COLLEGE BUS TRACKING HYBRID APP USING ADVANCED GOOGLE CLOUD SERVICES AND IOT

Team Members

1. Mr. S.A.K. Jainulabudeen, M.Tech., Assistant Professor
2. Nafeez Abbas J
3. Naresh D
4. Salman Khan S

COLLEGE BUS TRACKING HYBRID APP USING ADVANCED GOOGLE CLOUD SERVICES AND IOT

ABSTRACT

The current generation requires information on a regular basis. The use of technology has been steadily growing. As a result, we devised an innovative solution to "College Bus Tracking hybrid app using advanced Google cloud services and IOT," combining current technologies with the need for knowledge transmission. The device consists of a Mobile Application for students to use and a Hardware setup that shares the bus's location. The Hardware system uses IOT which includes an Arduino UNO, GSM, GPS modules, etc. The Hardware is setup in the bus such that its location can be accessed. The mobile app uses third party cloud services to access this location information, allowing students to know where they are and make plans accordingly.

I. INTRODUCTION

Vehicle tracking systems are widely used by fleet operators for fleet management functions such as establishing contact between the college server and the bus system, which is capable of delivering real-time data about the current location of buses, sending group messages, such as warning messages to students waiting at the next stop, changes in the current route, bus number, and so on, saving time of the students. A vehicle tracking system is an electronic device that is mounted in a vehicle which allows the owner or a third party to monitor the location of the vehicle. It works by employing GPS and GSM technologies to continuously track a moving vehicle. A microcontroller is serially connected to a GSM modem and GPS receiver, which is used to transmit the vehicle's position (latitude and longitude) from a remote location.

The first fully operational GPS/Loran-based vessel monitoring system keeps track of the workstation, communications, and onboard navigation systems, giving the marine fleet operator a complete image. In applications such as shipping, scheduling, harbour operations, and route verification, the device is a valuable tool for the fleet operator. Furthermore, this concept can be extended to the broader issue of secure hazardous cargo transportation.

This project is implemented using Android Studio with Flutter Framework and the Firebase. The application will automatically view maps and routes to various locations using the GPS system, as well as monitor the bus position using client-server technology and forward it to the client computer. It uses simple distance measurements between two locations and provides necessary route information so that people can easily board buses on the designated route. The user is given specific location information as well as the bus number so that the student can correctly locate the bus. The data is stored on a remote server. As a result, records can be easily manipulated on the computer itself, reducing the server load.

II. PROBLEM DEFINITION

Many colleges do use paper and archives to keep track of bus routes and schedules, as well as provide information by notifications, which is ineffective. As a result, a structured method of keeping records and providing information as needed is required. Furthermore, students are unaware of the bus's proper timing. Some students wait for the bus being unaware of that the bus had already been misses and they are late for the class. Therefore, a smart system is required that provides remote users with real-time bus information. As a result, we proposed a new system to address the shortcomings of the college transportation system.

III. SYSTEM ARCHITECTURE

3.1 GPS TRACKING UNIT ARCHITECTURE

A GPS tracking unit is a device that calculates and monitors its precise location, and thus that of its carrier, at regular intervals using the Global Positioning System. It is usually carried by a moving vehicle or human. The captured location data can be stored in the tracking device or transmitted via a cellular (GPRS or SMS), radio, or satellite modem embedded in the unit to a central location data base or an Internet-connected computer. This allows the asset's location to be displayed against a map backdrop in real time or later when analysing the track with GPS tracking software. Data tracking software is available for smartphones equipped with GPS.

In order to obtain the GPS signal and measure the coordinates, a GPS tracker must have a GPS module. Data loggers have a large memory to store the coordinates, while data pushers have a GSM/GPRS modem to send the information to a central computer via SMS or GPRS in the form of IP packets.

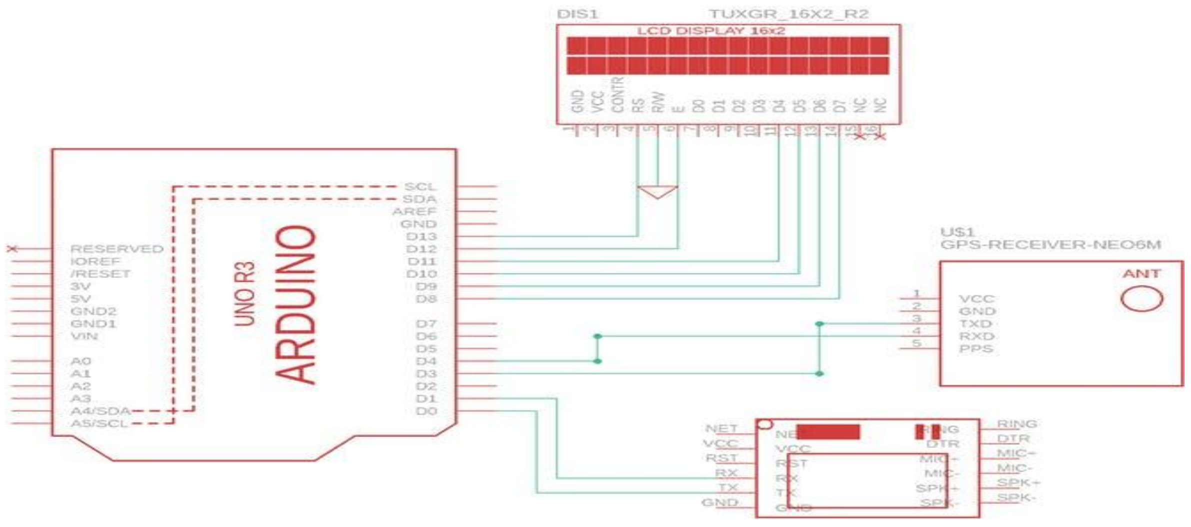
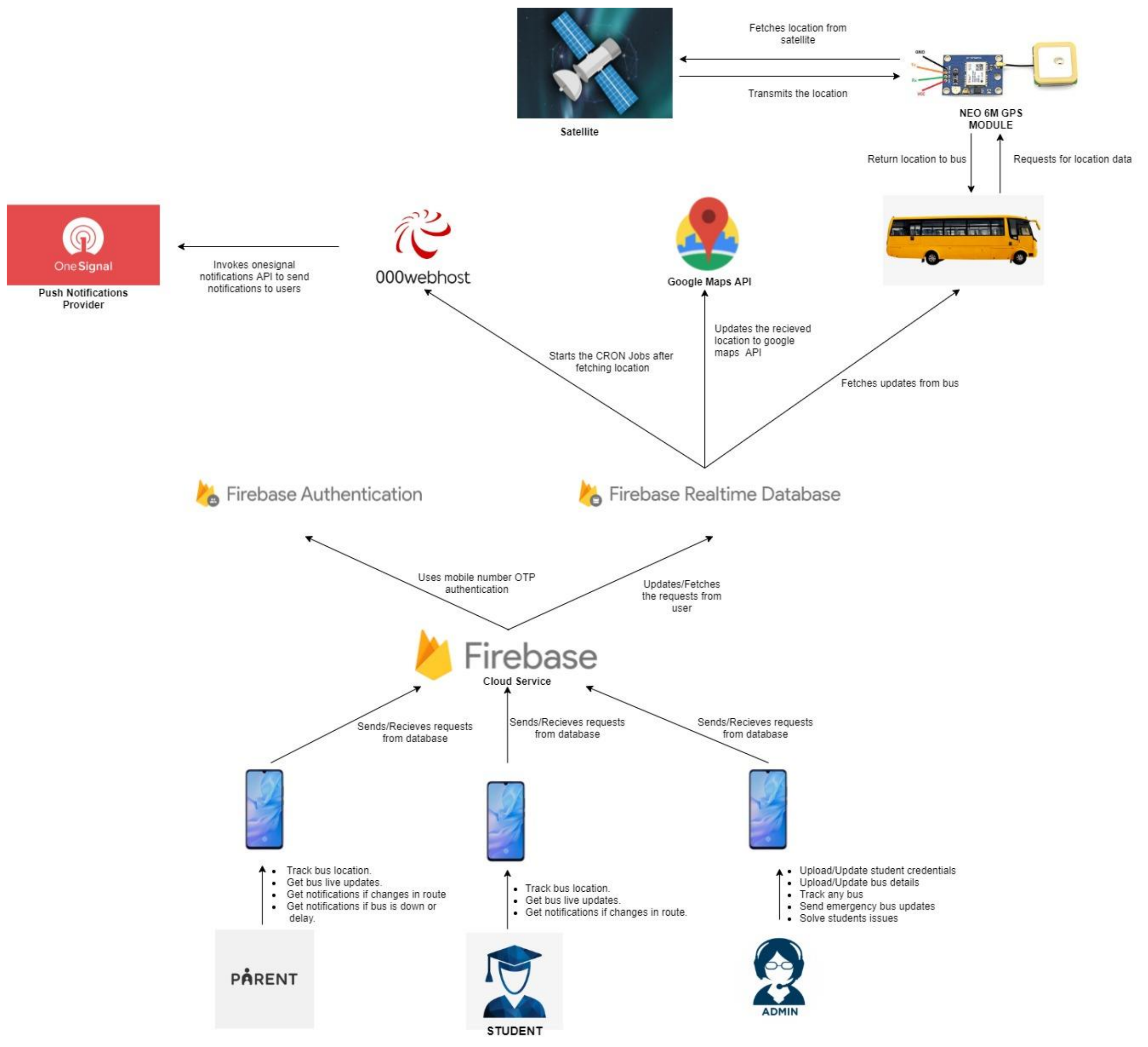


Figure 3.1 GPS Tracking Unit Architecture (Internet Source)

3.2 APPLICATION ARCHITECTURE

The GPS coordinates get stored in the firebase database which are then accessed by the application. The student can login into the app and has the privilege of selecting a route and bus from the list. The app shows the location of the bus on a map upon successful selections. The Admin on the other hand has access to the student details and bus locations. He / She uploads and manages the student data in the database and also can view the location of buses one at a time.



Fetches location from satellite
Transmits the location
Satellite
NEO 6M GPS MODULE
Return location to bus
Requests for location data
OneSignal
Push Notifications Provider
Invokes onesignal notifications API to send notifications to users
000webhost
Google Maps API
Starts the CRON Jobs after fetching location
Updates the recieved location to google maps API
Fetches updates from bus
Firebase Authentication
Firebase Realtime Database
Uses mobile number OTP authentication
Updates/Fetches the requests from user
Firebase
Cloud Service
Sends/Recieves requests from database
Sends/Recieves requests from database
Sends/Recieves requests from database
Track bus location.
Get bus live updates.
Get notifications if changes in route
Get notifications if bus is down or delay.
Track bus location.
Get bus live updates.
Get notifications if changes in route.
Upload/Update student credentials
Upload/Update bus details
Track any bus
Send emergency bus updates
Solve students issues
PARENT
STUDENT
ADMIN

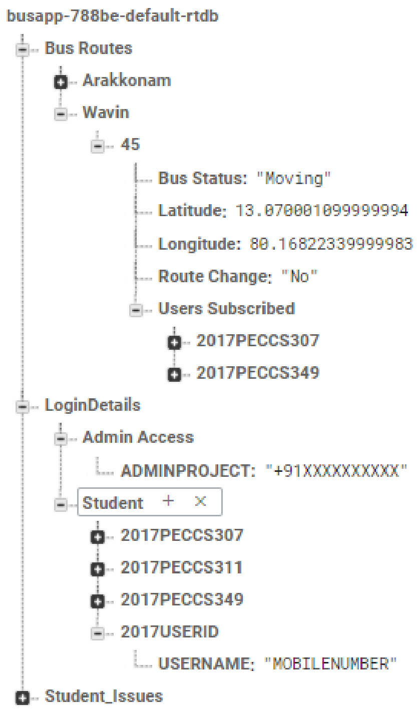
Figure 3.2 System Architecture

3.3 FIREBASE ARCHITECTURE

Firebase has two major sections

1) Bus Routes
2) Login Details

Bus Routes lists all available routes and the buses under each route. Each bus has its latitude and longitude stored and its status. Login Details contains two sections. Admin Access has the mobile number of the admin. Student section contains details all the students. It includes the students Name, Roll Number and Mobile Number.



```
busapp-788be-default-rtdb
  Bus Routes
    Arakkonam
    Wavin
      45
        Bus Status: "Moving"
        Latitude: 13.070001099999994
        Longitude: 80.16822339999983
        Route Change: "No"
        Users Subscribed
          2017PECCS307
          2017PECCS349
  LoginDetails
    Admin Access
      ADMINPROJECT: "+91XXXXXXXXXX"
    Student
      2017PECCS307
      2017PECCS311
      2017PECCS349
      2017USERID
        USERNAME: "MOBILENUMBER"
  Student_Issues
```

Figure 3.3 Firebase Architecture

IV. WORKING PROCEDURE

4.1 OVERVIEW

On installation or boot up, the location-based bus tracking device server should automatically start. The server database will then compare the current latitude and longitude value to the previous latitude and longitude value. The database manager manages the application's database, after which the user searches for the bus and retrieves information from the database for that specific location. The application then acts in accordance with the user interface, allowing the user to add, delete, and update the database.

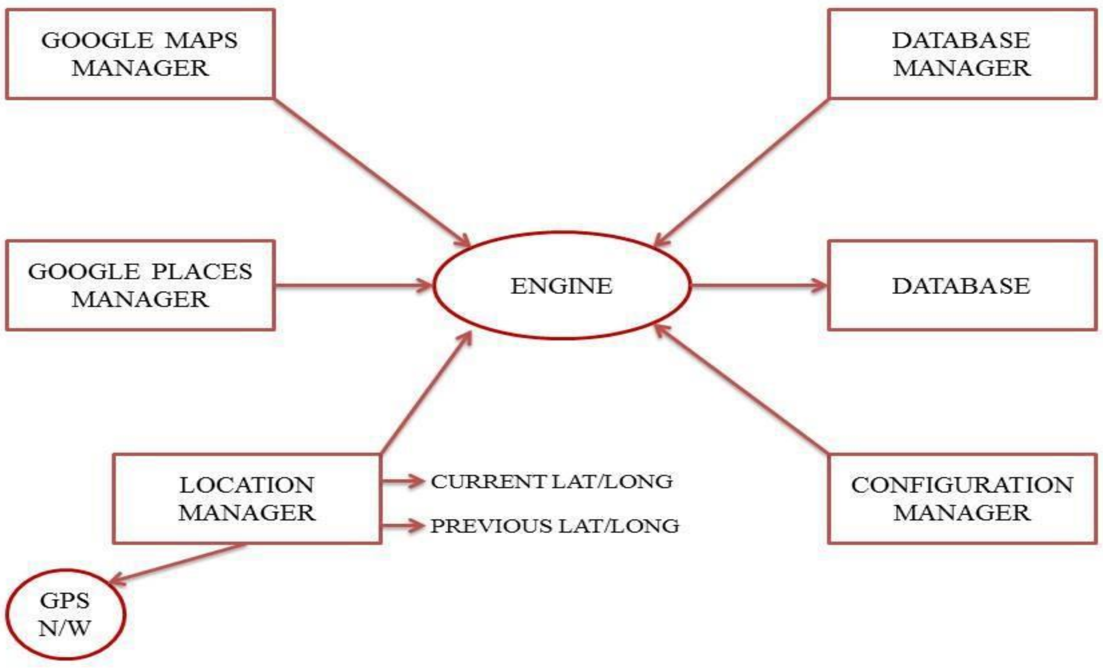


Figure 4.1 Working of a Location Based Service (LBS) Component (Internet Source)

4.2 IMPLEMENTATION

The app is developed using Flutter SDK. Flutter is a new open-source technology from Google that allows you to build native Android and iOS apps with a single codebase. The reactive development architecture is followed by Flutter, but with a twist. The most important thing to understand about reactive programming is that it automatically updates UI content when we change variables in the code. This idea is followed by React Native, but it makes use of the JavaScript bridge to access OEM widgets. However, since the app must cross this bridge each time it accesses a widget, it creates performance issues. Flutter, on the other hand, skips the bridge entirely and interacts with the native platform through Dart.

Project is designed in four modules where each module is responsible for different aspects.

Module 1 – Admin

The admin logs into the system with user name and password, the admin manages routes, students and drivers. Admin can add and update the students details and the driver details as well. Also, the Admin manages the list of routes and the buses available in each route. All the information is made persistent in the database. Admin manages the routes and makes changes in it if necessary. The Admin also has the access to the location of the buses where he / she can choose the Bus Number and view its current location using the Google Maps API.

Module 2 – Student / Parent

Student can login to the system using the Unique Register Number provided to him / her by the college and the mobile number submitted. Upon successful login, the student will be prompted to select the route and bus number to track it. Students can view the location of bus in real-time on the map. Students can also set an alarm which starts when the bus is nearing him. Student receives updates on route and bus changes as well.

Module 3 - Firebase

Firebase provides the NOSQL database for storing the student and bus record. Lot of API's and function calls have been used in order to create, read, update, delete (CRUD) operations for user and admin data. Other than that firebase also provides push notification facility to our app. Adding to this, another third-party provider called as 000webhost.com used to update the bus co-ordinates from the Arduino UNO to database.

Module 4 - Hardware

This module tracks and updates the bus coordinates to the database through API calls. It needs a proper working sim card with data transfer facility (GPRS). The GPS coordinates are made persistent in a third-party cloud using the GSM. The GSM makes an HTTP request to the Cloud and hence stores the coordinates which are in turn fetched at the Mobile App. Google Map API uses these coordinated to show the location of the bus.

V. CONCLUSION

The findings of this study show that having a good understanding of a particular domain will help you get better results. This Project has been implemented on Hybrid platform. In addition, various attributes have been applied to the project that will benefit the system. The specifications and criteria are described above. This project is implemented using Android Studio with Flutter Framework and the Firebase. The application will automatically view maps and routes to various locations using the GPS system, as well as monitor the bus position using client-server technology and forward it to the client computer. It uses simple distance measurements between two locations and provides necessary route information so that people can easily board buses on the designated route. The user is given specific location information as well as the bus number so that the student can correctly locate the bus. It uses remote server as its database. As a result, records can be easily manipulated on the computer itself, reducing the server load.

