Mobilna aplikacija za vježbanje čitanja s razumijevanjem na primjeru zadataka višestrukoga odabira

Agičić, Luka

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

Sveučilišni prijediplomski studij Računarstvo

Mobilna aplikacija za vježbanje čitanja s razumijevanjem na primjeru zadataka višestrukog odabira

Završni rad

Luka Agičić

Osijek, 2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P: Obrazac za ocjenu završnog rada na sveučilišnom prijediplomskom studiju

Ocjena završnog rada na sveučilišnom prijediplomskom studiju

Ime i prezime pristupnika:	Luka Agičić
Studij, smjer:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. pristupnika, god.	R4600, 27.07.2021.
JMBAG:	0165089100
Mentor:	doc. dr. sc. Dragana Božić Lenard
Sumentor:	Miljenko Švarcmajer, univ. mag. ing. comp.
Sumentor iz tvrtke:	
Naslov završnog rada:	Mobilna aplikacija za vježbanje čitanja s razumijevanjem na primjeru zadataka višestrukoga odabira
Znanstvena grana završnog rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rada:	Cilj je završnoga rada izraditi aplikaciju za vježbanje čitanja s razumijevanjem i primjene dosadašnjeg znanja engleskog jezika. Završni će rad biti izrađen u programskom okruženju Android Studio korištenjem programskog jezika Kotlin. Koristit će se i Jetpack Compose, moderni skup alata za korisničko sučelje, za razvoj mobilnih Android aplikacija. U sklopu će se završnog rada izraditi mobilna aplikacija za zadatak višestrukoga odabira. Tema rezervirana za Luku Agičića. Sumentor s FERIT-a Miljenko Švarcmajer.

Datum prijedloga ocjene završnog rada od strane mentora:	16.09.2024.
Prijedlog ocjene završnog rada od strane mentora:	Vrlo dobar (4)
Datum potvrde ocjene završnog rada od strane Odbora:	25.09.2024.
Ocjena završnog rada nakon obrane:	Vrlo dobar (4)
Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio sveučilišni prijediplomski studij:	26.09.2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O IZVORNOSTI RADA

Osijek, 26.09.2024.

Ime i prezime Pristupnika:	Luka Agičić
Studij:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R4600, 27.07.2021.
Turnitin podudaranje [%]:	12

Ovom izjavom izjavljujem da je rad pod nazivom**: Mobilna aplikacija za vježbanje čitanja s razumijevanjem na primjeru zadataka višestrukoga odabira**

izrađen pod vodstvom mentora doc. dr. sc. Dragana Božić Lenard

i sumentora Miljenko Švarcmajer, univ. mag. ing. comp.

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

Sadržaj

1. UVOD

S porastom upotrebe mobilnih tehnologija, pametni telefoni postali su neizostavan dio svakodnevnog života. Prema istraživanju iz srpnja 2024. godine, 70 % svjetske populacije koristi mobilni uređaj [1]. Jedan od načina kako provesti vrijeme za tako često korištenim uređajem je učenje. Učenje uz uporabu mobilnih telefona (eng. *Mobile Assisted Language Learning - MALL*) predstavlja inovativan, fleksibilan i personalizirani pristup učenju jezika korištenjem mobilnih uređaja [2]. Istraživanje iz 2012. godine pokazuje da 86 % učenja korištenjem mobilnih uređaja rezultira pozitivnim ishodom [3].

Mobilna aplikacija, koja je glavna tema ovog završnog rada, omogućava rješavanje zadatka s višestrukim odabirom (eng. Multiple Choice Cloze).

Nakon kratkog uvoda u tematiku i zadatak ovog završnog rada, predstavit će se trenutne mobilne aplikacije i mrežne stranice koje sadržavaju zadatke slične vrste kao iz naše mobilne aplikacije. U trećem su poglavlju opisani operacijski sustav Android, razvojno okruženje Android Studio, programski jezik Kotlin, radni okvir Jetpack Compose i Firebase baza podataka. U četvrtom poglavlju prikazan je razvoj aplikacije i funkcionalnosti. U petom poglavlju nalazi se zaključak u kojem su predstavljene glavne značajke aplikacije zajedno s njezinim prednostima i nedostatcima.

1.1. Zadatak završnog rada

Cilj je ovoga rada izraditi Android aplikaciju za rješavanje zadataka višestrukog odabira za čitanje s razumijevanjem. Aplikacija će sadržavati teorijsku osnovu (vokabular, fraze, razlike u značenju, poveznice i frazalne glagole) i zadatke višestrukog odabira iz različitih područja. U slučaju potrebe, korisnik će imati mogućnost dobivanja savjeta za rješavanje zadatka. Po završetku, dobit će povratnu informaciju o broju bodova, ocjeni te prikaz točnih rješenja.

2. PREGLED POSTOJEĆIH RJEŠENJA

Prema statistici iz srpnja 2024., na Google Play trgovini moglo se pronaći više od 1 600 000 aplikacija [4]. Distribucija aplikacija po kategorijama na Google Play trgovini dostupna je u prilogu [5]. Aplikacije za edukaciju zauzimaju svega 10,5% tržišta. U sljedećim je potpoglavljima prikazano pet rješenja, od kojih su 3 mobilne aplikacije, a dva rješenja su mrežne stranice. Svih pet rješenja sadrže zadatak višestrukog odabira. Objašnjene su njihove funkcionalnosti, prednosti i nedostatci.

2.1. English B2 FCE

English B2 FCE [6] je mobilna aplikacija koja je dostupna za iOS i Android uređaje. Izradila ju je američka tvrtka Shining Apps LLC u veljači 2024. godine. Osim aplikacije za FCE ispit, nudi se i pet drugih aplikacija namijenjenih za sličnu upotrebu, ali za različite razine znanja. Ima više od 10 000 preuzimanja i prosječnu ocjenu od 4,2.

Na početnoj stranici nalaze se zadaci nadopunjavanja teksta (eng. *Open Cloze*), višestrukog odabira (eng. *Multiple Choice*), oblikovanja riječi (eng. *Word Formation*), oblikovanja rečenice (eng. *Keyword Transformation*) i zadaci kreirani uz pomoć umjetne inteligencije (eng. *AI Exercises*). Svaka od navedenih kategorija sadrži više od 50 zadataka. Nakon rješavanja pojedinog zadatka dobije se povratna informacija u obliku bodova, postotka rješenosti i točnih rješenja. Dostupan je i sustav koji računa ukupan napredak korisnika u odnosu na sve moguće zadatke. Zadaci kreirani uz pomoć umjetne inteligencije nastali su zbog nedostatka relevantnih zadataka s preciznim rješenjima. Dakle, ova funkcionalnost omogućava korisniku neograničen broj novih zadataka koje je kreirala umjetna inteligencija.

Aplikacije ima i svoja ograničenja. Za neograničeno korištenje svih zadataka, potrebno je jednokratno platiti 12,99 $.

2.2. Exam Lift: English Practice

Exam Lift: English Practice [7] je mobilna aplikacija koja je dostupna za iOS i Android uređaje. Na Google Playu ima više od 500 000 preuzimanja i 17 300 recenzija s prosječnom ocjenom od 4,7.

Na prvom zaslonu korisnik odabire svoju razinu znanja, a ponuđene su mu osnovne razine (eng. *A2 Key for Schools*), srednje (eng. *B1 Preliminary for Schools*), naprednije (eng. *B2 First for Schools*), ostale (eng. *Other*) i opcija nepripremanja za ispit (eng. *I'm not preparing for an exam*). Nakon odabira razine prikazan je zaslon gdje se može započeti učenje. Na dnu zaslona prikazane su opcije moje putovanje (eng. *My Journey*), napredak (eng. *Progress*), moj profil (eng. *My Profile*) i trgovina (eng. *Shop*). Glavni je dio aplikacije moje putovanje. U ovom dijelu aplikacije rješavaju se zadaci. Na napretku se može pratiti rješavanje zadataka tijekom vremena, a na zaslonu moj profil mogu se mijenjati razine znanja. Svaka razina sadrži paket (eng. *Pack*) od četiriju pitanja. Ukoliko se ne posjeduje premium verzija, korisnik može rješavati jedan paket dnevno.

Exam Lift: English Practice je interaktivna aplikacija koja nudi rješavanje zadataka višestrukog odabira. Nedostatak je ove aplikacije obaveza plaćanja (4,99 $) kako bi se mogle iskoristiti sve funkcionalnosti.

2.3. The English Learning Lounge

The English Learning Lounge [8] je mobilna aplikacija koje je dostupna za iOS i Android uređaje. Na tržište ju je plasirao Neil Coghlan, a izradila ju je Anne Novas 2013. godine. Ima više od 10 000 preuzimanja i prosječnu ocjenu od 4,3.

Na početnom zaslonu prikazano je 12 opcija koje se kreću od First Words do TOEFL ispita. TOEFL je standardizirani ispit koji polažu studenti ukoliko žele studirati u Sjedinjenim Američkim Državama. Nakon odabira opcije, korisnik može odabrati Use of English ispit i pripadajuće zadatke. Na kraju rješavanja svakog zadatka dostupna je statistika za pripadajući zadatak, ali i za cjelinu i cijelu razinu. Ova funkcionalnost može biti korisna jer se i na standardiziranom ispitu moraju postići određeni postotci, kako za svaki zadatak pojedinačno, tako i za svaku cjelinu posebno. Nakon prikaza rezultata, moguće je klinuti na gumb Review koji prikazuje točne rezultate u cijelom ispitu. U ostalim ispitima nalazi se detaljan prikaz gramatike (eng. *Use of English*) te po jedno pojedinačno poglavlje za čitanje (eng. *Reading*), slušanje (eng. *Listening*) i govorenje (eng. *Speaking*). Svako od navedenih poglavlja sadrži zadatke višestrukog odabira.

Pristup ostalim zadacima dodatno se naplaćuje. Aplikacija je posljednji put nadograđena 2017. godine. Funkcionalnosti i namjena aplikacije nisu se godinama mijenjali, a dizajn joj je zastario, što su nedostaci navedene aplikacije.

2.4. Esl-lounge

Esl-lounge [9] je mrežna stranica Neila Coghlana. Omogućuje tri različita načina korištenja: kao student, učitelj i premium korisnik. U ovom poglavlju opisat će se korištenje iz perspektive studenta.

S lijeve strane na početnoj stranici nalazi se izbornik. Može se izabrati jedno od brojnih poglavlja poput uvoda u gramatiku (eng. *Grammar Guide*), zadataka iz gramatike (eng. *Grammar Exercises*), zadataka iz čitanja s razumijevanjem (eng. *Reading Exercises*), zadataka iz slušanja s razumijevanjem (eng. *Listening Exercises*), zadataka iz vokabulara (eng. *Vocabulary Exercises*), pripreme za ispit (eng. *Test Prep*) i frazalnih glagola (eng. *Phrasal Verbs*). Ista funkcionalnost ostvaruje se i na glavnom dijelu zaslona uz dodatnu mogućnost izravnog biranja razine. Prva je razina početnik (eng. *Beginner*), a posljednja napredni korisnik (eng. *Advanced*). Unutar poglavlja pripreme za ispit može se izabrati FCE. Unutar FCE može se izabrati Reading gdje se dolazi do zadatka višestrukog odabira. Na prvoj stranici nalazi se pregled (eng. *Overview*), savjeti (eng. *Tips*) i zadaci za vježbu (eng. *Practice Exercises*). Klikom na zadatke za vježbu otvara se nova stranica koja nudi 18 različitih zadataka. Nakon rješavanja zadatka, dobije se prikaz o broju ostvarenih bodova, ocjena i prikaz rješenja svakog zadatka pojedinačno.

Esl-lounge je mrežna stranica koja nudi velik broj zadataka i objašnjenja u obliku videozapisa. Učenici/studenti mogu bez ograničenja besplatno koristiti sav dostupan sadržaj. Nedostatak je ove mrežne stranice zastarjeli dizajn koji doprinosi neorganiziranosti stranice.

