UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Zaključna naloga

Razvoj sistema za učenje s karticami na mobilnih napravah

(Development of a flash card learning system for mobile devices)

Ime in priimek: Alen Andrašić
Študijski program: Računalništvo in informatika
Mentor: doc. dr. Peter Rogelj

Koper, avgust 2014

Ključna dokumentacijska informacija

Ime in PRIIMEK: Alen ANDRAŠIĆ

Naslov zaključne naloge: Razvoj sistema za učenje s karticami na mobilnih napravah

Kraj: Koper

Leto: 2014

Število listov: 48 Število slik: 36

Število prilog: 2 Število strani prilog: 1 Število referenc: 7

Mentor: doc. dr. Peter Rogelj

Ključne besede:

Programsko inženirstvo, aplikacija, učenje s ponavljanjem, Java, Android

Izvleček:

V zaključni nalogi so predstavljene glavne faze programskega inženirstva preko kreacije aplikacije za operacijski sistem Android. Cilj je prikazati vse korake — od definicije problema, študije izvedljivosti pa do analize zahtev, načrtovanja in same izvedbe ter testiranja aplikacije. Končni izdelek bo pripravljen za uporabo v obliki aplikacije Android in strežniškega modula. Namen same aplikacije je uporabnikom nuditi nov način učenja in obnavljanja znanja preko učenja s karticami in ponavljanjem. Za dosego cilja smo uporabili le brezplačno programsko opremo ter programska jezika Javo in PHP.

Key words documentation

Name and SURNAME: Alen ANDRAŠIĆ

Title of final project paper: Development of a flash card learning system for mobile devices

Place: Koper

Year: 2014

Number of pages: 48 Number of figures: 36

Number of appendices: 2 Number of appendix pages: 1 Number of references: 7

Mentor: Assist. Prof. Peter Rogelj, PhD

Keywords:

Software engineering, application, learning with repetition, Java, Android

Abstract:

This thesis presents the main phases of software ingeneering thgouh creation of an Android application. The maing goals is to include all steps - from problem definition, feasibility study, definition of requrements, system planning, implementation and testing. At the end, we will have a ready to use final product, composed with an Android application and a server side script. The application will provide its users with a new and easy way to learn and maintain their knowledge by studying using flashcard system and repetition. In order to achieve our goal we used only free software. The application is wrote in Java programming language, while the server side script is done using PHP.

Zahvala

Najprej bi se rad zahvalil mentorju, doc. dr. Petru Roglju, ki me je s koristnimi napotki usmerjal na poti do cilja. Vedno dosegljiv in z dobrimi predlogi je olajšal proces pisanja naloge in ustvarjanja aplikacije.

Prav tako bi se rad zahvalil družini in bližnjim ter seveda boljši polovici za spodbudo od prvega do zadnjega študijskega dne.

Hvala

Kazalo vsebine

Kazalo slik

1 Uvod

Namen zaključne naloge je predstavitev procesov in faz načrtovanja, vse od analize zahtev pa do realizacije produkta, ki bo lahko marsikomu uporaben. Pri tem se bomo osredotočili na mobilno aplikacijo za operacijski sistem Android, zaradi katerega se bomo srečali s programskim jezikom Javo. Poleg tega pa bomo pripravili tudi manjši strežniški del, in sicer v PHP-jeziku. Seveda bomo pri tem spoznali nekaj orodij, ki nam delo olajšujejo.

Za operacijski sistem Android smo se odločili zaradi njegove odprtosti, saj nam omogoča, da lahko izdelek ustvarimo in poganjamo na napravah povsem brezplačno.

Zaradi razvoja tehnologije in tendence ljudi, da čim več stvari počnejo na poti, smo se odločili, da bomo ustvarili aplikacijo, ki bi lahko omogočila učenje oziroma obnavljanje znanja na hiter in enostaven način. Uporabnikom bo omogočala pomoč pri učenju s kartičnim sistemom *(flashcard learning)* [1], ki ga poznamo že od leta 1834 [2]. Glede na odprtost sistema smo se pa odločili, da bomo tudi aplikacijo ustvarili na kar se da odprt način in s tem omogočili enostavno ustvarjanje učnih paketov in njihovo deljenje med uporabniki.

Najprej bomo predstavili definicijo problema, nato študijo izvedljivosti. Pripravili smo še analizo zahtev, med katere spadajo tako funkcijske kot nefunkcijske zahteve, ki ji bosta sledila še načrtovanje sistema in opis poteka testiranja. Seveda ne smemo pozabiti na to, da se razvoj tu ne konča, saj je sistem treba vzdrževati in dopolnjevati z novimi funkcionalnostmi ter slediti trendom v tem hitro spreminjajočem se svetu. Idej za nadaljnji razvoj ne manjka, zato bomo tudi temu posvetili nekaj besed.

2 Definicija problema

V trenutnem tehnološko zelo dobro razvitem obdobju nas na vsakem koraku spremljajo pametne naprave, zato bi nam koristilo, da bi nam poleg povečanja naše produktivnosti omogočale še lažje učenje oziroma drugačen pristop k učenju.

Težava, ki pri tem nastane, je, da si nismo vsi med seboj enaki in so naše zahteve, potrebe in načini učenja različni. To seveda pomeni, da je nemogoče pripraviti pripomoček, ki bi ustrezal vsakemu izmed nas. Vendar se lahko potrudimo narediti orodje, ki bo nekaterim ljudem le ustrezalo, in ga razviti na ta način, da ga bodo ljudje tudi uporabljali.

V osnovi gre za učenje s karticami (t. i. *flash card*), ki deluje na principu učenja s ponavljanjem (*spaced repetition*) [3]. Gre za kup kart, na katerih so zapisana vprašanja in odgovori (na eni strani karte je vprašanje, na drugi odgovor). Karte so v kupu obrnjene tako, da nam je vedno vidno le vprašanje. To preberemo in si poskušamo v sebi odgovoriti, nato pa preverimo, ali se naš odgovor ujema z zapisanim. Glede na to kako smo na vprašanje odgovorili, se ta karta postavi na nov kup. Tako karte sortiramo na n različnih kupov. En kup predstavlja lahka vprašanja, na katera smo odgovorili hitro in pravilo, drugi kup pa tista, na katera smo odgovorili pravilno, vendar pomanjkljivo (ali nismo bili prepričani, ali je odgovor pravilen), tretji pa vprašanja, na katera odgovora nismo poznali. Ko se bomo naslednjič učili, bomo najprej obdelali kup z vprašanji, na katerega nismo poznali odgovora. Manj pozornosti bomo posvetili kupu s pomanjkljivimi odgovori, najmanj pa tistemu, na katerem se nahajajo karte z vprašanji, na katera smo z lahkoto odgovorili. Cilj je, da znamo čez čas na vse karte odgovoriti pravilno in bodo posledično vse na kupu s pravilnimi odgovori, kar bo pomenilo, da odgovore poznamo. Vprašanja iz tega kupa bomo potem občasno obnavljali in svoje znanje s tem ohranjali. Če pa se pri kakem vprašanju zmedemo, ga lahko znova premaknemo na katerega izmed preostalih kupov.

Problem pa nastane, ko je treba ta princip prenesti na mobilno aplikacijo in jo narediti uporabniku prijazno. Težave se pojavijo pri frekvenci ponavljanja vprašanj glede na uporabnikovo znanje. Najprej je treba pripraviti pakete z vprašanji, ki bodo uporabniku na voljo. Omogočiti je seveda treba, da so paketi na voljo več uporabnikom, ne pa nujno tudi vsem. Hkrati je treba pripraviti orodje, s katerim se bo pakete na enostaven način kreiralo, uporabniki pa bodo morali te pakete brez večjih težav prenesti

na svojo napravo.

Zavedati se pa je treba, da je uporabnike težko prisiliti v uporabo take aplikacije. Ciljna publika so zato uporabniki, ki bi želeli svoje znanje razširiti oziroma ohranjati, pa naj bo to za osebne namene (učenje tujega jezika, splošno znanje glede določene tematike ...) ali pa je cilj zgolj naučiti se snov za šolski predmet, kjer bi lahko uporabniki/učitelji kreirali paket, ki bi ga lahko uporabljali ostali in generacije za njimi.

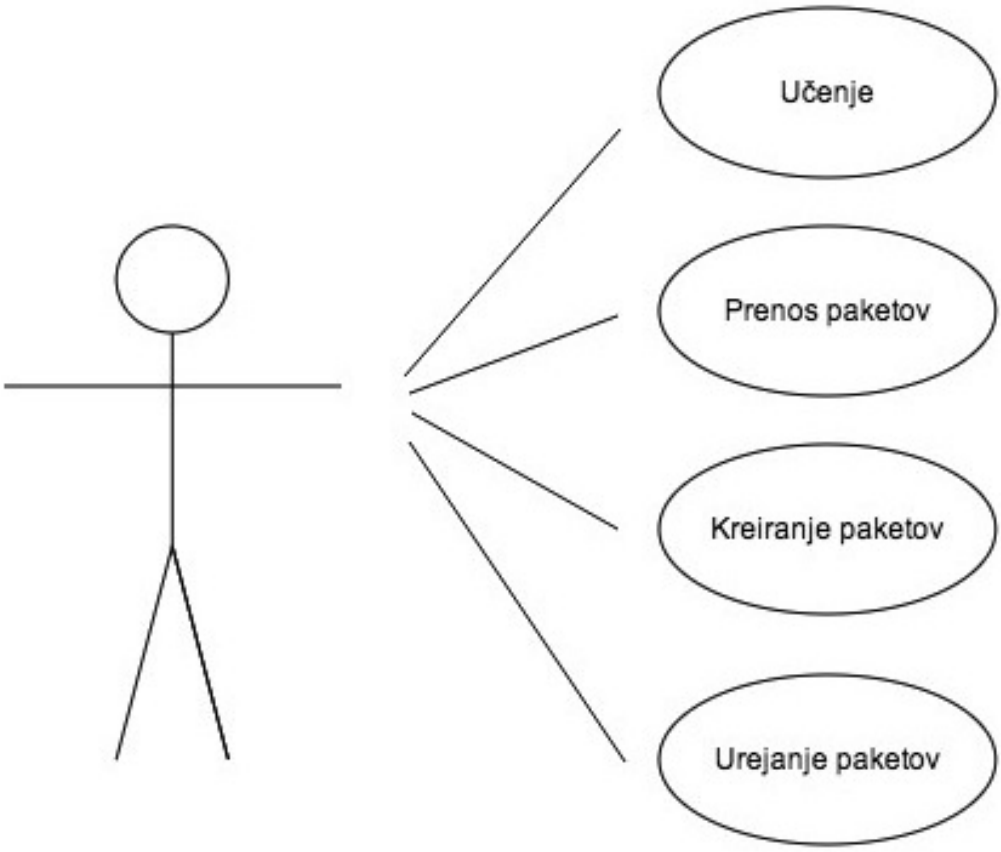
3 Študija izvedljivosti

Največja težava, s katero bi se lahko soočil naš izdelek, je nasičenost trga mobilnih aplikacij. V avgustu je bilo na Google Play Storu naloženo približno 1.329.594 aplikacij [4] in izstopati pri takem številu ni enostavno. Da bi aplikacija postala uspešna, bi bilo treba vložiti nekaj časa in truda v oglaševanje preko socialnih omrežij, s čimer bi privarčevali finančna sredstva, ki bi jih sicer morali vložiti v trženje. Prav tako bi bilo treba projektu (in ne samo aplikaciji) po končani načrtovani fazi ponuditi več časa in ga razširiti tako na spletu kot tudi na drugih platformah.

