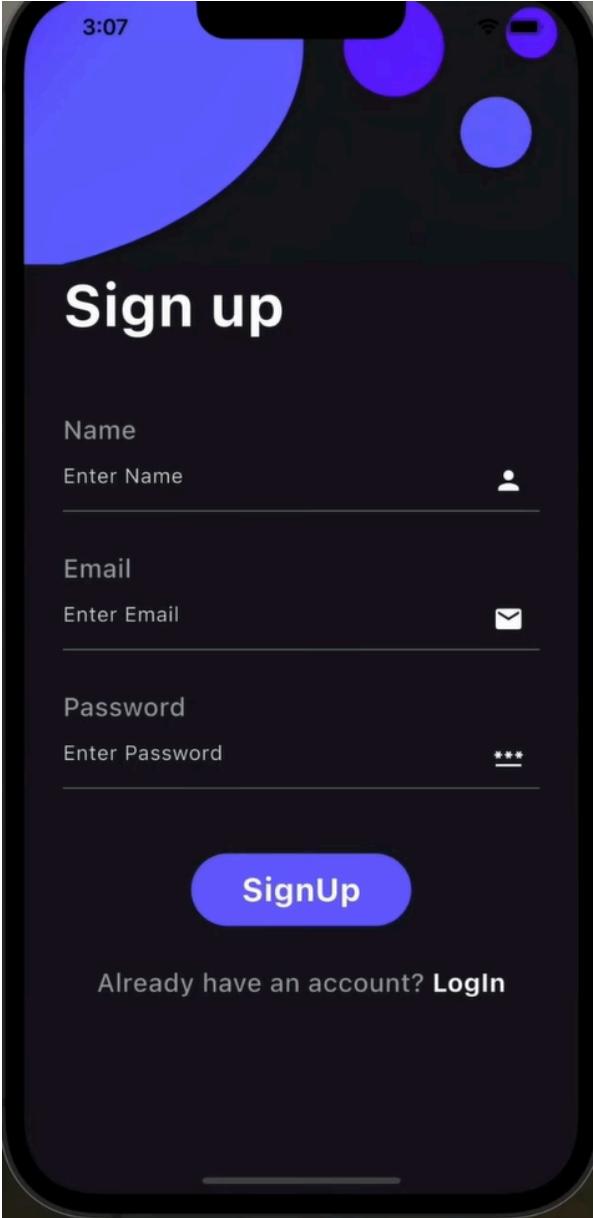


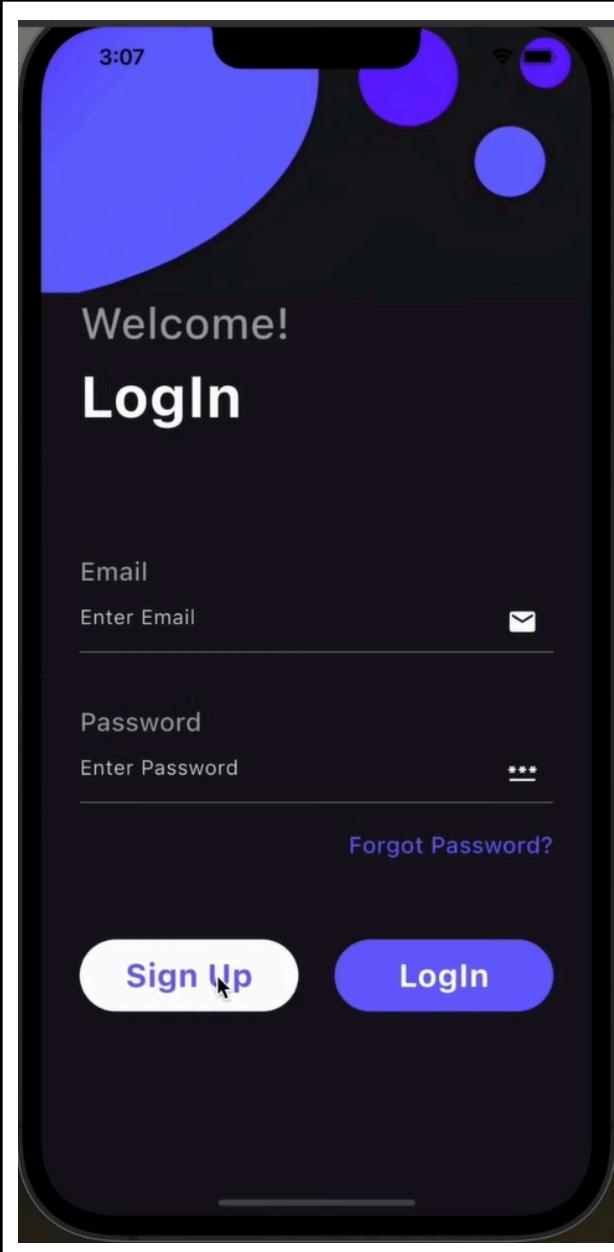
MPL Prerequisites Document

Name: Manorath Ital

Class : D15A

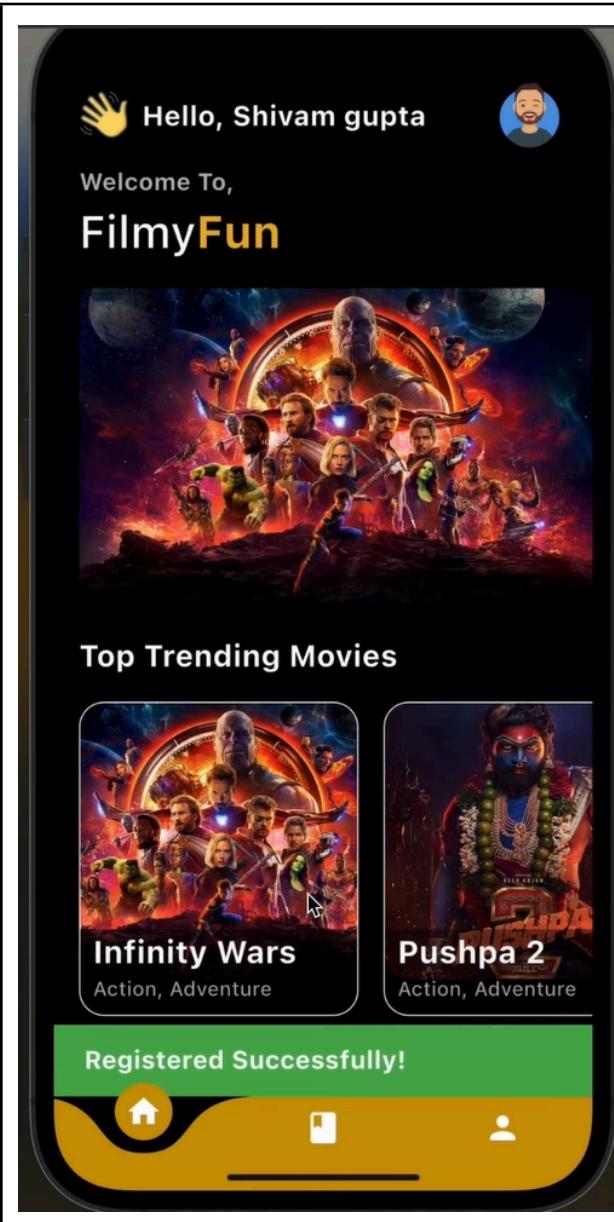
Roll no: 19

Screenshot	Features
	<p>Signup Page Features & Implementation:</p> <ol style="list-style-type: none">User Input Fields (Name, Email, Password) – Implemented using <code>TextField</code> in Flutter with validation.Icons for Input Fields – Added using <code>Icon</code> widget within <code>InputDecoration</code>.Password Visibility Toggle – Managed using <code>ObscureText</code> and <code>IconButton</code>.Signup Button – Created using <code>ElevatedButton</code> with Firebase authentication.Login Navigation – Implemented using <code>Navigator.push()</code> to switch to the login screen.UI Styling – Designed using <code>Container</code>, <code>Column</code>, <code>Padding</code>, and <code>BoxDecoration</code>.Firebase Authentication – <code>FirebaseAuth.createUserWithEmailAndPassword()</code> for user registration.



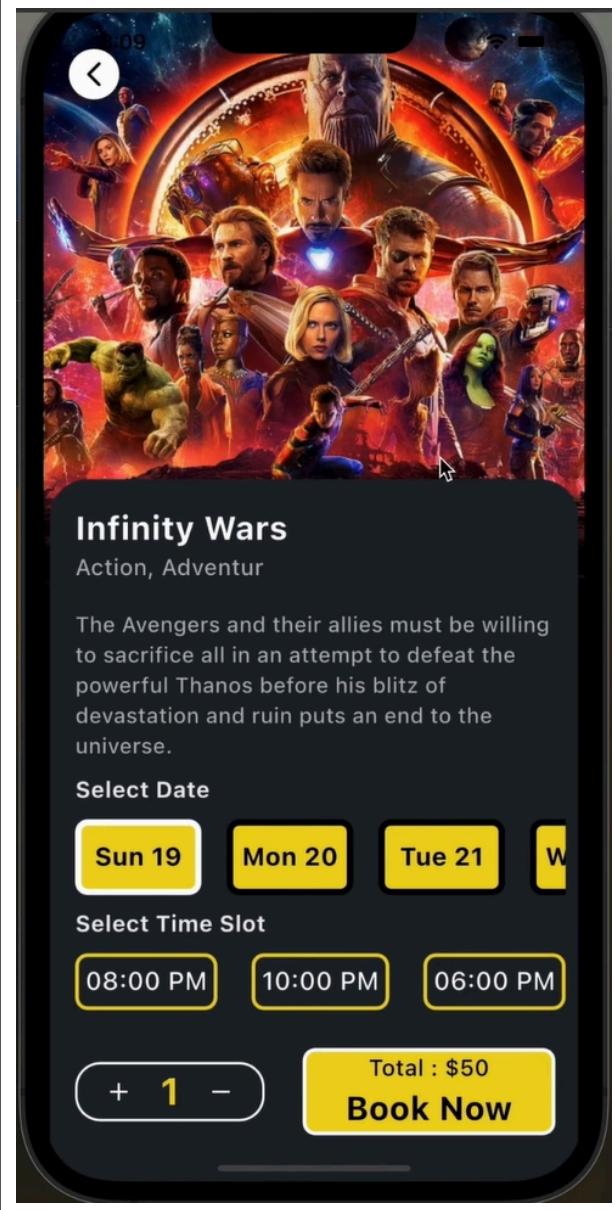
Login Page Features & Implementation:

1. **User Input Fields (Email, Password)** – Implemented using `TextField` with validation.
2. **Icons for Input Fields** – Integrated using `InputDecoration` with `Icon`.
3. **Password Visibility Toggle** – Managed using `ObscureText` and `IconButton`.
4. **Forgot Password Link** – Implemented using `GestureDetector` to navigate to password reset.
5. **Login & Signup Buttons** – Created using `ElevatedButton` and `OutlinedButton` with `onPressed` for navigation.
6. **UI Styling** – Designed using `Container`, `Column`, `Padding`, and `BoxDecoration`.
7. **Firebase Authentication** – `FirebaseAuth.signInWithEmailAndPassword()` for user login



Home Page Features & Implementation:

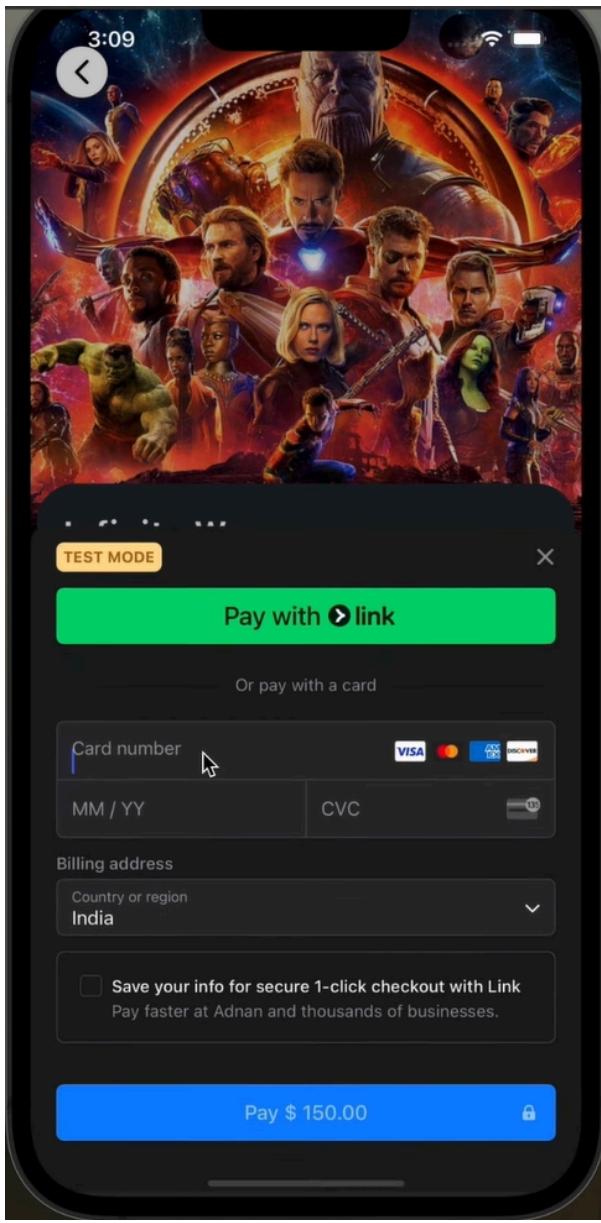
1. **User Greeting & Avatar** – Displayed using `Row` with `Text` and `CircleAvatar`, fetched from Firebase Authentication.
2. **App Branding (FilmyFun)** – Styled using `Text.rich` for multiple fonts and colors.
3. **Top Trending Movies Section** – Implemented using `ListView.builder` with Firebase Firestore integration.
4. **Movie Cards** – Built using `Container` with `ClipRRect` for rounded images and `Text` for movie details.
5. **Bottom Navigation Bar** – Created using `BottomNavigationBar` with icons for Home and Profile.
6. **SnackBar Notification (Registered Successfully!)** – Managed using `ScaffoldMessenger.showSnackBar()`.
7. **Smooth UI & Styling** – Achieved using `Column`, `Padding`, and `BoxDecoration`.



Movie Details Page

Features

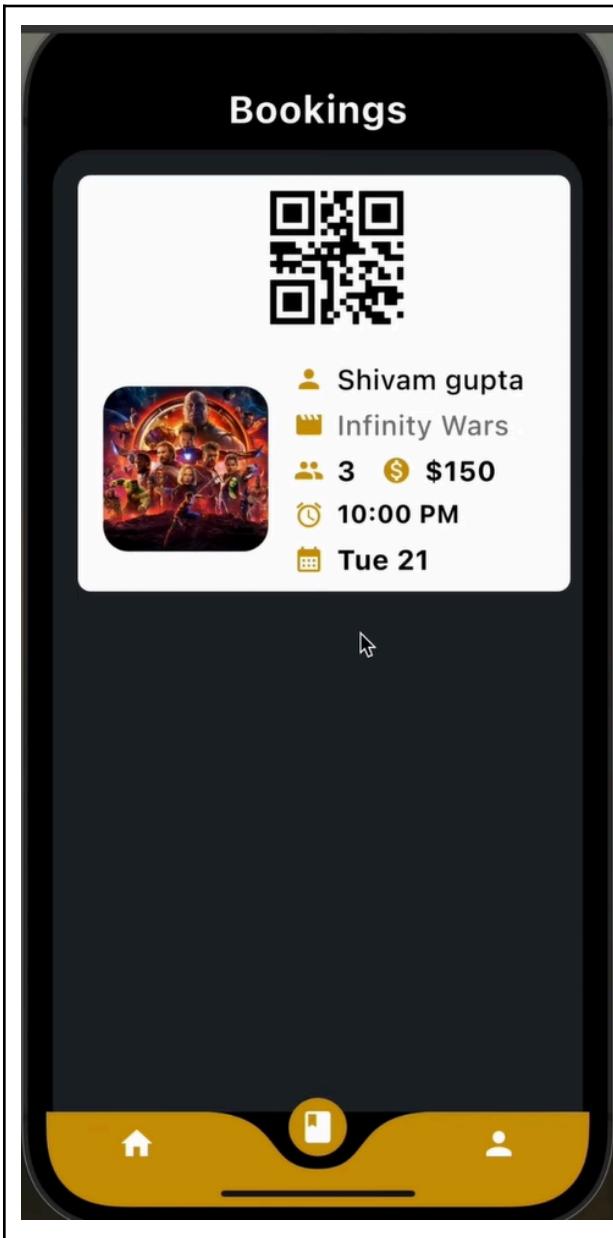
1. **Large Movie Poster** – Displays the movie prominently.
2. **Movie Title & Genre** – Clearly shown for easy identification.
3. **Short Movie Description** – Provides a brief overview of the movie.
4. **Date Selection** – Allows users to choose a preferred show date.
5. **Time Slot Selection** – Lets users select a specific time for the show.
6. **Ticket Quantity Selector** – Enables users to increase or decrease ticket count.
7. **Total Price Calculation** – Updates dynamically based on the selected number of tickets.
8. **Book Now Button** – A clear call-to-action for booking tickets.



Payment Page

Features

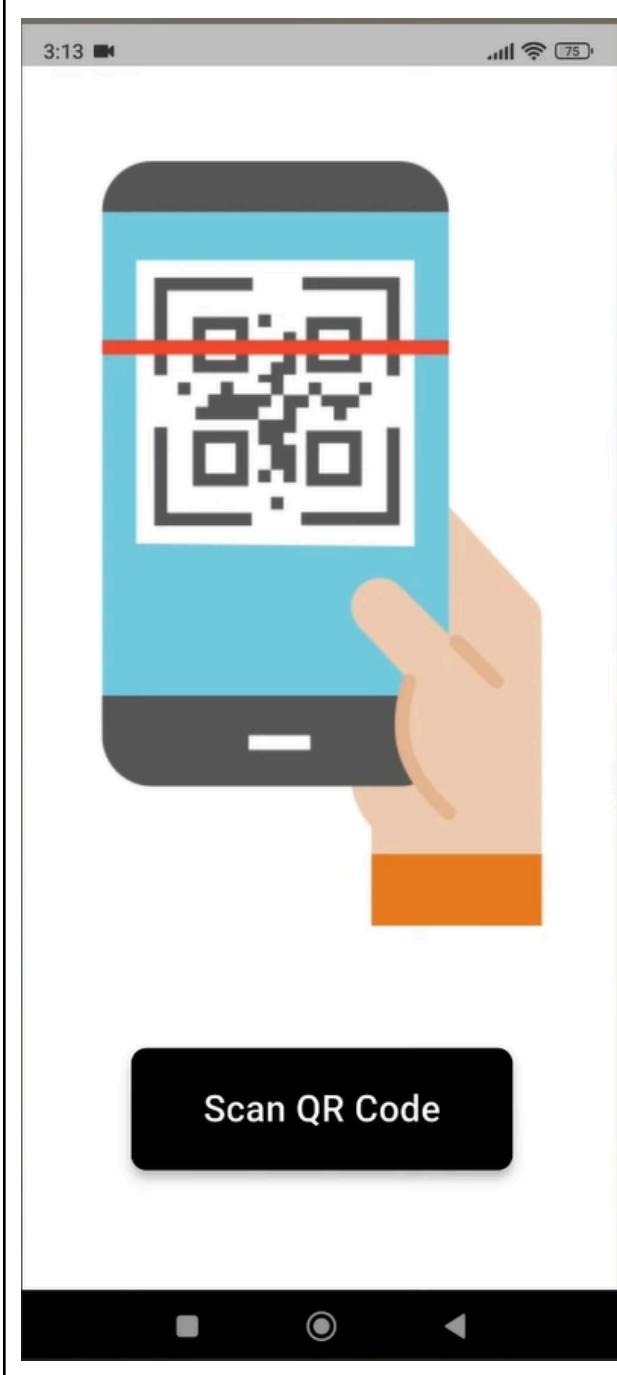
- **Movie Poster & Title** – Displays the selected movie.
- **Total Amount** – Shows the total price of the selected tickets.
- **Payment Options** –
 - Pay with a linked account (e.g., UPI, Wallets).
 - Pay using a credit/debit card.
- **Card Payment Form** –
 - Card number input.
 - Expiry date (MM/YY) field.
 - CVC security code input.
- **Billing Address Selection** – Users can choose their country/region.
- **Secure Checkout Option** – Users can save payment details for faster future transactions.
- **Pay Button** – Finalizes the payment process.



