POLITECHNIKA POZNAŃSKA

WYDZIAŁ AUTOMATYKI, ROBOTYKI I ELEKTROTECHNIKI

INSTYTUT ROBOTYKI I INTELIGENCJI MASZYNOWEJ

ZAKŁAD STEROWANIA I ELEKTRONIKI PRZEMYSŁOWEJ

SPRAWOZDANIE

APLIKACJA WEBOWA

JQUERY, CSS

JĘDRZEJ GRZEBISZ

(AUTOR/AUTORZY)

APLIKACJE MOBILNE I WBUDOWANE DLA INTERNETU PRZEDMIOTÓW

(NAZWA PRZEDMIOTU)

LABORATORIUM

(FORMA ZAJĘĆ {PROJEKT/LABORATORIUM/ĆWICZENIA})

PROWADZĄCY:

MGR INŻ. ADRIAN WÓJCIK

ADRIAN.WOJCIK@PUT.POZNAN.PL

POZNAŃ 15/06/2020

DANE SZCZEGÓŁOWE

Rok studiów: INŻ. III

Rok akademicki: 2019/2020

Termin zajęć: środa g. 13:30

Forma zajęć: online

Data wykonania: 2020/06/15

Temat projektu: Aplikacja webowa jQuery, CSS

Prowadzący: Adrian Wójcik

Skład grupy (nazwisko, imię, nr indeksu):

1. Grzebisz Jędrzej 135825

OCENA (*WYPEŁNIA PROWADZĄCY*)

Spełnienie wymogów redakcyjnych: ……

……

Wykonanie, udokumentowanie oraz opis wykonanych zadań: ……

……

Zastosowanie prawidłowego warsztatu programistycznego: ……

……

Spis treści

1 APLIKACJA WEBOWA

1.1 SPECYFIKACJA

Celem zadania było stworzenie z użyciem CSS oraz biblioteki jQuery wykresów wyświetlających położenie kątowe nakładki SenseHat z maszyny wirtualnej oraz obsługę matrycy LED na tejże nakładce. Umożliwione zostało także ustawienie przez użytkownika czasu próbkowania oraz maksymalnej liczby próbek dla wykresów. Wykres oraz obsługa matrycy możliwa jest do uruchomienia z jednego głównego pliku html.

1.2 MATRYCA LED – WARSTWA HTML/CSS

Na poniższym listingu prezentuję kod w języku HTML(fragment <body>) dla widoku obsługi matrycy LED. Korzystam tutaj z bloków DIV dla podzielenia strony, wykorzystywane są również pola input, z kilkoma pomocniczymi atrybutami(np. type, placeholder), czy też lista wyboru select-option. Konkretne divy oraz inne znaczniki mają przypisane konkretne id oraz klasę, by łatwo obsłużyć plik ze stylami.

```
1.  <body>
2.      <div id="container">
3.          <div id="header" class="mainHeader">
4.              Matryca LED
5.          </div>
6.          <div id="singleLedHeader" class="contentHeader">
7.              Pojedyncza dioda
8.          </div>
9.          <div id="textLedHeader" class="contentHeader">
10.             Tekst
11.         </div>
12.         <div id="singleLedContent" class="content">
13.             <input id="rowNumber" class="singleLedInput" type="number" placeho
    lder="Numer wiersza (0-7)" min="0" max="7" step="1">
14.             <input id="columnNumber" class="singleLedInput" type="number" plac
    eholder="Numer kulumny (0-7)" min="0" max="7" step="1"><br>
15.             Wybierz kolor:<br>
16.             <div class="selectDiv">
17.                 <select id="singleLedColor" class="colorSelect">
18.                     <option value="red">Red</option>
19.                     <option value="green">Green</option>
20.                     <option value="blue">Blue</option>
21.                     <option value="orange">Orange</option>
22.                     <option value="white">White</option>
23.                 </select>
24.                 <button type="button" class="button" id="singleLedButton">Zapa
    l</button>
25.             </div>
26.
27.         </div>
28.         <div id="textLedContent" class="content">
29.             <input id="textLed" class="textLedInput" type="text" placeholder="
    Wpisz tekst"><br>
30.             Wybierz kolor:<br>
31.             <div class="selectDiv">
32.                 <select id="textLedColor" class="colorSelect">
33.                     <option value="red">Red</option>
34.                     <option value="green">Green</option>
35.                     <option value="blue">Blue</option>
36.                     <option value="orange">Orange</option>
37.                     <option value="white">White</option>
38.                 </select>
39.                 <button type="button" class="button" id="textLedButton">Zapal<
    /button>
```

```
            </div>
        </div>
    </div>
</body>
```

