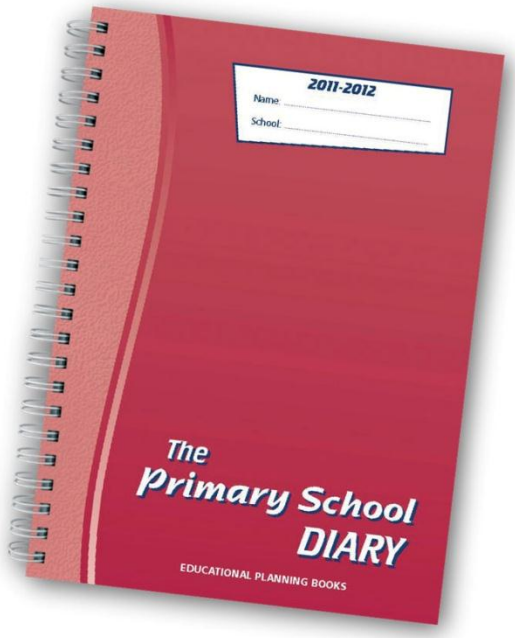
Mestna občina Celje
Komisija mladi za Celje



ŠOLSKI SPLETNI DNEVNIK

RAZISKOVALNA NALOGA

AVTORJI

Žiga AJDNIK
Tomaž KRAŠEK
Marko DEŽELAK

MENTOR

Dušan FUGINA

Celje, marec 2013

Šolski center Celje

Srednja šola za kemijo, elektrotehniko in računalništvo

Pot na lavo 22

3000 Celje

ŠOLSKI SPLETNI DNEVNIK

raziskovalna naloga

Avtorji:

Žiga AJDNIK, E-4. e
Tomaž KRAŠEK, E-4. e
Marko DEŽELAK, E-4. e

Mentor:

Dušan FUGINA

Mestna občina Celje, Mladi za Celje

Celje, 2013

Kazalo vsebine

Kazalo slik

Povzetek

V raziskovalni nalogi smo ustvarili spletno stran, s katero bi lahko nadomestili klasičen šolski dnevnik v papirni obliki. Ker se je elektronska redovalnica izkazala za odlično idejo, smo se odločil, da še poizkusimo z elektronskim dnevnikom. Za dosego cilja smo ustvarili spletno stran v ASP.NET ogrodju, ki komunicira s podatkovno bazo v kateri se hranijo podatki, ki so potrebni za delovanje šolskega sistema. Spletni strani smo dodali preprost varnostni sistem, tako da so podatki varni pred nezaželenimi očmi. Že na začetku raziskave smo ugotovili, da dnevnik vsebuje veliko več podatkov, kot pa smo jih pričakovali in posledično je stran postala obširnejša. Vendar je še vedno pregledna in enostavna za uporabo.

Abstract

In this research assignment we created a website, with which we could replace the classic school journal in analogue paper form. Because (e-redovalnica) is already in good use, we decided to make the same thing with journal. For accomplishing this goal we created a web page in ASP.NET framework, which communicates with the database, where all the data necessary for school system to work is stored. We than added a simple security system to the web page and therefore secured the data from the unwanted eyes. In the beginning of research we discovered that journal contains a lot more data that we expected and because of that web site become more extensive. But it's still transparent and easy to use.

Ključne besede / Keywords

Ključne besede:

Microsoft SQL Server, Evidenca pouka, Izostanek, Spletni dnevnik

Keywords:
Microsoft SQL Server, record lessons, abscence, web school diary

Kratice in okrajšave

SQL – strukturirani povpraševalni jezik

DDL – jezik da definiranje podatkov

DML – jezik za upravljanje s podatki

DCL – jezik za kontrolo nad podatki

TCL – jezik za kontrolo nad transakcijami

1 Uvod

V trenutni moderni družbi nas spremlja informacijska tehnologija in računalniki na vsakem koraku. Izboljšujejo nam kakovost življenja in nam pomagajo pri vsakdanjih opravilih. Vse več stvari lahko počnemo preko spleta, na kar smo se že navadili. Različna podjetja nam že ponujajo možnosti nakupovanja ali pa plačevanja položnic.

Ravno zaradi tega in že uveljavljene e-redovalnice, smo se odločili, da bi profesorjem še dodatno olajšali delo in vsakodnevna papirnata opravila spremenili v digitalno obliko. Preglednost dnevnika v trenutni papirnati obliki je sicer solidna, funkcionalnost pri pregledu statistike pa porazno zamudna. Zato je namen naše naloge, da bi zmanjšali čas za vpis, izpis v dnevnik in bistveno poenostavili pregled statistike. S tem pa ne bi zmanjšali same preglednost in uporabnosti dnevnika.

