

EX.NO	DATE	NAME OF THE EXPERIMENT	PAGE NO	MARKS (10)	STAFF SIGNATURE
1		Study and installation of Flutter/Kotlin multi-platform environment	1		
2		Develop an application that uses Widgets, GUI components, Fonts, and Colors.	4		
3		Develop a native calculator application.	8		
4		Develop a gaming application that uses 2-D animations and gestures.	13		
5		Develop a movie rating application (similar to IMDB).	17		
6		Develop an application to connect to a web service and to retrieve data with HTTP.	23		
7		Develop a simple shopping application.	27		
8		Design a web server supporting push notifications.	40		
9		Develop an application by integrating Google maps	45		
10		Mini Projects involving Flutter/Kotlin multi-platform	48		

**Aim:**

The aim of this guide is to help you set up a Flutter multi-platform development environment using Android Studio. This includes installing Flutter, configuring Android Studio, and creating a basic Flutter project that can be run on both Android and iOS platforms.

**1. Install Flutter SDK:**

- Download the Flutter SDK from the official website: Flutter SDK
- Extract the downloaded zip file to a location on your machine.
- Add the Flutter bin directory to your system PATH. This step is crucial for running Flutter commands from the terminal.

**2. Install Dart SDK:**

- Flutter requires Dart SDK. Download it from the Dart SDK website: Dart SDK
- Extract the Dart SDK and add its bin directory to your system PATH.

**3. Verify Flutter Installation**

- Open a terminal and run the following command to verify Flutter is correctly installed:  
**\$ flutter doctor**
- Fix any issues reported by flutter doctor until all checks pass.

**4. Install Android Studio:**

- Download and install Android Studio from the official website: Android Studio
- Open Android Studio, and install the Flutter and Dart plugins from the marketplace.

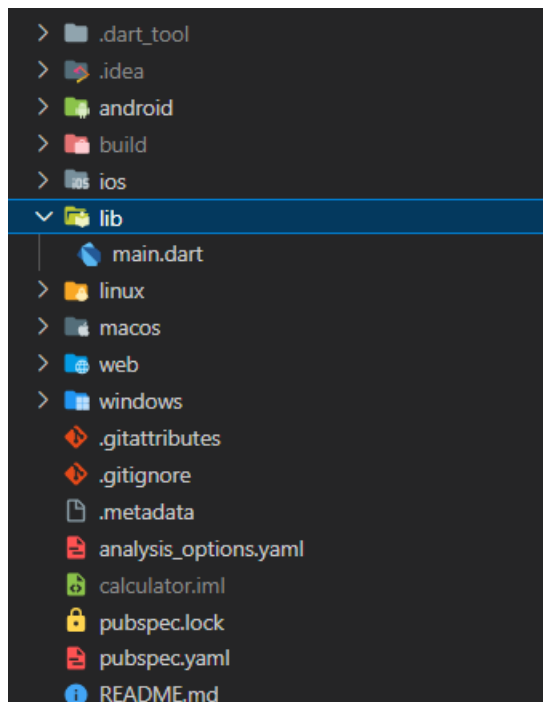
**5. Configure Flutter in Android Studio:**

- Open Android Studio, go to Preferences on macOS or Settings on Windows/Linux.
- Navigate to Languages & Frameworks > Flutter.
- Set the Flutter SDK path to the location where you extracted the Flutter SDK.

**5. Create a Flutter Project:**

- Open Android Studio and click on **File > New > New Flutter Project**.
- Choose a Flutter application template.
- Set the **project name, location**, and other details.
- Click Finish to create the project.

## Project Structure :



- **android/**: Android-specific code and configurations.
- **build/**: Auto-generated build files.
- **ios/**: iOS-specific code and configurations.
- **lib/**: Dart code for your Flutter application.
  - **main.dart**: The entry point of your Flutter app.
- **test/**: Folder for unit tests.
- **.gitignore**: File to specify files and directories to ignore in version control.
- **.metadata**: Flutter-specific metadata file.
- **.packages**: Flutter package dependencies.
- **.vscode/**: Configuration files for Visual Studio Code (if used).
- **android.iml**: Android Studio project file.
- **pubspec.lock**: Lock file specifying exact versions of dependencies.
- **pubspec.yaml**: YAML file for project configuration, including dependencies.

### 7. Run on Android Device:

- Connect an Android device or start an emulator.
- Open the terminal in Android Studio and navigate to your project directory.
- Run flutter devices to see the available devices.
- Run flutter run to build and run the Flutter app on the selected device.

### 8. Run on iOS Simulator (macOS only):

- Open the project in Android Studio.
- Open a terminal and navigate to your project directory.
- Run flutter devices to ensure an iOS simulator is available.
- Run flutter run with the target device set to the iOS simulator.

### 9. Study Notes:

- Understand the Flutter project structure, especially the lib directory where your Dart code resides.
- Explore the **pubspec.yaml** file for managing dependencies.
- Study Flutter widgets and their properties.
- Learn how to navigate between screens using **Navigator**.
- Understand the concept of **Stateful** and **Stateless** widgets.

**Result:**

Successfully Installation Of Flutter Multi-Platform Environment

**Aim:**

To Develop an application that uses Widgets, GUI components, Fonts, and Colors.

**Algorithm :****Widget Tree Structure:**

- The program begins with the main function, which calls the runApp method to start the Flutter application.
- The MyApp class is a stateless widget representing the entire application.
- MyApp creates a MaterialApp with a custom theme and sets the home page to an instance of MyHomePage.

**Home Page Widget (MyHomePage):**

- MyHomePage is a stateful widget that holds the mutable state of the counter.
- It has a corresponding state class \_MyHomePageState that extends State<MyHomePage>.

**State Class (\_MyHomePageState):**

- The state class \_MyHomePageState contains the mutable state for the counter.
- It includes an integer variable \_counter initialized to 0.
- There are two methods, \_incrementCounter and \_decrementCounter, to handle the increment and decrement operations, respectively.
- The setState method is used in both methods to trigger a rebuild of the UI when the counter changes.

**Build Method (build):**

- The build method is responsible for creating the widget tree.
- It returns a Scaffold widget, which provides the basic structure of the app, including an AppBar and a body.
- The body contains a Center widget with a Column of child widgets.
- The first child is a text widget displaying the label "Counter" with a specified style.
- The second child is another text widget displaying the current counter value, using a larger font size and a specific color.
- A SizedBox is used to add some spacing between the text and the buttons.
- The third child is a Row containing two ElevatedButton widgets with icons for increment and decrement operations.
- Each button has an onPressed callback linked to \_incrementCounter and \_decrementCounter methods.

**Increment and Decrement Methods:**

- incrementCounter and \_decrementCounter methods modify the \_counter variable using the setState function to trigger a rebuild of the UI.

**UI Update:**

- When the user taps the increment or decrement buttons, the corresponding \_incrementCounter or decrementCounter method is called.
- setState is used to notify Flutter that the internal state has changed, triggering a rebuild of the widget tree.
- The updated counter value is reflected in the UI.

## Program :

### main.dart

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Counter App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
        fontFamily: 'Roboto', // Setting a custom font
      ),
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatefulWidget {
  @override
  _MyHomePageState createState() => _MyHomePageState();
}

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

  void _incrementCounter() {
    setState(() {
      _counter++;
    });
  }

  void _decrementCounter() {
    setState(() {
      _counter--;
    });
  }

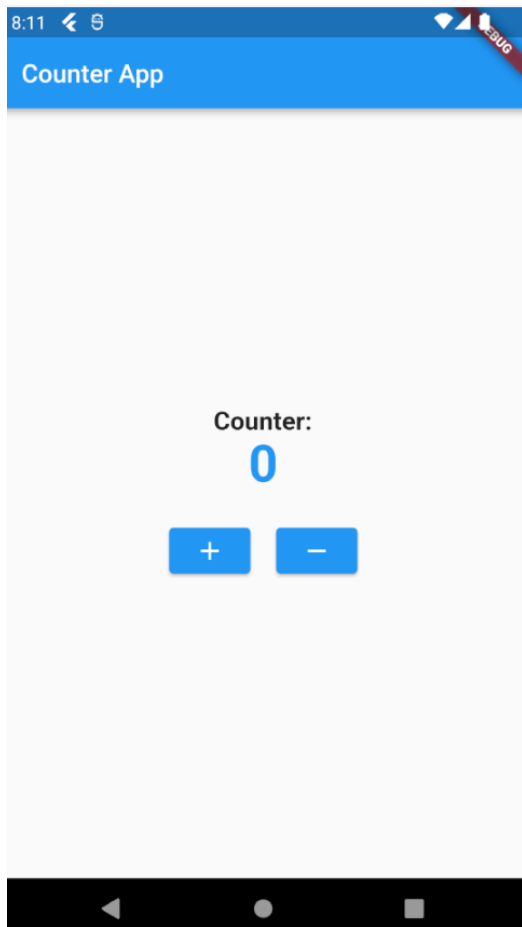
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Counter App'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
```

