

EXPERIMENT NO.1

Experiment No 1 1: Installation and Configuration of Flutter Environment.	
ROLL NO	35
NAME	Gaurang Milind Mapuskar
CLASS	D15-B
SUBJECT	MAD & PWA Lab
LO-MAPPE D	

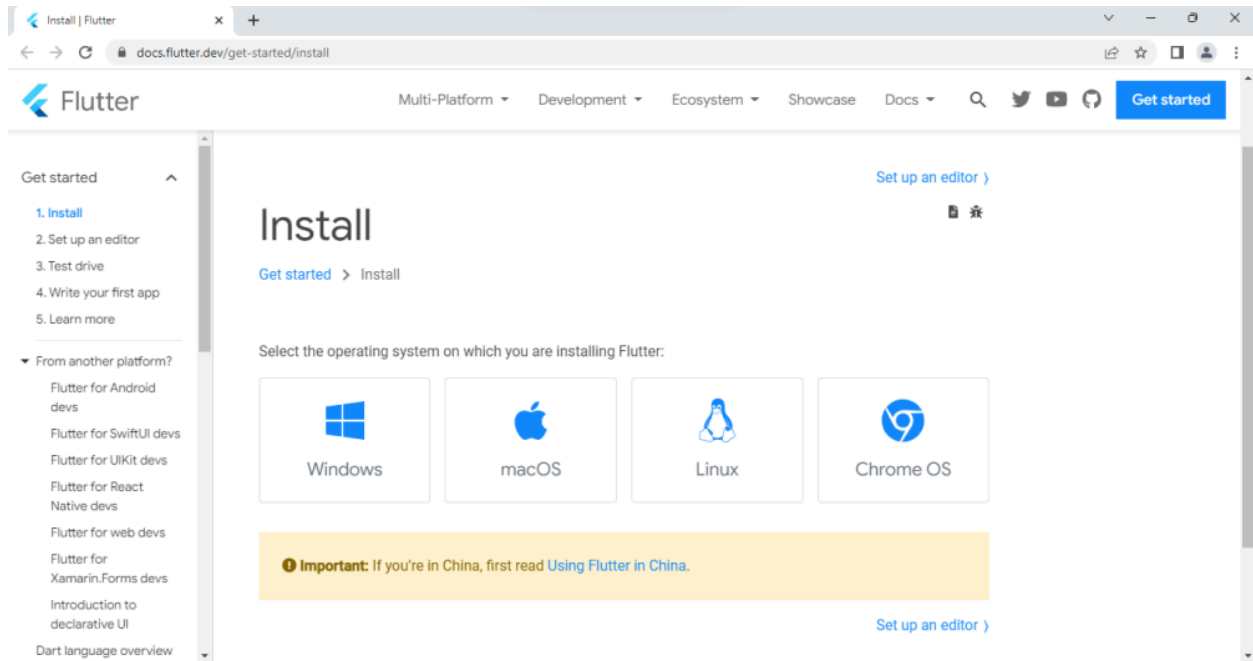
Theory:

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. Here are some of the key features and functionalities of Flutter:

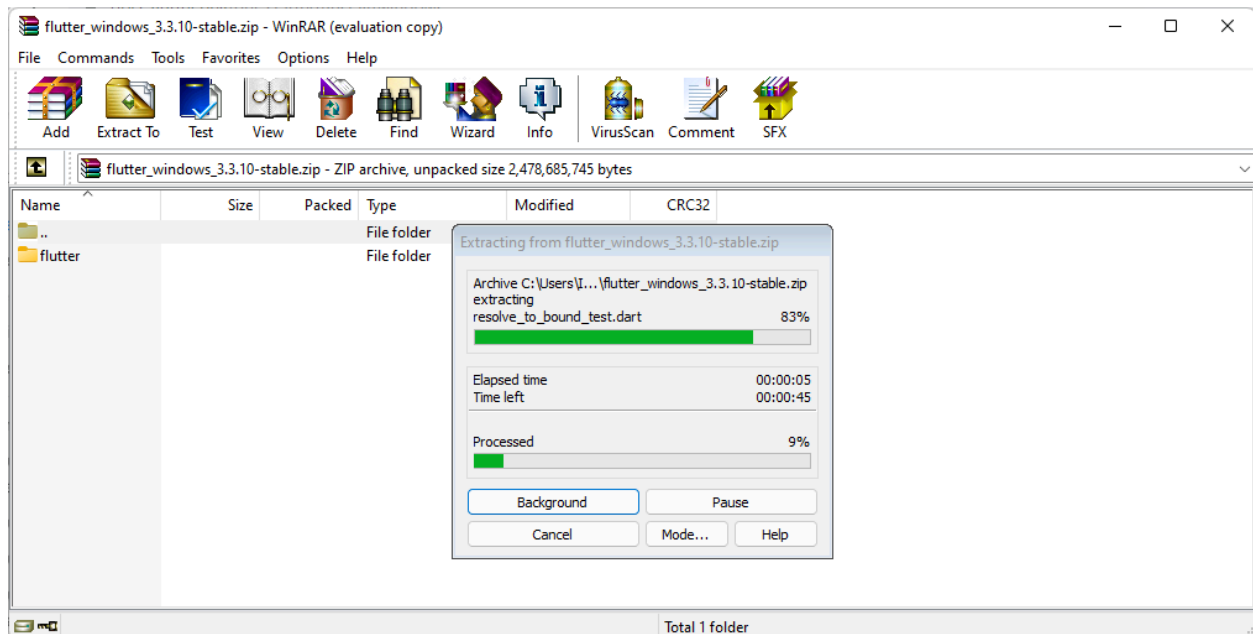
- **Single Codebase:**
Flutter allows developers to write code once and run it on multiple platforms, including iOS, Android, web, and desktop.
- **Rich Widget Library:**
Flutter comes with a comprehensive set of pre-designed and customizable widgets for creating modern and responsive user interfaces. These widgets are used to build the UI components, and developers can easily customize them to suit their app's design.
- **Expressive UI:**
Flutter allows for expressive and flexible UI designs. It supports various animations and provides a smooth and consistent experience across different platforms.
- **Dart Programming Language:**
Flutter uses Dart as its programming language. Dart is designed for building modern web and mobile applications and has features like strong typing and just-in-time compilation.
- **State Management:**
Flutter has a flexible state management system that allows developers to manage the state of their applications efficiently. There are various state management solutions available, catering to different preferences and project requirements.
- **Integration with Firebase:**
Flutter seamlessly integrates with Firebase, providing developers with a suite of backend services (such as authentication, database, and cloud functions) to enhance app functionality.
- **Performance:**
Flutter apps are compiled into native ARM code for optimal performance. The framework's architecture and use of the Skia graphics engine contribute to high-performance applications.