Booking Page

Features

1. **Header** – Displays "Bookings" at the top.
2. **QR Code** – Allows quick verification for entry.
3. **Booking Details Card** – Includes:
 - o **User Name** – Shows the name of the person who booked the ticket.
 - o **Movie Poster & Title** – Displays the booked movie.
 - o **Number of Tickets** – Indicates the total seats booked.
 - o **Total Price** – Shows the total amount paid.
 - o **Showtime** – Displays the selected movie time.
 - o **Date of Booking** – Shows the movie date.
4. **Bottom Navigation Bar** – Includes icons for navigating to home, booking history, and other sections.



QR Code Scanner Page

Features

1. **Illustration** – Shows a phone scanning a QR code for user guidance.
2. **Scan QR Code Button** – Triggers the device's camera to scan a QR code.
3. **Minimal UI** – Clean, simple, and easy to use.

EXPERIMENT-1

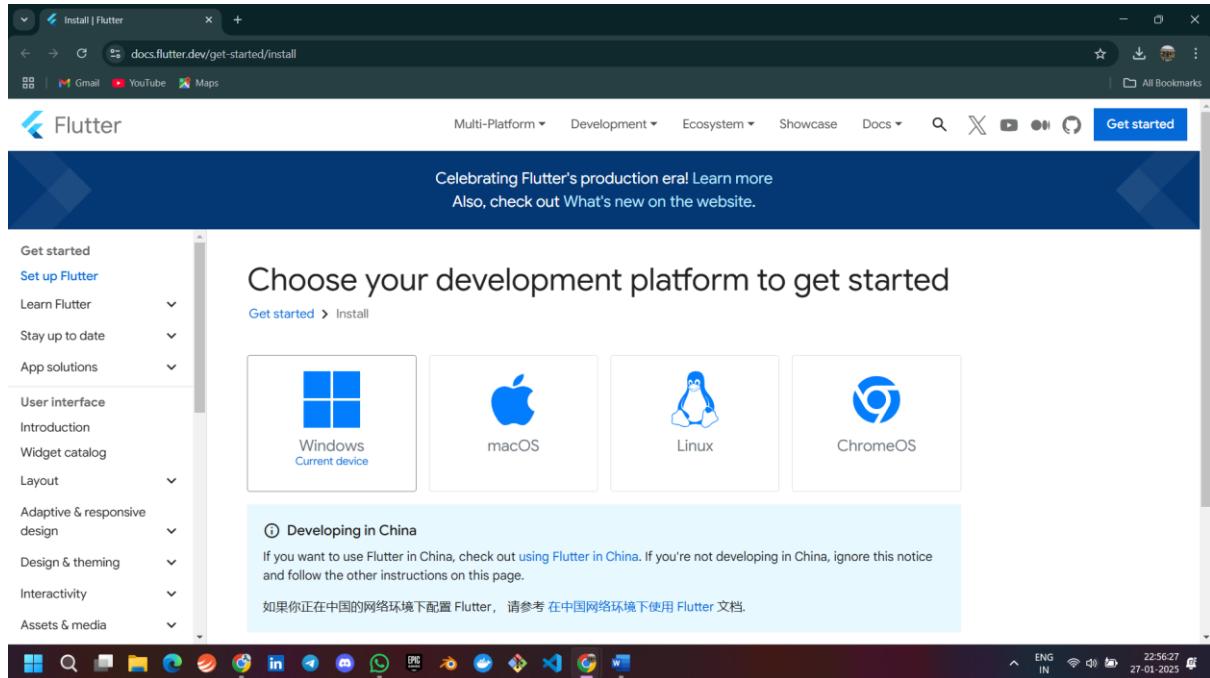
NAME: MANORATH ITAL

CLASS: D15A

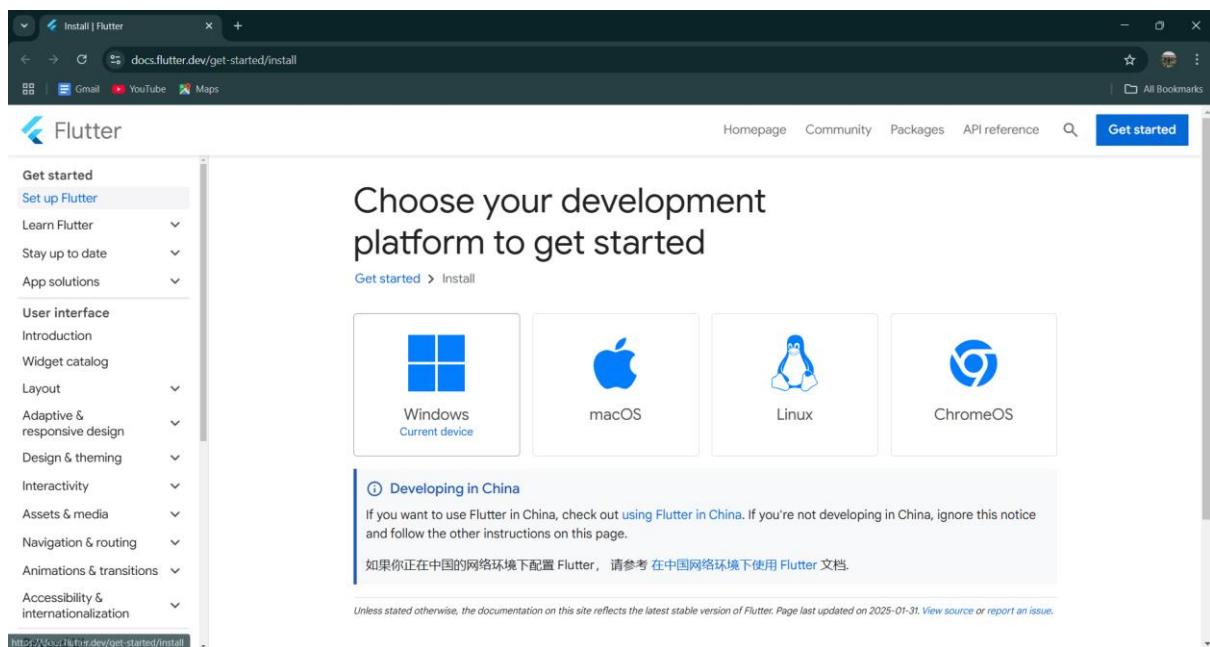
ROLLNO: 19

AIM: Installation and Configuration of Flutter Environment.

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



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



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

Download then install Flutter

To install Flutter, download the Flutter SDK bundle from its archive, move the bundle to where you want it stored, then extract the SDK.

1. Download the following installation bundle to get the latest stable release of the Flutter SDK.
[flutter_windows_3.27.3-stable.zip](#)
2. Create a folder where you can install Flutter.

Consider creating a directory at `%USERPROFILE% (C:\Users\{username}) or %LOCALAPPDATA% (C:\Users\{username}\AppData\Local)`.

Warning

Don't install Flutter to a directory or path that meets one or both of the following conditions:

- The path contains special characters or spaces.
- The path requires elevated privileges.

Contents

- Verify system requirements
- Hardware requirements
- Software requirements
- Configure a text editor or IDE
- Install the Flutter SDK
- Configure Android development
- Configure the Android toolchain in Android Studio
- Configure your target Android device
- Agree to Android licenses
- Check your development setup
- Run Flutter doctor
- Troubleshoot Flutter doctor issues
- Start developing Android

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

AukZip: Zip & RAR Extractor

Extract

Add Archive

Clear List

Preview

Select an item to preview

Output Path C:\Flutter

Extract All **Extract Selected Items**

Settings

Feedback

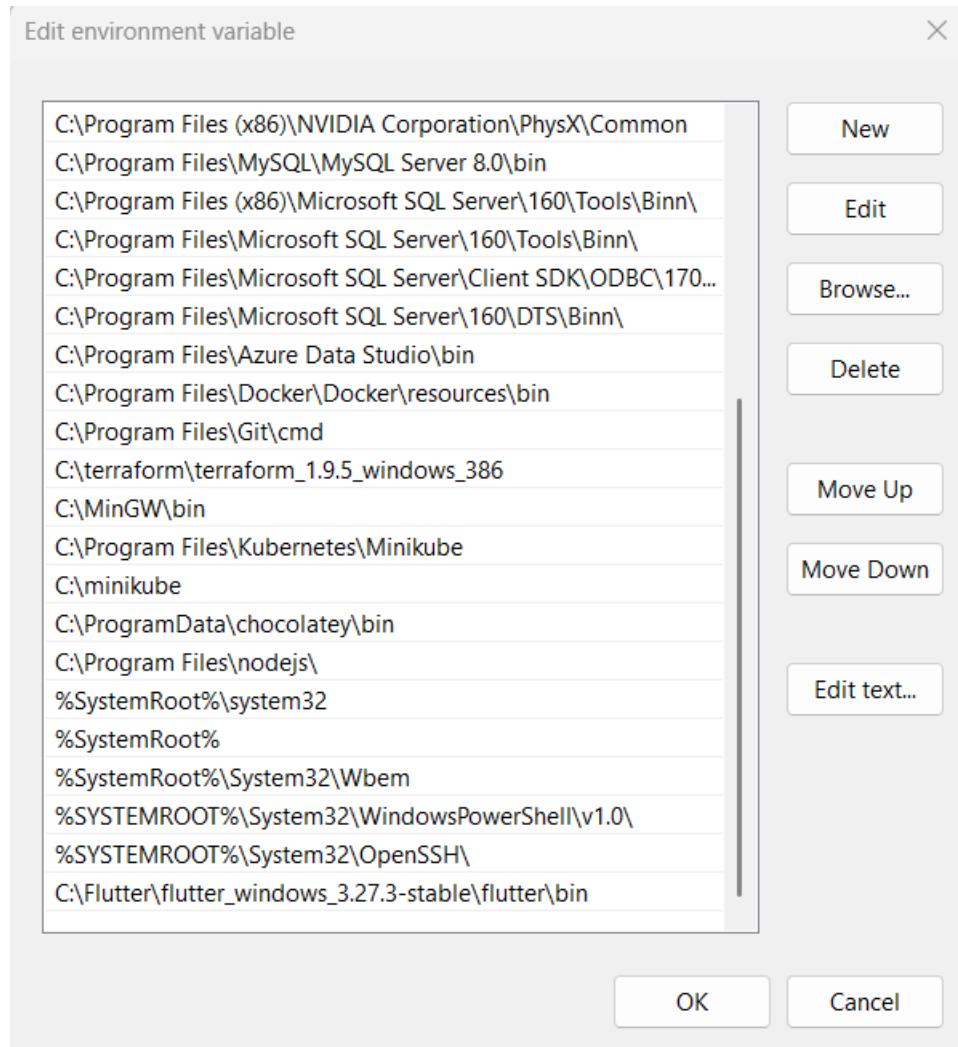
About

Step 5 :- Add Flutter to System PATH

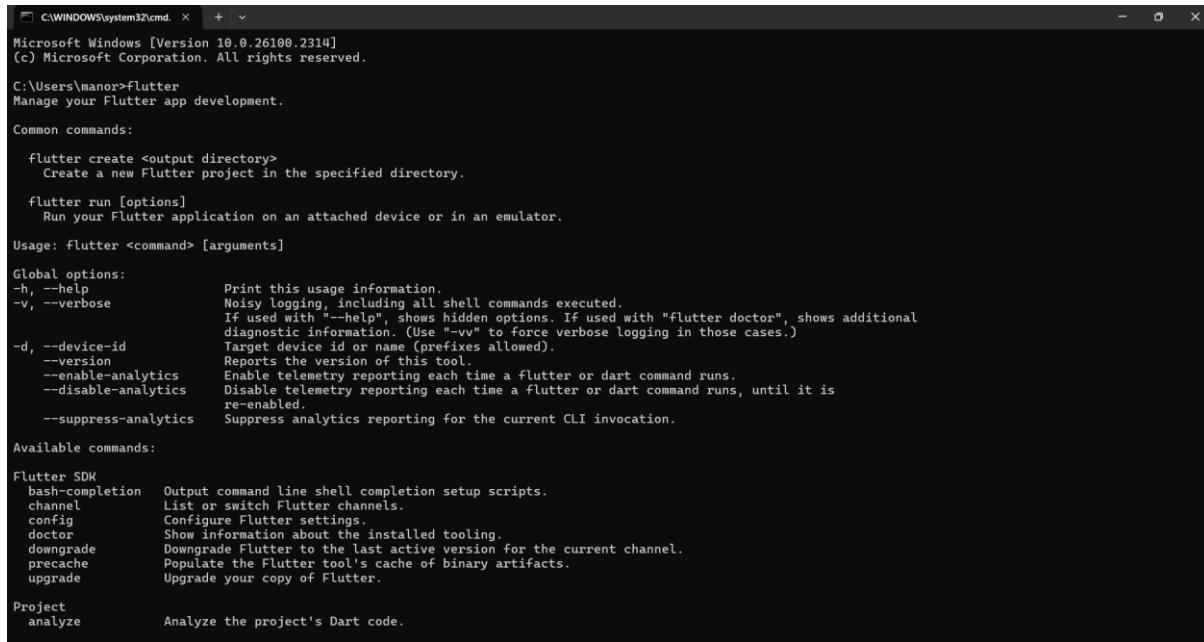
Right-click on the Start Menu > System > Advanced system settings > Environment Variables.

Under System Variables, find Path and click Edit.

Add the full path to the flutter/bin directory (e.g., C:\flutter\bin).



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



```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.26100.2314]
(c) Microsoft Corporation. All rights reserved.

C:\Users\manor>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>]

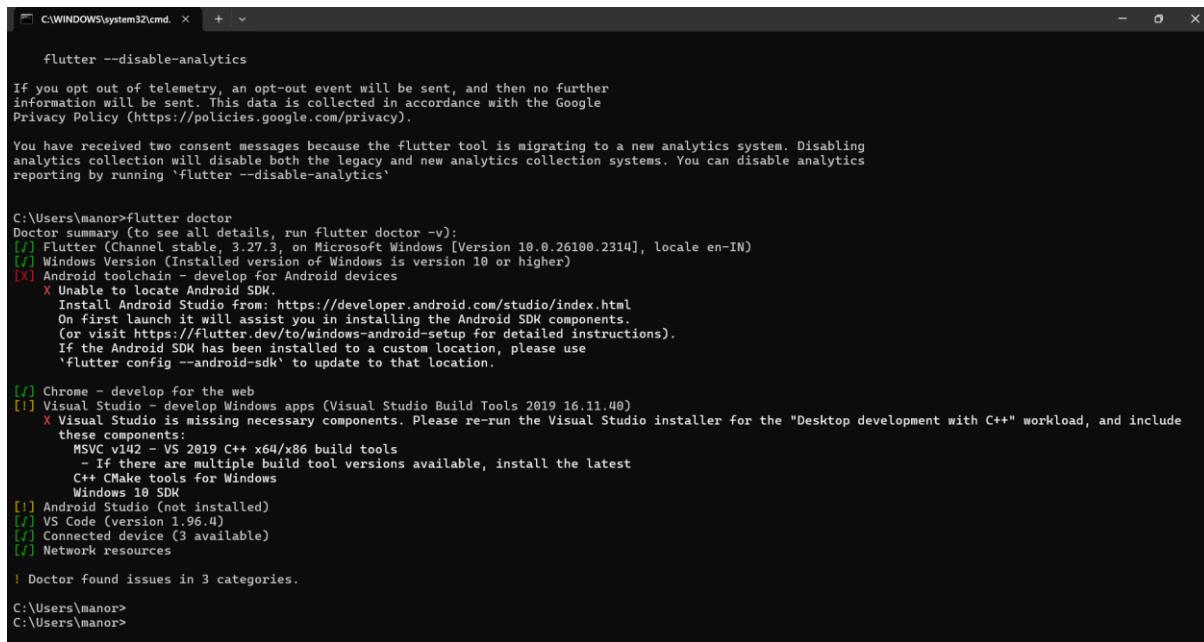
Global options:
  -h, --help           Print this usage information.
  -v, --verbose        Noisy logging, including all shell commands executed.
                      If used with "--help", shows hidden options. If used with "flutter doctor", shows additional
                      diagnostic information. (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.

Available commands:

Flutter SDK
  bash-completion  Output command line shell completion setup scripts.
  channel          List or switch Flutter channels.
  config           Configure Flutter settings.
  doctor           Show information about the installed tooling.
  downgrade        Downgrade Flutter to the last active version for the current channel.
  precache         Populate the Flutter tool's cache of binary artifacts.
  upgrade          Upgrade your copy of Flutter.

Project
  analyze          Analyze the project's Dart code.
```

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



```
C:\WINDOWS\system32\cmd. x + v
flutter --disable-analytics

If you opt out of telemetry, an opt-out event will be sent, and then no further
information will be sent. This data is collected in accordance with the Google
Privacy Policy (https://policies.google.com/privacy).

You have received two consent messages because the flutter tool is migrating to a new analytics system. Disabling
analytics collection will disable both the legacy and new analytics collection systems. You can disable analytics
reporting by running 'flutter --disable-analytics'

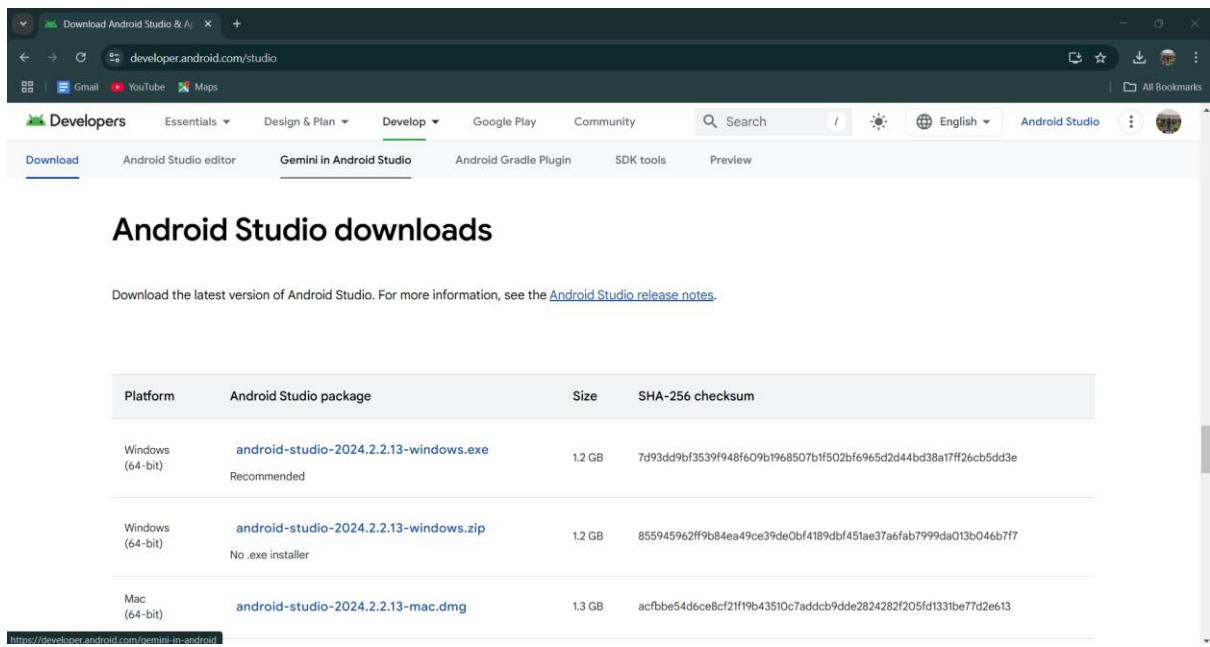
C:\Users\manor>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[!] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.26100.2314], locale en-IN)
[!] Windows Version (Installed version of Windows is version 10 or higher)
[!] Android toolchain - develop for Android devices
  X Unable to locate Android SDK.
    Install Android Studio from: https://developer.android.com/studio/index.html
    On first launch it will assist you in installing the Android SDK components.
    (Or visit https://flutter.dev/to/windows-android-setup for detailed instructions).
    If the Android SDK has been installed to a custom location, please use
    'flutter config --android-sdk' to update to that location.

[!] Chrome - develop for the web
[!] Visual Studio - develop Windows apps (Visual Studio Build Tools 2019 16.11.40)
  X Visual Studio is missing necessary components. Please re-run the Visual Studio installer for the "Desktop development with C++" workload, and include
    these components:
      MSVC V142 - VS 2019 C++ x64/x86 build tools
      - If there are multiple build tool versions available, install the latest
      C++ CMake tools for Windows
  Windows 10 SDK
[!] Android Studio (not installed)
[!] VS Code (version 1.96.4)
[!] Connected device (3 available)
[!] Network resources

! Doctor found issues in 3 categories.

C:\Users\manor>
C:\Users\manor>
```