Kar se finančnega stanja projekta tiče, nas ta (razen seveda časa), ne bo stal ničesar, saj bomo uporabljali brezplačno programsko opremo, za gostovanje pa bomo uporabili že obstoječi strežnik. Omeniti je treba, da je lahko aplikacija uporabna in delujoča tudi brez strežnika, vendar pa ima ta kljub temu izredno velik pomen. Glede na omenjeno, je izvedba projekta smiselna, saj bo iz zaključne naloge in brez dodatnih stroškov nastala aplikacija, ki zna marsikomu — tudi meni samemu — olajšati učenje in obnavljanje znanja.

4 Specifikacija zahtev

V poglavju specifikacije zahtev si bomo ogledali funkcijske in nefunkcijske zahteve za strežniški in Android modul. V funkcijskih si bomo ogledali vse komponente modulov, medtem ko se bomo v nefunkcijskih posvetili ostalim zadevam (varnost, odzivnost ...).



Slika 1: UML-diagram primera uporabe

4.1 Funkcijske zahteve

Uporabniku je že ob samem zagonu aplikacije iz samega uporabniškega vmesnika razvidno, kaj bo počel s klikom na enega izmed podanih gumbov. Kot je prikazano na diagramu primera uporabe, je to lahko učenje, prenos paketov, kreacija paketa, pre-

gled navodil za kreiranje paketov oz. implementacijo modula za kreiranje paketov na strežniku ter nastavitve za učenje.

4.1.1 Učenje

Uporabniku bodo ob vstopu na zaslon za učenje, če jih ima že naložene na telefon, prikazani paketi (v nasprotnem primeru se mu pokaže ustrezno obvestilo, da mora pakete najprej prenesti ali ustvariti). Po izbiri želenega paketa se mu morata prikazati vprašanje in gumb, ki mu poleg vprašanja prikaže še odgovor. Po kliku nanj pa se mora ta skriti in prikazati se mu morajo gumbi (»Lahko«, »Srednje«, »Težko«), ki ponazarjajo njegovo znanje in težavnost pri odgovarjanju na vprašanje. Postopek mora uporabnik ponavljati, dokler so vprašanja v paketu na voljo ali doseže cilj, ki si ga je sam zastavil (glej 4.1.6 Uporabniške nastavitve). Ponuditi se mu mora možnost za nadaljnje učenje ali pa povratek na prejšnji zaslon.

4.1.2 Prenos paketov

Možnost prenosa paketov mora uporabniku na enostaven način omogočiti pregled paketov, ki se nahajajo na strežniku. Poleg tega pa se mora s klikom na paket začeti prenos paketa. Uporabnik mora biti s sporočilom ustrezno obveščen o začetku prenosa, prenosu samem in o uspelem ali neuspelem prenosu paketa. Uporabnik lahko nato po želji nadaljuje s prenašanjem drugih paketov ali se vrne na prejšnji zaslon, od koder lahko začne postopek učenja (4.1.1 Učenje) ali pa prenesene pakete ureja (4.1.4 Urejanje paketov).

4.1.3 Kreacija paketa

Uporabniku mora biti omogočeno, da lahko paket sestavi sam. Ker modul za kreiranje paketov lahko obstaja že na nastavljenem strežniku (4.1.5 Navodila za kreiranje in implementacijo ter uporabo modula), mu moramo omogočiti, da to opravi preko brskalnika na telefonu. Pri tem se mu mora modul prikazati pravilno, kar bo predstavljalo polja za vnos naslova paketa, vprašanja, odgovora, URL-naslova ter gumb, ki bo paket na strežniku sestavil. Prav tako mora biti uporabniku omogočeno kreiranje znotraj same aplikacije, brez uporabe brskalnika. Pri tem mu moramo najprej predstaviti dialog, ki bo zahteval naslov in opis paketa. Po uspešni kreaciji praznega paketa ga bo lahko uporabnik urejal kot druge že prenesene pakete (4.1.4 Urejanje paketov).

4.1.4 Urejanje paketov

Poleg kreacije paketov mora biti uporabniku dovoljeno paket urejati. Na zaslonu za urejanje paketa mora imeti uporabnik na voljo pregled nad vsemi vprašanji, ki se v paketu nahajajo, in možnost spreminjanja vprašanja s klikom nanj. Pri tem se mu mora prikazati dialog, v katerem lahko ureja posamezno vprašanje. Prav tako mora imeti možnost dodajanja novih vprašanj, za kar mu mora biti na voljo gumb, ki se mora prav tako nahajati na seznamu vprašanj paketa.

4.1.5 Navodila za kreiranje in implementacijo ter uporabo modula

Navodila uporabniku predstavijo sam potek kreacije ter opišejo shranjevanje in lokacijo kreiranega paketa. Uporabniku mora biti predstavljen modul, ki si ga lahko po želji namesti na svoj strežnik, na katerega lahko nato preusmeri aplikacijo. Prikazana morajo biti navodila postavitve na strežnik in navodila za preusmeritev aplikacije nanj (4.1.6 Uporabniške nastavitve za učenje).

4.1.6 Uporabniške nastavitve za učenje

Uporabnik si bo lahko v nastavitvah učenje prilagodil svojim željam in zmožnostim, ki mu jih dopušča čas. Tako si bo lahko nastavil frekvenco ponavljanja vprašanj (kot ponavljanje se razume čas, kdaj bo vprašanje uporabniku znova na voljo v izbranem paketu), vendar le z vrednostmi »redko«, »srednje« in »pogosto«. Tako uporabniku čas, kdaj se bo vprašanje ponovilo (saj je to določeno dinamično), ne bo znan, vendar bo čez čas opazil, ali je to zanj prepogosto ali preredko.

Pomembo je tudi število vprašanj, ki si jih bo uporabnik nastavil za posamezno seanso. Aplikacija bo namreč seanso zaključila in posodobila uporabnikovo statistiko le, če ta učenje zaključi. Prav tako si bo lahko tu nastavil, ali želi, da ga aplikacija opozori, kdaj se bo v paketu pojavilo kako vprašanje na voljo za ponovno učenje. Kot že rečeno, bo tu tudi naslov strežnika, na katerem se bodo nahajali paketi.

4.2 Nefunkcijske zahteve

Uporabnik mora imeti, da lahko uporablja Androidovo komponento, telefon, na katerem teče operacijski sistem Android verzije 4.0 ali višje. Če bi pa nastavil strežniški modul na svojem strežniku, mora ta podpirati PHP.

4.2.1 Strežniški modul — varnost

Strežniški modul je varovan z geslom, ki ga lahko uporabnik v nastavitvah modula oziroma njegovi kodi spremeni. Sprememba je močno priporočljiva, saj bo geslo vedel vsak, ki bo kodo modula videl, ni pa obvezna. S tem se strežnik zavaruje na ta način, da lahko pakete dodajajo le tisti, ki geslo poznajo. Odločili bi se lahko tudi, da geslo odstranimo, vendar tudi to ni priporočljivo.

Na sam dostop do paketov na strežniku z aplikacijo in njegovo uporabo geslo ne vpliva. Poznati moramo le pravo pot do lokacije paketov.

4.2.2 Aplikacija Android

Aplikacija je hitra in performančno nezahtevna, zato deluje tudi na najslabši Android 4.0 napravi. Odločili smo se za spodnjo mejo sistema 4.0 zaradi API-ja in metod, ki nam jih ta ponuja (trenutno je v uporabi 85 % naprav s sistemom 4.0 in novejšim) [5]. Prav tako pa ni pomembna niti velikost zaslona, saj se aplikacija lepo prilagaja vsem velikostim (tudi tabličnemu računalniku).

4.2.2.1 Urejanje paketov

Paketi so pri samem urejanju medsebojno neodvisni. Z urejanjem enega ne posegamo v druge. Prav tako pri urejanju vprašanj spreminjamo samo eno vprašanje, na ostala vprašanja znotraj paketa pa to ne vpliva. Paketi so shranjeni kot JSON-datoteke na uporabnikovem telefonu in ne v spominu aplikacije. Tako omogočimo, da ob brisanju aplikacije (s tem se ohrani tudi uporabnikov napredek) paketi ostanejo nedotaknjeni. To omogoča tudi prenos na druge naprave.

4.2.2.2 Učenje

Tako kot pri urejanju se pri učenju spreminjajo samo vprašanja, ki so v trenutni učni seji. Ostala vprašanja ločimo in jih na ta način pustimo nedotaknjena. Če bi predčasno prekinili učno sejo z izhodom iz aplikacije, se spremembe prav tako ne shranijo, s čimer preprečimo morebitno poškodbo paketa.

4.2.3 Sestava paketa

Pri sestavi paketa moramo paziti, da se vprašanja shranijo na tak način, da jih bo lahko uporabnik kadarkoli urejal, prenašal, pošiljal in pregledoval, tudi brez aplikacije. Med uporabniki želimo namreč spodbuditi deljenje paketov na najlažji način. Prav tako mora biti paket shranjen na način, da se ob brisanju aplikacije sam paket ne poškoduje in lahko po novi namestitvi aplikacije uporabnik učenje nadaljuje. Paket se prav tako ne sme poškodovati pri urejanju, zato mora biti sestavljen tako, da lahko pri urejanju vprašanj vplivamo na vsako vprašanje posebej.

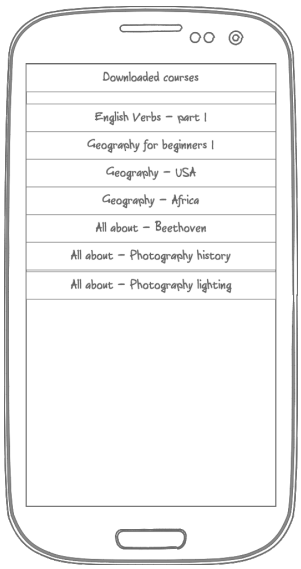
4.2.4 Predogled uporabniškega vmesnika

Za kreacijo predogleda se je poleg papirja uporabljalo orodje podjetja NinjaMock, ki je omogočalo brezplačno in enostavno kreiranje ter sam predogled. Se je pa izmed vseh ostalih preizkušenih orodij najbolj izkazalo zato, ker vsebuje veliko število samih Androidovih komponent (gumbi, dialogi ...).

