Inspiratron.org - Natural language processing, machine learning and cybersecurity

Making my first Internet of Things device (for measuring temperature, humidity, and motion)

by Nikola Miloševi? - Saturday, February 27, 2021

https://inspiratron.org/blog/2021/02/27/making-my-first-internet-of-things-device-for-measuring-temperature-humidity-and-motion/

About two years ago, I got acquainted with a lady who started talking at some point about a wish to make some Arduino device. How things usually go with me, I got interested. So I decided, back in time, to buy something called Freenove Arduino Ultimate Starter Kit. It contains pretty much all you need to get started, such as Arduino, a bunch of sensors ranging from motion, temperature, humidity, infrared, a bunch of buttons, LED diodes, joystick, display, etc. It comes with a lot of tutorials both for programming the board and how to connect things. I did some interesting things with it for learning. However, if we are talking about the Internet of things device, there was one crucial component missing and that was access to the internet. So in order to try to solve that thing, I bought as well something called Heltech Wifikit 8. Now given it costs about £10, or about 12 euros, I thought it is a display with a wifi connector. However, a couple of days ago, when I started playing with it, I realized it is quite a powerful controller board with display and wifi connectivity and you can program it using the same programming environment as Arduino.

So with that realization, I started a couple of days designing my board. Firstly on prototyping breadboard, and then soldering it into the device.