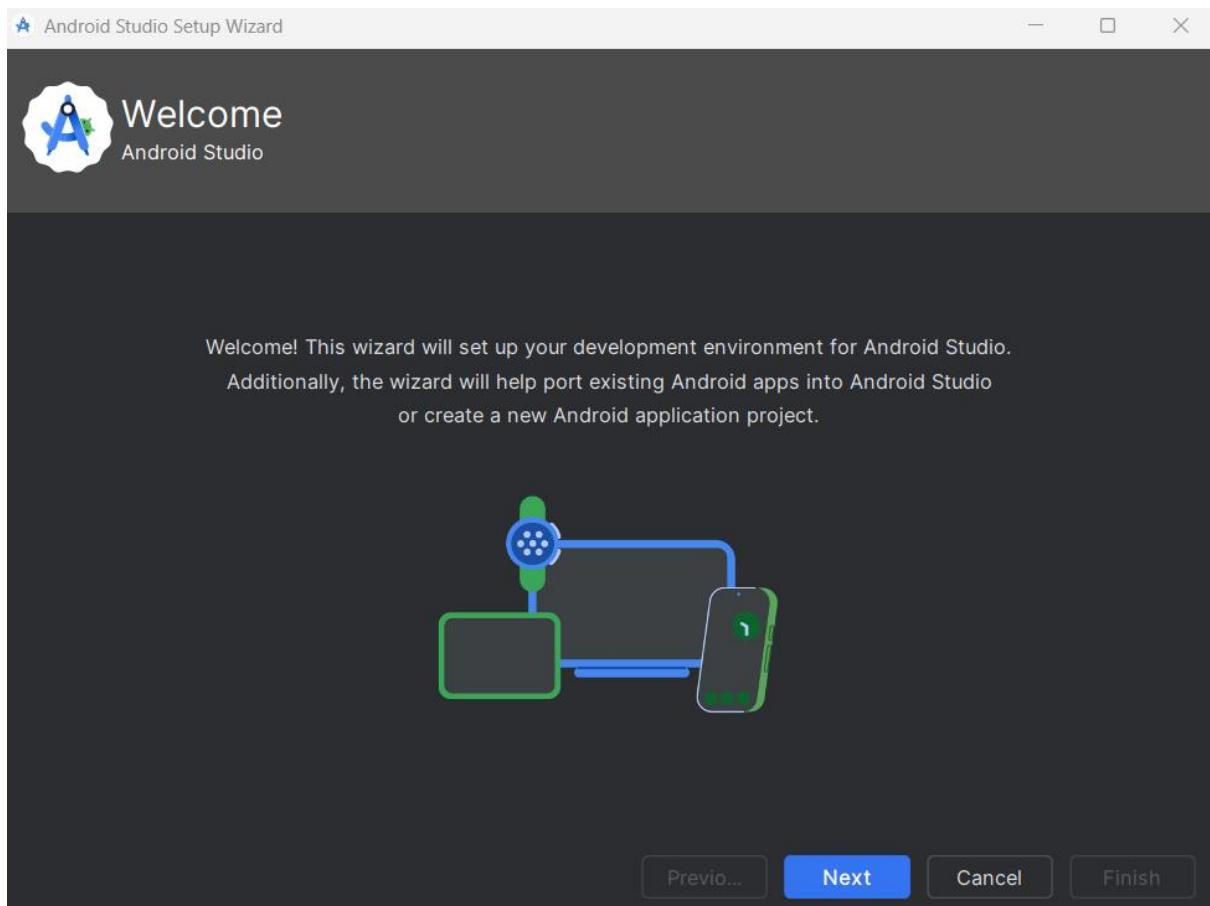
Step 8 : - Go to Android Studio and download the installer.



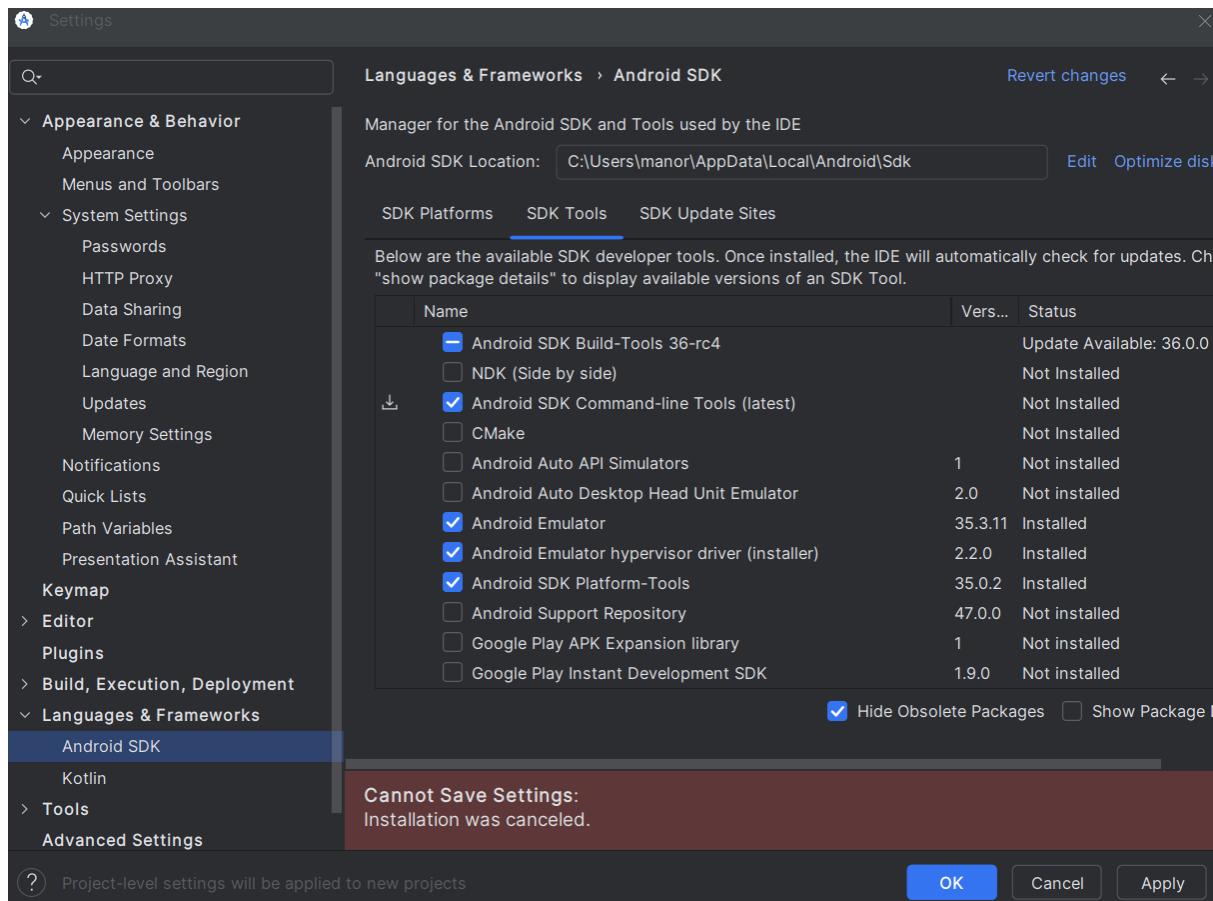
The screenshot shows a web browser window with the URL developer.android.com/studio. The page is titled "Android Studio downloads". It features a table with four columns: Platform, Android Studio package, Size, and SHA-256 checksum. The table lists three packages: Windows (64-bit) with "android-studio-2024.2.2.13-windows.exe" (1.2 GB, Recommended), Windows (64-bit) with "android-studio-2024.2.2.13-windows.zip" (1.2 GB, No .exe installer), and Mac (64-bit) with "android-studio-2024.2.2.13-mac.dmg" (1.3 GB). Below the table is a link to "https://developer.android.com/gemini-in-android".

Platform	Android Studio package	Size	SHA-256 checksum
Windows (64-bit)	android-studio-2024.2.2.13-windows.exe Recommended	1.2 GB	7d93dd9bf3539f948f609b1968507bf502bf6965d2d44bd38a17ff26cb5dd3e
Windows (64-bit)	android-studio-2024.2.2.13-windows.zip No .exe installer	1.2 GB	855945962ff9b84ea49ce39de0bf4189dbf45iae37a6fab7999da013b046b7f7
Mac (64-bit)	android-studio-2024.2.2.13-mac.dmg	1.3 GB	acfbbbe54d6ce8cf21f19b43510c7addcb9dde2824282f205fd1331be77d2e613

Step 9: - When the download is complete, open the .exe file and run it. Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.



Step 10: - Go to Preferences > Appearance & Behavior > System Settings > Android SDK. Select the SDK Tools tab and check Android SDK Command-line Tools and Install it.



Step 9: - Open a terminal and run the following command

1) flutter doctor --android-licenses

```
C:\Users\manor>flutter doctor --android-licenses
Warning: Additionally, the fallback loader failed to parse the XML.
Warning: Errors during XML parse: ] 62% Fetch remote repository...
Warning: Additionally, the fallback loader failed to parse the XML.ry...
[=====] 100% Computing updates...
6 of 7 SDK package licenses not accepted.
Review licenses that have not been accepted (y/N)? y
1/6: License android-googletv-license:
Terms and Conditions
This is the Google TV Add-on for the Android Software Development Kit License Agreement.
1. Introduction
1.1 The Google TV Add-on for the Android Software Development Kit (referred to in this License Agreement as the "Google TV Add-on" and specifically including the Android system files, packaged APIs, and Google APIs add-ons) is licensed to you subject to the terms of this License Agreement. This License Agreement forms a legally binding contract between you and Google in relation to your use of the Google TV Add-on.
1.2 "Google" means Google Inc., a Delaware corporation with principal place of business at 1600 Amphitheatre Parkway, Mountain View, CA 94043, United States.
2. Accepting this License Agreement
2.1 In order to use the Google TV Add-on, you must first agree to this License Agreement. You may not use the Google TV Add-on if you do not accept this License Agreement.
2.2 You can accept this License Agreement by:
(A) clicking to accept or agree to this License Agreement, where this option is made available to you; or
(B) by actually using the Google TV Add-on. In this case, you agree that use of the Google TV Add-on constitutes acceptance of the License Agreement from that point onwards.
2.3 You may not use the Google TV Add-on and may not accept the Licensing Agreement if you are a person barred from receiving the Google TV Add-on under the laws of the United States or other countries including the country in which you are resident or from which you use the Google TV Add-on.
2.4 If you are agreeing to be bound by this License Agreement on behalf of your employer or other entity, you represent and warrant that you have full legal
```

Accept all the licenses

```
C:\WINDOWS\system32\cmd. x + v

en agreements previously existing between Recipient and MIPS with respect to the subject matter hereof. This Agreement may only be amended or supplemented b
y a writing that refers explicitly to this Agreement and that is signed or otherwise accepted by duly authorized representatives of Recipient and MIPS.

10.5 Severability. In the event that any provision of this Agreement is finally adjudicated to be unenforceable or invalid under any applicable law, such un
enforceability or invalidity shall not render this Agreement unenforceable or invalid as a whole, and, in such event, such unenforceable or invalid provisio
n shall be interpreted so as to best accomplish the objectives of such provision within the limits of applicable law or applicable court decisions.

10.6 Export Regulations / Export Control. Recipient shall not export, either directly or indirectly, any product, service or technical data or system incorporating the Evaluation Materials without first obtaining any required license or other necessary approval from the U.S. Department of Commerce or any other governing agency or department of the United States Government. In the event any product is exported from the United States or re-exported from a foreign destination by Recipient, Recipient shall ensure that the distribution and export/re-export or import of the product is in compliance with all applicable laws, regulations, orders, or other restrictions of the U.S. Export Administration Regulations and the appropriate foreign government. Recipient agrees that neither it nor any of its subsidiaries will export/re-export any technical data, process, product, or service, directly or indirectly, to any country for which the United States government or any agency thereof or the foreign government from where it is shipping requires an export license, or other governmental approval, without first obtaining such license or approval. Recipient also agrees to implement measures to ensure that foreign national employees are authorized to receive any information controlled by U.S. export control laws. An export is "deemed" to take place when information is released to a foreign national wherever located.

10.7 Special Terms for Pre-Release Materials. If so indicated in the description of the Evaluation Software, the Evaluation Software may contain Pre-Release Materials. Recipient hereby understands, acknowledges and agrees that: (i) Pre-Release Materials may not be fully tested and may contain bugs or errors; (ii) Pre-Release materials are not suitable for commercial release in their current state; (iii) regulatory approvals for Pre-Release Materials (such as UL or FCC) have not been obtained, and Pre-Release Materials may therefore not be certified for use in certain countries or environments or may not be suitable for certain applications and (iv) MIPS can provide no assurance that it will ever produce or make generally available a production version of the Pre-Release Materials. MIPS is not under any obligation to develop and/or release or offer for sale or license a final product based upon the Pre-Release Materials and may unilaterally elect to abandon the Pre-Release Materials or any such development platform at any time and without any obligation or liability whatsoever to Recipient or any other person.

ANY PRE-RELEASE MATERIALS ARE NON-QUALIFIED AND, AS SUCH, ARE PROVIDED *AS IS* AND *AS AVAILABLE*, POSSIBLY WITH FAULTS, AND WITHOUT REPRESENTATION OR WARRANTY OF ANY KIND.

10.8 Open Source Software. In the event Open Source software is included with Evaluation Software, such Open Source software is licensed pursuant to the applicable Open Source software license agreement identified in the Open Source software comments in the applicable source code file(s) and/or file header as indicated in the Evaluation Software. Additional detail may be available (where applicable) in the accompanying on-line documentation. With respect to the Open Source software, nothing in this Agreement limits any rights under, or grants rights that supersede, the terms of any applicable Open Source software license agreement.

-----
Accept? (y/N): y
All SDK package licenses accepted

C:\Users\manor>
```

