Name:- Ajay Deshmukh
Div :- D15B
Roll no:- 11

MPL
Assignment No:01

DOS :- 28/2/2025

1) Explain the key features and advantage of using Flutter for mobile app development

Ans:- Flutter is an open-source UI framework developed by google for building natively compiled applications for mobile, web, and desktop from a Single Codebase.

*key features:-

1) Single Codebase:- Write one codebase and deploy it on Android, ios, web and Desktop.
2) Hot Reload:- Instantly see changes made in the code without restarting the app.
3) Widget-Based UI:- Everything in flutter is a Widget, making UI creation flexible and reusable.
4) Fast Performance:- Uses Dart's ahead-of-Time Compilation for high performance.
5) Customizable UI:- Rich set of material and cupertino Widgets for Seamless Android and ios design.
6) Strong Community & Google Support:- Regular updates and Strong community backing.

*Advantages:-

1) Fast Development:- Hot reload and a rich set of pre-built widgets speed up development
2) Consists UI Across Platform:- No need for Platform Specific UI coding.
3) Lower Development Cost:- A single team can handle multiple platforms.
4) Better Performance:- No reliance on Javascript bridges leading to smooth UI interactions.

Teacher's Sign:

2.1.b) How Flutter Differs from Traditional Approach & its Properties

→ Ans:- * Tradition Approach (Native Development)
→ Android :- Uses Java/Kotlin and XML for UI
→ Ios :- Uses Swift/objective C with storyboards or Sw
→ Separate Codebases :- Developers write and maintain Separa code for different platforms

* Flutter's Approach :-
→ Uses Dart for both frontend and backend logic
→ Employs a widget-based UI rather than XML-based design
→ Single codebase for both Android and Ios.
→ Own Rendering Engine (skia), eliminating platf dependency.

* Reasons for Flutter's Popularity
1) Faster Development with Hot Reload
2) Cross-Platform Development without Performance Trade-offs
3) Rich UI Capabilities with Pre-built widgets.
4) Backed by Google & strong Open-source Community.
5) Growing Adoption in Enterprises & startups.

Teacher's Sign.:

Concept of Widget Tree & Widget Composition in Flutter.

Ans:- Widget Tree in Flutter

-> In Flutter, everything is a widget (buttons, text, images, Containers etc)

-> These widgets are arranged in a tree-like structure called the widget Tree.

-> The tree structure determines the layout and behavior of UI elements.

* Widget Composition:-

-> Complex UIs are built by nesting widgets within each other.

-> StatelessWidgets are immutable, while StatefulWidgets maintain dynamic state.

Commonly Used Widgets & Their Roles in Widget Tree

*Basic Widgets:-

1) Container: Used for layout, padding, and Styling.
2) Row and Column: Arrange widgets horizontally and Vertically.
3) Text: Displays text Content.
4) Image Displays images from assets or network.

* Structural Widgets

1) Scaffold:- Provides a default layout structure with an app bar, body, and floating button.

Teacher's Sign.:

2) AppBar :- Creates a material-design-style app bar.
3) ListView :- Creates a scrollable list of widgets.

* Interactive Widgets :-
1) GestureDetector :- Detects user gestures like taps and swipes.
2) ElevatedButton :- A button with elevation effects.
3) TextField :- An input field for user text input.

Q.3.a) Importance of State Management in Flutter Applications.

→ State management controls the behavior and data flow in an app. it ensures
→ UI updates dynamically based on user actions.
→ Efficient data handling and component reusability
→ Maintainability of Complex applications.

Q.3.b) Comparison of State Management Approaches.

→ * Scenarios for Each Approach :-
→ SetState :- Small applications where minimal state changes occur.
→ Provider :- Medium-scale apps needing optimized UI Updates
→ Riverpod :- Large-scale applications requiring dependency injection and modular state management.

Approach	Best for	Pros	Cons
setState	Small apps	Simple to use	UI overhaills frequently
Provider	Medium apps	Optimized for performance	Require understanding of
Riverpod	Large apps	Scalable & Testable	Learning curve is higher

Teacher's Sign.:

Firebase Integration & Benefits

* Steps to Integrate Firebase in Flutter

1) Create a firebase project at Firebase Console.
2) Add an Android/ios app to the firebase Project
3) Download and add the google-service.json (for Android) or GoogleService-Info.plist (for ios) to the Flutter Project

4) Install Firebase dependencies (firebase_core, cloud_firestore)
5) Initialize Firebase in main.dart:

```
import 'package:firebase_core/firebase_core.dart';

Void main() async {
    widgetsFlutterBinding.ensureInitialized();
    await Firebase.initializeApp();
    runApp(MyApp());
}
```

* Benefits of firebase as a backend Solution

-> No need to manage backend Servers
-> Real-time database Synchronization
-> Integrated authentication (Google, Facebook)
-> Cloud Functions for business logic.

Q.4 b) Common Firebase Services Used in Flutter and Data Synchronization.

→ Common Firebase Services :-

1) Firebase Authentication :- User login / signup using email, Google, Facebook, etc.
2) Cloud Firebase :- NoSQL real-time database for storing app data.
3) Firebase Storage :- For storing images, videos and files.
4) Firebase Cloud Messaging (FCM) - Push notifications.
5) Firebase Analytics - Tracks user behaviour

* Data Synchronization in Firebase

→ Firestore uses real-time listeners to automatically update data when changes occur.

Fetching and listening to real time Firestore updates

```
FirebaseFirestore.instance.collection('users').snapshots().
    listen((snapshot) {
  for (var doc in snapshot.docs) {
    print(doc.data());
  }
});
```

Experiment No.	
Title.	
Roll No	
Name	
Class	
Subject	
Grade:	

Teacher's Sign.: