UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Miha Vidmar

Informatizacija procesov v patronažnem varstvu

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM PRVE STOPNJE

RAČUNALNIŠTVO IN INFORMATIKA

Mentor: doc. dr. Mira Trebar

Ljubljana, 2014

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00152 / 2013
Datum: 15.9.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MIHA VIDMAR**

Naslov: **INFORMATIZACIJA PROCESOV V PATRONAŽNEM VARSTVU**
INFORMATIZATION OF HOME CARE PROCESSES

Vrsta naloge: Diplomsko delo univerzitetnega študija prve stopnje

Tematika naloge:

V zdravstvu obstajajo različne informacijske rešitve, ki omogočajo elektronsko obdelavo podatkov. Pri tem pa se v patronažnem varstvu še vedno uporablja papirni delovni nalog, ki vključuje individualno pripravo urnikov z navodili in seznamom storitev za obiske pacientov na domu. Kandidat naj v diplomskem delu analizira obstoječe rešitve procesa obdelave delovnega naloga, določi ustrezne tehnološke rešitve in predlaga postopek avtomatskega pridobivanja podatkov o zahtevanih storitvah patronažnega varstva. Na osnovi pridobljenih podatkov iz zdravstvenega kartona pacienta naj opiše postopke in pripravi podatke za sinhronizacijo in podporo dela na terenu tako, da je možno z mobilno aplikacijo avtomatsko izvesti celoten proces obdelave. Spletna aplikacija in programski vmesniki naj bodo izdelani v .NET razvojnem okolju. Vključena mora biti tudi možnost vnosa in obdelave testnih podatkov za izvedbo testiranja v povezavi z mobilno aplikacijo, ki je bila izdelana v drugi diplomski nalogi.

Mentor:
Mira Trebar
doc. dr. Mira Trebar

Dekan:
prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisani/-a **Miha Vidmar**,

z vpisno številko 63000309

sem avtor diplomskega dela z naslovom:

Informatizacija procesov v patronažnem varstvu

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom

  doc. dr. Mire Trebar

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne _____ Podpis avtorja/-ice: _____

Zahvala

Iskrena hvala mentorici doc. dr. Miri Trebar za mentorstvo in pomoč, staršem in punci Maji za zaupanje in podporo. Posebna zahvala pa tudi sodelavkam v podjetju, ker so verjele, da se čudeži dogajajo.

Kazalo

1 Uvod 1
2 Informatizacija terenskega dela patronažnega varstva 3
2.1 Analiza potrebnih funkcionalnosti in vmesnikov 3
2.2 Podatki v sistemu patronažnega varstva 5
2.3 Primeri uporabe in potrebne storitve 8
3 Razvoj sistema za patronažno varstvo 11
3.1 Arhitektura sistema 11
3.2 Varnost sistema 13
3.3 Sinhronizacija 14
3.4 Tehnologije in orodja 15
4 Izvedba in delovanje sistema 17
4.1 Demo aplikacija 17
4.2 Programski vmesnik za integracijo v zdravstveni sistem 20
4.3 Skrbniška aplikacija 21
4.4 Programski vmesnik za mobilne naprave 26
5 Sklepne ugotovitve 31
6 Literatura 33

Seznam kratic in simbolov

API	Application Programming Interface
C#	programming language based on .NET Framework
CSS	Cascading Style Sheet
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
MVC	Model View Controller
.NET	Microsoft development framework
NoSQL	Not Only SQL
REST	REpresentational State Transfer
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer
TLS	Transport Layer Security
WCF	Windows Communication Foundation
WPF	Windows Presentation Foundation
WSDL	Web Services Description Language
XML	Extensible Markup Language

Povzetek

Diplomsko delo opisuje strežniški sistem za informatizacijo terenskega dela patronažne službe. Rešitev je izdelana kot vmesni člen med mobilno aplikacijo in obstoječimi zalednimi zdravstvenimi sistemi. Sestavljena je iz treh glavnih aplikacij: skrbniška aplikacija, programski vmesnik za integracijo z mobilno aplikacijo in programski vmesnik za integracijo v zaledni zdravstveni sistem. Podpira delo z uporabniki (patronažne sestre) in delovnimi nalogi, ki služijo kot evidenca storitev patronažne službe in se trenutno uporabljajo v papirnati obliki. Rešitev je bila razvita v tehnologiji Microsoft .NET, uporablja programske vmesnike REST za integracijo z mobilno aplikacijo in programske vmesnike SOAP za integracijo z zalednim zdravstvenim sistemom. Za testiranje in simulacijo celotnega procesa in toka podatkov sta bili razviti še dve dodatni komponenti, to sta avtomatski testi, ki simulirajo delovanje mobilne aplikacije in demo aplikacija, ki simulira zaledni zdravstveni sistem in preko katere lahko vnašamo testne podatke v naš sistem. Za delovanje celovitega sistema je bila razvita tudi mobilna aplikacija, ki pa je predmet druge diplomske naloge. Skupaj tako predstavljata osnovo za nadaljnji razvoj celovitega sistema za informatizirano podporo terenskega dela patronažnega varstva.

Ključne besede:

Patronažna služba, delovni nalog, integracija, zdravstveni sistem, informatizacija, spletna aplikacija

Abstract

This thesis describes the complete server side system for computerisation of home care service. The solution is built as an intermediate layer between the mobile application and the existing backend of the health institution. It consists of three main applications: custody application, web service for integration with the mobile application, and web service for integration with the backend of the health institution. It supports working with the users (nurses in home care) and work orders which act as a basis for which home care services need to be provided and are still handled in paper form. The solution is built on Microsoft .NET framework, provides a REST API for communication with the mobile app, and SOAP API for communicating with the health institution backend. For testing and simulation purposes of working within the entire process flow, two additional components were implemented, automated tests which cover the REST API and simulate the mobile application, and a demo smart client are used to enter the information into the system and simulate the backend. To cover the entire field work process, a mobile application was developed as part of another thesis. Together they represent the basis for creating a product that will enable complete end-to-end computerisation of home care service.

Key words:

Home care service, work order, integration, health system, computerisation, web application

1 Uvod

Informatizacija in ukinjanje papirnatega poslovanja prinašajo veliko pozitivnih stvari, kot so večja produktivnost, takojšnje informacije, boljši nadzor, večjo varnost podatkov in zmanjšanje napak.

V zadnji letih se pojavlja veliki trend na področju razvoja mobilnih rešitev. Mobilnost dobiva v svetu izjemno velike razsežnosti. Ker so pametne naprave sedaj že skoraj povsod uporabljene in predvsem podprte z vseprisotno povezljivostjo – hitri dostop do interneta imamo sedaj na voljo praktično povsod – se spreminja tudi obnašanje ljudi in vpliv mobilnih naprav na njihovo delo.

Informatizacija in prehod na mobilne naprave se pojavlja tudi v zdravstvu po svetu in v Sloveniji. Trenutno se v zdravstvu informatizira veliko storitev, vendar predvsem v okviru dela znotraj bolnišnic, zdravstvenih domov itn. Proces dela na terenu se še vedno izvaja na podlagi papirjev in je, kot smo ugotovili v okviru te diplome, ostal večinoma nespremenjen, kljub spremembam v tehnologiji.

Zato smo se odločili za pregled dela patronažnih sester, ker je eden izmed takih procesov, ki pri nas v Sloveniji še vedno poteka na papirju. Patronažno varstvo je opisano kot [1]: »Patronažno varstvo je posebna oblika zdravstvenega varstva, ki opravlja aktivno zdravstveno in socialno varstvo na varovancih v družini in skupnosti, ki so zaradi bioloških lastnosti ali zaradi določenih obolenj še posebej dovzetni za škodljive vplive iz okolja (definicija Svetovne zdravstvene organizacije).«

Poleg samega procesa dela patronažnih sester smo pregledali tudi delovanje informacijskih sistemov v zdravstvenih domovih. Zanimalo nas je, katere podatke se trenutno zbira za delo patronažnega varstva in pa predvsem, kako so postavljeni informacijski sistemi v zdravstvu v Sloveniji, da bi lažje predvideli umestitev naše rešitve v okvir teh sistemov.

Kot rešitev smo vzpostavili celovit sistem za informatizacijo terenskega dela patronažnega varstva, od integracije do zalednega sistema do mobilne aplikacije. V nalogi je predstavljen razvoj strežniškega dela sistema. Sestavljajo ga tri aplikacije: skrbniška aplikacija in dva programska vmesnika, ki omogočata integracijo z mobilno aplikacijo in zalednim zdravstvenim sistemom. Za testiranje in simulacijo pa sta bili razviti še komponenti za izvajanje avtomatskih testov v simulaciji mobilne aplikacije in demo aplikacija za vnos testnih podatkov o pacientih in delovnih nalogih v naš sistem.

2 Informatizacija terenskega dela patronažnega varstva

Patronažne sestre trenutno uporabljajo delovne naloge v papirnati obliki hodijo po terenu in si zapisujejo opravljene storitve. Ko se vrnejo, ročno vnašajo te podatke v zdravstvene informacijske sisteme. Na tak način izgubijo veliko časa, določeni nalogi se lahko izgubijo, pri samem pretipkavanju podatkov pa lahko prihaja do napak. V skladu s trendom mobilnosti in ukinjanja papirnatega poslovanja na vseh področjih življenja smo se odločili, da informatiziramo terensko delo patronažnih sester z mobilno aplikacijo.

Sistem sestavljata dva glavna dela:

- Mobilna aplikacija za sistem Android, ki je bila razvita v okviru diplomskega dela z naslovom Uporaba pametnih naprav za obdelavo podatkov v patronažnem varstvu [2];
- Strežniški sistem, ki je bil razvit v okviru te diplomske naloge in zagotavlja spletne storitve, preko katerih bo mobilna aplikacija pridobivala potrebne podatke za njeno delovanje. Sestavljen je iz:
    - programskega vmesnika za mobilno napravo,
    - skrbniške aplikacije za pregled sistema,
    - programskega vmesnika za integracijo v zdravstveno okolje in
    - demo aplikacije za simulacijo integracije zdravstvenega okolja.

Celovita rešitev naj bi omogočala izboljšavo procesa z uporabo obstoječe infrastrukture zdravstvenega zavoda in pametnih naprav, to se mobilni telefoni, ali pa tudi tablični računalniki. Mobilna aplikacija je namenjena poenostavitvi in optimizaciji terenskega dela patronažnih sester. Mi pa smo se z naši delom osredotočili na razvoj strežniškega dela za sinhronizacijo podatkov med zdravstvenim sistemom in mobilno aplikacijo, ki je potrebna za podporo njenega delovanja.

2.1 Analiza potrebnih funkcionalnosti in vmesnikov

Pred pričetkom izdelave smo si po pogovorih s patronažnimi sestrami in analizo dokumentacije obstoječih zdravstvenih sistemov (trenutno informatizirani del patronažnega varstva) pripravili seznam možnih primerov uporabe in vseh podatkov, ki se bodo prenesli na mobilno aplikacijo.

Analiza vključuje:

- pregled in opis trenutnega dela in podatkov, ki se uporabljajo v patronažnem varstvu,
- analizo možne umestitve rešitve v obstoječe okolje zdravstvenih sistemov,
- pripravo potrebnih funkcij za programski vmesnik do zdravstvenega sistema,
- definicijo okvirnih funkcionalnosti skrbniške aplikacije,
- pripravo potrebnih funkcij na strežniku za podporo in uskladitev delovanja z mobilno aplikacijo.

