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 Madhura Jangale of **D15A** semester **VI,** have successfully completed necessary experiments in the **MAD & PWA Lab** under my supervision in **VES Institute of Technology** during the academic year **2024-2025.**

Lab Assistant

Subject Teacher

Mrs. Kajal Joseph

Principal

Head of Department

Dr. Mrs. Shalu Chopra

Name of the Course : MAD & PWA Lab

Course Code : ITL604

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

Faculty Incharge : Mrs. Kajal Joseph.

Lab Teachers **:** Mrs. Kajal Joseph.

Email **:** kajal.jewani@ves.ac.in

Programme Outcomes: The graduate will be able to:

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

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

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

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

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

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

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

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

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

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

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

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

Program specific Outcomes

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

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

Lab Objectives:

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

Lab Outcomes:

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

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

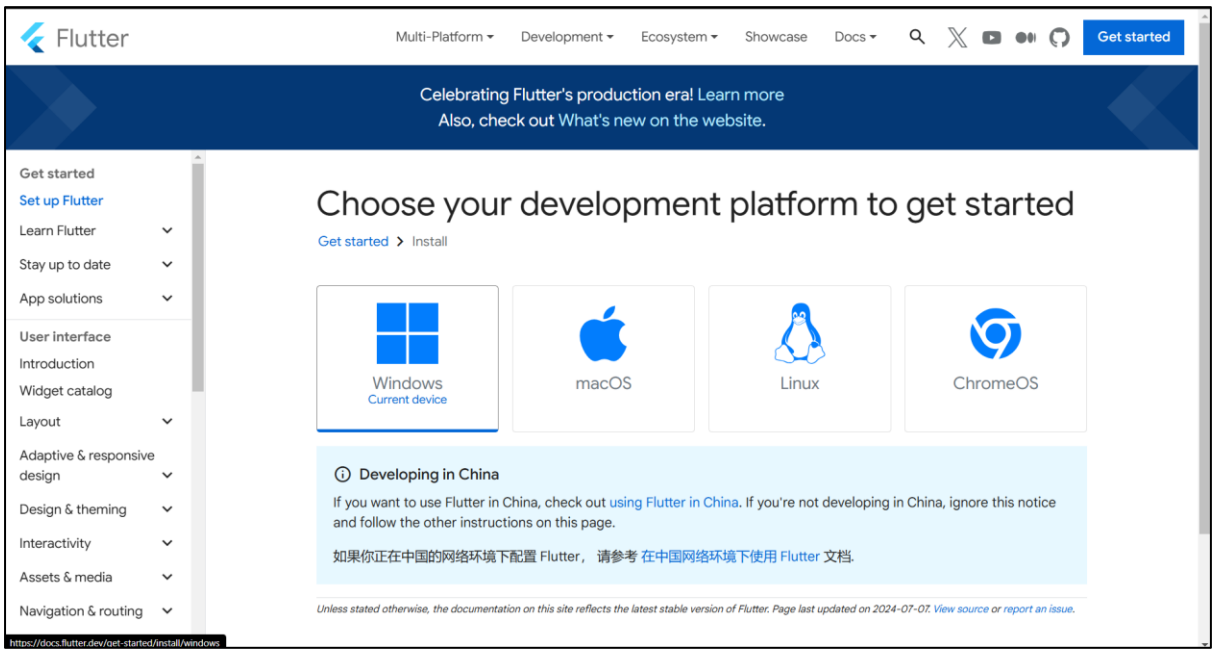
Journal

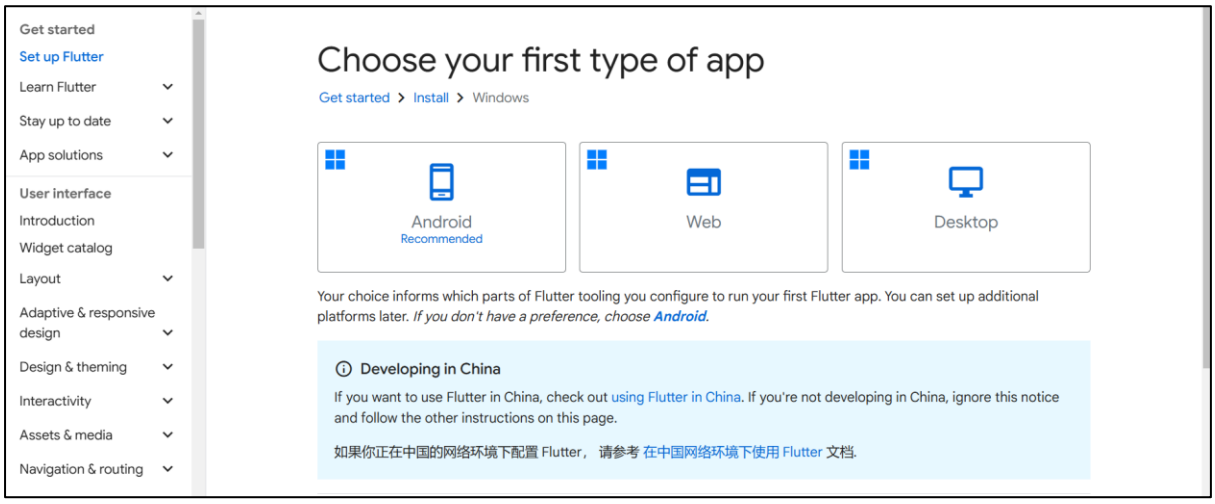
Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	

Aim: Installation and Configuration of Flutter Environment.

Install the Flutter SDK
Step 1: Download the installation bundle of the Flutter Software Development Kit for windows.
To download Flutter SDK, Go to its official website https://docs.flutter.dev/get-started/install ,

you will get the following screen.



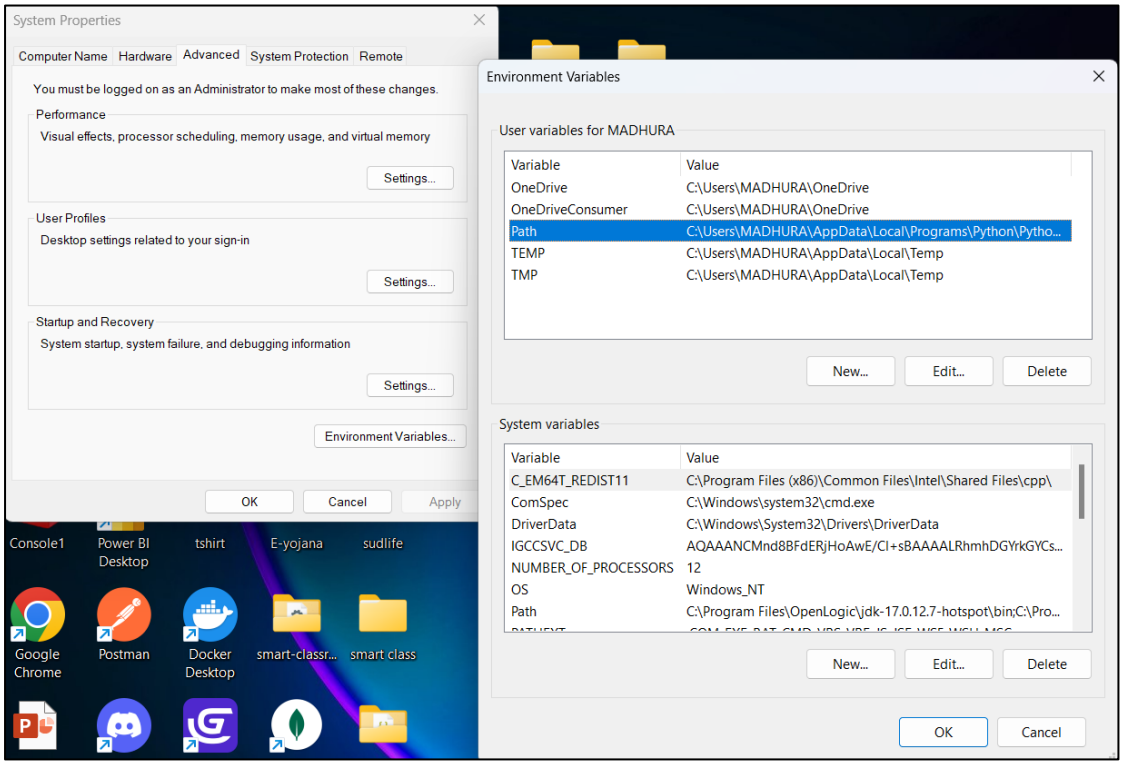


Step 2: Next, to download the latest Flutter SDK, click on the Windows icon. Here, you will find the download link for SDK.

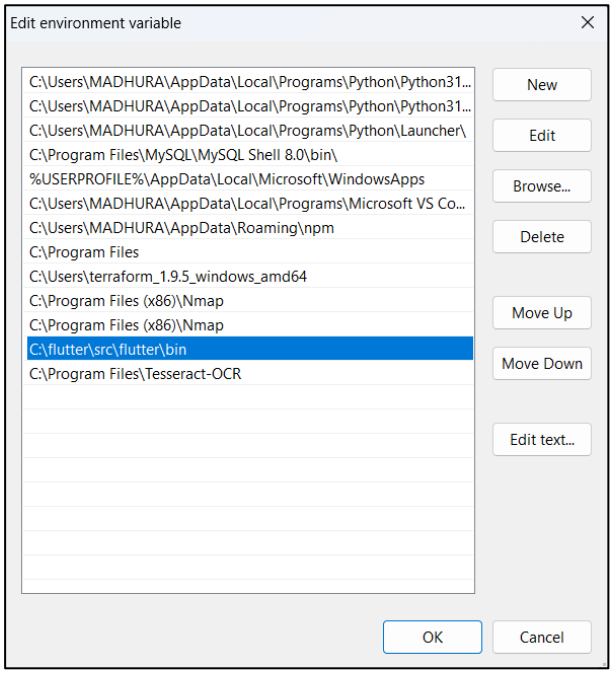
Step 3: When your download is complete, extract the zip file and place it in the desired installation folder or location, for example, C: /Flutter.

Step 4: To run the Flutter command in regular windows console, you need to update the system path to include the flutter bin directory. The following steps are required to do this:

- Go to MyComputer properties -> advanced tab -> environment variables. You will get the following screen.

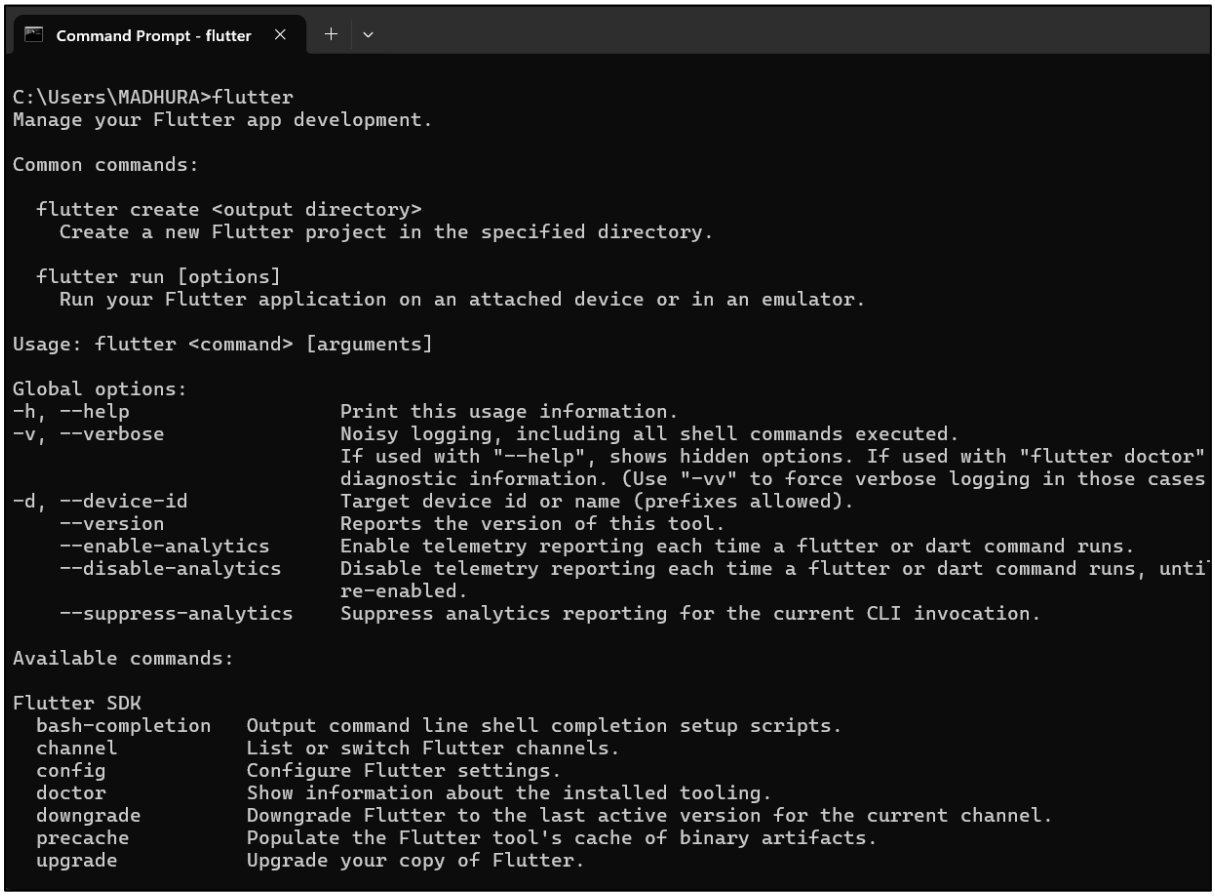


Now, select path -> click on edit. The following screen appears



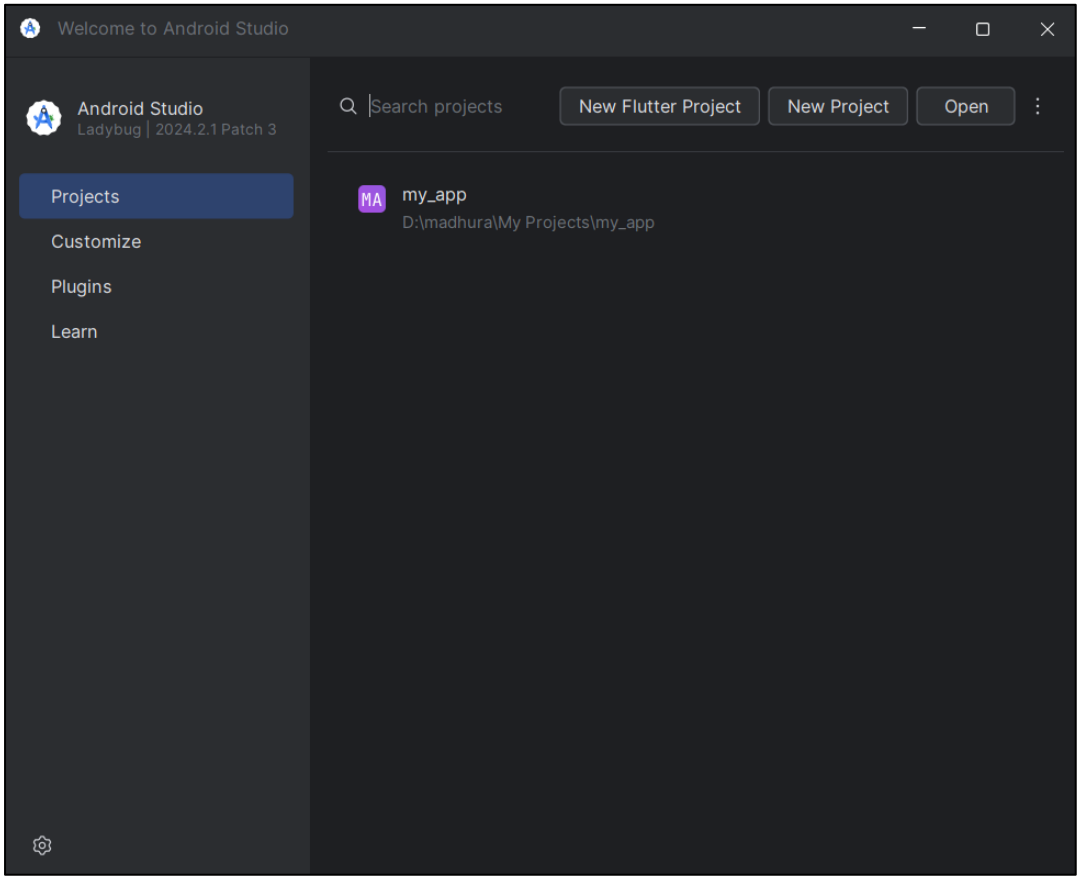
In the above window, click on New->write path of Flutter bin folder in variable value -> ok -> ok -> ok.

Step 5: Now, run the $ flutter command in command prompt.



Flutter is installed successfully.

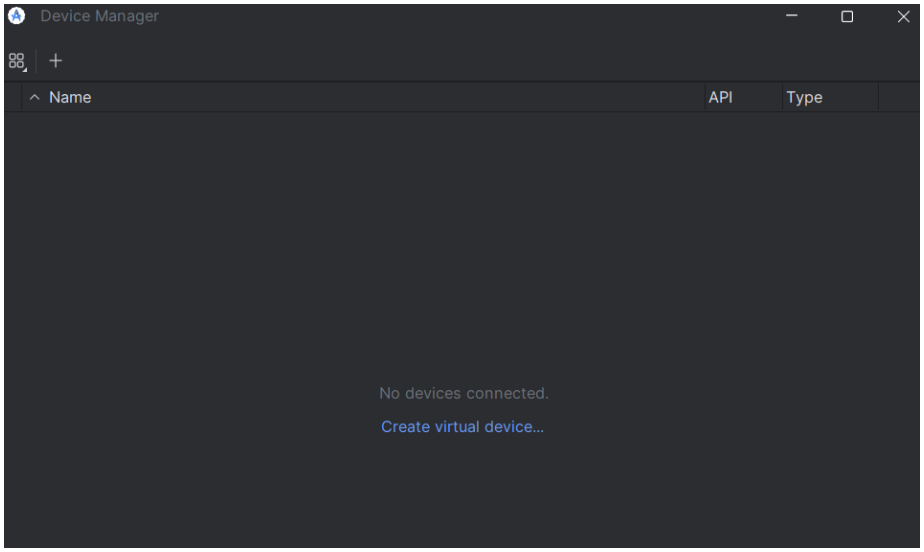
Step 6: Install the Android SDK. If the flutter doctor command does not find the Android SDK tool in your system, then you need first to install the Android Studio IDE.



