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 **Vaishnal Dilip Mali** 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

Project Title: Zomato **Roll No. 26**

Year/Sem/Class **:** D15A **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.

Project Title: Zomato **Roll No. 26**

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

Index

Sr. No	Experiment Title	LO	DOP	DOS	Grade
1.	To install and configure the Flutter Environment	LO1			
2.	To design Flutter UI by including common widgets.	LO2			
3.	To include icons, images, fonts in Flutter app	LO2			
4.	To create an interactive Form using form widget	LO2			
5.	To apply navigation, routing and gestures in Flutter App	LO2			
6.	To Connect Flutter UI with fireBase database	LO3			
7.	To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.	LO4			
8.	To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA	LO5			
9.	To implement Service worker events like fetch, sync and push for E-commerce PWA	LO5			
10.	To study and implement deployment of Ecommerce PWA to GitHub Pages.	LO5			
11.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.	LO6			
12.	Assignment-1	LO1,LO2 ,LO3			
13.	Assignment-2	LO4,LO5 ,LO6			

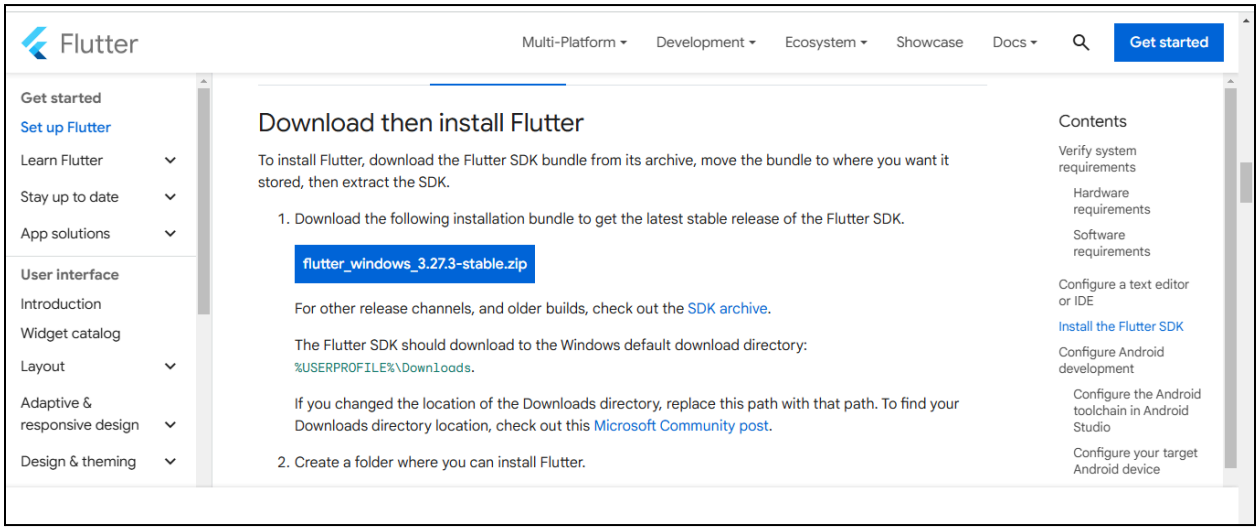
MAD & PWA Lab
Journal

Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	

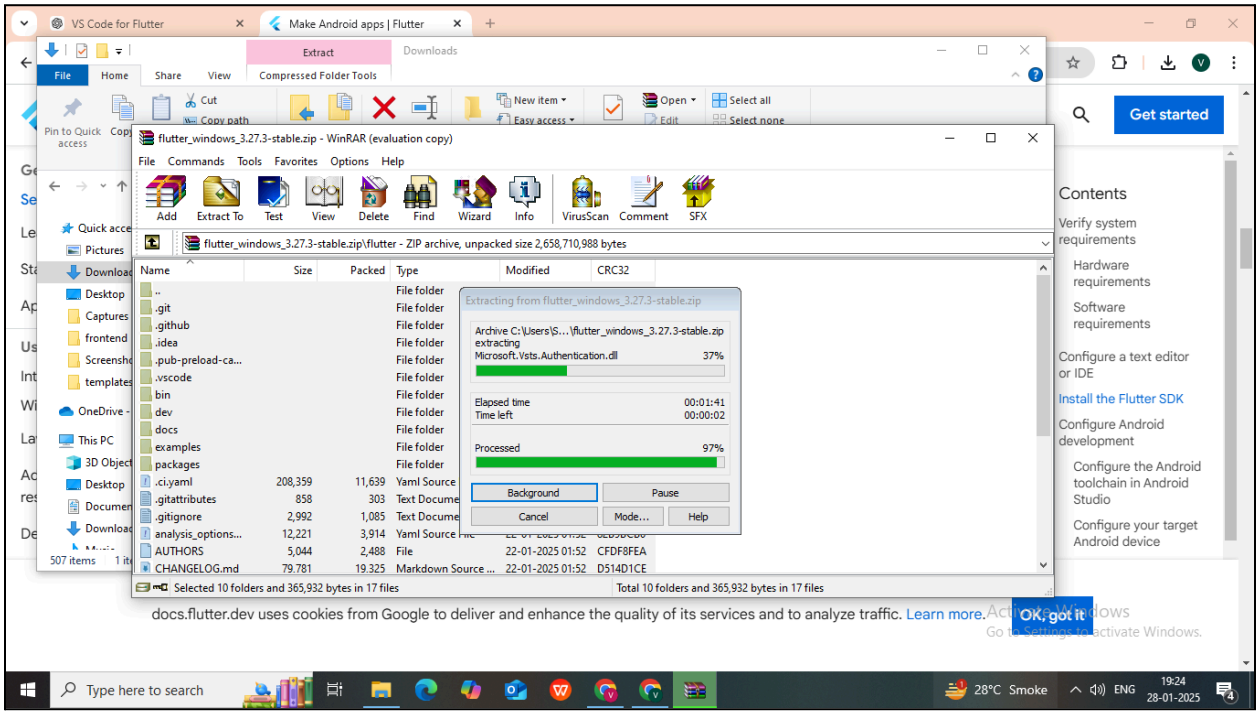
Project Title: Zomato **Roll No. 26**

Name:Vaishnal Mali
Div:D15A
Roll No.:27
Experiment 1:Installation of Flutter

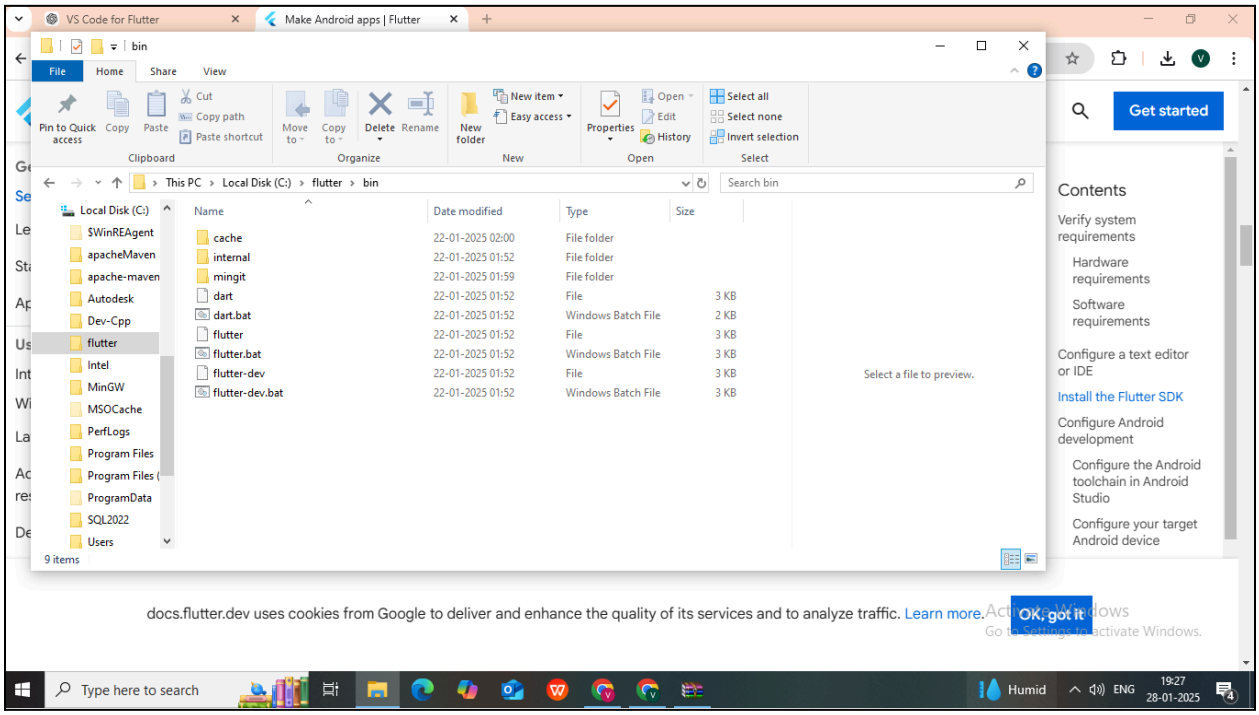
1.Download Flutter SDK



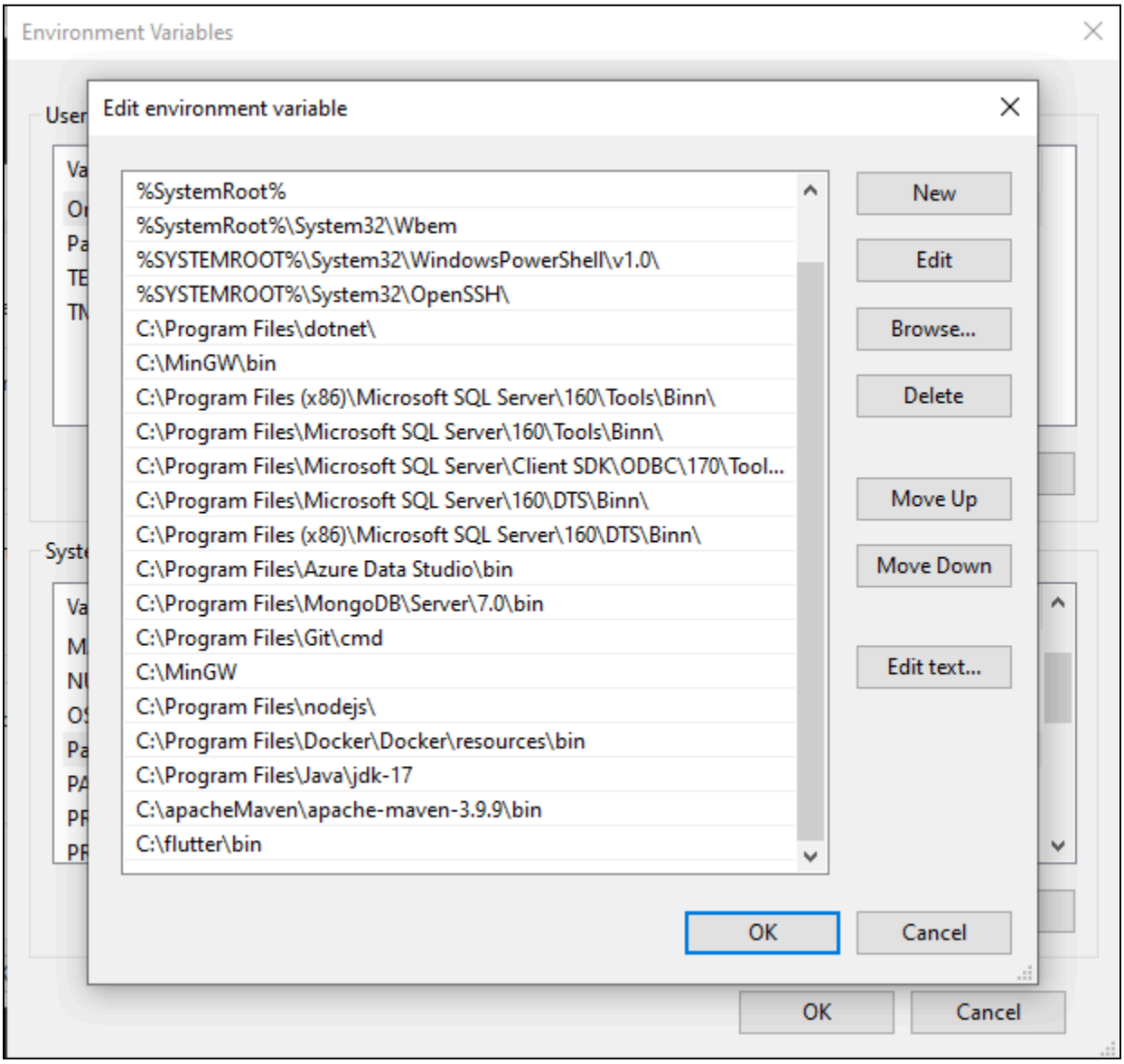
2.Extract Flutter SDK



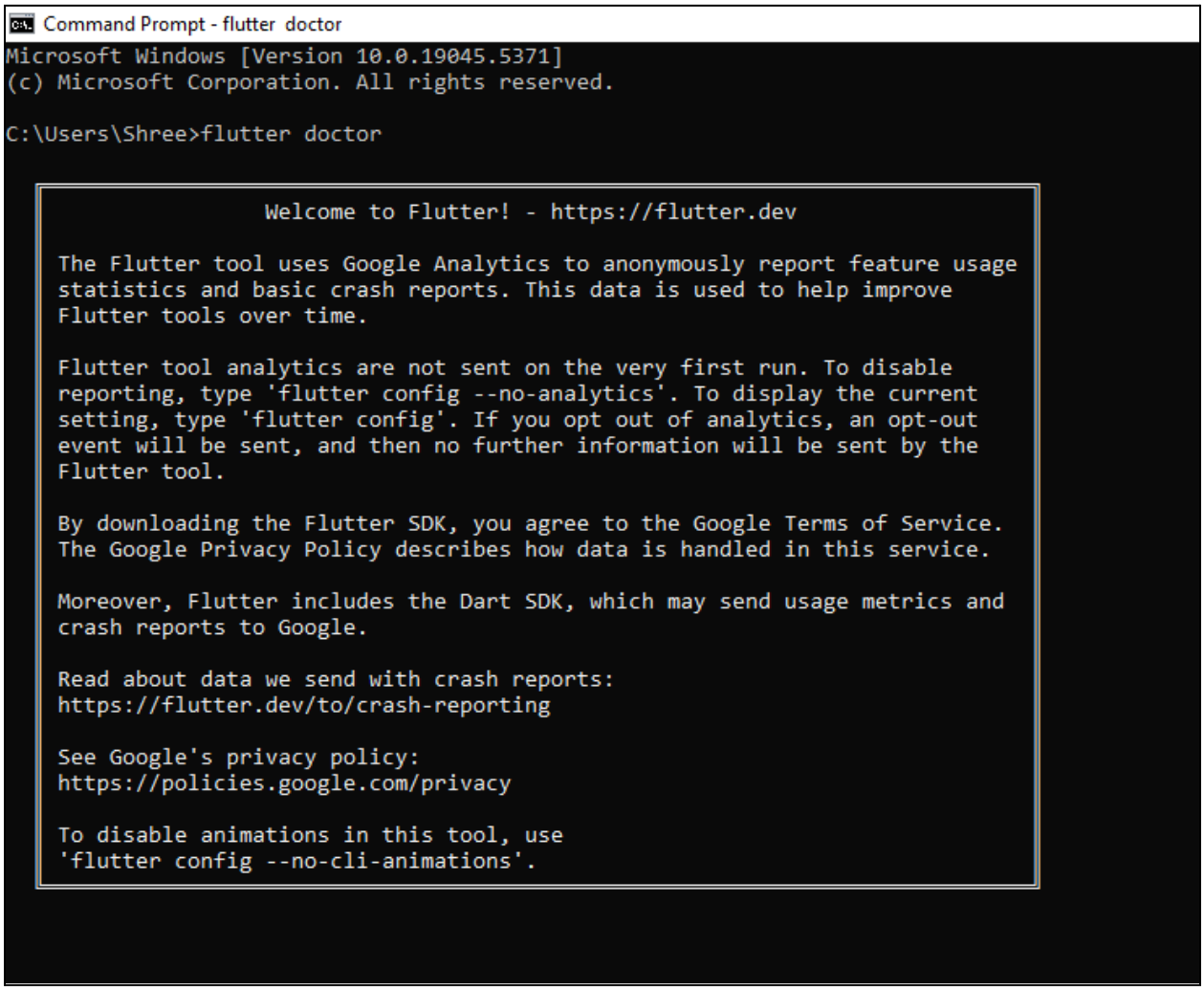
3.Copy the bin path



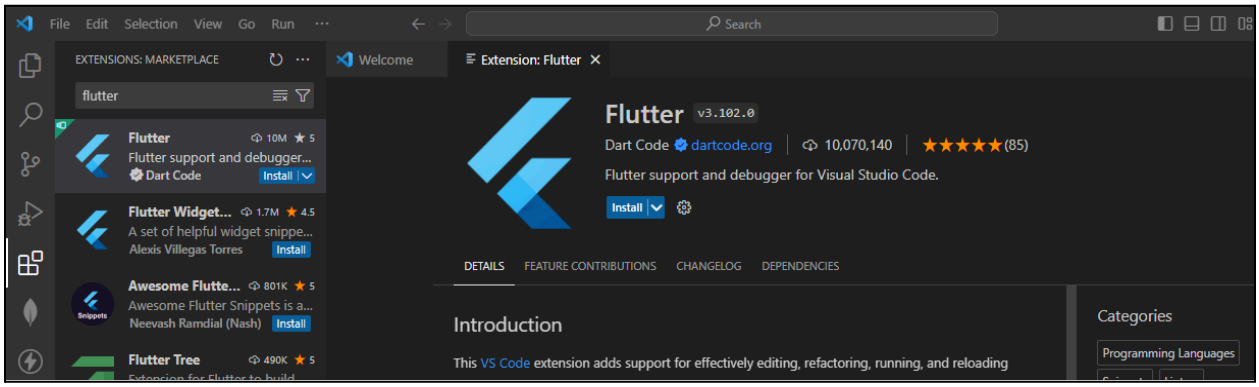
4.Add Flutter to System PATH



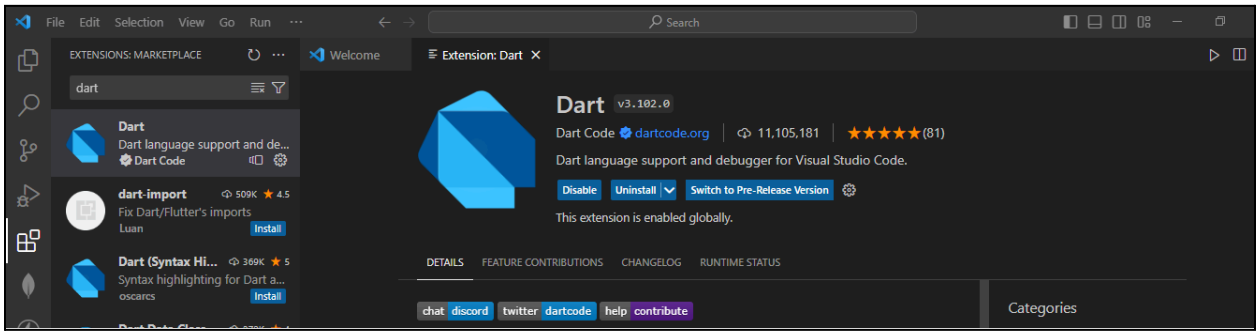
5.Verify the installation by the command **“flutter doctor”** in cmd



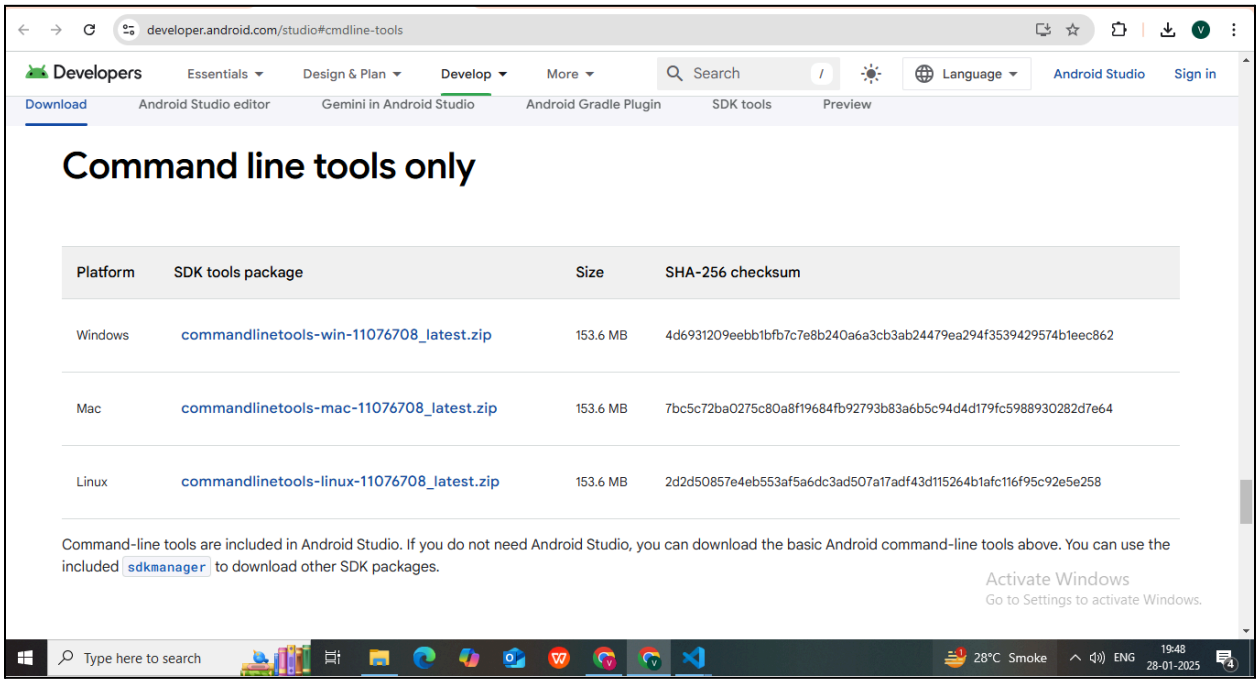
6.Install Flutter extension on VS code



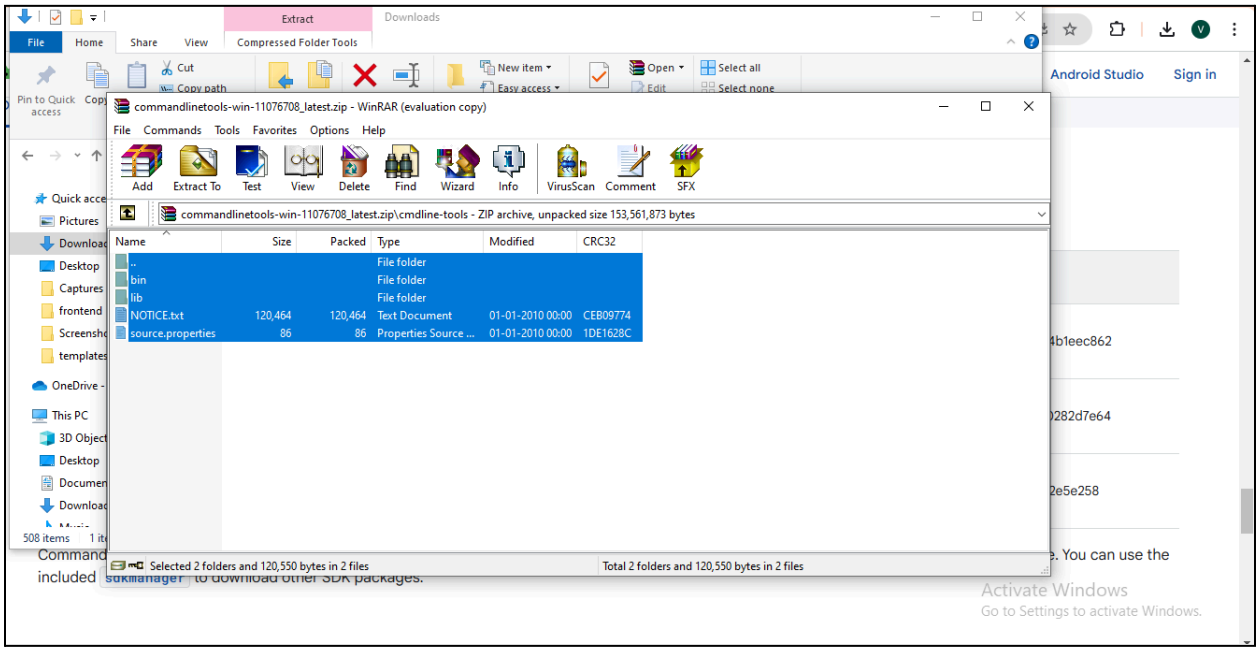
7.Install Dart extension on VS code



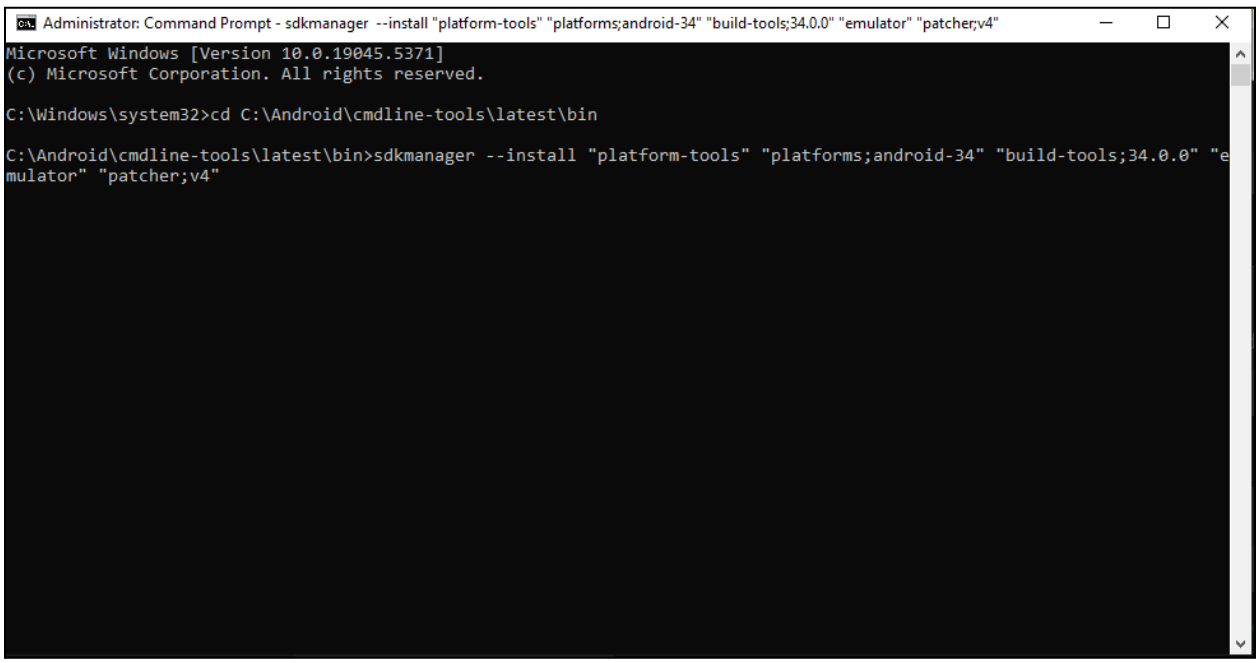
8.Install Android SDK & Command-Line Tools



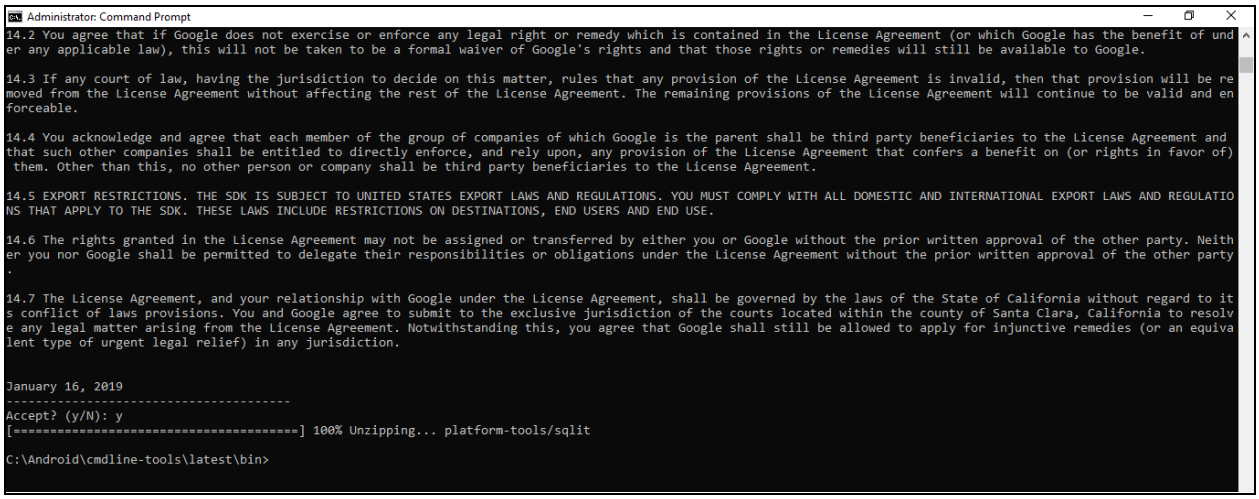
9.Extract the downloaded zip file to the specified location



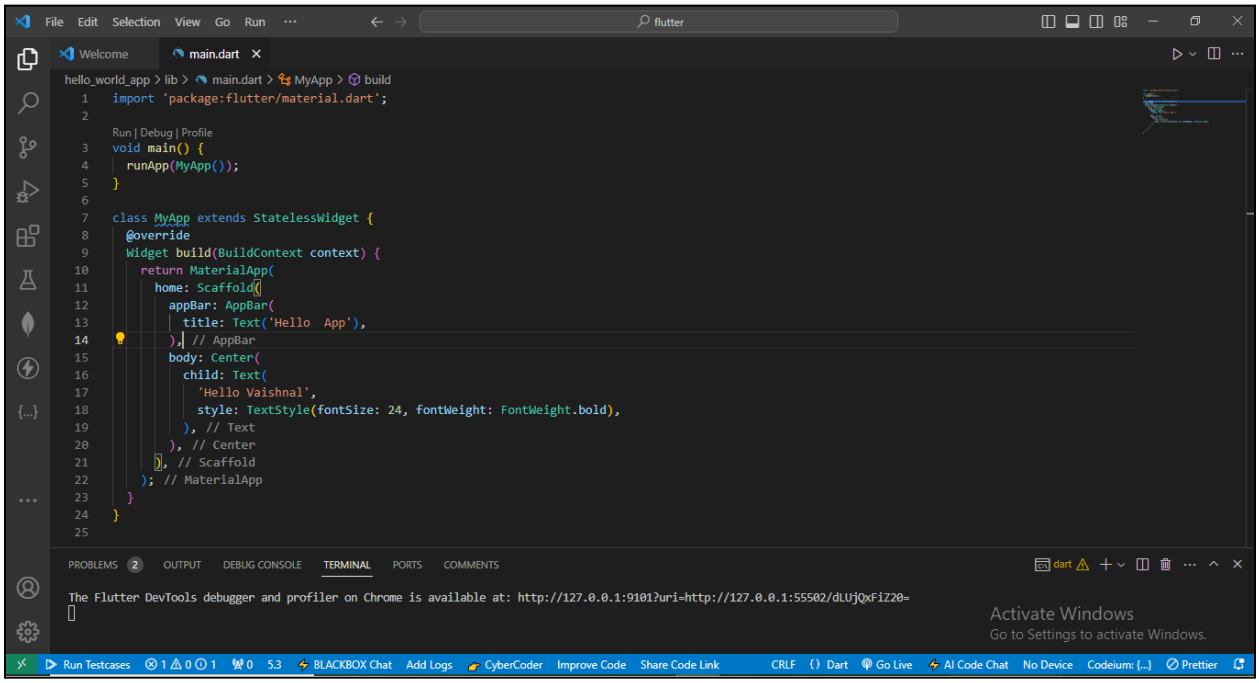
10.Open cmd as administrator and Install essential SDK components



Project Title: Zomato **Roll No. 26**



Code :



Output:

MAD & PWA Lab
Journal

Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

Name:Vaishnal Mali
Div:D15A
Roll No.:26
MPL Experiment 2

AIM : To design Flutter UI by including common widgets.

