Name : Vrushabh Ghuse **Roll no : 22** **D15A**

Experiment No : 1

Aim : To install and configure the Flutter Environment

Theory:

Flutter is an open-source UI software development toolkit created by Google. It is used for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter uses the Dart programming language and follows a reactive programming paradigm.

Key Concepts and Theoretical Aspects:

Widget Tree:

In Flutter, everything is a widget. Widgets are the basic building blocks of the user interface.
The entire UI is represented as a tree of widgets. Each widget can be a structural element (like a button or text) or a layout (like a row or column). Declarative UI:

Flutter follows a declarative approach, where the UI is described in terms of what it should look like based on the current state. Developers declare the desired state, and Flutter takes care of updating the UI to reflect that state. Hot Reload:

One of Flutter's key features is Hot Reload, which allows developers to instantly see the impact of the code changes without restarting the entire application. This accelerates the development process and enhances productivity. State Management:

Flutter provides various options for managing the state of an application. Understanding and implementing effective state management is crucial for building scalable and maintainable Flutter applications. Dart Language:

Flutter uses Dart as its programming language. Understanding the Dart language and its features is essential for efficient Flutter development.
Material Design and Cupertino Widgets:

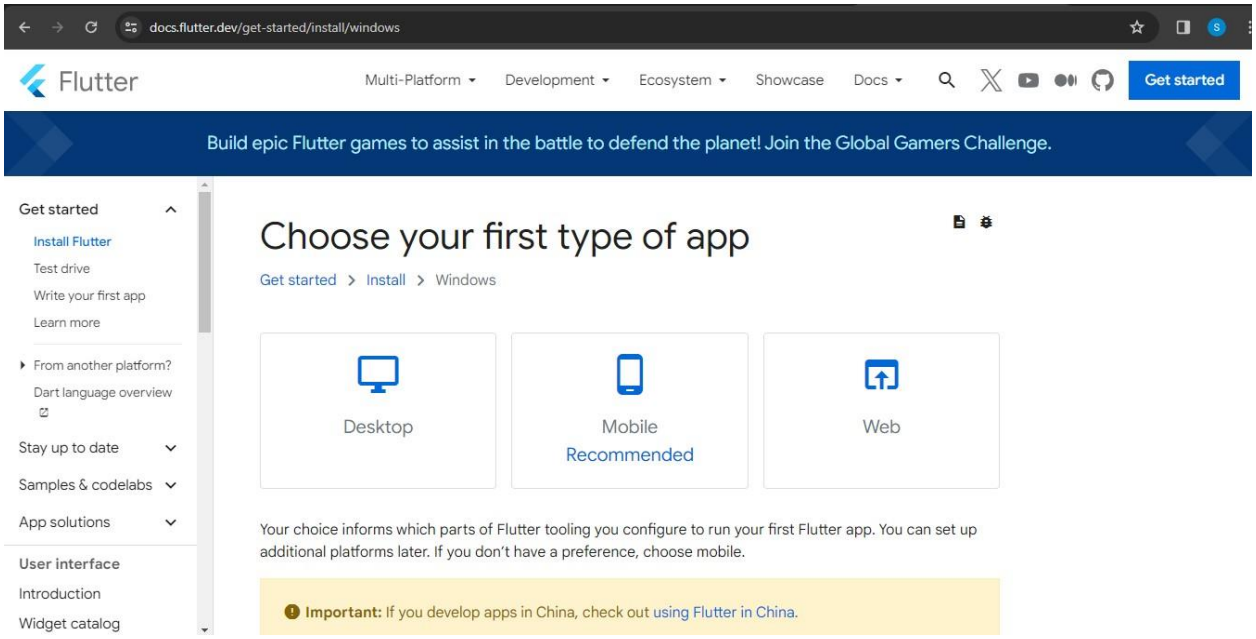
Flutter provides a set of widgets that implement the Material Design guidelines for Android and iOS-style widgets for iOS through the Cupertino library. Understanding

how to use these widgets helps in creating platform-specific user interfaces.
Asynchronous Programming:

Dart is designed to support asynchronous programming, which is important for handling tasks like network requests without blocking the UI thread. Understanding asynchronous programming concepts is crucial for developing responsive Flutter applications.

