Izvedba daljinskog pristupa bazi podataka radarskog sustava putem web servisa

Čižmešija, Filip

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Transport and Traffic Sciences / Sveučilište u Zagrebu, Fakultet prometnih znanosti**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:119:167847

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Transport and Traffic Sciences - Institutional Repository

SVEUČILIŠTE U ZAGREBU

FAKULTET PROMETNIH ZNANOSTI

Filip Čižmešija

IZVEDBA DALJINSKOG PRISTUPA BAZI PODATAKA RADARSKOG SUSTAVA PUTEM WEB SERVISA

ZAVRŠNI RAD

Zagreb, 2017.

SVEUČILIŠTE U ZAGREBU
FAKULTET PROMETNIH ZNANOSTI
ODBOR ZA ZAVRŠNI RAD

Zagreb, 24. travnja 2017.

Zavod: **Zavod za inteligentne transportne sustave**
Predmet: **Baze podataka**

ZAVRŠNI ZADATAK br. 4082

Pristupnik: **Filip Čižmešija (0036463398)**
Studij: Promet
Smjer: Informacijsko-komunikacijski promet

Zadatak: **Izvedba daljinskog pristupa bazi podataka radarskog sustava putem web servisa**

Opis zadatka:

U radu je potrebno izraditi programski sustav i sučelje koji će omogućiti pristup lokalnoj SQLite bazi podataka sustava Houston Radar. Podaci koje radar generira snimajući šest odvojenih trakova cestovne prometnice pohranjuju se u lokalnu bazu na samom uređaju. Baza je dostupna preko 3G Internet mreže, a novoizvedeni programski sustav na serveru svakih sat vremena dohvaća podatke i pohranjuje podatke u bazu. Putem web servisa moguće je pristupiti svim podacima na serveru. Potrebno je postaviti radar, kreirati sustav i testirati ga.

Zadatak uručen pristupniku: 28. travnja 2017.

Mentor:

prof. dr. sc. Tonči Carić

Predsjednik povjerenstva za završni ispit:

Sveučilište u Zagrebu

Fakultet prometnih znanosti

ZAVRŠNI RAD

IZVEDBA DALJINSKOG PRISTUPA BAZI PODATAKA RADARSKOG SUSTAVA PUTEM WEB SERVISA

IMPLEMENTATION OF REMOTE ACCESS TO THE RADAR SYSTEM DATABASE BY WEB SERVICES

Mentor: prof. dr. sc. Tonči Carić

Student: Filip Čižmešija, 0036463398

ZAGREB, 2017.

SAŽETAK:

Ovaj rad opisuje i daje rješenje problematike povezivanja više radarskih uređaja i dohvata podataka. Radi analize i objedinjenosti podataka na centralnom serveru, kreirana je aplikacija koja dohvaća podatke s radarskih sustava. Nakon dohvata podaci se pohranjuju na centralnom serveru.

Dohvat podataka izvršava se putem Interneta, a radaru se pristupa putem mobilne mreže TELNET protokolom. Centralnoj bazi se pristupa se web servisom, čija je prednost interoperabilnost neovisno o uređaju koji želi dohvatiti podatke.

Korisnik navedenog radarskog sustava podacima o svim uređajima može pristupiti preko servera, centralog mjesta na kojem se pohranjuju svi podaci. Sustavu je moguće pristupiti bilo gdje putem Interneta te vidjeti podatke sa svih radara.

KLJUČNE RIJEČI: Web servis, SOAP, REST, SQL, PHP, Apache, C#

SUMMARY:

This document describes and provides a solution to the problem of connecting multiple radar systems and obtaining data. For analysis and aggregation of data on a central server, an application that retrieves data from the radar systems has been created. After receipt, the data is stored on the central server.

Data retrieval is realised over the Internet, and the radar is accessed through the TELNET via mobile network. The central database is accessed by a web service whose advantage is interoperability irrespective of the device that wants to retrieve the data.

The user of the above radar system can access data on all devices via a server, a central place where all the data is stored. The system can be accessed anywhere via the Internet and see the data from all radars.

KEYWORDS: Web service, SOAP, REST, SQL, PHP, Apache, C#

Sadržaj

1. Uvod 1
2. Web servis 2
   2.1. Općenito o web servisima 2
   2.2. SOAP web servis 4
   2.3. Potreba primjene web servisa 5
3. Baze podataka Microsoft SQL i SQLite 7
   3.1. Microsoft SQL 7
   3.2. SQLite 8
   3.3. Shematski prikaz povezivanja baza 8
4. Radarski sustav „Houston Radar“ 9
5. Proces dohvata podataka s radara 10
   5.1. Aplikacija za dohvat podataka 10
      5.1.1. Pokretanje aplikacije i dohvat informacija o radaru 12
      5.1.2. Povezivanje na radar i dohvat podataka 13
   5.2. Struktura baze podataka Microsoft SQL server 15
6. Web servis za dohvat podataka 17
   6.1. Izrada SOAP web servisa 18
   6.2. Testiranje rada web servisa 21
7. Izrada PHP web aplikacije 25
   7.1. Izgled korisničkog sučelja 25
   7.2. Funkcionalnost web aplikacije 30
   7.3. Tablice unutar baze podataka korištene za rad web aplikacije 31
8. Sklopovska podrška i shema spajanja 34
9. Zaključak 36

Literatura 37

Popis kratica 38

Popis slika 39

Popis tablica 40

Popis primjera 40

Popis priloga 41

1. Uvod

Radi potrebe mjerenja prometnog toka i zagušenja cestovnog prometa kreirani su radarski uređaji. Radarski uređaji mjere brzine kretanja vozila, količinu vozila, duljinu vozila itd. Radarski uređaj Houston Radar SpeedLane jedan je od takvih uređaja. Ima funkcionalnost pohrane podataka o brzini, smjeru kretanja, duljini vozila i prometnom traku u kojem se vozilo kreće. Paralelno može pratiti i do osam prometnih traka.

Radarski uređaj ima opcionalan 3G modul i mogućnost centraliziranog slanja i prikaza podataka. S obzirom na ograničenost navedenog modula, pojavila se potreba kreiranja sustava koji bi objedinio više radara te imao mogućnost pohrane i prikaza podataka na jednom mjestu. Također, potrebno je omogućiti pristup sustavu putem interne mreže fakulteta. Za ostvarivanje navedenog i realizaciju pristupa, osim sklopovske i mrežne podrške, izrađena je web aplikacija za prikaz podataka i web servis za dohvat podataka. Sklopovska podrška sastoji se od web servera, baze podataka i radarskog uređaja dok se mrežna podška sastoji od usmjernika i modema za pristup mobilnoj mreži. U ovom radu detaljno ću prikazati sheme spajanja, sheme rada i način izrade te funkcionalnosti navedenog sustava.

2. Web servis

Web servisi su aplikacije koje služe za komunikaciju između različitih sustava, primjerice računalo i radarski uređaj. Također, moguće ih je primijeniti između različitih aplikacija radi međusobne komunikacije, primjerice web server i baza podataka. Radi potrebe razmjena informacija pojavila se i potreba za korištenjem web servisa zbog njihove jednostavne primjene i funkcionalnosti neovisno o programskom jeziku u kojemu je pisana krajnja aplikacija. Primjerice isti web servis jednako dobro može raditi na sustavu baziranom na Linux-u(Android, Ubuntu...) kao i na Windowsu odnosno iOS, macOS.

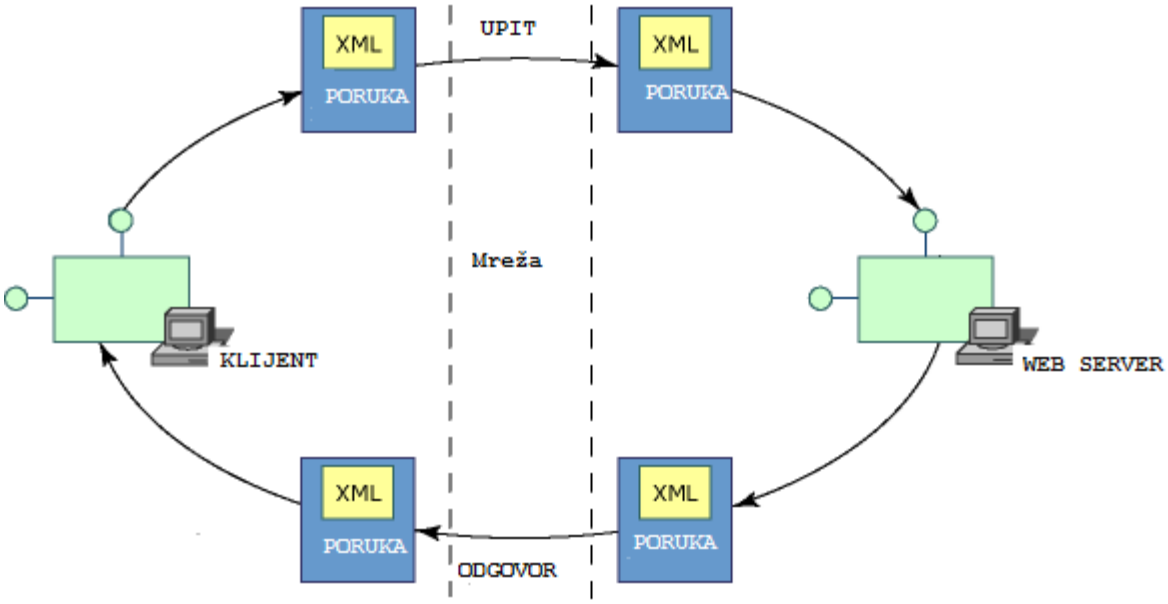
2.1. Općenito o web servisima

Ono što web servisu omogućuje povezivanje među različitim sustavima je upotreba HTTP (*HyperText Transfer Protocol*) protokola za komunikaciju. HTTP protokol je jedan od najraširenijih protokola koji se svakodnevno koristi. Upravo to svrstava taj protokol u sigurne protokole koji su dozvoljeni za upotrebu na svim operativnim sustavima, odnosno svim mrežama. Za komunikaciju nije potrebno dozvoljavati pristup nikakvim dodatnim portovima kao ni instalirati nikakav dodatni softver. Otvaranje portova nije poželjno, primjerice u velikim sustavima, iz sigurnosnih razloga.

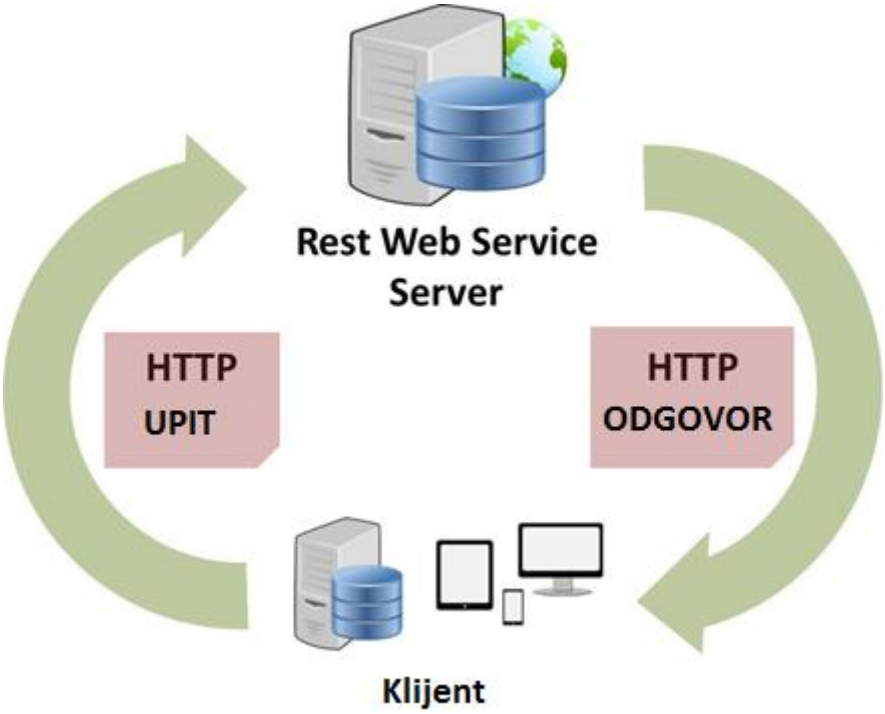
Povijesno gledano[1], web servisi se koristite 2002. godine. S godinama, njihovo korištenje postaje sve intenzivnije, pogotovo pojavom pametnih mobilnih uređaja.