CODING

1. **main.dart**

```
import 'package:bus_app/splashscreen.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';
/* Calling main function here */
Future<void> main() async {
    runApp(MaterialApp(title:"Find My Bus",home: SplashScreen(),));
}
```

2. **splashscreen.dart**

```
import 'package:bus_app/admin_dashboard.dart';
import 'package:bus_app/loginPage.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'dashboardUI.dart';
class SplashScreen extends StatefulWidget {
  @override
  _SplashScreenState createState() => _SplashScreenState();
}
class _SplashScreenState extends State<SplashScreen> {
  String _debugLabelString = "";
  String _emailAddress;
  String _externalUserId;
  bool _enableConsentButton = false;
```

```
// CHANGE THIS parameter to true if you want to test GDPR privacy
consent
bool _requireConsent = true;
@override
void initState() {
  super.initState();
  initPlatformState();
  Future<void> main_function() async{
    try {
      WidgetsFlutterBinding.ensureInitialized();
      await Firebase.initializeApp();
      /* Checking if user is already logged in to the app */
      SharedPreferences prefs = await SharedPreferences.getInstance();
      bool routeSelected =
      await prefs.getBool("routeSelected") == null ? false : true;
      String userId = await prefs.getString("userId");
      bool isAdmin = await prefs.getBool("accessLevel");
      bool new_login = await prefs.getBool("new_user");
      bool externaluserId = await prefs.getBool("externaluserId");
      final FirebaseAuth _auth = FirebaseAuth.instance;
      int authcheck;
    try {
     /* Checking if user session is valid */
      authcheck = _auth.currentUser.phoneNumber.length;
    } catch (e) {
      if(externaluserId==true) {
        await prefs.remove("externaluserId");
        await OneSignal.shared.removeExternalUserId();
      }
      authcheck = 0;
    }
  /* Initial app flow begins from here */
    runApp(MaterialApp(
        home: authcheck == 0 && userId == null
            ? LoginPage()
            : isAdmin == true
            ? AdminDashboard()
            : DashBoardUI(
            routeSelected: routeSelected, roll_number: userId)));
    }
    catch(E)
    {
      Fluttertoast.showToast(msg: "Check your internet connection");
    }
  }
  main_function();
 }
/* Inititialzing Onesignal platform */
 Future<void> initPlatformState() async {
  if (!mounted) return;
  OneSignal.shared.setLogLevel(OSLogLevel.verbose,
  OSLogLevel.none);
  var settings = {
    OSiOSSettings.autoPrompt: false,
    OSiOSSettings.promptBeforeOpeningPushUrl: true
  };
  OneSignal.shared.setNotificationReceivedHandler((OSNotification
  notification) {
    this.setState(() {
      _debugLabelString =
      "Received notification:
      \n${notification.jsonRepresentation().replaceAll("\\n", "\n")}";
    });
  });
  OneSignal.shared
  .setNotificationOpenedHandler((OSNotificationOpenedResult result) {
    this.setState(() {
      _debugLabelString =
      "Opened notification:
      \n${result.notification.jsonRepresentation().replaceAll("\\n", "\n")}";
    });
  });
  OneSignal.shared
      .setInAppMessageClickedHandler((OSInAppMessageAction action)
  {
    this.setState(() {
      _debugLabelString =
      "In App Message Clicked:
       \n${action.jsonRepresentation().replaceAll("\\n", "\n")}";
    });
  });
  OneSignal.shared
      .setSubscriptionObserver((OSSubscriptionStateChanges changes) {
    print("SUBSCRIPTION STATE CHANGED:
    ${changes.jsonRepresentation()}");
  });
  OneSignal.shared.setPermissionObserver((OSPermissionStateChanges
  changes) {
    print("PERMISSION STATE CHANGED:
    ${changes.jsonRepresentation()}");
```

```
    });
    OneSignal.shared.setEmailSubscriptionObserver(
            (OSEmailSubscriptionStateChanges changes) {
          print("EMAIL SUBSCRIPTION STATE CHANGED
          ${changes.jsonRepresentation()}");
        });
    await OneSignal.shared
        .init("your app id", iOSSettings: settings);
    OneSignal.shared
        .setInFocusDisplayType(OSNotificationDisplayType.notification);
  }
  @override
  Widget build(BuildContext context) {
    return Container(
      color:  Color(4281356286),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
        children: [
          Container(
            child: Image.asset("assets/pecLogo.png"),
          ),
          CircularProgressIndicator(backgroundColor: Colors.white)
        ],
      ),
    );
  }
}
```

3. **loginPage.dart**

```
import 'dart:async';
import 'package:bus_app/admin_dashboard.dart';
import 'package:bus_app/dashboardUI.dart';
import 'package:bus_app/help.dart';
import 'package:bus_app/otp_screen.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:keyboard_avoider/keyboard_avoider.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:shared_preferences/shared_preferences.dart';
class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}
class _LoginPageState extends State<LoginPage>
    with TickerProviderStateMixin, RelativeScale {
  AnimationController controller;
  final FirebaseAuth _auth = FirebaseAuth.instance;
  String _rollNumber;
  bool isLoading = false;
  String _verificationId;
  bool isbuttonVisible = false;
  final GlobalKey<FormState> _formkey = GlobalKey<FormState>();
  Animation<double> _animationforLogo, _animationforCard;
  TextEditingController textEditingController = TextEditingController();
```

```
@override
void initState() {
  super.initState();
  /* Initializing all the animations used in sign in page */
  controller = AnimationController(
      duration: const Duration(milliseconds: 500), vsync: this)
    ..repeat();
  _animationforLogo = Tween<double>(begin: 0, end:
  .6).animate(controller);
  _animationforCard = Tween<double>(
    begin: 0,
    end: .8,
  ).animate(controller);
  controller.forward();
}
@override
Widget build(BuildContext context) {
  initRelativeScaler(context);
  return WillPopScope(
    onWillPop: () {
      showDialog(
          context: this.context,
          child: AlertDialog(
            content: Column(
              mainAxisSize: MainAxisSize.min,
              children: [
                Text("Do you want to exit the app?"),
                Row(
                  mainAxisSize: MainAxisSize.min,
                  mainAxisAlignment: MainAxisAlignment.end,
                  crossAxisAlignment: CrossAxisAlignment.end,
                  children: [
                    FlatButton(
                      child: Text("Yes"),
                      onPressed: () => SystemNavigator.pop(),
                    ),
                    FlatButton(
                      child: Text("No"),
                      onPressed: () => Navigator.pop(context, true),
                    ),
                  ],
                )
              ],
            ),
          ));
    },
    child: Stack(
      children: [
        Container(
          color: Colors.white,
        ),
        Container(
          height: sy(370),
          color: Color(4281356286),
        ),
        Scaffold(
          resizeToAvoidBottomInset: false,
          body: KeyboardAvoider(
            autoScroll: true,
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.center,
              children: [
                /* App Logo here */
                ScaleTransition(
                    scale: _animationforLogo,
                    child: Image.asset("assets/pecLogo.png")),
                /* Form Contents here */
                ScaleTransition(
                  scale: _animationforCard,
                  child: Container(
                    height: (sy(65) + 80) > (sx(65) + 80)
                        ? (sy(65) + 80)
                        : (sx(65) + 80),
                    width: sy(500) > sx(500) ? sy(500) : sx(500),
                    child: Card(
                      shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(10.0),
                      ),
                      color: Colors.white,
                      elevation: 10,
                      child: Column(
                        children: [
                          Container(
                            margin: EdgeInsets.all(
                                sy(20) > sx(20) ? sy(20) : sx(20)),
                            child: Form(
                              key: _formkey,
                              child: TextFormField(
                                controller: textEditingController,
                                textCapitalization:
```

```
    TextCapitalization.characters,
maxLength: 12,
onChanged: (input) {
  if (input.length == 12) {
    setState(() {
      isbuttonVisible = true;
    });
  } else {
    setState(() {
      isbuttonVisible = false;
    });
  }
},
onSaved: (input) => _rollNumber = input,
textAlignVertical: TextAlignVertical.center,
decoration: InputDecoration(
  counterText: "",
  focusedBorder: OutlineInputBorder(
    borderSide:
        BorderSide(color: Color(4281356286)),
    borderRadius: BorderRadius.circular(50),
  ),
  enabledBorder: OutlineInputBorder(
    borderRadius: BorderRadius.circular(50),
    borderSide:
        BorderSide(color: Color(4281356286)),
  ),
  prefixIcon: Icon(
    Icons.person,
    color: Color(4281356286),
            ),
            border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(50),
            ),
            labelText: "Roll Number",
            labelStyle:
                TextStyle(color: Color(4281356286)),
          ),
        ),
      ),
    ),
    Container(
      margin: EdgeInsets.only(bottom: 10),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          // Sign In Button
          isLoading == false
              ? isbuttonVisible == true
                  ? ClipOval(
                      child: Material(
                        color: Color(4281356286),
                        child: InkWell(
                          onTap: signIn,

                          child: SizedBox(
                            height: sy(25) > sx(25)
                                ? sy(25)
                                : sx(25),
                            width: sy(25) > sx(25)
```

```
                    ? sy(25)
                    : sx(25),
                child: Icon(
                  Icons
                      .keyboard_arrow_right_sharp,
                  color: Colors.white,
                ),
              ),
            ),
          ),
        )
      : ClipOval(
          child: Material(
            color: Colors.black26,
            child: InkWell(
              child: SizedBox(
                height: sy(25) > sx(25)
                    ? sy(25)
                    : sx(25),
                width: sy(25) > sx(25)
                    ? sy(25)
                    : sx(25),
                child: Icon(
                  Icons
                      .keyboard_arrow_right_sharp,
                  color: Colors.white,
                ),
              ),
            ),
          ),
          )
      : Container(
          height:
              sy(20) > sx(20) ? sy(20) : sx(20),
          width:
              sy(20) > sx(20) ? sy(20) : sx(20),
          child: CircularProgressIndicator(
            backgroundColor: Color(4281356286),
          )),
  // Help Button
  Container(
    margin: EdgeInsets.only(left: 40),
    child: ClipOval(
      child: Material(
        color: Color(4281356286),
        child: InkWell(
          onTap: () {
            Navigator.push(
                context,
                MaterialPageRoute(
                    builder: (context) =>
                        Help()));
          },
          child: SizedBox(
            height: sy(25) > sx(25)
                ? sy(25)
                : sx(25),
            width: sy(25) > sx(25)
                ? sy(25)
                : sx(25),
                                          child: Icon(
                                            Icons.help,
                                            color: Colors.white,
                                          ),
                                        ),
                                      ),
                                    ),
                                  ),
                                )
                              ],
                            ),
                          )
                        ],
                      ),
                    ),
                  ),
                ),
              ],
              mainAxisAlignment: MainAxisAlignment.center,
            ),
          ),
          backgroundColor: Colors.transparent,
        )
      ],
    ),
  );
}
Future<void> signIn() async {
  setState(() {
    isLoading = true;
```

```
});
final formstate = _formkey.currentState;
formstate.save();
_rollNumber = _rollNumber.toUpperCase();
FocusScope.of(context).requestFocus(FocusNode());
int newOtp = 1;
SharedPreferences prefs = await SharedPreferences.getInstance();
String rollNumber = await prefs.getString("RollNumber");
String VerificationId = await prefs.getString("VerificationId");
String phoneNumber = await prefs.getString("PhoneNumber");
int resendToken = await prefs.getInt("ResendToken");
String savedTime = await prefs.getString("Time");
String username = await prefs.getString("userName");
if (savedTime != null) {
  newOtp = 0;
  final time = DateTime.parse(savedTime);
  if ((!(time.difference(DateTime.now()).isNegative)) &&
      (_rollNumber == rollNumber)) {
    Fluttertoast.showToast(
        msg: "Enter your recently sent OTP",
        toastLength: Toast.LENGTH_SHORT,
        gravity: ToastGravity.BOTTOM,
        timeInSecForIosWeb: 1,
        backgroundColor: Colors.white,
        textColor: Colors.black,
        fontSize: 16.0);
    Navigator.push(
        context,
        MaterialPageRoute(
            builder: (context) => Otp(
                  verificationId: VerificationId,
                  mobile: phoneNumber,
                  resendToken: resendToken,
                  rollNumber: _rollNumber,
                  username: username,
                 )));
      } else {
        newOtp = 1;
        await prefs.remove("RollNumber");
        await prefs.remove("VerificationId");
        await prefs.remove("PhoneNumber");
        await prefs.remove("ResendToken");
        await prefs.remove("Time");
      }
    }
    if (newOtp == 1) {
      Map<dynamic, dynamic> values, adminLogins, studentLogins,
userMobile;
      var db = FirebaseDatabase.instance.reference();
      await db.once().then((DataSnapshot snap) {
        values = snap.value;
        values = values['LoginDetails'];
        adminLogins = values['Admin Access'];
        studentLogins = values['Student'];
      });
      if (adminLogins.containsKey(_rollNumber)) {
        String number = adminLogins[_rollNumber].toString();
        await prefs.setString("userName", "ADMIN");
        await verifyPhoneNumber(number, _rollNumber, "ADMIN");
      } else if (studentLogins.containsKey(_rollNumber)) {
        userMobile = studentLogins[_rollNumber];
        String number = userMobile["Mobile"];
        String userName = userMobile["Name"];
        await prefs.setString("userName", userName);
        await verifyPhoneNumber(number, _rollNumber, userName);
      } else {
        setState(() {
          isLoading = false;
        });
        Fluttertoast.showToast(
            msg: "Unknown Login credentials",
            toastLength: Toast.LENGTH_SHORT,
            gravity: ToastGravity.BOTTOM,
            timeInSecForIosWeb: 1,
            backgroundColor: Colors.lightBlue,
            textColor: Colors.black,
            fontSize: 16.0);
      }
    }
  }
  void verifyPhoneNumber(
      String number, String rollNumber, String userName) async {
    SharedPreferences prefs = await SharedPreferences.getInstance();
    void showSnackbar(String message) {
      Fluttertoast.showToast(msg: message, toastLength:
Toast.LENGTH_LONG);
    }
    PhoneVerificationCompleted verificationCompleted =
        (PhoneAuthCredential phoneAuthCredential) async {
      await _auth.signInWithCredential(phoneAuthCredential);
      setState(() {
        isLoading = false;
      });
      showSnackbar("Phone number automatically verified");
      SharedPreferences prefs = await SharedPreferences.getInstance();
      await prefs.setString("userId", _rollNumber);
      await prefs.setString("userName", userName);
      if (userName == "ADMIN") {
        await prefs.setBool("accessLevel", true);
        Navigator.push(
            context, MaterialPageRoute(builder: (context) =>
AdminDashboard()));
      } else {
        Navigator.push(
            context,
            MaterialPageRoute(
                builder: (context) => DashBoardUI(
                    routeSelected: false, roll_number: _rollNumber)));
      }
    };
    PhoneVerificationFailed verificationFailed =
        (FirebaseAuthException authException) {
      showSnackbar(
          'Phone number verification failed. ${authException.message}');
      setState(() {
        isLoading = false;
      });
    };
    PhoneCodeSent codeSent =
        (String verificationId, [int forceResendingToken]) async {
      showSnackbar('Please check your phone for the verification code.');
      _verificationId = verificationId;
      await prefs.setString("VerificationId", verificationId);
```

```
  await prefs.setString("RollNumber", _rollNumber);
  await prefs.setInt("ResendToken", forceResendingToken);
  await prefs.setString(
      "Time", DateTime.now().add(Duration(minutes: 5)).toString());
  await prefs.setString("PhoneNumber", number);
  setState(() {
    isLoading = false;
  });
  Navigator.push(
      context,
      MaterialPageRoute(
          builder: (context) => Otp(
                verificationId: _verificationId,
                mobile: number,
                resendToken: forceResendingToken,
                rollNumber: _rollNumber,
                username: userName,
              )));
};
PhoneCodeAutoRetrievalTimeout codeAutoRetrievalTimeout =
    (String verificationId) {
  _verificationId = verificationId;
};
try {
  await _auth.verifyPhoneNumber(
    phoneNumber: number,
    timeout: const Duration(seconds: 60),
    verificationCompleted: verificationCompleted,
    verificationFailed: verificationFailed,
    codeSent: codeSent,
        codeAutoRetrievalTimeout: codeAutoRetrievalTimeout,
      );
    } catch (e) {
      showSnackbar("Failed to Verify Phone Number:");
      setState(() {
        isLoading = false;
      });
    }
  }
}
```