Installation of Flutter:

Step 1: Download the installation bundle of the Flutter Software Development Kit for windows.
To download Flutter SDK, Go to its official website
https://docs.flutter.dev/get-started/install ,
you will get the following screen.



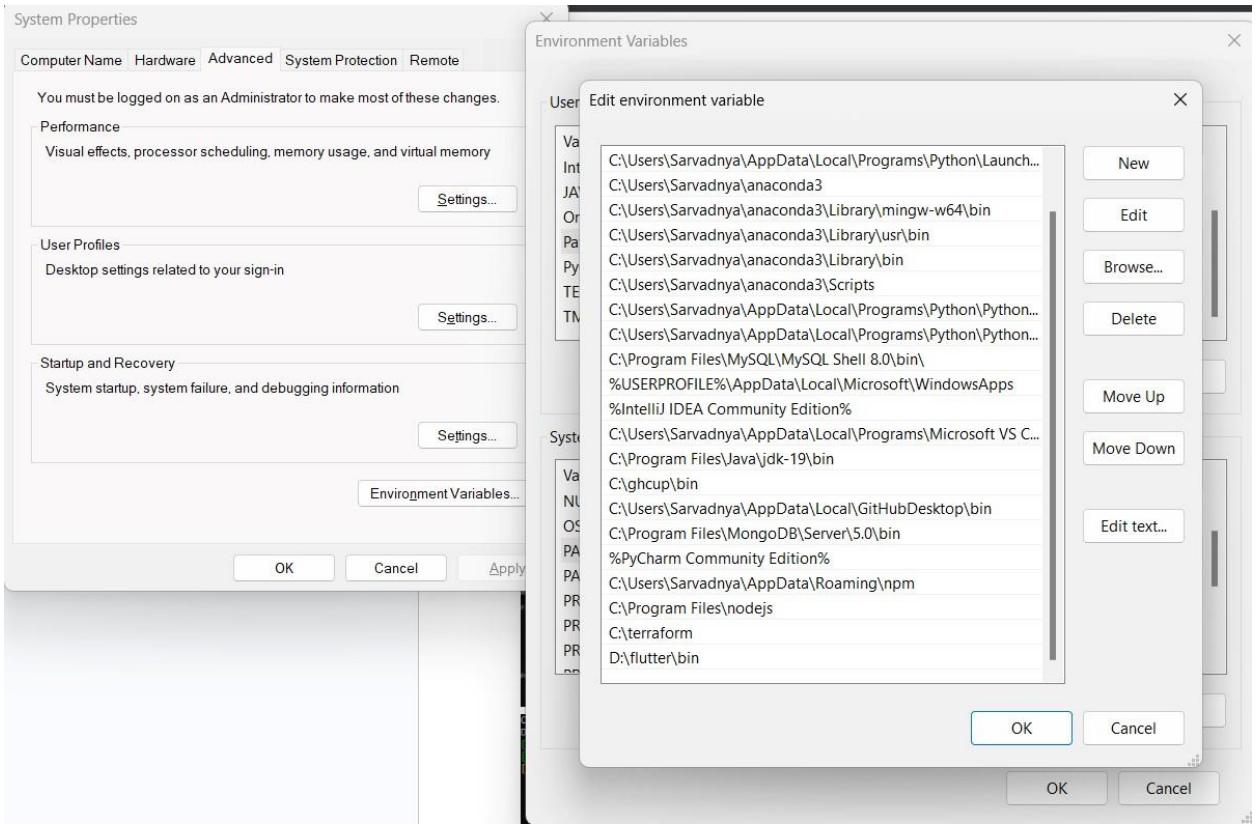
Step 2: Next, to download the latest Flutter SDK, click on the Windows icon. Here, you will find the download link for SDK.

