Mobile Application Architectures And Design patterns

- An application architecture describes the patterns and techniques used to design and build an application. The architecture gives you a roadmap and best practices to follow when building an application, so that you end up with a well-structured app. Software design patterns can help you to build an application.

- Mobile app architecture includes various layers or components, each with specific responsibilities, and it determines how data is processed, presented to users, and interacted with. It plays a critical role in the app's performance, scalability, maintainability, and security.

Key components of Mobile App Architecture

1) User Interface (UI) Layer : The UI layer is responsible for the presentation of the app to the user. It includes the visual elements and components that users interact with, such as screens, buttons, forms, navigation menus, and any graphical elements. It manages the layout and appearance of the app.

Common technologies used in the UI layer include UI frameworks, user interfacelibraries, and design tools that help create a visually appealing and responsive user experience.

2) Application Logic Layer

The application logic layer, also known as the business logic layer, houses the core functionality of the app. It includes algorithms, business rules, and processes that control the app's behavior.

This layer processes user input, orchestrates data retrieval and storage, and ensures the correct operation of the app's features.

3) Data Layer

The data layer manages data storage, retrieval, and communication with external data sources. It includes databases, server APIs, and any data repositories that the app interacts with. This layer ensures data integrity, security, and availability.

In the data layer, technologies like databases (SQL or NoSQL), RESTful or GraphQL APIs, and caching mechanisms are commonly used. Data management frameworks, Object-Relational Mapping (ORM) libraries, and data synchronization tools may also be part of this layer

Types of Mobile App Architecture

1) Monolithic Architecture

In monolithic architecture, all components and modules of an application are tightly integrated into a single, unified unit. In a monolithic architecture, the entire application, including the user interface, application logic, and data storage, is bundled together as a single codebase and runs within a single process.

Some advantanges of Monolithic Architecture include;

- Easier to develop, test, and initially deploy
- A wealth of knowledge and tools available for monolithic development
- Does not require communication between separate services

2) Microservices Architecture

In a microservices architecture, an application uses small, independent, and loosely coupled services that operate simultaneously to provide its functionality.

Instead of building a monolithic application where all features and functions are tightly integrated into a single codebase, a microservices architecture breaks down the application into a collection of individual services, each responsible for a specific set of tasks or functionalities.

Some advantages of Microservices Architectures include

- Microservices can be individually scaled up or down as needed
- Flexibility to choose the most appropriate technology for each service

- Smaller, focused teams can work on individual services, leading to faster development cycles and quicker time-to-market for new features.

3) Model-View-Controller (MVC)

Model-View-Controller (MVC) is a widely used architectural design pattern for developing software applications, particularly in web and mobile applications.

It divides the application into three interconnected components, each with a specific role and responsibility. The primary purpose of the MVC pattern is to separate the concerns of an application, making it easier to develop, maintain, and extend.

- ### Model:

It represents the data and business logic of the app. It encapsulates the core functionality and data management of the application. This component is responsible for data storage, retrieval, validation, and processing. It responds to requests from the Controller and notifies the View when the data changes.

- ### View:

Its responsibility is to render the user interface and present data to the user. It displays the information from the Model to the user in a visually appealing and understandable format.

- ### Controller:

It works like an intermediary between the Model and the View. It receives user input, processes it, and communicates with the Model to retrieve or update data. The Controller decides which View should be displayed to the user based on the user's actions.

4) Model-View-ViewModel (MVVM)

Model-View-ViewModel (MVVM) is an architectural design pattern used primarily in software development for building user interfaces. MVVM is especially popular in modern mobile and web applications.

It's an evolution of the Model-View-Controller (MVC) pattern, designed to enhance the separation of concerns and improve the testability and maintainability of code.

- Model:

The Model represents the application's data and business logic. It is responsible for data storage, retrieval, validation, and processing.

- View:

The View represents the user interface and is responsible for displaying data to the user. In MVVM, the View is the visual component of the application, such as a screen in a mobile app or a web page in a web application.

- ViewModel:

The ViewModel serves as an intermediary between the Model and the View. It encapsulates the presentation logic, transforms the data from the Model into a format that is suitable for the View, and handles user interactions.