feedback is limited to what to be expected from the house because for example motion sensors cannot provide real feedback because everything will trigger them. And the solar panels cannot track their full performance because they are inside most of the time. However, most of the applications in the project can be implemented as they are on a real house.

VI. Standards and safety:

The safety and standards set by the IEEE were considered when designing the project and using any outside source. Raspbian, Python and all software/libraries were obtained from an open source with the authorization from the manufacturer.

Appendix A

This appendix includes the code used for the fire alarm system, monitor power consumption, receive feedback form sensors and control the channel relay modules:

- The software for the fire alarm:

```
import RPi.GPIO as GPIO
from time import sleep
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
buzzer=12
SmkPin = 24
FirePin = 25
sprinkler = 17
GPIO.setup(buzzer,GPIO.OUT)
GPIO.setup(sprinkler,GPIO.OUT)
GPIO.setup(SmkPin,GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(FirePin,GPIO.IN, pull_up_down=GPIO.PUD_UP)
try:
    while True:
        if GPIO.input(FirePin) ==True :
            GPIO.output(buzzer,GPIO.HIGH)
            sleep(0.25)
            GPIO.output(buzzer,GPIO.LOW)
```

```
            sleep(0.15)
            GPIO.output(sprinkler,GPIO.HIGH)

        elif GPIO.input(SmkPin) == True:

            GPIO.output(buzzer,GPIO.HIGH)
            sleep(0.5)
            GPIO.output(buzzer,GPIO.LOW)
            sleep(0.5)
        else:
            GPIO.output(buzzer, GPIO.LOW)
            GPIO.output(sprinkler,GPIO.LOW)

except KeyboardInterrupt:
    GPIO.cleanup()
```

- The Software to monitor power consumption:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <Wire.h>
#include <Adafruit_INA219.h>
Adafruit_INA219 ina219_A;
Adafruit_INA219 ina219_B(0x41);
Adafruit_INA219 ina219_C(0x44);

void loop();

const char* ssid = "alban";
const char* password = "hkmksa101";
const char* mqtt_server = "192.168.137.102";

WiFiClient espPVsys;
PubSubClient client(espPVsys);

long now = millis();
long lastMeasure = 0;

void setup_wifi()
{
```

```
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("WiFi connected - ESP IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(String topic, byte* message, unsigned int length)
{
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
```

```
  }
  Serial.println();
  if(topic=="data"){
      Serial.print("received topic 'data' sending data");
      loop();
      }
  Serial.println();
}
void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("ESP8266Client")) {
      Serial.println("connected");
      client.subscribe("Data");
       } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 3 seconds");
      delay(3000);
    }
  }
}

void setup()
```

```
{
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);

  uint32_t currentFrequency;
  Serial.println("Hello!");
  ina219_A.begin();
  ina219_B.begin();
  ina219_C.begin();

  //use a lower 16V, 400mA range (higher precision on volts and amps):
  ina219_A.setCalibration_16V_400mA();
  ina219_B.setCalibration_16V_400mA();
  ina219_C.setCalibration_16V_400mA();

}

void loop()
{
   if (!client.connected()) {
    reconnect();
  }
```

```
if(!client.loop())
  client.connect("ESP8266Client");
 now = millis();
if (now - lastMeasure > 10000)
{
float shuntvoltage_A = 0;
float busvoltage_A = 0;
float current_mA_A = 0;
float loadvoltage_A = 0;
float power_mW_A = 0;

float shuntvoltage_B = 0;
float busvoltage_B = 0;
float current_mA_B = 0;
float loadvoltage_B = 0;
float power_mW_B = 0;

float shuntvoltage_C = 0;
float busvoltage_C = 0;
float current_mA_C = 0;
float loadvoltage_C = 0;
float power_mW_C = 0;