2.5. Flo-Joe

Flo-Joe je mrežna stranica kojoj je cilj pripremanje korisnika za polaganje Cambridge ispite poput FCE (eng. *First Certificate in English*), CAE (eng. *Certificate in Advanced English*) i CPE (eng. *Certificate of Proficiency in English*). Na stranici se može pristupiti različitim vježbama iz područja čitanja (eng. *Reading*), pisanja (eng. *Writing*), slušanja (eng. *Listening*) i uporabe engleskog jezika (eng. *Use of English*). Ovo mrežno rješenje sadrži zadatke višestrukog odabira. Prilikom rješavanja, nakon svakog odabranog odgovora, dobije se povratna informacija o (ne)točnosti odgovora. Ispod teksta zadatka nalaze se savjeti koji su usko vezani za svaki pojedini zadatak. Na dnu stranice nalaze se dodatni materijali.

Flo-Joe je odlično mrežno rješenje jer se svi potrebni materijali, zadaci i savjeti nalaze na jednom mjestu. Dodatni materijali pružaju korisniku mogućnost napretka. Nedostatak ove mrežne stranice je manjak zadataka za pojedino područje.

2.6. Osvrt na postojeća rješenja

Pregledom postojećih rješenja može se zaključiti da analizirane aplikacije imaju vrlo slične funkcionalnosti. English B2 FCE, Exam Lift: English Practice i The English Learning Lounge su aplikacije koje imaju zadatke iz čitanja s razumijevanjem, koriste sustav bodovanja i prate napredak korisnika. Glavna im je prednost prilagođeni pristup različitim razinama znanja, što ih čini izvrsnim alatom za učenje.

Međutim, većina aplikacija ima ograničenja u osnovnoj verziji, koja se mora nadoplatiti kako bi se ostvario puni pristup funkcionalnostima. S druge strane, mrežne stranice nude sav sadržaj besplatno, što ih čini privlačnima za korisnike koji ne žele plaćati aplikacije.

Zajednički nedostatak svih aplikacija i mrežnih stranica, osim Exam Lift: English Practice, je dizajn i manjak inovativnih ideja. Naime, sva pregledana rješenja svojevremeno su bila korektna, no nisu dograđivana, a inovacija i kreativnost zadržavaju postojeće i privlače nove korisnike. English B2 FCE taj je problem riješio zadacima koje generira umjetna inteligencija.

3. KORIŠTENE TEHNOLOGIJE I PROGRAMSKA ARHITEKTURA

3.1. Operacijski sustav Android

Android je najpopularniji operacijski sustav za mobilne uređaje. Prema istraživanju iz prvog kvartala 2024. godine, taj postotak iznosi 70,71 % [10]. Zbog široke raspostranjenosti Android je idealan za razvoj mobilnih aplikacija. Verzija Androida korištena u ovom radu je 14.

3.2. Razvojno okruženje Android Studio

Android Studio je službeno razvojno okruženje za Android aplikacije. Nudi skup alata za razvoj, testiranje i optimizaciju aplikacije. Integriran je s alatima za verzioniranje koda (npr. Github), Firebase bazom podataka i emulatorom. U ovom radu korištene su sve 3 navedene funkcionalnosti.

3.3. Programski jezik Kotlin

Kotlin je modern programski jezik koji je podržan za razvoj Android aplikacija. Zbog svoje jednostavne sintakse i mogućnosti istovremenog rada s Javom, popularnost Kotlina neprestano raste. Kotlin omogućava smanjenje broja linija koda i brži razvoj mobilnih aplikacija u odnosu na prijašnja rješenja. U ovom radu korištena je Kotlin verzija 1.9.22.

3.4. Jetpack Compose

Jetpack Compose je alat za izradu korisničkih sučelja na Androidu. Nasljednik je XML formata. Ubrzava, pojednostavljuje i nudi intuitivniji pristup za razvoj vizualno privačnih rješenja uz pisanje manje koda. U ovom radu korištene su brojne funkcionalnosti Jetpack Composea, ali najveći je naglasak bio na osnovnim strukturama poput Box, Column i Row čijim se kombinacijama mogu stvoriti moderna i atraktivna rješenja.

3.5. Firebase

Firebase je Googleova platforma koja nudi usluge poput autentifikacije, pohrane podataka i analitike u stvarnom vremenu. U razvoju Android aplikacija popularan je odabir zbog integriranosti u Android Studio. U ovom radu korištena je Firebase Firestore NoSQL baza podataka za pohranu zadataka.

4. RAZVOJ APLIKACIJE I PRIKAZ FUNKCIONALNOSTI

4.1. Početni zaslon

Početni je zaslon vrlo jednostavan. Na njemu se nalazi slika knjige i gumb koji vodi na sljedeću stranicu na kojoj se nalazi uvod, odnosno pregled teorijskih područja.

Slika 4.1. Početni zaslon aplikacije

Knjiga je tamno crvena pa je u dodatoteku colors.xml dodana takva nijansa crvene boje (burgundy).

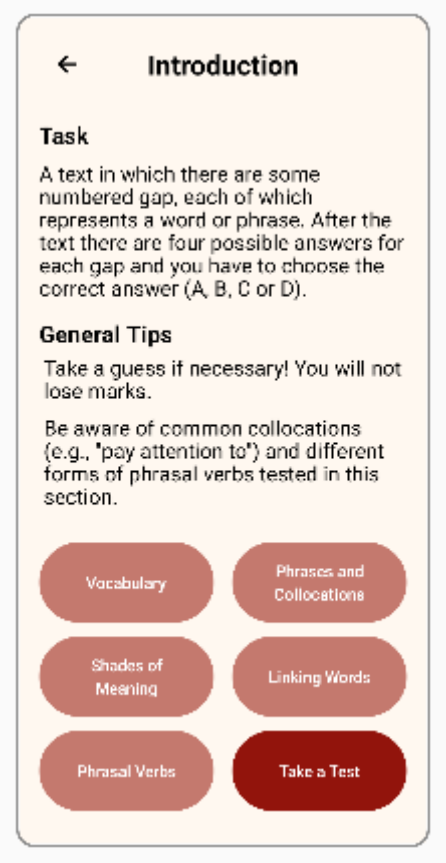
colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="dark_grey">#1E1E24</color>
    <color name="burgundy">#92140C</color>
    <color name="dark_burgundy">#580804</color>
    <color name="light_burgundy">#c4796e</color>
    <color name="beige">#FFF8F0</color>