Connecting to the internet

The first step was to making ESP8266 (Heltech Wifikit 8) display things and connect to the internet. Here is some code that I managed to write showing how to write things on display and connect to the internet.

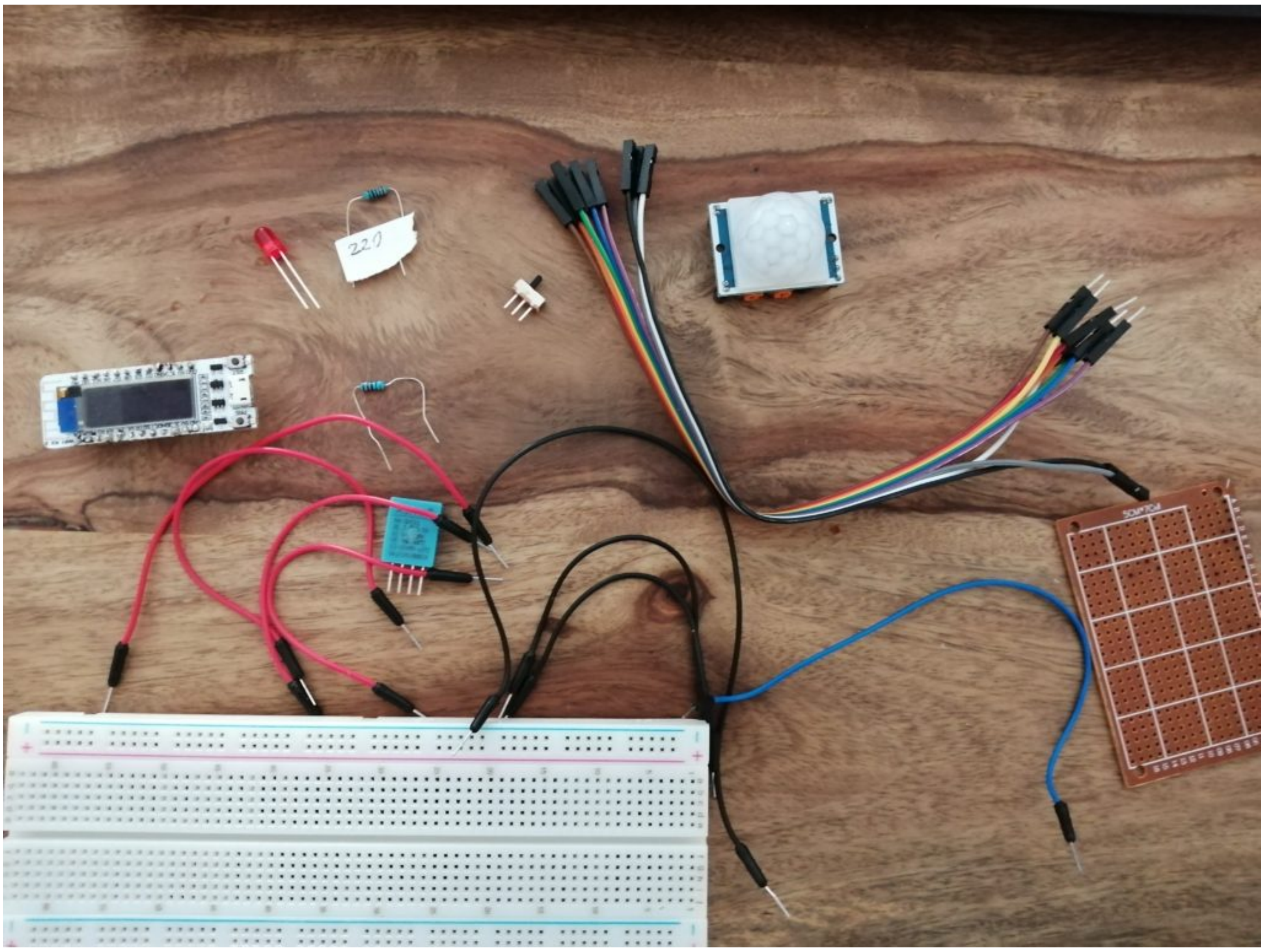
```
include <ESP8266WiFi.h>
include <ESP8266WiFiMulti.h>
ESP8266WiFiMulti wifiMulti;
void setup() {
   // put your setup code here, to run once:
   Heltec.begin(true /DisplayEnable Enable/, true /Serial Enable/);
   Heltec.display->init();
   Heltec.display->flipScreenVertically();
   Heltec.display->setFont(ArialMT_Plain_10);
 Heltec.display->drawString(0, 0, "Hello Nikola!");
 Heltec.display->display();
   delay(5000);
 WiFi.mode(WIFI_STA);
   wifiMulti.addAP("YOUR_NETWORK_ID", "YOUR_NETWORK_PASSWORD");
   Heltec.display->clear();
   Heltec.display->drawString(0, 0, "Connecting Wifi…");
   Heltec.display->display();
   delay(1000);
 if (wifiMulti.run() == WL_CONNECTED) {
     Heltec.display->clear();
     Heltec.display->drawString(0, 0, "WiFi connected");
     Heltec.display->drawString(0, 10, "IP address: ");
     String broadCast = IpAddress2String(WiFi.localIP());
     Heltec.display->drawString(0, 20, broadCast);
     Heltec.display->display();
 }
 }
void loop() {
   if (wifiMulti.run() != WL_CONNECTED) {
     Heltec.display->clear();
     Heltec.display->drawString(10, 0, "WiFi not connected!");
     delay(1000);
   } else {
     Heltec.display->clear();
     Heltec.display->drawString(0,0,"WiFi connected");
}
}
```

