

**SLOVENSKÁ TECHNICKÁ UNIVERZITA**  
**FAKULTA ELEKTROTECHNIKY A INFORMATIKY**

**Meteo-stanica**  
(Tehnická dokumentácia)  
Pokročilé informačné technológie

## Zadanie:

Cieľom zadania je monitorovať signály získané z reálnych senzorov. Monitorovanie je uskutočnené prostredníctvom web serveru, aby bola naplnená koncepcia IoT.

Ako hardvér a softvér je využitá platforma Raspberry Pi 1 B+ na ktorú je napojený pomocou káblikov teplotný senzor DHT11 a pre komunikáciu je využitý Ethernet, programovací jazyk Python a Java script pre webové rozhranie. Na vytvorenie rozhrania web-serveru snímajú hodnôt v reálnom čase je využitý program Node-Red.

## 1. Potrebné komponenty na projekt:

### 1.1. Hardvér

#### a) Raspberry Pi

Pre spustenie Raspberry Pi je potrebné napájanie zo zdroja, USB prepojenie klávesnice a myši, SD karta s nainštalovaným softvérom a HDMI pripojenie obrazovky.

#### b) Teplotný senzor DHT11 + káblíky

Teplotný senzor DHT11 je malý modul, ktorý poskytuje digitálne údaje o teplote a vlhkosti. Má jednoduché nastavenie a pre dátový signál je potrebný iba jeden vodič. Tieto senzory sú populárne pre použitie vo vzdialených meteorologických staniciach, monitorovaní pôdy a domácich automatizačných systémoch.

Hardvérová časť je zahrnutá v [dokumente architektúry systému](#). (Potrebné piny daných komponentov, schéma zapojenia, UML diagram)

### 1.2. Softvér

#### c) Raspbian

Na programovú časť projektu využijeme nainštalovaný Raspbian, v ktorom využijeme:

- Python - získanie dát zo senzora DHT11 + výpis do textového súboru Python
- Python – Javascript (klient – server) - tvorba serveru (HTTP)
- Node-RED – získanie dát zo senzora DHT11 + tvorba užívateľského rozhrania (HTTP : 1880) + vykreslenie aktuálnych hodnôt na web-server (HTTP : 1880/ui) vo forme text-graf-cíferník.

## 2. Získanie dát zo senzora DHT11

Ak máme správne zapojený senzor, pre získanie dát zo senzora je potrebné nainštalovať knižnicu Adafruit\_DHT pre daný snímač.

Po nainštalovaní potrebných knižníc vytvoríme nový textový dokument, kde budeme zapisovať hodnoty vo forme stringov zo snímača pre vytvorenie histórie na serveri.