```

Text(
  'Counter:',
  style: TextStyle(
    fontSize: 20.0,
    fontWeight: FontWeight.bold,
  ),
),
Text(
  '$_counter',
  style: TextStyle(
    fontSize: 40.0,
    color: Colors.blue,
    fontWeight: FontWeight.bold,
  ),
),
 SizedBox(height: 20.0),
 Row(
   mainAxisAlignment: MainAxisAlignment.center,
   children: [
     ElevatedButton(
       onPressed: _incrementCounter,
       child: Icon(Icons.add),
     ),
     SizedBox(width: 20.0),
     ElevatedButton(
       onPressed: _decrementCounter,
       child: Icon(Icons.remove),
     ),
   ],
 ),
),
),
),
),
);
}
}

```

**Output:**



**Result:**

Successfully Develop an application that uses Widgets, GUI components, Fonts, and Colors.



**Aim:**

To Develop a native calculator application.

**Algorithm :****Initialization:**

- Initialize the necessary variables, including \_output, num1, num2, and operand.
- Set up the UI structure using Flutter's MaterialApp and Scaffold widgets.

**Button Press Handling (operations function):**

- The operations function is called when a button is pressed.
- It performs different actions based on the pressed button:
- If the button is a digit (0-9), it appends the digit to the current output.
- If the button is ".", it adds a decimal point to the output if one doesn't already exist.
- If the button is an arithmetic operation (+, -, \*, /), it updates num1 with the current output value, sets the operand, and resets the output for the next input.
- If the button is "=", it calculates the result based on num1, num2, and the operand.
- If the button is "CLEAR", it resets all variables for a new calculation.
- The setState function is used to update the UI with the current output.

**Button Widget (button function):**

- The button function is a utility function to create a stylized button with a specified label and onPressed function.
- It returns an OutlinedButton widget with the given properties.

**UI Structure:**

- The UI is structured using Column and Row widgets to arrange buttons in a grid-like format.
- The top section displays the previous value (history) and the current output.
- The bottom section consists of rows of digit and operation buttons.

## Program :

### main.dart

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

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.blue,
        visualDensity: VisualDensity.adaptivePlatformDensity,
      ),
      home: const MyHomePage(),
    );
  }
}

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

  @override
  _MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  String output = "";
  String previousValue = "";

  String _output = "0";
  double num1 = 0.0;
  double num2 = 0.0;
  String operand = "";

  @override
  Widget build(BuildContext context) {
    operations(String value) {
      if (value == "CLEAR") {
        _output = "0";
        num1 = 0.0;
        num2 = 0.0;
        operand = "";
        previousValue = "";
      } else if (value == "+" || value == "-" || value == "/" || value ==
"X") {
        print(output);
      }
    }
  }
}
```

```

        num1 = double.parse(output);
        operand = value;
        previousValue = output + " " + operand;
        _output = "0";
    } else if (value == ".") {
        if (_output.contains(".")) {
            print("Already contains a decimal");
            return;
        } else {
            _output = _output + value;
        }
    } else if (value == "=") {
        num2 = double.parse(output);
        if (operand == "+") {
            _output = (num1 + num2).toString();
        }
        if (operand == "-") {
            _output = (num1 - num2).toString();
        }
        if (operand == "X") {
            _output = (num1 * num2).toString();
        }
        if (operand == "/") {
            _output = (num1 / num2).toString();
        }
        previousValue = output + " " + operand + " " + num2.toString();
        num1 = 0.0;
        num2 = 0.0;
        operand = "";
    } else {
        _output = _output + value;
    }

    setState(() {
        output = double.parse(_output).toStringAsFixed(2);
    });
}

Widget button(String buttonText, Function() onPressed) {
    return Expanded(
        child: OutlinedButton(
            onPressed: onPressed,
            style: ButtonStyle(
                padding: MaterialStateProperty.all(const
EdgeInsets.all(24.0)),
            ),
            child: Text(
                buttonText,
                style: const TextStyle(fontSize: 20.0, fontWeight:
FontWeight.bold),
            ),
        ),
    );
}

```

```

return Scaffold(
  appBar: AppBar(
    title: const Text("Calculator App"),
  ),
  body: Container(
    child: Column(
      children: <Widget>[
        Container(
          alignment: Alignment.centerRight,
          padding:
            const EdgeInsets.symmetric(vertical: 24.0, horizontal:
12.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.end,
            children: [
              Text(
                previousValue,
                style: const TextStyle(
                  fontSize: 20.0, fontWeight: FontWeight.normal),
              ),
              Text(
                output,
                style: const TextStyle(
                  fontSize: 36.0, fontWeight: FontWeight.bold),
              ),
            ],
          ),
        ),
        const Expanded(child: Divider()),
        Column(
          children: [
            Row(children: [
              button("7", () => operations("7")),
              button("8", () => operations("8")),
              button("9", () => operations("9")),
              button("/", () => operations("/")),
            ]),
            Row(children: [
              button("4", () => operations("4")),
              button("5", () => operations("5")),
              button("6", () => operations("6")),
              button("X", () => operations("X")),
            ]),
            Row(children: [
              button("1", () => operations("1")),
              button("2", () => operations("2")),
              button("3", () => operations("3")),
              button("-", () => operations("-")),
            ]),
            Row(children: [
              button(".", () => operations(".")),
              button("0", () => operations("0")),
              button("00", () => operations("00")),
            ]),
          ],
        ),
      ],
    ),
  ),
);

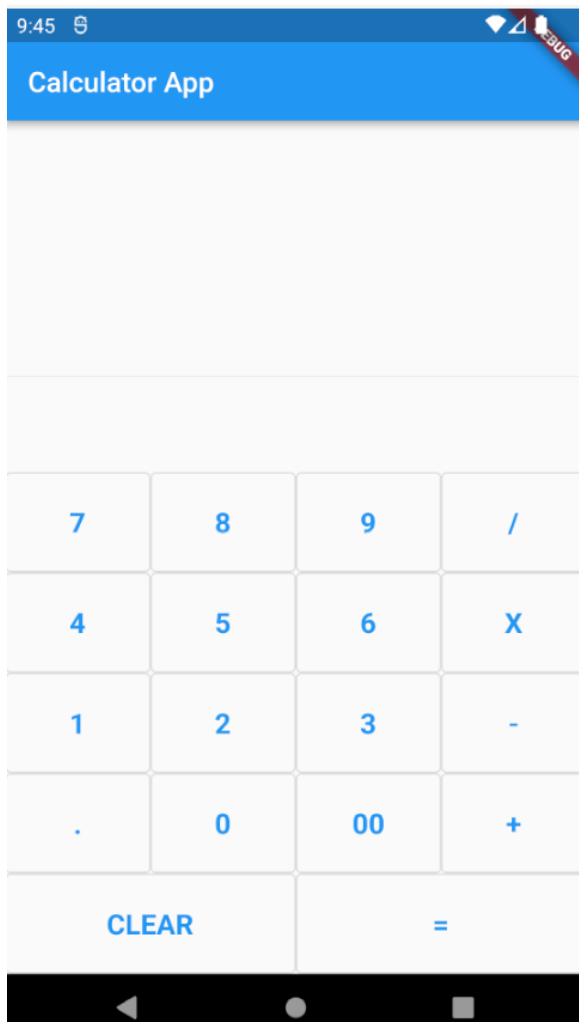
```

```

        button("+", () => operations("+")),
    ],
    Row(children: [
        button("CLEAR", () => operations("CLEAR")),
        button("=", () => operations("=")),
    ]),
  ],
),
),
),
);
}
}

```

**Output:**



**Result:**

Successfully Develop a native calculator application .

**Aim:**

To Develop a gaming application that uses 2-D animations and gestures.

**Algorithm :****Widget Tree Structure:**

- The program starts with the main function, calling runApp to initiate the Flutter application.
- MyApp is a stateless widget representing the entire application, and it creates a MaterialApp with the home set to an instance of MazeGame.

**Maze Game Widget (MazeGame):**

- MazeGame is a stateful widget with a corresponding state class \_MazeGameState.
- **State Class (\_MazeGameState):**
- \_MazeGameState contains the mutable state for the maze game.
- It includes a boolean variable success to track whether the player successfully completed the maze.

**Build Method (build):**

- The build method creates a Scaffold with an AppBar and a body containing a Center widget.
- Inside the Center, there's a ListView containing a Column with child widgets.
- The Maze widget is used to display the maze game with a specified player, columns, rows, wall thickness, wall color, finish, and a callback function onFinish triggered when the player reaches the destination.

**Game Completion (onFinish Callback):**

- The onFinish callback is triggered when the player successfully completes the maze.
- It sets the success variable to true and calls \_showSuccessDialog to display a congratulatory dialog box.

**Success Dialog (\_showSuccessDialog Method):**

- \_showSuccessDialog creates and displays an AlertDialog when the player successfully completes the maze.
- The dialog contains a title, content, and two ElevatedButton widgets for restarting the game or closing the dialog.
- Pressing the "Restart" button resets the game state, and pressing "Close" can perform additional actions.

Dependencies Packages:

```
dependencies:  
  flutter:  
    sdk: flutter  
  cupertino_icons: ^1.0.2  
  maze: ^3.0.0
```

**Program :**

Download Assets From : [https://github.com/ramtsp/flutter\\_Assets/tree/main/Ex4-assets](https://github.com/ramtsp/flutter_Assets/tree/main/Ex4-assets)

**main.dart**

```
import 'package:flutter/material.dart';  
import 'package:maze/maze.dart';  
  
void main() {  
  runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: MazeGame(),  
    );  
  }  
}  
  
class MazeGame extends StatefulWidget {  
  @override  
  _MazeGameState createState() => _MazeGameState();  
}  
  
class _MazeGameState extends State<MazeGame> {  
  bool success = false;  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(  
        title: Text('Maze Game'),  
      ),  
      body: Center(  
        child: ListView(  
          children: [  
            Column(  
              mainAxisAlignment: MainAxisAlignment.center,  
              children: [  
                Maze(  

```

```

        player: MazeItem(
          'assets/player.png',
          ImageType.asset,
        ),
        columns: 7, // Increase the number of columns
        rows: 7, // Increase the number of rows
        wallThickness: 4.0,
        wallColor: Colors.blue,
        finish: MazeItem(
          'assets/finish.png',
          ImageType.asset,
        ),
        onFinish: () {
          // Handle game completion
          setState(() {
            success = true;
          });
          _showSuccessDialog(context);
        },
      ),
      SizedBox(height: 20),
      if (success)
        Text(
          'Congratulations! You reached the destination!',
          style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold),
        ),
    ],
  ),
],
),
),
);
}

void _showSuccessDialog(BuildContext context) {
  showDialog(
    context: context,
    builder: (BuildContext context) {
      return AlertDialog(
        title: Text('Congratulations!'),
        content: Text('You successfully completed the maze!'),
        actions: [
          ElevatedButton(
            onPressed: () {
              Navigator.of(context).pop();
              // Reset the game state
              setState(() {
                success = false;
              });
            },
            child: Text('Restart'),
          ),
          ElevatedButton(

```

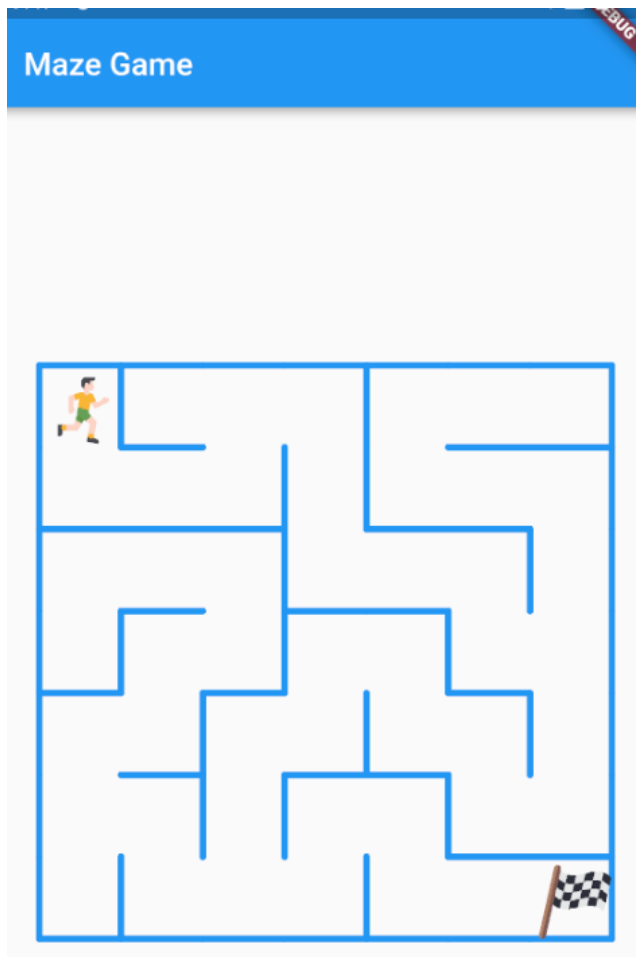


```

        onPressed: () {
          Navigator.of(context).pop();
          // Additional actions after closing the dialog
        },
        child: Text('Close'),
      ),
    ],
  ),
);
}
}

```

**Output:**



**Result:**

Successfully Develop a gaming application that uses 2-D animations and gestures.

**Aim:**

To Develop a movie rating application (similar to IMDB).

**Algorithm :****Model Class (Movie):**

- The Movie class represents a movie with attributes like id, title, overview, director, rating, and imageUrl.
- It includes a factory method fromJson to create a Movie instance from a JSON map.
- **Movie Service (MovieService):**
- MovieService is responsible for making API requests to retrieve movie data.
- The class includes methods like getMovies and getMovieDetails.
- The getMovies method fetches a list of popular movies from the TMDb API.
- The getMovieDetails method fetches details for a specific movie using its ID.

**App Entry Point (main):**

- The main function calls runApp to start the Flutter application with the MyApp widget as the root.

**Root Widget (MyApp):**

- MyApp is a stateless widget representing the entire application.
- It creates a MaterialApp with a title, theme, and sets MovieListScreen as the home screen.

**Movie List Screen (MovieListScreen):**

- MovieListScreen is a stateless widget displaying a list of movies.
- It includes a FutureBuilder to handle the asynchronous loading of movie data using MovieService.

**Movie Card Widget (MovieCard):**

- MovieCard is a stateless widget representing a card for each movie in the list.
- It includes an InkWell for a tap gesture, leading to the MovieDetailScreen.
- The widget displays the movie's image, title, director, and rating.

**Movie Detail Screen (MovieDetailScreen):**

- MovieDetailScreen is a stateless widget displaying detailed information about a specific movie.
- It includes a FutureBuilder to handle the asynchronous loading of movie details using MovieService.

**UI Building in Movie Detail Screen:**

- The UI includes the movie title, director, rating, an image of the movie, and an overview.

**Navigation Between Screens:**

- Tapping on a movie card in MovieListScreen navigates to the MovieDetailScreen with the selected movie's ID.

**Error Handling:**

- The FutureBuilder widget handles different states (loading, error, data) and displays appropriate widgets based on the state.

Dependencies Packages:

```
dependencies:  
  flutter:  
    sdk: flutter  
  http: ^1.0.0
```

Program :

```
import 'dart:convert';  
import 'package:flutter/material.dart';  
import 'package:http/http.dart' as http;  
  
void main() {  
  runApp(MyApp());  
}  
  
class Movie {  
  final int id;  
  final String title;  
  final String overview;  
  final String director;  
  final double rating;  
  final String imageUrl;  
  
  Movie({  
    required this.id,  
    required this.title,  
    required this.overview,  
    required this.director,  
    required this.rating,  
    required this.imageUrl,  
  });  
  
  factory Movie.fromJson(Map<String, dynamic> json) {  
    return Movie(  
      id: json['id'],  
      title: json['title'],  
      overview: json['overview'],  
      director: json['director'] ??  
        '', // Replace 'director' with the appropriate field from your  
API  
      rating: (json['vote_average'] ?? 0.0).toDouble(),  
      imageUrl: 'https://image.tmdb.org/t/p/w500${json['poster_path']}',  
    );  
  }  
}  
  
class MovieService {  
  final String apiKey =  
    '6e88b2c6b20e981d818f3d9a68b045d9'; // Replace with your TMDb API  
key
```

```

Future<List<Movie>> getMovies() async {
  final response = await http.get(
    Uri.parse('https://api.themoviedb.org/3/movie/popular?api_key=$apiKey'),
  );

  if (response.statusCode == 200) {
    final List<dynamic> data = json.decode(response.body)['results'];
    return data.map((json) => Movie.fromJson(json)).toList();
  } else {
    throw Exception('Failed to load movies');
  }
}

Future<Movie> getMovieDetails(int movieId) async {
  final response = await http.get(
    Uri.parse('https://api.themoviedb.org/3/movie/$movieId?api_key=$apiKey'),
  );

  if (response.statusCode == 200) {
    final Map<String, dynamic> data = json.decode(response.body);
    return Movie.fromJson(data);
  } else {
    throw Exception('Failed to load movie details');
  }
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Movie Rating App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MovieListScreen(),
    );
  }
}

class MovieListScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Movie List'),
      ),
      body: FutureBuilder<List<Movie>>(
        future: MovieService().getMovies(),
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.waiting) {

```

```

        return Center(child: CircularProgressIndicator());
    } else if (snapshot.hasError) {
        return Center(child: Text('Error: ${snapshot.error}'));
    } else if (!snapshot.hasData || snapshot.data!.isEmpty) {
        return Center(child: Text('No movies available.'));
    } else {
        return ListView.builder(
            itemCount: snapshot.data!.length,
            itemBuilder: (context, index) {
                return MovieCard(movie: snapshot.data![index]);
            },
        );
    }
},
),
);
}
}

class MovieCard extends StatelessWidget {
    final Movie movie;

    MovieCard({required this.movie});

    @override
    Widget build(BuildContext context) {
        return Card(
            elevation: 5,
            margin: EdgeInsets.all(10),
            child: InkWell(
                onTap: () {
                    Navigator.push(
                        context,
                        MaterialPageRoute(
                            builder: (context) => MovieDetailScreen(movieId: movie.id),
                        ),
                    );
                },
                child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                        Image.network(
                            movie.imageUrl,
                            height: 200,
                            width: double.infinity,
                            fit: BoxFit.cover,
                        ),
                        Padding(
                            padding: const EdgeInsets.all(10.0),
                            child: Column(
                                crossAxisAlignment: CrossAxisAlignment.start,
                                children: [
                                    Text(
                                        movie.title,

```

```

        style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold),
    ),
    SizedBox(height: 5),
    Text('Director: ${movie.director}'),
    SizedBox(height: 5),
    Text('Rating: ${movie.rating}'),
  ],
),
),
),
),
),
);
}
}

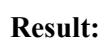
class MovieDetailScreen extends StatelessWidget {
  final int movieId;

  MovieDetailScreen({required this.movieId});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Movie Details'),
      ),
      body: FutureBuilder<Movie>(
        future: MovieService().getMovieDetails(movieId),
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.waiting) {
            return Center(child: CircularProgressIndicator());
          } else if (snapshot.hasError) {
            return Center(child: Text('Error: ${snapshot.error}'));
          } else {
            return Padding(
              padding: const EdgeInsets.all(16.0),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Text(
                    snapshot.data!.title,
                    style: TextStyle(fontSize: 24, fontWeight:
FontWeight.bold),
                  ),
                  SizedBox(height: 10),
                  Text('Director: ${snapshot.data!.director}'),
                  SizedBox(height: 10),
                  Text('Rating: ${snapshot.data!.rating}'),
                  SizedBox(height: 20),
                  Image.network(snapshot.data!.imageUrl),
                  SizedBox(height: 20),
                  Text('Overview: ${snapshot.data!.overview}'),
                ],
              ),
            );
          }
        },
      ),
    );
  }
}

```

**Output:**



22

**Aim:**

To Develop an application to connect to a web service and to retrieve data with HTTP.

