Šolski center Novo mesto  
Srednja elektro šola in tehniška gimnazija  
Šegova ulica 112  
8000 Novo mesto

**Mali pomočnik – mobilna aplikacija**(Maturitetna seminarska naloga)

Predmet: računalništvo  
Avtor: Živa Lavrič, T4B  
Mentor: Gregor Mede, univ. dipl. inž. rač. in inf.

Novo mesto, marec 2022

**POVZETEK IN KLJUČNE BESEDE**

Seminarska naloga vsebuje kratek opis sodobnega pametnega telefona in njegovo krajšo zgodovino. Ker je namen seminarske naloge izdelava mobilne aplikacije, v njej najdemo opise nekaterih vrst aplikacij, njihove prednosti in slabosti. Opisani so tudi programski jeziki v katerih aplikacijski razvijalci programirajo. Seminarska naloga vsebuje postopek izdelave aplikacije po korakih, katerim sem ohlapno sledila pri izdelavi svoje aplikacije. Izdelavo sem začela s preprostim načrtom, h kateremu sem se vračala v poteku izdelave. Aplikacija Mali Pomočnik vsebuje forum za registracijo in prijavo, funkcije kot so predvajalnik glasbe, časovnik in prikaz uporabnikovih podatkov. Postopki izdelave in rezultati so podrobneje opisani s slikovnimi prikazi za lažjo predstavo. V zadnjem delu naloge je prav tako slikovno prikazan končni produkt z opisanim delovanjem vseh funkcij.

KLJUČNE BESEDE: Mobilna aplikacija, XML, programski jezik, programsko orodje, podatkovna baza, funkcija

**Kazalo vsebine**

[1 UVOD 1](#_Toc100791792)

[2 PAMETNI MOBILNI TELEFONI 2](#_Toc100791793)

[2.1 Kratka zgodovina pametnih mobilnih telefonov 3](#_Toc100791794)

[3 MOBILNE APLIKACIJE 3](#_Toc100791795)

[3.1 Najbolj priljubljene vrste mobilnih aplikacij 4](#_Toc100791796)

[3.1.1 »Native« aplikacije 4](#_Toc100791797)

[3.1.2 Mobilne spletne aplikacije (Web applications) 4](#_Toc100791798)

[3.1.3 Hibridne mobilne aplikacije 5](#_Toc100791799)

[3.1.4 »Cross-platform« aplikacije 5](#_Toc100791800)

[3.2 Najboljši programski jeziki za razvoj aplikacij za Android 5](#_Toc100791801)

[3.2.1 Java 5](#_Toc100791802)

[3.2.2 Kotlin 5](#_Toc100791803)

[3.2.3 C++ 6](#_Toc100791804)

[3.2.4 C# 6](#_Toc100791805)

[3.2.5 Python 6](#_Toc100791806)

[3.2.6 HTML, CSS, JavaScript 6](#_Toc100791807)

[3.3 Proces načrtovanja mobilne aplikacije 7](#_Toc100791808)

[4 NAČRTOVANJE IN REALIZACIJA APLIKACIJE MALI POMOČNIK 9](#_Toc100791809)

[4.1 Načrt aplikacije 9](#_Toc100791810)

[4.2 Načrt izgleda aplikacije 10](#_Toc100791811)

[4.3 Izdelava XML datotek 10](#_Toc100791812)

[4.4 Ustvarjanje podatkovne baze 12](#_Toc100791813)

[4.4.1 Registracija 13](#_Toc100791814)

[4.4.2 Prijava 14](#_Toc100791815)

[4.5 Domače okno aplikacije 14](#_Toc100791816)

[4.5.1 Predvajalnik glasbe 15](#_Toc100791817)

[4.5.2 Časovnik 16](#_Toc100791818)

[4.5.3 Prikaz uporabnikovih podatkov 17](#_Toc100791819)

[4.6 Testiranje aplikacije 18](#_Toc100791820)

[5 ZAKLJUČEK 21](#_Toc100791821)

[6 ZAHVALA 22](#_Toc100791822)

[7 VIRI IN LITERATURA 23](#_Toc100791823)

[8 STVARNO KAZALO 24](#_Toc100791824)

[9 PRILOGE 25](#_Toc100791825)

**Kazalo slik**

[Slika 1 Načrt aplikacije - Mali pomočnik 9](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791826)

[Slika 2 Datoteke XML za prikaze na zaslonu 10](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791827)

[Slika 3 Urejanje gumbov v XML 10](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791828)

[Slika 4 Urejanje besedilnih polj v XML 11](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791829)

[Slika 5 Prikaz povezave dveh spremenljivk v XML 11](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791830)

[Slika 6 Prikaz Firebase v zavihku "Tools" 12](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791831)

[Slika 7 Koda razreda "User" 12](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791832)

[Slika 8 Prikaz implementacije View.OnClickListener v razred 13](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791833)

[Slika 9 Prikaz postavitev vnosnih oken in gumbov, v oknu za registracijo 13](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791834)

[Slika 10 Prikaz kode, za vpisovanje uporabnika v bazo 13](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791835)

[Slika 11 Prikaz dela metode userLogin 14](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791836)

[Slika 12 Prikaz načrta domačega okna 14](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791837)

[Slika 13 Prikaz kode ob pritisku na gumb logout 14](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791838)

[Slika 14 Primer gumba za začetek predvajanja v XML datoteki 15](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791839)

[Slika 15 Prikaz metode play 15](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791840)

[Slika 16 Potrebne spremenljivke za časovnik 16](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791841)

[Slika 17 Prikaz metode StartTimer 16](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791842)

[Slika 18 Prikaz metode pauseTimer 16](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791843)

[Slika 19 Prikaz metode resetTimer 17](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791844)

[Slika 20 Prikaz metode updateCountDownText 17](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791845)

[Slika 21 Metode orodja Firebase 17](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791846)

[Slika 22 Načrt okna ProfileAcc 17](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791847)

[Slika 23 Povezovanje izpisa in podatkovne baze 18](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791848)

[Slika 24 Dokončan form za registracijo 18](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791849)

[Slika 25 Uspešna registracija novega uporabnika 18](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791850)

[Slika 26 Izpisani podatki uporabnika v bazi podatkov 19](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791851)

[Slika 27 Prijavno okno aplikacije 19](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791852)

[Slika 28 Domača stran aplikacije 19](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791853)

[Slika 29 Izpisani podatki trenutno prijavljenega uporabnika 20](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791854)

[Slika 30 Okno predvajalnika glasbe 20](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791855)

[Slika 31 Okno časovnika 20](file:///C:\Users\Živa\Desktop\Seminarska%20naloga%20-%20Mali%20pomočnik.docx#_Toc100791856)

# UVOD

Moderna tehnologija je v zadnjih nekaj letih doživela razcvet, še posebej na področju mobilne tehnologije. Pametni telefoni so nam s svojimi storitvami olajšali tudi najpreprostejše življenjske opravke.