Flutter Installation:

Step 1: Install the Flutter SDK Download the Flutter Software Development Kit from the official website for Windows.



Step 2: Once the download is complete extract the zip file and place it in the desired folder.



Step 3: To run the Flutter command in the regular windows console, you need to update the system path to include the flutter bin directory. The following Steps are required to do this: Go to THIS PC -> Properties -> Advanced system settings -> Environment variables.

Step 4: Select the Path -> click on Edit and add the path to the Flutter bin folder.

Step 5: Run the flutter command in the Command Prompt.

```
Command Prompt - flutter  X  +  v
C:\Users\ASUS>flutter

A new version of Flutter is available!
To update to the latest version, run "flutter upgrade".

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.
--suppress-analytics       Suppress analytics reporting for the current CLI invocation.
--disable-telemetry        Disable telemetry reporting when this command runs.

Available commands:

Flutter SDK
bash-completion  Output command line shell completion setup scripts.
channel          List or switch Flutter channels.
config           Configure Flutter settings.
doctor           Show information about the installed tooling.
downgrade        Downgrade Flutter to the last active version for the current channel.
precache         Populate the Flutter tool's cache of binary artifacts.
upgrade          Upgrade your copy of Flutter.

Project
analyze          Analyze the project's Dart code.
assemble         Assemble and build Flutter resources.
build            Build an executable app or install bundle.
clean            Delete the build/ and .dart-tool/ directories.
create           Create a new Flutter project.
drive            Run integration tests for the project on an attached device or emulator.
gen-l10n         Generate localizations for the current project.
pub             Commands for managing Flutter packages.
run              Run your Flutter app on an attached device.
test            Run Flutter unit tests for the current project.

Tools & Devices
attach           Attach to a running app.
custom-devices  List, reset, add and delete custom devices.
devices         List all connected devices.
emulators       List, launch and create emulators.
install         Install a Flutter app on an attached device.
logs            Show log output for running Flutter apps.
screenshot      Take a screenshot from a connected device.
symbolize       Symbolize a stack trace from an AOT-compiled Flutter app.

Run "flutter help <command>" for more information about a command.
Run "flutter help -v" for verbose help output, including less commonly used options.
```

Also, 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.