Step 3: When your download is complete, extract the zip file and place it in the desired installation folder or location, for example, C: /Flutter.

Step 4: To run the Flutter command in regular windows console, you need to update the system path to include the flutter bin directory. The following steps are required to do this:

Step 4.1: Go to MyComputer properties -> advanced tab -> environment variables. You will get the following screen.

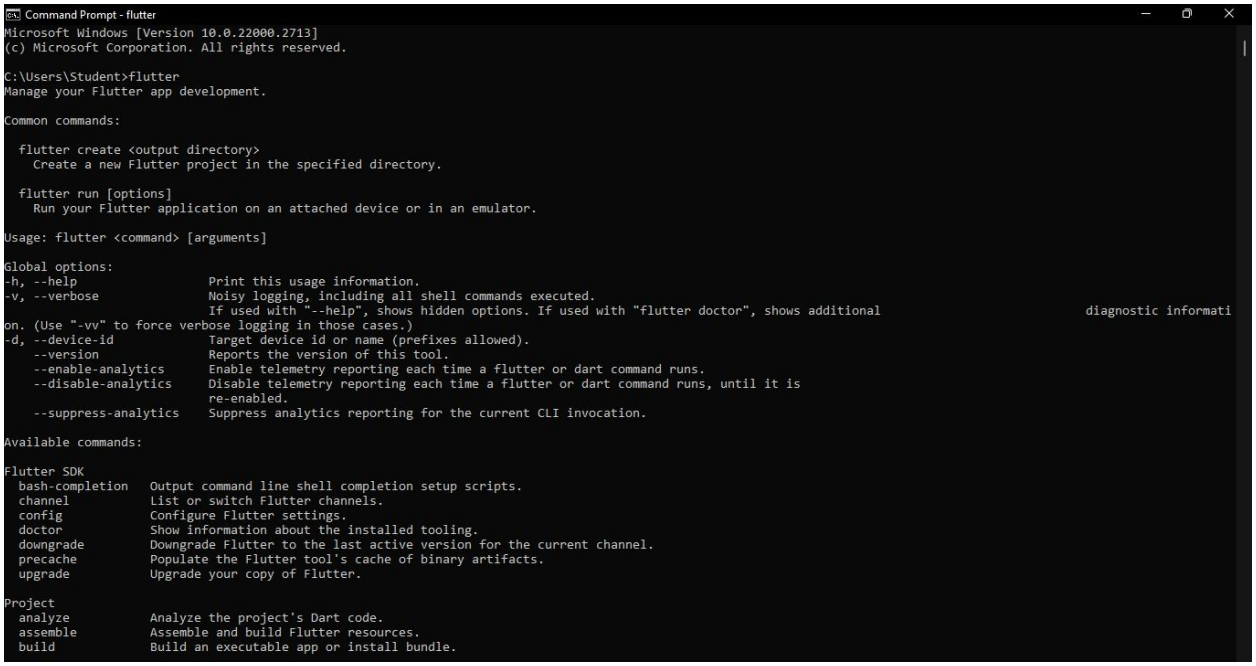
Editing Environment Variables:

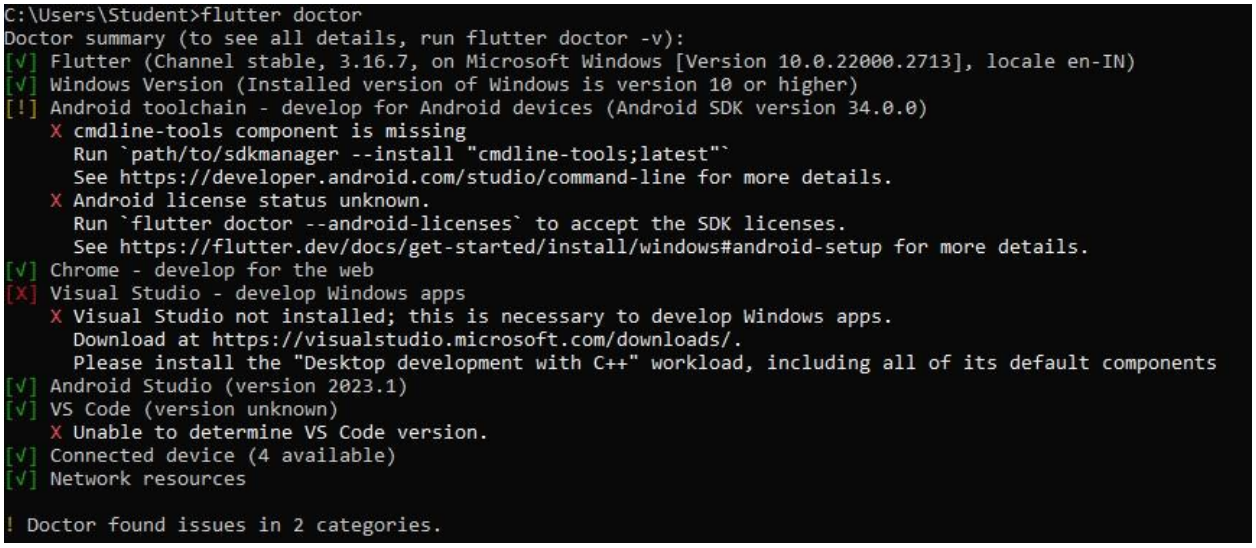


Step 4.3: In the above window, click on New->write path of Flutter bin folder in variable value -

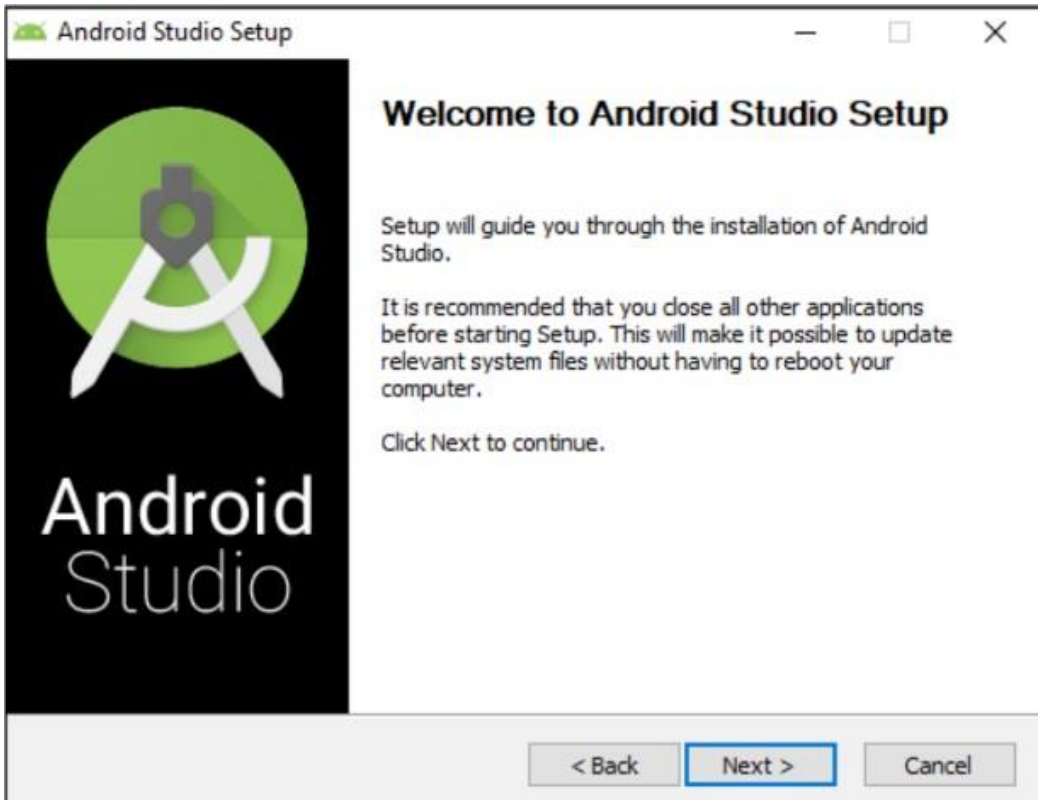
> ok -> ok -> ok.

