SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I

INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

Smjer Računarstvo

IZRADA MOBILNE APLIKACIJE ZA VOĐENJE EVIDENCIJE PRISUTNOSTI NA NASTAVI

Diplomski rad

Petar Nenadić

Osijek, 2022.

##### SADRŽAJ

[1. UVOD 1](#_Toc119678531)

[1.1. Zadatak diplomskog rada 2](#_Toc119678532)

[1.2. Pregled sličnih aplikacija 3](#_Toc119678533)

[2. PREGLED UPOTRIJEBLJENIH TEHNOLOGIJA 6](#_Toc119678534)

[2.1. Operacijski sustav Android 6](#_Toc119678535)

[2.2. Android Studio 7](#_Toc119678536)

[2.3. Kotlin 8](#_Toc119678537)

[2.4. XML 9](#_Toc119678538)

[2.5. Firebase 10](#_Toc119678539)

[2.6. Senzori i kamera 11](#_Toc119678540)

[2.7. AES algoritam za enkripciju 12](#_Toc119678541)

[3. RAZVOJ APLIKACIJE ZA VOĐENJE EVIDENCIJE PRISUTNOSTI NA NASTAVI 13](#_Toc119678542)

[3.1. Korisnički zahtjevi 13](#_Toc119678543)

[3.2. Dijagram toka 14](#_Toc119678544)

[3.3. Arhitekturalni dizajn sustava 16](#_Toc119678545)

[4. IMPLEMENTACIJA PROGRAMSKOG RJEŠENJA 18](#_Toc119678546)

[4.1. Dodavanje ovisnosti 20](#_Toc119678547)

[4.2. Struktura projekta 20](#_Toc119678548)

[4.3. Aplikacija za studente 23](#_Toc119678549)

[4.4. Aplikacija za profesore 32](#_Toc119678550)

[4.5. Testiranje aplikacija 42](#_Toc119678551)

[5. Zaključak 43](#_Toc119678552)

[LITERATURA 44](#_Toc119678553)

[SAŽETAK 46](#_Toc119678554)

[ABSTRACT 47](#_Toc119678555)

[ŽIVOTOPIS 48](#_Toc119678556)

[PRILOZI 49](#_Toc119678557)

# UVOD

Evidentiranje prisutnosti na nastavi često zahtijeva puno vremena, i sklono je ljudskim greškama i čestim zabilježavanjem netočnih podataka. Također, praćenje prisutnosti po studentu može biti zamoran proces zato što zahtijeva neku vrstu priručnika za praćenje dana prisutnosti. Pojavom pametnih uređaja, koji su jako zgodni za korištenje, ovaj proces se može u potpunosti automatizirati korištenjem prave tehnologije.

Glavni cilj ovog diplomskog rada je razviti aplikaciju baziranu na operativnom sustavu Android koja će ispuniti današnje standarde i zahtjeve korisnika. Za izradu ovog rada koriste se znanja stečena tijekom studiranja na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

Ovaj dokument je sastavljen od dva dijela: teorijski i praktični dio. U teorijskom dijelu navodit će se i objasniti izbori tehnologije korištene za izradu ove aplikacije. Drugo poglavlje, koje nosi naslov *„Pregled upotrijebljenih tehnologija“* sadrži opis svih tehnologija koje su upotrijebljene za izradu aplikacije. Bit će opisani:

* Operacijski sustav Android
* Android Studio
* Kotlin
* XML
* Firebase
* Senzori i kamera
* AES algoritam za enkripciju

Treće poglavlje, koje nosi naslov *„Razvoj aplikacije za vođenje evidencije prisutnosti“,* sadržavat će opisane korisničke zahtjeve na aplikaciju, dijagram toka, te arhitekturalni dizajn sustava.

U praktičnom dijelu, bit će opisan rad aplikacije. Praktični dio aplikacije sastoji se od poglavlja koje nosi naziv *„Implementacija programskog rješenja“*. U ovom poglavlju će biti opisan rad aplikacije kroz četiri potpoglavlja. Prije navođenja potpoglavlja, bit će opisano kreiranje novog projekta u Android Studiju. Prvo potpoglavlje, naziva *„Dodavanje ovisnosti“*, opisuje ovisnosti za programski jezik Kotlin koje su potrebne za izradu aplikacije. Drugo poglavlje, *„Struktura projekta“*, opisivat će sadržaj projekta, dozvole koje je potrebno unijeti u *AndroidManifest.xml* datoteku kako bi aplikacija dobila pristup svim potrebnim funkcijama. Također će biti opisan unos najčešće korištenih elemenata u XML-u, kao što su *ImageView*, *TextView* i *button*. Naposljetku, bit će opisano dodavanje API ključa za komunikaciju između pametnog uređaja i *Firebase*-a. Treće i četvrto potpoglavlje, naslova *„Aplikacija za studente“* i *„Aplikacija za profesore“*, će sadržavati detaljno opisane upute za rad s aplikacijama.

## Zadatak diplomskog rada

Izraditi mobilnu aplikaciju na klijent-server arhitekturi koja će korištenjem suvremenih komunikacijskih tehnologija omogućiti evidenciju prisutnosti studenata na nastavi uz korištenje smartphone uređaja.

## Pregled sličnih aplikacija

1. Attendy - aplikacija za upravljanje prisustvom. Koristi registar za upravljanje prisutnosti te podržava izvoz u razne formate (CSV, PDF, Excel). Potrebno je unositi prisutnost za svakog studenta što oduzima puno vremena. [15]



**Slika 1.2.1.** Logo Attendy aplikacije

1. Presli (Attendance Manager) – aplikacija za upravljanje popisom, omogućuje dodavanje bilješki za bilo koji unos. Također je potrebno unositi prisutnost za svakog studenta. [16]



**Slika 1.2.2.** Logo Presli aplikacije

1. Alora (Attendance Tracker App) – aplikacija za praćenje koja pomaže uštedjeti vrijeme. Omogućuje postavljanje grupa, te napomena. Nema mogućnost automatskog zabilježavanja prisutnosti, kao što je skeniranje QR koda. [17]



**Slika 1.2.3.** Logo Alora aplikacije

1. Self Attendance – aplikacija za praćenje dnevne prisutnosti. Aplikacija je uglavnom izrađena za studente, pa studenti sami moraju unositi kolegije u raspored. Podatke sprema u memoriju uređaja, te omogućuje sinkronizaciju s Google diskom. [18]



**Slika 1.2.4.** Logo Self Attendance aplikacije

1. QR Attendance – aplikacija koja se koristi generiranjem QR kodova kako bi zabilježila podatke o prisutnosti studenata na predavanjima. Koristi IMEI broj pametnog uređaja za evidentiranje prisutnosti, stoga, u slučaju gubitka uređaja, profesor mora unijeti IMEI broj studentovog novog uređaja kako bi se student prijavio na predavanje. [19]



**Slika 1.2.5.** Logo QR Attendance aplikacije

1. QR Attendance Control – aplikacija koja omogućuje kontrolu prisutnosti na događajima, vrijeme dolaska i odlaska. Za svaku prijavljenu osobu se kreira jedinstveni QR kod koji se koristi za prijavljivanje. [20]



**Slika 1.2.6.** Logo QR Attendance Control aplikacije

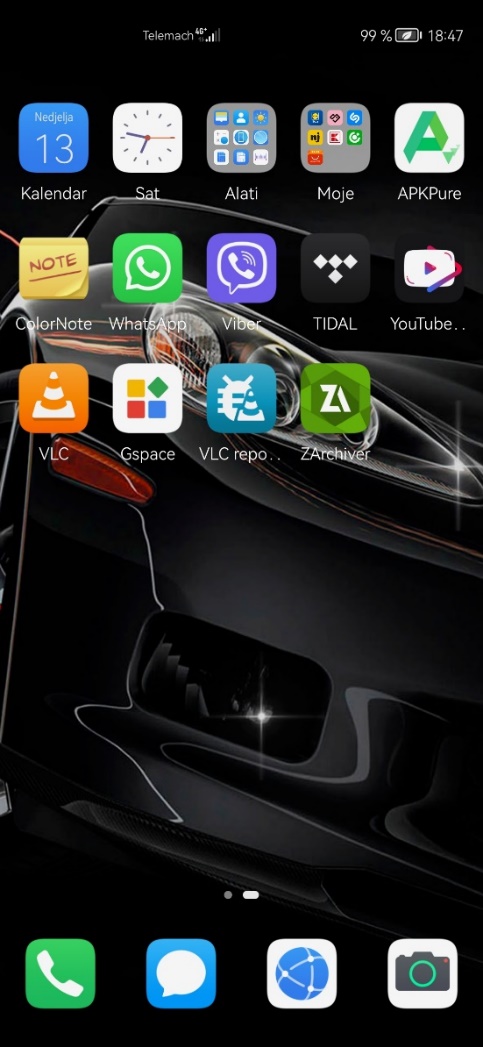
# PREGLED UPOTRIJEBLJENIH TEHNOLOGIJA

U ovom poglavlju opisat će se tehnologije koje su upotrijebljene za razvoj aplikacije.

## Operacijski sustav Android

Operacijski sustav Android je razvila tvrtka Google 2003. godine kako bi se razvijali programi za pametne mobilne uređaje koji imaju ekrane osjetljive na dodir. Verzija koja je korištena u ovom radu je Android 10 (API 29, slika 2.1.1.), s podrškom za minimalni sustav Android 5.0 (Lollipop, API 21).

Uređaji koji rade na operacijskom sustavu Android su već dugo vremena najprodavaniji na tržištu pametnih uređaja. Posljednja verzija Android operacijskog sustava je Android 13 (API 33).

**Slika 2.1.1.** Prikaz sustava Android 10

## Android Studio

Android studio je službeni IDE (engl. *Integrated Development Environment*) za razvoj aplikacija koje se pokreću na operativnom sustavu Android. Trenutno je najpopularnije razvojno okruženje za razvoj Android aplikacija te omogućuje instantno pokretanje koda na uređaju ili emulatoru. Omogućuje podršku za sve pametne uređaje, kao što su: mobiteli, satovi, tableti, televizori...

Android studio je dostupan za preuzimanje na Windows, MacOS i Linux operacijske sustave. Dolaskom na tržište zamijenio je Eclipse ADT te nudi niz značajki koje poboljšavaju produktivnost, a neke od njih su:

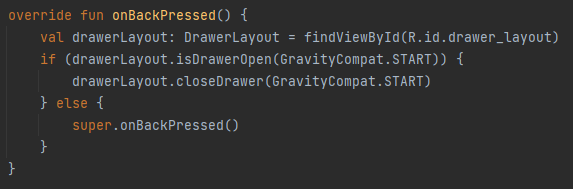
* Razvojni alati
* Brz emulator
* Podrška za programski jezik C++
* Jedinstveno okruženje kompatibilno sa svim uređajima koji pogone Android operativni sustav
* Dugi popis alata za testiranje i okvire

Budući da je za pisanje Android Studija korišten programski jezik Java, koji je ujedno i temelj za pisanje programskog koda za Android uređaje, potrebno je instalirati i najnoviju verziju *Java Development Kit*-a (JDK).

