MOBILNA ANDROID APLIKACIJA ZA POTPORU PRILAGODLJIVOM UČENJU

Završni rad

Branimir Kedačić

SADRŽAJ

1. UVOD .......................................................................................................................... 1
   1.1. ZADATAK ZAVRŠNOG RADA ............................................................................ 1

2. PROBLEM I PROGRAMSKI SLIČNE MOBILNE APLIKACIJE ........................................... 2
   2.1. POJAM UČENJA ............................................................................................... 2
   2.2. MOBILNO ILI M-UCENJE .............................................................................. 3
   2.3. PREDNOSTI I NEDOSTACI M-UCENJA .......................................................... 4
   2.4. SLIČNE APLIKACIJE .................................................................................... 5
   2.4.1. Aplikacija Cram ....................................................................................... 6
   2.4.2. Aplikacija AnkiDroid ............................................................................. 6
   2.4.3. Aplikacija Quizlet ................................................................................... 7
   2.4.4. Aplikacija StudyDroid ......................................................................... 7

3. IDEJNO RJEŠENJE MOBILNE APLIKACIJE .................................................................. 9
   3.1. DIJAGRAM RADA APLIKACIJE ................................................................... 9
   3.2. ARHITEKTURA BAZE PODATAKA .................................................................. 10
   3.3. MODEL SUČELJA I KORISNIČKIH OPCIJA ...................................................... 10

4. PROGRAMSKO RJEŠENJE MOBILNE APLIKACIJE ....................................................... 13
   4.1. ANDROID OS .................................................................................................. 13
   4.1.1. Povijest Androida ................................................................................. 13
   4.1.2. Arhitektura Androida ......................................................................... 14
   4.2. KORIŠTENI ALATI I TEHNOLOGIJE ............................................................... 15
   4.2.1. Android Studio ..................................................................................... 15
   4.2.2. Programski jezik Java .......................................................................... 16
   4.2.3. Android SDK ......................................................................................... 17
   4.2.4. Android AVD .......................................................................................... 17
   4.2.5. SQLite ....................................................................................................... 18
   4.3. KLJUČNI DIJELOVI KODA ............................................................................ 19
   4.3.1. API razine .............................................................................................. 19
   4.3.2. Baza podataka .......................................................................................... 19
   4.3.3. Registriranje i prijava korisnika ............................................................... 21
   4.3.4. Dodavanje skupova i kartica .................................................................. 23
   4.3.5. Brisanje kartica ....................................................................................... 25
   4.3.6. Prikaz reultata ........................................................................................ 26
   4.3.7. Kod za kviz .............................................................................................. 26

5. KORIŠTENJE I TESTIRANJE MOBILNE APLIKACIJE .................................................. 30
   5.1. PRIJAVA I REGISTRACIJA KORISNIKA ....................................................... 30
   5.2. GLAVNI IZBORNIK ...................................................................................... 31
   5.3. SKUPOVI I KARTICE ....................................................................................... 31
   5.4. PREGLED PITANJA ...................................................................................... 33
   5.5. BRISANJE KARTICA .................................................................................... 34
   5.6. KVIZ ............................................................................................................. 35
   5.7. PRIKAZ REZULTATA ................................................................................... 36
   5.8. TESTIRANJE APLIKACIJE .......................................................................... 37

6. ZAKLJUČAK ............................................................................................................... 40
1. UVOD

Napredak u tehnologiji je donio puno novina na području obrazovanja i učenja. Kroz godine su se razvijali različiti načini koji olakšavaju učenje. Prema [1], trenutno je mobilno učenje jedna od najčešće korištenih načina učenja. Danas gotovo svi posjeduju pametne telefone i koriste ih svakodnevno, a digitalizacijom podataka ljudi se rješavaju papira i sve više literature prevode u digitalni oblik. Veliki broj učenika i studenata koristi svoj pametni telefon koji ima uvijek uz sebe u svrhu učenja te tako djelomično zamjenjuju papirnate materijale digitalnim. Mobilno učenje omogućuje korištenje pametnog telefona bilo kada i bilo gdje kako bi se moglo učiti u pokretu i kako bi se svaki slobodan trenutak iskoristio što kvalitetnije i učinkovitije.

Cilj ovog završnog rada je izrada mobilne aplikacije za potporu prilagodljivom učenju na principu kartica. Mobilna aplikacija treba pružiti cjeļoviti sustav preko kojeg će se korisnik moći registrirati i prijaviti, te kreirati svoje skupove kartica pomoću kojih će moći testirati i pratiti svoje znanje i napredak u učenju.

U drugom poglavlju obrađuju se pojmovi kao što su mobilno učenje, kako se rješavaju problemi takvog učenja te se navode i analiziraju metode učenja i već poznate aplikacije koju su namijenjene rješavanju tog problema. Zatim se u trećem poglavlju razrađuje idejno rješenje aplikacije. Tu se navode ključni dijelovi aplikacije i način njenog ostvarenja. U četvrtom poglavlju opisuje se okolina i platforma koja će se koristiti za izradu aplikacije, te opis programskog rješenja i opis ključnih dijelova koda. U petom poglavlju se govori o načinima korištenja aplikacije i prikazuje testiranje različitih scenarija kako bi se prikazala ispravnost same aplikacije.

1.1. Zadatak završnog rada

U radu treba opisati najučinkovitije oblike učenja i poučavanja s naglaskom na prilagodljivom e-ucišenje u mobilnim okolinama. Također, treba osmislići model mobilne aplikacije (s bazom podataka) za Android okolinu koja će omogućiti definiranje sadržaja za učenje, praćenje uspješnosti savladavanja građiva i na temelju toga prilagodbu okoline za učenje. Mobilnu aplikaciju potrebno je programski ostvariti, ispitati na dovoljnom skupu primjera i prikladno analizirati s gledišta funkcionalnosti i korisničkog iskustva.
2. PROBLEM I PROGRAMSKI SLIČNE MOBILNE APLIKACIJE

2.1. Pojam učenja

Prema [2], učenje je složeni psihički proces promjene ponašanja na osnovi usvojenog znanja i iskustva. To je proces uskladištavanja podataka u skladištu pamćenja. Učenje se zapravo sastoji od više aktivnosti. Prema [3], neke od tih aktivnosti su:

- prikupljanje informacija - informacije se mogu prikupljati iz raznih izvora. Mogu se čuti od profesora ili pročitati u knjigama. Informacije se također mogu dobiti i s televizije, s interneta, novinama i časopisima.
- bilježenje informacija – sve informacije koje se dobiju moraju se zabilježiti na neki način, jer će se u suprotnom vrlo brzo zaboraviti.
- organizacija informacija – kako se informacije primaju iz različitih izvora, na različit način i u različito vrijeme, bitno je da se svi ti podaci organiziraju kako bi se kasnije lakše moglo snaći u svim tim podacima.
- razumijevanje informacija - sve informacije koje se dobiju, zabilježe i organiziraju moraju se razumjeti i shvatiti da način da postoji potpuno razumijevanje svih podataka. Ako se nešto nauči površno, moći će se ispričati samo glavne činjenice, ali ako se zaista razumije to što je naučeno moći će se shvatiti i iznijeti dublje značenje informacije.
- pamćenje – bitno je naći pravi način kako zapamtit svе te informacije. Dakle, mora se odabrati ono što je najvažnije i pronaći prikladan način učenja kako bi se moglo svjesno odlučiti što zapamtit.
- korištenje - sve naučeno mora se moći i upotrijebiti. Mora se znati objasniti, riješiti problem, pisati o tome ili raspravljati s nekim. Korištenjem informacije osigurava se da se naučeno ne zaboravi. Slika 2.1 prikazuje proces učenja.

Slika 2.1. Prikaz procesa učenja
2.2. Mobilno ili m-učenje


Slika 2.2. Prikaz mjesto korištenja m-učenja
2.3. Prednosti i nedostaci m-učenja

Mobilno učenje kao i e-učenje ima veliki broj prednosti u odnosu na klasičan način učenja, ali također ima i neke nedostatke koji ukazuju da će trebati određeni vremenski period da se ti oblici učenja dovedu do razine koja će biti zadovoljavajuća i prihvatljiva svim korisnicima.

Prema [6], prednosti mobilnog učenja su:

- Fleksibilnost: učenje nije ovisno o vremenu i mjestu gdje se korisnik nalazi.
- Raspodjela građiva: sadržaj koji korisnici uče je podijeljen na više manjih dijelova, što kao rezultat daje veću aktivnost i veći postotak prijeđenog građiva jer je većini korisnika lakše odvojiti deset do petnaest minuta više puta dnevno, nego jedan sat u komadu.
- Prilagodljivost: mogućnost određivanja svog tempa učenja (korisnik može pregledati materijale za učenje neograničeni broj puta, kada i na koji način to njemu odgovara). Korisnik može odabrati razinu interakcije s aplikacijom i koristiti razna multimiđijska sredstva (animacije, grafike, zvukove) kako građivo ne bi bilo monotono, a učenje zanimljivije. Također su im na raspolaganju predmeti i tečajevi koje nude različite institucije pa si korisnik može odabrati onu koja mu najviše odgovara ili više njih.
- Dostupnost: korisnik može pronaći različite programe koje ga zanimaju, čak i u slučaju kada ga njegova škola, fakultet ili poslovna institucija ne nude. Mnogi prestižni fakulteti i institucije nude mogućnost sudjelovanja u njihovim programima i to bez plaćanja.

Prema [7], nedostaci mobilnog učenja su:

- Ometanje: pri učenju na mobilnom uređaju vrlo je lako skrenuti pažnju s jedne stvari i prebaciti je na drugu. Ometanja mogu biti u obliku SMS poruka, raznih obavijesti, poziva i igrica.
- Trajanje baterije: korištenje raznih aplikacija za učenje, može uzrokovati smanjenje trajanja rada mobilnog uređaja, što može biti nezgodno ako je korisnik u pokretu i nemam gdje napuniti uređaj. Ovaj problem se donekle može riješiti posjedovanjem prijenosnog punjača.
- Memorija: ako se želi koristiti izvanmrežni način rada za učenje, može se zahtijevati preuzimanje do nekoliko gigabyte-a podataka na mobilni uređaj što nekim korisnicima, osobito onima sa starijim uređajima može predstavljati problem.
• Vid: današnji mobilni uređaji imaju prilično velike zaslone, a dugotrajno gledanje u njih uređaja može biti naporno i štetno za oči.

Prema [7], može se vidjeti usporedbu aktivnosti i karakteristika procesa učenja kroz tri različita pristupa prikazana na slici 2.3:

<table>
<thead>
<tr>
<th></th>
<th>Mučenje</th>
<th>E-učenje</th>
<th>Klasično učenje</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uporaba multimedia i interaktivnih alata</td>
<td>Da, s nekim ograničenjima</td>
<td>Da</td>
<td>Ograničeno</td>
</tr>
<tr>
<td>Prilika za učenje bilo gdje</td>
<td>Da</td>
<td>Ne</td>
<td>Da</td>
</tr>
<tr>
<td>Automatski samoevaluacijski testovi</td>
<td>Da</td>
<td>Da</td>
<td>Ne</td>
</tr>
<tr>
<td>Potrebna internetska veza</td>
<td>Većinom</td>
<td>Većinom</td>
<td>Ne</td>
</tr>
<tr>
<td>Evaluacija učenja</td>
<td>Da</td>
<td>Da</td>
<td>Ne</td>
</tr>
<tr>
<td>Kontakt s predavačem i drugim polaznicima</td>
<td>Ograničen</td>
<td>Ograničen</td>
<td>Da</td>
</tr>
<tr>
<td>Brze informacije i vljeti</td>
<td>Da</td>
<td>Ograničen</td>
<td>Ne</td>
</tr>
</tbody>
</table>

Slika 2.3. Usporedba različitih načina učenja

2.4. Slične aplikacije

Prije početka izrade mobilne aplikacije potrebno je obaviti istraživanje u kojem će se pregledati slične aplikacije. S obzirom da se radi o izradi Android mobilne aplikacije, za to istraživanje će se koristiti Google-ov Play Store. Play Store je on-line katalog mobilnih aplikacija namijenjen operacijskom sustavu Android. Na slici 2.4 može se vidjeti prikaz traženja aplikacije.

Slika 2.4. Prikaz rezultate s Google Play-a
2.4.1. Aplikacija Cram

Prema [8], Cram je aplikacija dostupna za Android i Apple uređaje. Ona omogućuje spajanje na web stranicu gdje se potom mogu pretraživati kartice već postavljene od drugih korisnika, kojih trenutno ih ima oko 75,000,000. U slučaju da ipak ne postoje kartice koje korisniku trebaju, može ih napraviti sam. Cram sadrži tri načina učenja, normalan način okretanja kartica, zatim Memorize mode gdje se prolazi kroz svaku karticu dok se ne dođe do kraja i zadnji način je Cram mode gdje korisnik odgovara na svaku karticu, u slučaju da pogriješi kartica će se nakon nekog vremena ponoviti i tako sve dok korisnik ne odgovori točno na sva pitanja. Na slici 2.5 može se vidjeti prikaz opcija koje Cram nudi.

