BYOD: BRING YOUR OWN DEVICE

(Employee Location Tracking)

MINOR PROJECT II

Submitted By:

Geet Jindal (9919103105)

Mudit Virmani (9919103106)

Sanyam Jain (9919103094)

Akshit Tyagi (9919103101)

Under Supervision Of:

Mr. Shariq Murtuza

Department of CSE/IT

Jaypee Institute of Information Technology University, Noida

MAY 2022

ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to Mr. SHARIQ MURTUZA, Jaypee Institute of Information Technology, India for his/her generous guidance, help and useful suggestions.

I express my sincere gratitude to ______________, Dept. of B.tech. (CSE) India, for his/her stimulating guidance, continuous encouragement, and supervision throughout the course of present work.

I also wish to extend my thanks to _____________ and other classmates for their insightful comments and constructive suggestions to improve the quality of this project work.

Signature(s) of Students

Geet Jindal (9919103105)

Sanyam Jain (9919103094)

Akshit Tyagi (9919103101)

Mudit Virmani (9919103106)

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place:

Date: December 1,2021

Name:	Enrolment:
Geet Jindal	9919103105
Sanyam Jain	9919103094
Akshit Tyagi	9919103101
Mudit Virmani	9919103106

CERTIFICATE

This is to certify that the work titled “BYOD-Bring Your Own Device” submitted by Name of Students of B.Tech.(CSE) of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Digital Signature of Supervisor

Name of Supervisor: Mr. Shariq Murtuza

Designation: ASSISTANT PROFESSOR(GRADE-II)

Date: May 13,2022

ABSTRACT

BYOD (bring your own device) is a policy that allows employees in an organization to use their personally owned devices for work-related activities. Those activities include tasks such as accessing emails, connecting to the corporate network, and accessing corporate apps and data.

It is derived from E-Token technology. E-token is a hardware device used for authentication. Using BYOD we can eliminate the need for hardware using software. Companies no longer need to keep a stock of inventory, manage the logistics of distribution, or consider expiration and token replacement.

Benefits of BYOD over having company devices:

- It is cost effective as companies do not have to set up their own devices and maintain them.
- Increased productivity as employees are comfortable on their own devices.
- No delay in delivering company devices.

TABLE OF CONTENT

LIST OF FIGURES

INTRODUCTION

In today's world of service, safety and security, proper personnel monitoring is vital to ensuring a successful operation. Customer satisfaction is directly linked to a profitable business. Site safety and security is dependent on how well a company monitors its employees. The Information Technology Department strived to develop a user-friendly application which would enable an employer to track the location of the employees to improve resource management. The central concept of the application is to enable the employer to track the location of the employee.

OBJECTIVE

We are using BYOD as an employee tracking system. Which will track user geolocation, call logs, browser history, app usage.

METHODOLOGY

SDLC Model used:

Incremental Model

- We are dividing the whole project into smaller modules.
- After deploying a module successfully, we will work on the next module following the CPMCD method.
- Feedback can be considered.

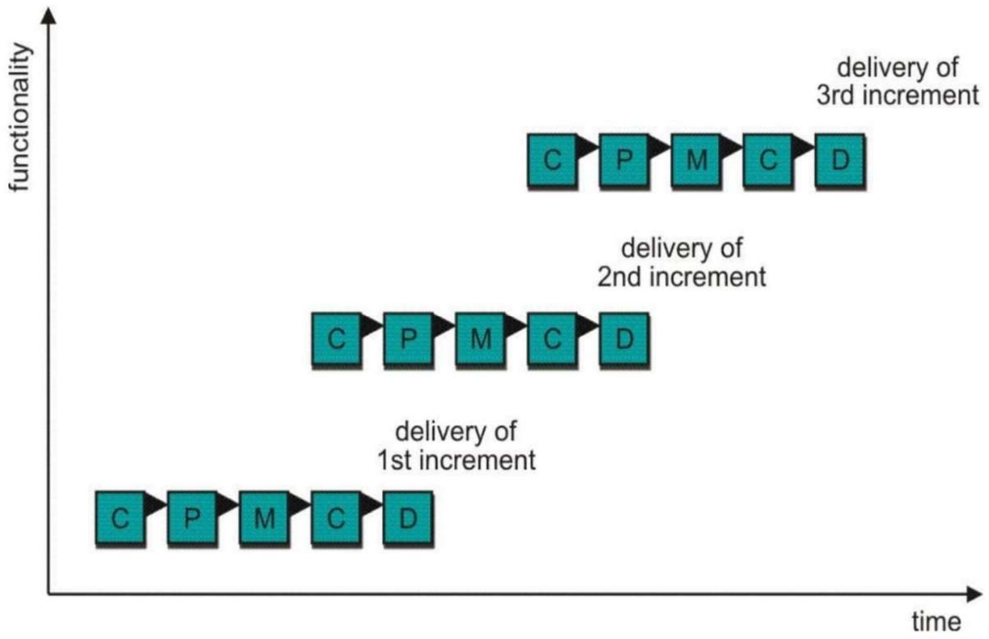

functionality
delivery of 3rd increment
C P M C D
delivery of 2nd increment
C P M C D
delivery of 1st increment
C P M C D
time

Fig 8.1

BACKGROUND STUDY

To build an android app that can track the real-time location of employees.

Bring your own device also called Bring your own Technology BYOT refers to being allowed to use one's personally owned device, rather than being required to use an officially provided device. The main focus of this application is on the workplace, where it refers to a policy of permitting employees to bring personally owned devices (laptops, tablets, smartphones, etc.) to work and use those devices to track and monitor employee’s activity.

Requirement Analysis

Functional requirements:

- An employee/employer should be able to login.
- Admin should be able to register a new employee
- Employers/managers should be able to view/track employee’s locations in real time.
- Employee’s geo location should be recorded for future reference.
- Employers should be able to view recorded data.

Non-Functional requirements:

- Application UI must be easy to understand.
- Application should handle all privacy concerns
- App should be robust.
- Login authentication must be secure and accurate.
- Server client connection should be secure.
- Location tracking must be accurate.
- There should be less delay to update results at server end.

Technology Used

- **Android Studio**
    - We used android version 8 API level 26 so the app can run on more than 82.7% of phones.
    - We emulated Google Pixel 2 with android Oreo.
- **Flutter**
    - Only one language, Dart, is used for different platforms.
    - Same Dart Scripts Work in iOS and Android Native Apps Without Modification.
    - Dart is Supported by Google.
    - Split Seconds reload.
- **Firebase**
    - Firebase's different backend-as-a-service (BaaS) features help you develop high-quality apps, grow your user base,

Detailed Design:

Use Case Diagram:

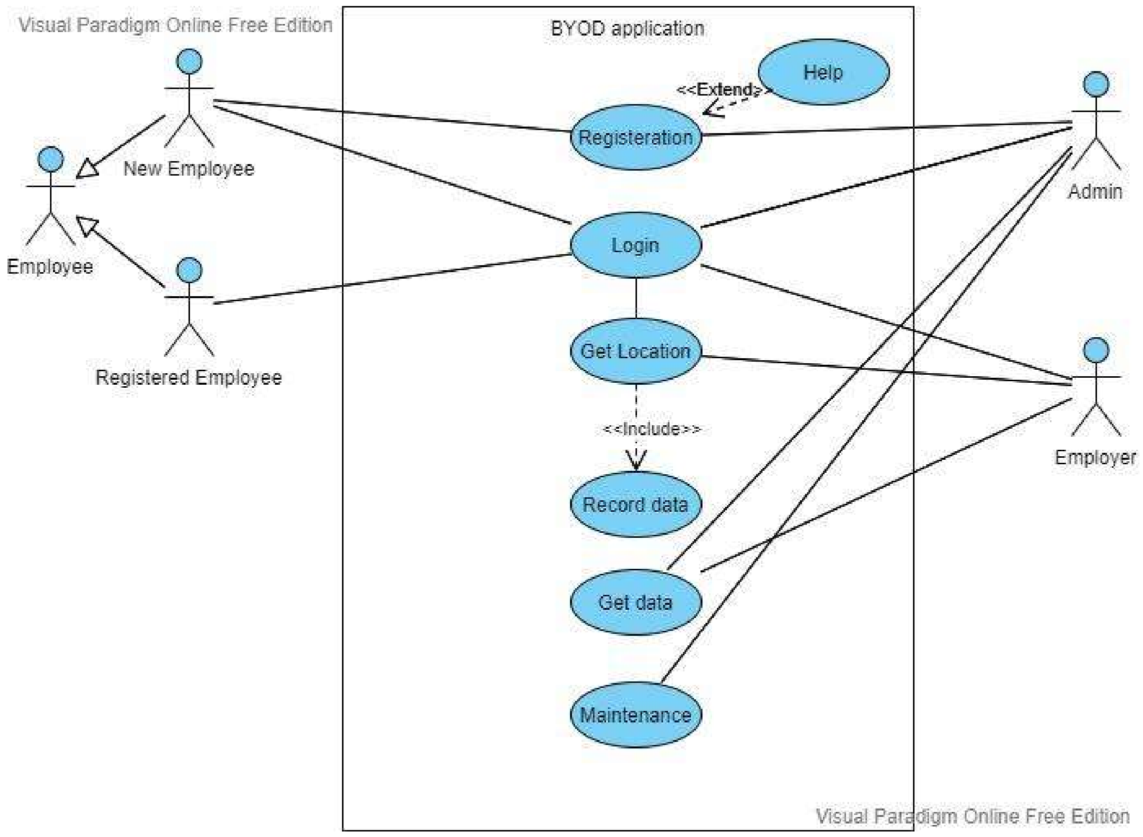


Fig 11.1

Activity Diagram:

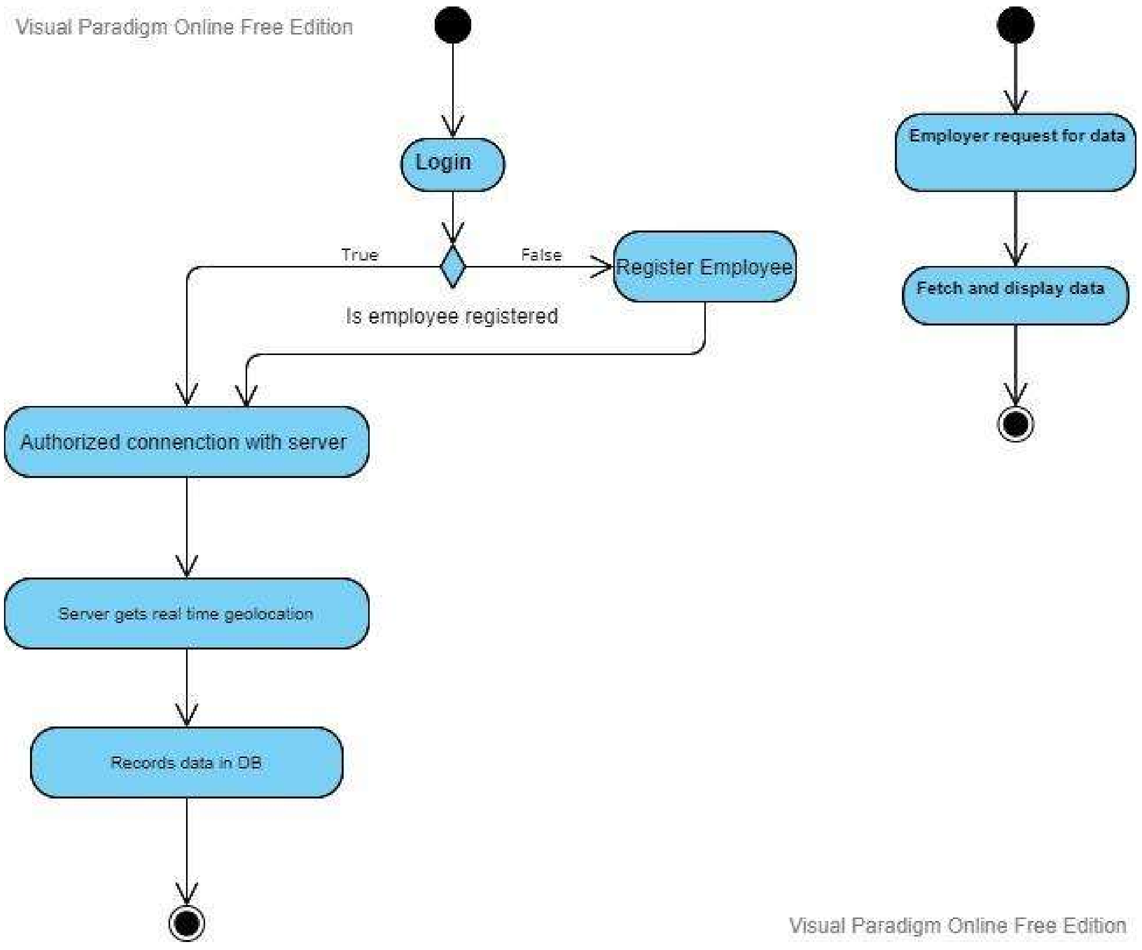

Visual Paradigm Online Free Edition
Login
True
False
Register Employee
Is employee registered
Authorized connenction with server
Server gets real time geolocation
Records data in DB
Employer request for data
Fetch and display data
Visual Paradigm Online Free Edition

Fig. 11.2

Implementation:

Main.dart

```dart
import 'dart:async';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:google_map_live/map.dart';
import 'package:location/location.dart' as state;
import 'package:permission_handler/permission_handler.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(MaterialApp(home: MyApp()));
}

class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  final state.Location location = state.Location();
  StreamSubscription<state.LocationData>? _locationSubscription;

  @override
  void initState() {
    super.initState();
    _requestPermission();
    location.changeSettings(interval: 3000, accuracy:
state.LocationAccuracy.high);
    location.enableBackgroundMode(enable: true);
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Color.fromRGBO(30, 30, 30, 1),
        title: Text('Byod Location Tracking'),
      ),

      body: Column(
        children: [
          TextButton(
              style: TextButton.styleFrom(
                shape: new RoundedRectangleBorder(borderRadius: new
BorderRadius.circular(15.0)),
               backgroundColor:Colors.blue,
               primary:Colors.white,
              ),
              onPressed: () {
                _getLocation();
              },
              child: Text('Push location')),
          TextButton(
              style: TextButton.styleFrom(
                shape: new RoundedRectangleBorder(borderRadius: new
BorderRadius.circular(15.0)),
                backgroundColor:Colors.green,
                primary:Colors.white,
              ),
              onPressed: () {
                _listenLocation();
              },
              child: Text('Allow live location')),
          TextButton(
              style: TextButton.styleFrom(
                shape: new RoundedRectangleBorder(borderRadius: new
BorderRadius.circular(15.0)),
                backgroundColor:Colors.red,
                primary:Colors.white,
              ),
              onPressed: () {
                _stopListening();
              },
              child: Text('Abort location')),
          Expanded(
              child: StreamBuilder(
            stream:
                FirebaseFirestore.instance.collection('location').snapshots(),
            builder: (context, AsyncSnapshot<QuerySnapshot> snapshot) {
              if (!snapshot.hasData) {
                return Center(child: CircularProgressIndicator());
              }
              return ListView.builder(
                  itemCount: snapshot.data?.docs.length,
                  itemBuilder: (context, index) {
                    return ListTile(
                      title:
                          Text(snapshot.data!.docs[index]['name'].toString()),
                      subtitle: Row(
                        children: [
                          Text(snapshot.data!.docs[index]['latitude']
                              .toString()),
                          SizedBox(
                            width: 20,
                          ),
                          Text(snapshot.data!.docs[index]['longitude']
                              .toString()),
                        ],
                      ),
                      trailing: IconButton(
                        icon: Icon(Icons.directions),
                        onPressed: () {
                          Navigator.of(context).push(MaterialPageRoute(
                              builder: (context) =>
                                  MyMap(snapshot.data!.docs[index].id)));
                        },
                      ),
                    );
                  });
            },
          )),
        ],
      ),
    );
  }

  _getLocation() async {
    try {
      final state.LocationData _locationResult = await location.getLocation();
      await
FirebaseFirestore.instance.collection('location').doc('user1').set({
        'latitude': _locationResult.latitude,
        'longitude': _locationResult.longitude,
        'name': 'Group57'
      }, SetOptions(merge: true));
    } catch (e) {
      print(e);
    }
  }

  Future<void> _listenLocation() async {
    _locationSubscription = location.onLocationChanged.handleError((onError) {
      print(onError);
      _locationSubscription?.cancel();
      setState(() {
        _locationSubscription = null;
      });
    }).listen((state.LocationData currentlocation) async {
      await
FirebaseFirestore.instance.collection('location').doc('user1').set({
        'latitude': currentlocation.latitude,
        'longitude': currentlocation.longitude,
        'name': 'Group57'
      }, SetOptions(merge: true));
    });
  }

  _stopListening() {
    _locationSubscription?.cancel();
    setState(() {
      _locationSubscription = null;
    });
  }

  _requestPermission() async {
    var status = await Permission.location.request();
    if (status.isGranted) {
      print('done');
    } else if (status.isDenied) {
      _requestPermission();
    } else if (status.isPermanentlyDenied) {
      openAppSettings();
    }
  }
}
```

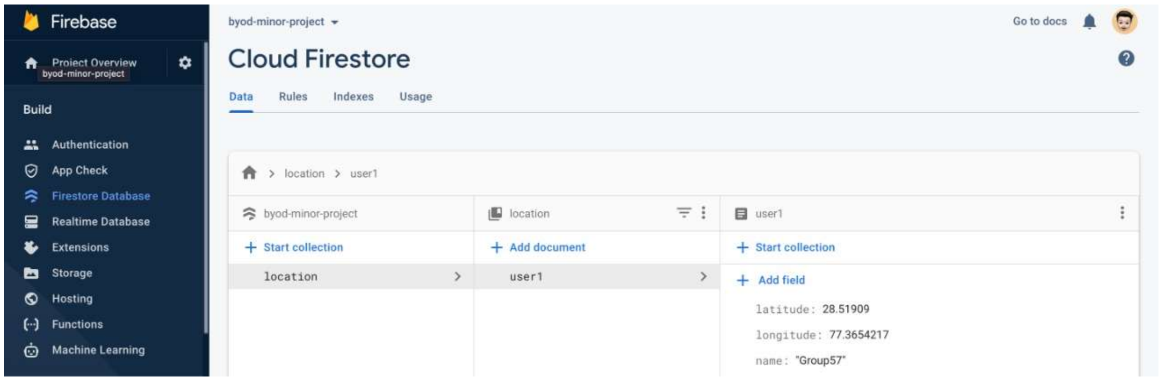
Map.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/material.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:location/location.dart' as loc;

class MyMap extends StatefulWidget {
  final String user_id;
  MyMap(this.user_id);
  @override
  _MyMapState createState() => _MyMapState();
}

class _MyMapState extends State<MyMap> {
  final loc.Location location = loc.Location();
  late GoogleMapController _controller;
  bool _added = false;
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        body: StreamBuilder(
      stream: FirebaseFirestore.instance.collection('location').snapshots(),
      builder: (context, AsyncSnapshot<QuerySnapshot> snapshot) {
        if (_added) {
```

```
        mymap(snapshot);
      }
      if (!snapshot.hasData) {
        return Center(child: CircularProgressIndicator());
      }
      return GoogleMap(
        mapType: MapType.normal,
        markers: {
          Marker(
              position: LatLng(
                snapshot.data!.docs.singleWhere(
                    (element) => element.id == widget.user_id)['latitude'],
                snapshot.data!.docs.singleWhere(
                    (element) => element.id == widget.user_id)['longitude'],
              ),
              markerId: MarkerId('id'),
              icon: BitmapDescriptor.defaultMarkerWithHue(
                  BitmapDescriptor.hueMagenta)),
        },
        initialCameraPosition: CameraPosition(
            target: LatLng(
              snapshot.data!.docs.singleWhere(
                  (element) => element.id == widget.user_id)['latitude'],
              snapshot.data!.docs.singleWhere(
                  (element) => element.id == widget.user_id)['longitude'],
            ),
            zoom: 14.47),
        onMapCreated: (GoogleMapController controller) async {
          setState(() {
            _controller = controller;
            _added = true;
          });
        },
      );
    },
  ));
}

Future<void> mymap(AsyncSnapshot<QuerySnapshot> snapshot) async {
  await _controller
      .animateCamera(CameraUpdate.newCameraPosition(CameraPosition(
          target: LatLng(
            snapshot.data!.docs.singleWhere(
                (element) => element.id == widget.user_id)['latitude'],
            snapshot.data!.docs.singleWhere(
                (element) => element.id == widget.user_id)['longitude'],
          ),
          zoom: 14.47)));
}}
```

Experimental Results and Analysis

Emulator Output and Database Logs