2)flutter doctor

```
C:\WINDOWS\system32\cmd. x + v

i) Pre-Release materials are not suitable for commercial release in their current state; (iii) regulatory approvals for Pre-Release Materials (such as UL or FCC) have not been obtained, and Pre-Release Materials may therefore not be certified for use in certain countries or environments or may not be suitable for certain applications and (iv) MIPS can provide no assurance that it will ever produce or make generally available a production version of the Pre-Release Materials. MIPS is not under any obligation to develop and/or release or offer for sale or license a final product based upon the Pre-Release Materials and may unilaterally elect to abandon the Pre-Release Materials or any such development platform at any time and without any obligation or liability whatsoever to Recipient or any other person.

ANY PRE-RELEASE MATERIALS ARE NON-QUALIFIED AND, AS SUCH, ARE PROVIDED *AS IS* AND *AS AVAILABLE*, POSSIBLY WITH FAULTS, AND WITHOUT REPRESENTATION OR WARRANTY OF ANY KIND.

10.8 Open Source Software. In the event Open Source software is included with Evaluation Software, such Open Source software is licensed pursuant to the applicable Open Source software license agreement identified in the Open Source software comments in the applicable source code file(s) and/or file header as indicated in the Evaluation Software. Additional detail may be available (where applicable) in the accompanying on-line documentation. With respect to the Open Source software, nothing in this Agreement limits any rights under, or grants rights that supersede, the terms of any applicable Open Source software license agreement.

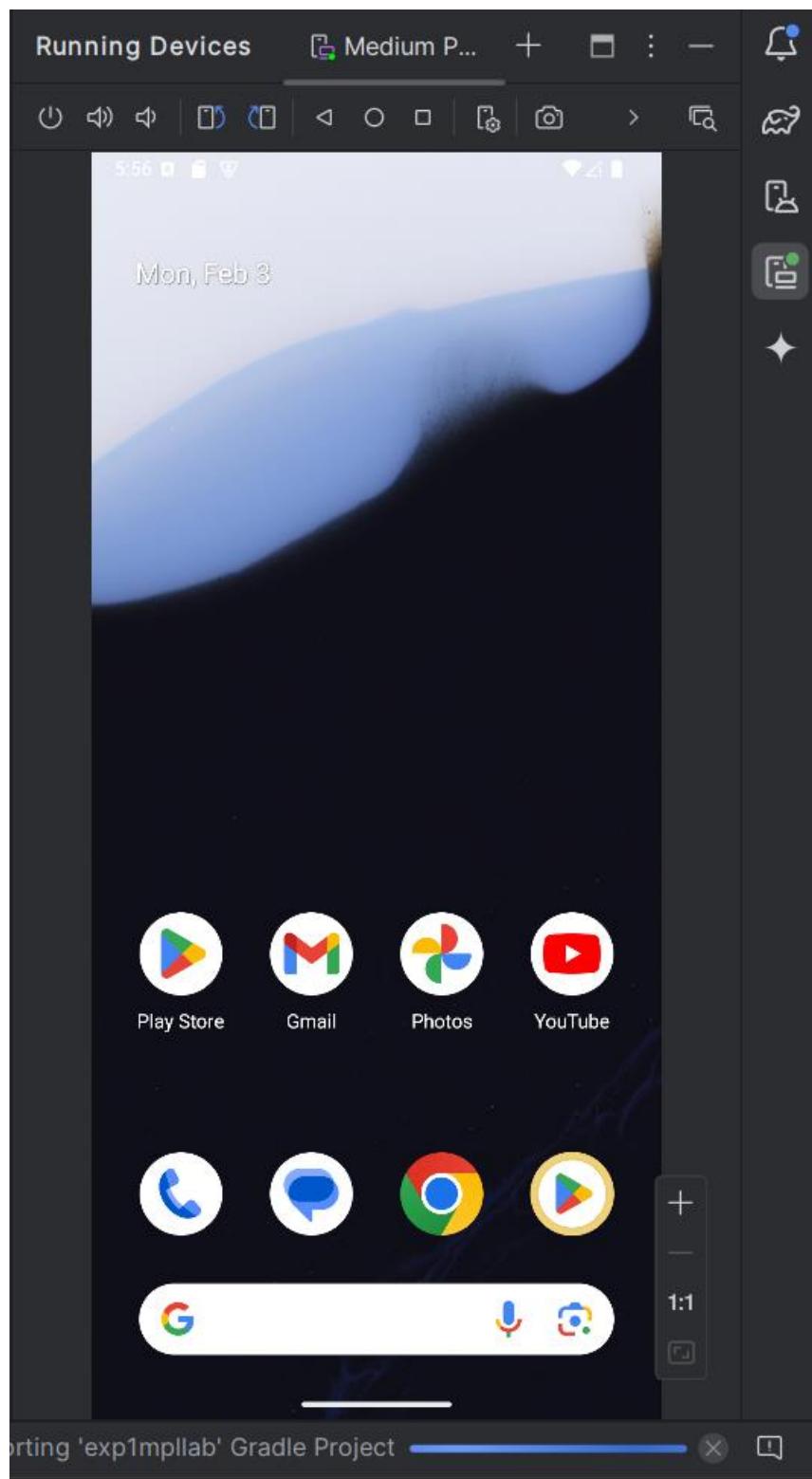
-----
Accept? (y/N): y
All SDK package licenses accepted

C:\Users\manor> flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[!] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.26100.2314], locale en-IN)
[!] Windows Version (Installed version of Windows is version 10 or higher)
[!] Android toolchain - develop for Android devices (Android SDK version 35.0.1)
[!] Chrome - develop for the web
[!] Visual Studio - develop Windows apps (Visual Studio Build Tools 2019 16.11.40)
  X Visual Studio is missing necessary components. Please re-run the Visual Studio installer for the "Desktop development with C++" workload, and include these components:
    - MSVC v142 - VS 2019 C++ x64/x86 build tools
      - If there are multiple build tool versions available, install the latest C++ CMake tools for Windows
      Windows 10 SDK
[!] Android Studio (Version 2024.2)
[!] VS Code (version 1.96.4)
[!] Connected device (3 available)
[!] Network resources

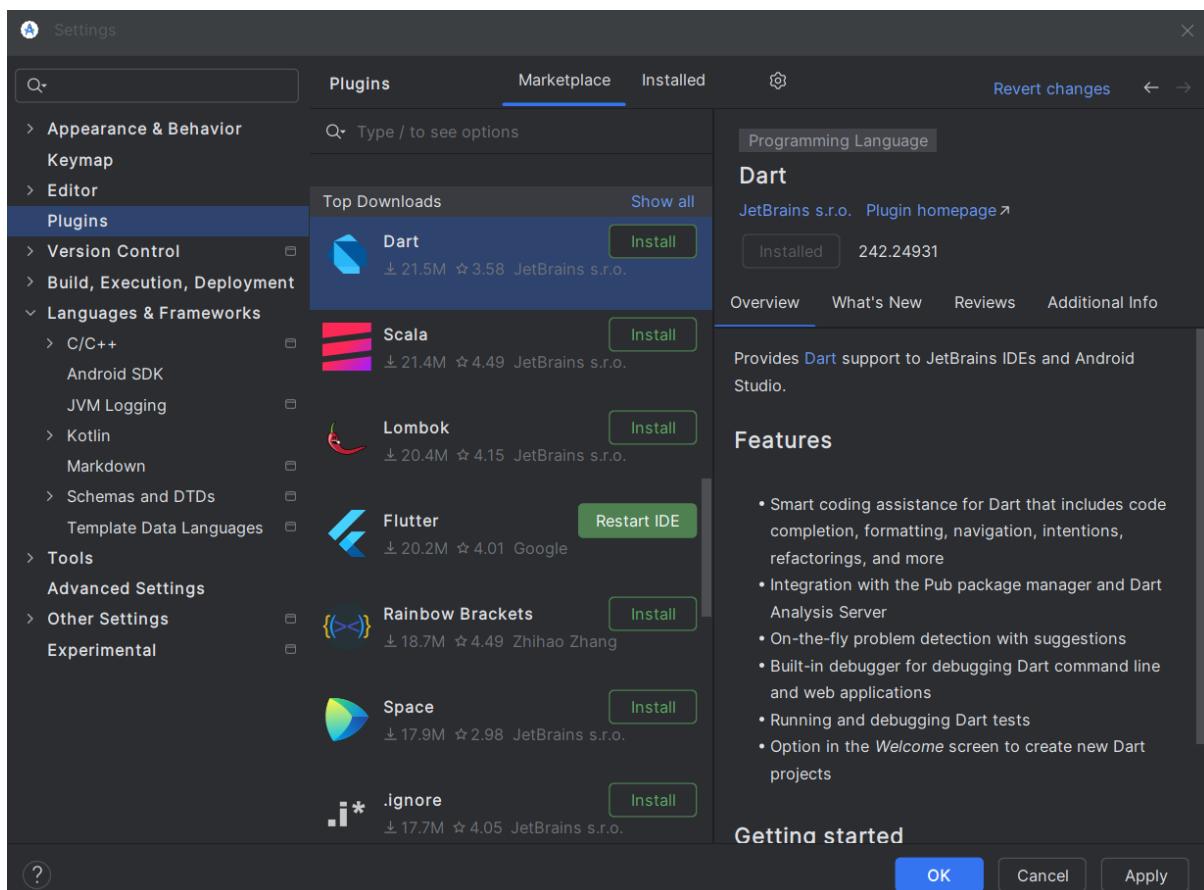
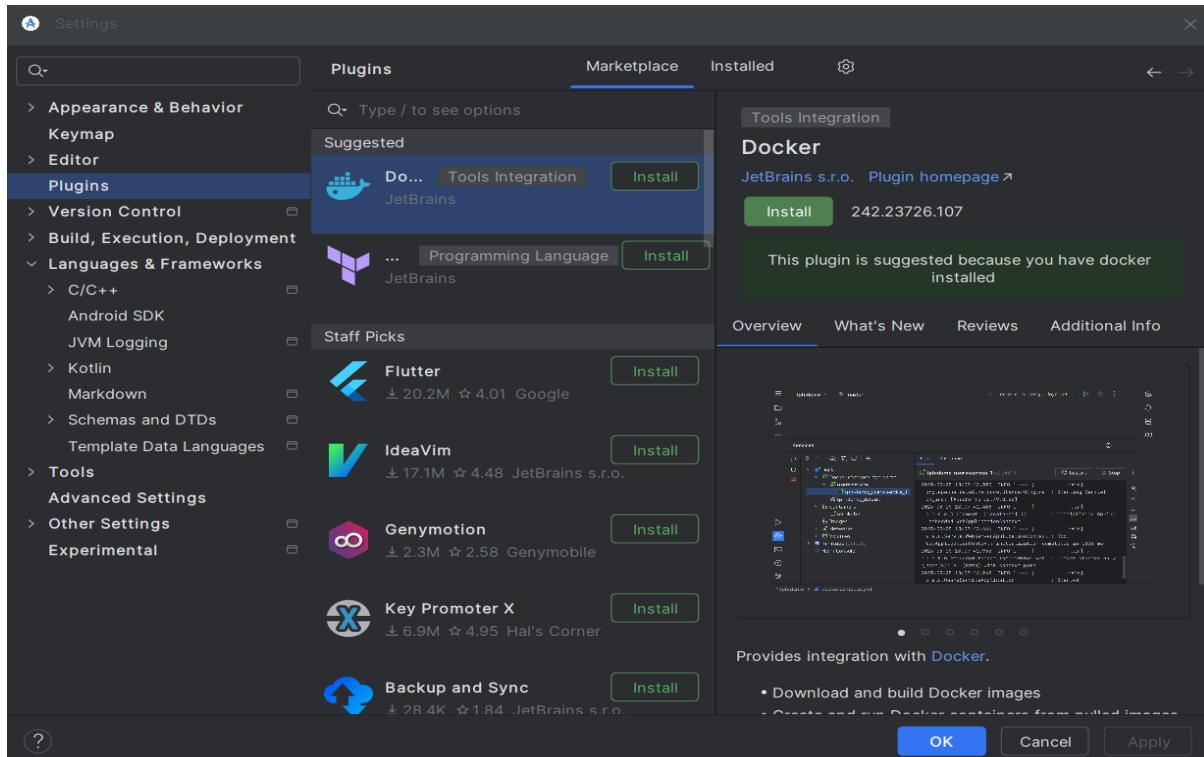
! Doctor found issues in 1 category.

C:\Users\manor>
```

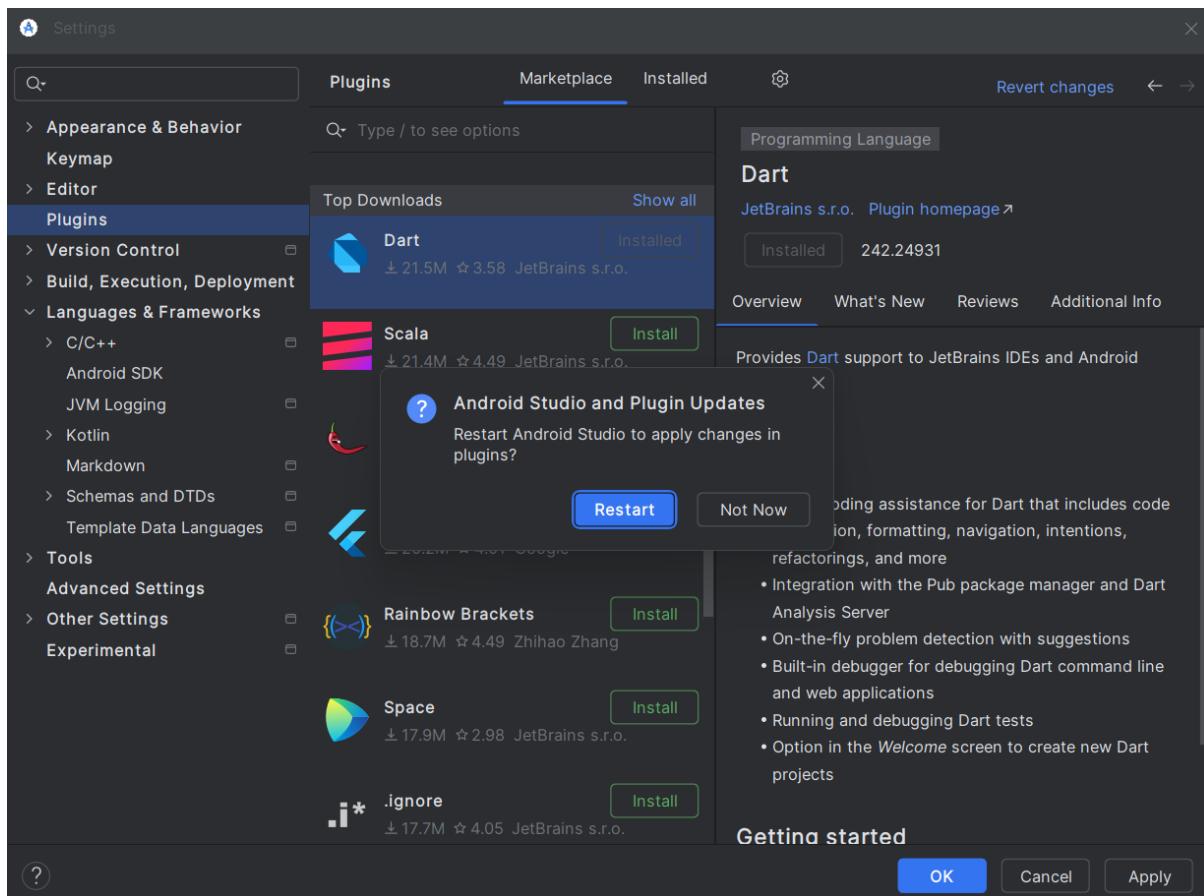
Step 10: - Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application



Step 11: - Now, install Flutter and Dart plugin for building Flutter application in Android Studio. These plugins provide a template to create a Flutter application, give an option to run and debug Flutter application in the Android Studio itself. Open the Android Studio and then go to File->Settings->Plugins. Now, search the Flutter plugin. If found, select Flutter plugin and click install.



Step 12: - Restart the Android Studio



Step 13: - Go to File > New Project > Create Flutter Project, then select the project name and location, and click Next to proceed.

 New Project X

Project name:

Project location: ...

Description:

Project type:

Organization:

Android language: Java Kotlin

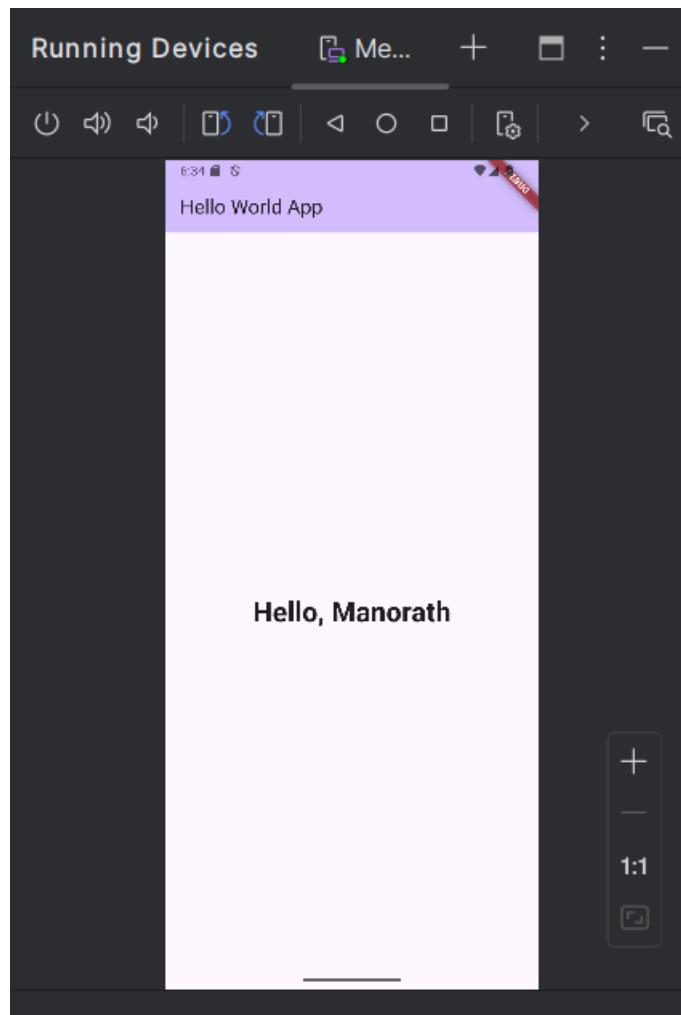
Platforms: Android iOS Linux MacOS Web Windows

When created, the new project will run on the selected platforms (others can be added later).

Create project offline

> More Settings

 Previous Cancel



Experiment-2

Name: Manorath Ital

Class : D15A

Roll no: 19

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.

Home Page

This home page is designed using Flutter and consists of various widgets to create an

engaging UI for a movie app called **FilmyFun**.

Widgets Used:

1. **Scaffold Widget** – Provides the basic structure of the app.
2. **Container Widget** – Used for layout styling and decorations.
3. **Row & Column Widgets** – Organize UI elements horizontally and vertically.
4. **Text Widget** – Displays textual content.
5. **Image.asset Widget** – Loads images from local assets.
6. **CarouselSlider Widget** – Implements an image slider for featured movie posters.
7. **ListView Widget** – Displays a scrollable list of trending movies.
8. **Stack Widget** – Used for overlaying text and effects on movie images.

UI Features:

1. Header Section:

- Displays a welcome message: "*Hello, Manorath*".
- Includes a wave emoji icon and a user profile picture.

2. App Title:

- The app name is displayed in a stylish way:
 - "Filmy" (White text)
 - "Fun" (Golden yellow text)

3. Carousel Image Slider:

- Showcases featured movie posters in a **carousel slider**.

- Users can swipe through movie images.

4. Trending Movies Section:

- Displays popular movies in a horizontal scrollable **ListView**.
- Each movie has:
 - A movie poster.
 - A title (e.g., *Infinity Wars*).
 - A genre (e.g., *Action, Adventure*).
 - A semi-transparent overlay for better readability.

Color Scheme:

- **Background:** Black for a cinematic feel.
- **Text Colors:** White and golden yellow for contrast.
- **Overlay Effects:** Black with transparency for better readability.

home.dart

Code:

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

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

  @override
  State<home> createState() =>
      _homeState();
}

class _homeState extends State<home> {
```

```
final List<String>imageUrls = [
    "images/infinity.jpg",
    "images/salman.jpg",
    "images/shahrukhmovies.png",
];

@Override
Widget build(BuildContext context) {
    return Scaffold(
        backgroundColor: Colors.black,
        body: Container(
            margin: EdgeInsets.only(top: 40.0, left: 20.0),
            child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                    Row(children: [
                        Image.asset("images/wave.png", height: 40, width: 40, fit: BoxFit.cover,),
                        SizedBox(width: 10.0,),
                        Text("Hello, Manorath",
                            style: TextStyle(
                                color: Colors.white,
                                fontSize: 22.0,
                                fontWeight: FontWeight.bold),
                    ),
                    ],
                    child: ClipRRect(
                        borderRadius: BorderRadius.circular(60),
                        child: Image.asset(
                            "images/Screenshot (55).png",
                            height: 60,
                            width: 60,
                            fit: BoxFit.cover,
                        ),
                    ),
                ],
            ),
        ),
    );
}
```

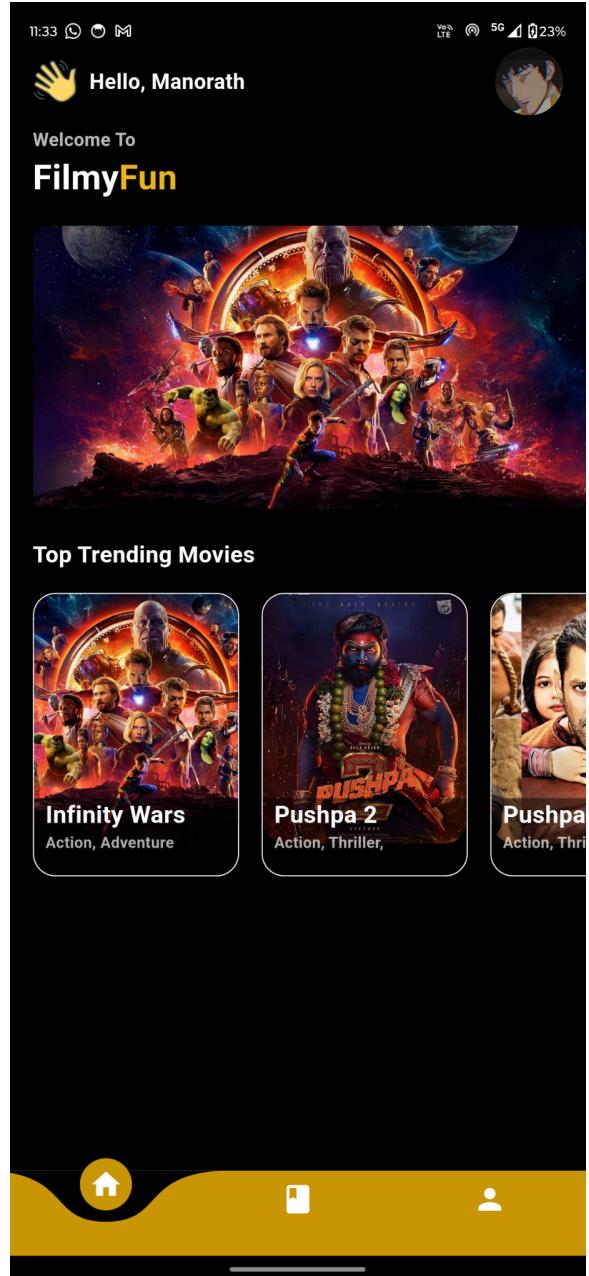
```
fontSize: 36.0,  
fontWeight: FontWeight.bold),),  
Text("Fun",  
style: TextStyle(  
color: Color(0xffedb41d),  
fontSize: 36.0,  
fontWeight: FontWeight.bold),)  
],  
,  
SizedBox(height: 20.0,),  
Center(child:  
CarouselSlider(  
items: imageUrl.map((url) {  
return Builder(builder: (context) {  
return Container(  
width: MediaQuery.of(context).size.width, //  
Fixed incorrect comma  
child: ClipRRect(  
borderRadius: BorderRadius.circular(10),  
child: Image.asset(url, fit: BoxFit.cover),  
),  
);  
});  
});  
}).toList(), options: CarouselOptions(  
height: 250,  
autoPlay: false,  
enlargeCenterPage: true,  
aspectRatio: 16/9,  
viewportFraction: 1.0  
,// Fixed missing .toList()  
)  
,  
SizedBox(height: 25.0,),  
Text("Top Trending Movies",  
style: TextStyle(  
color: Colors.white,  
fontSize: 24.0,  
fontWeight: FontWeight.bold),),  
SizedBox(height: 20.0,),  
Container(  
height: 250,  
child: ListView(  
scrollDirection: Axis.horizontal,  
children: [  
Container(  
decoration: BoxDecoration(border:  
Border.all(color:  
Colors.white),borderRadius:  
BorderRadius.circular(20)),  
child: Stack(  
children: [
```

