MOBILE APPLICATION

What is Flutter?

Flutter is a mobile application development framework that allows developers to create high-performance, beautiful, and responsive mobile applications for multiple platforms using a single codebase

What is Dart, and why is it used in Flutter?

Dart is a programming language developed by Google. It's often used in the context of Flutter, which is a UI toolkit created by Google to build natively compiled applications for mobile, web, and desktop from a single codebase.

why is it used in Flutter?

- Optimized for UI
- Dart is a modern programming language that is easy to learn and use.
- Ahead-of-Time (AOT) Compilation
- Hot Reload
- Single Codebase
- Comprehensive Library Support

What are the different types of widgets in Flutter?

- ✓ **Stateless widgets:** These widgets are immutable, meaning they cannot be changed once they are created.
- ✓ **Stateful widgets:** These widgets are mutable and can be updated dynamically based on user interactions or other events.
- ✓ **What is the difference between StatelessWidget and StatefulWidget in Flutter?**
- ✓ In Flutter, StatelessWidget and StatefulWidget are two different types of widgets that are used to create UI elements.

StatelessWidget	**StatefulWidget**
StatelessWidget is a widget that does not have any mutable state.	StatefulWidget is a widget that has a mutable state.
Once it is built, its properties cannot be changed.	It can change its properties based on user interactions.
Include Text, Icons, and Images.	Include Text Field, Checkbox, and Slider.

What is State in Flutter?

In Flutter, state refers to the data or information that can change dynamically during the lifetime of a widget.

What is the purpose of setState() in Flutter?

setState() is a method that is used to update the state of a Stateful widget in Flutter. When a widget is updated, the Flutter framework calls the build() method of the widget to rebuild its UI based on the new state.

What is the purpose of the MaterialApp widget in Flutter?
The MaterialApp widget is used to create a Material Design application in Flutter. It provides pre-built widgets that follow the Material Design specifications, such as AppBar, BottomNavigationBar, and FloatingActionButton.

What is the purpose of the Scaffold widget in Flutter?
The Scaffold widget is used to create the basic structure of a Material Design application in Flutter. It provides a top app bar, a bottom navigation bar, and a body widget.

What is the purpose of the BuildContext parameter in the build() method of a widget in Flutter?
The BuildContext parameter is used to access the widget tree and perform actions such as building or updating widgets.

SUMMARY

Categories of widget based on their features

- Platform specific widget e.g. Android specific widget (Material Widget) and IOS specific widget (Cupertino widget)
  Some Material Widget includes: -
  - ✓ Scaffold
  - ✓ AppBar
  - ✓ BottomNavigationBar
  - ✓ TabBar
  - ✓ TabBarView
  - ✓ ListTile
  - ✓ RaisedButton
  - ✓ FloatingActionButton
  - ✓ FlatButton

- **Layout widget:** - a layout widget is used to structure and position other widgets on the screen.
  Some of the popular layout widgets are as follows:

  - ✓ **Column**: Lays out its children in a vertical direction.
  - ✓ **Row**: Lays out its children in a horizontal direction.
  - ✓ **Stack**: Allows widgets to overlap each other.
  - ✓ **Container**: A versatile widget that can be used for styling, positioning, and sizing its child.
  - ✓ **Padding**: Adds padding around a child widget.
  - ✓ **Center**: Centers its child within itself.
  - ✓ **Align**: Aligns its child within itself using a specified alignment.

- **State maintenance widget**: - all widgets are either derived from StatelessWidget or StatefulWidget

- **Platform independent**
  Flutter provides large number of basic widgets to create simple as well as complex user interface in a platform independent manner.
  Examples.

1. Container

The Container widget is a versatile widget that can be used to hold single child widgets and add padding, margins, borders, and background colors.

Properties of container

- **alignment**: Aligns the child widget within the container.
- **padding:** Adds space inside the container around its child.
- **margin:** Adds space outside the container around its child.
- **decoration:** Decorates the container (e.g., border, background color).
- **height and width**: Sets the dimensions of the container.

