Simulacija igre kaladont

Horvat, Kristijan

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

Sveučilišni studij

SIMULACIJA IGRE KALADONT

Završni rad

Kristijan Horvat

Osijek, 2023.

SADRŽAJ

1. UVOD

Kaladont igra je popularna društvena igra koja zahtjeva kreativnost, brzinu razmišljanja i jezične vještine. Cilj igre je nastaviti riječ na posljednja dva slova prethodne riječi. U današnje vrijeme imamo mnogo različitih operacijskih sustava, te je potrebno pronaći rješenje kako bi se na što jednostavniji način to prebrodilo. Jedno rješenje je korištenje multiplatformskog sučelja kao što je Flutter. Glavni problem u igri je veliki broj riječi koje treba efikasno pretraživati i provjeravati njihovu točnost. U drugom poglavlju opisana su slična rješenja ovakvog problema, u trećem poglavlju opisani su alati korišteni za izradu Flutter aplikacije, četvrto poglavlje opisuje razvoj igre i korištene algoritme te peto poglavlje je zaključak rada.

1.1. Zadatak završnog rada

Izraditi aplikaciju koja će omogućiti igranje popularne igre kaladont. Potrebno je koristiti što potpuniji rječnik hrvatskih riječi, koji će poslužiti za provjeru ispravnosti korištenih riječi. Omogućiti provjeru koristeći brze algoritme za pretragu. Simulacija treba imati više težinskih razina igranja. Također, ponuditi i mogućnost predlaganja novih riječi ukoliko one u dotadašnjem rječniku ne postoje.

2. PRIJEDLOG RJEŠENJA

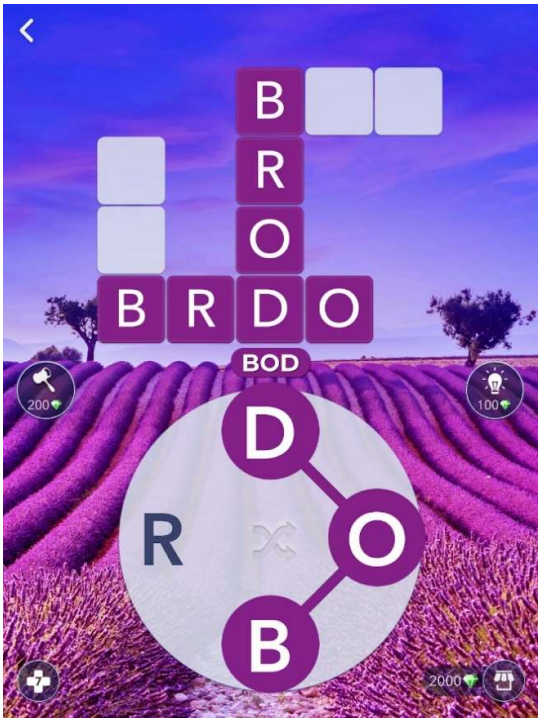
Aplikacija se sastoji od četiri zaslona, gdje prvi ima izbornik i naslovnicu igre u izborniku se može odabrati igranje igre, opis igre i postavke. Nakon klika na gumb za igru, korisniku se otvara zaslon igre na kojem je brojač bodova, riječ za igru, polje za unos riječi, gumb za potvrdu unesene riječi i tipkovnica. Klikom na gumb za opis igre, otvora se zaslon s opisom igranja igre te klikom na gumb za postavke otvara se zaslon s padajućim izbornikom za odabir težine igranja i ispod padajućeg izbornika nalazi se gumb za prijedlog novih riječi.

2.1. Opis postojećih rješenja

Postoje razne slične igre u trgovini aplikacijama, no igra Kaladonta na hrvatskom jeziku ne postoji. Postoje igre sa slaganjem riječi i pogađanjem riječi te razne križaljke. Neke koje su najpopularnije biti će prikazane u nastavku.

2.1.1. Riječi čuda: Križaljka

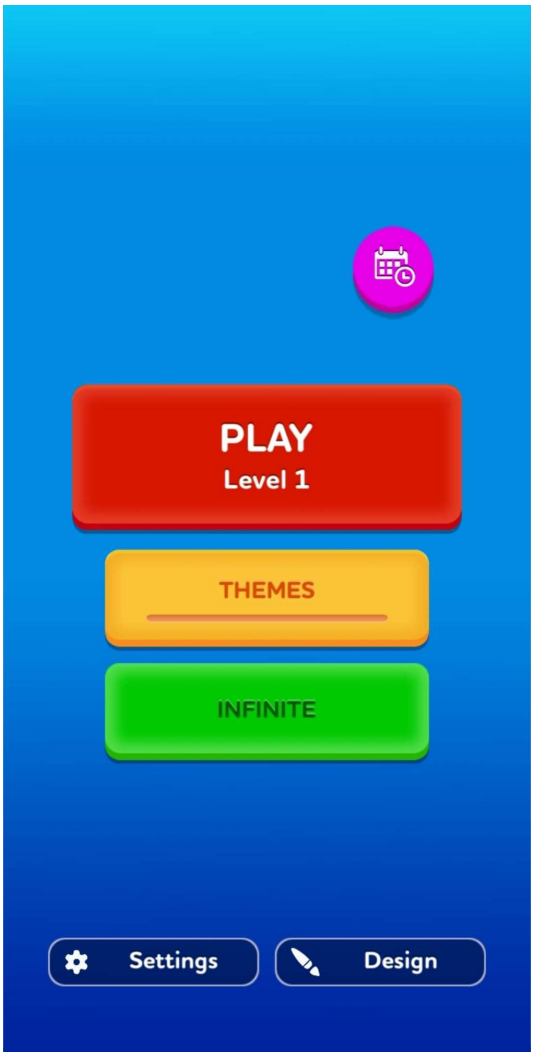
Aplikacija „*Riječi čuda: Križaljka*“ sastoji se od dva zaslona i također se mogu odabrati jezici igre te pregledati svoje rezultate. Postoje razine igranja koje se mijenjaju po težini. Igra se tako da se spajaju ponuđena slova te se križaljka popunjava riječima koje imaju smisla (Slika 2.1.1.). Igrom se proširuje vokabular i poboljšava pravopis igrača. [5]



