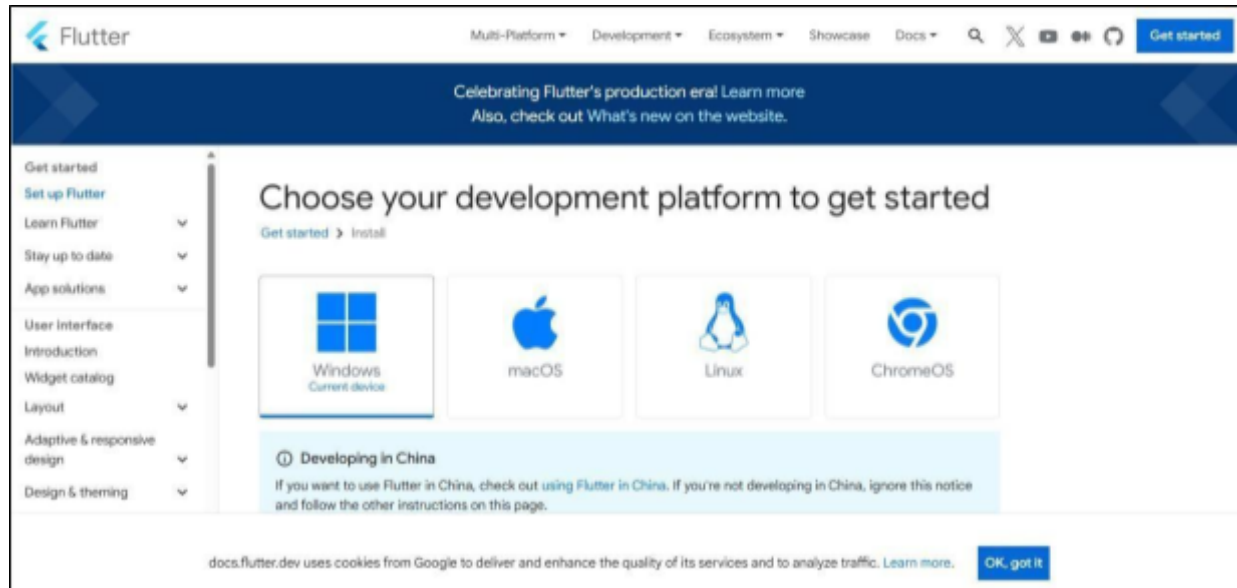


MAD & PWA Lab Journal

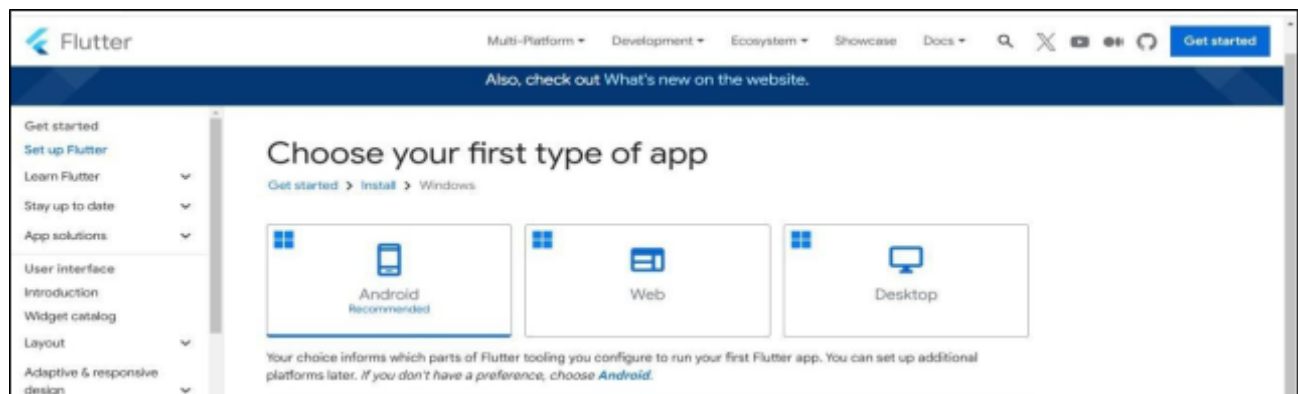
| | |
|-------------------|---|
| Experiment No. | 01 |
| Experiment Title. | To install and configure the Flutter Environment |
| Roll No. | 50 |
| Name | Siddhant Sathe |
| Class | D15A |
| Subject | MAD & PWA Lab |
| Lab Outcome | LO1: Understand cross platform mobile application development using Flutter framework |
| Grade: | |

AIM:- Installation and Configuration of Flutter Environment.

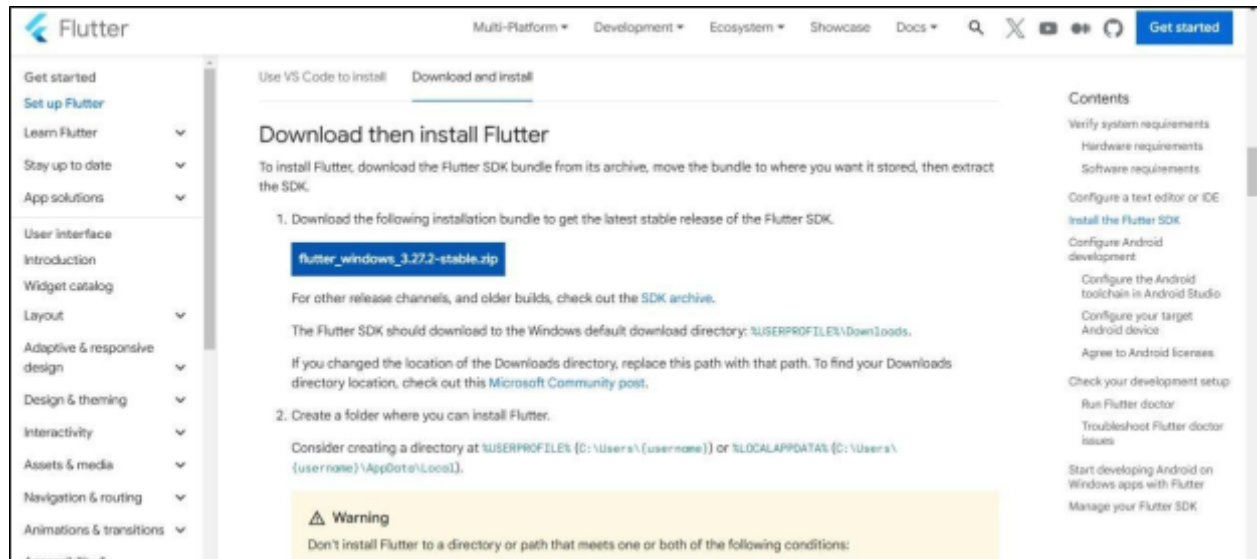
Step 1: Go to the official Flutter website: <https://docs.flutter.dev/get-started/install>



