Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Andrej Rus

Podpora predpisovanja zdravil

diplomsko delo

interdisciplinarni univerzitetni
študijski program prve stopnje
računalništvo in matematika

Mentor: dr. Andrej Brodnik
Somentor: Nenad Živković

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Zdravniške napake so redke, a imajo lahko hude posledice. Da bi se njihovo število še zmanjšalo, se uveljavljajo tako imenovani svetovalni sistemi, ki zdravniku samodejno ponudijo dodatno informacijo. Pri tem ostaja končna odločitev še vedno v rokah zdravnika. Omenjeni sistemi lahko vključujejo metode umetne inteligence in strojnega učenja ali pa ne. Sistemi, ki ne vključujejo metod umetne inteligence in strojnega učenja, se običajno naslanjajo na podatke, katerih količina je zelo velika, se neprestano posodabljajo ali pa je njihovo razumevanje zelo kompleksno. V diplomski nalogi začrtajte svetovalni sistem, ki bo uporaben pri predpisovanju zdravil. Ob tem uporabite podatke klinične poti, ki vsebuje predpisovanje zdravil in upoštevanje kontraindikacije predpisovanega zdravila. Sistem implementirajte za učinkovino klozapin v zdravljenju različnih psihoz. Implementirani sistem ovrednotite in razmislite o morebitnih razširitvah.

Zahvaljujem se mentorju doc. dr. Andreju Brodniku ter somentorju s strani podjetja Parsek Nenadu Živkoviću za pomoč pri pisanju diplomske naloge in ves čas, ki sta ga vložila. Posebna zahvala gre mojim staršem, starim staršem, dekletu, bratu, sestri in vsem prijateljem, ki so verjeli vame ter me redno opominjali, naj nalogo čim prej pripeljem do konca.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
CDS	clinical decision support	pomoč pri odločitvah v zdravstvu
CDSS	clinical decision support system	sistem za pomoč pri odločitvah v zdravstvu
FHIR	Fast Healthcare Interoperability Resources	sistem za upravljanje podatkovnih baz
EHR	electronic health record	elektronski zdravstveni karton
HTTP	HyperText Transfer Protocol	protokol na aplikacijski plasti, ki je temelj prenosa informacij na spletu
REST	Representational State Transfer	način interoperabilnosti med računalniškimi sistemi
SCRUD	search, create, read, update, delete	išči, ustvari, preberi, ponastavi, zbriši
JSON	JavaScript Object Notation	notacija za označevanje JavaScript objektov
HL	Health Language	jezik za izmenjavo medicinskih podatkov
API	Application Programming Interface	programski vmesnik
URL	Uniform Resource Locator	enolični krajevnik vira
SSL	Secure Sockets Layer	protokol za prenos podatkov zaupne narave

kratica	angleško	slovensko
TLS	Transport Layer Security	protokol za prenos podatkov zaupne narave
ATC	Anatomical Therapeutic Chemical	Anatomsko terapevtsko kemična
RBAC	Role-Based Access Control	kontrola dostopov, ki temelji na vlogah
ABAC	Attribute-Based Access Control	kontrola dostopov, ki temelji na atributih
HTML	HyperText Markup Language	označevalni jezik za strukturiranje in prikazovanje vsebin na spletu
CSS	Cascading Style Sheets	kaskadne stilske podloge

Povzetek

Naslov: Podpora predpisovanja zdravil

Avtor: Andrej Rus

Sistem za pomoč pri odločitvah v zdravstvu je pomemben del elektronskega zdravstvenega sistema. Diplomska naloga govori o tem, kako implementirati program, ki nudi pomoč zdravniku pri predpisovanju zdravil na klinični poti v primeru vpeljave zdravilne učinkovine klozapin v zdravljenje. Predstavljena rešitev je zgrajena po principu sistema, ki temelji na predhodnem znanju in ne vključuje umetne inteligence, ampak upošteva vnaprej podana pravila za odločanje klinične poti. Storitev je namenjena temu, da zmanjšuje napake, ki bi utegnile škodovati pacientu pri napačno predpisanem zdravilu, vendar ne omejuje zdravnika pri njegovem delu, temveč mu le pomaga.

Ključne besede: predpisovanje zdravil, klinična pot, elektronski zdravstveni karton, FHIR, EHR, CDS, CDSS, zdravstvo.

Abstract

Title: Support for medical prescriptions

Author: Andrej Rus

The clinical decision support system is an important part of the electronic healthcare system. The thesis focuses on how to implement a program that helps a doctor when prescribing medicines on the clinical pathway in case of introducing the active substance clozapine into treatment. The presented solution is built on the principle of a system that is based on previous knowledge and does not include artificial intelligence, but rather complies with the rules for determining the clinical pathway. The service is designed to minimize defects that could harm the patient in an incorrectly prescribed medicine but does not limit the doctor's work but only helps them.

Keywords: medication prescribe, clinical pathway, electronic health record, FHIR, EHR, CDS, CDSS, medical service.

Poglavje 1

Uvod

1.1 Motivacija

Zdravstvene napake so v Združenih državah Amerike tretji najpogostejši vzrok smrti [1]. Med te napake štejejo stanja, ki bi jih bilo mogoče preprečiti z idealnim ravnanjem in idealnimi razmerami. Na primer napačno postavljene diagnoze, napačno predpisana zdravljenja, nepravilno predpisani odmerki zdravil, slaba oprema, nepravočasno priskrbljena ali pretečena zdravila, slabe higienske razmere in drugo [2].

Avtorjeva osebna motivacija pri obravnavani temi je, da bi rad preizkusil, koliko truda je potrebno vložiti, da dobimo neko rešitev, ki bi v realnem primeru lahko rešila življenje.

Napredovanje na področju farmacije in zdravstva je hitro, zato zdravniki niso ves čas na tekočem s kontraindikacijami[1] določenih zdravil in bolezenskih stanj. Po eni strani pričakujemo od zdravnikov, da vse vedo, a vendar se stvari tako hitro spreminjajo, da je to skoraj nemogoče. Postavlja se vprašanje, zakaj ne bi zdravniku ponudili podpore pri njegovem

[1]Kóntraindikácija je kriterij za odložitev medicinskih ukrepov (uporabe določenega zdravila, medicinskega pripomočka ali postopka zdravljenja) zaradi stanja bolnika, dejavnikov ali drugih okoliščin. Kontraindikacijo za uporabo določenega medicinskega ukrepa lahko na primer predstavljajo sočasne bolezni, alergije, določen genotip, predhodni neželeni učinki zdravila ali skupine zdravil, starost, spol, predispozicije [3].

odločanju. Zbirke podatkov o kontraindikacijah zdravil imamo, podatke o pacientu imamo. Če to združimo, lahko pri predpisovanju specifičnega zdravila pacientu zdravstvenoinformatična rešitev pomaga zdravniku pri tem, da ga opozori na določene kontraindikacije.

