Experiment 03

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

Aim: To include icons, Prerequisite, fonts in Flutter app

Theory:

In Flutter, integrating icons, prerequisites, and custom fonts involves several steps that enhance the visual appeal and functionality of your app. Icons are an essential part of the user interface as they provide intuitive, visual representations of actions and content. Flutter offers a wide range of built-in icons via the `Icons` class, which can be used directly in widgets like `Icon`, `IconButton`, or `FloatingActionButton`. These built-in icons cover a vast range of common use cases. However, if you need custom icons, you can include external icon packs such as Font Awesome or use custom SVG or image files. To add custom icons, you typically need to use packages like `flutter_svg` or `font_awesome_flutter`, which you can include in the `pubspec.yaml` file by adding their respective dependencies. After adding the dependencies, you import the necessary package in your Dart files and use the custom icons as widgets.

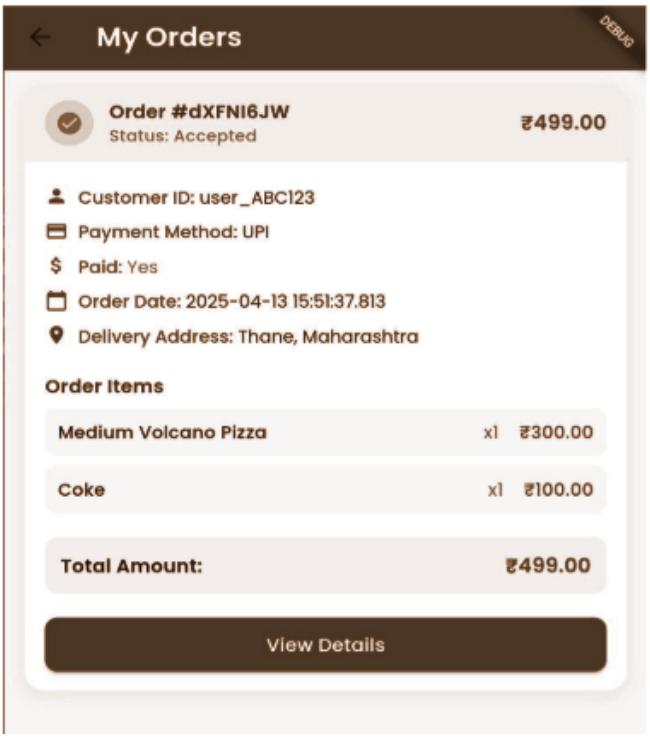
When adding icons, it's important to ensure that the required icon fonts or SVG files are added to your project's assets folder, and you need to declare these assets in the `pubspec.yaml` file. This ensures that your app can access and render the icons properly. For custom fonts, you can include them in your Flutter project by placing the font files in the `assets/fonts` folder and then specifying the fonts in the `pubspec.yaml` under the `fonts` section. After doing so, you can apply the custom fonts globally or locally using the `TextStyle` widget, improving the overall design consistency and user experience. These steps make your app more customizable and visually appealing by providing access to a wide array of design resources like icons and fonts.

Implementation:

The font "Poppins" is being used as a prerequisite via the **Google Fonts** package. This package allows you to easily apply custom fonts in your Flutter app. Specifically, the `GoogleFonts.poppins()` method is being used to style text elements throughout the application, such as in the `Text`, `RadioListTile`, `AlertDialog`, and `TextField` widgets.
The "Poppins" font is applied to enhance the visual appearance of the text by specifying the font family in various UI components.

For this to work properly, you need to include the **Google Fonts** package in your `pubspec.yaml` file under dependencies. This package fetches fonts directly from Google Fonts and makes them available for use within your Flutter app. The `google_fonts` package simplifies integrating custom fonts like Poppins and allows you to access a wide variety of fonts beyond the default ones available in Flutter.

Output:



Conclusion:

Integrating custom fonts such as "Poppins" in a Flutter application enhances the visual appeal and user experience by providing more flexibility in UI design. By using the **Google Fonts** package, you can easily include and manage fonts without worrying about manual font files or asset management. The code demonstrates how to apply the "Poppins" font to various text-based widgets, such as `Text`, `RadioListTile`, and `TextField`, ensuring consistency across the app's interface.