Vytvoríme si kód v pythone na získavanie potrebných hodnôt zo snímača, ktorý je zobrazený na Obr.1:

```
import Adafruit_DHT
import time

senzor = Adafruit_DHT.DHT11          #Senzory pre vyuzitie(DHT_11/22)
#senzor = Adafruit_DHT.DHT22
senzor_pin = 4                        #DHT data pin pripojeny na GPIO4
cyklus = True                         #premenna pre while cyklus
datalist = []

while cyklus:                         #nekonecny while cyklus

    try:
        vlh, tep = Adafruit_DHT.read_retry(senzor, senzor_pin) #nacitanie hodnot zo senzora do premennych
        tep_f = tep * 9/5.0 + 32      #teplota_f prevedena na fahrenheity

        dataDict = {
            "t": time.strftime('%H:%M:%S %d/%m/%Y'),
            "x": tep,
            "y": tep_f,
            "z": vlh,
        }
        dataList.append(dataDict)

        if vlh is not None and tep is not None and len(dataList)>0:      #podmienky pre premenne k 0
            print('TEPLOTA = ' + str(tep) + ' *C,' + 'TEPLOTA Fahrenheit = ' + str(tep_f) + ' F,' + 'VLHKOST = ' + str(vlh) + ' %.')      #vypisanie hodnot
            hod = str(dataList).replace("'", "\"")
            fo = open("hodnoty.txt", "w")
            fo.write("%s\r\n" %hod)      #ukladanie premennych do subora
            time.sleep(1800)

            # if len(dataList)>0:

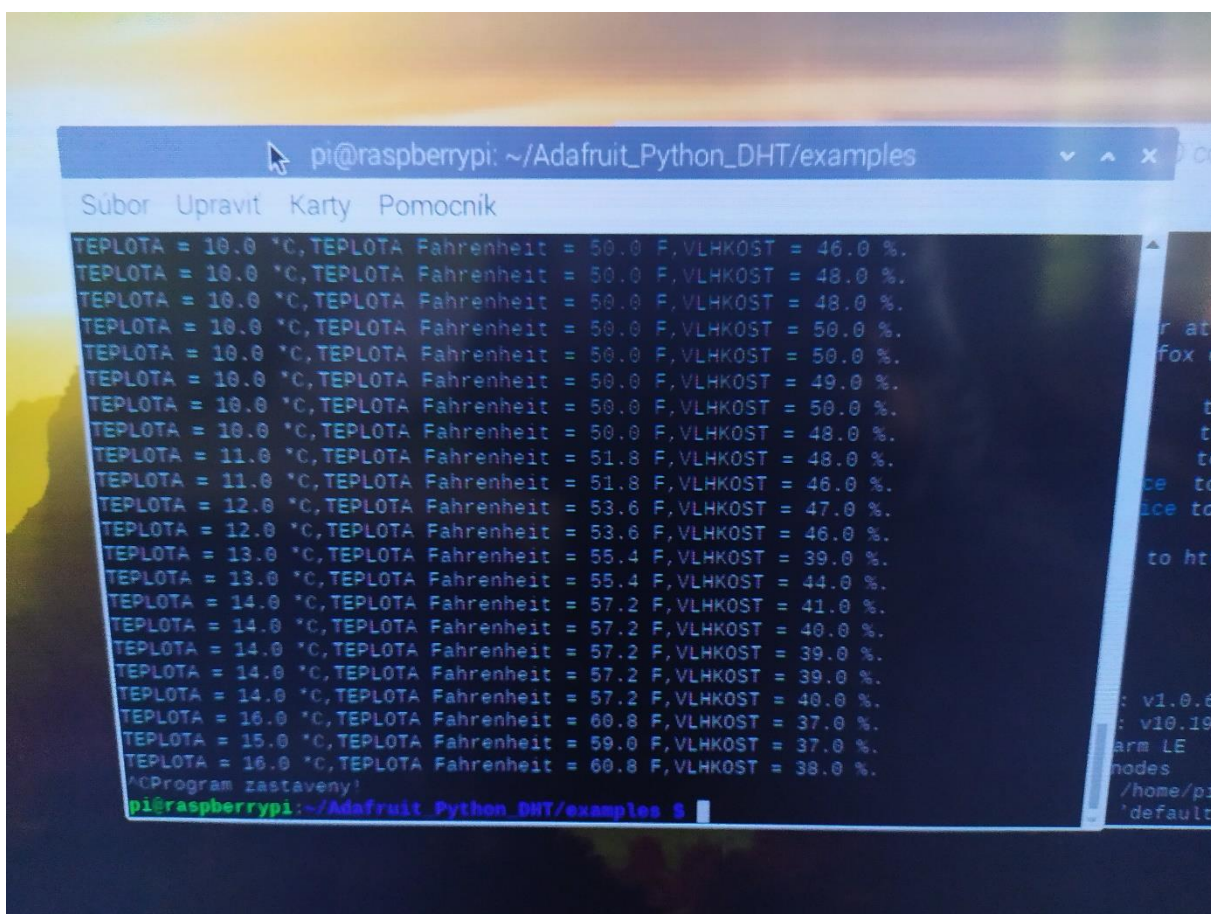
        else:
            print('CHYBA! Skus znova!')
            time.sleep(1)

    except KeyboardInterrupt:      #zastavenie cyklu klavesou CTRL+C
        print ('Program zastaveny!')      #ako ked koncime server v termialy
        cyklus = False
        fo.close()
        dataList = [];
```

Obr. 1 Kód v jazyku Python pre zaznamenávanie meraných hodnôt zo snímača

- dataDict – vytvorenie stringu zo želaných premenných do prázdneho poľa dataList
- vzorky berieme po pol hodine - time.sleep(1800)

Výpis hodnôt do terminálu, Obr.2.:



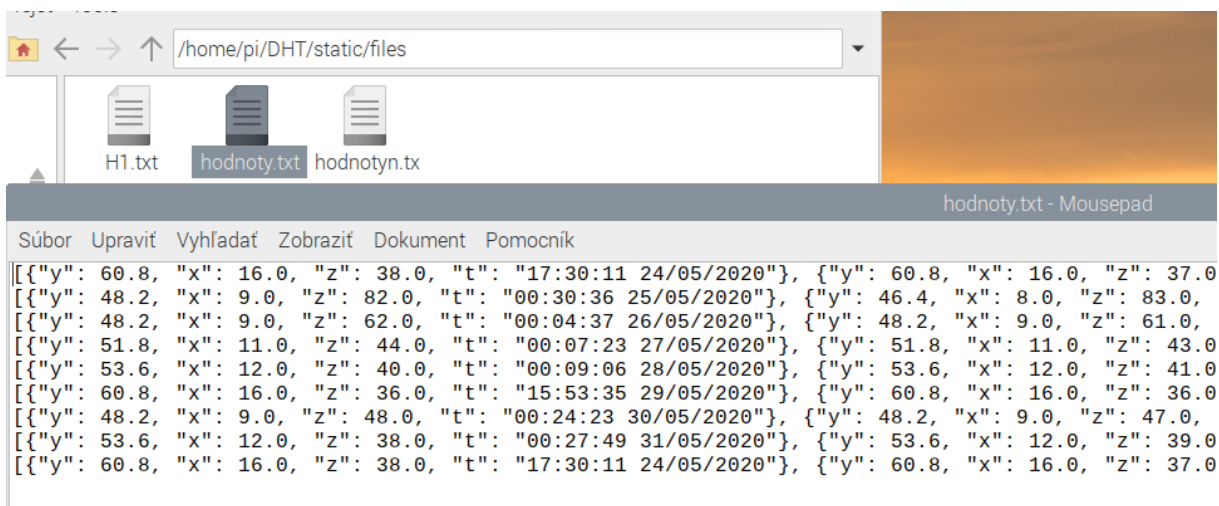
The screenshot shows a terminal window titled 'pi@raspberrypi: ~/Adafruit\_Python\_DHT/examples'. The window contains a menu bar with 'Súbor', 'Upraviť', 'Karty', and 'Pomocník'. The main area displays a list of sensor readings in a repeating pattern. Each line shows temperature in Celsius and Fahrenheit, and humidity in percentage. The readings are as follows:

TEPLOTA (°C)	Fahrenheit (°F)	VLHKOST (%)
10.0	50.0	46.0
10.0	50.0	48.0
10.0	50.0	48.0
10.0	50.0	50.0
10.0	50.0	49.0
10.0	50.0	50.0
10.0	50.0	48.0
11.0	51.8	48.0
11.0	51.8	46.0
12.0	53.6	47.0
12.0	53.6	46.0
13.0	55.4	39.0
13.0	55.4	44.0
14.0	57.2	41.0
14.0	57.2	40.0
14.0	57.2	39.0
14.0	57.2	40.0
14.0	57.2	39.0
14.0	57.2	40.0
16.0	60.8	37.0
15.0	59.0	37.0
16.0	60.8	38.0

The terminal ends with the prompt 'pi@raspberrypi:~/Adafruit\_Python\_DHT/examples \$'.

Obr. 2 Výpis zosnímaných hodnôt priamo do terminálu

Výpisom overíme, že kód funguje správne. Skontrolujeme aj textový dokument, Obr.3, a môžeme pokračovať na tvorbu serveru.



The screenshot shows a file manager window with the address bar set to '/home/pi/DHT/static/files'. It displays three files: 'H1.txt', 'hodnoty.txt', and 'hodnotyn.tx'. The 'hodnoty.txt' file is selected, and its contents are shown in a text editor titled 'hodnoty.txt - Mousepad'. The text in the editor is a JSON array of sensor data records, each containing temperature, humidity, and a timestamp. The records are as follows:

Record	y (°C)	x (%)	z (°C)	t (timestamp)
1	60.8	16.0	38.0	"17:30:11 24/05/2020"
2	48.2	9.0	82.0	"00:30:36 25/05/2020"
3	48.2	9.0	62.0	"00:04:37 26/05/2020"
4	51.8	11.0	44.0	"00:07:23 27/05/2020"
5	53.6	12.0	40.0	"00:09:06 28/05/2020"
6	60.8	16.0	36.0	"15:53:35 29/05/2020"
7	48.2	9.0	48.0	"00:24:23 30/05/2020"
8	53.6	12.0	38.0	"00:27:49 31/05/2020"
9	60.8	16.0	38.0	"17:30:11 24/05/2020"

Obr. 3 Textový dokument - zapisovanie hodnôt zo senzora

### 3. Tvorba serveru

Na tvorbu serveru, konkrétne histórie zobrazovania nameraných údajov využijeme náš pripravený textový dokument zo získaných hodnôt zo snímača. Pre stranu serveru využijeme opäť kód v Pythone a na stranu klienta vyžijeme JavaScript.

- Server

Vytvoríme ako app a spustíme server ako localhost – cez IP, kde zapíšeme kde konkrétne bude náš server prebiehať, Obr.4.

```
from threading import Lock
from flask import Flask, render_template, session, request, jsonify, url_for
import time

app = Flask(__name__)

app.config['DEBUG'] = 'True'

@app.route('/')
def hello():
    return render_template('tabs.html')

@app.route('/read/<string:num>')
def readmyfile(num):
    fo = open("static/files/hodnoty.txt", "r")
    rows = fo.readlines()
    return rows[int(num)-1]

if __name__ == '__main__':
    app.run(host="0.0.0.0", port=80, debug=True)
```

*Obr. 4 Kód v pythone pre stranu serveru*

- Klient

Stranu klienta realizujeme pomocou „templates,, v JavaScripte, kde si najskôr zadefinujeme potrebné kódovanie, skripty a záhlavie serveru, Obr.5.

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>LB-MeteoStanica</title>
    <!-- http://jqueryui.com/tabs/#mouseover -->
    <link rel="stylesheet" href="static/jquery-ui.min.css">
    <script type="text/javascript" src="//code.jquery.com/jquery-1.4.2.min.js"></script>
    <script type="text/javascript" src="//cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.5/socket.io.min.js"></script>
    <script src="static/jquery-3.2.1.min.js"></script>
    <script src="static/jquery-ui.min.js"></script>
    <script src="static/libs/jquery-1.11.1.min.js"></script>
    <script src="static/plotly-latest.min.js"></script>
    <script type="text/javascript" charset="utf-8">
```

*Obr. 5 Kód pre stranu klienta*

Vytvoríme si kód pre vykreslenie grafu – Príklad, zobrazený na Obr.6.:

```
$(document).ready(function(){
var layout = {
  title: 'TEPLOTA [°C]',
  xaxis: {
    title: 'Čas',
    range: [0,48]
  },
  yaxis: {
    title: 'T',
    range: [5,25]
  }
};

