Šolski center Celje

Srednja šola za kemijo, elektrotehniko in računalništvo

Medplatformski sistem za urnike in nadomeščanja

Raziskovalna naloga

AVTORJI

David Šket

Jani Pezdevšek

Klemen Uršič

Mentor

mag. Boštjan Resinovič

Celje, 2014

Šolski center Celje

Srednja šola za kemijo, elektrotehniko in računalništvo

Medplatformski sistem za urnike in nadomeščanja

Raziskovalna naloga

Avtorji:

David Šket
Jani Pezdevšek
Klemen Uršič

Mentor:

mag. Boštjan Resinovič

Mestna občina Celje, Mladi za Celje

Celje, 2014

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 z 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 knjižničnih portalih z navedbo, da je raziskovalna naloga 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 razpisnimi pogoji projekta Mladi za Celje.

Celje, _____ žig šole 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 vključiti v izvod za knjižnico, dovoljenje za objavo avtorja(-ice) fotografskega gradiva, katerega ni avtor(-ica) raziskovalne naloge, pa hrani šola v svojem arhivu.

DOVOLJENJE ZA OBJAVO AVTORSKE FOTOGRAFIJE V RAZISKOVALNI NALOGI

Podpisani,_____ _____, izjavljam, da sem avtor(-ica) fotografskega gradiva, navedenega v priloženem seznamu, in dovoljujem v skladu z 2. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, da se lahko uporabi pri pripravi raziskovalne naloge pod mentorstvom _____ _____ z naslovom_____, katere avtorji (-ice) so _____ _____, _____ _____, _____ _____.

Dovoljujem tudi, da sme Osrednja knjižnica Celje vključeno fotografsko gradivo v raziskovalno nalogo objaviti na knjižničnih portalih z navedbo avtorstva v skladu s standardi bibliografske obdelave.

Celje, _____ Podpis avtorja:

Priloga: - seznam fotografskega gradiva

Kazalo

Kazalo slik

Povzetek

V raziskovalni nalogi smo se odločili poenostaviti in izboljšati dostop do urnikov in nadomeščanj, zato smo izdelali sistem, ki podatke shranjuje v podatkovni bazi na oddaljenem strežniku ter jih zna posredovati programom s pomočjo spletnega servisa. Za dostop do podatkov smo napisali okenski program za platformo Windows ter programa za mobilni platformi Android in Windows Phone. Pri tem smo uporabili medplatformski pristop, ki omogoča uporabo iste kode za vse zgoraj omenjene platforme.

Abstract

In our research project we have decided to simplify and improve the access to timetables and substitutions of teachers, therefore we created a system which saves data to a database located on a remote server. It is able to forward this data to programs with the use of a web service. To access the data, we wrote a program for the Windows platform and mobile platforms Android and Windows Phone. We used multi-platform approach that allows the use of the same code for all the above mentioned platforms.

Ključne besede

medplatformski razvoj, mobilna aplikacija, informacijski sistem, medplatformska ogrodja in knjižnice, podatkovna baza, integrirano razvojno okolje, C#, Xamarin, Android, Windows Phone

Keywords

cross-platform development, mobile application, information system, cross-platform frameworks and libraries, data base, integrated development environment, C#, Xamarin, Android, Windows Phone

1 Uvod

Pridobivanje informacij nam vse bolj olajšuje tehnika. Še pred štirimi leti, ko smo pričeli šolanje v srednji šoli, smo urnike dobili na papirju. Letos, v četrtem letniku, imamo dostop do informacij o urnikih in nadomeščanjih na šolski spletni strani.

Način pridobivanja informacij je eden glavnih problemov moderne družbe, kjer se dostop ne šteje več v urah in minutah, ampak v sekundah. In problem pridobivanja podatkov je glavni razlog za nastanek te raziskovalne naloge. Naš namen je ustvariti aplikacije za različne platforme, ki bi dijakom, profesorjem in ostalim uslužbencem šole poenostavile dostop do urnikov in nadomeščanj.

1.1 Predstavitev problema

Urniki in nadomeščanja, ki jih uporabljajo dijaki in profesorji, so na večini izobraževalnih centrov dostopni na različne načine. Največkrat so to spletne strani izobraževalnih centrov, šolski radio, socialno omrežje Facebook ter okrožnice, ki so najpogosteje v papirnati obliki. Odločili smo se, da bomo ustvarili sistem, s pomočjo katerega bomo uporabnikom olajšali dostop do teh informacij. Ker smo tudi sami dijaki, nam je ta problem poznan, prav tako pa smo potrebo takšnega sistema z metodo spraševanja raziskali pri ostalih uporabnikih.

To nam je pri raziskovanju olajšalo delo, saj smo lahko razbrali, kaj je uporabnikom najbližje. S pomočjo vprašalnika smo izvedeli, da ima največ uporabnikov mobilne telefone s sistemom Android, Windows Phone in iOS. Več kot 80 % anketirancev se je odločilo, da bi jim bilo lažje, če bi lahko urnike in nadomeščanja preverili kar na mobilnem telefonu ali računalniku.

1.2 Hipoteze

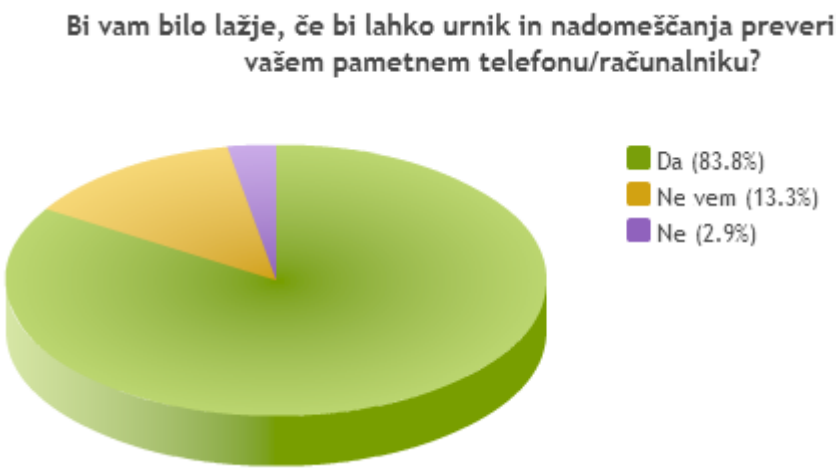
Postavili smo naslednje hipoteze:

- Sistem bo uporabnikom olajšal vsakodnevni dostop do potrebnih informacij o urnikih in nadomeščanjih.

- Sistem bo omogočal vpogled v urnike in nadomeščanja na platformah Windows, Android in Windows Phone.
- Pri programiranju bo mogoče velik del kode napisati po principu medplatformskega razvoja programov, kar nam bo količino dela znatno zmanjšalo, saj bo mogoče velik del kode brez popravkov uporabiti na vseh zgoraj naštetih platformah.

1.3 Opis raziskovalnih metod

Preden smo se lotili načrtovanja in izdelave, nas je seveda zanimalo, kakšne so potrebe in želje uporabnikov na tem področju. Zanimalo nas je, na kakšen način je to urejeno na drugih šolah, ali bi se uporabnikom takšen sistem zdel smiseln in ali bi ga uporabljali. Prav tako smo se posvetovali z našimi profesorji, ki so bili navdušeni nad našo idejo. Ker se je večina anketirancev odločila, da bi takšen sistem uporabljali, smo poiskali potrebno gradivo in ideje za zasnovo sistema.