1.2 Cilji

Osnovni cilj diplomske naloge je napisati delujoč program, ki bi nudil zdravniku pomoč pri predpisovanju zdravil na klinični poti. Zaradi široko zastavljenega problema se bomo osredotočili na le eno zdravilno učinkovino, ki bi jo zdravnik predpisoval. Pri implementaciji bo poudarek na spoznavanju standarda FHIR[2] in pisanju rešitve na način, da bo preprosto razširljiva na več zdravilnih učinkovin. Internacionalizaciji in varnosti pri implementaciji ne bomo posvečali pozornosti.

1.3 Klinična pot klozapina

Klinična pot je orodje, ki zdravstveneni ekipi omogoča racionalno in na znanstvenih dokazih utemeljeno obravnavo pacienta, spremljanje opravljenega dela in kazalnikov kakovosti, natančnejše dokumentiranje ter lažjo notranjo presojo zdravstvene prakse. Obenem pomaga pri seznanjanju pacienta s predvidenim potekom njegove zdravstvene obravnave in je dober pripomoček za izračun stroškov obravnave [4]. Del klinične poti je predpisovanje zdravil pacientu. Pri predpisovanju zdravil mora biti zdravnik pozoren na kontraindikacije. Preverjanje teh na podlagi vpeljave zdravilne učinkovine klozapin v zdravljenje obravnava to diplomsko delo.

Klozapin je snov za zdravljenje psihoz (npr. shizofrenija), ki se med drugimi nahaja v zdravilu Leponex. Gre za zdravilo, ki deluje na veliko različnih receptorjev in ima zaradi tega veliko število kontraindikacij z drugimi bolezenskimi stanji in zdravili. Nazorno kaže, na koliko različnih dejavnikov mo-

[2]FHIR® je registrirana blagovna znamka HL7.

rajo biti zdravniki pozorni pri vpeljavi enega zdravila v zdravljenje, obenem pa prikazuje, kako uporaben je lahko program, ki nudi pomoč pri predpisovanju takih zdravil.
Absolutne kontraindikacije za zdravljenje s Klozapinom [5]:

- bolniki, pri katerih ni možno opravljati rednih preiskav krvi;
- anamneza toksične ali idiosinkratične granulocitopenije/agranulocitoze (z izjemo granulocitopenije/agranulocitoze zaradi predhodne kemoterapije);
- anamneza agranulocitoze zaradi zdravila Leponex;
- okvarjeno delovanje kostnega mozga;
- neurejena epilepsija;
- alkoholne in druge toksične psihoze, zastrupitev z zdravili, komatozna stanja;
- cirkulatorni kolaps in/ali depresija centralnega živčevja iz kakršnegakoli vzroka;
- huda ledvična ali srčna obolenja (na primer miokarditis);
- aktivna bolezen jeter, povezana z navzeo, anoreksijo ali ikterusom; progresivna bolezen jeter, odpoved jeter;
- paralitični ileus;
- zdravljenja z zdravilom Leponex ni dovoljeno začeti ob sočasni uporabi snovi, za katere je znano, da s precejšnjo verjetnostjo povzročajo agranulocitozo; bolnikom je treba preprečiti, da bi sočasno uporabljali depo obliko katerega od antipsihotikov.

Vidimo, da je kontraindikacij zelo veliko in da se med seboj razlikujejo glede na način, kako jih lahko preverjamo. Primerjajmo kontraindikaciji:

1. paralitični ileus in

2. bolniki, pri katerih ni možno opravljati rednih preiskav krvi.

Pri prvi bomo lahko pri preverjanju kontraindikacij preprosto iskali po imenu bolezenskega stanja oziroma še boljše po njegovi šifri, ki ni vezana na jezik, ki ga uporabljamo. V drugem primeru pa ne gre za bolezensko stanje ali zdravilno učinkovino in je preverjanje po pacientovem zdravstvenem kartonu bolj zapleteno.

V diplomski nalogi se bomo osredotočili na kontraindikacije, ki jih je mogoče iskati po šifrah znotraj mednarodno uveljavljenih klasifikacij.

Poglavje 2

Uporabljena tehnologija

2.1 Standard FHIR

2.1.1 Splošno

FHIR (*angl. Fast Healthcare Interoperability*[1] *Resources*) je standard, ki opisuje podatkovne gradnike ter API za prenos zdravstvenih podatkov in nadgrajuje bolj togi in zapleteni specifikaciji, kot sta HL7 verzija 2 in HL7 verzija 3 [7]. Trend je, da je vedno večja želja po digitalizaciji zdravstvenih podatkov. Ko pacient obiskujeje različne zdravstvene ustanove (zdravstveni dom, zobna ambulanta, radiološka ambulanta, laboratoriji), moramo imeti enostaven in skladen način za deljenje informacij o pacientu med temi ,,proizvajalci" zdravstvenih podatkov. FHIR sloni na modernih tehnologijah, ki omogočajo učinkovito integracijo in prenos podatkov med različnimi zdravstvenimi ustanovami. Osnovni gradniki, ki sestavljajo FHIR, so [8]:

- Viri – majhne diskretne logične enote podatkov, ki imajo definirano obnašanje in pomen. So najmanjše prenosljive enote, znotraj standarda FHIR je okoli 150 različnih tipov. Njihova majhnost omogoča, da zdravnik, ko na primer potrebuje določene pacientove podatke (npr.

[1]Interoperabilnost je lastnost računalniških sistemov ali programov, da izmenjujejo podatke in jih pametno uporabijo [6].

podatke o alergijah), ne bo dobil ogromnega kosa vseh podatkov o pacientu, kot se je to dogajalo v preteklosti, ampak bo dobil le podatke, ki se nanašajo na vir alergije. Primeri virov: pacienti, stanja, zdravila, zdravstvene nege, podatki o zdravstvenem zavarovanju, alergije.

