Vivekanand Education Society's Institute of Technology

An Autonomous Institute Affiliated to University of Mumbai
Hashu Advani Memorial Complex, Collector Colony, Chembur East, Mumbai - 400074.

Department of Information Technology

CERTIFICATE

This is to certify that **_Eesha Chavan-08_____** of **D15A** semester **VI,** have successfully completed necessary experiments in the **MAD & PWA Lab** under my supervision in **VES Institute of Technology** during the academic year **2024-2025.**

Lab Assistant

Subject Teacher

Mrs. Kajal Joseph

Principal

Head of Department

Dr. Mrs. Shalu Chopra

Name of the Course : MAD & PWA Lab **Course Code :** ITL604

Year/Sem/Class : D15A/D15B **A.Y.:** 24-25

Faculty Incharge : Mrs. Kajal Joseph.

Lab Teachers : Mrs. Kajal Joseph.

Email : kajal.jewani@ves.ac.in

Programme Outcomes: The graduate will be able to:

PO1) Basic Engineering knowledge: An ability to apply the fundamental knowledge in mathematics, science and engineering to solve problems in Computer engineering.

PO2) Problem Analysis: Identify, formulate, research literature and analyze computer engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and computer engineering and sciences.

PO3) Design/ Development of Solutions: Design solutions for complex computer engineering problems and design system components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal and environmental considerations.

PO4) Conduct investigations of complex engineering problems using research-based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of information to provide valid conclusions.

PO5) Modern Tool Usage: Create, select and apply appropriate techniques, resources and modern computer engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6) The Engineer and Society: Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to computer engineering practice.

PO7) Environment and Sustainability: Understand the impact of professional computer engineering solutions in societal and environmental contexts and demonstrate knowledge of and need for sustainable development.

PO8) Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of computer engineering practice.

PO9) Individual and Team Work: Function effectively as an individual, and as a member or leader in diverse teams and in multidisciplinary settings.

PO10) Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations and give and receive clear instructions.

PO11) Project Management and Finance: Demonstrate knowledge and understanding of computer engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12) Life-long Learning: Recognize the need for and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Program specific Outcomes

PSO1) An ability to manage and analyze data / information effectively for making better decisions.

PSO2) Demonstrate the ability to use state of the art technologies and tools including Free and Open Source Software (FOSS) tools in developing software.

Lab Objectives:

Sr. No.	Lab Objectives
The Lab experiments aims:	
1	Learn the basics of the Flutter framework.
2	Develop the App UI by incorporating widgets, layouts, gestures and animation
3	Create a production ready Flutter App by including files and firebase backend service.
4	Learn the Essential technologies, and Concepts of PWAs to get started as quickly and efficiently as possible
5	Develop responsive web applications by combining AJAX development techniques with the jQuery JavaScript library.
6	Understand how service workers operate and also learn to Test and Deploy PWA.

Lab Outcomes:

Sr. No.	Lab Outcomes	Cognitive levels of attainment as per Bloom's Taxonomy
On Completion of the course the learner/student should be able to:		
1	Understand cross platform mobile application development using Flutter framework	L1, L2
2	Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation	L3
3	Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS	L3, L4
4	Understand various PWA frameworks and their requirements	L1, L2
5	Design and Develop a responsive User Interface by applying PWA Design techniques	L3
6	Develop and Analyse PWA Features and deploy it over app hosting solutions	L3, L4

Lab Objectives:

Sr. No.	Lab Objectives
The Lab experiments aims:	
1	Learn the basics of the Flutter framework.
2	Develop the App UI by incorporating widgets, layouts, gestures and animation
3	Create a production ready Flutter App by including files and firebase backend service.
4	Learn the Essential technologies, and Concepts of PWAs to get started as quickly and efficiently as possible
5	Develop responsive web applications by combining AJAX development techniques with the jQuery JavaScript library.
6	Understand how service workers operate and also learn to Test and Deploy PWA.

Lab Outcomes:

Sr. No.	Lab Outcomes	Cognitive levels of attainment as per Bloom's Taxonomy
On Completion of the course the learner/student should be able to:		
1	Understand cross platform mobile application development using Flutter framework	L1, L2
2	Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation	L3
3	Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS	L3, L4
4	Understand various PWA frameworks and their requirements	L1, L2
5	Design and Develop a responsive User Interface by applying PWA Design techniques	L3
6	Develop and Analyse PWA Features and deploy it over app hosting solutions	L3, L4

MAD & PWA Lab
Journal

Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	

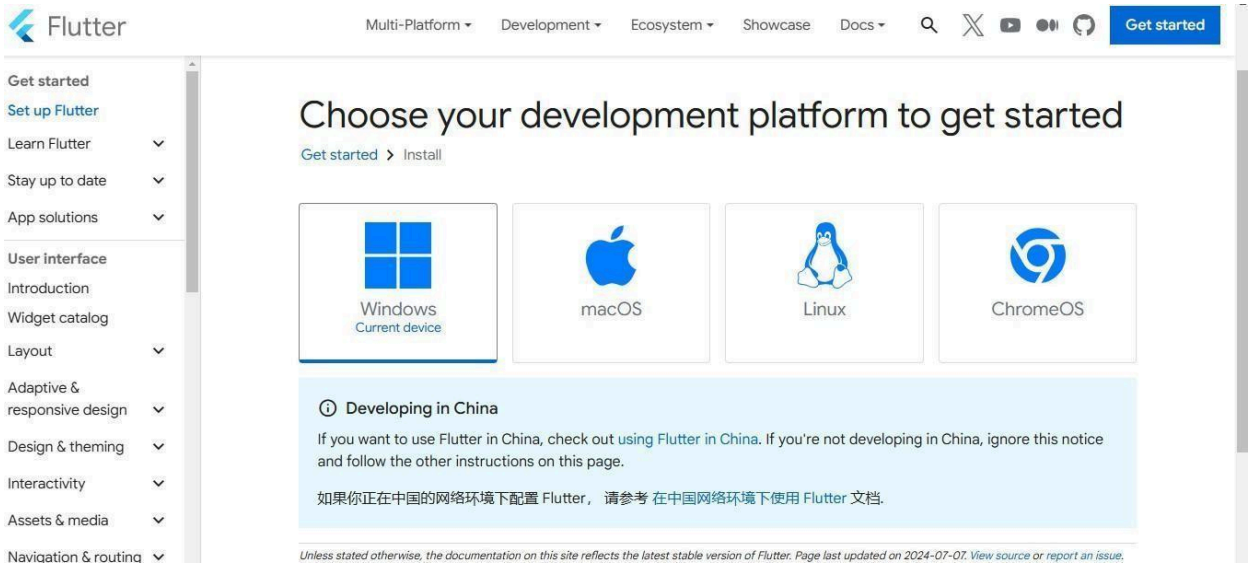
Project Title: The Souled Store Clone **Roll No. 08**

Eesha Chavan **D15A-08**

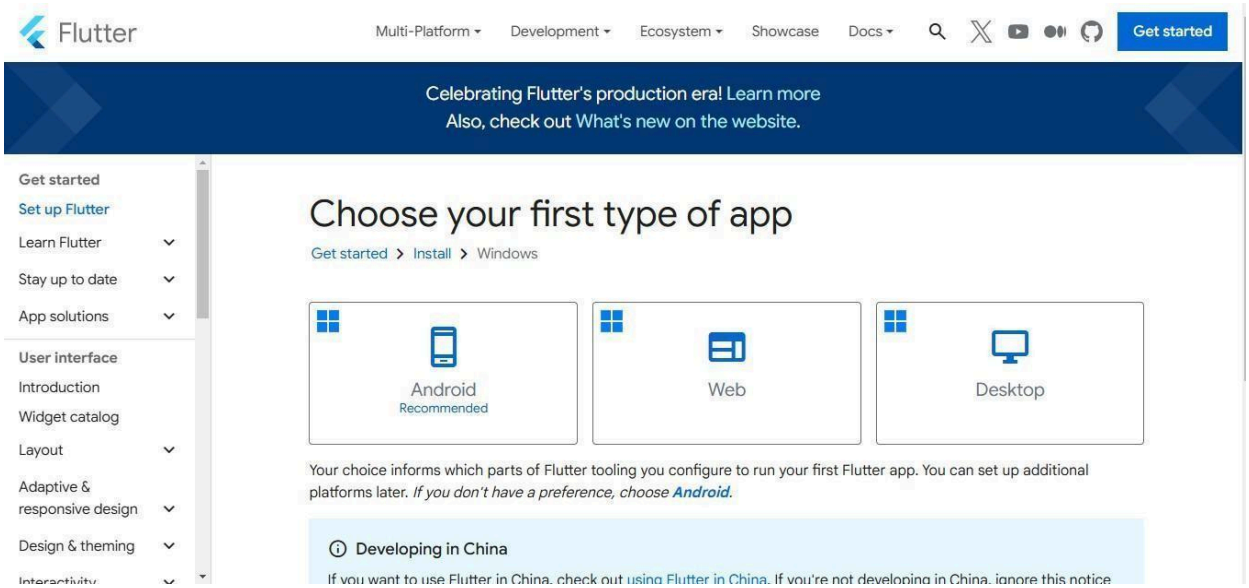
MPL Experiment 1

AIM: - Installation and Configuration of Flutter Environment.

Step 1: Go to the official Flutter website: https://docs.flutter.dev/get-started/install

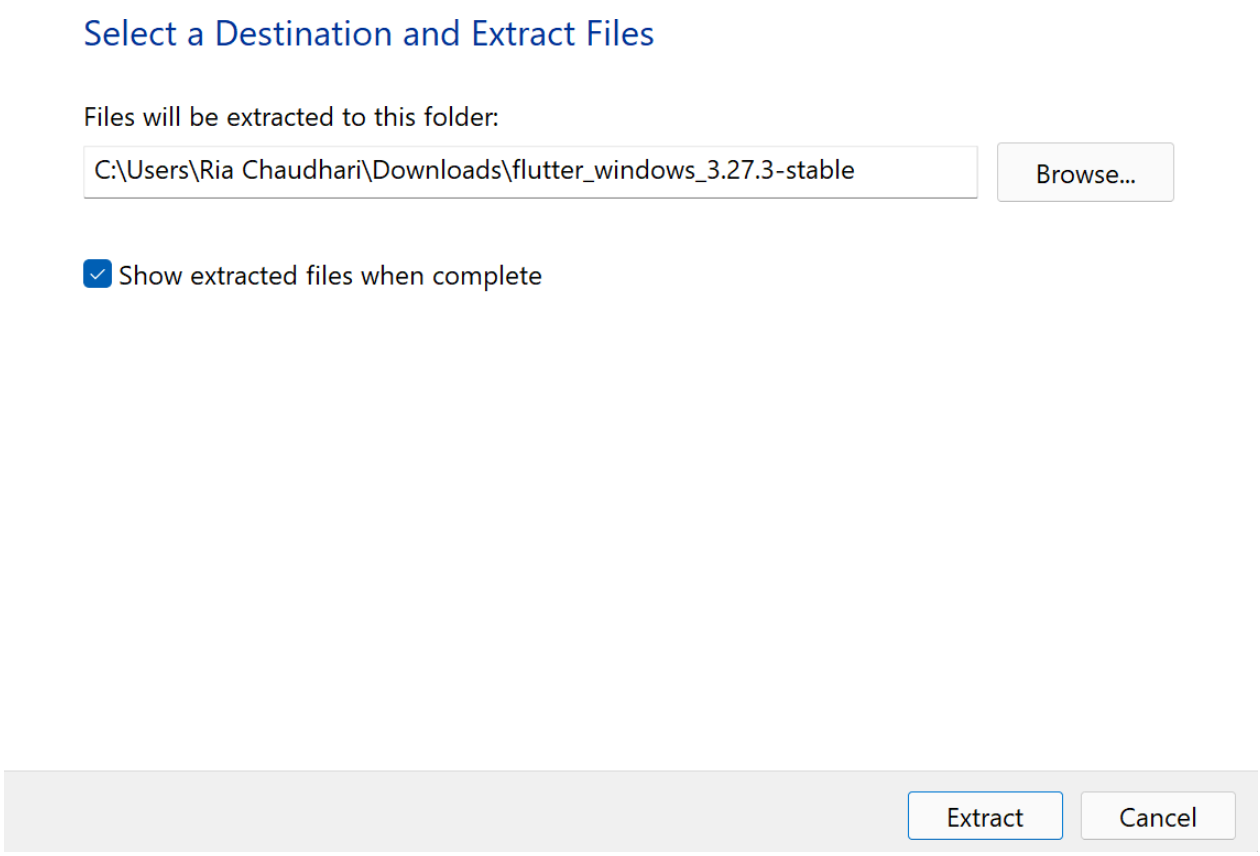


Step 2: To download the latest Flutter SDK, click on the Windows icon > Android

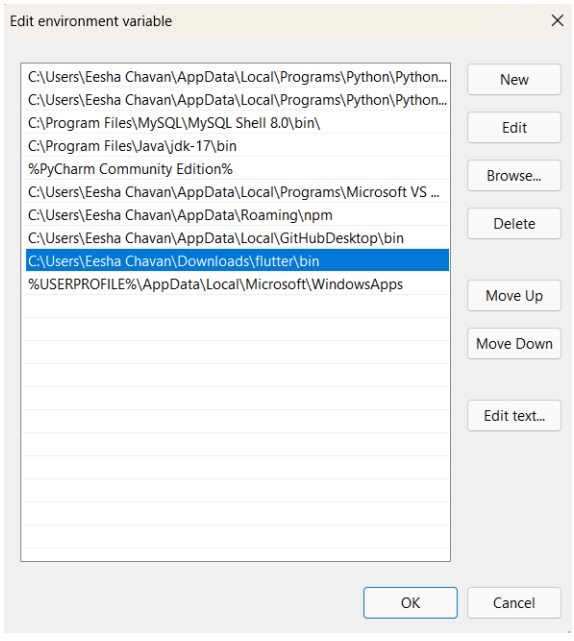


Step 3: For Windows, download the stable release (a .zip file).

Step 4: Extract the ZIP file to a folder (e.g., C:\flutter)



Step 5 :- Add Flutter to System PATH Right-click on the Start Menu > System > Advanced system settings > Environment Variables. Under System Variables, find Path and click Edit. Add the full path to the flutter/bin directory (e.g., C:\flutter\bin).



Step 6 : - Now, run the $ flutter command in command prompt

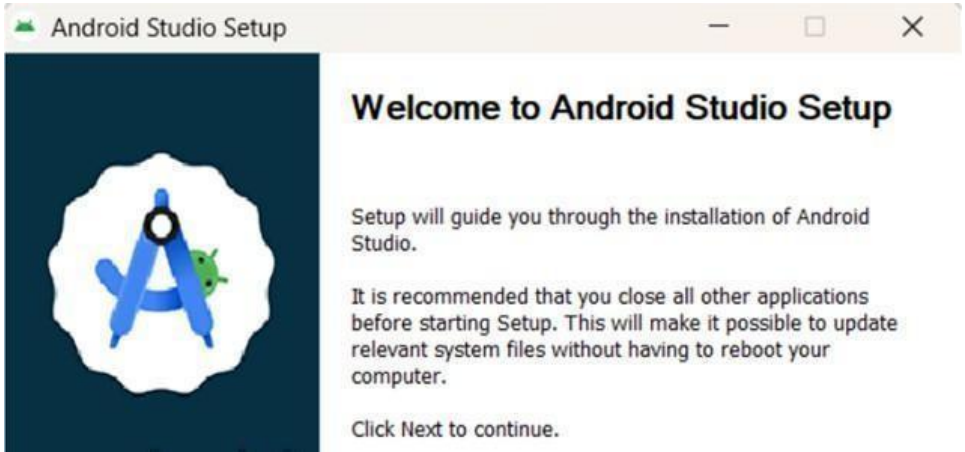
```
C:\Users\Eesha Chavan>flutter

  A new version of Flutter is available!

  To update to the latest version, run "flutter upgrade".

Manage your Flutter app development.

Common commands:

  flutter create <output directory>
    Create a new Flutter project in the specified directory.

  flutter run [options]
    Run your Flutter application on an attached device or in an emulator.

Usage: flutter <command> [arguments]

Global options:
-h, --help                  Print this usage information.
-v, --verbose               Noisy logging, including all shell commands executed.
                            If used with "--help", shows hidden options. If used with "flutter doctor", shows additional
                            diagnostic information. (Use "-vv" to force verbose logging in those cases.)
-d, --device-id             Target device id or name (prefixes allowed).
    --version               Reports the version of this tool.
    --enable-analytics      Enable telemetry reporting each time a flutter or dart command runs.
    --disable-analytics     Disable telemetry reporting each time a flutter or dart command runs, until it is
                            re-enabled.
```

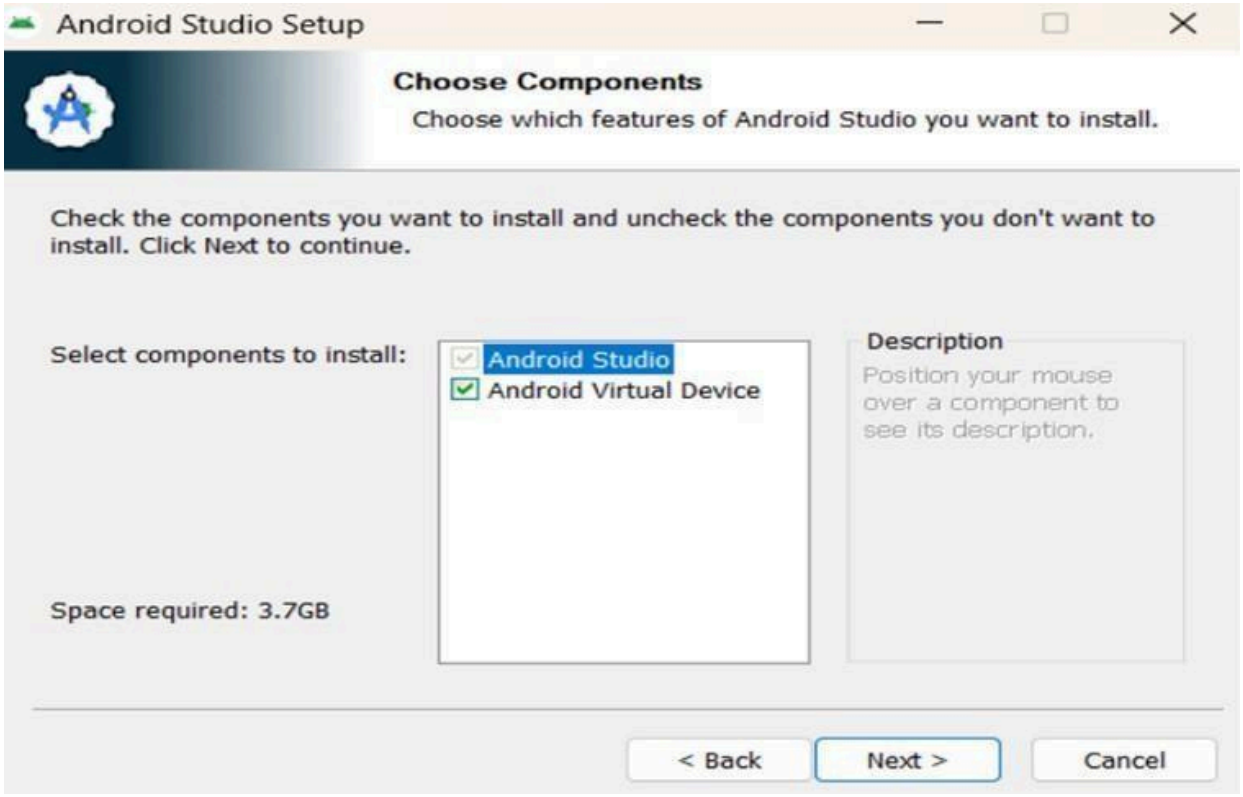
Step 7:- Run the $ flutter doctor command. This command checks for all the requirements of Flutter app development and displays a report of the status of your Flutter installation

```
C:\Users\Eesha Chavan>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.26100.3037], locale en-IN)
[√] Windows Version (Installed version of Windows is version 10 or higher)
[√] Android toolchain - develop for Android devices (Android SDK version 35.0.0)
[√] Chrome - develop for the web
[√] Visual Studio - develop Windows apps (Visual Studio Build Tools 2019 16.11.33)
[√] Android Studio (version 2024.1)
[√] VS Code (version 1.96.0)
[√] Connected device (3 available)
[√] Network resources

• No issues found!
```

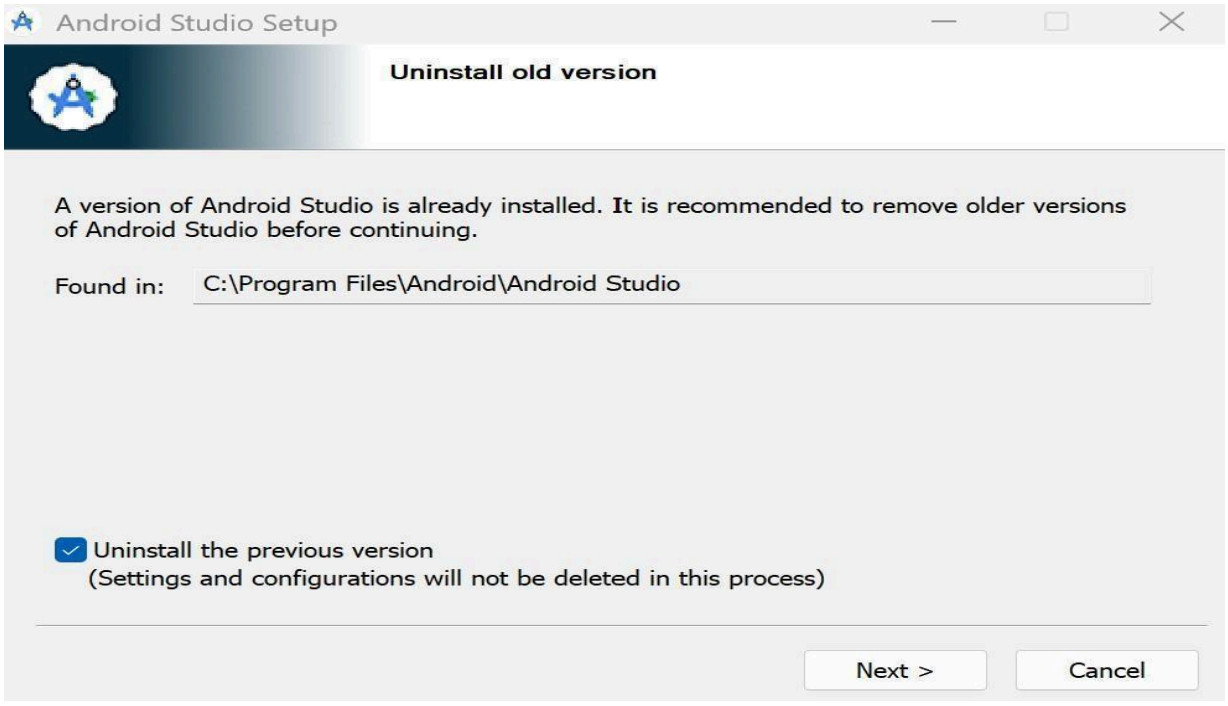
Step 8 : - Go to Android Studio and download the installer

Step 8.1: - When the download is complete, open the .exe file and run it. You will get the following dialog box

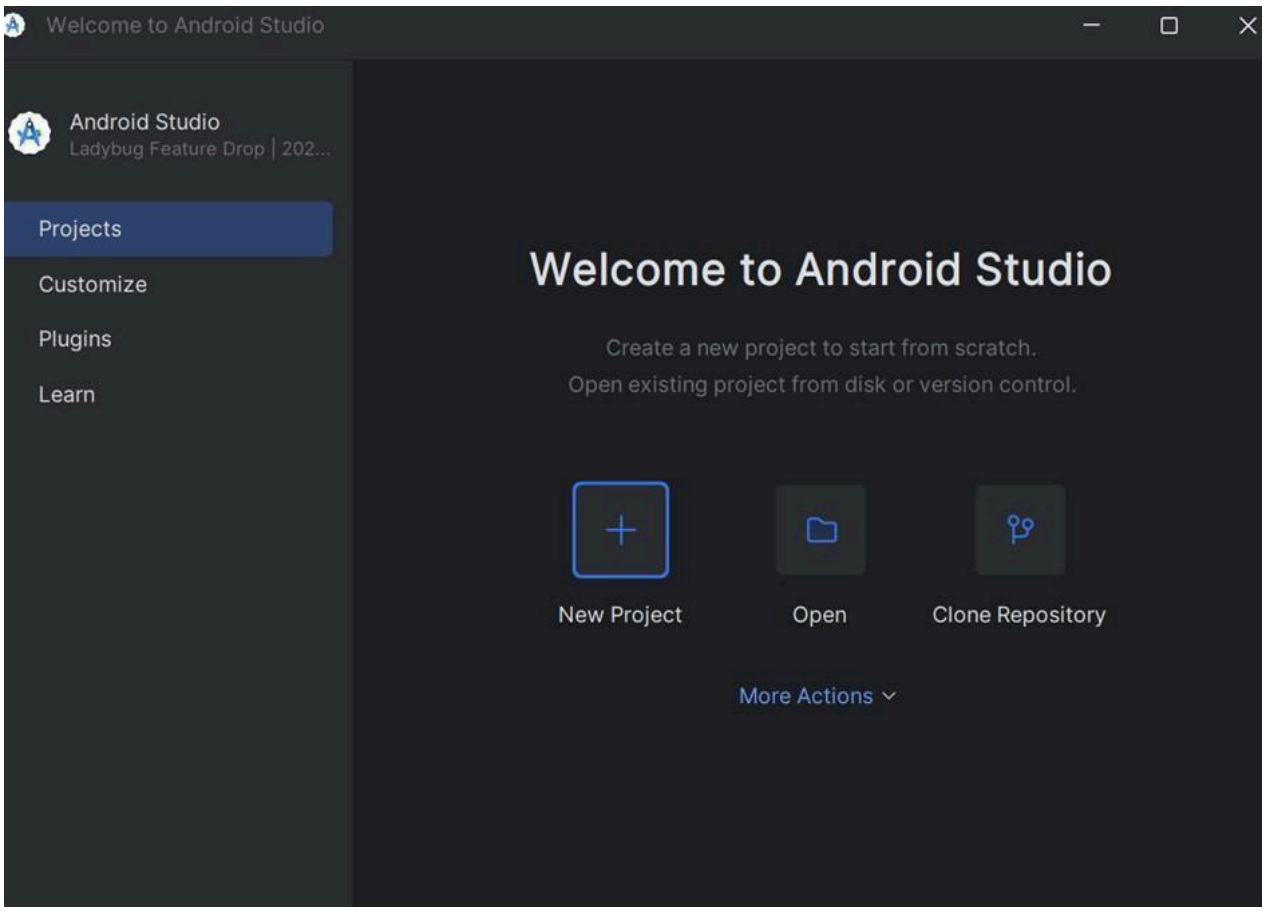




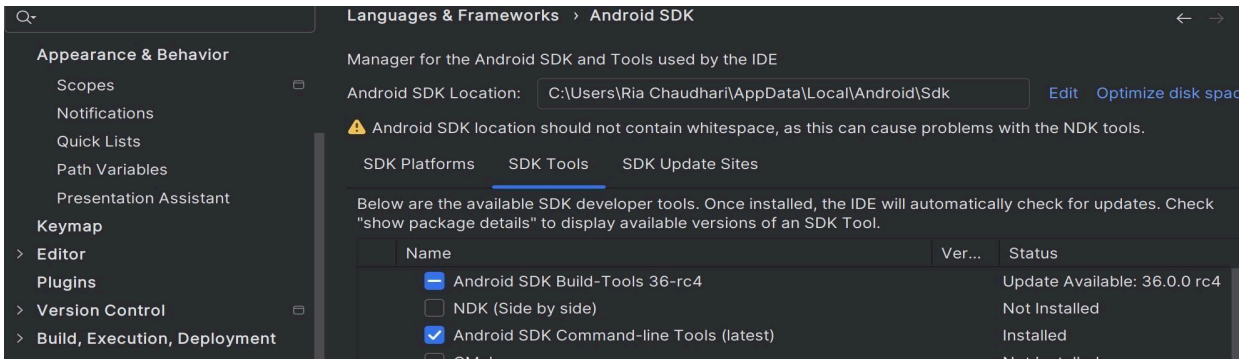
Step 8.3: - Change the destination as per your convenience and click on 'Next' Button.



