Sri Lanka Institute of Information Technology

“Wordly”: Word Guessing App

Report

IT19126234

Jayasinghe S.L

2022_REG_08

Current Trends in Software Engineering – SE4010

B.Sc. (Hons’) in Information Technology

Specializing in Software Engineering

SLIIT
Discover Your Future

Table of Contents

List of Figures

1. Project Description

"Wordly" is a quiz-based word guessing game developed as a mobile app. The app was developed using Flutter and Firebase.

A normal user can create an account and answer quizzes. In the quizzes, the user needs to select the correct answer(s) for the given definition. Users will earn points based on the number of right answers in a quiz and the points will be added to the user's total points in the system. There is a leaderboard where the users can see their rank based on their total points among all the other active users. Users can use their profile to view their details and the total points they have earned. Users can use this to edit their details too. Also, users can always add reviews to give feedback about the app. Admins can add word definitions and answers to the database through the app. One word definition has four answers, where one or multiple of them can be right. Admins can make normal users Admins or delete them. Admins can also view all the user feedback.

2. Implemented Functionalities

All the code related to the main functionalities implemented by Jayasinghe S.L (IT19126234) is available in the Appendices section of the report, or the complete project is available at https://github.com/rukshan99/wordly .

2.1 User Profile

The user can view their user details except for passwords through the user profile UI. There are three operations such as Read, Update and Delete. If any user needs to update their user name, he/she simply can edit and update their user name. Email and the points that the user has scored cannot be changed anytime. And user can delete their account from this application by using the delete button. It will delete all user records under logged user email.

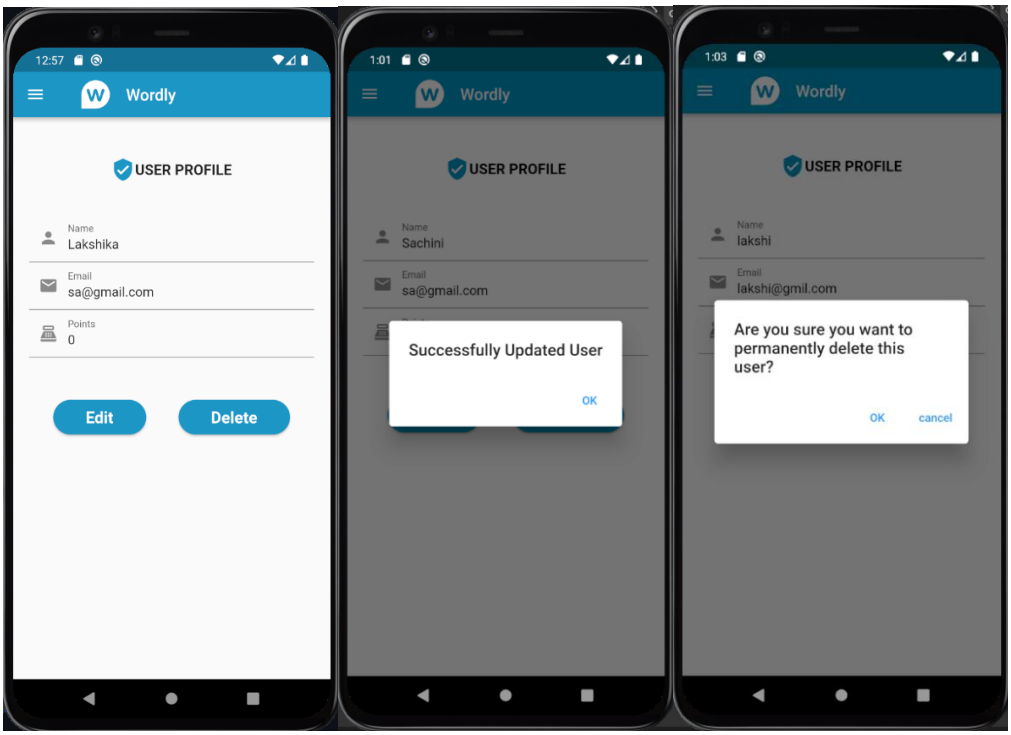

Figure 2.1 User profile Screen

2.2 Add review

Every user has the ability to add a rating and review for this application. They can add their star rate and comment by providing their email address. This is a “Create” operation. Added user reviews will be stored in the database.