</resources>
```

Slika 4.2. Prikaz colors.xml

4.2. Uvodni zaslon

Osnovna je ideja uvodnog zaslona pružiti korisniku teorijsku podlogu prije rješavanja zadatka. Analizom postojećih rješenja primijećeno je da se ova funkcionalnost ne nudi u besplatno pristupu ili uopće. Uvodni zaslon (*IntroductionScreen.kt*) sadrži naslov, gumb za odlazak na prethodni zaslon, opis glavnog zadatka i dva savjeta. Ispod savjeta nalazi se 6 gumbova koji se odnose na područja vokabulara (eng. *Vocabulary*), fraza i kolokacija (eng. *Phrases and Collocations*), razlika u značenju (eng. *Shades of Meaning*), poveznica (eng. *Linking Words*), frazalnih glagola (eng. *Phrasal Verbs*) i pokretanja testa (eng. *Take a Test*). Na slici 4.3. dostupan je prikaz uvodnog zaslona, a u nastavku se nalazi objašnjenje svakog dijela i način njegove implementacije.



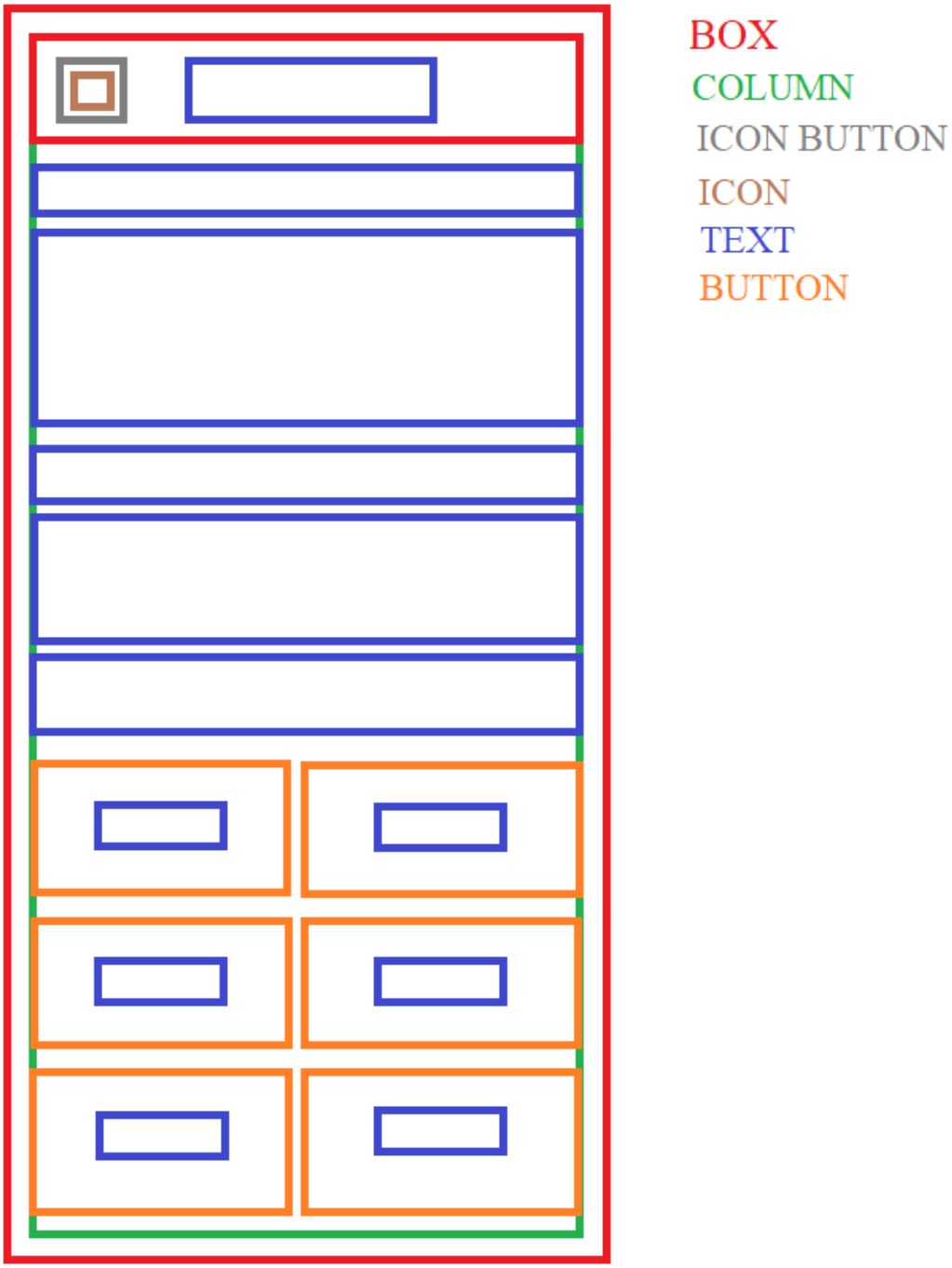
Slika 4.3. Prikaz uvodnog zaslona

4.2.1. Struktura uvodnog zaslona

Za strukturu slike 4.3. zaslužna je raspodjela elemenata. Raspodjelu elemenata obavlja Jetpack Compose. U prilogu [11] se može vidjeti više o osnovnim elementima koji su korišteni za izradu ovakvog rasporeda. Elementi koji su korišteni su Box, Column i Row. Osim njih, korištene su i funkcije IconButton, Icon, Button i Text. Svaki zahtijeva određenu tehniku stiliziranja, o čemu će

više biti riječi u poglavlju 4.2.4. Ovdje se želi staviti naglasak staviti na kostur korisničkog sučelja koji radi kako je zamišljeno neovisno o podacima koji su mu predani.

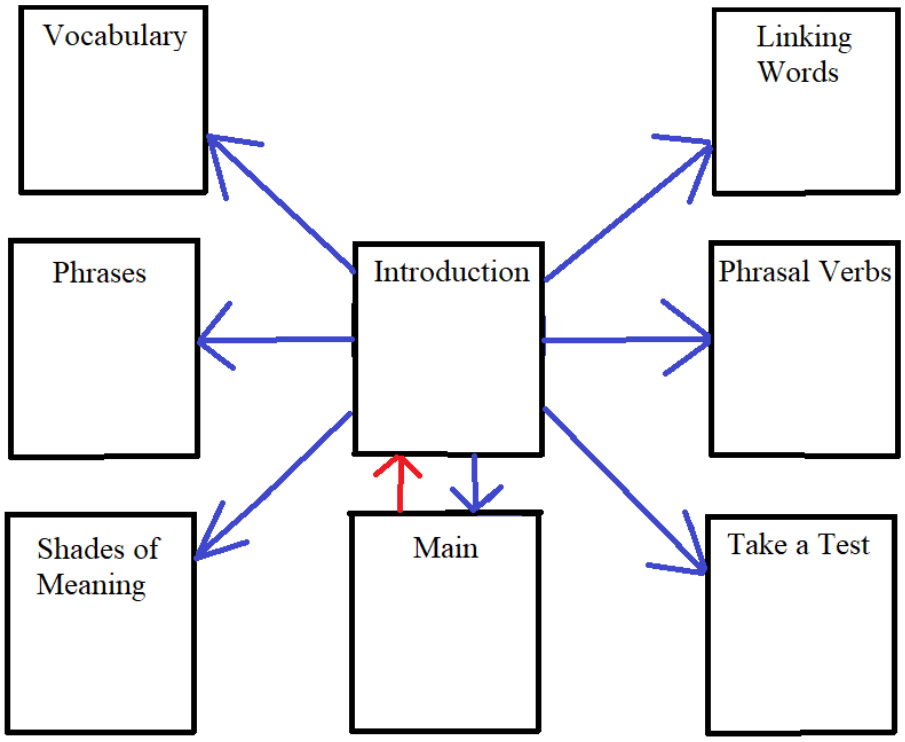
Prikaz korisničkog sučelja u pozadini, s pripadajućom legendom i objašnjenjima, nalazi se na slici 4.4.



Slika 4.4. Prikaz korisničko sučelja uvodnog zaslona

4.2.2. Navigacija uvodnog zaslona

Vrlo je važno razumjeti što će se dogoditi kada se klikne na određeni gumb. Za ovu funkcionalnost zaslužna je navigacija. Navigacija predstavlja skup funkcija koje su zaslužne za kretanje između zaslona. Kako je vidljivo u poglavljima 4.2. i 4.2.1., na uvodnom zaslonu postoji 6 gumbova i 1 slika (eng. *Icon*). Klikom na sliku strelice unazad korisnik je vraćen na počeni zaslon. Ovo se ostvaruje jednostavnom funkcijom onNavigateBack(). Ostalo su gumbovi. Svaki gumb vodi na zaslon na kojoj se nalazi detaljniji prikaz te teme. Dakle, klikom na Vocabulary gumb korisnik je odveden na Vocabulary zaslon, gdje može dobiti teorijsku podlogu iz područja vokabulara. Ova funkcionalnost implementirana je za svaki gumb na uvodnom zaslonu. Gumb Take a Test izravno započinje pokretanje zadatka. Prikaz navigacije, odnosno do kojeg se sve zaslona može doći s uvodnog zaslona nalazi se na slici 4.5.



Slika 4.5. Prikaz navigacije uvodnog zaslona

4.2.3. Rad s bazom podataka vezanih uz uvodni zaslon

Osim navigacije i raspodjele elemenata, ključan dio za ispravan rad uvodnog zaslona je povezivanje s bazom podataka. Povezivanje s bazom podataka je važno zato što omogućuje dodavanje novih tekstova bez izravnog mijenjanja koda čime bi se poštuje OCP (eng. *Open Closed Principle*). OCP kaže da klasa treba biti otvorena za proširenja, a zatvorena za promjenu. U ovom kontekstu to je ispunjeno. Ovakav pristup omogućio je ViewModel. ViewModel je klasa koja izlaže korisničko sučelje, a enkapsulira (sakriva) kako se to odvija iznutra[12]. Drugim riječima, korisnik dobiva prikaz podataka koji su njemu relevantni bez da dobiva uvid u način na koji je to učinjeno o čemu se brine ViewModel.

4.2.3.1. ViewModel uvodnog zaslona

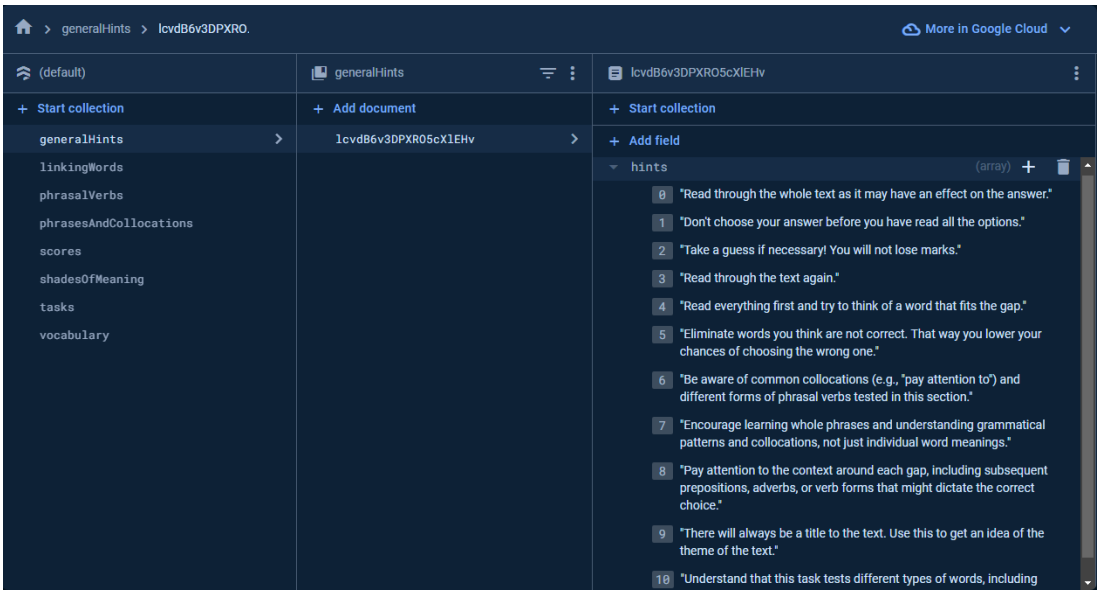
Na slici 4.6. nalazi se funkcija ViewModela koja se brine o prikazu savjeta (eng. *Hints*). Ova funkcija zove se loadGeneralHints. Kao argument predaje joj se taskId, niz znakova koji jednoznačno određuje kolekciju. Nakon toga funkcija traži kolekciju imena generalHints. Nakon što je pronašla kolekciju, prolazi kroz nju te iz svakog polja izvlači vrijednost i sprema ga u varijablu _generalHints.value.

```
Luka.Agicic
88  fun loadGeneralHints(taskId: String) {
89      db.collection( collectionPath: "generalHints").document(taskId)
90          .addSnapshotListener { documentSnapshot, e ->
91              if (e != null) {
92                  return@addSnapshotListener
93              }
94
95              if (documentSnapshot != null && documentSnapshot.exists()) {
96                  val hints = documentSnapshot.get("hints") as? List<String> ?: emptyList()
97                  _generalHints.value = hints
98              }
99          }
100 }
101
```

Slika 4.6. Prikaz funkcije loadGeneralHints u ViewModel-u

4.2.3.2. Firebase Firestore uvodnog zaslona

Ako su vrste podataka valjane, nazivi dobro napisani i taskId se poklapa s taskId iz baze, zadatak će moći povući savjete iz baze podataka. Na slici 4.7. nalazi se prikaz iz baze podataka. Prvo je prikazano ime kolekcije, pa njezin identifikator(taskId) i konačno puno nizova znakova (stringova) koji u ovom slučaju predstavljaju savjete.



Slika 4.7. Prikaz kolekcije generalHints

4.2.3.3. Savjeti na uvodnom zaslonu

Isprobano je više različitih brojeva, ali prikaz dvaju savjeta po uvodnom zaslonu pokazao se najboljim. Ovaj se broj može mijenjati predavanjem drukčijeg parametra funkciji. Međutim, drukčiji broj parametara nije svrsishodan iz dvaju razloga. Prvo, savjeti trebaju biti kratki i pomoći korisniku u datom trenutku, dakle više bi savjeta zahtijevalo više čitanja i trošenja energije na nešto što nije riješavanje zadatka. Drugi je razlog možda i važniji, a to je da se promjenom broja savjeta mijenja i korisničko sučelje. Na slici 4.8. može se vidjeti način na koji se odabire broj savjeta prikazanih po zaslonu.

```
val randomHints = remember(hints) {
    hints.shuffled().take( n: 2)
}
```

Slika 4.8. Prikaz izračuna broja savjeta

4.2.4. Funkcionalnosti Jetpack Compose-a uvodnog zaslona

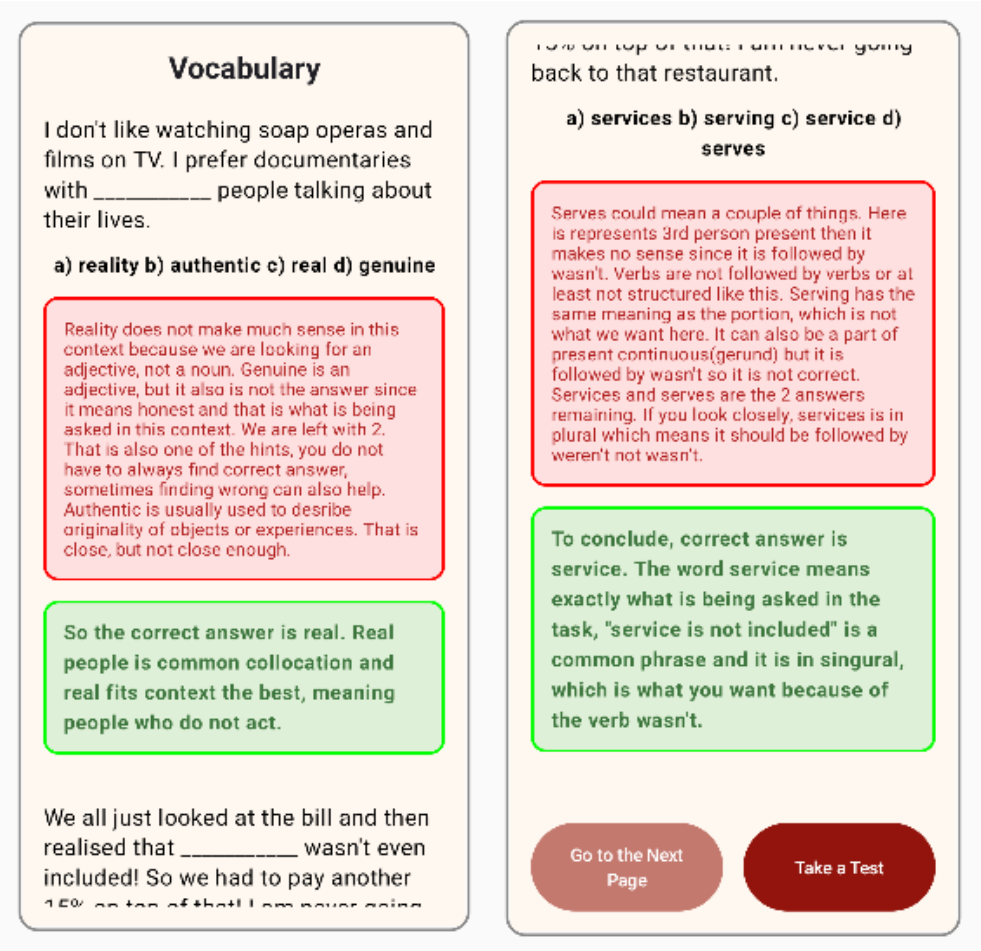
Na početku poglavlja u uvodnom zaslonu, točnije u poglavlju 4.2.1., spomenuto je stiliziranje. Svaki od elemenata Box, Column i Row ima identičnu sintaksu u obliku ime funkcije () {}. Uzet ćemo za primjer funkciju Box. Unutar običnih zagrada poziva se modifier koji omogućava stiliziranje gradivne (eng. *Composable*) funkcije [13]. Ova funkcionalnost znači da modifier ima jako veliki broj opcija, poput margine ili postavljanja okvira (eng. *Border*). Korištenjem operatora . poziva se svaki od unaprijed gotovih modifier funkcija. Dakle, za uređivanje s Jetpack Composeom ne moraju se pisati funkcije, već ih je dovoljno samo pozvati. Primjer korištenja Jetpack Composea može se vidjeti na slici 4.9. Korištene su funkcije border i background kako bi vizualno odvojili Box od ostatka zaslona. Funkcija border boja rub Box-a u sivo i čini rubove zaobljenima. Funkcija background postavlja boju Box-a u bež. Ostatak elemenata zaslužan je za raspodjelu kao u poglavlju 4.2.1. Dodatno, korištene su funkcije za pozicioniranje u sredinu(horizontalAlignment = Alignment.CenterHorizontally) te paramteri za uređivanje teksta,u ovom slučaju veličina fonta (fontSize = 24.sp) i podebljavanje slova (fontWeight = FontWeight.Bold).