Listing 1 Kod w HTML dla obsługi matrycy LED

Na kolejnych listingach 2, 3, 4. zaprezentowane zostały wybrane fragmenty z pliku .css dla matrycy LED.

```
#container {
    width: 800px;
    height: 330px;
    padding: 10px 10px 10px 10px;
    margin: auto;
    background-color: #666666;
    display: grid;
    grid-column-gap: 40px;
    grid-row-gap: 15px;
    grid-template-columns: 1fr 1fr;
    grid-template-rows: 50px 50px 200px;
    grid-template-areas:
        "header header"
        "singleLedHeader textLedHeader"
        "singleLedContent textLedContent";
}

.mainHeader {
    grid-area: header;
    height: 50px;
    background-color: #00ACEE;
    font-family: 'Josefin Sans', sans-serif;
    color: white;
    font-size: 40px;
    text-align: center;
    justify-content: center;
}

.contentHeader {
    grid-area: textLedHeader, singleLedHeader;
    font-family: 'Josefin Sans', sans-serif;
    color: white;
    font-size: 30px;
    text-align: center;
    padding: 10px 0px 10px 0px;
    background-color: #009BDD;
}
```

Listing 2 Plik stylizujący widok matrycy cz1

Całość widoku starałem się wykonać wykorzystując CSS Grid, dzieląc całość na kolumny oraz wiersze. Na powyższym fragmencie widzimy style dla całego kontenera trzymającego widok, oraz styl dla nagłówka głównego oraz podrzędnych. Sam rezultat w przeglądarce można zobaczyć na rysunku 1.

```
.content {
    font-family: 'Josefin Sans', sans-serif;
    font-size: 20px;
    grid-area: singleLedContent, textLedContent;
    padding: 10px;
    background-color: #80DD80;
    color: white;
}

```

```
.singleLedInput {
    width: 50%;
    border: 2px solid #aaa;
    border-radius: 4px;
    margin: 8px 8px 8px 0px;
    outline: none;
    transition: 0.3s;
    padding: 8px;
    box-sizing: border-box;
}
```

Listing 3 Plik stylizujący widok matrycy cz2

W powyższym fragmencie mamy zaprezentowany styl klas content oraz singleLedInput.

```
 .colorSelect {
    width: 50%;
    -webkit-appearance: none;
    cursor: pointer;
    font-family: 'Josefin Sans', sans-serif;
    font-size: 15px;
    background: #fff;
    border: 2px solid #aaa;
    border-radius: 4px;
    padding: 8px;
    margin: 8px 8px 8px 0px;
    outline: none;
    transition: 0.3s;
}

.button {
    background-color: #f0fb3e;
    border: none;
    color: #666666;
    text-align: center;
    font-size: 20px;
    margin-left: 20px;
    height: 50px;
    width: 100px;
    opacity: 0.8;
    transition: 0.3s;
    text-decoration: none;
    cursor: pointer;
  }
```

Listing 4 Plik stylizujący widok matrycy cz3

Ostatni zaprezentowany fragment pokazuje jak wyglądała stylizacja listy wyboru koloru oraz przycisku zapal.

1.3 MATRYCA LED – WARSTWA JS

W tej części zaprezentowane zostanie działanie obsługi matrycy po stronie JavaScript.

