LEBANESE UNIVERSITY
Faculty of Sciences II

SCIENTIFIC REPORT

Electronics Laboratory E2206

2nd year Project: Automated Greenhouse

By

Daniella Chebly
Elia Bou Karam
Elissa Bou Karam

Presented to Dr. MSAAED

May 6, 2019
4th Semester

I-Abstract

The aim of this project is to monitor and control the environmental conditions in a *greenhouse* using an *Arduino* that is programmed using *language C*. This will be achieved using multiple *sensors*, *power sources*, and *electronic devices* all connected to the *Arduino.*

Keywords: *greenhouse, Arduino, language C, sensors, power sources, electronic devices.*

II-Introduction

An automated greenhouse is a structure enclosed by glass or plastic and used for the cultivation or protection of tender plants under automatically controlled conditions. This provides the opportunity to grow crops out of their season and natural locations. The goal is to use technology to help in plant nurturing, minimize human resources, save water and minimize the use of excess resources. This requires the usage of an operating system able to receive, analyze and send data and thus communicate with the devices used ranging from sensors to electronic appliances.

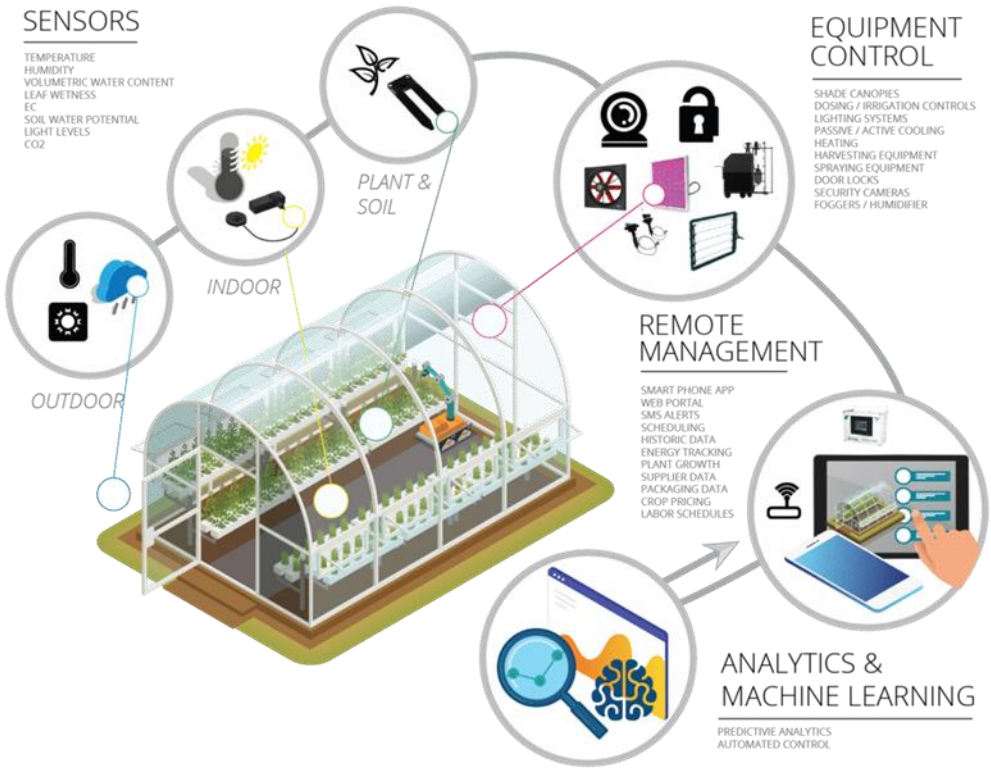

Figure 1: an example of an automated greenhouse [1]

In this project, the usage of Arduino is required. The Arduino board is a microcontroller used to connect to various devices; it is controlled using the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. [2] An Arduino Mega board will be used due to the need of numerous pins as outputs and input. The Arduino Mega 2560 is based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM (Pulse width modulation) outputs), 16 analog inputs, 4 UARTs (Universal Asynchronous Receiver/Transmitter) (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. [3]

