UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
KATEDRA ZA MIKRORAČUNARSKU ELEKTRONIKU

# Server za monitoring embedded sistema

Projekat iz predmeta Računarska Elektronika

Studenti :
Arpad Kočiš, EE-117/2015
Arnold Mesaroš, EE-105/2015

Mentor :
Prof. dr Ivan Mezei

Jun 2020

# Sadržaj

# SKRAĆENICE

| Akronim | Opis |
|---|---|
| IoT | Internet of Things |
| GPIO | General Purpose Input Output |
| HTML | HyperText Markup Language |
| HTTP | HyperText Transfer Protocol |
| PHP | Hypertext Preprocessor |
| ADC | Analog to Digital Converter |
| UI | User Interface |
| | |
| | |
| | |
| | |
| | |

# POGLAVLJE 1 – Opis sistema

## 1.1 Uvod

U prvom poglavlju ćemo opisati i pokazati rad našeg sistema, preko narednih nekoliko slika. Drugo poglavlje opisuje data uploadere, tj. aplikacije koje vrše očitavanje i slanje vrednosti sensora na server. U trećem poglavlju ćemo predstaviti rad web servera, php stranica i baze podataka u pozadini platforme. U četvrtom poglavlju iznesen je zaključak, dok peto poglavlje predstavlja spisak literature korištene prilikom izrade projekta.

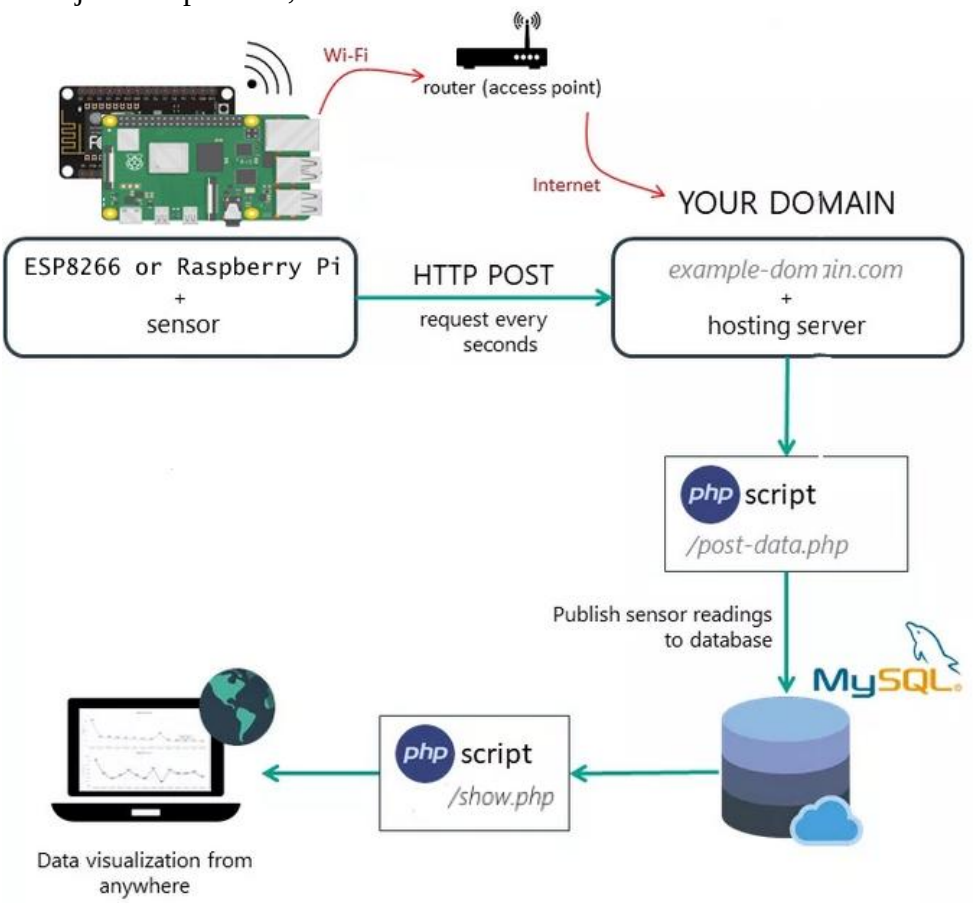
Prvo i drugo poglavlje napisao Arpad Kočiš, treće i četvrto poglavlje Arnold Mesaroš.

## 1.2 Opis sistema

Cilj projekta je razvoj sistema za monitoring senzora na udaljenim embedded uređajima preko interneta. Realizovan sistem sadrži web server, bazu podataka i uređaje koje očitavaju vrednosti senzora i to šalju na platform za praćenje [*1.Slika*]. Rezultat uspešno prikazuje podatke u realnom vremenu, nezavisno od geografske lokacije, sve to preko interneta. Omogućuje i olakšava razvoj savremenih IoT uređaja.

