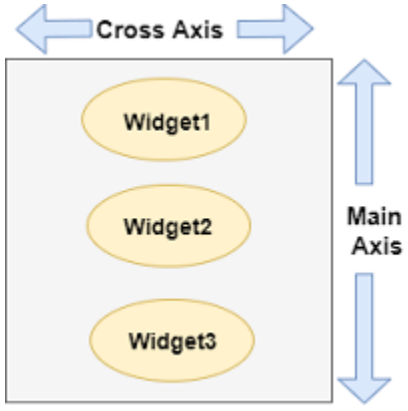
Aim - To design Flutter UI by including common widgets.

Theory -

1. Scaffold
   The Scaffold is a widget in Flutter used to implement the basic material design visual layout structure. It is quick enough to create a general-purpose mobile application and contains almost everything we need to create a functional and responsive Flutter app. This widget is able to occupy the whole device screen. In other words, we can say that it is mainly responsible for creating a base for the app screen on which the child widgets hold on and render on the screen. It provides many widgets or APIs for showing Drawer, SnackBar, BottomNavigationBar, AppBar, FloatingActionButton, and many more.

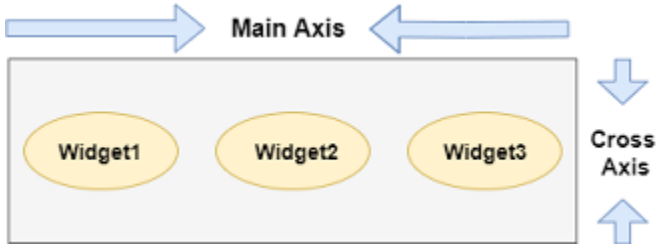
2. Container
   The container in Flutter is a parent widget that can contain a child widget and manage them efficiently through width, height, padding, background color, etc. It is a widget that combines common painting, positioning, and sizing of the child widgets.
   A container widget is the same as a tag in HTML. If this widget does not contain any child widget, it will fill the whole area on the screen automatically. Otherwise, it will wrap the child widget according to the specified height & width. This widget cannot render directly without any parent widget. We can use the Scaffold widget, Center widget, Padding widget, Row widget, or Column widget as its parent widget.

3. Column
   This widget arranges its children in a vertical direction on the screen. In other words, it will expect a vertical array of children widgets. If the child widgets need to fill the available vertical space, we must wrap the children widgets in an Expanded widget.
   A column widget does not appear scrollable because it displays the widgets within the visible view. So it is considered wrong if we have more children in a column that will not fit in the available space. If we want to make a scrollable list of column widgets, we need to use the ListView Widget.



4. Row

This widget arranges its children in a horizontal direction on the screen. In other words, it will expect child widgets in a horizontal array. If the child widgets need to fill the available horizontal space, we must wrap the children widgets in an Expanded widget.
A row widget does not appear scrollable because it displays the widgets within the visible view. So it is considered wrong if we have more children in a row which will not fit in the available space. If we want to make a scrollable list of row widgets, we need to use the ListView widget.



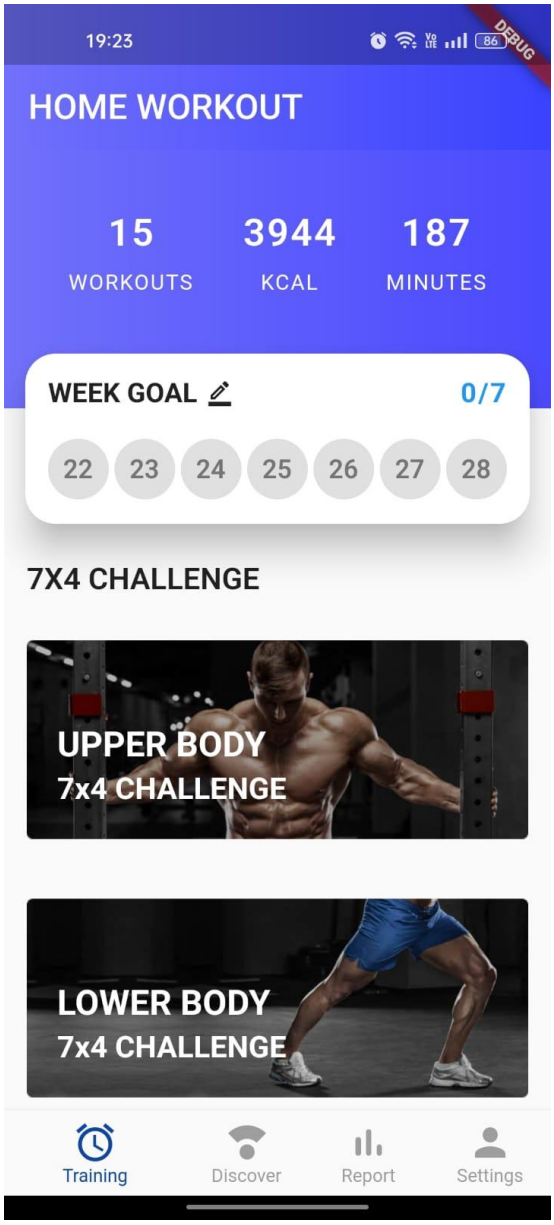
5. Stack
   The stack is a widget in Flutter that contains a list of widgets and positions them on top of each other. In other words, the stack allows developers to overlap multiple widgets into a single screen and renders them from bottom to top. Hence, the first widget is the bottommost item, and the last widget is the topmost item.