Step 5: Now, run the $ flutter command in command prompt.

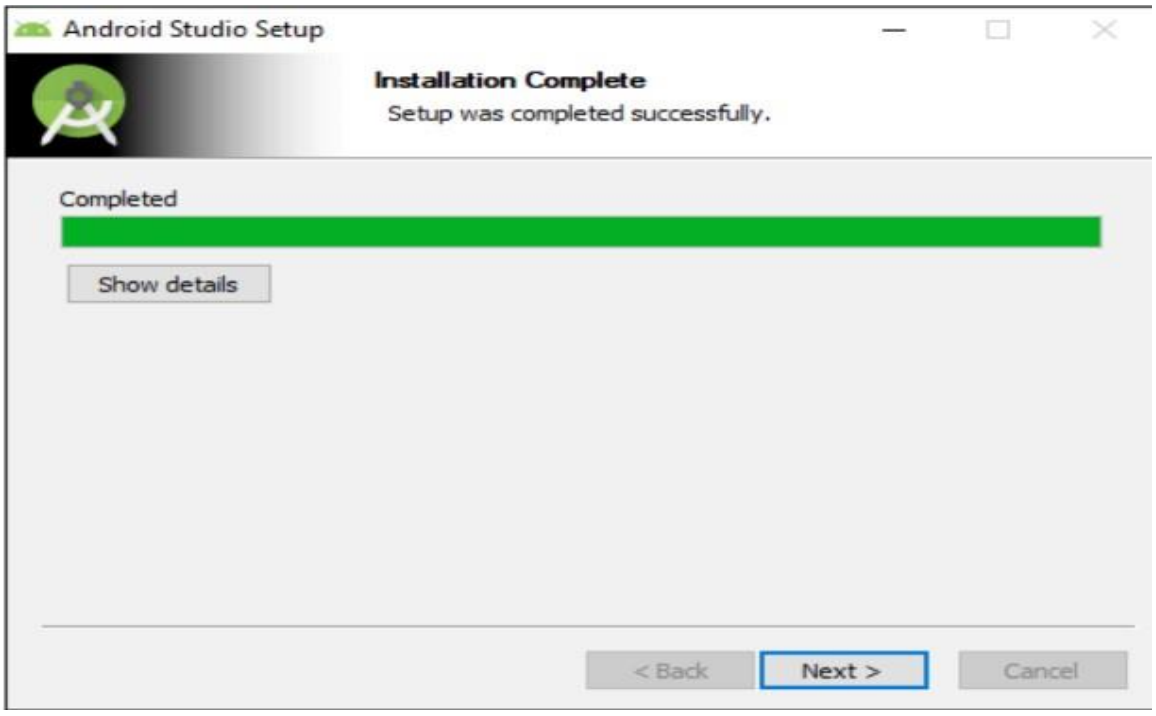




