

## EXPERIMENT NO 5

**Aim:** To include icons,images,charts in flutter app.

### Theory:

Flutter is an open-source UI framework developed by Google used to build fast, natively compiled applications for mobile, web, and desktop from a single codebase. It uses the Dart programming language and follows a widget-based architecture, where everything in the UI is a widget, making layouts highly customizable and flexible.

Flutter renders its own UI using the Skia graphics engine, which ensures high performance and a consistent look across platforms. Features like hot reload allow developers to see changes instantly, improving productivity, while rich built-in widgets and strong community support make Flutter a popular choice for modern app development.



### Key Widgets Used in Flutter Forms

#### 1. Form Widget

- Acts as a container for form fields
- Groups multiple input widgets together
- Helps in managing validation and submission

### Asset bundling

The assets subsection of the flutter section specifies files that should be included with the app. Each asset is identified by an explicit path (relative to the pubspec.yaml file) where the asset file is located. The order in which the assets are declared doesn't matter. The actual

directory name used (assets in first example or directory in the above example) doesn't matter.

During a build, Flutter places assets into a special archive called the *asset bundle* that apps read from at runtime.

Automatic transformation of asset files at build time

Flutter supports using a Dart package to transform asset files when building your app. To do this, specify the asset files and transformer package in your pubspec file. To learn how to do this and write your own asset-transforming packages, see [Transforming assets at build time](#).

## Loading assets

Your app can access its assets through an [AssetBundle](#) object.

The two main methods on an asset bundle allow you to load a string/text asset (loadString()) or an image/binary asset (load()) out of the bundle, given a logical key. The logical key maps to the path to the asset specified in the pubspec.yaml file at build time.

## Loading text assets

Each Flutter app has a [rootBundle](#) object for easy access to the main asset bundle. It is possible to load assets directly using the rootBundle global static from package:flutter/services.dart.

However, it's recommended to obtain the AssetBundle for the current BuildContext using [DefaultAssetBundle](#), rather than the default asset bundle that was built with the app; this approach enables a parent widget to substitute a different AssetBundle at run time, which can be useful for localization or testing scenarios.

Typically, you'll use DefaultAssetBundle.of() to indirectly load an asset, for example a JSON file, from the app's runtime rootBundle.

Outside of a Widget context, or when a handle to an AssetBundle is not available, you can use rootBundle to directly load such assets. For example:

```
import 'package:flutter/services.dart' show rootBundle;
```

```
Future<String> loadAsset() async {  
  return await rootBundle.loadString('assets/config.json');  
}
```

## Loading images

To load an image, use the [AssetImage](#) class in a widget's build() method.

For example, your app can load the background image from the asset declarations in the previous example:

```
return const Image(image: AssetImage('assets/background.png'));
```

Resolution-aware image assets

Flutter can load resolution-appropriate images for the current [device pixel ratio](#).

[AssetImage](#) will map a logical requested asset onto one that most closely matches the current [device pixel ratio](#).

For this mapping to work, assets should be arranged according to a particular directory structure:

.../image.png

.../Mx/image.png

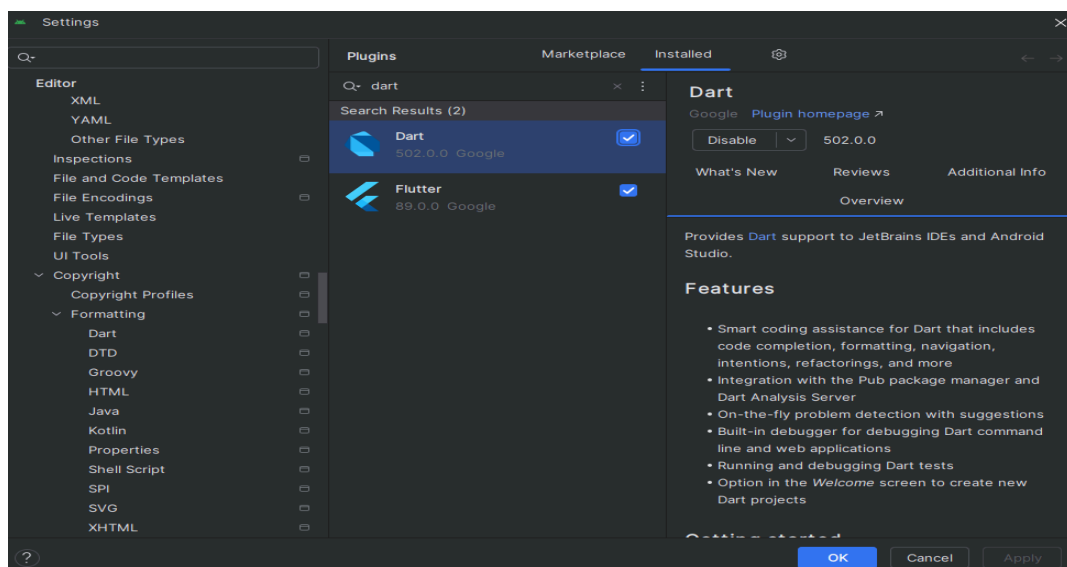
.../Nx/image.png

...etc.

## Procedures:

### STEP 1: Install Flutter & Dart Plugins in Android Studio

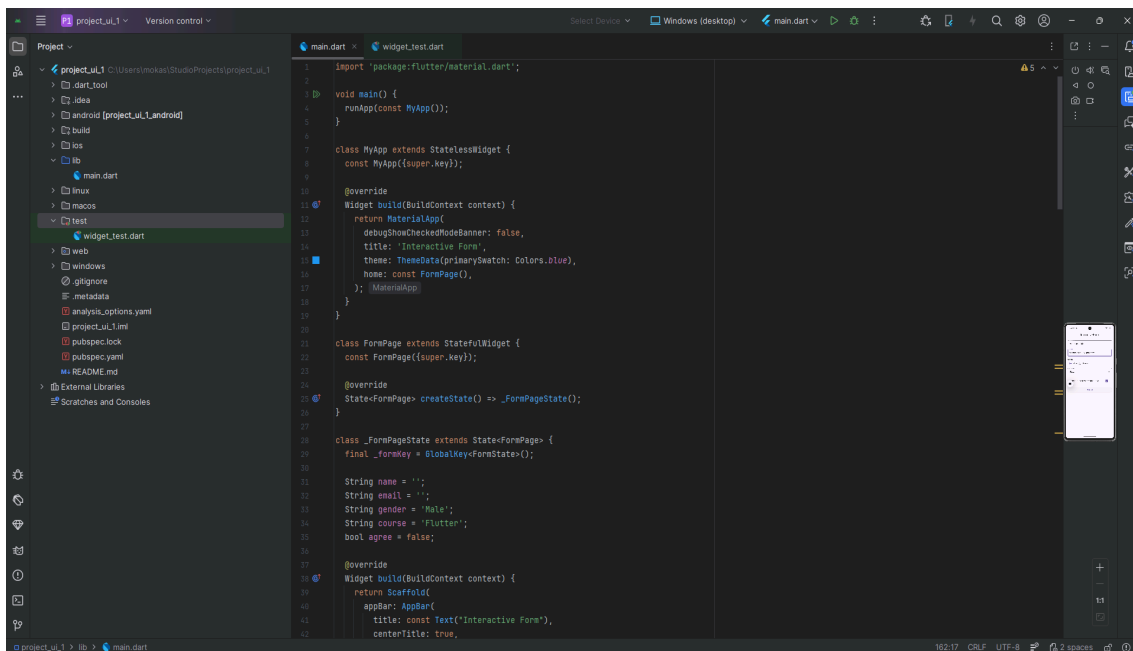
1. Open Android Studio
2. Go to Settings → Plugins
3. Search and Install- Flutter & Dart
4. Restart android studio



### STEP 2: Create a New Flutter Project

1. Open Android Studio

2. Click New Flutter Project
3. Select Flutter Application
4. Choose Flutter SDK path (example):  
C:\flutter
5. Enter:
  - o Project name: project\_ui\_1
  - o Language: Dart
6. Click Finish