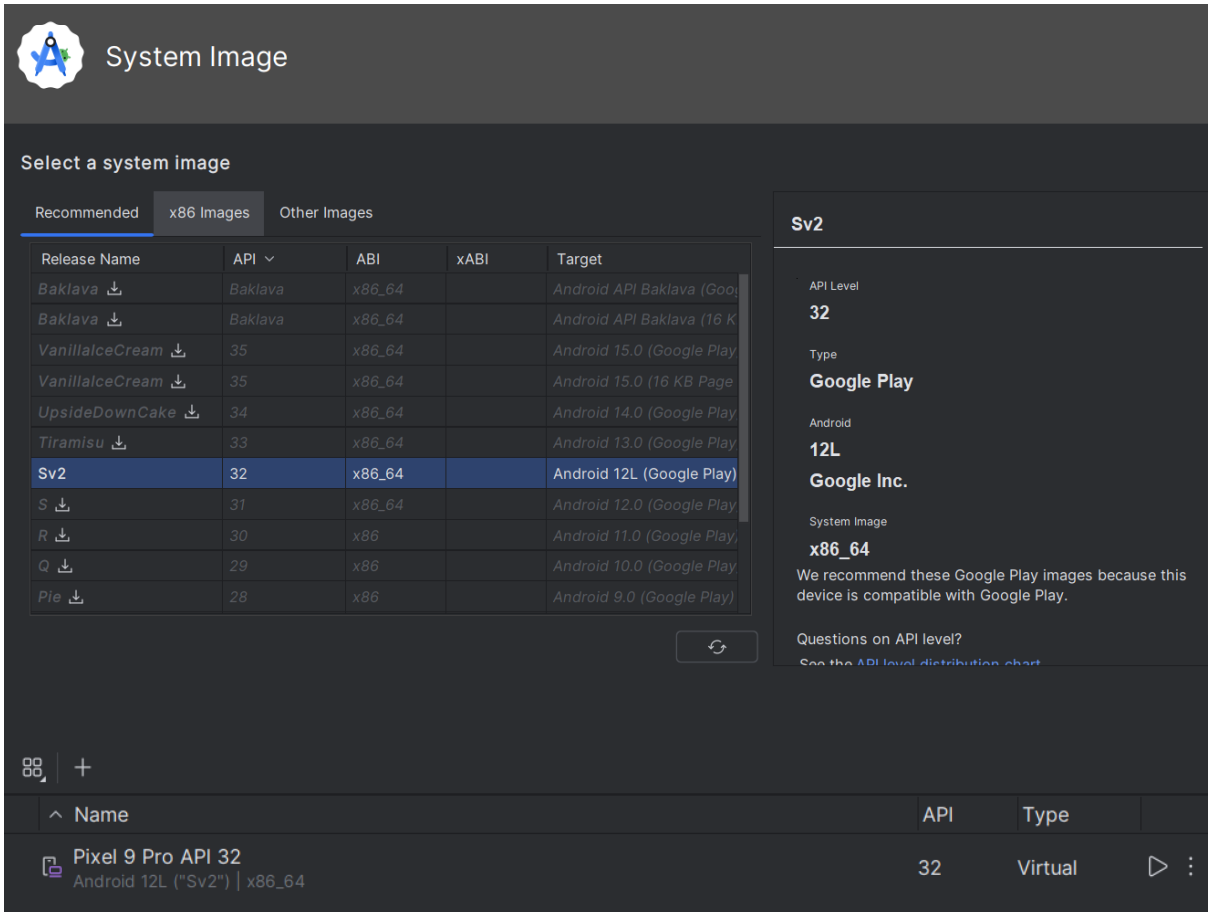
Step 7: Run Flutter Doctor command

```
C:\Users\MADHURA>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.27.1, on Microsoft Windows [Version 10.0.22631.4602], locale en-IN)
[√] Windows Version (Installed version of Windows is version 10 or higher)
[√] Android toolchain - develop for Android devices (Android SDK version 35.0.0)
[√] Chrome - develop for the web
[√] Visual Studio - develop Windows apps (Visual Studio Build Tools 2022 17.11.0)
[√] Android Studio (version 2024.2)
[√] VS Code (version 1.96.4)
[√] Connected device (3 available)
[√] Network resources

• No issues found!
```

Step 8: Create a new virtual device

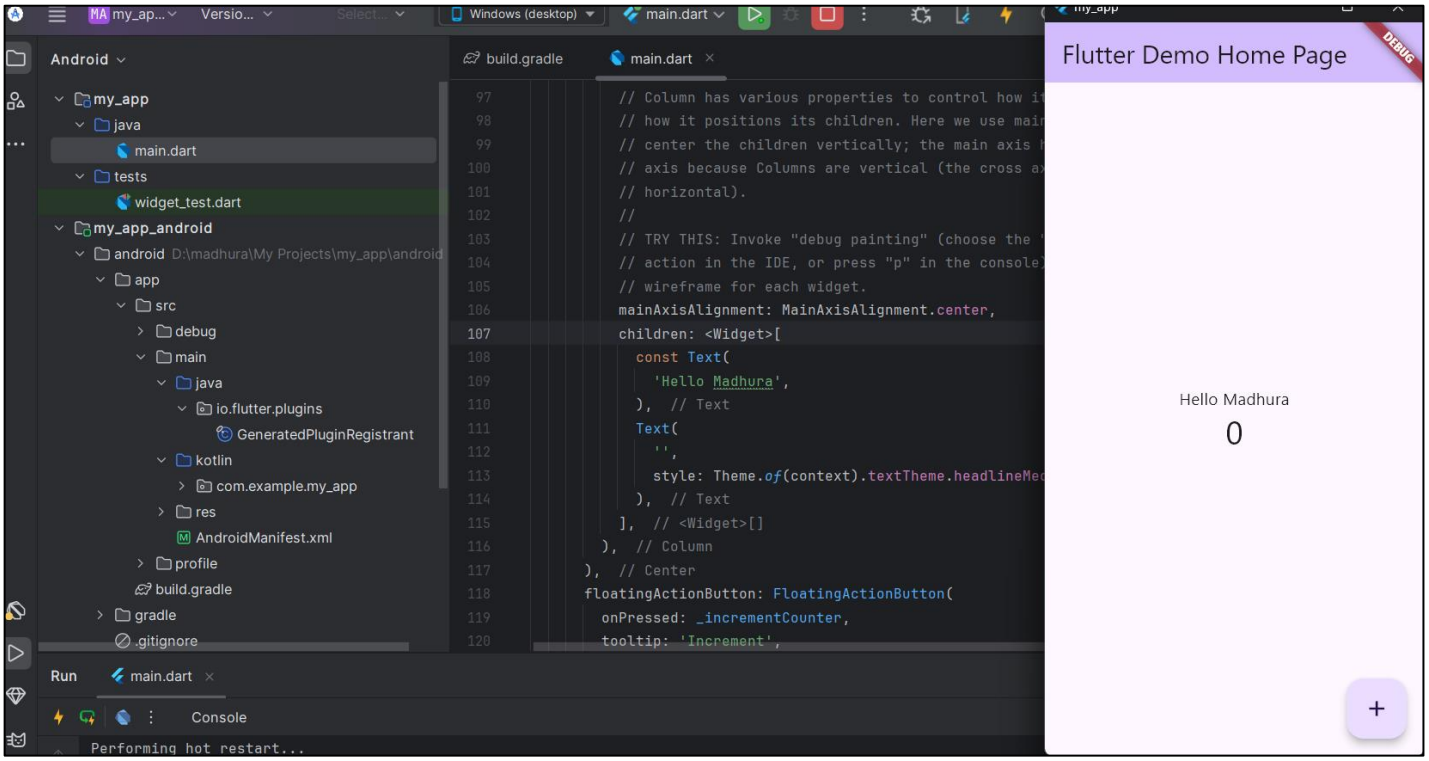


Choose your device definition and click on Next.
Select the system image for the latest Android version and click on Next.
Now, verify the all AVD configuration. If it is correct, click on Finish. The following screen appears.



Virtual device created.

Creating a hello world program in flutter



MAD & PWA Lab
Journal

Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

Aim: To design Flutter UI by including common widgets.

Theory:

Flutter follows a widget-based approach where everything in the UI is a widget. Widgets can be classified into two main types:

- Stateless Widgets: Do not change their state once built (e.g., Text, Container).
- Stateful Widgets: Can update dynamically based on user interaction (e.g., TextField, Checkbox).

Commonly Used Widgets in Flutter-

(a) Scaffold Widget

The Scaffold widget provides the basic structure for a Flutter app, including an AppBar, Drawer, FloatingActionButton, and BottomNavigationBar. It is a fundamental widget used to create a standard screen layout in Flutter.

(b) Container Widget

A Container is a box model widget that can hold other widgets. It is commonly used for adding padding, margins, borders, and background decorations.

(c) Row and Column Widgets

- Row: Arranges widgets horizontally.
- Column: Arranges widgets vertically.
  These two widgets are fundamental for designing layouts in Flutter.

(d) ListView Widget

The ListView widget is used for displaying a scrollable list of items. It is useful for showing large amounts of data dynamically.

(e) Stack Widget

The Stack widget is used to place widgets on top of each other. This is useful for creating overlapping UI elements such as banners, profile images, or layered designs.

(f) ElevatedButton Widget

The ElevatedButton widget is used for clickable buttons with a raised effect. It is a commonly used button in Flutter applications.

(g) TextField Widget

The TextField widget is used to take user input, such as entering a name, email, or password. It is commonly used in forms and authentication screens.

Code:

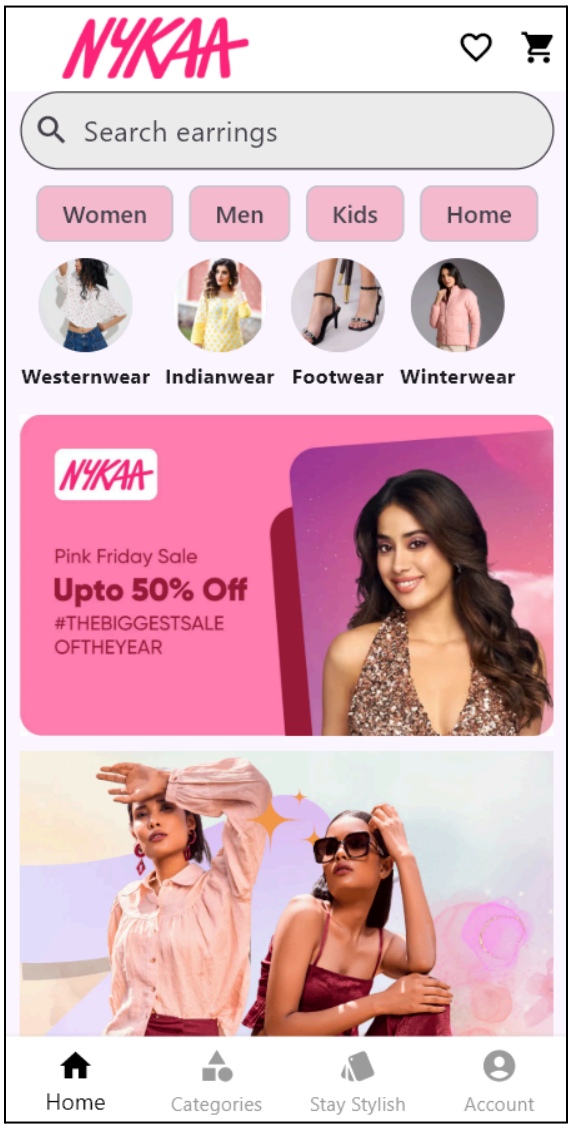
Home.dart file

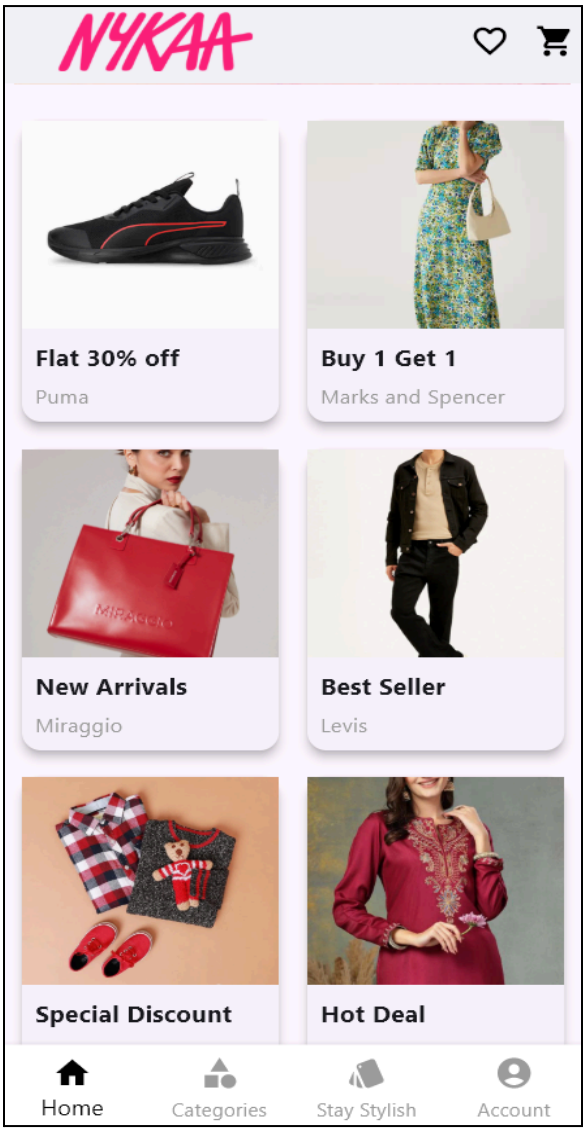
```
import 'package:flutter/material.dart';
import 'category.dart';
import 'bag.dart';

void main() {
 runApp(MyApp());
}
final List<Map<String, String>>
productDetails = [
 {"tagline": "Flat 30% off", "description":
"Puma"},
 {"tagline": "Buy 1 Get 1", "description":
"Marks and Spencer"},
 {"tagline": "New Arrivals", "description":
"Miraggio"},
 {"tagline": "Best Seller", "description":
"Levis"},
 {"tagline": "Special Discount", "description":
"HopScotch"},
 {"tagline": "Hot Deal", "description":
"Nayam"},
];
class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   debugShowCheckedModeBanner: false,
   home: NykaaHomePage(),
  );
 }
}

class NykaaHomePage extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    backgroundColor: Colors.white,
    elevation: 0,
    title:
Image.asset('assets/nykaa_logo.png',
height: 40),
    actions: [
     IconButton(icon:
Icon(Icons.favorite_border, color:
Colors.black), onPressed: () {}),
     IconButton(
      icon: const Icon(Icons.shopping_cart,
color: Colors.black),
      onPressed: () {
       Navigator.push(
        context,
        MaterialPageRoute(builder:
(context) => ShoppingBagScreen()), //
Replace with your target screen
       );
      },
     ),
    ],
   ),
   body: SingleChildScrollView(
    child: Column(
     children: [
      // Search Bar
      Padding(
       padding:
EdgeInsets.symmetric(horizontal: 10),
       child: TextField(
        decoration: InputDecoration(
         prefixIcon: Icon(Icons.search),
         hintText: "Search earrings",
         border:
OutlineInputBorder(borderRadius:
BorderRadius.circular(30)),
         filled: true,
         fillColor: Colors.grey[200],
        ),
       ),
      ),
      SizedBox(height: 10),

      // Category Tabs
      Padding(
       padding:
EdgeInsets.symmetric(horizontal: 10),
       child: Row(
```

```
            mainAxisAlignment:
MainAxisAlignment.spaceEvenly,
            children: ["Women", "Men", "Kids",
"Home"].map((category) {
              return Chip(
                label: Text(category),
                backgroundColor:
Colors.pink.shade100,
              );
            }).toList(),
          ),
        ),
        SizedBox(height: 10),

        // Scrollable Categories
        SizedBox(
          height: 90,
          child: ListView(
            scrollDirection: Axis.horizontal,
            padding:
EdgeInsets.symmetric(horizontal: 10), //
Padding around the entire list
            children: [
              Padding(
                padding: EdgeInsets.only(right:
10), // Space between the cards
                child: CategoryCard(category:
"Westernwear", image:
"assets/westernwear.png"),
              ),
              Padding(
                padding: EdgeInsets.only(right:
10),
                child: CategoryCard(category:
"Indianwear", image:
"assets/indianwear.png"),
              ),
              Padding(
                padding: EdgeInsets.only(right:
10),
                child: CategoryCard(category:
"Footwear", image: "assets/footwear.png"),
              ),
              Padding(
                padding: EdgeInsets.only(right:
10),
                child: CategoryCard(category:
"Winterwear", image:
"assets/winterwear.png"),
              ),
            ],
          ),
        ),
        SizedBox(height: 10),


        // Offer Banner
        Padding(
          padding:
EdgeInsets.symmetric(horizontal: 10),
          child:
Image.asset('assets/offer_banner.png', fit:
BoxFit.cover),
        ),
        SizedBox(height: 10),

        // Sale Banner
        Padding(
          padding:
EdgeInsets.symmetric(horizontal: 10),
          child:
Image.asset('assets/sale_banner.png', fit:
BoxFit.cover),
        ),
        SizedBox(height: 20),

        // Grid Section (2 columns, 3 rows)
        Padding(
          padding:
EdgeInsets.symmetric(horizontal: 10),
          child: GridView.builder(
            shrinkWrap: true,
            physics:
NeverScrollableScrollPhysics(),
            gridDelegate:
SliverGridDelegateWithFixedCrossAxisCou
nt(
              crossAxisCount: 2,
              mainAxisSpacing: 10,
              crossAxisSpacing: 10,
              childAspectRatio: 0.8,
            ),
            itemCount: productDetails.length,
            itemBuilder: (context, index) {
              return ProductCard(
                image: "assets/product${index +
1}.png",
```

```
                tagline:
productDetails[index]["tagline"]!,
                description:
productDetails[index]["description"]!,
              );
            },
          ),
        ),
        SizedBox(height: 20),
      ],
    ),
  ),

  // Bottom Navigation Bar
  bottomNavigationBar:
BottomNavigationBar(
    currentIndex: 0,
    selectedItemColor: Colors.black, // Set
selected icon color
    unselectedItemColor: Colors.grey, // Set
unselected icon color
    backgroundColor: Colors.white, //
Ensure background is visible
    showUnselectedLabels: true, // Show
labels even when unselected
    type: BottomNavigationBarType.fixed, //
Keeps icons and text aligned
    onTap: (index) {
      if (index == 1) {
        Navigator.push(
          context,
          MaterialPageRoute(builder:
(context) => CategoriesPage()),
        );
      }
    },
    items: [
      BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
      BottomNavigationBarItem(icon:
Icon(Icons.category), label: 'Categories'),
      BottomNavigationBarItem(icon:
Icon(Icons.style), label: 'Stay Stylish'),
      BottomNavigationBarItem(icon:
Icon(Icons.account_circle), label: 'Account'),
    ],
  ),
 );
}

// Category Card Widget
class CategoryCard extends
StatelessWidget {
 final String category;
 final String image;

 const CategoryCard({required
this.category, required this.image});

 @override
 Widget build(BuildContext context) {
  return Column(

   children: [

    CircleAvatar(
     backgroundImage:
AssetImage(image),
     radius: 30,
    ),
    SizedBox(height: 5),
    Text(category, style: TextStyle(fontSize:
12, fontWeight: FontWeight.bold)),

   ],
  );
 }
}

// Product Card Widget
class ProductCard extends StatelessWidget
{
 final String image;
 final String tagline;
 final String description;

 const ProductCard({required this.image,
required this.tagline, required
this.description});

 @override
 Widget build(BuildContext context) {
  return Card(
   elevation: 5,
   child: Column(
    crossAxisAlignment:
CrossAxisAlignment.start,
```

```
        children: [
          Expanded(
            child: Image.asset(image, fit:
BoxFit.cover, width: double.infinity),
          ),
          Padding(
            padding: EdgeInsets.all(8),
            child: Column(
              crossAxisAlignment:
CrossAxisAlignment.start,
              children: [
                Text(tagline, style:
TextStyle(fontWeight: FontWeight.bold,
fontSize: 14)),
                SizedBox(height: 5),
                Text(description, style:
TextStyle(fontSize: 12, color: Colors.grey)),
              ],
            ),
          ),
        ],
      ),
    );
  }
}
```

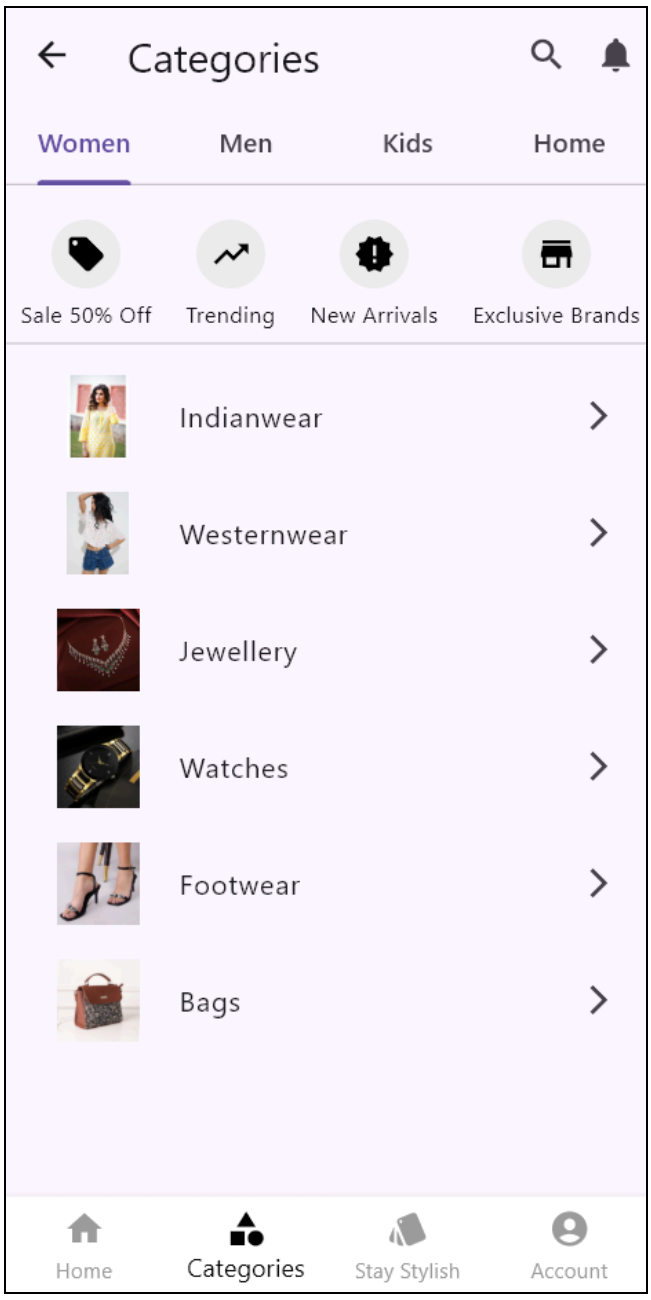




Category.dart file

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

class CategoriesPage extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return DefaultTabController(
    length: 4,
    child: Scaffold(
      appBar: AppBar(
        title: Text('Categories'),
        actions: [
          IconButton(icon: Icon(Icons.search),
onPressed: () {}),
          IconButton(icon:
Icon(Icons.notifications), onPressed: () {}),
        ],
        bottom: TabBar(
          tabs: [
            Tab(text: 'Women'),
            Tab(text: 'Men'),
            Tab(text: 'Kids'),
            Tab(text: 'Home'),
          ],
        ),
      ),

      body: Column(
        children: [
          SizedBox(height: 20),
          SingleChildScrollView(
            scrollDirection: Axis.horizontal,
            child: Row(
              children: [
                _buildCategoryIcon('Sale 50%
Off', Icons.local_offer),
                _buildCategoryIcon('Trending',
Icons.trending_up),
                _buildCategoryIcon('New
Arrivals', Icons.new_releases),
                _buildCategoryIcon('Exclusive
Brands', Icons.store),
              ],
            ),
          ),
          Divider(),
          Expanded(
            child: ListView(
              children: [
                _buildCategoryItem(context,
'Indianwear', 'assets/indianwear.png'),
                _buildCategoryItem(context,
'Westernwear', 'assets/westernwear.png'),
                _buildCategoryItem(context,
'Jewellery', 'assets/jewellery.png'),
                _buildCategoryItem(context,
'Watches', 'assets/watches.png'),
                _buildCategoryItem(context,
'Footwear', 'assets/footwear.png'),
                _buildCategoryItem(context,
'Bags', 'assets/bags.png'),
              ],
            ),
          ),
        ],
      ),

      bottomNavigationBar:
BottomNavigationBar(
        currentIndex: 1,
        selectedItemColor: Colors.black,
        unselectedItemColor: Colors.grey,
        backgroundColor: Colors.white,
        showUnselectedLabels: true,
        type: BottomNavigationBarType.fixed,
        items: [
          BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
          BottomNavigationBarItem(icon:
Icon(Icons.category), label: 'Categories'),
          BottomNavigationBarItem(icon:
Icon(Icons.style), label: 'Stay Stylish'),
          BottomNavigationBarItem(icon:
Icon(Icons.account_circle), label: 'Account'),
        ],
```