Razvoj mobilnih aplikacij je proces ustvarjanja programskih aplikacij, ki se izvajajo na mobilni napravi. Zato proces mobilnega razvoja vključuje ustvarjanje programskih svežnjev, ki jih je mogoče namestiti (koda, binarne datoteke, sredstva itd.), izvajanje zalednih storitev, kot je dostop do podatkov z API-jem, in testiranje aplikacije na ciljnih napravah. [1]

Namen aplikacije Mali Pomočnik je pomagati uporabniku pri lajšanju stresa in pomoč pri učenju. Seminarska naloga popisuje nekatere vrste aplikacij, ki jih danes poznamo. Izpostavljene so prednosti, slabosti ter razlike med njimi. Predstavljeni so tudi določeni programskimi jeziki, v katerih je možno programirati aplikacije. Ker mi je najbolj poznan programski jezik Java, bo aplikacija napisana s pomočjo le-tega. Orodje, katerim se bom ukvarjala je Android Studio saj ta omogoča najlažjo povezavo s podatkovno bazo, Firebase.

# PAMETNI MOBILNI TELEFONI

Pametni telefon je mobilni telefon, ki nam omogoča več kot le telefoniranje in pošiljanje sporočil. S pametnimi telefoni lahko brskamo po spletu, izvajamo programe, kot jih lahko s pomočjo računalnika. Uporabljajo zaslon na dotik, ki uporabnikom omogoča interakcijo z njimi. Obstaja ogromno aplikacij za pametne telefone, ti vključujejo igre, programe za osebno in poslovno uporabo, itd. [2] Spodaj je navedenih nekaj drugih lastnosti pametnega telefona:

* Povezava z internetom,
* Brskalnik,
* Vgrajen pomnilnik,
* Tipkovnica QWERTY, ki temelji na strojni ali programski opremi,
* Brezžična sinhronizacija z drugimi napravami, kot so prenosni ali namizni računalniki,
* Možnost prenosa aplikacij in samostojnega zagona,
* Možnost zagona več aplikacij hkrati,
* Zaslon na dotik,
* Wifi,
* Digitalni fotoaparat,
* Igranje iger,
* Možnost pošiljanja SMS,
* GPS [3]

Telefoni »BlackBerry« so bil eni izmed prvih priljubljenih pametnih telefonov. Ko so jim začeli dodali naprednejše funkcije, varnostne tehnike in integracije z orodji za upravljanje IT[[1]](#footnote-1), so začeli pridobivati ​​popularnost tudi v podjetjih.

Apple in Google sta si prizadevala izboljšati poslovne zmogljivosti svojih mobilnih operacijskih sistemov (OS), ki IT-ju omogočijo boljšo podporo naprav »iPhone« in »Android«.

Ker je oblika pametnega telefona manjša od namiznega računalnika, ga poslovni uporabniki običajno uporabljajo za hitra opravila, kot je pošiljanje e-pošte. Tablični računalniki in naprave »dva v enem« so se pojavili na trgu mobilnih naprav kot alternativa pametnim telefonom in osebnim računalnikom. [3]

## Kratka zgodovina pametnih mobilnih telefonov

Prvo različico tega, kar definiramo kot pametni telefon, je leta 1992 izumil IBM. Imenuje se Simon Personal Communicator (ali samo IBM Simon), na voljo je bil za nakup leta 1994. Prodanih je bilo več kot deset tisoč.

IBM Simon se je od prejšnjih generacij mobilnih telefonov razlikoval s funkcijami, kot so možnost pošiljanja in prejemanja e-pošte ter zaslon na dotik. Prišel je tudi z vgrajenimi programi:

* Koledar
* Imenik
* Koledar terminov
* Standardne in predvidene tipkovnice za vnos s pisalom
* Elektronska beležnica
* Ročno napisanimi pripombami

Zasnova, orodja in zmogljivost pametnih telefonov se je z leti močno razvila, vendar so številne funkcije enake kot tiste, ki jih najdemo v IBM Simonu in drugih zgodnjih modelih, so zdaj standard v današnjih pametnih telefonih. [4]

Izraz "pametni telefon" je postal vsakdanji, saj so pametni telefoni postali bolj razširjeni in cenovno dostopnejši vse skozi 2000-a leta.

# MOBILNE APLIKACIJE

Eden najpomembnejših elementov pametnega telefona je njegova možnost povezave s trgovino z aplikacijami. Trgovina z aplikacijami je centraliziran portal, kjer lahko uporabniki iščejo in prenašajo aplikacije za izvajanje na svojih telefonih. Tipična trgovina z aplikacijami ponuja na tisoče mobilnih aplikacij za produktivnost, igranje iger, obdelavo besedil, beleženje, organizacijo, socialne medije in drugo. [3]

Mobilna aplikacija je vrsta aplikacije, zasnovana za delovanje na mobilni napravi, ki je lahko pametni telefon ali tablični računalnik. Tudi če so aplikacije običajno majhne programske enote z omejeno funkcijo, jim vseeno uspe uporabnikom zagotoviti kakovostne storitve in izkušnje. V nasprotju z aplikacijami, zasnovanimi za namizne računalnike, se mobilne aplikacije odmikajo od integriranih sistemov programske opreme. Namesto tega vsaka mobilna aplikacija ponuja izolirano, omejeno funkcionalnost (npr. igra, kalkulator, internetni iskalnik).