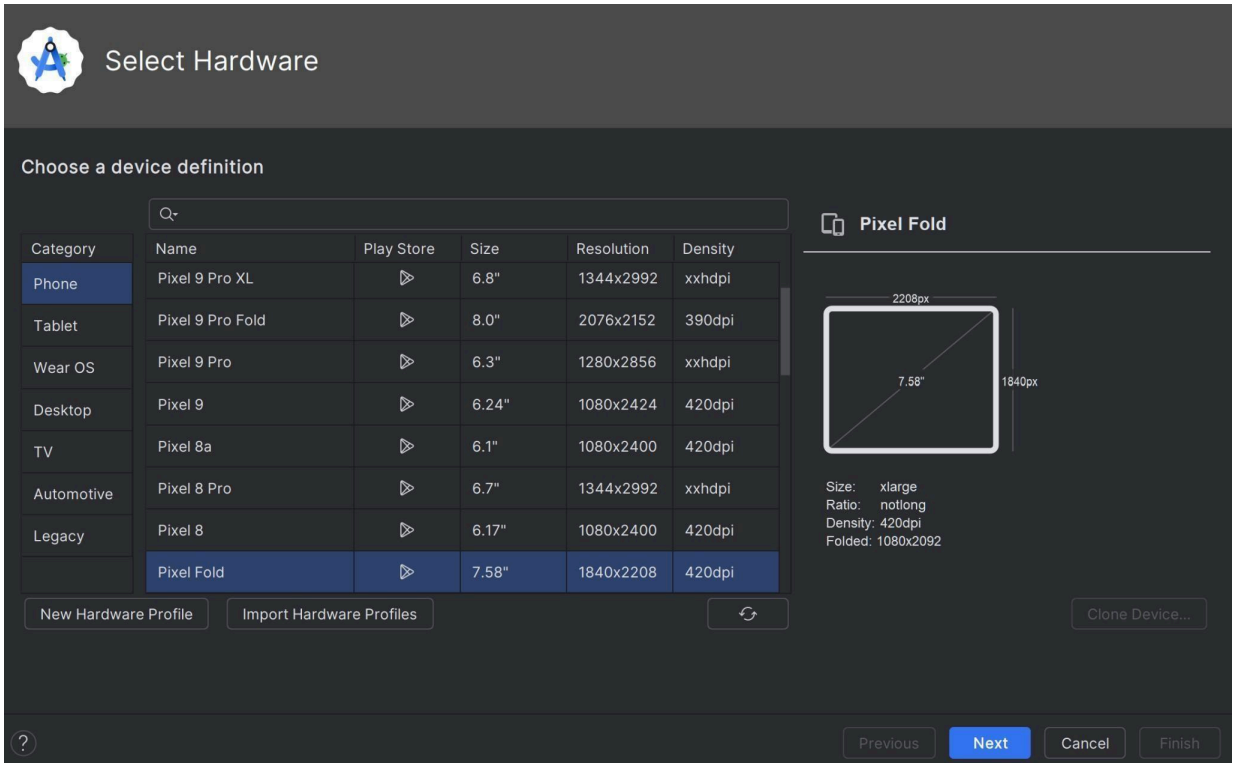
Step 8.4: - Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.



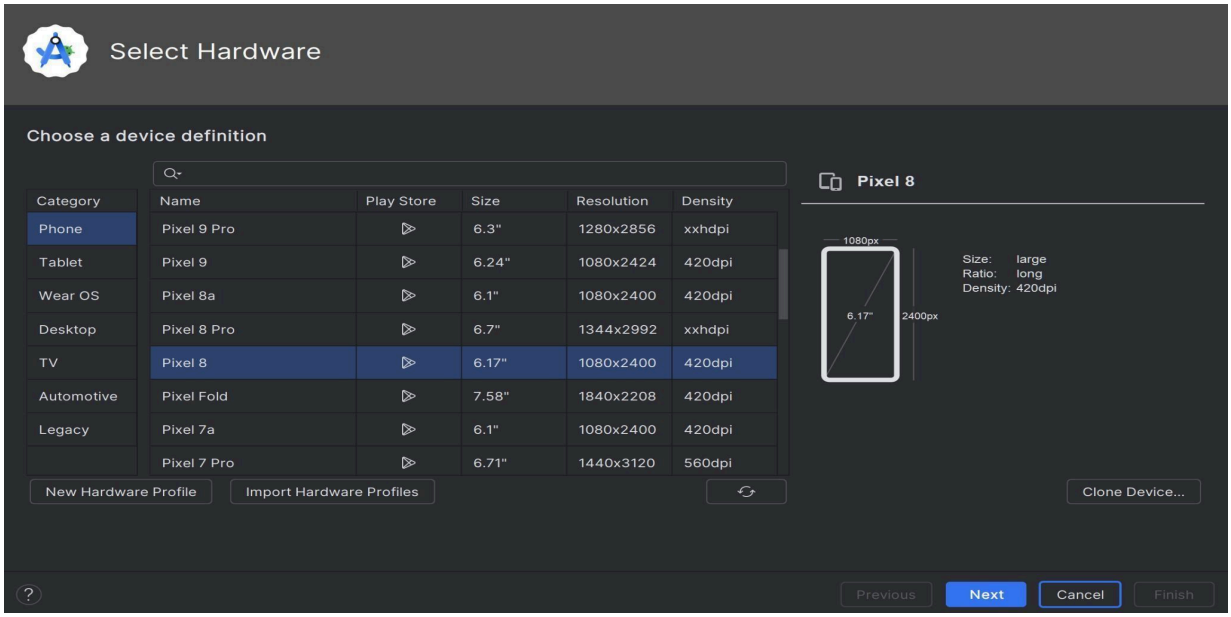
Step 8.5: - Go to Preferences > Appearance & Behavior > System Settings > Android SDK. Select the SDK Tools tab and check Android SDK Command-line Tools and Install it.

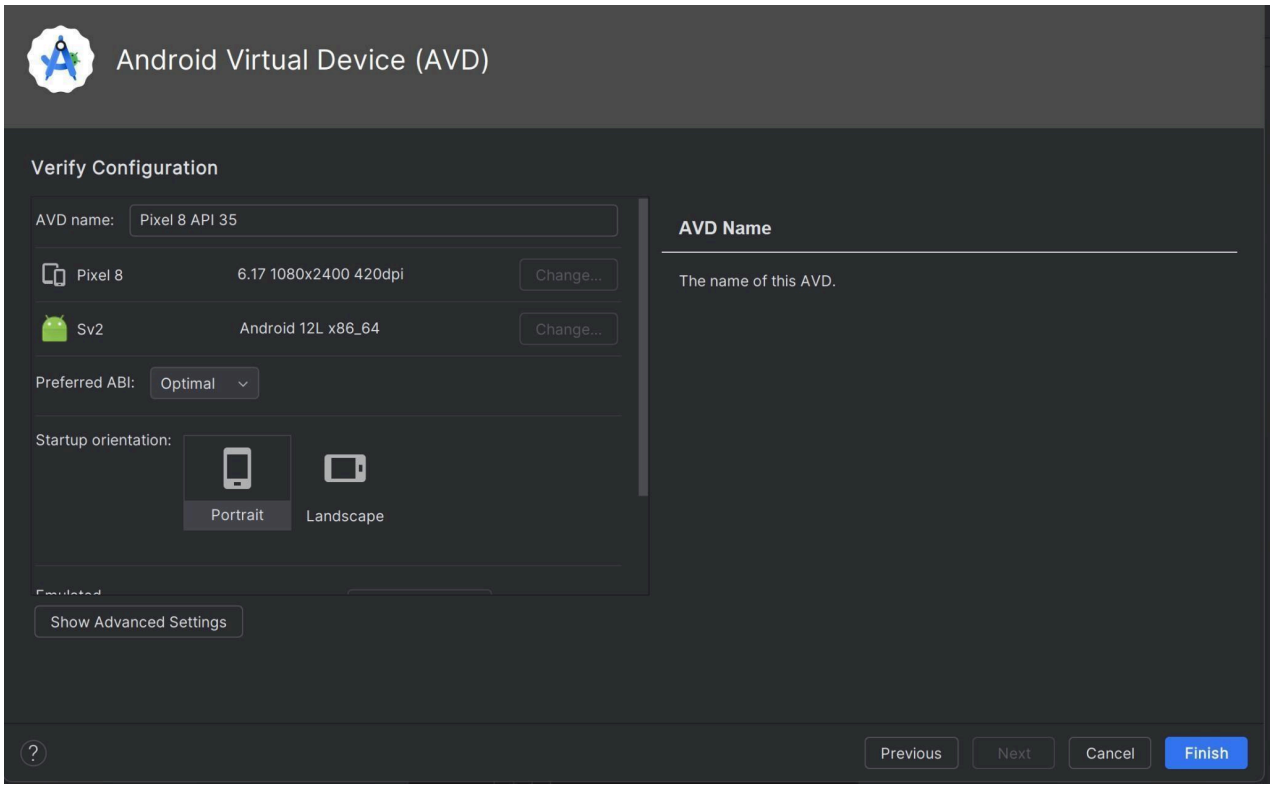


Step 10: - Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application

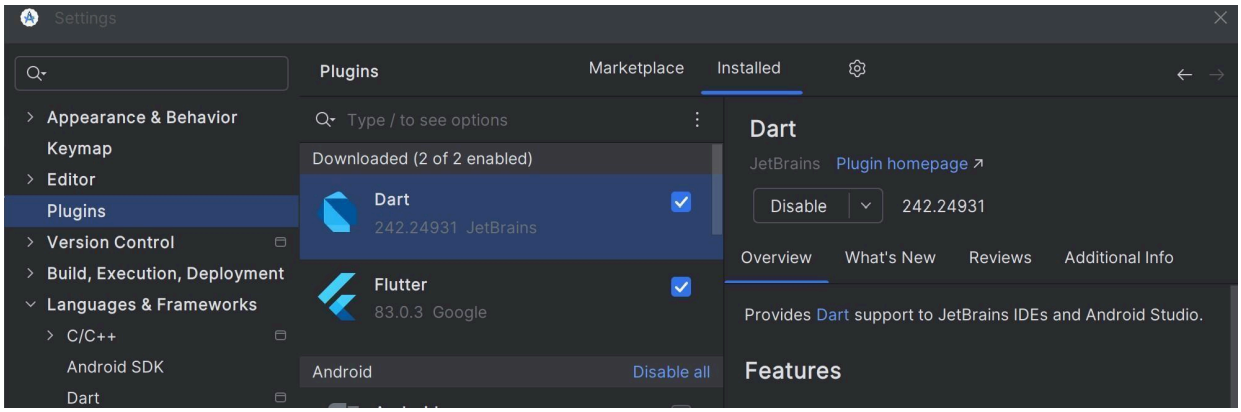


Step 10.1: - Open Android Studio and go to Tools > AVD Manager. Create a new virtual device.

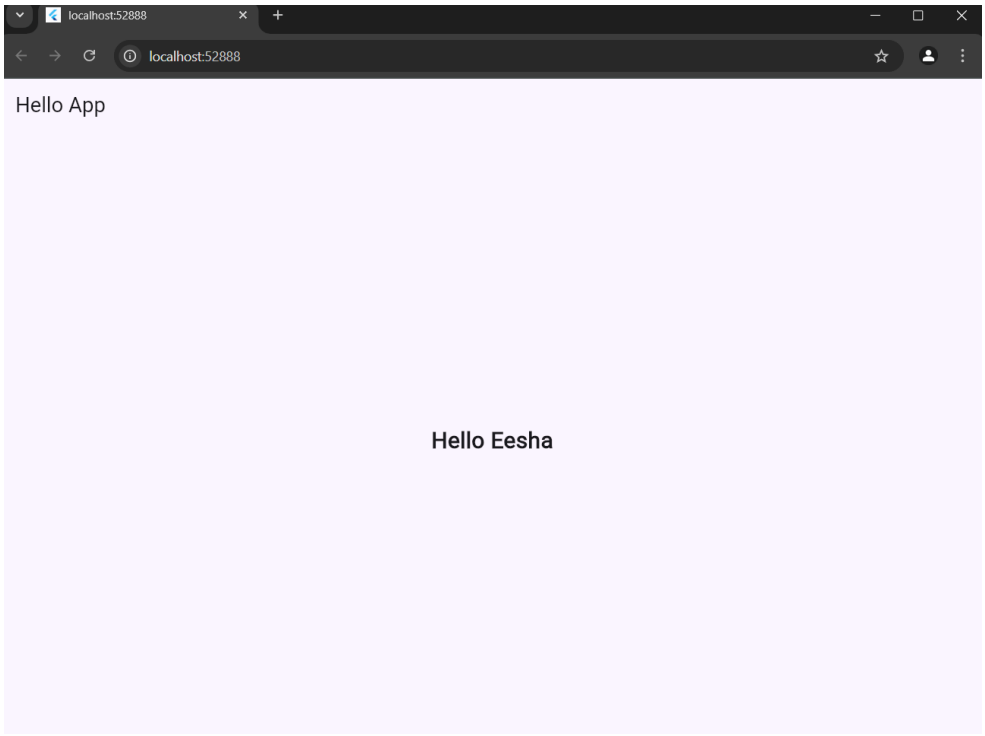




Step 11: - Now, install Flutter and Dart plugin for building Flutter application in Android Studio. These plugins provide a template to create a Flutter application, give an option to run and debug Flutter application in the Android Studio itself



Step 12: - Go to File > New Project > Create Flutter Project, then select the project name and location, and click Next to proceed




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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(title: Text('Hello App')),
        body: Center(
          child: Text(
            'Hello Eesha',
            style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
          ),
        ),
      ),
    );
  }
}
```

MAD & PWA Lab

Journal

Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

Eesha Chavan D15A-08

MPL EXP 2

Aim: To design Flutter UI by including common widgets

Theory:
Flutter UI is built using widgets, which are the basic building blocks of the application. Widgets in Flutter are classified into two types:

- Stateless Widgets – Do not change state once built. Example: Text, Icon, Container.
- Stateful Widgets – Can change state dynamically. Example: TextField, Checkbox, Switch

Common Widgets in Flutter:

1. Container
Used to hold other widgets and apply styling like padding, margin, and decoration.
2. Text
Displays text in the application with different styles. 3.Image
Displays images from assets, network, or memory.
4.Button Widgets
Includes ElevatedButton, TextButton, and OutlinedButton.
5.ListView
Used for displaying a scrollable list of widgets.
6.GridView
Used for displaying a collection of items in a grid format.

Syntax for Common Widgets:

1. Container

```
Widget Container(
width: 200,
  height: 100, color:
  Colors.blue, child:
  Center(
    child: Text('Hello Flutter!', style: TextStyle(color: Colors.white)),
  ),
)
```

2. Text

```
Widget Text(
  'Welcome to Flutter',
  style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
```

```
)
Image Widget
Image.asset('assets/flutter_logo.png', width: 100, height: 100)
```

3. ElevatedButton

```
Widget ElevatedButton(
onPressed: () {
   print('Button Pressed');
 },
 child: Text('Click Me'),
)
```

4. ListView

```
Widget ListView(
children: [
   ListTile(title: Text('Item 1')),
   ListTile(title: Text('Item 2')),
   ListTile(title: Text('Item 3')),
 ],
)
```

5. GridView

```
Widget
GridView.count(
crossAxisCount: 2, children: [
   Container(color: Colors.red, height: 100), Container(color:
   Colors.green, height: 100), Container(color: Colors.blue,
   height: 100), Container(color: Colors.yellow, height: 100),
 ],
)
```

Code:

main.dart:

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

```
import 'screens/login.dart';


void main() {
runApp(MyApp()); // Use MyApp instead of directly calling MaterialApp
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
debugShowCheckedModeBanner: false, // Removes the debug banner home:
      LoginPage(), // Set LoginPage as the initial screen
    );
  }
}
```

home.dart:

```
import 'package:flutter/material.dart';
import 'wishlist.dart'; // Import WishlistPage
import 'categories.dart'; // Import CategoriesPage
import 'profile.dart'; // Import ProfilePage
import 'cart.dart'; // Import CartPage


class HomePage extends StatefulWidget {
  @override
_HomePageState createState() => _HomePageState();
}


class _HomePageState extends State<HomePage> {
  int _selectedIndex = 0;
int _selectedCategory = 0;
int _currentBannerIndex = 0;
final PageController _bannerController = PageController();


  final List<String> _banners = [
```

```
];

final List<Map<String, String>> _bestsellers = [
  {
    'image': 'assets/bestseller1.jpg',
    'name': 'Premium Cotton T-Shirt',
    'category': 'Men's Wear',
    'mrp': '₹999',
    'tax': 'Incl. of all taxes'
  },
  {
    'image': 'assets/bestseller2.jpg',
    'name': 'Comfy Hoodie',
    'category': 'Winter Collection',
    'mrp': '₹1,499',
    'tax': 'Incl. of all taxes'
  },
  {
    'image': 'assets/bestseller3.jpg',
    'name': 'Classic Denim Jacket',
    'category': 'Jackets', 'mrp':
    '₹2,199',
    'tax': 'Incl. of all taxes'
  },
];

void _onItemTapped(int index) { if
  (index == 1) {
    // Navigate to Wishlist when Wishlist icon is tapped
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => WishListPage()),
    );
  } else if (index == 0) {
    // Navigate to Home when Home icon is tapped Navigator.pushReplacement
```

```
        context,
        MaterialPageRoute(builder: (context) => HomePage());
    } else if (index == 2) {
      // Navigate to Categories when Categories icon is tapped
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => CategoriesPage()),
      );
    } else if (index == 4) {
      // Navigate to Profile when Profile icon is tapped
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => ProfilePage()),
      );
    } else {
      setState(() {
        _selectedIndex = index;
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    double screenHeight = MediaQuery.of(context).size.height;
    double bannerHeight = screenHeight * 0.7;

    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 0,
        leading: IconButton(
          icon: Icon(Icons.menu, color: Colors.black, size: 28),
          onPressed: () {},
        ),
        title: Image.asset( 'assets/logo.jpg', height: 50,
```

```
28),
 centerTitle: true, actions: [
   IconButton(
     icon: Icon(Icons.search, color: Colors.black, size: 28), onPressed:
     {},
   ),
   IconButton(
     icon: Icon(Icons.shopping_cart, color: Colors.black, size:

     onPressed: () {
       Navigator.push( context,
         MaterialPageRoute(builder: (context) => CartPage()),
       );
     },
   ),
 ],
,
body: SingleChildScrollView( child: Column(
   crossAxisAlignment: CrossAxisAlignment.start, children: [
     Padding(
       padding: const EdgeInsets.symmetric(vertical: 10), child: Row(
         mainAxisAlignment: MainAxisAlignment.spaceEvenly,
         children: ['MEN', 'WOMEN',
'KIDS'].asMap().entries.map((entry) {
                  int index = entry.key;
                  String category = entry.value;
                  return GestureDetector(
                    onTap: () => setState(() => _selectedCategory =
index),
```

```
                          fontWeight: FontWeight.w500,
                          color: Colors.black,
                        ),
                      ),
                    if (_selectedCategory == index) Container(
                        margin: EdgeInsets.only(top: 4),
                        height: 3,
                        width: 50,
                        color: Colors.teal,
                      ),
                  ],
                ),
              );
            }).toList(),
          ),
        ),
        SizedBox(
          height: bannerHeight,
          child: PageView.builder(
            controller: _bannerController,
            onPageChanged: (index) => setState(() =>
_currentBannerIndex = index),
            itemCount: _banners.length,
            itemBuilder: (_, index) => Padding(
              padding: EdgeInsets.symmetric(horizontal: 8),
              child: ClipRRect(
                borderRadius: BorderRadius.circular(10), child:
                Image.asset(
                  _banners[index],
                  fit: BoxFit.cover,
                  width: double.infinity,
                ),
              ),
            ),
          ),
        ),
```

```
            SizedBox(height: 10)Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: List.generate(_banners.length, (index) {
                return Container(
                  margin: EdgeInsets.symmetric(horizontal: 4),
                  width: 10,
                  height: 10,
                  decoration: BoxDecoration( shape:
                    BoxShape.circle,
                    color: _currentBannerIndex == index ?
Colors.grey[700] : Colors.grey[400],
                  ),
                );
              }),
            ),
            SizedBox(height: 20), Padding(
              padding: EdgeInsets.symmetric(horizontal: 16),
              child: Text(
                "LATEST COLLECTION",
                style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold),
              ),
            ),
            SizedBox(height: 10), SizedBox(
              height: 150,
              child: ListView.builder(
                scrollDirection: Axis.horizontal,
                itemCount: _latestCollection.length,
                itemBuilder: (_, index) => Padding(
                  padding: EdgeInsets.symmetric(horizontal: 8),
                  child: ClipRRect(
                    borderRadius: BorderRadius.circular(10), child:
                    Image.asset(
                      _latestCollection[index],
                      width: 150,
                      height: 150,
```

```
                    fit: BoxFit.cover,
                  ),
                ),
              ),
            ),
          ),
          SizedBox(height: 20), Padding(
            padding: EdgeInsets.symmetric(horizontal: 16),
            child: Text(
              "SOULFUL BESTSELLERS",
              style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold),
            ),
          ),
          SizedBox(height: 10), SizedBox(
            height: 270,
            child: ListView.builder(
              scrollDirection: Axis.horizontal,
              itemCount: _bestsellers.length,
              itemBuilder: (_, index) => Padding(
                padding: EdgeInsets.symmetric(horizontal: 8),
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start, children:
                  [
                    ClipRRect(
                      borderRadius: BorderRadius.circular(10),
                      child: Image.asset(
                        _bestsellers[index]['image']!,
                        width: 120,
                        height: 180,
                        fit: BoxFit.cover,
                      ),
                    ),
                    SizedBox(height: 5),
                    Text(_bestsellers[index]['name']!,
                        style: TextStyle(fontSize: 14, fontWeight:
```

```
FontWeight.bold)),
                                        Text(_bestsellers[index]['category'
                                        ]!,
                                            style: TextStyle(fontSize: 12
Colors.grey)),
                                        Text(_bestsellers[index]['mrp']!,
                                        style:
TextStyle(fontSize: 14)),
                      Text(_bestsellers[index]['tax']!, style:
TextStyle(fontSize: 12)),
                    ],
                  ),
                ),
              ),
            ),
          ],
        ),
      ),
      bottomNavigationBar:
        BottomNavigationBar( currentIndex:
        _selectedIndex,
        onTap: _onItemTapped,
        selectedItemColor: Colors.teal,
        unselectedItemColor: Colors.black,
        items: [
          BottomNavigationBarItem(icon: Icon(Icons.home), label:
"Home"),
          BottomNavigationBarItem(icon: Icon(Icons.favorite), label:
"Wishlist"),
          BottomNavigationBarItem(icon: Icon(Icons.category), label:
"Categories"),
          BottomNavigationBarItem(icon: Icon(Icons.card_membership),
label: "Membership"),
          BottomNavigationBarItem(icon: Icon(Icons.person), label:
"Profile"),
        ],
      ),
    );
  }
```

register.dart:

```dart
import 'package:flutter/material.dart';
import 'login.dart'; // Importing login page for navigation

class RegisterPage extends StatefulWidget {
  @override
  _RegisterPageState createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  DateTime? selectedDate;
  String? selectedGender;

  @override
  Widget build(BuildContext context)
    { return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor:
        Colors.white, elevation: 0,
        leading: IconButton(
          icon: Icon(Icons.arrow_back, color: Colors.black),
          onPressed: () => Navigator.pushReplacement(
            context,
            MaterialPageRoute(builder: (context) => LoginPage()
          ),
        ),
      ),
      body: SingleChildScrollView(
        padding: EdgeInsets.symmetric(horizontal: 20),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
```

```
          ame Row(
children: [ Expanded(
child: Column(
crossAxisAlignment: CrossAxisAlignment.start, children: [
Text("First Name", style: TextStyle(fontSize: 14,
color: Colors.black)),
                                    TextField(
                                       decoration:
                                       InputDecoration(
                                       hintText: "First
                                       Name *",
                                       hintStyle: TextStyle(color:
                                       Colors.grey), border:
                                       OutlineInputBorder(),
                          ),
),
],
),
),
SizedBox(width: 10), Expanded(
child: Column(
crossAxisAlignment: CrossAxisAlignment.start, children: [
```

```
                                    Text("Last Name", style:
                                    TextStyle(fontSize: 14,
color: Colors.black)),

                                    TextField(
                                      decoration:
                                        InputDecoration(
                                        hintText: "Last Name
                                        *",
                                        hintStyle: TextStyle(color:
                                        Colors.grey), border:
                                        OutlineInputBorder(),
                          ),
                        ),
                      ],
                    ),
                  ),
                ],
              ),
              SizedBox(height: 15),

              // Mobile Number
              TextField(
                keyboardType: TextInputType.phone,
                decoration: InputDecoration(
                  hintText: "+91 *",
                  hintStyle: TextStyle(color: Colors.grey),
                  border: OutlineInputBorder(),
                ),
              ),
              SizedBox(height: 15),

              // Email ID
              Text("Email ID", style: TextStyle(fontSize: 14, color:
Colors.black)),
              TextField(
                keyboardType: TextInputType.emailAddress,
                decoration: InputDecoration(
```

```
                hintText: "Email ID *",
                hintStyle: TextStyle(color: Colors.grey),
                border: OutlineInputBorder(),
              ),SizedBox(height: 15),

            // Password
            Text("Password", style: TextStyle(fontSize: 14, color:
Colors.black)),
            TextField(
              obscureText: true,
              decoration: InputDecoration(
                hintText: "Password *",
                hintStyle: TextStyle(color: Colors.grey),
                border: OutlineInputBorder(),
                suffixIcon: Icon(Icons.visibility_off, color:
Colors.grey),
              ),
            ),
            SizedBox(height: 15),

            // Birthdate
            Text("Birthdate", style: TextStyle(fontSize: 14, color:
Colors.black)),
            TextField(
              readOnly: true,
              decoration: InputDecoration(
                hintText: "DD-MM-YYYY *",
                hintStyle: TextStyle(color: Colors.grey),
                border: OutlineInputBorder(),
                suffixIcon: Icon(Icons.calendar_today, color:
Colors.grey),
                                    ),
                                    onTap: () async {
                                      DateTime? pickedDate = await
                                        showDatePicker( context:
                                        context,
                                        initialDa
```

```
  te:                                    colorScheme:
  DateTime.                              ColorScheme.light(primary:
  now(),
  firstDate                              buttonTheme:
  :                                      ButtonThemeData(textTheme:
  DateTime(
  1900),
  lastDate: DateTime.now(),
  builder:
    (contex
    t,
    child)
    {
    return
    Theme(
      data:
        ThemeData.light().co
        pyWith(
        primaryColor:
        Colors.grey,
Colors.grey),

ButtonTextTheme.primary),
                      ),
                      child: child!,
                    );
                  },
                );

                if (pickedDate != null) {
                  setState(() {
                    selectedDate = pickedDate;
                  });
                }
              },
            ),
```

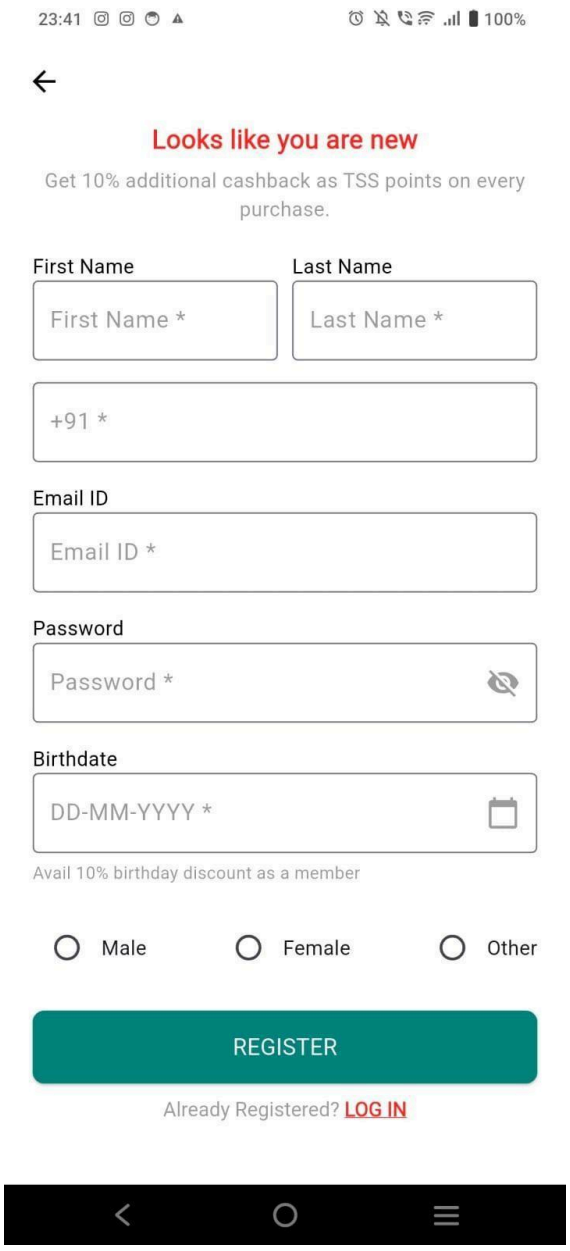
```
              SizedBox height: 5
```

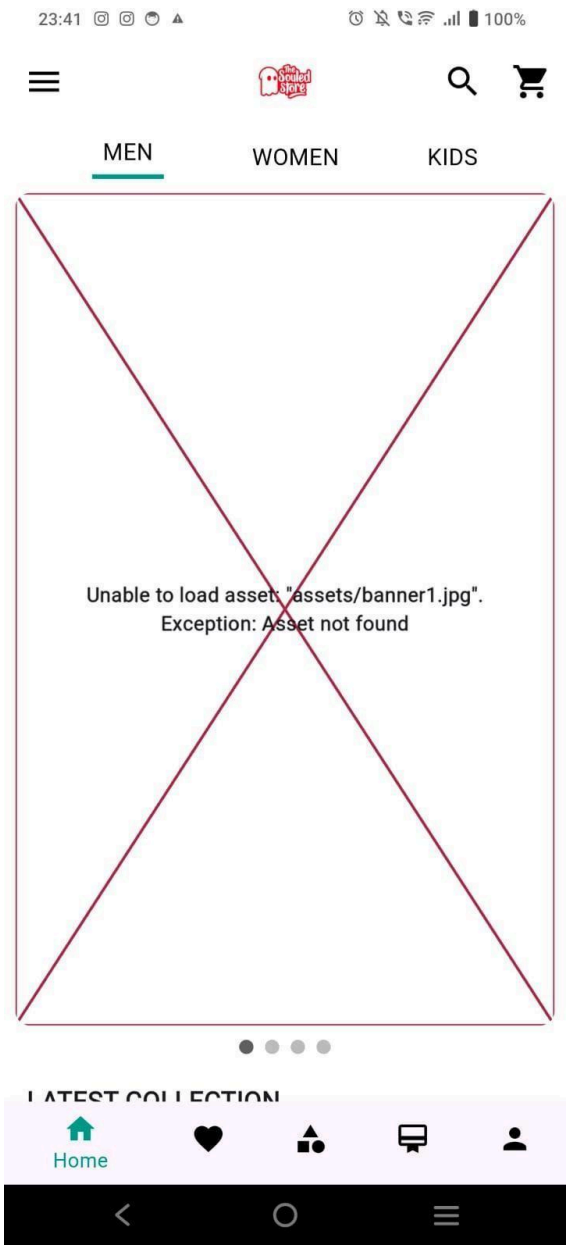
```
              // Birthday Discount Note Text(
                "Avail 10% birthday discount as a member",
                style: TextStyle(fontSize: 12, color: Colors.grey),
              ),
              SizedBox(height: 20),