```
      ),
    ),
  );
}

 Widget _buildCategoryIcon(String label,
IconData icon) {
  return Padding(
    padding: const
EdgeInsets.symmetric(horizontal: 10),
    child: Column(
      children: [
        CircleAvatar(
          backgroundColor:
Colors.grey.shade200,
          child: Icon(icon, color: Colors.black),
        ),
        SizedBox(height: 5),
        Text(label, textAlign: TextAlign.center,
style: TextStyle(fontSize: 12)),
      ],
    ),
  );
}

 Widget _buildCategoryItem(BuildContext
context, String label, String imagePath) {
  return ListTile(
    contentPadding:
EdgeInsets.symmetric(vertical: 10,
horizontal: 20),
    leading: Image.asset(imagePath, width:
70, height: 70),
    title: Text(
      label,
      style: TextStyle(fontSize: 15),
    ),
    trailing: Icon(Icons.arrow_forward_ios,
size: 20),
          ),
      );
    },
  );
}
}
```


Categories
Women
Men
Kids
Home
Sale 50% Off
Trending
New Arrivals
Exclusive Brands
Indianwear
Westernwear
Jewellery
Watches
Footwear
Bags
Home
Categories
Stay Stylish
Account

MAD & PWA Lab
Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

Aim: To include icons, images, fonts in Flutter app

Theory:

Using Icons in Flutter

Icons in Flutter can be added using the built-in Material Icons or custom icon packs.

(a) Material Icons

Flutter provides a collection of built-in Material Icons, which can be used with the Icon widget.

Eg: Icon(Icons.home, size: 30, color: Colors.blue)

(b) Custom Icons

If you need icons that are not available in the Material Icons set, you can use external icon packs like:

- Font Awesome (font_awesome_flutter package)
- Custom SVG Icons (flutter_svg package)

  Eg in pubspec.yaml file -

  dependencies:
    font_awesome_flutter: ^10.5.0

In code -

```
import 'package:font_awesome_flutter/font_awesome_flutter.dart';

IconButton(
  icon: FaIcon(FontAwesomeIcons.heart, color: Colors.red),
  onPressed: () {},
)
```

Adding Images in Flutter

Images can be loaded in Flutter from different sources like assets, network, or memory.

(a) Using Network Images

Network images are loaded from an online URL. Example:

Eg: Image.network("https://example.com/sample.jpg", width: 200, height: 150)

(b) Using Asset Images

To use images from the local project folder (assets/), follow these steps:

1. Place the image inside the assets/images/ folder.
2. Declare the image in pubspec.yaml:

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

In code: Image.asset("assets/images/sample.png", width: 200, height: 150)

Adding Custom Fonts in Flutter

Custom fonts improve the visual identity of an app.

Steps to Add a Custom Font:

1. Download the font and place it inside the assets/fonts/ folder.
2. Declare the font in pubspec.yaml:

```
flutter:
  fonts:
    - family: CustomFont
      fonts:
        - asset: assets/fonts/CustomFont-Regular.ttf
        - asset: assets/fonts/CustomFont-Bold.ttf
          weight: 700
```

In code -

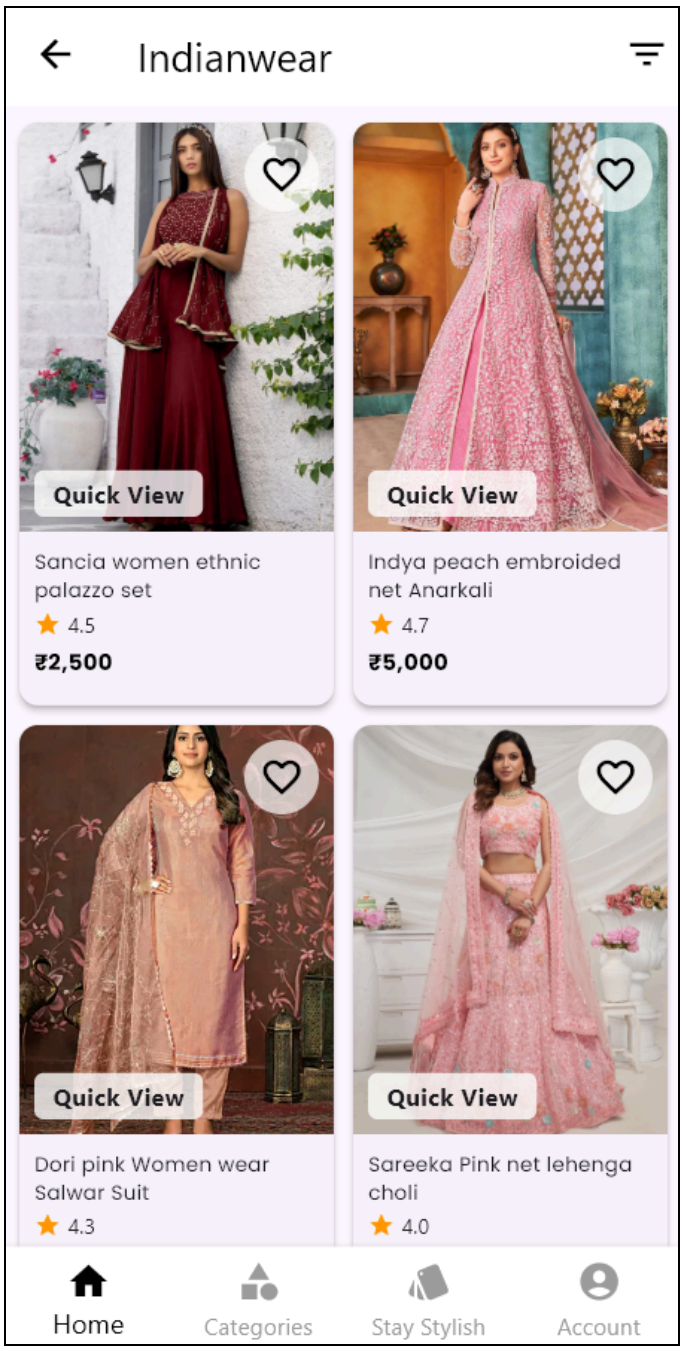
```
Text(
  "Hello, Flutter!",
  style: TextStyle(fontFamily: "CustomFont", fontSize: 20, fontWeight: FontWeight.bold),
)
```

Product.dart file

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

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

final List<Map<String, String>>
productDetails = [
 {"name": "Sancia women ethnic palazzo
set", "price": "₹2,500", "image":
"assets/palazzo.png", "rating": "4.5"},
 {"name": "Indya peach embroided net
Anarkali", "price": "₹5,000", "image":
"assets/anarkali.png", "rating": "4.7"},
 {"name": "Dori pink Women wear Salwar
Suit", "price": "₹1,200", "image":
"assets/salwar.png", "rating": "4.3"},
 {"name": "Sareeka Pink net lehenga choli",
"price": "₹800", "image":
"assets/lehenga.png", "rating": "4.0"},
];

class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   debugShowCheckedModeBanner: false,
   home: IndianwearPage(),
  );
 }
}

class IndianwearPage extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    backgroundColor: Colors.white,
    elevation: 0,
    title: Text(
     "Indianwear",
     style: TextStyle(color: Colors.black,
fontSize: 20),
    ),
    leading: IconButton(
     icon: Icon(Icons.arrow_back, color:
Colors.black),
     onPressed: () =>
Navigator.pop(context),
    ),
    actions: [
     IconButton(
      icon: Icon(Icons.filter_list, color:
Colors.black),
      onPressed: () {
       // Add filter functionality
      },
     ),
    ],
   ),
   body: CustomScrollView(
    slivers: [
     SliverPadding(
      padding:
EdgeInsets.symmetric(horizontal: 5, vertical:
5),
      sliver: SliverGrid(
       delegate:
SliverChildBuilderDelegate(
        (BuildContext context, int index)
{
         return ProductCard(
          name:
productDetails[index]["name"]!,
          price:
productDetails[index]["price"]!,
          image:
productDetails[index]["image"]!,
          rating:
productDetails[index]["rating"]!,
         );
        },
        childCount: productDetails.length,
       ),
       gridDelegate:
SliverGridDelegateWithFixedCrossAxisCou
nt(
        crossAxisCount: 2,
```

```
            mainAxisSpacing: 2,
            crossAxisSpacing: 2,
            childAspectRatio: 0.58,
          ),
        ),
      ),
    ],
   ),
   bottomNavigationBar:
BottomNavigationBar(
    currentIndex: 0,
    selectedItemColor: Colors.black,
    unselectedItemColor: Colors.grey,
    backgroundColor: Colors.white,
    showUnselectedLabels: true,
    type: BottomNavigationBarType.fixed,
    onTap: (index) {
     // Add navigation functionality if
needed
    },
    items: [
     BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
     BottomNavigationBarItem(icon:
Icon(Icons.category), label: 'Categories'),
     BottomNavigationBarItem(icon:
Icon(Icons.style), label: 'Stay Stylish'),
     BottomNavigationBarItem(icon:
Icon(Icons.account_circle), label: 'Account'),
    ],
   ),
  );
 }
}

class ProductCard extends StatelessWidget
{
 final String name;
 final String price;
 final String image;
 final String rating;

 const ProductCard({
  required this.name,
  required this.price,
  required this.image,
  required this.rating,
 });

 @override
 Widget build(BuildContext context) {
  return GestureDetector(
   onTap: () {
    // Navigate to the ProductDetailPage
    Navigator.push(
     context,
     MaterialPageRoute(
      builder: (context) =>
ProductDetailPage(
       name: name,
       price: price,
       image: image,
       rating: rating,
      ),
     ),
    );
   },
   child: Card(
    elevation: 3,
    shape:
RoundedRectangleBorder(borderRadius:
BorderRadius.circular(10)),
    child: Column(
     crossAxisAlignment:
CrossAxisAlignment.start,
     children: [
      // Product Image with Heart Icon &
Quick View Button
      Stack(
       children: [
        ClipRRect(
         borderRadius:
BorderRadius.vertical(top:
Radius.circular(10)),
         child: Image.asset(
          image,
          width: double.infinity,
          height: 220,  // Increased image
height
          fit: BoxFit.cover,
         ),
        ),
        // Heart Icon
        Positioned(
         top: 8,
         right: 8,
```

```
              child: Container(
                decoration: BoxDecoration(
                  color:
Colors.white.withOpacity(0.7),
                  shape: BoxShape.circle,
                ),
                child: IconButton(
                  icon:
Icon(Icons.favorite_border, color:
Colors.black),
                  onPressed: () {
                    // Add to wishlist functionality
                  },
                ),
              ),
            ),
            // Quick View Button
            Positioned(
              bottom: 8,
              left: 8,
              child: Container(
                padding:
EdgeInsets.symmetric(vertical: 3, horizontal:
10),
                decoration: BoxDecoration(
                  color:
Colors.white.withOpacity(0.8),
                  borderRadius:
BorderRadius.circular(5),
                ),
                child: Text(
                  "Quick View",
                  style: TextStyle(fontSize: 12,
fontWeight: FontWeight.bold),
                ),
              ),
            ),
          ],
        ),
        // Wrap the Column with a Flexible
widget to allow ListView
        Flexible(
          fit: FlexFit.tight,
          child: ListView(
            padding: EdgeInsets.all(8),
            shrinkWrap: true,  // Ensures
ListView doesn't take more space than
needed
            physics:
NeverScrollableScrollPhysics(),  // Disables
scrolling inside the ListView
            children: [
              Text(name, style:
TextStyle(fontSize: 10, fontFamily:
"Poppins")),
              SizedBox(height: 4),
              Row(
                children: [
                  Icon(Icons.star, color:
Colors.orange, size: 14),
                  SizedBox(width: 4),
                  Text(rating, style:
TextStyle(fontSize: 10, color:
Colors.black87)),
                ],
              ),
              SizedBox(height: 4),
              Text(price, style:
TextStyle(fontSize: 11, color: Colors.black,
fontWeight: FontWeight.bold, fontFamily:
"Poppins")),
            ],
          ),
        ],
      ),
    ),
  );
 }
}
```



ProductDetailPage.dart

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

class ProductDetailPage extends
StatelessWidget {
 final String name;
 final String price;
 final String image;
 final String rating;

const ProductDetailPage({
  required this.name,
  required this.price,
  required this.image,
  required this.rating,
});

@override
Widget build(BuildContext context) {
```

```
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.white,
      elevation: 0,
      title: Text(
        "Product Details",
        style: TextStyle(color: Colors.black,
fontSize: 20),
      ),
      leading: IconButton(
        icon: Icon(Icons.arrow_back, color:
Colors.black),
        onPressed: () =>
Navigator.pop(context),
      ),
    ),
    body: SingleChildScrollView(  // Makes
the entire page scrollable
      padding: const EdgeInsets.all(8.0),
      child: Container(
        margin:
EdgeInsets.symmetric(horizontal: 12.0),
        child: Column(
          crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
            // Product Image
            Center(
              child: ClipRRect(
                borderRadius:
BorderRadius.circular(6),
                child: Image.asset(
                  image,
                  width: 300,
                  height: 400,
                  fit: BoxFit.cover,
                ),
              ),
            ),
            SizedBox(height: 20),
            // Product Name
            Text(name, style:
TextStyle(fontSize: 18, fontFamily:
"Poppins")),
            SizedBox(height: 10),
            // Rating
            Row(
              children: [
                Icon(Icons.star, color:
Colors.orange, size: 15),
                SizedBox(width: 5),
                Text(rating, style:
TextStyle(fontSize: 14)),
              ],
            ),
            SizedBox(height: 10),
            // Price
            Text(price, style: TextStyle(fontSize:
18, color: Colors.black,fontFamily:
"Poppins", fontWeight: FontWeight.bold)),
            SizedBox(height: 20),
            // Product Description
            Text(
              "This is a great product. It offers a
blend of style, comfort, and quality. Perfect
for any occasion.",
              style: TextStyle(fontSize: 12, color:
Colors.black87),
            ),
            SizedBox(height: 20),
            // Select Size (like Nykaa)
            Text("Select Size", style:
TextStyle(fontSize: 14, fontWeight:
FontWeight.bold, fontFamily: "Poppins")),
            SizedBox(height: 10),
            DropdownButton<String>(
              isExpanded: true,
              hint: Text("Choose a size"),
              items: <String>['S', 'M', 'L', 'XL',
'XXL'].map((String size) {
                return
DropdownMenuItem<String>(
                  value: size,
                  child: Text(size),
                );
              }).toList(),
              onChanged: (value) {
                // Handle size selection
              },
            ),
            SizedBox(height: 20),
            // Product Details
            Text("Product Details", style:
TextStyle(fontSize: 14, fontWeight:
FontWeight.bold, fontFamily: "Poppins")),
            SizedBox(height: 10),
```

```
            Text(
              "Material: Cotton\nBrand:
XYZ\nCare: Hand wash\nColor:
Red\nPattern: Solid",
              style: TextStyle(fontSize: 12, color:
Colors.black87, fontFamily: "Poppins"),
            ),
            SizedBox(height: 20),
            // Delivery Date
            Text("Delivery Date", style:
TextStyle(fontSize: 14, fontWeight:
FontWeight.bold, fontFamily: "Poppins")),
            SizedBox(height: 10),
            Text(
              "Expected delivery: 3-5 business
days.",
              style: TextStyle(fontSize: 14, color:
Colors.black87, fontFamily: "Poppins"),
            ),
            SizedBox(height: 20),
            // Add to Cart Button
            Center(
              child: ElevatedButton(
                onPressed: () {
                  // Add to cart functionality (you
can implement it here)
                },
                style: ElevatedButton.styleFrom(
                  padding:
EdgeInsets.symmetric(vertical: 15,
horizontal: 100),

                  shape:
RoundedRectangleBorder(borderRadius:
BorderRadius.circular(30)),
                ),
                child: Text("Add to Cart", style:
TextStyle(fontSize: 16, color: Colors.pink,
fontFamily: "Poppins")),
              ),
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

pubspec.yaml

```
flutter:
  assets:
    - assets/pinkbg.jpg
    - assets/women.png
    - assets/indianwear.png
    - assets/footwear.png
    - assets/westernwear.png
    - assets/winterwear.png
    - assets/offer_banner.png
    - assets/sale_banner.png
    - assets/nykaa_logo.png
    - assets/product1.png
    - assets/product2.png
    - assets/product3.png
    - assets/product4.png
    - assets/product5.png
    - assets/product6.png
    - assets/foot.png
    - assets/ethnicwear.png
    - assets/watches.png
    - assets/jewellery.png
    - assets/bags.png
    - assets/lehenga.png
    - assets/salwar.png
    - assets/palazzo.png
    - assets/anarkali.png

  fonts:
    - family: Poppins
      fonts:
        - asset: assets/Poppins-Regular.ttf
        - asset: assets/Poppins-Bold.ttf
          weight: 700
```

← Product Details

Sancia women ethnic palazzo set

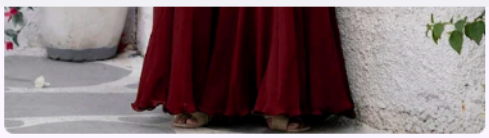
★ 4.5

₹2,500

This is a great product. It offers a blend of style, comfort, and quality. Perfect for any occasion.

Select Size

← Product Details



Sancia women ethnic palazzo set

★ 4.5

₹2,500

This is a great product. It offers a blend of style, comfort, and quality. Perfect for any occasion.

Select Size

Choose a size

Product Details

Material: Cotton
Brand: XYZ
Care: Hand wash
Color: Red
Pattern: Solid

Delivery Date

Expected delivery: 3-5 business days.