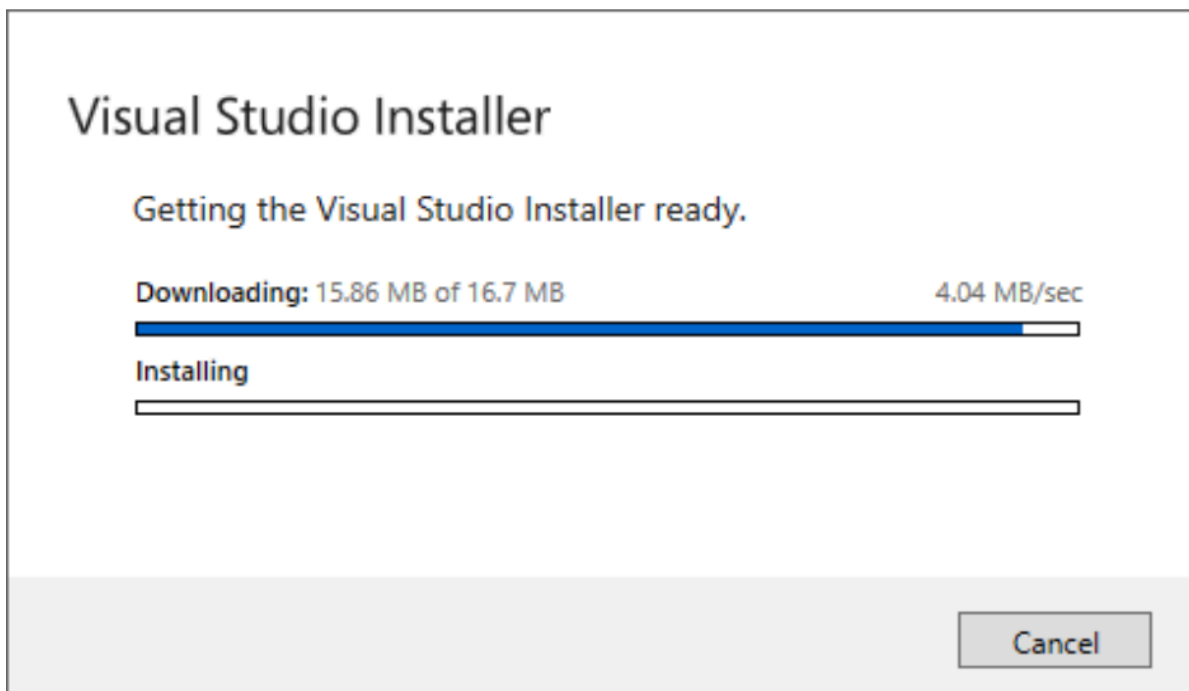
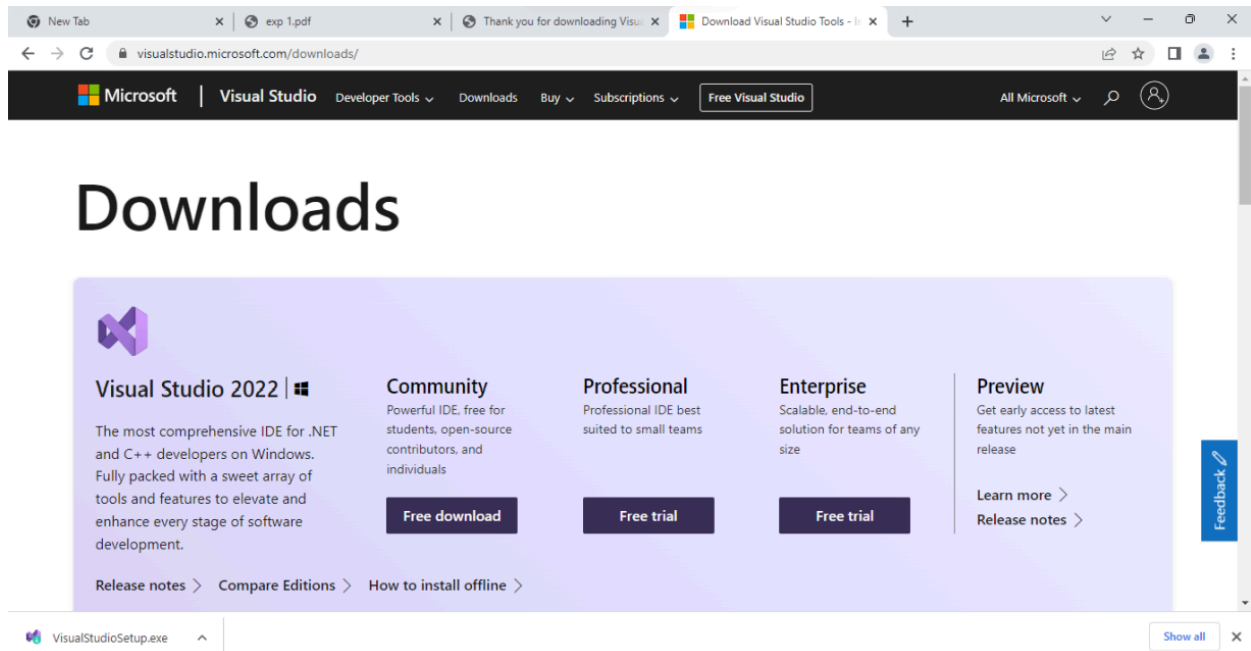
```
C:\Users\ASUS>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.10.5, on Microsoft Windows [Version 10.0.22621.3007], locale en-US)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[✓] Android toolchain - develop for Android devices (Android SDK version 33.0.2)
[✓] Chrome - develop for the web
[✓] Visual Studio - develop for Windows (Visual Studio Community 2022 17.5.0)
[✓] Android Studio (version 2022.1)
[✓] VS Code (version 1.85.1)
[✓] Connected device (3 available)
[✓] Network resources

• No issues found!

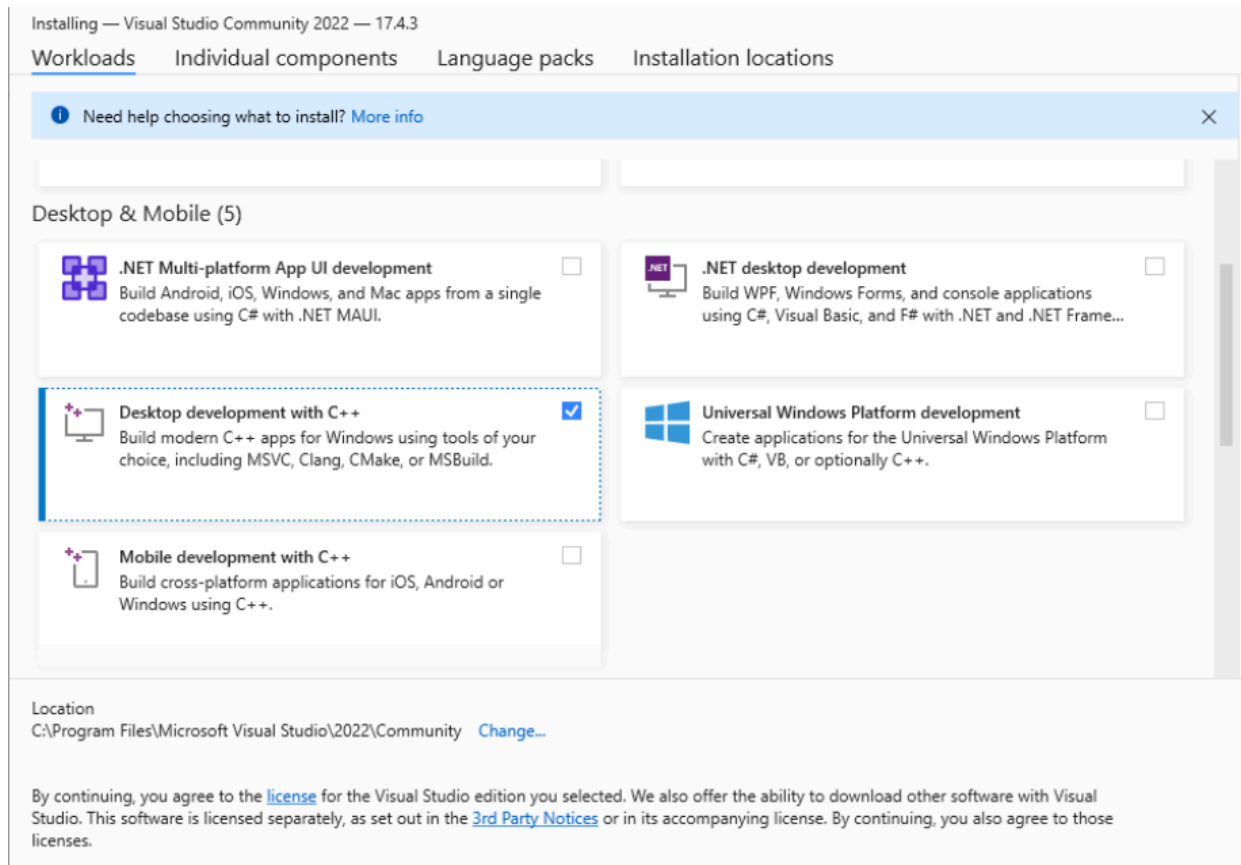
C:\Users\ASUS>
```