**Algorithm :****App Initialization (main function):**

- The program starts with the main function, which calls runApp to start the Flutter application with MyApp as the root widget.

**Root Widget (MyApp):**

- MyApp is a stateless widget representing the entire application.
- It creates a MaterialApp and sets MyHomePage as the home screen.

**Home Page Widget (MyHomePage):**

- MyHomePage is a stateful widget representing the main screen of the application.
- It includes a list of user data fetched from the web service.

**Initialization (initState method):**

- The initState method is called when the MyHomePage widget is created.
- Inside initState, the fetchData method is called to fetch user data from the web service.

**Data Fetching (fetchData method):**

- The fetchData method sends an HTTP GET request to the 'https://randomuser.me/api/?results=10' endpoint to retrieve user data.
- If the response status code is 200 (OK), the JSON data is decoded and stored in the data list.
- If there is an error or the response code is not 200, an exception is thrown.

**UI Building (build method):**

- The build method returns a Scaffold widget containing an AppBar and a ListView.builder.
- The ListView.builder generates a list of Card widgets, each representing user information.
- The user information includes a profile picture, name, email, street, city, and latitude.

**User Information Display (ListView.builder):**

- For each user in the data list, a Card is created with a ListTile containing user information.
- The CircleAvatar displays the user's profile picture, and the ListTile displays the user's name, email, street, city, and latitude.



### Dependencies Packages:

```
dependencies:  
  flutter:  
    sdk: flutter  
  http: ^1.1.0
```

### Program :

```
import 'dart:convert';  
import 'package:flutter/material.dart';  
import 'package:http/http.dart' as http;  
  
void main() {  
  runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: MyHomePage(),  
    );  
  }  
}  
  
class MyHomePage extends StatefulWidget {  
  @override  
  _MyHomePageState createState() => _MyHomePageState();  
}  
  
class _MyHomePageState extends State<MyHomePage> {  
  List<dynamic> data = [];  
  
  @override  
  void initState() {  
    super.initState();  
    fetchData();  
  }  
  
  Future<void> fetchData() async {  
    final response =  
      await  
http.get(Uri.parse('https://randomuser.me/api/?results=10'));  
  
    if (response.statusCode == 200) {  
      // If the server returns a 200 OK response, parse the data  
      Map<String, dynamic> userData = json.decode(response.body);  
      List<dynamic> users = userData['results'];  
      setState(() {  
        data = users;  
      });  
    }  
  }  
}
```

```

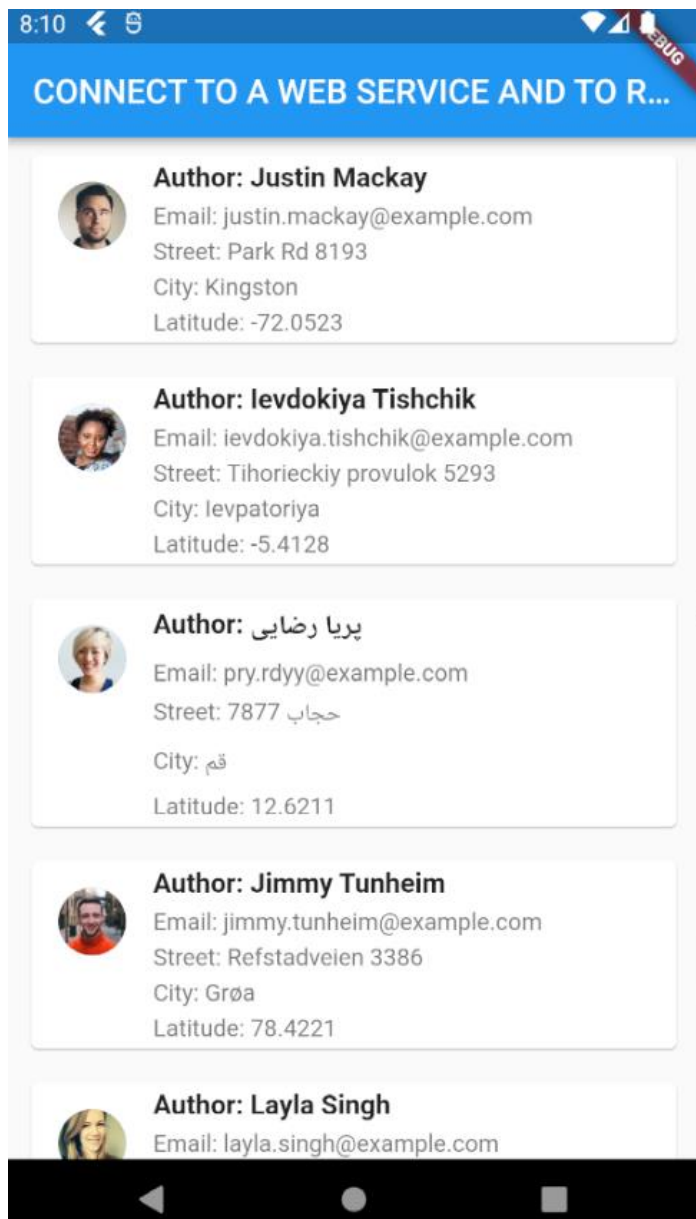
    } else {
        // If the server did not return a 200 OK response,
        // throw an exception.
        throw Exception('Failed to load data');
    }
}

@override
Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
            title: Text('CONNECT TO A WEB SERVICE AND TO RETRIEVE DATA '),
        ),
        body: ListView.builder(
            itemCount: data.length,
            itemBuilder: (context, index) {
                var user = data[index];
                var picture = user['picture'];
                var address = user['location'];
                var coordinates = address['coordinates'];

                return Card(
                    margin: EdgeInsets.symmetric(vertical: 10, horizontal: 15),
                    child: ListTile(
                        leading: CircleAvatar(
                            backgroundImage: NetworkImage(picture['large']),
                        ),
                        title: Text(
                            'Author: ${user['name']['first']}
                            ${user['name']['last']}',
                            style: TextStyle(fontWeight: FontWeight.bold),
                        ),
                        subtitle: Column(
                            crossAxisAlignment: CrossAxisAlignment.start,
                            children: [
                                SizedBox(height: 5),
                                Text('Email: ${user['email']}'),
                                SizedBox(height: 5),
                                Text(
                                    'Street: ${address['street']['name']}
                                    ${address['street']['number']}',
                                ),
                                SizedBox(height: 5),
                                Text('City: ${address['city']}'),
                                SizedBox(height: 5),
                                Text('Latitude: ${coordinates['latitude']}'),
                            ],
                        ),
                    ),
                );
            },
        );
}

```

### Output:



### Result:

Successfully Develop an application to connect to a web service and to retrieve data with HTTP.

**Aim:**

To Develop a simple shopping application.

**Algorithm :****Initialize Flutter Project:**

- Use Flutter CLI or an IDE to create a new Flutter project.

**Define Product Model:**

- Create a Dart class to represent the product model with attributes like id, name, price, and image.

**Create Product Data:**

- Define a list of sample products within the main Dart file or a dedicated data file.

**Design Product List Screen:**

- Create a widget for displaying a list of products using ListView.builder.

**Design Product Card Widget:**

- Create a widget for displaying a product card with details like image, name, price, and an "Add to Cart" button.

**Create Shopping Cart Model:**

- Define a shopping cart model to manage selected products.

**Design Shopping Cart Screen:**

- Create a screen to display the contents of the shopping cart, listing selected products.

**Integrate Navigation:**

- Implement navigation between the product list screen and the shopping cart screen using the Navigator class.

**Implement Add to Cart Functionality:**

- Update the ProductCard widget to handle the "Add to Cart" button tap and add the selected product to the shopping cart.

**Navigate to Shopping Cart Screen:**

- Add a button in the ProductListScreen to navigate to the shopping cart screen.

**Run the Application:**

- Execute the Flutter run command to test the application on an emulator or physical device.

**Test the Application:**

- Interact with the application, add products to the cart, and navigate between screens to ensure proper functionality.