Slika 2.5. Prikaz opcija aplikacije Cram

2.4.2. Aplikacija AnkiDroid

2.4.3. Aplikacija Quizlet

Prema [10], aplikacija Quizlet je dostupna samo na Android platformi. Za korištenje aplikacije potrebno je registrirati se i napraviti korisnički račun. Kao i Cram, Quizlet omogućuje korištenje već napravljenih skupova kartica, a također podržava i izvanmrežni način rada. Moguće je pratiti postotak točno i netočno odgovorenih pitanja, i vrijeme potrebno za prelaženje jednog skupa. Quizlet pruža još i dodatne načine učenja kao što su opcija da korisnik sam mora unositi tekstualni odgovor ili klasična igra „Memory“ vidljiva na slici 2.7, sastavljena od kreiranih kartica.

2.4.4. Aplikacija StudyDroid

Prema [9], StudyDroid je jedna od jednostavnijih aplikacija za korištenje. Za neke korisnike to će značiti prednost nad ostalim aplikacijama jer nije potrebno trošiti vrijeme na istraživanje kako aplikacija radi. Naravno, to ne znači da zaostaje za drugima po svojim mogućnostima.
Zanimljiva opcija ove aplikacije je da postoji mogućnost označavanja kartice kao naučena, što će ju prebaciti na kraj skupa. StudyDroid osim besplatne inačice vidljive na slici 2.8, ima i plaćenu inačicu koja otključava mogućnosti poput mijenjanja veličine i izgleda slova, te sinkroniziranja s web stranicom Quizlet aplikacije.

Slika 2.8. Prikaz opcija aplikacije StudyDroid
3. IDEJNO RJEŠENJE MOBILNE APLIKACIJE

3.1. Dijagram rada aplikacije

S ciljem lakšeg ostvarenja aplikacije, kreiran je dijagram tijeka koji prikazuje zamišljeni tijek i funkciju ove mobilne aplikacije. Prema slici 3.1 moguće je vidjeti da bi se korisnik nakon pokretanja aplikacije trebao prijaviti ili registrirati kako bi imao mogućnost daljnog korištenja aplikacije. Nakon uspješne prijave, korisnik ima mogućnost odabira različitih opcija s početnog zaslona. Ima mogućnost prikaza podataka koji su već spremljeni u bazu, zatim dodavanja novih podataka i brisanje starih podataka iz baze. Tu se nalazi i opcija pokretanja kviza, pri čemu se bira željeni skup s pripadajućim karticama i kreće testiranje znanja. Aplikacija bilježi broj točnih i netočnih odgovora, te na kraju prikazuje taj omjer brojčano i u postotcima.

Slika 3.1. Prikaz opcija aplikacije
3.2. Arhitektura baze podataka

Za potrebe izrade ove aplikacije, trebalo je osmisлити i projektirати базу података. База података садржи четири таблице коje су ključne за pravilno funkcioniranje, a to su tablica Korisnici, Setovi, Kartice i Rezultati. Svrha таблици Korisnici je да се u nju записују подаци прilikom registracije korisnika. Prilikom registracije korisnik je obavezан unijeti e-mail i zaporку, коjо se spremaju у Korisnik_Email односno Korisnik_Password, a obavezni су prilikom prijave korisnika. Atribut Korisnik_ID je primarni ključ i se koristi за razlikovanje korisnika, što značи да се сваки korisnik imati jedinstven ID. Atribut Korisnik_Logged прати да ли је korisnik prijavljen или не. Таблица Setovi se koristi за spremanje skupova pitanja. Set_ID je primarni ključ коjо daje svakом skupu jedinstveni броj, те е svaki skup imenovan помоћу Ime_Seta. Korisnik_ID_FK je strani ključ коjо se povezuje на Korisnik_ID. Таблица Kartice se sastoji од primarnог ključа Kartica_ID, ст раног ključа Set_ID_FK коjо se povezuje на Set_ID како bi se kartice mogle spremiti у odgovarajući skup, аттрибута Kartica_Pitanje, Kartica_Odgovor, у которые се спремају пitanja и odgovori, te Kartica_Odgovorena и Kartica_Odgovorena_T_N коjо služe да praćenje и bilježenje odgovora. Таблица Rezultati sadrži Rezultati_ID, strani ključ Korisnik_ID_FK_Rez помоћу коjег se povezuju rezultati kviza sa određenim korisnikom, и атtribute Rezultati_Set и Rezultati_Ukupni коjо se koriste за prikaz rezultata rješenih kvizova. Слика 3.2 prikazuje arhitekturu baze podataka.

![Slika 3.2. Prikaz baze podataka](image)

3.3. Model sučelja i korisničkih opcija

Za izradu modela сučела mobilne aplikacije korišten je program под називом Mockplus. Prema [11], то je računalni program namijenjen за бру израду prototipa aplikacija. Podрžava modele

Na slici 3.3 može se vidjeti početni zaslon aplikacije. On sadrži opcije za prijavu korisnika i za registraciju korisnika. Moguća je prijava već postojećeg korisnika ili odabir opcije registracije novog korisnika.

Slika 3.3. Početni zaslonski pola za prijavu i registraciju

Nakon prijave, korisnik dolazi do glavnog izbornika aplikacije, gdje su mu ponuđene različite korisničke opcije i mogućnosti. Korisnik ima uvid u sve skupove i kartice koji se nalaze u bazi i veoma lako može manipulirati s njima. Zaslon same igre je jednostavan i sadrži tipke koje su korisniku potrebne za korištenje poput okretanja kartice, označavanja točnog ili netočnog odgovora i prelaska na novu karticu. Neki od zaslona i opcija vidljivi su na slici 3.4. To su opcije kreiranja skupova i pregled istih, dodavanje kartica i detaljno pregledavanje pitanja i odgovora, opcije označavanja odgovora u kvizu i prikaz rezultata.
Slika 3.4. Različiti zasloni aplikacije
4. PROGRAMSKO RJEŠENJE MOBILNE APLIKACIJE

Programsko rješenje ove aplikacije izrađeno je na temelju idejnog rješenja, koje je prošireno definiranjem i uvidom u korištene alate i tehnologije potrebne za uspješno ostvarenje mobilne aplikacije.

4.1. Android OS


4.1.1. Povijest Androida

Android te ga nitko nema pravo koristiti sve dok ih on sam ne certificira kao kompatibilnog prema svojim standardima.

