Table of Contents

Introduction:

The aim of our project is to provide help people to control consumptions at their houses more systematically and arrange their use of light, fan properly not wasting them. If more people save these values the country grows faster. Using the App, the project can be used in manual mode or automatic mode. With this type of an increase in control, people's consumptions like lights at home and air-conditioning decrease and saving increases.

Fundamental Concept and use of Sensors in our Project:

3 sensors are selected to be used at our project. These sensors are Rain Sensor, Temperature Sensor and LDR Sensor. Rain Sensor is used at our project, if it rains outside the house, it detects the wet and closes the window using Servo considering the system is running at the automatic mode. Temperature Sensor is used at our project because if the inside temperature is higer than the temperature measured inside the house, the fan starts working. For example, if the temperature is set equal to 35 Celsius, the fan won't work because the desired value is 35, the fan is used at the project to decrease the temperature. LDR is used at the project for the lights at the house to open and close by using Relay at Automatic Mode. GPS data from phone application is used for measuring distance between home and user. If user is in between desired radius. The fan and lights of home works accordingly.

Other parts that are used in our Project:

In order to use the data which is gathered using sensors, extra materials are also used at our project. The fan is used to decrease the ambient temperature regarding to set temperature from app. The Relay is used in order to see changes that occur using LDR Sensor and for the Fan to work. Servo is also used to open and close the window. SD cart is used to record the data while trying sensors to show that they are working. ESP8266 Wi-Fi Module is used to connect the Arduino Mega with the Cell Phone. For the design of the house, lot of material such as hot silicone, tapes and plastic in order to make the window are used during the project work.

Circuit Schematic:

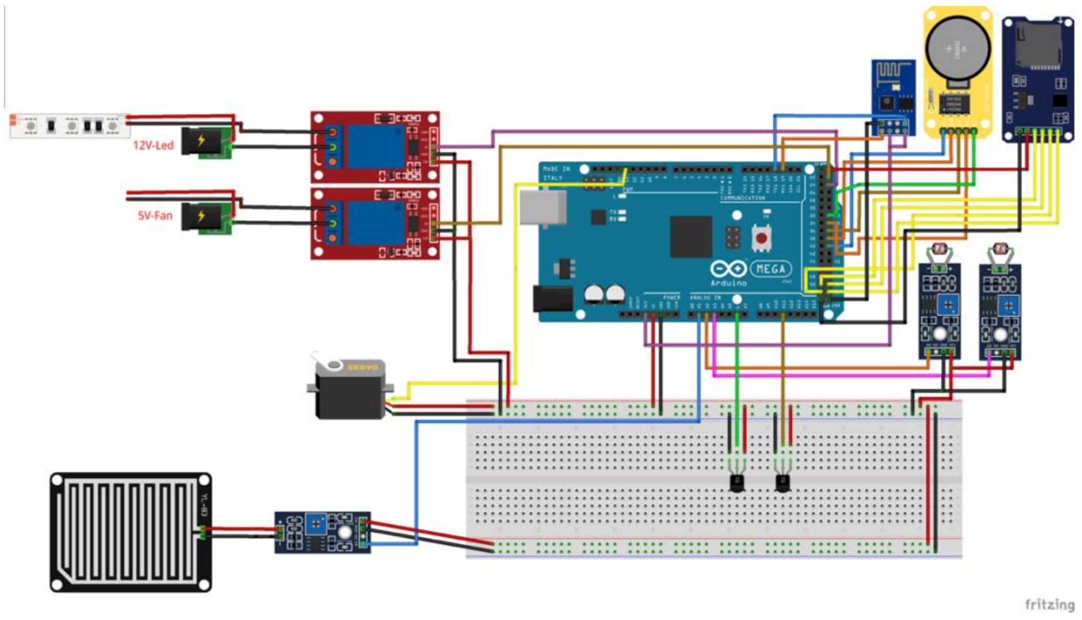
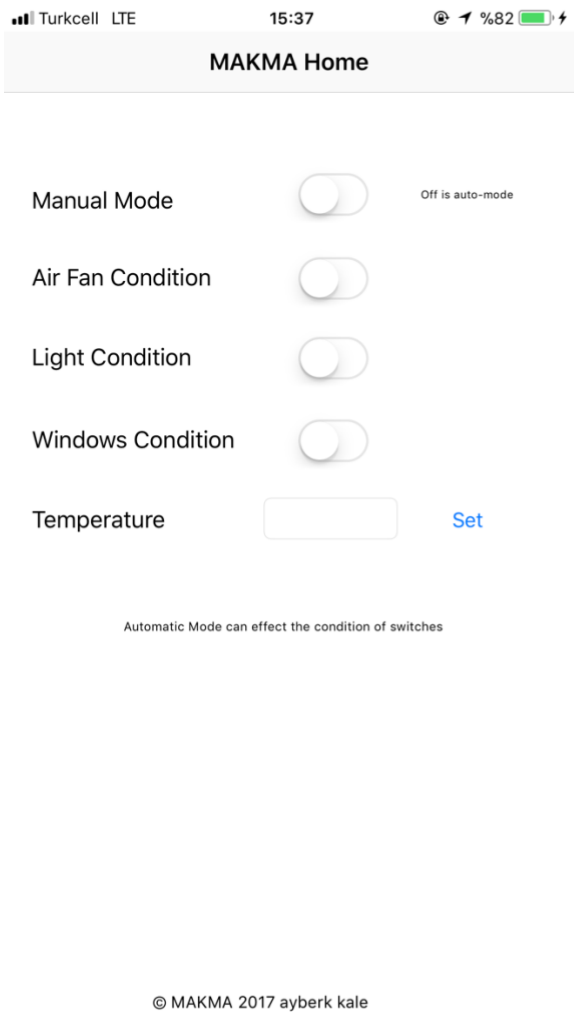


Figure 1: Schematic

Application for the Project:

Figure 2: Application Icon MAKMA Home


Turkcell LTE 15:37 %82

MAKMA Home

Manual Mode — Off is auto-mode

Air Fan Condition

Light Condition

Windows Condition

Temperature — Set

Automatic Mode can effect the condition of switches

© MAKMA 2017 ayberk kale

Figure 3: Application Interface

The application contains of ON/OFF buttons, Temperature Value Set, and background GPS sender.

Parts Used:

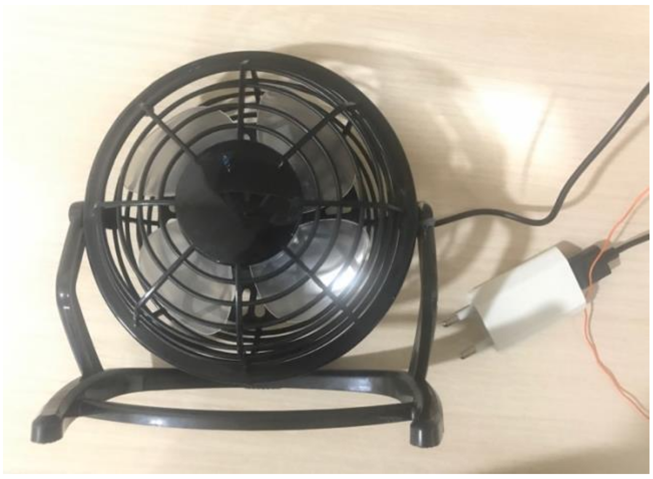


Figure 4:5V Air Fan

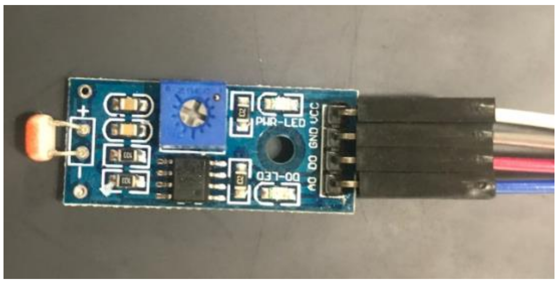


Figure 5: LDR with Control Module

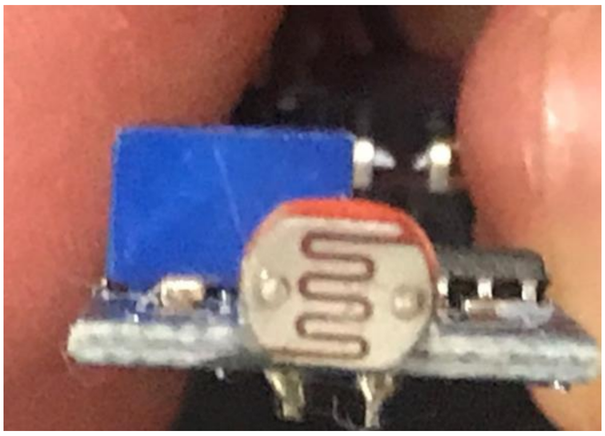


Figure 6:Light Dependent Resistor

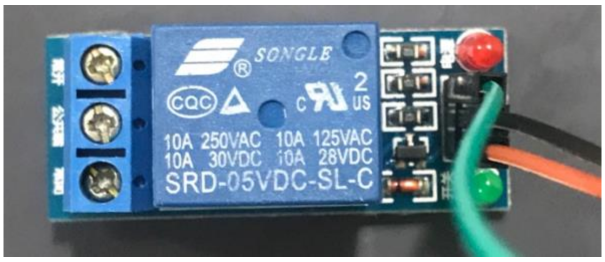


Figure 7: 5V Low Triggered Relay

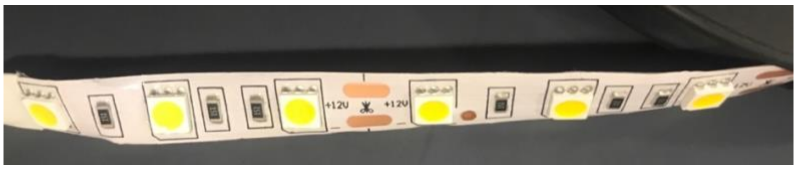


Figure 8:12 V LED Strip

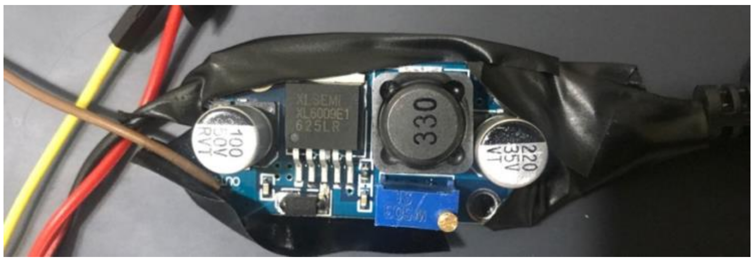


Figure 9:Step up converter used for 12 V to 12 V conversion

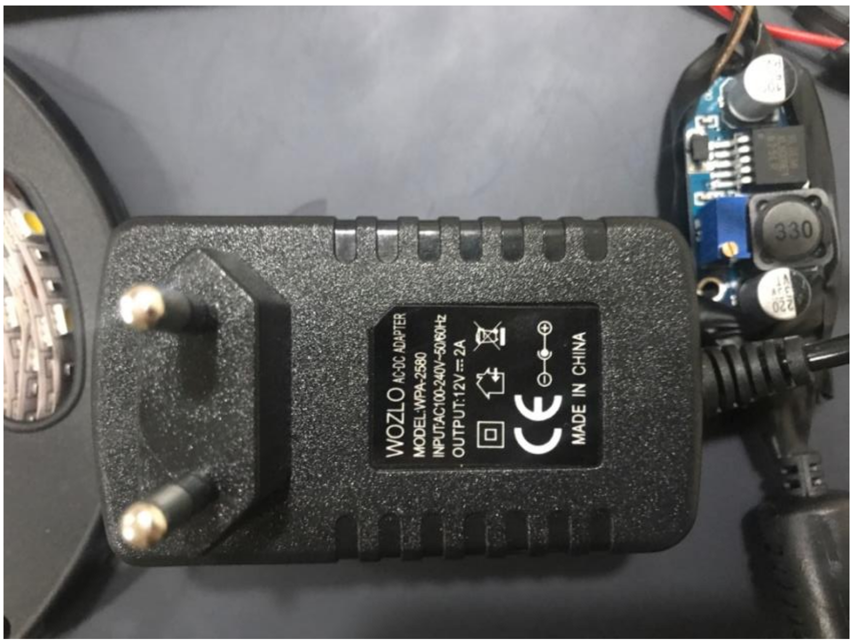


Figure 10:12V DC Adapter For LED Strip

Figure 11:LM35 Temperature Sensor

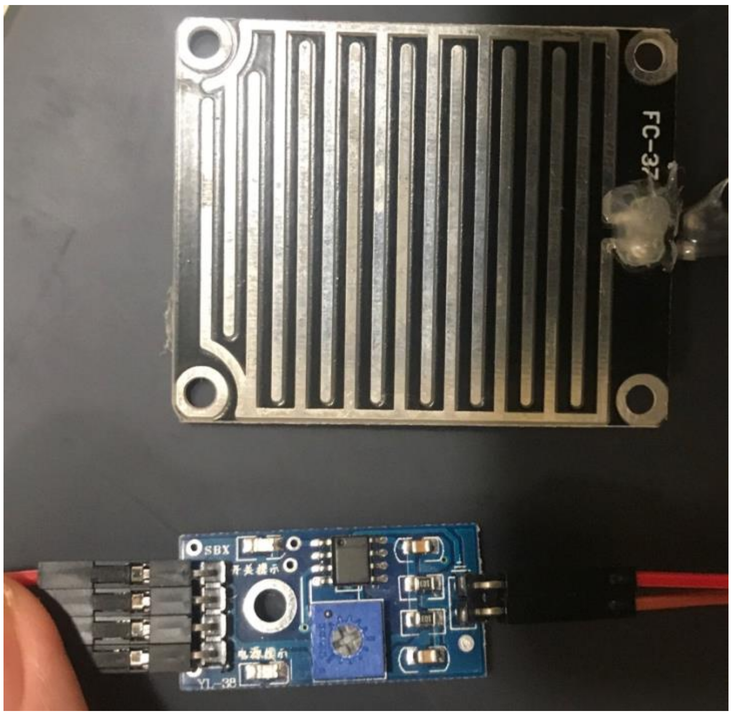


Figure 12: YL 83 Rain Sensor with Control Module

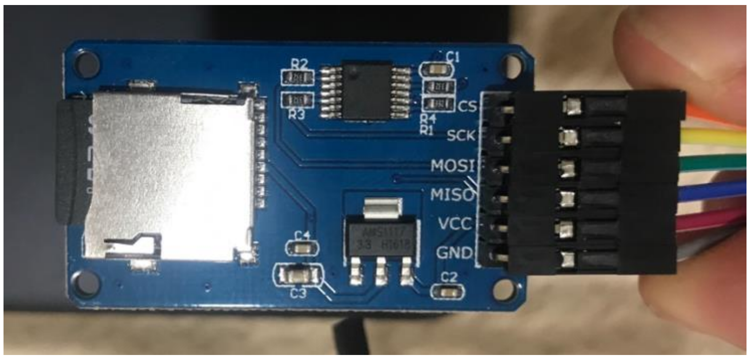


Figure 13: Micro SD Card Module

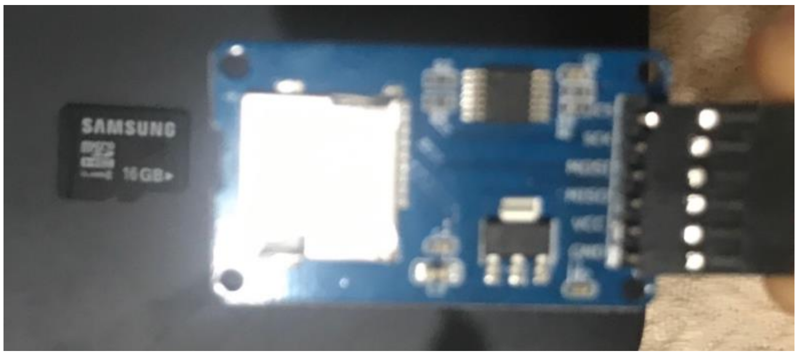


Figure 14: Micro SD Card

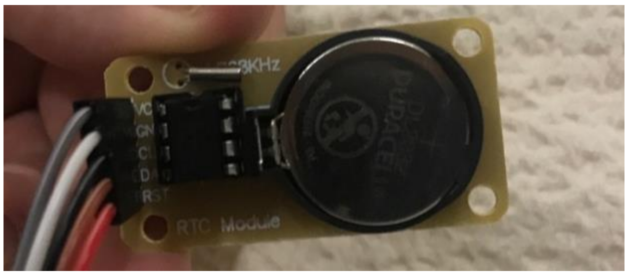


Figure 15:DS1302 RTC with Coin Cell

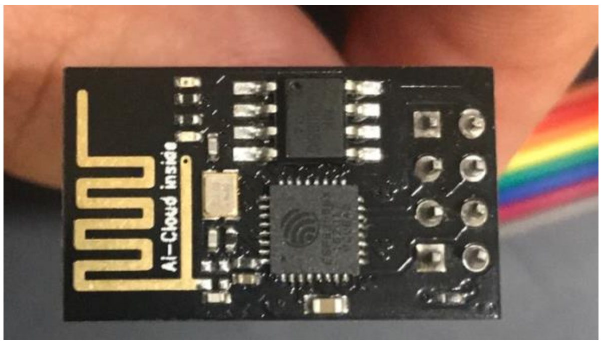


Figure 16: ESP 8266

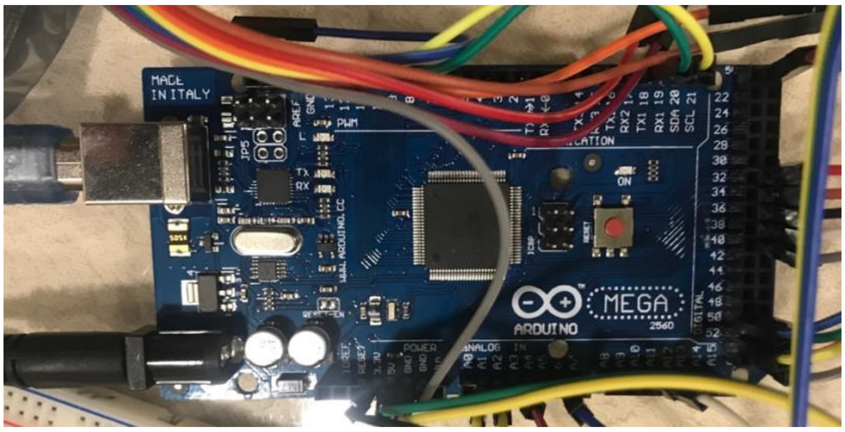


Figure 17: Arduino Mega 2560

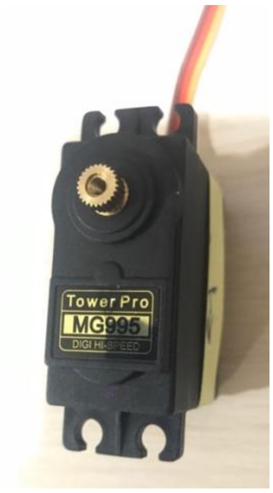


Figure 18:Metal Gear 995 Dual Ball Bearing Servo

Data Logging Sensors:

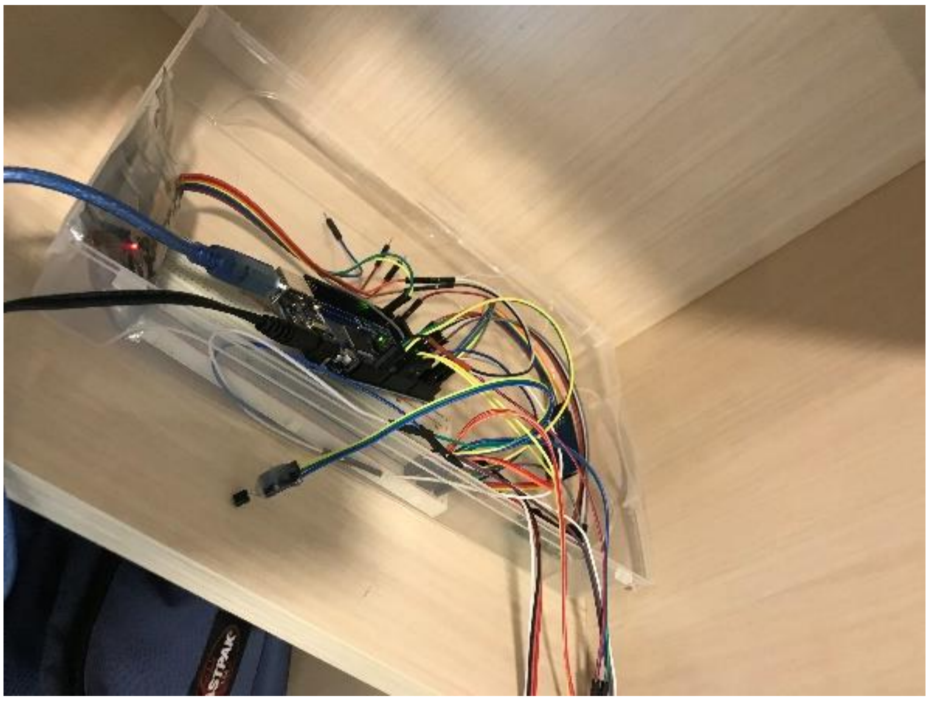


Figure 19: Data Logging Circuit

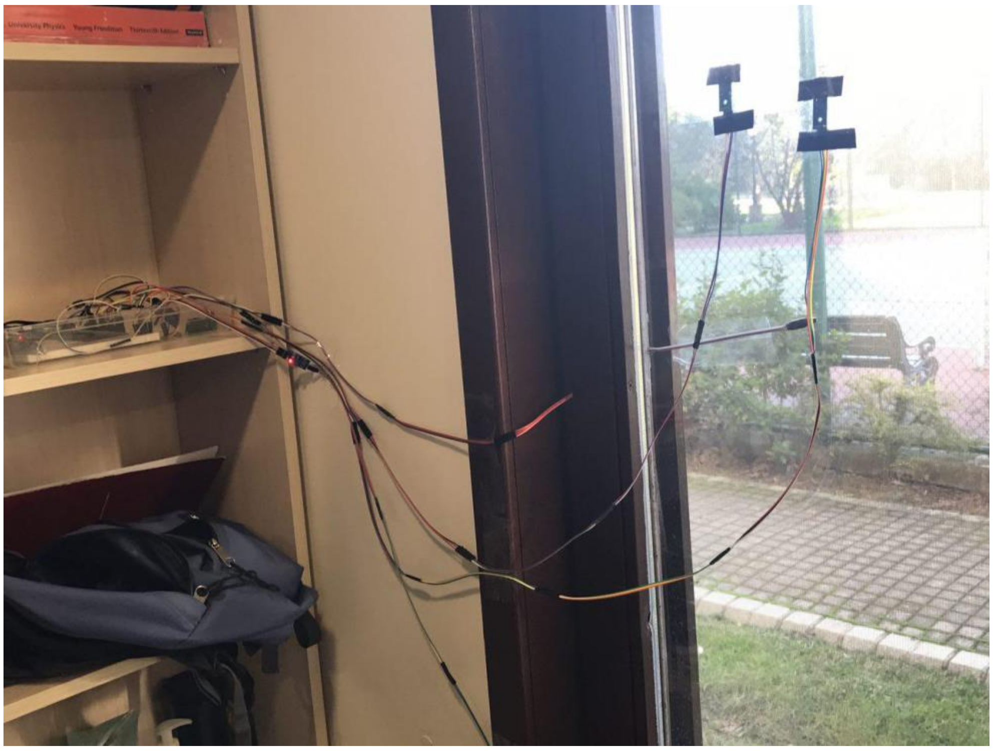


Figure 20: LDR and LM35 is attached to window in order to collect real world data

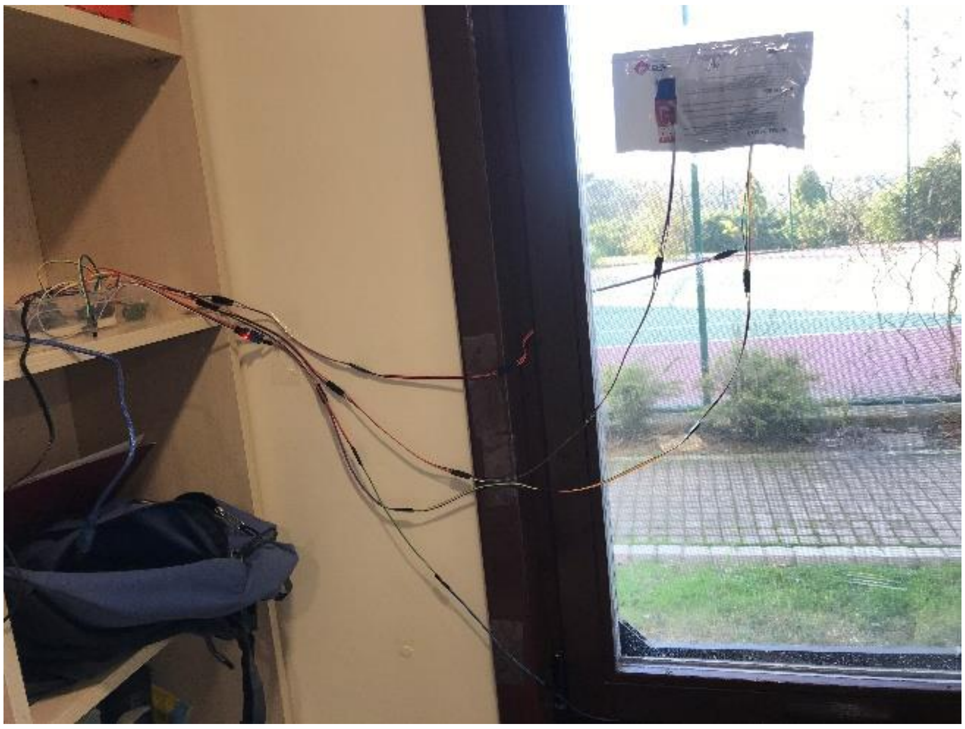


Figure 21: LDR's are covered from inside in order to prevent light leakage from room.

Figure 22: Rain Sensor Attached Outside to collect rain/water data

House Design:

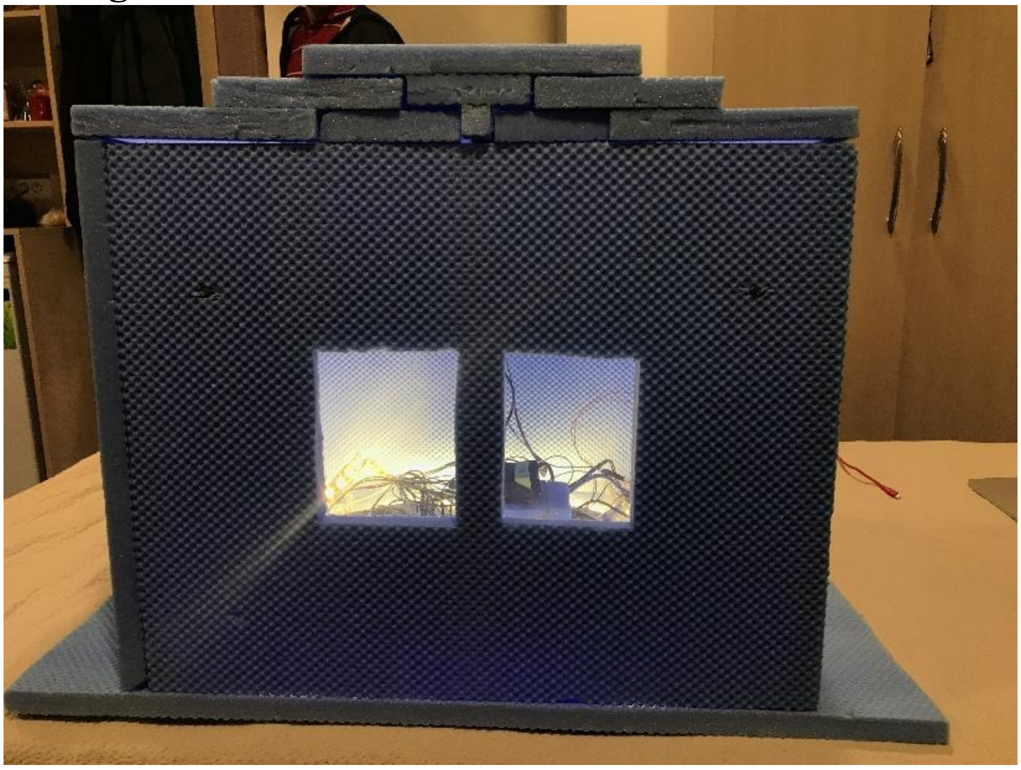


Figure 23:Front view of model LED is on

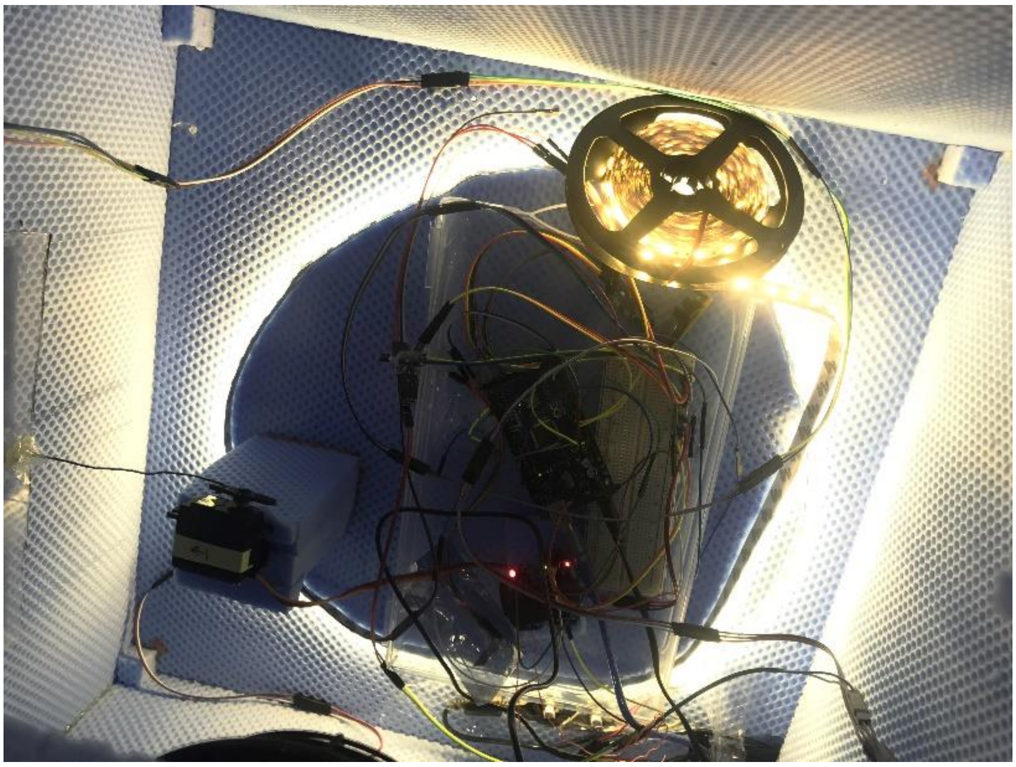


Figure 24: Inside View of Model

Plots and Analysis:

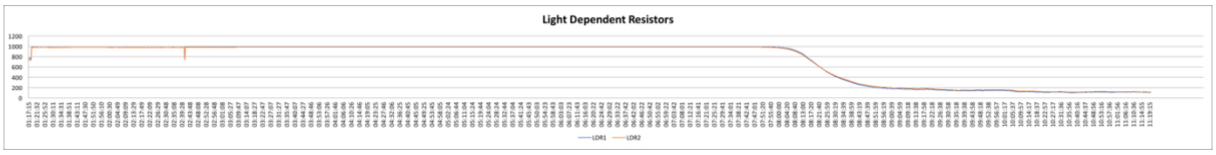


Figure 27: LDR Data from 27.12.2017 Sun Set Bits versus Time

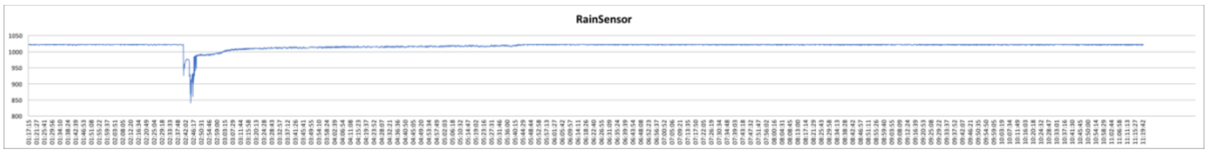


Figure 28: Rain Sensor Data from 27.12.2017 Bits versus Time

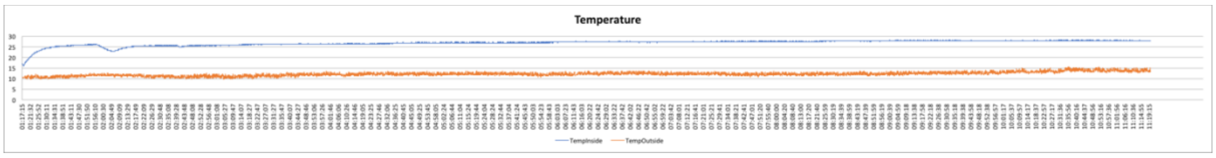
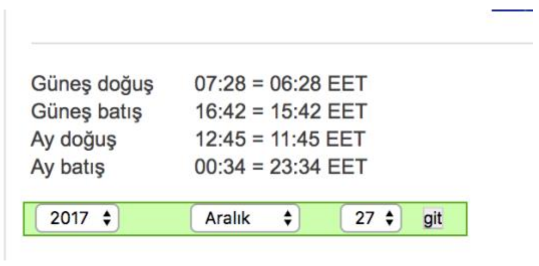


Figure 29:Temperature Data ℃ versus Time

In figure 27,28 and 29 the collected data from 27.12.2017 01:17 to 11:19 have been plotted. At figure 28 which represents the voltage data mapped to 0 to 1023 bits from Rain Sensor were plotted. At 02:40 the rain sensor has been wet and the read voltage dropped according to amount of water touched the surface of the sensor. In figure 29, the upper samples are from inside room temperature, and the below one is the temperature from outside. The temperature from outside increased to 15º from 10º after the sun set. In the figure 27 the LDR output voltage value to 0-1023 bits mapped values are plotted. When the sun set the output voltage dropped accordingly in exponential way. Near 7:30 sun start to set which is similar to what we found out from the web(figure 30).



Güneş doğuş	07:28 = 06:28 EET
Güneş batış	16:42 = 15:42 EET
Ay doğuş	12:45 = 11:45 EET
Ay batış	00:34 = 23:34 EET

Figure 30: Sariyer Sunset and Sundown Data: https://www.havaturkiye.com/weather/maps/city

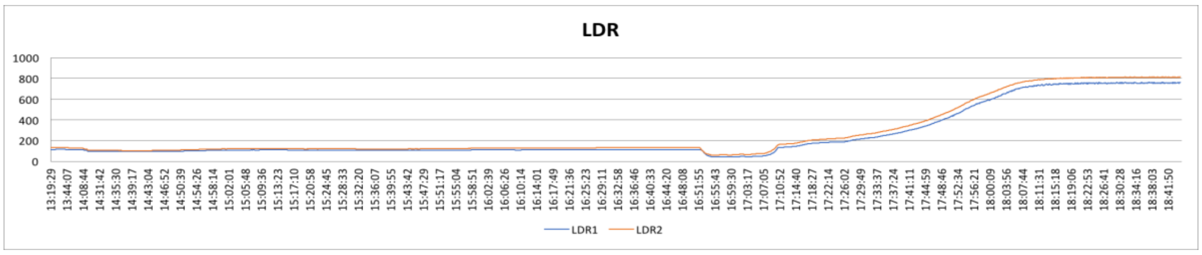


Figure 31:LDR Data from 26.12.2017 Sun Down Bits versus Time

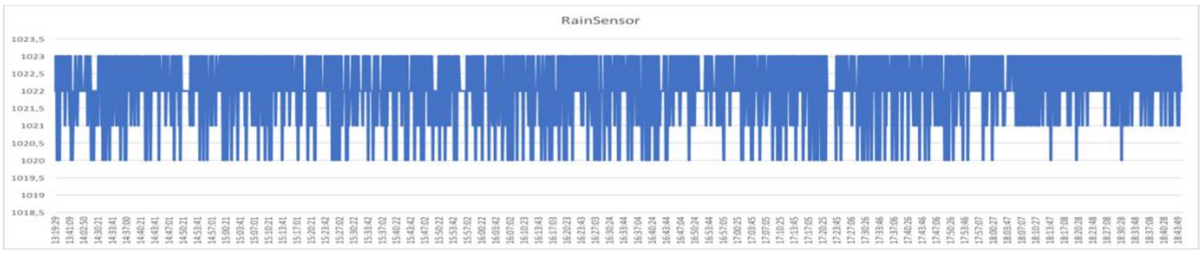


Figure 32:Rain Data From 26.12.2017 Bits versus Time

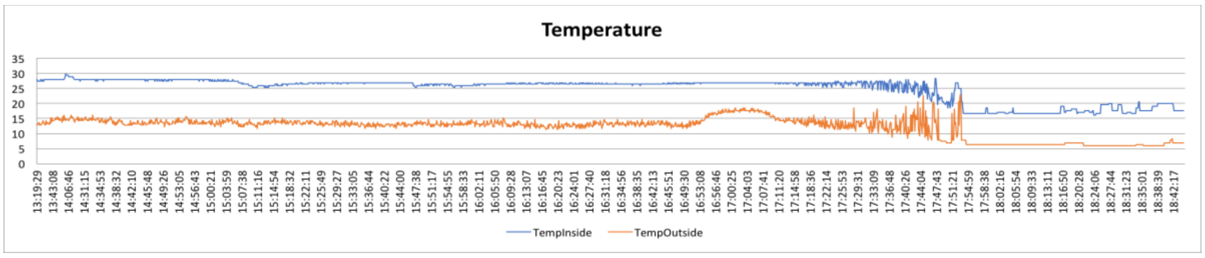


Figure 33:Temperature Data 26.12.2017 ℃ versus Time

Also from figure 31,32 and 33 the data stored is consistent all time. The peak change at figure a is result of change of curtains of room. Outside temperature also measure at near 10ºC which is correct when checked from web. The Temperature inconsistency solved in 27.12.2017.

From above figure c can be understand that the LDR value increased due to sun down condition near 17.30, which is consistent to figure v.

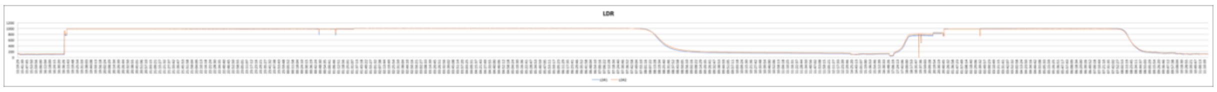


Figure 34:All Measure LDR Values

LDR sensor works as expected. At night it has high voltage output and at day time it decreased to near 200 bits.