Na sliki 2 lahko vidimo začetni zaslon aplikacije, na katerem so nam ponujeni štirje gumbi: »Izberi paket«, »Prenos paketov«, »Urejanje paketov« in »Navodila za kreacijo«.

Slika 3 pa prikazuje seznam paketov, ki jih je uporabnik že prenesel. Do tega zaslona pridemo, če na prvem izberemo »Izberi paket«.

Slika 2: Začetni zaslon aplikacije



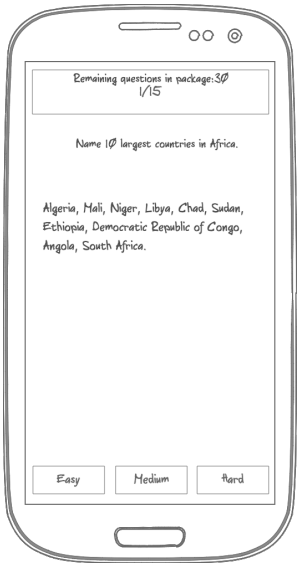
Slika 3: Prikaz prenesenih paketov

Slika 4 predstavlja zaslon učenja, ki se nam prikaže, ko izberemo enega izmed prenesenih paketov. Na vrhu se nahajata prikazovalnik števila vprašanj, na katera moramo odgovoriti v trenutni seansi, in število vseh vprašanj v paketu. Pod njima se nahaja vprašanje in gumb »Prikaži odgovor«.

Po kliku na ta gumb se nam, kot je razvidno iz slike 5, prikaže odgovor pod vprašanjem, namesto gumba »Prikaži odgovor« pa se prikažejo trije novi gumbi: »Lahko«, »Srednje« in »Težko«. S klikom na enega izmed njih se nam prikaže naslednje vprašanje in zaslon bo enak kot na prejšnji sliki. Če pa je bilo vprašanje zadnje, se namesto vprašanja izpiše »Trenutno ni vprašanj«, namesto gumba »Prikaži odgovor« pa imamo gumb »Nazaj na seznam«.

Slika 6 prikazuje zaslon za prenos novih paketov, ki ga dobimo, če na zaslonu iz slike 2 kliknemo na gumb »Prenos novih paketov«. Po izbiri enega izmed paketov se nam pojavi dialog, ki prikazuje stanje prenosa. Po zaključenem prenosu ostanemo na tem

Slika 4: Prikaz vprašanja



Slika 5: Prikaz vprašanja in odgovora po kliku na gumb »Prikaži odgovor«

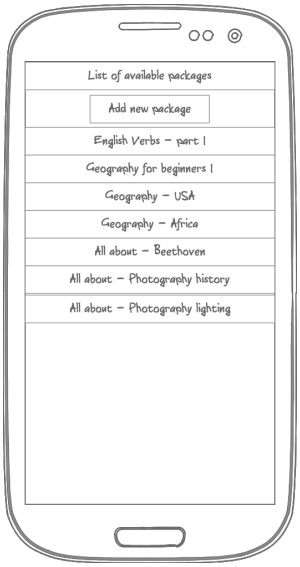
zaslonu in lahko po želji prenašamo nove pakete ali pa se vrnemo na prejšnji zaslon.

Slika 7 prikazuje zaslon za urejanje paketov. Tu lahko izbiramo med obstoječimi paketi ali pa s klikom na gumb »Dodaj paket« ustvarimo novega.

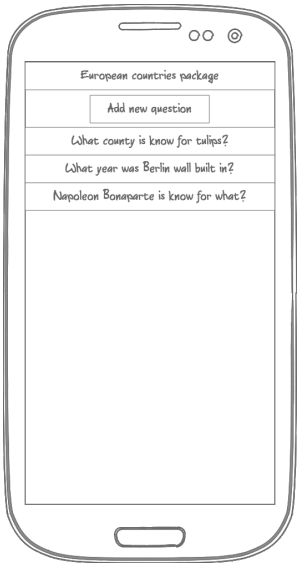
Ob kliku na obstoječi paket se nam prikaže zaslon (slika 8) z vprašanji paketa ter gumbom za dodajanje novega vprašanja.

S klikom na gumb za dodajanje vprašanja se nam prikaže drialog za dodajanje vprašanja (slika 9), če pa kliknemo na vprašanje, dobimo dialog za urejanje vprašanj (slika 10).

Slika 6: Seznam paketov na strežniku



Slika 7: Seznam paketov — zaslon urejanje



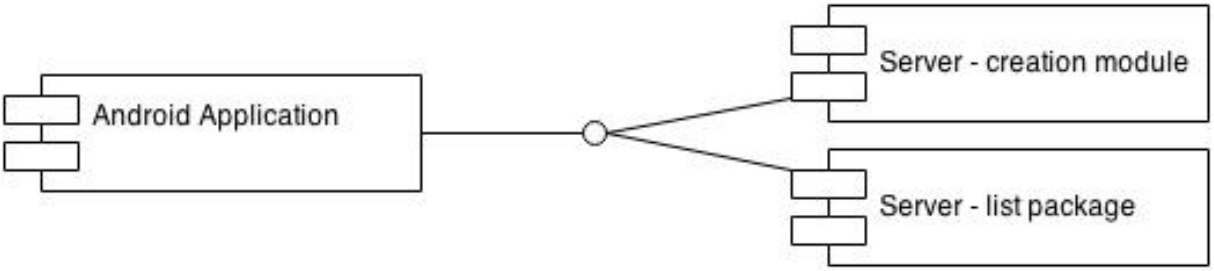
Slika 8: Urejanje izbranega paketa

Slika 9: Dodajanje novega vprašanja

Slika 10: Urejanje vprašanja

5 Načrtovanje sistema

Naš sistem je sestavljen iz dveh komponent: aplikacije (v tem primeru aplikacije Android, ki se pa lahko razširi na ostale sisteme) in strežniškega dela. Med seboj sta povezana preko JSON-datotek, s katerimi delujeta oba. Strežnik jih ustvarja, aplikacija pa nato uporablja (ima pa tudi aplikacija sama možnost ustvarjanja datoteke).



Slika 11: Android in strežniške komponente

5.1 JSON-datoteka

Da bi bolje razumeli njihovo povezavo, si moramo najprej ogledati JSON-datoteko in razumeti, v kakšni obliki jo strežnik ustvari in aplikacija nato uporablja.

```
"questions":[
        {
                "question":"Kaj pomeni beseda fact?",
                "answer":"Dejstvo.",
                "answerUrl":"nullUrl",
                "easyCounter":"0",
                "time":"201311152222"
        },
        {
                "question":"Kaj pomeni beseda Forrest?",
```

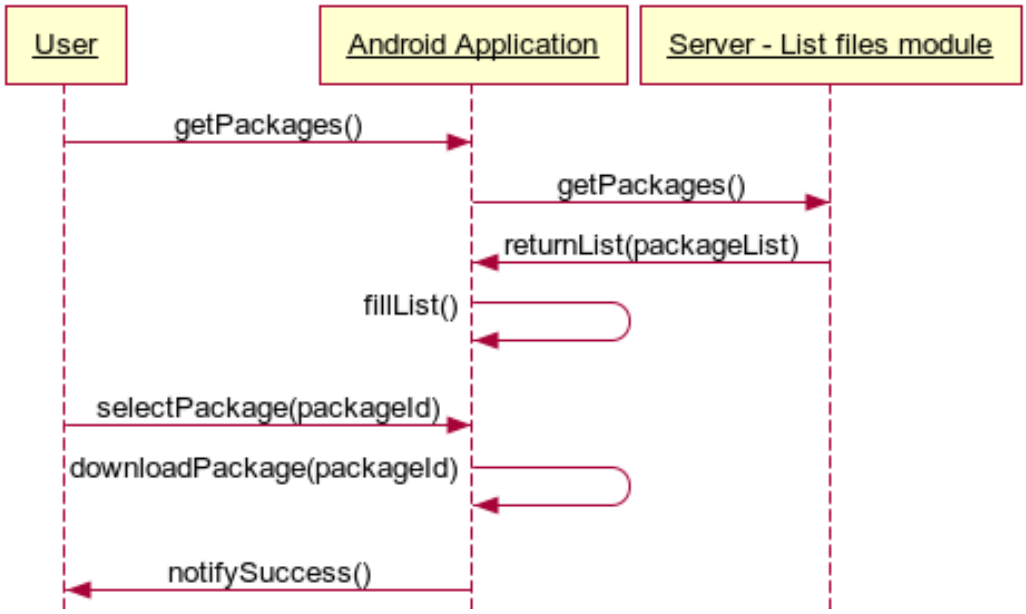
```
            "answer":"Gozd.",
            "answerUrl":"nullUrl",
            "easyCounter":"0",
            "time":"201311152222"
    }
```

V razdelku »questions« se bodo nahajala vsa vprašanja. Vsako izmed njih pa bo imelo naslednje lastnosti:

- question – vprašanje, ki ga bo uporabnik vnesel pri kreiranju paketa,
- answer – odgovor, ki ga bo uporabnik vnesel pri kreiranju paketa,
- answerUrl – če bo uporabnik pri kreaciji vnesel URL-naslov do slike, posnetka ali spletne strani, se bo le-ta tu izpisal, v nasprotnem primeru bo modul za kreacijo paketa kot vrednost vnesel »nullUrl«,
- easyCounter – privzeta vrednost tega števca bo pri kreaciji paketa vedno 0 in se bo spreminjala le v aplikaciji, ko bo uporabnik na vprašanje odgovoril z izbiro gumba »Lahko«,
- time – vrednost je kot privzeta nastavljena na zgoraj videno, iz katere se lahko lepo razloči format datuma in ure. Če je trenutni čas večji (besedo večje uporabljamo zato, ker za primerjavo časov uporabljamo funkcijo večje/manjše in čas v zapisani obliki obravnavamo kot število) od zapisanega, se vprašanje lahko prikaže. Ta vrednost se spremeni vsakič, ko uporabnik zaključi učno seanso – pomembo je dejstvo, da uporabnik seanso zaključi, saj se v nasprotnem primeru odgovori in spremembe ne upoštevajo. Vrednost 201311152222 v zgornjem primeru pomeni leto 2013, 11. mesec, 15. dan, ura pa je 22.22.

5.2 Povezava aplikacije in strežnika — prenos paketa

Ob prenosu paketa aplikacija strežniku pošlje zahtevo za seznam vseh paketov. Če od strežnika ne dobi odgovora (zaradi napačnega naslova v nastavitvah ali neobstoječe internetne povezave), se prikaže ustrezno obvestilo o napaki. Strežnik nato vrne seznam paketov v datoteki .json, ki jo aplikacija prebere in izriše uporabniku. S klikom na enega izmed izrisanih paketov ponovno sproži poizvedbo do strežnika, pri čemer se izbrani paket, ki je prav tako v datoteki .json, prenese na napravo samo. Pri tem aplikacija preveri, ali na mobilnem telefonu obstaja direktorij, v katerega se bodo paketi shranjevali. Če ga ni, ga v tem trenutku ustvari. Med prenosom aplikacija prikaže dialog, ki nakazuje potek samega prenosa. Ob uspešnem prenosu nas obvesti. Če se zaradi napake pri podatkovnem prenosu ta upočasni oziroma zatakne, aplikacija nekaj časa poskuša paket prenesti, in če ni uspešna, uporabnika o tem obvesti. Prenos v ozadju je prav tako mogoč.