W celu komunikacji z plikiem php wykorzystuję Ajaxa, w „mainie” skryptu na klinięcie odpowiedniego przycisku „Zapal” uruchamiamy funkcję ajaxSend z odpowiednim argumentem(event) w zależności czy wybieramy zapalenie pojedynczej diody, czy też wyświetlenie tekstu. Do php wysyłamy odpowiednie dane odczytane z pól do wprowadzania danych, dane te wysyłamy jako obiekt JS zgodny ze standardem json.

```
1.  let urlAddress;
2.  let dataDict;
3.
4.  function ajaxSend(event)
5.  {
6.      if (event.data.type == "single")
7.      {
8.          urlAddress = 'webApp/singleLedColor.php'
9.          dataDict = {
10.              row: $('#rowNumber').val(),
11.              column: $('#columnNumber').val(),
12.              color: $('#singleLedColor').val()
13.          }
14.     }
15.     else if (event.data.type == "text")
16.     {
17.         urlAddress = 'webApp/textLedColor.php'
18.         dataDict = {
19.             text: $('#textLed').val(),
20.             color: $('#textLedColor').val()
21.         }
22.     }
23.
24.     $.ajax({
25.         type: 'GET',
26.         url: urlAddress,
27.         data: dataDict,
28.         success: function(response) {
29.             console.log("Done");
30.             console.log(dataDict);
31.         }
32.     });
33. }
34.
35. $(document).ready(() => {
36.     $('#singleLedButton').click({type: "single"} ,ajaxSend);
37.     $('#textLedButton').click({type: "text"} ,ajaxSend);
38. });
```

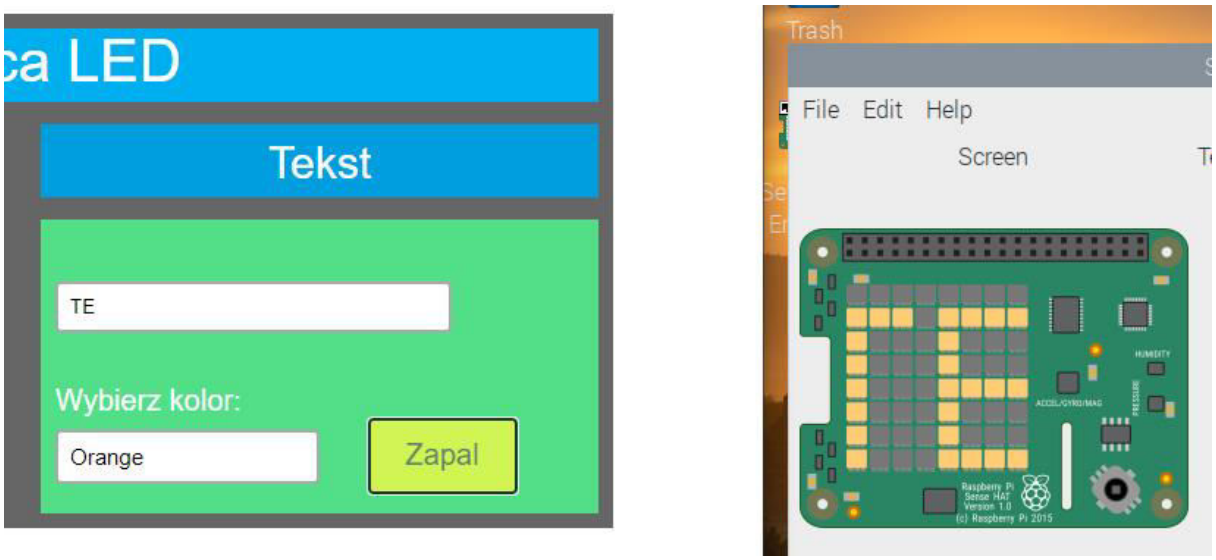
Listing 5 Kod JavaSrcipt dla widoku obsługi matrycy LED

W dalszej części pomijam wstawianie kodu z pliku php oraz skryptu pythona obsługującego nakładkę SenseHat, gdyż są to te same pliki, które wykorzystywane były chociażby przy tworzeniu aplikacji w Androidzie.

