# Aplikacija za učenje kondicionalnih rečenica

**Bionda, Filip Josip**



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

**Sveučilišni prijediplomski studij Računarstvo**

# Aplikacija za učenje kondicionalnih rečenica

**Završni rad**

**Filip Josip Bionda**

**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:** | Filip Josip Bionda |
| **Studij, smjer:** | Sveučilišni prijediplomski studij Računarstvo |
| **Mat. br. pristupnika, god.** | R4617, 27.07.2021. |
| **JMBAG:** | 0165089420 |
| **Mentor:** | doc. dr. sc. Dragana Božić Lenard |
| **Sumentor:** | Matej Arlović, univ. mag. ing. comp. |
| **Sumentor iz tvrtke:** | |
| **Naslov završnog rada:** | Aplikacija za učenje kondicionalnih rečenica |
| **Znanstvena grana završnog rada:** | **Programsko inženjerstvo (zn. polje računarstvo)** |
| **Zadatak završnog rada:** | Cilj je završnoga rada izraditi Android i iOS mobilnu aplikaciju za uvježbavanje kondicionalnih rečenica. Za izradu aplikacije će se koristiti razvojno okruženje Visual Studio Code unutar kojega je integriran Flutter kao framework koji koristi programski jezik Dart. Korisnik će moći izraditi svoj profil na kojemu će se bilježiti svi dosadašnji rezultati. Pri ulasku u aplikaciju korisnik će odabrati jedan od ponuđenih kondicionala gdje će mu se ponuditi teorijske postavke (tvorba, uporaba, iznimke) nakon čega će moći rješavati zadatke. U slučaju netočnog odgovora, korisniku će biti prikazano točno rješenje i objašnjenje. Tema rezervirana za Filipa Josipa |
| **Datum prijedloga ocjene završnog rada od strane mentora:** | 26.08.2024. |
| **Prijedlog ocjene završnog rada od strane mentora:** | Izvrstan (5) |
| **Datum potvrde ocjene završnog rada od strane Odbora:** | 11.09.2024. |
| **Ocjena završnog rada nakon obrane:** | Izvrstan (5) |
| **Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio sveučilišni prijediplomski studij:** | 11.09.2024. |

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

## IZJAVA O IZVORNOSTI RADA

**Osijek,** 11.09.2024.

| | |
|---|---|
| **Ime i prezime Pristupnika:** | Filip Josip Bionda |
| **Studij**: | Sveučilišni prijediplomski studij Računarstvo |
| **Mat. br. Pristupnika, godina upisa:** | R4617, 27.07.2021. |
| **Turnitin podudaranje [%]:** | 3 |

Ovom izjavom izjavljujem da je rad pod nazivom**: Aplikacija za učenje kondicionalnih rečenica**

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

i sumentora Matej Arlović, 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

Engleski jezik, kao globalno prihvaćen jezik, igra ključnu ulogu u međunarodnoj komunikaciji, obrazovanju i poslovanju. Usvajanje njegovih gramatičkih struktura, posebno kondicionala, predstavlja značajan izazov za mnoge učenike/studente. Kondicionali su složene konstrukcije koje omogućuju izražavanje uvjetnih situacija, mogućnosti i hipotetskih scenarija. Pravilno razumijevanje i primjena kondicionala zahtijevaju poznavanje pravila i sposobnost prepoznavanja odgovarajućih glagolskih vremena, što ih čini ključnim dijelom učenja engleskog jezika i neizostavnim za postizanje jezične kompetencije.

Paralelno s ovim izazovima, razvoj mobilnih tehnologija i sve veća dostupnost pametnih telefona značajno su transformirali pristup obrazovanju. Android i iOS operacijski sustavi dominiraju tržištem mobilnih uređaja, pružajući korisnicima pristup širokom spektru aplikacija. Mobilne aplikacije postale su ključan alat za personalizirano učenje, omogućujući učenicima/studentima pristup obrazovnim sadržajima na način koji je prilagođen njihovim individualnim potrebama. Ovaj tehnološki napredak omogućio je stvaranje inovativnih edukativnih alata koji pružaju interaktivno i pristupačno iskustvo učenja čineći obrazovanje dostupnijim nego ikad prije.

Iz kombinacije potrebe za učinkovitijim učenjem engleskog jezika i mogućnosti koje pružaju mobilne tehnologije, nastala je ideja za kreiranje aplikacije koja će korisnicima omogućiti samostalno učenje kondicionala. Aplikacija je osmišljena kako bi pružila detaljna objašnjenja gramatičkih struktura, pomažući korisnicima da bolje razumiju i usvoje ovaj kompleksni dio engleskog jezika. Na ovaj način, aplikacija ne samo da olakšava učenje kondicionala, već i potiče samostalnost u učenju te kontinuirano usavršavanje jezičnih vještina.

Ovaj se završni rad sastoji od šest poglavlja. Prvo poglavlje, *Uvod,* ukratko predstavlja ideju ovog završnog rada. U drugom poglavlju, *Istraživanje postojećih rješenja*, opisuju se i uspoređuju postojeća rješenja čija je svrha slična ili jednaka svrsi aplikacije kreirane za potrebe ovog završnog rada. U trećem se poglavlju opisuju tehnologije korištene u izradi aplikacije. Za kreiranje aplikacije korišten je Flutter razvojni okvir. Uz razvojni okvir Flutter koristi se i programski jezik Dart. Pri izradi aplikacije korišten je Firebase, odnosno njegove usluge Cloud Firestorea koja je zadužena za stvaranje baze podataka u kojoj se nalaze podatci o korisnicima, ispitima i rezultatima korisnika i Firebase Authentification koji se koristi za prijavu korisnika u aplikaciju. Korišten je i Hive kao lokalna baza podataka u koju se spremaju ispiti i rezultati rješavanja ispita kako bi se omogućilo korištenje aplikacije u uvjetima bez interneta. Četvrto poglavlje opisuje način izrade i

funkcionalnosti glavnih dijelova aplikacije. Peto poglavlje donosi prikaz korištenih modela unutar aplikacije, prikaz arhitekture koja je korištena za dohvaćanje, prikaz i ažuriranje podataka, te je opisana struktura Cloud Firestore baze podataka, kao i glavne funkcije pomoću kojih se podatci dohvaćaju i ažuriraju. U šestom poglavlju nalazi se zaključak u kojemu su sumirane glavne značajke aplikacije zajedno s prednostima i nedostatcima iste.

## 1.1. Zadatak završnog rada

Zadatak je završnog rada izrada mobilne aplikacije za Android i iOS operacijski sustav. Aplikacija će korisnicima omogućiti samostalno učenje gramatičkih struktura vezanih za kondicionale u engleskom jeziku. Korisnici će aplikacije imati mogućnost provjere stečenog znanja kroz ispite koji sadrže zadatke vezane za samo jedan tip kondicionala ili koji testiraju korisnikovo znanje kombinirajući više tipova kondicionala u jednome ispitu. Isto tako, za svaki riješeni ispit korisnik će imati opciju ponovnoga rješavanja i uvida u povijest rješavanja pojedinog ispita u svrhu praćenja napretka.

# 2. ISTRAŽIVANJE POSTOJEĆIH RJEŠENJA

Na tržištu aplikacija za učenje engleskog jezika, dostupno je mnogo rješenja koja kombiniraju učenje vokabulara i gramatike u jednoj aplikaciji. Ove aplikacije pokrivaju široka područja vokabulara i gramatičkih struktura često nudeći sveobuhvatne kurikulume koji pomažu korisnicima u cjelokupnom razumijevanju engleskog jezika.

Međutim, aplikacije koje se specifično bave samo jednim područjem gramatike, poput kondicionala u engleskom jeziku, su rijetke. Iako postoji malo takvih aplikacija, one ipak postoje i usmjerene su na korisnike koji žele detaljno proučiti ovu specifičnu gramatičku kategoriju. U ovom će se poglavlju opisati aplikacije koje se orijentiraju isključivo na kondicionale, s naglaskom na one koje su dostupne na Google Play Storeu i Apple App Storeu. Pružit će se pregled njihovih funkcionalnosti i pristupa učenju te usporediti s aplikacijom izrađenom za potrebe ovoga završnog rada.

## 2.1. Postojeća rješenja

Postojeća rješenja koja će biti opisana namijenjena su korištenju na Android i iOS operacijskim sustavima. Aplikacije koje će biti analizirane su Conditionals Grammar Test (Android i iOS), Conditionals (Android), LearnEnglish Grammar (iOS), English Grammar (iOS) i English Grammar: Learn & Test (Android).

### 2.1.1. Conditionals Grammar Test

Aplikacija Conditionals Grammar Test [1] osmišljena je kako bi korisnicima pružila tri različita načina provjere znanja vezanog uz kondicionalne rečenice u engleskom jeziku. Pri pokretanju aplikacije, korisnik može odabrati jedan od sljedećih načina rješavanja - 12 ROUNDS, TIME ATTACK i PRACTICE, koji su prikazani na početnom zaslonu.

Način 12 ROUNDS sastoji se od 12 ciklusa rješavanja, gdje korisnik dobiva rečenicu koju treba nadopuniti s jednim od ponuđenih rješenja koja se nalaze ispod prikazane rečenice. Ako korisnik odgovori na pitanje prije isteka brojača, dobiva dodatne bodove ovisno o preostalom vremenu. Tijekom 12 ciklusa, korisnik smije napraviti najviše pet pogrešaka. U slučaju da korisnik napravi pet pogrešaka ili odgovori na sva pitanja, prikazuje se novi zaslon s informacijama o rezultatu rješavanja ispita. Svaki od 12 ciklusa rješavanja po težini je zadatka isti i sastoji se od samo jedne rečenice koju treba nadopuniti. Ukoliko korisnik netočno odgovori na pitanje, gubi

jedan „život“ i dobiva negative bodove koji se oduzimaju od ukupnog rezultata koji je stalno prikazan u gornjem desnom kutu ekrana. Korisnik prilikom prikaza rezultata dobije samo prikaz rezultata bez objašnjenja grešaka. Postoji opcija ažuriranja rezultata na top 20 ljestvici, no ukoliko korisnik nije ostvario rezultat koji je bolji od rezultata korisnika koji se već nalazi na toj ljestvici, njegovo ime neće biti prikazano. Korisničko se ime postavlja prije ažuriranja rezultata na ljestvicu, odnosno prilikom prihvaćanja rezultata. Navedena opcija za slanje rezultata na ljestvicu nije obavezna i korisnik može odabrati da ne pošalje svoje rezultate te je preusmjeren na početni zaslon aplikacije.

TIME ATTACK način rješavanja temelji se na sličnom principu kao 12 ROUNDS, no uz ograničeno vrijeme rješavanja koje se sastoji od 180 sekundi. Cilj je točno riješiti što više pitanja unutar zadanog vremena, a na kraju se prikazuju rezultati. Zadatak je tipa nadopunjavanje praznina (isto kao i u 12 ROUNDS načinu), a nakon rješavanja ispita prikazuje se samo rezultat u bodovima. Korisnik i u ovome načinu rješavanja ima opciju ažuriranja svojih rezultata na top 20 ljestvicu.

Posljednji način, PRACTICE, nema vremenskog ograničenja i omogućuje korisniku da napravi neograničen broj pogrešaka. Ovaj je način dizajniran isključivo za vježbu kako bi korisnici mogli slobodno učiti i vježbati kondicionale bez pritiska vremenskoga ograničenja ili broja pogrešaka. Zadaci su nadopunjavanje praznina, a ponuđeni se odgovori nalaze ispod prikazane rečenice. U ovome načinu nema bodovanja i nakon što korisnik odgovori na sva pitanja, preusmjeren je na početnu stranicu. Korisniku na kraju rješavanja ispita za vježbu nisu objašnjene greške te mu nije ponuđena opcija za ponovno učenje gramatike.

U svim načinima rješavanja nalaze se svi tipovi kondicionala, a redoslijed je zadataka nasumičan.

Bitno je napomenuti kako su TIME ATTACK i PRACTICE načini rješavanja dostupni ukoliko korisnik koristi kupljenu verziju aplikacije. Cijena kompletne verzije aplikacije je 3 eura.

### 2.1.2. Conditionals

Aplikacija Conditionals [2] na početnom zaslonu nudi različite sekcije u obliku tipki. Ove sekcije uključuju učenje gramatike vezane uz kondicionale, vježbanje stečenog znanja o kondicionalima i provjeru znanja o kondicionalima.

Pritiskom na tipku za prikaz gramatike, korisniku se otvara novi ekran na kojemu može odabrati jedan od ponuđenih četiriju kondicionala. Na početku je objašnjeno što su kondicionali i

koji tipovi kondicionala postoje. Za svaki je kondicional objašnjeno kako se taj tip kondicionala tvori, kada se koristi i dani su primjeri rečenica koje sadrže taj tip kondicionala.

U sekciji za vježbu korisnici nakon odabira odgovora odmah dobivaju povratnu informaciju o točnosti i mogu ponoviti odgovor sve dok ne odaberu ispravan. U aplikaciji su dostupna 4 ispita za vježbu i svaki sadrži zadatke s popunjavanjem praznina, gdje su odgovori prikazani ispod ponuđene rečenice. Svaki ispit za vježbu sastoji se od 25 pitanja. Razina je težine svih zadataka jednaka. Nakon što korisnik riješi sve zadatke za vježbu, može odabrati opciju da ponovno riješi isti ispit ili da se vrati na početni zaslon. Tijekom rješavanja zadataka, nema vremenskog ograničenja, a nakon što korisnik riješi sve zadatke, nisu mu ponuđena objašnjena grešaka i nema ljestvice za usporedbu rezultata s ostalim korisnicima.

U sekciji za provjeru znanja korisnici također dobivaju povratnu informaciju o točnosti nakon svakog odgovora, ali odgovori se boduju i ulaze u konačan rezultat ispita. Informacije o (ne)točno riješenim zadatcima nalaze se iznad ponuđene rečenice koju treba nadopuniti. Nakon rješavanja ispita dobiva se rezultat u postotnom obliku. Također, korisnik može odabrati ponovno rješavanje istog ispita ili vraćanje na početni zaslon. Aplikacija nema nikakvu ljestvicu s rezultatima korisnika i korisniku nisu objašnjeni netočni odgovori te mu na kraju rješavanja nije ponuđena opcija za ponovno učenje gramatike. Unutar ispita za vježbanje i ispita za provjeru znanja nalaze se svi tipovi kondicionala i redoslijed je pitanja nasumičan. U aplikaciji su dostupna dva ispita za provjeru znanja i oba se sastoje od 50 pitanja. Pitanja su ista kao i u ispitima za vježbu, ali idu nasumičnim redoslijedom. Aplikacija je besplatna i ne nudi opciju dodatnog plaćanja za proširenje funkcionalnosti.

### 2.1.3. LearnEnglish Grammar

Aplikacija LearnEnglish Grammar [3] nudi korisniku učenje svih gramatičkih struktura koje se nalaze u engleskom jeziku. Korisnik na početnom ekranu ima dvije opcije: rješavanje ispita za vježbu ili ispita za provjeru znanja.Prva opcija sadrži gramatičke strukture podijeljene u više kategorija, ovisno o težini gradiva (Beginner, Elementary, Intermediate i Advanced). Svaka od navedenih kategorija može imati više paketa, npr. Intermediate Pack 1, 2 i 3. Svaka kategorija korisniku nudi rješavanje ispita za pojedinu gramatičku strukturu. Gramatička struktura je opisana u naslovu kartice unutar svake kategorije.

Za potrebe ovog završnog rada, opisat ćemo samo one dijelove aplikacije koji se bave kondicionalima, a oni se nalaze u Intermediate Pack 1 i Advanced Pack 1 kategorijama. U Intermediate Pack 1 se nalaze dva ispita za vježbu, „Conditionals: Zero & 1st“ i „Conditionals: 2nd & 3rd“. Oba ispita za vježbu se sastoje od 20 zadataka, zadatci su različitih tipova, a razlika između ta dva ispita je samo u gradivu kojeg pokrivaju. Korisnik rješava pitanja u kojima treba popuniti prazninu ponuđenim odgovorom, odgovoriti na postavljeno pitanje klikom na jedan od ponuđenih odgovora, poslušati zvuk i odabrati koja rečenica je u zvučnom isječku i poredati dijelove rečenice pravilnim redoslijedom. Kada korisnik odgovori na pitanje, prikazan mu je njegov odgovor i točnost samog. Korisniku nije prikazan točan odgovor, ali korisnik može poništavati svoje odgovore koliko god puta želi, što mu omogućuje ponovno rješavanje pitanja. Prilikom rješavanja ispita, korisnik se može kretati po pitanjima i nema vremenskog ograničenja. Na kraju, kada korisnik odluči da je završio rješavanje ispita, dobiva prikaz rezultata rješavanja. Prikazano mu je grafički koja je pitanja odgovorio točno (zelena boja), netočno (crvena boja) i neodgovorena pitanja (siva boja). Korisnik se nakon pregleda rezultata, može vratiti rješavati ispit za vježbu, kako bi ispravio svoje greške, a može i izaći iz ispita. U kategoriji Advanced Pack 1 nalazi se jedan ispit za vježbu, pod nazivom „Conditionals: Mixed“. Ovaj ispit je strukturno isti kao i prethodno dva opisana. Jedina razlika je u tome što ovaj ispit sadrži gramatičko gradivo iz oba ispita, tj. sadrži zadatke koji pokrivaju sve kondicionale. Tijekom rješavanja zadataka, nema vremenskog ograničenja, a nakon što korisnik riješi sve zadatke, nisu mu ponuđena objašnjena grešaka i nema ljestvice za usporedbu rezultata s ostalim korisnicima.

U dijelu koji je namijenjen za provjeru znanja, nalaze se ispiti po kategorijama. Svaki ispit je predstavljen jednom kategorijom, npr. Intermediate Pack 1. Ispit se sastoji od zadataka koji provjeravaju znanje svih gramatičkih cjelina koje se nalaze u toj kategoriji. Prilikom rješavanja ispita, korisnik može poništiti svoj odgovor, ali ga ne može provjeriti. Korisnik se i dalje može kretati kroz pitanja. Kada korisnik riješi ispit dobiva isti grafički prikaz kao i kod zadataka za vježbu, ali ovdje mu se nudi i prikaz njegovih odgovora, gdje može vidjeti gdje je pogriješio i što je odgovorio točno. Ispiti nemaju fiksni broj pitanja. Korisniku na kraju rješavanja ispita nije ponuđena opcija za učenje gramatike i nema ljestvice za usporedbu rezultata s drugim korisnicima aplikacije. Aplikacija je besplatna i ne nudi opciju dodatnog plaćanja za proširenje funkcionalnosti.

### 2.1.4. English Grammar

Aplikacija English Grammar [4] sadržava sva glagolska vremena. Vremena su razvrstana prema kategorijama Present, Past i Future. Također, aplikacija sadrži i kategorije vezane za izravni i neizravni govor zajedno s kondicionalima, pridjeve, priloge i pitanja zajedno sa zavisnim rečenicama. Svaka se kategorija sastoji od jednog ili više ispita čiji naslov predstavlja gramatičko gradivo koje se provjerava unutar ispita. Kategorije su označene različitim bojama.

Za potrebe ovog završnog rada opisat će se dio aplikacije koji se odnosi na kondicionale. On se nalazi u kategoriji Indirect & Conditionals. Korisnik ovdje može riješiti jedan ispit za vježbu vezan uz kondicionale. Odmah na početku rješavanja ispita može se otvoriti zaslon koji prikazuje kako se tvori pojedini kondicional i kada se koristi. Kada korisnik odgovori na pitanje, njegov odgovor bude označen crvenom bojom ukoliko je netočan ili zelenom ukoliko je točan. Ukoliko korisnik odgovori netočno, prikazan mu je odgovor koji je točan, ali ga korisnik ne može ponovno odabrati. Isto tako, nakon svakog korisnikovog odgovora na pitanje, postoji opcija zvučnog zapisa u kojem je izgovorena točna rečenica. Moguće je podesiti brzinu reprodukcije zvučnog zapisa na sporo, srednje i brzo. Broj pitanja nije prikazan. Kada korisnik odgovori na sva pitanja, usmjerava se na zaslon s rezultatima. Na zaslonu s rezultatima korisnik može vidjeti sve svoje odgovore, čija je točnost označena. Također se uz odgovore nalazi i tekst pitanja koji je ispunjen točnim odgovorima. Zadatci su u ispitu popunjavanje praznina, tako da korisnik, klikom na karticu s odgovorom, odgovara na postavljeno pitanje. Nakon rješavanja ispita, korisniku nije prikazana ljestvica s rezultatima i nije ponuđeno ponovno učenje gramatike ukoliko je korisnik loše riješio ispit.

Svaka kategorija sadrži i završni ispit koji pokriva svo gradivo koje se nalazi unutar kategorije. Prema tome, u kategoriji Indirect & Conditionals nalazi se završni ispit koji pokriva neizravni govor i kondicionale. Ispit je isto strukturiran kao i prethodno opisani ispit koji sadrži samo kondicionale. Jedina je razlika u tome što korisniku nije ponuđeno učenje gradiva tijekom rješavanja ispita. Nakon rješavanja nije prikazana ljestvica s bodovima kako bi korisnik mogao usporediti svoje rezultate s drugim korisnicima. Korisniku nije ponuđena opcija učenja gramatike vezane za ispit kojeg je rješavao ukoliko ga je loše riješio. Aplikacija je besplatna i ne nudi opciju dodatnog plaćanja za proširenje funkcionalnosti.

### 2.1.5. English Grammar: Learn & Test

Aplikacija English Grammar: Learn & Test [5] nudi učenje gramatike engleskog jezika. Opseg gradiva kojeg aplikacija pokriva je dosta velik. Kada korisnik uđe u aplikaciju, nalazi se na početnom ekranu, na kojemu ima nekoliko opcija. Korisnik može odabrati opcije za rješavanje ispita s objašnjenjem grešaka, stiskom na tipku s tekstom Grammar otvara se ekran na kojem korisnik može odabrati područje gramatike koje želi učiti, također može stiskom na tipku s tekstom Verb Tenses otvoriti ekran gdje korisnik bira koje će glagolsko vrijeme učiti. Korisnik također može odabrati opciju rješavanja ispita bez objašnjenja grešaka, pregled najčešćih grešaka u gramatici i pregled riječi koje se često pogrešno koriste. Aplikacija sadrži puno različitih gramatičkih struktura, a za potrebe će se ovog završnog rada opisati oni dijelove aplikacije koji se odnose na tematiku ovog završnog rada.

Klikom na tipku GRAMMAR, otvara se popis s karticama koje predstavljaju pojedine gramatičke strukture. Ovdje se nalazi i kartica s kondicionalima. Klikom na karticu koja predstavlja gramatiku vezanu za kondicionale, korisniku su ponuđene različite lekcije. Struktura lekcija je raznolika; od onih standardnih koje prikazuju kako se tvori pojedini kondicional pa do onih koje se bave određenim dijelom gramatike vezane uz kondicionale, npr. kada koristiti „when“, a kada „if“. Ukoliko je korisnik razumio gradivo, treba stisnuti tipku kojom to potvrđuje. To mu omogućava da kada prođe sve lekcije unutar jednog poglavlja, to poglavlje bude označeno kao završeno.

Korisnik se ne mora vraćati na početni zaslon kako bi riješio testove vezane za pojedino gradivo, već to može napraviti navigacijom na zaslon s ispitima pomoću navigacijske trake koja se nalazi na donjem dijelu ekrana. Korisnik rješava ispite koji imaju ponuđene odgovore. Kreće se po zadatcima kliznim pokretima prsta u lijevom i desnom smjeru. Kada odgovori na sva pitanja, automatski se nudi opcija završavanja rješavanja i prikazivanja rezultata ili nastavljanja rješavanja. Ukoliko korisnik stisne opciju za prikaz rezultata, isti mu se prikazuju. Točni su odgovori označeni zeleno, netočni crveno, a oni koji nisu odgovoreni sivo. Kod netočno odgovorenih pitanja, prikazan je i točan odgovor. Korisnik mora imati minimalno 40 % točno riješenih zadataka kako bi prošao ispit. Ispiti imaju vremensko ograničenje, koje varira od ispita do ispita, a svaki ispit ima i opciju za ponovno rješavanje ispita. Ovi ispiti nemaju objašnjenje u sklopu odgovora na pitanja.

Kada se korisnik vrati na početni zaslon može stisnuti na tipku EXPLANATION. Stiskom na nju, korisnik je usmjeren na zaslon s ispitima. Ovaj ekran sadrži listu unutar koje su kartice koje predstavljaju gradivo pokriveno kolekcijom ispita. Do ove razine, struktura i izgled ekrana na

kojem se prikazuju ispiti i ovog koji prikazuje ispite s objašnjenjem je jednak. Razlika između ispita s i bez objašnjenja je vidljiva. Može se primijetiti kako su ispiti s objašnjenjem dodatak aplikaciji kako bi korisnici bolje naučili neko gradivo, dok je uloga ispita bez objašnjenja provjera znanja. Ispiti s objašnjenjem nude objašnjenje nakon svakog odgovora neovisno o tome je li korisnik odgovorio točno ili ne. Za prolaz je potrebno ostvariti minimalno 40 % točnih odgovora. Nakon rješavanja ispita korisniku nije ponuđena ljestvica s bodovima kao ni ponovno učenje gramatike u slučaju lošeg rezultata. Aplikacija je besplatna i ne nudi opciju dodatnog plaćanja za proširenje funkcionalnosti.

## 2.2. Osvrt na postojeća rješenja

Analizom postojećih aplikacija može se zaključiti da sve nude alate za učenje gramatike kroz različite pristupe podjele sadržaja i strukturiranje zadataka. Aplikacije poput Conditionals Grammar Test, Conditionals, i English Grammar nemaju mogućnost dinamičkog dodavanja novih ispita, već se izmjene moraju unositi u kod, što predstavlja ograničenje u prilagodljivosti sadržaja. LearnEnglish Grammar nudi raznovrsnije tipove zadataka, uključujući nadopunjavanje, slaganje rečenica i unos teksta, omogućava i dinamično dodavanje ispita, dok English Grammar: Learn & Test sadrži samo zadatke za odabir točnog odgovora, ali omogućuje dinamičko dodavanje novih ispita.

Dodatno, neke aplikacije, poput Conditionals Grammar Test, zahtijevaju plaćanje za otključavanje dodatnih funkcionalnosti, što može predstavljati ograničenje za napredak među korisnicima koji traže besplatna rješenja.

U usporedbi s analiziranim aplikacijama, aplikacija razvijena za ovaj završni rad donosi nekoliko ključnih prednosti. Prvo, pruža korisniku detaljna objašnjenja za svaki tip kondicionala kroz dvije sekcije: Form i Usage. Unutar sekcije Form korisniku su prikazane tvorbe kondicionala u različitim vrstama rečenica, dok sekcija Usage objašnjava situacije u kojima se određeni kondicional koristi. Druga funkcionalnost uključuje rješavanje ispita s vremenskim ograničenjem, gdje korisnici nadopunjuju i glavnu (eng. main clause) i zavisnu surečenicu (eng. if-clause). Osim toga, aplikacija omogućuje korisnicima pregled svih svojih rezultata rješavanja za svaki ispit, što u nekim analiziranim aplikacijama nije moguće.

Za razliku od većine postojećih rješenja, aplikacija nudi dinamičko dodavanje novih ispita, omogućujući kontinuiranu prilagodbu sadržaja potrebama korisnika. Složenije aplikacije poput LearnEnglish Grammar i English Grammar: Learn & Test mogu poslužiti kao uzor za buduća

proširenja funkcionalnosti, posebno u pogledu raznovrsnosti zadataka i širokog spektra lekcija. Međutim, za razliku od nekih aplikacija koje zahtijevaju plaćanje za otključavanje dodatnih funkcionalnosti, što može biti ograničavajuće za korisnike, ova aplikacija nudi cjelokupno iskustvo bez takvih ograničenja, čime dodatno doprinosi svojoj pristupačnosti i korisničkom iskustvu.

Zaključno, aplikacija kombinira najbolje prakse iz postojećih rješenja i nudi inovativne značajke koje pružaju sveobuhvatan i prilagodljiv alat za učenje i provjeru znanja o kondicionalnim rečenicama u engleskom jeziku, s potencijalom za daljnji razvoj i integraciju složenijih funkcionalnosti.

# 3. KORIŠTENE TEHNOLOGIJE

U ovome će se poglavlju opisati korištene tehnologije za izradu aplikacije.

## 3.1. Flutter

Flutter [6] je razvojni okvir otvorenog koda za razvoj mobilnih, web i desktop aplikacija koji je razvio Google. Flutter je korišten za kreiranje i testiranje aplikacije na Android i iOS mobilnim uređajima. Ovaj je okvir omogućio razvoj jedinstvenog koda koji radi na objema platformama dosljedno pružajući korisničko iskustvo i ubrzavajući razvojni proces.

## 3.2. Dart

Dart [7] je objektno-orijentirani programski jezik kojeg je razvio Google, prvenstveno s ciljem olakšavanja razvoja brzih i responzivnih aplikacija za različite platforme. Dart je korišten kao programski jezik za razvoj aplikacije koju opisuje ovaj završni rad. Omogućava brzi razvoj i izvrsne performanse te pruža prednosti poput brzog "hot reload-a" i visoke učinkovitosti koda, što ubrzava razvojni proces i poboljšava korisničko iskustvo. Dart je u potpunosti integriran s Flutter okvirom, što ga čini idealnim izborom za razvoj modernih mobilnih aplikacija.

## 3.3. Firebase

Firebase [8] je platforma koju je razvio Google, a pruža niz alata i usluga za podršku razvoju aplikacija, uključujući poslužiteljske usluge, analitiku, autentifikaciju i još mnogo toga. U ovoj aplikaciji, Firebase je korišten za pružanje poslužiteljske usluge, omogućujući sigurno i pouzdano spremanje podataka te autentifikaciju korisnika. Konkretno, korišteni su Cloud Firestore za pohranu podataka u stvarnom vremenu i Firebase Authentication za upravljanje prijavom i registracijom korisnika, čime se osigurava visoka razina sigurnosti i jednostavnosti korištenja.

### 3.3.1. Cloud Firestore

Cloud Firestore [9] je skalabilna i fleksibilna NoSQL baza podataka u stvarnom vremenu te je dio Firebase platforme koju pruža Google. U ovoj aplikaciji, Cloud Firestore je korišten za pohranu i sinkronizaciju podataka, omogućujući visok stupanj dostupnosti i brz pristup informacijama. Njegova struktura temelji se na dokumentima i kolekcijama, što olakšava organizaciju i upravljanje podacima. Konkretno, Cloud Firestore je korišten za pohranu ispita i korisničkih rezultata, koji su dohvaćani i pohranjivani nakon rješavanja ispita. Također, svi korisnički podaci su sigurnosno spremljeni na Cloud Firestore.

### 3.3.2. Firebase Authentification

Firebase Authentication [10] je usluga iz Firebase platforme koja je korištena za autentifikaciju korisnika unutar aplikacije. Ova usluga omogućuje jednostavno upravljanje prijavom i registracijom korisnika putem različitih metoda, uključujući e-mail i lozinku te Google račun. Firebase Authentication pruža sigurnu autentifikaciju i upravljanje korisničkim identitetima, osiguravajući da su svi podaci korisnika zaštićeni i pouzdano upravljani.

## 3.4. Hive

Hive [11] je lagana, brza i učinkovita lokalna baza podataka koja je posebno dizajnirana za Flutter aplikacije. Hive je korišten za lokalnu pohranu podataka u aplikaciji. Konkretno, Hive je korišten za pohranu ispita i rezultata korisnika u lokalnu bazu podataka nakon dohvaćanja podataka s Cloud Firestorea. Za potrebe korištenja aplikacije bez internetske veze korištene su lokalne baze podataka za pohranu potrebnih podataka. Podatci s rezultatima uključuju rezultate koji su dohvaćeni s baze podataka te rezultate koje je korisnik sam kreirao rješavanjem ispita, a nakon toga odmah su pohranjeni u lokalnu bazu podataka. Ukoliko je korisnik riješio ispit bez internetske veze, rezultati će biti poslani s Hivea na Cloud Firestore čim se internetska veza obnovi. Hive je odabran zbog svoje brzine, učinkovitosti i jednostavnosti integracije s Flutterom.

# 4. PRIKAZ IZRADE I FUNKCIONALNOSTI APLIKACIJE

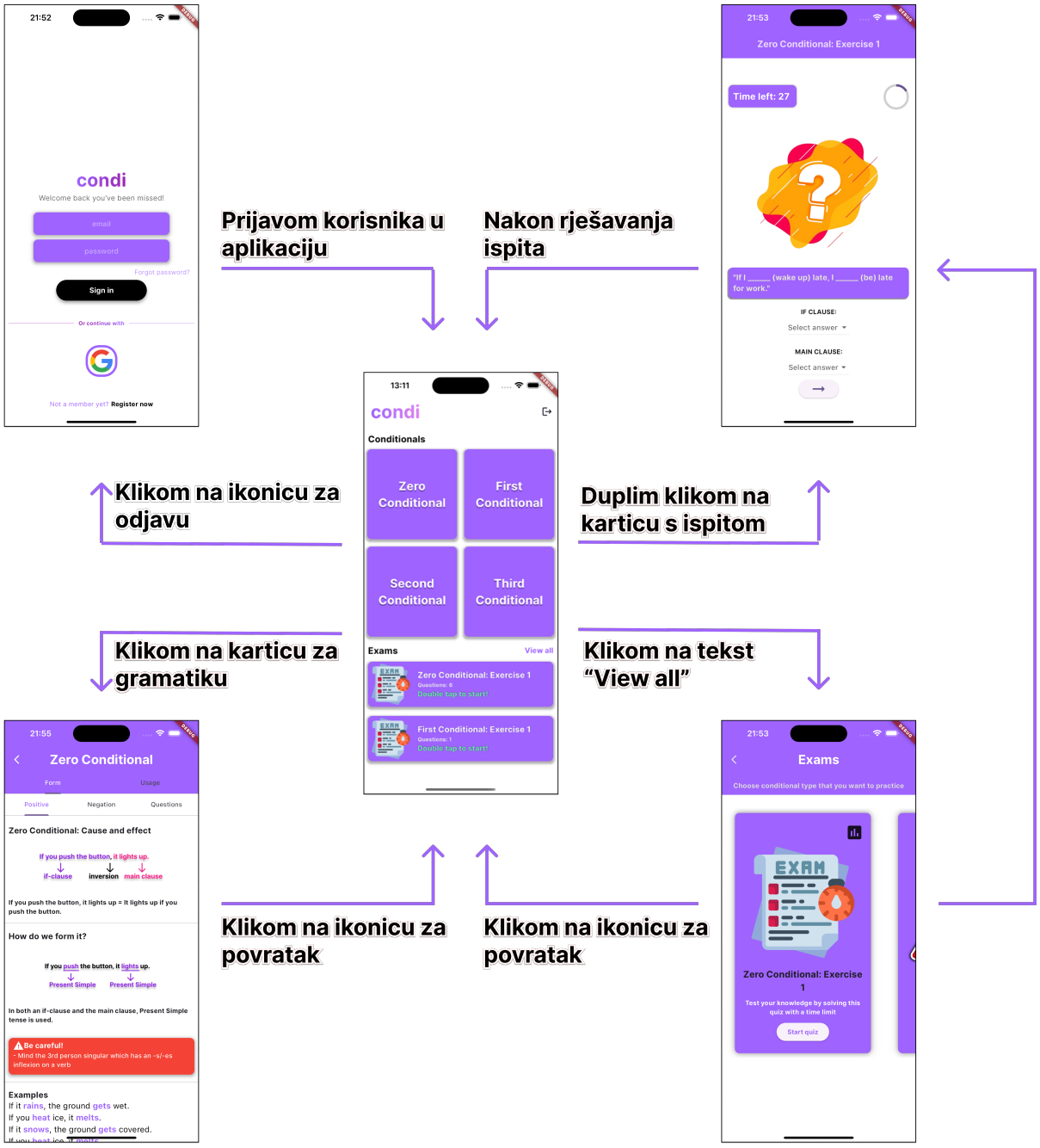
## 4.1. Navigacija unutar aplikacije

Navigacija je implementirana pomoću Navigatora [12], čijim se korištenjem zasloni stavljaju na stog (engl. *Stack*) [13], a pomoću ikonice koja se nalazi na traci aplikacije (engl. *AppBar*), vraća se na prethodni zaslon. Traka aplikacije prikaza je na Slika 4.1.


Slika 4.1. Traka aplikacije pomoću koje se kreće unutar aplikacije.

### 4.1.1. Koraci i putanje u aplikaciji

Korisnik započinje proces prijave u aplikaciju Condi unošenjem korisničkog imena i lozinke na početnom ekranu. Nakon uspješne prijave, preusmjerava se na glavni ekran aplikacije, gdje može vidjeti razne opcije za učenje gramatike i polaganje ispita. Klikom na ikonicu za odjavu u gornjem desnom kutu, korisnik se može odjaviti iz aplikacije. Klikom na karticu za gramatiku, poput Zero Conditional, prelazi na ekran s teorijskom obradom pojedine gramatičke cjeline. Na ovom ekranu korisnik može vidjeti gramatička pravila vezana za odabranu temu, a klikom na ikonicu za povratak vraća se na glavni ekran. Na glavnom ekranu, klikom na karticu s ispitom ili na tekst View all, korisnik prelazi na ekran s detaljima ispita, gdje može vidjeti informacije o ispitu i započeti ispit klikom na tipku Start quiz. Klikom na ikonicu za povratak vraća se na prethodni ekran. Prikaz dijagrama toka aplikacije nalazi se na Slika 4.2.




Slika 4.2. Dijagram toka aplikacije.

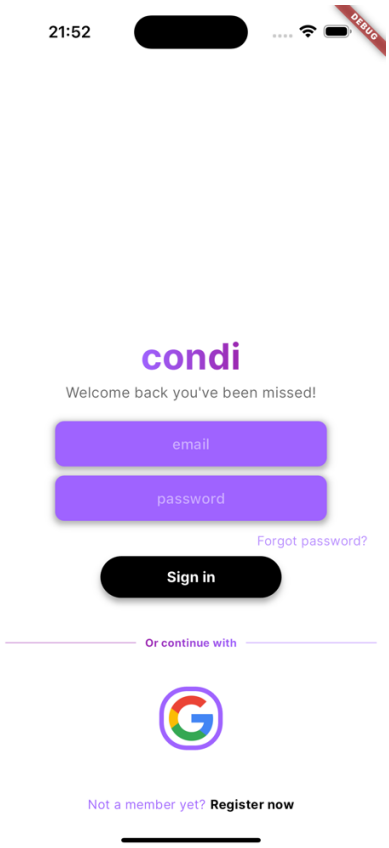
## 4.2. Prikaz glavnih dijelova aplikacije i njihovih funkcionalnosti

Svaki zaslon unutar aplikacije kreiran je kao kombinacija velikog broja widgeta/elemenata. Elementi mogu biti ugrađeni u razvojno okruženje u kojemu se radi ili se može kreirati vlastito okruženje. To omogućuje kreiranje i izmjenu već postojećih widgeta kako bi se dobilo određeno ponašanje koje do tada nije bilo podržano. Zbog toga, oni su idealni za prikaz korisničkog sučelja i pružanje različitih funkcionalnosti koje korisnik očekuje kada vidi korisničko sučelje.

Kao što je već spomenuto u prethodnome poglavlju, početni zaslon omogućava navigaciju prema ostalim dijelovima aplikacije koji će se detaljnije opisati u narednim poglavljima.

### 4.2.1. Zaslon za prijavu

Zaslon za prijavu početni je zaslon aplikacije. Na njemu se registrirani korisnik može prijaviti u aplikaciju, ali i povratiti zaboravljenu lozinku. Izgled zaslona za prijavu prikazan je na Slika 4.3.




Slika 4.3. Zaslon za prijavu.

Zaslon za prijavu koristi klasu AuthService, čija je zadaća implementacija logike za autentifikaciju korisnika. Unutar AuthServicea nalazi se funkcija za prijavu korisnika putem emaila i lozinke. Slika 4.4. prikazuje AuthService funkciju za autorizaciju korisnika putem emaila i lozinke.

```dart
Future<User?> signInWithEmailAndPassword(String email, String password) async {
  try {
    UserCredential userCredential = await _auth.signInWithEmailAndPassword(
      email: email,
      password: password,
    );
    return userCredential.user;
  } on FirebaseAuthException catch (e) {
    throw e;
  }
}
```

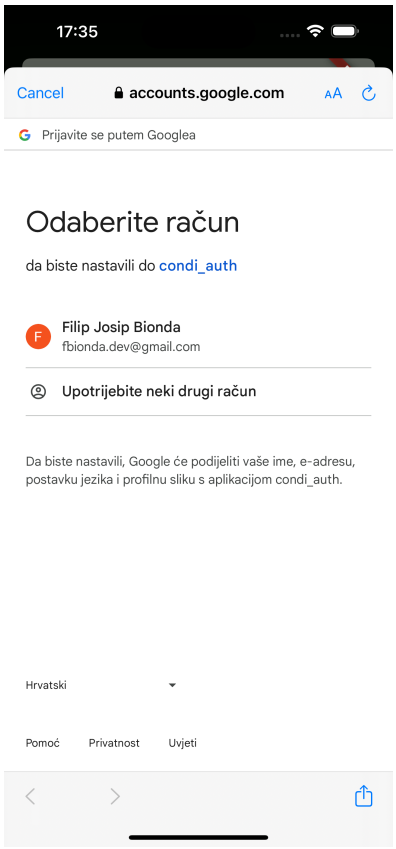
Slika 4.4. Implementacija funkcije za prijavu korisnika putem emaila i lozinke.

Također, na zaslonu za prijavu postoji i opcija prijave korisnika putem Google računa. Funkcija za prijavu korisnika putem Google računa implementirana je unutar AuthServicea, a prikaz njene logike nalazi se na Slika 4.5.

```dart
Future<User?> signInWithGoogle() async {
  final GoogleSignInAccount? googleUser = await _googleSignIn.signIn();

  if (googleUser == null) return null;

  final GoogleSignInAuthentication googleAuth =
      await googleUser.authentication;

  final credential = GoogleAuthProvider.credential(
    accessToken: googleAuth.accessToken,
    idToken: googleAuth.idToken,
  );

  UserCredential userCredential =
      await _auth.signInWithCredential(credential);
  User user = userCredential.user!;

  bool userExists = await checkIfUserExistsByEmail(user.email!);
  if (!userExists) {
    saveUserData(user);
  }
  return user;
}
```

Slika 4.5. Implementacija funkcije za prijavu korisnika putem Google računa.

Klikom na gumb za prijavu korisnika putem Googlea otvara se prozor unutar aplikacije koji nudi mogućnost odabira određenog Google računa s kojim se želimo prijaviti (Slika 4.6.).




Slika 4.6. Prozor za odabir Google računa za prijavu.

Ovaj je prozor implementiran pomoću biblioteke google_sign_in koja koristi nativne komponente za Android i iOS kako bi prikazala Google prijavni prozor.

### 4.2.2. Zaslon za registraciju

Klikom na tekst Register now na zaslonu za prijavu, otvara se novi zaslon za registraciju. Njegova je uloga omogućiti stvaranje novog računa. Korisnik ima dvije opcije: registracija putem emaila i lozinke ili registracija putem Google računa. Kod koji je zadužen za registraciju nalazi se u klasi AuthService i prikazan je na Slika 4.7.

```dart
Future<User?> signUpWithEmailAndPassword(String email, String password) async {
  try {
    UserCredential userCredential = await _auth.createUserWithEmailAndPassword(
      email: email,
      password: password,
    );
    User user = userCredential.user!;
    saveUserData(user);
    return user;
  } on FirebaseAuthException catch (e) {
    throw e;
  }
}
```

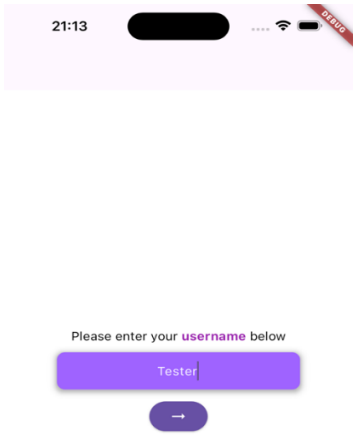
Slika 4.7. Implementacija registracije korisnika putem emaila i lozinke

Prilikom procesa registracije provjerava se postoji li korisnik u bazi podataka (Slika 4.8.).

```dart
Future<bool> checkIfUserHasCompletedProfileInfo(String uid) async {
  DocumentSnapshot<Map<String, dynamic>> snapshot =
      await FirebaseFirestore.instance.collection('users').doc(uid).get();

  if (snapshot['username'] == "") {
    return false;
  }
  return true;
}
```

Slika 4.8. Implementacija za provjeru postojanja korisnika u bazi podataka.

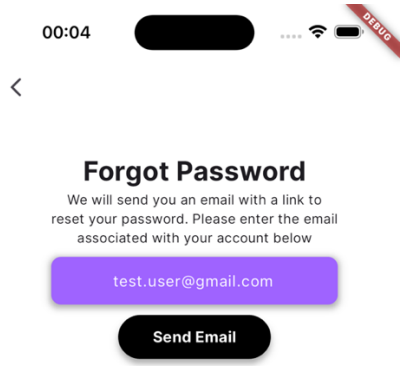
Ukoliko ne postoji, korisnik je preusmjeren na zaslon za unos korisničkog imena, što je prikazano na Slika 4.9.




Slika 4.9. Zaslon za unos korisničkog imena.

### 4.2.3. Povrat lozinke

U slučaju da je korisnik zaboravio svoju lozinku, može ju ponovno postaviti klikom na tekst Forgot password? Tada se korisniku otvara zaslon za povrat lozinke (Slika 4.10.)



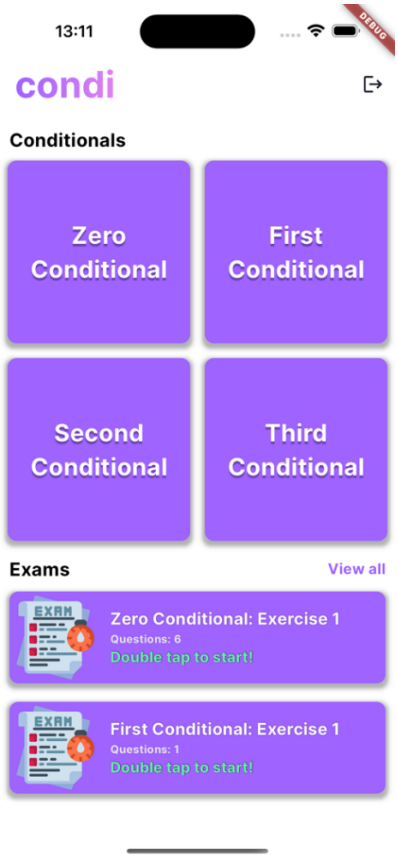

Slika 4.10. Zaslon za povrat lozinke.

Korisnik je upisao svoj email i stisnuo na tipku Send Email. Na korisnikov je email poslan zahtjev za ponovnim postavljanjem lozinke.

### 4.2.4. Početni zaslon

Početni zaslon (Slika 4.11.) otvara se ukoliko je korisnik već prijavljen u aplikaciju. On nudi opcije za učenje gramatike i provjeru znanja rješavanjem kratkih ispita. Ova se dva dijela nalaze pod Conditionals i Exams. Klikom na jednu od četiriju kartica pod Conditionals otvara se zaslon za učenje gramatike. Ispiti se pokreću duplim klikom (ovo je omogućeno jer u slučaju jednog klika, aplikacija može pružiti neželjeno ponašanje, npr. otvaranje zaslona za rješavanje ispita

prilikom slučajnog klika korisnika) na karticu ispita. Postoji i opcija odlaska na zaslon za prikaz ispita klikom na tekst View all.




Slika 4.11. Početni zaslon aplikacije.

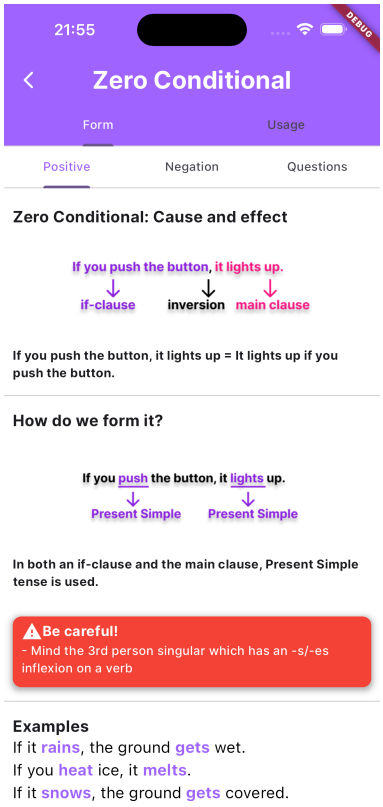
Korisnik ima i opciju odjave s trenutnog profila putem ikonice  koja se nalazi u gornjem desnom kutu (prikaz ikonice nalazi se na Slika 4.21, lijeva ikonica). Kod za odjavu korisnika prikazan je na Slika 4.12.

```
Future<void> signOut() async {
  try {
    await _auth.signOut();
    await _googleSignIn.signOut();
    print('User signed out!');
  } catch (e) {
    print('Error signing out: $e');
  }
}
```

Slika 4.12. Implementacija odjave korisnika.

### 4.2.5. Zaslon za gramatiku

Zaslon za gramatiku sastoji se od dva glavna dijela. To su Form i Usage. Unutar sekcije Form nalaze se tri odjeljka (Positive, Negation i Questions), čija je funkcija prikaz načina tvorbe kondicionala u izjavnim, negacijskim i upitnim rečenicama. Teorijska objašnjenja gramatike nalaze se u sklopu aplikacije i nisu preuzeta iz baze podataka.

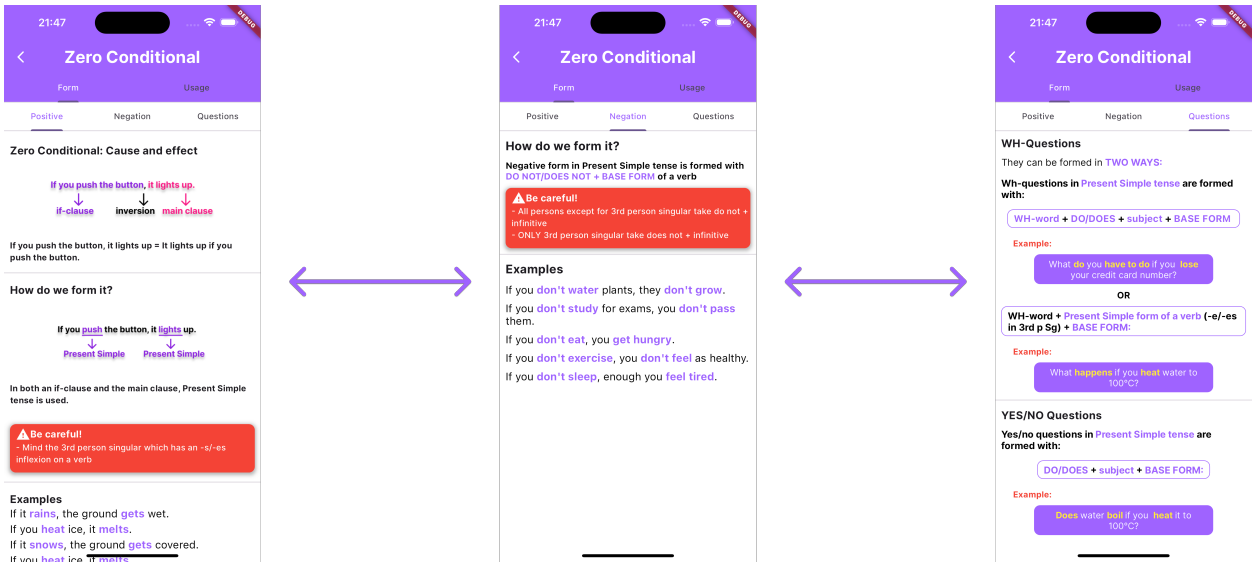



Slika 4.13. Form - zaslon za učenje tvorbe kondicionala.

Sekcija Usage nema odjeljke jer je njena funkcija prikaz načina korištenja odabranoga kondicionala. Izgled sekcije Usage nalazi se na Slika 4.15.

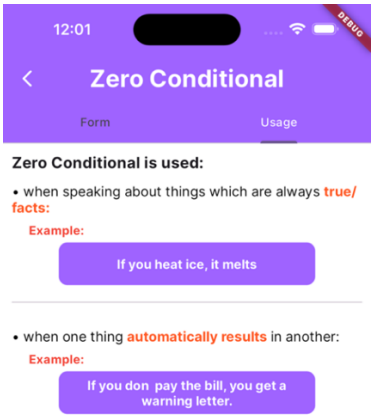
### 4.2.6. Odabir kondicionala

Korisnik odabire karticu vezanu za određeni kondicional. Kada se otvori zaslon za gramatiku, kliznim pokretima prsta u smjerovima lijevo-desno, korisnik dobiva uvid u tvorbu odabranog kondicionala u pozitivnim, negacijskim i upitnim rečenicama (Slika 4.14.)




Slika 4.14. Pregled tvorbe pozitivnih, negacijskih i upitnih kondicionalnih rečenica.

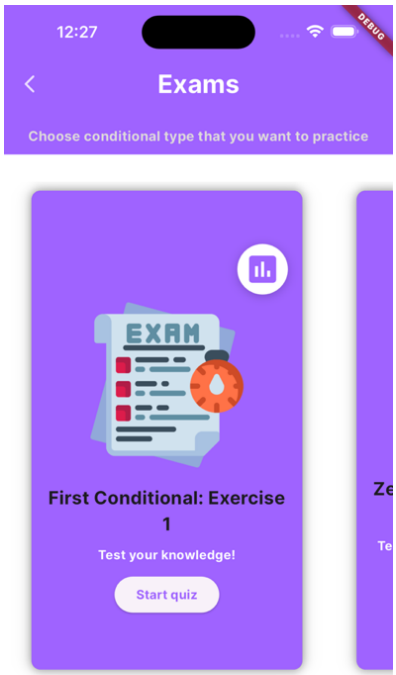
Klikom na sekciju Usage dobiva se uvid u vrste situacija u kojima se koristi nulti kondicional (Slika 4.15.).




Slika 4.15. Usage - zaslon za učenje uporabe kondicionala.

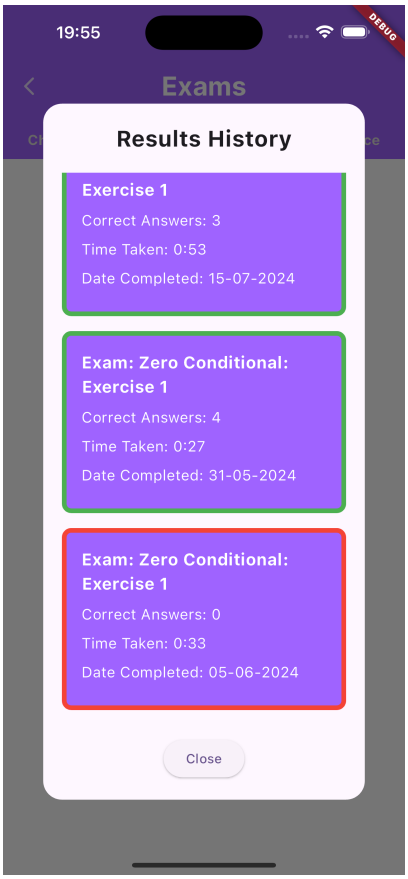
### 4.2.7. Zaslon za prikaz ispita

Kada korisnik izabere opciju View all na početnom zaslonu, otvara se zaslon za prikaz ispita (Slika 4.16.). Na ovome zaslonu korisnik dobiva uvid u sve dostupne ispite. Na početnom zaslonu korisnik ima uvid u samo dva nasumično odabrana ispita iz cijelog skupa ispita zbog čega je dodatno omogućen uvid u sve ispite. Prikaz ispita u obliku je kartice na kojoj se nalaze osnovni podatci vezani za ispit (naziv i opis ispita), ali se na njoj nalazi i ikonica kojom se otvara Dialog.




Slika 4.16. Zaslon za prikaz ispita.

Dialog predstavlja prozor koji se postavlja iznad zaslona za prikaz ispita, tj. korisnik dobiva osjećaj da je zaslon za prikaz ispita u pozadini. Izgled dijaloga prikazan je na Slika 4.17.

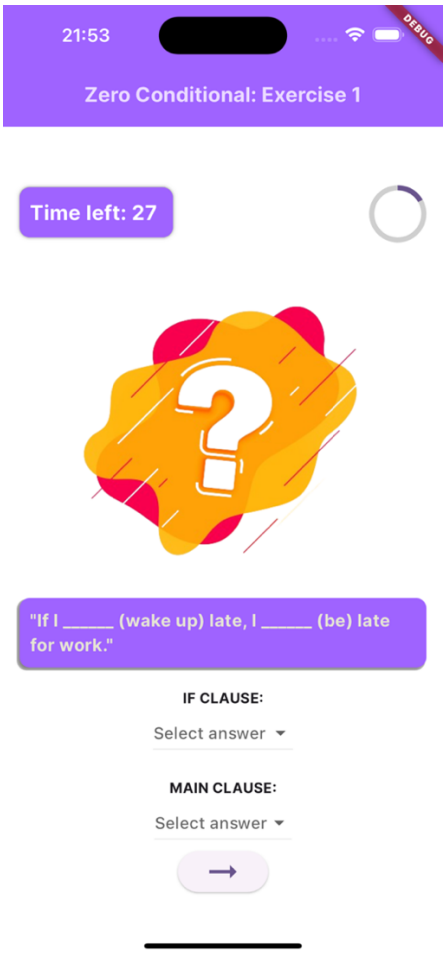



Slika 4.17. Prikaz rezultata korisnika.

Ukoliko je okvir rezultata obojen zelenom bojom, korisnik je uspješno riješio ispit. Ako je crvene boje, korisnik nije uspješno riješio ispit.

### 4.2.8. Zaslon za rješavanje ispita

Pritiskom na ElevatedButton widget, koji se nalazi na zaslonu za prikaz ispita, ili duplim klikom na karticu ispita (početni zaslon), započinje rješavanje ispita koje se odvija na zaslonu za rješavanje ispita. Korisnik rješava pitanje po pitanje. Postoji vremensko ograničenje, a korisnik odabire odgovore pomoću DropdownMenu widgeta koji korisniku spušta traku s mogućim odgovorima. Prikaz jednog pitanja na ispitu nalazi se na Slika 4.18.




Slika 4.18. Izgled pitanja na zaslonu za rješavanje ispita.

Postupak rješavanja ispita jednostavan je i intuitivan. Korisnik ima određeno vrijeme za rješavanje pojedinog pitanja, no može na pitanje odgovoriti prije isteka vremena i krenuti na novo pitanje. Kada vrijeme istekne, provjerava se točnost odabranih odgovora. Ako nema odabranih odgovora, sprema se prazan String, što ukazuje da korisnik nije odgovorio na pitanje. Pitanja se prikazuju nasumično. Implementacija nasumičnog redoslijeda pitanja prikazana je na Slika 4.19.

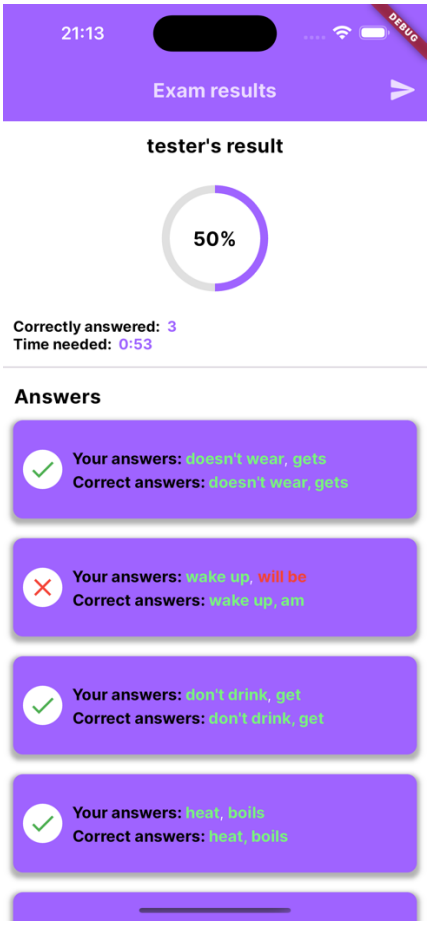
```
void shuffleQuestions() {
  widget.questions.shuffle(Random());
}
```

Slika 4.19. Implementacija nasumičnog redoslijeda pitanja.

### 4.2.1. Zaslon za prikaz detalja rješavanja ispita

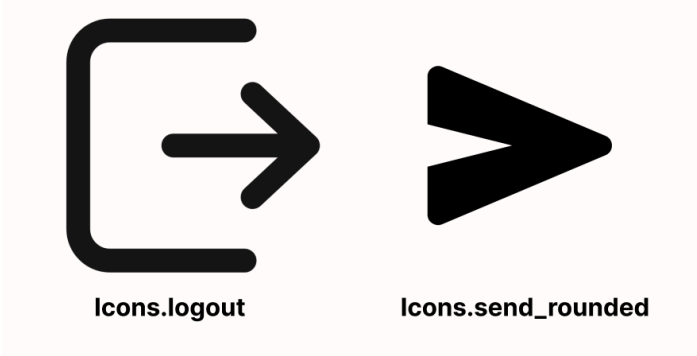
Nakon što korisnik odgovori na sva pitanja na ispitu ili nakon što istekne ukupno vrijeme rješavanja ispita, otvara se zaslon za prikaz detalja rješavanja ispita (Slika 4.20.). Na ovom zaslonu

korisnik dobiva uvid o vremenu koje mu je bilo potrebno za rješavanje, broju točnih odgovora, postotku uspješnosti rješavanja, odabranim odgovorima i točnim odgovorima.




Slika 4.20. Zaslon za prikaz detalja rješavanja ispita.

Nakon što korisnik završi s pregledom svih detalja, klikom na ikonicu (Slika 4.21, desna ikonica) u gornjem desnom kutu, izlazi s ekrana, a njegovi se rezultati spremaju na uređaj i šalju se na Cloud Firestore bazu podataka.




Slika 4.21. Ikonice korištene za prikaz operacija odjave i slanja rezultata.

# 5. RUKOVANJE PODATCIMA UNUTAR APLIKACIJE

Ovo poglavlje pruža pregled modela koji se koriste unutar aplikacije, objašnjavajući njihovu primjenu te način preuzimanja i pohranjivanja podataka u udaljenu bazu podataka (Cloud Firestore) i lokalnu bazu podataka (Hive) u situacijama bez pristupa internetu. Također, bit će predstavljena arhitektura za prikaz podataka na ekranu i rukovanje njima unutar aplikacije. Na kraju poglavlja, bit će prikazana struktura baze podataka na Cloud Firestoreu radi bolje razumljivosti organizacije i pohrane podataka u oblaku.

## 5.1. Modeli

U prethodnom poglavlju opisali smo izgled i funkcionalnosti aplikacije, ali je važno spomenuti kakvim sve modelima aplikacija raspolaže.

### 5.1.1. Pitanje

Ovaj model predstavlja klasu Question koja se koristi za pohranu i upravljanje podacima o pitanjima unutar aplikacije. Korištenjem Hivea, lokalne baze podataka, model omogućava serijalizaciju podataka, što olakšava njihovo spremanje i dohvaćanje. Svako pitanje ima tekst pitanja, točne odgovore za zavisnu i glavnu rečenicu te popise mogućih odgovora za oba dijela rečenice. Klasa Question pruža ključne podatke sloju korisničkog sučelja, omogućujući mu da pravilno prikaže potrebne informacije. Prikaz Question klase dan je na Slika 5.1.

```dart
@HiveType(typeId: 2)
class Question {
  @HiveField(0)
  final String text;

  @HiveField(1)
  final String ifClauseCorrectAnswer;

  @HiveField(2)
  final String mainClauseCorrectAnswer;

  @HiveField(3)
  final List<String> answersIfClause;

  @HiveField(4)
  final List<String> answersMainClause;

  Question({
    required this.text,
    required this.ifClauseCorrectAnswer,
    required this.mainClauseCorrectAnswer,
    required this.answersIfClause,
    required this.answersMainClause,
  });
}
```

Slika 5.1. Klasa Question.

### 5.1.2. Ispit

Klasa Exam predstavlja model koji se koristi za pohranu i organizaciju podataka o ispitima u aplikaciji. Korištenjem Hivea, ovaj model omogućava učinkovito spremanje i dohvaćanje podataka, uključujući tip ispita, naslov, sliku, opis i pripadajuća pitanja. Objekti tipa Exam omogućuju stvaranje elemenata korisničkog sučelja koji predstavljaju ispite, a sadrže sve potrebne informacije pohranjene unutar objekta Exam. Na taj način, korisničko sučelje može precizno prikazati ispit i sve povezane detalje vezane za njega. Klasa Exam prikazana je na Slika 5.2.

```dart
@HiveType(typeId: 1)
class Exam {
  @HiveField(0)
  final String type;

  @HiveField(1)
  final String title;

  @HiveField(2)
  final String imageUrl;

  @HiveField(3)
  final String description;

  @HiveField(4)
  final List<Question> questions;

  @HiveField(5)
  int numberOfQuestions = 0;

  @HiveField(6)
  final Uint8List? imageData;

  Exam({
    required this.type,
    required this.title,
    required this.imageUrl,
    required this.description,
    required this.questions,
    this.imageData,
  }) {
    numberOfQuestions = questions.length;
  }
}
```

Slika 5.2. Klasa Exam.

### 5.1.3. Rezultat

Klasa Result koristi se za generiranje i pohranu rezultata nakon završetka ispita. Ovi podaci mogu se prikazati u korisničkom sučelju kako bi korisnici mogli pregledati svoje rezultate, uključujući broj točnih odgovora i vrijeme provedeno na ispitu. Na taj način, klasa Result, pomoću njezinih instanci, omogućuje aplikaciji praćenje i prikaz povijesti korisničkih rezultata, što može biti korisno za analizu napretka korisnika tijekom vremena. Prikaz klase Result je na Slika 5.3.

```dart
@HiveType(typeId: 3)
class Result {
  @HiveField(0)
  final Map<int, Map<String, String?>> answers;

  @HiveField(1)
  final int correctAnswers;

  @HiveField(2)
  final int elapsedSeconds;

  @HiveField(3)
  final String examTitle;

  @HiveField(4)
  final DateTime timestamp;

  Result({
    required this.answers,
    required this.examTitle,
    required this.correctAnswers,
    required this.elapsedSeconds,
    required this.timestamp,
  });
```

Slika 5.3. Klasa Result.

## 5.2. Primjena MVVM obrasca

Ova aplikacija koristi MVVM [14] (Model-View-ViewModel) arhitektonski obrazac kako bi osigurala jasno odvajanje odgovornosti, olakšala upravljanje podacima i omogućila učinkovito prikazivanje podataka korisnicima. MVVM arhitektura omogućava aplikaciji da ostane modularna, jednostavna za održavanje i skalabilna, posebno kada se radi o rukovanju podacima iz udaljenih i lokalnih baza podataka.

### 5.2.1. Model

Model u ovoj aplikaciji obuhvaća sve podatke i poslovnu logiku, što je pohranjeno u Cloud Firestoreu i Hiveu. Cloud Firestore omogućuje sinkronizaciju podataka u stvarnom vremenu među korisnicima, dok Hive služi za lokalnu pohranu podataka, omogućujući pristup aplikaciji čak i bez internetske veze. Ova kombinacija osigurava da aplikacija ostane funkcionalna i pouzdana u svim uvjetima.

### 5.2.2. View

View predstavlja korisničko sučelje koje prikazuje podatke i omogućuje interakciju s korisnikom. U ovoj aplikaciji, korisničko sučelje uključuje različite zaslone, poput početnog

zaslona (Slika 4.11.), koji prikazuje nasumično odabrane ispite, i zaslona za pregled dostupnih ispita (Slika 4.16.) te dijaloga za prikaz rezultata (Slika 4.17.). Svaki od ovih zaslona koristi podatke koje im pruža ViewModel te ih prikazuje.

### 5.2.3. ViewModel

ViewModel služi kao posrednik između Modela i Viewa, upravljajući logikom i podatcima koje View, odnosno Widget prikazuje. U ovoj aplikaciji, glavne ViewModel klase su ExamViewModel i ResultViewModel. ExamViewModel upravlja podacima o ispitima, preuzima ih iz Cloud Firestorea i pohranjuje u lokalnu bazu podataka Hive. Na taj način osigurava da su ispiti uvijek dostupni, čak i kada nema internetske veze. ResultViewModel upravlja rezultatima ispita, pohranjujući ih lokalno i sinkronizirajući s Cloud Firestoreom kada je to moguće. Ove klase omogućuju aplikaciji da učinkovito upravlja podacima i osigura da su prikazani podaci uvijek točni i ažurirani, bez obzira na stanje mreže.

## 5.3. Firestore i Hive funkcionalnosti unutar aplikacije

Funkcionalnosti koje upravljaju korisnicima, ispitima i rezultatima implementirane su pomoću Cloud Firestorea i Hivea. Cloud Firestore predstavlja bazu podataka na oblaku, dok Hive predstavlja lokalnu bazu podataka. Prilikom prikaza koda, Hive funkcionalnost spominje se kao box varijabla, kojoj prethodi ime kutije, ovisno o tome za što je kutija korištena. Primjer toga su examBox i resultsBox.

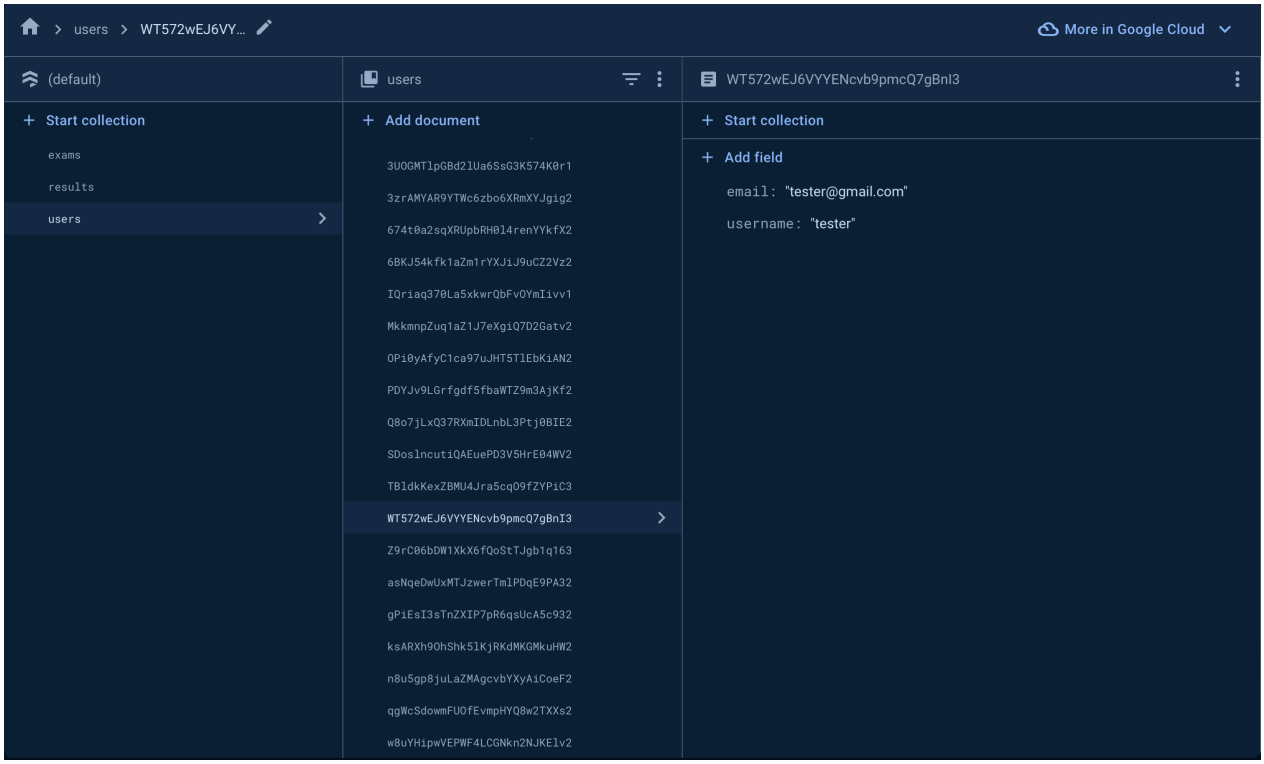
### 5.3.1. Struktura Cloud Firestore baze podataka

Unutar Cloud Firestore baze podataka, kreirane su tri kolekcije, a to su Users, Exams i Results koje predstavljaju prijavljene korisnike u aplikaciji, popis svih dostupnih ispita i rezultate svakog pojedinog korisnika.

### 5.3.2. Kolekcija za prijavljene korisnike

Kolekcija za prijavljene korisnike (Slika 5.4.) sadrži dokumente koji imaju nasumičan ID i sadrže sve potrebne podatke o prijavljenom korisniku. Ovo je glavna kolekcija unutar baze podataka jer nosi ključne informacije za daljnje upravljanje rezultatima korisnika. Razlog

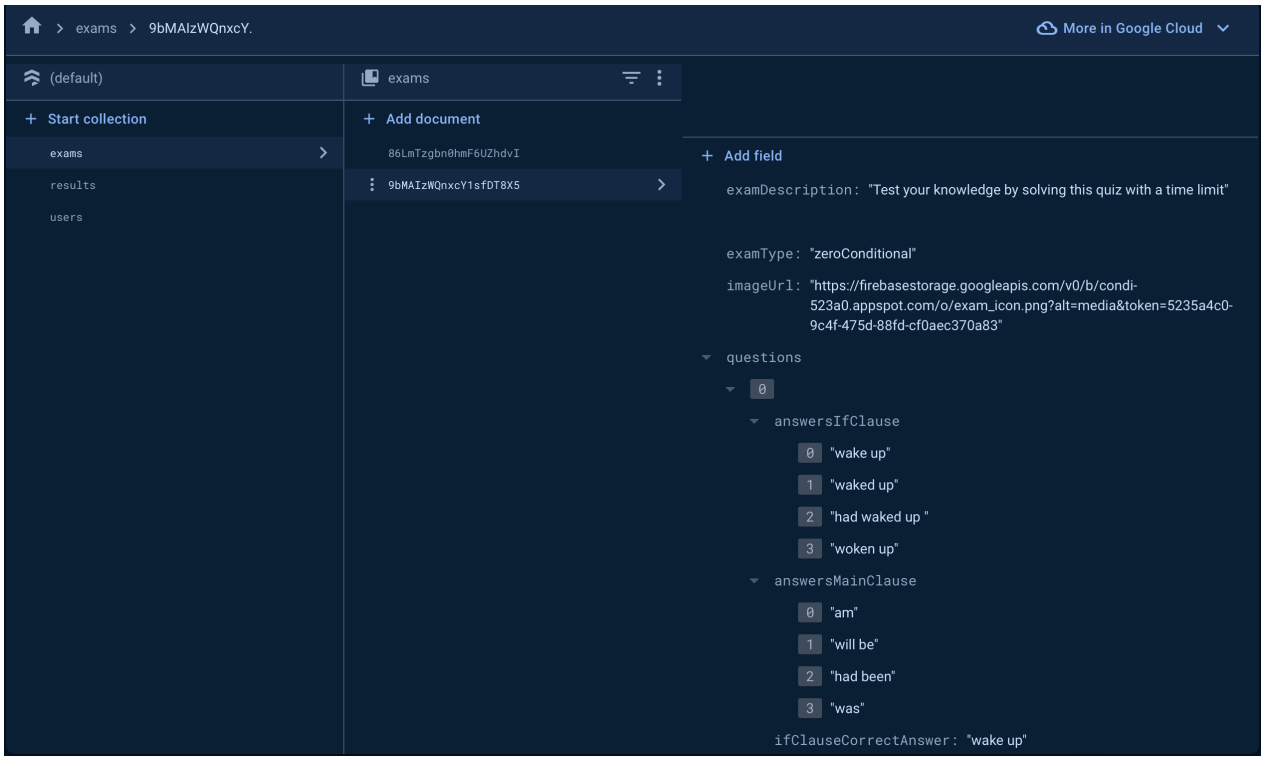
korištenja nasumičnog ID-a jest prevencija podudaranja dvaju ili više korisnika, što može rezultirati velikim problemima u radu i logici aplikacije.




Slika 5.4. Prikaz kolekcije za prijavljene korisnike.

### 5.3.3. Kolekcija za ispite

Ova kolekcija predstavlja jedinu pristupnu točku za aplikaciju u kojoj se mogu kreirati ispiti. Postoji jasna forma koju treba popuniti kako bi se kreirao ispit, no kreiranje ispita na ovaj način ima prednosti i nedostatke. Prednost je lagan postupak kreiranja i neograničenost u broju postavljenih pitanja i odgovora, dok su greške prilikom unosa podataka nedostatak. Ukoliko nije dobro definirano rukovanje iznimkama, aplikacija može prikazati pogrešne podatke ili se isključiti. Svaki ispit kreiran je prema klasi Exam unutar aplikacije. Slika 5.5. prikazuje strukturu kolekcije za ispite.

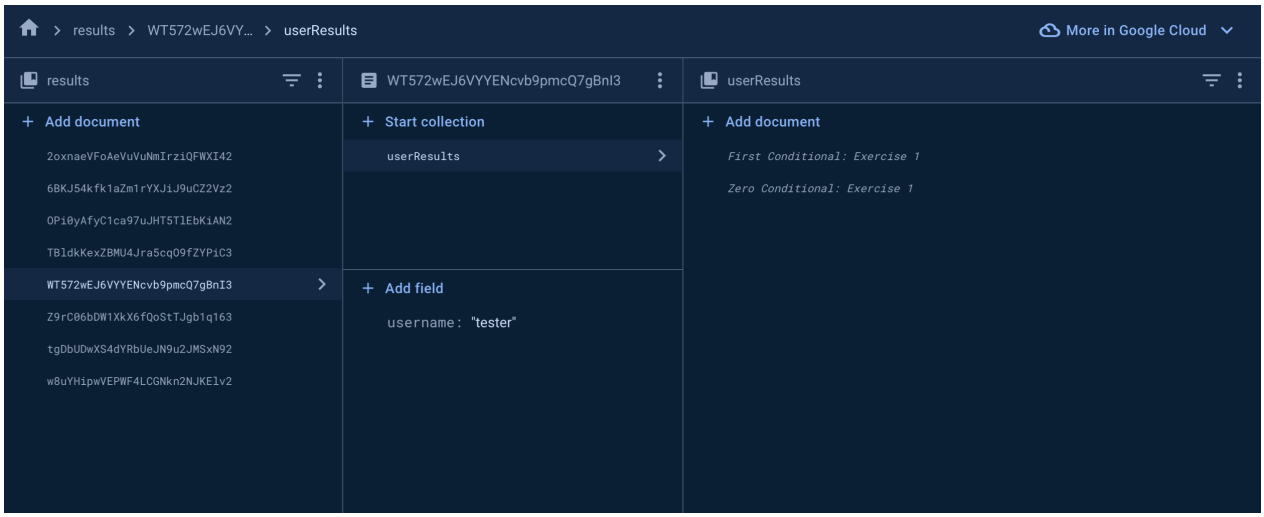



Slika 5.5. Prikaz kolekcije za ispite.

Struktura kolekcije nije složena. Svaki dokument predstavlja jedan ispit. Svaki ispit sadrži podatke o nazivu ispita, opisu ispita, vrsti kondicionala kojem ispit pripada, URL slike iz Cloud Storagea preko koje dobivamo sliku na ispitu unutar aplikacije te popis svih pitanja za ispit koja su definirana u obliku rječnika (Map).

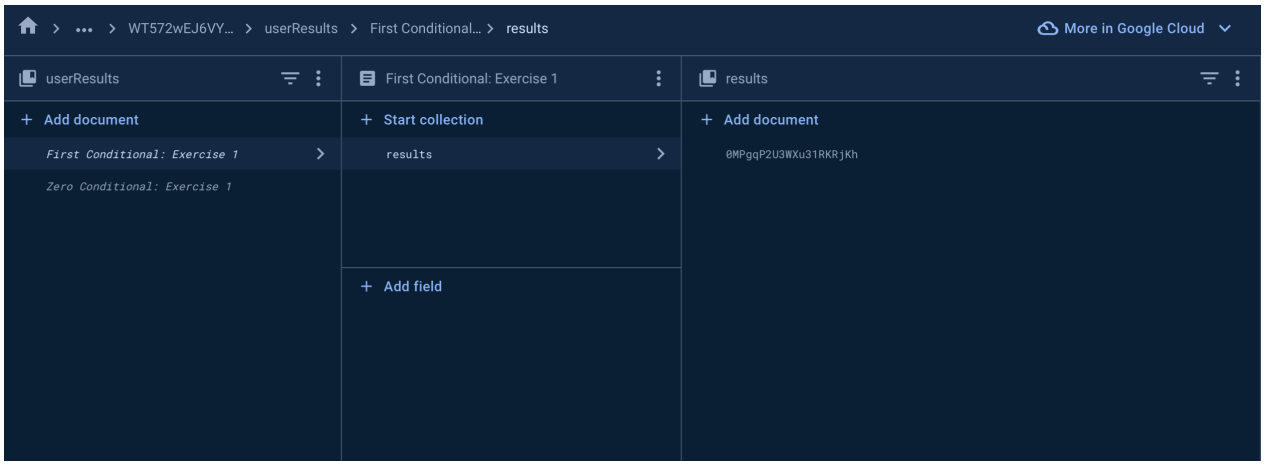
### 5.3.4. Kolekcija za rezultate

Kolekcija za rezultate najsloženija je kolekcija u bazi podataka. Sastoji se od triju slojeva. Prvi se sloj sastoji od korisničkih ID-eva kao naziva dokumenata. Svaki taj dokument sadrži kolekciju za prikaz rezultata korisnika za pojedini ispit (userResults) i korisničko ime korisnika. Drugi sloj započinje kolekcijom userResults koja sadrži dokumente čiji ID dokumenta odgovara nazivu ispita koje je korisnik riješio. Prikaz dosad spomenute strukture nalazi se na Slika 5.6.




Slika 5.6. Prikaz prvog i drugog sloja kolekcije za rezultate.

Zadnji sloj započinje dokumentom koji predstavlja ispit. Unutar ovog dokumenta nalazi se kolekcija koja sadrži sve rezultate korisnika (results). Dokumenti unutar prethodno spomenute kolekcije imaju nasumično generiran ID dokumenta. Prikaz posljednjeg sloja nalazi se na Slika 5.7.




Slika 5.7. Prikaz posljednjeg sloja kolekcije za prikaz rezultata.

### 5.3.5. Implementacija funkcija za dohvaćanje i ažuriranje kolekcija

Funkcije koje su korištene za dohvaćanje podataka s Cloud Firestorea su složene, a u radu su prikazani samo najrelevantniji dijelovi koji odrađuju najvažnije zadatke.

### 5.3.6. Funkcija za dohvaćanje i ažuriranje ispita

Funkcija initializeExams ključna je za dohvaćanje i ažuriranje podataka o ispitima unutar aplikacije. Ona prvo otvara lokalnu Hive bazu podataka, omogućujući pristup prethodno pohranjenim ispitima na uređaju. Zatim učitava te ispite i sprema ih u listu koja se koristi za prikaz u korisničkom sučelju. Nakon toga, funkcija provjerava dostupnost internetske veze i, ako je dostupna, dohvaća najnovije podatke iz Cloud Firestorea. U ovom koraku ažurira lokalnu bazu podataka s novim ili izmijenjenim ispitima te uklanja one koji više nisu aktualni. Na taj način, initializeExams() osigurava da su podaci o ispitima uvijek sinkronizirani i dostupni, pružajući korisnicima pouzdano i ažurirano iskustvo, bez obzira na stanje internetske veze.

```
Future<void> initializeExams() async {
  await _openBox();
  await _loadExams();
  await _checkForUpdates();
}
```

Slika 5.8. Implementacija funkcije za preuzimanje ispita.

### 5.3.7. Funkcije za dohvaćanje i ažuriranje rezultata

Funkcija fetchAllResults služi za dohvaćanje svih rezultata ispita za određenog korisnika. Prvo otvara lokalnu Hive bazu podataka specifičnu za tog korisnika. Ako je lokalna baza prazna, funkcija preuzima rezultate iz Cloud Firestorea i pohranjuje ih lokalno. Nakon što su podaci dohvaćeni, funkcija obavještava sve povezane prikaze da su rezultati ažurirani i spremni za prikaz, osiguravajući da su korisnički podaci uvijek dostupni i ažurirani. Prikaz funkcije dan je na Slika 5.9.

```dart
Future<void> fetchAllResults(String userId) async {
  final hiveBoxName = 'resultsBox_$userId';
  var resultsBox = await Hive.openBox(hiveBoxName);

  if (resultsBox.isEmpty) {
    final querySnapshot =
        await resultCollection.doc(userId).collection('userResults').get();

    for (var doc in querySnapshot.docs) {
      String examTitle = doc.id;
      await fetchResultsForExam(userId, examTitle);
    }
    notifyListeners();
  }
}
```

Slika 5.9. Implementacija funkcije za dohvaćanje rezultata.

Novi rezultat korisnika kreira se nakon korisnikovog rješavanja ispita te se sprema na Cloud Firestore, ali i u lokalnu bazu podataka. Ove su funkcionalnosti implementirane unutar funkcije submitResults koja služi za završavanje ispita, pri čemu kombinira lokalno spremanje rezultata u Hive bazu podataka i njihovo slanje na udaljenu bazu Cloud Firestore. Prvo, funkcija koristi ResultViewModel za spremanje rezultata lokalno. Nakon uspješnog spremanja, korisniku se prikazuje povratna informacija putem SnackBara koji potvrđuje predaju rezultata, a trenutni se prikaz zatvara. Na kraju, funkcija šalje rezultate na Cloud Firestore. Prikaz opisane funkcije nalazi se na Slika 5.10.

```dart
Future<void> submitResults(BuildContext context, int percentage) async {
  final resultViewModel = Provider.of<ResultViewModel>(context, listen: false);
  await resultViewModel.saveResultLocally(result);

  if (context.mounted) {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(
        content: Text('Results submitted!'),
        backgroundColor: Colors.green,
      ), // SnackBar
    );
    Navigator.of(context).pop();
  }

  await resultViewModel.sendResultToFirebase(result, percentage);
}
```

Slika 5.10. Implementacija funkcije za spremanje rezultata.

# 6. ZAKLJUČAK

Zadatak je ovog završnog rada bio izraditi mobilnu aplikacije za samostalno učenje i uvježbavanje kondicionalnih rečenica u engleskom jeziku. Nakon provedenoga istraživanja postojećih rješenja, zaključeno je da iako postoji veliki broj alata za učenje engleskog jezika, rijetki su oni koji se specijaliziraju isključivo za kondicionalne rečenice. Aplikacije poput Conditionals Grammar Test i Conditionals nude osnovne funkcionalnosti i jednostavne zadatke, no nedostaje dubinska analiza i mogućnost praćenja napretka korisnika, što je inkorporirano u ovu aplikaciju. Također, aplikacije poput LearnEnglish Grammar, English Grammar i English Grammar: Learn & Test nude mogućnosti učenja kondicionala, ali nisu specijalizirane samo za to područje. To može dovesti do nezainteresiranosti korisnika za aplikacijom ukoliko je teško pronaći traženo gradivo vezano uz kondicionale.

Za potrebe je razvoja mobilne aplikacije za Android i iOS operacijske sustave korišten je Flutter razvojni okvir. Firebase je korišten za pohranu podataka i autentifikaciju omogućavajući sigurnu pohranu korisničkih informacija i brzo dohvaćanje podataka. Hive je integriran kao lokalna baza podataka omogućujući rad aplikacije i u uvjetima bez internetske povezanosti. Ove tehnologije, zajedno s intuitivnim korisničkim sučeljem, čine aplikaciju funkcionalnom i lako dostupnom širokoj publici.

U funkcionalnom pogledu, aplikacija nudi zadatke usmjerene na samoučenje i provjeru znanja o kondicionalnim rečenicama. Korisnici mogu rješavati zadatke s nadopunjavanjem rečenica odabirom ispravnog odgovora iz padajućeg izbornika s ponuđenim opcijama. Aplikacija također uključuje ispite koji kombiniraju različite tipove kondicionala, što korisnicima omogućuje testiranje cjelokupnog razumijevanja ove gramatičke strukture. Uz to, aplikacija bilježi rezultate svakog ispita omogućujući korisnicima praćenje njihovog napretka. Ova funkcionalnost pomaže korisnicima da prepoznaju područja koja zahtijevaju dodatno učenje i omogućuje im praćenje vlastitog poboljšanja u razumijevanju i primjeni kondicionalnih rečenica.

Zaključno, ovaj završni rad predstavlja cjelovito rješenje za specifičnu potrebu učenja kondicionalnih rečenica. Aplikacija, razvijena korištenjem modernih tehnologija i pristupa usmjerenog na korisnika, omogućuje personalizirano i učinkovito učenje. Iako je aplikacija uspješno implementirana, prepoznati su i daljnji koraci za unaprjeđenje poput dodavanja novih lekcija, poboljšanja u dizajnu i funkcionalnosti sučelja te dodatnih alata za analizu korisničkog napretka, kao i dodavanje zadataka različitog tipa. Na taj način, aplikacija može postati još relevantnija i korisnija te doprinijeti boljem razumijevanju engleskog jezika među korisnicima.

# LITERATURA


[1] Conditionals Grammar Test - Apps on Google Play [online]. Dostupno na: https://play.google.com/store/apps/details?id=air.com.littlebigplay.games.free.conditionals&hl=en. [1.8.2024.].

[2] ENGLISH CONDITIONAL SENTENCES - Apps on Google Play [online]. Dostupno na: https://play.google.com/store/apps/details?id=com.em.emqscondtionalsentences&hl=en. [31.8.2024.].

[3] LearnEnglish Grammar (UK ed.)“ [online], 15-kol-2024. Dostupno na: https://apps.apple.com/us/app/learnenglish-grammar-uk-ed/id488099900. [31.8.2024.].

[4] Learn English Grammar Easily on the App Store [online]. Dostupno na: https://apps.apple.com/us/app/learn-english-grammar-easily/id1508746228. [31.8.2024.].

[5] English Grammar: Learn & Test - Apps on Google Play [online]. Dostupno na: https://play.google.com/store/apps/details?id=com.milinix.englishgrammartest&hl=en. [31.8.2024.].

[6] Flutter - Build apps for any screen [online]. Dostupno na: //flutter.dev/. [28.7.2024.].

[7] Dart programming language | Dart [online]. Dostupno na: https://dart.dev/. [28.7.2024.].

[8] Firebase | Google’s Mobile and Web App Development Platform [online]. Dostupno na: https://firebase.google.com/. [29.7.2024.].

[9] Cloud Firestore Store and sync app data at global scale [online]. Dostupno na: https://firebase.google.com/products/firestore. [30.8.2024.].

[10] Firebase Authentication | Simple, multi-platform sign-in [online]. Dostupno na: https://firebase.google.com/products/auth. [30.8.2024.].

[11] A., Goodman, Flutter & Hive Database: CRUD Example (updated) [online], 28-pros-2021. Dostupno na: https://www.kindacode.com/article/flutter-hive-database/. [28.7.2024.].

[13] What is Stack Data Structure? A Complete Tutorial - GeeksforGeeks [online]. Dostupno na: https://www.geeksforgeeks.org/introduction-to-stack-data-structure-and-algorithm-tutorials/. [7.7.2024.].

[14] Harnessing the Power of the MVVM Pattern in Flutter [online]. Dostupno na: https://www.dhiwise.com/post/architecting-flutter-apps-using-the-mvvm-pattern. [31.8.2024.].

# SAŽETAK

Aplikacije za učenje engleskog jezika postaju sve popularnije zbog svoje sposobnosti da korisnicima omoguće fleksibilno i interaktivno učenje jezika. Mnoge od njih kombiniraju učenje vokabulara i gramatike, što pomaže korisnicima da brzo i učinkovito savladaju jezik. Cilj je ovog završnog rada bio razviti mobilnu aplikaciju za samostalno učenje kondicionalnih rečenica u engleskom jeziku. Nakon provedenog istraživanja postojećih aplikacija, utvrđeno je da postoji manjak aplikacija koje se specifično bave ovom temom. Aplikacija je razvijena korištenjem Flutter okruženja i Dart programskog jezika, osiguravajući dostupnost na Android i iOS platformama. Korištene su Firebase i Hive tehnologije za pohranu podataka, omogućujući korisnicima siguran pristup i sinkronizaciju podataka. Aplikacija nudi interaktivno korisničko sučelje koje omogućuje personalizirano učenje i praćenje napretka kroz različite tipove kondicionalnih rečenica. Namijenjena je studentima i pojedincima koji žele poboljšati svoje razumijevanje kondicionalnih rečenica u engleskom jeziku. U budućnosti, aplikacija se može nadograditi dodavanjem novih lekcija, poboljšanjem dizajna i integracijom naprednijih alata za analizu napretka korisnika.

**Ključne riječi:** engleski jezik, interaktivno učenje, kondicionalne rečenice, mobilna aplikacija

# ABSTRACT

**Title: Application for Learning Conditional Sentences**

English language learning applications have become increasingly popular due to their ability to provide users with flexible and interactive language learning. Many of them combine vocabulary and grammar learning thus helping users to quickly and effectively master the language. This final paper aimed to develop a mobile application for self-learning of conditional sentences in English. After conducting research on existing applications, it was found that there is a lack of apps for conditional sentences. The application was developed using the Flutter framework and Dart programming language, ensuring availability on both Android and iOS platforms. Firebase and Hive technologies were used for data storage, providing users with secure access and data synchronization. The application offers an interactive user interface that allows personalized learning and progress tracking through various types of conditional sentences. It is intended for students and individuals who wish to improve their understanding of conditional sentences in English. In the future, the application can be enhanced by adding new lessons, improving the design and integrating more advanced tools for progress analysis.



**Keywords:** English language, interactive learning, conditional sentences, mobile application