Slika 1: Rezultati ankete

Obsežen del raziskovalne naloge je bila študija medplatformskega principa razvijanja informacijskih sistemov, saj se ta precej razlikuje od navadnega. Gradiva, ki smo jih našli na internetu, so bila v večini napisana v angleškem jeziku, kar je naše delo še dodatno otežilo. Nepoznavanje principa medplatformskega razvijanja je bil tudi razlog za večkratno popravljanje načrtov, saj se je pogosto zgodilo, da določena platforma enostavno ni podpirala določene funkcije, kar nas je ponovno vrnilo na sam začetek.

Pri analizi smo se zanašali na naše znanje o podatkovnih bazah in programiranju, predvsem pa smo hoteli sistem prilagoditi potrebam in željam uporabnikov.

2 Medplatformski razvoj aplikacij

Razvoj aplikacij je dolgotrajen proces, ki se začne z analizo problema, konča pa se takrat, ko se aplikacija zaradi določenega razloga ne uporablja več. Ker pa se zadnje čase zelo hitro razvija trg pametnih mobilnih telefonov, se hkrati veča povpraševanje po aplikacijah za njihove sisteme. Ker je teh sistemov več, se prav tako razlikujejo načini razvoja aplikacij. Zaradi tega se morajo razvijalci učiti novih programskih jezikov, postopkov izdelave in podobno. Spreminjajo se tudi potrebe na tržišču in delež operacijskih sistemov na trgu. Ker pa je morebitno povečanje števila razvijalcev lahko za podjetje velik finančni zalogaj, so potrebe po alternativi velike. S tem namenom se hkrati z mobilnimi operacijskimi sistemi razvijajo medplatformska ogrodja za razvoj mobilnih aplikacij, ki nam omogočajo razvoj aplikacij za več platform naenkrat. Teh ogrodij je na trgu iz dneva v dan več in čeprav so po značilnostih različna, je delo z njimi enostavnejše in cenejše tako za programerja kot za podjetje.

2.1 Medplatformska ogrodja

Poznamo veliko medplatformskih ogrodij. Med drugimi Rhodes, PhoneGap, MoSync, Appcelerator Titanium itd. Ker je o slednjih že bilo veliko napisano, smo se odločili za uporabo ogrodja Xamarin. Podjetje Xamarin je bilo ustanovljeno maja leta 2011. Ustanovili so ga razvijalci Mono, MonoTouch in Mono for Android medplatformskih ogrodij, ki temeljijo na Microsoft .NET platformi.

Programski paket Xamarin nam omogoča medplatformsko razvijanje aplikacij za sisteme Android in iOS v programskem jeziku C#, uporabljamo pa lahko Visual Studio ali Xamarin Studio, ki ga dobimo zraven programa. Brezplačna verzija nam sicer omogoča le uporabo slednjega, prav tako pa ni mogoče razvijanje aplikacij za iOS, kar je tudi razlog, zakaj se nismo odločili za izdelavo programa za platformo iOS.

2.2 Prednosti medplatformskega razvijanja aplikacij

Največja prednost medplatformskega razvijanja aplikacij je uporaba enega programskega jezika za več platform. Učenje novih programskih jezikov je zapleteno in zamudno. Medplatformsko razvijanje aplikacij nam prav tako omogoča, da del kode, ki se ponavlja pri vseh platformah, napišemo samo enkrat (npr. v obliki knjižnice) in uporabimo večkrat na različnih platformah. Podpora in dokumentacija se posodablja iz dneva v dan, razvijalci orodij nam zagotavljajo, da lahko vse, kar lahko naredimo z izvornim programskim okoljem za neko platformo, naredimo tudi v medplatformskem.

Slika 2 nam prikazuje shemo sistema za več platform, narejenega z uporabo medplatformskih ogrodij.

Slika 2: Shema sistema, narejenega z uporabo medplatformskih ogrodij. (Vir:xamarin.com)

2.3 Vrste mobilnih aplikacij, izdelanih z medplatformskimi ogrodji

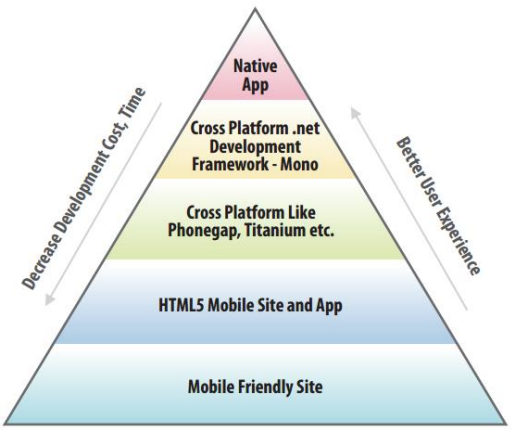
Poznamo več vrst mobilnih aplikacij, izdelanih z medplatformskimi ogrodji. Najenostavnejše za izdelavo so spletne strani, prilagojene za mobilne naprave. Vendar pa je slabost takšnih aplikacij nezmožnost dostopa do izvornih funkcionalnosti naprave, kot so na primer kamera, senzorji gibanja ipd.

Druga vrsta mobilnih aplikacij so spletne aplikacije (»web apps«). Takšne aplikacije podpirajo nekatere funkcionalnosti naprave, vendar so razvijalci zelo omejeni pri razvoju. Odvisni so med drugim tudi od podpore brskalnika in hitrosti interneta.

Tretja vrsta so hibridne aplikacije (»hybrid apps«). So mešanica izvornih in spletnih aplikacij. Velika prednost te vrste aplikacij je podpora prednosti obeh vrst. Aplikacija je videti kot izvorna aplikacija, tako da jo lahko ponudimo na spletnih trgovinah, vendar je za delovanje vseeno potrebna internetna povezava. Osnovo za aplikacijo moramo narediti za vsako platformo posebej.

Medplatformske aplikacije nam omogočajo hiter razvoj in podpirajo večino funkcij mobilne naprave. Ne potrebujemo razvojnih okolij za vsako platformo, vendar večina orodij podpira le Android, iOS in Windows Phone.

Zadnja vrsta mobilnih aplikacij pa so aplikacije, izdelane z .NET ogrodji. To nam omogoča izdelavo aplikacij z močnim razvojnim okoljem Visual Studio in veliko podporo, vendar pa so ogrodja večinoma plačljiva *(povzeto po M. Dimic, Medplatformski razvoj aplikacij).*



Slika 3: Ogrodja glede na uporabniške izkušnje in stroške ter čas razvoja aplikacij. (Vir:alliancetek.com)

3 Razvoj informacijskega sistema

3.1 Koraki dela

Razvoj informacijskega sistema je potekal po naslednjih korakih:

1. analiza problema
2. načrtovanje:
    - izbira okolij in programskih jezikov za delo
    - načrtovanje podatkovne baze
    - načrtovanje spletnega servisa
    - načrtovanje knjižnice
    - načrtovanje okenske aplikacije
    - načrtovanje spletne strani
    - načrtovanje Android aplikacije
    - načrtovanje Windows Phone aplikacije
3. izdelava podatkovne baze
4. izdelava spletnega servisa
5. izdelava Android aplikacije
6. izdelava Windows Phone aplikacije

3.2 Analiza problema

Analizirali smo naslednja vprašanja:

- Katere podatke potrebujemo?
- Kaj je za končne uporabnike najboljše in najenostavnejše za uporabo?
- Kaj omogoča Android aplikacija, kakšne so možnosti razvoja s programsko opremo Xamarin?
- Kaj omogoča Windows Phone aplikacija, kakšna je razširjenost Windows Phone naprav, katero programsko opremo potrebujemo za razvoj aplikacij za različne verzije Windows Phone sistemov?
- Kakšne so prednosti in slabosti medplatformskega razvijanja aplikacij?

