

1.ABSTRACT

A manufacturing company is producing various products in its factory. The company sells to distributors in various districts. The company is now selling products to all over Tamil Nadu. In future sell products in multiple states.

- Regional Sales Manager (RSM) For each state, controls all the below sales force.
- Assistant Sales Manager (ASM) Under each Regional Sales Manager there will be few Assistant Sales Managers who directly sells to distributors as well as controls few ASO below.
- Assistant Sales Officer (ASO) These people are the real representatives who visits the Distributors to sell products. These representatives should also visit outlets (such as Bakery, hotels etc.) who actually buy product from distributors.

TECH STACK :

FRONTEND :

FLUTTER

Flutter is an open-source UI software development kit created by Google. It is used to develop cross platform applications from a single codebase for any web browser, Fuchsia, Android, iOS, Linux, macOS, and Windows. Widely used for mobile app development

BACKEND:

DART:

Dart is an official high level programming language for flutter created by Google.

DATABASE:

FIREBASE:

Firebase is a google cloud service for storage, messaging, authentication etc...

2.INTRODUCTION

This new app helps salespeople sell better! It makes their daily work easier, keeps everyone connected, and gathers info to see how things are going.

2.1 SYSTEM OVERVIEW

Describe the app's overall functionality at a high level. Explain how the app will be used by different user roles (RSM, ASM, ASO) and how it will support their daily tasks.

- Each day all the RSM, ASM and ASO are to mark attendance and start the day.
- During attendance marking, it has to capture the current location.
- Representatives will visit distributors and book orders.
- Representatives will visit outlets and collect feedback and competitive product details. Provision is to be made to upload images.
- Provision to apply for leave by the ASO. These are to be intimated to ASM and is to be approved.
- ASM should be able to keep an eye on Monthly Orders and compare ASO results.
- RSM should be able to do whatever the ASM is doing along with consolidated Sales summary analysis.
- As and when there is a change in the price list, or added new product or functionality we need to send notifications from all the employees.

2.2 OBJECTIVE OF THE PROJECT

- Simplify Attendance Management: Develop a feature for RSMs, ASMs, and ASOs to easily mark attendance each day, ensuring all sales representatives start their day efficiently.
-
- Capture Location during Attendance: Incorporate functionality to capture the current location of RSMs, ASMs, and ASOs during attendance marking, providing real-time tracking capabilities.
-
- Order Booking at Distributors: Enable representatives to visit distributors and book orders seamlessly through the app, streamlining the sales process.
-
- Collect Feedback from Outlets: Allow representatives to visit outlets, such as bakeries and hotels, to gather feedback and details about competitive products. Include an option to upload images for reference.
-
- Leave Application for ASOs: Provide ASOs with the ability to apply for leave through the app. Notifications will be sent to ASMs for approval.
-
- Monitoring ASO Activities: Enable ASMs to monitor the activities of ASOs, ensuring efficient workflow and task management.
-
- Monthly Order Tracking: Allow ASMs to track monthly orders and compare the performance of ASOs, facilitating better sales management and analysis.
-
- Sales Summary Analysis: Provide RSMs with access to ASM activities and consolidated sales summary analysis, allowing for better decision-making and strategic planning.
-
- Price List Updates: Implement a feature to notify the sales force team of any changes in the price list, ensuring all representatives are informed promptly.

2.3 SCOPE OF THE PROJECT

Functional Scope:

The app will facilitate attendance marking for RSMs, ASMs, and ASOs, capturing their current location.

It will allow representatives to visit distributors and outlets to book orders and collect feedback, including uploading images.

ASOs can apply for leave through the app, with approval from ASMs.

ASMs can monitor ASO activities, track monthly orders, and compare ASO performance.

RSMs will have access to ASM activities and consolidated sales summary analysis.

Technical Scope:

The app will be developed using cross-platform technologies to ensure compatibility across different devices and operating systems.

It will utilize Google Firebase Cloud for database management, authentication, and other cloud services.

Integration with third-party services like Geolocator for location tracking will be included.

Security measures, such as encryption and user authentication, will be implemented to protect data.

User Scope:

The primary users of the app will be RSMs, ASMs, and ASOs within the sales force team.

They will interact with the app to perform various tasks related to sales operations, attendance management, order booking, feedback collection, and performance analysis.

Feature Scope:

The app will include features for attendance marking, order booking, feedback collection, leave management, activity monitoring, and sales analysis.

It will support functionalities such as location tracking, image uploading, and real-time notifications for price list updates.

3.REQUIREMENT AND SPECIFICATION

This section details the functional and non-functional requirements of the app. Here's a breakdown:

Functional Requirements:

User Roles and Functionalities: Define roles such as RSMs, ASMs, and ASOs, specifying their respective functionalities like attendance marking, order booking, feedback collection, and leave management.

Data Requirements: Specify the data needed for the app, including customer information, product details, orders, and feedback.

Reporting Needs: Define reporting requirements such as sales reports and performance analysis to aid decision-making.

Security Considerations: Outline security measures like user authentication and data encryption to protect sensitive information.

Non-functional Requirements:

Performance Expectations: Specify performance targets for response times and data synchronization to ensure efficient operation.

User Interface Design: Define requirements for the user interface, emphasizing usability and accessibility for ease of use.

Scalability: Ensure the app can accommodate future growth in terms of data volume and user base without compromising performance.

3.1 HARDWARE INTERFACE

Device Compatibility: This app size is below 22MB so it can run any kind of android device(Android 5.0 or Higher).

GPS Functionality: The app requires location tracking during attendance marking or order delivery, outline the need for devices with GPS capabilities.

Camera Access: If the app allows for image uploads (e.g., feedback collection), specify if devices with a built-in camera are necessary.

Connectivity Requirements: Detail any requirements for Wi-Fi or cellular network connectivity to ensure smooth data exchange and communication.

3.2 SOFTWARE INTERFACE

Database Management System (DBMS):

The app interacts with Google Firebase Cloud Firestore, a NoSQL cloud database or document-based database.

Data exchange is facilitated through Firebase SDKs and APIs, allowing seamless integration with the app's frontend.

Security Protocols:

Data exchanged between the app and Google Firebase Cloud is encrypted both in transit and at rest.

Firebase Authentication is utilized for user authentication, providing secure access to the app's features.

Firebase Security Rules are employed to control access to Firestore data, ensuring that only authorized users can read or write data according to defined rulesets.