## Kotlin

Kotlin je trenutno jedan od najmodernijih programskih jezika iako je započeo svoj razvoj 2011. godine. To je jezik koji se konstantno razvija i stalno se unaprjeđuje novim svojstvima i funkcionalnostima. Budući da uživa veliku podršku *Google*-a, očekuje se i daljnja podrška za Kotlin, zbog čega ga sve više programera koristi umjesto *Jave*. Može se koristiti i za razvoj *Web* i serverskih aplikacija.

Kotlin je korišten u ovom radu kako bi povezao sve prozore koje korisnik vidi u aplikaciji i dodao razne funkcionalnosti.

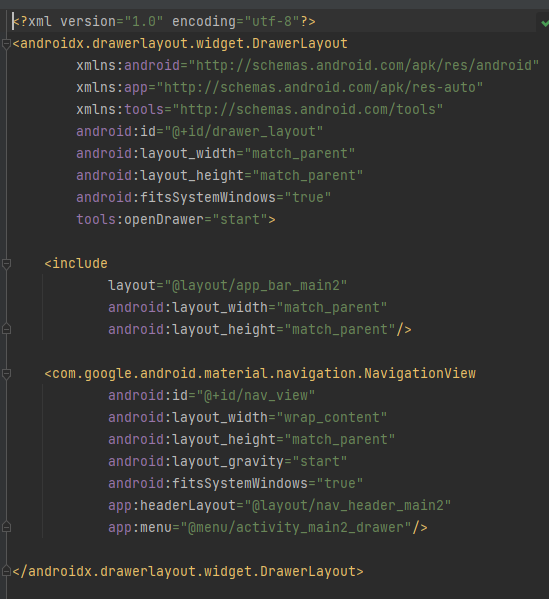


**Slika 2.3.1.** Primjer funkcije u programskom jeziku Kotlin

## XML

XML je jezik koji se koristi za označavanje podataka i dokumenata. Ideja ovog jezika je da se stvori jezik koji će biti moguće pročitati i razumjeti čak i ljudima koji se ne bave programiranjem. Zamišljen je da podatke čuva u tekstualnom obliku i uokviruje ih oznakama kako bi znali o kakvom podatku se radi.

XML naglašava upotrebljivost preko Interneta, jednostavnost i općenitost, ima jaku podršku preko Unicode-a za različite jezike. Iako je namjena XML-a prvenstveno za upotrebu u dokumentima, može se koristiti i za druge namjene, kao što su: razmjena podataka, odvajanje podataka od prezentacije, pohrana podataka i povećana dostupnost podataka. XML je standardizirani jezik, za što se pobrinuo World Wide Web Consortium. U ovom radu je korišten za kreiranje korisničkog sučelja.



**Slika 2.4.1.** Primjer XML koda

## Firebase

Firebase je platforma istoimene tvrtke koja se koristi za razvoj mobilnih i web aplikacija pomoću koje se može brzo izraditi aplikacija i povećati bazu angažiranih korisnika. Platforma uključuje nekoliko dobro integriranih značajki koje se mogu kombinirati, te posjeduje alate za analitiku, pozadinske programe za mobilne uređaje te alate za razvoj i maksimalno povećanje uspjeha aplikacija.



**Slika 2.5.1.** Firebasetablica

Firebase se koristi u ovom radu zato što omogućava pohranu podataka u stvarnom vremenu i što su ti podaci dostupni svugdje u svijetu. Za komunikaciju s Firebase-om je dovoljan pristup Internetu.

## Senzori i kamera

Gotovo svi Android uređaji imaju ugrađeno po nekoliko vrsta senzora pomoću kojih je moguće dobiti podatke o trenutnoj lokaciji, brzini kretanja te ubrzanju. Razne kretnje uređaja, kao što su drmanje i zakrivljenost, pokreću senzore za promatranje trodimenzionalnog položaja u prostoru. U ovom radu će se koristiti senzor pozicije koji koristi velik broj različitih metoda kako bi dostavio aplikaciji informacije o položaju korisnika. U Android Studiju, takve metode se nazivaju dostavljači lokacije (engl. *location providers*)i svaka se može koristiti na različite načine, ovisno o situaciji. Primjer dostavljača lokacije je GPS (engl. *Global Positioning System*) i mrežni dostavljači. Senzori položaja bit će upotrijebljeni u ovom radu kako bi se osigurala maksimalna udaljenost između pametnih uređaja koji skeniraju QR kod i uređaja koji generiraju taj isti kod.

Osim raznih senzora, većina pametnih uređaja ima ugrađene i kamere koje mogu poslužiti u razne svrhe, kao što je snimanje videa ili fotografiranje. U Android Studiju, pristup kameri se omogućuje dopuštenjem koje se mora upisati u *Manifest* datoteku, a to dopuštenje je:

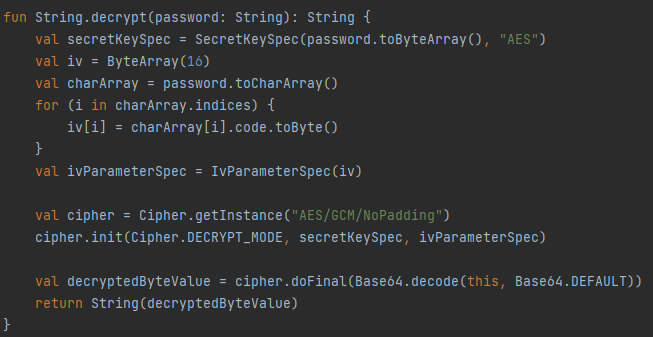
<uses-permission android:name="android.permission.CAMERA"/>.

Senzor kamere će se koristiti u ovom radu kako bi se očitao QR kod prilikom prijave studenta na predavanja.

## AES algoritam za enkripciju

Kako bi se osigurala jedinstvenost podataka, za enkripciju lozinki korišten je AES (engl. *Advanced Encryption Standard*) algoritam za enkripciju. U ovom radu koristi se takozvani Rijndael sustav kojeg je uspostavio Nacionalni institut za standarde i tehnologiju SAD-a (NIST) u 2001. godini. Rijndaelovu blok šifru su razvila dva belgijska kriptografa, Joan Daemen i Vincent Rijmen [14]. Karakteristike AES standarda su sljedeće:

* Šifriranje simetričnim ključem i simetričnim blokom
* 128-bitni podaci, 128/192/256-bitni ključevi
* Jači i brži od Triple DES-a
* Isporučuje punu specifikaciju i pojedinosti o dizajnu
* Softver koji se može implementirati u C programskom jeziku i JAVA-i
* Enkriptiranje korisničkih lozinki, podataka o QR kodu itd.



**Slika 2.7.1.** Primjer funkcije u Kotlinu za enkripciju pomoću AES standarda

# RAZVOJ APLIKACIJE ZA VOĐENJE EVIDENCIJE PRISUTNOSTI NA NASTAVI

U ovom poglavlju će se opisati korisnički zahtjevi na aplikaciju, dijagrami toka za aplikaciju za profesore i studente te arhitekturalni dizajn sustava.

## Korisnički zahtjevi

Prije početka izrade same aplikacije potrebno je postaviti korisničke zahtjeve. Aplikacija služi za prijavljivanje studenata na nastavu i imat će dvije vrste korisnika: profesore i studente. Pošto postoje dvije vrste korisnika, svaki će imati svoju aplikaciju, odnosno, radit će se dvije aplikacije. Jedna aplikacija će biti za profesore, a druga za studente. Svaka od aplikacija će imati svoje zahtjeve.

Aplikacija za studente će imati sljedeće zahtjeve:

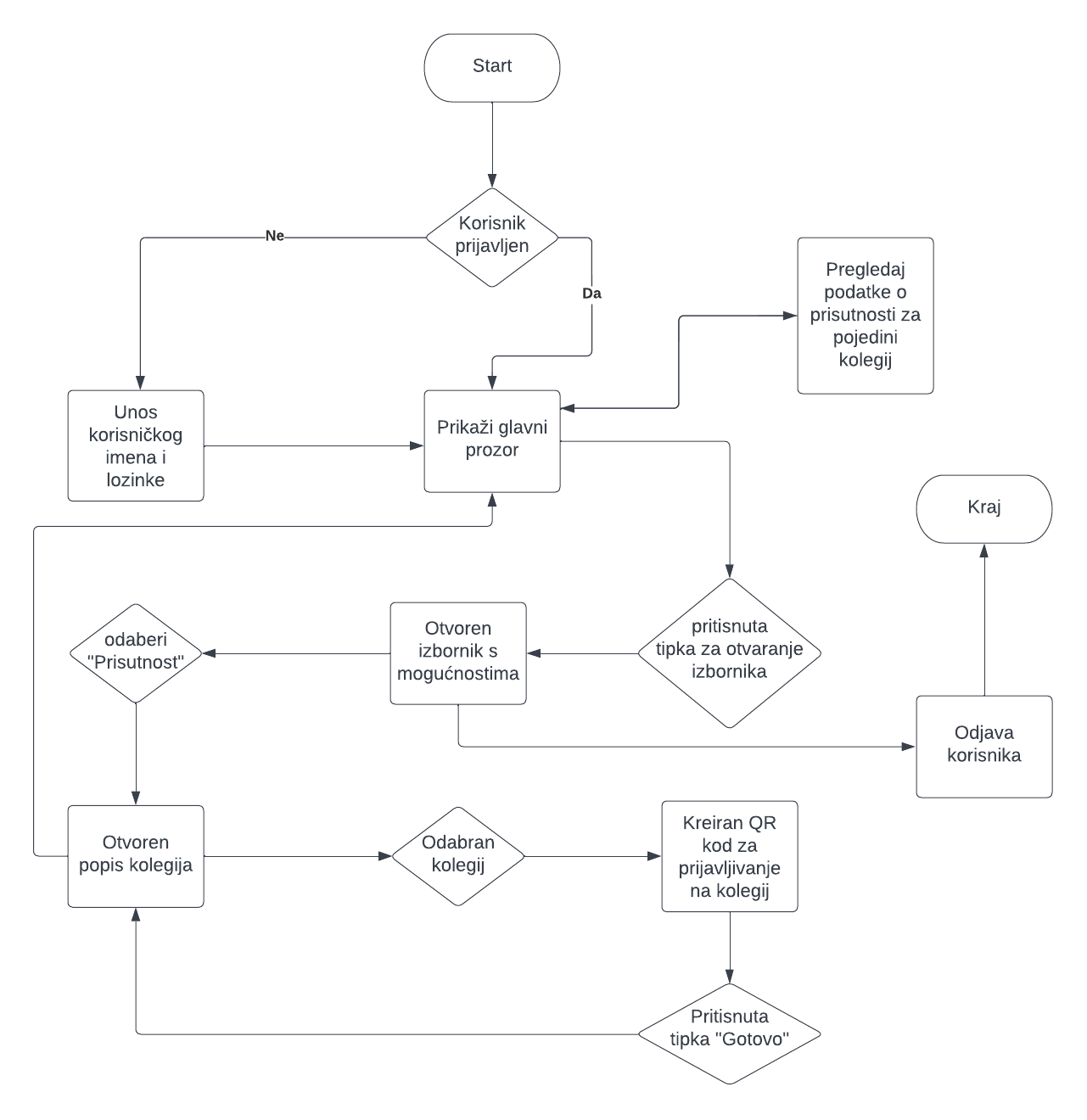
* Prijava u sustav – za prijavu u sustav potrebno je unijeti ispravan ID korisnika i odgovarajuću lozinku
* Promjena lozinke – za slučaj da je korisnik zaboravio lozinku ili želi kreirati svoju lozinku
* Pregled podataka o prisutnosti za pojedini kolegij
* Skeniranje QR koda za prijavu na nastavu
* Prikaz korisničkog profila s podacima o korisniku

