



Šolski center Celje

Srednja šola za kemijo, elektrotehniko in računalništvo

SPLETNA STRAN

Zaključna naloga

Avtorji:

Žiga Pušnik
Matic Peternel
Žiga Skaza

Mentor:

Mojmir KLOVAR univ. dipl. inž.

Celje, 2012

Kazalo

1	POVZETEK	2
2	UVOD	3
3	SPLETNA STRAN	4
3.1	Branje podatkov	4
3.1.1	Branje podatkov iz binarne .wlc datoteke	4
3.1.2	Branje podatkov s COM portom	5
3.1.3	Branje podatkov iz .txt in .html tekstovnih datotek	9
3.2	Prikazovanje podatkov	11
3.2.1	Prikazovanje trenutnih podatkov	11
3.2.2	Prikazovanje statistike in generiranje grafikonov	13
3.2.3	Smer vetra ponazorjena z kompasom	17
3.3	Podatkovna baza	18
3.4	Oblika in CSS	19
4	NADGRADNJA IN BODOČI CILJI	20
5	RAZPRAVA	21
6	ZAKLJUČEK	22
7	VIRI IN LITERATURA	23
8	ZAHVALA	24
9	KAZALO SLIK	25
10	PRILOGE	26

1 POVZETEK

Ljudje večkrat potrebujemo različne informacije. Da pa bi nam bile te informacije na voljo takrat, ko jih potrebujemo, vse podatke zbiramo, hranimo ter čim bolj organiziramo. Te podatke torej zberemo v bazi podatkov. Podatke torej imamo, vendar potrebujemo za predstavitev želene informacije še pripomočke in metode s katerimi bomo to informacijo obdelali.

V tej raziskovalni nalogi smo raziskovali kako na najboljši in najlažji način preberemo podatke iz vremenske postaje ter kako te informacije predelati oz. predstaviti tako, da bodo razumljivi ljudem, ki se ne spoznajo na meteorologijo.

Dlje ko smo raziskovali več načinov branja podatkov smo odkrili, a smo se na koncu odločili le za eno in sicer za branje podatkov iztxt in html tekstovnih datotek. Ugotavljali smo tudi kako te podatke pridobljene iz vremenske postaje predstaviti v spletni aplikaciji.

2 UVOD

Pridobivanje informacij je v informacijski družbi ključnega pomena, saj brez informacij ni podatkov in brez podatkov ni znanja. Če pa imamo informacije oz. podatke, ki so prikazani na takšen način, ki jih ne razumemo, je slabše kot to, da jih sploh ne bi imeli. Nekateri podatki oz. informacije nam brez predznanja nič ne pomenijo in to nas lahko moti, zato bi bilo potrebno vse podatke prikazovati na človeku prijazen način. Podatki s katerimi se še nikoli nismo srečali bi bilo potrebno razložiti in prikazovati na takšen način, da bi jih lahko razumeli ljudje z minimalnim predznanjem o tej temi.

Ljudje se pogovarjamo in družimo, ko gre za splošne teme se vsi več ali manj razumemo vendar, ko pa kdo začne govoriti o svoji stroki pa nas ostali iz druge stroke ne razumejo in zaradi tega lahko nastanejo problemi. Zamislimo si torej nekoga, ki nam začne govoriti o temi, ki je ne razumemo in nimamo predznanja o tem in se zato počutimo neprijetno. Če pa nam bi razložili podatke tako, da bi jih razumeli brez nekakšnega predznanja bi bilo vse lažje.

Ko smo v šoli izvedeli, da je na šolski strehi postavljena vremenska postaja smo se z mentorjem pogovorili kaj bi lahko naredili s podatki, ki nam jih ponuja ta vremenska postaja. Zato smo se s skupnimi močmi odločili, da bo naša skupina naredila spletno aplikacijo oz. spletno stran, ki bo prikazovala meteorološke podatke na človeku prijazen način, tako, da jih bodo razumeli vsi, mladi in tudi malo starejši, kot tudi ljudje z minimalnim znanjem o meteorologiji. Da pa bi lahko prikazovali meteorološke podatke tako, da bi jih vsi razumeli smo jih seveda morali razumeti mi, ki bomo to spletno stran postavili. Morali smo se naučiti, kaj vse potrebujemo za prikazovanje teh podatkov, da bodo podatki kar se da točni in da jih bodo razumeli vsi.

3 SPLETNA STRAN

3.1 Branje podatkov

V vsakdanjem jeziku pogosta izraza podatek in informacija predstavljata sopomenki, se pravi da ju uporabljamo za enak pomen. V informatiki pa je med njima bistvena razlika. Podatek v informatiki pomeni obdelovalna surovina. Iz podatkov v postopku obdelave dobimo informacijo. Podatki so kodirani, to pomeni da so zapisani s kodo. V sodobnih digitalnih napravah so podatki kodirani dvojiško oziroma binarno.

Če nam podatek v razumljivi obliki sporočijo in se iz tega nekaj novega naučimo smo prejeli informacijo. Informacija torej pomeni prirastek znanja. Iz enega podatka pa lahko dobimo različne informacije, odvisno pač od prejemnikovega predznanja.

Pred izdelavo spletne strani, smo se morali odločiti, kako bi brali podatke. Možnosti za branje podatkov je veliko.

3.1.1 Branje podatkov iz binarne .wllk datoteke

Za prvi način branja podatkov smo uporabili binarno datoteko Weatherlink. Za branje podatkov iz te datoteke smo napisali program, ki bi bral binarne podatke. Binarna datoteka WeatherLinka je razdeljena na več delov. Preden pridemo do potrebnih podatkov, kot so zapisi o temperaturi, vetru in ostalo, je potrebno izpustiti kar precej bitov. Prvih 18 bitov sestavlja naziv in verzija. Sledita dva dnevna zapisa, ki sta oba velika po 88 bitov in šele nato pridemo do potrebnih podatkov. Vse te potrebne korake smo zapisali v Microsoft Visual Studio v programskem jeziku C#.

Vendar pa takšen pristop ni bil primeren in je tudi zelo okoren, saj se datoteka posodobi vsako uro. Za vsak mesec se ustvari nova datoteka z imenom meseca in leta. Potrebno pa bi bilo razviti zelo zapleten algoritem, ki bi datoteko sproti dekodiral, shranjeval zapise, se pomikal med datotekami in računal povprečje nekaterih podatkov.

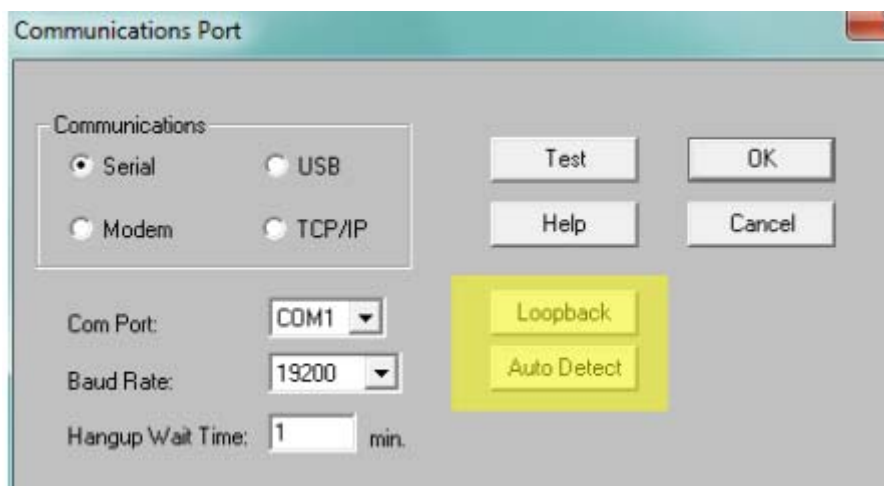
Z našim programom smo uspeli pridobiti le ključne podatke, kot so temperatura, pritisk, rosišče in podobno. Za pravilno delovanje in funkcionalnost pa to ni dovolj. Zato smo raziskali še druge načine branja podatkov. Ena izmed njih je COM port, ki je predstavljen v naslednjem poglavju.