- Reference – vezi med različnimi viri. Z njihovo uporabo se viri povežejo v mrežo informacij, ki predstavljajo zdravstveni karton.

- Profil – viri nimajo vgrajenih omejitev, kako se jih uporablja. To nalogo prevzemajo profili. Strani, ki si izmenjujejo podatke, definirajo način, kako bodo uporabljale vire s pomočjo profilov.

2.1.2 Prednosti uporabe standarda FHIR

Standard FHIR definira, kako se viri lahko izmenjujejo med različnimi sistemi. Na voljo imamo osnovne operacije SCRUD (*search* – išči, *create* – ustvari, *read* – preberi, *update* – ponastavi, *delete* – izbriši), poleg tega v primerjavi s preteklimi, bolj togimi specifikacijami (npr. HL7 verzija 2 in HL7 verzija 3) prinaša FHIR še izboljšave, kot so [8]:

- **Odprtokodnost:** omogoča, da so programske rešitve bolj transparentne in dostopne, obenem pa se je na podlagi tega oblikovala skupnost razvijalcev, prodajalcev in podjetij, ki prispevajo k izboljšavam specifikacije FHIR.

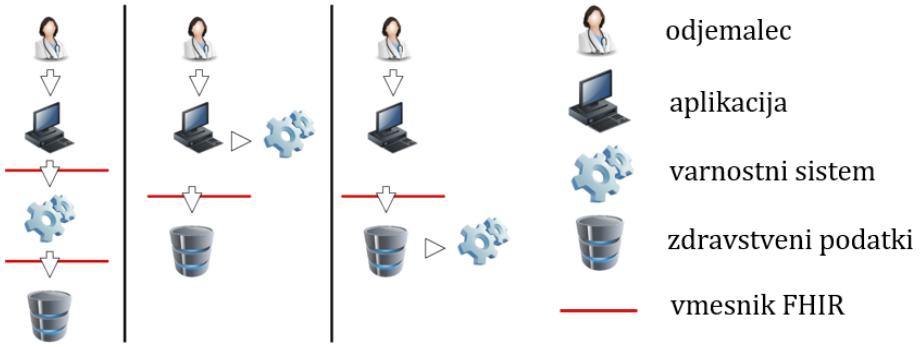
- **REST:** je arhitekturni stil, ki določa skupino arhitekturnih omejitev, ki omejujejo vlogo in funkcije elementov ter medsebojne povezave. Sestavljen je iz odjemalcev in strežnikov. Ko je odjemalec pripravljen na prehod na novo stanje, pošlje zahtevo, ki jo strežnik obdela, in nazaj pošlje pravilne odgovore. Zahteve in odgovori so zgrajeni okoli predstavitev virov (*angl. representations of resources*) [9]. Vsak vir, ki ga želi odjemalec nasloviti, je dostopen z enoznačnim naslovom URL, kar preprečuje dvoumnost in hkraten dostop do dveh različnih virov. REST

uporablja protokol HTTP, pri čemer se uporabljajo standardne metode HTTP: GET, PUT, POST in DELETE [10].

- **Dobra podpora razvijalcem:** na spletni strani razvijalcev `http://hl7.org/fhir/` najdemo veliko primerov uporabe, nasvetov za razvijalce in knjižnic. Poleg tega obstajajo razni odprtokodni projekti, ki nudijo podporo razvijalcem pri spoznavanju standarda FHIR. Eden takih je dostopen na naslovu `http://hapi.fhir.org/` in nudi grafični vmesnik za uporabo standarda FHIR.
- **Temelji na modernih spletnih standardih:** XML, JSON, OAuth, Atom, REST ...

2.1.3 Varnost

FHIR ni varnostni protokol, niti ne definira funkcionalnosti, ki bi bila povezana z varnostjo. Je pa res, da definira izmenjevalni protokol na način, ki ga je treba uporabljati z varnostnimi protokoli, definiranimi izven FHIR. V praksi se uporabljajo različni načini, kdaj in kako zagotoviti varnost.



Slika 2.1: Različne vmestitve varnostnega sistema.

Odjemalec komunicira z varnostnim sistemom (avtentikacija in kontrola dostopov) na različne načine, glede na implementacijo rešitve (slika 2.1). Specifikacija FHIR privzema, da varnostni sistem obstaja in da ga je možno namestiti pod ali nad FHIR API [7].

Pri področjih varnosti, kjer je za varnost poskrbljeno z ostalimi protokoli in standardi, gre za družino protokolov, ki jih na spletu najdemo pod kratico AAAS (*angl. Authentication, Authorization and Accounting with Secure Transport*) [11]:

- **Komunikacijska varnost** (*angl. secure transport*) – vsi izmenjani podatki morajo biti zaščiteni z uporabo TLS ali SSL (na primer https). TLS ali SSL zagotavlja varno izmenjavanje podatkov med dvema stranema, ki se začne z rokovanjem, kjer se odjemalec in strežnik sporazumeta glede enkripcijskih algoritmov ter kriptografskih ključev [12]. Na sliki 2.1 to pomeni, da moramo zagotoviti varen prenos informacij pri vsaki izmed belih puščic.

- **Avtentikacija** (*angl. authentication*) – priporočena je OAuth avtentikacija. Pri klasičnem modelu odjemalec-strežnik, se odjemalec navadno avtenticira z uporabniškim imenom in geslom (prva vertikala na sliki 2.1). To odpira vrsto problemov, kot so: ustvarjanje avtentikacijskih mehanizmov strežnika, shranjevanje gesel na strežniku in narava šibkih gesel. OAuth rešuje ta problem z uvedbo dodatnega nivoja za avtentikacijo in premik avtentikacije stran od strežnika. Namesto kombinacije uporabniškega imena in gesla, odjemalec dobi žeton, s katerim se avtenticira na strežniku. Žetoni se dodeljujejo s strani tretje osebe (avtentikacijskega strežnika), ki mu zaupata odjemalec in strežnik [13]. Modelu z avtentikacijo OAuth je najbolj podobna druga vertikala na sliki 2.1.

- **Avtorizacija in kontrola dostopov** (*angl. authorization and access control*) je namenjena temu, da nimajo vsi odjemalci enakih pravic do pridobivanja in spreminjanja podatkov. Poznamo dva modela za kontrolo dostopov, in sicer [14]:

  - RBAC (*angl. Role-Based Access Control*) je kontrola dostopov glede na različne vloge, ki jih dodeljujemo skupinam odjemalcev.