```
Box(
    modifier = Modifier
        .fillMaxSize()
        .padding(16.dp)
        .border(
            BorderStroke(2.dp, Color.Gray),
            shape = RoundedCornerShape(16.dp)
        )
        .padding(4.dp)
        .background(color = colorResource(id = R.color.beige))
) { this: BoxScope
    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp)
            .verticalScroll(scrollState),
        horizontalAlignment = Alignment.CenterHorizontally
    ) { this: ColumnScope
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .padding(bottom = 8.dp),
            contentAlignment = Alignment.Center
        ) { this: BoxScope
            IconButton(
                onClick = onNavigateBack,
                modifier = Modifier.align(Alignment.CenterStart)
            ) {
                Icon(Icons.Filled.ArrowBack, contentDescription = "Go back")
            }
            Text(
                text = "Introduction",
                style = MaterialTheme.typography.titleSmall.copy(
                    fontSize = 24.sp,
                    fontWeight = FontWeight.Bold
                ),
                textAlign = TextAlign.Center,
                modifier = Modifier.align(Alignment.Center)
```

Slika 4.9. Prikaz funkcionalnosti Jetpack Compose-a

4.3. Zaslon posvećen pojedinom području

Nakon što je korisnik pročitao opis zadatka i dva savjeta, može kliknuti na jedan od pet zaslona za pojedino područje. Zasloni za pojedino područje su slični, razlika je u sadržaju koji opisuju. Kako ne bi došlo do ponavljanja, fokus će biti stavljen na zaslonu vokabular. Na slici 4.10. nalazi se prikaz zaslona vokabular. Ovaj zaslon nema savjete jer sadrži objašnjenje rješenja pa nema smisla da se stvari ponavljaju.

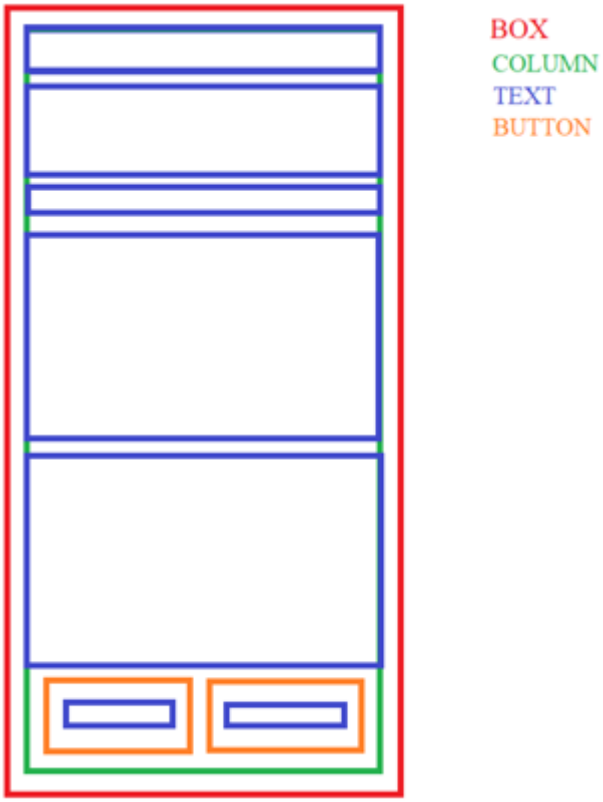


Slika 4.10. Prikaz zaslona vokabular

4.3.1. Struktura zaslona posvećenog pojedinom području

Strktura zaslona vokabular vrlo je jednostavna. Sastoji se od niza dijelova za tekst i 2 gumba na dnu zaslona. Tekst se povlači iz base, što je objašnjeno u poglavlju 4.3.3. Funkcionalnost gumba objašnjena je u poglavlju 4.3.2. Na slici 4.11. nalazi se prikaz strukture korisničko sučelja zaslona vokabular. Ipak, u stvarnosti se povlače dva zadatka, ali je zbog jednostavnosti prikaza na slici 4.11. prikazano kao da se povlači jedan zadatak. Mogućnost prikaza više zadataka na jednom

zaslonu omogućuje funkcija rememberScrollState() koja nudi funkcionalnost klizanja(eng. Scroll).



Slika 4.11. Prikaz korisničkog sučelja zaslona vokabular

4.3.2. Navigacija zaslona posvećenog pojedinom području

Za razliku od navigacije prikazane u poglavlju 4.2.2. navigacija zaslona vokabular puno je jednostavnija. Pomoću prvog gumba na dnu zaslona moguće je otići na sljedeći zaslon(u ovom slučaju Phrases and Collocations), dok je pomoću drugog moguće pokrenuti ispit. Ovakav pristup osigurava da je korisniku omogućen prolazak kroz cijelu teorijsku podlogu ili kroz samo jedno poglavlje, ukoliko misli da u njemu može ostvariti napredak. Na slici 4.11. nalazi se prikaz opisane funkcionalnosti. Dakle, klikom na prvi gumb korisnik je odveden do zaslona Phrases and Collocations(pomoću funkcije onNavigateToPhrases), a klikom na drugi gumb odveden je do zaslona TaskOneScreen(pomoću funkcije onNavigateToTaskOne).

Slika 4.12. Prikaz navigacije pomoću gumbova

4.3.3. Rad s bazom podataka zaslona posvećenog pojedinom području

Baza podataka za ovaj zaslon zadužena je za prikaz teksta zadatka, rješenja, objašnjenja i objašnjenja točnog zadatka. U poglavlju 4.2.3. je objašnjeno zašto se primjenjuje ovakav pristup i razlog ostaje isti i na ovom zaslonu.

4.3.3.1. ViewModel zaslona posvećenog pojedinom području

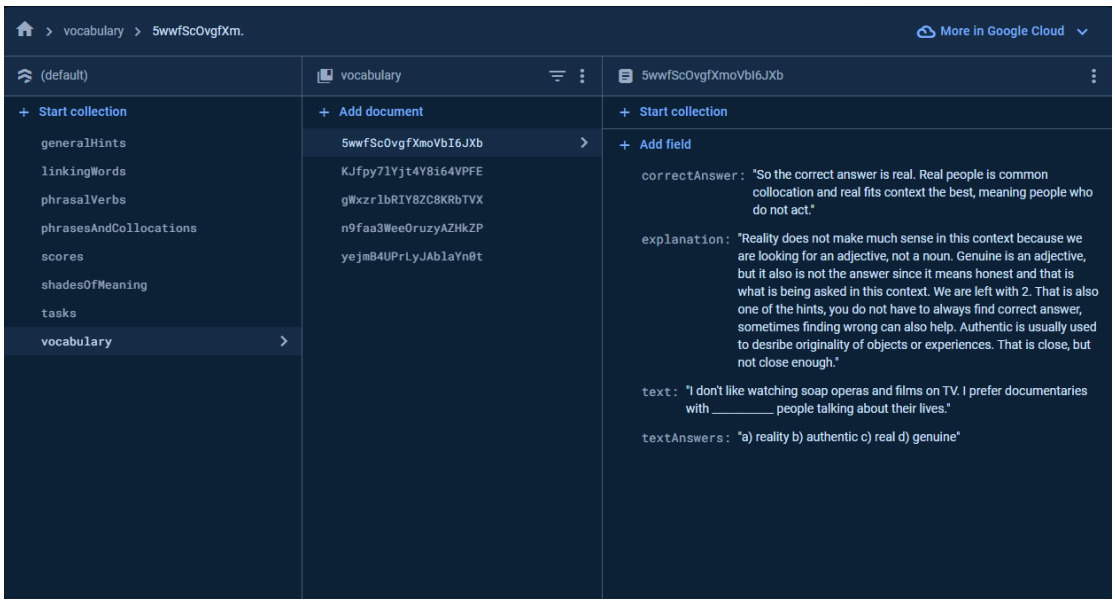
Na slici 4.13. nalazi se funkcija loadVocabularyData(). Ova funkcija pronalazi kolekciju vocabulary, u njoj traži nizove znakova i sprema ih u varijablu _vocabularyData. Također, ova funkcija se brine kako bi se prikazala samo dva dokumenta po zaslonu. Ovo je ostvareno uz pomoć funkcije shuffled(). Funkcija shuffled() radi nasumičan odabir i prikazuje n odabira, u ovom slučaju n je jednako 2.

```
Luka.Agicic *
fun loadVocabularyData() {
    db.collection( collectionPath: "vocabulary")
        .addSnapshotListener { querySnapshot, e ->
            if (e != null) {
                return@addSnapshotListener
            }

            if (querySnapshot != null && !querySnapshot.isEmpty) {
                val data = querySnapshot.documents.mapNotNull { documentSnapshot ->
                    documentSnapshot.data?.mapValues { it.value.toString() }
                }

                val selectedDocuments = if (data.size > 2) {
                    data.shuffled().take( n: 2)
                } else {
                    data
                }
                _vocabularyData.value = selectedDocuments
            }
        }
}
```

Slika 4.13. Prikaz funkcije loadVocabularyData u ViewModel-u

4.3.3.2. Firebase Firestore zaslona posvećenog pojedinom području

Na slici 4.14. nalazi se prikaz baze podataka. Kolekcija se zove vocabulary, identifikator je nasumično generiran, a imena polja su text, textAnswers, explanation i correctAnswers.



Slika 4.14. Prikaz kolekcije vocabulary

4.3.4. Funkcionalnosti Jetpack Compose-a vezanih uz zaslon posvećen pojedinom području

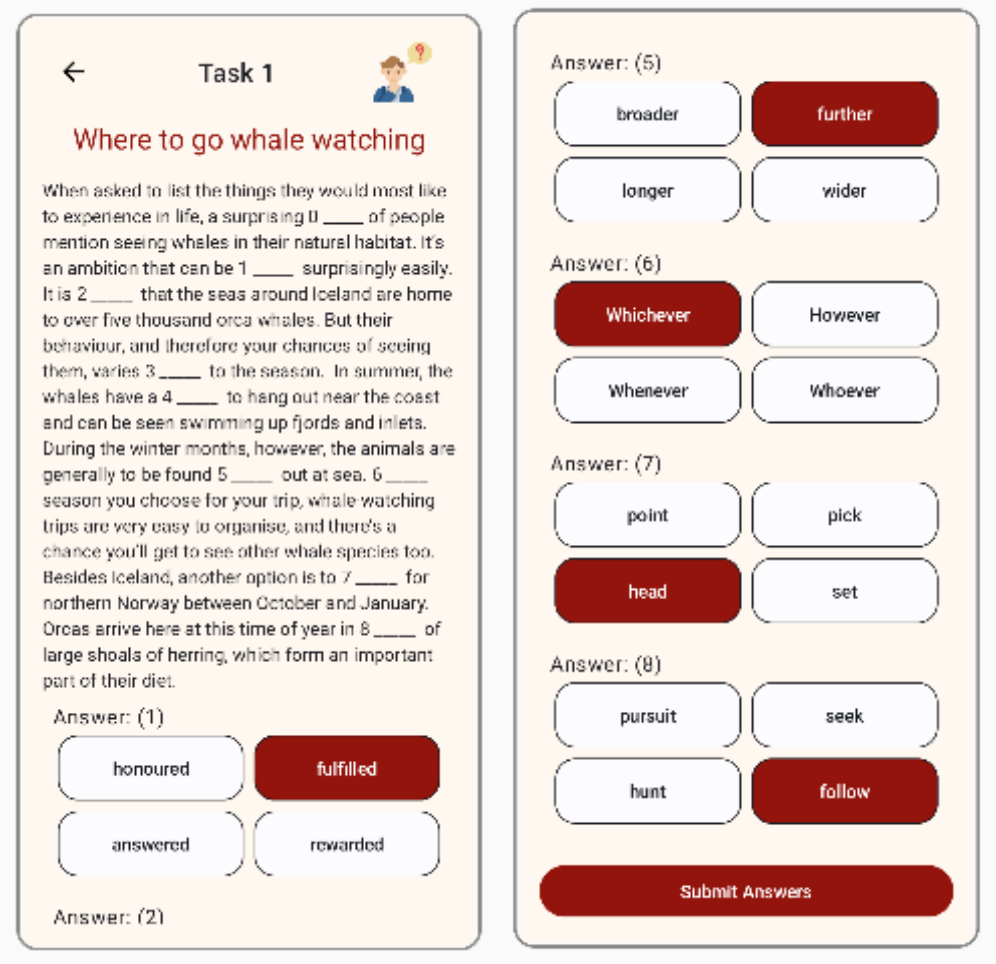
Osim funkcionalnosti spomenutih u poglavlju 4.2.4. zaslonu vokabulara se mogu opisati i dvije nove. Prva je prikaz u zelenoj, odnosno crvenoj kutiji, ovisno kojeg je naziva poruka. Poruka naziva explanation obojana je crveno dok je poruka naziva correctAnswer obojana zeleno. Na slici 4.15. prikazana je implementacija zeleno obojane kutije. Implementacija za crvenu je identična, samo se za funkciju Color preda druga heksadicemalna vrijednost. Osim bojanja, na slici je prisutno postavljanje obruba kao i mijenjanje svojstava slova poput veličine, boje i centriranosti.

```
Box(
    modifier = Modifier
        .fillMaxWidth()
        .padding(vertical = 8.dp)
        .border(
            border = BorderStroke(2.dp, Color.Green),
            shape = RoundedCornerShape(12.dp)
        )
        .background(Color( color: 0xFFDFF0D8), shape = RoundedCornerShape(12.dp))
        .padding(16.dp)
) { this: BoxScope
    Text(
        text = document["correctAnswer"] ?: "No correct answer available",
        style = MaterialTheme.typography.bodyLarge.copy(
            fontSize = 16.sp,
            fontWeight = FontWeight.Bold,
            color = Color( color: 0xFF3C763D)
        ),
        textAlign = TextAlign.Start
    )
}
```

Slika 4.15. Prikaz implementacije zelene kutije

4.4. Zaslon ispita

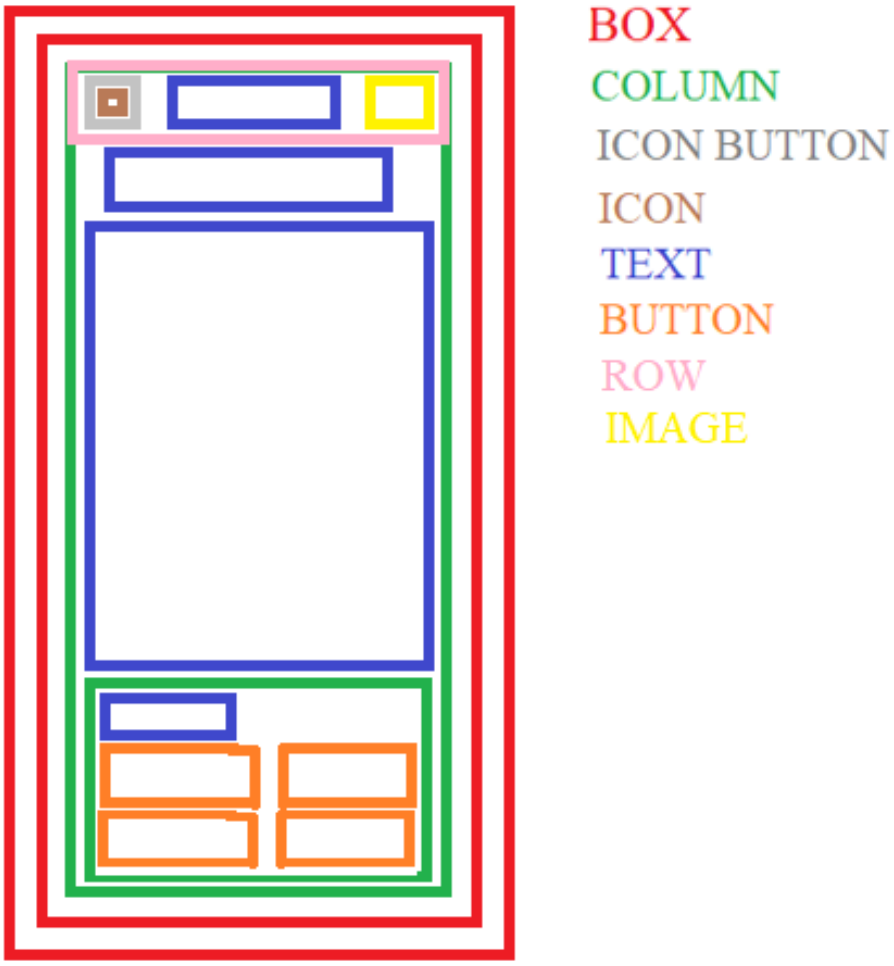
Nakon što je prošao kroz zaslone posvećene pojedinom području, korisnik je spreman za rješavanje ispita. Ispit će pronaći na zaslonu taskOneScreen. Na ovom zaslonu može se vidjeti gumb za odlazak unazad (čija se funkcionalnost ostvaruje pomoću onNavigateBack() funkcije). Pored gumba nalazi se tekst Task1 i slika dječaka koja predstavlja sustav savjeta (eng. Hints). Ispod navedenog nalazi se naslov teme te tekst zadatka. Nakon teksta zadatka nalazi se četiri gumba koji predstavljaju zadatak višestrukog odabira. Moguće je kliknuti na jedan od ta četiri gumba. Budući da je u tekstu 8 praznina, tako i ima 8 redova gumbova. Dakle, 32 gumba ukupno. Na dnu stranice nalazi se gumb Submit Answers. Klikom na njega korisnik predaje odgovore i odveden je na zaslon rezultata (eng. resultsScreen). Na slici 4.16 se mogu vidjeti sve navedene funkcionalnosti.



Slika 4.16. Prikaz zaslona ispit

4.4.1. Struktura zaslona ispita

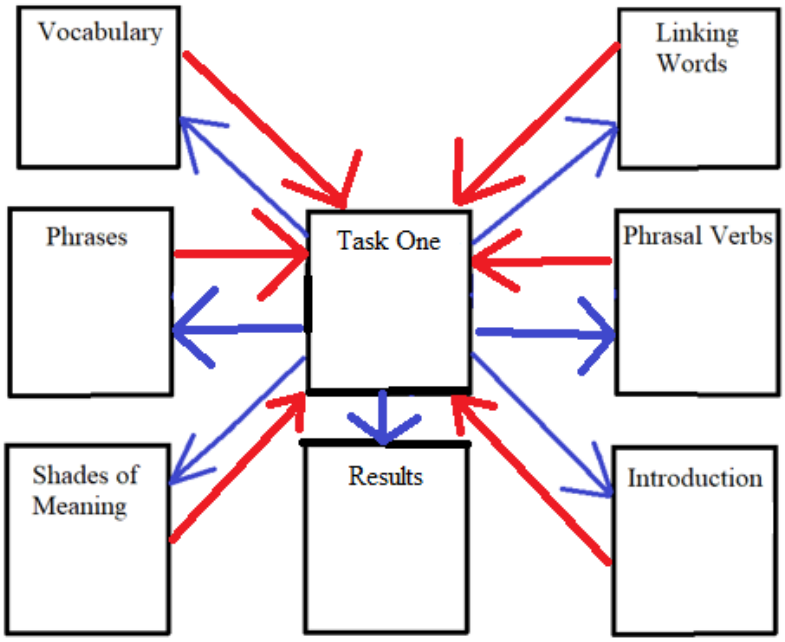
Struktura ispitnog zaslona najsloženija je od svih zaslona. Na njemu se nalazi 8 različitih elemenata. Popis elemenata nalazi se u legendi na slici 4.17. Na istoj slici se može se vidjeti da velik dio zaslona zauzima tekst. Osim teksta na slici se nalaze gumbi koji predstavljaju višestruki odabir. Više o tome kako oni rade može se pročitati u poglavlju 4.4.5. Dodatno, na slici se može primjeti slika. Klikom na ovu sliku otvaraju se savjeti. Više o ovoj funkcionalnosti može se pročitati u poglavlju 4.4.3.3. Na slici je prikazan jedan zadatak višestrukog odabira. Za prikaz više zadataka kao u poglavlju 4.4. korištena je mogućnost klizanja.



Slika 4.17. Prikaz korisničkog sučelja ispitnog zaslona

4.4.2. Navigacija zaslona ispita

Budući da je ostavljeno korisniku na izbor želi li prolaziti kroz zaslone teorijske podloge, navigacija ispitnog ekrana relativno je složena. Sa svakog zaslona se može doći do ispitnog ekrana. S ispitnog ekrana funkcijom povlačenje (eng. Swipe) moguće se vratiti na ekran s kojeg se krenulo. Osim toga, gumb na dnu stranice ispitnog zaslona omogućuje predaju zadatka i odlazak na zaslon rezultata. Navedene mogućnosti navigacije prikazane su na slici 4.18.



Slika 4.18. Prikaz navigacije ispitnog zaslona

4.4.3. Rad s bazom podataka zaslona ispita

Ispitni zaslon iz baze podataka povlači jako puno podataka. Za ispravan rad ispitnog zaslona iz baze podataka potrebno je povući točne odgovore (eng. correctAnswers), opciju za odabir (eng. dropdownOptions), tekst zadataka (eng. text) i naslov zadatka (eng. title). Za pohranjivanje opcija za odabir potrebno je koristiti kompleksne strukture podataka. Više o njihovom korištenju u prilogu. [14]

4.4.3.1. ViewModel zaslona ispita

Ispitni zaslon jedini je zaslon u kojem se koriste dvije funkcije ViewModela. Prva funkcija koju koristi je loadGeneralHints. Više o ovoj funkciji i njenoj ulozi može se pročitati u poglavlju 4.4.3.3. Druga funkcija koju koristi ispitni zaslon je loadRandomTask. Ova funkcija ima nekoliko zadaća i njezin prikaz se nalazi na slici 4.19. Prva funkcionalnost je generiranje nasumičnog zadatka pomoću funkcije shuffled(). Dodatno, korištena je klasa Log. Više o ovoj klasi može se pročitati u prilogu. [15] U ovom zadatku klasa Log je korištena kako bi se prikazala poruka ukoliko zadatak nije dobro učitan.

```
fun loadRandomTask() {
    db.collection( collectionPath: "tasks") CollectionReference
        .get() Task<QuerySnapshot!>
        .addOnSuccessListener { querySnapshot ->
            if (querySnapshot != null && !querySnapshot.isEmpty) {
                val randomDocument = querySnapshot.documents.shuffled().first()
                _currentTaskId.value = randomDocument.id
                Log.d( tag: "TaskViewModel",  msg: "Random task loaded with ID: ${randomDocument.id}")
                loadTask(randomDocument.id)
            }
        }
        .addOnFailureListener { it: Exception
            Log.e( tag: "TaskViewModel",  msg: "Failed to load tasks", it)
        }
}
```

Slika 4.19. Prikaz loadRandomTask funkcije

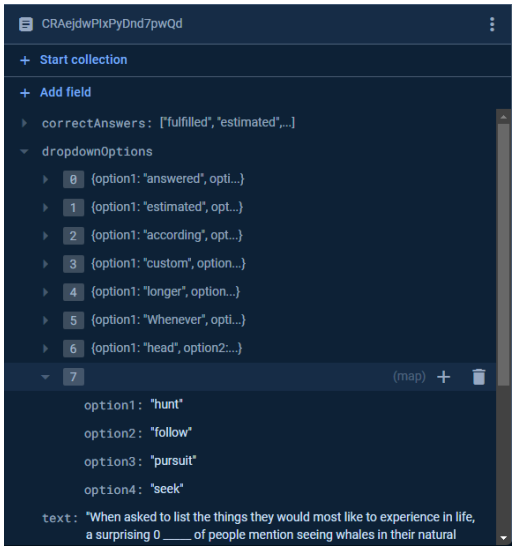
Na slici 4.19. nalazi se prikaz funkcije loadRandomTask. Unutar njezinog tijela poziva se funkcija loadTask. Funkcija loadTask zadužena je za pozivanje zadatka nakon što je nasumično određeno koji će to zadatak biti. Za ovu funkcionalnost funkcija loadTask zahtjeva predaju identifikatora kako bi znala o kojem se točno dokumentu radi. Nakon što je dobio identifikator, loadTask iterira kroz podatke, provjera postoje li i onda ako postoje ih dohvaća. Kod je malo složeniji jer se radi sa složenim strukturama podataka, pa treba voditi pažnje o više stvari nego kod jednostavnih struktura. Prikaz koda čija je funkcionalnost opisana u nekoliko pretohodnih rečenica nalazi se na slici 4.20.

```
Luka.Agicic *
fun loadTask(taskId: String) {
    db.collection( collectionPath: "tasks").document(taskId)
        .addSnapshotListener { documentSnapshot, e ->
            if (e != null) {
                return@addSnapshotListener
            }

            if (documentSnapshot != null && documentSnapshot.exists()) {
                val text = documentSnapshot.getString( field: "text") ?: ""
                val hints = documentSnapshot.get("hints") as? List<String> ?: emptyList()

                val title = documentSnapshot.getString( field: "title") ?: "No Title"

                val dropdownOptionsMapList = documentSnapshot.get("dropdownOptions")
                val dropdownOptions = (dropdownOptionsMapList as? List<Map<String, String>>)?.map { map ->
                    map.values.toList()
                } ?: emptyList()

                val correctAnswers = documentSnapshot.get("correctAnswers") as? List<String> ?: emptyList()
                _taskState.value = Task(text, hints, dropdownOptions, correctAnswers, title)
                _correctAnswers.value = correctAnswers
            }
        }
}
```

Slika 4.20. Prikaz funkcije loadTask

4.4.3.2. Firebase Firestore zaslona ispita

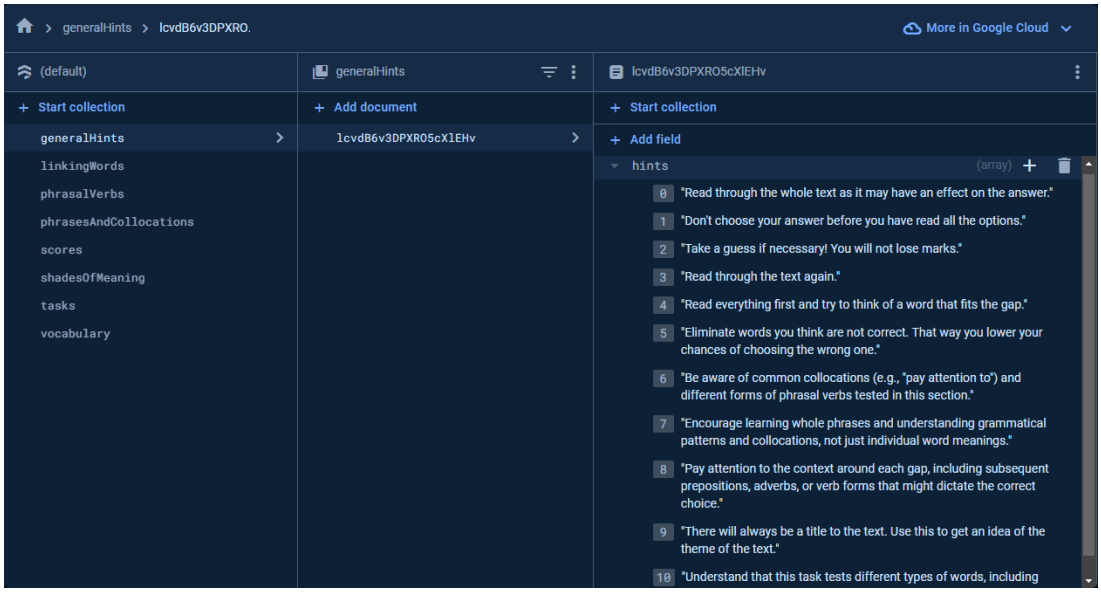
Prikaz polja za točne odgovore, opciju za odabir, tekst zadataka (na prikazu nedostaje naslov jer fizički ne stane na ekran, ali je prisutan u bazi podataka). Svi ovi podaci se povlače dinamično i prikazuju na ispitnom zaslonu.



Slika 4.21. Prikaz polja u kolekciji tasks

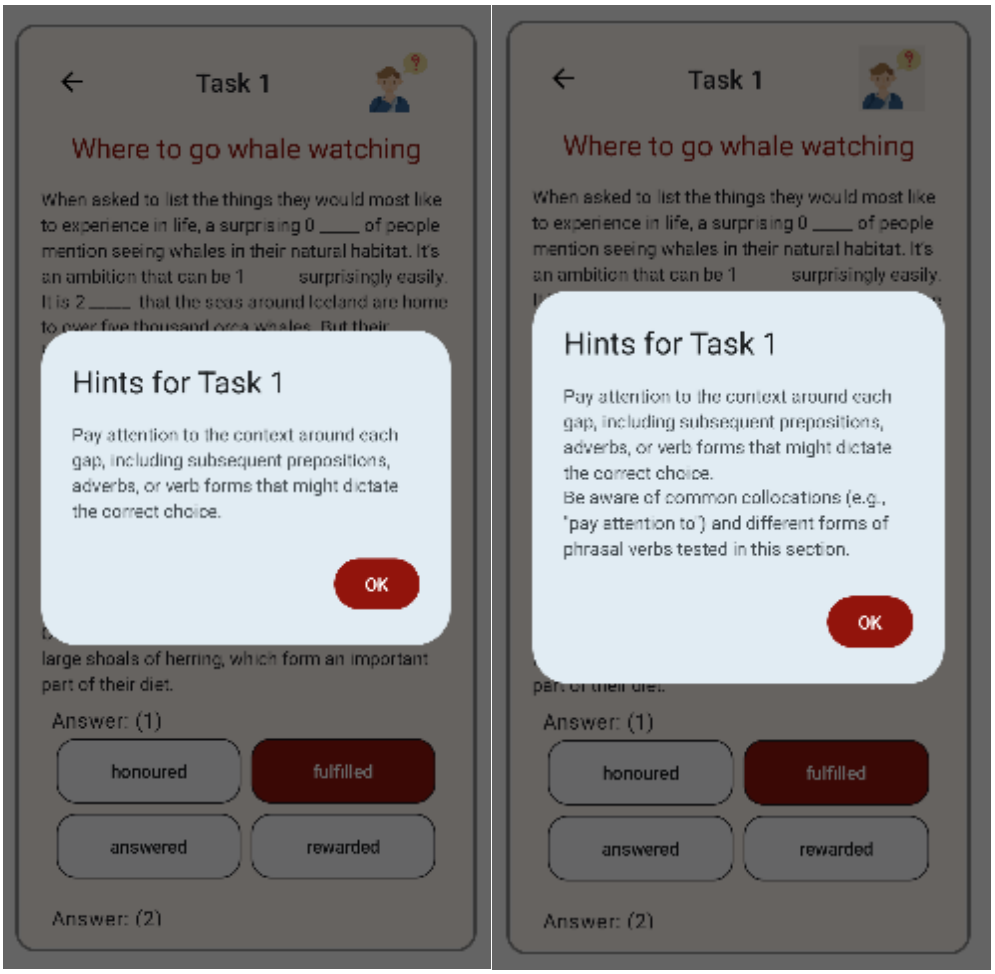
4.4.3.3. Savjeti zaslona ispita

Savjeti su funkcionalnost od koje svi korisnici imaju koristi. Novim korisnicima ukazuje na potencijalne probleme kojih nisu svjesni, a iskusne podsjeća na greške koje mogu nastati prilikom nepažnje. Savjeti su pohranjeni u bazi podataka. Prikaz savjeta nalazi se na slici 4.22.



Slika 4.22. Prikaz kolekcije generalHints

Kako je objašnjeno u poglavlju 4.2.3.3. kroz testiranja zaključeno je da je prikaz dva savjeta najbolji. Međutim, korisnik može birati da mu se prikaže nula, jedan ili dva savjeta. Ova funkcionalnost omogućena je klikom na sliku koja je uvedena u poglavlju 4.4.1. Budući da kod nije kompliciran, bolje bi bilo prikazati kako izgleda ispitni ekran kada se klikne slika jednom, a kako kada se klikne dvaput. Prikaz ove funkcionalnosti nalazi se na slici 4.23.



Slika 4.23. Prikaz savjeta

4.4.4. Funkcionalnosti Jetpack Compose-a zaslona ispita

Unatoč tome što je kod za prikaz savjeta jednostavan, koristi jednu komponentu Jetpack Compose-a koja u ovom radu dosada nije korištena. Riječ je o AlertDialog-u.[16] Alert Dialog je proširena gradivna funkcija bazne gradivne funkcije Dialog. Ovo proširenje omogućuje dodavanje naslova, teksta, slike, događaja kada se klikne izvan okvira, gumba za poništavanje te gumba za potvrđivanje. U implementaciji u ovom završnom radu korišten je naslov, tekst, događaj kada se klikne izvan ovkira i gumb za potvrđivanje. Prikaz navedenih funkcionalnosti može se vidjeti na slici 4.24.

```
if (currentHintIndex >= 0) {
    AlertDialog(
        onDismissRequest = { currentHintIndex = -1 },
        title = { Text( text: "Hints for Task 1") },
        text = {
            Column { this: ColumnScope
                for (i in 0 ≤ .. ≤ currentHintIndex) {
                    Text(randomHints[i])
                }
            }
        },
        confirmButton = {
            Button(
                onClick = { currentHintIndex = -1 },
                colors = ButtonDefaults.buttonColors(
                    containerColor = colorResource(id = R.color.burgundy),
                    contentColor = Color.White
                )
            ) { this: RowScope
                Text( text: "OK")
            }
        }
    )
}
```

Slika 4.24. Prikaz korištenja Alert Dialog-a

4.4.5. Prikaz funkcionalnosti višestrukog odabira

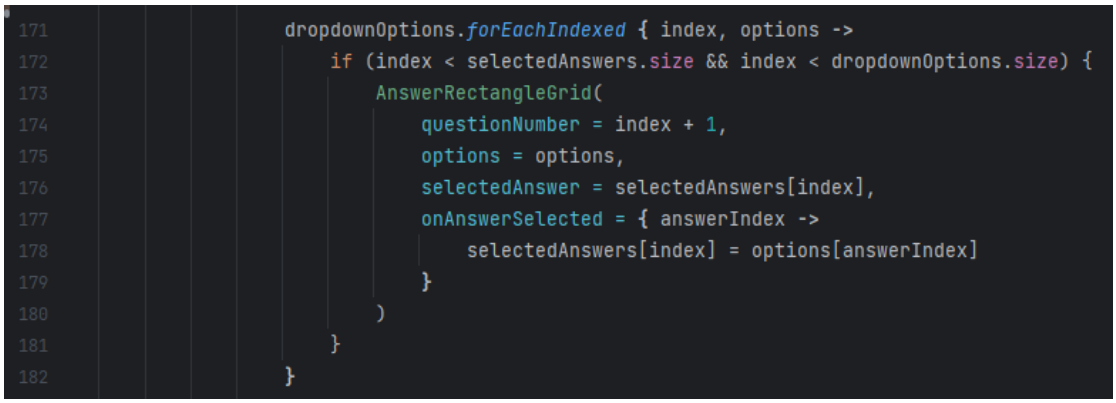
Jedna od najkompleksnijih funkcionalnosti aplikacije je funkcionalnost višestrukog odabira. Ova funkcionalnost sastoji se od 2 funkcije i pozivanja u glavnom dijelu programa. Funkcije su AnswerRectangleGrid i AnswerButtonRow. AnswerRectangleGrid brine se o formiranju gumba u formatu 2 stupca x 2 retka, odnosno 4 opcije. AnswerButtonRow brine se o označavanju pritusnutog odgovora. Unutar glavnog dijela programa za svaku povučenu opciju za odabir iz baze podataka on dodaje pripadajuće 4 nova gumba. Dakle, za 8 opcija za odabir, prikazano je 32 gumba, što je funkcionalnost koja se želi ostvariti. Na slici 4.25. prikazan je kod kojim je ostvarena funkcionalnost za AnswerRectangleGrid, na slici 4.26. funkcionalnost za AnswerButtonRow te na slici 4.27. poziv unutar glavnog dijela.