Date	Time	Temp Out	Hi Temp	Low Temp	Out Hum	Dew Pt.	Wind Speed	Wind Dir	Wind Run	Hi Speed
8.12.11	11:57	10.7	10.7	10.7	37	-3.4	1.8	NW	0.11	3.6
8.12.11	11:58	10.7	10.7	10.7	37	-3.4	2.7	NW	0.16	3.6
8.12.11	11:59	10.7	10.7	10.7	38	-3.0	1.8	W	0.11	4.0
8.12.11	12:00	10.7	10.7	10.7	38	-3.0	2.2	NW	0.13	4.0
8.12.11	12:01	10.7	10.7	10.7	38	-3.0	1.8	NW	0.11	3.6
8.12.11	12:02	10.8	10.8	10.7	39	-2.6	1.3	NW	0.08	2.7
8.12.11	12:03	10.8	10.8	10.8	38	-3.0	2.7	NNE	0.16	4.5
8.12.11	12:04	10.8	10.8	10.8	37	-3.4	2.2	NNE	0.13	3.1
8.12.11	12:05	10.8	10.8	10.8	37	-3.4	2.7	NNW	0.16	4.9
8.12.11	12:06	10.8	10.8	10.8	37	-3.4	1.3	NNW	0.08	3.6
8.12.11	12:07	10.8	10.8	10.8	37	-3.4	1.8	N	0.11	3.6
8.12.11	12:08	10.8	10.8	10.8	37	-3.4	1.8	W	0.11	3.1
8.12.11	12:09	10.8	10.8	10.8	38	-3.0	1.3	NW	0.08	2.2
8.12.11	12:10	10.8	10.8	10.8	39	-2.6	1.3	N	0.08	2.2
8.12.11	12:11	10.8	10.8	10.8	39	-2.6	1.3	NNE	0.08	2.2
8.12.11	12:12	10.8	10.8	10.8	39	-2.6	1.8	NNW	0.11	2.7
8.12.11	12:13	10.9	10.9	10.9	38	-2.9	2.7	N	0.16	4.0
8.12.11	12:14	10.9	10.9	10.9	38	-2.9	2.7	N	0.16	4.5
8.12.11	12:15	10.9	10.9	10.9	38	-2.9	1.8	NNW	0.11	4.0
8.12.11	12:16	10.9	10.9	10.9	38	-2.9	1.3	ENE	0.08	2.7
8.12.11	12:17	10.9	10.9	10.9	39	-2.5	0.9	NE	0.05	1.8
8.12.11	12:18	10.9	10.9	10.9	39	-2.5	1.8	NE	0.11	3.1
8.12.11	12:19	10.9	10.9	10.9	39	-2.5	1.8	NE	0.11	3.6
8.12.11	12:20	10.9	10.9	10.9	39	-2.5	1.8	E	0.11	2.7

Slika 1: Datoteka WetherLink s končnico .wlk

3.1.2 Branje podatkov s COMportom

Že prej omenjeni program WeatherLink bere potrebne podatke direktno iz COMporta, zato smo analizirali tudi možnost takšnega načina pridobivanja podatkov. Najprej je bilo potrebno analizirati delovanje prenosa, podatkovni niz, ki pa je velik 95 bajtov. Nato smo morali ugotoviti še ostale nastavitve prenosa podatkov, kot je hitrost prenosa in kdaj se predpomnilnik porta počisti, če sploh se. Pomagali smo si z originalnim programom WeatherLink. Na srečo smo našli originalne nastavitve v čarovniku in tako pridobili vse potrebne informacije za izdelavo konzolne aplikacije, ki bi potrebne podatke brala vsako minuto in bi jih shranjevala v tekstovno datoteko.



Slika 2: Nastavitve COM porta

```

static SerialPort Open_Serial_Port()
{
    try
    {
        SerialPort thePort = new SerialPort("COM1", 19200, Parity.None, 8, StopBits.One);

        thePort.ErrorReceived += new SerialErrorReceivedEventHandler(SerialPort_ErrorReceived);

        thePort.ReadTimeout = 2500;
        thePort.WriteTimeout = 2500;

        thePort.DtrEnable = true;

        thePort.Open();

        return (thePort);
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.ToString());
        return (null);
    }
}

```

Slika 3: Ukazi za čiščenje predpomnilnika

Sprogramirati je bilo potrebno tudi ostale nastavitve, kot je izpraznitev predpomnilnika, ki se mora počistiti potem ko zahtevamo podatkovni niz. To nam je uspelo z ukazoma počisti vhodni in izhodni predpomnilnik.

```

thePort.DiscardInBuffer();
thePort.DiscardOutBuffer();

```

Slika 4: Ukazi za čiščenje predpomnilnika

Ob prejetju podatkovnega niza pa smo podatke morali nekako pretvoriti in shraniti. Ker smo že dekodirali .wtk format datoteke, nam ta del ni povzročal preglavic, saj je zaporedje podatkov skoraj identičen. Nekaj težav pa smo imeli z pretvarjanjem ameriških enot, kot so milje na uro, količina dežja v inčih, temperatura v fahrenheitih in podobno. Vsem je znano, da se pri pretvorbi merskih enot izgubi natančnost, rezultat pa so velikokrat dolga decimalna števila, ki pa jih je potrebno zaokrožiti. Mi smo podatke zaokrožili na dve decimalni vejici.

```

int hours,
    minutes;
string timeString;
DateTime currTime;

barTrend = Convert.ToInt32((sbyte)loopByteArray[3]);
barometer = (float)(BitConverter.ToInt16(loopByteArray, 7)) / 1000 * 2.54f;
insideTemp = (((float)(BitConverter.ToInt16(loopByteArray, 9)) / 10) - 32) * 5/9;
insideHumidity = Convert.ToInt32(loopByteArray[11]);
outsideTemp = (((float)(BitConverter.ToInt16(loopByteArray, 12)) / 10) - 32) * 5/9;
outsideHumidity = Convert.ToInt32(loopByteArray[33]);
windDirection = BitConverter.ToInt16(loopByteArray, 16);
currWindSpeed = ((float)Convert.ToInt32(loopByteArray[14]))/10 * 1.609344f;
avgWindSpeed = ((float)Convert.ToInt32(loopByteArray[15]))/10 * 1.609344f;
dayRain = (float)(BitConverter.ToInt16(loopByteArray, 50)) / 100 * 2.54f;

currTime = DateTime.Now;
receiveDate = DateTime.Now;

```

Slika 5: Shranjevanje in pretvorba podatkov

Ko smo program zapisali, smo ga morali še testirati. Konzolna aplikacija je pravilno delovala že v prvem poskusu, kar je rezultat dobre analize načina prenosa podatkov.