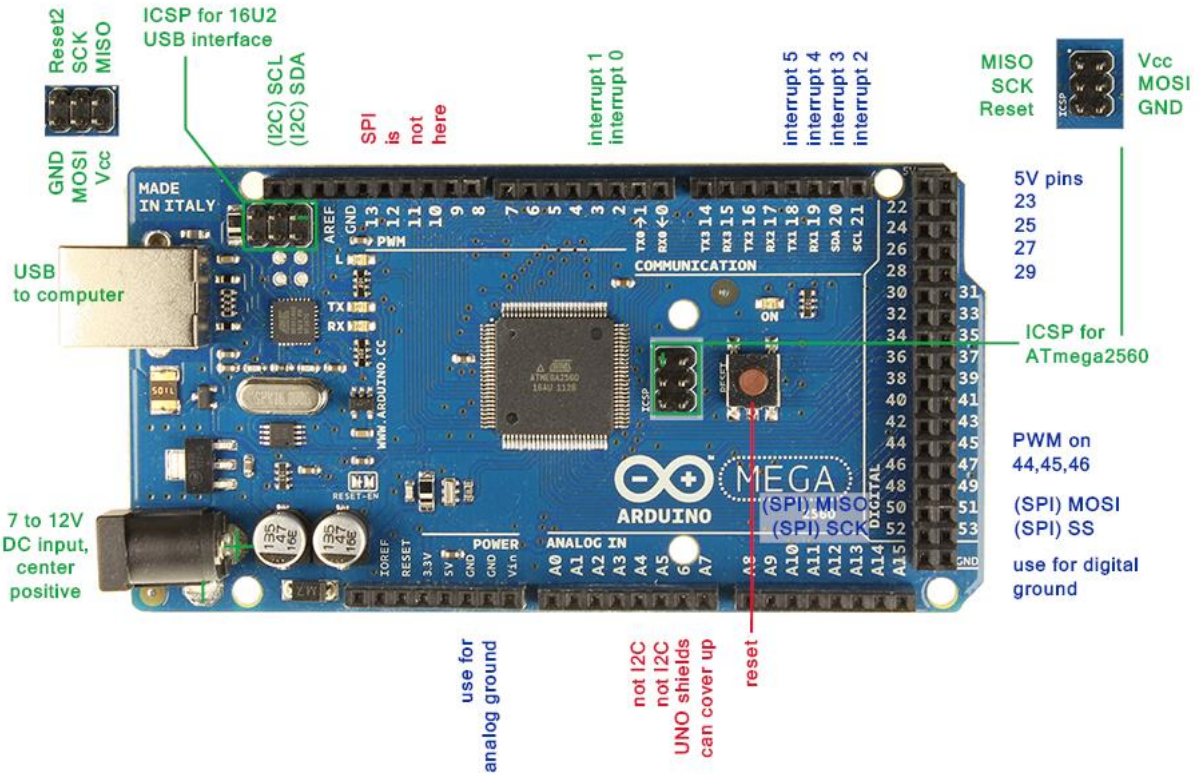

Figure 2: labeled Arduino Mega board [4]

The use of 15 digital pins and 8 analog pins is needed to monitor and control this automated greenhouse containing 6 plants. To each plant its own soil moisture sensor and water solenoid valve. All solenoid valves are connected to a water pump taking water from a 3L water tank. A solar panel is used to charge a 12V battery using a solar panel controller. This battery is connected to a DC-DC converter to lower the voltage to 5.1V in order to power the Arduino board and the sensors. Whereas 12V are used to power the appliances including the water pump, the water solenoid valves, a fan and a programmable RGB LED strip.