1.1 Cilj

Zadali smo si, da bomo izdelali enostavno in uporabniku prijazno aplikacijo, katera bo omogočala enostavnejšo pripravo zaključnih poročil. Pri tem smo si zadali naslednje cilje:

-enostaven in hiter uporabniški vmesnik, ki pa bo funkcionalen ter omogočal preprosto prijavo ter spreminjane, dodajanje in brisanje podatkov

-enostavnejše pregled nad podatki

-lažje ustvarjanje zaključnih poročil

2 Spletna stran

2.1 Teoretični uvod

2.1.1 Podatkovna baza

2.1.1.1 SQL

SQL ali strukturirani povpraševalni jezik za delo s podatkovnimi bazami (angl. *Structured Query Language*) je najbolj razširjen in standardiziran jezik za delo s relacijskimi podatkovnimi bazami. Sestavljen je iz štirih delov:

-jezik za definicijo podatkov (DDL); z njim ustvarimo podatkovno bazo, definiramo in spreminjamo shemo podatkovne baze in ustvarjamo poglede

-jezik za rokovanje s podatki(DML); z njim vstavljamo, spreminjamo, brišemo in povprašujemo po podatkih. DML je lahko postopkoven, potrebno je navesti postopek kako priti do rezultata ali pa nepostopkoven, kjer pa samo opišemo kakšen rezultat želimo dobiti

-jezik za kontrolo nad podatki(DCL); z njim nastavljamo pravice uporabnikom podatkovne baze

-jezik za kontrolo nad transakcijami(TCL); z njim nadziramo transakcije.

2.1.2 Uporabniški vmesnik

2.1.2.1 HTML

HTML ali jezik za označevanje nadbesedila(Hyper Text Markup Language) je označevalni jezik namenjen za izdelovanje spletnih strani. Sestavljen je iz značk, npr. za odstavek, katere predstavljajo osnovo spletne strani.

2.1.2.2 CSS

Z začetkov HTML-ja se je povečala izdelava spletnih strani. HTML je vseboval značke za urejanje izgleda strani, a so bile te premalo zmogljive, poleg tega so bile prepletene z vsebino. Ustvarili so CSS(kaskadne stilske podloge), s katerimi definiramo HTML značko, kako se naj prikaže na spletni strani. Določamo lahko barvo, zamike, velikost, poravnave, itd.

Bistvo CSS je ločiti vsebino spletne strani od njenega oblikovanja. S tem omogočimo večjo preglednost nad vsebino, urejanje več HTML značk naenkrat in lažje ustvarjanje in urejanje stilov. S tem preprečimo ponavljanje kode spletne strani, kar posledično pomeni, manjši HTML dokument.

Stile lahko definiramo v novi datoteki in kasneje v glavi HTML dokumenta definiramo povezavo do te datoteke. Namesto nove datoteke, lahko stile pišemo tudi kar v glavi HTML dokumenta ali pa kar neposredno zraven značke, kar ni najboljše saj je namen stilov, da ločimo oblikovanje od vsebine.

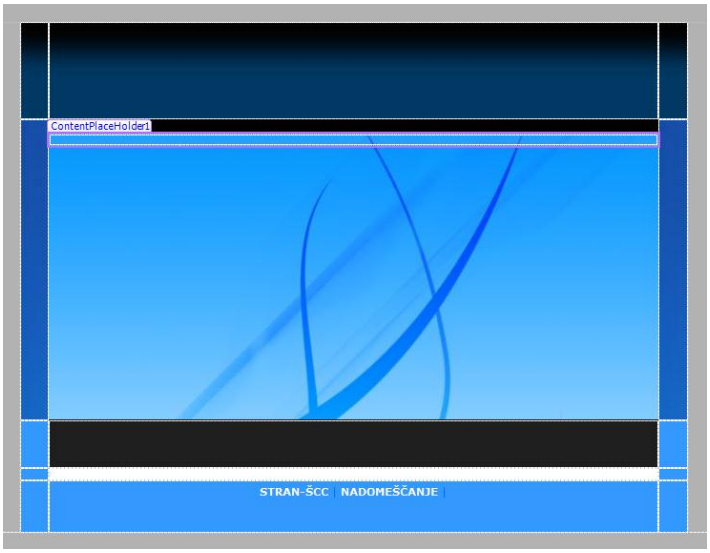
2.1.2.3 C#

C# je programski jezik tretje generacije programskih jezikov. Nastal je pod okriljem Microsofta. Ustvarjen je bil z namenom, da bi bil preprost, kvaliteten, objektno orientiran ter internetno usmerjen programski jezik.

Spletna stran je ustvarjena v programsko razvojnem okolju Microsoft Visual Studio. MS Visual Studio je nastal že leta 1995. Nudi nam ustvarjanje konzolnih, grafičnih ali spletnih aplikacij. Podpira pa večje število programskih jezikov kot so: C#, C++ in Visual Basic. Za razvoj spletne strani pa smo uporabili spletno programsko okolje asp.net ki nam omogoča prikazovanje spletnih strani. Na voljo je od leta 2002. Za sam programski del kode pa smo uporabili nam najbolj znan C# ki ga je razvil Microsoft in je tudi največkrat uporabljen jezik pri programiranju spletnih strani v asp.net. Spletno stran pa smo oblikovali z CSS(Cascading Style Sheets) s katerim lahko določaš pravila kako se naj elementi zgeldajo na strani.