Zaradi omejenih virov strojne opreme zgodnjih mobilnih naprav so se mobilne aplikacije izogibale več funkcionalnosti. Kljub temu, da so naprave, ki se uporabljajo danes, veliko bolj izpopolnjene, mobilne aplikacije še vedno ostajajo ozko funkcionalne. Tako lastniki mobilnih aplikacij potrošnikom omogočajo, da ročno izberejo funkcije, ki bi jih morale imeti njihove naprave.

## Najbolj priljubljene vrste mobilnih aplikacij

Veliko vprašanje, ki se pojavi, ko morajo razvijalci mobilnih aplikacij razviti aplikacije, je vrsta aplikacije, ki jo je treba razviti. Ta zahteva se ne razlikuje samo glede na zahteve naročnika, temveč tudi glede na zasičenost trga in na raven udobja razvijalcev s tehnologijo. [5]

### »Native« aplikacije

Te aplikacije so kodirane v določenem programskem jeziku, kot je Objective C za iOS ali Java za Android (Preprosto povedano, ne morete namestiti in uporabljati aplikacije za Android na iPhoneu in obratno). Te aplikacije so tudi drage za razvoj, saj so zasnovane samo za eno vrsto operacijskega sistema in jih je treba podvojiti za druge operacijske sisteme, zato se imenujejo native, izvirajo iz določene platforme ali naprave. [5]

Glavna prednost domačih aplikacij je njihova visoka zmogljivost in odlična uporabniška izkušnja. Dostop do širokega nabora API[[2]](#footnote-2)-jev prav tako pomaga pospešiti razvojno delo in razširiti meje uporabe aplikacij. Native aplikacije je mogoče prenesti samo iz trgovin z aplikacijami in jih namestiti neposredno v naprave. Zato morajo najprej prestati strog postopek založništva. Najpomembnejša pomanjkljivost domačih aplikacij je njihova cena. Za izdelavo, podporo in vzdrževanje aplikacije za Android in iOS v bistvu potrebujete dve razvojni skupini, kar lahko zviša ceno projekta. [6]

### Mobilne spletne aplikacije (Web applications)

To niso prave aplikacije, ampak dejansko spletna mesta, ki so zasnovana tako, da izgledajo in se počutijo kot mobilna aplikacija. Delujejo v brskalniku vaših pametnih telefonov in so običajno kodirani v CSS, HTML5 ali Javascript. Ker vedno več spletnih mest uporablja HTML5, se meja med spletnimi aplikacijami in spletnimi stranmi zmanjšuje. [5]

Ker bodo vse osebne baze podatkov shranjene na strežniku, lahko uporabniki aplikacijo uporabljajo le, če imajo internetno povezavo. To je glavna pomanjkljivost spletnih aplikacij – vedno zahtevajo dobro internetno povezavo. Poleg tega razvijalci nimajo toliko API-jev, s katerimi delujejo. Učinkovitost bo povezana tudi z delom brskalnika in omrežno povezavo. [6]

### Hibridne mobilne aplikacije

To so aplikacije, ki so delno native in deloma spletna aplikacija. Tako kot native aplikacije jih je mogoče najti v trgovini z aplikacijami in tudi izkoristiti funkcije mobilnega telefona. Toda tako kot spletne aplikacije se za svoje razvojne zahteve zanašajo na HTML5. [5]

Hibridne aplikacije je enostavno in hitro razviti, kar je očitna prednost. Dobite tudi enotno kodno zbirko za vse platforme. To zniža stroške vzdrževanja in poenostavi postopek posodabljanja. Razvijalci lahko izkoristijo tudi številne API-je za funkcije, kot sta »gyroscope« ali »geolocation«. Po drugi strani pa hibridne aplikacije nimajo velike hitrosti in zmogljivosti. Prav tako aplikacija morda ne bo videti enako na dveh ali več platformah. [6]

### »Cross-platform« aplikacije

Namen teh cross-platform aplikacij je rešiti problem hibridne zmogljivosti in težave s stroški pri pisanju različnih maternih jezikov za vsako mobilno platformo. Programerji morajo programirati samo enkrat in prevesti v številne različice native aplikacij, ki ustrezajo vsaki različni platformi. Obstaja veliko cross-platform okvirjev. Vsaka vrsta bo imela različne prednosti in slabosti. Glede na cilj izdelave aplikacije bo programer izbral primeren okvir. [6] )

## Najboljši programski jeziki za razvoj aplikacij za Android

Čeprav je Kotlin uradni jezik za Android, obstaja veliko drugih jezikov, ki se lahko uporabljajo za razvoj aplikacij za Android.

### Java

Najprej je bila Java uradni jezik za razvoj aplikacij za Android (zdaj pa jo je nadomestil Kotlin) in je posledično tudi najbolj uporabljen jezik. Številne aplikacije v Trgovini Play so zgrajene z Javo in je tudi Googlov najbolj podprt jezik.

### Kotlin

Kotlin je zdaj uradni jezik za razvoj aplikacij za Android, ki ga je Google razglasil od leta 2019. Kotlin je cross-platform programski jezik, ki se lahko uporablja kot alternativa Javi za razvoj aplikacij za Android. Leta 2017 je bil predstavljen tudi kot sekundarni »uradni« jezik Java. Kotlin lahko sodeluje z Javo in deluje na Java Virtual Machine. Edina pomembna razlika je v tem, da Kotlin odstrani nekaj funkcije Jave, kot so izjeme ničelnega kazalca. Prav tako odpravlja potrebo po končanju vsake vrstice s podpičjem.

### C++

C++ se lahko uporablja za razvoj aplikacij za Android z uporabo Android Native Development Kit (NDK). Vendar pa aplikacije ni mogoče ustvariti v celoti z uporabo C++ in NDK se uporablja za implementacijo delov aplikacije v izvorni kodi C++. To pomaga pri uporabi knjižnic kod C++ za aplikacijo, kot je potrebno. Čeprav je C++ v nekaterih primerih uporaben za razvoj aplikacij za Android, ga je veliko težje nastaviti in je veliko manj prilagodljiv. [7]

### C#

C# ima čistejšo in enostavnejšo sintakso kot Java, zaradi česar je kodiranje z njim sorazmerno lažje. Prej je bila največja pomanjkljivost C# ta, da je lahko deloval samo v sistemih Windows, saj je uporabljal .NET Framework. Vendar je to težavo rešil Xamarin[[3]](#footnote-3). Android je cross-platform izvedba skupne jezikovne infrastrukture. Zdaj, Xamarin. Orodja za Android je mogoče uporabiti za pisanje native aplikacij za Android in skupno rabo kode na več platformah.