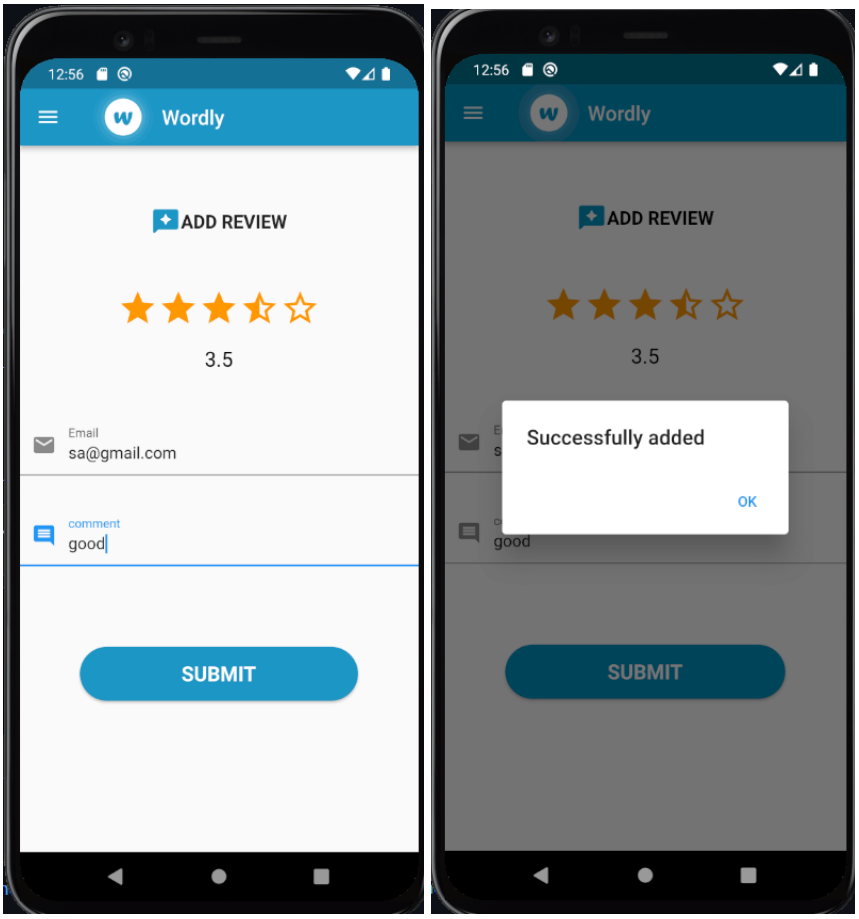

Figure 2.2 Add Review Screen

2.3 Display reviews list

This is a “read” operation. Only Admin can view all the reviews which have been added by users as a comment or ratings. The review list item consists of the email, comment and star rating point. If the admin needs to quickly filter a review they can search by keyword.

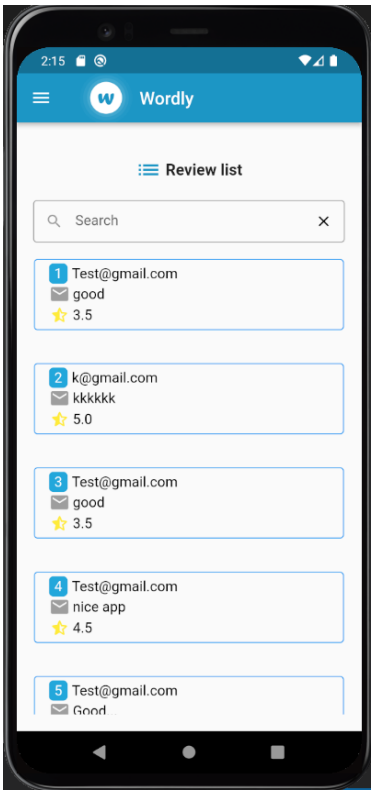

Figure 2.2 Reviews list Screen

2.4 Splash

Figure 2.3 Splash

3. References

- Flutter documentation: https://docs.flutter.dev/
- Firebase documentation: https://firebase.flutter.dev/
- Background animations: https://felixblaschke.medium.com/fancy-background-animations-in-flutter-4163d50f5c37