4.1.2. Arhitektura Androida

Prema [15], arhitektura Androida se sastoji od više slojeva. Na najnižem sloju se nalazi Linux 2.6. jezgra koja sadrži različite upravljačke programe za međuprosesnu komunikaciju. Najvažniji od njih su IPC upravljački programi za izmjenu podataka između različitih procesa i upravljačkih jedinica za upravljanje napajanjem. Iznad sloja jezgre se nalaze različite knjižice koje su napisane u programskom jeziku C i C++.

Neke od tih knjižica su:

- Surface Manager – knjižnica koja nadzire iscravanje grafičkog sučelja
- OpenGL | ES – knjižnica koja služi za sklopovsko ubrzavanje 3D prikaza i za 3D programsku rasterizaciju
- SGL – 2D knjižnica korištena za većinu aplikacija
- FreeType – knjižnica čija je svrha iscravanje fontova
- SSL (Secure Sockets Layer) - knjižnica za sigurnu komunikaciju putem interneta
- SQLite – knjižnica koja upravlja bazom podataka

Slijedeći sloj je Android okruženje koje sadrži dvije komponente. To su jezgrene knjižice i Dalvik virtualni stroj koji služi za pokretanje aplikacija kao zasebnih procesa. Sloj koji se nadovezuje na ovaj je aplikacijski okvir koji dozvoljava upotrebu svih aplikacijskih programskih sučelja koji su korišteni za bazne aplikacije. Njihovo korištenje omogućava upravljanje programskim paketima, pozivima, prozorima, aktivnostima, lokacijama i resursima. Najviši sloj u ovoj arhitekturi je aplikacijski sloj kojeg čine sve korisničke aplikacije, neke od njih mogu biti ugrađene na uređaj, a veliki broj njih se može naknadno preuzeti. Ovaj sloj je jedini koji je vidljiv krajnjem korisniku. Slika 4.1 prikazuje arhitekturu Android operacijskog sustava.
4.2. Korišteni alati i tehnologije

4.2.1. Android Studio


1. Alatne trake – omogućuju provođenje širokog spektra radnji, uključujući pokretanje aplikacije i pokretanje Android alata.
2. Navigacijska traka - pomaže u kretanju kroz projekt i otvaranju datoteka za uređivanje.
3. Prozor za uređivanje - mjesto gdje se stvara i mijenja kod. Ovisno o trenutnoj vrsti datoteke, prozor se može razlikovati.
4. Prozor s alatima - se kreće rubom prozora integriranog razvojnog okruženja i sadrži gumbe koji omogućuju proširivanje ili sažimanje pojedinačnih prozora alata.

5. Prozori alata - omogućuju pristup određenim zadacima kao što su upravljanje projektima, pretraživanje, kontrola inačice i još mnogo toga.

6. Traka stanja - prikazuje status projekta i sva upozorenja ili poruke.

Slika 4.2. Prikaz sučelja Android Studia

4.2.2. Programski jezik Java

4.2.3. Android SDK

Prema [18], Android SDK je skup alata koji se koriste za razvoj Android aplikacija. Svaki od alata se može preuzeti zasebno pomoću SDK Manager-a. Moguće ga je pokrenuti na svim operacijskim sustavima. SDK podržava razvoj aplikacija za sve inačice Androida, što znači da svaki puta kada izade nova inačica Androida, SDK alati se trebaju nadograditi. Također sadrži različite alate pomoću kojih se mogu simulirati zasloni na dodir, razne postavke uređaja poput Bluetootha i GPS-a. Uobičajeno se sastoji od:

- Primjera kodova
- Emulatora
- Knjižica
- Sučelja aplikacijskog programa
- Programa za uklanjanje grešaka

Slika 4.3 prikazuje različite alate koji se mogu preuzeti putem Android SDK.

![Slika 4.3. Prikaz Android SDK Managera](image)

4.2.4. Android AVD

Prema [19], Android AVD (Virtual Device Manager) je integrirani alat koji služi za kreiranje virtualnoga mobilnog uređaja, odnosno emulatora. On omogućuje testiranje Android aplikacije bez potrebe za korištenjem fizičkog uređaja. Prije nego se krene koristiti emulator, potrebno je kreirati virtualni mobilni uređaj. Taj proces se obavlja preko AVD Managera, gdje postoji
mogućnost odabira inačice Androida, veličine zaslona, gustoće zaslona, količine memorije, SD kartice i sličnih opcija. AVD je veoma koristan jer omogućuje testiranje aplikacije na virtualnim uređajima različitih inačica, dimenzija i specifikacija, te na taj način pruža uvid kako se aplikacija ponaša na različitim uređajima radi bolje prilagodbe. Slika 4.4 prikazuje emulirani Google Nexus uređaj.

Slika 4.4. Emulirani Android uređaj

4.2.5. SQLite

Prema [20], SQLite je programski paket koji služi za upravljanje relacijskim bazama podataka. Relacijske baze podataka se koriste za pohranu podataka koji su definirani od strane korisnika u tablice. Osim pohrane i upravljanja podacima, poslužitelj baze podataka može obraditi složene naredbe upita koji kombiniraju podatke iz više tablica kako bi generirali izvješća i sažetke podataka. Neke od glavnih značajki SQLite-a su:

- Nije potreban odvojeni poslužitelj za pristup bazi podataka zato što se svi podaci spremaju direktno na disk
- Nema potrebu za instalacijom i konfiguracijom
- Cijela baza podataka se nalazi u jednoj datoteci što omogućava lako korištenje na različitim platformama
- Podržava pohranu velike količine podataka (baze podataka mogu biti veličine do nekoliko terabyte-a)
4.3. Ključni dijelovi koda

4.3.1. API razine


![Slika 4.5. Prikaz definiranja API razina](image)

4.3.2. Baza podataka

Kada se kreira baza podataka preko SQL-a i DBHelper-a, potrebno je nadglasiti (override) metode onCreate i onUpdate prikazane na slici 4.7. OnCreate se poziva prilikom prvog pokretanja aplikacije gdje se izvršavaju SQL izjave:

- CREATE_TABLE_USERS
- CREATE_TABLE_SETS
- CREATE_TABLE_CARDS
- CREATE_TABLE_USER_RESULTS

OnUpdate se poziva prilikom unošenja nekih promjena u strukturi bazi podataka, te se izvršavaju SQL izjave:

- DROP_TABLE_USERS
- DROP_TABLE_SETS
- DROP_TABLE_CARDS
- DROP_TABLE_USER_RESULTS

Zatim se ponovo poziva onCreate metoda kako bi se ponovo kreirale nove tablice s unesenim izmjenama. Svaka od tih SQL izjava se izvršava pomoću naredbe db.execSQL.
Slika 4.7. onCreate i onUpgrade metode

SQL izjava **CREATE_TABLE_USERS** kreira tablicu **TABLE_USERS** s pripadajućim atributima i njihovim tipovima podataka. **DROP_TABLE_USERS** služi za brisanje tablice **TABLE_USERS**. Izjava **SELECT_ALL_USERS** se koristi za odabir svih atributa iz tablice **TABLE_USERS**. Analogno ovim izjavama se izvršavaju i izjave za sve ostale tablice što je prikazano na slici 4.8.

