**SVEUČILIŠTE U ZAGREBU**

**FAKULTET ORGANIZACIJE I INFORMATIKE**

**V A R A Ž D I N**

Antonio Brkić, 46322/17-R

Andrea Danzante, 46341/17-R

Domagoj Ergović, 46340/17-R

Marijan Hranj, 46330/17-I

**AIR1715 - PillCare – Tehnička dokumentacija**

**PROJEKTNI RAD IZ KOLEGIJA ANALIZA I RAZVOJ PROGRAMA**

Varaždin, 2017.

**SVEUČILIŠTE U ZAGREBU**

**FAKULTET ORGANIZACIJE I INFORMATIKE**

**V A R A Ž D I N**

Antonio Brkić

Matični broj: 46322/17-R

Studij: Organizacija poslovnih sustava

Andrea Danzante

Matični broj: 46341/17-R

Domagoj Ergović

Matični broj: 46340/17.R

Marijan Hranj

Matični broj: 46330/17-I

Studij: Informacijsko i Programsko inženjerstvo

**AIR1715 - PillCare – Brkić, Danzante, Ergović, Hranj**

**GitHub:** https://github.com/AndDanza/PillCare.git

**Mentor:**

Dr. Sc. Zlatko Stapić

Varaždin, listopad 2017.

**Sadržaj**

[1. Uvod 1](#_Toc500602381)

[2. PillCare 1](#_Toc500602382)

[2.1. Svrha aplikacije 1](#_Toc500602383)

[2.2. Pojmovi korišteni u projektu 1](#_Toc500602384)

[2.3. Reference 2](#_Toc500602385)

[2.4. Specifikacija funkcionalnih zahtjeva 2](#_Toc500602386)

[2.5. O proizvodu 3](#_Toc500602387)

[2.5.1. Perspektiva proizvoda 3](#_Toc500602388)

[2.5.2. Funkcija proizvoda 3](#_Toc500602389)

[2.5.3. Korištene tehnologije 3](#_Toc500602390)

[2.6. Ograničenja 3](#_Toc500602391)

[2.7. Pretpostavke, zavisnosti i rješenja 4](#_Toc500602392)

[3. Artefakti 5](#_Toc500602393)

[3.1. Arhitektura 5](#_Toc500602394)

[3.2. MVC (Model-View-Controller) arhitekturalni dizajn 7](#_Toc500602395)

[3.3. Use Case 8](#_Toc500602396)

[3.4. ERA model 9](#_Toc500602397)

[3.5. Dijagram klasa 10](#_Toc500602398)

[3.5.1. Sprint 1 10](#_Toc500602399)

[3.5.2. Sprint 2 10](#_Toc500602400)

[3.6. Dijagram slijeda 11](#_Toc500602401)

[3.6.1. Registracija 11](#_Toc500602402)

[3.6.2. Prijava 12](#_Toc500602403)

[3.7. Dijagram aktivnosti 13](#_Toc500602404)

[3.7.1. Izmjena podataka 13](#_Toc500602405)

[4. Web Servis 14](#_Toc500602406)

[4.1. Korisnici web servisa 15](#_Toc500602407)

[4.2. Specifikacija Web servisa 15](#_Toc500602408)

1. Uvod

U današnje vrijeme uz užurbani život, ljudi su sve više i više izloženi velikim količinama stresa i umora koje su najčešći uzroci raznih oboljenja. Napretkom medicine javlja se sve veći broj lijekova koji se primjenjuju u terapijama za razne bolesti uzrokovane spomenutim uzrocima, a sve je veći broj korisnika terapija.

Uvidom u trendove razvoja mobilnih aplikacija i uvođenjem mobilne tehnologije u svakodnevni život, sve je veći broj korisnika mobilnih uređaja. Kako vrijeme odmiče sve mlađi ljudi obolijevaju, a sve stariji upoznaju novije tehnologije te njihovu primjenu u svakodnevnom životu.

Pomnijim proučavanjem metoda liječenja tabletama, primjenom mobilnih tehnologija i kontaktima s osobama koje imaju propisane terapije lijekovima, došli smo na ideju kreirati aplikaciju koja će pomoći korisnicima android uređaja i terapija tabletama praćenje napretka njihovog liječenja i pomoć pri istom.

Smatramo da će aplikacija zaživjeti i steći veliku popularnost među korisnicima zbog svoje jednostavnosti, ali velike važnosti za bolesnike. Uz pregleda lijekova te detaljnih pregleda uputa o korištenju korisnik na jednom mjestu može saznati sve potrebne informacije o tretmanu. Također zbog velikog broja raznih tableta koje korisnici uzimaju teško je pratiti kad je vrijeme za tabletu te pratiti koliko ih je još ostalo što bi u aplikaciji bilo sve nadohvat prsta. Uz navedene mogućnosti aplikacija bi obavještavala korisnika o broju preostalih tableta, vremenu uzimanju te mogućnost bilježenja pregleda kod liječnika.

1. PillCare
   1. Svrha aplikacije

Aplikacija je namijenjena osobama koje imaju problema sa srčanim bolestima te im ista omogućuje prikaz i odabir lijekova za bolesti. Na ovaj način se prate sam unos tableta te napredak liječenja za pojedinu bolest.

* 1. Pojmovi korišteni u projektu

**API** (*Application Programming Interface*) - sučelje za programiranje aplikacija čija je namjena pristup uslugama ili resursima operacijskog sustava ili nekog drugog složenog programa

**GPS** (*Global Positioning Sysem*) - omogućuje pouzdano pozicioniranje, navigaciju i vremenske usluge korisnicima širom svijeta

Korisnik – osoba koja koristi aplikaciju za pristup podacima i kontrolu uzimanja terapije (lijeka)

**Lijek** – podrazumijeva lijek (najčešće tablete) koje korisnik mora uzimati u određenom periodu

**Terapija** – osnovna jedinica praćenja korisnikova uzimanja lijeka, sadrži sve podatke koje sustavu trebaju kako bi mogao obavještavati i pomoći korisniku

**Pojedinačna doza** – korisnik u receptu ima propisanu dozu koju mora uzeti u jednom uzimanju lijeka (jedna, dvije, pola tablete …)

**Konzumiranje** – broj pojedinačnih doza koje korisnik mora uzeti u jednom danu (jednom, dvaput dnevno…)

**Pregled** – podsjetnik koji korisnik može unijeti ukoliko ima zakazan pregled kod liječnika

**Dnevni pregled** – popis događaja koje korisnik ima zakazano na određeni dan, može sadržavati termine pregleda uz već određen termin uzimanja lijeka

* 1. Reference

IEEE Std 830-1998, Software Requirements Specifications

* 1. Specifikacija funkcionalnih zahtjeva

1. **Registracija novih korisnika**- Prije početka korištenja aplikacije korisnik mora kreirati svoj korisnički račun kako bi osigurao svoje podatke te mogao učitati iste na novi uređaj prilikom prijave.
2. **Prijava novih korisnika**- Prijava korisnika odvija se pomoću korisničkog imena i lozinke koji se provjeravaju u online bazi.
3. **Popis lijekova**- Korisniku je omogućen pregled svih lijekova koje je moguće pratiti u aplikaciji.
4. **Prikaz upute o lijeku**- Odabirom lijeka iz liste lijekova dostupne su informacije o lijeku (pakiranju) te upute o korištenju.
5. **Unos nove terapije**- Korisnik odabirom lijeka može unijeti podatke terapije te započeti praćenje uzimanja lijeka.
6. **Pokretanje / Zaustavljanje vremena terapije**- Svaka terapija se nakon unosa mora pokrenuti te također postoji opcija prekida terapije u slučaju promjene lijekova.
7. **Dodavanje novog pregleda**- Uz praćenje uzimanja lijekova korisnik može pohraniti termin pregleda kod osobnog liječnika.
8. **Prikaz povijesti pregleda**- Svi prošli pregledi bilježe se u sustavu kako bi korisnik imao uvid u povijest.
9. **Mapni prikaz ljekarni**- Na mapi unutar aplikacije moguće je pronaći najbliže ljekarne.
10. **Prikaz obavijesti o stanju broja tableta u kutiji**- Kada broj tableta dostigne kritičnu razinu korisniku se ispisuje obavijest koju može ukloniti tek kad je terapija prekinuta ili nova doza kupljena.
11. **Prikaz obavijesti o nadolazećem pregledu**- Samim time to korisnik može pohraniti termin pregleda kod korisnika kreirana je mogućnost obavijesti o nadolazećem pregledu.
12. **Prikaz obavijesti za uzimanje terapije**- Neposredno prije uzimanja tablete korisniku se ispisuje obavijest kako ne bi zaboravio uzeti ili kako bi ga podsjetilo na uzimanje terapije.
13. **Postavke vremena za prikaz obavijesti**- Obavijesti mogu biti postavljene po korisnikovoj volji.
14. **Prikaz dnevnog rasporeda**- Korisnik može na kalendaru pregledati po danu kad su nadolazeća vremena uzimanja tableta ili pregleda.
15. **Barcode skener** – skeniranjem pakiranja lijeka korisnik može dobiti sve podatke o lijeku koji postoje u bazi.
    1. O proizvodu
       1. Perspektiva proizvoda

Aplikacija PillCare mobilna je aplikacija namijenjena korisnicima Android uređaja. Ciljani korisnici su sve osobe koje imaju problema s vođenjem evidencije uzimanja lijekova, informiranja o lijekovima te kontrolom terapije.

* + 1. Funkcija proizvoda

Osnovna funkcionalnost proizvoda je obavještavanje korisnika o primjeni terapije (uzimanju lijeka) te pravovremene obavijesti o stanju pakiranja lijeka te pregledima kod liječnika.

* + 1. Korištene tehnologije

**GitHub** – sustav za verzioniranje koda (proveden kroz alat SourceTree)

**Web**: MySQL, PHP (izrada baze u alatima MySQL Workbench te phpMyAdmin i izrada web servisa u jeziku PHP)

**Mobilne**: Java, SQLite (aplikacija je rađena u Android Studiu, a sadrži uz online bazu i jednu lokalnu)

* 1. Ograničenja

Ograničenja se prvenstveno odnose na pristup internetu koji je neophodan za izvođenje svih funkcionalnosti. Aplikacija može djelovati i bez interneta, no tada može raspolagati podacima koji su do tada prikupljeni bez da je omogućena izmjena postojećih ili dodavanje novih.

* 1. Pretpostavke, zavisnosti i rješenja

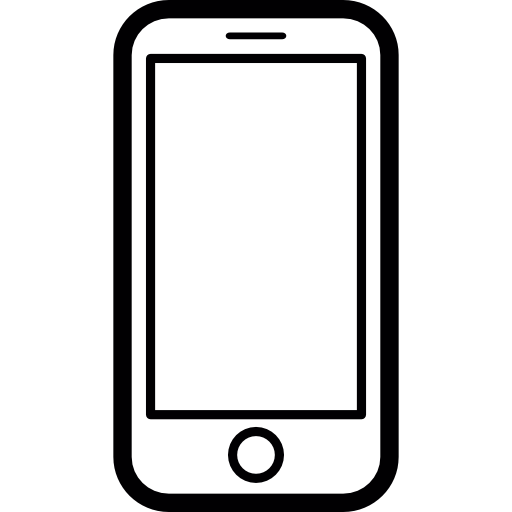
Kako bi korisnik mogao neometano koristiti aplikaciju potrebna mu je internetska povezanost. Internet je potreban za pristup bazi podataka kako bi korisnik imao ažurne informacije. Iz razloga što korisnik može izgubiti vezu s internetom u aplikaciji postoji lokalna baza podataka kako bi korisnik i dalje mogao koristiti aplikaciju, ali moći će raditi s postojećim podacima u lokalnoj bazi te neće moći pregledavati listu svih lijekova ili dodavati novu terapiju.

1. Artefakti
   1. Arhitektura

Korisnik

 Baza podataka

phpMyAdmin

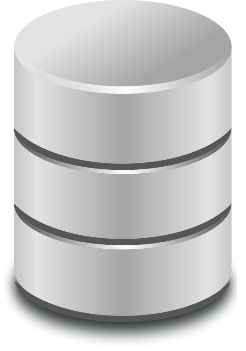
 

JSON (skripte)

POST

Aplikacija

Web Servis





Lokalna baza podataka

Google Maps

Prilikom korištenja aplikacije korisnik pomoću sučelja koristi aplikaciju te provodi razne naredbe i akcije koje rezultiraju promjenama u lokalnoj bazi, online bazi ili na samom sučelju. Aplikacija komunicira sa Web Servisom čija je zadaća omogućiti pristup i CRUD operacije nad online bazom podataka, dok je lokalna baza podataka popunjena korisnikovim podacima potrebnim za rad bez internetske veze. Web servis podatke dostavlja iz baze u obliku JSON datoteke. Budući da aplikacija PillCare sadrži funkcionalnost mapnog prikaza najbližih ljekarni, uz pomoć Google Mapsa prikazati ćemo iste na karti.

* 1. MVC (Model-View-Controller) arhitekturalni dizajn

Controller

(Activity klase)

3

1

Model

(Klase database modula)

View

(XML datoteke)

4

2

Između MVVM i MVC arhitekturalnog dizajna razvoja aplikacija odabrali smo MVC pristup koji kao šta i samo ime govori koristi: sučelje(View), klase upravitelja (Controller) te klasa baze podataka odnosno entiteta (Model).

Strelice između prikazanih dijelova aplikacije prikazuju komunikaciju između slojeva sustava:

1. Obavijesti koje sučelje šalje čim je korisnik odabrao neku opciju ili izvršio određenu akciju na ekranu
2. Na temelju izvršenih akcija korisnika klasa upravitelja provodi određene unaprijed zadane akcije i korake. Važno je napomenuti da upravitelj može kontrolirati i više od jednog sučelja odnosno *view*-a.
3. Prilikom provođenja raznih akcija, klasa upravitelja možda treba pohraniti ili učitati podatke te stoga komunicira s klasama entiteta
4. Klase entiteta vraćaju, izmjenjuju ili brišu podatke na temelju naredbe upravitelja te ih isporučuju istom na daljnju obradu

Na primjeru našeg projekta, MVC proveden je tako da sučelje predstavljaju xml datoteke koje sadrže podatke za generiranje i prikaz sučelja korisniku, klase upravitelj su klase *activity*-a koje „hvataju“ akcije sa sučelje i kontoliraju prikaz i rad s podacima na sučelju te zadnji dio dizajna je model koji sačinjavaju sve klase modula baze podataka odnosno podaci pohranjeni unutar klasa baze.

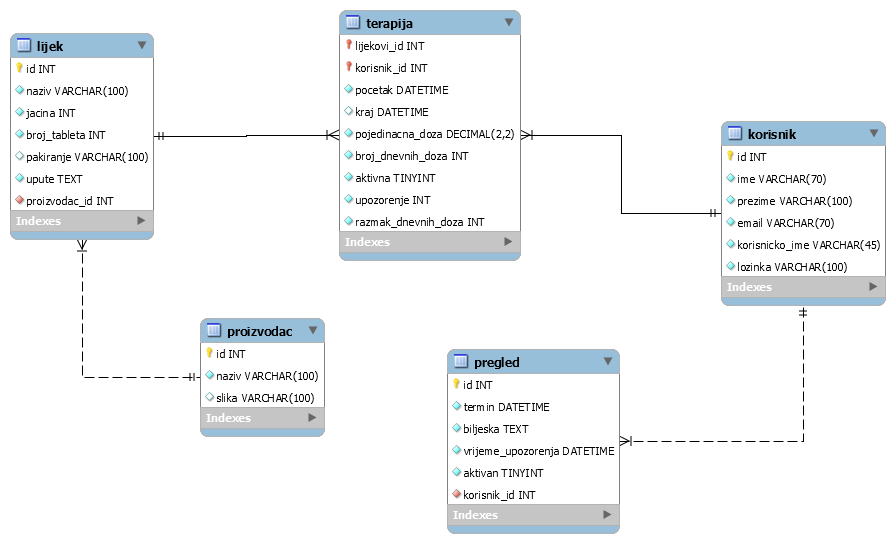
* 1. Use Case



Slika . Use Case dijagram

Na slici 1 je prikazan dijagram slučaja korištenja koji detaljnije prikazuje sam način korištenja i upravljanja aplikacijom. Aplikaciju koristi jedan korisnik koji se prvo mora registrirati te nakon toga se sa unesenim korisničkim imenom i lozinkom prijavljuje u aplikaciju. Nakon što se korisnik uspješno prijavi u aplikaciju, korisniku se kao početni ekran prikazuje lista lijekova. Odabirom pojedinog lijeka, korisniku se prikazuju upute o samom lijeku, odnosno način na koji se lijek uzima. Osim toga, korisnik ima mogućnost dodavanja nove terapije te istu terapiju pokreće ili zaustavlja. Prilikom dodavanja nove terapije, korisnik unosi određene podatke koje se odnose na samo konzumiranje lijeka, kao što je npr. dnevni unos tableta. Ukoliko se broj tableta prilikom uzimanja nove terapije smanji do definirane razine, korisniku se ispisuje obavijest, odnosno upozorenje o nedostatku tableta u kutiji za određeni lijek. Također, korisniku će biti prikazano upozorenje, tj. podsjetnik za konzumiranje tableta u definiranom vremenskom periodu. Ukoliko korisnik želi dodati novi lijek, postoji mogućnost bar-code skenera te se željeni lijek skenira i nakon toga dodaje u bazu lijekova. Kako bi korisnik mogao kontrolirati odrađene i nove preglede, isti se mogu dodavati te će korisnik imati uvid u povijest pregleda kojima je prisustvovao. Također, korisniku će biti prikazana obavijest koja ga podsjeća na dogovoreni pregled za određeni dan. Prikaz dnevnog rasporeda odnosi se na same preglede odrađene taj isti dan te lijekove koje je korisnik konzumirao. Odabir samih funkcionalnosti moguće je odabrati u samom izborniku te se u istom nalazi mogućnost odabira ljekarni koji je zapravo mapni prikaz najbližih ljekarni kako bi korisnik mogao nabaviti potrebne lijekove i na taj način pratiti liječenje. Odabirom korisničkog profila, korisnik može mijenjati vlastite podatke kao što su korisničko ime, lozinka te e-mail.

* 1. ERA model



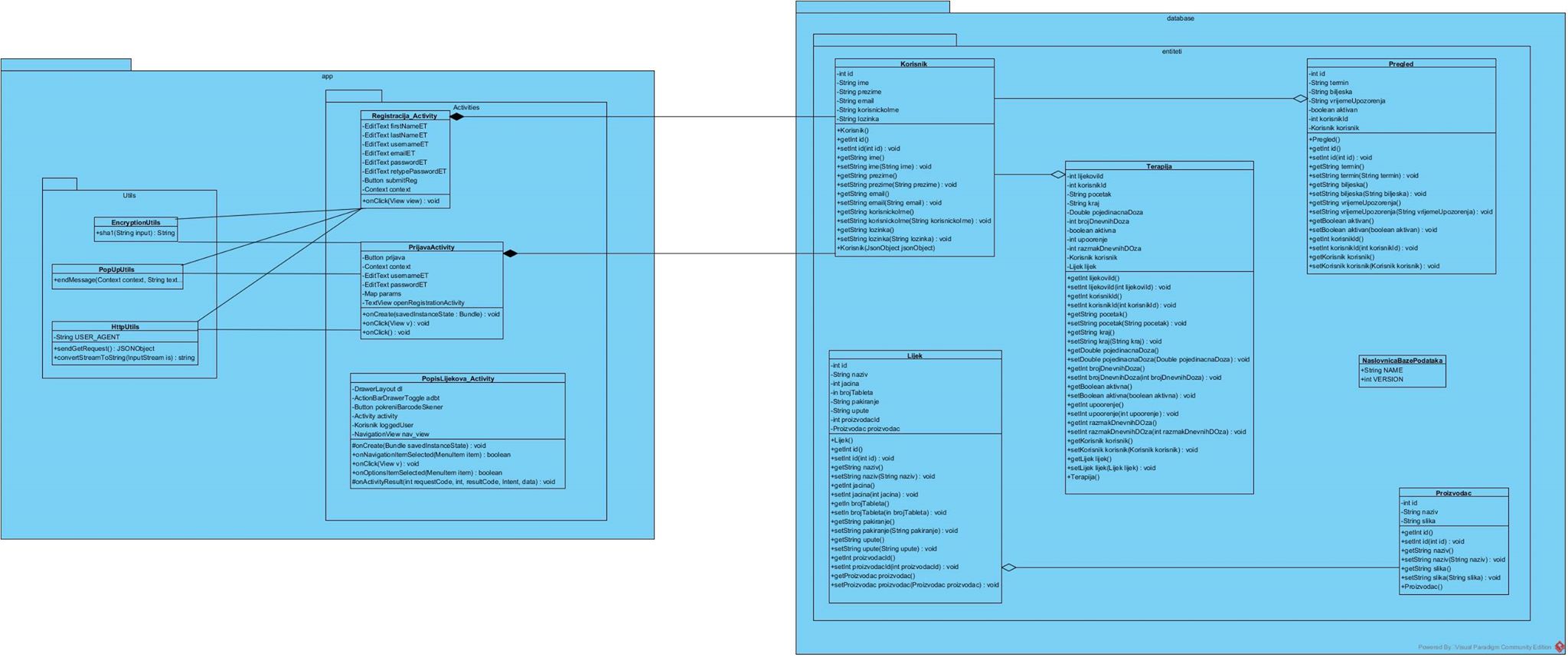
Slika . ERA model

Na slici 2 je prikazan ERA model projekta PillCare koji je kreiran u WorkBenchu te je učitana u phpMyAdmin gdje se administrira ili uređuje. Vidljivo je kako sam ERA model sadrži 5 tablica*: lijekovi, proizvodac*, *terapija*, *korisnik* te *pregled*. Sve tablice su prikazane sa primarnim ključevima i svojstvenim atributima te su povezane sa drugim tablicama. Korisnik može imati više pregleda te se vanjski ključ na korisnika nalazi u tablici *pregled.* Budući da korisnik može upotrebljavati više lijekova te isti lijek može koristiti više korisnika, u model je dodan asocijativni entitet *terapija* te isti sadrži vanjske ključeve na spomenute dvije tablice.

Atributi tablica:

*Napomena- u tablicama se nalaze* ***primarni ključ****, nazivi atributa te vanjski ključ)*

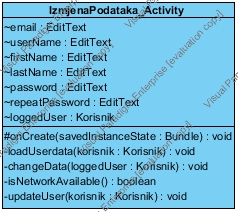
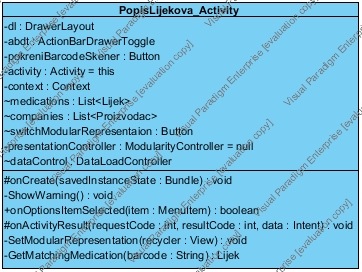
* **lijekovi** (**id\_lijekovi**, naziv, jacina, broj\_tableta, pakiranje, upute, *proizvodac\_id*)
* **proizvodac (proizvodac\_id,** naziv, slika**)**
* **korisnik** (**korisnik\_id,** ime, prezime, email, korisnicko\_ime, lozinka)
* **terapija** (pocetak, kraj, pojedinacna\_doza, broj\_dnevnih\_doza, aktivna, upozorenje, razmak\_dnevnih\_doza, lijekovi\_id, korisnik\_id)
* **pregled** (**pregled\_id**, termin, biljeska, vrijeme\_upozorenja, aktivan, korisnik\_id)
  1. Dijagram klasa
     1. Sprint 1

s

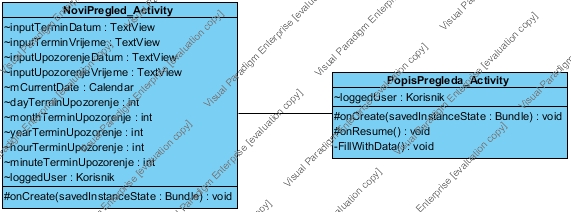
Slika . Dijagram klasa

Slika 3 prikazuje dijagram klasa kreiran za prvi sprint. Dijagram klasa prikazuje dva modula koji u sebi sadrže pakete koji se sastoje od definiranih klasa. Vidljivo kako su klase povezane i ovisne o drugim klasama na način da su povezane vezama asocijacije, agregacije ili kompozicije.

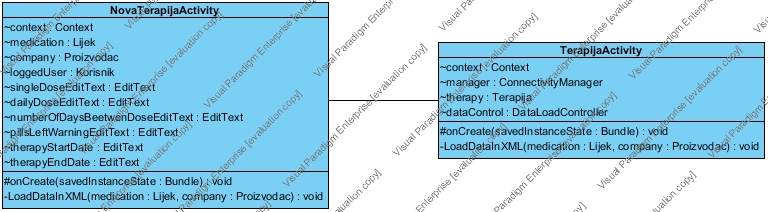
* + 1. Sprint 2

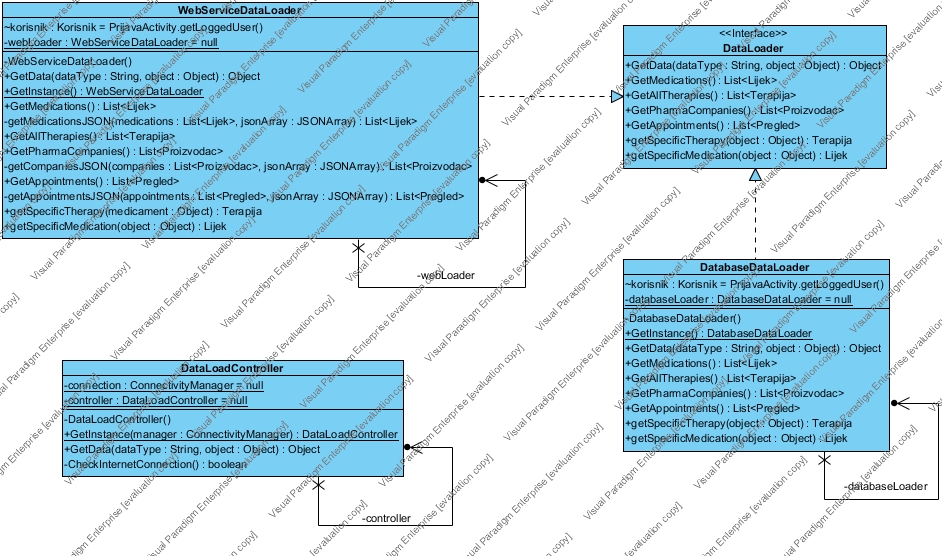
Slika . Funkcionalnost Izmjena podataka Slika . Funkcionalnost Prikaz popisa lijekova



