etf+grb

**UNIVERZITET U ISTOČNOM SARAJEVU**

**ЕLEKTROTEHNIČKI FAKULTET**

**SEMINARSKI RAD**

**KONTROLERI I U/I UREĐAJI**

**Praćenje rada DHT22, NTC i DS18B20 senzora**

Student:

Milanović Darko

Mentor:

Prof.dr. Slobodan Lubura

*oktobar 2019. Istočno Sarajevo*

Sadržaj

[Uvod 3](#_Toc21986600)

[O Arduino-u 3](#_Toc21986601)

[Softver i programiranje 4](#_Toc21986602)

[Tema projekta i realizacija 5](#_Toc21986603)

[Ideja projekta 5](#_Toc21986604)

[Potrebni alati 5](#_Toc21986605)

[Šema i simulacija projekta 5](#_Toc21986606)

[O senzorima 7](#_Toc21986607)

[Funkcionalnost šeme 8](#_Toc21986608)

[Aplikacija za prikaz rezultata mjerenja 11](#_Toc21986609)

[Viruelna komunikacija simulacije projekta i aplikacije za prikaz 16](#_Toc21986610)

[Baza podataka 18](#_Toc21986611)

[Veb-stranica 20](#_Toc21986612)

[Zakljucak 25](#_Toc21986613)

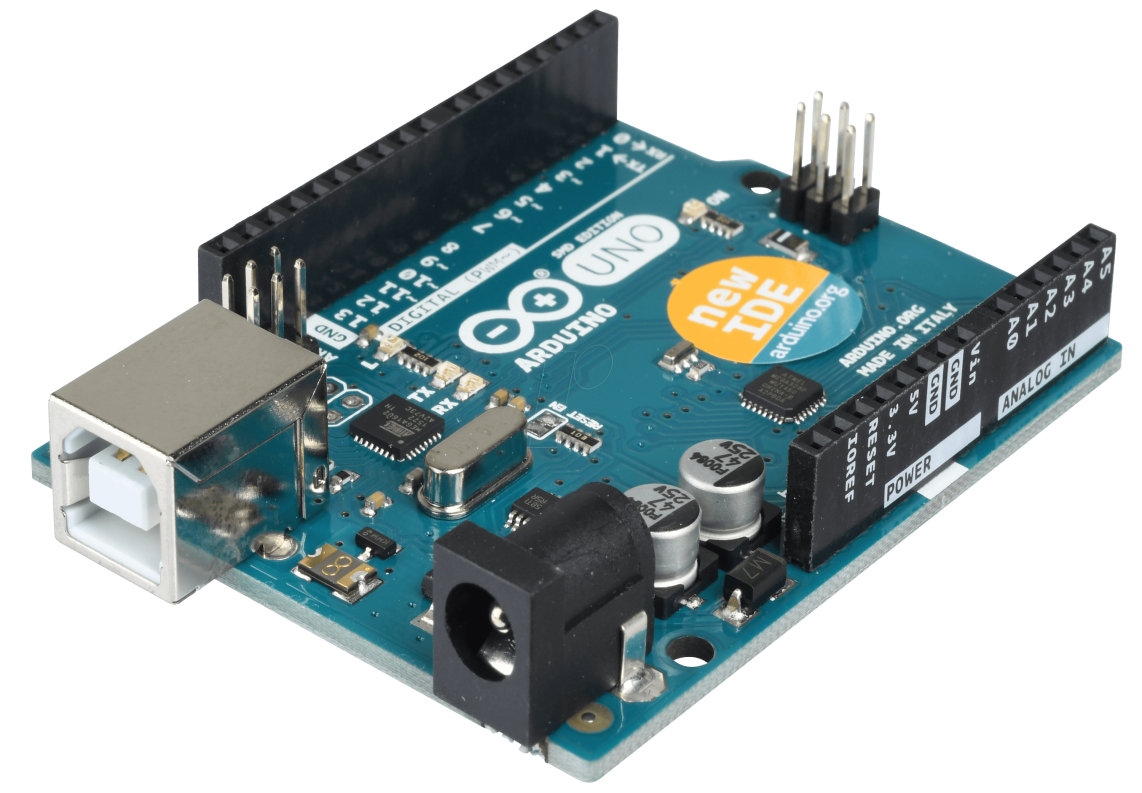
[Literatura 26](#_Toc21986614)

# Uvod

## O Arduino-u

**Arduino** je fizičko-računarska platforma (razvojni sistem) otvorenog koda. Hardver se sastoji od jednostavnog otvorenog hardverskog dizajna Arduino ploče sa Atmel AVR procesorom i pratećim ulazno-izlaznim elementima, tačnije, na sebi poseduje mikrokontroler. Softver se sastoji od razvojnog okruženja koje čine standardni kompajler i bootloader koji se nalazi na samoj ploči.