Aplikacija za profesore će imati sljedeće zahtjeve:

* Prijava u sustav – za prijavu u sustav potrebno je unijeti ispravan ID korisnika i odgovarajuću lozinku
* Promjena lozinke – za slučaj da je korisnik zaboravio lozinku ili da želi kreirati svoju lozinku
* Kreiranje QR koda – za evidentiranje prisutnosti studenata na pojedini kolegij
* Spremanje popisa u bazu podataka nakon evidentiranja studenata
* Pregled podataka o prisutnosti pojedinih studenata
* Kreiranje tablice s popisom svih studenata i slanje na e-mail
* Prikaz korisničkog profila s podacima o korisniku

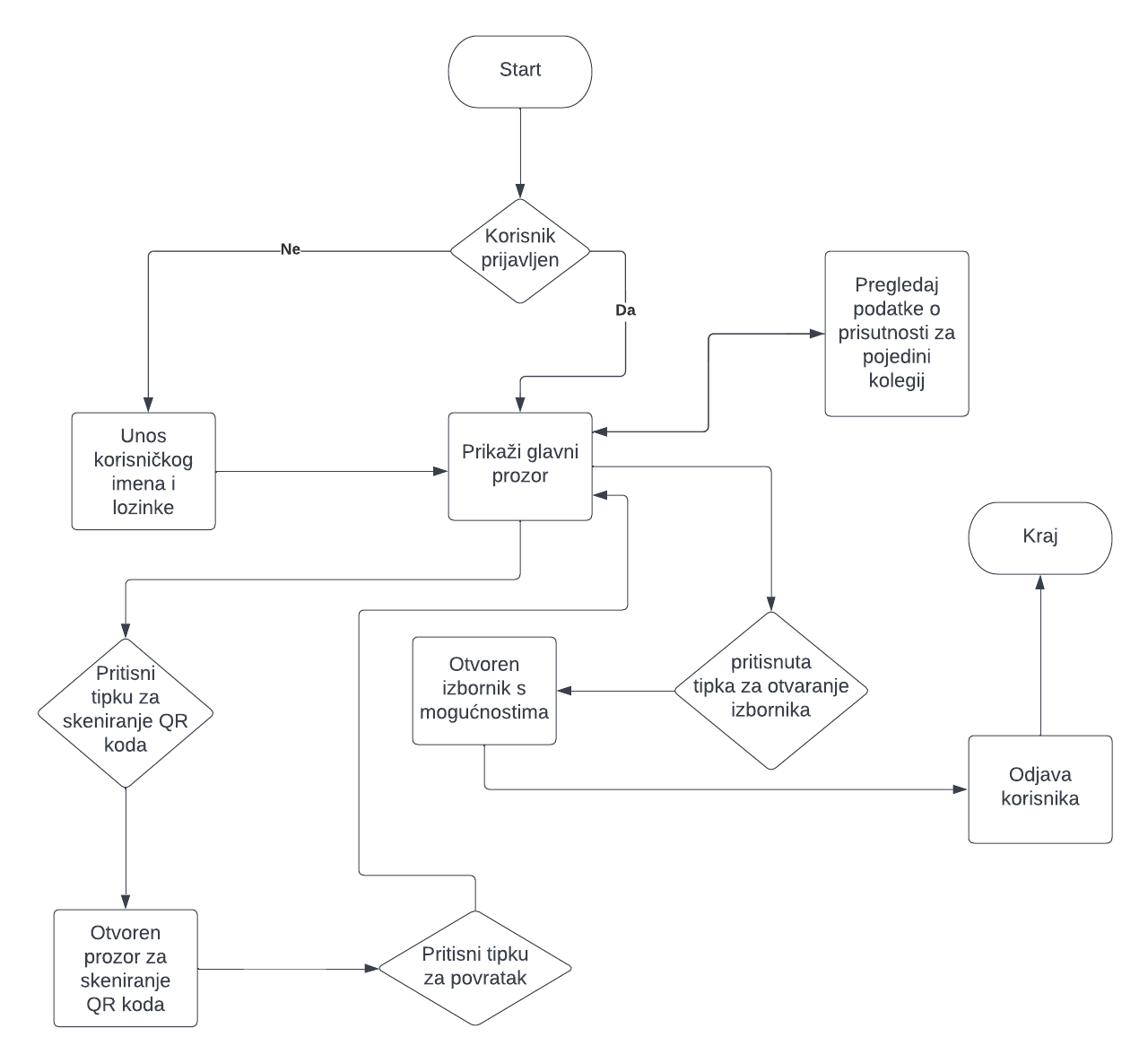
## Dijagram toka

Kako bi bio jasan tijek aplikacije izrađen je dijagram toka. Ovaj dijagram prikazuje sve promjene od pokretanja aplikacije preko korištenja do odjave korisnika. Na slici 3.2.1 prikazan je dijagram toka aplikacije za profesore.



**Slika 3.2.1.** Dijagram toka za aplikaciju za profesore

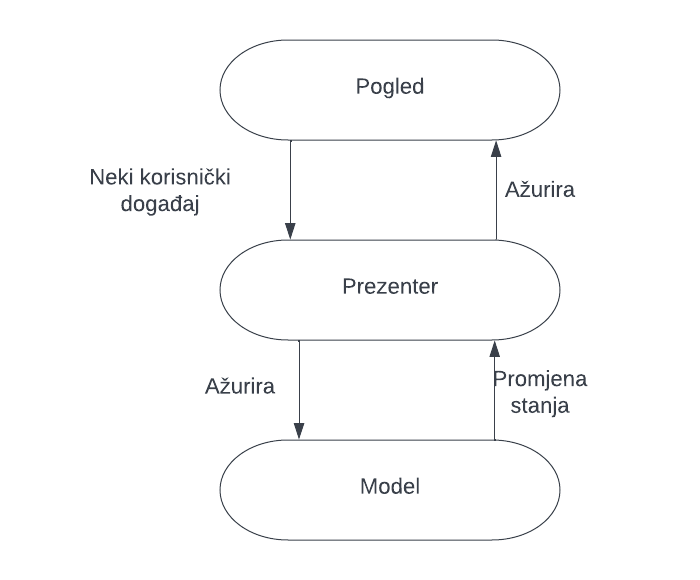
Dijagram toka za aplikaciju za studente prikazan je na slici 3.2.2.



**Slika 3.2.2.** Dijagram toka za aplikaciju za studente

## Arhitekturalni dizajn sustava

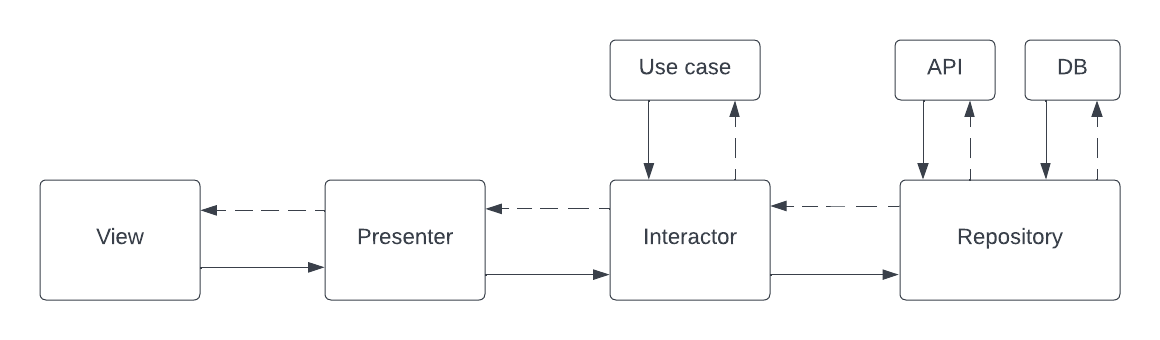
Za izradu aplikacija odabrana je MVP (*Model-View-Presenter*) arhitektura koja predstavlja derivaciju *Model-View-Controler* arhitekture i upotrebljava se za izradu korisničkih sučelja [13]. MVP arhitektura se sastoji od tri dijela koji su međusobno povezani. *Model* sadrži sve modele koje aplikacija koristi, kao što su podaci o korisnicima i očitavanje podataka sa senzora za lokaciju. *View* dio arhitekture se brine za sve ono što je „vidljivo korisniku“, kao što je prikaz korisničkog sučelja, teksta... *Presenter* povezuje *Model* i *View,* koristi podatke iz modela kako bi ih prikazao na zaslonu korisnika. Dijagram za *Model-View-Presenter* je prikazan na slici 3.3.1.



**Slika 3.3.1.** *Model-View-Presenter* dijagram

Struktura koda prikazana je na slici 3.3.2., sastoji se od četiri skupine koje surađuju i nadopunjuju se kako bi pregled koda bio što lakši. Te skupine su:

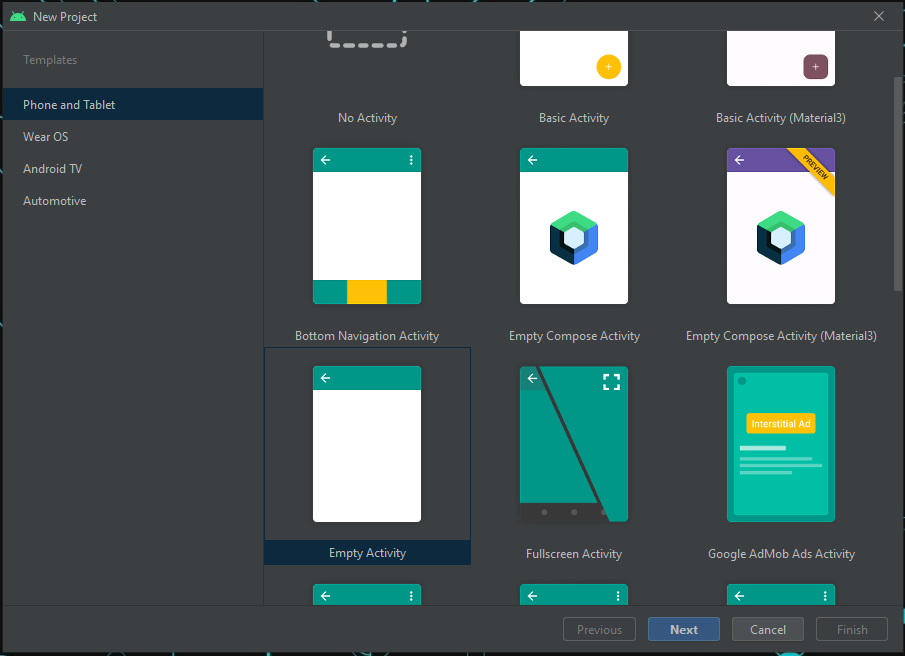
* *View* – brine se a prikaz podataka na zaslonu uređaja
* *Presenter* – dohvaća podatke i prosljeđuje ih *View* dijelu
* *Interactor* – *presenter* koristi funkcije napisane u kodu pomoću interaktora
* *Repository* – koristi se za komunikaciju s bazom podataka i vanjskim servisima