Sl. 2.1.1. zaslon u Riječi čuda: Križaljka

2.1.2. Word Search - Words Puzzle Game

Aplikacija „*Word Search – Words Puzzle Game*“ sastoji se od dva zaslona. Moguć je odabir razine igranja i tema. Igra se tako da se u križaljci pronalaze riječi koje su ponuđene na dnu zaslona (Slika 2.1.2.). Također je moguće ugasiti glazbu i efekte te pratiti napredak. Riječi se u križaljci mogu označavati gore, dolje, lijevo, desno i dijagonalno. Teme riječi u igri su životinje, boje, gradovi i povijest. [6]



Sl. 2.1.2. Početni zaslon u igri Word Search – Words Puzzle Game

2.1.3. Wordle!

Aplikacija *„Wordle!“* sastoji se od pet zaslona. Moguće je igrati klasično popunjavanje križaljke (Slika 2.1.3.) tako da su na zaslonu iscrtana polja za slova koja je popunjavaju i na dnu zaslona je tipkovnica kojom se upisuju slova. Igra daje mogućnost dobivanja nagrada za svaku riješenu

križaljku svakoga dana. Također je moguće mijenjanje teme igre, zvuk igre te se može uređivati profil igrača. [7]

Sl. 2.1.3. Zaslon igre u igri Wordle!

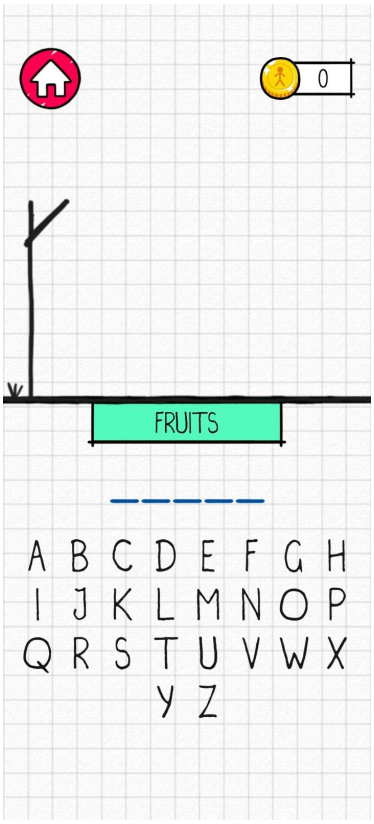
2.1.4. Wordpaste

Aplikacija *„Wordpaste“* sastoji se od pet zaslona. Ova aplikacija je verzija kaladont igre na engleskom jeziku u kojoj je moguć odabir više razina igranja te na zaslonu igre predložena je riječ na koju igrač pomoću tipkovnice upisuje novu riječ s početna dva slova s kojima završava predložena riječ na ekranu (Slika 2.1.4.). Postoje tri života koja se smanjuju upisivanjem riječi više puta. Vrijeme pogađanja riječi se smanjuje ili povećava ovisno o razini igranja. Također postoji zaslon za pomoć u igri i zaslon postavki. [8]

Sl. 2.1.4. Zaslon igre u igri Wordpaste

2.1.5. Hangman

Aplikacija „Hangman“ sastoji se od dva zaslona. Igra pogađanja riječi uz prikazan broj slova riječi i temu pod koju spada riječ (Slika 2.1.5.). Pogađanjem riječi nastavlja se igra dok pogrešnim upisivanjem riječi iscrtava se dio po dio lika čovjeka koji visi s vješala. Omogućeno je mijenjanje jezika igre, gašenje glazbe te gašenje zvukova dodira u igri. [9]



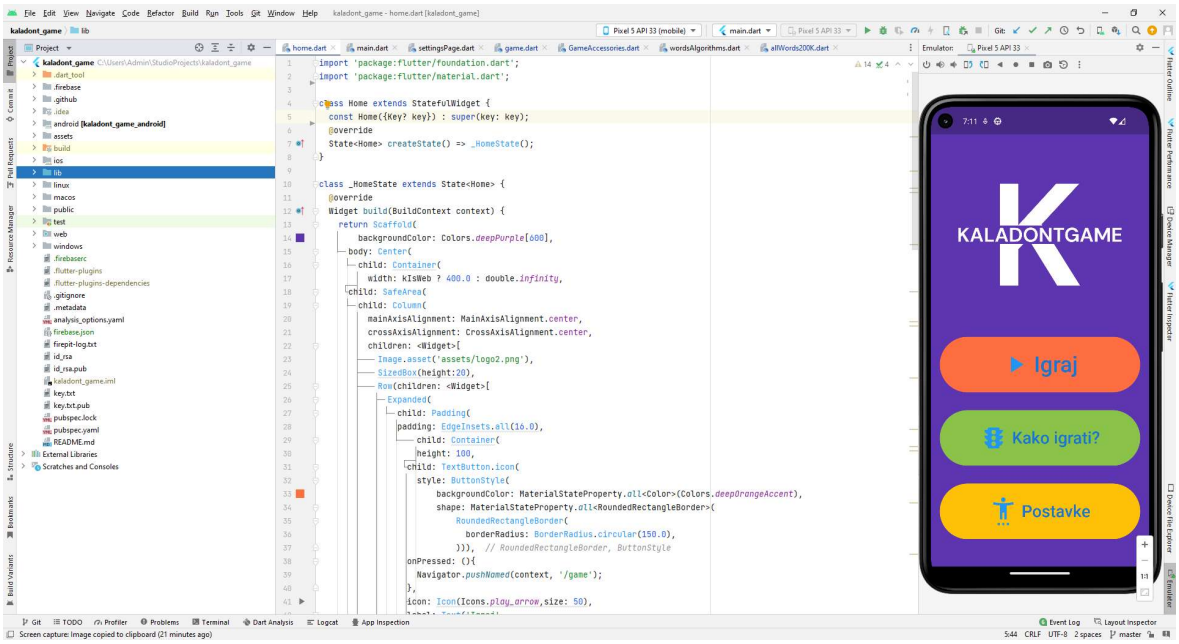
Sl. 2.1.5. Zaslon igre u igri Hangman

3. TEHNOLOGIJE KORIŠTENE U IZRADI RJEŠENJA

Aplikacija simulacije kaladont igre izrađena je u Android Studio-u. Korišteni razvojni okvir je Flutter, a kod je napisan u Dart-u. Povezivanje aplikacije na bazu podataka korištena je Firebase platforma.

3.1. Android Studio

Android Studio je integrirano razvojno okruženje razvijeno za razvoj Android aplikacija. Napravljen od tvrtke Google, te podržan na: Windows (Slika 3.1.), Linux i MacOS-u. Olakšava pronalaženje grešaka u kodu, refaktoriranje koda te automatsko dovršavanje riječi. Također ima ugrađeni Android emulator za testiranje aplikacija. [1]



Sl. 3.1. Android Studio

3.2. Flutter

Flutter je framework otvorenog koda kojeg je napravio Google za razvoj višeplatformskih aplikacija. Podržan je na MacOS-u , Windows-u i Linux-u. Omogućava razvoj Android, IOS, Windows i web aplikacija. Kod se piše u Dart programskom jeziku, dok je sam Flutter napisan u C++-u. Flutter se sastoji od widgeta od kojih su najčešći Flexbox za raspored stupaca i redova te MaterialApp koji je korijen aplikacije i sadrži druge widgete kojima implementira navigaciju. Za IOS koriste se komponente Cupertino. Također ima Widgete sa stanjem i bez stanja, prema [11]. Koristi dinamičko prevođenje koje omogućuje korisniku pregled promjena bez ponovnog pokretanja aplikacije. [2]

3.3. Dart

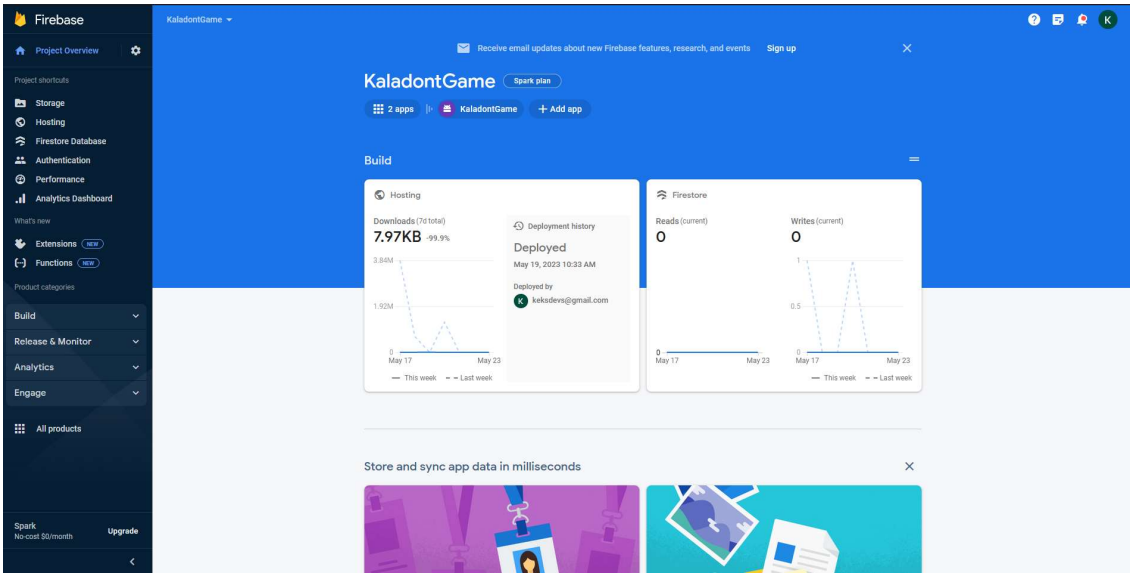
Dart je objektno orijentiran programski jezik. Razvija ga Google. Koristi se za razvoj mobilnih aplikacija, web aplikacija te serverskih aplikacija. Otvorenog je koda s BSD licencom. Postoji i DartPad u kojem se može izvršavati Dart kod u web pregledniku. Dart također implementira sakupljanje smeća (engl. *garbage collector*) tako da korisnik ne mora kontrolirati oslobađanje memorije, prema [10]. Može se prevesti u JavaScript ili u strojni kod. [3]

```
32    void incrementPoint(){
33      points++;
34      setState(() => points=points);
35    }
```

Sl. 3.2. Dart kod

3.4. Firebase

Firebase je Google-ova razvojna platforma za web i mobilne aplikacije na kojoj se može pratiti analitika podataka o korištenju aplikacija, autentifikacija korisnika, baze podataka u stvarnom vremenu i još mnogo toga. Jednostavna je za povezivanje s Flutter aplikacijom (Slika 3.3.) te je do neke mjere besplatna što je veliki plus za manje aplikacije. [4]



Sl. 3.3. Prikaz Firebase sučelja

4. RAZVOJ APLIKACIJE

U ovom poglavlju je ukratko opisan razvoj aplikacije. Početna ideja aplikacije je bila da se može mobilnim uređajima igrati popularna igra Kaladonta, jer ne postoji niti jedna takva igra na Google Play ili App Store trgovini. Flutter projekt stvara se naredbom „*flutter create naziv_projekta*“. Aplikacija se sastoji od četiri zaslona, nadalje će se opisati razvoj zaslona po zaslon.

4.1. Početni zaslon

Početni zaslon se prikazuje prvi ulaskom u aplikaciju. U njemu su definirana tri gumba:

1) Igraj
2) Kako igrati?
3) Postavke

Također je prikazan logo aplikacije s nazivom „*KaladontGame*“.