The goal is to automatically monitor and control the plants' environment using the devices mentioned above and a few more. The detection of dry soil by a soil moisture sensor of a specific plant should trigger watering this plant by turning on the pump and the corresponding solenoid valve. If the temperature of the greenhouse is too high for the plants as detected by the temperature sensor, a fan is turned on to cool the environment. An LDR sensor used to detect light allows the RGB LED strip to be turned on to provide light to the plants thus maximizing growth in the absence of natural sunlight light. A flame sensor detects fire and sets off an alarm consisting of a successive on/off buzzer along with a red light provided by the RGB LED strip. Finally, the entering of an intruder is identified by the ultrasonic sensor consequently constantly turning on the buzzer and red light thus scaring away any animal that might harm the plants.

In the following sections, the functioning and connection of the devices used will be further discussed.

III-Devices

The following is a list of all the devices placed in the project model:

- Sensors:
  - Soil moisture sensors
  - Temperature and humidity sensor
  - LDR sensor
  - Flame sensor
  - Ultrasonic sensor
- Appliances:
  - Solar panel
  - Relays
  - DC-DC down converter
  - Batteries
  - RGB LED strips: a programmable one and a non-programmable one
  - Water pump
  - Water solenoid valves
  - Fan
- Arduino:
  - Arduino Mega board
- Cables:
  - Arduino to USB connection cable
  - Jumper wires
  - Copper cables

A detailed explanation of the devices will be discussed in the following subsection.

- Sensors:
  - Soil moisture sensor:

    This sensor uses capacitance to measure dielectric permittivity of the surrounding medium. In soil, dielectric permittivity is a function of the water content. The sensor creates a voltage proportional to the dielectric permittivity, and therefore the water content of the soil. [5] This sensor has 3 pins: a 5V pin (V_{cc}), a ground pin (GND), and an analog pin (A0). 6 soil moisture sensors were used (one for each plant).

    Returned Values: from 0 (completely dry) to 1023 (completely moist). (air/soil humidity - ambient conditions). [6]

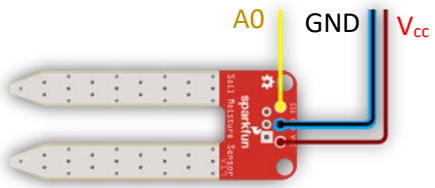

Figure 3: soil moisture sensor [6]

- Temperature and humidity sensor DHT11:
  The ratio of moisture in the air to the highest amount of moisture at a particular air temperature is called relative humidity. Humidity sensors work by detecting changes that alter electrical currents or temperature in the air. [7] The DHT11 uses a capacitive humidity sensor and a thermistor (a temperature dependent resistor) to measure the surrounding air and sends a digital signal through its digital output pin (DOUT). It has two other input pins: a 5V pin (VCC) and a ground pin (GND).

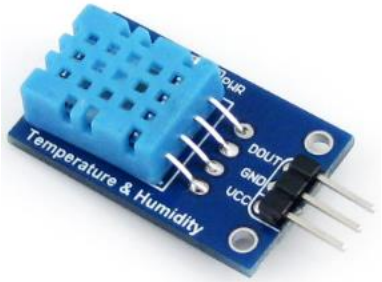

Figure 4: DHT11 temperature and humidity sensor

- Light dependent resistor (LDR) sensor:
  It is a photodiode which is a type of semiconductor that is sensitive to light. In the presence of light, this semiconductor absorbs the energy of the photons thus electrons are freed from the valence band to be at the conduction band and free to move therefore making the semiconductor more conductive and less resistive. This opposite occurs in the absence of light where the semiconductor's conductivity decreases, and its resistivity increases due to the drastic decrease in the number of free electrons.
  The LDR module used send the level of brightness through a digital pin (D0). It also has 2 other pins: 5V pin (VCC), ground pin (GND).