              // Gender Selection Row(
                mainAxisAlignment: MainAxisAlignment.spaceBetween,
                children: [
                  _genderOption("Male"),
                  _genderOption("Female"),
                  _genderOption("Other"),
                ],
              ),
              SizedBox(height: 20),

              // Register Button
 Teal ColorizedBox(
  width: double.infinity, child:
  ElevatedButton(
    style: ElevatedButton.styleFrom(
      backgroundColor: Color(0xFF31827C), // New Darker

      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(8),
      ),
    ),
    onPressed: () {
      // Register button functionality
    },
    child: Padding(
      padding: EdgeInsets.symmetric(vertical: 14),
      child: Text("REGISTER", style: TextStyle(fontSize:
          16, color:
Colors.white)),SizedBox(height:
```

```
            10),

            // Already Registered? LOG IN Center(
              child: GestureDetector(
                onTap: () {
                  Navigator.pushReplacement(
                    context,
                    MaterialPageRoute(builder: (context) =>
LoginPage()),
                                        );
                                      },
                                      c
                                        h
                                        i
                                        l
                                        d
                                        (
                                          text: "Already Registered? ",
                                          style: TextStyle(color:
                                          Colors.grey, fontSize: 14),
                                          children: [
                                            TextSpan(
                                              text: "LOG IN",
```

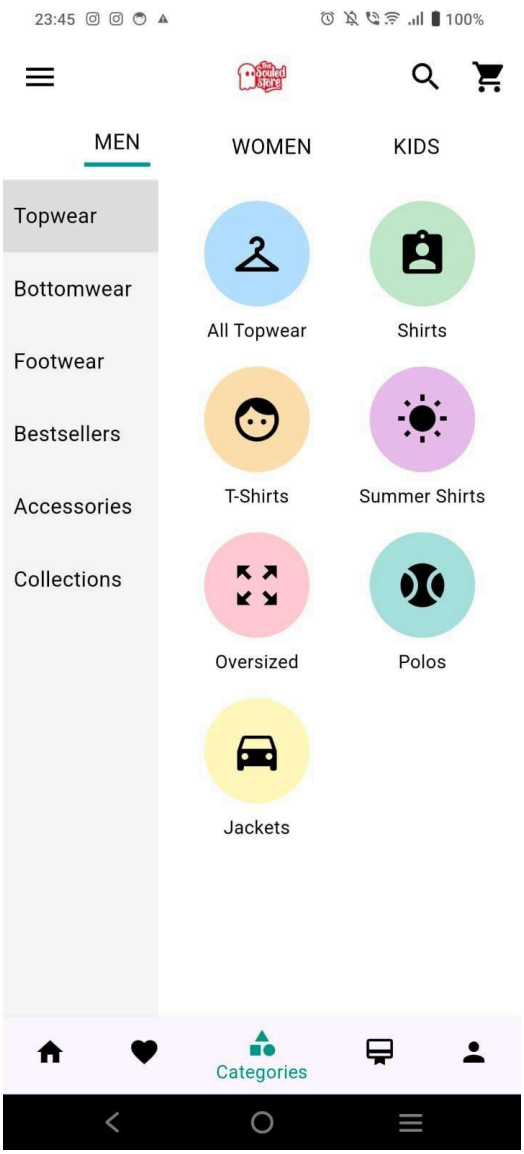
```
                        style: TextStyle(
                         color: Colors.red,
                         fontWeight: FontWeight.bold,
                         decoration: TextDecoration.underline,
                        ),
                      ),
                    ],
                  ),
                ),
              ),
            ),
            SizedBox(height: 20),
          ],
        ),
      ),
    );
  }

  // Gender Option Widget
  Widget _genderOption(String text) {
    return Row(
      children: [ Radio(
          activeColor: Color(0xFF31827C), // New Teal Color for
Selected Option
          value: text,
          groupValue: selectedGender,
          onChanged: (value) {
            setState(() {
              selectedGender = value.toString();
            });
          },
        ),
        Text(text, style: TextStyle(fontSize: 14)),
      ],
    );
  }
```

Output:

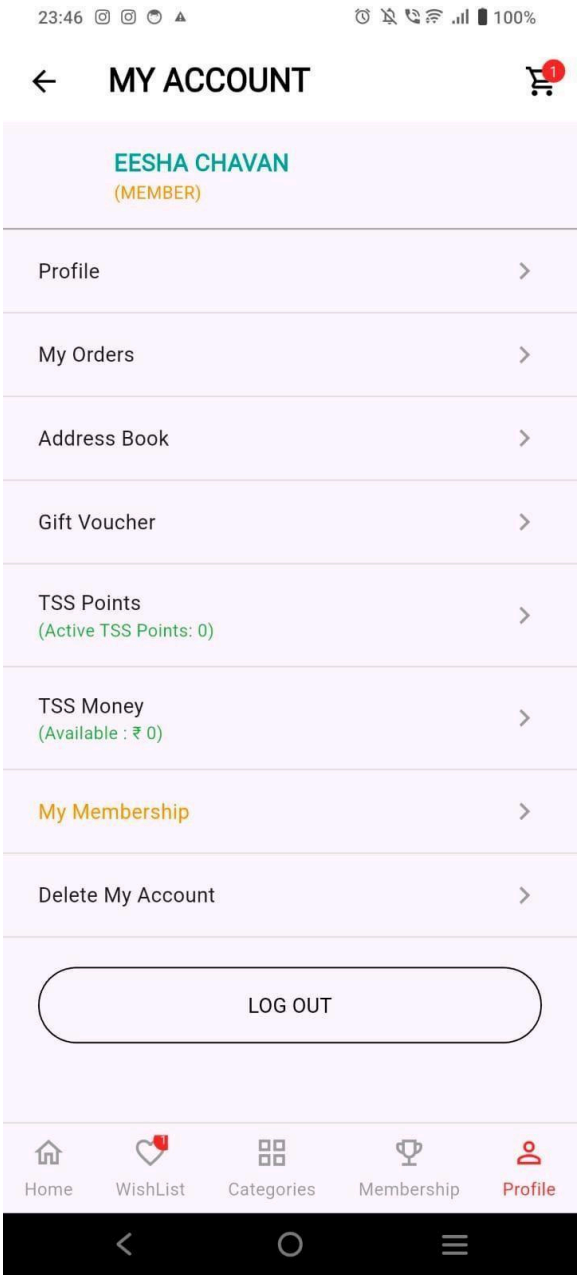

23:41
100%
Looks like you are new
Get 10% additional cashback as TSS points on every purchase.
First Name
Last Name
First Name *
Last Name *
+91 *
Email ID
Email ID *
Password
Password *
Birthdate
DD-MM-YYYY *
Avail 10% birthday discount as a member
Male
Female
Other
REGISTER
Already Registered? LOG IN

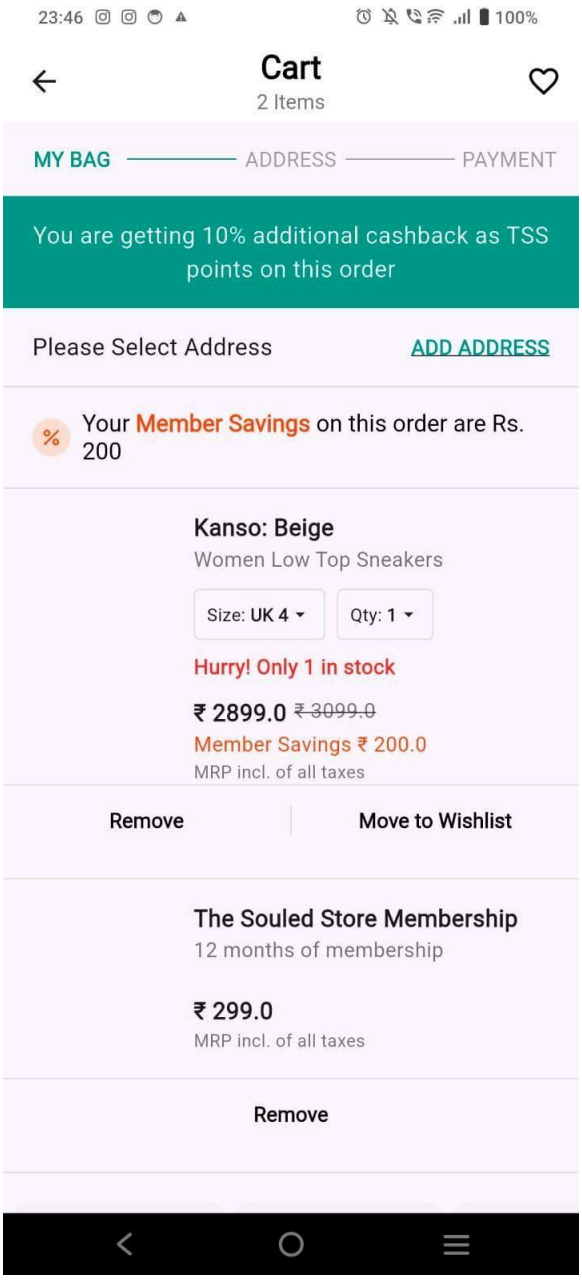

23:41
100%
MEN
WOMEN
KIDS
Unable to load asset: "assets/banner1.jpg".
Exception: Asset not found
Home

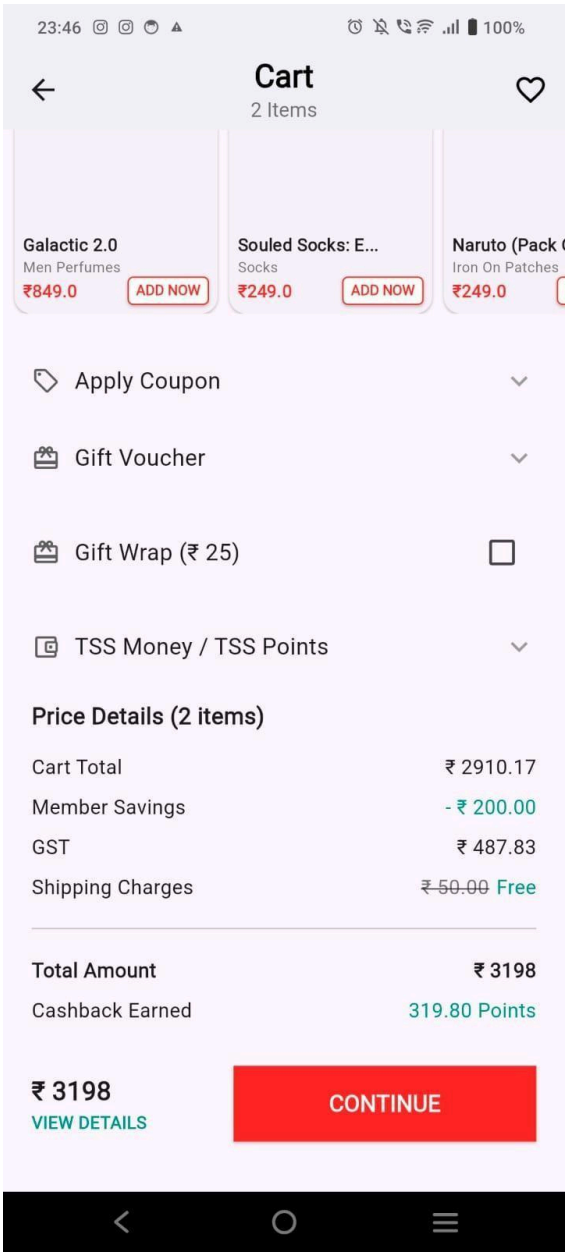
Project Title: The Souled Store Clone **Roll No. 08**



Project Title: The Souled Store Clone **Roll No. 08**






23:46
100%
Cart
2 Items
Galactic 2.0
Men Perfumes
₹849.0
ADD NOW
Souled Socks: E...
Socks
₹249.0
ADD NOW
Naruto (Pack
Iron On Patches
₹249.0
Apply Coupon
Gift Voucher
Gift Wrap (₹ 25)
TSS Money / TSS Points
Price Details (2 items)
Cart Total
₹ 2910.17
Member Savings
- ₹ 200.00
GST
₹ 487.83
Shipping Charges
₹ 50.00 Free
Total Amount
₹ 3198
Cashback Earned
319.80 Points
₹ 3198
VIEW DETAILS
CONTINUE


23:50
100%
WishList
Image
Textured Shirt: Sky Blue
MOVETOCART
Home
WishList

MAD & PWA Lab
Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL Experiment 3

Name: Eesha Chavan **Class:** D15A **Roll no:** 8

Aim: To include icons, images, Fonts in Flutter app.

Theory:

Incorporating icons, images, and custom fonts in a Flutter application enhances the visual appeal and improves the user experience. Flutter provides different ways to add these elements:

1. Icons:

Icons can be added using the built-in `Icons` class with the `Icon` widget. Custom icons can also be used via the `flutter_launcher_icons` package.

2. Images:

Images can be displayed in two ways:

- **From assets**: Images stored locally in the app's assets folder can be loaded using the `Image.asset()` method.
- **From the internet**: Images can be fetched dynamically from a URL using the `Image.network()` method.

Examples:

- **Loading an image from assets:**

```
Image.asset(
  'assets/images/sample.png'
  , fit: BoxFit.cover,
  width: double.infinity,
)
```

- **Loading an image from a URL:**

```
Image.network(
  'https://example.com/sample.jpg',
width : double.infinity,
fit : BoxFit.cover,
```

```
)
```

3. Fonts:

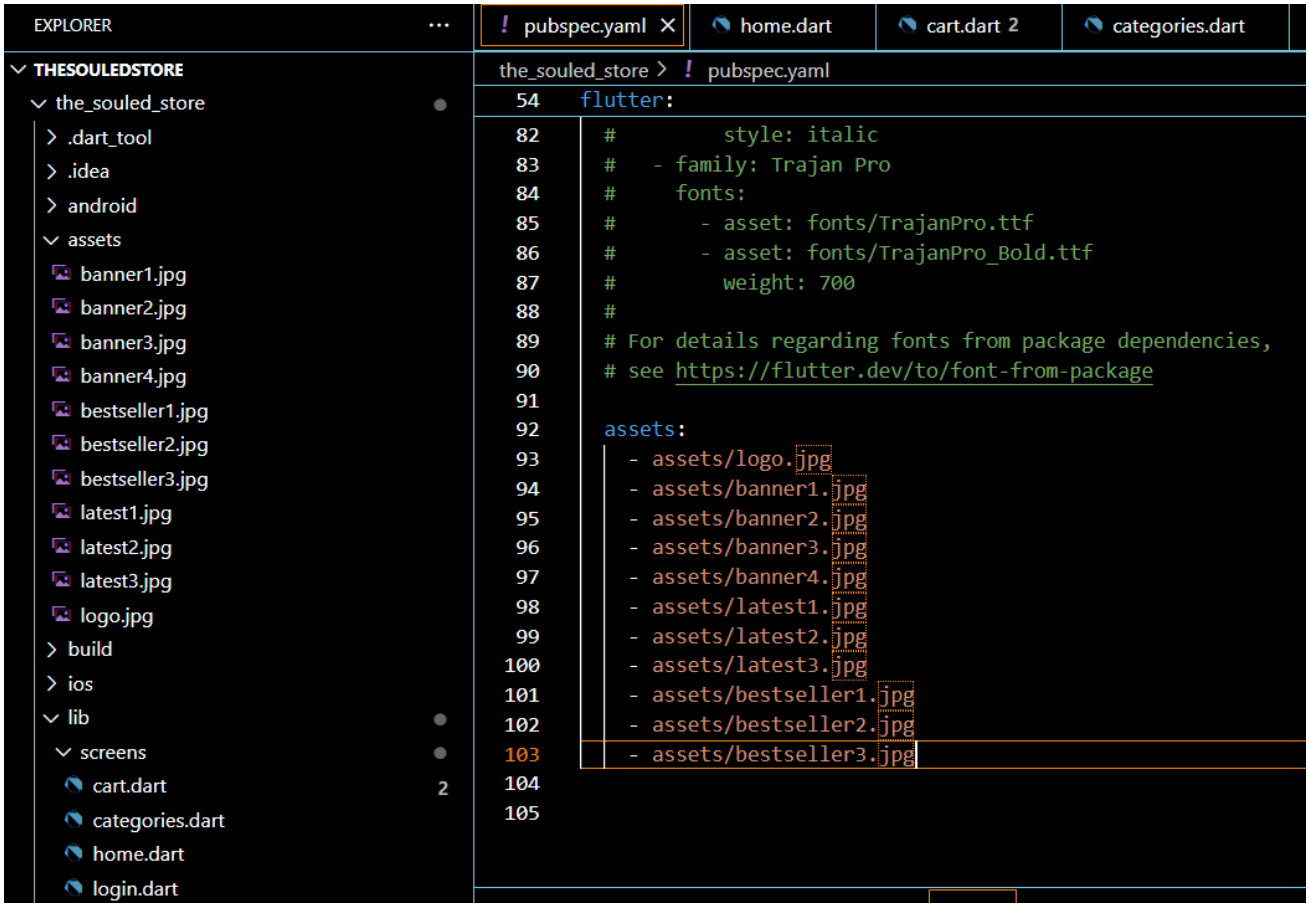
Custom fonts improve typography and branding. To add custom fonts, font files must be placed in the `assets/fonts/` directory and declared in `pubspec.yaml` under the `flutter` section. Using `TextStyle` with the `fontFamily` property applies the custom font to text elements.

Steps:

Step 1: Create an `assets` folder inside the project directory. Inside the `assets` folder, create an `images` folder and add the required images.

Step 2: Open `pubspec.yaml` and add the following under the `flutter` section:

```
flutter:
  assets
  :
    - assets/images/
```



Step 3: Run the following command in the terminal to apply the changes:

```
flutter pub get
```

Code:

home.dart:

```
import 'package:flutter/material.dart';
import 'wishlist.dart'; // Import WishlistPage
import 'categories.dart'; // Import CategoriesPage
import 'profile.dart'; // Import ProfilePage
import 'cart.dart'; // Import CartPage

class HomePage extends StatefulWidget {
  @override
  _HomePageState createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  int _selectedIndex = 0;
  int _selectedCategory = 0;
  int _currentBannerIndex = 0;
  final PageController _bannerController = PageController();

  final  List<String>  _banners  =
    [ 'assets/banner1.jpg',
    'assets/banner2.jpg'
    ,
    'assets/banner3.jpg'
    ,
    'assets/banner4.jpg'
    ,
  ];

  final List<String> _latestCollection = [
    'assets/latest1.jpg',
    'assets/latest2.jpg',
    'assets/latest3.jpg',
  ];

  final List Map String  String   _bestsellers =
```

```
    {
      'image': 'assets/bestseller1.jpg','name': 'Premium Cotton Shirt',
      'category': 'Womens shirts',
      'mrp': '₹999',
      'tax': 'Incl. of all taxes'
    },
    {
      'image': 'assets/bestseller2.jpg',
      'name': 'Comfy Tees',
      'category': 'Womens tshirts',
      'mrp': '₹499',
      'tax': 'Incl. of all taxes'
    },
    {
      'image': 'assets/bestseller3.jpg',
      'name': 'Trending sneakers',
      'category': 'Sneakers',
      'mrp': '₹2,199',
      'tax': 'Incl. of all taxes'
    },
  ];


  void _onItemTapped(int index) {
    if (index == 1) {
      // Navigate to Wishlist when Wishlist icon is tapped
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => WishListPage()),
      );
    } else if (index == 0) {
      // Navigate to Home when Home icon is tapped
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => HomePage()),
      );
    } else if (index == 2) {
```

```
    // Navigate to Categories when Categories icon is tapped
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => CategoriesPage()),
    );
  } else if (index == 4) {
    // Navigate to Profile when Profile icon is tapped
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => ProfilePage()),
    );
  } else {
    setState(() {
      _selectedIndex = index;
    });
  }
}

@override
Widget build(BuildContext context) {
  double screenHeight = MediaQuery.of(context).size.height;
  double bannerHeight = screenHeight * 0.7;

  return Scaffold(
    backgroundColor: Colors.white,
    appBar: AppBar(
      backgroundColor: Colors.white,
      elevation: 0,
      leading: IconButton(
        icon: Icon(Icons.menu, color: Colors.black, size: 28),
        onPressed: () {},
      ),
      title: Image.asset(
        'assets/logo.jpg',
        height: 50,
```

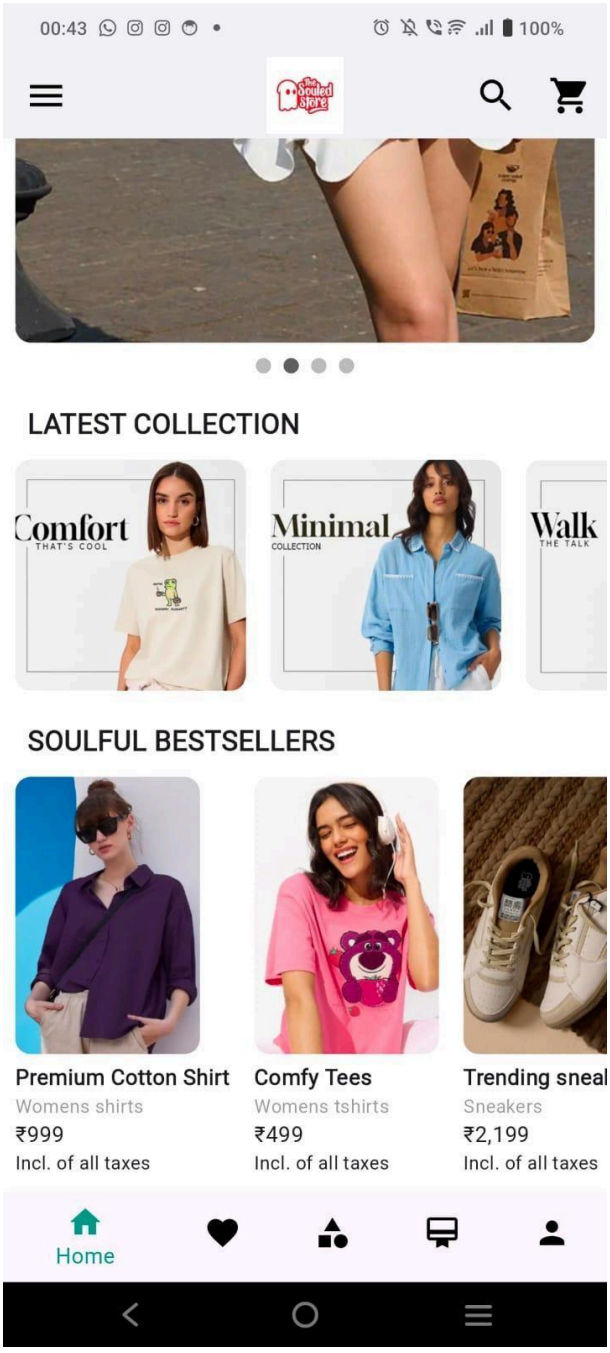
```
        ),
        centerTitle: true,
        actions: [
          IconButton(
            icon: Icon(Icons.search, color: Colors.black, size: 28),
            onPressed: () {},
          ),
          IconButton(
            icon: Icon(Icons.shopping_cart, color: Colors.black, size: 28),
            onPressed: () {
              Navigator.push(
                context,
                MaterialPageRoute(builder: (context) => CartPage()),
              );
            },
          ),
        ],
      ),
      body: SingleChildScrollView(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Padding(
              padding: const EdgeInsets.symmetric(vertical: 10),
              child: Row(
                mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                children: ['MEN', 'WOMEN',
'KIDS'].asMap().entries.map((entry) {
                  int index = entry.key;
                  String category = entry.value;
                  return GestureDetector(
                    onTap: () => setState(() => _selectedCategory = index),
                    child: Column(
                      children: [
                        Text(
```

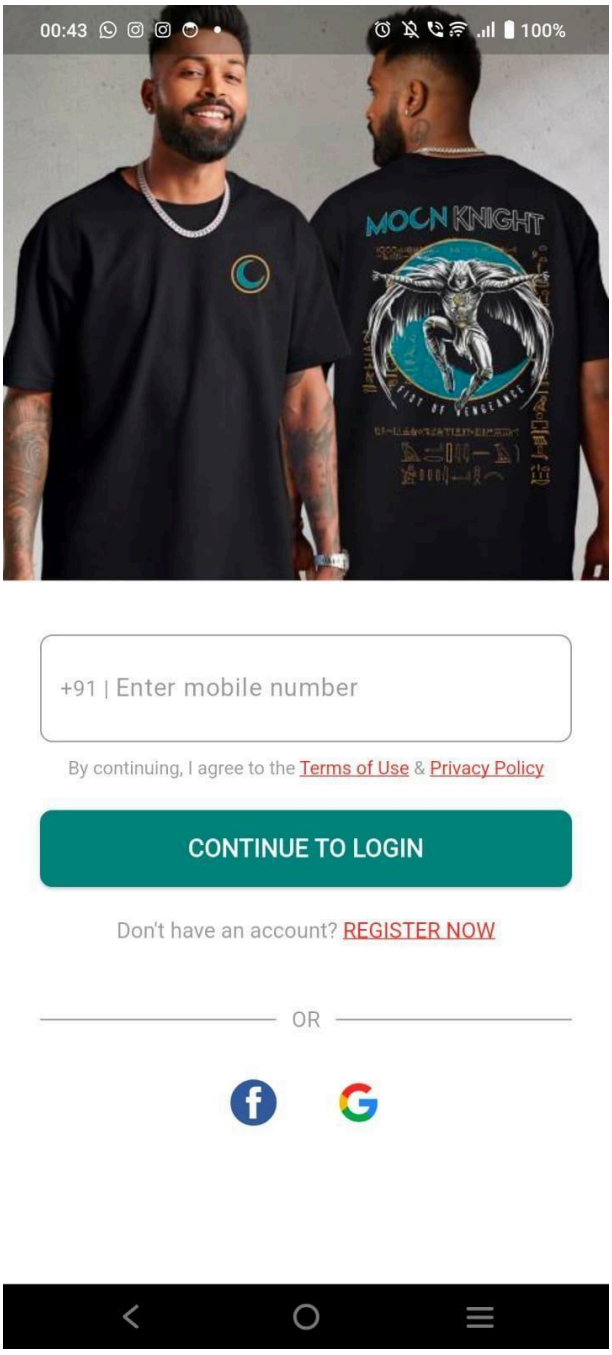
```
                        category,
                        style: TextStyle(
                          fontSize: 16,
                          fontWeight: FontWeight.w500,
                          color: Colors.black,
                        ),
                      ),
                      if (_selectedCategory == index)
                        Container(
                          margin: EdgeInsets.only(top: 4),
                          height: 3,
                          width: 50,
                          color: Colors.teal,
                        ),
                    ],
                  ),
                );
                                      }).toList(),
                                  ),
                                ),
                                SizedBox(
                                  height: bannerHeight,
                                  child: PageView.builder(
                                    controller: _bannerController,
                                    onPageChanged: (index) =>
                                    setState(() => _currentBannerIndex
= index),
                                    itemCount: _banners.length,
                                    itemBuilder: (_, index) =>
                                    Padding(
                                      padding:
                                      EdgeInsets.symmetric(
                                      horizontal: 8),
                                      child: ClipRRect(
                                        borderRadius:
                                        BorderRadius.cir
                                                    cular 10
```

```
        child:                                  ),
        Image.asset(                          ),
          _banners[index],                    SizedBox(height: 5),
          fit: BoxFit.cover,                  Text(_bestsellers[index]['name']!,
          width: double.infinity),                style: TextStyle(fontSize: 14,
                                                  fontWeight:
                ),
              ),                              Text(_bestsellers[index]['category']!
            ),                                    style: TextStyle(fontSize: 12,
          ),                                      color:
          S
                                              Text(_bestsellers[index]['mrp']!,
                                              style:



FontWeight.bold)),



Colors.grey)),

TextStyle(fontSize: 14)),
                    Text(_bestsellers[index]['tax']!, style:
TextStyle(fontSize: 12)),
                  ],
                ),
              ),
            ),
          ),
        ],
      ),
    ),
    bottomNavigationBar: BottomNavigationBar(
      currentIndex: _selectedIndex,
      onTap: _onItemTapped,
      selectedItemColor: Colors.teal,
      unselectedItemColor: Colors.black,
      items: [
```

```
          BottomNavigationBarItem icon: Icon Icons home   label: "Home"
          BottomNavigationBarItem icon: Icon Icons favorite   label:
"Wishlist"
          BottomNavigationBarItem icon: Icon Icons category   label:
"Categories"),
          BottomNavigationBarItem(icon: Icon(Icons.card_membership), label:
"Membership"),
          BottomNavigationBarItem(icon: Icon(Icons.person), label:
"Profile"),
        ],
      ),
    );
  }
}
```

Output:


00:43
100%
LATEST COLLECTION
Comfort
THAT'S COOL
Minimal
COLLECTION
Walk
THE TALK
SOULFUL BESTSELLERS
Premium Cotton Shirt
Womens shirts
₹999
Incl. of all taxes
Comfy Tees
Womens tshirts
₹499
Incl. of all taxes
Trending sneal
Sneakers
₹2,199
Incl. of all taxes
Home


00:43
100%
MOON KNIGHT
+91 | Enter mobile number
By continuing, I agree to the Terms of Use & Privacy Policy
CONTINUE TO LOGIN
Don't have an account? REGISTER NOW
OR

MAD & PWA Lab
Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL Experiment 4

Name: Eesha Chavan **Class:** D15A **Roll no:** 8

Aim: To create an interactive Form using form widget

Theory:

Creating an interactive form in Flutter requires using form-related widgets to collect and validate user input efficiently. The `Form` widget, combined with `TextFormField`, provides a structured way to manage input fields. Various input widgets like `TextField`, `DropdownButton`, `Checkbox`, `Radio`, and `Switch` allow users to enter data in different formats.

Validation and state management can be handled using the `GlobalKey<FormState>` to validate inputs before submission. Wrapping the form in a `SingleChildScrollView` ensures smooth scrolling when multiple fields are present.

A `RaisedButton` (deprecated) or `ElevatedButton` can trigger validation and submission logic. To enhance usability and create a responsive experience, proper padding, spacing, and `InputDecoration` should be applied.

Steps:

Step 1: Create a new Flutter project or open an existing one.

Step 2: Define a `Form` widget inside a StatefulWidget to manage user input.

Step 3: Use `TextFormField` for text input fields with validation logic.

Step 4: Include other input widgets such as `DropdownButton`, `Checkbox`, `Radio`, and `Switch` for additional user selections.

Step 5: Wrap the form inside a `SingleChildScrollView` to ensure smooth scrolling.

Step 6: Implement an `ElevatedButton` to trigger form validation and submission.

Step 7: Use `GlobalKey<FormState>` to manage form validation.

Code:
register.dart:

```dart
import 'package:flutter/material.dart';
import 'login.dart'; // Importing login page for navigation


class RegisterPage extends StatefulWidget {
```

```
  @override
  _RegisterPageState createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  DateTime? selectedDate;
  String? selectedGender;

  // Form Key for validation
  final _formKey = GlobalKey<FormState>();

  // Controllers for text fields
  final TextEditingController _firstNameController =
TextEditingController();
  final TextEditingController _lastNameController =
TextEditingController();
  final TextEditingController _phoneController = TextEditingController();
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController =
TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 0,
        leading: IconButton(
          icon: Icon(Icons.arrow_back, color: Colors.black),
          onPressed: () => Navigator.pushReplacement(
            context,
            MaterialPageRoute(builder: (context) => LoginPage()),
          ),
        ),
```

```
      ),
      body: SingleChildScrollView(
        padding: EdgeInsets.symmetric(horizontal: 20),
        child: Form(
          key: _formKey, // Assign the form key
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Center(
                child: Column(
                  children: [
                    Text(
                      "Looks like you are new",
                      style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold, color: Colors.red),
                    ),
                    SizedBox(height: 5),
                    Text(
                      "Get 10% additional cashback as TSS points on every
purchase.",
                                                textAlign: TextAlign.center
                                                style: TextStyle(fontSize: 14
                                                color: Colors.grey),
                                              ),
                                            ],
                                          ),
                                        ),
              SizedBox(height: 20),

              // First Name & Last Name
              Row(
                children: [
                  Expanded(
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
```

```
                                    Text("First Name", style:
                                    TextStyle(fontSize: 14,
color: Colors.black)),

                                    TextFormField(
                                      controller:
                                      _firstNameController,
                                      decoration:
                                      InputDecoration(
                                        hintText: "First Name *",
                                        hintStyle:
                                        TextStyle(color:
                                        Colors.grey), border:
                                        OutlineInputBorder(),
                                      ),
                                      validator: (value) {
                                        if (value == null ||
                                          value.isEmpty) {
                                          return 'First Name is
                                          required';
                        }
                        return null;
                      },
                    ),
                  ],
                ),
              ),
              SizedBox(width: 10),
              Expanded(
                child: Column(
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    Text("Last Name", style: TextStyle(fontSize: 14,
color: Colors.black)),
                                    TextFormField(
                                      controller:
                                      _lastNameController,
```

```
decoration:
InputDecoration
  hintText: "Last Name *"
  hintStyle:
  TextStyle color:
  Colors grey   border:
  OutlineInputBorder

validator:  value
  if  value == null ||
  value isEmpty
```

```
                            return 'Last Name is required';
                          }
                          return null;
                        },
                      ),
                    ],
                  ),
                ),
              ],
            ),
            SizedBox(height: 15),

            // Mobile Number
            TextFormField(
              controller: _phoneController,
              keyboardType: TextInputType.phone,
              decoration: InputDecoration(
                hintText: "+91 *",
                hintStyle: TextStyle(color: Colors.grey),
                border: OutlineInputBorder(),
              ),
              validator: (value) {
                if (value == null || value.isEmpty) {
                  return 'Phone number is required';
                } else if (value.length != 10) {
                  return 'Phone number must be 10 digits';
                }
                return null;
              },
            ),
            SizedBox(height: 15),

            // Email ID
            Text("Email ID", style: TextStyle(fontSize: 14, color:
Colors.black)),
```

```
              TextFormField(
                controller: _emailController,
                keyboardType: TextInputType.emailAddress,
                decoration: InputDecoration(
                  hintText: "Email ID *",
                  hintStyle: TextStyle(color: Colors.grey),
                  border: OutlineInputBorder(),
                ),
                validator: (value) {
                  if (value == null || value.isEmpty) {
                    return 'Email is required';
                  } else if
(!RegExp(r'^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$').hasMatch(value)) {
                    return 'Enter a valid email address';
                  }
                  return null;
                },
              ),
              SizedBox(height: 15),

              // Password
              Text("Password", style: TextStyle(fontSize: 14, color:
Colors.black)),
              TextFormField(
                controller: _passwordController,
                obscureText: true,
                decoration: InputDecoration(
                  hintText: "Password *",
                  hintStyle: TextStyle(color: Colors.grey),
                  border: OutlineInputBorder(),
                  suffixIcon: Icon(Icons.visibility_off, color:
Colors.grey),
                                    ),
                                    validator: (value) {
                                      if (value == null || value.isEmpty
                                      {
```

```
                return 'Password is required'

            return null


        SizedBox height: 15

        // Birthdate
        Text "Birthdate"  style: TextStyle fontSize: 14  color:
Colors black
        TextFormField
          readOnly: true
          decoration: InputDecoration
            hintText: "DD-MM-YYYY *"
            hintStyle: TextStyle color: Colors grey
            border: OutlineInputBorder
            suffixIcon: Icon Icons calendar_today  color:
Colors grey
                                    onTap:    async
                                      DateTime? pickedDate =
                                        await
                                        showDatePicker
                                        context: context
                                        initia
                                        lDate:
                                        DateTi
                                        me now

                                        firstD
                                        ate:
Colors grey
                                        DateTi
                                        me 190
                                        0
                                        lastDate: DateTime now
                                        builde
```

```
      r:                                          e(
      (con                                          data:
      text                                            ThemeData.ligh
      ,                                               t().copyWith(
      chil                                            primaryColor:
      d) {                                            Colors.grey,
      retu                                            colorScheme:
      rn                                              ColorScheme.light(prima
      Them                                            ry:
                                                      buttonTheme:
                                                      ButtonThemeData(textThe
                                                      me:
ButtonTextTheme.primary),
                        ),
                        child: child!,
                      );
```

```
            },
          );

          if (pickedDate != null) {
            setState(() {
              selectedDate = pickedDate;
            });
          }
        },
        validator: (value) {
          if (selectedDate == null) {
            return 'Birthdate is required';
          }
          return null;
        },
      ),
      SizedBox(height: 5),

      // Birthday Discount Note
      Text(
        "Avail 10% birthday discount as a member",
        style: TextStyle(fontSize: 12, color: Colors.grey),
      ),
      SizedBox(height: 20),

      // Gender Selection
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: [
          _genderOption("Male"),
          _genderOption("Female"),
          _genderOption("Other"),
        ],
      ),
      SizedBox(height: 20),
```

```
Color
//RegisterButton
SizedBox
width:dou
```

```
b                                   e
l                                   v
e                                   a
.                                   t
i                                   e
n                                   d
f                                   B
i                                   u
n                                   t
i                                   t
t                                   o
y                                   n
,                                   (
c                                     style:
h                                     ElevatedButton.styleFrom(
i                                       backgroundColor:
l                                       Color(0xFF31827C), // New
d                                       Darker Teal
:                                       shape:
E                                       RoundedRectangleBorder(
l                                         borderRadius:
                                          BorderRadius.circular(8

                  ),
                ),
                onPressed: () {
                  // Validate the form
                  if (_formKey.currentState!.validate()) {
                    // If the form is valid, proceed with registration
                    print("Registration Successful");
                    // Add your registration logic here
                  }
                },
                child: Padding(
                  padding: EdgeInsets.symmetric(vertical: 14),
                  child: Text("REGISTER", style: TextStyle(fontSize: 16,
color: Colors.white))
```

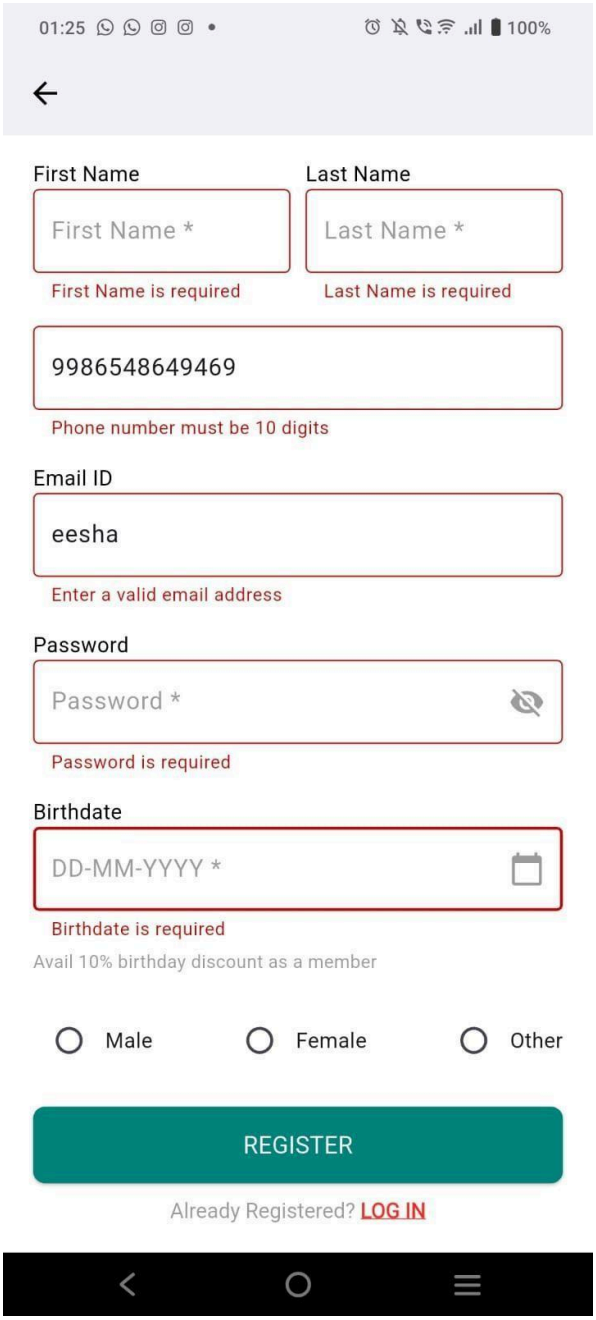
```
SizedBox height: 10

// Already Registered? LOG IN
Center
  child: GestureDetector
    onTap:
      Navigator pushReplacement
        context
```

```
                                                MaterialPageRoute(builder:
                                                (context) =>
LoginPage()),

                                              );
                                            },
                  child: RichText(
                    text: TextSpan(
                      text: "Already Registered? ",
                      style: TextStyle(color: Colors.grey, fontSize: 14),
                      children: [
                        TextSpan(
                          text: "LOG IN",
                          style: TextStyle(
                            color: Colors.red,
                            fontWeight: FontWeight.bold,
                            decoration: TextDecoration.underline,
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
              ),
              SizedBox(height: 20),
            ],
          ),
        ),
      ),
    );
  }

  // Gender Option Widget
  Widget _genderOption(String text) {
    return Row(
      children: [
        Radio(
```

```
          activeColor: Color(0xFF31827C), // New Teal Color for Selected
Option
          value: text,
          groupValue: selectedGender,
          onChanged: (value) {
            setState(() {
              selectedGender = value.toString();
            });
          },
        ),
        Text(text, style: TextStyle(fontSize: 14)),
      ],
    );
  }
}
```

Output:

Register Page:



MAD & PWA Lab
Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL Experiment 5

Name: Eesha Chavan **Class:** D15A **Roll no:** 8

Aim: To apply navigation, routing and gestures in Flutter App.

Theory:

Navigation & Routing:
Flutter provides multiple ways to navigate between screens (pages):

1. **Using Navigator.push() and Navigator.pop()**
   Push a new screen onto the stack and pop it to return to the previous one.

2. **Named Routes**
   Define routes in MaterialApp and navigate using Navigator.pushNamed().

3. **GoRouter Package**
   A declarative approach to handle navigation more efficiently.

Gestures:
Flutter's GestureDetector and InkWell widgets help in detecting user interactions like taps, swipes, and long presses. Some common gestures include:

- **onTap:** Detects a tap event.
- **onDoubleTap:** Recognizes double taps.
- **onLongPress:** Detects a long press on a widget.
- **onHorizontalDrag & onVerticalDrag:** Detects drag/swipe motions.

Steps:
Step 1: Create a Flutter project and define multiple screens.
Step 2: Set up named routes in `MaterialApp`.
Step 3: Implement navigation using `Navigator.push()` and `Navigator.pushNamed()`.
Step 4: Use `GestureDetector` to detect taps, swipes, and long presses.
Step 5: Add buttons or swipes to navigate between screens.
Code:

main.dart:

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

void main() async {
```

```
WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
runApp(MyApp()); // Use MyApp instead of directly calling MaterialApp
}


class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
debugShowCheckedModeBanner: false, // Removes the debug banner home:
      LoginPage(), // Set LoginPage as the initial screen
    );
  }
}
```

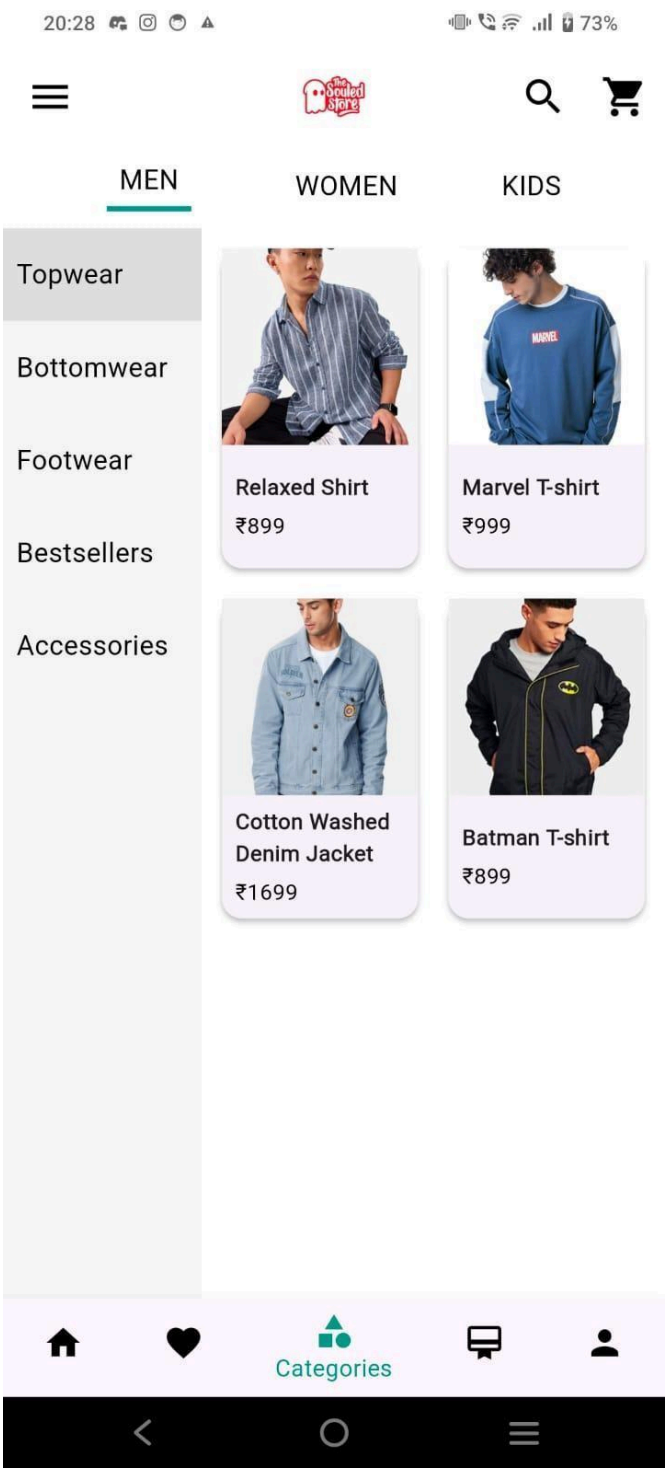
bottom navbar:

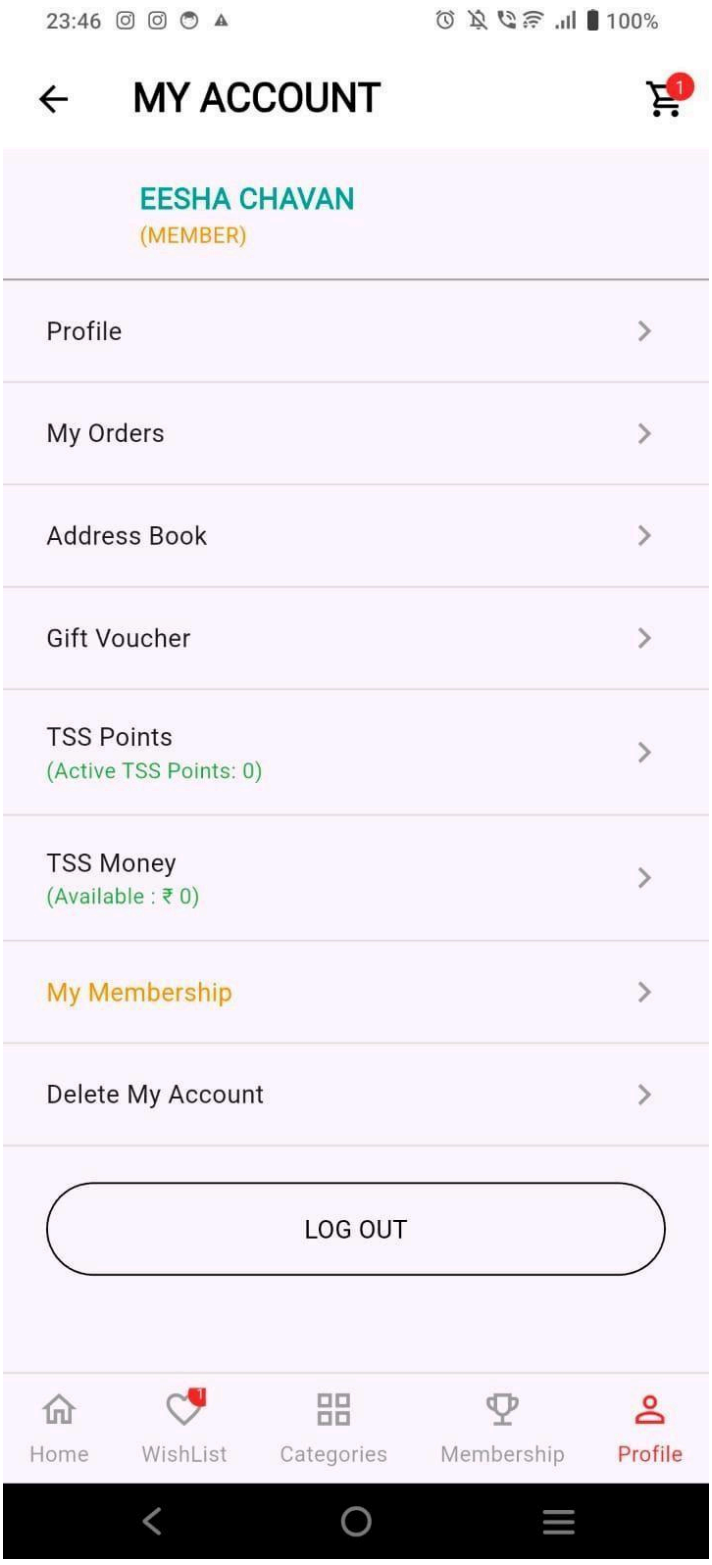
```
void _onItemTapped(int index) {
    if (index == 1) {
// Navigate to Wishlist when Wishlist icon is tapped
      Navigator.pushReplacement(
context,
MaterialPageRoute(builder: (context) => WishListPage()),
      );
} else if (index == 0) {
// Navigate to Home when Home icon is tapped Navigator.pushReplacement(
context,
MaterialPageRoute(builder: (context) => HomePage()),
      );
} else if (index == 2) {
// Navigate to Categories when Categories icon is tapped
      Navigator.pushReplacement(
context,
```

```
        MaterialPageRoute(builder: (context) => CategoriesPage()),
      );
    } else if (index == 4) {
      // Navigate to Profile when Profile icon is tapped
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => ProfilePage()),
      );
    } else {
      setState(() {
        _selectedIndex = index;
      });
    }
  }
```

Output:

Project Title: The Souled Store Clone **Roll No. 08**




23:46
100%
MY ACCOUNT
EESHA CHAVAN
(MEMBER)
Profile
My Orders
Address Book
Gift Voucher
TSS Points
(Active TSS Points: 0)
TSS Money
(Available : ₹ 0)
My Membership
Delete My Account
LOG OUT
Home
WishList
Categories
Membership
Profile

MAD & PWA Lab

Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	

MPL Experiment 6

Name: Eesha Chavan **Class:** D15A **Roll no:** 8

Aim: How To Set Up Firebase with Flutter for iOS and Android Apps

Steps to Set Up Firebase with Flutter:

Step 1:

Go to the Firebase Console (https://console.firebase.google.com/). Click on "Add Project" and follow the steps to create your Firebase project. Once the project is created, select the Flutter option for connecting your app with Firebase.

Step 2:

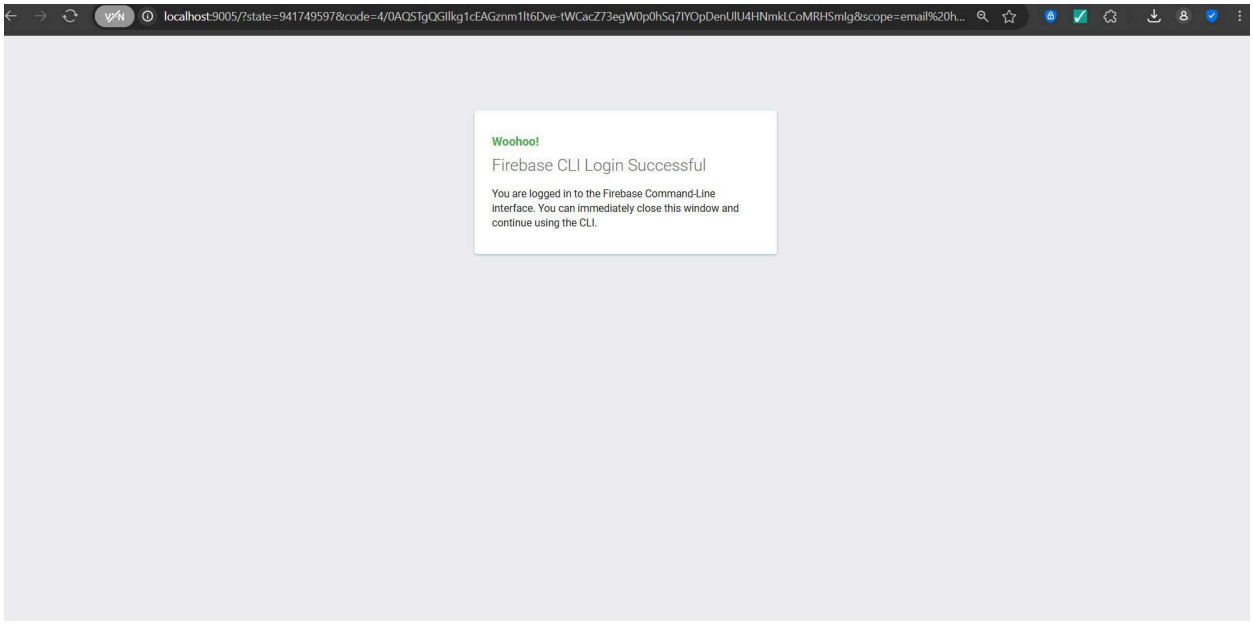
Open **Windows PowerShell** (or any terminal you prefer).

Run the following commands to install Firebase CLI and verify the installation:

```
npm install -g firebase-tools
firebase --version
firebase login
```

This will install the **Firebase CLI**, check the version, and log you into your Firebase account.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Eesha Chavan> npm install -g firebase-tools

added 631 packages in 1m

71 packages are looking for funding
  run `npm fund` for details
npm notice
npm notice New major version of npm available! 10.2.4 -> 11.1.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.1.0
npm notice Run npm install -g npm@11.1.0 to update!
npm notice
PS C:\Users\Eesha Chavan> firebase --version
13.32.0
PS C:\Users\Eesha Chavan>
PS C:\Users\Eesha Chavan> firebase login
i  Firebase optionally collects CLI and Emulator Suite usage and error reporting information to help improve our products. Data is collected in accordance w
ith Google's privacy policy (https://policies.google.com/privacy) and is not used to identify you.

? Allow Firebase to collect CLI and Emulator Suite usage and error reporting information? Yes
i  To change your data collection preference at any time, run `firebase logout` and log in again.

Visit this URL on this device to log in:
https://accounts.google.com/o/oauth2/auth?client_id=563584335869-fgrhgmd47bqnekij5i8b5pr03ho849e6.apps.googleusercontent.com&scope=email%20openid%20https%3A
%2F%2Fwww.googleapis.com%2Fauth%2Fcloudplatformprojects.readonly%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Ffirebase%20https%3A%2F%2Fwww.googleapis.com%2Fa
uth%2Fcloud-platform&response_type=code&state=941749597&redirect_uri=http%3A%2F%2Flocalhost%3A9005

Waiting for authentication...

+  Success! Logged in as eeshachavan17@gmail.com
PS C:\Users\Eesha Chavan>
```



Step 3:

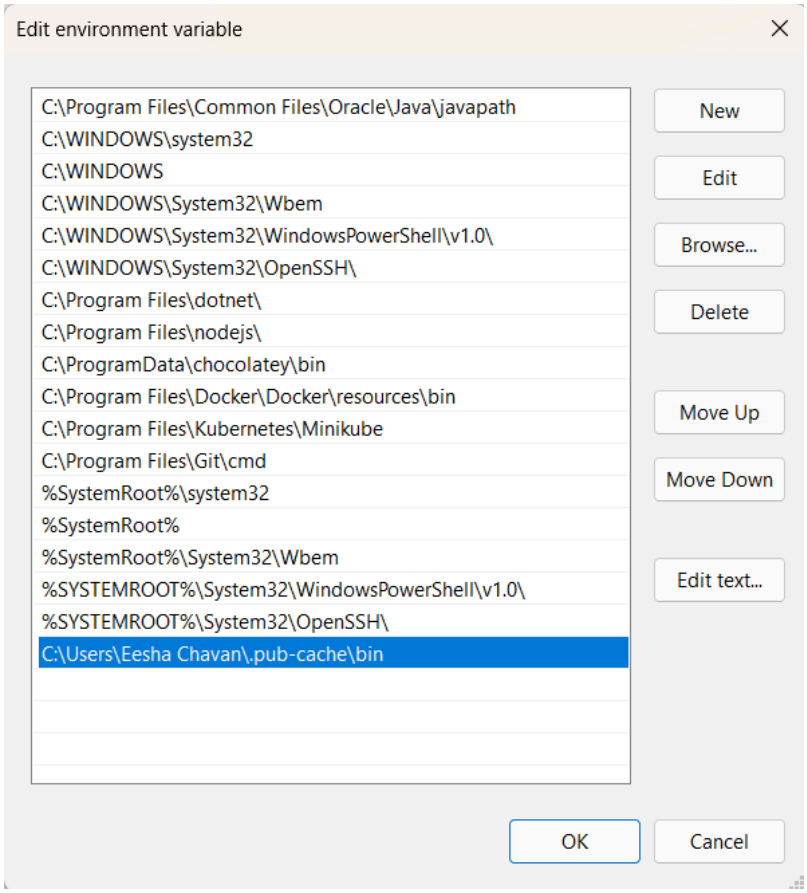
Open your Flutter app in **Android Studio**.

In the terminal of Android Studio, run the following command to activate flutterfire_cli:

```
dart pub global activate flutterfire_cli
```

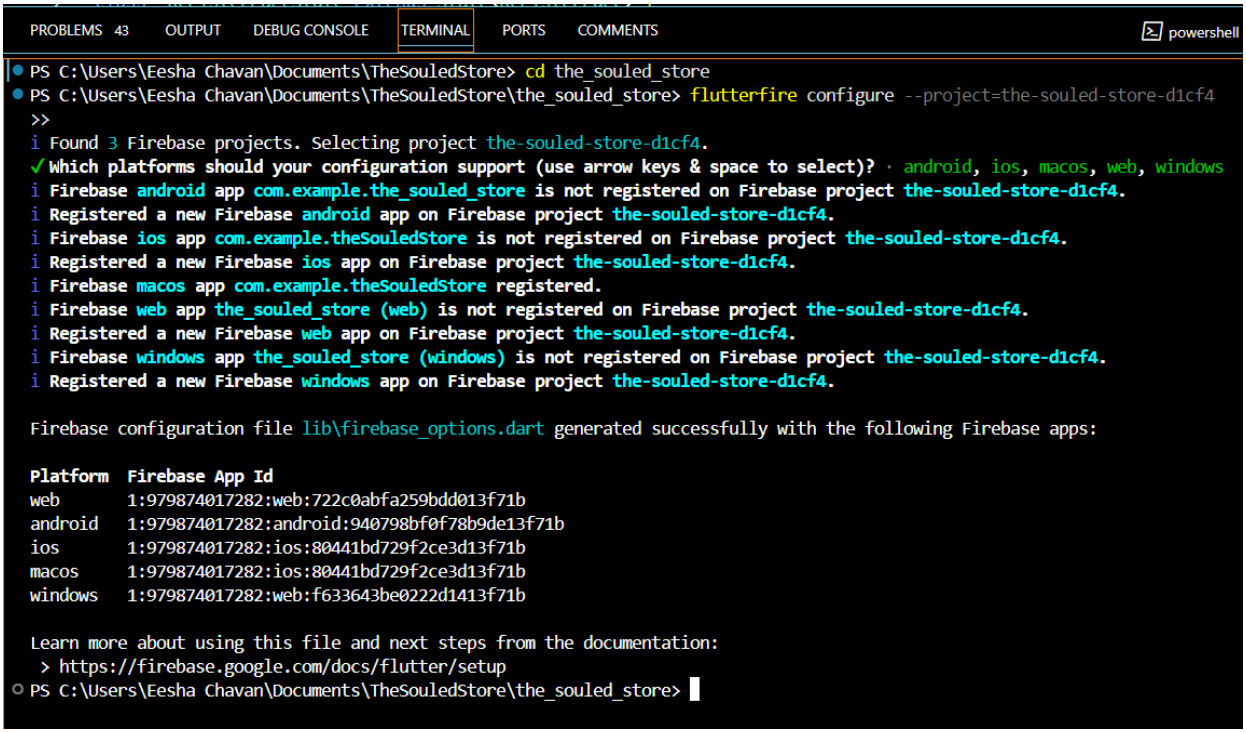
Add flutterfire to your Environment Variables. You may need to restart Android Studio after

adding it.



Step 4:

Run the following command to configure Firebase with your project: `flutterfire configure --project=dmart-c9f2c` Replace `dmart-c9f2c` with your Firebase project ID.



Step 5:

Run this command to add Firebase Core dependency to your app:

```
flutter pub add firebase_core
```

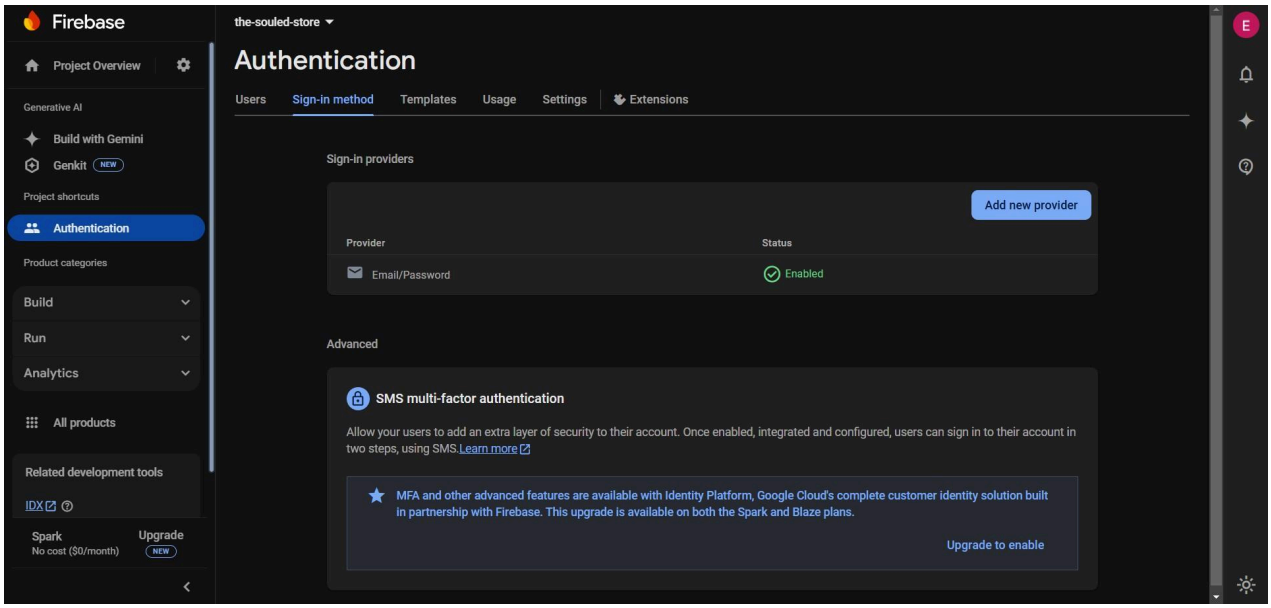
```
PS C:\Users\Eesha Chavan\Documents\TheSouledStore\the_souled_store> flutter pub add firebase_core
Resolving dependencies...
Downloading packages... (6.7s)
  async 2.11.0 (2.13.0 available)
  boolean_selector 2.1.1 (2.1.2 available)
  characters 1.3.0 (1.4.0 available)
  clock 1.1.1 (1.1.2 available)
  collection 1.19.0 (1.19.1 available)
  fake_async 1.3.1 (1.3.3 available)
+ firebase_core 3.12.1
+ firebase_core_platform_interface 5.4.0
+ firebase_core_web 2.21.1
  leak_tracker 10.0.7 (10.0.9 available)
  leak_tracker_flutter_testing 3.0.8 (3.0.9 available)
  matcher 0.12.16+1 (0.12.17 available)
  material_color_utilities 0.11.1 (0.12.0 available)
  meta 1.15.0 (1.16.0 available)
  path 1.9.0 (1.9.1 available)
+ plugin_platform_interface 2.1.8
  source_span 1.10.0 (1.10.1 available)
  stack_trace 1.12.0 (1.12.1 available)
  stream_channel 2.1.2 (2.1.4 available)
  string_scanner 1.3.0 (1.4.1 available)
  term_glyph 1.2.1 (1.2.2 available)
  test_api 0.7.3 (0.7.4 available)
  vm_service 14.3.0 (15.0.0 available)
Changed 4 dependencies!
19 packages have newer versions incompatible with dependency constraints.
Try `flutter pub outdated` for more information.
```

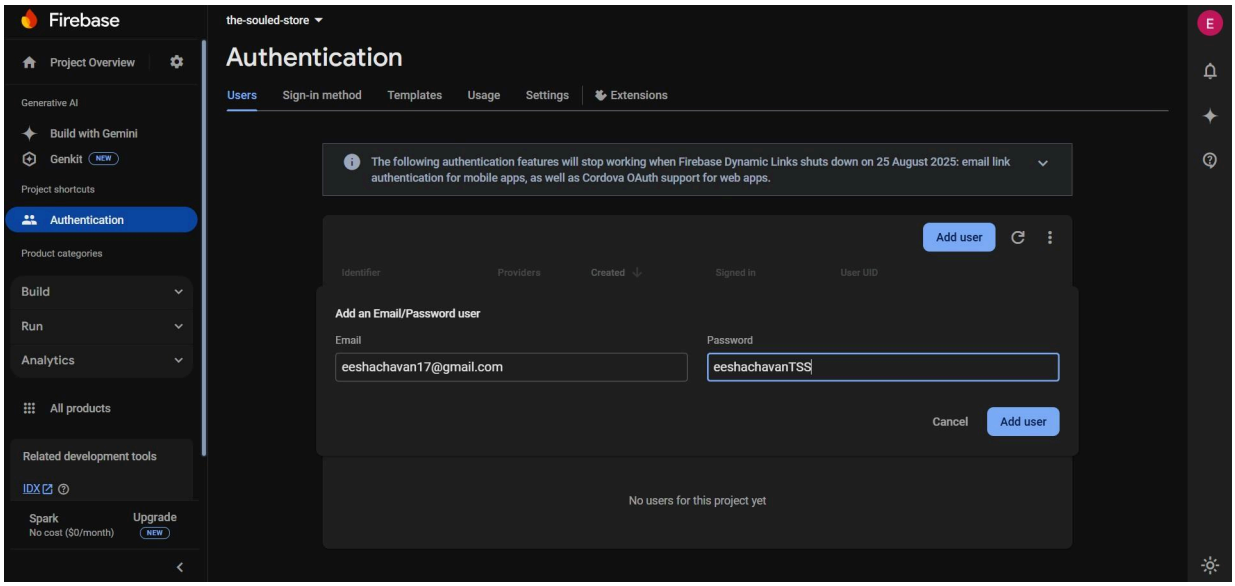
Firebase Authentication SetUp

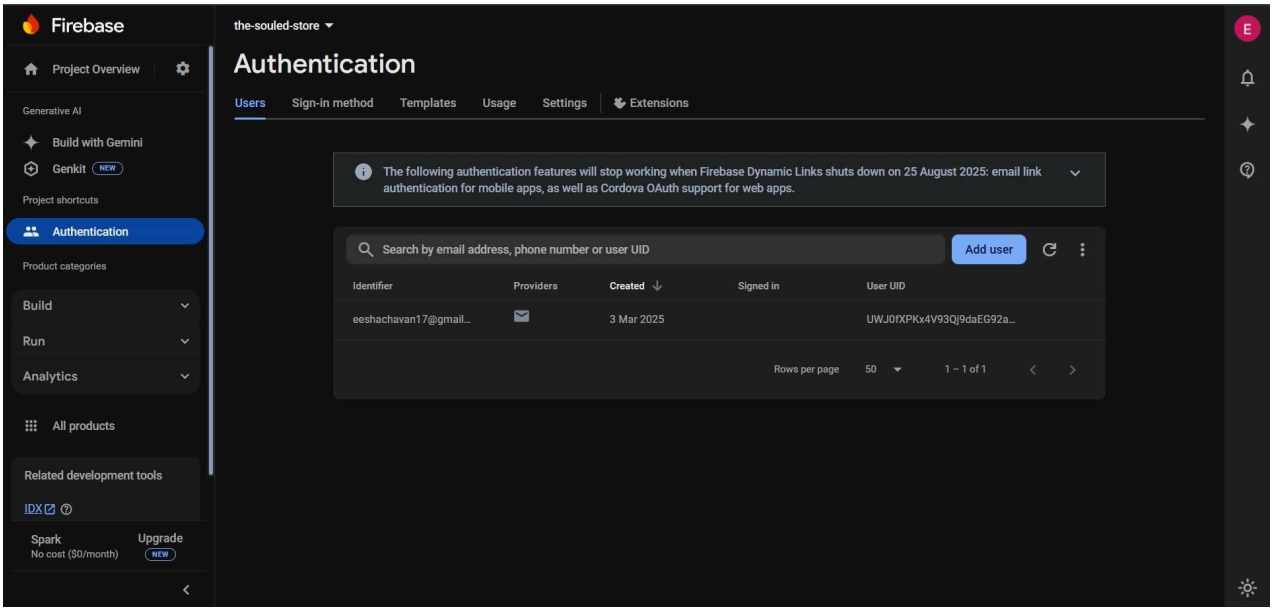
Step 1:

Go to the Firebase Console (https://console.firebase.google.com/). In your Firebase project, navigate to **Authentication**. Under the **Sign-in method** tab, enable **Email/Password** sign-in. Once enabled, go to the **Users** section and click on **Add User**. Enter a **username** (email) and a

password for the new user.







Step 2:

Open your app in **Android Studio**.

In the terminal, run the following command to add the Firebase Authentication dependency:

```
flutter pub add firebase_auth
```

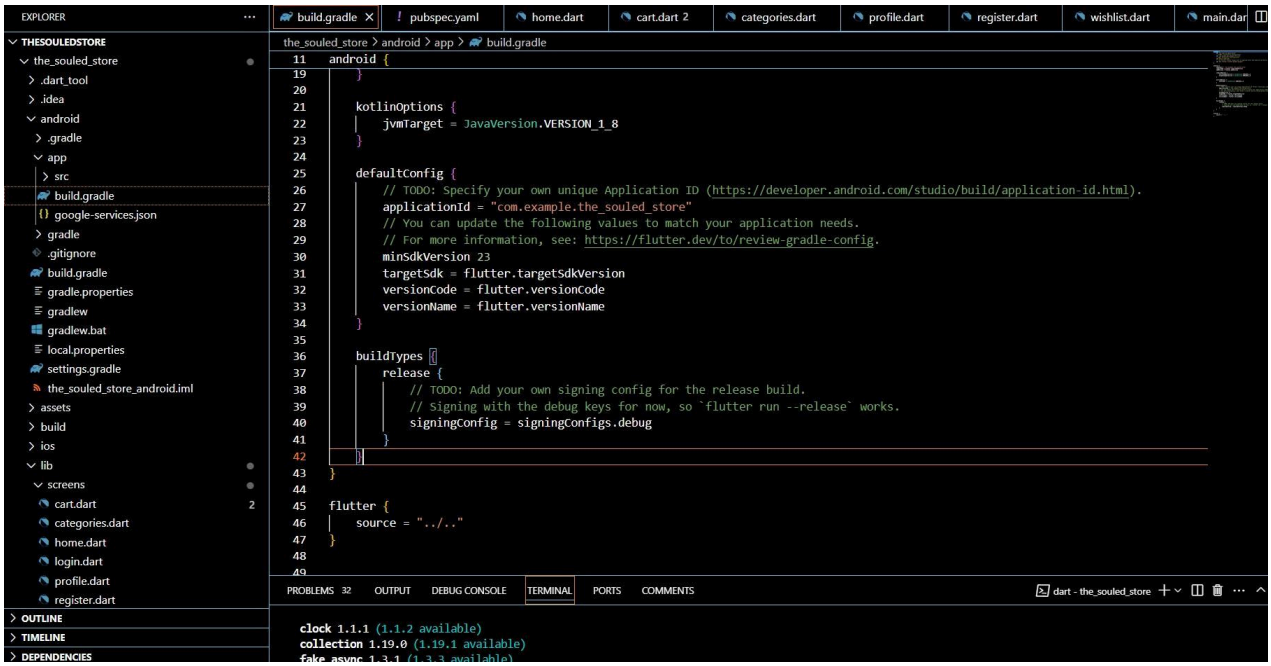
```
PS C:\Users\Eesha Chavan\Documents\TheSouledStore\the_souled_store> flutter pub add firebase_auth
Resolving dependencies...
Downloading packages... (6.9s)
+ _flutterfire_internals 1.3.53
  async 2.11.0 (2.13.0 available)
  boolean_selector 2.1.1 (2.1.2 available)
  characters 1.3.0 (1.4.0 available)
  clock 1.1.1 (1.1.2 available)
  collection 1.19.0 (1.19.1 available)
  fake_async 1.3.1 (1.3.3 available)
+ firebase_auth 5.5.1
+ firebase_auth_platform_interface 7.6.1
+ firebase_auth_web 5.14.1
+ http_parser 4.1.2
  leak_tracker 10.0.7 (10.0.9 available)
  leak_tracker_flutter_testing 3.0.8 (3.0.9 available)
  matcher 0.12.16+1 (0.12.17 available)
  material_color_utilities 0.11.1 (0.12.0 available)
  meta 1.15.0 (1.16.0 available)
  path 1.9.0 (1.9.1 available)
  source_span 1.10.0 (1.10.1 available)
  stack_trace 1.12.0 (1.12.1 available)
  stream_channel 2.1.2 (2.1.4 available)
  string_scanner 1.3.0 (1.4.1 available)
  term_glyph 1.2.1 (1.2.2 available)
  test_api 0.7.3 (0.7.4 available)
+ typed_data 1.4.0
  vm_service 14.3.0 (15.0.0 available)
Changed 6 dependencies!
19 packages have newer versions incompatible with dependency constraints.
Try `flutter pub outdated` for more information.
```

Step 3:

If you encounter any issues, add the following configuration to your `android/app/build.gradle` file:

```
android {
    defaultConfig
        { minSdk =
        23
        targetSdk = flutter.targetSdkVersion
        versionCode = flutter.versionCode
        versionName = flutter.versionName
    }
}
```

This ensures that the Firebase dependencies are compatible with your Android app.

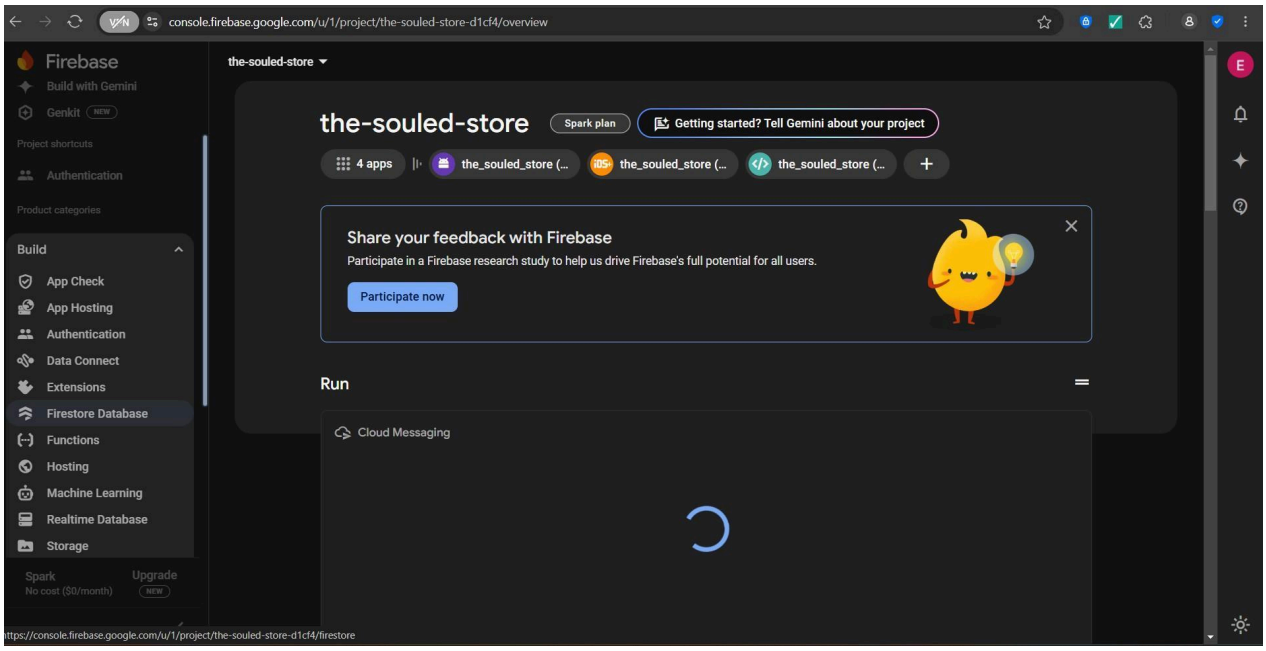


Now, firebase is successfully connected to our app.

Firestore Database Setup

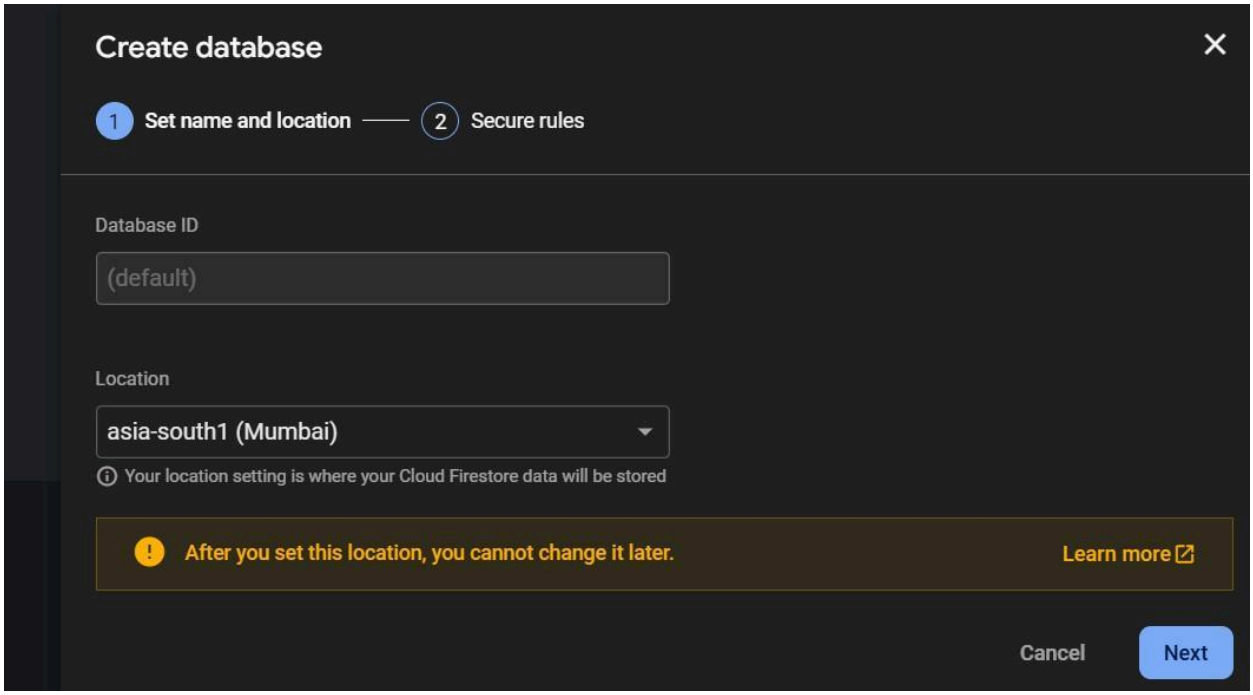
Step 1:

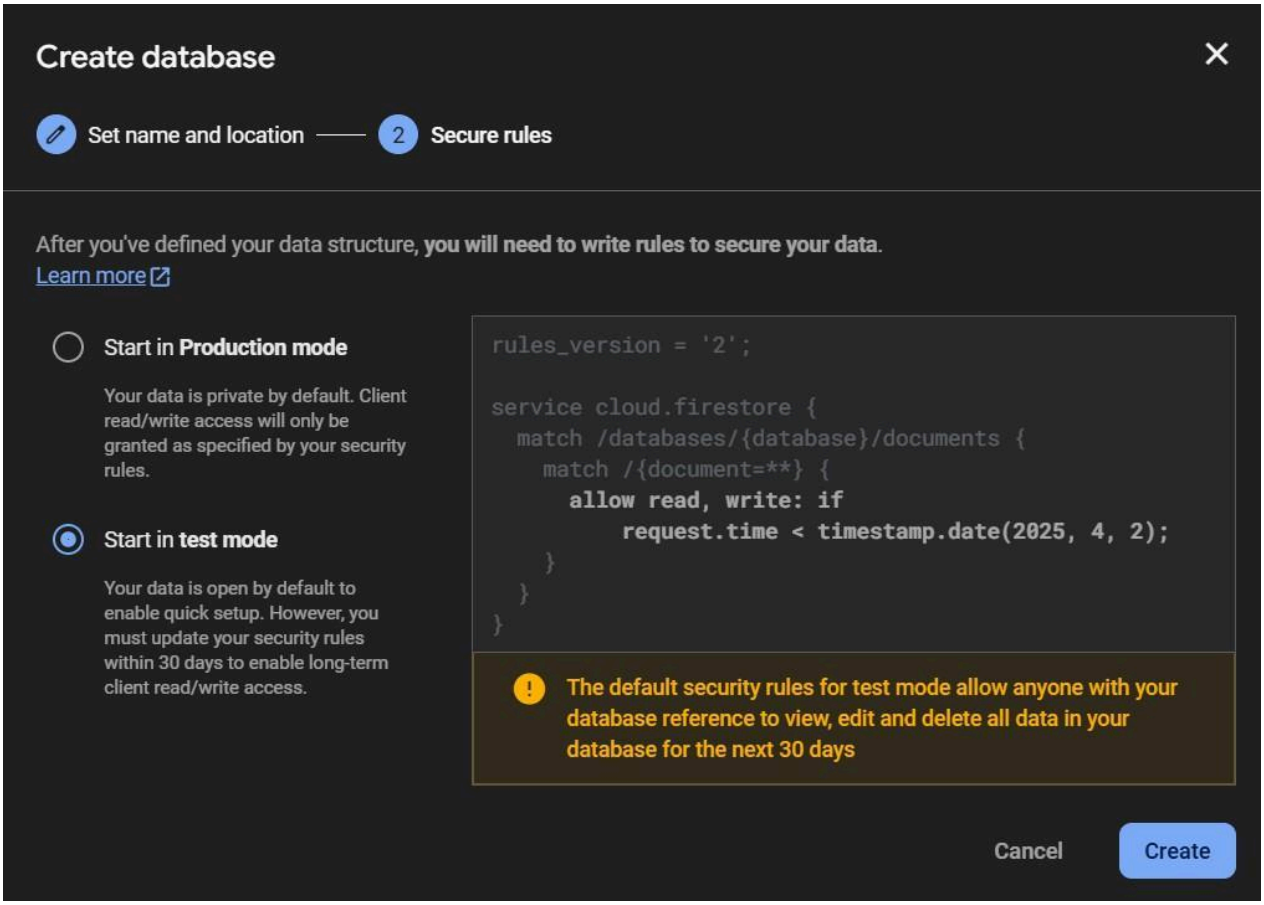
Select **Firestore Database** from **Build** in the sidebar.



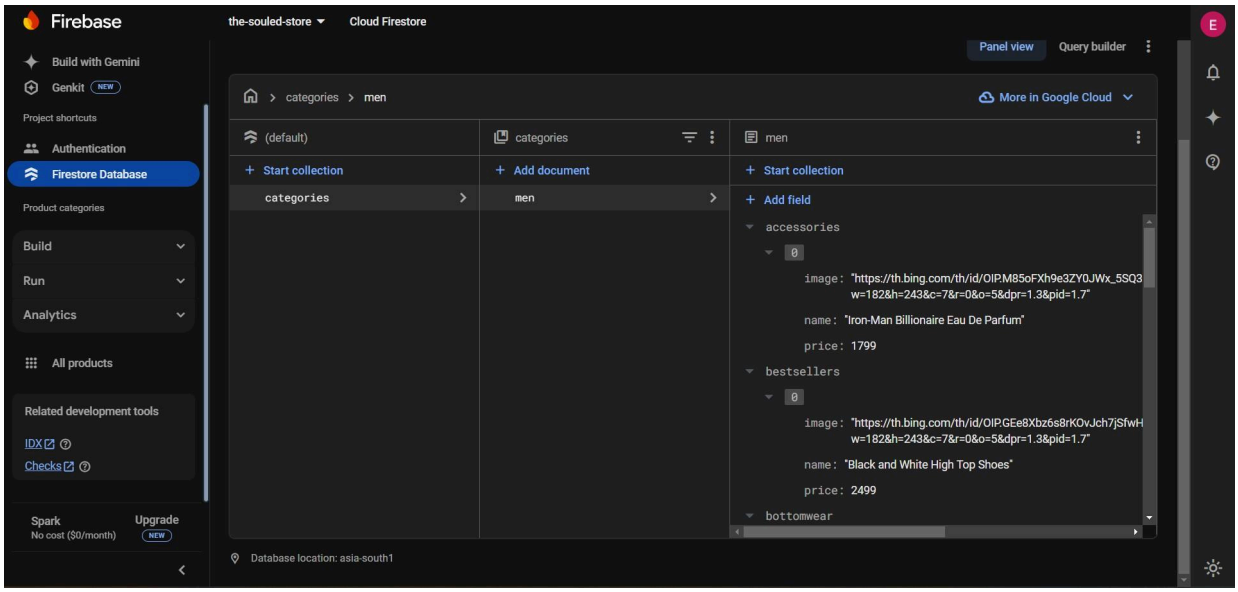
Then click on Create database. For location select asia-south1(Mumbai). Then choose **Start in test mode** (for development).

Click **Next** and choose a location, then **Enable**.





Step 2:
Start by creating a **collection**. Add a **document** within the collection. Define the **fields** as per your project requirements.



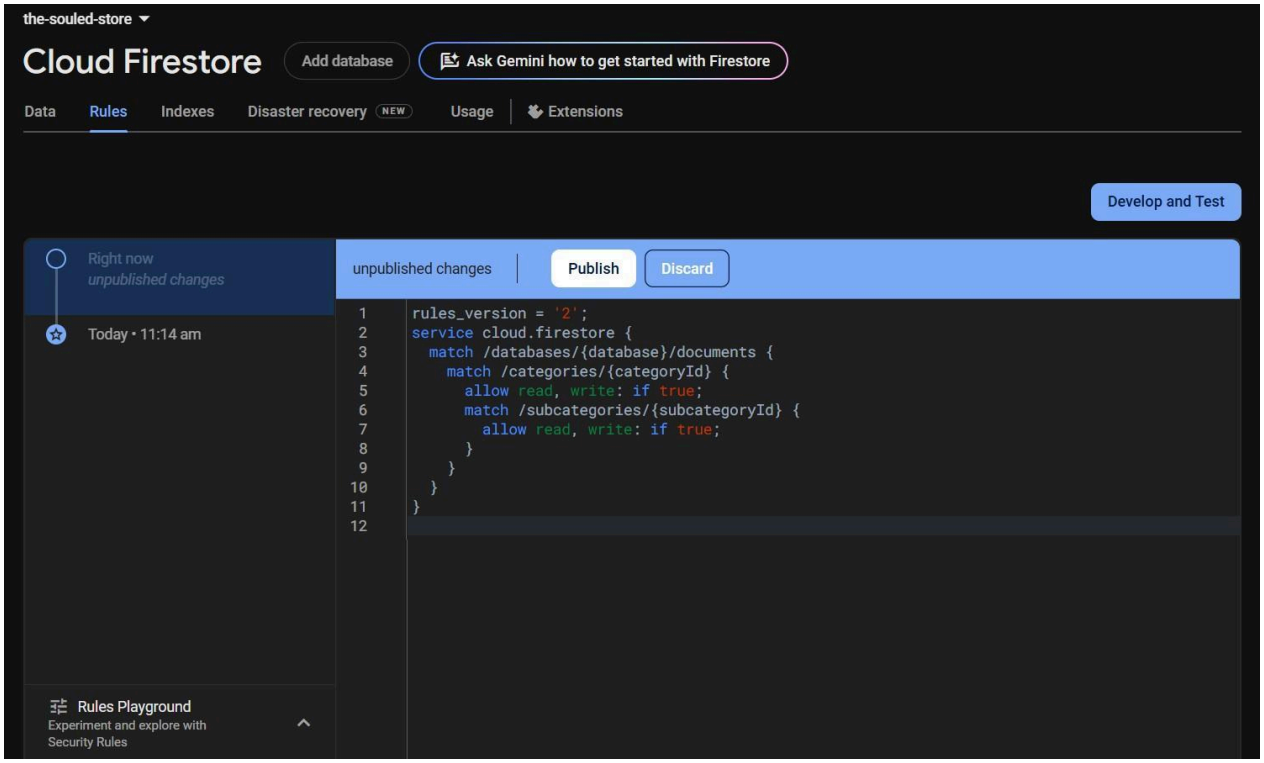
Step 3:
Run the following command to add Firestore to your Flutter app:

```
flutter pub add cloud_firestore
```

Step 4:
Go to Firebase Console → Firestore Database → Rules
Set the following rules:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /categories/{categoryId} {
      allow read, write: if true;
      match /subcategories/{subcategoryId}
        { allow read, write: if true;
      }
    }
  }
}
```

and then click **Publish** to apply changes.



Code:

login.dart:

```dart
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'register.dart';
import 'home.dart';

class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController =
TextEditingController();
  final FirebaseAuth _auth = FirebaseAuth.instance;

  void _login() async {
    try {
      await _auth.signInWithEmailAndPassword(
        email: emailController.text.trim(),
        password: passwordController.text.trim(),
      );
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => HomePage()),
      );
    } catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text("Login failed: ${e.toString()}")),
      );
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      body: Column(
```

```
        children: [
          Expanded(
            flex: 45,
            child: Container(
              decoration: BoxDecoration(
                image: DecorationImage(
                  image: NetworkImage(

"https://rukminim2.flixcart.com/image/612/612/xif0q/t-shirt/c/e/o/m-195139
-the-souled-store-original-imaghy3nqsm7ctkg.jpeg?q=70",
                  ),
                  fit: BoxFit.cover,
                ),
              ),
            ),
          ),
          Expanded(
            flex: 55,
            child: Container(
              padding: EdgeInsets.symmetric(horizontal: 24, vertical: 25),
              decoration: BoxDecoration(
                color: Colors.white,
                borderRadius: BorderRadius.vertical(top:
Radius.circular(20)),
              ),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  SizedBox(height: 10),
                  TextField(
                    controller: emailController,
                    keyboardType: TextInputType.emailAddress,
                    decoration: InputDecoration(
                      labelText: "Email",
                      border: OutlineInputBorder(),
                    ),
                  ),
                  SizedBox(height: 10),
                  TextField(
                    controller: passwordController,
```

```
                    obscureText: true,
                    decoration: InputDecoration(
                      labelText: "Password",
                      border: OutlineInputBorder(),
                    ),
                  ),
                  SizedBox(height: 20),
                  ElevatedButton(
                    style: ElevatedButton.styleFrom(
                      backgroundColor: Color(0xFF31827C),
                      minimumSize: Size(double.infinity, 50),
                      shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(8)),
                    ),
                    onPressed: _login,
                    child: Text(
                      "LOGIN",
                      style: TextStyle(color: Colors.white, fontSize: 16,
fontWeight: FontWeight.bold),
                    ),
                  ),
                  SizedBox(height: 20),
                  Center(
                    child: GestureDetector(
                      onTap: () {
                        Navigator.push(
                          context,
                          MaterialPageRoute(builder: (context) =>
RegisterPage()),
                                                );
                                              },
                                              c
                                                h
                                                i
                                                l
                                                d
                                                :
                                                R
                                                i
                                                c
                                                h
                                                T
                                                e
                                                x
                                                t

Colors.grey),
```

```
t
e
x
t
:
T
e
x
t
S
p
a
n

  style: TextStyle fontSize: 14
  color:

  children:
    TextSpan text: "Don't
    have an account? "
    TextSpan
      text: "REGISTER NOW"
```

```
                              style: TextStyle(color: Colors.red,
decoration: TextDecoration.underline),
                            ),
                          ],
                        ),
                      ),
                    ),
                  ),
                ],
              ),
            ),
          ),
        ],
      ),
    );
  }
}
```

signup.dart:

```
import 'package:flutter/material.dart'
import 'package:firebase_auth/firebase_auth.dart'; // Import Firebase Auth
import 'login.dart'; // Importing login page for navigation

class RegisterPage extends StatefulWidget {
  @override
_RegisterPageState createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  DateTime? selectedDate;
String? selectedGender;

// Form Key for validation
final _formKey = GlobalKey<FormState>();

// Controllers for text fields
  final TextEditingController _firstNameController =
TextEditingController
```

```
  final TextEditingController _lastNameController =
TextEditingController();
  final          TextEditingController          _phoneController          =
  TextEditingController();  final  TextEditingController  _emailController
  =       TextEditingController();       final       TextEditingController
  _passwordController =
TextEditingController();

  // Firebase Auth instance
  final FirebaseAuth _auth = FirebaseAuth.instance;

  // Function to handle user registration
  Future<void> _registerUser() async {
    if (_formKey.currentState!.validate()) {
      try {
        // Create user with email and password
        UserCredential userCredential = await
_auth.createUserWithEmailAndPassword(
          email: _emailController.text.trim(),
          password: _passwordController.text.trim(),
        );

        // If registration is successful, print user details
        print("User Registered: ${userCredential.user!.email}");

        // Navigate to the login page after successful registration
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (context) => LoginPage()),
        );
      } on FirebaseAuthException catch (e) {
        // Handle registration errors
        String errorMessage = "Registration failed. Please try again.";
        if (e.code == 'weak-password') {
          errorMessage = "The password provided is too weak.";
        } else if (e.code == 'email-already-in-use') {
          errorMessage = "The account already exists for that email.";
        }

        // Show error message to the user
        ScaffoldMessenger of(context) showSnackBar(
```

```
          SnackBar(content: Text(errorMessage)),
        );
      } catch (e) {
        // Handle other errors
        print("Error: $e");
        ScaffoldMessenger.of(context).showSnackBar(
          SnackBar(content: Text("An unexpected error occurred. Please try
again.")),
        );
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 0,
        leading: IconButton(
          icon: Icon(Icons.arrow_back, color: Colors.black),
          onPressed: () => Navigator.pushReplacement(
            context,
            MaterialPageRoute(builder: (context) => LoginPage()),
          ),
        ),
      ),
      body: SingleChildScrollView(
        padding: EdgeInsets.symmetric(horizontal: 20),
        child: Form(
          key: _formKey, // Assign the form key
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Center(
                child: Column(
                  children: [
                    Text(
                      "Looks like you are new",
```

```
                    style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold, color: Colors.red),
                  ),
                  SizedBox(height: 5),
                  Text(
                    "Get 10% additional cashback as TSS points on every
purchase.",                                          textAlign: TextAlign.center
                                                     style: TextStyle(fontSize: 14
                                                     color: Colors.grey),
                                                   ),
                                                 ],
                                               ),
                                             ),
              SizedBox(height: 20),

              // First Name & Last Name
              Row(
                children: [
                  Expanded(
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Text("First Name", style: TextStyle(fontSize: 14,
color: Colors.black)),                       TextFormField(
                                               controller:
                                               _firstNameController,
                                               decoration:
                                               InputDecoration(
                                                 hintText: "First Name *",
                                                 hintStyle:
                                                 TextStyle(color:
                                                 Colors.grey), border:
                                                 OutlineInputBorder(),
                                               ),
                                               validator: (value) {
                                                 if (value == null ||
                                                   value.isEmpty) {
                                                   return 'First Name is
                                                   required';
                                 }
                                 return null;
```

```
                ),
                SizedBox(width: 10),
                Expanded(
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      Text("Last Name", style: TextStyle(fontSize: 14,
color: Colors.black)),                    TextFormField(
                                            controller:
                                            _lastNameController,
                                            decoration:
                                            InputDecoration(
                                              hintText: "Last Name *",
                                              hintStyle:
                                              TextStyle(color:
                                              Colors.grey), border:
                                              OutlineInputBorder(),
                                            ),
                                            validator: (value) {
                                              if (value == null ||
                                                value.isEmpty) {
                                                return 'Last Name is
                                                required';
                          }
                          return null;
                        },
                      ),
                    ],
                  ),
                ),
              ],
            ),
            SizedBox(height: 15),

            // Mobile Number
            TextFormField(
              controller: _phoneController,
              keyboardType: TextInputType.phone,
              decoration: InputDecoration(
                hintText: "+91 *",
                hintStyle: TextStyle(color: Colors.grey),
                  border: OutlineInputBorder

                validator:  value
```

```
                  if (value == null || value.isEmpty) {return 'Phone number is
                  required';
                  } else if (value.length != 10) {
                    return 'Phone number must be 10 digits';
                  }
                  return null;
                },
              ),
              SizedBox(height: 15),

              // Email ID
              Text("Email ID", style: TextStyle(fontSize: 14, color:
Colors.black)),
              TextFormField(
                controller: _emailController,
                keyboardType: TextInputType.emailAddress,
                decoration: InputDecoration(
                  hintText: "Email ID *",
                  hintStyle: TextStyle(color: Colors.grey),
                  border: OutlineInputBorder(),
                ),
                validator: (value) {
                  if (value == null || value.isEmpty) {
                    return 'Email is required';
                  } else if
(!RegExp(r'^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$').hasMatch(value)) {
                    return 'Enter a valid email address';
                  }
                  return null;
                },
              ),
              SizedBox(height: 15),

              // Password
              Text("Password", style: TextStyle(fontSize: 14, color:
Colors.black)),
              TextFormField(
                controller: _passwordController,
                obscureText: true,
                decoration: InputDecoration(
                  hintText: "Password *",
```

```
                                                hintStyle:
                                                TextStyle color:
                                                Colors grey   border:
Colors grey                                     OutlineInputBorder
                                                suffixIcon:
                                                Icon Icons visibility_off  color:


                                              validator:  value
                                                if  value ==
                                                  null ||
                                                  value isEmpty
                                                    return
                                                  'Password is
                                                  required'

                    return null


              SizedBox height: 15

              // Birthdate
              Text "Birthdate"  style: TextStyle fontSize: 14  color:
Colors black
              TextFormField
                readOnly: true
                decoration: InputDecoration
                  hintText: "DD-MM-YYYY *"
                  hintStyle: TextStyle color: Colors grey
                  border: OutlineInputBorder
                  suffixIcon: Icon Icons calendar_today  color:
                                          Colors grey
Colors grey
```

```
),
onTap: () async {
  DateTime? pickedDate =
    await
    showDatePicker(
    context: context,
    initia
    lDate:
    DateTi
    me.now
    (),
    firstD
    ate:
    DateTi
    me(190
    0),
    lastDate: DateTime.now(),
    builde
      r:
ButtonTextTheme.primary),
```

```
(con
text
,
chil
d) {
retu
rn
Them
e(
  data:
    ThemeData.ligh
    t().copyWith(
    primaryColor:
    Colors.grey,
    colorScheme:
    ColorScheme.light(prima
    buttonTheme:
    ButtonThemeData(textThe
```

```
                ),
                child: child!,
              );
            },
          );

          if (pickedDate != null) {
            setState(() {
              selectedDate = pickedDate;
            });
          }
        },
        validator: (value) {
          if (selectedDate == null) {
            return 'Birthdate is required';
          }
          return null;
        },
      ),
      SizedBox(height: 5),

      // Birthday Discount Note
      Text(
        "Avail 10% birthday discount as a member",
        style: TextStyle(fontSize: 12, color: Colors.grey),
      ),
      SizedBox(height: 20),

      // Gender Selection
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: [
          _genderOption("Male"),
          _genderOption("Female"),
          _genderOption("Other"),
        ],
      ),
      SizedBox(height: 20),

      // Register Button
```

```
                                                (
                                                    style:
                                                    ElevatedButton.styleFrom(
                                                      backgroundColor:
                                                      Color(0xFF31827C), // New
                                                      Darker Teal
Color
                                                      shape:
                                                      RoundedRectangleBorder(
                                                        borderRadius:
                                                        BorderRadius.circular(8),
                                                  ),
                                                ),
                                                onPressed: _registerUser, // Call
                                                the registration
function
                                                child: Padding(
                                                  padding:
                                                  EdgeInsets.symmetric(vertical:
                                                  14),
                                                  child: Text("REGISTER", style:
                                                  TextStyle(fontSize: 16,
color: Colors.white)),
                  ),
                ),
              ),
              SizedBox(height: 10),
                                          vigator.p
a                                         ushReplac
    p                                     ement(
    :                                     context,
    (                                           MaterialPageRoute(builder:
    )                                           (context) =>
    {
                    );
                },
                child: RichText(
                  text: TextSpan(
                    text: "Already Registered? ",
                    style: TextStyle(color: Colors.grey, fontSize: 14),
                    children: [
                      TextSpan(
                        text: "LOG IN",
                        style: TextStyle(
                          color: Colors.red,
```

```
                            fontWeight: FontWeight.bold,
                            decoration: TextDecoration.underline,
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
              ),
              SizedBox(height: 20),
            ],
          ),
        ),
      ),
    );
  }

  // Gender Option Widget
  Widget _genderOption(String text) {
    return Row(
      children: [
        Radio(
          activeColor: Color(0xFF31827C), // New Teal Color for Selected
Option                                      value: text,
                                          )
                                          {
                                              setState(() {
                                                selectedGender =
                                                value.toString();
                                              });
                                            },
                                          ),
                                          Text(text, style: TextStyle(fontSize:
                                          14)),
      ],
    );
  }
}
```

categories.dart:

```dart
import 'package:flutter/material.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'home.dart';
import 'wishlist.dart';
import 'profile.dart';

class CategoriesPage extends StatefulWidget {
  @override
  _CategoriesPageState createState() => _CategoriesPageState();
}

class _CategoriesPageState extends State<CategoriesPage> {
  int _selectedIndex = 2; // Default index for Categories
  int _selectedTab = 0; // 0: MEN, 1: WOMEN, 2: KIDS
  int _selectedCategory = -1; // Track selected category

  // Categories for Men, Women, and Kids
  final List<String> _menCategories = [
    'topwear',
    'bottomwear',
    'footwear',
    'bestsellers',
    'accessories',
  ];

  final List<String> _womenCategories = [
    'topwear',
    'bottomwear',
    'footwear',
    'bestsellers',
    'accessories',
  ];

  final List<String> _kidsCategories = [
    'just_launched',
    'bestsellers',
    'boys',
    'girls',
    'accessories',
```

```
];

// Display names for categories (for UI)
final Map<String, String> _displayNames = {
  'topwear': 'Topwear',
  'bottomwear': 'Bottomwear',
  'footwear': 'Footwear',
  'bestsellers': 'Bestsellers',
  'accessories': 'Accessories',
  'just_launched': 'Just Launched',
  'boys': 'Boys',
  'girls': 'Girls',
};



// Function to get collection name based on selected tab
String _getCollectionName() {
  switch (_selectedTab) {
    case 0:
      return 'men';
    case 1:
      return 'women';
    case 2:
      return 'kids';
    default:
      return 'men';
  }
}

void _onItemTapped(int index) {
  if (index == 1) {
    // Navigate to Wishlist when Wishlist icon is tapped
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => WishListPage()),
    );
  } else if (index == 0) {
    // Navigate to Home when Home icon is tapped
    Navigator.pushReplacement(
```

```
        context,
        MaterialPageRoute(builder: (context) => HomePage()),
      );
    } else if (index == 2) {
      // Already on categories page
      setState(() {
        _selectedIndex = index;
      });
    } else if (index == 4) {
      // Navigate to Profile when Profile icon is tapped
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => ProfilePage()),
      );
    } else {
      setState(() {
        _selectedIndex = index;
      });
    }
  }

  List<Widget> _buildProductGrid(List<dynamic>? products) {
    if (products == null || products.isEmpty) {
      return [
        Center(
          child: Text(
            "No products available",
            style: TextStyle(fontSize: 16),
          ),
        ),
      ];
    }

    return List.generate(products.length, (index) {
      final product = products[index];
      return Card(
        elevation: 2,
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
```

```
                                    AspectRatio(
                                      aspectRatio: 1,
                                      child: product['image'] != null
                                          ? Image.network(
                                              p
                                    adingBuilder: (context, child,
                                    loadingProgress) { if
                                    (loadingProgress == null) return
                                    child;
                                                  return Center(
                                                    child:
                                                    CircularProgressIndic
null                                                ator(
                                                      value:
                                                      loadingProgress exp
                                                      ectedTotalBytes !=

                                                          ?
                                                              loadingProg
                                                              ress.cumula
                                                              tiveBytesLo
                                                              aded /
                                                              loadingProg
                                                              ress.expect
                                                              edTotalByte
                                                              s!
                                                          : null,
                                        ),
                                    );
                                  },
                                                errorBuilder:
                                                  (context, error,
                                                  stackTrace) { return
                                                  Container(
50),                                                color: Colors.grey[200
                                                    child:
                                                    Icon(Icons.image_not_suppo
                                                    rted, size:

                                    );
                                  },
                                )
                              : Container(
                                  color: Colors.grey[200]
                                  child: Icon Icons.image_not_supported  size: 50
```

```
Padding
  padding: const EdgeInsets all 8.0
  child: Column
    crossAxisAlignment: CrossAxisAlignment start
    children:
      Text
        product 'name'  ?? 'No Name'
        style: TextStyle
          fontWeight: FontWeight bold
```

```
                    fontSize: 14,
                  ),
                  maxLines: 2,
                  overflow: TextOverflow.ellipsis,
                ),
                SizedBox(height: 4),
                Text(
                  '₹${product['price']?.toString() ?? '0'}',
                  style: TextStyle(
                    color: Colors.black,
                    fontWeight: FontWeight.w500,
                    fontSize: 14,
                  ),
                ),
              ],
            ),
          ),
        ],
      ),
    );
  });
}

@override
Widget build(BuildContext context) {
  List<String> currentCategories = _selectedTab == 0
      ? _menCategories
      : _selectedTab == 1
          ? _womenCategories
          : _kidsCategories;

  return Scaffold(
    backgroundColor: Colors.white,
    appBar: AppBar(
      backgroundColor: Colors.white,
      elevation: 0,
      leading: IconButton(
        icon: Icon(Icons.menu, color: Colors.black, size: 28),
        onPressed: () {},
      ),ding(
    padding: const EdgeInsets.symmetric(vertical: 10), child:
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly, children:
      ['MEN', 'WOMEN'
```

```
'KIDS'].asMap().entries.map((entry) {
                int index = entry.key;
                String category = entry.value;
                return GestureDetector(
                  onTap: () => setState(() {
                    _selectedTab = index;
                    _selectedCategory = -1; // Reset selected category
                  }),
                  child: Column(
                    children: [
                      Text(
                        category,
                        style: TextStyle(
                          fontSize: 16,
                          fontWeight: FontWeight.w500,
```

```
                        color: Colors.black,
                      ),
                    ),
                    if (_selectedTab == index)
                      Container(
                        margin: EdgeInsets.only(top: 4),
                        height: 3,
                        width: 50,
                        color: Colors.teal,
                      ),
                  ],
                ),
              );
            }).toList(),
          ),
        ),
        // Categories and Products
        Expanded(
          child: Row(
            children: [
              // Left Side: Categories List
              Container(
                width: MediaQuery.of(context).size.width * 0.3,
                color: Colors.grey[100],
                child: ListView(
                  children:
currentCategories.asMap().entries.map((entry) {
                    int index = entry.key;
                    String category = entry.value;
                    return GestureDetector(
                      onTap: () => setState(() {
                        _selectedCategory = index;
                      }),
                      child: Container(
                        color: _selectedCategory == index
                            ? Colors.grey[300]
                            : Colors.grey[100],
                        padding:
                            EdgeInsets.symmetric(vertical: 16,
horizontal: 8),
```

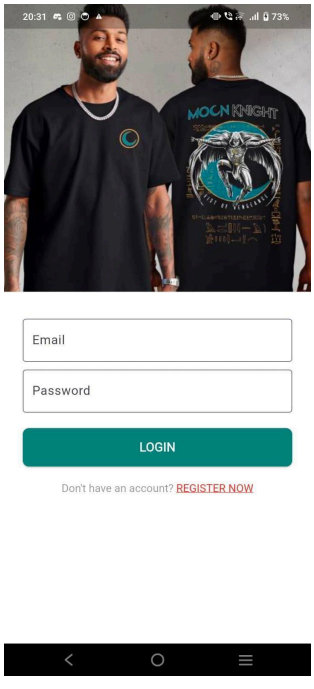
```
                          child: Text(
                            _displayNames[category] ?? category,
                            style: TextStyle(
                              fontSize: 16,
                              fontWeight: FontWeight.w500,
                              color: Colors.black,
                            ),
                          ),
                        ),
                      );
                    }).toList(),
                  ),
                ),
                // Right Side: Products
                Expanded(
                  child: _selectedCategory == -1
                      ? Center(
                          child: Text(
                            "Select a category to view products",
                            style: TextStyle(fontSize: 16),
                          ),
                        )
                      : FutureBuilder<DocumentSnapshot>(
                          future: FirebaseFirestore.instance
                              .collection('categories')
                              .doc(_getCollectionName())
                              .get(),
                          builder: (context, snapshot) {
                            if (snapshot.connectionState ==
                                ConnectionState.waiting) {
                              return Center(child:
CircularProgressIndicator());
                            }

                                                    if
                                                      (snapshot.
                                                      hasError)
                                                      { return
                                                      Center(
                                                        child: Text(
                                                          "Error loading products:
${snapshot.error}",
                                                          style: TextStyle color:
                                                          Colors red
```

```
                                ),
                              );
                            }

                                        if (!snapshot.hasData ||
!snapshot.data!.exists) {
                                          return Center(
                                            child: Text("No data available"
                                          );
                            }

                            // Get the category field name
                            String categoryField =
                                currentCategories[_selectedCategory];

                                        // Get products from the selected
                                        category
field

                                        List<dynamic>? products =
                                            (snapshot.data!.data() as
dynamic>)[categoryField];                   Map<String,


                            return Padding(
                              padding: const EdgeInsets.all(8.0),
                              child: GridView.count(
                                crossAxisCount: 2,
                                childAspectRatio: 0.7,
                                crossAxisSpacing: 10,
                                mainAxisSpacing: 10,
                                children: _buildProductGrid(products),
                              ),
                            );
                          },
                        ),
                      ),
                    ),
                  ],
                ),
              ),
            ],
          ),
          bottomNavigationBar: BottomNavigationBar(
```

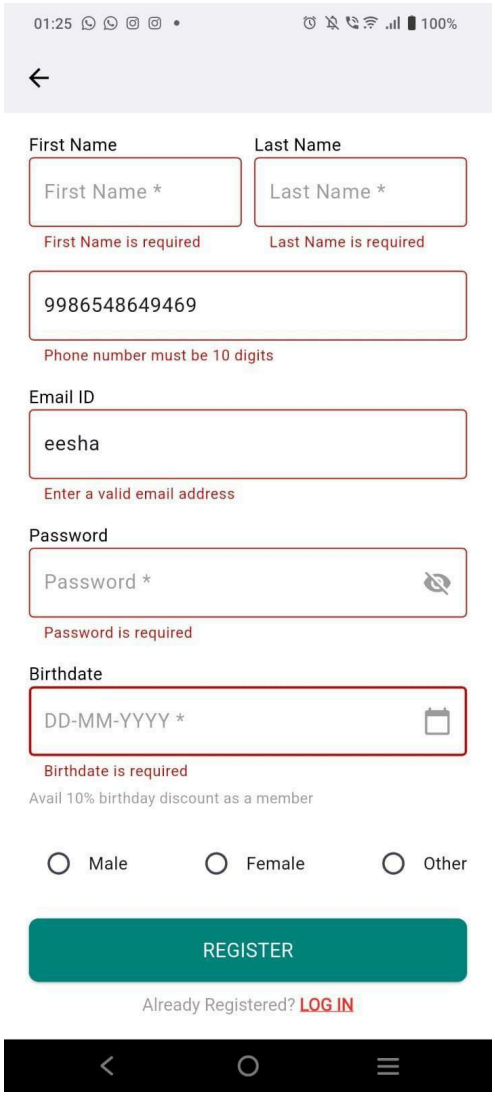
```
currentIndex: _selectedIndex, onTap:
        _onItemTapped,
selectedItemColor: Colors.teal,
unselectedItemColor: Colors.black, items: [
BottomNavigationBarItem(icon: Icon(Icons.home), label: "Home"),
          BottomNavigationBarItem(icon: Icon(Icons.favorite), label:
"Wishlist"),
          BottomNavigationBarItem(icon: Icon(Icons.category), label:
"Categories"),
          BottomNavigationBarItem(icon: Icon(Icons.card_membership),
label: "Membership"),
          BottomNavigationBarItem(icon: Icon(Icons.person), label:
"Profile"),
        ],
      ),
    );
  }
}
```

Output:

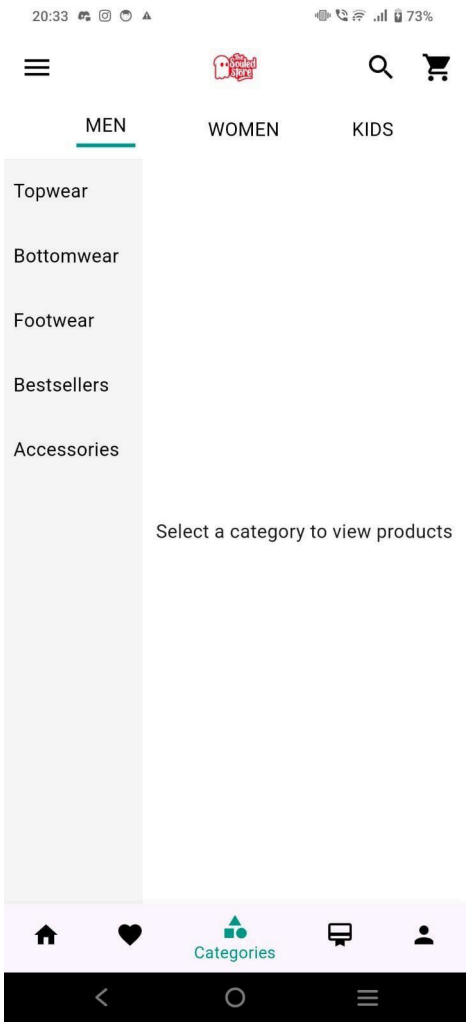
Login Page:



Signup page:

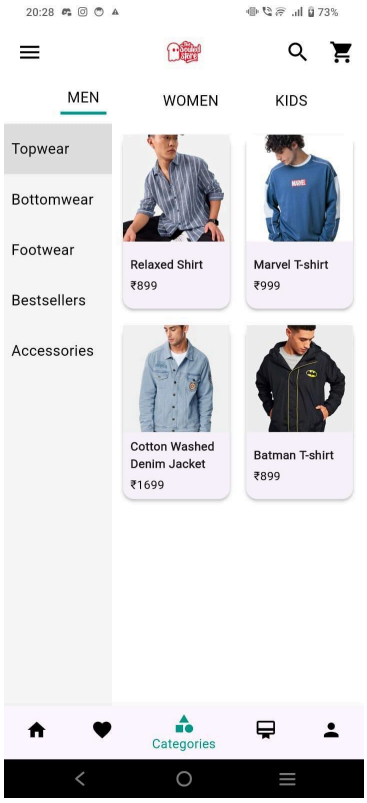

01:25
100%
First Name
Last Name
First Name *
Last Name *
First Name is required
Last Name is required
9986548649469
Phone number must be 10 digits
Email ID
eesha
Enter a valid email address
Password
Password *
Password is required
Birthdate
DD-MM-YYYY *
Birthdate is required
Avail 10% birthday discount as a member
Male
Female
Other
REGISTER
Already Registered? LOG IN

Category Page:



Project Title: The Souled Store Clone **Roll No. 08**

Product List Page:



MAD & PWA Lab
Journal

Experiment No.	07
Experiment Title.	To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	

Ria Chaudhari D15A07
Eesha Chavan D15A08
Shraeyaa Dhaigude D15A15

PWA EXP7

Aim: To write meta data of your PWA in a Web app manifest file to enable “add to homescreen feature”.

