



LATE BHAUSAHEB HIRAY S.S. TRUST'S INSTITUTE OF COMPUTER APPLICATION

ISO 9001-2008 CERTIFIED

**S.N. 341, Next to New English School, Govt. Colony, Bandra (East),
Mumbai – 400051, Tel: 91-22-26570892/3181**

Date:6/08/2022

CERTIFICATE OF APPROVAL

This is to certify that the project report titled

**“RIDERS CLUB”
(NAVIGATION SYSTEM AND UTILITY)**

**BY
SALIL BHAGAT (202199)**

Under the guidance of

Director

Examiner

Internal Guide

Abstract

Riders around the world are the type of community which need personalized interface for the particular utilities.

This project deals with developing an android application which provides users with navigation system utility, for user to find quickest way to reach its destination.

In order to develop RIDERS CLUB application, a number of Technologies must be studied and understood. These include multi-tiered architecture, server and client-side scripting techniques, implementation technologies such as flutter (dart), google firebase, google maps API, etc.

This is a project with the objective to develop an application where a user is provided with a navigation application and also to know about the technologies used to develop such an application. This document will discuss each of the underlying technologies to create and implement a navigation application.

Acknowledgement

In completing this graduate project, I have been fortunate to have help, support and encouragement from many people. I would like to acknowledge them for their cooperation.

First, I would like to thank **Dr. Minesh Ade and Prof. Sadhana Ojha**, our project advisor, for guiding me through each and every step of the process with knowledge and support.

Thank you for your advice, guidance and assistance.

I would also like to thank **Prof. Prakash Sakharkar and Prof. Divakar Jha**, who have showed immense patience and understanding throughout the project and provided various suggestions.

DECLARATION

I hereby declare that the project entitled, Riders Club done at Hiray College, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university. The project is done in partial fulfillment of the requirements for the award of the degree of **MASTERS OF COMPUTER APPLICATIONS** to be submitted as a second semester project as part of our curriculum.

Name and Signature of the student,
Salil Bhagat

TABLE OF CONTENTS

- 1) Introduction
- 2) Project Description
- 3) Diagrams
- 4) Design And Implementation Of Riders Club
- 5) Application Page every option Details
- 6) Database Design
- 7) Limitation and Future Enhancement
- 8) Conclusion
- 9) References

INTRODUCTION

Riders club is very sophisticated and well developed application, which can be considered as the one of best utility for riding community. As it is basically a navigation system for riders.

The objective of this project is to develop community driven navigation based application which is solely targeted towards professional riding and adventure community and give them personalized interface and utility while riding through one point to other.

Riders club contains google authentication system which provide secure login system to users. Then the user is sent to home page here user is required to drop location pin on the destination where it wants to go.

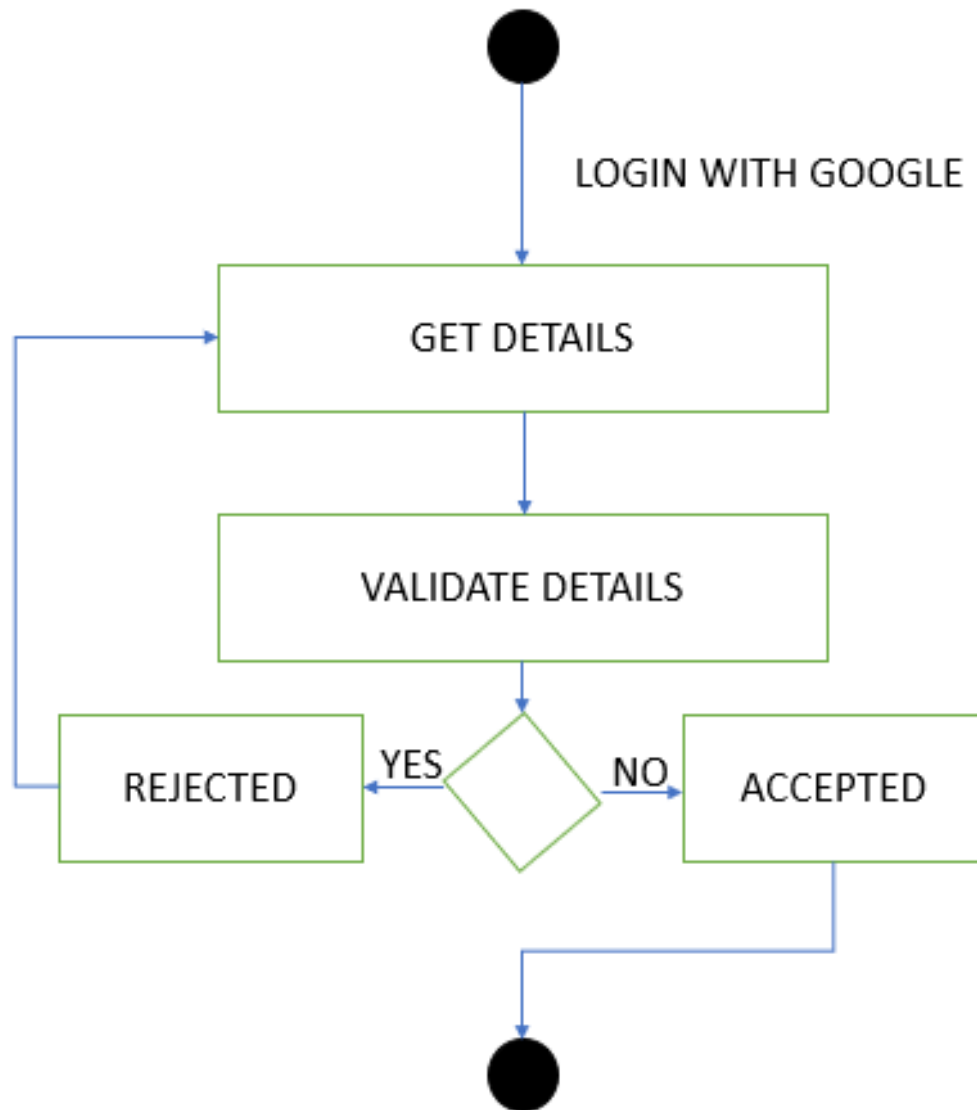
After the destination selection user is shown estimated distance from its current location. Here user is also provided with navigate button. When clicked the user is shown the shortest route to the destination. This is the basic working of Riders club.

