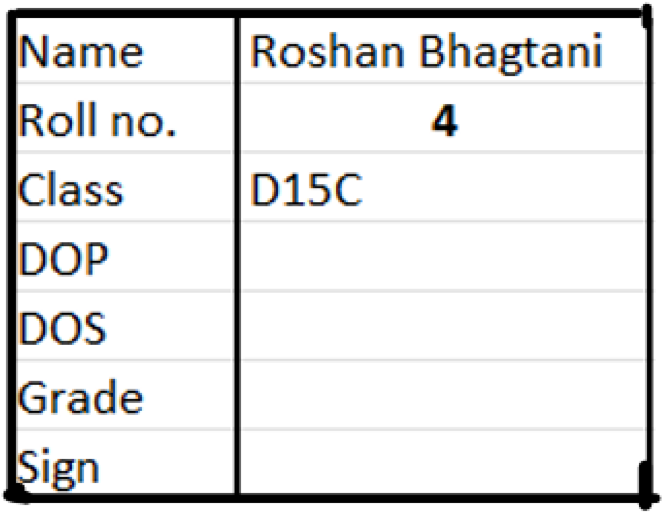
Experiment 01



Name	Roshan Bhagtani
Roll no.	**4**
Class	D15C
DOP	
DOS	
Grade	
Sign	

Aim: Installation and Configuration of Flutter Environment.

Theory:

Flutter is an open-source UI software development toolkit created by Google, used for building natively compiled applications for mobile, web, and desktop from a single codebase. It uses the Dart programming language and provides a rich set of pre-built widgets, making UI development faster and more consistent across platforms.

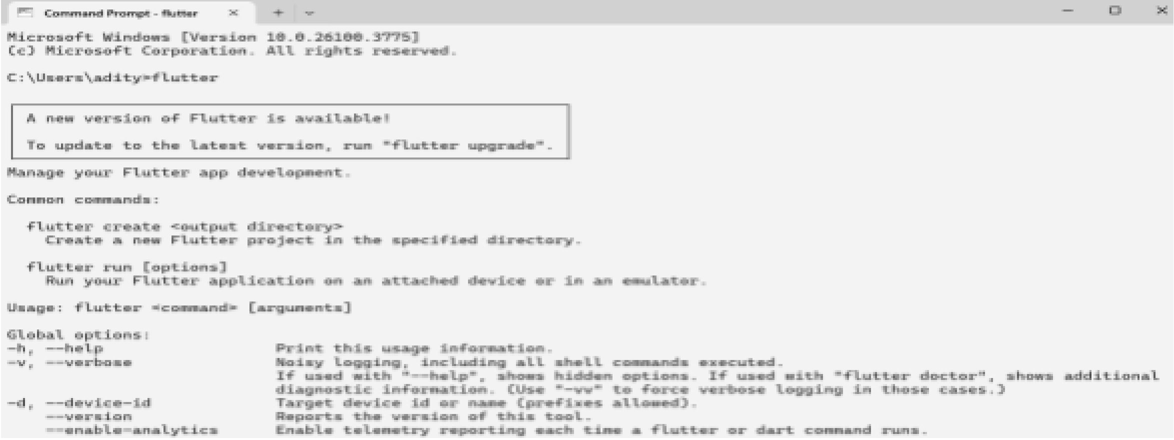
To start developing with Flutter, the first step is to **install the Flutter SDK** on your system. Flutter supports all major operating systems including Windows, macOS, and Linux. The SDK includes all the tools necessary for developing and compiling Flutter applications.

Once the SDK is downloaded and extracted, the **system path** must be configured to include the Flutter binary so that Flutter commands can be run from any terminal window. After setting up the path, you can run the `flutter doctor` command to check for any missing dependencies. This command verifies the environment setup, checks for required tools like Dart, Android Studio, and Android SDK, and ensures everything is properly configured.

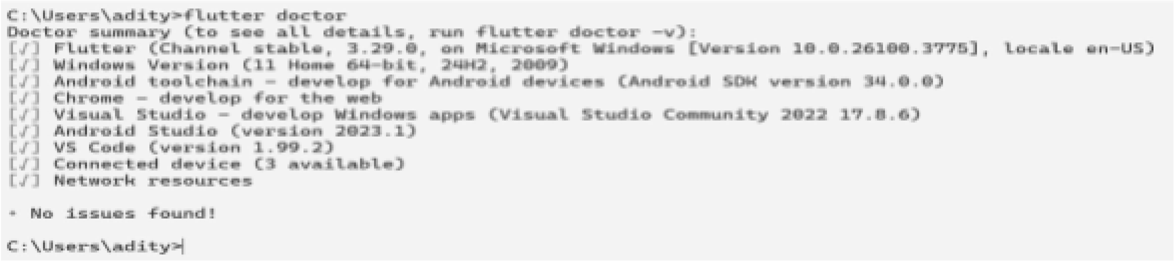
Next, an **IDE (Integrated Development Environment)** like **Android Studio**, **Visual Studio Code**, or **IntelliJ IDEA** should be installed. These IDEs provide Flutter plugins that offer features like hot reload, debugging tools, and widget trees for easier app development.

Finally, to test apps on a physical device or emulator, the **Android SDK** and **AVD (Android Virtual Device)** must be configured within Android Studio. Once everything is set up, a simple `flutter run` command can be used to run the app on a connected device or emulator.

Output:



```
Microsoft Windows [Version 10.0.26100.3775]
(c) Microsoft Corporation. All rights reserved.

C:\Users\adity>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.
    --enable-analytics      Enable telemetry reporting each time a flutter or dart command runs.
```



```
C:\Users\adity>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.29.0, on Microsoft Windows [Version 10.0.26100.3775], locale en-US)
[√] Windows Version (11 Home 64-bit, 24H2, 2009)
[√] Android toolchain - develop for Android devices (Android SDK version 34.0.0)
[√] Chrome - develop for the web
[√] Visual Studio - develop Windows apps (Visual Studio Community 2022 17.8.6)
[√] Android Studio (version 2023.1)
[√] VS Code (version 1.99.2)
[√] Connected device (3 available)
[√] Network resources

• No issues found!

C:\Users\adity>
```

Conclusion:

Installing and configuring the Flutter environment is a crucial first step for any developer aiming to build cross-platform applications efficiently. With the Flutter SDK, Dart, and a well-configured IDE like Android Studio or Visual Studio Code, developers can create high-performance apps for Android, iOS, web, and desktop from a single codebase. The use of tools like `flutter doctor` ensures that all components are correctly set up, reducing setup issues and streamlining the development process.