Mobilna Android aplikacija za potporu pri izboru prikladnog lijeka

Bilić, David

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:392022

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-26**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni preddiplomski studij Računarstvo

MOBILNA ANDROID APLIKACIJA ZA POTPORU PRI IZBORU PRIKLADNOG LIJEKA

Završni rad

David Bilić

SADRŽAJ

1. UVOD ... 1
1.1. Zadatak završnog rada ... 2
2. PROBLEM POVIŠENOG KRVNOG TLAKA I ODABIR LIJEKOVA ... 3
2.1. Osnovni pojmovi vezani za korištenje lijeka ... 3
2.2. Arterijska hipertenzija ... 3
2.3. Ukupni kardiovaskularni rizik i čimbenici rizika ... 4
2.4. Model terapijskog pristupa primijenjen u mobilnoj aplikaciji ... 6
2.4.1. Nefarmakološka terapija ... 8
2.4.2. Farmakološka terapija ... 8
2.5. Slični pristupi i rješenja ... 12
2.6. Idejno rješenje aplikacije ... 14
3. OSNOVE PLATFORME FIREBASE I MODEL BAZE PODATAKA ... 16
3.1. Osnovne informacije vezane uz platformu Firebase ... 16
3.2. Firebase Authentication ... 16
3.3. Firebase Realtime Database ... 17
3.4. Model baze podataka ... 18
4. PROGRAMSKO RJEŠENJE ANDROID MOBILNE APLIKACIJE ... 20
4.1. Android Studio ... 20
4.1.1. Manifest ... 21
4.1.2. Resursi ... 21
4.1.3. Izgled i kontrole korisničkog sučelja ... 22
4.1.4. Aktivnosti ... 23
4.2. Izrada aktivnosti za prijavu i registriranje korisnika ... 25
4.3. Izrada upitnika ... 28
4.4. Izrada aktivnosti za unos nuspojava ... 44
5. ISPITIVANJE RADA I ANALIZA REZULTATA APLIKACIJE ... 45
5.1. Ispitivanje rada aplikacije ... 45
5.2. Analiza rezultata ... 45
5.3. Usporedba rezultata ispitivannja ... 48
6. ZAKLJUČAK ... 49
7. LITERATURA ... 50
ŽIVOTOPIS ... 52
SAŽETAK ... 53
ABSTRACT ... 54
PRILOZI (NA DVD-u) ... 55

1. UVOD

Cilj ovog rada je izrada mobilne Android aplikacije s bazom podataka koja sadrži u sebi nekoliko lijekova, a ima namjeru što vjernije liječniku pružiti potporu za izbor ispravne terapije liječenja, te najprikladnijih lijekova za početak liječenja.

Prije svega, potrebno je definirati probleme čiju će se teorijsku podlogu obraditi u glavnom dijelu rada i na temelju kojih će se objasniti postupak prepisivanja lijeka. Danas su bolesti srca i krvnih žila, poput ateroskleroze, angine pectoris i sličnih, glavni uzrok smanjene kvalitete života ili čak prerane smrtnosti stanovništva. Upravo zbog toga, za potrebe ove aplikacije je odabrana ova česta skupina bolesti, a kao česta poteškoća koja izaziva ili pogoršava kardiovaskularne bolesti izabran je povišeni krvni tlak u arterijama, odnosno - arterijska hipertenzija. Važno je napomenuti da ovaj rad ne može obraditi sve slučajeve hipertenzije , te da je napravljen u svrhu proširenja osobnog znanja o razvoju aplikacija, a teorijska podloga, slike, te korištene tablice iz literature u idućem poglavlju su prilagođeni potrebama aplikacije i nekada nisu dovoljni za najtočnije pružanje potpore za izbor lijeka u svim slučajevima već za aproksimaciju.

Mobilna Android aplikacija koja će biti ostvarena u ovom radu, bit će programski implementirana u programskom jeziku Java, dok će baza podataka za lijekove i korisnike biti ostvarena pomoću Google-ove platforme Firebase. Za izradu praktičnog dijela ovog rada su korištena predznanja iz niza predmeta iz programske skupine predmeta, a posebno je bilo bitno proučiti rad s platformom Firebase, te proširiti znanje iz razvoja mobilnih Android aplikacija.

U poglavlju 2 dana je teorijska podloga o krvnom tlaku, arterijskoj hipertenziji i ukupnom kardiovaskularnom riziku. U tom poglavlju će biti objašnjen postupak odabira prikladnih početnih lijekova za liječenje arterijske hipertenzije, koji ovisi o nekoliko čimbenika, ako je lijek uopće potreban. U poglavlju 3 slijedi upoznavanje s uslugama Google Firebase i to - Authentication i Realtime Database koje su korištene u radu, te objašnjenje modela baze podataka potrebnog za ispravan rad aplikacije. U poglavlju 4, objašnjeni su osnovni pojmovi bitni za rad u razvojnom okruženju Android Studio, te programiranje u programskom jeziku Java. U poglavlju 5 je obavljeno ispitivanje ispravnosti rada aplikacije i prikazana analiza rezultata.

1.1. Zadatak završnog rada

U teorijskom dijelu rada treba objasniti postupak odabira lijeka za dijagnosticiranu bolest, te razraditi model mobilne aplikacije s bazom podataka koja omogućuje registriranje i prijavu pacijenta , te utvrđivanje i izbor odgovarajućeg ponuđenog lijeka s obzirom na kontraindikacije, međudjelovanja i upozorenja. Također, aplikacija treba generirati osnovne preporuke/upozorenja za uzimanje lijeka, te omogućiti prikaz, potvrđivanje postojećih i dodavanje novih nuspojava pacijenta. Mobilnu Android aplikaciju treba programski ostvariti, ispitati na odgovarajućem skupu ulaznih podataka i prikladno analizirati.

2. PROBLEM POVIŠENOG KRVNOG TLAKA I ODABIR LIJEKOVA

U ovom poglavlju bit će objašnjena teorijska podloga o lijekovima, arterijskoj hipertenziji, terapijskom pristupu u određenim slučajevima, te ostalim informacija potrebnim za izradu i ispravan rad aplikacije.

2.1. Osnovni pojmovi vezani za korištenje lijeka

Danas za gotovo svaki zdravstveni problem postoji određena skupina lijekova koja ga može razriješiti ili olakšati. Pri odabiru između više sličnih, prikladnih lijekova, potrebno je poznavati uputstava lijeka. Točnije, potrebno je saznati koje su njegove indikacije, kontraindikacije, kakva su njegova međudjelovanja s drugim lijekovima, te koja upozorenja korisnik treba slijediti. Prema [1], indikacija lijeka predstavlja stanja i bolesti u kojima se lijek treba preferirati, dok kontraindikacija predstavlja stanje u kojemu lijek nije dozvoljeno koristiti. Također, pažnju treba obratiti i na nuspojave koje se prema [1] definiraju kao sve štetne reakcije uzrokovane uzimanjem lijeka bez obzira na pridržavanje propisanog doziranja i načina primjene lijeka. Nuspojave mogu nastati odmah ili nakon nekog vremena.

2.2. Arterijska hipertenzija

Da bi se pojasnilo što je arterijska hipertenzija, potrebno je ukratko pomoću literature pojasniti što predstavlja krvni tlak, te kakve vrste krvnog tlaka postoje. Prema [2], krvni tlak predstavlja pritisak kojeg krv stvara na stijenkama krvnih žila. Krv upravo iz tog razloga i protječe krvnim žilama, jer se nalazi pod utjecajem određenog tlaka. Taj tlak stvara srce svojim radom, ono zapravo djeluje kao pumpa. Pri istiskivanju krvi iz srca (sistola), tlak se povisuje, a pri vraćanju krvi u srce (dijastola), tlak se snižava. Stoga postoje dvije vrijednosti krvnog tlaka - sistolički i dijastolički.

Dakle, sistolički tlak je onaj tlak koji nastaje u arterijama pri stezanju srca, a dijastolički tlak nastaje pri opuštanju srca. Dijastolički tlak predstavlja najniži tlak kojemu se arterije izlažu. Bitno je naglasiti da se krvni tlak izražava pomoću milimetara stupca žive ili mmHg: Ako krvni tlak žile iznosi 100 mmHg, to znači da bi ta sila bila dovoljna podignuti stupac Hg na razinu od 100 mm [3, str. 16]. Širina krvnih žila prema [3, str. 16] također utječe na visinu krvnog tlaka: ako je krvna žila široka, većina krvi prolazi središtem žile, umjesto uz njezinu stijenku, pa je otpor protjecanju malen.

Normalna vrijednost krvnog tlaka prema [2], koja omogućuje normalan život bez oštećivanja krvožilnog sustava iznosi 120/80 mmHg. Tijekom starenja, krvni tlak se polako povisuje, ali on

svakako ne bi smio prelaziti vrijednost od 140/90 mmHg, a ta vrijednost predstavlja gornju granicu normalnog krvnog tlaka.

Sada se može definirati arterijska hipertenzija. Prema [3, str. 13], kada su vrijednosti arterijskog tlaka toliko visoke da sistolički prelazi 140 mmHg i/ili dijastolički prelazi 90mmHg, radi se o hipertenziji, odnosno povišenom arterijskom tlaku. Tablica 2.1 prikazuje klasifikacije arterijskog tlaka po literaturi [4, str. 9], a ova tablica će biti korištena pri unosu izmjerenog tlaka u aplikaciju. Za potrebe ove aplikacije neće korištene sve klasifikacije, točnije izolirana sistolička, radi kompleksnosti rješenja.

Tablica 2.1 *Definicija i klasifikacija arterijskog tlaka (mmHg)* [4, str. 9]

KATEGORIJA	SISTOLIČKI	DIJASTOLIČKI
Optimalan	< 120	< 80
Normalan	120-129	80-84
Visoko normalan	130-139	85-89
Hipertenzija		
Stupanj 1 (Blaga hipertenzija)	140-159	90-99
Stupanj 2 (Umjerena hipertenzija)	160-179	100-109
Stupanj 3 (Teška hipertenzija)	>=180	>=110
Izolirana sistolička	>=140	< 90

2.3. Ukupni kardiovaskularni rizik i čimbenici rizika

Poznavajući klasifikacije izmjerenog arterijskog tlaka treba biti svjestan da odabir prikladnog lijeka ne ovisi samo o vrijednostima izmjerenog arterijskog tlaka. U obzir je također potrebno uzeti životne navike pacijenta, njegovu starost i ostale bolesti od kojih pacijent boluje koje spadaju u rizične čimbenike i pomoću njih se računa ukupni kardiovaskularni rizik. Prema [3, str. 13], kako bi se odredila najtočnija granica za definiranje hipertenzije, potrebno je poznavati visinu arterijskog tlaka i ukupni kardiovaskularni rizik. Kardiovaskularni rizik uvelike ovisi o izboru prikladnog lijeka. Prema [3, str. 13], ukupni kardiovaskularni rizik se računa pomoću vrijednosti krvnih tlakova, te prisutnosti nekih čimbenika rizika i bolesti koji će biti navedeni. Treba primjetiti da ne utječu svi čimbenici rizika i bolesti jednako na izračun ukupnog kardiovaskularni rizik.

Kardiovaskularni rizik se može podijeliti u četiri kategorije:

- Nizak dodatni rizik
- Umjereni dodatni rizik
- Visok dodatni rizik
- Vrlo visok dodatni rizik

Prema [4, str. 10], pojam rizik se odnosi na 10-godišnji rizik od nastanka kardiovaskularnog događaja, dok pojam 'dodatni' govori da je rizik iznad prosječnog.

Tablica 2.2 prikazuje stratifikaciju ukupnog kardiovaskularnog rizika po uzoru na [4, str. 10]. Važno je uzeti u obzir da je ovo starija tablica, te da su rizici danas čak i niži od ovih navedenih u ovom modelu, a potpora za odabir prikladne terapije će biti pojašnjena u idućem poglavlju.

Tablica 2.2 *Stratifikacija kardiovaskularnog rizika u četiri kategorije* [4, str. 10]

	Arterijski tlak (mmHg)				
Rizični čimbenici, SOO ili bolest	Normalan	Visoko normalan	Stupanj 1. AH	Stupanj 2. AH	Stupanj 3. AH
0 rizičnih čimbenika	Prosječan rizik	Prosječan rizik	Nizak dodatni rizik	Umjereni dodatni rizik	Visok dodatni rizik
1-2 rizičnih čimbenika	Nizak dodatni rizik	Nizak dodatni rizik	Umjereni dodatni rizik	Umjereni dodatni rizik	Vrlo visok dodatni rizik
>= 3 čimbenika, MS, SOO ili ŠB	Umjereni dodatni rizik	Visok dodatni rizik	Visok dodatni rizik	Visok dodatni rizik	Vrlo visok dodatni rizik
Razvijena KV ili bubrežna bolest	Vrlo visok dodatni rizik	Vrlo visok dodatni rizik	Vrlo visok dodatni rizik	Vrlo visok dodatni rizik	Vrlo visok dodatni rizik

AH – arterijska hipertenzija, MS – metabolički sindrom, SOO – subkliničko oštećenje organa, ŠB – šećerna bolest, KV – kardiovaskularni. Isprekidana crta prikazuje varijabilnost hipertenzije s obzirom ukupni KV rizik.

Tablica 2.3 konkretnije navodi neke od čimbenika rizika korištenih u tablici 2.2 koji utječu na prognozu kardiovaskularnog rizika . Važno je napomenuti, da su u tablici 2.3 navedeni samo neki čimbenici rizika i bolesti koji će biti korišteni u aplikaciji pri stvaranju upitnika, te da u tablicama iz originalne literature u [3, str. 14], te [4, str. 12] ima puno više navedenih čimbenika i bolesti. Ovdje treba obratiti pozornost na to da se „Visoko normalna“ vrijednost krvnog tlaka također broji kao jedan rizični čimbenik.

Također je potrebno napomenuti da bi aplikacija trebala prepoznavati metabolički sindrom. Prema [5, str. 8438], metabolički sindrom je prisutan postojanjem barem tri od sljedećih rizičnih čimbenika: abdominalna pretilost, povišena glukoza natašte, visoko normalna ili viša klasifikacija krvnog tlaka, te dislipidemija.

Tablica 2.3 *Neki čimbenici rizika koji pomažu pri izračunu ukupnog kardiovaskularnog rizika koji će biti korišteni u aplikaciji.* [3, str. 14], [4, str. 12]

Uobičajeni čimbenici rizika	**Čimbenici rizika koji bolesnika svrstavaju u visoki/vrlo visoki KV rizik**
Arterijski tlak >= 130/85 mmHg	Arterijski tlak >= 180/110 mmHg
Godine (M >=55, Ž >= 65)	Prisutna bubrežna bolest
Pušenje	Postojanje 3 do 5 čimbenika iz uobičajenih rizika
Glukoza natašte povišena (5.6-6.9 mmol/l)	Šećerna bolest
Abdominalna pretilost (opseg struka: M>102cm, Ž>88cm), BMI >= 30kg/m²	Prisutna kardiovaskularna bolest
Dislipidemija (snižen HDL kolesterol i povišeni Trigliceridi)	Oštećenje organa
Obiteljska anamneza prerane KV bolesti (M<55, Ž<65)	

2.4. Model terapijskog pristupa primijenjen u mobilnoj aplikaciji

Nakon upoznavanja s vrijednostima sistoličkog i dijastoličkog tlaka, te s procjenom ukupnog kardiovaskularnog rizika, treba pojasniti ciljeve liječenja i pristupe liječenju. Prema [6] vrijedi da je za hipertoničare ciljna vrijednost arterijskog tlaka ispod 140/90 mmHg, a u slučaju dijabetesa ili bubrežne bolesti ispod 130/80 mmHg. U ovom potpoglavlju će se promotriti pristup terapiji kakav će koristiti aplikacija, a zatim će se pojasniti farmakološki i nefarmakološki pristup terapiji.

Ovisno o stupnju hipertenzije pacijenta, te ukupnoj procjeni kardiovaskularnog rizika, odlučuje se je li pacijentu dovoljna nefarmakološka terapija, te ako nakon nekog vremena nefarmakološka terapija ne postigne željene ciljeve arterijskog tlaka, potrebno je prijeći na farmakološku terapiju. U težim slučajevima je odmah potrebno prijeći na farmakološku terapiju.

S obzirom na male različitosti pristupa liječenju prikazanih u tablicama iz literature [3,str. 23], te [4, str. 33], napravljena je tablica 2.4 koja za potrebe aplikacije i ostvarivanja rješenja aproksimira tablice iz navedenih literatura. Tablica 2.4 će u aplikaciji biti korištena nakon što pacijent unese vrijednosti izmjerenog tlaka, te nakon što mu se na osnovu unosa ponuđenih čimbenika rizika dodjeli određeni stupanj kardiovaskularnog rizika. Pomoću tablice 2.4 će aplikacija utvrđivati je li potrebna farmakološka terapija, te je li potrebna odmah ili nakon određenog perioda promjena životnih navika. Važno je opet napomenuti da su, kao što je već rečeno, neki rizični čimbenici i bolesti iz originalne literature izostavljeni radi pojednostavljivanja rješenja i pri ispitivanju ih se neće uzimati kao ulazne podatke.

Tablica 2.4 *Ovisnost odabira terapije o visini izmjerenog tlaka i ukupnom kardiovaskularnom riziku kakav će koristiti aplikacija* [3,str. 23], [4, str. 33]

Arterijski tlak (mmHg)					
Ostali čimbenici rizika ili bolest	Normalan	Visoko normalan	Stupanj 1. AH	Stupanj 2. AH	Stupanj 3. AH
0 rizičnih čimbenika	Bez intervencije	Bez intervencije	Promjene životnih navika, ukoliko ne dođe do poboljšanja nakon nekoliko **mjeseci**, farmakološka terapija	Promjene životnih navika, ukoliko ne dođe do poboljšanja nakon nekoliko **tjedana**, farmakološka terapija	Promjene životnih navika + farmakološka terapija
1-2 rizična čimbenika	Bez intervencije	Bez intervencije	Promjene životnih navika, ukoliko ne dođe do poboljšanja nakon nekoliko **mjeseci**, farmakološka terapija	Promjene životnih navika, ukoliko ne dođe do poboljšanja nakon nekoliko **tjedana**, farmakološka terapija	Promjene životnih navika + farmakološka terapija
>=3 rizična čimbenika, MS, ŠB ili OO	Promjena životnih navika	Promjena životnih navika	Promjene životnih navika + farmakološka terapija	Promjene životnih navika + farmakološka terapija	Promjene životnih navika + farmakološka terapija
Ustanovljena KV bolest, bubrežna bolest, šećerna bolest + OO/uobičajeni čimbenik rizika	Promjena životnih navika	Promjene životnih navika	Promjene životnih navika + farmakološka terapija	Promjene životnih navika + farmakološka terapija	Promjene životnih navika + farmakološka terapija

AH – arterijska hipertenzija, MS – metabolički sindrom, SOO – subkliničko oštećenje organa, ŠB – šećerna bolest, KV – kardiovaskularni.

U tablici 2.4 bojama su označene kategorije kardiovaskularnog rizika jednako kao i u tablici 2.2, a ovo su neke tvrdnje koje su pojasnile kod kojih osoba koristiti farmakološku terapiju [3, str. 22]:

- Stupanj hipertenzije 2 i 3 uz bilo koji kardiovaskularni rizik nakon nekoliko tjedana promjene životnih navika ili odmah
- Stupanj hipertenzije 1 uz visok kardiovaskularni rizik
- Stupanj hipertenzije 1 uz nizak/umjeren kardiovaskularni rizik ako promjena životnih navika kroz određeno vrijeme nije bila učinkovita
- Nije preporučena farmakološka terapija ako je arterijski tlak visoko normalan

2.4.1. Nefarmakološka terapija

U tablici 2.4 vidljivo je da je u većini slučajeva preporučena promjena životnih navika, a nekada je samo to i dovoljno. Prema [7, str. 476], postoje mnogi čimbenici rizika na koje pacijenti mogu utjecati, a uključuju zdraviju prehranu, fizičku aktivnost, dijabetes, pretilost i pušenje. Nove životne navike su u današnjem svijetu jako podcijenjene, a mogu znatno pripomoći i kod sniženja arterijskog tlaka i kod kontrole ukupnog kardiovaskularnog rizika. Prema [8], liječnici uvijek pokušavaju izbjeći liječenje koje bi pacijentima smetalo u svakodnevnom životu. Prije propisivanja lijekova, poželjno je pokušati s nefarmakološkom terapijom. Nefarmakološka terapija podrazumijeva:

- Prestanak pušenja
- Smanjenje tjelesne mase
- Smanjenje unosa alkohola
- Tjelesna aktivnost
- Smanjenje unosa kuhinjske soli
- Povećanje unosa voća i povrća i smanjenje unosa zasićenih i ukupnog unosa masti

2.4.2. Farmakološka terapija

Tablica 2.4 uz nefarmakološku terapiju, u nekim slučajevima preporučuje i farmakološku. Farmakološka terapija zahtijeva korištenje lijekova. Prema [8], postoji više različitih lijekova koji smanjuju krvni tlak na različite načine. Neki liječnici započinju terapiju s jednim lijekom,a kasnije, ako je potrebno, dodaju drugi. Drugi pak propišu jedan lijek, a u slučaju njegove neučinkovitosti daju drugu vrstu lijeka, dok prethodnu izbacuju.

Potrebno je pronaći prikladne lijekove za početak i održavanje antihipertenzivnog liječenja, preko monoterapije ili u kombinacijama. Ovi lijekovi će biti pohranjeni u bazu podataka:

1. **Diuretici (tiazidi)**
2. **Beta blokatori**
3. **Blokatori kalcijskih kanala**
4. **ACE inhibitori**
5. **Blokatori angiotenzinskih receptora (ARB)**

Svaka od ovih skupina može imati svojstva, prednosti i ograničenja koji pomažu pri donošenju odluke koja skupina će biti najprikladnija za kojeg bolesnika. Neki od bitnijih čimbenika za odabir lijeka su: visina tlaka, visina kardiovaskularnog rizika, te indikacije i kontraindikacije određenih

lijekova u određenim stanjima (poput dijabetesa, bubrežnih bolesti, trudnoće itd.). Cilj ove aplikacije je pomoću navedenih čimbenika preporučiti najprikladnije lijekove za početak liječenja, ako su oni potrebni.

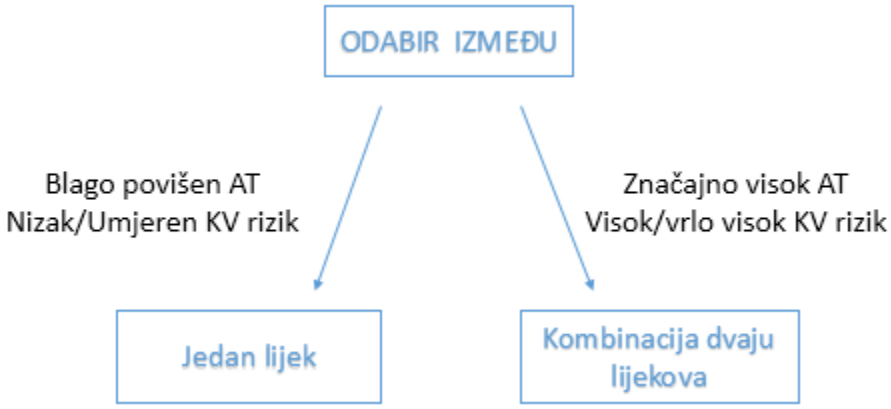
Ovisno o ukupnom broju lijekova korištenih u aplikaciji treba razlikovati:

- Monoterapiju
- Kombiniranu terapiju dvaju lijekova

Prema [3, str. 40] je vidljivo kada otprilike treba pristupiti liječenju s kombiniranom terapijom:

– Monoterapija – blago povišen arterijski tlak/nizak ili umjeren rizik
– Kombinacija dvaju lijekova – visok arterijski tlak/visok ili vrlo visok rizik

S obzirom na to da se ova aplikacija treba baviti podrškom za preporuku početne terapije, dovoljna joj je samo ova parafraza jer se ona ne bavi slučajevima u kojima je potrebno prijeći na drugi lijek ili kombinaciju jer prethodni nije imao dovoljan učinak sniženja tlaka. U skladu s ovim napravljen je maleni dijagram odlučivanja pomoću kojega će aplikacija odrediti prikladan broj lijekova u slučaju potrebe za farmakološkom terapijom. Dijagram prikazan na slici 2.5 je napravljen je po uzoru na dijagram iz [3, str. 40], te je ovo samo dio izvornog dijagrama.



Slika 2.5 *Odabir monoterapije ili kombinirane terapije*

Tablica 2.6 će biti od velike koristi, jer ona prikazuje prednost navedenih antihipertenzivnih lijekova u određenim stanjima koji će biti korišteni u aplikaciji. Važno je napomenuti da su u tablici 2.6 navedena samo neka stanja iz originalnih literatura za potrebe aplikacije. Sva stanja iz tablice 2.6 će se koristiti pri stvaranju upitnika.

Tablica 2.6 *Prednost pojedinih lijekova za ponuđena stanja u aplikaciji* [3, str. 3], [4, str. 41-42], *te* [6]

Diuretici	**Beta blokatori**	**Blokatori kalcijevih kanala**	**ACE inhibitori**	**Blokatori angiotenzinskih receptora (ARB)**
Starija dob	Mlađa dob	Starija dob	Mlađa dob	Mlađa dob
Crna rasa	Angina pectoris	Crna rasa	Asimptomatska ateroskleroza (OO)	Šećerna bolest
Pretilost	Trudnoća	Angina pectoris	Metabolički sindrom	Metabolički sindrom
	Prethodni infarkt miokarda	Periferna arterijska bolest	Bubrežna disfunkcija(OO)/ zatajivanje/proteinurija	Bubrežna disfunkcija(OO)/ zatajivanje/proteinurija
	Glaukom	Trudnoća	Mikroalbuminurija (OO)	Mikroalbuminurija (OO)
	Aneurizma aorte	Asimptomatska ateroskleroza (OO)	Šećerna bolest	Prethodni infarkt miokarda
		Metabolički sindrom	Prethodni infarkt miokarda	HLK(OO)
		HLK(OO)	Periferna arterijska bolest	Stanja u kojima se ne podnosi ACE inhibitor zbog kašlja
			HLK(OO)	

OO – „oštećenje organa“, HLK –„hipertrofija lijeve klijetke“

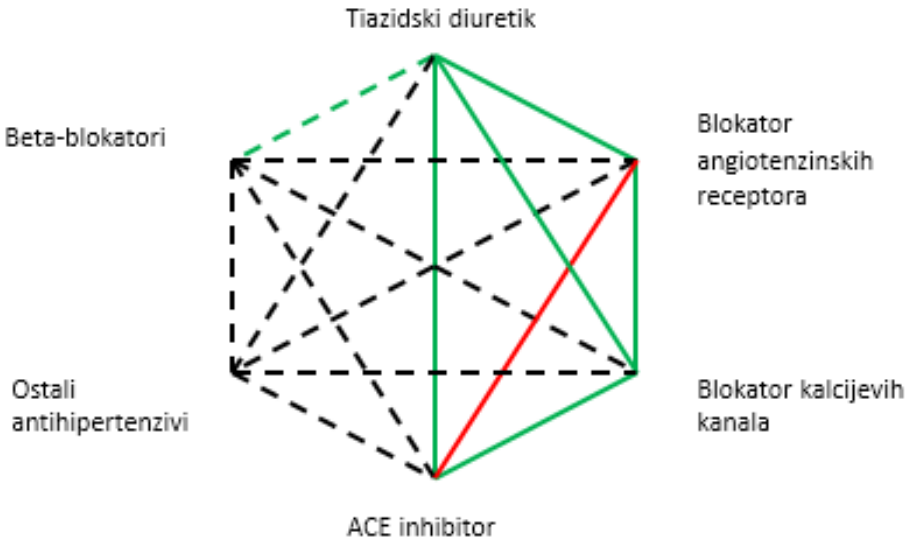
Sada su poznati prioriteti uzimanja određenog antihipertenziva u određenim stanjima, a s obzirom na to da su svi antihipertenzivi dokazano dovoljno dobri za snižavanje tlaka, da bi se omogućila eliminacija lijekova, potrebna je izrada tablice 2.7 koja prikazuje sigurne i moguće kontraindikacije lijekova. Tablica 2.7 Prikazuje neke kontraindikacije iz literature koje će biti korištene pri stvaranju upitnika. Tablica 2.7 je napravljena po uzoru na [3, str. 29].

Tablica 2.7 *Sigurne i moguće kontraindikacije za upotrebu antihipertenziva*, [3, str. 29]

NAZIV LIJEKA	SIGURNE KONTRAINDIKACIJE	MOGUĆE KONTRAINDIKACIJE
Tiazidski diuretici	Ulozi	Metabolički sindrom Nepodnošenje glukoze Trudnoća Hiperkalcemija Hiperkaliemija
Beta blokatori	Astma A-V blok (stupanj 2 ili 3) Ne preporučuju se dijabetičarima	Metabolički sindrom Nepodnošenje glukoze Sportaši i fizički aktivni pacijenti Kronična opstruktivna plućna bolest
Blokatori kalcijevih kanala (dihidropirinski)	A-V blok (stupanj 2 ili 3)	Tahiaritmija Zatajenje srca
ACE inhibitori	Trudnoća Angioneurotski edem Obostrana stenoza bubrežnih arterija Hiperkaliemija	Potencijalne trudnice
Blokatori angiotenzinskih receptora (ARB)	Trudnoća Obostrana stenoza bubrežnih arterija Hiperkaliemija	Potencijalne trudnice

Saznanjem o potrebi za kombinacijama lijekova u težim slučajevima s većim povišenjem krvnog tlaka i visokim ukupnim kardiovaskularnim rizikom, potrebno je istražiti koje kombinacije lijekova su moguće. Tu postoji jedna loša strana, što je eventualno izlaganje bolesnika nepotrebnom lijeku, ali s pozitivne strane: oba lijeka se mogu primjeniti u niskoj dozi, a zbog toga vjerojatnije neće doći do nuspojava u odnosu na potpunu dozu monoterapije [4, str. 43], te početak liječenja kombiniranjem dvaju lijekova ranije postiže ciljne vrijednosti arterijskog tlaka u odnosu na monoterapiju [4, str. 43].

Prema [3, str. 41], vidljivo je da su kombinacije različitih antihipertenziva moguće su ako im je mehanizam djelovanja komplementaran, ako su zajedno efikasniji od pojedinačnoga, ako nema negativnog međudjelovanja i ako postoji manji broj nuspojava u kombinaciji nego u monoterapiji. Primjerice, poznavanjem da ACE inhibitor i Blokator angiotenzinskih receptora (ARB) imaju vrlo sličan utjecaj, saznaje se je da ih ne treba kombinirati. Slika 2.8 prikazuje moguće kombinacije antihipertenziva po uzoru na [9, str. 2193].



Slika 2.8 *Moguće kombinacije antihipertenziva* [9, str. 2193]

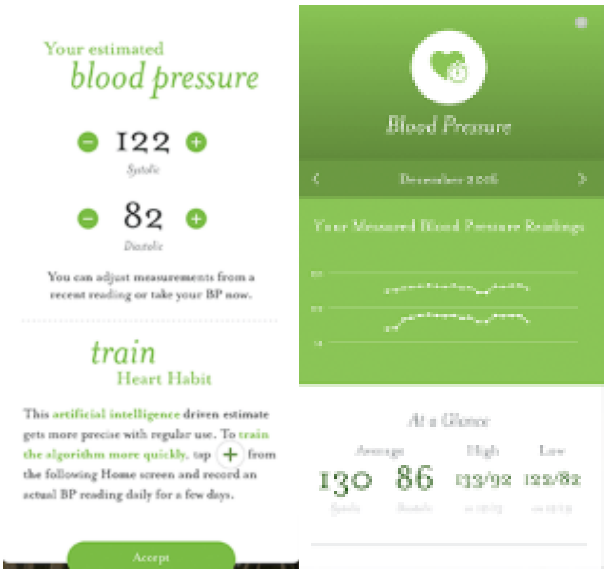
Slijedi pojašnjenje linija korištenih u dijagramu:

- Zelena puna linija – poželjne kombinacije
- Zelena isprekidana linija – korisna kombinacija s ograničenjima : prema [3, str. 41], ne propisivati bolesnicima oboljelim od metaboličkog sindroma ili s rizikom od nastanka dijabetesa
- Crna isprekidana linija – manje istražene kombinacije
- Crvena linija – kombinaciju treba izbjegavati

Nakon prolaska kroz teorijsku podlogu o hipertenziji, kardiovaskularnom riziku i pristupu liječenja, treba prijeći na osnove Firebase-a, platforme pomoću koje je izrađena baza podataka.

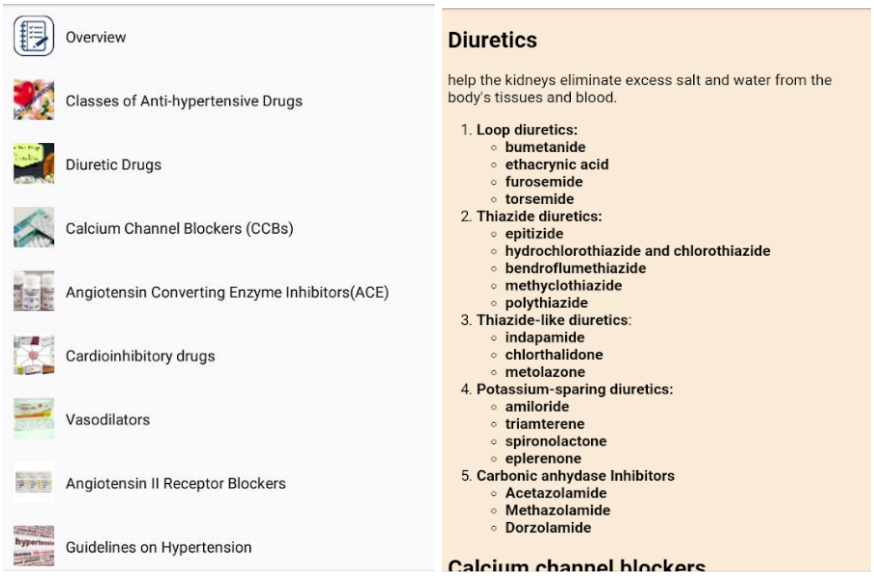
2.5. Slični pristupi i rješenja

Rješenje koje se bavi sličnim problemom je iOS aplikacija pod nazivom Heart Habit. Heart Habit je pametan, lak i kreativan način praćenja i upravljanja vašim rizikom od visokog krvnog tlaka, srčanog udara i moždanog udara [10]. Ova aplikacija je izrađena u suradnji s fizičarima, nutricionistima i farmaceutima.



Slika 2.9 *Heart Habit*

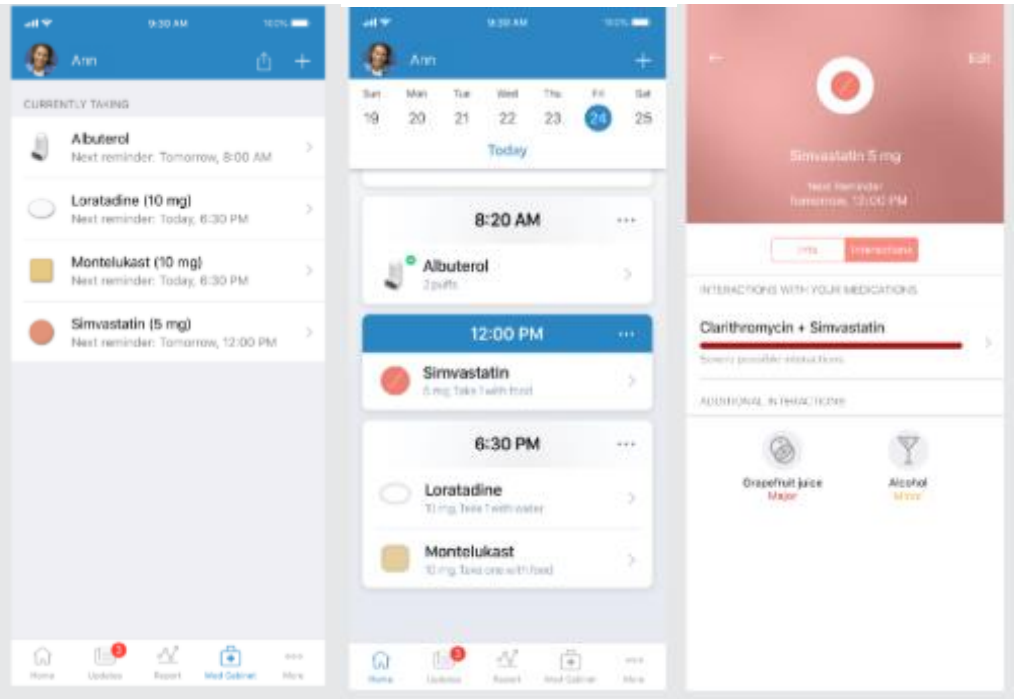
Sljedeća aplikacija se naziva Anti-hypertensive Drugs. Sadrži mehanizme akcija, indikacija, štetnih učinaka i kontraindikacija svih klasa antihipertenziva. Ovdje su i ažurirane smjernice za identificiranje i upravljanje hipertenzijom [11].



Slika 2.10 Anti-hypertensive Drugs [11]

Tu je također i aplikacija pod nazivom Medisafe. Ova aplikacija predstavlja podsjetnik i raspored za uzimanje lijekova. Također omogućuje i prikaz doziranja te način uzimanja lijeka. Prema [12], istraživanje pokazuje da su pacijenti oboljeli od arterijske hipertenzije pokazali veću upornost

prema liječenju koristeći ovu mobilnu aplikaciju, a korištenje ovakve tehnologije bi moglo dovesti do veće kontrole kroničnih bolesti i nižih troškova zdravstvenog sustava.