DESIGN AND DEVELOPMENT ENVIROMENT

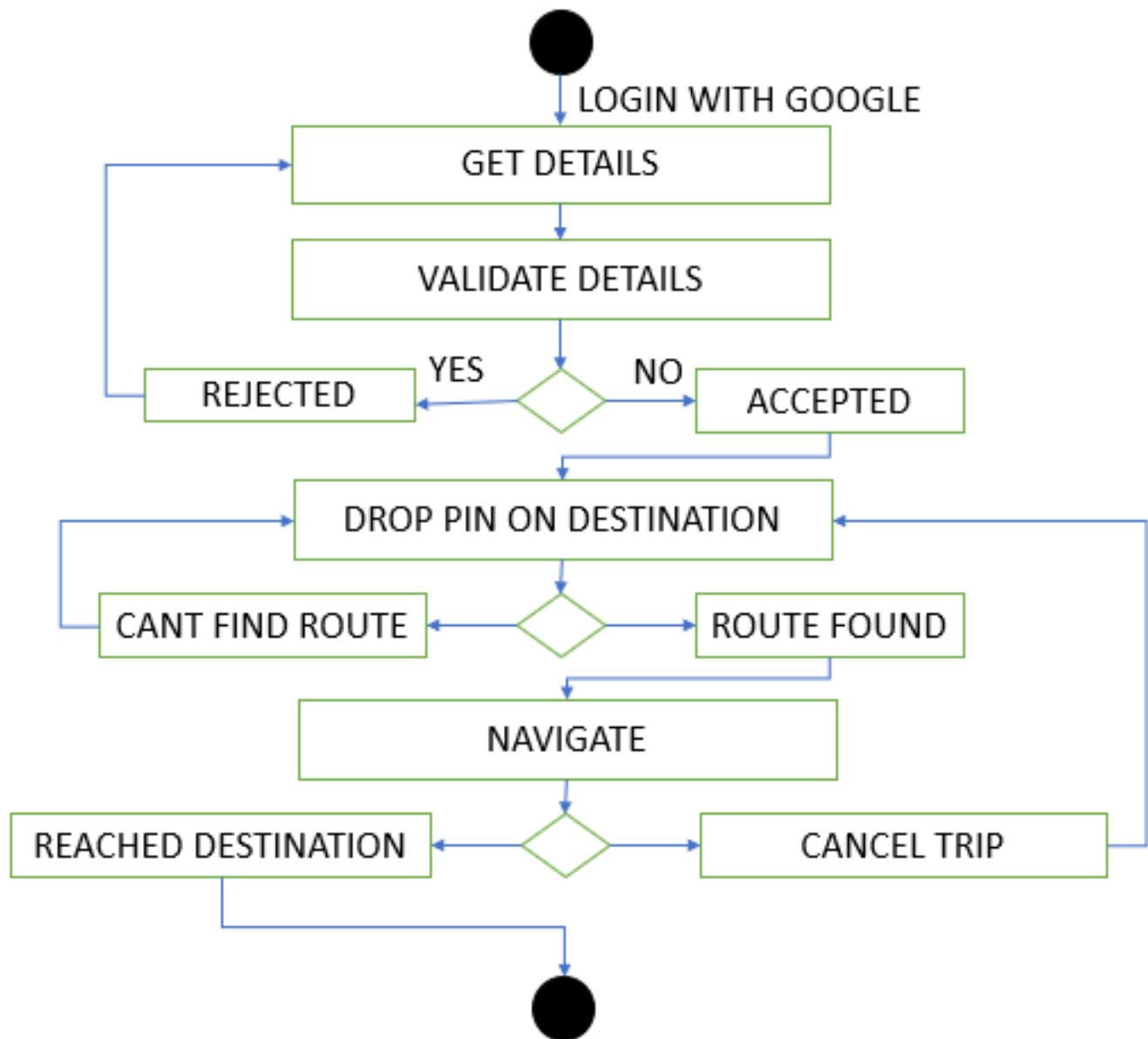
- Flutter
- Google authentication system
- Google maps API
- Visual studio code