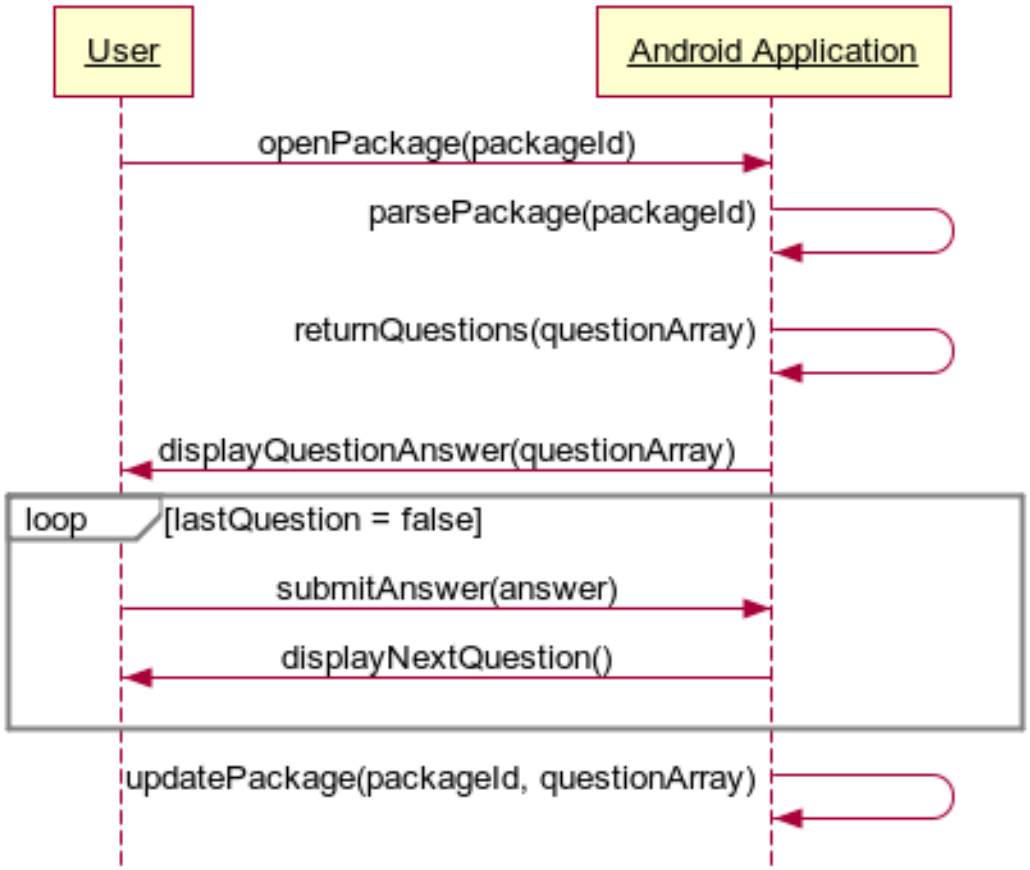
Slika 12: Diagram prenosa paketa

5.3 Komponenta Android

Aplikacija se deli na tri komponente: učenje, ustvarjanje paketov in urejanje paketov (če seveda obravnavamo le aplikacijske komponente in opustimo povezavo med aplikacijo in strežnikom).

5.3.1 Učenje

Učenje je poglavitna funkcija aplikacije. Ob njenem začetku se sprehodi skozi datoteko .json izbranega paketa in pri vsakem vprašanju preveri, ali je trenutni čas večji od tistega, ki je zapisan v lastnosti »time«. Če je tako, se vprašanje doda na kup vprašanj, ki jih bomo uporabniku ponudili. Po končanem pregledovanju časov vseh vprašanj iz nastavitev preberemo, koliko vprašanj želi uporabnik imeti v eni seansi. Iz našega kupa nato naključno izberemo poljubno število vprašanj, nakar se učenje lahko začne. Če pri nobenem vprašanju čas ni večji od trenutnega, se izpiše sporočilo, da vprašanj trenutno še ni na voljo. Izpiše se pa tudi čas, ko bo na voljo naslednje vprašanje.



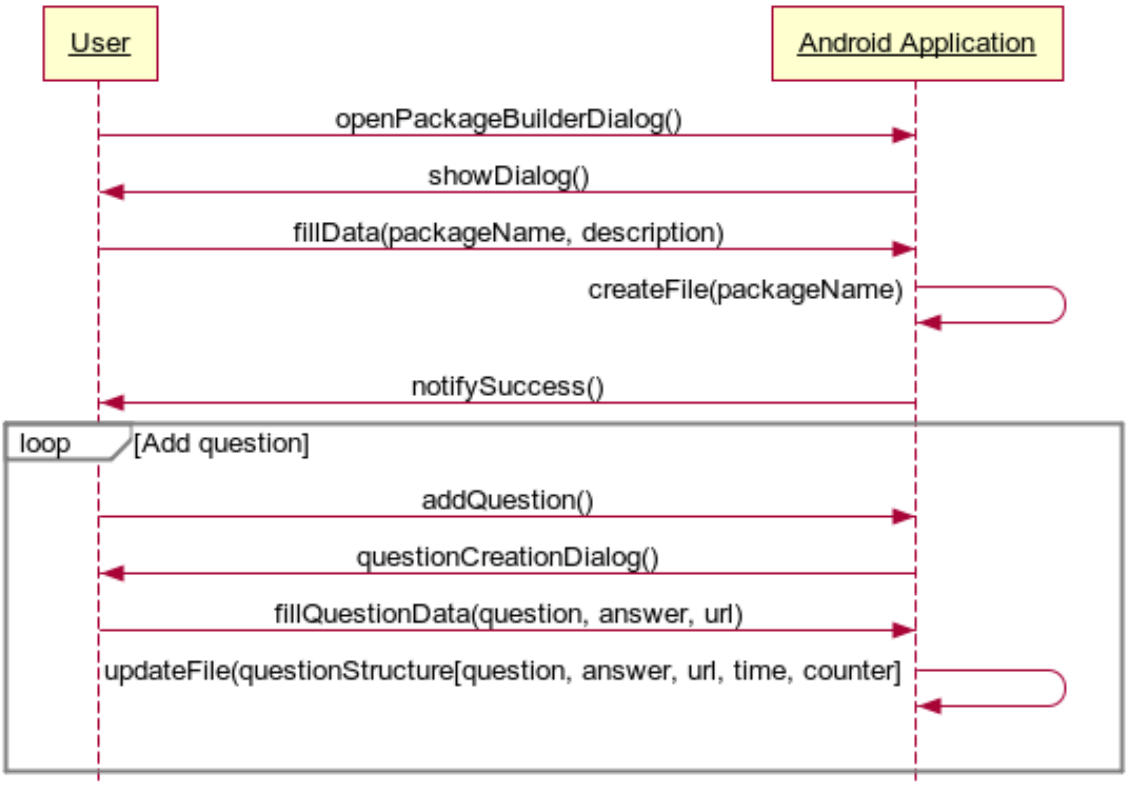
Slika 13: Diagram poteka učenja

Ob odgovorjenem vprašanju in izbrani eni izmed ponujenih možnosti se na nov

kup zapišejo podatki o vprašanju, pri katerih je »time« nastavljen na trenuten čas, »easyCounter« pa je povečan za eno število, če je uporabnik pri kliku na enega izmed gumbov izbral možnost »Lahko«. Z odgovorom na zadnje vprašanje se datoteka, v kateri se nahajajo vprašanja, posodobi s podatki s kupa, na katerem so uporabnikovi odgovori.

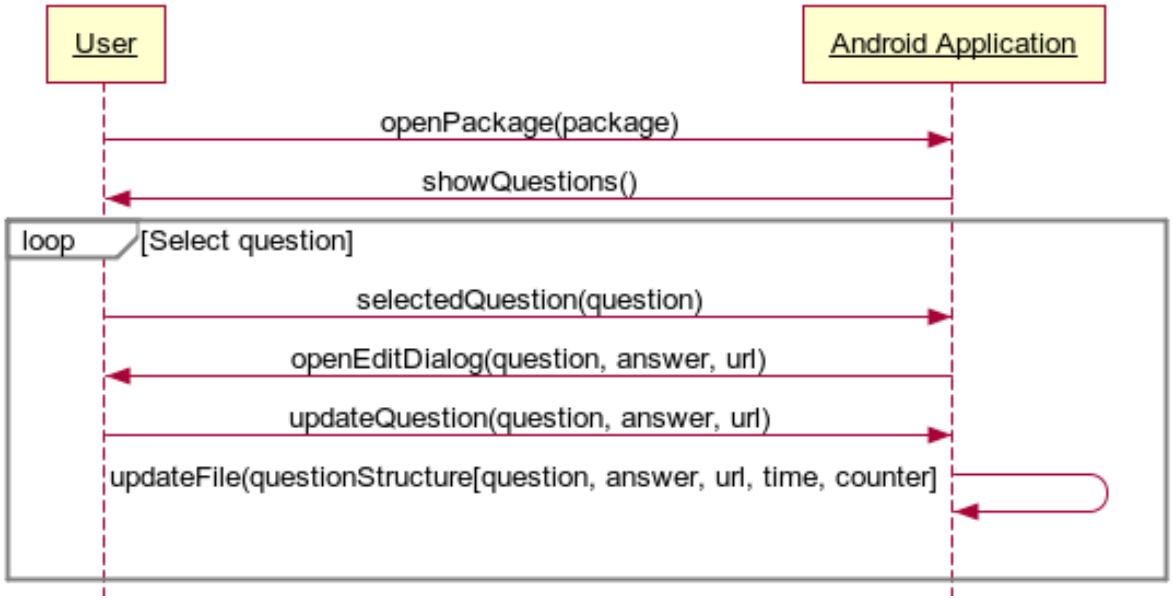
5.3.2 Kreiranje in urejanje paketov

Pri kreaciji novega paketa se ustvari datoteka JSON, napolnjena s prazno strukturo, prilagojeno naši aplikaciji. Dodajanje novega vprašanja je pri tem delu enako kot dodajanje novega vprašanja pri urejanju paketov.



Slika 14: Diagram kreiranja paketa neposredno v aplikaciji

Urejanje paketov ponuja dve možnosti: dodajanje novega vprašanja (enako kot pri kreiranju paketa) in urejanje vprašanj, kjer lahko spremenimo vprašanje, odgovor in/ali URL-naslov za odgovor.