Arduino hardver se programira koristeći programski jezik zasnovan na Wiring jeziku (sintaksa i biblioteke). U osnovi je sličan C++ programskom jeziku sa izvesnim pojednostavljenjima i izmenama. Integrisano razvojno okruženje je zasnovano na Processing-u.



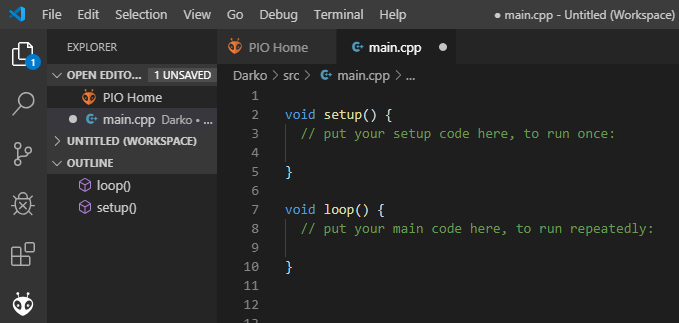
Slika 1. Arduino Uno

# Softver i programiranje

Arduino programi se pisu u C/C++ programskom jeziku, mada korisnici moraju da definišu samo dve funkcije kako bi napravili izvršni program. Te funkcije su:

* setup() – funkcija koja se izvršava jednom na početku i služi za početna podešavanja
* loop() – funkcija koja se izvršava u petlji sve vreme dok se ne isključi ploča

Pri tome, setup() predstavlja dio koda kojim se vrši podešavanje Arduino kontrolera (ulaza i izlaza, komunikacije sa računarom ili nekim drugim uređajem i slično)., dok je loop() dio koda koji Arduino stalno ponavlja.



Slika 2. Arduino code

# Tema projekta i realizacija

**Praćenje rada DHT22,NTC i DS18B20** senzora**, čuvanje dobijenih vrijednosti u bazi podataka te njihov prikaz na veb stranici.**

## Ideja projekta

Ideja projekta je demonstriranje kako korisnik u svakom trenutku sa bilo kog mjesta, ako je povezan na internet na svom računaru ili bilo kom uređaju, može imati pristup podacima temperature i vlažnosti koje želi da prati na nekom proizvoljnom mjestu.

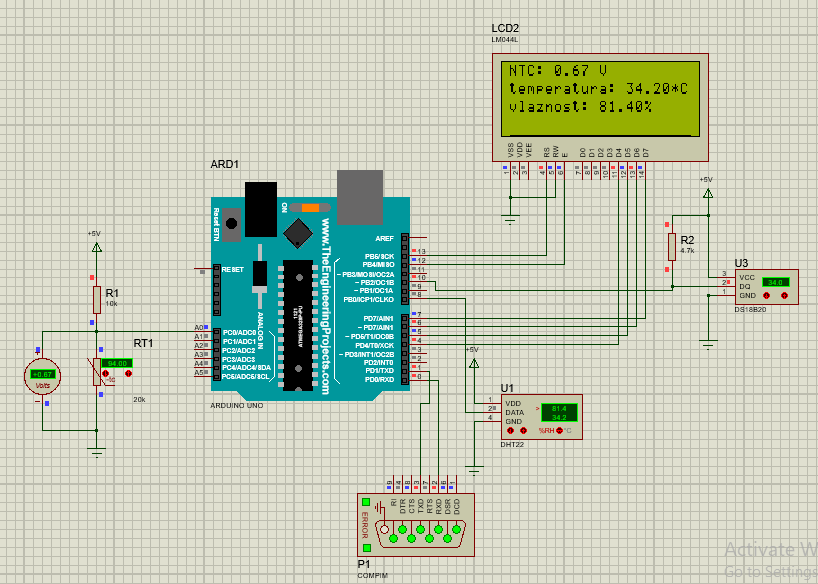
## Potrebni alati

U ovom projektu korišćeno je više softvera, kako za programiranje tako i za simulaciju. U projektu su korišćeni sljedeći softveri :

1. Proteus 8 (softver za crtanje šema i simulaciju)
2. Visual Studio Code (softver za programiranje Arduina PLATFORM IO)
3. Visual Studio 2013 (softver u kome je kreirana aplikacija u c#)
4. Free Virtual Serial Ports (virtuelna komunikaciju proteusa i aplikacije)
5. Visual Studio Code (softver za izradu veb stranice)
6. Wampserver 64 (simulacija servera kako bi podatke iz C# aplikacije smjestili u bazu podataka a kasnije te iste podatke koristili za prikaz na veb stranicu).

## Šema i simulacija projekta

Šema projekta je urađena u programu Proteus 8, a njena funkcionalnost odrađena u Visual Studio Code-u. U Visual Studio 2013 je urađena aplikacija u programskom jeziku C# (Windows Form application), a veza izmedju aplikacije u C# i mikrokontrolera je ostvarena preko Free Virtual Serial Ports.