1.4 MATRYCA LED – DZIAŁANIE

Na kolejnych rysunkach 1 oraz 2 widać wygląd opisywanego widoku oraz jego działanie w zestawieniu z nakładką SenseHat.

Rysunek 1 Widok obsługi matrycy LED



Rysunek 2 Działanie prezentowanego widoku

1.5 WYKRESY RPY – WARSTWA HTML/CSS

W tej części zaprezentowany zostanie widok wykresów położenia kątowego nakładki SenseHat. Same wykresy zostały zrealizowane na bazie wzoru dostarczonego przez prowadzącego. Dodana została jednak możliwość edycji czasu próbkowania oraz maksymalnej liczby zapamiętywanych próbek jako pola, które może obsłużyć użytkownik.

Na listingu 6 widać kod HTML widoku wykresów

```
<!DOCTYPE html>
<html lang="pl">
    <!-- HTML file head: web page metadata -->
<head>
    <meta charset="utf-8">
    <title>Pomiar kątów RPY</title>
    <!-- jQuery -->
    <script type="text/javascript" src="https://cdn.jsdelivr.net/npm/jquery@3.2.1/dist/jquery.min.js"></script>

    <!-- Latest compiled and minified CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">

    <!-- Latest compiled and minified JavaScript -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"></script>

    <!-- Chart.js library -->
    <script type="text/javascript" src="https://cdn.jsdelivr.net/npm/chart.js@2.8.0"></script>

    <!-- JavaScript file -->
    <script type="text/javascript" src="scripts/rpyChart.js"></script>

    <!-- CSS style for chart -->
    <link rel="stylesheet" type="text/css" href="styles/rpyChart.css">
</head>

<!-- HTML file body: web page content -->
<body>
    <h1 class="text-center">Roll Pitch Yaw</h1>
    <div id="datagrabber" class="container border border-primary rounded datagrabber">
        <ul class="nav">
            <li><button id="start" type="button" class="btn btn-primary menuitem">Start</button></li>
            <li><button id="stop" type="button" class="btn btn-primary menuitem">Stop</button></li>
            <li><div class="border border-primary rounded menuitem"><span>Czas próbkowania: <span id="sampletime"></span> ms</span></div></li>
            <li><div class="border border-primary rounded menuitem"><span>Max. liczba probek: <span id="samplenumber"></span></span></div></li>
        </ul>
        <div class="chartcontainer">
            <canvas id="chart"></canvas>
        </div>
        <div class="config">
            <input id="sampleTime" class="userInput" type="number" placeholder="Czas próbkowania[ms]">
            <input id="maxSamplesNumber" class="userInput" type="number" placeholder="Max. liczba próbek">
            <button type="button" id="saveSettings" class="btn btn-primary menuitem">Ustaw</button>
```

```
        </div>
    </div>
</body>

</html>
```

Listing 6 Kod HTML widoku wykresów

W tym wypadku wykorzystywane są dodatkowe style Bootstrap, jedynie pola do wprowadzenia danych przez użytkownika na temat czasu próbkowani i maksymalnej liczby próbek zostały ostylowane przeze mnie w CSS, odpowiadający temu fragment pliku .css zaprezentowany jest na kolejnym listingu.

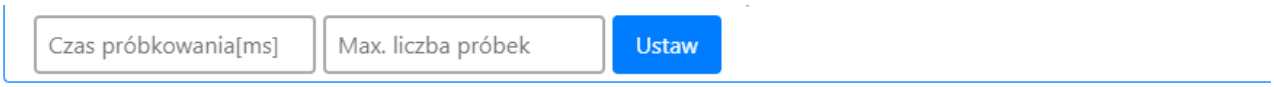
```
.config {
    height: 40px;
    padding-left: 15px;
    padding-right: 15px;
    display: flex;
    flex-direction: row;
}

.userInput {
    border: 2px solid #aaa;
    border-radius: 4px;
    margin-right: 5px;
    outline: none;
    transition: 0.3s;
    padding: 8px;
    box-sizing: border-box;
}

.userInput:focus {
    border-color: dodgerblue;
    box-shadow: 0 0 8px 0 dodgerblue;
}
```

Listing 7 Kod CSS widoku wykresów

W tym wypadku zastosowany został Flexbox który umożliwia proste umieszczanie elementów obok siebie w HTML, dzięki temu uzyskałem efekt taki jak na poniższym rysunku.