Sl. 4.1. Prikaz početnog zaslona

```
26      initialRoute: '/',
27      routes: {
28        '/': (context) => const Home(),
29        '/game': (context) => const Game(),
30        '/ranking': (context) => const Ranking(),
31        '/settings': (context) => const SettingsPageMy(),
32      },
```

Sl. 4.2. Prikaz Dart koda navigacije među zaslonima

Implementacija izgleda početnog zaslona ostvarena je tako da sve centrirano na sredinu te zamotano u kontejner i raspoređeno u jedan stupac. U stupcu je logo i još tri retka u kojima su gumbovi (Slika 4.3.). Svaki gumb ima svoju ikonu i natpis te su mu zaobljeni rubovi.

```
return Scaffold(
    backgroundColor: Colors.deepPurple[600],
  body: Center(
    child: Container(
      width: kIsWeb ? 400.0 : double.infinity,
    child: SafeArea(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      crossAxisAlignment: CrossAxisAlignment.center,
      children: <Widget>[
      Image.asset('assets/logo2.png'),
      SizedBox(height:20),
      Row(children: <Widget>[
        Expanded(
          child: Padding(
            padding: EdgeInsets.all(16.0),
              child: Container(
                height: 100,
              child: TextButton.icon(
                style: ButtonStyle(
                    backgroundColor: MaterialStateProperty.all<Color>(Colors.deepOrangeAccent),
                    shape: MaterialStateProperty.all<RoundedRectangleBorder>(
                        RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(150.0),
                        ))),  // RoundedRectangleBorder, ButtonStyle
                onPressed: (){
                  Navigator.pushNamed(context, '/game');
                },
                icon: Icon(Icons.play_arrow,size: 50),
                label: Text('Igraj',
                  style: TextStyle(
                    fontSize: 40,
                    foreground: Paint()
                      ..style = PaintingStyle.fill
                      ..strokeWidth = 2
                      ..color = Colors.blue[700]!,
                  ),  // TextStyle
                ),  // Text
            )),),  // TextButton.icon, Container, Padding, Expanded
        SizedBox(height:20),
```

Sl. 4.3. Prikaz dijela koda početne stranice

4.2. Zaslon igre

Na zaslonu igre prikazana je riječ koju igra daje odnosno riječ na koju igrač treba nastaviti igrati. Prikazan je brojač bodova koji na svaku uspješno pogođenu riječ dodaje po jedan bod. Također su

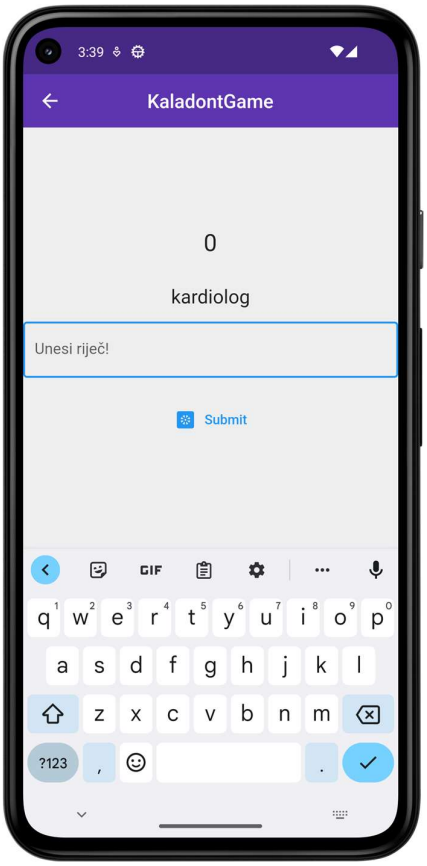
prikazani polje za unos riječi koristeći dostupnu tipkovnicu i gumb za potvrdu unosa riječi. (Slika 4.5.)

```
class Rijec {
  int n = 0;
  Uint8List s = Uint8List(50);
  String str = "";

  @override
  String toString() {
    print("n: " + n.toString() + " s: " + s.toString() + " str: " + str);
    return super.toString();
  }
```

Sl. 4.4. Prikaz koda klase koja predstavlja riječ

Klasa Rijec ima u sebi broj koji predstavlja broj znakova riječi, riječ predstavljenu bajtovima i riječ predstavljenu slovima. Također ima prepisanu funkciju za ispis. (Slika 4.4.)



Sl. 4.5. Prikaz zaslona igre

Prva riječ generirana otvaranjem zaslona je jedna slučajna riječ u rječniku s riječima za razinu igranja odabranu u postavkama. Iduće korisnik upisuje riječ koja počinje na posljednja dva slova prikazane riječi. Klikom na gumb potvrde unosa, riječ se provjerava tako da se prvo provjeri počinje li upisana riječ sa zadnja dva slova ponuđene riječi te postoji li uopće ta riječ u rječniku.

```
107  int TestirajRijec(String rijec) {
108    int dg, gg;
109    double s;
110    int k;
111    Rijec q = Rijec();
112    q = NapraviRijec(rijec + "\n");
113    dg = 0;
114    gg = rj.length - 1;
115    for (;;) {
116      s = (dg + gg) / 2;
117      if (dg > gg) return 1;
118      k = rj[s.round()].Usporedi(rj[s.round()], q);
119      if (k == 0) return 0;
120      if (k == -1) dg = s.round() + 1;
121      if (k == 1) gg = s.round() - 1;
122    }
123  }
```