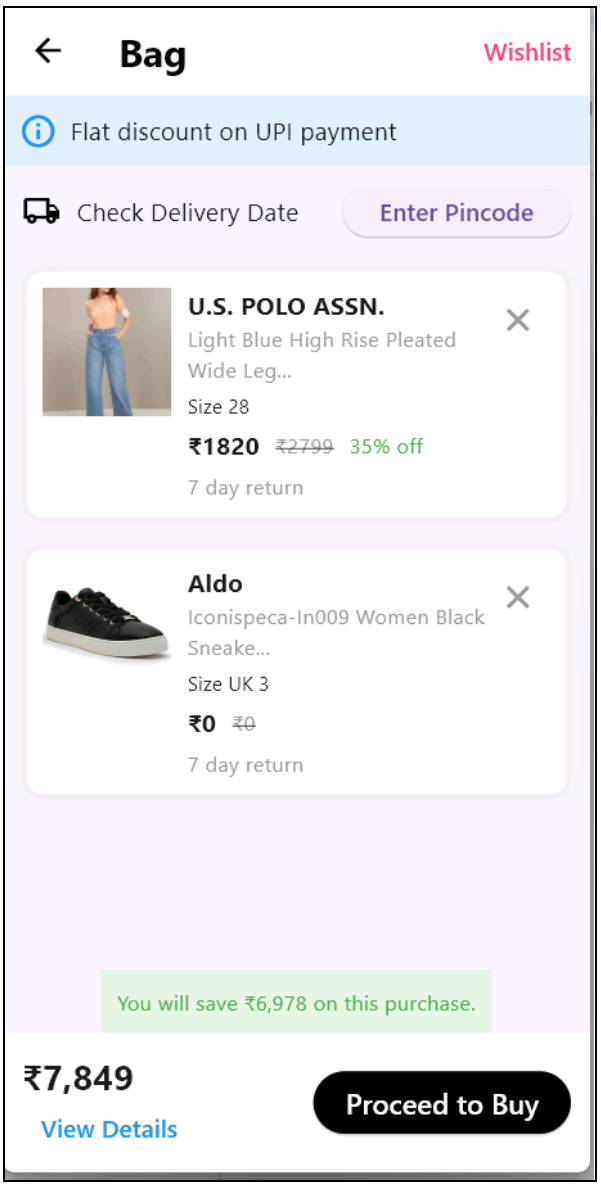
Add to Cart

Bag.dart

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

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

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

 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   debugShowCheckedModeBanner: false,
   home: ShoppingBagScreen(),
  );
 }
}

class ShoppingBagScreen extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    backgroundColor: Colors.white,
    elevation: 0,
    leading: IconButton(
     icon: const Icon(Icons.arrow_back,
color: Colors.black),
     onPressed: () {
      Navigator.pop(context); // Go back to
the previous screen
     },
    ),
    title: const Text(
     "Bag",
     style: TextStyle(color: Colors.black,
fontSize: 22, fontWeight: FontWeight.bold),
    ),
    actions: [
     TextButton(
      onPressed: () {},
      child: const Text(
       "Wishlist",
       style: TextStyle(color:
Colors.pinkAccent, fontSize: 14),
      ),
     ),
    ],
   ),
   body: Column(
    children: [


     // UPI Discount Info
     Container(
      padding: const EdgeInsets.all(10),
      color: Colors.blue[50],
      child: const Row(
       children: [
        Icon(Icons.info_outline, color:
Colors.blue),
        SizedBox(width: 8),
        Text("Flat discount on UPI
payment"),
       ],
      ),
     ),

     // Check Delivery Date Section
     Container(
      padding: const EdgeInsets.all(12),
      child: Row(
       children: [
        const
Icon(Icons.local_shipping_outlined, color:
Colors.black),
        const SizedBox(width: 10),
        const Text("Check Delivery Date"),
        const Spacer(),
        ElevatedButton(
         onPressed: () {},
         child: const Text("Enter
Pincode"),
        ),
       ],
      ),
     ),

     // Product List
     Expanded(
      child: ListView(
       children: [
        ProductItem(
```

```
            imageUrl:
"https://uspoloassn.in/cdn/shop/products/3_feb4870a-798f-4663-9eed-768c9e04a376_3024x.jpg?v=1686246326", // Replace with actual image URL
            brand: "U.S. POLO ASSN.",
            title: "Light Blue High Rise Pleated Wide Leg...",
            size: "Size 28",
            price: 1820,
            originalPrice: 2799,
            discount: "35% off",
          ),
          ProductItem(
            imageUrl:
"https://images-static.nykaa.com/media/catalog/product/tr:h-800,w-800,cm-pad_resize/c/3/c35b0abICONISPECAIN009_1.jpg", // Replace with actual image URL
            brand: "Aldo",
            title: "Iconispeca-In009 Women Black Sneake...",
            size: "Size UK 3",
            price: 0, // Update price
            originalPrice: 0, // Update original price
            discount: "", // Update discount if needed
          ),
        ],
      ),
    ),

    // Savings Section
    Container(
      padding: const EdgeInsets.all(10),
      color: Colors.green[50],
      child: const Text(
        "You will save ₹6,978 on this purchase.",
        style: TextStyle(color: Colors.green, fontSize: 12),
      ),
    ),

    // Total Price & Proceed to Buy Button
    Container(
      padding: const EdgeInsets.all(12),
      color: Colors.white,
      child: Row(
        mainAxisAlignment:
MainAxisAlignment.spaceBetween,
        children: [
          Column(
            crossAxisAlignment:
CrossAxisAlignment.start,
            children: [
              const Text(
                "₹7,849",
                style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
              ),
              TextButton(
                onPressed: () {},
                child: const Text("View Details", style: TextStyle(color: Colors.blue)),
              ),
            ],
          ),
          ElevatedButton(
            style: ElevatedButton.styleFrom(
              backgroundColor: Colors.black,
              padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 15),
            ),
            onPressed: () {},
            child: const Text("Proceed to Buy", style: TextStyle(color: Colors.white, fontSize: 16)),
          ),
        ],
      ),
    ),
  ],
 ),
);
}
}

class ProductItem extends StatelessWidget {
 final String imageUrl;
 final String brand;
 final String title;
```

```
 final String size;
 final int price;
 final int originalPrice;
 final String discount;

 const ProductItem({
  required this.imageUrl,
  required this.brand,
  required this.title,
  required this.size,
  required this.price,
  required this.originalPrice,
  required this.discount,
 });

 @override
 Widget build(BuildContext context) {
  return Container(
    margin: const
EdgeInsets.symmetric(vertical: 10,
horizontal: 15),
    padding: const EdgeInsets.all(10),
    decoration: BoxDecoration(
     color: Colors.white,
     borderRadius:
BorderRadius.circular(10),
     boxShadow: [
      BoxShadow(color:
Colors.grey.shade200, blurRadius: 5,
spreadRadius: 1),
     ],
    ),
    child: Row(
     crossAxisAlignment:
CrossAxisAlignment.start,
     children: [
      Image.network(imageUrl, width: 80,
height: 80, fit: BoxFit.cover),
      const SizedBox(width: 10),
      Expanded(
       child: Column(
        crossAxisAlignment:
CrossAxisAlignment.start,
        children: [
         Text(brand, style: const
TextStyle(fontSize: 14, fontWeight:
FontWeight.bold)),
         Text(title, style: const
TextStyle(fontSize: 12, color: Colors.grey)),
         const SizedBox(height: 5),
         Text(size, style: const
TextStyle(fontSize: 11)),
         const SizedBox(height: 5),
         Row(
          children: [
           Text("₹$price", style: const
TextStyle(fontSize: 14, fontWeight:
FontWeight.bold)),
           const SizedBox(width: 10),
           Text(
            "₹$originalPrice",
            style: const TextStyle(fontSize:
12, decoration: TextDecoration.lineThrough,
color: Colors.grey),
           ),
           const SizedBox(width: 10),
           Text(discount, style: const
TextStyle(fontSize: 12, color: Colors.green)),
          ],
         ),
         const SizedBox(height: 5),
         const Text("7 day return", style:
TextStyle(fontSize: 12, color: Colors.grey)),
        ],
       ),
      ),
      IconButton(
       icon: const Icon(Icons.close, color:
Colors.grey),
       onPressed: () {},
      ),
     ],
    ),
   );
 }
}
```


Bag
Wishlist
Flat discount on UPI payment
Check Delivery Date
Enter Pincode
U.S. POLO ASSN.
Light Blue High Rise Pleated Wide Leg...
Size 28
₹1820 ₹2799 35% off
7 day return
Aldo
Iconispeca-In009 Women Black Sneake...
Size UK 3
₹0 ₹0
7 day return
You will save ₹6,978 on this purchase.
₹7,849
View Details
Proceed to Buy

MAD & PWA Lab
Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

Aim:

To create an interactive form using Form widget

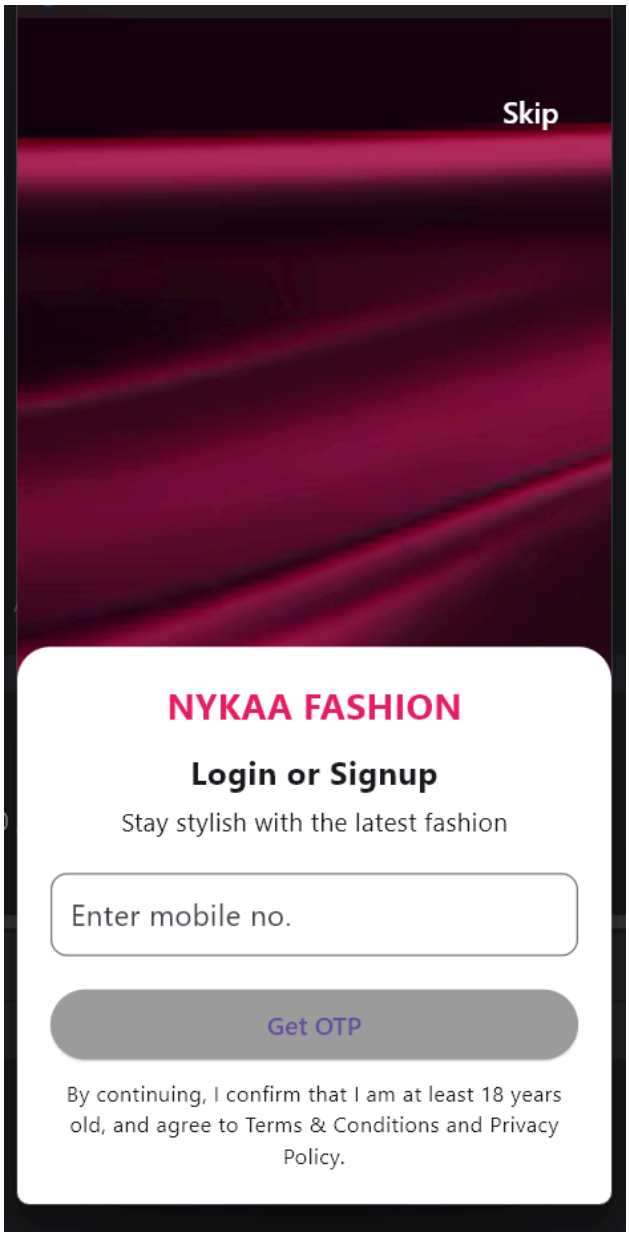
Program:

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

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

class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   debugShowCheckedModeBanner: false,
   home: LoginPage(),
  );
 }
}

class LoginPage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   body: Stack(
    children: [
     Container(
      decoration: BoxDecoration(
       image: DecorationImage(
        image: AssetImage('assets/pinkbg.jpg'), // Add a maroon silk texture
        fit: BoxFit.cover,
       ),
      ),
```

```
),
Padding(
  padding: EdgeInsets.only(top: 40, right: 20),
  child: Align(
    alignment: Alignment.topRight,
    child: TextButton(
      onPressed: () {},
      child: Text(
        "Skip",
        style: TextStyle(color: Colors.white, fontSize: 16),
      ),
    ),
  ),
),
Align(
  alignment: Alignment.bottomCenter,
  child: Container(
    padding: EdgeInsets.all(20),
    decoration: BoxDecoration(
      color: Colors.white,
      borderRadius: BorderRadius.only(
        topLeft: Radius.circular(20),
        topRight: Radius.circular(20),
      ),
    ),
    child: Column(
      mainAxisSize: MainAxisSize.min,
      children: [
        Text(
          "NYKAA FASHION",
          style: TextStyle(
            fontSize: 20,
            fontWeight: FontWeight.bold,
            color: Colors.pink,
          ),
        ),
        SizedBox(height: 10),
        Text(
          "Login or Signup",
```

```
              style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
            ),
            SizedBox(height: 5),
            Text("Stay stylish with the latest fashion"),
            SizedBox(height: 20),
            TextField(
              keyboardType: TextInputType.phone,
              decoration: InputDecoration(
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(10),
                ),
                hintText: "Enter mobile no.",
              ),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {},
              style: ElevatedButton.styleFrom(
                backgroundColor: Colors.grey,
                minimumSize: Size(double.infinity, 50),
              ),
              child: Text("Get OTP"),
            ),
            SizedBox(height: 10),
            Text(
              "By continuing, I confirm that I am at least 18 years old, and agree to Terms &
Conditions and Privacy Policy.",
              textAlign: TextAlign.center,
              style: TextStyle(fontSize: 12),
            ),
          ],
        ),
      ),
    ),
  ),
 );
}
}
```

Output:



Common widgets used-

1. Scaffold
    - The main structure of the screen.
    - Provides the base layout like body.

2. Stack
    - Places widgets on top of each other (background image, button, and login form).

3. Container
    - Used for styling and layout purposes (background image, white login box).

4. BoxDecoration
    - Adds styling inside Container (background image, rounded corners).

5. DecorationImage
    - Used inside BoxDecoration to set an image as a background.

6. Align
    - Positions widgets at specific places (Skip button at the top-right, login box at the bottom).

7. Padding
    - Adds spacing around widgets.

8. TextButton
    - Used for the "Skip" button.

9. Column
    - Arranges widgets vertically inside the login box.

10. Text
    - Displays static text (NYKAA FASHION, Login or Signup, Stay stylish...).

11. SizedBox
    - Adds empty space between widgets.

12. TextField
    - Takes user input (mobile number).

13. InputDecoration
    - Adds a border and placeholder text (Enter mobile no.) inside TextField.

14. OutlineInputBorder
    - Provides a border for the TextField.

15. ElevatedButton
    - The "Get OTP" button.

16. TextStyle
    - Used to style text (color, font size, weight).

17. BorderRadius
    - Rounds corners of the white login box.

MAD & PWA Lab
Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

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

Theory:

Flutter provides tools to handle navigation, routing, and gestures, allowing users to move between screens and interact with the app smoothly. These features help create a user-friendly experience in mobile applications.

---

1. Navigation in Flutter

Navigation is the process of moving between different screens (or pages) in a Flutter app. Flutter uses a stack-based approach for navigation, where new screens are pushed onto the stack and removed when the user navigates back.

Types of Navigation:

- Push Navigation: Moves to a new screen and adds it to the stack.
- Pop Navigation: Removes the current screen and returns to the previous one.
- Named Routes: Uses pre-defined route names to navigate.
- Navigation with Data: Allows passing data between screens when navigating.

---

2. Routing in Flutter

Routing helps in managing different screens efficiently. Instead of manually handling each screen transition, Flutter allows defining routes in a structured way.

Types of Routing:

- Direct Routing: Navigates to a specific screen using explicit methods.
- Named Routing: Uses a predefined route name to navigate, making the app more organized.

Routing improves app maintainability, especially in apps with multiple screens.

---

3. Gestures in Flutter

Gestures enable user interaction in Flutter applications. Flutter provides built-in gesture detection capabilities for touch-based interactions.

Common Gestures:

- Tap: A single touch interaction.
- Double Tap: Two quick consecutive taps.

- Long Press: Holding a touch for a longer duration.
- Swipe: Moving a finger across the screen.
- Drag: Moving an object by pressing and holding it.

Gestures are essential for making apps interactive and responsive.

---

4. Combining Navigation and Gestures

Navigation and gestures can be combined to enhance user experience. For example:

- Tapping on a button can navigate to another screen.
- Swiping a card can delete an item or move to another page.
- Dragging an element can reposition items within the app.

Navigation, routing, and gestures are fundamental to creating an interactive Flutter application. Navigation allows movement between screens, routing helps manage screens efficiently, and gestures enable touch interactions. Mastering these concepts helps in developing dynamic and user-friendly Flutter applications.

Code:

home.dart file

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

void main() {
 runApp(MyApp());
}
final List<Map<String, String>>
productDetails = [
 {"tagline": "Flat 30% off", "description":
"Puma"},
 {"tagline": "Buy 1 Get 1", "description":
"Marks and Spencer"},
 {"tagline": "New Arrivals", "description":
"Miraggio"},
 {"tagline": "Best Seller", "description":
"Levis"},
 {"tagline": "Special Discount", "description":
"HopScotch"},
 {"tagline": "Hot Deal", "description":
"Nayam"},
];
class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
   return MaterialApp(
     debugShowCheckedModeBanner: false,
     home: NykaaHomePage(),
   );
 }
}

class NykaaHomePage extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     appBar: AppBar(
```

```
      backgroundColor: Colors.white,
      elevation: 0,
      title:
Image.asset('assets/nykaa_logo.png',
height: 40),
      actions: [
        IconButton(icon:
Icon(Icons.favorite_border, color:
Colors.black), onPressed: () {}),
        IconButton(
          icon: const Icon(Icons.shopping_cart,
color: Colors.black),
          onPressed: () {
            Navigator.push(
              context,
              MaterialPageRoute(builder:
(context) => ShoppingBagScreen()), //
Replace with your target screen
            );
          },
        ),
      ],
    ),
    body: SingleChildScrollView(
      child: Column(
        children: [
          // Search Bar
          Padding(
            padding:
EdgeInsets.symmetric(horizontal: 10),
            child: TextField(
              decoration: InputDecoration(
                prefixIcon: Icon(Icons.search),
                hintText: "Search earrings",
                border:
OutlineInputBorder(borderRadius:
BorderRadius.circular(30)),
                filled: true,
                fillColor: Colors.grey[200],
              ),
            ),
          ),
          SizedBox(height: 10),

          // Category Tabs
          Padding(
            padding:
EdgeInsets.symmetric(horizontal: 10),
            child: Row(
              mainAxisAlignment:
MainAxisAlignment.spaceEvenly,
              children: ["Women", "Men", "Kids",
"Home"].map((category) {
                return Chip(
                  label: Text(category),
                  backgroundColor:
Colors.pink.shade100,
                );
              }).toList(),
            ),
          ),
          SizedBox(height: 10),

          // Scrollable Categories
          SizedBox(
            height: 90,
            child: ListView(
              scrollDirection: Axis.horizontal,
              padding:
EdgeInsets.symmetric(horizontal: 10), //
Padding around the entire list
              children: [
                Padding(
                  padding: EdgeInsets.only(right:
10), // Space between the cards
                  child: CategoryCard(category:
"Westernwear", image:
"assets/westernwear.png"),
                ),
                Padding(
                  padding: EdgeInsets.only(right:
10),
                  child: CategoryCard(category:
"Indianwear", image:
"assets/indianwear.png"),
                ),
                Padding(
                  padding: EdgeInsets.only(right:
10),
                  child: CategoryCard(category:
"Footwear", image: "assets/footwear.png"),
                ),
                Padding(
                  padding: EdgeInsets.only(right:
10),
                  child: CategoryCard(category:
"Winterwear", image:
"assets/winterwear.png"),
```

```
              ),
            ],
          ),
        ),
        SizedBox(height: 10),


        // Offer Banner
        Padding(
          padding:
EdgeInsets.symmetric(horizontal: 10),
          child:
Image.asset('assets/offer_banner.png', fit:
BoxFit.cover),
        ),
        SizedBox(height: 10),

        // Sale Banner
        Padding(
          padding:
EdgeInsets.symmetric(horizontal: 10),
          child:
Image.asset('assets/sale_banner.png', fit:
BoxFit.cover),
        ),
        SizedBox(height: 20),

        // Grid Section (2 columns, 3 rows)
        Padding(
          padding:
EdgeInsets.symmetric(horizontal: 10),
          child: GridView.builder(
            shrinkWrap: true,
            physics:
NeverScrollableScrollPhysics(),
            gridDelegate:
SliverGridDelegateWithFixedCrossAxisCou
nt(
              crossAxisCount: 2,
              mainAxisSpacing: 10,
              crossAxisSpacing: 10,
              childAspectRatio: 0.8,
            ),
            itemCount: productDetails.length,
            itemBuilder: (context, index) {
              return ProductCard(
                image: "assets/product${index +
1}.png",
                tagline:
productDetails[index]["tagline"]!,
                description:
productDetails[index]["description"]!,
              );
            },
          ),
        ),
        SizedBox(height: 20),
      ],
    ),
  ),

  // Bottom Navigation Bar
  bottomNavigationBar:
BottomNavigationBar(
    currentIndex: 0,
    selectedItemColor: Colors.black, // Set
selected icon color
    unselectedItemColor: Colors.grey, // Set
unselected icon color
    backgroundColor: Colors.white, //
Ensure background is visible
    showUnselectedLabels: true, // Show
labels even when unselected
    type: BottomNavigationBarType.fixed, //
Keeps icons and text aligned
    onTap: (index) {
      if (index == 1) {
        Navigator.push(
          context,
          MaterialPageRoute(builder:
(context) => CategoriesPage()),
        );
      }
    },
    items: [
      BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
      BottomNavigationBarItem(icon:
Icon(Icons.category), label: 'Categories'),
      BottomNavigationBarItem(icon:
Icon(Icons.style), label: 'Stay Stylish'),
      BottomNavigationBarItem(icon:
Icon(Icons.account_circle), label: 'Account'),
    ],
  ),
 );
}
```