Slika 15: Diagram urejanja paketov

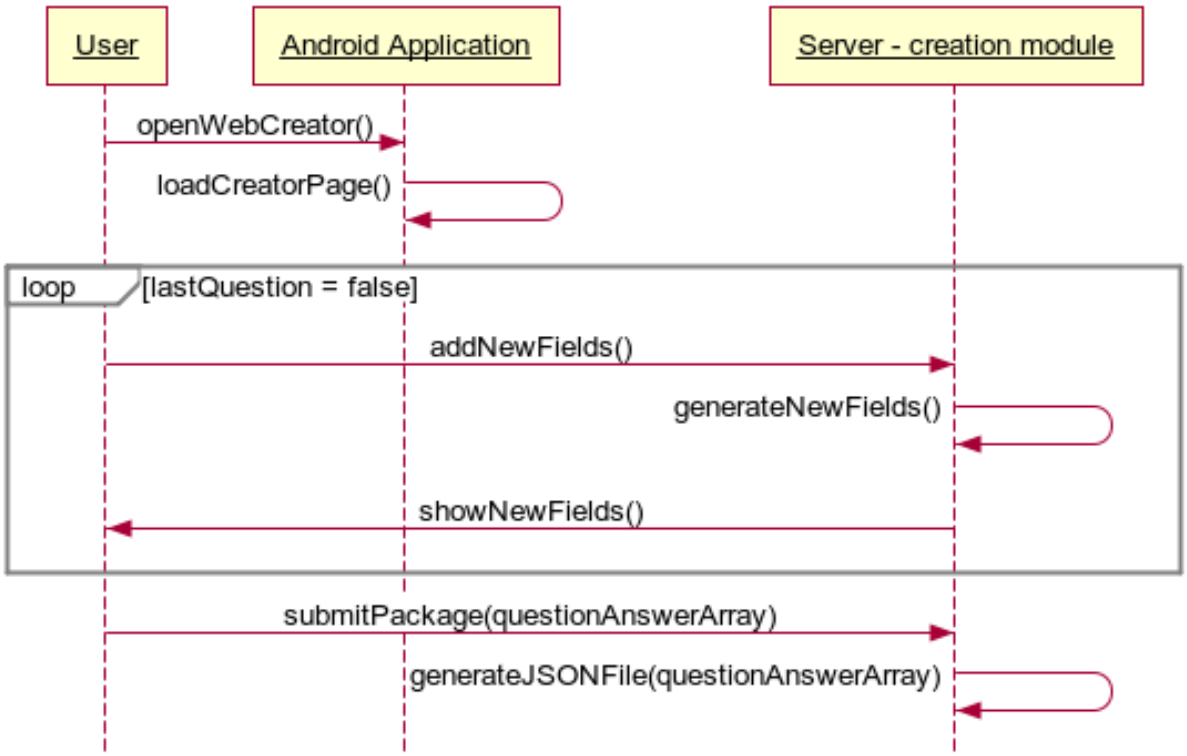
5.3.3 Uporabniške nastavitve

Uporabniške nastavitve (pogostost ponavljanja vprašanja, število vprašanj na seanso, notifikacije in URL-naslov do strežnika), ki si jih bo uporabnik lahko prilagodil na zaslonu za nastavitve, bomo shranjevali v komponenti *SharedPreferences* [7].

5.4 Komponenta strežnik

Strežnik bo naši aplikaciji ob vsaki kreaciji paketa pripravil datoteko s seznamom vseh paketov, ki se na njem nahajajo. Aplikacija bo zahtevek za datoteko poslala po HTTP-protokolu na strežnik, nastavljen v uporabniških nastavitvah, s čimer bo prejela datoteko s povezavami do datoteke na njem.

Strežnik pa omogoča kreiranje paketa. Pri dokončanem kreiranju se ustvari JSON-datoteka z vnesenimi vprašanji in posodobljena datoteka s seznamom vseh paketov, ki se na strežniku nahajajo.



Slika 16: Diagram kreiranja paketa preko brskalnika

6 Načrtovanje komponent

Za optimalno izkoriščenost potrebuje naš sistem dve komponenti: aplikacijo za mobilni telefon (v našem primeru je to OS Android) in strežniški del.

Aplikacija je vmesnik, preko katerega uporabnik uporablja poglavitne funkcionalnosti našega sistema — učenje ter kreacijo in urejanje paketov.

Da pa lahko ustvarja pakete, ki bodo za prenos na voljo ostalim uporabnikom, je pomemben strežniški del. Ta prav tako vsebuje sistem za kreacijo paketov, ki so nato vidni vsem uporabnikom, povezanim na ta strežnik.

6.1 Aplikacija Android

Najpomembnejša funkcionalnost, ki mora delovati vedno pravilno, je modul *WriteToFile*, ki smo mu namenili največ pozornosti in mora biti kot tak zelo robusten, saj lahko v nasprotnem primeru izbriše uporabnikov napredek. Prav njega smo zato uporabili tudi na različnih mestih. So pa pomembni tudi ostali moduli in funkcionalnosti, ki si jih bomo pogledali preko sekvenčnih diagramov.

6.1.1 Učenje

Po uporabnikovi izbiri paketa se najprej izvede metoda *ReadFromFile*, s pomočjo modula *JSONParser*, ki iz izbrane JSON-datoteke prebere vprašanja, nakar se izmed vseh vprašanj izberejo in vrnejo le tista, ki so že pripravljena za učenje (datum in čas v strukturi vprašanja morata biti manjša od trenutnih). Ostala vprašanja se kot objekti prestavijo v začasno JSON-strukturo.

Ker nas (performančno) ne stane, pred vsakim shranjevanjem vprašanja v strukturo preverimo uporabnikove nastavitve in ali so te spremenjene, saj to vpliva na zapis posameznega vprašanja.

FillQuestionAnswer poskrbi, da bodo izpolnjena polja za vprašanje in odgovor, ter jih naredi uporabniku vidna. Ob kliku na gumb »Prikaži odgovor« se prikažejo odgovor in gumbi, ki označujejo naše poznavanje odgovora. S klikom na enega izmed njih se posodobljeni objekt (vprašanje) premakne v prej ustvarjeno JSON-strukturo in z metodo *FillNextQuestion* se naloži naslednje vprašanje. Z odgovorom na zadnje vprašanje se vsebina datoteke prepiše z JSON-strukturo preko modula *WriteToFile*.

Tako dobimo posodobljeno datoteko, pri čemer pa modul *NotificationReceiver* poskrbi, da se nastavi alarm, ki bo ob pravem trenutku sprožil notifikacijo za učenje (v odvisnosti od prvega vprašanja, ki bo ponovno aktivno za učenje).

6.1.2 Prenos paketov

Za prenos paketa poskrbi modul *DownloadCourse*, ki po uporabnikovi izbiri želenega paketa iz prikazanega seznama z asinhronim procesom poskrbi, da se prenos začne. Med procesom se sproži dialog, ki uporabniku prikazuje stanje procesa prenosa. Da pa lahko uporabniku pred prenosom prikaže pakete, mora najprej s strežnika (katerega URL-naslov pridobi s pomočjo komponente *SharedPreferences*) prenesti seznam vseh paketov, ki se nahajajo na strežniku. Nato s pomočjo modula *ParseCourseList* iz datoteke izpiše pakete in vsakemu gumbu dodeli URL-naslov paketa na strežniku. Ta URL-naslov nato pri prenosu uporabi *DownloadCourse*.

6.1.3 Kreiranje in urejanje paketov

Modul za kreiranje paketov v aplikaciji *PackageEditorPicker* poskrbi za prikaz obstoječih paketov, ki jih lahko z modulom *PackageEditor* nato uredimo. *PackageEditorPicker* pa omogoča kreacijo novega paketa s pomočjo gumba »Ustvarjanje paketov«. Ta ustvari nov paket (v dialog se vnese naslov paketa) in prazno JSON-datoteko s predefinirano strukturo.

Ob izbiri novega ali kakega drugega paketa se zažene modul *PackageEditor*, ki s pomočjo funkcije *FillListwithQuestions* prikaže vsa vprašanja, ki jih lahko uredimo s funkcijo *EditQuestion*, lahko pa dodamo novo vprašanje z uporabo funkcionalnosti *AddNewQuestion*.

Obe metodi pri zapisu in kreiranju JSON-datoteke uporabita modul *WriteToFile*, ki se uporablja pri procesu učenja.

6.2 Strežniški del

Strežniški del je nekoliko enostavnejši. Sestavljen je iz dveh modulov: *ListFiles* in *Encode* ter prve strani, ki nam omogoča ustvarjanje paketa. Edina funkcionalnost te strani je, da nam ob kliku gumba doda nova vnosna polja. Ko zaključimo ustvarjanje paketa, se s pomočjo modula *Encode* ustvari JSON-datoteka. Hkrati pa se posodobi datoteka *courses.json*, ki se prav tako nahaja na strežniku in vsebuje le seznam vseh datotek za učenje. Nova datoteka se vedno ustvari v direktoriju *data*. *ListFiles* pa nam prikaže vse datoteke, ki se v tem direktoriju nahajajo, in hkrati osveži datoteko *courses.json*.

Pomembno je vedeti, da tako kreacija preko strežnika kot preko aplikacije vedno ustvarita enako strukturo in sta zato lahko kompatibilni.

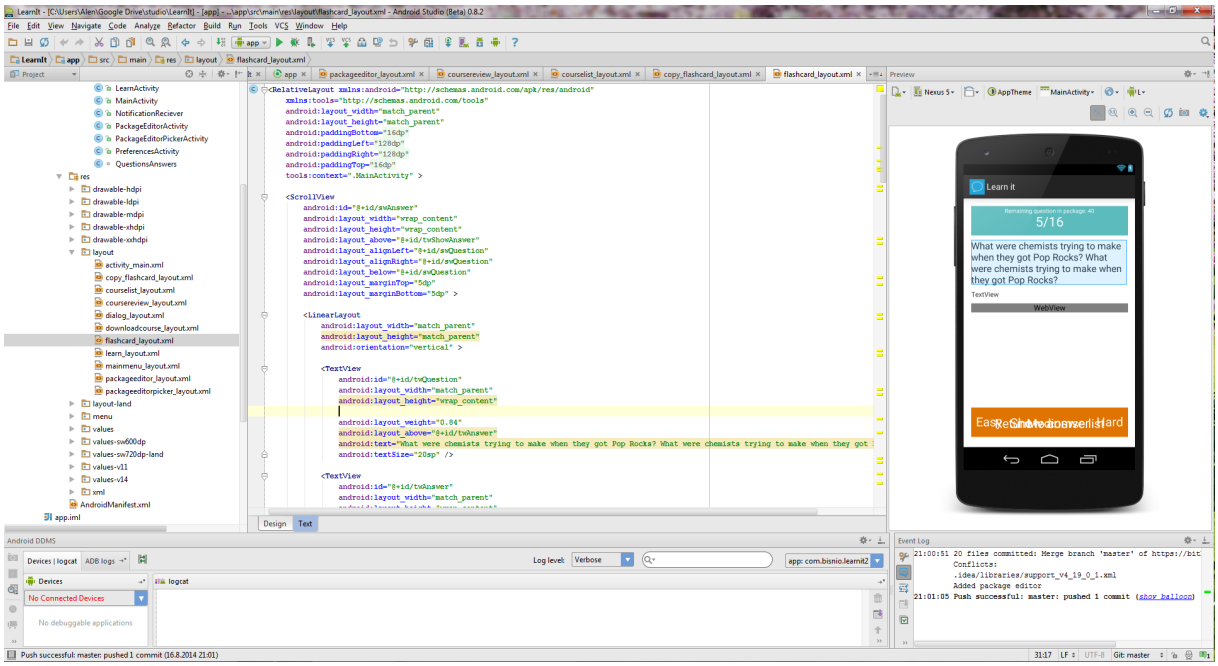
7 Izvedba

Da smo se lahko projekta lotili, smo si naprej morali namestiti vso potrebno programsko opremo, saj smo le tako lahko izpolnili vse svoje zahteve. Seveda smo si izbrali le brezplačne aplikacije, ki so pa seveda več kot dovolj učinkovite za naše potrebe.

