Noapteș Maria

Email: *maria.noaptes29@gmail.com*

Environment monitoring system

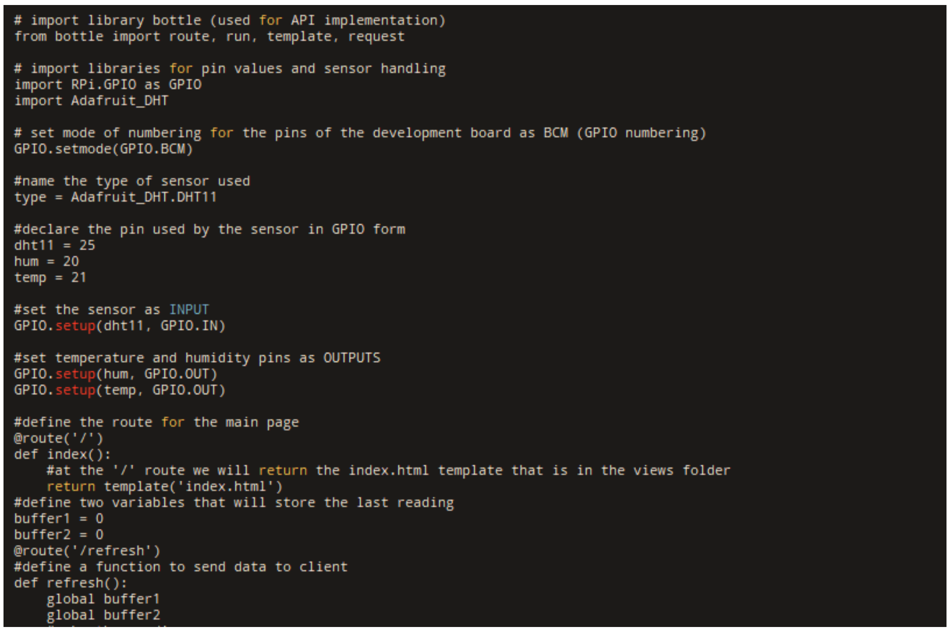
This project is meant to be a prototype for a system which monitors the temperature and the humidity in the house.
The system makes the environment conditions available to the client through an interface that can be accessed from the browser and through 2 leds that are lighted up when a certain threshold is exceeded.
The red led is lighted up if the temperature goes beyond 25 degrees Celsius and the blue one whenever humidity reaches a value greater than 60%.
The client interface that provides the exact measures of these environment conditions is available in the local network by accessing the https://localhost:5000/ link in a browser(if you are in the Raspberry Pi OS; if not, replace localhost with the IP assigned to your Raspberry Pi in the local network).

For assembling this project I used:
- Development board (Raspberry Pi 3 model A+)
- DHT11 Humidity and Temperature sensor
- Breadboard HQ 830 points
- 6 x male-to-female jumper wires
- 3 x male-to-male jumper wire
- 2 x 150 Ω resistors
- 1 x 5100 Ω resistor
- 2 x LED

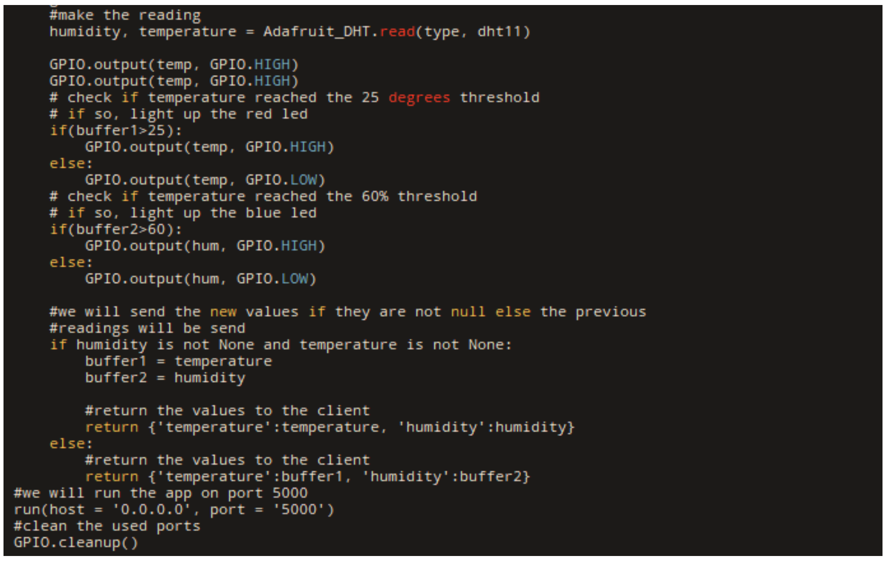
Installing dependencies:
```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
cd Adafruit_Python_DHT
sudo apt-get install build-essential python-dev
sudo python setup.py install
```

The following two portions of code are made with the help of the Plusivo kit guide (lessons 1, 14 and 22).