### Python

Android ne podpira native razvoja Pythona. To lahko storimo z različnimi orodji, ki pretvorijo aplikacije Python v pakete Android, ki se lahko izvajajo v napravah Android. Primer tega je Kivy, ki je odprtokodna knjižnica Python, ki se uporablja za razvoj mobilnih aplikacij. Podpira Android in tudi spodbuja hiter razvoj aplikacij. Vendar pa je slaba stran tega, da za Kivy ne bo native prednosti, saj ni izvorno podprt.

### HTML, CSS, JavaScript

Aplikacije za Android je mogoče ustvariti z uporabo HTML, CSS in JavaScript z uporabo ogrodja Adobe PhoneGap, ki ga poganja Apache Cordova. Okvir PhoneGap v bistvu omogoča uporabo veščin spletnega razvoja za ustvarjanje hibridnih aplikacij, ki so prikazane prek »WebView«, vendar so pakirane kot aplikacija. [7]

## Proces načrtovanja mobilne aplikacije

1. Strategija

Prva faza procesa razvoja mobilne aplikacije je definiranje strategije za razvoj ideje v aplikacijo. Ker se cilji ene aplikacije lahko razlikujejo od drugih, še vedno obstaja poseben vpliv na strategijo mobilnosti, ki jo je treba obravnavati med razvojnim procesom.

V tej fazi se:

* Identificira uporabnike aplikacije
* Razišče konkurenco
* Določi cilje in cilje aplikacije
* Izberemo mobilno platformo za aplikacijo

1. Analiza in načrtovanje

Analiza in načrtovanje se začneta z definiranjem primerov uporabe in zajemanjem podrobnih funkcionalnih zahtev. Določi se zahteve za aplikacijo in pripravi se načrt izdelka. Del faze načrtovanja vključuje prepoznavanje veščin, potrebnih za razvoj aplikacije. Na primer, mobilne platforme iOS in Android uporabljajo različne sklade razvojne tehnologije. Če je cilj izdelati mobilno aplikacijo za mobilno platformo iOS in Android, bo potrebna ekipa za razvoj mobilnih naprav vključevati razvijalce za iOS in razvijalce za Android. Izbere se tudi ime aplikacije.

1. Oblikovanje uporabniškega vmesnika / UX

Namen zasnove aplikacije je zagotoviti brezhibno in enostavno uporabniško izkušnjo z izpopolnjenim videzom. Uspeh mobilne aplikacije je določen glede na to, kako dobro uporabniki sprejemajo in izkoristijo vse njene funkcije. Cilj oblikovanja uporabniškega vmesnika/UX mobilnih aplikacij je ustvarjanje odlične uporabniške izkušnje, zaradi česar bo vaša aplikacija interaktivna, intuitivna in uporabniku prijazna.

1. Razvoj aplikacij

Načrtovanje ostaja sestavni del te faze v procesu razvoja mobilnih aplikacij. Tipičen projekt mobilne aplikacije je sestavljen iz treh sestavnih delov: zaledne/strežniške tehnologije, API-jev in sprednjega dela mobilne aplikacije.

Back-End/Strežniška tehnologija

Ta del vključuje bazo podatkov in objekte na strani strežnika, ki so potrebni za podporo funkcij vaše mobilne aplikacije. Če uporabljate obstoječo zaledno platformo, bodo morda potrebne spremembe za podporo želene mobilne funkcionalnosti.

API

Vmesnik za programiranje aplikacij (API) je način komunikacije med aplikacijo in zadnjim strežnikom/bazo podatkov.

Front-End mobilne aplikacije

Sprednji del je izvorna mobilna aplikacija, ki jo bo uporabljal končni uporabnik. V večini primerov so mobilne aplikacije sestavljene iz interaktivnih uporabniških izkušenj, ki uporabljajo API in ozadje za upravljanje podatkov.

Za ozadje lahko uporabite skoraj kateri koli spletni programski jezik in baze podatkov. Aplikacije za iOS je mogoče razviti s programskim jezikom Objective-C ali Swift. Aplikacije za Android so zgrajene predvsem z uporabo Jave ali Kotlina.

1. Testiranje

Izvajanje temeljitega testiranja zagotavljanja kakovosti (QA) med procesom razvoja mobilnih aplikacij naredi aplikacije stabilne, uporabne in varne. Poznavanje funkcionalnih zahtev in ciljev aplikacije bo pomagalo izdelati natančne testne primere. Izvajanje temeljitega testiranja zagotavljanja kakovosti med procesom razvoja mobilnih aplikacij naredi aplikacije stabilne, uporabne in varne:

* Testiranje uporabniške izkušnje
* Funkcionalno testiranje
* Testiranje zmogljivosti
* Testiranje naprav in platform

1. Uvajanje in podpora

Za izdajo domače mobilne aplikacije morate svojo aplikacijo poslati v trgovine z aplikacijami, Apple App Store za aplikacije iOS in Google Play za aplikacije za Android.

Za izdajo aplikacije v trgovini z aplikacijami je potrebna priprava metapodatkov, vključno z:

* Naslovom aplikacije
* Opisom
* Kategorijami
* Ključnimi besedami
* Ikono za zagon
* Posnetki zaslona

[8]

# NAČRTOVANJE IN REALIZACIJA APLIKACIJE MALI POMOČNIK

Že v samem začetku sem se odločila, da bom za aplikacijo uporabila programski jezik Javo, kajti mi je ta najbolj poznan. Izmed nešteto orodij za razvoj programske opreme za android, sem izbrala Android Studio. Orodje naj bi bilo prva izbira za veliko drugih Android razvijalcev. To je uradno integrirano okolje za razvoj aplikacij za Android, ki omogoča enostavno urejanje kode, odpravljanje napak in testiranje.