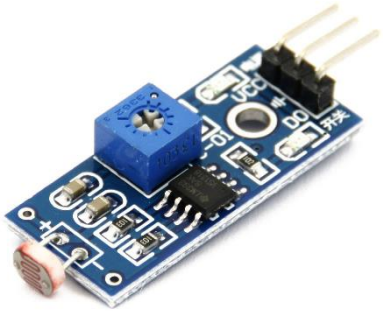

Figure 5: LDR sensor module

- Flame sensor:
  Flame detectors include an electronic circuit with an electromagnetic radiation receiver. A flame sensor uses the IR (infrared) flame flicker techniques, which enables the sensor to operate through a layer of oil, water vapor, dust, or ice. Most IR flame sensors are designed to respond to 4.3μm light emitted by hydrocarbon flames. [8]
  The used flame sensor has 3 pins: digital output pin (D0), 5V pin (VCC) and a ground pin (⏚)

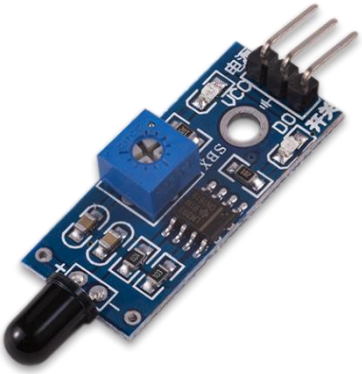

Figure 6: flame sensor

- Ultrasonic:
  These sensors send an ultrasonic wave when commanded by the "trig" pin (Trig) and receive the wave after it reflect on an object and send the received data through the "echo" pin (Echo). A measurement of the distance between the sensor and the object can be made by measuring the time difference between sending and receiving the ultrasonic wave. This detected time difference is in milliseconds, converted to seconds and multiplied by the speed of sound in air, the distance is obtained. This sensor has 2 more pins: a 5V pin (VCC) and a ground pin (GND).

Figure 7: ultrasonic sensor

- Appliances:
  - Solar panel:

    It is a photovoltaic cell used to capture sunlight and transform it into electricity. Through the photoelectric effect, electrons that absorb photons will jump across the energy gap barrier, from the valence to the conduction band, and flow out around the circuit, generating electricity. [9]

    The solar panel used can supply a maximum output of 18V. A controller is used to stabilize the current flow from the solar to the battery. This controller detects the battery's voltage when first connected and sets the voltage to be supplied either to 12V or to 6V. The battery used here supplies 12V/3.2A.

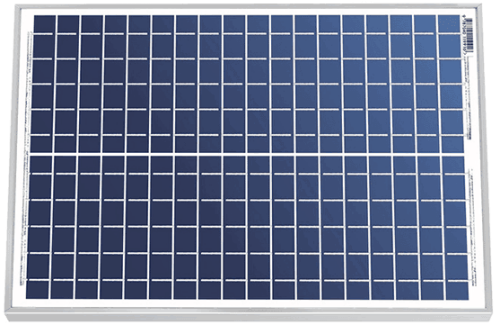

Figure 8: an example of a solar panel

  - Relays:

    Two LOW level trigger relay modules were used: an 8-channel relay module and a 4-channel relay module. The modules used are wired to the normally open contact. When a LOW pulse is delivered, the armature in the relay switches from the normally closed contact to the normally open contact thus a closed circuit is present between the voltage source and the device. The relay modules need an external power supply of 5V.

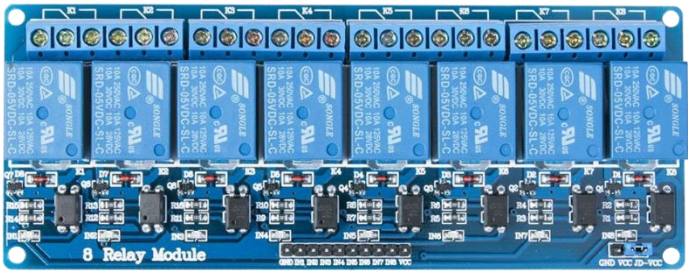

Figure 9: 8-channel LOW trigger relay module

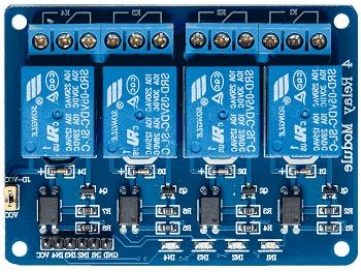

Figure 10: 4-channel LOW triggered relay module

- DC-DC down converter:
  A DC-DC down converter steps its input DC voltage to a lower DC voltage at its output. In this project, it is used to step down the 12V from the battery connected to the solar panel to 5.01V to power the Arduino board and the sensors. This connection has been done using a PCB board.