Step 6: Now we will resolve all the issues and install and add the missing tools required for Flutter app development.

Step 6.1: Download and install Visual Studio.

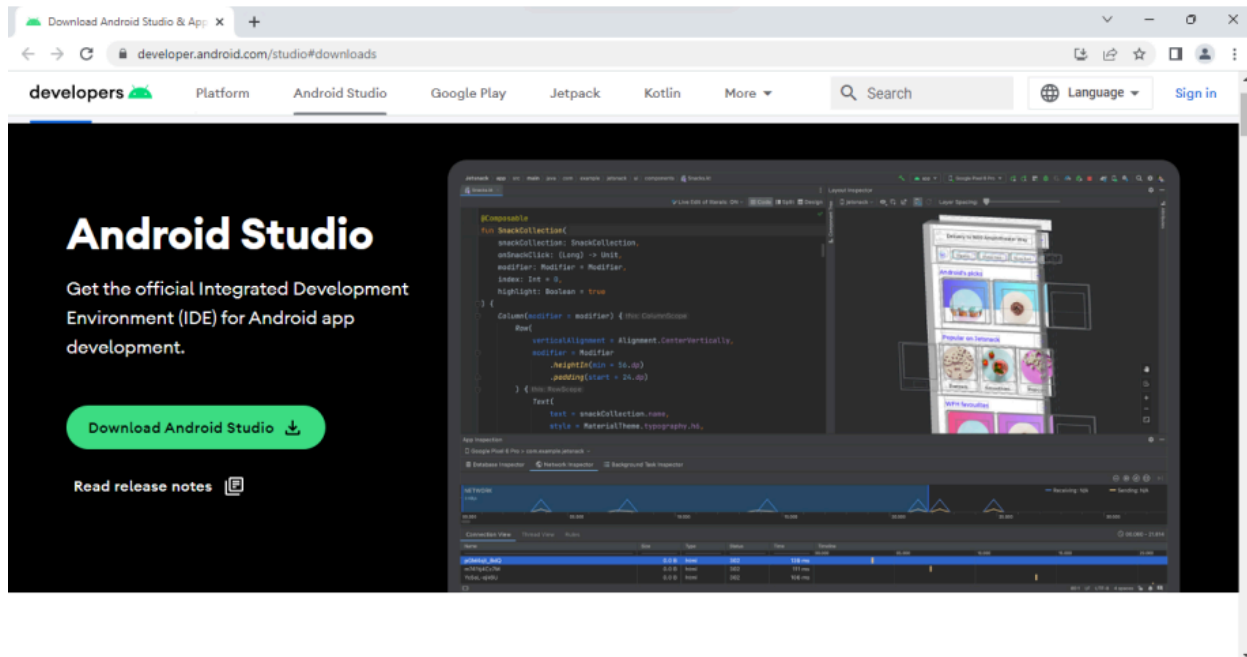