THEORY:
Flutter UI Structure
Everything in Flutter is a Widget: Widgets are the basic building blocks of a Flutter UI. They describe what the UI should look like and manage the structure and behavior of the app.
Composition over Inheritance: Flutter follows a composition-based approach where smaller widgets combine to form complex UIs.
Widget Tree: Every Flutter UI is built using a widget tree, where widgets are arranged hierarchically.
Types of Widgets
Flutter has two main types of widgets:
Stateless Widgets: Do not store any state and remain static unless rebuilt.
Example: Text, Icon, Container
Stateful Widgets: Can change dynamically based on user interactions or data changes.
Example: TextField, Checkbox, Slider

Common Flutter Widgets and Their Properties
1. Basic Widgets
Container: Holds and styles child widgets (padding, margin, borders).
Container(width: 200, height: 100, color: Colors.blue, child: Text("Hello"))

Text: Displays static text with styles.
Text("Hello, Flutter!", style: TextStyle(fontSize: 20, color: Colors.red))

Image: Displays images from assets/network/memory.
Image.network('https://example.com/image.png', width: 100, height: 100)

2. Layout Widgets
Row: Arranges widgets horizontally.

Row(children: [Icon(Icons.star), Text("Flutter")])

Column: Arranges widgets vertically.
Column(children: [Text("Welcome"), ElevatedButton(onPressed: () {}, child: Text("Click Me"))])

Stack: Overlays widgets on top of each other.
Stack(children: [Container(color: Colors.blue, width: 200, height: 200), Positioned(top: 50, child: Icon(Icons.star))])

3. Interactive Widgets
ElevatedButton: Raised button with elevation.
ElevatedButton(onPressed: () {}, child: Text("Click Me"))

TextField: User input field.
TextField(decoration: InputDecoration(labelText: 'Enter Name'))

Checkbox: Toggle checkbox.
Checkbox(value: true, onChanged: (bool? newValue) {})

4. Navigation Widgets
Navigator.push: Navigates to another screen.
Navigator.push(context, MaterialPageRoute(builder: (context) => SecondScreen()))

BottomNavigationBar: Bottom menu navigation.
BottomNavigationBar(items: [BottomNavigationBarItem(icon: Icon(Icons.home), label: "Home")])

5. List and Grid Widgets
ListView: Scrollable list.
ListView(children: [ListTile(title: Text("Item 1"))])

GridView: Displays items in a grid.
GridView.count(crossAxisCount: 2, children: [Container(color: Colors.blue)])

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

```
void main() {
  runApp(const MyApp());
}
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false, // Removes the debug banner
      title: 'Flutter UI',
      theme: ThemeData(
        primarySwatch: Colors.blue, // Sets theme color
      ),
      home: const HomeScreen(), // Calls the home screen widget
    );
  }
}
class HomeScreen extends StatelessWidget {
  const HomeScreen({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Flutter UI Example'),
        centerTitle: true,
      ),
      body: Center(
        child: Text(
          'Hello, Flutter!',
          style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
        ),
      ),
    );
  }
}
```

CODE:

1.Homepage

```
// lib/screens/home_screen.dart
import 'package:flutter/material.dart';

class HomeScreen extends
StatelessWidget {
  const HomeScreen({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Column(
          children: [
            Padding(
              padding: const
EdgeInsets.all(16.0),
              child: Row(
                children: [
                  const
Icon(Icons.person_outline),
                  const Spacer(),
                  IconButton(
                    icon: const Icon(Icons.menu),
                    onPressed: () {},
                  ),
                ],
              ),
            ),
            Padding(
              padding: const
EdgeInsets.symmetric(horizontal: 16.0),
              child: TextField(
                decoration: InputDecoration(
                  hintText: 'Restaurant name,
cuisine, or a dish...',
                  prefixIcon: const
Icon(Icons.search),
                  border: OutlineInputBorder(
                    borderRadius:
BorderRadius.circular(10),
                    borderSide: BorderSide.none,
                  ),
                  filled: true,
                  fillColor: Colors.grey[200],
                ),
              ),
            ),
            const SizedBox(height: 16),
            SingleChildScrollView(
              scrollDirection: Axis.horizontal,
              padding: const
EdgeInsets.symmetric(horizontal: 16),
              child: Row(
                children: [
                  _buildFilterChip('PRO'),
                  _buildFilterChip('Cuisine'),
                  _buildFilterChip('Rating'),
                  _buildFilterChip('Popular'),
                ],
              ),
            ),
            const SizedBox(height: 16),
            _buildPromoBanner(),
            const SizedBox(height: 20),
            const Padding(
              padding:
EdgeInsets.symmetric(horizontal: 16.0),
              child: Text(
                'Eat what makes you happy',
                style: TextStyle(
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ),
```

```
          const SizedBox(height: 16),
          Expanded(
            child: GridView.count(
              crossAxisCount: 4,
              padding: const
EdgeInsets.symmetric(horizontal: 16),
              children: [

_buildCategoryItem('Healthy','assests/icon
s/healthy.jpeg'),
                _buildCategoryItem('Biryani',
'assests/icons/biryani.jpeg'),
                _buildCategoryItem('Pizza',
'assests/icons/pizza.jpeg'),
                _buildCategoryItem('Noodles',
'assests/icons/noodles.jpeg'),
                _buildCategoryItem('Chicken',
'assests/icons/chiken.jpeg'),
                _buildCategoryItem('Burger',
'assests/icons/burger.jpeg'),
                _buildCategoryItem('Cake',
'assests/icons/cake.jpeg'),
                _buildCategoryItem('More',
'assests/icons/healthy.jpeg'),
              ],
            ),
          ),
        ],
      ),
    ),
    bottomNavigationBar:
BottomNavigationBar(
      selectedItemColor: Colors.red,
      unselectedItemColor: Colors.grey,
      items: const [
        BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
        BottomNavigationBarItem(icon:
Icon(Icons.search), label: 'Search'),
        BottomNavigationBarItem(icon:
Icon(Icons.person), label: 'Profile'),
      ],
    ),
  );
 }

 Widget _buildFilterChip(String label) {
  return Container(
    margin: const EdgeInsets.only(right:
8),
    child: Chip(
      label: Text(label),
      backgroundColor: Colors.grey[200],
    ),
  );
 }

 Widget _buildPromoBanner() {
  return Container(
    margin: const
EdgeInsets.symmetric(horizontal: 16),
    child: Row(
      children: [
        Expanded(
          child: Container(
            height: 100,
            decoration: BoxDecoration(
              color: Colors.orange[300],
              borderRadius:
BorderRadius.circular(10),
            ),
            child: const Center(
              child: Text(
                '60% OFF\nnew something',
                style: TextStyle(
                  color: Colors.white,
                  fontSize: 16,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ),
          ),
        ),
        const SizedBox(width: 16),
```

```
          Expanded(
            child: Container(
              height: 100,
              decoration: BoxDecoration(
                color: Colors.blue[200],
                borderRadius:
BorderRadius.circular(10),
              ),
              child: const Center(
                child: Text(
                  'billig\ndiscounts',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 16,
                    fontWeight: FontWeight.bold,
                  ),
                ),
              ),
            ),
          ),
        ],
      ),
    );
  }

  Widget _buildCategoryItem(String name,
String imagePath) {
  return Column(
    children: [
      Image.asset(imagePath, width: 50,
height: 50), // Load image instead of an
icon
      SizedBox(height: 5),
      Text(name, style: TextStyle(fontSize:
14)),
    ],
  );
}

}
```

2.eat-healthy screen

```
//
lib/screens/eat_healthy/eat_healthy_screen
.dart
import 'package:flutter/material.dart';

class EatHealthyScreen extends
StatelessWidget {
  const EatHealthyScreen({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Column(
          children: [
            _buildHeader(),
            _buildDeliveryOptions(),
            Expanded(
              child: SingleChildScrollView(
                child: Column(
                  crossAxisAlignment:
CrossAxisAlignment.start,
                  children: [
                    _buildSegmentedControl(),
                    const SizedBox(height: 16),
                    _buildDishCard(
                      'Plant Protein Bowl',
                      '★ 4.2',
                      'Veg preparation having raw
plant based...',
                      'assests/bowl.jpeg',
                    ),
                    _buildDishCard(
                      'Spring Veg Plater',
                      '★ 4.5',
                      'Veg preparation having raw
plant based...',
                      'assests/platter.jpeg',
                    ),
                  ],
                ),
              ),
            ),
            _buildBottomNavigationBar(),
          ],
        ),
      ),
    );
  }

  Widget _buildHeader() {
    return Container(
      padding: const EdgeInsets.all(16),
      child: Column(
        crossAxisAlignment:
CrossAxisAlignment.start,
        children: [
          Row(
            children: [
              IconButton(
                icon: const
Icon(Icons.arrow_back),
                onPressed: () {},
              ),
              const Text(
                'Eat Healthy',
                style: TextStyle(
                  fontSize: 20,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ],
          ),
          const Spacer(),
          Container(
            padding: const
EdgeInsets.symmetric(
              horizontal: 8,
              vertical: 4,
            ),
```

```
            decoration: BoxDecoration(
              color: Colors.green,
              borderRadius:
BorderRadius.circular(4),
            ),
            child: const Row(
              children: [
                Text(
                  '4.2 ★',
                  style: TextStyle(color:
Colors.white),
                ),
                Icon(
                  Icons.arrow_forward_ios,
                  size: 12,
                  color: Colors.white,
                ),
              ],
            ),
          ),
        ],
      ),
      const Text(
        'Naturally, Hyderabad',
        style: TextStyle(color:
Colors.grey),
      ),
    ],
  ),
 );
}

Widget _buildDeliveryOptions() {
 return Container(
  padding: const
EdgeInsets.symmetric(horizontal: 16),
  child: Row(
   children: [
    _buildDeliveryOption(
     Icons.delivery_dining,
     'DELIVERY',
     isSelected: true,
    ),
    const SizedBox(width: 16),
    _buildDeliveryOption(
     Icons.restaurant,
     'DINING',
    ),
    const SizedBox(width: 16),
    _buildDeliveryOption(
     Icons.star,
     'REVIEWS',
    ),
   ],
  ),
 );
}

Widget _buildDeliveryOption(IconData
icon, String label, {bool isSelected =
false}) {
 return Expanded(
  child: Container(
   padding: const
EdgeInsets.symmetric(vertical: 8),
   decoration: BoxDecoration(
    border: Border(
     bottom: BorderSide(
      color: isSelected ? Colors.black :
Colors.grey,
      width: 2,
     ),
    ),
   ),
   child: Row(
    mainAxisAlignment:
MainAxisAlignment.center,
    children: [
     Icon(icon, color: isSelected ?
Colors.black : Colors.grey),
     const SizedBox(width: 4),
     Text(
      label,
      style: TextStyle(
```

```
            color: isSelected ? Colors.black :
Colors.grey,
            fontWeight: isSelected ?
FontWeight.bold : FontWeight.normal,
          ),
        ),
      ],
    ),
  ),
 );
}

Widget _buildSegmentedControl() {
 return Container(
  padding: const
EdgeInsets.symmetric(horizontal: 16),
  child: Row(
   children: [
    _buildSegmentButton('Full Menu',
isSelected: true),
    _buildSegmentButton('Healthy'),
   ],
  ),
 );
}

Widget _buildSegmentButton(String
label, {bool isSelected = false}) {
 return Expanded(
  child: Container(
   padding: const
EdgeInsets.symmetric(vertical: 8),
   decoration: BoxDecoration(
    border: Border(
     bottom: BorderSide(
      color: isSelected ? Colors.black :
Colors.grey,
      width: 2,
     ),
    ),
   ),
   child: Text(
    label,
    textAlign: TextAlign.center,
    style: TextStyle(
     color: isSelected ? Colors.black :
Colors.grey,
     fontWeight: isSelected ?
FontWeight.bold : FontWeight.normal,
    ),
   ),
  ),
 );
}

Widget _buildDishCard(String title,
String rating, String description, String
imagePath) {
 return Container(
  padding: const EdgeInsets.all(16),
  child: Row(
   crossAxisAlignment:
CrossAxisAlignment.start,
   children: [
    Expanded(
     flex: 2,
     child: Column(
      crossAxisAlignment:
CrossAxisAlignment.start,
      children: [
       Row(
        children: [
         const Icon(Icons.circle, color:
Colors.green, size: 12),
         const SizedBox(width: 4),
         Text(
          title,
          style: const TextStyle(
           fontWeight:
FontWeight.bold,
           fontSize: 16,
          ),
         ),
        ],
```

```
            ),
            const SizedBox(height: 4),
            Container(
              padding: const
EdgeInsets.symmetric(
                horizontal: 4,
                vertical: 2,
              ),
              decoration: BoxDecoration(
                color: Colors.green[100],
                borderRadius:
BorderRadius.circular(4),
              ),
              child: Text(
                rating,
                style: const
TextStyle(fontSize: 12),
              ),
            ),
            const SizedBox(height: 4),
            Text(
              description,
              style: const TextStyle(
                color: Colors.grey,
                fontSize: 12,
              ),
            ),
          ],
        ),
      ),
      const SizedBox(width: 16),
      Stack(
        alignment:
Alignment.bottomRight,
        children: [
          ClipRRect(
            borderRadius:
BorderRadius.circular(8),
            child: Image.asset(
              imagePath,
              width: 100,
              height: 100
                fit: BoxFit.cover,
              ),
            ),
            Container(
              padding: const
EdgeInsets.symmetric(
                horizontal: 12,
                vertical: 4,
              ),
              decoration: BoxDecoration(
                color: Colors.white,
                borderRadius:
BorderRadius.circular(4),
              ),
              child: const Text(
                'ADD',
                style: TextStyle(
                  color: Colors.green,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ),
          ],
        ),
      ],
    ),
  );
}

Widget _buildBottomNavigationBar() {
  return BottomNavigationBar(
    selectedItemColor: Colors.black,
    unselectedItemColor: Colors.grey,
    items: const [
      BottomNavigationBarItem(
        icon: Icon(Icons.home),
        label: '',
      ),
      BottomNavigationBarItem(
        icon: Icon(Icons.search),
        label: '',
      ),
```

```
        BottomNavigationBarItem(
          icon: Icon(Icons.person),
          label: '',
        ),
      ],
    );
  }
}
```

3.dish-detail screen

```
// lib/screens/dish_detail_screen.dart
import 'package:flutter/material.dart';

class DishDetailScreen extends StatefulWidget {
  const DishDetailScreen({Key? key}) : super(key: key);

  @override
  State<DishDetailScreen> createState() => _DishDetailScreenState();
}

class _DishDetailScreenState extends State<DishDetailScreen> {
  int quantity = 1;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.transparent,
        elevation: 0,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.black),
          onPressed: () => Navigator.pop(context),
        ),
        actions: [
          Container(
            margin: const EdgeInsets.all(8),
            padding: const EdgeInsets.symmetric(horizontal: 8, vertical: 4),
            decoration: BoxDecoration(
              color: Colors.green,
              borderRadius: BorderRadius.circular(12),
            ),
            child: const Row(
              children: [
                Text(
                  '4.2 ',
                  style: TextStyle(color: Colors.white),
                ),
                Icon(Icons.star, color: Colors.white, size: 16),
              ],
            ),
          ),
        ],
      ),
```

```
      body: SingleChildScrollView(
        child: Column(
          crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
            Image.asset(
              'assests/bowl.jpeg',
              width: double.infinity,
              height: 200,
              fit: BoxFit.cover,
            ),
            Padding(
              padding: const
EdgeInsets.all(16.0),
              child: Column(
                crossAxisAlignment:
CrossAxisAlignment.start,
                children: [
                  const Row(
                    children: [
                      Icon(Icons.circle, color:
Colors.green, size: 12),
                      SizedBox(width: 8),
                      Text(
                        'Plant Protein Bowl',
                        style: TextStyle(
                          fontSize: 20,
                          fontWeight:
FontWeight.bold,
                        ),
                      ),
                    ],
                  ),
                  const SizedBox(height: 8),
                  const Text(
                    'Add On',
                    style: TextStyle(
                      fontSize: 16,
                      fontWeight:
FontWeight.w500,
                    ),
                  ),
                  const SizedBox(height: 8),
                  _buildAddOnOption('Veg
protein', '₹40'),
                  _buildAddOnOption('Pepper',
'₹40'),
                  _buildAddOnOption('Extra
Veggies', '₹40'),

_buildAddOnOption('Mushrooms', '₹40'),
                  _buildAddOnOption('Corn',
'₹40'),
                  _buildAddOnOption('Chilli
Powder', '₹40'),
                  const SizedBox(height: 16),
                  const Text(
                    'Choose Your Protein',
                    style: TextStyle(
                      fontSize: 16,
                      fontWeight:
FontWeight.w500,
                    ),
                  ),
                  const SizedBox(height: 8),
                  Row(
                    children: [
                      const Text('-'),
                      Container(
                        margin: const
EdgeInsets.symmetric(horizontal: 8),
                        padding: const
EdgeInsets.symmetric(
                          horizontal: 16,
                          vertical: 4,
                        ),
                        decoration: BoxDecoration(
                          border: Border.all(color:
Colors.grey),
                          borderRadius:
BorderRadius.circular(4),
                        ),
                        child: Text('$quantity'),
                      ),
```

```
                const Text('+'),
                const Spacer(),
                ElevatedButton(
                  onPressed: () {},
                  style:
ElevatedButton.styleFrom(
                    backgroundColor:
Colors.red,
                    shape:
RoundedRectangleBorder(
                      borderRadius:
BorderRadius.circular(8),
                    ),
                  ),
                  child: const Text(
                    'Add ₹199',
                    style: TextStyle(color:
Colors.white),
                  ),
                ),
              ],
            ),
          ],
        ),
      ),
    ),
  );
 }

  Widget _buildAddOnOption(String title,
String price) {
   return Container(
     margin: const EdgeInsets.only(bottom:
8),
     child: Row(
       children: [
         Checkbox(
           value: false,
           onChanged: (value) {},
         ),
         Text(title),
           const Spacer(),
           Text(price),
         ],
       ),
     );
   }
}
```

4.checkout-screen

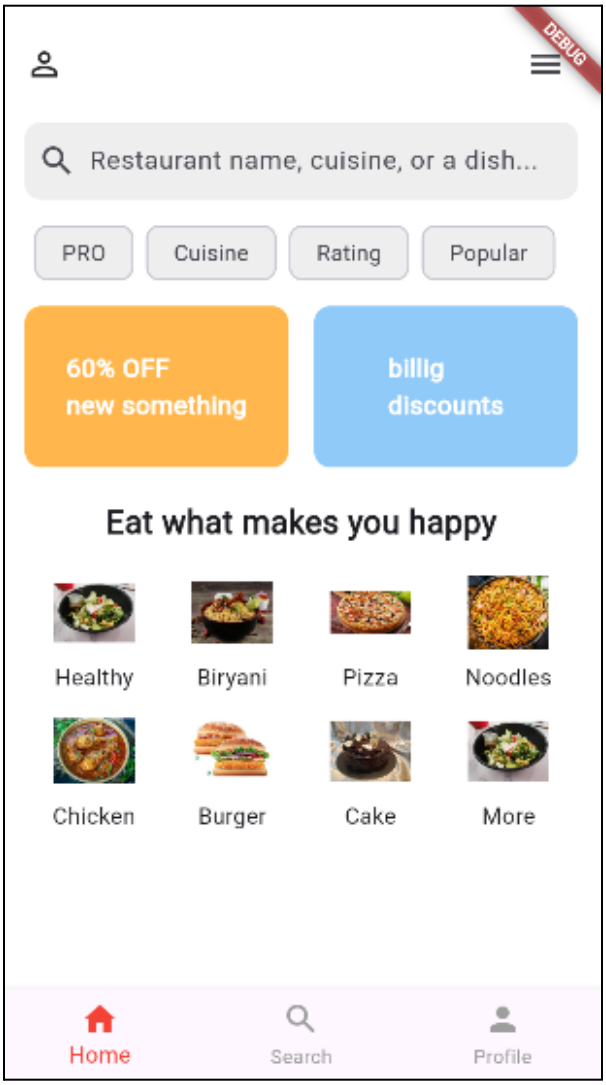
```
// lib/screens/checkout_screen.dart
import 'package:flutter/material.dart';

class CheckoutScreen extends StatelessWidget {
  const CheckoutScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 0,
        title: const Text(
          'Checkout',
          style: TextStyle(color: Colors.black),
        ),
        leading: IconButton(
          icon: const Icon(Icons.arrow_back, color: Colors.black),
          onPressed: () => Navigator.pop(context),
        ),
      ),
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              const Text(
                'Offers',
                style: TextStyle(
                  fontSize: 16,
                  fontWeight: FontWeight.bold,
                ),
              ),,
              const SizedBox(height: 8),
              Row(
                mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                children: [
                  _buildPaymentOption('GPay', Icons.g_mobiledata),
                  _buildPaymentOption('FPay', Icons.payment),
                  _buildPaymentOption('Paytm', Icons.payment),
                  _buildPaymentOption('Card', Icons.credit_card),
                ],
              ),
              const SizedBox(height: 24),
              const Text(
                'Delivery Instructions',
                style: TextStyle(
                  fontSize: 16,
                  fontWeight: FontWeight.bold,
                ),
              ),
              const SizedBox(height: 8),
              Container(
                padding: const EdgeInsets.all(12),
                decoration: BoxDecoration(
                  border: Border.all(color: Colors.grey.shade300),
                  borderRadius: BorderRadius.circular(8),
                ),
                child: const Row(
                  children: [
                    Icon(Icons.location_on_outlined),
                    SizedBox(width: 8),
                    Text('Deliver to default location'),
```

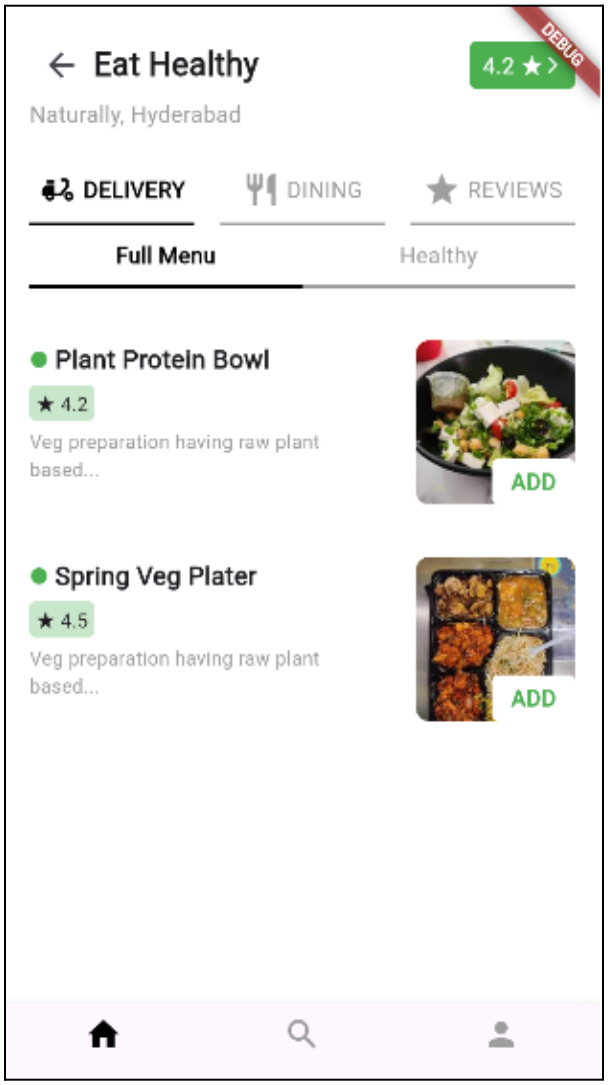
```
                Spacer(),
                Text(
                 'Change',
                 style: TextStyle(color:
Colors.red),
                ),
               ],
              ),
             ),
             const SizedBox(height: 16),
             Container(
              padding: const
EdgeInsets.all(12),
              decoration: BoxDecoration(
               color: Colors.grey.shade100,
               borderRadius:
BorderRadius.circular(8),
              ),
              child: Row(
               children: [
                const
Icon(Icons.eco_outlined, color:
Colors.green),
                const SizedBox(width: 8),
                Expanded(
                 child: Column(
                  crossAxisAlignment:
CrossAxisAlignment.start,
                  children: [
                   const Text(
                    'Climate conscious
delivery',
                    style:
TextStyle(fontWeight: FontWeight.bold),
                   ),
                   Text(
                    'Let us be more
sustainable by reducing plastic cutlery',
                    style: TextStyle(
                     color:
Colors.grey.shade600,
                     fontSize: 12,
                    ),
                   ),
                  ],
                 ),
                ),
               ],
              ),
             ),
             const SizedBox(height: 24),
             const Divider(),
             _buildBillItem('Item Total',
'₹199.00'),
             _buildBillItem('Delivery Fee',
'₹35.00'),
             _buildBillItem('Platform Fee',
'₹3.00'),
             const Divider(),
             _buildBillItem('Grand Total',
'₹234.00', isBold: true),
            ],
           ),
          ),
         ),
         bottomNavigationBar: Container(
          padding: const EdgeInsets.all(16),
          decoration: const BoxDecoration(
           color: Colors.white,
           boxShadow: [
            BoxShadow(
             color: Colors.black12,
             blurRadius: 4,
            ),
           ],
          ),
          child: Row(
           mainAxisAlignment:
MainAxisAlignment.spaceBetween,
           children: [
            const Column(
             mainAxisSize:
MainAxisSize.min,
             crossAxisAlignment:
```

```
    CrossAxisAlignment.start,
          children: [
            Text(
              '₹234.00',
              style: TextStyle(
                fontSize: 18,
                fontWeight: FontWeight.bold,
              ),
            ),
            Text(
              'View Detailed Bill',
              style: TextStyle(
                color: Colors.red,
                decoration:
TextDecoration.underline,
              ),
            ),
          ],
        ),
        ElevatedButton(
          onPressed: () {},
          style: ElevatedButton.styleFrom(
            backgroundColor: Colors.red,
            padding: const
EdgeInsets.symmetric(
              horizontal: 32,
              vertical: 12,
            ),
            shape:
RoundedRectangleBorder(
              borderRadius:
BorderRadius.circular(8),
            ),
          ),
          child: const Text(
            'Place Order',
            style: TextStyle(color:
Colors.white),
          ),
        ),
      ],
    ),
  ),
 );
}

 Widget _buildPaymentOption(String
title, IconData icon) {
  return Container(
    padding: const EdgeInsets.all(12),
    decoration: BoxDecoration(
      border: Border.all(color:
Colors.grey.shade300),
      borderRadius:
BorderRadius.circular(8),
    ),
    child: Column(
      children: [
        Icon(icon),
        const SizedBox(height: 4),
        Text(title),
      ],
    ),
  );
}

 Widget _buildBillItem(String title, String
amount, {bool isBold = false}) {
  return Padding(
    padding: const
EdgeInsets.symmetric(vertical: 8),
    child: Row(
      mainAxisAlignment:
MainAxisAlignment.spaceBetween,
      children: [
        Text(
          title,
          style: TextStyle(
            fontWeight: isBold ?
FontWeight.bold : FontWeight.normal,
          ),
        ),
        Text(
          Amount,
```

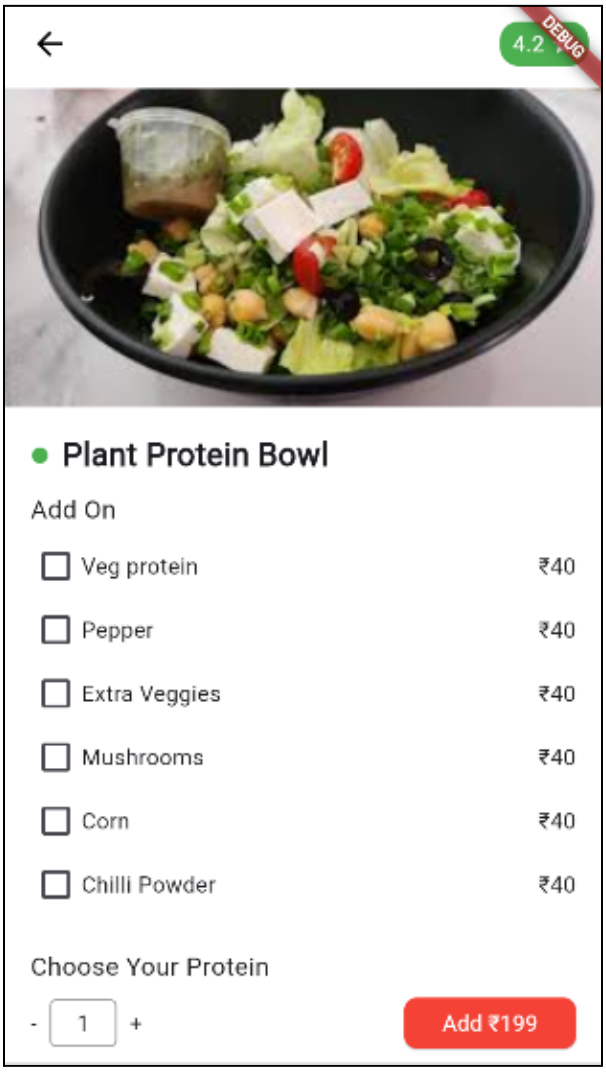
```
      ),
    );
  }

  Widget _buildPaymentOption(String
title, IconData icon) {
    return Container(
      padding: const EdgeInsets.all(12),
      decoration: BoxDecoration(
        border: Border.all(color:
Colors.grey.shade300),
        borderRadius:
BorderRadius.circular(8),
      ),
      child: Column(
        children: [
          Icon(icon),
          const SizedBox(height: 4),
          Text(title),
        ],
      ),
    );
  }

  Widget _buildBillItem(String title, String
amount, {bool isBold = false}) {
    return Padding(
      padding: const
EdgeInsets.symmetric(vertical: 8),
      child: Row(
        mainAxisAlignment:
MainAxisAlignment.spaceBetween,
        children: [
          Text(
            title,
            style: TextStyle(
              fontWeight: isBold ?
FontWeight.bold : FontWeight.normal,
            ),
          ),
          Text(
            amount,
            style: TextStyle(
              fontWeight: isBold ?
FontWeight.bold : FontWeight.normal,
            ),
          ),
        ],
      ),
    );
  }
}
```

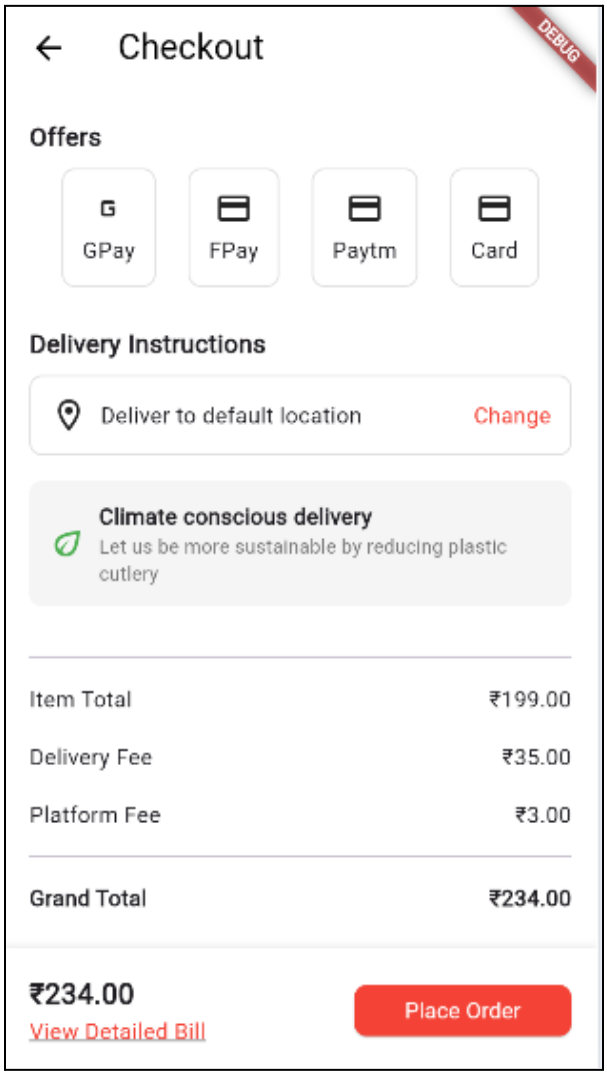
OUTPUT:


DEBUG
Restaurant name, cuisine, or a dish...
PRO
Cuisine
Rating
Popular
60% OFF
new something
billig
discounts
Eat what makes you happy
Healthy
Biryani
Pizza
Noodles
Chicken
Burger
Cake
More
Home
Search
Profile


DEBUG
Eat Healthy
4.2
Naturally, Hyderabad
DELIVERY
DINING
REVIEWS
Full Menu
Healthy
Plant Protein Bowl
4.2
Veg preparation having raw plant based...
ADD
Spring Veg Plater
4.5
Veg preparation having raw plant based...
ADD

Project Title: Zomato **Roll No. 26**





MAD & PWA Lab
Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

NAME:VAISHNAL DILIP MALI
CLASS:D15A
ROLL NO:26

AIM:To include fonts,icons and images in Flutter app.

