TPRG 343 Quiz

Name:	**Wong Jie Ying**
Id Number:	**2350080-DCS**
Marks	/20

1. Explain the core concepts of Android programming and how they apply in Flutter.

[3 marks]

- **Activity and Lifecycle**

  In Android, an Activity is a screen with a lifecycle (`onCreate`, `onStart`, etc.). In Flutter, a `Widget` represents the UI, with lifecycle management handled by the `State` class.

- **Views and Widgets**

  Android uses `Views` for UI elements. In Flutter, everything is a `Widget`, the fundamental building block for creating the user interface.

- **Layouts and Composability**

  Android uses layouts like `LinearLayout` to arrange views. Flutter uses widgets like `Column` and `Row` to build layouts through a compositional approach.

2. Identify and describe three key features of Flutter that enhance mobile application development.

[2 marks]

- **Hot Reload**

  Allows developers to see changes in real-time without restarting the app, speeding up the development process.

- **Cross-Platform Development**

  Enables the creation of apps for both iOS and Android using a single codebase, reducing development time and costs.

- **Rich Widget Library**

  Provides a vast collection of customizable widgets that make it easy to build complex UIs with a native look and feel.

3. Discuss the differences between traditional Android programming and Flutter in terms of performance and development speed.

[3 marks]

Aspect	Traditional Android	Flutter
Performance	Uses native components, leading to optimized performance but may vary with complexity.	Compiles to native code and uses a high-performance rendering engine, generally providing smooth performance.
Development Speed	Requires separate codebases for different platforms, which slows down development. Build times can be long.	Allows for a single codebase across platforms and offers hot reload, speeding up development and testing.
UI Design	Provides platform-specific components and APIs, which can require more work for customization.	Offers a customizable UI framework with consistent design across platforms and easier complex UI creation.

4. Describe the role of Dart in Flutter development. Why was it chosen as the primary programming language for Flutter?

[2 marks]

- **Performance**

  Dart's Ahead-of-Time (AOT) compilation compiles code to native ARM, improving app performance and startup times.
- **Hot Reload**

  Dart's features support Flutter's hot reload, allowing developers to see changes instantly without restarting the app.
- **Single Codebase**

  Dart allows Flutter to maintain one codebase for both Android and iOS apps, simplifying development and maintenance.

5. Explain how Flutter manages cross-platform development while maintaining performance and native-like experience.

[2 marks]

- **Custom Widgets**

  Flutter provides a rich set of customizable widgets that mimic native components, ensuring a consistent look and feel across platforms.
- **Skia Rendering Engine**

  Flutter uses Skia to render UI directly to the screen, bypassing platform-specific UI components and delivering smooth, high-performance graphics.
- **Ahead-of-Time (AOT) Compilation**

  Flutter compiles Dart code to native ARM code, optimizing performance and startup times, similar to native apps.

6. List and explain the software tools required for developing a Flutter application.

[2 marks]

- Flutter SDK
- Android Studio
- Visual Studio Code
- Command-Line Tools

7. What are widgets in Flutter? Describe how they are used in building the user interface.

[2 marks]

- Widgets in Flutter are the building blocks of the user interface. They define how UI elements look and function such as buttons, text, and images.
- **Composition**

  Widgets are combined to build complex interfaces. For example, a Column widget can contain multiple Text and Button widgets arranged vertically.
- **Customization**

  Widgets can be customized through properties. For instance, a Container widget can be styled with padding, margins, and colors.
- **Reactivity**

  Widgets are immutable and can be rebuilt in response to state changes. Flutter uses a reactive framework where the UI updates automatically when the state changes.

8. Compare the application architecture of Flutter with that of native Android development.

[2 marks]

Aspect	Flutter	Native Android
Architecture	Reactive, widget-based. UI is built by composing widgets in a declarative style.	MVC or MVVM, using XML for UI layouts and Java/Kotlin for behavior.
State Management	Managed within widgets, using various solutions like Provider.	Managed with LiveData, ViewModel, or directly in activities/fragments.
Rendering	Managed within widgets, using various solutions like Provider.	Uses native Android rendering engines, which can vary.
Platform Integration	Uses platform channels for native communication.	Directly integrates with Android APIs.
Codebase	Single codebase for both Android and iOS.	Separate codebases for Android and iOS if using native development.

9. Discuss the significance of Flutter’s hot reload feature in the development process.

[1 mark]

- **Faster Iteration**

  Developers can instantly view the effects of code changes, which accelerates testing and refining of UI and functionality.
- **Improved Productivity**

  Reduces the need for lengthy app restarts, saving time and increasing efficiency during development.
- **Enhanced Experimentation**

  Facilitates quick experimentation with different design and layout options, making it easier to fine-tune the user interface.
- **Preserves State**

  Maintains the app's state while applying changes, allowing developers to continue from where they left off without losing progress.

10. Provide a brief overview of how Flutter applications are deployed on Android devices.

[1 mark]

- Set Up Development Environment
- Configure Flutter Project
- Build APK
- Connect Device
- Run the App
- Debugging and Testing
- Release