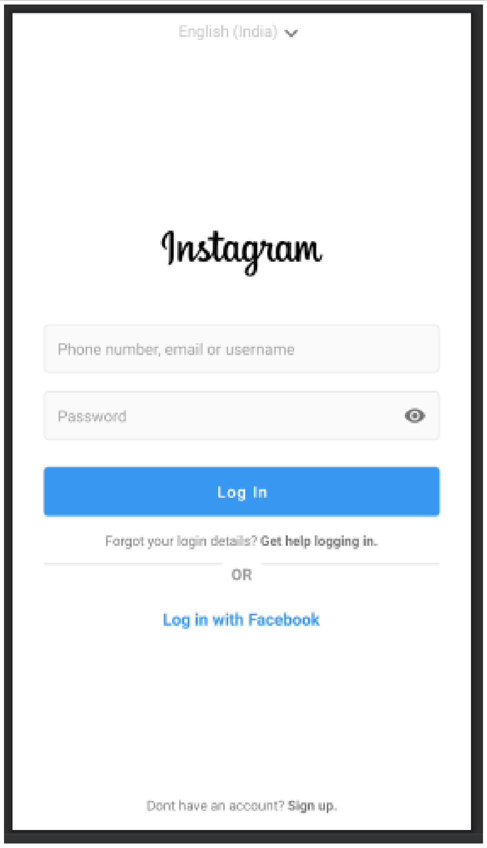
MAD & PWL Lab

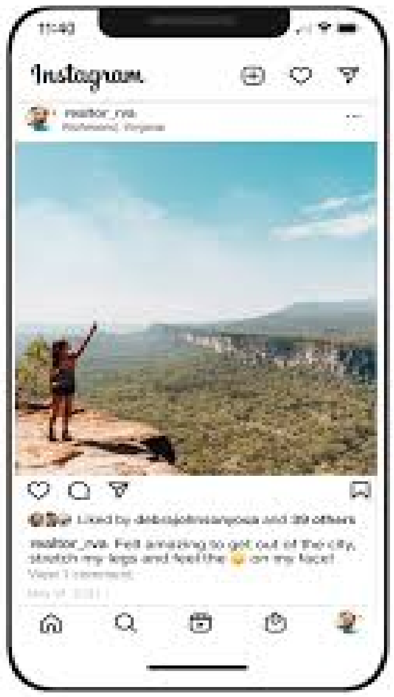
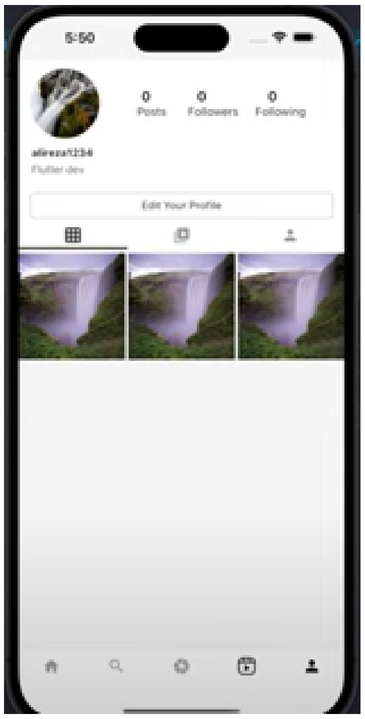
Sanket More

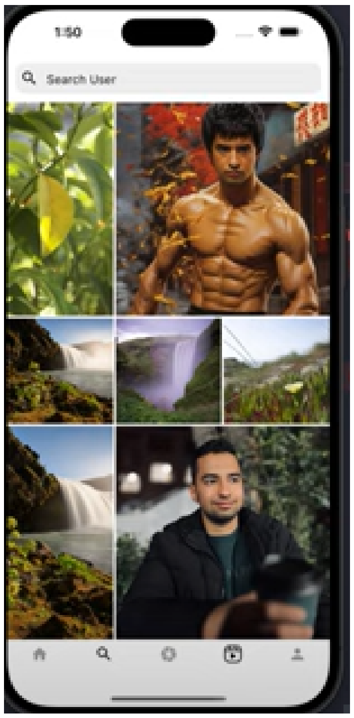
D15A 30

AIM: Selecting features for application development, the features should comprise:

1. Common widgets
2. Should include icons, images, charts etc.
3. Should have an interactive Form
4. Should apply navigation, routing and gestures
5. Should connect with FireBase database

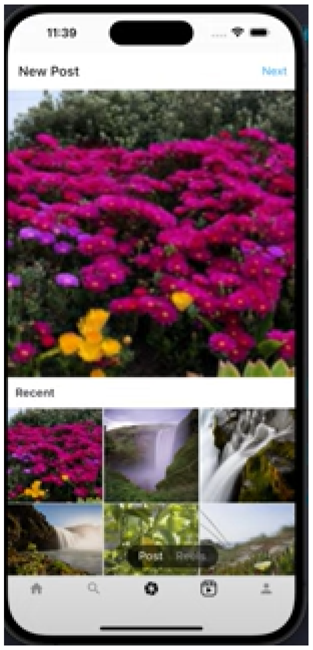
Screen Shot	Features
 English (India) Instagram Phone number, email or username Password Log In Forgot your login details? Get help logging in. OR Log in with Facebook Dont have an account? Sign up.	Landing Page :- 1. Sign in using email address 2. Firebase for authentication 3. Firebase : a. User : Username , email and password , likes, comments, followers and following.

	User Home Page:- 1. **Feed**: A scrollable stream of posts from people you follow, including photos, videos, and recommended content. 2. **Stories**: Temporary posts that disappear after 24 hours, displayed at the top of the home page. 3. **Reels**: Short-form video content showcasing trends, challenges, and popular clips from users. 4. **Activity/Notifications**: Updates on interactions like likes, comments, new followers, and mentions.
 0 Posts 0 Followers 0 Following alireza1234 Flutter dev Edit Your Profile	User Profile Page:- 1. **Profile Picture**: A customizable image that represents the user, displayed at the top of the profile page. 2. **Bio**: A short, editable text section where users can share information about themselves 3. **Posts Grid**: A visual grid showcasing all the posts a user has shared 4. **Followers/Following Count**: The number of people following the user and the number of people the user follows.



User Search Page:-

1. **Search Bar**: A tool at the top to quickly find users, hashtags, or locations by typing keywords.
2. **Suggested Accounts**: Personalized recommendations of accounts to follow based on your interests and activity.
3. **Trending Hashtags**: A section showing popular or trending hashtags that are currently gaining attention.
4. **Explore Feed**: A dynamic feed of content tailored to your interests, including posts from accounts you don't follow.



Add new post page:-

1. **Image/Video Upload**: Choose and upload photos or videos from your device or capture new content.
2. **Caption**: Add a description, hashtags, and emojis to your post.
3. **Tag People**: Tag other users in your photo or video to notify them.
4. **Location**: Option to add a location to your post for increased visibility.

MPL Experiment 1

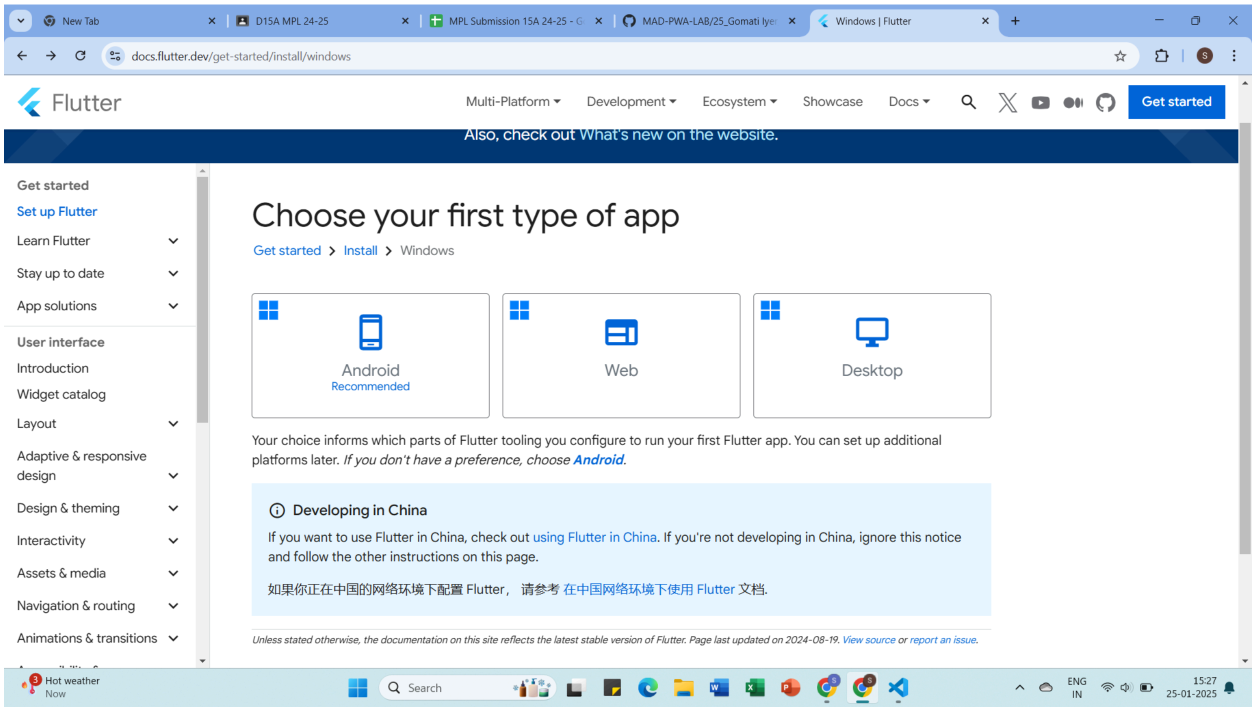
Sanket More

D15A 29

AIM: Installation and configuration of flutter Environment.

CODE:

Flutter Installation



docs.flutter.dev/get-started/install/windows

Also, check out What's new on the website.

Choose your first type of app

Get started > Install > Windows

Android Recommended | Web | Desktop

Your choice informs which parts of Flutter tooling you configure to run your first Flutter app. You can set up additional platforms later. *If you don't have a preference, choose **Android**.*

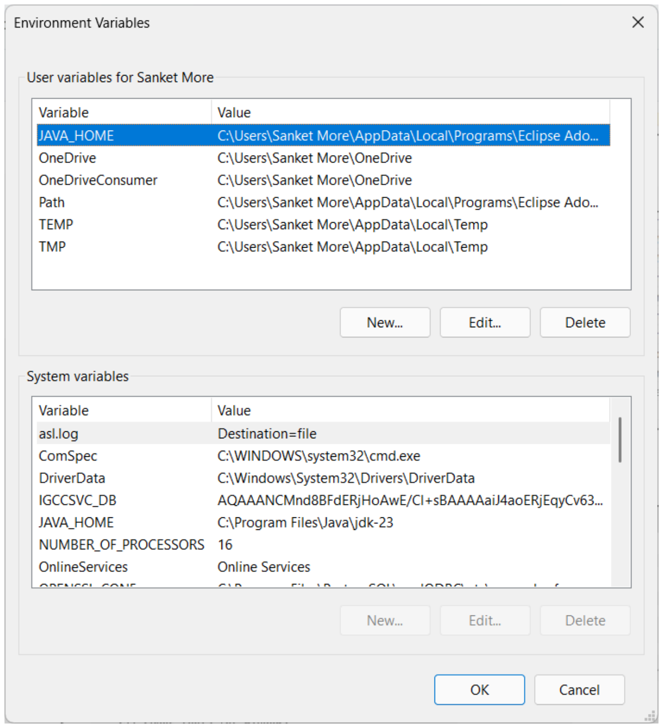
Developing in China

If you want to use Flutter in China, check out using Flutter in China. If you're not developing in China, ignore this notice and follow the other instructions on this page.

如果你正在中国的网络环境下配置 Flutter， 请参考 在中国网络环境下使用 Flutter 文档。

Unless stated otherwise, the documentation on this site reflects the latest stable version of Flutter. Page last updated on 2024-08-19. View source or report an issue.