Vloga torej opredeljuje dovoljenja, ki jih ima posamezni odjemalec. Pri vsakem poizkusu dostopanja do virov se preveri odjemalčeva vloga, potem pa mu je dostop dovoljen ali prepovedan.

– ABAC (*angl. Attribute-Based Access Control*) je kontrola dostopov s pomočjo dodatnih atributov. Pri vsakem dostopanju do vira se preverijo varnostni atributi in če je atribut v skladu z varnostnim sistemom, ima odjemalec pravico do dostopanja do virov. Pri standardu FHIR je ta atribut imenovan varnostna oznaka (*angl. security label*).

- **Zabeležba** (*angl. accounting/audit log*) – FHIR zagotavlja vire, ki omogočajo sledenje izvora, avtorstvo, zgodovino, statuse in dostope do virov.

2.1.4 Strežnik FHIR

Na spletu najdemo veliko strežnikov, ki uporabljajo standard FHIR in so javno dostopni. Z viri rokujemo z osnovnimi HTTP metodami, kot so *GET, POST, PUT, UPDATE, DELETE*. Pri izbiri strežnika, ki je podlaga za pisanje te diplomske naloge, so kriteriji za izbiro brezplačna uporaba, napolnjenost strežnika s testnimi podatki ter morebiten grafični vmesnik. Na podlagi teh kriterijev smo izbrali strežnik FHIR, dostopen na internetnem naslovu `http://hapi.fhir.org`. Določene vire na strežniku smo morali zaradi ozko zastavljenega problema (omejitev za zdravilno učinkovino klozapin) tvoriti sami. Ker brezplačni testni strežniki ne zagotavljajo, da bi podatki, vnešeni s strani zunanjih odjemalcev, trajno ostali na strežniku, naredimo skripto, ki ob morebitnem izbrisu testnih podatkov le-te ponovno ustvari.

2.2 Pomoč pri odločitvah v zdravstvu

Pomoč pri odločitvah v zdravstvu v spletu največkrat najdemo v obliki kratice CDS (*angl. Clinical Decision Support*). Gre za širok pojem, ki združuje veliko različnih načinov, kako lahko računalnik pomaga zdravniku pri njegovem delu. Posvetili se bomo podpori zdravniku pri odločanju pri predpisovanju zdravil na klinični poti.

Zdravila se vse bolj razvijajo in so v večini primerov vključena v medicinske terapije. To prinese veliko pozitivnih učinkov, obenem pa se moramo zavedati, da lahko povzročijo tudi znatno škodo, če so predpisana napačno. Sistemi za pomoč pri odločitvah v zdravstvu (*angl. clinical decision support system – CDSS*) so dveh tipov, in sicer [15]:

- sistemi, ki temeljijo na predhodnem znanju (*angl. knowledge-based CDSS*) – v večini primerov jih sestavljajo trije deli: baza znanja, logika za odločanje in mehanizem za komunikacijo;
- sistemi, ki se ,,učijo sami" in ne temeljijo na predhodnem znanju (*angl. non-knowledge-based CDSS*). Temeljijo na umetni inteligenci in strojnem učenju, ki omogoča računalniškim programom, da se učijo iz preteklih primerov. Ker pri teh sistemih težko pojasnimo, zakaj se je sistem odločil, kot se je odločil, se v praksi manj uporabljajo [15].

Področij, kjer nam lahko računalnik pomaga pri predpisovanju zdravil, je več. Lahko preverjamo neskladje sestavin zdravila z alergijami pacienta, predpisano doziranje, neskladje dveh sočasno predpisanih zdravil, neskladje zdravila s preteklimi pacientovimi boleznimi, podvajanje terapije in drugo. Pri tem je pomembno, da imamo dobro zastavljen in enoten sistem zdravil, bolezni, stanj in meritev.

2.2.1 Dogodkovno gnano programiranje

Dogodkovno gnano programiranje je pristop programiranja, pri katerem na posamezen dogodek ,,obesimo" (*angl. hook*) rokovalnik (*angl. handler*). Ko

se dogodek zgodi, se sproži eden ali več rokovalnikov, ki se na dogodek odzovejo in ga obravnavajo [16]. V našem primeru je vir dogodkov uporabnik, ki komunicira s programom preko spletne aplikacije.

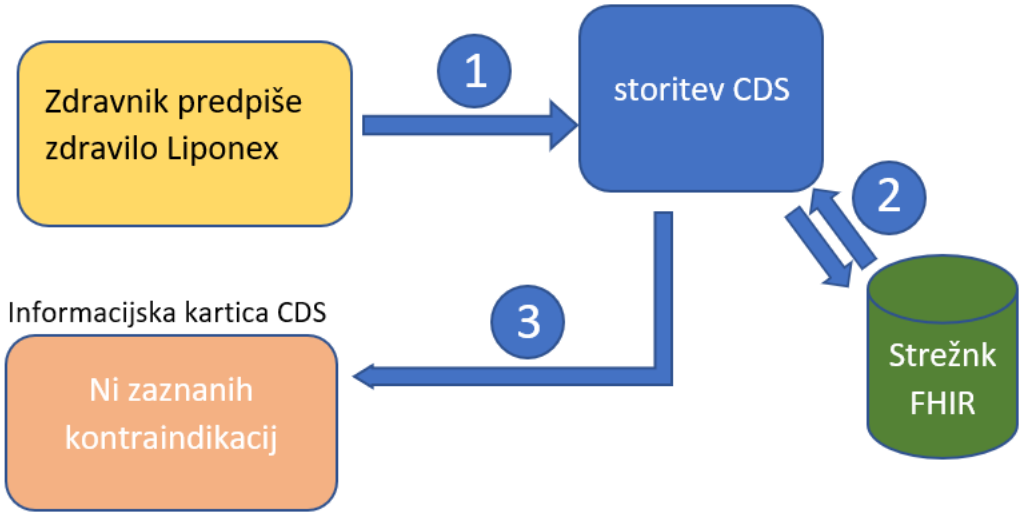
Pri implementaciji storitve CDS smo uporabili specifikacijo CDS Hooks. Gre za specifikacijo, ki opisuje RESTful vmesnik storitve CDS ter komunikacijo med sistemom CDS in elektronskim zdravstvenim kartonom (*angl. electronic health record – EHR*). Komunikacija poteka z uporabo protokola HTTP, izmenjani podatki pa so v obliki JSON struktur [17]. Da vzpostavimo storitev CDS na podlagi specifikacije CDS Hooks, nam pomaga specifikacija OpenAPI, ki je dostopna na naslovu `https://cds-hooks.org/specification/1.0-api.yaml`. Specifikacija OpenAPI s pomočjo orodja Swagger[2] omogoča generiranje programske kode (metode, funkcije, konstrukti), ki nam kasneje služi kot temelj za storitev CDS.