3.3 Načrtovanje

3.3.1 Izbira programskih jezikov in razvojnih okolij

Na voljo smo imeli več programskih jezikov: Java, C#, PHP, MSSQL, MySQL …

Na podlagi analize različnih programskih jezikov smo se odločili, da bomo za izdelavo podatkovne baze uporabili programski jezik SQL in sistem za upravljanje podatkovnih baz MySQL, predvsem zaradi boljše podpore. Aplikacijo, namenjeno za osebne računalnike, smo se odločili izdelati s pomočjo programskega paketa Visual Studio in s programskim jezikom C#. Prav tako smo za izdelavo spletnega portala uporabili Visual Studio in .NET platformo. Tudi pri izdelavi Windows Phone aplikacije smo se odločili za programski paket Visual Studio z Windows Phone SDK in programski jezik C#. Za izdelavo android aplikacije pa smo se odločili za programski paket Xamarin, saj nam je omogočal uporabo programskega jezika C# in enake knjižnice za dostop do podatkov kot v ostalih aplikacijah.

3.3.2 Načrtovanje podatkovne baze

Načrtovanja smo se lotili s pristopom od zgoraj navzdol (top-down), kar pomeni, da smo najprej preverili, katere informacije nam bo posredoval program, nato pa smo analizirali, katere podatke bomo za to potrebovali. Nato smo se lotili izdelave E-R diagrama, kjer smo dodali entitete, atribute, primarne ključe in povezave med tabelami. Ta načrt smo pretvorili v logični model in ga izdelali v programu Toad Data Modeler. Program nam je avtomatsko generiral kodo za izdelavo MySQL podatkovne baze.

3.3.3 Načrtovanje spletnega servisa

Ker se podatkovna baza nahaja na oddaljenem strežniku, smo za dostop do podatkov potrebovali spletni servis. Odločili smo se za spletni servis .NET. Pri izdelavi smo si morali pogledati omejitve različnih platform. Na koncu smo se odločili za izdelavo 13-ih metod. Za zaščito smo poskrbeli z izdelavo uporabniških računov z različnimi pravicami pri podatkovni bazi, tako da za spletni servis nismo naredili posebej avtentifikacije.

3.3.4 Načrtovanje knjižnice

Najpomembnejši del naše naloge je skupni del kode, ki smo jo pozneje dodali projektom v obliki knjižnice. Za izdelavo te smo najprej poiskali karakteristike vsake platforme, na katere moramo biti pozorni pri izdelavi. Potem smo naredili okvirni načrt, katere metode bomo potrebovali, nato pa vsako posebej razvili. Najprej smo knjižnico povezali s spletnim servisom, podatke prebrali, jih shranili v za to potrebne spremenljivke in jih pretvorili v obliko, ki smo jo pozneje potrebovali v aplikacijah.

3.3.5 Načrtovanje okenske aplikacije

Ko smo končali s knjižnico, smo začeli z načrtovanjem aplikacij. Delo smo si razdelili, tako da je vsak naredil eno aplikacijo. Pri načrtu za okensko aplikacijo smo najprej naredili prototip, da smo preverili pravilno delovanje ostalih komponent. Potem smo naredili načrt za grafični videz aplikacije ter določili funkcije, ki jih bo imela končna aplikacija.

3.3.6 Načrtovanje spletne strani

Pri izdelavi spletne strani smo se odločili, da bomo najprej naredili čisto preprosto spletno stran, ki bo podobna okenski aplikaciji in bo uporabnikom lažja za uporabo.

3.3.7 Načrtovanje Windows Phone aplikacije

Za Windows Phone aplikacijo smo najprej fizično naredili načrt za grafični vmesnik aplikacije. Ta je zaradi drugačne platforme izgledala malce drugačna kot ostale aplikacije. Za razvojno okolje smo izbrali Visual Studio z Windows Phone SDK-jem.