Setting up Environment Variables



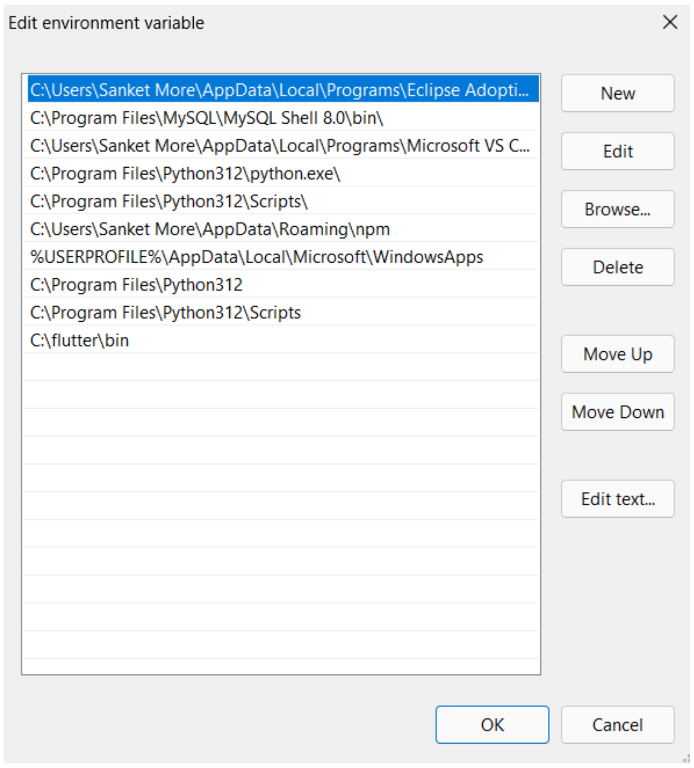
Environment Variables

User variables for Sanket More

Variable	Value
JAVA_HOME	C:\Users\Sanket More\AppData\Local\Programs\Eclipse Ado...
OneDrive	C:\Users\Sanket More\OneDrive
OneDriveConsumer	C:\Users\Sanket More\OneDrive
Path	C:\Users\Sanket More\AppData\Local\Programs\Eclipse Ado...
TEMP	C:\Users\Sanket More\AppData\Local\Temp
TMP	C:\Users\Sanket More\AppData\Local\Temp

System variables

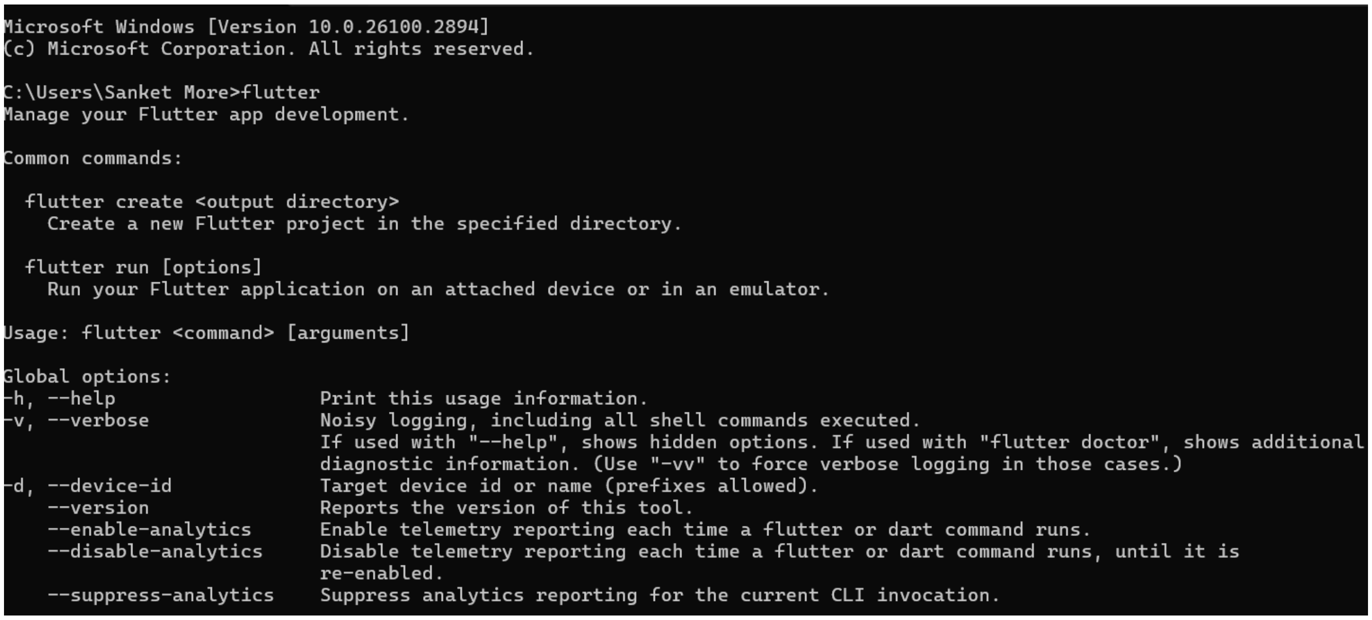
Variable	Value
asl.log	Destination=file
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
IGCCSVC_DB	AQAAANCMnd8BFdERjHoAwE/Cl+sBAAAAaiJ4aoERjEqyCv63...
JAVA_HOME	C:\Program Files\Java\jdk-23
NUMBER_OF_PROCESSORS	16
OnlineServices	Online Services



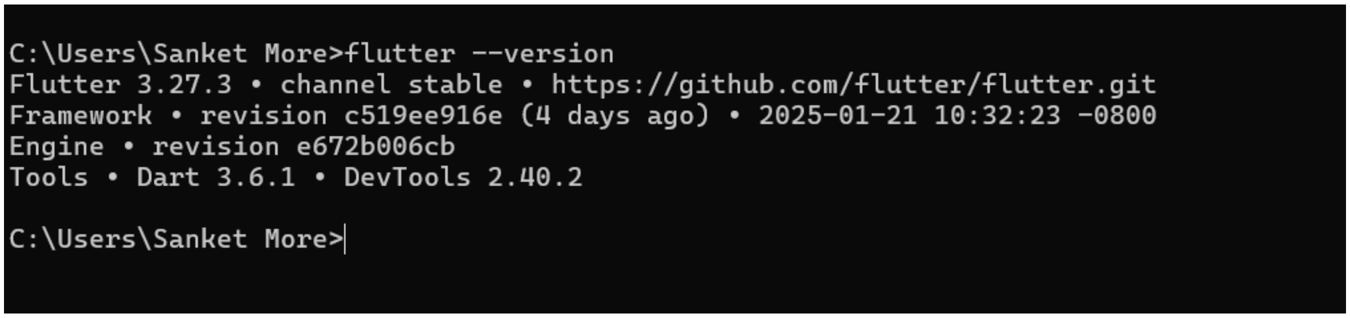
Edit environment variable

```
C:\Users\Sanket More\AppData\Local\Programs\Eclipse Adopti...
C:\Program Files\MySQL\MySQL Shell 8.0\bin\
C:\Users\Sanket More\AppData\Local\Programs\Microsoft VS C...
C:\Program Files\Python312\python.exe\
C:\Program Files\Python312\Scripts\
C:\Users\Sanket More\AppData\Roaming\npm
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
C:\Program Files\Python312
C:\Program Files\Python312\Scripts
C:\flutter\bin
```

Checking for flutter on cmd:

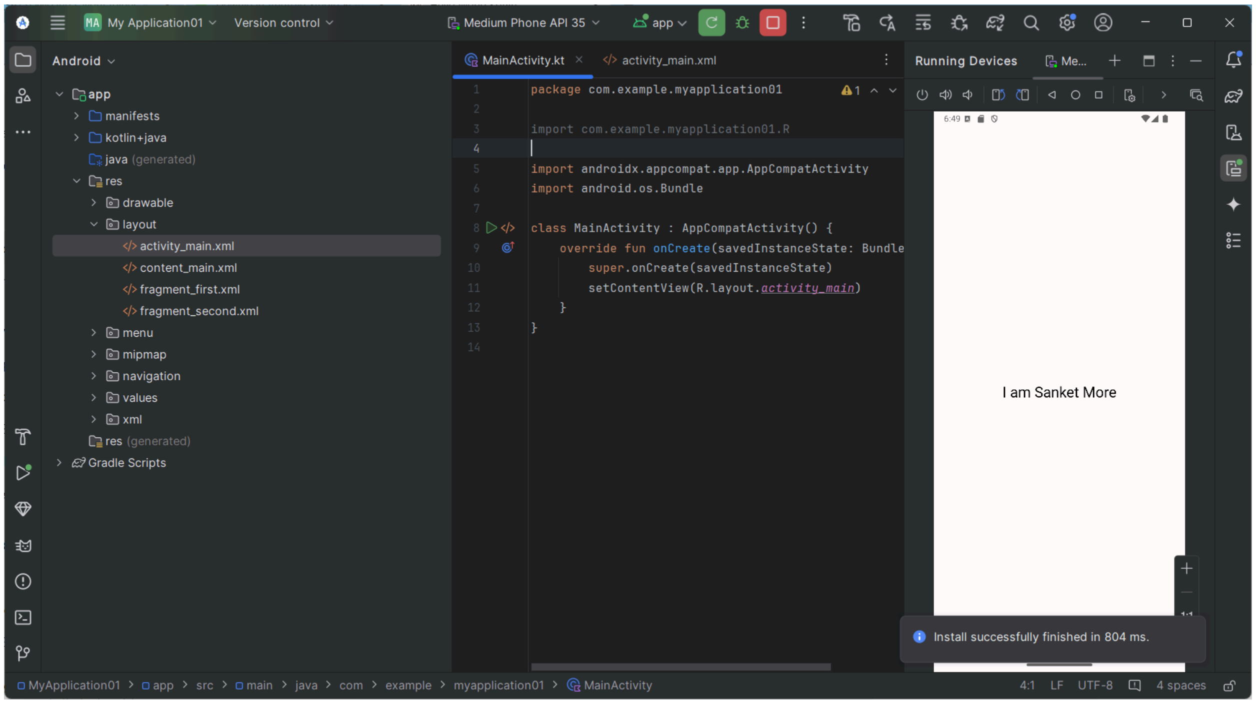


```
Microsoft Windows [Version 10.0.26100.2894]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Sanket More>flutter
Manage your Flutter app development.

Common commands:

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

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

Usage: flutter <command> [arguments]

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



```
C:\Users\Sanket More>flutter --version
Flutter 3.27.3 • channel stable • https://github.com/flutter/flutter.git
Framework • revision c519ee916e (4 days ago) • 2025-01-21 10:32:23 -0800
Engine • revision e672b006cb
Tools • Dart 3.6.1 • DevTools 2.40.2

C:\Users\Sanket More>
```

Flutter Basic Code:



```
package com.example.myapplication01

import com.example.myapplication01.R

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

Install successfully finished in 804 ms.

MPL Experiment-2

Sanket More

D15A 29

AIM:To design Flutter UI using 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.

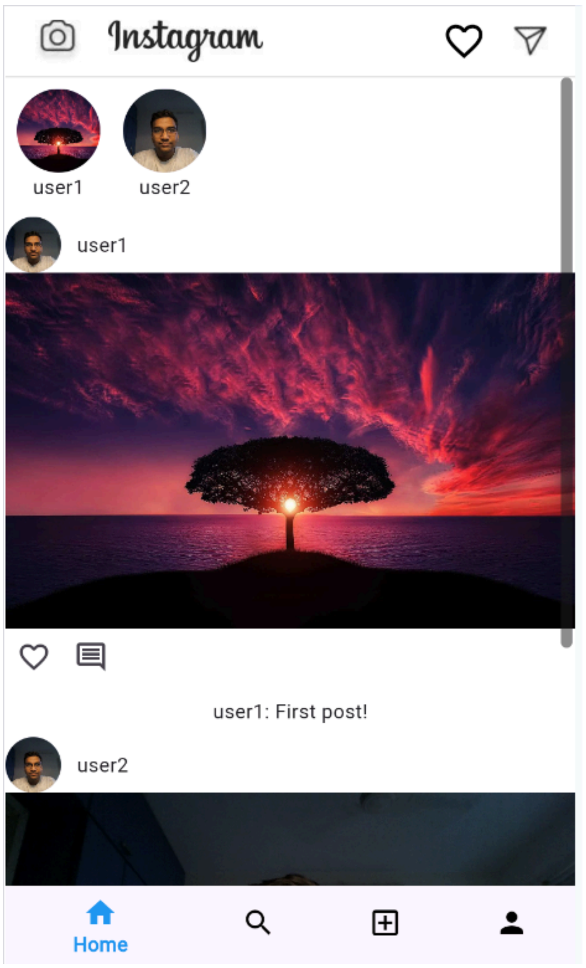
(e) Stack Widget
The Stack widget is used to place widgets on top of each other. This is useful for creating
overlapping UI elements such as banners, prole images, or layered designs.
(f) ElevatedButton Widget
The ElevatedButton widget is used for clickable buttons with a raised effect. It is a commonly
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
```
import 'package:flutter/material.dart';
import 'package:flutter_screenutil/flutter_screenutil.dart';

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

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  final List<Map<String, dynamic>> _posts = [
    {
      'user': 'user1',
      'imageUrl': 'images/post.jpg',
      'caption': 'First post!',
      'likes': 100,
      'comments': 20,
    },
    {
      'user': 'user2',
      'imageUrl': 'images/post2.jpg',
      'caption': 'Another day, another post.',
      'likes': 50,
      'comments': 10,
    },
    // ... more posts
  ];

  final List<Map<String, dynamic>> _stories = [
    {
      'user': 'user1',
      'imageUrl': 'images/post.jpg', // Replace with your story image URLs
    },
    {
      'user': 'user2',
      'imageUrl': 'images/post2.jpg',
    },
    // ... more stories
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        elevation: 0,
        toolbarHeight: 50.h,
        title: Image.asset(
          'images/instagram.jpg',
          height: 30.h,
        ),
        leading: Padding(
          padding: EdgeInsets.only(left: 16.w),
          child: Image.asset(
            'images/camera.jpg',
            height: 24.h,
          ),
        ),
        actions: [
          Padding(
            padding: EdgeInsets.only(right: 16.w),
            child: Icon(
              Icons.favorite_border_outlined,
              color: Colors.black,
              size: 28.sp,
            ),
          ),
          Padding(
            padding: EdgeInsets.only(right: 16.w),
            child: Image.asset(
              'images/send.jpg',
              height: 24.h,
            ),
          ),
        ],
        backgroundColor: Colors.white,
        bottom: PreferredSize(
          preferredSize: Size.fromHeight(1.h),
          child: Container(
            height: 1.h,
            color: Colors.grey[300],
          ),
        ),
      ),
      body: SingleChildScrollView(
        // Added SingleChildScrollView
        child: Column(
          children: [
            // Stories Section
            SizedBox(
              height: 100.h, // Adjust height as needed
              child: ListView.builder(
                scrollDirection: Axis.horizontal,
                itemCount: _stories.length,
                itemBuilder: (context, index) {
                  return StoryWidget(_stories[index]);
                },
              ),
            ),
            // Posts Section
            ListView.builder(
              shrinkWrap: true, // Important for nested ListViews
```

```
              physics:
                  const NeverScrollableScrollPhysics(), // Disable
scrolling of inner ListView
              itemCount: _posts.length,
              itemBuilder: (context, index) {
                return PostWidget(_posts[index]);
              },
            ),
          ],
        ),
      ),
    );
  }
}

class StoryWidget extends StatelessWidget {
  final Map<String, dynamic> storyData;

  const StoryWidget(this.storyData, {super.key});

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.all(8.0),
      child: Column(
        children: [
          CircleAvatar(
            radius: 30.r,
            backgroundImage: AssetImage(storyData['imageUrl'] ??
                'images/default_profile.png'), // Use story image
          ),
          Text(storyData['user'] ?? ""),
        ],
      ),
    );
  }
}

class PostWidget extends StatelessWidget {
  final Map<String, dynamic> postData;
```

```dart
const PostWidget(this.postData, {super.key});

@override
Widget build(BuildContext context) {
  return Column(
    children: [
      // User Info
      Row(
        children: [
          CircleAvatar(
            radius: 20.r,
            backgroundImage: const AssetImage('images/post2.jpg'),
          ),
          SizedBox(width: 10.w),
          Text(postData['user'] ?? ""),
        ],
      ),

      // Post Image
      Image.asset(postData['imageUrl'] ?? ""),

      // Post Actions (Likes, Comments)
      Row(
        children: [
          IconButton(
            icon: const Icon(Icons.favorite_border),
            onPressed: () {
              // Handle like
            },
          ),
          IconButton(
            icon: const Icon(Icons.comment_outlined),
            onPressed: () {
              // Handle comment
            },
          ),
        ],
      ),
            // Caption
            Padding(
              padding: const EdgeInsets.all(8.0),
              child:
                  Text('${postData['user'] ?? ""}: ${postData['caption'] ??
""}'),
            ),
          ],
        );
    }
}
```