$('#form#rec').submit(function(event) {
var $link = "/read/"+$('#value').val();
console.log($link);
$.ajax({
  type: "GET",
  url: $link,
  success:function(data)
  { console.log(data);
    data = JSON.parse(data);
    console.log(data);
    n = Object.keys(data).length;
    console.log(n);

    x1 = [];
    y1 = [];

    for (var i=0; i< n; i++){
      x1.push(data[i].t);
      y1.push(data[i].x); }
    let trace = [{
      x: x1,
      y: y1 }];
    Plotly.newPlot($('#plotdiv')[0], trace,layout);
  }
}).done(function( o ) {
// do something
});
return false;});
```

Obr. 6 Kód pre vykreslenie grafov na strane klienta

Kde načítavame hodnoty z uložených súborov v stringu a následne ich vieme podľa dní vykresliť na stránke, pomocou vytvoreného formulára.

Príklad vytvorenia tela stránky, ako bude vyzerat', zobrazuje Obr.7.:

```
<body>

<div id="tabs">
  <ul>
    <li><a href="#tabs-1">TEPLOTA</a></li>
    <li><a href="#tabs-2">VLHKOSŤ</a></li>
    <li><a href="#tabs-3">TEPLOTA FEHRENHEIT</a></li>
    <li><a href="#tabs-4">CELKOVO</a></li>
  </ul>
  <div id="tabs-1">
    <h1>História meraných hodnôt od 24.05.-31.05 2020</h1>
    <h2>Voľba dní:</h2>
    <form id="rec" method="POST" action="#">