4.SYSTEM DESIGN

4.1 UML DIAGRAM

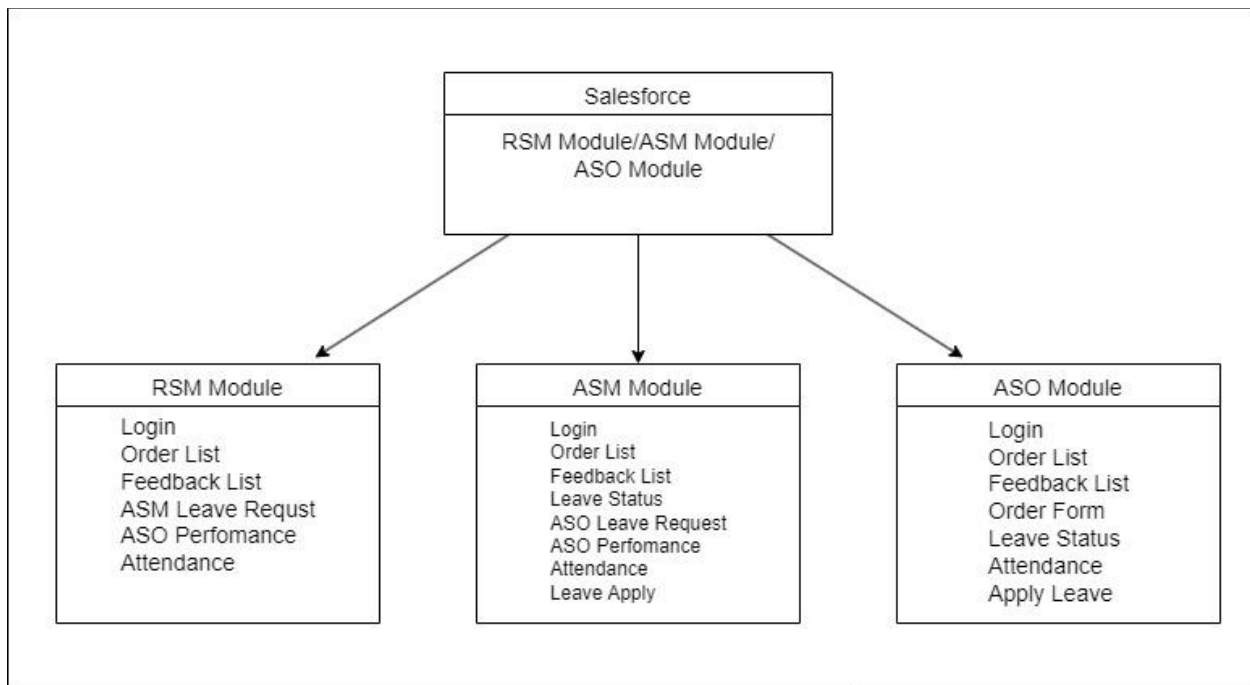


Fig.4.1.1. Flow Chart for Salesforce Mobile App

4.2.2 FLOW-CHART DIAGRAM

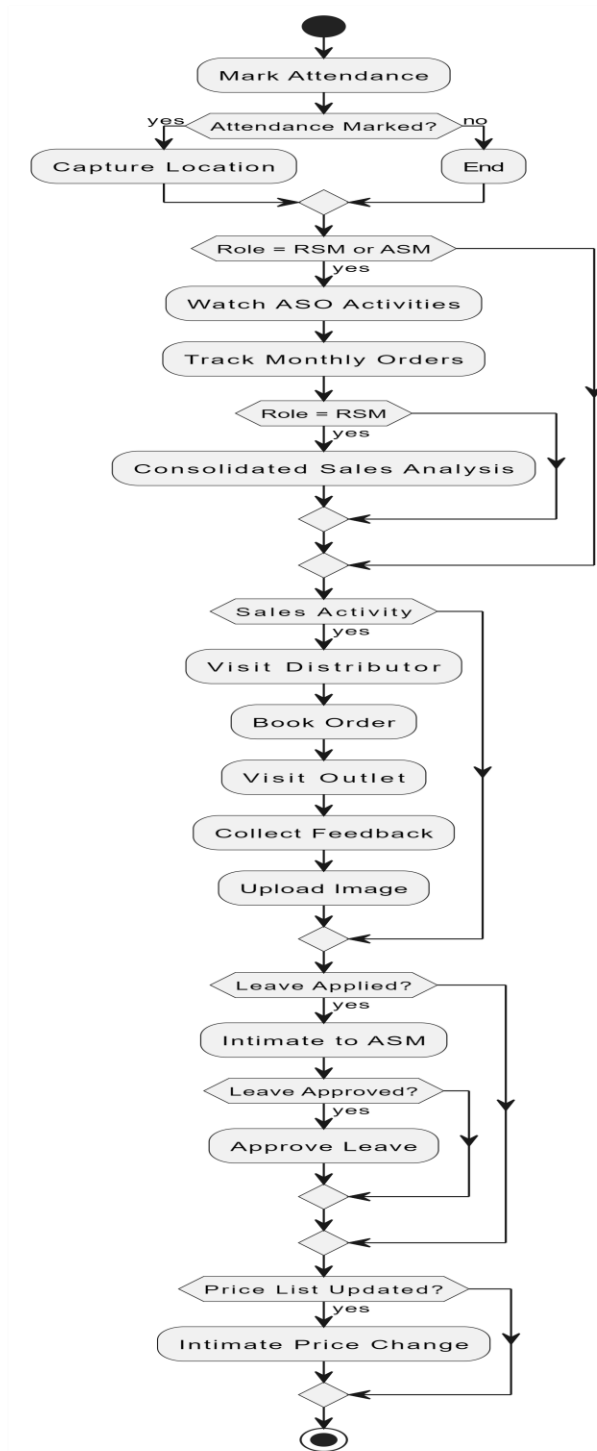


Fig.4.2.2. Flow Chart for Salesforce Mobile App

4.2.1 USE CASE DIAGRAM

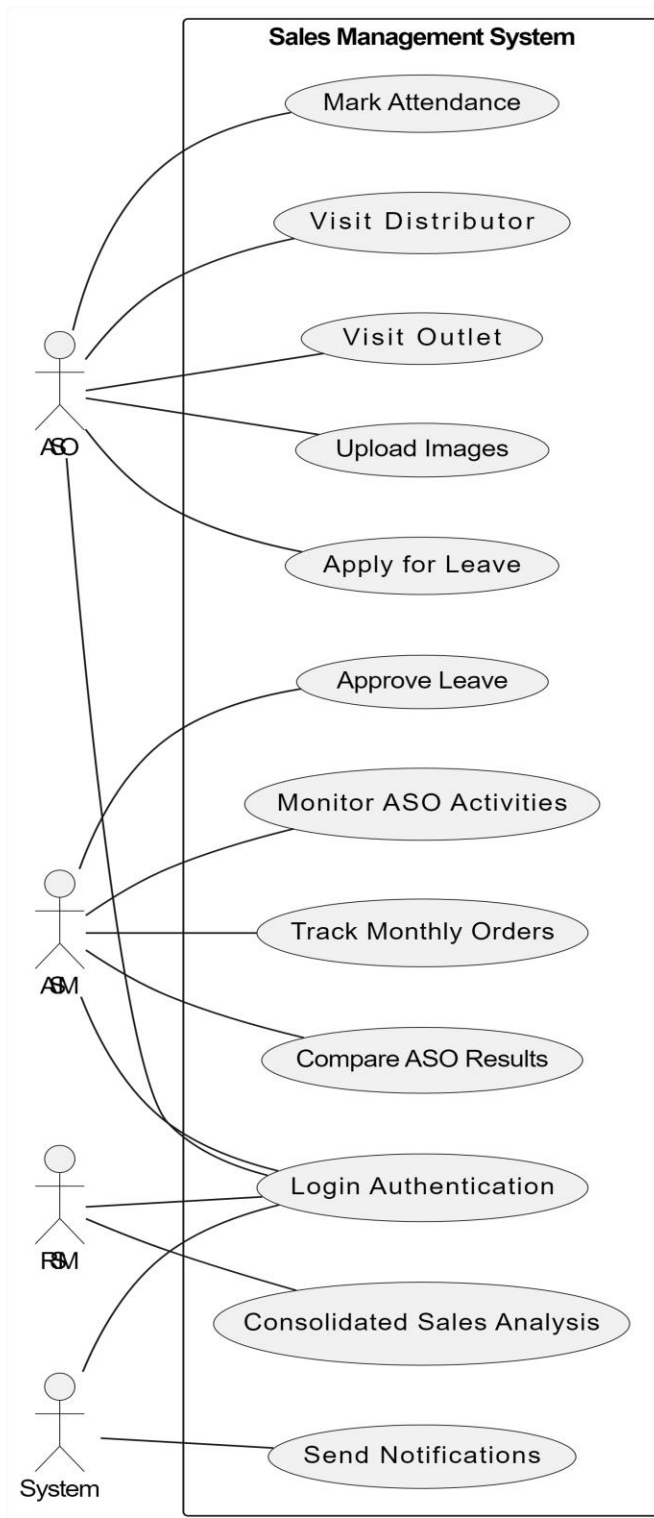


Fig.4.2.1. Flow Chart for Salesforce Mobile App

5.IMPLEMENTATION

5.1. FLUTTER

Flutter is an open-source UI software development kit (SDK) developed by Google. It is used to build natively compiled applications for mobile, web, and desktop from a single codebase. Flutter provides a fast and expressive way to build user interfaces, with features like hot reload for rapid iteration and a rich set of pre-designed widgets for building beautiful and responsive apps.

Single Codebase: Flutter allows developers to write code once and deploy it on multiple platforms, including Android, iOS, web, and desktop.

Fast Development: With features like hot reload, developers can see changes to their code reflected instantly in the app, speeding up the development process and enabling rapid iteration.

Rich Set of Widgets: Flutter provides a comprehensive set of pre-designed widgets for building complex user interfaces, including material design and Cupertino (iOS-style) widgets.

High Performance: Flutter apps are compiled directly to native machine code, resulting in high performance and smooth animations.

Customizable: Flutter allows for extensive customization of UI components, enabling developers to create unique and visually appealing designs.

Open-source: Flutter is an open-source project, with a large and active community contributing to its development and ecosystem.

5.2. DART

Dart: Dart is an object-oriented, class-based programming language developed by Google and also official flutter framework language.