```
Luka.Agicic
@Composable
fun AnswerRectangleGrid(
    questionNumber: Int,
    options: List<String>,
    selectedAnswer: String?,
    onAnswerSelected: (Int) -> Unit
) {
    Column(
        modifier = Modifier
            .fillMaxWidth()
            .padding(8.dp)
    ) { this: ColumnScope
        Text( text: "Answer: ($questionNumber)", color = colorResource(id = R.color.dark_grey))
        Row(
            modifier = Modifier.fillMaxWidth(),
            horizontalArrangement = Arrangement.SpaceAround
        ) { this: RowScope
            AnswerButtonRow(
                options = options.subList(0, 2),
                selectedAnswer = selectedAnswer,
                onAnswerSelected = onAnswerSelected,
                questionNumber = questionNumber
            )
        }
        Row(
            modifier = Modifier.fillMaxWidth(),
            horizontalArrangement = Arrangement.SpaceAround
        ) { this: RowScope
            AnswerButtonRow(
                options = options.subList(2, 4),
                selectedAnswer = selectedAnswer,
                onAnswerSelected = onAnswerSelected,
                questionNumber = questionNumber,
                startAnswerIndex = 2
            )
        }
    }
}
```

Slika 4.25. Funkcija AnswerRectangleGrid

```
Luka.Agicic
@Composable
fun AnswerButtonRow(
    options: List<String>,
    selectedAnswer: String?,
    onAnswerSelected: (Int) -> Unit,
    questionNumber: Int,
    startAnswerIndex: Int = 0
) {
    Row(modifier = Modifier.fillMaxWidth()) { this: RowScope
        options.forEachIndexed { localIndex, option ->
            Column(modifier = Modifier.weight(1f)) { this: ColumnScope
                val absoluteAnswerIndex = localIndex + startAnswerIndex
                val isSelected = selectedAnswer == option
                val buttonColors = ButtonDefaults.buttonColors(
                    containerColor = if (isSelected) colorResource(id = R.color.burgundy) else MaterialTheme.colorScheme.surface,
                    contentColor = if (isSelected) Color.White else MaterialTheme.colorScheme.onSurface
                )

                Button(
                    onClick = { onAnswerSelected(absoluteAnswerIndex) },
                    modifier = Modifier
                        .fillMaxWidth()
                        .padding(4.dp),
                    colors = buttonColors,
                    border = BorderStroke(
                        width = 1.dp,
                        color = colorResource(id = R.color.dark_grey)
                    ),
                    shape = RoundedCornerShape(16.dp)
                ) { this: RowScope
                    Text(option, modifier = Modifier.padding(8.dp))
                }
            }
        }
    }
}
```

Slika 4.26. Funkcija AnswerButtonRow

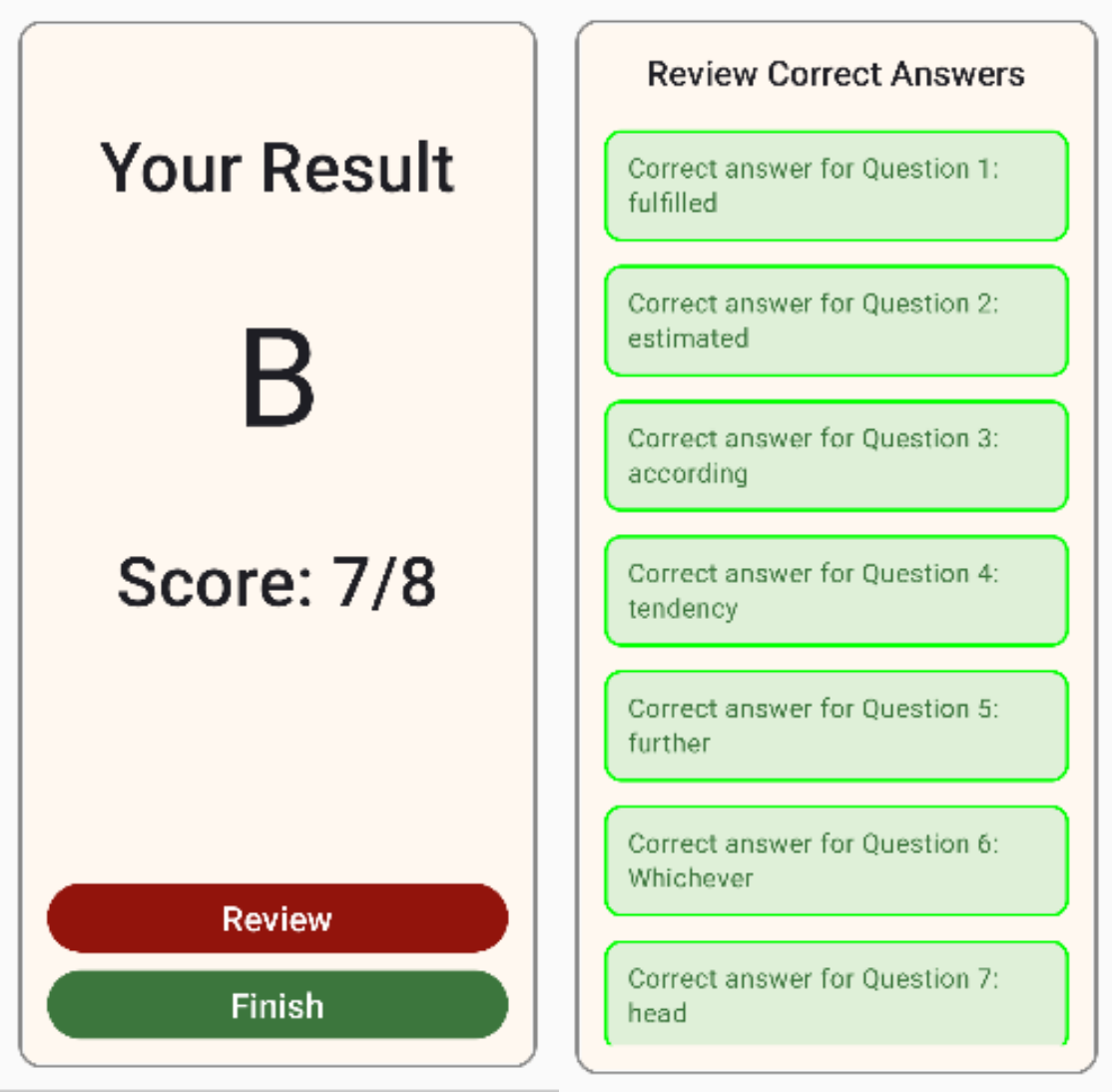


```
dropdownOptions.forEachIndexed { index, options ->
    if (index < selectedAnswers.size && index < dropdownOptions.size) {
        AnswerRectangleGrid(
            questionNumber = index + 1,
            options = options,
            selectedAnswer = selectedAnswers[index],
            onAnswerSelected = { answerIndex ->
                selectedAnswers[index] = options[answerIndex]
            }
        )
    }
}
```

Slika 4.27. Poziv unutar funkcije zaslona ispita

4.5. Zasloni rezultata i provjere

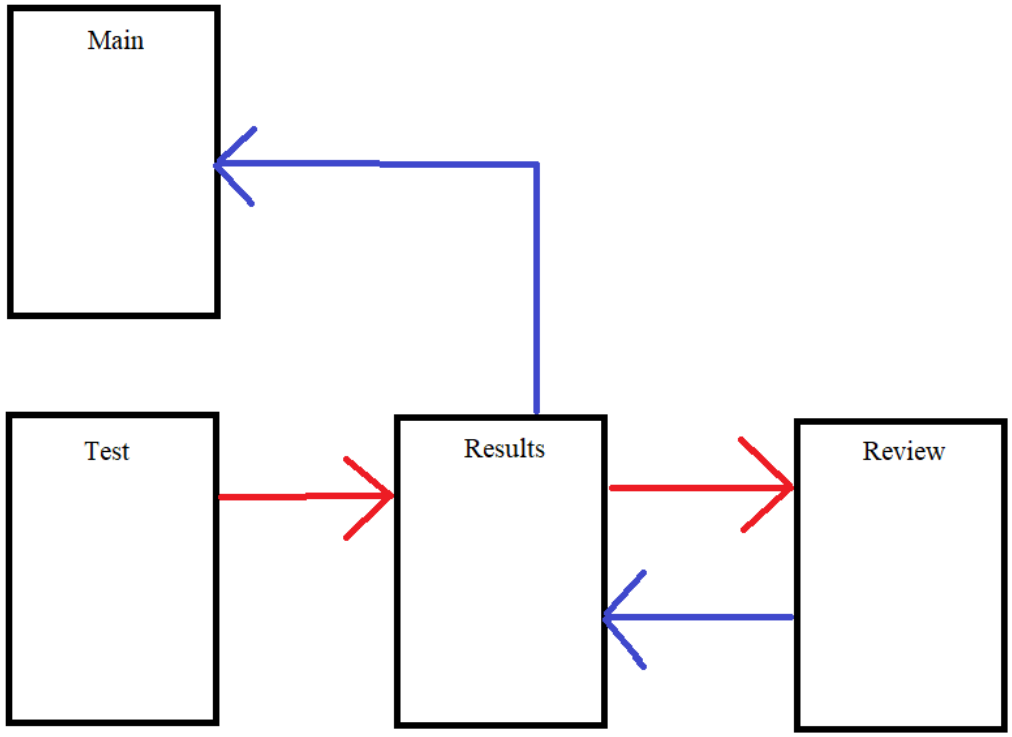
Zasloni rezultata i provjere obrađuju se zajedno jer imaju blisko vezane funkcionalnosti. Zaslon rezultata pokaže se nakon što kosnik preda ispit. Ukoliko ga zanimaju točna rješenja, može kliknuti na zaslon provjere. Ukoliko ne želi provjeriti rezultate, može kliknuti gumb završi (eng. Finish) s kojim završava ispit u cjelosti. Na slici 4.28. prikazane su opisane funkcionalnosti.



Slika 4.28. Prikaz zaslona rezultata i zaslona provjere

4.5.1. Navigacija zaslona rezultata i provjere

Navigacija između ovih zaslona vrlo je jednostavna. Do zaslona rezultata može se doći samo preko gumba na dnu ekrana na zaslonu ispita. Do zaslona provjere može se doći samo ako se klikne gumb na dnu zaslon rezultata. Na slici 4.29. nalazi se prikaz opisanog ponašanja.



Slika 4.29. Prikaz navigacije između zaslona rezultata i provjere

4.5.2. Rad s bazom podataka zaslona rezultata i provjere

Na zaslonu rezultata najvažnije je pratiti točne odgovore. Ova funkcionalnost omogućena je predajom identifikatora zadatka (eng. taskId) u argument funkcije ResultsScreen. TaskId predaje se i u funkciju ReviewScreen. Ovo je bitno kako zaslon provjere ne bi izbacivao točne odgovore od nasumičnog zadatka, već od točnog onog zadatka koji je nasumično izvučen. U poglavlju 4.4.3.1. detaljnije je objašnjeno kako ova funkcionalnost radi. Na slici 4.30. prikazano je pozivanje funkcije loadTask unutar zaslona provjere.