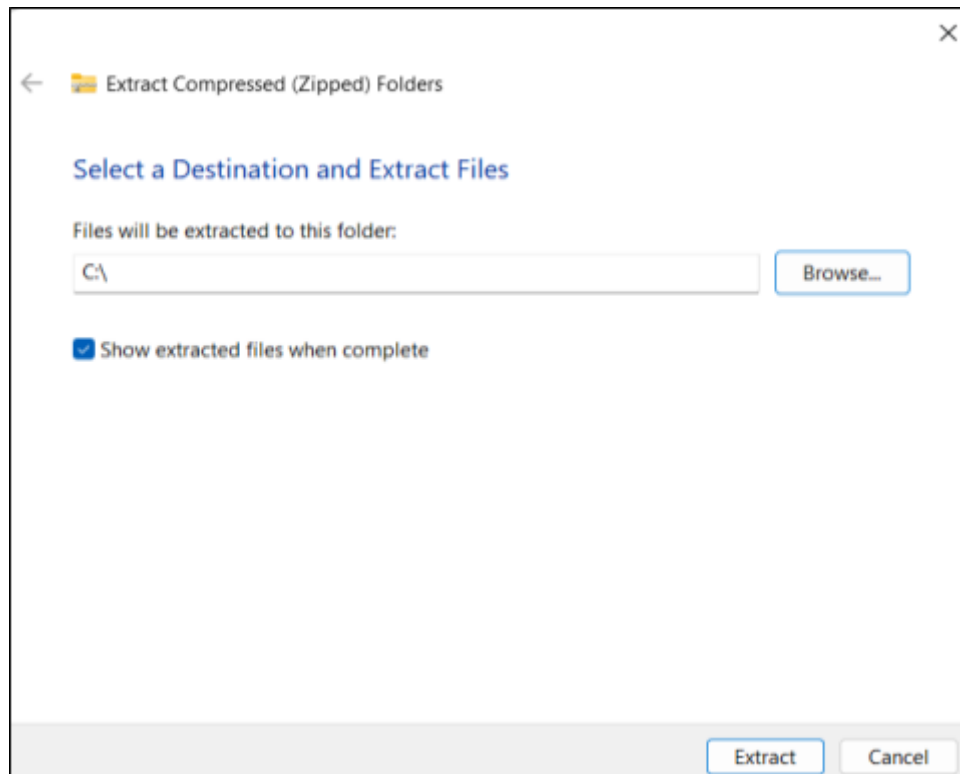
Step 2: To download the latest Flutter SDK, click on the Windows icon > Android



Step 3: For Windows, download the stable release (a .zip file).



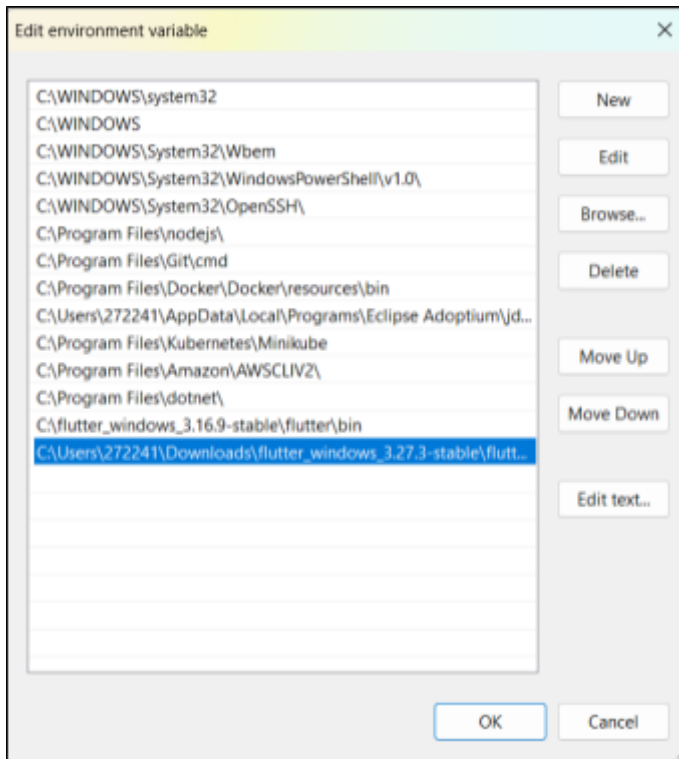
Step 4: Extract the ZIP file to a folder (e.g., C:\flutter).



Step 5 :- Add Flutter to System PATH

Right-click on the Start Menu > System > Advanced system settings > Environment Variables. Under System Variables, find Path and click Edit.

Add the full path to the flutter/bin directory (e.g., C:\flutter\bin).



