AIM : To design Flutter UI by including common widgets.

THEORY :
Flutter widgets are the building blocks of Flutter applications, allowing developers to create user interfaces by assembling and combining various widgets. Widgets in Flutter are objects that represent parts of the user interface, such as buttons, text, images, layout structures, and more. They are the primary elements used to compose the visual and interactive components of a Flutter app. Flutter widgets empower developers to craft dynamic and expressive user interfaces effortlessly. From foundational elements like Container and Text to complex structures like Row, Column, and Stack, Flutter's widget-based architecture offers a rich toolkit for building aesthetically pleasing and responsive mobile applications.
Widgets in Flutter are divided into two main categories:

1. Stateless Widgets:
   Stateless widgets are immutable, meaning their properties (also known as parameters) cannot change once they are constructed. They are used to represent UI components that don't change over time (i.e., they don't have any internal state). Examples include Container, Text, Icon, Image, Row, Column, etc.

2. Stateful Widgets:
   Stateful widgets are mutable and maintain state that might change during the lifetime of the widget. They are used to represent UI components that can change in response to user actions, data changes, etc. Examples include TextField, Checkbox, Radio, Slider, ListView, GridView, etc.

Widgets in Flutter follow a hierarchical structure, where widgets can be nested inside other widgets to build complex UI layouts. This hierarchy forms a widget tree, with a single root widget at the top.

Code:

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

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

class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
   return MaterialApp(
     title: 'ERP Application',
     theme: ThemeData(
```

```
      primarySwatch: Colors.blue,
      visualDensity: VisualDensity.adaptivePlatformDensity,
    ),
    home: MyHomePage(),
  );
 }
}

class MyHomePage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     appBar: AppBar(
       title: Text('ERP Dashboard'),
     ),
     body: Padding(
       padding: const EdgeInsets.all(16.0),
       child: Column(
         crossAxisAlignment: CrossAxisAlignment.start,
         children: [
           Text(
             'Sales Overview',
             style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
           ),
           SizedBox(height: 16),
           Row(
             mainAxisAlignment: MainAxisAlignment.spaceBetween,
             children: [
               _buildStatCard('Total Sales', '\$500,000'),
               _buildStatCard('Orders', '1,200'),
               _buildStatCard('Customers', '800'),
             ],
           ),
           SizedBox(height: 32),
           Text(
             'Inventory Management',
             style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
           ),
           SizedBox(height: 16),
           Row(
             mainAxisAlignment: MainAxisAlignment.spaceBetween,
             children: [
               _buildStatCard('Total Products', '2,500'),
               _buildStatCard('Stock on Hand', '5,000'),
               _buildStatCard('Low Stock Alerts', '50'),
             ],
           ),
         ],
       ),
```

```
    ),
  );
 }

 Widget _buildStatCard(String title, String value) {
   return Card(
     elevation: 3,
     child: Padding(
       padding: const EdgeInsets.all(16.0),
       child: Column(
         crossAxisAlignment: CrossAxisAlignment.start,
         children: [
           Text(
             title,
             style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
           ),
           SizedBox(height: 8),
           Text(
             value,
             style: TextStyle(fontSize: 16),
           ),
         ],
       ),
     ),
   );
 }
}
```

Widgets used in above code:

1. Text: Displays text on the screen. It's used to show headings such as "Sales Overview" and "Inventory Management".

2. AppBar: Represents the top app bar containing a title (Text widget) indicating the current page or screen.

3. Padding: Used to add padding around its child widget. In this case, it adds padding around the Column widget to create space between its children.

4. Column: Arranges its children vertically in a single column. It's used to organize the content vertically in the body of the app.

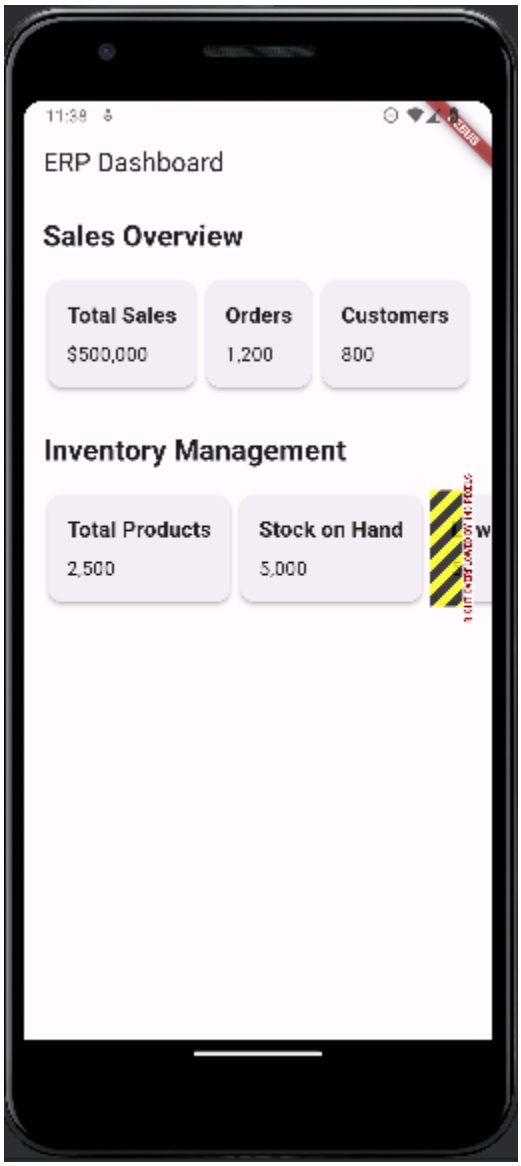
5. SizedBox: Creates an empty box with a specified height. It's used to add space between widgets vertically.

6. Row: Arranges its children horizontally in a single row. It's used to display multiple widgets side by side.

7. Card: Represents a material design card. It's used to display each statistic with a consistent elevation and styling.

8. ElevatedButton: A button widget that responds to touches by elevating, and it's typically used for actions. In this case, it's not used directly in the provided code, but it's a widget that could be used elsewhere in the app.

OUTPUT:



Conclusion:

In this experiment, I explored the usage and behavior of various Flutter widgets in the context of building a simple ERP (Enterprise Resource Planning) dashboard. I understood that Flutter widgets can be effectively composed together to create complex user interfaces. The flexibility provided by Flutter widgets allows for the creation of

versatile layouts that adapt to different screen sizes and orientations. Overall, I got a sight of the power and versatility of Flutter widgets in building rich and interactive user interfaces.