Razvoj aplikacije „MyLingo“ za učenje stranog jezika

Sliško, Mario

Sveučilište u Rijeci – Odjel za informatiku

Preddiplomski jednopredmetni studij informatike

Mario Sliško

Razvoj aplikacije „MyLingo“ za učenje stranog jezika

Završni rad

Mentor: doc. dr.sc Marija Brkić Bakarić

Rijeka, Rujan, 2020.

Sadržaj

Sažetak

U radu je prikazan proces izrade mobilne aplikacije. U uvodu je naveden opis tehnologije za izradu mobilnih aplikacija, a potom je predstavljen operacijski sustav za koji je razvijena aplikacija. Objašnjeni su svi alati koji su korišteni te je opisana sva funkcionalnost aplikacije. Detaljno je prikazan postupak izrade mobilne aplikacije u Flutter alatu. Svrha aplikacije je samovrednovanje poznavanja stranog jezika.

Ključne riječi: mobilna aplikacija, višeplatformski razvoj mobilnih aplikacija, iOS, Flutter, Dart,

1. Uvod

Ne tako davno mobilni telefoni su bili korišteni isključivo za komunikaciju. Mobitel (skraćeno prema mobilni telefon) je prenosivi telefon koji omogućuje glasovno, pismeno i u novije vrijeme slikovno komuniciranje [1]. Primarna funkcionalnost mobitela je bila glasovni poziv. Kasnije je omogućeno slanje SMS (Short Message Service) te MMS (Multimedia Messaging Service) poruka. Naglim razvojem mobilnih tehnologija, raste broj značajki koje su nudili mobiteli. To je naznačilo nastanak mobilnih aplikacija. Mobilna aplikacija je program namijenjen korištenju na mobilnim uređajima, tabletima, pametnim satovima i raznim drugim uređajima. U 1997. godini, Nokia 6110 je sadržavala ugrađenu verziju arkadne igre „Snake“, koja se može smatrati prvom mobilnom aplikacijom [2] (Slika 1).

Slika 1:Nokia 6110 Snake

Zahvaljujući dostupnosti mnogobrojnih alata za razvoj aplikacija, te velikoj potražnji od strane korisnika, naglo raste broj aplikacija dostupnih korisnicima. Izrađuju se za svakakve namjene i potrebe. Danas se aplikacije mogu svrstati u 3 skupine:

1. Izvorne aplikacije (engl. Native apps)
2. Hibridne aplikacije (engl. Hybrid apps)
3. Web aplikacije (engl. Web apps) [3].

Izvorne aplikacije su aplikacije koje su izrađene isključivo za jedan operacijski sustav, odnosno aplikacije koje su izrađene za iOS ne mogu se koristiti na Androidu. Prednosti izvornih aplikacija su visoke performanse. Glavna mana je veliki trošak izrade aplikacije za svaki operacijski sustav posebno. Hibridne aplikacije su kompletna suprotnost od izvornih aplikacija. Njihova glavna karakteristika je da se mogu koristiti na više operacijskih sustava. Služi se kombiniranjem elemenata izvornih i web aplikacija. Hibridne aplikacije u osnovi su web aplikacije koje su stavljene u ljusku izvorne aplikacije [4]. Web aplikacije koriste internetski preglednik za prikaz sadržaja. Najčešće su pisane u HTML5, JavaScript i CSS programskim jezicima [5].

2. Operacijski sustav iOS

iOS je naziv za operacijski sustav mobilnih uređaja tvrtke Apple. Predstavljen je 2007. godine na iPhone-u prve generacije (iPhone 1). Ubrzo se proširio na uređaje kao što su iPod Touch, iPad, te u novije vrijeme Apple Watch i Apple TV. Svi nabrojani uređaji se izrazito razlikuju u veličini ekrana, stoga je Apple odlučio napraviti posebne inačice operacijskih sustava za svaki od uređaja. Da bi se mogao iskoristiti pun potencijal velikog ekrana na iPad-u, Apple predstavlja inačicu iOS 13 verzije, iPadOS 13 specijalno dizajniranu za Apple-ove tablet uređaje. Uz to postoje i watchOS te tvOS operacijski sustavi. Početna zamisao Apple-a je bila da se aplikacije razvijaju kao web aplikacije za preglednik Safari, odnosno Apple nije htio dopustiti razvoj aplikacija od strane vanjskih developera [13]. S obzirom na veliko protivljenje raznih developera, Apple je odlučio predstaviti App Store. App Store je distribucijska platforma gdje su pohranjene sve aplikacije dostupne korisnicima za instalaciju. Također je najavljen dolazak Apple-ovog SDK-a (alat za razvijanje softvera), koji će omogućiti vanjskim developerima razvijanje aplikacija za App Store. Danas postoji preko 1.8 milijuna aplikacija na njihovoj trgovini [14]. iOS uređaji dolaze sa pred instaliranim aplikacijama kao što su kalkulator, Email, Safari, Apple Maps, FaceTime i mnoge druge. iOS je drugi najzastupljeniji operacijski sustav za mobilne uređaje pokrivajući 38.98%. Jedini operacijski sustav ispred njega je Android sa 51.59% [15]. Glavni razlog tome je Android-ova otvorenost za sve proizvođače mobilnih uređaja, za razliku od iOS uređaja koje proizvodi samo Apple. Android se aktivno koristi na više od 2 milijarde uređaja, dok se iOS aktivno koristi na 1.3 milijarde uređaja [16]. Još jedna od prednosti Androida je cijena. Android uređaji su pristupačni svima. Postoje uređaji za obične korisnike, koji ne zahtijevaju mnogo, čija je cijena prema tome manja. Također postoje i „flagship“ uređaji sa najnovijim značajkama, i naravno višom cijenom. Suprotno tome, Apple uređaji su namijenjeni za korisnike koji su spremni odvojiti veću svotu novaca za kupnju mobilnog uređaja. Nebitno da li to bio najbolji ili najlošiji model. Općenito su Apple uređaji poprilično zatvoreni, ne daju korisniku puno mogućnosti prilagođavanja uređaja. Usprkos svim nedostatcima, Apple i dalje uspijeva konkurirati Androidu te je u nekim pogledima bolji od njega. U tablici je prikazana usporedba iOS i Android operacijskih sustava.

Svojstva	**Android**	**iOS**
Razvijač	Google	Apple
Operacijski sustav	Linux	OS X, Unix
Widgeti	Da	Ne, osim u obavjestima
Programiran u	C, C++, java	C, C++, objektni-C
Komunikacija	Google Hangouts	iMessage
Internetski preglednik	Google Chrome	Safari
Dostupnost izvornog koda	Otvoreni softver	Zatvoreni, s komponentama otvorenog softvera

Tablica: Usporedba iOS-a i Androida [21]

3. Alati za razvoj aplikacije

3.1 Flutter

Flutter je Google-ov paket za razvoj programa koji služi za izradu aplikacija na raznim operacijskim sustavima. Prva verzija Fluttera , pod nazivom Sky, razvijena je 2015. godine s namjerom da aplikacije mogu prikazivati sliku na 120 fps-a (frames per second), što do tada, nije bilo moguće izvesti na uređajima sa ekranima od 60 Hz [7]. U početku je primarni operacijski sustav bio Android, no pri objavi nove verzije Flutter-a, poboljšava se izvođenje aplikacija na iOS uređajima za 50%. Glavna prednost Flutter-a je mogućnost razvoja aplikacija za većinu operacijskih sustava, pritom smanjujući troškove izrade aplikacija za developere. Ključne karakteristike Fluttera su:

1. Brz razvoj
2. Fleksibilno korisničko sučelje
3. Performanse izvornih aplikacija

Brz razvoj omogućuje Skia 2D grafika koju koriste Google Chrome i Android [8]. Flutter je dizajniran da ne prikazuje zastoje prilikom animacije iako se dogodi usporavanje aplikacije. Funkcija „hot reload“ omogućuje prikazivanje promjena napravljenih u kodu u realnom vremenu. Postoje 2 vrste widgeta dizajniranih za uređaje koji ih koriste. Material Design widgeti koriste Google-ov dizajn, Cupertino widgeti koriste iOS dizajn. Na slici 2 je prikazano korištenje Cupertino widgeta u aplikaciji MyLingo (Slika 2). Nakon što se korisnik registrira, automatski mu se šalje mail za potvrdu registracije. *CupertinoAlertDialog* funkcija prikazuje korisniku poruku da mu je poslan mail u obliku pop-up prozorčića.

```
showDialog(context: context,
  builder: (BuildContext contex) =>
      CupertinoAlertDialog(
        title: Text("Potvrda e-maila."),
        content: Text(
            "Poslan vam je mail za potvrdu prijave."),
      ),
);
```

Slika 2:Cupertino

3.2 Dart

Flutter aplikacije se pišu u programskom jeziku Dart. Dart je klijentu optimiziran deklarativan programski jezik za brze aplikacije na bilo kojoj platformi [6]. Koristi „Just In Time“ kompilaciju koja omogućuje „hot reload“. Također koristi „ahead-of-time“ kompilaciju, omogućujući Flutter aplikacijama visoke performanse na mobilnim uređajima. Navedeno je da je Dart deklarativan programski jezik, ali on se smatra jezikom koji spada u više skupina programskih paradigmi. Može se koristiti kao proceduralan, funkcionalan ili objektno-orijentiran, te se smatra da je objektno-orijentirani pristup najprirodniji [9]. Neke od prethodnih verzija su imale problem pri izvođenju koda. Trebao se izvoditi u pregledniku, a niti jedan standardni preglednik nije podržavao Dart. Taj problem se rješavao na 3 načina:

1. Prevođenje Dart koda u JavaScript
2. Izvođenje koda u Dartium pregledniku
3. Izvođenje koda pomoću posebne virtualne mašine Dart VM [10].

Osim za kreiranje mobilnih aplikacija, Dart se koristi i za izradu Web aplikacija. Jednostavnim kombiniranjem Dart i html elemenata omogućeno je kreiranje web stranica za svakojake namjene.

3.3 Android Studio

Android Studio je razvojni program namijenjen prvenstveno za izradu aplikacija na Android operacijskom sustavu. Prilikom izrade MyLingo aplikacije, Android Studio je korišten samo kao tekst editor. Da bi se Flutter mogao koristiti u Android Studiu, potrebno je instalirati Flutter plugin, što je moguće napraviti direktno iz Android Studia (Slika 3). Time je omogućeno razvijanje iOS aplikacija unutar njega. Android Studio je moguće instalirati na Windows, Linux i MacOS. Pri izradi aplikacije za Android operacijski sustav, moguće je testirati aplikacije pomoću virtualnog uređaja. Moguće je odabrati jedan od mobilnih uređaja u ponudi, te je potrebno odabrati operacijski sustav za koji će aplikacija biti namijenjena. Nažalost, proces za iOS nije toliko jednostavan. Ukoliko se želi testirati iOS aplikaciju, potrebno je koristiti Mac, jer taj proces nije moguće izvoditi na drugim operacijskim sustavima. Android Studio na macOS uređaju, koji ima instaliran Xcode, automatski prepoznaje iOS Simulator.

Slika 3:Android Studio + Flutter

3.4 Xcode

Xcode je razvojno okruženje za macOS koje služi za izradu aplikacija za sve Apple uređaje. Osim izrade aplikacije, Xcode je neophodan za optimiziranje, testiranje i objavljivanje aplikacije na App Store [11]. Bez Xcode-a testiranje MyLingo aplikacije ne bi bilo moguće. Xcode je potrebno instalirati na macOS uređaj, koji se koristi za razvoj aplikacije, putem AppStore-a. Programski jezici koje podržava: Swift, Objektni-C, C, C++, Java, AppleScript, Python i još mnogo drugih [12]. Da bi se iOS aplikacija objavila na App Store potrebno je unutar Xcode-a postaviti Bundle ID, koji jedinstveno predstavlja aplikaciju. Također je potrebno imati Apple developer račun, te kreirati ili odabrati tim, ukoliko nemamo tim, moguće je odabrati osobni tim (samo jedna osoba).