```
latitude: 28.51909
longitude: 77.3654217
name: "Group57"
```

Fig 13.2.1

A “Enable live location” button, with the same style as above button, and when pressed, calls _listenLocation function. This function collects location data from geolocation API continuously, and stores the collected data in the firebase database.

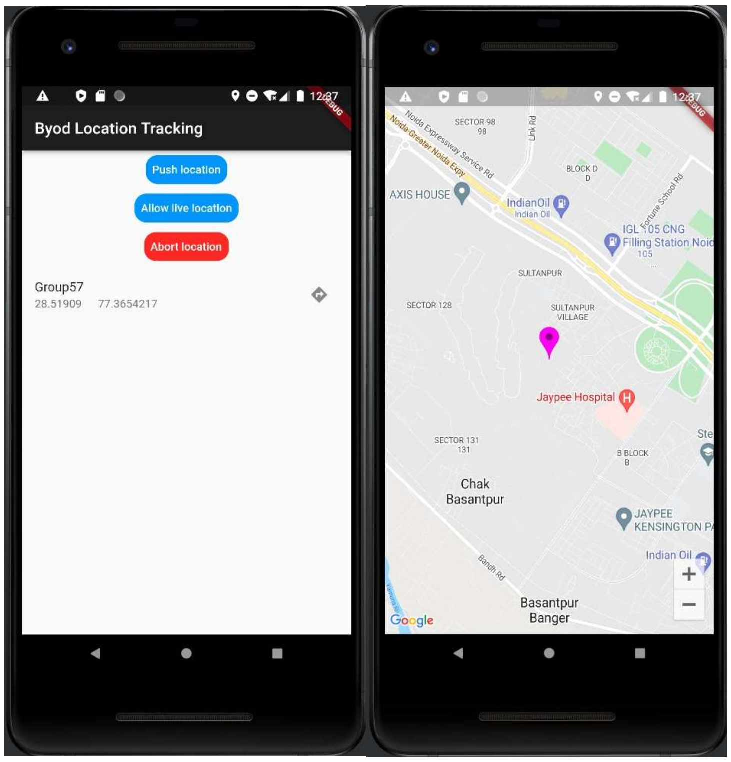
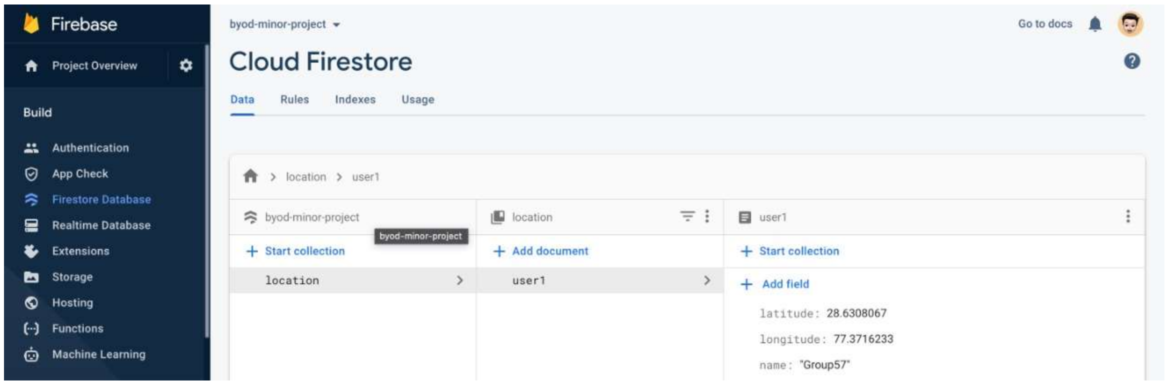


Fig 13.2.2