shuntvoltage_A = ina219_A.getShuntVoltage_mV();
```

```
busvoltage_A = ina219_A.getBusVoltage_V();
current_mA_A = ina219_A.getCurrent_mA();
power_mW_A = ina219_A.getPower_mW();
loadvoltage_A = busvoltage_A + (shuntvoltage_A / 1000);

float SV_A = shuntvoltage_A;
static char S_voltage_A[15];
dtostrf(SV_A, 6, 2, S_voltage_A);

float BV_A = busvoltage_A;
static char b_voltage_A[15];
dtostrf(BV_A, 6, 2, b_voltage_A);

float C_mA_A = current_mA_A;
static char c_mAmp_A[15];
dtostrf(C_mA_A, 6, 2, c_mAmp_A);

float P_mW_A = power_mW_A;
static char pwr_mW_A[15];
dtostrf(P_mW_A, 6, 2, pwr_mW_A);

float LV_A = loadvoltage_A;
static char l_voltage_A[15];
dtostrf(LV_A, 6, 2, l_voltage_A);
```

```
shuntvoltage_B = ina219_B.getShuntVoltage_mV();
busvoltage_B = ina219_B.getBusVoltage_V();
current_mA_B = ina219_B.getCurrent_mA();
power_mW_B = ina219_B.getPower_mW();
loadvoltage_B = busvoltage_B + (shuntvoltage_B / 1000);

float SV_B = shuntvoltage_B;
static char S_voltage_B[15];
dtostrf(SV_B, 6, 2, S_voltage_B);

float BV_B = busvoltage_B;
static char b_voltage_B[15];
dtostrf(BV_B, 6, 2, b_voltage_B);

float C_mA_B = current_mA_B;
static char c_mAmp_B[15];
dtostrf(C_mA_B, 6, 2, c_mAmp_B);

float P_mW_B = power_mW_B;
static char pwr_mW_B[15];
dtostrf(P_mW_B, 6, 2, pwr_mW_B);

float LV_B = loadvoltage_B;
```

```
static char l_voltage_B[15];
dtostrf(LV_B, 6, 2, l_voltage_B);



shuntvoltage_C = ina219_C.getShuntVoltage_mV();
busvoltage_C = ina219_C.getBusVoltage_V();
current_mA_C = ina219_C.getCurrent_mA();
power_mW_C = ina219_C.getPower_mW();
loadvoltage_C = busvoltage_C + (shuntvoltage_C / 1000);

float SV_C = shuntvoltage_C;
static char S_voltage_C[15];
dtostrf(SV_C, 6, 2, S_voltage_C);

float BV_C = busvoltage_C;
static char b_voltage_C[15];
dtostrf(BV_C, 6, 2, b_voltage_C);

float C_mA_C = current_mA_C;
static char c_mAmp_C[15];
dtostrf(C_mA_C, 6, 2, c_mAmp_C);

float P_mW_C = power_mW_C;
static char pwr_mW_C[15];
```

```
dtostrf(P_mW_C, 6, 2, pwr_mW_C);

float LV_C = loadvoltage_C;
static char l_voltage_C[15];
dtostrf(LV_C, 6, 2, l_voltage_C);

client.publish("shuntvoltage", S_voltage_A);
client.publish("busvoltage", b_voltage_A);
client.publish("current_mA", c_mAmp_A);
client.publish("power_mW", pwr_mW_A);
client.publish("loadvoltage", l_voltage_A);

client.publish("shuntvoltage_B", S_voltage_B);
client.publish("busvoltage_B", b_voltage_B);
client.publish("current_mA_B", c_mAmp_B);
client.publish("power_mW_B", pwr_mW_B);
client.publish("loadvoltage_B", l_voltage_B);

client.publish("shuntvoltage_C", S_voltage_C);
client.publish("busvoltage_C", b_voltage_C);
client.publish("current_mA_C", c_mAmp_C);
client.publish("power_mW_C", pwr_mW_C);
client.publish("loadvoltage_C", l_voltage_C);
```