Slika 2.11 Medisafe [13]

2.6. Idejno rješenje aplikacije

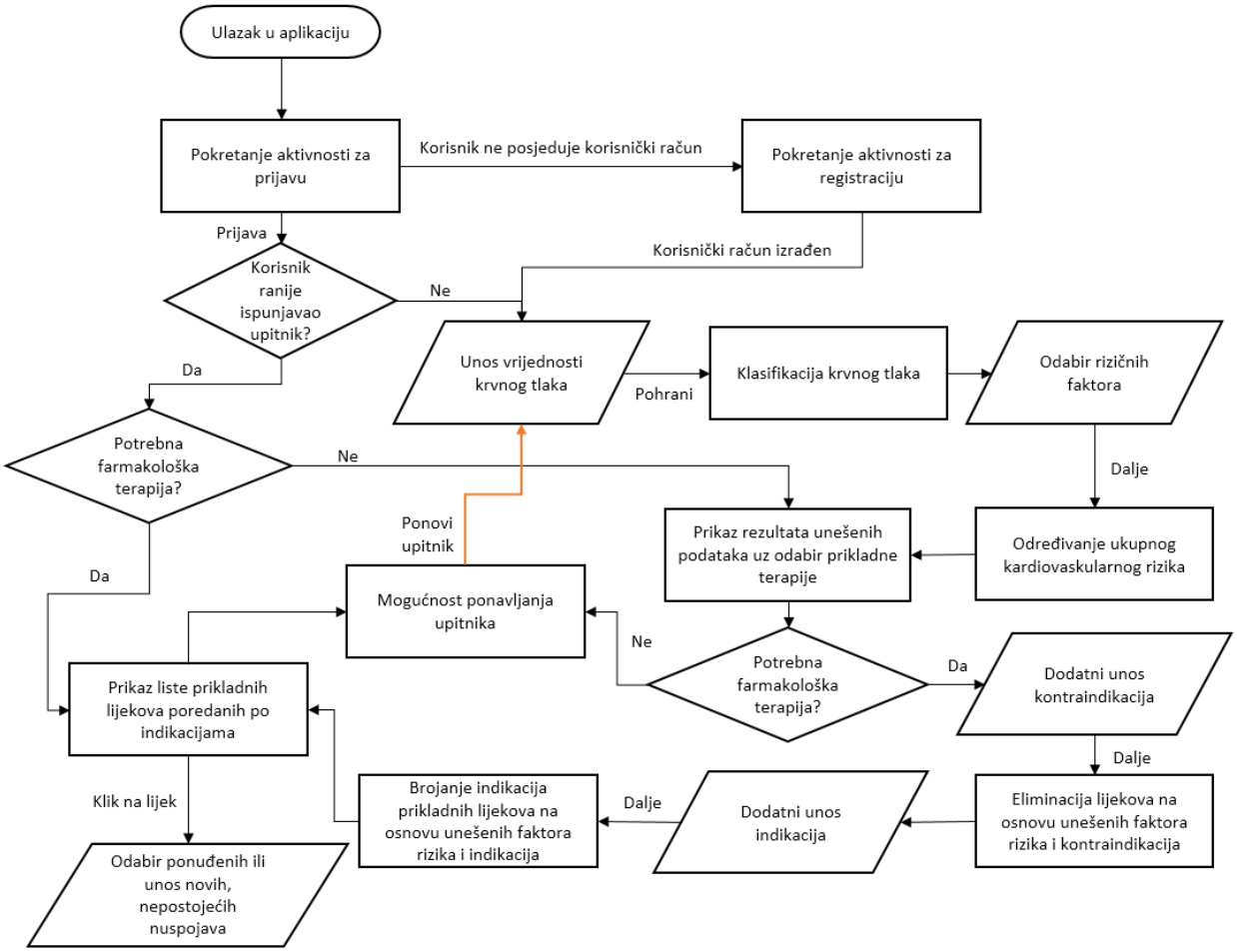
Prije izrade aplikacije je potrebno osmisliti način na koji će aplikacija biti ostvarena uzimajući u obzir sve zahtjeve koje ona treba ispunjavati.

Korisnik prvo mora izraditi korisnički račun ako ga već ne posjeduje. Ako ga posjeduje, kada se prijavi ili ako je ostao prijavljen, odmah ga se prebacuje na jedan od tri ekrana aplikacije ovisno o tome je li već ispunjavao upitnik, te je li mu potrebna farmakološka terapija. Ako upitnik nije ispunjen, korisnik unosi vrijednosti krvnih tlakova i pohranjuje ih, a aplikacija po tablici 2.1 svrstava uneseni krvni tlak u određenu kategoriju. Zatim korisnik odabire čimbenike rizika iz tablice 2.3, te mu se po tablici 2.2 dodjeljuje određena visina ukupnog kardiovaskularnog rizika. Ovdje je važno prepoznati postojanje metaboličkog sindroma, kao i nekih važnijih čimbenika rizika koji također predstavljaju indikacije ili kontraindikacije nekih lijekova.

Slijedi prikaz dosadašnjih rezultata uz preporuku prikladne terapije na osnovu tablice 2.4 i izračun broja preporučenih lijekova po slici 2.5. Ako korisniku nije preporučena farmakološka terapija, upitnik je za njega gotov, on vidi nefarmakološku terapiju i ima samo mogućnost da ponovi upitnik. Ako je korisniku preporučena farmakološka terapija, prikazana mu je i nefarmakološka

terapija, a nakon toga slijedi unos dodatnih kontraindikacija iz tablice 2.7 koje do sada nisu bile ponuđene u čimbenicima rizika. Nakon unosa dodatnih kontraindikacija, aplikacija sve do sada unesene kontraindikacije koristi da bi eliminirala neprikladne lijekove. Slijedi unos dodatnih indikacija iz tablice 2.6, koje također do sada nisu bile ponuđene čimbenicima rizika, a nakon potvrde aplikacija sortira sve prikladne lijekove po broju unesenih indikacija. Nakon eliminacije neprikladnih i sortiranja prikladnih lijekova, otvara se prozor koji korisniku prikazuje sve prikladne lijekove i njihov broj indikacija. Tu je također spomenut broj preporučenih lijekova određenog po slici 2.5 uz upozorenja koja po slici 2.8 govore koje lijekove ne treba kombinirati.

Korisnik ima mogućnost klika na lijek, što će rezultirati otvaranjem prozora za odabir poznatih nuspojava za taj lijek, a može unijeti i neku novu, do sada nepostojeću nuspojavu u bazu podataka. Slika 2.11 prikazuje opisani model aplikacije.



Slika 2.12 *Zamišljeni model aplikacije*

3. OSNOVE PLATFORME FIREBASE I MODEL BAZE PODATAKA

Ovaj završni rad zahtijeva korištenje baze podataka, da bi se omogućila autentikacija korisnika i pohrana njihovih odgovora i rezultata. Za ostvarenje baze podataka je odabrana platforma koja je ostvarena od strane Google-a pod nazivom Firebase. U ovom dijelu završnog rada će se pojasniti platforma Firebase i neke njezine osnovne usluge koje su korištene u aplikaciji, a zatim slijedi prolazak kroz model baze podataka koji ćemo koristiti.

3.1. Osnovne informacije vezane uz platformu Firebase

Firebase je platforma Backend-as-a-Service namijenjena razvoju mobilnih i web aplikacija koja pomaže pri bržem i jednostavnijem smišljanju i podizanju korisničke baze podataka, a čak nudi i analiziranje prometa koje može biti od velike pomoći u slučaju profesionalnog bavljenja razvojem aplikacija. Njegova velika odlika je ta što pri korištenju Firebase-a korisnik ne mora brinuti o upravljanju poslužiteljem niti napisati ijednu liniju koda za upravljanje poslužiteljem.

Platforma Firebase u sebi sadrži brojne usluge koje svojom jednostavnošću uvelike pomažu pri razvijanju što kvalitetnije aplikacije i ostvarenju željenog cilja.

Neke najvažnije od njih su:

- *Google Analytics za Firebase*
- *Firebase Authentication*
- *Realtime Database*
- *Hosting*
- *Storage,* za pohranu slika i određenih dokumenata
- *Crash Reporting*

Google Analytics za Firebase je upravo ono spomenuto što pruža mogućnost uvida u ponašanje korisnika aplikacije: sve od broja posjeta aplikacije, do broja rušenja aplikacije i njezinih prihoda. Za izradu ove aplikacije će biti dovoljni Firebase Authentication i Realtime Database pa će zbog toga biti pojašnjene.

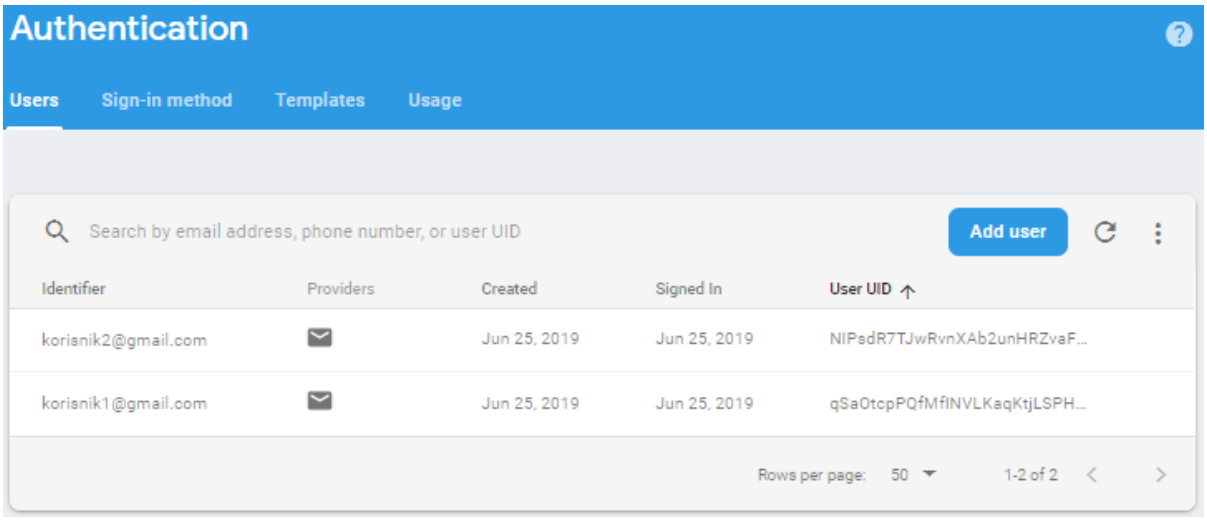
3.2. Firebase Authentication

Većina kvalitetnih aplikacija mora znati kako identificirati korisnika. Poznavanje korisnikovog identiteta omogućuje aplikaciji da na siguran način pohranjuje korisničke podatke na Cloud i da svakom korisniku pruži personaliziran izgled aplikacije. Prema [14], Firebase Authentication

pruža backend usluge, lako razumljiv pribor za razvijanje programske podrške (SDK), te pripremljene biblioteke koje omogućuju registraciju i logiranje korisnika u aplikaciju. Ona omogućuje autentikaciju korisnika putem lozinke, broja mobitela ali i popularnih društvenih mreža, poput Facebook-a, Twitter-a ili Google-a.

Prvo što je potrebno učiniti da bi se korisniku omogućilo korištenje aplikacije jest potražiti od njega osobne informacije. Te informacije mogu sadržavati adresu elektroničke pošte zajedno sa željenom lozinkom ili OAuth token s jedne od popularnih društvenih mreža. Tada se te osobne informacije prosljeđuju Firebase Authentication SDK-u.

Nakon uspješne autentikacije, korisnik može pristupiti svom korisničkom profilu i informacijama na njemu, te kontrolirati korisnički pristup podacima koji su zapisani u bazi. Firebase Authentication usluga također pruža mogućnosti kontroliranja korisničke lozinke, kao potvrđivanja elektroničke pošte, odnosno identiteta korisnika. Primjer izgleda Firebase Authentication usluge uz neke izrađene korisničke račune ove aplikacije prikazuje slika 3.2.

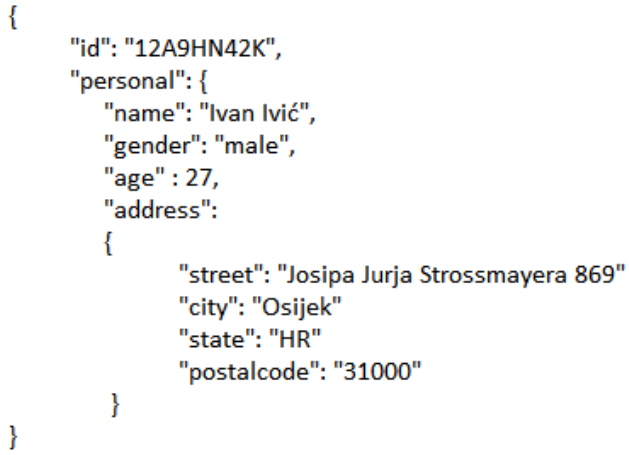


Slika 3.2 *Primjer izgleda Firebase Authentication usluge*

3.3. Firebase Realtime Database

Uz autentikaciju korisnika, potrebna je i usluga koja će omogućiti ostvarenje modela baze podataka za pohranu informacija od spomenutih korisnika i općenito funkcioniranje aplikacije. Usluga Firebase Realtime Database služi za stvaranje hijerarhijske baze podataka pohranjene u oblaku računala. „Realtime“ označava da su svi podaci su sinkronizirani i ažuriraju se u stvarnom vremenu za svakog pojedinog klijenta. Pri gradnji međuplatformske aplikacije, svi klijenti dijele jednu instancu Realtime Baze podataka. Prema [15], svi podaci se prikazuju kao JSON datoteke. JSON je vrlo jednostavan, razumljiv i čitljiv datotečni format, a istovremeno je jednostavan za

stvaranje, parsiranje i obradu na računalima. Izgled jednostavne JSON datoteke je prikazan na slici 3.3.



Slika 3.3 *Primjer jednostavne JSON datoteke*

3.4. Model baze podataka

Prvi i osnovni entitet koji je potreban za izradu ove aplikacije je entitet *User*. Svaki korisnik će kao primarni ključ imati jedinstveni ID, te atribute e-mail i boolean varijablu pod nazivom testTaken koja će omogućiti automatsko učitavanje prijašnjeg rezultata upitnika umjesto ponovnog otvaranja upitnika ako ga je korisnik već ispunio. Kada korisnik odabere opciju da ponovno riješi upitnik, testTaken će se postaviti na false, a prijašnji rezultat će se obrisati.

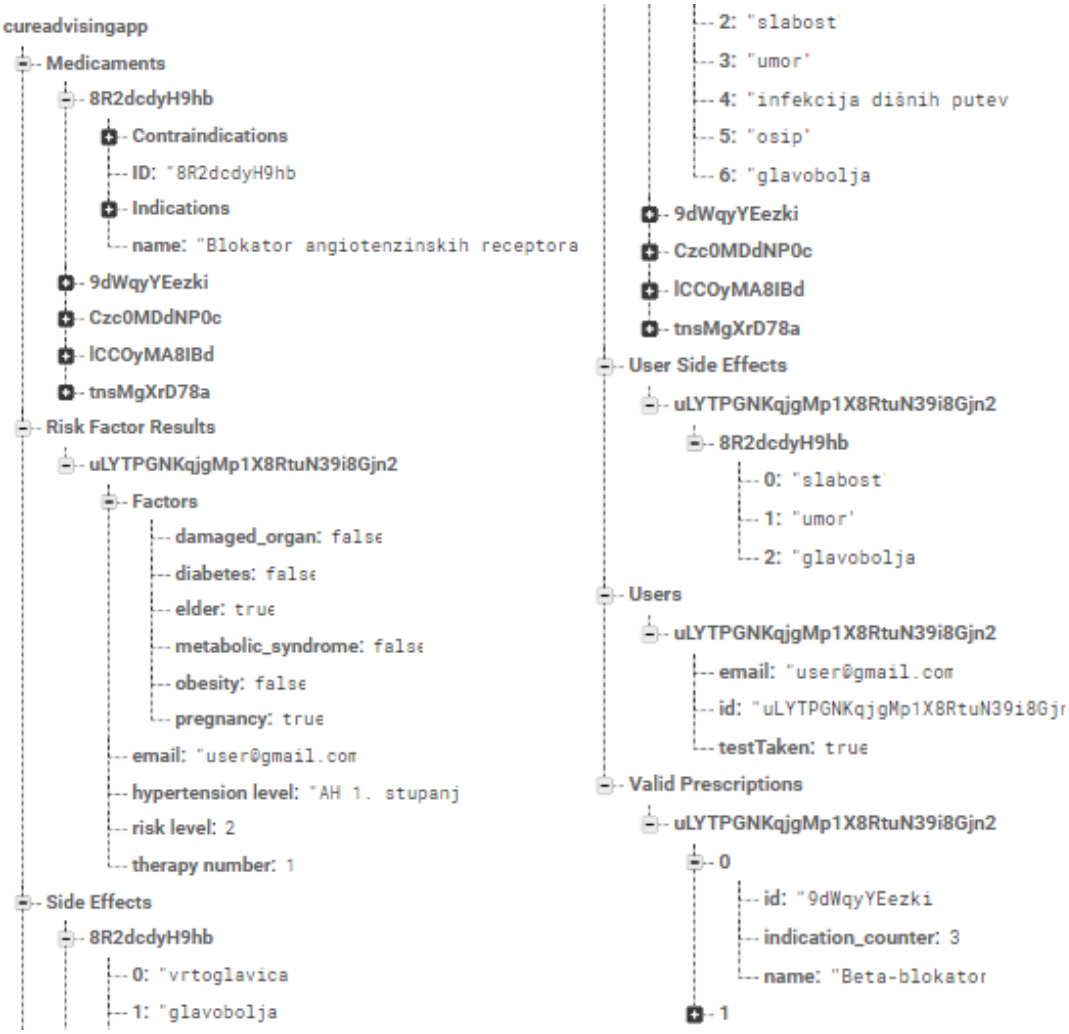
Sljedeći entitet je entitet *Risk Factor Results*, a najvažnije informacije koje će tu biti pohranjene su neki čimbenici rizika koji su važni za daljnje formiranje upitnika poput oštećenja organa, klasifikacija korisnikove hipertenzije, te therapy number, a to je broj koji će odrediti treba li korisniku preporučiti da odabere 1 ili više lijekova koji mu odgovaraju. Ovdje je primarni ključ opet korisnikov ID.

Entitet *Medicaments* kao instance drži pet lijekova navedenih u potpoglavlju 2.4.2.. Svaki lijek ima jedinstveni ID, te svaki sadrži ime, listu indikacija, te listu kontraindikacija iz tablica 2.6, te 2.7.

Valid Prescriptions je sljedeći entitet, koji je vrlo važan za prikaz rezultata. On za svakog korisnika ima listu gdje svaki element liste sadrži ID lijeka, naziv lijeka, te broj indikacija koji se prebrojao upitnikom. Taj broj indikacija će omogućiti da sve lijekove koji kontraindikacijama neće štetiti korisniku poredamo prioritetno po broju indikacija. Entitet *Side Effects* za svaki lijek sadrži jednu

listu nuspojava. Primarni ključ svake liste nuspojava je ID lijeka. Ovdje će korisnik imati mogućnost nadodati novu nuspojavu, tako da ju i drugi korisnici mogu odabrati ako već ne postoji. Ostao je još entitet *User Side Effects*, koji za svakog korisnika ima jednu instancu. Ovdje svaki korisnik na kraju upitnika može sačuvati svoje nuspojave, za svaki lijek odvojeno.