```
location
user1
latitude: 28.6308067
longitude: 77.3716233
name: "Group57"
```

Fig 13.2.3

A "Push location" button, with blue colour and rounded borders, which when pressed calls _getLocation() function. This function collects location data from geolocation API and stores them in respective variables. Since this function is of asynchronous type, it will wait for location data and will not be executed until it gets a location.

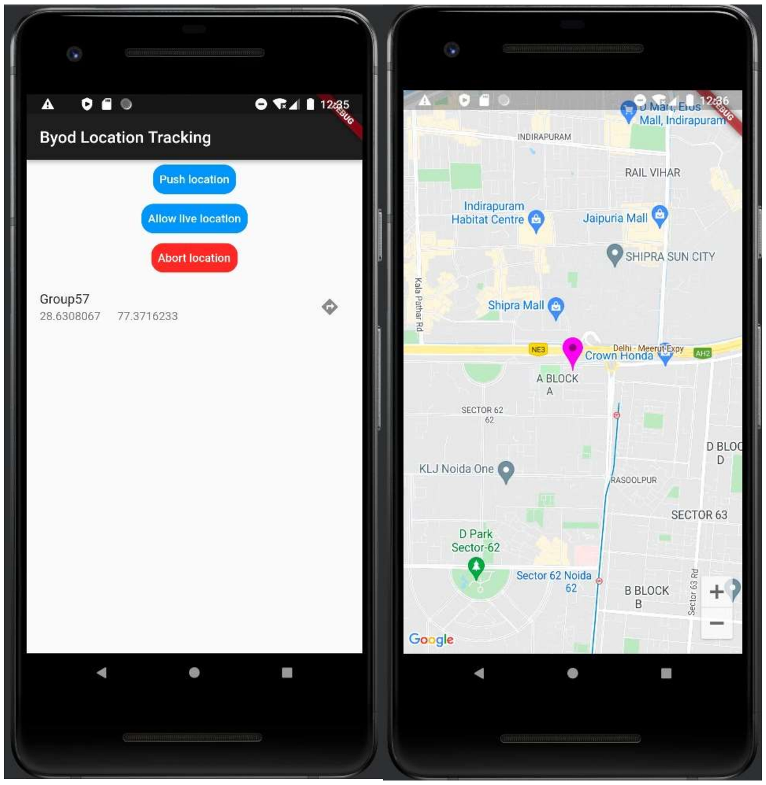


Fig 13.2.4

A “Stop location” button is coloured red, has a rounded border, and calls _stopListening() function. This function stops collecting data from the API and sets the subscription to null and the last location is logged.

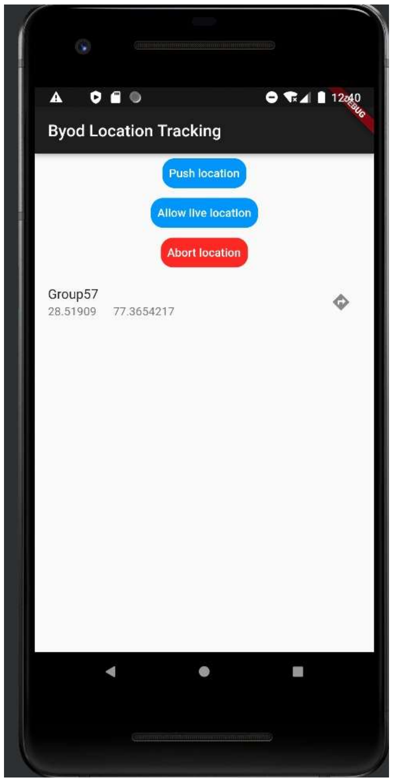
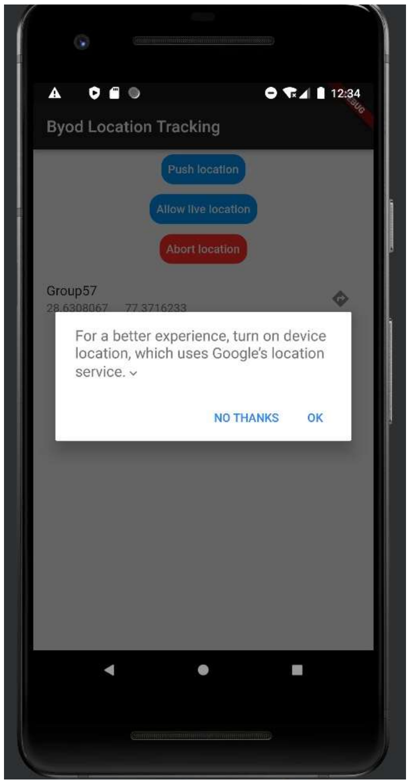


Fig 13.2.5

Interval to update live location is set to 300ms, accuracy is set to high, and the app is enabled in background. When the app is launched, _requestPermission() function checks whether the app has location permission or not. If it does not have the required permission, it prompts a request message to the user to give permission, this function runs until permission is granted.



For a better experience, turn on device location, which uses Google's location service.

Fig 13.2.6

CONCLUSION

- All the functionalities have been fulfilled.
- The Data of Location of the Employee is properly saved in the Firebase.

FUTURE SCOPE

- Try to create modules for further tracking, like browser history.
- Try to Alert last location data to the employer when GPS is turned off.

REFERENCES

- https://developer.android.com/studio
- https://docs.flutter.dev/
- https://dart.dev/guides
- https://developer.android.com/docs
- https://developer.android.google.cn/reference
- https://developers.google.com/maps/documentation/geolocation/overview
- https://firebase.google.com/docs