### STEP 3: Code

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

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

```
class MyApp extends StatelessWidget {  
  const MyApp({super.key});  
  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      debugShowCheckedModeBanner: false,  
      home: const AnalyticsDashboard(),  
    );  
  }  
}
```

```
class AnalyticsDashboard extends StatelessWidget {  
  const AnalyticsDashboard({super.key});  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      backgroundColor: const Color(0xffff5f6fa),  
      appBar: AppBar(  
        title: const Text("Analytics"),  
        centerTitle: true,  
      ),  
      body: SingleChildScrollView(  
        child: Column(  
          children: [  
  
            /// HEADER IMAGE  
  
            Stack(  

```

```

children: [
  Image.network(
    "https://images.unsplash.com/photo-1500530855697-b586d89ba3ee",
    height: 180,
    width: double.infinity,
    fit: BoxFit.cover,
  ),
  Container(
    height: 180,
    color: Colors.black.withOpacity(0.4),
  ),
  const Positioned(
    left: 20,
    bottom: 20,
    child: Text(
      "Weekly Overview",
      style: TextStyle(
        color: Colors.white,
        fontSize: 22,
        fontWeight: FontWeight.bold),
    ),
  ),
],
),

const SizedBox(height: 20),

/// ICON STATS
Padding(

```

```

padding: const EdgeInsets.symmetric(horizontal: 16),
child: Row(
  children: [
    _statBox(Icons.shopping_bag, "Orders", "320"),
    _statBox(Icons.people, "Users", "1.2K"),
  ],
),
),

const SizedBox(height: 20),

/// LINE GRAPH
_chartCard(
  title: "Sales Trend",
  icon: Icons.show_chart,
  chart: LineChart(_lineChartData()),
),

const SizedBox(height: 20),

/// BAR GRAPH
_chartCard(
  title: "Monthly Performance",
  icon: Icons.bar_chart,
  chart: BarChart(_barChartData()),
),
],
),
),

```

```

/// BOTTOM NAV

bottomNavigationBar: BottomNavigationBar(
  currentIndex: 0,
  items: const [
    BottomNavigationBarItem(icon: Icon(Icons.home), label: "Home"),
    BottomNavigationBarItem(icon: Icon(Icons.analytics), label: "Stats"),
    BottomNavigationBarItem(icon: Icon(Icons.person), label: "Profile"),
  ],
),
);
}

/// STAT BOX

Widget _statBox(IconData icon, String title, String value) {
  return Expanded(
    child: Card(
      margin: const EdgeInsets.symmetric(horizontal: 8),
      shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(16)),
      child: Padding(
        padding: const EdgeInsets.all(16),
        child: Column(
          children: [
            Icon(icon, size: 32, color: Colors.blue),
            const SizedBox(height: 8),
            Text(title),
            Text(
              value,
              style: const TextStyle(

```



```

        fontSize: 18, fontWeight: FontWeight.bold),
      ),
    ],
  ),
),
),
);
}

```

/// CHART CARD

```

Widget _chartCard({
  required String title,
  required IconData icon,
  required Widget chart,
}) {
  return Padding(
    padding: const EdgeInsets.symmetric(horizontal: 16),
    child: Card(
      shape: RoundedRectangleBorder(borderRadius: BorderRadius.circular(16)),
      child: Padding(
        padding: const EdgeInsets.all(16),
        child: Column(
          children: [
            Row(
              children: [
                Icon(icon, color: Colors.blue),
                const SizedBox(width: 8),
                Text(title,
                  style: const TextStyle(

```

```

        fontSize: 16, fontWeight: FontWeight.bold)),
    ],
  ),
  const SizedBox(height: 20),
  SizedBox(height: 200, child: chart),
],
),
),
),
);
}