3.4.1 Simulator

Simulator je aplikacija koja se prikazuje na računalu kreirajući virtualnu mašinu koja predstavlja određeni mobilni uređaj. Za aplikaciju MyLingo, korišten je simulator mobilnog uređaja iPhone 11. Prilikom instaliranja Xcode programa, automatski se instalira simulator alat. Simulator je neophodan alat za testiranje iOS aplikacija, ukoliko nema fizičkog uređaja za testiranje. Pri pokretanju nude se svi Apple mobilni uređaji, različitih veličina zaslona, pa je poželjno testiranje aplikacije na više uređaja. Simulator ima samo neke od funkcionalnosti pravog uređaja, odnosno samo one funkcionalnosti koje su potrebne za testiranje aplikacija.

3.5 Firebase

Firebase je Google-ova platforma za izradu mobilnih i web aplikacija. Firebase je bila zasebna tvrtka do 2014. godine kada ih je kupio Google [17]. Nastala je od kompanije Envolve, koja je pružala uslugu integracije online razgovora na web aplikacijama. Usprkos njenoj namjeni, developeri su koristili Envolve kako bi pohranjivali podatke koji su bili više od poruka u razgovoru. To je navelo Envolve tim da kreiraju Firebase. Sa Firebase-om se može :

1. Izrađivati bolje aplikacije
2. Poboljšati kvalitetu aplikacije
3. Uzorkovati rast poslovanja [18].

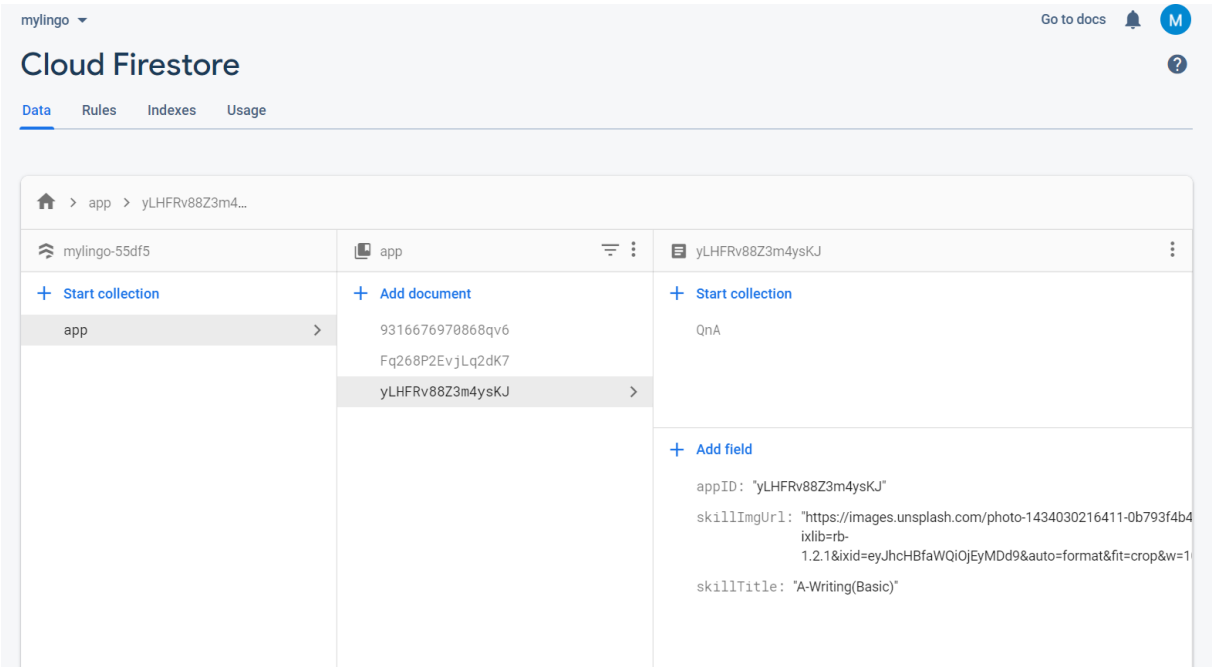
Firebase platforma se sastoji od 18 proizvoda koji su podijeljeni u 3 grupe: razvoj, kvaliteta i rast. Prvi proizvod za razvoj aplikacija je Realtime Database (baza podataka u realnom vremenu). Podaci unutar baze su sinkronizirani u realnom vremenu i ostaju dostupni korisnicima kada aplikacija nema vezu s internetom. Da bi se osigurala sigurnost aplikacije, moguće je navesti sigurnosna pravila koja određuju koji korisnici mogu pristupiti određenim podacima. Google je optužen da prati korisnike bez njihovog znanja pomoću Firebase-a. Podignuta je tužba u srpnju 2020. godine u kojoj glasi da Google sprema podatke, i vidi šta korisnici gledaju, iako su korisnici isključili tu mogućnost u aplikaciji [19]. Za izradu MyLingo aplikacije korišten je Firebase-ov Cloud Firestore.