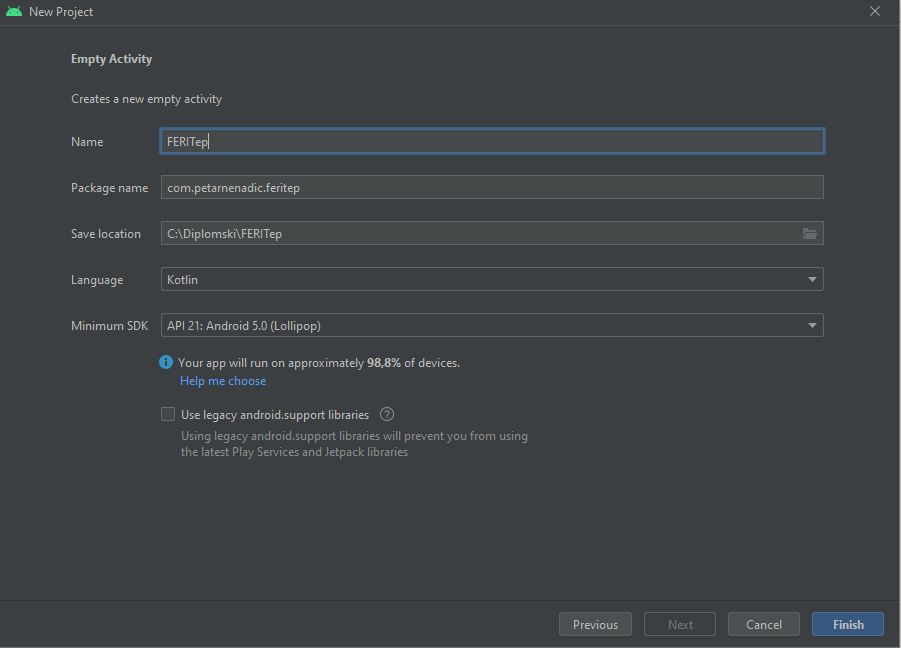
**Slika 3.3.2.** Struktura koda aplikacije

# IMPLEMENTACIJA PROGRAMSKOG RJEŠENJA

Prije početka razvoja aplikacije, potrebno je kreirati novi projekt. Zatim je potrebno odabrati vrstu početnog zaslona aplikacije (slika 3.1), nakon toga naziv aplikacije, lokaciju gdje će projekt biti spremljen te domenu pod kojom će projekt biti spremljen(slika 3.2). Nakon odabira najniže inačice operacijskog sustava na kojemu će se aplikacija moći koristiti, u ovom slučaju je to Android 5.0 (Lollipop), potrebno je kliknuti na gumb *Finish*, čime se kreira projekt.



**Slika 3.1.** Odabir izgleda početnog zaslona



**Slika 3.2.** Odabir naziva projekta, paketa, lokacije projekta, programskog jezika i najniže verzije SDK

## Dodavanje ovisnosti

Kod izrade aplikacije, prvi korak je dodavanje ovisnosti (engl. *dependencies*) koje će biti korištene. Ovisnosti se dodaju u *Build.gradle* datoteku i nužne su za uspješno *build*-anje aplikacije. Također, ovisnosti omogućuju korištenje raznih biblioteka koje nije potrebno spremati na pametni uređaj. Slika 4.1.1. prikazuje popis korištenih ovisnosti za izradu aplikacija.

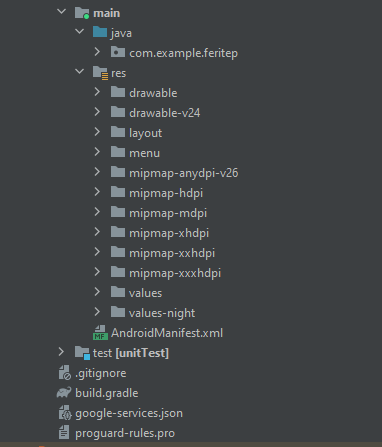


**Slika 4.1.1.** Korištene ovisnosti

## Struktura projekta

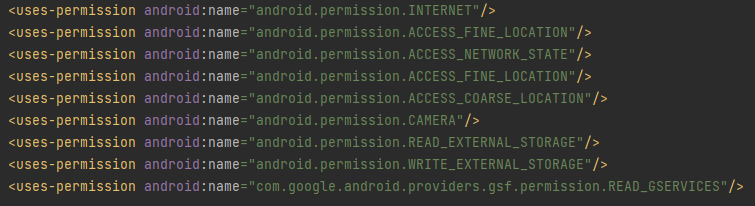
Problem ovog diplomskog rada ostvaren je objektno orijentiranim jezikom Kotlin. Uz pomoć Android Studia i njegovih alata koji su potrebni za uspješno programiranje aplikacija ostvaren je ovaj projekt. Projekt se sastoji od:

* java/ : sadrži izvorne kodove u Kotlinu
* res/ : sadrži ikone, datoteke XML, definicije grafičkih korisničkih sučelja
* test/ : sadrži datoteke za testiranje aplikacije



**Slika 3.1.1.** Sadržaj projekta

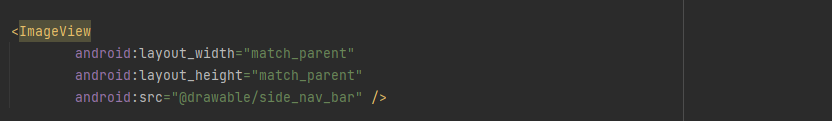
*AndroidManifest.xml* datoteka sadrži nekoliko bitnih stvari o svojstvima aplikacije koje određuju samu funkcionalnost aplikacije. Neka od tih bitnih stvari je informacija koje dozvole aplikacija koristi. Ako ove dozvole ne bi bile navedene, rješenje aplikacije se ne bi moglo uspješno implementirati. Također, u *AndroidManifest.xml* datoteci su navedene i neke osnovne informacije i elementi. Primjeri dozvola su prikazani u sljedećem kodu:



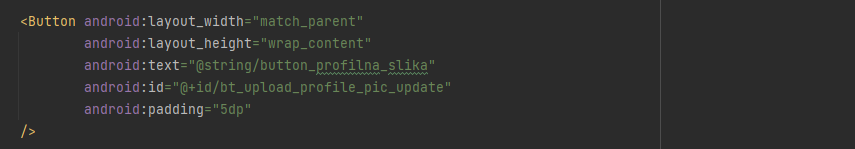
Prikaz sučelja ili *layout-*a sastoji se od nekoliko osnovnih elemenata, kao što su:

* *EditText* – okvir za prikaz i uređivanje teksta
* *ImageView* – okvir za prikazivanje slika
* *Button* – tipka koja izvršava radnju deklariranu u kodu

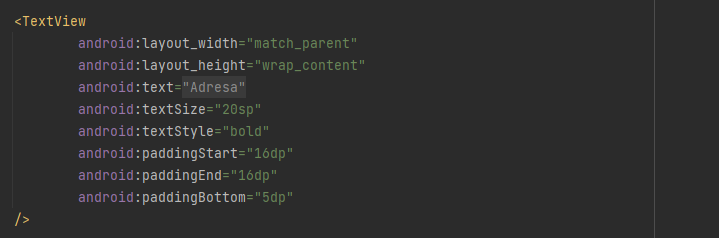
Primjer za prikaz slike je prikazan u kodu ispod:



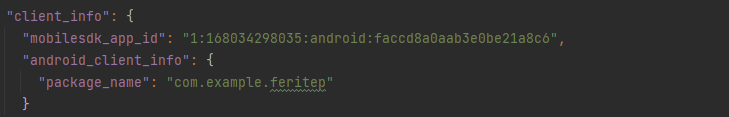
Primjer za *Button* je prikazan u kodu ispod:



Primjer za tekst je prikazan u kodu ispod:

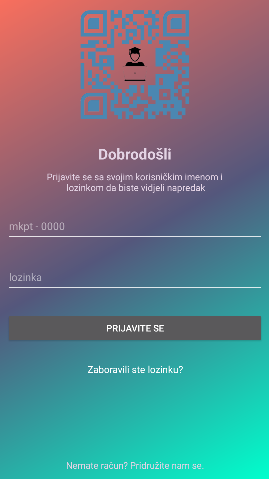


*Google-services.json* datoteka sadrži podatke koji su potrebni kako bi se aplikacija povezala na Firebase, kao što je API ključ, naziv projekta, osnovne informacije o aplikaciji. Primjer je prikazan ispod:



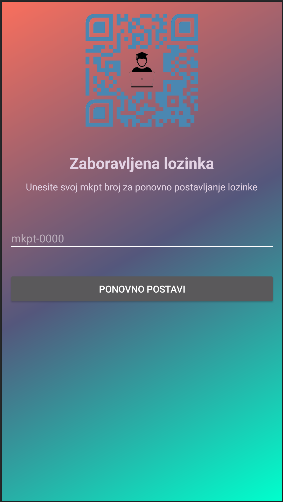
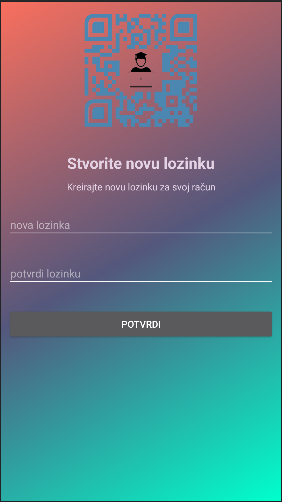
## Aplikacija za studente

Početni zaslon aplikacije za studente se sastoji od poruke dobrodošlice i gumba za prijavu korisnika. Klikom na gumb „PRIJAVITE SE“, korisnik se usmjerava na stranicu za prijavu u sustav. Dodani su zaštitni logo aplikacije i dva polja za unos korisničke oznake i lozinke, koji su definirani pri registraciji (slika 4.3.1).

**Slika 4.3.1.** Početni zaslon aplikacije (lijevo) i zaslon za prijavu u sustav (desno)

U slučaju da je korisnik zaboravio svoju lozinku, ispod gumba „PRIJAVITE SE“ nalazi se link za oporavak lozinke pod nazivom „Zaboravili ste lozinku?“. Pritiskom na taj link, korisnik se preusmjerava na stranicu za oporavak lozinke (slika 4.3.2).