Program bi nato tudi uporabili in ga razvijali, dokler ne bi bil popolnoma funkcionalen in uporaben, vendar smo naleteli na težavo, katere pravzaprav nismo pričakovali. Do COM porta lahko dostopa samo en proces naenkrat, to je najverjetneje narejeno iz varnostnih razlogov. Se pravi, če smo želeli brati z našo aplikacijo, potem WeatherLink ne bi smel biti zagnan.

```

C:\Documents and Settings\Administrator\Desktop\bin\Debug\COM.exe
Sončni zahod: 11:36
Datum: 17.3.2012 0:00:00, čas: 11:36:00.8437500
System.UnauthorizedAccessException: Access to the port 'COM1' is denied.
   at System.IO.Ports.InternalResources.WinIOError(Int32 errorCode, String str)
   at System.IO.Ports.SerialStream..ctor(String portName, Int32 baudRate, Parity
parity, Int32 dataBits, StopBits stopBits, Int32 readTimeout, Int32 writeTimeou
t, Handshake handshake, Boolean dtrEnable, Boolean rtsEnable, Boolean discardNul
l, Byte parityReplace)
   at System.IO.Ports.SerialPort.Open()
   at COM.Program.Open_Serial_Port() in C:\4. izpitna enota\COM\COM\Program.cs:l
ine 184
Barometer: 0,00cm Hg Umirejeno
Notranja temperatura: 0
Notranja vlažnost: 0%
Zunanja temperatura: 0
Zunanja vlažnost: 0%
Smer vetra: Sever @ 0 stopinj
Trenutna hitrost vetra: 0km/h
Povprečna hitrost vetra v desetih minutah:0km/h
Dnevna količina dežja: 0cm
Sončni vzhod: 11:36
Sončni zahod: 11:36

```

Slika 6: Konzolna aplikacija z zaprtim programom WeatherLink

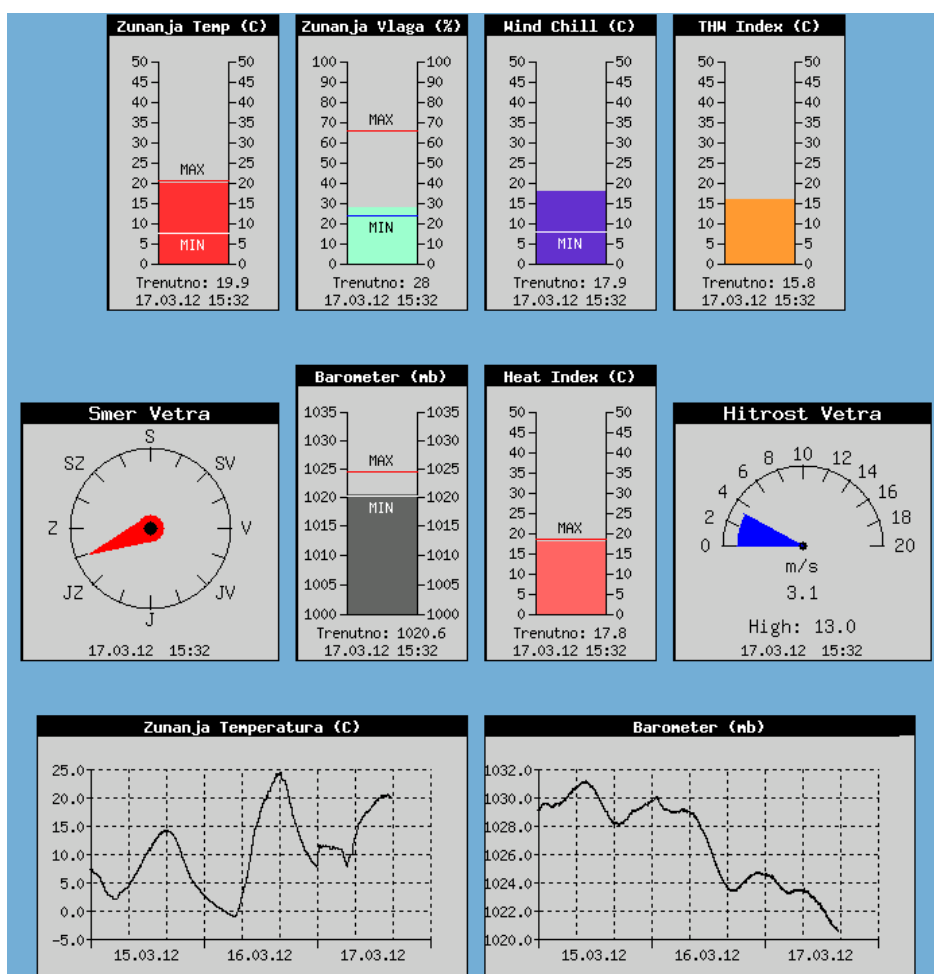

```

C:\Documents and Settings\Administrator\Desktop\bin\Debug\COM.exe
Branje podatkov iz com porta01
Datum: 17.3.2012 0:00:00, čas: 11:39:00
Barometer: 76.69cm Hg Pada počasi
Notranja temperatura: 28,72222
Notranja vlažnost: 18%
Zunanja temperatura: 18,27778
Zunanja vlažnost: 32%
Smernost vetra: Jug @ 190 stopinj
Trenutna hitrost vetra: 2,414016km/h
Povprečna hitrost vetra v desetih minutah:2,253082km/h
Dnevna količina dežja: 0cm
Sončni vzhod: 6:07
Sončni zahod: 18:08

```

Slika 7: Konzolska aplikacija z odprtim programom WeatherLink



Zato smo se zaradi prej omenjene težave dogovorili, da takšen način branja opustimo in raziščemo druge metode in pristope. K temu je pripomoglo dejstvo, da je WeatherLink zelo močno orodje, ki omogoča prikazovanje napredne statistike, s katero si pomagajo tudi meteorologi, naša spletna stran pa mora prikazovati podatke na uporabniku preprost način, ki takšnih in podobnih izpisov ne vsebuje. Na kratko povedano, škoda bi bilo zavreči WeatherLink, samo zaradi naše aplikacije, ker je dovolj potencialen, da bodo lahko z njegovo pomočjo to temo raziskovali še druge generacije za nami. Zato smo našli in uporabili nov pristop pri branju podatkov.



Slika 8: Prikaz napredne statistike v WeatherLinku

3.1.3 Branje podatkov iz .txt in .html tekstovnih datotek

WeatherLink ima takšno možnost konfiguracije, da generira in posodablja tekstovne datoteke z meteorološkimi podatki. Ustvarja lahko minutne ter dnevne zapise. Minutni zapisi se posodablajo, zato datoteka vsebuje samo eno vrstico, to je trenutno stanje, medtem ko večdnevni zapisi vsebujejo več tisoč vrstic minutnih zapisov za zadnjih osem dni.

 data.html	14.3.2012 12:43	Firefox HTML Doc...	1 KB
 downld08.txt	15.3.2012 17:33	Text Document	2.846 KB

Slika 9: Prikaz datotek v raziskovalcu

Datoteka data.html vsebuje samo trenuten zapis podatkov, velika prednost te datoteke pa je, da v .xml datoteki nastavimo kateri podatki naj se shranjujemo. Na ta način se močno poveča uporabnost in pa enostavnost programiranja, saj v datoteki ni nepotrebnih podatkov, ki splošnega uporabnika ne zanimajo. XML datoteka vsebuje niz, ki ga lahko poljubno spreminjamo. Ker pa data.html datoteko uporabljajo tudi druge skupine, smo se morali dogovoriti in sestaviti takšen niz, ki bo zadostoval vsem skupinam. Če bi na lastno pest spreminjali ta niz, bi morali spremeniti najprej kodo v naši aplikaciji, prav tako pa bi to morale storiti tudi druge skupine. Kako je ta niz sestavljen pa prikazuje besedilo spodaj.

```
<!--date-->;<!--time-->;<!--sunriseTime-->;<!--sunsetTime-->;<!--outsideTemp-->;<!--windChill-->;<!--hiOutsideTemp-->;<!--lowOutsideTemp-->;<!--outsideHumidity-->;<!--barometer-->;<!--barTrend-->;<!--windDir-->;<!--windDirection-->;<!--windSpeed-->;<!--wind10Avg-->;<!--rainRate-->;<!--solarRad-->;<!--uv-->
```

Slika spodaj prikazuje način vpisa podatkov v data.html datoteki.

```
14.03.12;12:43; 6:13;18:04;15.0;15.0;15.0;1.2;51;1025.9;Pada Pocasi;11;N;0.4;;0.0;628;2.9
```

Slika 10: Način vpisa podatkov v data.html

Če dobro pogledamo lahko opazimo, da so podatki zapisani brez enot, vsi pa so ločeni z podpičjem. Ker pa naša spletna stran bere direktno iz te datoteke, je potrebno podatke ločiti in zapisati enote. Takšnega načina izpisa podatkov v naši aplikaciji si ne smemo dovoliti. Zato smo napisali algoritem, ki te podatke loči. Zapisan je v posebni metodi, ki jo kličemo vsako minuto, kadar je spletna stran generirana. Metodo kličemo v metodi Load().