2. Row

The Row widget arranges its children widgets in a horizontal array.

Properties of row

- **mainAxisAlignment:** Aligns the children horizontally.
- **crossAxisAlignment:** Aligns the children vertically.
- **children:** A list of child widgets to be displayed in the row.

3. Column

The Column widget arranges its children widgets in a vertical array.

Properties of Column

- **mainAxisAlignment**: Aligns the children vertically.
- **crossAxisAlignmen**t: Aligns the children horizontally.
- **children:** A list of child widgets to be displayed in the column.

4. Text

The Text widget displays a string of text with a single style.

Properties of Text

- **data**: The string to display.
- **style:** Defines the text's style (e.g., font size, color).
- **textAlign**: Aligns the text within its bounds.
- **overflow**: How visual overflow should be handled (e.g., ellipsis).

5. Image

The Image widget displays an image.

Properties of Image

- image: The image to display (e.g., NetworkImage, AssetImage).
- width and height: Sets the dimensions of the image.
- fit: How the image should be inscribed into the space allocated (e.g., BoxFit.cover).

6. Icon

The Icon widget displays a graphic icon from a font family.

Properties of Icon

- **icon**: The icon to display (e.g., Icons.add).
- **size:** Sets the size of the icon.
- **color**: Sets the color of the icon.

7. Scaffold

The Scaffold widget provides the basic material design visual layout structure.

Properties of Scaffold

- **appBar**: Displays a horizontal bar typically shown at the top of the app.
- **body**: The primary content of the scaffold.
- **floatingActionButton**: A button that typically floats over the body.
- **drawer:** A panel that slides in from the side.

8. ListView

The **ListView** widget arranges its children in a scrollable linear array.

Properties of ListView

- **children**: A list of widgets to be displayed in the list.
- **scrollDirection:** Specifies the direction in which the list scrolls (Axis.horizontal or Axis.vertical).

What padding property does in mobile application

The padding property is used to control the space between the content of a widget and its border or boundary. Here are five key points about the padding property in mobile applications:

1. **Adds Space Inside a Widget**: Padding creates space inside a widget, between the content and the widget's boundary, making the content more visually appealing and easier to read.
2. **Customizable for All Sides**: Padding can be applied uniformly to all sides or customized individually for each side (top, bottom, left, right), providing flexibility in design.
3. **Improves Touch Target Size**: In interactive elements, padding can increase the touch target size, enhancing the user experience by making it easier to interact with buttons or other touchable elements.
4. **Controls Layout and Spacing**: Padding helps in controlling the overall layout and spacing of the UI elements, ensuring a consistent and organized appearance across the application.
5. **Enhances Visual Hierarchy**: By adding padding, developers can create visual separation between different elements, helping users to distinguish and focus on specific parts of the UI

Note:

- **Scaffold**: A top-level container that provides a structure for the visual interface, including an app bar, body, floating action button, drawer, and bottom navigation bar.
- **AppBar**: A horizontal bar typically displayed at the top of the app that contains the title, actions, and navigation elements.
- **BottomNavigationBar**: A widget displayed at the bottom of the app that allows users to navigate between different sections or pages.
- **TabBar**: A widget that displays a horizontal row of tabs, each associated with a different view.
- **TabBarView**: A widget that displays the views associated with the tabs in a TabBar.
- **ListTile**: A widget that represents a single row in a list and typically contains a title, subtitle, leading icon, and trailing icon.
- **RaisedButton**: A material design button that elevates when pressed and is used to trigger actions.
- **FloatingActionButton**: A circular button that floats above the content and is used for primary actions in the app.
- **FlatButton**: A material design button that does not have an elevation and is used for less prominent actions.

Type of Layout Widgets

1. Single child layout widget
2. Multiple child layout widget

Single Child Widget

In this category, widgets will have only one widget as its child and every widget will have a special layout functionality.