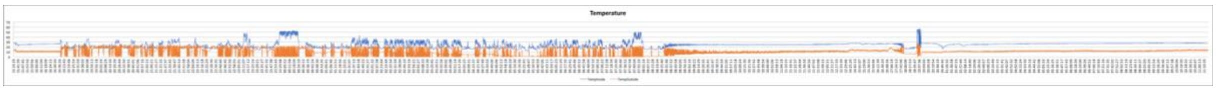


Figure 35:All Measured Temperature Values

Temperature data has some inconsistency in some parts in earlier sampling times but by increasing sample amount at one time solved the problem. The temperature data has sampled three times from analog pin and the last measured one is saved as measurements.

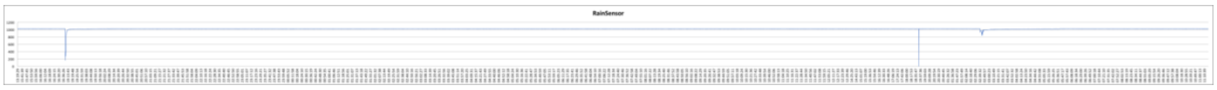


Figure 36:All Measured Rain Values

Rain sensor works as expected. At dry condition, it has high voltage output and at wet condition it decreased to near 200 bits.

The bits are 0 to 5volts which mapped to 0 to 1023 bits with analogRead() function. The read voltage bit turned it ºC with below equations.

float analogVoutside = analogRead(tempPinOutside);

analogVoutside = (analogVoutside / 1023) * 5000; //convert analog to mili voltage

TemperatureMeasuredOutside = analogVoutside / 10, 0; // mV to temperature

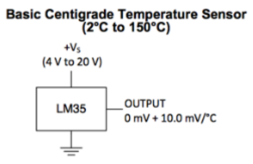
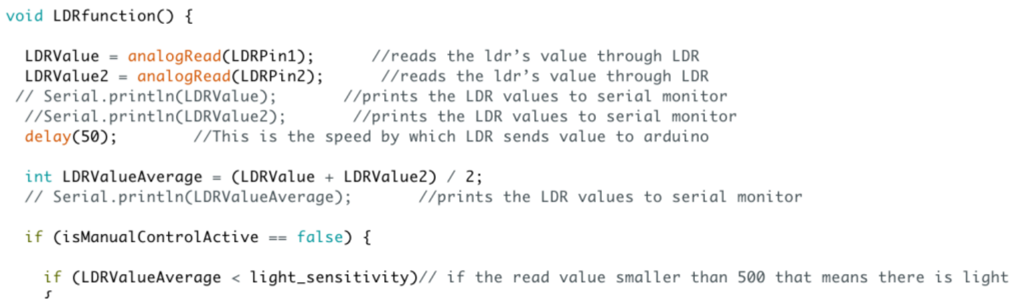
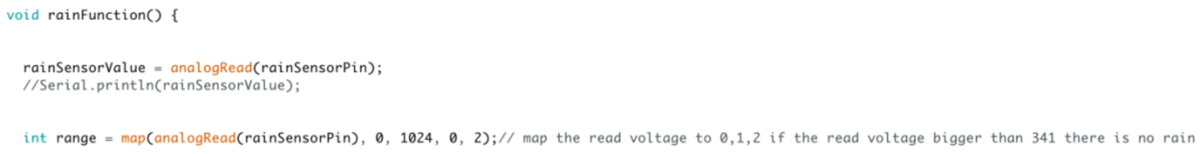


Figure 37: LM35 Output Voltage



```
void LDRfunction() {

  LDRValue = analogRead(LDRPin1);      //reads the ldr's value through LDR
  LDRValue2 = analogRead(LDRPin2);      //reads the ldr's value through LDR
 // Serial.println(LDRValue);       //prints the LDR values to serial monitor
  //Serial.println(LDRValue2);       //prints the LDR values to serial monitor
  delay(50);        //This is the speed by which LDR sends value to arduino

  int LDRValueAverage = (LDRValue + LDRValue2) / 2;
  // Serial.println(LDRValueAverage);       //prints the LDR values to serial monitor

  if (isManualControlActive == false) {

    if (LDRValueAverage < light_sensitivity)// if the read value smaller than 500 that means there is light
    {
```

Figure 38:LDR Output Handling

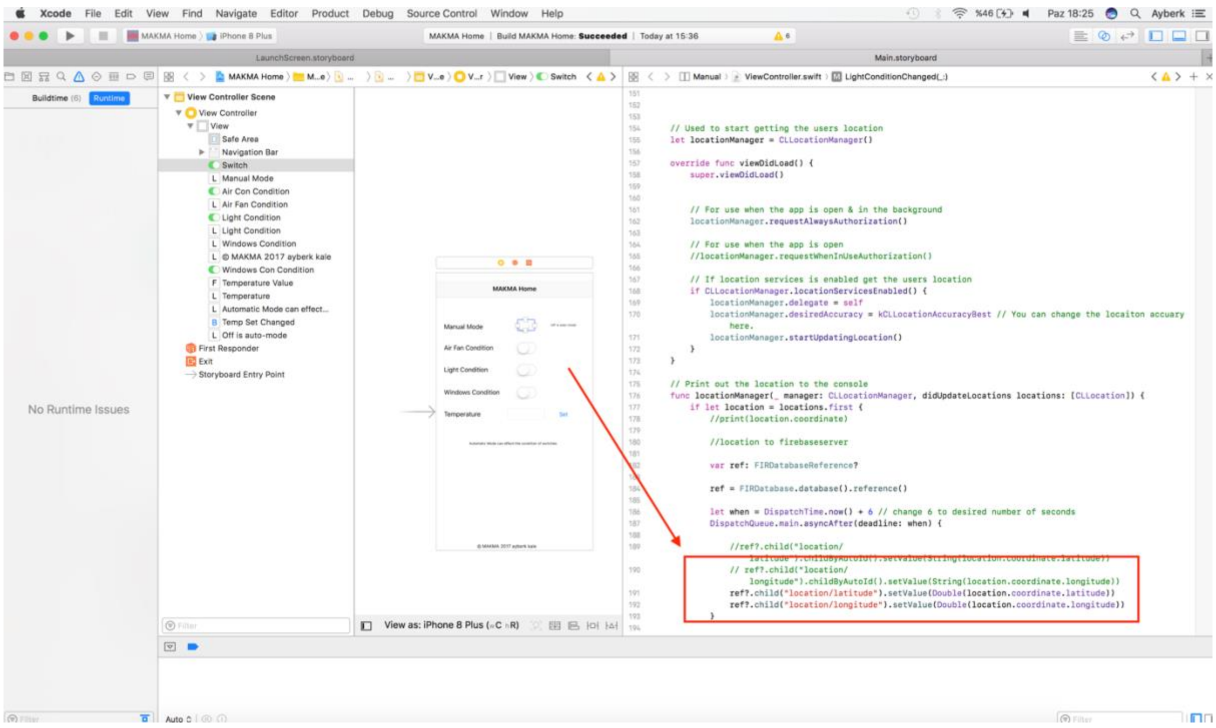


```
void rainFunction() {

  rainSensorValue = analogRead(rainSensorPin);
  //Serial.println(rainSensorValue);

  int range = map(analogRead(rainSensorPin), 0, 1024, 0, 2);// map the read voltage to 0,1,2 if the read voltage bigger than 341 there is no rain
```

Figure 39:RainSensor Output Handling

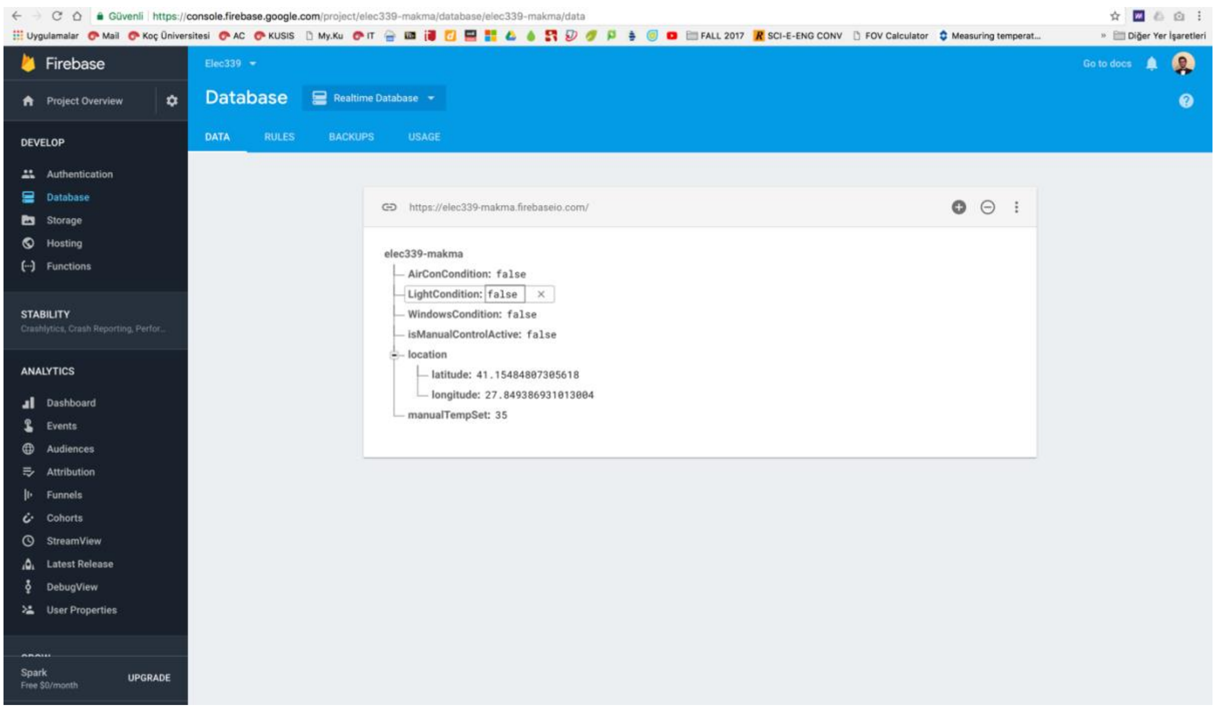
Wireless Communication Working Principle:

The data from app such as button conditions, temperature set value and location sent to Firebase server with Swift Xcode iPhone Application.



```
// Used to start getting the users location
let locationManager = CLLocationManager()

override func viewDidLoad() {
    super.viewDidLoad()


    // For use when the app is open & in the background
    locationManager.requestAlwaysAuthorization()

    // For use when the app is open
    //locationManager.requestWhenInUseAuthorization()

    // If location services is enabled get the users location
    if CLLocationManager.locationServicesEnabled() {
        locationManager.delegate = self
        locationManager.desiredAccuracy = kCLLocationAccuracyBest // You can change the locaiton accuary
            here.
        locationManager.startUpdatingLocation()
    }
}

// Print out the location to the console
func locationManager(_ manager: CLLocationManager, didUpdateLocations locations: [CLLocation]) {
    if let location = locations.first {
        //print(location.coordinate)

        //location to firebaseserver

        var ref: FIRDatabaseReference?

        ref = FIRDatabase.database().reference()

        let when = DispatchTime.now() + 6 // change 6 to desired number of seconds
        DispatchQueue.main.asyncAfter(deadline: when) {

            //ref?.child("location/
                latitude").childByAutoId().setValue(String(location.coordinate.latitude))
            // ref?.child("location/
                longitude").childByAutoId().setValue(String(location.coordinate.longitude))
            ref?.child("location/latitude").setValue(Double(location.coordinate.latitude))
            ref?.child("location/longitude").setValue(Double(location.coordinate.longitude))
        }
```

Figure 40:Xcode Application Code

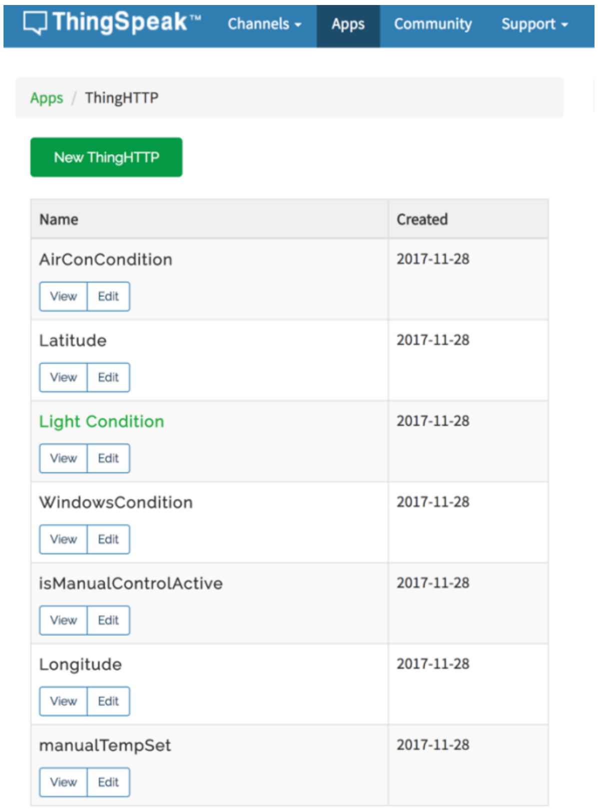


https://elec339-makma.firebaseio.com/

```
elec339-makma
├─ AirConCondition: false
├─ LightCondition: false
├─ WindowsCondition: false
├─ isManualControlActive: false
├─ location
│   ├─ latitude: 41.15484807305618
│   └─ longitude: 27.849386931013004
└─ manualTempSet: 35
```

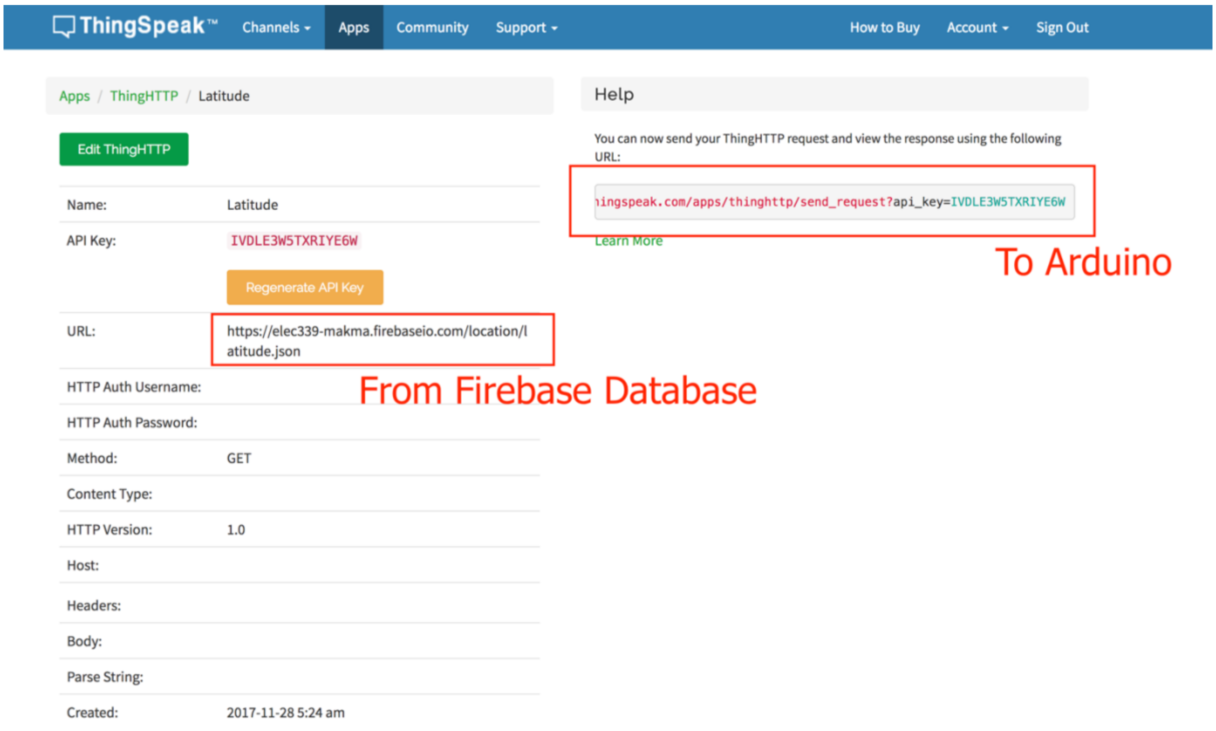
Figure 41:Firebase Database Values From App

The data at Firebase pulled to Thingspeak using ThingHTTP via Firebase REST API .json format.



Name	Created
AirConCondition	2017-11-28
Latitude	2017-11-28
Light Condition	2017-11-28
WindowsCondition	2017-11-28
isManualControlActive	2017-11-28
Longitude	2017-11-28
manualTempSet	2017-11-28

Figure 42:All ThingHTTP channels at Thingspeak.com

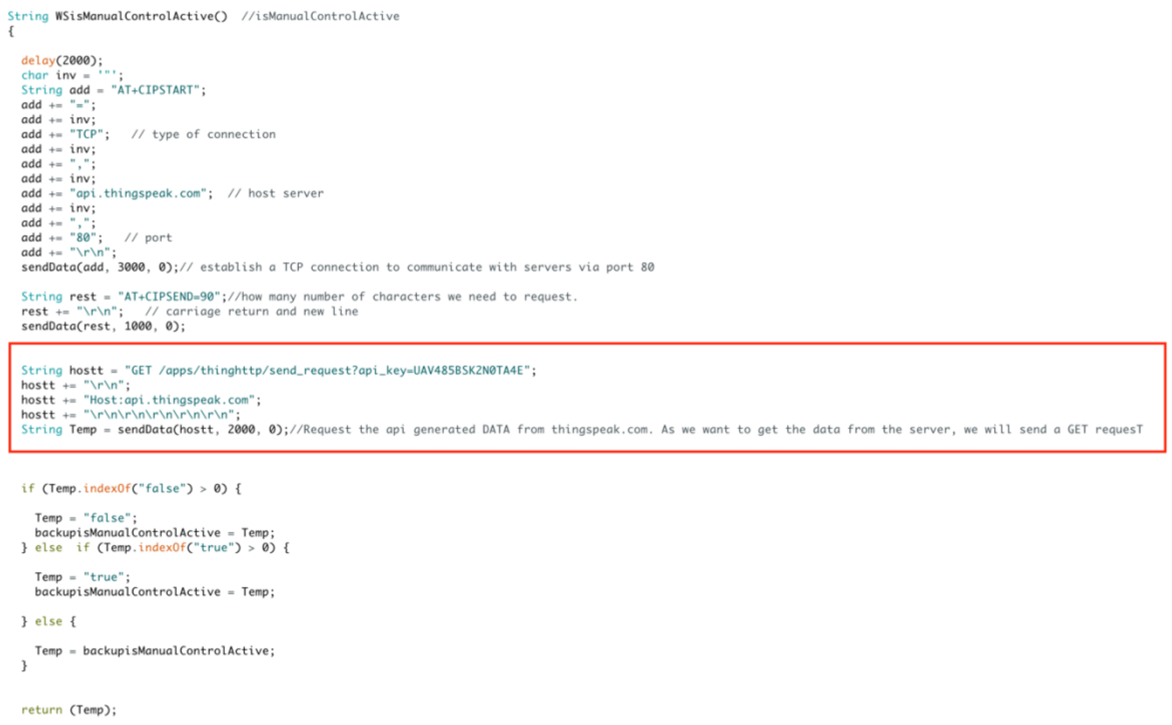


Name:	Latitude
API Key:	IVDLE3W5TXRIYE6W
URL:	https://elec339-makma.firebaseio.com/location/latitude.json
HTTP Auth Username:	
HTTP Auth Password:	
Method:	GET
Content Type:	
HTTP Version:	1.0
Host:	
Headers:	
Body:	
Parse String:	
Created:	2017-11-28 5:24 am

You can now send your ThingHTTP request and view the response using the following URL:

```
ingspeak.com/apps/thinghttp/send_request?api_key=IVDLE3W5TXRIYE6W
```