Osnovni princip algoritma deluje tako, da odpremo datoteko, preberemo vsebino v nek niz, nato pa ta niz razbijemo v polje nizov z ukazom .Split().

```
FileStream datoteka = File.Open(@"C:\Inetpub\wwwroot\data.html", FileMode.Open, FileAccess.Read, FileShare.ReadWrite);
StreamReader beri = new StreamReader(datoteka);
string vrstica = beri.ReadLine();
string[] minutniPodatki = vrstica.Split(';');
int smer = Convert.ToInt16(minutniPodatki[11]);
```

Slika 11: Del programske kode

Downld.08.txt vsebuje minutne podatke zadnjih osem dni, ker pa smo se odločili, da naša spletna stran prikazuje samo podatke zadnjih petih dni ob 12.00 uri je ostalih več tisoč vrstic neuporabnih. Do želenih zapisov smo prišli tako, da smo napisali iskalni algoritem, ki se požene samo enkrat ob zagonu spletne strani. Princip delovanja pa je takšen.

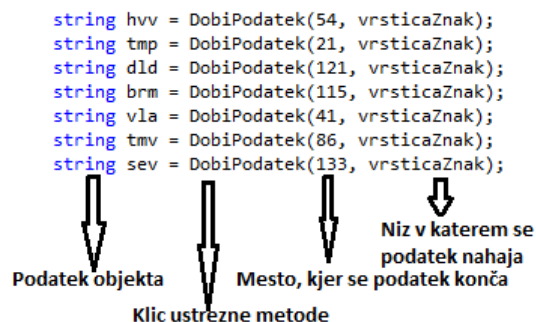
Algoritem najprej odpre datoteko in prebere vsako vrstico, dokler ne pride do konca, natanko petkrat. Vsakič pa se nastavi želen datum, ki se od današnjega po dnevih pomakne za en dan nazaj. Ko prebere vrstico jo najprej razbije in preveri če ustreza datumu, nato pa preveri, če ustreza ura. Ko najdemo pravo vrstico shranimo niz in pokličemo metode za branje podatkov.

Način branja, je zelo zanimiv. Spodnja slika prikazuje problem branja podatkov, naslednji odstavek pa rešitev problema.

Date	Time	Temp Out	Hi Temp	Low Temp	Out Hum	Dew Pt.	wind Speed	wind Dir	wind Run	Hi Speed	Hi Dir	wind chill
8.03.12	0:01	-0.2	-0.2	-0.2	67	-5.6	0.0	---	0.00	0.0	---	-0.2
8.03.12	0:02	-0.3	-0.2	-0.3	67	-5.7	0.0	E	0.00	0.4	E	-0.3
8.03.12	0:03	-0.3	-0.3	-0.3	68	-5.5	0.0	E	0.00	0.4	E	-0.3
8.03.12	0:04	-0.3	-0.3	-0.3	68	-5.5	0.0	E	0.00	0.4	E	-0.3
8.03.12	0:05	-0.3	-0.3	-0.3	68	-5.5	0.0	E	0.00	0.4	E	-0.3
8.03.12	0:06	-0.4	-0.3	-0.4	68	-5.6	0.4	E	0.03	0.9	E	-0.4
8.03.12	0:07	-0.4	-0.4	-0.4	68	-5.6	0.4	E	0.03	0.4	E	-0.4
8.03.12	0:08	-0.5	-0.4	-0.5	68	-5.7	0.4	E	0.03	0.4	E	-0.5
8.03.12	0:09	-0.5	-0.5	-0.5	68	-5.7	0.0	E	0.00	0.4	E	-0.5
8.03.12	0:10	-0.6	-0.5	-0.6	69	-5.5	0.0	E	0.00	0.4	E	-0.6
8.03.12	0:11	-0.6	-0.6	-0.6	69	-5.5	0.0	E	0.00	0.4	E	-0.6
8.03.12	0:12	-0.6	-0.6	-0.6	69	-5.6	0.0	E	0.00	0.4	E	-0.6
8.03.12	0:13	-0.6	-0.6	-0.6	69	-5.6	0.0	E	0.00	0.4	E	-0.6
8.03.12	0:14	-0.7	-0.7	-0.7	69	-5.7	0.4	E	0.03	0.4	E	-0.7
8.03.12	0:15	-0.7	-0.7	-0.7	70	-5.5	0.0	E	0.00	0.4	E	-0.7
8.03.12	0:16	-0.7	-0.7	-0.7	70	-5.5	0.0	E	0.00	0.4	E	-0.7
8.03.12	0:17	-0.7	-0.7	-0.7	70	-5.5	0.0	E	0.00	0.4	E	-0.7
8.03.12	0:18	-0.7	-0.7	-0.7	70	-5.5	0.0	E	0.00	0.4	E	-0.7
8.03.12	0:19	-0.8	-0.7	-0.8	70	-5.6	0.4	E	0.03	0.9	E	-0.8
8.03.12	0:20	-0.8	-0.8	-0.8	70	-5.6	0.4	E	0.03	0.9	E	-0.8
8.03.12	0:21	-0.8	-0.8	-0.8	70	-5.6	0.4	E	0.03	0.9	E	-0.8
8.03.12	0:22	-0.8	-0.8	-0.8	70	-5.6	0.4	E	0.03	0.4	E	-0.8
8.03.12	0:23	-0.8	-0.8	-0.8	70	-5.6	0.0	E	0.00	0.4	E	-0.8
8.03.12	0:24	-0.8	-0.8	-0.8	70	-5.6	0.4	E	0.03	0.4	E	-0.8
8.03.12	0:25	-0.8	-0.8	-0.8	71	-5.4	0.4	E	0.03	0.9	E	-0.8
8.03.12	0:26	-0.9	-0.8	-0.9	71	-5.5	0.4	E	0.03	0.4	E	-0.9
8.03.12	0:27	-0.9	-0.9	-0.9	71	-5.5	0.0	E	0.00	0.4	E	-0.9
8.03.12	0:28	-0.9	-0.9	-0.9	71	-5.5	0.0	---	0.00	0.0	---	-0.9
8.03.12	0:29	-0.9	-0.9	-0.9	71	-5.5	0.0	---	0.00	0.0	---	-0.9
8.03.12	0:30	-0.9	-0.9	-0.9	71	-5.5	0.0	---	0.00	0.0	---	-0.9
8.03.12	0:31	-0.9	-0.9	-0.9	71	-5.5	0.0	---	0.00	0.0	---	-0.9
8.03.12	0:32	-0.9	-0.9	-0.9	71	-5.5	0.0	E	0.00	0.4	E	-0.9
8.03.12	0:33	-1.0	-0.9	-1.0	71	-5.6	0.0	E	0.00	0.4	E	-1.0
8.03.12	0:34	-1.0	-1.0	-1.0	71	-5.6	0.0	---	0.00	0.0	---	-1.0
8.03.12	0:35	-1.0	-1.0	-1.0	71	-5.6	0.0	E	0.00	0.4	E	-1.0
8.03.12	0:36	-1.0	-1.0	-1.1	72	-5.4	0.0	E	0.00	0.4	E	-1.0
8.03.12	0:37	-1.1	-1.1	-1.1	72	-5.5	0.0	---	0.00	0.0	---	-1.1
8.03.12	0:38	-1.1	-1.1	-1.1	72	-5.5	0.0	---	0.00	0.0	---	-1.1

Slika 12: Način zapisa v datoteki downld08.txt

Težava je nastala zaradi tega, ker se zapisi podatkov ne začnejo vedno na istem mestu, ampak se na istem mestu končajo. Rešitev smo hitro našli in težavo prav tako hitro odpravili. Podatke beremo znak po znak in namesto, da bi jih brali naprej jih beremo nazaj dokler ne pridemo do praznega mesta, potem pa za toliko polj kot smo šli nazaj, ta podatek preberemo naprej, nato ga seveda še shranimo v objekt. Analizirana spodnja koda prikazuje klic ustreznih metod in shranjevanje v ustrezne objekte.