```
}

// Category Card Widget
class CategoryCard extends
StatelessWidget {
 final String category;
 final String image;

 const CategoryCard({required
this.category, required this.image});

 @override
 Widget build(BuildContext context) {
  return Column(

    children: [

      CircleAvatar(
        backgroundImage:
AssetImage(image),
        radius: 30,
      ),
      SizedBox(height: 5),
      Text(category, style: TextStyle(fontSize:
12, fontWeight: FontWeight.bold)),

    ],
  );
 }
}

// Product Card Widget
class ProductCard extends StatelessWidget
{
 final String image;
 final String tagline;
 final String description;

 const ProductCard({required this.image,
required this.tagline, required
this.description});

 @override
 Widget build(BuildContext context) {
  return Card(
    elevation: 5,
    child: Column(
      crossAxisAlignment:
CrossAxisAlignment.start,
      children: [
        Expanded(
          child: Image.asset(image, fit:
BoxFit.cover, width: double.infinity),
        ),
        Padding(
          padding: EdgeInsets.all(8),
          child: Column(
            crossAxisAlignment:
CrossAxisAlignment.start,
            children: [
              Text(tagline, style:
TextStyle(fontWeight: FontWeight.bold,
fontSize: 14)),
              SizedBox(height: 5),
              Text(description, style:
TextStyle(fontSize: 12, color: Colors.grey)),
            ],
          ),
        ),
      ],
    ),
  );
 }
}
```

category.dart file

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

class CategoriesPage extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return DefaultTabController(
   length: 4,
   child: Scaffold(
    appBar: AppBar(
     title: Text('Categories'),
     actions: [
      IconButton(icon: Icon(Icons.search),
onPressed: () {}),
      IconButton(icon:
Icon(Icons.notifications), onPressed: () {}),
     ],
     bottom: TabBar(
      tabs: [
       Tab(text: 'Women'),
       Tab(text: 'Men'),
       Tab(text: 'Kids'),
       Tab(text: 'Home'),
      ],
     ),
    ),
    body: Column(
     children: [
      SizedBox(height: 20),
      SingleChildScrollView(
       scrollDirection: Axis.horizontal,
       child: Row(
        children: [
         _buildCategoryIcon('Sale 50%
Off', Icons.local_offer),
         _buildCategoryIcon('Trending',
Icons.trending_up),
         _buildCategoryIcon('New
Arrivals', Icons.new_releases),
         _buildCategoryIcon('Exclusive
Brands', Icons.store),
        ],
       ),
      ),
      Divider(),
      Expanded(
       child: ListView(
        children: [
         _buildCategoryItem(context,
'Indianwear', 'assets/indianwear.png'),
         _buildCategoryItem(context,
'Westernwear', 'assets/westernwear.png'),
         _buildCategoryItem(context,
'Jewellery', 'assets/jewellery.png'),
         _buildCategoryItem(context,
'Watches', 'assets/watches.png'),
         _buildCategoryItem(context,
'Footwear', 'assets/footwear.png'),
         _buildCategoryItem(context,
'Bags', 'assets/bags.png'),
        ],
       ),
      ),
     ],
    ),
    bottomNavigationBar:
BottomNavigationBar(
     currentIndex: 1,
     selectedItemColor: Colors.black,
     unselectedItemColor: Colors.grey,
     backgroundColor: Colors.white,
     showUnselectedLabels: true,
     type: BottomNavigationBarType.fixed,
     items: [
      BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
      BottomNavigationBarItem(icon:
Icon(Icons.category), label: 'Categories'),
      BottomNavigationBarItem(icon:
Icon(Icons.style), label: 'Stay Stylish'),
      BottomNavigationBarItem(icon:
Icon(Icons.account_circle), label: 'Account'),
     ],
    ),
   ),
  );
 }

 Widget _buildCategoryIcon(String label,
IconData icon) {
  return Padding(
```

```
    padding: const
EdgeInsets.symmetric(horizontal: 10),
    child: Column(
      children: [
        CircleAvatar(
          backgroundColor:
Colors.grey.shade200,
          child: Icon(icon, color: Colors.black),
        ),
        SizedBox(height: 5),
        Text(label, textAlign: TextAlign.center,
style: TextStyle(fontSize: 12)),
      ],
    ),
  );
 }

 Widget _buildCategoryItem(BuildContext
context, String label, String imagePath) {
  return ListTile(
    contentPadding:
EdgeInsets.symmetric(vertical: 10,
horizontal: 20),
    leading: Image.asset(imagePath, width:
70, height: 70),
    title: Text(
      label,
      style: TextStyle(fontSize: 15),
    ),
    trailing: Icon(Icons.arrow_forward_ios,
size: 20),
    onTap: () {
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) =>
IndianwearPage(), // Ensure
`IndianwearPage` is defined in
`product.dart`
        ),
      );
    },
  );
 }
}
```

product.dart file

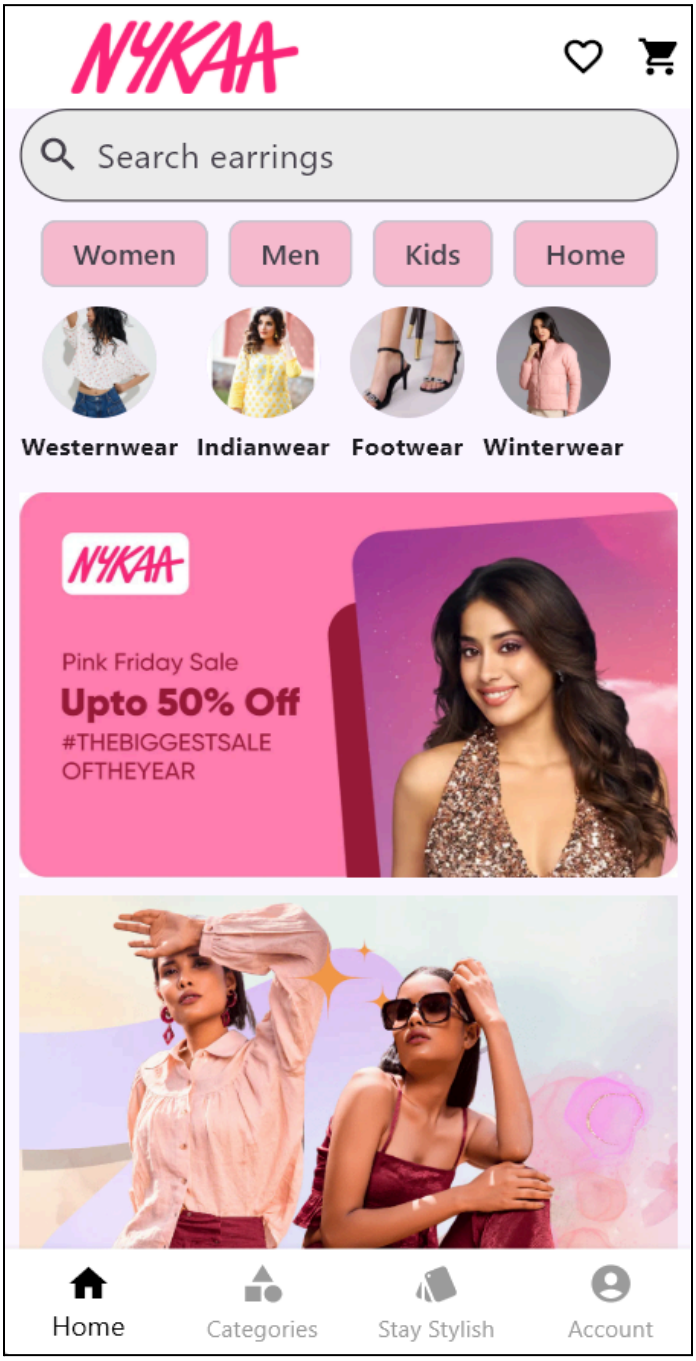
```
import 'package:flutter/material.dart';
import 'ProductDetailPage.dart';

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

final List<Map<String, String>>
productDetails = [
 {"name": "Sancia women ethnic palazzo
set", "price": "₹2,500", "image":
"assets/palazzo.png", "rating": "4.5"},
 {"name": "Indya peach embroided net
Anarkali", "price": "₹5,000", "image":
"assets/anarkali.png", "rating": "4.7"},
 {"name": "Dori pink Women wear Salwar
Suit", "price": "₹1,200", "image":
"assets/salwar.png", "rating": "4.3"},
 {"name": "Sareeka Pink net lehenga choli",
"price": "₹800", "image":
"assets/lehenga.png", "rating": "4.0"},
];

class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   debugShowCheckedModeBanner: false,
   home: IndianwearPage(),
  );
 }
}

class IndianwearPage extends
StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    backgroundColor: Colors.white,
    elevation: 0,
    title: Text(
     "Indianwear",
     style: TextStyle(color: Colors.black,
fontSize: 20),
    ),
    leading: IconButton(
     icon: Icon(Icons.arrow_back, color:
Colors.black),
     onPressed: () =>
Navigator.pop(context),
    ),
    actions: [
     IconButton(
      icon: Icon(Icons.filter_list, color:
Colors.black),
      onPressed: () {
       // Add filter functionality
      },
     ),
    ],
   ),
   body: CustomScrollView(
    slivers: [
     SliverPadding(
      padding:
EdgeInsets.symmetric(horizontal: 5, vertical:
5),
      sliver: SliverGrid(
       delegate:
SliverChildBuilderDelegate(
        (BuildContext context, int index)
{
         return ProductCard(
          name:
productDetails[index]["name"]!,
          price:
productDetails[index]["price"]!,
          image:
productDetails[index]["image"]!,
          rating:
productDetails[index]["rating"]!,
         );
        },
        childCount: productDetails.length,
       ),
       gridDelegate:
SliverGridDelegateWithFixedCrossAxisCou
nt(
        crossAxisCount: 2,
        mainAxisSpacing: 2,
        crossAxisSpacing: 2,
        childAspectRatio: 0.55,
       ),
```

```
          ),
        ),
      ],
    ),
    bottomNavigationBar:
BottomNavigationBar(
      currentIndex: 0,
      selectedItemColor: Colors.black,
      unselectedItemColor: Colors.grey,
      backgroundColor: Colors.white,
      showUnselectedLabels: true,
      type: BottomNavigationBarType.fixed,
      onTap: (index) {
        // Add navigation functionality if
needed
      },
      items: [
        BottomNavigationBarItem(icon:
Icon(Icons.home), label: 'Home'),
        BottomNavigationBarItem(icon:
Icon(Icons.category), label: 'Categories'),
        BottomNavigationBarItem(icon:
Icon(Icons.style), label: 'Stay Stylish'),
        BottomNavigationBarItem(icon:
Icon(Icons.account_circle), label: 'Account'),
      ],
    ),
  );
 }
}

class ProductCard extends StatelessWidget
{
 final String name;
 final String price;
 final String image;
 final String rating;

 const ProductCard({
  required this.name,
  required this.price,
  required this.image,
  required this.rating,
 });

 @override
 Widget build(BuildContext context) {
  return GestureDetector(
    onTap: () {
      // Navigate to the ProductDetailPage
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) =>
ProductDetailPage(
            name: name,
            price: price,
            image: image,
            rating: rating,
          ),
        ),
      );
    },
    child: Card(
      elevation: 3,
      shape:
RoundedRectangleBorder(borderRadius:
BorderRadius.circular(10)),
      child: Column(
        crossAxisAlignment:
CrossAxisAlignment.start,
        children: [
          // Product Image with Heart Icon &
Quick View Button
          Stack(
            children: [
              ClipRRect(
                borderRadius:
BorderRadius.vertical(top:
Radius.circular(10)),
                child: Image.asset(
                  image,
                  width: double.infinity,
                  height: 220,  // Increased image
height
                  fit: BoxFit.cover,
                ),
              ),
              // Heart Icon
              Positioned(
                top: 8,
                right: 8,
                child: Container(
                  decoration: BoxDecoration(
                    color:
Colors.white.withOpacity(0.7),
                    shape: BoxShape.circle,
                  ),
```

```
                child: IconButton(
                  icon:
Icon(Icons.favorite_border, color:
Colors.black),
                  onPressed: () {
                    // Add to wishlist functionality
                  },
                ),
              ),
            ),
            // Quick View Button
            Positioned(
              bottom: 8,
              left: 8,
              child: Container(
                padding:
EdgeInsets.symmetric(vertical: 3, horizontal:
10),
                decoration: BoxDecoration(
                  color:
Colors.white.withOpacity(0.8),
                  borderRadius:
BorderRadius.circular(5),
                ),
                child: Text(
                  "Quick View",
                  style: TextStyle(fontSize: 12,
fontWeight: FontWeight.bold),
                ),
              ),
            ),
          ],
        ),
        // Wrap the Column with a Flexible
widget to allow ListView
        Flexible(
          fit: FlexFit.tight,
          child: ListView(
            padding: EdgeInsets.all(8),
            shrinkWrap: true,  // Ensures
ListView doesn't take more space than
needed
            physics:
NeverScrollableScrollPhysics(),  // Disables
scrolling inside the ListView
            children: [
              Text(name, style:
TextStyle(fontSize: 10, fontFamily:
"Poppins")),
              SizedBox(height: 4),
              Row(
                children: [
                  Icon(Icons.star, color:
Colors.orange, size: 14),
                  SizedBox(width: 4),
                  Text(rating, style:
TextStyle(fontSize: 10, color:
Colors.black87)),
                ],
              ),
              SizedBox(height: 4),
              Text(price, style:
TextStyle(fontSize: 11, color: Colors.black,
fontWeight: FontWeight.bold, fontFamily:
"Poppins")),
            ],
          ),
        ],
      ),
    ),
  );
 }
}
```

Output:

After clicking category in bottom navbar it will navigate to category.dart page
And
After clicking shopping cart icon in topbar it will redirect to bag.dart page



Bag

Wishlist

Flat discount on UPI payment

Check Delivery Date | Enter Pincode

U.S. POLO ASSN.

Light Blue High Rise Pleated Wide Leg...

Size 28

₹1820 ~~₹2799~~ 35% off

7 day return

Aldo

Iconispeca-In009 Women Black Sneake...

Size UK 3

₹0 ~~₹0~~

7 day return

You will save ₹6,978 on this purchase.

₹7,849

View Details

Proceed to Buy

Categories

Women | Men | Kids | Home

Sale 50% Off | Trending | New Arrivals | Exclusive Brand

Indianwear

Westernwear

Jewellery

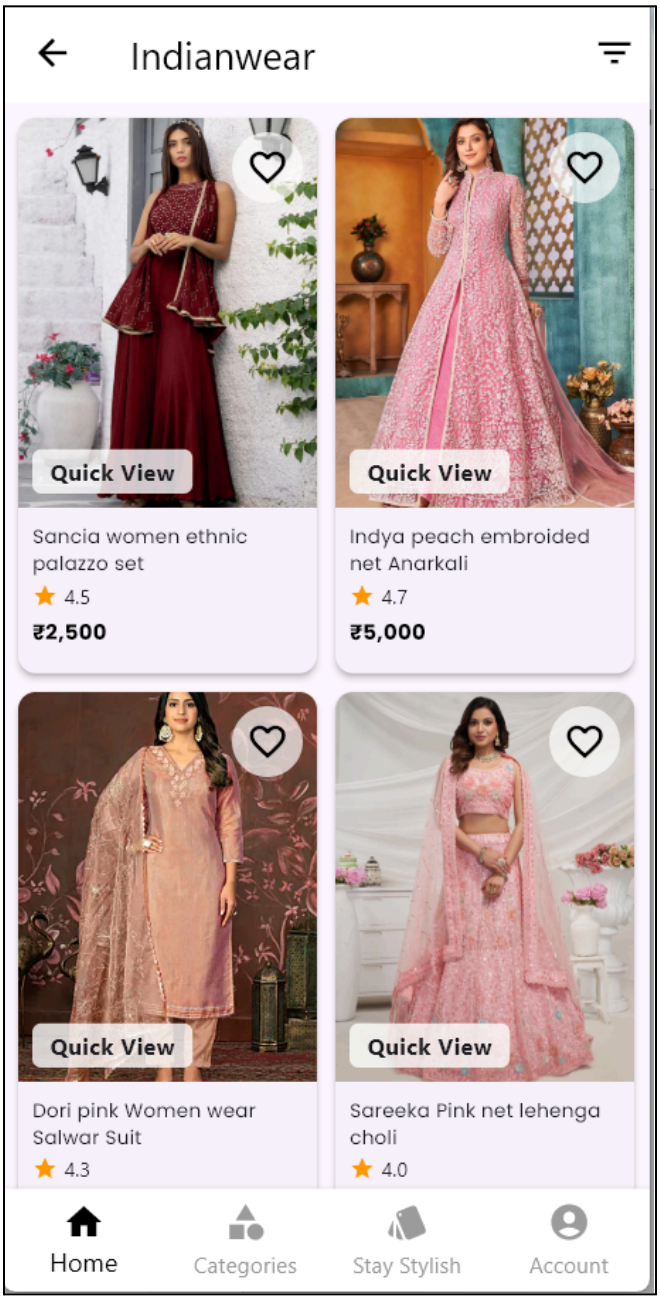
Watches

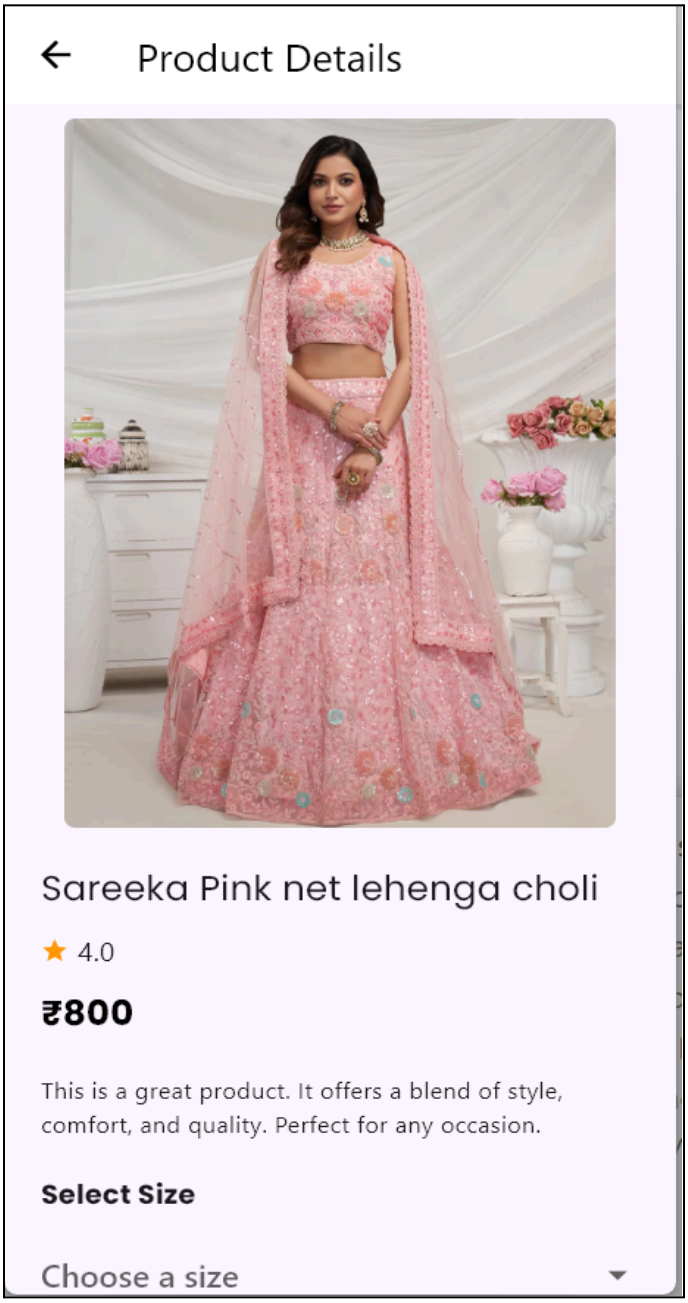
Footwear

Bags

Home | Categories | Stay Stylish | Account

After clicking on Indian wear from above categories page it will redirect to product.dart page
And
After clicking on product card it will redirect to ProductDetailPage.dart page



