**GLOBAL ACADEMY OF TECHNOLOGY**

## Department of Computer Science and Engineering

### (Accredited by NBA 2022-2025)

#### Rajarajeshwari Nagar, Bengaluru – 560 098

**2022-2023**

Visvesvaraya Technological University

### JnanaSangama, Belagavi – 590018, Karnataka

**A Mini Project Report on**

### “ReSellify – preowned item selling application”

*Submitted in partial fulfillment of the requirement for*

**Mini Project (20CSEP69) of VI semester**

### Bachelor of Engineering in

**Computer Science and Engineering** *Submitted By*

|  |  |
| --- | --- |
| **Sharath T S** | **(1GA20CS180)** |
| **Shashank P** | **(1GA20CS186)** |

### *Under the Guidance of*

### Prof. R C Ravindranath

### Assistant Professor

### Dept. of CS&E

**GLOBAL ACADEMY OF TECHNOLOGY**

## Department of Computer Science and Engineering

### (Accredited by NBA 2022-2025)

**Rajarajeshwari Nagar, Bengaluru – 560 098**

# CERTIFICATE

This is to certify that the Mini Project entitled **“ReSellify”** carried out by SharathTS **(1GA20CS180),** ShashankP **(1GA20CS186),** as a partial fulfillment for the award of Bachelor’s Degree in Computer Science and Engineering for Mini Project [20CSEP69] as prescribed by **Visvesvaraya Technological University**, **Belagavi** during the year 2022-2023.

|  |  |
| --- | --- |
| **SIGNATURE OF THE GUIDE** | **SIGNATURE OF THE HOD** |
| **Prof. R C Ravindranath** | **Dr. Kumaraswamy S** |
| **Assistant Professor,** | **Professor & Head,** |
| **Dept. of CS&E,** | **Dept. of CS&E,** |
| **GAT, Bengaluru.** | **GAT, Bengaluru.** |

**External Exam**

Name of the Examiner Signature with date

1**.** \_

2**.** \_

**GLOBAL ACADEMY OF TECHNOLOGY**

**Department of Computer Science and Engineering**

### (Accredited by NBA 2019-2022)

**Rajarajeshwari Nagar, Bengaluru – 560 098**

# DECLARATION

We, **Sharath T S**, bearing 1GA20CS180, **Shashank P**, bearing 1GA20CS186, students of Sixth Semester B.E, Department of Computer Science and Engineering, Global Academy of Technology, Rajarajeshwarinagar Bengaluru, declare that the Mini Project entitled “**ReSellify-preowned item selling application**” has been carried out by us and submitted in partial fulfillment of the course requirements for the award of degree in Bachelor of Engineering in Computer Science and Engineering from Visvesvaraya Technological University, Belagavi during the academic year 2022-2023.

#### Sharath T S, 1GA20CS180 ………………..

Shashank P, 1GA20CS186 **………………..**

Place: Bengaluru Date: 22/07/2023

# ABSTRACT

The current online resale market lacks a user-friendly and secure platform that efficiently connects buyers and sellers of pre-owned items. Existing platforms often suffer from cumbersome interfaces, limited search functionality, and a lack of trust and security measures. This leads to a suboptimal experience for users who are looking to sell their used items or find quality secondhand products. Additionally, the absence of a reliable verification system raises concerns about the legitimacy and trustworthiness of users on these platforms. Buyers are hesitant to make purchases without a verified or genuine owner, while sellers face challenges in building credibility and attracting potential customers. Furthermore, the absence of a streamlined and secure payment system adds another layer of complexity to the resale process, hindering seamless transactions and increasing the risk of fraudulent activities. There is a clear need for a mobile application like ReSellify that addresses these challenges by providing a user-friendly interface, secure payment options and robust verification mechanisms. Resellify aims to create a trusted and convenient platform that enhances the overall experience of buying and selling pre-owned items online, this platform pay once the product is received by the customer to reduce deception.

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance crowned our efforts with success.

We consider ourselves proud, to be part of the **Global Academy of Technology** family, the institution which stood by our way in our endeavors.

We express our deep and sincere thanks to our Principal **Dr. N. Ranapratap Reddy** for his support.

We are grateful to **Dr. Kumaraswamy S**, Professor and HOD, Dept of CSE who is the source of inspiration and of invaluable help in channelizing my efforts in the right direction.

We wish to thank our guide **R C Ravindranath,** AssistantProfessor, Dept. of CS&E for guiding and correcting various documents with attention and care. He has taken lot of pain to go through the document and make necessary corrections as and when needed.

We would like to thank the faculty members and supporting staff of the Department of CS&E, GAT for providing all the support for completing the Project work.

Finally, we are grateful to our parents and friends for their unconditional support and help during the course of our Project work.

#### Sharath T S, 1GA20CS180

#### Shashank P, 1GA20CS186

# TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Chapter No.** |  | **Chapter Name** | **Page No.** |
|  |  | **ABSTRACT** | i |
|  |  | LIST OF FIGURES | v |
|  |  | LIST OF TABLES | vi |
|  |  |  |  |
| **1.** |  | **INTRODUCTION** |  |
|  | 1.1 | Motivation | 1 |
|  | 1.2 | **OBJECTIVES OF THE PROJECT** | 1 |
|  | 1.3 | Project Report Outline | 2 |
| **2.** |  | **LITERATURE SURVEY** | 3 |
| **3.** |  | **REQUIREMENT SPECIFICATION** |  |
|  | 3.1 | Software Requirements | 5 |
|  | 3.2 | Hardware Requirements | 6 |
| **4.** |  | **DESIGN** |  |
|  | 4.1 | High Level Design | 7 |
| **5.** |  | **IMPLEMENTATION** |  |
|  | 5.1 | Pubspec.yaml | 10 |
|  | 5.2 | Main.dart | 11 |
|  | 5.3 | Home.dart | 12 |
|  | 5.4 | Product.php | 21 |
| **6.** |  | **TESTING** | 36 |
| **7.** |  | **RESULTS** |  |
|  | 7.1 | Snapshots | 37 |
|  |  | **CONCLUSION** | 40 |
|  |  | **LIMITATIONS** | 41 |
|  |  | **FUTURE ENHANCEMENTS** | 42 |
|  |  | **REFERENCES** | 43 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| 4.1.1 | Sequence diagram for user login | 7 |
| 4.1.2 | Sequence diagram for product list | 7 |
| 4.1.3 | Sequence diagram for product search | 8 |
| 4.1.4 | Sequence diagram for chatting | 8 |
| 4.1.5 | Sequence diagram for product removal | 9 |
| 4.1.6 | Class diagram | 9 |
| 7.1.1 | Login Page | 37 |
| 7.1.2 | OTP Page | 37 |
| 7.1.3 | Welcome Page | 38 |
| 7.1.4 | Home Page | 38 |
| 7.1.5 | Categories Page | 38 |
| 7.1.6 | Chat Page | 39 |
| 7.1.7 | Chatting Page | 39 |
|  |  |  |

**CHAPTER 1**

**INTRODUCTION**

# Motivation

# The motivation for developing the Resellify application for preowned products:

# Resellify is driven by a strong motivation to foster sustainability and promote ethical consumption practices. By offering a platform for buying and selling preowned products, Resellify aims to contribute to a more resource-efficient and environmentally friendly lifestyle. Embracing a circular economy model, the application seeks to extend the lifespan of goods, reduce waste, and counteract the culture of overconsumption. With Resellify, users can save money on purchases, as preowned items are often more cost-effective than new ones. The platform provides a user-friendly and convenient way for individuals to declutter their possessions, make some extra money by selling items they no longer need, and discover good deals on second-hand products. Moreover, Resellify endeavors to create a vibrant community of like-minded buyers and sellers, fostering connections and interactions among its users. As the demand for second-hand goods continues to grow, Resellify also presents an entrepreneurial opportunity to tap into a thriving market, aligning both financial success and social impact.

# Objectives of the project

# The objective of ReSellify is to provide a user-friendly and secure mobile application that serves as a trusted marketplace for buying and selling pre-owned items. The key objectives of ReSellify include:

# Convenience and ease-of-use: ReSellify aims to create a seamless and intuitive user experience, allowing users to easily list items for sale, browse through available products, and make purchases with minimal effort.

# Trust and security: ReSellify strives to establish a high level of trust and security within the platform. This includes implementing a robust verification system to authenticate users, ensuring secure payment cash on delivery options for buyers and sellers to build trust.