```
  Serial.println("**********First
IAN219**********");
  Serial.println("Measuring voltage and current with ina219_A ...");
  Serial.print("Bus Voltage_A:   "); Serial.print(busvoltage_A); Serial.println(" V");
  Serial.print("Shunt Voltage_A: "); Serial.print(shuntvoltage_A); Serial.println(" mV");
  Serial.print("Load Voltage_A:  "); Serial.print(loadvoltage_A); Serial.println(" V");
  Serial.print("Current_A:       "); Serial.print(current_mA_A); Serial.println(" mA");
  Serial.print("Power_A:         "); Serial.print(power_mW_A); Serial.println(" mW");

  Serial.println("**********Second
IAN219**********");
  Serial.println("Measuring voltage and current with ina219_b ...");
  Serial.print("Bus Voltage_B:   "); Serial.print(busvoltage_B); Serial.println(" V");
  Serial.print("Shunt Voltage_B: "); Serial.print(shuntvoltage_B); Serial.println(" mV");
  Serial.print("Load Voltage_B:  "); Serial.print(loadvoltage_B); Serial.println(" V");
  Serial.print("Current_B:       "); Serial.print(current_mA_B); Serial.println(" mA");
  Serial.print("Power_B:         "); Serial.print(power_mW_B); Serial.println(" mW");

  Serial.println("**********Third
IAN219**********");
  Serial.println("Measuring voltage and current with ina219_c ...");
  Serial.print("Bus Voltage_C:   "); Serial.print(busvoltage_C); Serial.println(" V");
  Serial.print("Shunt Voltage_C: "); Serial.print(shuntvoltage_C); Serial.println(" mV");
```

```
  Serial.print("Load Voltage_C:  "); Serial.print(loadvoltage_C); Serial.println(" V");
  Serial.print("Current_C:       "); Serial.print(current_mA_C); Serial.println(" mA");
  Serial.print("Power_C:         "); Serial.print(power_mW_C); Serial.println(" mW");

Serial.println("**********************************************************************
********");
  Serial.println("");
  Serial.println("");
  Serial.println("");
  delay(500);
  }
}
```

- The Software to receive feedback form sensors:

```
#include <DHTesp.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <Wire.h>
#include <Adafruit_ADS1015.h>
#include "DHT.h"

Adafruit_ADS1115 ads;

const char* ssid = "alban";
const char* password = "hkmksa101";
const char* mqtt_server = "192.168.137.102";

WiFiClient espSensors;
PubSubClient client(espSensors);

const int DHTPin = D3;

DHT dht(DHTPin, DHT11);

long now = millis();
long lastMeasure = 0;

void setup_wifi() {
  delay(10);
  //We start by connecting to a WiFi network
  Serial.println();
```

```
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("WiFi connected - ESP IP address: ");
  Serial.println(WiFi.localIP());
}
void callback(String topic, byte* message, unsigned int length)
{
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;
  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();
}
void reconnect()
{
  while (!client.connected())
  {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("ESPSensors")) {
```

```
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 3 seconds");
      delay(3000);
    }
  }
}
void setup()
{
  ads.setGain(GAIN_TWO);                // 2x gain   +/- 2.048V  1 bit = 1mV      0.0625mV
  ads.begin();
  dht.begin();
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
void loop()
{
  if (!client.connected()) {
    reconnect();
  }
  if(!client.loop())
    client.connect("espSensors");
  now = millis();
  if (now - lastMeasure > 10000)
  {
```