4. Appendices

4.1 Controllers/user_controller.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:wordly/models/user.dart';

class UserController {
  final FirebaseFirestore _db = FirebaseFirestore.instance;

//delete user details
  deleteUserDetails(String email) {
    collectionRef.where('email', isEqualTo: email).get().then((snapshot) => {
          _db.runTransaction((Transaction transaction) async {
            snapshot.docs.forEach(
                (DocumentSnapshot doc) => {transaction.delete(doc.reference)});
          })
        });
  }

//fetch user profile details
  Future<dynamic> getUserData(String? email) async {
    QuerySnapshot snapshot = await collectionRef.where('email', isEqualTo: email).get();
    dynamic user = snapshot.docs[0].data();
     snapshot.docs.forEach(
              (DocumentSnapshot doc) => {
                user = doc.data()
              });
    print(user);
    return user;
  }




//update user profile details
```

```
  updateUserDetails( String name, String email) async {
    try {
      _db.runTransaction((Transaction transaction) async {
        var snapshots =
            await collectionRef.where('email', isEqualTo: email).get();

        snapshots.docs.forEach((DocumentSnapshot doc) {
          doc.reference.update({'name': name});
        });

      });
    } catch (e) {
      print(e.toString());
    }
  }
}
```

4.2 Screens/user_profile.dart

```
import 'package:firebase_auth/firebase_auth.dart' hide User;
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/material.dart';
import 'package:wordly/utils/color.dart';
import 'package:wordly/widgets/header_container.dart';
import '../controllers/user_controller.dart';
import '../models/user.dart';
import '../widgets/main_drawer.dart';
import '../widgets/user_drawer.dart';

class UserProfileScreen extends StatefulWidget {
  const UserProfileScreen({Key? key}) : super(key: key);

  @override
  _UserProfileScreenState createState() => _UserProfileScreenState();
}
```

```
class _UserProfileScreenState extends State<UserProfileScreen> {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  final userController = UserController();

  late String userName, userEmail, _Name;
  late int userPoints = 0;
  late dynamic userProf;

  @override
  void initState() {
    super.initState();
  }

  Future<dynamic> getUserDetils() async{
     final user = _auth.currentUser;
    final userProf = await userController.getUserData(user?.email.toString());
     if ( userProf != null){
     setState(() {
      userName = userProf['name'];
      userEmail = userProf['email'];
      userPoints = userProf['points'];
     });
     }
  }


  updateUserDetails() async{
    try{
```

```
    await userController.updateUserDetails(userName,userEmail);
    showUpdateAlert();
   } catch (e) {
        showError(e.toString());
   }
}

deleteUser() async{
  try{
    showDialog(
      context: context,
      builder: (BuildContext context) {
        return AlertDialog(
          title: const Text('Are you sure you want to permanently delete this user?'),
          actions: <Widget>[
            TextButton(
                onPressed: () async {
                  await userController.deleteUser(userEmail);
                  showAlert();
                },
                child: const Text('OK'),
                ),

            TextButton(
                onPressed: () {
                  Navigator.of(context, rootNavigator: true).pop();
                },
            child: const Text('cancel')),
          ],
        );
```

```
      });
    } catch (e) {
        showError(e.toString());
    }
  }

  showAlert() {
    showDialog(
      context: context,
      builder: (BuildContext context) {
        return AlertDialog(
          title: const Text('Successfully Deleted User'),
          actions: <Widget>[
            TextButton(
              onPressed: () {
                Navigator.of(context).pop();
                navigateToHome();
              },
              child: const Text('OK')),
          ],
        );
      });
  }

  showUpdateAlert() {
    showDialog(
      context: context,
      builder: (BuildContext context) {
        return AlertDialog(
          title: const Text('Successfully Updated User'),
```