Slika 4.8. SQL izjave

4.3.3. Registriranje i prijava korisnika

Registriranje korisnika odvija se na način da se upisana adresa e-pošte i zaporka spremaju u varijable **email** i **password** pomoću funkcije **getText().toString()**. Nakon toga se pomoću
funkcije `email.isEmpty() || password.isEmpty()` provjerava jesu li polja za upis prazna. Ako jesu, izbacit će se poruka „Potrebno je upisati podatke u oba polja“. Ako polja sadrže tekst vrši se usporedba unesene adrese e-pošte i zaporke s podacima o korisnicima iz baze pomoću funkcije `GetUser(email, password)`. Ta funkcija traži postoji li korisnik s takvom adresom e-pošte i zaporkom u bazi. Zatim se provjeravaju tri različita uvjeta. Funkcijom:

- `!email.contains("@gmail.com")` - provjerava se je li unesena adresa e-pošte sa @gmail domenom.
- `password.length() < 5` – provjerava se dužina zaporke koja treba imati najmanje 5 znakova
- `email.length() < 12` – provjerava se dužina adrese e-pošte koja treba imati najmanje 12 znakova

Nakon te provjere, u bazu se dodaje novi korisnik s upisnom adresom e-pošte i zaporkom pomoću funkcije `inserUser`, zatim se izbacuje obavijest da je korisnik registriran i preusmjerava se na početni zaslon aplikacije. Prikaz procesa registracije vidljiv je na slici 4.9.

```java
private void Register() {
    String email = etEmailReg.getText().toString();
    String password = etPassReg.getText().toString();
    if(email.isEmpty() || password.isEmpty()) {
        DisplayToast("Potrebno je upisati podatke u oba polja.");
    } else if(DBHelper.getInstance(this).getUser(email, password) != null) {
        DisplayToast("Korisnik vec registriran.");
    } else if(!email.contains("@gmail.com")) {
        DisplayToast("Potrebno se prijaviti s Google računom.");
    } else if(password.length() < 5) {
        DisplayToast("Zaporka mora imati 5 ili više znakova.");
    } else if(email.length() < 12) {
        DisplayToast("Potrebno uneti dužu adresu e-pošte. Barem 2 znaka prije @domene.");
    } else {
        DBHelper.getInstance(this).insertUser(new User(email, password, 0));
        DisplayToast("Korisnik registrovan.");
        startActivity(new Intent(RegisterActivity.this, LoginActivity.class));
        finish();
    }
}
```

Slika 4.9. Funkcija registriranja korisnika

Prijava korisnika odvija se tako da se unesena adresa e-pošte i zaporka uspoređuju s bazom registriranih korisnika. Ukoliko korisnik postoji, dohvaćaju se podaci o korisniku pomoću njegove adrese e-pošte te se postavlja da je neki korisnik prijavljen pomoću funkcije `session.SetLoggedIn(true)`. 
Zatim se pomoću funkcije `DBHelper.getInstance(this).getUserByEmail(email)` dohvaćaju svi podaci o korisniku koji se prijavljuje te se funkcijom `toBeLogged.setIsLogged(1)` postavlja da je on prijavljen. Potom se iz aktivnosti `LoginActivity` prelazi u aktivnost `MainMenuActivity`, što je zapravo glavni izbornik aplikacije i pomoću funkcije `putExtra` prosljeđuje mu se e-mail prijavljenog korisnika. Ako korisnik ne postoji javlja se greška u obliku obavijesti „Kriva adresa e-pošte ili lozinka“ što je vidljivo na slici 4.10.

```
private void login() {
    String email = etEmailLog.getText().toString();
    String password = etPasswordLog.getText().toString();

    if (DBHelper.getInstance(this).getUser(email, password)) {
        session.setLogin(true);
        User toBeLogged = DBHelper.getInstance(this).getUserByEmail(email);
        toBeLogged.setIsLogged(true);
        DBHelper.getInstance(this).updateUser(toBeLogged);
        Intent toMainActivity = new Intent(LoginActivity.this, MainMenuActivity.class);
        toMainActivity.putExtra("EMAIL", email);
        startActivity(toMainActivity);
        finish();
    } else {
        Toast.makeText("Krivi uokvir s-poste ili zaporka.");
    }
}
```

Slika 4.10. Funkcija prijave korisnika

4.3.4. Dodavanje skupova i kartica


```
btnAddSet.setOnClickListener(
    new View.OnClickListener() {
        public void onClick(View v) {
            AlertDialog.Builder alertDialog = new AlertDialog.Builder(ChooseSetActivity.this);
            alertDialog.setTitle("Novi set.
            alertDialog.setMessage("Izaberite skup.");

            final EditText inputSetName = new EditText(ChooseSetActivity.this);
            LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
                LinearLayout.LayoutParams.MATCH_PARENT,
                LinearLayout.LayoutParams.MATCH_PARENT);
            lp.gravity = Gravity.CENTER_HORIZONTAL;
            inputSetName.setLayoutParams(lp);
            alertDialog.setView(inputSetName);
        }
    });
```

Slika 4.11. Kreiranje dijaloga

Nakon što je dijalog kreiran i prikazan, pomoću funkcije `getAllSets()` dohvaćaju se imena svih skupova i sprema ih u listu (ArrayList). Nakon toga pomoću funkcije
equals(inputSetName.getText().toString()) provjerava postoji li u bazi skup s identičnim imenom. Ako postoji javlja grešku da skup s istim imenom već postoji, a ako ne postoji poziva se funkcija insertSet prikazana na slici 4.12, za dodavanje novog skupa u bazu podataka.

Slika 4.12. Dodavanje skupa

Na slici 4.13 prikazana je funkcija za dodavanje skupa u bazu podataka, pomoću funkcije put stavlja ime skupa u SET_NAME i na isti način stavlja user_id u USER_ID_FK. Zatim funkcijom writableDatabase.insert sprema te vrijednosti u bazu podataka.