Writing on screen is pretty straightforward using Heltec.display->drawString(0, 0, "Hello Nikola!"); once everything is initialized. First, two numbers are positions of the first character (so the top left in this case). After drawing you can call Heltec.display->display() to show it or clear() to clear screen. If you write multiple strings at the same location they would just be drawn one over the other.

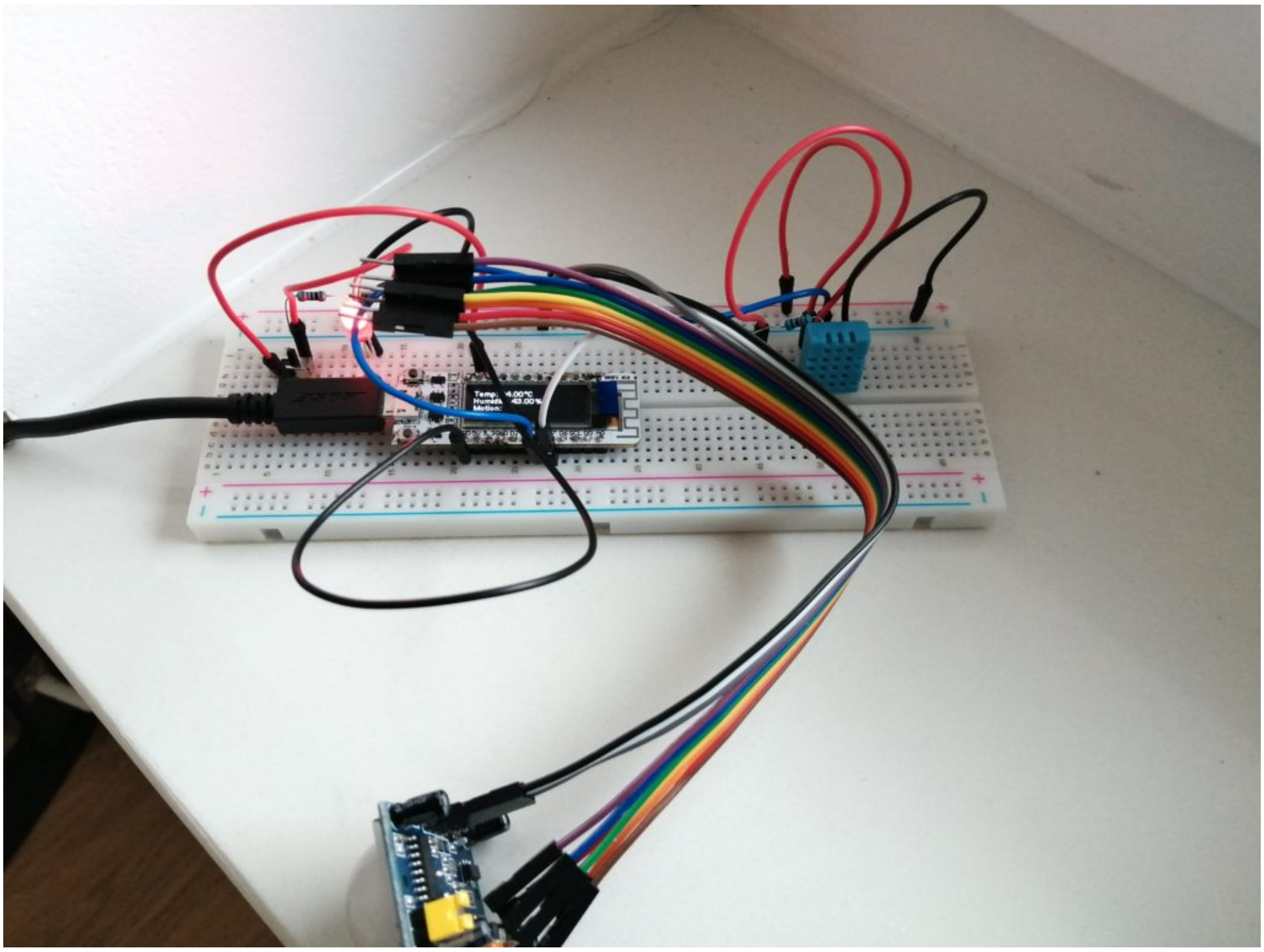
Now as everything works I could move forward designing a circuit.

Designing a circuit

Previously, I have already made Arduino powered circuit measuring temperature and humidity using a D11 sensor that came with Arduino Starter Kit. Now I wanted to try doing a similar thing, but with the board having internet connectivity, so it sends data online and I could check them online by accessing a website or app. In addition to D11, I also decided it would be cool to have a motion sensor, so I can put the device somewhere close to my doors and it can act both as a security device and as a measurement for temperature and humidity. I used a breadboard that came with Arduino kit to design the board. However, it didn't all go smoothly. I faced some strange issue that if all is connected, and I connect the device via USB to electricity, the screen does not work. However, if I disconnect the pin outputting power (5V) from the ESP8266 to other components, the screen loads, and then I can put the pin back and everything works smoothly. However, this is not something you can have on the device, so I managed to solve the issue with a two-way switch. The switch would stop power to go out of the board until the screen loads, and then you can switch it, and you can get sensor readings.