# Wide range of categories and items: ReSellify aims to cater to diverse consumer needs by offering a wide range of categories and items available for resale. From fashion and electronics to home appliances and more, the platform seeks to accommodate various product types to meet user demands.

# Cost-effectiveness and sustainability: ReSellify encourages sustainable consumption by promoting the resale of pre-owned items, allowing users to save money and reduce waste. The objective is to create a platform where individuals can find quality secondhand products at affordable prices.

# By achieving these objectives, ReSellify intends to become the go-to mobile application for individuals looking to buy or sell pre-owned items, offering a reliable and convenient platform that meets the needs of both buyers and sellers.

# Literature Survey

# Requirement Specification

# Software Requirements

# Hardware Requirements

# Literature Survey

# Design

# High Level Design

# Implementation

# Testing

# Result

**CHAPTER 2**

**LITERATURE SURVEY**

It is essential to note that the above methodology is a general outline inferred from the given information. For a comprehensive understanding of the actual methodology used in the study, referring to the original research paper or Rise of Mobile Commerce: The increasing popularity of mobile devices has led to a surge in mobile commerce. Consumers are now using their smartphones and tablets to shop online, making mobile optimization crucial for businesses. Companies need to ensure their websites and payment gateways are mobile-friendly to capitalize on this growing trend.

Shift in Consumer Behavior: Mobile devices have transformed consumer behavior. People now spend more time on mobile apps and websites than ever before. Businesses must adapt their marketing strategies to target mobile users effectively and provide a seamless mobile experience to attract and retain customers

* Research Design: The study likely employs a descriptive research design to understand and describe Generation Z consumers' attitudes and behaviors towards second-hand purchases.
* Data Collection: Data collection methods may include surveys, interviews, or focus group discussions to gather information directly from Generation Z consumers. Surveys could be administered online to reach a broader sample.
* Sampling Technique: The study may use a purposive or convenience sampling technique to target Generation Z consumers, particularly those who have been buying second-hand products for at least three years.
* Variables Studied: The research explores various factors influencing Generation Z's decision to buy second-hand products. These factors include low prices, the desire to appear unique, nostalgia, and trust.
* Analysis: Statistical analysis and qualitative thematic analysis may be used to analyze the collected data. The study may identify correlations between Generation Z's purchase intentions and the determining factors mentioned in the research.
* Limitations: The study might acknowledge potential limitations, such as sample size, generalizability, and possible biases in self-reported data.
* Conclusion: The findings of the study likely conclude that Generation Z's interest in buying second-hand products, particularly clothes, is influenced by factors like low prices, a desire to appear unique, nostalgia, and trust in the online purchase process. publication is recommended.

In conclusion, the mobile market's rapid evolution has had a profound impact on businesses, necessitating their adaptation to stay relevant and competitive. Embracing mobile technologies, leveraging data insights, and employing innovative marketing approaches are crucial for businesses looking to thrive in the ever-changing mobile-driven world. To understand the current state of the used product marketplace and the problems faced, such as deceptive marketing and deceptive users who can mislead this system, leading to financial losses.

**CHAPTER 3**

**REQUIREMENT SPECIFICATION**

**3.1 Software Requirement**

* Flutter SDK: Install the Flutter software development kit (SDK) on your development machine. Flutter provides tools for building native Android and iOS applications from a single codebase.
* Android Studio: Depending on your target platforms, you may need to install Android Studio for Android development
* XAMPP: Install XAMPP, which includes Apache, MySQL, PHP, and other necessary components for setting up a local development server.
* MySQL: Configure MySQL to work with XAMPP. You will use MySQL as the database management system to store data related to user accounts, product listings, and other application information.
* PHP: PHP will be used to create server-side scripts to handle various functionalities, such as user authentication, database operations, and handling HTTP requests from the Flutter application.
* Text Editor / Integrated Development Environment (IDE): You can use any text editor or IDE of your choice to write Flutter code and PHP scripts. Popular choices include Visual Studio Code, Android Studio
* Android Emulator: For testing the application on virtual devices, you'll need to set up Android emulators for Android development.
* Device for Physical Testing: To test the application on real devices, you will need an Android device for Android testing.
* Version Control System: You can use Git as a version control system to track changes and collaborate with other developers.
* Firebase: To implement phone number of user authentication

**3.2 Hardware Requirement**

The hardware requirements for running and testing the ReSellify application would depend on two main aspects: the development environment and the target deployment platforms.

**3.2.1 Development Environment:**

* Processor: A multi-core processor (e.g., Intel Core i5 or higher) for faster code compilation and smoother development.
* RAM: At least 8 GB of RAM to handle the memory requirements of running development tools, emulators, and multiple browser tabs.
* Storage: Sufficient free storage space (at least 20 GB) for installing the development tools, SDKs, and project files.
* Graphics Card: A dedicated graphics card is not necessary for Flutter development, but it can provide a better overall experience if you plan to work on graphics-intensive applications.

**3.2.2 Target Device Platforms:**

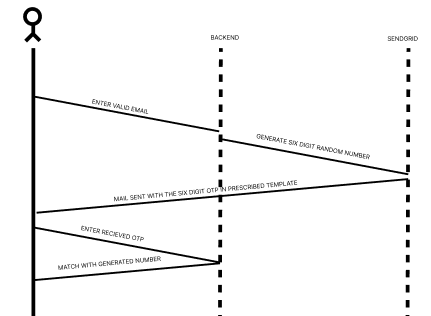
* The hardware requirements for the target deployment platforms (i.e., the devices on which the ReSellify app will run) are determined by the minimum specifications of the platforms you plan to support. For example: Android Devices: The application should run on various Android devices, so consider the minimum Android version and hardware requirements you want to support (e.g., Android 6.0 and above, different screen resolutions, different processor architectures, etc.).

**CHAPTER 4**

**DESIGN**

**4.1 High-Level Design**

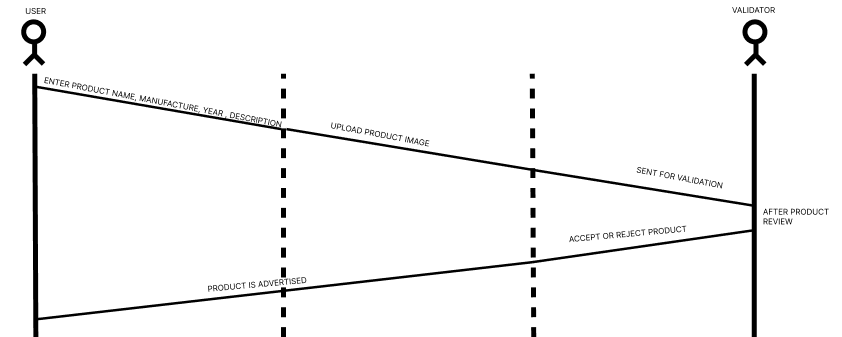
* **USER LOGIN**

****

**Fig 4.1. Sequence diagram for user login**

When a user attempts to log in for the first time, they are required to provide their email. The backend generates a random number and adds it to the email template, which is sent to SendGrid. The user receives a one-time password via email and enters it during authentication. If the entered password matches the sent random number, authentication is successful. For new users, a welcome page prompts them to fill in basic details about themselves. If the user has already logged in previously, they will be redirected to the home page.

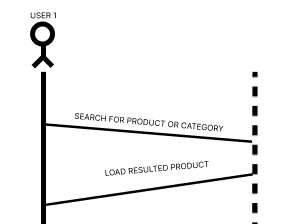
* **PRODUCT LISTING**

****

**Fig 4.2 Sequence-diagram for product listing**

When a user attempts to advertise or sell a product in the current session, they must enter required form fields that meet specific patterns. Once the information is entered, the user proceeds to a new page to upload images of the product and submit it. A validator then checks the entered information and product details. If the validator is satisfied with the product, it is accepted. Otherwise, the product may be rejected.

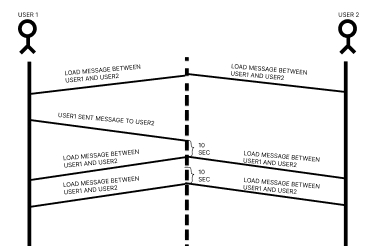
* **SEARCH FOR PRODUCT**

****

**Fig 4.3. Sequence diagram for product searching**

When a user searches for the product-by-product name, product description or product category the SQL query is generated and the results of search will be listed.