Theory: -

Flutter is a versatile open-source UI framework , which allows developers to build natively
compiled applications for mobile, web, and desktop platforms from a single codebase. One of the key strengths of Flutter is its flexibility in creating highly customizable UIs. This practical focuses
on incorporating essential visual elements—icons, images, and custom fonts—into a Flutter application. These elements enhance the visual appeal and usability of the app, providing an engaging experience for users.
A flutter app when built has both assets (resources) and code. Assets are available and deployed
during runtime. The asset is a file that can include static data, configuration files, icons, and
images. The Flutter app supports many image formats, such as JPEG, WebP, PNG, GIF, animated
WebP/GIF, BMP, and WBMP.
Visual elements play a significant role in app development.
• **Enhanced User Experience**: Images and icons make your app visually appealing and user
friendly.
• **Information Conveyance**: They convey information quickly and intuitively. A well chosen icon can replace lengthy text.
• **Branding**: Custom icons and images reinforce your app's branding, making it memorable.

➢ **Adding Icons in Flutter**

Flutter provides built-in material design icons through the Icons class. Custom icons can also

be added using third-party packages such as flutter_launcher_icons and font_awesome_flutter.

```
Icon(
  Icons.home,
  size: 40,
);
```

➢ **Adding Images in Flutter**

Flutter supports images from three sources:

1. **Assets (Stored locally in the project)**

• Place the image inside the assets/images folder in the project.
• Declare the image in pubspec.yaml

```
flutter:
  assets:
    - assets/images/sample.png
```

• Display the image in the app

```
Image.asset('assets/images/sample.png');
```

2. **Network (Fetched from the internet)**

Displaying images from the internet or network is very simple. Flutter provides a built-in method Image.network to work with images from a URL. The Image.network method also
allows you to use some optional properties, such as height, width, color, fit, and many more.

```
Image.network('https://example.com/sample.jpg');
```

3. **Memory or File (Stored on the device)**

➢ Adding Custom Fonts in Flutter

By default, Flutter uses the Roboto font, but custom fonts can be added for a unique UI.

• Download the font and place it in the assets/fonts/ folder.
• Declare the font in pubspec.yaml
• Use the font in the app

```
Text(
  'Custom Font Example',
  style: TextStyle(fontFamily: 'CustomFont', fontSize: 24),
);
```

CODE:

Homepage.dart

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

// First, add this typography class at the
top of your file
class AppTypography {
  static final TextStyle displayLarge =
GoogleFonts.poppins(
    fontSize: 24,
    fontWeight: FontWeight.bold,
    letterSpacing: -0.3,
    color: Colors.white,
  );

  static final TextStyle titleMedium =
GoogleFonts.poppins(
    fontSize: 16,
    fontWeight: FontWeight.w600,
    letterSpacing: -0.2,
    color: Colors.white,
  );

  static final TextStyle bodySmall =
GoogleFonts.poppins(
    fontSize: 12,
class HomeScreen extends
StatelessWidget {
  const HomeScreen({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Stack(
        children: [
          Container(
            decoration: BoxDecoration(
              gradient: LinearGradient(
                begin: Alignment.topCenter,
                end: Alignment.bottomCenter,
                colors: [
                  Colors.black.withOpacity(0.8),
                  Colors.black,
                ],
              ),
            ),
          ),
          SingleChildScrollView(
            child: SafeArea(
              child: Column(
                crossAxisAlignment:
CrossAxisAlignment.start,
```

```
    fontWeight: FontWeight.normal,
    letterSpacing: 0.1,
    color: Colors.grey[400],
  );
}

          children: [
                    // Updated CircleAvatar
with specific size and styling
                    const CircleAvatar(
                      radius: 20,
                      backgroundColor:
Color(0xFF282828),
                      child: Icon(
                        Icons.person,

color: Colors.white,
                        size: 24,
                      ),
                    ),
                    const SizedBox(width: 16),
                    Container(
                      padding: const
EdgeInsets.symmetric(
                          horizontal: 16, vertical:
8),
                      decoration:
BoxDecoration(
                        color: const
Color(0xFF1DB954),
                        borderRadius:
BorderRadius.circular(20),
                      ),
                      child: const Text('All',
                          style: TextStyle(color:
Colors.white)),
                    ),
                    const SizedBox(width: 8),
                    Container(
                      padding: const
EdgeInsets.symmetric(
                          horizontal: 16, vertical:
                children: [
                  Padding(
                    padding: const
EdgeInsets.all(16.0),
                    child: Row(

  borderRadius: BorderRadius.circular(20),
                      ),
                      child: const Text('Music',
                          style: TextStyle(color:
Colors.white)),
                    ),
                    const SizedBox(width: 8),
                    Container(
                      padding: const
EdgeInsets.symmetric(
                          horizontal: 16, vertical:
8),
                      decoration:
BoxDecoration(
                        color: const
Color(0xFF282828),
                        borderRadius:
BorderRadius.circular(20),
                      ),
                      child: const
Text('Podcasts',
                          style: TextStyle(color:
Colors.white)),
                    ),
                  ],
                ),
              ),
              const PlaylistSection(
                title: 'On Repeat',
                image:
'assets/images/on_repeat.jpg',
                isLarge: true,
              ),
              const PlaylistSection(
                title: 'Trending Now India',
                image:
```

```
8),
                    decoration:
BoxDecoration(
                      color: const
Color(0xFF282828),

                ),
                image:
'assets/images/liked_songs.jpg',
                isLarge: true,
              ),
              const SectionTitle(title: 'Your
top mixes'),
              SingleChildScrollView(
                scrollDirection:
Axis.horizontal,
                padding: const
EdgeInsets.symmetric(horizontal: 16),
                child: Row(
                  children: const [
                    MixCard(
                      title: 'Arijit Singh Mix',
                      subtitle: 'Pritam,
Vishal-Shekhar and Atif Aslam',
                      image:
'assets/images/arijit.jpg',
                    ),
                    SizedBox(width: 16),
                    MixCard(
                      title: 'Chill Mix',
                      subtitle: 'Kendrick Lamar,
The Weeknd, Future',
                      image:
'assets/images/chill.jpg',
                    ),
                    SizedBox(width: 16),
                    MixCard(
                      title: 'R&B Mix',
                      subtitle: 'Kali Uchis,
SZA',
                      image:
'assets/images/rnb.jpg',
                    ),
'assets/images/trending_india.jpg',
                      isLarge: true,
                    ),
                    const PlaylistSection(
                      title: 'Liked Songs',
                    ),
                  ),
                ),
                const Positioned(
                  left: 0,
                  right: 0,
                  bottom: 0,
                  child: NowPlayingBar(),
                ),
              ],
            ),
          );
        }
      }

class PlaylistSection extends
StatelessWidget {
  final String title;
  final String image;
  final bool isLarge;

  const PlaylistSection({
    Key? key,
    required this.title,
    required this.image,
    this.isLarge = false,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.all(16.0),
      child: Row(
        children: [
          ClipRRect(
            borderRadius:
BorderRadius.circular(4),
```

```
                ],
              ),
            ),
            const SizedBox(height: 100),
          ],
            ),
        ),
        const SizedBox(width: 16),
        Expanded(
          child: Text(
            title,
            style:
AppTypography.titleMedium, // Updated
typography
            overflow: TextOverflow.ellipsis,
// Added overflow handling
            maxLines: 2,
          ),
        ),
        Material(
          color: Colors.transparent,
          child: IconButton(
            icon: const Icon(
              Icons.more_vert,
              color: Colors.white,
              size: 24,
            ),
            onPressed: () {},
          ),
        ),
      ],
    ),
  );
 }
}

class MixCard extends StatelessWidget {
  final String title;
  final String subtitle;
  final String image;

  const MixCard({
            child: Image.asset(
              image,
              width: isLarge ? 80 : 60,
              height: isLarge ? 80 : 60,
              fit: BoxFit.cover,
@override
  Widget build(BuildContext context) {
    return SizedBox(
      width: 160,
      child: Column(
        crossAxisAlignment:
CrossAxisAlignment.start,
        children: [
          ClipRRect(
            borderRadius:
BorderRadius.circular(4),
            child: Image.asset(
              image,
              width: 160,
              height: 160,
              fit: BoxFit.cover,
            ),
          ),
          const SizedBox(height: 8),
          Text(
            title,
            style:
AppTypography.titleMedium, // Updated
typography
            overflow: TextOverflow.ellipsis, //
Added overflow handling
            maxLines: 1,
          ),
          const SizedBox(height: 4),
          Text(
            subtitle,
            style: AppTypography.bodySmall,
// Updated typography
            overflow: TextOverflow.ellipsis, //
Added overflow handling
            maxLines: 2,
          ),
```

```
    Key? key,
    required this.title,
    required this.subtitle,
    required this.image,
  }) : super(key: key);

  }

class NowPlayingBar extends
StatelessWidget {
  const NowPlayingBar({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      color: const Color(0xFF282828),
      padding: const EdgeInsets.all(8),
      child: Row(
        children: [
          ClipRRect(
            borderRadius:
BorderRadius.circular(4),
            child: Image.asset(
              'assets/images/perfect_1d.jpg',
              width: 40,
              height: 40,
              fit: BoxFit.cover,
            ),
          ),
          const SizedBox(width: 12),
          Expanded(
            child: Column(
              crossAxisAlignment:
CrossAxisAlignment.start,
              mainAxisSize:
MainAxisSize.min,
              children: [
                Text(
                  'Perfect',
                  style:
AppTypography.titleMedium, // Updated
          ],
        ),
      );
    }

      'One Direction',
                style:
AppTypography.bodySmall, // Updated
typography
                overflow:
TextOverflow.ellipsis,
              ),
            ],
          ),
        ),
        Material(
          color: Colors.transparent,
          child: IconButton(
            icon: const Icon(
              Icons.devices_outlined,
              color: Colors.white,
              size: 24,
            ),
            onPressed: () {},
          ),
        ),
        Material(
          color: Colors.transparent,
          child: IconButton(
            icon: const Icon(
              Icons.favorite_border,
              color: Colors.white,
              size: 24,
            ),
            onPressed: () {},
          ),
        ),
        Material(
          color: Colors.transparent,
          child: IconButton(
            iconSize: 32,
            icon: const Icon(
```

```
typography
                overflow:
TextOverflow.ellipsis,
              ),
              Text(
                  ),
        ],
      ),
    );
  }
}

class SectionTitle extends StatelessWidget
{
  final String title;

  const SectionTitle({Key? key, required
this.title}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.all(16.0),
      child: Text(
        title,
        style: AppTypography.displayLarge,
// Updated typography
      ),
    );
  }
}
```

```
              Icons.play_circle_filled,
              color: Colors.white,
            ),
            onPressed: () {},
          ),
```

Register.dart

```
import 'package:flutter/material.dart';
import
'package:spotify_clone/features/auth/scree
ns/sign_in_screen.dart';
import 'package:spotify_clone/main.dart';

class Register extends StatefulWidget {
  const Register({Key? key}) : super(key:
key);

  @override
  State<Register> createState() =>
_RegisterState();
}

class _RegisterState extends
State<Register> {
  bool _obscurePassword = true;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: const
Color(0xFF1C1B1B),
      body: SafeArea(
        child: SingleChildScrollView(
          child: Padding(
            padding: const
EdgeInsets.all(24.0),
            child: Column(
              crossAxisAlignment:
CrossAxisAlignment.start,
              children: [
                Row(
                  children: [
```

```
Navigator.pop(context),
                ),
                const SizedBox(width: 10),
                Image.asset(
'assets/images/spotify_logo.png',
                  height: 150,
                ),
              ],
            ),
            const SizedBox(height: 32),
            const Text(
              'Register',
              style: TextStyle(
                color: Colors.white,
                fontSize: 28,
                fontWeight: FontWeight.bold,
              ),
            ),
            const SizedBox(height: 8),
            Row(
              children: [
                const Text(
                  'If You Need Any Support ',
                  style: TextStyle(color:
Colors.grey),
                ),
                TextButton(
                  onPressed: () {},
                  child: const Text(
                    'Click Here',
                    style: TextStyle(color:
Colors.green),
                  ),
                ),
              ],
            ),
                  IconButton(
                    icon: const
Icon(Icons.arrow_back, color:
Colors.white),
                    onPressed: () =>
hintText: 'Full Name',
                  hintStyle: const
TextStyle(color: Colors.grey),
                  filled: true,
                  fillColor:
Colors.grey.withOpacity(0.1),
                  border: OutlineInputBorder(
                    borderRadius:
BorderRadius.circular(12),
                    borderSide:
BorderSide.none,
                  ),
                ),
              ),
              const SizedBox(height: 16),
              TextField(
                style: const TextStyle(color:
Colors.white),
                decoration: InputDecoration(
                  hintText: 'Enter Email',
                  hintStyle: const
TextStyle(color: Colors.grey),
                  filled: true,
                  fillColor:
Colors.grey.withOpacity(0.1),
                  border: OutlineInputBorder(
                    borderRadius:
BorderRadius.circular(12),
                    borderSide:
BorderSide.none,
                  ),
                ),
              ),
              const SizedBox(height: 16),
              TextField(
                obscureText:
_obscurePassword,
```

```
            const SizedBox(height: 32),
            TextField(
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
TextStyle(color: Colors.grey),
                filled: true,
                fillColor:
Colors.grey.withOpacity(0.1),
                border: OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
                suffixIcon: IconButton(
                  icon: Icon(
                    _obscurePassword
                        ? Icons.visibility_off
                        : Icons.visibility,
                    color: Colors.grey,
                  ),
                  onPressed: () {
                    setState(() {
                      _obscurePassword =
!_obscurePassword;
                    });
                  },
                ),
              ),
            ),
            const SizedBox(height: 32),
            SizedBox(
              width: double.infinity,
              height: 56,
              child: ElevatedButton(
                onPressed: () {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                      builder: (context) =>
const MainScreen()),
                style: const TextStyle(color:
Colors.white),
                decoration: InputDecoration(
                  hintText: 'Password',
                  hintStyle: const
Color(0xFF42C83C),
                  shape:
RoundedRectangleBorder(
                    borderRadius:
BorderRadius.circular(30),
                  ),
                ),
                child: const Text(
                  'Create Account',
                  style: TextStyle(
                    fontSize: 16,
                    fontWeight:
FontWeight.bold,
                  ),
                ),
              ),
            ),
            const SizedBox(height: 32),
            Row(
              mainAxisAlignment:
MainAxisAlignment.center,
              children: [

Image.asset('assets/images/google_logo.pn
g', height: 40),
                const SizedBox(width: 20),

Image.asset('assets/images/apple_logo.png
', height: 40),
              ],
            ),
            const SizedBox(height: 32),
            Row(
              mainAxisAlignment:
MainAxisAlignment.center,
              children: [
                const Text(
```

```
                  );
                },
                style:
ElevatedButton.styleFrom(
                  backgroundColor: const
TextButton(
                  onPressed: () {
                    Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) =>
const SignInScreen()),
                    );
                  },
                  child: const Text(
                    'Sign In',
                    style: TextStyle(color:
Colors.white),
                  ),
                ),
              ],
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

```
                  'Do You Have An Account?
',
                  style: TextStyle(color:
Colors.grey),
                ),
```

Sign_in.dart

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

import 'register.dart';

class SignInScreen extends StatefulWidget
{
 const SignInScreen({Key? key}) :
super(key: key);

 @override
 State<SignInScreen> createState() =>
_SignInScreenState();
}

class _SignInScreenState extends
State<SignInScreen> {
 bool _obscurePassword = true;

 @override
 Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: const
Color(0xFF1C1B1B),
    body: SafeArea(
      child: SingleChildScrollView(
        child: Padding(
          padding: const
EdgeInsets.all(24.0),
          child: Column(
            crossAxisAlignment:
CrossAxisAlignment.start,
            children: [
              Row(
                children: [
```

```
Navigator.pop(context),
                ),
                const SizedBox(width: 12),
                Image.asset(

'assets/images/spotify_logo.png',
                  height: 150,
                ),
              ],
            ),
            const SizedBox(height: 32),
            const Text(
              'Sign In',
              style: TextStyle(
                color: Colors.white,
                fontSize: 28,
                fontWeight: FontWeight.bold,
              ),
            ),
            const SizedBox(height: 8),
            Row(
              children: [
                const Text(
                  'If You Need Any Support ',
                  style: TextStyle(color:
Colors.grey),
                ),
                TextButton(
                  onPressed: () {},
                  child: const Text(
                    'Click Here',
                    style: TextStyle(color:
Colors.green),
                  ),
                ),
              ],
            ),
                  IconButton(
                    icon: const
Icon(Icons.arrow_back, color:
Colors.white),
                    onPressed: () =>
                  hintText: 'Enter Username Or
Email',
                  hintStyle: const
TextStyle(color: Colors.grey),
                  filled: true,
                  fillColor:
Colors.grey.withOpacity(0.1),
                  border: OutlineInputBorder(
                    borderRadius:
BorderRadius.circular(12),
                    borderSide:
BorderSide.none,
                  ),
                ),
              ),
              const SizedBox(height: 16),
              TextField(
                obscureText:
_obscurePassword,
                style: const TextStyle(color:
Colors.white),
                decoration: InputDecoration(
                  hintText: 'Password',
                  hintStyle: const
TextStyle(color: Colors.grey),
                  filled: true,
                  fillColor:
Colors.grey.withOpacity(0.1),
                  border: OutlineInputBorder(
                    borderRadius:
BorderRadius.circular(12),
                    borderSide:
BorderSide.none,
                  ),
                  suffixIcon: IconButton(
                    icon: Icon(
                      _obscurePassword
```

```
            const SizedBox(height: 32),
            TextField(
              style: const TextStyle(color:
Colors.white),
              decoration: InputDecoration(
 setState(() {
                    _obscurePassword =
!_obscurePassword;
                  });
                },
              ),
            ),
            ),
            const SizedBox(height: 16),
            Align(
              alignment:
Alignment.centerRight,
              child: TextButton(
                onPressed: () {},
                child: const Text(
                  'Recovery Password',
                  style: TextStyle(color:
Colors.grey),
                ),
              ),
            ),
            const SizedBox(height: 32),
            SizedBox(
              width: double.infinity,
              height: 56,
              child: ElevatedButton(
                onPressed: () {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                      builder: (context) =>
const MainScreen()),
                  );
                },
                style:
ElevatedButton.styleFrom(
                  backgroundColor: const
                    ? Icons.visibility_off
                    : Icons.visibility,
                  color: Colors.grey,
                ),
                onPressed: () {
                          ),
              ),
              child: const Text(
                'Sign In',
                style: TextStyle(
                  fontSize: 16,
                  fontWeight:
FontWeight.bold,
                ),
              ),
            ),
          ),
          const SizedBox(height: 16),
          const Center(
            child: Text(
              'Or',
              style: TextStyle(color:
Colors.grey),
            ),
          ),
          const SizedBox(height: 16),
          Row(
            mainAxisAlignment:
MainAxisAlignment.center,
            children: [

Image.asset('assets/images/google_logo.pn
g', height: 40),
              const SizedBox(width: 20),

Image.asset('assets/images/apple_logo.png
', height: 40),
            ],
          ),
          const SizedBox(height: 32),
          Row(
            mainAxisAlignment:
```

```
Color(0xFF42C83C),
                  shape:
RoundedRectangleBorder(
                    borderRadius:
BorderRadius.circular(30),
Colors.grey),
                ),
                TextButton(
                  onPressed: () {
                    Navigator.push(
                      context,
                      MaterialPageRoute(
                        builder: (context) =>
const Register()),
                      );
                  },
                  child: const Text(
                    'Register Now',
                    style: TextStyle(color:
Colors.blue),
                  ),
                ),
              ],
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

MainAxisAlignment.center,
children: [
const Text(
'Not A Member? ',
style: TextStyle(color:

Profile_screen.dart

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

class ProfileScreen extends
StatelessWidget {
 const ProfileScreen({Key? key}) :
super(key: key);

 @override
 Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black,
    body: Column(
      children: [
       const Expanded(flex: 2, child:
_TopPortion()),
        Expanded(
          flex: 3,
          child: Padding(
            padding: const
EdgeInsets.all(16.0),
            child: Column(
              crossAxisAlignment:
CrossAxisAlignment.start,
              children: [
                Text(
                  "Aryan",
                  style:
Theme.of(context).textTheme.headlineSm
all?.copyWith(
                      color: Colors.white,
                      fontWeight:
FontWeight.bold,
                    ),
                ),
                const SizedBox(height: 8),
                const _ProfileInfoRow(),
```

```
copyWith(
                    color: Colors.white,
                    fontWeight:
FontWeight.bold,
                  ),
                ),
                const SizedBox(height: 16),
                const _PlaylistCard(
                  title: "My playlist #3",
                  saves: "0 saves",
                  imageUrl
              ),
            ),
          ),
          //const _BottomNavBar(),
        ],
      ),
    );
  }
}

class _ProfileInfoRow extends
StatelessWidget {
  const _ProfileInfoRow({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const
EdgeInsets.symmetric(vertical: 8.0),
      child: Row(
        children: [
          Text(
            "1 follower",
            style:
Theme.of(context).textTheme.bodyMediu
m?.copyWith(
                color: Colors.grey,
              ),
          ),
          const Text(" • ", style:
                const SizedBox(height: 24),
                Text(
                  "Playlists",
                  style:      ),
    );
  }
}

class _PlaylistCard extends
StatelessWidget {
  final String title;
  final String saves;
  final String? imageUrl; // New optional
parameter

  const _PlaylistCard({
    Key? key,
    required this.title,
    required this.saves,
    this.imageUrl, // Make it optional
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      padding: const EdgeInsets.all(12),
      decoration: BoxDecoration(
        color: Colors.grey[900],
        borderRadius:
BorderRadius.circular(8),
      ),
      child: Row(
        children: [
          Container(
            width: 60,
            height: 60,
            decoration: BoxDecoration(
              color: Colors.grey[800],
              borderRadius:
BorderRadius.circular(4),
            ),
            child: imageUrl != null
```

```
TextStyle(color: Colors.grey)),
          Text(
            "19 following",
            style:
Theme.of(context).textTheme.bodyMediu
m?.copyWith(
                  color: Colors.grey,
                ),
          ),
 : const Icon(Icons.music_note, color:
Colors.white),
          ),
          const SizedBox(width: 12),
          Column(
            crossAxisAlignment:
CrossAxisAlignment.start,
            children: [
              Text(
                title,
                style: const TextStyle(
                  color: Colors.white,
                  fontWeight: FontWeight.bold,
                ),
              ),
              const SizedBox(height: 4),
              Text(
                saves,
                style: TextStyle(color:
Colors.grey[400]),
              ),
            ],
          ),
        ],
      ),
    );
  }
}

class _TopPortion extends StatelessWidget
{
  const _TopPortion({Key? key}) :
super(key: key);
          ? Image.network(imageUrl!, fit:
BoxFit.cover)
                    end:
Alignment.bottomCenter,
            colors: [Color(0xFF9370DB),
Colors.black],
          ),
        ),
      ),
      SafeArea(
        child: Padding(
          padding: const
EdgeInsets.all(16.0),
          child: Column(
            children: [
              Row(
                children: [
                  IconButton(
                    icon: const
Icon(Icons.arrow_back, color:
Colors.white),
                    onPressed: () {
                      Navigator.pop(context);
                    },
                  ),
                  const Spacer(),
                  IconButton(
                    icon: const
Icon(Icons.more_vert, color:
Colors.white),
                    onPressed: () {},
                  ),
                ],
              ),
              const SizedBox(height: 20),
              Container(
                width: 100,
                height: 100,
                decoration: BoxDecoration(
                  color: Colors.purple[200],
                  shape: BoxShape.circle,
                ),
```

```
  @override
  Widget build(BuildContext context) {
   return Stack(
    children: [
     Container(
      decoration: const BoxDecoration(
       gradient: LinearGradient(
        begin: Alignment.topCenter,
 "A",
                style:
Theme.of(context).textTheme.displayLarg
e?.copyWith(
                    color: Colors.black,
                    fontWeight:
FontWeight.bold,
                   ),
               ),
              ),
             ),
            ],
           ),
          ),
         ),
        ],
       );
  }
}
```

```
              child: Center(
               child: Text(
```

Sign_in_screen.dart

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

import 'register.dart';

class SignInScreen extends StatefulWidget
{
 const SignInScreen({Key? key}) :
super(key: key);

 @override
 State<SignInScreen> createState() =>
_SignInScreenState();
}

class _SignInScreenState extends
State<SignInScreen> {
 bool _obscurePassword = true;

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   backgroundColor: const
Color(0xFF1C1B1B),
    body: SafeArea(
     child: SingleChildScrollView(
      child: Padding(
       padding: const
EdgeInsets.all(24.0),
        child: Column(
         crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
           Row(
            children: [
             IconButton(
              icon: const
Icon(Icons.arrow_back, color:
Colors.white),
```

```
),
                const SizedBox(width: 12),
                Image.asset(
'assets/images/spotify_logo.png',
                  height: 150,
                ),
              ],
            ),
            const SizedBox(height: 32),
            const Text(
              'Sign In',
              style: TextStyle(
                color: Colors.white,
                fontSize: 28,
                fontWeight: FontWeight.bold,
              ),
            ),
            const SizedBox(height: 8),
            Row(
              children: [
                const Text(
                  'If You Need Any Support ',
                  style: TextStyle(color:
Colors.grey),
                ),
                TextButton(
                  onPressed: () {},
                  child: const Text(
                    'Click Here',
                    style: TextStyle(color:
Colors.green),
                  ),
                  onPressed: () =>
Navigator.pop(context),
                Email',
                hintStyle: const
TextStyle(color: Colors.grey),
                filled: true,
                fillColor:
Colors.grey.withOpacity(0.1),
                border: OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
              ),
            ),
            const SizedBox(height: 16),
            TextField(
              obscureText:
_obscurePassword,
                style: const TextStyle(color:
Colors.white),
                decoration: InputDecoration(
                  hintText: 'Password',
                  hintStyle: const
TextStyle(color: Colors.grey),
                filled: true,
                fillColor:
Colors.grey.withOpacity(0.1),
                border: OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
                suffixIcon: IconButton(
                  icon: Icon(
                    _obscurePassword
                        ? Icons.visibility_off
                        : Icons.visibility,
                  color: Colors.grey,
                ),
```

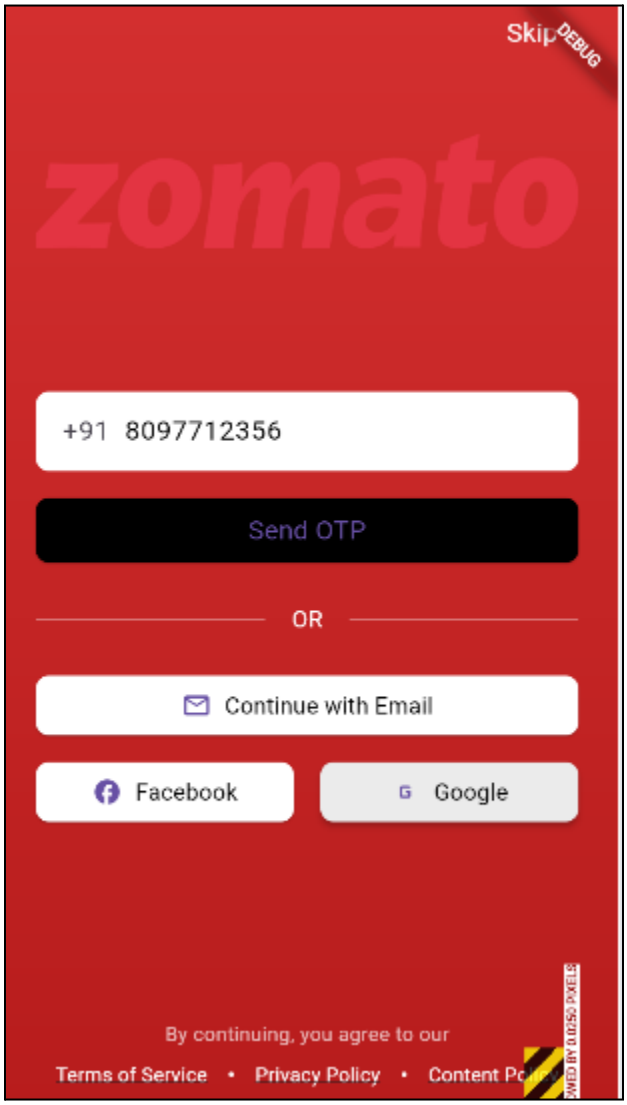
```
              ),
             ],
            ),
            const SizedBox(height: 32),
            TextField(
             style: const TextStyle(color:
Colors.white),
             decoration: InputDecoration(
              hintText: 'Enter Username Or
_obscurePassword = !_obscurePassword;
                });
              },
             ),
            ),
           ),
           const SizedBox(height: 16),
           Align(
            alignment:
Alignment.centerRight,
            child: TextButton(
             onPressed: () {},
             child: const Text(
              'Recovery Password',
              style: TextStyle(color:
Colors.grey),
             ),
            ),
           ),
           const SizedBox(height: 32),
           SizedBox(
            width: double.infinity,
            height: 56,
            child: ElevatedButton(
             onPressed: () {
              Navigator.push(
               context,
               MaterialPageRoute(
                  builder: (context) =>
const MainScreen()),
              );
             },
             style:
                onPressed: () {
                 setState(() {
                              ),
               child: const Text(
                'Sign In',
                style: TextStyle(
                 fontSize: 16,
                 fontWeight:
FontWeight.bold,
                ),
               ),
              ),
             ),
            ),
            const SizedBox(height: 16),
            const Center(
             child: Text(
              'Or',
              style: TextStyle(color:
Colors.grey),
             ),
            ),
            const SizedBox(height: 16),
            Row(
             mainAxisAlignment:
MainAxisAlignment.center,
             children: [

Image.asset('assets/images/google_logo.pn
g', height: 40),
              const SizedBox(width: 20),

Image.asset('assets/images/apple_logo.png
', height: 40),
             ],
            ),
            const SizedBox(height: 32),
            Row(
             mainAxisAlignment:
MainAxisAlignment.center,
             children: [
              const Text(
               'Not A Member? ',
```

```
ElevatedButton.styleFrom(
                backgroundColor: const
Color(0xFF42C83C),
                shape:
RoundedRectangleBorder(
                  borderRadius:
BorderRadius.circular(30),
                ),

  ),
              TextButton(
                onPressed: () {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                        builder: (context) =>
const Register()),
                  );
                },
                child: const Text(
                  'Register Now',
                  style: TextStyle(color:
Colors.blue),
                ),
              ),
            ],
          ),
        ],
      ),
    ),
  ),
 );
 }
}
```