### Dependencies Packages:

```
dependencies:  
  flutter:  
    sdk: flutter  
  animate_do: ^2.1.0  
  page_transition: ^2.1.0  
  cupertino_icons: ^1.0.2  
  font_awesome_flutter: ^10.6.0
```

### Program :

Download Assets From : [https://github.com/ramtsp/flutter\\_Assets/tree/main/Ex7-assets/images](https://github.com/ramtsp/flutter_Assets/tree/main/Ex7-assets/images)

### main.dart

```
import 'package:animate_do/animate_do.dart';  
import 'package:shoppingapp/Pages/ShopPage.dart';  
import 'package:flutter/material.dart';  
import 'package:page_transition/page_transition.dart';  
  
void main() =>  
  runApp(MaterialApp(debugShowCheckedModeBanner: false, home:  
    HomePage()));  
  
class HomePage extends StatefulWidget {  
  @override  
  _HomePageState createState() => _HomePageState();  
}  
  
class _HomePageState extends State<HomePage> with TickerProviderStateMixin  
{  
  late AnimationController _scaleController;  
  late Animation<double> _scaleAnimation;  
  
  bool hide = false;  
  
  @override  
  void initState() {  
    // TODO: implement initState  
    super.initState();  
  
    _scaleController =  
      AnimationController(vsync: this, duration: Duration(milliseconds:  
600));  
  
    _scaleAnimation = Tween<double>(begin: 1.0, end: 30.0)  
      .animate(_scaleController)  
      ..addListener((status) {  
        if (status == AnimationStatus.completed) {  
          Navigator.push(context,
```

```

        PageTransition(type: PageTransitionType.fade, child:
ShopPage())));
    }
  });
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Container(
      width: double.infinity,
      decoration: BoxDecoration(
        image: DecorationImage(
          image: AssetImage('assets/images/splash.jpg'),
          fit: BoxFit.cover)),
      child: Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(begin: Alignment.bottomRight,
colors: [
          Colors.black.withOpacity(.9),
          Colors.black.withOpacity(.4),
        ])),
        child: Padding(
          padding: const EdgeInsets.all(30.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            mainAxisAlignment: MainAxisAlignment.end,
            children: <Widget>[
              FadeInUp(
                duration: Duration(milliseconds: 1000),
                child: Text(
                  "Brand New Perspective",
                  style: TextStyle(
                    color: Colors.white,
                    fontSize: 40,
                    fontWeight: FontWeight.bold),
                )),
              SizedBox(
                height: 20,
              ),
              FadeInUp(
                duration: Duration(milliseconds: 1300),
                child: Text(
                  "Let's start with our summer collection.",
                  style: TextStyle(color: Colors.white, fontSize: 20),
                )),
              SizedBox(
                height: 100,
              ),
              InkWell(
                onTap: () {
                  setState(() {
                    hide = true;
                  });
                }
              );
            ],
          ),
        ),
      ),
    ),
  );
}

```

```

        _scaleController.forward();
    },
    child: AnimatedBuilder(
      animation: _scaleController,
      builder: (context, child) => Transform.scale(
        scale: _scaleAnimation.value,
        child: FadeInUp(
          duration: Duration(milliseconds: 1500),
          child: Container(
            height: 50,
            decoration: BoxDecoration(
              color: Colors.white,
              borderRadius: BorderRadius.circular(50)),
            child: Center(
              child: hide == false
                ? Text(
                    "Get Start",
                    style: TextStyle(
                      fontWeight: FontWeight.bold),
                )
                : Container(),
            ),
          ),
        ),
      ),
    ),
    child: SizedBox(
      height: 20,
    ),
    child: FadeInUp(
      duration: Duration(milliseconds: 1700),
      child: Container(
        height: 50,
        decoration: BoxDecoration(
          border: Border.all(color: Colors.white),
          borderRadius: BorderRadius.circular(50)),
        child: Center(
          child: Text(
            "Create Account",
            style: TextStyle(
              color: Colors.white, fontWeight:
FontWeight.bold),
          ),
        ),
      ),
    ),
    child: SizedBox(
      height: 30,
    ),
  ],
),
),
),
),
),
);

```

```
}
}
```

## ShopPage.dart

```
import 'package:animate_do/animate_do.dart';
import 'package:shoppingapp/Pages/CategoryPage.dart';
import 'package:flutter/material.dart';

class ShopPage extends StatefulWidget {
  @override
  _ShopPageState createState() => _ShopPageState();
}

class _ShopPageState extends State<ShopPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SingleChildScrollView(
        child: Column(
          children: <Widget>[
            FadeInUp(
              duration: Duration(milliseconds: 1000),
              child: Container(
                height: 500,
                decoration: BoxDecoration(
                  image: DecorationImage(
                    image: AssetImage('assets/images/background.jpg'),
                    fit: BoxFit.cover)),
                child: Container(
                  decoration: BoxDecoration(
                    gradient: LinearGradient(
                      begin: Alignment.bottomRight,
                      colors: [
                        Colors.black.withOpacity(.8),
                        Colors.black.withOpacity(.2),
                      ],
                    )),
                  child: Padding(
                    padding: const EdgeInsets.only(top: 50.0),
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.start,
                      mainAxisAlignment: MainAxisAlignment.spaceBetween,
                      children: <Widget>[
                        Row(
                          mainAxisAlignment: MainAxisAlignment.end,
                          children: <Widget>[
                            FadeInUp(
                              duration: Duration(milliseconds: 1200),
                              child: IconButton(
                                icon: Icon(
                                  Icons.favorite,
                                  color: Colors.white,
                                ),
                                onPressed: () {},
                              ),
                            FadeInUp(
```





```

child: Column(
  children: <Widget>[
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: <Widget>[
        Text(
          "Categories",
          style: TextStyle(
            color: Colors.black,
            fontSize: 18,
            fontWeight: FontWeight.bold),
        ),
        Text("All")
      ],
    ),
    SizedBox(
      height: 20,
    ),
    Container(
      height: 150,
      child: ListView(
        scrollDirection: Axis.horizontal,
        children: <Widget>[
          makeCategory(
            image: 'assets/images/beauty.jpg',
            title: 'Beauty',
            tag: 'beauty'),
          makeCategory(
            image: 'assets/images/clothes.jpg',
            title: 'Clothes',
            tag: 'clothes'),
          makeCategory(
            image: 'assets/images/perfume.jpg',
            title: 'Perfume',
            tag: 'perfume'),
          makeCategory(
            image: 'assets/images/glass.jpg',
            title: 'Glass',
            tag: 'glass'),
        ],
      ),
    ),
    SizedBox(
      height: 40,
    ),
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: <Widget>[
        Text(
          "Best Selling by Category",
          style: TextStyle(
            color: Colors.black,
            fontSize: 18,
            fontWeight: FontWeight.bold),
        ),
        Text("All")
      ],
    ),
    SizedBox(

```

```

        height: 20,
      ),
      Container(
        height: 150,
        child: ListView(
          scrollDirection: Axis.horizontal,
          children: <Widget>[
            makeBestCategory(
              image: 'assets/images/tech.jpg', title: 'Tech'),
            makeBestCategory(
              image: 'assets/images/watch.jpg',
              title: 'Watch'),
            makeBestCategory(
              image: 'assets/images/perfume.jpg',
              title: 'Perfume'),
            makeBestCategory(
              image: 'assets/images/glass.jpg',
              title: 'Glass'),
          ],
        ),
      ),
      SizedBox(
        height: 80,
      ),
    ],
  ),
),
);
}

```

```

Widget makeCategory({image, title, tag}) {
  return AspectRatio(
    aspectRatio: 2 / 2.2,
    child: Hero(
      tag: tag,
      child: GestureDetector(
        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (context) => CategoryPage(
                image: image,
                title: title,
                tag: tag,
              )),
        ),
      ),
      child: Material(
        child: Container(
          margin: EdgeInsets.only(right: 20),
          decoration: BoxDecoration(
            borderRadius: BorderRadius.circular(10),
            image: DecorationImage(
              image: AssetImage(image), fit: BoxFit.cover)),
          child: Container(
            padding: EdgeInsets.all(10),
            decoration: BoxDecoration(

```

```

        borderRadius: BorderRadius.circular(10),
        gradient:
          LinearGradient(begin: Alignment.bottomRight, colors: [
            Colors.black.withOpacity(.8),
            Colors.black.withOpacity(.0),
          ]),
        child: Align(
          alignment: Alignment.bottomLeft,
          child: Text(
            title,
            style: TextStyle(
              color: Colors.white,
              fontWeight: FontWeight.bold,
              fontSize: 16),
          )),
      ),
    ),
  ),
);
}

Widget makeBestCategory({image, title}) {
  return AspectRatio(
    aspectRatio: 3 / 2.2,
    child: Container(
      margin: EdgeInsets.only(right: 20),
      decoration: BoxDecoration(
        borderRadius: BorderRadius.circular(10),
        image:
          DecorationImage(image: AssetImage(image), fit: BoxFit.cover)),
      child: Container(
        padding: EdgeInsets.all(10),
        decoration: BoxDecoration(
          borderRadius: BorderRadius.circular(10),
          gradient: LinearGradient(begin: Alignment.bottomRight, colors: [
            Colors.black.withOpacity(.8),
            Colors.black.withOpacity(.0),
          ]),
        child: Align(
          alignment: Alignment.bottomLeft,
          child: Text(
            title,
            style: TextStyle(
              color: Colors.white,
              fontWeight: FontWeight.bold,
              fontSize: 16),
          )),
      ),
    ),
  );
}
}

```

### CategoryPage.dart

```

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

```

```

class CategoryPage extends StatefulWidget {
  final String? title;
  final String? image;
  final String? tag;

  const CategoryPage({Key? key, this.title, this.image, this.tag}) : super(key:
key);

  @override
  _CategoryPageState createState() => _CategoryPageState();
}

class _CategoryPageState extends State<CategoryPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      body: SingleChildScrollView(
        child: Column(
          children: <Widget>[
            Hero(
              tag: widget.tag!,
              child: Material(
                child: Container(
                  height: 360,
                  decoration: BoxDecoration(
                    image: DecorationImage(
                      image: AssetImage(widget.image!),
                      fit: BoxFit.cover
                    )
                  ),
                ),
              ),
            child: Container(
              padding: EdgeInsets.all(10),
              decoration: BoxDecoration(
                gradient: LinearGradient(
                  begin: Alignment.bottomRight,
                  colors: [
                    Colors.black.withOpacity(.8),
                    Colors.black.withOpacity(.1),
                  ]
                )
              ),
            ),
            child: Column(
              children: <Widget>[
                SizedBox(height: 40,),
                Row(
                  mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: <Widget>[
                    IconButton(
                      icon: Icon(Icons.arrow_back_ios, color:
Colors.white),

                      onPressed: () {
                        Navigator.pop(context);
                      },
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.end,
                      children: <Widget>[