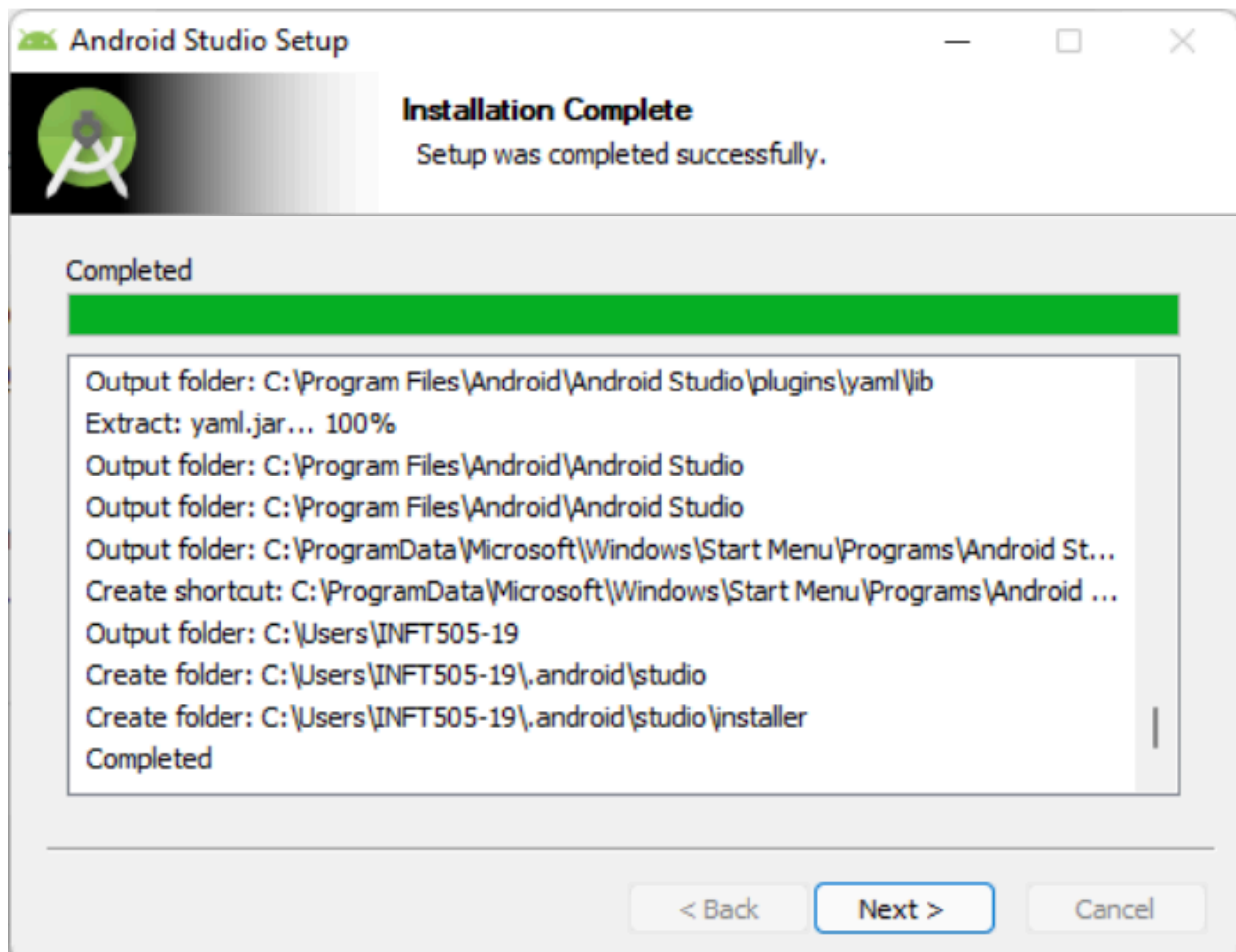
While Installing Visual Studio, make sure to select Desktop development with C++.



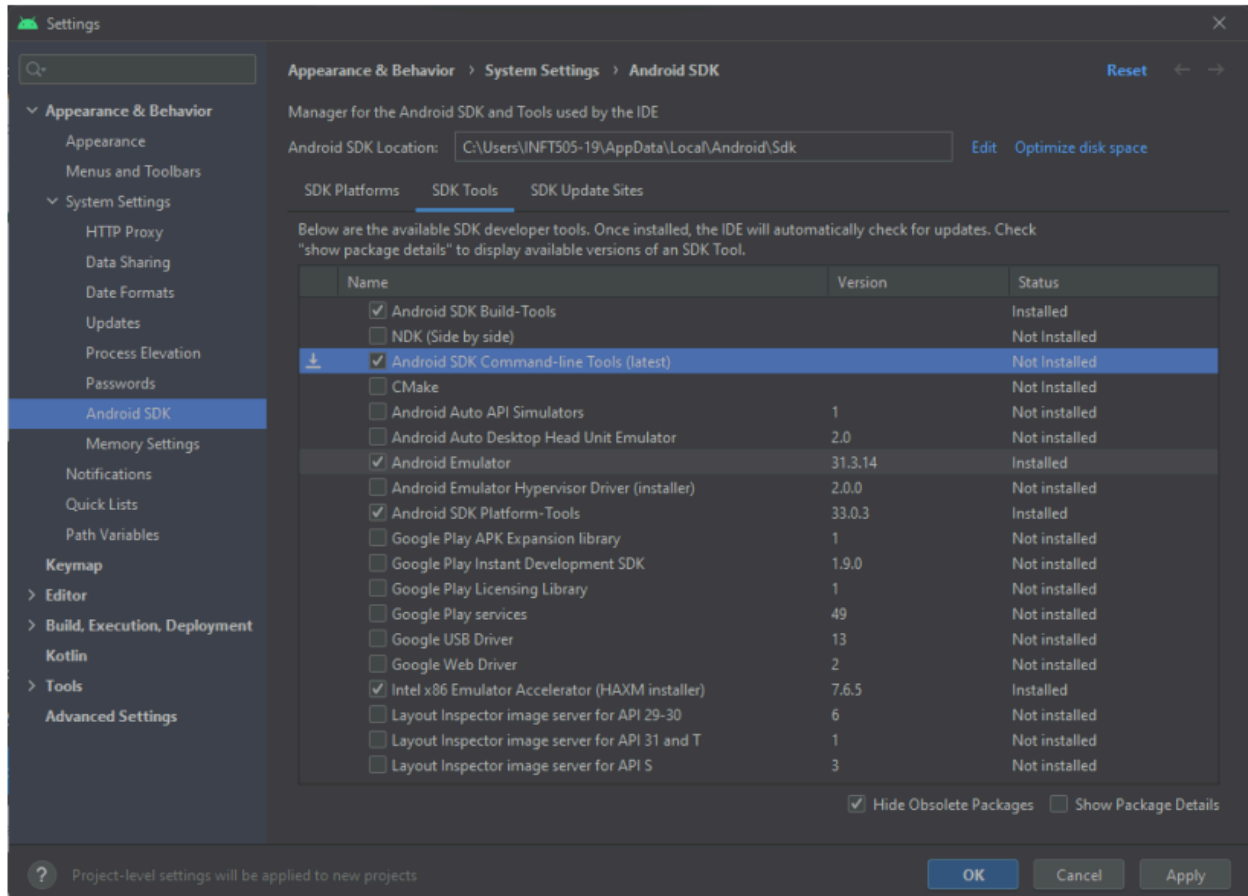
We can see that the Visual Studio issue has been solved.