Ko zdravnik obravnava določenega pacienta, ima pred seboj odprt njegov elektronski zdravstveni karton. Vnašanje podatkov v karton (npr. predpis zdravila) predstavlja dogodke, na katere se odzovejo ustrezni rokovalniki. Pošlje se zahteva na storitev za pomoč pri odločitvah v zdravstvu (na sliki 2.2 puščica, označena s številko 1). Storitev CDS obravnava zahtevo ter vrne odgovor zdravniku (na sliki 2.2 puščica, označena s številko 3) v obliki kartic CDS (*angl. CDS Cards*), ki jih predpisuje specifikacija CDS Hooks. Kartice so lahko treh različnih tipov, in sicer informacijska kartica, kartica z namigom ali kartica s povezavo do pametne aplikacije [17]. Če storitev CDS potrebuje dodatne informacije o pacientu, pošlje zahtevo na strežnik FHIR (na sliki 2.2 puščica, označena s številko 2).

2.3 Klasifikacije zdravil in bolezenskih stanj

To, da lahko v bazah podatkov iščemo zdravila in bolezenska stanja po šifrah, nam omogočajo razne klasifikacije.

[2]Swagger je odprtokodno orodje za preprosto razvijanje, dokumentiranje in testiranje APIjev.



Slika 2.2: Struktura in potek delovanja po zdravnikovi aktivnosti.

Slovenska baza zdravil, ki je dostopna na `http://www.cbz.si/cbz/bazazdr2.nsf`, uporablja anatomsko-terapevtsko-kemični klasifikacijski sistem. Anatomsko-terapevtsko-kemični (ATC) klasifikacijski sistem je sistem razvrščanja zdravil, ki ga nadzoruje Sodelujoči center za statistično obdelavo zdravil Svetovne zdravstvene organizacije. Glede na ta sistem dobi vsako zdravilo tako imenovano oznako ATC. Vsako zdravilo (kot končni pripravek) lahko ima le eno oznako ATC [18].
Za klasifikacijo bolezenskih stanj se v Sloveniji uporablja klasifikacija MKB-10-AM, ki je avstralska modifikacija desete revizije Mednarodne klasifikacije bolezni in sorodnih zdravstvenih problemov [19]. Bolezenska stanja se delijo po kategorijah glede na to, kateri del telesa prizadenejo [20].

Slovenska stran, kjer najdemo obe zgoraj omenjeni klasifikaciji in kjer so pri posamezni zdravilni učinkovini napisane še osnovne informacije, indikacije, odmerjanje, kontraindikacije, posebna opozorila, interakcije, neželeni učinki, informacije o prevelikem odmerjanju, farmakodinamika, farmakokinetika, je dostopna na naslovu `https://mediately.co`. Uporabnost strani je v tem, da lahko preko API-ja dostopamo do določenih podatkov in tudi do

šifre zdravilne učinkovine, ki je potem uporabna za iskanje znotraj podatkov strežnika FHIR.

2.4 Programski jezik Go

Programski jezik Go (ali Golang) je ustvarilo podjetje Google leta 2007, odprtokoden je postal leta 2009, leta 2012 pa je bila izdana prva stabilna verzija. Razvijalci Goja so imeli cilj narediti preprost programski jezik, ki bi bil podoben sodobnim dinamično tipiziranim jezikom (npr. Python), ob tem pa bi ohranil hitrost, primerljivo s prevajanimi jeziki, kot sta C in C++. Veliko truda so vložili v razvoj učinkovitega čistilca pomnilnika (*angl. garbage collector*) in v čim krajši čas prevajanja programov [21].

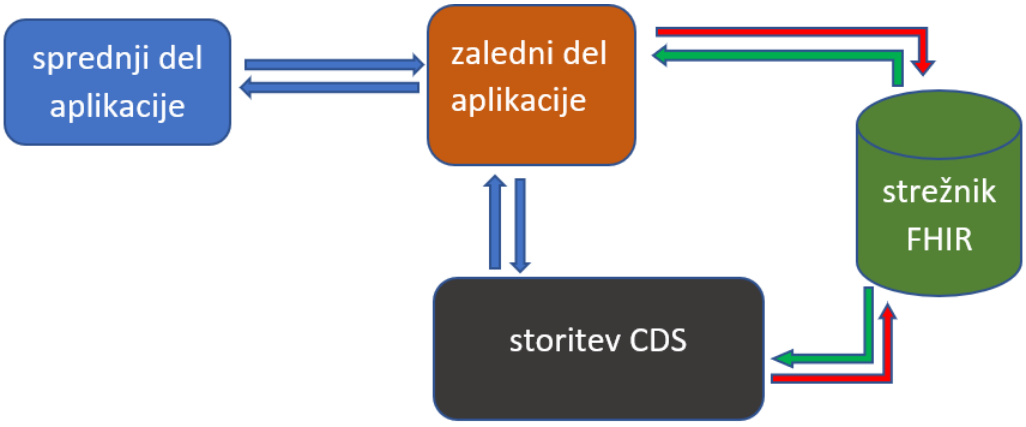
Za samo implementacijo rešitve je izbira programskega jezika dokaj poljubna in ni omejena na točno določen jezik. Pri pregledovanju primerov, ki jih najdemo v pomoč pri implementaciji, se Go ne pojavlja, kar lahko pripisujemo temu, da je jezik dokaj nov in na tem področju neuveljavljen.

Poglavje 3

Implementacija

3.1 Arhitektura

Arhitekturo rešitve (slika 3.1) sestavljajo spletna aplikacija, ki je ločena na sprednji in zaledni del, storitev CDS, kjer je napisana logika za pomoč pri predpisovanju zdravil, ter strežnik FHIR, ki zagotavlja zdravstvene podatke. Modre puščice na sliki 3.1 predstavljajo komunikacijo, rdeče pošiljanje poizvedbe, zelene pa prejemanje odgovora na poizvedbo.



