Module I: Basics of Flutter Programming

1. **What is Flutter and which programming language does it use?**
   Answer: Flutter is an open-source UI SDK developed by Google, used for building natively compiled apps for mobile, web, and desktop using a single codebase. It uses the Dart programming language.
2. **Explain the widget lifecycle in Flutter.**
   Answer: The widget lifecycle includes creation (initState()), updating (didUpdateWidget()), and destruction (dispose()). It allows handling initialization and cleanup tasks.
3. **Differentiate between Widget Tree and Element Tree.**
   Answer: Widget Tree defines the configuration, while Element Tree represents instances of widgets that Flutter uses for rendering.
4. **What is the significance of the main() function in Flutter?**
   Answer: main() is the entry point of a Flutter application. It calls runApp() to attach the widget tree to the screen.
5. **How is Flutter installed on a system?**
   Answer: Download the SDK from flutter.dev, extract it, add the bin folder to the PATH, and verify installation using flutter doctor.
6. **What is StatelessWidget vs StatefulWidget?**
   Answer: StatelessWidget does not store state that changes over time; StatefulWidget holds mutable state that can change during the widget's lifetime.
7. **How does Flutter render the UI?**
   Answer: Flutter uses its own rendering engine (Skia) to draw UI elements directly on the screen using a canvas.
8. **Why is Dart used with Flutter?**
   Answer: Dart compiles to native ARM code and also to JavaScript, enabling both fast startup and high performance.
9. **What command is used to create a new Flutter project?**
   Answer: flutter create <project_name>
10. **How do you check if Flutter is properly installed?**
    Answer: Run flutter doctor in the terminal to verify the setup and check for missing dependencies.

Module II: Developing Flutter UI

1. **What is the purpose of SafeArea in Flutter?**
   Answer: It avoids system UI intrusions (e.g., notches, status bar) by padding content to stay within safe boundaries.
2. **Name and describe two layout widgets in Flutter.**
   Answer: Column – aligns children vertically; Row – aligns children horizontally.

3. **How do you handle user gestures in Flutter?**
   Answer: Using GestureDetector, which listens to touch events like taps, drags, and swipes.
4. **Explain how AnimatedContainer works.**
   Answer: It animates changes in its properties like size, color, etc., without requiring explicit animation controllers.
5. **What is the use of Navigator and Named Routes?**
   Answer: Navigator manages stack-based routing; Named Routes define paths centrally for better navigation management.
6. **What are TabBar and TabBarView used for?**
   Answer: TabBar displays tabs, while TabBarView shows corresponding content. Together, they implement tab-based navigation.
7. **List two gesture widgets in Flutter.**
   Answer: Draggable, DragTarget
8. **What is a RichText widget?**
   Answer: It displays text with multiple styles (colors, fonts) using TextSpan.
9. **How is BottomNavigationBar implemented in Flutter?**
   Answer: It is placed at the bottom of the screen and allows switching between different views or pages.
10. **Explain the use of Form widget.**
    Answer: It groups input widgets for validation and submission.

Module III: Creating Production Ready Apps

1. **How can you include external libraries in Flutter?**
   Answer: By adding dependencies in pubspec.yaml and running flutter pub get.
2. **What are the ways to read/write files in Flutter?**
   Answer: Using dart:io package functions like File.readAsString(), File.writeAsString().
3. **Explain JSON parsing in Flutter.**
   Answer: Use dart:convert library to convert JSON strings into Dart maps or classes using jsonDecode().
4. **How do you add Firebase to a Flutter project?**
   Answer: Add Firebase SDK, configure via Firebase Console, initialize Firebase in main().
5. **What is Cloud Firestore?**
   Answer: A NoSQL cloud database from Firebase used for storing and syncing app data in real-time.

6. **What is widget testing in Flutter?**
   Answer: Testing UI components in isolation using flutter_test package to validate behavior and layout.
7. **Explain how to deploy a Flutter app to Android.**
   Answer: Build APK or App Bundle using flutter build apk or flutter build appbundle, then upload to Play Console.
8. **List steps to configure Firestore security rules.**
   Answer: Use Firebase Console → Firestore → Rules tab → Define access policies using Firestore rule syntax.
9. **What is the role of flutterfire CLI?**
   Answer: Helps in automating Firebase setup in Flutter apps.
10. **What is the difference between hot reload and hot restart?**
    Answer: Hot reload injects updated code into the Dart VM, preserving state. Hot restart restarts the app, resetting state.