OUTPUT:-



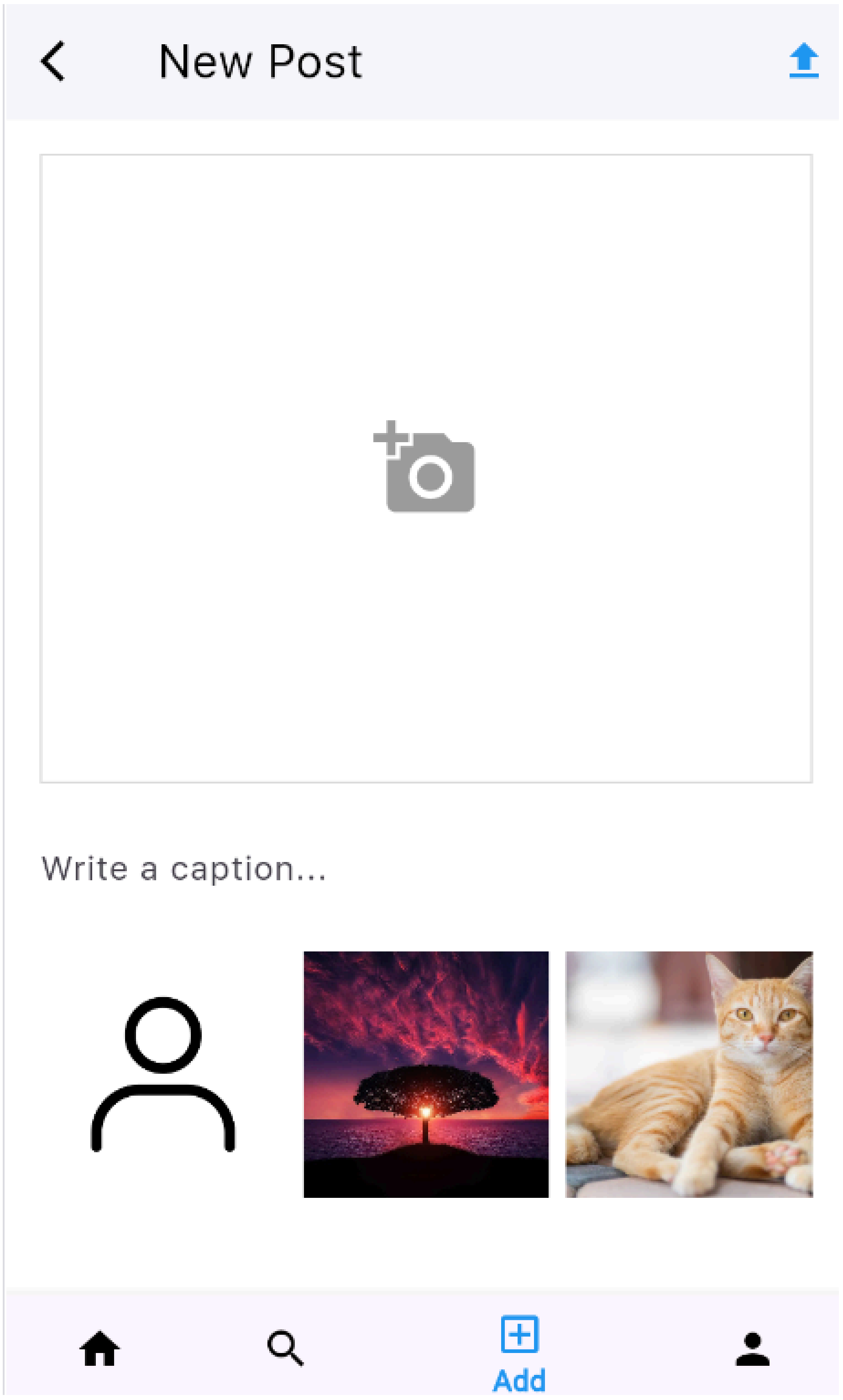
user1: First post!

add_post_screen.dart:-

```dart
// add_post_screen.dart
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
import 'dart:io';

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

  @override
  State<AddPostScreen> createState() => _AddPostScreenState();
}

class _AddPostScreenState extends State<AddPostScreen> {
  File? _image;
  final TextEditingController _captionController =
TextEditingController();
  bool _isLoading = false;
  List<String> _previousImageAssets = [];

  @override
  void initState() {
    super.initState();
    _loadPreviousImages();
  }

  Future<void> _loadPreviousImages() async {
    setState(() {
      _previousImageAssets = [
        'images/person.png', // Replace with your actual image paths
        'images/post.jpg', // Make sure these images are in your assets
        'images/cat.png', // Replace with a valid image in your assets
      ];
    });
  }
```

```
Future<void> _pickImage(ImageSource source) async {
  final pickedFile = await ImagePicker().pickImage(source: source);

  setState(() {
    if (pickedFile != null) {
      _image = File(pickedFile.path);
    } else {
      print('No image selected.');
    }
  });
}

void _uploadPost() async {
  setState(() {
    _isLoading = true;
  });

  if (_image != null) {
    String caption = _captionController.text;

    // Simulate upload (replace with your actual logic)
    await Future.delayed(const Duration(seconds: 2));

    print("Uploading image: ${_image!.path}");
    print("Caption: $caption");

    setState(() {
      _image = null;
      _captionController.clear();
      _isLoading = false;
    });

    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('Post uploaded successfully!')),
    );
  } else {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('Please select an image')),
    );
    setState(() {
        _isLoading = false;
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 1,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back_ios, color: Colors.black),
          onPressed: () {
            Navigator.pop(context);
          },
        ),
        title: const Text('New Post',
            style: TextStyle(color: Colors.black, fontFamily:
'SFProText')),
        actions: [
          IconButton(
            onPressed: _isLoading ? null : _uploadPost,
            icon: _isLoading
                ? const CircularProgressIndicator()
                : const Icon(Icons.upload, color: Colors.blue),
          ),
        ],
      ),
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              GestureDetector(
                onTap: () {
                  showModalBottomSheet(
                    context: context,
                    builder: (BuildContext context) {
                      return Column(
                        mainAxisSize: MainAxisSize.min,
                        children: <Widget>[
                          ListTile(
                            leading: const Icon(Icons.photo_library),
                            title: const Text('Photo Library'),
                            onTap: () {
                              _pickImage(ImageSource.gallery);
                              Navigator.pop(context);
                            },
                          ),
                          ListTile(
                            leading: const Icon(Icons.camera_alt),
                            title: const Text('Camera'),
                            onTap: () {
                              _pickImage(ImageSource.camera);
                              Navigator.pop(context);
                            },
                          ),
                        ],
                      );
                    },
                  );
                },
                child: Container(
                  height: 300,
                  width: double.infinity,
                  decoration: BoxDecoration(
                    border: Border.all(color: Colors.grey.shade300),
                  ),
                  child: _image != null
                      ? Image.file(_image!, fit: BoxFit.cover)
                      : const Center(
                          child: Icon(Icons.add_a_photo,
                              size: 50, color: Colors.grey)),
                ),
              ),
              const SizedBox(height: 16),
              TextField(
                controller: _captionController,
                style: const TextStyle(fontFamily: 'SFProText'),
                decoration: const InputDecoration(
                  hintText: 'Write a caption...',
                  hintStyle: TextStyle(fontFamily: 'SFProText'),
                  border: InputBorder.none,
                ),
                maxLines: null,
              ),
              const SizedBox(height: 16),
              GridView.builder(
                shrinkWrap: true,
                physics: const NeverScrollableScrollPhysics(),
                gridDelegate: const
SliverGridDelegateWithFixedCrossAxisCount(
                  crossAxisCount: 3,
                  crossAxisSpacing: 8,
                  mainAxisSpacing: 8,
                ),
                itemCount: _previousImageAssets.length,
                itemBuilder: (context, index) {
                  return Image.asset(
                    _previousImageAssets[index],
                    fit: BoxFit.cover,
                  );
                },
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

OUTPUT:-



New Post

Write a caption...

MPL Experiment-3

Sanket More
D15A 29

AIM:To include images,icons and 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_utter package)
● Custom SVG Icons (utter_svg package)

Eg in pubspec.yaml le -

```
dependencies:
 font_awesome_utter: ^10.5.0
```

In code -

```
import 'package:font_awesome_utter/font_awesome_utter.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:

```
utter:
 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:

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

In code -

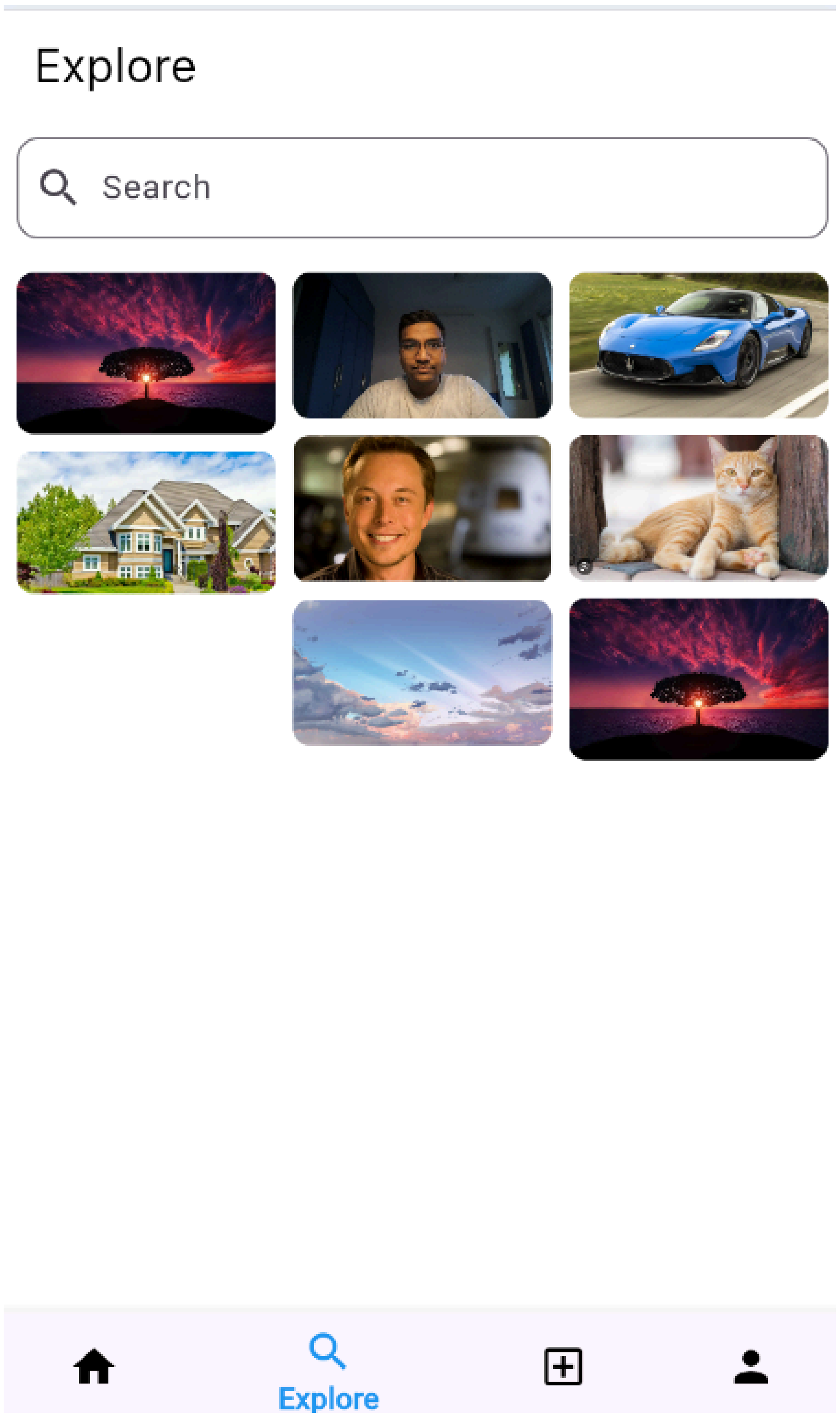
```
Text(
 "Hello, Flutter!",
 style: TextStyle(fontFamily: "CustomFont", fontSize: 20, fontWeight:
FontWeight.bold),
)
```

CODE:-

explore.dart

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

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

  @override
  State<ExploreScreen> createState() => _ExploreScreenState();
}

class _ExploreScreenState extends State<ExploreScreen> {
  final List<String> _imageUrls = [
    'images/post.jpg',
    'images/post2.jpg',
    'images/car.png',
    'images/cat.png',
    'images/elon.png',
    'images/house.png',
    'images/post.jpg',
    'images/sky.png',
  ];

  // Add a TextEditingController for the search bar
  final TextEditingController _searchController = TextEditingController();

  // Add a filtered list of image URLs
  List<String> _filteredImageUrls = [];

  @override
  void initState() {
    super.initState();
    _filteredImageUrls = _imageUrls; // Initialize with all images
  }

  void _filterImages(String query) {
    setState(() {
      if (query.isEmpty) {
        _filteredImageUrls = _imageUrls; // Show all images if search is
empty
      } else {
        _filteredImageUrls = _imageUrls
            .where((imageUrl) =>
                imageUrl.toLowerCase().contains(query.toLowerCase()))
            .toList();
      }
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 0,
        title: const Text('Explore',
            style: TextStyle(color: Colors.black, fontFamily:
'SFProText')),
        bottom: PreferredSize(
          // Add a search bar below the app bar
          preferredSize: const Size.fromHeight(60.0),
          child: Padding(
            padding: const EdgeInsets.all(8.0),
            child: TextField(
              controller: _searchController,
              onChanged: _filterImages, // Filter images on text change
              decoration: InputDecoration(
                hintText: 'Search',
                prefixIcon: const Icon(Icons.search),
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(10.0),
                ),
              ),
            ),
          ),
        ),
      ),
      body: Padding(
        padding: const EdgeInsets.all(8.0),
        child: MasonryGridView.count(
          crossAxisCount: 3,
          mainAxisSpacing: 8.0,
          crossAxisSpacing: 8.0,
          itemCount: _filteredImageUrls.length, // Use the filtered list
          itemBuilder: (context, index) {
            return ClipRRect(
              borderRadius: BorderRadius.circular(8.0),
              child: Image.asset(
                _filteredImageUrls[index], // Use the filtered list
                fit: BoxFit.cover,
              ),
            );
          },
        ),
      ),
    );
  }
}
```