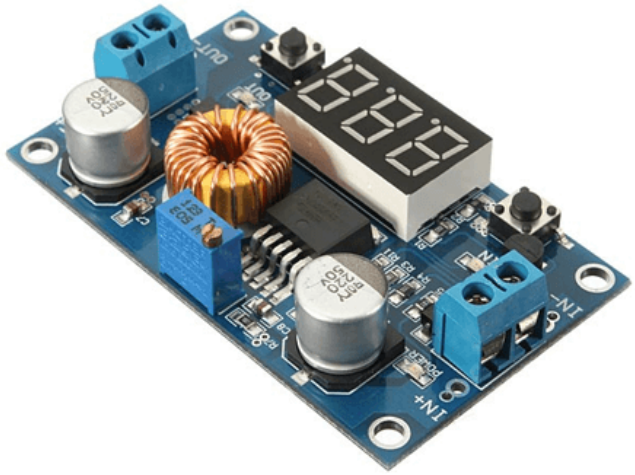

Figure 11: DC-DC down converter

- Batteries:
  Two batteries were used with the following supply indications:
  - $12V/3.2A$ connected to the solar system
  - $12V/7A$ used to power the non-programmable RGB LED strip

- RGB LED strips:
  Two types of strips were used:
  - Programmable RGB LED strip: each color is connected to a relay switch thus allowing the program to control which colors are on or off.
  - Non-programmable RGB LED strip: need a supply of 12V/6A

- Water pump:
  It is used to pump water through the 7mm hoses when needed. It is connected to a relay switch thus allowing the program to control when it is turned on/off. It needs a supply of 12V to function.

- Water solenoid valves:
  They are water switches that allow water to flow to specific 5mm hoses when commanded to do so since they are connected to the relay switches. They require a supply of 12V to function.

- Fan:
  It is used for cooling, requires an input of 12V, is connected to a relay switch and has the dimensions of $7cm \times 7cm$.