2.2 Oblikovanje

Ogrodje spletne strani smo naredili s pomočjo table. Zaradi lepšega zgleda in boljše preglednosti smo tabelo postavili na sredino strani, prostor strani smo razdelili na več delov. V osrednji del pa smo postavili ContentPlaceHolder. Tako smo naredili glavno stran(Master Page). Kajti ContentPlaceHolder nam omogoča, prikazovanje podatkov iz podstrani.



Slika 1: Osnovna postavitev

Za lažje in bolj pregledno oblikovanje smo vsakemu delu tabele ustvarili razred(class). Kasneje pa smo razrede porabili v drugi datoteki s končnico .css. Da pa je glavna stran uporabila CSS nastavitve smo jih povezali s pomočjo stavka (`href="style/style1.css" rel="stylesheet" type="text/css"`). Nato pa smo vsakem razredu določili svoje nastavitve kot so barva, velikost, postavitev, ozadje … S tem smo očitno zmanjšali število vrstic kode v glavnem programu. Velika prednost takšnega načina dela je tudi to, da jih napišemo samo enkrat in jih lahko uporabimo večkrat.

```
td.body_content
{
    padding: 5px;
    background-image: url(../images_template1/bg_content.jpg);
    background-repeat: repeat;
}
td.below_header
{
    background-color: #1F1F1F;
    height: 50px;
    color: #CCCCCC;
    padding: 5px;
}
```

Slika 2: Del CSS datoteke

2.3 Shranjevanje podatkov

V šolskem sistemu pomeni dnevnik zelo veliko. Zato smo se morali zelo potruditi pri izdelavi podatkovne baze, ki bo nadomestila standardno papirno obliko dnevnika. Izdelavo podatkovne baze smo se lotili od vrha navzdol (top-down), kar pomeni da smo se najprej lotili velikih enot, katere smo v podatkovni bazi predstavili s tabelami in jim kasneje dodali podrobnosti, ki so predstavljene z atributi v tabelah. Osnovne tabele so bile:

-Dijak

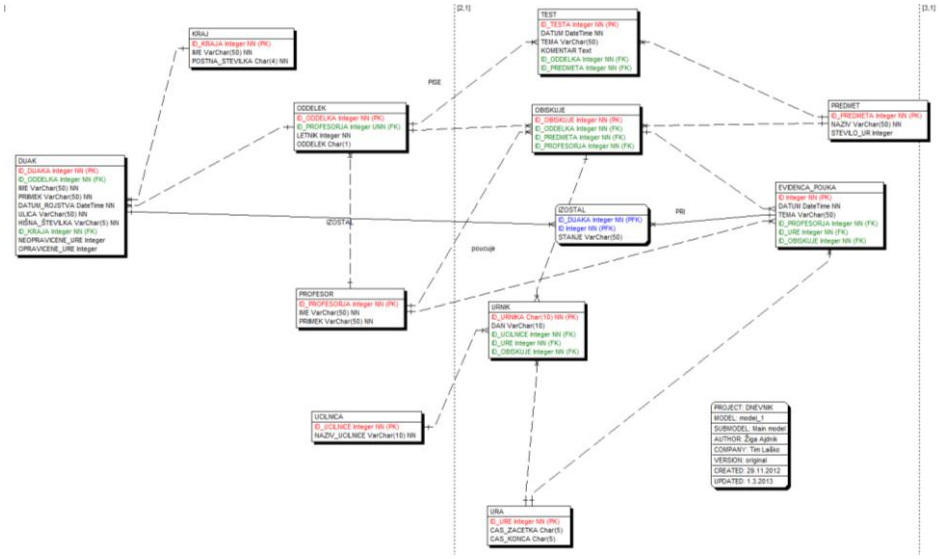
-Oddelek

-Predmet

-Profesor

-Evidenca_Pouka

S temi tabelami lahko shranimo eno šolsko uro. Ker smo želeli, da bi naš dnevnik omogočal več, kot samo beleženje ur, smo dodali še tabelo »Izostanek«, v kateri se beležijo izostanki pri pouku. V zdajšnji obliki dnevnika so napovedani tudi vsi testi, zato smo ustvarili novo vmesno tabelo in jo povezali z predmetom in oddelkom. Ker nismo hoteli, da bi pršilo do zmešnjav pri predmetih na tehničnih šolah smo dodali tabelo obiskuje, s katero smo zagotovili, da ima posamezen oddelek(program na srednjih šolah) samo predmete z njihovega programa (npr. oddelek računalniških tehnikov ne more imeti predmeta kemijskih tehnikov). Hkrati smo tudi zagotovili da profesorji iz istega predmeta ne morejo vpisovati oddelkov, katerih ne učijo razen v posebnih primerih(nadomeščanja). Posamezno smo ustvarili podatkovno bazo urnik, za katero smo pa kasneje ugotovili, da jo lahko implementiramo v že obstoječo podatkovno bazo za dnevnik. Za ustvarjanje podatkovnega modela smo si pomagali s CASE Studiom.