## Načrt aplikacije

Slika Načrt aplikacije - Mali pomočnik

Namen programa je pomoč pri sprostitvi, za kar je namenjen predvajalnik glasbe. Časovnik je pomoč uporabniku za učenje, ki odšteva 25 minut. V glavnem programu je prikazan izpis podatkov uporabnika, ki je prijavljen. Pred vstopom v glavni program se mora uporabnik registrirati. V primeru, da je bil že predhodno registriran se le prijavi, kar mu omogoči dostop do glavnega programa. Načrt programa je prikazan na sliki 1.

## Načrt izgleda aplikacije

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenOrodje Android studio, nam omogoča oblikovanje izgleda aplikacije, brez potrebe prenašanja dodatnih orodij. Na sliki 2 so prikazane vse XML[[4]](#footnote-4) datoteke za prijavo, registracijo, časovnik, medijski predvajalnik in XML datoteka za prikaz domače strani programa, kjer so ustvarjeni gumbi za vse glavne funkcije.

Slika Datoteke XML za prikaze na zaslonu

## Izdelava XML datotek

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenXML je označevalni jezik, podoben HTML, vendar brez vnaprej določenih oznak za uporabo. Namesto tega lahko določamo svoje lastne oznake, zasnovane posebej za naše potrebe. Android Studio ob vsaki novi ustvarjeni aktivnosti (ang. Activity), sam ustvari novo XML datoteko z istim naslovom kot ga ima razred.

Slika Urejanje gumbov v XML

Vsakemu besedilnemu polju, gumbu, sliki, dodelimo svoj id, preko katerega ga lahko v razredu povežemo z tam ustvarjenimi spremenljivkami. Vsako polje mora imeti različen id.

Zaradi veliko preusmerjanja na različna okna to najlažje storimo z gumbi. V XML datoteki jih deklariramo kot »Button«, kot na sliki 3. Imamo tudi možnost izbire slikovnega gumba, kateremu lahko dodelimo sliko iz mape »drawable«. Navadnim gumbom se lahko dodeli tudi besedilo.

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjen Za besedilna polja največ uporabljamo »TextView« in »EditText« (slika 4). Polje za urejanje je večinoma uporabljeno v forumih za registracijo in prijavo, kjer želimo, da uporabnik vpiše svoje podatke.

Slika Urejanje besedilnih polj v XML

Na slikah 3 in 4 je prikazano tudi urejanje položaja (»layout«), s katerim lahko dodamo obrobo besedilu ali gumbu, layout nam tudi omogoča povezavo z drugim besedilom ali gumbom, če želimo da sta ta v položaju povezana. Povežemo jih s klicem id kot na sliki 5.



Slika Prikaz povezave dveh spremenljivk v XML

## Ustvarjanje podatkovne baze

Slika, ki vsebuje besede besedilo, posnetek zaslona, črna, zaslon

Opis je samodejno ustvarjenČe želimo, da bo aplikacija imela funkcionalen form za registracijo in prijavo, je potrebno implementirati podatkovno bazo. Najenostavnejši način je z orodjem Firebase. Android studio nam implementiranje olajša, saj v zavihku orodja (slika 6), imamo na voljo Firebase. Google Firebase je programska oprema za razvoj aplikacij, ki jo podpira Google, razvijalcem omogoča razvoj aplikacij za iOS, Android in spletne aplikacije. Firebase ponuja orodja za sledenje analitiki, poročanje in odpravljanje zrušitev aplikacij.

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjen

Slika Prikaz Firebase v zavihku "Tools"

Slika Koda razreda "User"

Za povezavo med bazo podatkov in aplikacijo, je bilo potrebno implementirati Firebase Authentication SDK. Po povezavi s Firebase, nadaljujemo z izdelavo novega razreda, v našem primeru imenovan »User«.

Razredu dodelimo atribute katere želimo, da se pokažejo v naši bazi podatkov. Dodelimo mu še prazen konstruktor in konstruktor v katerega pošljemo iste atribute kot razredu, v 2. konstruktorju atribute razreda in konstruktorja povežemo s seboj s pomočjo ukaza »this.«. Naslednji korak je ustvarjanje projekta na orodju Firebase (slika 7).

### Registracija

Ob registraciji od uporabnika želimo izvedeti nekaj podatkov (polno ime, vzdevek, e-mail in geslo) glej sliko 9. Za povezavo vseh gumbov in vnosnih polij XML datoteki in razredu »Signin«, vsakemu izmed polij dodamo svoj id, katerega v razredu kličemo s pomočjo findViewById(R.id. »id željenega polja«). Pod vsemi vnosnimi polji dodamo tudi 2 gumba. Prvi gumb nam omogoča registracijo, s pritiskom na drugi gumb pa se odpre stran za prijavo. Zaradi gumbov je potrebno v razred Signin implementirati »View.OnClickListener« (slika 8).



Slika Prikaz implementacije View.OnClickListener v razred

Slika Prikaz postavitev vnosnih oken in gumbov, v oknu za registracijo

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenTa nam omogoča uporabo metode »OnClick«, s katero lahko ob pritisku na gumb ustvarjamo željena dejanja. Ob pritisku na »REGISTER« gumb kličemo metodo »registerUser«. V metodi na začetku navedemo pogoje, v primeru, da uporabnik pusti katerokoli polje prazno, dodamo še pogoja za preverjanje dolžine gesla in preverjanje ali je e-mail realen.

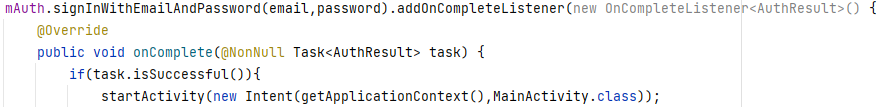
Slika Prikaz kode, za vpisovanje uporabnika v bazo

Na sliki 10, ustvarimo novega uporabnika z geslom in emailom. V primeru, da se je uporabnik uspešno registriral (if(task.isSuccessful)), se v bazo podatkov vpišejo njegovi podatki. Ob uspešni registraciji se na ekranu pokaže »Register successfull!«, če je registracija neuspešna pa se pojavi »Register failed!«.