```
double adc0, adc1, adc2, adc3;
lastMeasure = now;
float h = dht.readHumidity();
float t = dht.readTemperature();
float f = dht.readTemperature(true);
adc0 = 100 - (ads.readADC_SingleEnded(0)/327.67);
adc1 = (ads.readADC_SingleEnded(1)/327.67);
adc2 = 100 - (ads.readADC_SingleEnded(2)/327.67);
adc3 = 100 - (ads.readADC_SingleEnded(3)/327.67);

float hic = dht.computeHeatIndex(t, h, false);
static char temperatureTemp[7];
dtostrf(hic, 6, 2, temperatureTemp);

static char humidityTemp[7];
dtostrf(h, 6, 2, humidityTemp);

 float SM = adc0;
 static char soil[15];
 dtostrf(SM, 6, 2, soil);

 float wtrlvl = adc1;
 static char waterlvl[15];
 dtostrf(wtrlvl, 6, 2, waterlvl);

 float RI = adc2;
 static char rain[15];
 dtostrf(RI, 6, 2, rain);
```

```
    float UVi = adc3;
    static char UV[15];
    dtostrf(UVi, 6, 2, UV);

   client.publish("room/temperature", temperatureTemp);
   client.publish("room/humidity", humidityTemp);

   client.publish("garden/soil", soil);
   client.publish("garden/waterlvl", waterlvl);
   client.publish("garden/rain", rain);
   client.publish("UVindex", UV);

   Serial.println("");
   Serial.print("   soil water:"); Serial.print(soil);
   Serial.print("   water level:"); Serial.println(waterlvl);Serial.print("\n");

   Serial.print("   rain index:"); Serial.print(rain);
   Serial.print("   UV index:"); Serial.println(UV);Serial.print("\n");

   Serial.print("   Humidity: ");Serial.print(h);Serial.println(" %");Serial.print("\n");

   Serial.print("   Temperature: ");Serial.print(t);Serial.print(" C    ");
   Serial.print(f);Serial.print("F");Serial.println("\n");

   Serial.print("   Heat index: "); Serial.print(hic);Serial.print(" C    ");

Serial.println("##################################################################
##########");
   delay(500);  }
}
```

- The Software to control the channel relay modules:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <Wire.h>

const char* ssid = "alban";                 //WIFI name
const char* password = "hkmksa101";         //WiFi Password
const char* mqtt_server = "192.168.137.102"; //Raspberry Pi IP address
WiFiClient espRelay;
PubSubClient client(espRelay);

const int outlet1 = D5;
const int outlet2 = D6;
const int outlet3 = D7;
const int outlet4 = D8;

void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("WiFi connected - ESP IP address: ");
```

```
  Serial.println(WiFi.localIP());
}
void callback(String topic, byte* message, unsigned int length)
{
  Serial.print("Message arrived on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;

  for (int i = 0; i < length; i++) {
    Serial.print((char)message[i]);
    messageTemp += (char)message[i];
  }
  Serial.println();
  if(topic=="room/outlet1"){
      Serial.print("Changing Room outlet1 to ");

      if(messageTemp == "on"){
        digitalWrite(outlet1, HIGH);
        Serial.print("On");
      }
      else if(messageTemp == "off"){
        digitalWrite(outlet1, LOW);
        Serial.print("Off");
      }
    }
  else if(topic=="room/outlet2") {
    Serial.print("Changing Room outlet2 to ");
```

```
  if(messageTemp == "on"){
    digitalWrite(outlet2, HIGH);
    Serial.print("On");
  }
  else if(messageTemp == "off"){
    digitalWrite(outlet2, LOW);
    Serial.print("Off");
  }
}
  else if(topic=="room/outlet3"){
    Serial.print("Changing Room outlet3 to ");

  if(messageTemp == "on"){
    digitalWrite(outlet3, HIGH);
    Serial.print("On");
  }
  else if(messageTemp == "off"){
    digitalWrite(outlet3, LOW);
    Serial.print("Off");
  }
}



else if(topic=="room/outlet4"){
    Serial.print("Changing Room outlet4 to ");

  if(messageTemp == "on"){
    digitalWrite(outlet4, HIGH);
    Serial.print("On");
```

```
    }
    else if(messageTemp == "off"){
      digitalWrite(outlet4, LOW);
      Serial.print("Off");
    }
  }
  Serial.println();
}
void reconnect()
{
  // Loop until we're reconnected
  while (!client.connected())
  {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("ESPRelay"))
    {
      Serial.println("connected");
      client.subscribe("room/outlet1");
      client.subscribe("room/outlet2");
      client.subscribe("room/outlet3");
      client.subscribe("room/outlet4");
    }
    else
    {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 3 seconds");
      // Wait 5 seconds before retrying
      delay(3000);
```

```
    }
  }
}
void setup()
{
  pinMode(outlet1, OUTPUT);
  pinMode(outlet2, OUTPUT);
  pinMode(outlet3, OUTPUT);
  pinMode(outlet4, OUTPUT);
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

//////////////////////////////////////////////////////////////
void loop()
{
  if (!client.connected()) {
    reconnect();
  }
  if (!client.loop())
      client.connect("ESPRelay");
}
```