Step 6 :- Now, run the \$ flutter command in command prompt.

```
Administrator: Command Prompt - flutter
C:\Users\INF505-02>flutter
Manage your Flutter app development.

Common commands:

  flutter create <output directory>
    Create a new Flutter project in the specified directory.

  flutter run [options]
    Run your Flutter application on an attached device or in an emulator.

Usage: flutter <command> [arguments]

Global options:
-h, --help          Print this usage information.
-v, --verbose       Noisy logging, including all shell commands executed.
                    If used with "--help", shows hidden options. If used with "flutter doctor", shows additional
                    diagnostic information. (Use "-vv" to force verbose logging in those cases.)
-d, --device-id     Target device id or name (prefixes allowed).
--version           Reports the version of this tool.
--enable-analytics  Enable telemetry reporting each time a flutter or dart command runs.
--disable-analytics Disable telemetry reporting each time a flutter or dart command runs, until it is
                    re-enabled.
--suppress-analytics Suppress analytics reporting for the current CLI invocation.

Available commands:

Flutter SDK
  bash-completion  Output command line shell completion setup scripts.
  channel          List or switch Flutter channels.
```

Step 7:- Run the \$ flutter doctor command. This command checks for all the requirements of Flutter app development and displays a report of the status of your Flutter installation

```
Command Prompt - flutter - flutter doctor

C:\Users\Student.VESIT505-22>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.19045.5371], locale en-IN)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[!] Android toolchain - develop for Android devices (Android SDK version 34.0.0)
    X cmdline-tools component is missing
      Run 'path/to/sdkmanager --install "cmdline-tools;latest"'
      See https://developer.android.com/studio/command-line for more details.
    X Android license status unknown.
      Run 'flutter doctor --android-licenses' to accept the SDK licenses.
      See https://flutter.dev/to/windows-android-setup for more details.
[✓] Chrome - develop for the web
[✗] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[✓] Android Studio (version 2021.3)
[✓] VS Code (version 1.96.4)
[✓] Connected device (3 available)
[✓] Network resources

! Doctor found issues in 2 categories.

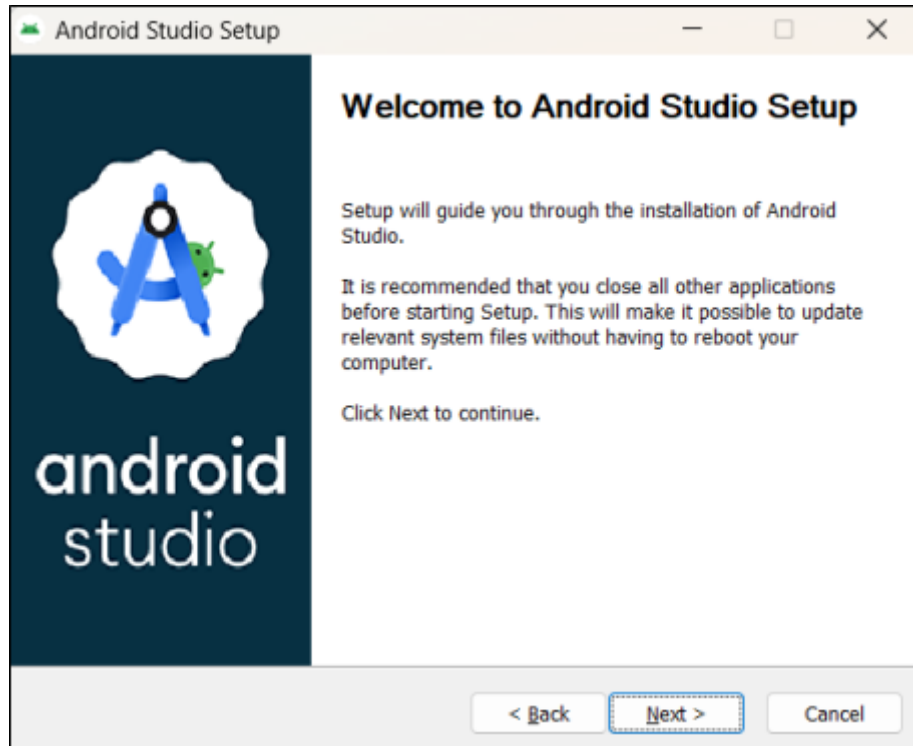
C:\Users\Student.VESIT505-22>
```

Step 8 :- Go to Android Studio and download the installer.

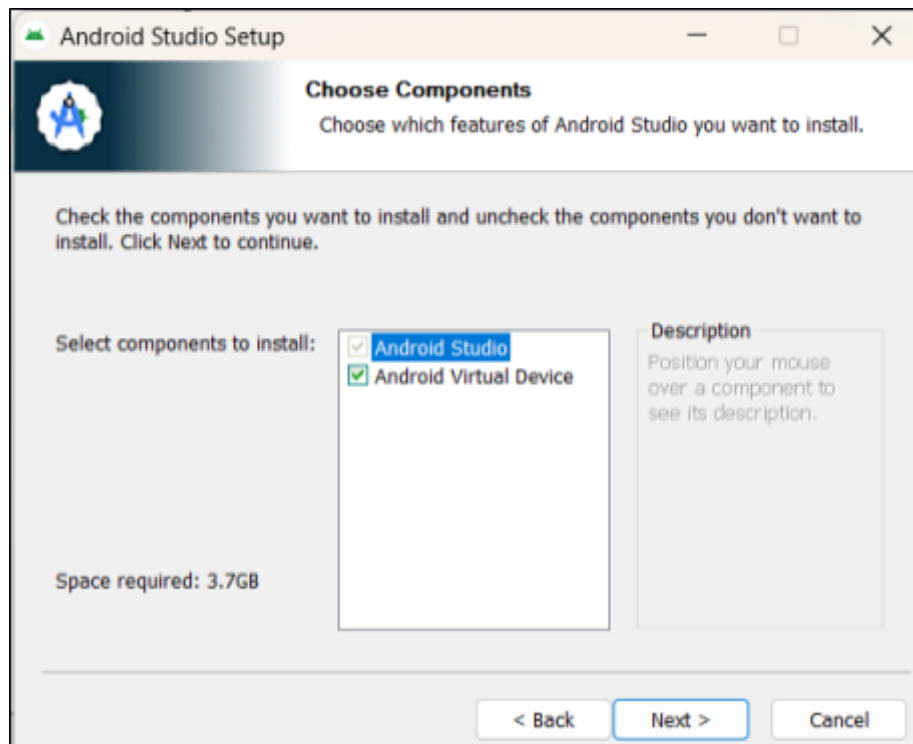
Download the latest version of Android Studio. For more information, see the [Android Studio release notes](#).

| Platform | Android Studio package | Size | SHA-256 checksum |
|-------------------|---|--------|---|
| Windows (64-bit) | android-studio-2024.2.2.13-windows.exe Recommended | 1.2 GB | 7d93d9bf3539f948f609b19e8507b1f502bf69e5d3d44bd38a7ff26c5d3e |
| Windows (64-bit) | android-studio-2024.2.2.13-windows.zip No .exe installer | 1.2 GB | 8559459e2ff9b84ea49ce39de0bf4189dbf451ae37a7ab7999da013b046b7f7 |
| Mac (64-bit) | android-studio-2024.2.2.13-mac.dmg | 1.3 GB | acf8be54d6ce8cf2f1f9b435f0c7addcb9dde2824282f205fd133f8e77d2e613 |
| Mac (64-bit, ARM) | android-studio-2024.2.2.13-mac_arm.dmg | 1.3 GB | 688f8d007e612f3f0c18f316179079dc45e5f93dbd1e6a7d4d380c4cfca35edf7 |
| Linux (64-bit) | android-studio-2024.2.2.13-linux.tar.gz | 1.3 GB | b7fe1ed4a7959bdaca7a8fd57461dbbf9a205e123cc218ed82be88e8b7998cb6 |

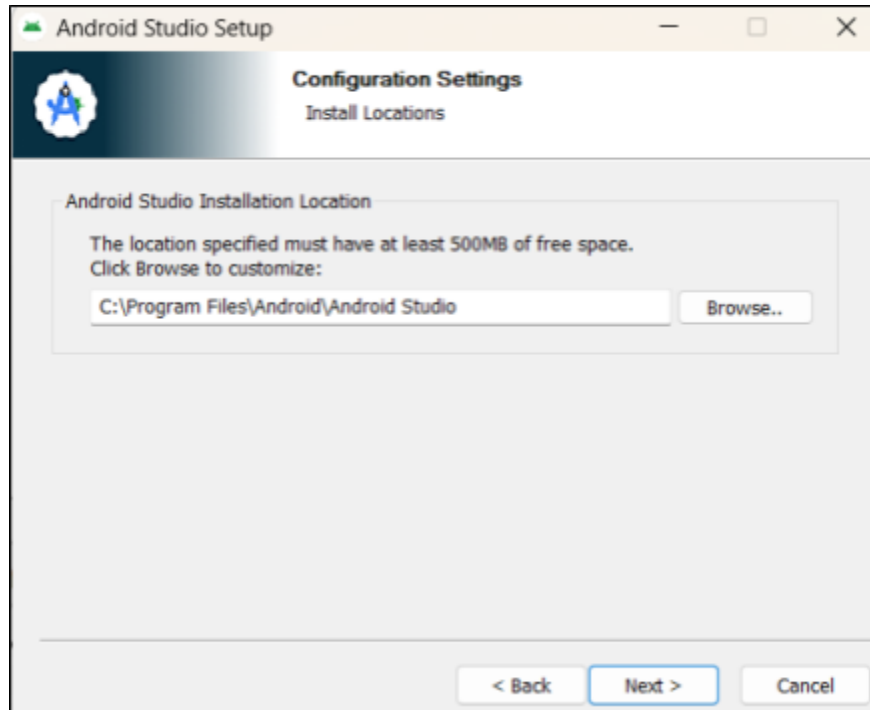
Step 8.1: - When the download is complete, open the .exe file and run it. You will get the following dialog box



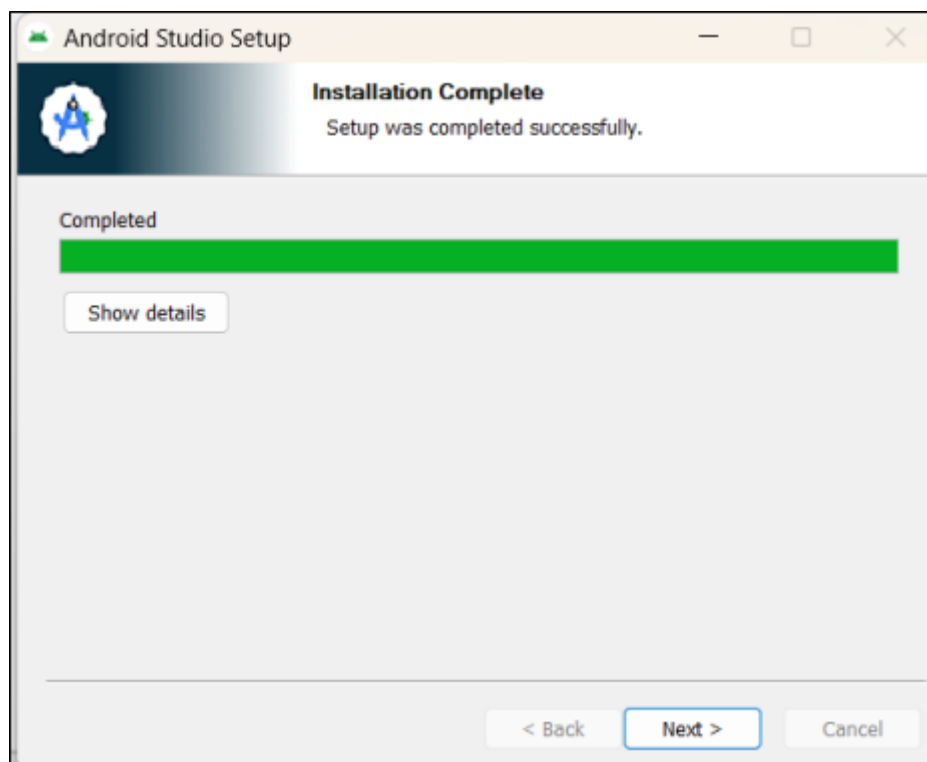
Step 8.2: - Select all the Checkboxes and Click on 'Next' Button.