3.5.1 Cloud Firestore

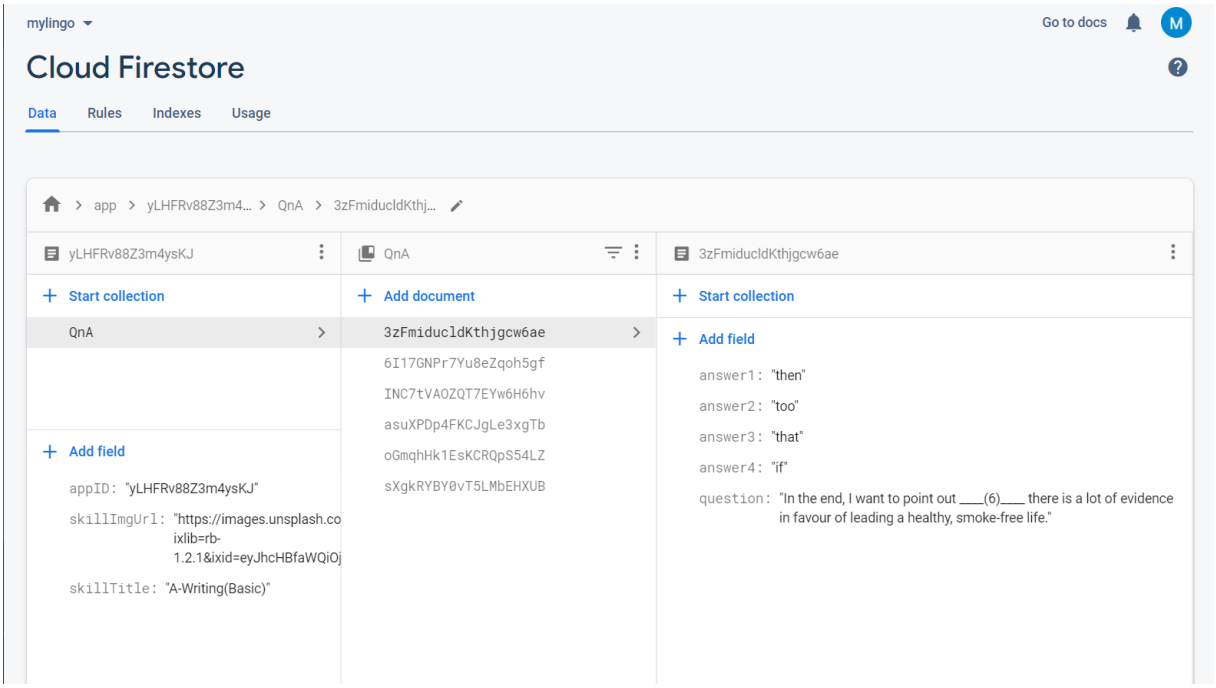
Cloud Firestore je NoSql baza podataka koja omogućuje jednostavno pohranjivanje, sinkroniziranje i dohvaćanje podataka za mobilne i web aplikacije [20]. Kao i Realtime Database sinkronizira podatke u realnom vremenu, i omogućuje offline podršku. Glavne mogućnosti Cloud Firestore-a:

- Fleksibilnost
- Ažuriranja u realnom vremenu
- Offline podrška

Na slici 4 je prikazana aplikacija MyLingo unutar Cloud Firestore-a. Kolekcija „app“ je glavna kolekcija unutar koje se nalaze svi podaci (Slika 4). Sastoji se od 3 dokumenta koji predstavljaju vještine i razine koji se ispituju u aplikaciji. Unutar svakog dokumenta se nalazi podkolekcija „QnA“ koja se sastoji od dokumenata u kojima su zapisana pitanja i odgovori (slika 5).



Slika 4:Cloud Firestore (1)



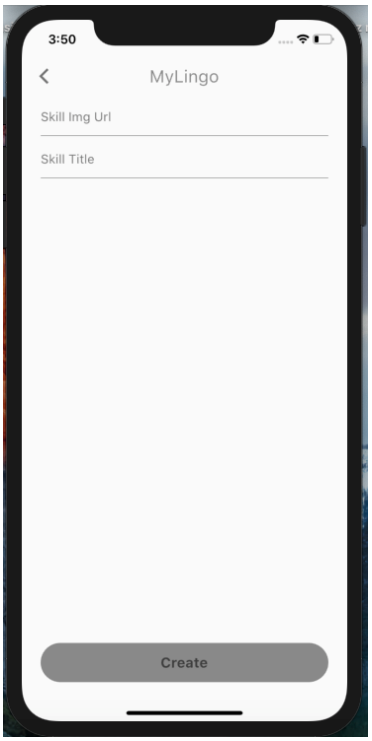
Slika 5:Cloud Firestore (2)

4. Opis aplikacije MyLingo

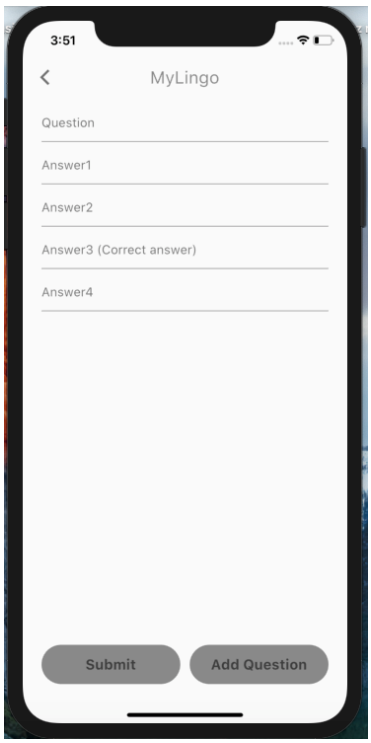
Aplikacija MyLingo je izrađena u Flutter alatu koji je pisan u programskom jeziku Dart. Pri pokretanju aplikacije prikazuje se obrazac za prijavu koji od korisnika traži da unese e-mail i lozinku. Ukoliko korisnik nema račun, može se prebaciti na obrazac za registraciju u koji se unosi ime, prezime, korisničko ime, e-mail i lozinka. Automatski se vrši provjera da li je upisani e-mail pravilnog formata te da li je lozinka dulja od 6 znakova. Nakon izvršene registracije, šalje se mail za potvrdu registracije. Prije potvrde se nije moguće uspješno prijaviti u aplikaciju. U slučaju da je korisnik zaboravio lozinku, ima mogućnost resetiranja lozinke. Otvara se obrazac u koji je potrebno unijeti email, na koji je onda poslan link za resetiranje lozinke. Prilikom prijave provjerava se je li korisnik koji se prijavljuje administrator. Ukoliko jest administrator, tada on ima mogućnost kreiranja novih zadataka u aplikaciji. Pritiskom na gumb „+“ otvara se obrazac za unos URL-a slike, koja će služiti kao pozadina vještine, i naslov vještine. Ukoliko je unesen ispravan email, otvara se dodatni prozor u kojem se unosi pitanje i ponuđeni odgovori. Aplikacije je postavljena tako da postoji samo jedan administrator, a svi novi korisnici koji se registriraju imaju ulogu običnog korisnika. Na početnom zaslonu aplikacije, prikazuju se vještine i određene razine za vještinu. Pritiskom na jednu od vještina otvara se prozor sa svim pitanjima i ponuđenim odgovorima. Ukoliko je odabran odgovor točan, gumb tog odgovora će postati zelen, u suprotnom će postati crven. Redoslijed odgovora se prilikom svakog otvaranja mijenja. Nakon odgovaranja na pitanja, klikom na gumb, može se provjeriti statistika točnosti odgovorenih pitanja. Gumbom nazad, korisnik se može vratiti na početni zaslon. U gornjem desnom kutu se nalazi „Logout“ gumb s kojim se korisnik odjavljuje iz aplikacije. Ukoliko korisnik zatvori aplikaciju, bez odjave, aplikacija će zapamtiti korisnikovu prijavu. Na slikama 6 do 11 prikazan je izgled aplikacije (Slika 6 – Slika 11).

