Name: Soham H. Nimbalkar
Roll no: 49 D15B.

Assignment-01

Q1>

a) Key features

* Single Codebase :-
Flutter allows developers to write code once & deploy it on both ios & android platforms. This reduces development time & effort Compared to maintaining different codebases for each platform.

* Hot Reload: one of flutter standard features is Hot Reload, which allows developers to instantly see the results of code changes without restarting the entire application. This significantly speeds up the development process and facilitates quick iterations.

* Dart Programming language :-
Flutter uses Dart programming language which is designed for building web, mobile & server applications. Dart's Syntax is easy to learn for developers coming from various programming backgrounds.

Advantages:-

* Faster development :-
The ability to use a Single Codebase & the Hot Reload feature significantly speeds up the development process. This allows developers to iterate quickly, fix bugs on the ~~stay~~ fly.

* Cost-effective :- Developing & maintaining a Single Codebase for multiple platforms reduces development cost.

Q1> b) Flutter differs from traditional approaches to mobile app development in several ways & its unique features contribute to its popularity in the developer community.

① Single Codebase for multiple platforms.

* Traditional approach :- In traditional app development, developers need to maintain Separate Codebase for ios and android.

* Flutter's approach :- With flutter, developers write Code once & deploy it on both ios & android platforms. This significantly reduces the overhead of maintaining multiple Codebases & accelerates the development process.

② Expressive UI with Custom widgets :- Flutter provides a rich set of Customisable widgets that enable developers to create visually appealing and expressive user interfaces, custom animations & transitions can be implemented easily.

Q2) a) Widget Tree

* Hierarchy of widgets :- The widget tree is a hierarchical Structure where each widget Serves a specific purpose and can contain other widgets. The tree represents the entire user interface, starting from the root widget & branching out into various Subwidgets.

* Immutable & reusable :-
widgets in flutter are immutable, meaning they cannot be changed once created when a change is created, a new widget is created.

* Widget Composition :-
Building blocks :- flutter encourages a Composition-based approach to UI Construction. Developers build complex interfaces by Combining & nesting simple widgets to create more sophisticated structures.

* Reusable Component :- widgets are designed to be reusable. Developers can encapsulate specific functionality or design elements into custom widgets. These custom widgets can then be used in multiple parts of the application, promoting code reusability.

Q 2.b) 1> Container :-
⇒ The 'Container' widget is a versatile box model that can ~~either~~ Contain other widgets. it is often used for layout purpose & provides properties for customization such as padding, margin.

eg :-
```
Container (
    Padding : EdgeInsets.all(16.0),
    margin : EdgeInsets.Symmetric(vertical:8.0),
    decoration : BoxDecoration(
        Color : Colors.blue,
        borderRadius : BorderRadius.circular(10.0),
    ),
    child : Text('This is a Container'),
)
```

2) Row & Column:

⇒ These widgets allow for the arrangement of child widgets horizontally (Row) or Vertically (Column). They are fundamentals for creating flexible layouts.

```
Row (
  children: [
    Icon (Icons.star),
    Text ('Text');
  ],
);
```

```
Column (
  children: [
    Text ('Title'),
    Text ('Subtitle'),
  ],
)
```

Q3) State management in flutter is crucial for :-

⇒ ① Responsiveness : updates UI based on user interactions & dynamic data change.

② Hot Reload :- Facilitates efficient development with flutters Hot reload features.

③ Dynamic UI : Enables dynamic content updates & responsiveness to external factors.

④ Form & input handling : Manages input validation, error messages & form submissions.

⑤ Performance optimization :- avoids unnecessary widget rebuilds for improved performance.

(Q3b) 1> SetState :-

• Characteristics :-

→ Built-in flutter method

→ Simple & suitable for local state within a widgets.

→ limited to the scope of a single widget.

• Use Cases :-

→ Suitable for small to medium-sized applications with limited shared state.

→ Quick & straightforward implementation for managing local UI-related state.

2> Provider :-

• characteristics :-

→ External package ('Provider') offering a variety of state management solutions.

→ Supports different types of state, including primitive, object & stream-based state.

• Use Cases :-

→ ~~Sh~~ Shared state across multiple widgets.

→ ~~More~~ Managing dependency injection efficiently.

→ When a lightweight, straightforward solution is preffered.

3) Riverpod :-

- Characteristics :-

* Advanced State management Solution built on top of provider.
* extends the functionality of provider, offering features like dependency injection
* Supports both Synchronous & asynchronous State changes.

- Use Cases:

* Complex applications with a need for Scalability & maintainability
* when dependency injection is a crucial part of the ~~State~~ State management strategy.
* Project that can benefit from a more feature rich provider-like Solution.