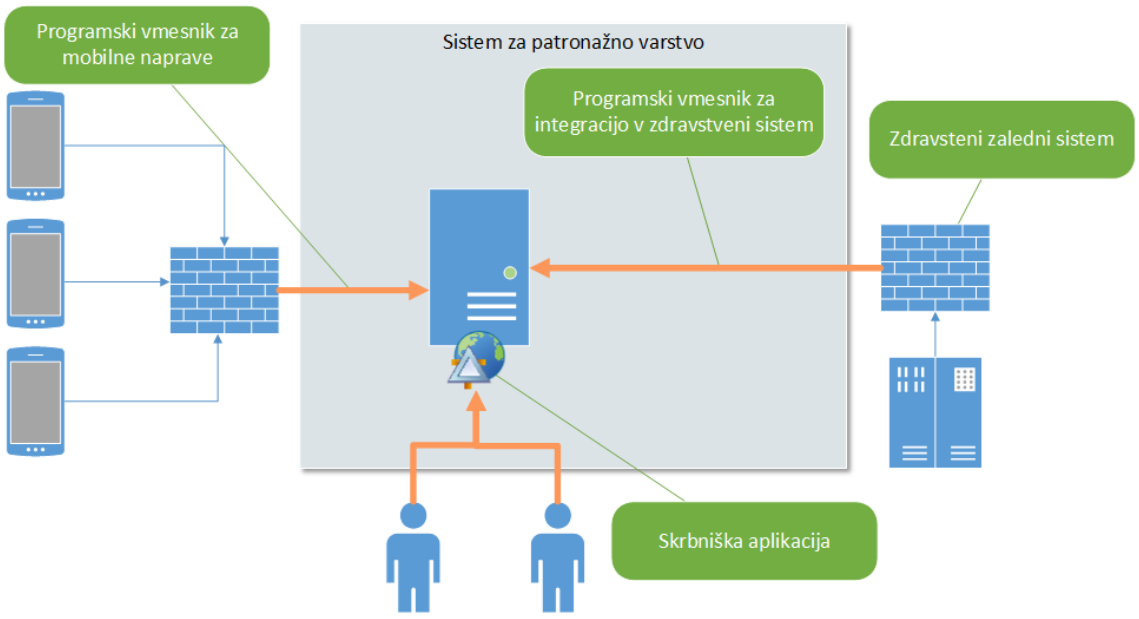
Osnovo za definicijo zahtev strežniškega dela smo pripravili v povezavi s potrebami mobilne aplikacije na podlagi pregleda trenutnega postopka dela patronažnih sester:

- Ob prihodu v ambulanto - mobilna aplikacija opravi sinhronizacijo obiskov samo za tekoči dan ali tudi za prihodnje dni (odločitev je prepuščena implementaciji mobilne aplikacije, strežnik podpira iskanje po datumu).
- Patronažna sestra odide na teren – prikažemo urnik obiskov in domače naslove pacientov.
- Patronažna sestra pride do pacienta:
    - Začetek obiska:
        - Zgodovina obiskov pacienta in podatki o pacientu.
        - Kontaktni podatki (telefon, bližnji člani, itn.).
        - Podatki o delovnem nalogu.
    - Zaključek obiska:
        - Posodobitev podatkov o delovnem nalogu.
        - Zaključevanje delovnega naloga.
        - Vnos potrebnih podatkov za poročanje zavarovalnicam, itn.
- Prihod sestre nazaj v ambulanto - sinhronizacija podatkov v mobilni aplikaciji nazaj na strežnik.

Zanimalo nas je tudi kako umestiti strežniško rešitev v obstoječe informacijsko okolje zdravstvenih ustanov. V večini javnih institucij se dogaja konsolidacija sistemov v eno glavno aplikacijo, saj to po večini omogoča bolj učinkovito delovanje in upravljanje, ker so vsi podatki in procesi vodeni v enem sistemu. Zato smo se pri snovanju strežniške rešitve osredotočili na možnost, da rešitev ne bo vključena v zdravstveno ustanovo kot samostojna rešitev, ampak bo integrirana z obstoječimi rešitvami. Predlagana umestitev v obstoječi zdravstveni sistem je predstavljena na Slika 1.

Za vnos in popravljanje delovnih nalogov in uporabnikov ni predvidena skrbniška funkcionalnost, ampak bo namesto tega uporabljen programski vmesnik za integracijo. Skrbniška aplikacija tako omogoča samo enostavne preglede in razširitev s funkcionalnostmi,

ki so specifične za uporabo na mobilni aplikaciji (npr. izdaja kod za inicializacijo mobilne aplikacije).

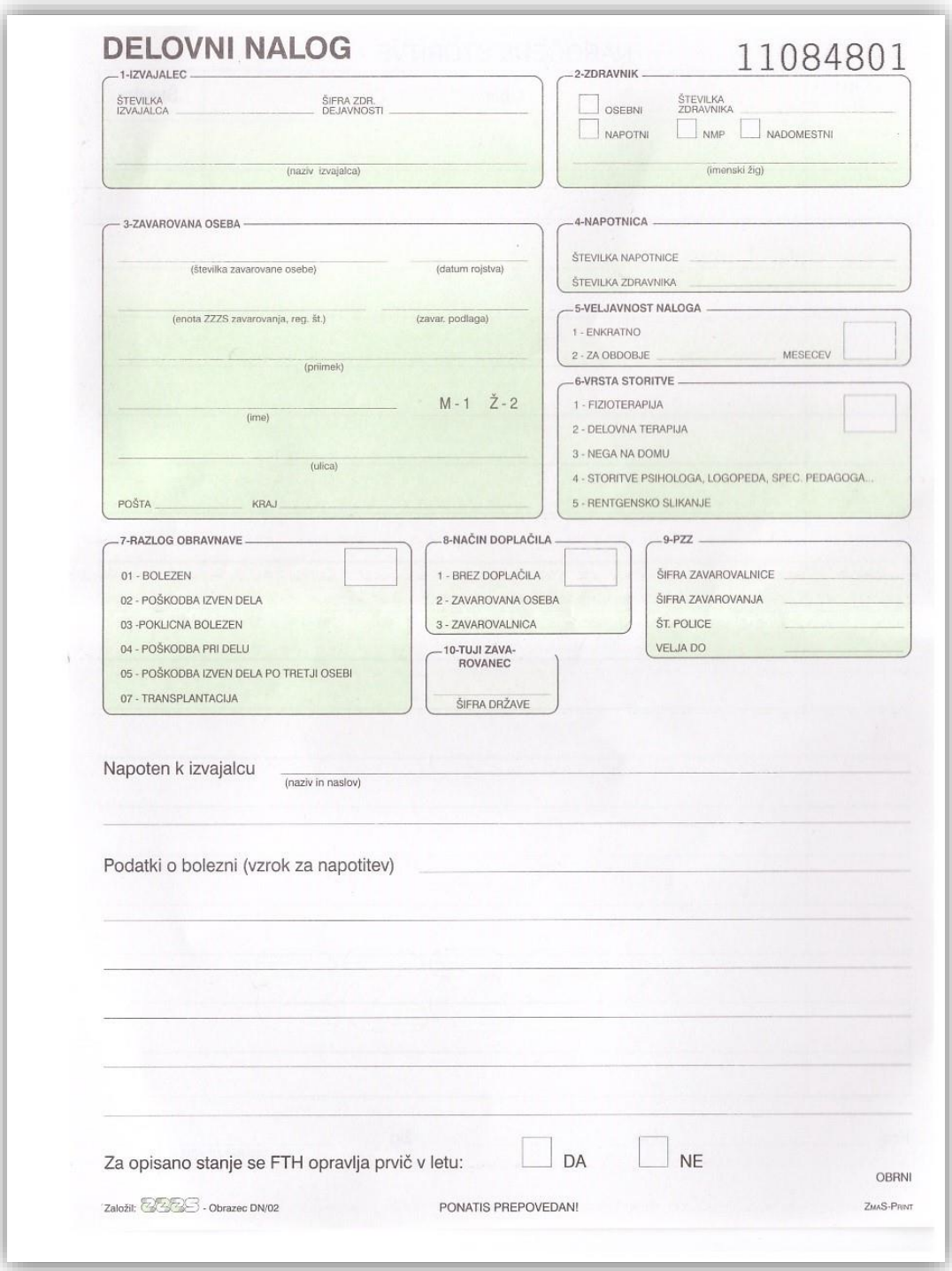


Slika 1 Shema umestitve patronažnega sistema v zdravstveni sistem

2.2 Podatki v sistemu patronažnega varstva

Osnova za delo patronažnih sester je delovni nalog (Slika 2, Slika 3), kateri vsebuje vse podatke o pacientu in zdravniku, ki ga je izdal. V dodatnih poljih je podan razlog za izvedbo nege in vse naročene storitve, ki jih bo sestra morala opraviti na obisku pri pacientu.

Poleg delovnega naloga se v obstoječih informacijskih sistemih beležijo še drugi podatki za razne statistične obdelave in poročanje zavarovalnicam, kot so: anamneza, oblika varstva, vrsta obiska, naročnik obiska, vrsta zdravstveno vzgojne dejavnosti, vrsta intervencije, mesto intervencije, faza urejanja, obisk družine, vrsta družine, vrsta vozila, oblika naselja, kategorija pacienta in družine, plačnik storitev, proračunski plačnik ali podjetje, seznam storitev in ostali. Teh podatkov zaradi preobsežnega vnosa preko mobilne naprave in zaradi tega, ker niso potrebni za delo na terenu nismo predvideli znotraj rešitve in posledično tudi niso vključeni v mobilno aplikacijo.



DELOVNI NALOG

11084801

1-IZVAJALEC

ŠTEVILKA IZVAJALCA _____ ŠIFRA ZDR. DEJAVNOSTI _____

(naziv izvajalca)

2-ZDRAVNIK

☐ OSEBNI ŠTEVILKA ZDRAVNIKA _____

☐ NAPOTNI ☐ NMP ☐ NADOMESTNI

(imenski žig)

3-ZAVAROVANA OSEBA

(številka zavarovane osebe) (datum rojstva)

(enota ZZZS zavarovanja, reg. št.) (zavar. podlaga)

(priimek)

(ime) M - 1 Ž - 2

(ulica)

POŠTA _____ KRAJ _____

4-NAPOTNICA

ŠTEVILKA NAPOTNICE _____

ŠTEVILKA ZDRAVNIKA _____

5-VELJAVNOST NALOGA

1 - ENKRATNO

2 - ZA OBDOBJE _____ MESECEV ☐

6-VRSTA STORITVE

1 - FIZIOTERAPIJA ☐

2 - DELOVNA TERAPIJA

3 - NEGA NA DOMU

4 - STORITVE PSIHOLOGA, LOGOPEDA, SPEC. PEDAGOGA...