```



```

        FadeInUp(duration: Duration(milliseconds: 1700),
child: makeProduct(image: 'assets/images/glass.jpg', title: 'Glass', price:
'100\$')),
        FadeInUp(duration: Duration(milliseconds: 1800),
child: makeProduct(image: 'assets/images/perfume.jpg', title: 'Perfume', price:
'100\$')),
        FadeInUp(duration: Duration(milliseconds: 1900),
child: makeProduct(image: 'assets/images/person.jpg', title: 'Person', price:
'100\$')),
    ],
  ),
),
],
),
),
);
}

```

```

Widget makeProduct({image, title, price}) {
  return Container(
    height: 200,
    width: double.infinity,
    margin: EdgeInsets.only(bottom: 20),
    decoration: BoxDecoration(
      borderRadius: BorderRadius.circular(10),
      image: DecorationImage(
        image: AssetImage(image),
        fit: BoxFit.cover
      )
    ),
    child: Container(
      padding: EdgeInsets.all(10),
      decoration: BoxDecoration(
        borderRadius: BorderRadius.circular(10),
        gradient: LinearGradient(
          begin: Alignment.bottomRight,
          colors: [
            Colors.black.withOpacity(.8),
            Colors.black.withOpacity(.1),
          ]
        )
      ),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: <Widget>[
          FadeInUp(duration: Duration(milliseconds: 1400), child: Align(
            alignment: Alignment.topRight,
            child: Icon(Icons.favorite_border, color: Colors.white,)),
        ),
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            crossAxisAlignment: CrossAxisAlignment.end,
            children: <Widget>[
              Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: <Widget>[
                  FadeInUp(duration: Duration(milliseconds: 1500), child:
Text(title, style: TextStyle(color: Colors.white, fontSize: 20),)),

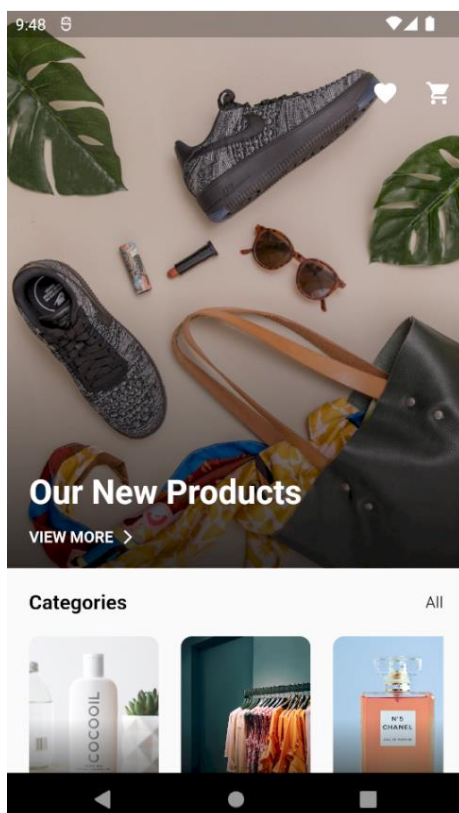
```

```

        FadeInUp(duration: Duration(milliseconds: 1500), child:
Text(price, style: TextStyle(color: Colors.white, fontSize: 30, fontWeight:
FontWeight.bold),)),
    ],
  ),
  FadeInUp(duration: Duration(milliseconds: 2000), child: Container(
    width: 40,
    height: 40,
    margin: EdgeInsets.only(bottom: 10),
    decoration: BoxDecoration(
      shape: BoxShape.circle,
      color: Colors.white
    ),
    child: Center(
      child: Icon(Icons.add_shopping_cart, size: 18, color:
Colors.grey[700],),
    ),
  ),
),
),
),
),
),
),
),
),
);
}
}

```

### Output:



### Result:

Successfully Develop a simple shopping application.



**Aim:**

To Design a web server supporting push notifications.

**Algorithm :****Initialize Awesome Notifications:**

- Call `AwesomeNotifications().initialize` in the main function to initialize the Awesome Notifications library.
- Define a notification channel and channel group to categorize notifications.

**Request Notification Permission:**

- Check if the app is allowed to send notifications using `AwesomeNotifications().isNotificationAllowed`.
- If not allowed, request permission using `AwesomeNotifications().requestPermissionToSendNotifications`.

**Set Notification Listeners:**

- In the `initState` method of `MyApp`, set up notification listeners using `AwesomeNotifications().setListeners`.
- Listeners include methods for handling notification creation, display, dismissal, and action reception.

**Build Flutter Application:**

- Create a Flutter application with a `MaterialApp` as the root widget.
- Use `Scaffold` with an `AppBar` and an `ElevatedButton` to trigger the display of a notification.

**Notification Button Press Handling:**

- Inside the `ElevatedButton` `onPressed` callback, use `AwesomeNotifications().createNotification` to send a notification.
- Define the notification content using `NotificationContent`.

**Implement NotificationController:**

- Create a separate class `NotificationController` to handle notification-related methods.
- The class includes methods for notification creation, display, dismissal, and action reception.
- These methods are annotated with `@pragma("vm:entry-point")` to ensure they are recognized by the Dart VM.

**Run the Flutter Application:**

- Use the Flutter CLI or an IDE to run the application on an emulator or physical device.
- Ensure that the notification library is correctly configured and that permissions are granted.

**Testing the Application:**

- Interact with the application by tapping the "Show Notification" button.
- Observe the behavior of the notifications and verify that the notification-related methods in `NotificationController` are called appropriately.

**Additional Considerations:**

- Explore customization options provided by the Awesome Notifications library to enhance the appearance and behavior of notifications.
- Handle more complex scenarios such as scheduled notifications or notifications with specific actions.

#### Android Manifest Configuration:

- Update the **AndroidManifest.xml** file.

```
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
```

#### Dependencies Packages:

```
dependencies:
  flutter:
    sdk: flutter
  cupertino_icons: ^1.0.2
  awesome_notifications: ^0.8.2
```

#### Program :

##### main.dart

```
import 'package:awesome_notifications/awesome_notifications.dart';
import 'package:flutter/material.dart';
import
'package:flutter_local_notifications_tutorial/notification_controller.dart';

void main() async {
  await AwesomeNotifications().initialize(null, [
    NotificationChannel(
      channelGroupKey: "basic_channel_group",
      channelKey: "basic_channel",
      channelName: "Basic Notification",
      channelDescription: "Basic notifications channel",
    )
  ], channelGroups: [
    NotificationChannelGroup(
      channelGroupKey: "basic_channel_group",
      channelGroupName: "Basic Group",
    )
  ]);
  bool isAllowedToSendNotification =
    await AwesomeNotifications().isNotificationAllowed();
  if (!isAllowedToSendNotification) {
    AwesomeNotifications().requestPermissionToSendNotifications();
  }
  runApp(const MyApp());
}

class MyApp extends StatefulWidget {
```

```

const MyApp({super.key});

@override
State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  @override
  void initState() {
    AwesomeNotifications().setListeners(
      onActionReceivedMethod: NotificationController.onActionReceivedMethod,
      onNotificationCreatedMethod:
        NotificationController.onNotificationCreatedMethod,
      onNotificationDisplayedMethod:
        NotificationController.onNotificationDisplayedMethod,
      onDismissActionReceivedMethod:
        NotificationController.onDismissActionReceivedMethod);
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: Scaffold(
        appBar: AppBar(
          title: Text('Awesome Notifications Demo'),
          backgroundColor: Colors.blue, // Set background color to primary blue
        ),
        body: Center(
          child: ElevatedButton(
            onPressed: () {
              AwesomeNotifications().createNotification(
                content: NotificationContent(
                  id: 1,
                  channelKey: "basic_channel",
                  title: "Hello Parasuram!",
                  body: "Yay! I have Push notifications working now!",
                ),
              );
            },
            style: ElevatedButton.styleFrom(
              primary: Colors.blue, // Set background color to blue
            ),
            child: Text(
              'Show Notification',
              style: TextStyle(color: Colors.white), // Set text color to white
            ),
          ),
        ),
      ),
    );
  }
}