```
        actions: <Widget>[
          TextButton(
              onPressed: () {
                Navigator.of(context).pop();
                // navigateToProfile();
              },
              child: const Text('OK')),
        ],
      );
    });
}
navigateToHome() async {
   Navigator.pushReplacementNamed(context, "welcome");
}
navigateToProfile() async {
   Navigator.pushReplacementNamed(context, "userProfile");
}
showError(String errormessage) {
  showDialog(
     context: context,
     builder: (BuildContext context) {
       return AlertDialog(
         title: const Text('ERROR'),
         content: Text(errormessage),
         actions: <Widget>[
           TextButton(
              onPressed: () {
                Navigator.of(context).pop();
              },
              child: const Text('OK')),
```

```
          ],
        );
      });
}

@override
Widget build(BuildContext context) {
  return FutureBuilder(
    future: getUserDetils(),
    builder: (BuildContext context, AsyncSnapshot snapshot) {
    return Scaffold(
      appBar: AppBar(
      title: Row(
        mainAxisAlignment: MainAxisAlignment.start,
        children: [
          Image.asset(
            'assets/img/logo.jpg',
            fit: BoxFit.cover,
            height: 60.0,
          ),
          Container(
            padding: const EdgeInsets.all(8.0),
            child: const Text(
              'Wordly',
              style: TextStyle(fontFamily: 'Righteous', fontSize: 20.0),
            ),
          )
        ],
      ),
      backgroundColor: const Color.fromARGB(255, 28,150,197),
```

```
),
drawer: const UserDrawer(),
body: Container(
  padding: const EdgeInsets.only(bottom: 30),
  child: Column(
    children: <Widget>
      Expanded(
        flex: 1,
        child: Container(
          padding: const EdgeInsets.all(20.0),
          child: Form(
            key: _formKey,
            child: Column(
              children: <Widget>[
                Container(),
                const SizedBox(
                  height: 30,
                ),
                Wrap(
                  crossAxisAlignment: WrapCrossAlignment.center,
                  children: const [
                    Icon(Icons.verified_user,
                        color: Color.fromARGB(255, 28,150,197), size: 30),
                    Text(
                      "USER PROFILE",
                      style: TextStyle(fontSize: 18, fontWeight: FontWeight.w800),
                    )
                  ]),
                const SizedBox(
                  height: 40.0,
```

```
),
TextFormField(
  // ignore: missing_return
  initialValue: userName,
  validator: (input) {
    if (input!.isEmpty) return 'Enter Name';
  },
  decoration: const InputDecoration(
      labelText: 'Name', prefixIcon: Icon(Icons.person)),
  onChanged: (String value) async{
    userName = value;
  },
),
TextFormField(
  // ignore: missing_return
  initialValue: userEmail,
  readOnly : true,
  validator: (input) {
    if (input!.isEmpty) return 'Enter Email';
  },
  decoration: const InputDecoration(
      labelText: 'Email', prefixIcon: Icon(Icons.email)),
),
TextFormField(
  // ignore: missing_return
  initialValue: userPoints.toString(),
  readOnly : true,
  decoration: const InputDecoration(
  labelText: 'Points',
  prefixIcon: Icon(Icons.point_of_sale)),
```

```
  ),
const SizedBox(
  height: 50.0,
),
  Row(
    children: <Widget>[
      const SizedBox(
    width: 30.0,
    ),
      ElevatedButton(
        style: ElevatedButton.styleFrom(
          primary: purpleColors, // background
          onPrimary: Colors.white,
          padding: const EdgeInsets.fromLTRB(40, 10, 40, 10),
          shape: RoundedRectangleBorder(
              borderRadius:
                  BorderRadius.circular(100.0)), // foreground
        ),
        onPressed: updateUserDetails,
        child: const Text(
          'Edit',
          style: TextStyle(
              fontSize: 20.0, fontWeight: FontWeight.bold),
        ),
      ),
      const SizedBox(
      width: 40.0,
      ),
      ElevatedButton(
        style: ElevatedButton.styleFrom(
```