It is the primary language used to write Flutter apps. Dart is optimized for building web, server, and mobile applications, and it features strong typing, asynchronous programming support, and a modern syntax.

Dart is designed to be easy to learn and productive for developers, with features like optional static typing, garbage collection, and support for both just-in-time (JIT) and ahead-of-time (AOT) compilation.

5.3. FIREBASE

Firebase is a cloud platform developed by Google that provides a suite of tools and services for building and managing web and mobile applications. It offers various features, including:

Realtime Database: A NoSQL cloud database that allows developers to store and sync data between users in real-time. It's often used for applications that require real-time updates, such as chat apps or collaborative editing tools.

Authentication: Firebase Authentication provides easy-to-use SDKs and ready-made UI libraries to authenticate users with email/password, phone number, social media accounts (Google, Facebook, Twitter, etc.), and more.

Cloud Firestore: Firestore is a flexible, scalable database for mobile, web, and server development. It's a NoSQL document database that allows for more structured data storage and querying compared to the Realtime Database.

Cloud Functions: Firebase Cloud Functions allows developers to run backend code in response to events triggered by Firebase features and HTTPS requests. It's often used for tasks like sending notifications, processing payments, or running custom business logic.

Storage: Firebase Storage provides secure file uploads and downloads for images, videos, and other files. It's designed to scale automatically and integrate seamlessly with Firebase Authentication and Firebase Security Rules.

Hosting: Firebase Hosting allows developers to deploy web apps and static content with a single command. It provides SSL encryption, global CDN distribution, and custom domain support out of the box.

Firebase Cloud Messaging (FCM) enables developers to send push notifications to users across platforms (iOS, Android, and web). It supports targeted messaging, user segmentation, and analytics integration.

6.APPENDICES

6.1 CODING

Main.dart

```
import 'package:flutter/material.dart';  
import 'package:firebase_core/firebase_core.dart';
```

```

import 'package:salesforce/api/firebase_api.dart';
import 'package:salesforce/services/signIn_or_not.dart';

import 'firebase_options.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );

  // this is for FCM messaging
  await FirebaseApi().initNotification();

  runApp(
    const MyApp(),
  );
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Salesforce',

```

```

theme: ThemeData(
  colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
  useMaterial3: true,
  primarySwatch: Colors.blue,
),
home: const SignInOrNot(),
);
}
}

```

rsm_page.dart

```

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_nav_bar/google_nav_bar.dart';
import 'package:salesforce/components/present_absent_dialog.dart';
import 'package:salesforce/screens/aso_monthly_performence.dart';
import 'package:salesforce/screens/employee_activity.dart';
import 'package:salesforce/screens/login_page.dart';
import 'package:salesforce/screens/show_feedback.dart';
import 'package:salesforce/screens/show_order_list.dart';
import 'package:salesforce/services/attendance_marking.dart';

```

```

class RSM extends StatefulWidget {
  const RSM({super.key});

  @override

```

```

State<RSM> createState() => _RSMState();
}

class _RSMState extends State<RSM> {
  void signOutUser() async {
    // debugPrint('Enter function');

    // If user press confirm then logout
    if (await _showDialogWindow() == true) {
      await FirebaseAuth.instance.signOut();

      Navigator.pushReplacement(
        context, MaterialPageRoute(builder: (context) => const LogInPage()));
    }
  }
}

Future<bool> _showDialogWindow() async {
  return await showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text(
        'Logout',
      ),
      backgroundColor: Colors.lightBlue.shade100,
      content: const Text(
        'Are you sure want to log out salesforce?',
        style: TextStyle(fontSize: 15),

```



```

    ),
    actions: [
        TextButton(
            onPressed: () => Navigator.pop(context, false),
            child: const Text('Cancel'),
        ),
        TextButton(
            onPressed: () => Navigator.pop(context, true),
            child: const Text('Confirm'),
        ),
    ],
),
);
}

```

```

void markAttendace() async {
    bool present = await PresentAbsentDialog().presentOrAbsentDialog(context);
    var userId = FirebaseAuth.instance.currentUser!.uid;
    await AttendanceMarking().markingAttendance(userId, present);
}

```

```

int selectedIndex = 0;

```

```

List<Widget> navigationPages = [
    ShowOrderList(
        currentUserRoll: 'RSM',
    ),

```

```

ShowFeedback(),
Center(
  child: EmployeeActivity(
    employeeRoll: 'ASM',
  ),
),
Center(
  child: ASOMonthlyPerformance(),
),
];

```

```

void navigateBottomBar(int index) {
  setState(() {
    selectedIndex = index;
  });
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text("RSM"),
      backgroundColor: Colors.lightBlue,
      actions: [
        IconButton(
          onPressed: signOutUser,
          icon: const Icon(Icons.logout_rounded),

```

```

    )
  ],
),
body: navigationPages[selectedIndex],
bottomNavigationBar: ClipRRect(
  borderRadius: const BorderRadius.all(Radius.circular(30.0)),
  child: Container(
    color: const Color.fromARGB(255, 166, 225, 255),
    child: Padding(
      padding: const EdgeInsets.all(8.0),
      child: GNav(
        onTabChange: (index) => navigateBottomBar(index),
        backgroundColor: const Color.fromARGB(255, 166, 225, 255),
        gap: 8,
        // color: Colors.green,
        activeColor: Colors.white,
        hoverColor: Colors.white,
        tabBackgroundColor: Colors.lightBlue,
        padding: const EdgeInsets.all(16),
        tabs: const [
          GButton(
            icon: Icons.home,
            text: 'Home',
          ),
          GButton(
            icon: Icons.thumb_up,
            text: 'Feedback',
          ),

```

```

        GButton(
          icon: Icons.group,
          text: 'ASM activity',
        ),
        GButton(
          icon: Icons.bar_chart,
          text: 'Statistics',
        ),
      ],
    ),
  ),
),
drawer: Drawer(
  backgroundColor: Colors.lightBlue.shade100,
  child: Padding(
    padding: const EdgeInsets.only(top: 25.0),
    child: ListView(
      children: [
        ListTile(
          onTap: markAttendance,
          title: const Text(
            "Attendance",
            style: TextStyle(fontWeight: FontWeight.bold),
          ),
          subtitle: const Text("Take attendance for today"),
          leading: const Icon(Icons.calendar_today_rounded),
        ),

```

```

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

```

asm_page.dart

```

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_nav_bar/google_nav_bar.dart';
import 'package:salesforce/components/performance.dart';
import 'package:salesforce/components/present_absent_dialog.dart';
import 'package:salesforce/screens/aso_monthly_perfomence.dart';
import 'package:salesforce/screens/employee_activity.dart';
import 'package:salesforce/screens/login_page.dart';
import 'package:salesforce/screens/show_feedback.dart';
import 'package:salesforce/screens/show_order_list.dart';
import 'package:salesforce/services/apply_leave.dart';
import 'package:salesforce/services/attendance_marking.dart';

```

```

class ASM extends StatefulWidget {
  const ASM({super.key});

  @override

```

```

State<ASM> createState() => _ASMState();
}

class _ASMState extends State<ASM> {
  int selectedIndex = 0;

  List<Widget> navigationPages = [
    Center(
      child: ShowOrderList(
        currentUserRoll: 'ASM',
      ),
    ),
    ShowFeedback(),
    Center(child: EmployeeActivity(employeeRoll: 'ASM')),
    Center(child: EmployeeActivity(employeeRoll: 'ASO')),
  ];

  void navigateBottomBar(int index) {
    setState() {
      selectedIndex = index;
    });
  }

  void signOutUser() async {
    // debugPrint('Enter function');

    // If user press confirm then logout

```

```

if (await _showDialogWindow() == true) {
  await FirebaseAuth.instance.signOut();

  Navigator.pushReplacement(
    context, MaterialPageRoute(builder: (context) => const LoginPage()));
}
}

```

```

Future<bool> _showDialogWindow() async {
  return await showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text(
        'Logout',
      ),
      backgroundColor: Colors.lightBlue.shade100,
      content: const Text(
        'Are you sure want to log out salesforce?',
        style: TextStyle(fontSize: 15),
      ),
      actions: [
        TextButton(
          onPressed: () => Navigator.pop(context, false),
          child: const Text('Cancel'),
        ),
        TextButton(
          onPressed: () => Navigator.pop(context, true),

```

```

        child: const Text('Confirm'),
      ),
    ],
  ),
);
}

```

```

void markAttendance() async {
  bool present = await PresentAbsentDialog().presentOrAbsentDialog(context);
  var userId = FirebaseAuth.instance.currentUser!.uid;
  await AttendanceMarking().markingAttendance(userId, present);
}