Slika 3.1: Arhitektura rešitve.

3.2 Polnilec strežnika FHIR

Polnilec strežnika FHIR je enostaven program v jeziku Go, ki omogoča polnjenje strežnika FHIR s podatki, ki jih želimo imeti. Podatki o pacientih na strežniku so namreč zelo osnovni in ne vsebujejo primernih testnih primerov za študijo problema, ki ga obravnavamo v tej diplomski nalogi. Poleg tega se nekateri podatki znotraj javno dostopnih brezplačnih strežnikov sčasoma brišejo in najbolj varen način je, da v primeru izbrisa podatkov zaženemo program, ki nam strežnik FHIR znova napolni s podatki, ki jih potrebujemo za naš primer.
Podatki, ki jih polnilec pošilja na strežnik FHIR so programsko nespremenljivi (*angl. hard-ccoded*). Podatki so razdeljeni na posamezne vire FHIR in vsak vir je z ločeno zahtevo poslan na strežnik. Da pri večkratnem zaganjanju ne bi prišlo do kopičenja enakih podatkov, se pred polnjenjem posameznega vira izvede poizvedba na strežnik, če morebiti ta vir že obstaja.

3.3 Opis rešitve

Gre za dogodkovno gnano programiranje, kjer so dogodki zdravnikovi vnosi v spletno aplikacijo. Za vsakega izmed predvidenih dogodkov smo implementirali rokovalnik, ki ustrezno obravnava dogodek in se nanj odzove.

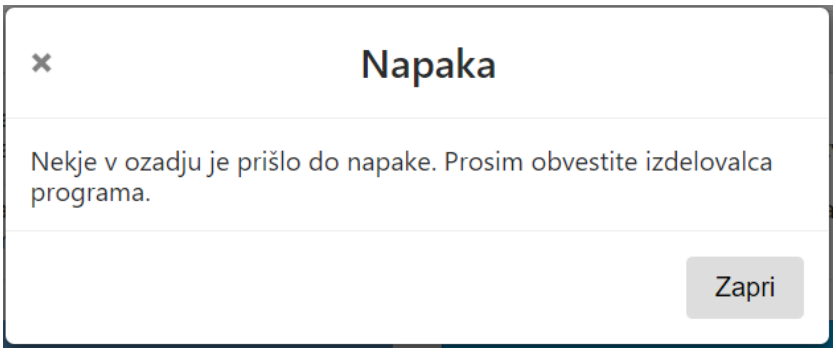
3.3.1 Spletna aplikacija

Sprednji del spletne aplikacije je narejen s pomočjo jezikov HTML5, CSS in JavaScript. Gre za enostaven prikaz, kako bi izgledal vmesnik, kjer ima zdravnik vpogled v osnovne informacije svojih pacientov ter jim predpisuje zdravila. S sprednjim delom aplikacije se nismo poglobljeno ukvarjali, ampak nam je bilo bolj pomembno, da se da pošiljati zahteve zalednemu delu ter da lahko prikazujemo odgovore, ki jih dobimo.
Zaledni del spletne aplikacije je narejen s pomočjo programskega jezika Go. Implementacijo smo začeli z definiranjem vmesnika, nato pa smo s pomočjo

orodja Swagger generirali funkcije, metode in konstrukte, ki so predstavljali temelj za nadaljnje delo. Predvideli smo, da sprednji del aplikacije komunicira z zalednim, ko želimo:

- prikazati vse paciente zdravnika,
- preveriti predpis določene zdravilne učinkovine s kontraindikacijami,
- predpisati določeno zdravilno učinkovino,
- preveriti bolezensko stanje.

Zaledni del pošilja zahteve strežniku FHIR in komunicira s storitvijo CDS, da lahko zagotovi odgovor sprednjemu delu, kot ga želimo. V primeru napake zdravnik dobi obvestilo (slika 3.2).



Slika 3.2: Obvestilo zdravniku o napaki.

Zahteva na strežnik FHIR dobi kot odgovor celoten vir FHIR, zato v zalednem delu te informacije zreduciramo in sprednjemu delu pošljemo le podatke, ki jih bo potreboval za prikaz.

Pri uporabi spletne aplikacije ima zdravnik na prvi strani seznam vseh svojih pacientov (slika 3.3). Za prikaz poskrbi poizvedba zalednega dela na strežnik FHIR, ki išče po viru pacient in ima kot iskalni parameter FHIR ID zdravnika:
GET http://hapi.fhir.org/baseDstu3/Patient?general-practitioner=3852668
V tem primeru je *http://hapi.fhir.org/baseDstu3/* osnovni URL za dostop

(*angl. endpoint*), z besedo *Patient* povemo, kateri vir FHIR želimo nasloviti, iskalni parameter služi temu, da izmed vseh pacientov dobimo kot odgovor na poizvedbo le tiste, ki imajo splošnega zdravnika s šifro *3852668*, *GET* pa je metoda HTTP, ki jo uporabljamo.

Ime	Priimek	Spol	Starost
Andrej	Rus	male	18
Stanka	Napolitanka	female	52
Boris	Novak	male	72

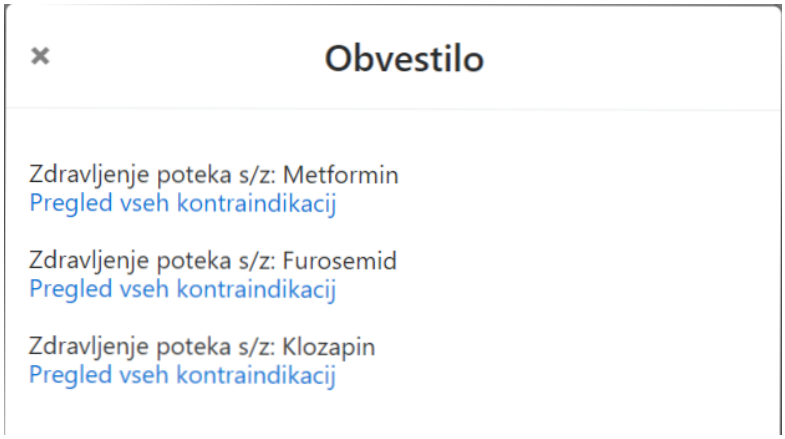
Slika 3.3: Prikaz vseh zdravnikovih pacientov.