### Prijava

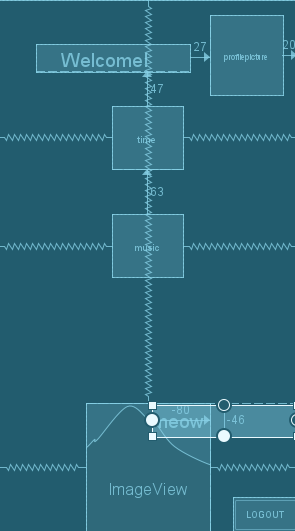
Prijava uporabnika je zelo podobna registraciji. Prav tako ima dva gumba, enega za prijavo in drugega, ki odpre okno za registracijo v primeru, da uporabnik še ni ustvaril svojega profila. Za prijavo bo uporabnik rabil vpisati le e-mail in geslo s katerim se je predhodno registriral. Prav tako v razred »Login« implementiramo »View.OnClickListener«. Z pritiskom gumba za prijavo kličemo metodo »userLogin«.

Slika Prikaz dela metode userLogin

Na sliki 11 vidimo, da ob uspešni prijavi, trenutnemu uporabniku odpre okno glavnega dela programa. Zaradi orodja Firebase nam ni potrebno preverjati ali se e-mail in geslo ujemata z registriranim uporabnikom, saj nam to omogoča orodje samo, s klicem »mAuth.signInWithEmailAndPassword«, glej sliko 11.

## Domače okno aplikacije

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenNa sliki 12 imamo prikazane 4 glavne gumbe in sicer timer, music, profilepicture in logout. Prvi trije omenjeni gumbi so v XML datoteki naslovljeni kot ImageButton, logout pa je naslovljen kot navadni gumb. Vsi gumbi so namenjeni temu, da ob pritisku na njih odprejo določena okna. Z logout gumbom kličemo metodo s pomočjo FirebaseAuth, signOut(), ki uporabnika odjavi iz profila in ponovno odpre okno za prijavo, prikazano na sliki 13.

Slika Prikaz načrta domačega okna

Slika Prikaz kode ob pritisku na gumb logout

### Predvajalnik glasbe

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenKo v domačem oknu pritisnemo na gumb za predvajalnik glasbe nas usmeri na novo okno, kjer se rečeni predvajalnik nahaja. Tukaj metode kličemo že preko XML datoteke, ko na gumbu kličemo »onClick = »imeMetode«, glej sliko 14. To naredimo za vse 4 gumbe. Klicane metode ustvarimo v razredu »MusicPlayer«.

Slika Primer gumba za začetek predvajanja v XML datoteki

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjen

Slika Prikaz metode play

V deklariramo novo spremenljivko tipa MusicPlayer. Metode kličemo v XML datoteki, ker želimo da metoda »OnCreate« ostane prazna, saj je to metoda ki se prva izvede ob odprtju okna. V primeru, da bi imeli predvajalnik glasbe v metodi OnCreate bi ta zavzel preveč prostora na pomnilniku.

Z metodo »play«, prikazano na sliki 15, ustvarimo MediaPlayer, z pomirjajoči glasbo, ki ji imamo shranjeno v mapi raw. Ob koncu glasbe se MediaPlayer samodejno ustavi. V primeru, da mediaPlayer ni prazen, začne predvajati glasbo.

Metoda »pause« nam v primeru, da MediaPlayer ni prazen omogoči ustaviti glasbo in jo predvajati naprej po izbiri. Nazadnje nam metoda »stop« izpusti MediaPlayer, da ne zavzema več prostora na pomnilniku. Njegovo vrednost nastavi na null. Zadnji gumb, back to home, nas preusmeri nazaj na domačo stran aplikacije.

### Časovnik

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenZa delujoč časovnik je potrebno deklarirati nekaj novih spremenljivk, glej sliko 16.

Slika Potrebne spremenljivke za časovnik

V datoteki XML smo za začetek odštevanja in premor ustvarili isti gumb (»StartPausebutton«). Ob pritisku na gumb, preverimo ali se čas že odšteva, v primeru da se, kličemo metodo »pauseTimer«, drugače se kliče metoda »startTimer« (slika 17).

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjen

Slika Prikaz metode StartTimer

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenMetoda odšteva čas na določen interval, ki je nastavljen na 1 sekundo. Vsako sekundo preverja koliko časa je ostalo in tako posodobi čas na zaslonu. Ko se nastavljeni čas izteče, se gumb spremeni v »Start«, postane neviden. Na zaslonu postane viden reset gumb. Med tem ko se čas še izteka se »Start« spremeni v »Pause«, reset gumb postane neviden. Metoda startTimer je prikazana na sliki 17.

Slika Prikaz metode pauseTimer

PauseTimer, prekine odštevanja časovnika in »Pause« spremeni v »Start«, reset gumb ponovno postane viden, glej sliko 18.

Slika, ki vsebuje besede besedilo, notranji, posnetek zaslona

Opis je samodejno ustvarjenZ resetTimer preostali čas nastavimo na prvotnega, reset gumb postane neviden, gumb za začetek pa viden.

Slika Prikaz metode resetTimer

Slika, ki vsebuje besede besedilo

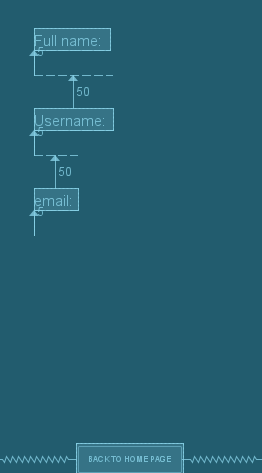
Opis je samodejno ustvarjen

Slika Prikaz metode updateCountDownText

Za točen prikaz odštevanja časa je bilo potrebno ustvariti metodo, glej sliko 20. Z njo čas ki nam ostane pretvorimo v minute in sekunde, dobljene rezultate nato shranimo v nov String.

### Prikaz uporabnikovih podatkov

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenŽelimo, da ob prijavi vsakega posameznega uporabnika aplikacija izpiše podatke tistega, ki je trenutno prijavljen. Ob kliku na gumb profilepicture, se nam odpre okno kjer so vsi zaželeni podatki izpisani, glej sliko 22. To dosežemo s pomočjo metod orodja Firebase, kot vidimo na sliki 21.

Slika Metode orodja Firebase

Slika Načrt okna ProfileAcc

Naslednji korak je ustvarjanje novega objekta tipa User.

V primeru, da objekt, ki smo ga ustvarili ni prazen, mu dodelimo atribute, ki so shranjeni v razredu User. Nato spremenljivkam spremenimo tekst z določenimi lastnostmi trenutnega uporabnika (prikazano na sliki 23).

