Embedded Systems Final Project
(Intelligent Kids Room)

Professor

Dr.Hosseini

Group Members

Sadra Heidari Moghadam: 96441119

Mobina Kashanian: 96522321

Date

2/3/2021

ABSTRACT

In this project report we analyze the implementation of Brightness control and meteorology display for the kids room. We also see how this IOT project works and what are the inputs, outputs and hardware connections. We also give information about code parts.

INTRODUCTION

The goal of this project is to control the brightness of a room (specially for the kids room) and check the meteorology of the room containing humidity and temperature. This project is an IOT (Internet Of Things) project that has software and hardware part.

SOFTWARE

Software part is coded with C language and Arduino app. It needs knowledge of Arduino Uno and ESP8266 hardware programming. For the user interface HTML, CSS and Javascript is used.

HARDWARE

Hardware part contains some hardware elements that has been connected to each other and with an order that is specialized with software program or hardware datasheet. We used Arduino Uno to control the ESP8266 module and other elements. The Arduino Uno module is connected to the laptop to get the Arduino program installed in it. We can see all of the hardware parts of this IOT project, down below:

1) Arduino Uno
2) ESP8266
3) DHT11 sensor
4) Modem
5) Breadboard
6) LED
7) Relay
8) Jumper wire

APPLICATION

To use this IOT project, user must turn the hardware on (using Arduino application) and put the hardware into the kid's room. Then user types 192.168.43.181 in the URL place of a web browser (using any device) and in this local website user can control the brightness of kid's room and check the meteorology of the room. In parallel, relay turns the lights off or on in given hours of day. So if the parents have forgotten to control brightness manually, relay controls it automatically. User must not forget to set its network ssid and password in the Arduino application in the below part.

```
const char* ssid = "GalaxyA313";
const char* password = "1234567891";
```

PROJECT DESCRIPTION

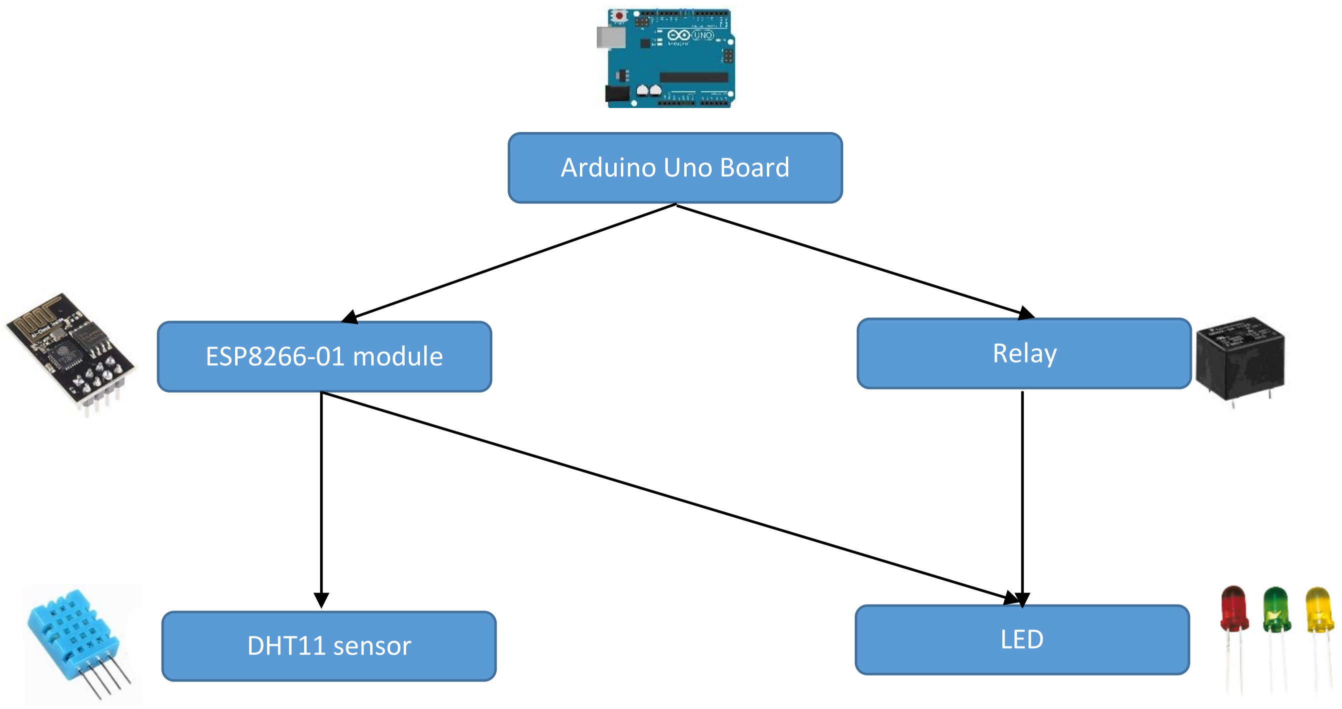
In this project we use ESP8266-01 module to connect to the internet network. It has web implementation for the ease of use and connect whenever user wants.