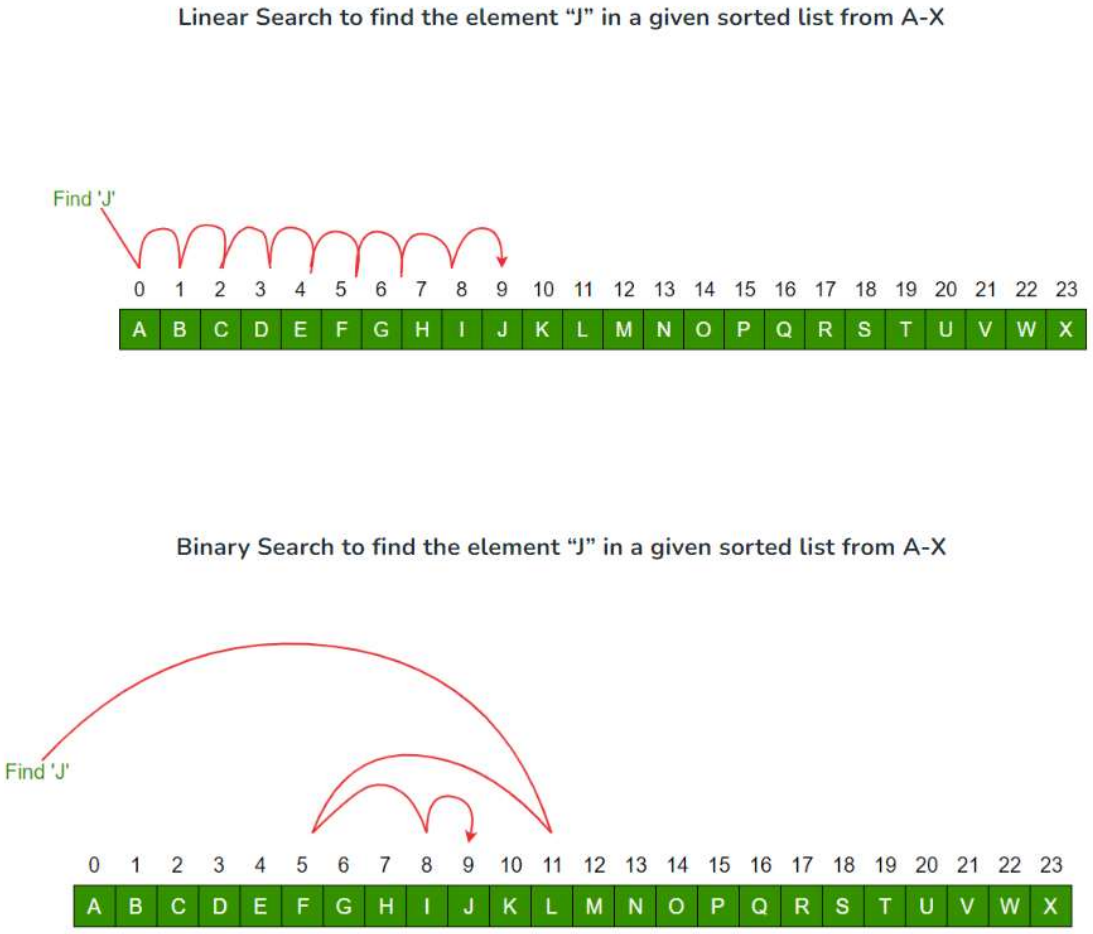
Sl. 4.6. Prikaz dijela koda algoritma pretraživanja postojanja riječi

```
int Usporedi(Rijec rijec1, Rijec rijec2) {
  int i, mn, slicnost = 0;
  if (rijec1.n == rijec2.n) {
    for (i = 0; i < rijec1.n; i++) {
      if (rijec1.s[i] == rijec2.s[i]) slicnost++;
      if (rijec1.s[i] < rijec2.s[i]) return -1;
      if (rijec1.s[i] > rijec2.s[i]) return 1;
    }
    return 0;
  } else {
    if (rijec1.n < rijec2.n)
      mn = rijec1.n;
    else
      mn = rijec2.n;
    for (i = 0; i < mn; i++) {
      if (rijec1.s[i] == rijec2.s[i]) slicnost++;
      if (rijec1.s[i] < rijec2.s[i]) return -1;
      if (rijec1.s[i] > rijec2.s[i]) return 1;
    }
    if (slicnost == rijec1.n)
      return -1;
    else
      return 1;
  }
}
```

Sl. 4.7. Prikaz dijela koda algoritma za usporedbu riječi

Provjera riječi obavljena je tako da je rječnik pretvoren u brojeve te sortiran od najmanjeg do najvećeg. Binarnim pretraživanjem (Slika 4.9.) pretražuje se brže nego da se pretražuje sekvencijalnim pretraživanjem. Kod sekvencijalnog pretraživanja (Slika 4.10.) nije potrebno sortirati niz, dok je kod binarnog potrebno. Sekvencijalno pretraživanje ima složenost O(n), dok je složenost binarnog pretraživanja O(log n). Algoritam sekvencijalnog pretraživanja ide po redu te ispituje je li taj onaj koji je tražen, dok algoritam binarnog pretraživanja prepolovljuje niz na pola sve dok ne nađe traženi broj, a usmjerava se u koju polovicu će ići iduće tako da provjeri je li broj manji ili veći od traženog. (Slika 4.8.) U ovom rješenju odlučeno je da će se koristiti binarno pretraživanje jer je skup riječi jako velik te se sekvencijalnim pretraživanjem uspori igra toliko da je to primjetno i nepoželjno kod igranja. Skup riječi je abecedno sortirani skup koji se sastoji od oko dvjesto tisuća riječi ne dužih od 50 znakova i kraćih od 3 znaka. To su riječi koje su imenice

u nominativu, bez znakova kojih nema u hrvatskoj abecedi. Riječi se učitavaju u memoriju pri pokretanju aplikacije kako bi se još ubrzao proces pretraživanja. Funkcija Usporedi (Slika 4.7.) napravljena je jer drugačiji format zapisa riječi.