Z izbiro posameznika se zdravniku odprejo njegove osnovne informacije, ki jih dobimo s klicem na strežnik FHIR, kjer kot iskalni parameter uporabimo FHIR ID pacienta.

Slika 3.4: Zavihek vpis stanja.

Če želi zdravnik vnesti novo bolezensko stanje pacienta, to naredi v zavihku Vpis stanja (slika 3.4). Rezultat preverjanja stanja bodo izpisana trenutno aktivna zdravljenja oziroma terapije ter povezave do strani, kjer so na-

vedene kontraindikacije za vsakega od njih (slika 3.5). Povezave do prikazanih kontraindikacij bodo zdravnika popeljale na stran `https://mediately.co`, ki jo razvijajo slovenski razvijalci in temelji na slovenski centralni bazi zdravil.



Slika 3.5: Obevstilo po vpisu stanja.

Slika 3.6: Preverjanje predpisa.

Zavihek Predpis zdravila (slika 3.6) omogoča pomoč zdravniku pri predpisovanju novih zdravil. Ko želi predpisati novo zdravilo, mora vnesti stanje, ki ga želi zdraviti, ter zdravilno učinkovino (npr. klozapin). Klasifikacije ATC zdravniku ni treba poznati na pamet, saj se ob pritisku na gumb Preveri predpis (slika 3.6) ta izpolni s pomočjo podatkov na strani Mediately.

Ko zdravnik opravi preverjanje, lahko zdravilo tudi predpiše (ne glede na to, ali so bile zaznane kontraindikacije ali ne) s pritiskom na gumb Predpiši zdravilo (slika 3.6 spodaj desno).

3.3.2 Storitev CDS

V podpoglavju 2.2 smo zapisali, da imamo dve vrsti sistemov za pomoč pri odločanju v zdravstvu. Pri implementaciji naše rešitve se bomo osredotočili na implementacijo sistema, ki temelji na predhodnem znanju. Tak sistem je sestavljen iz treh enot:

- **baza znanja** – kontraindikacije, ki jih je priskrbel strokovnjak na področju zdravstva;
- **logika za odločanje** – osnovana na primerjanju pacientovih stanj in zdravstvenih oskrb z informacijami iz baze znanja;
- **mehanizem za komunikacijo** – API, osnovan na specifikaciji CDS Hooks.

Zdravnik dobi obvestilo v dveh primerih, in sicer ko želi pacientu na novo predpisati zdravilo ali ko vpiše novo stanje v spletno aplikacijo. V rešitvi smo se osredotočili na predpisovanje ene zdravilne snovi – klozapin – a je bil vseeno eden izmed ciljev ta, da je rešitev preprosto razširljiva na več zdravilnih učinkovin.

Kontraindikacije zdravilne učinkovine smo zapisali v formatu TOML (*angl. Tom's Obvious, Minimal Language*), ki je označevalni jezik in omogoča preprosto dekodiranje in potem branje ter iskanje (slika 3.7). Vse kontraindikacije posamezne snovi so zapisane v strukturi pod zdravilno učinkovino, kar

```
[zdravila]
    [zdravila.Klozapin]
        [[zdravila.Klozapin.kontraindikacije]]
            ime = "Epilepsija"
            koda = "G40"

        [[zdravila.Klozapin.kontraindikacije]]
            ime = "Paralitični ileus"
            koda = "K56"
```

Slika 3.7: Primer zapisa v TOML formatu.

nam omogoča, da hitro dostopamo do njih. Pri vsaki smo zapisali ime kontraindikacije in šifro. Ime je namenjeno temu, da je dokument bolj pregleden in človeku razumljiv na prvi pogled, šifra pa omogoča iskanje pri preverjanju kontraindikacij znotraj storitve CDS.

Pri preverjanju zdravilne učinkovine storitev CDS dobi informacijo, katera je zdravilna učinkovina, ki jo preverjamo, ter FHIR ID pacienta, ki mu je bila učinkovina predpisana. Na podlagi tega storitev CDS pošlje zahtevo po stanjih in zdravstvenih oskrbah pacienta na strežnik FHIR. Pridobljene podatke primerja s podatki o kontraindikacijah predpisane zdravilne učinkovine. Če je kontraindikacija najdena, bo zdravnik to izvedel preko obvestila, ki se mu bo prikazalo znotraj spletne aplikacije.

3.4 Ovrednotenje rešitve

Ko na klinični poti pride do tega, da mora zdravnik predpisati pacientu zdravilo, si lahko pomaga s programom, ki smo ga naredili. Celotna rešitev torej služi svojemu namenu, zato bi lahko rekli, da smo bili pri reševanju uspešni. Odločili smo se, da bomo storitev CDS pisali ločeno od zalednega dela spletne aplikacije, kar bi bila tudi ena od možnih arhitektur. To omogoča, da lahko storitev uporablja več različnih spletnih aplikacij, ki so si med seboj različne in se razvijajo neodvisno od storitve.

Posamezni deli, ki smo jih implementirali, med seboj dobro komunicirajo

in ne prihaja do neželenih napak. Implementacija rešitve je bila postavljena na tako raven, kot smo jo načrtovali na začetku, a dopušča še veliko nadgradenj (poglavje 4). V realnem primeru bi nam lahko program pri predpisu klozapina dejansko pomagal ter s tem rešil življenje, kar se nanaša na začetno motivacijo, ki smo jo imeli za delo.

3.4.1 Dodajanje nove zdravilne učinkovine

Pri implementaciji smo stremeli k temu, da bi bilo dodajanje nove zdravilne učinkovine v sistem preprosto, kar nam je uspelo. Naknadno smo dodali še zdravilno učinkovino furosemid, kar ni predstavljalo povečanja števila vrstic programske kode. Potrebno je bilo le dodati nove kontraindikacije v bazo znanja, kar pomeni, da smo rešitev zastavili dovolj splošno.

Poglavje 4

Zaključek

Osnovni cilj, da bi napisali delujoč program, ki bi nudil zdravniku pomoč pri predpisovanju zdravil (pri omejitvi na eno zdravilno učinkovino) na klinični poti, je bil dosežen. Naš program nudi pomoč pri predpisovanju klozapina. Prav tako je bil dosežen cilj, da bomo pisali rešitev na način, da bo preprosto razširljiva na več zdravilnih učinkovin, kar smo potrdili s tem, da smo dodali še eno zdravilno učinkovino in pri tem ni bilo potrebe po dodajanju novih vrstic kode.