Slika 3. Šema projekta

Simulacija u programu Proteus 8 prikazana je na slici 3.

Za šemu projekta korišćene su sljedeće komponente:

* ARDUINO UNO
* LCD display 20x4
* DHT22, senzor temperature i vlažnosti
* RES, kao otpornici potrebni za spajanje sa pojedinim komponentama
* COMPIM, kao virtuelni port
* POWER, kao napajanje
* NTC
* DS18B20
* GRUOND, kao uzemljenje

### O senzorima

**NTC**

NTC termistori (od engl. *Negative Temperature Coefficient*) imaju negativni koeficijent temperaturne promjene otpora, to jest kako se temperatura povećava tako se smanjuje električni otpor. Temperaturno im je područje od –50 do 150 °C, a primjenjuju se na primjer za stabilizaciju električnog napona (koristi se nelinearno naponsko-strujna karakteristika) i za mjerenje temperature u termometrima.

**DHT22**

Senzori koji se često koriste u praksi za mjerenje temperature i vlažnosti zraka su DHT11 i DHT22. Senzor DHT22 je jeftini senzor koji ima mogućnost mjerenja temperature u opsegu od 40 do +125 sa preciznošću od +0.5 stepeni i vlažnosti u opsegu od 0 do 100% sa preciznošću od 2-5%

**DS18B20**

DS18B20 digitalni temperaturni senzor ima 9-12 bitnu rezoluciju merenja temperature u Celzijusima i alarmnu funkciju koju korisnik može da podesi (i gornji i donji nivo). DS18B20 komunicira sa mikrokontrolerom kao masterom preko 1-Wire magistrale koja zahteva samo jednu liniju za prenos podataka (i gnd). Dodatno, DS18B20 ne mora da ima spoljašnje napajanje, već može da se napaja i preko linije za podatke ("parazitira").

Svaki DS18B20 ima jedinstven 64-bit serijski kod, koji omogućava korišćenje više senzora na istoj magistrali. Na taj način moguće je jednim procesorom upravljati sa puno senzora distribuirano na velikom prostoru. Primeri primene ovih senzora su: upravljanje grejanjem i klimatizacijom, monitoring temperature unutar zgrada, opreme, mašina i u upravljačkim i procesnim sistemima. Ključna svojstva:

• Temperaturni opseg je -55°C do +125°C

• Tačnost: ±0.5°C (u opsegu -10°C do +85°C)

• Rezolucija se može podesiti u opsegu od 9 -12 bita

• Nisu potrebne spoljašnje komponente

• U modu parazitnog napajanja je dovoljno 2 pina za rad (DQ i GND)

• Svaki senzor ima jedinstveni 64-bit serijski kod u internoj ROM

• Fleksibilno podešavanje alarma

Način spajanja i citanja podataka svakog od senzora se jasno vidi u kodu sa komentarima.

### Funkcionalnost šeme

Funkcionalnost šeme nacrtane u Proteus-u ostvaruje se C kodom prikazanim na kodnom listingu 1, korišćenjem programa Visual Studio Code.

#include <Arduino.h>

#include <LiquidCrystal.h>

#include <SimpleDHT.h>

#include <OneWire.h> //DS1820 biblioteka

#include <DallasTemperature.h> //DS1820 biblioteka

#define PINWIRE 10

OneWire oneWire(PINWIRE);

DallasTemperature sensors(&oneWire);

float tempC = 0;

void NTC(); // Funkcija u kojoj su pokupljeni podaci sa NTC i DHT

int pinDHT22 = 8;

SimpleDHT22 dht22(pinDHT22);

LiquidCrystal lcd(13, 12, 4, 5, 6, 7);

int ThermistorPin = A0;

int Vo;

float R1 = 10000;

float logR2, R2, T;

float c1 = 1.009249522e-03, c2 = 2.378405444e-04, c3 = 2.019202697e-07;

void setup () {

Serial.begin(9600);

lcd.begin(20,4); //Pokretanje lcd displeja

sensors.begin(); //Pokretanje DS1820 senzora

}

void loop () {

lcd.clear();

sensors.requestTemperatures(); //Nacin kupljenja vr. sa DS1820

tempC =sensors.getTempCByIndex(0);

lcd.setCursor(0,0);

lcd.print("C: ");

lcd.print(tempC);//Ispisivanje vr na lcd-u

lcd.print(" stepeni");

lcd.setCursor(0,1);

delay(2000);

lcd.clear();

NTC(); //NTC i DHT funkcija

Serial.println(tempC);

}

void NTC() //NTC i DHT22

{

float temperatura=0;

float vlaznost=0;

dht22.read2(&temperatura,&vlaznost,NULL);//Citanje podataka DHT

Vo = analogRead(ThermistorPin);

float Napon;