Web servisi rade po principu zahtjeva (*HTTP Request*) i odgovora (*HTTP Response*). Korisnik šalje upit (*Request*) prema web servisu te dobiva odgovor (*HTTP Response*). Upiti i odgovori unaprijed su definirani ovisno o vrsti web servisa koji se koristi. Primjerice SOAP (*Simple Object Access Protocol*) web servisi koriste poruke u XML (*EXtensible Markup Language*) zapisu, dok RESTful (*Representational state transfer*) web servisi koriste HTTP poruke. Na slikama 1 i 2 prikazan je princip rada web SOAP i RESTful web servisa.

[1] Databaseanswers.org. (2017). History of Web Services. [online] Available at: http://www.databaseanswers.org/web_services_history.htm [Accessed 15 Jun. 2017]



Slika 1: Prikaz rada SOAP web servisa [7]



Slika 2: Prikaz rada RESTful web servisa [8]

2.2. SOAP web servis

Prema Mitchell, L.J., SOAP[2] je vrsta web servisa inicijalno predstavljena od strane Microsofta. Bazira se na RPC servisu (Remote Procedure Call) te koristi posebnu XML sintaksu. S obzirom na postojanje WSDL-a (Web Service Description Language), opisnog jezika web servisa, vrlo ga je jednostavno implementirati u bilo koju aplikaciju. WSDL sadrži ulazne parametre, vrste varijabli koje se koriste, metode, parametre i izlazne vrijednosti koje su dostupne. Također zapisan je u svojevrsnom XML formatu, no taj format je lakše razumljiv strojevima nego ljudima. Također, WSDL nije uvjet za rad web servisa no daleko pojednostavljuje upotrebu i implementaciju web servisa. Ukoliko se koristi WSDL, svaka aplikacija „pročita“ WSDL web servisa te na temelju toga određuje koji su ulazni parametri i koja je očekivana sintaksa odgovora.

Upit prema web servisu može i ne mora sadržavati određene ulazne parametre. Također neki ulazni parametri mogu biti opcionalni. U primjerima 1 i 2 prikazan je web servis koji je izrađen u okviru ovog rada. Prvo je prikazan SOAP Request, a nakon toga SOAP Response. Vidljivo je da su ulazni parametri limit i radar, dok su izlazni parametri vrijeme, speed, lane i direction.

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:urn="urn:server">
   <soapenv:Header/>
   <soapenv:Body>
      <urn:podatak
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
         <limit xsi:type="xsd:int">1</limit>
         <radar xsi:type="xsd:int">1</radar>
      </urn:podatak>
   </soapenv:Body>
</soapenv:Envelope>
```

Primjer 1: SOAP Request

[2] Mitchell, L. (2016). PHP web services. 2nd ed. Sebastopol: O'Reilly Media, p.66

```
<SOAP-ENV:Envelope SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
      <ns1:podatakResponse xmlns:ns1="urn:server">
         <return xsi:type="SOAP-ENC:Array" SOAP-ENC:arrayType=":[2]">
            <item xsi:type="xsd:int">1</item>
            <item xsi:type="xsd:">
               <item>
                  <vrijeme xsi:type="xsd:string">2015-10-14
17:00:24.000</vrijeme>
                  <speed xsi:type="xsd:string">18</speed>
                  <lane xsi:type="xsd:string">4</lane>
                  <direction xsi:type="xsd:string">2</direction>
               </item>
            </item>
         </return>
      </ns1:podatakResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Primjer 2: SOAP Response

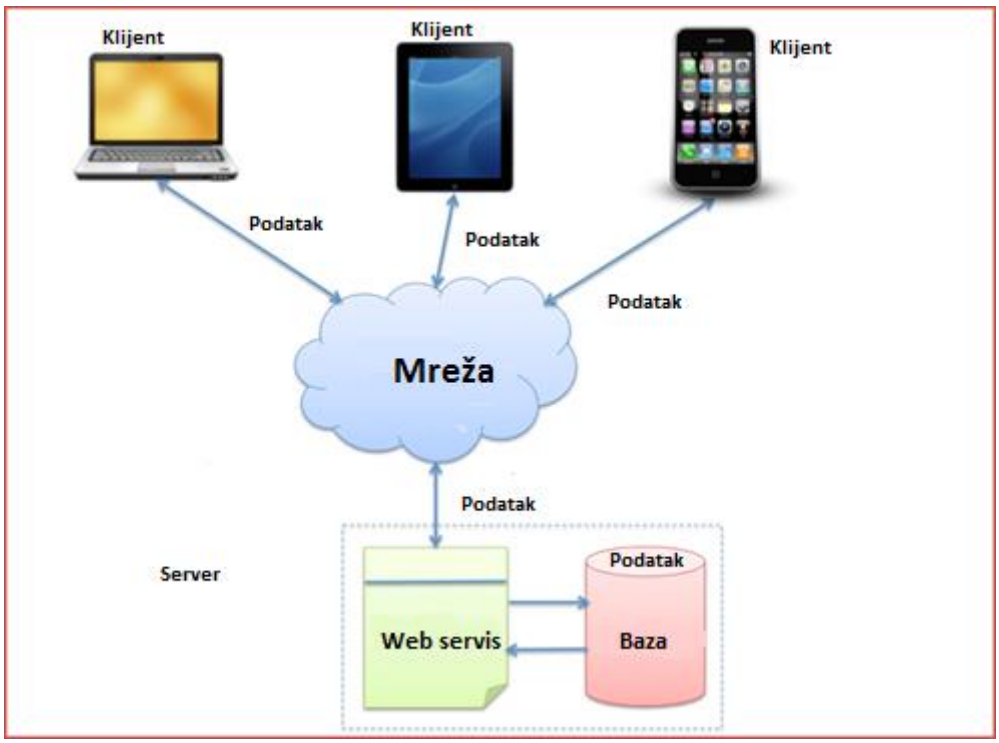
2.3. Potreba primjene web servisa

Web servisi omogućuju sigurno izvršavanje određenih metoda, dohvat podataka iz baze i slično. Primjerice, želimo li dohvatiti podatke iz baze podataka, sigurnije je kreirati web servis za dohvat istih nego direktno pristupati samoj bazi, pogotovo putem Interneta. Ukoliko se pristupa direktno na bazu, direktan pristup vrlo lako može sigurnosna prijetnja na bazu podataka, odnosno podataka pohranjenih u njoj. Ukoliko se putem web servisa pristupa istoj, puno je veća sigurnost s obzirom da se web servisom jasno definira tko što može dohvatiti. Također, pošto web servis radi preko HTTP protokola nisu potrebne nikakve dodatne konfiguracije, dok bi recimo za pristup na bazu bilo potrebno otvarati portove za bazu (primjerice portove 1433,3306,1521...).

Na nekim uređajima nije ni moguć direktan pristup bazi podataka nego je ostvariv isključivo putem web servisa. Android okruženje je najbolji primjer takvog pristupa. Nije moguće doći do baze podataka na udaljenom serveru direktno nego isključivo posredstvom web servisa.

Samim time može se doći do zaključka da su web servisi primjenjivi na svim uređajima, neovisno o operativnom sustavu te svim aplikacijama, neovisno kojim programskim jezikom

pisane. Web servisi su svuda oko nas i svakodnevno u upotrebi iako možda i nismo svjesni da ih koristimo. Na slici 3 prikazan je shematski pristup bazi podataka putem web servisa putem Interneta, neovisno o operativnom sustavu odnosno aplikaciji.



Slika 3: Prikaz pristupa bazi putem web servisa [9]

3. Baze podataka Microsoft SQL i SQLite

Relacijske baze podataka su baze koje se zasnivaju na relacijskom modelu. U ovom se modelu podaci organiziraju u skup relacija između kojih su definirane određene veze. „Bazu podataka promatramo kao zbirku podatkovnih zapisa pohranjenih na računalu koja je lako dostupna korisnicima i aplikacijama.“[3] Bazama se upravlja strukturnim upitnim jezikom (SQL - Structured Query Language). Osnove relacijskih baza razvijene su od strane kompanije IBM.

Postoji nekoliko popularnih sustava baza podataka kao što su: Microsoft SQL Server, Oracle Database, MySQL, SQLite i drugi. Za pristup i obradu navedenih baza koristi se SQL jezik.

Projekt radarskog sustava zasniva se na dvije vrste baza podataka: lokalnoj bazi na samom radarskom sustavu, koja je bazirana na SQLite-u te Microsoft SQL-u na serveru gdje se obrađuju i prikazuju podaci.

3.1. Microsoft SQL

Microsoft SQL server[4] je relacijska baza razvijena od strane Microsofta. Koristi T-SQL jezik za upite prema bazi. Naziv T-SQL dolazi od Transact SQL što znači da može izvršavati i kompleksnije upite nad bazom. Baza podataka je poslužiteljski orijentirana baza jer je pokrenuta na serveru, a klijenti se spajaju na bazu. Odvojena je od same krajnje aplikacije. Nad bazom je moguć i udaljeni pristup s drugih klijenata uz uvjet propuštanja određenih portova.

[3] Fakultet prometnih znanosti. (2017). Uvod u relacijske baze podataka. [online] Available at: http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf [Accessed 30 Aug. 2017].

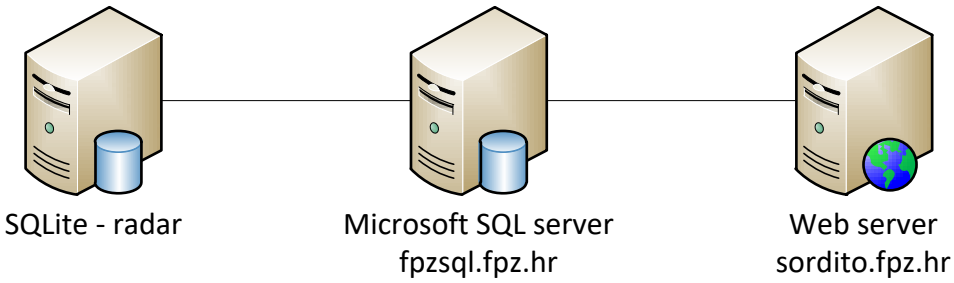
[4] Msdn.microsoft.com. (2017). Transact-SQL Reference (Transact-SQL). [online] Available at: https://msdn.microsoft.com/en-us/library/ms189826.aspx [Accessed 15 Jun. 2017].

3.2. SQLite

SQLite[5] je također relacijska baza podataka. Za razliku od Microsoftove, ova baza je uključena u samu krajnju aplikaciju. Ne postoji mogućnost udaljenog spajanja na samu bazu upravo zbog tog ograničenja. Pogodne su za lokalnu upotrebu s ograničenom količinom podataka.

3.3. Shematski prikaz povezivanja baza

S obzirom na potrebu korištenja različitih tipova baza i različitih mogućnosti upravljanja istima, napravljen je shematski plan povezivanja baza koji je prikazan na slici 4.



Slika 4: Shematski prikaz baza podataka

Na radarskom sustavu nalazi se SQLite baza podataka. Server na adresi fpzsql.fpz.hr dohvaća podatke iz lokalne baze podataka radara. Web server na adresi sordito.fpz.hr služi za dohvat i prikaz podataka kao i za web servis. Na serveru gdje se nalazi baza podataka instaliran je Microsoft SQL server 2016, dok je na serveru gdje je web server instaliran Apache. Međusobno su povezani lokalnom mrežom fakulteta. Komunikacija između radara i baze podataka izvršava se putem Interneta.