IV-Code

```
#include <SimpleDHT.h> //temperature sensor library

//constant values to be used later in the program
#define DRY 0
#define MOIST 400
#define WET 650
#define DARK 200
#define HOT 24
#define ANGLE 80
#define ALERT 60

//constant values used to assign number of digital pins
#define valve2 26
#define valve3 29
#define valve4 30
#define valve5 31
#define valve6 27
#define valve7 28
#define RED 9
#define GREEN 10
#define BLUE 11
#define buzzer 12
#define flame_sensor 13
#define trigPin 22
#define echoPin 23
#define pinDHT11 24
#define fan 8

//variables used to assign number analog pins
int pumpt = A7;
const int ldrPin = A0;

//variables to be used to store and calculate values
int flame_detected;
long duration;
int distance;

//defining the temperature sensor pin
SimpleDHT11 dht11(pinDHT11);

//defining void functions to be used
void water();
void sunlight();
void flame();
```

```
void intruder();
void heat();

//setup code to initialize the system
void setup() {
  Serial.begin(9600); //must be the same for all sensors

  //setting the mode of the digital pins (input or output)
  pinMode(ldrPin, INPUT);
  pinMode(RED, OUTPUT);
  pinMode(GREEN, OUTPUT);
  pinMode(BLUE, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(flame_sensor, INPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(pinDHT11, INPUT);
  pinMode(fan, OUTPUT);
  pinMode(valve2,OUTPUT);
  pinMode(valve3,OUTPUT);
  pinMode(valve4,OUTPUT);
  pinMode(valve5,OUTPUT);
  pinMode(valve6,OUTPUT);
  pinMode(valve7,OUTPUT);
  pinMode(pumpt,OUTPUT);

  //initializing the signal of all devices connected to relays (HIGH is off, LOW is on)
  digitalWrite(buzzer,LOW);
  digitalWrite(valve2,HIGH);
  digitalWrite(valve3,HIGH);
  digitalWrite(valve4,HIGH);
  digitalWrite(valve5,HIGH);
  digitalWrite(valve6,HIGH);
  digitalWrite(valve7,HIGH);
  digitalWrite(fan,HIGH);
  digitalWrite(RED,HIGH);
  digitalWrite(GREEN,HIGH);
  digitalWrite(BLUE,HIGH);
}

//main code to be run repeatedly
void loop() {
  //calling all functions
  water();
  sunlight();
  flame();
```

```
  intruder();
  heat();

  //1.5s of delay between each loop
  delay(1500);
}

//function used to check soil moisture level and control the watering mechanism
void water(){
  //defining the soil moisture sensor analog pin numbers and mode of operation (Read, Write...)
  int sensorValue2 = analogRead(A8);
  int sensorValue3 = analogRead(A9);
  int sensorValue4 = analogRead(A10);
  int sensorValue5 = analogRead(A11);
  int sensorValue6 = analogRead(A12);
  int sensorValue7 = analogRead(A13);

  //setting the signal of the pump for it to be turned off (255 in analog is similar to HIGH in digital)
  analogWrite(pumpt, 255);

  /*Serial.print("plant 1: ");
  Serial.println(sensorValue2);
  Serial.print("plant 2: ");
  Serial.println(sensorValue3);
  Serial.print("plant 3: ");
  Serial.println(sensorValue4);
  Serial.print("plant 4: ");
  Serial.println(sensorValue5);
  Serial.print("plant 5: ");
  Serial.println(sensorValue6);
  Serial.print("plant 6: ");
  Serial.println(sensorValue7);*/

  //testing the soil moisture level and acting upon that reading
  if (sensorValue2>=DRY && sensorValue2<MOIST){ //waters the plants
    //Serial.println(sensorValue2);
    //Serial.println("Soil for plant 1 is dry.");
    analogWrite(pumpt,0); //sends a 0 signal to the relays to turn on the pump, similar to LOW in
digital
    digitalWrite(valve2,LOW);
    Serial.print("Plant 1 is being watered.\n");
    delay(1000);
    analogWrite(pumpt,255); //pump should be turned off before valve to prevent errors in the pipes
    delay(100);
    digitalWrite(valve2,HIGH);
    Serial.println("Water for plant 1 is turned off.");
```

```
}
else if(sensorValue2>=MOIST && sensorValue2<WET){ //no water is supplied to the plants
  //Serial.println(sensorValue2);
  //Serial.println("Soil for plant 1 moist.");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve2,HIGH);
  //Serial.println("Water for plant 1 is off");