R2 = R1 \* (1023.0 / (float)Vo - 1.0);

logR2 = log(R2);

T = (1.0 / (c1 + c2\*logR2 + c3\*logR2\*logR2\*logR2));

T = T - 273.15;

T = (T \* 9.0)/ 5.0 + 32.0;

Napon=(float)Vo\*5/1023; //napon na krajevima NTC-a

lcd.setCursor(0,0);

lcd.print("NTC: ");

lcd.print(Napon);

lcd.print(" V");

lcd.setCursor(0,1);

lcd.print("temperatura: ");

lcd.print(temperatura);//Ispisivanje temp na lcd

lcd.print("\*C");

lcd.setCursor(0,2);

lcd.print("vlaznost: ");

lcd.print(vlaznost);//Ispisivanje vlaznosti na lcd

lcd.print("%");

delay(1000);

Serial.println(Napon);

Serial.println(temperatura);

Serial.println(vlaznost);

}

Kodni listing 1. Funkcionalnost šeme

Za realizaciju ovo projekta prilikom programiranja u Visual Studio Code-u bilo je potrebno koristili Platform IO i određene biblioteke. Biblioteke koje nisu instalirane u Visual Studio Code-u, instaliraju se praćenjem sjljedećih koraka:

1.Platform IO(Libraries)

2.Search libraries

3.Install

Potrebne biblioteke:

#include <Arduino.h>

#include <LiquidCrystal.h>

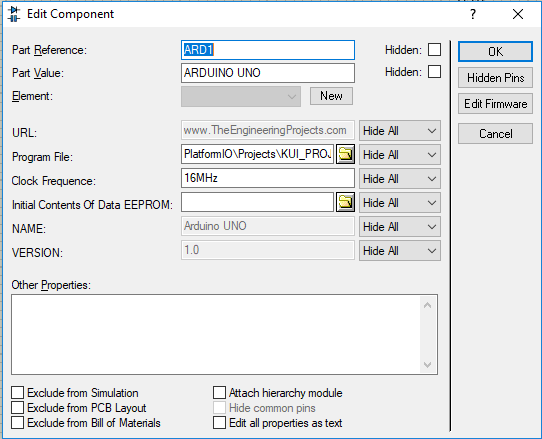
#include <SimpleDHT.h>

#include <OneWire.h>

#include <DallasTemperature.h>

Kodni listing 2. Potrebne biblioteke

Povezivanje Proteus-a sa Arduinom je vrlo jednostavno. Dvostrukim klikom na Arduino u Proteusu otvara se prozor prikazan na slici 4 :



Slika 4.Dodavanje koda napisanog u Visual Studio Code-u

Klikom na ikonicu Program File pronađe se .hex fajl projekta napisanog u VSCode-u, a potom se, klikom na dugme OK, zatvara prozor koji smo otvoren dvostrukim lijevim klikom na Arduino. Nakon slijeda ovih koraka, moguće je pokrenuti simulaciju.

## Aplikacija za prikaz rezultata mjerenja

Podatke prikazane na LCD displeju u Proteus-u trebalo je poslati u C# aplikaciju.

Na kodnom listingu 3 prikazan je kompletan C# kod aplikacije.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

using System.IO.Ports;

using System.Text.RegularExpressions;

namespace PROJEKAT\_KUI2019

{

public partial class KUI : Form

{

MySqlConnection konekcija = new MySqlConnection("Server=localhost;Database=mydatabase;Uid=root;Pwd=;");

int trenutni\_id = 0;

public KUI()

{

CheckForIllegalCrossThreadCalls = false;

InitializeComponent();

toolStripLabel1.Text = DateTime.Now.ToString();

try

{

serialPort1.Open();

}

catch

{

MessageBox.Show("greska");

}

}

private void s(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)

{

String napon =serialPort1.ReadLine();

String temperatura = serialPort1.ReadLine();

String vlaznost = serialPort1.ReadLine();

String tempc = serialPort1.ReadLine();

textBox1.Text = napon;

textBox2.Text = temperatura;

textBox3.Text = vlaznost;

textBox4.Text = tempc;

richTextBox1.AppendText(DateTime.Now.ToString()+"\n");

richTextBox1.AppendText(textBox1.Text);

richTextBox1.AppendText(textBox2.Text);

richTextBox1.AppendText(textBox3.Text);

richTextBox1.AppendText(textBox4.Text);

richTextBox1.AppendText("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

String datum = DateTime.Now.ToString();

float ntc = (float)Convert.ToDouble(napon);

float dht\_temperatura = (float)Convert.ToDouble(temperatura);

float dht\_vlaznost = (float)Convert.ToDouble(vlaznost);

float ds\_temperatura = (float)Convert.ToDouble(tempc);

progressBar1.Value = Convert.ToInt32(dht\_temperatura);

try

{

konekcija.Open();