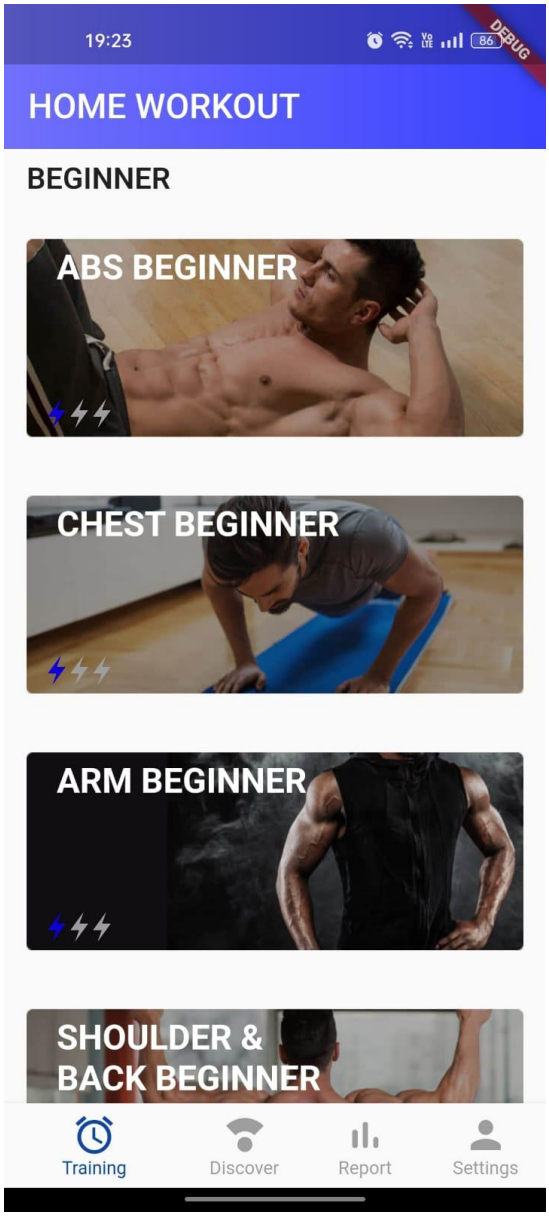
Output -

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        elevation: 0,
        title: Text(
          'HOME WORKOUT',
          style: TextStyle(
            color: Colors.white,
            fontSize: 23,
          ),
        ),
        flexibleSpace: Container(
          decoration: BoxDecoration(
              gradient: LinearGradient(colors: [
            Color.fromARGB(255, 113, 113, 251),
```

```
      Color.fromARGB(236, 60, 60, 254)
    ])),
  ),
),
body: SingleChildScrollView(
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      SizedBox(
        width: MediaQuery.of(context).size.width,
        height: 235,
        child: Stack(
          children: [
            Positioned(
              child: Container(
                padding: EdgeInsets.all(10),
                height: 170,
                decoration: BoxDecoration(
                    gradient: LinearGradient(colors: [
                  Color.fromARGB(255, 113, 113, 251),
                  Color.fromARGB(236, 60, 60, 254)
                ])),
                child: Column(
                  children: [
                    SizedBox(
                      height: 20,
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.spaceAround,
                      children: [
                        Column(
                          children: [
                            Text(
                              '15',
                              style: TextStyle(
                                color: Colors.white,
                                fontSize: 25,
                                letterSpacing: 1.2,
                                fontWeight: FontWeight.w600,
                              ),
                            ),
                            SizedBox(
                              height: 10,
                            ),
```

```
    Text(
      'WORKOUTS',
      style: TextStyle(
        color: Colors.white,
        fontSize: 14,
        letterSpacing: 1,
      ),
    )
  ],
),
Column(
  children: [
    Text(
      '3944',
      style: TextStyle(
        color: Colors.white,
        fontSize: 25,
        letterSpacing: 1.2,
        fontWeight: FontWeight.w600,
      ),
    ),
    SizedBox(
      height: 10,
    ),
    Text(
      'KCAL',
      style: TextStyle(
        color: Colors.white,
        fontSize: 14,
        letterSpacing: 1,
      ),
    )
  ],
),
```





Conclusion -

We learned various new widgets such as column, row, stack, sizedbox, scaffold, appbar etc. We implemented these widgets and created an app. Different attributes of these widgets were used to change the style, design, color, position, and other properties of widgets.