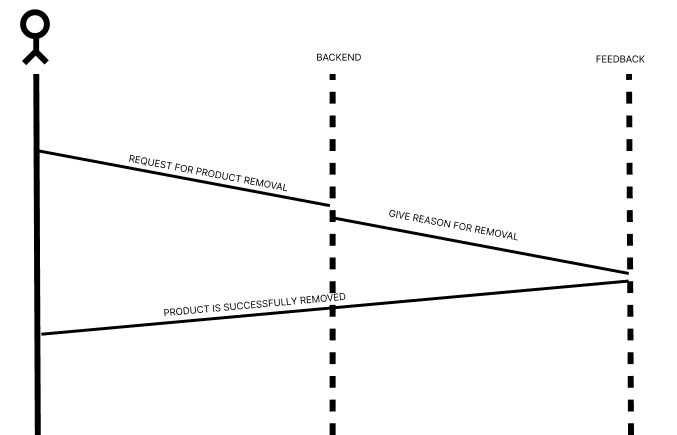
* **IN-APP CHATTING**

****

**Fig 4.4. Sequence diagram for chatting**

Every time a user enters the chatroom, the data from previous chats will be fetched, and this process is repeated every ten seconds. If a sender posts a message, the receiver will see the message after a ten-second delay. This ensures real-time updates and synchronization between users in the chatroom.

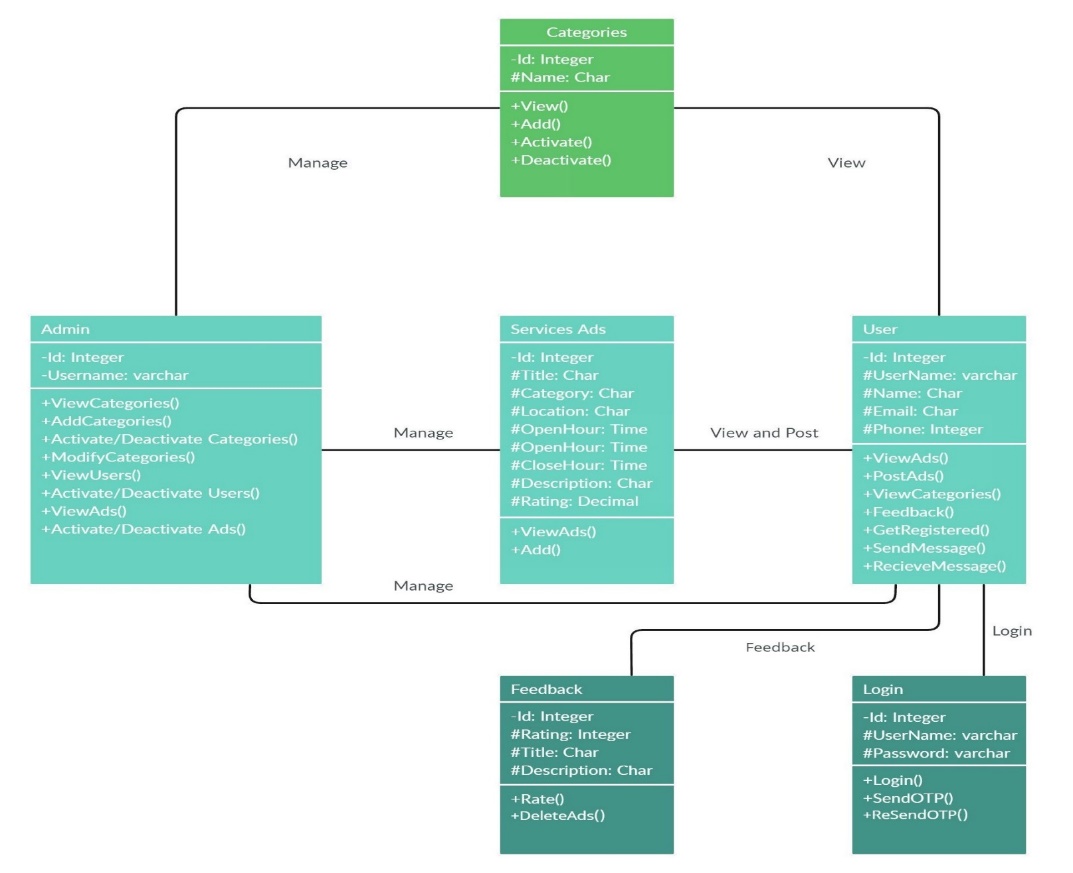
* **REMOVAL OF PRODUCT**

****

**Fig 4.5. Sequence diagram for product removal**

If the user wants to remove his product, then he need to enter the feedback and the reason for removal will be declared and the product will be deleted. This feedback system allows for transparency and helps in understanding the reasons for product removal.

* **CLASS DIAGRAM**

****

**Fig 4.6. Sequence diagram for class diagram**

This shows entire projects as class diagram with login, feedback, admin, user, categories, services ads as the classes with there’s attributes mentioned below ‘-‘being private,’#’ being protected and at last there methods with ‘+’ saying it as public.

**CHAPTER 5**

**IMPLEMENTATION**

* 1. **Pubspec.yaml**

name: resellify

description: A new Flutter project.

version: 1.0.0+1

environment:

sdk: '>=2.19.6 <3.0.0'

dependencies:

flutter:

sdk: flutter

cupertino\_icons: ^1.0.2

firebase\_core: ^2.14.0

provider: ^6.0.5

carousel\_slider: ^4.2.1

image\_picker\_plus: ^0.5.2

dotted\_border: ^2.0.0+3

pinput: ^2.2.31

cloud\_firestore: ^4.8.2

firebase\_storage: ^11.2.3

firebase\_auth: ^4.6.3

fluttertoast: ^8.2.2

lottie: ^2.4.0

sql\_conn: ^0.0.3

http: ^1.1.0

shared\_preferences: ^2.2.0

timeago: ^3.5.0

dev\_dependencies:

flutter\_test:

sdk: flutter

. flutter\_lints: ^2.0.0

icons\_launcher: ^2.1.3

flutter:

assets:

- images/logo.jpeg

- images/resellify.jpeg

- images/resellify12.jpeg

- images/resellify2.jpeg

- images/bike.jpg

- images/car.jpeg

- images/fridge.jpeg

- images/fridge2.jpeg

- images/laptop.jpeg

- images/television.jpeg

- images/

* 1. **Main**

import "dart:js";

import 'package:flutter/material.dart';

import "package:provider/provider.dart";

import "package:resellify/SelectedCategoryDetailsForm.dart";

import 'package:resellify/ImagePickerApp.dart';

import "package:resellify/validator\_form.dart";

import "./UserDetail.dart";

import "./categorydisp.dart";

import "./chathist.dart";

import "./formCategory.dart";

// import "formData.dart";

import "./home.dart";

import "./product\_detail.dart";

import "./search.dart";

import "./login.dart";

import "./pinputdemo.dart";

import "./search\_results.dart";

import "./shoppingcart.dart";

import "./welcome.dart";

// ignore: unused\_import

import "Datafetch.dart";

import "chat.dart";

import "dataStore/basic.dart";

import "./Category.dart";

import "./splashscreen.dart";

import "location.dart";

import "myProduct.dart";

import "otpPage.dart";

import "validator.dart";

void main() {

runApp(const MyApp());

}

class MyApp extends StatefulWidget {

const MyApp({Key? key}) : super(key: key);

@override

State<MyApp> createState() => \_MyApp();

}

class \_MyApp extends State<MyApp> {

// const \_MyAppState({super.key});

void initState() {

super.initState();

print("===========refresh=================");

}

@override

Widget build(BuildContext context) {

return ChangeNotifierProvider(

create: (context) => BasicModel(),

child: MaterialApp(

debugShowCheckedModeBanner: false,

home: Builder(

builder: (BuildContext context) {

// return SplashScreen();

return MyApp1();

},

),

routes: {

'/ImagePickerApp': (context) => ImagePickerApp(),

'/welcomepage': (context) => welcome(),

'/splash2page': (context) => splash2(),

'/home': (context) => MyApp1(),

'/details': (context) => product(),

'/selling': (context) => CategoryCard(),

'/categoryForm': (context) => FormCategory(),

'/newChat': (context) => DataFetchingScreen(),

'/user': (context) => userDetails(),

"/search": (context) => search(),

"/result": (context) => result(),

"/cat": (context) => categorydisp(),

"/cart": (context) => myCarts(),

"/pin": (context) => MyVerify(),

"/VerifyOtpByEmail": (context) => VerifyOtpByEmail(),

"/lo": (context) => login(),

"/selectedCategoryDetailsForm": (context) =>

SelectedCategoryDetailsForm(),

"/LocationForm": (context) => LocationForm(),

"/MyProduct": (context) => myproduct(),

"/validator": (context) => validator(),

"/myValidation": (context) => const Vform(),

},

),

);