MySqlCommand upit = konekcija.CreateCommand();

upit.CommandText = "INSERT INTO `mjerenja`(`ID`, `vrijeme\_datum`, `temperatura\_dht`, `vlaznost\_dht`, `ntc`, `temperatura\_ds`) VALUES ('" + trenutni\_id + "','" + datum + "','" + dht\_temperatura + "','" + dht\_vlaznost + "','" + ntc + "','" + ds\_temperatura + "');";

upit.ExecuteNonQuery();

}

catch(Exception ex)

{

MessageBox.Show(ex.Message,"Greska");

}

konekcija.Close();

}

private void timer1\_Tick(object sender, EventArgs e)

{

toolStripLabel1.Text = DateTime.Now.ToString();

}

private void button1\_Click(object sender, EventArgs e)

richTextBox1.Clear();

}

}

}

Kodni listing 3. Kod aplikacije

Takođe i pri izradi ove aplikacije bilo je potrebno uključiti određene biblioteke, većina njih je bila uključena po default-u, samim kreiranjem projekta. Na sledecu biblioteku potrebno je obratiti paznju, a to je:

**1. using MySql.Data.MySqlClient;**

Biblioteka služi da bi se omogućilo povezivanje aplikacije sa bazom podataka. Sve biblioteke koristene pri izradi aplikacije su:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

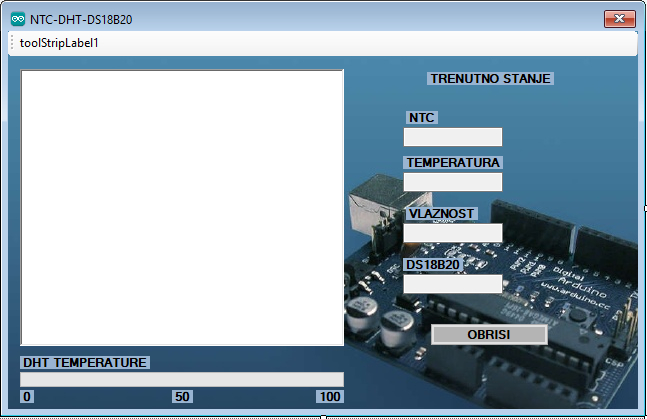
using MySql.Data.MySqlClient;

using System.Text.RegularExpressions;

Kodni listing 4. Potrebne biblioteke za izradu aplikacije

Izgled aplikacije je prikazan na slici 5. U lijevom boksu u odredjenom vremenskom intervalu se prikazuju rezultati sa senzora jedan ispod drugog, dok se u desnom dijelu aplikacije prikazuju trenutne vrijednosti izmjerene na senzorima.

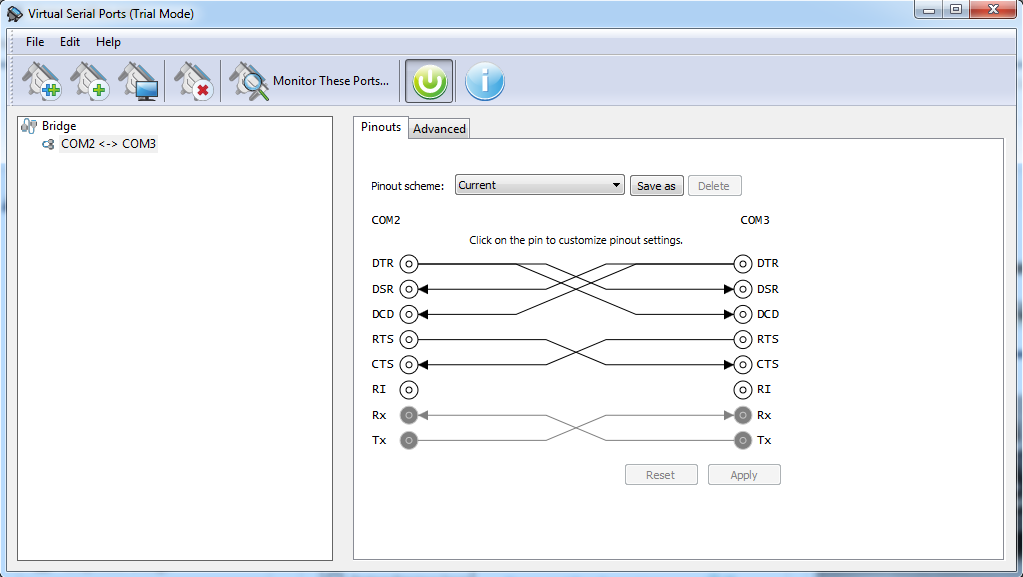
Preko progressbar-a je prikazana trenutna temperatura izmjerena na DHT22 senzoru u skali od 0 do 100.