[5] Sqlite.org. (2017). About SQLite. [online] Available at: https://www.sqlite.org/about.html [Accessed 15 Jun. 2017].

4. Radarski sustav „Houston Radar“

Radarski sustav korišten u radu je proizvođača Houston Radar[6]. Model radara je SpeedLane. Pomoću navedenog radara moguće je pratiti i do 8 prometnih traka. Precizno mjeri brzinu, duljinu vozila te određuje u kojoj traci se vozilo kreće. Također, pohranjuje statističke podatke te je moguće doći do informacija o prometnom opterećenju, zagušenosti, prosječnoj brzini i slično. Posjeduje i video kameru pomoću koje je moguće bilježiti fotografije s mjesta mjerenja, kao i prijenos video signala uživo.

Napaja se pomoću eksterne baterije u rasponu napona od 7 – 15 V. Poveziv je s računalom ili drugim uređajima putem Bluetooth konekcije, klasične LAN mreže ili RS232 priključka.

Otporan je na vremenske uvijete te može raditi u svim uvjetima od -40 do 85 °C.

Koristi bazu podataka SQLite gdje pohranjuje sve izmjerene vrijednosti. Zbog samog tipa baze limitiran je na 1 000 000 zapisa.

Slika 5: Houston Radar SpeedLane

[6] Houston-radar.com. (2017). Houston Radar - The Next Generation Ultra Low Power Traffic Calming Speed Radar. [online] Available at: http://houston-radar.com/speedlane.php [Accessed 15 Jun. 2017].

5. Proces dohvata podataka s radara

Radarski sustav sadrži bazu podataka SQLite. Na serveru, na kojemu je pohranjena baza i koji služi za web servis kao i web prikaz, instalirana je baza Microsoft SQL. Zbog nemogućnosti spajanja direktno na SQLite, što je već ranije opisano, kreirana je aplikacija za dohvat podataka s radara i pohranu u bazu na serveru. Proizvođač, Houston Radar ustupio nam je SDK kit pomoću kojeg je moguć dohvat podataka.

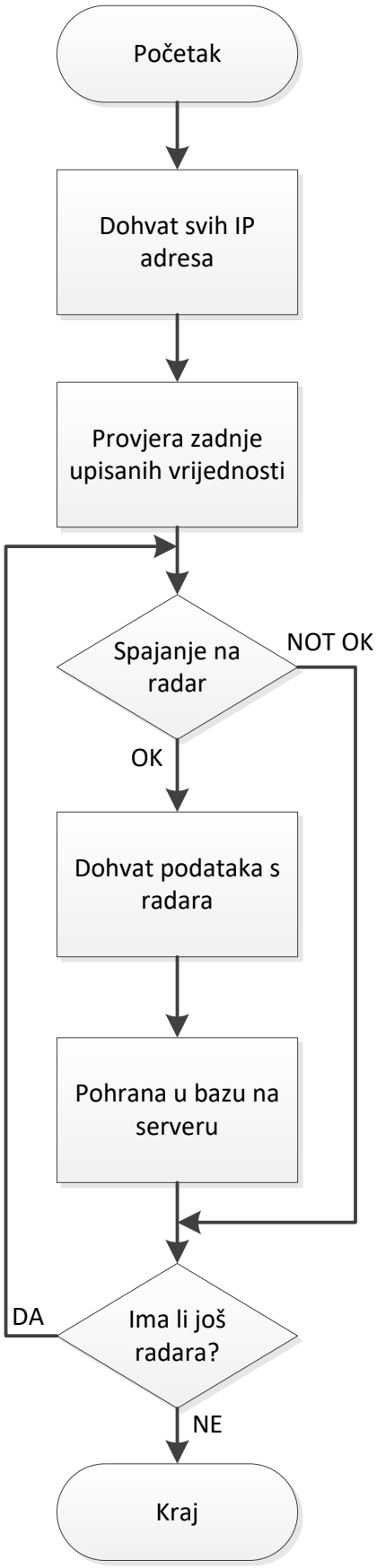
5.1. Aplikacija za dohvat podataka

Aplikacija za dohvat podataka razvijena je uz pomoć SDK kita od strane proizvođača radara. Pisana je u C# programskom jeziku te prilagođena sustavu i okolini gdje se koristi. Ima mogućnost spajanja na više radarskih sustava putem Interneta te pohranu istih u lokalnu bazu servera.

Redoslijed izvršavanja radnji:

1. Aplikacija se pokreće jednom u sat vremena te provjerava popis svih radara u bazi na serveru s pripadajućom IP adresom.
2. Nakon provjere svih radara, za svaki pojedinačno se provjerava zadnji upis u bazi na serveru.
3. Nakon dohvata, spaja se na svaki radar pojedinačno i dohvaća podatke i pohranjuje ih u memoriju.
4. Nakon dohvata podataka iste podatke unosi u bazu podataka na serveru.

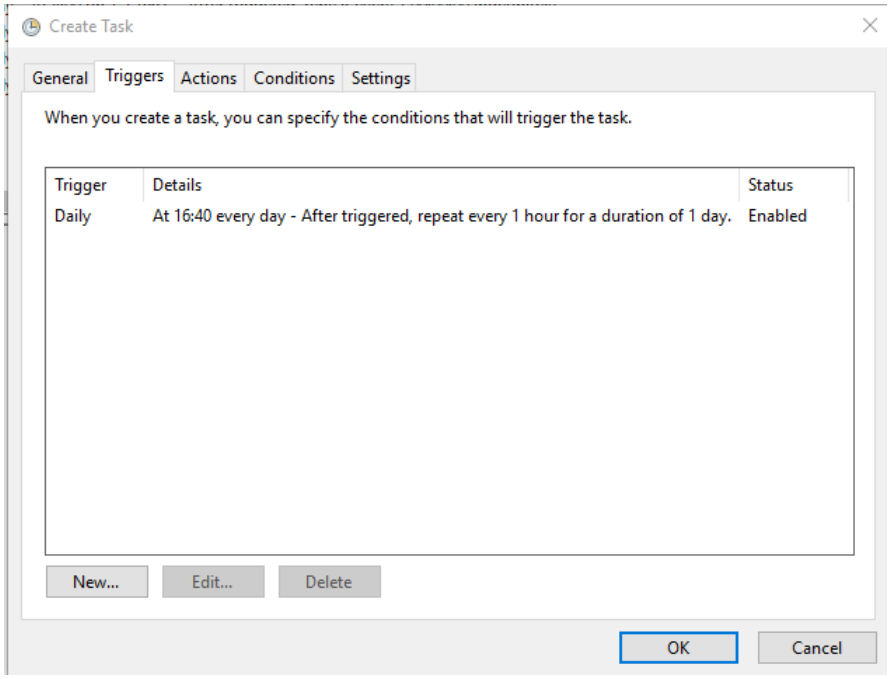
Na slici 6 prikazan je dijagram toka rada aplikacije za dohvat podataka s radarskog sustava.



Slika 6: Dijagram toka dohvata podataka

5.1.1. Pokretanje aplikacije i dohvat informacija o radaru

Aplikacija je konfigurirana da se pokreće jednom u sat vremena. Ovisno o potrebi to može biti češće, odnosno rjeđe. Navedeno se konfigurira u Task Scheduleru Windows operativnog sustava. Prikaz konfiguracije vidljiv je na slici 7.



Slika 7: Konfiguracija Task Scheduler-a

Nakon pokretanja aplikacije aplikacija se spaja na server na adresi fpzsql.fpz.hr i dohvaća podatke o svim radarima evidentiranim u bazi. Baza sadrži podatke i IP adresi te kratki opis o svakom radaru.

Metoda koja se koristi za spajanje na bazu i dohvat podataka prikazana je na slici 8.

```
1 reference
private void inicijalnoSQL()
{
    string connectionString = "server=fpzsql.fpz.hr\\SQLEXPRESS;database=radar;integrated Security=SSPI;";

    using (SqlConnection _con = new SqlConnection(connectionString))
    {
        string queryStatement = "SELECT [ip] ,[opis] FROM [radar].[dbo].[radar_info]";

        using (SqlCommand _cmd = new SqlCommand(queryStatement, _con))
        {
            DataTable customerTable = new DataTable("IP");

            SqlDataAdapter _dap = new SqlDataAdapter(_cmd);

            _con.Open();
            _dap.Fill(customerTable);
            _con.Close();
            if (customerTable.Rows.Count > 0)
            {
                listaIP.Clear();
                for (int i = 0; i < customerTable.Rows.Count; i++)
                {
                    listaIP.Add(customerTable.Rows[i][0].ToString());
                }
            }
        }

    }

}
```

Slika 8: Metoda za dohvat liste svih radara

5.1.2. Povezivanje na radar i dohvat podataka

Nakon dohvata svih radara aplikacija se spaja na radarski sustav uz pomoć SDK-a proizvođača. Pošto je riječ o SQLite bazi podataka, spajanje na radar izvršava se u konzoli pomoću telneta, a podaci koji se dobivaju su u obliku stringa. Uspostava konekcije prikazana je na slici 9.

```
1 reference
private void ConnectToRadar(string IPadress)
{
    rdr = new radarCommClassThd();

    //Use IP (remote connection)?
    rdr.IPaddr = IPadress; //change to whatever you want to read from some central store.
    rdr.portnum = 23; //use correct port # for your system where radar will be found
    rdr.DoSerialConnect = false;

    rdr.ReadTimeout = rdr.WriteTimeout = 5000; //In ms. 5 seconds. Adjust as desired. Longer for network, shorter for ser

    rdr.RadarEventRadarFound += new radarCommClassThd.RadarEventRadarFoundHandler(rdr_RadarEventRadarFound);
    rdr.RadarEventGetInfoDone += new radarCommClassThd.RadarEventGetInfoDoneHandler(rdr_RadarEventGetInfoDone);
    rdr.RadarEventRadarNotFound += new radarCommClassThd.RadarEventNotFoundHandler(rdr_RadarEventRadarNotFound);
    rdr.RadarEventEx += new radarCommClassThd.RadarEventExHandler(rdr_RadarEventEx);
    rdr.RadarEventCommErr += new radarCommClassThd.RadarEventCommErrHandler(rdr_RadarEventCommErr);
    //there are other radar events if you want some more info on connection progress etc. but not really required.
    radarfound = false;

    //Make background thread. This makes the connection process more reliable and happens in the background.
    Thread conthd = new Thread(rdr.Connect);
    conthd.Start();
}
```

Slika 9: Metoda za uspostavu konekcije

Ukoliko je konekcija uspješna, aplikacija dohvaća podatke, obrađuje string te generira polje za unos. String pretvara u polje na temelju novog retka odnosno razmaka u redu. Funkcionalnost obrade stringa i upisa u Microsoft SQL bazu na serveru prikazana je na slici 10.

```
try
{
    //Make up approporiate query. See HoustonRadarSpeedLaneSDKQuickStartGuide for more info about table
    //schemas and queries.
    const string query = "select id,Time,Speed,Lane,Direction,Length from targets where id>=102501 AND id<105001 AND Lane>0 ";
    Console.WriteLine("Issuing a manual query:" + query);
    schema.QuerySQL(query, out response);
    Console.WriteLine(response);
    string[] linija = response.Split(new string[]{"\r\n","\n"},StringSplitOptions.None);
    CultureInfo ci = new CultureInfo("en-US");
    for (int i = 0; i < linija.Length; i++)
    {
        string[] s = linija[i].Split(' ');
        for (int j = 0; j < s.Length; j++)
        {
            int pos = s[j].IndexOf('=');
            s[j] = s[j].Remove(0, pos + 1);
            Console.WriteLine(s[j]);
        }
        DateTime vrijeme = DateTime.Parse(s[1] + " " + s[2]);

        string upit = "INSERT INTO [dbo].[radar_stanje] ([radar_id],[id_radara] ,[vrijeme],[speed],[lane],[direction],[length])VALUES";
        upit += "(1," + s[0] + ",'" + vrijeme.ToString("yyyy-MM-dd HH:mm:ss.fff") + "'," + s[3] + "," + s[4] + "," + s[5] + "," + s[6] + ")";
        Console.WriteLine(upit);
        SQLInsert(upit);

    }
```

Slika 10: Dohvat i pohrana podataka

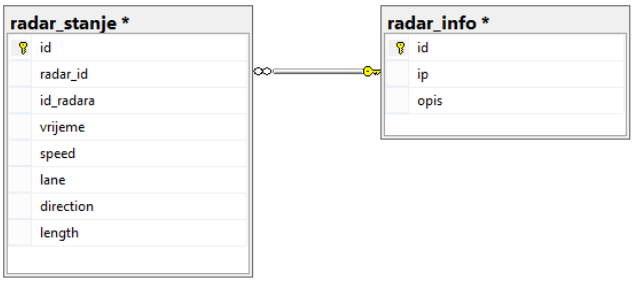
String query sadrži upit prema SQLite bazi na radaru. Dohvaća id, vrijeme, brzinu, prometni trak, smjer i duljinu vozila; u ovom slučaju za sve zapise s id-om većim od 102501. Varijabla response sadrži podatak dobiven s radara.