}

}

* 1. **Home**

import 'dart:convert';

import 'package:carousel\_slider/carousel\_slider.dart';

import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;

import 'package:resellify/dataStore/handelLocalStorageData.dart';

import 'package:resellify/product\_detail.dart';

import 'otpPage.dart';

final List<String> imgList = [

"http://localhost/api/phpBackEnd/src/uploads/carosuel pic1.png",

"http://localhost/api/phpBackEnd/src/uploads/carosuel pic2.png",

];

final List<String> imgLabel = ['', '', '', '', '', ''];

final List<Widget> imageSliders = imgList

.map((item) => Container(

child: Container(

margin: EdgeInsets.all(5.0),

child: ClipRRect(

borderRadius: BorderRadius.all(Radius.circular(5.0)),

child: Stack(

children: <Widget>[

Image.network(

item,

fit: BoxFit.fill,

width: double.infinity,

height: 200.0,

),

Positioned(

bottom: 0.0,

left: 0.0,

right: 0.0,

child: Container(

padding: EdgeInsets.symmetric(

vertical: 10.0, horizontal: 20.0),

child: Text(

'${imgLabel?[imgList.indexOf(item)]}',

// '${}imgLabel[imgList.indexOf(item)}]',

// ' ${imgList.indexOf(item)} ',

style: TextStyle(

color: Colors.white,

fontSize: 20.0,

fontWeight: FontWeight.bold,

),

),

),

),

],

)),

),

))

?.toList() ??

[];

class MyApp1 extends StatefulWidget {

const MyApp1({Key? key}) : super(key: key);

@override

State<MyApp1> createState() => \_homeState();

}

class \_homeState extends State<MyApp1> {

List<dynamic> productsList = [];

List<dynamic> productImages = [];

Future<void> getData() async {

try {

print("user details ==================================>");

// print(await getLocalFataByKey("userId"));

print(await getLocalData());

print("user details ==================================>");

http.Response result = await http.get(

Uri.parse(

"http://localhost/api/phpBackEnd/src/product.php?route=get-all-products"),

headers: {

'Content-Type': 'application/json',

"Accept": "application/json",

"Access-Control\_Allow\_Origin": "\*"

});

print("got");

dynamic responseMap = jsonDecode(result.body);

print(responseMap["data"]);

// final data = jsonDecode(responseMap);

setState(() {

productsList = responseMap["data"];

// productImages = responseMap["data"]["images"];

});

print("responseMap===================================");

print(await getLocalFataByKey("userId"));

} catch (e) {

print(e);

}

}

@override

void initState() {

// TODO: implement initState

super.initState();

print("VerifyOtpByEmail.id=======================================>");

getData();

}

@override

Widget build(BuildContext context) {

return MaterialApp(

debugShowCheckedModeBanner: false,

home: Scaffold(

body: Container(

color: Color(0xFFDAEAFD),

// height: 400,

child: ListView(

shrinkWrap: true,

children: [

Padding(

padding: EdgeInsets.only(top: 20, left: 10, right: 10),

child: Row(

mainAxisAlignment: MainAxisAlignment.spaceBetween,

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Align(

alignment: Alignment.topLeft,

child: Text(

"ReSellify",

// style: style1,

),

),

Align(

alignment: Alignment.topRight,

child: IconButton(

onPressed: () {

Navigator.pushNamed(context, '/user');

},

icon: Icon(

Icons.account\_circle,

size: 32,

color: Colors.teal,

),

),

),

],

)),

Padding(

padding: EdgeInsets.all(10),

child: GestureDetector(

onDoubleTap: () {

Navigator.pushNamed(context, '/search');

},

child: Container(

color: Color(0xFFD9D9D9),

// decoration: ,

child: Row(children: [

Expanded(

child: TextField(

decoration: InputDecoration(

hintText: ' Search..',

border: InputBorder.none,

fillColor: Colors.white),

),

),

IconButton(

icon:

Icon(Icons.search, color: Color(0xFF140A8C)),

onPressed: () {

Navigator.pushNamed(context, '/search');

},

),

])),

)),

Padding(

padding: EdgeInsets.only(top: 20),

child: Container(

child: CarouselSlider(

options: CarouselOptions(

autoPlay: true,

aspectRatio: 2.0,

enlargeCenterPage: true,

),

items: imageSliders,

),

),

),

Column(

children: [

Padding(

padding: EdgeInsets.only(top: 10, bottom: 10, left: 20),

child: Align(

alignment: Alignment.topLeft,

child: Text('TOP CATEGORY'),

)),

Row(

mainAxisAlignment: MainAxisAlignment.spaceEvenly,

children: [

Column(

children: [

IconButton(

icon: Icon(

Icons.car\_repair,

color: Colors.black54,

),

iconSize: 50,

onPressed: () async {

Navigator.pushNamed(

context, '/result', arguments: {

"searchText": "Car",

"userId": await getLocalFataByKey("userID")

});

},

),

Text(

"Car",

style: TextStyle(

fontSize: 16.0,

fontWeight: FontWeight.w400,

),

)

],

),

Column(

children: [

IconButton(

icon: Icon(

Icons.bike\_scooter,

color: Colors.black54,

),

iconSize: 50,

onPressed: () async {

Navigator.pushNamed(

context, '/result', arguments: {

"searchText": "BIke",

"userId": await getLocalFataByKey("userID")

});

},

),

Text(

"Bike",

style: TextStyle(

fontSize: 16.0,

fontWeight: FontWeight.w400,

),

)

],

),

Column(

children: [

IconButton(

icon: Icon(

Icons.table\_bar,

color: Colors.black54,

),

iconSize: 50,

onPressed: () async {

Navigator.pushNamed(

context, '/result', arguments: {

"searchText": "Furniture",

"userId": await getLocalFataByKey("userID")

});

},

),

Text(

"Furniture",

style: TextStyle(

fontSize: 16.0,

fontWeight: FontWeight.w400,

),

)

],

),

Column(

children: [

IconButton(

icon: Icon(

Icons.book,

color: Colors.black54,

),

iconSize: 50,

onPressed: () async {

Navigator.pushNamed(

context, '/result', arguments: {

"searchText": "others",

"userId": await getLocalFataByKey("userID")

});

},

),

Text(

"Book",

style: TextStyle(

fontSize: 16.0,

fontWeight: FontWeight.w400,

),

)

],

),

],

),

Padding(

padding: EdgeInsets.only(top: 10, bottom: 10, left: 20),

child: Align(

alignment: Alignment.topLeft,

child: Text('NEWLY ADDED'),

)),

],

),

GridView.count(

shrinkWrap: true,

padding: const EdgeInsets.all(20),

crossAxisSpacing: 10,

mainAxisSpacing: 10,

crossAxisCount: 2,

children: productsList.map((pro) {

print(pro["images"].split(",")[0]);

String imageUrl = pro?["images"]?.split(",")?[0] ?? " ";

String productName = pro?['productName']

.substring(

0,

pro?['productName'].length <= 10

? pro['productName'].length : 10)

.toString() ??

"..."; // Provide a default value if productName is null

String price = pro?['price']?.toString() ??