5 - RENTGENSKO SLIKANJE

7-RAZLOG OBRAVNAVE

01 - BOLEZEN ☐

02 - POŠKODBA IZVEN DELA

03 -POKLICNA BOLEZEN

04 - POŠKODBA PRI DELU

05 - POŠKODBA IZVEN DELA PO TRETJI OSEBI

07 - TRANSPLANTACIJA

8-NAČIN DOPLAČILA

1 - BREZ DOPLAČILA ☐

2 - ZAVAROVANA OSEBA

3 - ZAVAROVALNICA

9-PZZ

ŠIFRA ZAVAROVALNICE _____

ŠIFRA ZAVAROVANJA _____

ŠT. POLICE _____

VELJA DO _____

10-TUJI ZAVAROVANEC

ŠIFRA DRŽAVE

Napoten k izvajalcu _____

(naziv in naslov)

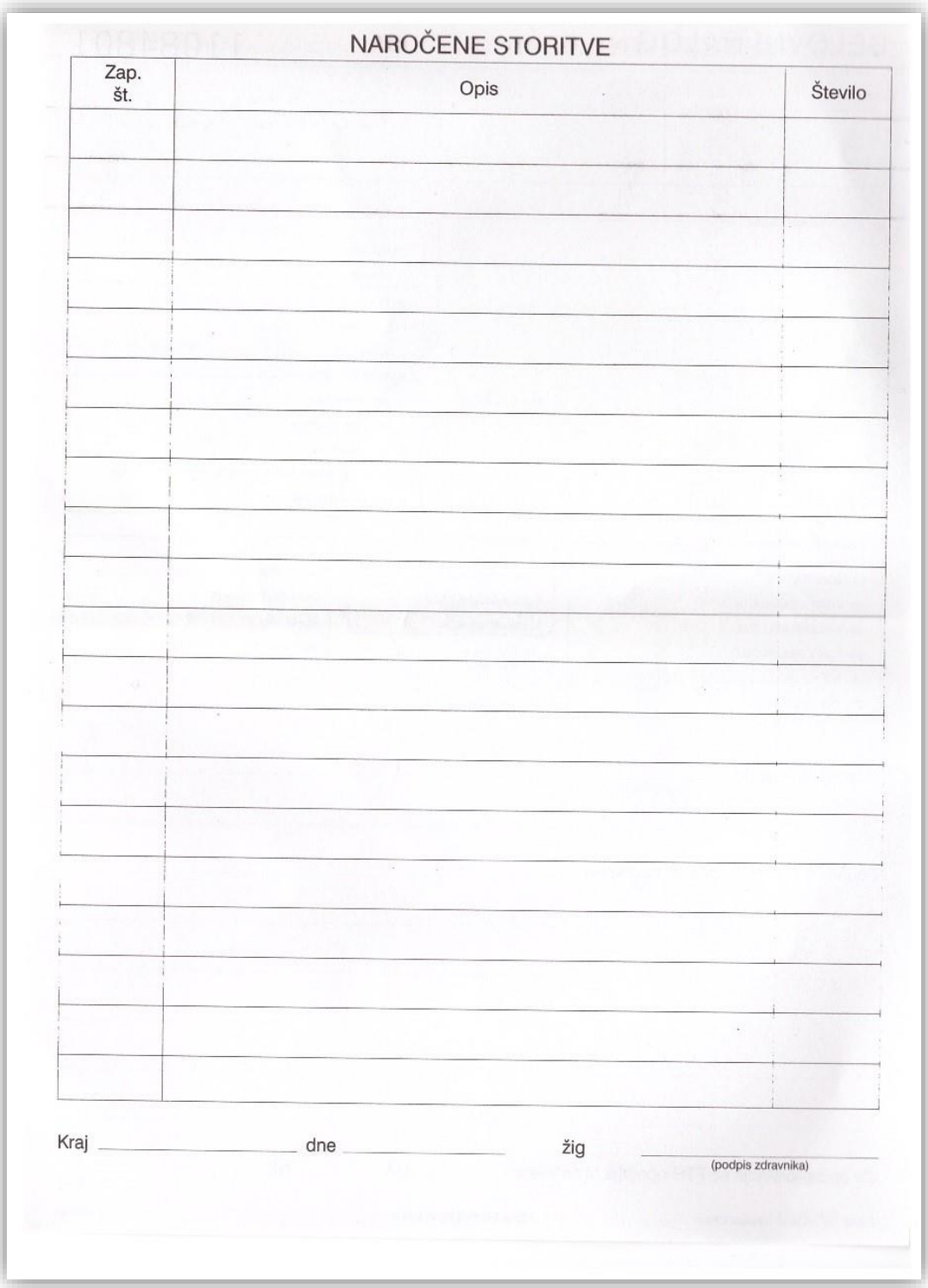
Podatki o bolezni (vzrok za napotitev) _____

Za opisano stanje se FTH opravlja prvič v letu: ☐ DA ☐ NE

OBRNI

Založil: ZZZS - Obrazec DN/02 PONATIS PREPOVEDAN! ZMAS-PRINT

Slika 2 Delovni nalog - sprednja stran



NAROČENE STORITVE

Zap. št.	Opis	Število

Kraj _____ dne _____ žig _____

(podpis zdravnika)

Slika 3 Delovni nalog - zadnja stran

2.3 Primeri uporabe in potrebne storitve

Po pregledu vseh možnih primerov uporabe in potrebnih vnosnih mask je bilo hitro očitno, da niso vsi ustrezni za vnos na terenu, predvsem zaradi podatkov. Mobilna aplikacija mora biti prilagojena za enostavno in pregledno uporabo na majhnemu zaslonu in mora vključevati minimalno količino podatkov zaradi pomanjkanja fizične tipkovnice. Osnova, ki se ji ne moremo izogniti, so podatki na delovnem nalogu, kar pa ni tako kritično, saj se na terenu večino podatkov samo pregleduje. Na podlagi tega smo uskladili naslednje možnosti, ki so se nam zdele najbolj koristne in uporabne za posredovanje na mobilno aplikacijo.

Pri snovanju strežniškega dela rešitve (funkcije, ki morajo biti na voljo) smo upoštevali naslednje uporabniške zahteve z vidika mobilne aplikacije:

- Inicializacija mobilne aplikacije:
    - Predpogoj je, da uporabnik še nima nameščene mobilne aplikacije.
    - Skrbniško orodje mora podpirati izdajo kode za določenega uporabnika.
    - Kodo je potrebno vnesti v mobilno aplikacijo ob namestitvi.
    - Mobilna aplikacija kliče strežnik in dobi nazaj avtorizacijski žeton za naknadne prijave.
- Prijava na strežnik (pri vsakdanji uporabi) - pri prijavi pošlje uporabniško ime in avtorizacijski žeton.
- Pregled delovnih nalogov - seznam nalogov za tekoči dan in podrobnosti delovnega naloga.
- Podrobnosti delovnega naloga:
    - Pridobi se jih lahko iz prejšnjega klica, ki bo vračal vse podrobnosti.
    - Na voljo je dodatna funkcija, ki lahko osveži samo podatke enega naloga v primeru sinhronizacijskih težav.
- Dodajanje storitev:
    - Na voljo je funkcija, s katero se lahko vnese dodatna storitev na nalog.
    - Na voljo je funkcija, ki vrne šifrant storitev za izbor v aplikaciji.
- Obdelava delovnega naloga – na voljo sta dve funkciji, ki omogočata zmanjšanje ali povečanje:
    - določenega števila storitev.
    - števila potrebnih obiskov.

Skrbniška aplikacija mora podpirati pregled in opravljanje nalog na podlagi zgornjih zahtev tako, da omogoča:

- pregled uporabnikov in izdajo mobilnih kod ter
- pregled in zapiranje ter ponovno odpiranje delovnih nalogov.

Programski vmesnik za integracijo v zdravstveni sistem mora omogočati prenos podatkov iz obstoječega informacijskega sistema:

- dodajanje, spreminjanje in brisanje uporabnikov mobilne aplikacije (npr. vse patronažne sestre),
- dodajanje, spreminjanje in brisanje delovnih nalogov v našem sistemu ter
- pridobivanje spremenjenih podatkov preko mobilne aplikacije od zadnje sinhronizacije podatkov.

3 Razvoj sistema za patronažno varstvo

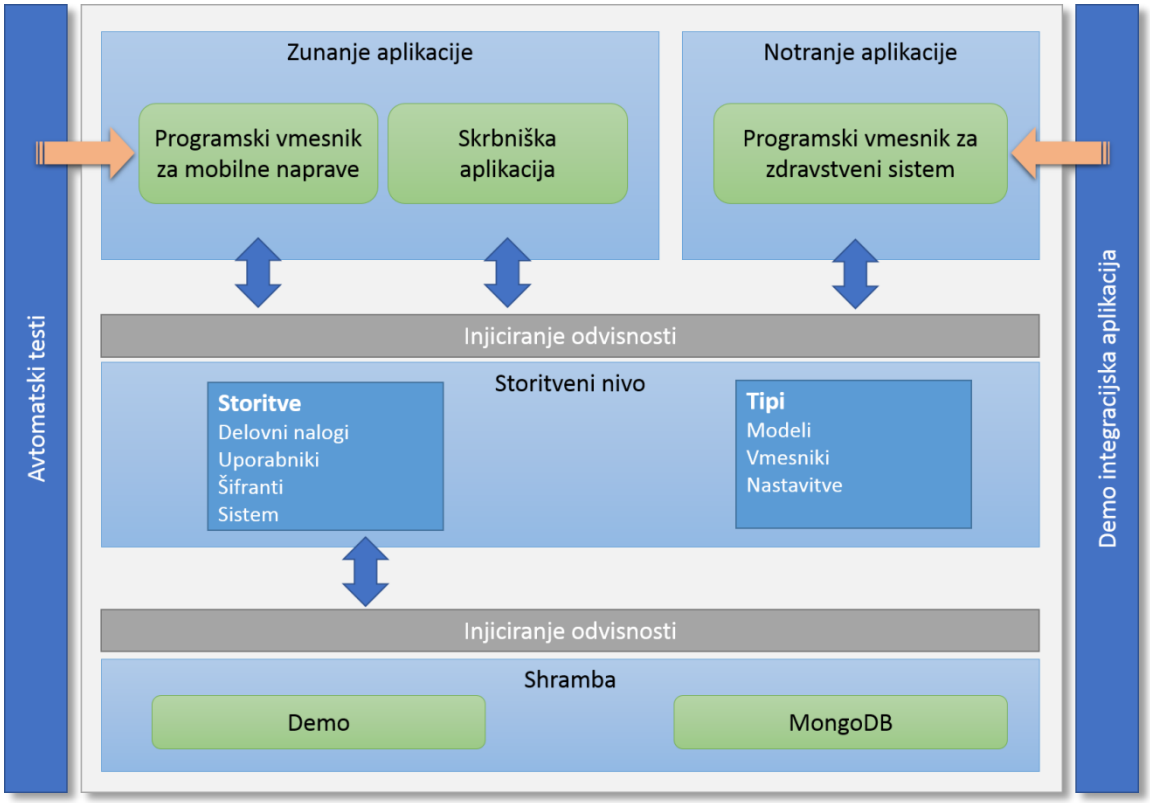
Po pridobitvi in analizi vseh zahtev smo pričeli z razvojem sistema. Potrebno je bilo definirati arhitekturo, jo razdeliti na glavne komponente, doreči in uskladiti tehnologije za integracijske vmesnike in izbrati orodja za razvoj.

3.1 Arhitektura sistema

Celoten strežniški sistem (Slika 4), ki smo ga vzpostavili za podporo mobilni aplikaciji, je sestavljen iz več nivojev, in sicer:

- **Zunanje aplikacije** - aplikaciji, ki sta odprti proti končnim uporabnikom:
  - Programski vmesnik za mobilne naprave: vsebuje vse funkcije, ki so potrebne za delovanje mobilne aplikacije.
  - Skrbniška aplikacija: vsebuje osnovne funkcije za pregled in upravljanje delovnih nalogov, pregled uporabnikov in izdajo mobilnih kod.
- **Notranje aplikacije** – aplikacija, ki je odprta proti internim zdravstvenim sistemom. Ker lahko zahteva višjo varnost in ločeno namestitev v varno interno okolje zdravstvene ustanove, je narejena kot ločena aplikacija:
  - Programski vmesnik za integracijo v zdravstveni sistem: vsebuje vse funkcije, ki so potrebne, da strežnik pridobi podatke o uporabnikih in delovnih nalogih za zagotavljanje podatkov mobilnim napravam.
- **Storitveni nivo** (angl. service layer) :
  - Storitve: zagotavlja vse storitve, ki so potrebne za delovanje končnih aplikacij, tako zunanjih kot tudi notranjih (npr. pregled delovnih nalogov, pregled uporabnikov, pridobivanje šifrantov itn.).
  - Tipi: vsebuje vse tipe, ki nastopajo v sistemu, kot so modeli (npr. uporabniki, delovni nalogi), vmesniki (npr. interni vmesniki do storitev in shrambe) in nastavitve, to je trenutna konfiguracija sistema.
- **Nivo shrambe** (angl. persistence layer), ki dolgoročno hrani stanje oz. podatke v sistemu. Implementirani sta dve rešitvi, vendar je sistem odprt za dodajanje kakršnekoli druge shrambe oz. implementacije druge podatkovne baze:
  - Demo: začasno hrani podatke in jih ob vsakem zagonu obnovi na začetno vnaprej definirano stanje. Uporablja se za zagon avtomatskih testov, saj se tako zaganjajo vedno nad istim stanjem sistema in s tem omogočajo celovito testiranje programskega vmesnika za mobilne naprave.