Slika 3.4 prikazuje model baze podataka s jednim izrađenim korisnikom i ispunjenim upitnikom.



Slika 3.4 *Prikaz baze podataka*

4. PROGRAMSKO RJEŠENJE ANDROID MOBILNE APLIKACIJE

4.1. Android Studio

Za izradu mobilne Android aplikacije koristi se integrirano razvojno okruženje (IDE) Android Studio. Android aplikacije mogu biti napisane koristeći programske jezike Kotlin, Java ili C++. Ova aplikacija pisana je u programskom jeziku Java. Android SDK alati prevode programski kod sa svim podacima i resursnim datotekama i stvaraju od toga datoteku -.apk. Ta datoteka služi da bi se aplikacija instalirala na jedan od uređaja s Android operacijskim sustavom.

Prema [16], svaka Android aplikacija je zaštićena sljedećim Android zaštitnim značajkama:

- Android operacijski sustav je višekorisnički Linux sustav gdje svaka aplikacija predstavlja jednog korisnika
- Sustav svakoj aplikaciji dodjeljuje korisnički ID koji je korišten samo od strane sustava i nepoznat je aplikaciji. Sustav određuje dopuštenja za sve datoteke u aplikaciji tako da samo ID koji je dodijeljen toj aplikaciji može pristupiti navedenim datotekama
- Svaki proces ima svoj vlastiti VM (virtualnu mašinu) tako da se programski kod aplikacije pokreće odvojeno od ostalih aplikacija
- Prema zadanim postavkama, svaka aplikacija se pokreće kao nezavisan Linux proces. Android sustav pokreće taj proces kada jedna od komponenti aplikacija treba biti izvršena, a zatim ga ubija kada on više nije potreban ili kada je sustavu potrebna memorija za pokretanje drugih aplikacije

Moguće je ostvariti dijeljenje podataka između aplikacija ako im se dodjeli isti Linux korisnički ID. Također, svaka aplikacija može zatražiti pravo pristupa podatcima uređaja poput kontakata, SD kartici, kameri ili Bluetoothu. Naravno korisnik mora odobriti ovaj pristup.

Android aplikacija se sastoji od sljedećih osnovnih komponenti:

- *Activities*
- *Services*
- *Broadcast receivers*
- *Content providers*

Od komponenti će se proći samo kroz aktivnosti, jer je aktivnost jedina komponenta koja će biti potrebna u aplikaciji. Ali prije toga je potrebno objasniti Manifest datoteku, mapu resursa aplikacije, te neke od osnovnih kontrola za stvaranje korisničkog sučelja.

4.1.1. Manifest

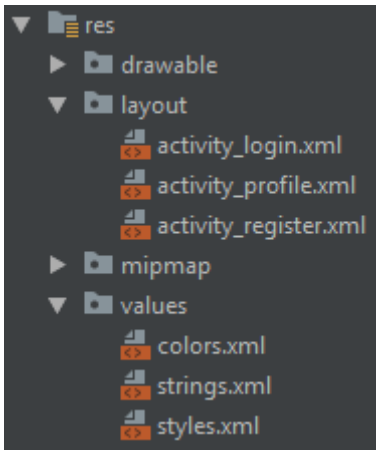
Nakon stvaranja Android Studio projekta, u mapi app/manifests unutar strukture projekta se nalazi datoteka AndroidManifest.xml. Ta datoteka sadrži sve vitalne informacije o aplikaciji, a pisana je XML opisnim jezikom. Ona definira strukturu aplikacije, sadrži naziv paketa, sve aktivnosti, servise, pružatelje sadržaja, primatelje prijenosa, intent filtere koji opisuju kako pojedina komponenta može biti pokrenuta, dozvole, definira ikonu, naziv aplikacije kakav će biti prikazan korisniku ispod ikone, verziju aplikacije itd. Jednostavna Manifest datoteka je prikazana na slici 4.1.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="hr.ferit.davidbilic.cureadvisingapplication">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        tools:ignore="GoogleAppIndexingWarning">
        <activity android:name=".SideEffectsActivity"/>
        <activity android:name=".FinalActivity" />
        <activity android:name=".IndicationsActivity" />
        <activity android:name=".ContraindicationsActivity" />
        <activity android:name=".AppropriateTherapyActivity" />
        <activity android:name=".RiskFactorsActivity" />
        <activity android:name=".LoginActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".RegisterActivity" />
        <activity android:name=".PressureActivity" />
    </application>
</manifest>
```

Slika 4.1 *Izgled jednostavne Manifest datoteke*

4.1.2. Resursi

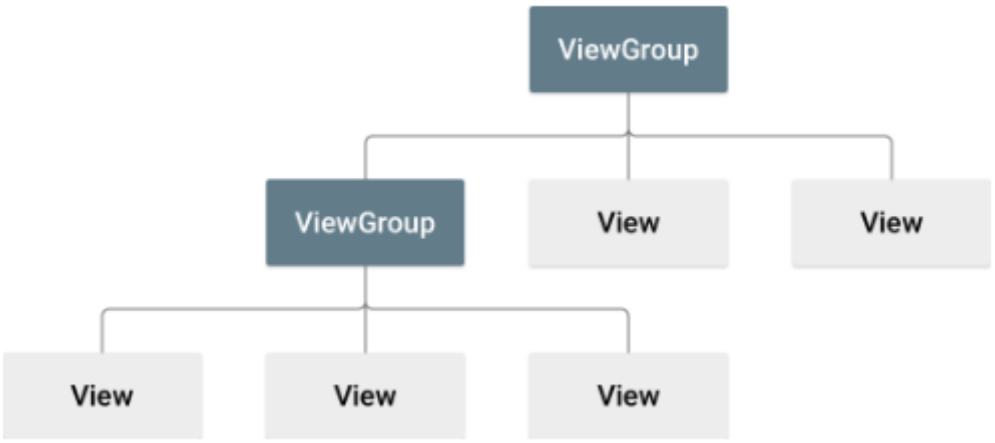
Kod razvoja Android aplikacija, kod se uvijek odvaja od resursa. Resursi su sve dodatne datoteke poput: izgleda korisničkih sučelja za aktivnost (engl. layouts), slika, ikonica aplikacije, stringova, boja, stilova, dimenzija itd. Ovakvo razdvajanje značajno olakšava naknadne izmjene ali i prilagodbe različitim uređajima. Svi resursi za aplikaciju su smješteni u mapi res unutar projekta, a oni se zajedno sa svima pakiraju u već spomenutu .apk datoteku. Osnovna struktura resursa za Android aplikaciju prikazana je slikom 4.2.



Slika 4.2 *Resursi Android aplikacije*

4.1.3. Izgled i kontrole korisničkog sučelja

Na slici 4.2 postoji mapa pod nazivom layout. Ta mapa za sadrži datoteke koje definiraju izglede različitih korisničkih sučelja koje mogu predstavljati izglede aktivnosti ili nešto drugo. Navedene datoteke su pisane u XML formatu i unutar njih se definiraju različiti elementi ili kontrole korisničkog sučelja. Svaki *layout* sadrži korijenski element koji mora biti ili View ili ViewGroup objekt. Zatim se unutar tog korijenskog elementa postavljaju ostali elementi koji će definirati kompletan izgled korisničkog sučelja. Najčešće korijenski element bude ViewGroup objekt, a unutar njih se smještaju View elementi. Slika 4.3 prikazuje UI hijerarhiju prema [17].



Slika 4.3 *Prikaz UI hijerarhije* [17]

Najčešće dvije klase koje su izvedene iz ViewGroup klase su LinearLayout i Realtive layout.

LinearLayout je najjednostavniji oblik *layouta* u kojemu se kontrole smještaju jedna ispod druge, ili jedna pored druge, ovisno o tome jesmo li postavili okomitu ili vodoravnu orijentaciju. Korisno svojstvo LinearLayout-a je svojstvo weight koje omogućuje raspoređivanje elemenata takvo da, okomito ili vodoravno, zauzimaju određene dijelove ekrana.

RelativeLayout je *layout* koji s druge strane omogućuje da se kontrole smještaju unutar njega u relativno u odnosu na druge elemente ili u odnosu na neke točke na ekranu kao npr. dolje lijevo. Najčešće se jedna kontrola smjesti u odnosu na točku na ekrana, a ostale se nadovezuju jedna od druge pomoću svojstava layout_toRightOf, layout_toLeftOf itd.

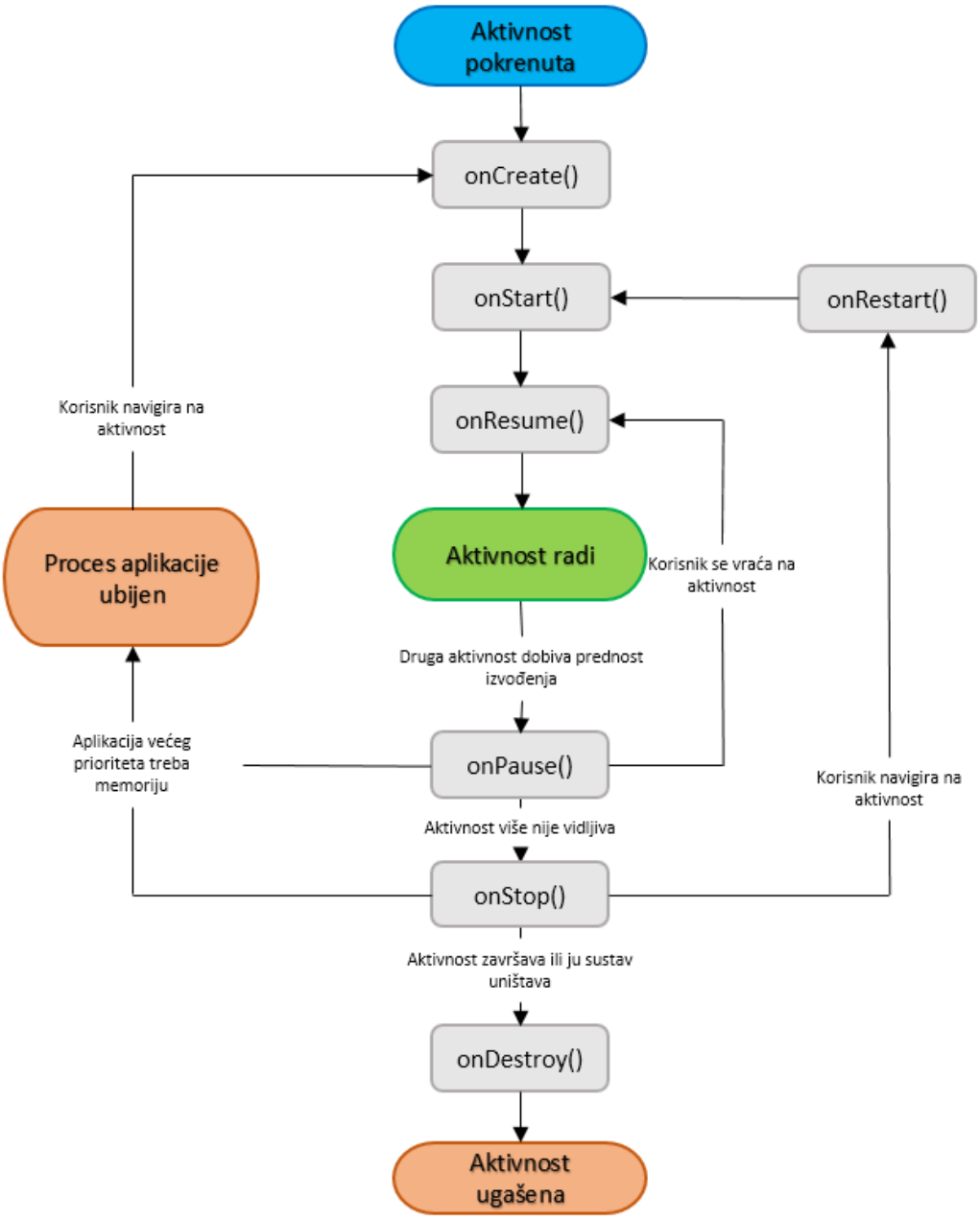
Sada nakon osnovnih korijenskih elemenata treba prijeći na UI kontrole. UI kontrole su izvedene iz klase View, a neke osnovne su:

- **TextView** – omogućuje prikaz tekstualnog sadržaja, ključno svojstvo joj je text
- **EditText** – omogućuje unos alfanumeričkih znakova od strane korisnika, ima mogućnost skrivanja prikaza unesenog teksta, prikaz hint-a za unos, mogućnost unosa samo brojeva itd.
- **Button** – kontrola koja treba pokrenuti neku akciju, na njega najčešće u Java kodu tj. Activity klasi postavljamo OnClickListener() sučelje koje će se aktivirati pritiskom na gumb i obaviti akciju koju smo definirali programom
- **ImageView** – omogućuje prikaz slike na zaslonu

4.1.4. Aktivnosti

Prema [18], klasa Acitivity je ključna komponenta android aplikacije, i način na koji su aktivnosti ostvarene i povezane čini aplikaciju, a u Activity klasu se piše programski kod koji ostvaruje funkcionalnost korisničkog sučelja. Aktivnost predstavlja jedan zaslon aplikacije, primjerice LoginAcitivity ili RegisterActivity, od kojih svaka ima svoju layout XML datoteku u mapu s resursima. Pomoću intent filtera u Manifest datoteci se može odrediti koja će biti glavna aktivnost koja će se prva pokrenuti kada pritisnemo na ikonicu aplikacije. Aktivnosti za razliku od većine drugih programskih kodova nemaju main() metodu koja će predstavljati ulaznu točku programa već one imaju svoj životni ciklus. Taj životni ciklus se sastoji od callback metoda koje se pozivaju svaki puta kada neki događaj utječe na životni ciklus aktivnosti. Takav događaj može predstavljati:

stvaranje aktivnosti, pauziranje aktivnosti, zaustavljanje aktivnosti, uništavanje aktivnosti itd. Životni ciklus aktivnosti je prikazan slikom 4.4 po uzoru na [19].



Slika 4.4 *Životni ciklus aktivnosti* [19]

Budući da su resursi odvojeni od koda, to znači da je korisničko sučelje s kontrolama u .xml formatu također odvojeno od koda aktivnosti kojemu pripada. Zato je to korisničko sučelje potrebno „napuhati“ prilikom stvaranja odgovarajuće aktivnosti svaki puta kada se aktivnost pokreće. To se po slici 4.4 odvija u callback metodi OnCreate(), a „napuhivanje“ obavlja metoda setContentView() kojoj se kao argument daje resursna datoteka iz mape layout koja predstavlja korisničko sučelje za određenu aktivnost.

4.2. Izrada aktivnosti za prijavu i registriranje korisnika

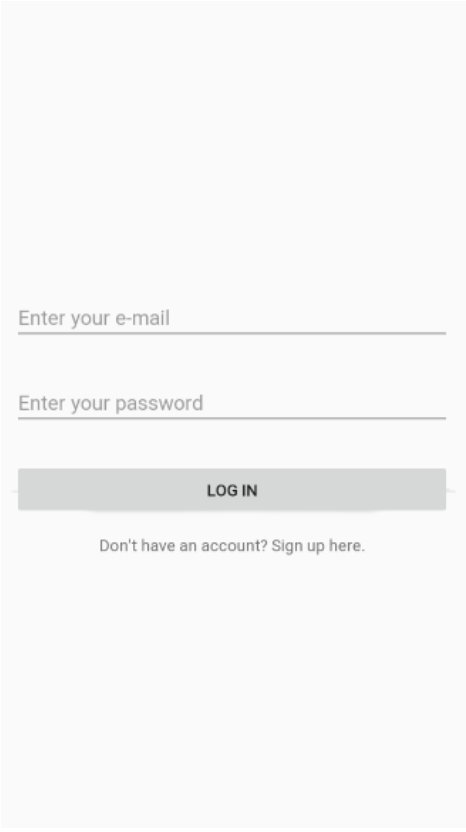
Kada god se u aplikaciji radi s Firebase Authentication uslugom, bit će potreban objekt FirebaseAuth klase. Kada se podatak zapisuje u Firebase Realtime Database uslugu, potreban je objekt DatabaseReference klase kojemu se pri stvaranju predaje najbliži čvor u bazi podataka s kojim se želi raditi. Također tu je i objekt klase FirebaseUser koji se koristi samo za dohvaćanje korisničkog ID-a iz Authentication usluge. Taj ID će predstavljati primarne ključeve korisnika u bazi podataka.

Instanciranje objekata navedenih klasa je prikazano slikom 4.5. Ova tri objekta se koriste u svakoj aktivnosti i jednako su nazvani u svakoj aktivnosti.

```
firebaseAuthenticationService = FirebaseAuth.getInstance();
user= firebaseAuthenticationService.getCurrentUser();
databaseRef= FirebaseDatabase.getInstance().getReference();
```

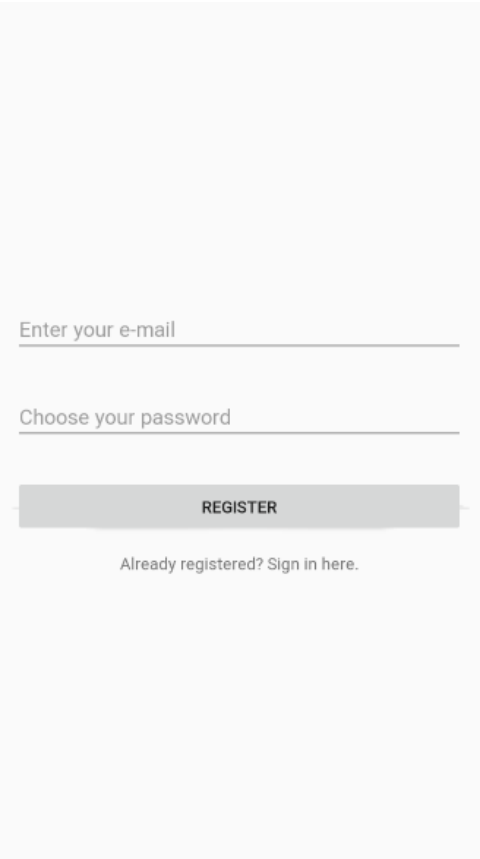
Slika 4.5 *Instanciranje objekata za rad s bazom podataka*

Kada korisnik uđe u aplikaciju, otvara mu se aktivnost za prijavu. Ona je nazvana LoginActivity, a njezin izgled je prikazan na slici 4.6.



Slika 4.6 *Izgled aktivnosti za prijavu*

Korisnik ovdje ima opciju da klikom na TextView otvori aktivnost za registraciju, koja je izgledom jednaka kao i aktivnost za prijavu, a prikazana je na slici 4.7.



Slika 4.7 *Izgled aktivnosti za registriranje*

Pri pokretanju metode za registraciju, već u onStart() metodi sa Slike 4.4 pomoću instance FirebaseAuth klase slijedi provjera je li korisnik ostao ulogiran u aplikaciju da ga se odmah preusmjeri na početak upitnika tj. na aktivnost za unos vrijednosti krvnog tlaka. Ova metoda je prikazana slikom 4.8.