Step 6.2: Install the Android SDK. If the flutter doctor command does not find the Android SDK tool in your system, then you need first to install the Android Studio IDE.



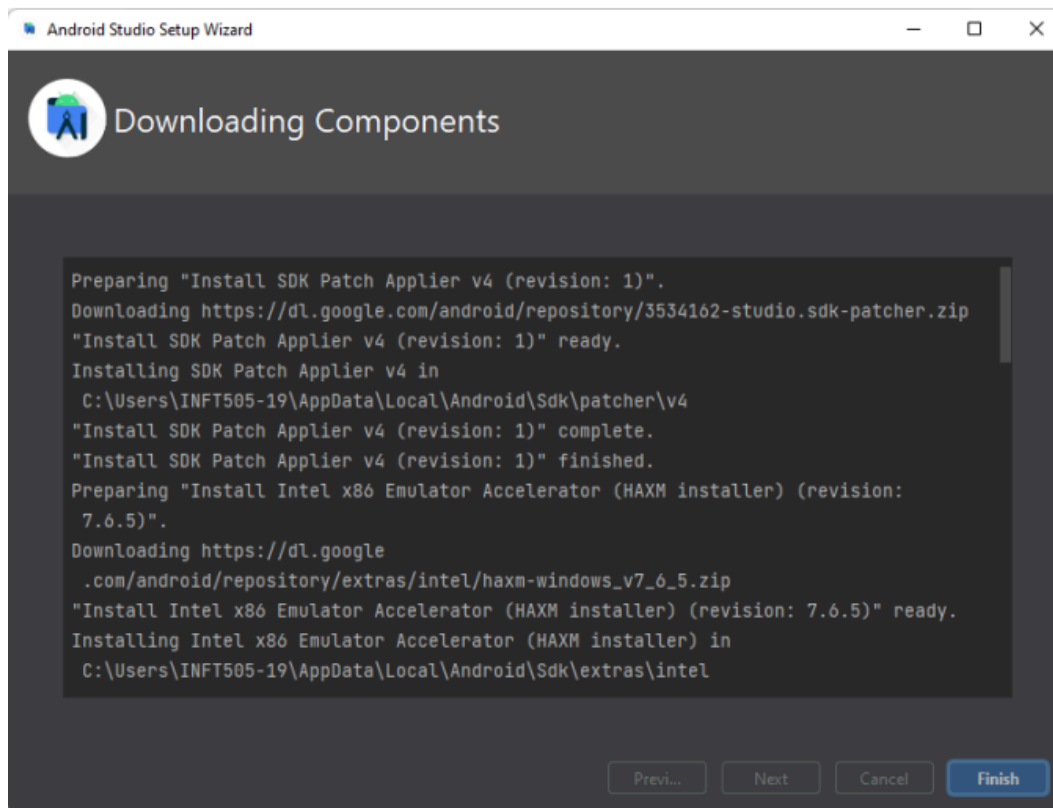
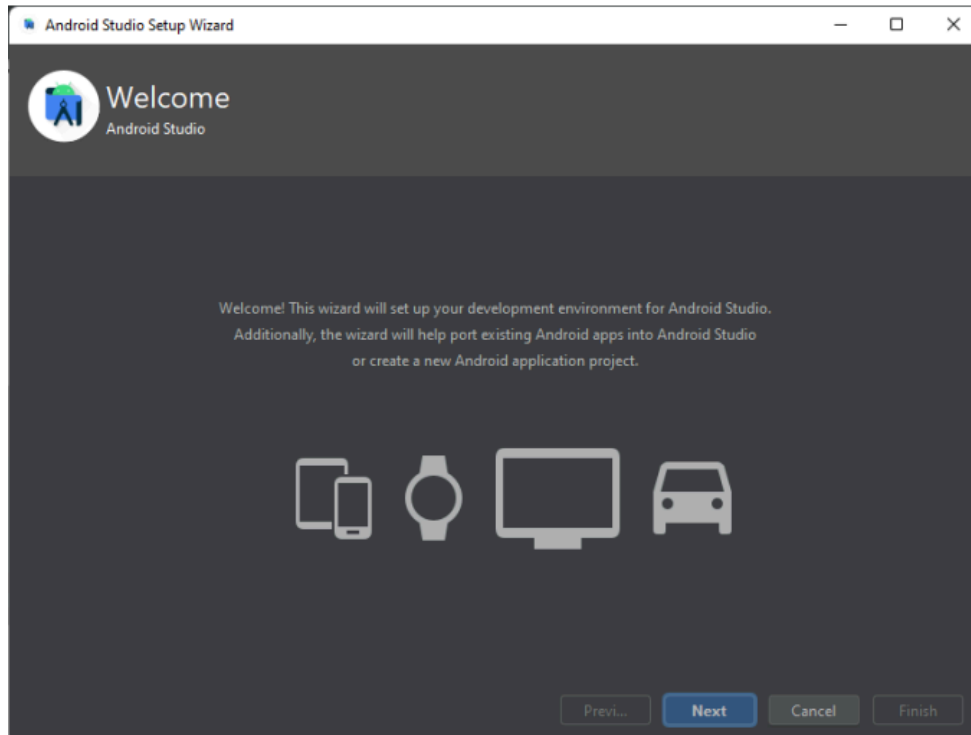