```
LaunchedEffect(taskId) { this: CoroutineScope
    if (taskId != null) {
        taskViewModel.loadTask(taskId)
    }
}
```

Slika 4.30. Poziv loadTask funkcije unutar zaslona provjere

4.5.2.1. ViewModel zaslona rezultata i provjere

Kako je prikazano u poglavlju 4.5.2. točni odgovori najvažniji su element koji se prati prilikom izrade zaslona rezultata i provjere. Prolaskom kroz točne odgovore i njihovim oblikovanjem u zelene pravokutnike sa zaobljenim rubovima dobiva se izgled koji je prikazan u poglavlju 4.5. Kod za ovu funkcionalnost nalazi se na slici 4.31.

```
76          correctAnswers.forEachIndexed { index, answer ->
77              Box(
78                  modifier = Modifier
79                      .fillMaxWidth()
80                      .padding(vertical = 8.dp)
81                      .border(
82                          border = BorderStroke(2.dp, Color.Green),
83                          shape = RoundedCornerShape(12.dp)
84                      )
85                      .background(Color( color: 0xFFDFF0D8), shape = RoundedCornerShape(12.dp))
86                      .padding(16.dp)
87              ) { this: BoxScope
88                  Text(
89                      text = "Correct answer for Question ${index + 1}: ${answer}",
90                      style = MaterialTheme.typography.bodyLarge.copy(
91                          fontSize = 18.sp,
92                          color = Color( color: 0xFF3C763D)
93                      ),
94                      modifier = Modifier.align(Alignment.CenterStart)
95                  )
96              }
97          }
```

Slika 4.31. Prikaz prolaska kroz točne odgovore

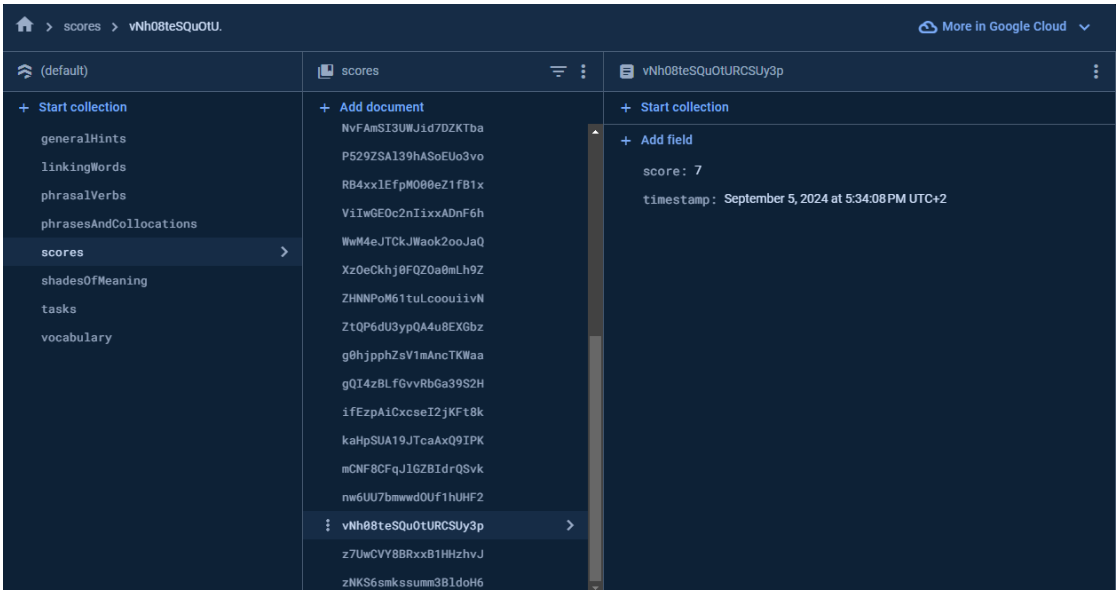
4.5.2.2. Firebase Firestore zaslona rezultata i provjere

Glavni zadatak baze podataka na zaslonu rezultata je pratiti broj točnih odgovora od ukupnog broja odgovora. Rezultat ove funkcionalnosti je ocjena u obliku slova A, B, C ili F. Ova slova predstavljaju način ocjenjivanja u Engleskoj (ocjena A predstavlja izvrstan, ocjena B vrlo dobar, ocjena C dobar i ocjena F nedovoljan). Prikazan ovoga je u poglavlju 4.5. Kod zaslužan za takvu funkcionalnost nalazi se na slici 4.32.

```
val totalQuestions = 8
val percentage = (scoreState.correctAnswers.toFloat() / totalQuestions) * 100

val gradeText = when {
    percentage < 60 -> "F"
    percentage < 75 -> "C"
    percentage < 90 -> "B"
    else -> "A"
}
```

Slika 4.32. Prikaz izračuna ocjene

Osim prikaza ocjene, Firebase Firestore pohranjuje i broj točnih odgovora i vrijeme predaje ispita. Za vrijeme predaje uzima se kada korisnik klikne gumb Finish na zaslonu rezultata. Prikaz ove funkcionalnosti u Firebase Firestore bazi podataka nalazi se na slici 4.33.



Slika 4.33. Prikaz rezultata u bazi podataka

5. ZAKLJUČAK

Pregledom sličnih rješenja poput English B2 FCE, The English Learning Lounge ili Exam Lift: English Practice uočeno je da sve navedene aplikacije nude zadatak višestrukog odabira s povratnim rezultatom. Međutim, sve pregledane aplikacije zahtjevaju plaćanje kako bi se dobio potpuni pristup svim funkcionalnostima. Prednost razvijene aplikacije u sklopu ovog završnog rada je što nema ograničenja u vidu plaćanja. Također, sustavom savjeta pokušano je skratiti proces učenja korisnika, bio on početnik ili napredni korisnik.

Za razvoj ove aplikacije korišten je programski jezik Kotlin, koji je zbog svoje jednostavne sintakse i sve veće popularnosti postao idealan izbor za razvoj Android aplikacija. Kotlin smanjuje broj linija napisanih linija koda i time ubrzava proces razvoja aplikacije. Za izradu korisničkog sučelja korišten je Jetpack Compose, moderan alat, koji također smanjuje potrebu za pisanje velikih količina koda u odnosu na tradicionalna rješenja poput XML-a. Pohrana podataka odrađena je na Firebase Firestore bazi podataka koja je integrirana u razvojno okruženje Android Studio. Omogućava pohranu zadataka i rezultata u stvarnom vremenu.

Aplikacija je orgranizirana da korisniku nudi teorijsku podlogu kroz 5 zaslona posvećenih pojedinom području. Osim toga, omogućava rješavanje ispita te na kraju prikazuje rezultate i točna rješenja. Svaki zaslon ima specifičnu funkcionalnost koja korisniku olakšava shvaćanje pojedinog područja. Posebno je koristan zaslon provjere gdje korisnik može usporediti svoje rezultate s točnim rezultatima i vidjeti gdje je pogriješio.

Zaključno, razvijena aplikacija predstavlja pristupačno rješenje koje koristi moderne tehnologije za razvoj mobilnih aplikacija. Puni potencijal ove aplikacije može se ostvariti kada bi se dodalo još sadržaja, poput zadataka za slušanje ili pisanje. Također, moderna rješenja poput zadataka generiranih umjetnom inteligencijom zasigurno mogu poboljšati korisničko iskustvo korištena aplikacije.

LITERATURA

[1] “Digital Around the World” [online]. Available: https://datareportal.com/global-digital-overview. [Accessed: 10.9.2024.].

[2] S. F., Isamiddinovna, “Mobile Applications As A Modern Means Of Learning English,” in *2019 International Conference on Information Science and Communications Technologies (ICISCT)*, pp. 1–5, 2019.

[3] L. F. M. G., Pedro, C. M. M. de O., Barbosa, C. M. das N., Santos, “A critical review of mobile learning integration in formal educational contexts,” *Int. J. Educ. Technol. High. Educ.*, no. 1, vol. 15, p. 10, Mar. 2018.

[4] “Google Play Store: number of apps 2024” [online]. Available: https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/. [Accessed: 6.9.2024.].

[5] A., Turner, “How Many Apps In Google Play Store? (2024)” [online], 17-Feb-2022. .

[6] “English B2 FCE - Apps on Google Play” [online]. Available: https://play.google.com/store/apps/details?id=com.uoe.english_b2&hl=en_US. [Accessed: 7.9.2024.].

[7] “Exam Lift: English Practice, Aplikacije na Google Playu” [online]. Available: https://play.google.com/store/apps/details?id=org.cambridgeenglish.examprepapp&hl=hr. [Accessed: 10.9.2024.].

[8] “The English Learning Lounge - Apps on Google Play” [online]. Available: https://play.google.com/store/apps/details?id=com.annanovasit.englishlearninglounge&hl=en_US. [Accessed: 7.9.2024.].

[9] “Learn English Grammar, Vocabulary, Reading & Listening” [online]. Available: https://www.esl-lounge.com/student/. [Accessed: 10.9.2024.].

[10] “Mobile OS market share worldwide 2009-2024” [online]. Available: https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/. [Accessed: 10.9.2024.].

[11] “Compose layout basics | Jetpack Compose” [online]. Available: https://developer.android.com/develop/ui/compose/layouts/basics. [Accessed: 10.9.2024.].

[12] “ViewModel overview” [online]. Available: https://developer.android.com/topic/libraries/architecture/viewmodel. [Accessed: 10.9.2024.].

[13] “Compose modifiers | Jetpack Compose” [online]. Available: https://developer.android.com/develop/ui/compose/modifiers. [Accessed: 10.9.2024.].

[14] “Choose a data structure | Firestore” [online]. Available: https://firebase.google.com/docs/firestore/manage-data/structure-data. [Accessed: 11.9.2024.].

[15] “Log” [online]. Available: https://developer.android.com/reference/kotlin/android/util/Log. [Accessed: 11.9.2024.].

[16] “Dialog | Jetpack Compose” [online]. Available: https://developer.android.com/develop/ui/compose/components/dialog. [Accessed: 12.9.2024.].

SAŽETAK

Sve su češće mobilne aplikacije za učenje engleskog jezika. Mnoge od njih nude zadatke s višestrukim odabirom, ali nijedna uz to ne nudi teorijsku podlogu i detaljne savjete. Cilj ovog završnog rada bio je izraditi mobilnu aplikaciju za vježbanje čitanja s razumijevanjem na primjeru zadataka višestrukog odabira. Nakon provedenog istraživanja, utvrđeno je da su je dostupnih rješenja malo i da su sva dostupna rješenja vrlo slična. Aplikacija je izrađena u programskom jeziku Kotlin i dostupna na operacijskom sustavu Android. Za pohranu podataka korištena je Firebase Firestore baza podataka. Aplikacija nudi interaktivno korisničko sučelje, zadatke koji vode korisnika kroz teoriju i na kraju rješavanje ispita. U budućnosti aplikacija se može proširiti dodavanjem novih katerogija ili modernih rješenja poput zadataka generiranih uz pomoć umjetne inteligencije.

Ključne riječi: engleski jezik, interaktivno učenje, mobilna aplikacija, višestruki odabir

ABSTRACT

Title: Application for reading comprehension on the example of multiple choice tasks.

Mobile application for learning English are becoming more and more common. Many of them offer multiple choice tasks, but none of them offer theoretical background and detailed hints. The goal of this final work was to create mobile application for reading comprehension on the example of multiple choice tasks. After conducting research, it was determined that there are a few available solutions and all of them are very similar. The application was created in the Kotlin programming language and is available for Android operating system. The Firebase Firestore database was used for data storage. The application offers interactive user interface, tasks that guide use through the theoretical parts and finally solving the exam. In the future, the application can be expanded by adding new categories or modern solutions like tasks generated with the help of artificial intelligence.

Keywords: English language, interactive learning, mobile application, multiple choice