```
@Override
protected void onStart() {
    super.onStart();
    //ako je korinik već ulogiran, prebaci ga na aktivnost za unos tlaka
    if(firebaseAuthenticationService.getCurrentUser()!= null){
        finish();
        startActivity(new Intent( packageContext: RegisterActivity.this, PressureActivity.class));
    }
}
```

Slika 4.8 *Preusmjeravanje već prijavljenog korisnika*

Ako korisnik pritisne na gumb za „REGISTER“, započinje process zapisivanja korisnika u Firebase Authentication uslugu pomoću spomenutog objekta FirebaseAuth klase. Slika 4.9

prikazuje poziv metode za kreiranje novog korisnika u Authentication usluzi koji se odvija nakon provjere unesenih podataka. Pomoću onComplete listnera mi „osluškujemo“ je li metoda za zapis korisnika u Authentication uslugu dovršena. Ako je dovršena, te ako je zapis novog korisnika uspješno izvršen, poziva se metoda saveUserInDatabase() koja stvara objekt klase User, te zapisuje korisnika u bazu podataka pod entitet *Users*. Zatim se automatski pokreće aktivnost za unos vrijednosti tlakova.

```
firebaseAuthenticationService.createUserWithEmailAndPassword(email, password)
    .addOnCompleteListener( activity: this, (task) → {
         progressDialog.dismiss();
         if(task.isSuccessful()){
             //korisnik je uspješno kreiran i ulogiran
          saveUserInDatabase();
          finish();
          startActivity(new Intent( packageContext: RegisterActivity.this, PressureActivity.class));
          Toast.makeText(getApplicationContext(),  text: "User registered!" , Toast.LENGTH_SHORT).show();
       }else{
           Toast.makeText(getApplicationContext(),  text: "Error while registering user." , Toast.LENGTH_SHORT).show();
         }
    });
```

```
public void saveUserInDatabase(){
    FirebaseUser user = firebaseAuthenticationService.getCurrentUser();
    String email= etEmail.getText().toString().trim();
    String id = user.getUid();
    User newUser = new User(email, id);
    databaseRef.child("Users").child(id).setValue(newUser);
}
```

Slika 4.9 *Kreiranje novog korisnika i njegov zapis u bazu podataka*

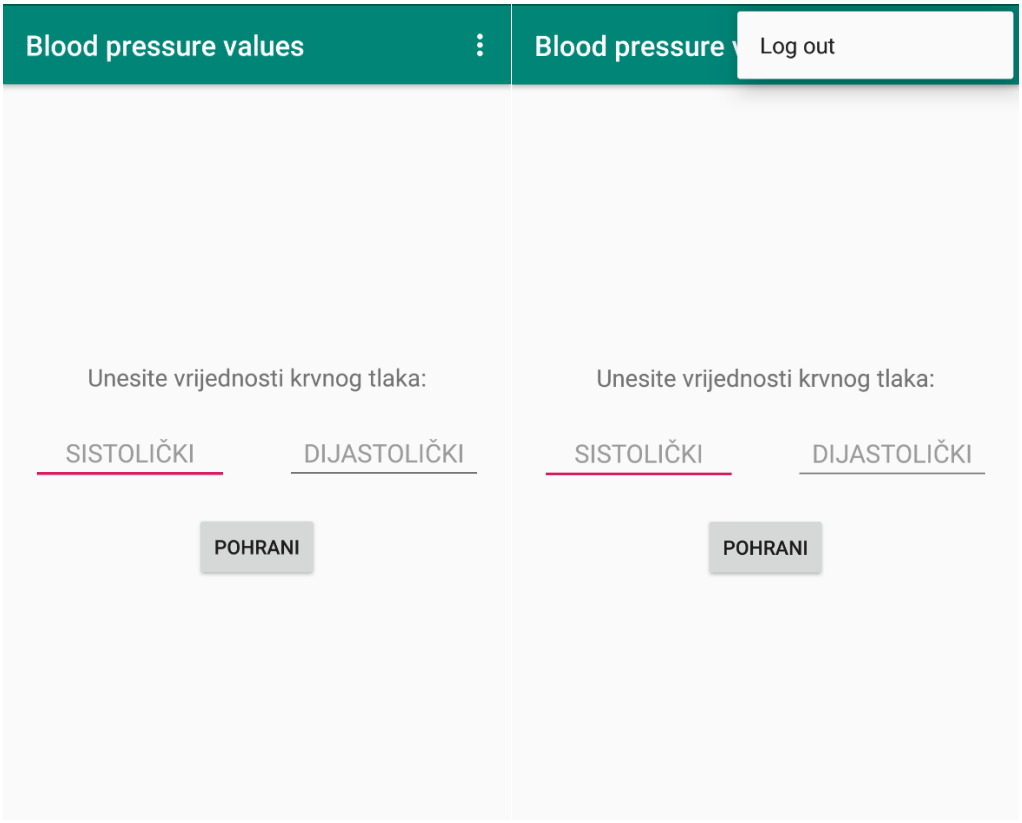
Aktivnost za prijavu prilikom pritiska na gumb za prijavu s kontrola učitava uneseni email i lozinku, te ako nisu prazni pokuša prijaviti korisnika u Firebase Authentication uslugu. Ako je prijava uspješna, poziva se metoda setAppropriateUI() koja korisnika smješta u ispravnu aktivnost ovisno o tome je li ispunio test, te ako ga je ispunio, treba li mu farmakološka terapija. Slika 4.10 prikazuje poziv metode za prijavu korisnika, a metoda setAppropriateUI() će biti kasnije objašnjena.

```
firebaseAuthenticationService.signInWithEmailAndPassword(email, password)
    .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
     progressDialog.dismiss();
     if(task.isSuccessful()){
           //start the profile activity and finish login activity
       setAppropriateUI();
     }else{
       Toast.makeText(getApplicationContext(), text: "Error, invalid email or password.", Toast.LENGTH_SHORT).show();
     }
    }
});
```

Slika 4.10 *Proces prijave u Authentication uslugu*

4.3. Izrada upitnika

Nakon što se korisnik registrira i prijavi (ili samo prijavi), otvara mu se aktivnost za unos vrijednosti krvnih tlakova, a zove se PressureActivity. Njezin izgled je prikazan na slikom 4.11.



Slika 4.11 *Aktivnost za unos vrijednosti tlakova*

Treba napomenuti da je u svaku aktivnost je implementiran izbornik s jednom opcijom za odjavu iz aplikacije. Također, svaka aktivnost upitnika odmah pri učitavanju provjerava je li izgubljena veza s Firebase Authentication uslugom, ako je, aktivnost se završava i pokreće se aktivnost za prijavu. Nakon unosa vrijednosti tlakova, pritisak na gumb „POHRANI“ će prvo provjeriti je li jedan od EditText-ova ostao prazan, a ako nije, poziva se metoda checkValues() koja po tablici 2.1 određuje klasifikaciju krvnog tlaka i u isto vrijeme provjerava ispravnost unesenih podataka. Ako su podaci ispravni, određuje se klasifikacija krvnog tlaka i metoda vraća vrijednost true. Nakon toga se dodijeljena klasifikacija pomoću Intent objekta i njegove metode putExtra() šalje u iduću aktivnost. Metoda checkValues() je prikazana na slici 4.12. Klasifikacija se u ovoj metodi obavlja dodjeljivanjem pripadajućeg stringa String varijabli hypertensionLvl.

```
private boolean checkValues(){
    int systolic= Integer.parseInt(etSYS.getText().toString().trim());
    int diastolic= Integer.parseInt(etDIAS.getText().toString().trim());

    if(systolic <= 90 || diastolic<= 50){
        Toast.makeText( context: this, text: "Values are invalid, please insert correct values", Toast.LENGTH_LONG).show()
        return false; }

    else if(( systolic >= 180) || (diastolic >= 110)){
        hypertensionLvl = "AH 3. stupanj" ;
        Toast.makeText( context: this, text: "Level 3. Values saved.", Toast.LENGTH_LONG).show();
        return true; }

    else if(((179 >= systolic)&&( systolic >= 160)) || ((109 >= diastolic)&&( diastolic >= 100))){
        hypertensionLvl = "AH 2. stupanj";
        Toast.makeText( context: this, text: "Level 2. Values saved.", Toast.LENGTH_LONG).show();
        return true; }

    else if(((159 >= systolic)&&( systolic >= 140)) || ((99 >= diastolic)&&( diastolic >= 90))){
        hypertensionLvl = "AH 1. stupanj";
        Toast.makeText( context: this, text: "Level 1. Values saved.", Toast.LENGTH_LONG).show();
        return true; }

    else if(((139 >= systolic)&&( systolic >= 130)) || ((89 >= diastolic)&&( diastolic >= 85))){
        hypertensionLvl = "Visoko normalan";
        Toast.makeText( context: this, text: "Normal High. Values saved.", Toast.LENGTH_LONG).show();
        return true; }

    else if(((129 >= systolic)&&( systolic >= 120)) || ((84 >= diastolic)&&( diastolic >= 80))){
        hypertensionLvl = "Normalan";
        Toast.makeText( context: this, text: "Normal. Values saved.", Toast.LENGTH_LONG).show();
        return true; }

    else if(((120 > systolic)&&( systolic > 90)) || ((80 > diastolic)&&( diastolic > 50))){
        hypertensionLvl = "Optimalan";
        Toast.makeText( context: this, text: "Optimal. Values saved.", Toast.LENGTH_LONG).show();
        return true; }

    else{
        Toast.makeText( context: this, text: "Something went wrong, please try again.", Toast.LENGTH_LONG).show();
        return false; } }
```

Slika 4.12 *Metoda za provjeru ispravnosti unesenih podataka i dodjelu razine krvnog tlaka*

Nakon što je klasifikacija krvnog tlaka određena, pokreće se aktivnost za unos postojećih čimbenika kardiovaskularnog rizika iz tablice 2.3. Izgled ove aktivnosti prikazan je slikom 4.13, a aktivnost je nazvana RiskFactorsActivity.

Slika 4.13 *Aktivnost za odabir čimbenika rizika*

Ova aktivnost pohranjuje dohvaćenu klasifikaciju krvnog tlaka, a njezin prvi zadatak je stvaranje varijabli u koje se pohranjuju određeni čimbenicii rizika poput prisutnosti šećerne bolesti, metaboličkog sindroma, pretilosti, starije dobi i oštećenja organa. Razlog dodatne pohrane navedenih pet rizičnih čimbenika je taj da ovi čimbenici rizika označavaju indikacije ili kontraindikacije nekih lijekova i bit će nam potrebni kasnije za elimininranje lijekova kao i ostale kontraindikacije koje će naknadno biti ponuđene, a nije poželjno iste čimbenike nuditi više puta da ih korisnik mora više puta iznova unositi. Slika 4.14 prikazuje navedene varijable, a svaki čimbenik rizika ima postavljenu zadanu vrijednost false. Tu su također i klasifikacija krvnog tlaka koja se dohvaća iz prošle aktivnosti, te cjelobrojna vrijednost visine kardiovaskularnog rizika.

```
private String hypertensionLvl;
private int riskLvl = 1;
private boolean obesity = false;
private boolean elder = false;
private boolean metabolic_syndrome = false;
private boolean diabetes = false;
private boolean damaged_organ = false;
```

Slika 4.14 *Najvažnije varijable RiskFactorsActivity-ja*

Pritisak na gumb „DALJE“ pokreće metodu calculateRiskLvl(). Ova najvažnija metoda mora izračunati prisustvo metaboličkog sindroma pomoću brojača koji se zove counterMetabolic, te postaviti sve varijable sa slike 4.14**,** osim klasifikacije krvnog tlaka koja je već poznata i dohvaćena u onCreate() metodi. Treba primijetiti da je riskLvl cjelobrojna vrijednost od jedan do četiri, gdje svaki broj pripada jednom retku iz najljevijeg stupca tablice 2.4 za određivanje terapije. Slika 4.15 prikazuje prvi dio opisane metode. Slika 4.16 prikazuje drugi dio opisane metode.

```
private void calculateRiskLvl(){
    int counter = 0;
    //BROJAC KOJI CE PRIKAZATI PRISUTNOST METABOLICKOG SINDROMA, ZA POCETAK SE PROVJERAVA VRIJEDNOST TLAKA
    int counterMetabolic=0;
    if (hypertensionLvl.matches( regex: "Visoko normalan") || hypertensionLvl.matches( regex: "AH 1. stupanj")
            || hypertensionLvl.matches( regex: "AH 2. stupanj") || hypertensionLvl.matches( regex: "AH 3. stupanj")){
        counterMetabolic++;
    }
    //AKO JE SVE PRAZNO
    if(!chkSmoking.isChecked() && !chkElder.isChecked() && !chkObesity.isChecked()
            && !chkDyslipidemia.isChecked() && !chkAnamnesis.isChecked()
            && !chkGlucose.isChecked() && !chkDamagedOrgan.isChecked()
            && !chkDiabetes.isChecked() && !chkCardiovascular.isChecked() && !chkKidney.isChecked()){

        riskLvl = 1;
        Toast.makeText( context: this,  text: "RISK LVL 1 - NO RISK", Toast.LENGTH_LONG).show();
        return;
     //AKO NIJE SVE PRAZNO, OBAVI SVE PROVJERE
    }else {
        if(chkSmoking.isChecked()){
            counter++;
        }if(chkElder.isChecked()) {
            counter++;
            elder=true;
        }if(chkObesity.isChecked()){
            counter++;
            counterMetabolic++;
            obesity = true;
        }if(chkDyslipidemia.isChecked()){
            counter++;
            counterMetabolic++;
        }if(chkAnamnesis.isChecked()){
            counter++;
        }if(chkGlucose.isChecked()){
            counter++;
            counterMetabolic++;
        }
        if(chkDamagedOrgan.isChecked()){
            damaged_organ = true;
        }
        if(chkDiabetes.isChecked()){
            diabetes=true;
        }
        if(counterMetabolic>=3){
            metabolic_syndrome=true;
        }
```

Slika 4.15 *Prvi dio metode calculateRiskLvl()*

Prvi dio metode računa prisustvo metaboličkog sindroma te ključnih pet čimbenika rizika sa slike 4.14. Ovdje se također pomoću varijable counter broji koliko je ukupno uobičajenih čimbenika

rizika korisnik odabrao. Na početku se prvo provjerava slučaj u kojemu nije odabran niti jedan čimbenik rizika .

```
//SLIJEDI PROVJERA KOJEM RETKU TABLICE 2.4. KORISNIK PRIPADA
    if((counter>=3 || metabolic_syndrome==true) && (!chkDamagedOrgan.isChecked()) &&
            (!chkDiabetes.isChecked()) && (!chkCardiovascular.isChecked()) && (!chkKidney.isChecked())){
        riskLvl=3;
        Toast.makeText( context: this,  text: "Risk LVL3", Toast.LENGTH_LONG).show();
        return;

    }else if((counter==1 || counter ==2) && (!chkDamagedOrgan.isChecked())
            && (!chkDiabetes.isChecked()) && (!chkCardiovascular.isChecked()) && (!chkKidney.isChecked())){
        riskLvl = 2;
        Toast.makeText( context: this,  text: "Risk LVL2", Toast.LENGTH_LONG).show();
        return;

    }else if((counter == 0 && chkDiabetes.isChecked() && (!chkDamagedOrgan.isChecked()) &&
            (!chkCardiovascular.isChecked()) && (!chkKidney.isChecked())) ||
            (counter == 0 && chkDamagedOrgan.isChecked() && (!chkDiabetes.isChecked())
                    && (!chkCardiovascular.isChecked()) && (!chkKidney.isChecked()))){
        riskLvl=3;
        Toast.makeText( context: this,  text: "Risk LVL3", Toast.LENGTH_LONG).show();
        return;