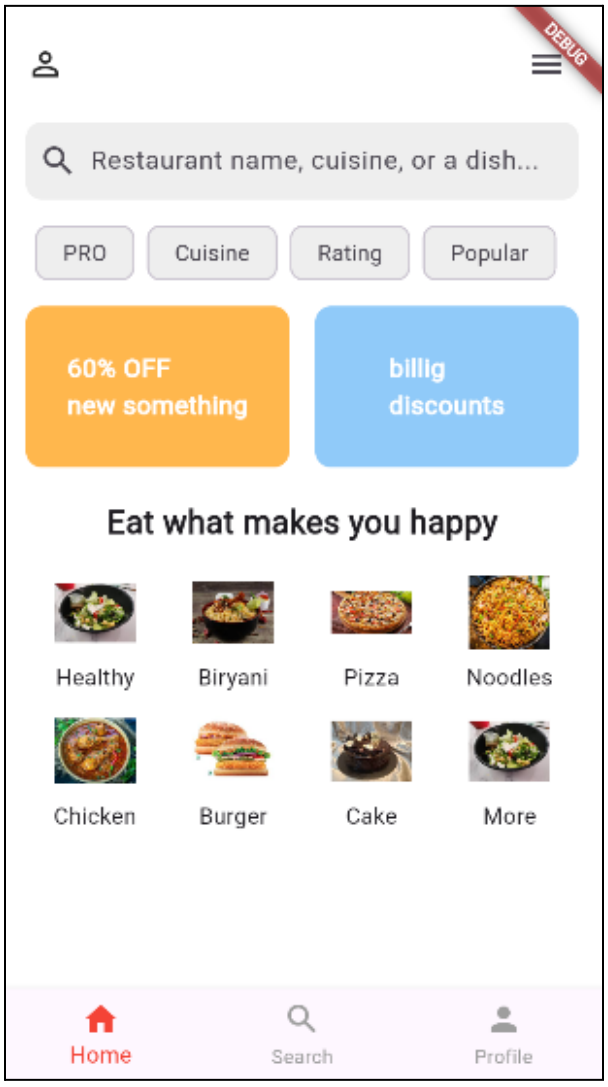
```
                  style: TextStyle(color:
Colors.grey),
```

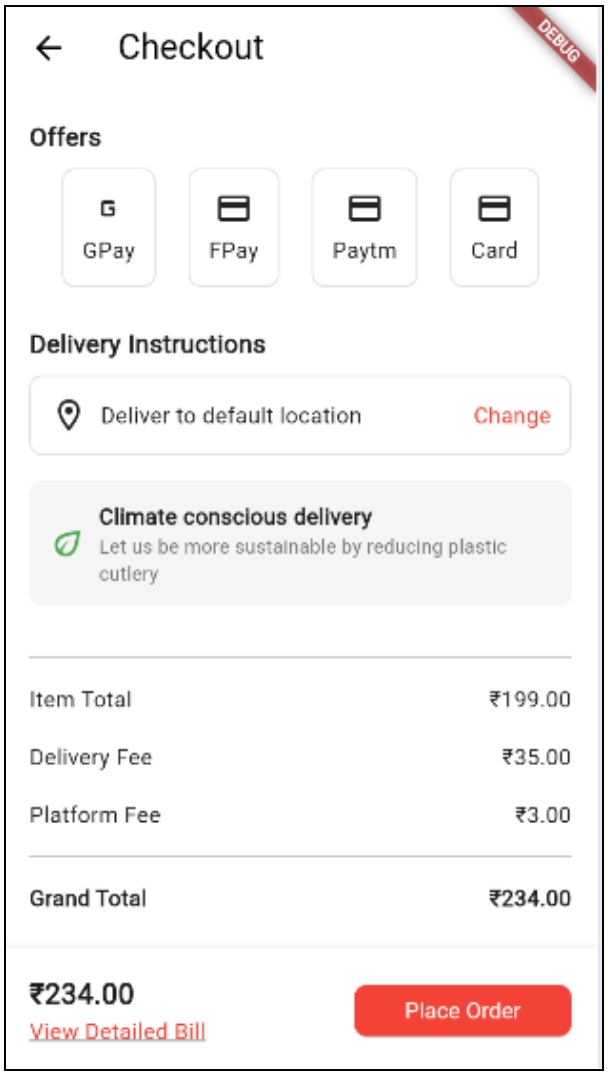
OUTPUT:

Register.dart



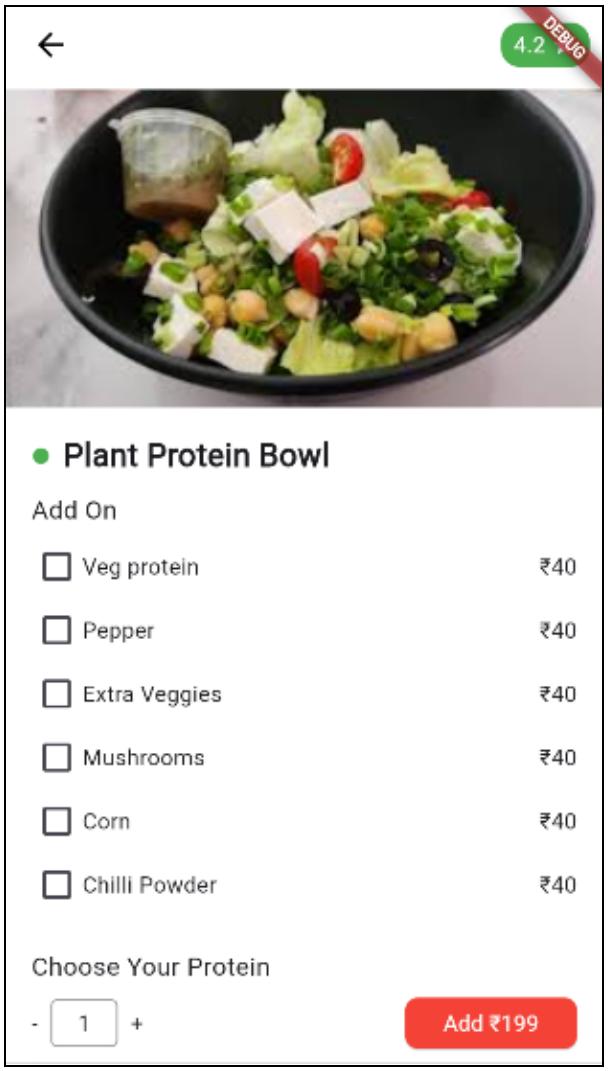
Home_page.dart





Search_page.dart

Project Title: Zomato **Roll No. 26**



MAD & PWA Lab
Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

Name:Vaishnal Mali
Div:D15A
Roll No.:26
MPL Experiment 4

AIM : To create an interactive Form using form widget

THEORY :

1. Introduction

Interactive forms are widely used in applications to collect user input in a structured way. These forms enhance user experience by providing a smooth and responsive interface. One common technique to ensure security in forms is OTP (One-Time Password) verification. OTP verification is commonly used for authentication and user identity confirmation.

2. Objective

To design an interactive form using form widgets.
To implement OTP verification to authenticate users.
To validate user inputs dynamically.

3. Form Widgets in UI Development

Form widgets are essential UI elements used to collect input from users. Some commonly used form widgets include:

TextField/Input Field – To enter text or numbers.
Dropdowns – To select options.
CheckBoxes and Radio Buttons – For multiple-choice selections.
Buttons – To submit the form.
Snackbar/Dialog – To show messages to users.

4. OTP Verification Technique

OTP verification is a security mechanism used in various applications for authentication. It typically involves the following steps:

User Enters Phone Number/Email
The user provides their phone number or email to receive an OTP.
OTP Generation
The server generates a random OTP and sends it via SMS or email.

User Enters OTP
The user inputs the received OTP into the form.
OTP Validation
The entered OTP is compared with the generated OTP.
If it matches, the user is authenticated.

CODE :
Login-screen

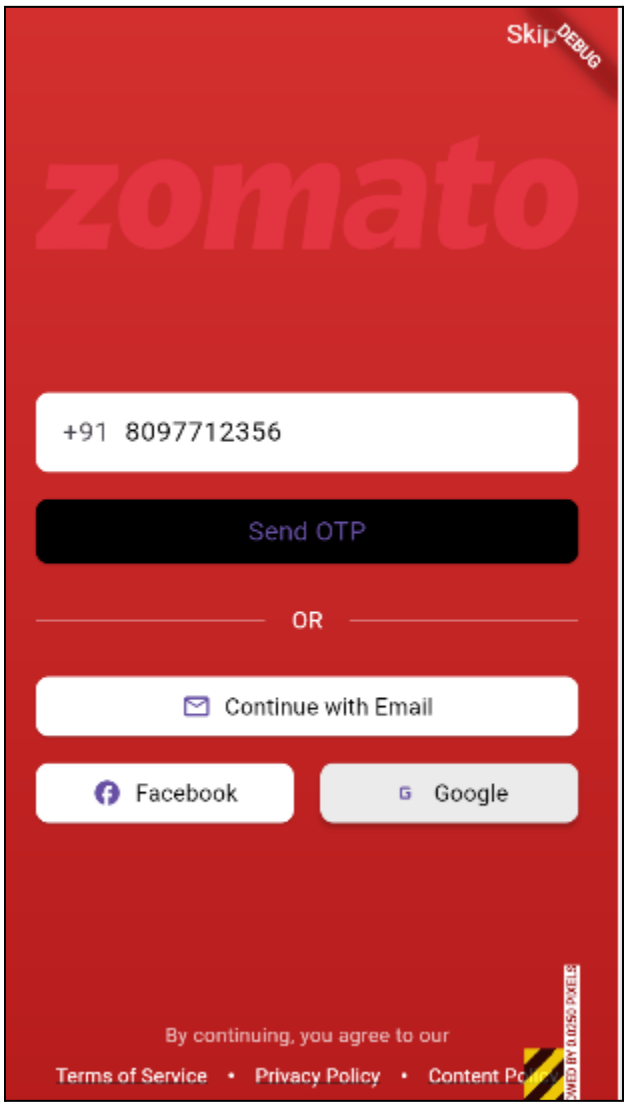
```
import 'package:flutter/material.dart';

class LoginScreen extends
StatelessWidget {
  const LoginScreen({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
            colors: [
              Colors.red.shade700,
              Colors.red.shade900,
            ],
          ),
        ),
        child: SafeArea(
          child: Padding(
            padding: const
EdgeInsets.symmetric(horizontal: 24.0),
            child: Column(
              children: [
                // Top bar with Skip button
                Row(
                  mainAxisAlignment:
MainAxisAlignment.end,
                  children: [
                    TextButton(
                      onPressed: () {},
                      child: const Text(
                        'Skip',
                        style: TextStyle(
                          color: Colors.white,
                          fontSize: 16,
                        ),
                      ),
                    ),
                  ],
                ),

                // Zomato Logo
                const SizedBox(height: 40),
                Image.asset(

'assests/Zomato_Logo.svg.png',  // Make
sure this path matches your asset location
                  height: 80,  // Adjust the
height as needed
                ),
                const SizedBox(height: 80),

                // Phone Input
                TextField(
                  keyboardType:
TextInputType.phone,
                  style: const TextStyle(color:
Colors.black),
```

```
              decoration: InputDecoration(
                filled: true,
                fillColor: Colors.white,
                prefixText: '+91  ',
                hintText: 'Phone number',
                border: OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(8),
                  borderSide:
BorderSide.none,
                ),
                contentPadding: const
EdgeInsets.symmetric(
                  horizontal: 16,
                  vertical: 16,
                ),
              ),
            ),

            const SizedBox(height: 16),

            // Send OTP Button
            SizedBox(
              width: double.infinity,
              child: ElevatedButton(
                onPressed: () {},
                style:
ElevatedButton.styleFrom(
                  backgroundColor:
Colors.black,
                  padding: const
EdgeInsets.symmetric(vertical: 16),
                  shape:
RoundedRectangleBorder(
                    borderRadius:
BorderRadius.circular(8),
                  ),
                ),
                child: const Text(
                  'Send OTP',
                  style: TextStyle(fontSize:
16),
                ),
              ),
            ),

            // OR divider
            const Padding(
              padding:
EdgeInsets.symmetric(vertical: 24.0),
              child: Row(
                children: [
                  Expanded(
                    child: Divider(color:
Colors.white38, thickness: 1),
                  ),
                  Padding(
                    padding:
EdgeInsets.symmetric(horizontal: 16.0),
                    child: Text(
                      'OR',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 14,
                      ),
                    ),
                  ),
                  Expanded(
                    child: Divider(color:
Colors.white38, thickness: 1),
                  ),
                ],
              ),
            ),

            // Email Button
            SizedBox(
              width: double.infinity,
              child: ElevatedButton.icon(
                onPressed: () {},
                icon: const
Icon(Icons.email_outlined),
                label: const Text('Continue
with Email'),
```

```
                style:
ElevatedButton.styleFrom(
                  backgroundColor:
Colors.white,
                  foregroundColor:
Colors.black,
                  padding: const
EdgeInsets.symmetric(vertical: 16),
                  shape:
RoundedRectangleBorder(
                    borderRadius:
BorderRadius.circular(8),
                  ),
                ),
              ),
            ),

            const SizedBox(height: 16),

            // Social Buttons
            Row(
              children: [
                Expanded(
                  child: ElevatedButton.icon(
                    onPressed: () {},
                    icon: const
Icon(Icons.facebook),
                    label: const
Text('Facebook'),
                    style:
ElevatedButton.styleFrom(
                      backgroundColor:
Colors.white,
                      foregroundColor:
Colors.black,
                      padding: const
EdgeInsets.symmetric(vertical: 16),
                      shape:
RoundedRectangleBorder(
                        borderRadius:
BorderRadius.circular(8),
                      ),
                    ),
                  ),
                ),
                const SizedBox(width: 16),
                Expanded(
                  child: ElevatedButton.icon(
                    onPressed: () {},
                    icon: const
Icon(Icons.g_mobiledata),
                    label: const Text('Google'),
                    style:
ElevatedButton.styleFrom(
                      backgroundColor:
Colors.white,
                      foregroundColor:
Colors.black,
                      padding: const
EdgeInsets.symmetric(vertical: 16),
                      shape:
RoundedRectangleBorder(
                        borderRadius:
BorderRadius.circular(8),
                      ),
                    ),
                  ),
                ),
              ],
            ),

            // Terms text
            const Spacer(),
            const Text(
              'By continuing, you agree to
our',
              style: TextStyle(
                color: Colors.white70,
                fontSize: 12,
              ),
            ),
            Row(
              mainAxisAlignment:
MainAxisAlignment.center,
```

```
                children: [
                  TextButton(
                    onPressed: () {},
                    child: const Text(
                      'Terms of Service',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 12,
                        decoration:
TextDecoration.underline,
                      ),
                    ),
                  ),
                  const Text(
                    '•',
                    style: TextStyle(color:
Colors.white),
                  ),
                  TextButton(
                    onPressed: () {},
                    child: const Text(
                      'Privacy Policy',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 12,
                        decoration:
TextDecoration.underline,
                      ),
                    ),
                  ),
                  const Text(
                    '•',
                    style: TextStyle(color:
Colors.white),
                  ),
                  TextButton(
                    onPressed: () {},
                    child: const Text(
                      'Content Policy',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 12,
                        decoration:
TextDecoration.underline,
                      ),
                    ),
                  ),
                ],
              ),
              const SizedBox(height: 8),
            ],
          ),
        ),
      ),
    );
  }
}
```

OUTPUT :


Skip
DEBUG
zomato
+91 8097712356
Send OTP
OR
Continue with Email
Facebook
Google
By continuing, you agree to our
Terms of Service • Privacy Policy • Content Po

MAD & PWA Lab
Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

NAME:VAISHNAL DILIP MALI
CLASS:D15A
ROLL NO.:26

AIM:To apply navigation, routing and gestures in Flutter app.

Theory:

In Flutter, the screens and pages are known as routes, and these routes are just a widget. In Android, a route is similar to an Activity.
In any mobile app, navigating to different pages defines the workflow of the application, and the way to handle the navigation is known as routing. Flutter provides a basic routing class MaterialPageRoute and two methods Navigator.push() and Navigator.pop() that shows how to navigate between two routes. The following steps are required to start navigation in your application.
Gestures enable the app to respond to user interactions, making the application more dynamic and responsive.

➢ Navigation and Routing in Flutter

Navigation is the process of moving between different screens or pages in an app. Flutter provides a simple and effective way to handle this through the use of the Navigator widget and
routes.

1. Using Navigator Widget

The Navigator widget manages a stack of routes, allowing for pushing and popping routes on
the stack.
• **Pushing a Route**: To navigate to a new screen, use Navigator.push().
• **Popping a Route**: To go back to the previous screen, use Navigator.pop().

```
ElevatedButton(
  onPressed: () {
```

```
   Navigator.push(
                                                    Aryan Patankar – D15A / 33
     context,
     MaterialPageRoute(builder: (context) => SecondScreen()),
   );},
);
```

2. **Named Routes**
Flutter also allows the use of named routes to navigate, which can make the routing process
cleaner, especially in larger applications.

```
MaterialApp(
  initialRoute: '/',
  routes: {
    '/': (context) => HomeScreen(),
    '/second': (context) => SecondScreen(),
  },
);
```

Navigate to the route using Navigator.pushNamed()

```
Navigator.pushNamed(context, '/second');
```

3. Handling Gestures in Flutter
Gestures refer to user interactions with the app, such as taps, swipes, pinches, and drags. Flutter provides several widgets and gesture detectors to handle these interactions.

4.Tap Gestures
The most common gesture is the tap, which can be handled using the GestureDetector widget or specific buttons like InkWell or ElevatedButton.

5.Long Press Gesture
For long press gestures, Flutter provides the onLongPress callback in GestureDetector or InkWell.

6.Swipe and Drag Gestures
Flutter also provides swipe and drag gesture handling. The onHorizontalDragUpdate and onVerticalDragUpdate callbacks are used for dragging gestures.

Main.dart

```
import 'package:flutter/material.dart';
import
'features/auth/screens/home_screen.dart';
import
'features/auth/screens/search_screen.dart';
import
'features/auth/screens/profile_page.dart';
import
'features/auth/screens/music_screen.dart';
import
'features/auth/screens/sign_in_screen.dart';
import
'features/auth/screens/loading_screen.dart';
import
'features/auth/screens/register_screen.dart';
import
'package:flutter_riverpod/flutter_riverpod.
dart';
import 'config/theme.dart';


void main() {
 runApp(
   const ProviderScope(
     child: SpotifyClone(),
   ),
 );
}
class SpotifyClone extends
StatelessWidget {
 const SpotifyClone({Key? key}) :
super(key: key);
// Keep instances of screens to maintain
their state
 final List<Widget> _screens = const [
   HomeScreen(),
   SearchScreen(),
   MusicScreen(),
   ProfileScreen(),
 ];
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner:
false,
    title: 'Spotify Clone',
    theme: AppTheme.lightTheme,
    home: const LoadingScreen(), //
Changed initialRoute to home
    routes: {
     '/signin': (context) => const
SignInScreen(),
     '/main': (context) => const
MainScreen(),
    },
   );
 }
}

class MainScreen extends StatefulWidget
{
 const MainScreen({Key? key}) :
super(key: key);

 @override
 State<MainScreen> createState() =>
_MainScreenState();
}

class _MainScreenState extends
State<MainScreen> {
 int _selectedIndex = 0;
type: BottomNavigationBarType.fixed,
        backgroundColor: const
Color(0xFF282828),
        selectedItemColor: Colors.white,
        unselectedItemColor: Colors.grey,
        selectedIconTheme: const
IconThemeData(
          size: 28,
```

```
void _onItemTapped(int index) {
   setState(() {
     _selectedIndex = index;
   });
  }

  @override
  Widget build(BuildContext context) {
   return WillPopScope(
     // Handle back button press
     onWillPop: () async {
       if (_selectedIndex != 0) {
         setState(() {
           _selectedIndex = 0;
         });
         return false;
       }
       return true;
     },
     child: Scaffold(
       // Use IndexedStack to preserve state
of all screens
       body: IndexedStack(
         index: _selectedIndex,
         children: _screens,
       ),
       bottomNavigationBar: Theme(
         data: Theme.of(context).copyWith(
           canvasColor: Colors.black,
         ),
         child: BottomNavigationBar(
           currentIndex: _selectedIndex,
           onTap: _onItemTapped,

BottomNavigationBarItem(
              icon: Padding(
                padding:
EdgeInsets.only(bottom: 4),
                child:
Icon(Icons.library_music),
              ),
              color: Colors.white,
            ),
            unselectedIconTheme: const
IconThemeData(
              size: 24,
              color: Colors.grey,
            ),
            selectedLabelStyle: const
TextStyle(
              fontSize: 12,
              fontWeight: FontWeight.bold,
            ),
            unselectedLabelStyle: const
TextStyle(
              fontSize: 12,
            ),
            items: const [
              BottomNavigationBarItem(
                icon: Padding(
                  padding:
EdgeInsets.only(bottom: 4),
                  child: Icon(Icons.home_filled),
                ),
                label: 'Home',
              ),
              BottomNavigationBarItem(
                icon: Padding(
                  padding:
EdgeInsets.only(bottom: 4),
                  child: Icon(Icons.search),
                ),
                label: 'Search',
              ),
```

Sign_in.dart

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

import 'register.dart';

class SignInScreen extends StatefulWidget
```

```
            label: 'Library',
           ),
           BottomNavigationBarItem(
            icon: Padding(
             padding:
EdgeInsets.only(bottom: 4),
             child: Icon(Icons.person),
            ),
            label: 'Profile',
           ),
          ],
         ),
        ),
       ),
     );
   }
}
```

```
Navigator.pop(context),
                ),
                const SizedBox(width: 12),
                Image.asset(

'assets/images/spotify_logo.png',
                  height: 150,
                ),
```

```
{
 const SignInScreen({Key? key}) :
super(key: key);

 @override
 State<SignInScreen> createState() =>
_SignInScreenState();
}

class _SignInScreenState extends
State<SignInScreen> {
 bool _obscurePassword = true;

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   backgroundColor: const
Color(0xFF1C1B1B),
    body: SafeArea(
     child: SingleChildScrollView(
      child: Padding(
       padding: const
EdgeInsets.all(24.0),
        child: Column(
         crossAxisAlignment:
CrossAxisAlignment.start,
         children: [
          Row(
           children: [
            IconButton(
             icon: const
Icon(Icons.arrow_back, color:
Colors.white),
              onPressed: () =>
          hintText: 'Enter Username Or
Email',
            hintStyle: const
TextStyle(color: Colors.grey),
            filled: true,
            fillColor:
Colors.grey.withOpacity(0.1),
            border: OutlineInputBorder(
```

```
            ],
          ),
          const SizedBox(height: 32),
          const Text(
            'Sign In',
            style: TextStyle(
              color: Colors.white,
              fontSize: 28,
              fontWeight: FontWeight.bold,
            ),
          ),
          const SizedBox(height: 8),
          Row(
            children: [
              const Text(
                'If You Need Any Support ',
                style: TextStyle(color:
Colors.grey),
              ),
              TextButton(
                onPressed: () {},
                child: const Text(
                  'Click Here',
                  style: TextStyle(color:
Colors.green),
                ),
              ),
            ],
          ),
          const SizedBox(height: 32),
          TextField(
            style: const TextStyle(color:
Colors.white),
            decoration: InputDecoration(
 setState(() {
                _obscurePassword =
!_obscurePassword;
              });
            },
          ),
        ),
      ),
                borderRadius:
BorderRadius.circular(12),
                borderSide:
BorderSide.none,
              ),
            ),
          ),
          const SizedBox(height: 16),
          TextField(
            obscureText:
_obscurePassword,
            style: const TextStyle(color:
Colors.white),
            decoration: InputDecoration(
              hintText: 'Password',
              hintStyle: const
TextStyle(color: Colors.grey),
              filled: true,
              fillColor:
Colors.grey.withOpacity(0.1),
              border: OutlineInputBorder(
                borderRadius:
BorderRadius.circular(12),
                borderSide:
BorderSide.none,
              ),
              suffixIcon: IconButton(
                icon: Icon(
                  _obscurePassword
                      ? Icons.visibility_off
                      : Icons.visibility,
                color: Colors.grey,
                ),
                onPressed: () {
                                ),
              ),
              child: const Text(
                'Sign In',
                style: TextStyle(
                  fontSize: 16,
                  fontWeight:
FontWeight.bold,
```

```
            const SizedBox(height: 16),
            Align(
              alignment:
Alignment.centerRight,
              child: TextButton(
                onPressed: () {},
                child: const Text(
                  'Recovery Password',
                  style: TextStyle(color:
Colors.grey),
                ),
              ),
            ),
            const SizedBox(height: 32),
            SizedBox(
              width: double.infinity,
              height: 56,
              child: ElevatedButton(
                onPressed: () {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                      builder: (context) =>
const MainScreen()),
                  );
                },
                style:
ElevatedButton.styleFrom(
                  backgroundColor: const
Color(0xFF42C83C),
                  shape:
RoundedRectangleBorder(
                    borderRadius:
BorderRadius.circular(30),
Colors.grey),
                ),
                TextButton(
                  onPressed: () {
                    Navigator.push(
                      context,
                      MaterialPageRoute(
                        builder: (context) =>
                ),
              ),
            ),
            const SizedBox(height: 16),
            const Center(
              child: Text(
                'Or',
                style: TextStyle(color:
Colors.grey),
              ),
            ),
            const SizedBox(height: 16),
            Row(
              mainAxisAlignment:
MainAxisAlignment.center,
              children: [

Image.asset('assets/images/google_logo.pn
g', height: 40),
                const SizedBox(width: 20),

Image.asset('assets/images/apple_logo.png
', height: 40),
              ],
            ),
            const SizedBox(height: 32),
            Row(
              mainAxisAlignment:
MainAxisAlignment.center,
              children: [
                const Text(
                  'Not A Member? ',
                  style: TextStyle(color:
```

Register_screen.dart

```
import 'package:flutter/material.dart';
import
'package:spotify_clone/features/auth/scree
ns/sign_in_screen.dart';
import
'package:spotify_clone/features/auth/scree
ns/register.dart';
```

```
const Register()),
                    );
                  },
                  child: const Text(
                   'Register Now',
                   style: TextStyle(color:
Colors.blue),
                  ),
                ),
              ],
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

```
class RegisterScreen extends
StatelessWidget {
 const RegisterScreen({Key? key}) :
super(key: key);

 @override
 Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: const
Color(0xFF1C1B1B),
    body: SafeArea(
     child: Column(
      children: [
       Padding(
        padding: const
EdgeInsets.all(16.0),
        child: Row(
         children: [
          IconButton(
           icon: const
Icon(Icons.arrow_back, color:
Colors.white),
            onPressed: () =>
Navigator.pop(context),
          ),
         ],
        ),
       ),
       Expanded(
        child: SingleChildScrollView(
         child: Padding(
          padding: const
EdgeInsets.symmetric(horizontal: 24.0),
                                builder:
(context) =>
                    Register()), //
Navigate to ThirdScreen
                );
                // Handle register
               },
               style:
```

```
child: Column(
          crossAxisAlignment:
CrossAxisAlignment.center,
          children: [
           Image.asset(

'assets/images/spotify_logo.png',
            width: //235
```

```
                    500,
                  height: 150,
                ),
                const SizedBox(height: 32),
                const Text(
                  'Enjoy Listening To Music',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 24,
                    fontWeight:
FontWeight.bold,
                  ),
                ),
                const SizedBox(height: 16),
                const Text(
                  'Spotify is a proprietary
Swedish audio\nstreaming and media
services provider',
                  textAlign:
TextAlign.center,
                  style: TextStyle(
                    color: Colors.grey,
                    fontSize: 16,
                  ),
                ),
                const SizedBox(height: 32),
                SizedBox(
                  width: double.infinity,
                  height: 56,
                  child: ElevatedButton(
                    onPressed: () {
                      Navigator.push(
                        context,
                        MaterialPageRoute(
style: TextStyle(
                      color: Colors.white,
                      fontSize: 16,
                    ),
                  ),
                ),
              ],
            ),
ElevatedButton.styleFrom(
                      backgroundColor: const
Color(0xFF42C83C),
                      shape:
RoundedRectangleBorder(
                        borderRadius:
BorderRadius.circular(30),
                      ),
                    ),
                    child: const Text(
                      'Register',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 16,
                        fontWeight:
FontWeight.bold,
                      ),
                    ),
                  ),
                ),
                const SizedBox(height: 16),
                TextButton(
                  onPressed: () {
                    Navigator.push(
                      context,
                      MaterialPageRoute(
                        builder: (context) =>
                            SignInScreen()), //
Navigate to ThirdScreen
                    );
                    // Handle sign in
                  },
                  child: const Text(
                    'Sign in',
```

Home_screen.dart

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

// First, add this typography class at the
top of your file
```

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

```
body: Stack(
    children: [
      Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
            colors: [
```

```
class AppTypography {
 static final TextStyle displayLarge =
GoogleFonts.poppins(
   fontSize: 24,
   fontWeight: FontWeight.bold,
   letterSpacing: -0.3,
   color: Colors.white,
 );

 static final TextStyle titleMedium =
GoogleFonts.poppins(
   fontSize: 16,
   fontWeight: FontWeight.w600,
   letterSpacing: -0.2,
   color: Colors.white,
 );

 static final TextStyle bodySmall =
GoogleFonts.poppins(
   fontSize: 12,
   fontWeight: FontWeight.normal,
   letterSpacing: 0.1,
   color: Colors.grey[400],
 );
}

class HomeScreen extends
StatelessWidget {
 const HomeScreen({Key? key}) :
super(key: key);

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   8),
                  decoration:
BoxDecoration(
                    color: const
Color(0xFF1DB954),
                    borderRadius:
BorderRadius.circular(20),
                  ),
```

```
            Colors.black.withOpacity(0.8),
            Colors.black,
          ],
        ),
      ),
    ),
    SingleChildScrollView(
      child: SafeArea(
        child: Column(
          crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
            Padding(
              padding: const
EdgeInsets.all(16.0),
              child: Row(
                children: [
                  // Updated CircleAvatar
with specific size and styling
                  const CircleAvatar(
                    radius: 20,
                    backgroundColor:
Color(0xFF282828),
                    child: Icon(
                      Icons.person,
                      color: Colors.white,
                      size: 24,
                    ),
                  ),
                  const SizedBox(width: 16),
                  Container(
                    padding: const
EdgeInsets.symmetric(
                      horizontal: 16, vertical:
 child: const Text('Podcasts',
                      style: TextStyle(color:
Colors.white)),
                    ),
                  ],
                ),
              ),
              const PlaylistSection(
                  child: const Text('All',
                    style: TextStyle(color:
Colors.white)),
                  ),
                  const SizedBox(width: 8),
                  Container(
                    padding: const
EdgeInsets.symmetric(
                      horizontal: 16, vertical:
8),
                    decoration:
BoxDecoration(
                      color: const
Color(0xFF282828),
                      borderRadius:
BorderRadius.circular(20),
                    ),
                    child: const Text('Music',
                      style: TextStyle(color:
Colors.white)),
                  ),
                  const SizedBox(width: 8),
                  Container(
                    padding: const
EdgeInsets.symmetric(
                      horizontal: 16, vertical:
8),
                    decoration:
BoxDecoration(
                      color: const
Color(0xFF282828),
                      borderRadius:
BorderRadius.circular(20),
                    ),