Rysunek 3 Fragment ustawień użytkownika dla wykresów

1.6 WYKRESY RPY – WARSTWA JS

Na kolejnych listingach prezentuję fragmenty kodu w JS obsługujące pokazywanie wykresów na ekranie, pomijając fragment w którym zadeklarowane zostały wszelakie zmienne użyte w programie.

```
1.  /**
2.  * @brief Set selected sample time and maximum number of samples
3.  */
4.  function setSettings()
5.  {
6.      if ($('#sampleTime').val())
7.      {
8.          sampleTimeMsec = parseInt($('#sampleTime').val());
9.          sampleTimeSec = sampleTimeMsec/1000;
10.         $("#sampletime").text(sampleTimeMsec.toString());
11.     }
12.     if ($('#maxSamplesNumber').val())
13.     {
14.         maxSamplesNumber = parseInt($('#maxSamplesNumber').val());
15.         $("#samplenumber").text(maxSamplesNumber.toString());
16.     }
17.
18.     configDict = {
19.             sampleTimeMsc: sampleTimeMsec,
20.             maxSamplesNb: maxSamplesNumber
21.             }
22.
23.     $.ajax({
24.         type: 'GET',
25.         url: 'webApp/config.php',
26.         data: configDict,
27.         success: function(response) {
28.             console.log("configOK");
29.             console.log(configDict);
30.         }
31.     });
32. }
```

Listing 8 Kod w JS - aktualizacja ustawień, zapis do pliku

Powyższa funkcja wywoływana jest w momencie naciśnięcia przycisku Ustaw(wywołanie w document,ready()) jeżeli pola dla czasu próbkowania oraz maksymalnej liczby próbek nie są puste, to aktualna wartość sampleTime oraz maxSamplesNumber jest nadpisywana, tworzony jest obiekt JS przechowujący te wartości oraz całość wysyłana jest do pliku PHP, analogicznie jak dla zapalania matrycy LED

```
1.  /**
2.  * @brief Add new values to next data point.
3.  * @param r New y-axis roll value
4.  * @param p New y-axis pitch value
5.  * @param y New y-axis yaw value
6.  */
7.  function addData(r, p, y)
8.  {
9.      if(yRollData.length > maxSamplesNumber) // same length of roll, pitch, yaw
10.     {
11.         removeOldData();
12.         lastTimeStamp += sampleTimeSec;
13.         xData.push(lastTimeStamp.toFixed(4));
14.     }
15.     yRollData.push(r);
```

```
16.     yPitchData.push(p);
17.     yYawData.push(y);
18.     myChart.update();
19. }
20.
21. /**
22. * @brief Remove oldest data point.
23. */
24. function removeOldData()
25. {
26.     xData.splice(0,1);
27.     yRollData.splice(0,1);
28.     yPitchData.splice(0,1);
29.     yYawData.splice(0,1);
30. }
31.
32. /**
33. * @brief Start request timer
34. */
35. function startTimer()
36. {
37.     timer = setInterval(ajaxJSON, sampleTimeMsec);
38. }
39.
40. /**
41. * @brief Stop request timer
42. */
43. function stopTimer()
44. {
45.     clearInterval(timer);
46. }
```