Slika 13: Klic metod in shranjevanje v ustrezne objekte

3.2 Prikazovanje podatkov

Prikaz podatkov na uporabniku preprost, zanimiv in praktičen način je zelo pomemben faktor, ki vpliva na priljubljenost vseh izdelkov, ki so namenjeni podajanju in predstavljanju informacij nekemu končnemu uporabniku. Prikaz podatkov je ključni element, ki obiskovalca pritegne in ga spremeni v uporabnika, ali pa ga odvrne in izdelka oz. storitve nikoli več ne bo uporabil.

Zato smo prikazu podatkov namenili precej časa. Najprej smo se morali odločiti katere podatke bomo sploh prikazovali, nato določili postavitev teh podatkov in izbrali najustreznejše načine prikaza. Vsi smo se strinjali, da bodo vse podatke spremljali slike in da bomo vodili nekakšno evidenco vseh zapisov. Za prikaz podatkov smo izbrali spletno aplikacijo.

3.2.1 Prikazovanje trenutnih podatkov

Spletna stran ima za trenutno vreme na razpolago prikazovanje desetih podatkov, za katere menimo, da so najpomembnejši. Ti podatki so: temperatura, pritisk, vlažnost, količina dežja, hitrost vetra, temperatura vetra, rosišče, sevanje, toplotni indeks in UV indeks. Vsi podatki so prikazovani v najprimernejših enotah. Na podlagi podatkov izbiramo sliko, ki nam prikazuje vreme in je prikazana nad podatki.

Prikazovanje podatkov za trenutno vreme je sestavljeno tako, da se pooblaščen oseba vpiše oziroma prijavi v že ustvarjen račun in samo on/a ima dovoljenje, da se odloči katere od desetih podatkov, želi prikazovati. To smo ustvarili tako, da pooblaščen oseba klikne na zelene podatke in nato potrdi spremembe, kot prikazuje spodnja leva slika.

<input type="radio"/> Temperatura	<input type="radio"/> Temperatura vetra
<input type="radio"/> Pritisk	<input type="radio"/> Rosišče
<input type="radio"/> Vlažnost	<input type="radio"/> Sevanje
<input type="radio"/> Količina dežja	<input type="radio"/> Toplotni indeks
<input type="radio"/> Hitrost vetra	<input type="radio"/> UV indeks

Vse shranjene spremembe shranimo v posebno datoteko. Vsak podatek ima svojo številko. Vse številke podatkov se shranjujejo v datoteko kot nizi, ki so med seboj ločeni, da jih je mogoče kasneje tudi prebrati. Datoteka je ustvarjena samo za branje, kar pomeni da ne moremo sami vpisati številk.

Zapisali smo program, ki preverja, kateri podatki so bili izbrani, pri čemer smo si pomagali s datoteko, ki smo jo ustvarili ravno s tem namenom. Program pa prav tako na podlagi zapisov v datoteki, izpiše oziroma prikaže potrebne podatke na spletno stran.

Slika 14: Izbira podatkov za prikaz

Ne prikazujemo pa samo podatke za trenutno vreme, ampak tudi vreme za prejšnjih pet dni, ki so prikazani na levi strani spletne strani. Za ta del, je bil ravno tako potreben program. Ta program vsebuje izpisovanje podatkov in sicer ne izpisujemo vse, kot smo to storili za prikazovanje trenutnega vremena, ampak samo podatke kot so: temperatura, pritisk, vlažnost, količina dežja, hitrost vetra in temperatura vetra. Poleg tega je bilo potrebno sprogramirati, da je prikazovalo dneve v pravilnem vrstnem redu, kot si sledijo v tednu.

Odločili smo se, da bomo prikazovali podatke tudi za zadnjih pet dni ob 12:00 uri, saj mnogo uporabnikov zanima vpogled v stanje preteklih nekaj dni, koristno pa je tudi za logično primerjavo. Poleg tega takšen prikaz aplikacijo dopolni in jo naredi bolj dinamično. Podatke prebere spletna stran iz datoteke downld08.txt samo enkrat ob zagonu, saj se za razliko od minutnih podatkov ti ne rabijo osveževati.



Slika 15: Vreme zadnjih 5 dni

3.2.2 Prikazovanje statistike in generiranje grafikonov

Odločili smo se, da uporabimo za izpis statistike grafikone in tabele. Ti so prikazani posebej, kar pomeni, da ko preverjamo statistiko na primer za nek dan prejšnjega meseca ob strani ne bodo prikazani trenutni in tedenski podatki, saj bi bila tako stran prenatrpana s podatki in informacijami, ki pa jih uporabnik niti ne bi mogel pregledati. Zato je statistika prikazana na svojih straneh spletne strani. Ker pa uporabnika ne zanima kakšno je bilo vreme prejšnji mesec neke minute, smo napisali aplikacijo, ki teče na strežniku, bere trenutne podatke, računa povprečje in nato to povprečje vsako uro shranjuje v bazo. V podatkovno bazo se shrani tudi datum in ura zapisa. Dinamičnost spletne strani pa ustvarjajo digitalna ura, datum in kompas, ki služi prikazovanju smeri vetra.

Ob strani je viden tudi vizualen prikaz takratnega stanja, saj si tako obiskovalci naše spletne strani lahko lažje predstavljajo prejšnje stanje. Prav tako takšen prikaz naredi spletno stran veliko bolj atraktivno, prijazno uporabniku in veliko bolj uporabno ter funkcionalno.

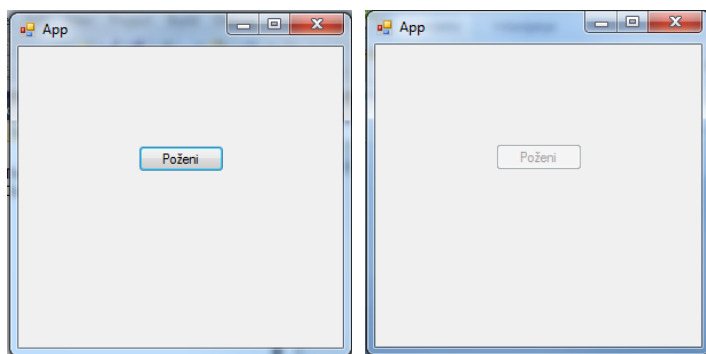
Iz podobnega razloga je tudi pri vsaki napovedi vizualna ponazoritev vremena. Te metode se meteorologi in tudi novinarji poslužujejo od samih začetkov napovedovanja vremena.

Stvar je tudi sprogramirana dovolj dinamično, da spletna stran oziroma algoritem sam določi prejšnje dneve na osnovi današnjega dne in poišče ustrezne zapise. Zaradi lažjega razumevanja so tudi dnevi predstavljeni z imenom v tednu namesto z konkretnim datumom.