Brightness control has two implementations as we said above:

1) Manually: This helps the parents to manually control the brightness and check meteorology.
2) Automatically: This helps in parallel with relay to control lights on each hour

ESP8266-01 has two GPIO (General-purpose input/output) ports and we use both ports. The first port for DHT11 module and the second one for a LED.

BLOCK DIAGRAM



INPUT & OUTPUTS OF BLOCK DIAGRAM

1) **Arduino Uno Board:** This module uses Arduino C code generated in Arduino app and commands relay and ESP8266-01 module to work as the code is written.

2) **ESP8266-01 module:** This module is used to connect to the Wi-Fi and gets the command of Arduino and commands LED and DHT11 module using two GPIO ports

3) **DHT11 module:** This module gets command from ESP8266-01 module and returns the humidity and temperature of environment to ESP.

4) **LED:** This element simulates the light of kids room and gets command from ES8266-01

5) **Relay:** This module is for the automatic part of the project and gets command directly from Arduino Uno module and commands every hour to the LED to turn on or turn off

CODE EXPLANATION

In the code, at the beginning we include ESP8266 and DHT libraries to use the functions of them.

```
#include <ESP8266WiFi.h>
#include "DHT.h"
```

Then we define DHT pin as pin 0 and create an object of type DHT, set LED pin to 2, generate Wi-Fi server and set network ssid and password.

```
#define DHTPIN 0
#define DHTTYPE DHT11

DHT dht(DHTPIN,DHTTYPE,15);

const char* ssid = "GalaxyA313";
const char* password = "1234567891";


WiFiServer server(80);
int led = 2;
int brightness = 0;
```

After that in **<u>Setup()</u>** method we begin serial communication for printing logs, begin DHT module, set pin modes and initialize pins.

```
Serial.begin(115200);
delay(100);

dht.begin();
delay(100);

pinMode(led,OUTPUT);
digitalWrite(led,0);
```

Then in **Setup()** method we connect to the Wi-Fi with the given ssid and password and print some information including localIP.

```
Serial.println();
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);

WiFi.begin(ssid,password);

while (WiFi.status() != WL_CONNECTED)
{
  delay(250);
  Serial.print(".");
}
Serial.println("");
Serial.println("WiFi Connected.");

server.begin();
Serial.println("Server Started");

Serial.println(WiFi.localIP());
```

In **Loop()** function if a client has connected to the server we get humidity and temperature using DHT module and save them in two variables.

```
WiFiClient client = server.available();
if (!client)
{
  return;
}

Serial.println("new client");

while (!client.available())
{
  delay(1);
}

float h = dht.readHumidity();
float t = dht.readTemperature();
Serial.println("humadity");
Serial.println(h);
Serial.println("temperature");
Serial.println(t);
```

Then we control brightness using request functions, that are called when we press a button in web page and then we write analog value from 0 to 255 on LED port. After that we use flush function that waits until all outgoing characters in buffer have been sent.

```
String req = client.readStringUntil('\r');
Serial.println(req);
client.flush();
//-----------------------------------------
if (req.indexOf("/on") != -1)
{
  brightness = 255;
}
else if (req.indexOf("/off") != -1)
{
  brightness = 0;
}

if (req.indexOf("/increase") != -1)
{
  brightness += 20;
}
else if (req.indexOf("/decrease") != -1)
{
  brightness -= 20;
}

if(brightness > 255)
{
  brightness = 255;
}
else if(brightness < 0)
{
  brightness = 0;
}
delay(25);
analogWrite(led,brightness);
client.flush();
//-----------------------------------------
```