```

```
@override
```

```

Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text("ASM"),
      backgroundColor: Colors.lightBlue,
      actions: [
        IconButton(
          onPressed: signOutUser,
          icon: const Icon(Icons.logout_rounded),
        ),
      ],
    ),
    body: navigationPages[selectedIndex],
  ),

```



```

bottomNavigationBar: ClipRRect(
  borderRadius: const BorderRadius.all(Radius.circular(30.0)),
  child: Container(
    color: const Color.fromARGB(255, 166, 225, 255),
    child: Padding(
      padding: const EdgeInsets.all(8.0),
      child: GNav(
        onTabChange: (index) => navigateBottomBar(index),
        backgroundColor: const Color.fromARGB(255, 166, 225, 255),
        gap: 8,
        // color: Colors.green,
        activeColor: Colors.white,
        hoverColor: Colors.white,
        tabBackgroundColor: Colors.lightBlue,
        padding: const EdgeInsets.all(16),
        tabs: const [
          GButton(
            icon: Icons.home,
            text: 'Home',
          ),
          GButton(
            icon: Icons.thumb_up,
            text: 'Feedback',
          ),
          GButton(
            icon: Icons.work,
            text: 'My Activity',
          ),

```

```

        GButton(
          icon: Icons.group,
          text: 'ASO Activity',
        ),
      ],
    ),
  ),
),
),
drawer: Drawer(
  backgroundColor: Colors.lightBlue.shade100,
  child: Padding(
    padding: const EdgeInsets.only(top: 25.0),
    child: ListView(
      children: [
        ListTile(
          onTap: markAttendace,
          title: const Text(
            "Attendance",
            style: TextStyle(fontWeight: FontWeight.bold),
          ),
          subtitle: const Text("Take attendance for today"),
          leading: const Icon(Icons.calendar_today_rounded),
        ),
        ListTile(
          onTap: () {
            showDialog(
              context: context,

```

```

builder: (BuildContext context) {
  return const LeaveDialog(
    employeeRoll: 'ASM',
  );
},
);
},
title: const Text(
  "Apply Leave",
  style: TextStyle(fontWeight: FontWeight.bold),
),
subtitle: const Text("Submit Leave Request"),
leading: const Icon(Icons.access_time_filled_outlined),
),
ListTile(
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) => ASOMonthlyPerformance());
      ),
    title: const Text(
      "ASO Performance",
      style: TextStyle(fontWeight: FontWeight.bold),
    ),
    subtitle: const Text("Statistics"),
    leading: const Icon(Icons.bar_chart),
  ),

```

```

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

```

aso_page.dart

```

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_nav_bar/google_nav_bar.dart';
import 'package:salesforce/screens/employee_activity.dart';
import 'package:salesforce/screens/show_feedback.dart';
import 'package:salesforce/screens/show_order_list.dart';
import 'package:salesforce/screens/book_order.dart';
import 'package:salesforce/components/present_absent_dialog.dart';
import 'package:salesforce/screens/login_page.dart';
import 'package:salesforce/services/apply_leave.dart';
import 'package:salesforce/services/attendance_marking.dart';

class ASO extends StatefulWidget {
  const ASO({super.key});

  @override
  State<ASO> createState() => _ASOState();

```

```
}
```

```
class _ASOState extends State<ASO> {
```

```
  int selectedIndex = 0;
```

```
  List<Widget> navigationPages = [
```

```
    Center(child: ShowOrderList(currentUserRoll: 'ASO')),
```

```
    ShowFeedback(),
```

```
    const Center(child: BookNewOrder()),
```

```
    Center(child: EmployeeActivity(employeeRoll: 'ASO')),
```

```
  ];
```

```
  void navigateBottomBar(int index) {
```

```
    setState(() {
```

```
      selectedIndex = index;
```

```
    });
```

```
  }
```

```
  void signOutUser() async {
```

```
    // If user press confirm then logout
```

```
    if (await _showDialogWindow() == true) {
```

```
      await FirebaseAuth.instance.signOut();
```

```
      Navigator.pushReplacement(
```

```
        context, MaterialPageRoute(builder: (context) => const LogInPage()));
```

```
    }
```

```
  }
```

```

Future<bool> _showDialogWindow() async {
  return await showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: const Text(
        'Logout',
      ),
      backgroundColor: Colors.lightBlue.shade100,
      content: const Text(
        'Are you sure want to log out salesforce?',
        style: TextStyle(fontSize: 15),
      ),
      actions: [
        TextButton(
          onPressed: () => Navigator.pop(context, false),
          child: const Text('Cancel'),
        ),
        TextButton(
          onPressed: () => Navigator.pop(context, true),
          child: const Text('Confirm'),
        ),
      ],
    ),
  );
}

void markAttendace() async {
  bool present = await PresentAbsentDialog().presentOrAbsentDialog(context);

```

```

var userUid = FirebaseAuth.instance.currentUser!.uid;

await AttendanceMarking().markingAttendance(userUid, present);
}

```

```

@override

```

```

Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text("ASO"),
      backgroundColor: Colors.lightBlue,
      actions: [
        IconButton(
          onPressed: signOutUser,
          icon: const Icon(Icons.logout_rounded),
        )
      ],
    ),
    body: navigationPages[selectedIndex],
    bottomNavigationBar: ClipRRect(
      borderRadius: const BorderRadius.all(Radius.circular(30.0)),
      child: Container(
        color: const Color.fromARGB(255, 166, 225, 255),
        child: Padding(
          padding: const EdgeInsets.all(8.0),
          child: GNav(
            onTabChange: (index) => navigateBottomBar(index),
            backgroundColor: const Color.fromARGB(255, 166, 225, 255),
            gap: 8,

```

```

// color: Colors.green,
activeColor: Colors.white,
hoverColor: Colors.white,
tabBackgroundColor: Colors.lightBlue,
padding: const EdgeInsets.all(16),
tabs: const [
  GButton(
    icon: Icons.home,
    text: 'Home',
  ),
  GButton(
    icon: Icons.thumb_up,
    text: 'Feedback',
  ),
  GButton(
    icon: Icons.assignment,
    text: 'Book Order',
  ),
  GButton(
    icon: Icons.work_history,
    text: 'Activity',
  ),
],
),
),
),
),
),
drawer: Drawer(

```



```

backgroundColor: Colors.lightBlue.shade100,
child: Padding(
  padding: const EdgeInsets.only(top: 25.0),
  child: ListView(
    children: [
      ListTile(
        onTap: markAttendace,
        title: const Text(
          "Attendance",
          style: TextStyle(fontWeight: FontWeight.bold),
        ),
        subtitle: const Text("Take attendance for today"),
        leading: const Icon(Icons.calendar_today_rounded),
      ),
      ListTile(
        onTap: () {
          showDialog(
            context: context,
            builder: (BuildContext context) {
              return const LeaveDialog(
                employeeRoll: 'ASO',
              );
            },
          );
        },
        title: const Text(
          "Apply Leave",
          style: TextStyle(fontWeight: FontWeight.bold),

```

```
    ),  
    subtitle: const Text("Submit Leave Request"),  
    leading: const Icon(Icons.access_time_filled_outlined),  
  ),  
],  
,  
,  
,  
,  
);  
}  
}
```

6.2. DATABASE CODEING

READ OPERATION FROM FIREBASE USING DART.

```
final currentUser = FirebaseAuth.instance.currentUser;

final userUID = currentUser!.uid;

final dbRef = FirebaseDatabase.instance.ref().child("Employees");

await dbRef.child(userUID).once().then(
  (event) {
    DataSnapshot snapshot = event.snapshot;
    // pop the loading circle
    Navigator.pop(context);

    setState(
      () {
        Map<Object?, Object?> dataMap =
          snapshot.value as Map<Object?, Object?>;
        String userRole = dataMap['Role'] as String;

        if (userRole == "RSM") {
          Navigator.pushReplacement(context,
            MaterialPageRoute(builder: (context) => const RSM()));
        } else if (userRole == "ASM") {
          Navigator.pushReplacement(context,
            MaterialPageRoute(builder: (context) => const ASM()));
        }
      }
    );
  }
);
```

```

    } else if (userRole == "ASO") {
      Navigator.pushReplacement(context,
        MaterialPageRoute(builder: (context) => const ASO()));
    }
  },
);