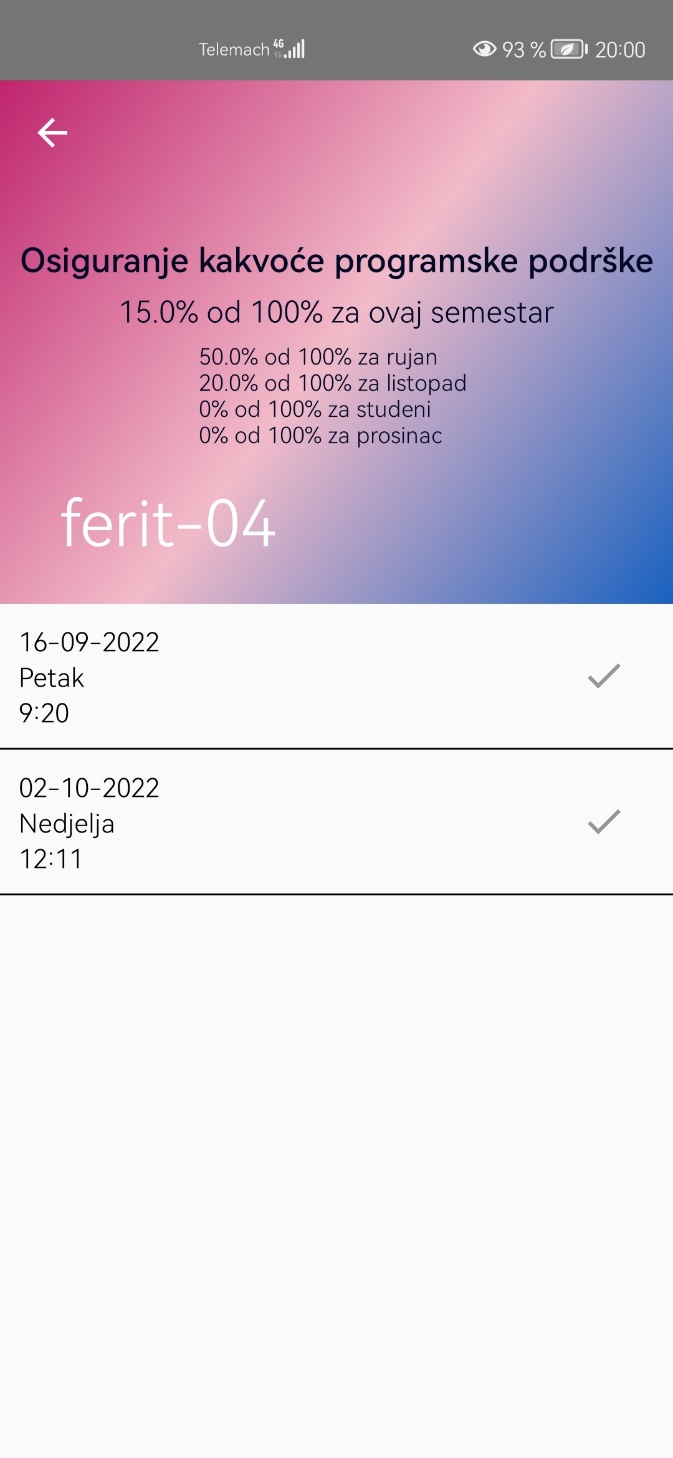
 

**Slika 4.3.2.** Zasloni za ponovno postavljanje lozinke

Nakon uspješne prijave u sustav, otvara se zaslon s popisom kolegija koje student pohađa u tekućem semestru (slika 4.3.3.). Student odabire kolegij nakon čega se otvara stranica koja sadrži detalje o prisutnosti, kao što su datum i vrijeme kada je student skenirao QR kod s nastavnikova uređaja (slika 4.3.4.).

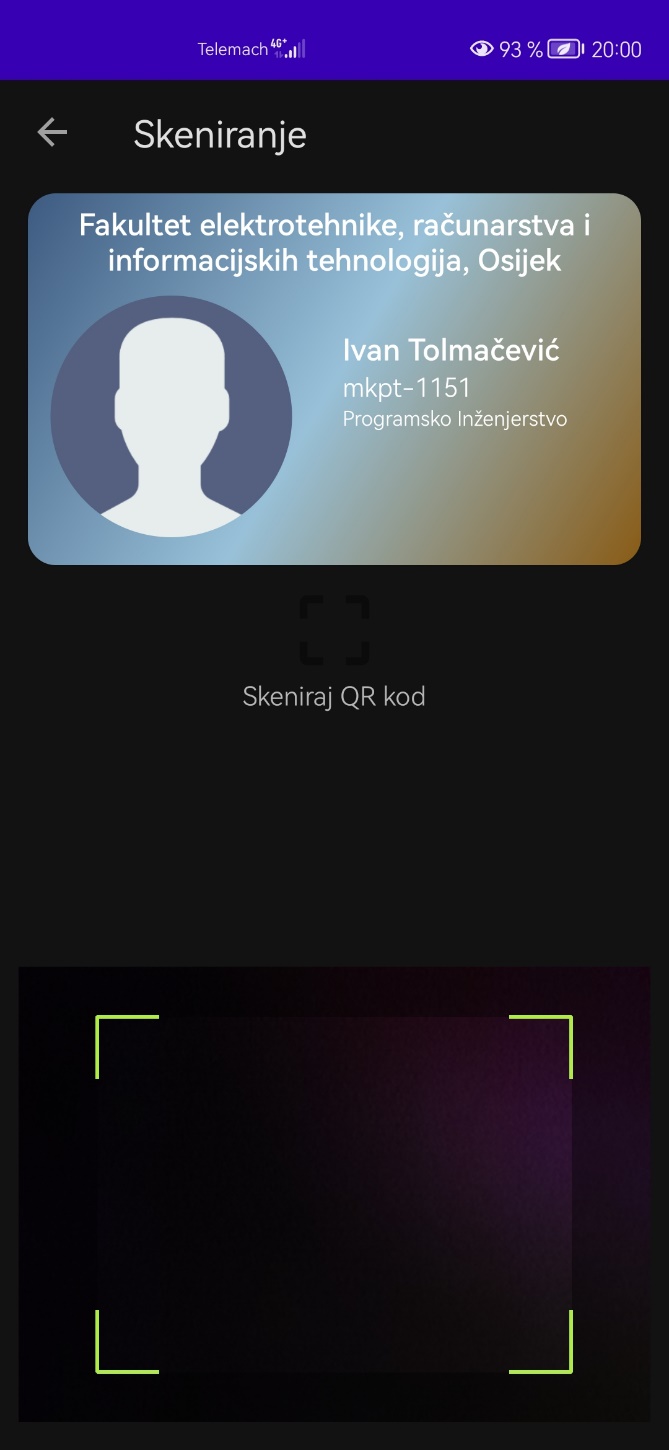


**Slika 4.3.3.** Prikaz kolegija koje student pohađa

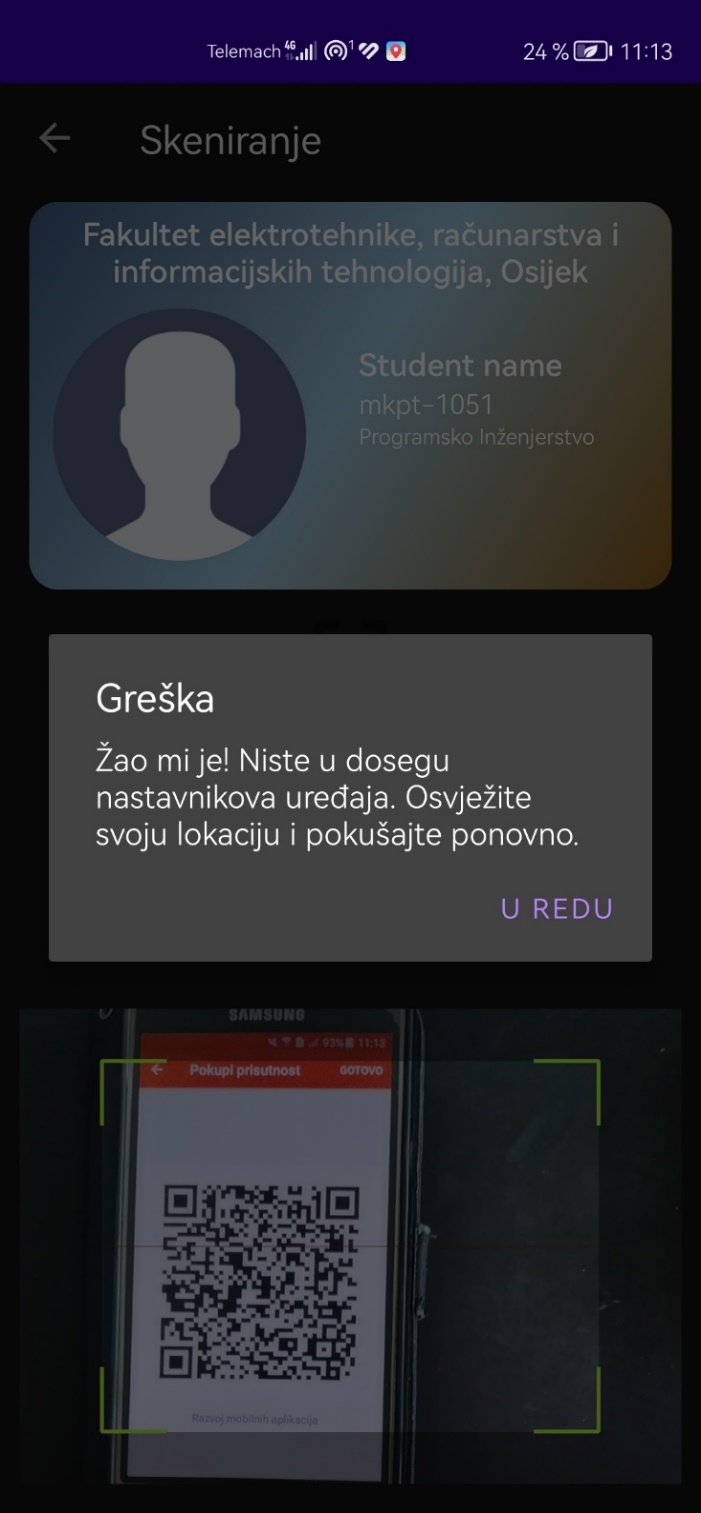


**Slika 4.3.4.** Prikaz detalja o kolegiju

Na početnom zaslonu se nalazi i gumb koji vodi na zaslon za skeniranje QR koda (slika 4.3.5.). Aplikacija koristi kameru uređaja kako bi skenirala QR kod koji je automatski generiran putem aplikacije za nastavnike. Aplikacija koristi lokacijske usluge kako bi se osiguralo da studenti ne mogu skenirati QR kod ako su udaljeni više od 10 metara od nastavnikova uređaja. U slučaju da je uređaj koji student skenira QR kod udaljen više od 10 metara od nastavnikova uređaja, pojavljuje se skočni prozor koji obavještava studenta da je previše udaljen (slika 4.3.6.).