Once the android Studio is installed, Download the Android SDK Command-Line Tools present in the Android SDK section in Android Studio.

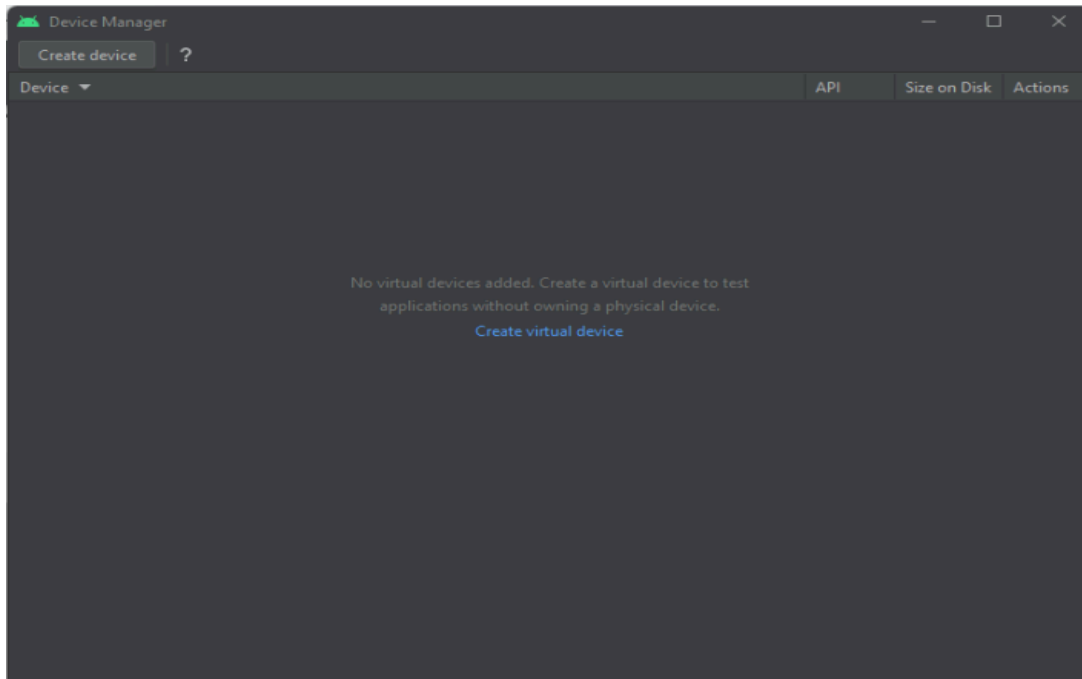


Run the flutter doctor command and run flutter doctor –android licenses to accept all the required licenses.

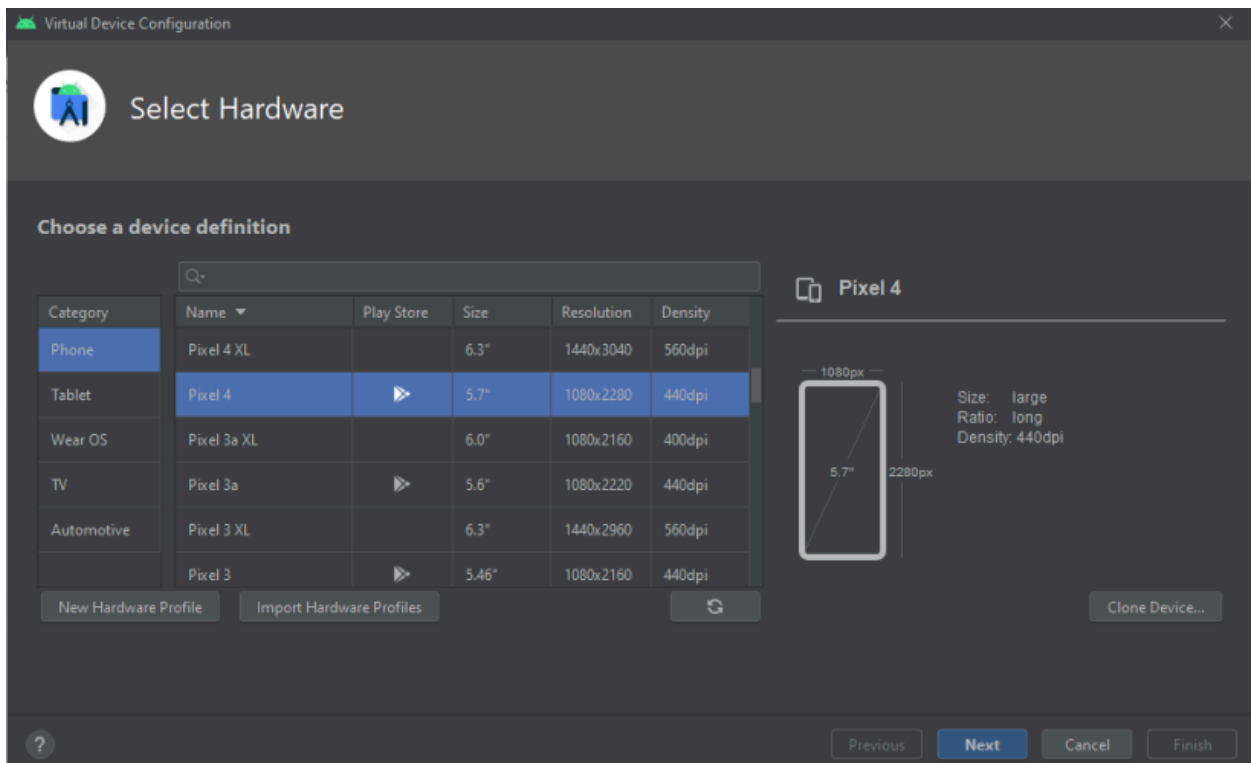
Step 7: Next, you need to set up an Android emulator. It is responsible for running and testing the Flutter application.