```

WRITE OPERATION FROM FIREBASE USING DART.

```

DateTime currentDateTime = DateTime.now();
String currentDate = DateFormat('dd/MM/yyyy').format(currentDateTime);
String currentTiming = DateFormat('HH:mm').format(currentDateTime);
String googleMapLocation;

```

```

if (await CurrentLocation().checkLocationPermission() == true) {
  googleMapLocation = await CurrentLocation().getCurrentLocation();
} else if (await CurrentLocation().requestLocationPermission() == true) {
  googleMapLocation = await CurrentLocation().getCurrentLocation();
} else {
  return;
}

```

```

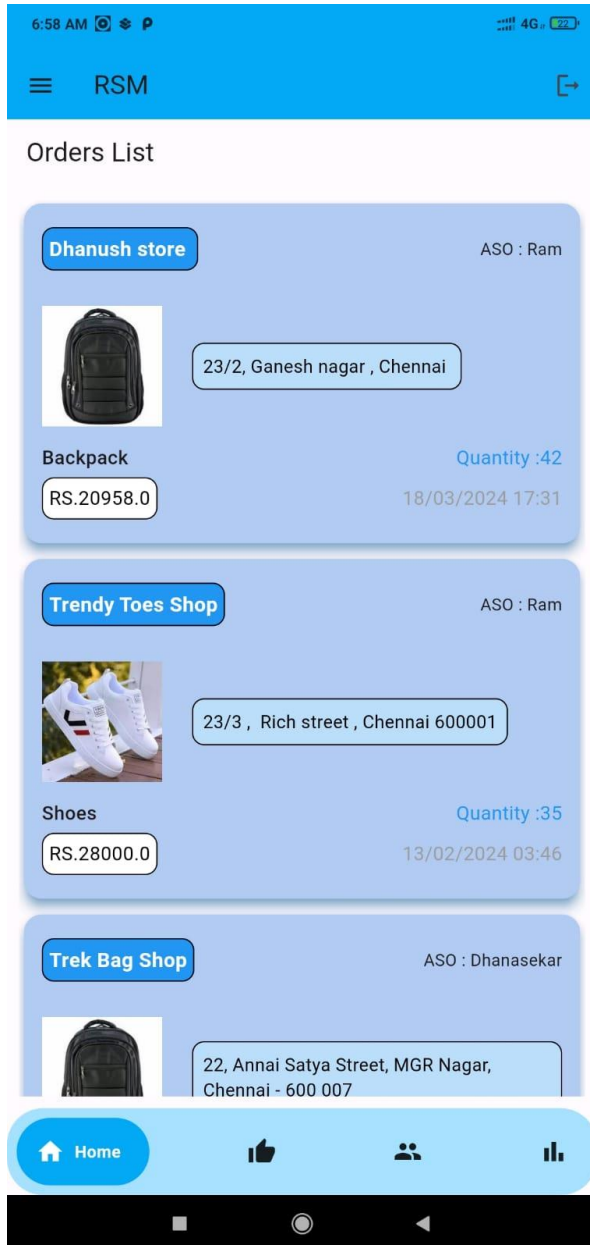
await FirebaseFirestore.instance
  .collection('Attendance')
  .doc(userUid)
  .set(

```

```
{  
  currentDate: {  
    'Present': present,  
    'Location': googleMapLocation,  
    'Timing': currentTiming,  
  }  
},  
SetOptions(merge: true),  
);
```