- **Padding**: Used to arrange its child widget by the given padding. Here, padding can be provided by EdgeInsets class.
- **Align**: Align its child widget within itself using the value of alignment property.
- **SizedBox:**
- **Center**
- **Container**
- **SizedBox**

Multiple Child Widgets

In this category, a given widget will have more than one child widgets and the layout of each widget is unique.

- **Row** - Allows to arrange its children in a horizontal manner.
- **Column** - Allows to arrange its children in a vertical manner.
- **ListView** - Allows to arrange its children as list.
- **GridView** - Allows to arrange its children as gallery.
- **Expanded -** Used to make the children of Row and Column widget to occupy the maximum possible area

Animation:

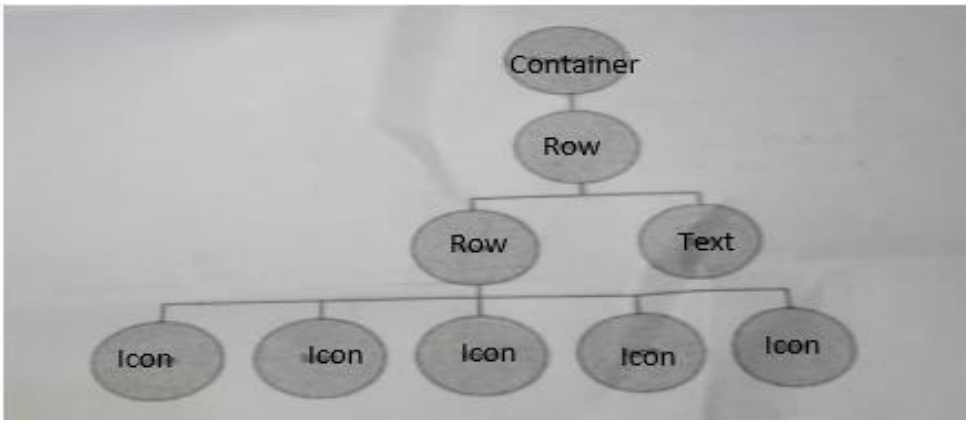
Animation is a process of showing a series of images / picture in a particular order within a specific duration to give an illusion of movement

Animation have two distinct values: Start value and End value.

SAMPLE CODING QUESTIONS

Question 03:

Write a program to create a mobile application with its widgets arranged as shown in the tree diagram below.




```dart
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Widget Arrangement Example'),
        ),
        body: Container(
          child: Row(
            children: [
              const Row(
                children: [
                  Icon(Icons.star),
                  Icon(Icons.star),
                  Icon(Icons.star),
                  Icon(Icons.star),
                ],
              ),
              Text('Example Text'),
            ],
          ),
        ),
      ),
    );
  }
}

```

QUESTION TWO

Write a program to add a drawer which is populated with items in mobile application interface by using the following steps

1. Create a Scaffold.
2. Add a drawer.
3. Populate the drawer with items.
4. Close the drawer programmatically. *[8 marks]*

```
Scaffold(
        appBar: AppBar(
          title: Text('Drawer Navigation'),
        ),
      body: Text('Drawer Navigation'),
      drawer: Drawer(child: ListView(
        children: [
          Text('item 1'),
          Text('item 2'),
          Text('item 3')
        ],
      ),
    ),
        endDrawer: const Drawer(),
    ),
    );
```

QUESTION TWO

Write a program to create a login interface as shown in the figure below which you can write email address and password for login process, also you can be able to show or hide a password.