4. **otp_screen.dart**

```
import 'dart:async';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:flutter/material.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'admin_dashboard.dart';
import 'dashboardUI.dart';
class Otp extends StatefulWidget {
  const Otp({Key key, @required this.verificationId,@required
this.mobile,@required this.resendToken, @required
this.rollNumber,@required this.username}) : super(key: key);
  final String verificationId,mobile, rollNumber,username;
  final int resendToken;
  @override
  _OtpState createState() => _OtpState(verificationId:
verificationId,mobile: mobile,resendToken: resendToken, rollNumber:
rollNumber,username: username);
}
```

```
class _OtpState extends State<Otp> {
  _OtpState({Key key, @required this.verificationId,@required
this.mobile,@required this.resendToken, @required
this.rollNumber,@required this.username});
  final FirebaseAuth _auth = FirebaseAuth.instance;
  String verificationId, mobile, rollNumber,username;
  int resendToken;
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  Color otp = Color.fromRGBO(48, 79, 254, .1);
  Color otpBorder = Color.fromRGBO(231, 239, 246, .7);
  FocusNode pin2FocusNode;
  FocusNode pin3FocusNode;
  FocusNode pin4FocusNode;
  FocusNode pin5FocusNode;
  FocusNode pin6FocusNode;
  int secondsLeft = 0;
  int secondsRight = 0;
  int minute = 1;
  bool resend = false;
  Timer _timer;
  String OTP = '';
  final textController1 = TextEditingController();
  final textController2 = TextEditingController();
  final textController3 = TextEditingController();
  final textController4 = TextEditingController();
  final textController5 = TextEditingController();
  final textController6 = TextEditingController();
  final _formKey = GlobalKey<FormState>();
  void nextField({String value, FocusNode focusNode}) {
    if (value.length == 1) {
      focusNode.requestFocus();
    }
  }
  void signInWithPhoneNumber(String _otp) async{
    void showSnackbar(String message) {
      Fluttertoast.showToast(msg: message);
    }
    try {
      showSnackbar("Successfully signed in");
      SharedPreferences prefs = await SharedPreferences.getInstance();
      await prefs.remove("RollNumber");
      await prefs.remove("VerificationId");
      await prefs.remove("PhoneNumber");
      await prefs.remove("ResendToken");
      await prefs.remove("Time");
      await prefs.setString("userId", rollNumber);
      await prefs.setString("userName", username);
      if(username=="ADMIN")
      {
        await prefs.setBool("accessLevel", true);
        Navigator.push(context, MaterialPageRoute(builder: (context) =>
            AdminDashboard()));
      }
      else
      {
        Navigator.push(context, MaterialPageRoute(builder:
(context)=>DashBoardUI(routeSelected: false, roll_number:
rollNumber)));
```

```
      }
     } catch (e) {
      showSnackbar("Failed to sign in: " + e.toString());
      AlertDialog alert = AlertDialog(
        title: Text('Login Error'),
        content: Text(e.toString()),
      );
      showDialog(
        context: context,
        builder: (BuildContext context) {
          return alert;
        },
      );
    }
  }
  void validateOtp(val) {
    if (val.length == 6) {
        signInWithPhoneNumber(OTP);
    } else {
      {
        AlertDialog alert = AlertDialog(
          title: Text('Data Invalid'),
          content: Text('Enter a valid OTP!'),
        );
        showDialog(
          context: context,
          builder: (BuildContext context) {
            return alert;
          },
        );
      }
    }
  }
  void validationHandler() {
    AlertDialog alert = AlertDialog(
      title: Text('Data Invalid'),
      content: Text('OTP must be a number'),
    );
    showDialog(
      context: context,
      builder: (BuildContext context) {
        return alert;
      },
    );
  }
  void startTimer() {
    _timer = Timer.periodic(Duration(seconds: 1), (timer) {
      setState(() {
        if (minute == 1) {
          minute--;
          secondsLeft = 5;
          secondsRight = 9;
        } else {
          if (secondsRight == 0 && secondsLeft == 0) {
            setState(() {
              resend = true;
            });
            _timer.cancel();
          } else {
            if (secondsRight == 0) {
              secondsLeft--;
              secondsRight = 9;
            } else {
              secondsRight--;
            }
          }
        }
      });
    });
  }
  void resendCodeHandler(String number) async{
    SharedPreferences prefs = await SharedPreferences.getInstance();
    void showSnackbar(String message) {
      Fluttertoast.showToast(msg:
  message,toastLength:Toast.LENGTH_LONG);
    }
    PhoneVerificationCompleted verificationCompleted =
        (PhoneAuthCredential phoneAuthCredential) async {
      await _auth.signInWithCredential(phoneAuthCredential);
      showSnackbar("Phone number automatically verified");

      SharedPreferences prefs = await SharedPreferences.getInstance();
      await prefs.setString("userId", rollNumber);
      await prefs.setString("userName", username);
      if(username=="ADMIN")
      {
        await prefs.setBool("accessLevel", true);
        Navigator.push(context, MaterialPageRoute(builder: (context) =>
            AdminDashboard()));
      }
      else
```

```
      {
        Navigator.push(context, MaterialPageRoute(builder:
(context)=>DashBoardUI(routeSelected: false, roll_number:
rollNumber)));


      }




    };
    PhoneVerificationFailed verificationFailed =
        (FirebaseAuthException authException) {
      showSnackbar('Phone number verification failed. Code:
${authException.code}. Message: ${authException.message}');
    };
    PhoneCodeSent codeSent =
        (String VerificationId, [int forceResendingToken]) async {
      showSnackbar('Please check your phone for the verification code.');
      verificationId=VerificationId;
      await prefs.remove("VerificationId");
      await prefs.remove("ResendToken");
      await prefs.remove("Time");
      await prefs.setString("VerificationId",verificationId);
      setState(() {
        resendToken=forceResendingToken;
      });
      await prefs.setInt("ResendToken", forceResendingToken);
      await prefs.setString("Time",DateTime.now().add(Duration(minutes:
5)).toString());
    };
    PhoneCodeAutoRetrievalTimeout codeAutoRetrievalTimeout =
        (String verificationId) {
    };
    try {
      await _auth.verifyPhoneNumber(
        phoneNumber: number,
        timeout: const Duration(seconds: 60),
        verificationCompleted: verificationCompleted,
        verificationFailed: verificationFailed,
        codeSent: codeSent,
        codeAutoRetrievalTimeout: codeAutoRetrievalTimeout,
        forceResendingToken: resendToken
      );
      startTimer();
    } catch (e) {
      showSnackbar("Failed to Verify Phone Number: ${e}");
    }
  }
  void onPressHandler() {
    if (resend) {
      minute = 1;
      secondsLeft = 0;
      secondsRight = 0;
      setState(() {
        resend = false;
      });
      resendCodeHandler(mobile);
    }
  }
  @override
  void initState() {
    super.initState();
    startTimer();
    pin2FocusNode = FocusNode();
    pin3FocusNode = FocusNode();
    pin4FocusNode = FocusNode();
    pin5FocusNode = FocusNode();
    pin6FocusNode = FocusNode();
    textController1.addListener(() { OTP = ""; });
    textController2.addListener(() { OTP = ""; });
    textController3.addListener(() { OTP = ""; });
    textController4.addListener(() { OTP = ""; });
    textController5.addListener(() { OTP = ""; });
    textController6.addListener(() { OTP = ""; });
  }
  void dispose() {
    _timer.cancel();
    pin2FocusNode.dispose();
    pin3FocusNode.dispose();
    pin4FocusNode.dispose();
    pin5FocusNode.dispose();
    pin6FocusNode.dispose();
    textController1.dispose();
    textController2.dispose();
    textController3.dispose();
    textController4.dispose();
    textController5.dispose();
    textController6.dispose();
    super.dispose();
  }
  bool _isNumeric(String val) {
    return double.tryParse(val) != null;
```

```
}
@override
Widget build(BuildContext context) {
  return RelativeBuilder(
      builder: (context, screenHeight, screenWidth, sy, sx) {
        return Scaffold(
          resizeToAvoidBottomPadding: false,
          appBar: AppBar(
            centerTitle: true,
            title: Text(
              'Verify',
              style: TextStyle(fontSize: sy(15) > sx(15) ? sy(15) : sx(15)),
            ),
            backgroundColor: primary,
          ),
          body: Container(
            height: screenHeight,
            width: screenWidth,
            color: background,
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.center,
              children: [
                Flexible(
                  child: ListView(
                    children: [
                      Padding(
                          padding: EdgeInsets.symmetric(
                              horizontal: 0,
                              vertical: sy(25) > sx(25) ? sy(25) : sx(25))),
                      Image.asset(
                        'assets/user-interface.png',
                        width: sy(75) > sx(75) ? sy(75) : sx(75),
                        height: sy(75) > sx(75) ? sy(75) : sx(75),
                      ),
                      Padding(
                          padding: EdgeInsets.symmetric(
                              horizontal: 0,
                              vertical: sy(15) > sx(15) ? sy(15) : sx(15))),
                      Container(
                        color: background,
                        child: Column(
                          children: [
                            Container(
                              width: sx(300) < sy(300) ? sx(300) : sy(300),
                              child: Text(
                                'OTP has been sent to you on your mobile phone.
                                Please enter it below.',
                                style: TextStyle(
                                    fontSize: sy(12) > sx(12) ? sy(12) : sx(12)),
                                textAlign: TextAlign.center,
                              ),
                            ),
                            Padding(
                                padding: EdgeInsets.symmetric(
                                    horizontal: 0,
                                    vertical: sy(10) > sx(10) ? sy(10) : sx(10))
                            ),
                            Form(
                              key: _formKey,
                              child: Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                        Container(
                          width: sy(32) > sx(32) ? sy(32) : sx(32),
                          height: sy(32) > sx(32) ? sy(32) : sx(32),
                          margin: EdgeInsets.symmetric(
                              horizontal: sx(6), vertical: 0),
                          child: TextFormField(
                            style: TextStyle(
                                fontSize:
                                sy(16) > sx(16) ? sy(16) : sx(16),
                                color: secondary
                            ),
                            keyboardType: TextInputType.number,
                            validator: (value) {
                              if (value.isNotEmpty) {
                                if(!_isNumeric(value)) {
                                  validationHandler();
                                }
                              }
                              return null;
                            },
                            textAlign: TextAlign.center,
                            cursorColor: secondary,
                            controller: textController1,
                            decoration: InputDecoration(
                              fillColor: otp,
                              filled: true,
                              enabledBorder: OutlineInputBorder(
                                borderRadius: BorderRadius.circular(10),
                                borderSide: BorderSide(
                                  color: otpBorder,
                                ),
                              ),
                              focusedBorder: OutlineInputBorder(
                                borderRadius: BorderRadius.circular(10),
                                borderSide: BorderSide(
                                    color: Color.fromRGBO(
                                        48, 79, 254, .2)),
                              ),
                            ),
                            onChanged: (value) {
                              if (  _formKey.currentState.validate()) {
                                nextField(
                                    value: value,
                                    focusNode: pin2FocusNode);
                              }
                            },
                          ),
                        ),
                        Container(
                          width: sy(32) > sx(32) ? sy(32) : sx(32),
                          height: sy(32) > sx(32) ? sy(32) : sx(32),
                          margin: EdgeInsets.symmetric(
                              horizontal: sx(6), vertical: 0),
                          child: TextFormField(
                            focusNode: pin2FocusNode,
                            style: TextStyle(
                              fontSize:
                              sy(16) > sx(16) ? sy(16) : sx(16),
                        color: secondary,
),
keyboardType: TextInputType.number,
validator: (value) {
  if (value.isNotEmpty) {
    if(!_isNumeric(value)) {
      validationHandler();
    }
  }
  return null;
},
textAlign: TextAlign.center,
cursorColor: secondary,
controller: textController2,
decoration: InputDecoration(
  fillColor: otp,
  filled: true,
  enabledBorder: OutlineInputBorder(
      borderRadius:
      BorderRadius.circular(10),
      borderSide: BorderSide(
        color: otpBorder,
      )),
  focusedBorder: OutlineInputBorder(
    borderRadius: BorderRadius.circular(10),
    borderSide: BorderSide(
        color: Color.fromRGBO(
            48, 79, 254, .2)),
  ),
),
    onChanged: (value) {
      if (  _formKey.currentState.validate()) {
        nextField(
            value: value,
            focusNode: pin3FocusNode);
      }
    },
  ),
),
Container(
  width: sy(32) > sx(32) ? sy(32) : sx(32),
  height: sy(32) > sx(32) ? sy(32) : sx(32),
  margin: EdgeInsets.symmetric(
      horizontal: sx(6), vertical: 0),
  child: TextFormField(
    focusNode: pin3FocusNode,
    style: TextStyle(
      color: secondary,
      fontSize:
      sy(16) > sx(16) ? sy(16) : sx(16),),
    keyboardType: TextInputType.number,
    validator: (value) {
      if (value.isNotEmpty) {
        if(!_isNumeric(value)) {
          validationHandler();
        }
      }
      return null;
    },
    textAlign: TextAlign.center,
    cursorColor: secondary,
    controller: textController3,
    decoration: InputDecoration(
      fillColor: otp,
      filled: true,
      enabledBorder: OutlineInputBorder(
          borderRadius:
          BorderRadius.circular(10),
          borderSide: BorderSide(
            color: otpBorder,
          )),
      focusedBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(
                48, 79, 254, .2)),
      ),
    ),
    onChanged: (value) {
      if (  _formKey.currentState.validate()) {
        nextField(
            value: value,
            focusNode: pin4FocusNode);
      }
    },
  ),
),
Container(
  width: sy(32) > sx(32) ? sy(32) : sx(32),
  height: sy(32) > sx(32) ? sy(32) : sx(32),
  margin: EdgeInsets.symmetric(
      horizontal: sx(6), vertical: 0),
  child: TextFormField(
    focusNode: pin4FocusNode,
    style: TextStyle(
      color: secondary,
      fontSize:
      sy(16) > sx(16) ? sy(16) : sx(16),),
    keyboardType: TextInputType.number,
    validator: (value) {
      if (value.isNotEmpty) {
        if(!_isNumeric(value)) {
          validationHandler();
        }
      }
      return null;
    },
    textAlign: TextAlign.center,
    cursorColor: secondary,
    controller: textController4,
    decoration: InputDecoration(
      fillColor: otp,
      filled: true,
      enabledBorder: OutlineInputBorder(
          borderRadius:
          BorderRadius.circular(10),
          borderSide: BorderSide(
            color: otpBorder,
          )),
      focusedBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
          color: Color.fromRGBO(
            48, 79, 254, .2)),
      ),
    ),
    onChanged: (value) {
      if (  _formKey.currentState.validate()) {
        nextField(
          value: value,
          focusNode: pin5FocusNode);
      }
    },
  ),
),
Container(
  width: sy(32) > sx(32) ? sy(32) : sx(32),
  height: sy(32) > sx(32) ? sy(32) : sx(32),
  margin: EdgeInsets.symmetric(
      horizontal: sx(6), vertical: 0),
  child: TextFormField(
    focusNode: pin5FocusNode,
    style: TextStyle(
      color: secondary,
      fontSize:
      sy(16) > sx(16) ? sy(16) : sx(16),),
    keyboardType: TextInputType.number,
    validator: (value) {
      if (value.isNotEmpty) {
        if(!_isNumeric(value)) {
```

```
      validationHandler();
    }
  }
  return null;
},
textAlign: TextAlign.center,
cursorColor: secondary,
controller: textController5,
decoration: InputDecoration(
  fillColor: otp,
  filled: true,
  enabledBorder: OutlineInputBorder(
      borderRadius:
      BorderRadius.circular(10),
      borderSide: BorderSide(
        color: otpBorder,
      )),
  focusedBorder: OutlineInputBorder(
    borderRadius: BorderRadius.circular(10),
    borderSide: BorderSide(
        color: Color.fromRGBO(
          48, 79, 254, .2)),
  ),
),
onChanged: (value) {
  if (  _formKey.currentState.validate()) {
    nextField(
        value: value,
        focusNode: pin6FocusNode);
  }
```

```
    },
  ),
),
Container(
  width: sy(32) > sx(32) ? sy(32) : sx(32),
  height: sy(32) > sx(32) ? sy(32) : sx(32),
  margin: EdgeInsets.symmetric(
      horizontal: sx(6), vertical: 0),
  child: TextFormField(
    focusNode: pin6FocusNode,
    style: TextStyle(
      color: secondary,
      fontSize:
      sy(16) > sx(16) ? sy(16) : sx(16),),
    keyboardType: TextInputType.number,
    validator: (value) {
      if (value.isNotEmpty) {
        if(!_isNumeric(value)) {
          validationHandler();
        }
      }
      return null;
    },
    textAlign: TextAlign.center,
    cursorColor: secondary,
    controller: textController6,
    decoration: InputDecoration(
      fillColor: otp,
      filled: true,
      enabledBorder: OutlineInputBorder(
                  borderRadius:
                  BorderRadius.circular(10),
                  borderSide: BorderSide(
                    color: otpBorder,
                  )),
              focusedBorder: OutlineInputBorder(
                borderRadius: BorderRadius.circular(10),
                borderSide: BorderSide(
                    color: Color.fromRGBO(
                        48, 79, 254, .2)),
              ),
            ),
            onChanged: (value) {
              if (  _formKey.currentState.validate()) {
              }
            },
          ),
        ),
      ],
    ),
  ),
  Padding(
      padding: EdgeInsets.symmetric(
          horizontal: 0,
          vertical: sy(8) > sx(8) ? sy(8) : sx(8))),
  FlatButton(
    padding: EdgeInsets.symmetric(
        vertical: 5, horizontal: 20),
    textColor: primary,
    color: background,
    child: Text(
      'Verify',
      style: TextStyle(
          fontSize: sy(12) > sx(12) ? sy(12) : sx(12)),
    ),
    onPressed: () => {
      OTP += textController1.text,
      OTP += textController2.text,
      OTP += textController3.text,
      OTP += textController4.text,
      OTP += textController5.text,
      OTP += textController6.text,
      pin2FocusNode.unfocus(),
      pin3FocusNode.unfocus(),
      pin4FocusNode.unfocus(),
      pin5FocusNode.unfocus(),
      pin6FocusNode.unfocus(),
      validateOtp(OTP),
    },
  ),
  Padding(
      padding: EdgeInsets.symmetric(
          horizontal: 0,
          vertical: sy(10) > sx(10) ? sy(10) : sx(10))),
  Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Container(
        child: Text(
          'Didn\'t receive code?  ',
          style: TextStyle(
              fontSize:
              sy(12) > sx(12) ? sy(12) : sx(12)),
          textAlign: TextAlign.center,
        ),
      ),
      Container(
        child: Text(
          '$minute:$secondsLeft$secondsRight',
          style: TextStyle(
              fontSize:
              sy(12) > sx(12) ? sy(12) : sx(12),
              color: primary),
          textAlign: TextAlign.center,
        ),
      ),
    ],
  ),
  resend == true
      ? FlatButton(
    padding: EdgeInsets.symmetric(
        vertical: 5, horizontal: 20),
    textColor: secondary,
    color: background,
    child: Text(
      'Resend Code',
      style: TextStyle(
          fontSize:
          sy(12) > sx(12) ? sy(12) : sx(12)),
    ),
                    onPressed: () => onPressHandler(),
                   )
                     : FlatButton(
                    padding: EdgeInsets.symmetric(
                        vertical: 5, horizontal: 20),
                    textColor: secondary,
                    color: background,
                    child: Text(
                      'Resend Code',
                      style: TextStyle(
                          fontSize:
                          sy(12) > sx(12) ? sy(12) : sx(12)),
                    ),
                   ),
                  ],
                ),
               )
              ],
            ),
          ),
         ],
       ),
      ),
    );
   });
  }
}
```

5. **dashboardUI.dart**

```
import 'package:bus_app/userprofile.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'busUpdatesStudent.dart';
import 'mapScreen.dart';
import 'routeSelection.dart';
import 'package:geolocator/geolocator.dart';
import 'package:permission_handler/permission_handler.dart';
import 'package:firebase_database/firebase_database.dart';
class DashBoardUI extends StatefulWidget {
  const DashBoardUI(
      {Key key, @required this.routeSelected, @required
this.roll_number})
      : super(key: key);
  final String roll_number;
  final bool routeSelected;
  @override
  _DashBoardState createState() =>
      _DashBoardState(routeSelected: routeSelected, roll_number:
roll_number);
}
class _DashBoardState extends State<DashBoardUI> {
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  bool opts = false;
  bool isload=false;
  String token;
  double latitude, longitude;
  _DashBoardState(
      {Key key, @required this.routeSelected, @required
this.roll_number});
  final String roll_number;
  final bool routeSelected;
  String userName = "";
  Future<String> loadUserName() async {
    SharedPreferences prefs = await SharedPreferences.getInstance();
    String user = await prefs.getString("userName");
    setState(() {
      userName = "Hi " + user;
    });
    return "Hai";
  }
  int radioValue = -1;
  void _handleRadioValueChanged(int value) {
    setState(() {
      radioValue = value;
    });
  }
  @override
  Widget build(BuildContext context) {
    return WillPopScope(
      onWillPop: () {
        showDialog(
            context: this.context,
            child: AlertDialog(
          content: Column(
            mainAxisSize: MainAxisSize.min,
            children: [
              Text("Do you want to exit the app?"),
              Row(
                mainAxisSize: MainAxisSize.min,
                mainAxisAlignment: MainAxisAlignment.end,
                crossAxisAlignment: CrossAxisAlignment.end,
                children: [
                  FlatButton(
                    child: Text("Yes"),
                    onPressed: () => SystemNavigator.pop(),
                  ),
                  FlatButton(
                    child: Text("No"),
                    onPressed: () => Navigator.pop(context, true),
                  ),
                ],
              )
            ],
          ),
        ));
    },
    child: RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
      return Scaffold(
        body: Container(
          height: screenHeight,
          width: screenWidth,
          color: background,
        padding: EdgeInsets.symmetric(
          vertical: sy(10),
          horizontal: sy(18),
        ),
        child: Column(
          mainAxisSize: MainAxisSize.min,
          children: [
            ListView(
              shrinkWrap: true,
              children: [
                Row(
                  mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: [
                    Container(
                        child: loadUserName() != null
                            ? Text(
                                userName,
                                style: TextStyle(
                                    color: primary,
                                    fontStyle: FontStyle.italic,
                                    fontSize:
                                        sy(14) > sx(14) ? sy(14) : sx(14)),
                                textAlign: TextAlign.center,
                              )
                            : Container()),
                    InkWell(
                      onTap: () {
                        Navigator.push(
                            context,
                            MaterialPageRoute(
                                builder: (context) => UserProfile()));
                      },
                      child: Container(
                        child: Icon(
                          Icons.supervised_user_circle_outlined,
                          color: primary,
                          size: 40,
                        ),
                      ),
                    ),
                  ],
                ),
                Padding(
                  padding:
                      EdgeInsets.symmetric(vertical: sy(20), horizontal: 0),
                ),
                routeSelected == true
                    ? InkWell(
                        child: Container(
                          child: MapScreen(),
                          height: sy(200),
                          decoration: BoxDecoration(
                            border: Border.all(
                              width: 2.0,
                              color: Colors.black,
                              style: BorderStyle.solid,
                            ),
                            borderRadius:
                                BorderRadius.all(Radius.circular(10)),
                          ),
                        ),
                        onLongPress: () {
                          return Navigator.push(
                            context,
                            MaterialPageRoute(
                                builder: (context) => MapScreen()),
                          );
                        },
                      )
                    : Container(
                        height: sy(200),
                        child: Center(
                          child: Container(
                            width: sx(250),
                            margin: EdgeInsets.only(top: sy(100)),
                            child: RaisedButton(
                              child: Text(
                                'Select Route',
                                style: TextStyle(
                                    color: background, fontSize: 16),
                              ),
                              color: primary,
                              shape: RoundedRectangleBorder(
                                  borderRadius:
                                      new BorderRadius.circular(30.0)),
                              onPressed: () {
                                Navigator.push(
                                    context,
                                    MaterialPageRoute(
                                        builder: (context) =>
```

```
                              RouteSelection()));
                        },
                      ),
                    ),
                  ),
                  decoration: BoxDecoration(
                      border: Border.all(
                        width: 2.0,
                        color: Colors.black,
                        style: BorderStyle.solid,
                      ),
                      borderRadius:
                          BorderRadius.all(Radius.circular(10)),
                      image: DecorationImage(
                        image:
                            ExactAssetImage('assets/dashboard.jpg'),
                        fit: BoxFit.cover,
                      )),
                ),
        Padding(
          padding:
              EdgeInsets.symmetric(vertical: sy(20), horizontal: 0),
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Container(
              padding: EdgeInsets.symmetric(
                  horizontal: sy(10), vertical: 0),
              width: sx(220),
                height: sy(125),
                child: InkWell(
                  onTap: () {
                    showDialog<void>(
                        context: context,
                        builder: (BuildContext context) {
                          int selectedRadio = 1;
                          return AlertDialog(
                            content: StatefulBuilder(
                              builder: (BuildContext context,
                                  StateSetter setState) {
                                return Column(
                                    mainAxisSize: MainAxisSize.min,
                                    children: [
                              Text(
                                  "Do you want reminders if the bus is
within 5 kms of your current location?"),
                              ListTile(
                                  title: Text("Enable"),
                                  leading: Radio<int>(
                                    value: 1,
                                    groupValue: selectedRadio,
                                    onChanged: (int value) {
                                      setState(() =>
                                          selectedRadio =
                                              value);
                                    },
                                  )),
                              ListTile(
                                  title: Text("Disable"),
                                  leading: Radio<int>(
                                    value: 0,
                                    groupValue: selectedRadio,
                                    onChanged: (int value) {
                                      setState(() =>
                                          selectedRadio =
                                              value);
                                    },
                                  )),
                              Center(
                                child: isload==false?RaisedButton(
                                  child: Text("Update"),
                                  onPressed: () async {
                                    try{
                                      setState(() =>
                                      isload =
                                      true);

                                      if (selectedRadio == 1 &&
                                          routeSelected ==
                                              true) {
                                        // get the route and bus number
                                        SharedPreferences
                                        prefs =
                                        await SharedPreferences
                                            .getInstance();
                                        String route = prefs
                                            .getString("route");
                                        String busNumber = prefs
                                            .getString("bus");
                                   final Geolocator
```

```
geolocator =
Geolocator()
  ..forceAndroidLocationManager;
var status =
await Permission
    .location
    .request();
if (status ==
    PermissionStatus
        .granted) {
  await geolocator
      .getCurrentPosition(
      desiredAccuracy:
      LocationAccuracy
          .best)
      .then((Position
  position) {
    setState(() {
      latitude =
          position
              .latitude;
      longitude =
          position
              .longitude;
    });
  }).catchError((e) {
  });
}
var db =
FirebaseDatabase
```

```
                                                  .instance
                                                  .reference();
                                              await prefs.setBool("externaluserId",
true);
                                              await
OneSignal.shared.setExternalUserId(roll_number);
                                              await
OneSignal.shared.setSubscription(true);
                                              await db.once().then(
                                                      (DataSnapshot
                                                  snap) {
                                                    db
                                                        .child(
                                                        "Bus Routes")
                                                        .child(route)
                                                        .child(busNumber)
                                                        .child(
                                                        "Users Subscribed")
                                                        .child(
                                                        roll_number)
                                                        .set({
                                                      "Latitude":
                                                      latitude,
                                                      "Longitude":
                                                      longitude
                                                    });
                                                  });
                                              Fluttertoast.showToast(msg:
"Reminder enabled successfully");
                                            } else if (selectedRadio ==
                                                1 &&
                                                routeSelected ==
      false) {
   showDialog(
     context:
     this.context,
     child: AlertDialog(
      content: Column(
       mainAxisSize:
       MainAxisSize
         .min,
       children: [
        Text(
          "Select a bus route and bus
number to set the reminder!"),
        FlatButton(
         child: Text(
           "Ok"),
         onPressed: () =>
           Navigator.pop(
             context,
             true),
        ),
       ],
      ),
     ));
  } else if (selectedRadio ==
    0 &&
    routeSelected ==
      true) {
   SharedPreferences
   prefs =
   await SharedPreferences
```

```
                                             .getInstance();
                                           String route = prefs
                                             .getString("route");
                                           String busNumber = prefs
                                             .getString("bus");
                                           var db =
                                           FirebaseDatabase
                                             .instance
                                             .reference();
                                           await
OneSignal.shared.setSubscription(false);
                                           await db.once().then(
                                               (DataSnapshot
                                             snap) {
                                              db
                                                 .child(
                                                 "Bus Routes")
                                                 .child(route)
                                                 .child(busNumber)
                                                 .child(
                                                 "Users Subscribed")
                                                 .child(
                                                 roll_number)
                                                 .remove();
                                              Fluttertoast.showToast(msg:
"Reminder disabled successfully");
                                             });
                                          } else if (selectedRadio ==
                                            0 &&
                                            routeSelected ==
                                              false) {
```

```
    showDialog(
        context:
        this.context,
        child: AlertDialog(
          content: Column(
            mainAxisSize:
            MainAxisSize
                .min,
            children: [
              Text(
                  "Select a bus route and bus
number to set the reminder!"),
              FlatButton(
                child: Text(
                    "Ok"),
                onPressed: () =>
                    Navigator.pop(
                        context,
                        true),
              ),
            ],
          ),
        ));
  }
  setState(() =>
  isload =
      false);
}
catch(e)
{
  setState(() =>
                                    isload =
                                    false);
                                    Fluttertoast.showToast(msg:
"Something went wrong");
                                   }
                                   Navigator.pop(
                                     context, true);
                                  },
                                  color: secondary,
):CircularProgressIndicator(backgroundColor: Colors.white,),
                                 )
                                ]);
                             },
                            ),
                           );
                          });
                       },
                       child: Card(
                        child: Center(
                          child: Icon(
                         Icons.notifications_on_outlined,
                         size: sx(60),
                         color: primary,
                        )),
                        color: background,
                        shape: RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(15)),
                        shadowColor: secondary,
                        elevation: 3,
                       ),
                      ),
                      decoration: BoxDecoration(
                        borderRadius:
                          BorderRadius.all(Radius.circular(10))),
                     ),
                     Container(
                      padding: EdgeInsets.symmetric(
                        horizontal: sy(10), vertical: 0),
                      width: sx(220),
                      height: sy(125),
                      child: InkWell(
                       onTap: () {
                        Navigator.push(
                          context,
                          MaterialPageRoute(
                            builder: (context) => BusUpdates(roll_number:
roll_number,)));
                       },
                       child: Card(
                        child: Center(
                          child: Icon(
                         Icons.directions_bus_rounded,
                         size: sx(60),
                         color: primary,
                        )),
                        color: background,
                        shape: RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(15)),
                        shadowColor: secondary,
                        elevation: 3,
                       ),
                      ),
                    ),
                  ],
                ),

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

6. **mapScreen.dart**