```
                primary: purpleColors, // background
                onPrimary: Colors.white,
                padding: const EdgeInsets.fromLTRB(40, 10, 40, 10),
                shape: RoundedRectangleBorder(
                    borderRadius:
                        BorderRadius.circular(100.0)), // foreground
              ),
              onPressed: deleteUser,
              child: const Text(
                'Delete',
                style: TextStyle(
                    fontSize: 20.0, fontWeight: FontWeight.bold),
              ),
            ),
          ],
        ),
      ),
    ),
  );
 });
 }
}
```

4.3 Model/review.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';

class Review {
  final String email;
  final double ratingValue;
  final String comment;


  Review({
    required this.email,
    required this.ratingValue,
    required this.comment
  });

  factory Review.fromJson(Map<String, dynamic> json, DocumentReference docRef) {
    // ignore: avoid_print
    print(json['review']);
    return Review(
        email: json['email'] as String,
        ratingValue: json['ratingValue'] as double,
        comment: json['comment'] as String);
  }

  Map<String, dynamic> toMap() {
    return {"email": email,"ratingValue": ratingValue, "comment": comment};
  }
}
```

4.4 Controller/review_controller.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:wordly/models/review.dart';

class ReviewController {
  final FirebaseFirestore _db = FirebaseFirestore.instance;

  CollectionReference collectionRef =
      FirebaseFirestore.instance.collection('reviews');


    // Add a review
  addReview(Review reviewObj) async {
```

```
    try {
      _db.runTransaction((Transaction transaction) async {
        await _db.collection('reviews').doc().set(reviewObj.toMap());
      });
    } catch (e) {
      print(e.toString());
    }
  }

  // Get all reviews
  Stream<List<Review>> getReviews(String email) {
    return _db.collection('reviews').snapshots().map((snapshot) => snapshot.docs
        .map((doc) => Review.fromJson(doc.data(), doc.reference))
        .toList());
  }

  getAllReviews() {
    return _db.collection('reviews').snapshots();
  }

}
```

4.5 Screens/review.dart

```
import 'package:firebase_auth/firebase_auth.dart' hide User;
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/material.dart';
import 'package:wordly/models/review.dart';
import 'package:wordly/utils/color.dart';
import 'package:wordly/widgets/header_container.dart';
import 'package:flutter_rating_bar/flutter_rating_bar.dart';
import '../controllers/review_controller.dart';
import '../models/user.dart';
import 'package:wordly/widgets/main_drawer.dart';


import '../widgets/user_drawer.dart';


class ReviewScreen extends StatefulWidget {
  const ReviewScreen({Key? key}) : super(key: key);
```

```
  @override
  _ReviewScreenState createState() => _ReviewScreenState();
}

class _ReviewScreenState extends State<ReviewScreen> {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  final reviewController = ReviewController();

  late String _comment,_email;
  late double _ratingValue = 0 ;

  TextEditingController textarea = TextEditingController();

  submit() async {
     if (_formKey.currentState!.validate()) {
       _formKey.currentState!.save();

       try {
          Review reviewObj = Review(email: _email,ratingValue: _ratingValue, comment: _comment);
          await reviewController.addReview(reviewObj);
          showAlert();

       } catch (e) {
         showError(e.toString());
       }
     }
  }
```

```
showAlert() {
  showDialog(
      context: context,
      builder: (BuildContext context) {
        return AlertDialog(
          title: const Text('Successfully added'),
          actions: <Widget>[
            TextButton(
                onPressed: () {
                  Navigator.of(context).pop();
                  navigateToHome();
                },
                child: const Text('OK')),
          ],
        );
      });
}
navigateToReview() async {
    Navigator.pushReplacementNamed(context, "review");
}

navigateToHome() async {
    Navigator.pushReplacementNamed(context, "home");
}

showError(String errormessage) {
  showDialog(
      context: context,
      builder: (BuildContext context) {
```

```
        return AlertDialog(
          title: const Text('ERROR'),
          content: Text(errormessage),
          actions: <Widget>[
            TextButton(
                onPressed: () {
                  Navigator.of(context).pop();
                },
                child: const Text('OK')),
          ],
        );
      });
}


@override
void initState() {
  super.initState();
}


@override
Widget build(BuildContext context) {
  return Scaffold(
    resizeToAvoidBottomInset: false,
    appBar: AppBar(
      title: Row(
        mainAxisAlignment: MainAxisAlignment.start,
        children: [
          Image.asset(
```