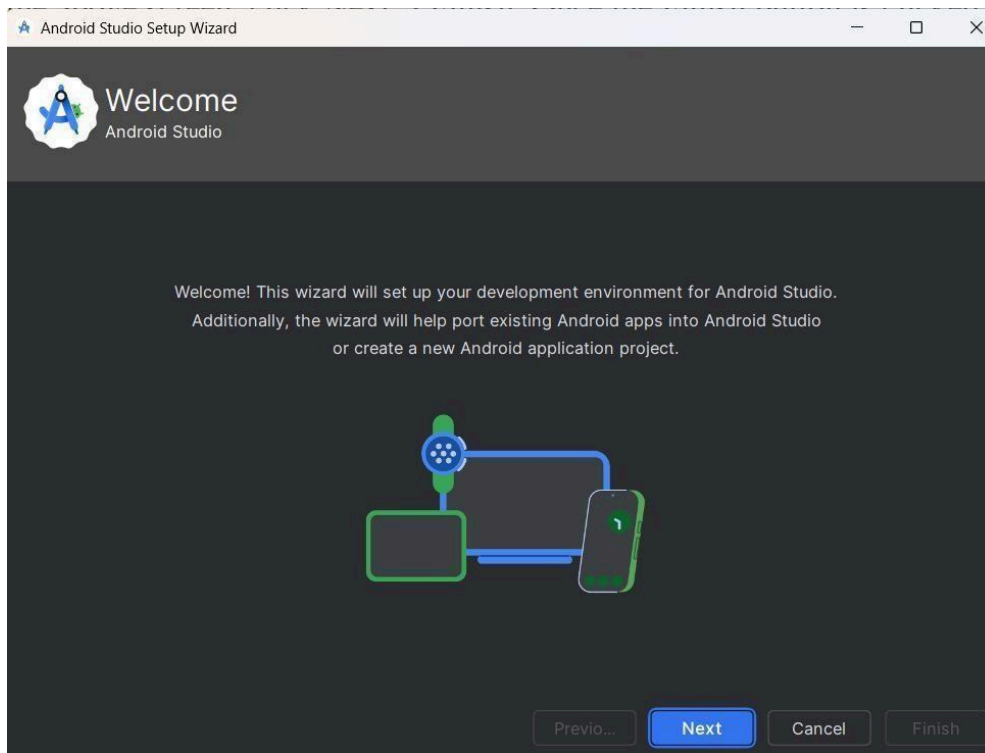


Step 8.3: - Change the destination as per your convenience and click on 'Next' Button.

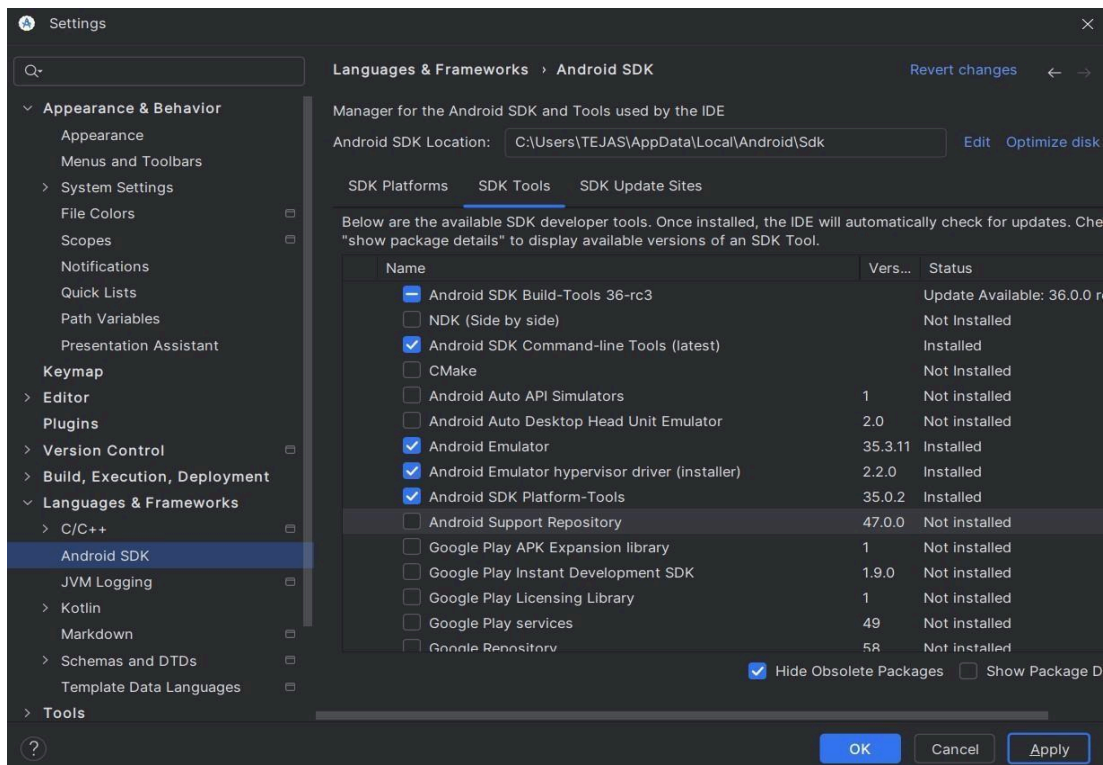


Step 8.4: - Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.