Očitavanje senzora je realizovano na dva načina. U prvom slučaju korišćen je Arduino Uno kao ADC za Raspberry pi. Vrednost fotorezistora očitava arduino uno i vrednost šalje raspberry pi-u preko serijskog interfejsa, koji ga posle iskazuje na grafikonu i šalje na web server HTTP POST metodom.

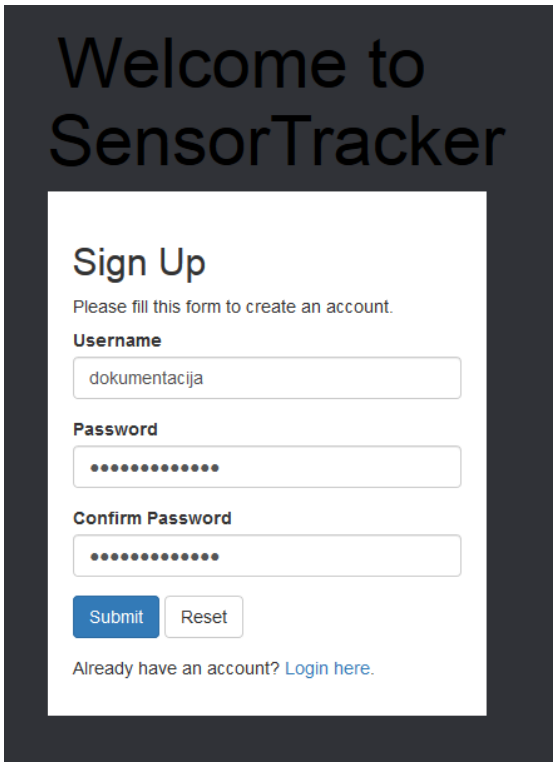
U drugom slučaju korišćen je ESP8266 mikrokontroler za očitavanje drugog fotorezistora. Vrednosti su očitani sa analognih ulaza GPIO pinova. Pročitane vrednosti uređaji šalju na isti web server, istom metodom kao u prvom slučaju. Pristigle vrednosti na web server php kod dodeljuje tabelama u mysql bazi podataka. To vrši na osnovu zadatog parametra, koji je korisničko ime. Platforma vrši vizualizaciju unetih podataka, sa maksimalno tri senzora.



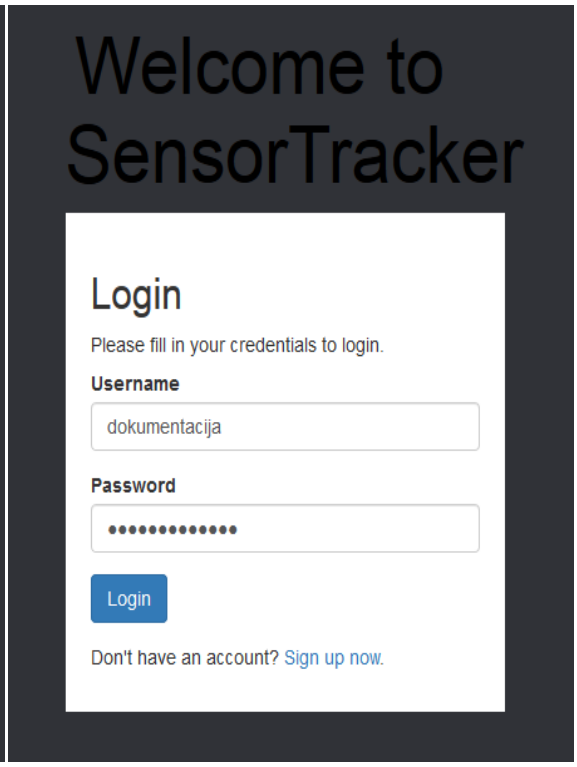

*1. Slika : Blok šema sistema*

## 1.3 Opis platforme

Pristup platformu se radi preko stranice http://sensortracker.zyrosite.com, gde se nalazi dugme "Sign up" i "Login". Klikom na prvi otvori se forma za registraciju [*2.Slika*]. Pristup se vrši klikom na dugme "Login" [*3.Slika*].