Theory:

Regular Web App

A regular web app is a website that is designed to be accessible on all mobile devices such that the content gets fit as per the device screen. It is designed using a web technology stack (HTML, CSS, JavaScript, Ruby, etc.) and operates via a browser. They offer various native-device features and functionalities. However, it entirely depends on the browser the user is using. In other words, it might be possible that you can access a native-device feature on Chrome but not on Safari or Mozilla Firefox because the browsers are incompatible with that feature.

Progressive Web App

Progressive Web App (PWA) is a regular web app, but some extras enable it to deliver an excellent user experience. It is a perfect blend of desktop and mobile application experience to give both platforms to the end-users.

Difference between PWAs vs. Regular Web Apps:

Progressive Web is different and better than a Regular Web app with features like:

1. Native Experience

Though a PWA runs on web technologies (HTML, CSS, JavaScript) like a Regular web app, it gives user experience like a native mobile application. It can use most native device features, including push notifications, without relying on the browser or any other entity. It offers a seamless and integrated user experience that it is quite tough for one to differentiate between a PWA and a Native application by considering its look and feel.

2. Ease of Access

Unlike other mobile apps, PWAs do not demand longer download time and make memory space available for installing the applications. The PWAs can be shared and installed by a link, which cuts down the number of steps to install and use. These applications can easily keep an app icon on the user's home screen, making the app easily accessible to the users and helps the brands remain in the users' minds, and improving the chances of interaction.

3. Faster Services
PWAs can cache the data and serve the user with text stylesheets, images, and other web content even before the page loads completely. This lowers the waiting time for the end-users and helps the brands improve the user engagement and retention rate, which eventually adds value to their business.

Features of the Progressive Web App
The main features are:
● Progressive
They work for every user, regardless of the browser chosen because they are built at the base with progressive improvement principles.
● Responsive
They adapt to the various screen sizes: desktop, mobile, tablet, or dimensions that can later become available.
● App-like
They behave with the user as if they were native apps, in terms of interaction and navigation.
● Updated
Information is always up-to-date thanks to the data update process offered by service workers.
● Secure
Exposed over HTTPS protocol to prevent the connection from displaying information or altering the contents.
● Searchable
They are identified as “applications” and are indexed by search engines.
● Reactivable
Make it easy to reactivate the application thanks to capabilities such as web notifications.
● Installable
They allow the user to “save” the apps that he considers most useful with the corresponding icon on the screen of his mobile terminal (home screen) without having to
face all the steps and problems related to the use of the app store.
● Linkable
Easily shared via URL without complex installations.

Code:

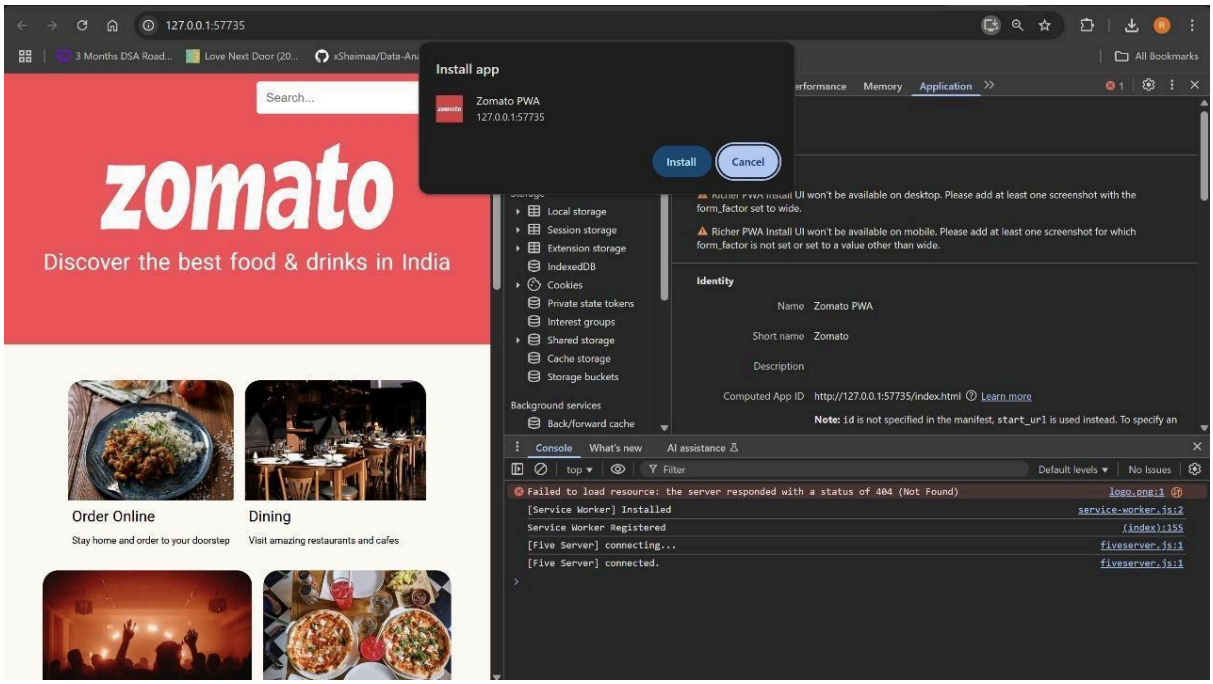
//manifest.json

```
{
    "name": "Zomato PWA",
    "short_name": "Zomato",
    "start_url": "./index.html",
    "display": "standalone",
    "background_color": "#ffffff",
    "theme_color": "#ff5a5f",
    "orientation": "portrait",
    "scope": "./",
    "icons": [
      {
        "src": "images/icon-192x192.png",
        "sizes": "192x192",
        "type": "image/png"
      },
      {
        "src": "images/icon-512x512.png",
        "sizes": "512x512",
        "type": "image/png"
      }
    ]
  }
```

//service-worker.js

```
self.addEventListener('install', function(e) {
    console.log('[Service Worker] Installed');
  });

  self.addEventListener('fetch', function(e) {
    console.log('[Service Worker] Fetch intercepted:', e.request.url);
  });
```

Output:



Shortcut added to home screen

Conclusion:

Hence, we learnt how to write a metadata of our E-commerce website PWA in a Web App Manifest File to enable add to homescreen feature.

MAD & PWA Lab
Journal

Experiment No.	08
Experiment Title.	To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Ria Chaudhari D15A07
Eesha Chavan D15A08
Shraeyaa Dhaigude D15A15

PWA EXP8

Aim: To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA.

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed.

You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.

What can we do with Service Workers?

- You can manage all network traffic of the page and do any manipulations. For example, when the page requests a CSS file, you can send plain text as a response or when the page requests an HTML file, you can send a png file as a response. You can also send a true response too.
- You can Cache

You can cache any request/response pair with Service Worker and Cache API and you can access these offline content anytime.

- You can manage Push Notifications

You can manage push notifications with Service Worker and show any information message to the user.

- You can Continue

Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

What can’t we do with Service Workers?

● You can’t access the Window
You can’t access the window, therefore, You can’t manipulate DOM elements. But, you can communicate to the window through post Message and manage processes that you
want.
● You can’t work it on 80 Port
Service Worker just can work on HTTPS protocol. But you can work on localhost during
development.

Codes:

```
//Serviceworker.js
// sw.js - Complete Service Worker for E-commerce PWA
const CACHE_NAME = "ecommerce-pwa-v2";
const urlsToCache = ["/", "/index.html", "/offline.html", "/css/main.min.css", "/js/app.min.js"];

self.addEventListener("install", (event) => {
  event.waitUntil(
  caches.open(CACHE_NAME).then((cache) => {
      console.log("[Service Worker] Caching files");
      return cache.addAll(urlsToCache);
    })
  );
});

self.addEventListener("activate", (event) => {
  event.waitUntil(
  caches.keys().then((cacheNames) => {
      return Promise.all(
        cacheNames.map((cache) => { if
        (cache !== CACHE_NAME) {
            console.log("[Service Worker] Deleting old cache:", cache);
            return caches.delete(cache);
          }
        })
      );
    })
  );
});
```

```
self.addEventListener("fetch", (event) => {
  event.respondWith(
    fetch(event.request).catch(() => caches.match(event.request).then((response) => { return
      response || caches.match("/offline.html");
    }))
  );
});
```

// Fetch Event

```
// ======================
self.addEventListener('fetch', (event) => {
const { request } = event;
const url = new URL(request.url);
// 1. Skip non-GET requests and chrome-extension if
(request.method !== 'GET' || url.protocol ===
'chrome-extension:') {
return;
}
```

// 2. API Requests (Network First with Cache Fallback)

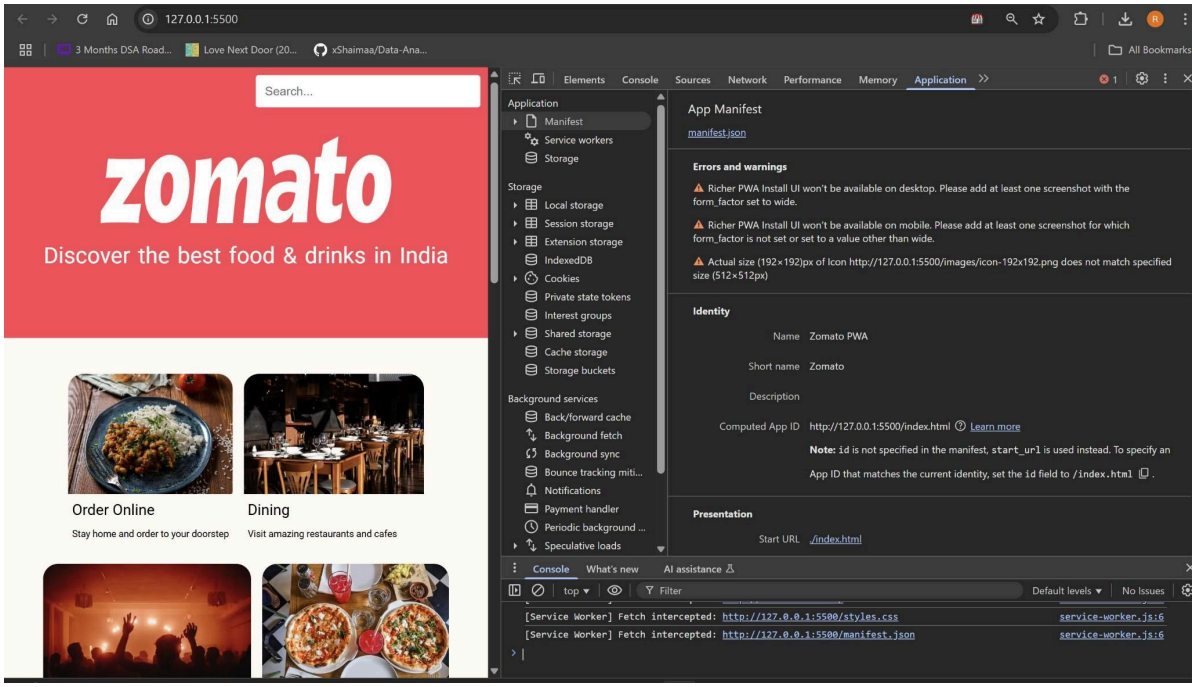
```
if (url.pathname.startsWith('/api/')) {
event.respondWith(
fetch(request)
.then(networkResponse => {
// Cache successful API responses if
(networkResponse.ok) {
const clone = networkResponse.clone();
caches.open(API_CACHE)
.then(cache => cache.put(request, clone));
}
return networkResponse;
})
.catch(() => {
// Return cached version if available return
caches.match(request)
.then(cachedResponse => cachedResponse || Response.json(
{ error: 'Network error' },
{ status: 503 }
));
})
);
```

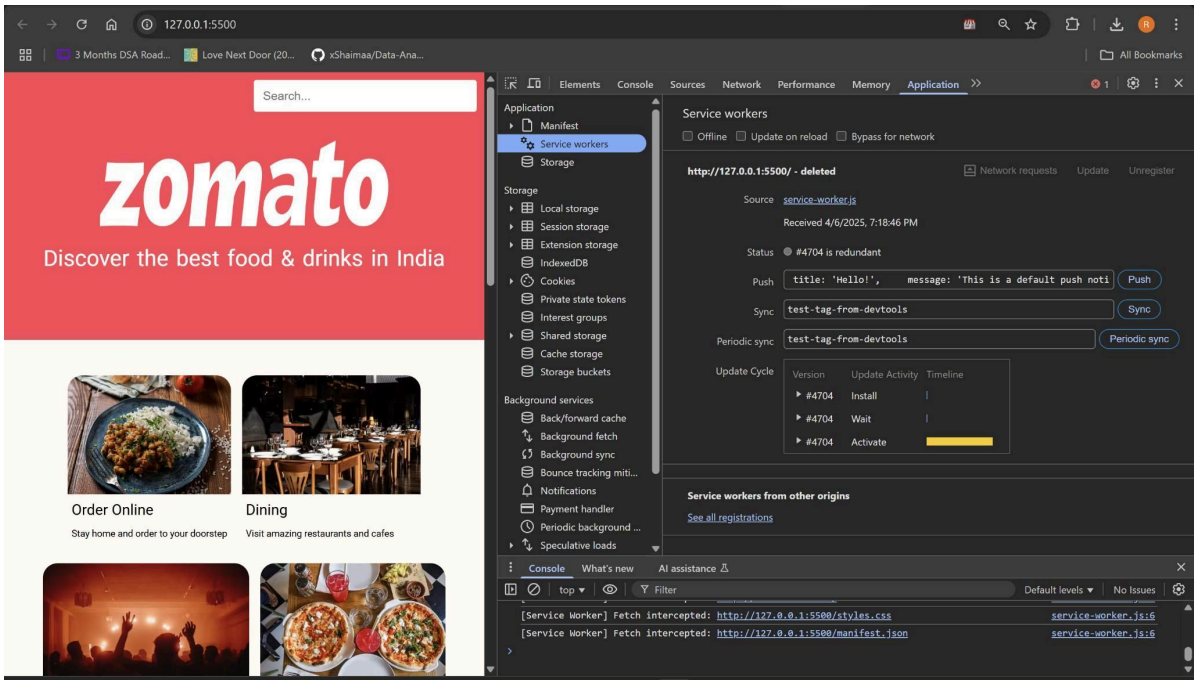
```
return;
}

// 3. Static Assets (Cache First with Network Fallback)
event.respondWith( caches.match(request)
.then(cachedResponse => {
// Return cached version if found if
(cachedResponse) {
return cachedResponse;
}
// Otherwise fetch from network
return fetch(request)
.then(networkResponse => {
// Cache successful responses if
(networkResponse.ok) {
const clone = networkResponse.clone();
caches.open(CACHE_NAME)
.then(cache => cache.put(request, clone));
}
return networkResponse;
})
.catch(() => {
// Special handling for HTML pages
if (request.headers.get('accept').includes('text/html')) {
return caches.match('/offline.html');
}
// Return placeholder for images
if (request.headers.get('accept').includes('image')) { return
caches.match('/images/placeholder-product.jpg');
}
});
})
);
});
// ======================


 // Background Sync
// ======================
self.addEventListener('sync', (event) => { if
(event.tag === 'sync-cart') { event.waitUntil(
// Get cart data from IndexedDB
```

```
getCartData()
.then(cartItems => {
return fetch('/api/cart/sync', {
method: 'POST',
headers: { 'Content-Type': 'application/json' },
body: JSON.stringify(cartItems)
});
})
.then(() => {
return showNotification('Cart Synced', 'Your cart has been
updated');
})
.catch(err => { console.error('Sync
failed:', err);
})
);
}
});
// ======================
// Push Notifications
// ======================
self.addEventListener('push', (event) => { let
data = {};
try {
data = event.data.json();
} catch (e) {
data = {
title: 'New Update',
body: 'Check out our latest products!', icon:
'/icons/icon-192x192.png',
url: '/'
};
}
const options = {
body: data.body,
icon: data.icon || '/icons/icon-192x192.png',
badge: '/icons/icon-96x96.png',
data: {
url: data.url || '/'
}
};
event.waitUntil(
self.registration.showNotification(data.title, options)
);
```

```
});
self.addEventListener('notificationclick', (event) => {
event.notification.close();
event.waitUntil(
clients.matchAll({ type: 'window' })
.then(clientList => {
for (const client of clientList) {
if (client.url === event.notification.data.url && 'focus' in
client) {
return client.focus();
}
}
if (clients.openWindow) {
return clients.openWindow(event.notification.data.url);
}
})
);
});
// ======================
// Helper Functions
// ======================
async function getCartData() {
// In a real app, you would use IndexedDB
return new Promise(resolve => { resolve([]);
});
}
async function showNotification(title, body) {
return self.registration.showNotification(title, { body });
}
```

Output:


127.0.0.1:5500
Search...
zomato
Discover the best food & drinks in India
Order Online
Stay home and order to your doorstep
Dining
Visit amazing restaurants and cafes
Elements Console Sources Network Performance Memory Application
App Manifest
manifest.json
Errors and warnings
Identity
Name Zomato PWA
Short name Zomato
Description
Computed App ID http://127.0.0.1:5500/index.html Learn more
Presentation
Start URL ./index.html
[Service Worker] Fetch intercepted: http://127.0.0.1:5500/styles.css
[Service Worker] Fetch intercepted: http://127.0.0.1:5500/manifest.json


127.0.0.1:5500
Search...
zomato
Discover the best food & drinks in India
Order Online
Stay home and order to your doorstep
Dining
Visit amazing restaurants and cafes
Elements Console Sources Network Performance Memory Application
Service workers
Offline Update on reload Bypass for network
http://127.0.0.1:5500/ - deleted
Source service-worker.js
Received 4/6/2025, 7:18:46 PM
Status #4704 is redundant
Push title: 'Hello!', message: 'This is a default push noti
Sync test-tag-from-devtools
Periodic sync test-tag-from-devtools
Update Cycle
Version Update Activity Timeline
#4704 Install
#4704 Wait
#4704 Activate
Service workers from other origins
See all registrations
[Service Worker] Fetch intercepted: http://127.0.0.1:5500/styles.css
[Service Worker] Fetch intercepted: http://127.0.0.1:5500/manifest.json

MAD & PWA Lab

Journal

Experiment No.	09
Experiment Title.	To implement Service worker events like fetch, sync and push for E-commerce PWA
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Ria Chaudhari D15A07
Eesha Chavan D15A08
Shraeyaa Dhaigude D15A15

PWA EXP9

Aim: To implement Service worker events like fetch, sync and push for E-commerce PWA.

Theory:

Service Worker

Service Worker is a script that works on browser background without user interaction independently. Also, It resembles a proxy that works on the user side. With this script, you can track network traffic of the page, manage push notifications and develop “offline first” web applications with Cache API.

Things to note about Service Worker:

- A service worker is a programmable network proxy that lets you control how network requests from your page are handled.
- Service workers only run over HTTPS. Because service workers can intercept network requests and modify responses, "man-in-the-middle" attacks could be very bad.
- The service worker becomes idle when not in use and restarts when it's next needed.

You cannot rely on a global state persisting between events. If there is information that you need to persist and reuse across restarts, you can use IndexedDB databases.

- Service workers make extensive use of promises, so if you're new to promises, then you should stop reading this and check out Promises, an introduction.

Fetch Event

You can track and manage page network traffic with this event. You can check existing cache, manage “cache first” and “network first” requests and return a response that you want.

Of course, you can use many different methods but you can find in the following example a “cache first” and “network first” approach. In this example, if the request’s and current location’s origin are the same (Static content is requested.), this is called “cacheFirst” but if you request a targeted external URL, this is called “networkFirst”.

- CacheFirst - In this function, if the received request has cached before, the cached response is returned to the page. But if not, a new response requested from the network.
- NetworkFirst - In this function, firstly we can try getting an updated response from the network, if this process completed successfully, the new response will be cached and returned. But if this process fails, we check whether the request has been cached before or not. If a cache exists, it is returned to the page, but if not, this is up to you.You can return dummy content or information messages to the page.

Sync Event
Background Sync is a Web API that is used to delay a process until the Internet connection is stable. We can adapt this definition to the real world; there is an e-mail client application that works on the browser and we want to send an email with this tool. Internet connection is broken while we are writing e-mail content and we didn't realize it. When completing the writing, we click the send button.

Push Event
This is the event that handles push notifications that are received from the server. You can apply any method with received data.
We can check in the following example.

"Notification.requestPermission();" is the necessary line to show notification to the user. If you don't want to show any notification, you don't need this line.
In the following code block is in sw.js file. You can handle push notifications with this event. In this example, I kept it simple. We send an object that has "method" and "message" properties.If the method value is "pushMessage", we open the information notification with the "message" property.

Code:

```
//Serviceworker.js
// sw.js - Complete Service Worker for E-commerce PWA const
CACHE_NAME = 'ecommerce-pwa-v2';
const API_CACHE = 'ecommerce-api-v1'; const
ASSETS_TO_CACHE = [
'/',
'/index.html', '/manifest.json',
'/offline.html',
'/css/main.min.css',
'/js/app.min.js',
'/icons/icon-192x192.png',
'/icons/icon-512x512.png',
'/images/placeholder-product.jpg'
];
// ======================
// Install Event
// ======================
self.addEventListener('install', (event) => {
event.waitUntil(
caches.open(CACHE_NAME)
.then((cache) => {
console.log('[Service Worker] Cache opened');
```

```
return cache.addAll(ASSETS_TO_CACHE);
})
.then(() => self.skipWaiting())
);
});
// ======================
// Activate Event
// ======================
self.addEventListener('activate', (event) => {
event.waitUntil(
caches.keys().then((cacheNames) => { return
Promise.all( cacheNames.map((cacheName)
=> {
if (cacheName !== CACHE_NAME && cacheName !== API_CACHE) {
console.log('[Service Worker] Deleting old cache:',
cacheName);
return caches.delete(cacheName);
}
})
);
})
.then(() => self.clients.claim())
);
});
// ======================
// Fetch Event
// ======================
self.addEventListener('fetch', (event) => {
const { request } = event;
const url = new URL(request.url);
// 1. Skip non-GET requests and chrome-extension if
(request.method !== 'GET' || url.protocol ===
'chrome-extension:') {
return;
}
// 2. API Requests (Network First with Cache Fallback) if
(url.pathname.startsWith('/api/')) { event.respondWith(
fetch(request)
.then(networkResponse => {
// Cache successful API responses if
(networkResponse.ok) {
const clone = networkResponse.clone();
caches.open(API_CACHE)
```

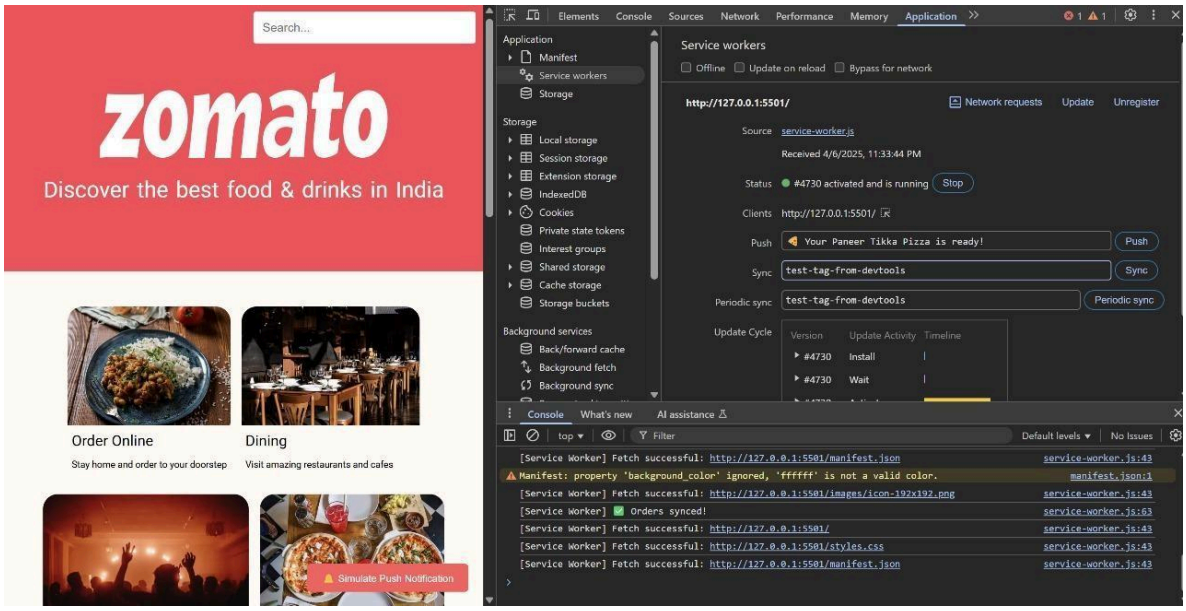
```
.then(cache => cache.put(request, clone));
}
return networkResponse;
})
.catch(() => {
// Return cached version if available return
caches.match(request)
.then(cachedResponse => cachedResponse || Response.json(
{ error: 'Network error' },
{ status: 503 }
));
})
);
return;
}
// 3. Static Assets (Cache First with Network Fallback)
event.respondWith(
caches.match(request)
.then(cachedResponse => {
// Return cached version if found if
(cachedResponse) {
return cachedResponse;
}
// Otherwise fetch from network
return fetch(request)
.then(networkResponse => {
// Cache successful responses if
(networkResponse.ok) {
const clone = networkResponse.clone();
caches.open(CACHE_NAME)
.then(cache => cache.put(request, clone));
}
return networkResponse;
})
.catch(() => {
// Special handling for HTML pages
if (request.headers.get('accept').includes('text/html')) {
return caches.match('/offline.html');
}
// Return placeholder for images
if (request.headers.get('accept').includes('image')) { return
caches.match('/images/placeholder-product.jpg');
}
});
```

```
})
);
});
// ======================
// Background Sync
// ======================
self.addEventListener('sync', (event) => { if
(event.tag === 'sync-cart') { event.waitUntil(
// Get cart data from IndexedDB
getCartData()
.then(cartItems => {
return fetch('/api/cart/sync', {
method: 'POST',
headers: { 'Content-Type': 'application/json' },
body: JSON.stringify(cartItems)
});
})
.then(() => {
return showNotification('Cart Synced', 'Your cart has been
updated');
})
.catch(err => { console.error('Sync
failed:', err);
})
);
}
});
// ======================
// Push Notifications
// ======================
self.addEventListener('push', (event) => { let
data = {};
try {
data = event.data.json();
} catch (e) {
data = {
title: 'New Update',
body: 'Check out our latest products!', icon:
'/icons/icon-192x192.png',
url: '/'
};
}
const options = {
```