DIAGRAMS

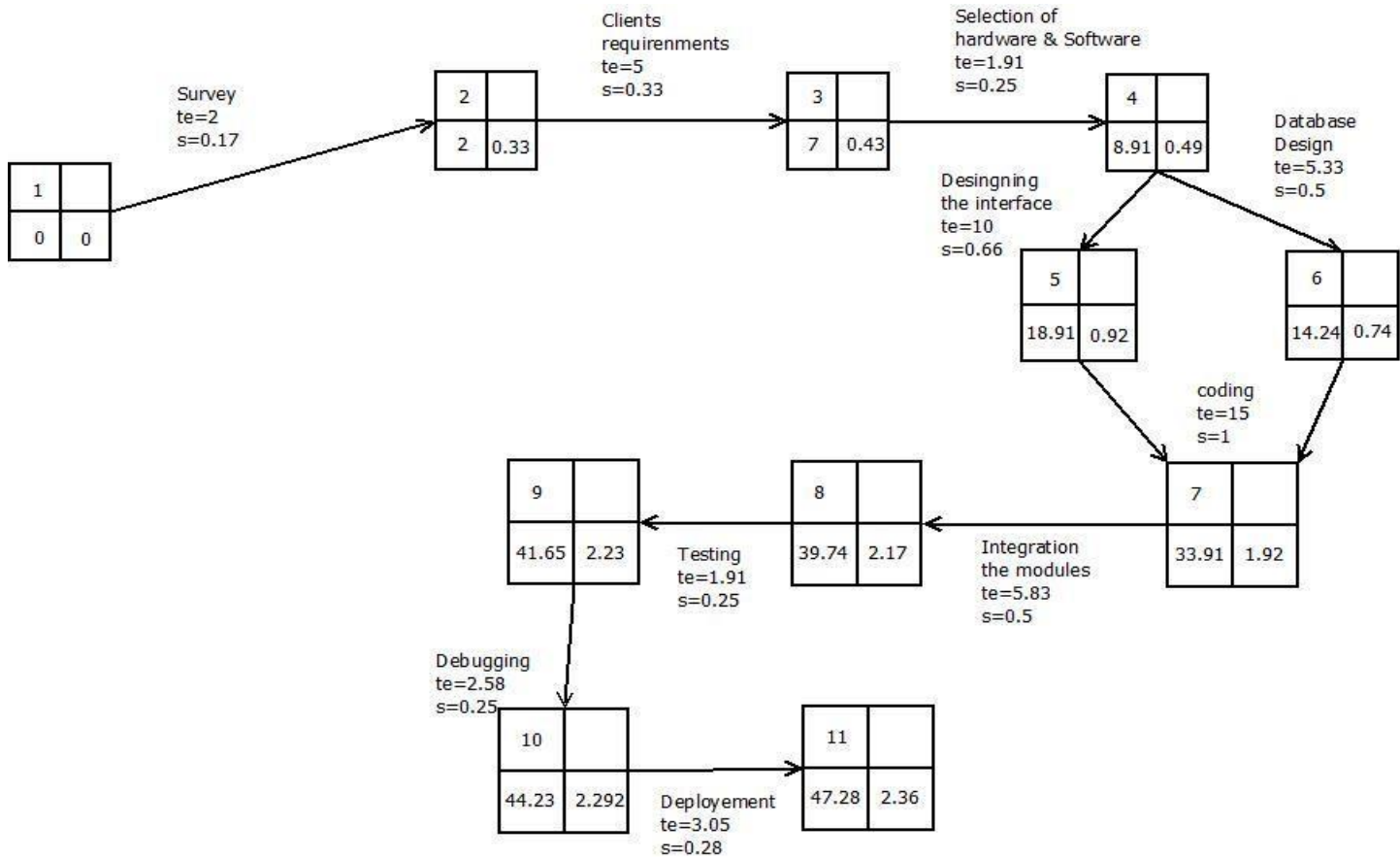
Login Activity:



User Activity Diagram:



Pert Chart:



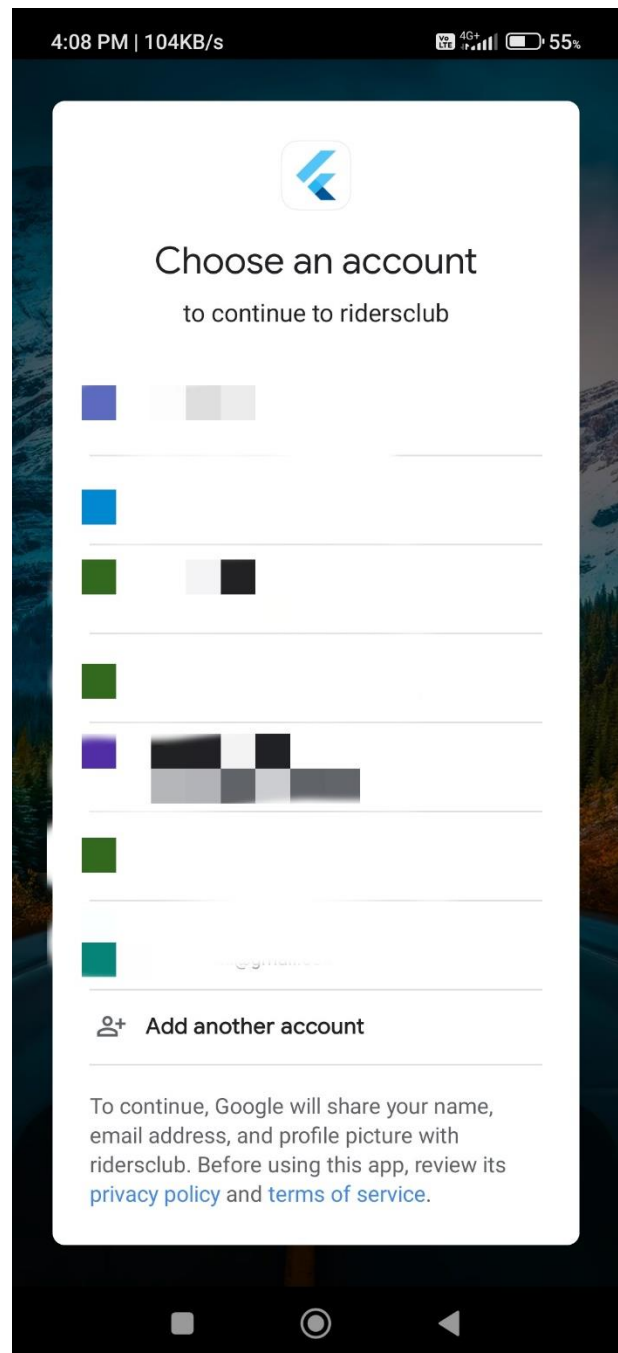
Pert Chart Calculation:

Activities	Optimistic	Most likely	Pessimistic	Expected time	Standard Deviation
Survey	1	2	3	2	0.33
Client's requirements	4	5	6	5	0.33
Selection of hardware & Software	1	2	2.5	1.91	0.25
Designing the Interface	8	1	12	10	0.66
Database Design	4	6	7	5.33	0.5
Coding	12	15	18	15	1
Integrating the modules	4	6	7	5.83	0.5
Testing	1	2	2.5	1.91	0.25
Debugging	2	2.5	3.5	2.58	0.25
Deployment	2.3	3	4	3.05	0.28

Login Page:

This is login Page.

1. The login module allow user to login into system using google authentication system.
2. In login page user has to authenticate using google authentication interface.
3. User must have his intended google account in system or add one in such case.
4. Algorithm:-
Step1: The User will press sign in with google.
Step2: account will be authenticated.



Running code:-

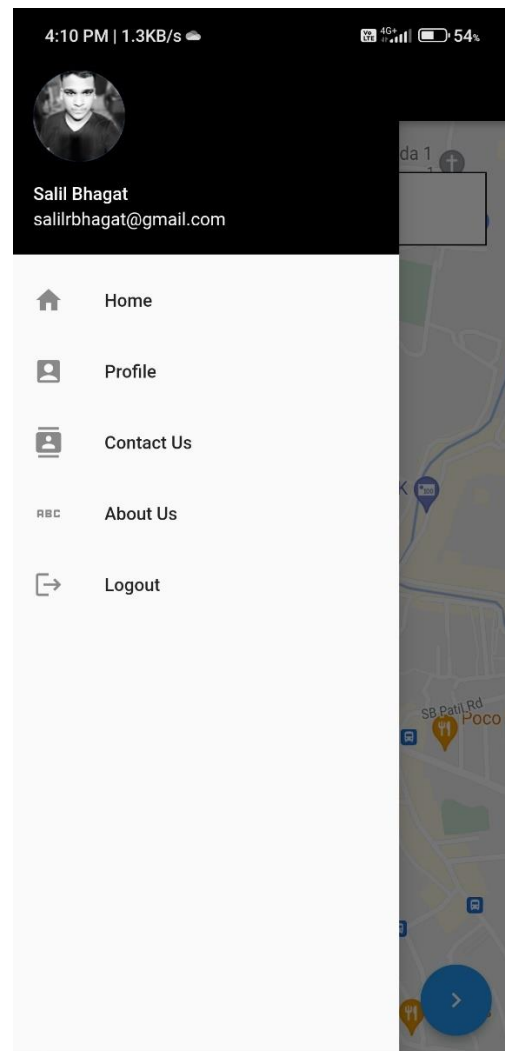
```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:sign_button/sign_button.dart';
import 'auth_service.dart';

class LoginPage extends StatefulWidget {
  const LoginPage({Key? key}) : super(key: key);
  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  @override
  Widget build(BuildContext context) {
    final Size size = MediaQuery.of(context).size;
    return Scaffold(
      body: Container(
        decoration: BoxDecoration(
          image: DecorationImage(
            image: AssetImage("assets/images/road.jpg"),
            fit: BoxFit.cover,
          ),
        ),
      child: Stack(
        children: [
          Container(
            margin: const EdgeInsets.fromLTRB(30, 0, 30, 250),
            child: Center(
              child: CircleAvatar(
                backgroundImage: AssetImage('assets/images/logo.gif'),
                radius: 100,
              ),
            ),
          ),
          Container(
            padding: EdgeInsets.symmetric(horizontal: 40, vertical: 35),
            margin:
              EdgeInsets.only(top: MediaQuery.of(context).size.height * 0.85),
            width: double.infinity,
            height: 450,
            decoration: BoxDecoration(
              color: const Color(0x000000).withOpacity(0.5),
              borderRadius: BorderRadius.only(
                topRight: Radius.circular(50),
                topLeft: Radius.circular(50))),
            child: Column(
              children: [
                Row(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    SignInButton(
                      buttonType: ButtonType.google,
```

Home Page:
This is the home page or welcome page.
In home page it displays map where we have to drop the destination location.
It also contains menu slider.

This is the home page or welcome page.