```
import 'dart:async';
import 'dart:typed_data';

import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:shared_preferences/shared_preferences.dart';
class MapScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Find My Bus',
      home: MapSample(),
    );
  }
}
class MapSample extends StatefulWidget {
  @override
  State<MapSample> createState() => MapSampleState();
}
```

```
class MapSampleState extends State<MapSample> {
  Completer<GoogleMapController> _controller = Completer();
  static double Lat = 11.127;
  static double Lng = 78.6569;
  static LatLng lastPosition = LatLng(Lat, Lng);
  static double zoomLevel=14;
  String route, busNumber;
  void getValues() async {
    SharedPreferences prefs = await SharedPreferences.getInstance();
    route = await prefs.getString("route");
    busNumber = await prefs.getString("bus");
  }
  Map<dynamic, dynamic> values;
  var db = FirebaseDatabase.instance.reference();
  Marker marker;
  Circle circle;
  CameraPosition _busPosition = CameraPosition(
      bearing: 192.8334901395799,
      target: LatLng(Lat ,Lng),
      zoom: zoomLevel,
  );
  Timer timer;
  @override
  void initState() {
    super.initState();
    Future<int> check_not_null=_initialTrackBus();
    check_not_null.then((value) {
      if (value == 1)
        timer = Timer.periodic(Duration(seconds: 2), (timer) =>
_findmyBus());
      else {
        Fluttertoast.showToast(msg: "Bus is not moving at this time");
      }
    });
  }
  Future<Uint8List> getMarker() async {
    ByteData byteData = await DefaultAssetBundle.of(context).load(
        "assets/busMarker.png");
    return byteData.buffer.asUint8List();
  }
```

```
@override
Widget build(BuildContext context) {
  return new Scaffold(
    body: GoogleMap(
      markers: Set.of((marker != null) ? [marker] : []),
      circles: Set.of((circle != null) ? [circle] : []),
      initialCameraPosition: _busPosition,
      onMapCreated: (GoogleMapController controller) {
        _controller.complete(controller);
      },
    ),
  );
}
void updateMarker (LatLng location, Uint8List imageData) {
  setState(() {
    marker = Marker(
        markerId: MarkerId("Bus"),
        position: location,
        draggable: false,
        zIndex: 2,
        flat: true,
        anchor: Offset(0.5, 0.5),
      icon: BitmapDescriptor.fromBytes(imageData)
    );
    circle = Circle(
        circleId: CircleId("car"),
        radius: 200,
        zIndex: 1,
        strokeColor: Colors.blue,
        strokeWidth: 2,
        center: location,
        fillColor: Colors.blue.withAlpha(50)
    );
  });
}
Future<void> _findmyBus() async {
  await getValues();

  final GoogleMapController controller = await _controller.future;

  await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      values = values[route];
      values = values[busNumber];
      double aPos = values['Latitude'];
      double bPos = values['Longitude'];
      setState(() {
        Lat = aPos;
        Lng = bPos;
      });
    });
    await controller.getZoomLevel().then((value) {
      zoomLevel=value;
    }
    );
    CameraPosition _busPosition1 = CameraPosition(
        target: LatLng(Lat ,Lng),
          zoom: zoomLevel,
    );
    Uint8List imageData = await getMarker();
    updateMarker(LatLng(Lat, Lng), imageData);
controller.animateCamera(CameraUpdate.newCameraPosition(_busPositi
on1));
  }
  Future<int> _initialTrackBus() async {
    await getValues();
    var db = FirebaseDatabase.instance.reference();
    int result=0;
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      values = values[route];
      values = values[busNumber];
      if(values['Latitude'].runtimeType==double)
      {
        result=1;
      }
    });
    return Future.value(result);
  }
}
```

7. **routeSelection.dart**

```
import 'package:firebase_messaging/firebase_messaging.dart';
import 'package:flutter/material.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'dashboardUI.dart';
class RouteSelection extends StatefulWidget {
  @override
  _RouteSelectionState createState() => _RouteSelectionState();
}
class _RouteSelectionState extends State<RouteSelection> {
  bool isloading=false;
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  List<String> temp = ['Select any route'], busStateList = ['Select Your
Bus'];
  String dropDownValue = 'Select any route';
  String busDropDown = 'Select Your Bus';
  Map<dynamic, dynamic> dbValues, busValues;
  String token;
  @override
  Widget build(BuildContext context) {
    Map<dynamic, dynamic> values;
    Iterable<dynamic> routes, buses;
    List<String> routeList = ['Select any route'];
    var db = FirebaseDatabase.instance.reference();
    db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      routes = values.keys;
      routes.forEach((element) {
        routeList.add(element);
      });
      setState(() {
        temp = routeList;
        dbValues = values;
      });
    });
```

```
SharedPreferences prefs;
return RelativeBuilder(
    builder: (context, screenHeight, screenWidth, sy, sx) {
  return Scaffold(
    appBar: AppBar(
      title: Text("Pick a Route"),
      backgroundColor: primary,
      centerTitle: true,
    ),
    body: Center(
      child: Column(
          crossAxisAlignment: CrossAxisAlignment.center,
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Container(
              width: 120,
              margin: EdgeInsets.only(bottom: 30, right: 20),
              child: Image.asset("assets/pecLogo.png"),
            ),
            DropdownButtonHideUnderline(
              child: Container(
                margin:
                    EdgeInsets.symmetric(vertical: sy(3), horizontal: 0),
                padding: EdgeInsets.only(left: 5),
                width: sx(230),
                decoration: ShapeDecoration(
                  shape: RoundedRectangleBorder(
                    side: BorderSide(
                        width: 1.0,
                        style: BorderStyle.solid,
                        color: secondary),
                    borderRadius: BorderRadius.all(Radius.circular(5.0)),
                  ),
                ),
                child: DropdownButton(
                  isExpanded: true,
                  value: dropDownValue,
                  icon: Icon(
                    Icons.arrow_drop_down,
                    color: primary,
                  ),
                  iconSize: 25,
                  elevation: 16,
                  style: TextStyle(color: primary),
                  onChanged: (String newValue) {
                    List<String> busList = ['Select Your Bus'];
                    values = dbValues[newValue];
                    buses = values.keys;
                    buses.forEach((element) {
                      busList.add(element);
                    });
                    setState(() {
                      dropDownValue = newValue;
                      busStateList = busList;
                      busValues = values;
                      busDropDown = 'Select Your Bus';
                    });
                  },
                  items: temp.map<DropdownMenuItem<String>>((String
  value) {
                    return DropdownMenuItem<String>(
                      value: value,
                      child: Text(value),
                    );
                  }).toList(),
                ),
              ),
            ),
            DropdownButtonHideUnderline(
              child: Container(
                margin:
                    EdgeInsets.symmetric(vertical: sy(3), horizontal: 0),
                padding: EdgeInsets.only(left: 5),
                width: sx(230),
                decoration: ShapeDecoration(
                  shape: RoundedRectangleBorder(
                    side: BorderSide(
                        width: 1.0,
                        style: BorderStyle.solid,
                        color: secondary),
                    borderRadius: BorderRadius.all(Radius.circular(5.0)),
                  ),
                ),
                child: DropdownButton(
                  isExpanded: true,
                  value: busDropDown,
                  icon: Icon(
                    Icons.arrow_drop_down,
                    color: primary,
                  ),
                  iconSize: 25,
                  elevation: 16,
                  style: TextStyle(color: primary),
                  underline: Container(
                    height: 2,
                    color: primary,
                  ),
                  onChanged: (String newValue) {
                    values = busValues[newValue];
                    setState(() {
                      busDropDown = newValue;
                    });
                  },
                  items: busStateList
                      .map<DropdownMenuItem<String>>((String value) {
                    return DropdownMenuItem<String>(
                      value: value,
                      child: Text(value),
                    );
                  }).toList(),
                ),
              ),
            ),
            isloading==false?RaisedButton(
              color: primary,
              child: Text(
                'OK',
                style: TextStyle(color: background),
              ),
              onPressed: () async {
                setState(() {
                  isloading=true;
                });
                try{
                  if (dropDownValue != 'Select any route' &&
                      busDropDown != 'Select Your Bus') {
                    prefs = await SharedPreferences.getInstance();
                    String rollNumber = await prefs.getString("userId");
                    if (await prefs.getString("route") != null) {
                      String routeName = await prefs.getString("route");
                      String busNumber = await prefs.getString("bus");
                      var db = FirebaseDatabase.instance.reference();
                      await OneSignal.shared.setSubscription(false);
                      await db.once().then((DataSnapshot snap) {
                        db
                            .child("Bus Routes")
                            .child(routeName)
                            .child(busNumber)
                            .child("Users Registered")
                            .child(rollNumber)
                            .remove();
                        db
                            .child("Bus Routes")
                            .child(routeName)
                            .child(busNumber)
                            .child("Users Subscribed")
                            .child(rollNumber)
                            .remove();
                      });
                    }
                    await prefs.setBool("routeSelected", true);
                    await prefs.setString("route", dropDownValue);
                    await prefs.setString("bus", busDropDown);
                    FirebaseMessaging _firebaseMessaging =
                    FirebaseMessaging();
                    _firebaseMessaging.getToken().then((value) {
                      setState(() {
                        token = value;
                      });
                    });
                    var db = FirebaseDatabase.instance.reference();
                    await db.once().then((DataSnapshot snap) {
                      db
                          .child("Bus Routes")
                          .child(dropDownValue)
                          .child(busDropDown)
                          .child("Users Registered")
```

```
                              .child(rollNumber)
                              .update({
                            "Fcm_Token": token,
                          });
                        });
                        Navigator.push(
                            context,
                            MaterialPageRoute(
                                builder: (context) => DashBoardUI(
                                  routeSelected: true,
                                  roll_number: rollNumber,
                                )));
                      } else {
                        Fluttertoast.showToast(
                            msg: "Select Route",
                            toastLength: Toast.LENGTH_SHORT,
                            gravity: ToastGravity.CENTER,
                            timeInSecForIosWeb: 1,
                            backgroundColor: Colors.red,
                            textColor: Colors.white,
                            fontSize: 16.0);
                      }
                      setState(() {
                        isloading=false;
                      });
                    }
                    catch(e)
                    {
                      setState(() {
                        isloading=false;
                      });
                      Fluttertoast.showToast(msg: "Something went wrong");
                    }
                  },
                ):CircularProgressIndicator(backgroundColor: Colors.blue,),
              ]),
        ),
      );
    });
  }
}
```

8. routeChange.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:keyboard_avoider/keyboard_avoider.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:fluttertoast/fluttertoast.dart';
class RouteChange extends StatefulWidget {
  @override
  _RouteChangeState createState() => _RouteChangeState();
}
class _RouteChangeState extends State<RouteChange> {
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  List<String> temp = ['Select any route'], busStateList = ['Select Your
Bus'];
  String dropDownValue = 'Select any route';
  String busDropDown = 'Select Your Bus';
  Map<dynamic, dynamic> dbValues, busValues;
  bool readOnly = true;
  bool revert_button=false,update_button=false;
  String routeChangeValue = "";
  Map<dynamic, dynamic> values;
  Iterable<dynamic> routes, buses;
  String number;
  String token;
  var db;
  void getDatabase() {
    List<String> routeList = ['Select any route'];
    db = FirebaseDatabase.instance.reference();
    db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      routes = values.keys;
      routes.forEach((element) {
        routeList.add(element);
      });
      setState(() {
        temp = routeList;
        dbValues = values;
      });
    });
  }
  @override
  void initState() {
    super.initState();
    getDatabase();
  }
  @override
  void dispose() {
    super.dispose();
  }
  @override
  Widget build(BuildContext context) {
    return RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
      return Scaffold(
        resizeToAvoidBottomInset: false,
        appBar: AppBar(
          title: Text("Change Route"),
          backgroundColor: primary,
          centerTitle: true,
        ),
        body: KeyboardAvoider(
          autoScroll: true,
          child: Center(
            child: Container(
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.center,
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
              Container(
              child: Image.asset(
                'assets/pecLogo.png',
                height: sy(100),
                fit: BoxFit.cover,
              )),
                  Divider(height: sy(10),color: Colors.white,),
                  DropdownButtonHideUnderline(
                    child: Container(
                      padding: EdgeInsets.only(left: 5),
                      margin:
                          EdgeInsets.symmetric(vertical: sy(3), horizontal: 0),
                      width: sx(230),
                      decoration: ShapeDecoration(
                        shape: RoundedRectangleBorder(
                          side: BorderSide(
                              width: 1.0,
                              style: BorderStyle.solid,
                              color: secondary),
                          borderRadius: BorderRadius.all(Radius.circular(5.0)),
                        ),
                      ),
                      child: DropdownButton(
                        value: dropDownValue,
                        icon: Icon(
                          Icons.arrow_drop_down,
                          color: primary,
                        ),
                        iconSize: 25,
                        elevation: 16,
                        style: TextStyle(color: primary),
                        isExpanded: true,
                        underline: Container(
                          height: 2,
                          color: primary,
                        ),
                        onChanged: (String newValue) {
                          List<String> busList = ['Select Your Bus'];
                          values = dbValues[newValue];
                          buses = values.keys;
                          buses.forEach((element) {
                            busList.add(element);
                          });
                          setState(() {
                            dropDownValue = newValue;
                            busStateList = busList;
                            busValues = values;
                            busDropDown = 'Select Your Bus';
                          });
                        },
                        items: temp.map<DropdownMenuItem<String>>((String
value) {
                          return DropdownMenuItem<String>(
                            value: value,
          child: Text(value),
         );
        }).toList(),
       ),
      ),
     ),
     DropdownButtonHideUnderline(
      child: Container(
       margin:
         EdgeInsets.symmetric(vertical: sy(3), horizontal: 0),
       padding: EdgeInsets.only(left: 5),
       width: sx(230),
       decoration: ShapeDecoration(
        shape: RoundedRectangleBorder(
         side: BorderSide(
           width: 1.0,
           style: BorderStyle.solid,
           color: secondary),
         borderRadius: BorderRadius.all(Radius.circular(5.0)),
        ),
       ),
       child: DropdownButton(
        value: busDropDown,
        isExpanded: true,
        icon: Icon(
         Icons.arrow_drop_down,
         color: primary,
        ),
        iconSize: 25,
        elevation: 16,
        style: TextStyle(color: primary),
        underline: Container(
         height: 2,
         color: primary,
        ),
        onChanged: (String newValue) {
         values = busValues[newValue];
         setState(() {
          busDropDown = newValue;
          routeChangeValue = values["Route Change"];
         });
        },
      items: busStateList
          .map<DropdownMenuItem<String>>((String value) {
        return DropdownMenuItem<String>(
          value: value,
          child: Text(value),
        );
      }).toList(),
    ),
  ),
),
Container(
  margin: EdgeInsets.symmetric(vertical: sy(7), horizontal: 0),
  width: sx(230),
  height: sy(35),
  child: TextFormField(
    readOnly: readOnly,
    initialValue: routeChangeValue,
    cursorColor: secondary,
    focusNode: FocusNode(canRequestFocus: false),
    onChanged: (input) => number = input,
    keyboardType: TextInputType.number,
    decoration: InputDecoration(
      labelText: "Bus Number",
      focusColor: secondary,
      labelStyle: TextStyle(
        color: primary,
        fontSize: sx(18),
      ),
      enabledBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(5),
        borderSide: BorderSide(
          color: secondary,
        ),
      ),
      focusedBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(5),
        borderSide:
            BorderSide(color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      errorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(5),
        borderSide:
            BorderSide(color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      focusedErrorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(5),
        borderSide:
            BorderSide(color: Color.fromRGBO(48, 79, 254, .2)),
      ),
    ),
  ),
),
RaisedButton(
  color: primary,
  child: Text(
    'Edit',
    style: TextStyle(color: background),
  ),
  onPressed: () async {
    setState(() {
      readOnly = false;
    });
    Fluttertoast.showToast(msg: "Now you can edit the route");
  },
),
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    readOnly == true
        ? Container(
            margin: EdgeInsets.symmetric(
                vertical: 0, horizontal: sx(5)),
            child: revert_button==false?RaisedButton(
              color: primary,
              child: Text(
                'Revert',
                style: TextStyle(color: background),
              ),
              onPressed: () async {
                setState(() {
                  revert_button=true;
                });
                if (dropDownValue != "Select any route" &&
                    busDropDown != "Select Your Bus") {
```

```
                              db
                                  .child("Bus Routes")
                                  .child(dropDownValue)
                                  .child(busDropDown)
                                  .update({"Route Change": "No"});
                              setState(() {
                                revert_button=false;
                              });
                              Fluttertoast.showToast(msg: "Route Reverted");
                            }
                            else{
                              setState(() {
                                revert_button=false;
                              });
                              Fluttertoast.showToast(msg: "Kindly select the
route");
                            }
                          },
                        ):CircularProgressIndicator(backgroundColor:
Colors.white,),
                      )
                    : Container(
                        margin: EdgeInsets.symmetric(
                            vertical: 0, horizontal: sx(5)),
                        child: RaisedButton(
                          color: primary,
                          child: Text(
                            'Revert',
                            style: TextStyle(color: background),
                          ),
                          onPressed: null,
                        ),
                      ),
                readOnly == false
                    ? Container(
                        margin: EdgeInsets.symmetric(
                            vertical: 0, horizontal: sx(5)),
                        child: update_button==false?RaisedButton(
                          color: primary,
                          child: Text(
                            'Update',
                            style: TextStyle(color: background),
                          ),
                          onPressed: () async {
                            setState(() {
                              update_button=true;
                            });
                            if (dropDownValue != "Select any route" &&
                                busDropDown != "Select Your Bus") {
                              db
                                  .child("Bus Routes")
                                  .child(dropDownValue)
                                  .child(busDropDown)
                                  .update({"Route Change": number});
                              setState(() {
                                update_button=false;
                              });
                              Fluttertoast.showToast(msg: "Route Updated");
                            }
                            else
                              {
                                setState(() {
                                  update_button=false;
                                });
                                Fluttertoast.showToast(msg: "Kindly select the
route");
                              }
                          },
                        ):CircularProgressIndicator(backgroundColor:
Colors.white,),
                      )
                    : Container(
                        margin: EdgeInsets.symmetric(
                            vertical: 0, horizontal: sx(5)),
                        child: RaisedButton(
                          color: primary,
                          child: Text(
                            'Update',
                            style: TextStyle(color: background),
                          ),
                          onPressed: null,
                        ),
                      )
                ],
                    )
                  ],
                ),
              ),
            ),
          ),
        );
      });
    }
  }
```