Server side code:

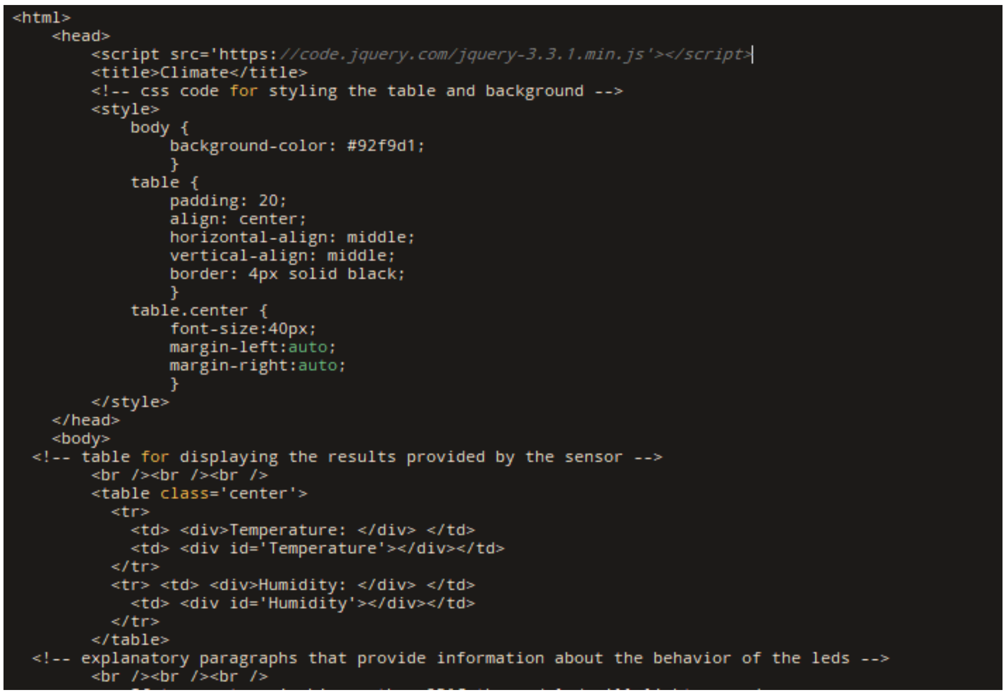


```
# import library bottle (used for API implementation)
from bottle import route, run, template, request

# import libraries for pin values and sensor handling
import RPi.GPIO as GPIO
import Adafruit_DHT

# set mode of numbering for the pins of the development board as BCM (GPIO numbering)
GPIO.setmode(GPIO.BCM)

#name the type of sensor used
type = Adafruit_DHT.DHT11

#declare the pin used by the sensor in GPIO form
dht11 = 25
hum = 20
temp = 21

#set the sensor as INPUT
GPIO.setup(dht11, GPIO.IN)

#set temperature and humidity pins as OUTPUTS
GPIO.setup(hum, GPIO.OUT)
GPIO.setup(temp, GPIO.OUT)

#define the route for the main page
@route('/')
def index():
    #at the '/' route we will return the index.html template that is in the views folder
    return template('index.html')
#define two variables that will store the last reading
buffer1 = 0
buffer2 = 0
@route('/refresh')
#define a function to send data to client
def refresh():
    global buffer1
    global buffer2
```

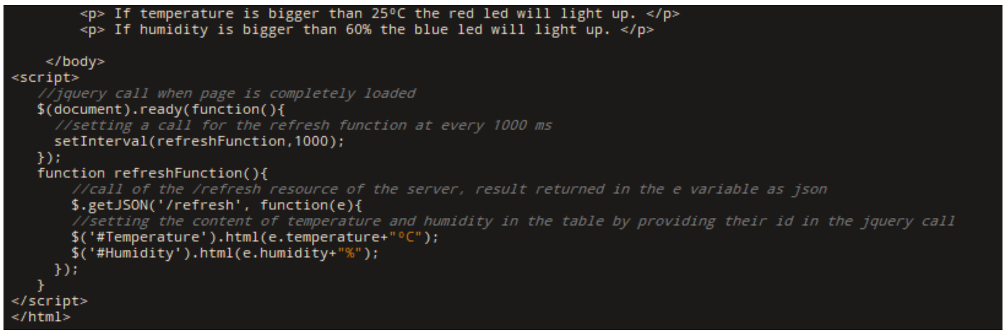


```
    #make the reading
    humidity, temperature = Adafruit_DHT.read(type, dht11)

    GPIO.output(temp, GPIO.HIGH)
    GPIO.output(temp, GPIO.HIGH)
    # check if temperature reached the 25 degrees threshold
    # if so, light up the red led
    if(buffer1>25):
        GPIO.output(temp, GPIO.HIGH)
    else:
        GPIO.output(temp, GPIO.LOW)
    # check if temperature reached the 60% threshold
    # if so, light up the blue led
    if(buffer2>60):
        GPIO.output(hum, GPIO.HIGH)
    else:
        GPIO.output(hum, GPIO.LOW)

    #we will send the new values if they are not null else the previous
    #readings will be send
    if humidity is not None and temperature is not None:
        buffer1 = temperature
        buffer2 = humidity

        #return the values to the client
        return {'temperature':temperature, 'humidity':humidity}
    else:
        #return the values to the client
        return {'temperature':buffer1, 'humidity':buffer2}
#we will run the app on port 5000
run(host = '0.0.0.0', port = '5000')
#clean the used ports
GPIO.cleanup()
```