```
ClipRRect(  
    borderRadius: BorderRadius.circular(20),  
    child: Image.asset("images/infinity.jpg",  
    height: 220,  
    width: 180,  
    fit: BoxFit.cover,  
,  
,  
Container(  
    padding: EdgeInsets.only(left: 10.0),  
    margin: EdgeInsets.only(top: 180),  
    height: 220,  
    width: 180,  
    decoration:  
        BoxDecoration(color:  
            Colors.black45,  
        borderRadius:  
            BorderRadius.only(  
                bottomRight: Radius.circular(20),  
                bottomLeft: Radius.circular(20))),  
    child: Column(  
        crossAxisAlignment:  
            CrossAxisAlignment.start,  
        children: [  
            Text("Infinity Wars", style: TextStyle(color:  
                Colors.white, fontSize: 25.0, fontWeight:  
                    FontWeight.bold),),  
            Text("Action, Adventure", style:  
                TextStyle(color: Color.fromARGB(173, 255,  
                    255, 255), fontSize: 16.0, fontWeight:  
                        FontWeight.bold),),  
        ],  
    ),  
),  
],  
),  
SizedBox(width: 20.0),  
Container(  
    decoration: BoxDecoration(border:  
        Border.all(color:  
            Colors.white),borderRadius:  
            BorderRadius.circular(20)),  
    child: Stack(  
        children: [  
            ClipRRect(  
                borderRadius: BorderRadius.circular(20),  
                child: Image.asset("images/pushpa.jpg",  
                    height: 220,  
                    width: 180,  
                    fit: BoxFit.cover,  
)  
        ],  
    ),  
),
```

```
),  
Container(  
padding: EdgeInsets.only(left: 10.0),  
margin: EdgeInsets.only(top: 180),  
height: 220,  
width: 180,  
decoration:  
BoxDecoration(color:  
Colors.black45,  
borderRadius:  
BorderRadius.only(  
bottomRight: Radius.circular(20),  
bottomLeft: Radius.circular(20))),  
child: Column(  
crossAxisAlignment:  
CrossAxisAlignment.start,  
children: [  
Text("Pushpa 2", style: TextStyle(color:  
Colors.white, fontSize: 25.0, fontWeight:  
FontWeight.bold),),  
Text("Action, Thriller,", style: TextStyle(color:  
Color.fromARGB(173, 255, 255, 255),  
fontSize: 16.0, fontWeight:  
FontWeight.bold),),  
],  
),  
),  
SizedBox(width: 20.0),  
Container(  

```

```
Colors.black45,  
borderRadius:  
BorderRadius.only(  
bottomRight: Radius.circular(20),  
bottomLeft: Radius.circular(20))),  
child: Column(  
crossAxisAlignment:  
CrossAxisAlignment.start,  
children: [  
Text("Pushpa 2", style: TextStyle(color:  
Colors.white, fontSize: 25.0, fontWeight:  
FontWeight.bold),),  
Text("Action, Thriller,", style: TextStyle(color:  
Color.fromARGB(173, 255, 255, 255),  
fontSize: 16.0, fontWeight:  
FontWeight.bold),),  
,  
,  
,  
,  
,  
)  
,  
,  
])  
));
```



EXPERIMENT NO: - 03

Manorath Ital

D15A/19

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

Theory: -

Incorporating Icons, Images, and Custom Fonts into FilmyFun App

FilmyFun is a user-friendly movie ticket booking app that allows users to easily select showtimes and reserve tickets. A crucial aspect of creating an engaging and visually appealing app is incorporating various visual elements, such as icons, images, and custom fonts. These elements enhance the overall user experience and help the app stand out.

Like most modern mobile applications, FilmyFun uses resources such as assets and code to provide a dynamic user experience. Assets, such as images, icons, and custom fonts, are integral in delivering a rich and intuitive interface. These resources are bundled with the app and are made available during runtime.

Visual Elements and Their Role in the App

Visual elements—icons, images, and fonts—play an essential role in app development. These elements are used to improve usability, convey information quickly, and enhance branding.

1. **Enhanced User Experience:** Icons and images make the app visually attractive and easier to navigate.
2. **Information Conveyance:** Icons help reduce text and convey actions intuitively.
3. **Branding:** Custom icons and images align with FilmyFun's branding, making it memorable and recognizable.

Incorporating Icons in FilmyFun

Icons serve as intuitive visual shortcuts, providing a quick way for users to identify actions and navigate the app. Flutter provides a wide variety of built-in icons through the Icons class, and custom icons can be added via third-party packages like flutter_launcher_icons or font_awesome_flutter.

Example of adding an icon:

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

Adding Images in FilmyFun

Images are crucial for enhancing the visual experience in FilmyFun. Flutter allows for images to be added from multiple sources:

1. Assets (Stored Locally)

- To add an image stored locally in the project, place the image in the assets/images/ folder.
- Declare the image path in the pubspec.yaml file.

flutter:

assets:

- assets/images/movie_poster.png

- Display the image in the app using the Image.asset method:

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

Incorporating Custom Fonts in FilmyFun

While Flutter's default font is **Roboto**, you may want to add custom fonts to align with your app's branding and overall design aesthetic. Adding custom fonts is simple:

1. **Download the font** and place it in the assets/fonts/ folder of the project.
2. **Declare the font** in the pubspec.yaml file.

flutter:

fonts:

- family: CustomFont

fonts:

- asset: assets/fonts/CustomFont-Regular.ttf

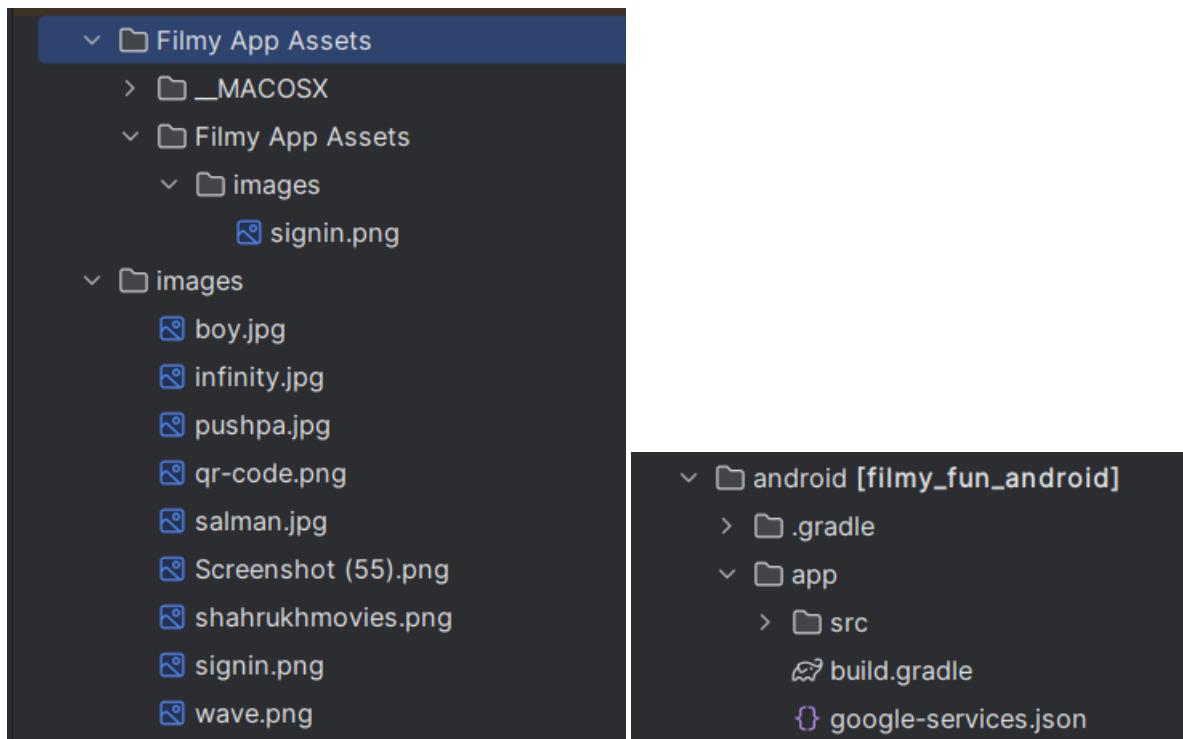
3. **Use the custom font** in your app:

```
Text(
```

```
'Welcome to FilmyFun!',
```

```
style: TextStyle(fontFamily: 'CustomFont', fontSize: 24),
```

```
);
```



Pubspec.yaml

```

name: filmy_fun
description: "FilmyFun-Movie Ticket Booking App"
# The following line prevents the package from being accidentally published to
# pub.dev using `flutter pub publish`. This is preferred for private packages.
publish_to: 'none' # Remove this line if you wish to publish to pub.dev

# The following defines the version and build number for your application.
# A version number is three numbers separated by dots, like 1.2.43
# followed by an optional build number separated by a +.
# Both the version and the builder number may be overridden in flutter
# build by specifying --build-name and --build-number, respectively.
# In Android, build-name is used as versionName while build-number used as versionCode.
# Read more about Android versioning at https://developer.android.com/studio/publish/versioning
# In iOS, build-name is used as CFBundleShortVersionString while build-number is used as
CFBundleVersion.
# Read more about iOS versioning at
#
https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CoreFoundationKeys.html
# In Windows, build-name is used as the major, minor, and patch parts
# of the product and file versions while build-number is used as the build suffix.
version: 1.0.0+1

environment:
  sdk: ^3.6.2

# Dependencies specify other packages that your package needs in order to work.

```

```

# To automatically upgrade your package dependencies to the latest versions
# consider running `flutter pub upgrade --major-versions`. Alternatively,
# dependencies can be manually updated by changing the version numbers below to
# the latest version available on pub.dev. To see which dependencies have newer
# versions available, run `flutter pub outdated`.

dependencies:
  flutter:
    sdk: flutter
  carousel_slider: ^5.0.0
  curved_navigation_bar: ^1.0.3 # Use the latest version
  intl: ^0.18.0
  firebase_auth: ^5.4.2
  cloud_firestore: ^5.6.3
  random_string: ^2.3.1
  firebase_core: ^3.11.0
  shared_preferences:
    flutter_stripe:
    http:

# The following adds the Cupertino Icons font to your application.
# Use with the CupertinoIcons class for iOS style icons.
cupertino_icons: ^1.0.8

dev_dependencies:
  flutter_test:
    sdk: flutter

# The "flutter_lints" package below contains a set of recommended lints to
# encourage good coding practices. The lint set provided by the package is
# activated in the `analysis_options.yaml` file located at the root of your
# package. See that file for information about deactivating specific lint
# rules and activating additional ones.
flutter_lints: ^5.0.0

# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspec

# The following section is specific to Flutter packages.
flutter:

  # The following line ensures that the Material Icons font is
  # included with your application, so that you can use the icons in
  # the material Icons class.
  uses-material-design: true

  # To add assets to your application, add an assets section, like this:
  assets:

```

```

- images/
# - images/a_dot_ham.jpeg

# An image asset can refer to one or more resolution-specific "variants", see
# https://flutter.dev/to/resolution-aware-images

# For details regarding adding assets from package dependencies, see
# https://flutter.dev/to/asset-from-package

# To add custom fonts to your application, add a fonts section here,
# in this "flutter" section. Each entry in this list should have a
# "family" key with the font family name, and a "fonts" key with a
# list giving the asset and other descriptors for the font. For
# example:
# fonts:
#   - family: Schyler
#     fonts:
#       - asset: fonts/Schyler-Regular.ttf
#       - asset: fonts/Schyler-Italic.ttf
#         style: italic
#   - family: Trajan Pro
#     fonts:
#       - asset: fonts/TrajanPro.ttf
#       - asset: fonts/TrajanPro_Bold.ttf
#         weight: 700
#
# For details regarding fonts from package dependencies,
# see https://flutter.dev/to/font-from-package

```

Home.dart

```

import 'package:carousel_slider/carousel_slider.dart';
import 'package:filmy_fun/pages/detail_page.dart';
import 'package:flutter/material.dart';

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

  @override
  State<home> createState() => _homeState();
}

class _homeState extends State<home> {
  final List<String> imageUrls = [
    "images/infinity.jpg",
    "images/salman.jpg",
    "images/shahrukhmovies.png",
  ];
  @override

```

```
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black,
    body: Container(
      margin: EdgeInsets.only(top: 40.0, left: 20.0),
      child:
        Column(crossAxisAlignment: CrossAxisAlignment.start, children: [
          Row(
            children: [
              Image.asset(
                "images/wave.png",
                height: 40,
                width: 40,
                fit: BoxFit.cover,
              ),
              SizedBox(
                width: 10.0,
              ),
              Text(
                "Hello, Manorath",
                style: TextStyle(
                  color: Colors.white,
                  fontSize: 22.0,
                  fontWeight: FontWeight.bold),
              ),
              Spacer(),
              Padding(
                padding: const EdgeInsets.only(right: 20.0),
                child: ClipRRect(
                  borderRadius: BorderRadius.circular(60),
                  child: Image.asset(
                    "images/Screenshot (55).png",
                    height: 60,
                    width: 60,
                    fit: BoxFit.cover,
                  ),
                ),
              ),
            ],
          ),
          SizedBox(height: 10.0),
          Text(
            "Welcome To",
            style: TextStyle(
              color: Color.fromARGB(186, 255, 255, 255),
              fontSize: 19.0,
              fontWeight: FontWeight.bold),
          ),
        ]),
    ),
  );
}
```

```

Row(
  children: [
    Text(
      "Filmy",
      style: TextStyle(
        color: Colors.white,
        fontSize: 36.0,
        fontWeight: FontWeight.bold),
    ),
    Text(
      "Fun",
      style: TextStyle(
        color: Color(0xffedb41d),
        fontSize: 36.0,
        fontWeight: FontWeight.bold),
    )
  ],
),
SizedBox(
  height: 20.0,
),
Center(
  child: CarouselSlider(
    items: imageUrl.map((url) {
      return Builder(builder: (context) {
        return Container(
          width: MediaQuery.of(context)
            .size
            .width, // Fixed incorrect comma
          child: ClipRRect(
            borderRadius: BorderRadius.circular(10),
            child: Image.asset(url, fit: BoxFit.cover),
          ),
        );
      });
    }).toList(),
    options: CarouselOptions(
      height: 250,
      autoPlay: false,
      enlargeCenterPage: true,
      aspectRatio: 16 / 9,
      viewportFraction: 1.0), // Fixed missing .toList()
  )),
SizedBox(
  height: 25.0,
),
Text(
  "Top Trending Movies",
)

```

```

style: TextStyle(
  color: Colors.white,
  fontSize: 24.0,
  fontWeight: FontWeight.bold),
),
SizedBox(
  height: 20.0,
),
Container(
  height: 250,
  child: ListView(
    scrollDirection: Axis.horizontal,
    children: [
      GestureDetector(
        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) => DetailPage(
                image: "images/infinity.jpg",
                name: "Infinity Wars",
                shortdetail: "Action, Adventure",
                moviedetail:
                  "Avengers: Infinity War is a 2018 American superhero film based on the
                  Marvel Comics superhero team the Avengers. Produced by Marvel Studios and distributed by Walt
                  Disney Studios Motion Pictures, it is the sequel to The Avengers (2012) and Avengers: Age of Ultron
                  (2015), and the 19th film in the Marvel Cinematic Universe (MCU)",
                price: '50',
              )));
        },
      ),
      child: Container(
        decoration: BoxDecoration(
          border: Border.all(color: Colors.white),
          borderRadius: BorderRadius.circular(20)),
        child: Stack(
          children: [
            ClipRRect(
              borderRadius: BorderRadius.circular(20),
              child: Image.asset(
                "images/infinity.jpg",
                height: 220,
                width: 180,
                fit: BoxFit.cover,
              ),
            ),
            Container(
              padding: EdgeInsets.only(left: 10.0),
              margin: EdgeInsets.only(top: 180),
            )
          ],
        )
      )
    ],
  )
)

```


12:00

87%

Hello, Manorath

Welcome To
FilmyFun

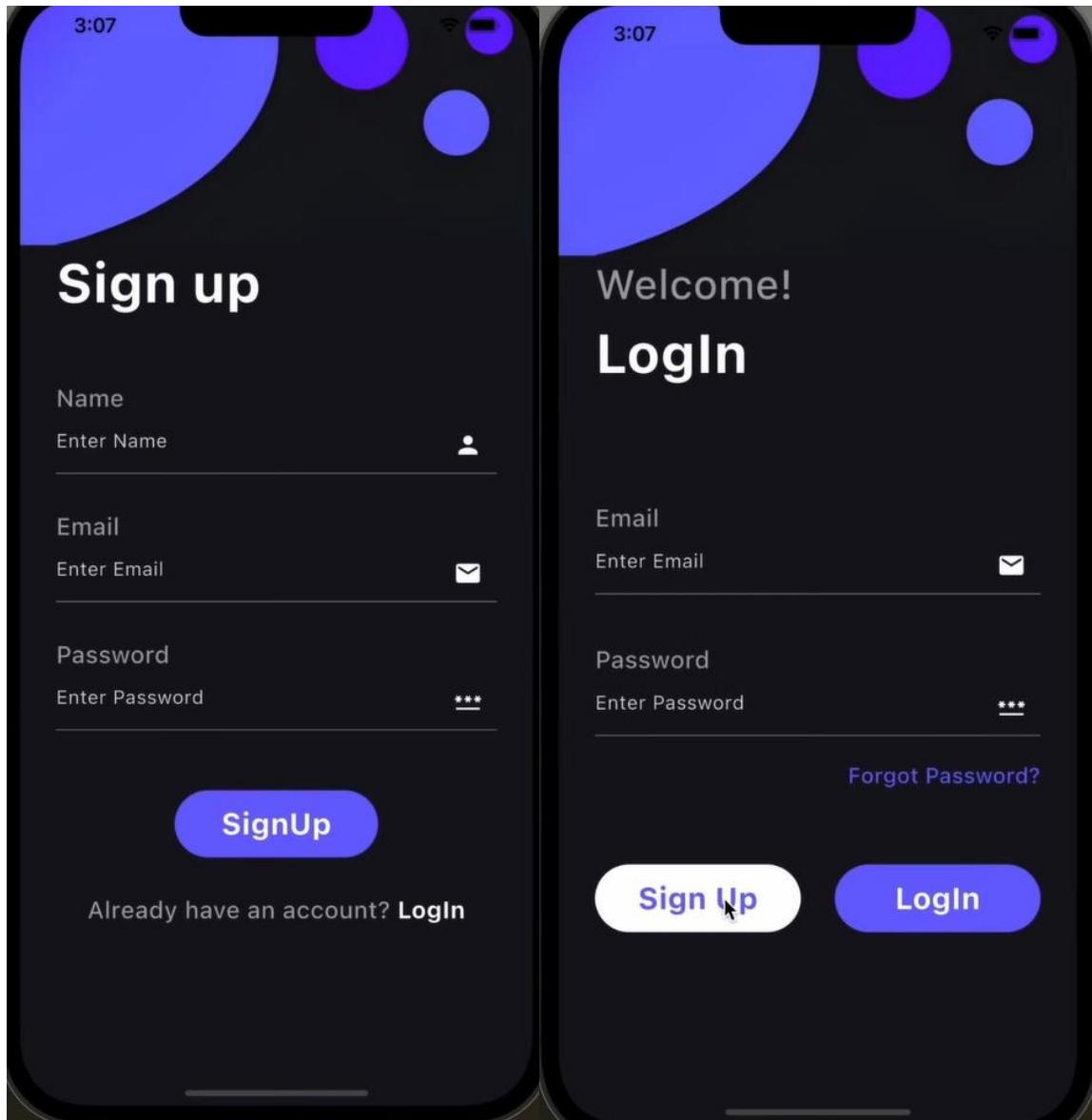
A large movie poster for "Avengers: Infinity War" featuring Thanos and many other Marvel characters.

Top Trending Movies



ICONS:

```
buildInputField("Name", "Enter Name", Icons.person, namecontroller),
buildInputField("Email", "Enter Email", Icons.email, mailcontroller),
buildInputField("Password", "Enter Password", Icons.lock, passwordcontroller, obscureText: true),
const SizedBox(height: 30.0),
```

**Conclusion**

By incorporating icons, images, and custom fonts into the FilmyFun app, you can create an intuitive, visually appealing, and engaging user experience. These visual elements contribute to the overall aesthetic, enhance usability, and ensure your app stands out in a competitive market. Whether displaying movie posters, providing easy navigation with icons, or using custom fonts for unique branding, these resources will elevate the look and feel of your app.

Experiment 4

Name:Manorath Ital
Batch:D15A-19

Aim:To create interactive form

SignUp page

Code:

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

import 'package:filmy_fun/pages/login.dart';

class Signup extends StatefulWidget {

const Signup({super.key});

@Override

State<Signup> createState() => _SignupState();

}

class _SignupState extends State<Signup> {

@Override

Widget build(BuildContext context) {

return Scaffold(


backgroundColor: Colors.black,


body: Padding(


padding: const EdgeInsets.symmetric(horizontal: 20.0), // Adds padding to align left


child: Column(


crossAxisAlignment: CrossAxisAlignment.start, // Aligns content to left


children: [


Image.asset("images/signin.png"),


const SizedBox(height: 20.0), // Adds spacing
}
```

```
const Text(  
    "Welcome!",  
    style: TextStyle(  
        color: Color.fromARGB(157, 255, 255, 255),  
        fontSize: 34.0,  
        fontWeight: FontWeight.w500,  
    ),  
,  
const Text(  
    "SignUp",  
    style: TextStyle(  
        color: Colors.white,  
        fontSize: 45.0,  
        fontWeight: FontWeight.bold,  
    ),  
,  
const SizedBox(height: 30.0), // Adjust spacing  
const Text(  
    "Name",  
    style: TextStyle(  
        color: Colors.white,  
        fontSize: 20.0,  
        fontWeight: FontWeight.w500,  
    ),
```

```
 ),  
 TextField(  
 decoration: InputDecoration(  
 hintText: "Enter Name",  
 hintStyle: TextStyle(  
 color: Colors.white,  
 ),  
 suffixIcon: Icon(Icons.password,  
 color: Colors.white,),  
 ),  
 ),  
 const SizedBox(height: 50.0), // Adjust spacing  
 const Text(  
 "Email",  
 style: TextStyle(  
 color: Colors.white,  
 fontSize: 20.0,  
 fontWeight: FontWeight.w500,  
 ),  
 ),  
 TextField(  
 decoration: InputDecoration(  
 hintText: "Enter Email",  
 hintStyle: TextStyle(  

```

```
color: Colors.white,  
,  
suffixIcon: Icon(Icons.email, color: Colors.white,),  
,  
,  
const SizedBox(height: 50.0), // Adjust spacing  
const Text(  
"Password",  
style: TextStyle(  
color: Colors.white,  
fontSize: 20.0,  
fontWeight: FontWeight.w500,  
,  
,  
TextField(  
decoration: InputDecoration(  
hintText: "Enter Password",  
hintStyle: TextStyle(  
color: Colors.white,  
,  
suffixIcon: Icon(Icons.password,  
color: Colors.white,),  
,  
,
```

```
const SizedBox(height: 50.0), // Adjust spacing

const Text(
  "Confirm Password",
  style: TextStyle(
    color: Colors.white,
    fontSize: 20.0,
    fontWeight: FontWeight.w500,
  ),
),

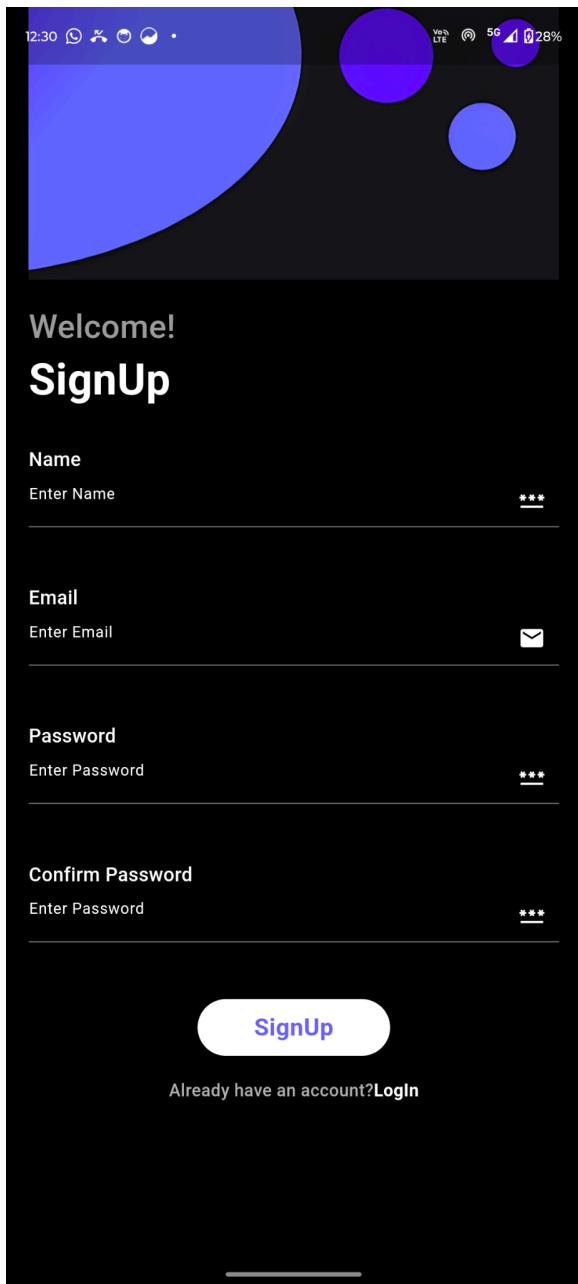
TextField(
  decoration: InputDecoration(
    hintText: "Enter Password",
    hintStyle: TextStyle(
      color: Colors.white,
    ),
    suffixIcon: Icon(Icons.password,
    color: Colors.white,),
  ),
),

SizedBox(height: 50.0,),

Center(
  child: Column(
    children: [
      Container(
```

```
width: 170,  
padding: EdgeInsets.all(10),  
decoration: BoxDecoration(  
color: Colors.white,  
borderRadius: BorderRadius.circular(30),  
,  
child: Text(  
"SignUp",  
style: TextStyle(  
color: Color(0xff6b63ff),  
fontSize: 25.0,  
fontWeight: FontWeight.bold,  
,  
textAlign: TextAlign.center,  
,  
,  
SizedBox(height: 20.0), // Space between button and text  
Row(  
mainAxisAlignment: MainAxisAlignment.center,  
children: [  
Text(  
"Already have an account?",  
style: TextStyle(  
color: Color.fromARGB(175, 255, 255, 255),
```

```
fontSize: 18.0,  
fontWeight: FontWeight.w500,  
,  
,  
GestureDetector(  
onTap:() {Navigator.push(context, MaterialPageRoute(builder: (context)=> Login()));},  
child: Text(  
"Login",  
style: TextStyle(  
color: Colors.white,  
fontSize: 18.0,  
fontWeight: FontWeight.bold,  
,),],),],),],),);});}
```



Login page

Code:

```
import 'package:filmy_fun/pages/bottomnav.dart';

import 'package:flutter/material.dart';

import 'package:filmy_fun/pages/signup.dart';

class Login extends StatefulWidget {

const Login({super.key});

@Override

State<Login> createState() => _LoginState();

}

class _LoginState extends State<Login> {

@Override

Widget build(BuildContext context) {

return Scaffold(

backgroundColor: Colors.black,

body: Padding(


padding: const EdgeInsets.symmetric(horizontal: 20.0), // Adds padding to align left

child: Column(


crossAxisAlignment: CrossAxisAlignment.start, // Aligns content to left

children: [


Image.asset("images/signin.png"),


const SizedBox(height: 20.0), // Adds spacing

const Text(


"Welcome!",

style: TextStyle(
```

```
color: Color.fromARGB(157, 255, 255, 255),  
fontSize: 34.0,  
fontWeight: FontWeight.w500,  
,  
,  
const Text(  
"Login",  
style: TextStyle(  
color: Colors.white,  
fontSize: 45.0,  
fontWeight: FontWeight.bold,  
,  
,  
const SizedBox(height: 50.0), // Adjust spacing  
const Text(  
"Email",  
style: TextStyle(  
color: Colors.white,  
fontSize: 20.0,  
fontWeight: FontWeight.w500,  
,  
,  
TextField(  
decoration: InputDecoration(  

```

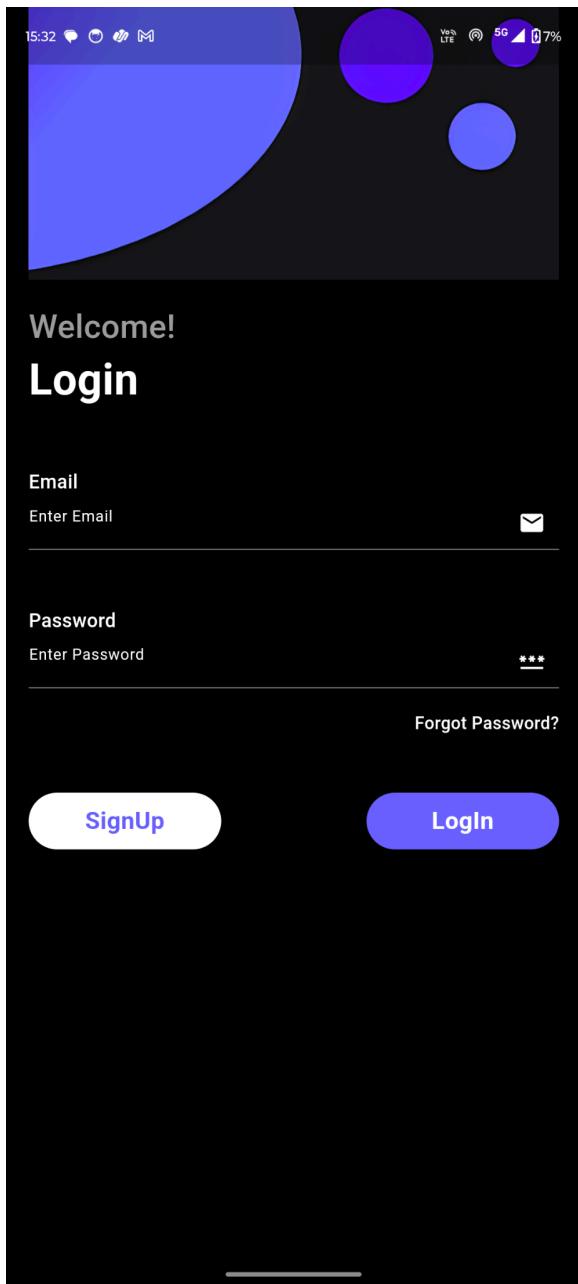
```
hintText: "Enter Email",  
hintStyle: TextStyle(  
color: Colors.white,  
,  
suffixIcon: Icon(Icons.email, color: Colors.white,),  
,  
,  
const SizedBox(height: 50.0), // Adjust spacing  
const Text(  
"Password",  
style: TextStyle(  
color: Colors.white,  
fontSize: 20.0,  
fontWeight: FontWeight.w500,  
,  
,  
TextField(  
decoration: InputDecoration(  
hintText: "Enter Password",  
hintStyle: TextStyle(  
color: Colors.white,  
,  
suffixIcon: Icon(Icons.password,  
color: Colors.white,),
```

```
),
),
const SizedBox(height: 20.0),
Row(
mainAxisAlignment: MainAxisAlignment.end,
children: [
Text(
"Forgot Password?",
style: TextStyle(
color: Colors.white,
fontSize: 18.0,
fontWeight: FontWeight.w500,
),
)
],
SizedBox(height: 50.0,),
Row(
mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [
Row(
mainAxisAlignment: MainAxisAlignment.center,
children: [
GestureDetector(
onTap:() {Navigator.push(context, MaterialPageRoute(builder: (context)=> Signup()));},

```

```
child: Container(  
    width: 170,  
    padding: EdgeInsets.all(10),  
    decoration: BoxDecoration(color: Colors.white,  
        borderRadius: BorderRadius.circular(30)),  
    child: Text("SignUp",style: TextStyle(color: Color(0xff6b63ff),  
        fontSize: 25.0,  
        fontWeight: FontWeight.bold),  
    textAlign: TextAlign.center,  
,  
,  
,  
,  
,  
],  
,  
Row(  
    mainAxisAlignment: MainAxisAlignment.center,  
    children: [  
        GestureDetector(  
            onTap: () {Navigator.push(context, MaterialPageRoute(builder: (context)=> Bottomnav()));},  
            child: Container(  
                width: 170,  
                padding: EdgeInsets.all(10),  
                decoration: BoxDecoration(  
                    color: Color(0xff6b63ff),
```

```
borderRadius: BorderRadius.circular(30)),  
child: Text("LogIn",style: TextStyle(color: Colors.white,  
fontSize: 25.0,  
fontWeight: FontWeight.bold),  
textAlign: TextAlign.center,  
),  
,  
,  
],  
)  
],  
)  
],  
)  
,  
)  
);  
}  
}
```



EXPERIMENT NO: - 05

MANORATH ITAL

D15A/19

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

Theory: -

In Flutter, the screens and pages are known as routes, and these routes are just a widget. In Android, a route is similar to an Activity.

In any mobile app, navigating to different pages defines the workflow of the application, and the way to handle the navigation is known as routing. Flutter provides a basic routing class MaterialPageRoute and two methods Navigator.push() and Navigator.pop() that shows how to navigate between two routes. The following steps are required to start navigation in your application.

Gestures enable the app to respond to user interactions, making the application more dynamic and responsive.

➤ Navigation and Routing in Flutter

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

1. Using Navigator Widget

The Navigator widget manages a stack of routes, allowing for pushing and popping routes on the stack.

- Pushing a Route: To navigate to a new screen, use Navigator.push().
- Popping a Route: To go back to the previous screen, use Navigator.pop().

```
ElevatedButton(
    onPressed: () {
        Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => SecondScreen()),
        );
    });
);
```

2. Named Routes

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

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

Navigate to the route using `Navigator.pushNamed()`

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

Handling Gestures in Flutter

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

Tap Gestures

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

Long Press Gesture

For long press gestures, Flutter provides the `onLongPress` callback in `GestureDetector` or `InkWell`.

Swipe and Drag Gestures

Flutter also provides swipe and drag gesture handling. The `onHorizontalDragUpdate` and `onVerticalDragUpdate` callbacks are used for dragging gestures.

Code:-

main.dart

```
import 'package:filmy_fun/pages/detail_page.dart';
import 'package:filmy_fun/service/constant.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:filmy_fun/pages/login.dart';
import 'package:filmy_fun/pages/signup.dart';
import 'package:filmy_fun/pages/home.dart';
import 'package:filmy_fun/pages/bottomnav.dart';
import 'package:filmy_fun/pages/booking.dart';
import 'package:flutter_stripe/flutter_stripe.dart';

void main()async {
    WidgetsFlutterBinding.ensureInitialized();
    Stripe.publishableKey= publishedKey;
    await Firebase.initializeApp();
    runApp(const MyApp());
}

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

    // This widget is the root of your application.
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: 'FilmyFun',
            debugShowCheckedModeBanner: false,
            theme: ThemeData(
                // This is the theme of your application.
                //
                // TRY THIS: Try running your application with "flutter run". You'll see
                // the application has a purple toolbar. Then, without quitting the app,
                // try changing the seedColor in the colorScheme below to Colors.green
                // and then invoke "hot reload" (save your changes or press the "hot
                // reload" button in a Flutter-supported IDE, or press "r" if you used
                // the command line to start the app).
                //
                // Notice that the counter didn't reset back to zero; the application
                // state is not lost during the reload. To reset the state, use hot
                // restart instead.
                //

```

```

// This works for code too, not just values: Most code changes can be
// tested with just a hot reload.
colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
useMaterial3: true,
),
home: Signup(),
);
}
}

class MyHomePage extends StatefulWidget {
const MyHomePage({super.key, required this.title});

// This widget is the home page of your application. It is stateful, meaning
// that it has a State object (defined below) that contains fields that affect
// how it looks.

// This class is the configuration for the state. It holds the values (in this
// case the title) provided by the parent (in this case the App widget) and
// used by the build method of the State. Fields in a Widget subclass are
// always marked "final".

final String title;

@Override
State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
int _counter = 0;

void _incrementCounter() {
  setState(() {
    // This call to setState tells the Flutter framework that something has
    // changed in this State, which causes it to rerun the build method below
    // so that the display can reflect the updated values. If we changed
    // _counter without calling setState(), then the build method would not be
    // called again, and so nothing would appear to happen.
    _counter++;
  });
}

@Override
Widget build(BuildContext context) {
  // This method is rerun every time setState is called, for instance as done
  // by the _incrementCounter method above.
  //
  // The Flutter framework has been optimized to make rerunning build methods
}

```

```

// fast, so that you can just rebuild anything that needs updating rather
// than having to individually change instances of widgets.
return Scaffold(
  appBar: AppBar(
    // TRY THIS: Try changing the color here to a specific color (to
    // Colors.amber, perhaps?) and trigger a hot reload to see the AppBar
    // change color while the other colors stay the same.
    backgroundColor: Theme.of(context).colorScheme.inversePrimary,
    // Here we take the value from the MyHomePage object that was created by
    // the App.build method, and use it to set our appbar title.
    title: Text(widget.title),
  ),
  body: Center(
    // Center is a layout widget. It takes a single child and positions it
    // in the middle of the parent.
    child: Column(
      // Column is also a layout widget. It takes a list of children and
      // arranges them vertically. By default, it sizes itself to fit its
      // children horizontally, and tries to be as tall as its parent.
      //
      // Column has various properties to control how it sizes itself and
      // how it positions its children. Here we use mainAxisAlignment to
      // center the children vertically; the main axis here is the vertical
      // axis because Columns are vertical (the cross axis would be
      // horizontal).
      //
      // TRY THIS: Invoke "debug painting" (choose the "Toggle Debug Paint"
      // action in the IDE, or press "p" in the console), to see the
      // wireframe for each widget.
      mainAxisAlignment: MainAxisAlignment.center,
      children: <Widget>[
        const Text(
          'You have pushed the button this many times:',
        ),
        Text(
          '_counter',
          style: Theme.of(context).textTheme.headlineMedium,
        ),
      ],
    ),
  ),
  floatingActionButton: FloatingActionButton(
    onPressed: _incrementCounter,
    tooltip: 'Increment',
    child: const Icon(Icons.add),
  ), // This trailing comma makes auto-formatting nicer for build methods.
);

```

```
}
```

```
}
```

Signup.dart

```
import 'package:filmy_fun/service/database.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:filmy_fun/pages/login.dart';
import 'package:filmy_fun/service/shared_pref.dart';
import 'package:random_string/random_string.dart';
import 'bottomnav.dart';

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

  @override
  State<Signup> createState() => _SignupState();
}

class _SignupState extends State<Signup> {
  String email = "", password = "", name = "";
  TextEditingController namecontroller = TextEditingController();
  TextEditingController passwordcontroller = TextEditingController();
  TextEditingController mailcontroller = TextEditingController();

  registration() async {
    if (passwordcontroller.text.isNotEmpty &&
        namecontroller.text.isNotEmpty &&
        mailcontroller.text.isNotEmpty) {
      try {
        UserCredential userCredential = await FirebaseAuth.instance
            .createUserWithEmailAndPassword(
                email: mailcontroller.text, password: passwordcontroller.text);
        String id = randomAlphaNumeric(10);
        Map<String, dynamic> userInfoMap = {
          "Name": namecontroller.text,
          "Email": mailcontroller.text,
          "Id": id,
          "Image": ""
        };
        await SharedpreferenceHelper().saveUserDisplayname(namecontroller.text);
        await SharedpreferenceHelper().saveUserEmail(mailcontroller.text);
        await SharedpreferenceHelper().saveUserID(id);
        await SharedpreferenceHelper().saveUserImage("");
        await DatabaseMethods().addUserDetails(userInfoMap, id);
        ScaffoldMessenger.of(context).showSnackBar(SnackBar(
          backgroundColor: Colors.green,
          content: Text(