At last we write HTML, CSS and JavaScript code in another way and we use client.print(s) to print the HTML, CSS and JavaScript code that we are written in different way in Arduino app (the html page is available in ZIP file).

```
String s = "HTTP/1.1 200 OK\r\nContent-Type: text/html\r\n\r\n";
s += "<head>";
s += "<meta charset=\"utf-8\">";
s += "<meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">";
s += "<script src=\"https://code.jquery.com/jquery-2.1.3.min.js\"></script>";
s += "<link rel=\"stylesheet\" href=\"https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css/bootstrap.min.css\">";
s += "</head>";

s += "<body>";
s += "<div>";
s += "<div style=\"text-align:center;width:100%;padding:20px;background-color:#3EECAC\">";
s += "<div class=\"row\" style=\"text-align:center\">";
s += "<h1 class=\"col-md-12\">Embeded Systems Project</h1>";
s += "<div class=\"col-md-3\"></div>";
s += "<h3 class=\"col-md-3\">Sadra Heidari Moghadam</h3>";
s += "<h3 class=\"col-md-3\">Mobina Kashanian</h3>";
s += "<div class=\"col-md-3\"></div>";
s += "</div>";
s += "</div>";
s += "<div style=\"text-align:center\">";

s += "<h3 style=\"margin-top:40px\">ON/OFF</h3>";
s += "<div class=\"row\">";
s += "<div class=\"col-md-3\"></div>";
s += "<div class=\"col-md-3\">";
s += "<input class=\"btn btn-block btn-lg btn-success\" style=\"margin:auto;margin-top:30px;font-size:28px;border-radius:100%
s += "</div>";
s += "<div class=\"col-md-3\">";
s += "<input class=\"btn btn-block btn-lg btn-danger\" style=\"margin:auto;margin-top:30px;font-size:28px;border-radius:100%;
s += "</div>";
s += "<div class=\"col-md-3\"></div>";
s += "</div>";
s += "<div style=\"margin-top:40px; height:3px; width:100%; background-color:#3EECAC\"></div>";
s += "<h3 style=\"margin-top:40px\">Brightness Control</h3>";
s += "<div class=\"row\">";
s += "<div class=\"col-md-3\"></div>";
s += "<div class=\"col-md-3\">";
s += "<input class=\"btn btn-block btn-lg btn-success\" style=\"margin:auto;margin-top:30px;font-size:28px;border-radius:100%
s += "</div>";
s += "<div class=\"col-md-3\">";
s += "<input class=\"btn btn-block btn-lg btn-danger\" style=\"margin:auto;margin-top:30px;font-size:28px;border-radius:100%;
s += "</div>";
s += "<div class=\"col-md-3\"></div>";
s += "</div>";
s += "<div style=\"margin-top:40px; height:3px; width:100%; background-color:#3EECAC\"></div>";
s += "<h3 style=\"margin-top:40px\"> Wireless Meteorology</h3>";
s += "<div class=\"row voffset\" style=\"margin-top:40px\">";
s += "<h4 class=\"col-md-3\">Temperature: </h4><h4 class=\"col-md-3\">" + String(t) + "</h4>";
s += "<h4 class=\"col-md-3\">Humidity: </h4><h4 class=\"col-md-3\">" + String(h) + "</h4>";
s += "</div>";
s += "</div>";
s += "</div>";
s += "<script>function on() {$.get(\"/on\");}</script>";
s += "<script>function off() {$.get(\"/off\");}</script>";
s += "<script>function increase() {$.get(\"/increase\");}</script>";
s += "<script>function decrease() {$.get(\"/decrease\");}</script>";
s += "</body>";
client.print(s);
```

We have a different code to control the relay and for the automatic part that uses TimeLib library and sets relay to port 8 and sets relay behavior for 24 hours and some variables for status of relay, hour, and minute and we set hour and minute.

```
#include <TimeLib.h>
#include <SD.h>

int relay = 8;
int relay_behaviour[24] = {0 ,0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1};
double current_hour = 0;
double current_minute = 0;
int relay_status = 0;
int seconds=0, minutes=59, hours=3;
```

We have a function to set relay status based on given status for hours.

```
void set_relay_status()
{
  relay_status = relay_behaviour[(int)hours];
  Serial.println("*******");
  Serial.println(relay_status);
  if(relay_status == 1)
    digitalWrite(relay,LOW);
  else
    digitalWrite(relay,HIGH);
  delay(100);
}
```

In **Setup()** method we begin serial communication and set pin mode of relay to OUTPUT and set relay status at the beginning.