                  SizedBox(width: 16),
                  MixCard(
                    title: 'Chill Mix',
                    subtitle: 'Kendrick Lamar,
The Weeknd, Future',
                    image:
'assets/images/chill.jpg',
```

```
                title: 'On Repeat',
                image:
'assets/images/on_repeat.jpg',
                isLarge: true,
              ),
              const PlaylistSection(
                title: 'Trending Now India',
                image:
'assets/images/trending_india.jpg',
                isLarge: true,
              ),
              const PlaylistSection(
                title: 'Liked Songs',
                image:
'assets/images/liked_songs.jpg',
                isLarge: true,
              ),
              const SectionTitle(title: 'Your
top mixes'),
              SingleChildScrollView(
                scrollDirection:
Axis.horizontal,
                padding: const
EdgeInsets.symmetric(horizontal: 16),
                child: Row(
                  children: const [
                    MixCard(
                      title: 'Arijit Singh Mix',
                      subtitle: 'Pritam,
Vishal-Shekhar and Atif Aslam',
                      image:
'assets/images/arijit.jpg',
                    ),

const PlaylistSection({
  Key? key,
  required this.title,
  required this.image,
  this.isLarge = false,
}) : super(key: key);

@override
                    ),
                    SizedBox(width: 16),
                    MixCard(
                      title: 'R&B Mix',
                      subtitle: 'Kali Uchis,
SZA',
                      image:
'assets/images/rnb.jpg',
                    ),
                  ],
                ),
              ),
              const SizedBox(height: 100),
            ],
          ),
        ),
        const Positioned(
          left: 0,
          right: 0,
          bottom: 0,
          child: NowPlayingBar(),
        ),
      ],
    ),
  );
 }
}

class PlaylistSection extends
StatelessWidget {
 final String title;
 final String image;
 final bool isLarge;

              size: 24,
            ),
            onPressed: () {},
          ),
        ),
      ],
    ),
```

```
  Widget build(BuildContext context) {
   return Padding(
     padding: const EdgeInsets.all(16.0),
     child: Row(
      children: [
       ClipRRect(
        borderRadius:
BorderRadius.circular(4),
        child: Image.asset(
         image,
         width: isLarge ? 80 : 60,
         height: isLarge ? 80 : 60,
         fit: BoxFit.cover,
        ),
       ),
       const SizedBox(width: 16),
       Expanded(
        child: Text(
         title,
         style:
AppTypography.titleMedium, // Updated
typography
          overflow: TextOverflow.ellipsis,
// Added overflow handling
          maxLines: 2,
        ),
       ),
       Material(
        color: Colors.transparent,
        child: IconButton(
         icon: const Icon(
          Icons.more_vert,
          color: Colors.white,

const SizedBox(height: 8),
       Text(
        title,
        style:
AppTypography.titleMedium, // Updated
typography
        overflow: TextOverflow.ellipsis, //
Added overflow handling
    );
   }
  }

class MixCard extends StatelessWidget {
  final String title;
  final String subtitle;
  final String image;

  const MixCard({
   Key? key,
   required this.title,
   required this.subtitle,
   required this.image,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
   return SizedBox(
     width: 160,
     child: Column(
      crossAxisAlignment:
CrossAxisAlignment.start,
      children: [
       ClipRRect(
        borderRadius:
BorderRadius.circular(4),
        child: Image.asset(
         image,
         width: 160,
         height: 160,
         fit: BoxFit.cover,
        ),
       ),
                width: 40,
        height: 40,
        fit: BoxFit.cover,
       ),
       ),
       const SizedBox(width: 12),
       Expanded(
        child: Column(
```

```
          maxLines: 1,
        ),
        const SizedBox(height: 4),
        Text(
          subtitle,
          style: AppTypography.bodySmall,
// Updated typography
          overflow: TextOverflow.ellipsis, //
Added overflow handling
          maxLines: 2,
        ),
      ],
    ),
  );
 }
}

class NowPlayingBar extends
StatelessWidget {
 const NowPlayingBar({Key? key}) :
super(key: key);

 @override
 Widget build(BuildContext context) {
  return Container(
   color: const Color(0xFF282828),
   padding: const EdgeInsets.all(8),
   child: Row(
    children: [
     ClipRRect(
      borderRadius:
BorderRadius.circular(4),
      child: Image.asset(
       'assets/images/perfect_1d.jpg',
 ),
     ),
     Material(
      color: Colors.transparent,
      child: IconButton(
       icon: const Icon(
        Icons.favorite_border,
        color: Colors.white,
        size: 24,
       ),
       onPressed: () {},
      ),
     ),
     Expanded(
      child: Column(
       crossAxisAlignment:
CrossAxisAlignment.start,
       mainAxisSize:
MainAxisSize.min,
       children: [
        Text(
         'Perfect',
         style:
AppTypography.titleMedium, // Updated
typography
         overflow:
TextOverflow.ellipsis,
        ),
        Text(
         'One Direction',
         style:
AppTypography.bodySmall, // Updated
typography
         overflow:
TextOverflow.ellipsis,
        ),
       ],
      ),
     ),
     Material(
      color: Colors.transparent,
      child: IconButton(
       icon: const Icon(
        Icons.devices_outlined,
        color: Colors.white,
        size: 24,
       ),
       onPressed: () {},
      ),
     ),
      style:
AppTypography.displayLarge, // Updated
typography
    ),
   );
  }
}
```

```
            ),
            onPressed: () {},
          ),
        ),
        Material(
          color: Colors.transparent,
          child: IconButton(
            iconSize: 32,
            icon: const Icon(
              Icons.play_circle_filled,
              color: Colors.white,
            ),
            onPressed: () {},
          ),
        ),
      ],
    ),
  );
 }
}

class SectionTitle extends StatelessWidget
{
 final String title;

 const SectionTitle({Key? key, required
this.title}) : super(key: key);

 @override
 Widget build(BuildContext context) {
  return Padding(
    padding: const EdgeInsets.all(16.0),
    child: Text(
      Title,
```

Search_screen.dart

```
import 'package:flutter/material.dart';
import
'package:flutter_riverpod/flutter_riverpod.
dart';
import
'package:spotify_clone/models/search_res
ult.dart';
```

```
Future<void> _performSearch(String
query) async {
   if (query.isEmpty) return;

   setState(() {
     _isLoading = true;
   });
```

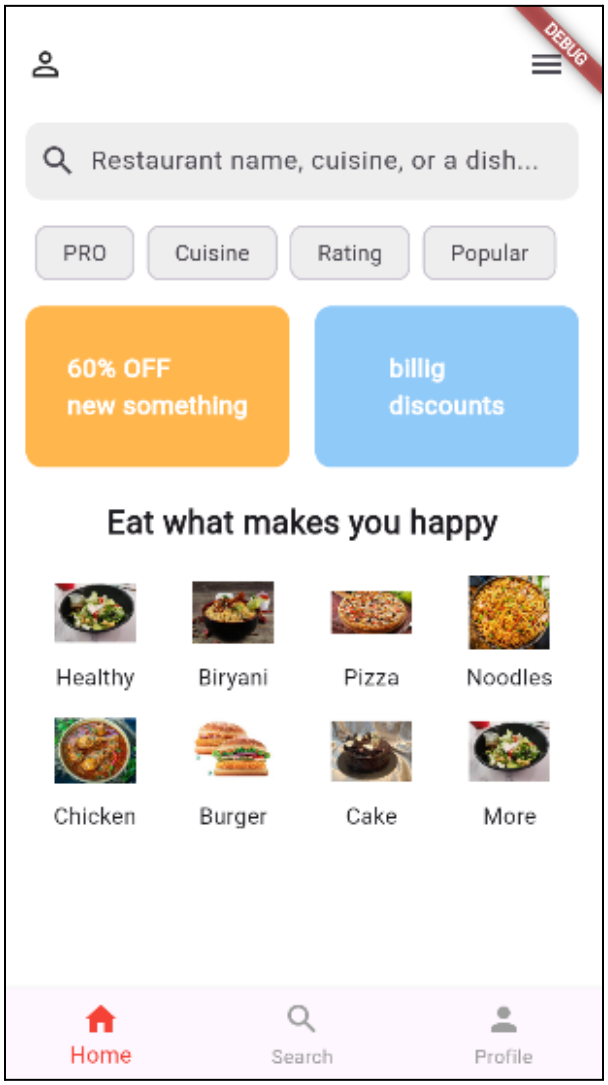
```
import
'package:spotify_clone/services/youtube_s
ervice.dart';
import
'package:spotify_clone/providers/audio_pr
ovider.dart';
import
'package:spotify_clone/services/youtube_e
xplode_service.dart';

final youtubeServiceProvider =
Provider((ref) => YouTubeService());
final audioPlayerProvider =
StateNotifierProvider<AudioPlayerNotifie
r, AudioPlayerState>((ref) =>
AudioPlayerNotifier());

class SearchScreen extends
ConsumerStatefulWidget {
  const SearchScreen({Key? key}) :
super(key: key);

  @override
  ConsumerState<SearchScreen>
createState() => _SearchScreenState();
}

class _SearchScreenState extends
ConsumerState<SearchScreen> {
  final TextEditingController
_searchController =
TextEditingController();
  List<SearchResult> _searchResults = [];
  bool _isLoading = false;

      body: SafeArea(
      child: Column(
        children: [
          Padding(
            padding: const
EdgeInsets.all(16.0),
            child: TextField(
    final youtubeService =
ref.read(youtubeServiceProvider);
    final results = await
youtubeService.searchSongs(query);

    setState(() {
      _searchResults = results;
      _isLoading = false;
    });
  }

  Future<void> _playSong(SearchResult
song) async {
    final youtubeService =
ref.read(youtubeServiceProvider);
    final audioUrl = await
youtubeService.getAudioUrl(song.id);

    if (audioUrl != null) {
      await
ref.read(audioPlayerProvider.notifier).play
Song(audioUrl, song);
    }
  }

  @override
  Widget build(BuildContext context) {
    final audioState =
ref.watch(audioPlayerProvider);

    return Scaffold(
      backgroundColor: const
Color(0xFF1C1B1B),

_searchResults[index];
                final isPlaying =
audioState.currentSong?.id == song.id;

                return ListTile(
                  leading:
Image.network(song.thumbnail),
                  title: Text(
```

```
            controller: _searchController,
            style: const TextStyle(color:
Colors.white),
            decoration: InputDecoration(
              hintText: 'Search for songs...',
              hintStyle: const
TextStyle(color: Colors.grey),
              filled: true,
              fillColor:
Colors.grey.withOpacity(0.1),
              border: OutlineInputBorder(
                borderRadius:
BorderRadius.circular(12),
                borderSide: BorderSide.none,
              ),
              suffixIcon: IconButton(
                icon: const Icon(Icons.search,
color: Colors.grey),
                onPressed: () =>
_performSearch(_searchController.text),
              ),
            ),
            onSubmitted: _performSearch,
          ),
        ),
        if (_isLoading)
          const CircularProgressIndicator()
        else
          Expanded(
            child: ListView.builder(
              itemCount:
_searchResults.length,
              itemBuilder: (context, index) {
                final song =
EdgeInsets.all(16),
            color: Colors.black,
            child: Row(
              children: [
                Image.network(

audioState.currentSong!.thumbnail,
                  height: 50,
                  song.title,
                  style: const
TextStyle(color: Colors.white),
                ),
                subtitle: Text(
                  song.channelTitle,
                  style: const
TextStyle(color: Colors.grey),
                ),
                trailing: IconButton(
                  icon: Icon(
                    isPlaying &&
audioState.isPlaying
                        ? Icons.pause
                        : Icons.play_arrow,
                    color: Colors.white,
                  ),
                  onPressed: () {
                    if (isPlaying) {

ref.read(audioPlayerProvider.notifier).togg
lePlay();
                    } else {
                      _playSong(song);
                    }
                  },
                ),
              );
            },
          ),
        ),
        if (audioState.currentSong != null)
          Container(
            padding: const
lePlay();
                  },
                ),
              ],
            ),
          ),
        ],
      ),
```

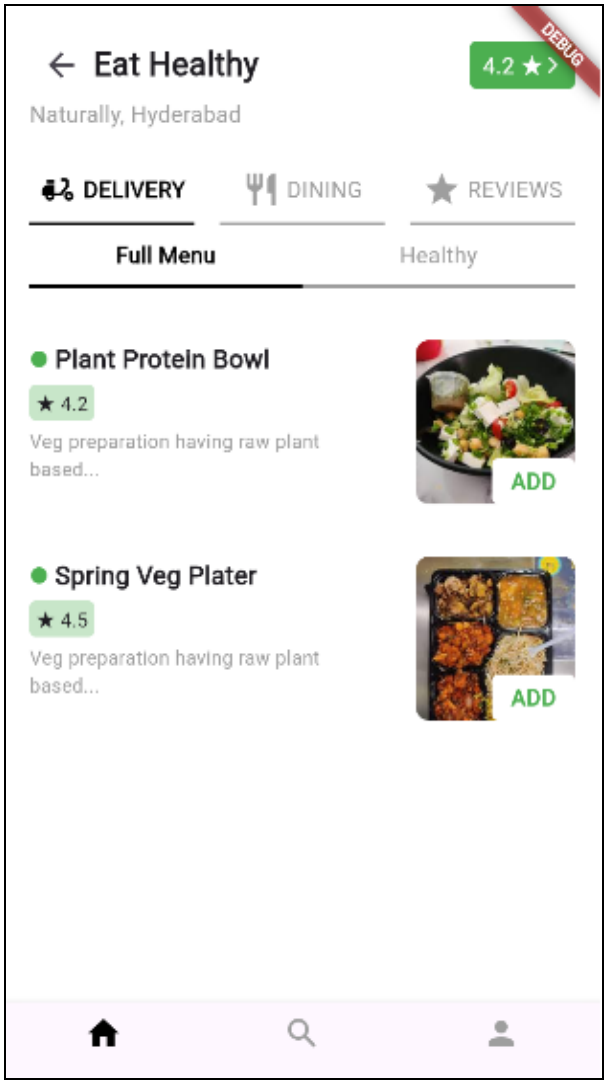
```
                ),                                          ),
                const SizedBox(width: 16),                );
                Expanded(                                  }
                  child: Column(                          }
                    crossAxisAlignment:
CrossAxisAlignment.start,
                    children: [
                      Text(

audioState.currentSong!.title,
                        style: const
TextStyle(color: Colors.white),
                        maxLines: 1,
                        overflow:
TextOverflow.ellipsis,
                      ),
                      Text(

audioState.currentSong!.channelTitle,
                        style: const
TextStyle(color: Colors.grey),
                      ),
                    ],
                  ),
                ),
                IconButton(
                  icon: Icon(
                    audioState.isPlaying ?
Icons.pause : Icons.play_arrow,
                    color: Colors.white,
                  ),
                  onPressed: () {

ref.read(audioPlayerProvider.notifier).togg
```

Project Title: Zomato **Roll No. 26**

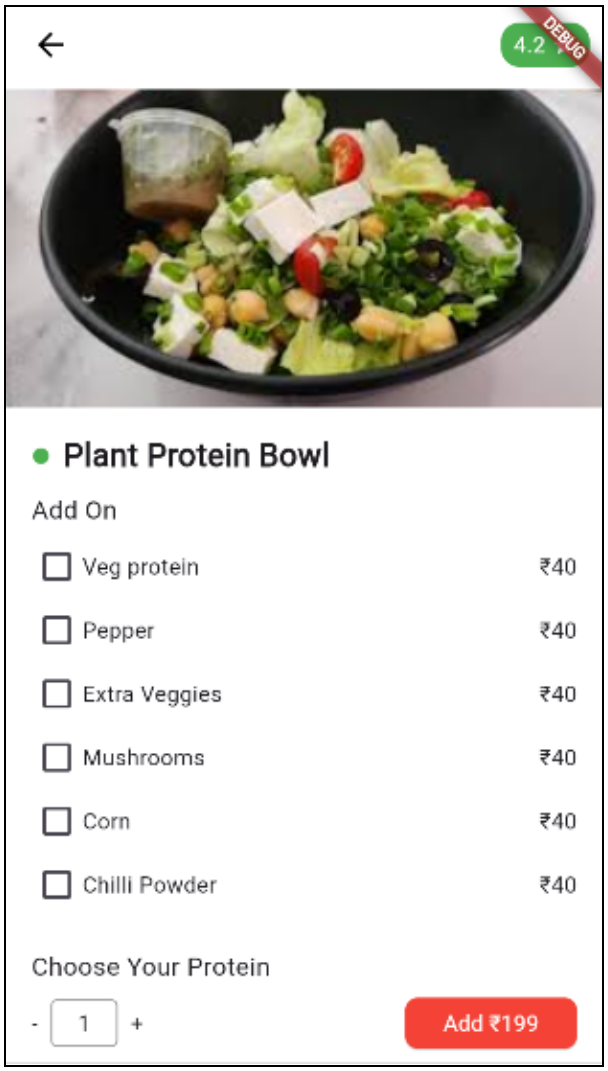
OUTPUT:



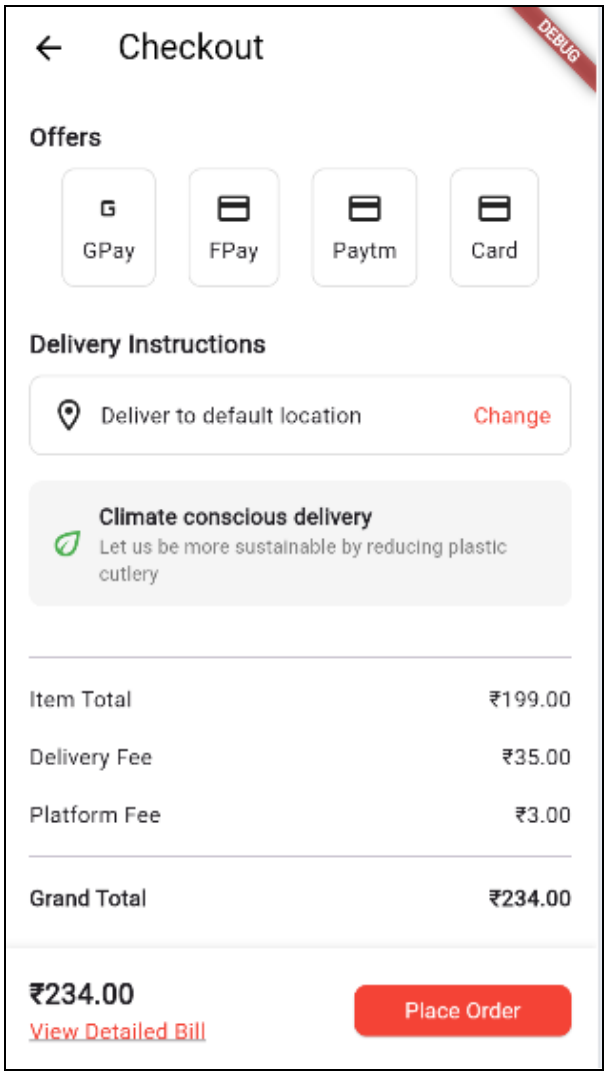
Project Title: Zomato **Roll No. 26**



Project Title: Zomato **Roll No. 26**



Project Title: Zomato **Roll No. 26**



MAD & PWA Lab
Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	26
Name	Vaishnal Dilip Mali
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:	

EXPERIMENT NO: - 06

Name:- Vaishnal Mali **Class:-** D15A **Roll:No: -** 26

AIM: - To connect Flutter UI with Firebase database.

Theory: -

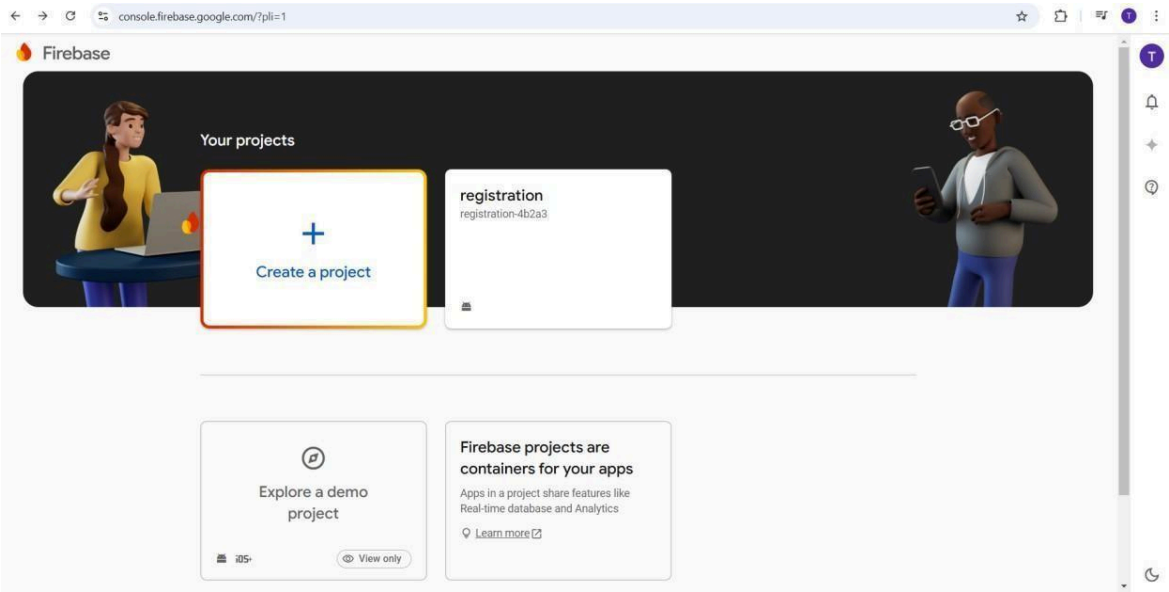
Flutter is an open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. Firebase, a Backend-as-a- Service (BaaS) platform, provides real-time database, authentication, and cloud storage services, making it a powerful backend solution for Flutter applications.

By integrating Firebase with Flutter, developers can store and retrieve data in real time, authenticate users, and manage cloud-based data efficiently. This is particularly useful for applications requiring dynamic content updates and user interactions.

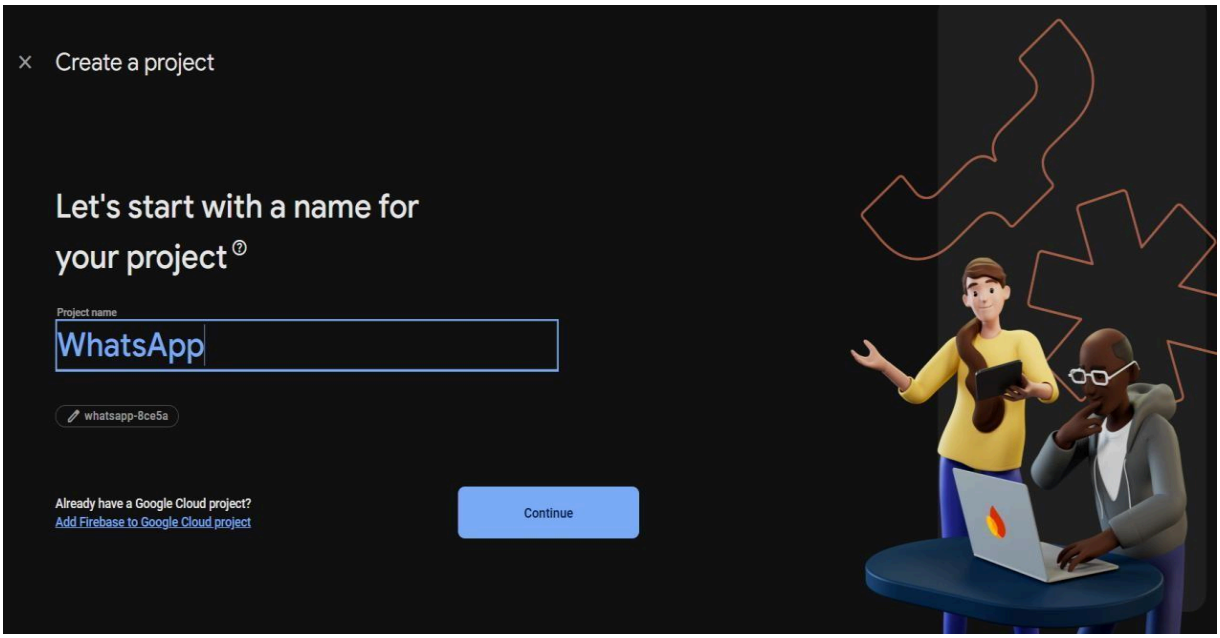
- **Steps to Connect Flutter UI with Firebase Database**

Step 1:

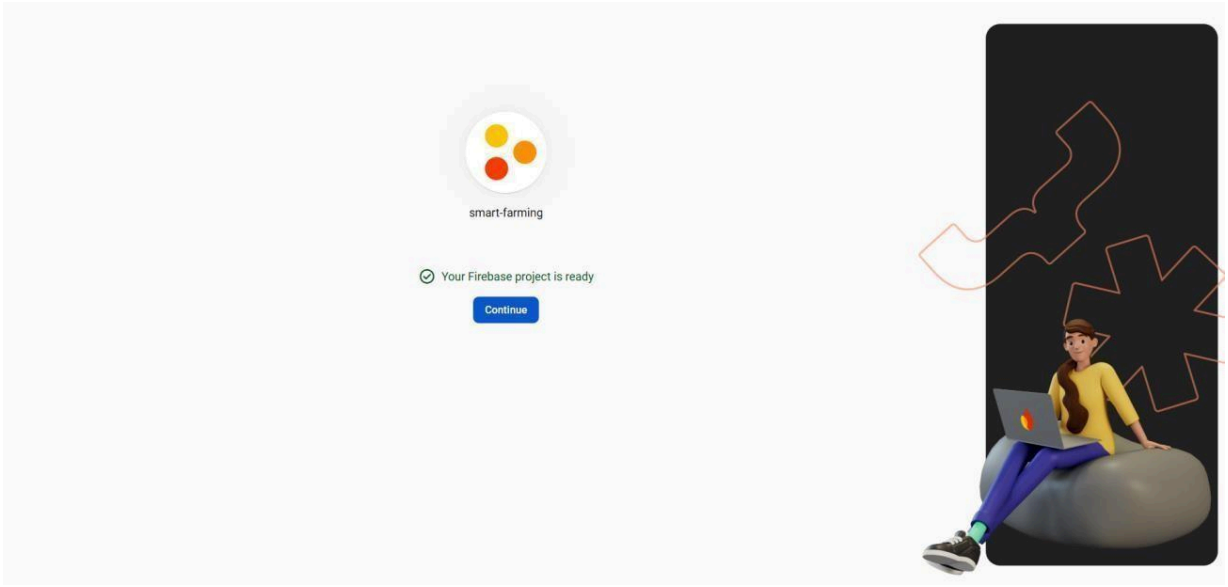
1.1) Go to Firebase Console and Create a Firebase Project



1.2) Click on Create a Project and give it a suitable name.

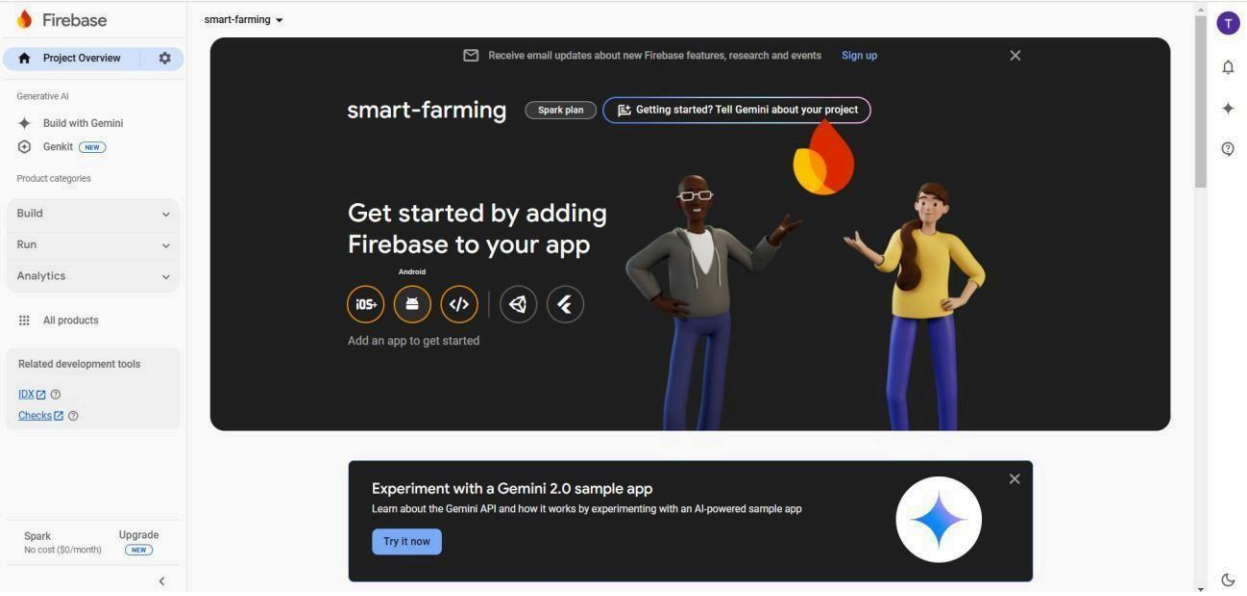


1.3) Enable Google Analytics (optional) & Click continue and complete the setup

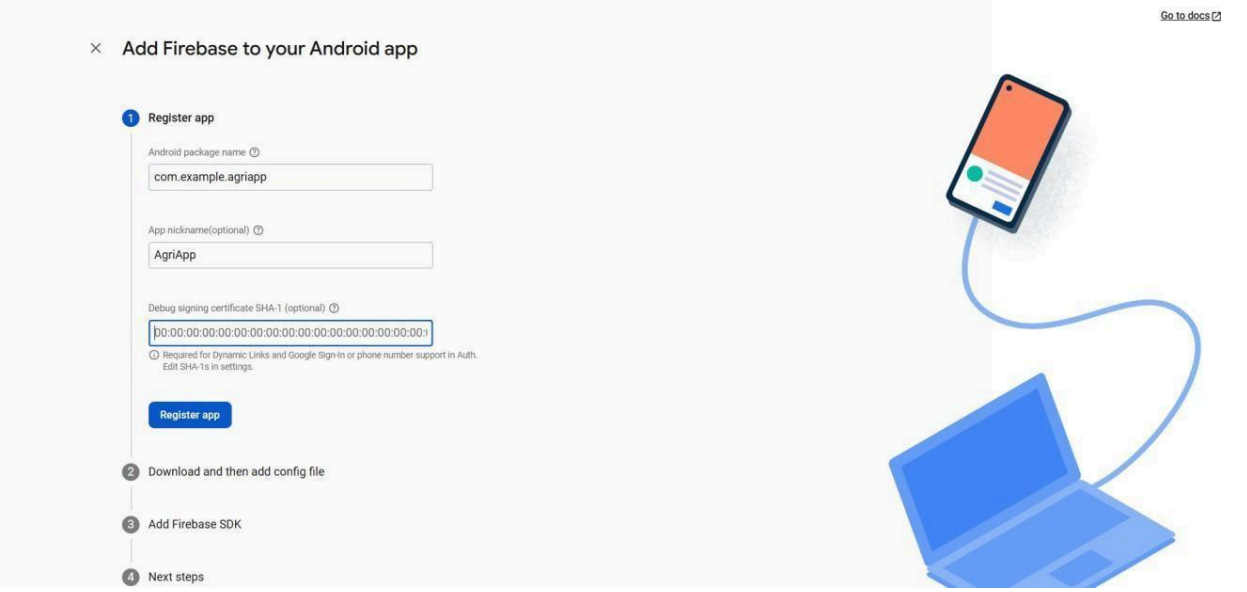


Step 2:- Add Firebase to Your Flutter App

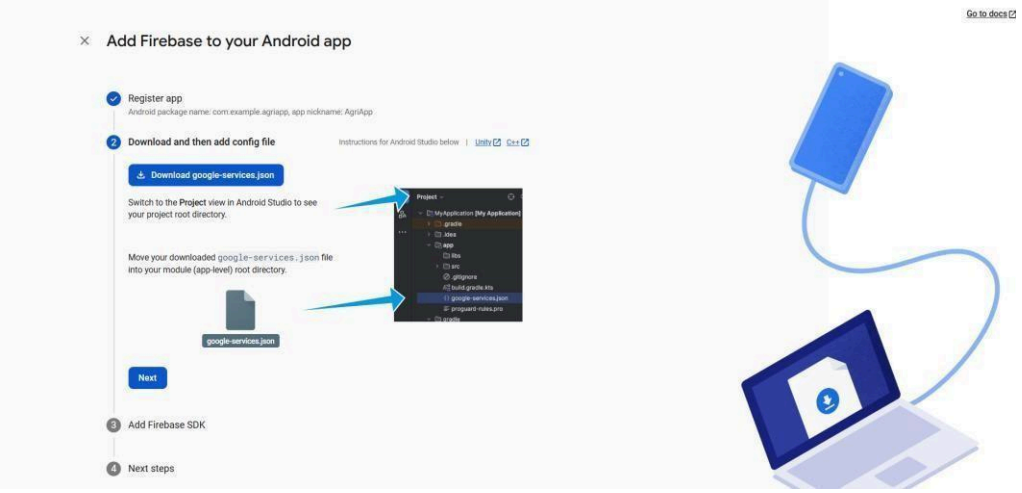
2.1) Click on Android/iOS/Web based on your Flutter application

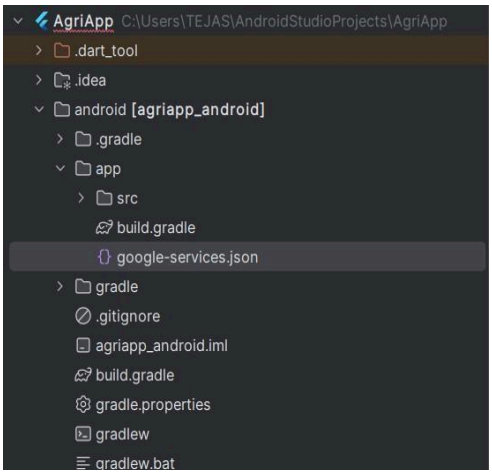


2.2) Register your app with a unique package name (found in android/app/build.gradle for Android).

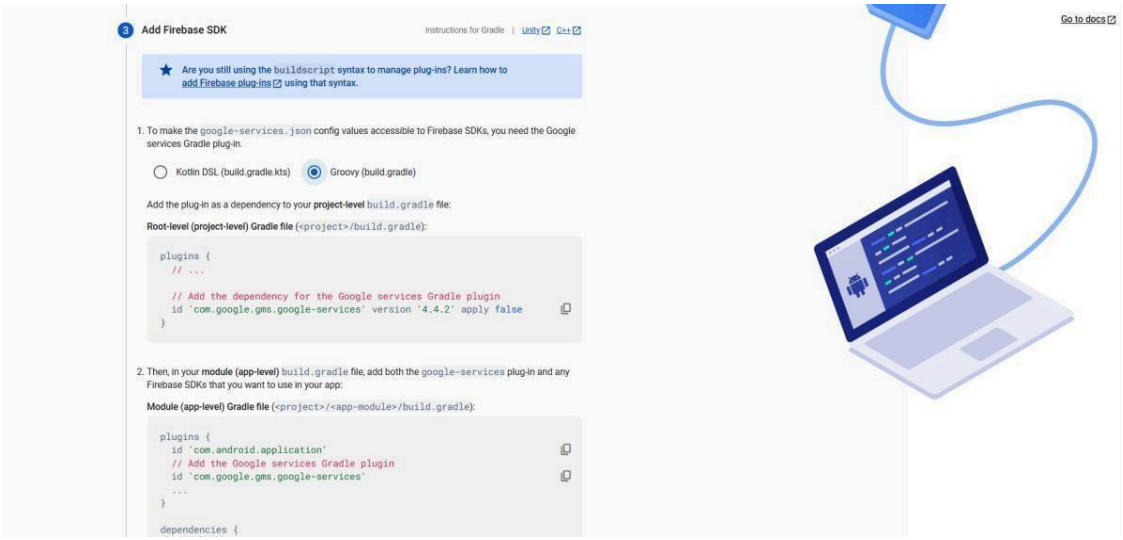


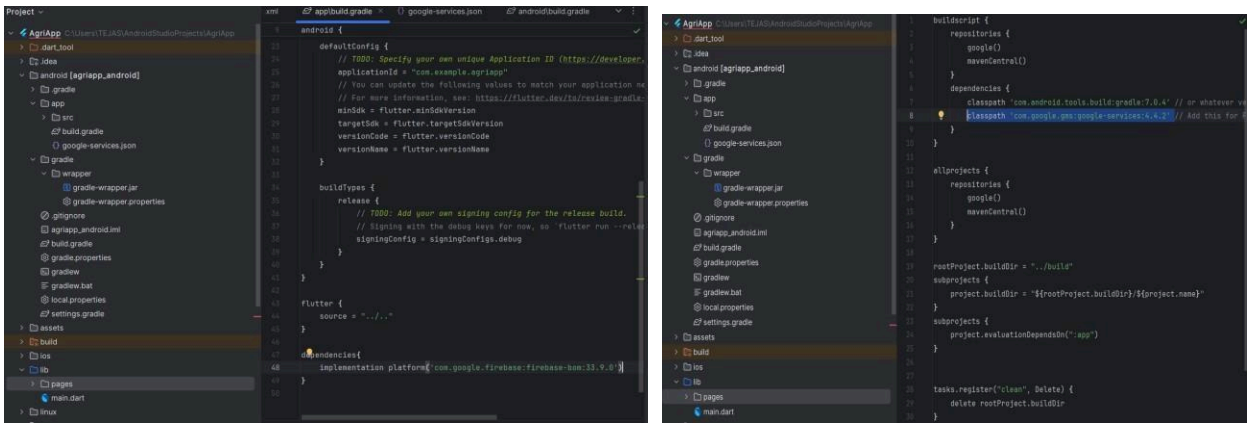
2.3) Download the google-services.json (for Android) & place the JSON file inside android/app/ directory.





2.4) Add Firebase SDK dependencies to android/build.gradle



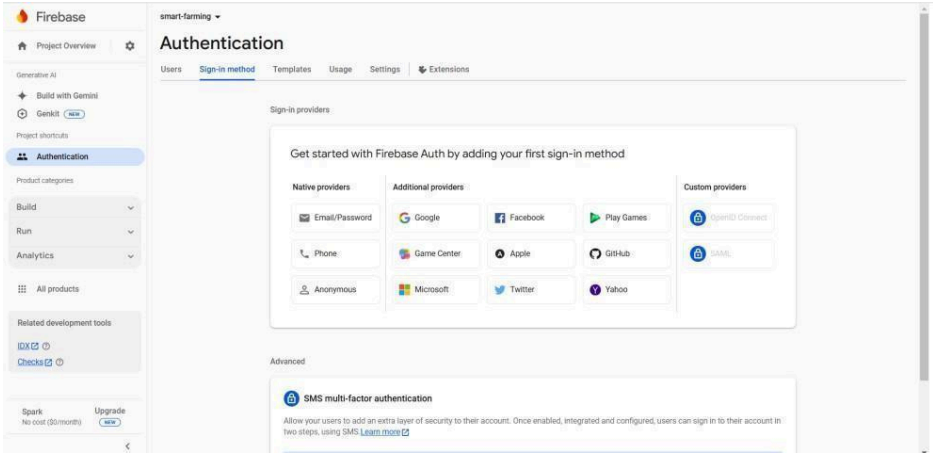


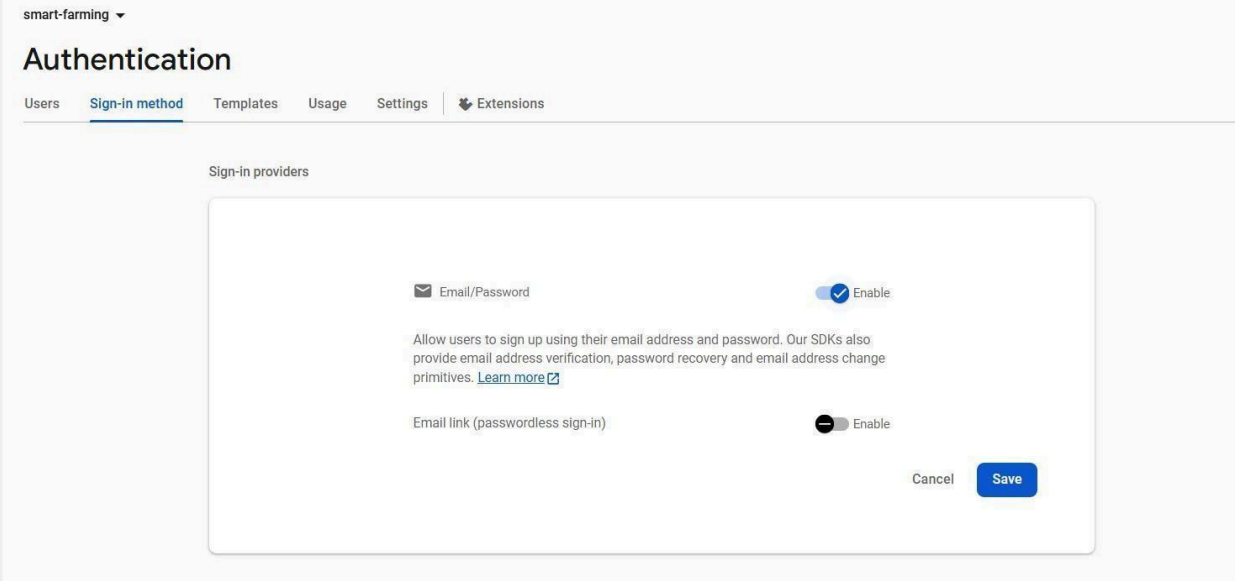
Step 3: - Add Firebase Authentication to Your App

3.1) Add Firebase Authentication Dependencies

