St. Francis Institute of Technology, Mumbai-400 103
Department of Information Technology

A.Y. 2024-2025
Class: TE-ITA/B, Semester: VI

Subject: MAD & PWA LAB

Experiment – 7: Testing and deploying production ready Flutter App on Android platform.

1. Aim: To test and deploy production ready Flutter App on Android platform. 2. Objectives: After study of this experiment, the student will be able to

- Test a production ready Flutter App.
- Deploy the Flutter App on Android/iOS phone.

3. Outcomes: After study of this experiment, the student will be able to

- Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android/iOS

4. Prerequisite: Dart Programming Language.

5. Requirements: Android Studio, Flutter framework, Internet Connection, mobile PC suite software, USB data cable.

6. Pre-Experiment Exercise:

Brief Theory:

Testing mobile application

Testing is one of the most important phases of mobile app development. You can't build a high-quality app without testing it. The Flutter framework provides comprehensive support for Flutter automated testing of mobile apps. Automated tests help to ensure that your app performs correctly before you publish it while retaining your features and bug fix velocity.

Categories of automated testing:

- A unit test tests a single function, method, or class.
- A widget test (in other UI frameworks referred to as component test) tests a single widget.
- An integration test tests a complete app or a large part of an app.

Deploying the mobile app on a mobile device

Steps to run mobile app on a real device:

1. Install mobile PC suite software on the computer.
2. Enable USB debugging mode in Android device.
3. Connect the mobile device via USB data cable to the computer.