- MongoDB: vsebuje implementacijo za delo z NoSQL MongoDB bazo [4].



Slika 4 Visokonivojski pregled arhitekture sistema

Nastavitve trenutno zastavljenega delovanja so definirane tako, da sistem podpira naslednje nastavitve:

- **DatabaseRepository:** podpira možnosti »Demo« ali »MongoDB«, s čimer menjamo način hrambe podatkov.
- **DatabaseConnectionString:** če imamo nastavljeno vrednost »MongoDB«, vnesemo, kje je dostopna baza.
- **LoginType:** trenutno podpira samo vrednosti »AuthToken« ali »UsernamePassword«, v odvisnosti od tega, kako bi želeli, da se uporabnik avtenticira. Privzeto je trenutno nastavljen »AuthToken«, ker se s tem načinom avtenticira mobilna naprava.

Med aplikativnim, storitvenim nivojem in nivojem shrambe je tako imenovano injiciranje odvisnosti (angl. dependency injection), ki nam zagotavlja, da so nivoji med seboj šibko povezani (angl. loosely coupled) in lahko enostavno spremenimo implementacijo katerekoli storitve ali pa dodamo drugo implementacijo shrambe, če se držimo zahtevanih vmesnikov.

Poleg naštetih nivojev oziroma aplikacij sistem vsebuje še dve dodatni komponenti oz. aplikaciji, ki sta namenjeni predvsem testiranju oz. simulaciji delovanja celotnega sistema:

- **Avtomatski testi**: vsebujejo logiko za testiranje vmesnikov do mobilne naprave. Ker izdelava mobilne naprave ni sestavni del te diplome, smo z avtomatskimi testi lahko zagotovili neodvisno fazo testiranja tako, da vmesniki do mobilne naprave res delujejo. Poleg tega zagotavljajo t. i. regresijske teste, s katerimi zagotovimo, da ob spremembah v sistemu nismo pokvarili vmesnikov.
- **Demo-integracijska aplikacija**: ker skrbniška aplikacija ne omogoča vnosa uporabnikov, delovnih nalogov in upravljanja z njimi, smo razvili aplikacijo, ki deluje nad programskim vmesnikom za integracijo v zdravstveni sistem. S tem smo lahko potrdili delovanje vmesnika in simulirali delovanje sistema znotraj zdravstvene ustanove.

3.2 Varnost sistema

Varnosti smo se posvetili predvsem pri vmesniku do same mobilne naprave, saj so varnostni mehanizmi za različne ustanove različni in bi se tako prilagodili posamezni vključitvi določene zdravstvene ustanove. Zaradi tega je skrbniška aplikacija trenutno dostopna za vse uporabnike, prav tako je programski vmesnik za integracijo dostopen vsem aplikacijam in nima avtentikacijskega mehanizma.

Pri mobilni aplikaciji je predviden dostop z uporabniškim imenom in geslom. Za prenos uporabniškega imena in gesla do strežnika se uporablja standard »HTTP Basic Authentication« [3]. Omogočena je uporaba preko varnega transportnega kanala (SSL/TLS), s čimer zagotovimo transportno varnost. Uporabniško ime se pridobi ob vnosu uporabnika v sistem, za geslo pa se uporablja avtentikacijski žeton, ki ga pridobimo pri inicializaciji s pomočjo uporabnikove mobilne kode.

Ker so zdravstveni podatki kritični s stališča varnosti, smo želeli omejiti tudi mobilno napravo, s katere lahko uporabnik dostopa. Želeli smo, da uporabnik ne more enostavno namestiti aplikacije na drugo napravo in potem dostopati do podatkov s svojim uporabniškim imenom in geslom. Zaradi tega ima vsaka naprava izdano svojo mobilno kodo.

Mobilne kode

Postopek inicializacija aplikacije oz. izdaje mobilnih kod je sledeč:

- Za vsako napravo (vezano na uporabnika) se izda žeton za inicializacijo (mobilna koda) preko skrbniške aplikacije.

- Uporabnik si namesti mobilno aplikacijo. Ob prvi namestitvi mora vnesti svoje uporabniško ime in inicializacijski žeton.
- Naprava se poveže na strežnik in pridobi avtentikacijski žeton. S tem se je inicializacijski žeton pobrisal in ga ni mogoče več uporabiti.
- Avtentikacijski žeton mora aplikacija varno hraniti na telefonu (npr. kriptirati z neko krajšo uporabniško kodo), s čimer se prepreči enostavna kraja. Daljše kot je geslo, težje bo razbiti enkripcijo, vendar breme varnosti prenesemo na uporabnika, ki mora vnašati daljše geslo.
- Ob vsaki naslednji uporabi se aplikacija na strežnik prijavi z uporabniškim imenom in avtentikacijskim žetonom.

Za bolj prijazno uporabo svetujemo, da se avtentikacijski žeton hrani v seji določen čas (npr. do 10 minut brez uporabe aplikacije) in uporabniku ni potrebno vsakič znova vnašati njegovega gesla. Vendar je ta odločitev prepuščena izvedbi mobilne aplikacije.

V primeru uporabe varnega kanala (SSL/TLS) za povečanje transportne varnosti prav tako priporočamo, da mobilna naprava uporabi t. i. pripenjanje javnega ključa (angl. public key pinning) strežniškega digitalnega potrdila, saj s tem onemogoča določene napade na transport.

Pri sami avtentikaciji strežnik pridobi tudi avtorizacijske podatke (vloga uporabnika), ki pa v tej fazi niso razdelane in se ne preverjajo – potrebna bi bila bolj podrobna analiza zdravstvenih sistemov in različnih vlog znotraj njih. S trenutno zasnovo je bila urejena možnost za bodočo nadgradnjo z različnimi vlogami uporabnikov.

3.3 Sinhronizacija

Ker so v podatkovni tok vključeni trije neodvisni sistemi (mobilna naprava, strežnik, zdravstveni zaledni sistem), lahko pride do težav pri sinhronizaciji različnih podatkov, kot so:

- kateri podatki se pošiljajo kam in kdaj,
- kaj se zgodi, če ali ko sta dva podatka v različnih sistemih spremenjena istočasno.

Za reševanje teh težav je vpeljan sinhronizacijski žeton. Za sinhronizacijski žeton je uporabljen podatek »DateTime.Now.Ticks()« in predstavlja število 100-nanosekundnih intervalov od 1. januarja 001 [5]. Sinhronizacijski žeton se posodobi ob vsaki spremembi podatka v sistemu in ima dva namena:

- Ob spremembi na mobilni aplikaciji se istočasno s spremembo pošlje na strežnik tudi sinhronizacijski žeton. S tem lahko preverimo, da se podatek (uporabnik, delovni nalog itn.) ni spremenil od zadnje sinhronizacije. V primeru spremembe bo strežnik vrnil napako mobilni aplikaciji in potrebna bo ponovna sinhronizacija

problematičnega naloga s strežnika. Uporabnik mora sicer potem ponovno narediti želene spremembe, vendar se na ta način zagotovi, da ne pride do izgube nobenih spremenjenih podatkov.

- Služi tudi temu, da zdravstveni zaledni sistem lahko pridobi spremenjene podatke od zadnje sinhronizacije. Ob vsaki sinhronizaciji zaledni zdravstveni sistem pridobi trenutno veljavni sinhronizacijski žeton. S tem žetonom lahko pri naslednji sinhronizaciji pridobimo samo tiste podatke, ki so bili spremenjeni od prejšnje sinhronizacije.

3.4 Tehnologije in orodja

Ker je osnovni cilj strežniškega dela zagotavljanje podatkov za mobilno napravo preko spletnega vmesnika, je bil izbor tehnologije neodvisen od samega razvoja mobilne aplikacije. Poleg samih vsebinskih aplikativnih zahtev si želimo tudi vzpostaviti sistem, ki ga bo enostavno vzdrževati in v bodoče možno nagraditi. Zato so na izbor tehnologij in orodij vplivale naslednje zahteve:

- morajo podpirati razvoj spletne aplikacije,
- morajo podpirati razvoj programskega aplikacijskega vmesnika za mobilne naprave,
- želimo, da je aplikacija skalabilna,
- želimo, da je grajena na najnovejših tehnologijah in s tem pripravljena tudi za bodočo rast,
- tehnologija mora podpirati avtomatske teste.

Za osnovno tehnologijo razvoja smo izbrali platformo Microsoft Windows in njihovo razvojno okolje .NET Framework. Po zadnjih podatkih je približno 33 odstotkov spletnih aplikacij, ki temeljijo na platformi Microsoft Windows [6].

Za razvojno okolje smo uporabljali operacijski sistem Windows 7 in brezplačno izvedbo razvojnega okolja Microsoft Visual Studio Express 2012. Izbrana verzija platforme .NET Framework je 4.5, ker je privzeta za razvoj na Visual Studio 2012. Pri razvoju smo izmed jezikov .NET, ki so na voljo, vedno izbrali C#.

Ker smo želeli slediti dobrim načelom razvoja, smo uporabljali tudi sistem za nadzor različic (angl. version control). Izbrali smo si brezplačno orodje Git [7], ki spada v družino distribuiranih orodij za nadzor različic.