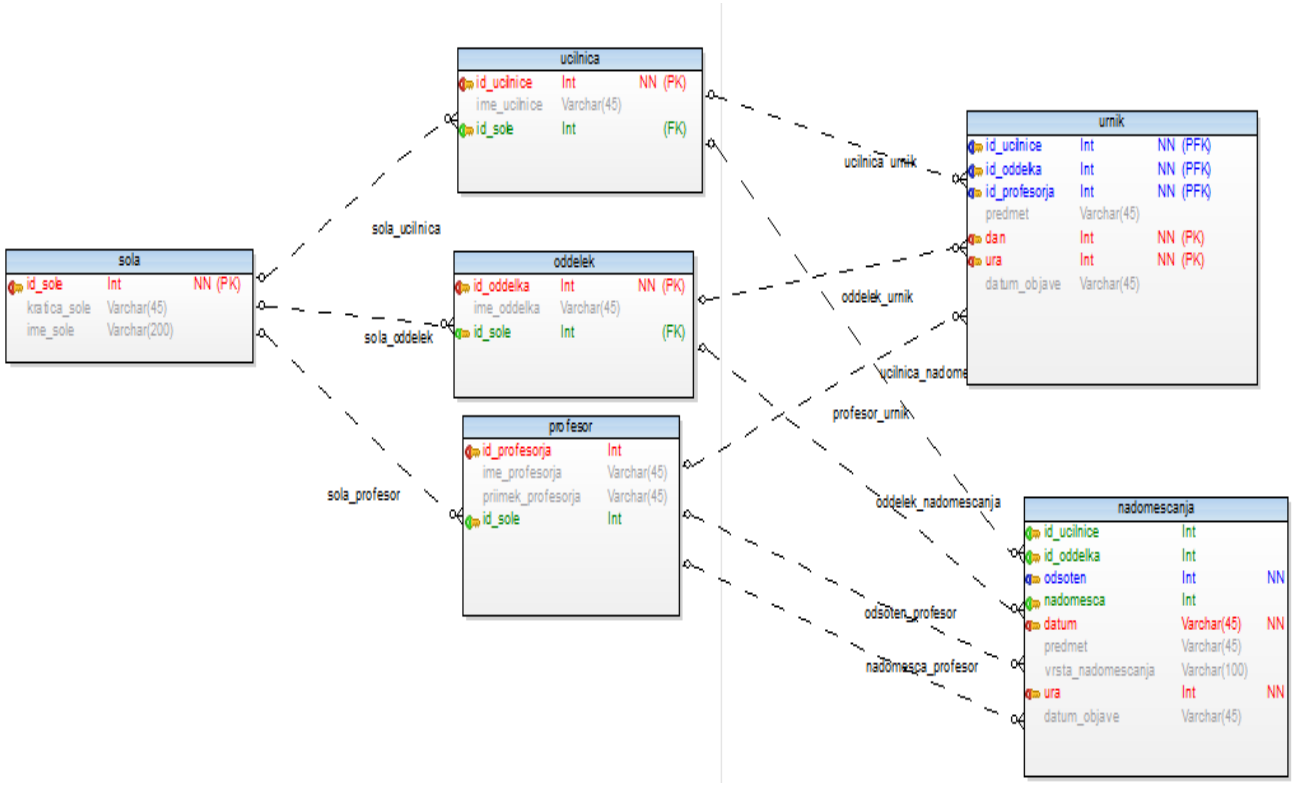
3.3.8 Načrtovanje Android aplikacije

Pri izdelavi Android aplikacije smo izdelali načrt, kako bo izgledala sama aplikacija in kaj naj bi omogočala uporabnikom. Ko smo ta načrt izdelali, smo se lotili izdelave v Xamarin Studiu.

Naša aplikacija bo vsebovala urnike za vse učitelje, dijake in za učilnice. Prav tako bomo prikazovali vsa nadomeščanja za 2 dni, starejša nadomeščanja bodo avtomatsko izbrisana.

3.4 Izdelava podatkovne baze

Podatkovna baza predstavlja temelje našega sistema, zato je bilo potrebno poskrbeti za čim boljši načrt. Da bo osnova čim boljša, smo poskrbeli s skrbno zastavljenimi vprašanji in normalizacijo. Odločili smo se za 6 tabel s podatki.



Slika 4: Model podatkovne baze, izdelan s programom Toad Data Modeler.

Ko smo končali z načrtovanjem, smo s pomočjo programa Toad Data Modeler izdelali logični model podatkovne baze. Program nam je avtomatsko generiral SQL kodo za izdelavo podatkovne baze na MySQL strežniku. Podatke, ki smo jih dobili od Šolskega centra Celje, smo nato s pomočjo posebnega programa vpisali v bazo. Podatke smo dobili v .txt datoteki. Program smo naredili tako, da je iz datoteke prebral vrstico, jo razstavil, shranil potrebne podatke v bazo in to ponavljal, dokler ni prišel do konca datoteke. Za vpis podatkov o nadomeščanjih smo naredili poseben okenski program, ampak več o tem pozneje. Na strežniku je bilo potrebno dodeliti pravice uporabnikom. Ker bo večina uporabnikov do

informacij le dostopala, smo naredili uporabnika »userUrnik«, ki ima pravice le za dostop do podatkov.

3.5 Izdelava spletnega servisa

Spletni servis je posebna spletna aplikacija, ki jo potrebujemo zaradi varnosti. Pošiljanje podatkov o bazi po internetu ni varno, zato je bolje, če se z aplikacijo povežemo na spletni servis, ta pa je lokalno povezan s podatkovno bazo.

Izbrali smo spletni servis za .NET, na katerem je 13 metod za dostop do podatkov podatkovne baze. Vsaka metoda vrne podatke v formatu XML. Servis smo naložili na IIS strežnik, ki smo ga konfigurirali glede na zahteve servisa in pozneje še spletne strani.

```
private string connectionString = @"server=localhost;database=urnikscc;userid=userUrnik;password=userUrnik123;charset=utf8;";
private MySqlConnection connection;
private MySqlCommand command;
private MySqlDataReader reader;
string query;

[WebMethod]
0 references
public DataTable UrnikOddelek(string ime_oddelka)
{
    connection = new MySqlConnection(connectionString);
    query = "SELECT predmet,ime_ucilnice,ime_profesorja,priimek_profesorja, dan,ura FROM urnik,ucilnica,oddelek,profesor WHERE
    command = new MySqlCommand(query,connection);
    connection.Open();
    reader = command.ExecuteReader();
    DataTable dt1 = new DataTable();
    dt1.TableName = ime_oddelka;
    dt1.Load(reader);
    connection.Close();
    return dt1;
}
```

Slika 5: Del programske kode s spletnega servisa.

```
▼<e1b diffgr:id="e1b1" msdata:rowOrder="0">
   <predmet>ANG</predmet>
   <ime_ucilnice>E07</ime_ucilnice>
   <ime_profesorja>Tjaša</ime_profesorja>
   <priimek_profesorja>Ogrizek</priimek_profesorja>
   <dan>1</dan>
   <ura>6</ura>
  </e1b>
▼<e1b diffgr:id="e1b2" msdata:rowOrder="1">
   <predmet>SLO</predmet>
   <ime_ucilnice>D02A</ime_ucilnice>
   <ime_profesorja>Andreja</ime_profesorja>
   <priimek_profesorja>Tkalec</priimek_profesorja>
   <dan>1</dan>
   <ura>3</ura>
  </e1b>
▼<e1b diffgr:id="e1b3" msdata:rowOrder="2">
   <predmet>RU</predmet>
   <ime_ucilnice>E15</ime_ucilnice>
   <ime_profesorja>Dušan</ime_profesorja>
   <priimek_profesorja>Bombač</priimek_profesorja>
   <dan>1</dan>
   <ura>7</ura>
  </e1b>
▼<e1b diffgr:id="e1b4" msdata:rowOrder="3">
   <predmet>MAT</predmet>
   <ime_ucilnice>E16</ime_ucilnice>
   <ime_profesorja>Nataša</ime_profesorja>
   <priimek_profesorja>Besednjak</priimek_profesorja>
   <dan>1</dan>
   <ura>8</ura>
  </e1b>
▼<e1b diffgr:id="e1b5" msdata:rowOrder="4">
   <predmet>IEK</predmet>
   <ime_ucilnice>D02A</ime_ucilnice>
   <ime_profesorja>Bojan</ime_profesorja>
   <priimek_profesorja>Šuster</priimek_profesorja>
   <dan>1</dan>
   <ura>9</ura>
  </e1b>
▼<e1b diffgr:id="e1b6" msdata:rowOrder="5">
   <predmet>ITKv/7</predmet>
   <ime_ucilnice>E05</ime_ucilnice>
   <ime_profesorja>Vojko</ime_profesorja>
   <priimek_profesorja>Podkrajšek</priimek_profesorja>
   <dan>1</dan>
   <ura>1</ura>
  </e1b>
```


Slika 6: Del XML datoteke, ki jo vrne spletni servis pri uporabi metode UrnikOddelka za oddelek E1B.