Slika . Funkcionalnost Pregled



Slika . Funcionalnost Terapija

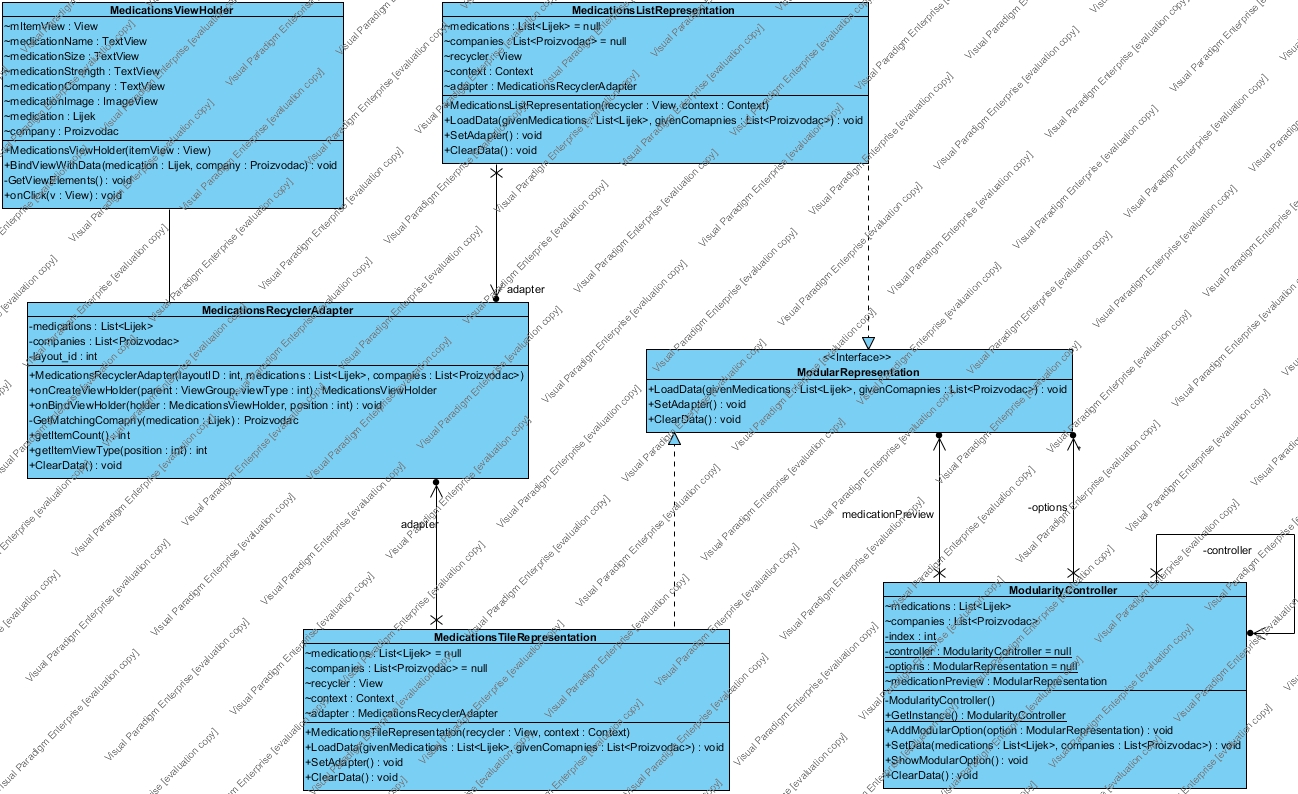


Slika . Klase DataLoaders paketa

Na slici iznad prikazane su klase DataLoaders paketa čija je osnovna namjena dohvaćanje podataka sa web servisa ili lokalne baze podataka.

Ponašanje klasa definirano je sučeljem DataLoader koje sadrži metode koje obije klase moraju sadržavati. Kao što im i samo ime kaže WebServiceDataLoader i DatabaseDataLoader su klase koje implementiraju sučelje te služe za dohvaćanje podataka sa web servisa odnosno iz lokalne baze.

DataLoadController je klasa koja „upravlja“ dohvaćanjem podataka točnije određuje koja će se klasa pozvati odnosno hoće li podaci biti učitani sa web servisa ili lokalne baze. Na temelju boolean varijable koja definira postoji li konekcija sa internetom određuje se od kud će podaci biti dohvaćeni.