}
else if(sensorValue2>=WET){ //no water is supplied to the plants
  //Serial.println(sensorValue2);
  //Serial.println("Soil for plant 1 wet");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve2,HIGH);
} //these conditions are applied to all 6 soil moisture sensors and valves

if (sensorValue3>=DRY && sensorValue3<MOIST){
  //Serial.println(sensorValue3);
  Serial.println("Soil for plant 2 dry");
  analogWrite(pumpt,0);
  digitalWrite(valve3,LOW);
  Serial.println("Plant 2 is being watered");
  delay(1000);
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve3,HIGH);
  Serial.println("Water for plant 2 is turned off");
}
else if(sensorValue3>=MOIST && sensorValue3<WET){
  //Serial.println(sensorValue3);
  //Serial.println("Soil for plant 2 moist");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve3,HIGH);
}
else if(sensorValue3>=WET){
  //Serial.println(sensorValue3);
  //Serial.println("Soil for plant 2 wet");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve3,HIGH);
}

if (sensorValue4>=DRY && sensorValue4<MOIST){
  //Serial.println(sensorValue4);
```

```
  Serial.println("Soil for plant 3 dry");
  analogWrite(pumpt,0);
  digitalWrite(valve4,LOW);
  Serial.println("Plant 3 is being watered");
  delay(1000);
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve4,HIGH);
  Serial.println("Water for plant 3 is turned off");
 }
 else if(sensorValue4>=MOIST && sensorValue4<WET){
  //Serial.println(sensorValue4);
  //Serial.println("Soil for plant 3 moist");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve4,HIGH);
 }
 else if(sensorValue4>=WET){
  //Serial.println(sensorValue4);
  //Serial.println("Soil for plant 3 wet");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve4,HIGH);
 }

 if (sensorValue5>=DRY && sensorValue5<MOIST){
  //Serial.println(sensorValue5);
  Serial.println("Soil for plant 4 dry");
  analogWrite(pumpt,0);
  digitalWrite(valve5,LOW);
  Serial.println("Plant 4 is being watered");
  delay(1000);
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve5,HIGH);
  Serial.println("Water for plant 4 is turned off");
 }
 else if(sensorValue5>=MOIST && sensorValue5<WET){
  //Serial.println(sensorValue5);
  //Serial.println("Soil for plant 4 moist");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve5,HIGH);
 }
 else if(sensorValue5>=WET){
  //Serial.println(sensorValue5);
```

```
  //Serial.println("Soil for plant 4 wet");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve5,HIGH);
 }

 if (sensorValue6>=DRY && sensorValue6<MOIST){
  //Serial.print("soil 5:");
  //Serial.println(sensorValue6);
  Serial.println("Soil for plant 5 dry");
  analogWrite(pumpt,0);
  digitalWrite(valve6,LOW);
  Serial.println("Plant 5 is being watered");
  delay(1000);
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve6,HIGH);
  Serial.println("Water for plant 5 is turned off");
 }
 else if(sensorValue6>=MOIST && sensorValue6<WET){
  //Serial.print("soil 5:");
  //Serial.println(sensorValue6);
  //Serial.println("Soil for plant 5 moist");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve6,HIGH);
 }
 else if(sensorValue6>=WET){
  //Serial.print("soil 5:");
  //Serial.println(sensorValue6);
  //Serial.println("Soil for plant 5 wet");
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve6,HIGH);
 }

 if (sensorValue7>=DRY && sensorValue7<MOIST){
  //Serial.println(sensorValue7);
  Serial.println("Soil for plant 6 dry");
  analogWrite(pumpt,0);
  digitalWrite(valve7,LOW);
  Serial.println("Plant 6 is being watered");
  delay(1000);
  analogWrite(pumpt,255);
  delay(100);
  digitalWrite(valve7,HIGH);
```

```
    Serial.println("Water for plant 6 is turned off");
  }
  else if(sensorValue7>=MOIST && sensorValue7<WET){
    //Serial.println(sensorValue7);
    //Serial.println("Soil for plant 6 moist");
    analogWrite(pumpt,255);
    delay(100);
    digitalWrite(valve7,HIGH);
  }
  else if(sensorValue7>=WET){
    //Serial.println(sensorValue7);
    //Serial.println("Soil for plant 6 wet");