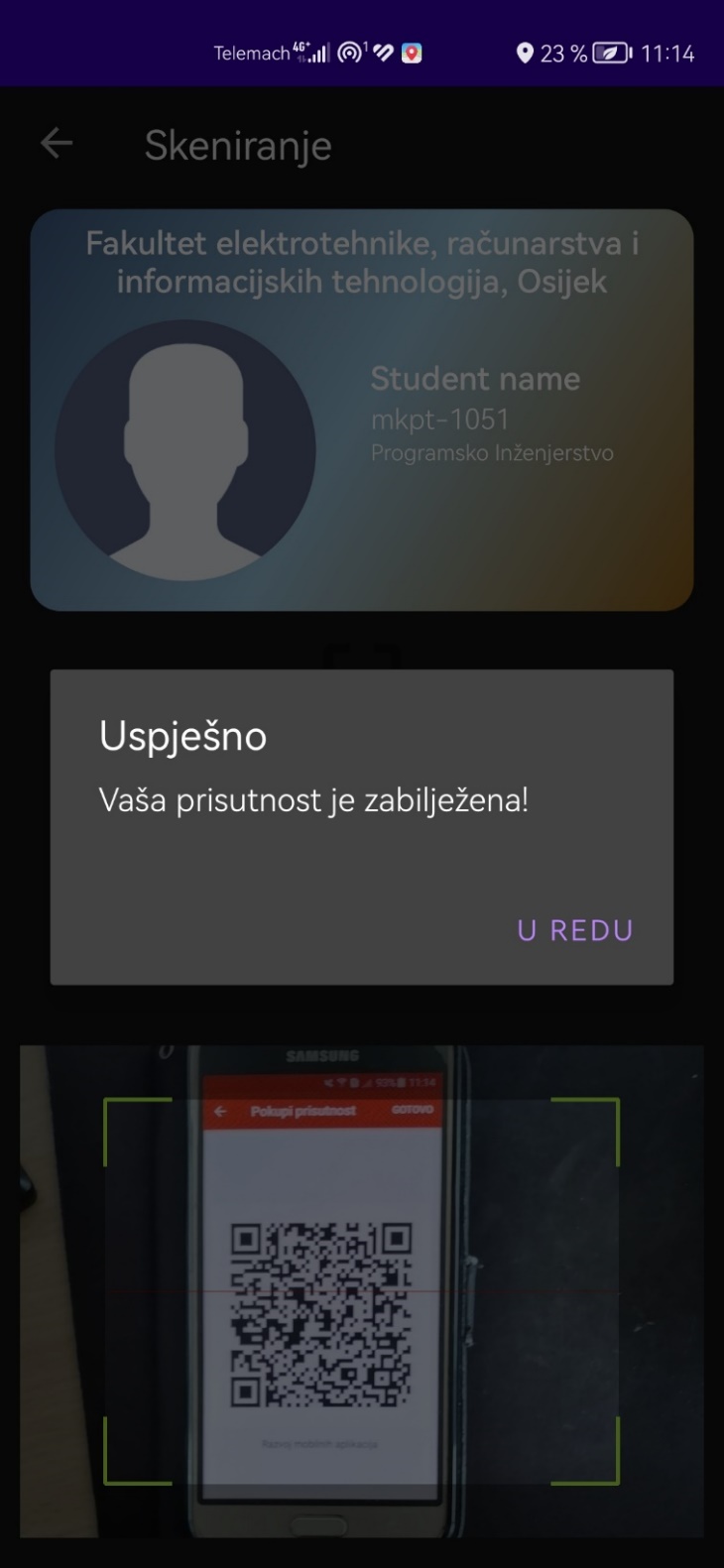


**Slika 4.3.5.** Zaslon za skeniranje QR koda



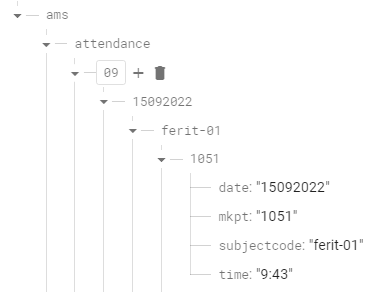
**Slika 4.3.6.** Obavijest za udaljenost

Ako je student uspješno skenirao QR kod za prijavu na nastavu na zaslonu uređaja se pojavljuje obavijest (slika 4.3.7.)



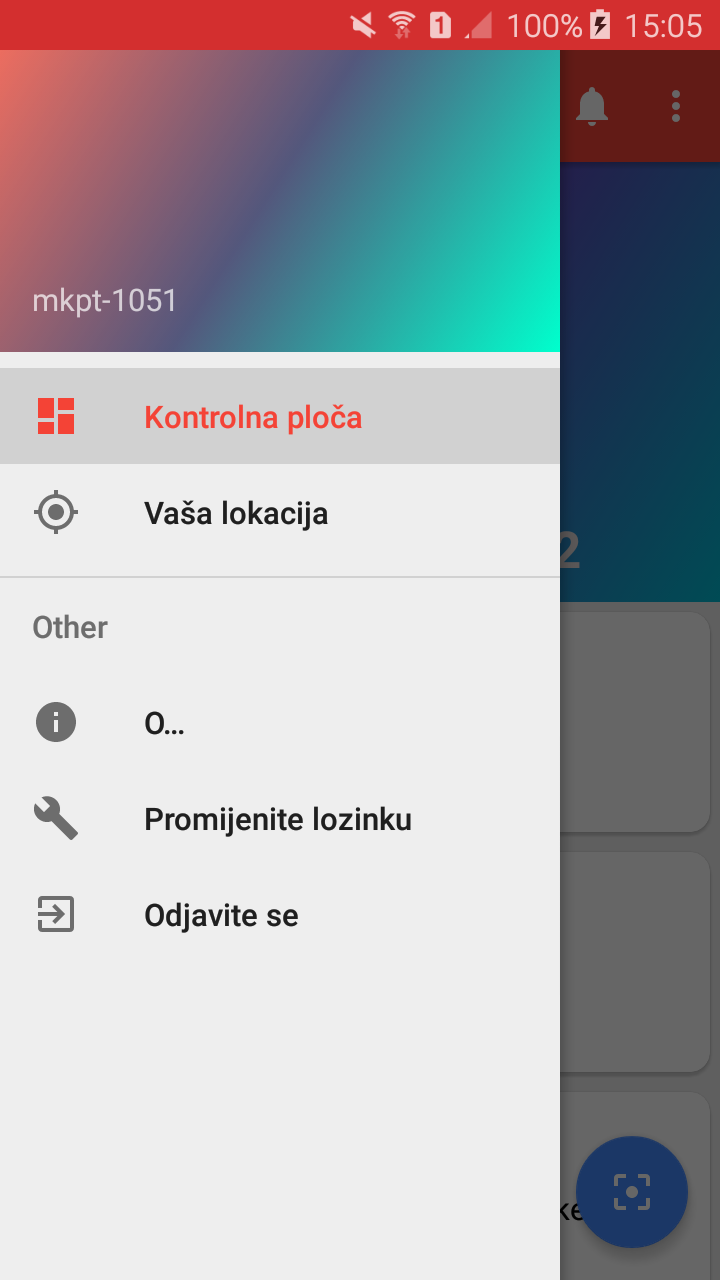
**Slika 4.3.7.** Obavijest o uspješnom prijavljivanju na kolegij

Nakon što student skenira QR kod s profesorova pametnog uređaja, u Firebase se upisuju podaci o prijavi (slika 4.3.8.). Ti podaci su kasnije vidljivi i studentu i profesoru.



**Slika 4.3.8.** Izgled tablice u Firebase-u nakon prijave studenta na predavanje

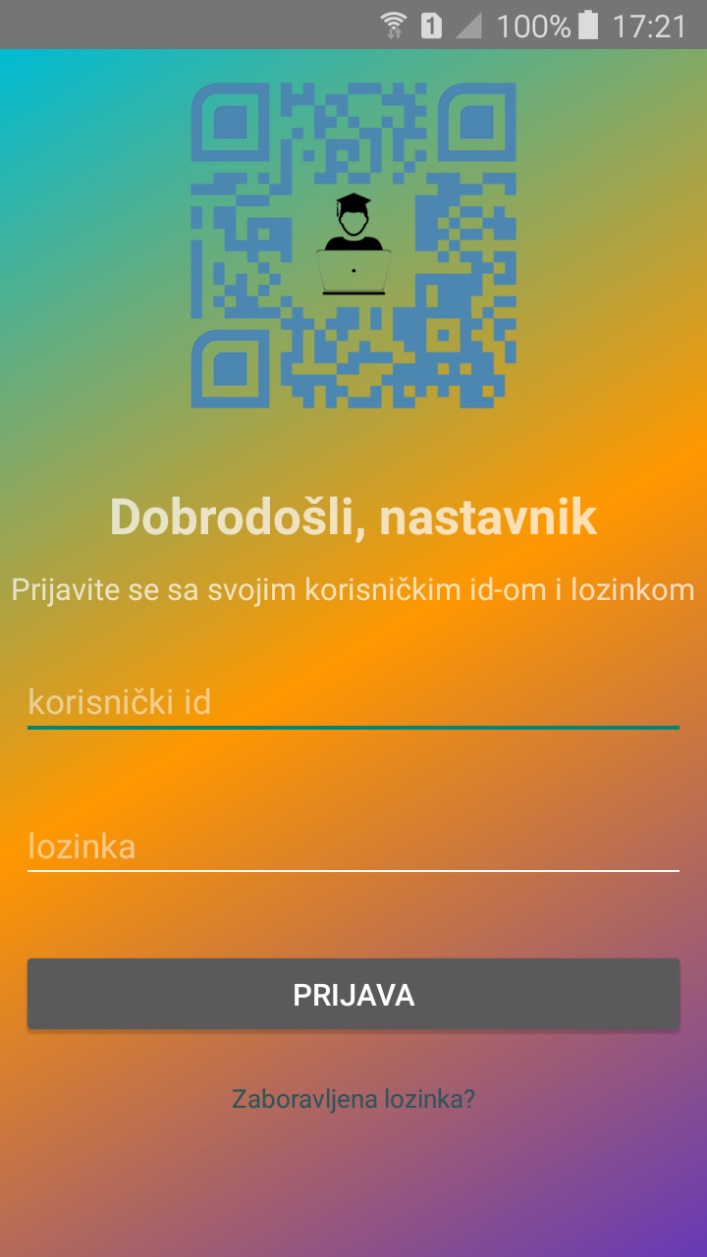
Na početnoj stranici, u gornjem lijevom kutu, nalazi se ikona kojom se otvara izbornik s opcijom za pregled kolegija koje student pohađa, promjenu lozinke i odjavu iz aplikacije (slika 4.3.9.).

