1. Explain the benefits of using Flutter over other cross-platform frameworks.

Answer:
Flutter offers several benefits compared to other cross-platform frameworks:

- **Single Codebase**: Write once, run on Android, iOS, Web, and Desktop.
- **Hot Reload**: Instantly see code changes without restarting the app.
- **Native-like Performance**: Flutter uses its own rendering engine (Skia), so apps are fast and smooth without relying on native UI components.
- **Customizable UI**: Rich set of pre-designed widgets and the ability to create custom ones for pixel-perfect designs.
- **Strong Community & Backing**: Supported by Google, with an active developer community and regular updates.
- **Fewer Compatibility Issues**: Since Flutter draws the UI itself, apps look consistent across platforms.

---

2. Describe the role of Dart in Flutter. What are its advantages for mobile development?

Answer:

- **Role of Dart:** Dart is the programming language used by Flutter to build apps. It powers both the UI and business logic. Flutter compiles Dart code into native ARM or x86 machine code, ensuring high performance.

Advantages of Dart for mobile development:

- **Fast Compilation**: Supports both Just-in-Time (JIT) for quick development (hot reload) and Ahead-of-Time (AOT) for optimized release builds.
- **Object-Oriented & Modern**: Similar to Java, C#, and JavaScript, making it easy to learn.
- **Strong Typing + Null Safety**: Reduces runtime errors.
- **Asynchronous Programming**: async and await make handling futures and streams easier.
- **UI-Friendly**: Dart was designed with Flutter in mind, making it well-suited for reactive programming and building widget trees.

---

3. Outline the steps to set up a Flutter development environment.

Answer:
Steps to set up Flutter:

1. **Install Flutter SDK**
    - Download from the official Flutter website.
    - Extract and add Flutter to system PATH.
2. **Install Android Studio / VS Code**
    - Install one IDE of choice.
    - Add Flutter and Dart plugins.
3. **Set Up Android Emulator / iOS Simulator**
    - For Android: Install Android Studio, SDK tools, and create a virtual device.
    - For iOS (macOS only): Install Xcode and set up a simulator.
4. **Verify Installation**
    - Run flutter doctor in terminal to check dependencies.
    - Fix any issues reported by the doctor.
5. **Create First Project**
    - Run:
    - flutter create my_app
    - cd my_app
    - flutter run
    - This runs the default counter app.

---

4. Describe the basic Flutter app structure, explaining main.dart, the main function, and the widget tree.

Answer:

- **main.dart**:
    - The entry point file of a Flutter app.
    - Contains the main() function, which launches the app.
- **main function**:
    - Standard entry point in Dart.

- Calls runApp(), which takes a widget (usually the root widget) and inflates it into the screen.
- Example:
- void main() {
- runApp(MyApp());
- }

- **Widget Tree**:
  - Flutter apps are built as a hierarchy of widgets.
  - Two main widget types:
    - **StatelessWidget**: UI that doesn't change.
    - **StatefulWidget**: UI that can change dynamically.
  - Example widget tree:
  - MaterialApp
  - └── Scaffold
  - ├── AppBar
  - └── Body
  - └── Center

└── Text("Hello Flutter")