3.2.2.1 Aplikacija za izračun povprečja in shranjevanja v podatkovno bazo

Aplikacija za izračun povprečja je prav tako zapisana v Visual Studio razvojnem okolju in sicer v jeziku C#. Je namizna aplikacija, njena sestava pa je dokaj preprosta. Vsebuje en timer, ki se sproži vsako minuto. Ko se timer sproži, se pokliče metoda za izračun povprečja, nato pa program preveri ali je polna ura, če je ura recimo dvanajst popoldne, se bodo vsi povprečni zapisi shranili v podatkovno bazo. Aplikacija je uporabna, ker omogoča takšen način shranjevanja, ki pa je pogoj za prikaz mogoče malo naprednejše statistike in vizualnih ponazoritev, kot so grafikoni. Aplikacija prav tako bere trenutne podatke, vzame smer vetra v stopinjah, poišče sliko sever.png in jo obrne za toliko stopinj, kot je zapisano v datoteki. Nato pa sliko shrani pod imenom kurzor.png. To se zgodi vsako minuto.

Vizualna zgradba te aplikacije je zelo preprosta, saj vsebuje samo en gumb, ki sproži timer. Dostop do te aplikacije imajo samo skrbniki spletnega strežnika, ne pa tudi uporabniki in obiskovalci naše spletne strani.



Slika 16: Vizualna aplikacija

3.2.2.2 Prikaz statistike

Statistika je prikazana z tabelo, ki prikazuje povprečje enournih podatkov za en dan. Uporabnik lahko na koledarju sam izbere leto, mesec in pa dan. Stvar je dovolj dinamična, da če zapisov za tisti dani ni, se spletna stran ne poruši ampak se ne zgodi nič. V statistiki so prikazani vsi tipi podatkov, od temperature do pritiska. Maksimalno število zapisov za en dan pa je natanko 24, kolikor je tudi ur v dnevu.



februar		marec 2012					april
pon	tor	sre	čet	pet	sob	ned	
<u>27</u>	<u>28</u>	<u>29</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	
<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>	<u>9</u>	<u>10</u>	<u>11</u>	
<u>12</u>	<u>13</u>	<u>14</u>	<u>15</u>	<u>16</u>	17	<u>18</u>	
<u>19</u>	<u>20</u>	<u>21</u>	<u>22</u>	<u>23</u>	<u>24</u>	<u>25</u>	
<u>26</u>	<u>27</u>	<u>28</u>	<u>29</u>	<u>30</u>	<u>31</u>	<u>1</u>	
<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>	<u>8</u>	

Slika 17:Koledar za izbiro zelenega dneva

S to metodo obiskovalca pritegnemo, saj dobi vpogled v celotno zgodovino zapisov, od kar je postavljena vremenarska postaja.

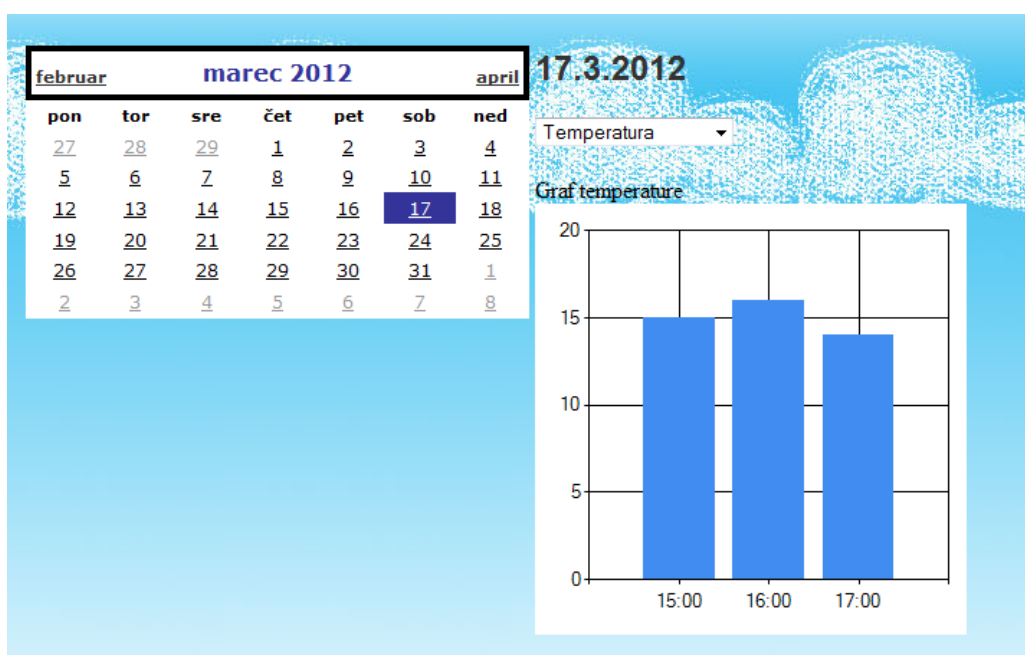


Statistika								
17.3.2012								
ID	Datum	Ura	Temperatura	Pritisk	Padavine	Hitrost_Vetra	Vlaznost	Temperatura_Vetra
3	17.3.2012	15:00	15	1022,6	0	1,5	45	18
4	17.3.2012	16:00	16	1023	0	1,8	44	15
5	17.3.2012	17:00	14	1022	0	1,1	55	10

Slika 18: Prikaz podatkov zelenega dneva

3.2.2.3 Prikaz grafikov

Grafikoni so zelo pomemben element naše spletne strani, saj omogočajo preprost način prikaza naprednih podatkovnih struktur. Spletna stran od uporabnika zahteva katere vrste grafikov želi uporabnik analizirati, oziroma dostopati do njih, nato klikne na zelen datum in prikaže se stolpčni diagram. Prav tako na tej strani uporabnik nima vpogleda v trenutno stanje in pa stanje zadnjih pet dni. Grafikoni so dinamični, kar pomeni, da se diferenca med enotami prilagaja velikosti vnesenih podatkov in številu zapisov. To pomeni, da če so za današnji dan opravljene šele trije zapisi, se bodo na grafikonu prikazali samo trije stolpci.



Slika 19: Prikaz grafikona

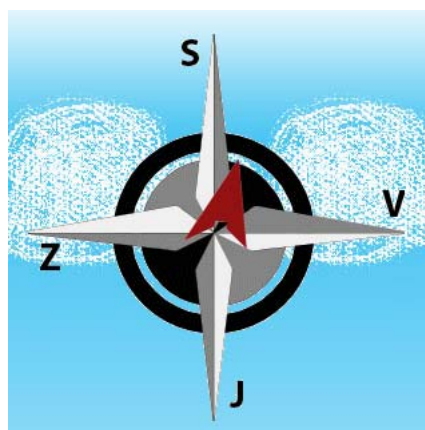
3.2.3 Smer vetra ponazorjena z kompasom

Ljudje, si lažje predstavljamo stvari v okolju, če jih vidimo, ali pa če nam nekdo neko stvar opiše, nariše ali razloži. Najboljši način prikaza nekih informacij je seveda vizualna predstavitev, zato smo tudi smer vetra, namesto, da bi jo zapisali kot število stopinj ali pa kot smeri neba, kot so sever, jug, vzhod in zahod, odločili za vizualno ponazoritev s kompasom.

V zgornjem desnem okvirčku naše spletne strani je ta element tudi viden. Gre za kompas, ki ima na sredini kurzor, ki se premika v tisto smer, v katero veter piha. Stvar zgloda zelo preprosta, a je bilo pravzaprav takšen dinamičen element zelo težko sprogramirati.