```
          'assets/img/Logo.png',
          fit: BoxFit.cover,
          height: 60.0,
        ),
        Container(
          padding: const EdgeInsets.all(8.0),
          child: const Text(
            'Wordly',
            style: TextStyle(fontFamily: 'Righteous', fontSize: 20.0),
          ),
        )
      ],
    ),
    backgroundColor: const Color.fromARGB(255, 28,150,197),
  ),
  drawer: const UserDrawer(),
  body: Container(
    padding: const EdgeInsets.only(bottom: 30),
    child: Column(
      children: <Widget>[
        Expanded(
          flex: 1,
          child: Container(
            child: Form(
              key: _formKey,
              child: Column(
                children: <Widget>[
                  Container(),
                  const SizedBox(
                    height: 60,
```

```
),
Wrap(
  crossAxisAlignment: WrapCrossAlignment.center,
  children: const [
    Icon(Icons.reviews_rounded,
        color: Color.fromARGB(255, 28,150,197), size: 30),
    Text(
      "ADD REVIEW",
      style: TextStyle(fontSize: 18, fontWeight: FontWeight.w800),
    )
  ]),
 const SizedBox(
  height: 50.0,
),
RatingBar(
  initialRating: 0,
  direction: Axis.horizontal,
  allowHalfRating: true,
  itemCount: 5,
  ratingWidget: RatingWidget(
  full: const Icon(Icons.star, color: Colors.orange),
  half: const Icon(
    Icons.star_half,
    color: Colors.orange,
  ),
  empty: const Icon(
    Icons.star_outline,
    color: Colors.orange,
  )),
  onRatingUpdate: (value) {
```

```
    setState(() {
      _ratingValue = value;
    });
  }),

 const SizedBox(height: 5.0),
 // Display the rate in number
 Container(
   width: 50,
   height: 50,
   // decoration: const BoxDecoration(
   //     color: Colors.red, shape: BoxShape.rectangle),
   alignment: Alignment.center,
   child: Text(
     _ratingValue != null ? _ratingValue.toString() : 'Rate it!',
     style: const TextStyle(color: Colors.black, fontSize: 20),
   ),
 ),
 const SizedBox(height: 30.0),
 TextFormField(
   // ignore: missing_return
   validator: (input) {
     if (input!.isEmpty) return 'Enter Email';
   },
   decoration: const InputDecoration(
       labelText: 'Email', prefixIcon: Icon(Icons.email)),
   onSaved: (input) => _email = input!,
 ),
const SizedBox(
   height: 30.0,
```

```
),
TextFormField(
  // ignore: missing_return
  keyboardType: TextInputType.multiline,
  // maxLines: 3,
  controller: textarea,
  validator: (input) {
    if (input!.isEmpty) return 'Enter Comment';
  },
  decoration: const InputDecoration(
    labelText: 'comment',
    prefixIcon: Icon(Icons.comment)
  ),
  onSaved: (input) => _comment = input!,
),
const SizedBox(
  height: 80.0,
),
ElevatedButton(
  style: ElevatedButton.styleFrom(
    primary: purpleColors,
    onPrimary: Colors.white,
    padding: const EdgeInsets.fromLTRB(100, 15, 100, 15),
    shape: RoundedRectangleBorder(
        borderRadius:
            BorderRadius.circular(100.0)),
  ),
  onPressed: submit,
  child: const Text(
    'SUBMIT',
```

```
                    style: TextStyle(
                        fontSize: 20.0,
                        fontWeight: FontWeight.bold,
                        color: Colors.white),
                  ),
                ),
                ],
              ),
            ),
          ),
        )
      ],
    ),
  ),
 );
 }
}
```

4.6 Screens/review_List.dart

```
import 'dart:math';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter_speed_dial/flutter_speed_dial.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:provider/provider.dart';
import 'package:wordly/providers/review_provider.dart';
import 'package:wordly/widgets/main_drawer.dart';
import 'package:shared_preferences/shared_preferences.dart';
```

```
import 'package:substring_highlight/substring_highlight.dart';
import 'package:flutter_switch/flutter_switch.dart';

import '../controllers/review_controller.dart';
import '../models/review.dart';

class ReviewListScreen extends StatefulWidget {
  const ReviewListScreen({Key? key}) : super(key: key);