```yaml
dependencies:
  flutter:
    sdk: flutter
  firebase_core: ^3.11.0
  firebase_auth: ^5.4.2  # For authentication
  cloud_firestore: ^5.6.3  # For Firestore, if you need it
  firebase_messaging: ^15.2.2
  http: ^0.13.3
  image_picker: ^1.0.4
  tflite_flutter: ^0.11.0
  image: ^3.2.0
  url_launcher: ^6.1.14
```

3.2) Enable Authentication in Firebase Console
Go to **Firebase Console →**
Authentication.
Click on **Sign-in method** and enable **Email/Password** (or any other method like Google).
Click Save





3.3) Implement Authentication in Flutter Modify main.dart

```
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';

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

Step 4: -Configure Firebase Realtime Database

4.1) Go to Firebase Console → Realtime Database.

4.2) Click **Create Database** → Choose location → Set rules (for development, set read/write to true).

4.3) Click **Publish**.

Code:-

Register_signup page

```
import 'package:flutter/material.dart';
import
'package:spotify_clone/features/auth/sc
 reens/sign_in_screen.dart';
import
'package:spotify_clone/features/auth/sc
 reens/register.dart';

class RegisterScreen extends
StatelessWidget {
  const RegisterScreen({Key? key}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: const
Color(0xFF1C1B1B),
body: SafeArea(
        child: Column(
          children: [
            Padding(
              padding: const
EdgeInsets.all(16.0),
              child: Row(
                children: [
                  IconButton(
                    icon: const
Icon(Icons.arrow_back, color:
Colors.white),
                    onPressed: () =>
Navigator.pop(context),
                  ),
                ],
              ),
            ),
            Expanded(

      child: SingleChildScrollView(
              child: Padding(
                padding: const
EdgeInsets.symmetric(horizontal:
24.0),
                child: Column(
                  crossAxisAlignment:
CrossAxisAlignment.center,
children: [
                    Image.asset(

'assets/images/spotify_logo.png',
                      width: //235
                          500,
                      height: 150,
                    ),
                    const SizedBox(height:
32),
                    const Text(
                      'Enjoy Listening To
Music',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 24,
                        fontWeight:
FontWeight.bold,
                      ),
                    ),
                    const SizedBox(height:
16),
                    const Text(
                      'Spotify is a proprietary
Swedish audio\nstreaming and media
services provider',
                      textAlign:
TextAlign.center,
                      style: TextStyle(
                        color: Colors.grey,
                        fontSize: 16,
                ),
                    ),
                    const SizedBox(height:
32),
                    SizedBox(
                      width: double.infinity,
                      height: 56,
                      child: ElevatedButton(
                        onPressed: () {
                          Navigator.push(
                            context,
                            MaterialPageRoute(
                              builder: (context)
=>
                                  Register()), //
Navigate to ThirdScreen
```

```
                                    style:
ElevatedButton.styleFrom(
                      backgroundColor:
const Color(0xFF42C83C),
                      shape:
RoundedRectangleBorder(
                        borderRadius:
BorderRadius.circular(30),
                      ),
                    ),
                    child: const Text(
                      'Register',
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 16,
                        fontWeight:
FontWeight.bold,
                      ),
                    ),
                  ),

                                ),
                const SizedBox(height:
16),
                TextButton(
                  onPressed: () {
                    Navigator.push(
                        context,
                        MaterialPageRoute(
                            builder: (context)
=>

SignInScreen()), // Navigate to
ThirdScreen
                      );
                    // Handle sign in
                  },
                  child: const Text(
                    'Sign in',
                    style: TextStyle(
                      color: Colors.white,
                      fontSize: 16,
                    ),
                  ),
                ),
              ],
            ),
          ),
        ),
      ),
                      );
                    // Handle register
                  },
                ],
              ),
            ),
          );
        }
      }
```

Register.dart

```
import 'package:flutter/material.dart';
import
'package:spotify_clone/features/auth/sc
 reens/sign_in_screen.dart';
import
'package:spotify_clone/main.dart';

class Register extends StatefulWidget {
  const Register({Key? key}) :
super(key: key);

  @override
  State<Register> createState() =>
_RegisterState();
}

class _RegisterState extends
State<Register> {
  bool _obscurePassword = true;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: const
Color(0xFF1C1B1B),
      body: SafeArea(
        child: SingleChildScrollView(
          child: Padding(
            padding: const
EdgeInsets.all(24.0),
            child: Column(
              crossAxisAlignment:
CrossAxisAlignment.start,
              children: [
                Row(
                  children: [
                    IconButton(
                      icon: const

Icon(Icons.arrow_back, color:
Colors.white),
                      onPressed: () =>
Navigator.pop(context),
                    ),
                    const SizedBox(width: 10),
                    Image.asset(

'assets/images/spotify_logo.png',
height: 150,
                    ),
                  ],
                ),
                const SizedBox(height: 32),
                const Text(
                  'Register',
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 28,
                    fontWeight:
FontWeight.bold,
                  ),
                ),
                const SizedBox(height: 8),
                Row(
                  children: [
                    const Text(
                      'If You Need Any Support
',
                      style: TextStyle(color:
Colors.grey),
                    ),
                    TextButton(
                      onPressed: () {},
                      child: const Text(
                        'Click Here',
                        style: TextStyle(color:
Colors.green),
  borderRadius:
),
                    ),
                  ],
                ),
                const SizedBox(height: 32),
                TextField(
                  style: const TextStyle(color:
Colors.white),
                  decoration:
InputDecoration(
                    hintText: 'Full Name',
                    hintStyle: const
TextStyle(color: Colors.grey),
                    filled: true,
                    fillColor:
Colors.grey.withOpacity(0.1),
                    border:
OutlineInputBorder(
                      borderRadius:
```

```
        BorderRadius.circular(12),

                  borderSide:
BorderSide.none,
                ),
              ),
            ),
            const SizedBox(height: 16),
            TextField(
              style: const TextStyle(color:
Colors.white),
              decoration:
InputDecoration(
                hintText: 'Enter Email',
                hintStyle: const
TextStyle(color: Colors.grey),
                filled: true,
                fillColor:
Colors.grey.withOpacity(0.1),
                border:
OutlineInputBorder(

BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
              ),
            ),
            const SizedBox(height: 16),
            TextField(
              obscureText:
_obscurePassword,
              style: const TextStyle(color:
Colors.white),
              decoration:
InputDecoration(
                hintText: 'Password',
                hintStyle: const
TextStyle(color: Colors.grey),
                filled: true,
                fillColor:
Colors.grey.withOpacity(0.1),
                border:
OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
                suffixIcon: IconButton(
 icon: Icon(
                    _obscurePassword
                      ? Icons.visibility_off
                      : Icons.visibility,
                    color: Colors.grey,
                  ),
                  onPressed: () {
                    setState(() {
                      _obscurePassword =
!_obscurePassword;
Row(
                                        });
                  },
                ),
              ),
            ),
            const SizedBox(height: 32),
            SizedBox(
              width: double.infinity,
              height: 56,
              child: ElevatedButton(
                onPressed: () {
                  Navigator.push(
                    context,
                    MaterialPageRoute(
                      builder: (context) =>
const MainScreen()),
                  );
                },
                style:
ElevatedButton.styleFrom(
                  backgroundColor: const
Color(0xFF42C83C),
                  shape:
RoundedRectangleBorder(
                    borderRadius:
BorderRadius.circular(30),
                  ),
                ),
                child: const Text(
                  'Create Account',
                  style: TextStyle(
                    fontSize: 16,
                    fontWeight:
FontWeight.bold,
                  ),
                ),
              ),
            ),
            const SizedBox(height: 32),
```

```
            mainAxisAlignment:
MainAxisAlignment.center,
            children: [

Image.asset('assets/images/google_logo
 .png', height: 40),
              const SizedBox(width: 20),

Image.asset('assets/images/apple_logo.
 png', height: 40),
             ],
           ),
           const SizedBox(height: 32),
           Row(
            mainAxisAlignment:
MainAxisAlignment.center,
            children: [
             const Text(
              'Do You Have An
Account? ',
               style: TextStyle(color:
Colors.grey),
             ),
             TextButton(
              onPressed: () {
               Navigator.push(
                context,
                MaterialPageRoute(
                   builder: (context) =>
const SignInScreen()),
                );
              },
              child: const Text(
               'Sign In',
               style: TextStyle(color:
Colors.white),
              ),
Sign_in_screen.dart

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

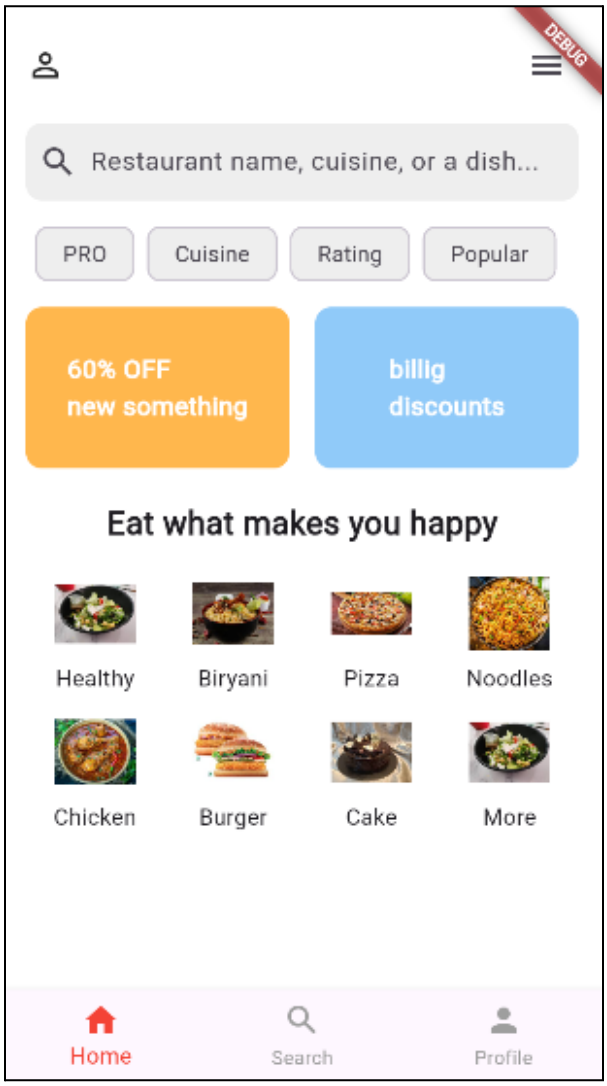
Register.dart

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

import 'register.dart';

class SignInScreen extends
StatefulWidget {
 const SignInScreen({Key? key}) :
super(key: key);

 @override
 State<SignInScreen> createState() =>
_SignInScreenState();
}

class _SignInScreenState extends
State<SignInScreen> {
 bool _obscurePassword = true;

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   backgroundColor: const
Color(0xFF1C1B1B),
   body: SafeArea(
    child: SingleChildScrollView(
     child: Padding(
      padding: const
EdgeInsets.all(24.0),
      child: Column(
       crossAxisAlignment:
CrossAxisAlignment.start,
       children: [
        Row(
         children: [
          IconButton(
           Colors.green),
           ),
 icon: const Icon(Icons.arrow_back,
color: Colors.white),
           onPressed: () =>
Navigator.pop(context),
          ),
          const SizedBox(width: 12),
```

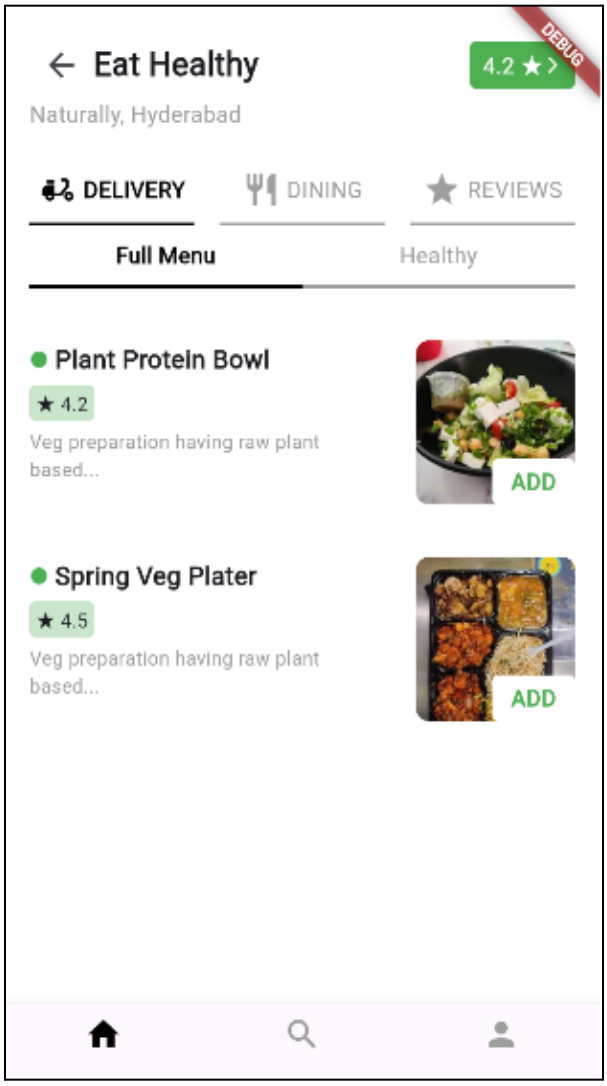
```
'assets/images/spotify_logo.png',
                  height: 150,
                ),
              ],
            ),
            const SizedBox(height: 32),
            const Text(
              'Sign In',
              style: TextStyle(
                color: Colors.white,
                fontSize: 28,
                fontWeight:
FontWeight.bold,
              ),
            ),
            const SizedBox(height: 8),
            Row(
              children: [
                const Text(
                  'If You Need Any Support
',
                  style: TextStyle(color:
Colors.grey),
                ),
                TextButton(
                  onPressed: () {},
                  child: const Text(
                    'Click Here',
                    style: TextStyle(color:

fillColor: Colors.grey.withOpacity(0.1),

                ),
              ],
            ),
            const SizedBox(height: 32),
            TextField(
              style: const TextStyle(color:
Colors.white),
              decoration:
InputDecoration(
                hintText: 'Enter Username
Or Email',
                hintStyle: const
TextStyle(color: Colors.grey),
                filled: true,
                fillColor:
Colors.grey.withOpacity(0.1),
                border:
                    Image.asset(
OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
              ),
            ),
            const SizedBox(height: 16),
            TextField(
              obscureText:
_obscurePassword,
              style: const TextStyle(color:
Colors.white),
              decoration:
InputDecoration(
                hintText: 'Password',
                hintStyle: const
TextStyle(color: Colors.grey),
                filled: true,
                          SizedBox(
              width: double.infinity,
                border:
OutlineInputBorder(
                  borderRadius:
BorderRadius.circular(12),
                  borderSide:
BorderSide.none,
                ),
                suffixIcon: IconButton(
                  icon: Icon(
                    _obscurePassword
                        ? Icons.visibility_off
                        : Icons.visibility,
                    color: Colors.grey,
                  ),
                  onPressed: () {
                    setState(() {
                      _obscurePassword =
!_obscurePassword;
                    });
                  },
                ),
              ),
            ),
            const SizedBox(height: 16),
            Align(
              alignment:
```

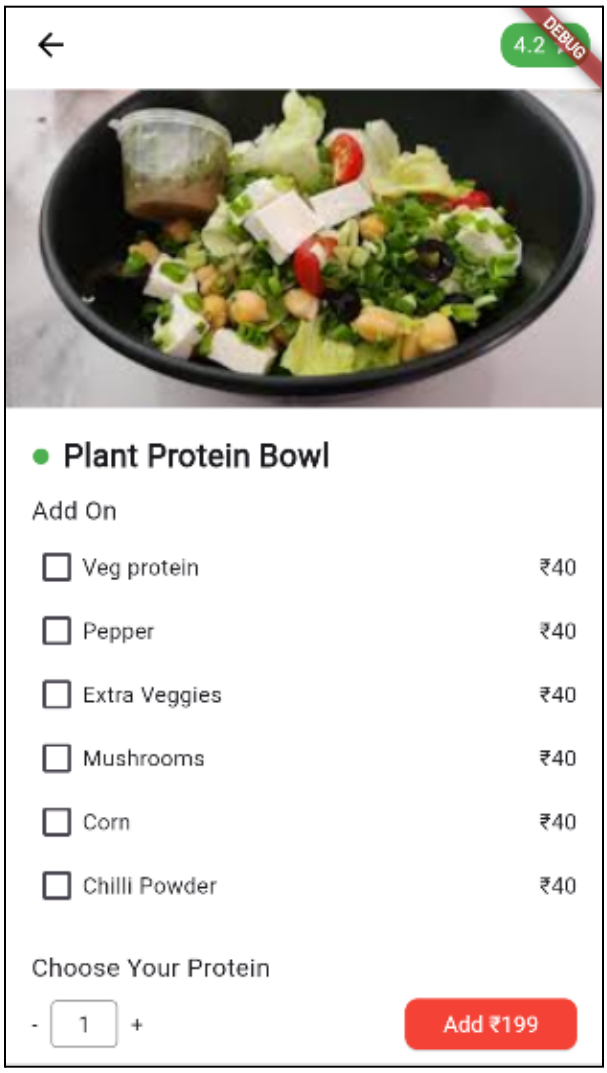
```
Alignment.centerRight,
                child: TextButton(
                  onPressed: () {},
                  child: const Text(
                    'Recovery Password',
                    style: TextStyle(color:
Colors.grey),
                  ),
                ),
              ),
              const SizedBox(height: 32),

),

                height: 56,
                child: ElevatedButton(
                  onPressed: () {
                    Navigator.push(
                      context,
                      MaterialPageRoute(
                          builder: (context) =>
const MainScreen()),
                    );
                  },
                  style:
ElevatedButton.styleFrom(
                    backgroundColor: const
Color(0xFF42C83C),
                    shape:
RoundedRectangleBorder(
                      borderRadius:
BorderRadius.circular(30),
                    ),
                  ),
                  child: const Text(
                    'Sign In',
                    style: TextStyle(
                      fontSize: 16,
                      fontWeight:
FontWeight.bold,
                    ),
                  ),
                ),
              ),
              const SizedBox(height: 16),
              const Center(
                child: Text(
                  'Or',
style: TextStyle(color:
Colors.grey),
                ),
                              ),
                ),
              const SizedBox(height: 16),
              Row(
                mainAxisAlignment:
MainAxisAlignment.center,
                children: [

Image.asset('assets/images/google_logo
.png', height: 40),
                  const SizedBox(width: 20),

Image.asset('assets/images/apple_logo.
png', height: 40),
                ],
              ),
              const SizedBox(height: 32),
              Row(
                mainAxisAlignment:
MainAxisAlignment.center,
                children: [
                  const Text(
                    'Not A Member? ',
                    style: TextStyle(color:
Colors.grey),
                  ),
                  TextButton(
                    onPressed: () {
                      Navigator.push(
                        context,
                        MaterialPageRoute(
                            builder: (context) =>
const Register()),
                      );
                    },
                    child: const Text(
                      'Register Now',
                      style: TextStyle(color:
Colors.blue),


                  ],
                ),
              ],
            ),
          ),
        ),
```

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

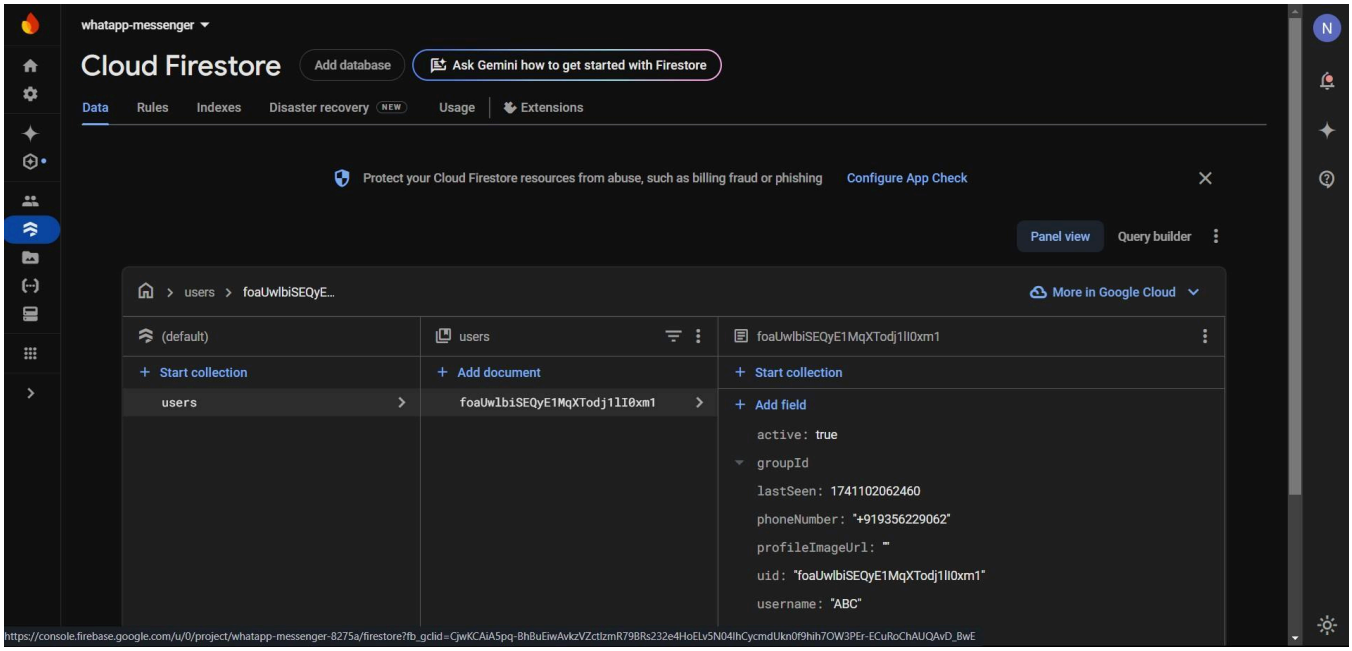

DEBUG
Restaurant name, cuisine, or a dish...
PRO
Cuisine
Rating
Popular
60% OFF
new something
billig
discounts
Eat what makes you happy
Healthy
Biryani
Pizza
Noodles
Chicken
Burger
Cake
More
Home
Search
Profile

Project Title: Zomato **Roll No. 26**




DEBUG
4.2
Plant Protein Bowl
Add On
Veg protein ₹40
Pepper ₹40
Extra Veggies ₹40
Mushrooms ₹40
Corn ₹40
Chilli Powder ₹40
Choose Your Protein
- 1 +
Add ₹199

User details get fetched from the database in the profile 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.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	

Experiment No. 7

Vaishnal Mali
D15A - 26

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

Online Reference:
https://developer.mozilla.org/en-US/docs/Web/Manifest
https://www.geeksforgeeks.org/making-a-simple-pwa-under-5-minutes/

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:
A 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.