Razvoj aplikacije je bil izveden v celoti v programskem jeziku Java, medtem ko smo strežniški del sestavili s pomočjo skriptnega jezika PHP. Za shranjevanje podatkov tako v aplikaciji kot na strežniku smo uporabili JSON-strukturo.

7.1 Android Studio

Za sam IDE smo imeli na izbiro Eclipse, ki je za razvoj aplikacij Android v uporabi že dlje časa, in Android Studio, ki ga je razvil in nedavno predstavil Google. Glede na to da Google razvija Android (z njegovim razvojem sicer ni začel, ampak je ustvarjalce in operacijski sistem kupil), smo se odločili, da izberemo Android Studio.



Slika 17: Android Studio

Za sam SDK nam ni bilo treba skrbeti, saj nas je za namestitev povprašal kar IDE sam. Prav tako nam omogoča uvoz in uporabo knjižnic, kar nam je prišlo prav pri

uporabi knjižnice Robotium za kreacijo in poganjanje unit testov nad aplikacijo.

Novost, ki je na voljo le v Android Studiu, pa nam je omogočila, da lahko predvidimo videz aplikacije na različnih napravah že med ustvarjanjem samega videza aplikacije. Gre namreč za možnost hkratnega pregleda nad XML-kodo in živega predogleda videza na simulirani napravi. Poleg tega omogoča pregled na več simuliranih napravah hkrati, kar je zelo zaželeno, saj so zasloni naprav, ki poganjajo operacijski sistem Android, različnih velikosti.

7.2 Grafični vmesnik

Kodiranje grafičnega vmesnika se sicer v celoti izvaja v Android Studiu, vendar pa smo kljub temu potrebovali grafiko za gumbe, ikono aplikacije in ostale elemente v aplikaciji sami. Ker nam Android Studio tega ne omogoča, smo za njihovo kreacijo uporabili GIMP, ki je odprtokodni in brezplačni grafični urejevalnik. Ta je več kot dovolj zmogljiv za naše potrebe, zato poseganje po plačljivih alternativah ne bi bilo smiselno.

7.3 Strežniški del

Glede na to da je naš strežniški del dokaj enostaven, nismo imeli potreb za kak zahtevnejši IDE in smo uporabili le urejevalnik tekstovnih dokumentov. Izbrali smo Notepad++, ki je prav tako brezplačen in nam je omogočil obarvanje kode ter s tem lažje pisanje. Za prenos kode na strežnik smo uporabili brezplačno aplikacijo FileZilla.

8 Testiranje

Pogledali si bomo White box in Black box testiranje aplikacije. Prav tako smo izvedli še beta testiranje delovanja aplikacije v njenem bistvenem pomenu — učenju. Na množici ljudi smo preverjali, ali se lahko z njeno pomočjo česa naučijo.

8.1 White box testiranje

Kot način white box testiranja smo izbrali unit teste. JUnit testi omogočajo enostavno implementacijo tudi za aplikacije Android. Za uspešno preizkušanje smo pa vseeno morali uporabiti zunanjo knjižnico, imenovano Robotium [6] . Metode, ki so nam pri preizkušanju najbolj pomagale, so:

- **assertCurrentActivity()**: metoda preveri, ali se nahajamo na pričakovanem zaslonu,
- **clickOnButton/clickOnText**: s pomočjo teh dveh metod smo se sprehajali med različnimi zasloni, saj sta simulirali klik na gumb ali tekst,
- **waitForText()**: ena izmed najuporabnejših metod pri preverjanju samega pravilnega delovanja — z njeno pomočjo lahko namreč preverimo, ali je na zaslonu trenutno sporočilo, ki ga pričakujemo (s tem lahko preverimo pravilnost izpisanega na zaslonu, v dialogih, nastavitvah ...),
- **enterText()**: metoda za vnašanje besedila v polja, uporabljena pri preizkušanju dodajanja/urejanja vprašanj in na zaslonu za nastavitve,
- **sendKey()**: simulira pritisk tipk, kot so menu in nazaj, in nam omogoča, da lahko pridemo do zaslona z uporabniškimi nastavitvami.

Po vsaki spremembi v kodi, ki bi lahko vplivala na delovanje aplikacije, smo pognali vse testne scenarije, ki so večkrat pripomogli k odkrivanju napak. Testni scenariji, ki smo jih lahko s pomočjo zgoraj omenjenih metod preizkušali, so:

- **pravilno delovanje zaslona za prikaz paketov**: preizkušali smo, ali se izpišejo paketi, če se ti nahajajo v direktoriju na telefonu, in ali se prikaže sporočilo, če paketov ni,

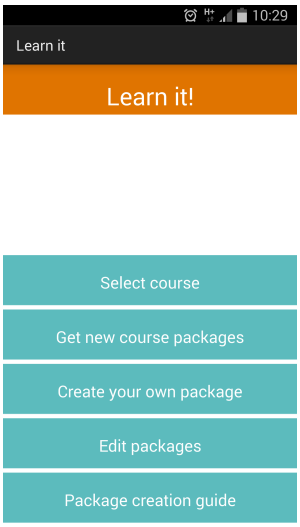
- **povezava do strežnika**: če je v nastavitvah vnesen URL do strežnika, so se ob kliku na gumb »Prenos paketov« morali v sprejemljivem času izpisati paketi, ki se na tem strežniku nahajajo, prav tako smo morali o poteku biti obveščeni z dialogom,
- **prenos paketa s strežnika**: ob kliku na paket se je moral začeti prenos, tudi v tem primeru smo morali biti obveščeni z dialogom,
- **kreiranje paketov znotraj aplikacije**: poskusili smo narediti paket in ga tudi poimenovati; paket smo odprli in ustvarili nekaj vprašanj,
- **urejanje paketov znotraj aplikacije**: paket, ki smo ga pri predhodnem preizkušanju ustvarili, smo sedaj ponovno odprli in prej ustvarjena vprašanja spremenili, nekatera izmed teh smo tudi poskusili odstraniti,
- **spreminjanje števila vprašanj v nastavitvah**: v uporabniških nastavitvah smo spremenili število vprašanj, ki jih želimo imeti v seansi, nato smo nastavitve zapustili in na zaslonu za učenje preverili, ali se število ujema (to smo poskusili za vse vrednosti),
- **spreminjanje URL-naslova za strežnik**: v uporabniških nastavitvah smo spremenili URL-naslov strežnika in preverili, ali se v primeru pravega naslova paketi na zaslonu za prenos prikažejo in ali se, če je URL-naslov napačen, na omenjenem zaslonu prikaže primerno sporočilo,
- **učenje**: s pomočjo unit testa smo lahko preizkušali, ali se ob kliku na gumb »Prikaži odgovor« ta pravilno skrije in ali se gumbi »Lahko«, »Srednje« in »Težko« prikažejo pravilno. Prav tako smo lahko preverili, ali se po končanem odgovarjanju na vsa vprašanja prikaže pravi zaslon, ki nas obvesti, da smo učenje zaključili.

8.2 Black box testiranje

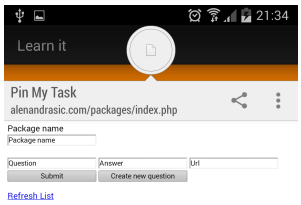
Vseh stvari se seveda z unit testni ne da preizkusiti, zato se moramo za dodatno preizkušanje ročno sprehoditi skozi vse funkcionalnosti aplikacije in strežniškega dela. Tak način preizkušanja smo izbrali le ob večjih spremembah, saj za razliko od avtomatskih unit testov zahteva veliko našega časa.

Glede na to da brez paketov aplikacije same ne moremo testirati, smo se odločili, da najprej preverimo delovanje funkcionalnosti kreacije paketov.

8.2.1 Kreacija paketa — brskalnik

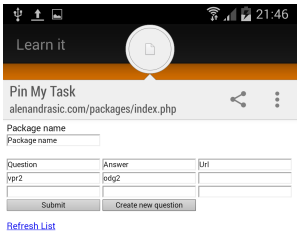


Slika 18: Začetni zaslon

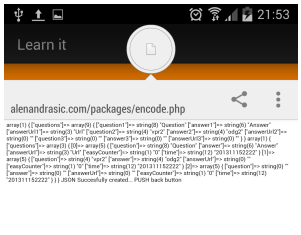


Slika 19: Kreacija paketa preko brskalnika

Ob kliku gumba »Kreiraj svoj paket« se nam, kot je razvidno iz slike 19, v brskalniku odpre stran, preko katere lahko testiramo modul za kreacijo paketa. Izpolnili smo začetna polja, ki so nam že prikazana. Ob kliku na gumb »Dodaj novo vprašanje« smo preverjali, ali se pravilno dodajo vsa tri nova polja, ne glede na to, ali so predhodna že izpolnjena ali ne.



Slika 20: Dodajanje vprašanja — brskalnik

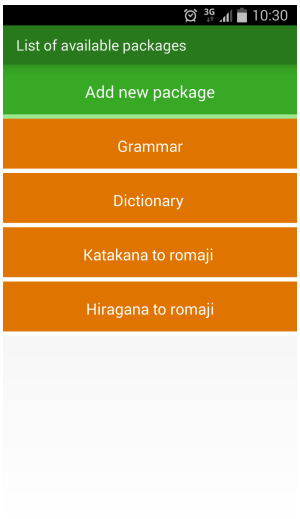


Slika 21: Uspešno kreirani paket

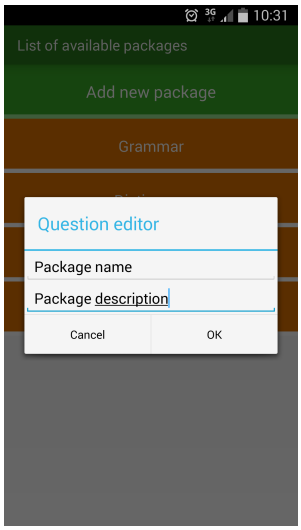
Dodajanje novih polj za vprašanja težav ni povzročalo. Za zaključek kreacije paketa smo nato kliknili gumb »Oddaj«, ki nam je zaradi lažjega pregleda nad potekom kreacije izpisal kreirano datoteko .json. V produkcijski verziji aplikacije bi bil prikazan zaslon drugačen, vendar lahko tudi tu iz zadnje vrstice razberemo, da je bila kreacija uspešna (»Succesfully created ... PUSH back button«). Da lahko potrdimo uspešno kreacijo, se nato odpravimo na zaslon za prenos novih paketov, kjer ugotovimo, da se naš na novo kreirani paket zares nahaja na seznamu.