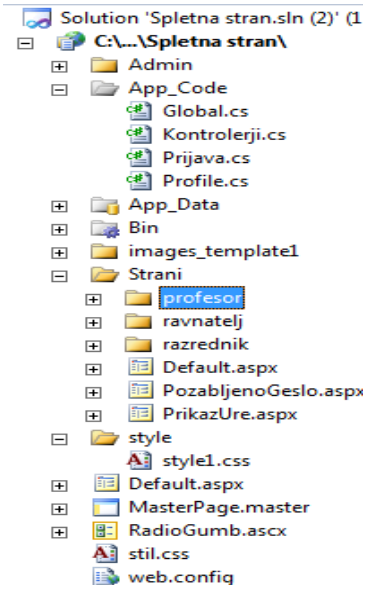


Slika 3: Podatkovni model v CASE Studio

Pri izbiri SUPBja smo izbirali med Microsoft SQL Server in Microsoft Accessu, vendar smo se po krajšem premisleku odločili za SQL Server, saj Access ne podpira delo z več uporabniki naenkrat.

2.4 Struktura spletne strani

Spletno stran, smo poskušali narediti tako, da bi šolski dnevnik najboljše prestavili v elektronsko obliko. S strukturo programa smo omogočili, da bo spletni šolski dnevnik omogočal hitreje vpisovanje ur in manjkajočih dijakov. Omogočal bo tudi, urejanje in vpisovanje dijakov in profesorjev. Profesorji imajo hiter pregled nad njihovimi urniki in možnosti vpisovanje datumov testov.



Slika 4: Struktura spletne strani

V programu Visual Studio imamo takšno strukturo programa:

-App_Code: Ta mapa vsebuje kodo za povezavo z bazo, razrede za dostopanje do podatkovne baze, programsko kodo za prijavo in uporabnike.

-Strani: V tej mapi je večina strani, ki smo jih razdelil v skupine(profesor,ravnatelj,razrednik)

-Style: Tukaj je shranjena datoteka, v kateri je zapisana oblika naše spletne strani

-MasterPage.master- Naša glavna stran

2.5 Povezovanje z podatkovno bazo

```
public static class Global
{
    public static string ConnectionString
    {
        get { return @"Server=E05-12\SQLEXPRESS;Trusted_Connection=yes;database=DNEVNIK;connection timeout=30"; }
    }
}
```

Slika 5: Connection String

Za povezavo z SQL bazo smo ustvarili razred, ki smo ga poimenovali Global.cs. V njem smo ustarili dve lastnosti, tipa string, v katero smo shranili povezovalna stavka(angl. Connection string), za povezovanje z podatkovno bazo za dnevnik in podatkovno bazo z uporabniki.

Za lažje in hitrejše upravljanje z bazo, smo ustvarili t.i. »kontrolerje« v katerih smo že vse poizvedbe napisali v naprej, kar omogoča veliko bolj pregledno in lažje vpisovati podatke v podatkovno bazo in brati iz nje.

```
public static void DodajDijaka(Dijak d)
{

    int id = 0;
    SqlCommand ukaz = new SqlCommand("SELECT MAX(ID_DIJAKA) as ID_DIJAKA FROM DIJAK");
    SqlParameter p1 = new SqlParameter("@idDijak", System.Data.SqlDbType.Int);
    p1.Value = d.IdDijaka;
    ukaz.Parameters.Add(p1);
    SqlParameter p2 = new SqlParameter("@idOddelek", System.Data.SqlDbType.Int);
    p2.Value = d.IdOddelka;
    ukaz.Parameters.Add(p2);
    SqlParameter p3 = new SqlParameter("@ime", System.Data.SqlDbType.VarChar, 50);
    p3.Value = d.Ime;
    ukaz.Parameters.Add(p3);
    SqlParameter p4 = new SqlParameter("@priimek", System.Data.SqlDbType.VarChar, 50);
    p4.Value = d.Priimek;
    ukaz.Parameters.Add(p4);
    SqlParameter p5 = new SqlParameter("@datumRojstva", System.Data.SqlDbType.DateTime);
    p5.Value = d.DatumRojstva;
    ukaz.Parameters.Add(p5);
    SqlParameter p6 = new SqlParameter("@ulica", System.Data.SqlDbType.VarChar, 50);
    p6.Value = d.Ulica;
    ukaz.Parameters.Add(p6);
    SqlParameter p7 = new SqlParameter("@hisnaSt", System.Data.SqlDbType.VarChar, 5);
    p7.Value = d.HisnaStevilka;
    ukaz.Parameters.Add(p7);
    SqlParameter p8 = new SqlParameter("@idKraja", System.Data.SqlDbType.Int);
    p8.Value = d.IdKraja;
    ukaz.Parameters.Add(p8);
    SqlParameter p9 = new SqlParameter("@neopraviceneure", System.Data.SqlDbType.Int);
    p9.Value = d.NeopraviceneUre;
    ukaz.Parameters.Add(p9);
    SqlParameter p10 = new SqlParameter("@opraviceneure", System.Data.SqlDbType.Int);
    p10.Value = d.OpraviceneUre;
    ukaz.Parameters.Add(p10);
```