9. **busUpdatesStudent.dart**

```
    import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
class BusUpdates extends StatefulWidget {
  const BusUpdates(
      {Key key, @required this.roll_number})
      : super(key: key);
  final String roll_number;
  @override
  _BusUpdatesState createState() => _BusUpdatesState(roll_number:
roll_number);
}
class _BusUpdatesState extends State<BusUpdates> {
  _BusUpdatesState(
      {Key key, @required this.roll_number});
  final String roll_number;
  Map<dynamic, dynamic> values,temp;
  List<String> notifications = [];
  var db = FirebaseDatabase.instance.reference();
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  Future<List<String>> getNotifications() async {
    List<String> notificationlist = [];
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['LoginDetails'];
      values = values['Student'];
      values = values[roll_number];
      values = values['Your Notifications'];
      values.keys.forEach((element) {
        if(element!='Dummy')
        {
    notificationlist.add(element);
    notifications.add(values[element]);
        }
      });
    });
    return notificationlist;
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Bus Updates"),
        backgroundColor: primary,
        centerTitle: true,
      ),
      body: Center(
        child: FutureBuilder(
    future: getNotifications(),
    builder:
        (BuildContext context, AsyncSnapshot<List> snapshot) {
      switch (snapshot.connectionState) {
        case ConnectionState.none:
          return new Text('Waiting to start');
        case ConnectionState.waiting:
          return new Text('Loading...');
        default:
          if (snapshot.hasError) {
            return new Text('Error: ${snapshot.error}');
          } else {
              if(notifications.length == 0) {
                return Container(
                  child: Center(
                    child: Text("No new updates"),
                  ),
                );
              }
            return new ListView.builder(
                itemBuilder: (context, index) =>
                    Container(
                              margin: EdgeInsets.only(top: 20,left: 30,right: 30),
                              child: Row(
                                crossAxisAlignment: CrossAxisAlignment.center,
                                mainAxisAlignment: MainAxisAlignment.center,
                                children: [
                                  Flexible(
                                    child: ClipRRect(
                                      borderRadius: BorderRadius.only(
                                        topLeft: Radius.circular(32),
                                        bottomLeft: Radius.circular(32),
                                      ),
                                      child: ListTile(
                                        title: Text(
                                          notifications[index], style: TextStyle(color:
  background),
                                        ),
                                        tileColor: secondary,
                                      ),
                                    ),
                                  ),
                                  IconButton(icon: Icon(Icons.remove_circle_outlined),
  onPressed: ()async{
                                    await FirebaseDatabase.instance.reference()
                                        .child('LoginDetails').child('Student')
                                        .child(roll_number).child('Your Notifications')
                                        .child(snapshot.data[index])
                                        .remove();
                                    Fluttertoast.showToast(msg: "Notification removed
  successfully");
                                    setState(() {
                                      notifications.removeAt(index);
                                    });
                                  })
                                ],
                              ),
                            ),
                  itemCount: snapshot.data.length);
            }
          }
        },
      ),
    ),
     );
    }
  }
```
10.userProfile.dart
```
import 'package:bus_app/routeSelection.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'loginPage.dart';
class UserProfile extends StatefulWidget {
  @override
  _UserProfileState createState() => _UserProfileState();
}
class _UserProfileState extends State<UserProfile> {
  List<String> temp = ['Select any route'], busStateList = ['Select Your
Bus'];
  String dropDownValue = 'Select any route';
  String busDropDown = 'Select Your Bus';
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  final FirebaseAuth _auth = FirebaseAuth.instance;
  bool isload=false;
  String local_routename, local_route;
  Map<dynamic, dynamic> dbValues, busValues;
  bool routeSelected = false;
  List<String> routeList = ['Select any route'];
  Map<dynamic, dynamic> values;
  Iterable<dynamic> routes, buses;
  String bus, route;
  loadUserdata() async {
    SharedPreferences prefs = await SharedPreferences.getInstance();
    local_routename = await prefs.getString("route");
    local_route = await prefs.getString("bus");
    routeSelected = await prefs.getBool("routeSelected");
    if (routeSelected == null || routeSelected == false) {
      local_route = "No Route selected";
      local_routename = "No Route selected";
    }
    var db = FirebaseDatabase.instance.reference();
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      routes = values.keys;
      routes.forEach((element) {
        if (!routeList.contains(element)) routeList.add(element);
      });
      temp = routeList;
      dbValues = values;
    });
    return "Done Processing";
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("User Profile"),
        backgroundColor: primary,
      ),
      body: Center(
        child: FutureBuilder(
          future: loadUserdata(),
          builder: (BuildContext context, AsyncSnapshot snapshot) {
            switch (snapshot.connectionState) {
              case ConnectionState.none:
                return new Text('Waiting to start');
              case ConnectionState.waiting:
                return new Text('Loading...');
              default:
                if (snapshot.hasError) {
                  return new Text('Error: ${snapshot.error}');
                } else {
                  return Center(
                    child: Column(
                      children: [
                        Material(
                          elevation: 4.0,
                          shape: CircleBorder(),
                          clipBehavior: Clip.hardEdge,
                          color: Colors.transparent,
              child: Ink.image(
                image: AssetImage('assets/user.png'),
                fit: BoxFit.cover,
                width: 120.0,
                height: 120.0,
                child: InkWell(
                  onTap: () {},
                ),
              ),
            ),
            Container(
                margin: EdgeInsets.only(top: 20),
                child: Text(
                  "Your Route is: $local_routename",
                  style: TextStyle(
                      fontSize: 20, fontWeight: FontWeight.bold),
                )),
            Container(
                margin: EdgeInsets.only(top: 20),
                child: Text(
                  "Your Bus number is: $local_route",
                  style: TextStyle(
                      fontSize: 20, fontWeight: FontWeight.bold),
                )),
            Container(
              margin: EdgeInsets.only(top: 20),
              child: RaisedButton(
                color: primary,
                onPressed: () {
                  Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) => RouteSelection()));
                },
                child: Text("Edit" ,style: TextStyle(color:
  Colors.white),),
              ),
            ),
            Container(
                child: Center(
              child: isload==false? RaisedButton(
                color: primary,
                    child: Text("Logout" ,style: TextStyle(color:
Colors.white),),
                    onPressed: () async {
                      try{
                        setState(() {
                          isload=true;
                        });
                        await OneSignal.shared.setSubscription(false);
                        await OneSignal.shared.removeExternalUserId();
                        await _auth.signOut();
                        SharedPreferences prefs =
                        await SharedPreferences.getInstance();
                        await prefs.clear();
                        setState(() {
                          isload=false;
                        });
                        Navigator.push(
                            context,
                            MaterialPageRoute(
                                builder: (context) => LoginPage()));
                      }
                      catch(e)
                      {
                        Fluttertoast.showToast(msg: "Something went
wrong");
                      }
                    },
                  ):CircularProgressIndicator(backgroundColor:
Colors.white),
                ))
              ],
              mainAxisAlignment: MainAxisAlignment.center,
            ),
          );
        }
      }
    },
  ),
),
);
}
}
```

11. help.dart

```
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:relative_scale/relative_scale.dart';
class Help extends StatefulWidget {
  @override
  _HelpState createState() => _HelpState();
}
class _HelpState extends State<Help> {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
        home: Scaffold(
        backgroundColor: Color.fromARGB(255,255,255,255),
        appBar: AppBar(
            backgroundColor: Color(0xFF304FFE),
            title: Text('Help'),
          centerTitle: true,
        ),
      /*theme: ThemeData(
        primarySwatch: Colors.blue,
      ),*/
      body: MyHelpPage(title: 'Help'),
        ),
    );
  }
}
class MyHelpPage extends StatefulWidget {
  MyHelpPage({Key key, this.title}) : super(key: key);
  final String title;
  @override
  _MyHelpPageState createState() => _MyHelpPageState();
}
class _MyHelpPageState extends State {
  bool isloading=false;
final formKey = GlobalKey<FormState>();
String roll_no;
String _message;
Color primary = Color(0xFF304FFE);
Color secondary = Color(0xFF788Af4);
Color background = Color(0xfffefefe);
 @override
 Widget build(BuildContext context) {
  return RelativeBuilder(
     builder: (context, screenHeight, screenWidth, sy, sx)
  {
   return Form(
    key: formKey,
    child: Column(
     children: [
      Padding(
       padding: const EdgeInsets.only(right: 0, left: 0),),
      Container(
        padding: EdgeInsets.all(20.0),
        margin: const EdgeInsets.only(right: 0, left: 0),
        width: sx(550),
        child: TextFormField(
         decoration: new InputDecoration(
          labelText: 'Roll No.',
          labelStyle: TextStyle(
           color: primary,
          ),
          prefixIcon: const Icon(
           Icons.person,
           color: Color(0xFF788AF4),
          ),
          fillColor: background,
          border: new OutlineInputBorder(
           borderRadius: new BorderRadius.circular(25.0),
           borderSide: const BorderSide(
             color: Color(0xFF788AF4), width: 10.0
           ),
          ),
          focusedBorder: new OutlineInputBorder (
           borderRadius: new BorderRadius.circular(25.0),
           borderSide: const BorderSide(color: Color(0xFF788AF4),
           ),
          ),
         ),
         validator: (val) {
          if (val.length == 0) {
           return 'Roll No. field should not be empty';
          }
          else if (val.length != 12) {
            return 'Please enter valid Roll No.';
          }
          else {
            return null;
          }
        },
        onSaved: (val) => roll_no = val,
      )
    ),
    Container(
      padding: EdgeInsets.all(20.0),
      width: sx(550),
      margin: const EdgeInsets.only(right: 0, left: 0),
      child: TextFormField(
        maxLines: 4,
        decoration: InputDecoration(
          labelText: 'Message',
          labelStyle: TextStyle(
            color: primary,
          ),
          prefixIcon: const Icon(
            Icons.mail,
            color: Color(0xFF788AF4),
          ),
          fillColor: Colors.white,
          border: new OutlineInputBorder(
            borderRadius: new BorderRadius.circular(25.0),
            borderSide: new BorderSide(
            ),
          ),
          focusedBorder: new OutlineInputBorder (
            borderRadius: new BorderRadius.circular(25.0),
            borderSide: const BorderSide(color: Color(0xFF788AF4),
            ),
          ),
        ),
        validator: (val) {
          if (val.length == 0) {
            return 'Message field should not be empty';
          }
          else {
            return null;
          }
        },
        onSaved: (val) => _message = val,
      )
    ),
     isloading==false?  FractionallySizedBox(
      widthFactor: 0.5,
      child: RaisedButton(
        color: Color(0xFF304FFE),
        onPressed: () async {
          final form = formKey.currentState;
          if (form.validate()) {
            form.save();
            setState(() {
              isloading = true;
            });
            try {
              await FirebaseDatabase.instance.reference().child(
                  "Student_Issues").update(
                  {roll_no: {"Issue": _message}});
              Fluttertoast.showToast(
                  msg: "Issue reported successfully");
              setState(() {
                isloading=false;
              });
            }
          catch(e)
          {
             setState(() {
               isloading=false;
             });
          }
        }},
        shape: RoundedRectangleBorder(
          borderRadius: new BorderRadius.circular(30.0),
          side: BorderSide(color: secondary),
        ),
        child: Text(
          'Report',
          style: TextStyle(
                    color: Colors.white, fontSize: 15,
                  ),
                ),
              )
            ):CircularProgressIndicator(backgroundColor: Colors.blue,),
          ],
        ),
      );
    });
  }
}
```

12. admindashboard.dart

```
import 'package:bus_app/adminProfile.dart';
import 'package:bus_app/busFileUpload.dart';
import 'package:bus_app/manualBusUpload.dart';
import 'package:bus_app/manualStudentUpload.dart';
import 'package:bus_app/routeChange.dart';
import 'package:bus_app/student_issues.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'adminBusSearch.dart';
import 'adminStudentUpdate.dart';
import 'excel.dart';
class AdminDashboard extends StatefulWidget {
  @override
  _AdminDashboardState createState() => _AdminDashboardState();
}
class _AdminDashboardState extends State<AdminDashboard> {
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  String userName = "";
  bool isload=false;
  Future<String> loadUserName() async {
    SharedPreferences prefs = await SharedPreferences.getInstance();
    String user = await prefs.getString("userName");
    setState(() {
      userName = "Hi " + user;
    });
    return "Hai";
  }
  @override
  Widget build(BuildContext context) {
    return WillPopScope(
      onWillPop: () {
        showDialog(
            context: this.context,
            child: AlertDialog(
              content: Column(
                mainAxisSize: MainAxisSize.min,
                children: [
                  Text("Do you want to exit the app?"),
                  Row(
                    mainAxisSize: MainAxisSize.min,
                    mainAxisAlignment: MainAxisAlignment.end,
                    crossAxisAlignment: CrossAxisAlignment.end,
                    children: [
                      FlatButton(
                        child: Text("Yes"),
                        onPressed: () => SystemNavigator.pop(),
                      ),
                      FlatButton(
                        child: Text("No"),
                        onPressed: () => Navigator.pop(context, true),
                      ),
                    ],
                  )
                ],
              ),
            ));
      },
      child: RelativeBuilder(
          builder: (context, screenHeight, screenWidth, sy, sx) {
        return Scaffold(
          resizeToAvoidBottomPadding: false,
          body: Container(
            height: screenHeight,
            width: screenWidth,
            color: background,
            padding: EdgeInsets.symmetric(
              vertical: sy(10),
              horizontal: sy(18),
            ),
            child: Column(
              children: [
                ListView(
                  shrinkWrap: true,
                  children: [
                    Padding(
                      padding:
                          EdgeInsets.symmetric(vertical: sy(10), horizontal: 0),
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.spaceBetween,
                      children: [
                        Container(
                            child: loadUserName() != null
                                ? Text(
                                    userName,
                                    style: TextStyle(
                                        color: primary,
                                        fontWeight: FontWeight.bold,
                                        fontSize:
                                            sy(14) > sx(14) ? sy(14) : sx(14)),
                                    textAlign: TextAlign.center,
                                  )
                                : Container()),
                        isload==false?InkWell(
                          onTap: () async {
                            setState(() {
                              isload=true;
                            });
                            String mobile_num;
                            Map<dynamic, dynamic> advalues;
                            var db = FirebaseDatabase.instance.reference();
                            await db.once().then((DataSnapshot snap) {
                              advalues = snap.value;
                              advalues = advalues['LoginDetails'];
                              advalues = advalues['Admin Access'];
                              mobile_num = advalues['ADMINPROJECT'];
```

```
                              setState(() {
                                isload=false;
                              });
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) => Admin_Profile(
                                            mobile: mobile_num,
                                          )));
                            });
                          },
                          child: Container(
                            child: Icon(
                              Icons.supervised_user_circle_outlined,
                              color: primary,
                              size: 40,
                            ),
                          ),
                        ):CircularProgressIndicator(backgroundColor:
Colors.white,),
                      ],
                    ),
                    Padding(
                      padding:
                          EdgeInsets.symmetric(vertical: sy(20), horizontal: 0),
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                        Container(
                          padding: EdgeInsets.symmetric(
                              horizontal: sy(10), vertical: 0),
                          width: sx(220),
                          height: sy(100),
                          child: InkWell(
                            onTap: () {
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) =>
                                          AdminBusSelection()));
                            },
                            child: Card(
                              child: Center(
                                  child: Column(
                                mainAxisAlignment: MainAxisAlignment.center,
                                children: [
                                  Icon(
                                    Icons.location_searching_rounded,
                                    size: sx(50),
                                    color: primary,
                                  ),
                                  Text("Locate Bus")
                                ],
                              )),
                              color: background,
                              shape: RoundedRectangleBorder(
                                  borderRadius: BorderRadius.circular(15)),
                              shadowColor: secondary,
                              elevation: 3,
                            ),
                          ),
                          decoration: BoxDecoration(
                              borderRadius:
                                  BorderRadius.all(Radius.circular(10))),
                        ),
                        Container(
                          padding: EdgeInsets.symmetric(
                              horizontal: sy(10), vertical: 0),
                          width: sx(220),
                          height: sy(100),
                          child: InkWell(
                            onTap: () {
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) => RouteChange()));
                            },
                            child: Card(
                              child: Center(
                                  child: Column(
                                mainAxisAlignment: MainAxisAlignment.center,
                                children: [
                                  Icon(
                                    Icons.alt_route_outlined,
                                    size: sx(50),
                                    color: primary,
                                  ),
                                  Text("Route Change")
                                ],
                              )),
                              color: background,
                              shape: RoundedRectangleBorder(
                                  borderRadius: BorderRadius.circular(15)),
                              shadowColor: secondary,
                              elevation: 3,
                            ),
                          ),
                          decoration: BoxDecoration(
                              borderRadius:
                                  BorderRadius.all(Radius.circular(10))),
                        ),
                      ],
                    ),
                    Padding(
                      padding:
                          EdgeInsets.symmetric(vertical: sy(4), horizontal: 0),
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                        Container(
                          padding: EdgeInsets.symmetric(
                              horizontal: sy(10), vertical: 0),
                          width: sx(220),
                          height: sy(100),
                          child: InkWell(
                            onTap: () {
                              return showDialog<void>(
                                context: context,
                                builder: (BuildContext context) {
                                  return AlertDialog(
                                    title: Text('Select the form of Upload'),
                                    content: SingleChildScrollView(
                                      child: ListBody(
                                        children: <Widget>[
                          RaisedButton(
                            child: Text(
                              "Upload File",
                              style:
                                  TextStyle(color: background),
                            ),
                            color: primary,
                            onPressed: () {
                              Navigator.pop(context,true);
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) =>
                                          MyFiles()));
                            },
                          ),
                          RaisedButton(
                            child: Text(
                              "Manual Upload",
                              style:
                                  TextStyle(color: background),
                            ),
                            color: primary,
                            onPressed: () {
                              Navigator.pop(context,true);
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) =>
                                          ManualStudentUpload()));

                            },
                          )
                        ],
                      ),
                    ),
                    actions: <Widget>[
                      TextButton(
                        child: Text(
                          'Close',
                          style: TextStyle(color: secondary),
                        ),
                        onPressed: () {
                          Navigator.of(context).pop();
                        },
                      ),
                    ],
                  );
                },
              );
            },
            child: Card(
              child: Center(
                  child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Icon(
                    Icons.cloud_upload_outlined,
                    size: sx(50),
                    color: primary,
                  ),
                  Text("Upload Student"),
                  Text("Data"),
                ],
              )),
              color: background,
              shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(15)),
              shadowColor: secondary,
              elevation: 3,
            ),
          ),
          decoration: BoxDecoration(
              borderRadius:
                  BorderRadius.all(Radius.circular(10))),
        ),
        Container(
          padding: EdgeInsets.symmetric(
              horizontal: sy(10), vertical: 0),
          width: sx(220),
          height: sy(100),
          child: InkWell(
            onTap: () {
              return showDialog<void>(
                context: context,
                builder: (BuildContext context) {
                  return AlertDialog(
                    title: Text('Select the form of Upload'),
                    content: SingleChildScrollView(
                      child: ListBody(
                        children: <Widget>[
                          RaisedButton(
                            child: Text(
                              "Upload File",
                              style:
                                  TextStyle(color: background),
                            ),
                            color: primary,
                            onPressed: () {
                              Navigator.pop(context,true);
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) =>
                                          BusFiles()));
                            },
                          ),
                          RaisedButton(
                            child: Text(
                              "Manual Upload",
                              style:
                                  TextStyle(color: background),
                            ),
                            color: primary,
                            onPressed: () {
                              Navigator.pop(context,true);
                              Navigator.push(
                                  context,
                                  MaterialPageRoute(
                                      builder: (context) =>
                                          ManualBusUpload()));
                            },
                          )
                        ],
                      ),
                    ),
                    actions: <Widget>[
                      TextButton(
                        child: Text(
                          'Close',
                          style: TextStyle(color: secondary),
                        ),
                        onPressed: () {
                          Navigator.of(context).pop();
                        },
                      ),
                    ],
                  );
                },
              );
            },
            child: Card(
              child: Center(
                  child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Icon(
                    Icons.cloud_upload_outlined,
                    size: sx(50),
                    color: primary,
                  ),
                  Text("Upload Bus"),
                  Text("Details"),
                ],
              )),
              color: background,
              shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(15)),
              shadowColor: secondary,
              elevation: 3,
            ),
          ),
          decoration: BoxDecoration(
              borderRadius:
                  BorderRadius.all(Radius.circular(10))),
        ),
      ],
    ),
    Padding(
      padding:
          EdgeInsets.symmetric(vertical: sy(4), horizontal: 0),
    ),
    Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Container(
          padding: EdgeInsets.symmetric(
              horizontal: sy(10), vertical: 0),
          width: sx(220),
          height: sy(100),
          child: InkWell(
            onTap: () {
              Navigator.push(
                  context,
                  MaterialPageRoute(
                      builder: (context) => StudentUpdate()));
            },
            child: Card(
              child: Center(
                  child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Icon(
                    Icons.update,
                    size: sx(50),
                    color: primary,
                  ),
                  Text("Update Student"),
                  Text("Data")
                ],
              )),
              color: background,
              shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(15)),
              shadowColor: secondary,
              elevation: 3,
            ),
          ),
          decoration: BoxDecoration(
              borderRadius:
                  BorderRadius.all(Radius.circular(10))),
        ),
        Container(
          padding: EdgeInsets.symmetric(
              horizontal: sy(10), vertical: 0),
          width: sx(220),
          height: sy(100),
          child: InkWell(
            onTap: () {
              Navigator.push(
                  context,
                  MaterialPageRoute(
                      builder: (context) => Student_Issues()));
            },
            child: Card(
              child: Center(
                  child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Icon(
                    Icons.person_pin,
                    size: sx(50),
                    color: primary,
                  ),
                  Text("Student Request")
                ],
              )),
              color: background,
              shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(15)),
              shadowColor: secondary,
              elevation: 3,
            ),
          ),
          decoration: BoxDecoration(
              borderRadius:
                  BorderRadius.all(Radius.circular(10))),
        ),
      ],
    ),
  ],
),
          ],
        ),
      ),
    );
  }),
);
}
}
```