Težavo smo rešili po korakih. Razvojno okolje Visual Studio WebDeveloper ne vsebuje nikakršnih elementov, s katerimi bi lahko poljubno obračali slike. Zato smo uporabili že prej omenjeno namizno aplikacijo, ki vzame sliko sever.png, ki je pravzaprav kurzor z prozornim ozadjem in jo obrne za toliko stopinj, kot je zapisano v datoteki. To sliko shrani pod imenom kurzor.png. Šele to sliko prikaže naša spletna stran. Ker je slika prozorna, je v ozadju lepo viden kompas, ki je nastavljen kot ozadje celice v kateri se kurzor nahaja. Dalje je bilo potrebno tudi ugotoviti, kako sliko osveževati, ne da bi bilo potrebno posodobiti celo stran, saj se slika naloži samo enkrat. Naš brskalnik smo »prelisičili« tako, da smo v url slike napisali poizvedbo, ki vrne nič, ampak se vedno, ko se ta poizvedba izvede slika ponovno naloži. Ta koda je prikazana spodaj, ker je DateTime.Now vrednost vsakič različna, se poizvedba zažene vsakič, ko se posodobi update panel. Kar pomeni, da se slika posodobi vsako minuto in kurzor se obrne vedno, ko se smer vetra spremeni. Seveda to poteka na minutnih intervalih, saj se tudi podatki posodablajo vsako minuto.

```
Image1.ImageUrl = "~/Slike/kurzor.png?myTime=" + DateTime.Now;
```

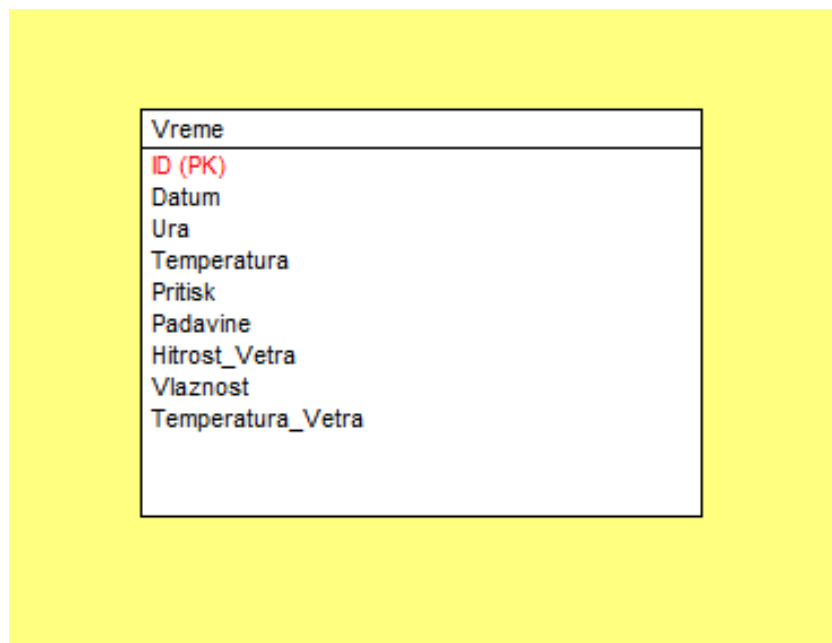


Slika 20: Smer vetra

3.3 Podatkovna baza

Podatkovna baza smo kreirali na podlagi podatkov, ki jih prejmemo iz vremenske postaje. Na podlagi teh podatkov smo tako izračunali povprečje, ki smo jih nato shranili v bazo. Podatki se shranjujejo preko aplikacije, ki smo jo zapisali v programskem jeziku C#, ti so tako zapisani v bazo vsako uro. Baza vsebuje eno tabelo, saj jo potrebujemo samo za prikaz podatkov v obliki grafov na spletni strani.

Za kreiranje baze smo uporabili pristop top-down, ki pa je bil rahlo okoren, glede na to, da imamo eno tabelo, zaradi česar tudi ni bila potrebna normalizacija. Za primarni ključ smo določili atribut ID, ki je tipa integer in se samodejno povečuje. Medtem ko se ostali podatki prenašajo iz aplikacije in so tipa float, izjemi sta Datum in Ura, ki nam povesta kdaj so bili podatki zapisani in kakšni so bili povprečni podatki.



Slika 21: E-R diagram tabele

3.4 Oblika in CSS

Spletna stran je narejena v programskem razvojnem okolju Microsoft Visual-Studio. MS Visual-Studio ima na razpolago več programskih jezikov (visualBasic, C++, C#, ...). To razvojno okolje omogoča izdelovanje spletnih strani v asp.net v katerem je napisana koda za prikazovanje spletne strani. Programski del kode pa je napisan v Visual C#. Oblika je napisana s pomočjo CSS (CascadingStyleSheets).

Gradnja spletne strani se začne s tem, da postavimo tabelo na sredino spletne strani, ker ima Visual C# možnost MasterPage (nekakšna stran nad stranmi v kateri se prikazujejo podatki)

naredimo obliko le enkrat in postavimo ContentPlaceholder (orodje kjer se prikazujejo podatki iz postrani) na sredino.



Slika 22: Tabela v C#

Seveda ima tabela

lahko več celic ampak dokler ne vemo koliko podatkov bomo prikazovali je za začetek tabela 3x3 zadosti velika. Tabelo pa zaradi oblike velikokrat postavimo na sredino strani.

Za obliko se uporablja CSS ki se lahko uporablja v zunanji datoteki, v glavi ali pa kar v tagu (oznaki). S CSS lahko definiramo stil HTML oz. XHTML elementov, kako naj se prikažejo na strani. Določamo lahko barve, velikosti, oblike, pozicije, ... S CSS zmanjšamo količino kode, saj omogočimo več stranmi uporabo tega kar smo napisali v CSS, kar bistveno zmanjša količino kode, prav tako pa nam prihrani delo, saj moramo CSS napisati samo enkrat in ga nato lahko večkrat uporabimo.

Da na kratko opišem kodo CSS ki sem jo največkrat uporabil.

```
Ustvarimo ← #levo
nov id.    ← {
            background: url('/Spletna stran/Slike/L1.jpg') repeat-y;
            }
            #desno
            {
            background: url('/Spletna stran/Slike/D2.jpg') repeat-y;
            }
            }
```

Pot do slike

Slika 23: CSS koda

Ko smo ustvarili nov id in v njega lahko vpišemo kar želimo glede oblike. V tem primeru smo uporabili ukaz za ozadje katerega smo uporabili v naši spletni strani. V HTML oznaki potem samo uporabimo ta stil (<td id="levo">) in se nam ta slika ponovi horizontalno, saj smo napisali repeat-y.

4 NADGRADNJA IN BODOČI CILJI

Nadgradnja spletne strani je povsem odprte narave, vendar je priporočljiva. Potrebno je slediti trendom, saj nas v nasprotnem primeru povozí čas. Spletna stran ni nikoli zaključen projekt, saj jo je vedno mogoče izboljšati oz. obogatiti in s tem narediti posebno.

Naš bodoči cilj je izboljšava same spletne strani iz oblikovnega vidika in morda tudi iz vidika prikazovanja podatkov.

5 RAZPRAVA

Problematika podatkov, ki so prikazani na tak način, da jih ne razumemo bi bilo treba popraviti tako, da bi jih vsi razumeli zato smo se odločili, da bomo poskušali narediti spletno stran, ki bo prikazovala podatke iz vremenske postaje.

Iskali smo načine kako prebirati podatke iz vremenske postaje in smo na koncu našli pravo, najlažjo možnost, ki se nam je ponudila.

Podatki so bili shranjeni v podatkovno bazo, da lahko prikazujemo grafe in statistiko na spletni strani.

Oblikovana je bila spletna stran za prikazovanje podatkov iz vremenske postaje in izpostavili smo tudi, kako bi lahko spletno stran nadgradili, da bi bila spletna stran še bolj uporabna in razumljiva.

6 ZAKLJUČEK

Namen naše raziskovalne naloge je bila postavitve spletne strani s prikazom podatkov iz vremenske postaje. Najpomembnejše je bilo dobiti podatke iz vremenske postaje. Načinov za pridobitev teh podatkov je bilo več in nekaj izmed njih smo tudi bolj natančno navedli. Zatem jih je bilo potrebno prikazati oziroma izpisati na spletno stran. Skozi posamezne postopke, ki so navedeni in opisani v sami raziskovalni nalogi je tako nastal končni izdelek.