Slika 4.13. Unos novog skupa

4.3.5. Brisanje kartica

Kako bi se obrisala kartica, potrebno je pritisnuti i držati na određenu karticu u popisu pitanja. Prvo se funkcijom `if(!rBtnGetCardsBySet.isChecked())` ispituje je li odabrana opcija pregleda kartica po skupovima. Ako nije, prikazuje se obavijest da je potrebno odabrati pregled po skupovima. Ako je, onda se dugim klikom pokreće funkcija `deleteCard` za brisanje kartice te obavijest da je kartica obrisana. Nakon što je kartica obrisana, pomoću funkcije `populateList`, ponovo se prikazuju sve kartice u skupu bez kartice koja je obrisana. Ova funkcionalnost prikazana je na slici 4.15.
Slika 4.15. Brisanje kartice

4.3.6. Prikaz rezultata
Za prikaz rezultata prvo se poziva funkcija `getData` sa slike 4.16 koja dohvaća rezultate svakog skupa trenutno prijavljenog korisnika pomoću funkcije `getUserResultByUserId(logged.getId())` te nakon što se dobiju svi rezultati oni se prikazuju u listi. U slučaju da nema nijednog riješenog kviza, pomoću funkcije `View.VISIBLE` pojavljuje se poruka zaslonu da zasad nema riješenih kvizova.

Slika 4.16. Dohvaćanje rezultata

4.3.7. Kod za kviz
Za pokretanje kviza potrebno je stisnuti i držati ime skupa na listi koji želimo pokrenuti. Nakon što se kviz pokrene prvo se vrši provjera ima li u pokrenutom skupu kartica pomoću funkcije `isEmpty`. Ako ih nema izbacuje se obavijest da je nemoguće pokrenuti kviz i korisnik se pomoću tipke vraća na `MainMenuActivity` aktivnost. Funkcija `setVisibility` sa slike 4.17 prikazuje ili skriva određene elemente sučelja.
Ako postoji, onda se prvo poziva **nextQuestion** funkcija prikazana na slici 4.18 koja provjerava na kojem pitanju se korisnik nalazi. Ukoliko se nalazi na zadnjem pitanju, funkcija zaustavlja kviz, dohvaća i sprema broj kartica u skupu pomoću funkcije **int nbrOfAnswers = this.cardsForQuiz.size()** i izračunava postotak točnosti odgovora funkcijom **float r = (float) corectAnswers/nbrOfAnswers** te prikazuje dijalog s rezultatima kviza, a ako se ne nalazi na zadnjem pitanju poziva se funkcija **currentCard.getQuestion** koja prikazuje tekst sljedećeg pitanja.

Nakon izvršavanja funkcije **nextQuestion**, pritiskom na tipku „**Okreni**“ stvara se dijalog koji omogućuje korisniku pregled ili unos odgovora. Ako se pritisne pozitivna tipka „**Provjeri**“, poziva se funkcija **getAnswer** i prikazuje se tekst odgovora. Ukoliko je pritisnuta negativna tipka „**Unesi**“, uklanja se tekst pitanja funkcijom **tvQuiz.setVisibility(View.GONE)** i prikazuje se polje za unos odgovora funkcijom **etAnswer.setVisibility(View.VISIBLE)** što je vidljivo na slici 4.19.
Pritiskom na tipku „Potvrdi“, vrši se provjera je li tekst odgovora unesenog u polje za unos teksta odgovara odgovoru trenutne kartice. Na slici 4.20 viđi se da se prvo dohvaća tekst funkcijom `etAnswer.getText()` iz polja za unos teksta, te se uspoređuje s odgovorom trenutne kartice pomoću funkcije `currentCard.getAnswer().equals(answer)`. Ako odgovara, povećava se brojač točnih odgovora, postavlja se vidljivost elemenata za prikaz idućeg pitanja te se prikazuje obavijest da je kartica točno odgovorena.

Ukoliko tekst unesenog odgovora nije jednak tekstu odgovora trenutne kartice povećava se brojač netočnih odgovora funkcijom `currentCard.setAnswered_true_false(2)`. Broj 2 u toj funkciji označava da je kartica netočno odgovorena kao što broj 1 označava da je kartica točno odgovorena. Zatim se funkcijom `llButtonShowCorrectAnswer.setVisibility(View.VISIBLE)` prikazuje tipka za prikaz točnog odgovora što je vidljivo na slici 4.21. Pritiskom na tu tipku, kreira se i prikazuje dijalog sa točnim odgovorom.
Slika 4.21. Provjera netočnog odgovora

Pitiskom na tipku „Sljedeća“ u aplikaciji proces se ponavlja sve dok brojač kartica ne postane jednak ukupnom broju kartica u skupu i tada se skup završava. Ukoliko korisnik tijekom rješavanja kviza, pritisne sistemsku tipku nazad pozvati će se funkcija onBackPressed prikazana na slici 4.22.

```java
public void onBackPressed() {
    super.onBackPressed();
    this.clearData();
    Toast.makeText(this, "Kviz prekinut. Podaci obrisani.", Toast.LENGTH_LONG).show();
    finish();
    new Intent(getApplicationContext(), QuizActivity.class);
}
```

Slika 4.22. Funkcija onBackPressed

Potom se poziva funkcija clearData prikazana na slici 4.23 koja briše sve podatke o odgovorenim karticama u kvizu funkcijom item.setAnswered_true_false(0), kviz se zaustavlja i korisnik se vraća na glavni izbornik aplikacije.

```java
private void clearData() {
    for (Card item:
        this.cardsForQuiz) {
        item.setAnswered_true_false(0);
        DBHelper.getInstance(this).updateCardIsAnswerTrueFalse(item);
    }
}
```

Slika 4.23. Funkcija clearData
5. KORIŠTENJE I TESTIRANJE MOBILNE APLIKACIJE

5.1. Prijava i registracija korisnika

Kako bi mogli koristiti aplikaciju Pametne kartice, potrebno je na pametni telefon instalirati datoteku .apk aplikacije ili pokrenuti aplikaciju preko Android Studia. Nakon instalacije korisnik može pristupiti aplikaciji tako da u glavnom izborniku pametnog telefona pritisne na pripadajuću kućicu (Sl. 5.1).

Slika 5.1. Pokretanje aplikacije

Nakon pokretanja aplikacije, pojavljuje se početni zaslon aplikacije. Na početnom zaslonu korisnik ima mogućnost prijave i registracije. Ako je korisnik već registriran, može se samo prijaviti i krenuti s korištenjem aplikacije (Sl. 5.2).

