TNGS Learning Solutions
AWS Solutions Architect
Online Course
3 Tier Application Architecture

3 Tier Application Architecture

- A three-tier application architecture, also known as a three-tier model or three-tier architecture, is a software architecture pattern that divides an application into three interconnected layers or tiers.
- Each tier has specific responsibilities and functions, and this separation helps improve scalability, maintainability, and flexibility in software development.

3 Tier Application Architecture

- A three-tier application architecture, also known as a three-tier model or three-tier architecture, is a software architecture pattern that divides an application into three interconnected layers or tiers.
- Each tier has specific responsibilities and functions, and this separation helps improve scalability, maintainability, and flexibility in software development.

Presentation Tier (User Interface)

- The presentation tier, also known as the user interface (UI) tier, is the topmost layer and is responsible for interacting with users.
- It presents information to users in a human-readable format and collects user input.
- This tier includes user interfaces such as web browsers, mobile apps, desktop applications, or other client-side components.
- It is responsible for rendering the user interface, handling user input validation, and often includes user authentication and session management.

Application Tier (Logic or Business Logic)

- The application tier, also known as the logic or business logic tier, is the middle layer.
- It contains the core application logic, business rules, and processing capabilities.
- This tier handles user requests from the presentation tier, processes data, and orchestrates interactions between various components.
- It may include services, APIs, middleware, and application servers.
- It is responsible for business logic execution, data manipulation, and communication with the data tier.

Data Tier (Data Storage)

- **Modularity**: Separating the application into three tiers makes it easier to develop, test, and maintain each component independently.
- **Scalability**: Each tier can be scaled independently based on demand. For example, you can scale the presentation tier to handle more user requests without affecting the application or data tiers.
- **Security**: Separation of concerns allows for better security practices, as each tier can have its own access control and security measures.

Data Tier (Data Storage)

- **Reuse**: Components in one tier can be reused across multiple applications, promoting code reusability.
- **Flexibility**: Changes or updates to one tier do not necessarily impact the other tiers, making it easier to adapt to evolving requirements.
- **Performance**: Load balancing and caching mechanisms can be applied to improve application performance.

Data Tier (Data Storage)

- **Maintenance**: Maintenance and updates can be performed on one tier without affecting the others, minimizing downtime and risks.
- **Interoperability**: Different technologies and programming languages can be used in each tier as long as they can communicate through defined interfaces (e.g., RESTful APIs).

Data Tier (Data Storage)

- Three-tier architecture is a well-established pattern used in various types of applications, including web applications, enterprise software, and mobile apps.
- It provides a structured and organized way to design and build complex systems while offering the flexibility to adapt to changing requirements and scaling needs.