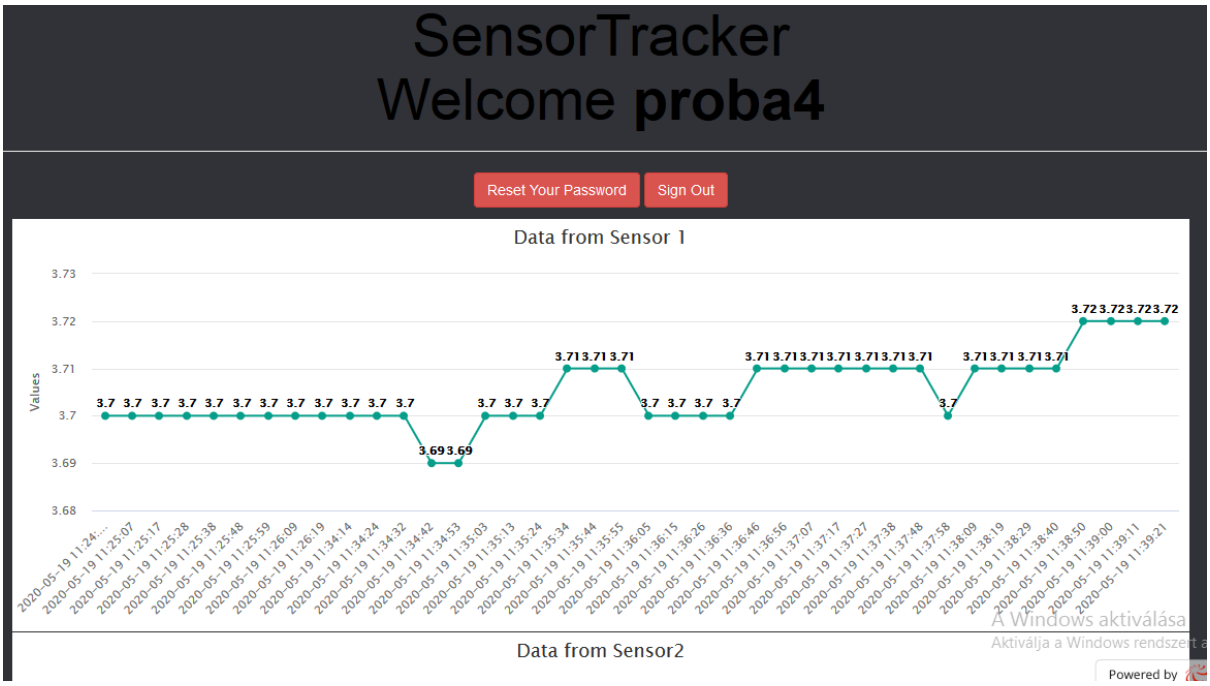



*2. Slika : Kreiranje korisničkog naloga*




*3.Slika : Pristup korisničkom nalogu*

Uspešnom ulogovanjem otvori se korisnički nalog, koji u zavisnosti od unetih vrednosti pokazuje jedan, dva ili tri grafikona. Pored toga korisničko ime, dugme za odjavu i dugme za promenu lozinke [*4.Slika*].




*4. Slika : Početna strana korisničkog servisa*

# POGLAVLJE 2 – Data uploaders

## 2.1 Raspberry pi

Raspberry pi je kompjuter veličine kreditne kartice, koji poseduje GPIO pinove. Ovo svojstvo omogućuje razvoj različitih embedded sistema i korišćenje gomilu senzora [1]. Posle podizanja linux operativnog sistema može da se programira iz daljine ali moguće je i instaliranje Qt razvojnog okruženja. Iz ovog razloga idealan je za testiranje našeg sistema. Razvoj aplikacije je počeo implementacijom očitavanja senzora.

### 2.1.1 Serialna komunikacija Arduina i Raspberry pi

Pri projektovanju hteli smo da čitamo senzor preko ADC-a ali na kraju smo se odlučili za arduino zbog lakoće proširenja broja senzora i zbog familiarnosti sa tom pločom.

Foto otpronik smo vezali redno sa otpornikom od 10kΩ i očitanje se vrši na analognom ulazu A0 na arduinu [*5.Slika*].

```
//Constants
const int pResistor = A0; // Photoresistor at Arduino analog pin A0

//Variables
int value;
//int value_invert;         // Store value from photoresistor (0-1023)

void setup(){
 pinMode(pResistor, INPUT);// Set pResistor - A0 pin as an input (optional)
 Serial.begin(9600);

}

void loop(){
 value = analogRead(pResistor);

 Serial.print(value);
 Serial.print(",");
 Serial.flush();

 delay(10);
}
```

*5. Slika : Očitavanje senzora*

Kod za Arduino je prilično jednostavan. Podesili smo A0 port kao ulaznu, i zatim se očitava vrednost od 0 – 1023. Podesimo Baud rate sa Serial.begin za serijsku komunikaciju sa raspberry pi-om, koju smo vezali USB konekcijom. Ispisujemo vrednost sa Serial.print i dodali smo zarez za kasnije lakše odvajanje podataka.