Sl. 4.8. Prikaz rada sekvencijalnog i binarnog pretraživanja

```
Procedure binary_search
   A ← sorted array
   n ← size of array
   x ← value to be searched

   Set lowerBound = 1
   Set upperBound = n

   while x not found
      if upperBound < lowerBound
         EXIT: x does not exists.

      set midPoint = lowerBound + ( upperBound - lowerBound ) / 2

      if A[midPoint] < x
         set lowerBound = midPoint + 1

      if A[midPoint] > x
         set upperBound = midPoint - 1

      if A[midPoint] = x
         EXIT: x found at location midPoint
   end while

end procedure
```

Sl. 4.9. Prikaz pseudokoda binarnog pretraživanja

```
procedure linear_search (list, value)

   for each item in the list
      if match item == value
         return the item's location
      end if
   end for

end procedure
```

Sl. 4.10. Prikaz pseudokoda sekvencijalnog pretraživanja

Algoritam pretvaranja riječi u brojeve radi tako da se prvo slovo abecede pretvori u broj nula, drugo slovo abecede u broj 1 i tako do kraja abecede. (Slika 4.11.)

```
51    Rijec NapraviRijec(String str) {
52      int k = 0, josk = 0, n = 0;
53      Rijec rijec = Rijec();
54      for (int i = 0; i < str.length; i++) {
55        josk = 0;
56        if (str[k] == "\n") break;
57        if (str[k] == 0) break;
58        if (str[k] == 'a') rijec.s[n] = 0;
59        if (str[k] == 'b') rijec.s[n] = 1;
60        if (str[k] == 'c') rijec.s[n] = 2;
```

Sl. 4.11. Prikaz dijela koda algoritma pretvaranja riječi u broj

Time se također smanjuje veličina riječi zbog znakova kao što su: nj, lj, dž, koji su predstavljeni s dva znaka, no kada se pretvore u brojeve predstavljeni su s jednim znakom. Ako je riječ prihvaćena bira se iduća riječ iz rječnika. Svakim ulaskom u zaslon igre, sve riječi se izmiješaju. (Slika 4.12.)

```
47    List shuffle(List items) {
48      var random = Random();
49      for (var i = items.length - 1; i > 0; i--) {
50        var n = random.nextInt(i + 1);
51        var temp = items[i];
52        items[i] = items[n];
53        items[n] = temp;
54      }
55      return items;
56    }
```

Sl. 4.12. Prikaz dijela koda miješanja riječi u listi

Igra se igra sve dok se ne pojavi riječ koja završava na slova -nt ili aplikacija u svom rječniku ne može pronaći iduću riječ, te ako je korisnik upisao riječ koja završava na ta slova, aplikacija će ispisati da je pobijedio prikazano na slici 4.13. No ako korisnik unese riječ koja ne postoji odnosno ne nalazi se u rječniku, aplikacija će ispisati da je upisao riječ koja ne postoji te ponuditi gumb za igranje nove igre.

Sl. 4.13. Prikaz zaslona kada igrač pobjedi

Također ako korisnik unese riječ koja ne počinje na zadnja dva slova ponuđene riječi aplikacija neće htjeti provjeravati uopće riječ sve dok se ne unese riječ s odgovarajuća prva dva slova. Aplikacija također ne prihvaća riječi koje je korisnik već unio tokom trenutne igre (Slika 4.14.), tako što svaku unesenu i prihvaćenu riječ sprema u posebnu listu.

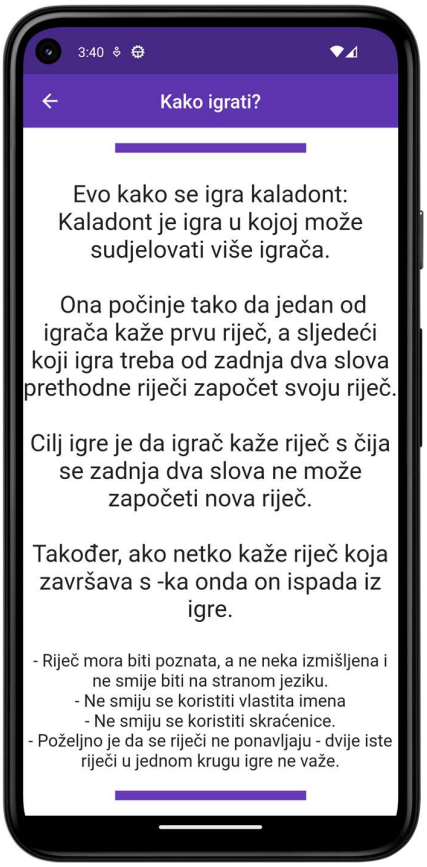
Sl. 4.14. Prikaz zaslona kada igrač ponovi riječ

Dizajn zaslona (Slika 4.14.) sastoji se od jednog stupca i više kontejnera složenih jedan ispod drugog. U kojima su:

1) Prikaz bodova
2) Riječ aplikacije
3) Polje za unos riječi
4) Gumb za potvrdu riječi
5) Gumb za resetiranje igre

4.3. Zaslon objašnjenja igranja

Na zaslonu objašnjenja kako se igra, prikazan je tekst s objašnjenjem kao na slici 4.15.