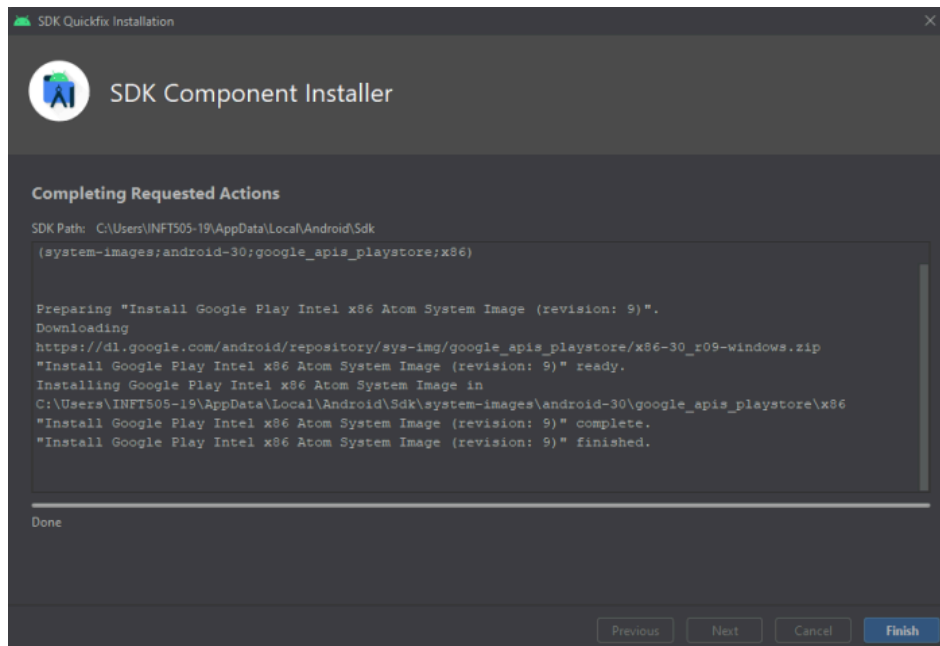
Step 7.1: To set an Android emulator, go to Android Studio > Tools > SDK Manager and select Create Virtual Device. You will get the following screen:



Step 7.2: Choose your device definition and click on Next.



Step 7.3: Select and download the latest operating system for our Emulator and click on Finish.

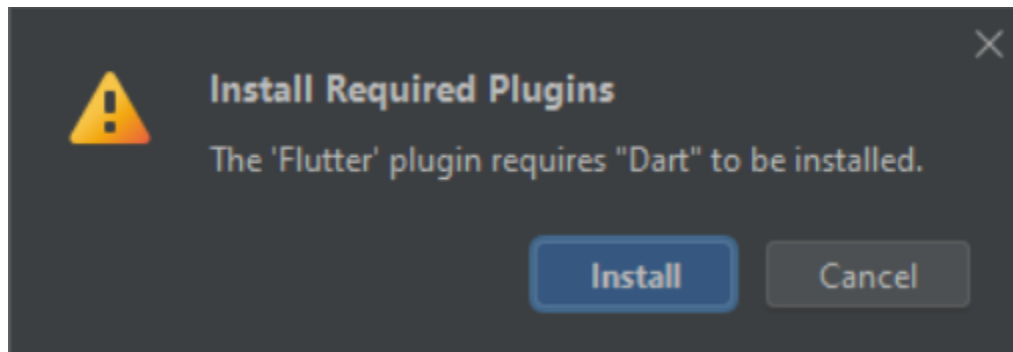


Step 7.4: Click on the Run button and the following screen will be displayed.



Step 8: Now, install Flutter and Dart plugins for building the Flutter application in Android Studio. These plugins provide a template to create a Flutter application and give the option to run and debug the Flutter application in the Android Studio itself.

Open the Android Studio and then go to File->Settings->Plugins. Now, search the Flutter plugin. If found, select the Flutter plugin and click install. When you click on install, it will ask you to install Dart plugin as below screen. Click Install to proceed.



Finally when all these Steps are followed restart the Android Studio once and then your Flutter environment is successfully configured.

\

Conclusion:

In conclusion, the successful installation of Flutter in this lab experiment demonstrates its ease of setup. This versatile framework empowers developers to create cross-platform applications efficiently, fostering innovation in mobile app development