Running Code:

```
class HomePage extends StatefulWidget {
  @override
  _HomePageState createState() => _HomePageState();
}
class _HomePageState extends State<HomePage> {
  LatLng? destLocation = LatLng(19.0760, 72.8777);
  Location location = Location();
  loc.LocationData? _currentPosition;
  final Completer<GoogleMapController?> _controller = Completer();
  String? _address;
  //String? user = FirebaseAuth.instance.currentUser!.email ??
  FirebaseAuth.instance.currentUser!.displayName;
  @override
  void initState() {
    // TODO: implement initState
    super.initState();
    getCurrentLocation();
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Riders Club',
        ),
        backgroundColor: Colors.black,
      ),
      drawer: Drawer(
        child: ListView(
          // Important: Remove any padding from the ListView.
          padding: EdgeInsets.zero,
          children: <Widget>[
            GestureDetector(
              onTap: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => ProfilePage()),
                );
              },
            ),
            child: UserAccountsDrawerHeader(
              decoration: BoxDecoration(
                color: Colors.black,
              ),
              accountName: Text(
                FirebaseAuth.instance.currentUser!.displayName!,
              ),
              accountEmail: Text(FirebaseAuth.instance.currentUser!.email!),
              currentAccountPicture: CircleAvatar(
                child: Image.network(
```

```

        FirebaseAuth.instance.currentUser!.photoURL!,
    ),
),
),
),
ListTile(
  leading: Icon(Icons.home),
  title: Text("Home"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => HomePage()),
    );
  },
),
ListTile(
  leading: Icon(Icons.account_box),
  title: Text("Profile"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => ProfilePage()),
    );
  },
),
ListTile(
  leading: Icon(Icons.contacts),
  title: Text("Contact Us"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => ContactusPage()),
    );
  },
),
ListTile(
  leading: Icon(Icons.abc_outlined),
  title: Text("About Us"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => AboutusPage()),
    );
  },
),
ListTile(
  leading: Icon(Icons.logout),
  title: Text("Logout"),
  onTap: () {
    AuthService().signOut();
    Navigator.pop(context);
  },
),

```

```

        },
    ),
],
),
),
floatingActionButton: FloatingActionButton(
    child: Icon(Icons.navigate_next),
    onPressed: () {
        Navigator.of(context).pushAndRemoveUntil(
            MaterialPageRoute(
                builder: (context) => NavigationScreen(
                    destLocation!.latitude, destLocation!.longitude),
            ),
            (route) => false);
    }),
body: Stack(
    children: [
        GoogleMap(
            zoomControlsEnabled: false,
            initialCameraPosition: CameraPosition(
                target: destLocation!,
                zoom: 16,
            ),
            onCameraMove: (CameraPosition? position) {
                if (destLocation != position!.target) {
                    setState(() {
                        destLocation = position.target;
                    });
                }
            },
            onCameraIdle: () {
                print('camera idle');
                getAddressFromLatLng();
            },
            onTap: (latLng) {
                print(latLng);
            },
            onMapCreated: (GoogleMapController controller) {
                _controller.complete(controller);
            },
        ),
        Align(
            alignment: Alignment.center,
            child: Padding(
                padding: const EdgeInsets.only(bottom: 35.0),
                child: Image.asset(
                    'assets/images/pick.png',
                    height: 45,
                    width: 45,
                ),
            ),
        ),
    ],
),

```

```

    ),
    Positioned(
      top: 40,
      right: 20,
      left: 20,
      child: Container(
        decoration: BoxDecoration(
          border: Border.all(color: Colors.black),
          color: Colors.white,
        ),
        padding: EdgeInsets.all(20),
        child: Text(_address ?? 'Pick your destination address',
          overflow: TextOverflow.visible, softWrap: true),
      ),),),),));}
getAddressFromLatLng() async {
  try {
    GeoData data = await Geocoder2.getDataFromCoordinates(
      latitude: destLocation!.latitude,
      longitude: destLocation!.longitude,
      googleMapApiKey: "*****");
    setState(() {
      _address = data.address;
    });
  } catch (e) {
    print(e);
  }
}
getCurrentLocation() async {
  bool _serviceEnabled;
  PermissionStatus _permissionGranted;

  _serviceEnabled = await location.serviceEnabled();
  final GoogleMapController? controller = await _controller.future;

  if (!_serviceEnabled) {
    _serviceEnabled = await location.requestService();
    if (!_serviceEnabled) {
      return;
    }
  }
  _permissionGranted = await location.hasPermission();
  if (_permissionGranted == PermissionStatus.denied) {
    _permissionGranted = await location.requestPermission();
    if (_permissionGranted != PermissionStatus.granted) {
      return;
    }
  }
  if (_permissionGranted == loc.PermissionStatus.granted) {
    location.changeSettings(accuracy: loc.LocationAccuracy.high);
    _currentPosition = await location.getLocation();
    controller?.animateCamera(CameraUpdate.newCameraPosition(CameraPosition(

```



```

        target:
            LatLng(_currentPosition!.latitude!, _currentPosition!.longitude!),
            zoom: 16,
        ));
        setState(() {
            destLocation =
                LatLng(_currentPosition!.latitude!, _currentPosition!.longitude!);
        });
    }
}
}

```

Profile Page:

This shows profile of user wher user can see its name ,picture and unique code



Running code:

```
class ProfilePage extends StatefulWidget {
  @override
  _ProfilePageState createState() => _ProfilePageState();
}
class _ProfilePageState extends State<ProfilePage> {
  //String? user = FirebaseAuth.instance.currentUser!.email ??
  FirebaseAuth.instance.currentUser!.displayName;
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: const Text(
          'Riders Club',
        ),
        backgroundColor: Colors.black,
      ),
      drawer: Drawer(
        child: ListView(
          // Important: Remove any padding from the ListView.
          padding: EdgeInsets.zero,
          children: <Widget>[
            GestureDetector(
              onTap: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => ProfilePage()),
                );
              },
            ),
            UserAccountsDrawerHeader(
              decoration: BoxDecoration(
                color: Colors.black,
              ),
              accountName: Text(
                FirebaseAuth.instance.currentUser!.displayName!,
              ),
              accountEmail: Text(FirebaseAuth.instance.currentUser!.email!),
              currentAccountPicture: CircleAvatar(
                child: Image.network(
                  FirebaseAuth.instance.currentUser!.photoURL!,
                ),
              ),
            ),
            ListTile(
              leading: Icon(Icons.home),
              title: Text("Home"),
              onTap: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => HomePage()),
                );
              },
            ),
          ],
        ),
      ),
    );
  }
}
```

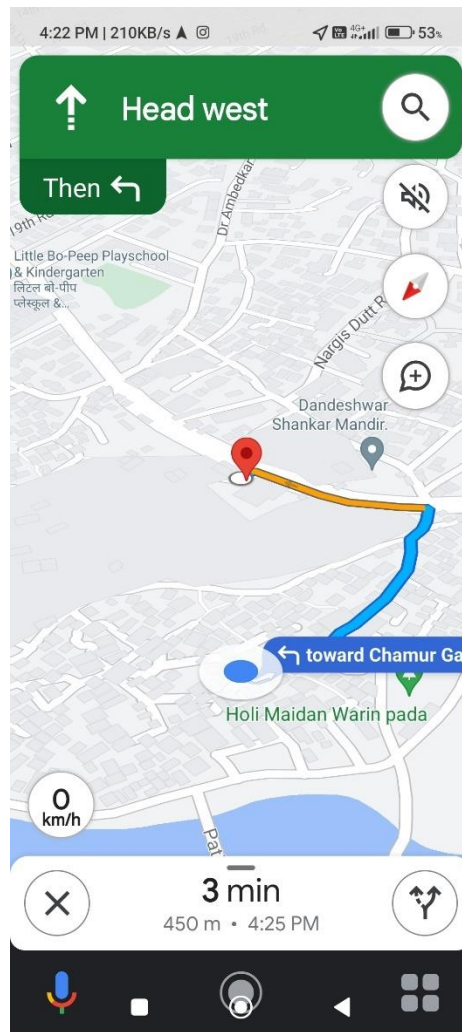
```

ListTile(
  leading: Icon(Icons.account_box),
  title: Text("Profile"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => ProfilePage()),
    );
  },
),
ListTile(
  leading: Icon(Icons.contacts),
  title: Text("Contact Us"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => ContactusPage()),
    ); },),
ListTile(
  leading: Icon(Icons.abc_outlined),
  title: Text("About Us"),
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => AboutusPage()),
    ); },),
ListTile(
  leading: Icon(Icons.logout),
  title: Text("Logout"),
  onTap: () {
    AuthService().signOut();
    Navigator.pop(context);
  },),],),),
body: Container(
  color: Colors.white,
  width: MediaQuery.of(context).size.width,
  child: Column(
    mainAxisAlignment: MainAxisAlignment.center,
    children: <Widget>[
      Image.network(
        FirebaseAuth.instance.currentUser!.photoURL!,
        height: 200,
        width: 200,
      ),
      Text(
        FirebaseAuth.instance.currentUser!.displayName!,
        style: const TextStyle(
          fontSize: 30,
          fontWeight: FontWeight.bold,
          color: Colors.black),),
    ],
  ),
)

```

```
const SizedBox(
  height: 10,
),
Text(
  FirebaseAuth.instance.currentUser!.email!,
  style: const TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold,
    color: Colors.black),
),
const SizedBox(
  height: 20,
),
Text(
  FirebaseAuth.instance.currentUser!.uid,
  style: const TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold,
    color: Colors.black),
),
const SizedBox(
  height: 20,
),
Text(
  FirebaseAuth.instance.currentUser!.phoneNumber!,
  style: const TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold,
    color: Colors.black),),),
Container(
  padding: EdgeInsets.symmetric(horizontal: 40, vertical: 35),
  margin: EdgeInsets.only(
    top: MediaQuery.of(context).size.height * 0.30),
  child: Column(
    children: [
      Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          MaterialButton(
            padding: const EdgeInsets.all(10),
            height: 50,
            minWidth: 250,
            color: Colors.black,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(50)),
            child: const Text(
              'LOGOUT',
              style: TextStyle(color: Colors.white, fontSize: 15)),),
          onPressed: () {
            AuthService().signOut();
            Navigator.pop(context);},),),],),),),],),),),);}}
```

Navigation Screen:



Code:

```
class NavigationScreen extends StatefulWidget {
  final double lat;
  final double lng;
  NavigationScreen(this.lat, this.lng);

  @override
  State<NavigationScreen> createState() => _NavigationScreenState();
}
class _NavigationScreenState extends State<NavigationScreen> {
  final Completer<GoogleMapController?> _controller = Completer();
  Map<PolylineId, Polyline> polylines = {};
  PolylinePoints polylinePoints = PolylinePoints();
  Location location = Location();
  Marker? sourcePosition, destinationPosition;
  loc.LocationData? _currentPosition;
  LatLng curLocation = LatLng(19.0760, 72.8777);
  StreamSubscription<loc.LocationData>? locationSubscription;
  @override
  void initState() {
    // TODO: implement initState
    super.initState();
    getNavigation();
  }
}
```

```

    addMarker();
  }
  @override
  void dispose() {
    locationSubscription?.cancel();
    super.dispose();
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: sourcePosition == null
        ? Center(child: CircularProgressIndicator())
        : Stack(
            children: [
              GoogleMap(
                zoomControlsEnabled: false,
                polylines: Set<Polyline>.of(polylines.values),
                initialCameraPosition: CameraPosition(
                  target: curLocation,
                  zoom: 16,
                ),
                markers: {sourcePosition!, destinationPosition!},
                onTap: (latLng) {
                  print(latLng);
                },
                onMapCreated: (GoogleMapController controller) {
                  _controller.complete(controller);
                },
              ),
              Positioned(
                top: 30,
                left: 15,
                child: GestureDetector(
                  onTap: () {
                    Navigator.of(context).pushAndRemoveUntil(
                      MaterialPageRoute(builder: (context) => MyApp()),
                      (route) => false);
                  },
                  child: Icon(Icons.arrow_back),
                ),
              ),
              Positioned(
                bottom: 10,
                right: 10,
                child: Container(
                  width: 50,
                  height: 50,
                  decoration: BoxDecoration(
                    shape: BoxShape.circle, color: Colors.blue),
                  child: Center(
                    child: IconButton(
                      icon: Icon(

```

```

        Icons.navigation_outlined,
        color: Colors.white,
      ),
      onPressed: () async {
        await launchUrl(Uri.parse(
          'google.navigation:q=
${widget.lat}, ${widget.lng}&key=*****'
        ));
      },
    ),
  ),
))
],
),
);
}
getNavigation() async {
  final GoogleMapController? controller = await _controller.future;
  location.changeSettings(accuracy: loc.LocationAccuracy.high);
  _currentPosition = await location.getLocation();
  curLocation =
    LatLng(_currentPosition!.latitude!, _currentPosition!.longitude!);
  locationSubscription =
    location.onLocationChanged.listen((LocationData currentLocation) {
      controller?.animateCamera(CameraUpdate.newCameraPosition(CameraPosition(
        target: LatLng(currentLocation.latitude!, currentLocation.longitude!),
        zoom: 16,
      )));
    });
  if (mounted) {
    controller
      ?.showMarkerInfoWindow(MarkerId(sourcePosition!.markerId.value));
    setState(() {
      curLocation =
        LatLng(currentLocation.latitude!, currentLocation.longitude!);
      sourcePosition = Marker(
        markerId: MarkerId(currentLocation.toString()),
        icon:
          BitmapDescriptor.defaultMarkerWithHue(BitmapDescriptor.hueBlue),
        position:
          LatLng(currentLocation.latitude!, currentLocation.longitude!),
        infoWindow: InfoWindow(
          title:
            '${double.parse((getDistance(LatLng(widget.lat,
widget.lng)).toStringAsFixed(2)))} km',
          onTap: () {
            print('market tapped');
          },
        ),
      );
    });
    getDirections(LatLng(widget.lat, widget.lng));
  }
});
}
}