```

### notification\_controller.dart

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

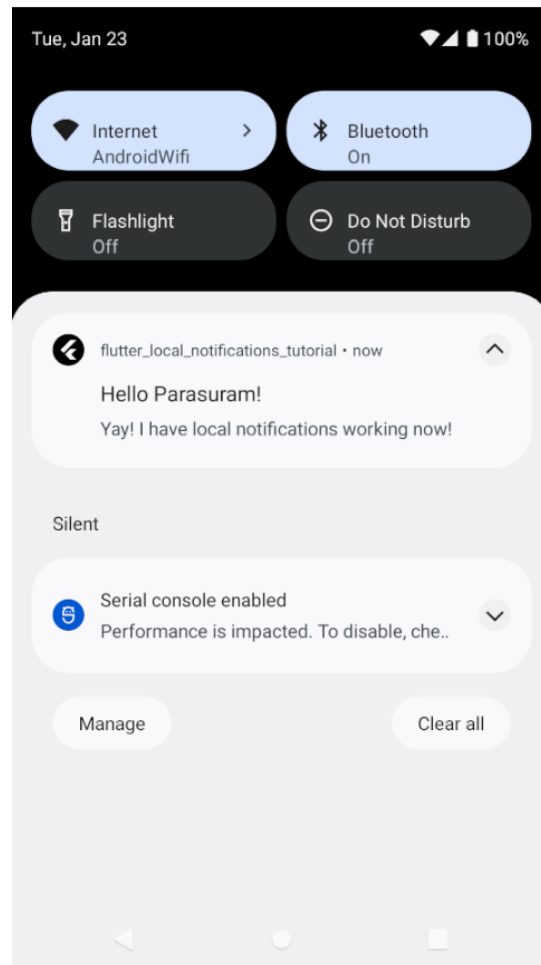
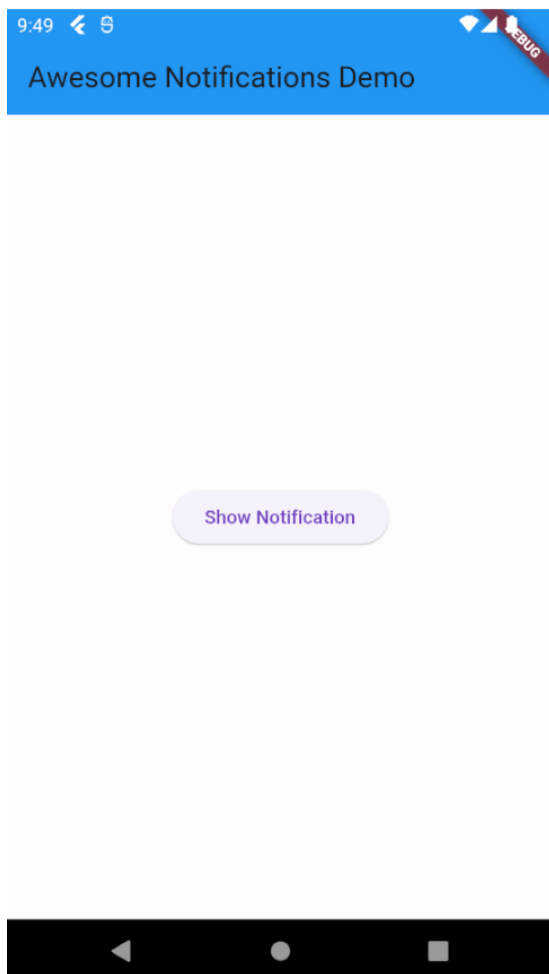
class NotificationController {
  /// Use this method to detect when a new notification or a schedule is created
  @pragma("vm:entry-point")
  static Future<void> onNotificationCreatedMethod(
    ReceivedNotification receivedNotification) async {}

  /// Use this method to detect every time that a new notification is displayed
  @pragma("vm:entry-point")
  static Future<void> onNotificationDisplayedMethod(
    ReceivedNotification receivedNotification) async {}

  @pragma("vm:entry-point")
  static Future<void> onDismissActionReceivedMethod(
    ReceivedAction receivedAction) async {}

  /// Use this method to detect when the user taps on a notification or action
  button
  @pragma("vm:entry-point")
  static Future<void> onActionReceivedMethod(
    ReceivedAction receivedAction) async {}
}
```

## Output:



## Result:

Successfully Design a web server supporting push notifications.

**Aim:**

To Develop an application by integrating Google maps .

**Algorithm :****Initialize Flutter Project:**

- Use Flutter CLI or an IDE to create a new Flutter project.

```
$ flutter create google_maps_integration
```

```
$ cd google_maps_integration
```

**Add Dependencies:**

- Open the pubspec.yaml file and add the Google Maps Flutter plugin as a dependency.

```
dependencies:
```

```
  flutter:
```

```
    sdk: flutter
```

```
  google_maps_flutter: ^2.0.6
```

**Run flutter pub get:**

- Execute the flutter pub get command to fetch and install the new dependency.

```
$ flutter pub get
```

**Create Google Maps API Key:**

- Obtain a Google Maps API key from the Google Cloud Console.

**Enable Google Maps API:**

- Enable the Google Maps API for Android and iOS in the Google Cloud Console.

**Android Manifest Configuration:**

- Update the **AndroidManifest.xml** file with the Google Maps API key.

```
<application>
```

```
<meta-data
```

```
  android:name="com.google.android.geo.API_KEY"
```

```
  android:value="YOUR_API_KEY_HERE" />
```

```
</application>
```

**Create MyApp and MyMap Widgets:**

- Implement a simple Flutter app with a MyApp widget containing a MyMap widget.

#### MyMap Widget:

- Create a MyMap stateful widget with a GoogleMap widget.
- Use the onMapCreated callback to get the reference to the GoogleMapController.

#### GoogleMap Initialization:

- Initialize the GoogleMap widget with an initial camera position and default location.

#### Run the Application:

- Use the Flutter CLI or an IDE to run the application on an emulator or physical device.

**\$ flutter run**

#### Testing the Application:

- Interact with the application, and you should see a Google Map displayed on the screen with the specified initial camera position.

#### Dependencies Packages:

```
dependencies:
  flutter:
    sdk: flutter
  google_maps_flutter: ^2.2.8
```

#### Program :

##### main.dart

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

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

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

class MyMap extends StatefulWidget {
  @override
  _MyMapState createState() => _MyMapState();
}

class _MyMapState extends State<MyMap> {
  GoogleMapController? mapController;

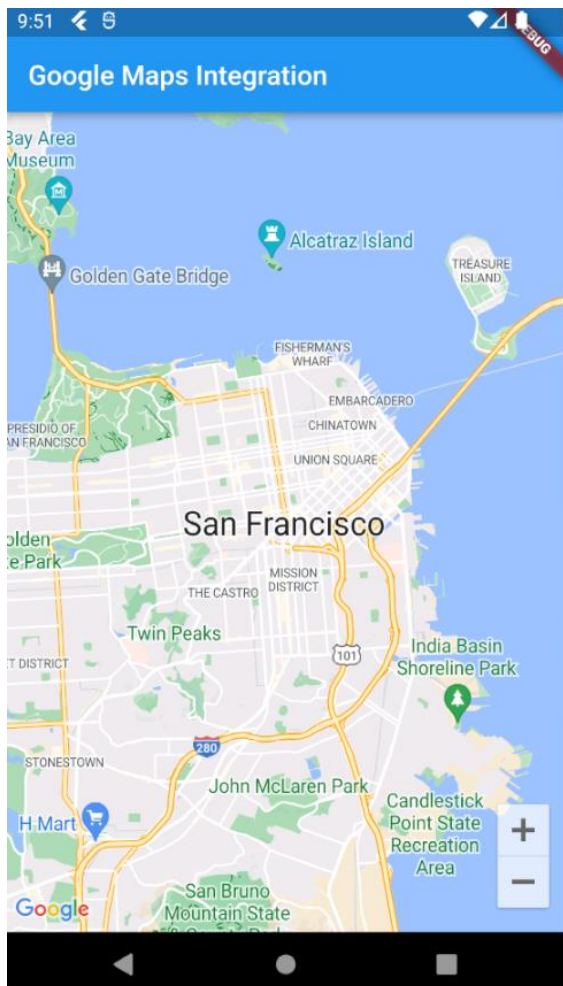
  @override
```

```

Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Google Maps Integration'),
    ),
    body: GoogleMap(
      onMapCreated: (controller) {
        setState(() {
          mapController = controller;
        });
      },
      initialCameraPosition: CameraPosition(
        target: LatLng(12.17, 79.04), // Default location (San Francisco)
        zoom: 12.0,
      ),
    ),
  );
}

```

**Output:**



**Result:**

Successfully Develop an application by integrating Google maps.



**Aim:**

To Develop Mini Projects involving Flutter/Kotlin multi-platform .

**Algorithm :****Define ToDo Class:**

- Create a class ToDo with attributes id, todoText, and isDone.
- Include a static method to generate a sample list of todos.

**Initialize State:**

- Create a stateful widget (Home) with a state class (\_HomeState).
- Initialize state variables such as todosList, \_foundToDo, and \_todoController.

**Build Main UI:**

- Implement the build method in \_HomeState to construct the main UI using Flutter widgets.
- Include a search box, a list view to display todos, and an input field for adding new todos.

**Handle ToDo Changes:**

- Implement a method to handle changes in todo status (\_handleToDoChange).
- Toggle the isDone property of the selected todo.

**Delete ToDo Item:**

- Implement a method to delete a todo item (\_deleteToDoItem).
- Remove the selected todo from the todosList.

**Add ToDo Item:**

- Implement a method to add a new todo item (\_addToDoItem).
- Create a new ToDo object and add it to the todosList.

**Run Filter:**

- Implement a method (\_runFilter) to filter todos based on the entered keyword.
- Update the \_foundToDo list with the filtered results.

**Search Box Widget:**

- Create a separate method (searchBox) to build the search box widget.

**ToDo Item Widget:**

- Create a separate stateless widget (ToDoItem) to display each todo item. Include checkboxes, todo text, and a delete button.

**AppBar Widget:**

- Implement a method (\_buildAppBar) to create the app bar with a menu icon and user avatar.

### Run Application:

- Run the application using flutter run.

### Test Application:

- Test the application by interacting with the UI, adding, checking, and deleting todos.