8.2.2 Kreacija in urejanje paketa — aplikacija

Ob prihodu na zaslon za kreiranje in urejanje paketa se nam izriše seznam vseh paketov, ki jih imamo na voljo. Nad njimi je gumb »Ustvarjanje paketa«, ki nam odpre dialog za vnos naslova in opisa paketa. Preizkušali smo, ali se ob kreiranju paketov ta takoj prikaže na seznamu in ali mu kak poseben znak povzroči težave. Pri sami kreaciji smo nato preverili še strukturo JSON-datoteke, ki mora biti ustreznega formata, da lahko uporabnik dodaja nova vprašanja.



Slika 22: Urejanje — seznam paketov

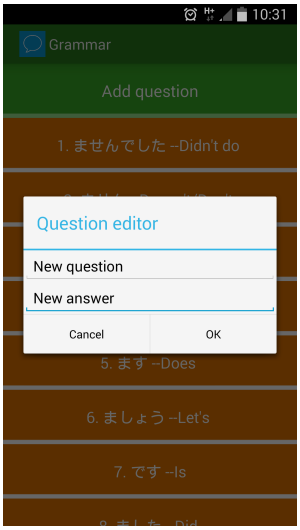


Slika 23: Dodajanje novega paketa

Ob izbiri enega izmed paketov se nam nato izpišejo vsa vprašanja, ki se v njem nahajajo. Ponovno se na vrhu pojavi gumb, vendar tokrat za dodajanje novega vprašanja, pri katerem smo preizkušali enake stvari kot pri kreaciji paketa, saj dialog in seznam delujeta na enak način.

Zadnji preizkus je urejanje in brisanje vprašanj. Ob kliku na eno izmed vprašanj se nam prikaže dialog, ki nam omogoča urejanje in izbris — polji za vprašanje in odgovor sta že izpolnjeni, poleg gumbov »Ok« in »Prekliči« se nam pa izriše še gumb »Odstrani vprašanje«. Takoj po spremenjenem podatku in kliku na gumb »Ok« se je moralo vprašanje posodobiti, če smo pa kliknili na gumb »Prekliči«, so se morale spremembe razveljaviti. Prav tako se je moral seznam vprašanj takoj posodobiti, če smo izbrali gumb »Odstrani vprašanje« in s tem vprašanje odstranili.

Po končanem odstranjevanju in dodajanju vprašanj smo morali preveriti, ali so ostala vprašanja v sami datoteki JSON ostala nespremenjena.



Slika 24: Dodajanje vprašanja — aplikacija

Slika 25: Urejanje vprašanja

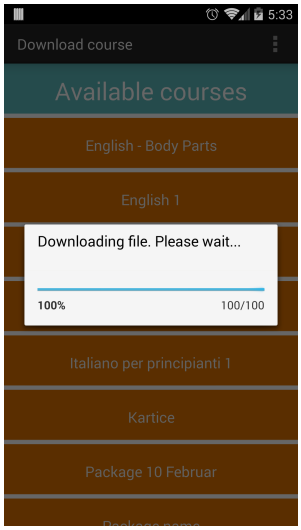
8.2.3 Prenos paketov

Naslednji korak je testiranje prenosa paketov. Ob vstopu na zaslon se nam najprej prikaže dialog, ki prikazuje nalaganje prikaza paketov, ki so na voljo. Če na strežniku ni paketa oziroma je pri prenosu prišlo do napake, se uporabniku prikaže ustrezno sporočilo. V nasprotnem primeru se uporabniku seveda prikažejo naslovi paketov. Ker se lahko pri prenosu zaradi slabe internetne povezave ali napake na strežnišfki strani kaj zalomi, smo implementirali funkcionalnost »pull to refresh«, ki uporabniku omogoča, da z enostavnim potegom akcijo prenosa seznama paketov ponovi (med osveževanjem lahko vidimo premikajoče se barvne črte na vrhu seznama).

Preizkuševalcem se je nekajkrat zgodilo, da se je med prenosom povezava prekinila, in jim je zato ta funkcionalnost prišla še kako prav, z njo pa se niso v nobenem primeru pojavile težave.

Po kliku na želeni paket se nam je vnovič prikazal dialog, ki je prikazoval stanje prenosa, vendar je tokrat prikazoval stanje prikaza izbranega paketa. Da preverimo, ali se je paket pravilno prenesel, odpremo zaslon za učenje, kjer bi nas moral pričakati na seznamu prenesenih paketov.

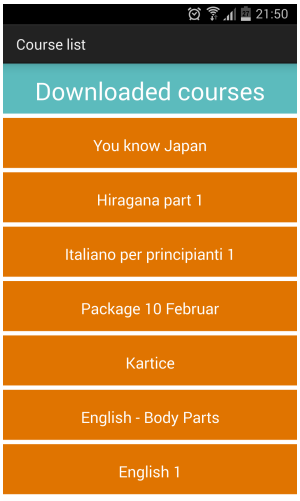
Slika 26: Seznam paketov na strežniku



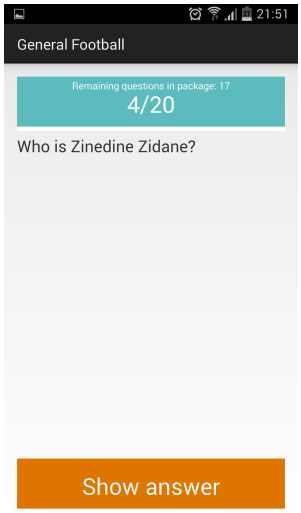
Slika 27: Prenos paketa

8.2.4 Učenje

Ob kliku na gumb za učenje se nam prikaže, kot lahko vidimo na sliki 28, seznam paketov, ki smo jih že prenesli. S klikom na enega izmed teh pa se nam odpre zaslon za učenje.



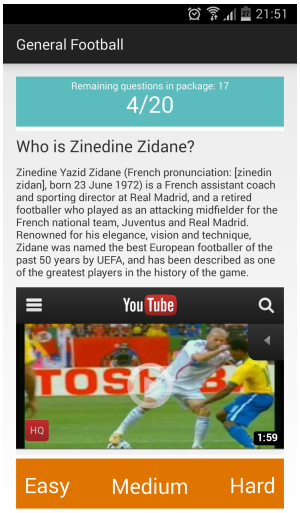
Slika 28: Seznam prenesenih paketov



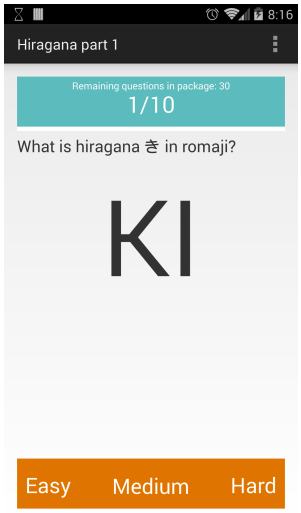
Slika 29: Prikaz vprašanja pri učenju

Na tem zaslonu nam je najprej na voljo vprašanje z nekaj podatki o paketu samem in gumb »Prikaži odgovor« (slika 29). S klikom nanj se nam (slika 30) prikaže odgovor, ki je lahko tekstovni, spletna stran, videoposnetek ...

Na sliki je prikazana vsebina, ki je za test najprimernejša, in sicer je to tekstovni odgovor z dodano povezavo do posnetka na YouTubu, kar je treba pravilno prikazati, saj se mora biti uporabnik zmožen premikati navzgor in navzdol ne glede na dolžino.



Slika 30: Odgovor — spletna vsebina



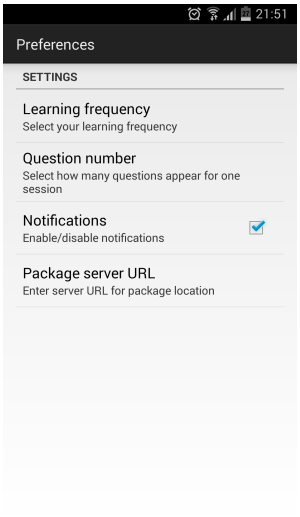
Slika 31: Odgovor — tekstovna vsebina

Pri testu smo morali biti pozorni tudi na prikaz spletne strani v navezi s tekstovnim odgovorom, saj se mora biti uporabnik tam zmožen premikati v vse smeri, hkrati pa tudi navzgor in navzdol neodvisno od spletne strani. To nam je uspelo zaradi pravilne uporabe Androidovega WebViewa v kombinaciji s ScrollViewom. Upoštevati smo pa morali tudi dolžino odgovora samega, saj bi bil prikaz kratkega odgovora z manjšo pisavo (ki jo uporabimo pri daljših odgovorih) neroden in nepregleden. Ravno zato smo aplikacijo pripravili tako, da ima na izbiro več velikosti pisav, ki jo nato sama izbere glede na dolžino odgovora (slika 31).

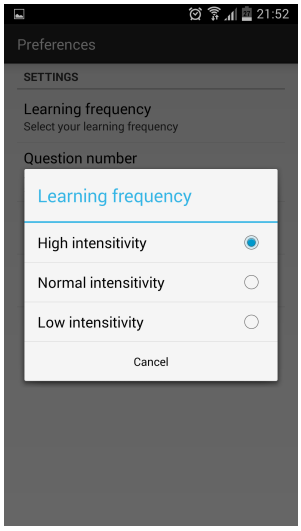
S klikom na enega izmed treh gumbov aplikacija v ozadju opravi svoje, nam pa se prikaže naslednje vprašanje. Postopek ponavljamo, dokler nas ne dočaka zaslon, ki nam sporoči, da smo učenje zaključili.