Slika 6:Obrazac za registraciju

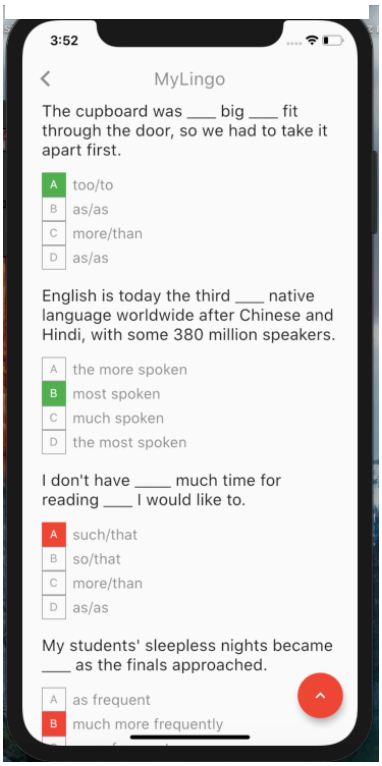
Slika 7:Obrazac za prijavu



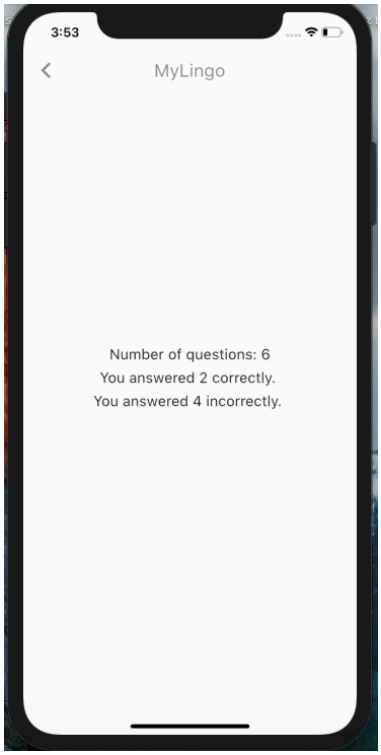
Slika 8:Obrazac za unos testa



Slika 10:Obrazac za unos pitanja



Slika 9:Rješavanje testa



Slika 11:Statistika

5. Izrada aplikacije MyLingo

Za izradu aplikacije korišteni su: Andorid Studio, Flutter plugin, Xcode, Firebase (Cloud Firestore). U Android Studiu je potrebno instalirati Flutter plugin. Također je potrebno s Macbook-ovog App Store-a skinuti i instalirati Xcode. Pomoću Xcode možemo postaviti Iphone simulator koji je korišten za testiranje aplikacije. Aplikacije se također mogu testirati na pravom uređaju. Pokretanjem naredbe „flutter docotor“ u terminalu, možemo provjeriti da li je flutter uspješno instaliran na računalo. Ukoliko provjera prođe uspješno, sve je spremno za početak izrade aplikacije.

5.1 Izrada obrazaca za unos podataka

Obrazac za unos podataka se sastoji od više *TextFormField*-ova u koje se unose traženi podaci. U body dijelu koda se prvo provjerava stanje bool varijable *_isLoading* (Slika 12). Ukoliko je varijabla postavljena na „true“ prikazati će se simbol za učitavanje. U suprotnom se prikazuje obrazac u kojem je uključena automatska validacija koja služi za provjeru ispravnosti unesenih podataka. Unutar *TextFormField*-a se poziva *validator* funkcija koja u ovom slučaju vraća tekst „Unesite ime:“ ukoliko nije unesen niti jedan znak u obrazac za unos imena. Prikaz slike vrši se unošenjem URL-a slike i postavljanjem zadovoljavajuće veličine za naš uređaj.

```
body: _isLoading
    ? Container(
        child: Center(
          child: CircularProgressIndicator(),
        ), // Center
      ) // Container
    : Form(
        autovalidate: true,
        key: _formKey,
        child: Container(
          padding: EdgeInsets.symmetric(horizontal: 24),
          child: Column(
            children: [
              SizedBox(height: 20),
              Image.network("https://img.freepik.com/free-vector/illustration-language-concept_53876-20610.jpg?size=626&ext=jpg",
                  width: MediaQuery.of(context).size.width,fit:BoxFit.cover), // Image.network
              SizedBox(height: 20),
              TextFormField(
                validator: (val) {
                  return val.isEmpty ? "Unesite ime: ." : null;
                },
                decoration: InputDecoration(labelText: "Ime"),
                onChanged: (val) {
                  firstName = val;
                },
              ), // TextFormField
```

Slika 12: Obrazac za unos (signup.dart)

Pritiskom na gumb „Registrirajte se“ poziva se funkcija *signUp* (Slika 13).

```
final _formKey = GlobalKey<FormState>();
String email, password, firstName, lastName, username;
AuthService authService = new AuthService();
bool _isLoading = false;

signUp() async {
  if (_formKey.currentState.validate()) {
    setState(() {
      _isLoading = true;
    });
    await authService.signUp(username,email, password).then((value) {
      if (value != null) {
        setState(() {
          _isLoading = false;
        });
        Hfunctions.saveLogin(isLoggedin: true);
        Navigator.pushReplacement(
            context, MaterialPageRoute(builder: (context) => SignIn()));
      }
    });
  }
}
```