Step 8.5: - Go to Preferences > Appearance & Behavior > System Settings > Android SDK. Select the SDK Tools tab and check Android SDK Command-line Tools and Install it.



Step 9:- Open a terminal and run the following command

```
C:\Users\INFT505-02>flutter doctor --android-licenses
[=====] 40% Fetch remote repository... etch remote repository...
Warning: Additionally, the fallback loader failed to parse the XML.
[=====] 73% Fetch remote repository... etch remote repository...
Warning: Additionally, the fallback loader failed to parse the XML.
[=====] 100% Computing updates...
All SDK package licenses accepted.
```

```
Administrator: Command Prompt - flutter - flutter doctor

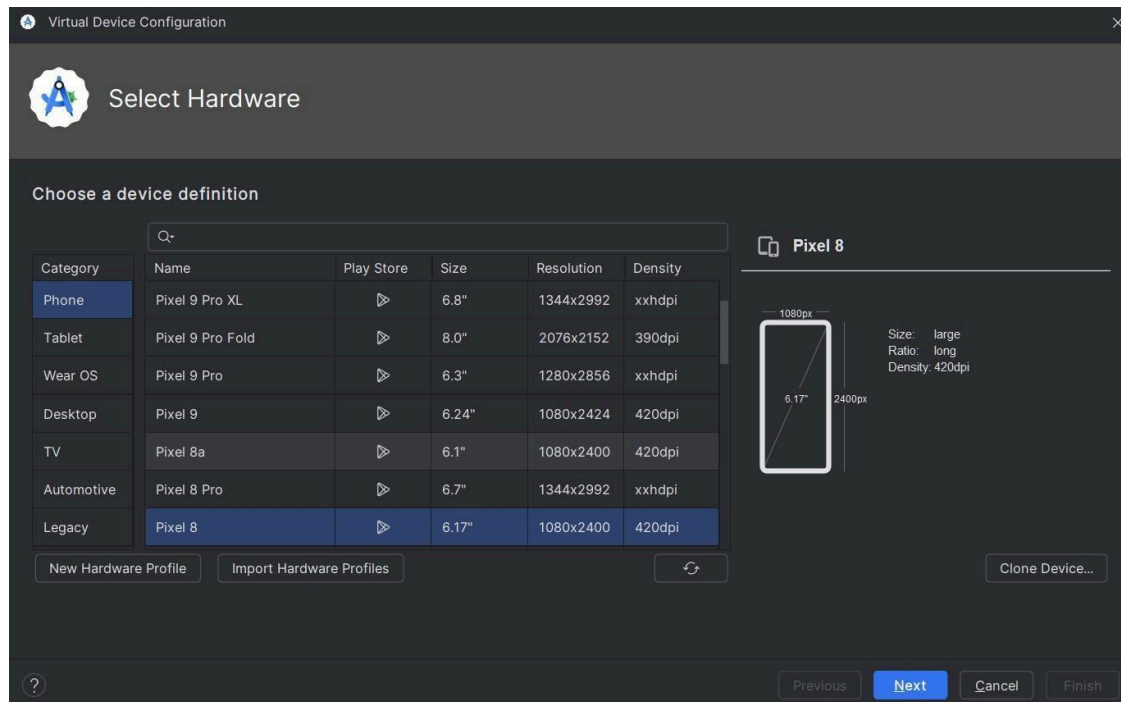
You have received two consent messages because the flutter tool is migrating to a new analytics system. Disabling
analytics collection will disable both the legacy and new analytics collection systems. You can disable analytics
reporting by running `flutter --disable-analytics`

C:\Users\INFT505-02>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.19045.5371], locale en-US)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[✓] Android toolchain - develop for Android devices (Android SDK version 33.0.1)
[✓] Chrome - develop for the web
[X] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[✓] Android Studio (version 2021.3)
[✓] VS Code (version 1.72.2)
[✓] Connected device (3 available)
[✓] Network resources

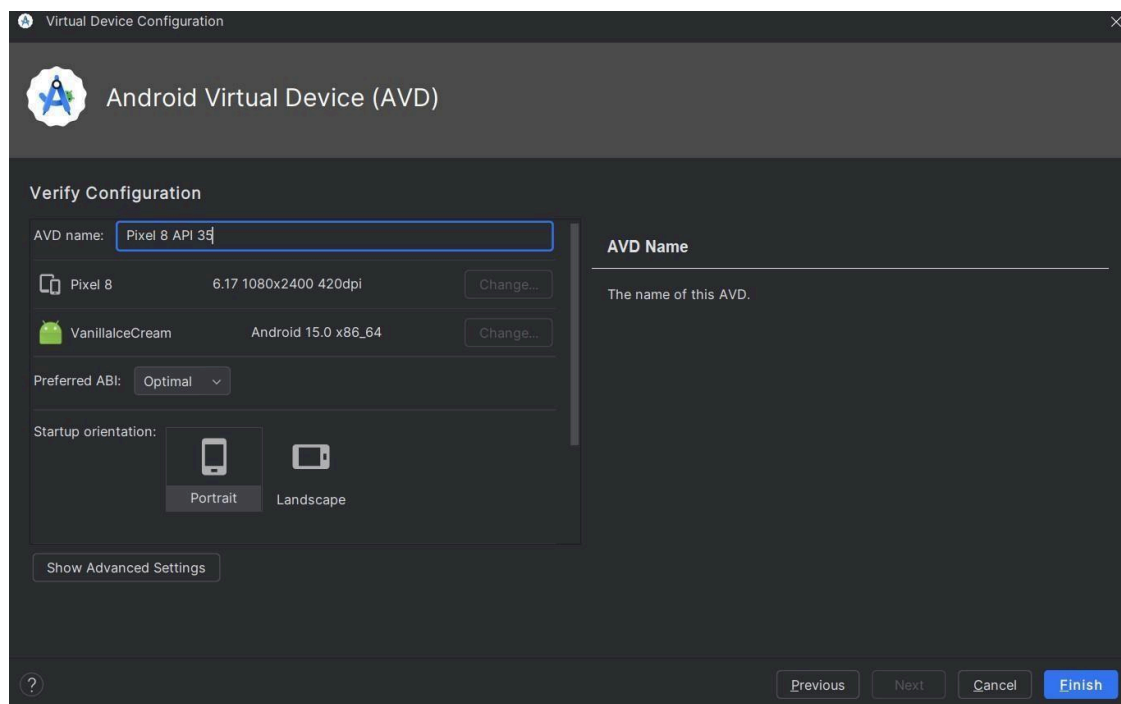
! Doctor found issues in 1 category.

C:\Users\INFT505-02>
```

