Experiment 02 : To design Flutter UI by including common widgets.

Name: Anurag Gaiwal Roll No. 17
Div: D15A Batch: A

SAMPLE CODE:main.dart

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


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


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


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


  @override
```

```
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.white24,
    appBar: AppBar(
      centerTitle: true,
      backgroundColor: Colors.white24,
      leading: IconButton(
        onPressed: () {
          Navigator.pop(context);
        },
        icon: Icon(Icons.arrow_back),
      ),
      title: Text(
        "Settings",
        style: TextStyle(color: Colors.white),
      ),
    ),
    body: SafeArea(
      child: Padding(
        padding: EdgeInsets.fromLTRB(10, 20, 10, 0),
        child: SingleChildScrollView(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
```

```
_buildSectionHeader("PROFILE"),
SizedBox(height: 2),
_buildProfileButton(
  label: "Anurag Gaiwal D15A\n VESIT \t\t \t\t",
  onPressed: () {
    print("You pressed Profile Button");
  },
),
SizedBox(height: 10),
_buildSectionHeader("FEATURES"),
SizedBox(height: 10),
_buildFeatureButton(
  label: "Memories \t \t \t \t \t \t \t \t \t \t \t",
  icon: Icons.calendar_today,
  onPressed: () {
    print("You pressed Memories Button");
  },
),
SizedBox(height: 2),
_buildFeatureButton(
  label: "Blocked Profile",
  icon: Icons.block,
  onPressed: () {
    print("You pressed Blocked Profile Button");
```

```
  },
),
SizedBox(height: 10),
_buildSectionHeader("SETTINGS"),
SizedBox(height: 10),
_buildFeatureButton(
  label: "Notifications \t \t",
  icon: Icons.notifications,
  onPressed: () {
    print("You pressed Notifications Button");
  },
),
SizedBox(height: 2),
_buildFeatureButton(
  label: "Time Zone \t \t \t \t \t \t \t \t \t   ",
  icon: Icons.access_time,
  onPressed: () {
    print("You pressed Time Zone Button");
  },
),
SizedBox(height: 2),
_buildFeatureButton(
  label: "Others \t \t\t\t\t\t\t\t\t\t",
  icon: Icons.settings_suggest,
```

```
  onPressed: () {
    print("You pressed Others Button");
  },
),
SizedBox(height: 10),
_buildSectionHeader("ABOUT"),
SizedBox(height: 10),
_buildFeatureButton(
  label: "Share BeReal\t \t \t",
  icon: Icons.share,
  onPressed: () {
    print("You pressed Share BeReal Button");
  },
),
SizedBox(height: 2),
_buildFeatureButton(
  label: "Rate \t\t\t\t \t \t  \t  ",
  icon: Icons.star_outline,
  onPressed: () {
    print("You pressed Rate Button");
  },
),
SizedBox(height: 2),
_buildFeatureButton(
```

```
                  label: "Help \t \t \t \t \t \t \t \t \t \t  \t \t \t
",
                  icon: Icons.help_outline,
                  onPressed: () {
                    print("You pressed Help Button");
                  },
                ),
                SizedBox(height: 2),
                _buildFeatureButton(
                  label: "About \t \t\t\t\t\t\t\t\t\t",
                  icon: Icons.info,
                  onPressed: () {
                    print("You pressed About Button");
                  },
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }

  Widget _buildSectionHeader(String title) {
    return Row(
```

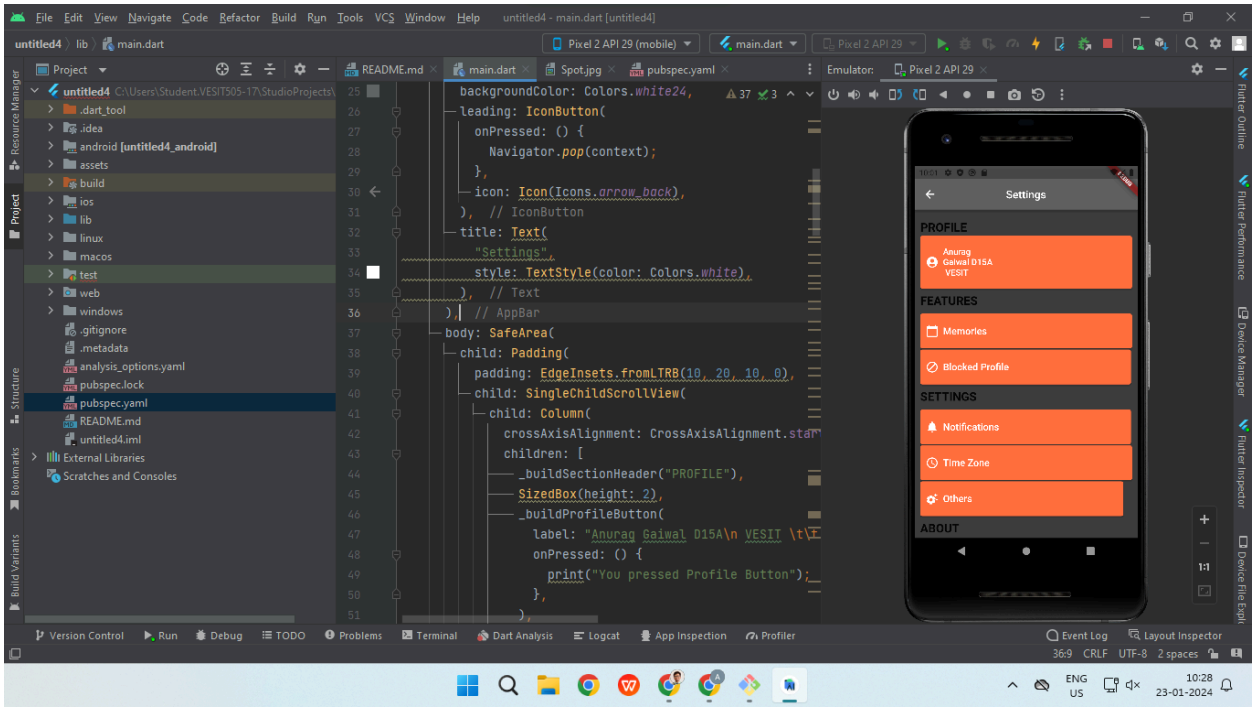
```
      children: [
        Text(
          title,
          style: TextStyle(fontSize: 22, fontWeight: FontWeight.bold),
        ),
        VerticalDivider(
          color: Colors.transparent,
          thickness: 1,
          width: 10, // Adjust the width as needed
        ),
      ],
    );
  }