Listing 9 Kod w JS - funkcje dodania, usuwanie, startu i zatrzymania timera

Powyżej widać kod kilku funkcji, który w zasadzie nie był szczególnie zmieniany w porównaniu do dostarczonych przykładów, jedynie do funkcji addData przekazywane są trzy argumenty zawierające dane o r, p, y wstawiane są one do odpowiednich tablic oraz aktualizowany jest wykres.

```
1.  function ajaxJSON()
2.  {
3.      $.ajax(urlDefault, {
4.          type: 'GET', dataType: 'json',
5.          success: function(responseJSON, status, xhr) {
6.              addData(+responseJSON.Roll, +responseJSON.Pitch, +responseJSON.Yaw
    );
7.          }
8.      });
9.  }
```

Listing 10 Kod w JS - zapytanie z wykorzystaniem Ajax

Powyższa funkcja wysyła zapytanie, którego celem jest pobranie danych z pliku odnośnie położenia kątowego przechowywanego w pliku w formacie json.

Fragment kodu odpowiedzialny za inicjalizację wykresu został pominięty, gdyż jest on identyczny jak w dostarczonych materiałach z drobnymi różnicami, takimi jak zmiana koloru, tytułu oraz tym że jest przystosowany do prezentowania trzech przebiegów zamiast jednego.

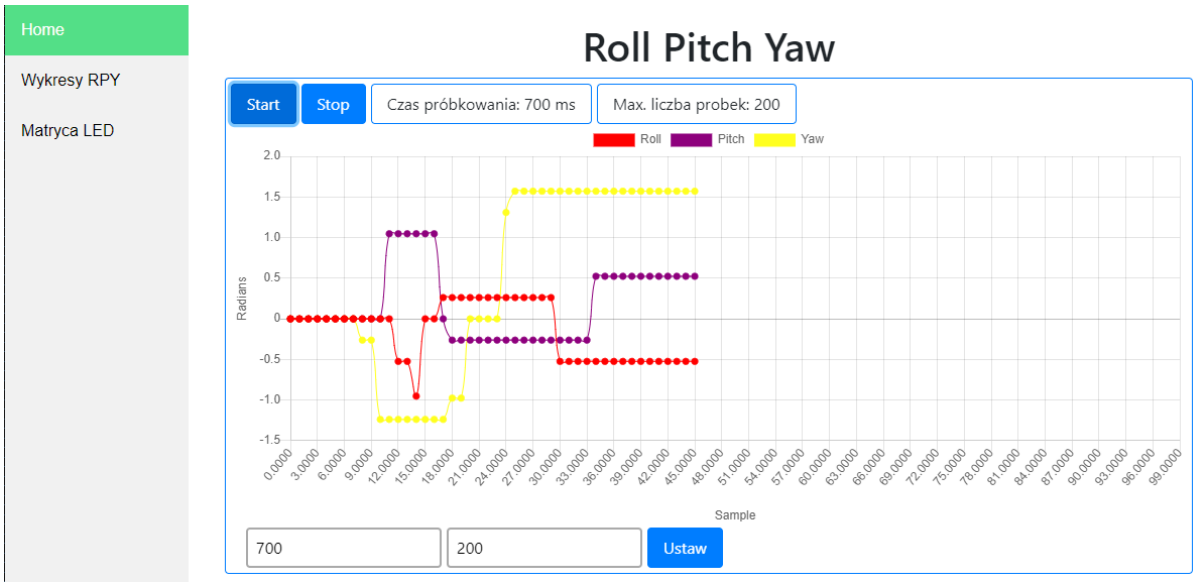
```
1.  $(document).ready(() => {
2.      chartInit();
3.      $("#start").click(startTimer);
4.      $("#stop").click(stopTimer);
5.      $("#saveSettings").click(setSettings);
6.      $("#sampletime").text(defaultSampleTimeMsec.toString());
7.      $("#samplenumber").text(defaultMaxSamplesNumber.toString());
8.  });
```

Listing 11 Kod w JS - document.ready()