OUTPUT:-



MPL Experiment - 4

Sanket More

D15A 29

AIM: To create an interactive form using form widget

Theory:

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

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

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

Steps:

1. **Create a new Flutter project or open an existing one.**
    - You can create a new Flutter project using the command: `flutter create form_example`.
2. **Define a Form widget inside a StatefulWidget to manage user input.**

- Use a `StatefulWidget` to maintain the state of the form, allowing for dynamic validation and input handling.

3. **Use TextFormField for text input fields with validation logic.**

   - Use `TextFormField` for user input, with built-in validation to ensure that the fields are not left empty and that the data is in the correct format.

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

   - Add interactive widgets like `DropdownButton`, `Checkbox`, and `Switch` to collect different types of user input.

5. **Wrap the form inside a SingleChildScrollView to ensure smooth scrolling.**

   - This ensures that when the keyboard appears or the form becomes long, users can still scroll through the form fields.

6. **Implement an ElevatedButton to trigger form validation and submission.**

   - Use an `ElevatedButton` to trigger the validation of the form fields and perform any necessary actions after the form is validated.

7. **Use GlobalKey to manage form validation.**

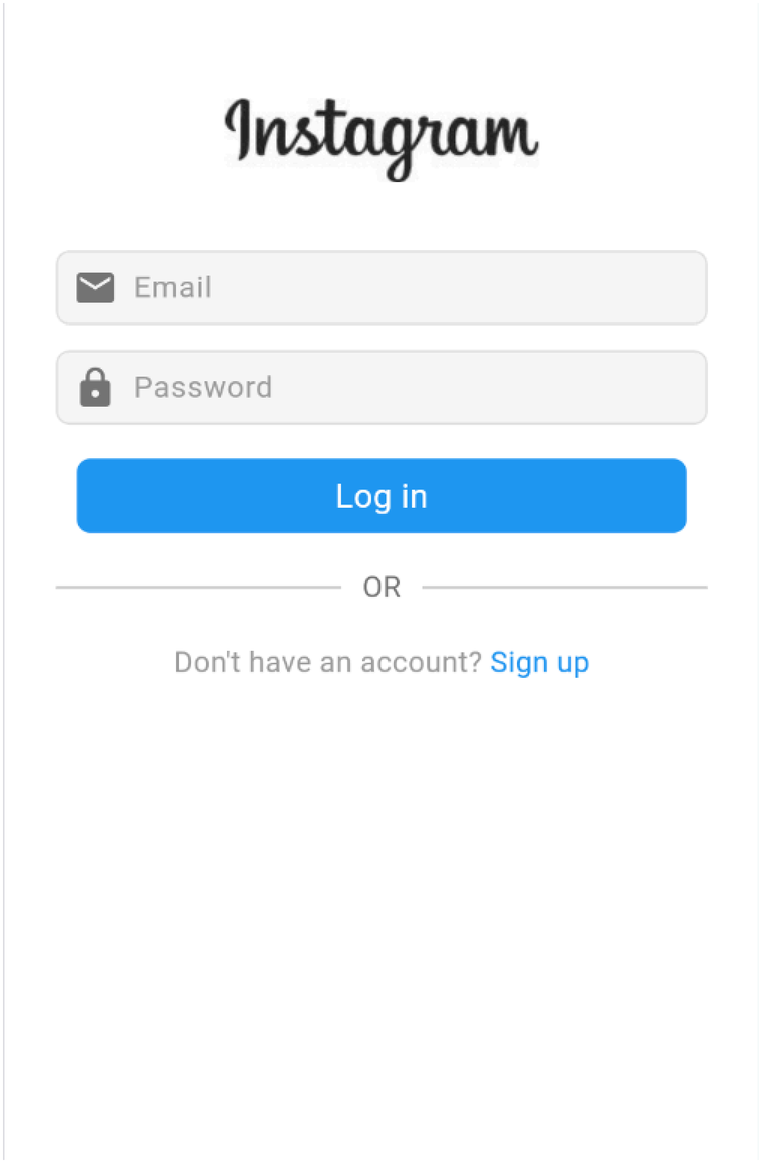
   - A `GlobalKey<FormState>` is essential to manage the form state, particularly for validation and form submission. The key allows you to validate all fields before submission.

CODE:-

signup.dart

```
import 'dart:io';

import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
import 'package:flutter_screenutil/flutter_screenutil.dart';

class SignupScreen extends StatefulWidget {
  final Function(BuildContext) showLogin;
  const SignupScreen({Key? key, required this.showLogin}) : super(key:
key);

  @override
  State<SignupScreen> createState() => _SignupScreenState();
}

class _SignupScreenState extends State<SignupScreen> {
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _usernameController =
TextEditingController();
  final TextEditingController _nameController =
      TextEditingController(); // Added name controller
  final TextEditingController _passwordController =
TextEditingController();
  final TextEditingController _confirmPasswordController =
      TextEditingController();

  File? _profileImage;

  Future<void> _pickImage() async {
    final pickedFile =
        await ImagePicker().pickImage(source: ImageSource.gallery);
    if (pickedFile != null) {
      setState(() {
        _profileImage = File(pickedFile.path);
      });
    }
  }
  void _signup() {
    // Implement your signup logic here
    print('Name: ${_nameController.text}');
    print('Email: ${_emailController.text}');
    print('Username: ${_usernameController.text}');
    print('Password: ${_passwordController.text}');
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white, // Instagram background color
      body: SafeArea(
        child: Padding(
          padding: EdgeInsets.symmetric(horizontal: 24.w), // Consistent
padding
          child: SingleChildScrollView(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.stretch, // Align to
start
              children: [
                SizedBox(height: 40.h),

                Center(
                  // Center the title
                  child: Text(
                    'Sign Up',
                    style: TextStyle(
                      fontSize: 24.sp,
                      fontWeight: FontWeight.w500,
                      color: Colors.black,
                    ),
                  ),
                ),

                SizedBox(height: 30.h),

                GestureDetector(
                  // Image picker area
                  onTap: _pickImage,
                  child: CircleAvatar(
                    radius: 40.r,
                    backgroundColor: Colors.grey[200], // Light background
                    backgroundImage: _profileImage != null
                        ? FileImage(_profileImage!)
                        : null,
                    child: _profileImage == null
                        ? Icon(
                            Icons.add_a_photo,
                            size: 30.r,
                            color: Colors.grey,
                          )
                        : null,
                  ),
                ),

                SizedBox(height: 20.h),

                _buildTextField(_nameController, 'Full Name',
                    Icons.person), // Added name field
                SizedBox(height: 15.h),
                _buildTextField(_emailController, 'Email', Icons.email),
                SizedBox(height: 15.h),
                _buildTextField(
                    _usernameController, 'Username',
Icons.person_outline),
                SizedBox(height: 15.h),
                _buildTextField(_passwordController, 'Password',
Icons.lock,
                    obscureText: true),
                SizedBox(height: 15.h),
                _buildTextField(
                    _confirmPasswordController, 'Confirm Password',
Icons.lock,
                    obscureText: true),

                SizedBox(height: 25.h),

                _buildSignupButton(), // Styled button
                SizedBox(height: 25.h),

                _buildOrDivider(),

                SizedBox(height: 25.h),

                _buildLoginLink(), // Instagram-style login link
              ],
            ),
          ),
        ),
      ),
    );
  }

Widget _buildOrDivider() {
  return Row(
    children: [
      Expanded(
        child: Divider(
          color: Colors.grey[400],
          thickness: 1,
        ),
      ),
      Padding(
        padding: EdgeInsets.symmetric(horizontal: 10.w),
        child: Text(
          'OR',
          style: TextStyle(
            fontSize: 14.sp,
            color: Colors.grey[600],
          ),
        ),
      ),
      Expanded(
        child: Divider(
          color: Colors.grey[400],
          thickness: 1,
        ),
      ),
      ],
    );
  }

  Widget _buildSignupButton() {
    return ElevatedButton(
      onPressed: _signup,
      child: const Text('Sign Up'),
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.blue, // Instagram blue
        minimumSize: Size(double.infinity, 44.h), // Consistent button
height
        shape: RoundedRectangleBorder(
          // Rounded corners
          borderRadius: BorderRadius.circular(8.r),
        ),
        textStyle: TextStyle(
          fontSize: 16.sp,
          fontWeight: FontWeight.w500,
        ),
      ),
    );
  }

  Widget _buildLoginLink() {
    return Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          "Already have an account? ",
          style: TextStyle(fontSize: 14.sp, color: Colors.grey),
        ),
        GestureDetector(
          onTap: () {
            widget.showLogin(context);
          },
          child: Text(
            "Log in",
            style: TextStyle(
              color: Colors.blue,
              fontWeight: FontWeight.w500,
              fontSize: 14.sp,
            ),
          ),
        ),
      ],
    );
  }

  Widget _buildTextField(
      TextEditingController controller, String labelText, IconData icon,
      {bool obscureText = false}) {
    return Container(
      height: 44.h,
      decoration: BoxDecoration(
        color: Colors.grey[100], // Light gray background
        borderRadius: BorderRadius.circular(8.r), // Rounded corners
        border: Border.all(color: Colors.grey[300]!), // Added border
      ),
      child: TextField(
        controller: controller,
        obscureText: obscureText,
        style: TextStyle(fontSize: 16.sp, color: Colors.black),
        decoration: InputDecoration(
          labelText: labelText,
          labelStyle: TextStyle(color: Colors.grey),
          prefixIcon: Icon(
            icon,
            color: Colors.grey[600],
          ),
          contentPadding:
              EdgeInsets.symmetric(horizontal: 15.w, vertical: 12.h),
          border: InputBorder.none,
          focusedBorder: InputBorder.none,
          enabledBorder: InputBorder.none,
        ),
      ),
    );
  }
}
```