"0"; // Provide a default value if productName is null

return InkWell(

onTap: () async {

Navigator.pushNamed(context, '/details', arguments: {

"productId": pro?["id"],

"userId": await getLocalFataByKey("userId"),

});

},

onDoubleTap: () {},

customBorder: CircleBorder(

eccentricity: BorderSide.strokeAlignOutside,

),

child: Stack(

children: [

Image.network(

imageUrl,

// fit: BoxFit.cover,

fit: BoxFit.fill,

width: double.maxFinite,

height: 300.0,

),

Align(

alignment: Alignment.bottomLeft,

child: Container(

color: Color.fromARGB(255, 231, 248, 255),

child: ListTile(

leading: Column(

mainAxisAlignment:

MainAxisAlignment.spaceEvenly,

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

productName.toUpperCase(),

style: TextStyle(

color: Color.fromARGB(255, 83, 77, 77),

fontSize: 17.0,

fontWeight: FontWeight.w400,

letterSpacing: 1.0,

),

),

Text(

"₹${price?.toString() ?? ' '}",

style: TextStyle(

color: Color.fromARGB(255, 83, 77, 77),

fontSize: 16.0,

),

),

],

),

trailing: Icon(

// Icons.shopping\_cart,

Icons.shopping\_cart\_outlined,

// color: Color.fromARGB(255, 83, 77, 77),

color: Colors.indigo,

), // Replace this with your desired trailing widget

),

),

),

],

),

);

}).toList(),

),

],

),

),

bottomNavigationBar: BottomAppBar(

color: Color(0xFFBFC1DB),

child: Row(

mainAxisAlignment: MainAxisAlignment.spaceAround,

children: [

IconButton(

onPressed: () {},

icon: Icon(

Icons.home,

color: Color(0xFF140A8C),

)),

IconButton(

onPressed: () {

Navigator.pushNamed(context, '/cat');

},

icon: Icon(

Icons.category\_outlined,

color: Color(0xFF140A8C),

)),

IconButton(

onPressed: () {

Navigator.pushNamed(

context, '/selectedCategoryDetailsForm');

},

icon: Icon(

Icons.sell,

color: Color(0xFF140A8C),

)),

IconButton(

onPressed: () {

Navigator.pushNamed(context, '/newChat');

},

icon: Icon(

Icons.message,

color: Color(0xFF140A8C),

)),

IconButton(

onPressed: () {

Navigator.pushNamed(context, '/cart');

},

icon: Icon(

Icons.shopping\_cart,

color: Color(0xFF140A8C),

)),

],

),

),

));

}

}

* 1. **Product.php**

<?php declare(strict\_types=1);

include 'Route.php';

include 'index.php';

include 'connection.php';

include 'otpMail.php';

$route = new Route();

$payload = array();

try {

$method = $route->getRequestMethod();

$endPoint = $route->getEndPoint();

switch ($endPoint) {

case "step-one":

$route->route(array("stepOne"), "POST", $payload, $route);

break;

case "step-two":

$route->route(array("stepTwo"), "POST", $payload, $route);

break;

case "get-uploaded-images":

$route->route(array("getUploadedImages"), "POST", $payload, $route);

break;

case "get-all-products":

$route->route(array("getAllProducts"), "GET", $payload, $route);

break;

case "get-single-products":

$route->route(array("getSingleProducts"), "GET", $payload, $route);

break;

case "get-chat-count-of-a-product":

$route->route(array("getChatCountOfAProduct"), "GET", $payload, $route);

break;

case "get-single-products-cart-count":

$route->route(array("getInsightsOfAProduct"), "GET", $payload, $route);

break;

case "add-to-cart":

$route->route(array("addToCart"), "POST", $payload, $route);

break;

case "get-cart-details":

$route->route(array("getCartDetails"), "GET", $payload, $route);

break;

case "remove-from-cart":

$route->route(array("removeFromCart"), "DELETE", $payload, $route);

break;

case "is-product-is-in-cart":

$route->route(array("isProductIsInCart"), "GET", $payload, $route);

break;

case "get-my-products":

$route->route(array("getMyProducts"), "GET", $payload, $route);

break;

case "search":

$route->route(array("search"), "GET", $payload, $route);

break;

case "search-trending":

$route->route(array("searchTrending"), "GET", $payload, $route);

break;

case "increment-product-view-count":

$route->route(array("incrementProductViewCount"), "PUT", $payload, $route);

break;

case "add-to-search-history":

$route->route(array("addToSearchHistory"), "POST", $payload, $route);

break;

case "remove-from-search-history":

$route->route(array("removeFromSearchHistory"), "DELETE", $payload, $route);

break;

case "change-product-status":

$route->route(array("changeProductStatus"), "POST", $payload, $route);

break;

case "get-unique-category-list":

$route->route(array("getUniqueCategoryList"), "GET", $payload, $route);

break;

case "validator":

$route->route(array("validator"), "GET", $payload, $route);

break;

case "initiate-chat":

$route->route(array("initiateChat"), "POST", $payload, $route);

break;

default:

$route->NotFound404Error();

}

} catch (Exception $e) {

echo $e->getMessage();

}

function stepOne($payload, &$route)

{

$requestBody = file\_get\_contents('php://input');

$data = json\_decode($requestBody, true);

$resultFromToken = $route->verifyToken($data["token"]);

if (!$resultFromToken) {

$route->UnAuthenticationError();

}

$id = $resultFromToken["id"];

if ($id <= 0) {

$route->UnAuthenticationError();

}

$conn = new Connection();

$model = $data["Model"];

$productName = $data["title"];

$yearOfPurchase = $data["purchasedYear"];

$description = $data["description"];

$noOfOwner = $data["noOfOwner"];

$location = $data["location"];

$price = $data["price"];

$category = $data["category"];

$sql = "INSERT INTO product (`model`,`productName`,`yearOfPurchase`,`description`,`noOfOwner` ,`category`,`userId`,`location`,`price`) values ('$model' , '$productName','$yearOfPurchase','$description','$noOfOwner','$category','$id','$location','$price');";

$result = $conn->mysqli->execute\_query($sql);

if ($result != 1) {

$route->InternalServerError();

}

$sql = "SELECT `id` FROM PRODUCT WHERE userId='$id' and productName = '$productName' and model = '$model'";

$result = $conn->mysqli->execute\_query($sql);

$productDetails = json\_encode(mysqli\_fetch\_assoc($result));

$productDetails = json\_decode($productDetails, true);

$route->setResponse(array("data" => $productDetails), "step one completed", "sucess");

}

function stepTwo($payload, &$route)

{

try {

$requestBody = file\_get\_contents('php://input');

$data = json\_decode($requestBody, true);

$resultFromToken = $route->verifyToken($\_GET["token"]);

if (!$resultFromToken) {

$route->UnAuthenticationError();

}

$userId = $resultFromToken["id"];

if ($userId <= 0) {

$route->UnAuthenticationError();

}

if (isset($\_FILES['images'])) {

$images = $\_FILES['images'];

$uploadDir = "uploads/";

$image = $\_FILES["images"];

$imageName = $image["name"];

$imageTmpName = $image["tmp\_name"];

$imageType = $image["type"];

$imageSize = $image["size"];

$imageError = $image["error"];

if ($imageError === UPLOAD\_ERR\_OK) {

$destination = $uploadDir . $imageName;

if (move\_uploaded\_file($imageTmpName, $destination)) {

$imagePath = "http://localhost/api/phpBackEnd/src/uploads/$imageName";

$productId = $\_GET["id"];

$sql = "INSERT INTO images (`userId`,`productId`,`url`) values ($userId,'$productId','$imagePath');";

$conn = new Connection();

$result = $conn->mysqli->execute\_query($sql);

if ($result != 1) {

$route->InternalServerError();

}

$sql = "UPDATE product set status = 'COMPLETED' where id = $productId and status='INCOMPLETE'";

$result = $conn->mysqli->execute\_query($sql);

if ($result != 1) {

$route->InternalServerError();

}

$route->setResponse(

array(

"data" => array(

"userId" => $userId,

"productId" => $productId

)

),

"Image uploaded",

"sucess"

);

} else {

echo "Error moving the uploaded image.";

}

} else {

echo "Error during image upload. Error code: " . $imageError;

}

return;

}

} catch (e) {

$route->InternalServerError();

}

}

function getUploadedImages($payload, &$route)

{

try {

$conn = new Connection();

$requestBody = file\_get\_contents('php://input');

$data = json\_decode($requestBody, true);

$resultFromToken = $route->verifyToken($\_GET["token"]);

if (!$resultFromToken) {

$route->UnAuthenticationError();

}

$userId = $resultFromToken["id"];

$productId = $\_GET["id"];

if ($userId <= 0) {

$route->UnAuthenticationError();

}

$sql1 = "SELECT \* from images where productId = $productId";

$iamgesArray = [];

$images = $conn->mysqli->execute\_query($sql1);

while ($row = mysqli\_fetch\_assoc($images)) {

array\_push($iamgesArray, $row);

}

$route->setResponse($iamgesArray, "sucess", "sucess");

} catch (err) {

$route->InternalServerError();

}

}

function getAllProducts($payload, &$route)

{

try {

$conn = new Connection();

$sql2 = "SELECT p.\*, chatCount,

GROUP\_CONCAT(i.url) as images,

COALESCE(c.cart\_count, 0) as cart\_count

FROM product as p

JOIN images as i ON p.id = i.productId

LEFT JOIN (

SELECT productId, COUNT(\*) as cart\_count

FROM cart

GROUP BY productId

) as c ON p.id = c.productId

LEFT JOIN (

SELECT productid, COUNT(\*) as chatCount

FROM chats

GROUP BY productid

) as ch ON p.id = ch.productid

WHERE p.id = i.productId

AND p.status NOT IN ('INCOMPLETE', 'SOLD','CANCEL','COMPLETE')