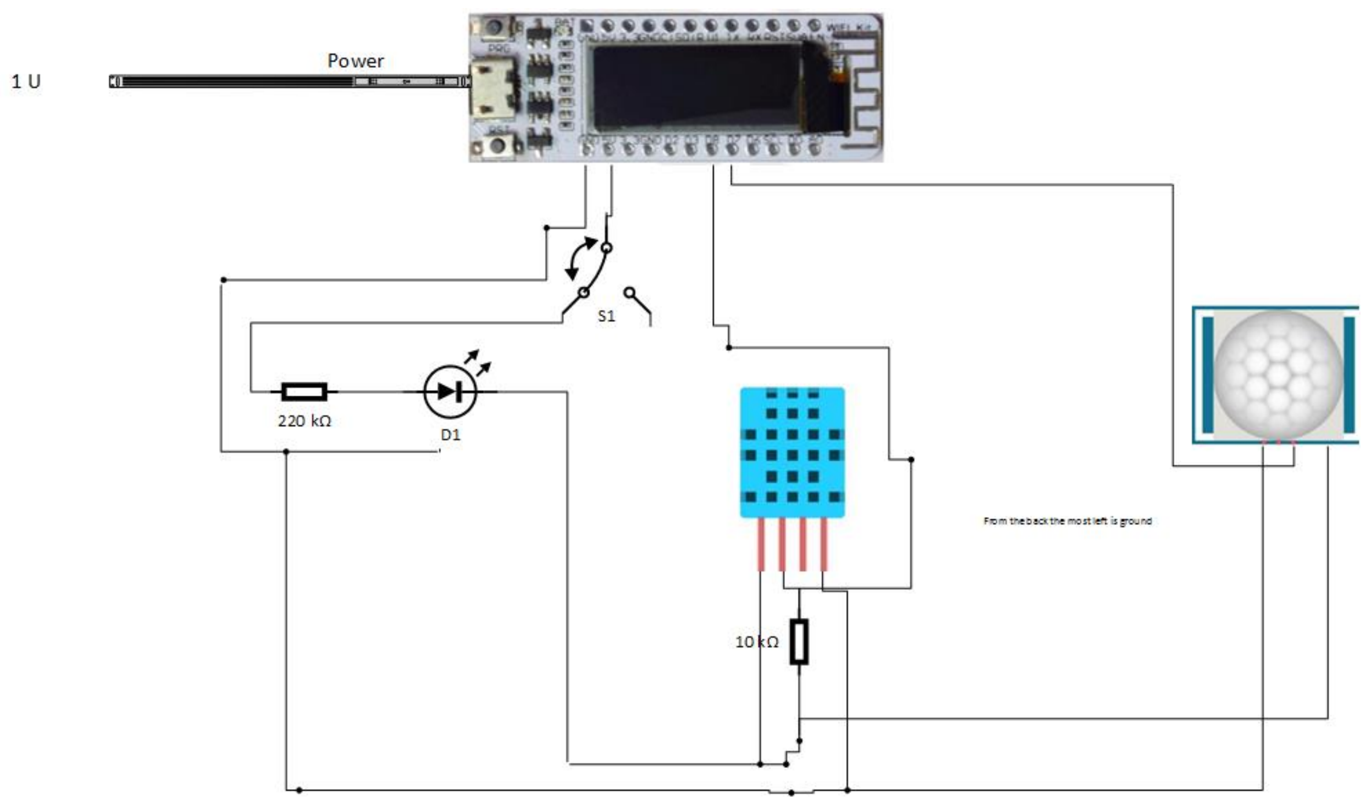


Components needed for the project



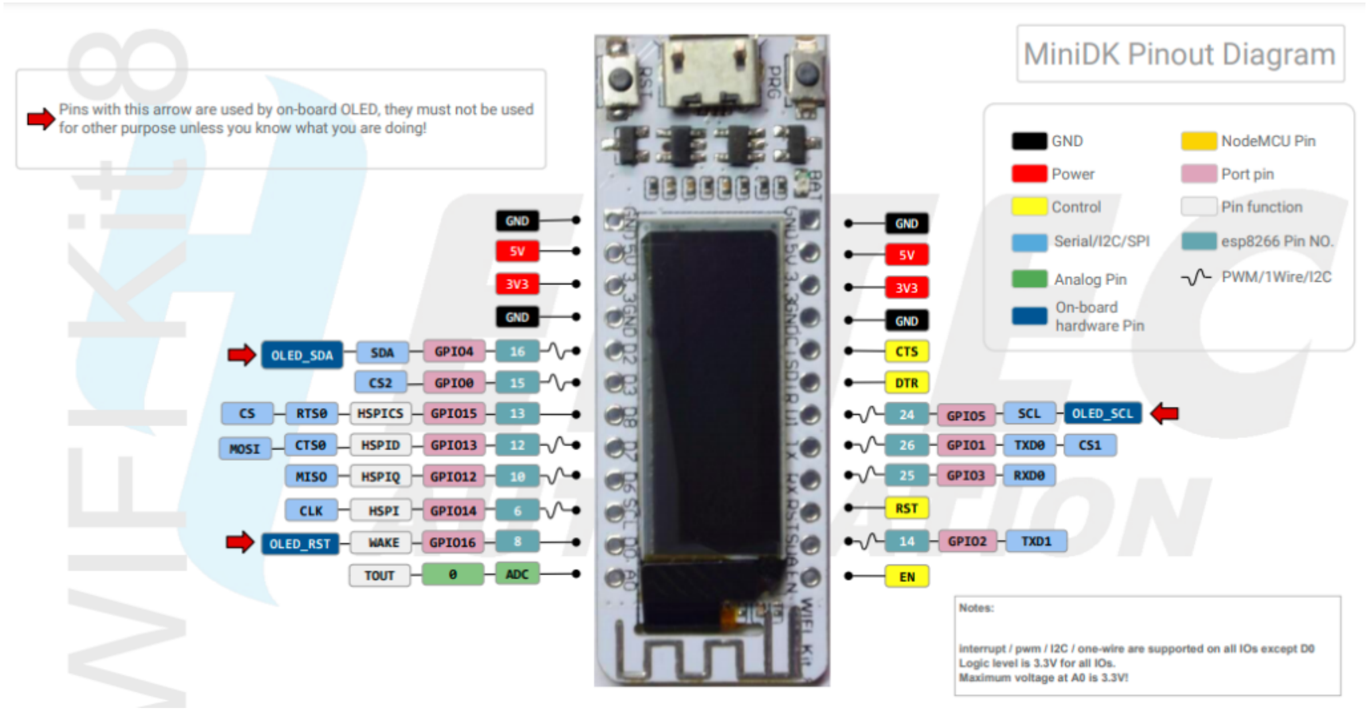
Final prototype on the breadboard

I understand, it is not easy to see from this picture how the connections go, so I tried to draw a schematic diagram of the device.



Schema of the connections for the designed device

When it came down to programming and reading inputs, I had a little issue with understanding pin numbering. So I connected D11 (temperature and humidity sensor) to input D8 on ESP8266 and motion sensor D7. Initially, I thought, in code, I would read from pin ids 7 and 8, however that is not the case. With ESP8266 comes the following diagram.



Pins with this arrow are used by on-board OLED, they must not be used for other purpose unless you know what you are doing!

Notes:
interrupt / pwm / I2C / one-wire are supported on all IOs except D0
Logic level is 3.3V for all IOs.
Maximum voltage at A0 is 3.3V!

ESP 8266 WifiKit 8 pinout diagram

The actual ids of the pins are in this pink column starting with GPI, so pin 8 is GPI015, meaning its id is 15, while pin 7 has GPI013, meaning the id is 13.