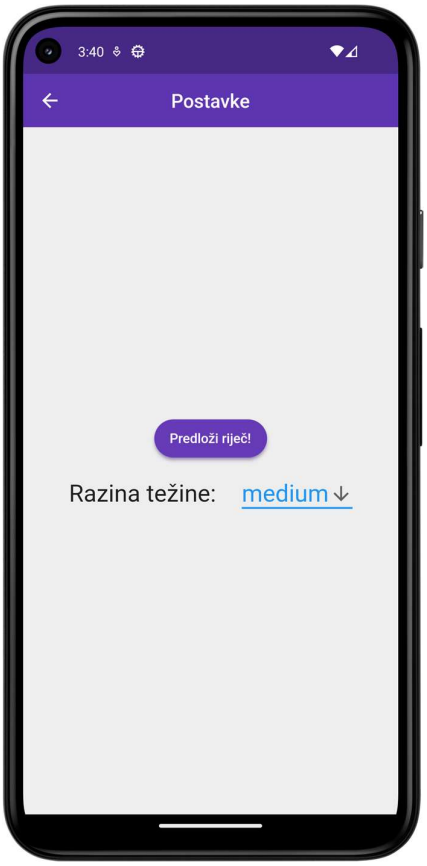
Sl. 4.15. Prikaz zaslona objašnjenja igre

```
Row(children: <Widget>[
  Expanded(
    child: RichText(
      textAlign: TextAlign.center,
      text: const TextSpan(
        text: '\nEvo kako se igra kaladont:\n',
        style: TextStyle(
          fontSize: 25.0,
          color: Colors.black87,
          fontFamily: 'Montserrat',
        ),  // TextStyle
        children: <TextSpan>[
          TextSpan(text: 'Kaladont je igra u kojoj može sudjelovati više igrača.\n\n'),
          TextSpan(text: ' Ona počinje tako da jedan od igrača kaže prvu riječ,'),
          TextSpan(text: ' a sljedeći koji igra treba od zadnja dva slova '),
          TextSpan(text: 'prethodne riječi započet svoju riječ.\n\n'),
          TextSpan(text: 'Cilj igre je da igrač kaže riječ s čija se zadnja dva slova ne može započeti nova riječ.\n\n'),
          TextSpan(text: 'Također, ako netko kaže riječ koja završava s -ka onda on ispada iz igre.\n\n'),
          TextSpan(text: '- Riječ mora biti poznata, a ne neka izmišljena i ne smije biti na stranom jeziku.\n',style: TextStyle(
            fontSize: 18.0)),  // TextStyle, TextSpan
          TextSpan(text: '- Ne smiju se koristiti vlastita imena \n',style: TextStyle(
              fontSize: 18.0)),  // TextStyle, TextSpan
          TextSpan(text: '- Ne smiju se koristiti skraćenice. \n',style: TextStyle(
              fontSize: 18.0)),  // TextStyle, TextSpan
          TextSpan(text: '- Poželjno je da se riječi ne ponavljaju - dvije iste riječi u jednom krugu igre ne važe.\n',style: TextStyle(
              fontSize: 18.0)),  // TextStyle, TextSpan
        ],  // <TextSpan>[]
      ),  // TextSpan
    )  // RichText
  ),  // Expanded
] // <Widget>[]
),  // Row
```

Sl. 4.16. Prikaz jednog retka zaslona objašnjenja igre

4.4. Zaslon postavki igre

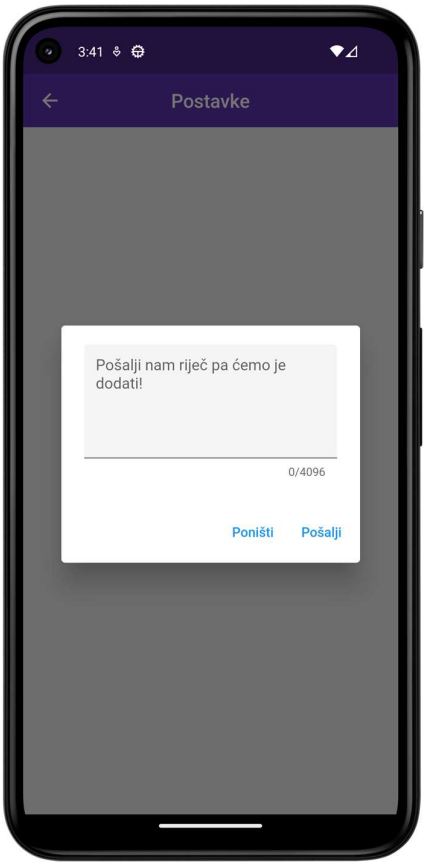
Zaslon postavki igre prikazuje padajući izbornik za odabir razine igranja prikazan na slici 4.17. te gumb koji otvara polje za prijedlog riječi koje se šalju u Firebase bazu. Razina igranja (Slika 4.18.) se temelji na tome koliki broj riječi je dostupan aplikaciji za igranje, npr. na razini „*easy*“ dostupna je svaka stota riječ sveukupnog rječnika, dok su na razini „*hard*“ dostupne sve riječi iz rječnika. Gumb za prijedlog novih riječi otvara dijaloški okvir (engl. *Dialog Box*) u kojem se može unijeti riječ (Slika 4.19.) te ju poslati u Firebase bazu podataka (Slika 4.20.).



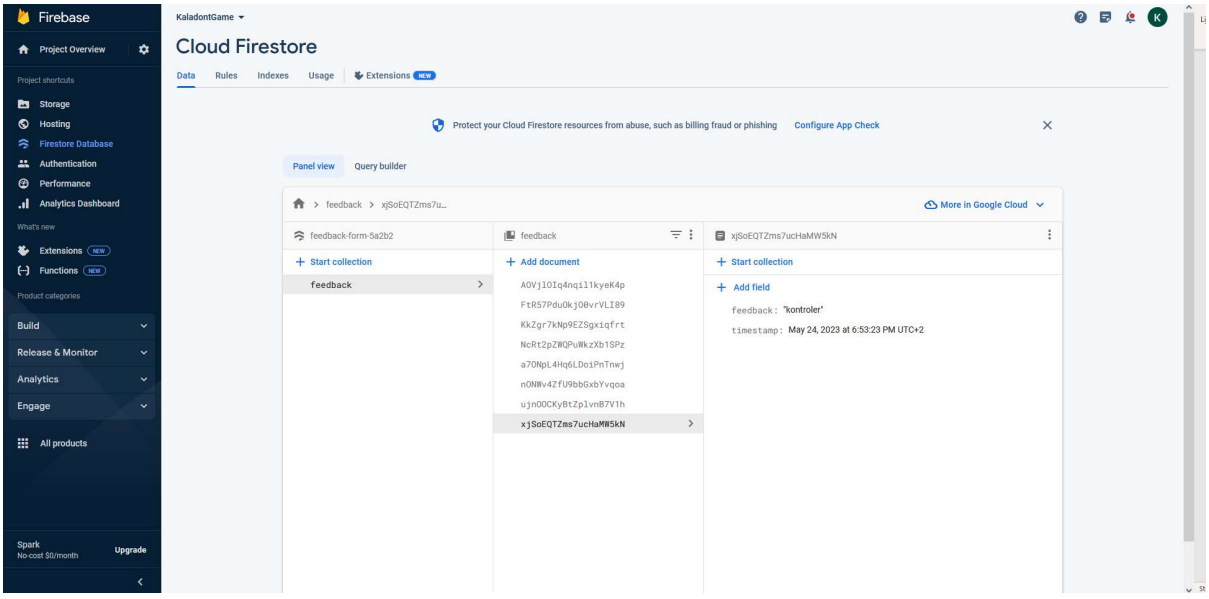
Sl. 4.17. Prikaz zaslona postavki igre

```
void ChangeWordsByLevel(){
  gameWords=[];
  if(getLevel()=="easy"){
    gameWords=getElements(loadWordsImenice(), 100);
  }
  if(getLevel()=="medium"){
    gameWords=getElements(loadWordsImenice(), 50);
  }
  if(getLevel()=="hard"){
    gameWords=getElements(loadWordsImenice(), 1);
  }
  shuffledWords = shuffle(gameWords);
}
```