### 2.1.2 QT i serialna komunikacija

U biblioteci QT postoji dve klase za uspostavljanje serialne komunikacije koje smo koristili, QSerialPort i QSerialPortInfo. Prvo što smo uradili je da očitavanje ID arduina i njegovo upoređivanje sa poznatom identifikacijom [*6.Slika*].

```
foreach(const QSerialPortInfo &serialPortInfo, QSerialPortInfo::availablePorts()){
    //  check if the serialport has both a product identifier and a vendor identifier
    if(serialPortInfo.hasProductIdentifier() && serialPortInfo.hasVendorIdentifier()){
        //  check if the product ID and the vendor ID match those of the arduino uno
        if((serialPortInfo.productIdentifier() == arduino_uno_product_id)
                && (serialPortInfo.vendorIdentifier() == arduino_uno_vendor_id)){
            arduino_is_available = true; //    arduino uno is available on this port
            arduino_uno_port_name = serialPortInfo.portName();
        }
    }
}
```

*6. Slika : Provera identifikacije uređaja*

Zatim sledi podešavanje osnovne stvari da se omogući komunikacija, uspostaviti vezu između funkcije koja čita podatak koja daje signal slotu koji je objekat tipa QSerialPort. U funkciji readSerial čitamo 3 broja  u bafer i zatim ih razdvajamo gde je razmak, na kraju razdvajamo vrednost sa drugog mesta i šaljemo na LCD i graf. U destruktoru je napravljena obična provera, da li je serijalna komunikacija otvorena i ako jeste onda da se isključi [*7.Slika*].

```
void MainWindow::readSerial()
{
    /*
     * readyRead() doesn't guarantee that the entire message will be received all at once.
     * The message can arrive split into parts.  Need to buffer the serial data and then parse for the temperature value.
     *
     */
    QStringList buffer_split = serialBuffer.split(","); //  split the serialBuffer string, parsing with ',' as the separator

    //  Check to see if there less than 3 tokens in buffer_split.
    //  If there are at least 3 then this means there were 2 commas,
    //  means there is a parsed voltage value as the second token (between 2 commas)
    if(buffer_split.length() < 3){
        // no parsed value yet so continue accumulating bytes from serial in the buffer.
        serialData = arduino->readAll();
        serialBuffer = serialBuffer + QString::fromStdString(serialData.toStdString());
        serialData.clear();
    }else{
        // the second element of buffer_split is parsed correctly, update the voltage value on lcdNumber
        serialBuffer = "";
        //qDebug() << buffer_split << "\n";
        parsed_data = buffer_split[1];

        qDebug() << "Voltage: " << parsed_data << "\n";
        updateVoltage(parsed_data);
    }

}
```

*7. Slika : Provera identifikacije uređaja*

### 2.1.3 QCustomPlot

Hteli smo da QT aplikacija pokazuje vrednost u realnom vremenu, to je moguće sa mnogo vidžeta koji su ponuđeni, ali hteli smo i da stavljamo vrednost na graf u realnom vremenu. Za to najbolje rešenje smo našli u biblioteci QCustomPlot, koji po difoltu nije ugrađen [2]. Prilično je jednostavna za korišćenje i nudi pravljenje grafa koji se kreće, a na sajtu QCustomPlota se nalazi primer kod kako to da se realizuje u ovoj bibilioteci [3].

U konstruktoru smo napravili timer, pa zatim dodali linije, podesili ose i labele. Napravili smo funkciju realtimeDataSlot u klasi MainWindow, koja meri vreme i kada prođe određeno vreme plotuje podatak na graf. Ubačena je još funkcija da reskalira osu i na kraju da osveži graf. Na kraju u konstruktoru povezali smo timer i funkciju realtimeDataSlot i timer smo podesili na 100ms [*8.Slika*].

```
void MainWindow::realtimeDataSlot(){

    static QTime time(QTime::currentTime());
    double key = time.elapsed()/1000.0;
    static double lastPointKey = 0;
    voltageValue = parsed_data.toInt();
    //qDebug( "the key before if is: %f", key );

    if(key - lastPointKey > 0.002)
    {
        ui->customPlot->graph(0)->addData(key,voltageValue);
        lastPointKey = key;
    }

    /* make key axis range scroll right with the data at a constant range of 8. */
    ui->customPlot->graph(0)->rescaleValueAxis();
    ui->customPlot->xAxis->setRange(key, 8, Qt::AlignRight);
    ui->customPlot->replot();
   // qDebug( "the key after if is : %f", key );
}
```

*8. Slika : Provera identifikacije uređaja*

## 2.2 ESP8266

Modul je sistem na čipu sa integrisanim TCP/IP protokolom i GPIO pinovima [4]. Izabrali smo kao dodatni IoT uređaj za očitavanje senzora, zato što je lak za programiranje i pored toga jako je popularan, što znači da postoji dobra korisnička podrška na internet. Koristi se u raznim embedded uređajima, data loggerima.