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjen

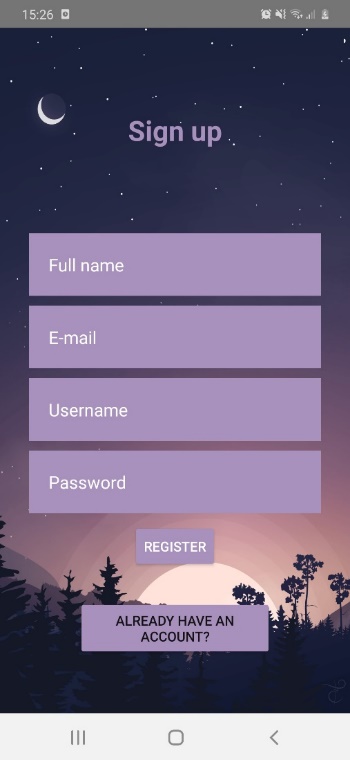
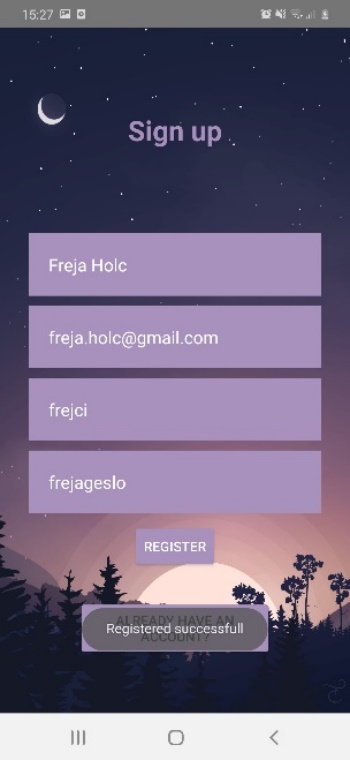
Slika Povezovanje izpisa in podatkovne baze

## Testiranje aplikacije

Ob zagonu aplikacije se nam odpre form za registracijo. Ob vpisu podatkov uporabnika in pritiskom na gumb se nam pokaže sporočilo, da je bila registracija uspešna, prikazano na sliki 24 in 25.

Slika Dokončan form za registracijo

Slika Uspešna registracija novega uporabnika

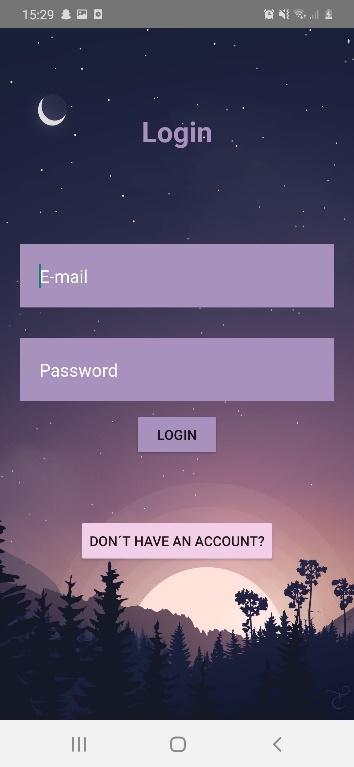
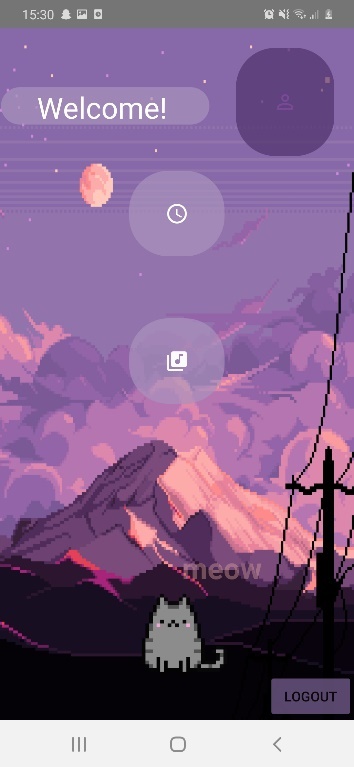


Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjenKo odpremo naš projekt v Firebase in preverimo real time database, se v bazo vpišejo uporabnikovi podatki, z dodeljenim UID ( slika 26).

Slika Izpisani podatki uporabnika v bazi podatkov

Ob uspešni registraciji se nam odpre prijavno okno (slika 27), kjer vpišemo e-mail in geslo s katerim smo se registrirali. Po prijavi se nam odpre domača stran aplikacije, kjer se nahajajo gumbi do drugih funkcij aplikacije in sicer gumb za prikaz podatkov uporabnika, časovnik, predvajalnik glasbe. Imamo tudi možnost odjave s gumbom logout (slika 28).



Slika Prijavno okno aplikacije

Slika Domača stran aplikacije

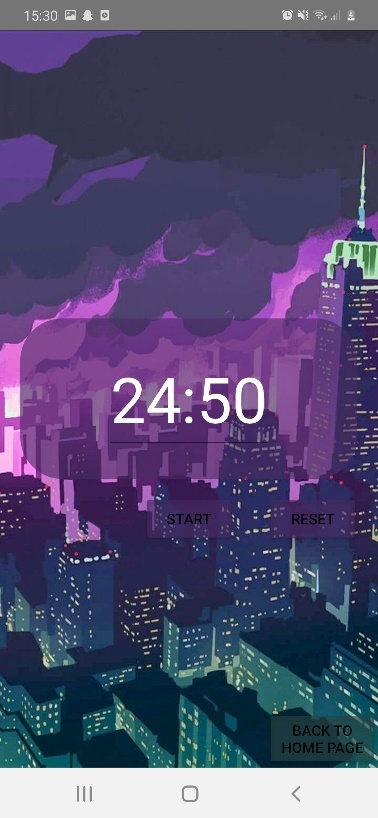
Slika, ki vsebuje besede preslikava

Opis je samodejno ustvarjenNa sliki 29 je prikazan profil trenutno vpisanega uporabnika, kateremu izpiše njegov e-mail, uporabniško ime in polno ime.