Sl. 4.18. Prikaz Dart koda funkcije odabira razine težine igranja igre



Sl. 4.19. Prikaz dijaloškog okvira za prijedlog riječi



Sl. 4.20. Prikaz jedne predložene riječi u bazi podataka

Izgled zaslona postavki (Slika 4.17.) ostvaren je u jednom stupcu u kojem su gumb za otvaranje dijaloškog okvira (engl. *Dialog Box*) i retka u kojem je padajući izbornik.

5. ZAKLJUČAK

U ovom radu proučen je razvoj Flutter aplikacije za popularnu igru Kaladont, uz korištenje Firebase baze podataka za prijedlog novih riječi. Tijekom razvoja aplikacije korištene su različite funkcionalnosti koje pruža Flutter okvir, kao što su reaktivno ažuriranje sučelja i podrška za više platformi. Prednosti korištenja Flutter-a za razvoj aplikacije su njegova brzina izrade, fleksibilnost i mogućnost dijeljenja koda između različitih platformi. U današnje vrijeme sve više vremena ljudi provode na mobilnim uređajima te raste potreba za mobilnim aplikacijama i igrama. Multiplatformski razvoj uvelike pomaže programerima jer nije potrebno pisati više različitih kodova za jednu aplikaciju. Aplikacija ima još mjesta za nadogradnju, kao što su istjecanje vremena za ponuditi iduću riječ i mogućnost igranja s više igrača. Također bi bilo dobro dodati praćenje rezultata drugih igrača.

LITERATURA

[1] AndroidTarget, „Android Studio“, Dostupno: https://www.techtarget.com/searchmobilecomputing/definition/Android-Studio. [Pristupljeno: 26.svi.2023.]

[2] Flutter, „FAQ“, Dostupno: https://docs.flutter.dev/resources/faq. [Pristupljeno: 26.svi.2023.]

[3] Javatpoint, „What is Dart Programming“, Dostupno: https://www.javatpoint.com/flutter-dart-programming. [Pristupljeno: 26.svi.2023.]

[4] Medium, „What is Firebase? The complete story, abridged“, Dostupno:https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0. [Pristupljeno: 26.svi.2023.]

[5] Riječi čuda: Križaljka, Dostupno: https://play.google.com/store/apps/details?id=com.fugo.wow&hl=hr. [Pristupljeno: 18.srp.2023.]

[6] Word Search - Words Puzzle Game, Dostupno: https://play.google.com/store/apps/details?id=com.playvalve.wsjourney&hl=hr. [Pristupljeno: 18.srp.2023.]

[7] Wordle!, Dostupno: https://apps.apple.com/us/app/wordle/id1095569891?platform=iphone. [Pristupljeno: 18.srp.2023.]

[8] Wordpaste, Dostupno: https://balkanandroid.com/wordpaste-engleska-verzija-poznate-igrice-kaladont/. [Pristupljeno: 18.srp.2023.]

[9] Hangman, Dostupno: https://play.google.com/store/apps/details?id=com.tellmewow.senior.hangman&hl=hr. [Pristupljeno: 18.srp.2023.]

[10] J. Sande, Dart Apprentice: Fundamentals, Kodeco, 1.stu.2022.

[11] M.L. Napoli, Beginning Flutter: A Hands On Guide To App Development, Wrox, 20.ruj.2019.

SAŽETAK

Cilj ovog rada je bio napraviti multiplatformsku aplikaciju za simulaciju igranja popularne igre Kaladont. Razvoj aplikacije ostvaren je u Android Studio-u i Flutter razvojnom okviru korištenjem programskog jezika Dart te Firebase baze podataka za prijedloge novih riječi za igru. Rezultat je da aplikacija pametno i brzo predlaže nove riječi te ne dopušta da se upisuju riječi koje ne postoje ili nemaju smisla odnosno ne omogućava varanje u igri. Omogućava učenje novih riječi, pospješuje brže razmišljanje i zabavu. Objašnjava kako igrati igru te omogućuje predlaganje novih riječi koje još nisu u rječniku aplikacije. Također, daje korisniku priliku okušati se u raznim težinama igre.

Ključne riječi: Dart, Firebase, Flutter, kaladont igra

ABSTRACT

Kaladont game simulation

The aim of this project was to develop a cross-platform application for simulating the popular game of Kaladont. The application was developed using Android Studio and the Flutter framework, utilizing the Dart programming language, and Firebase database for suggesting new words in the game. As a result, the application intelligently and quickly suggests new words, preventing the input of non-existent or nonsensical words, thus preventing cheating in the game. It facilitates learning new words, enhances quick thinking, and provides entertainment. It explains how to play the game and allows users to suggest new words that are not yet in the application's dictionary. Additionally, it offers the user the opportunity to challenge themselves with various difficulty levels of the game.

Keywords: Dart, Firebase, Flutter, Kaladont game