Autorzy	
Kostiukov Oleksii	231972
Uladzimir Lipski	238961
Vikatr Hasiul	231862

Prowadzący	Dr inż. Marek Woda
Termin	Środa, godzina: 13:15

Zastosowanie systemów wbudowanych

Raspberry Pi - Telegram Bot i web-aplikacja dla śledzenia klimatu otoczenia

1 Cel projektu

Celem danego projektu jest wykorzystanie platformy *Raspberry* do realizacji *Telegram bot'u* oraz punktu pomiarowego.

Telegam bot jest aplikacją wykorzystującą interfejs aplikacji *Telegram* w celu komunikacji z wybranymi użytkownikami (którzy posiadają możliwość komunikacji ze stworzonym *bot'em*).

Punkt pomiarowy. Platforma *Raspberry* umożliwia podłączenie licznych czujników, dane z których można gromadzić na urządzeniu lub wysyłać do serwerów zdalnych. W danym projekcie zostaną podłączone czujniki:

1. temperatury,
2. wilgotności,
3. światła

dane z których będą przechowywane na urządzeniu w celu przetwarzania i wyświetlania na stronie **WEB** w postaci interaktywnego wykresu.

Dodatkowo dany zbiór danych zostanie wykorzystany przez *Telegram bot* w celu powiadomienia użytkownika o aktualnych danych.

2 Implementacja

2.1 Pobór danych za pomocą czujników

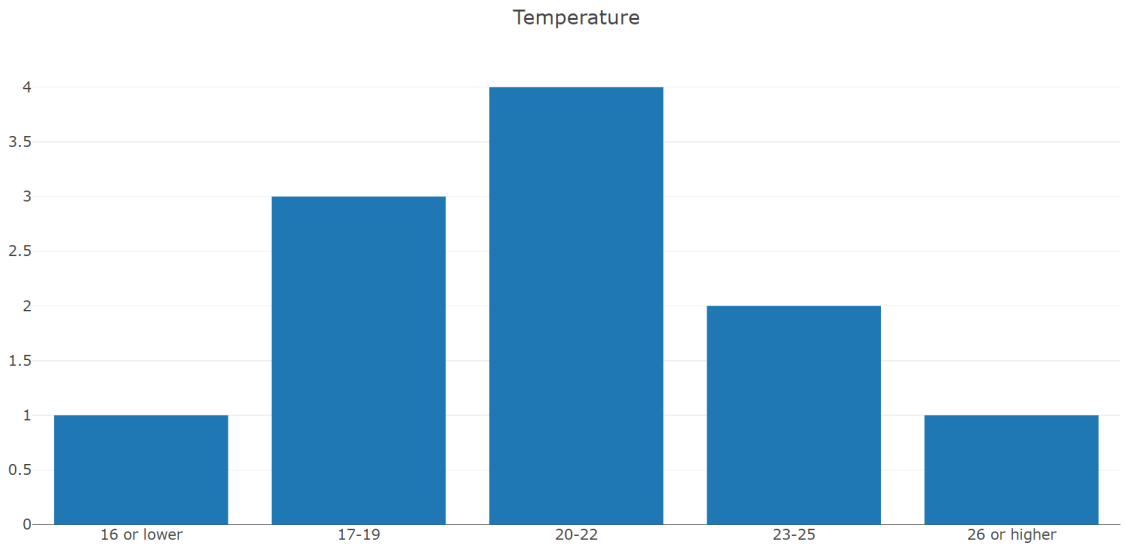
2.2 Podłączenie do sieci Internet

2.3 Komunikacja z Telegram botem

2.4 Aplikacja internetowa

2.4.1 Cel aplikacji internetowej

Celem aplikacji internetowej jest wizualizacja danych, pobranych za pomocą czujników platformy Raspberry Pi. Dane są wyświetlane za pomocą wykresów, a mianowicie histogramów. Wykresy pokazują jak często występuje każdy z zadanych zakresów temperatury, natężenia światła albo wilgotności. Wykresy w przeglądarce muszą być odświeżone asynchronicznie od razu po zmianie pliku, które te dane przechowuje.