Prva faza implementacije je bio očitavanje senzora i sredjivanje podataka za "POST request". Post request je metoda za slanje podataka na server za kreiranje ili ažuriranje resursa [5]. Očitali smo vrednosti sa analognog ulaza A0. Napravili smo još 2 parametra da bi mogli testirati sva tri grafikona na platformi. String koji se šalje, sastoji se iz korisničkog imena i vrednosti svih promenljivih, kao što je pokazano dole *[9.Slika]*.

```
// Read the input on analog pin 0
int sensorValue = analogRead(A0);

// Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V)
float sensor1 = sensorValue * (5.0 / 1023.0);
float sensor2 = voltage + 2;
float sensor3 = voltage * 2;

// print out the value you read
//Serial.println(voltage);

// Priprema podatke za "HTTP POST request"
String httpRequestData = "user=" + username + "&value1=" + String(sensor1) +
    "&value2=" + String(sensor2) + "&value3=" + String(sensor3) + "";
Serial.print("httpRequestData: ");
Serial.println(httpRequestData);
```

*9.Slika : C/C++ kod za očitavanje senzora i srednjivanje izlaza*

Druga faza implementacije je bio inicijalizacija konekcije *[10.Slika]*. Da bi podaci bili uspešno poslati, prvo se definiše konekcija sa WiFi mrežom, zadaje se link ka php skripti koja vrši konekciju sa bazom podataka i na kraju zadaje se korisničko ime kao parametar za određivanje tabele u bazi podataka.

Treća faza je slanje zapakovanih podataka na server *[11.Slika]*.

```
// Konekcija sa WiFi mrežom
const char* ssid     = "yourNetwork";
const char* password = "yourPassword";

//Konecija sa serverom u slučaju localhost
//const char* serverName = "http://192.168.1.5/phpmyadmin/public_html/post_data.php";

//Konekcija sa serverom
const char* serverName = "http://sensortracker.000webhostapp.com/post_2.php";

//Zadavanje korisničkog imena
String username = "korisnickoIme";
```

*10.Slika : C/C++ kod za inicijalizaciju konekcije*

```
    // Šalje HTTP POST request
    int httpResponseCode = http.POST(httpRequestData);

    if (httpResponseCode>0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);
    }
    else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }
    // Free resources
    http.end();
     }
```

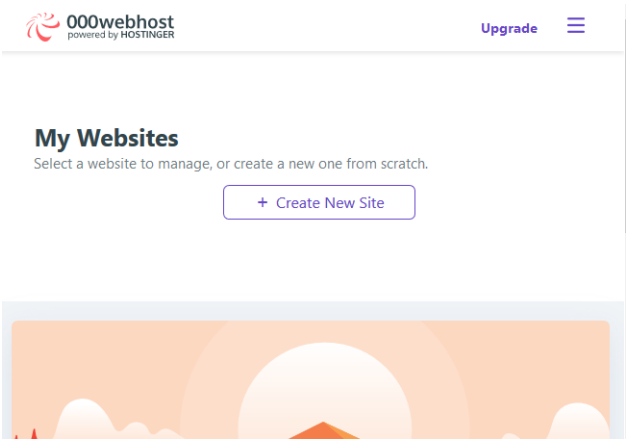
*11.Slika : C/C++ kod za slanje HTTP POST request*

# POGLAVLJE 3 – Platforma

Pri planiranju platforme, prvenstvena ideja je bila da napravimo localhost server. Ideju smo brzo odbacili zato što bi server bio pristupan samo sa lokalne mreže, što je problem pri projektovanju embedded sistema koji bi trebali da rade van lokalne mreže. Iz ovog razloga smo se opredelili pored web servera.

## 3.1 Hosting, domain i podizanje platform

Za hosting izabrali smo www.000webhost.com iz jednostavnih razloga. Prvo što je besplatan, a po drugo ima cPanel. Posle registracije, stranica nas odmah uputi na kreiranje sopstvenog sajta *[12.Slika]*.

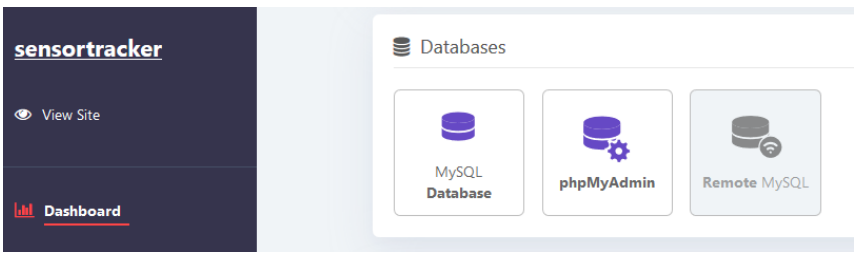



*12.Slika : Kreiranje nove stranice*

Pri kreiranju izaberemo naš domain ime. U slučaju da već imamo svoje, možemo to zadati, ali ako nemamo nije ni to problem, jednostavno ga kreiramo, s' tim da će imati produženje ".000webhostapp.com". Tako je dinamički deo našeg sajt postao http://sensortracker.000webhostapp.com. Statički deo, tzv. "landing page" smo napravili u Zyro editoru, koji se koriti "drag and place" metodom, što znači iz padajućeg menija izabere se panel i stavi se na sajt. Isto ima mogućnost kreiranja domain imena. U našem slučaju postao je http://sensortracker.zyrosite.com.