      <input type="text" name="value" id="value" placeholder="Deň No.:">
      <input type="submit" value="Set">
    </form>
    <h2>Zobrazenie:</h2>
    <div id="plotdiv" style="width:800px;height:450px; display: inline-block"></div>
  </div>
```

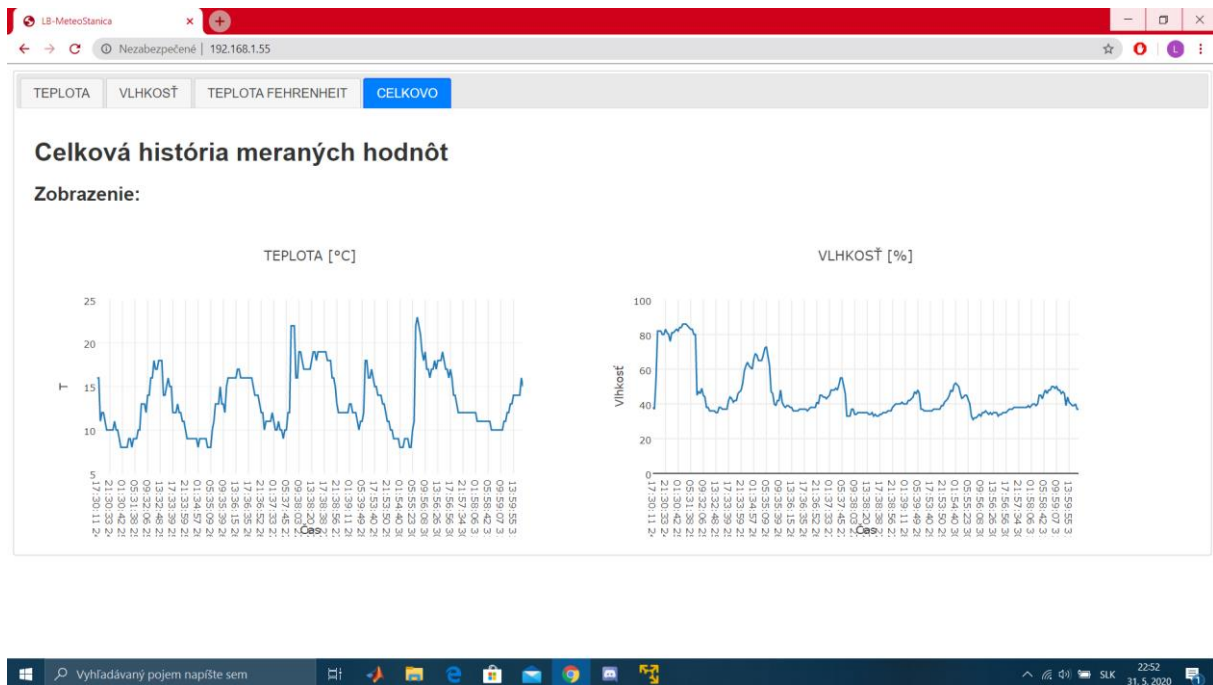
Obr. 7 Kód pre usporiadanie divízií na strane klienta

V tejto sekcii vytvárame divízie (bloky textu) ,<div>, a názvy odrážok na stránke a v nich konkrétne názov na stránke <h>. Vkladáme aj formulár,<form> pre výpis daného dňa - „stringu,, hodnôt z textového súboru. Vykreslenie grafu <div id> a následne ukončenie divízie </div>. Takto pokračujeme až po posledný graf, ktorý chceme vykresliť na stránku a tak si dokážeme vytvoriť históriu snímaných hodnôt na web-server.

- Webserver - výsledok:

Výsledkom tvorby serveru je história nameraných hodnôt zo snímača, ktoré je zobrazená na následných obrázkoch, Obr.8 a Obr.9 v podaní webovej stránky:

**Celková história údajov:**



Obr. 8 Webserver - zobrazenie celkovej histórie

**Príklad: teplota - história údajov:**

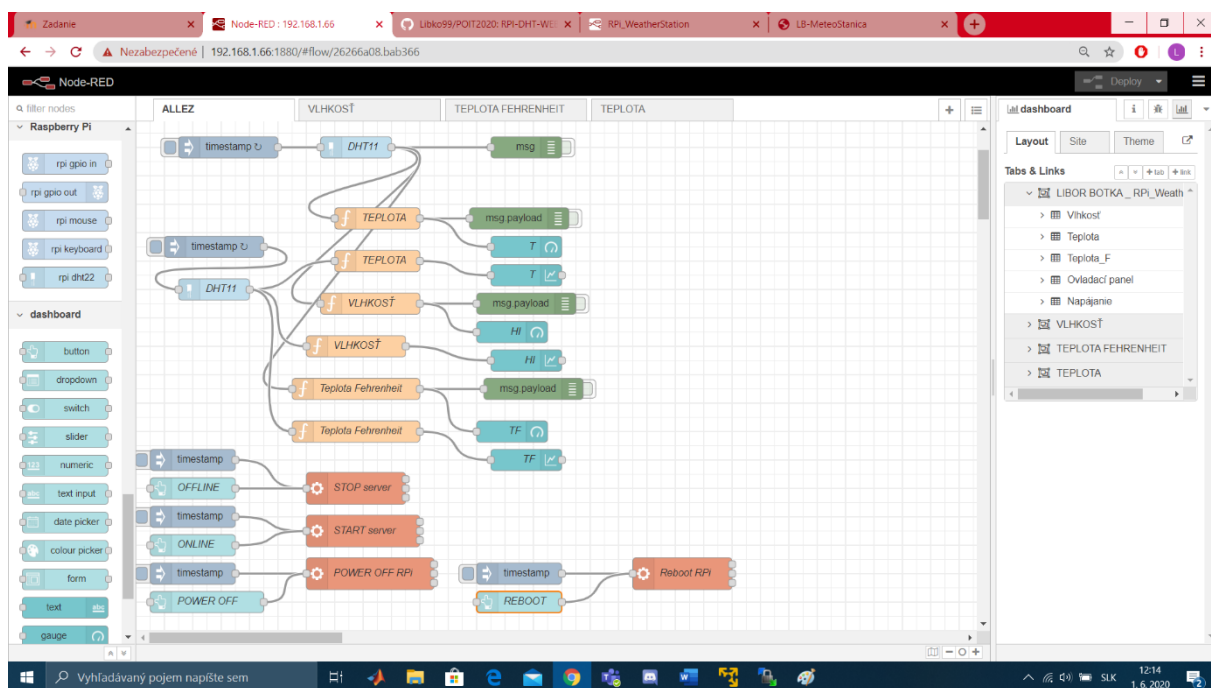


Obr. 9 Webserver - zobrazenie teploty z dňa č.2