Polje linija nastaje razdvajanjem dobivenog rezultata s obzirom na novi redak („\r\n“). Dobiveno polje iterira se prvom for petljom te se naknadno razdvaja u polje s koje nastaje dijeljenjem prethodnog stringa s obzirom na razmak. Nakon toga ponovno se iterira novonastalo polje s drugom for petljom gdje briše prazne znakove. Datetime varijabla vrijeme pohranjuje podatak o vremenu koji je prethodnim razdvajanjem bio razdvojen. U konačnici slijedi Insert upit prema bazi koji pohranjuje obrađeno polje u SQL bazu na serveru.

Ukoliko postoji još radara ova akcija će se ponavljati ovisno o tome koliko je radara bilo u inicijalnom čitanju baze radara na Microsoft SQL serveru

5.2. Struktura baze podataka Microsoft SQL server

Baza samog radara na Microsoft SQL serveru sastoji se od pet različitih tablica. Od navedenih, dvije tablice služe za učitavanje podataka i omogućavanje nesmetanog rada sustava, dok prestale služe za rad web sučelja. Na slici 11 prikazana je shema tih dviju tablica i njihova međusobna veza.



Slika 11: Tablice unutar baze namijenjene za pohranu podataka

U tablicama 1 i 2 opisana je struktura pojedine SQL tablice

NAZIV ATRIBUTA	**VRSTA ATRIBUTA**	**KRATKI OPIS**
ID	INT	Identifikator
IP	NVARCHAR	IP adresa radara
OPIS	NVARCHAR	Kratki opis navedenog radara

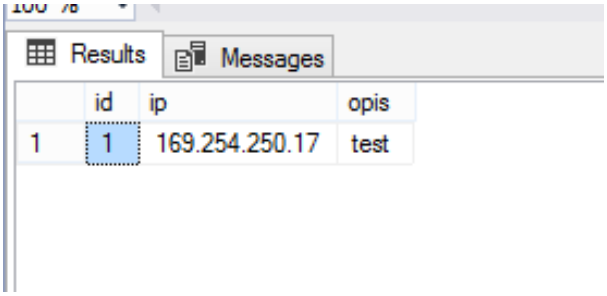
Tablica 1: Opis SQL tablice „radar_info“

NAZIV ATRIBUTA	**VRSTA ATRIBUTA**	**KRATKI OPIS**
ID	INT	Identifikator
RADAR_ID	INT	Strani ključ nad tablicom radar_info
ID_RADARA	INT	Identifikator iz SQLite baze podataka
VRIJEME	DATETIME	Vrijeme zapisa s radara
SPEED	INT	Brzina vozila
LANE	LANE	Prometna traka

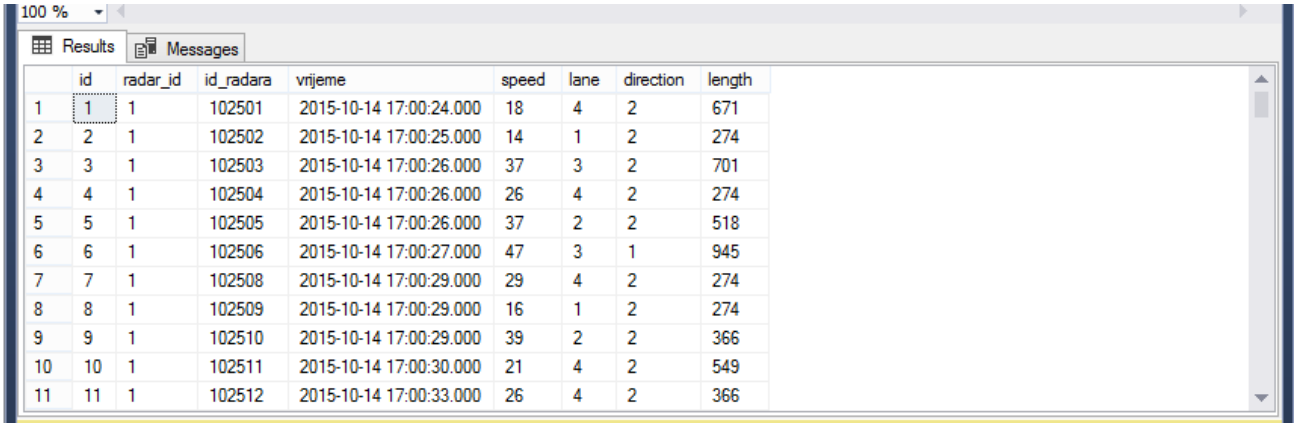
DIRECTION	INT	Smjer
LENGTH	INT	Duljina vozila

Tablica 2: Opis SQL tablice „radar_stanje“

Na slikama 12 i 13 prikazana je struktura i podaci unutar samih tablica „radar_info“ i „radar_stanje“.



Slika 12: Prikaz tablice „radar_info“



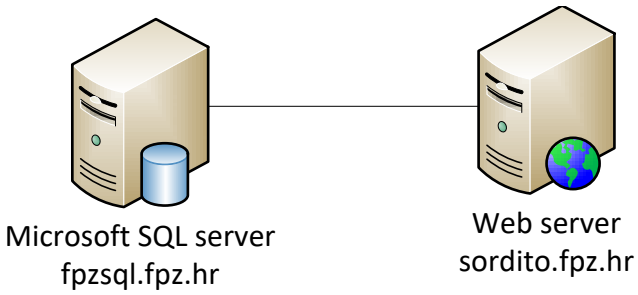
	id	radar_id	id_radara	vrijeme	speed	lane	direction	length
1	1	1	102501	2015-10-14 17:00:24.000	18	4	2	671
2	2	1	102502	2015-10-14 17:00:25.000	14	1	2	274
3	3	1	102503	2015-10-14 17:00:26.000	37	3	2	701
4	4	1	102504	2015-10-14 17:00:26.000	26	4	2	274
5	5	1	102505	2015-10-14 17:00:26.000	37	2	2	518
6	6	1	102506	2015-10-14 17:00:27.000	47	3	1	945
7	7	1	102508	2015-10-14 17:00:29.000	29	4	2	274
8	8	1	102509	2015-10-14 17:00:29.000	16	1	2	274
9	9	1	102510	2015-10-14 17:00:29.000	39	2	2	366
10	10	1	102511	2015-10-14 17:00:30.000	21	4	2	549
11	11	1	102512	2015-10-14 17:00:33.000	26	4	2	366

Slika 13: Prikaz tablice „radar_stanje“

6. Web servis za dohvat podataka

Za dohvat podataka sa servera na kojem se nalazi baza Microsoft SQL server, kreiran je SOAP web servis. SOAP kao standard odabran je zbog svoje rasprostranjenosti i jednostavnosti primjene. Za izradu web servisa korišten je PHP programski jezik. PHP je odabran zbog velike količine različitih funkcionalnosti namijenjenih za SOAP web servise. Prethodno navedene funkcionalnosti odnose se na serversku i na klijentsku stranu web servisa.

Web servis i web sučelje smješteni su na posebnom serveru. Server se nalazi na adresi sordito.fpz.hr , dok se baza nalazi na serveru fpzsql.fpz.hr. Na slici 14 detaljno je prikazana shema spajanja web servera i baze podataka. Na web serveru instaliran je Apache zajedno s PHP-om.



Slika 14: Shematski prikaz baze podataka i web servera

Pri kreiranju web servisa korištena je besplatna „Open Source“ biblioteka za izradu SOAP web servisa imena NuSOAP server. Biblioteka je razvijena od strane web zajednice i otvorenog je koda, a znatno olakšava izradu i implementaciju SOAP web servisa. Uz serversku biblioteku postoji i klijentska NuSOAP client koja je kasnije korištena za dohvat podataka.

6.1. Izrada SOAP web servisa

Web servis ima jednu osnovnu funkciju - dohvat podataka s radara ovisno o ulaznim parametrima. Funkcija kreirana za dohvat podataka nazvana je „podatak“. Sadrži ulazne parametre limit i radar. Varijabla limit poprima vrijednost broja rezultata koji želimo da nam web servis vrati, dok varijabla radar označava identifikaciju radara.

```
require_once('lib/nusoap.php');

$server = new nusoap_server;

$server->configureWSDL('server', 'urn:server');

$server->wsdl->schemaTargetNamespace = 'urn:server';
```

Prilog 1: Inicijalizacija NuSOAP biblioteke

U prilogu 1 vidljiva je inicijalizacija NuSOAP biblioteke i kreiranje objekta server. Također, iz navedenoga koda vidljivo Je kako web servis sadrži WSDL koji ima opisnu funkciju.

```
$server->register('podatak',
                  array('limit' => 'xsd:int','radar' => 'xsd:int'),
//parameter
                  array('return' => 'tns:array2'),  //output
                  'urn:server',   //namespace
                  'urn:server#podatak',  //soapaction
                  'rpc', // style
                  'encoded', // use
                  'vrati podatak');  //description
```

Prilog 2: Definiranje ulaznih parametara

U Prilogu 2 prikazana je funkcija „podatak“. U njoj je sadržano polje sa ulaznim parametrima. Prvi element polja nazvan je limit, dok je drugi nazvan radar. Oba navedena spadaju pod varijable tipa integer. Navedeno opisuje karakterističnosti ulaznih parametara. Sljedeće polje nazvano je array2, a služi za ispisivanje dobivenog rezultata. Sljedeći elementi opisuju samo pozivanje upita prema web servisu. Između ostalog, tu je riječ o nazivu same akcije prema web servisu, stilu (RPC), opisu same funkcije i sl.

```
$server->wsdl->addComplexType(
        'array2',
        'complexType',
        'array',
        'all',
        'SOAP-ENC:Array',
        array(),
        array(
            array(
                'ref' => 'SOAP-ENC:arrayType',
                'wsdl:arrayType' => 'tns:Array[]'
            )
        )
);


$server->wsdl->addComplexType(
    'Array',
    'complexType',
    'array',
    'all',
    'SOAP-ENC:Array',
    array(
      'kaunt' => array('name' => 'kaunt', 'type' => 'xsd:int'),
        'vrijeme' => array('name' => 'vrijeme', 'type' => 'xsd:string'),
        'lane' => array('name' => 'lane', 'type' => 'xsd:int'),
        'speed' => array('name' => 'speed', 'type' => 'xsd:int'),
        'direction' => array('name' => 'direction', 'type' => 'xsd:int')
    )
);
```

Prilog 3: Definiranje izlaznih parametara

Predhodno je opisano da je izlazni parametar funkcije podatak array2. U prilogu 3 opisani su izlazni patametri array2 i array.