Slika 5.2. Početni zaslon aplikacije
Ukoliko korisnik prvi put koristi aplikaciju, obvezan je proći kroz proces registriranja. Pri registriranju obavezan je unos Google adrese e-pošte koja mora imati barem dva znaka prije @domene, i lozinke koja mora sadržavati najmanje pet znakova (Sl. 5.3).

5.2. Glavni izbornik

Nakon prijave, korisnik se automatski prosljeđuje na glavni izbornik aplikacije. Tu mu se pruža uvid u glavne funkcionalnosti aplikacije te prikaz korisničkog računa s kojim je prijavljen. Sve radnje koje korisnik može izvesti kreću iz i završavaju u glavnom izborniku (Sl. 5.4).

5.3. Skupovi i kartice

Kako bi korisnik mogao pokrenuti kviz, prvo je potrebno kreirati skup i u njega dodati kartice s pitanjima. To može napraviti pritiskom na opciju Odaberi skup, gdje se zatim može odabrati opcija Dodaj skup i tada korisnik unosi ime željenog skupa. Ime skupa mora biti unikatno. (Sl. 5.5).
Kada je skup kreiran, korisnik u tablici može vidjeti sve već postojeće skupove. Brisanje određenog skupa se vrši pritiskom na ime skupa nakon čega se pojavljuje opcija za brisanje skupa (Sl. 5.6).

Za dodavanje kartica, korisnik treba pritisnuti na opciju **Dodaj karticu**. Tada je prvo potrebno odabrati u koji već kreirani skup korisnik želi dodati novu karticu. Nakon toga, jednostavno se unosi željeno pitanje i odgovor te pritiskom na opciju **Dodaj** i pozitivne potvrde kartica se dodaje u skup. Ako korisnik pokuša dva puta dodati istu karticu pojavit će se obavijest da je takva kartica već dodana.
5.4. Pregled pitanja

Nakon što korisnik kreira sve željene skupove i kartice, pruža mu se i mogućnost uvida u iste. Odabiram opciju **Pregledaj kartice** u glavnom izborniku, dobiva se izlistanje svih kartica. Kartice se mogu sortirati na dva načina. Prvi način prikaza je odabirom na prikaz **Sve** gdje se prikazuju sve kreirane kartice (Sl. 5.8).

![Slika 5.8. Prikaz svih kartica](image)

Drugi način prikaza je odabirom prikaza **Po skupovima**, gdje je korisnik potreban prvo odabrati iz kojeg skupa želi prikazati kartice. Nakon tog odabira, prikazuju mu se samo pitanja iz odabranog skupa (Sl. 5.9).
Prilikom pregleda kartica, korisnik ima mogućnost detaljnijeg uvida u karticu, tako da mu se prikažu odabranо pitanje i odgovor jedno ispod drugog. To može napraviti pritiskom na bilo koju od kartica neovisno o načinu pregleda (Sl. 5.10).

5.5. Brisanje kartica

Kako bi korisnik obrisao neku već kreiranu karticu, potrebno je iz glavnog izbornika odabrati opciju Pregledaj kartice. Nakon toga mora odabrati prikaz kartica Po skupovima, nakon toga dugim pritiskom na željeno pitanje i odgovor vrši se brisanje odabranog i korisnik dobiva obavijest da je odabrana kartica uspješno obrisana (Sl. 5.11).
5.6. Kviz

Za pokretanje kviza za provjeru znanja korisnik mora odabrati opciju **Odaberi skup** iz glavnog izbornika. Zatim se dugim klikom na željeni skup na zaslonu prikazuje poruka da li želite pokrenuti kviz na koju korisnik može odgovoriti s **Da** ili **Ne**. U slučaju da odabrani skup ne sadrži niti jednu karticu, kviz se neće moći pokrenuti. Ako kartice postoje, kviz će se pokrenuti (Sl. 5.12).

Nakon pokretanja kviza, korisniku se na zaslonu prikazuje prvo pitanje, dvije korisničke opcije i brojilo koji pokazuje na kojem je korisnik pitanju trenutno. Ako korisnik odabere **Sljedeća** na zaslonu će se pojaviti obavijest da je prvo potrebno odgovoriti na trenutnu karticu kako bi se moglo prijeći na sljedeću. U slučaju odabira **Okreni**, pojavljuje se upit želi li korisnik provjeriti odgovor ili unijeti svoj (Sl. 5.13). Ako se odabere opcija **Provjeri**, na zaslonu će se prikazati odgovor za pripadajuće pitanje i dvije nove opcije. Korisnik pomoću njih označava je li znao...
odgovor na pitanje ili ne. Nakon što to označi, pritiskom na tipku Sljedeća može prijeći na sljedeće pitanje (Sl. 5.13).

Slika 5.13. Provjera odgovora

Ukoliko korisnik odabere opciju Unesi, na zaslonu uredaja će mu se pojaviti polje za unos odgovora. Važno je znati da je unos odgovora osjetljiv na unos velikih slova, malih slova, zareza i točaka. Nakon što se unese odgovor, korisnik treba pritisnuti Potvrdi. Ako je odgovor točan, izbacuje se obavijest da je kartica odgovorena i odgovor je točan. U slučaju da je uneseni odgovor netočan, izbacuje se obavijest da je kartica odgovorena, ali odgovor nije točan i pojavljuje se opcija prikaza odgovora. Nakon što je odabrana ta opcija, na zaslonu se prikazuje točan odgovor tog pitanja i nakon toga se može prijeći na sljedeću karticu (Sl. 5.14).

Slika 5.14. Unos odgovora

5.7. Prikaz rezultata

Nakon završenog kviza, korisniku se prikazuje obavijest o uspješnosti rješavanja kviza u brojčanom i postotnom obliku (Sl. 5.15.). Nakon toga, kviz završava i korisnik je preusmjeren.
nazad na glavni izbornik. Iz glavnog izbornika moguće je pristupiti i opciji **Pregledaj rezultate**. Tu se nalazi prikaz prijašnje riješenih skupova i postotak riješenosti. Izlistanje je takvo da se najnoviji riješeni kviz uvijek postavlja na prvo mjesto (Sl. 5.15).

![Slika 5.15. Prikaz rezultata](image)

Kako bi osigurali da aplikacija radi bez grešaka testiranje aplikacije je provedeno na više različitih korisničkih računa. Na svakom računu testirani su različiti skupovi i pitanja. Sve korisničke opcije su testirane da ni u jednom trenutku ne bi došlo do rušenja aplikacije.