****

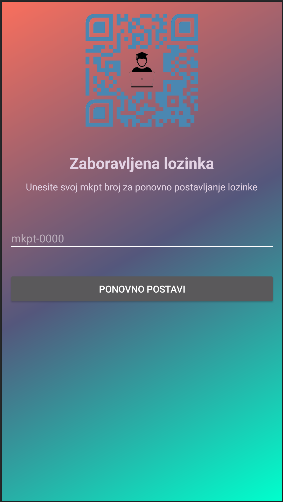
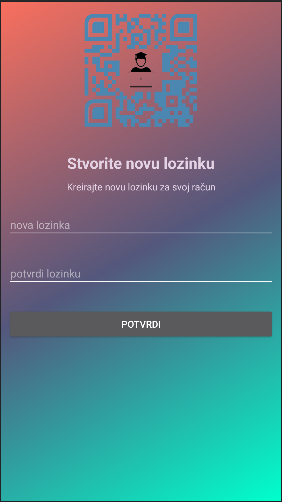
**Slika 4.3.9.** Izbornik s opcijama

## Aplikacija za profesore

Početni zaslon aplikacije (slika 4.4.1) za profesore sastoji se od poruke dobrodošlice, polja za unos korisničkog ID-a i lozinke te gumba za prijavu korisnika. Ispod gumba „Prijava“ nalazi se link pod nazivom „Zaboravljena lozinka?“ koji preusmjerava korisnika na stranicu za ponovno postavljanje lozinke (slika 4.4.2).



**Slika 4.4.1.** Početni zaslon aplikacije za profesore

**Slike 4.4.2.** Zasloni za ponovno postavljanje lozinke

Nakon što se profesor uspješno prijavi, preusmjerava se na glavni zaslon s popisom studenata koji pohađaju njegove kolegije (slika 4.4.3).



**Slika 4.4.3.** Popis studenata

Klikom na pojedinog studenta, otvara se zaslon s popisom kolegija koje student sluša kod profesora koji je trenutno prijavljen u aplikaciju (slika 4.4.4).



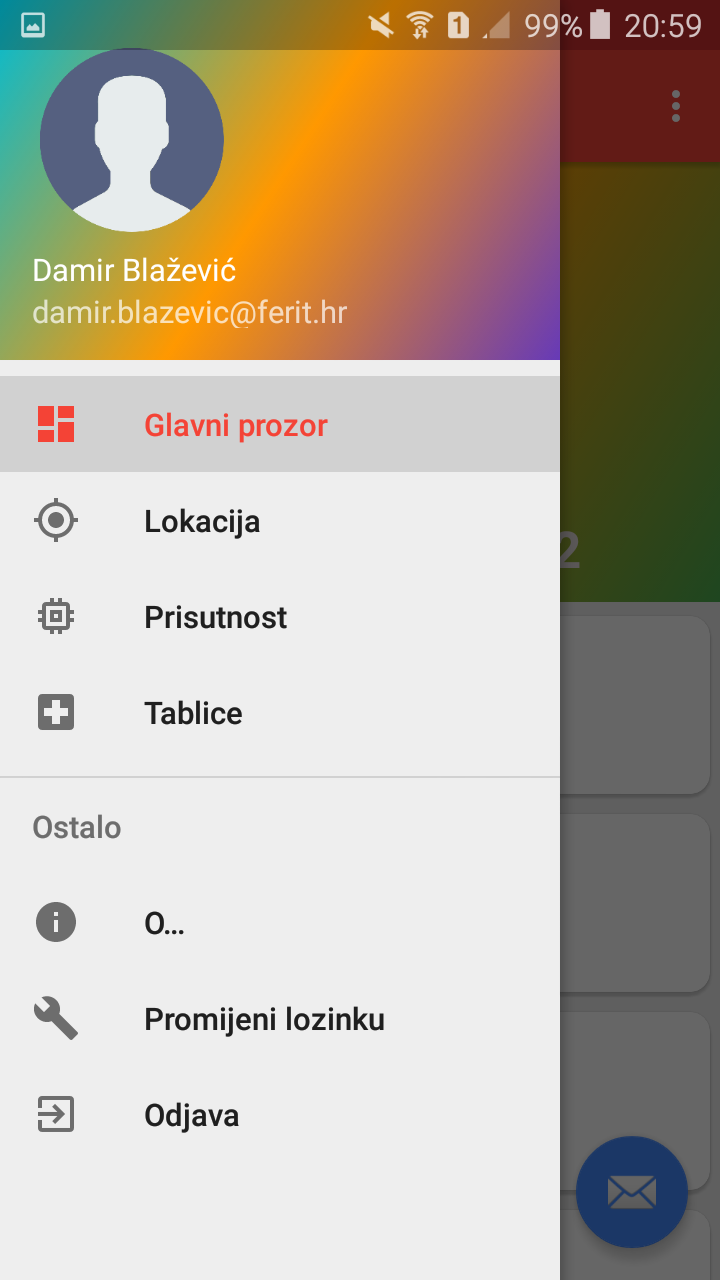
**Slika 4.4.4.** Popis kolegija koje student sluša kod profesora koji je trenutno prijavljen u aplikaciju

Odabirom kolegija otvara se prozor s detaljima o kolegiju, kao što su: ID studenta, kod kolegija, ukupni postotak prisutnosti na kolegiju, postotak prisutnosti za svaki mjesec u kojem se kolegij izvodi te datum i vrijeme za svaku prijavu studenta (slika 4.4.5).



**Slika 4.4.5.** Prozor s detaljima o kolegiju

Na glavnom zaslonu aplikacije, u gornjem lijevom kutu, nalazi se gumb za otvaranje izbornika s opcijama za povrat na glavni zaslon, opcija za prikupljanje prisutnosti te opcija za kreiranje tablica s detaljima o prisutnosti studenata (slika 4.4.6).



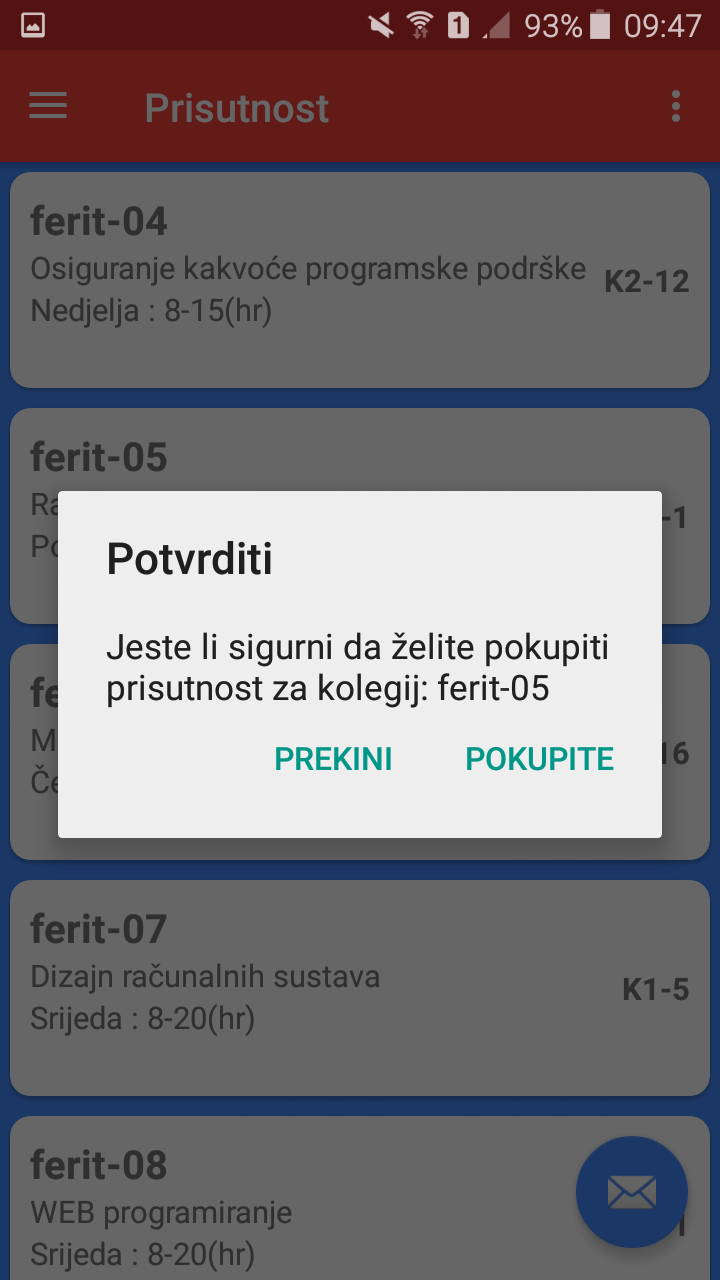
**Slika 4.4.6.** Izbornik s opcijama

Pritiskom na gumb „Prisutnost“ otvara se prozor s popisom kolegija koje predaje profesor koji je trenutno prijavljen u aplikaciju (slika 4.4.7).



**Slika 4.4.7.** Popis kolegija koje profesor predaje

Pritiskom na pojedini kolegij, pojavljuje se obavijest koja traži potvrdu za prikupljanje prisutnosti za odabrani kolegij. Nakon što profesor pritisne gumb „Pokupite“ (slika 4.4.8.), aplikacija provjerava vrijeme i datum na pametnom uređaju i uspoređuje ih s vremenom izvođenja kolegija koji su zapisani u Firebase-u. Ako se datum i vrijeme na Firebase*-*u i pametnom uređaju podudaraju generira se QR kod koji služi za prijavljivanje studenata na nastavu (slika 4.4.9.).