OUTPUT:-



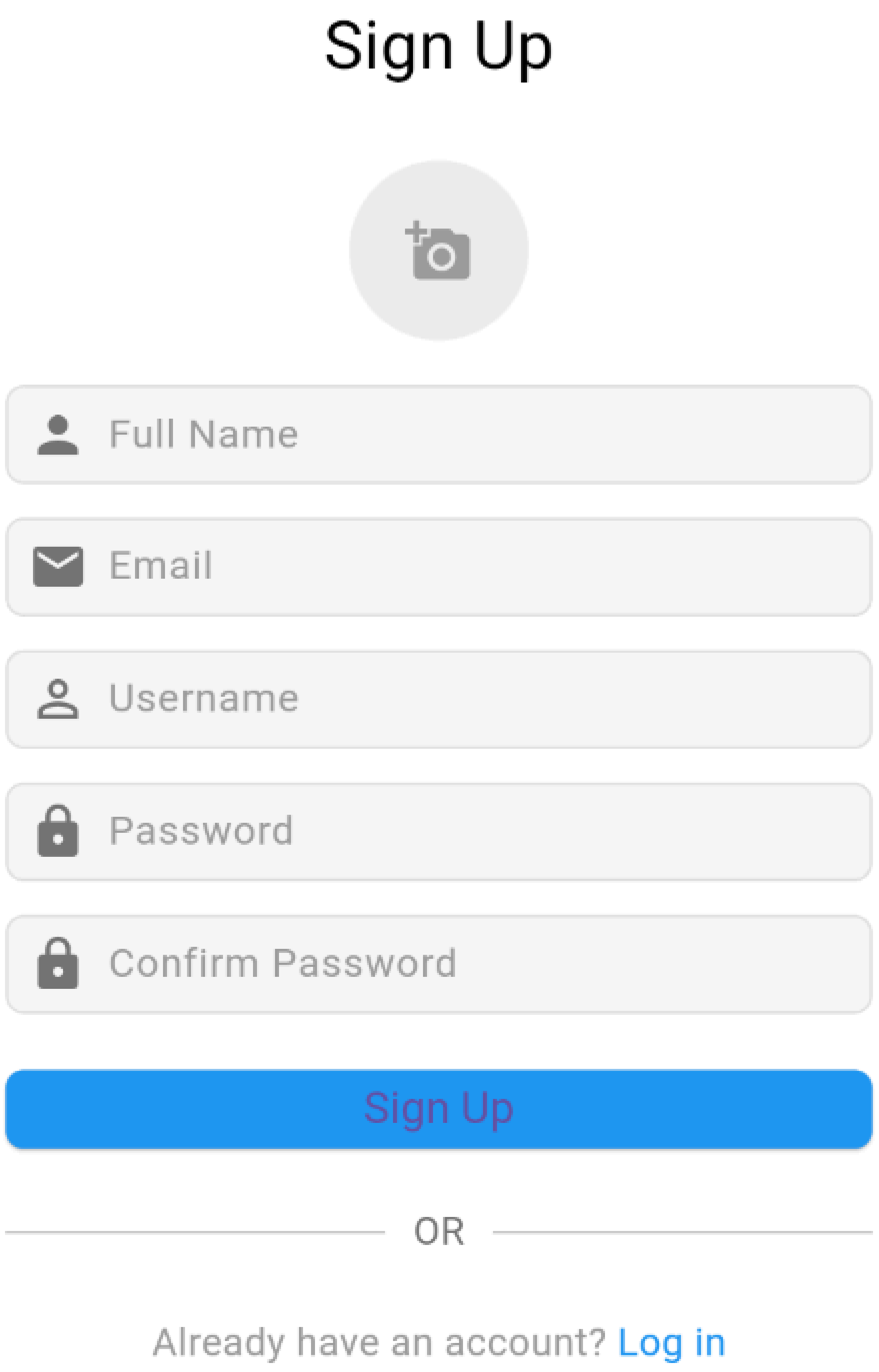
login_screen.dart

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

class LoginScreen extends StatefulWidget {
  final Function(BuildContext) showSignup;
  const LoginScreen({Key? key, required this.showSignup}) : super(key:
key);
```

```
  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final email = TextEditingController();
  final password = TextEditingController();
  FocusNode emailFocus = FocusNode();
  FocusNode passwordFocus = FocusNode();

  @override
  void dispose() {
    email.dispose();
    password.dispose();
    emailFocus.dispose();
    passwordFocus.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      resizeToAvoidBottomInset: false,
      backgroundColor: Colors.white,
      body: SafeArea(
        child: Padding(
          padding: EdgeInsets.symmetric(horizontal: 24.w),
          child: SingleChildScrollView(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.stretch,
              children: [
                SizedBox(height: 60.h),
                Center(
                  child: Image.asset(
                    'images/logo.jpg',
                    height: 50.h,
                  ),
                ),
                SizedBox(height: 40.h),
```

```dart
              _buildTextField(email, emailFocus, 'Email', Icons.email),
              SizedBox(height: 15.h),
              _buildTextField(
                  password, passwordFocus, 'Password', Icons.lock),
              SizedBox(height: 20.h),
              _buildLoginButton(),
              SizedBox(height: 20.h),
              _buildOrDivider(),
              SizedBox(height: 20.h),
              _buildSignupButton(context),
            ],
          ),
        ),
      ),
    ),
  );
}

Widget _buildOrDivider() {
  return Row(
    children: [
      Expanded(
        child: Divider(
          color: Colors.grey[400],
          thickness: 1,
        ),
      ),
      Padding(
        padding: EdgeInsets.symmetric(horizontal: 10.w),
        child: Text(
          'OR',
          style: TextStyle(
            fontSize: 14.sp,
            color: Colors.grey[600],
          ),
        ),
      ),
      Expanded(
        child: Divider(
          color: Colors.grey[400],
          thickness: 1,
        ),
      ),
    ],
  );
}

Widget _buildSignupButton(BuildContext context) {
  return Padding(
    padding: EdgeInsets.symmetric(horizontal: 10.w),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          "Don't have an account? ",
          style: TextStyle(
            fontSize: 14.sp,
            color: Colors.grey,
          ),
        ),
        GestureDetector(
          onTap: () {
            widget.showSignup(context);
          },
          child: Text(
            "Sign up",
            style: TextStyle(
              fontSize: 14.sp,
              color: Colors.blue,
              fontWeight: FontWeight.w500,
            ),
          ),
        ),
      ],
    ),
  );
}

Widget _buildLoginButton() {
  return Padding(
    padding: EdgeInsets.symmetric(horizontal: 10.w),
    child: InkWell(
      onTap: () {
        // Handle login here
      },
      child: Container(
        alignment: Alignment.center,
        width: double.infinity,
        height: 44.h,
        decoration: BoxDecoration(
          color: Colors.blue,
          borderRadius: BorderRadius.circular(8.r),
        ),
        child: Text(
          'Log in',
          style: TextStyle(
            fontSize: 16.sp,
            color: Colors.white,
            fontWeight: FontWeight.w500,
          ),
        ),
      ),
    ),
  );
}

Padding _buildTextField(TextEditingController controller, FocusNode
focusNode,
    String labelText, IconData icon) {
  return Padding(
    padding: EdgeInsets.symmetric(horizontal: 0.w),
    child: Container(
      height: 44.h,
      decoration: BoxDecoration(
        color: Colors.grey[100],
        borderRadius: BorderRadius.circular(8.r),
        border: Border.all(color: Colors.grey[300]!),
      ),
      child: TextField(
        style: TextStyle(fontSize: 16.sp, color: Colors.black),
        controller: controller,
        focusNode: focusNode,
        decoration: InputDecoration(
          labelText: labelText,
          labelStyle: TextStyle(color: Colors.grey),
          prefixIcon: Icon(
            icon,
            color: focusNode.hasFocus ? Colors.blue : Colors.grey[600],
          ),
          contentPadding:
              EdgeInsets.symmetric(horizontal: 15.w, vertical: 12.h),
          border: InputBorder.none,
          focusedBorder: InputBorder.none,
          enabledBorder: InputBorder.none,
        ),
      ),
    ),
  );
}
}
```

OUTPUT:-


Sign Up

Full Name

Email

Username

Password

Confirm Password

Sign Up

OR

Already have an account? Log in

MPL Experiment-5

Sanket More
D15A 29

AIM: To apply navigation and routing in flutter application.

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-dened route names to navigate.
● Navigation with Data: Allows passing data between screens when navigating.

2. **Routing in Flutter**

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

Types of Routing:

● Direct Routing: Navigates to a specic screen using explicit methods.
● Named Routing: Uses a predened 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 nger 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 seciently, and gestures enable touch interactions. Mastering these concepts helps in developing dynamic and user-friendly Flutter applications.

CODE:
navigation.dart
```
import 'package:flutter/material.dart';
import 'package:flutter_screenutil/flutter_screenutil.dart';
import '../screen/home.dart';
import '../screen/explore.dart';
import '../screen/add_post_screen.dart';

class NavigationScreen extends StatefulWidget {
  const NavigationScreen({super.key});
  @override
  State<NavigationScreen> createState() => _NavigationScreenState();
}

class _NavigationScreenState extends State<NavigationScreen> {
  int _selectedIndex = 0;

  static const List<Widget> _widgetOptions = <Widget>[
    HomeScreen(),
    ExploreScreen(),
    AddPostScreen(),
  ];

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: _widgetOptions.elementAt(_selectedIndex),
      ),
      bottomNavigationBar: BottomNavigationBar(
        items: const <BottomNavigationBarItem>[
          BottomNavigationBarItem(
            icon: Icon(Icons.home),
            label: 'Home',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.search),
            label: 'Explore',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.add_box_outlined),
            label: 'Add',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.person),
            label: 'Profile',
          ),
        ],
        currentIndex: _selectedIndex,
        selectedItemColor: Colors.blue, // Highlight color
        unselectedItemColor: Colors.black, // Set unselected item color to
black
        selectedLabelStyle:
            const TextStyle(fontWeight: FontWeight.bold), // Bold selected
label
        onTap: _onItemTapped,
      ),
    );
  }
}
```

add_post_screen.dart

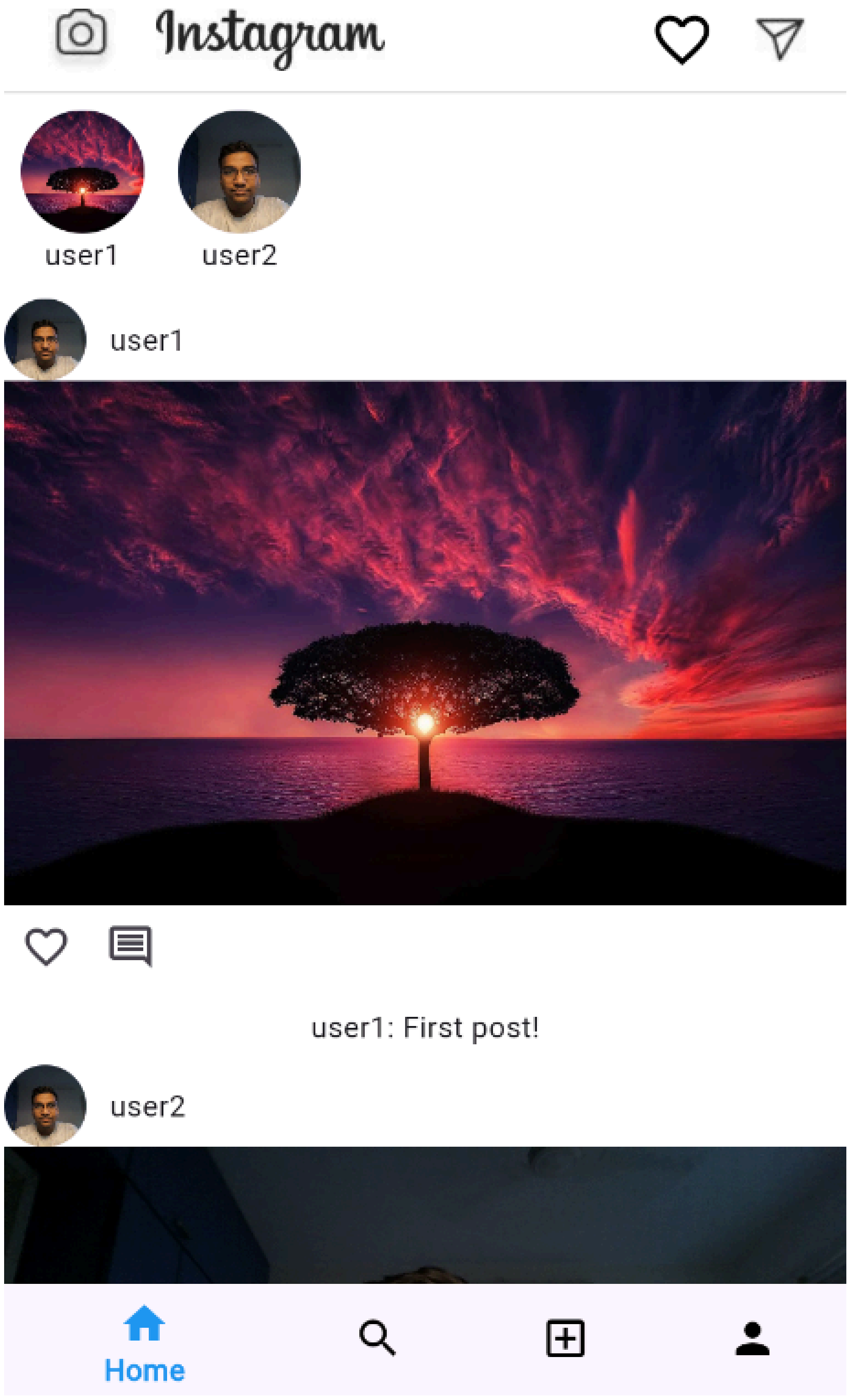
```
// add_post_screen.dart
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
import 'dart:io';

