**RestAPI:**

what is RestApi ?

A REST API, or Representational State Transfer Application Programming Interface, is a set of rules and conventions for building and interacting with web services. It is an architectural style for designing networked applications and is commonly used to create web services that are scalable, stateless, and can be easily consumed by a variety of clients, including web browsers and mobile applications.

Key characteristics and principles of a REST API include:

Resources: In a REST API, resources are identified by unique URIs (Uniform Resource Identifiers). These resources can represent objects, data, or entities in the application, such as user profiles, products, articles, or any other data.

HTTP Methods: REST APIs use standard HTTP methods to perform operations on resources. The most common HTTP methods used in REST are:

GET: Retrieve data from the server.

POST: Create new resources on the server.

PUT: Update or replace existing resources.

PATCH: Partially update an existing resource.

DELETE: Remove a resource from the server.

Stateless: A fundamental principle of REST is statelessness. Each request from a client to the server must contain all the information needed to understand and process the request. The server doesn't maintain any client-specific state between requests.

Representations: Resources in a REST API can have multiple representations, such as JSON, XML, or HTML, which are used to exchange data between clients and servers. Clients can request specific representations using content negotiation.

Uniform Interface: REST APIs use a uniform and consistent interface, which simplifies client-server interactions. This interface includes using standard HTTP methods and following naming conventions for URIs.

Client-Server Architecture: REST separates the client from the server, allowing both to evolve independently. This separation helps improve scalability and flexibility.

Layered System: REST allows for a layered architecture, where intermediaries such as proxies, caches, and load balancers can be ad

1. What is & How are REST APIs Statelessness/fulness ?

Statelessness:IP,HTTP

A RESTful API is designed to be stateless, which means that each request from a client to the server must contain all the information needed to understand and process the request. The server doesn't store any information about the client's previous requests or interactions. Each request should be self-contained, and the server should not rely on any client-specific context. This design principle has several benefits, including scalability, simplicity, and reliability.

In a stateless architecture:

Clients send requests to the server, including all the necessary information.

The server processes the request, responds, and doesn't keep track of client state between requests.

This design makes it easier to scale the system, as any server can handle any request without needing access to session or client-specific data.

It simplifies error handling and recovery because there's no need to worry about the server maintaining state.

A stateful API is like a waiter who remembers your order and preferences throughout the meal. For example, if you order a burger and fries, the waiter will remember that you also asked for extra ketchup and a side of ranch dressing. When you request additional items or demand changes to your order, the server can keep track of your preferences and ensure that your meal is served exactly as you requested.

A stateless API, on the other hand, is like a fast-food restaurant where each order is handled separately and independently. If you order a burger and fries, the restaurant will prepare your order without knowing your previous orders or preferences. When you order additional items, the place will treat each request as a separate transaction and will not maintain any previous context or history.

1. common status code ?

Ans: 200 OK - the request was successful and the response contains the requested data

201 Created - the request was successful and a new resource was created

400 Bad Request - the request was invalid or missing required parameters

401 Unauthorized - the client needs to authenticate to access the resource

404 Not Found - the requested resource was not found

500 Internal Server Error - an unexpected error occurred on the server

1. What are the difference between REST and SOAP ?

REST (Representational State Transfer) and SOAP (Simple Object Access Protocol) are two different architectural styles and protocols used for building web services. Here are some key differences between REST and SOAP: Architectural Style: REST is an architectural style that uses a set of constraints and principles to design networked applications. It is more of a design philosophy. SOAP is a protocol with a specific set of rules for structuring messages and defining how they should be processed. Protocol: REST can use various underlying protocols, but it is most commonly associated with HTTP. SOAP is a protocol in itself and can be used over a variety of lower-level protocols, including HTTP, SMTP, and more.