## 4. Node-Red

Na vykreslenie aktuálnych hodnôt na web-server zvolíme Node-Red, kde dokážeme jednoducho naprogramovať časť pre klient-server pomocou prepojenia „blok+kód“, a vytvoríť prúdy vďaka ktorým spojazdníme server. Opäť je potrebné nainštalovať Node-Red a k nemu aj ďalšie knižnice či skripty. Po inštalácii otvoríme prostredie Node-Red a otvoríme v prostredí Chrome na lokálnej IP:1880, viditeľné na Obr.10:



Obr. 10 Prostredie Node-Red pre tvorbu serveru

V prostredí pracujeme z blokmi, tzv. nody, a skladáme prúdy. Pre získanie reálnych hodnôt priamo zo senzora potrebujeme stiahnuť v manage palette node pre DHT a nainštalovať do programu. Podobne inštalujeme aj dashboard, ktorý nám posluží na grafiku, v našom prípade ide o grafy a ciferníky.

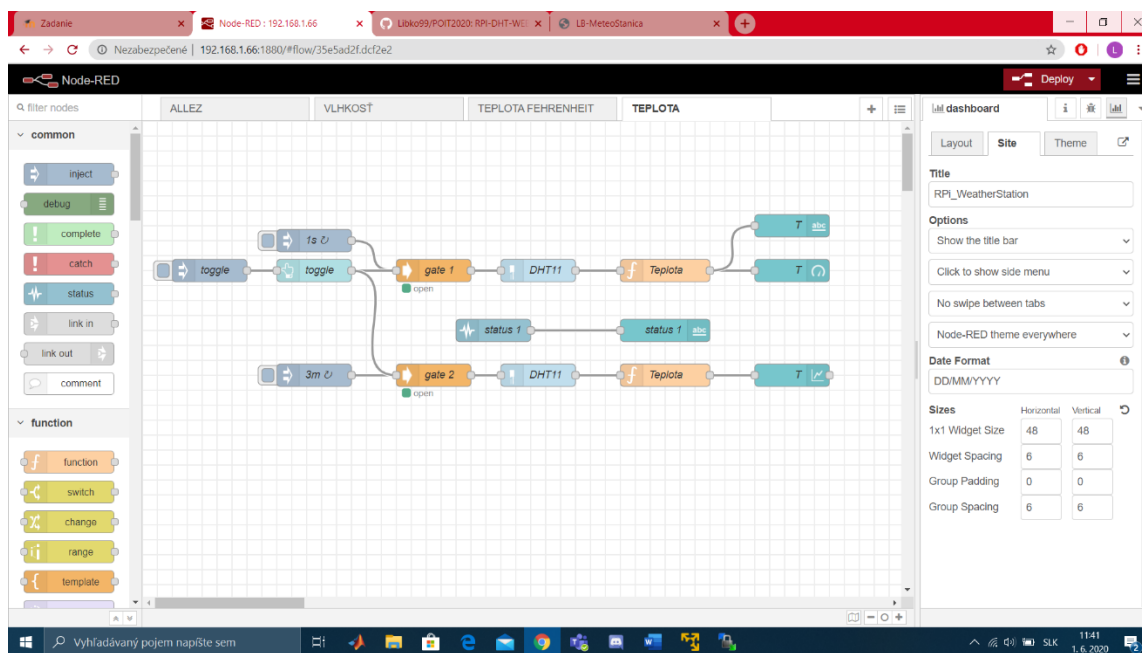
Prúdy pozostávajú zo:

- **Vstupov** – timestamp, kde sme nastavili nekonečný cyklus zo vzorkovaním každú sekundu.
- **Senzora** – nainštalovaný DHT balík, nastavený na data PIN
- **Funkcie** – každá veličina má svoju funkciu, kde sú nastavené hodnoty aké majú byť vykreslené na výstupe
- **Výstupov** – v podobe výpisu do systému (kontrola – nevidí používateľ) , gauge – ciferníkov, grafov – nastavené priamo v blokoch na požadované hodnoty.