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

  @override
  State<AddPostScreen> createState() => _AddPostScreenState();
}

class _AddPostScreenState extends State<AddPostScreen> {
  File? _image;
  final TextEditingController _captionController =
TextEditingController();
  bool _isLoading = false;
```

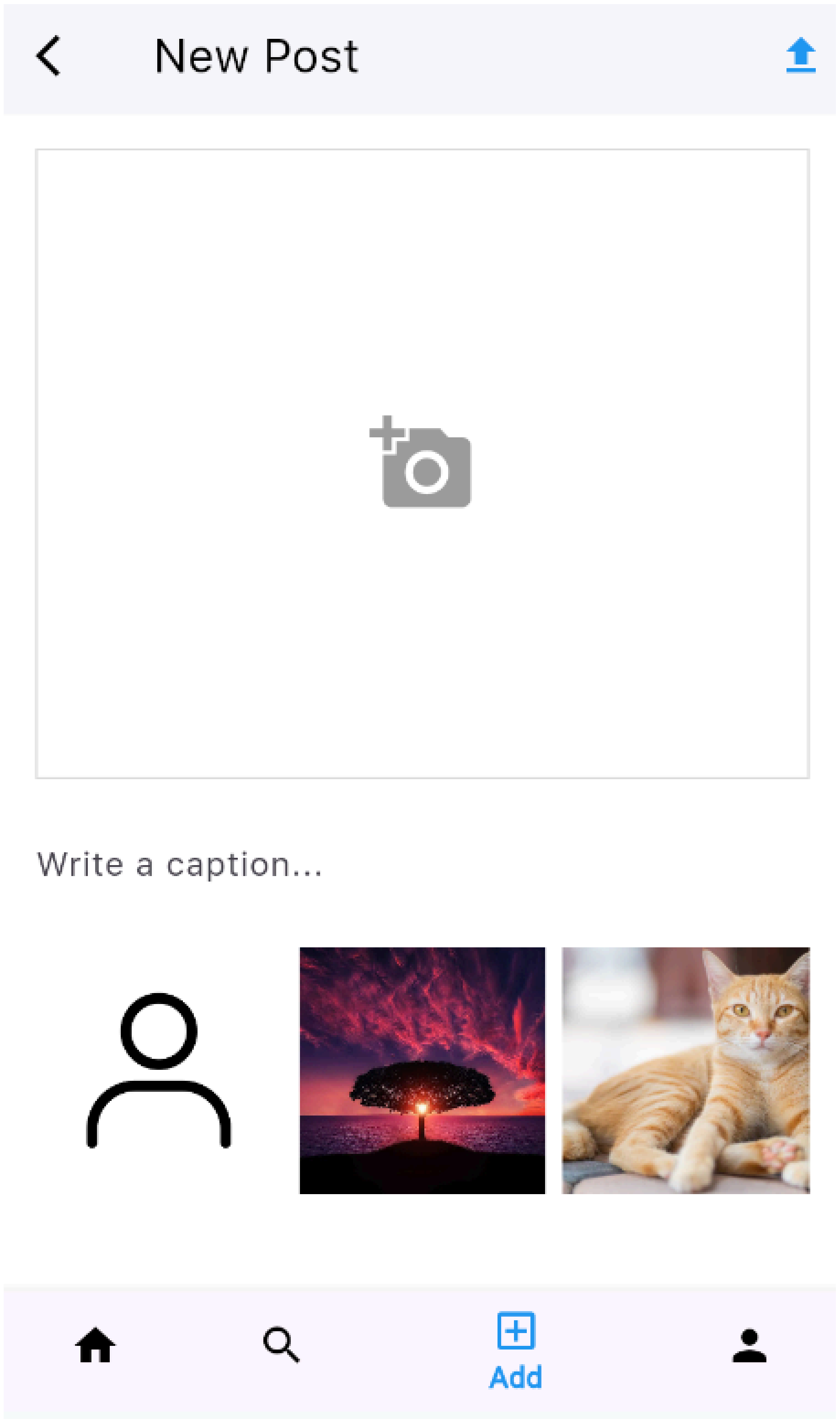
```
List<String> _previousImageAssets = [];

@override
void initState() {
  super.initState();
  _loadPreviousImages();
}

Future<void> _loadPreviousImages() async {
  setState(() {
    _previousImageAssets = [
      'images/person.png', // Replace with your actual image paths
      'images/post.jpg', // Make sure these images are in your assets
      'images/cat.png', // Replace with a valid image in your assets
    ];
  });
}

Future<void> _pickImage(ImageSource source) async {
  final pickedFile = await ImagePicker().pickImage(source: source);

  setState(() {
    if (pickedFile != null) {
      _image = File(pickedFile.path);
    } else {
      print('No image selected.');
    }
  });
}

void _uploadPost() async {
  setState(() {
    _isLoading = true;
  });

  if (_image != null) {
    String caption = _captionController.text;

    // Simulate upload (replace with your actual logic)
    await Future.delayed(const Duration(seconds: 2));
```

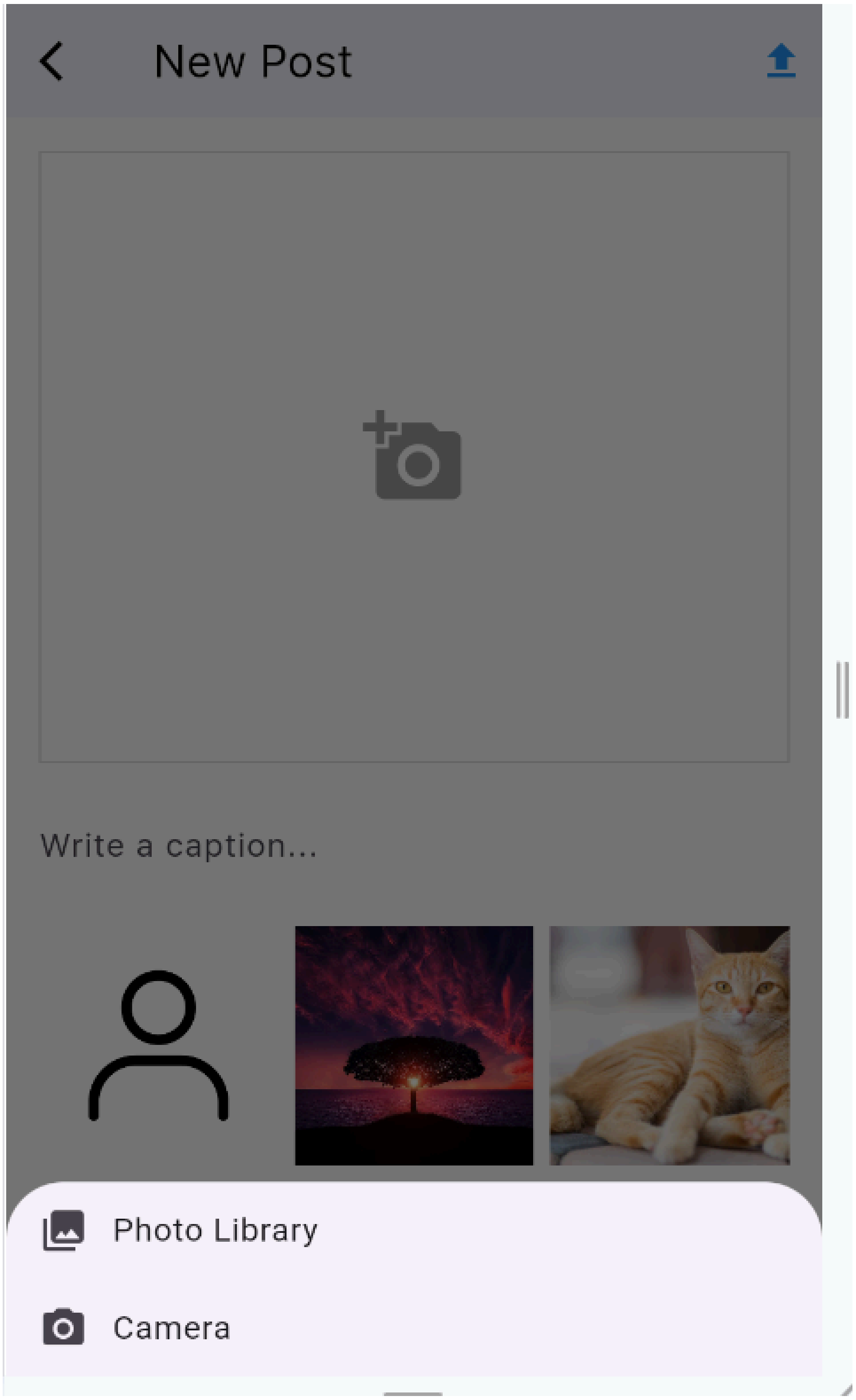
```
      print("Uploading image: ${_image!.path}");
      print("Caption: $caption");

      setState(() {
        _image = null;
        _captionController.clear();
        _isLoading = false;
      });

      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('Post uploaded successfully!')),
      );
    } else {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('Please select an image')),
      );
      setState(() {
        _isLoading = false;
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.white,
        elevation: 1,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back_ios, color: Colors.black),
          onPressed: () {
            Navigator.pop(context);
          },
        ),
        title: const Text('New Post',
            style: TextStyle(color: Colors.black, fontFamily:
'SFProText')),
        actions: [
          IconButton(
            onPressed: _isLoading ? null : _uploadPost,
            icon: _isLoading
                ? const CircularProgressIndicator()
                : const Icon(Icons.upload, color: Colors.blue),
          ),
        ],
      ),
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              GestureDetector(
                onTap: () {
                  showModalBottomSheet(
                    context: context,
                    builder: (BuildContext context) {
                      return Column(
                        mainAxisSize: MainAxisSize.min,
                        children: <Widget>[
                          ListTile(
                            leading: const Icon(Icons.photo_library),
                            title: const Text('Photo Library'),
                            onTap: () {
                              _pickImage(ImageSource.gallery);
                              Navigator.pop(context);
                            },
                          ),
                          ListTile(
                            leading: const Icon(Icons.camera_alt),
                            title: const Text('Camera'),
                            onTap: () {
                              _pickImage(ImageSource.camera);
                              Navigator.pop(context);
                            },
                          ),
                        ],
                      );
```

```
                    },
                  );
                },
                child: Container(
                  height: 300,
                  width: double.infinity,
                  decoration: BoxDecoration(
                    border: Border.all(color: Colors.grey.shade300),
                  ),
                  child: _image != null
                      ? Image.file(_image!, fit: BoxFit.cover)
                      : const Center(
                          child: Icon(Icons.add_a_photo,
                              size: 50, color: Colors.grey)),
                ),
              ),
              const SizedBox(height: 16),
              TextField(
                controller: _captionController,
                style: const TextStyle(fontFamily: 'SFProText'),
                decoration: const InputDecoration(
                  hintText: 'Write a caption...',
                  hintStyle: TextStyle(fontFamily: 'SFProText'),
                  border: InputBorder.none,
                ),
                maxLines: null,
              ),
              const SizedBox(height: 16),
              GridView.builder(
                shrinkWrap: true,
                physics: const NeverScrollableScrollPhysics(),
                gridDelegate: const
SliverGridDelegateWithFixedCrossAxisCount(
                  crossAxisCount: 3,
                  crossAxisSpacing: 8,
                  mainAxisSpacing: 8,
                ),
                itemCount: _previousImageAssets.length,
                itemBuilder: (context, index) {
                  return Image.asset(
                    _previousImageAssets[index],
                    fit: BoxFit.cover,
                  );
                },
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

OUTPUT:-




New Post
Write a caption...
Add

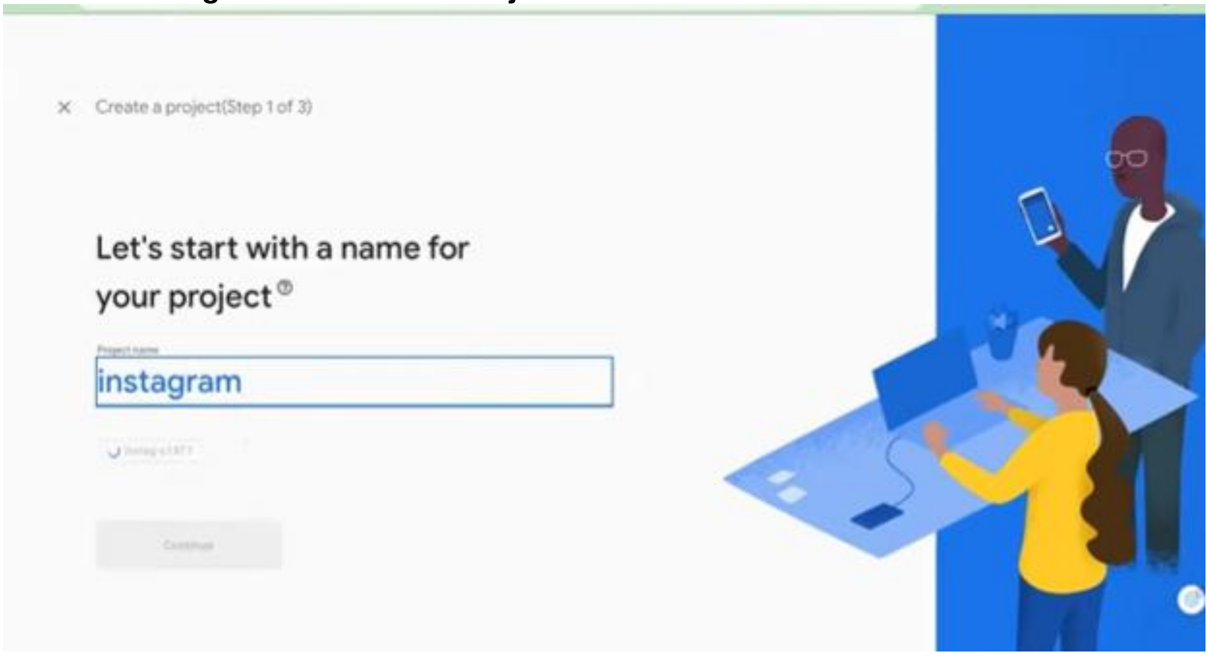

New Post
Write a caption...
Photo Library
Camera

MPL EXPERIMENT NO:- 6

Name:-Sanket More D15A Roll-no:-29

Aim:- To Connect flutter UI with firebase database

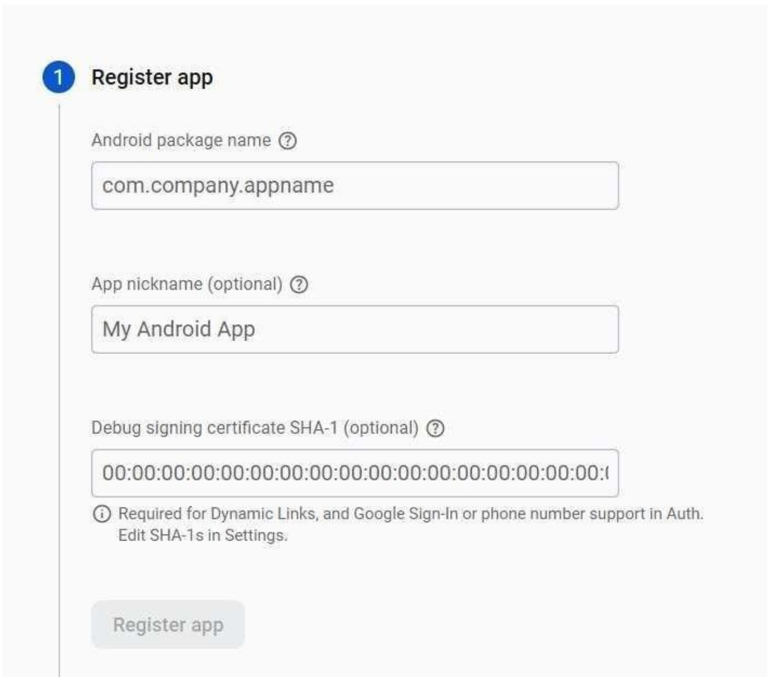
Creating a New Firebase Project



× Create a project(Step 1 of 3)
Let's start with a name for your project
Project name
instagram
Continue