Module IV: Introduction to Progressive Web Apps (PWA)

1. **What is a Progressive Web App (PWA)?**
   Answer: A web application that uses modern web technologies to deliver an app-like experience with offline capabilities and performance.
2. **List characteristics of a PWA.**
   Answer: Responsive, Offline-capable, Installable, Secure (HTTPS), Linkable.
3. **Compare PWAs and Hybrid Apps.**
   Answer: PWAs run in browsers with web tech; Hybrid apps use native wrappers like Cordova for app store distribution.
4. **What are the requirements for a PWA?**
   Answer: HTTPS, Web App Manifest, Service Workers.
5. **Explain the role of HTTPS in PWAs.**
   Answer: Ensures secure communication, required for service workers and push notifications.
6. **What is a web app manifest?**
   Answer: A JSON file that defines how the app appears to the user (icons, name, theme).
7. **Name one PWA framework.**
   Answer: Workbox, Angular PWA, or Lighthouse.
8. **Give an example use case for a PWA.**
   Answer: News website with offline access to latest articles.
9. **What does "installable" mean for PWAs?**
   Answer: Users can add it to their home screen like a native app.

10. **Mention two benefits of PWAs.**
    Answer: Offline functionality and low friction installation.

Module V: Creating Responsive UI

1. **What is responsive web design?**
   Answer: A design approach that ensures web content adapts to various screen sizes and orientations.
2. **What's the difference between responsive and adaptive design?**
   Answer: Responsive adapts fluidly to screen sizes; adaptive uses predefined layouts for specific breakpoints.
3. **Name a tool/library for responsive UI.**
   Answer: Material UI, Bootstrap, jQuery Mobile.
4. **What are media queries in CSS?**
   Answer: Rules that apply CSS styles conditionally based on screen characteristics like width or resolution.
5. **Define flexible grid-based layout.**
   Answer: Layouts using relative units (%, vw, vh) to adapt to screen sizes.
6. **How does responsive design enhance UX?**
   Answer: Provides consistent, accessible, and visually pleasing experience on all devices.
7. **What is a breakpoint in responsive design?**
   Answer: Specific screen width where layout/style changes to accommodate different devices.
8. **What is the role of flexible images?**
   Answer: Images that scale with screen size using CSS rules like max-width: 100%.
9. **How can media queries help with performance?**
   Answer: Avoids unnecessary content loading and renders only what is needed for that screen size.
10. **What's the difference between mobile-first and desktop-first design?**
    Answer: Mobile-first designs for smaller screens first and then scales up; desktop-first starts large and scales down.

Module VI: Web App Manifest & Service Workers

1. **What is the purpose of manifest.json?**
   Answer: It defines metadata about the web app like name, icons, start URL, and theme for installability.
2. **How can you make a web app installable?**
   Answer: By including a valid manifest.json and registering a service worker.

3. **What are the key properties of manifest.json?**
   Answer: name, short_name, start_url, icons, theme_color, background_color, display.
4. **What is a service worker?**
   Answer: A script that runs in the background, handling network requests, caching, and push notifications.
5. **Explain the service worker lifecycle.**
   Answer: Register → Install → Activate → Event handling (fetch, push, sync).
6. **What is offline functionality in PWAs?**
   Answer: Ability to access content and functionality even without internet, enabled by caching strategies.
7. **How does caching work in service workers?**
   Answer: Resources are stored using the Cache API, then retrieved during offline or slow network.
8. **What are push notifications in PWAs?**
   Answer: Alerts sent from server to client, even when the app is not open, using service workers.
9. **Explain background sync.**
   Answer: Allows delayed actions (like form submission) to be queued and executed when connectivity is restored.
10. **How is Lighthouse used in PWA development?**
    Answer: It audits web apps for PWA compliance and performance, providing improvement suggestions.