```



```

),
const Text(
  "SignUp",
  style: TextStyle(
    color: Colors.white,
    fontSize: 45.0,
    fontWeight: FontWeight.bold,
  ),
),
const SizedBox(height: 30.0),
buildInputField("Name", "Enter Name", Icons.person, namecontroller),
buildInputField("Email", "Enter Email", Icons.email, mailcontroller),
buildInputField("Password", "Enter Password", Icons.lock, passwordcontroller, obscureText:
true),
const SizedBox(height: 30.0),
Center(
  child: Column(
    children: [
      GestureDetector(
        onTap: registration,
        child: GestureDetector(
          onTap: () {
            if (namecontroller.text.isNotEmpty &&
                mailcontroller.text.isNotEmpty &&
                passwordcontroller.text.isNotEmpty) {
              setState(() {
                name = namecontroller.text;
                email = mailcontroller.text;
                password = passwordcontroller.text;
              });
              registration();
            }
          },
        ),
      ),
      child: Container(
        width: 170,
        padding: const EdgeInsets.all(12),
        decoration: BoxDecoration(
          color: Colors.white,
          borderRadius: BorderRadius.circular(30),
        ),
        child: const Text(
          "SignUp",
          style: TextStyle(
            color: Color(0xff6b63ff),
            fontSize: 25.0,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
      ),
    ],
  ),
);

```

```

        ),
        ),
        ),
        ),
        const SizedBox(height: 20.0),
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            const Text(
              "Already have an account?",
              style: TextStyle(
                color: Color.fromARGB(175, 255, 255, 255),
                fontSize: 18.0,
                fontWeight: FontWeight.w500,
              ),
            ),
            GestureDetector(
              onTap: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(
                    builder: (context) => const Login()));
              },
              child: const Text(
                " LogIn",
                style: TextStyle(
                  color: Colors.white,
                  fontSize: 18.0,
                  fontWeight: FontWeight.bold,
                ),
              ),
            ),
          ],
        ),
        const SizedBox(height: 30.0),
      ],
    ),
  ),
),
],
),
),
),
),
),
);
}
}

Widget buildInputField(String label, String hint, IconData icon,
  TextEditingController controller,
  {bool obscureText = false}) {

```

```

return Padding(
  padding: const EdgeInsets.only(bottom: 20.0),
  child: Column(
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      Text(
        label,
        style: const TextStyle(
          color: Colors.white,
          fontSize: 20.0,
          fontWeight: FontWeight.w500,
        ),
      ),
      TextField(
        controller: controller,
        obscureText: obscureText,
        cursorColor: Colors.white, // Cursor color set to white
        style: const TextStyle(color: Colors.white), // Text color set to white
        decoration: InputDecoration(
          hintText: hint,
          hintStyle: const TextStyle(color: Colors.white), // Hint text color set to white
          suffixIcon: Icon(icon, color: Colors.white),
          enabledBorder: const UnderlineInputBorder(
            borderSide: BorderSide(color: Colors.white), // Underline color
          ),
          focusedBorder: const UnderlineInputBorder(
            borderSide: BorderSide(color: Colors.white), // Underline when focused
          ),
        ),
      ),
    ],
  );
}
}

```

Login.dart

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:filmy_fun/pages/bottomnav.dart';
import 'package:filmy_fun/pages/home.dart';
import 'package:filmy_fun/service/database.dart';
import 'package:filmy_fun/service/shared_pref.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:filmy_fun/pages/signup.dart';

class Login extends StatefulWidget {

```

```

const Login({super.key});

@Override
State<Login> createState() => _LoginState();
}

class _LoginState extends State<Login> {
String email = "", password = "", myname = "", myid = "", myimage = "";
TextEditingController passwordcontroller = TextEditingController();
TextEditingController mailcontroller = TextEditingController();

userLogin() async {
try {
await FirebaseAuth.instance.signInWithEmailAndPassword(
email: email,
password: password,
);
QuerySnapshot querySnapshot =
await DatabaseMethods().getUserbyemail(email);
myname = "${querySnapshot.docs[0]["Name"]}";
myid = "${querySnapshot.docs[0]["Id"]}";
myimage = "${querySnapshot.docs[0]["Image"]}";

await SharedpreferenceHelper().saveUserImage(myimage);
await SharedpreferenceHelper().saveUserEmail(email);
await SharedpreferenceHelper().saveUserDislayName(myname);
await SharedpreferenceHelper().saveUserID(myid);
Navigator.push(context, MaterialPageRoute(builder: (context) => Bottomnav()));
} on FirebaseAuthException catch (e) {
String errorMessage = "";
if (e.code == 'user-not-found') {
ScaffoldMessenger.of(context).showSnackBar(SnackBar(
content: Text("No user found for that email."),
style: TextStyle(fontSize: 18.0, color: Colors.black),
));
}
} else if (e.code == 'wrong-password') {
ScaffoldMessenger.of(context).showSnackBar(SnackBar(
backgroundColor: Colors.white,
content: Text("Wrong password provided."
));
}
}
}

@Override
Widget build(BuildContext context) {
return Scaffold(
backgroundColor: Colors.black,

```

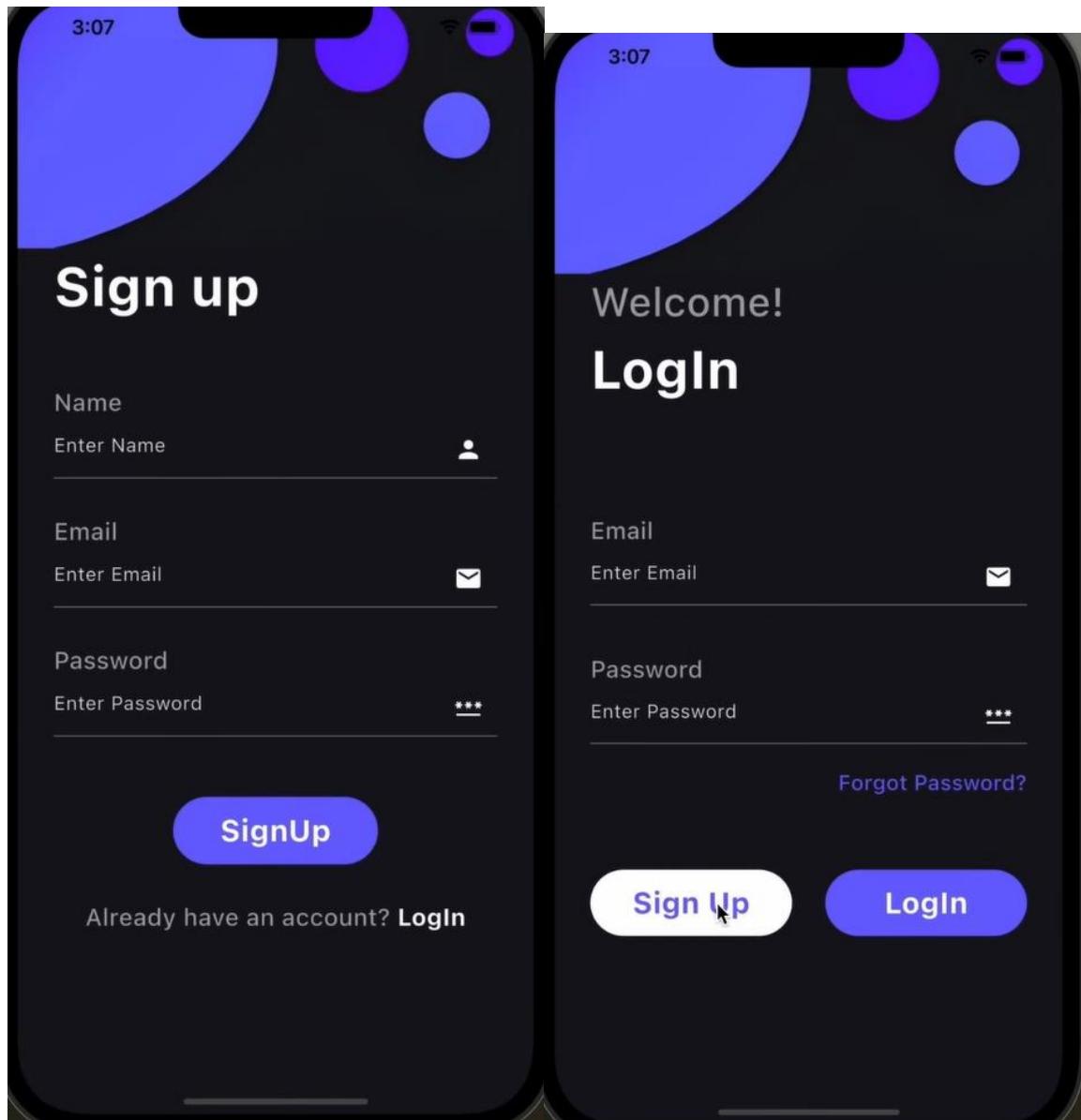
```
body: Padding(
  padding: const EdgeInsets.symmetric(horizontal: 20.0),
  child: Column(
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      Image.asset("images/signin.png"),
      const SizedBox(height: 20.0),
      const Text(
        "Welcome!",
        style: TextStyle(
          color: Color.fromARGB(157, 255, 255, 255),
          fontSize: 34.0,
          fontWeight: FontWeight.w500,
        ),
      ),
      GestureDetector(
        onTap: () {
          if (mailcontroller.text.isNotEmpty &&
              passwordcontroller.text.isNotEmpty) {
            setState(() {
              email = mailcontroller.text;
              password = passwordcontroller.text;
              userLogin();
            });
          }
        },
      ),
      const Text(
        "Login",
        style: TextStyle(
          color: Colors.white,
          fontSize: 45.0,
          fontWeight: FontWeight.bold,
        ),
      ),
    ],
  ),
  const SizedBox(height: 50.0),
  const Text(
    "Email",
    style: TextStyle(
      color: Colors.white,
      fontSize: 20.0,
      fontWeight: FontWeight.w500,
    ),
  ),
  TextField(
    controller: mailcontroller,
    cursorColor: Colors.black,
    style: const TextStyle(color: Colors.white),
```