4. Engaging Approach
As already shared, the PWAs can employ push notifications and other native device features more efficiently. Their interaction does not depend on the browser user uses. This eventually improves the chances of notifying the user regarding your services, offers, and other options related to your brand and keeping them hooked to your brand. In simpler words, PWAs let you maintain the user engagement and retention rate.

5. Updated Real-Time Data Access
Another plus point of PWAs is that these apps get updated on their own. They do not demand the end-users to go to the App Store or other such platforms to download the update and wait until installed.
In this app type, the web app developers can push the live update from the server, which reaches the apps residing on the user's devices automatically. Therefore, it is easier for the mobile app developer to provide the best of the updated functionalities and services to the end-users without forcing them to update their app.

6. Discoverable
PWAs reside in web browsers. This implies higher chances of optimizing them as per the Search Engine Optimization (SEO) criteria and improving the Google rankings like that in websites and other web apps.

7. Lower Development Cost
Progressive web apps can be installed on the user device like a native device, but it does not demand submission on an App Store. This makes it far more cost-effective than native mobile applications while offering the same set of functionalities.

Pros and cons 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.

- **Offline** — Once more it is about putting the user before everything, avoiding the usual error message in case of weak or no connection. The PWA are based on two particularities: first of all the 'skeleton' of the app, which recalls the page structure, even if its contents do not respond and its elements include the header, the page layout, as well as an illustration that signals that the page is loading.

Weaknesses refer to:

- IOS support from version 11.3 onwards;
- Greater use of the device battery;
- Not all devices support the full range of PWA features (same speech for iOS and Android operating systems);
- It is not possible to establish a strong re-engagement for iOS users (URL scheme, standard web notifications);
- Support for offline execution is however limited;
- Lack of presence on the stores (there is no possibility to acquire traffic from that channel);
- There is no "body" of control (like the stores) and an approval process;
- Limited access to some hardware components of the devices;
- Little flexibility regarding "special" content for users (eg loyalty programs, loyalty, etc.).

Code:

manifest.json:-

```
{
  "name": "Xtrack",
  "short_name": "Xtrack",
  "start_url": "ani.html",
  "scope": "./",
  "icons": [
    {
     "src": "contract.png",
     "sizes": "192x192",
     "type": "image/png"
    },
    {
```

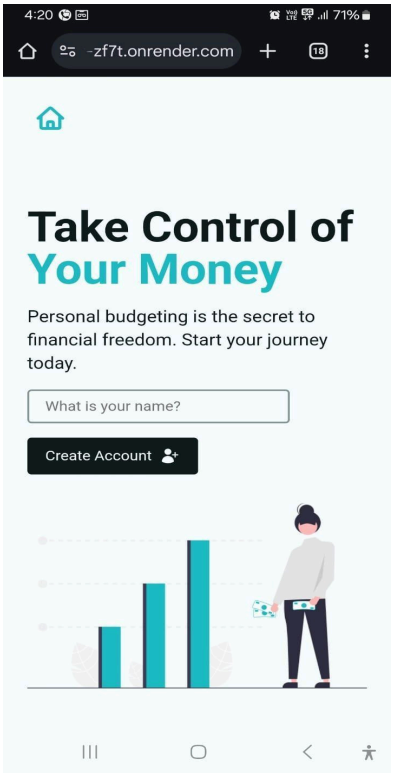
```
        "src": "contract.png",
        "sizes": "512x512",
        "type": "image/png"
       }
     ],
     "theme_color": "#ffd31d",
   "background_color": "#333",
   "display": "standalone"
 }
```

Add the link tag to link to the manifest.json file

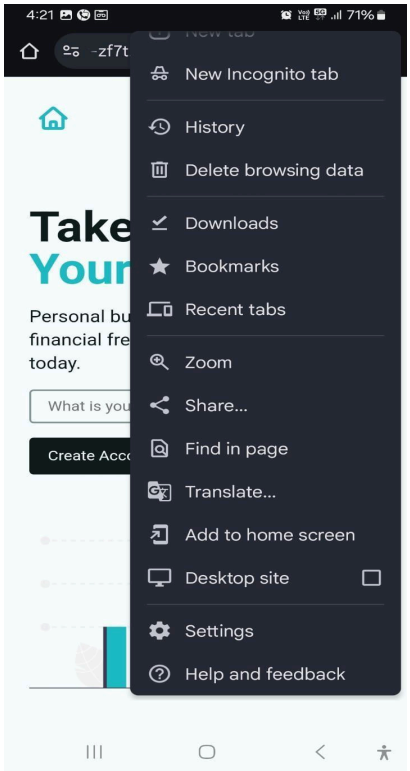
```
<html>
   <head>
      <link rel="manifest" href="manifest.json">
      <script src="myscript.js"></script>
      <title>Xtrack</title>
      <style>
```

Output with Steps:

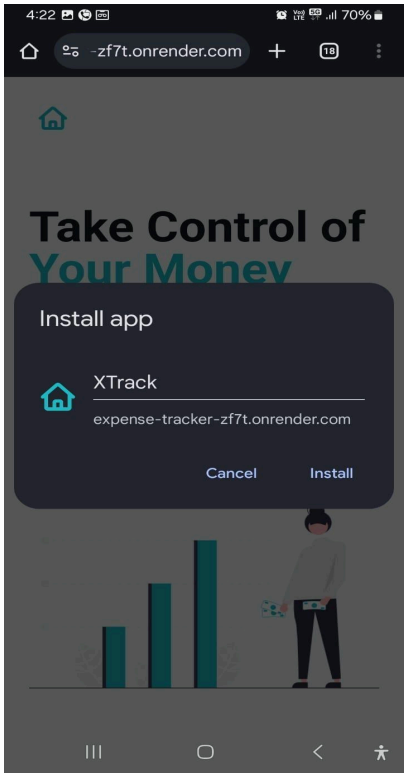
1. Go to google chrome and open your website link:



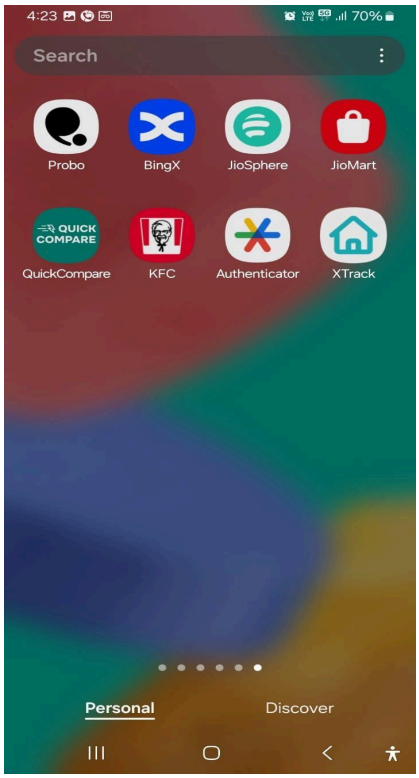
2. Then click on three dots at top right corner, find and click on add to home screen:



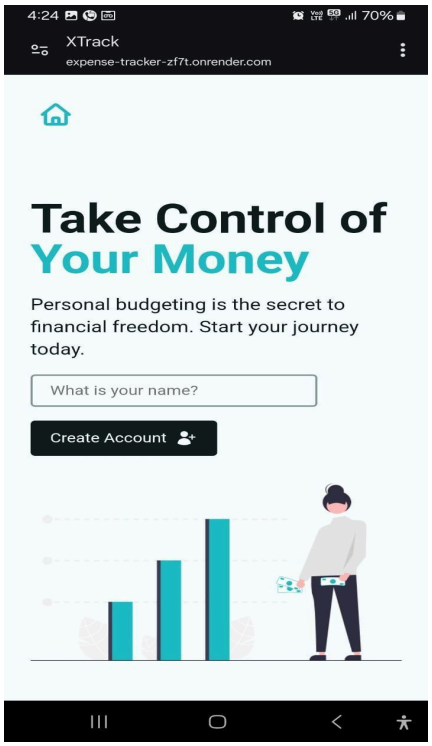
3. Then click on install:



4. Once installed you will see application in your mobile:



5. Final Output:



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.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No. 8

Vaishnal Mali

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 dominate **Network Traffic**
  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.

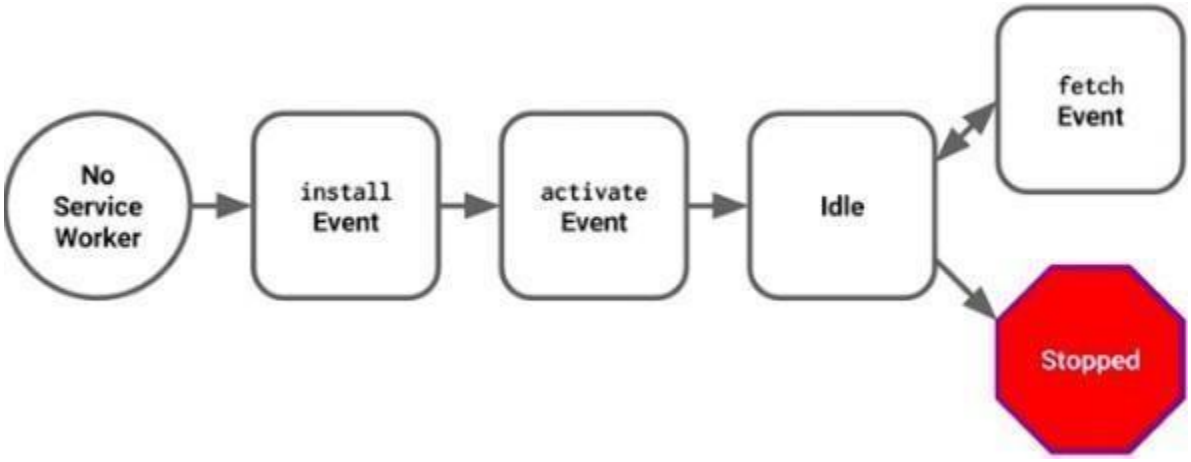
- You can manage **Push Notifications**
  You can manage push notifications with Service Worker and show any information message to the user.
- You can **Continue**
  Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

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.

Service Worker Cycle



A service worker goes through three steps in its life cycle:

- Registration
- Installation
- Activation

Registration
To install a service worker, you need to register it in your main JavaScript code.

Registration tells the browser where your service worker is located, and to start installing it in the background. Let's look at an example:

main.js

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('/
  service-worker.js')
  .then(function(registration) {
    console.log('Registration successful, scope is:', registration.scope);
  })
  .catch(function(error) {
    console.log('Service worker registration failed, error:', error);
  })
```

This code starts by checking for browser support by examining **navigator.serviceWorker**. The service worker is then registered with navigator.serviceWorker.register, which returns a promise that resolves when the service worker has been successfully registered. The scope of the service worker is then logged with registration.scope. If the service worker is already installed, navigator.serviceWorker.register returns the registration object of the currently active service worker.

The scope of the service worker determines which files the service worker controls, in other words, from which path the service worker will intercept requests. The default scope is the location of the service worker file, and extends to all directories below. So if service-worker.js is located in the root directory, the service worker will control requests from all files at this domain.

You can also set an arbitrary scope by passing in an additional parameter when registering. For

example: main.js

```
navigator.serviceWorker.register('/service-worker.js', {
  scope: '/app/'
});
```

In this case we are setting the scope of the service worker to /app/, which means the service worker will control requests from pages like /app/, /app/lower/ and /app/lower/lower, but not from pages like /app or /, which are higher.

If you want the service worker to control higher pages e.g. /app (without the trailing slash) you can indeed change the scope option, but you'll also need to set the Service-Worker-Allowed HTTP Header in your server config for the request serving the service worker script.

main.js

```
navigator.serviceWorker.register('/app/service-worker.js'
  , { scope: '/app'
})
```

Installation

Once the browser registers a service worker, installation can be attempted. This occurs if the service worker is considered to be new by the browser, either because the site currently doesn't have a registered service worker, or because there is a byte difference between the new service worker and the previously installed one.

A service worker installation triggers an install event in the installing service worker. We can include an install event listener in the service worker to perform some task when the service worker installs. For instance, during the install, service workers can precache parts of a web app so that it loads instantly the next time a user opens it (see caching the application shell). So, after that first load, you're going to benefit from instant repeat loads and your time to interactivity is going to be even better in those cases. An example of an installation event listener looks like this:

service-worker.js

```
// Listen for install event, set callback
self.addEventListener('install', function(event) {
  // Perform some task
});
```

Activation

Once a service worker has successfully installed, it transitions into the activation stage. If there are any open pages controlled by the previous service worker, the new service worker enters a waiting state. The new service worker only activates when there are no longer any pages loaded that are still using the old service worker. This ensures that only one version of the service worker is running at any given time.

When the new service worker activates, an activate event is triggered in the activating service worker. This event listener is a good place to clean up outdated caches (see the Offline Cookbook for an example).

service-worker.js

```
self.addEventListener('activate', function(event) {
 // Perform some task
});
```

Once activated, the service worker controls all pages that load within its scope, and starts listening for events from those pages. However, pages in your app that were loaded before the service worker activation will not be under service worker control. The new service worker will only take over when you close and reopen your app, or if the service worker calls **clients.claim()**. Until then, requests from this page will not be intercepted by the new service worker. This is intentional as a way to ensure consistency in your site.

Code

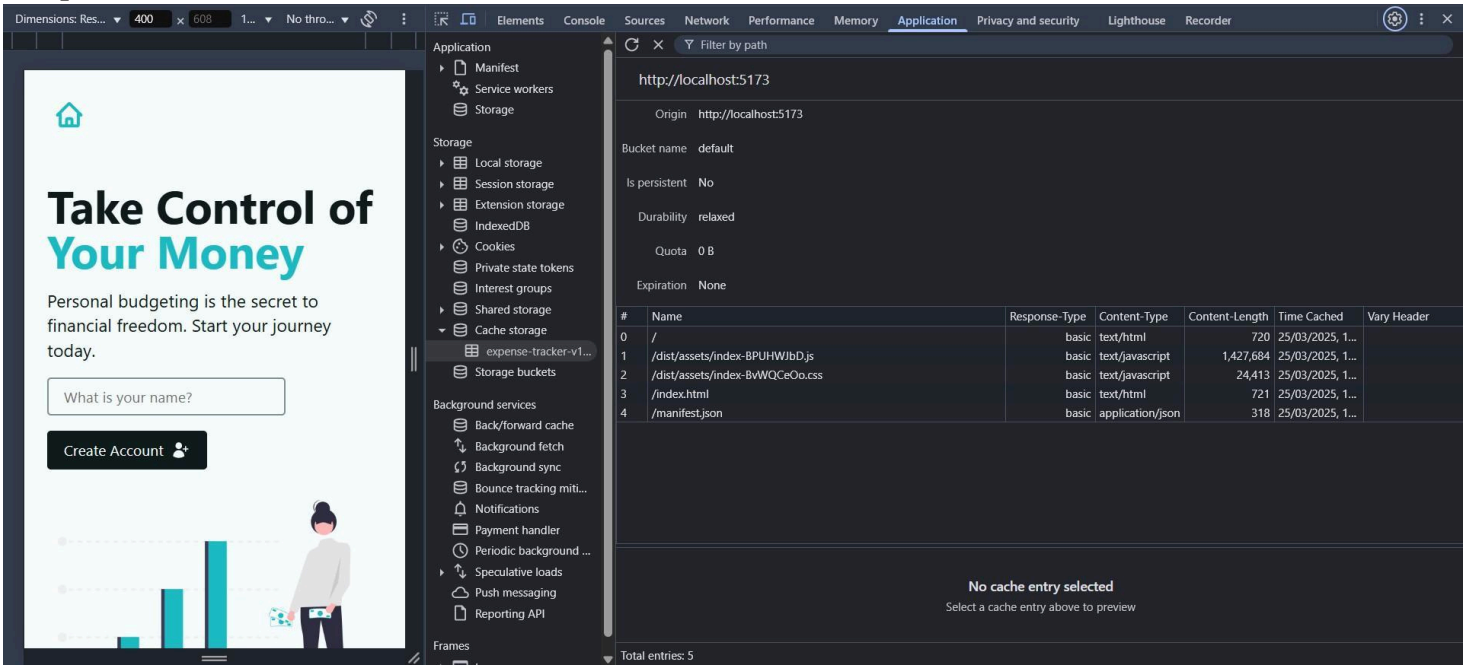
sw.js

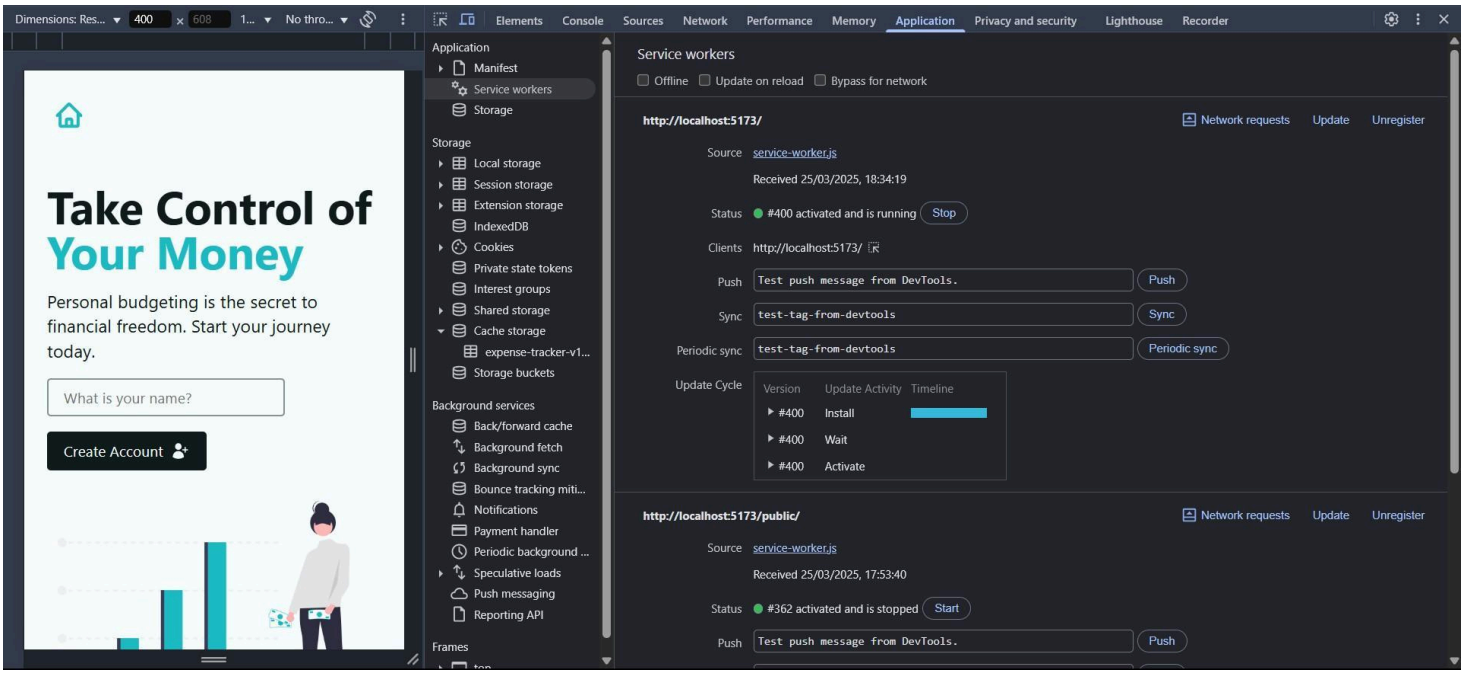
```
self.addEventListener("install", function (event)
   { event.waitUntil(preLoad());
});

var filesToCache = [
   '/',
   '/menu',
   '/contactUs',
   '/offline.html',
];

var preLoad = function () {
   return caches.open("offline").then(function (cache) {
      // caching index and important routes
      return cache.addAll(filesToCache);
   });
};

self.addEventListener("fetch", function (event) {
   event.respondWith(checkResponse(event.request).catch(function
   () {
      return returnFromCache(event.request);
   }));
   event.waitUntil(addToCache(event.request));
});

var checkResponse = function (request) {
   return new Promise(function (fulfill, reject)
   {
      fetch(request).then(function (response)
         { if (response.status !== 404) {
```

```
            fulfill(response)
          } else {
            reject();
          }
        }, reject);
    });
};

var addToCache = function (request) {
    return caches.open("offline").then(function (cache)
        { return fetch(request).then(function (response)
        {
          return cache.put(request, response);
        });
    });
};

var returnFromCache = function (request) {
    return caches.open("offline").then(function (cache) {
        return cache.match(request).then(function
        (matching) {
          if (!matching || matching.status == 404)
            { return cache.match("offline.html");
          } else {
            return matching;
            }
          });
    });
}
```

Project Title: Zomato **Roll No. 26**

Output:





Sample Code with Output

Index.html

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/favicon.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="manifest" href="manifest.json" />
    <title>XTrack</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

main.jsx

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import App from './App.jsx'
import './index.css'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>,
)

// Service Worker Registration
if ('serviceWorker' in navigator) {
  window.addEventListener('load', () => {
    navigator.serviceWorker.register('public/service-worker.js') // Path to your service worker
file
      .then((registration) => {
        console.log('[Service Worker] Registered with scope:', registration.scope);

        // Check if the service worker is active
        if (registration.active) {
          console.log('[Service Worker] Active');
```

```
        }

        // Check if the service worker is installing
        if (registration.installing) {
          console.log('[Service Worker] Installing');
        }

        // Check if the service worker is waiting
        if (registration.waiting) {
          console.log('[Service Worker] Waiting');
        }

        // Listen for state changes
        registration.addEventListener('updatefound', () => {
          console.log('[Service Worker] Update found');
          const installingWorker = registration.installing;
          if (installingWorker) {
            installingWorker.addEventListener('statechange', () => {
              if (installingWorker.state === 'installed') {
                if (navigator.serviceWorker.controller) {
                  console.log('[Service Worker] New content is available and will be used when
all tabs for this page are closed.');
                  // You can prompt the user to reload here if needed
                } else {
                  console.log('[Service Worker] Content is cached for offline use.');
                }
              }
            });
          }
        });
      })
                                                .catch((error) => {
                                                  console.error('[Service Worker] Registration
                                                  failed:', error);
                                                });
  });

} else {
  console.warn('[Service Worker] Not supported in this browser.');
}
```

service-worker.js

```
// public/service-worker.js

const CACHE_NAME = 'expense-tracker-v1'; // Change this when you update the service worker
const urlsToCache = [
  '/', // Cache the root URL (index.html)
  '/index.html',
  '/manifest.json', // if you have one
  // Correct paths to your built assets
  'dist/assets/index-BPUHWJbD.js', // Replace with the actual path
  'dist/assets/index-BvWQCeOo.css',
];

// Install Event: Cache static assets
self.addEventListener('install', (event) => {
  console.log('[Service Worker] Install');
  event.waitUntil(
    caches.open(CACHE_NAME)
      .then((cache) => {
        console.log('[Service Worker] Caching app shell');
        return cache.addAll(urlsToCache);
      })
      .catch((error) => {
        console.error('[Service Worker] Caching failed:', error);
      })
  );
});

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

```
  );
});

// Fetch Event: Serve from cache or network
self.addEventListener('fetch', (event) => {
  console.log('[Service Worker] Fetch', event.request.url);
  event.respondWith(
    caches.match(event.request)
      .then((response) => {
        // Cache hit - return response
        if (response) {
          console.log('[Service Worker] Found in cache:', event.request.url);
          return response;
        }
        // Not in cache - return fetch request
        console.log('[Service Worker] Not found in cache, fetching:', event.request.url);
        return fetch(event.request);
      })
  );
});
```

Conclusion: Successfully installed and activated the process of new service worker for pwa.

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.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No. 9

Vaishnal Mali
D15A - 26

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".

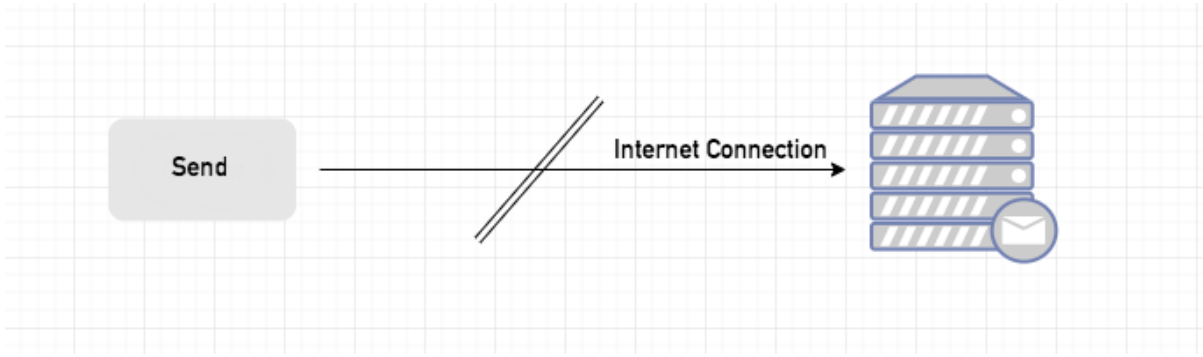
- **CacheFirst** - In this function, if the received request has cached before, the cached response is returned to the page. But if not, a new response requested from the network.
- **NetworkFirst** - In this function, firstly we can try getting an updated response from the network, if this process completed 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.

```
self.addEventListener("fetch", function (event) {
    const req = event.request;
    const url = new URL(req.url);

    if (url.origin === location.origin) {
        event.respondWith(cacheFirst(req));
    }
    else {
        event.respondWith(networkFirst(req));
    }
});

async function cacheFirst(req) {
    return await caches.match(req) || fetch(req);
}

async function networkFirst(req) {
    const cache = await caches.open("pwa-dynamic");
    try {
        const res = await fetch(req);
        cache.put(req, res.clone());
        return res;
    } catch (error) {
        const cachedResponse = await cache.match(req);
        return cachedResponse || await caches.match("./noconnection.json");
    }
}
```

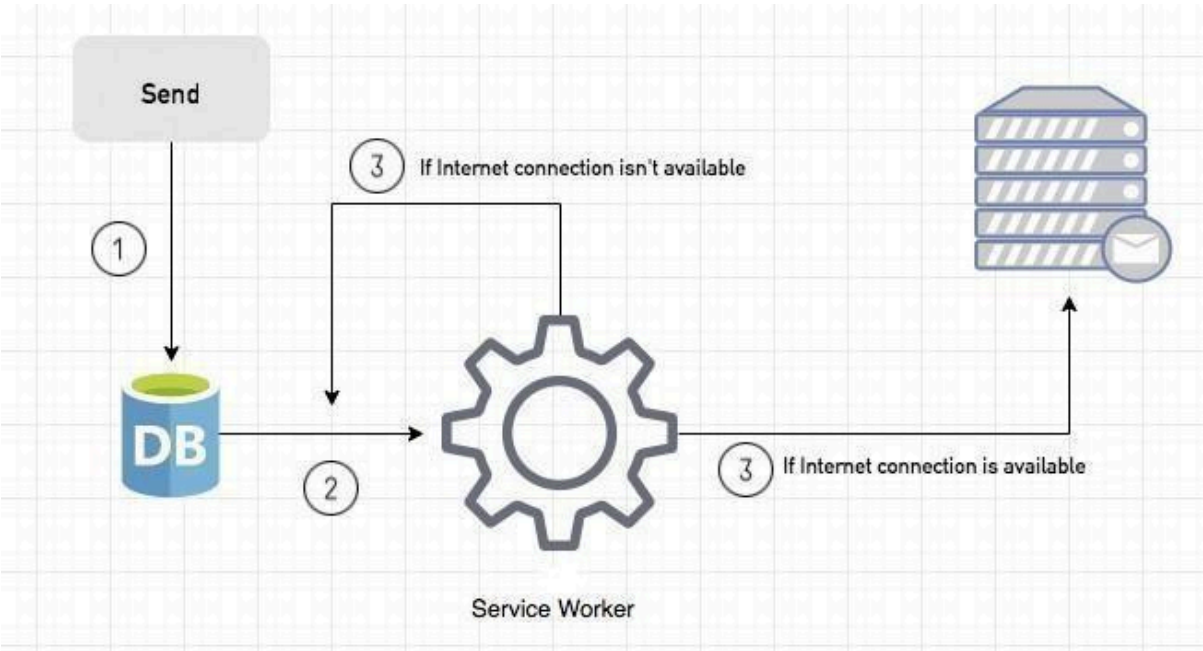
Sync Event

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.

Here is a job for the Background Sync.
The following view shows the classical process of sending email to us. If the Internet Connection is broken, we can't send any content to Mail Server.



Here, you can create any scenario for yourself. A sample is in the following for this case.



1. When we click the “send” button, email content will be saved to IndexedDB.
2. Background Sync registration.
3. **If the Internet connection is available**, all email content will be read and sent to Mail Server.

   If the Internet connection is unavailable, the service worker waits until the connection is available even though the window is closed. When it is available, email content will be sent to Mail Server.

   You can see the working process within the following code block.

Event Listener for Background Sync Registration

```
document.querySelector("button").addEventListener("click", async () => {
    var swRegistration = await navigator.serviceWorker.register("sw.js");
    swRegistration.sync.register("helloSync").then(function () {
        console.log("helloSync success [main.js]");
    });
});
```

Event Listener for sw.js

```
self.addEventListener('sync', event => {
    if (event.tag == 'helloSync') {
        console.log("helloSync [sw.js]");
    }
});
```

Push Event

This is the event that handles push notifications that are received from the server. You can apply any method with received data.

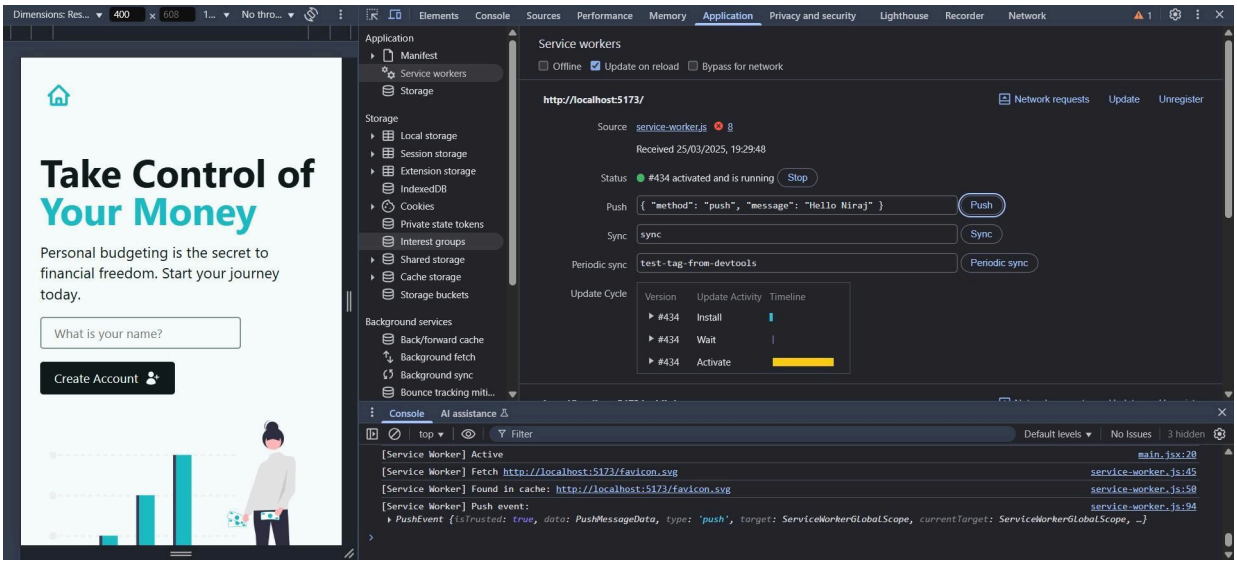
We can check in the following example.
“Notification.requestPermission();” is the necessary line to show notification to the user. If you don’t want to show any notification, you don’t need this line.

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.