- **Tlačidlá** – tlačidlá na vypnutie raspberry pi, pomocou sudo príkazu do terminálu



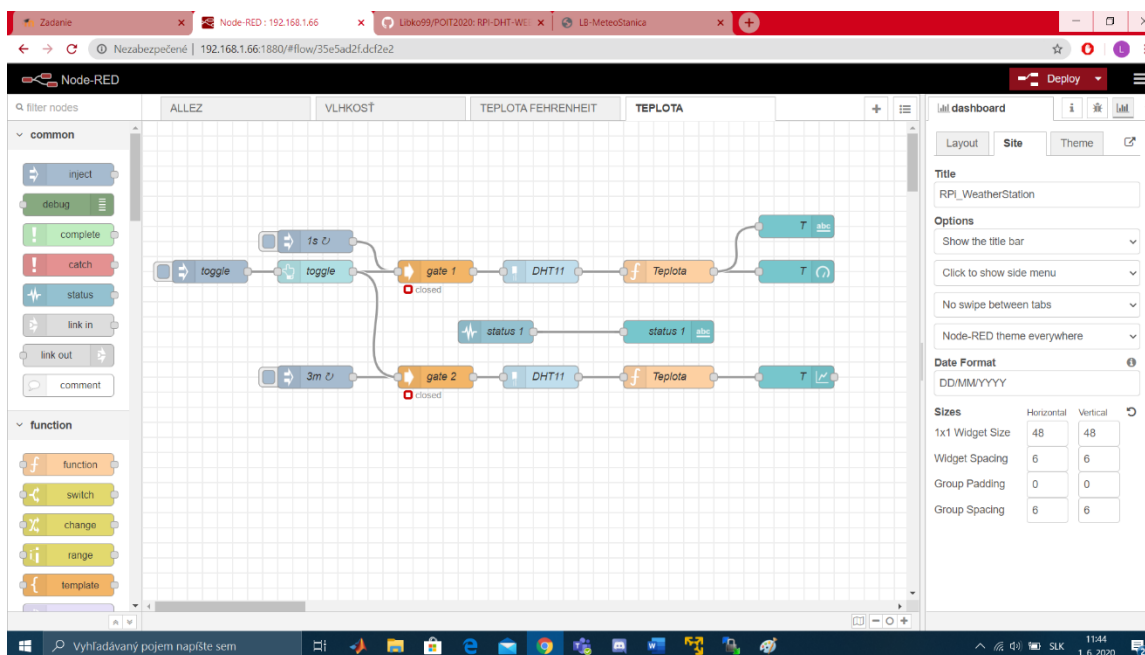
Dôležité je vytvoriť si karty, ktoré budú na stránku na výber v ponuke a vieme si prehľadne preklikať, čo potrebujeme vidieť. V nich si ešte vytvoríme skupiny do ktorých budeme zaraďovať potrebné bloky, ktoré chceme vykresliť na stránke, aby sme zabezpečili lepší vzhľad a prehľadnosť stránky. Následne si ukážeme ošetrenie START/STOP tlačidla v jednotlivých kartách, ktoré sme ošetrili pomocou bloku GATE, zobrazené na Obr.11/12:

### START – gate open:



Obr. 11 Ošetrenie tlačidla START - Gate open

### STOP – gate close:



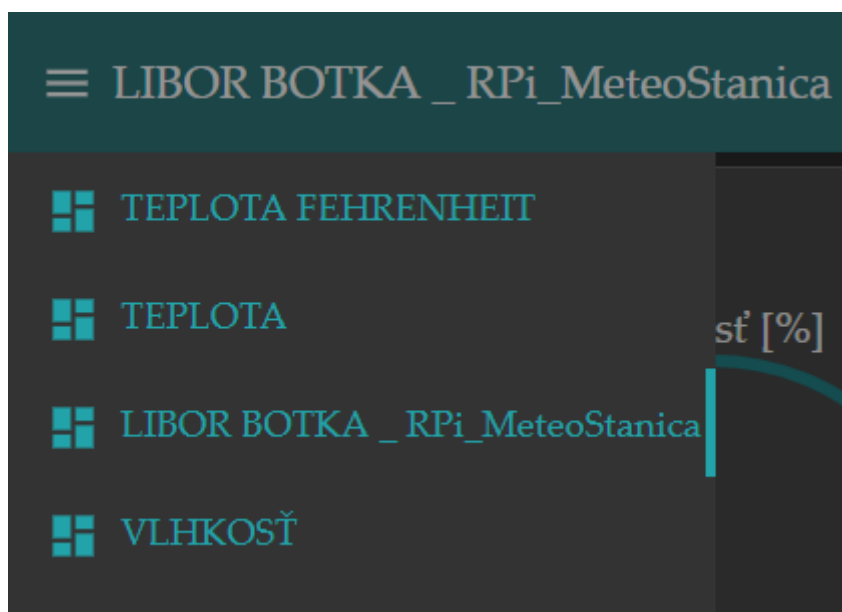
Obr. 12 Ošetrenie tlačidla STOP - Gate close

Prietok dát v prúdoch ošetríme vďaka funkcií toggle, teda prepínaniu open/close hodnôt. Vďaka bloku status – výpis, vypíšeme stav brán na web – open/close.

- Webserver - výsledok:

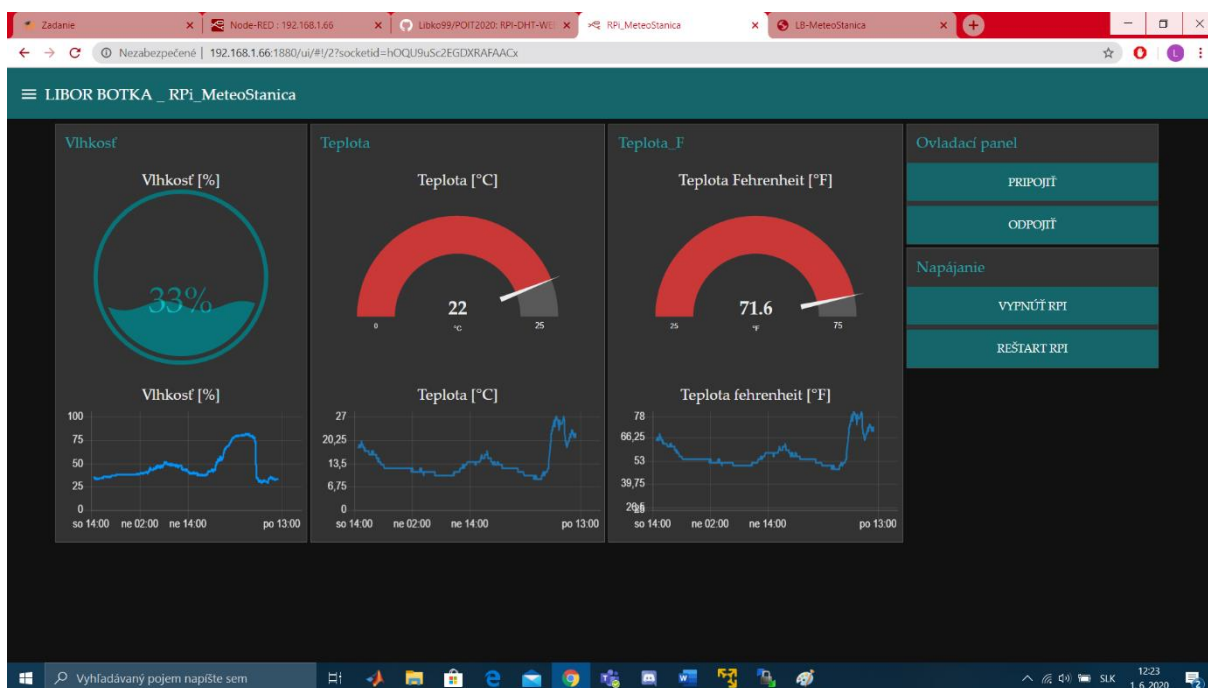
Výsledkom tvorby prúdov v programe Node-Red je aktuálne zobrazovanie hodnôt zo senzora, pričom vytvorené užívateľské prostredie je zobrazené na následných Obr. 13,14,15:

Voľba menu na stránke:



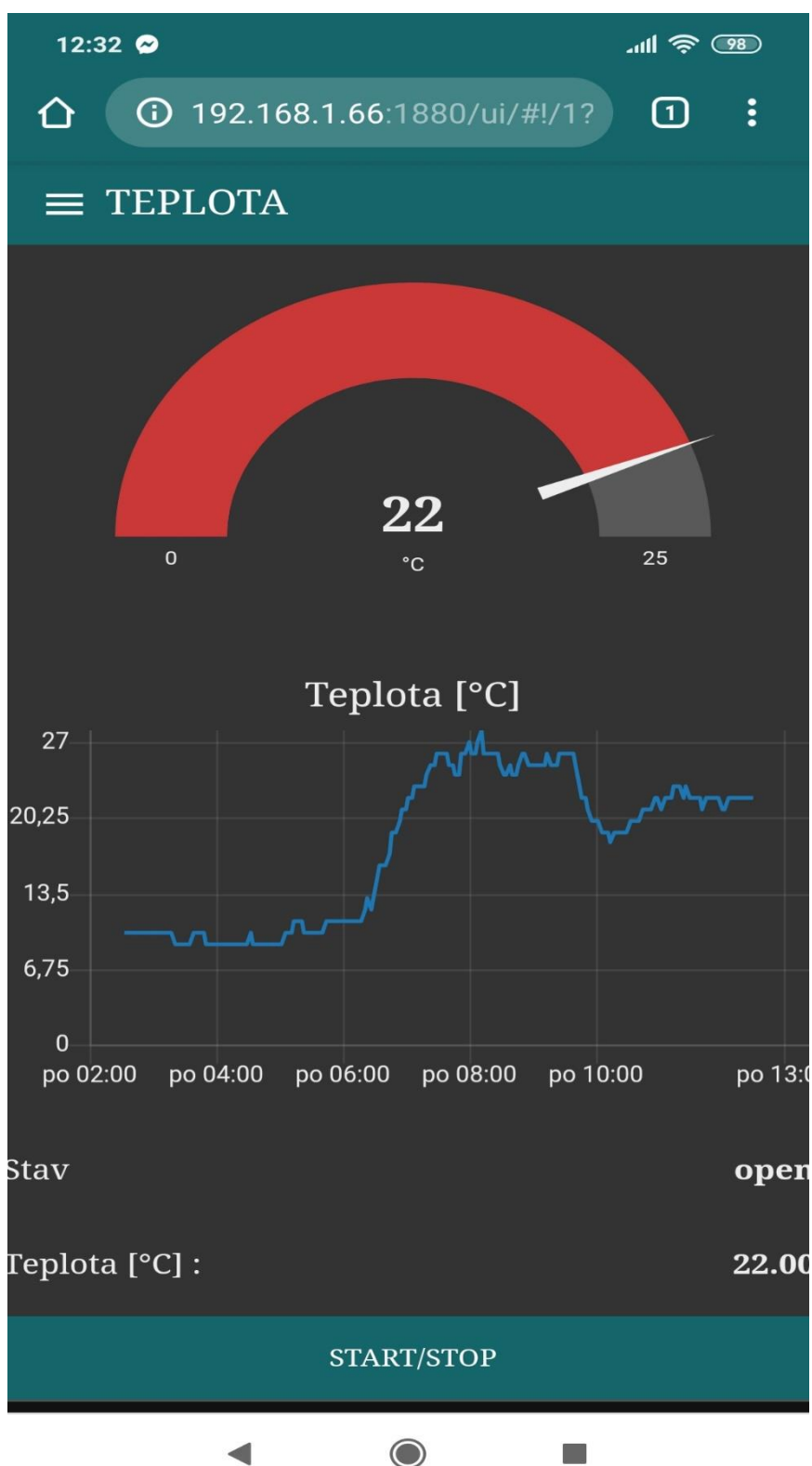