```

/// LINE CHART DATA

```

LineChartData _lineChartData() {
  return LineChartData(
    gridData: FlGridData(show: true),
    borderData: FlBorderData(show: false),
    titlesData: FlTitlesData(show: true),
    lineBarsData: [
      LineChartBarData(
        isCurved: true,
        spots: [
          FlSpot(0, 1),
          FlSpot(1, 3),
          FlSpot(2, 2),
          FlSpot(3, 4),
          FlSpot(4, 3),
        ],
        barWidth: 4,

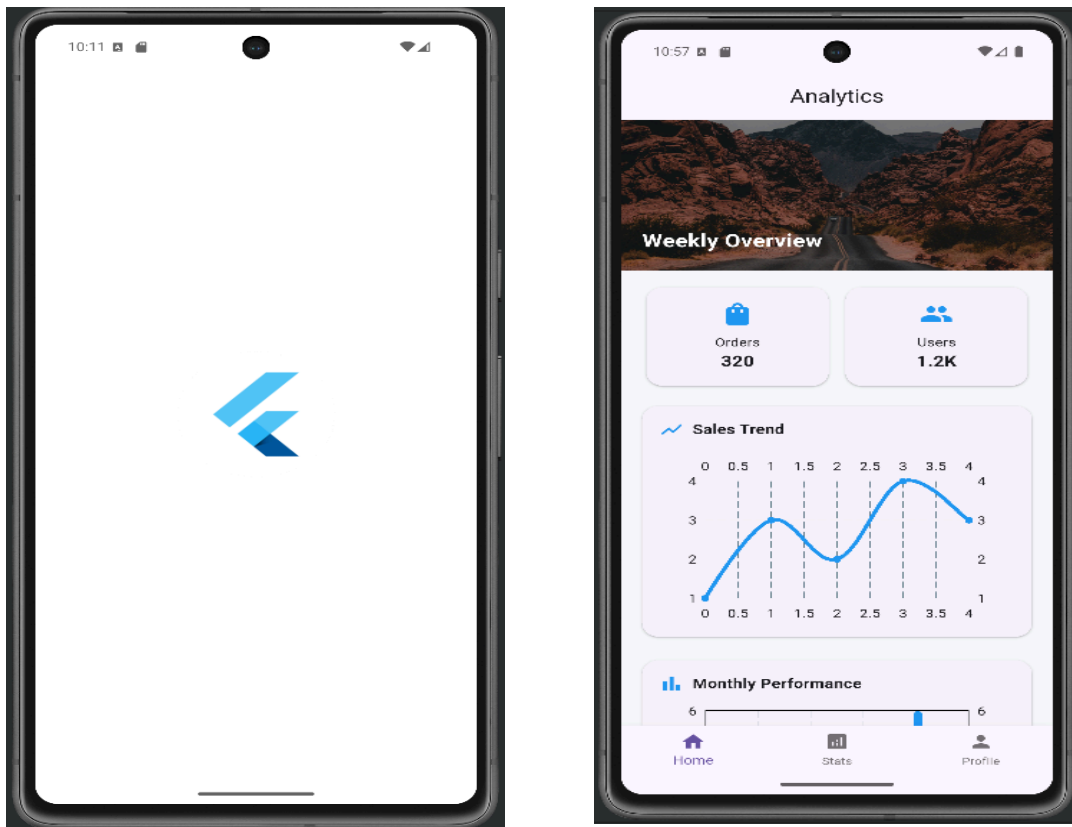
```

```
        color: Colors.blue,  
        dotData: FlDotData(show: true),  
    )  
  ],  
);  
}
```

/// BAR CHART DATA

```
BarChartData _barChartData() {  
  return BarChartData(  
    barGroups: [  
      BarChartGroupData(x: 0, barRods: [  
        BarChartRodData(toY: 5, color: Colors.blue)  
      ]),  
      BarChartGroupData(x: 1, barRods: [  
        BarChartRodData(toY: 3, color: Colors.blue)  
      ]),  
      BarChartGroupData(x: 2, barRods: [  
        BarChartRodData(toY: 4, color: Colors.blue)  
      ]),  
      BarChartGroupData(x: 3, barRods: [  
        BarChartRodData(toY: 6, color: Colors.blue)  
      ]),  
    ],  
  );  
}
```

## STEP 6: Output



## Conclusion:

Flutter provides a powerful and flexible UI framework to build visually rich and interactive applications by effectively integrating icons, images, and graphs. Icons enhance usability by clearly representing actions and features, while images improve user engagement and visual appeal. Graphs and charts allow developers to present complex data in a clear, analytical, and user-friendly manner, making Flutter suitable for dashboards, analytics panels, and data-driven applications. By combining these elements with Flutter's widget-based architecture and third-party libraries like fl chart, developers can create modern, responsive, and professional-quality mobile applications efficiently.