Slika 6: Del kode za metode za dodajanje dijaka

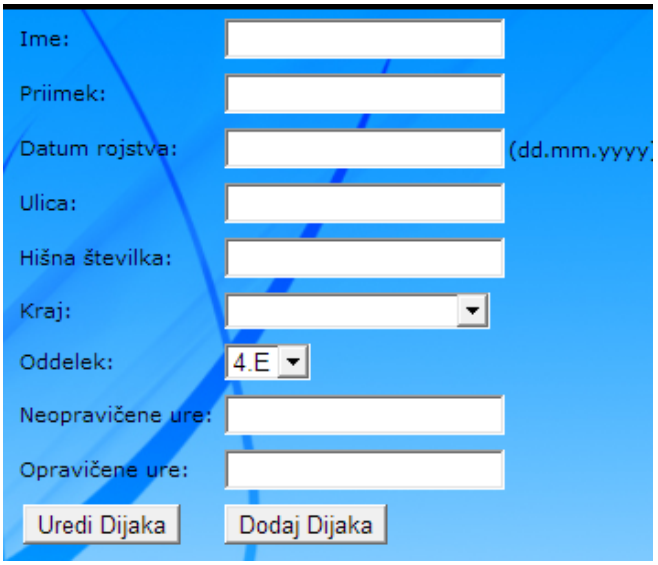
```
using (SqlConnection conn = new SqlConnection(Global.ConnectionString))
{
    ukaz.Connection = conn;
    try
    {
        conn.Open();
        using (SqlDataReader bralnik = ukaz.ExecuteReader())
        {
            while (bralnik.Read())
            {
                id = int.Parse(bralnik["ID_DIJAKA"].ToString());
            }
            p1.Value = id + 1;
        }
        ukaz.CommandText = "INSERT INTO DIJAK VALUES(@idDijak,@idOddelek,@ime,@priimek,@datumRojstva,@ulica,@hisnaSt,@idKraja,@neopraviceneUre,@opraviceneUre)";
        ukaz.ExecuteNonQuery();

    }
    catch { throw new Exception("Napaka pri povezovanju z bazo!"); }
}
```

Slika 7: Metoda za dodajanje dijaka v podatkovno bazo

Najprej ustvarimo parameter za vsak atribut v tabeli dijak, ki se kasneje vpiše v bazo. S tem procesom smo ustvarili nov primerek dijaka. Ime in priimek sta atributa tipa besedilo(string), datum rojstva se vpiše po vnosni maski »dd.mm.llll«. Kraj in oddelek se izbereta na izbirnem meniju(listbox), ki jih pa uporabnik ne vpisuje sam, ampak jih izbere. Opravičene ure in neopravičene na dodajanju dijaka ne vpišejo, vendar se uporabijo pri urejanju dijaka.

S pritiskom na gumb »Uredi Dijaka«, lahko razrednik ali ravnatelj popravi podatke o dijakov. Podatki se za razliko od dodajanja ne dodajo, ampak se posodobijo.

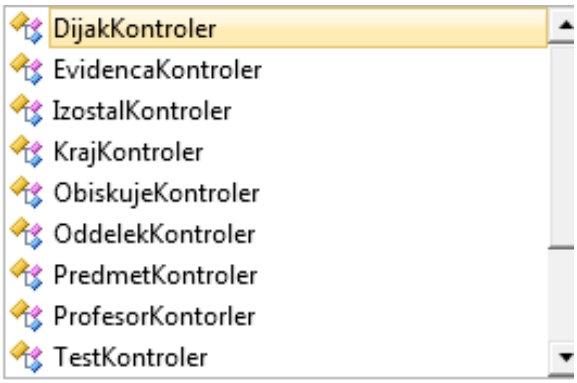
Spletna stran za dodajanje dijaka zgleda tako:



Slika 8: Obrazec za dodajanje dijaka

Dijaku v podatkovni bazi se vpišejo zgornji atributi. V programski kodi strani se preberejo oddelki iz podatkovne baze in kraji. Z pritiskom na gumb »Dodaj Dijaka« se sproži odzivna metoda ki dijaka preko ustvarjenih razredov vpiše v podatkovno bazo. Dijaka lahko v podatkovno bazo doda samo razrednik posameznega oddelka ali ravnatelj.

Vsi ustvarjeni razredi za lažje dostopanje do podatkovne baze:



Slika 9: Kontrolerji

Vsak razred(kontroler) vsebuje metode za vpisovanje in branje podatkov. Vsaka metoda vsebuje vstopne podatke kot so : id_kraja,ime_priimek dijaka… V glavnem programu tako lažje dostopamo do vseh podatkov ki so zapisani v podatkovni bazi. Do teh razredov lahko dostopamo brez da bi ustvarili nov objekt tega razreda.