8.2.5 Preverjanje nastavitev

Čeprav nastavitev ni veliko, te močno vplivajo na uporabniško izkušnjo, zato jih je bilo treba testirati do potankosti. Najprej smo preizkusili nastavitve frekvence za učenje. Da uporabnika razbremenimo razmišljanja o tem, kako pogosto se želi učiti, mu namesto časovnih merskih enot ponudimo tri možnosti. To so: »Visoka intenzivnost«, »Srednja intenzivnost« in »Niza intenzivnost«. Te lahko uporabnik med učenjem kadarkoli spreminja in že v naslednji seansi se bodo spremembe upoštevale. Ker je v časovnem obdobju vrednosti, ki jih ponudimo končnemu uporabniku, težko testirati, smo za prvo fazo testiranja te vrednosti zmanjšali, in sicer na dve minuti za visoko intenzivnost, štiri minute za srednjo in šest minut za nizko. Najprej smo na vsa vprašanja odgovorili z vrednostjo, nastavljeno na visoko intenzivnost. Takoj ko smo na vsa vprašanja odgovorili, smo paket še enkrat zagnali, vendar nas je, kot pričakovano, pričakalo sporočilo, da vprašanj trenutno ni na voljo. To smo ponovili čez tri minute, ko pa so nas že pričakala vsa vprašanja. Enako smo storili tudi za ostali dve nastavitvi frekvence. Za srednjo intenzivnost smo poskusili zagnati paket takoj, ko smo ga rešili, tri minute za tem in nazadnje še pet minut za tem, ko so se po pričakovanju prvič pojavila vprašanja. Tudi na nizki intenzivnosti so se vprašanja pojavila, ko smo pričakovali.



Slika 32: Uporabniške nastavitve

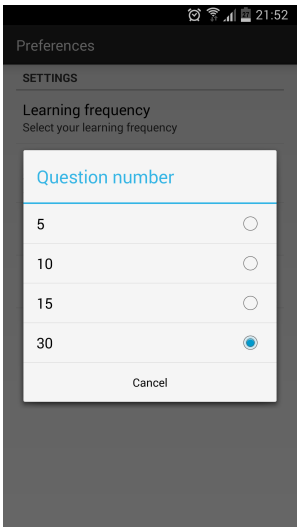


Slika 33: Frekvenca ponavljanja vprašanj

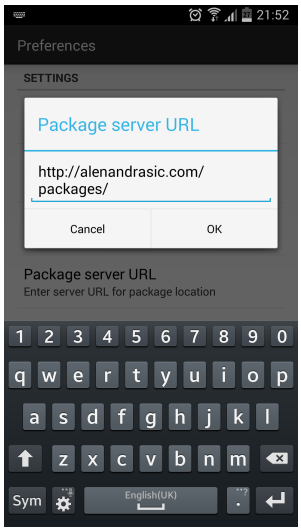
Test smo ponovili še, da smo na vprašanja odgovarjali mešano in vselej smo po določenem številu pretečenih minut prejeli le tista vprašanja, ki bi jih po pretečenem času pričakovali. Ker je ta funkcionalnost aplikacije ena izmed najpomembnejših, smo test večkrat ponovili. Na koncu smo časovne vrednosti nastavili na končne vrednosti, izražene v urah. Te smo več dni testirali in vselej se je aplikacija izkazala.

Testiranje se pa dokončno zaplete, ko na neko vprašanje večkrat zaporedoma odgo-

vorimo s klikom na gumb »Lahko«. Zaradi večje učinkovitosti se bodo tista vprašanja, na katere smo odgovorili z izbiro tega gumba, redkeje ponavljala in tako dajala prednost vprašanjem, na katera ne odgovorimo z lahkoto. Tudi to funkcionalnost smo testirali najprej s krajšim časom, nato pa z daljšim. V vseh primerih nam je aplikacija ponudila vprašanja ob pričakovanem času, čeprav smo morali biti pri preverjanju ponavljanj kar natančni, saj je najdaljše obdobje, po katerem se vprašanje ponovi, kar en mesec. Pri preizkušanju ponavljanja vprašanj smo si pomagali tudi s sistemom za notifikacije. Kot je razvidno iz slike 32, nas lahko aplikacija obvesti, ko so vprašanja na voljo. Naslednja nastavitev, ki si jo lahko uporabnik prilagaja, je število vprašanj za posamezno seanso. Ker mora uporabnik seanso vedno izpeljati do konca, da se datoteka posodobi, je pomembno, da si izbere vrednost, ki mu najbolj ustreza. Če ima paket večje število vprašanj, kot si jih je uporabnik določil za posamezno seanso, lahko uporabnik ponavlja seanse za paket, dokler vprašanj ne zmanjka. Ampak ker je bistvo aplikacije to, da uporabnik odgovarja, ko ima čas, ni pomembo, ali odgovori na vsa vprašanja. Med naštetimi možnostmi smo za vsako izbrano odprli paket za učenje in preverili, ali se res prikaže pravo število vprašanj in ali se po odgovoru na zadnje vprašanje v seansi uporabnikova datoteka posodobi, ne glede na to, ali je vprašanje zadnje ali ne.



Slika 34: Število vprašanj v seansi

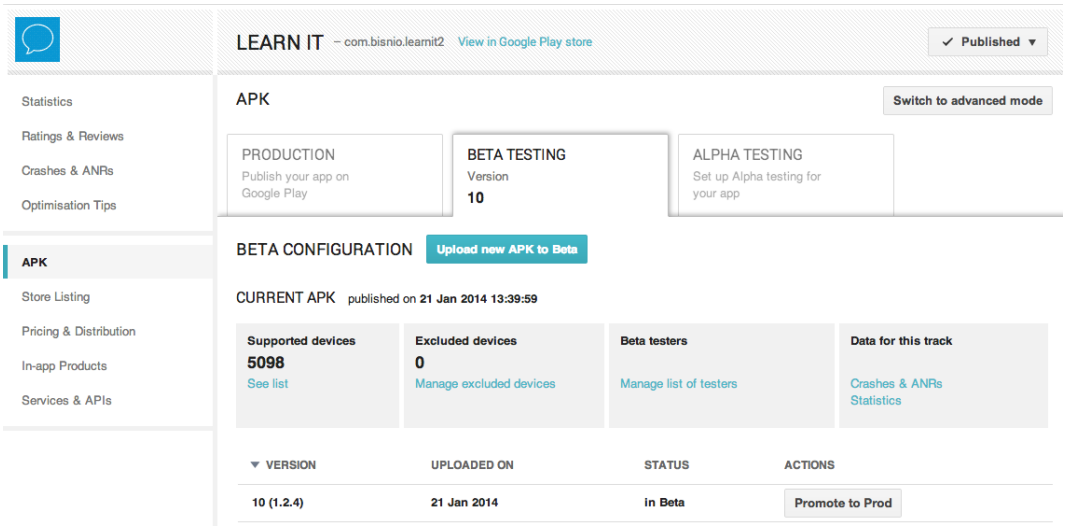


Slika 35: URL-naslov strežnika

Zadnja nastavitev je URL-naslov strežnika. Ta je privzeto nastavljen tako, da uporabnika usmeri na privzeti strežnik aplikacije, na katerem se nahajajo vsi javni paketi, namenjeni vsem uporabnikom aplikacije. Vendar pa je čar aplikacije v tem, da si lahko strežniški modul namesti vsak, ki ima svoj strežnik ali gostovanje. Vse, kar mora storiti, je, da sledi navodilom za postavitev strežniškega modula in nato v polje »URL strežnika paketov« vnese URL-naslov do strežnika, na katerem je nameščen modul.

8.3 Beta testiranje

Odločili smo se, da bomo aplikacijo preizkusili na množici uporabnikov in tako dobili povratne informacije o pomanjkljivosti aplikacije ter preizkusili, ali aplikacija zares pomaga pri učenju.



Slika 36: Beta testiranje

Na Play Storu smo si ustvarili račun, izdelali aplikacijo in seveda naložili namestitveno datoteko .apk. Na Googlovem socialnem omrežju smo ustvarili krog, v katerega smo povabili beta uporabnike, ki smo jim na Play Storu omogočili prenos aplikacije.

S pomočjo uporabnikov, ki so aplikacijo testirali, smo tako pripravili kar deset novih različic. Večjih napak v delovanju aplikacije same ni bilo, tako da posodobitve predstavljajo izboljšanje uporabniške izkušnje pri njenem delovanju. Uporabnikom je bilo naročeno, da aplikacijo med testiranjem uporabljajo za učenje (skladno s tem so bili pripravljeni paketi po njihovih željah) in med tem časom večkrat preverijo vse funkcionalnosti – od prenosa paketov do učenja samega in seveda spreminjanja nastavitev. Na koncu smo pri najvztrajnejših uporabnikih preverili tudi znanje (učenje novih angleških besed) in rezultati samega delovanja aplikacije so bili spodbudni. Znanje smo ponovno preverili nekaj mesecev po prenehanju uporabe in ugotovili, da je pri tistih, ki so aplikacijo uporabljali dlje časa, usvojeno znanje še vedno prisotno.

9 Zaključek

Da bi bil sistem še boljši, bi bilo treba vložiti še veliko časa. Da bi pa bil sistem poleg tega še uspešen, bi ga bilo treba razvijati neprestano, saj bi lahko le tako sledili trendom.

Prva izmed stvari, ki bi jo morali dodati, je možnost pošiljanja svojih učnih paketov ostalim uporabnikom aplikacije kar iz aplikacije same. Tudi trenutno je to mogoče, vendar moramo do paketov priti preko aplikacije za pregledovanje datotek, vendar uporabnik s tem deli tudi svoj napredek. Zato bi bilo treba napredek in sama vprašanja ločiti ali omogočiti ponastavitev napredka na paketu.

Temu bi sledila še možnost nalaganja paketa, ki smo ga ustvarili s pomočjo aplikacije, neposredno na strežnik.

Ker bi bilo tako počasi paketov na strežniku ogromno, bi bilo treba urediti tudi sistem kategoriziranja in preverjanja vsebine paketov, da ta ne bi bila sporna. Dodati bi bilo treba tudi način ocenjevanja paketov.

Naslednji korak bi bila izboljšava strežniškega dela, da bi tudi ta omogočala urejanje paketa in ne samo kreacijo.

Kot je razvidno, je že sedaj zamisli za nadgradnjo veliko. Vendar pa naš namen ni bil ustvariti popolno aplikacijo, vendar prikazati faze programskega inženirstva in pri tem ustvariti aplikacijo, ki bi lahko bila komu uporabna. S pomočjo te aplikacije bi lahko nekoga, ki o programskem inženirstvu ne ve veliko, marsikaj naučili.

10 Literatura

[1] P. A. Wozniak, Optimization of learning, Master's Thesis, University of Technology in Poznan, 1990.

[2] Favell Lee Mortimer, The Clumsiest People in Europe: Or, Mrs. Mortimer's Bad-Tempered Guide to the Victorian World, 2006 Edition.

[3] Pimsleur, Paul, A Memory Schedule, The Modern Language Journal (Blackwell Publishing): 73–75.

[4] AppBrain — Google Play Store statistic, http://www.appbrain.com/stats/number-of-android-apps, Accessed: 23. 8. 2014.

[5] Platform versions — number of devices running a given version, https://developer.android.com/about/dashboards/index.html?utm_source=ausdroid.net, Accessed: 24. 8. 2014.

[6] Robotium library, https://code.google.com/p/robotium/, Accessed: 23. 8. 2014

[7] Shared Preferences,

http://developer.android.com/reference/android/content/SharedPreferences.html, Accessed: 23. 8. 2014.

Priloge

Izvorna koda aplikacije, strežniškega dela in aplikacija sama se nahajajo na priloženem CD-ju.