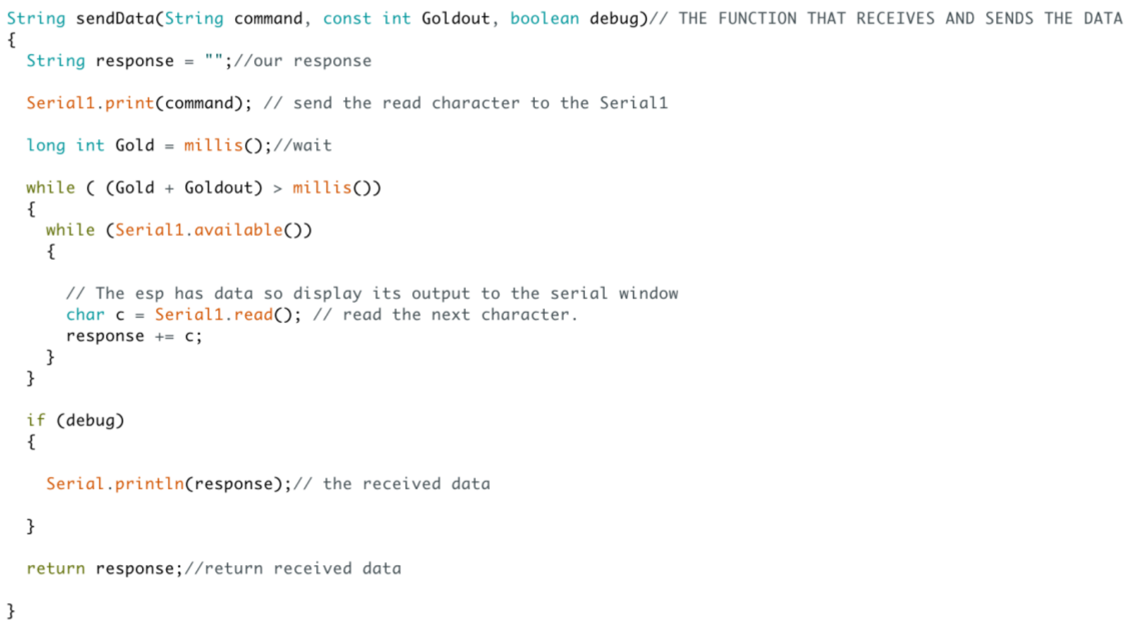
Figure 43:Thingspeak Example HTTP and API Key , Latiude Data



```
String WSisManualControlActive()  //isManualControlActive
{

  delay(2000);
  char inv = '"';
  String add = "AT+CIPSTART";
  add += "=";
  add += inv;
  add += "TCP";   // type of connection
  add += inv;
  add += ",";
  add += inv;
  add += "api.thingspeak.com";  // host server
  add += inv;
  add += ",";
  add += "80";   // port
  add += "\r\n";
  sendData(add, 3000, 0);// establish a TCP connection to communicate with servers via port 80

  String rest = "AT+CIPSEND=90";//how many number of characters we need to request.
  rest += "\r\n";   // carriage return and new line
  sendData(rest, 1000, 0);

  String hostt = "GET /apps/thinghttp/send_request?api_key=UAV485BSK2N0TA4E";
  hostt += "\r\n";
  hostt += "Host:api.thingspeak.com";
  hostt += "\r\n\r\n\r\n\r\n\r\n";
  String Temp = sendData(hostt, 2000, 0);//Request the api generated DATA from thingspeak.com. As we want to get the data from the server, we will send a GET requesT

  if (Temp.indexOf("false") > 0) {

    Temp = "false";
    backupisManualControlActive = Temp;
  } else  if (Temp.indexOf("true") > 0) {

    Temp = "true";
    backupisManualControlActive = Temp;

  } else {

    Temp = backupisManualControlActive;
  }


  return (Temp);
```

Figure 44: Thingspeak Handling on Arduino Side

After that GET request sent to server using API Key of HTTP instance using TCP connection to Thingspeak.



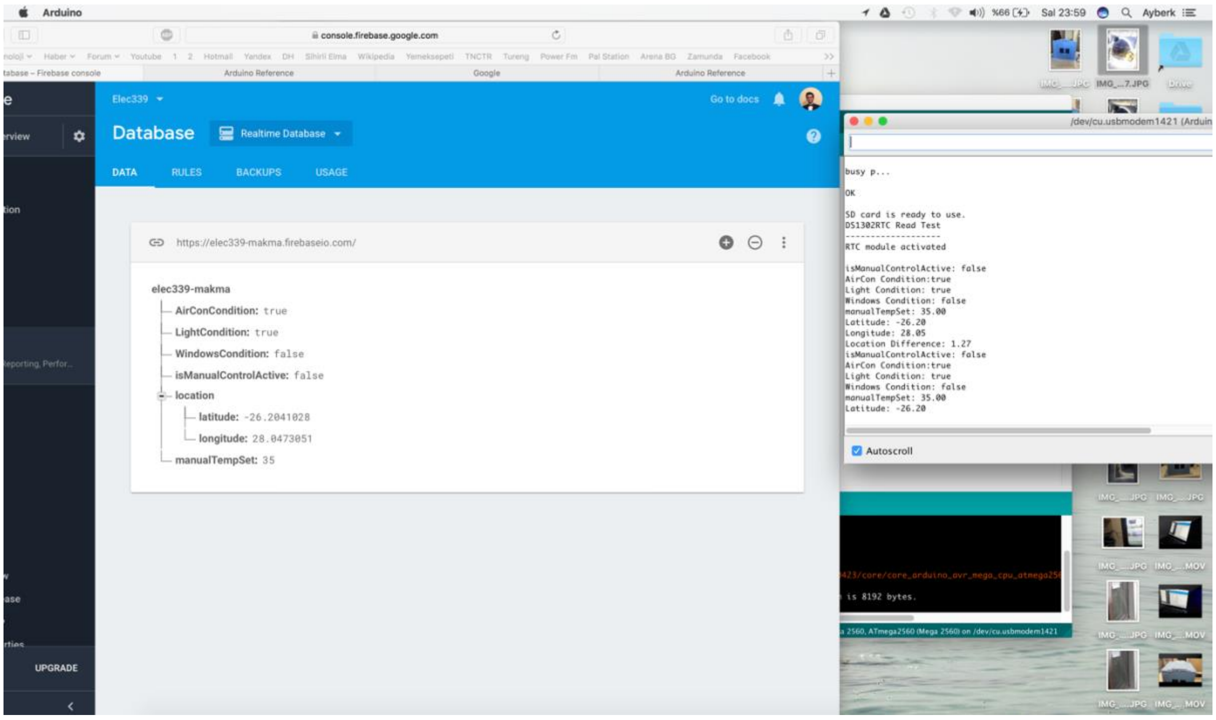
```
String sendData(String command, const int Goldout, boolean debug)// THE FUNCTION THAT RECEIVES AND SENDS THE DATA
{
  String response = "";//our response

  Serial1.print(command); // send the read character to the Serial1

  long int Gold = millis();//wait

  while ( (Gold + Goldout) > millis())
  {
    while (Serial1.available())
    {

      // The esp has data so display its output to the serial window
      char c = Serial1.read(); // read the next character.
      response += c;
    }
  }

  if (debug)
  {

    Serial.println(response);// the received data

  }

  return response;//return received data

}
```

Figure 45:ESP8266 Handling to retrieve data from Thingspeak from serial port

And received response from server read from serial1 (ESP8266) port. After that received data has filtered to retract the wanted string. Then the retrived value is converted Boolean, double or string regarding to which one is desired.

For latitude and longitude to house distance below formula is used.

```
void LocationDifferenceCalculator() {// location difference calculator in terms from latitude and longitude
  float pi = 3.14159265359;
  float R = 6371;  //kilometres earths radius
  float latr1 = userLocationLat * (pi / 180);
  float latr2 = CompLocationLat * (pi / 180);
  float deltaLat = (CompLocationLat - userLocationLat) * (pi / 180);
  float deltaLong = (userLocationLong - CompLocationLong) * (pi / 180);

  float a = (sin(deltaLat / 2) * sin(deltaLat / 2)) + cos(latr1) * cos(latr2) * sin(deltaLong / 2) *
sin(deltaLong / 2);

  float c = 2 * atan2(sqrt(a), sqrt(1 - a));
  LocationDifference =  R * c;
}
```