### 5.8. Testiranje aplikacije

Za potrebe testiranja aplikacije kreiran je skup pitanja pod nazivom „Osnove automatskog upravljanja“ što je vidljivo na slici 5.16.

![Slika 5.16. Pregled kreiranih skupova](image)

Skup je napunjen karticama s pitanjima i odgovorima kako bi se moglo provesti testiranje rada kviza. Na slici 5.17 se mogu vidjeti dodavanje i pregled nekih od kreiranih pitanja.
Kako bi se testirala ispravnost kviza, odabrao se kreirani skup i pokrenuli kviz. Prilikom prolaska kroz pitanja testirane su opcije provjere pitanja, prikazano na slici 5.18.

Na slici 5.19 vidi se da je prvo testirana opcija provjere pitanja, koja je uspješno prikazala odgovor postavljenog pitanja i opcije za označavanje odgovora.

Slika 5.18. Pokretanje kviza i prikaz pitanja

Slika 5.19. Prikaz odgovora opcijom „Provjeri“
Na sljedećem pitanju testirana je opcija korisničkog unosa odgovora. Provjerena je ispravnost unosa točnog i netočnog odgovora. Oba dva slučaja su uzrokovala predviđene radnje. Na slici 5.20 prikazan je unos netočnog odgovora, gdje se nakon unosa netočnog odgovora prikazuje opcija za prikaz točnog odgovora.

Slika 5.20. Unos netočnog odgovora

Pri unosu točnog odgovora aplikacija je prikazala opcije za označavanje i prelazak na sljedeće pitanje kao što je zahtijevano u tom slučaju što je prikazano na slici 5.21. Nakon što se provjeri i označe ostale kartice, prikazuje se postotak riješenosti kao što je opisano i prikazano u programskom rješenju.

Slika 5.21. Unos točnog odgovora

Prikazani oblik učenja s karticama dokazano šteti vrijeme jer osigurava da se korisnik koncentrira samo na ono što je važno, i time olakšava sam proces učenja. Primjena takvog načina učenja na pametni telefon uvelike proširuje mogućnosti, jer korisnik uz sebe ne mora imati nikakve papirnate materijale da bi učio i može učiti u bilo koje vrijeme i na bilo kojem mjestu.
6. ZAKLJUČAK

U ovom završnom radu ostvarena je mobilna aplikacija za testiranje i provjeru znanja koja funkcionira na temelju samostalnog unošenja materijala za učenje. Korisničke opcije ostvarene u aplikaciji omogućuju dodavanje i brisanje skupova i kartica, praćenje rezultata i testiranje. Tijekom izrade usvojena su znanja potrebna za razvoj Android aplikacija uz pomoć odgovarajućih programskih alata. Programska dio ovog rada izrađen je u Java programskom jeziku na temelju korištenja različitih aktivnosti (Activity) kako bi se ostvarile potrebne funkcionalnosti aplikacije. Izgled aplikacije je izrađen u jeziku za označavanje (XML) u samom Android Studiu. Sve što je prezentirano u idejnom rješenju je uspješno ostvareno i testirano na dovoljnom broju primjera. Spremanje podataka je obavljeno pomoću lokalne SQLite baze podataka gdje se spremaju svi korisnički podaci. Obrađene su značajke korištenja ovakvih sustava učenja i njihovih prednosti u odnosu na konvencionalne načine. Razvoj aplikacije je osim prikazanog rješenja, potaknuo i razmišljanje kako bi se ovakav tip aplikacije mogao unaprijediti i poboljšati. Mogućnost dodavanja slika i vremenski ograničeni kviz su samo od nekih načina kako bi se proces učenja mogao učiniti boljim. Prednost ovakvog načina učenja je mogućnost učenja bilo gdje i bilo kada bez puno pripreme, te je na taj način vrijeme trajanja učenja učinkovito iskorišteno. Istraživanje sličnih postojećih sustava dovelo je do zaključka da uz sadašnji trend razvoja i dostupnosti mobilnih uređaja, ovakvi načini učenja će se sve više razvijati.
LITERATURA


[2] Učenje
http://os-jakovle.skole.hr/naslovnica/uspjesiucenika/jo_malo_o_u_enju?only_mod_instanc
e=52_1304_0&st3_action=move_doc&st3_id [posjećeno 10.06.2017.]

[3] Što je učenje


http://www.molenet.org.uk/ [posjećeno 12.06.2017.]

https://bib.irb.hr/datoteka/695645.ZR_Knezevic_Mlearning.pdf [posjećeno 12.06.2017.]


[8] Best flashcard apps

[9] Flashcard apps for Android
http://www.makeuseof.com/tag/6-flash-card-apps-for-android-compared-which-is-the-best/
[posjećeno 14.06.2017.]

[10] Flash Card apps for students


[12] Android OS

[13] Sve o Android operacijskom sustavu
[14] Značajke Android operacijskog sustava
   https://hr.wikipedia.org/wiki/Android_(operacijski_sustav)#Zna%C4.8Dajke
   [posjećeno 20.06.2017.]

   http://blog.dnevnik.hr/print/id/1631938909/arhitektura-androida.html
   [posjećeno 20.06.2017.]

[16] Android studio

[17] Java programski jezik
   http://pcchip.hr/softver/korisni/java-sto-kako-i-zasto/ [posjećeno 23.06.2017.]

[18] Android SDK
   https://en.wikipedia.org/wiki/Android_software_development#Android_SDK
   [posjećeno 25.06.2017.]

[19] Android AVD

[20] SQLite
   https://www.safaribooksonline.com/library/view/using-sqlite/9781449394592/ch01.html
   [posjećeno 27.06.2017.]
SAŽETAK


Ključne riječi: Android, Java, kviz, mobilno učenje, prilagodljiva aplikacija.

MOBILE ANDROID APPLICATION FOR ADAPTIVE LEARNING

ABSTRACT

The main goal of this paper was to develop a Android mobile application that allows the user options for adaptive learning in such a way that he can set up the materials he considers essential. The advantages of this application is the adaptability of places and learning times. It was necessary to develop an application which allows creation of sets, adding of question and answer cards, a quiz to check the knowledge and a way to track the results. The application is written in the Java programming language with the help of Android Studio and other necessary software tools. The database is written using the SQLite software package. Testing of the application shows that user interface is simple and intuitive to use. All application capabilities were tested on a sufficient number of instances to ensure flawless operation.

Keywords: Android, Java, quiz, customizable application, learning
ŽIVOTOPIS

PRILOZI (na CD-u)

Prilog 1. Datoteke završnog rada (docx i pdf)

Prilog 2. Programski kod Android mobilne aplikacije