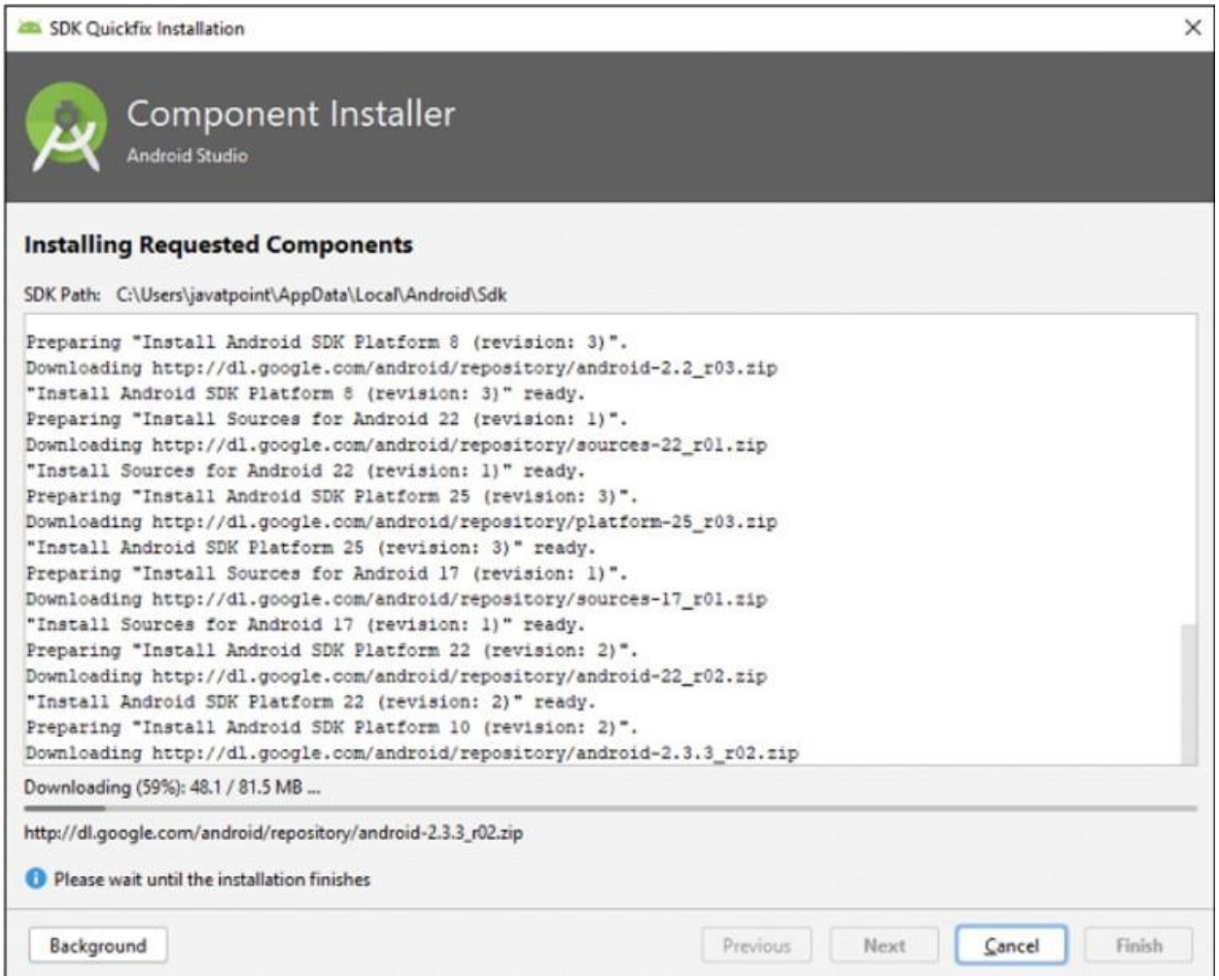
Installation of Android Studio:



Step : Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.