### Dependencies Packages:

```
dev_dependencies:s
  flutter_test:
    sdk: flutter
  flutter_lints: ^2.0.0
```

### Program :

Download Assets From [https://github.com/ramtsp/flutter\\_Assets/tree/main/Ex10-assets/images](https://github.com/ramtsp/flutter_Assets/tree/main/Ex10-assets/images)

#### main.dart

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

const Color tdRed = Color(0xFFDA4040);
const Color tdBlue = Color(0xFF5F52EE);
const Color tdBlack = Color(0xFF3A3A3A);
const Color tdGrey = Color(0xFF717171);
const Color tdBGColor = Color(0xFFEEEEFF5);

class ToDo {
  String? id;
  String? todoText;
  bool isDone;

  ToDo({
    required this.id,
    required this.todoText,
    this.isDone = false,
  });

  static List<ToDo> todoList() {
    return [
      ToDo(id: '01', todoText: 'Morning Excercise', isDone: true),
      ToDo(id: '02', todoText: 'Buy Groceries', isDone: true),
      ToDo(id: '03', todoText: 'Check Emails'),
    ];
  }
}
```

```

        ToDo(id: '04', todoText: 'Team Meeting'),
        ToDo(id: '05', todoText: 'Work on mobile apps for 2 hours'),
        ToDo(id: '06', todoText: 'Dinner with Jenny'),
    ];
}
}

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

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

    @override
    Widget build(BuildContext context) {
        SystemChrome.setSystemUIOverlayStyle(
            SystemUiOverlayStyle(statusBarColor: Colors.transparent));
        return MaterialApp(
            debugShowCheckedModeBanner: false,
            title: 'ToDo App',
            home: Home(),
        );
    }
}

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

    @override
    State<Home> createState() => _HomeState();
}

class _HomeState extends State<Home> {
    final todosList = ToDo.todoList();
    List<ToDo> _foundToDo = [];
    final _todoController = TextEditingController();

    @override
    void initState() {

```

```

    _foundToDo = todosList;
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: tdBGColor,
      appBar: _buildAppBar(),
      body: Stack(
        children: [
          Container(
            padding: EdgeInsets.symmetric(
              horizontal: 20,
              vertical: 15,
            ),
            child: Column(
              children: [
                searchBox(),
                Expanded(
                  child: ListView(
                    children: [
                      Container(
                        margin: EdgeInsets.only(
                          top: 50,
                          bottom: 20,
                        ),
                        child: Text(
                          'All Todos',
                          style: TextStyle(
                            fontSize: 30,
                            fontWeight: FontWeight.w500,
                          ),
                        ),
                      ),
                    ],
                  ),
                for (ToDo todo in _foundToDo.reversed)
                  ToDoItem(
                    todo: todo,
                    onToDoChanged: _handleToDoChange,
                    onDeleteItem: _deleteToDoItem,

```

```

        ),
      ],
    ),
  ),
],
),
),
Align(
  alignment: Alignment.bottomCenter,
  child: Row(children: [
    Expanded(
      child: Container(
        margin: EdgeInsets.only(
          bottom: 20,
          right: 20,
          left: 20,
        ),
        padding: EdgeInsets.symmetric(
          horizontal: 20,
          vertical: 5,
        ),
        decoration: BoxDecoration(
          color: Colors.white,
          boxShadow: const [
            BoxShadow(
              color: Colors.grey,
              offset: Offset(0.0, 0.0),
              blurRadius: 10.0,
              spreadRadius: 0.0,
            ),
          ],
          borderRadius: BorderRadius.circular(10),
        ),
        child: TextField(
          controller: _todoController,
          decoration: InputDecoration(
            hintText: 'Add a new todo item',
            border: InputBorder.none,
          ),
        ),
      ),
    ),
  ],
),

```

```

    ),
    Container(
      margin: EdgeInsets.only(
        bottom: 20,
        right: 20,
      ),
      child: ElevatedButton(
        child: Text(
          '+',
          style: TextStyle(
            fontSize: 40,
          ),
        ),
        onPressed: () {
          _addToDoItem(_todoController.text);
        },
        style: ElevatedButton.styleFrom(
          primary: tdBlue,
          minimumSize: Size(60, 60),
          elevation: 10,
        ),
      ),
    ),
  ],
),
],
);
}

void _handleToDoChange(ToDo todo) {
  setState(() {
    todo.isDone = !todo.isDone;
  });
}

void _deleteToDoItem(String id) {
  setState(() {
    todosList.removeWhere((item) => item.id == id);
  });
}

```

```

}

void _addToDoItem(String toDo) {
  setState(() {
    todosList.add(ToDo(
      id: DateTime.now().millisecondsSinceEpoch.toString(),
      todoText: toDo,
    ));
  });
  _todoController.clear();
}

void _runFilter(String enteredKeyword) {
  List<ToDo> results = [];
  if (enteredKeyword.isEmpty) {
    results = todosList;
  } else {
    results = todosList
      .where((item) => item.todoText!
        .toLowerCase()
        .contains(enteredKeyword.toLowerCase()))
      .toList();
  }

  setState(() {
    _foundToDo = results;
  });
}

Widget searchBox() {
  return Container(
    padding: EdgeInsets.symmetric(horizontal: 15),
    decoration: BoxDecoration(
      color: Colors.white,
      borderRadius: BorderRadius.circular(20),
    ),
    child: TextField(
      onChanged: (value) => _runFilter(value),
      decoration: InputDecoration(
        contentPadding: EdgeInsets.all(0),

```

```

        prefixIcon: Icon(
          Icons.search,
          color: tdBlack,
          size: 20,
        ),
        prefixIconConstraints: BoxConstraints(
          maxHeight: 20,
          minWidth: 25,
        ),
        border: InputBorder.none,
        hintText: 'Search',
        hintStyle: TextStyle(color: tdGrey),
      ),
    ),
  );
}

AppBar _buildAppBar() {
  return AppBar(
    backgroundColor: tdBGColor,
    elevation: 0,
    title: Row(mainAxisAlignment: MainAxisAlignment.spaceBetween, children: [
      Icon(
        Icons.menu,
        color: tdBlack,
        size: 30,
      ),
      Container(
        height: 40,
        width: 40,
        child: ClipRRect(
          borderRadius: BorderRadius.circular(20),
          child: Image.asset('assets/images/avatar.jpeg'),
        ),
      ),
    ]),
  );
}
}

```



```

class ToDoItem extends StatelessWidget {
  final ToDo todo;
  final onToDoChanged;
  final onDeleteItem;

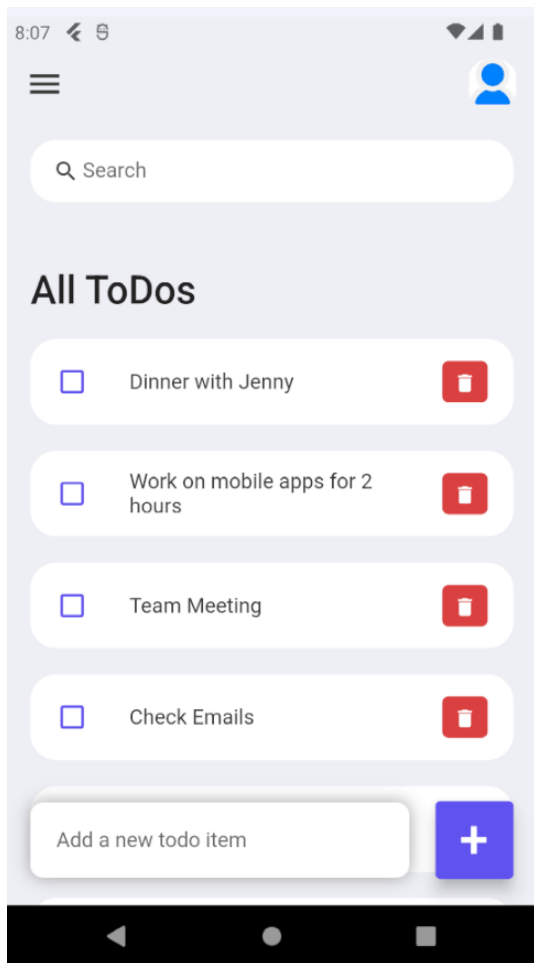
  const ToDoItem({
    Key? key,
    required this.todo,
    required this.onToDoChanged,
    required this.onDeleteItem,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      margin: EdgeInsets.only(bottom: 20),
      child: ListTile(
        onTap: () {
          onToDoChanged(todo);
        },
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(20),
        ),
        contentPadding: EdgeInsets.symmetric(horizontal: 20, vertical: 5),
        tileColor: Colors.white,
        leading: Icon(
          todo.isDone ? Icons.check_box : Icons.check_box_outline_blank,
          color: tdBlue,
        ),
        title: Text(
          todo.todoText!,
          style: TextStyle(
            fontSize: 16,
            color: tdBlack,
            decoration: todo.isDone ? TextDecoration.lineThrough : null,
          ),
        ),
        trailing: Container(
          padding: EdgeInsets.all(0),
          margin: EdgeInsets.symmetric(vertical: 12),

```

```
        height: 35,
        width: 35,
        decoration: BoxDecoration(
          color: tdRed,
          borderRadius: BorderRadius.circular(5),
        ),
        child: IconButton(
          color: Colors.white,
          iconSize: 18,
          icon: Icon(Icons.delete),
          onPressed: () {
            onDeleteItem(todo.id);
          },
        ),
      ),
    ),
  );
}
```

### Output:



### Result:

Successfully Develop Mini Projects involving Flutter/Kotlin multi-platform.