Slika 5.Interfejs korisničke aplikacije

### Viruelna komunikacija simulacije projekta i aplikacije za prikaz

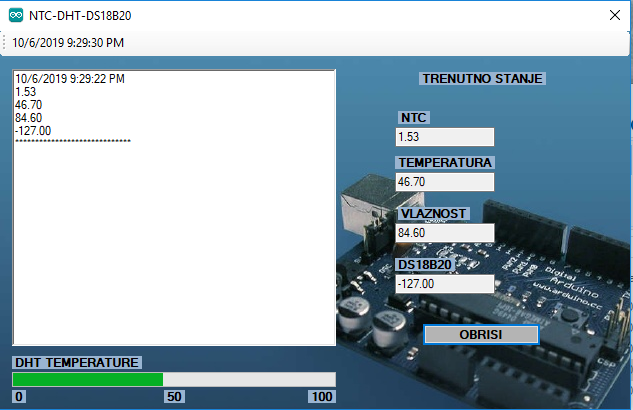
Za slanje podataka iz softvera u kome je rađena simulacija projekta u aplikaciju koristi se virtuelni serijski port (Free Virtual Serial Ports) na kome je potrebno izabrati odgovarajući par virtuelnih portova koji će međusobno komunicirati. U ovom slučaju, izabran je par COM2 i COM3(u aplikaciji COM1 i COM2). Prvi port predstavlja port sa koga se šalju podaci a drugi port predstavlja port za primanje podataka i on se unosi u aplikaciji. Ako je port pogrešan, korisnik dobija odgovarajuće obavještenje. Podešvanja portova detaljno su prikazana na slici 6.



Slika 6.Podešavanje Virtual Serial Porta

Nakod određenih podešavanja izvršenih u Virtual Serial Port i Visual Studio 2013 moguće je pokrenuti aplikacije kako bi međusobno komunicirale. Bildovanjem koda u VSCode-u i ubacivanjem .hex fajla u Protesu moguće je pokrenuti aplikaciju u C.

Na slici 7 prikazan je izgled aplikacije nakon primanja podataka iz Proteusa:



Slika 7.Interfejs aplikacije prilikom prilikom primanja podataka

## Baza podataka

Jedan od bitnijih dijelova ovog projekta je bio čuvanje podataka sa senzora koji su prikazivani u aplikaciji. Podaci se čuvaju u bazi podataka koristeći virtuelni Wampserver kao simulativni server. U kodu aplikacije u C# potrebno je ostvariti konekciju na bazu, kao i odgovarajućim upitom izvršiti slanje podataka u bazu. Dio koda koji to izvršava prikazan je na kodnom listingu 5.

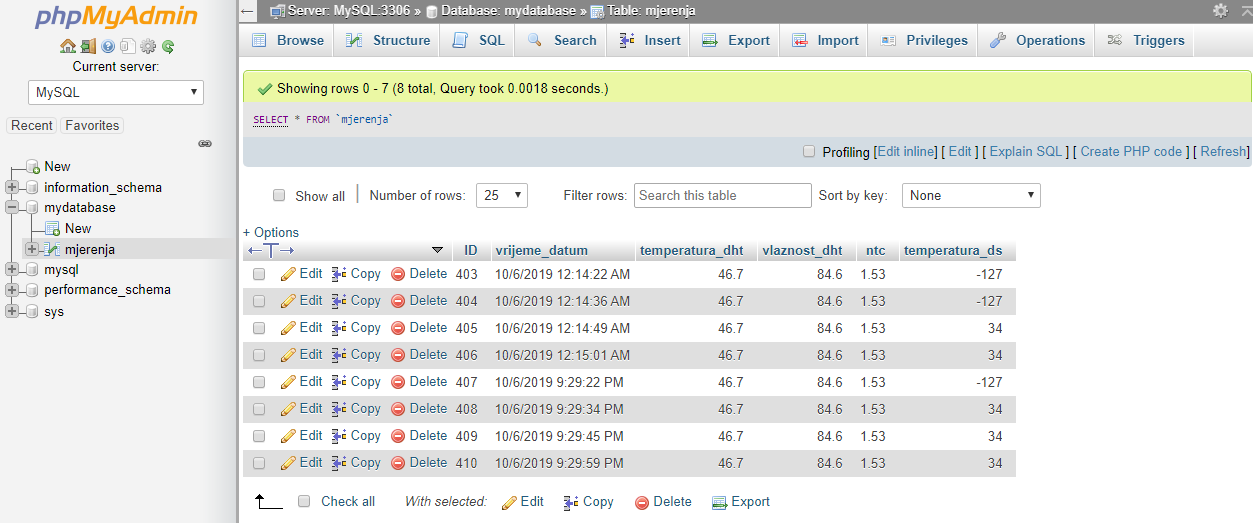
MySqlCommand upit = konekcija.CreateCommand();

upit.CommandText = "INSERT INTO `mjerenja`(`ID`, `vrijeme\_datum`, `temperatura\_dht`, `vlaznost\_dht`, `ntc`, `temperatura\_ds`) VALUES ('" + trenutni\_id + "','" + datum + "','" + dht\_temperatura + "','" + dht\_vlaznost + "','" + ntc + "','" + ds\_temperatura + "');";

upit.ExecuteNonQuery();