GROUP BY i.productId, p.createdAt

ORDER BY p.createdAt DESC;;

";

$products = $conn->mysqli->execute\_query($sql2);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse($productsArray, "sucess", "sucess");

} catch (e) {

print\_r(e);

}

}

function getMyProducts($payload, &$route)

{

try {

$userId = $\_GET["userId"];

$conn = new Connection();

$sql2 = "SELECT p.\*,

GROUP\_CONCAT(i.url) as images,

COALESCE(c.cart\_count, 0) as cart\_count,chatCount

FROM product as p

JOIN images as i ON p.id = i.productId

LEFT JOIN (

SELECT productId, COUNT(\*) as cart\_count

FROM cart

GROUP BY productId

) as c ON p.id = c.productId

LEFT JOIN (

SELECT productid, COUNT(\*) as chatCount

FROM chats

GROUP BY productid

) as ch ON p.id = ch.productid

WHERE p.id = i.productId

AND p.userId =$userId

GROUP BY i.productId, p.createdAt

ORDER BY p.createdAt DESC;";

$products = $conn->mysqli->execute\_query($sql2);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse($productsArray, "sucess", "sucess");

} catch (e) {

print\_r(e);

}

}

function getSingleProducts($payload, &$route)

{

try {

$conn = new Connection();

$id = $route->getQuery("productId");

$sql2 = "SELECT p.\* , GROUP\_CONCAT(url) as images FROM product as p,images as i where p.id = i.productId and p.id = $id and p.status not in ('INCOMPLETE','SOLD') GROUP by i.productId;";

$products = $conn->mysqli->execute\_query($sql2);

// print\_r("imagesResult");

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse($productsArray, "sucess", "sucess");

} catch (e) {

print\_r(e);

}

}

function getChatCountOfAProduct($payload, &$route)

{

try {

$conn = new Connection();

$id = $route->getQuery("productId");

$sql2 = "Select count(productid) as count from chats where productid =$id;";

$products = $conn->mysqli->execute\_query($sql2);

// print\_r("imagesResult");

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse($productsArray, "sucess", "sucess");

} catch (err) {

print(err);

}

}

function getInsightsOfAProduct($payload, &$route)

{

try {

$conn = new Connection();

$id = $route->getQuery("productId");

$sql2 = "SELECT COUNT(c.productId) as cartCount FROM cart as c , product as p WHERE p.id = c.productId and p.id =$id;";

$products = $conn->mysqli->execute\_query($sql2);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse($productsArray[0], "sucess", "sucess");

} catch (e) {

print\_r(e);

}

}

function isProductIsInCart($payload, &$route)

{

try {

$conn = new Connection();

$pid = $route->getQuery("productId");

$uid = $route->getQuery("userId");

$sql2 = "SELECT \* FROM cart as c WHERE c.productId =$pid and c.userId=$uid;";

$products = $conn->mysqli->execute\_query($sql2);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$isPresent = false;

if (sizeof($productsArray) > 0) {

$isPresent = true;

}

$route->setResponse(array("isPresent" => $isPresent), "sucess", "sucess");

} catch (e) {

print\_r(e);

}

}

function addToCart($payload, &$route)

{

try {

$conn = new Connection();

$pid = $route->getQuery("productId");

$uid = $route->getQuery("userId");

$sql1 = "SELECT \* FROM cart as c WHERE c.productId = $pid and c.userId =$uid;";

$sql2 = "SELECT \* FROM product as p WHERE p.id= $pid and p.userId =$uid;";

$products = $conn->mysqli->execute\_query($sql1);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

if (sizeof($productsArray) >= 1) {

$route->setResponse(array("data" => "This product is already exist in cart"), "This product is already exist in cart", "failure");

}

$products = $conn->mysqli->execute\_query($sql2);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

if (sizeof($productsArray) >= 1) {

$route->setResponse(array("data" => "You cont add your product in cart"), "This product is already exist in cart", "failure");

}

$sql2 = "INSERT INTO `cart` (`id`, `userId`, `productId`, `createdAt`, `updatedAt`) VALUES (NULL, '$uid', '$pid', current\_timestamp(), current\_timestamp());";

$products = $conn->mysqli->execute\_query($sql2);

if ($products == 1) {

$route->setResponse(array("data" => "product added to cart"), "product added to cart", "sucess");

} else {

$route->InternalServerError();

}

} catch (e) {

print\_r(e);

}

}

function removeFromCart($payload, &$route)

{

try {

$conn = new Connection();

$pid = $route->getQuery("productId");

$uid = $route->getQuery("userId");

$sql1 = "SELECT \* FROM cart as c WHERE c.productId = $pid and c.userId =$uid;";

$products = $conn->mysqli->execute\_query($sql1);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

if (sizeof($productsArray) == 0) {

$route->setResponse(array("data" => "This product not exist in cart"), "This product not exist in cart", "failure");

}

$sql1 = "DELETE FROM cart WHERE `userId` = '$uid' and `productId` = '$pid';";

$products = $conn->mysqli->execute\_query($sql1);

if ($products == 1) {

$route->setResponse(array("data" => "product removed from cart"), "Product removed from cart", "sucess");

} else {

$route->InternalServerError();

}

} catch (e) {

print\_r(e);

}

}

function getCartDetails($payload, &$route)

{

try {

$conn = new Connection();

$uid = $route->getQuery("userId");

$sql1 = "SELECT p.\* , GROUP\_CONCAT(i.url) as images FROM cart as c , product as p , images as i WHERE c.productId = p.id and i.productId = p.id and c.userId =$uid GROUP BY p.id order by p.createdAt DESC;";

$products = $conn->mysqli->execute\_query($sql1);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse(array("data", $productsArray), "Here is a cart details", "sucess");

} catch (e) {

print\_r(e);

}

}

function getUniqueCategoryList($payload, &$route)

{

try {

$conn = new Connection();

$sql1 = "SELECT DISTINCT category FROM product;";

$category = $conn->mysqli->execute\_query($sql1);

$categoryArray = [];

while ($row = mysqli\_fetch\_assoc($category)) {

array\_push($categoryArray, $row);

}

$route->setResponse($categoryArray, "unique category", "sucess");

} catch (e) {

print\_r(e);

}

}

function search($payload, &$route)

{

try {

$conn = new Connection();

$search = $route->getQuery("search");

$sql = "SELECT p.\*,

GROUP\_CONCAT(i.url) as images,

COALESCE(c.cart\_count, 0) as cart\_count,

COALESCE(ch.chatCount, 0) as chat\_count

FROM product as p

JOIN images as i ON p.id = i.productId

LEFT JOIN (

SELECT productId, COUNT(\*) as cart\_count

FROM cart

GROUP BY productId

) as c ON p.id = c.productId

LEFT JOIN (

SELECT productid, COUNT(\*) as chatCount

FROM chats

GROUP BY productid

) as ch ON p.id = ch.productid

WHERE (p.productName LIKE '%$search%'

OR p.model LIKE '%$search%'

OR p.description LIKE '%$search%'

OR p.yearOfPurchase LIKE '%$search%'

OR p.price LIKE '%$search%'

OR p.noOfOwner LIKE '%$search%'

OR p.location LIKE '%$search%'

OR p.category LIKE '%$search%')

AND p.status NOT IN ('INCOMPLETE', 'SOLD','CANCEL')

GROUP BY p.id

LIMIT 5;

";

$searchResult = $conn->mysqli->execute\_query($sql);

$searchResultArray = [];

while ($row = mysqli\_fetch\_assoc($searchResult)) {

array\_push($searchResultArray, $row);

}

$route->setResponse($searchResultArray, 'search result', 'sucess');

} catch (e) {

print\_r(e);

}

}

function searchTrending($payload, &$route)

{

try {

$conn = new Connection();

$userId = $route->getQuery('userId');

$sql = "SELECT p.\*, COUNT(c.id) as noOfCart FROM product as p

JOIN cart as c ON p.id = c.productId

GROUP BY p.id

ORDER BY p.views DESC, noOfCart DESC LIMIT 5;";

$trendingResult = $conn->mysqli->execute\_query($sql);