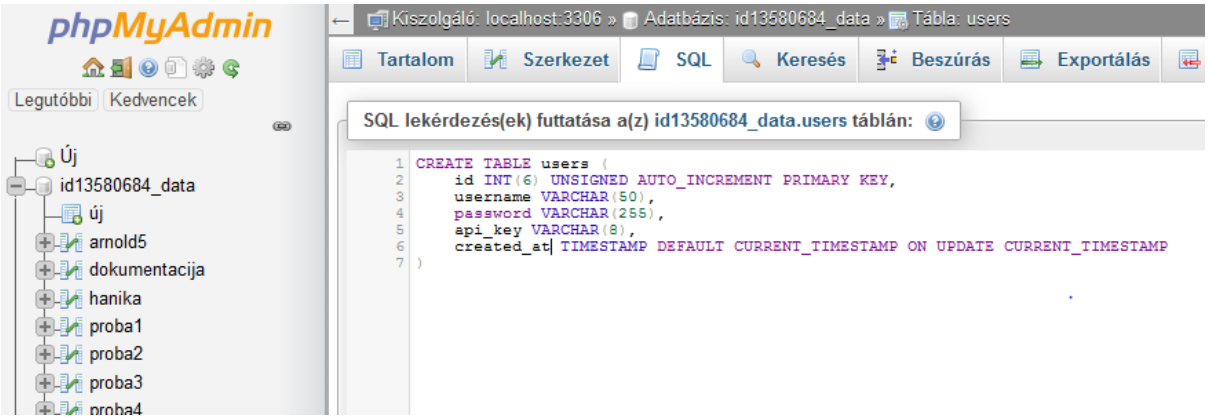
## 3.2 Baza podataka

Baza podataka je neophodna za skladišćenje podataka pročitanih sa senzora [6]. Za ovaj projekat koristili smo MYSQL bazu podataka. Da bi smo mogli koristiti prvo ga treba kreirati u cPanelu *[13.Slika]*.




*13.Slika : Kreiranje nove baze podataka*

Sledeće što ćemo uraditi je konfiguracija baze podataka *[14.Slika]*. Za rad nam treba tabela za korisničke naloge, gde će se upisivati nova korisnička imena, šifre, redni brojevi korisnika i osmocifren verifikacioni ključevi, tzv. "api_key"-ovi, koji će biti korišćeni u naprednim verzijama našeg sistema.




*14.Slika : Kreiranje table sa korisničkim podacima*

Otvaranje novih korisničkih tabela vršimo iz register.php, a postavljanje podataka očitanih sa senzora u bazu podataka vršimo iz post.php. O ovim metodama ćemo detaljnije u narednim poglavljama.

## 3.3 Korisnički sistem

Korisnički sistem sastoji se iz mehanizma za ulogovanje i glavne stranice za prikazivanje podataka. Realizovano je preko php filova register.php, login.php i welcome.php.

Register.php vrši registraciju novog korisnika. Uzeli smo gotov kod [7] i modifikovali ga tako da generiše tzv. "api_key" *[15.Slika]* i sa svakom regostracijom napravi novu tabelu u bazi podataka sa nazivom novoregistrovanog korisnika *[16.Slika]*. Tabela sadrži 5 polja, koja su "id" za dodavanje rednog broja unetog podatka, tri vrednosti za dodavanje senzora i polje koji beleži vreme dodavanja vrednosti.

/public_html/register.php

```
70      //PRAVIMO RANDOM STRING API_KEY
71      function generateRandomString($length = 10) {
72      return substr(str_shuffle(str_repeat($x
            ='0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ', ceil($length/strlen($x)) )),1
            ,$length);}
73      $api_key = generateRandomString(8);
74
75      // Prepare an insert statement
76      $sql = "INSERT INTO users (username, password, api_key) VALUES (?, ?, ?)";
```

*15.Slika : Kreiranje ključa „api_key"*

/public_html/register.php

```
110
111     // sql to create table
112     $sql = "CREATE TABLE $username (
113     id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
114     value1 VARCHAR(10),
115     value2 VARCHAR(10),
116     value3 VARCHAR(10),
117     reading_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
118     )";
119
120     if (mysqli_query($conn, $sql)) {
121         echo "Table MyGuests created successfully";
122     } else {
123         echo "Error creating table: " . mysqli_error($conn);
124     }
125
126     mysqli_close($conn);
127
128 }
129 ?>
```

*16.Slika : Kreiranje korisničke table*

Login.php vrši logovanje korisnika u svoj nalog omogućivši pristup stranici sa podacima o senzorima. Kod smo preuzeli i nismo modifikovali.