Slika Izpisani podatki trenutno prijavljenega uporabnika

Sliki 30 in 31 pa prikazujeta okna za časovnik in predvajalnik glasbe.

Slika, ki vsebuje besede besedilo, narava, oblak, nočno nebo

Opis je samodejno ustvarjen

Slika Okno predvajalnika glasbe

Slika Okno časovnika

# ZAKLJUČEK

Aplikacije so v današnjih časih postale vsakodnevna stvar. Zaradi svoje enostavne dostopnosti in velikega števila jih uporabljamo za pomoč pri vsakdanjih opravilih, kot so opomniki, budilke, kalkulatorji, beležke. Vsak posameznik ima zaradi pametnih telefonov vedno pri roki raznorazne pripomočke. Namen seminarske naloge je bil izdelati osnovno mobilno aplikacijo za Android. Posledično je bilo potrebno raziskati katere vrste aplikacij poznamo in katera bo najbolj ustrezala načrtu aplikacije – Mali pomočnik. Vse štiri vrste aplikacij so za lažje razumevanje krajše opisane. Za razvoj mobilnih aplikacij je na voljo veliko programskih jezikov, kot so Kotlin, Java, C#, Python. Izbirala sem med nekaj najpopularnejših jezikov, ki so za lažjo primerjavo prav tako kratko opisani. Ker pa se z razvojem mobilne aplikacije srečujem prvič, je bilo potrebno proces načrtovanja in izdelave podrobneje raziskati. Zadnji korak je bil izbira orodja, v katerem bom izdelovala aplikacijo. Zaradi vseh ugodnosti in priporočil ostalih razvijalcev aplikacij, sem se posledično odločila za Android Studio.

Za začetek izdelave Malega pomočnika, je bilo potrebno načrtovati vse funkcije, ki sem jih v aplikacijo želela vključiti. Ena izmed teh funkcij je bila tudi uvedba registracije in prijave, za katero je bilo potrebno ustvariti in povezati bazo podatkov. Ob raziskovanju implementacije podatkovne baze, je bil splet zasičen z veliko različnimi načini. Mojemu cilju je najbolj ugajalo orodje Firebase, ki je povezan tudi z Android Studio, zato je bil proces implementacije veliko lažji. Ker je namen Malega pomočnika, pomoč pri sprostitvi in učenju, sta bili naslednji uvedeni funkciji časovnik in predvajalnik glasbe. Zadnja implementirana stvar je bila izpis podatkov trenutno vpisanega uporabnika.

Trenutno stanje aplikacije ni še namenjeno končnemu uporabniku, saj bi mu bile funkcije nerazumljive in nejasne. Z razvojem Malega Pomočnika bi nadaljevala z nadgradnjo trenutnega izgleda in sposobnosti, ki jih nudi. S časoma bi bilo dodanih vedno več funkcij, ki bi uporabniku pomagale z razbremenjevanjem stresa in razbremenjevanjem nasičenosti informacij, ki posameznika »bombardirajo« v vsakdanjem življenju.

# ZAHVALA

Za pomoč pri seminarski nalogi bi se zahvalila mentorju Gregorju Medetu za vse spodbudne besede in svetovanju. Zahvalila bi se tudi vsem sošolcem, s katerimi sem se lahko posvetovala ko sem sama obtičala. Hvala Adamu, ki je pazil, da se med delom seminarske naloge ne zamotim preveč. Nazadnje pa sem najbolj hvaležna svoji mami, ki je testirala vse verzije moje aplikacije in poslušala moje nenehno razlaganje o projektu, čeprav ga ni razumela. Kljub vsemu me je spodbujala, da nadaljujem tudi ko sem želela obupati. Najiskrenejše se vam zahvaljujem.

# VIRI IN LITERATURA

|  |  |
| --- | --- |
| [1] | aws, „aws amazon,“ 10 april 2022. [Elektronski]. Available: https://aws.amazon.com/mobile/mobile-application-development/. |
| [2] | C. Hope, „Computer Hope,“ 6 april 2022. [Elektronski]. Available: https://www.computerhope.com/jargon/s/smartphone.htm. |
| [3] | A. Provazza, „techtarget,“ 9 apirl 2022. [Elektronski]. Available: https://www.techtarget.com/searchmobilecomputing/definition/smartphone. |
| [4] | cellularsales, „cellularsales,“ 11 april 2022. [Elektronski]. Available: https://www.cellularsales.com/blog/a-mostly-quick-history-of-smartphones/. |
| [5] | outsource2india, „outsource2india,“ 10 april 2022. [Elektronski]. Available: https://www.outsource2india.com/software/mobile-applications/tools-technologies.asp. |
| [6] | A. Mroczkowska, „droidsonroids,“ 8 april 2022. [Elektronski]. Available: https://www.thedroidsonroids.com/blog/what-is-a-mobile-app-app-development-basics-for-businesses. |
| [7] | GeeksforGeeks, „GeeksforGeeks,“ 9 april 2022. [Elektronski]. Available: https://www.geeksforgeeks.org/top-programming-languages-for-android-app-development/. |
| [8] | Invonto, „Invonto,“ 9 april 2022. [Elektronski]. Available: https://www.invonto.com/insights/mobile-app-development-process/. |

# STVARNO KAZALO

A

aplikacija · 1, 4, 5, 7, 8

Č

časovnik · 10, 14, 17

Časovnik · 10

F

Firebase · 10, 11, 13

funkcija · 3

M

metoda · 14

metode · 12

Metode · 13

P

podatkovna baza · 3

podatkovno bazo · 10

program · 4, 6

programski jezik · 6

programski opremi · 2

programskih aplikacij · 1

X

XML · 11, 13

# PRILOGE

1. Priloga : Povezava do celotne kode programa <https://github.com/zoviii>
2. Slika, ki vsebuje besede besedilo, oranžna, temno

   Opis je samodejno ustvarjenPriloga: E-mail in geslo, ki omogoča dostop do podatkovne baze na spletnem portalu <https://firebase.google.com/> :

1. Information Technology [↑](#footnote-ref-1)
2. application programming interface [↑](#footnote-ref-2)
3. Xamarin je Microsoftovo podjetje s programsko opremo [↑](#footnote-ref-3)
4. Extensible Markup Language [↑](#footnote-ref-4)