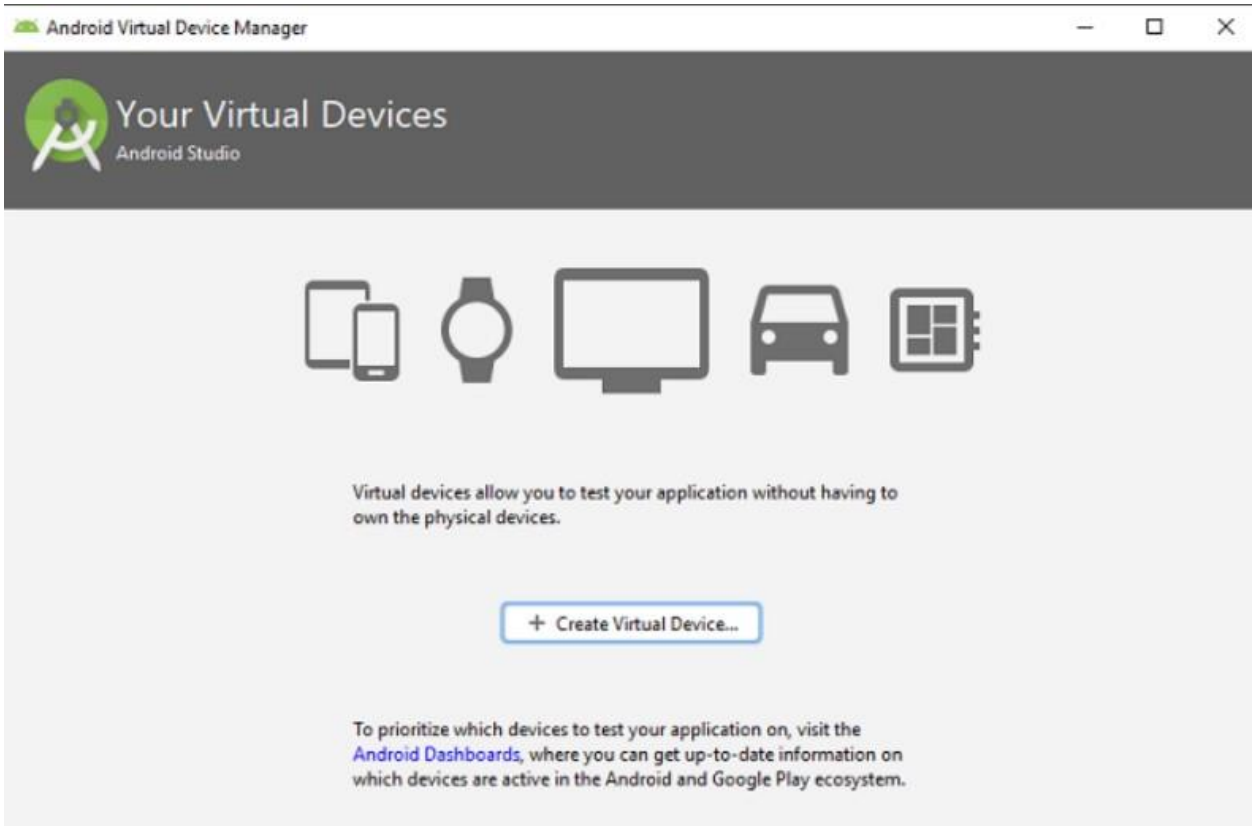


Step : In the above screen, click Next-> Finish. Once the Finish button is clicked, you need to choose the 'Don't import Settings option' and click OK. It will start the Android Studio.



Step 8: Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application.

Step 8..2: To set an Android emulator, go to Android Studio > Tools > Android > AVD Managerand select Create Virtual Device. Or, go to Help->Find Action->Type Emulator in the search box. You will get the following screen.