Array2 je izlazni parametar sastavljen od dva arraya. Prvi array vratit će ukupan broj vraćenih podataka. Taj podatak je bitan za daljnju manipulaciju podataka na klijentskoj strani. Primjer navedenog je iteriranje kroz polje, ... i sl.

Array je izlazni parametar koji vraća sve podatke koje web servis odrađuje. Registrirane su izlazne varijable kaunt - kao broj ukupno vraćenih redova, vrijeme - kao podatak o vremenu spremljen u bazi, lane - kao podatak o prometnom traku, speed - o brzini vozila i direction - o smjeru kretanja vozila. Kao što je opisano u strukturi same tablice, unutar baze vidljivo je da nisu vraćeni svi podaci već samo određeni. Razlog tome je nevažnost samih podataka za krajnjega korisnika, ali potrebnih za normalno fukcioniranje sustava.

```
function podatak($limit,$radar) {

            $database = new
PDO('sqlsrv:Server=fpzsql.fpz.hr;Database=radar;ConnectionPooling=0',
'user', 'pass');
            $database->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);
            $sql="SELECT TOP (".$limit.") [id]
      ,[radar_id]
      ,[id_radara]
      ,[vrijeme]
      ,[speed]
      ,[lane]
      ,[direction]
      ,[length]
  FROM [radar].[dbo].[radar_stanje] where radar_id =".$radar;
            $stmt = $database->prepare($sql);
            $stmt->execute();
            $row = $stmt->fetchAll(PDO::FETCH_BOTH);
            $kaunt=$stmt->rowCount();
            foreach($row as $rows){
                  $polje[] = array(
            'vrijeme'=>$rows['vrijeme'],
            'speed'=>$rows['speed'],
            'lane'=>$rows['lane'],
            'direction'=>$rows['direction']
      );

            }

      return array($kaunt,$polje);

}
```

Prilog 4: Funkcija dohvata podataka

U prilogu 4 opisana je funkcija koja dohvaća podatke iz baze. SQL upitom prema Microsoft SQL bazi dohvaćaju se svi podaci iz baze. Varijable limit i radar su ulazne varijable i one poprimaju vrijednosti unesene u web servis kao upit. U sintaksi je vidljivo *TOP („.$limit.“)* koji određuje koliko redaka želimo da nam upit vrati. U uvjetnom dijelu upita određujemo koje podatke želimo prikazati.

Varijabla kaunt poprima ukupni broj redaka vraćene tablice. Navedena varijabla je bitna radi točnosti, budući da varijabla limit može sadržavati brojku veću no što u tablici postoji redaka. Primjer: ukoliko zatražimo *TOP (5000)* redaka dok tablica ima svega 2000 redaka, Varijabla kaunt prikazat će brojku od 2000.

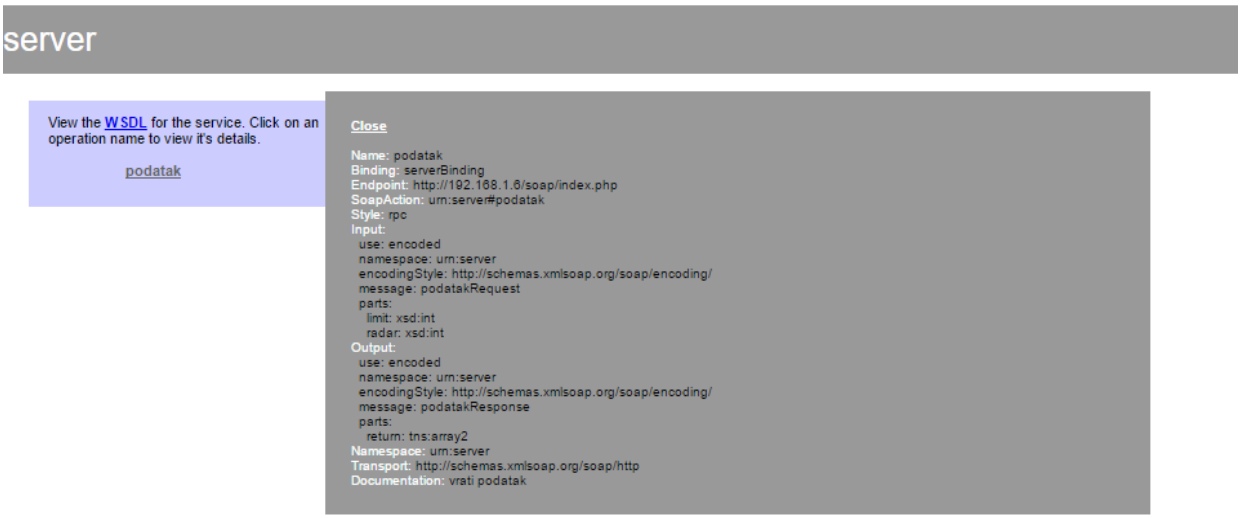
Iteracijom kroz vraćeni rezultat pomoću foreach petlje u polje nazvano polje pohranjuju se vrijednosti koje je potrebno prikazati. Da bi prikaz bio ispravan, broj elemenata polja treba biti jednak broju postavljenom u inicijalizaciji na početku samog web servisa.

Na kraju, uz pomoć returna vraćamo dobivene rezultate te se oni pomoću početno registriranog izlaza prikazuju u odgovoru web servisa.

6.2. Testiranje rada web servisa

Nakon kreiranja web servisa potrebno ga je pokrenuti na web serveru. Za tu funkciju pobrinut će se Apache i PHP koji su već instalirani na serveru. Prilikom postavljanja kreirane PHP datoteke obavezno je potrebno i postaviti biblioteku.

Ukoliko sve ispravno radi, u trenutku kada na web pregledniku otvorimo link, na adresi gdje je postavljen web servis, trebali bi dobiti prikaz kao na slici 15.



Slika 15: Izgled web servisa u pregledniku

Na početnom prikazu prvo su navedene sve funkcije koje sadrži web servis. U ovom slučaju web servis sadrži samo jednu funkciju - podatak. Klikom na link podatak otvara se

mali skočni prozor koji opisuje web servis. Tu se nalaze ulazni i izlazni parametri te nazivi akcija nad samom funkcijom.

Klikom na WSDL link otvara se automatski generiran Web Service Description Language, odnosno opis web servisa namijenjen za komunikaciju među računalima. WSDL datoteka prikazana je u Primjeru 3.

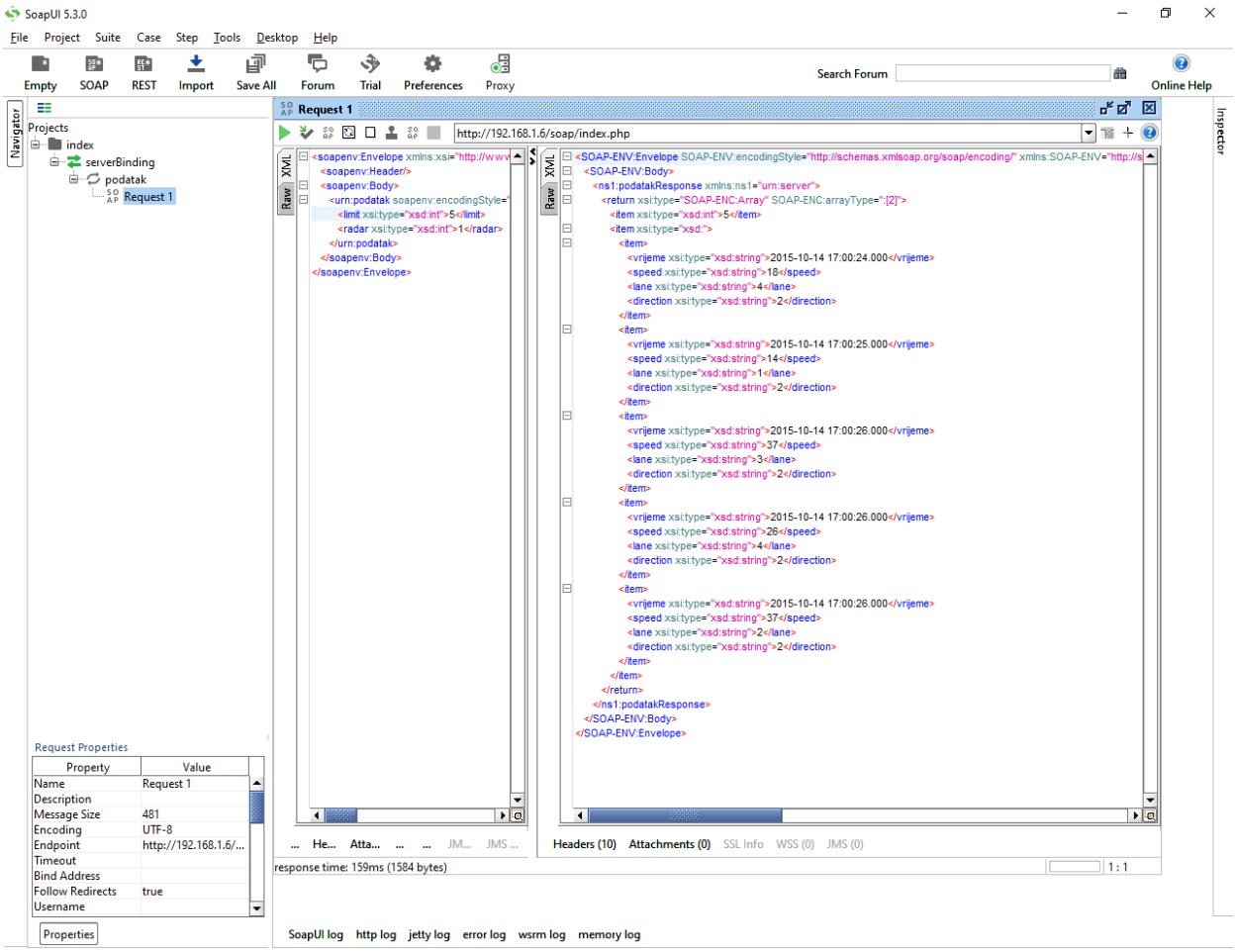
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:server"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="urn:server">
<types>
<xsd:schema targetNamespace="urn:server"
>
 <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
 <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
 <xsd:complexType name="Array">
  <xsd:complexContent>
   <xsd:restriction base="SOAP-ENC:Array">
  <xsd:all>
   <xsd:element name="kaunt" type="xsd:int"/>
   <xsd:element name="vrijeme" type="xsd:string"/>
   <xsd:element name="lane" type="xsd:int"/>
   <xsd:element name="speed" type="xsd:int"/>
   <xsd:element name="direction" type="xsd:int"/>
  </xsd:all>
   </xsd:restriction>
  </xsd:complexContent>
 </xsd:complexType>
 <xsd:complexType name="array2">
  <xsd:complexContent>
   <xsd:restriction base="SOAP-ENC:Array">
    <xsd:attribute ref="SOAP-ENC:arrayType" wsdl:arrayType="tns:Array[]"/>
   </xsd:restriction>
  </xsd:complexContent>
 </xsd:complexType>
</xsd:schema>
</types>
<message name="podatakRequest">
  <part name="limit" type="xsd:int" />
  <part name="radar" type="xsd:int" /></message>
<message name="podatakResponse">
  <part name="return" type="tns:array2" /></message>
<portType name="serverPortType">
  <operation name="podatak">
    <documentation>vrati podatak</documentation>
    <input message="tns:podatakRequest"/>
    <output message="tns:podatakResponse"/>
  </operation>
</portType>
```

