# MAD Assignment No 1

Name : Ansari Mohammed Sharjeel
Class : D15B Roll No : 03

Q1 a Explain the key features and advantages of using flutter for mobile app development.

Ans Flutter is a cross platform UI toolkit developed by Google for building natively compiled applications for mobile, web. and advantages include i) ~~Hot~~ Expressive UI: Flutter provides a rich set of customisable widgets for creating visually apealing interfaces.

ii) Hot Reload : Enables developers to instantly view changes without restarting app.

iii) Widget based architecture : UI components in Flutter are widgets making development modular and customisable.

iv) Single codebase : Develope onee, deploy everywhere, reducing development time & effort.

v) Strong community support : A large and active community contributes to a wealth of resources and packages.

Q1 b Discuss how flutter framework differs from traditional approaches and why it has gained popularity in the developer community.

Ans. i) Flutter uces reactive framework, wheras traditional approaches are typically imperat

ii) Flutter offers a consistent UI accross platform ensuring a native look and feel.

iii) Use of dark language and widget based appra ch enhances developer productivity.
iv) popularity arises from the efficient develop ment process, performance and vibrant community.

Q2 a Describe concept of widget tree in Flutter Explain how widget composition is used for building complex user interfaces.

Ans i) In flutter the widget is a fundamental concept that represents the heirarchy of user interfaces elements in an application Everything in flutter is a widget whether its a button text image or even entire applicat -ion itself. Widget are arraged in a way of tree structure where each widget can have zero or more children forming heirarchy
ii) The widget tree is composed of various types of widget each carrying a specific purpose. widget can be broadly categorised into two. a) stateless b) state ful.
iii) stateless widgets are immutable and dont have any internal state, while stateful widg -ts can change their internal state during their life time.

Q2 b provide examples of community used widgets and their rules in creating a widgets tree.

Ans Examples of commonly used widgets. ~~tree~~ are

i) Material app : Defines the basic structure of a flutter.
ii) Saffold : Represents the basic visual structure of the app. including the app bar and body
iii) Container : A box model than can contain other widgets providing layout and styling
iv) Row and column : Arrange child widgets horizontally or vertically
v) List view : Displays a scrolling lists of widgets.
vi) Floating Action Button : Represents a floating action button.

Q3 a. Discuss the importance of state managment in flutter application.

Ans State managment is a crucial aspect of building robust and efficient flutter application. In flutter "state" refers to data that influences the appearacne and behaviour of widgets. Managing state effectively is essential for creating responsive, ~~state~~ ~~effectively~~ dynamic and scalable applications. Here are some key reason why state managment is important in Flutter.

i) User Interface.
ii) Performance optimisation.
iii) Code Maintainability.
iv) Reusability & Modularity.

Sundaram
FOR EDUCATIONAL USE

v) Persistance and Navigation.
vi) Stateful widget limitations.
vii) Concurrency and Asynchronous Operations.

Q3 b. Compare and contrast the different state managment approaches available in flutter such as set state. Provider and Riverpod. Provide scenarious where each approach is suitable.

Ans I) set state : 'set state' is most straight-forward way to manage state in flutter. It is build into framework and is easy to understand for beginners.

Appropriate for simple UIs : For small to moderately complex UIs where the state changes are localised and widget tree is not deeply nested 'net state' can be sufficient.

Cons :

i) 'set state' is limited to widget where it is called and descendants.

ii) It triggers a rebuild of the entire widget and its subtree potentially causing performance issues for larger applications.

~~Suitable scenarios:~~

~~i) Small~~

II) Provider :

Pros :

i) providers allows for scoped and localized state managment reducing the need for

Sundaram
FOR EDUCATIONAL USE

prop drilling

ii) Easy navigation: It is easy to integrate into flutter applications and offers a good blance between simplicity and flexibility

iii) large community support:
provider is wisely used ~~to~~ and has good community support.

Cons:

i) Global scope: In some cases global state might be unintentionally created.

ii) Suitable scenarios:

i) Applications of varying sizes with moderate to complex UIs

ii) Suitable where a centralised state management solution is needed but without the complexity of other solutions.

~~iii)~~

II) Riverpod:

Pros.: i) Scoped and flexible.

ii) provider Inheritance

iii) Immutable and reactive.

iv) cons: i) Learning curve: Similar to 'Provider'

ii) Some of advanced features may not be necessary for simplex applications adding unneccessary complexity.

Suitable scenarios:

i) large and complex applications:

ii) Situations where a more sophisticated scalable and reactive state managment.

solution is required
projects where dependency injection is a crucial consideration.

Q4a. Explain process of integrating firebase with a flutter application. Discuss benefits of using firebase as a backend solution.

Ans i) Creating a firebase project:
- Go to firebase console and create a new project
- Follow the setup instructions.

ii) Add firebase to flutter project
- In your flutter project add fire base SDK dependencies to the '.yaml' file.

iii) Initialize firebase:
- Import firebase packages and initialise firebase in 'main.dart' file.

iv) Configure firebase services.
- Depending on services you want to use (authentication, firebase etc) configure them by following specific setup instructions provided by firebase.

v) Use firebase services ~~to~~ in your app code.

Benefits of using firebase:
i) Realtime database.
ii) Authentication
iii) Cloud ~~functions~~ functions
iv) cloud fire store

vi) Hosting and analytics.
vii) Authentication state managment.
viii) Secure and scalable.
ix) Easy setup and integration.

Q4 b. Highlight the firebase services commonly used in flutter development and provide a breif overview of how data synchronisation is achieved.

Ans Common Firebase services in Flutter Development are :

i) Authentication : Firebase Authentication for user sign-in

ii) Firestore : A NO SQL database for real time data synchronisation

iii) Firebase Cloud Messaging (FCM) : push Notification for engaging users.

* Data synchronization :

i) Listners and Streams : Fire base services use listners and streams extensively. Flutter developers can use stream-based API to listen for changes in data whether its in firestore the Realtime Database or other firebase services.

ii) Reactively updating UI : Flutter's stream builder widget is commonly used to reactively update UI components based on changes in data streams. When data changes on server the stream emmits new data trigerring ~~or~~ rebuild of the associate UI.

iii) Offline Support : Firebase services provide build in offline support. Flutter apps can work seamlessly offline and when connectivity is restored changes made offline are automatically synchronised with the server.