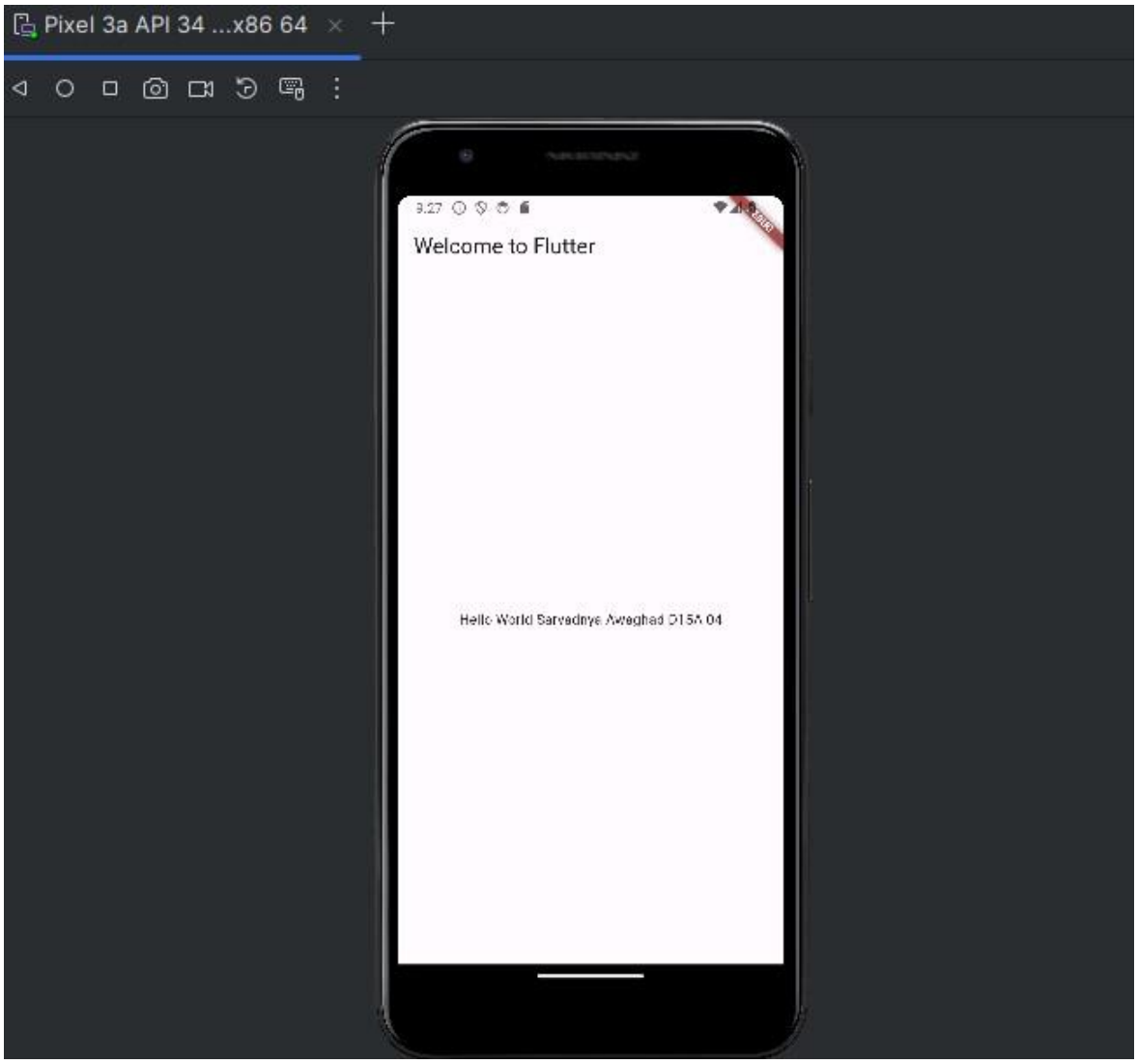
Step 8.2: Choose your device definition and click on Next.
Step 8.3: Select the system image for the latest Android version and click on Next.
Step 8.4: Now, verify the all AVD configuration. If it is correct, click on Finish. The following screen appears.

Step 8.5: Last, click on the icon pointed into the red color rectangle. The Android emulator displayed as below screen.

Flutter Hello World App :

```
import 'package:flutter/material.dart';
void main() {
  runApp(const MyApp());
}
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return MaterialApp( title: 'Welcome
    to Flutter', home: Scaffold(
        appBar: AppBar(
          title: const Text('Welcome to Flutter'),
        ),
        body: const Center(
          child: Text('               Hello World \n Sarvadnya Awaghad D15A 04'),
```

```
        ),
      ),
    );
  }
}
```



Conclusion : In the above experiment , we have successfully configured and installed flutter and android studio in our desktop. AVD emulator is installed and flutter app is executed successfully.