6.3 MODULE

RSM MODULE



ASM MODULE

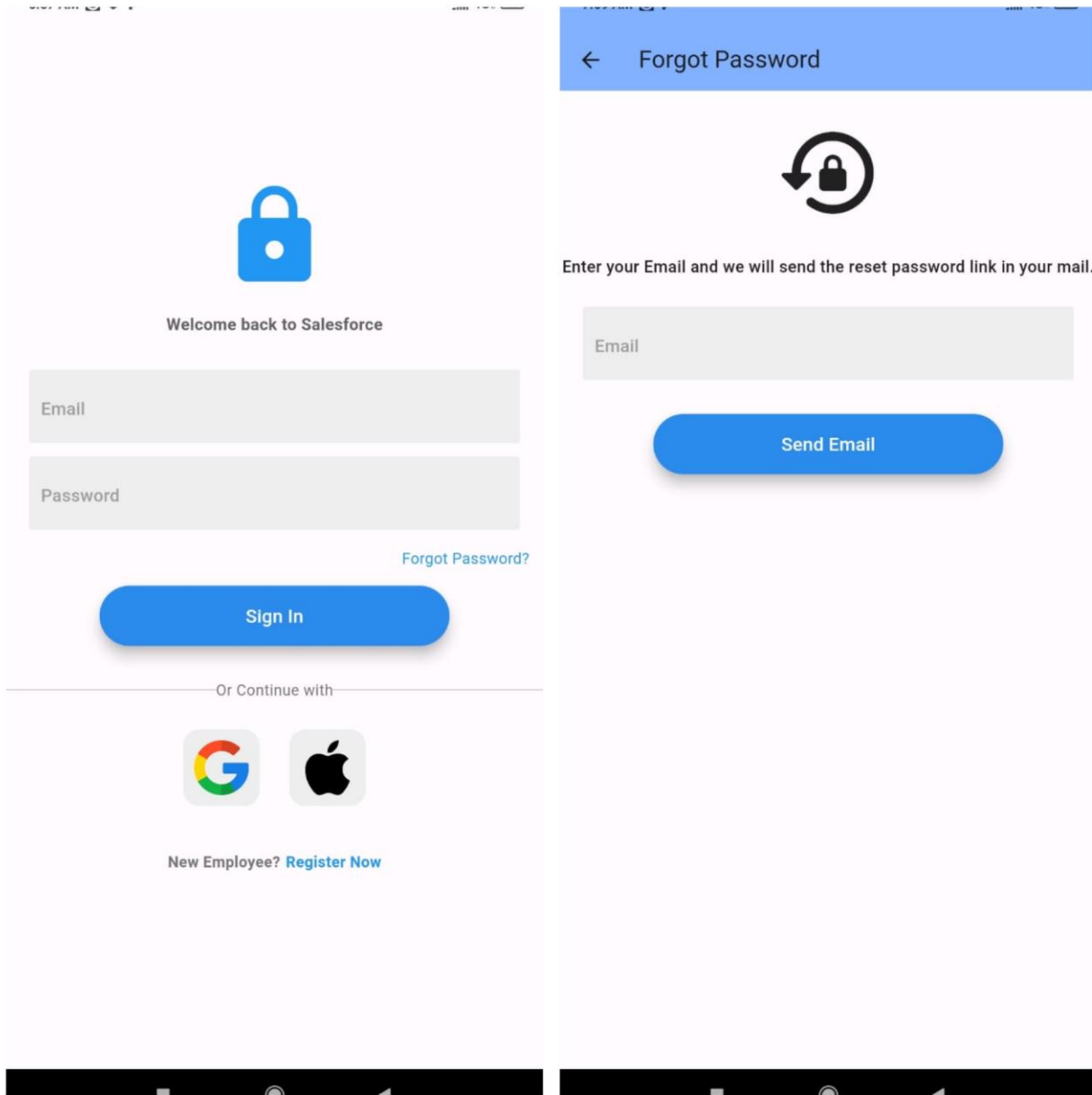


ASO MODULE

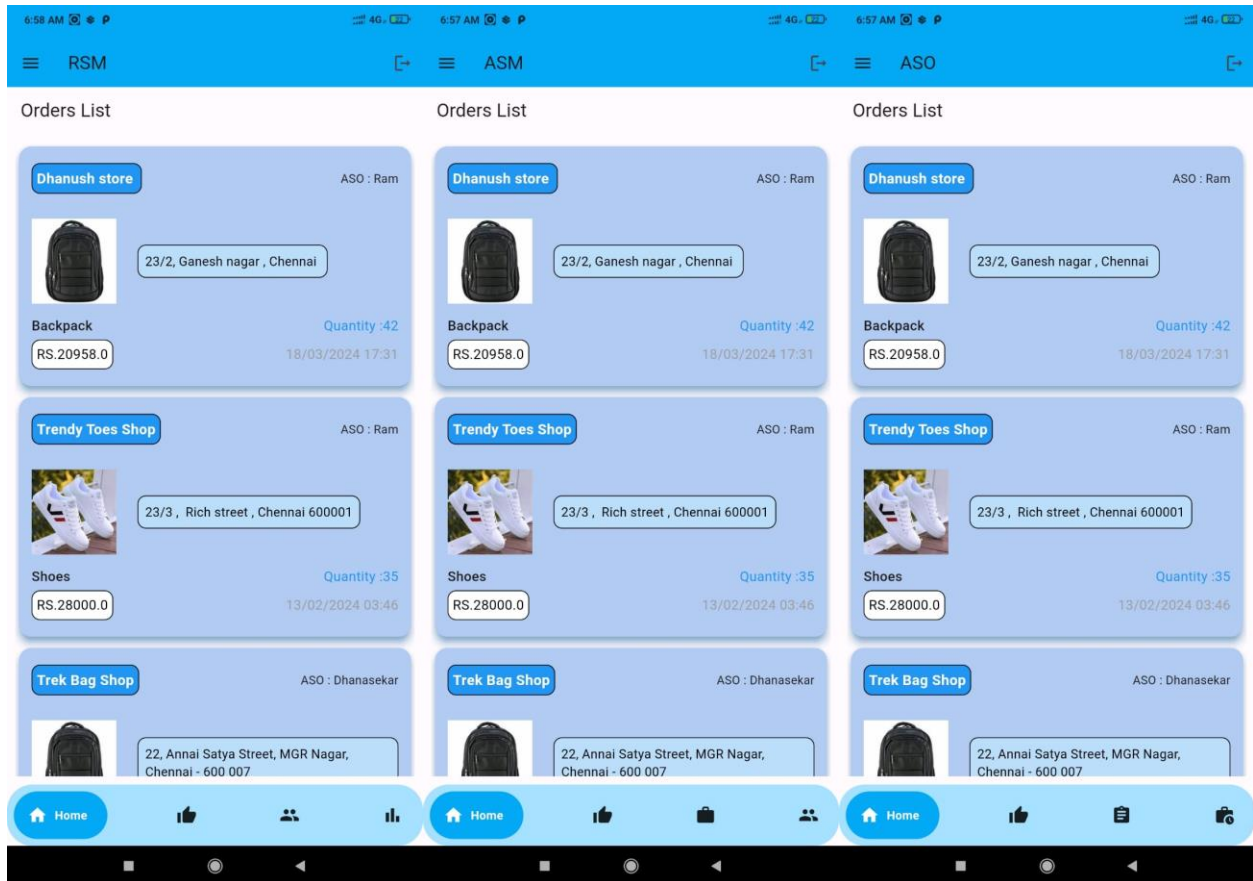


6.4 SCREEN SHOTS

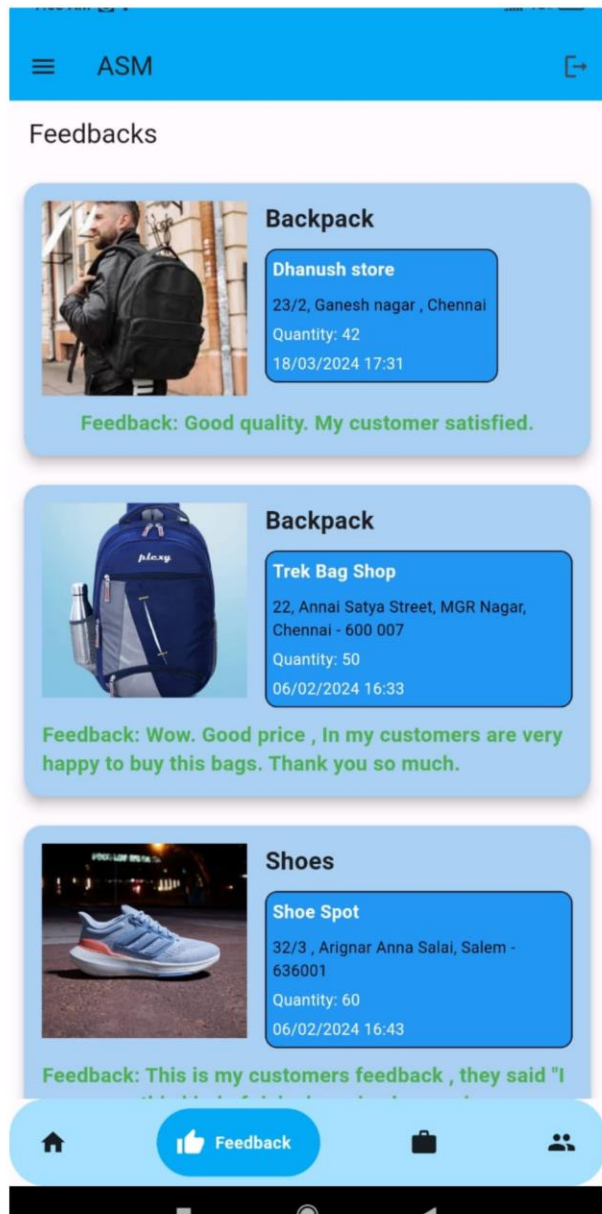
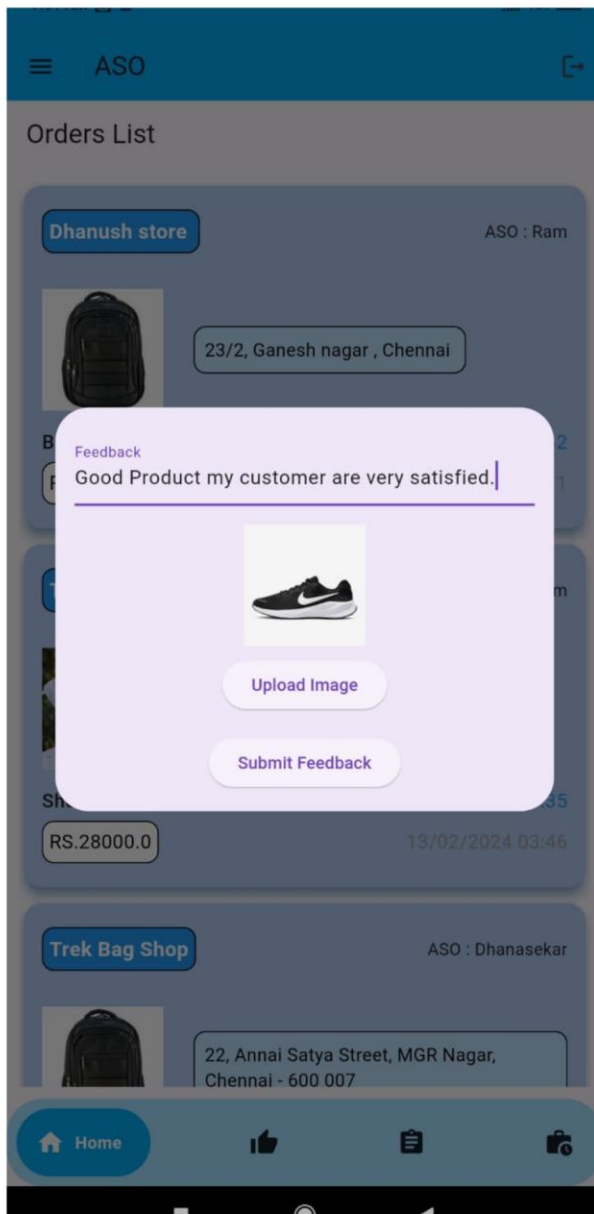
LOGIN PAGE