```
<binding name="serverBinding" type="tns:serverPortType">
  <soap:binding style="rpc"
transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="podatak">
    <soap:operation soapAction="urn:server#podatak" style="rpc"/>
    <input><soap:body use="encoded" namespace="urn:server"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></input>
    <output><soap:body use="encoded" namespace="urn:server"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></output>
  </operation>
</binding>
<service name="server">
  <port name="serverPort" binding="tns:serverBinding">
    <soap:address location="http://192.168.1.6/soap/index.php"/>
  </port>
</service>
</definitions>
```

Primjer 3: Prikaz WSDL datoteke kreiranog web servisa

Testiranje rada web servisa provodimo na nekoliko načina. Konfiguracijom klijentske strane u PHP-u zatim pozivanjem servisa ili pomoću gotovih alata dostupnih na Internetu za testiranje web servisa. za testiranje web servisa postoji velik broj dostupnih alata, no u ovom slučaju, zbog njegove jednostavnosti i široke primjenjivosti, korišten je program SoapUI.

SoapUI je alat koji služi za testiranje rada web servisa. Mogu se koristiti SOAP i REST servisi. Prilikom pokretanja aplikacije potrebno je odrediti vrstu servisa. U ovom slučaju testira se SOAP servis. Nakon odabira SOAP web servisa potrebno je priložiti WSDL datoteku. Potom aplikacija generira sve funkcije web servisa. Potrebno je pokrenuti funkciju, u ovom slučaju podatak, te unijeti ulazne parametre. Nakon unosa ulaznih parametara pokreće se upit te nakon obrade web servis vraća rezultat. Na slici 16 prikazan je izgled aplikacije SoapUI te prikaz testiranja web servisa izrađenog za dohvat podataka.



Slika 16: Prikaz aplikacije SoapUI i testiranje web servisa

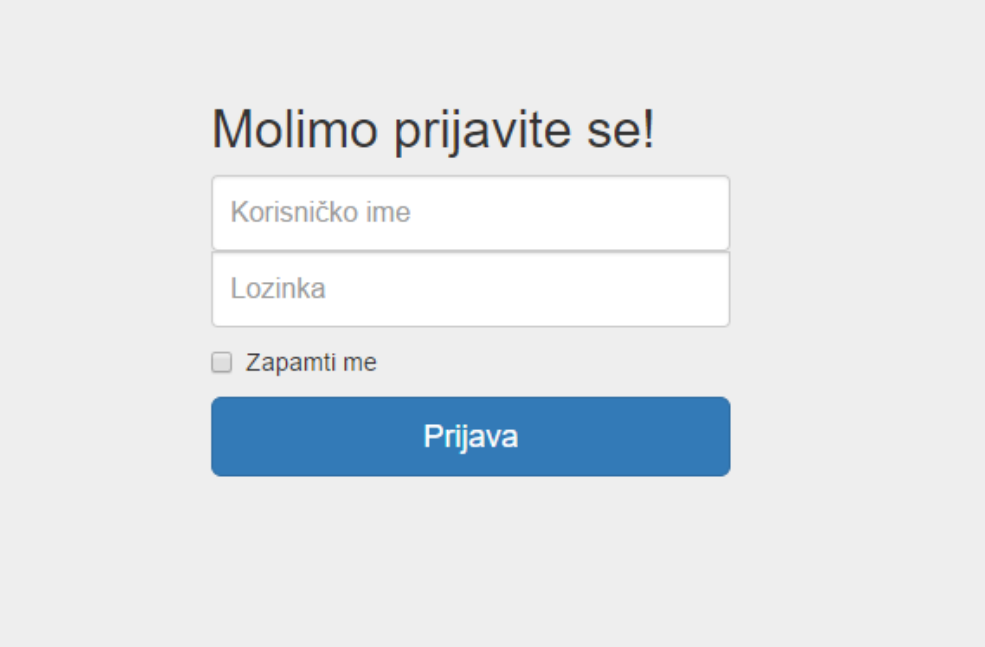
Iz priloženog je vidljivo da su upitom prema web servisu vraćeni podaci koji su bili zadani prilikom kreiranja samog web servisa.

7. Izrada PHP web aplikacije

Za potrebe prezentacije podataka iz web servisa, kao i za administraciju radara, kreirana je web aplikacija u PHP programskom jezik. Aplikacija sadrži login pomoću korisničkog imena i lozinke.Postoje dvije vrste prava pristupa: administratorsko i korisničko. Administrator ima mogućnost vidjeti stanje svih radara, dodavati i brisati radare te dodavati i brisati korisnike koji imaju pristup web aplikaciji, dok korisnik vidi samo stanje radara.

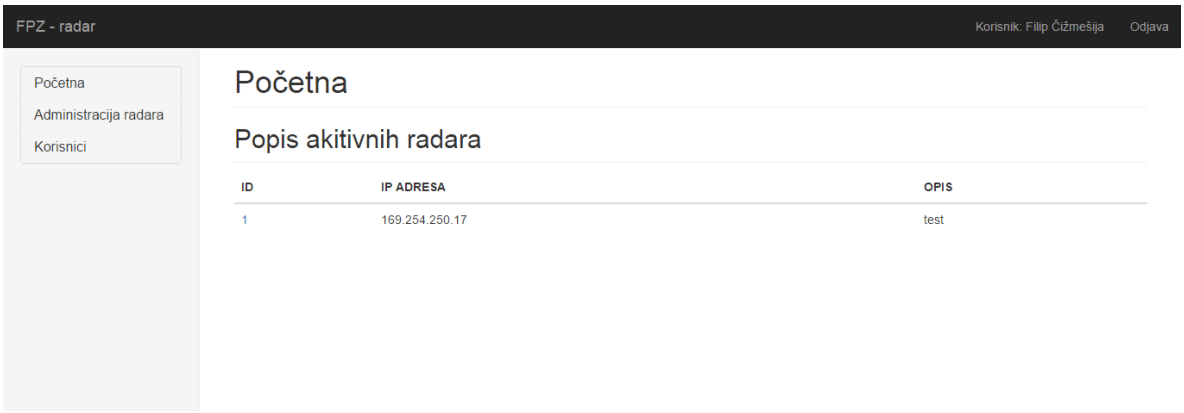
7.1. Izgled korisničkog sučelja

Prilikom pristupanja web adresi na kojoj se nalazi web aplikacija, prikazuje se login prikaz kao što je vidljivo na slici 17.



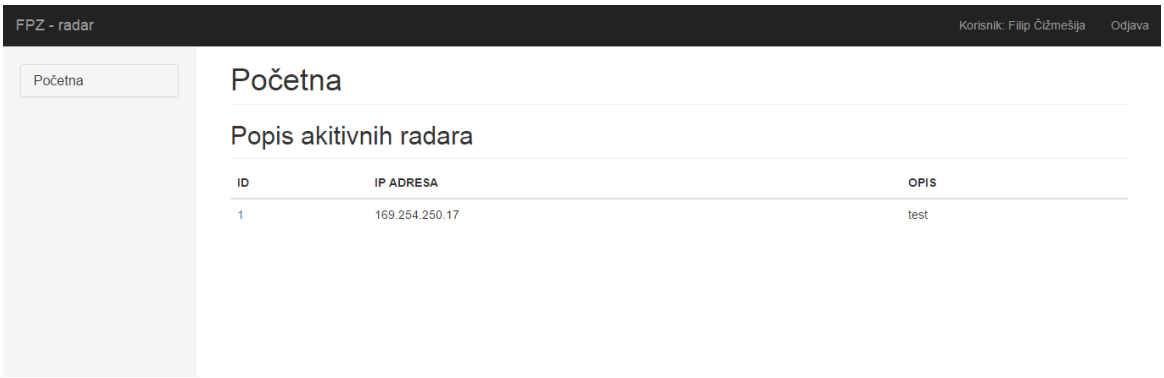
Slika 17: Izgled login forme

Nakon Uspješne prijave vidljiv nam je administratorski pogled prikazan na slici 18.



Slika 18: Administratorski prikaz