3.6 Izdelava knjižnice

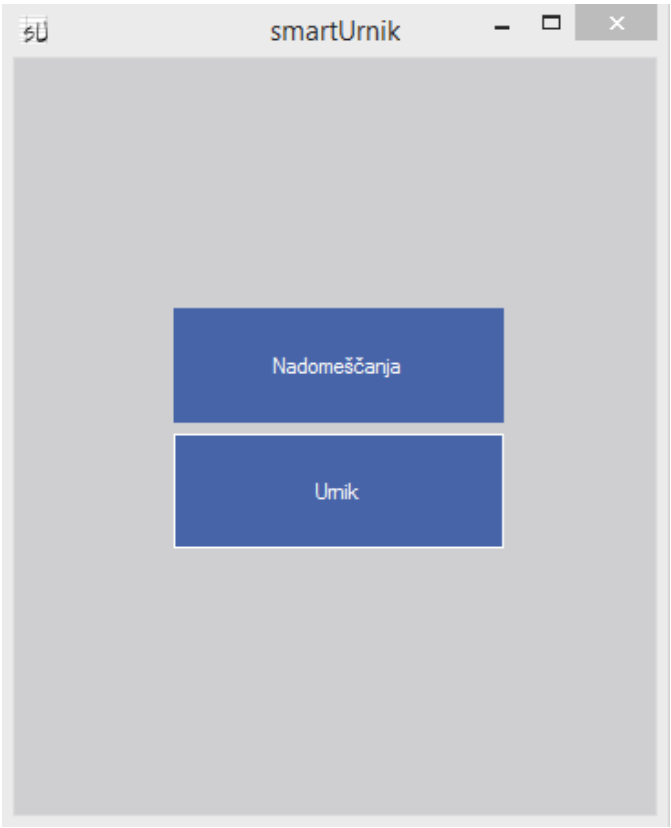
Kot že omenjeno, smo knjižnico ustvarili kot jedro vsem aplikacijam. Knjižnico smo izdelali v programskem jeziku C# v Visual Studiu, saj nam je ta najbolj poznan in ga lahko uporabimo v vseh aplikacijah. Načrtovanje skupnega dela programske kode je zahtevno ravno zaradi omejitev različnih platform. Visual Studio nam zato ponuja posebno obliko knjižnice imenovano »Portable Class Library«, ki nam omeji možnosti glede na izbrane platforme. Zaradi teh omejitev, pa tudi z namenom, da razbremenimo mobilne telefone, smo se odločili za malo drugačen dostop k problemu. Najprej smo ustvarili navadno knjižnico, vključili metode, ki jih potrebujemo za delovanje in jih prilagodili omejitvam različnih platform. Nato

smo to isto knjižnico pretvorili v spletno aplikacijo in jo naložili na IIS strežnik. Kasneje smo morali knjižnico še malo popraviti, saj smo pri načrtovanju izpustili nekaj podrobnosti.

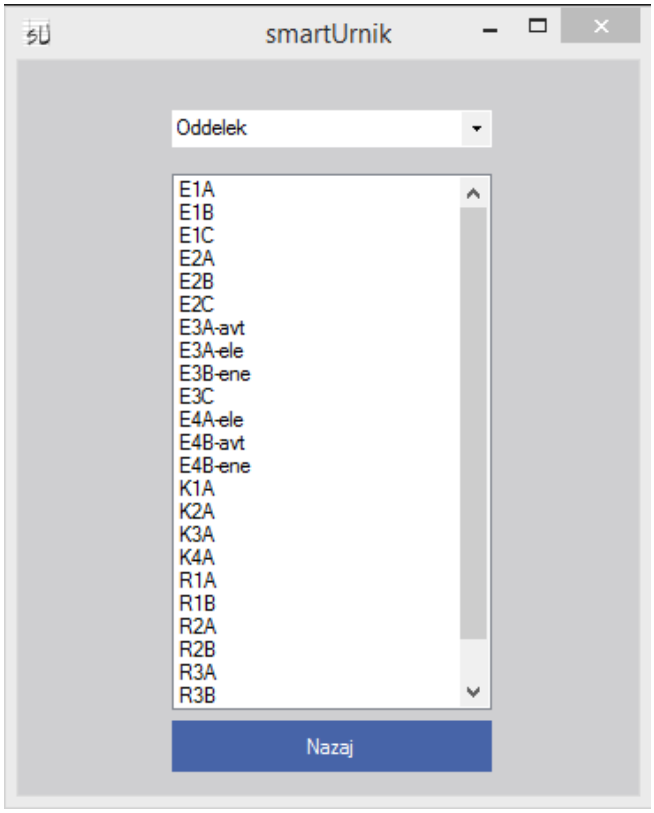
3.7 Izdelava okenske aplikacije

Pri izdelavi le-te smo sledili načrtu, ki smo ga pripravili na začetku. Na začetku smo v aplikacijo vključili knjižnico za dostop do podatkov. Nato smo v prvem oknu dodali objekt za izbiro urnika (listview) ter gumba za izpis urnika in nadomeščanj. Na drugo okno smo dodali mrežo za izpis urnika (datagridview) in gumb za nazaj. Enako smo naredili še s tretjim oknom. Ker smo znanje za izdelavo okenske aplikacije pridobili že v šoli, s tem nismo imeli težav.

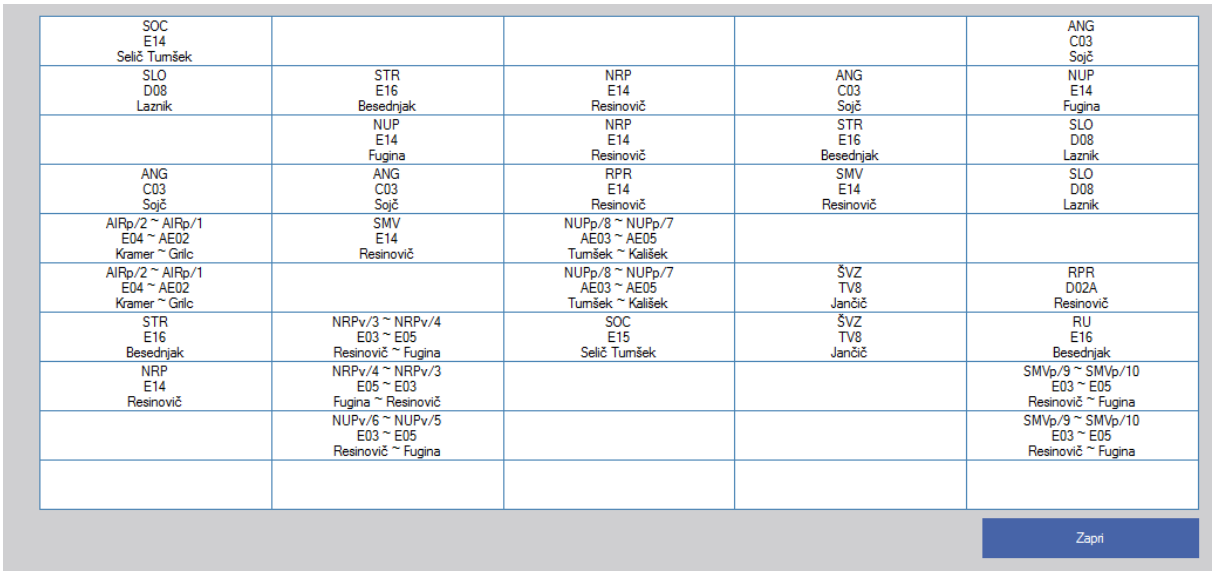
Slike 7, 8 in 9 prikazujejo delovanje okenske aplikacije.



Slika 7: Prvo okno okenske aplikacije.



Slika 8: Drugo okno okenske aplikacije.