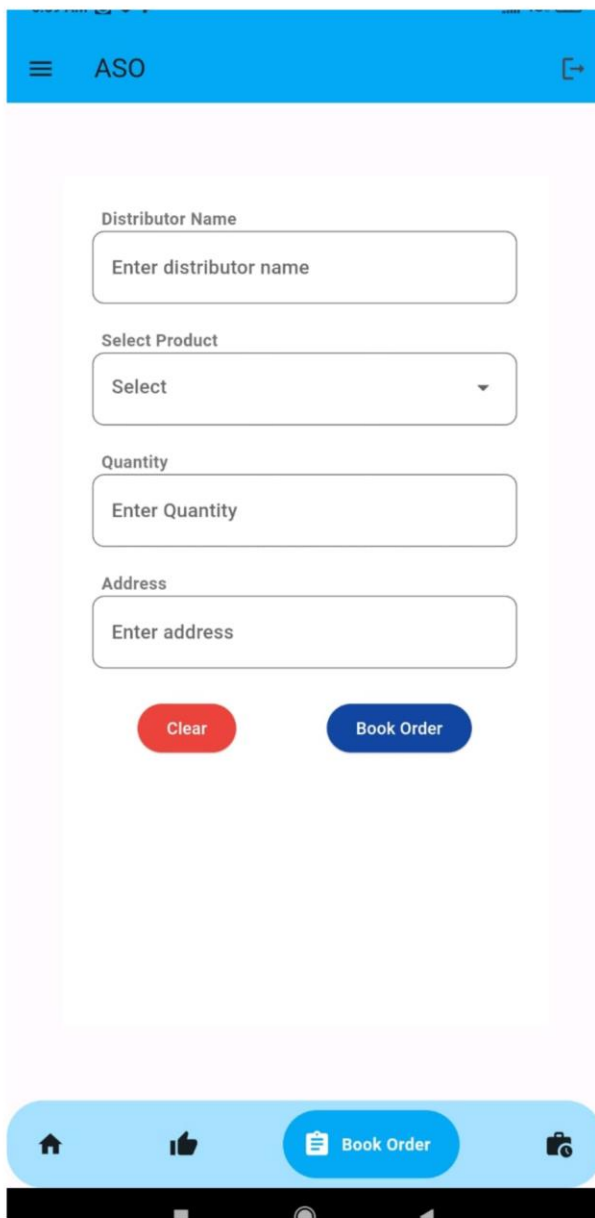
HOME PAGE



FEEDBACK PAGE



ORDER FORM



ASO

Distributor Name

Enter distributor name

Select Product

Select

Quantity

Enter Quantity

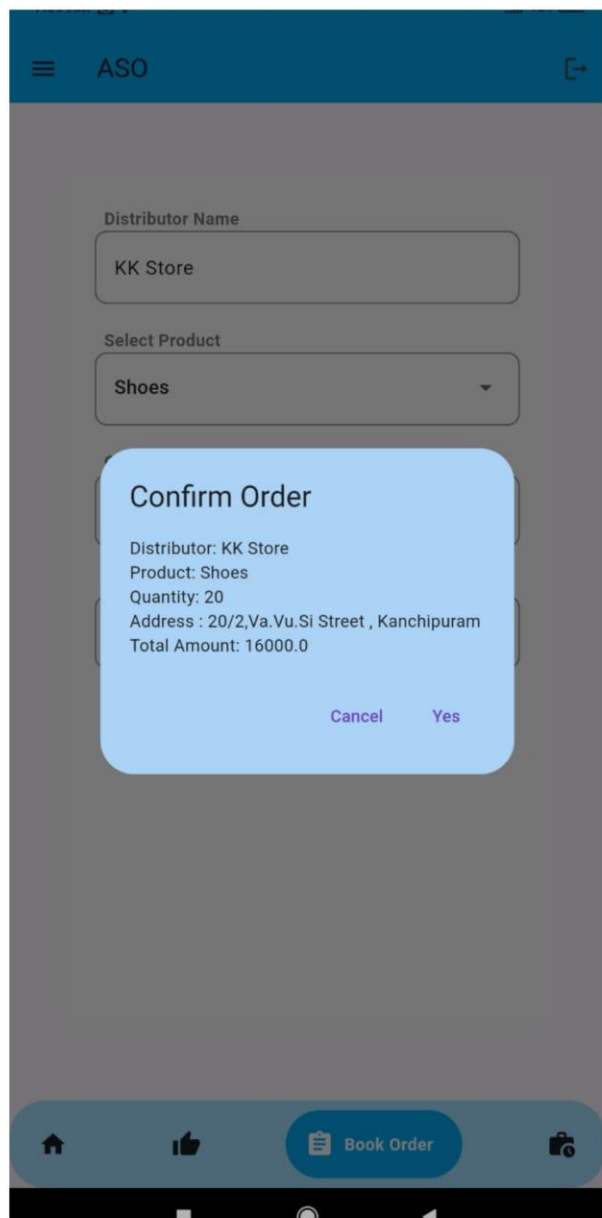
Address

Enter address

Clear Book Order

Home Like Book Order Cart

This screenshot shows the 'ASO' (Add to Order) form. It features a light blue header with a menu icon, the text 'ASO', and a share icon. The form itself is a white card with rounded corners containing four input fields: 'Distributor Name' (with placeholder 'Enter distributor name'), 'Select Product' (a dropdown menu currently showing 'Select'), 'Quantity' (with placeholder 'Enter Quantity'), and 'Address' (with placeholder 'Enter address'). Below these fields are two buttons: a red 'Clear' button and a blue 'Book Order' button. At the bottom of the screen is a navigation bar with icons for Home, Like, Book Order (highlighted with a white background), and Cart.



ASO

Distributor Name

KK Store

Select Product

Shoes

Confirm Order

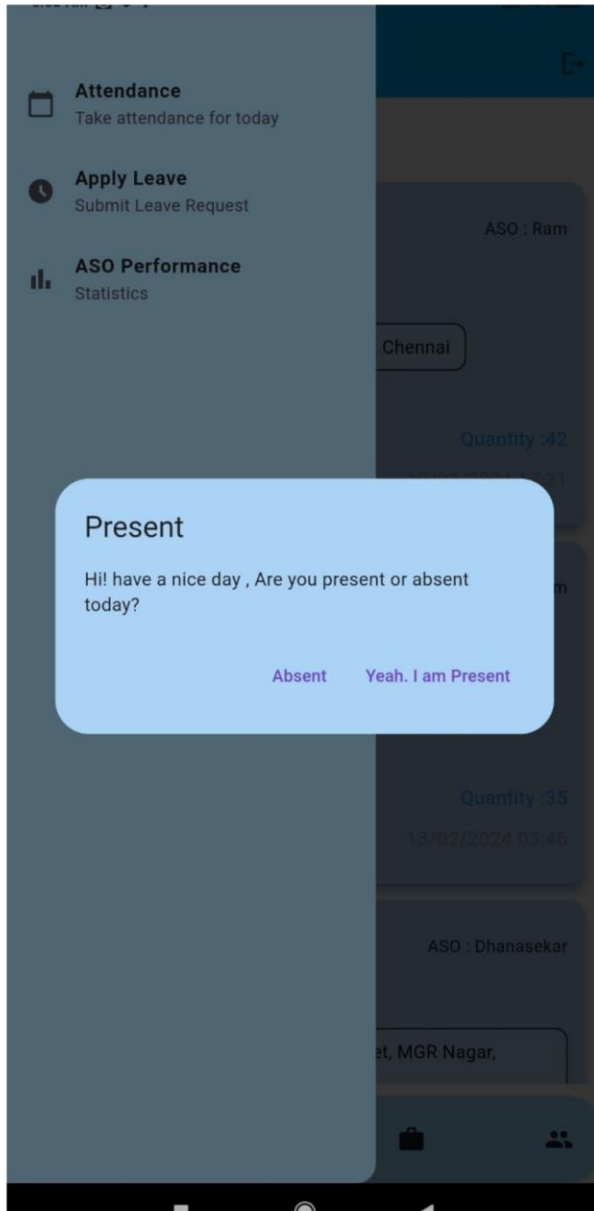
Distributor: KK Store
Product: Shoes
Quantity: 20
Address : 20/2,Va.Vu.Si Street , Kanchipuram
Total Amount: 16000.0

Cancel Yes

Home Like Book Order Cart

This screenshot shows the 'ASO' form with a dark blue header. The 'Distributor Name' field now contains 'KK Store' and the 'Select Product' dropdown shows 'Shoes'. A light blue modal dialog box is overlaid on the form, titled 'Confirm Order'. It displays the order details: 'Distributor: KK Store', 'Product: Shoes', 'Quantity: 20', 'Address : 20/2,Va.Vu.Si Street , Kanchipuram', and 'Total Amount: 16000.0'. At the bottom of the modal are two buttons: 'Cancel' and 'Yes'. The bottom navigation bar is identical to the previous screenshot, with the 'Book Order' button highlighted.