2.6 Vpis ure

Torek	Sreda	Četrtek	Petek
/	/	/	/
/	/	/	/
/	/	/	/
/	/	/	/
/	/	/	/
/	/	/	STROKA-MODERNE VSEBINE 4 E E03
/	/	/	STROKA-MODERNE VSEBINE 4 E E03
/	/	/	NAPREDNA UPORABA PODATKOVNIH BAZ 4 E E07
/	/	/	/

Slika 10: Urnik za prijavljenega profesorja

Ko se uporabnik vpiše, se mu prikaže urnik. Urnik je izdelan s tabelo, ki vsebuje gumbe. Za vsak dan posebej je drugi gumb. Na urniku se izpiše katere ure ima profesor ki je prijavljen. Z pritiskom na izbrano uro, je uporabnik preusmerjen na stran v kateri lahko izbere manjkajoče dijake in vpiše kaj se je pri tisti uri dogajalo. Z pritiskom na gumb »Dodaj v evidenco« se vpiše točen datum in ura, tema in profesor. Na desni strani so izpisani dijaki, ki jih z klikom na polje za označitev(angl. CheckBox) izberemo in se vpišejo kot manjkajoči.

Razrednik lahko kasneje prikaže zgodovino zapisov in vsake dijaka ki je manjkal, lahko označi ali je manjkal opravičeno ali neopravičeno.

Slika 11: Vpis ure

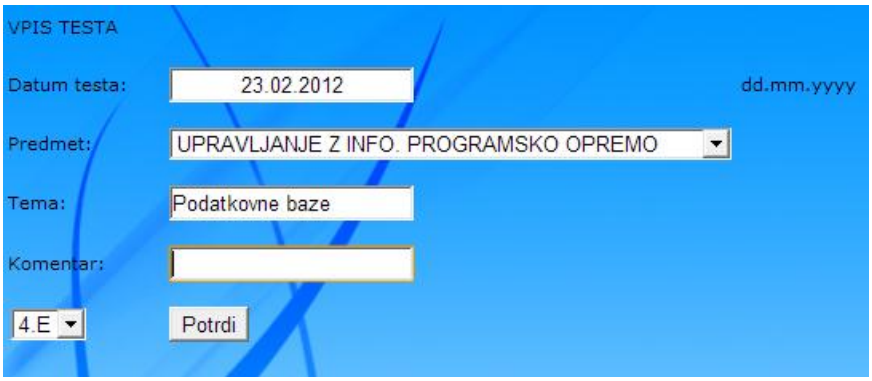
2.7 Evidenca pouka

Vsak razrednik ima vpogled v evidenco pouka. Pri vsaki uri se vidi kateri dijaki so bili manjkajoči, kaj je bila tema pri pouku in kdaj je bila ta ura opravljena. Zapisane vnose ni možno popravljati. Evidenca je razdeljena na razrede in profesorje. Z izborom razreda se prikaže celotna zgodovina vnosov razreda razvrščeno po tednih. Enak pogled v evidenco ima tudi profesor za svoje ure.

Poleg pregleda zgodovine pouka imajo tudi profesorji vpogled v ure vsakega predmeta. Razrednik pa ima možnost vpogleda v neopravičene in opravičene ure posameznega njegovega dijaka.

2.8 Vpis testa

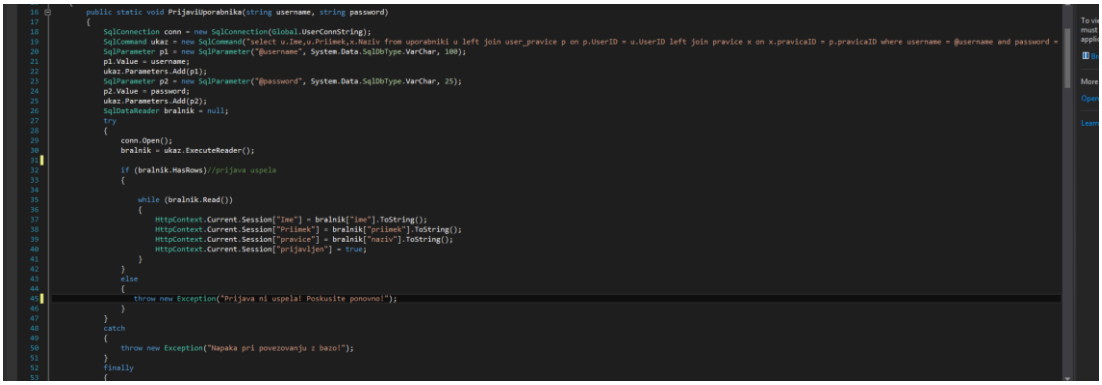
Na spletnemu dnevniku, lahko tudi profesor vpiše test. Določi datum testa, predmet izbere z seznama njegovih predmetov, nato pa lahko še napiše temo testa in komentar. Na koncu pa še izbere oddelek katerega uči in z gumbom »Potrdi« se v bazo vpiše test.