SOC E14 Selič Tumšek				ANG C03 Sojč
SLO D08 Laznik	STR E16 Besednjak	NRP E14 Resinovič	ANG C03 Sojč	NUP E14 Fugina
	NUP E14 Fugina	NRP E14 Resinovič	STR E16 Besednjak	SLO D08 Laznik
ANG C03 Sojč	ANG C03 Sojč	RPR E14 Resinovič	SMV E14 Resinovič	SLO D08 Laznik
AIRp/2 ~ AIRp/1 E04 ~ AE02 Kramer ~ Grilc	SMV E14 Resinovič	NUPp/8 ~ NUPp/7 AE03 ~ AE05 Tumšek ~ Kališek		
AIRp/2 ~ AIRp/1 E04 ~ AE02 Kramer ~ Grilc		NUPp/8 ~ NUPp/7 AE03 ~ AE05 Tumšek ~ Kališek	ŠVZ TV8 Jančič	RPR D02A Resinovič
STR E16 Besednjak	NRPv/3 ~ NRPv/4 E03 ~ E05 Resinovič ~ Fugina	SOC E15 Selič Tumšek	ŠVZ TV8 Jančič	RU E16 Besednjak
NRP E14 Resinovič	NRPv/4 ~ NRPv/3 E05 ~ E03 Fugina ~ Resinovič			SMVp/9 ~ SMVp/10 E03 ~ E05 Resinovič ~ Fugina
	NUPv/6 ~ NUPv/5 E03 ~ E05 Resinovič ~ Fugina			SMVp/9 ~ SMVp/10 E03 ~ E05 Resinovič ~ Fugina

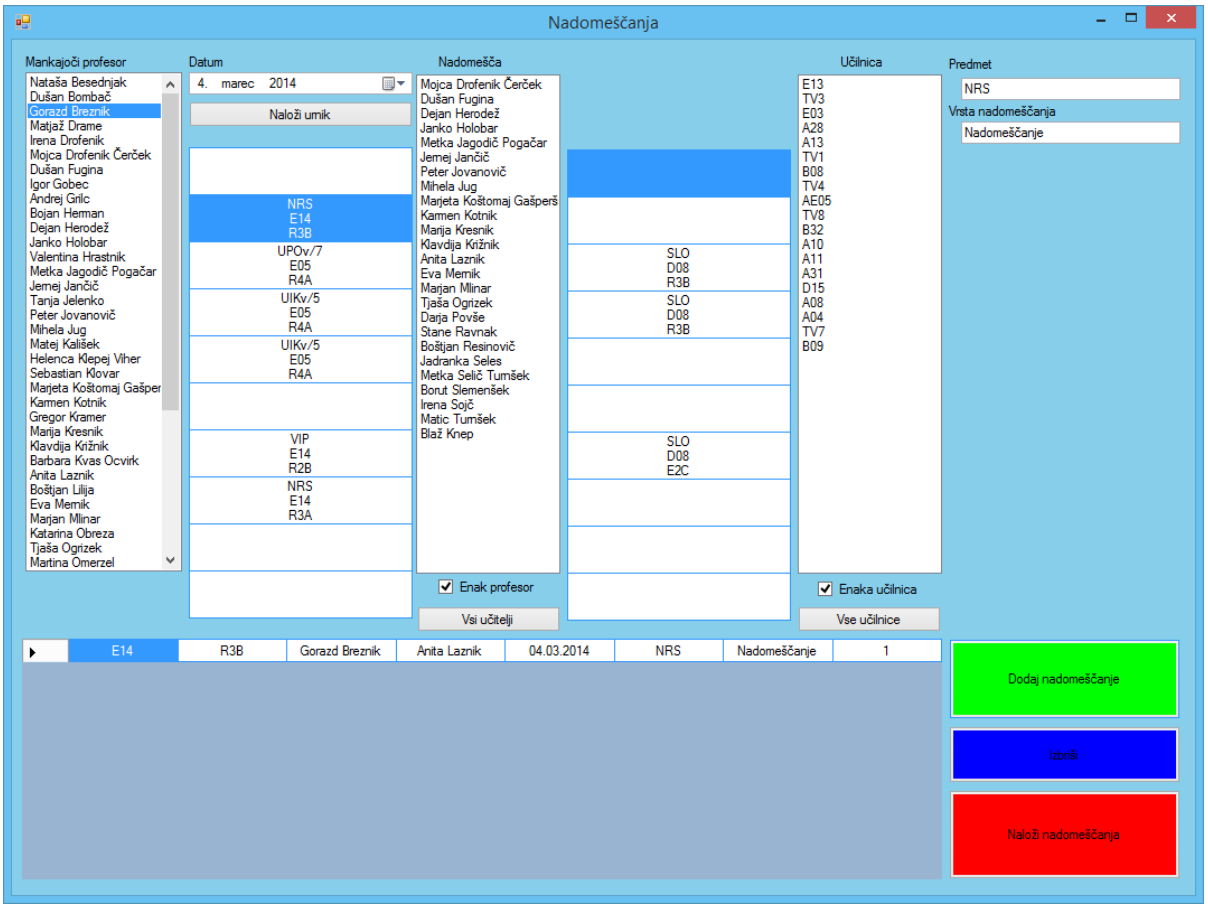
Zapri

Slika 9: Tretje okno okenske aplikacije.

3.8 Izdelava okenske aplikacije za vpis nadomeščanj

Najprej smo izdelali uporabniški vmesnik celotne aplikacije. Ko smo to naredili, smo se lotili samega delovanja aplikacije in za pomoč prosili profesorja, ki je na naši šoli odgovoren za vpis nadomeščanj. Izvedeli smo, katere lastnosti so dobre pri programu, ki ga uporabljajo zdaj, in katere bi bilo potrebno izboljšati. Aplikacija omogoča vpisovanje nadomeščanj v podatkovno bazo. Podatke vpisujemo v pravilnem vrstnem redu. Izdelali smo jo tako, da je enostavna za uporabnika in lahko obvladljiva, brez kakršnegakoli programerskega znanja.

Slika 10 prikazuje videz okenske aplikacije za vpis nadomeščanj.



Slika 10: Videz okenske aplikacije za vpis nadomeščanj.

3.8.1 Delovanje okenske aplikacije za vpis nadomeščanj

Najprej izberemo želenega profesorja in datum odsotnosti. Prikaže se nam njegov urnik za izbrani dan. S pritiskom na želeno uro, se nam izpišejo profesorji, ki so to uro prosti . Ko izberemo želenega profesorja, se hkrati prikaže tudi njegov urnik za ta dan. Program nam omogoča tudi zamenjavo učilnice, predmeta in določitev vrste nadomeščanja. Ko vse to izberemo, kliknemo gumb dodaj nadomeščanje. Ko dodamo vsa nadomeščanja za določen dan, s klikom na gumb »Naloži nadomeščanja« vsa nadomeščanja vpišemo v podatkovno bazo.

3.9 Izdelava spletne strani

Izdelali smo preprosto spletno stran, na kateri lahko izberemo oddelek, profesorja ali učilnico. Kot v vseh ostalih aplikacijah smo tudi za spletno stran uporabili enako knjižnico.