  @override
  _ReviewListScreenState createState() => _ReviewListScreenState();
}

class _ReviewListScreenState extends State<ReviewListScreen> {

  final reviewController = ReviewController();

  final TextEditingController _searchController = TextEditingController();

  List<QueryDocumentSnapshot> _resultsList = [];

  List<QueryDocumentSnapshot> _searchResultsList = [];


   @override
  void initState() {
    super.initState();
    _searchController.addListener(_onSearchChanged);
  }
```

```
@override
void dispose() {
  _searchController.removeListener(_onSearchChanged);
  _searchController.dispose();
  super.dispose();
}

/// Initializes the necessary state changes needed after performing a search operation
_onSearchChanged() {
  List<QueryDocumentSnapshot> filteredResultsList = [];
  _resultsList.forEach((element) {
    Review currentReview = Review.fromJson(
        element.data() as Map<String, dynamic>, element.reference);
    String formattedSearchText = _searchController.text.toLowerCase();

    if (currentReview.email.toLowerCase().contains(formattedSearchText)) {
      filteredResultsList.add(element);
    }
  });
  setState(() {
    _searchResultsList = filteredResultsList;
  });
}

Widget buildBody(BuildContext context) {
  return StreamBuilder<QuerySnapshot>(
    stream: reviewController.getAllReviews(),
    builder: (context, snapshot) {
      if (snapshot.hasError) {
        return Text('Error ${snapshot.error}');
```

```
      }
      if (snapshot.hasData) {
        // ignore: avoid_print
        print("Document -> ${snapshot.data!.docs.length}");
        _resultsList = snapshot.data!.docs;
        //Renders the user list based on the search criteria
        if (_searchController.text.isEmpty) {
          return buildList(context, _resultsList);
        } else {
          return buildList(context, _searchResultsList);
        }
      }
      return buildList(context, []);
    },
  );
}

//Load list and convert to a list view
Widget buildList(BuildContext context, List<DocumentSnapshot> snapshot) {
  int _currentReviewNumber= 0;

  return ListView(
      children: snapshot
          .map((data) => listItemBuild(context, data, ++_currentReviewNumber))
          .toList());
}

//Load Single Review Object a single item
Widget listItemBuild(
    BuildContext context, DocumentSnapshot data, int reviewNumber) {
```

```
final reviewObj =
    Review.fromJson(data.data() as Map<String, dynamic>, data.reference);
final String formattedReviewNumberText = " " + reviewNumber.toString() + " ";

return Padding(
    key: ValueKey(reviewObj.email),
    padding: const EdgeInsets.symmetric(vertical: 19, horizontal: 1),
    child: Dismissible(
      key: Key(reviewObj.email.toString()+
        Random().nextInt(10000).toString()),
      background: Container(
       color: const Color.fromARGB(255, 32,167,219),
       child: const Padding(
        padding: EdgeInsets.all(15),
        child: Icon(Icons.delete, color: Colors.red, size: 50),
       ),
      ),
      child: Container(
       decoration: BoxDecoration(
        border: Border.all(color: Colors.blue),
        borderRadius: BorderRadius.circular(4),
       ),
       child: SingleChildScrollView(
        child: ListTile(
         title: InkWell(
          child: Column(children: <Widget>[
           Row(children: <Widget>[
            Container(
             child: Text(formattedReviewNumberText,
              style: const TextStyle(color: Colors.white)),
```

```
    decoration: const BoxDecoration(
        borderRadius:
            BorderRadius.all(Radius.circular(5)),
        color: Color.fromARGB(255, 32,167,219)),
    padding: const EdgeInsets.all(2.0),
    margin: const EdgeInsets.only(right: 5.0),
  ),
  Flexible(
      child: SubstringHighlight(
    text: reviewObj.email,
    term: _searchController.text,
    textStyle: const TextStyle(
        // non-highlight style
        color: Colors.black,
        fontSize: 16),
    textStyleHighlight: const TextStyle(
      // highlight style
      color: Colors.black,
      backgroundColor: Colors.yellow,
    ),
  )),
]),
Row(children: <Widget>[
  Container(
    child: const Icon(Icons.email, color: Colors.grey),
    margin: const EdgeInsets.only(right: 3.0),
  ),
  Flexible(
      child: SubstringHighlight(
    text: reviewObj.comment,
```