4.1 Možnosti za izboljšavo

4.1.1 Internacionalizacija in lokalizacija

Internacionalizacija (pogosto v obliki kratice i18n) in lokalizacija (pogosto v obliki kratice l10n) pomenita prilagajanje programske opreme na različne jezike, regionalne razlike in tehnične potrebe območij [22].
To, da spletna aplikacija ni prilagojena na različne jezike, se dobro vidi v pogledu, ko zdravnik vidi vse svoje paciente (slika 3.3). Vidimo, da so pri spolu uporabljene angleške besede (*male* in *female*), ker gre za podatke, ki jih pridobimo s strežnika FHIR in tam standard omejuje uporabljene besede v polju spol na angleški jezik.
Pri programiranju rešitve, ki jo bo uporabnik (pacient ali zdravnik) upora-

bljal, je treba poskrbeti, da ne pride do težav z uporabo različnih jezikov, ter zagotoviti, da bomo zaledni del ter storitev CDS programirali neodvisno od jezikov. Pri implementaciji smo na strani storitve CDS uporabljali angleščino, saj sta standard FHIR in specifikacija CDS Hooks v angleškem jeziku, medtem ko smo pri spletni aplikaciji uporabljali slovenščino.

Programski jezik Go ponuja knjižnico, ki odgovarja na potrebe internacionalizacije ter lokalizacije in se imenuje text. Gre za skupek paketov, ki nam lahko pomagajo pri različnih kodiranjih znakov, preoblikovanjih besedila in obdelavi besedil glede na različne jezike in jezikovne razlike [23].

4.1.2 Baza kontraindikacij

Kontraindikacije so bile v okviru diplomske naloge napisane v formatu TOML, od koder jih program prebere in ugotovi, če obstajajo kontraindikacije s predpisano zdravilno učinkovino. Če bi želeli narediti bazo kontraindikacij vseh možnih zdravilnih učinkovin, bi bil bolj primeren drugačen način hrambe, ki omogoča shranjevanje in hitro iskanje po veliki količini podatkov. Pri naši implementaciji smo uporabili spletno aplikacijo Mediately, ki nudi informacije o kontraindikacijah za vsako zdravilno učinkovino, a so te podane v tekstovni obliki znotraj spletne strani in brez šifer.

Če bi želeli program dvigniti na višjo raven, bi bilo treba našo storitev CDS povezati z eno od že obstoječih baz kontraindikacij.

4.1.3 Varnost

Če bi želeli zagotoviti tajnost podatkov, bi morali implementirati varnostni sistem (opisano v poglavju 2.1.3).

Literatura

[1] Martin A Makary and Michael Daniel. Medical error - the third leading cause of death in the us. *BMJ*, 353, 2016.

[2] Richard N. Keers. Causes of Medication Administration Errors in Hospitals: a Systematic Review of Quantitative and Qualitative Evidence. *Drug Safety*, 36, 2013.

[3] ZRC SAZU. *Farmacevtski terminološki slovar*, 2011.

[4] Breda Hajnrih. *Priročnik za oblikovanje kliničnih poti*. Ministrstvo za zdravje, 2009.

[5] Univerzitetna psihiatrična klinika Ljubljana. *Klinična pot - uporaba specifičnih zdravil (valproat, litij, karbamazepin, klozapin)*, 3.0 edition, 2016. Interno.

[6] Mate Beštek and Andrej Brodnik. *Interoperability and mHealth – Precondition for Successful eCare*, pages 345–374. Springer International Publishing, Cham, 2015.

[7] HL7.org. Introducing HL7 FHIR. Dosegljivo: `http://hl7.org/fhir/summary.html`. [Dostopano 31. 5. 2018].

[8] HL7.org. FHIR for Executives – Whitepaper. Dosegljivo: `http://www.ringholm.com/persist/FHIR_for_executives.pdf`, 2015. [Dostopano 25. 8. 2018].

[9] Roy Thomas Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

[10] Leonard Richardson and Sam Ruby. *RESTful Web Services*. O'ReillyMedia, 2007.

[11] HL7.org. FHIR Security . Dosegljivo: `http://hl7.org/fhir/security.html`. [Dostopano 4. 9. 2018].

[12] T. Dierks and E. Rescorla. *The Transport Layer Security (TLS) Protocol, Version 1.2*. RTFM, Inc., 2008.

[13] D. Hardt. The oauth 2.0 authorization framework. RFC 6749, RFC Editor, October 2012. `http://www.rfc-editor.org/rfc/rfc6749.txt`.

[14] National Institute of Standards and Technology. *SP 800-162, Guide to Attribute Based Access Control (ABAC) Definition and Considerations*, 2014.

[15] Eta S. Berner. *Clinical Decision Support Systems*. Springer, 2016.

[16] Vivek Gupta, Ethan Jackson, Shaz Qadeer, and Sriram Rajamani. P: Safe asynchronous event-driven programming. Technical report, November 2012.

[17] HL7.org. CDS Hooks Overview. Dosegljivo: `https://cds-hooks.org/`. [Dostopano 4. 6. 2018].

[18] WHO Collaborating Centre for Drug Statistics Methodology. ATC/DDD Methodology History. Dosegljivo: `https://www.whocc.no/atc_ddd_methodology/history/`, 2018. [Dostopano 4. 9. 2018].

[19] Nacionalni inštitut za javno zdravje. MKB-10-AM, verzija 6. Dosegljivo: `http://www.nijz.si/sl/podatki/mkb-10-am-verzija-6`, 2017. [Dostopano 8. 9. 2018].

[20] Svetovna zdravstvena organizacija. *Mednarodna klasifikacija bolezni in sorodnih zdravstvenih problemov za statistične namene*, 2008.

[21] Google. The go programming language - frequently asked questions. Dosegljivo: `https://golang.org/doc/faq`. [Dostopano 4. 9. 2018].

[22] Alex M. Andrew. Software without frontiers: A multi-platform, multi-cultural, multi-nation approach. *Robotica*, 16(1):119–121, January 1998.

[23] Google. Dokumentacija paketa text. Dosegljivo: `https://godoc.org/golang.org/x/text`. [Dostopano 4. 9. 2018].