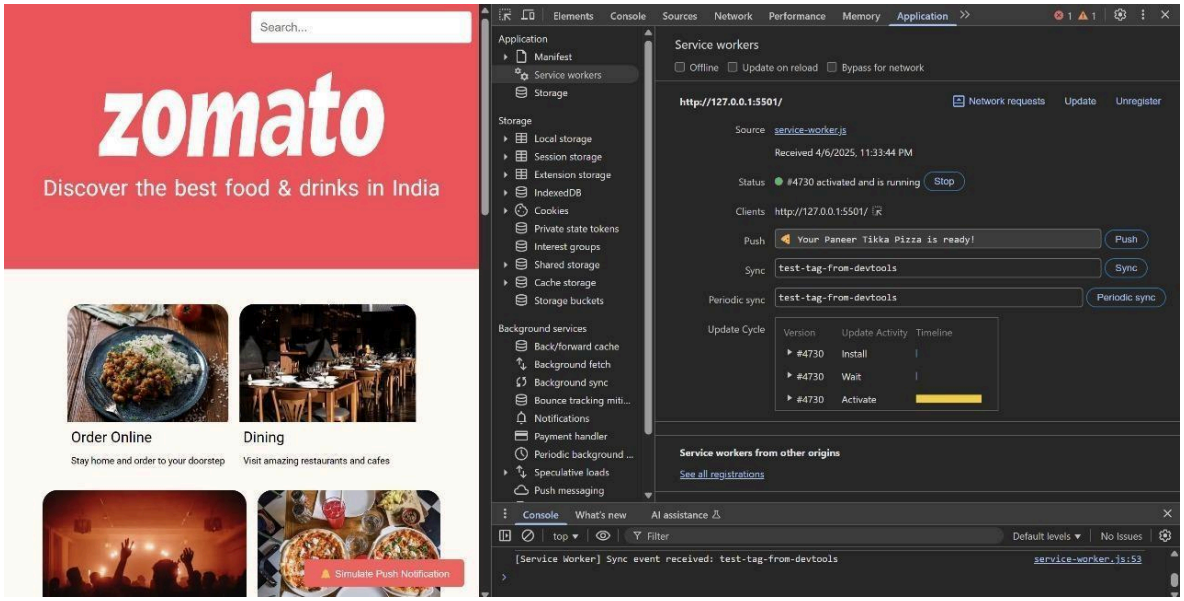
```
body: data.body,
icon: data.icon || '/icons/icon-192x192.png',
badge: '/icons/icon-96x96.png',
data: {
url: data.url || '/'
}
};
event.waitUntil(
self.registration.showNotification(data.title, options)
);
});
self.addEventListener('notificationclick', (event) => {
event.notification.close();
event.waitUntil(
clients.matchAll({ type: 'window' })
.then(clientList => {
for (const client of clientList) {
if (client.url === event.notification.data.url && 'focus' in
client) {
return client.focus();
}
}
if (clients.openWindow) {
return clients.openWindow(event.notification.data.url);
}
})
);
});
// ======================
// Helper Functions
// ======================
async function getCartData() {
// In a real app, you would use IndexedDB
return new Promise(resolve => { resolve([]);
});
}
async function showNotification(title, body) {
return self.registration.showNotification(title, { body });
}
```

Output:

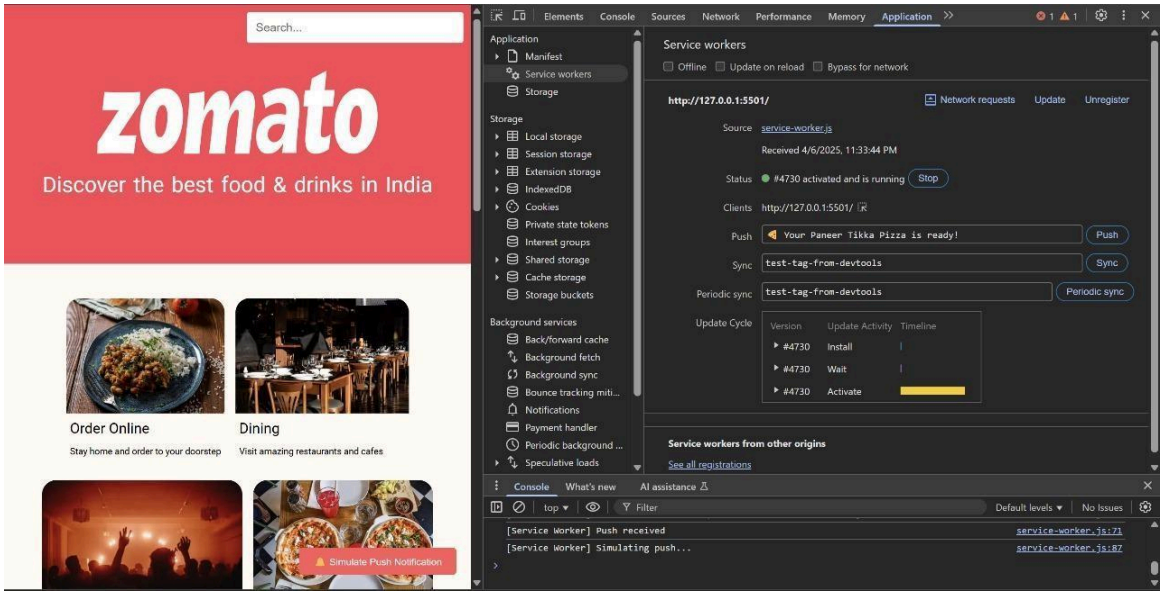
Fetch event:



Sync event:



Push:


Search...
zomato
Discover the best food & drinks in India
Order Online
Stay home and order to your doorstep
Dining
Visit amazing restaurants and cafes
Simulate Push Notification
Elements
Console
Sources
Network
Performance
Memory
Application
Service workers
Offline
Update on reload
Bypass for network
http://127.0.0.1:5501/
Network requests
Update
Unregister
Source service-worker.js
Received 4/6/2025, 11:33:44 PM
Status #4730 activated and is running
Stop
Clients http://127.0.0.1:5501/
Push
Your Paneer Tikka Pizza is ready!
Push
Sync
test-tag-from-devtools
Sync
Periodic sync
test-tag-from-devtools
Periodic sync
Update Cycle
Version
Update Activity
Timeline
#4730 Install
#4730 Wait
#4730 Activate
Service workers from other origins
See all registrations
Console
What's new
AI assistance
Filter
Default levels
No Issues
[Service Worker] Push received
service-worker.js:71
[Service Worker] Simulating push...
service-worker.js:87

MAD & PWA Lab
Journal

Experiment No.	10
Experiment Title.	To study and implement deployment of Ecommerce PWA to GitHub Pages.
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Ria Chaudhari D15A07
Eesha Chavan D15A08 Shraeyaa Dhaigude D15A15

PWA EXP10

Aim:
To study and implement deployment of Ecommerce PWA to GitHub Pages.

Theory:

GitHub Pages

Public web pages are freely hosted and easily published. Public webpages hosted directly from your GitHub repository. Just edit, push, and your changes are live.

GitHub Pages provides the following key features:

1. Blogging with Jekyll
2. Custom URL
3. Automatic Page Generator

Reasons for favoring this over Firebase:

1. Free to use
2. Right out of github
3. Quick to set up

GitHub Pages is used by Lyft, CircleCI, and HubSpot.
GitHub Pages is listed in 775 company stacks and 4401 developer stacks.

Pros

1. Very familiar interface if you are already using GitHub for your projects.
2. Easy to set up. Just push your static website to the gh-pages branch and your website is ready.
3. Supports Jekyll out of the box.
4. Supports custom domains. Just add a file called CNAME to the root of your site, add an A record in the site's DNS configuration, and you are done.

Cons

1. The code of your website will be public, unless you pay for a private repository.
2. Currently, there is no support for HTTPS for custom domains. It's probably coming soon though.
3. Although Jekyll is supported, plug-in support is rather spotty.

Firebase

The Realtime App Platform. Firebase is a cloud service designed to power real-time, collaborative applications. Simply add the Firebase library to your application to gain access to a shared data structure; any changes you make to that data are automatically synchronized with the Firebase cloud and with other clients within milliseconds.

Some of the features offered by Firebase are:

1. Add the Firebase library to your app and get access to a shared data structure. Any changes made to that data are automatically synchronized with the Firebase cloud and with other clients within milliseconds.
2. Firebase apps can be written entirely with client-side code, update in real-time out-of-the-box, interoperate well with existing services, scale automatically, and provide strong data security.
3. Data Accessibility- Data is stored as JSON in Firebase. Every piece of data has its own URL which can be used in Firebase's client libraries and as a REST endpoint. These URLs can also be entered into a browser to view the data and watch it update in real-time.

Reasons for favoring over GitHub Pages:

1. Realtime backend made easy
2. Fast and responsive

Instacart, 9GAG, and Twitch are some of the popular companies that use Firebase
Firebase has a broader approval, being mentioned in 1215 company stacks
& 4651 developers stacks

Pros

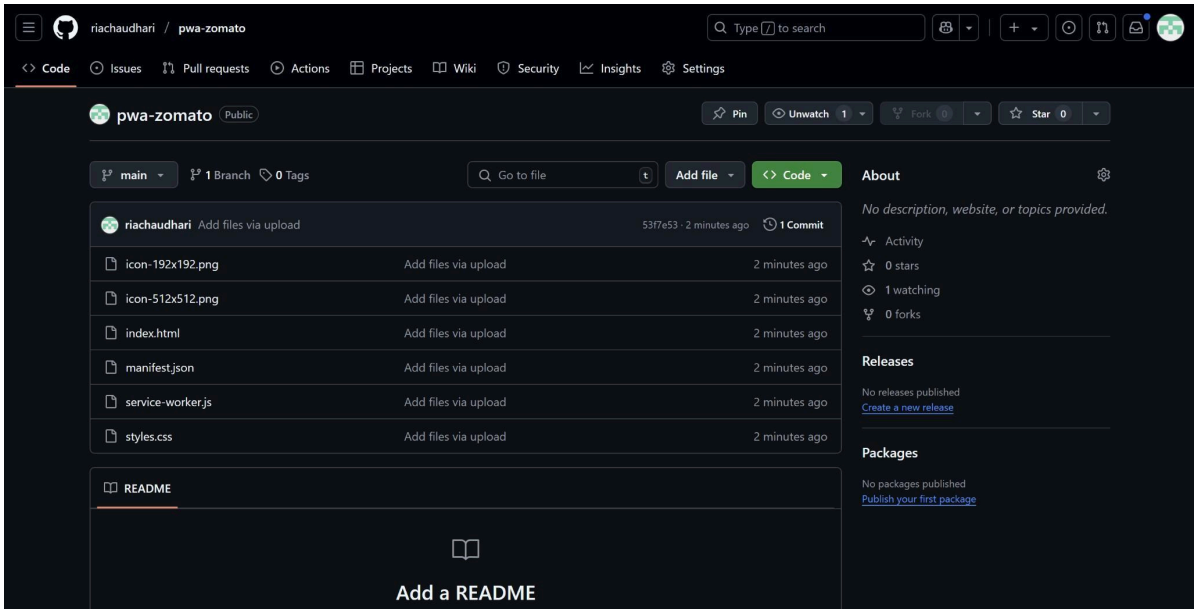
1. Hosted by Google. Enough said.
2. Authentication, Cloud Messaging, and a whole lot of other handy services will be available to you.
3. A real-time database will be available to you, which can store 1 GB of data.
4. You'll also have access to a blob store, which can store another 1 GB of data.
5. Support for HTTPS. A free certificate will be provisioned for your custom domain within 24 hours.

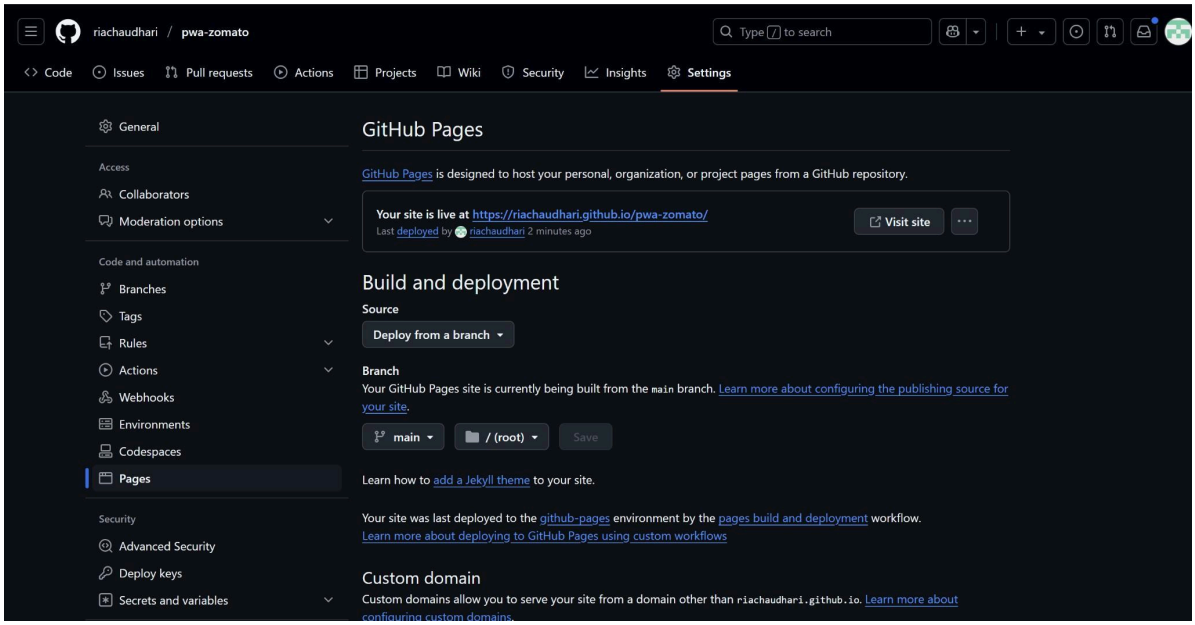
Cons

1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.
3. No in-built support for any static site generator.

Link to our GitHub repository: https://github.com/riachaudhari/pwa-zomato

Github Screenshot:





MAD & PWA Lab

Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	

Ria Chaudhari D15A07
Eesha Chavan D15A08
Shraeyaa Dhaigude D15A15

PWA EXP11

Aim: To use google Lighthouse PWA Analysis Tool to test the PWA functioning.

Theory:

Google Lighthouse :

Google Lighthouse is a tool that lets you audit your web application based on a number of parameters including (but not limited to) performance, based on a number of metrics, mobile compatibility, Progressive Web App (PWA) implementations, etc. All you have to do is run it on a page or pass it a URL, sit back for a couple of minutes and get a very elaborate report, not much short of one that a professional auditor would have compiled in about a week.

The best part is that you have to set up almost nothing to get started. Let's begin by looking at some of the top features and audit criteria used by Lighthouse.

Key Features and Audit Metrics:

Google Lighthouse has the option of running the Audit for Desktop as well as mobile version of your page(s). The top metrics that will be measured in the Audit are:

● Performance:

Measures loading speed, content display time, and overall site efficiency (score out of 100).

● PWA Score (Mobile):

Evaluates adherence to Google's Baseline PWA checklist, including Service Worker implementation, offline functionality, and script-disabled performance.

● Accessibility:

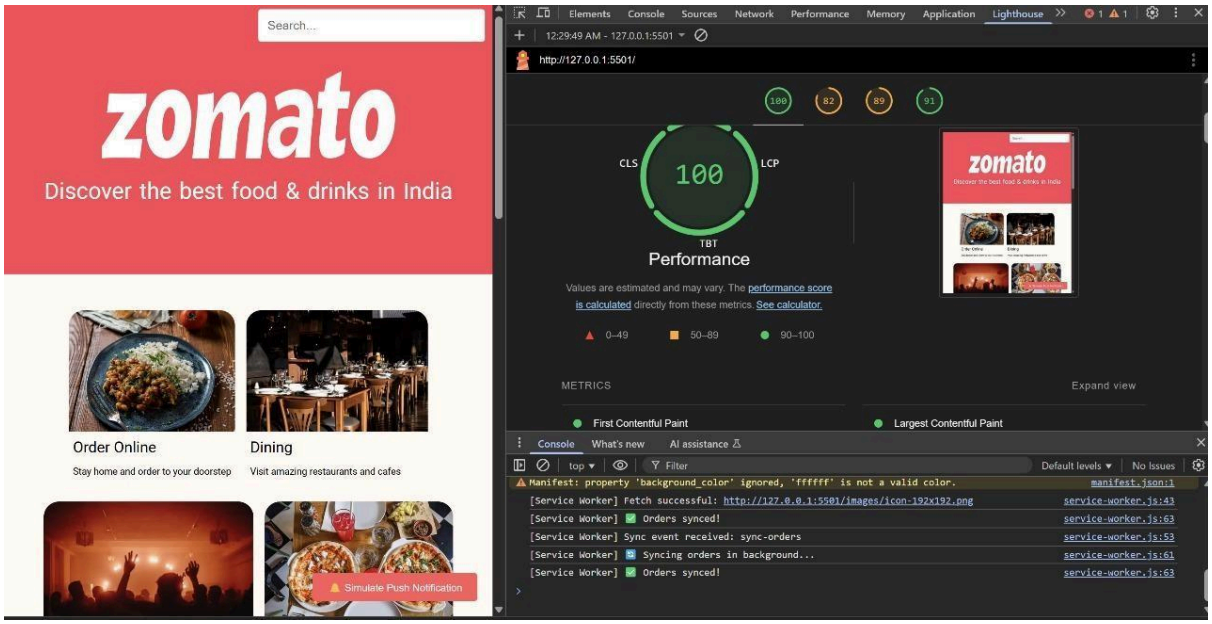
Assesses website accessibility features like aria- attributes, screen-reader compatibility, and semantic HTML tags. Scores are pass/fail.

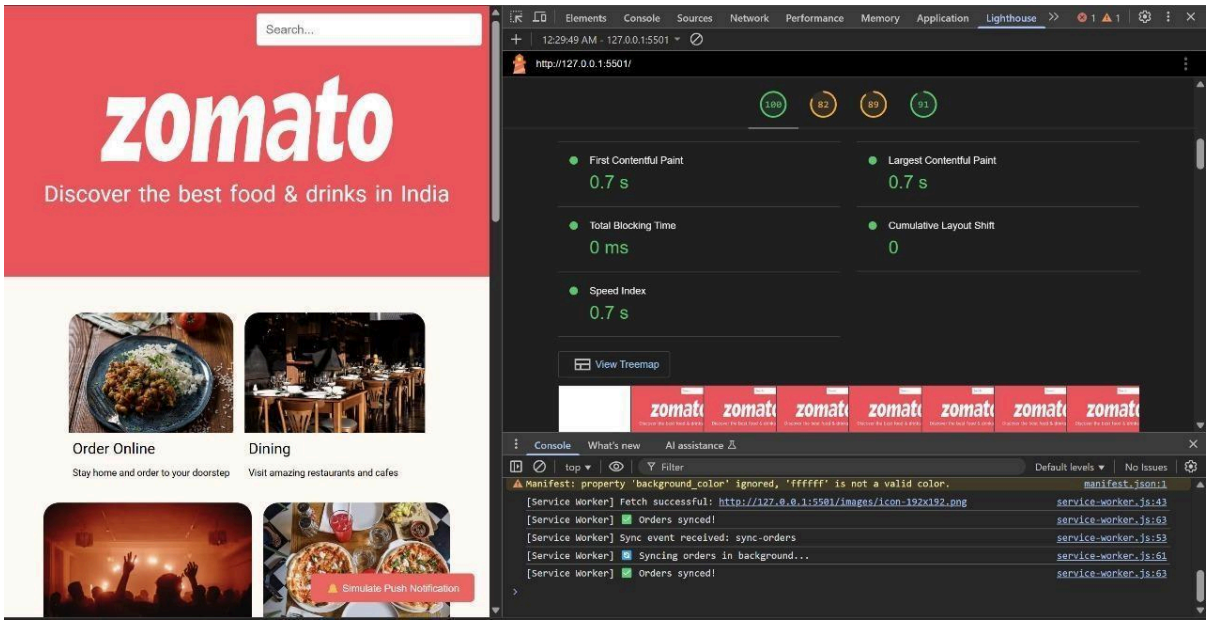
● Best Practices:

Checks adherence to industry standards, including HTTPS usage and security measures.

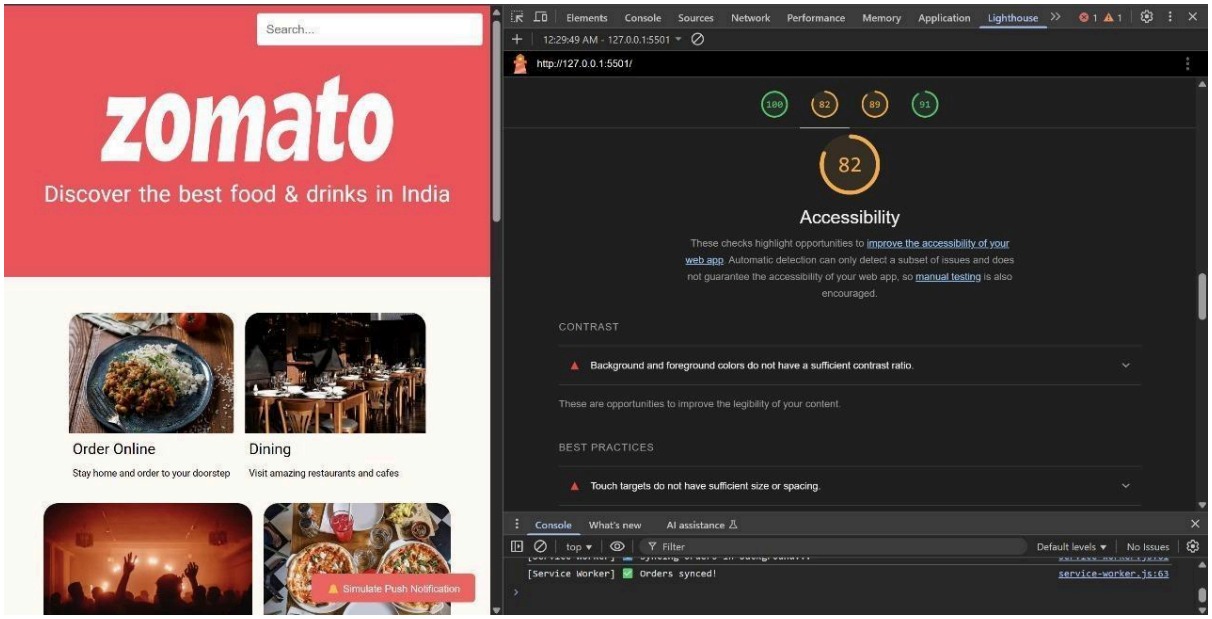
Output:

a) Performance

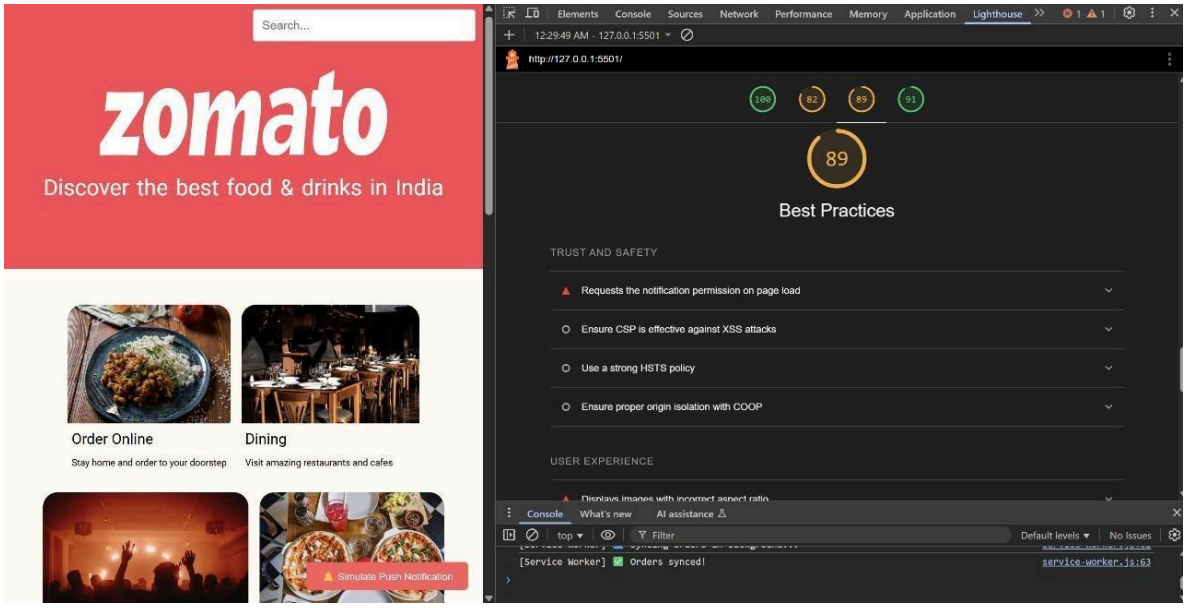




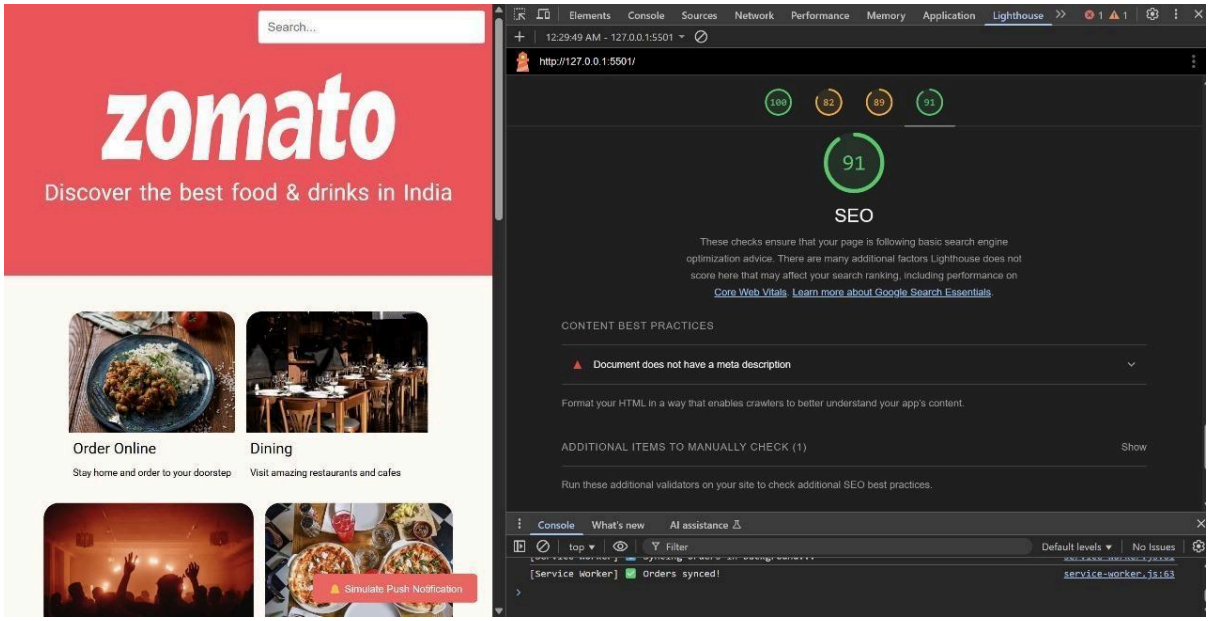
b) Accessibility



c) Best Practices



d) Search Engine Optimization (SEO)



Conclusion:

Thus we successfully used google Lighthouse PWA Analysis Tool for testing the PWA functioning.

MAD & PWA Lab
Journal

Experiment No.	Assignment-1
Assignment 1 Questions	1. Flutter Overview:Explain the key features and advantages of using Flutter for mobile app development. Discuss how the Flutter framework differs from traditional approaches and why it has gained popularity in the developer community. 2. Widget Tree and Composition: Describe the concept of the widget tree in Flutter. Explain how widget composition is used to build complex user interfaces. Provide examples of commonly used widgets and their roles in creating a widget tree. 3. State Management in Flutter: Discuss the importance of state management in Flutter applications. Compare and contrast the different state management approaches available in Flutter, such as setState, Provider, and Riverpod. Provide scenarios where each approach is suitable. 4. Firebase Integration in Flutter: Explain the process of integrating Firebase with a Flutter application. Discuss the benefits of using Firebase as a backend solution. Highlight the Firebase services commonly used in Flutter development and provide a brief overview of how data synchronization is achieved.
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	

MAD & PWA Lab
Journal

Experiment No.	Assignment-2
Assignment 2 Questions	1. Define Progressive Web App (PWA) and explain its significance in modern web development.Discuss the key characteristics that differentiate PWAs from traditional mobile apps 2. Define responsive web design and explain its importance in the context of Progressive Web Apps.Compare and contrast responsive, fluid, and adaptive web design approaches. 3. Describe the lifecycle of Service Workers, including registration, installation, and activation phases. 4. Explain the use of IndexedDB in the Service Worker for data storage.
Roll No.	08
Name	Eesha Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4:Understand various PWA frameworks and their requirements LO5: Design and Develop a responsive User Interface by applying PWA Design techniques LO6:Develop and Analyze PWA Features and deploy it over app hosting solutions
Grade:	