ATTENDANCE WITH CURRENT LOCATION CAPTURE



LEAVE APPLY

7:01 AM 4G

Attendance
Take attendance for today

Apply Leave
Submit Leave Request

ASO : Ram

Chennai

Apply for Leave

Start Date: 2024-04-17

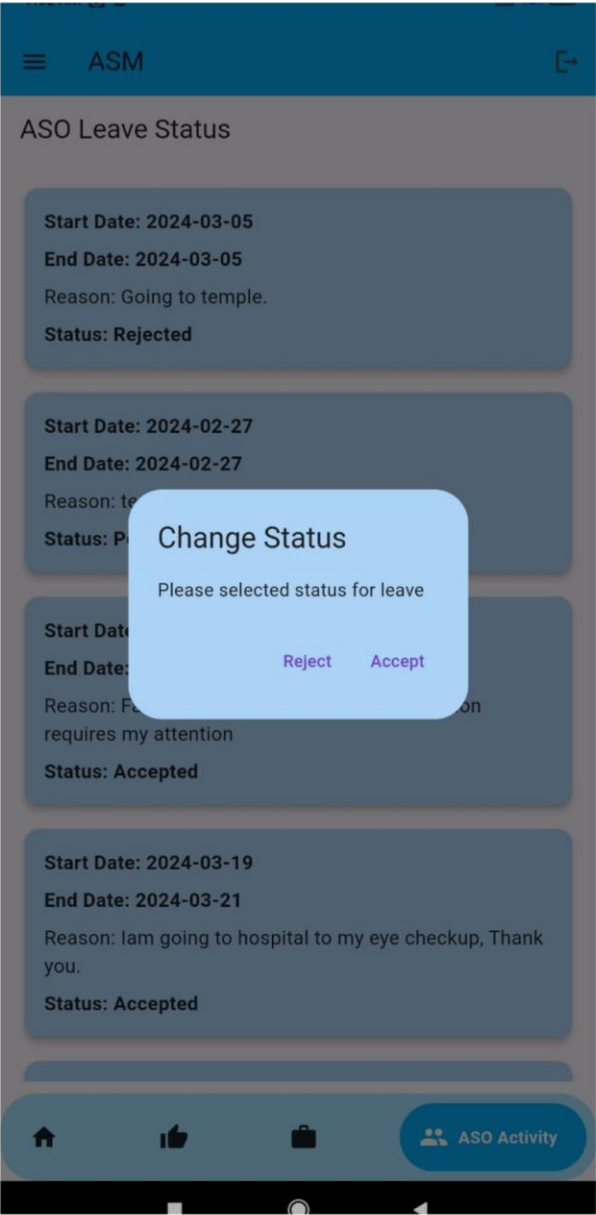
End Date: 2024-04-12

Reason for Leave
Going to hospital.

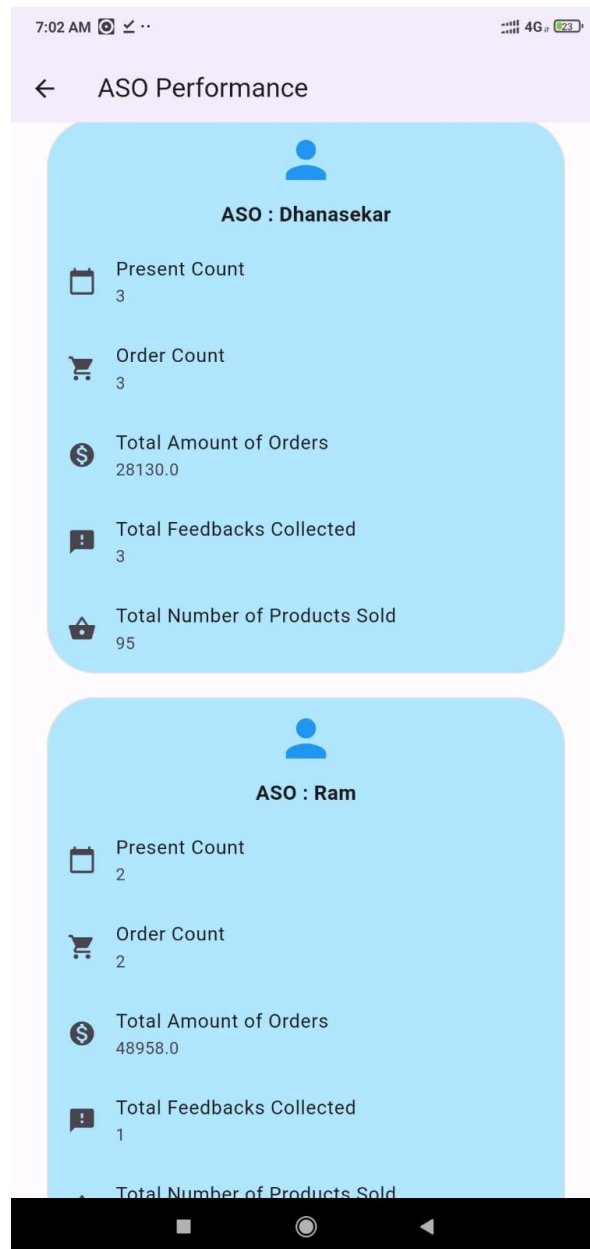
ASO : Dhanasekar

et, MGR Nagar,

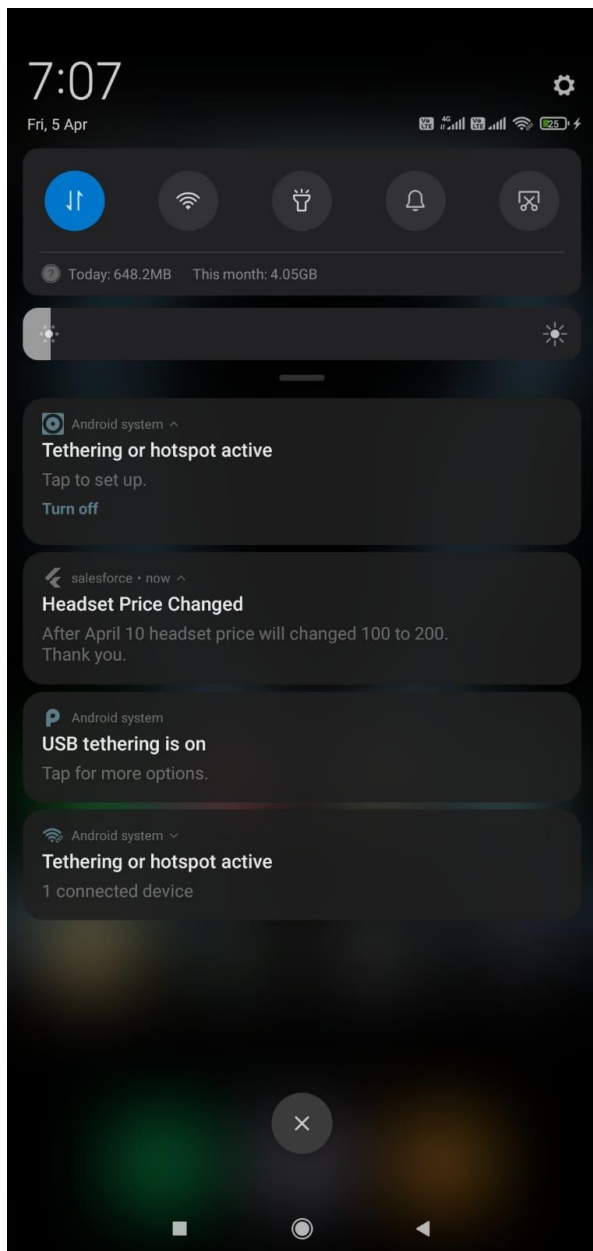
LEAVE APPLY STATUS



ASO PERFORMANCE



NOTIFICATION



7.CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The conclusion section should summarize the key points discussed in the document, highlighting the importance of the app for the manufacturing company's sales management. It should emphasize how the app addresses the specified requirements and contributes to improving the efficiency and effectiveness of the sales process.

7.2 FUTURE SCOPE

The future scope section should discuss potential enhancements or additions to the app that could be implemented in future versions. This could include:

Real-time Analytics: Integration of analytics tools to provide real-time insights into sales performance, distributor interactions, etc.

Integration with CRM Systems: Integration with existing CRM systems to streamline sales and customer management processes.

Support for Additional Regions/Districts: Expansion of the app's coverage to support sales operations in additional regions or districts beyond Tamilnadu, Kerala, Karnataka.

Flutter Scope: In future versions, consider leveraging Flutter for cross-platform development to support both Android and iOS platforms, enabling wider reach and accessibility for users.

7.0 Bibliography

<https://pub.dev/>

<https://docs.flutter.dev/>

<https://stackoverflow.com/>

<https://chat.openai.com/>

<https://gemini.google.com/>

<https://www.youtube.com/@collectivaknowledgeacademy>

<https://www.youtube.com/@createdbykoko>