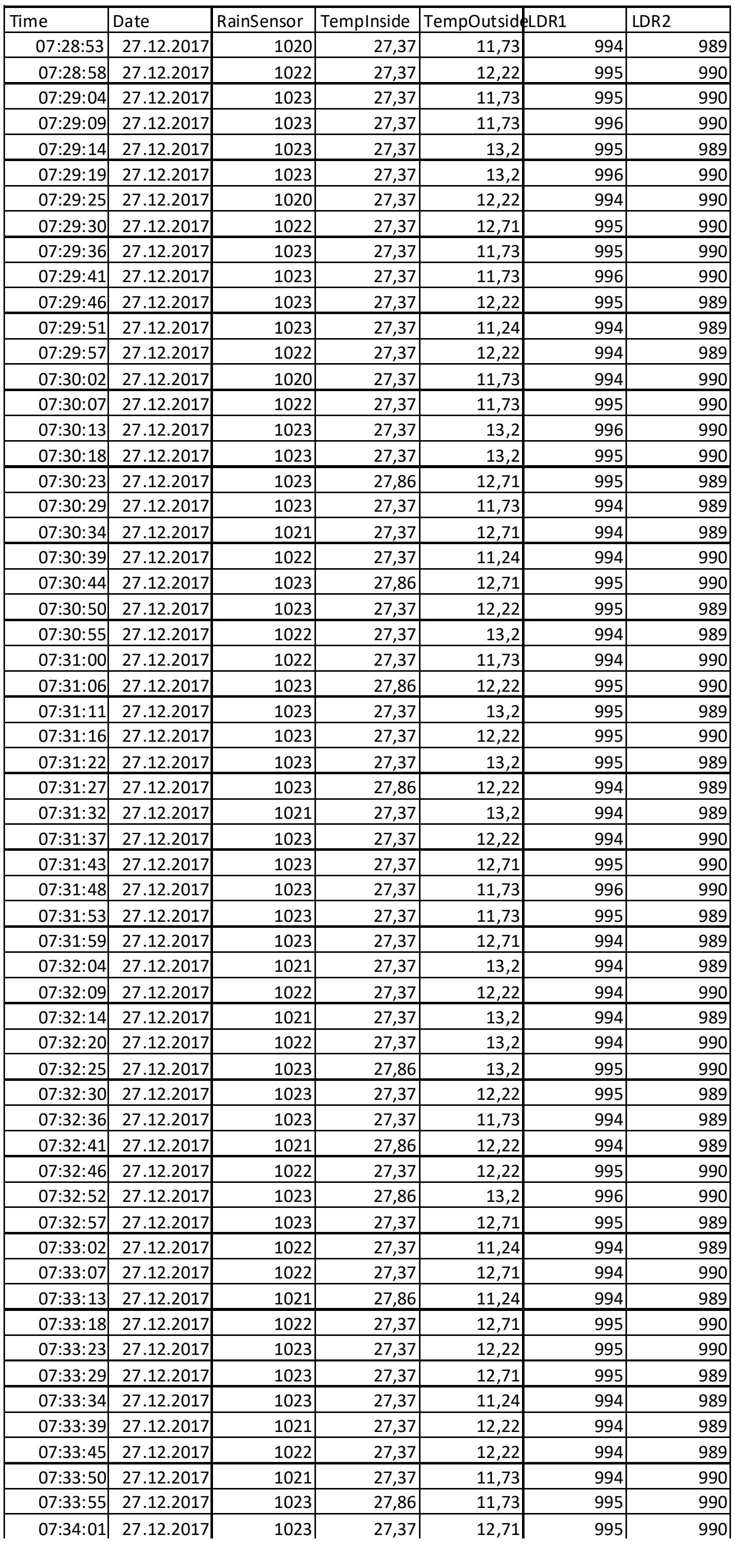


```
https://elec339-makma.firebaseio.com/

elec339-makma
  AirConCondition: true
  LightCondition: true
  WindowsCondition: false
  isManualControlActive: false
  location
    latitude: -26.2041028
    longitude: 28.0473051
  manualTempSet: 35
```

```
busy p...

OK

SD card is ready to use.
DS1302RTC Read Test
--------------------
RTC module activated

isManualControlActive: false
AirCon Condition:true
Light Condition: true
Windows Condition: false
manualTempSet: 35.00
Latitude: -26.20
Longitude: 28.05
Location Difference: 1.27
isManualControlActive: false
AirCon Condition:true
Light Condition: true
Windows Condition: false
manualTempSet: 35.00
Latitude: -26.20
```

Figure 46:Firebase Database and Data from Arduino Serial Plotter

Sample Data:



Time	Date	RainSensor	TempInside	TempOutside	LDR1	LDR2
07:28:53	27.12.2017	1020	27,37	11,73	994	989
07:28:58	27.12.2017	1022	27,37	12,22	995	990
07:29:04	27.12.2017	1023	27,37	11,73	995	990
07:29:09	27.12.2017	1023	27,37	11,73	996	990
07:29:14	27.12.2017	1023	27,37	13,2	995	989
07:29:19	27.12.2017	1023	27,37	13,2	996	990
07:29:25	27.12.2017	1020	27,37	12,22	994	990
07:29:30	27.12.2017	1022	27,37	12,71	995	990
07:29:36	27.12.2017	1023	27,37	11,73	995	990
07:29:41	27.12.2017	1023	27,37	11,73	996	990
07:29:46	27.12.2017	1023	27,37	12,22	995	989
07:29:51	27.12.2017	1023	27,37	11,24	994	989
07:29:57	27.12.2017	1022	27,37	12,22	994	989
07:30:02	27.12.2017	1020	27,37	11,73	994	990
07:30:07	27.12.2017	1022	27,37	11,73	995	990
07:30:13	27.12.2017	1023	27,37	13,2	996	990
07:30:18	27.12.2017	1023	27,37	13,2	995	990
07:30:23	27.12.2017	1023	27,86	12,71	995	989
07:30:29	27.12.2017	1023	27,37	11,73	994	989
07:30:34	27.12.2017	1021	27,37	12,71	994	989
07:30:39	27.12.2017	1022	27,37	11,24	994	990
07:30:44	27.12.2017	1023	27,86	12,71	995	990
07:30:50	27.12.2017	1023	27,37	12,22	995	989
07:30:55	27.12.2017	1022	27,37	13,2	994	989
07:31:00	27.12.2017	1022	27,37	11,73	994	990
07:31:06	27.12.2017	1023	27,86	12,22	995	990
07:31:11	27.12.2017	1023	27,37	13,2	995	989
07:31:16	27.12.2017	1023	27,37	12,22	995	990
07:31:22	27.12.2017	1023	27,37	13,2	995	989
07:31:27	27.12.2017	1023	27,86	12,22	994	989
07:31:32	27.12.2017	1021	27,37	13,2	994	989
07:31:37	27.12.2017	1023	27,37	12,22	994	990
07:31:43	27.12.2017	1023	27,37	12,71	995	990
07:31:48	27.12.2017	1023	27,37	11,73	996	990
07:31:53	27.12.2017	1023	27,37	11,73	995	989
07:31:59	27.12.2017	1023	27,37	12,71	994	989
07:32:04	27.12.2017	1021	27,37	13,2	994	989
07:32:09	27.12.2017	1022	27,37	12,22	994	990
07:32:14	27.12.2017	1021	27,37	13,2	994	989
07:32:20	27.12.2017	1022	27,37	13,2	994	990
07:32:25	27.12.2017	1023	27,86	13,2	995	990
07:32:30	27.12.2017	1023	27,37	12,22	995	989
07:32:36	27.12.2017	1023	27,37	11,73	994	989
07:32:41	27.12.2017	1021	27,86	12,22	994	989
07:32:46	27.12.2017	1022	27,37	12,22	995	990
07:32:52	27.12.2017	1023	27,86	13,2	996	990
07:32:57	27.12.2017	1023	27,37	12,71	995	989
07:33:02	27.12.2017	1022	27,37	11,24	994	989
07:33:07	27.12.2017	1022	27,37	12,71	994	990
07:33:13	27.12.2017	1021	27,86	11,24	994	989
07:33:18	27.12.2017	1022	27,37	12,71	995	990
07:33:23	27.12.2017	1023	27,37	12,22	995	990
07:33:29	27.12.2017	1023	27,37	12,71	995	989
07:33:34	27.12.2017	1023	27,37	11,24	994	989
07:33:39	27.12.2017	1021	27,37	12,22	994	989
07:33:45	27.12.2017	1022	27,37	12,22	994	989
07:33:50	27.12.2017	1021	27,37	11,73	994	990
07:33:55	27.12.2017	1023	27,86	11,73	995	990
07:34:01	27.12.2017	1023	27,37	12,71	995	990

Figure 47:Sample Data from 27.12

All sample data: https://drive.google.com/a/ku.edu.tr/file/d/12_cUcf1F59r_zswV-PcSC61phOj_h-50/view?usp=sharing

References:

- LDR:
  - http://www.instructables.com/id/LDR-Sensor-Module-Users-Manual-V10/
  - http://kennarar.vma.is/thor/v2011/vgr402/ldr.pdf

- LM35:
  - http://maker.robotistan.com/arduino-dersleri-11-sicaklik-olcumu/
  - https://ieeeras.ku.edu.tr/dokumanlar/
  - https://www.arduino.cc/en/Tutorial/ReadAnalogVoltage
  - http://www.ti.com/lit/ds/symlink/lm35.pdf

- Rain Sensor:
  - http://www.instructables.com/id/Arduino-Modules-Rain-Sensor/
  - http://henrysbench.capnfatz.com/henrys-bench/arduino-sensors-and-input/arduino-rain-sensor-module-guide-and-tutorial/
  - https://www.openhacks.com/uploadsproductos/rain_sensor_module.pdf

- Servo:
  - https://ieeeras.ku.edu.tr/dokumanlar/
  - http://www.electronicoscaldas.com/datasheet/MG995_Tower-Pro.pdf

- Relay:
  - http://www.projehocam.com/arduino-role-uygulamasi/

- ESP8266:
  - http://maker.robotistan.com/esp8266-ile-iot-dersleri-1-esp8266-modulunu-guncelleme/
  - http://www.instructables.com/id/noobs-guide-to-ESP8266-with-Arduino-Mega-2560-or-U/
  - http://www.instructables.com/id/ESP8266-Wifi-Tutorial/

- Data: Fetching From Server API:
  - http://techiesms.com/fetching-data-from-any-website/
  - https://www.youtube.com/watch?v=4vKxGHGYOtI
  - https://www.mathworks.com/help/thingspeak/thinghttp-app.html

- Mobile Application with Firebase Database:
    - https://www.youtube.com/watch?v=1deyxn5jVXk
    - https://www.youtube.com/watch?v=2GCvqfRCn58
    - https://www.youtube.com/watch?v=PHWMj152AMc
    - https://firebase.google.com/docs/reference/swift/firebasecore/api/reference/Classes?authuser=0
    - https://firebase.google.com/docs/reference/rest/database/

- SD Card Module:
    - http://howtomechatronics.com/tutorials/arduino/arduino-sd-card-data-logging-excel-tutorial/

- DS 1302 RTC:
    - http://www.instructables.com/id/Real-Time-Clock-DS1302/
    - http://playground.arduino.cc/Main/DS1302

- Calculate distance, bearing and more between Latitude/Longitude points
    - https://www.movable-type.co.uk/scripts/latlong.html

Source Code:

- https://drive.google.com/drive/folders/13xvIGi1Kzl2oEcoevbkYgs89ilnOFwIw?usp=sharing

Project Video:

- https://drive.google.com/a/ku.edu.tr/file/d/1MLxyJ2DMZZlCk-PdCqYPSQtWJTU6pmU4/view?usp=sharing