```
decoration: const InputDecoration(
    hintText: "Enter Email",
    hintStyle: TextStyle(color: Colors.grey),
),
),
const SizedBox(height: 50.0),
const Text(
    "Password",
    style: TextStyle(
        color: Colors.white,
        fontSize: 20.0,
        fontWeight: FontWeight.w500,
),
),
TextField(
    controller: passwordcontroller,
    cursorColor: Colors.black,
    style: const TextStyle(color: Colors.white),
    obscureText: true,
    decoration: const InputDecoration(
        hintText: "Enter Password",
        hintStyle: TextStyle(color: Colors.grey),
),
),
const SizedBox(height: 20.0),
Row(
    mainAxisAlignment: MainAxisAlignment.end,
    children: [
        Text(
            "Forgot Password?",
            style: TextStyle(
                color: Colors.white,
                fontSize: 18.0,
                fontWeight: FontWeight.w500,
),
        ),
    ]
),
const SizedBox(height: 50.0),
Row(
    mainAxisAlignment: MainAxisAlignment.spaceBetween,
    children: [
        GestureDetector(
            onTap: () {
                Navigator.push(context,
                    MaterialPageRoute(builder: (context) => Signup()));
},
            child: Container(
```

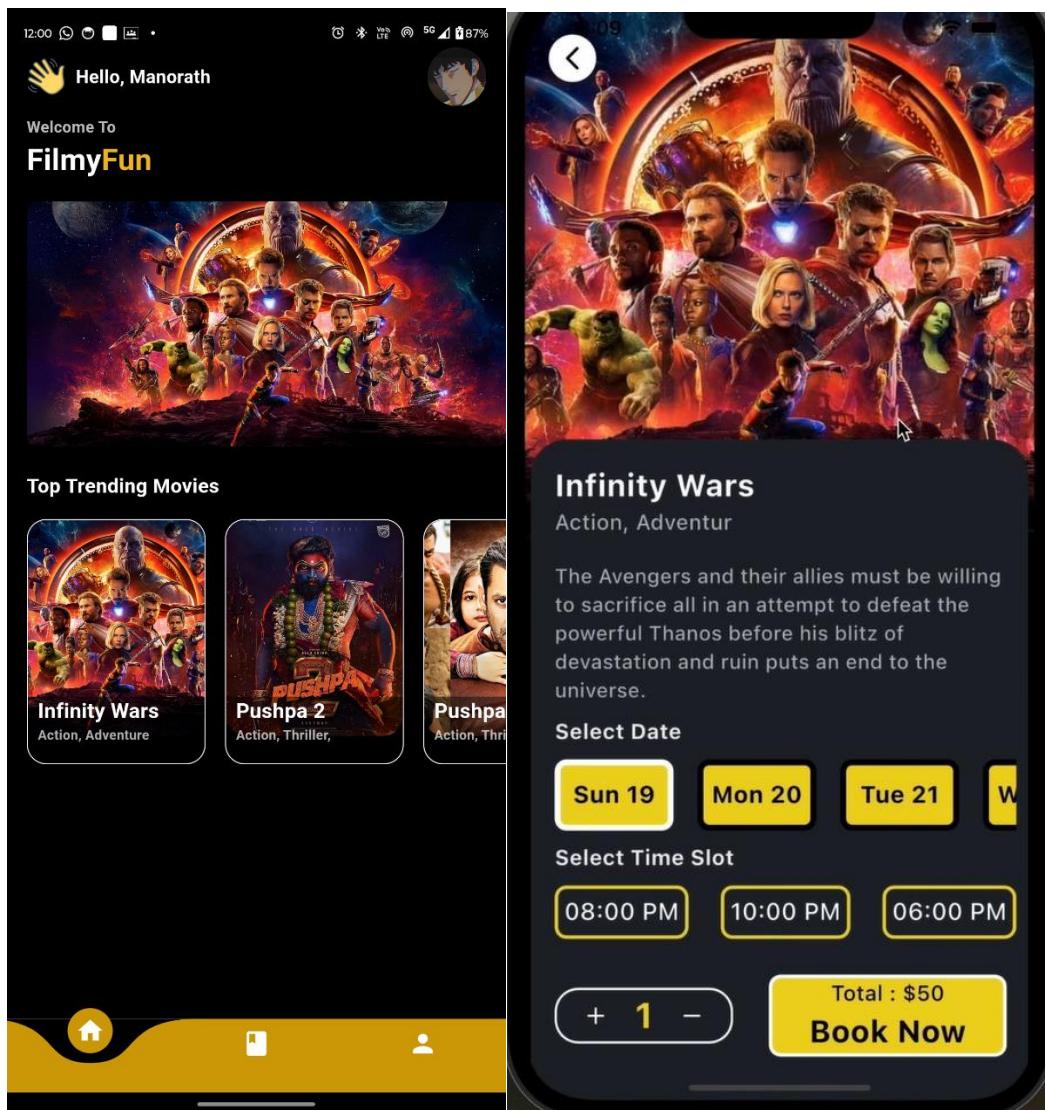
```
width: 170,
padding: EdgeInsets.all(10),
decoration: BoxDecoration(
  color: Colors.white,
  borderRadius: BorderRadius.circular(30)),
child: Text(
  "SignUp",
  style: TextStyle(
    color: Color(0xff6b63ff),
    fontSize: 25.0,
    fontWeight: FontWeight.bold),
  textAlign: TextAlign.center,
),
),
),
),
GestureDetector(
onTap: () {
  Navigator.push(
    context,
    MaterialPageRoute(
      builder: (context) => const Bottomnav())));
},
child: Container(
  width: 170,
  padding: EdgeInsets.all(10),
  decoration: BoxDecoration(
    color: Color(0xff6b63ff),
    borderRadius: BorderRadius.circular(30)),
  child: Text(
    "Login",
    style: TextStyle(
      color: Colors.white,
      fontSize: 25.0,
      fontWeight: FontWeight.bold),
    textAlign: TextAlign.center,
),
),
),
),
],
),
],
),
);
}
}
```

OUTPUT:

After clicking on Already have an account? It navigates to login



On Homepage after clicking on Movie image it navigates to Details page



EXPERIMENT NO: - 06

Name:- Manorath Ital

Class:- D15A

Roll:No: - 19

AIM: - To connect Flutter UI with Firebase database.

Theory: -

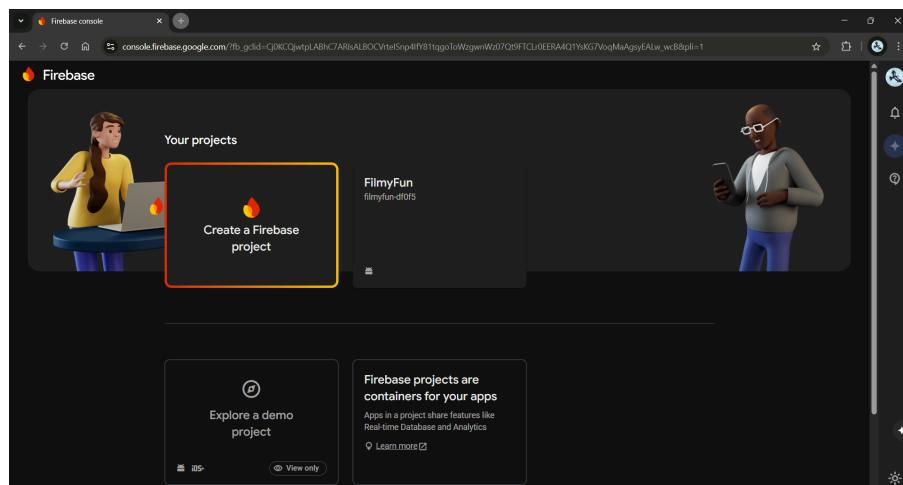
Flutter is an open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. Firebase, a Backend-as-a-Service (BaaS) platform, provides real-time database, authentication, and cloud storage services, making it a powerful backend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

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

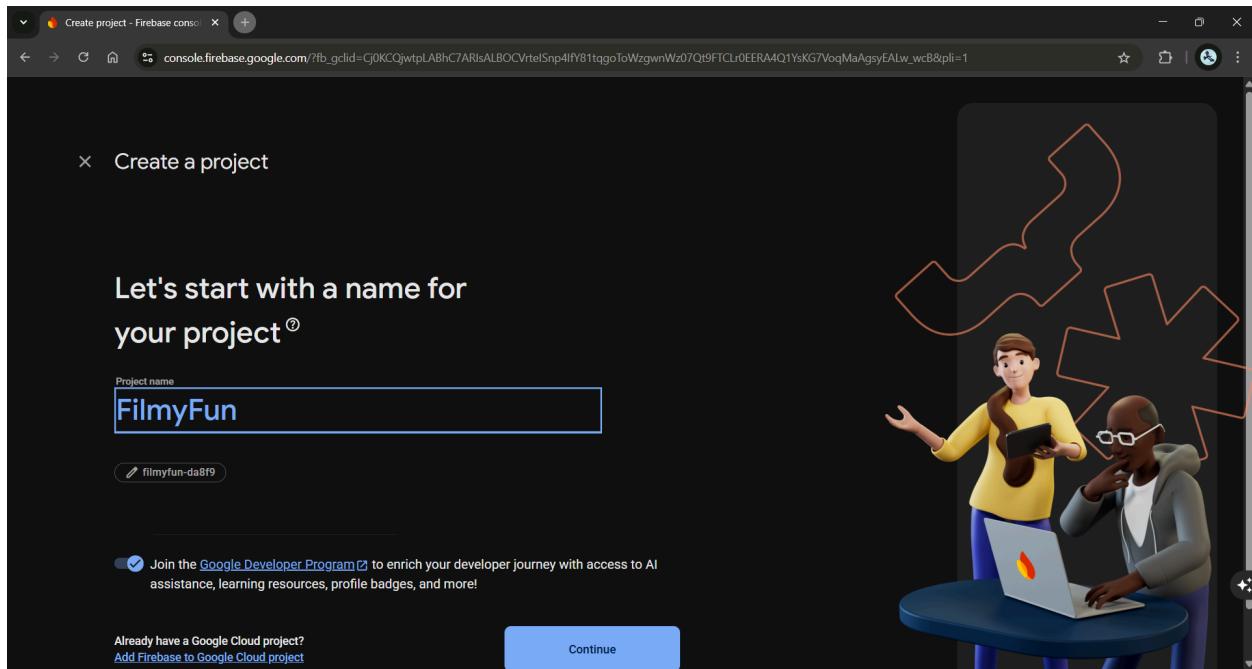
Steps to Connect Flutter UI with Firebase Database

Step 1:

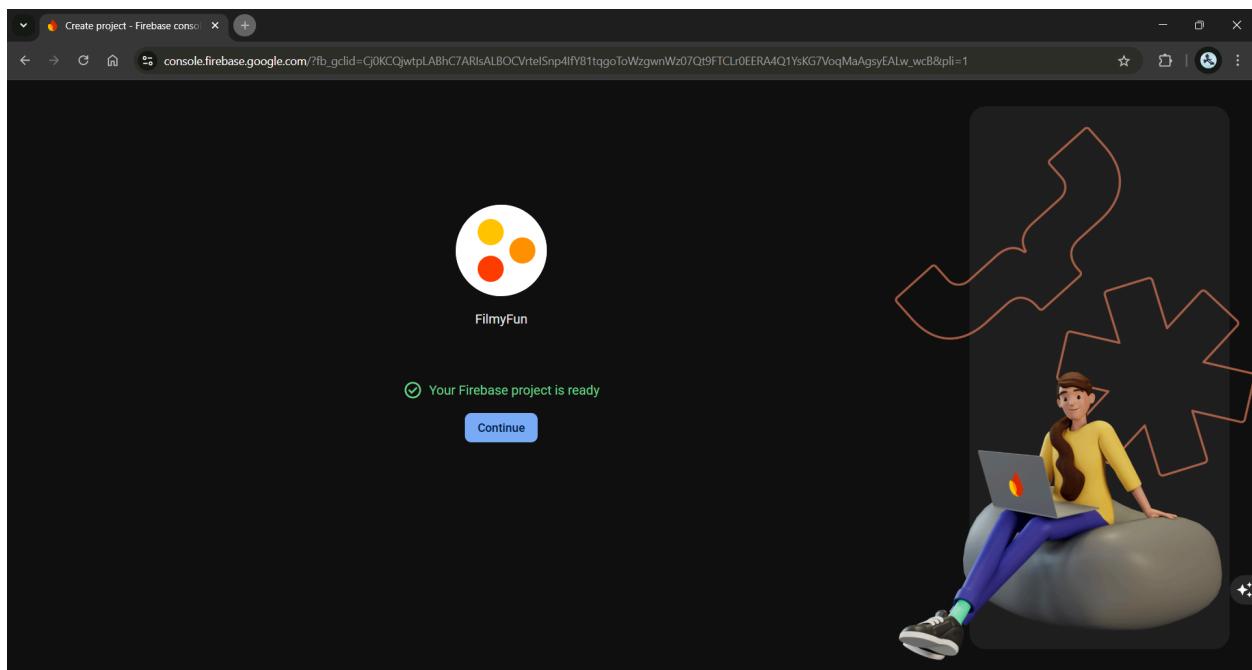
1.1) Go to Firebase Console and Create a Firebase Project



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

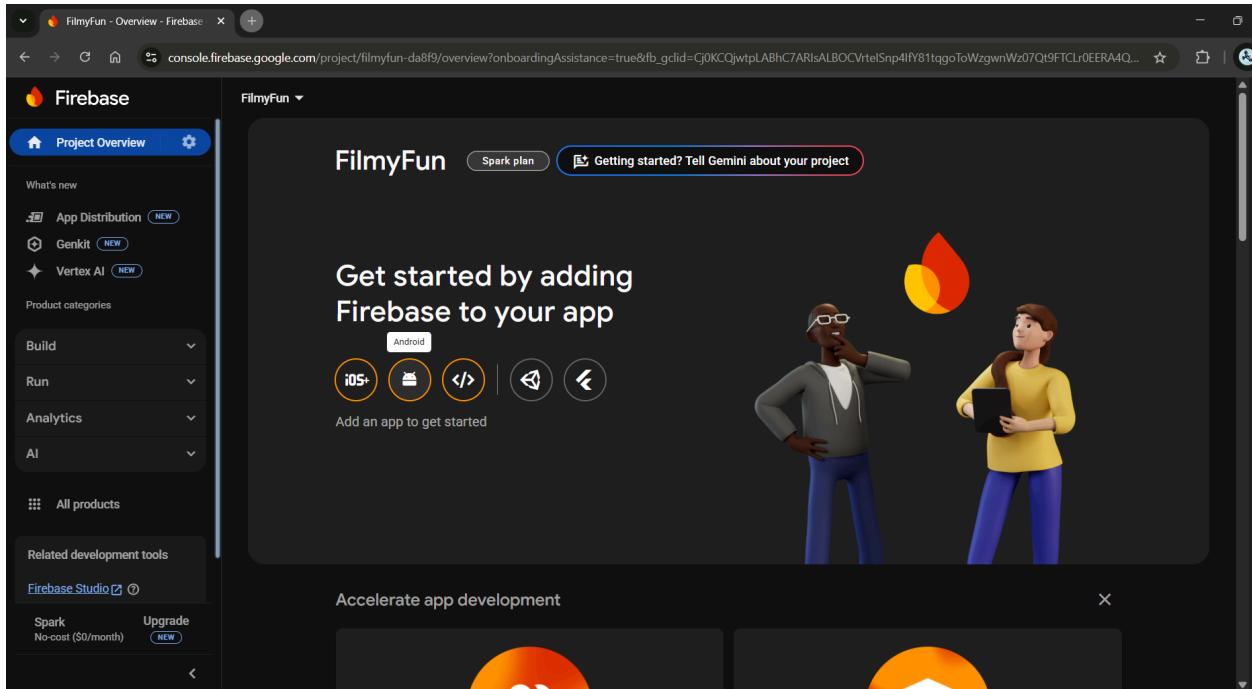


1.3) Enable Google Analytics (optional) & Click continue and complete the setup

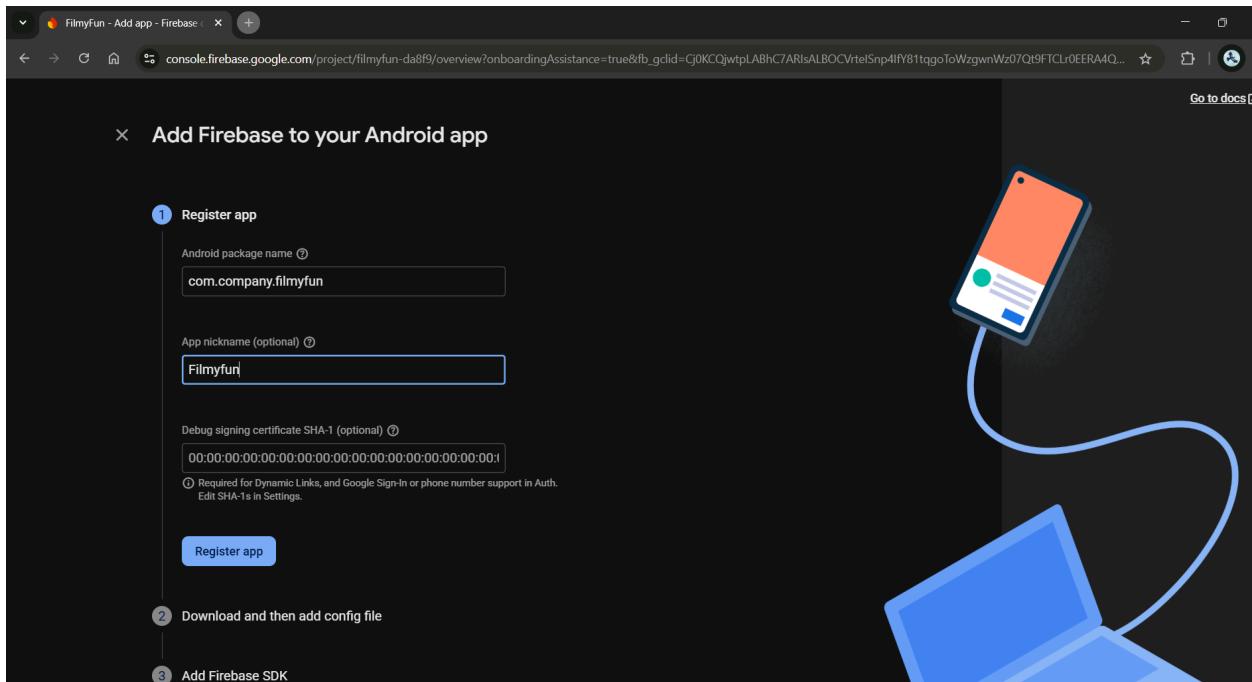


Step 2:- Add Firebase to Your Flutter App

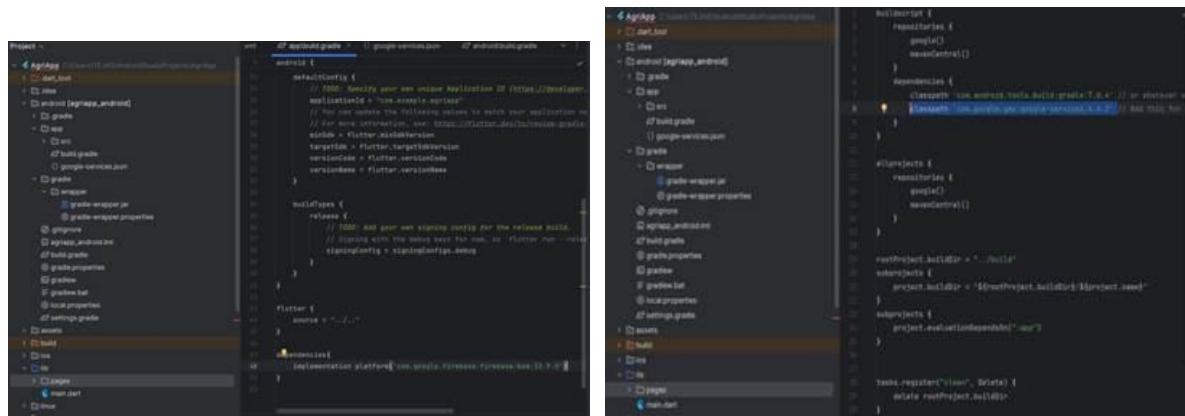
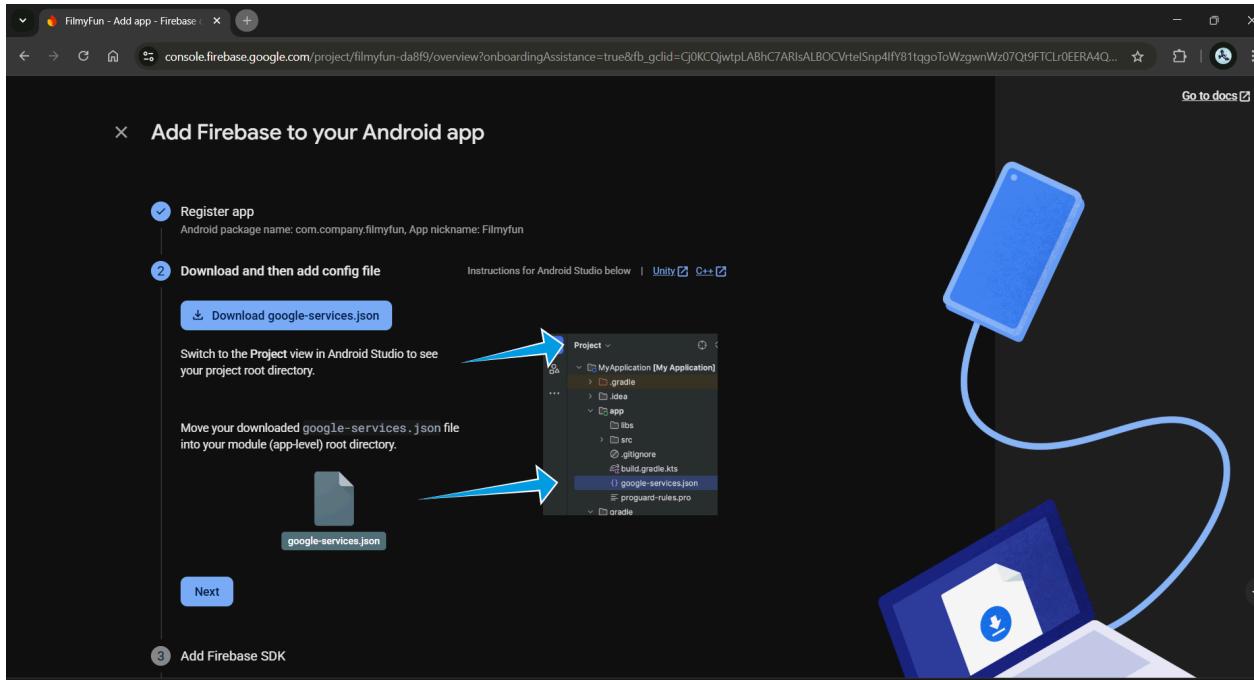
2.1) Click on Android/iOS/Web based on your Flutter application



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



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



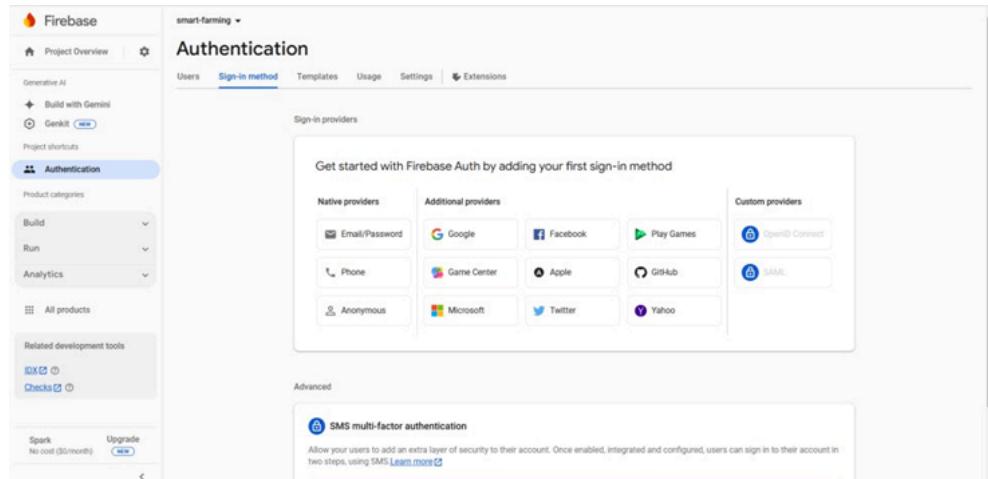
Step 3: - Add Firebase Authentication to Your App

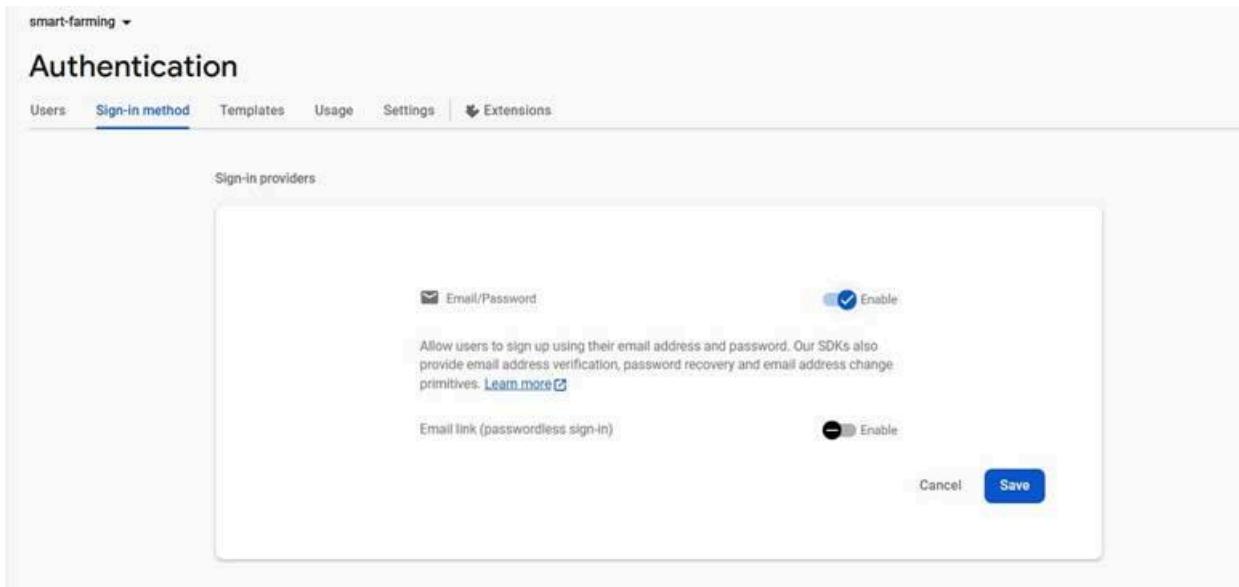
3.1) Add Firebase Authentication Dependencies

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

3.2) Enable Authentication in Firebase Console Go to Firebase Console → Authentication.

Click on Sign-in method and enable Email/Password (or any other method like Google).





3.3) Implement Authentication in Flutter Modify main.dart

```
import
'package:firebase_core/firebase_co
re.dart'; import
'package:firebase_auth/firebase_a
uth.dart',
```

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

Step 4: -Configure Firebase Realtime Database

4.1) Go to Firebase Console → Realtime Database.

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

4.3) Click Publish.

After that data gets stored in database

The screenshot shows the Firebase Authentication console for the project "FilmyFun". The left sidebar has "Authentication" selected. The main area is titled "Authentication" and contains tabs for "Users", "Sign-in method", "Templates", "Usage", "Settings", and "Extensions". A message at the top states: "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." Below this is a table listing users:

Identifier	Providers	Created	Signed In	User UID
manorath@gmail.com	✉️	Feb 19, 2025	Feb 19, 2025	3yqBAtNuKybNs065kSDpZJ3...
final@gmail.com	✉️	Feb 17, 2025	Feb 17, 2025	LXNVSBnQSACFYUQEWZwOB...
user@gmail.com	✉️	Feb 16, 2025	Feb 16, 2025	wjKAn5xPY6dVw4qkUPsMoK...
2022.manorath.ital@ve...	✉️	Feb 16, 2025	Feb 16, 2025	BWZlwjuvsG08VLldQMC3Cjy...
manorathital27@gmail...	✉️	Feb 16, 2025	Feb 16, 2025	SGc2jCQe5Nh7rP8GO0ZCVEO...

The screenshot shows the Firebase Firestore console for the project "FilmyFun". The left sidebar has "Firestore Database" selected. The main area is titled "Cloud Firestore" and shows a hierarchical view of collections: "users" > "2xK2GZTR5K". This collection contains documents with the following data:

Document ID	Email	Id	Image	Name
688x39jon7	"manorathital27@gmail.com"	"2xK2GZTR5K"	-	"Manorath Ital"
887C0213Q9				
Nuach58210				
b980H1o779				