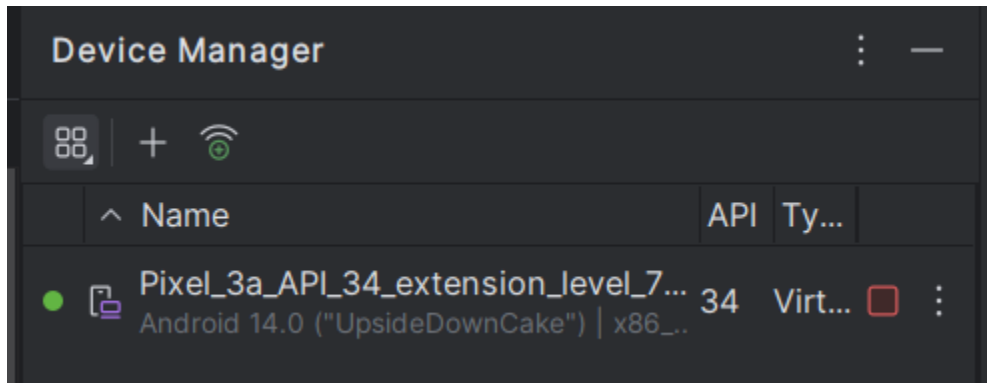
Step 10:- Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application



Step 10.1:- Open Android Studio and go to Tools > AVD Manager. Create a new virtual device.

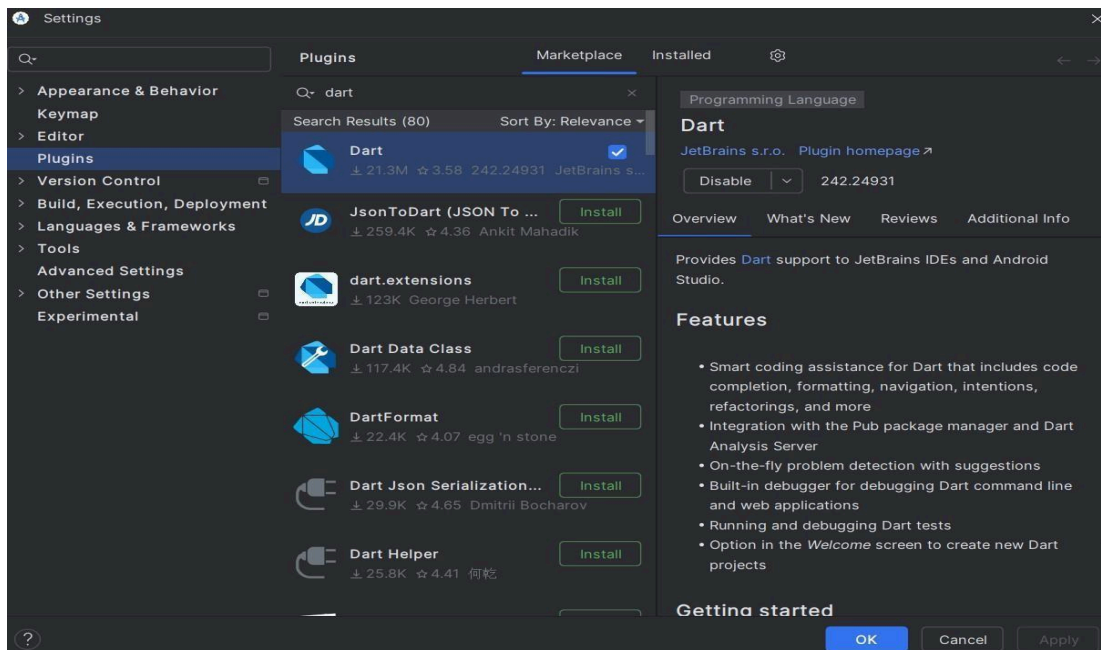
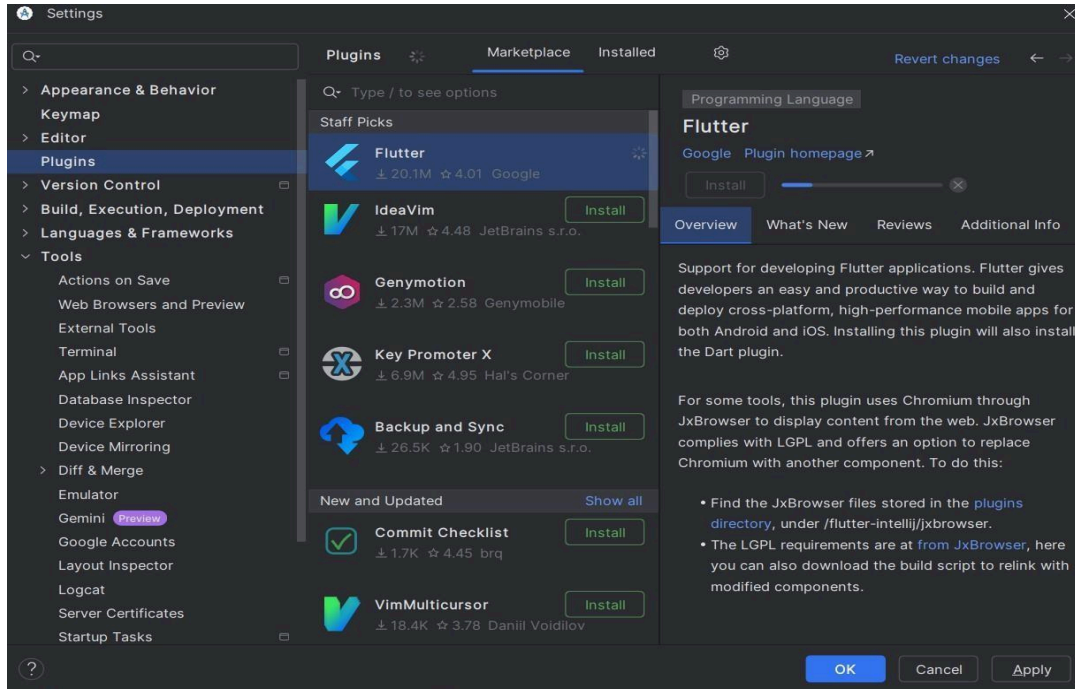