Slika 12: Obrazec za dodajanje testa

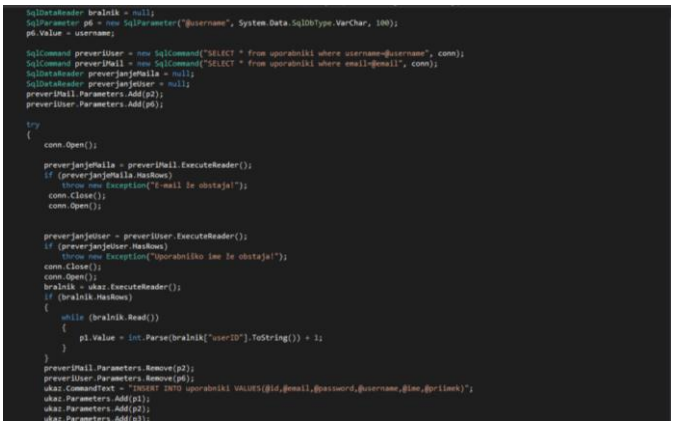
2.9 Varnost

Osnovna ideja je bila, da se profesor prijavi v sistem in mu izpiše urnik, s katerim kasneje doda uro. Najprej smo hoteli uporabiti vgrajen ASP.NET sistem za upravljanje z uporabniki, a smo se kasneje odločili da ustvarimo svojega, ker nam je ASPjev sistem povzročal kar veliko težav. Ustvarili smo preprosto podatkovno bazo s dvema osnovnima tabelama uporabniki in pravice. Povezali smo ju in nastala je nova vmesna tabela v kateri se vidi kakšno pravico ima kateri uporabnik. V programu smo ustvarili nov razred v katerem smo upravljali s uporabniki.



Slika 13: Metoda, ki prijavi uporabnika v sistem

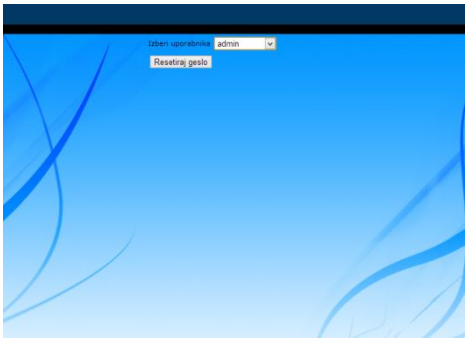
Ustvarili smo novo metodo, ki sprejme uporabniško ime in geslo, ter preveri če v bazi obstaja takšen uporabnik. Poizvedba združi tabelo uporabniki in pravice, tako da hkrati še vidimo kakšne pravice ima uporabnik. Za pravice smo se odločili, da bodo sestavljene hirahično npr. razrednik je hkrati tudi profesor, profesor pa ni nujno, da je razrednik. Če je prijava uspešna, se v trenutno sejo(angl. session) shrani ime uporabnika, priimek uporabnika, njegove pravice in pa ali je profesor prijavljen ali ne. Če je prijava neuspešna vržemo izjemo, v kateri povemo, da prijava ni uspela. Če pride do izjeme pri povezovanju z bazo vržemo novo izjemo v kateri povemo, da je prišlo do napake pri povezovanju z bazo. Odločili smo se, da bo lahko novega uporabnika ustvaril samo ravnatelj, ker bi lahko drugače prišlo do zlorabe. Za ustvarjanje uporabnika smo ustvarili novo metodo, ki kot vhodne podatke sprejme uporabniško ime,geslo, elektronsko pošto , ime in priimek. Potem smo napisali poizvedbo, ki vrne največji ID iz podatkovne baze. Preden smo vse skupaj dodali v podatkovno bazo smo z dvema poizvedbama preverili če obstaja v bazi enako uporabniško ime in če obstaja elektronski naslov. Če obstaja vržemo izjemo, v kateri povemo do kakšne napake je prišlo. Vse napake izpisujemo v zato namenjeni oznaki(angl. label).



Slika 14: Metoda, za ustvarjanje novega uporabnika

Ravnatelju smo tudi omogočili, da ponastavi geslo uporabniku(profesorju), če ga je ta slučajno pozabil. Ustvarili smo obrazec, na katerem je spustni seznam(angl. drop down list) v katerem so vsi uporabniki. Ravnatelj s pritiskom na gumb »Resetiraj« sproži odzivno metodo, ki najprej

pokliče generator gesla, potema pa izbranem uporabniku zamenja geslo. Novo geslo mu nato pošlje preko elektronskega naslova na uporabnikov elektronski naslov.



Slika 15: Obrazec za resetiranje gesla

3 Razprava

Med izdelavo smo odkrili kar nekaj hroščev in lukenj v že vgrajenem ASP.NET sistemu za administracijo spletne strani. Srečali smo se z več težavami pri prijavi sistem in nedelujočimi funkcijami npr. nikakor nam ni uspelo vzpostaviti profilov, katere smo hoteli uporabiti za shranjevanje dodatnih podatkov katere smo potrebovali za naš sistem.