Rysunek 1: Histogram temperatury

2.4.2 Wybór platformy web-serwera

Aplikacja internetowa jest stworzona za pomocą frameworku Flask. Flask - framework do tworzenia aplikacji internetowych w języku Python, należący do kategorii tak zwanych mikroframeworkow - minimalistycznych ram aplikacji internetowych, które celowo zapewniają tylko najbardziej podstawowe funkcje.

Zaletami frameworku sa:

1. Elastyczność
2. Minimalistyczność bez straty mocy
3. Prosty w nauce i obsłudze
4. Łatwe rutowanie adresów URL
5. Łatwa rozszerzalność

2.4.3 Generowanie wykresów w przeglądarce

Dla utworzenia wykresów w przeglądarce używana jest biblioteka Plotly, napisana w języku JavaScript. Biblioteka pozwala na utworzenie wykresu wybranego typu, który dla danej aplikacji jest typem 'bar', odpowiadającym histogramowi. Dane dla wykresów są przekazywane od serwera do przeglądarki jako odpowiedz na żądanie HTTP.

Funkcja do utworzenia wykresu przyjmuje jako parametr "id" elementu z tagiem "div", w którym będzie się znajdować dany wykres.

```
function drawChart(chart_divId, chart_title) {
    var trace = {
        type: 'bar',
        x: chartRanges[chart_title],
        y: chartData[chart_title],
    };

    var data = [ trace ];

    chart_title = chart_title.charAt(0).toUpperCase() + chart_title.slice(1);
    var layout = {
      title: chart_title,
      font: {size: 18}
    };

    Plotly.react(chart_divId, data, layout);
}
```

Rysunek 2: Funkcja do tworzenia histogramu za pomocą biblioteki Plotly.js

Dane z serwera są przekazywane w postaci tablicy. Dodatkowym zadaniem skryptu w języku JavaScript jest kategoryzowanie (rozrzucenie) danych dla różnych zakresow, zanim wykres będzie stworzony.

Wartości dla osi X są nazwami zakresów, do których może należeć wartość przekazana z serwera. Dane zakresy są zdefiniowane są statyczne i zadeklarowane na samym początku skryptu.

```
var chartRanges = {
            "temperature" : ["16 or lower", "17-19", "20-22", "23-25", "26 or higher"],
            "light": ["349 or lower","350-450", "451-550", "551-650", "651 or higher"],
            "humidity":["0-20", "21-40", "41-60", "61-80", "81-100"]
            }
```

Rysunek 3: Zdefiniowany zakresy w skrypcie JavaScript

2.4.4 Asynchroniczne wczytywanie danych z pliku

Dane, pobrane z czujników za pomocą platformy Raspberry Pi, są przechowywane w pliku z rozszerzeniem ".csv". Dla asynchronicznego wczytywania danych z pliku używana jest biblioteka "watchdog", napisana w języku Python.

Celem biblioteki jest monitorowanie wybranych plików. Wybór plików odbywa się za pomocą wyrażeń regularnych. Jeżeli jakiś z monitorowanych plików zostaje zmieniony, to do serwera jest wysyłany sygnał. Do obsługi różnego rodzaju sygnałów używana jest biblioteka "flask_socketio".

Dla każdego sygnału mogą być zdefiniowane osobne funkcje. Na przykład jeżeli monitorowany plik został zmieniony i zapisany, to do serwera zostaje wysłany sygnał, który wywołuje funkcje "on_modified", która była zdefiniowana dla śledzenia danego sygnału.

Po otrzymaniu sygnalu funkcja "on_modified" linijka po linijce wczytuje dane z pliku dlatego, żeby te dane wysłać do przeglądarki.