Step 10.2: - Click on the icon pointed into the red color rectangle. The Android emulator displayed as below screen



Step 11:- Now, install Flutter and Dart plugin for building Flutter application in Android Studio. These plugins provide a template to create a Flutter application, give an option to run and debug Flutter application in the Android Studio itself

Step 11.1:- Open the Android Studio and then go to File->Settings->Plugins. Now, search the Flutter plugin. If found, select Flutter plugin and click install



Step 11.2:- Restart the Android Studio

Step 12:- Go to File > New Project > Create Flutter Project, then select the project name and location, and click Next to proceed.

```

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

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

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

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // TRY THIS: Try running your application with "flutter run". You'll see
        // the application has a purple toolbar. Then, without quitting the app,
        // try changing the seedColor in the colorScheme below to Colors.green
        // and then invoke "hot reload" (save your changes or press the "hot
        // reload" button in a Flutter-supported IDE, or press "r" if you used
        // the command line to start the app).
        //
        // Notice that the counter didn't reset back to zero; the application
        // state is not lost during the reload. To reset the state, use hot
        // restart instead.
        //
        // This works for code too, not just values: Most code changes can be
        // tested with just a hot reload.
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyHomePage(title: 'Flutter Demo Home Page'),
    );
  }
}

```

```
}  
}
```

```
class MyHomePage extends StatefulWidget {  
  const MyHomePage({super.key, required this.title});  
  
  // This widget is the home page of your application. It is stateful, meaning  
  // that it has a State object (defined below) that contains fields that affect  
  // how it looks.  
  
  // This class is the configuration for the state. It holds the values (in this  
  // case the title) provided by the parent (in this case the App widget) and  
  // used by the build method of the State. Fields in a Widget subclass are  
  // always marked "final".  
  
  final String title;  
  
  @override  
  State<MyHomePage> createState() => _MyHomePageState();  
}  
  
class _MyHomePageState extends State<MyHomePage> {  
  int _counter = 0;  
  
  void _incrementCounter() {  
    setState(() {  
      // This call to setState tells the Flutter framework that something has  
      // changed in this State, which causes it to rerun the build method below  
      // so that the display can reflect the updated values. If we changed  
      // _counter without calling setState(), then the build method would not be  
      // called again, and so nothing would appear to happen.  
      _counter++;  
    });  
  }  
}
```

```

@override
Widget build(BuildContext context) {
  // This method is rerun every time setState is called, for instance as done
  // by the _incrementCounter method above.
  //
  // The Flutter framework has been optimized to make rerunning build methods
  // fast, so that you can just rebuild anything that needs updating rather
  // than having to individually change instances of widgets.
  return Scaffold(
    appBar: AppBar(
      // TRY THIS: Try changing the color here to a specific color (to
      // Colors.amber, perhaps?) and trigger a hot reload to see the AppBar
      // change color while the other colors stay the same.
      backgroundColor: Theme.of(context).colorScheme.inversePrimary,
      // Here we take the value from the MyHomePage object that was created by
      // the App.build method, and use it to set our appBar title.
      title: Text(widget.title),
    ),
    body: Center(
      // Center is a layout widget. It takes a single child and positions it
      // in the middle of the parent.
      child: Column(
        // Column is also a layout widget. It takes a list of children and
        // arranges them vertically. By default, it sizes itself to fit its
        // children horizontally, and tries to be as tall as its parent.
        //
        // Column has various properties to control how it sizes itself and
        // how it positions its children. Here we use mainAxisAlignment to
        // center the children vertically; the main axis here is the vertical
        // axis because Columns are vertical (the cross axis would be
        // horizontal).
        //
        // TRY THIS: Invoke "debug painting" (choose the "Toggle Debug Paint"
        // action in the IDE, or press "p" in the console), to see the
        // wireframe for each widget.

```

```

        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          const Text(
            'Hello Siddhant',
          ),
          Text(
            '$_counter',
            style: Theme.of(context).textTheme.headlineMedium,
          ),
        ],
      ),
    floatingActionButton: FloatingActionButton(
      onPressed: _incrementCounter,
      tooltip: 'Increment',
      child: const Icon(Icons.add),
    ), // This trailing comma makes auto-formatting nicer for build methods.
  );
}
}

```

Output:

Hello Siddhant
0