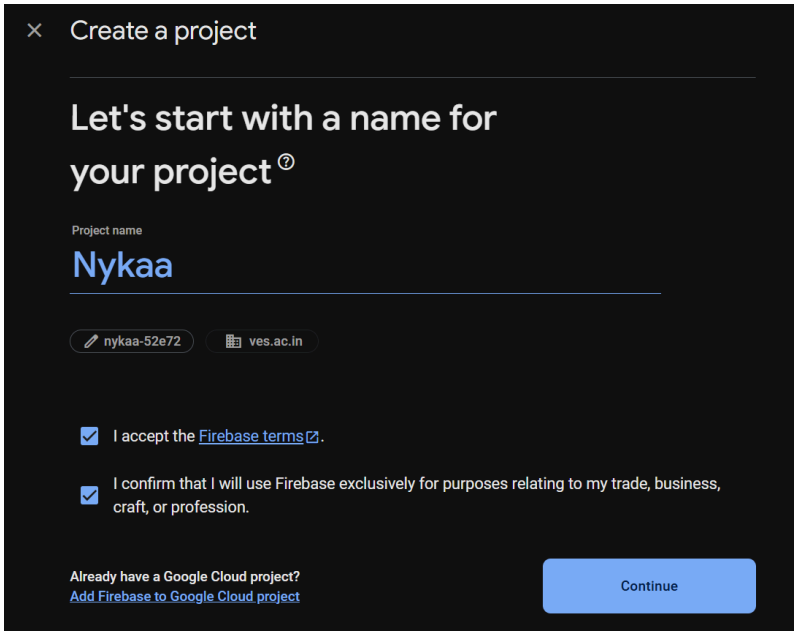


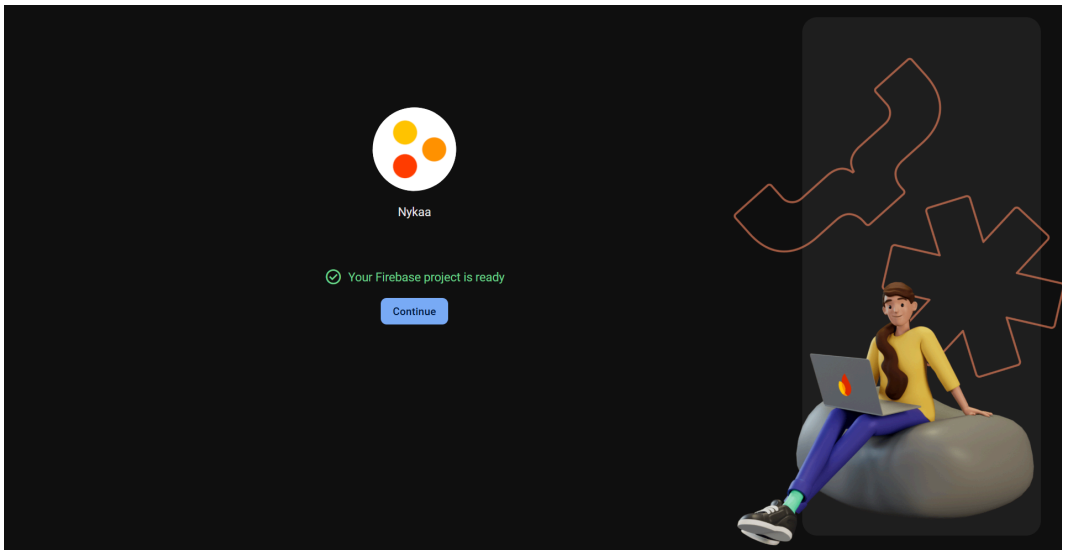
MAD & PWA Lab
Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	20
Name	Madhura Jangale
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:	

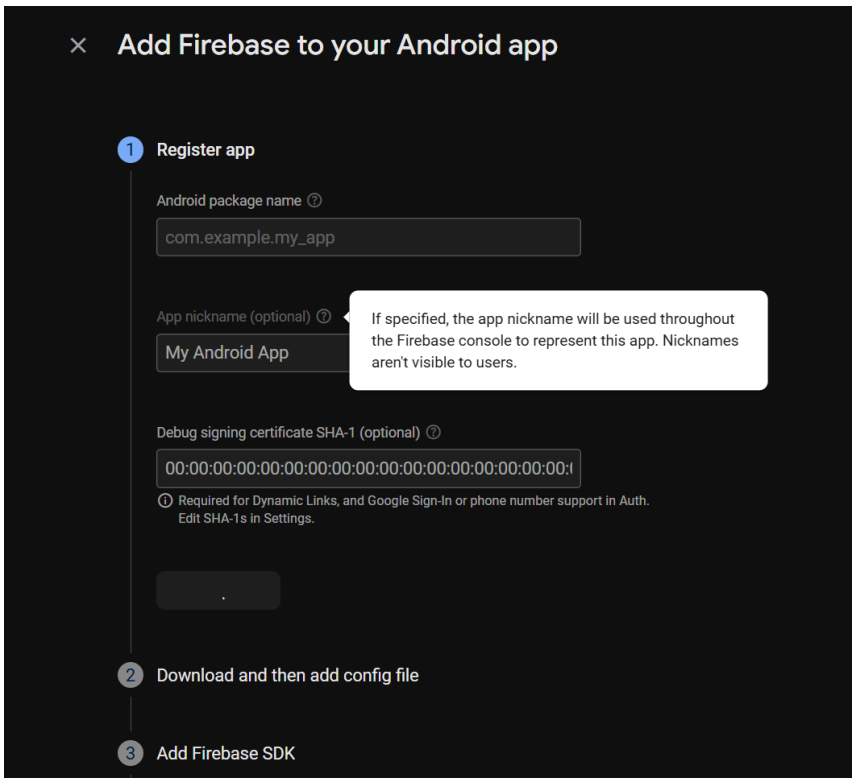
Creating a New Firebase Project

First, log in with your Google account to manage your Firebase projects. From within the Firebase dashboard, select the Create new project button and give it a name





In order to add Android support to our Flutter application, select the Android logo from the dashboard. This brings us to the following screen:



The most important thing here is to match up the Android package name that you choose here with the one inside of our application.

Downloading the Config File

The next step is to add the Firebase configuration file into our Flutter project. This is important as it contains the API keys and other critical information for Firebase to use.

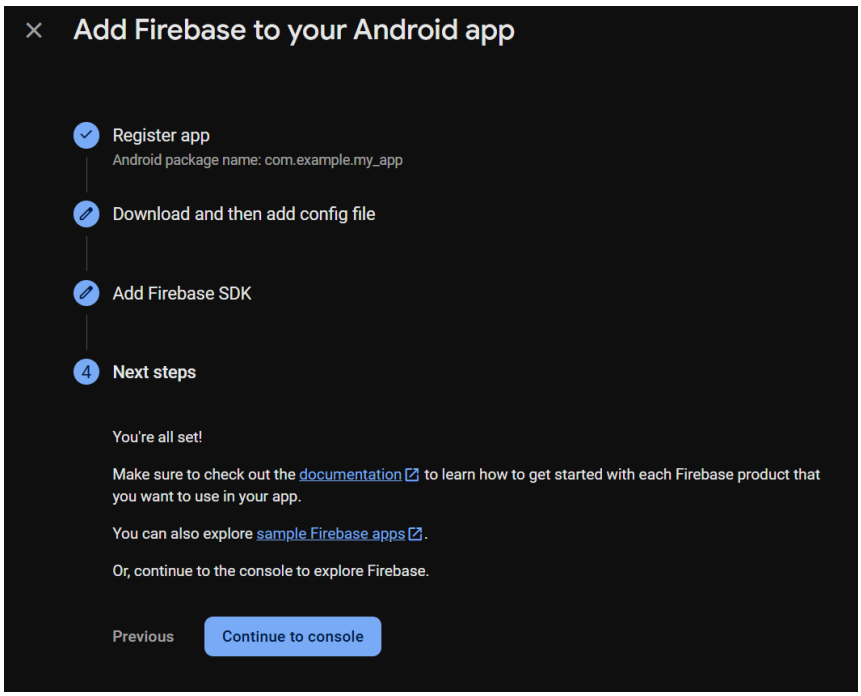
Select Download `google-services.json` from this page

In android/build.gradle add-

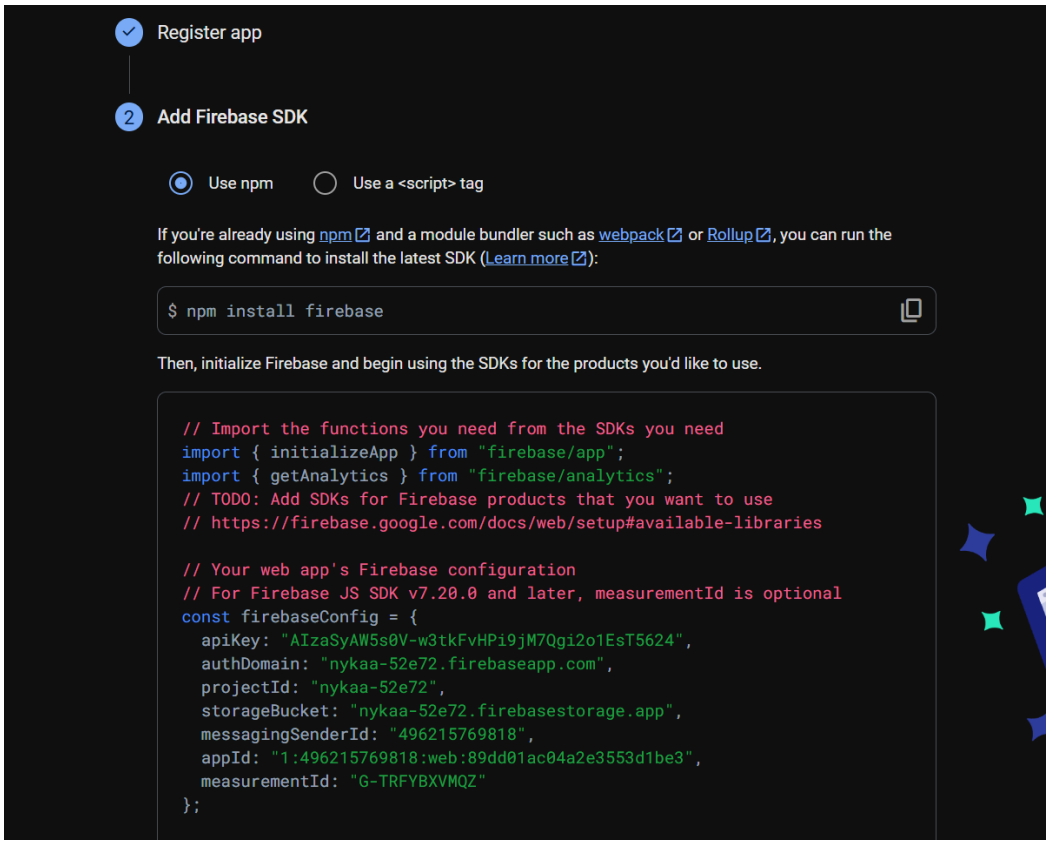
```
dependencies {
  classpath 'com.google.gms.google-services:4.4.2'
}
```

In app/build.gradle add-

```
dependencies{
  implementation platform('com.google.firebase:firebase-bom:33.9.0')
}
```



Lets setup web app-



In pubspec.yaml:

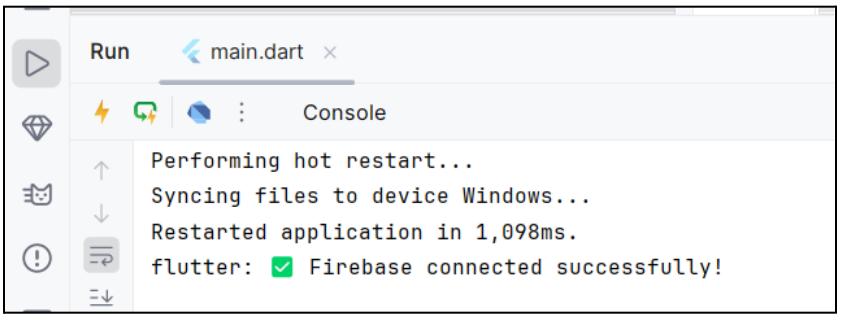
dependencies:
  flutter:
    sdk: flutter

```
  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.8
  firebase_core: ^3.11.0
  firebase_auth: ^5.4.2
  cloud_firestore: ^5.6.3
  firebase_storage: ^12.4.2
  firebase_messaging: ^15.2.2
```

Firebase connection in code:

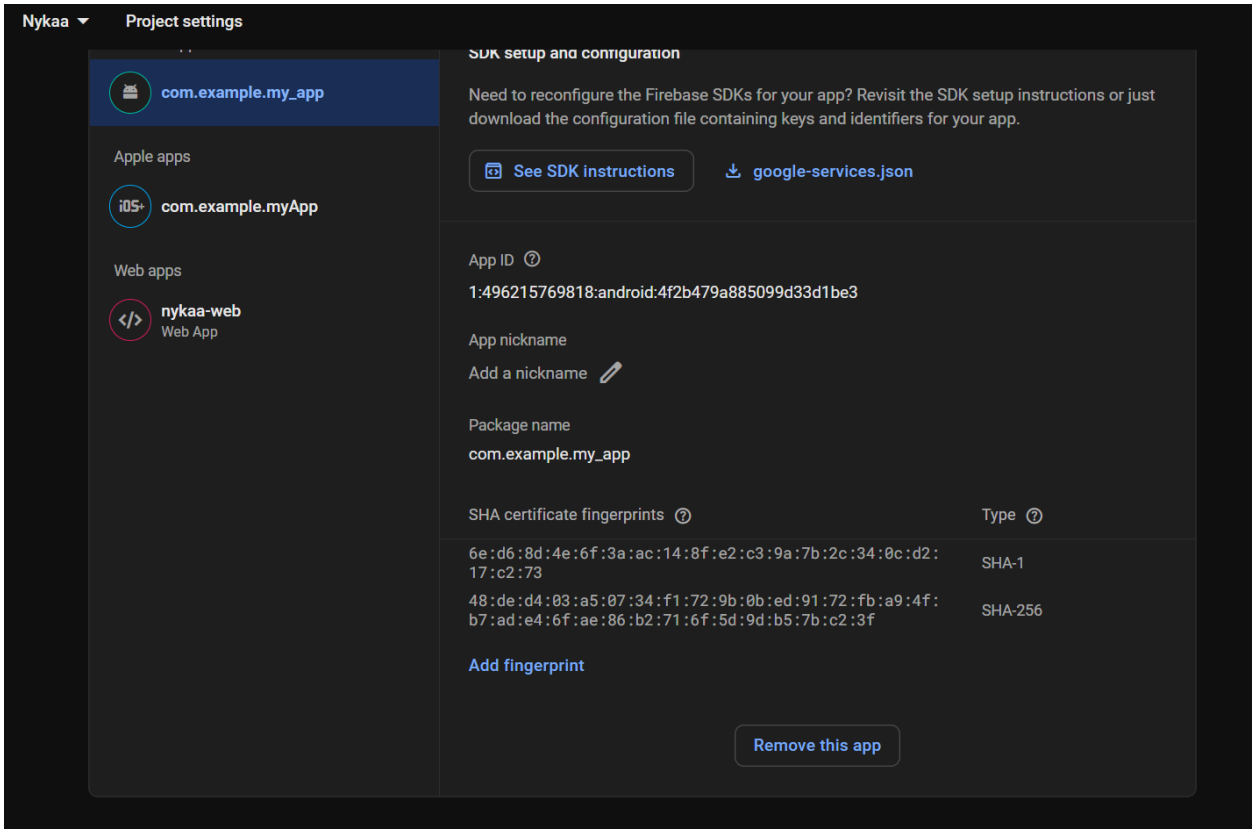
```
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'home.dart';

void main() async {
 WidgetsFlutterBinding.ensureInitialized();

 try {
   await Firebase.initializeApp(
      options: const FirebaseOptions(
        apiKey: "AIzaSyAW5s0V-w3tkFvHPi9jM7Qgi2o1EsT5624",
        authDomain: "nykaa-52e72.firebaseapp.com",
        projectId: "nykaa-52e72",
        storageBucket: "nykaa-52e72.firebasestorage.app",
        messagingSenderId: "496215769818",
        appId: "1:496215769818:web:89dd01ac04a2e3553d1be3",
        measurementId: "G-TRFYBXVMQZ"));
   print("✅ Firebase connected successfully!");
 } catch (e) {
   print("❌ Firebase initialization failed: $e");
 }

 runApp(MyApp());
}
```

Output:

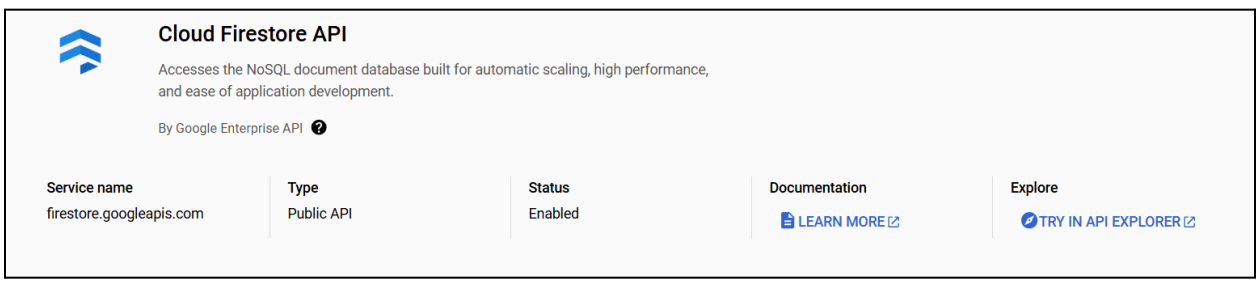


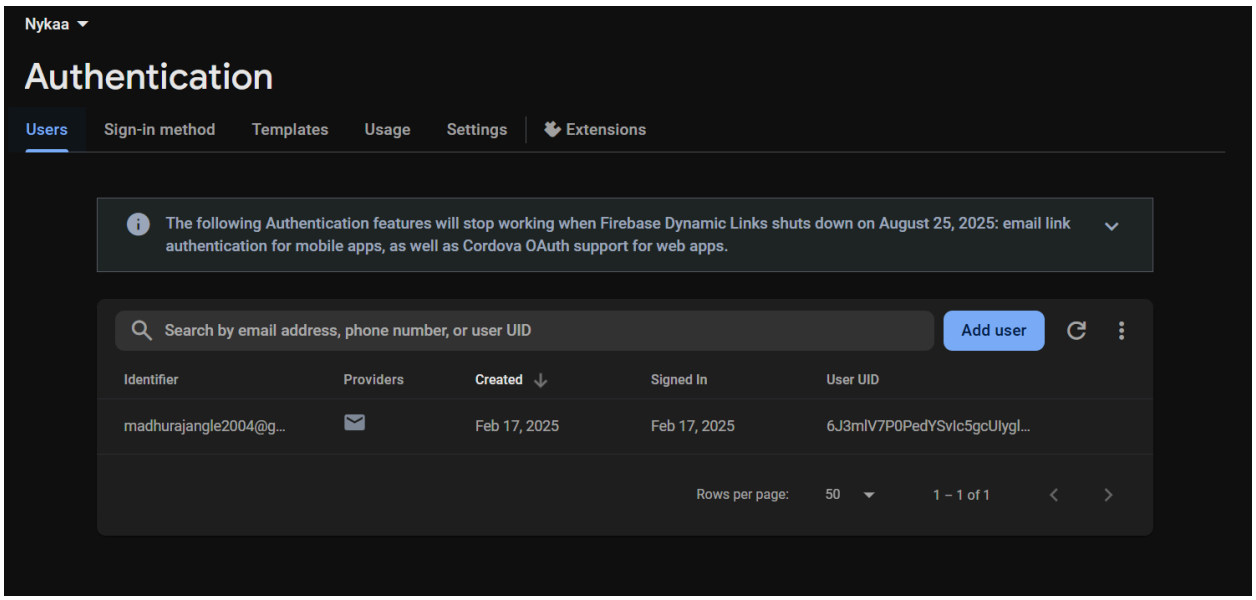
Generate SHA key for authentication-

```
C:\Users\MADHURA>keytool -list -v -keystore %USERPROFILE%\.android\debug.keystore -alias androiddebugkey -storepass andr
oid
Alias name: androiddebugkey
Creation date: 26-Jan-2025
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: C=US, O=Android, CN=Android Debug
Issuer: C=US, O=Android, CN=Android Debug
Serial number: 1
Valid from: Sun Jan 26 23:42:14 IST 2025 until: Tue Jan 19 23:42:14 IST 2055
Certificate fingerprints:
         SHA1: 6E:D6:8D:4E:6F:3A:AC:14:8F:E2:C3:9A:7B:2C:34:0C:D2:17:C2:73
         SHA256: 48:DE:D4:03:A5:07:34:F1:72:9B:0B:ED:91:72:FB:A9:4F:B7:AD:E4:6F:AE:86:B2:71:6F:5D:9D:B5:7B:C2:3F
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 1

C:\Users\MADHURA>
```



Enable the firestore API-




Nykaa
Authentication
Users
Sign-in method
Templates
Usage
Settings
Extensions
The following Authentication features will stop working when Firebase Dynamic Links shuts down on August 25, 2025: email link authentication for mobile apps, as well as Cordova OAuth support for web apps.
Search by email address, phone number, or user UID
Add user
Identifier
Providers
Created
Signed In
User UID
madhurajangle2004@g...
Feb 17, 2025
Feb 17, 2025
6J3mlV7P0PedYSvlc5gcUlygl...
Rows per page:
50
1 – 1 of 1

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.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	

Aim:

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

Theory:

A Progressive Web Application (PWA) is a type of web application that offers an enhanced user experience similar to native mobile apps. It is built using standard web technologies (HTML, CSS, JavaScript) but incorporates modern features like offline functionality, push notifications, and home screen installation.