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:



1 Register app

Android package name

com.company.appname

App nickname (optional)

My Android App

Debug signing certificate SHA-1 (optional)

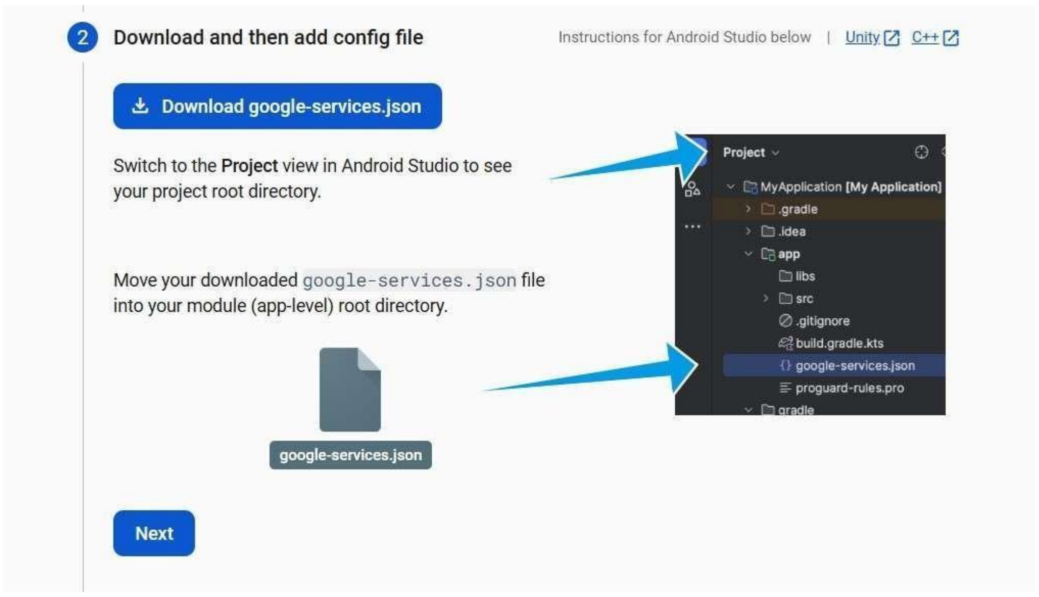
00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:(

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

Register app

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

Then download the google-services.json file, that you will get.


2 Download and then add config file

Instructions for Android Studio below | Unity C++

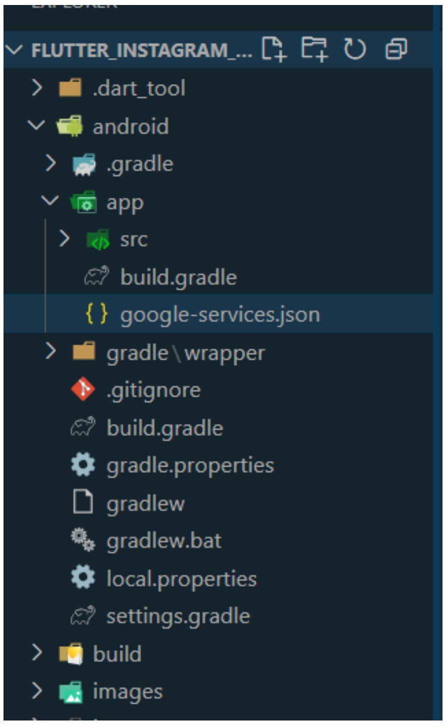
Download google-services.json

Switch to the **Project** view in Android Studio to see your project root directory.

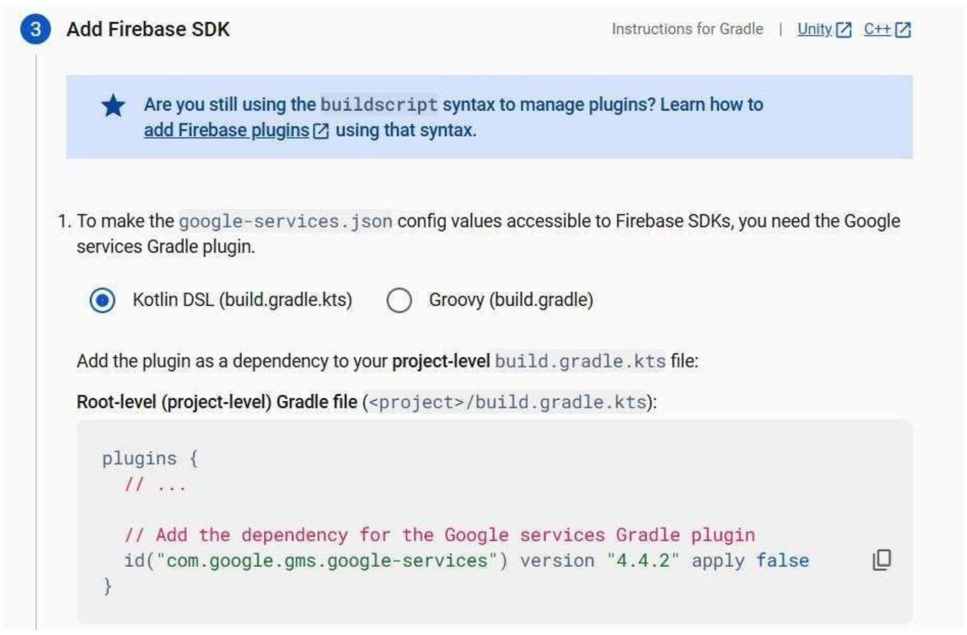
Move your downloaded `google-services.json` file into your module (app-level) root directory.

Next

put that file in the android folder (root level)



then select the build.gradle.kts (Kotlin DSL) part, and then follow the rest instructions



3 **Add Firebase SDK**

Instructions for Gradle | Unity C++

★ **Are you still using the `buildscript` syntax to manage plugins? Learn how to add Firebase plugins using that syntax.**

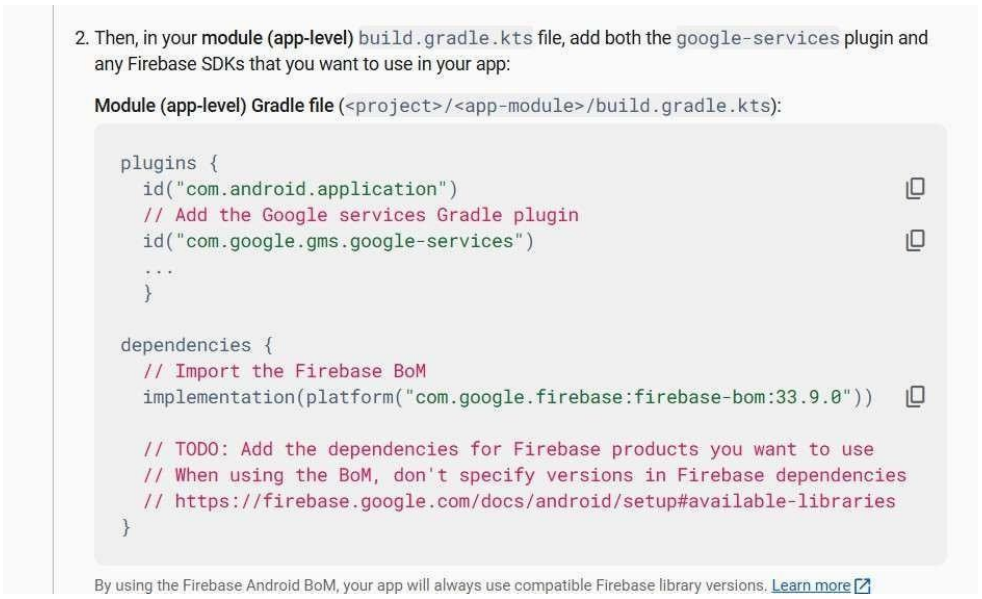
1. To make the `google-services.json` config values accessible to Firebase SDKs, you need the Google services Gradle plugin.

   [✓ Kotlin DSL (build.gradle.kts)] Groovy (build.gradle)

   Add the plugin as a dependency to your **project-level** `build.gradle.kts` file:

   Root-level (project-level) Gradle file (`<project>/build.gradle.kts`):

```
plugins {
  // ...

  // Add the dependency for the Google services Gradle plugin
  id("com.google.gms.google-services") version "4.4.2" apply false
}
```

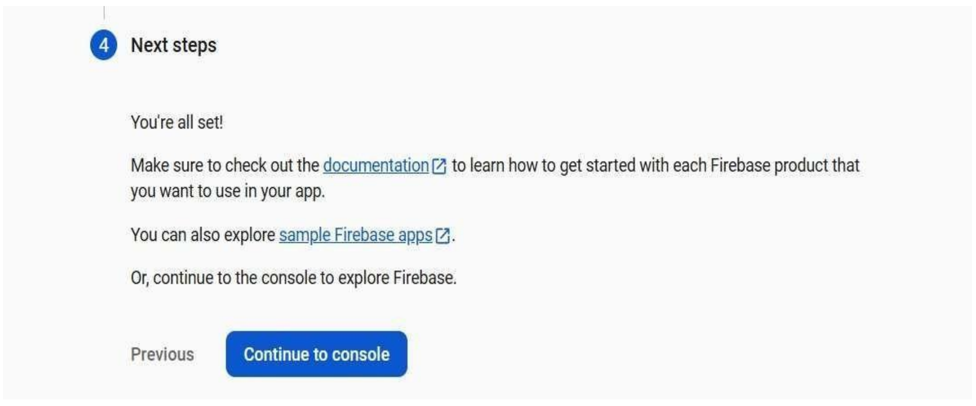


2. Then, in your **module (app-level)** `build.gradle.kts` file, add both the `google-services` plugin and any Firebase SDKs that you want to use in your app:

   Module (app-level) Gradle file (`<project>/<app-module>/build.gradle.kts`):

```
plugins {
  id("com.android.application")
  // Add the Google services Gradle plugin
  id("com.google.gms.google-services")
  ...
  }

dependencies {
  // Import the Firebase BoM
  implementation(platform("com.google.firebase:firebase-bom:33.9.0"))

  // TODO: Add the dependencies for Firebase products you want to use
  // When using the BoM, don't specify versions in Firebase dependencies
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

By using the Firebase Android BoM, your app will always use compatible Firebase library versions. Learn more



4 **Next steps**

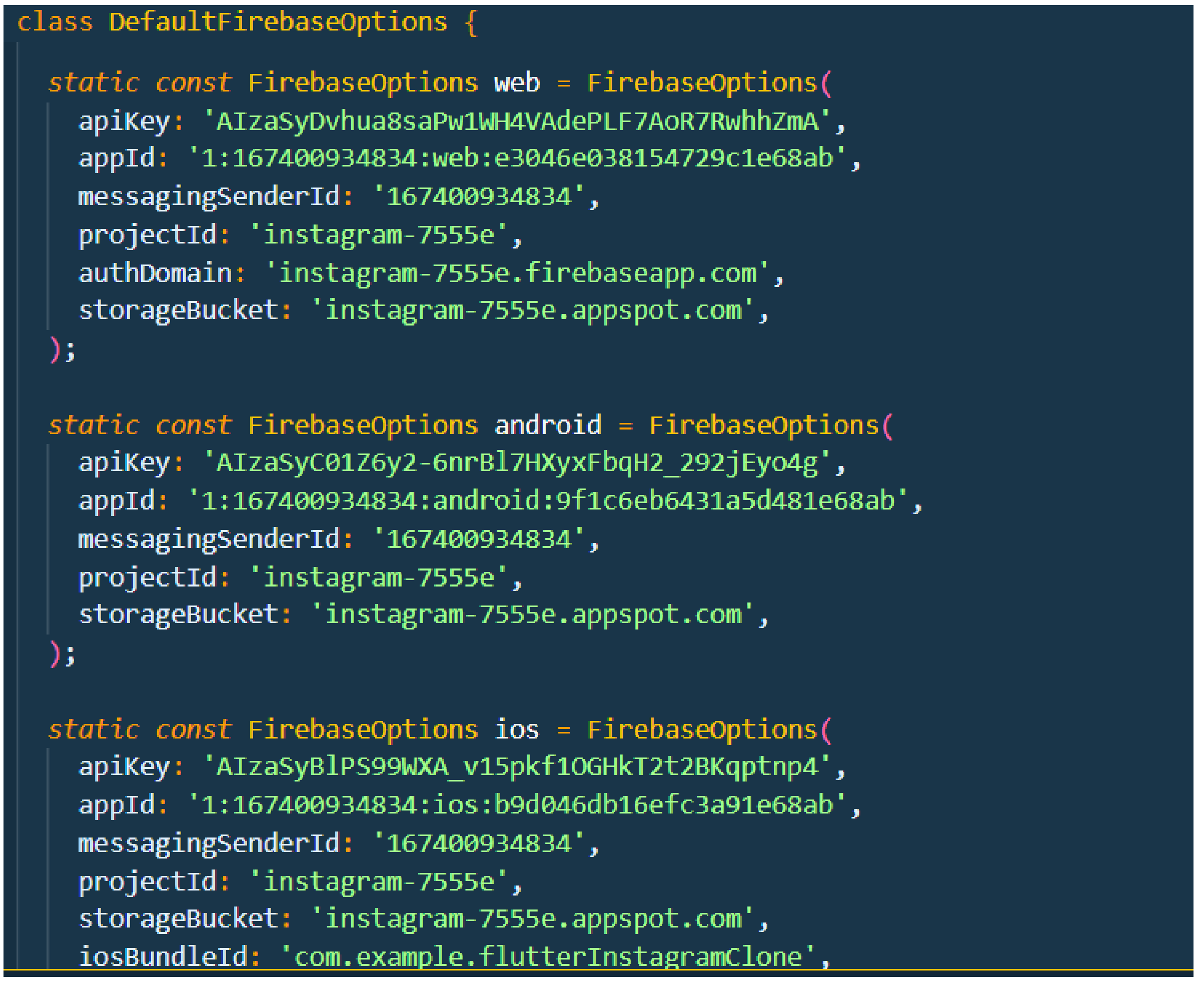
You're all set!

Make sure to check out the documentation to learn how to get started with each Firebase product that you want to use in your app.

You can also explore sample Firebase apps.

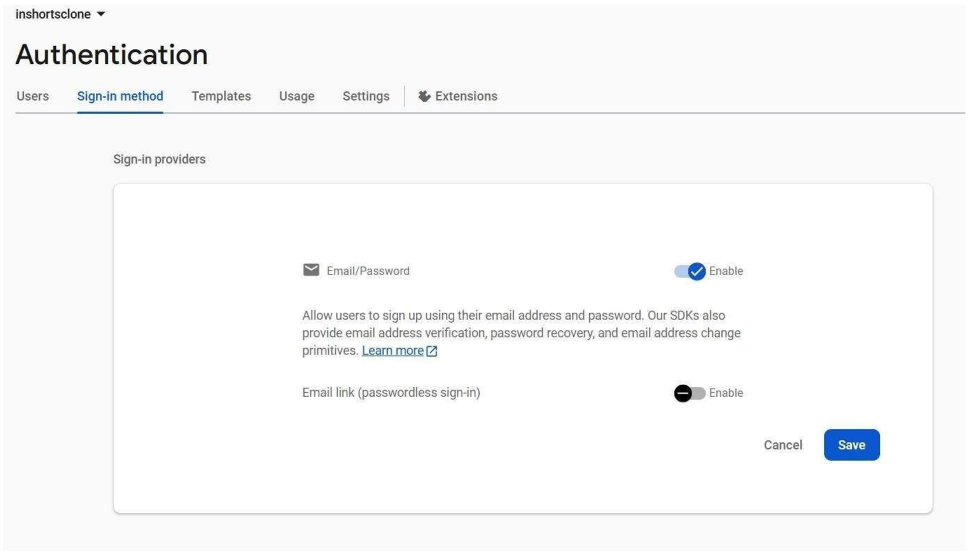
Or, continue to the console to explore Firebase.

Generate the firebase_options.dart file, based on the google-services.json file



```
class DefaultFirebaseOptions {

  static const FirebaseOptions web = FirebaseOptions(
    apiKey: 'AIzaSyDvhua8saPw1WH4VAdePLF7AoR7RwhhZmA',
    appId: '1:167400934834:web:e3046e038154729c1e68ab',
    messagingSenderId: '167400934834',
    projectId: 'instagram-7555e',
    authDomain: 'instagram-7555e.firebaseapp.com',
    storageBucket: 'instagram-7555e.appspot.com',
  );

  static const FirebaseOptions android = FirebaseOptions(
    apiKey: 'AIzaSyC01Z6y2-6nrBl7HXyxFbqH2_292jEyo4g',
    appId: '1:167400934834:android:9f1c6eb6431a5d481e68ab',
    messagingSenderId: '167400934834',
    projectId: 'instagram-7555e',
    storageBucket: 'instagram-7555e.appspot.com',
  );

  static const FirebaseOptions ios = FirebaseOptions(
    apiKey: 'AIzaSyBlPS99WXA_v15pkf1OGHkT2t2BKqptnp4',
    appId: '1:167400934834:ios:b9d046db16efc3a91e68ab',
    messagingSenderId: '167400934834',
    projectId: 'instagram-7555e',
    storageBucket: 'instagram-7555e.appspot.com',
    iosBundleId: 'com.example.flutterInstagramClone',
```

In your part select the sign-in method and enable it.



inshortsclone

Authentication

Users | Sign-in method | Templates | Usage | Settings | Extensions

Sign-in providers

Email/Password — Enable [✓]

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery, and email address change primitives. Learn more

Email link (passwordless sign-in) — Enable

Cancel | Save

Code:- Authenction_logic

```
import 'dart:io';


import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter_instagram_clone/data/firebase_service/firestor.dart';
import 'package:flutter_instagram_clone/data/firebase_service/storage.dart';


import 'package:flutter_instagram_clone/util/exeption.dart';


class Authentication {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  Future<void> Login({
    required String email,
    required String password,
  }) async {
    try {
      await _auth.signInWithEmailAndPassword(
          email: email.trim(), password: password.trim());
    } on FirebaseException catch (e) {
      throw exceptions(e.message.toString());
    }
  }


  Future<void> Signup({
    required String email,
    required String password,
    required String passwordConfirme,
    required String username,
    required String bio,
    required File profile,
  }) async {
    String URL;
```

```
try {
  if (email.isNotEmpty &&
      password.isNotEmpty &&
      username.isNotEmpty &&
      bio.isNotEmpty) {
    if (password == passwordConfirme) {
      // create user with email and password
      await _auth.createUserWithEmailAndPassword(
        email: email.trim(),
        password: password.trim(),
      );
      // upload profile image on storage

      if (profile != File('')) {
        URL =
            await StorageMethod().uploadImageToStorage('Profile', profile);
      } else {
        URL = '';
      }

      // get information with firestor

      await Firebase_Firestor().CreateUser(
        email: email,
        username: username,
        bio: bio,
        profile: URL == ''
            ? 'https://firebasestorage.googleapis.com/v0/b/instagram-
8a227.appspot.com/o/person.png?alt=media&token=c6fcbe9d-f502-4aa1-8b4b-ec37339e78ab'
            : URL,
      );
    } else {
      throw exceptions('password and confirm password should be same');
```

```
          }
        } else {
          throw exceptions('enter all the fields');
        }
      } on FirebaseException catch (e) {
        throw exceptions(e.message.toString());
      }
    }
  }
```

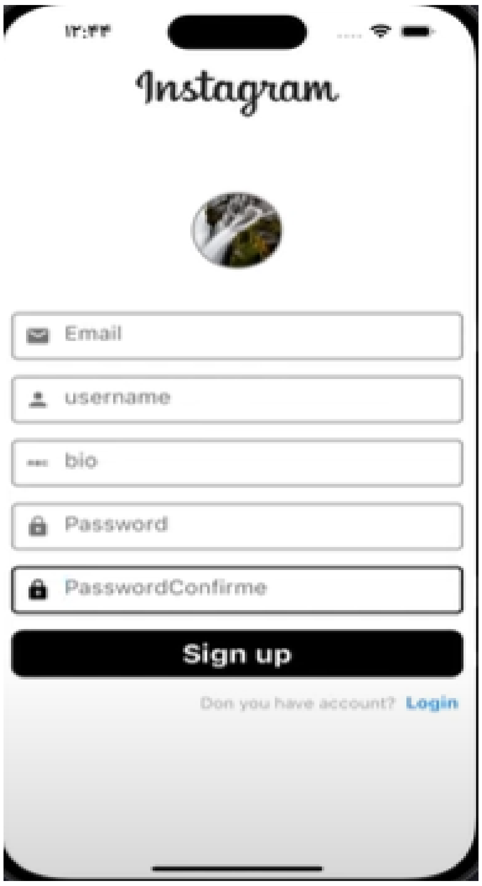
Login_screen

```
import 'dart:io';

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter_instagram_clone/data/firebase_service/firestor.dart';
import 'package:flutter_instagram_clone/data/firebase_service/storage.dart';

import 'package:flutter_instagram_clone/util/exeption.dart';

class Authentication {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  Future<void> Login({
    required String email,
    required String password,
  }) async {
    try {
      await _auth.signInWithEmailAndPassword(
          email: email.trim(), password: password.trim());
    } on FirebaseException catch (e) {
      throw exceptions(e.message.toString());
    }
  }

  Future<void> Signup({
    required String email,
    required String password,
    required String passwordConfirme,
    required String username,
    required String bio,
    required File profile,
  }) async {
    String URL;
    try {
      if (email.isNotEmpty &&
          password.isNotEmpty &&
          username.isNotEmpty &&
          bio.isNotEmpty) {
        if (password == passwordConfirme) {
          // create user with email and password
          await _auth.createUserWithEmailAndPassword(
            email: email.trim(),
            password: password.trim(),
          );
          // upload profile image on storage

          if (profile != File('')) {
            URL =
                await StorageMethod().uploadImageToStorage('Profile', profile);
          } else {
            URL = '';
          }

          // get information with firestor

          await Firebase_Firestor().CreateUser(
            email: email,
            username: username,
            bio: bio,
            profile: URL == ''
                ? 'https://firebasestorage.googleapis.com/v0/b/instagram-
8a227.appspot.com/o/person.png?alt=media&token=c6fcbe9d-f502-4aa1-8b4b-ec37339e78ab'
                : URL,
          );
        } else {
          throw exceptions('password and confirm password should be same');
        }
      } else {
        throw exceptions('enter all the fields');
      }
    } on FirebaseException catch (e) {
      throw exceptions(e.message.toString());
    }
  }
}
```

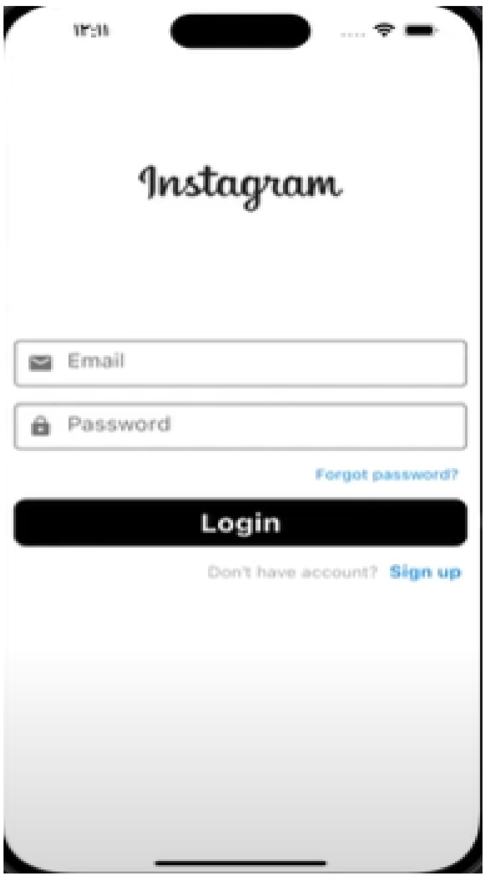
Home_Screen

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:flutter_instagram_clone/widgets/post_widget.dart';
import 'package:flutter_screenutil/flutter_screenutil.dart';

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

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  FirebaseAuth _auth = FirebaseAuth.instance;
  FirebaseFirestore _firebaseFirestore = FirebaseFirestore.instance;
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      resizeToAvoidBottomInset: false,
      backgroundColor: Colors.white,
      appBar: AppBar(
        centerTitle: true,
        elevation: 0,
        title: SizedBox(
          width: 105.w,
          height: 28.h,
          child: Image.asset('images/instagram.jpg'),
        ),
        leading: Image.asset('images/camera.jpg'),
        actions: [
          const Icon(
            Icons.favorite_border_outlined,
            color: Colors.black,
            size: 25,
          ),
          Image.asset('images/send.jpg'),
        ],
        backgroundColor: const Color(0xffFAFAFA),
      ),
      body: CustomScrollView(
        slivers: [
          StreamBuilder(
            stream: _firebaseFirestore
                .collection('posts')
                .orderBy('time', descending: true)
                .snapshots(),
            builder: (context, snapshot) {
              return SliverList(
                delegate: SliverChildBuilderDelegate(
                  (context, index) {
                    if (!snapshot.hasData) {
                      return Center(child: CircularProgressIndicator());
                    }
                    return PostWidget(snapshot.data!.docs[index].data());
                  },
                  childCount:
                      snapshot.data == null ? 0 : snapshot.data!.docs.length,
                ),
              );
            },
          )
        ],
      ),
    );
  }
}
```

OUTPUT :



Instagram
Email
username
bio
Password
PasswordConfirme
Sign up
Don you have account? Login



Instagram
Email
Password
Forgot password?
Login
Don't have account? Sign up