4 Izvedba in delovanje sistema

Izmenjava podatkov v sistemu patronažnega varstva poteka med različnimi sistemi (Slika 5):

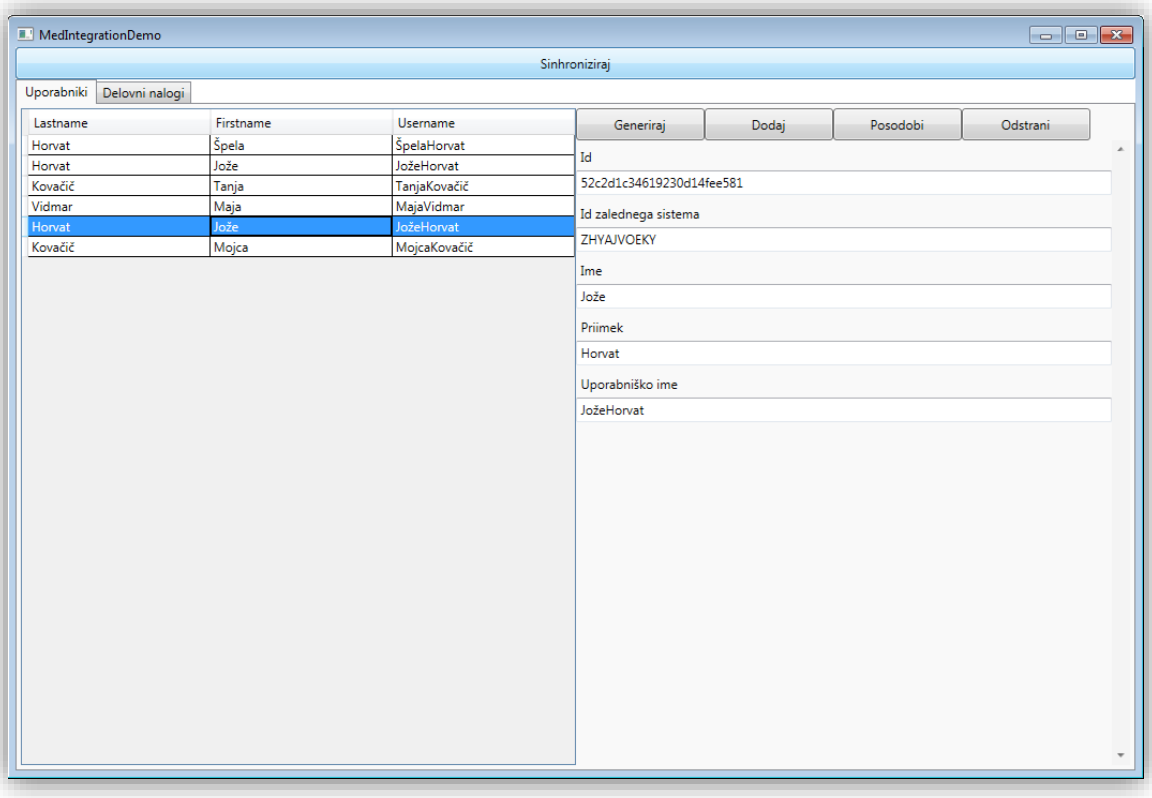
1. V prvi fazi zdravstveni zaledni sistem pošlje podatke o uporabnikih (enkratna operacija) in podatke o delovnih nalogih (periodično, npr. dnevno).
2. Sistem za patronažno varstvo hrani podatke in jih zagotavlja mobilni aplikaciji, ki lahko v svojo bazo prenese podatke za določeni dan.
3. Po zaključenem delu (npr. zaključenem obisku pri pacientu) se posodobljeni podatki z mobilne aplikacije pretočijo nazaj v sistem za patronažno varstvo.
4. Zdravstveni zaledni sistem (zopet ob določenih periodah) izvaja poizvedbe v sistem za patronažno varstvo za spremembe delovnih nalogov. Če spremembe obstajajo, jih pridobi in lahko posodobi v svojem sistemu. V našem primeru smo zaledni sistem simulirali z demo aplikacijo.

Slika 5 Tok podatkov v sistemu

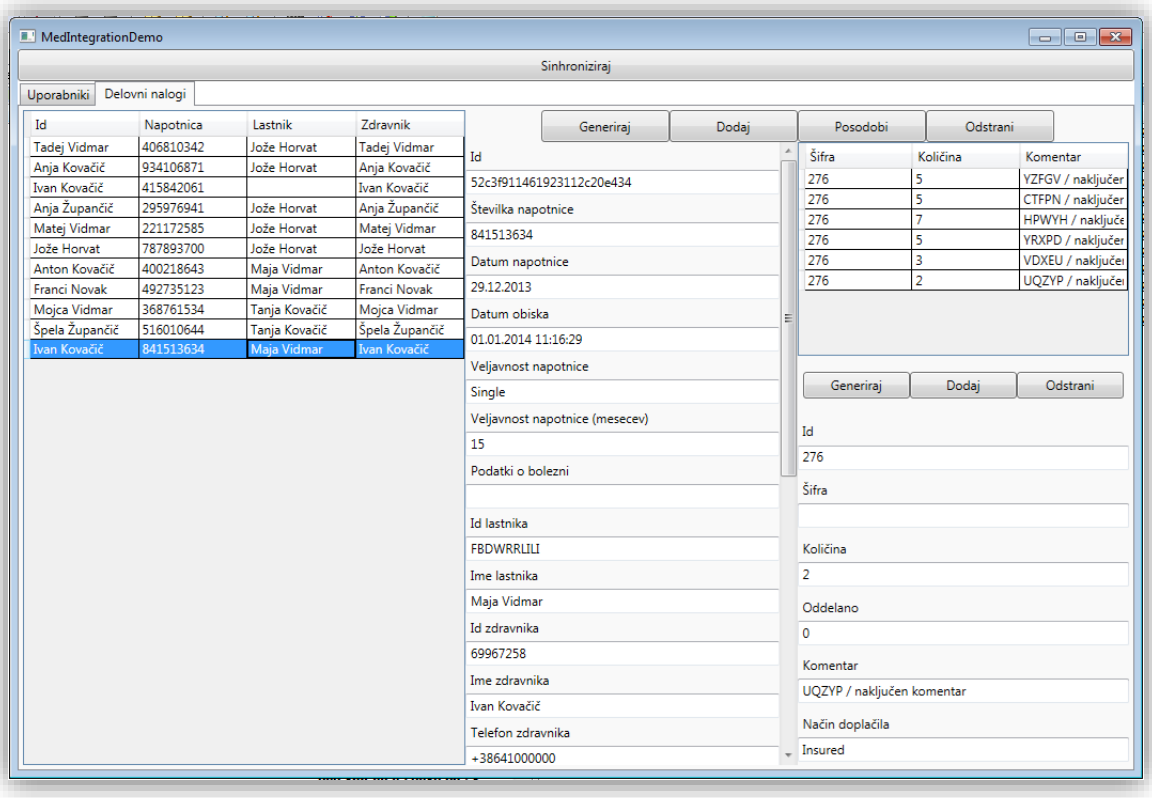
4.1 Demo aplikacija

Demo aplikacija služi simulaciji integracije naše rešitve v okolje zdravstvene ustanove (simulacija zdravstvenega sistema). Z njo lahko v sistem pošiljamo in spreminjamo podatke o uporabnikih in delovnih nalogih. V osnovi se loči na dva razdelka, in sicer za delo nad uporabniki (Slika 6) in za delo nad delovnimi nalogi (Slika 7). Pri vsakem so funkcionalnosti večinoma enake, razlika je le v podatkih, ki so upoštevani.

Uporabljena je tehnologija .NET WPF (Windows Presentation Foundation) [8], ki se uporablja za programiranje bogatih pametnih odjemalčevih aplikacij. Izbrana je bila predvsem zato, ker omogoča hiter razvoj enostavne aplikacije za operacijski sistem Windows.



Slika 6 Upravljanje z uporabniki v demo aplikaciji



Slika 7 Upravljanje z delovnimi nalogi v demo aplikaciji

Z glavnim gumbom »Sinhroniziraj« se aplikacija poveže na strežnik preko programskega vmesnika SOAP in pridobi podatke iz sistema. V ta namen pošlje vrednost sinhronizacijskega žetona 0 in na ta način pridobi vse podatke in ne samo spremembe.

Po pridobitvi podatkov s strežnika napolni sezname oziroma tabele na levi strani aplikacije. Z dvojnim klikom na katerokoli vrstico se izpolnijo vsi podatki o izbranem uporabniku ali delovnem nalogu na desni strani aplikacije.

Generiraj

Z gumbom »Generiraj« lahko sami definiramo testne podatke:

- naključna imena oseb iz vnaprej definiranega seznama slovenskih imen in priimkov,
- naključna števila za numerična polja,
- vsaj približno pravilne naključne besede za večino tekstovnih polj (na primer za polje »Naslov« vedno generira prvo besedo Ulica in potem naključni niz nekaj znakov),

Nekatera polja imajo fiksne vrednosti, na primer za mesto se vedno vnese »Ljubljana«. Za polja, ki imajo definiran nabor vrednosti iz nekega šifranta, izbere iz njega naključno vrednost. Vse podatke je možno ročno spreminjati. Funkcionalnost služi samo kot pomoč hitrejšemu dodajanju testnih podatkov.

Dodaj

Gumb »Dodaj« omogoča dodajanje novega uporabnika, delovnega naloga ali storitev na delovnem nalogu v sistem.

Posodobi

Gumb »Posodobi« poskrbi za posodobitev izbranega uporabnika ali delovnega naloga s trenutno vnesenimi podatki.

Odstrani

Gumb »Odstrani« omogoča odstranitev izbranega uporabnika, delovnega naloga ali storitev na delovnem nalogu iz sistema in zažene ponovno sinhronizacijo samo za tiste objekte, ki smo jih spreminjali (uporabniki ali delovnimi nalogi).

4.2 Programski vmesnik za integracijo v zdravstveni sistem

Programski vmesnik zaženemo v zagonski vrstici operacijskega sistema z ukazom »MedIntegrationServiceRunner.exe«. Uporabljeni tehnologiji sta NET WCF (Windows Communication Foundation) [9] in arhitekturni stil za spletne storitve SOAP [10]. Pri integraciji v zdravstveni sistem smo za razliko od programskega vmesnika za mobilne naprave (programski vmesnik REST) izbrali SOAP, in sicer zato, ker je precej bolj standarden za uporabo v velikih podjetjih, predvsem zaradi daljše zgodovine uporabe, preverjenega delovanja, široke podpore tudi v starejših razvojnih programih in enostavnejše razširitve z naprednejšimi funkcionalnostmi [11].

Za lažjo integracijo je na voljo datoteka WSDL s specifikacijo celotnega vmesnika. Na voljo so naslednje funkcije:

GetModifiedWorkOrders

Metoda sprejme sinhronizacijski žeton kot vhod in vrne seznam spremenjenih delovnih nalogov od vrednosti sinhronizacijskega žetona naprej. Če na vhodu podamo vrednost 0, vrne vse delovne naloge v sistemu. Več podrobnosti o sinhronizaciji in sinhronizacijskem žetonu je opisano v Poglavju 3.3.

Poleg spremenjenih delovnih nalogov vrne tudi zadnjo vrednosti sinhronizacijskega žetona s katerim lahko poizvedujemo ob naslednjem klicu in tako dobimo zopet samo spremenjene naloge.

AddWorkOrders

Metoda sprejme seznam objektov tipa delovni nalog in jih doda v sistem. Kot izhod vrne isti seznam, vendar imajo objekti dodan sinhronizacijski žeton in interne identifikatorje sistema.

UpdateWorkOrder

Metoda sprejme objekt tipa delovni nalog in ga znotraj sistema posodobi z vsemi podatki podanega delovnega naloga.

RemoveWorkOrder

Metoda sprejme objekt tipa delovni nalog in ga znotraj sistema odstrani.

GetAllUsers

Metoda vrne seznam objektov vseh trenutno vnešenih uporabnikov v sistemu.

AddUser

Metoda sprejme objekt tipa uporabnik in uporabnika doda v sistem. Kot rezultat vrne objekt istega tipa, vendar z dodanim internim identifikatorjem sistema.

UpdateUser

Metoda sprejme objekt tipa uporabnik in ga znotraj sistema posodobi z vsemi podatki podanega objekta.

RemoveUser

Metoda sprejme objekt tipa uporabnik in ga znotraj sistema odstrani.

CodeListMedicalServices

Metoda služi posodobitvi šifranta zdravstvenih storitev. Sprejme seznam objektov tipa zdravstvenih storitev in posodobi interno bazo tako, da vse predhodne vrednosti pobriše in doda poslane.