V uvodu smo si zadali naslednje hipoteze:

Hiter, enostaven in prijazen uporabniški vmesnik, ki bo funkcionalen

Hitrost smo dosegli s optimizacijo programske kode. Za enostaven in prijazen uporabniški vmesnik je bistveno, da je pregleden. Zato smo ukaze smiselno razvrstili in določili funkcije ter ime ki ga upravljajo.

Omogočanje preproste prijave ter spreminjane, dodajanje in brisanje podatkov

Da smo zagotovili preprosto prijavo smo ustvarili svoj sistem za upravljanje z uporabniki, ki se odpira v svojem oknu. Da lahko uporabnik ob morebitnih spremembah ali napakah uredi podatke, pa smo dodali gumbe za dodajanje, brisanje in spreminjanje.

Enostavnejši pregled nad podatki

Kot je že vsem znano, je štetje ur(predmetov in izostankov) v sedanji obliki dnevnika zamudno, možnosti da pride do napake, pa je veliko. Problema smo se lotili s pomočjo poizvedb, kajti z njimi lahko na hiter in enostaven način obdelamo velike količine podatkov.

Lažje ustvarjanje zaključnih poročil

Z zbranimi podatki je omogočeno enostavnejše zaključevanje poročim, kajti vsi podatki (ure, izostanki) so prešteti in smiselno predstavljeni.

Ob končanju izdelka smo prišli do spoznanja, da bi lahko dnevniku dodali še redovalnico in bi tako ustvarili celovit sistem za upravljanje šole.

4 Zaključek

Potrdili smo vse naše hipoteze. Z izgledom izdelka smo zadovoljni, ker smo ustvarili preprosto spletno stran, kateri funkcionalnosti ne manjka. Uporabniku omogoča varno shranjevanje in spreminjanje podatkov, vpogled nad urnikom in vpisanimi urami.

Raziskovalne naloge smo se lotili z željo ustvariti sistem, ki bi lahko nadomestil šolski dnevnik. Ob delu smo spoznali, da je delo brez dobro izdelanega načrta skoraj nemogoče. Na začetku je bil naš načrt nepopoln, zato smo se morali vračati nazaj na prejšnje korake. Če bi bil načrt boljši teh težav nebi imeli.

5 Viri in literatura

- BONAČIĆ, D. Kratek priročnik jezika C# in razlike z jezikom C++. 1. Izd. Maribor: Fakulteteta za elektrotehniko, računalništvo in informatiko, Inštitut za informatiko, 2008
- Spletna stran w3school / nazadnje preverjena 7.3.2013

http://www.w3schools.com/html/default.asp

- Spletna stran CodeProject / nazadnje preverjena 6.3.2013

http://www.codeproject.com/Articles/32545/Exploring-Session-in-ASP-Net

- Spletna stran css3.info / nazadnje preverjena 7.3 . 2013

http://www.css3.info/preview/rounded-border/

- Spletna stran msdn.microsoft / nazadnje preverjena 5.3.2013

http://msdn.microsoft.com/sl-si/vstudio/

- Spletna strani sl.wikipedia.si / nazadnje preverjena 5.3.2013

http://sl.wikipedia.org/wiki/CSS

Zahvala

Radi bi se zahvalili našem profesorju in mentorju g. Dušanu Fugini za pomoč pri izdelavi raziskovalne naloge.

IZJAVA*

Mentor (-ica) ,_____, v skladu z 2. in 17. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, zagotavljam, da je v raziskovalni nalogi naslovom _____,
katere avtorji (-ice) so _____ _____, _____, _____:

- besedilo v tiskani in elektronski obliki istovetno,
- pri raziskovanju uporabljeno gradivo navedeno v seznamu uporabljene literature,
- da je za objavo fotografij v nalogi pridobljeno avtorjevo (-ičino) dovoljenje in je hranjeno v šolskem arhivu;
- da sme Osrednja knjižnica Celje objaviti raziskovalno nalogo v polnem besedilu na spletnih portalih z navedbo, da je nastala v okviru projekta Mladi za Celje,
- da je raziskovalno nalogo dovoljeno uporabiti za izobraževalne in raziskovalne namene s povzemanjem misli, idej, konceptov oziroma besedil iz naloge ob upoštevanju avtorstva in korektnem citiranju,
- da smo seznanjeni z razpisni pogoji projekta Mladi za Celje.

Celje, _____ žig šole Šola

Podpis mentorja(-ice)

Podpis odgovorne osebe

*** Pojasnilo**

V skladu z 2. in 17. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje je potrebno **podpisano izjavo mentorja(-ice) in odgovorne osebe šole uvezati v izvod za knjižnico**, dovoljenje za objavo avtorja(-ice) fotografskega gradiva, katerega ni avtor(-ica) raziskovalne naloge, pa hrani šola v svojem arhivu.