  Widget _buildProfileButton({required String label, required VoidCallback
onPressed}) {
    return Align(
      alignment: Alignment.topLeft,
      child: ElevatedButton.icon(
        onPressed: onPressed,
        icon: Icon(
          Icons.account_circle,
          color: Colors.white,
        ),
        label: Text(
```

```
          label,
          style: TextStyle(color: Colors.white, fontSize: 16),
        ),
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.deepOrangeAccent,
          padding: EdgeInsets.fromLTRB(10, 20, 247, 20),
        ),
      ),
    );
  }


  Widget _buildFeatureButton({required String label, required IconData
icon, required VoidCallback onPressed}) {
    return Align(
      alignment: Alignment.topLeft,
      child: ElevatedButton.icon(
        onPressed: onPressed,
        icon: Icon(
          icon,
          color: Colors.white,
        ),
        label: Text(
          label,
          style: TextStyle(color: Colors.white, fontSize: 18),
        ),
```

```
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.deepOrangeAccent,
          padding: EdgeInsets.fromLTRB(10, 20, 225, 20),
        ),
      ),
    );
  }
}
```



Explanation

`main.dart` File:

- The `main.dart` file is the entry point of your Flutter application.
- It includes the `main` function, which calls `runApp` to start the Flutter application.
- The `runApp` function takes an instance of the `MyApp` widget as an argument.

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

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

class AccountApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
 return MaterialApp(
 home: Account(),
 );
 }
}
```

`Account` Class:

- The `Account` class is a `StatelessWidget` representing the main screen of your application.
- It includes the `Scaffold` widget, which provides the basic structure for the visual interface.
- The app bar, body, and other UI elements are defined within the `Scaffold`.

dart

Copy code

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

 @override
 Widget build(BuildContext context) {
 // ... (rest of the code)
 }
}
```

UI Elements in `Account` Class:

- The code within the `build` method defines the layout and appearance of the screen.
- Sections like "Profile," "FEATURES," "SETTINGS," and "ABOUT" are created using the `_buildSectionHeader` function.
- Buttons are created using the `_buildProfileButton` and `_buildFeatureButton` functions.
- Each button has an `onPressed` callback, which is a function that gets executed when the button is pressed.

dart

Copy code

```
Widget _buildSectionHeader(String title) {
 // ... (code to build section header)
 }

 Widget _buildProfileButton({required String label, required VoidCallback
onPressed}) {
 // ... (code to build profile button)
 }

 Widget _buildFeatureButton({required String label, required IconData icon,
required VoidCallback onPressed}) {
 // ... (code to build feature button)
 }
```

Usage of Widgets:

- Widgets like `Row`, `Column`, `SizedBox`, `ElevatedButton`, and `Align` are used to structure the UI.
- The `SafeArea` widget ensures that content is displayed within the safe area of the screen.
- The `SingleChildScrollView` widget allows scrolling when the content overflows the screen.

List of Widgets

- Flutter Scaffold
- Flutter Container
- Flutter Row & Column
- Flutter Text
- Flutter TextField
- Flutter Buttons
- Flutter Stack
- Flutter Forms
- Flutter AlertDialog
- Flutter Icons
- Flutter Images
- Flutter Card
- Flutter Tabbar
- Flutter Drawer
- Flutter Lists
- Flutter GridView
- Flutter Toast
- Flutter Checkbox
- Flutter Radio Button
- Flutter Progress Bar
- Flutter Snackbar
- Flutter Tooltip
- Flutter Slider
- Flutter Switch
- Flutter Charts
- Bottom Navigation Bar

Flutter Themes

Flutter Table

Flutter Calendar

Flutter Animation

Common widgets

1. Column
2. Row
3. Stack
4. Container
5. List View

Reference
https://docs.flutter.dev/cookbook
https://www.javatpoint.com/flutter
Youtube Reference :
//common 5 widgets
https://youtu.be/-z26yE9g0Hg
// adding google fonts
https://www.youtube.com/watch?v=km2P_KQJyO0

Aim
Theory
Syntax
Widget and properties
Code and output

//Row and Column alignment

Aligning widgets

You control how a row or column aligns its children using the `mainAxisAlignment` and `crossAxisAlignment` properties. For a row, the main axis runs horizontally and the cross axis runs vertically. For a column, the main axis runs vertically and the cross axis runs horizontally.