Kodni listing 5.Kreiranje konekcije i upita za slanje podataka

Pokretanjem C# aplikacije uspostavlja se konekcija na bazu, pri tome i prethodni softveri moraju biti pokrenuti. Ako u se u Proteus-u mijenjaju vrijednosti temperature i vlažnosti, nakon par odrađenih simulacija se u bazi podataka čuvaju izmjerene vrijednosti. Zbog nedovljno velike brzine slanja podataka, prilikom pokretanja aplikacije prvo mjerenje će biti prikazano na drugom mjestu, drugo na trećem i tako respektivno. Na sljedečoj slici je prikazan dio baze podataka na lokalnom serveru.



Slika 8.Prikaz baze podataka

U bazi podataka postoje mnoge mogućnosti rada kao što je pisanje i izvršavanje upita napisanih u SQL-u. Takođe, ako korisnik želi podatke smjestiti na prenosivi memorijski uređaj, moguće je preuzeti bazu na računar i dalje manipulisati sa rezultatima mjerenja. Takođe bitno je napomenuti da u bazi postoji opcija štampanja pa korisnik ima i tu mogućnost ukoliko želi pismeni prikaz podataka.

## Veb-stranica

Podatke iz baze je potrebno prikazivati na veb-stranici. Na sljedećem kodnom listingu prikazaćemo kod napisan u HTML-u, CSS-u, PHP-u i JavaScript programskom jeziku.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Title</title>

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

<style>

body {

background-image: url("kar1.jpg");

}

#myTable {

border-collapse: collapse;

border: 5px solid blue;

overflow: auto;

margin: auto;

width: 75%;

}

.klasa {

font-size: 18pt;

color: yellow;

}

tr {

height: 37px;

text-align: left;

background-color: blue;

border: 1px;

font-size: 15pt;

color: blue;

}

tr td {

text-align: center;

}

th {

text-align: left;

background-color: blue;

border: 1px;

}

tr:nth-child(even) {

background-color: white;

color: black;

}

tr:nth-child(odd) {

background-color: white;

color: black;

}

tr:hover {

background-color: yellow;

color: black;

}

#dugme {

text-align: center;

margin: auto;

margin-left: 45%;

}

#dugme > hover {

background-color: red;

color: #FFFFFF;

}

#naslov {

font-family: Arial;

font-size: 25pt;

margin: auto;

text-align: center;

color: blue;

background-color: white;

}

.chart {

margin-left: 350px;

width: 75%;

}

</style>

<form>

<p id="naslov" class="w3-animate-left">Prikazivanje vrijednosti sa senzora DHT NTC DS18B20</p>

<br>

<button type="button" class="btn btn-info" id="dugme" onclick="Osvjezi\_tabelu()">Osvježi</button>

<br>

<br>

</form>

</head>

<body>

<?php

$konekcija= new mysqli('localhost', 'root', '', 'mydatabase');

mysqli\_set\_charset($konekcija, 'UTF8');

if($konekcija->connect\_error)

{

echo "Nije uspjela konekcija".$konekcija->connect\_error;

}

else{

echo "";

}

$upit= 'SELECT \* FROM mjerenja ORDER BY 'ID' DESC ';

$ispistabele='<table class="w3-animate-top" id="myTable">

<tr>

<td>Redni broj</td>

<td>Datum</td>

<td>Temperatura</td>

<td>Vlaznost</td>

<td>NTC</td>

<td>Temperatura\_DS</td>

</tr>';

$result=$konekcija->query($upit);

if($result->num\_rows>0)

{

while ($red= $result->fetch\_assoc())

{

$ispistabele.= '

<tr id="red">

<td>'.$red['ID'].'</td>

<td>'.$red['vrijeme\_datum'].'</td>

<td>'.$red['temperatura\_dht'].'ºC</td>

<td>'.$red['vlaznost\_dht'].'%</td>

<td>'.$red['ntc'].'V</td>

<td>'.$red['temperatura\_ds'].'ºC</td>

</tr>';

}

$ispistabele.='

</table>';

echo $ispistabele;

}

?>

<br>

<br>

<div class="chart" id="chart\_div" style="width: 400px; height: 120px; "></div>

</body>

<script type="text/javascript">

function Osvjezi\_tabelu() {

window.location.reload();

}

google.charts.load('current', { 'packages': ['gauge'] });

google.charts.setOnLoadCallback(drawChart);

var x = document.getElementById("myTable").rows[1].cells[2].innerHTML;

var y = document.getElementById("myTable").rows[1].cells[3].innerHTML;

var b = parseFloat(x);

var a = parseFloat(y);

console.log(a);