Welcome.php vrši prikazivanje podataka. Radi tako što uspostavi konekciju sa bazom podataka, preuzme podatke i prilagodi za rad *[17.Slika]*. Prikazivanje vrši html deo koda pomoću unapred definisanog grafikona, uzetog sa stranice https://code.highcharts.com/highcharts.js. Za tri grafikona imamo tri identična modula.

/public_html/welcome.php

```
20  $sql = "SELECT id, value1, value2, value3, reading_time FROM $user order by reading_time desc limit 40";
21
22  $result = $conn->query($sql);
23
24  while ($data = $result->fetch_assoc()){
25      $sensor_data[] = $data;
26  }
27
28  $readings_time = array_column($sensor_data, 'reading_time');
29
30  $value1 = json_encode(array_reverse(array_column($sensor_data, 'value1')), JSON_NUMERIC_CHECK);
31  $value2 = json_encode(array_reverse(array_column($sensor_data, 'value2')), JSON_NUMERIC_CHECK);
32  $value3 = json_encode(array_reverse(array_column($sensor_data, 'value3')), JSON_NUMERIC_CHECK);
33  $reading_time = json_encode(array_reverse($readings_time), JSON_NUMERIC_CHECK);
34
```

*17.Slika : Preuzimanje i modifikacija podataka*

/public_html/welcome.php

```
87
88  var chartT = new Highcharts.Chart({
89    chart:{ renderTo : 'chart-temperature' },
90    title: { text: 'Data from Sensor 1' },
91    series: [{
92      showInLegend: false,
93      data: value1
94    }],
95    plotOptions: {
96      line: { animation: false,
97        dataLabels: { enabled: true }
98      },
99      series: { color: '#059e8a' }
100   },
101   xAxis: {
102     type: 'datetime',
103     categories: reading_time
104   },
105   yAxis: {
106     title: { text: 'Values' }
107   },
108   credits: { enabled: false }
109 });
```

*18.Slika : Grafikon*

## 3.4 Postavljanje podataka u MYSQL bazu podataka

Postavljanje se vrši POST metodom, koji podržava zahtev HTTP protokola. Zahteva da se sa veb server prihvata podatak poslat u okviru HTTP zahteva. Često se koristi pri dostavljanju popunjenih veb formulara ili pri slanju datoteka .U našem slučaju, opisan je u php datoteci "post". Radi tako što razkomadi string poslat sa data uploadera. String je u obliku "user" + korisničko ime + "value1" + vrednost prvog senzora + "value2" + vrednost drugog senzora + "value3" + vrednost trećeg senzora. Delove stringa smešta u varijable. Posle uspostavljene veze sa bazom podataka, na osnovu korisničkog imena smesti vrednosti senzora u odgovarajuću tabelu *[19.Slika]*.

/public_html/post_2.php

```php
    $user = test_input($_POST["user"]);
    $value1 = test_input($_POST["value1"]);
    $value2 = test_input($_POST["value2"]);
    $value3 = test_input($_POST["value3"]);

    // Create connection
    $conn = new mysqli($servername, $username, $password, $dbname); //$servername, $username, $password, $dbname)
    // Check connection
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    $sql = "INSERT INTO $user (value1, value2, value3)
    VALUES ('" . $value1 . "', '" . $value2 . "', '" . $value3 . "')";