Slika 13: funkcija signUp()

Slika 13 prikazuje asinkronu funkciju *signUp* koja postavlja stanje bool varijable *_isLoading* na vrijednost „true“ ukoliko je prošla validacija. Poziva se funkcija *signUp* iz klase *AuthService* i čeka njena povratna vrijednost (Slika 14). Asinkrone funkcije nam omogućuju da se izvrše druge operacije prije nego se one završe. Ako „*authService.signUp*“ ima povratnu vrijednost, varijabla *_isLoading* se postavlja na „true“, poziva se funkcija *saveLogin* iz klase *Hfunctions*, te se prikaz prebacuje na obrazac za prijavu (Slika 15).

```
Future<String> signUp(String username, String email, String password) async{
  FirebaseUser user= (await _auth.createUserWithEmailAndPassword
    (email: email, password: password)).user;

  try{
    await user.sendEmailVerification();
    return user.uid;
  }catch(e){
    print(e.toString());
  }
}
```

Slika 14: AuthService.signUp

```
  static saveLogin({@required bool isLoggedin}) async{
    SharedPreferences prefs = await SharedPreferences.getInstance();
    prefs.setBool(userLoggedInKey, isLoggedin);
  }
```

Slika 15:Hfunctions.saveLogin

5.2 Prikaz ponuđenih vještina

Prikaz vještina se može smatrati početnim zaslonom aplikacije. U glavnom djelu koda prvo definiramo da se ekran može pomicati (engl. scroll). Kreiranjem toka (engl. stream) sve promjene koje se dogode u bazi podataka će direktno utjecati na promjene u toku. Ukoliko nema podataka u toku prikazuje se simbol za učitavanje. U suprotnom se kreira *ListView*, koji prikazuje vještine na način da poziva klasu *skillTile* i zadaje joj tražene vrijednosti (Slika 16). *SkillTile* je klasa u kojoj se definira akcija pritiskom na jednu od ponuđenih vještina, te se određuje izgled „pločice“ koja predstavlja određenu vještinu.

```
body: SingleChildScrollView(
  child: Container(
    height: 1440,
    padding: EdgeInsets.symmetric(horizontal: 24),
    child: Column(
      children: [
      StreamBuilder(
          stream: skillStream,
          builder: (context, snapshot){
        ------ if(snapshot.data == null) {return CircularProgressIndicator();}
          else
            {return
        ---------- Expanded(
              child: ListView.builder(
                itemCount: snapshot.data.documents.length,
                itemBuilder: (context, index){
                  return skillTile(
                    appID: snapshot.data.documents[index].data["appID"],
                    imgUrl: snapshot.data.documents[index].data["skillImgUrl"],
                    title: snapshot.data.documents[index].data["skillTitle"],
                  ); // skillTile
```

Slika 16: Tok (stream)

Prije prikaza vještina, poziva se *initState* funkcija u kojoj se poziva *getAppData* funkcija iz *DBService* klase (Slika 17). Funkcija *getAppData* se povezuje s Cloud Firestore bazom podataka i dohvaća vrijednosti koje se nalaza u „app“ kolekciji (Slika 18).

```
@override
void initState() {
  //print("${widget.appID}");
  dbService.getAppData().then((val){
    setState(() {
      skillStream = val;
    });
  });
  currentUser();
  super.initState();
}
```

Slika 17: initState

```
getAppData() async{
  return Firestore
      .instance
      .collection("app")
      .snapshots();
}
```

Slika 18:getAppData

InitState funkcija također poziva *currentUser* funkciju (Slika 17) pomoću koje se vrši provjera da li je prijavljeni korisnik *admin*. Ukoliko je user ID jednak administratorovom, vrijednost varijable *_admin* se postavlja na „true“, te se korisniku omogućuje unos novih zadataka.

5.3 Prikaz pitanja i ponuđenih odgovora

Pritiskom na određenu vještinu navigator nas prebacuje na prikaz pitanja i odgovora. Prvo se pokreće *initState* funkcija koja postavlja vrijednost varijable *questionSnapshot*. U tijelu Widgeta se postavlja da se ekran može pomicati i provjerava se vrijednost varijable *questionSnapshot* postavljene od strane funkcije *initState*. Ukoliko vrijednost nije null, kreira se *ListView* u kojem će se prikazivati pitanja i odgovori, pozivajući klasu *QuestionTile*. U *QuestionTile* klasi se prikazuje pitanje i svi ponuđeni odgovori. Odabirom odgovora se provjerava da li je on točan, ukoliko jest, vrijednost varijable *_correct* se povećava za 1, u suprotnom se vrijednost varijable *_incorrect* povećava za 1 (Slika 19).

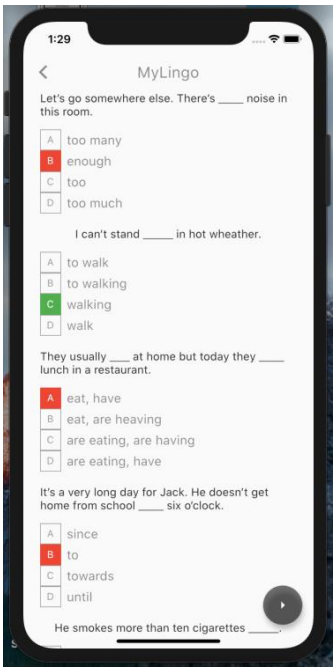
```
GestureDetector(
  onTap: (){
    if(!widget.questionModel.answered){
      if(widget.questionModel.answer1 == widget.questionModel.correctAnswer){
        selectedAnswer = widget.questionModel.answer1;
        widget.questionModel.answered=true;
        _correct = _correct + 1;
        setState(() {
        });
      }
      else{
        selectedAnswer = widget.questionModel.answer1;
        widget.questionModel.answered=true;
        _incorrect = _incorrect + 1;
        setState(() {
        });
      }
    }
  },
  child: AnswerTile(
    correctAnswer: widget.questionModel.correctAnswer,
    answerText: widget.questionModel.answer1,
    answers: "A",
    selectedAnswer: selectedAnswer,
  ), // AnswerTile
), // GestureDetector
```