```
self.addEventListener('push', event => {
    if (event && event.data) {
        var data = event.data.json();
        if (data.method === "pushMessage") {
            event.waitUntil(self.registration.showNotification("Test App", {
                body: data.message
            }));
        }
    }
});
```

You can use Application Tab from Chrome Developer Tools for testing push notification.



Code:

service-worker.js

```
// public/service-worker.js


const CACHE_NAME = 'expense-tracker-v1'; // Change this when you update the service
worker
const urlsToCache = [
  '/', // Cache the root URL (index.html)
  '/index.html',
  '/manifest.json', // if you have one
  // Correct paths to your built assets
  'dist/assets/index-BPUHWJbD.js', // Replace with the actual path
  'dist/assets/index-BvWQCeOo.css',
];


self.addEventListener('install', (event) => {
  console.log('[Service Worker] Install');
  event.waitUntil(
    caches.open(CACHE_NAME)
      .then((cache) => {
        console.log('[Service Worker] Caching app shell');
        return cache.addAll(urlsToCache);
      })
      .catch((error) => {
        console.error('[Service Worker] Caching failed:', error);
      })
  );
});


self.addEventListener('activate', (event) => {
  console.log('[Service Worker] Activate');
  event.waitUntil(
    caches.keys().then((cacheNames) => {
```

```
      return Promise.all(
        cacheNames.map((cacheName) => {
          if (cacheName !== CACHE_NAME) {
            console.log('[Service Worker] Deleting old cache:', cacheName);
            return caches.delete(cacheName);
          }
        })
      );
    })
  );
});

// Fetch Event: Cache-first strategy
self.addEventListener('fetch', (event) => {
  console.log('[Service Worker] Fetch', event.request.url);
  event.respondWith(
    caches.match(event.request)
      .then((response) => {
        if (response) {
          console.log('[Service Worker] Found in cache:', event.request.url);
          return response;
        }
        return fetch(event.request)
          .then((response) => {
            if (!response || response.status !== 200 || response.type !== 'basic') {
              return response;
            }
            const responseToCache = response.clone();
            caches.open(CACHE_NAME)
              .then((cache) => {
                cache.put(event.request, responseToCache);
              });
            return response;
```

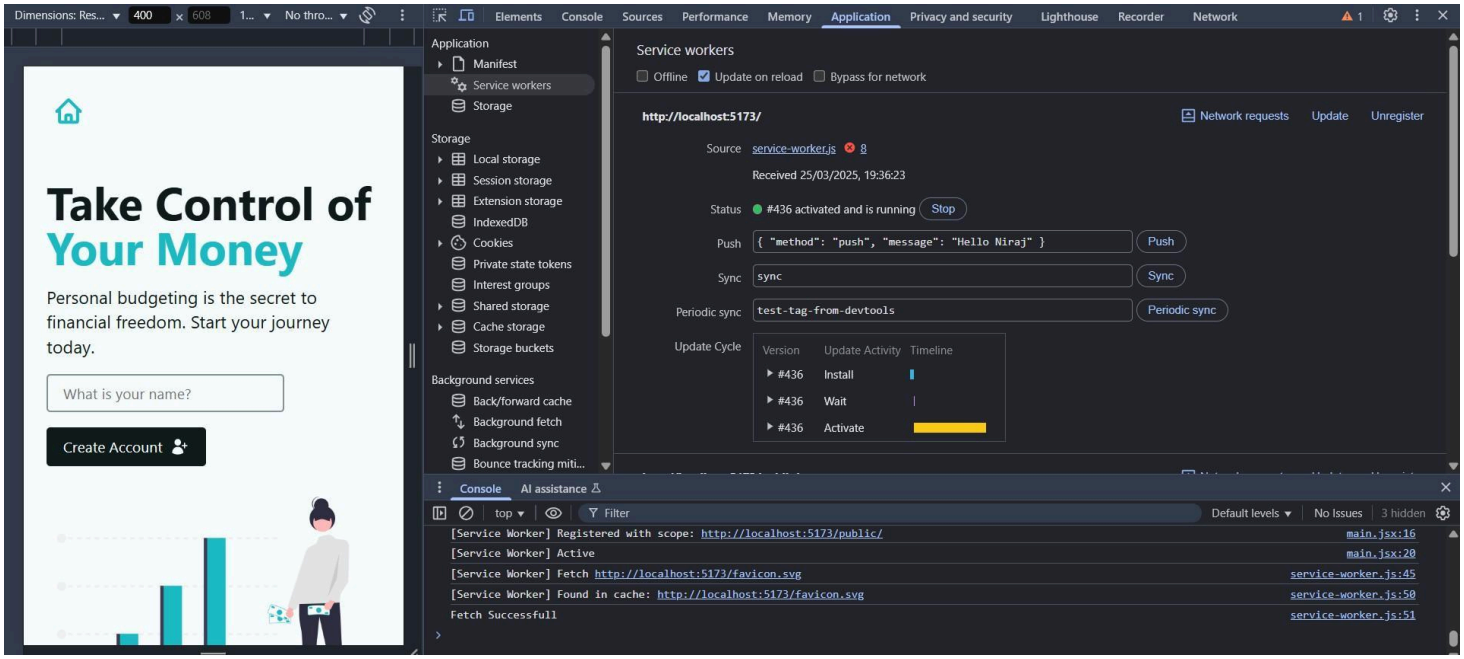
```
                                                    })
                                                    .catch((error) => {
                                                      console.error('[Service Worker] Fetch
                                                      failed:', error);
                                                      // Handle network errors (e.g., show
                                                      offline message)
                                                    });
);                                                })
});



// Sync Event: Handle background sync
self.addEventListener('sync', (event) => {
  console.log('[Service Worker] Sync event:', event.tag);
  if (event.tag === 'sync-expenses') {
    event.waitUntil(
      // Your background sync logic here (e.g., send expenses to server)
      syncExpenses()
        .then(() => {
          console.log('[Service Worker] Expenses synced successfully');
        })
        .catch((error) => {
          console.error('[Service Worker] Expense sync failed:', error);
          // Handle sync errors (e.g., retry later)
        })
    );
}
});



// Push Event: Handle push notifications
self.addEventListener('push', function(event) {
  console.log('[Service Worker] Push event:', event);
  let data;
  try {
```

```
    data = event.data ? event.data.json() : {};
  } catch (error) {
    console.error('Error parsing push notification data:', error);
    data = { title: 'Error', message: 'Could not parse notification data.' };
  }

  if (Notification.permission === 'granted') {
    event.waitUntil(
      self.registration.showNotification(data.title || 'Xtrack',
        { body: data.message || 'No message provided.',
        icon: data.icon || '/your-notification-icon.png', // Provide a default icon
      })
    );
  } else {
    console.log('Notification permission not granted. Notification will not be
shown.');
  }
});



//Helper function for sync event
async function syncExpenses() {
  // Replace this with your actual expense syncing logic
  // This example just simulates a delay
  return new Promise((resolve) => {
    setTimeout(resolve, 2000); // Simulate a 2-second delay
  });
}
```

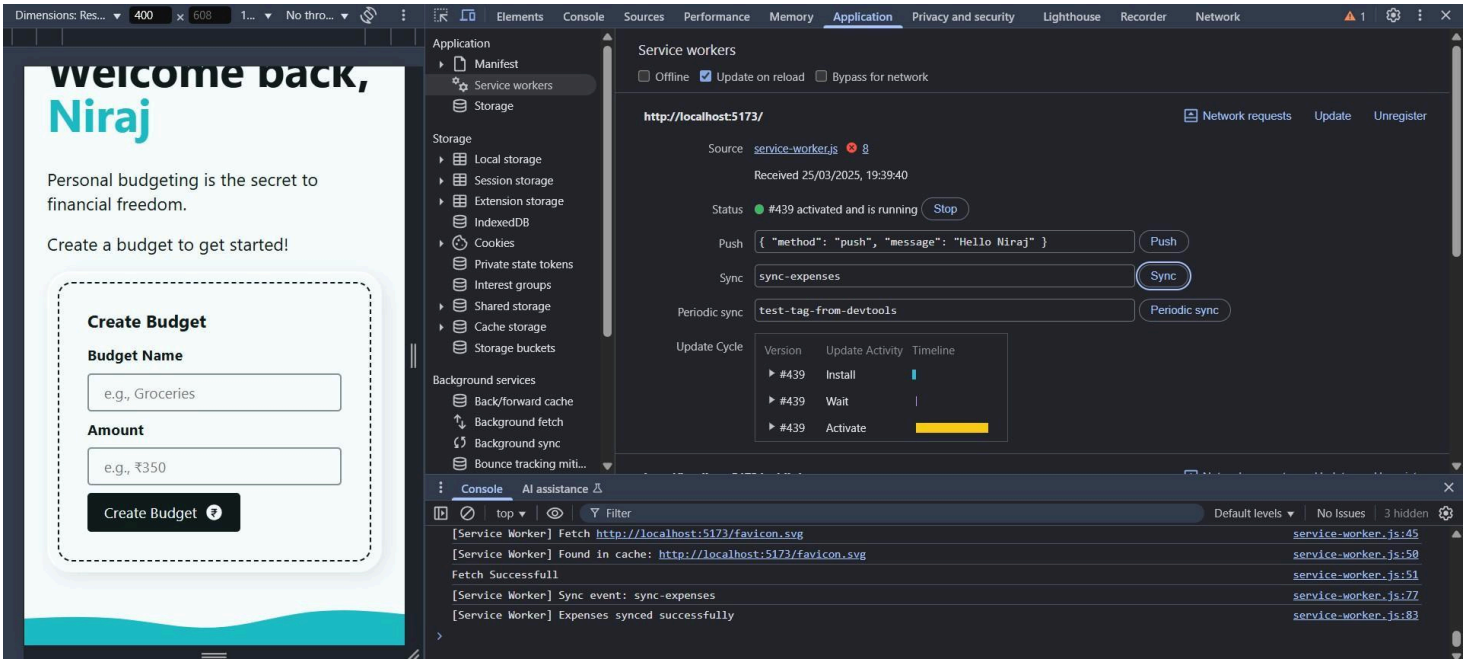
Output:

Project Title: Zomato **Roll No. 26**

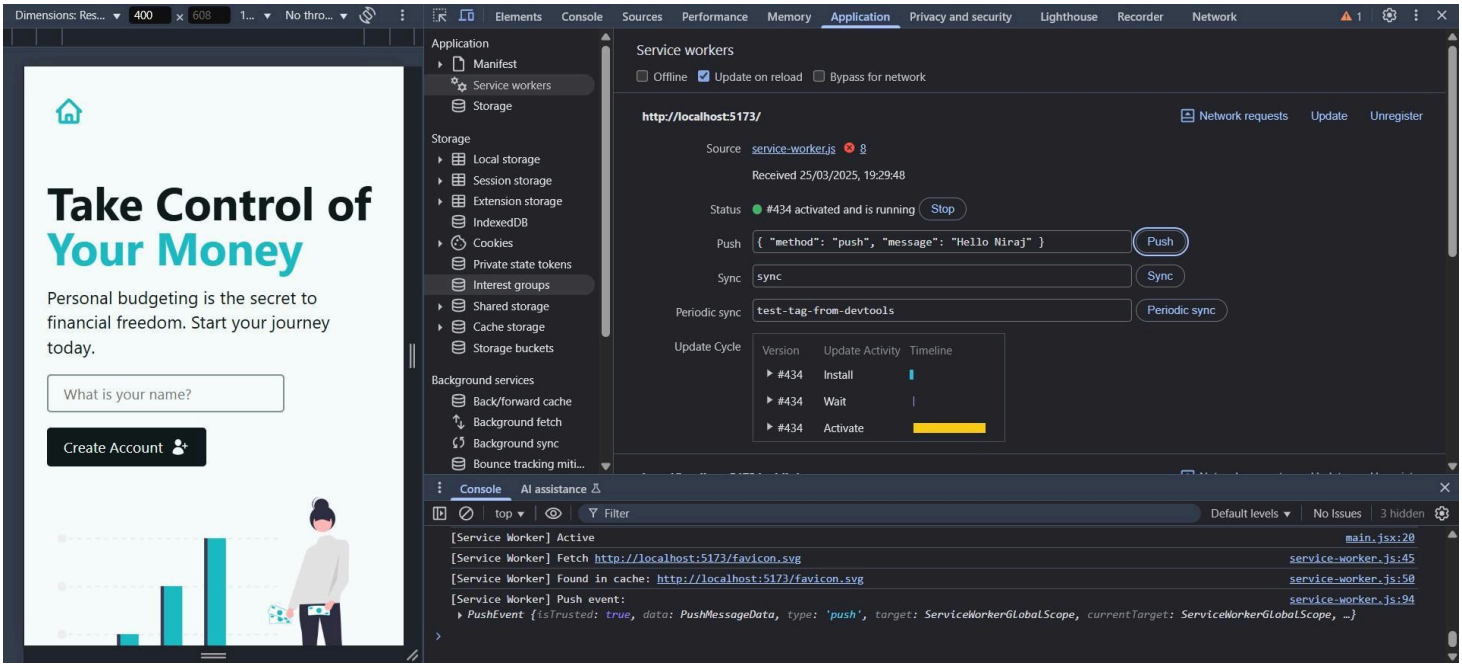
Fetchevent



Sync event



Push event



Conclusion: Successfully implemented Service worker events like fetch, sync and push for PWA.

MAD & PWA Lab
Journal

Experiment No.	10
Experiment Title.	To study and implement deployment of Ecommerce PWA to GitHub Pages.
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment
No. 10 MAD
& PWA Lab

Vaishnal Mali
D15A/26

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

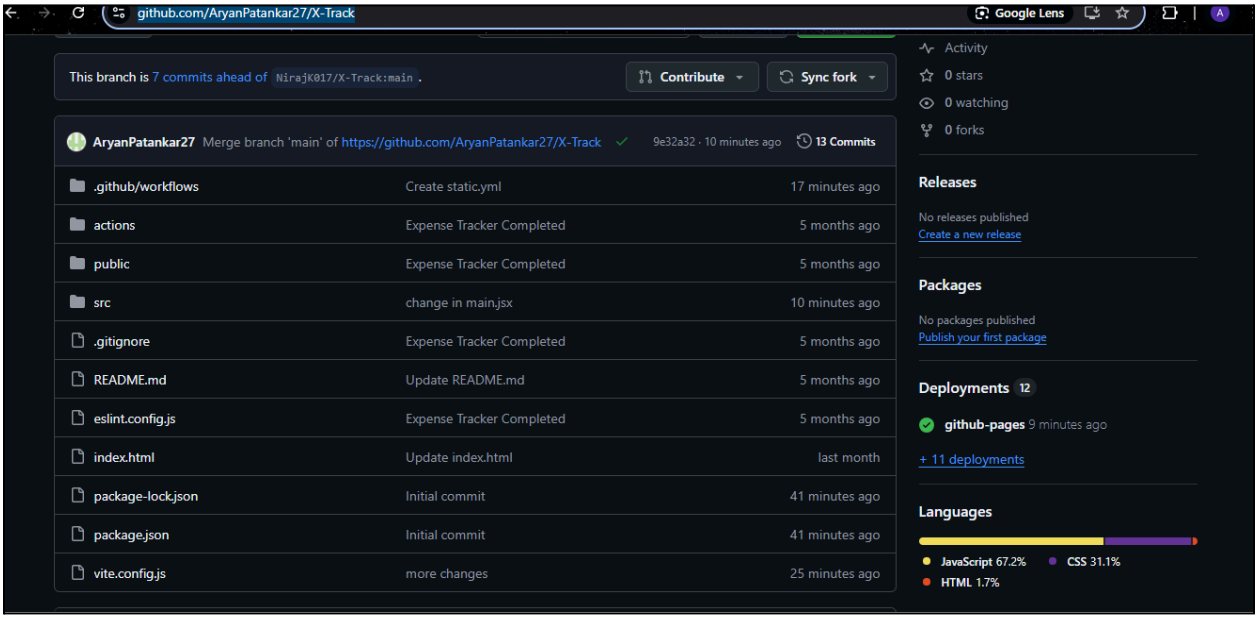
1. Hosted by Google. Enough said.
2. Authentication, Cloud Messaging, and a whole lot of other handy services will be available to you.
3. A real-time database will be available to you, which can store 1 GB of data.
4. You'll also have access to a blob store, which can store another 1 GB of data.
5.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

Cons

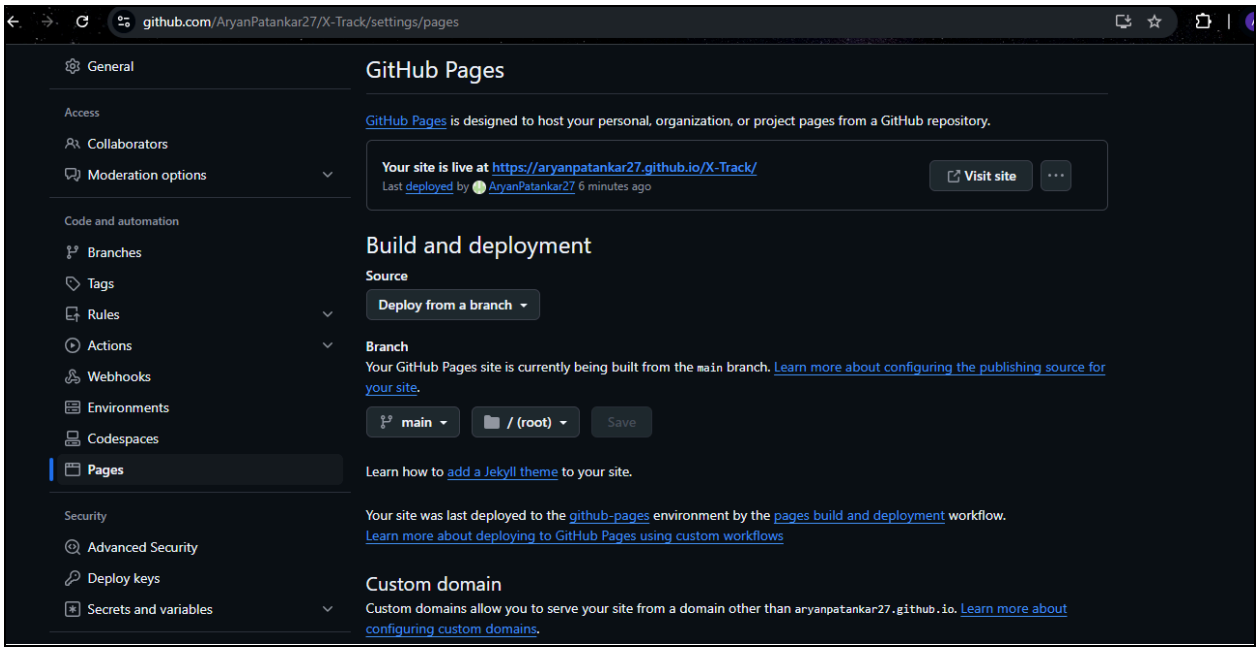
1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.
3. No in-built support for any static site generator.

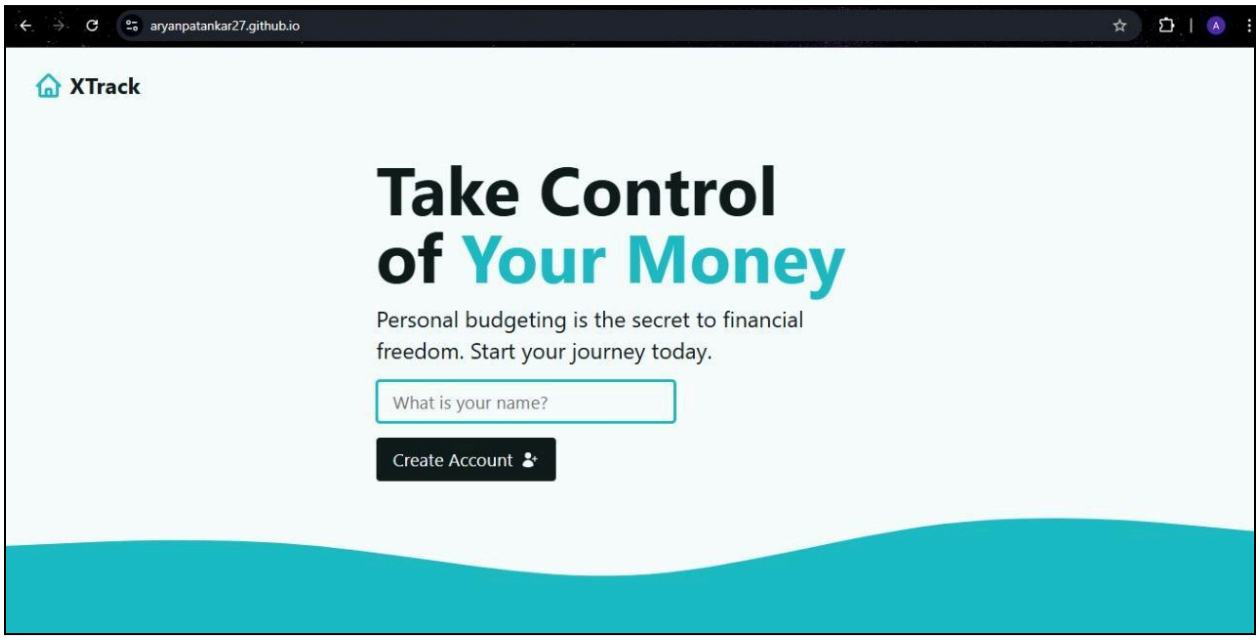
Link to our GitHub repository: https://github.com/AryanPatankar27/X-Track

Github Screenshot:



Project Title: Zomato **Roll No. 26**





MAD & PWA Lab
Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	26
Name	Vaishnal Dilip Mali
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	

Experiment 11

Vaishnal Mali
D15A - 26

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: This score is an aggregation of how the page fared in aspects such as (but not limited to) loading speed, time taken for loading for basic frame(s), displaying meaningful content to the user, etc. To a layman, this score is indicative of how decently the site performs, with a score of 100 meaning that you figure in the 98th percentile, 50 meaning that you figure in the 75th percentile and so on.

PWA Score (Mobile): Thanks to the rise of Service Workers, app manifests, etc., a lot of modern web applications are moving towards the PWA paradigm, where the objective is to make the application behave as close as possible to native mobile applications. Scoring points are based on the Baseline PWA checklist laid down by Google which includes Service Worker implementation(s), viewport handling, offline functionality, performance in script-disabled environments, etc.

Accessibility: As you might have guessed, this metric is a measure of how accessible your website is, across a plethora of accessibility features that can be implemented in your page (such as the 'aria-' attributes like aria-required, audio captions, button names, etc.). Unlike the other metrics though, Accessibility metrics score on a pass/fail basis i.e. if all possible elements of the page are not screen-reader friendly (HTML5 introduced features that would make pages easy to interpret for screen readers used by visually challenged people like tag names, tags such as <section>, <article>, etc.), you get a 0 on that score. The aggregate of these scores is your Accessibility metric score.

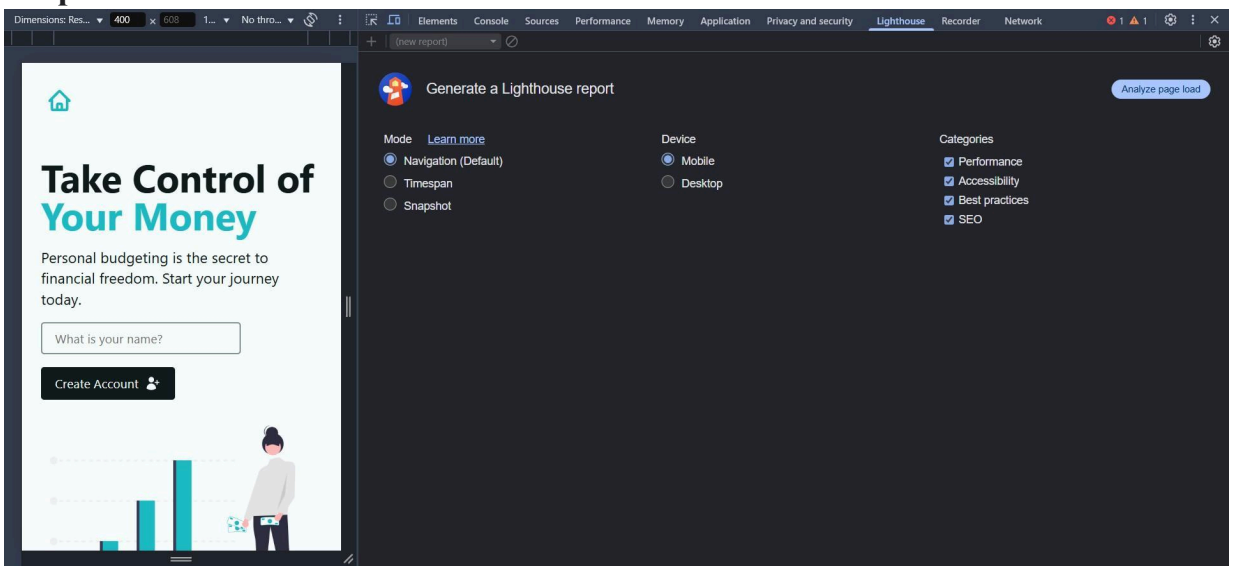
Best Practices: As any developer would know, there are a number of practices that have been deemed 'best' based on empirical data. This metric is an aggregation of many such points, including but not limited to:Use of HTTPS Avoiding the use of deprecated code elements like tags, directives, libraries, etc. Password input with paste-into disabled
Geo-Location and cookie usage alerts on load, etc.

Improvement In Code:

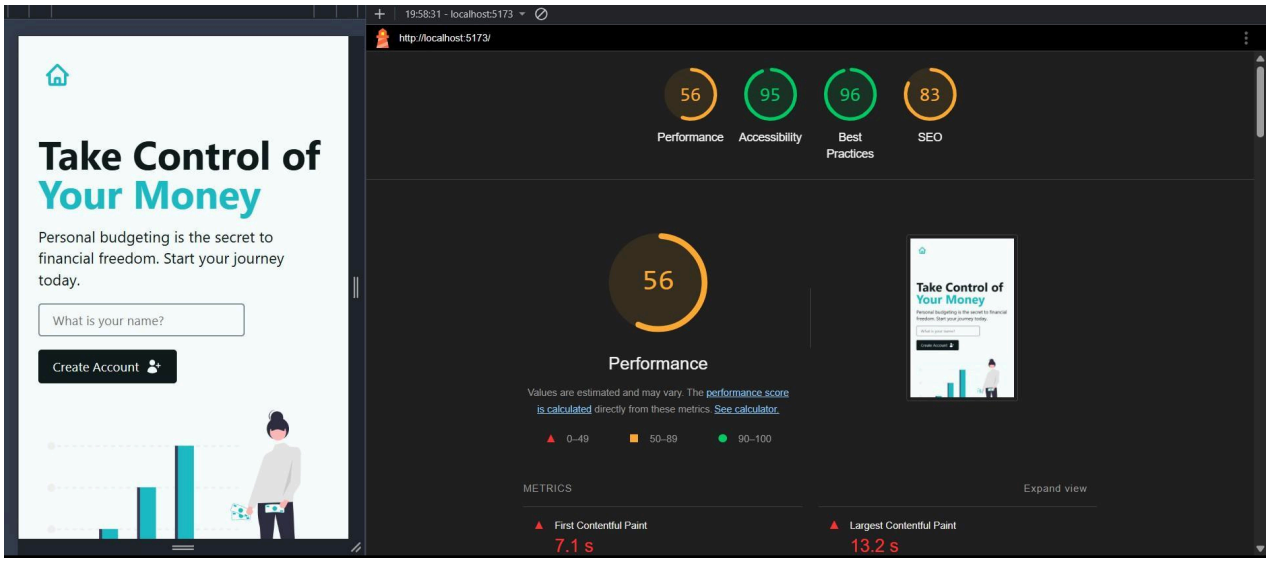
Index.html

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/favicon.svg" />
     <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
     <meta name="description" content="A concise and accurate description
of your Expense Tracker app. This will appear in search engine results.">
    <link rel="manifest" href="manifest.json" />
    <title>XTrack</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

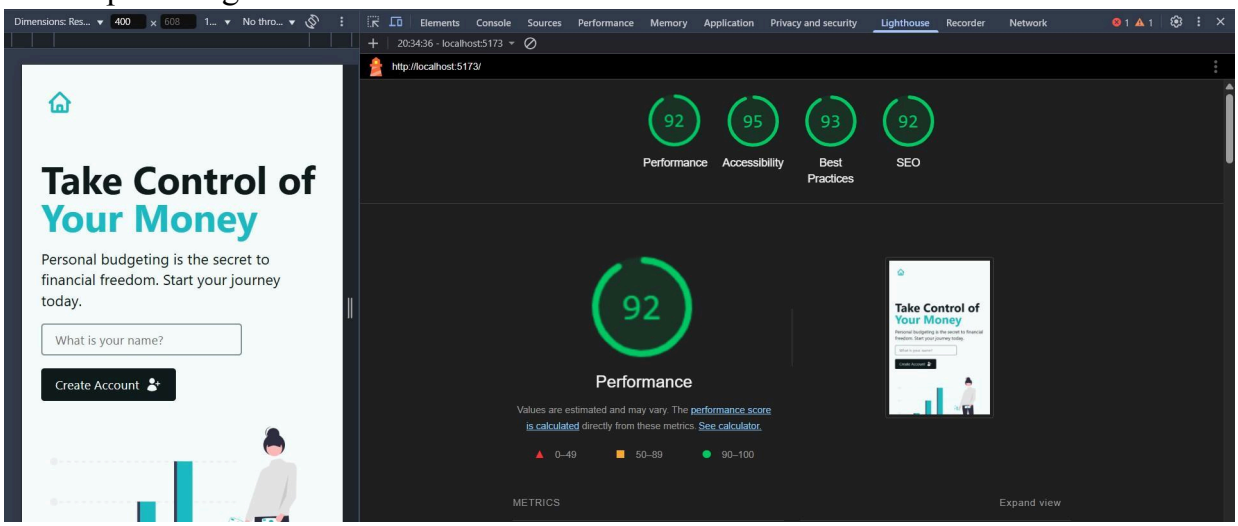
Output:



Before Optimizing Code:



After Optimizing Code:



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