So the final code for the sketch that was uploaded to the device looks the following way and can be seen on my GitHub.

Making a web service for the device

Now the device is done, for it to be a complete IoT device, it needs a web service to send data to and to present data. I have made a couple of Google searches to find whether there is any free Python hosting that would allow me also some sort of database. To my quite big surprise, I found exactly what I was looking for. The site is called PythonAnywhere. They do exactly what I was looking for. They provide Flask and MySQL hosting for free. There are some limitations, like the size of files and database. However, for my needs what they provide was perfect.

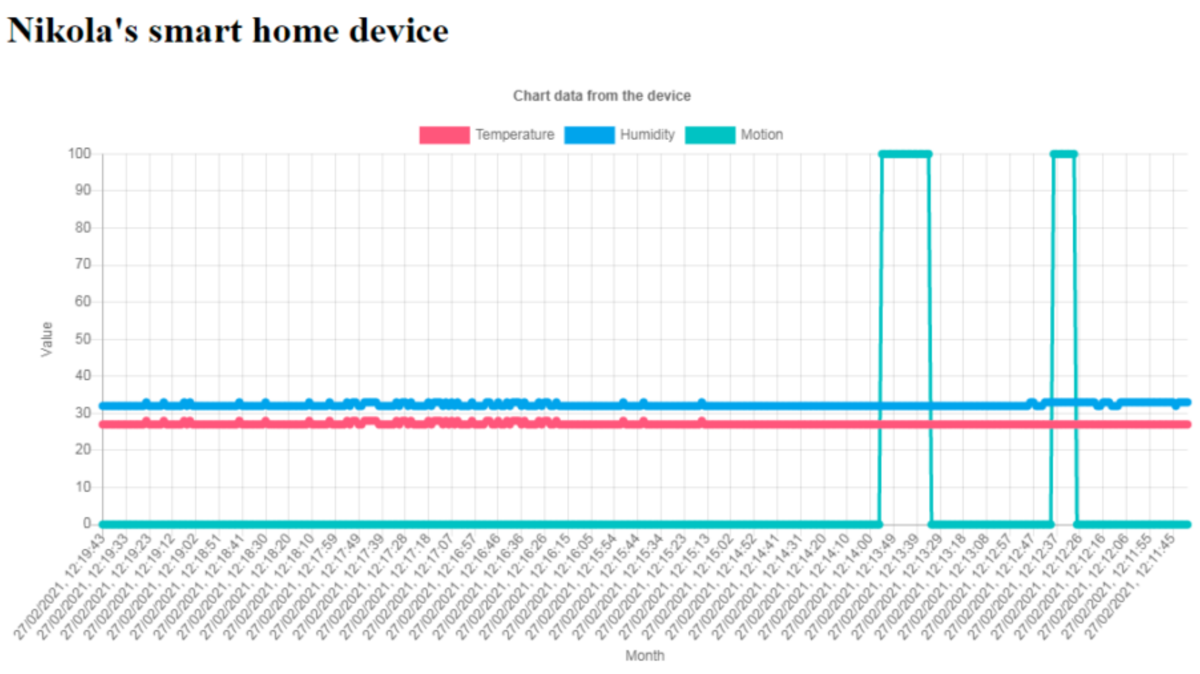
So I wrote a simple Flask service with two main endpoints. The first would accept the data from the device and store in the database. This endpoint will also make sure, I don't store too much data, so it would delete data older than 2 days. The second endpoint would provide me with the user interface for the visualization. For this, I used a template that utilized Chart.js.