**Slika 4.4.8.** Zaslon s obavijesti

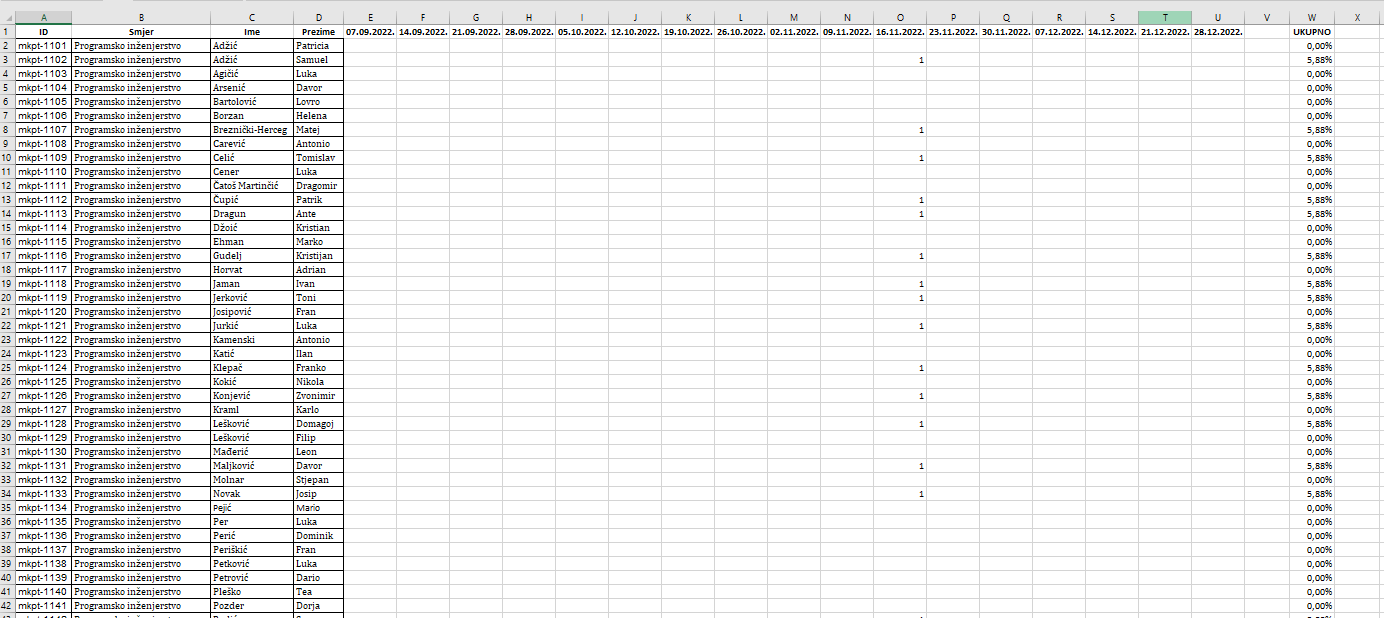


**Slika 4.4.9.** QR kod za prijavljivanje prisutnosti

Kada profesor, u izborniku s opcijama odabere „Tablice“, otvara se prozor s popisom kolegija (slika 4.4.10.) te se klikom na pojedini kolegij generira tablica s popisom svih studenta koji slušaju odabrani kolegij. U toj tablici se nalaze detalji o prisutnosti za pojedinog studenta, kao što su datum kada je student skenirao QR kod kako bi potvrdio svoju prisutnost na kolegiju te ukupna prisutnost (slika 4.4.11.).



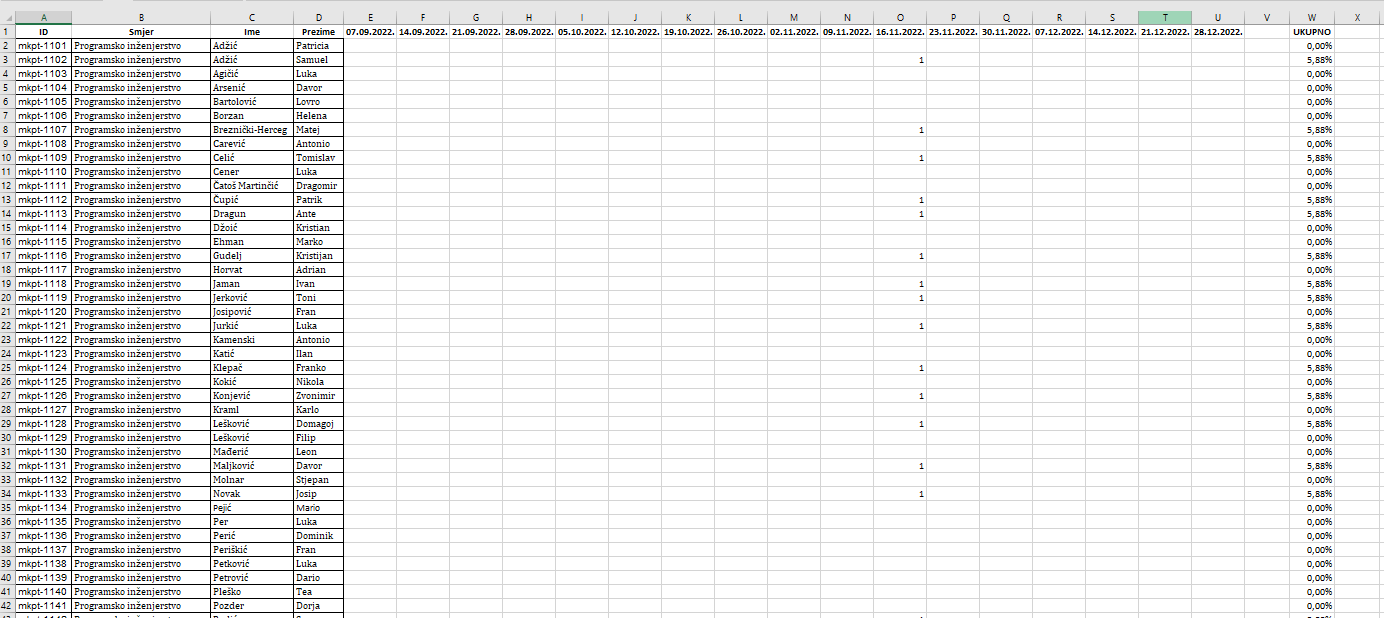
**Slika 4.4.10.** Popis kolegija



**Slika 4.4.11.** Prikaz tablice sa sveukupnom prisutnošću

## Testiranje aplikacija

Testiranje aplikacija provedeno je na studentima 2. godine preddiplomskog studija Elektrotehnike i informacijske tehnologije, smjer Komunikacije i informatika, te na 2. godini preddiplomskog studija Računarstvo, smjerovi Programsko inženjerstvo i Računalno inženjerstvo. U testiranju je sudjelovalo 97 studenata. Slika 4.5.1. pokazuje tablicu s popisom studenata nakon evidentiranja prisutnosti putem aplikacije. Nažalost, neki studenti nisu mogli biti prijavljeni zbog toga što su koristili pametne uređaje s iOS operacijskim sustavom, pa je njihova prisutnost morala biti evidentirana potpisima na papiru.



**Slika 4.5.1.** Tablica s popisom studenata

# Zaključak

U ovom radu opisan je način dizajniranja i korištenje mobilne aplikacije za vođenje evidencije prisutnosti. Cilj je bio napraviti najjednostavniju aplikaciju za vođenje evidencije prisutnosti i demonstrirati njeno korištenje. Tijekom razvoja aplikacije, došlo se do zaključka da je korišteno puno funkcionalnosti programskog jezika Kotlin bez kojih aplikacija ne bi mogla funkcionirati. Pošto je aplikacija rađena za pametne uređaje s operacijskim sustavom Android nije ih moguće koristiti na uređajima koji rade na iOS operacijskom sustavu.

Za izradu ovakvog sustava preporučeno je da više ljudi radi na tome, od osobe koja će osmisliti cijeli koncept, preko dizajnera, programera za Android i iOS operacijske sustave, pa sve do marketinških stručnjaka koji će adekvatno predstaviti proizvod na tržištu.

# LITERATURA

[1] B.Phillips, C. Stewart, K. Marsciano, Android Programming: The Big Nerd Ranch Guide (3rd Edition), USA, 2017.

[2] R. Meier, Professional Android 4 Application Development, John Wiley and Sons, Indianapolis, USA, 2012.

[3] M. Gargenta, Learning Android, O'Reilly Media, Sebastopol, 2011.

[4] Official Android Developer page: <https://developer.android.com/>

[5] Official Firebase page: <https://firebase.google.com/>

[6] Official Kotlin page: <https://kotlinlang.org/>

[7] FreeCodeCamp YouTube page: <https://www.youtube.com/c/Freecodecamp/featured>

[8] Philipp Lackner YouTube page: <https://www.youtube.com/c/PhilippLackner>

[9] Traversy Media YouTube page: <https://www.youtube.com/c/TraversyMedia>

[10] F. Babić, L. Kordić, N. Srivastava, Kotlin Coroutines by Tutorials, 3rd Edition, Ray Wenderlich, USA, 2022.

[11] Official StackOverflow page: <https://stackoverflow.com/>

[12] Razvoj mobilnih aplikacija, priručnik za laboratorijske vježbe, Fakultet elektrotehnike, računarstva i informacijskih tehnologija, Osijek, 2019.

[13] Model-view-presenter, Wikipedia:

<https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter> [studeni 2021.]

[14] Advanced Encryption Standard, Wikipedia:

<https://en.wikipedia.org/wiki/Advanced_Encryption_Standard> [listopad 2022.]

[15] Attendy – Attendance register:

<https://play.google.com/store/apps/details?id=fr.gym.mattnissartdevelopment.attendanceApp>

[16] Presli – Attendance Manager:

<https://play.google.com/store/apps/details?id=com.couchbits.presence>

[17] Alora – Attendance Tracker App:

<https://play.google.com/store/apps/details?id=com.smartlogicinc.attendancemanager>

[18] Self Attendance:

<https://play.google.com/store/apps/details?id=saurabhrao.selfattendance>

[19] QR Attendance:

<https://play.google.com/store/apps/details?id=com.mdisattendanceapp>

[20] QR Attendance Control:

<https://play.google.com/store/apps/details?id=com.qrattendancecontrol>

# SAŽETAK

Zadatak ovog diplomskog rada je bio izraditi mobilnu aplikaciju za evidenciju prisutnosti na nastavi. Bilo je potrebno izraditi plan izrade, bazu podataka i arhitekturu datoteka. Nadalje, trebalo je unaprijed zadati svrhe i vrste objekata, dodijeliti prava različitim korisnicima te detaljno voditi evidenciju prisutnosti za svaki pojedini kolegij kojeg student sluša.

Nakon postavljanja logičkog dijela, sva pažnja je posvećena korisničkom iskustvu te je dizajnirano sučelje koje može manipulirati bazom podataka i pristupiti svim svojstvima aplikacije.

**Ključne riječi:** Android, mobilna, aplikacija, evidencija, prisutnost, nastava, student, profesor

# ABSTRACT

The premise of this thesis was to create a mobile application for attendance records. It was necessary to create plan for whole process, database and file architecture. Furthermore, it was necessary to predetermine the purposes and types of facilities, to distribute rights to different users in order to keep detailed records of attendance for each individual course that student listens to.

After setting up the logical part, all attention is paid to the user experience and an interface has been designed that can manipulate databases and access all application properties.

**Keywords:** Android, mobile, application, records, attendance, teaching, student, professor

# ŽIVOTOPIS

Petar Nenadić rođen je 12. veljače 1992. godine u Splitu, Republika Hrvatska. 2006. godine je završio osnovnu školu Stjepan Radić u Imotskom nakon čega iste godine upisuje Opću Gimnaziju dr. Mate Ujevića u Imotskom. Poslije završetka srednje škole 2010. godine upisuje Fakultet Elektrotehnike, Strojarstva i Brodogradnje (FESB) u Splitu. U akademskoj godini 2014./2015. prebacuje se na Fakultet Elektrotehnike, Računarstva i Informacijskih Tehnologija (FERIT) u Osijeku gdje 2018. godine završava stručni studij Elektrotehnike, smjer Informatika. Iste godine upisuje studij Razlikovnih obveza na istom fakultetu. 2020. godine upisuje diplomski studij Računarstva, smjer Programsko inženjerstvo.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Potpis autora

# PRILOZI

**Prilog 1.** Link za aplikaciju: https://github.com/nenadicpetar/Aplikacija-za-vo-enje-evidencije-prisutnosti-na-nastavi