Med samo izdelavo smo naleteli tudi na težave, ki pa smo jih s skupnimi močmi uspeli odpraviti. Težave so se pojavile že kar pri prvem koraku, kako pridobiti podatke iz vremenske postaje, da jih bomo lahko prikazovani na naši spletni strani. S težavami smo se sproti ukvarjali, zato je bilo potrebno napisati program kako »prisluškovati« podatkom iz COM porta. Težava pa je tudi nastala ko smo dobili datoteko z končnico .wlk saj je še nismo nikoli uporabljali in je bilo potrebno pridobivati podatke iz nje, vendar je bilo za to več rešitev in smo morali razmisliti za katero odločitev se bomo odločili. Ko smo končno ugotovili kako prikazovati podatke na spletni strani se je bilo treba odločiti katere podatke bomo prikazovali, da ne bo prevelike nasičenosti s podatki in bo tako nastala samo še ena spletna stran s preveliko količino podatkov.

7 VIRI IN LITERATURA

- Spletna stran Codeproject / nazadnje preverjena 16.3.2012

<http://www.codeproject.com/Articles/4416/Beginners-guide-to-accessing-SQL-Server-through-C>

- Spletna stran Microsoftovega razvojnega pomočnika / nazadnje preverjena 15.3.2012

<http://msdn.microsoft.com/en-us/default>

- Spletna stran Microsoft ASP.NET / nazadnje preverjena 16.3.2012

<http://www.asp.net/web-pages>

- Spletna stran w3school / nazadnje preverjena 14.3.2012

<http://www.w3schools.com/sql/default.asp>

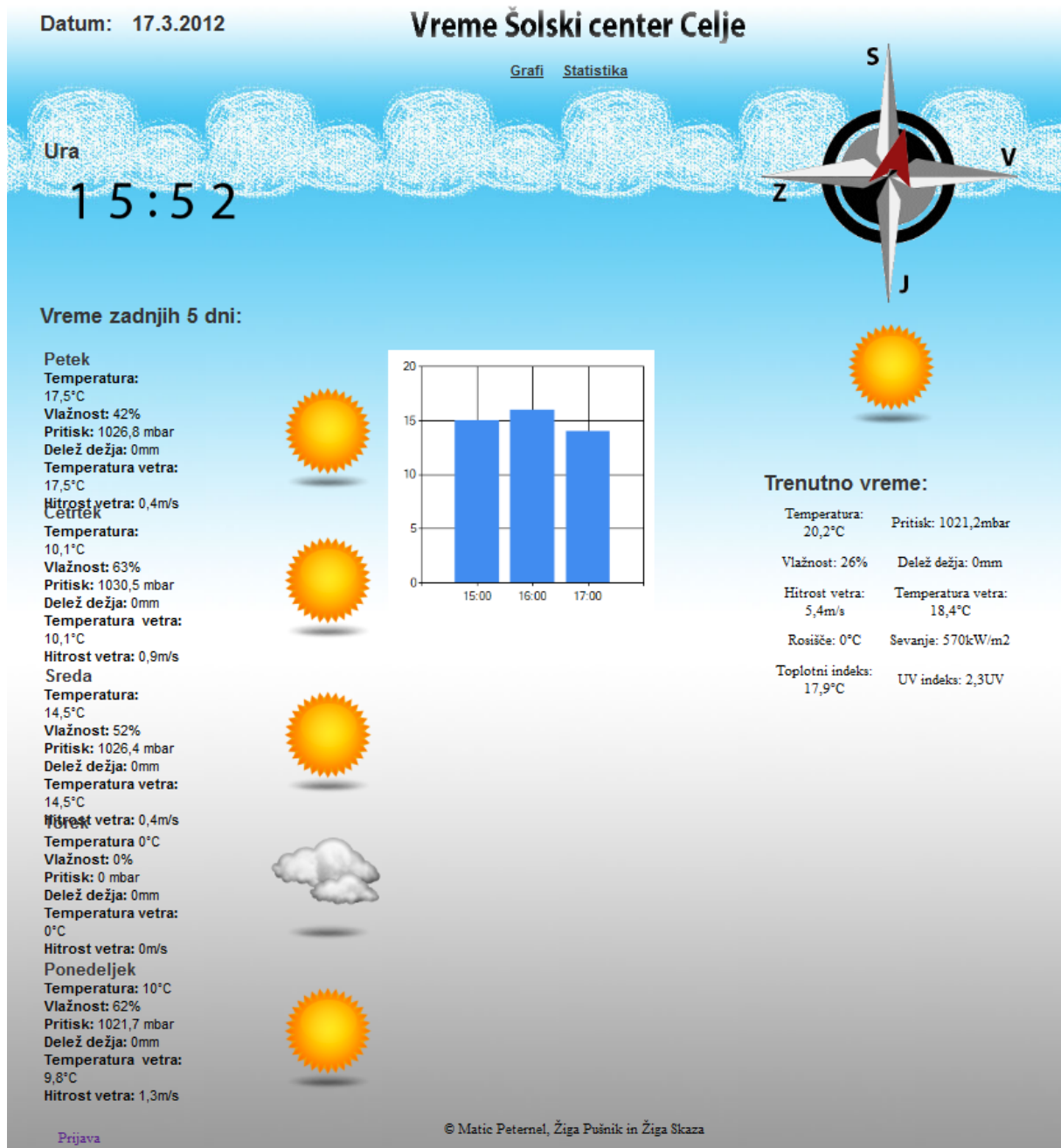
8 ZAHVALA

Naša delovna skupina bi se rada zahvalila mentorju MojmirjuKlovarju za pomoč pri izbiri teme za to nalogo in kasneje za pomoč pri uresničitvi naših želja. Seveda pa gre zahvala tudi osebjju, ki so nam bili v veliko pomoč pri končni uresničitvi naših zamisli in želja.

9 KAZALO SLIK

<i>Slika 1: Datoteka WetherLink s končnico .wlk</i>	5
<i>Slika 2: Nastavitve COM porta</i>	5
<i>Slika 3: Odsek programske kode aplikacije v C#</i>	5
<i>Slika 4: Ukazi za čiščenje predpomnilnika</i>	6
<i>Slika 5: Shranjevanje in pretvorba podatkov</i>	7
<i>Slika 6: Konzolska aplikacija z zaprtim programom WeatherLink</i>	7
<i>Slika 7: Konzolska aplikacija z odprtim programom WeatherLink</i>	8
<i>Slika 8: Prikaz napredne statistike v WeatherLinku</i>	8
<i>Slika 9: Prikaz datotek v raziskovalcu</i>	9
<i>Slika 10: Način vpisa podatkov v data.html</i>	9
<i>Slika 11: Del programske kode</i>	9
<i>Slika 12: Način zapisa v datoteki downld08.txt</i>	10
<i>Slika 13: Klic metod in shranjevanje v ustrezne objekte</i>	11
<i>Slika 14: Izbira podatkov za prikaz</i>	12
<i>Slika 15: Vreme zadnjih 5 dni</i>	13
<i>Slika 16: Vizualna aplikacija</i>	14
<i>Slika 17: Koledar za izbiro zelenega dneva</i>	15
<i>Slika 18: Prikaz podatkov zelenega dneva</i>	15
<i>Slika 19: Prikaz grafikona</i>	16
<i>Slika 20: Smer vetra</i>	17
<i>Slika 21: E-R diagram tabele</i>	18
<i>Slika 22: Tabela v C#</i>	19
<i>Slika 23: CSS koda</i>	19
<i>Slika 24: Končna spletna stran</i>	26

10 PRILOGE



Slika 24: Končna spletna stran