Slika 19:QuestionTile

Nakon provjere se poziva widget *AnswerTile* u kojem se također vrši provjera da li je odabrani odgovor točan (Slika 20). Ukoliko je, odgovor poprima zelenu boju. Ako je odabrani odgovor netočan, poprima crvenu boju (Slika 21).

```
29          decoration: BoxDecoration(
30            border: Border.all(
31              color: widget.answerText == widget.selectedAnswer ?
32              (widget.selectedAnswer == widget.correctAnswer
33                  ? Colors.green : Colors.red)
34                  : Colors.grey,
35            ), // Border.all
36            color: widget.answerText == widget.selectedAnswer ?
37            (widget.selectedAnswer == widget.correctAnswer
38                ? Colors.green : Colors.red)
39                : Colors.transparent,
40          borderRadius: BorderRadius.horizontal()
41          ), // BoxDecoration
42
43        ), // Container
44
```

Slika 20:AnswerTile



Slika 21:iOS Simulator

5.4 AppBar i logout

Naziv aplikacije „MyLingo“ je prikazan na svakom ekranu. Napravljen je kao zasebni widget koji se može pozivati iz svake datoteke, uz uvjet da se uveze njegov paket (Slika 22).

```
Widget appBar(BuildContext context) {
  return RichText(
      text: TextSpan(
        style: TextStyle(fontSize: 22),
        children: <TextSpan>[
          TextSpan(
              text: 'MyLingo',
              style: TextStyle(fontWeight: FontWeight.w400, color: Colors.black45)),
        ], // <TextSpan>[]
      ), // TextSpan
    ); // RichText
}
```

Slika 22:appBar

Nakon uspješne prijave, na appBar-u se pojavljuje gumb „Logout“. Pritiskom na tipku, poziva se funkcija *signout*, korisnika se uspješno odjavljuje iz aplikacije, te se mora ponovno prijaviti ukoliko želi rješavati neke od zadataka. Funkcija *signout* poziva iz klase *AuthService* funkciju *signOut* te postavlja vrijednost *bool* varijable *isLoggedin* na „false“ unutar funkcije *saveLogin*

(Slika 23). Time se poništava vrijednost koja je bila zadana prilikom prijave, te nas vraća na obrazac za ponovnu prijavu.

```
void signout()async {
  await authService.signOut().then((value) {
    Hfunctions.saveLogin(isLoggedin: false);
    Future.delayed(Duration.zero, () {
      Navigator.pushReplacement(
          context, MaterialPageRoute(builder: (contex) => SignIn()));
    }); // Future.delayed
  });
}
```

Slika 23:signout funkcija

5.5 Unos novih zadataka

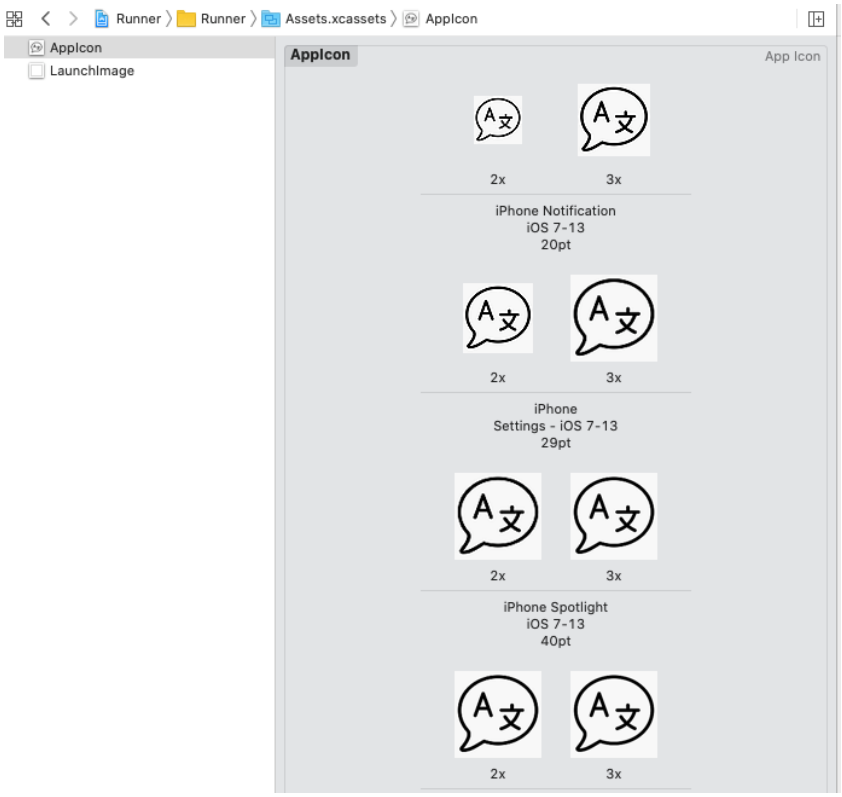
Ukoliko je prijavljeni korisnik administrator, imat će mogućnost kreiranja novih zadataka za rješavanje. Pritiskom na gumb „+“ otvara se obrazac u koji se unose podaci za vještinu. Pritiskom na gumb „Create“ poziva se funkcija *createInApp*. Nakon validacije unesenih podataka, postavlja se vrijednost bool varijable *_isLoading* na „true“. Varijabla *appID*, koja predstavlja Document ID u Cloud Firestore-u, dobiva nasumičnu alfanumeričku vrijednost. *appMap* se sastoji od key->value parova koji su tipa string. Poziva se funkcija *addAppData* iz *dbService* klase koja postavlja vrijednost *_isLoading* varijable na „false“, te prebacuje korisnika na novi prikaz u koji se unose pitanja i odgovori (slika 24).

```
19  createInApp() async {
20    if (_formKey.currentState.validate()) {
21      setState(() {
22        _isLoading = true;
23      });
24
25      appID = randomAlphaNumeric(16);
26      Map<String, String> appMap = {
27        "appID": appID,
28        "skillImgUrl": skillImageUrl,
29        "skillTitle": skillTitle,
30      };
31      await dbService.addAppData(appMap, appID).then((value) {
32        setState(() {
33          _isLoading = false;
34          Navigator.pushReplacement(
35              context, MaterialPageRoute(builder: (context) => AddQuestion(appID)));
36        });
37      });
38    }
39  }
```