4.3 Skrbniška aplikacija

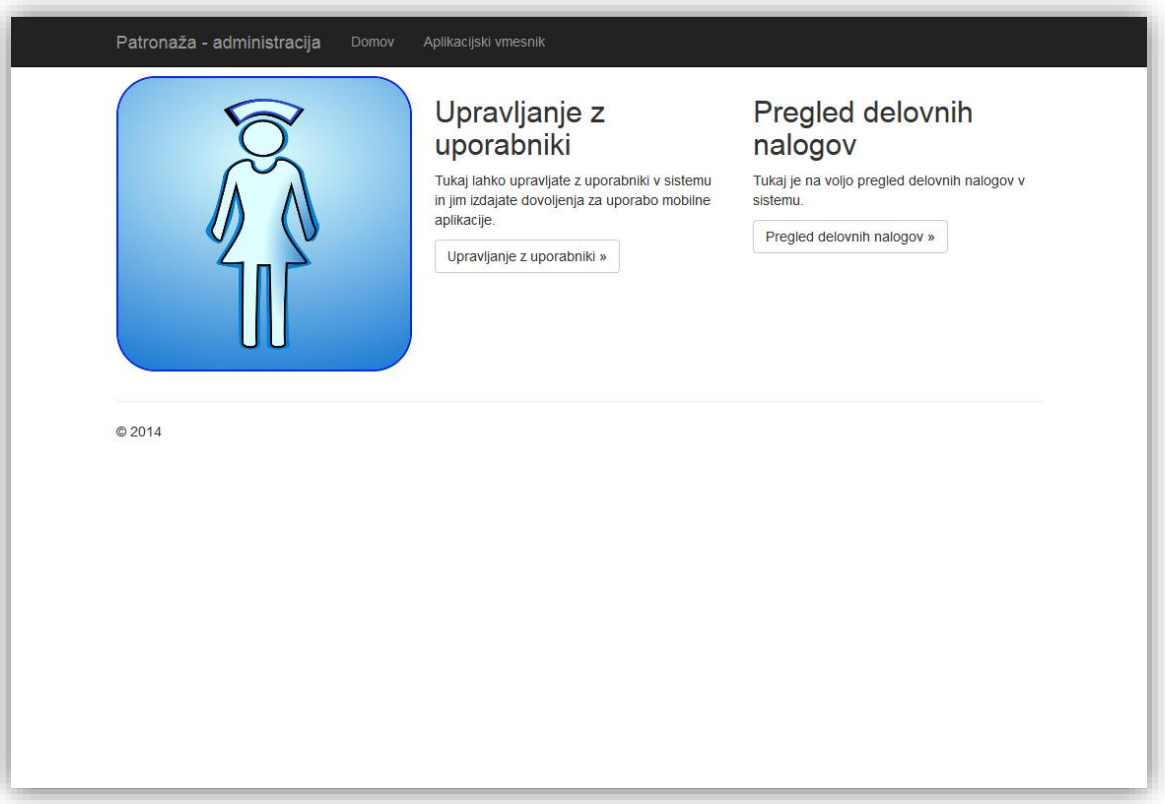
Skrbniška aplikacija je izdelana kot spletna aplikacija. Uporabljena je tehnologija .NET MVC [12] ki je trenutno uveljavljena kot najbolj standardni način razvoja spletnih aplikacij na platformi .NET, saj omogoča enostavno ločitev programske logike, modela podatkov in izgleda aplikacije po principu MVC [13]. Za HTML oz. CSS je uporabljen Twitter Bootstrap 3.0 [14].

Aplikacija omogoča upravljanje z uporabniki in pregled delovnih nalogov. Njene funkcionalnosti so naslednje:

Vstopna stran

Vstopna stran služi predstavitvi in vsebuje štiri glavne povezave (Slika 8):

- Domov je na voljo na vseh straneh in nas vrne na vstopno stran.
- Aplikacijski vmesnik je na voljo na vseh straneh in nas preusmeri na stran s tehničnimi navodili za integracijo z mobilno aplikacijo.
- Upravljanje z uporabniki nas preusmeri na razdelek za upravljanje z uporabniki v sistemu (patronažne sestre).
- Pregled delovnih nalogov nas preusmeri na razdelek za pregledovanje delovnih nalogov znotraj sistema. Trenutno omogoča pregled vseh podrobnosti nalogov in določene akcije nad njimi.

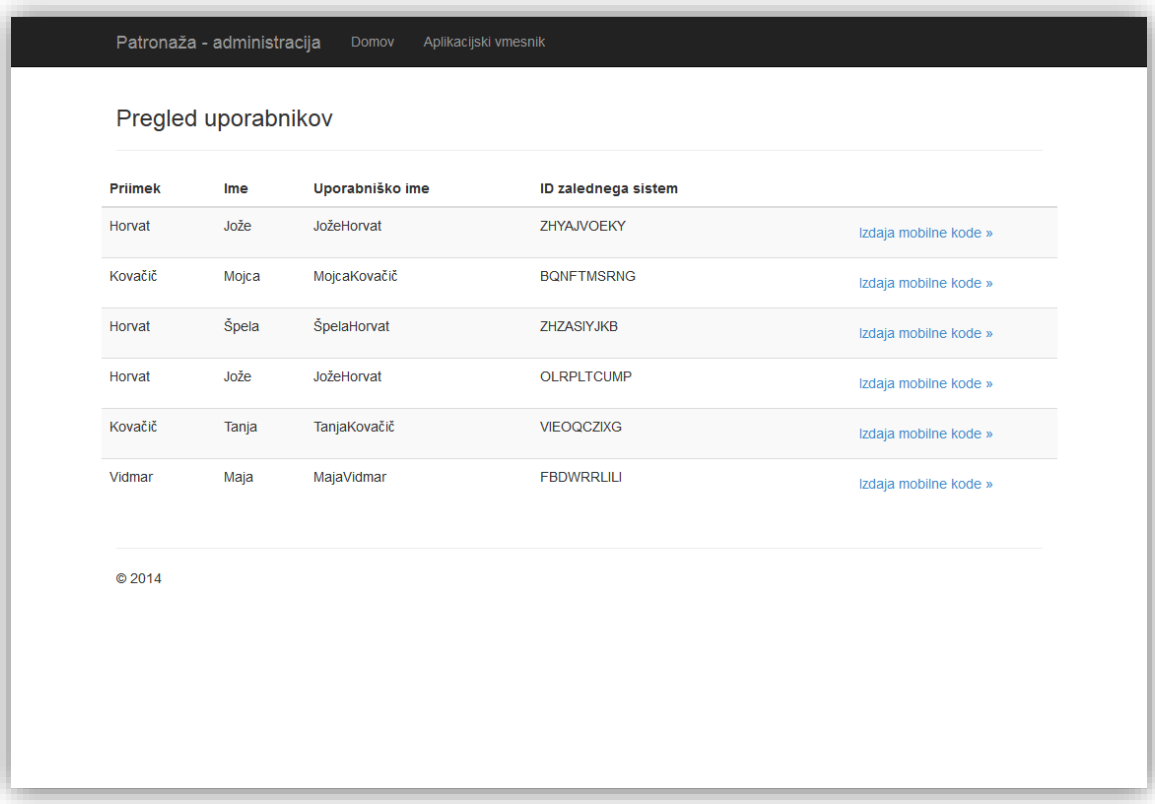


Slika 8 Vstopna stran skrbniške aplikacije

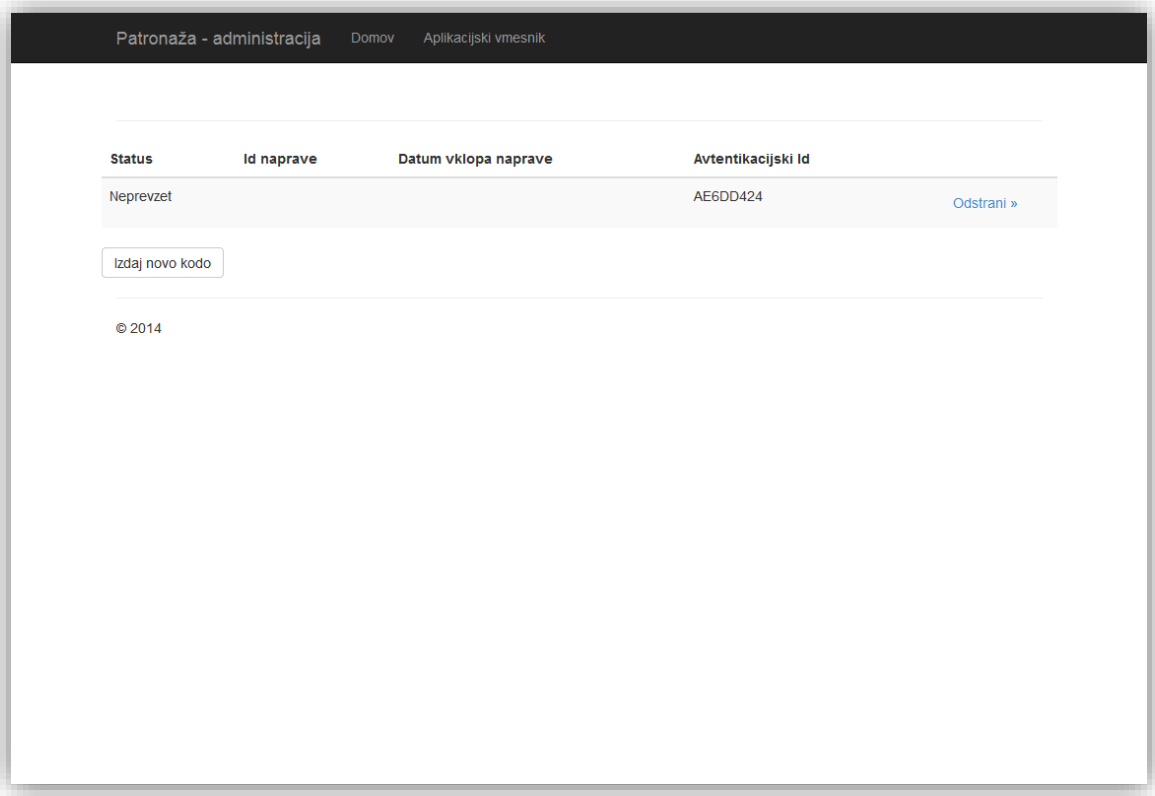
Upravljanje z uporabniki

Na strani pregled uporabnikov je na voljo seznam vseh uporabnikov, ki so bili sinhronizirani iz zalednega zdravstvenega sistema (Slika 9). Če je na voljo več kot 10 uporabnikov se prikaže seznam na več straneh s povezavami na posamezno stran. Z gumbom »Izdaja mobilne kode« lahko za določenega uporabnika upravljamo z mobilnimi kodami (za več o mobilnih kodah glej poglavje 3.2).

Mobilne kode lahko izdamo ali odstranimo (Slika 10). Vidi se tudi status, ali so že bile prevzete (oz. inicializirane za neko mobilno napravo) ali ne. Če kodo odstranimo, blokiramo dostop do aplikacije za izbrani telefon.