The code in Flask looked the following way:

```
from flask import Flask
import MySQLdb as mdb
from flask import render_template
from flask import request

app = Flask(__name__)


@app.route('/')
def analytics():
    username = 'XXXX'
    password = 'XXXX'
    hostname = 'XXXX.mysql.pythonanywhere-services.com'
    db = 'XXXX$smarthome'
    conn = mdb.connect(host=hostname, user=username, passwd=password, db=db)
    cursor = conn.cursor()
    # Select data for the last 24 hours = 24*60*60 = 86400
    sql = 'select * from entries  order by id desc limit 86400'
    cursor.execute(sql)
    rows = cursor.fetchall()
    times = []
    temperatures = []
    humidity = []
    movement = []
    for row in rows:
        times.append(row[1].strftime("%d/%m/%Y, %H:%M:%S"))
        temperatures.append(float(row[2][:-2]))
        humidity.append(float(row[3][:-1]))
        if row[4]=='':
            movement.append(0)
        else:
            movement.append(int(row[4])*100)
    conn.close()
    return render_template('chart_page.html',datetimes=times, temperature=temperatures,humidit
ies = humidity,motion = movement)

@app.route('/add_data')
def add_data():
    username = 'XXXX'
    password = 'XXXX'
    hostname = 'XXXX.mysql.pythonanywhere-services.com'
    db = 'XXXX$smarthome'
    conn = mdb.connect(host=hostname, user=username, passwd=password, db=db)
    temperature = request.args.get('temp')
    humidity = request.args.get('humid')
    motion = request.args.get('motion')
    ip = request.args.get('ip')
    cursor = conn.cursor()
    sql = "INSERT INTO entries (temperature, humidity,movement,ip) VALUES (%s, %s,%s,%s)"
    val = (temperature, humidity,motion,ip)
    cursor.execute(sql, val)
    conn.commit()
    sql = "DELETE FROM entries WHERE id NOT IN (SELECT id FROM (SELECT id FROM entries ORDER B
Y id DESC LIMIT 180000) foo);"
    cursor.execute(sql)
    conn.commit()
    conn.close()
    return "OK"
```

The template page you can view on GitHub, so I do not overcrowd this post with the code. The method *render_template* changes variable inside {{}} with the arrays created in the root endpoint method, filling javascript that plots chart with data.

Final webpage of that endpoint looks in the following way:



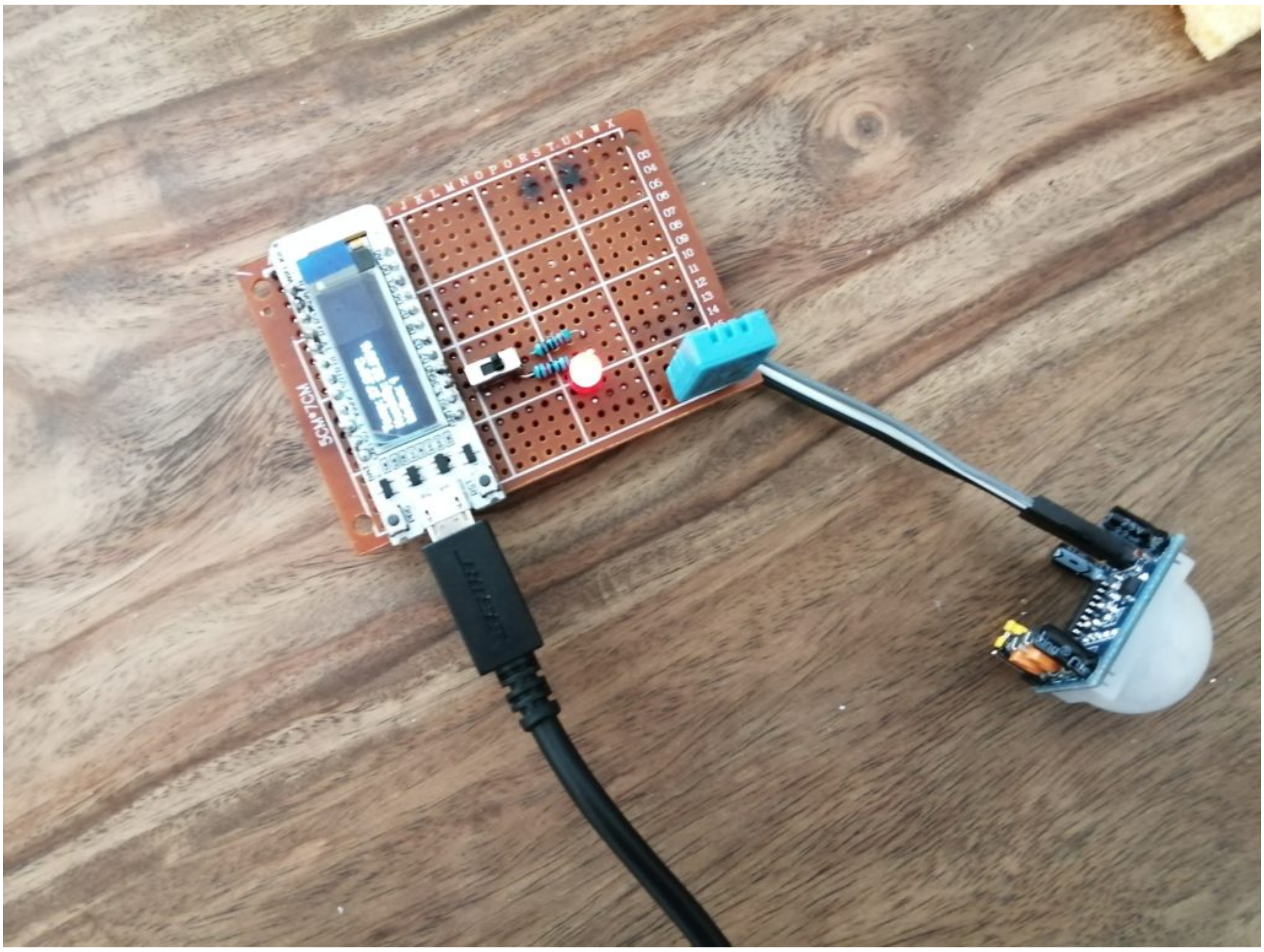
Plot from the web interface for the device