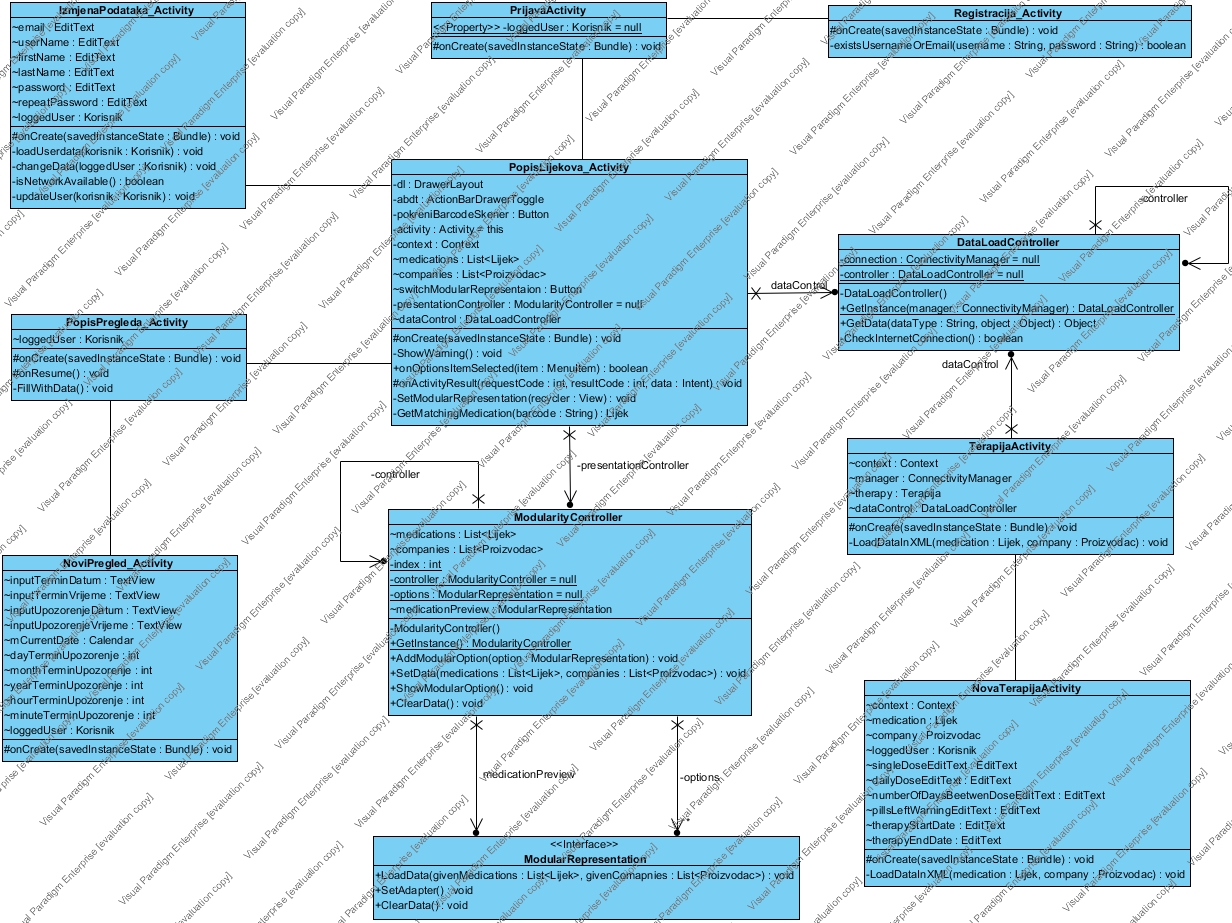


Slika . Adapteri za modularni prikaz popisa lijekova

Na slici iznad prikazane su klase koje su korištenje pri prikazu popisa lijekova.

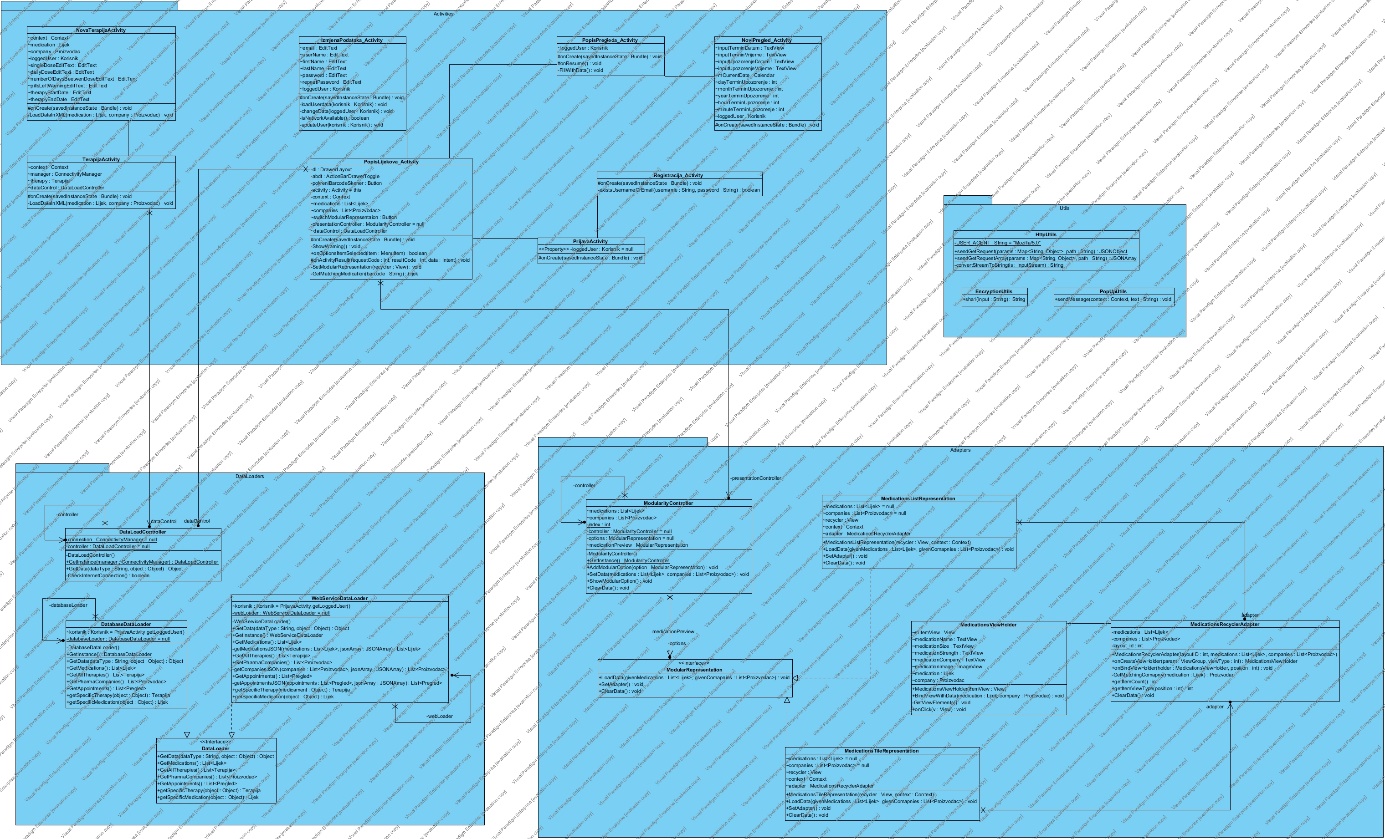
Sučelje ModularRepresentation propisuje ponašanje klasa za prikaz podataka. MedicationsTileRepresentation i MedicationsListRepresentation implementraju sučelje te kontroliraju prikaz podataka u obliku pločica (tiles) ili liste. Obije klase rada sa RecyclerView-om, ViewHolderom i Adapterom. ViewHolder kontrolira ponašanje jednog view-a odnosno item-a RecyclerView-a, klasa adaptera zadužena je za sve ViewHoldere kreirane za item-e koji će biti prikazani. Prije spomenute klase koje implemetiraju sučelje uz to što upravljaju adapterom sadržavaju dodatne metode i polja za rad.

ModularityController je klasa koja „upravlja“ izmjenom i prikazom različitih pregleda lijekova. Klasa sadrži listu mogućih prikaza te metode za prikaz jednog od mogućih prikaza, dodavanje novih te dodatne metode za rad.