Obr. 13 Zobrazenie ponuky v hlavnom menu stránky

Meteo-stanica celkovo:



Obr. 14 Náhľad rozhrania meteo-stanice pre celkové snímané hodnoty

Vykreslenie teploty pomocou mobilu:



Obr. 15 Náhľad rozhrania pre jednotlivý blok - [Teplota] zobrazený pomocou mobilu

## 5. Záver

Výsledkom práce je snímanie teploty a vlhkosti vzduchu, kde si vďaka platforme Raspberry pi a senzoru DHT11 dokážeme vytvoriť jednoduchú meteo-stanicu a online sledovať na web-serveri namerané veličiny.

V projekte sme si jednoduchým kódom získali hodnoty zo snímača a nechali ich vypísať do súboru. Následne sme použili hodnoty zo súboru využili na archiváciu údajov a vykreslenie histórie priamo na server cez klienta, kde sme využili kód v Pythone a Javascript.

Nakoniec sme si vyskúšali aj vytvoriť užívateľské prostredie na serveri pomocou programu Node-Red, kde sledujeme aktuálne hodnoty teplôt či vlhkosti vzduchu. Tento program bol vybraný pre rozšírenie vedomostí v rámci predmetu pokročilé informačné technológie a skúške aj niečoho iného ako bolo využité na cvičeniach.

## Prílohy

DOKUMENT ARCHITEKTÚRY APLIKÁCIE :

[Architektúra RPi.pdf](#)

LINK:

Všetky podklady sú nahrané na verzionovací systém : <https://github.com/Libko99/POIT2020>