U lijevom izborniku vidljiva je navigacija koja sadrži početnu stranicu, administraciju radara te korisnike. Korisnički prikaz u navigaciji sadrži samo početnu stranicu kao i prava pristupa isključivo početnoj stranici. Prikaz nakon prijave u sustav s korisničkim pravima prikazan je na slici 19.

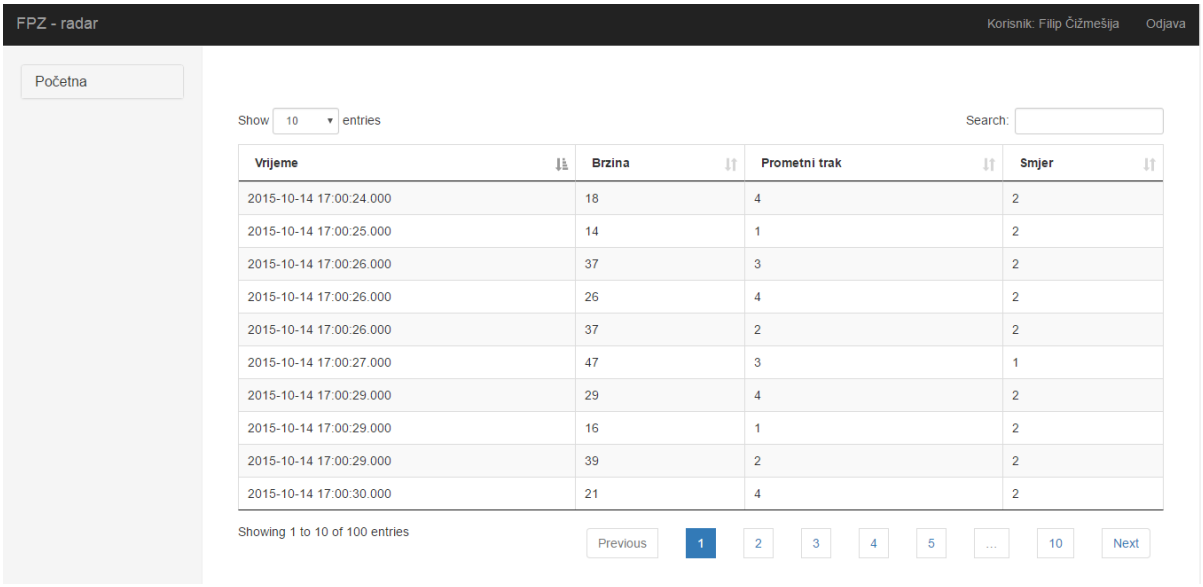


Slika 19: Korisnički prikaz

U segmentu početna stranica nalazi se popis svih aktivnih radara. U segmentu administracija radara također se nalazi popis svih radara uz mogućnost dodavanja novih, odnosno brisanja postojećih. Segment korisnici sadrži popis svih korisnika s njihovim pravima. U navedenom segmentu, omogućena je administracija svih postojećih korisnika kao i dodavanje novih.

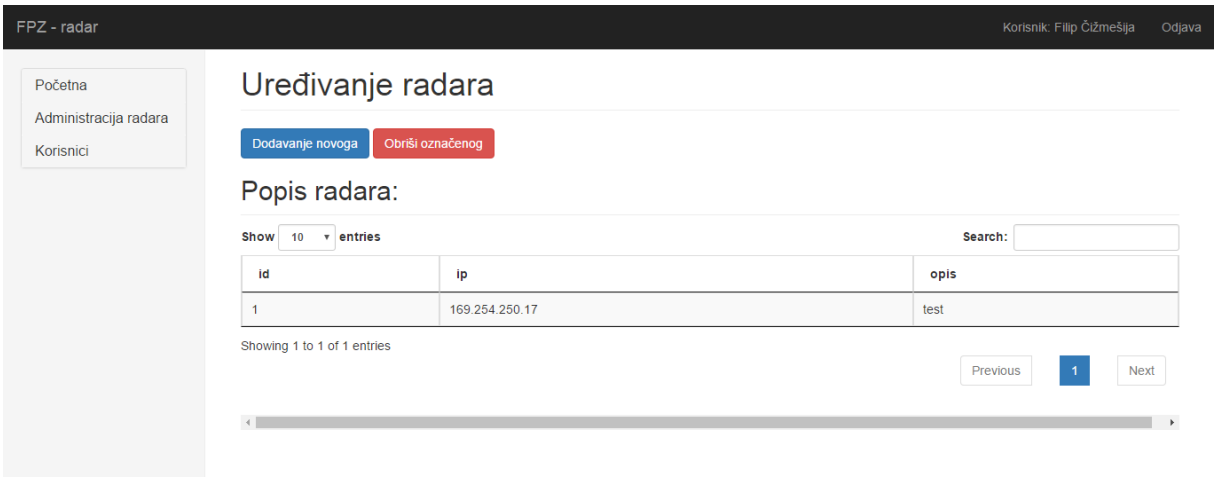
Na početnoj stranici nalazi se popis svih radara sa pripadajućim identifikacijskim oznakama. Prilikom odabiranja određene oznake (ID), u novoj kartici web preglednika

dobivamo sve dostupne podatke o navedenom radaru. Podaci koji su vidljivi u tom prikazu dohvaćeni su pomoću SOAP web servisa koji je ranije kreiran. Sadrži iste podatke, a to su vrijeme, brzina, prometni trak i smjer kretanja. Nužni parametri za dobivanje rezultata, za prethodno kreiran web servis, su id i limit. Pritom je ID odabran označavanjem dok je limit postavljen na 100 prikaza. Detaljan prikaz podataka vidljiv je na slici 20.

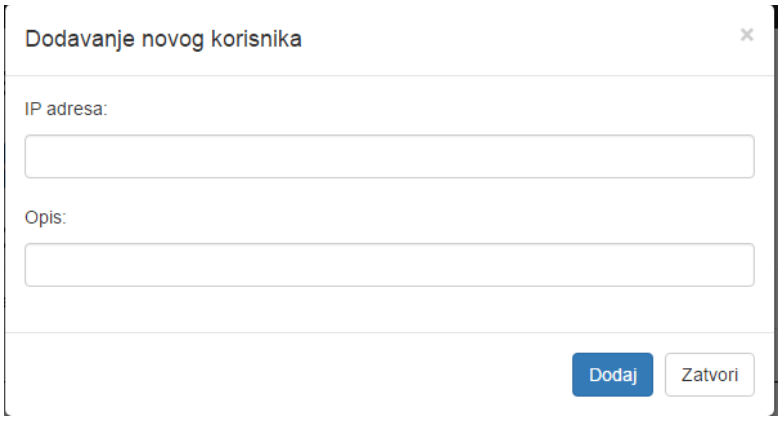


Slika 20: Detaljan prikaz podataka s radara

Segment administracije radara sadrži popis svih radara te mogućnost administracije istih. Radari se uklanjanju na način da se označi redak u tablici klikom na isti te nakon toga klikom na tipku obriši označenog. Dodavanje je po principu klika na gumb dodavanje novoga gdje se naknadno otvara skočni prozor u koji se unose podaci. Prilikom unosa, pootrebno je unijeti IP adresu ili DNS zapis radara kao i kratki opis. Izgled prozora za uređivanje i prikaz skočnog prozora za unos podataka prikazan je na slikama 21 i 22.

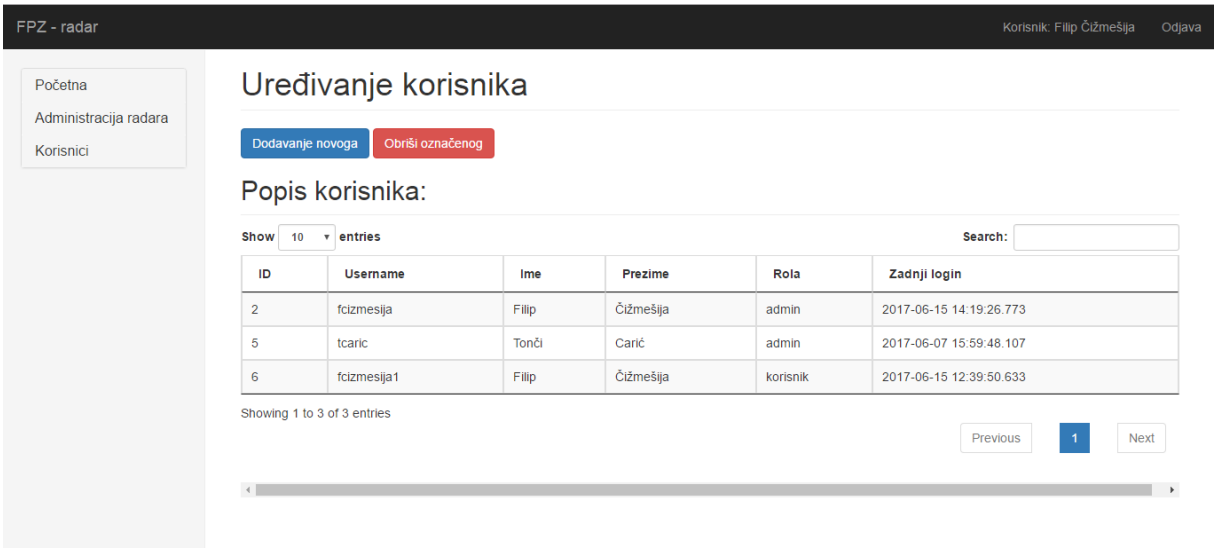


Slika 21: Prikaz aktivnih radara i mogućnost uređivanja

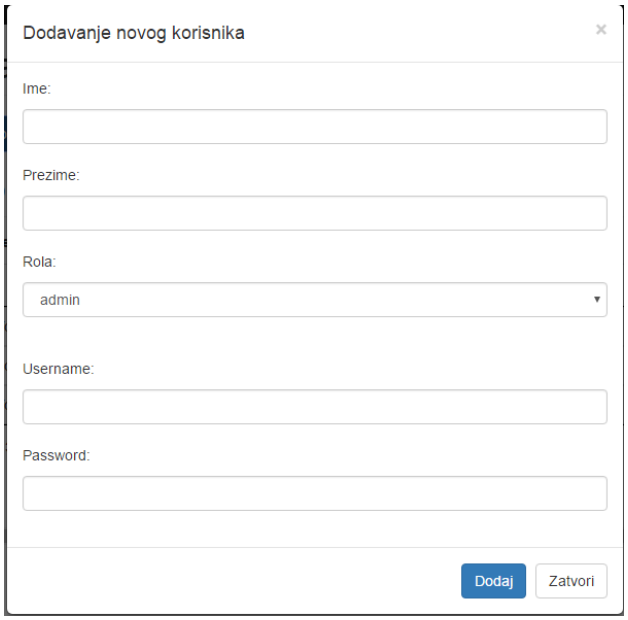


Slika 22: Skočni prozor za dodavanje novog radara

Segment korisnika sadrži popis korisnika, funkcionalnost brisanja postojećega korisnika kao i funkcionalnost dodavanja novog korisnika. Postupak brisanja je isti kao i postupak brisanja radara. Dodavanje novih korisnika također je isto kao i u slučaju dodavanja novih radara. Na slikama 23 i 24 prikazan je izgled segmenta uređivanja korisnika kao i dodavanja novog.



Slika 23: Prikaz korisnika sustava



Slika 24: Skočni prozor za dodavanje novog korisnika

7.2. Funkcionalnost web aplikacije

Kao što je ranije, navedeno za dohvat podataka s radara koristi se web servis. U web aplikaciji korištena je NuSOAP biblioteka, klijentska verzija za dohvat podataka. U prilogu 5 prikazan je PHP kod za dohvat podataka s web servisa.