13. adminBusSearch.dart

```
import 'package:bus_app/adminMap.dart';
import 'package:flutter/material.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:relative_scale/relative_scale.dart';
class AdminBusSelection extends StatefulWidget {
  @override
  _AdminBusSelectionState createState() => _AdminBusSelectionState();
}
class _AdminBusSelectionState extends State<AdminBusSelection> {
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  List<String> temp = [], busStateList = [];
  Map<dynamic, dynamic> dbValues, busValues;
  bool isload=true;
  Map<dynamic, dynamic> values;
  Iterable<dynamic> routes, buses;
  void getDataBase() async {
    List<String> routeList = [];
    var db = FirebaseDatabase.instance.reference();
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      routes = values.keys;
      routes.forEach((element) {
        routeList.add(element);
      });
      setState(() {
        temp = routeList;
        dbValues = values;
      });
    });
    setState(() {
      isload=false;
    });
  }
  @override
  void initState() {
    super.initState();
    getDataBase();
  }
  @override
  Widget build(BuildContext context) {
    return RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
      return Scaffold(
        appBar: AppBar(
          centerTitle: true,
          title: Text(
            'Locate Bus',
            style: TextStyle(fontSize: sy(15) > sx(15) ? sy(15) : sx(15)),
          ),
          backgroundColor: primary,
        ),
        body: Container(
          height: screenHeight,
          width: screenWidth,
          color: background,
          padding: EdgeInsets.symmetric(
            vertical: sy(10),
            horizontal: sy(14),
          ),
          child: Column(
            children: [
              Container(
                child: Text(
                  'Select a Route',
                  style: TextStyle(fontSize: sy(13) > sx(13) ? sy(13) : sx(13)),
                ),
              ),
              Padding(
                  padding: EdgeInsets.symmetric(
                      horizontal: 0,
                      vertical: sy(10) > sx(10) ? sy(10) : sx(10))),
              isload==false?
              ListView.builder(
                shrinkWrap: true,
                primary: false,
                itemCount: temp.length,
                itemBuilder: (context, index) {
                  return ListTile(
                    title: Container(
                      padding: EdgeInsets.all(
                        sx(20),
                      ),
                      decoration: BoxDecoration(
                        borderRadius: BorderRadius.circular(5),
                        border: Border.all(
                          color: Color.fromRGBO(120, 138, 244, .3),
                        ),
                      ),
                      child: Text(temp[index]),
                    ),
                    onTap: () {
                      Navigator.push(
                          context,
                          MaterialPageRoute(
                              builder: (context) => BusSelection(
                                    route: temp[index],
                                  )));
                    },
                  );
                },
              ):Center(child: CircularProgressIndicator(backgroundColor:
Colors.white,),)
            ],
          ),
        ),
      );
    });
  }
}
class BusSelection extends StatefulWidget {
  const BusSelection({Key key, @required this.route}) : super(key: key);
  final String route;
  @override
  _BusSelectionState createState() => _BusSelectionState(route: route);
}
class _BusSelectionState extends State<BusSelection> {
  _BusSelectionState({Key key, @required this.route});
  final String route;
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  List<String> busStateList = [];
  Map<dynamic, dynamic> values;
  double latitude, longitude;
  Iterable<dynamic> buses;
  bool isload=true;
  void getDataBase() async {
    List<String> busList = [];
    var db = FirebaseDatabase.instance.reference();
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      values = values[route];
      buses = values.keys;
      buses.forEach((element) {
        busList.add(element);
      });
      setState(() {
        busStateList = busList;
        values = values;
      });
    });
    setState(() {
      isload=false;
    });
  }
  @override
  void initState() {
    super.initState();
    getDataBase();
  }
  @override
  Widget build(BuildContext context) {
```

```
return RelativeBuilder(
    builder: (context, screenHeight, screenWidth, sy, sx) {
  return Scaffold(
    appBar: AppBar(
      centerTitle: true,
      title: Text(
        'Locate Bus',
        style: TextStyle(fontSize: sy(15) > sx(15) ? sy(15) : sx(15)),
      ),
      backgroundColor: primary,
    ),
    body: Container(
      height: screenHeight,
      width: screenWidth,
      padding: EdgeInsets.symmetric(
        vertical: sy(10),
        horizontal: sy(14),
      ),
      child: Column(
        children: [
          Container(
            child: Text(
              'Select a Bus Number ( $route )',
              style: TextStyle(fontSize: sy(13) > sx(13) ? sy(13) : sx(13)),
            ),
          ),
          Padding(
              padding: EdgeInsets.symmetric(
                  horizontal: 0,
                  vertical: sy(10) > sx(10) ? sy(10) : sx(10))),
          isload==false?ListView.builder(
            shrinkWrap: true,
            primary: false,
            itemCount: busStateList.length,
            itemBuilder: (context, index) {
              return ListTile(
                title: Container(
                  padding: EdgeInsets.all(
                    sx(20),
                  ),
                  decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(5),
                    border: Border.all(
                      color: Color.fromRGBO(120, 138, 244, .3),
                    ),
                  ),
                  child: Text(busStateList[index]),
                ),
                onTap: () {
                  Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) => AdminMapView(
                              route: route,
                              busNumber: busStateList[index])));
                },
              );
            },
          ):CircularProgressIndicator(backgroundColor: Colors.white,)
        ],
      ),
    ),
  );
});
}
}
```

14.adminMap.dart

```
import 'dart:async';
import 'dart:typed_data';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
class AdminMapView extends StatefulWidget {
  const AdminMapView({Key key, @required this.route, @required
this.busNumber})
      : super(key: key);
  final String route;
  final String busNumber;
  @override
  State<AdminMapView> createState() =>
      _AdminMapViewState(route: route, busNumber: busNumber);
}
```

```
class _AdminMapViewState extends State<AdminMapView> {
  _AdminMapViewState({Key key, @required this.route, @required
this.busNumber});
  final String route;
  final String busNumber;
  Completer<GoogleMapController> _controller = Completer();
  static double Lat = 11.127;
  static double Lng = 78.6569;
  static double zoomLevel = 14;
  Map<dynamic, dynamic> values;
  var db = FirebaseDatabase.instance.reference();
  Marker marker;
  Circle circle;
  CameraPosition _busPosition = CameraPosition(
    bearing: 192.8334901395799,
    target: LatLng(Lat, Lng),
    zoom: zoomLevel,
  );
  Timer timer;
  @override
  void initState() {
    super.initState();
    Future<int> check_not_null=_initialTrackBus();
    check_not_null.then((value) {
      if (value == 1)
        timer = Timer.periodic(Duration(seconds: 2), (timer) =>
_findmyBus());
      else {
        Fluttertoast.showToast(msg: "Bus is not moving at this time");
      }
    });
  }
  @override
  void dispose() {
    super.dispose();
    timer.cancel();
  }
  Future<Uint8List> getMarker() async {
    ByteData byteData =
        await DefaultAssetBundle.of(context).load("assets/busMarker.png");
    return byteData.buffer.asUint8List();
  }
```

```
@override
Widget build(BuildContext context) {
  return new Scaffold(
    body: GoogleMap(
      markers: Set.of((marker != null) ? [marker] : []),
      circles: Set.of((circle != null) ? [circle] : []),
      initialCameraPosition: _busPosition,
      onMapCreated: (GoogleMapController controller) {
        _controller.complete(controller);
      },
    ),
  );
}
void updateMarker(LatLng location, Uint8List imageData) {
  setState(() {
    marker = Marker(
        markerId: MarkerId("Bus"),
        position: location,
        draggable: false,
        zIndex: 2,
        flat: true,
        anchor: Offset(0.5, 0.5),
        icon: BitmapDescriptor.fromBytes(imageData));
    circle = Circle(
        circleId: CircleId("car"),
        radius: 200,
        zIndex: 1,
        strokeColor: Colors.blue,
        strokeWidth: 2,
        center: location,
        fillColor: Colors.blue.withAlpha(50));
  });
}
Future<void> _findmyBus() async {
  final GoogleMapController controller = await _controller.future;
  var db = FirebaseDatabase.instance.reference();
  await db.once().then((DataSnapshot snap) {
    values = snap.value;
    values = values['Bus Routes'];
    values = values[route];
    values = values[busNumber];
```

```
      double aPos = values['Latitude'];
      double bPos = values['Longitude'];
      setState(() {
        Lat = aPos;
        Lng = bPos;
      });
    });
    await controller.getZoomLevel().then((value) {
      zoomLevel = value;
    });
    CameraPosition _busPosition1 = CameraPosition(
      target: LatLng(Lat, Lng),
      zoom: zoomLevel,
    );
    Uint8List imageData = await getMarker();
    updateMarker(LatLng(Lat, Lng), imageData);
controller.animateCamera(CameraUpdate.newCameraPosition(_busPositi
on1));
  }
  Future<int> _initialTrackBus() async {
    var db = FirebaseDatabase.instance.reference();
    int result=0;
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
      values = values[route];
      values = values[busNumber];
      if(values['Latitude'].runtimeType==double)
        {
          result=1;
        }
    });
    return Future.value(result);
  }
}
```

15.adminProfile.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:keyboard_avoider/keyboard_avoider.dart';
import 'package:relative_scale/relative_scale.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'loginPage.dart';
class Admin_Profile extends StatefulWidget {
  const Admin_Profile({Key key, @required this.mobile}) : super(key:
key);
  final String mobile;
  @override
  _Admin_ProfileState createState() => _Admin_ProfileState(mobile:
mobile);
}
class _Admin_ProfileState extends State<Admin_Profile> {
  _Admin_ProfileState({Key key, @required this.mobile});
  final String mobile;
  String newnumber;
  bool readOnly = true, editPressed = false;
  final GlobalKey<FormState> _formkey = GlobalKey<FormState>();
  bool isload=false,logout_load=false;
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  final FirebaseAuth _auth = FirebaseAuth.instance;
  @override
  Widget build(BuildContext context) {
    return RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
      return Scaffold(
        resizeToAvoidBottomPadding: false,
        appBar: AppBar(
          centerTitle: true,
          title: Text("Admin Details"),
          backgroundColor: primary,
        ),
        body: KeyboardAvoider(
          autoScroll: true,
          child: Container(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.center,
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Icon(
                  Icons.supervised_user_circle,
                  color: primary,
                  size: sx(90),
                ),
                Padding(
                  padding: EdgeInsets.only(top: sy(30)),
                ),
                Row(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    Flexible(
                        flex: 2,
                        child: Container(
                          child: Form(
                            key: _formkey,
                            child: TextFormField(
                              focusNode: FocusNode(canRequestFocus: false),
                              initialValue: mobile,
                              onSaved: (input) => newnumber = input,
                              validator: (input) {
                                if (input.length == 0 && input.length < 13) {
                                  return "Enter a valid mobile number";
                                } else if (input.length > 13) {
                                  return "Enter a valid mobile number";
                                }
                                if (input == mobile) {
                                  return "Old number and new number \n are same";
                                }
                              },
                              cursorColor: secondary,
                              readOnly: readOnly,
                              decoration: InputDecoration(
                                labelText: "Mobile Number",
                                focusColor: secondary,
                                labelStyle: TextStyle(
                                  color: primary,
                                ),
                                enabledBorder: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(10),
                                  borderSide: BorderSide(
                                    color: secondary,
                                  ),
                                ),
                                focusedBorder: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(10),
                                  borderSide: BorderSide(
                                      color: Color.fromRGBO(48, 79, 254, .2)),
                                ),
                                errorBorder: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(10),
                                  borderSide: BorderSide(
                                      color: Color.fromRGBO(48, 79, 254, .2)),
                                ),
                                focusedErrorBorder: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(10),
                                  borderSide: BorderSide(
                                      color: Color.fromRGBO(48, 79, 254, .2)),
                                ),
                              ),
                            ),
                          ),
                        )),
                    Flexible(
                        child: Container(
                      margin: EdgeInsets.only(
                        left: sx(15),
                      ),
                      decoration: BoxDecoration(
                        color: primary,
                        border: Border.all(color: secondary),
                        borderRadius: BorderRadius.circular(40),
                      ),
                      child: IconButton(
                          icon: Icon(
                            Icons.edit_outlined,
                            color: background,
                          ),
                          onPressed: () {
                            setState(() {
                              readOnly = false;
                              editPressed = true;
                            });
                          }),
                    ))
                  ],
                ),
                editPressed == true
                    ? Container(
                        margin: EdgeInsets.only(top: 20),
                        child: isload==false?RaisedButton(
                          color: primary,
                          onPressed: () async {
                            setState(() {
                              isload=true;
                            });
                            final formstate = _formkey.currentState;
                            if (formstate.validate()) {
                              formstate.save();
                              setState(() {
                                readOnly = true;
                              });
                              Fluttertoast.showToast(msg: "Please wait...");
                              await FirebaseDatabase.instance
                                  .reference()
                                  .child("LoginDetails")
                                  .child("Admin Access")
                                  .update({"ADMINPROJECT": newnumber});
                              setState(() {
                                isload=false;
                                editPressed=false;
                              });
                              Fluttertoast.showToast(
                                  msg: "Mobile number updated successfully");
                            }
                            else
                              {
                                setState(() {
                                  isload=false;
                                });
                              }
                          },
                          child: Text(
                            "Save",
                            style: TextStyle(color: background),
                          ),
                        ):CircularProgressIndicator(backgroundColor:
          Colors.white,))
                    : Container(),
                Container(
                    margin: EdgeInsets.only(top: 20),
                    child: Center(
                      child: logout_load==false?RaisedButton(
                        color: primary,
                        child: Text(
                          "Logout",
                          style: TextStyle(color: background),
                        ),
                        onPressed: () async {
                          setState(() {
                            logout_load=true;
                          });
                          await _auth.signOut();
                          SharedPreferences prefs =
                              await SharedPreferences.getInstance();
                          await prefs.clear();
                          setState(() {
                            logout_load=false;
                          });
                          Fluttertoast.showToast(msg: "Logging you Out");
                          Navigator.push(
                              context,
                              MaterialPageRoute(
                                  builder: (context) => LoginPage()));
                        },
                      ):CircularProgressIndicator(backgroundColor:
          Colors.white,),
                    ))
                ],
              ),
            ),
          ),
        );
      });
    }
          }
```

16.busFileUpload.dart