Slika 24:createInApp()

5.6 Ikona aplikacije

Ikona aplikacije je jedan od najbitnijih aspekata svake mobilne aplikacije. Ona je prva stvar koju korisnik vidi i po čemu pamti aplikaciju. Svaka Flutter aplikacija ima zadanu ikonu u obliku Flutter simbola. Moguće je tu ikonu promijeniti s personaliziranom ikonom koja će jedinstveno prikazivati aplikaciju. Za kreaciju MyLingo ikone korišten je online program App Icon Generator. Prvi korak je odabir slike koja će činiti ikonu, te odabrati za koje je uređaje namijenjena. App Icon Generator tada generira set od 11 slika različitih dimenzija. Svaka slika predstavlja funkcionalnost za koju će biti korištena (slika 25).



Slika 25: Umetanje ikone

6. Zaključak

Prilikom izrade završnog rada stekao sam osnovno znanje razvoja aplikacija za mobilne uređaje. Znatno mi je olakšano korištenje Flutter alata i Dart programskog jezika zahvaljujući raznoj dokumentaciji i online video lekcijama. Iako nisam imao prijašnjeg iskustva s Dart-om, poznavanje jezika C#, te općenito C programskih jezika, pomoglo mi je pri učenju Dart sintakse. Znajući za popularnost Apple proizvoda na tržištu, smatram da bi mi stečeno znanje razvoja aplikacija za njihove uređaje moglo koristiti u budućnosti. Također bi mi pomoglo pri razvoju aplikacija za druge operacijske sustave.

Svrha MyLingo aplikacije je samovrednovanje poznavanja stranog jezika. Osim potrebe za popunjavanjem baze, aplikaciju je moguće nadograditi na razne načine. Moguće je dodati druge načine registracije korisnika (npr. registracija putem Google ili Facebook računa). Također se može dodati prijava korisnika otiskom prista ili skeniranjem lica, ukoliko uređaj ima te mogućnosti. Još jedna od funkcionalnosti koja bi dodatno poboljšala aplikaciju je praćenje napretka korisnika, i prikazivanje pitanja ovisno o njegovom napretku.

7. Literatura

1. Mobitel
   http://enciklopedija.lzmk.hr/clanak.aspx?id=26055 pristupljeno 4.9.2020.
2. Prva mobilna aplikacija
   https://inventionland.com/inventing/the-history-of-mobile-apps/ pristupljeno 4.9.2020.
3. Podjela aplikacija
   https://thinkmobiles.com/blog/popular-types-of-apps/ pristupljeno 4.9.2020.
4. Hibridne aplikacije
   https://searchsoftwarequality.techtarget.com/definition/hybrid-application-hybrid-app pristupljeno 4.9.2020
5. Web aplikacije
   https://thinkmobiles.com/blog/popular-types-of-apps/ pristupljeno 4.9.2020.
6. Dart
   „https://dart.dev/“ pristupljeno 4.9.2020.
7. Sky
   https://arstechnica.com/gadgets/2015/05/googles-dart-language-on-android-aims-for-java-free-120-fps-apps/ pristupljeno 5.9.2020.
8. Skia
   https://github.com/flutter/flutter pristupljeno 6.8.2020.
9. Dart paradigma
   David Kopec(2014). *Dart for Absolute Beginners*, pristupljeno 7.9.2020.
10. Izvođenje dart programskog koda
    https://m.vidi.hr/index.php/racunala/novosti/google-dart-2-x pristupljeno 7.9.2020.
11. Xcode
    https://developer.apple.com/library/archive/documentation/ToolsLanguages/Conceptual/Xcode_Overview/ pristupljeno 7.9.2020.
12. Xcode – programski jezici
    https://www.quora.com/Which-programming-languages-does-Xcode-support pristupljeno 7.9.2020.
13. Safari web aplikacije
    https://9to5mac.com/2011/10/21/jobs-original-vision-for-the-iphone-no-third-party-native-apps/ pristupljeno 8.9.2020.
14. App Store aplikacije
    https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/ pristupljeno 8.9.2020.
15. iOS zastupljenost na tržištu
    https://web.archive.org/web/20150602171818/https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1 pristupljeno 8.9.2020.
16. Aktivni korisnici iOS-a i Androida
    https://pcchip.hr/softver/must-have/android-vs-ios-operativni-sustav/ pristupljeno 8.9.2020.
17. Google kupuje Firebase

https://techcrunch.com/2014/10/21/google-acquires-firebase-to-help-developers-build-better-realtime-apps/?guccounter=1&guce_referrer=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnLw&guce_referrer_sig=AQAAAG0lHZOgnJspLwhCL1plH5xuBJGan3qTWrMlYol5qS4qX4dsArGc5Hgkjk68eTbkkqMLYMvezlzR9fn0NbDRVWbe2npCy4xE8UDRHCCgJnX-UIDFd7SLaMaxXN8h5u8TGDbyOwjPDHcDbWD6lYMumiVgxf3Ht-f4cJ6ywQUFJh1K
pristupljeno 6.9.202

18. Firebase značajke
https://medium.com/codingurukul/introduction-to-firebase-f9f6ccc8a785 pristupljeno 6.9.2020.
19. Google Firebase tužba
https://www.reuters.com/article/us-alphabet-google-privacy-lawsuit-idUSKCN24F2N4
pristupljeno 6.9.2020.
20. Cloud Firestore
https://firebase.google.com/products/firestore pristupljeno 6.9.2020.
21. Tablica usporedbe iOS-a i Androida
https://www.researchgate.net/figure/Differences-between-Android-and-IOS_tbl1_329736359 pristupljeno 16.9.2020

8. Popis slika

9. Prilozi

Uz rad priložena je mobilna aplikacija MyLingo.