```
require_once „../lib/nusoap.php“;

$client = new nusoap_client(„http://localhost/soap/?wsdl“, true);
$error  = $client->getError();

if ($error) {
    echo „<h2>Constructor error</h2><pre>“ . $error . „</pre>“;
}

$result = $client->call(„podatak“, array(„limit“ => „100“,“radar“=>$id));

echo '<table id=“example“ class=“table table-striped table-bordered“
cellspacing=“0“ width=“100%“>';
echo ' <thead>
  <tr>
     <th>Vrijeme</th>
     <th>Brzina</th>
     <th>Prometni trak</th>
     <th>Smjer</th>
  </tr>
 </thead>
 <tbody>';
for($i=0;$i<$result[0];$i++){
      echo '<tr>
      <td>'.$result[1]['item'][$i]['vrijeme'].'</td>
      <td>'.$result[1]['item'][$i]['speed'].'</td>
      <td>'.$result[1]['item'][$i]['lane'].'</td>
      <td>'.$result[1]['item'][$i]['direction'].'</td>
      </tr>
      ';

}
echo '</tbody><table>';
```

Prilog 5: Kod za dohvat podataka s web servisa

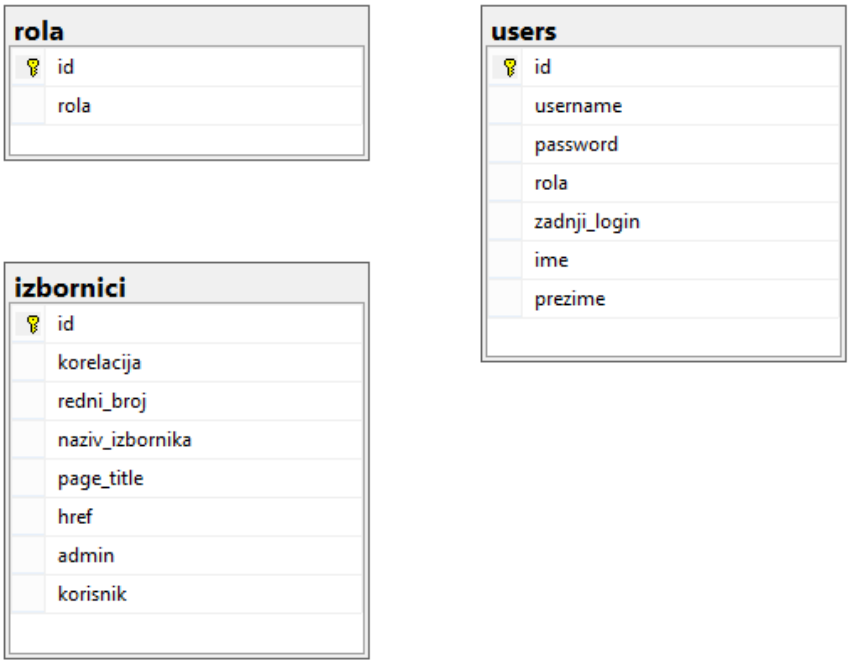
Nakon prilaganja biblioteke potrebno je kreirati objekt client. Konstruktor objekta je nusoap_client kojemu je ulazni parametar WSDL web servisa. Na temelju WSDL datoteke biblioteka zna koji su ulazni, a koji izlazni parametri. Nakon inicijalizacije poziva se upit prema web servisu. Prilikom pozivanja potrebno je odabrati koju se funkciju poziva te navesti ulazne parametre za upit. U ovom slučaju funkcija je podatak, a ulazni parametri su limit i id

radara. Id radara dobivamo preko linka koji je odabran iz pomoć GET medote, dok je limit postavljen na 100 rezultata.

Po završetku dohvata slijedi ispis tablice iteriranjem kroz rezultat koji se nalazi u polju result.

7.3. Tablice unutar baze podataka korištene za rad web aplikacije

Unutar postojeće baze podataka imena radar korištene su preostale tri tablice. Te tablice služe za omogućavanje normalnog rada web aplikacije. Na slici 25 prikazana je struktura tih tablica, a u tablicama 3, 4 i 5 njihov detaljan opis.



Slika 25: Tablice korištene za rad web aplikacije

NAZIV ATRIBUTA	VRSTA ATRIBUTA	KRATKI OPIS
ID	INT	Identifikator
ROLA	NVARCHAR	Popis dostupnih rola

Tablica 3: Opis SQL tablice „rola“

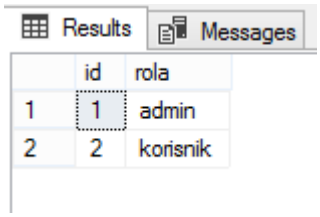
NAZIV ATRIBUTA	VRSTA ATRIBUTA	KRATKI OPIS
ID	INT	Identifikator
KORELACIJA	INT	Veza s glavnim izbornikom
REDNI_BROJ	INT	Poredak izbornika
NAZIV_IZBORNIKA	NVARCHAR	Naziv izbornika
PAGE_TITLE	NVARCHAR	<title> web stranice
HREF	NVARCHAR	Web adresa izbornika
ADMIN	INT	Vidljivost ovisno o roli
KORISNIK	INT	Vidljivost ovisno o roli

Tablica 4: Opis SQL tablice „izbornici“

NAZIV ATRIBUTA	VRSTA ATRIBUTA	KRATKI OPIS
ID	INT	Identifikator
USERNAME	NVARCHAR	Korisničko ime
PASSWORD	NVARCHAR	Lozinka
ROLA	NVARCHAR	Korisnička rola
ZADNJI_LOGIN	DATETIME	Vrijeme zadnje prijave
IME	NVARCHAR	Ime korisnika
PREZIME	NVARCHAR	Prezime korisnika

Tablica 5: Opis SQL tablice „users“

Na slikama 26, 27 i 28 prikazan je sadržaj pojedine tablice.



Slika 26: Prikaz tablice rola

Results | Messages

	id	korelacija	redni_broj	naziv_izbornika	page_title	href	admin	korisnik
1	1	1	1	Početna	NULL	NULL	1	0
2	2	1	2	Početna	Početna	index.php	1	0
3	3	3	100	nevidljivi	NULL	NULL	0	0
4	4	3	100	Radar	Radar	radar.php	1	0
5	5	5	10	Korisnici	NULL	NULL	1	0
6	6	5	11	Korisnici	Korisnici	korisnici.php	1	0

Slika 27: Prikaz tablice izbornici

Results | Messages

	id	username	password	rola	zadnji_login	ime	prezime
1	2	fcizmesija	9922EB1FDCBB32A0EA01F2D8C6E2339A	admin	2017-06-15 12:20:55.290	Filip	Čižmešija

Slika 28: Prikaz tablice users

8. Sklopovska podrška i shema spajanja

Cijeli sustav sastoji se od nekoliko različitih elemenata. Ti elementi služe za komunikaciju, pohranu podataka, obradu podataka i dohvat podataka.

Za cijeli sustav korišteno je kao što je prikazano na slici 29:

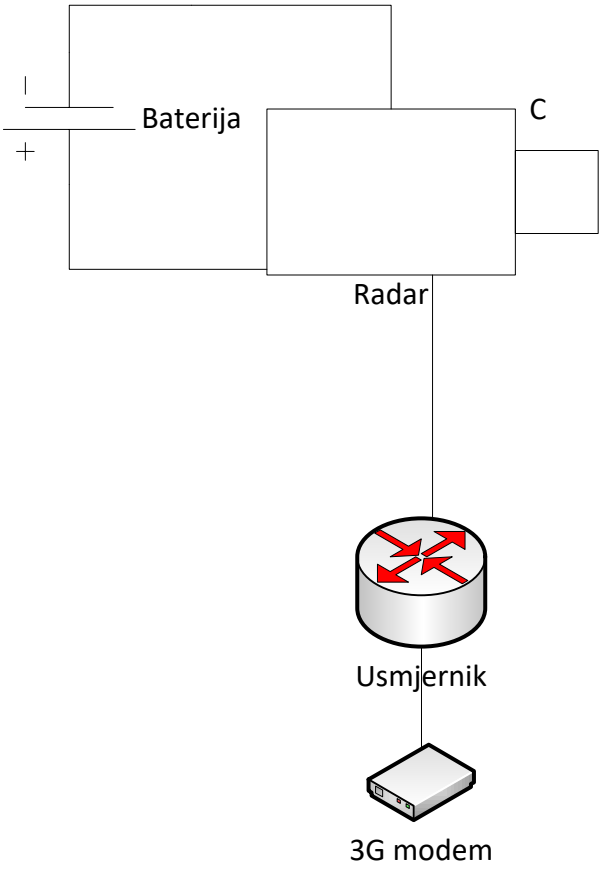
a) Houston Radar SpeedLane
b) Baterijski sustav napajanja radara
c) TP-LINK MR3040 3G/4G usmjernik
d) ZTE MF100 3G USB Modem

Na strani poslužitelja korišteno je:

- Server s instaliranom bazom podataka
- Server s instaliranim web serverom

Slika 29: Spoj radara s mrežnom opremom

Na slici 30 prikazan je shematski spoj opreme.



Slika 30: Shematski spoj opreme

Serveri na kojima se nalazi baza i web servis, odnosno web aplikacija, međusobno su povezani. Također, zajedno su povezani na Internet. Radarski sustav spojen je mrežnim kabelom na usmjernik. Na usmjerniku se nalazi 3G modem koji uspostavlja konekciju prema Internetu. Putem Interneta je ostvarena međusobna komunikacija između servera i radara. Za spajanje na radar koristi se port 23 za TELNET. Na usmjerniku je podignut Dynamic DNS servis. On služi za komunikaciju u DNS zapisu zbog dinamičke IP adrese mrežnog operatera. Korišten je besplatni servis operatera No-IP.

9. Zaključak

Potreba za mjerenjem prometnog toka i zagušenja cestovnog prometa pojavljuje se radi unaprjeđenja sigurnosti prometnih entiteta, ali i radi povećanja kvalitete usluge. Radarski sustav Houston Radar služi za mjerenje prometa na cestovnim prometnicama. Proizvođač daje vlastito razvijeni sustav za daljinsko očitavanje podataka uz mjesečnu pretplatu. Ovaj sustav je razvijen s ciljem objedinjavanja podataka s više radarskih sustava bez plaćanja pretplate. Sami sustav koji je razvijen nema ograničenja koliko radara se može povezati no optimalno bi bilo do oko 100 radara. Podatke je moguće prikupljati neovisno o tome gdje je radar postavljen zahvaljujući pristupu putem mobilne mreže.

Podaci se nalaze na centraliziranom serveru gdje je moguće iste i obrađivati. Zahvaljujući izradi web servisa podatke je moguće implementirati u bilo koji drugi sustav. Web servis može se koristiti osim trenutno na web aplikaciji, za pristup putem aplikacije za pametne telefone ili neke računalne aplikacije neovisno o operativnom sustavu i uređaju.Web servis i web aplikacija dostupni su unutar interne mreže Fakulteta prometnih znanosti. Izradom ovog sustava, izvedenog pomoću web servisa, omogućena je centralizirana obrada podataka s više radarskih sustava.

Literatura

[1] Databaseanswers.org. (2017). History of Web Services. [online] Available at: http://www.databaseanswers.org/web_services_history.htm [Accessed 15 Jun. 2017]

[2] Mitchell, L. (2016). PHP web services. 2nd ed. Sebastopol: O'Reilly Media, p.66

[3] Fakultet prometnih znanosti. (2017). Uvod u relacijske baze podataka. [online] Available at: http://files.fpz.hr/Djelatnici/tcaric/Tonci-Caric-Baze-podataka.pdf [Accessed 30 Aug. 2017].

[4] Msdn.microsoft.com. (2017). Transact-SQL Reference (Transact-SQL). [online] Available at: https://msdn.microsoft.com/en-us/library/ms189826.aspx [Accessed 15 Jun. 2017].

[5] Sqlite.org. (2017). About SQLite. [online] Available at: https://www.sqlite.org/about.html [Accessed 15 Jun. 2017].

[6] Houston-radar.com. (2017). Houston Radar - The Next Generation Ultra Low Power Traffic Calming Speed Radar. [online] Available at: http://houston-radar.com/speedlane.php [Accessed 15 Jun. 2017].

[7] URL: https://msdn.microsoft.com/en-us/library/esw638yk(v=vs.100).aspx , pristupljeno lipanj, 2017

[8] URL: https://www.chemaxon.com/products/jchem-web-services/, pristupljeno lipanj, 2017

[9] URL: http://home.hit.no/~hansha/documents/software/software_development/topics/software_architecture.htm, pristupljeno lipanj, 2017

Popis kratica

DNS	Domain Name System
HTTP	HyperText Transfer Protocol
LAN	Local area network
PHP	PHP: Hypertext Preprocessor
RESTful, REST	Representational state transfer
RPC	Remote procedure call
SDK	Software development kit
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
WSDL	Web Services Description Language
XML	EXtensible Markup Language

Popis slika

Slika 1: Prikaz rada SOAP web servisa [6]

Slika 2: Prikaz rada RESTful web servisa [7]

Slika 3: Prikaz pristupa bazi putem web servisa [8]

Slika 4: Shematski prikaz baza podataka

Slika 5: Houston Radar SpeedLane

Slika 6: Dijagram toka dohvata podataka

Slika 7: Konfiguracija Task Scheduler-a

Slika 8: Metoda za dohvat liste svih radara

Slika 9: Metoda za uspostavu konekcije

Slika 10: Dohvat i pohrana podataka

Slika 11: Tablice unutar baze namijenjene za pohranu podataka

Slika 12: Prikaz tablice „radar_info“

Slika 13: Prikaz tablice „radar_stanje“

Slika 14: Shematski prikaz baze podataka i web servera

Slika 15: Izgled web servisa u pregledniku

Slika 16: Prikaz aplikacije SoapUI i testiranje web servisa

Slika 17: Izgled login forme

Slika 18: Administratorski prikaz

Slika 19: Korisnički prikaz

Slika 20: Detaljan prikaz podataka s radara

Slika 21: Prikaz aktivnih radara i mogućnost uređivanja

Slika 22: Skočni prozor za dodavanje novog radara

Slika 23: Prikaz korisnika sustava

Slika 24: Skočni prozor za dodavanje novog korisnika

Slika 25: Tablice korištene za rad web aplikacije

Slika 26: Prikaz tablice rola

Slika 27: Prikaz tablice izbornici

Slika 28: Prikaz tablice users

Slika 29: Spoj radara s mrežnom opremom

Popis tablica

Tablica 1: Opis SQL tablice „radar_info“

Tablica 2: Opis SQL tablice „radar_stanje“

Tablica 3: Opis SQL tablice „rola“

Tablica 4: Opis SQL tablice „izbornici“

Tablica 5: Opis SQL tablice „users“

Popis primjera

Primjer 1: SOAP Request

Primjer 2: SOAP Response

Primjer 3: Prikaz WSDL datoteke kreiranog web servisa

Popis priloga

Prilog 1: Inicijalizacija NuSOAP biblioteke

Prilog 2: Definiranje ulaznih parametara

Prilog 3: Definiranje izlaznih parametara

Prilog 4: Funkcija dohvata podataka

Prilog 5: Kod za dohvat podataka s web servisa

Sveučilište u Zagrebu
Fakultet prometnih znanosti
10000 Zagreb
Vukelićeva 4

IZJAVA O AKADEMSKOJ ČESTITOSTI I SUGLASNOST

Izjavljujem i svojim potpisom potvrđujem kako je ovaj završni rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu što pokazuju korištene bilješke i bibliografija.

Izjavljujem kako nijedan dio rada nije napisan na nedozvoljen način, niti je prepisan iz necitiranog rada, te nijedan dio rada ne krši bilo čija autorska prava.

Izjavljujem također, kako nijedan dio rada nije iskorišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

Svojim potpisom potvrđujem i dajem suglasnost za javnu objavu završnog rada pod naslovom **IZVEDBA DALJINSKOG PRISTUPA BAZI PODATAKA RADARSKOG SUSTAVA PUTEM WEB SERVISA**

na internetskim stranicama i repozitoriju Fakulteta prometnih znanosti, Digitalnom akademskom repozitoriju (DAR) pri Nacionalnoj i sveučilišnoj knjižnici u Zagrebu.

Student/ica:

U Zagrebu, 1.9.2017

(potpis)