PWAs aim to bridge the gap between traditional websites and mobile apps by providing fast, reliable, and engaging experiences across all devices.

Evolution of Web Apps to PWAs-

Web applications have evolved significantly over the years:

- Traditional Websites – Early websites were static and lacked interactivity.
- Dynamic Web Apps – Introduced AJAX and JavaScript for better user interactions.
- Single Page Applications (SPA) – Frameworks like React, Angular, and Vue improved performance but still relied on internet connectivity.
- Progressive Web Apps (PWA) – Introduced offline functionality, app-like experiences, and better performance, making them competitive with native apps.

Core Principles of PWA-

PWAs follow specific principles to ensure a high-quality experience:

1. Progressive – Works for all users, regardless of the browser or device.
2. Responsive – Adapts to various screen sizes and device types.
3. Connectivity Independent – Functions even with limited or no internet access.
4. App-like Feel – Provides an immersive experience similar to native apps.

5. Secure – Uses HTTPS to protect data and user privacy.
6. Discoverable – Indexed by search engines, making them easy to find.
7. Re-engageable – Supports push notifications to keep users engaged.
8. Installable – Users can add the app to their home screen without an app store.
9. Linkable – Easily shared via a URL without requiring installation.

Key Technologies Behind PWAs-

PWAs rely on modern web technologies to deliver enhanced functionality:

a) Service Workers

- A background script that enables caching, push notifications, and offline functionality.
- It helps store web assets so that users can access content even without the internet.

b) Web App Manifest

- A JSON file that defines the app's metadata, such as name, icon, theme color, and display mode.
- Enables home screen installation and native-like appearance.

c) HTTPS (Secure Context)

- Ensures data privacy and security by encrypting communication between the app and server.
- Required for service workers and push notifications.

d) Responsive Design

- Uses CSS media queries and flexible layouts to ensure compatibility across different devices and screen sizes.

Benefits of PWAs-

a) Improved Performance

- PWAs load faster due to caching and preloading mechanisms.
- Minimizes server requests by storing data locally.

b) Works Offline

- Cached content allows users to browse even without an internet connection.
- Useful for news websites, e-commerce platforms, and social media apps.

c) No App Store Dependency

- PWAs do not require approval from app stores like Google Play or Apple App Store.
- Users can install PWAs directly from the browser, reducing friction in the adoption process.

d) Cost-Effective

- A single PWA works across multiple platforms, reducing development and maintenance costs.
- Eliminates the need for separate native apps for iOS and Android.

e) Increased User Engagement

- Features like push notifications help re-engage users and improve retention rates.
- Users can receive updates and alerts even when the app is not open.

f) SEO-Friendly

- Unlike native apps, PWAs are indexed by search engines, making them discoverable.
- Better ranking potential in Google Search results.

Implementation:

Step 1: Add a link to the file named manifest.json in index.html.

```
<link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
```

Step 2: Create a manifest.json file in the same directory. This file basically contains information about the web application. Some basic information includes the application name,

starting URL, theme color, and icons. All the information required is specified in the JSON format. The source and size of the icons are also defined in this file.

```
{
  "short_name": "React App",
  "name": "Create React App Sample",
  "icons": [
    {
      "src": "faviconn.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logoo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logoo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

Conclusion:-
Hence, we learnt how to write a metadata of our 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 PWA.
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	

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

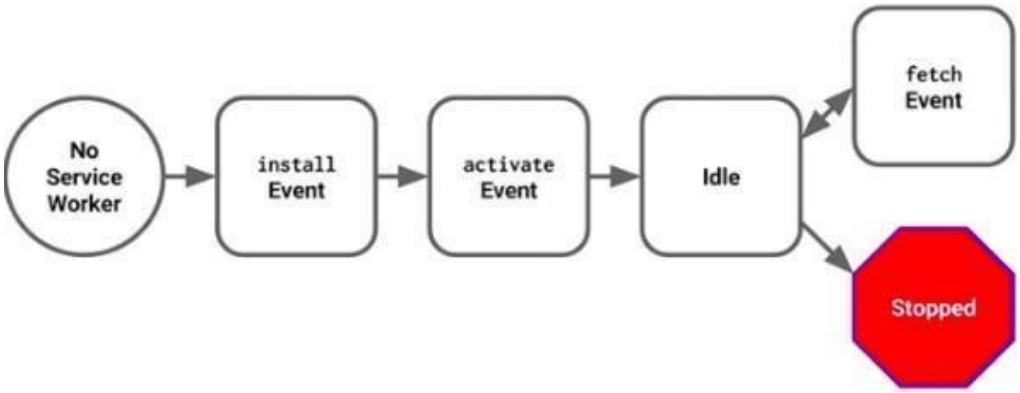
- You can manage **Push Notifications**
  You can manage push notifications with Service Worker and show any information message to the user.
- You can **Continue**
  Although Internet connection is broken, you can start any process with Background Sync of Service Worker.

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

```
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.

```
// Service Worker Script

const CACHE_NAME = 'blogbreeze-v1'; // Cache name to identify the version of cached content
const urlsToCache = [
'/', // Home page
'index.html', // Main HTML file
'ani.html', // Any additional pages you want to cache
'/src/assets/react.svg', // App icon
'/public/icon.png', // Favicon
'/src/main.jsx', // Main JS file for app functionality
//  'https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css' // FontAwesome for icons
];

// Install Service Worker
self.addEventListener('install', (event) => {
console.log('Service Worker: Installed');
event.waitUntil(
caches.open(CACHE_NAME)
.then((cache) => {
return cache.addAll(urlsToCache);
})
);
});

// Activate Service Worker
self.addEventListener('activate', (event) => {
console.log('Service Worker: Activated');
```

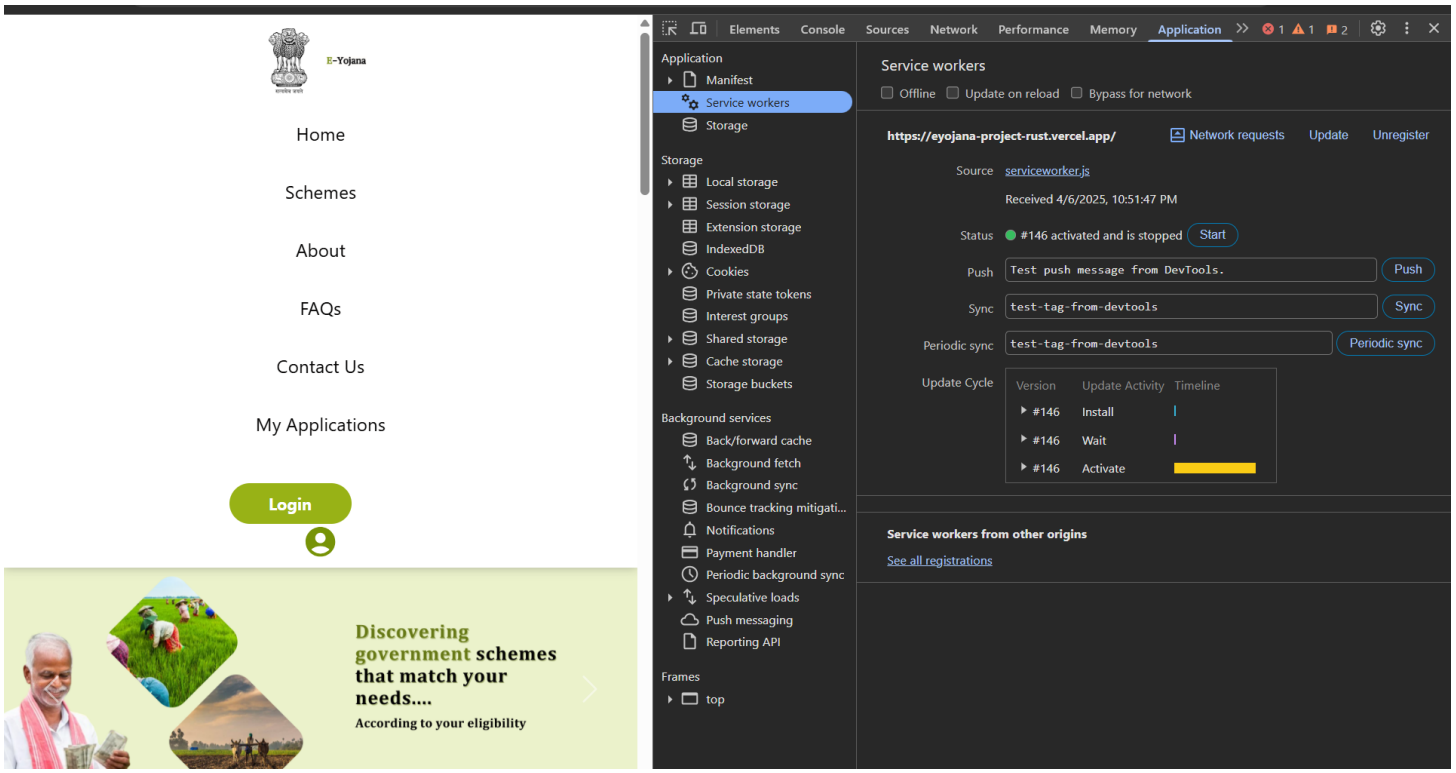
```
// Remove old caches if there are any
event.waitUntil( caches.keys().then((cacheNames)
=> {
return Promise.all( cacheNames.map((cacheName)
=> {
if (cacheName !== CACHE_NAME) { return
caches.delete(cacheName);
}
})
);
})
);
});

// Fetch event: Intercept network requests and serve cached content
self.addEventListener('fetch', (event) => {
console.log('Service Worker: Fetching', event.request.url);
event.respondWith(
caches.match(event.request)


.then((cachedResponse) => {
// Return cached content if found, otherwise fetch from network return
cachedResponse || fetch(event.request);
})
);
})
```

Code:

We will need to create another file for the PWA, that is, serviceworker.js in the same directory. This file handles the configuration of a service worker that will manage the working of the application.

```
var staticCacheName = "eyojana";

self.addEventListener("install", function (e) {
e.waitUntil(
    caches.open(staticCacheName).then(function (cache) {
    return cache.addAll(["/"]);
    })
);
});
self.addEventListener("fetch", function (event) {
console.log(event.request.url);

event.respondWith(
```

```
    caches.match(event.request).then(function (response) {
    return response || fetch(event.request);
    })
);
});
```



Conclusion : Thus we have learnt to code and register a service worker, and complete the install and activation process for a new service.

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.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

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

SERVICE WORKER FOR PUSH NOTIFICATIONS :

```
const CACHE_NAME = 'blogbreeze-v1';
const filesToCache = [
  '/',
  'index.html',
  'manifest.json',
  '/src/assets/react.svg',
  '/public/icon.png',
  '/src/index.css',
  '/src/main.jsx',
```

```
];

// Install event: Caching important files
self.addEventListener('install', (event) => {
console.log('Service Worker: Installed');
event.waitUntil(
    caches.open(CACHE_NAME)
      .then((cache) => {
        console.log('Service Worker: Caching files');
        return cache.addAll(filesToCache);
      })
  );
});

// Activate event: Cleaning up old caches
self.addEventListener('activate', (event) => {
console.log('Service Worker: Activated');
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: Intercepting requests and serving from cache if available
```

```
self.addEventListener('fetch', (event) => {
  console.log('Service Worker: Fetching', event.request.url);

  event.respondWith(
    caches.match(event.request)
      .then((cachedResponse) => {
        if (cachedResponse) {
          console.log('Service Worker: Returning cached response');
          return cachedResponse; // Return the cached response if available
        }

        return fetch(event.request)
          .then((networkResponse) => {
            caches.open(CACHE_NAME).then((cache) => {
              console.log('Service Worker: Caching new response for', event.request.url);
              cache.put(event.request, networkResponse.clone()); // Cache the new response
            });
            return networkResponse;
          });
      })
  );
});

self.addEventListener('push', (event) => {
    console.log('Push notification received:', event);

    if (event && event.data) {
      // Parse the push notification data
      const data = event.data.json();

      if (data.method === 'pushMessage') {
        console.log('Push notification sent with message:', data.message);
```

```
    // Set up notification options (like body, icon, etc.)
    const options = {
      body: data.message || 'Hello, this is a default message!', // Default or push message
      icon: '/src/assets/react.svg', // Icon for the notification
      badge: '/src/assets/react.svg', // Badge icon for the notification
    };

    // Check if the notification permission is granted before showing the notification
    if (Notification.permission === 'granted') {
      // Show the notification with a title
      event.waitUntil(
        self.registration.showNotification('Blog Breeze', options)
      );
    } else {
      console.log('Notification permission not granted yet.');
    }
  }
 }
});
```

SERVICE WORKER SCRIPT FOR SYNC AND FETCH :

```
const CACHE_NAME = 'blogbreeze-v1';
const filesToCache = [
  '/',
  'index.html',
  'manifest.json',
  '/src/assets/react.svg',
  '/public/icon.png',
  '/src/index.css',
```

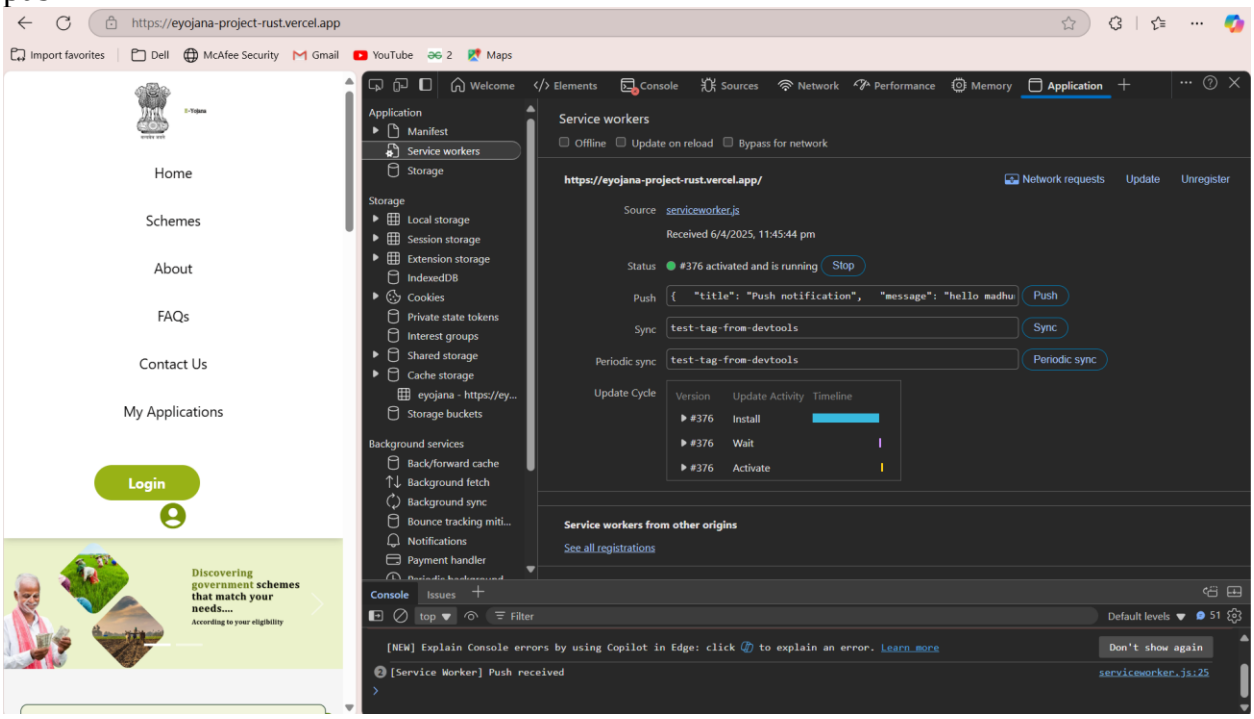
```
  '/src/main.jsx',

];

// Install event: Caching important files
self.addEventListener('install', (event) => {
console.log('Service Worker: Installed');
event.waitUntil(
    caches.open(CACHE_NAME)
      .then((cache) => {
        console.log('Service Worker: Caching files');
        return cache.addAll(filesToCache);
      })
  );
});

// Activate event: Cleaning up old caches
self.addEventListener('activate', (event) => {
console.log('Service Worker: Activated');
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: Intercepting requests and serving from cache if available
self.addEventListener('fetch', function (event) {
  console.log('Service Worker: Fetching', event.request.url);

  // Skip caching for Firebase API requests
  if (event.request.url.includes('firestore.googleapis.com')) {
    // Always fetch Firebase data from the network (no cache)
    event.respondWith(fetch(event.request));
    return;
  }

  event.respondWith(
    checkResponse(event.request)
      .catch(function () {
        console.log('Fetch from cache successful!');
        return returnFromCache(event.request);
      })
  );

  console.log('Fetch successful!');
  event.waitUntil(addToCache(event.request));
});

// Sync event: Handling background sync
self.addEventListener('sync', function (event) {
if (event.tag === 'syncMessage') {
    console.log('Sync successful!');
    // You can add more background sync logic here
  }
});
```

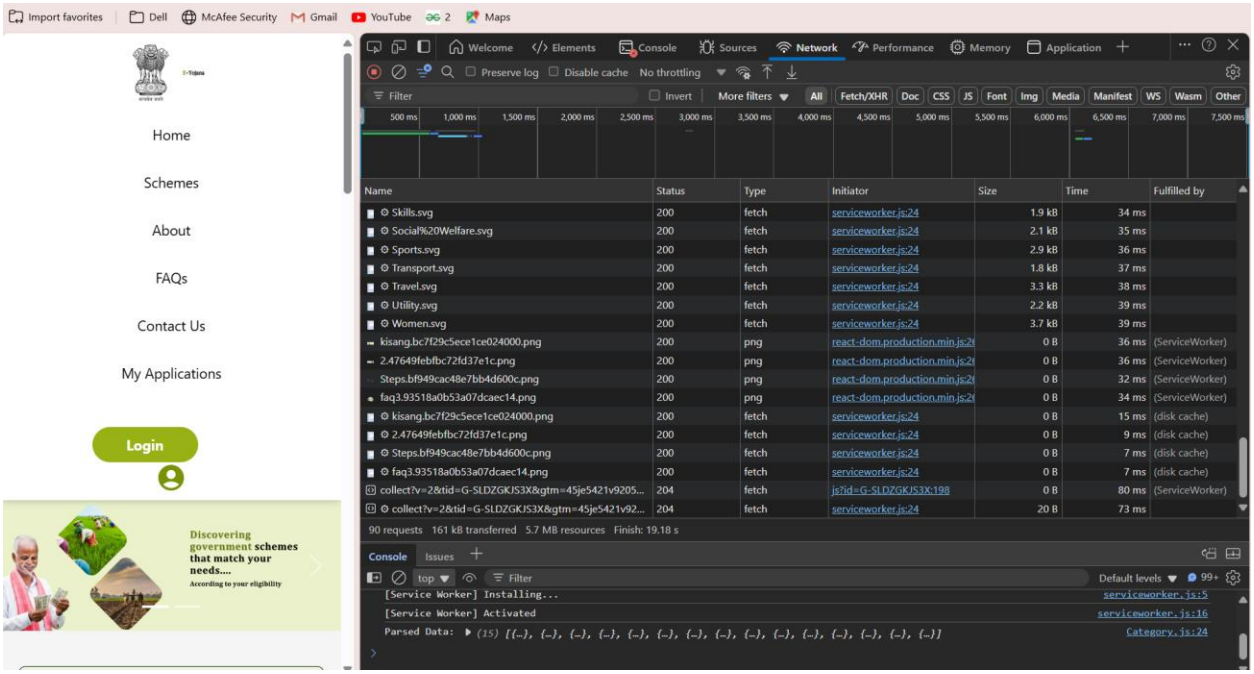
```
// Placeholder functions for cache and response handling
function checkResponse(request) {
  return caches.match(request)
    .then(function (cachedResponse) {
      if (cachedResponse) {
        return cachedResponse;
      }
      return fetch(request);
    });
}

function returnFromCache(request) {
  return caches.match(request);
}

function addToCache(request) {
  return fetch(request)
    .then(function (response) {
      if (!response || response.status !== 200 || response.type !== 'basic') {
        return response;
      }
      return caches.open(CACHE_NAME)
        .then(function (cache) {
          console.log('Service Worker: Adding to cache', request.url);
          cache.put(request, response);
          return response;
        });
    });
}
```

OUTPUT :