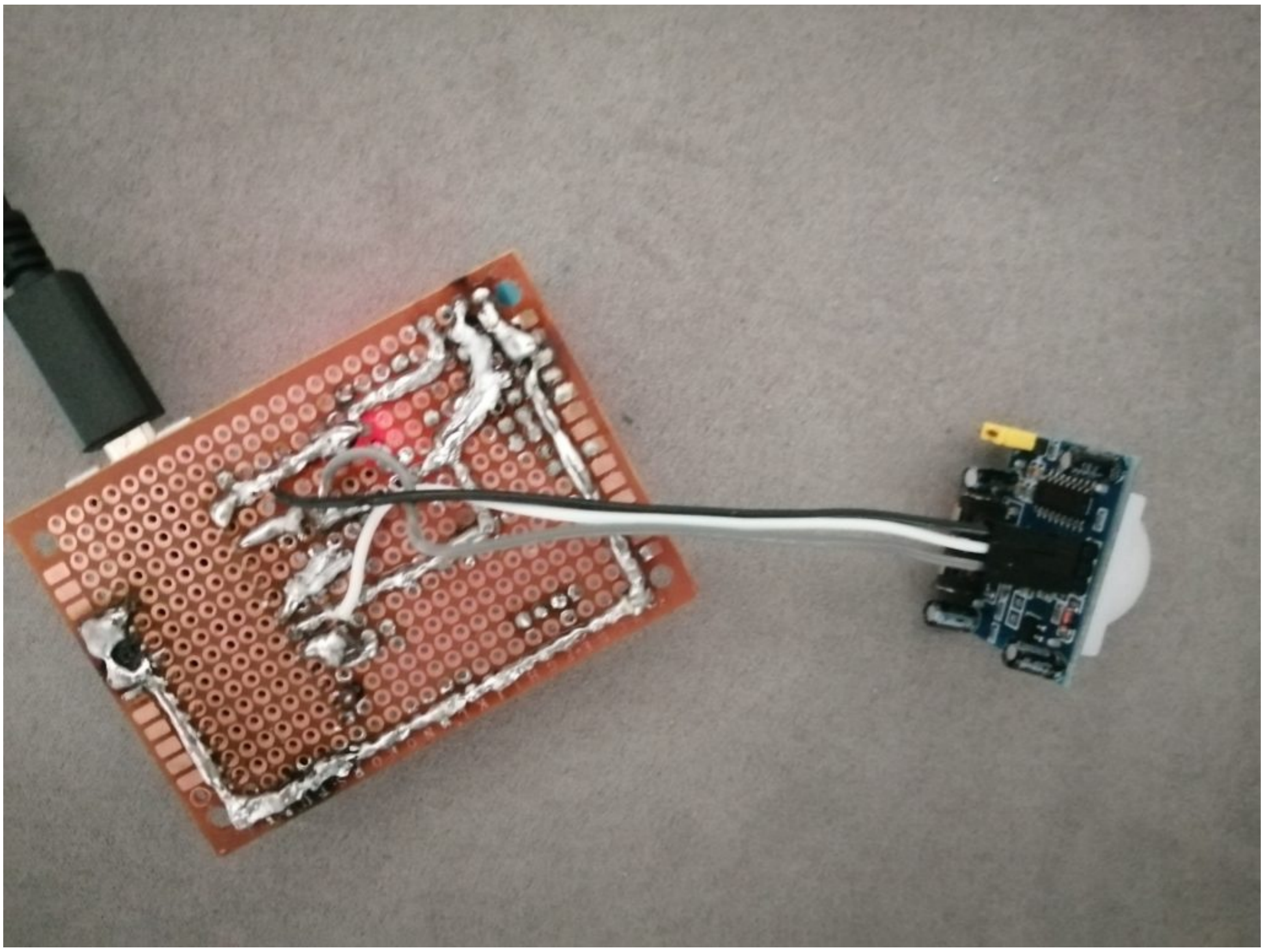
Soldering the device

Once the web service and the device work and they are connected to each other, it is time to finalize the device, so it can be permanent, and solder pieces together on a small board. Freenove kit comes with 3-4 soldering boards that are 5x7 cm, which is a pretty good size for this kind of device. I had to buy a new soldering iron, and the one I picked was Meterk Soldering Iron Kit 60 W 14 in 1. It comes with a case, iron, some tweezers, soldering wires, and basically everything one needs.

I was soldering for the first time in my life. Probably I used way more soldering wire then it was necessary, but it all went more or less okay. I had to unsolder some bits, cause I made some mistakes, or solder went where it was not supposed to. One issue once soldering was that on both sensors, the data pin is in the middle, while ground and VCC are on the sides. It was quite problematic to solder both sensors to the boards (which was initial plan) without wires crossing on the board. So I soldered external wires to motion sensor. If I ever make a case, it would make sense for it to be a bit more out and higher compared to other sensor.

Finally, it all looked like this:


Front side of the finally soldered device


Back side and soldering

And it worked, after a little bit of playing around it. I would recommend to solder sensor by sensor and test in each instance whether everything works before moving forward. At least, that made it easier for me to troubleshoot if anything went wrong. The final stage would be to make some casing for the device. This is something that I was thinking about quite a bit, as you end up with a custom device with a custom dimensions. I may just reuse some carton box and cut it for the casing. More proper, plastic casing seems a bit more dificult to make and I may be lacking some equipment for it.

Cost

At the end, when you sum up the costs, in order to make a device like this, it costs about 20 euros:

- 12 euro is the WifiKit
- ~ 3 euros are sensors each (so in total about 6-7)
- some 1-2 euros for soldering wire and soldering board
- Maybe about 1-2 euros for LED diode, switch and all other wiring used

Given that non-smart humidity/temperature meters on Amazon cost between 13-20 euros and motion detectors between 10-30 euros, without internet connectivity, it doesn't sound like a bad deal. The most similar device I found is at least 30 euros.

Future possible integrations

From the perspective of the device it is the most important that it records data and sends it to the web service. Once we have data we can do more integrations. It is possible to make a mobile app pushing notifications using the web service. Web service needs a small modification to send data back as json, and then mobile app can manipulate them. Also, it can be integrated into Google Assistant for example, using API. For this we would need to find hosting with HTTPS hosting, which is usually not free. So I gave up on this idea until I get some free HTTPS python hosting. However, it may be also possible to make a chatbot on Facebook or other platform that would send notifications in form of messages or answer to some questions about data device is recording. This I may implement in the future and if I do, I would surely blog about it.