```
import 'dart:async';
import 'dart:io';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:path/path.dart' as path;
import 'package:path_provider/path_provider.dart';
import 'package:flutter_file_manager/flutter_file_manager.dart';
import 'package:permission_handler/permission_handler.dart';
import 'package:excel/excel.dart';
import 'package:relative_scale/relative_scale.dart';
class BusFiles extends StatefulWidget {
  @override
  _BusFilesState createState() => _BusFilesState();
}
class _BusFilesState extends State<BusFiles> {
  var missingValue, validData;
  var actionNeeded = [];
  // Roll numbers with missing or invalid data are added to it
  String busNumber;
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  Future<void> readXlsx(path) async {
    var file = path;
    var bytes = File(file).readAsBytesSync();
    var excel = Excel.decodeBytes(bytes); // read the excel file
    Map<dynamic, dynamic> values;
    var db = FirebaseDatabase.instance.reference();
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['Bus Routes'];
    });
    for (var table in excel.tables.keys) {
      var title =
          excel.tables[table].rows[0]; // access the header of excel sheet
      var hasRoute = title.contains("Route"); // check for required files
      var hasBus = title.contains("Bus Number");
      if (hasRoute && hasBus) {
        Fluttertoast.showToast(msg: "Uploading Data");
        // if all present
        var busIndex = title.indexOf("Bus Number");
        var routeIndex = title.indexOf("Route");
        var data = excel.tables[table].rows
            .sublist(1); // accessing the content of excel sheet
        for (var row in data) {
```

```
  // for each row in excel sheet except header
  missingValue =
      false; // initialize as no missing values present in the row
  if (row.contains(null)) {
    // check for missing values
    missingValue = true;
    actionNeeded.add(row[busIndex]);
  }
  if (!missingValue) {
    // if no missing data and for valid data, add to db
    busNumber = row[busIndex]
        .toString()
        .trim()
        .substring(0, row[busIndex].toString().length - 2);
    await db
        .child("Bus Routes")
        .child(row[routeIndex].toString().trim())
        .child(busNumber)
        .set({
      "Bus Status": "Not Moving",
      "Latitude": "xxx",
      "Longitude": "xxx",
      "Route Change": "No"
    });
  }
}
Fluttertoast.showToast(msg: "Data Uploaded");
if (actionNeeded.isNotEmpty == true) {
  return showDialog<void>(
    context: context,
    builder: (BuildContext context) {
      return AlertDialog(
        title: Text('Error : Failed to Upload'),
        content: SingleChildScrollView(
          child: Container(
            height: 200,
            width: 200,
            child: ListView.builder(
              shrinkWrap: true,
              primary: false,
              itemCount: actionNeeded.length,
              itemBuilder: (context, index) {
                    return ListTile(
                      title: Container(
                        padding: EdgeInsets.all(
                          20,
                        ),
                        decoration: BoxDecoration(
                          borderRadius: BorderRadius.circular(5),
                          border: Border.all(
                            color: Color.fromRGBO(120, 138, 244, .3),
                          ),
                        ),
                        child: Text(actionNeeded[index]),
                      ),
                    );
                  },
                ),
              ),
            ),
            actions: <Widget>[
              TextButton(
                child: Text(
                  'Ok',
                  style: TextStyle(color: secondary),
                ),
                onPressed: () {
                  actionNeeded = [];
                  Navigator.of(context).pop();
                },
              ),
            ],
          );
        },
      );
    }
  } else {
    return showDialog<void>(
        context: context,
        builder: (BuildContext context) {
          return AlertDialog(
            title: Text("Error"),
            content: Text(
                "File to be uploaded doesn\'t contain required fields" +
                " [ Route and Bus Number]"),
            actions: [
              TextButton(
                child: Text(
                  'Ok',
                  style: TextStyle(color: secondary),
                ),
                onPressed: () {
                  Navigator.of(context).pop();
                },
              ),
            ],
          );
        });
  }
 }
}
@override
Widget build(BuildContext context) {
  return RelativeBuilder(
      builder: (context, screenHeight, screenWidth, sy, sx) {
    return Scaffold(
      appBar: AppBar(
        centerTitle: true,
        title: Text(
          'Upload Bus Information',
          style: TextStyle(fontSize: sy(15) > sx(15) ? sy(15) : sx(15)),
        ),
        backgroundColor: primary,
      ),
      body: SingleChildScrollView(
        child: Container(
          color: background,
          padding: EdgeInsets.symmetric(
            vertical: sy(10),
            horizontal: sy(14),
          ),
          child: Column(
            children: [
              Container(
                child: Text(
                  'Select the file to upload',
    style: TextStyle(fontSize: sy(13) > sx(13) ? sy(13) : sx(13)),
  ),
),
Padding(
    padding: EdgeInsets.symmetric(
        horizontal: 0,
        vertical: sy(10) > sx(10) ? sy(10) : sx(10))),
FutureBuilder(
    future: _getSpecificFileTypes(),
    builder: (BuildContext context, AsyncSnapshot snapshot) {
      switch (snapshot.connectionState) {
        case ConnectionState.none:
          return new Text('Waiting to start');
        case ConnectionState.waiting:
          return new Text('Loading...');
        default:
          if (snapshot.hasError) {
            return new Text('Error: ${snapshot.error}');
          } else {
            return ListView.builder(
              shrinkWrap: true,
              primary: false,
              itemCount: snapshot.data.length,
              itemBuilder: (context, index) {
                return ListTile(
                  title: Container(
                    padding: EdgeInsets.all(
                      sx(20),
                    ),
                    decoration: BoxDecoration(
                      borderRadius: BorderRadius.circular(5),
                      border: Border.all(
                        color: Color.fromRGBO(120, 138, 244, .3),
                      ),
                    ),
                    child: Text(
                        path.basename(snapshot.data[index].path)),
                  ),
                  onTap: () {
                    Fluttertoast.showToast(
                        msg: "Please wait while verifying this file");
                    readXlsx(snapshot.data[index].path);
```

```
                              }
                            );
                          },
                        );
                      }
                    }
                  }),
                ],
              ),
            ),
          ),
        );
      });
    }
    // get all files that match these extensions
    Future _getSpecificFileTypes() async {
      var status = await Permission.storage.request();
      var root = await getExternalStorageDirectory();
      var files = await FileManager(root: root)
          .filesTree(extensions: ["xlsx", "xlsm", "xlsb", "xltx", "xltm"]);
      return files;
    }
  }
```

17.busUpdatesStudent.dart

```
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
class BusUpdates extends StatefulWidget {
  const BusUpdates(
      {Key key, @required this.roll_number})
      : super(key: key);
  final String roll_number;
  @override
  _BusUpdatesState createState() => _BusUpdatesState(roll_number:
roll_number);
}
class _BusUpdatesState extends State<BusUpdates> {
  _BusUpdatesState(
      {Key key, @required this.roll_number});
  final String roll_number;
  Map<dynamic, dynamic> values,temp;
  List<String> notifications = [];
```

```
var db = FirebaseDatabase.instance.reference();
Color primary = Color(0xFF304FFE);
Color secondary = Color(0xFF788Af4);
Color background = Color(0xfffefefe);
Future<List<String>> getNotifications() async {
  List<String> notificationlist = [];
  await db.once().then((DataSnapshot snap) {
    values = snap.value;
    values = values['LoginDetails'];
    values = values['Student'];
    values = values[roll_number];
    values = values['Your Notifications'];
    values.keys.forEach((element) {
      if(element!='Dummy')
      {
        notificationlist.add(element);
        notifications.add(values[element]);
      }
    });
  });
  return notificationlist;
}
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text("Bus Updates"),
      backgroundColor: primary,
      centerTitle: true,
    ),
    body: Center(
      child: FutureBuilder(
        future: getNotifications(),
        builder:
            (BuildContext context, AsyncSnapshot<List> snapshot) {
          switch (snapshot.connectionState) {
            case ConnectionState.none:
              return new Text('Waiting to start');
            case ConnectionState.waiting:
              return new Text('Loading...');
            default:
              if (snapshot.hasError) {
                return new Text('Error: ${snapshot.error}');
              } else {
                 if(notifications.length == 0) {
                   return Container(
                     child: Center(
                       child: Text("No new updates"),
                     ),
                   );
                 }
                return new ListView.builder(
                   itemBuilder: (context, index) =>
                       Container(
                         margin: EdgeInsets.only(top: 20,left: 30,right: 30),
                         child: Row(
                           crossAxisAlignment: CrossAxisAlignment.center,
                           mainAxisAlignment: MainAxisAlignment.center,
                           children: [
                             Flexible(
                               child: ClipRRect(
                                 borderRadius: BorderRadius.only(
                                   topLeft: Radius.circular(32),
                                   bottomLeft: Radius.circular(32),
                                 ),
                                 child: ListTile(
                                   title: Text(
                                     notifications[index], style: TextStyle(color:
background),
                                   ),
                                   tileColor: secondary,
                                 ),
                               ),
                             ),
                             IconButton(icon:
Icon(Icons.remove_circle_outlined), onPressed: ()async{
                               await FirebaseDatabase.instance.reference()
                                   .child('LoginDetails').child('Student')
                                   .child(roll_number).child('Your Notifications')
                                   .child(snapshot.data[index])
                                   .remove();
                               Fluttertoast.showToast(msg: "Notification removed
successfully");
                               setState(() {
                                     notifications.removeAt(index);
                                    });
                                  })
                                ],
                              ),
                            ),
                      itemCount: snapshot.data.length);
                  }
              }
            },
          ),
        ),
      );
    }
  }
```

18.excel.dart

```
import 'dart:async';
import 'dart:io';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:path/path.dart' as path;
import 'package:path_provider/path_provider.dart';
import 'package:flutter_file_manager/flutter_file_manager.dart';
import 'package:permission_handler/permission_handler.dart';
import 'package:excel/excel.dart';
import 'package:relative_scale/relative_scale.dart';
class MyFiles extends StatefulWidget {
  @override
  _MyFilesState createState() => _MyFilesState();
}
class _MyFilesState extends State<MyFiles> {
  var missingValue, validData;
  var actionNeeded = [];
  // Roll numbers with missing or invalid data are added to it
  var phone, phoneLen;
  var countryCode = "+91";
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  Future<void> readXlsx(path) async {
    var file = path;
    var bytes = File(file).readAsBytesSync();
    var excel = Excel.decodeBytes(bytes); // read the excel file
    Map<dynamic, dynamic> values;
    var db = FirebaseDatabase.instance.reference();
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['LoginDetails'];
      values = values['Student'];
    });
    for (var table in excel.tables.keys) {
      Fluttertoast.showToast(msg: "Uploading Data");
      var title =
          excel.tables[table].rows[0]; // access the header of excel sheet
      var hasName = title.contains("Name"); // check for required files
      var hasRollNo = title.contains("Roll Number");
      var hasPhone = title.contains("Phone Number");
      if (hasName && hasRollNo && hasPhone) {
        // if all present
        var phoneIndex = title.indexOf("Phone Number");
        var rollNoIndex = title.indexOf("Roll Number");
        var nameIndex = title.indexOf("Name");
        var data = excel.tables[table].rows
            .sublist(1); // accessing the content of excel sheet
        for (var row in data) {
          // for each row in excel sheet except header
          missingValue =
              false; // initialize as no missing values present in the row
          validData =
              true; // initialized as valid data - phone numbers are of length 10
& Roll no = 12
          if (row.contains(null)) {
            // check for missing values
            missingValue = true;
            actionNeeded.add(row[rollNoIndex]);
          } else {
            // if no missing values present
            phoneLen = row[phoneIndex].toString().length - 2;

            phone = row[phoneIndex].toString().substring(0, phoneLen);
            if (phone.length != 10 ||
                row[rollNoIndex].toString().length != 12) {
              // check length of phone number && Roll number
              validData = false; // data to be uploaded only if true
              actionNeeded.add(row[rollNoIndex]);
            } else {
              phone = countryCode + phone;
            }
          }
          if (!missingValue && validData) {
            // if no missing data and for valid data, add to db
            await db
                .child("LoginDetails/Student")
                .child(row[rollNoIndex])
                .set({"Name": row[nameIndex], "Mobile": phone});
            await db
                .child("LoginDetails/Student")
                .child(row[rollNoIndex])
                .child("Your Notifications")
                .set({"Dummy": "notification"});
          }
        }
        Fluttertoast.showToast(msg: "Data Uploaded");
        if (actionNeeded.isNotEmpty == true) {
          return showDialog<void>(
            context: context,
            builder: (BuildContext context) {
              return AlertDialog(
                title: Text('Error : Failed to Upload'),
                content: SingleChildScrollView(
                  child: Container(
                    height:200,
                    width: 200,
                    child: ListView.builder(
                      shrinkWrap: true,
                      primary: false,
                      itemCount: actionNeeded.length,
                      itemBuilder: (context, index) {
                        return ListTile(
                          title: Container(
                            padding: EdgeInsets.all(
                              20,
                            ),
                            decoration: BoxDecoration(
                              borderRadius: BorderRadius.circular(5),
                              border: Border.all(
                                color: Color.fromRGBO(120, 138, 244, .3),
                              ),
                            ),
                            child: Text(actionNeeded[index]),
                          ),
                        );
                      },
                    ),
                  ),
                ),
                actions: <Widget>[
                  TextButton(
                    child: Text(
                      'Ok',
                      style: TextStyle(color: secondary),
                    ),
                    onPressed: () {
                      actionNeeded = [];
                      Navigator.of(context).pop();
                    },
                  ),
                ],
              );
            },
          );
        }
       } else {
        return showDialog<void>(
            context: context,
            builder: (BuildContext context) {
              return AlertDialog(
                title: Text("Error"),
                content: Text(
                    "File to be uploaded doesn\'t contain required fields" +
                        " [ Roll Number, Phone Number, Name ]"),
                actions: [
                  TextButton(
                    child: Text(
                      'Ok',
                      style: TextStyle(color: secondary),
                    ),
                    onPressed: () {
                      Navigator.of(context).pop();
                    },
                  ),
                ],
              );
            });
       }
     }
   }
   @override
   Widget build(BuildContext context) {
     return RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
       return Scaffold(
        appBar: AppBar(
          centerTitle: true,
          title: Text(
            'Upload Student Information',
            style: TextStyle(fontSize: sy(15) > sx(15) ? sy(15) : sx(15)),
          ),
          backgroundColor: primary,
        ),
        body: SingleChildScrollView(
          child: Container(
            color: background,
            padding: EdgeInsets.symmetric(
              vertical: sy(10),
              horizontal: sy(14),
            ),
            child: Column(
              children: [
                Container(
                  child: Text(
                    'Select the file to upload',
                    style: TextStyle(fontSize: sy(13) > sx(13) ? sy(13) : sx(13)),
                  ),
                ),
                Padding(
                    padding: EdgeInsets.symmetric(
          horizontal: 0,
          vertical: sy(10) > sx(10) ? sy(10) : sx(10))),
   FutureBuilder(
       future: _getSpecificFileTypes(),
       builder: (BuildContext context, AsyncSnapshot snapshot) {
        switch (snapshot.connectionState) {
          case ConnectionState.none:
            return new Text('Waiting to start');
          case ConnectionState.waiting:
            return new Text('Loading...');
          default:
            if (snapshot.hasError) {
              return new Text('Error: ${snapshot.error}');
            } else {
              return ListView.builder(
                shrinkWrap: true,
                primary: false,
                itemCount: snapshot.data.length,
                itemBuilder: (context, index) {
                  return ListTile(
                    title: Container(
                      padding: EdgeInsets.all(
                        sx(20),
                      ),
                      decoration: BoxDecoration(
                        borderRadius: BorderRadius.circular(5),
                        border: Border.all(
                          color: Color.fromRGBO(120, 138, 244, .3),
                        ),
                      ),
                      child: Text(
                          path.basename(snapshot.data[index].path)),
                    ),
                    onTap: () {
                      Fluttertoast.showToast(msg: "Please wait while
                      verifying this file");
                      readXlsx(snapshot.data[index].path);
                    }
                  );
                },
              );
            }
```

```
                    }
                  }),
              ],
            ),
          ),
        ),
      );
    });
  }
  // get all files that match these extensions
  Future _getSpecificFileTypes() async {
    var status = await Permission.storage.request();
    var root = await getExternalStorageDirectory();
    var files = await FileManager(root: root)
        .filesTree(extensions: ["xlsx", "xlsm", "xlsb", "xltx", "xltm"]);
    return files;
  }
}
```

19.manualBusUpload.dart

```
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:relative_scale/relative_scale.dart';
class ManualBusUpload extends StatefulWidget {
  @override
  _ManualBusUploadState createState() => _ManualBusUploadState();
}
class _ManualBusUploadState extends State<ManualBusUpload> {
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  bool isload=false;
  String route, busNumber;
  final GlobalKey<FormState> _formkey = GlobalKey<FormState>();
  var db = FirebaseDatabase.instance.reference();
  Map<dynamic, dynamic> values;
  @override
  Widget build(BuildContext context) {
    return RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
      return Scaffold(
appBar: AppBar(
  title: Text("Upload Bus Information"),
  centerTitle: true,
  backgroundColor: primary,
),
body: Center(
  child: SingleChildScrollView(
    child: Form(
      key: _formkey,
      child: Container(
        height: sy(300),
        width: sx(400),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Container(
                margin: EdgeInsets.only(bottom: 15),
                child: Image.asset(
                  'assets/pecLogo.png',
                  height: sy(100),
                  fit: BoxFit.cover,
                )),
            Flexible(
              flex: 3,
              child: TextFormField(
                keyboardType: TextInputType.name,
                onChanged: (input) => route = input,
                validator: (input) {
                  if (input.length == 0) {
                    return "Route cannot be empty";
                  }
                },
                cursorColor: secondary,
                decoration: InputDecoration(
                  labelText: "Route",
                  focusColor: secondary,
                  labelStyle: TextStyle(
                    color: primary,
                  ),
                  enabledBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                      color: secondary,
                    ),
                  ),
                  focusedBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                        color: Color.fromRGBO(48, 79, 254, .2)),
                  ),
                  errorBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                        color: Color.fromRGBO(48, 79, 254, .2)),
                  ),
                  focusedErrorBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                        color: Color.fromRGBO(48, 79, 254, .2)),
                  ),
                ),
              ),
            ),
            SizedBox(
              height: 15,
            ),
            Flexible(
              flex: 3,
              child: TextFormField(
                keyboardType: TextInputType.number,
                onChanged: (input) => busNumber = input,
                validator: (input) {
                  if (input.length == 0 && input.length < 10) {
                    return "Bus Number cannot be empty";
                  }
                },
                cursorColor: secondary,
                decoration: InputDecoration(
                  labelText: "Bus Number",
                  focusColor: secondary,
                  labelStyle: TextStyle(
                    color: primary,
                  ),
                  enabledBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                      color: secondary,
                    ),
                  ),
                  focusedBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                        color: Color.fromRGBO(48, 79, 254, .2)),
                  ),
                  errorBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                        color: Color.fromRGBO(48, 79, 254, .2)),
                  ),
                  focusedErrorBorder: OutlineInputBorder(
                    borderRadius: BorderRadius.circular(10),
                    borderSide: BorderSide(
                        color: Color.fromRGBO(48, 79, 254, .2)),
                  ),
                ),
              ),
            ),
            SizedBox(
              height: 15,
            ),
            isload==false?RaisedButton(
              child: Text(
                "Upload",
                style: TextStyle(color: background),
              ),
              color: primary,
              onPressed: () async {
                setState(() {
                  isload = true;
                });
                await db.once().then((DataSnapshot snap) {
                  values = snap.value;
                  values = values['Bus Routes'];
                });
                if (_formkey.currentState.validate()) {
                  Fluttertoast.showToast(msg: "Uploading data");
```