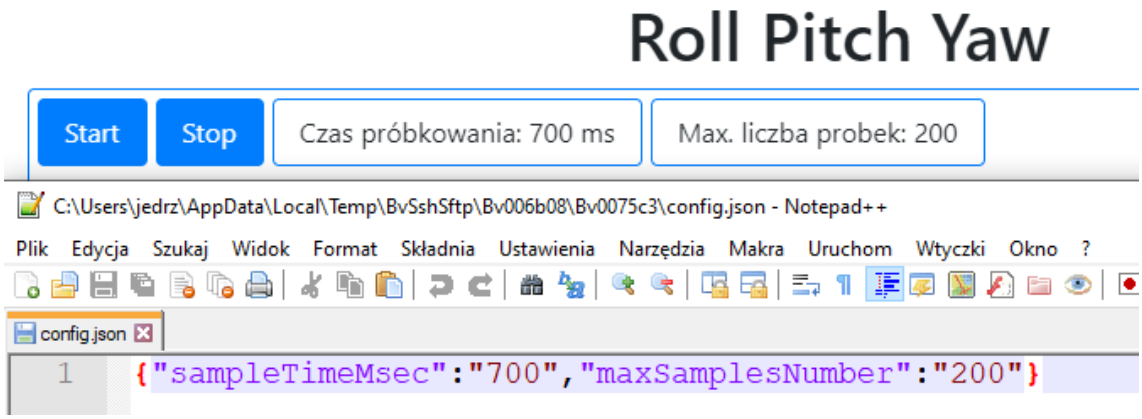
Ostatni już fragment to rzeczy dziejące się po załadowaniu strony, jak widać inicjalizujemy tutaj wykres, wywołujemy funkcje start oraz stopTimer po wykryciu naciśnięcia odpowiedniego przycisku, to samo tyczy się zapisu ustawień. Na końcu wyświetlamy w odpowiednich polach dane odnośnie czasu próbkowania oraz maksymalnej liczby próbek.

1.7 WYKRESY RPY – DZIAŁANIE

Poniższy rysunek pokazuje działanie wykresów oraz podgląd jak wygląda plik przechowujący dane wprowadzane od użytkownika na serwerze.



Rysunek 4 Wykresy RPY – działanie



Rysunek 5 Wykresy RPY - plik z konfiguracją

1.8 CAŁOŚĆ W JEDNYM WIDOKU

Jak widać na screenach z działania aplikacji, po lewej stronie mamy menu pozwalające przełączać się pomiędzy zaimplementowanymi widokami. Kod w HTML oraz CSS dla całości widoczne są na ostatnich dwóch listingach.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Aplikacja Webowa</title>
  <link rel="stylesheet" href="styles/index.css">
</head>
<body>

  <ul>
    <li><a class="active" href="">Home</a></li>
    <li><a href="rpyChart.html" target="frame">Wykresy RPY</a></li>
    <li><a href="ledMatrix.html" target="frame">Matryca LED</a></li>
  </ul>

  <div class="frameDiv">
    <iframe name="frame" width="1000px" height="570px" style="border:0;"></ifr
ame>
  </div>

</body>
```

Listing 12 Kod HTML - wspólny widok

Mamy tutaj bardzo prosty szablon strony, po kliknięciu na dany element w menu w <iframe> wstawiany jest wybrany widok.

```
body {
  margin: 0;
}

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  width: 15%;
  background-color: #f1f1f1;
  position: fixed;
  height: 75%;
  overflow: auto;
}

li a {
  display: block;
  color: #000;
  padding: 16px 16px;
  text-decoration: none;
  font-family: 'Josefin Sans', sans-serif;
}

li a.active {
  background-color: #80DD80;
  color: white;
}

li a:hover {
  background-color: #009BDD;
  color: white;
}
```

```
33.
34. .frameDiv {
35.   margin-left: 15%;
36.   padding: 16px 16px;
37. }
```

Listing 13 Kod CSS - wspólny widok, menu

W tworzeniu bocznego menu posłużyłem się propozycją ze strony w3schools, która wydaje się dość prosta i jak najbardziej spełnia swoją rolę.