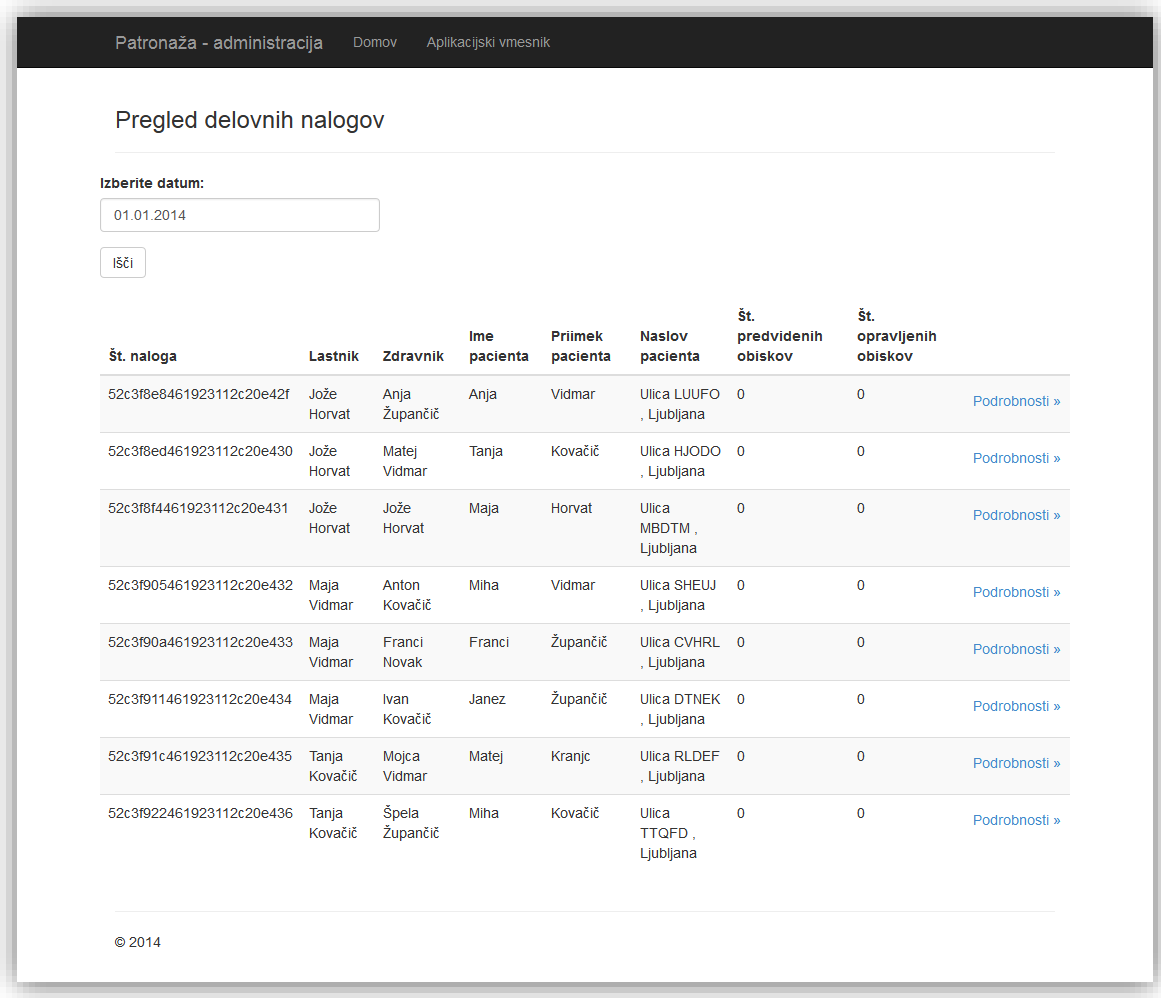
Slika 9 Upravljanje z uporabniki v skrbniški aplikaciji



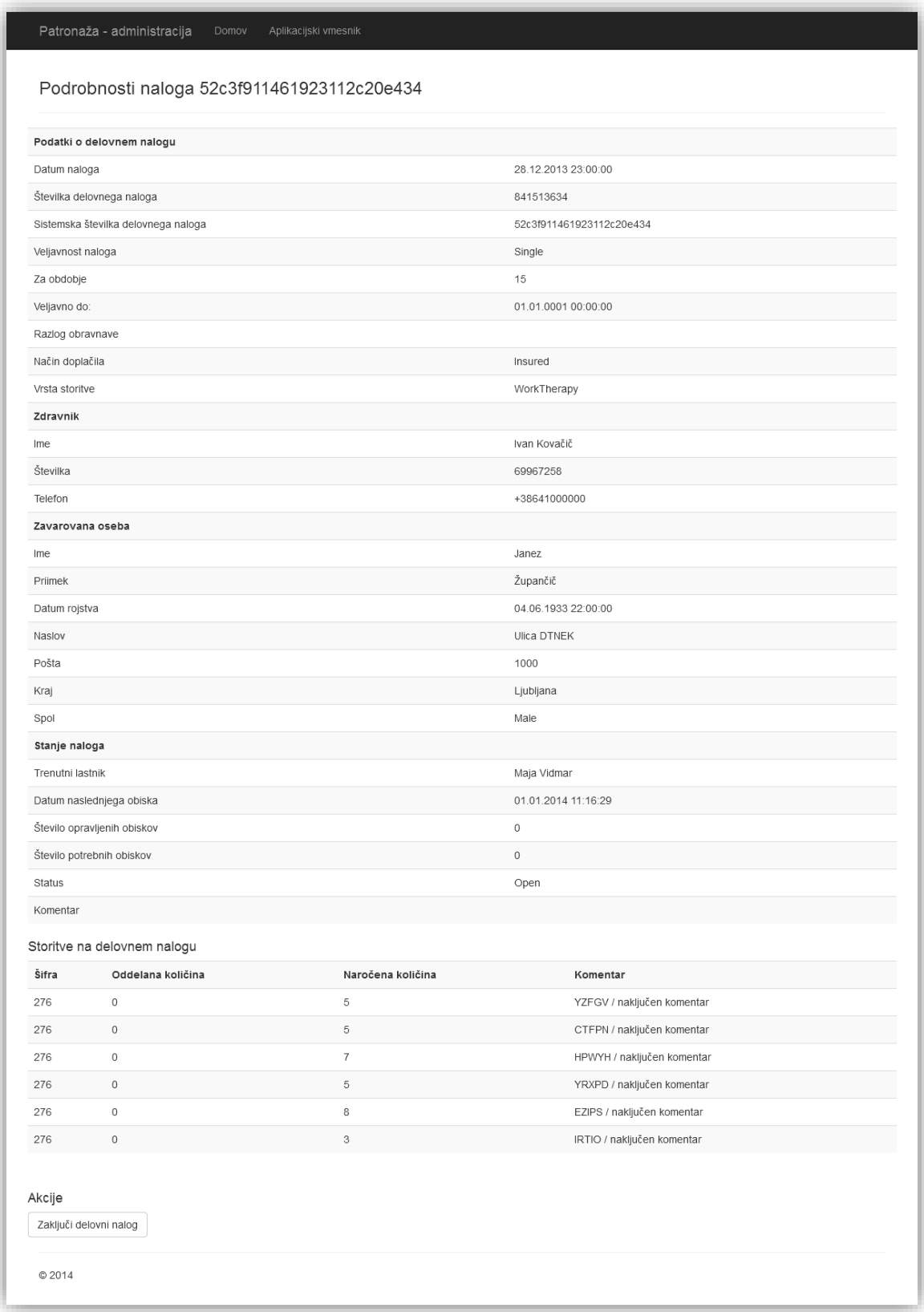
Slika 10 Pregled in izdajanje mobilnih kod

Pregled delovnih nalogov

V oknu Pregled delovnih nalogov (Slika 11) je na voljo seznam vseh odprtih nalogov, ki jih mora patronažna služba opraviti za izbrani datum. S povezavo na podrobnosti lahko pregledamo vse podatke (Slika 12) za tiste naloge, ki so shranjeni v sistemu. Nad nalogom lahko izvedemo dve operaciji, in sicer zaključevanje ter ponovno odpiranje določenega naloga. Če je nalog zaprt, se ne bo več pošiljal na mobilno aplikacijo, še vedno pa je na voljo v pregledu v skrbniški aplikaciji.