Ponedeljek	Torek	Sreda	Četrtek	Petek
SOC E14 Selič Turnšek				ANG C03 Sojč
SLO D08 Laznik	STR E16 Besednjak	NRP E14 Resinovič	ANG C03 Sojč	NUP E14 Fugina
	NUP E14 Fugina	NRP E14 Resinovič	STR E16 Besednjak	SLO D08 Laznik
ANG C03 Sojč	ANG C03 Sojč	RPR E14 Resinovič	SMV E14 Resinovič	SLO D08 Laznik
AIRp/2 ~ AIRp/1 E04 ~ AE02 Kramer ~ Grilc	SMV E14 Resinovič	NUPp/8 ~ NUPp/7 AE03 ~ AE05 Turnšek ~ Kališek		
AIRp/2 ~ AIRp/1 E04 ~ AE02 Kramer ~ Grilc		NUPp/8 ~ NUPp/7 AE03 ~ AE05 Turnšek ~ Kališek	ŠVZ TV8 Jančič	RPR D02A Resinovič
STR E16 Besednjak	NRPv/3 ~ NRPv/4 E03 ~ E05 Resinovič ~ Fugina	SOC E15 Selič Turnšek	ŠVZ TV8 Jančič	RU E16 Besednjak
NRP E14 Resinovič	NRPv/4 ~ NRPv/3 E05 ~ E03 Fugina ~ Resinovič			SMVp/9 ~ SMVp/10 E03 ~ E05 Resinovič ~ Fugina
	NUPv/6 ~ NUPv/5 E03 ~ E05 Resinovič ~ Fugina			SMVp/9 ~ SMVp/10 E03 ~ E05 Resinovič ~ Fugina

Oddelki

E1C
E2A
E2B
E2C
E3A-avt
E3A-ele
E3B-ene
E3C
E4A-ele
E4B-avt
E4B-ene
K1A
K2A
K3A
K4A
R1A
R1B
R2A
R2B
R3A
R3B

Slika 11: Videz spletne strani.

3.10 Izdelava Windows Phone aplikacije

Windows Phone je tretji na lestvici najbolj priljubljenih operacijskih sistemov med uporabniki, a postaja vedno bolj priljubljen.

Aplikacijo smo izdelali v Visual Studiu 2010 z Windows Phone SDK. Ker smo bili s tem okoljem že dokaj seznanjeni zaradi dela v šoli, z njim nismo imeli težav. Skoraj popolnoma novo in neznano nam je bilo programiranje Windows Phona.

Največ težav smo imeli pri izdelavi oblike urnika, saj Windows Phone nima elementa GridView, zato smo uporabili Grid, s katerim smo zgradili mrežo, v katero smo vstavili elemente TextBlock in v njih s pomočjo zanke vpisovali vsak podatek posebej.

Za sezname oddelkov, profesorjev in učilnic smo uporabili element List in vanj vpisali vse potrebne podatke.

```
            if (stolpec == 5)
            {
                stolpec = 0;
                vrstica++;
            }
            stolpec++;

            if (vrstica == 11) break;
        }
    }

    private void ListRazredi_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
        JedroWSSoapClient jedro = new JedroWSSoapClient();

        jedro.UrnikOddelkaACompleted += new EventHandler<UrnikOddelkaACompletedEventArgs>(jedro_UrnikOddelkaACompleted);
        jedro.UrnikOddelkaAAsync(ListRazredi.SelectedItem.ToString());

        SmartUrnik.Title = "SmartUrnik - " + ListRazredi.SelectedItem.ToString();
    }

    private void ListProfesorji_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
        JedroWSSoapClient jedro = new JedroWSSoapClient();

        jedro.UrnikProfesorjaACompleted += new EventHandler<UrnikProfesorjaACompletedEventArgs>(jedro_UrnikProfesorjaACompleted);
        jedro.UrnikProfesorjaAAsync(ListProfesorji.SelectedItem.ToString());

        SmartUrnik.Title = "SmartUrnik - " + ListProfesorji.SelectedItem.ToString();
    }

    private void ListUcilnice_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
        JedroWSSoapClient jedro = new JedroWSSoapClient();

        jedro.UrnikUcilniceACompleted += new EventHandler<UrnikUcilniceACompletedEventArgs>(jedro_UrnikUcilniceACompleted);
        jedro.UrnikUcilniceAAsync(ListUcilnice.SelectedItem.ToString());

        SmartUrnik.Title = "SmartUrnik - " + ListUcilnice.SelectedItem.ToString();
    }
```

Slika 12: Del kode Windows Phone aplikacije.

3.11 Delovanje Windows Phone aplikacije

Ob zagonu aplikacije dobimo enostaven pregled imen zaslonov, med katerimi se premikamo ali z dotikom na ime zaslona ali pa s prstom potegnemo levo ali desno.

Ko pri seznamih izberemo določen element, se nam na zaslonu »Urnik« izpiše urnik izbranega objekta.

Aplikacija deluje na vseh pametnih telefonih z Windows Phone verzijo 7.1 ali novejšo.

Slika 13: Videz Windows Phone aplikacije.

Slika 14: Videz urnika na Windows Phone aplikaciji.

3.12 Izdelava Android aplikacije

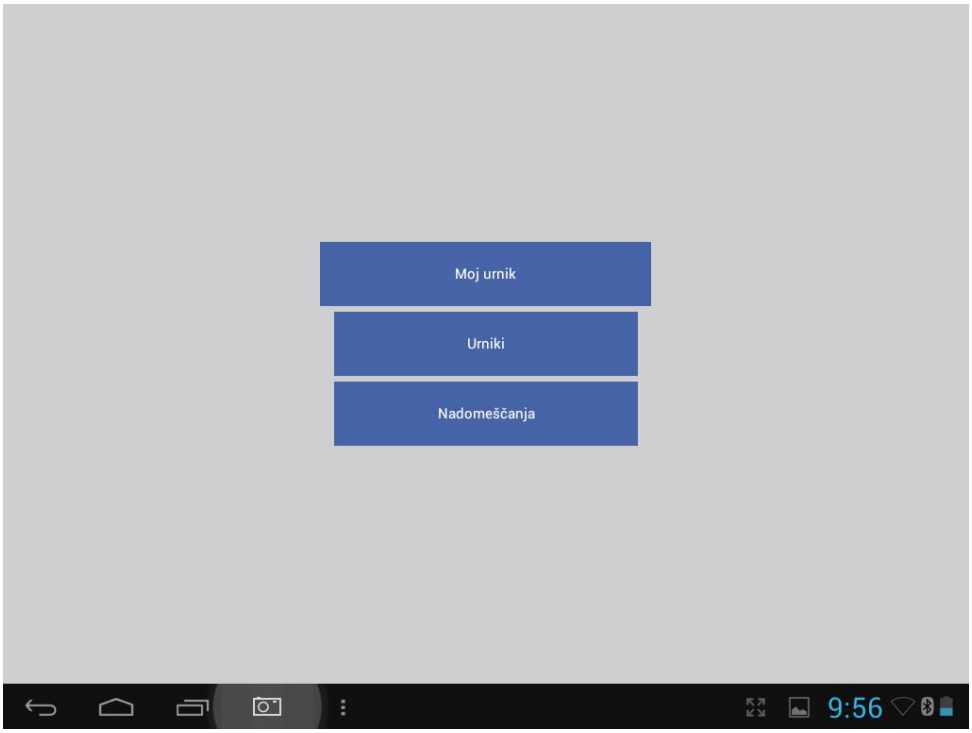
Telefoni s sistemom Android so trenutno najbolj priljubljeni in najbolj uporabljani med uporabniki. Aplikacijo smo izdelali v razvojnem okolju Xamarin studio, ki je bil za nas nov, zato smo imeli na začetku nekaj težav z razlikami. Kasneje je izdelava potekala brez večjih težav, zmotilo pa nas je dejstvo, da pri vnaprej narejenih elementih (listview, gridview) ni mogoče spreminjati velikosti in barve pisave ipd., saj smo za vpis uporabili Arrayadapter, ki te možnosti nima. Če bi želeli spremeniti te lastnosti, bi morali sprogramirati svoj adapter, za kar pa bi potrebovali veliko več znanja in časa. Podobno kot pri Windows Phone tudi Android aplikacije za oblikovanje grafične podobe programa uporablja podporo XAML, ki je bila za nas prav tako nova.

```
4  using Android.Runtime;
5  using Android.Views;
6  using Android.Widget;
7  using Android.OS;
8
9  namespace smartUrnikAndroid
10 {
11     [Activity (Label = "smartUrnik v0.3", MainLauncher = true, ScreenOrientation=Android.Content.PM.ScreenOrientation.Landscape)]
12     public class MainActivity : Activity
13     {
14
15         protected override void OnCreate (Bundle bundle)
16         {
17             RequestWindowFeature(WindowFeatures.NoTitle);
18             base.OnCreate (bundle);
19             SetContentView (Resource.Layout.Main);
20             var pokaziMoj = FindViewById<Button> (Resource.Id.buttonMy);
21             pokaziMoj.Click+=(sender, e) => {
22                 StartActivity(typeof(MojUrnik));
23             };
```

