Experiment 2	
Name	Chinmay Chaudhari
Rollno	6
Class	D15C
DOP	
DOS	
Sign	
Grade	

AIM: - To design Flutter UI by including common widgets.

Description

Flutter is a UI toolkit that empowers developers to create cross-platform applications with interactive and aesthetically pleasing designs. The foundation of Flutter's UI design lies in its widget-based architecture. Widgets are classified into two main types: **Stateful Widgets** and **Stateless Widgets**, which govern the behavior and appearance of UI components.

Stateless Widgets

Stateless widgets are immutable, meaning their properties cannot change during runtime. These widgets do not store any state and are ideal for static UI components such as text, images, or icons. Examples include `Text`, `Icon`, and `Container`. They serve the purpose of displaying information that does not rely on dynamic updates.

Stateful Widgets

Stateful widgets, on the other hand, are mutable and maintain dynamic states. Their properties can change based on user interaction or application logic during runtime. Stateful widgets include an internal mechanism to manage state changes using `State` objects. Examples are `Checkbox`, `Slider`, and `TextField`. These widgets are crucial for creating interactive elements like forms, animations, or real-time data updates.

Common Widgets

- **Structural Widgets:** `Container`, `Row`, and `Column` are used to define the layout structure of the UI.
- **Interactive Widgets:** `Button`, `Switch`, `Slider`, and `TextField` allow user interactions.
- **Decorative Widgets:** `Image`, `Icon`, and `Text` enhance the visual appeal.

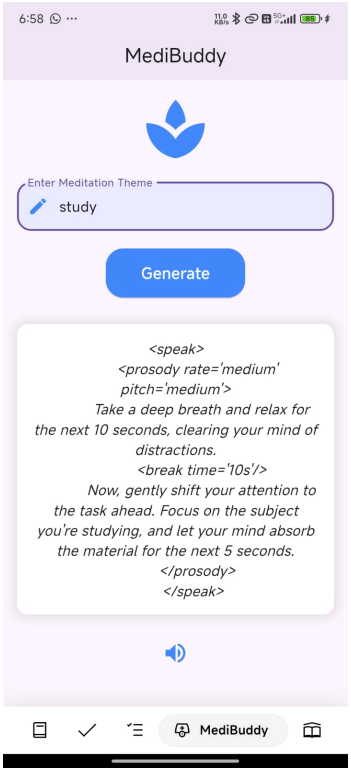
- **Input Widgets:** Widgets like `TextField` enable data input from users.
- **Gestures:** Widgets such as `GestureDetector` handle user interactions like taps, swipes, or pinches.

State Management

State management plays a vital role in handling dynamic updates. Popular approaches include:

- **Provider** and **Riverpod**: Simplify state management and dependency injection.
- **Bloc (Business Logic Component)**: Structures logic and states effectively for complex applications.

Output:



GitHub Link - [medi_buddy_view.dart]

Conclusion:

Designing Flutter UI with common widgets offers a hands-on experience in building high-quality interfaces that balance functionality and aesthetics. By understanding the distinction between stateless and stateful widgets, along with leveraging Flutter's rich widget library, developers can craft responsive and engaging applications. The inclusion of state management and platform-specific styles ensures that applications are not only interactive but also consistent across different operating systems. This experiment lays a strong foundation for creating intuitive and user-centric designs, unlocking the potential of Flutter in modern app development.