    analogWrite(pumpt,255);
    delay(100);
    digitalWrite(valve7,HIGH);
  }
}

//function to detect sunlight and control lighting conditions
void sunlight(){
  int ldrStatus = analogRead(ldrPin);
  //Serial.println(ldrStatus);
  if (ldrStatus >= DARK) {
    //Serial.println(ldrStatus);
    Serial.print("It's DARK, LED turned on. \n");
    digitalWrite(RED,LOW);
    digitalWrite(GREEN,LOW);
    digitalWrite(BLUE,LOW);
  } else {
    //Serial.println(ldrStatus);
    Serial.print("Its BRIGHT, LED turned off. \n");
    digitalWrite(RED,HIGH);
    digitalWrite(GREEN,HIGH);
    digitalWrite(BLUE,HIGH);
  }
}

//function to detect fire and turn on alarm system
void flame(){
  flame_detected = digitalRead(flame_sensor);
  if (flame_detected == 0)
  {
    Serial.println("Flame detected...! take action immediately.");
    digitalWrite(buzzer, HIGH);
    digitalWrite(RED, LOW);
    digitalWrite(GREEN,HIGH);
```

```
    digitalWrite(BLUE,HIGH);
    delay(200);
    digitalWrite(buzzer, LOW);
    digitalWrite(RED, LOW);
    delay(200);
    digitalWrite(buzzer, HIGH);
    digitalWrite(RED, HIGH);
    delay(200);
    digitalWrite(buzzer, LOW);
    digitalWrite(RED, LOW);
    delay(200);
    digitalWrite(buzzer, HIGH);
    digitalWrite(RED, HIGH);
    delay(200);
    digitalWrite(buzzer, LOW);
    digitalWrite(RED, LOW);
    delay(200);
  }
  else
  {
    //Serial.println("No flame detected. stay cool");
    digitalWrite(buzzer, LOW);
    //digitalWrite(LED, LOW);
  }
}

//function to detect the presence of any intruder and turn on alarm
void intruder(){
  //clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  //sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  //reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);

  //calculating the distance
  distance = duration*0.034/2;

  /*// Prints the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.println(distance);*/
```

```
  if(distance<ALERT){
    Serial.print("Warning! Intruder!");
    digitalWrite(buzzer,HIGH);
    digitalWrite(GREEN,HIGH);
    digitalWrite(BLUE,HIGH);
    digitalWrite(RED,LOW);
    delay(200);
  }
  else{
    digitalWrite(buzzer,LOW);
  }
}

//function to measure temperature and control fan
void heat(){
  byte temperature = 0;
  byte humidity = 0;
  int err = SimpleDHTErrSuccess;
  if ((err = dht11.read(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess) { //detects
reading error
    //Serial.print("Read DHT11 failed, err=");
    //Serial.println(err);
    delay(1000);
    return;
  }

  //Serial.print("Temperature: ");
  //Serial.print((int)temperature);
  //Serial.print(" *C \n");
  //Serial.print((int)humidity); Serial.println(" H");

  //tests temperature value and turns on/off fan
  if((int)temperature>=HOT){
    digitalWrite(fan,LOW);
    /*Serial.print("Temperature: ");
    Serial.print((int)temperature);*/
    Serial.print(" *C \n");
    Serial.print("Fan is on.\n");
  }
  else{
    digitalWrite(fan,HIGH);
    //Serial.print("Fan is off.\n");
  }
}
```

V-Result

Figure 12: the project model

The model was run for a period of time where the flowers were automatically watered and taken care of. The temperature was maintained at 24℃. During night time, artificial light was provided to the flowers. Throughout one and a half weeks, the seasonal flowers renewed and started to blossom again thus indicating that they were living in healthy environmental conditions.

VI-References

[1] https://www.postscapes.com/smart-greenhouses/
[2] https://www.arduino.cc/en/guide/introduction
[3] https://store.arduino.cc/usa/mega-2560-r3
[4] https://forum.arduino.cc/index.php?topic=125908.15
[5] https://wiki.eprolabs.com/index.php?title=Moisture_Sensor
[6] https://www.circuito.io/app?components=512,11061
[7]https://electronicsforu.com/resources/electronics-components/humidity-sensor-basic-usage-parameter
[8] https://www.azosensors.com/article.aspx?ArticleID=335
[9] https://www.renewablesinafrica.com/how-does-a-solar-panel-work/