$searchResultArray = [];

while ($row = mysqli\_fetch\_assoc($trendingResult)) {

array\_push($searchResultArray, $row);

}

$sql2 = "SELECT \* FROM `recentlysearchhistory` as r , `product` as p WHERE r.userId =$userId and p.id = r.productId ORDER BY r.createdAt DESC limit 5";

$recentSearchResult = $conn->mysqli->execute\_query($sql2);

$recentSearchResultArray = [];

while ($row = mysqli\_fetch\_assoc($recentSearchResult)) {

array\_push($recentSearchResultArray, $row);

}

$route->setResponse(array("trendingResult" => $searchResultArray, "recentSearch" =>

$recentSearchResultArray, ), "search result", "sucess");

} catch (e) {

$route->InternalServerError();

}

}

function incrementProductViewCount($payload, &$route)

{

try {

$conn = new Connection();

$productId = $route->getQuery("productId");

$sql = "update product set views =views+1 where id = $productId;";

$result = $conn->mysqli->execute\_query($sql);

if ($result) {

$route->setResponse(array("data" => "sucessfull incremented product view"), "sucessfull incremented product view", "sucess");

}

} catch (e) {

$route->InternalServerError();

}

}

function addToSearchHistory($payload, &$route)

{

try {

$productId = $route->getQuery("productId");

$userId = $route->getQuery("userId");

$sql1 = "select \* from recentlysearchhistory where productId=$productId and $userId = $userId";

$conn = new Connection();

$isAlredyExist = $conn->mysqli->execute\_query($sql1);

$count = 0;

while ($row = mysqli\_fetch\_assoc($isAlredyExist)) {

$count += 1;

}

// update

if ($count >= 1) {

$sql2 = "update recentlysearchhistory set createdAt = current\_timestamp() where productId=$productId and $userId = $userId";

$isInserted = $conn->mysqli->execute\_query($sql2);

if ($isInserted == 1) {

$route->setResponse(array(["updated sucess"]), "updated sucess", "sucess");

} else {

$route->setResponse(array(["not updated"]), "not updated", "failure");

}

} else {

$sql2 = "INSERT INTO `recentlysearchhistory` (`id`, `userId`, `productId`, `createdAt`) VALUES (NULL, $userId, $productId, current\_timestamp());";

$isInserted = $conn->mysqli->execute\_query($sql2);

if ($isInserted == 1) {

$route->setResponse(array(["Inserted sucess"]), "Inserted sucess", "sucess");

} else {

$route->setResponse(array(["not Inserted"]), "not Inserted", "failure");

}

}

} catch (err) {

$route->InternalServerError();

}

}

function removeFromSearchHistory($payload, &$route)

{

try {

$productId = $route->getQuery("productId");

$userId = $route->getQuery("userId");

$sql = "DELETE FROM recentlysearchhistory WHERE userId=$userId and productId=$productId";

$conn = new Connection();

$isDeleted = $conn->mysqli->execute\_query($sql);

if ($isDeleted) {

$route->setResponse(array("data", "Removed sucesfully"), "Removed sucesfully", "sucess");

} else {

$route->setResponse(array("data", "Not removed"), "Not removed", "failure");

}

} catch (err) {

$route->InternalServerError();

}

}

function changeProductStatus($payload, &$route)

{

try {

$allowedStatus = ["CANCEL", "SOLD", "VERIFIED"];

$productId = $route->getQuery("productId");

$status = $route->getQuery("status");

if (in\_array($status, $allowedStatus)) {

$sql = "update product set status='$status' where id = $productId";

$conn = new Connection();

$isUpdated = $conn->mysqli->execute\_query($sql);

if (!$isUpdated) {

$route->setResponse(array("data", "Not updated"), "Not updated", "failure");

}

$sql2 = "delete from cart where productId = '$productId'";

$isdeleted = $conn->mysqli->execute\_query($sql2);

$route->setResponse(array("data", "updated sucesfully"), "updated sucesfully", "sucess");

}

$route->RequirementsNotMatchedError();

} catch (e) {

print(e);

}

}

function validator($payload, &$route)

{

try {

$conn = new Connection();

$sql2 = "SELECT p.\* , GROUP\_CONCAT(i.url) as images FROM product as p , images as i WHERE i.productId = p.id and p.status = 'COMPLETED' GROUP BY p.id order by p.createdAt DESC;";

$products = $conn->mysqli->execute\_query($sql2);

$productsArray = [];

while ($row = mysqli\_fetch\_assoc($products)) {

array\_push($productsArray, $row);

}

$route->setResponse($productsArray, "sucess", "sucess");

} catch (err) {

}

}

function initiateChat($payload, &$route)

{

try {

$conn = new Connection();

$userId = $route->getQuery("userId");

$productId = $route->getQuery("productId");

$sql2 = "SELECT name from user WHERE id = $userId;";

$user = $conn->mysqli->execute\_query($sql2);

$secondPartyName = "";

while ($row = mysqli\_fetch\_assoc($user)) {

$secondPartyName = $row["name"];

}

$sql3 = "SELECT u.name , u.id from user as u , product as p WHERE p.id = $productId and u.id = p.userId;";

$first = $conn->mysqli->execute\_query($sql3);

$firstPartyName = "";

$firstPartyId = "";

while ($row = mysqli\_fetch\_assoc($first)) {

$firstPartyName = $row["name"];

$firstPartyId = $row["id"];

}

$sql4 = "INSERT INTO `chats` (`firstparty`, `secondparty`, `productid`, `sellerid`, `firstpartyname`, `secondpartyname`) VALUES ('$firstPartyId', '$userId', '$productId', '$firstPartyId', '$firstPartyName', '$secondPartyName');";

$isInserted = $conn->mysqli->execute\_query($sql4);

if ($isInserted == 1) {

$route->setResponse(array("data" => "chat initiated"), "chat initiated", "sucess");

}

$route->RequirementsNotMatchedError();

} catch (err) {

}

}

?>

**CHAPTER 6**

**TESTING**

**6.1 Testing**

Testing is the evaluation of the software against requirements gathered from users and system specifications. Testing identifies important defects, flaws, or an error in the application code that must be fixed. It also assesses the feature of a system. Testing assesses the quality of the product.

**6.2 Unit Testing**

Unit testing refers to the testing of certain functions and areas of the code. It gives the ability to verify that all the functions work as expected. Eventually, it helps to identify failures in the algorithms as well as logic to help improve the quality of the code that composes a certain function.

**6.3 Integration Testing**

Integration testing is basically a logical extension of unit testing. In simple words, two tested units are combined into a component and the interface between them is tested. It identifies problems that occur when different units are combined. The different modules of this project have undergone integration testing while being merged.

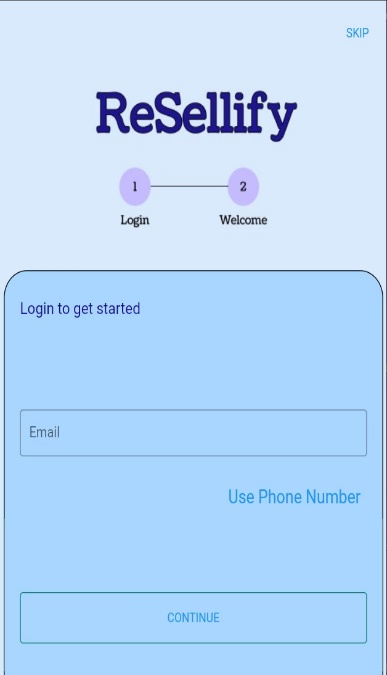
**6.4 System Testing**

System testing tests the behavior of the whole system as defined by the scope of the development project. It might include tests based on risks as well as requirement specifications, business processes, use cases, or other high-level descriptions of system behavior, interactions with the operating systems, and system resources. It is most often the final test performed to verify that the system meets the specification and its objectives. System testing has been performed at the completion of each feature and is still taking place to make improvements to the existing system.

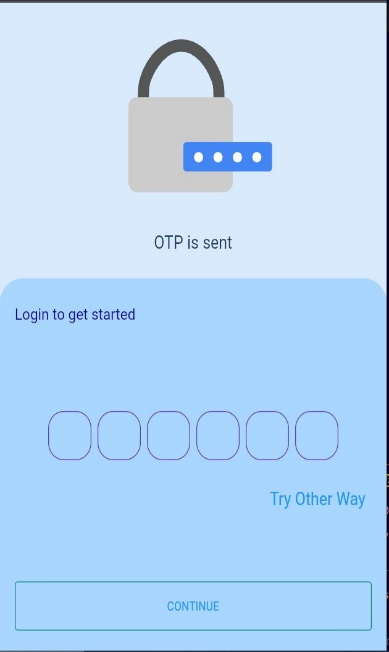
**CHAPTER 7**

**RESULTS**

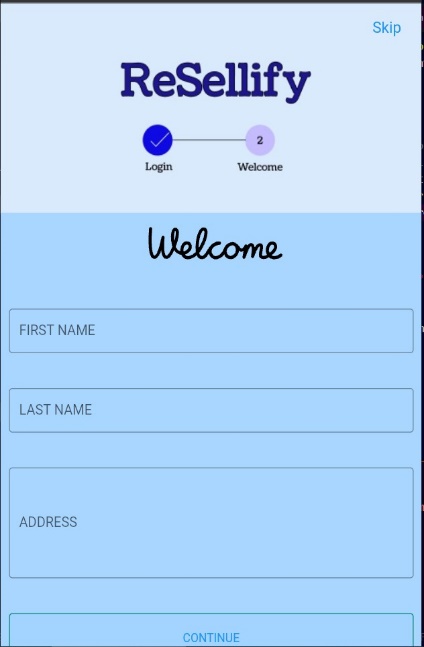
**7.1 Snapshots:**

****

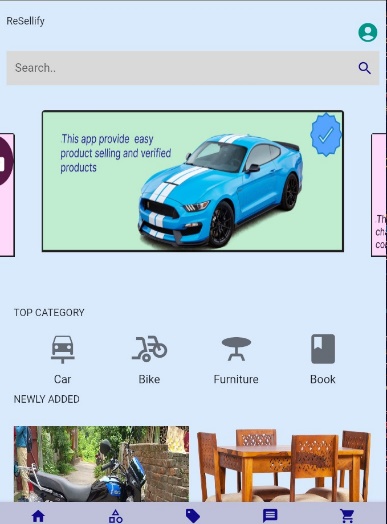
**Fig 7.1: Login Page**



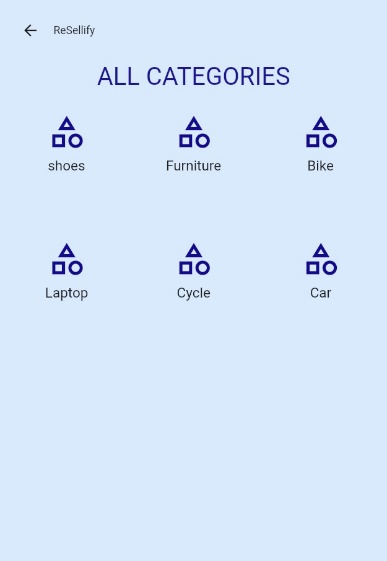
**Fig 7.2: OTP Page**



**Fig 7.3: Welcome Page**



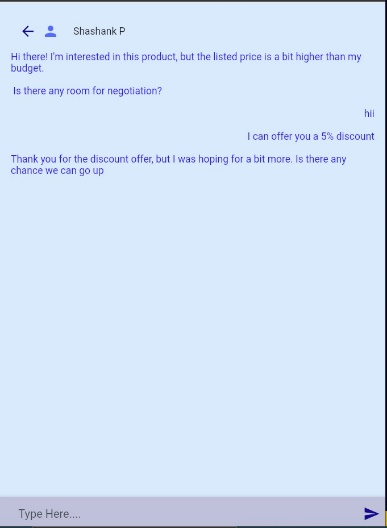
**Fig 7.4: Home Page**



**Fig 7.5: Categories Page**



**Fig 7.6: Chat Page**



**Fig 7.7: Chatting Page**

## CONCLUSION

## In conclusion, the Resellify mobile application project presents a promising solution for facilitating the buying and selling of preowned products. The application aims to address the growing demand for a user-friendly and secure platform to engage in the secondhand marketplace.

## Throughout the project, considerable effort has been dedicated to understanding the challenges faced by both buyers and sellers in the used product marketplace. By leveraging modern technology and innovative features, Resellify provides a seamless and efficient user experience, allowing users to explore a wide range of preowned products conveniently.

## The project's success relies on its ability to tackle deceptive marketing and deceptive user activities, which can mislead the system and lead to financial losses for unsuspecting users. As the platform gains traction, it will be crucial to implement robust verification and authentication mechanisms, as well as strict content moderation, to maintain a trustworthy and reliable environment for all users.

## Furthermore, Resellify's emphasis on data security and user privacy enhances the application's appeal and instills confidence among users to engage in transactions without fear of unauthorized access or data breaches.

## By combining traditional marketing communication methods with modern tools, Resellify aims to attract and retain a diverse user base, ensuring the application's continued growth and relevance in the ever-changing mobile market.

## Overall, the Resellify mobile application project holds significant potential to revolutionize the preowned product marketplace and empower users to buy and sell with confidence. With continuous improvements, diligent monitoring, and a commitment to user satisfaction, Resellify is poised to become a leading platform in the thriving world of secondhand commerce.

**LIMITATION**

**Limitation of our project:**

**Scalability:** Depending on the architecture and infrastructure setup, the project's scalability may be limited. As the user base and data volume grow, it could require more substantial resources and optimization to handle increasing traffic effectively.

**Performance:** The performance of the application may be affected if the backend server is not optimized correctly or if there are network latency issues. Slow response times could lead to a less satisfactory user experience.

**Security:** While efforts may have been made to ensure security, PHP and MySQL applications could be vulnerable to common web application security threats like SQL injection and cross-site scripting (XSS) attacks. Regular security audits and updates are necessary to maintain a secure environment.

**Compatibility:** Flutter is primarily focused on mobile app development, which means that cross-platform compatibility may not extend to all devices and operating systems. Some older or less common devices may experience compatibility issues.

**Limited Platform Integration**: Depending on the project's scope, the integration with certain device features (e.g., specific hardware sensors, biometrics) may be limited or require additional work due to Flutter's cross-platform nature.

**Offline Functionality**: Depending on the application's design, offline functionality might be challenging to implement in Flutter, especially when dealing with data synchronization with the server.

**Community Support**: The project's long-term success could be impacted by the availability and reliability of community support for Flutter, MySQL, and PHP. If these technologies become less popular or experience a decline in community engagement, finding help and resources may become more challenging.

**Data Integrity and Recovery**: In the event of a server failure or data corruption, ensuring data integrity and implementing proper data recovery mechanisms becomes crucial for maintaining user trust and avoiding significant data loss.

**FUTURE ENHANCEMENT**

**Hosting on Cloud Infrastructure**: Migrating the Resellify application to a cloud-based hosting service, such as Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure, can offer numerous benefits. Cloud hosting provides scalability, ensuring that the application can handle increased traffic and user demand without any major infrastructure changes. It also enhances the app's reliability, as cloud providers typically offer high availability and redundancy to minimize downtime.

**Cloud-Based Database and Storage**: Storing data in a cloud-based database, such as Google Cloud Firestore or AWS DynamoDB, can improve data accessibility and enable real-time synchronization across devices. Cloud-based storage solutions like AWS S3 or Google Cloud Storage can efficiently handle media files and product images, ensuring faster load times and reducing the burden on the app's backend servers.

**Real-Time Chat and Notifications**: Incorporating real-time chat functionality between buyers and sellers can facilitate faster communication and negotiation. Additionally, push notifications for important updates, messages, and transaction statuses will keep users informed and engaged.

**Secure Payment Gateways**: Integrating secure and widely trusted payment gateways will enhance the app's credibility and instill confidence in users to make secure transactions.

**Location-Based Services**: Leveraging location-based services can provide users with localized product listings and help buyers find sellers nearby for easier and faster transactions.

**Multi-Language Support:** Adding multi-language support will cater to a diverse user base, potentially expanding the app's reach to international markets

## REFERENCES

1. Department of Marketing, Branding & Tourism, Middlesex University, Business School, W109 Williams Building, Hendon Campus, The Burroughs, London NW4 4BT, England, UK
2. Department of Marketing, Branding & Tourism, Middlesex University, Business School, W107 Williams Building, Hendon Campus, The Burroughs, London NW4 4BT, England, UK
3. Praveen K. Kopalle TuckSchool of Business, Dartmouth College, Hanover, New Hampshire 03755, kopalle@dartmouth.edu
4. Donald R. Lehmann Graduate School of Business, Columbia University, New York, New York 10027, [drl2@columbia.edu](mailto:drl2@columbia.edu)