Slika 11 Pregled delovnih nalogov v skrbniški aplikaciji

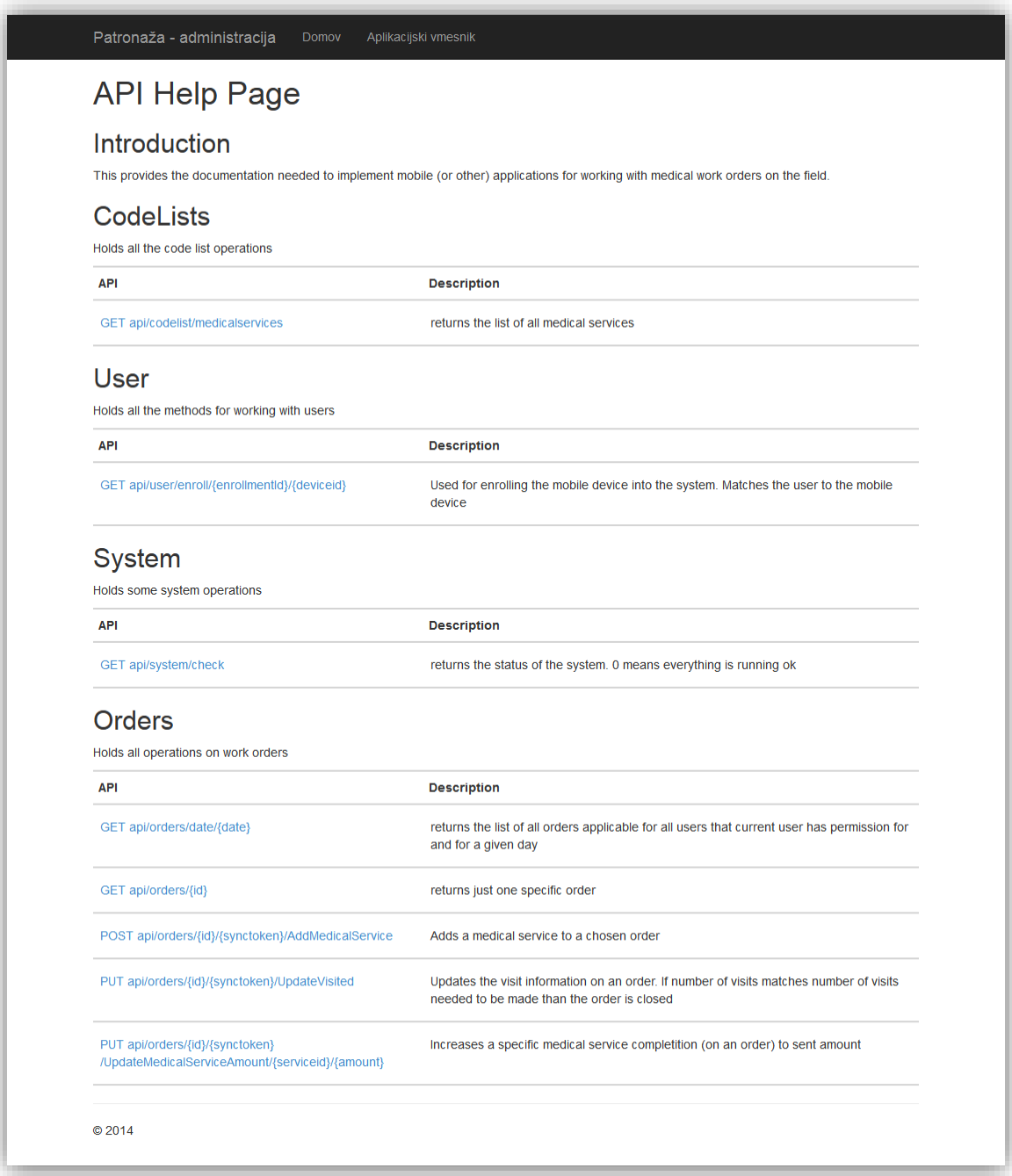


Slika 12 Pregled podrobnosti posameznega delovnega naloga

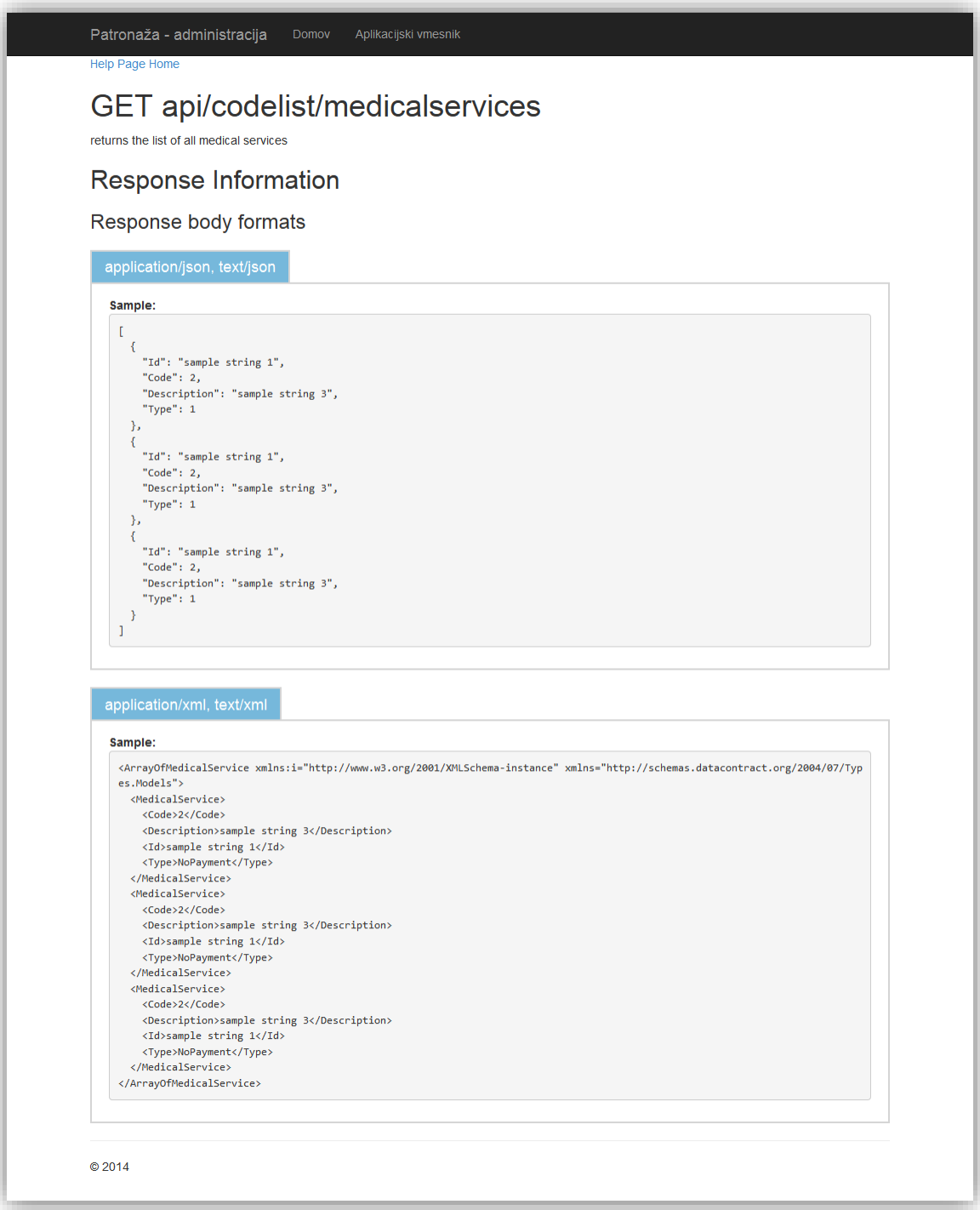
4.4 Programski vmesnik za mobilne naprave

Programski vmesnik za mobilne naprave je izdelan z uporabo arhitekture RESTful [15], ki je trenutno še neuradno uveljavljena kot standard za razvoj programskih vmesnikov v svetu mobilnih naprav, ker je enostaven za razumevanje, poteka preko protokola HTTP in ima široko podporo v vseh razvojnih okoljih mobilnih aplikacij. Za razvoj smo uporabili .NET Web API [16], ker je trenutno Microsoftova prednostna tehnologija za pisanje programskih vmesnikov REST.

V skrbniški aplikaciji je na voljo stran z navodili za programski vmesnik (Slika 13). Navedene so vse funkcije, ki jih lahko mobilna naprava kliče, prav tako je za vsako funkcijo na voljo stran s podrobnostmi klica in rezultati posameznega klica (Slika 14).



Slika 13 Stran s specifikacijo integracijskih vmesnikov za mobilno aplikacijo



Slika 14 Podrobnost posameznega klica integracijske metode

Podroben opis vseh metod je na voljo znotraj same aplikacije, tukaj pa bomo vključili samo kratek pregled. Glede na vhodni klic je mogoče menjati tip parametrov med formatoma XML ali JSON. Sistem bo v enakem, to je v izbranem formatu vrnil rezultat.

Metode se delijo v štiri sklope:

- operacije nad šifranti,
- operacije nad uporabniki,
- operacije nad sistemom in
- operacije nad naročili.

Klici so podani v sintaksi, ki je tudi nekako neuradno uveljavljen standard za REST api klice: TIP URI, kjer je TIP enak tipu http klica (GET, POST, PUT, itn.), URI je enak celotni poti metode vključno s parametri, ki so podani z zavitimi oklepaji {}.

1. Operacije nad šifranti

GET api/codelist/medicalservices

Metoda vrne šifrant vseh zdravstvenih storitev.

2. Operacije nad uporabniki

GET api/user/enroll/{enrollmentId}/{deviceid}

Metodo se uporabi za vključevanje naprave v sistem in pridobivanje avtorizacijskega žetona. Kot vhod sprejme identifikator za vklop naprave (parameter »enrollmentId«) in opis naprave, ki je prosti opis tipa naprave in služi samo informativnemu vpogledu znotraj skrbniške aplikacije.

3. Operacije nad sistemom

GET api/system/check

Metoda se uporablja samo v primeru težav s komunikacijo in razhroščevanju mobilne aplikacije ali strežnika. Ob klicu vrne stanje strežniškega sistema kot naslednje vrednosti:

- 0 – sistem deluje.
- 1 – sistemska napaka na strežniku - ni dostopa do nivoja storitev.
- 2 – sistemska napaka na strežniku – ni dostopa do nivoja hrambe.
- 3 – baza ni dostopna.

4. Operacije nad delovnimi nalogi

GET api/orders/date/{date}

Metoda sprejme datum, za katerega vrne vse delovne naloge.

GET api/orders/{id}

Metoda vrne delovni nalog za podani identifikator. Uporabna je predvsem v primeru, ko pride do napake pri sinhronizaciji naloga, da lahko pridobimo samo problematičen delovni nalog in ni potrebna sinhronizacija vseh delovnih nalogov.

POST api/orders/{id}/{synctoken}/AddMedicalService

Glede na poslani id delovnega naloga doda nanj novo storitev, ki je poslana kot POST parameter. Podatek »synctoken« je sinhronizacijski žeton, ki služi že omenjenemu odpravljanju težav sinhronizacije med različnimi deli sistema. V primeru, da ima mobilna aplikacija staro vrednost, bo strežnik poslal napako in zahteval ponovno sinhronizacijo.

PUT api/orders/{id}/{synctoken}/UpdateVisited

Metoda posodobi obiske na delovnem nalogu s podanim identifikatorjem. Sinhronizacijski žeton služi istemu namenom, kot v prejšnji metodi.

PUT api/orders/{id}/{synctoken}/UpdateMedicalServiceAmount/{serviceid}/{amount}

Metoda posodobi količino na zdravstveni storitvi na izbranem delovnem nalogu s podano vrednostjo (»amount«).

5 Sklepne ugotovitve

Sistem za podporo patronažne nege je zasnovan dovolj široko in do te mere, da je zrel tudi za dodelavo v storitev, ki bi lahko delovala v realnem okolju znotraj zdravstvenega sistema. Tehnično je zasnovan dovolj enostavno, da ni potrebno veliko dodatnega učenja za nadaljevanje razvoja, in vseeno dovolj kompleksno, da ga je mogoče razširiti z dodatnimi funkcionalnostmi. Prav tako je izveden dovolj neodvisno, da bi ga verjetno bilo mogoče umestiti v katerokoli zdravstveno okolje, seveda s potrebnimi dodelavami in morebitnimi prilagoditvami, kot je na primer avtentikacija uporabnikov skrbniške aplikacije in mogoče tudi avtentikacija na nivoju integracije zalednega sistema. Možno bi bilo vključiti tudi različne nivoje uporabniških vlog za skrbniško aplikacijo, prav tako pa bi se na iste vloge lahko nadgradilo programski vmesnik za mobilno aplikacijo.

Posebnih tehničnih in drugih težav s samim razvojem ni bilo, predvsem zaradi tega, ker imam zaradi številnih izkušenj na delovnem mestu razvijalca programskih in informacijskih rešitev veliko izkušenj z razvojem spletnih aplikacij. Največji izziv je predstavljalo razumevanje problematike patronažnega dela, trenutnega načina dela patronažnih sester in izbora funkcionalnosti, ki jih je smiselno podpreti znotraj mobilne aplikacije. Pri tem je bilo potrebno tudi veliko sodelovanja za razvoj primerov uporabe in razvoja programskih vmesnikov z Erminom Kentrićem pri njegovem razvoju mobilne aplikacije [2].

6 Literatura

[1] Uradni list, »Navodilo za izvajanje preventivnega zdravstvenega varstva na primarni ravni«, 1998, pogl. 1.1.3. Dostopno na: http://www.uradni-list.si/1/content?id=7259

[2] Ermin Kentrić, »The use of smart devices for data processing in the home care service«, 2013, Diplomsko delo. Dostopno na: http://eprints.fri.uni-lj.si/2315/

[3] Internet Engineering Task Force (IETF), Network Working Group, »HTTP Authentication: Basic and Digest Access Authentication«, 1999. Dostopno na: http://www.ietf.org/rfc/rfc2617.txt

[4] »MongoDB«, 2014. Dostopno na: http://www.mongodb.org/

[5] Microsoft Developer Network, »DateTime.Ticks Property“, 2014, Dostopno na: http://msdn.microsoft.com/en-us/library/system.datetime.ticks(v=vs.110).aspx

[6] W3Techs, »Usage of operating systems for websites«, 2014. Dostopno na: http://w3techs.com/technologies/overview/operating_system/all

[7] »Git«, 2014, http://git-scm.com/

[8] Microsoft Developer Network, »Windows Presentation Foundation«, 2014. Dostopno na: http://msdn.microsoft.com/en-us/library/ms754130(v=vs.110).aspx

[9] Microsoft Developer Network, »Windows Communication Foundation« 2014. Dostopno na: http://msdn.microsoft.com/en-us/library/dd456779(v=vs.110).aspx

[10] W3C, »SOAP Version 1.2«, 2007. Dostopno na: http://www.w3.org/TR/soap/

[11] Pautasso, C., Zimmermann, O., Leymann, F., »RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision«, 2008. Dostopno na: http://www.jopera.org/docs/publications/2008/restws

[12] Microsoft, »ASP.NET MVC«, 2014. Dostopno na: http://www.asp.net/mvc

[13] Microsoft Developer Network, »Model-View-Controller«, 2014. Dostopno na: http://msdn.microsoft.com/en-us/library/ff649643.aspx

[14] »Bootstrap«, 2014. Dostopno na: http://getbootstrap.com/

[15] Fielding, Roy Thomas, »Architectural Styles and the Design of Network-based Software Architectures«, 2000. Dostopno na: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[16] Microsoft, »ASP.NET Web API«, 2014. Dostopno na: http://www.asp.net/web-api