Slika . Sve klase kreirane u prva dva sprinta

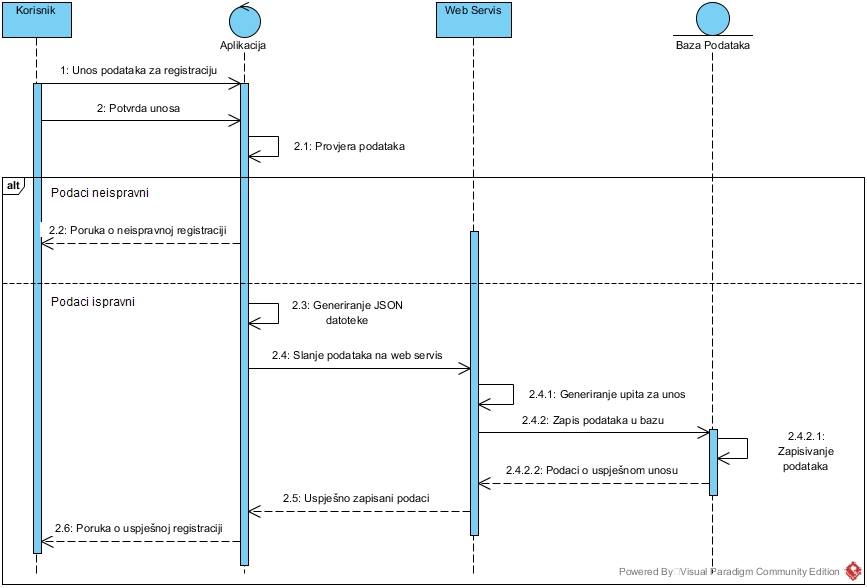
Slika 10 prikazuje sve klase koje su kreirane tokom rada na projektu u prva dva sprinta. Ovaj dijagram povezuje sve dosad prezentirane dijelove (klase) te ih objedinjuje u jedan dijagram koji prikazuje klase i veze među njima.



Slika . Dijagram klasa svih paketa

Na slici prikazani su svi paketi korišteni u aplikaciji. Ova slika objedinjuje sve dosad prezentirane i pojašnjene klase te pakete u kojima se nalaze. Također uz sve navedeno prikazana je komunikacija među klasama i paketima.

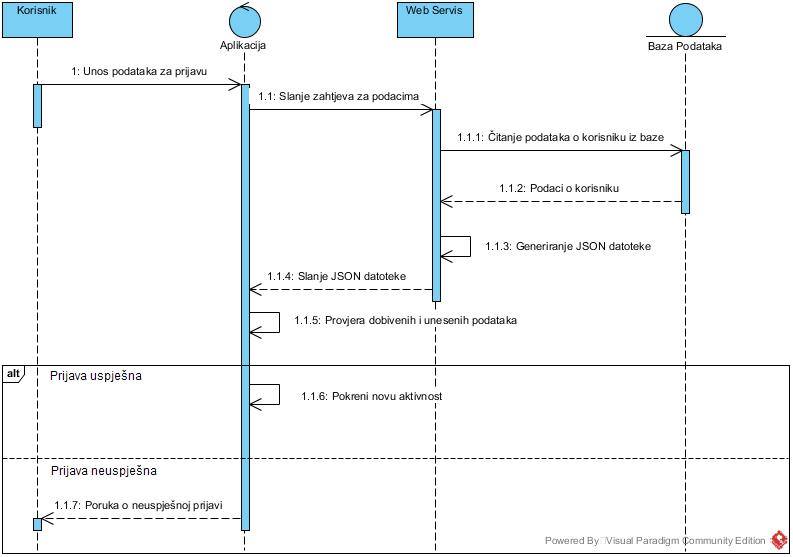
* 1. Dijagram slijeda
     1. Registracija



Slika . Dijagram slijeda Registracija

Dijagram opisuje komunikaciju svih elemenata sustava i vanjskog elementa (korisnika). Korisnik najprije unosi podatke za registraciju, zatim se potvrđuje unos te se podaci provjeravaju. Ukoliko je unos neispravan ispisuje se poruka o neuspješnoj registraciji. S druge strane ukoliko je unos ispravan, generira se JSON datoteka te se podaci šalju dalje na Web servis gdje se generira upit te se podaci zapisuju u bazu i odakle se dalje šalju uspješno zapisani podaci.

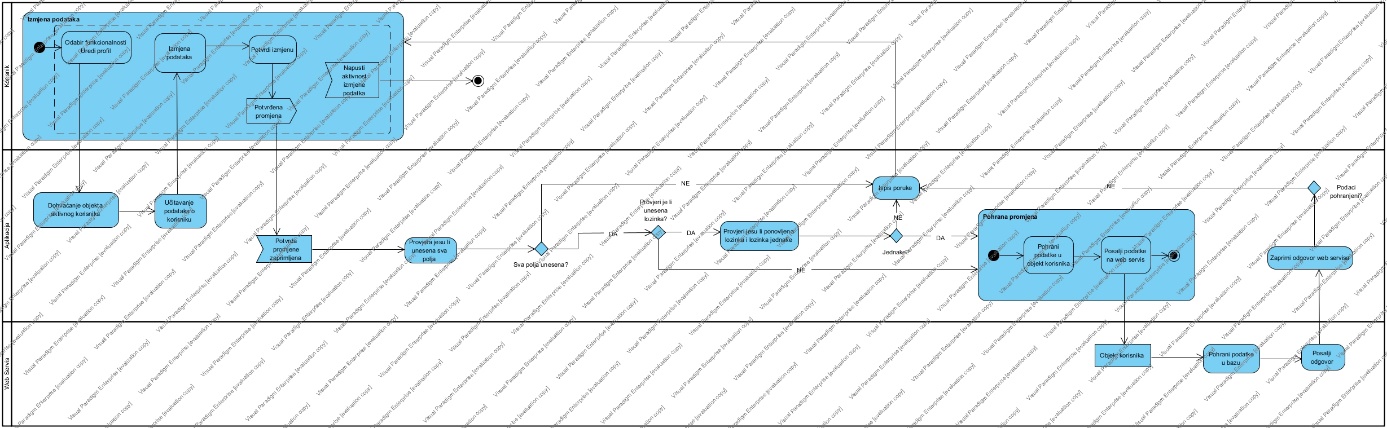
* + 1. Prijava



Slika . Dijagram slijeda Prijava

Dijagram opisuje komunikaciju svih elemenata sustava i samog korisnika. Najprije se unose podaci za prijavu te se dalje šalju sami zahtjevi za podacima gdje se uz pomoć Web servisa čitaju iz baze te se generira i šalje JSON datoteka na temelju koje se provjeravaju dobiveni i uneseni podaci. Ukoliko je prijava uspješna, pokreće se nova aktivnost, no ukoliko je prijava neuspješna ispisuje se poruka o neuspješnoj prijavi.