References

"10A Solar Charge Controller, Solar Panel Controller 10amp 12V/24V Auto Paremeter Adjustable LCD Display Solar Panel Regulator with Dual USB Load Timer Setting ON/Off Hours." *Amazon*, Amazon, www.amazon.com/gp/product/B07H8DS5VT/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

Adafruit Industries. "INA219 High Side DC Current Sensor Breakout - 26V ±3.2A Max." *Adafruit Industries Blog RSS*, www.adafruit.com/product/904.

"BESTEK 150W Power Inverter DC 12V to 110V AC Converter 4.2A Dual USB Car Adapter, Thinner Design with ETL Listed." *Amazon*, Amazon, www.amazon.com/gp/product/B07DVXNSDH/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"ELEGOO Upgraded 37 in 1 Sensor Modules Kit with Tutorial for Arduino UNO R3 MEGA 2560 Nano." *Amazon*, Amazon, www.amazon.com/gp/product/B01MG49ZQ5/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"Element14 Raspberry Pi 3 B Motherboard." *Amazon*, Amazon, www.amazon.com/gp/product/B07BDR5PDW/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"HiLetgo 2pcs ESP8266 NodeMCU LUA CP2102 ESP-12E Internet WiFi Development Board Open Source Serial Wireless Module Works Great with Arduino IDE/Micropython (Pack of 2PCS)." *Amazon*, Amazon, www.amazon.com/gp/product/B010N1SPRK/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"HSU Development Kit for Raspberry Pi 3 and Arduino with 16 Different Sensor Modules,Hundreds Electronic Components,Other Necessary Accessories and Big Carrying Case (Advanced)." *Amazon*, Amazon, www.amazon.com/gp/product/B073WVFGMK/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"JBtek 8 Channel DC 5V Relay Module for Arduino Raspberry Pi DSP AVR PIC ARM." *Amazon*, Amazon, www.amazon.com/gp/product/B00KTELP3I/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"MakerFocus 2pcs ESP8266 NodeMCU LUA CH340 ESP-12E Internet WiFi Development Board 4M Flash Serial Wireless Module Internet for Arduino." *Amazon*, Amazon, www.amazon.com/gp/product/B01N8UUE3L/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1.

"Running on Raspberry Pi." *Node-Red*, nodered.org/docs/hardware/raspberrypi.

"Uxcell 5Pcs 6V 180mA Poly Mini Solar Cell Panel Module DIY for Light Toys Charger 133mm x 73mm." *Amazon*, Amazon, www.amazon.com/uxcell-180mA-Solar-Module-Charger/dp/B073Y6CXJZ/ref=pd_ybh_a_2?_encoding=UTF8&psc=1&refRID=R4HYEASHP4ZV643D3NHX.

"Wildgame Innovations 12 Volt EDRENALINE Rechargeable Tab Style Battery." *Amazon*, Amazon, www.amazon.com/Wildgame-Innovations-eDRENALINE-Rechargeable-Battery/dp/B0031AUPOC/ref=pd_ybh_a_2?_encoding=UTF8&psc=1&refRID=BDZ6SJK5CH4XYQ473394.