```

```

getDirections(LatLng dst) async {
  List<LatLng> polylineCoordinates = [];
  List<dynamic> points = [];
  PolylineResult result = await polylinePoints.getRouteBetweenCoordinates(
    '*****',
    PointLatLng(curLocation.latitude, curLocation.longitude),
    PointLatLng(dst.latitude, dst.longitude),
    travelMode: TravelMode.driving);
  if (result.points.isNotEmpty) {
    result.points.forEach((PointLatLng point) {
      polylineCoordinates.add(LatLng(point.latitude, point.longitude));
      points.add({'lat': point.latitude, 'lng': point.longitude});
    });} else {
    print(result.errorMessage);
  }
  addPolyLine(polylineCoordinates);
}

addPolyLine(List<LatLng> polylineCoordinates) {
  PolylineId id = PolylineId('poly');
  Polyline polyline = Polyline(
    polylineId: id,
    color: Colors.blue,
    points: polylineCoordinates,
    width: 5,    );
  polylines[id] = polyline;
  setState(() {});
}

double calculateDistance(lat1, lon1, lat2, lon2) {
  var p = 0.017453292519943295;
  var c = cos;
  var a = 0.5 -
    c((lat2 - lat1) * p) / 2 +
    c(lat1 * p) * c(lat2 * p) * (1 - c((lon2 - lon1) * p)) / 2;
  return 12742 * asin(sqrt(a));
}

double getDistance(LatLng destposition) {
  return calculateDistance(curLocation.latitude, curLocation.longitude,
    destposition.latitude, destposition.longitude);
}

addMarker() {
  setState(() {
    sourcePosition = Marker(
      markerId: MarkerId('source'),
      position: curLocation,
      icon: BitmapDescriptor.defaultMarkerWithHue(BitmapDescriptor.hueAzure),);
    destinationPosition = Marker(
      markerId: MarkerId('destination'),
      position: LatLng(widget.lat, widget.lng),

      icon: BitmapDescriptor.defaultMarkerWithHue(BitmapDescriptor.hueCyan),
    ); });}}

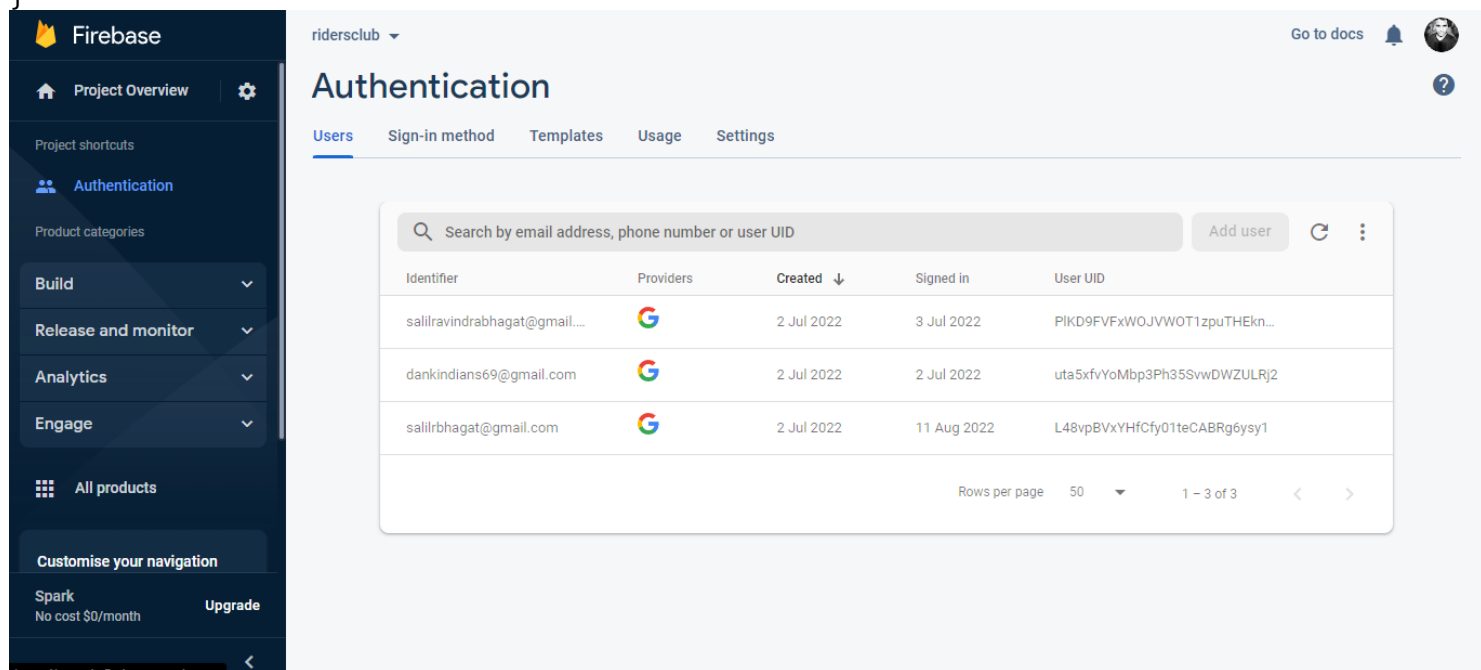
```


Database Design:-

```
class AuthService {
  handleAuthState() {
    //return widget
    return StreamBuilder(
      stream: FirebaseAuth.instance.authStateChanges(),
      builder: (BuildContext context, snapshot) {
        if (snapshot.hasData) {
          return HomePage();
        } else {
          return const LoginPage();
        }
      }
    );
  }

  signInWithGoogle() async {
    final GoogleSignInAccount? googleUser =
      await GoogleSignIn(scopes: <String>["email"]).signIn();
    final GoogleSignInAuthentication googleAuth =
      await googleUser!.authentication;
    final credential = GoogleAuthProvider.credential(
      accessToken: googleAuth.accessToken,
      idToken: googleAuth.idToken,
    );
    return await FirebaseAuth.instance.signInWithCredential(credential);
  }

  signOut() {
    FirebaseAuth.instance.signOut();
  }
}
```



The screenshot displays the Firebase Authentication console interface. On the left is a dark sidebar with the Firebase logo and navigation options: Project Overview, Authentication (selected), Build, Release and monitor, Analytics, Engage, and All products. The main content area is titled 'Authentication' and includes tabs for Users, Sign-in method, Templates, Usage, and Settings. The 'Users' tab is active, showing a table of users with columns for Identifier, Providers, Created, Signed in, and User UID. There are three users listed, all using Google as a provider. At the bottom of the sidebar, there is a 'Customise your navigation' section with 'Spark' (No cost \$0/month) and an 'Upgrade' button.

Identifier	Providers	Created ↓	Signed in	User UID
salilravindrabhagat@gmail...	Google	2 Jul 2022	3 Jul 2022	PIKD9FVFxWOJVWOT1zpuTHEkn...
dankindians69@gmail.com	Google	2 Jul 2022	2 Jul 2022	uta5xfvYoMbp3Ph35SvwDWZULRj2
salilrbhagat@gmail.com	Google	2 Jul 2022	11 Aug 2022	L48vpBVxYHfCfy01teCABRg6ysy1

Rows per page: 50 | 1 - 3 of 3

Limitations :-

- Need internet connection i.e. no offline mode available.
- Limited no. Of functionalities.
- Reliance on google API and ecosystem.
- Needs polished UI.

Future Enhancement :-

- Include more reliable logic to backend code.
- Complete the full automation of application.
- Improve UI.
- More functions.
- More map types.
- Debug minor bugs.

Conclusion:

The internet itself is very crucial in developing various communities. This mainly focuses on riding community. It is very important to provide the demand and vacuum present in the community. Riders club is the utility which bridges the gap between the supply and demand with its personalized interface. Rider's club has huge potential in terms of growth and can be converted into profit making contender by various means.

Reference:

- www.google.com
- www.wikipedia.com
- www.youtube.com
- **Software project management book.**