    }else{
        riskLvl=4;
        Toast.makeText( context: this,  text: "Risk LVL4", Toast.LENGTH_LONG).show();
        return;
    }
```

Slika 4.16 *Drugi dio metode calculateRiskLvl()*

Drugi dio metode, ovisno o varijabli counter, te o četiri značajnija čimbenika rizika sa slike 4.13, određuje ostale cjelobrojne vrijednosti čimbenika rizika i dodjeljuje ih varijabli riskLvl.

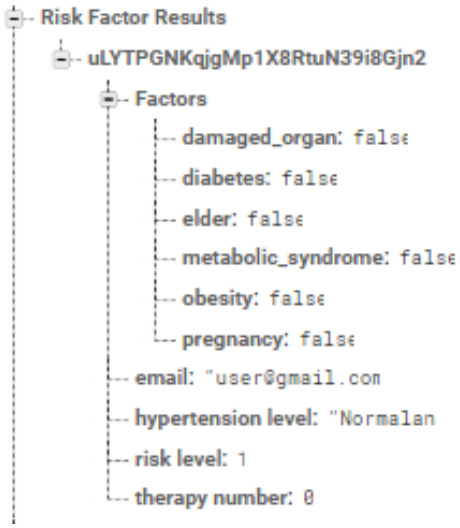
Sada kada su sve varijable rizika postavljene, poziva se metoda saveResultInDatabase(). Ako korisnik nema hipertenziju, ne trebaju mu lijekovi i u testTaken atribut u bazi podataka mu se postavlja vrijednost true jer je za njega upitnik gotov i idućoj aktivnosti mu slijedi prikaz rezultata s opcijom da ponovi upitnik. Zatim se stvara objekt UserInfo klase. UserInfo klasa pohranjuje email, razinu krvnog tlaka, cjelobrojnu razinu rizika, a sadrži i therapy_number cjelobrojnu varijablu koja predstavlja treba li korisniku preporučiti uzimanje nula, jedan ili dva lijeka. Ova varijabla se pri stvaranju objekta postavlja na nula i njezina vrijednost se ne postavlja konstruktorom, već se u sljedećoj aktivnosti individualno ažurira u bazi podataka.

Zatim se stvara objekt RiskFactor result klase u kojeg se pohranjuju metabolički sindrom, pretilost, starost, dijabetes i oštećenje organa. RiskFactorResult također ima atribut pregnancy koji se u konstruktoru postavlja na false, a kasnije mu se dodjeli vrijednost kada korisniku bude ponuđena opcija „Trudnoća“ pri odabiru kontraindikacija. Navedena dva objekta se pohranjuju u bazu podataka u entitet *Risk Factor Results*, a primarni ključ im je ID korisnika. Slika 4.17 prikazuje

pojašnjenu metodu saveResultInDatabase(), dok slika 4.18 prikazuje izgled jednog RiskFactorResult objekta objavljenog u bazu podataka.

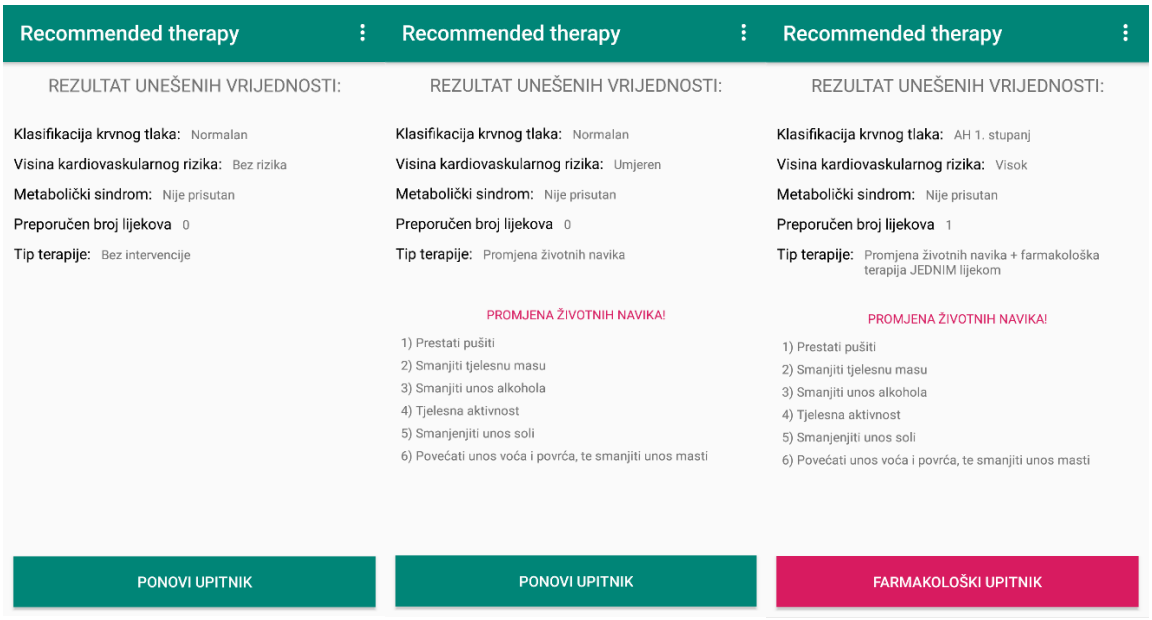
```
private void saveResultInDatabase(){
    FirebaseUser user = firebaseAuthenticationService.getCurrentUser();
    //AKO FARMAKOLOSKA TERAPIJA NIJE POTREBNA, UPISUJEMO POD USERS DA JE TEST RIJESEN
    if(hypertensionLvl.matches( regex: "Optimalan") || hypertensionLvl.matches( regex: "Normalan")
            || hypertensionLvl.matches( regex: "Visoko normalan")){
        databaseRef.child("Users").child(user.getUid()).child("testTaken").setValue(true);
    }
    //U SVAKOM SLUCAJU SPREMAMO RIKFACTORRESULT OBJEKT U BAZU
    String email= user.getEmail();
    UserInfo userInfo= new UserInfo(email, hypertensionLvl, riskLvl); //therapyNumber automatski postavljen na 0
    RiskFactorResult result = new RiskFactorResult(obesity, elder, metabolic_syndrome, diabetes, damaged_organ);
    databaseRef.child("Risk Factor Results").child(user.getUid()).setValue(userInfo);
    databaseRef.child("Risk Factor Results").child(user.getUid()).child("Factors").setValue(result);
}
```

Slika 4.16 *Metoda saveResultInDatabase()*



Slika 4.17 *Izgled pohranjenih čimbenika rizika u bazi podataka*

Nakon što je rezultat čimbenika rizika pohranjen u bazu podataka, pokreće se iduća aktivnost za prikaz dosadašnjih rezultata. Naziv ove aktivnosti je AppropriateTherapyActivity i ona prvo iz baze podataka iz *Risk Factor Results* entiteta pročita dosad unesene podatke. Njezin zadatak je prikazati rezultate dosad unesenih podataka na ekranu, odrediti tip terapije po tablici 2.4, savjetovati korisniku kako da promjeni životne navike ako je to potrebno, te odrediti broj potrebnih lijekova po slici 2.5. Ova aktivnost je specifična jer joj se izgled mijenja ovisno o tome je li korisniku potrebna farmakološka terapija ili nije. Ako korisniku nije potrebna farmakološka terapija, on nema opciju da nastavi dalje s upitnikom već samo da ponovi test. Sva tri moguća izgleda ove aktivnosti su prikazana slikom 4.18.



Slika 4.18 *Rezultati unesenih čimbenika rizika*

Ako korisniku nije potrebna farmakološka terapija, on ima opciju ponavljanja upitnika i dok ga ne ponovi, svaki puta kada se ulogira će ga se prebaciti na upravo ovu aktivnost i prikazivat će mu prethodne rezultate. Klik na „PONOVI UPITNIK“ će iz baze obrisati instancu entiteta *Risk Factor Results* za ovog korisnika, a testTaken atribut entiteta *Users* će postaviti na false. Ako je korisniku potrebna farmakološka terapija, on ima opciju da nastavi farmakološki upitnik. Njegov testTaken je tada još uvijek postavljen na false i postavit će ga se na true tek kada ispuni upitnik do kraja. Ako takav korisnik ovom trenutku zatvori aplikaciju, on će morati ponovno unijeti vrijednosti tlaka i čimbenike rizika. Metoda za dohvaćanje razine hipertenzije, kardiovaskularnog rizika te navedenih čimbenika rizika iz baze podataka je prikazuje slika 4.19.

```
private void catchPreviousResult() {
    //DOHVACANJE RAZINE HIPERTENZIJE I RIZIKA IZ BAZE PODATAKA
    DatabaseReference DatabaseRiskReference = FirebaseDatabase.getInstance()
            .getReference( path: "Risk Factor Results/"+ user.getUid().toString());
    DatabaseRiskReference.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            hypertensionLvl = dataSnapshot.child("hypertension level").getValue(String.class);
            riskLvl = dataSnapshot.child("risk level").getValue(int.class);
            metabolic_syndrome = dataSnapshot.child("Factors").child("metabolic_syndrome").getValue(boolean.class);
            diabetes = dataSnapshot.child("Factors").child("diabetes").getValue(boolean.class);
            damaged_organ = dataSnapshot.child("Factors").child("damaged_organ").getValue(boolean.class);
            obesity = dataSnapshot.child("Factors").child("obesity").getValue(boolean.class);
            elder = dataSnapshot.child("Factors").child("elder").getValue(boolean.class);

            tvHypertensionTXT.setText(hypertensionLvl);
            //MORAMO OVDJE POZIVATI METODE JER JE OVO ASINKRONA FUNKCIJA
            hideUIcontrols();
            setValues();  //POSTAVLJANJE TEXTVIEWA i therapy_numbera
            uploadTherapyNumber(); //POSTAVLJANJE U BAZU PODATAKA
        }
```

Slika 4.19 *Dohvaćanje podataka odmah po pokretanju aktivnosti*

Ova metoda nakon dohvaćanja podataka u sebi poziva metodu za promjenu korisničkog sučelja te metodu setValues() za popunjavanje kontrola i postavljanje varijable therapy_number. Varijabla therapy_number se aproksimira po slici 2.5, a kontrole se popunjavaju po tablici 2.4. Također je pomoću metode uploadTherapyNumber() potrebno dodijeljenu vrijednost varijabli therapy_number upisati u bazu podataka, jer je ona do sada za sve slučajeve bila 0. Slika 4.20 prikazuje samo dio metode setValues() da bi se prikazalo na koji način ona popunjava kontrole i određuje preporučen broj lijekova therapy_number.

```
}else if(hypertensionLvl.matches( regex: "AH 1. stupanj")) {
    if (riskLvl == 1) {
        therapy_number =1;
        tvRiskTXT.setText("Nizak");
        tvInterventionTXT.setText("Promjena životnih navika, ukoliko ne dođe do poboljšanj...");
        tvTherapyNumberTXT.setText(String.valueOf(therapy_number));
        return;
    } else if (riskLvl == 2) {
        therapy_number =1;
        tvRiskTXT.setText("Umjeren");
        tvInterventionTXT.setText("Promjena životnih navika, ukoliko ne dođe do poboljšanj...");
        tvTherapyNumberTXT.setText(String.valueOf(therapy_number));
        return;
    } else if (riskLvl == 3) {
        therapy_number =1;
        tvRiskTXT.setText("Visok");
        tvInterventionTXT.setText("Promjena životnih navika + farmakološka terapija JEDNIM...");
        tvTherapyNumberTXT.setText(String.valueOf(therapy_number));
        return;
    } else {
        therapy_number =2;
        tvRiskTXT.setText("Vrlo visok");
        tvInterventionTXT.setText("Promjena životnih navika + farmakološka terapija KOMBIN...");
        tvTherapyNumberTXT.setText(String.valueOf(therapy_number));
        return;
    }
}else if(hypertensionLvl.matches( regex: "AH 2. stupanj")) {
```

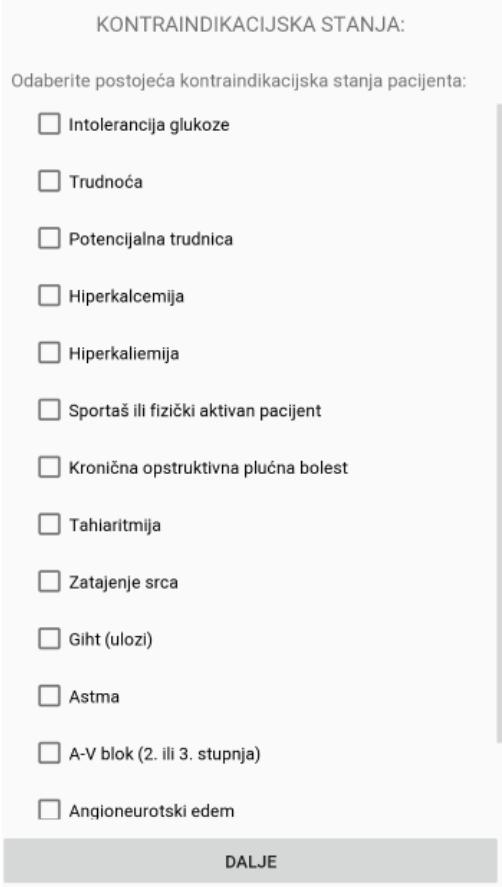
Slika 4.20 *Postavljanje kontrola i therapy_number varijable*

Ako korisnik klikne na gumb „FARMAKOLOŠKI UPITNIK“, poziva se metoda startContraindicationActivity() koja u novu aktivnost prenosi objekt RiskFactorResult klase popunjen dohvaćenim čimbenicima rizika. Slika 4.21 prikazuje opisanu metodu.

```
private void startContraindicationActivity(){
    Intent explicitIntent = new Intent( packageContext: AppropriateTherapyActivity.this, ContraindicationsActivity.class);
    RiskFactorResult rfResult = new RiskFactorResult(obesity, elder, metabolic_syndrome, diabetes, damaged_organ);
    explicitIntent.putExtra(RiskFactorsActivity.RISK_FACTOR_RESULT_KEY, rfResult);
    finish();
    startActivity(explicitIntent);
}
```

Slika 4.21 *Slanje čimbenika rizika i pokretanje nove aktivnosti*

Sada se pokreće nova aktivnost. Naziv joj je ContraindicationsActivity i služi nam za određivanje korisničkih kontraindikacija i eliminaciju lijekova, tako što je neke kontraindikacije već dohvatila u obliku objekta RiskFactorResult klase iz prošle aktivnosti, a neke dodatne kontraindikacije će korisnik sada moći unijeti preko checkBox kontrola. Izgled ove aktivnosti prikazuje slika 4.22.



Slika 4.22 *Aktivnost za dodatni unos kontraindikacija*

Za ovu aktivnost je stvorena klasa Medicament i u bazu podataka u entitet *Medicaments* je dodano pet navedenih lijekova. Slika 4.23 prikazuje atribute klase Medicament i jedan njezin ekvivalent u bazi podataka. Jedina razlika je ta što klasa Medicament dodatno ima atribut indication_counter koji će se koristiti u sljedećoj aktivnosti za indikacije, a konstruktorom se postavlja na nula. Taj atribut nije unesen u entitet *Medicaments* u bazi podataka. Slika 4.23 prikazuje atribute i neke metode klase Medicament, a s desne strane se vidi jedan od upisanih lijekova u bazi podataka

```java
public class Medicament {

    private String ID;
    private String name;
    private ArrayList<String> Contraindications;
    private ArrayList<String> Indications;
    private int indication_counter;

    public String getID(){return ID;}
    public String getName(){return name;}
    public ArrayList<String> getContraindications(){
        return Contraindications;}
    public ArrayList<String> getIndications(){
        return Indications;}
    public int getIndication_counter(){
        return indication_counter;}
    public String getContraindicationAt(int position){
        return Contraindications.get(position);}
    public String getIndicationAt(int position){
        return Indications.get(position);}

    //PRONALAZAK KONTRAINDIKACIJA I INDIKACIJA
    public boolean findContraindication(String wanted){...}
    public void findIndication(String wanted){...}
```

```
Medicaments
  8R2dcdyH9hb
    Contraindications
      0: "trudnoća
      1: "potencijalna trudnoć
      2: "hiperkaliemija
      3: "stenoza bubrežnih arterij
    ID: "8R2dcdyH9hb
    Indications
      0: "mladi"
      1: "dijabetes
      2: "metabolički sindror
      3: "bubrežna disfunkcij
      4: "bubrežno zatajivanje/proteinuri
      5: "mikroalbuminurij
      6: "prethodni infarkt miokard
      7: "hipertrofija lijeve klijet
    name: "Blokator angiotenzinskih receptora
```

Slika 4.23 *Dio klase Medicament i jedan unešeni lijek u bazi podataka*

U ovoj aktivnosti je stvorena lista stringova userList u koju će se pohranjivati korisnikove kontraindikacije, onih s prethodnih aktivnosti kao i onih trenutno odabranih. Također su stvorene reference na pet lijekova. U OnCreate() metodi se poziva metoda catchPreviousResult() koja dohvaća objekt klase RiskFactorResult iz prošle aktivnosti, te pet lijekova iz baze podataka koje se pridružuje stvorenim referencama. Slika 4.24 prikazuje stvorene reference na lijekove, dohvaćanje RiskFactorResult objekta iz prošle aktivnosti, te dohvaćanje lijekova.

```java
//region LIJEKOVI
Medicament ThiazideDiuretic;
Medicament BetaBlocker;
Medicament CAblocker;
Medicament ACEI;
Medicament ARB;
//endregion
private void catchPreviousResult() {
    Intent catchingIntent = this.getIntent();
    rfResult= (RiskFactorResult) catchingIntent
            .getSerializableExtra(RiskFactorsActivity.RISK_FACTOR_RESULT_KEY);
    databaseRef.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            ThiazideDiuretic = dataSnapshot.child("tnsMgXrD78a").getValue(Medicament.class);
            BetaBlocker = dataSnapshot.child("9dWqyYEezki").getValue(Medicament.class);
            CAblocker = dataSnapshot.child("CzcOMDdNP0c").getValue(Medicament.class);
            ACEI = dataSnapshot.child("lCCOyMA8IBd").getValue(Medicament.class);
            ARB = dataSnapshot.child("8R2dcdyH9hb").getValue(Medicament.class);
```

Slika 4.24 *Dohvaćanje lijekova iz baze podataka, te rizičnih čimbenika iz prošle aktivnosti*

Stvorena je i klasa MedicamentValidation koja za svaki od pet lijekova sadrži bool varijablu koja je inicijalno postavljena na true i označava je li taj lijek prikladan za korisnika. Ako korisnik unese kontraindikaciju koja se nalazi u listi kontraindikacija nekog lijeka, za taj lijek se spomenuta bool varijabla postavlja na false, što označava da za njega ovaj lijek nije prikladan i odmah se eliminira. Slika 4.25 prikazuje klasu MedicamentValidation.

```
public class MedicamentValidation implements Serializable {

    boolean DIURETICvalid;
    boolean BBvalid;
    boolean CAvalid;
    boolean ACEIvalid;
    boolean ARBvalid;

    //GETTERI
    public boolean getDIURETICvalid() { return DIURETICvalid; }
    public boolean getBBvalid() { return BBvalid; }
    public boolean getCAvalid() { return CAvalid; }
    public boolean getACEIvalid() { return ACEIvalid; }
    public boolean getARBvalid() { return ARBvalid; }


    //SETTERI
    public void setDIURETICvalid(boolean validity) { this.DIURETICvalid=validity; }
    public void setBBvalid(boolean validity) { this.BBvalid=validity; }
    public void setCAvalid(boolean validity) { this.CAvalid=validity; }
    public void setACEIvalid(boolean validity) { this.ACEIvalid=validity; }
    public void setARBvalid(boolean validity) { this.ARBvalid=validity; }

    //DEFAULT
    public MedicamentValidation(){
        this.DIURETICvalid = true;
        this.BBvalid = true;
        this.CAvalid = true;
        this.ACEIvalid = true;
        this.ARBvalid = true;
    }
}
```

Slika 4.25 *MedicamentValidation klasa*

Klik na gumb „DALJE“ će započeti popunjavanje korisničke liste kontraindikacija svim odabranim kontraindikacijama i kontraindikacijama čimbenika rizika iz dohvaćenog RiskFactorsResult objekta. Ovdje će se provjeriti spomenuti slučaj trudnoće i postaviti atribut za trudnoću u RiskFactorsResult objekt. Zatim će popunjenu korisničku listu usporediti s listom kontraindikacija svakog lijeka iz baze podataka zasebno. Ako se pronađe podudaranje kontraindikacija, bool vrijednost MedicamentValidation objekta za lijek čija je kontraindikacija se

postavlja na false. Slika 4.26 prikazuje metodu za popunjavanje korisnikove liste kontraindikacija, a slika 4.27 prikazuje metodu za postavljanje bool vrijednosti atributa objekta MedicamentValidation klase. Postavljanjem ovih bool vrijednosti je dovršena eliminacija neprikladnih lijekova.

