

**SWE 413: ENGINEERING MOBILE APPS**

**NAME: Jolayemi Olugbenga David**

**MATRIC NUMBER: 21/10MSS006**

**FACULTY: Faculty of Computing and Applied Sciences**

**QUESTION:**

1. Write extensively on stateless and stateful widgets in flutter.

**SUBMITTED TO:**

Mr. Dare Oladipo

**THOMAS ADEWUMI UNIVERSITY, OKO-IRESE**

**SUBMISSION DATE:**

17th November 2024

SOLUTION:

1. Widgets are the basic components that serves as the graphical elements which the user sees and interacts with such as buttons, images and dialogue boxes. There are different ways in which these widgets are managed and ways by which their states are monitored; each of this has a significant role on the development process of the app. The 2 common types of widgets are **STATELESS** and **STATEFUL Widgets.** Discussed below are a couple of notable facts about each of them stating how they work, and in which instance one might be used over the other:
   * **STATELESS WIDGET:**  These are widgets whose “state” cannot be altered once they have been created. Once built, these widgets are “immutable”, which means that changes to the variables, icons, buttons or retrieval of data on such page will not affect the state of the app as it will remain static. Common examples of stateless widgets are Icon, IconButton, Text and Image. Typically, stateless widgets subclass “StatelessWidget”. Below is an example of the StatelessWidget in use:

import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Container();

}

}

The code above represents a stateless widget. MyApp is the name of the stateless widget that is being invoked by the runApp() method and then extends a stateless widget. A build function is being invoked which is being overridden in MyApp, but then takes BuildContext as a parameter. The BuildContext is paramount to development as it is used to locate each widget within the widget tree.

* + **STATEFUL WIDGET:** These are widgets whose state can be changed after they’ve been built. These states are “mutable” meaning that they can change multiple times over during their lives. This basically points to the fact that the app’s state can change several times depending on variables, inputs and user interactions. It is majorly used when user interaction is required and alteration to user interface is paramount. Examples includes CheckBox, Form and TextField. Below is an example of how StatefulWidget is implemented:

import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatefulWidget {

@override

\_MyAppState createState() => \_MyAppState();

}

class \_MyAppState extends State<MyApp> {

@override

Widget build(BuildContext context) {

return Container();

}

}

The above code snippet can be broken down into; MyApp, which is being called from the runApp() and then extends to a stateful widget. After this, we then create a state function in the MyApp class. This createState() generates a mutable state for this widget at a fixed point in the tree. This function then instantiates a state of the subclass being called. The \_MyAppState is responsible for the modifications on all other widgets. This class overrides the build function, which accepts a BuildContext parameter. The build function returns a customizable widget that defines the app's user interface. As a stateful widget, the build method is executed repeatedly, rebuilding the entire UI to reflect any changes.