4. Go to the IDE and select the real device to run Flutter App.
5. Click on ‘Run’ to run the App.

7. Laboratory Exercise

A. Program

1. Write 4-5 test cases for your mobile application

main.dart ● adb.exe widget_test.dart ●

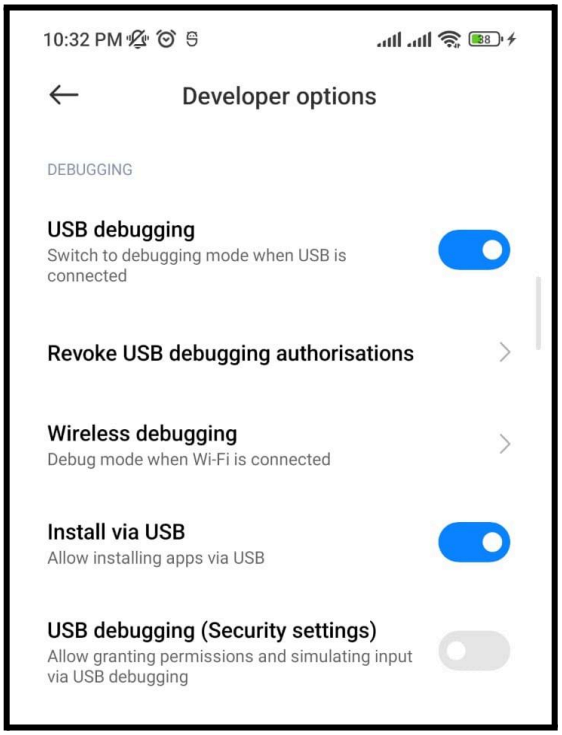
test > widget_test.dart > main

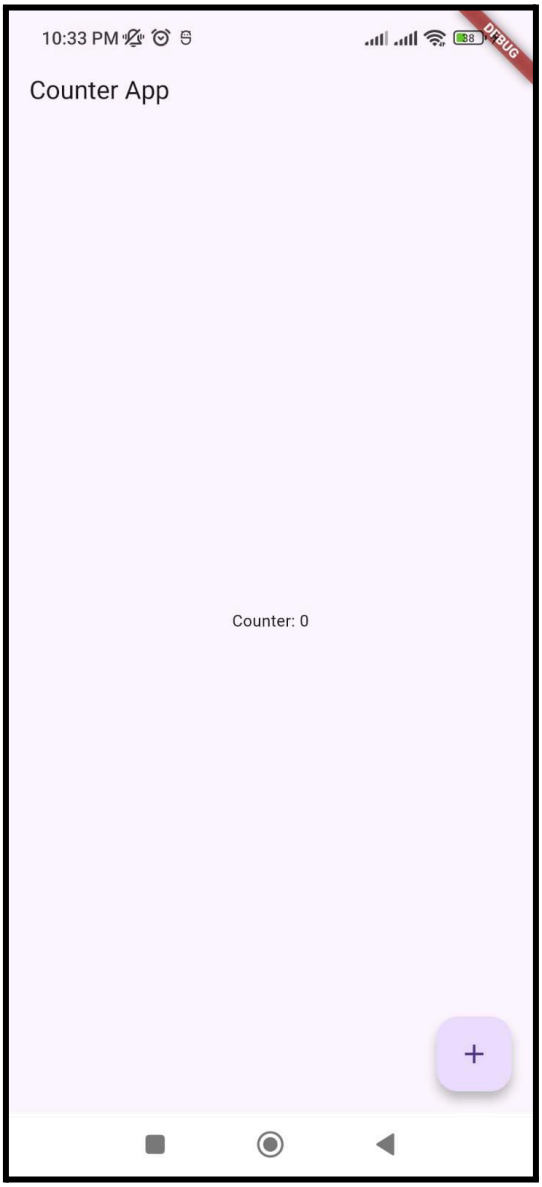
```dart
import 'package:flutter/material.dart';
import 'package:flutter_test/flutter_test.dart';
import 'package:my_test_app/main.dart';

Run | Debug
void main() {
  // Test Case 1: Check if app loads with initial counter 0
  Run | Debug
  testWidgets('App loads with initial counter 0', (WidgetTester tester) async {
    await tester.pumpWidget(MyApp());
    expect(find.text('Counter: 0'), findsOneWidget);
  });

  // Test Case 2: Check if tapping + icon increments counter
  Run | Debug
  testWidgets('Tap on + icon increments counter', (WidgetTester tester) async {
    await tester.pumpWidget(MyApp());
    await tester.tap(find.byIcon(Icons.add));
    await tester.pump();
    expect(find.text('Counter: 1'), findsOneWidget);
  });

  // Test Case 3: Check if AppBar title is present
  Run | Debug
  testWidgets('App has AppBar with correct title', (WidgetTester tester) async {
    await tester.pumpWidget(MyApp());
    expect(find.text('Counter App'), findsOneWidget);
  });

  // Test Case 4: Check if FAB (FloatingActionButton) is present
  Run | Debug
  testWidgets('FAB button is present', (WidgetTester tester) async {
    await tester.pumpWidget(MyApp());
    expect(find.byType(FloatingActionButton), findsOneWidget);
  });

  // Test Case 5: Check if counter Text widget exists
  Run | Debug
  testWidgets('Text widget displays counter value', (WidgetTester tester) async {
    await tester.pumpWidget(MyApp());
    expect(find.byKey(Key('counter')), findsOneWidget);
  });
}
```