```
private void fillUserList() {
    userList.clear();
    if(rfResult.getMetabolic_syndrome()) userList.add("metabolički sindrom");
    if(rfResult.getDiabetes()) userList.add("dijabetes");
    if(chkGlucose_intolerance.isChecked()) userList.add("intolerancija glukoze");
    if(chkPregnancy.isChecked()){ userList.add("trudnoća"); rfResult.setPregnancyTrue();}
    if(chkPotential_pregnancy.isChecked()) userList.add("potencijalna trudnoća");
    if(chkHypercalcemia.isChecked()) userList.add("hiperkalcemija");
    if(chkHyperkalemia.isChecked()) userList.add("hiperkaliemija");
    if(chkAthlete.isChecked()) userList.add("sportaši");
    if(chkPulmonary_disease.isChecked()) userList.add("plućna opstruktivna bolest");
    if(chkTachyarrhythmia.isChecked()) userList.add("tahiaritmija");
    if(chkHeart_failure.isChecked()) userList.add("zatajenje srca");
    if(chkGout.isChecked()) userList.add("giht");
    if(chkAsthma.isChecked()) userList.add("astma");
    if(chkAV_block.isChecked()) userList.add("A-V blok (2. ili 3. stupanj)");
    if(chkAngioneurotic_edema.isChecked()) userList.add("angioneurotski edem");
    if(chkRenal_arteries_narrowing.isChecked()) userList.add("stenoza bubrežnih arterija");
}
```

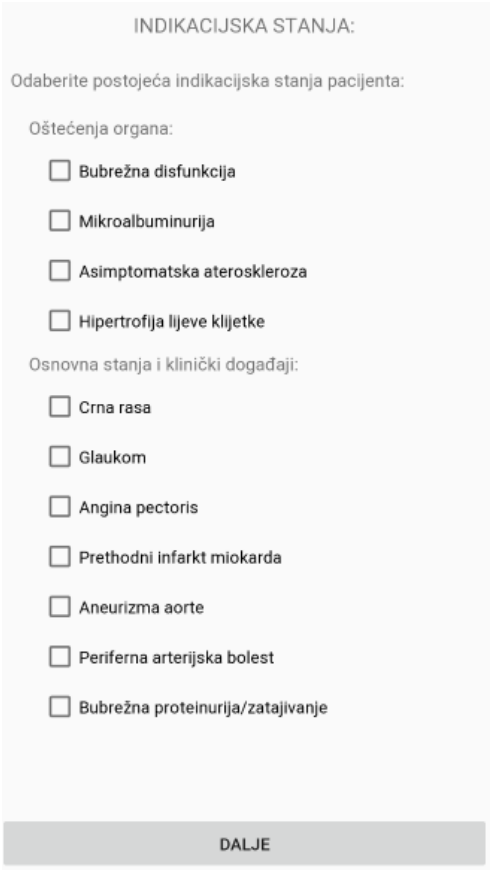
Slika 4.26 *Punjenje korisničke liste kontraindikacija*

```
private void setValidMedicaments() {
    //PUNJENJE USER LISTE
    fillUserList();
    medValidation = new MedicamentValidation();
    //region ELIMINACIJA LIJEKOVA
    //AKO JE PRONADENA BILO KOJA KONTRAINDIKACIJA LIJEK NIJE VALJAN
    for(String userContraindication: userList){
        if(ThiazideDiuretic.findContraindication(userContraindication)){
            medValidation.setDIURETICvalid(false);
            break; } }
    for(String userContraindication: userList){
        if(BetaBlocker.findContraindication(userContraindication)){
            medValidation.setBBvalid(false);
            break; } }
    for(String userContraindication: userList){
        if(CAblocker.findContraindication(userContraindication)){
            medValidation.setCAvalid(false);
            break; } }
    for(String userContraindication: userList){
        if(ACEI.findContraindication(userContraindication)){
            medValidation.setACEIvalid(false);
            break; } }
    for(String userContraindication: userList){
        if(ARB.findContraindication(userContraindication)){
            medValidation.setARBvalid(false);
            break; } }
    //endregion
}
```

Slika 4.27 *Eliminacija neprikladnih lijekova*

Kada se korisnička lista popuni i postave svi atributi objekta MedicamentValidation klase, tada će se u bazi podataka ažurirati atribut „pregnancy“ u *Risk Factor Results* entitetu, a zatim će se proslijediti RiskFactorResult objekt i MedicamentValidation objekt u iduću aktivnost, te pokrenuti iduća aktivnost.

Sada je pokrenuta aktivnost za odabir indikacija pod nazivom IndicationsActivity, a ona ima opciju da skrije odjeljak s oštećenjima organa označenim sa „OO“ u tablici 2.6, ako korisnik u aktivnosti za odabir čimbenika rizika nije odabrao oštećenje organa. Izgled aktivnosti prikazan je slikom 4.28.



Slika 4.28 *Aktivnost za odabir indikacija*

U ovoj aktivnosti je najvažnije napomenuti da se ovdje dohvaćaju objekti RiskFactorResult te MedicamentValidation klasa. Ovdje je stvorena i lista lijekova pod nazivom medicamentList. Lijekovi se iz baze podataka dohvaćaju na isti način, ali ovaj puta se dohvaćaju samo oni kojima su prikladni po MedicamentValidation objektu, odnosno oni čija vrijednost atributa iznosi true. Ovdje je također stvorena korisnička lista userList koja će ovaj puta sadržavati korisnikove indikacije. Klik na gumb „DALJE“ će prvo popuniti userList u ovisnosti o odabranim checkBox-ovima i nekim čimbenicima rizika RiskFactorsResult objekta kao i u prošloj aktivnosti. Slijedi

popunjavanje liste prikladnih lijekova ovisno o MedicamentValidation objektu. U listu lijekova se pohranjuju samo lijekovi koji nisu eliminirani. Aplikacija prvo provjeri je li lijek prikladan, ako je, onda za svaku indikaciju iz korisničke liste pretraži indikacije prikladnog lijeka pomoću metode findIndication() iz klase Medicament. Zatim pohrani lijek. Svaki puta kada metoda findIndication() kada pronađe indikaciju koja se podudara s onom iz korisničke liste, povećava varijablu indication_counter. Taj indication_counter će omogućiti prikaz broja podudarnih indikacija određenog lijeka, da bi se lakše odredilo koji lijek je prikladniji, te da bi se lijekovi sortirali po broju podudaranja indikacija. Slika 4.29 prikazuje metodu findIndication() iz klase Medicament, te dio metode za popunjavanje liste prikladnih lijekova. Slika 4.29 prikazuje samo uvjet za unos tiazidskog diuretika u listu, te prebrojavanje njegovih indikacija, ali taj uvjet se ponavlja za svaki od pet lijekova.

```
public void findIndication(String wanted){
    for(String item:Indications){
        if (item.matches(wanted)){
            indication_counter++;
            return;
        }
    }
}
```

```
private void fillMedicamentList(){
    medicamentList.clear();
    //AKO JE LIJEK PRIKLADAN, PREBROJI PODUDARNE INDIKACIJE
    if(medValidation.getDIURETICvalid()){
        ThiazideDiuretic.resetIndicationCounter();
        for(String userIndication: userList){
            ThiazideDiuretic.findIndication(userIndication);
        }//TE POHRANI LIJEK
        medicamentList.add(ThiazideDiuretic); }
```

Slika 4.29 *Metoda za pronalazak indikacije i dio metode fillMedicamentList() uz prebrojavanje broja indikacija lijeka*

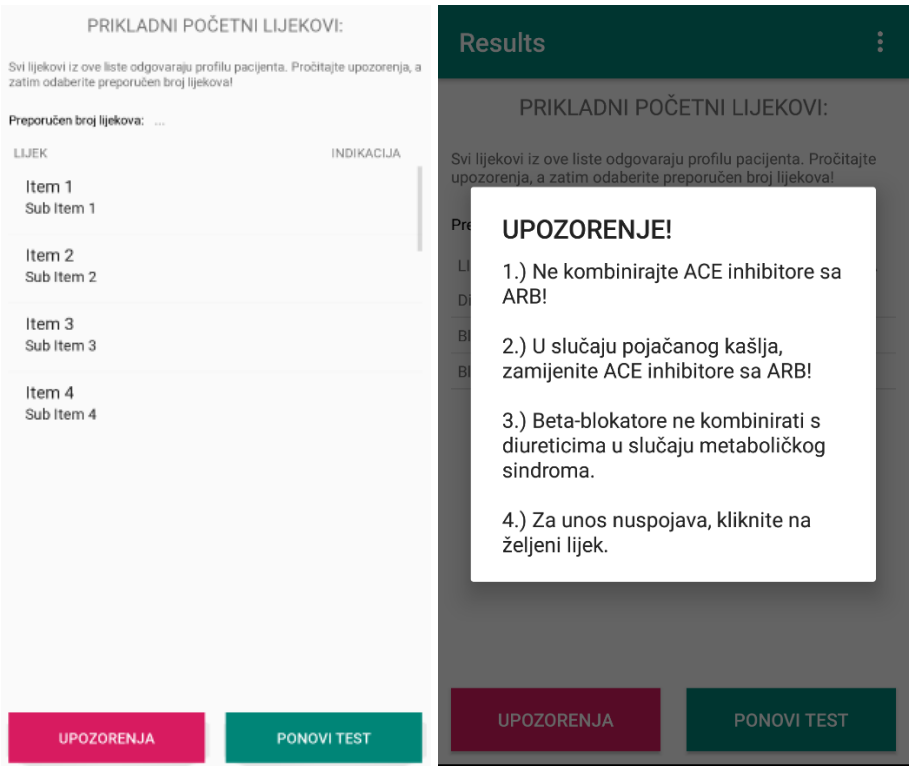
Nakon što se lista lijekova popuni svim prikladnim lijekovima, svi elementi liste se kopiraju u polje, a to polje se sortira po broju indikacija lijeka. Zatim se lista lijekova opet popunjava po polju, što znači da će ovaj puta i onda biti sortirana po broju indikacija. Stvaramo klasu SimplifiedMedicament koja za razliku od klase Medicamet ne sadrži listu indikacija i listu kontraindikacija, već samo naziv, ID, i brojač indikacija lijeka. Stvorena je lista SimplifiedMedicament objekata i popunjena je po sortiranoj listi Medicament objekata. Zatim je ta lista „pojednostavljenih lijekova“ zapisana u bazu podataka pod entitet *Valid Prescriptions* koja za primarni ključ ima korisnikov ID. Također se sada opet pod korisnikov atribut testTaken postavlja vrijednost true. Slijedi pokretanje posljednje aktivnosti upitnika. Slika 4.30 prikazuje metodu za pohranu prikladnih pojednostavljenih zapisa lijekova u bazu podataka, te postavljanje korisničkog atributa testTaken .