[8 marks]

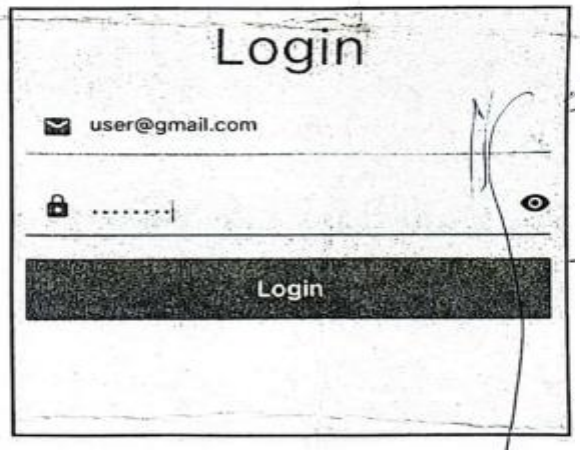
```dart
import 'package:flutter/material.dart';

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

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

  @override
  State<HamzApp> createState() => _HamzApp();
}

class _HamzApp extends State<HamzApp> {

  bool show = true;


  @override

  Widget build(BuildContext context){
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: const Text("Login",style: TextStyle(
            fontSize: 28,
          ),),
          centerTitle: true,
        ),

        body:Center(
          child: Column(
            children: [
              const TextField(
                decoration: InputDecoration(
                  prefixIcon: Icon(Icons.email),
                  hintText: "user@gmail.com",
                  labelText: "user@gmail.com",
                ),
              ),
```

```
42                const SizedBox(
43                  height: 10,
44                ),
45
46                TextField(
47                  obscureText: show,
48                  decoration: InputDecoration(
49                    hintText: "Password",
50                    labelText: ".........",
51                    prefixIcon: const Icon(Icons.lock),
52                    suffixIcon: IconButton(
53                        onPressed: () {
54                        setState(() {
55                        show = !show;
56                      });
57                    },
58                        icon: Icon(show ? Icons.
   visibility : Icons.visibility_off)),
59                    ),
60                  ),
61                ],
62              ),
63            ),
64          ),
65      );
66
67    }
68 }
```

QUESTION TWO

Write a program to create a login interface as shown in the figure below which you can write email address and password for login process, also you can be able to show or hide a password.

[8 marks]



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

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

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

  @override
  State<HamzApp> createState() => _HamzApp();
}

class _HamzApp extends State<HamzApp> {

  bool show = true;


  @override

  Widget build(BuildContext context){
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: const Text("Login",style: TextStyle(
            fontSize: 28,

        ),),
        centerTitle: true,
      ),

      body:Center(
        child: Column(
          children: [
            const TextField(
              decoration: InputDecoration(
                prefixIcon: Icon(Icons.email),
                hintText: "user@gmail.com",
                labelText: "user@gmail.com",
              ),
            ),
```

```
42              const SizedBox(
43                height: 10,
44              ),
45
46              TextField(
47                obscureText: show,
48                decoration: InputDecoration(
49                  hintText: "Password",
50                  labelText: ".........",
51                  prefixIcon: const Icon(Icons.lock),
52                  suffixIcon: IconButton(
53                      onPressed: () {
54                      setState(() {
```

Bottom Navigation Bar

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

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

  @override

  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: const Text('Bottom Navigation Bar'),
          centerTitle: true,
        ),
        bottomNavigationBar: BottomNavigationBar(
          backgroundColor: Colors.greenAccent,
            items: const [
              BottomNavigationBarItem(icon: Icon(Icons.home,color: Colors.blue,),
              label: 'Home'),

              BottomNavigationBarItem(icon: Icon(Icons.settings,color: Colors.blue
,),
                  label: 'settings'),
```

```
27
28                BottomNavigationBarItem(icon: Icon(Icons.favorite,color: Colors.blue
   ,),
29                    label: 'Favorite'),
30
31                BottomNavigationBarItem(icon: Icon(Icons.share,color: Colors.blue,),
32                    label: 'share'),
33
34                BottomNavigationBarItem(icon: Icon(Icons.notifications,color: Colors.
   blue,),
35                    label: 'Noti'),
36              ],
37          ),
38        ),
39      );
40    }
41 }
```