Slika 15: Del kode Android aplikacije.

3.12.1 Delovanje Android aplikacije

Ob zagonu aplikacije se nam prikaže okno, na katerem imamo možnosti izbire prikaza urnika ali nadomeščanj. Ob pritisku na gumb »Urniki«, se nam bo odprl seznam vseh razredov. Na vrhu aplikacije imamo še tri gumbe. Izbiramo lahko med oddelki, profesorji in učilnicami. Ob pritisku na določen objekt na seznamu se nam odpre urnik za izbrano vrednost. Če pritisnemo gumb »Nadomeščanja«, se nam izpišejo vsa nadomeščanja za današnji in naslednji dan. Aplikacija deluje na vseh Android operacijskih sistemih, ki imajo naložen sistem Android 2.1 oz. novejši.



Slika 16: Videz Android aplikacije na tablici TPAD.

Slika 17: Videz urnika v Android aplikaciji na tablici TPAD.

4 Zaključek

4.1 Hipoteze

Vse hipoteze so bile potrjene:

1. Z okensko, Android in Windows Phone aplikacijo, ter spletno stranjo smo uporabnikom olajšali dostop do urnikov in nadomeščanj. Z mobilnimi aplikacijami smo omogočili, da lahko uporabniki kjerkoli in kadarkoli z uporabo interneta dostopajo do želenih urnikov in pogledajo vsa nadomeščanja.
2. Razvili smo aplikacije za Windows, Android in Windows Phone platforme. Aplikacije prikazujejo urnike in nadomeščanja, ki so shranjena v podatkovni bazi.
3. Z uporabo medplatformskega principa razvoja aplikacij smo si olajšali delo pri vseh aplikacijah. Izdelali smo knjižnico, ki je skupno jedro vsem aplikacijam. Ta nam je omogočila del kode za pridobivanje podatkov, kar je približno 40 % kode, brez popravkov uporabiti na več platformah. Z uporabo te knjižnice lahko skupne popravke pri podatkih spremenimo tako, da spremenimo knjižnico in s tem popravimo vse aplikacije.

4.2 Težave ob izvedbi

Pri vsem tem smo naleteli na mnoge težave. Pri reševanju smo si pomagali z uradno dokumentacijo, raznimi objavami na forumih in blogih, video vodiči ipd. Z željo dokončati projekt smo težave rešili. Največ težav je bilo s pridobivanjem podatkov iz podatkovne baze v mobilne telefone, vendar smo težave na koncu rešili. Težave so se pojavile pri izdelavi Android aplikacije, saj nam je ta način razvoja nov.

4.3 Kaj smo se naučili?

Pri izdelavi smo prišli do veliko novih spoznanj, saj smo delali z novimi programskimi okolji, različnimi načini razvijanja aplikacij, prav tako pa smo spoznali načine, ki se uporabljajo za

razvijanje aplikacij tudi v velikih podjetjih. Naučili smo se tudi timskega dela, ki je bilo pomembno pri naši nalogi.

4.4 Smernice za nadaljnje delo

Med analizo problema se je porodila tudi ideja, da bi lahko sistem razvijali še naprej. Najprej imamo namen urediti obveščanja o nadomeščanjih za vse platforme. Aplikacija bi vsake pol ure preverjala, če so objavljena nova nadomeščanja. V tem primeru bi o tem obvestila uporabnika. V okenski aplikaciji v obliki oblačka, v Android aplikaciji kot obvestilo »Notification«, pri Windows Phone pa kot obvestilo »Toast« na zgornji strani ekrana.

Nameravamo izdelati tudi Android pripomoček (widget), ki bo glede na uro in dan prikazoval trenutno in naslednjo uro, pri tem pa bi upošteval tudi možna nadomeščanja.

5 Zahvala

Za pomoč pri izdelavi se zahvaljujemo našemu mentorju, profesorju mag. Boštjanu Resinoviču, za pomoč in podporo pri izdelavi raziskovalne naloge.

Zahvaljujemo se tudi profesorici Aniti Laznik, ki nam je pomagala pri teoretičnem načrtovanju okenske aplikacije za vpis nadomeščanj in nam svetovala, kaj bi bilo za uporabnike aplikacije najbolj enostavno za uporabo.

Zahvaljujemo se tudi našim sošolcem, ki so nam posredovali povratne informacije o uporabnosti in funkcionalnosti vseh aplikacij.

6 Viri

Olson, S. et al. 2012. Profesional *Cross-Platform Mobile Development in C#.* Indianapolis, John Wiley & Sons, Inc.

Dimic, M. 2012. *Medplatformski razvoj aplikacij.* Diplomska naloga, Fakulteta za računalništvo in informatiko, Ljubljana.

Xamarin. 2014. Dosegljivo na: http://xamarin.com/faq (datum gleda: 23. 2. 2014)

Cross-platform. Wikipedia. 2014. Dosegljivo na: http://en.wikipedia.org/wiki/Crossplatform (datum gleda:10. 1. 2014)

.NET Framework. Wikipedia. 2014. Dosegljivo na: http://en.wikipedia.org/wiki/.NET_Framework (datum gleda: 23. 2. 2014)

Comparison Chart for Mobile App Development Methods. AllianceTek. 2012. Dosegljivo na: http://www.alliancetek.com/downloads/article/comparison-chart-for-mobile-app.pdf (datum gleda: 25. 2. 2014)