```
    term: _searchController.text,
    textStyle: const TextStyle(
        // non-highlight style
        color: Colors.black,
        fontSize: 16),
    textStyleHighlight: const TextStyle(
      // highlight style
      color: Colors.black,
      backgroundColor: Colors.yellow,
    ),
  )),
]),
Row(children: <Widget>[
  Container(
    child: const Icon(Icons.star_half_rounded, color: Colors.yellow),
    margin: const EdgeInsets.only(right: 3.0),
  ),
  Flexible(
      child: SubstringHighlight(
    text: reviewObj.ratingValue.toString(),
    term: _searchController.text,
    textStyle: const TextStyle(
        // non-highlight style
        color: Colors.black,
        fontSize: 16),
    textStyleHighlight: const TextStyle(
      // highlight style
      color: Colors.black,
      backgroundColor: Colors.yellow,
    ),
```

```
                    )),
                  ]),
                ]),
              ),
            ),
          ),
        ),
        ));
}

//Build Widget
@override
Widget build(BuildContext context) {
  return Scaffold(
    resizeToAvoidBottomInset: false,
    appBar: AppBar(
      title: Row(
        mainAxisAlignment: MainAxisAlignment.start,
        children: [
          Image.asset(
            'assets/img/Logo.png',
            fit: BoxFit.cover,
            height: 60.0,
          ),
          Container(
            padding: const EdgeInsets.all(8.0),
            child: const Text(
              'Wordly',
              style: TextStyle(fontFamily: 'Righteous', fontSize: 20.0),
            ),
```

```
      )
    ],
  ),
  backgroundColor: const Color.fromARGB(255, 28,150,197),
),
drawer: const MainDrawer(),
body: Container(
  padding: const EdgeInsets.all(19),
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.center,
    mainAxisAlignment: MainAxisAlignment.start,
    children: <Widget>[
      Container(),
      const SizedBox(
        height: 20,
      ),
      Wrap(
          crossAxisAlignment: WrapCrossAlignment.center,
          children: const [
            Icon(Icons.list,
                color: Color.fromARGB(255, 28,150,197), size: 30),
            Text(
              " Review list",
              style: TextStyle(fontSize: 18, fontWeight: FontWeight.w800),
            )
          ]),
      const SizedBox(
        height: 20,
      ),
      TextField(
```

```
            controller: _searchController,
            decoration: InputDecoration(
              prefixIcon: const IconButton(
                color: Colors.black,
                icon: Icon(Icons.search),
                iconSize: 20.0,
                onPressed: null,
              ),
              suffixIcon: IconButton(
                  color: Colors.black,
                  icon: const Icon(Icons.clear),
                  iconSize: 20.0,
                  onPressed: () => _searchController.clear()),
              contentPadding: const EdgeInsets.only(left: 25.0),
              hintText: 'Search',
              border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(4.0))),
          ),
          Flexible(child: buildBody(context))
        ],
      ),
    ),
  );
 }
}
```

4.7 Screens/splash.dart

```
import 'package:animated_splash_screen/animated_splash_screen.dart';
import 'package:flutter/material.dart';
import 'package:page_transition/page_transition.dart';

import 'package:wordly/screens/welcome.dart';
import 'package:wordly/utils/color.dart';

class SplashScreen extends StatefulWidget {
  const SplashScreen({Key? key}) : super(key: key);

  @override
  _SplashScreenState createState() => _SplashScreenState();
}

class _SplashScreenState extends State<SplashScreen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        resizeToAvoidBottomInset: false,
        body: AnimatedSplashScreen(
          duration: 3000,
          splash: 'assets/img/logo.jpg',
          splashIconSize: 160,
          nextScreen: const WelcomeScreen(),
          splashTransition: SplashTransition.sizeTransition,
          pageTransitionType: PageTransitionType.bottomToTop,
```

```
            backgroundColor: purpleColors,
        )
    );
  }
}
```