```
private void saveResulnDatabase() {
    simplifiedMedicamentList.clear();
    //KREIRANA SIMPLIFIED LISTA DA IZBACIMO INDIKACIJE I KONTRAINDIKACIJE!!!
    for(Medicament medicament: medicamentList){
        simplifiedMedicamentList.add(new SimplifiedMedicament(medicament.getID(),
                medicament.getName(), medicament.getIndication_counter()));
    }
   DatabaseReference databasePrescriptionRef = FirebaseDatabase.getInstance().
           getReference( path: "Valid Prescriptions/" + user.getUid().toString());
   databasePrescriptionRef.setValue(simplifiedMedicamentList);
   DatabaseReference databaseUserRef =  FirebaseDatabase.getInstance()
           .getReference( path: "Users/"+user.getUid());
   databaseUserRef.child("testTaken").setValue(true);
}
```

Slika 4.30 *Popunjavanje pojednostavljene liste prikladnih lijekova i njezin zapis u bazu podataka uz postavljanje testTaken atributa*

Sada je pokrenuta aktivnost za prikaz rezultata pod nazivom FinalActivity. Sadrži listu SimplifiedMedicament objekata koju dohvaća iz entiteta *Valid Prescriptions* iz baze podataka. To je lista svih prikladnih lijekova, s pripadajućim brojem indikacija. Za prikaz te liste je korištena kontrola ListView. Također se dohvaća atribut therapy number za broj potrebnih lijekova iz entiteta *Risk Factor Results*. Ako je preporučeni broj lijekova viši od broja lijekova koji su prikladni za korisnika, korisniku se prikazuje savjet da razmisli o drugim antihipertenzivnim lijekovima.

Nakon dohvaćanja liste prikladnih lijekova, popunjava se listView kontrola i ona izlistava sve prikladne lijekove poredane po broju indikacija. S obzirom na to da slika 2.8 prikazuje moguće kombinacije antihipertenziva, postoji gumb za upozorenje kojeg korisnik treba pročitati prije odabira lijeka. Klik na taj gumb korisniku daje savjete poput, na primjer, koje lijekove da ne kombinira. Izgled ove aktivnosti zajedno s dijalogom koji iskače pritiskom na „UPOZORENJE“ su prikazani slikom 4.31.



Slika 4.31 *Izgled aktivnosti za prikaz rezultata i upozorenja*

Važno je napomenuti da ako korisnik izađe iz aplikacije, ako ne klikne na „PONOVI TEST“ kada idući puta uđe, otvorit će mu se opet ova aktivnost. Dakle aktivnost za prijavu će u ranije spomenutoj metodi setAppropriateUI() provjeriti treba li korisnik lijek i je li ispunio upitnik. Ako je upitnik ispunjen, a lijek mu nije potreban, bit će prebačen na AppropriateTherapyActivity, a ako je korisnik ispunio upitnik i potrebna mu je farmakološka terapija, bit će prebačen na FinalActivity. Ako ga nije ispunio, ili ako je odabrao „PONOVI TEST“ onda se pokreće PressureActivity.

Ako korisnik odabere jedan element listView-a, pomoću onItemClickListener() metode mu se omogući pokretanje aktivnosti za unos nuspojava. Njoj se prenose informacije o odabranom elementu liste, odnosno SimplifiedMedicament objektu koji je odabran. Slika 4.32 prikazuje navedenu metodu.

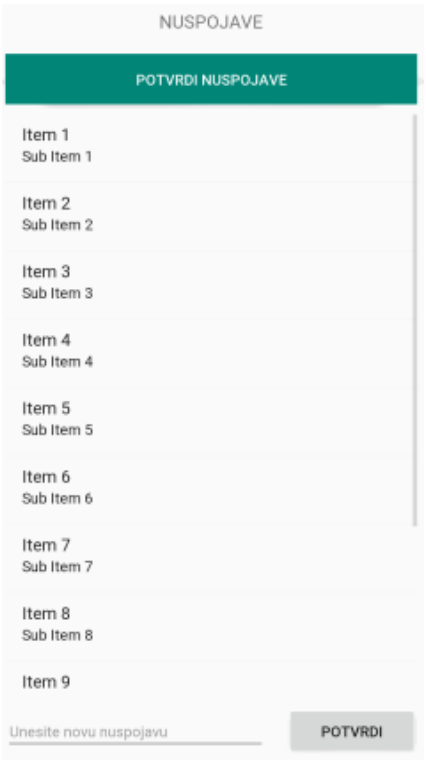
```
listView.setOnItemClickListener((parent, view, position, id) → {
        SimplifiedMedicament simplifiedMedicament = simplifiedMedicamentList.get(position);
        Intent explicitIntent = new Intent(getApplicationContext(), SideEffectsActivity.class);
        explicitIntent.putExtra(FinalActivity.SIMPLIFIED_MEDICAMENT_KEY, simplifiedMedicament);
        startActivity(explicitIntent);
});
```

Slika 4.32 *Metoda za klik na element liste*

4.4. Izrada aktivnosti za unos nuspojava

Posljednja aktivnost aplikacije se zove SideEffectsActivity i radi nje je u bazi podataka izrađen entitet *Side Effects* koji sadrži listu nuspojava za svaki od pet lijekova. Ova aktivnost preko Intenta dohvaća odabrani lijek listView kontrole iz prošle aktivnosti, te iz baze podataka ovisno o odabranom lijeku povlači njegove nuspojave iz entiteta *Side Effects* i puni svoj listView i listu nuspojava.

Tada se stvara lista korisnikovih nuspojava, i svaki puta kada on klikne na jednu nuspojavu ponudi mu se mogućnost da ju unese u listu svojih nuspojava, pritiskom na „Da“ on ju dodaje na svoju listu. Kada klikne na gumb „POTVRDI NUSPOJAVE“, sve nuspojave iz korisničke liste se zapisuju u bazu podataka u novi entitet *User Side Effects* pod korisnikov ID, te pod ID odabranog lijeka. Korisnik također ima mogućnost unijeti novu nepostojeću nuspojavu u bazu podataka za lijek koji je odabrao, a ta nuspojava će se odmah prikazati u listi nuspojava na ekranu i korisnik će odmah moći i nju odabrati. Izgled ove aktivnosti je prikazan na slici 4.33.



Slika 4.33 *Izgled aktivnosti za odabir nuspojava*

Još je važno napomenuti da su problemi poput dodavanja istih nuspojava više puta ili dodavanja nove postojeće nuspojave onemogućeni.

5. ISPITIVANJE RADA I ANALIZA REZULTATA APLIKACIJE

Nakon izrade aplikacije potrebno je i prikladno ispitati njezinu funkcionalnost i ispravnost. Izvršena su tri različita ispitivanja, gdje u prvom neće biti potrebna farmakološka terapija, dok u druga dva hoće.

5.1. Ispitivanje rada aplikacije

U prvom ispitivanju su unesene vrijednosti tlakova 135/85. Za čimbenike rizika je odabrana šećerna bolest bez ostalih čimbenika rizika. Ovi parametri bi u tablici 2.4 trebali rezultirati terapiji u 2. stupcu i 3. retku. Ovaj ispit ne bi smio imati mogućnost nastaviti dalje na farmakološki upitnik.

U drugom ispitivanju je unesena vrijednost tlaka 140/80, što bi trebalo rezultirati arterijskom hipertenzijom prvog stupnja iako dijastolički tlak ne odgovara hipertenziji. Za čimbenike rizika odabrani su abdominalna pretilost, te starija dob. Ovi parametri bi u tablici 2.4 trebali rezultirati terapiji u 3. stupcu i 2. retku. Od dodatnih kontraindikacija su odabrani angioneurotski edem i astma. Od dodatnih indikacija je odabrana periferna arterijska bolest. Treba primijetiti da su u čimbenicima rizika odabrani starija dob i pretilost, što su također indikacije nekih lijekova.

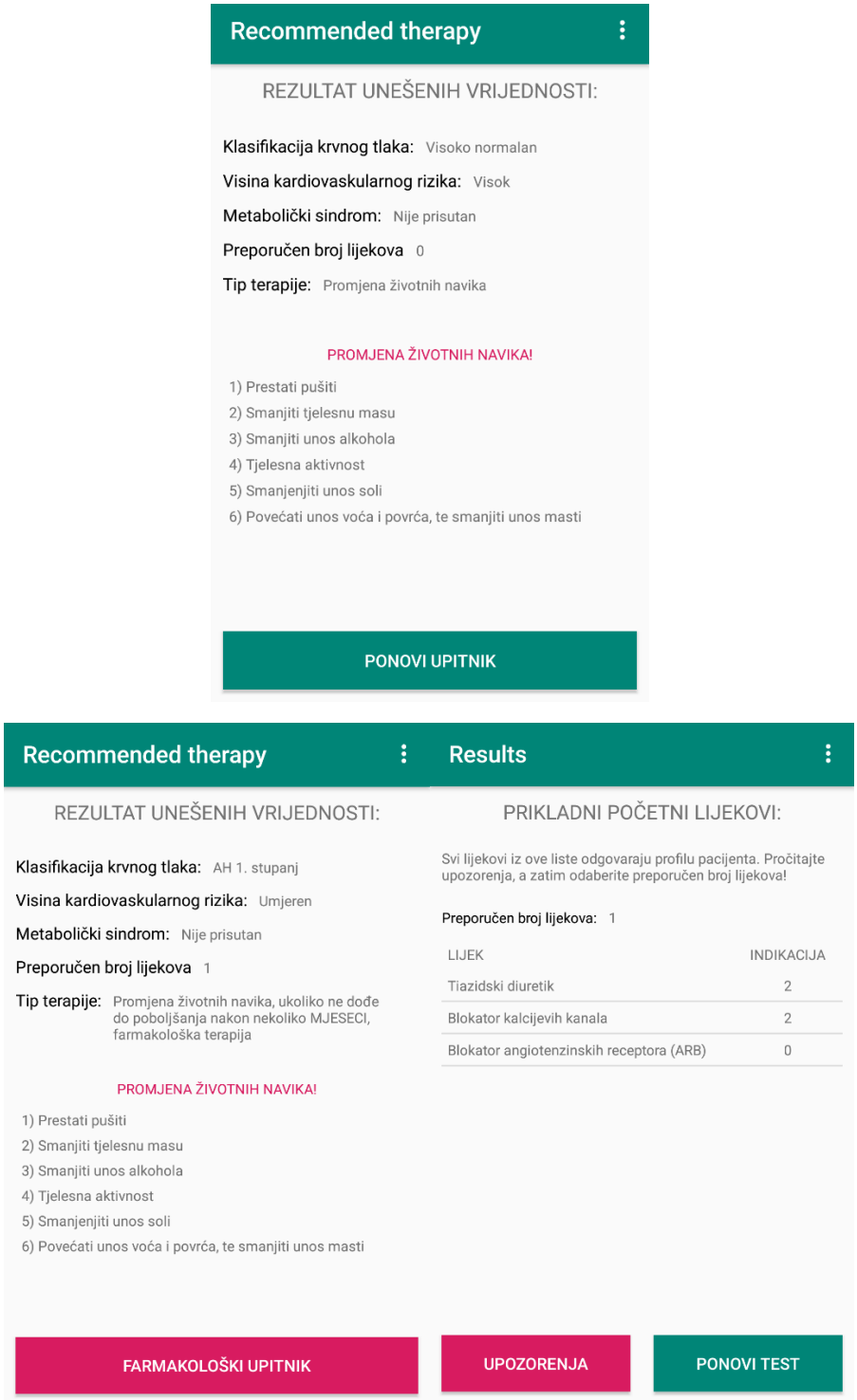
U treće ispitivanje je unesena vrijednost od 160/100 uz dislipidemiju i povišenu glukozu natašte. Ova dva čimbenika rizika u kombinaciji s razinom tlaka iznad visoko normalne trebaju rezultirati prepoznavanjem metaboličkog sindroma. Ovi parametri bi u tablici 2.4 trebali rezultirati terapiji u 4. stupcu i 3. retku. Od dodatnih kontraindikacija su izabrani angioneurotski edem, giht i nepodnošenje glukoze, ali treba uočiti da je metabolički sindrom također kontraindikacija ili indikacija nekih lijekova, te da korisnik nije odabrao stariju dob, mlađa dob predstavlja indikaciju kod nekih lijekova. Od indikacija je odabrana angina pectoris. Unesena je nova nuspojava i odabrane su 2 postojeće.

5.2. Analiza rezultata

Prvo ispitivanje je očekivano preporučilo nefarmakološku terapiju, bez mogućnosti nastavka upitnika. Dodijeljeni ukupni kardiovaskularni rizik je visok, a metabolički sindrom nije prisutan. Prilikom izlaska iz aplikacije i ponovnog ulaska, odmah su se učitali rezultati testa s opcijom ponavljanja testa.

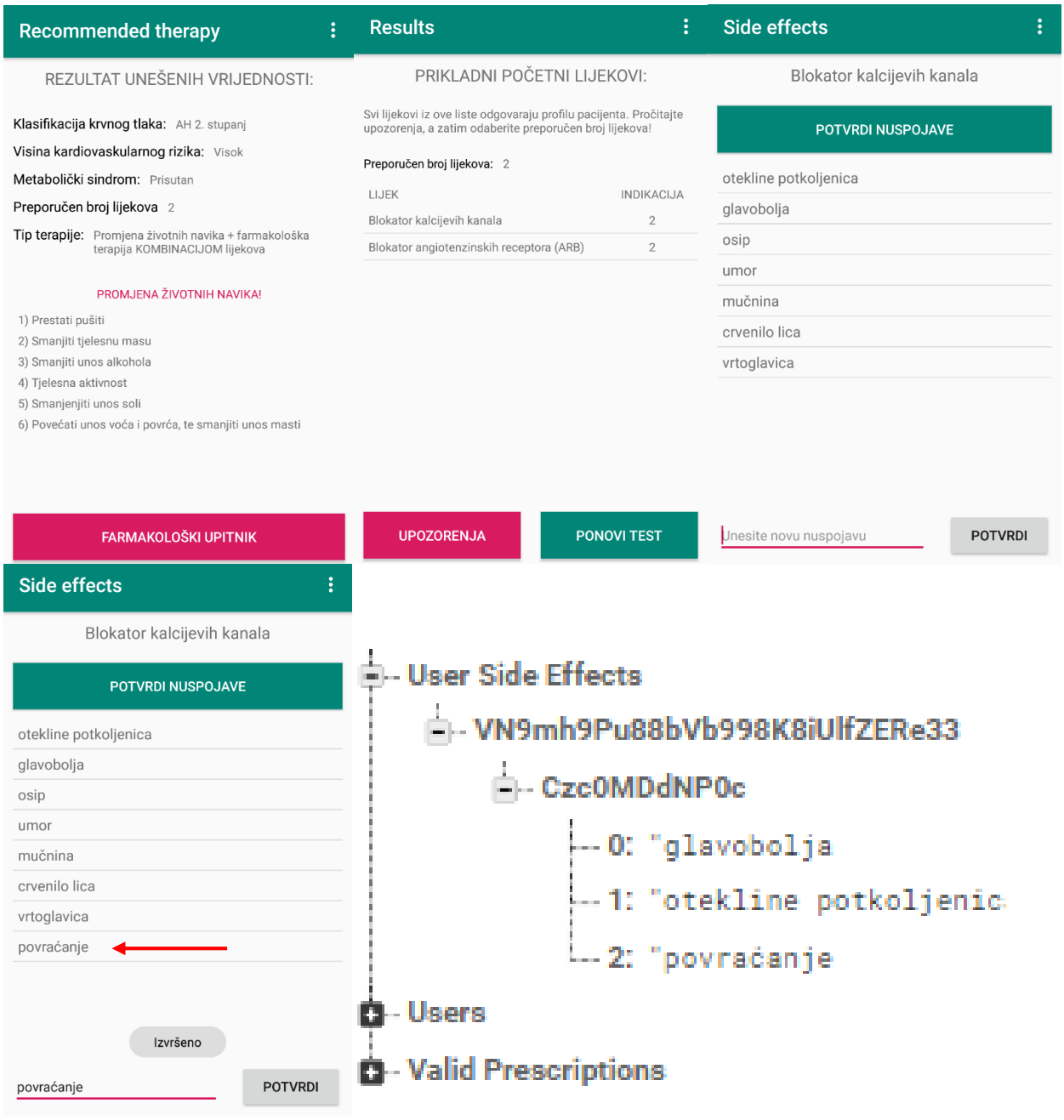
Drugo ispitivanje je preporučilo promjenu životnih navika na nekoliko mjeseci, a zatim farmakološku terapiju kao i očekivano po tablici 2.4 Ukupni rezultat nakon odabranih kontraindikacija i indikacija je ispisao tri prikladna lijeka, te preporučio uzimanje jednog od

prikladnih. Lijekovi su poredani silazno po broju indikacija. Slika 5.1 prikazuje rezultate prvog i drugog ispitivanja uz prikaz prikladnih lijekova drugog ispitivanja.



Slika 5.1 *Rezulati prvog i drugog ispitivanja*

Treće ispitivanje je prepoznalo metabolički sindrom, odredilo visoku razinu kardiovaskularnog rizika, te preporučilo uzimanje dva lijeka. Dodana je i nova nuspojava koja se odmah učitala na ekran. Odabiranjem nuspojava se u bazi podataka stvorio čvor s korisnikovim nuspojavama za odabrani lijek. Slika 5.2 prikazuje rezultate trećeg ispitivanja.



Slika 5.2 *Rezulati trećeg ispitivanja*

5.3. Usporedba rezultata ispitivannja

Da bi se imao bolji uvid u rezultate ispitivanja, izrađene su tablice 5.3 i 5.4 koje prikazuju usporedbu unesenih parametara i dobivenih rezultata. Tablica 5.3 prikazuje ulazne podatke prethodna 3 ispitivanja, dok tablica 5.4 prikazuje dobivene rezultate.

Tablica 5.3 *Ulazni podaci*

Vrijednosti tlakova	Rizični čimbenici	Dodatne kontraindikacije	Dodatne indikacije
135/85	- Šećerna bolest	/	/
140/80	- Pretilost - Starija dob	- Astma - Angioneurotski edem	- Periferna arterijska bolest
160/100	- Dislipidemija - Povišena glukoza natašte	- Angioneurotski edem - Giht - Nepodnošenje glukoze	- Angina pectoris

Tablica 5.4 *Dobiveni rezultati*

Klasifikacija krvnog tlaka	Visina kardiovaskularnog rizika / Metabolički sindrom	Preporučena terapija	Preporučeni broj lijekova / Preporučeni lijekovi (Broj indikacija)
Visoko normalan	Visok / Nije prisutan	Promjena životnih navika	0
AH 1. stupanj	Umjeren / Nije prisutan	Promjena životnih navika, ukoliko ne dođe do poboljšanja, nakon nekoliko mjeseci farmakološka terapija	1/ Tiazidski diuretik (2), blokator kalcijevih kanala(2), ARB(0)
AH 2. stupanj	Visok / Prisutan	Promjena životnih navika + farmakološka terapija kombinacijom lijekova	2/ Blokator kalcijevih kanala(2), ARB(2)

Vidljivo je da su sva tri ispitivanja dala očekivane rezultate prema tablicama iz poglavlja 2. Tablica 5.3 prikazuje unesene vrijednosti krvnih tlakova, sve odabrane rizične čimbenike, te sve dodatno odabrane indikacije i kontraindikacije. Tablica 5.4. prikazuje klasifikaciju krvnog tlaka, visinu kardiovaskularnog rizika ovisno o unesenoj vrijednosti tlaka te odabranim čimbenicima rizika. Također prikazuje prisutnost metaboličkog sindroma, preporučenu terapiju, preporučen broj lijekova, te sve prikladne lijekove uz broj podudarnih indikacija za provedena ispitivanja.

6. ZAKLJUČAK

Cilj završnog rada bio je izrada mobilne Android aplikacije za pomoć pri izboru prikladnog lijeka. Kao što navodi literatura, čak jedna trećina stanovništva na Zemlji boluje od povišenog krvnog tlaka, dok je 2010. godine hipertenzija uzrokovala čak oko 7 000 000 smrtnih slučaja. Zbog toga je uzet primjer arterijske hipertenzije. Osnovni zadaci ove aplikacije su pružiti potporu liječniku za izbor između farmakološke i nefarmakološke terapije, a zatim mu eliminirati sve neprikladne lijekove za liječenje arterijske hipertenzije pomoću kontraindikacija. Liječnik prije toga za svakog pacijenta može napraviti korisnički račun unoseći njegovu adresu elektroničke pošte i lozinku. Također je omogućeno brojanje indikacija, odnosno stanja u kojima lijeku treba dati prednost, i prikaz njihovog broja. Liječnik također ima mogućnost odabira određenog lijeka, te unos nuspojava ako do njih dođe, a uvijek ih može provjeriti u bazi podataka. Ako dođe do nove nepostojeće nuspojave, nju može unijeti u bazu podataka i tako svima ostalima omogućiti njezin odabir. Pri odabiru prikladnog lijeka, liječnik također ima upozorenja koja mu govore koje lijekove ne smije kombinirati.

Predloženo i ostvareno programsko rješenje mobilne aplikacije pokazalo se ispravnim i učinkovitim, a ispitivanjem funkcionalnosti aplikacije utvrđeno je da aplikacija izvršava ono što se od nje očekuje prema svim navedenim tablicama u poglavlju s teorijskom podlogom o problemu. Ova aplikacija bi mogla daljnjom razradom teorijske podloge i dodavanjem novih funkcionalnosti postati još preciznija i korisnija.

7. LITERATURA

[1] Čitate li upute o lijekovima?, https://www.plivazdravlje.hr/aktualno/clanak/13252/Citate-li-upute-o-lijekovima.html, zadnja posjeta 03.09.2019.

[2] Što je krvni tlak?, https://www.plivazdravlje.hr/tekst/clanak/7926/Sto-je-krvni-tlak.html, zadnja posjeta 22.06.2019.

[3] Lj.F. Pranjković, Uloga ljekarnika u liječenju hipertenzije, Medicus, Vol. 23, Supl. 1, br. 1 Zagreb 2014.

[4] Hrvatsko društvo za hiperzenziju, 2007. Smjernice za dijagnosticiranje i liječenje arterijske hipertenzije

[5] H. Adamu Kakudi, C. Kiong Loo, F. Ming Moy , N. Masuyama, K. Pasupa, Diagnosing Metabolic Syndrome Using Genetically Optimised Bayesian ARTMAP, IEEE Access, Vol. 7, str. 8437-8438, studeni 2018.

[6] Msd priručnici, Arterijska hipertenzija, http://www.msd-prirucnici.placebo.hr/msd-prirucnik/kardiologija/arterijska-hipertenzija, zadnja posjeta 20.06.2019.

[7] D. Gyamf, C. Obirikorang, E. Acheampong, K. Owusu Danquah, E. Adu Asamoah, F. Zarah Liman, E.Nsenbah Batu, Prevalence of Pre-hypertension and Hypertension and its Related Risk Factors among Undergraduate Students in a Tertiary Institution, Ghana, Alexandria Journal of Medicine, Vol. 54, No. 1, str. 475-476, prosinac 2018.

[8] Msd priručnici, Visoki krvni tlak, http://www.msd-prirucnici.placebo.hr/msd-za-pacijente/bolesti-srca-i-krvnih-zila/visoki-krvni-tlak, zadnja posjeta 26.06.2019.

[9] 2013 ESH/ESC Guidelines for the management of arterial hypertension. Dostupno na: https://www.escardio.org/Guidelines/Clinical-Practice-Guidelines/Arterial-Hypertension-Management-of, zadnja posjeta 27.06.2019.

[10] Top ten best blood pressure apps, https://www.medicalnewstoday.com/articles/320095.php, zadnja posjeta 27.08.2019.

[11] Google play, ANTI-HYPERTENSIVE DRUGS, https://play.google.com/store/apps/details?id=com.andromo.dev689476.app769578&hl=en_US, zadnja posjeta 28.08.2019.

[12] R.L. Wade, B. Clancey, J. Michaeli, Improvement In Antihypertensive And Cholesterol-Lowering Medication Persistence Using A Mobile Technology Application, Value in Health, Vol. 19, No. 3, - str. A306, svibanj 2016.

[13] Medisafe, https://www.medisafeapp.com/ , zadnja posjeta 06.09.2019.

[14] Firebase Authentication, https://firebase.google.com/docs/auth, zadnja posjeta 28.06.2019.

[15] Structure Your Database, https://firebase.google.com/docs/database/android/structure-data, zadnja posjeta 05.09.2019.

[16] Application Fundamentals, https://developer.android.com/guide/components/fundamentals?hl=en#Manifest, posjećeno zadnja posjeta 28.06.2019.

[17] Layouts, https://developer.android.com/guide/topics/ui/declaring-layout?hl=en, posjećeno zadnja posjeta 28.06.2019.

[18] Introduction to Activities, https://developer.android.com/guide/components/activities/intro-activities?hl=en, zadnja posjeta 05.09.2019.

[19] Understand the Activity lifecycle, https://developer.android.com/guide/components/activities/activity-lifecycle?hl=en, zadnja posjeta 28.06.2019.

ŽIVOTOPIS

David Bilić rođen je 09.03.1998 u Slavonskom Brodu, Hrvatska. Stanuje u Slavonskom Brodu. Godine 2003. upisuje OŠ Bogoslav Šulek u Slavonskom Brodu. Nakon odličnog uspjeha u osnovnoj školi, 2012. upisuje se u Tehničku školu Slavonski Brod gdje je sudjelovao na državnom natjecanju iz Osnova Elektrotehnike. 2016. godine, nakon odličnog uspjeha ostvaruje izravan upis na preddiplomski sveučilišni studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijeku. Od programskih jezika poznaje strukture C-a, C++-a, C#-a, Jave, HTML-a, CSS-a. Izvrsno poznaje engleski jezik. Od vrlina navodi marljivost i želju za napretkom.

SAŽETAK

Cilj završnog rada je razrada modela mobilne Android aplikacije s ciljem pružanja potpore za izbor najprikladnijeg lijeka, što je pokazano za arterijsku hipertenziju korištenjem metode eliminiranja i prioritetnog sortiranja prikladnih lijekova. Aplikacija je ostvarena u programskom jeziku Java pomoću razvojnog okruženja Android Studio. Baza podataka je ostvarena pomoću platforme Firebase, te njezine dvije usluge – Authentication, te Realtime Database. Korisniku je omogućen unos vrijednosti krvnih tlakova, te čimbenika rizika koji određuju prikladan tip terapije. Zatim, ako je potrebna farmakološka terapija, korisnik unosi svoje dodatne kontraindikacije i indikacije, te dobiva listu prikladnih lijekova sortiranih po broju indikacija. Aplikacija također generira osnovna upozorenja za kombiniranje lijekova. Korisnik na kraju ima mogućnost odabira svojih nuspojava, te unosa novih, nepostojećih, u bazu podataka.

Ispitivanjem funkcionalnosti aplikacije za tri slučaja koji predstavljaju moguće ulazne parametre bolesnika, utvrđeno je da aplikacija omogućuje navedene funkcionalnosti i daje ispravne rezultate, odnosno omogućuje ispravan izbor lijeka za unesene čimbenike rizika, te ostale ulazne parametre. Ova aplikacija može se nadograditi dodatnim funkcionalnostima i aktivnostima, te drugim skupinama bolesti i lijekova koji bi proširili njezinu primjenu.

Ključne riječi: Android, aplikacija, arterijska hipertenzija, indikacije, kontraindikacije, nuspojave

ABSTRACT

The aim of this final paper is to develop a model of a mobile Android application in order to support the selection of the most appropriate drug, as demonstrated for arterial hypertension using the method of elimination and prioritization of suitable drugs. The application is written in the Java programming language in the Android Studio development environment. The database was created using the Firebase platform and its two services - Authentication and Realtime Database. The user is allowed to enter blood pressure values and risk factors that determine the appropriate type of therapy. Then, if pharmacological therapy is required, the user enters their additional contraindications and indications, and receives a list of suitable medications sorted by number of indications. The app also generates basic alerts for combining drugs. In the end, the user has the opportunity to select their side effects and to enter new, nonexistent ones in the database.

By examining the functionality of the application for three cases that represent possible patient input parameters, it was found that the application provides the stated functionality and gives the correct results, that is, enables the correct choice of medicine for the entered risk factors, and other input parameters. This application can be upgraded with additional functionalities and activities, and other disease and drug classes that would expand its application.

Keywords: Android, app, arterial hypertension, contraindications, indications, side effects

PRILOZI (NA DVD-u)

Prilog 1: Završni rad u docx i pdf formatu

Prilog 2: Projekt mobilne aplikacije u Android Studiu