// var vlaznost = document..getElementById("myTable").rows[0].cells.length;

function drawChart() {

var data = google.visualization.arrayToDataTable([

['Label', 'Value'],

['temperatura', b],

['vlažnost', a]

]);

var options = {

width: 600, height: 300,

greenFrom: 0, greenTo: 50,

redFrom: 90, redTo: 100,

yellowFrom: 50, yellowTo: 90,

minorTicks: 5

};

var chart = new google.visualization.Gauge(document.getElementById('chart\_div'));

chart.draw(data, options);

}

</script>

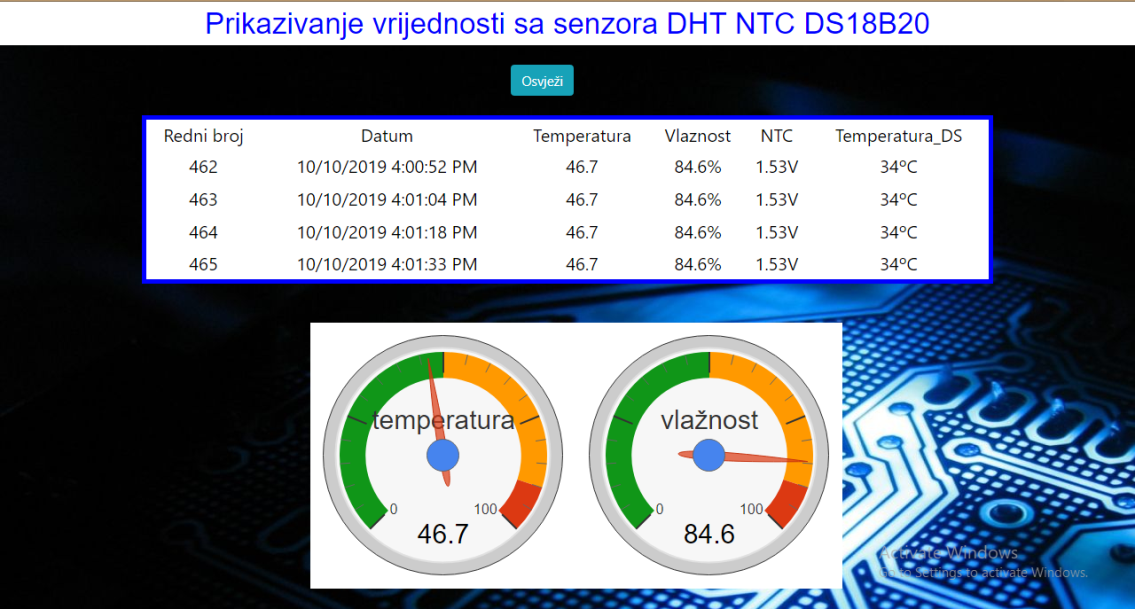
</html>

Kodni listing 6.Kod za izradu veb stranice

Veb stranica je tako uradjena da dinamčki prikazuje vrijednosti koje joj se prosleđuju iz baze. Klikom na dugme OSVJEZI korisniku se osim prethodnih mjerenja prikazuju u dodatna mjerenja koja su u međuvremenu pristigla u bazu. To je urađeno u JavaScript programskom jeziku pomoću funkcije koja osvježava prikaz veb stranice koja je prikazana u narednom dijelu.

<**script**>  
 **function***Osvjezi\_tabelu*(){  
 **window**.**location**.reload();  
 }  
</**script**>

Na slici 9 prikazan je izgled veb-stanice sa podacima dobijenim iz baze.



Slika 9.Izgled veb-stranice

# Zakljucak

Ovim radom objašnjena je i realizovana izrada projekta sa veb stranicom na kojoj se prikazuju rezultati mjerenja DHT, NTC i DS1820 senzora, što je ujedno i bio cilj ovog rada. Detaljno su objašnjene pojedine komponente, simulacija u proteusu, izrada aplikacije u c#, kod u Visal Studio Code-u, kod uVisual Studio 2013, HTML, CSS i Java Script kod, kao i pravljenje baze podataka(PhpMyAdmin). Kroz izradu ovog projekta analizirane su tehnologije koje se danas dosta koriste i za koje se vjeruje da će nadalje biti sve više korištene.

# Literatura

1. [www.theengineeringprojects.com](http://www.theengineeringprojects.com)
2. <https://sr.wikipedia.org/wiki/Arduino>
3. <http://www.elektronika.ftn.uns.ac.rs>
4. Slobodan Lubura i Nikola Kukrić. *Praktikum za izvodjenje vježbi iz Kontrolera i U/I uredjaja*.Istočno Sarajevo: Elektrotehnički Fakultet u Istočnom Sarajevu, 2017.
5. <https://www.w3schools.com/>