```
                          await db
                            .child("Bus Routes")
                            .child(route.trim())
                            .child(busNumber.trim())
                            .set({
                           "Bus Status": "Not Moving",
                           "Latitude": "xxx",
                           "Longitude": "xxx",
                           "Route Change": "No"
                          });
                          setState(() {
                           isload=false;
                          });
                          Fluttertoast.showToast(
                            msg: "Data uploaded successfully");
                         }
                         else{
                          setState(() {
                           isload = false;
                          });
                         }
                        },
                      ):CircularProgressIndicator(backgroundColor: Colors.white,)
                     ],
                    ),
                   ),
                  ),
                 ),
                ),
               );
              });
             }
            }
```

20.manualStudentUpload.dart

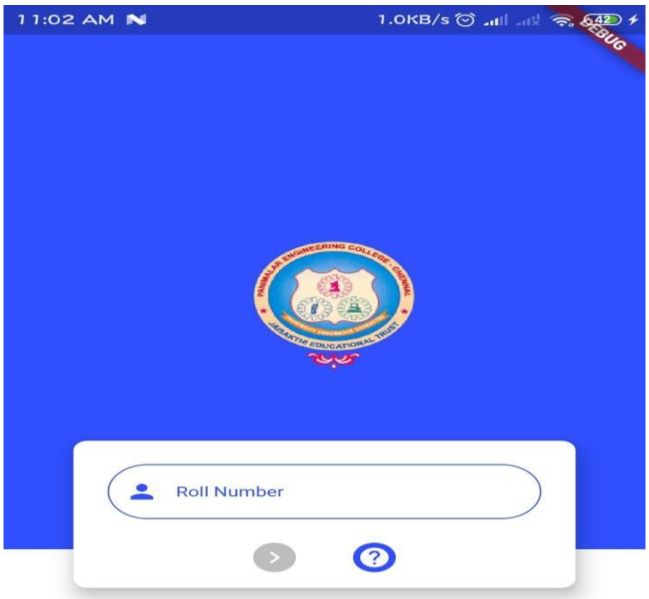
```
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:relative_scale/relative_scale.dart';
class ManualStudentUpload extends StatefulWidget {
 @override
```

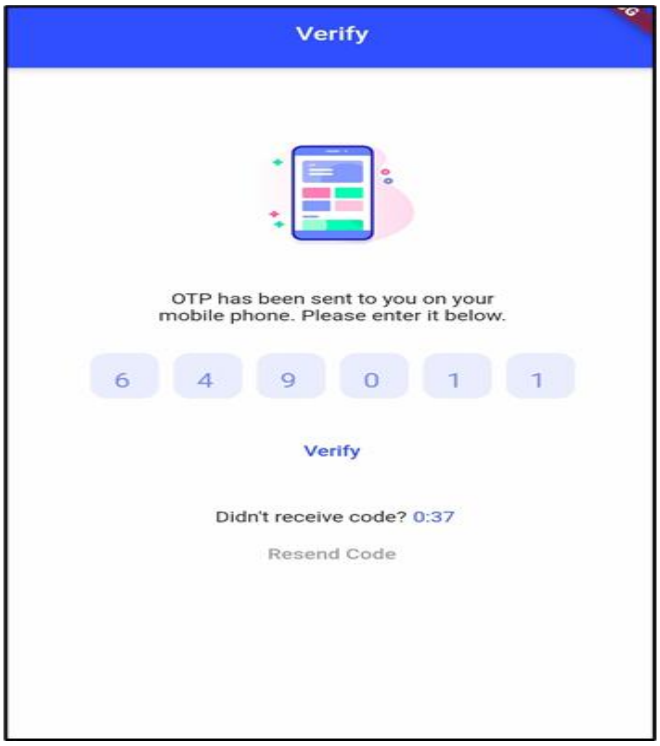
```
  _ManualStudentUploadState createState() =>
_ManualStudentUploadState();
}
class _ManualStudentUploadState extends State<ManualStudentUpload>
{
  Color primary = Color(0xFF304FFE);
  Color secondary = Color(0xFF788Af4);
  Color background = Color(0xfffefefe);
  bool isload = false;
  String name, rollNumber, mobileNumber = "+91";
  final GlobalKey<FormState> _formkey = GlobalKey<FormState>();
  var db = FirebaseDatabase.instance.reference();
  Map<dynamic, dynamic> values;
  @override
  Widget build(BuildContext context) {
    return RelativeBuilder(
        builder: (context, screenHeight, screenWidth, sy, sx) {
      return Scaffold(
        appBar: AppBar(
          title: Text("Upload Student Information"),
          centerTitle: true,
          backgroundColor: primary,
        ),
        // resizeToAvoidBottomInset: false,
        body: Center(
          child: SingleChildScrollView(
            child: Form(
              key: _formkey,
              child: Container(
                height: sy(350),
                width: sx(400),
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    Flexible(
                      flex: 5,
                      child: Image.asset('assets/pecLogo.png'),
                    ),
                    SizedBox(
                      height: 20,
                    ),
                    Flexible(
  flex: 3,
  child: TextFormField(
    keyboardType: TextInputType.name,
    onChanged: (input) => name = input,
    validator: (input) {
      if (input.length == 0) {
        return "Name cannot be empty";
      }
    },
    cursorColor: secondary,
    decoration: InputDecoration(
      labelText: "Name",
      focusColor: secondary,
      labelStyle: TextStyle(
        color: primary,
      ),
      enabledBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
          color: secondary,
        ),
      ),
      focusedBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      errorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      focusedErrorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
    ),
  ),
),
SizedBox(
  height: 15,
),
Flexible(
  flex: 3,
  child: TextFormField(
    textCapitalization: TextCapitalization.characters,
    onChanged: (input) => rollNumber = input,
    validator: (input) {
      if (input.length == 0) {
        return "Enter a valid roll number";
      }
    },
    cursorColor: secondary,
    decoration: InputDecoration(
      labelText: "Roll Number",
      focusColor: secondary,
      labelStyle: TextStyle(
        color: primary,
      ),
      enabledBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
          color: secondary,
        ),
      ),
      focusedBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      errorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      focusedErrorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
    ),
  ),
),
SizedBox(
  height: 15,
),
Flexible(
  flex: 3,
  child: TextFormField(
    keyboardType: TextInputType.number,
    onChanged: (input) => mobileNumber = ("+91" + input),
    validator: (input) {
      if (input.length == 0 && input.length < 10) {
        return "Enter a valid mobile number";
      } else if (input.length < 10) {
        return "Enter a valid mobile number";
      }
    },
    cursorColor: secondary,
    decoration: InputDecoration(
      labelText: "Mobile Number",
      focusColor: secondary,
      labelStyle: TextStyle(
        color: primary,
      ),
      enabledBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
          color: secondary,
        ),
      ),
      focusedBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      errorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
      focusedErrorBorder: OutlineInputBorder(
        borderRadius: BorderRadius.circular(10),
        borderSide: BorderSide(
            color: Color.fromRGBO(48, 79, 254, .2)),
      ),
    ),
  ),
),
SizedBox(
  height: 10,
),
isload==false?RaisedButton(
  child: Text(
    "Upload",
    style: TextStyle(color: background),
  ),
  color: primary,
  onPressed: () async {
    setState(() {
      isload = true;
    });
    await db.once().then((DataSnapshot snap) {
      values = snap.value;
      values = values['LoginDetails'];
      values = values['Student'];
    });
    if (_formkey.currentState.validate()) {
      Fluttertoast.showToast(msg: "Uploading data");
      await db
          .child("LoginDetails/Student")
          .child(rollNumber.trim())
          .set({
        "Name": name.trim(),
        "Mobile": mobileNumber.trim()
      });
      await db
          .child("LoginDetails/Student")
          .child(rollNumber.trim())
          .child("Your Notifications")
          .set({"Dummy": "notification"});
      setState(() {
        isload=false;
      });
      Fluttertoast.showToast(
          msg: "Data uploaded successfully");
    }
```

```
                  else
                    {
                      isload = false;
                    }
                },
              ):CircularProgressIndicator(backgroundColor: Colors.white,)
            ],
          ),
        ),
      ),
    ),
  ),
);
});
}
}
```

SCREENSHOTS

- **Student Login**


11:02 AM
1.0KB/s
DEBUG
PANIMALAR ENGINEERING COLLEGE - CHENNAI
JAISAKTHI EDUCATIONAL TRUST
Roll Number
>


Verify

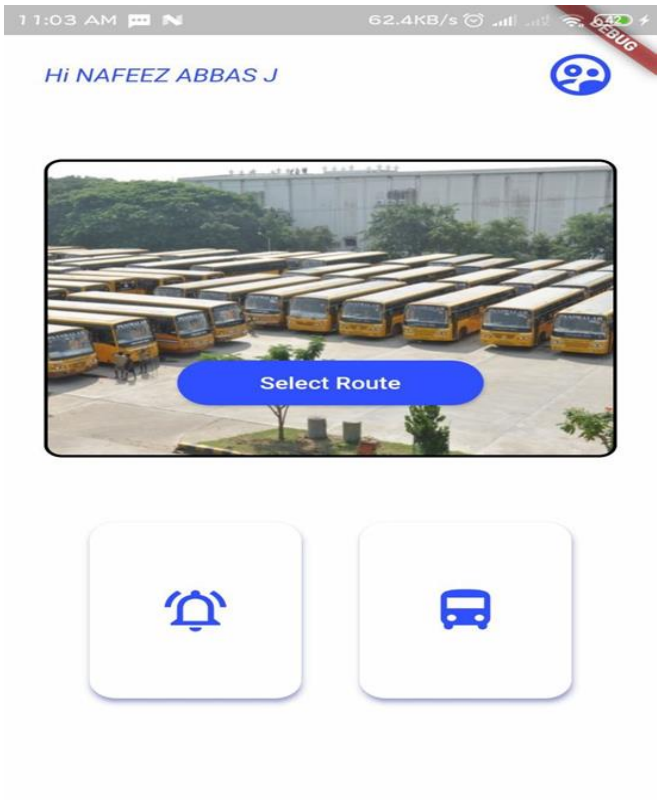
OTP has been sent to you on your mobile phone. Please enter it below.

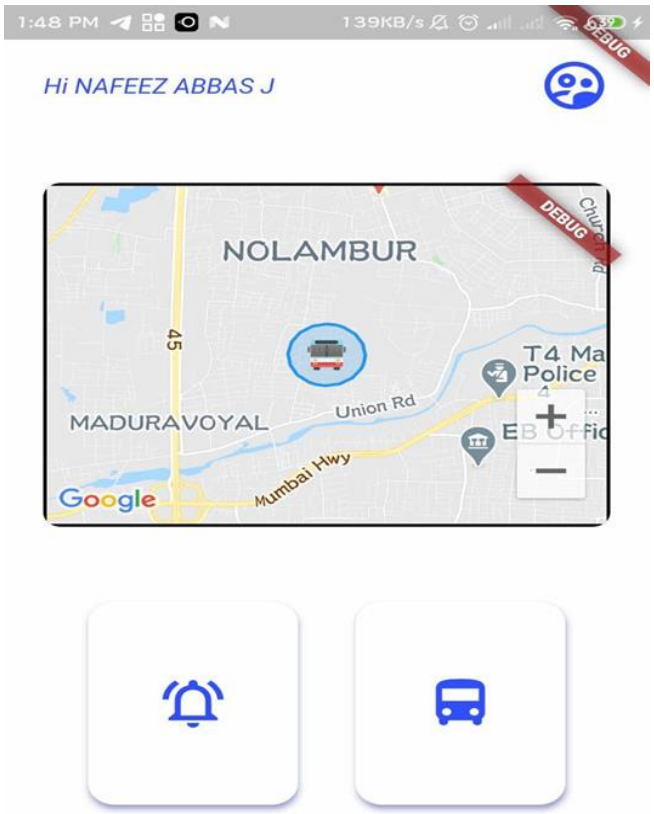
6 4 9 0 1 1

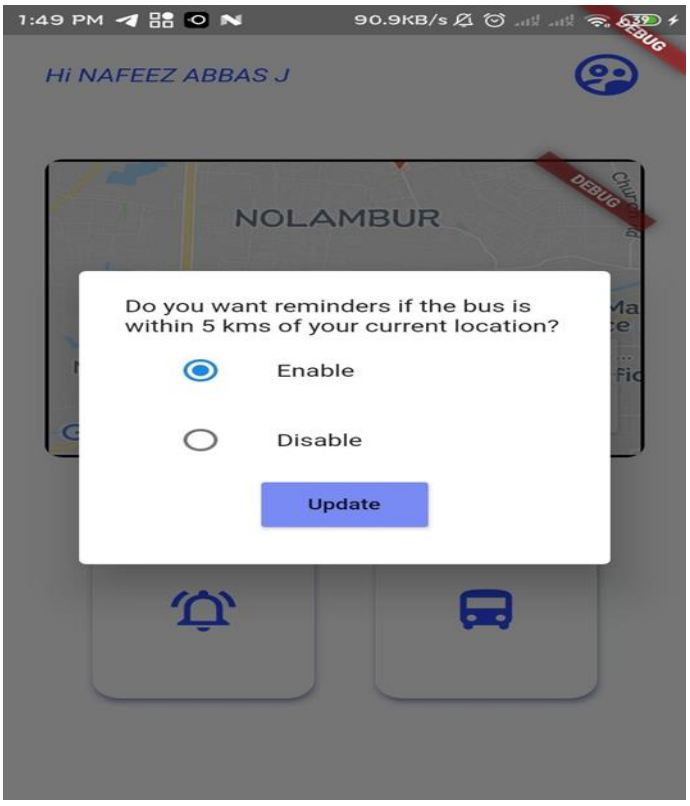
Verify

Didn't receive code? 0:37

Resend Code


11:03 AM
62.4KB/s
DEBUG
Hi NAFEEZ ABBAS J
Select Route


1:48 PM
139KB/s
DEBUG
Hi NAFEEZ ABBAS J
NOLAMBUR
45
T4 Ma
Police
MADURAVOYAL
Union Rd
EB Offic
Mumbai Hwy
Google

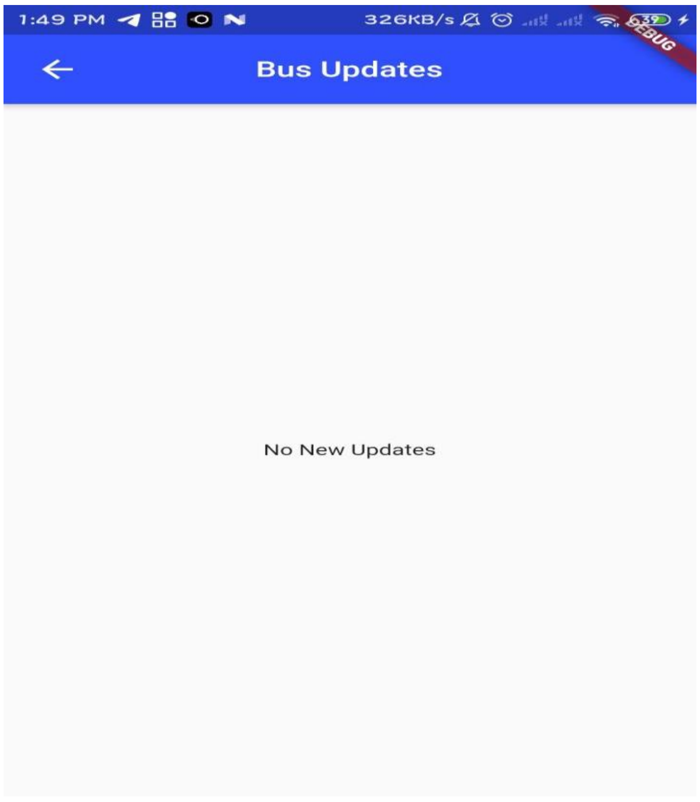


Do you want reminders if the bus is within 5 kms of your current location?

[✓ Enable]

Disable

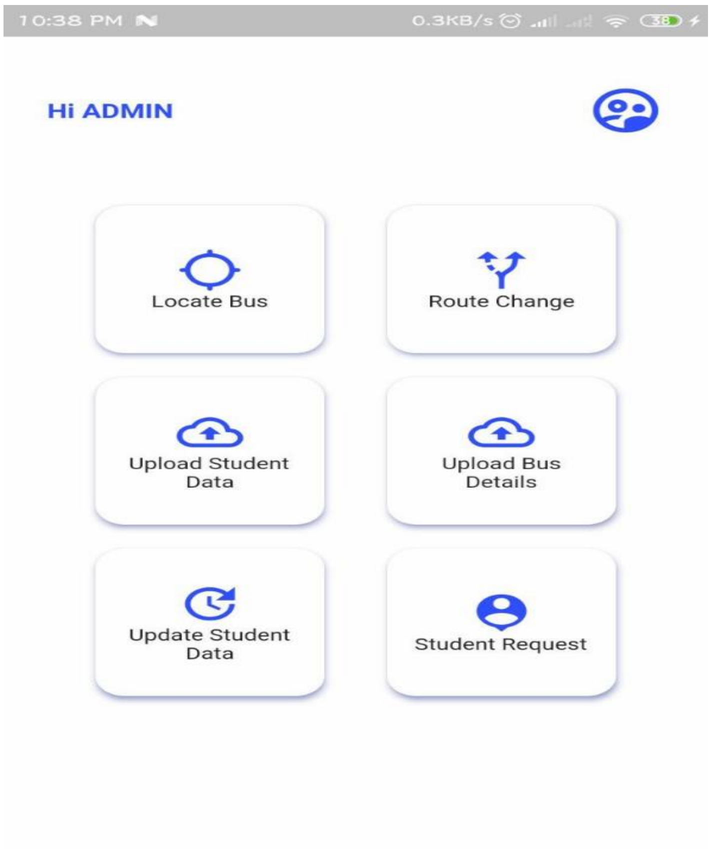
Update

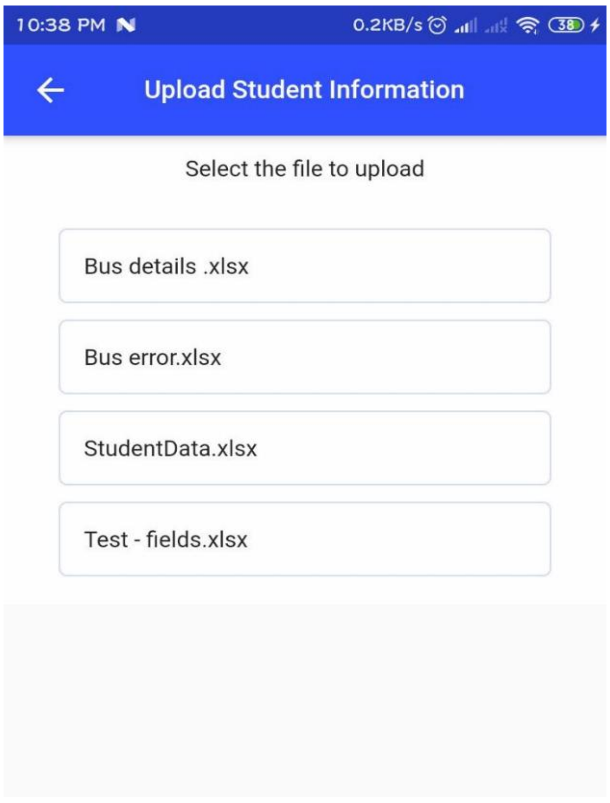


Bus Updates

No New Updates

➢ **Admin Login**





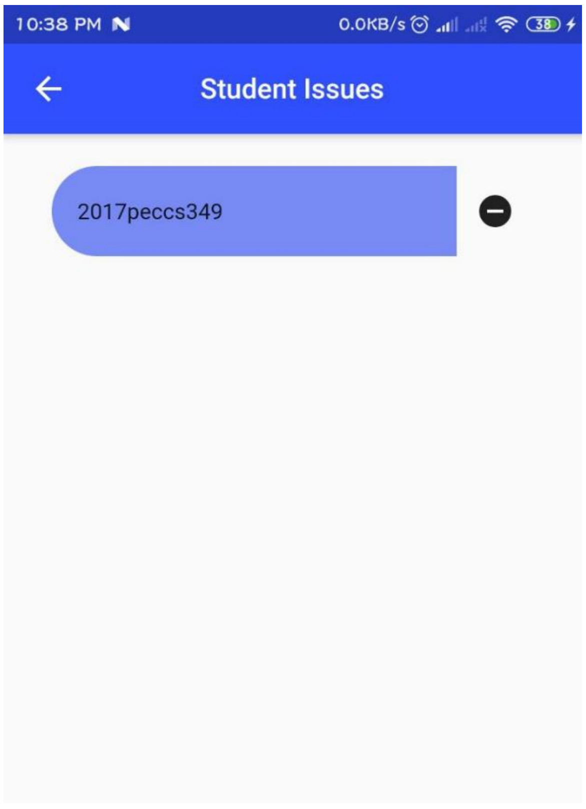
Select the file to upload

Bus details .xlsx

Bus error.xlsx

StudentData.xlsx

Test - fields.xlsx



2017peccs349

FUTURE WORKS

In future there will be an implementation for tracking a student’s location automatically and provide updates if student transfers from one bus to another. It will be very helpful to parents for tracking the student’s exact location. There will be also Estimated Time of Arrival (ETA) of bus which makes the tracking app more precise to get bus location. There will be an individual login for teachers/staffs which will be implemented in future.