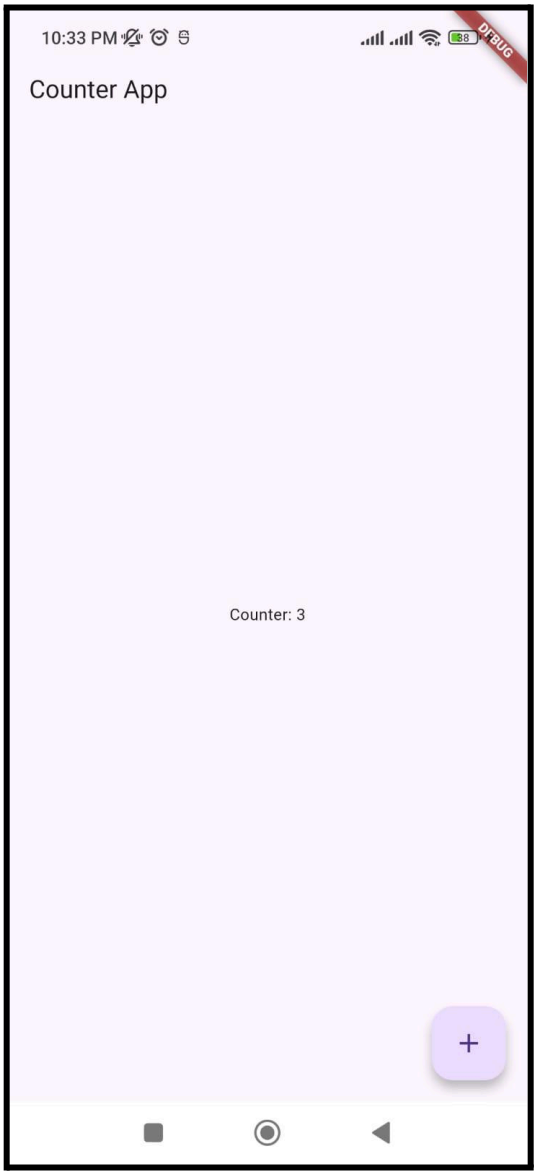
2. Deploy and run the app on your mobile phone.

```
PS C:\Users\pilla\my_test_app> flutter test
>>
00:01 +5: All tests passed!
PS C:\Users\pilla\my_test_app> flutter devices
>>
Found 4 connected devices:
  M2007J17I (mobile) • 393c7a27 • android-arm64  • Android 12 (API 31)
  Windows (desktop)  • windows  • windows-x64    • Microsoft Windows [Version 10.0.22631.5039]
  Chrome (web)       • chrome   • web-javascript • Google Chrome 135.0.7049.43
  Edge (web)         • edge     • web-javascript • Microsoft Edge 135.0.3179.66

Run "flutter emulators" to list and start any available device emulators.

If you expected another device to be detected, please run "flutter doctor" to diagnose potential issu
to wait for connected devices with the "--device-timeout" flag. Visit https://flutter.dev/setup/ for
PS C:\Users\pilla\my_test_app>
```

```
PS C:\Users\pilla\my_test_app> flutter run
Launching lib\main.dart on M2007J17I in debug mode...
Running Gradle task 'assembleDebug'...                          1,590ms
√ Built build\app\outputs\flutter-apk\app-debug.apk
Installing build\app\outputs\flutter-apk\app-debug.apk...           4.1s
I/flutter (18804): [IMPORTANT:flutter/shell/platform/android/android_context_vk_impeller.cc(60)] Using the I
).
Syncing files to device M2007J17I...                               50ms

Flutter run key commands.
r Hot reload.
R Hot restart.
h List all available interactive commands.
d Detach (terminate "flutter run" but leave application running).
c Clear the screen
q Quit (terminate the application on the device).

A Dart VM Service on M2007J17I is available at: http://127.0.0.1:64283/2Gx3GLHbzUA=/
W/Looper  (18804): PerfMonitor looperActivity : package=com.example.my_test_app/.MainActivity time=0ms laten
-1  historyMsgCount=2 (msgIndex=1 wall=290ms seq=3 late=7ms h=android.app.ActivityThread$H w=110) (msgIndex=
android.app.ActivityThread$H w=159)
The Flutter DevTools debugger and profiler on M2007J17I is available at: http://127.0.0.1:9101?uri=http://12
I/Choreographer(18804): Skipped 298 frames!  The application may be doing too much work on its main thread.
I/AdrenoGLES-0(18804): QUALCOMM build                   : 16c8186230, I159ae7f0bb
I/AdrenoGLES-0(18804): Build Date                       : 04/05/21
I/AdrenoGLES-0(18804): OpenGL ES Shader Compiler Version: EV031.32.02.10
I/AdrenoGLES-0(18804): Local Branch                     :
I/AdrenoGLES-0(18804): Remote Branch                    :
I/AdrenoGLES-0(18804): Remote Branch                    :
I/AdrenoGLES-0(18804): Reconstruct Branch               :
I/AdrenoGLES-0(18804): Build Config                     : S P 10.0.7 AArch64
I/AdrenoGLES-0(18804): Driver Path                      : /vendor/lib64/egl/libGLESv2_adreno.so
I/AdrenoGLES-0(18804): PFP: 0x016ee197, ME: 0x00000000
D/SurfaceView(18804): UPDATE null, mIsCastMode = false
D/hw-ProcessState(18804): Binder ioctl to enable oneway spam detection failed: Invalid argument
W/Looper  (18804): PerfMonitor doFrame : time=106ms vsyncFrame=0 latency=4967ms procState=-1 historyMsgCount
ate=7ms h=android.app.ActivityThread$H w=110) (msgIndex=2 wall=3116ms seq=4 late=291ms h=android.app.Activity
D/SurfaceView(18804): UPDATE Surface(name=SurfaceView[com.example.my_test_app/com.example.my_test_app.MainAc
nMode = false
D/SurfaceControl(18804): nativeSetScreenProjection ===> setScreenProjection
D/SurfaceControl(18804): nativeSetScreenProjection ===> setScreenProjection
D/SurfaceControl(18804): nativeSetScreenProjection ===> setScreenProjection
D/DecorView[](18804): getWindowModeFromSystem  windowmode is 1
D/DecorView[](18804): updateDecorCaptionStatus displayWindowDecor is false
D/DecorView[](18804): onWindowFocusChanged hasWindowFocus true
```


10:32 PM
Developer options
DEBUGGING
USB debugging
Switch to debugging mode when USB is connected
Revoke USB debugging authorisations
Wireless debugging
Debug mode when Wi-Fi is connected
Install via USB
Allow installing apps via USB
USB debugging (Security settings)
Allow granting permissions and simulating input via USB debugging


10:33 PM
DEBUG
Counter App
Counter: 0



B. Result/Observation

1. Print out of program code and output.

8. Post-Experimental Exercise

A. Questions:

1. Describe the steps to publish a flutter mobile application.

B. Conclusion:

1. Write what you have learnt in the experiment.

C. References:

1. https://docs.flutter.dev/testing
2. https://flutteragency.com/how-to-run-test-flutter-app-on-a-real-device/ 3. Beginning App Development with Flutter: Create Cross-Platform Mobile Apps, By Rap Payne, 2019.