```

*19.Slika : Postavljanje vrednosti senzora u bazu podataka*

# POGLAVLJE 4 – Zaključak

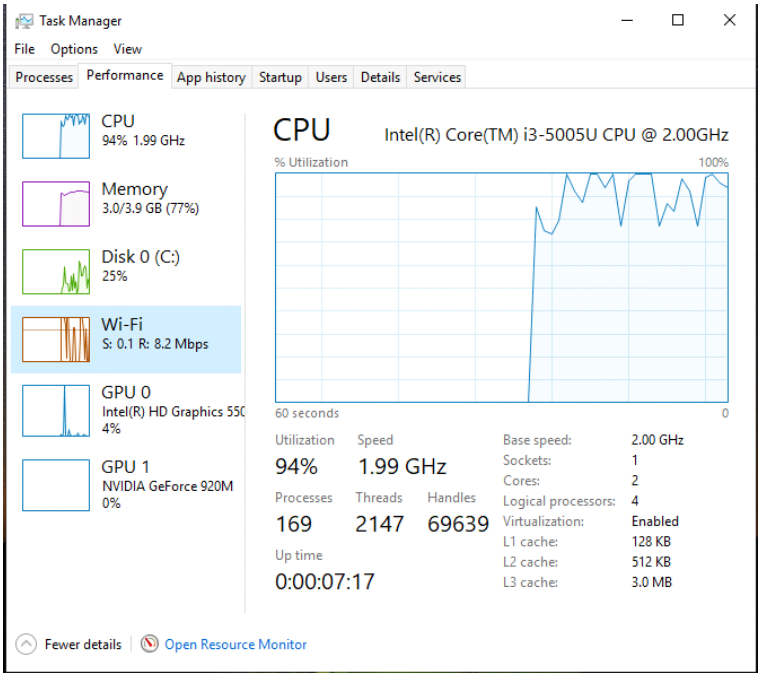
Ako se uzme u obzir da je nivo internetizacije u elektronskim uređajima u sve većem porastu, sve više mesta se stvara za projektovanje i monitoring uređaja, čijom upotrebom bi se razni procesi mogli optimizovati i učiniti efikasnijim.

Cilj ovog rada je bilo projektovanje sistema za prikupljanje, obradu i nadgledanje podataka, sa mogućnošću lakog povezivanja uređaja na njega. Akcenat je stavljen na projektovanje i realizaciju platforma, i sistema koji može da uspostavi konekciju s njim. Korišćeni senzori su upotrebljeni u svrhe testiranja i demonstracije. Rezultati testiranja su zadovoljavajuća.

Na osnovu svojih mogućnosti, može se zaključiti da ovaj uređaj ispunjava sve prethodno postavljene zahteve i da je uspešno realizovan, kako hardverski, tako i softverski. Rad na ovom projektu omogućio je autorima bliže upoznavanje sa konekcijom među računarskih mreža, IoT sistemima i sa projektovanjem embedded uređaja.

Jedan način poboljšanja bi bio kreiranje biblioteke, koji bi sadržao sve korišćene funkcije, i omogućio developerima korišćenje "SensorTracker" sistema, jednostavnim uključivanjem u izvorni kod. Druga ideja je dodavanje povratne sprege od platforma ka aplikacijama, koji omogućava interakciju sa uređajima. Treća ideja je proširivanje i dodavanje korisničkog interfejsa. Arduino uno ima još 5 analogna ulaza za senzore koje možemo uzeti kao dodatak našoj ploči. Za budući UI referencu smo uzeli od task managera sa windows 10 operativnog sistema *[20.Slika]*. Graf i osnovne informacije se pokazuju na ikonama, a kad kliknemo na nju pojavljuju se detalji i uvećan graf. Ovo je prilično pregledno i korisno u slučaju da imamo više različitih senzora. Naravno za neke stvari, npr. ako merimo sobnu temperaturu graf nam nije potrebna, samo trenutna temperatura, jer u normalnim uslovima sobna temperatura se menja jako sporo. Ali kao što smo radili sa svetlosnim senzorom tad brzo imamo promene pa zato ima smisla da se osmatra graf u realnom vremenu.

Uređaj je dobra polazna osnova za dalja buduća unapređenja, bilo u fakultetskom ili u komercijalnom smislu.




*20.Slika : Ideja korisničkog interfejsa*

## Literatura

[1] Raspberry Pi Foundation, "What is a Raspberry Pi" [Online]. Available: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi. [Accessed: 25-Maj-2020].

[2] Emanuel Eichhammer, "QCustomPlot" [Online]. Available: https://www.qcustomplot.com. [Accessed: 26-Maj-2020].

[3] Vinay Divakar, "Plot Real Time Graphs Qt Embedded Linux" [Online]. Available: https://github.com/deeplyembeddedWP/Plot-Real-Time-Graphs-Qt-Embedded-Linux. [Accessed: 26-Maj-2020].

[4] "the INTERNET of THINGS with ESP8266" [Online]. Available: http://esp8266.net. [Accessed: 20-Maj-2020].

[5] W3schools, "HTTP Request ethods" [Online]. Available: https://www.w3schools.com/tags/ref_httpmethods.asp. [Accessed: 22-Maj-2020].

[6] RandomNerdTutorials.com, "Visualize Your Sensor Readings from Anywhere in the World (ESP32/ESP8266 + MySQL + PHP)" [Online]. Available: https://randomnerdtutorials.com/visualize-esp32-esp8266-sensor-readings-from-anywhere. [Accessed: 23-Maj-2020].

[7] Tutorial Republic, "PHP MYSQL Login System" [Online]. Available: https://www.tutorialrepublic.com/php-tutorial/php-mysql-login-system.php. [Accessed: 23-Maj-2020].