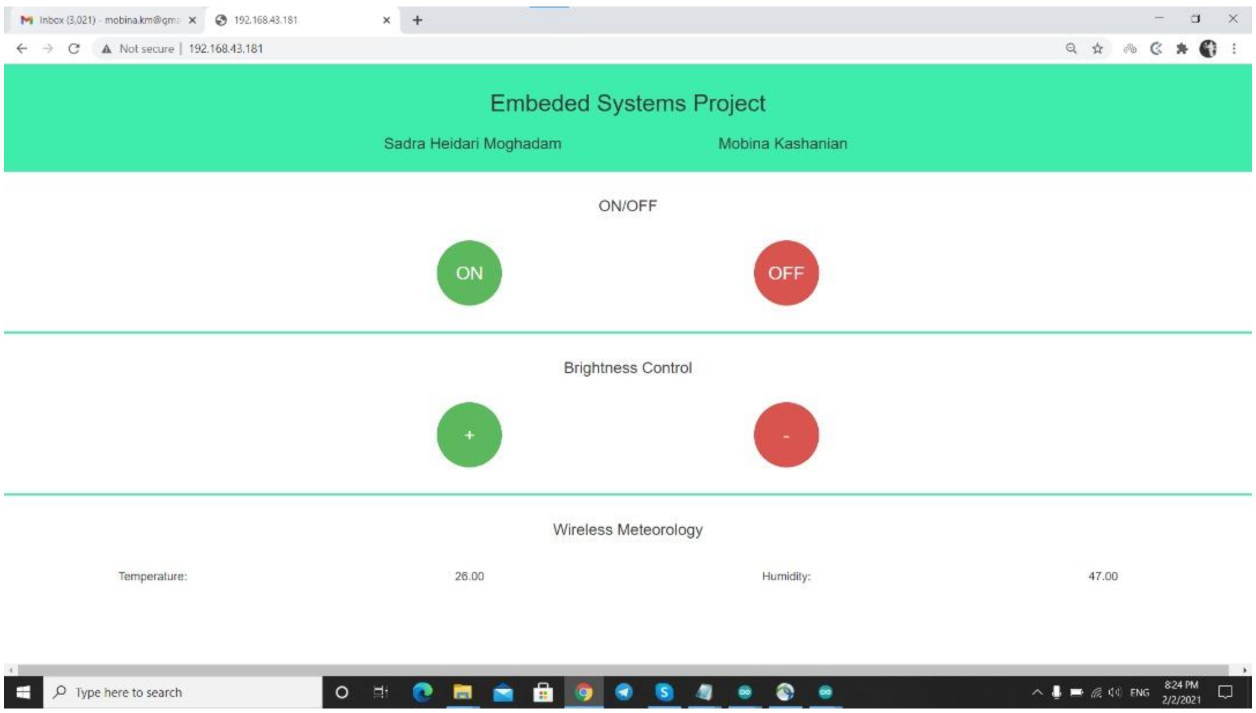
```
void setup()
{
  Serial.begin(115200);
  delay(100);
  pinMode(relay, OUTPUT);

  set_relay_status();
}
```

In **Loop()** method we set relay status each time and set hour every 60 minutes.

```
void loop()
{
  time_t t=now();
  seconds=second(t);
  Serial.println(minute(t));
  Serial.println(hour(t));
  minutes +=minute(t);
  if(minutes == 60)
  {
    hours ++;
  }
  set_relay_status();
}
```

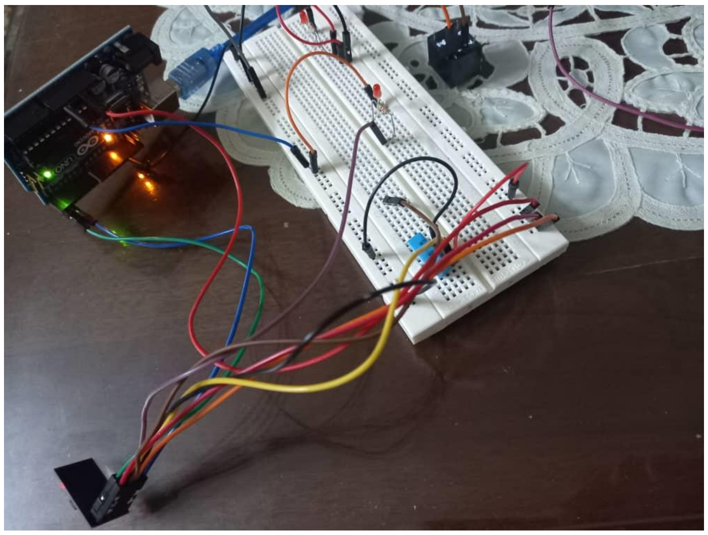
At last we can see the picture of website down below:



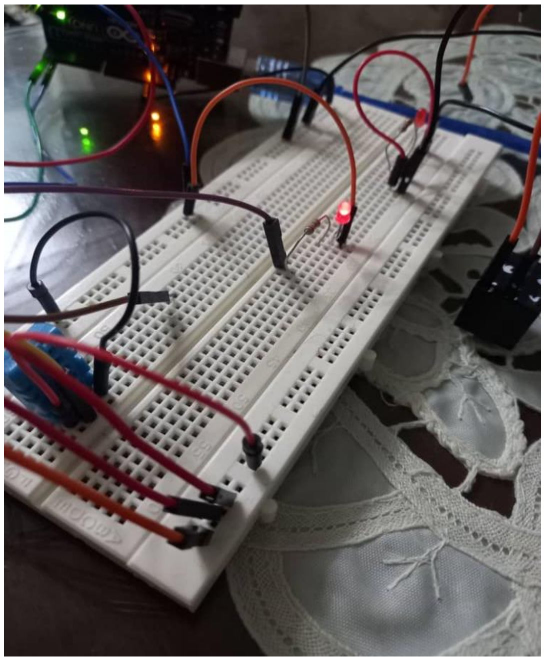
HARDWARE EXPLANATION

You can see pictures of different states of hardware.

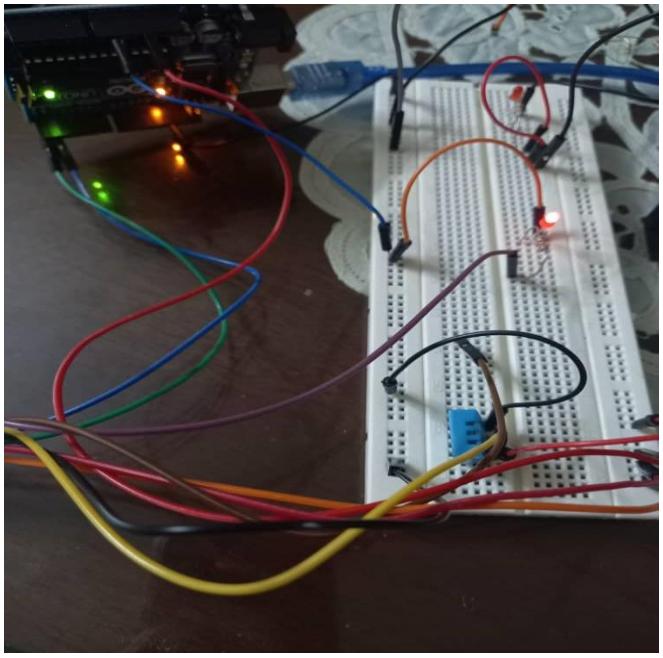
1) Brightness control and meteorology display hardware when it is off:



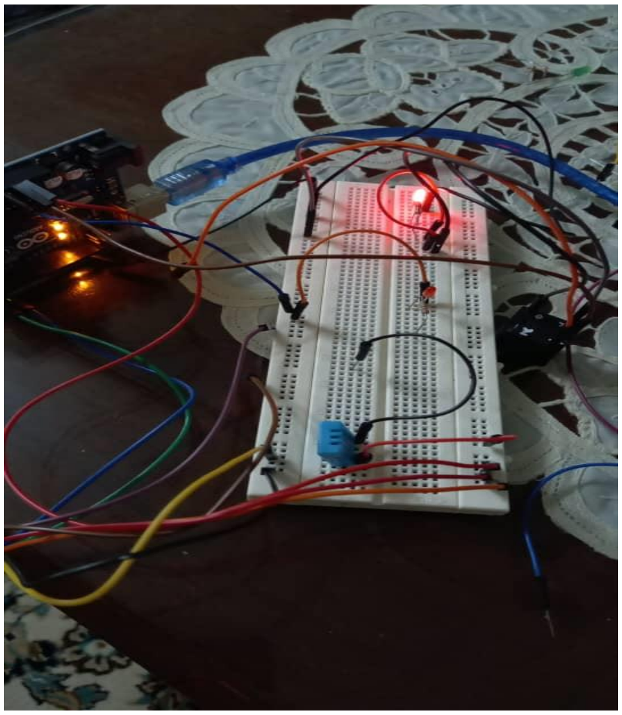
2) Brightness control and meteorology display hardware when it is half-bright:



3) Brightness control and meteorology display hardware when it is full-bright:



4) Switching On the light using relay



ADVANTAGES

The use of this IOT project is easy for the manual and automatic part. User can use it wherever he wants and if he forgets to turn the lights off manually, the relay turns it off. An another advantage is that the hardware power consumption is very low and the price of it is low too.

DISADVANTAGES

For very high voltages the hardware does not work and maybe it high voltage causes a high damage to the hardware.

CHALLENGES

We had so many challenges for this project. One of them was the connection of ESP and Relay modules and the low number of ESP GPIO's. Another one was connecting ESP to internet and constructing a website and connect Arduino code to the website functions. There were so many little challenges too that took a huge amount of time from our team.