Monitorowanie pliku jest osobna funkcja i to monitorowanie musi się odbywać w osobnym wątku, żeby nie blokować aplikacji.

```
@socketio.on('connect')
def test_connect():
    global thread
    if thread is None:
        thread = socketio.start_background_task(target=background_thread)
```

Rysunek 4: Początek wątku, monitorującego zmianę w pliku

```
class CsvWatcher(RegexMatchingEventHandler):
    csv_file = [r".*\.csv"]
    def __init__(self):
        super().__init__(self.csv_file)

    def on_modified(self,event):
        global csvFileName
        data = readfile(csvFileName)
        socketio.emit('modified', {'data': data})


def background_thread():
    global observer
    path = os.path.join("..", "Sensors")
    event_handler = CsvWatcher()
    observer = Observer()
    observer.schedule(event_handler, path, recursive=True)
    observer.start()
```

Rysunek 5: Funkcja reagująca na sygnał po zmianie pliku

2.4.5 Asynchroniczne odświeżanie wykresów w przeglądarce

Język JavaScript również pozwala na zdefiniowanie socket'ów. Celem socketów jest asynchroniczna obsługa sygnałów, wysłanych z serwera.

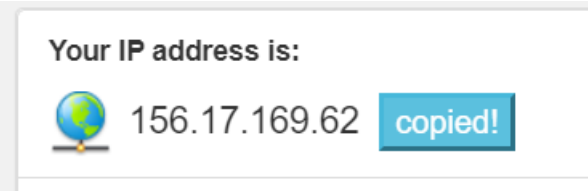
Jak jest pokazane na rysunku 5, po zmianie pliku serwer wywołuje funkcje "on_modified", która wczytuje plik i wysyła sygnał z wczytanymi danymi do przeglądarki. Po otrzymaniu sygnału skrypt JavaScript odświeża wykresy w przeglądarce.

```
var socket = io.connect('http://' + document.domain + ':' + location.port);

socket.on('modified', function(data) {
    var newData = data['data'];
    start(newData)
});

function start(data){
        initializeArrays();
        refreshData(data);
}
```

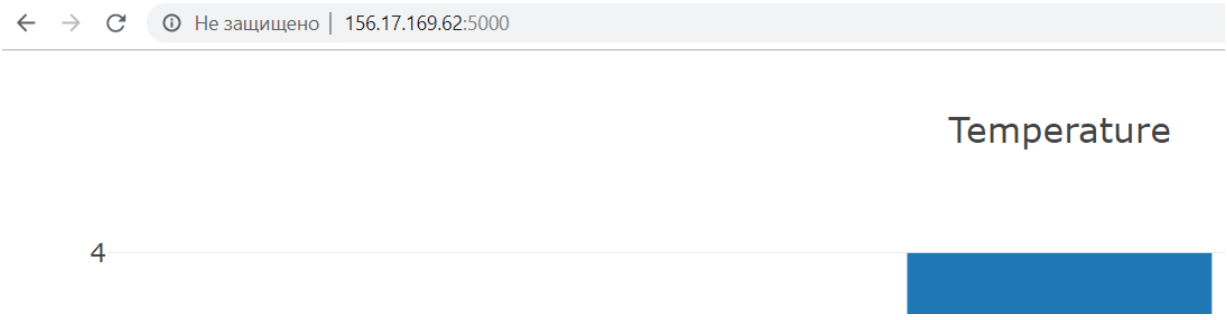
Rysunek 6: Funkcja odświeżająca wykresy po otrzymaniu sygnału z serwera

2.4.6 Dostęp do aplikacji w sieci prywatnej

Dla uruchamiania aplikacji ktora będzie dostępna w sieci prywatnej, potrzebne jest odpowiednie ustawienie parametru "host". Jeżeli parametr "host" będzie ustawiony na "0.0.0.0", to aplikacja będzie dostępna pod adresem, którym jest zewnętrzny adres IP. Adres IP można sprawdzić na przykład na stronie https://www.myip.com/.



Rysunek 7: Adres IP na stronie https://www.myip.com/



Rysunek 8: Uruchomiona aplikacja pod zewnętrznym adresem IP