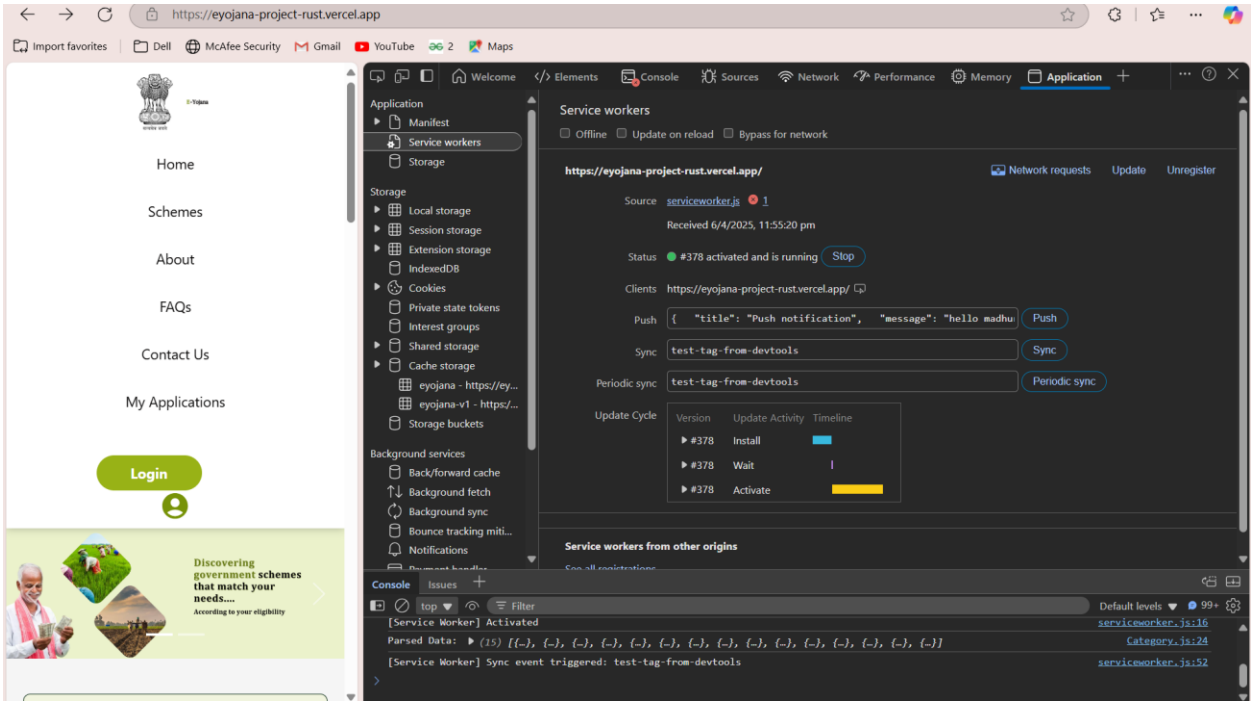
push



fetch



sync



Conclusion : Thus we learnt to implement 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.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Aim: To study and implement deployment of 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

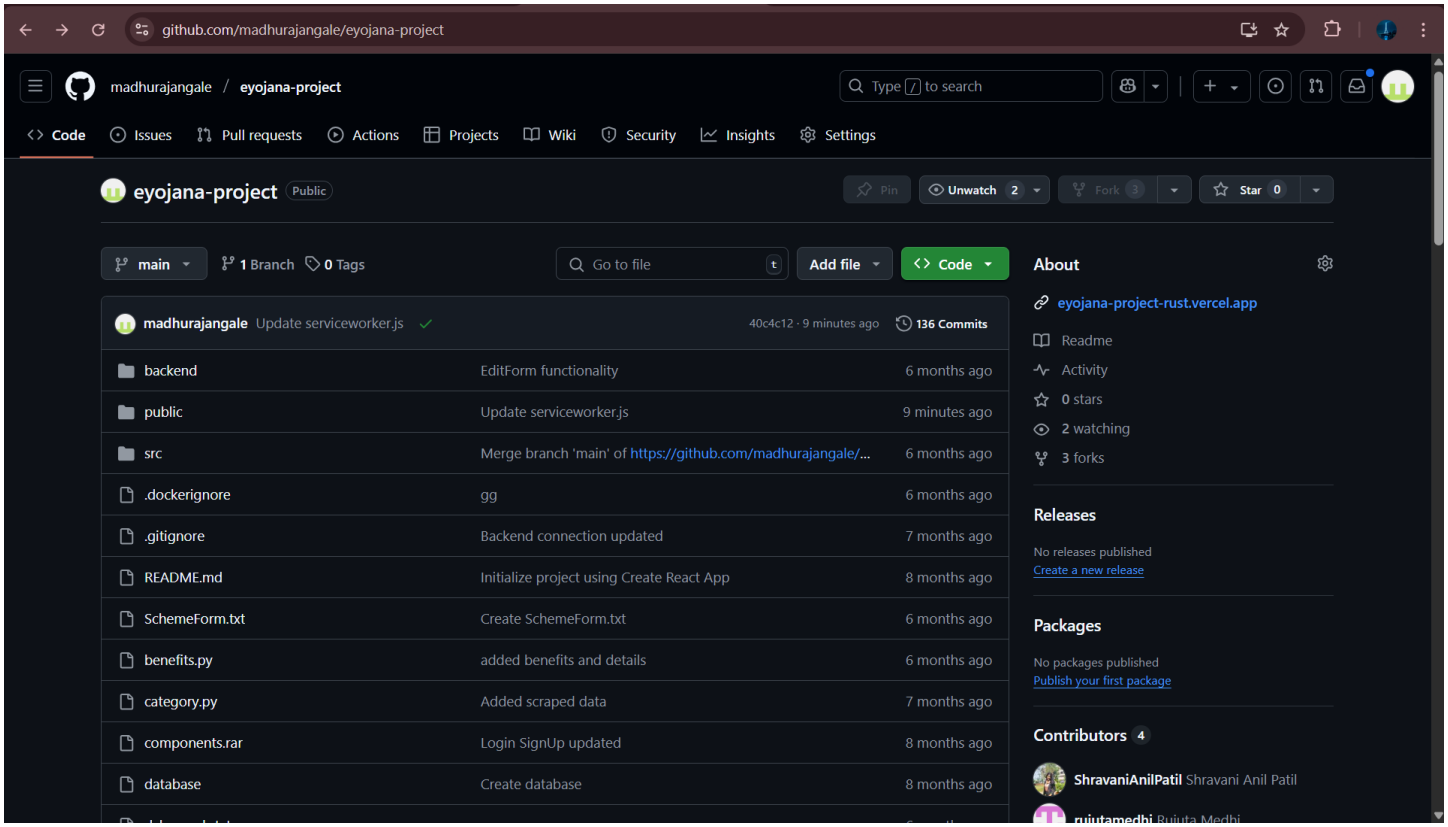
1. Hosted by Google. Enough said.
2. Authentication, Cloud Messaging, and a whole lot of other handy services will be available to you.
3. A real-time database will be available to you, which can store 1 GB of data.
4. You'll also have access to a blob store, which can store another 1 GB of data.
5.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

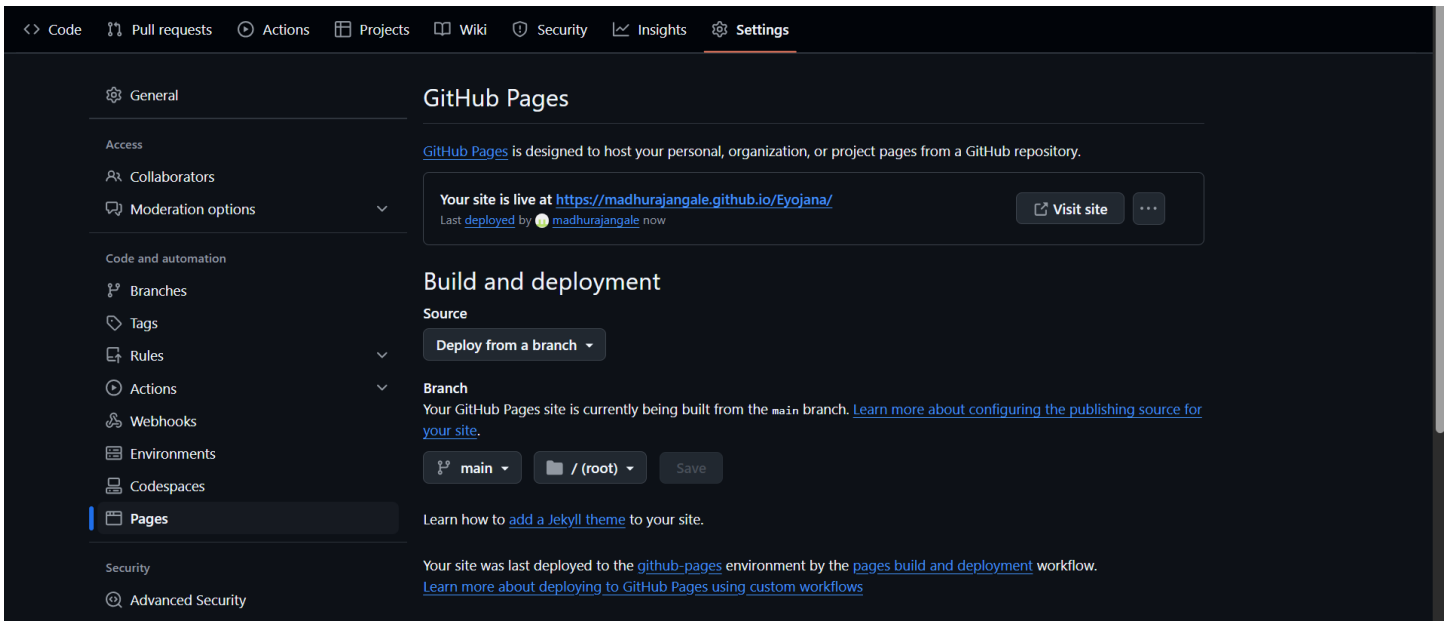
Cons

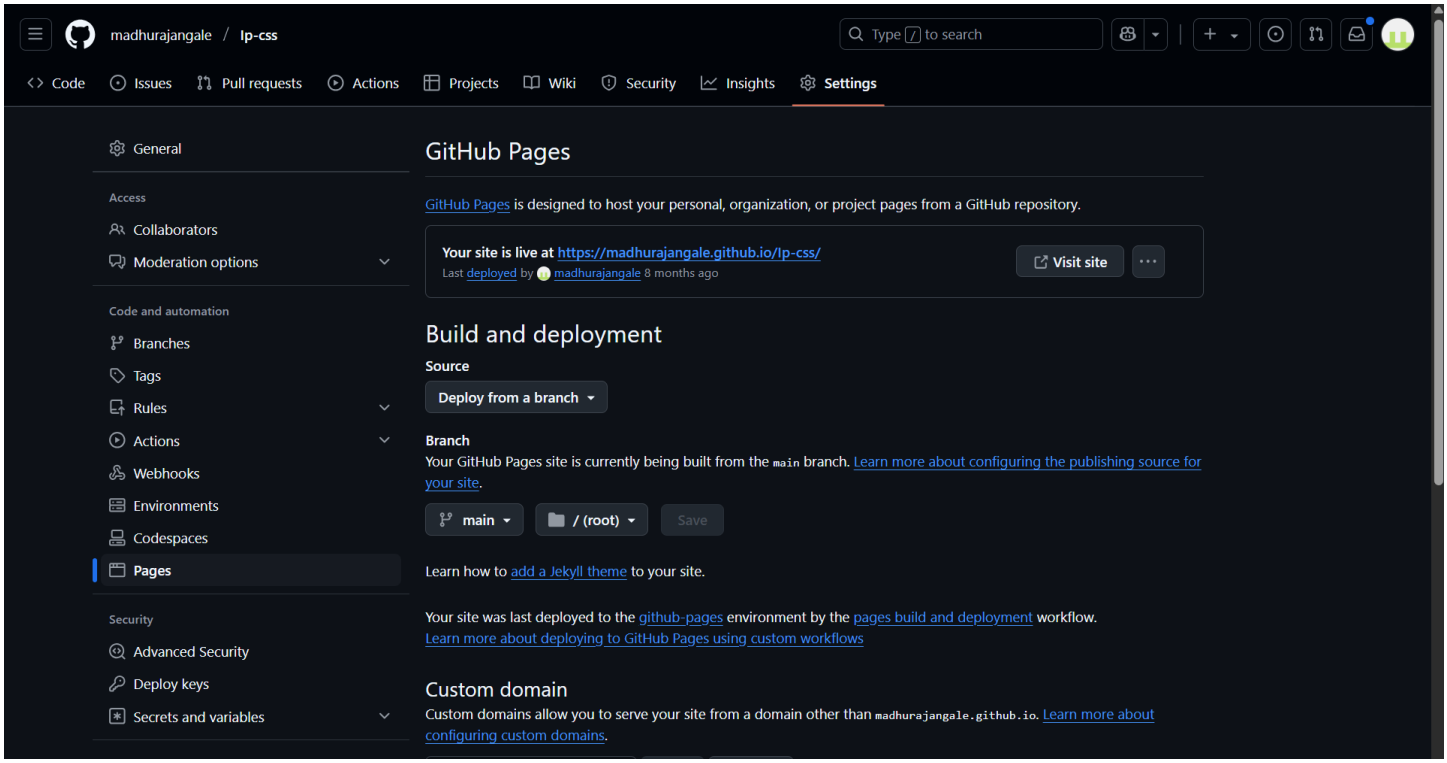
1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.
3. No in-built support for any static site generator.

Link to our GitHub repository: https://github.com/madhurajangale/eyojana-project/

Github Screenshot:







Deployed link: https://madhurajangale.github.io/lp-css/

MAD & PWA Lab
Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	20
Name	Madhura Jangale
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

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

Theory :

Reference : https://www.semrush.com/blog/google-lighthouse/

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

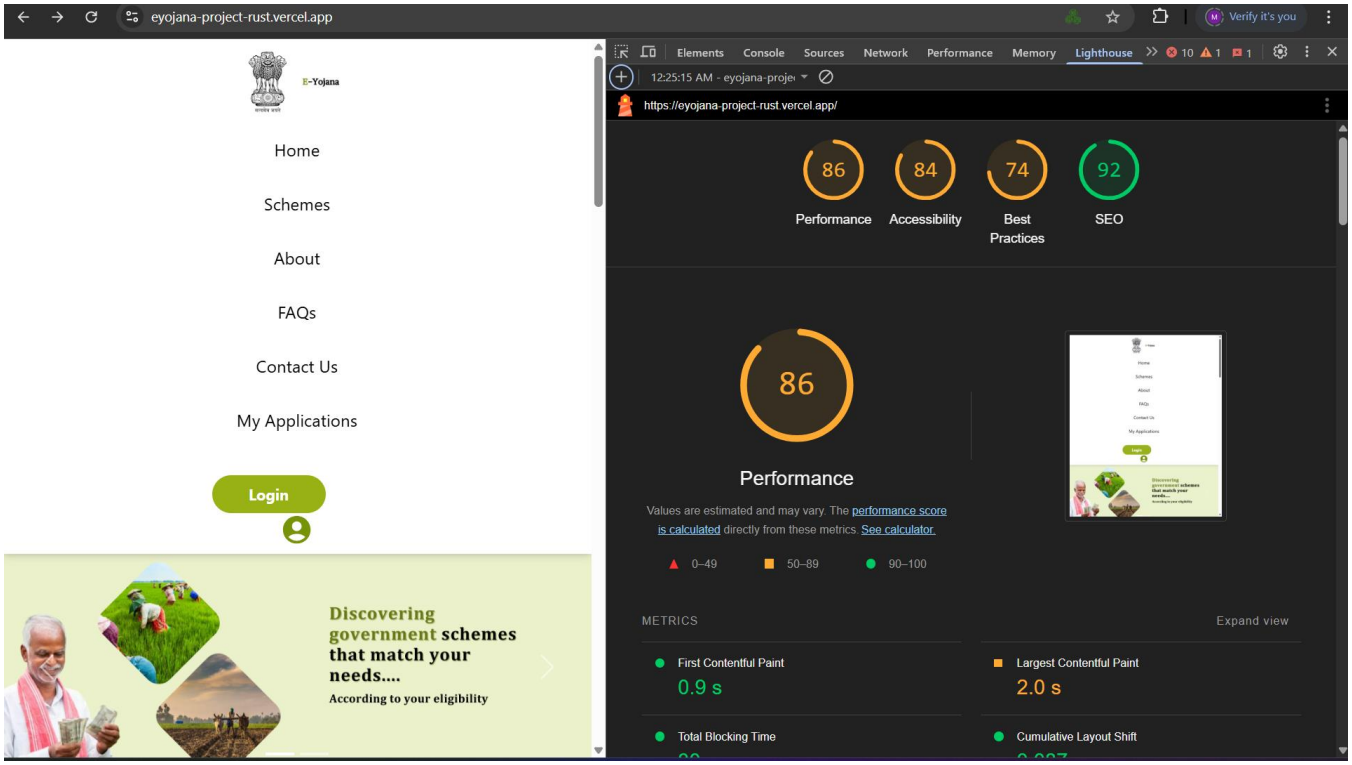
Google Lighthouse has the option of running the Audit for Desktop as well as mobile version of your page(s). The top metrics that will be measured in the Audit are:

1. **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

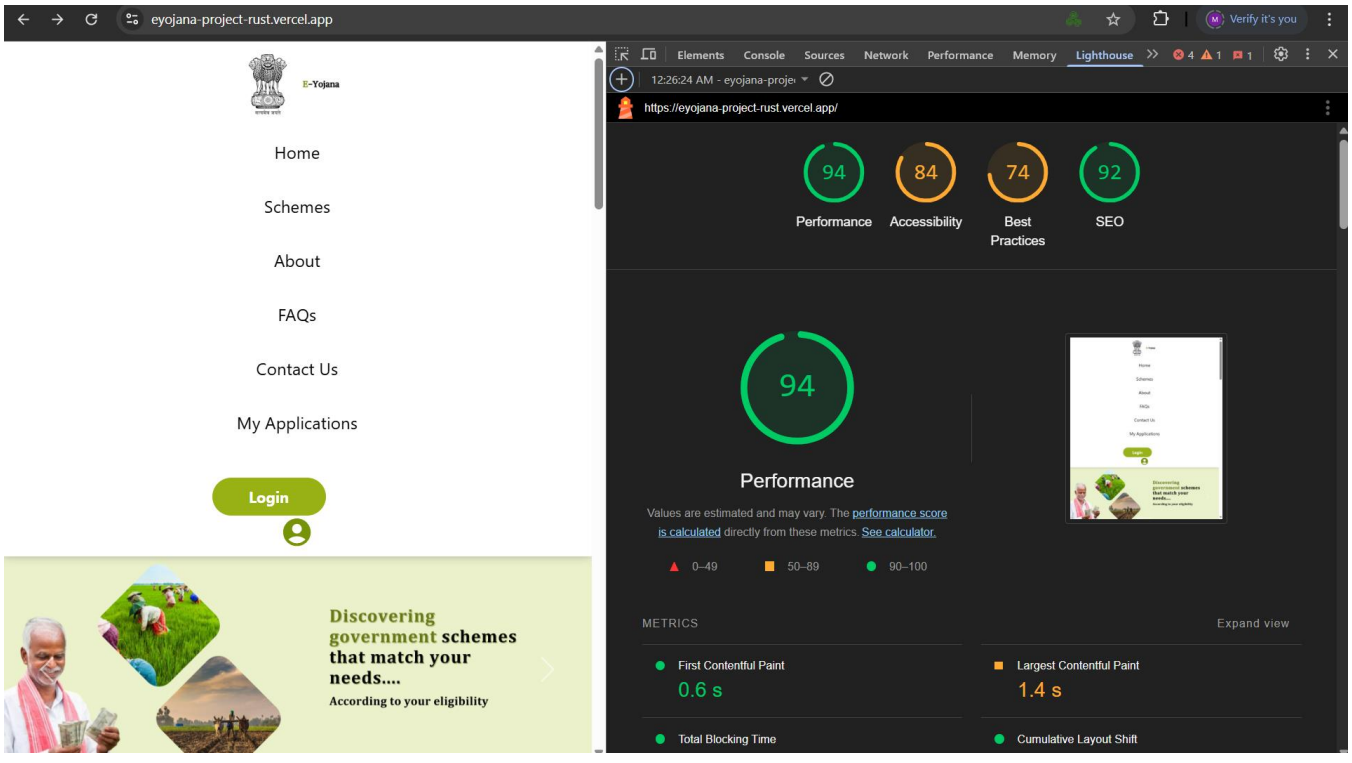
you figure in the 98th percentile, 50 meaning that you figure in the 75th percentile and so on.

2. **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.
3. **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.
4. **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

Performance before changes



Performance after changes



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