Client side code:

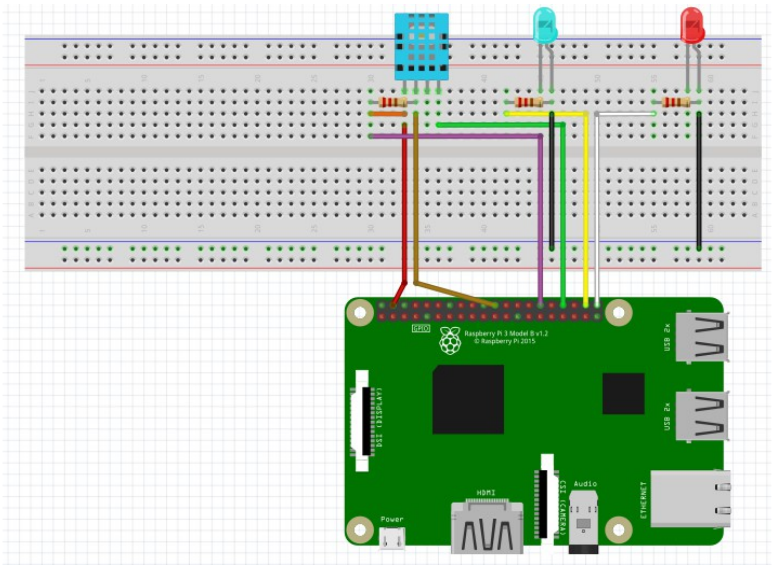


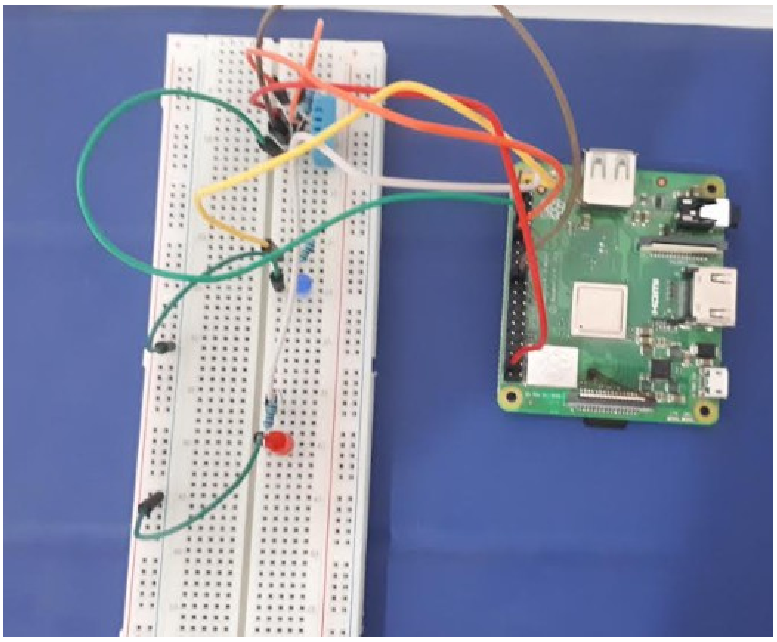
```
<html>
    <head>
        <script src='https://code.jquery.com/jquery-3.3.1.min.js'></script>
        <title>Climate</title>
        <!-- css code for styling the table and background -->
        <style>
            body {
                background-color: #92f9d1;
                }
            table {
                padding: 20;
                align: center;
                horizontal-align: middle;
                vertical-align: middle;
                border: 4px solid black;
                }
            table.center {
                font-size:40px;
                margin-left:auto;
                margin-right:auto;
                }
        </style>
    </head>
    <body>
  <!-- table for displaying the results provided by the sensor -->
        <br /><br /><br />
        <table class='center'>
          <tr>
            <td> <div>Temperature: </div> </td>
            <td> <div id='Temperature'></div></td>
          </tr>
          <tr> <td> <div>Humidity: </div> </td>
            <td> <div id='Humidity'></div></td>
          </tr>
        </table>
  <!-- explanatory paragraphs that provide information about the behavior of the leds -->
        <br /><br /><br />
```



```
        <p> If temperature is bigger than 25ºC the red led will light up. </p>
        <p> If humidity is bigger than 60% the blue led will light up. </p>

    </body>
<script>
   //jquery call when page is completely loaded
   $(document).ready(function(){
     //setting a call for the refresh function at every 1000 ms
     setInterval(refreshFunction,1000);
   });
   function refreshFunction(){
       //call of the /refresh resource of the server, result returned in the e variable as json
       $.getJSON('/refresh', function(e){
       //setting the content of temperature and humidity in the table by providing their id in the jquery call
       $('#Temperature').html(e.temperature+"ºC");
       $('#Humidity').html(e.humidity+"%");
     });
   }
</script>
</html>
```

Connections:





Link to demo [here]()
Hackster:
https://www.hackster.io/marianoaptes29/environment-monitoring-system-2b904e