* 1. Dijagram aktivnosti
     1. Izmjena podataka



Slika . Dijagram aktivnosti Izmjena podataka

Dijagram opisuje način na koji korisnik mijenja postojeće podatke. Odabirom na menu mogućnost *Uredi profil* korisniku se prikazuje aktivnost sa svim podacima osim lozinke. Podaci se odnose na ime, prezime, korisničko ime i email. Ukoliko korisnik mijenja neke od navedenih podataka, ali ne i lozinku, aplikacija provjerava jesu li unesena sva polja sa podacima. Ukoliko jesu, podaci se ažuriraju i spremaju u bazu. No, ukoliko neki od podataka nisu uneseni, korisniku se ispisuje poruka da treba unijeti sve podatke. Ako korisnik uz neki od podataka želi promijeniti i lozinku, potrebno je unijeti i potvrdu lozinku pri čemu se provjerava unesena lozinka sa potvrdom lozinke. Ako su lozinka i potvrda lozinke jednake, korisnički podaci se ažuriraju i spremaju u bazu.

1. Web Servis

Kako bi aplikacija PillCare radila u skladu sa potrebama korisnika, kreirali smo web servis. Na ovaj način aplikacija koristi vlastitu bazu podataka u kojoj se nalaze / će se nalaziti potrebni podaci, kao što su: korisnici, lijekovi, pregledi i terapije. Za potrebe rada ove aplikacije unajmili smo server na stranicama 000webhost-a (<https://www.000webhost.com/>) na kojem je instaliran phpMyAdmin za lakši pristup i upravljanje bazom podataka. Za potrebe izrade aplikacije te njeno povezivanje sa bazom podataka kreirane su pripadne skripte koje služe za formatiranje podataka u JSON format, a iste se nalaze na već navedenom serveru. Kako nam za ispravno funkcioniranje aplikacije nije bila bitna web aplikacija, arhitekturu web servisa prikazali bi na sljedeći način:



Baza podataka

phpMyAdmin

POST

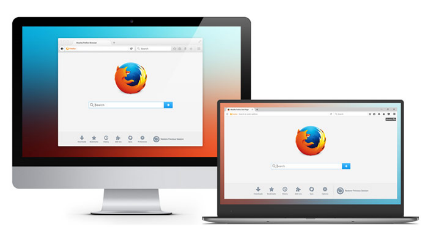
POST/GET



Web Server

HTML (prikaz podataka)

JSON (skripte)



Web Preglednik Mobilna aplikacija

* 1. Korisnici web servisa

Kao što je prije napomenuto, za potrebe rada aplikacije nije bila potrebna izrada web aplikacije zbog čega su jedini korisnici web servisa, odnosno aplikacije PillCare korisnici mobilne aplikacije. Navedene osobe na indirektan način korištenjem mobilne aplikacije koriste skripte na web servisu kako bi dohvatili određene podatke ili ažurirali iste.

**Podaci za prijavu i rad na Web serveru:**

1. Stranica Web servera: <https://www.000webhost.com/>
2. Login (000webhost):
   1. Korisničko ime: andreadanzante@gmail.com
   2. Lozinka: airprojekt
3. Projekt **pillcare**
   1. Stranica: <https://pillcare.000webhostapp.com/> - korisnici
   2. Stranica: <https://pillcare.000webhostapp.com/lijekovi.php> - lijekovi
   3. Stranica: <https://pillcare.000webhostapp.com/terapija.php> - terapije
   4. Stranica: <https://pillcare.000webhostapp.com/pregled.php> - pregledi
4. PhpMyAdmin – automatski preko Web servera ili preko poveznice:
   1. <https://databases-auth.000webhost.com/index.php>
      1. Korisničko ime: id3348135\_air1715
      2. Lozinka: airprojekt
   2. Specifikacija Web servisa

SPECIFIKACIJA SERVISA – JSON

SERVISI:

* dodajKorisnika
* dodajPregled
* dodajTerapiju
* korisnik
* lijekovi
* pokreniZaustaviTerapiju
* pregled
* proizvodac
* provjeriKorisnika
* specificnaTerapija
* terapija
* traziLijek

METODE SERVISA:

* Putanja: https://pillcare.000webhostapp.com

1. **dodajKorisnika** Putanja: /dodajKorisnika.php
   1. Ulaz: user, update
   2. Izlaz: Dodani novi korisnik ili izmijenjen postojeći
   3. Metoda: GET
2. **dodajPregled** Putanja: /dodajPregled.php
   1. Ulaz: appointment
   2. Izlaz: Dodani novi pregled
   3. Metoda: GET
3. **dodajTerapiju** Putanja: /dodajTerapiju.php
   1. Ulaz: therapy
   2. Izlaz: Dodana nova terapija
   3. Metoda: GET
4. **korisnik** Putanja: /korisnik.php
   1. Ulaz: username
   2. Izlaz: Dohvaćeni podaci o korisniku
   3. Metoda: GET
5. **lijekovi** Putanja: /lijekovi.php
   1. Ulaz: user\_id
   2. Izlaz: Dodana terapija korisniku na temelju odabranog lijeka
   3. Metoda: GET
6. **pokreniZaustaviTerapiju** Putanja: /pokreniZaustaviTerapiju.php
   1. Ulaz: user, medicament, start
   2. Izlaz: Pokrenuta ili zaustavljena terapija za određenog korisnika
   3. Metoda: GET
7. **pregled** Putanja: /pregled.php
   1. Ulaz: user
   2. Izlaz: Podaci o pregledima određenog korisnika
   3. Metoda: GET
8. **proizvodac** Putanja: /proizvodac.php
   1. Ulaz: type
   2. Izlaz: Podaci o proizvođačima
   3. Metoda: GET
9. **provjeriKorisnika** Putanja: /provjeriKorisnika.php
   1. Ulaz: username, email
   2. Izlaz: Korisnik na temelju korisničkog imena i emaila
   3. Metoda: GET
10. **specificnaTerapija** Putanja: /specificnaTerapija.php
    1. Ulaz: user, medicament
    2. Izlaz: Podaci o terapiji za određenog korisnika
    3. Metoda: GET
11. **terapija** Putanja: /terapija.php
    1. Ulaz: user
    2. Izlaz: Podaci o terapiji
    3. Metoda: GET
12. **traziLijek** Putanja: /traziLijek.php
    1. Ulaz: med\_id
    2. Izlaz: Podaci o lijeku
    3. Metoda: GET