what is the difference between rest and soap ? when should you choose one over the other in your application ?

REST :Representational State Transfer

SOAP : Simple Object Access Protocol

REST and SOAP both define how to build API
REST is a set of architectural principles

SOAP is a Protocol

REST the request done through HTTP
SOAP can handle data through any of layer protocols

REST return messages in a variety of formats
SOAP return messages as xml doc

REST stateless interaction
SOAP can be stateless or stateful

REST faster and attuned to the needs of lightweight web/mobile applications
SOAP slower due to the XML parsing overhead and larger message sizes suitable for complex operations requiring more extensive protocols.

REST security is handled using HTTPS,OAuth
SOAP has built-in security

REST often used for CRUD or public APIS
SOAP often used in enterprise applications requiring high reliability or operations that require ACID compliance and formal contracts between client and server.

Use REST when you want:
limited resources and bandwidth
fast performance
support multiple data formats
statelessness
ease of coding
cashing

SOAP:
built-in security
stateful
formal means of communication
advanced error handling capabilities.

Summary
REST is a set of guidelines that offers flexible implementation
SOAP is a protocol with specific requirements

can you explain how RESTful APIS work? How do http methods like get , post, put and delete interact with each other?

The client sends a request to the server
The server confirms that the client has the right to make that request.
The server receives the request and processes it.
The server returns a response to the

GET is used to retrieve existing resources without modifying them.

POST is for creating new resources.

PUT is used to update existing resources or create them if they don’t exist.

DELETE is for removing resources.

the methods are designed to work with each other to facilitate a full lifecycle of resource management

when it comes to securing your api what solutions do you use ? have you heard of OAuth and JWT ? when would you prefer one over the other ?

JWT:

While JWTs are indeed self-contained and allow for stateless authentication, they do not inherently manage sessions. Instead, they carry information that can be verified without server-side session storage.

OAuth:

OAuth does not necessarily maintain session state on the server. It can use access tokens (which can be JWTs) that are stateless.

OAuth focuses on authorization, allowing third-party access to resources without sharing credentials, but it does not manage user authentication directly (though it can be combined with OpenID Connect for authentication).

how would you handle rate limiting issues ? if you have a high number of requests how can you manage the flow without overloading your server ?

Identify Rate Limiting Requirements
Implement Rate Limiting Algorithms
Use Middleware and Libraries
Centralized Rate Limiting
Caching Strategies
Throttling Techniques
Queueing Requests
Monitor Usage Patterns
Provide Feedback to Clients
Load Balancing

what does CORS mean? how would you solve cross-origin issues when making requests from a different domain?

CORS stands for Cross-Origin Resource Sharing. It is a mechanism that allows resources to be requested from an application running on a different domain than the one from which they originated

To solve cross-origin issues
Configure the Server
Use a Proxy Server
JSONP (JSON with Padding)
Use Specific Browser Extensions
Handle Preflight Requests

when building an api do you think its better to return data in json or xml format ?why ?

JSON is generally the better choice for APIs

JSON's less verbose structure
more efficient in terms of payload size and parsing speed
is the standard format for modern web APIs
JSON works seamlessly with JavaScript and is well-supported in many programming languages, frameworks, and platforms.

If you have an API dealing with large data sets ,how can you use pagination or cursor-based pagination to avoid overwhelming your server ?

there is tow common methods

Offset-Based Pagination
Parameters: Use page and limit.
Pros: Simple to implement.
Cons: Can be inefficient with high offsets, leading to performance issues.

Cursor-Based Pagination:
Parameters: Use a cursor (e.g., an ID) to fetch the next set of results.
Pros: More efficient, provides consistent results even with data changes.
Cons: More complex to implement.

Have you heard of GraphQL? How can it be an alternatives to REST API in some cases ?

GraphQL is a query language for APIs and a runtime for fulfilling those queries with existing data.

GraphQL is likely a better choice if you have these considerations:
limited bandwidth, and you want to minimize the number of requests and responses
multiple data sources, and you want to combine them at one endpoint
client requests that vary significantly, and you expect very different responses

When building an API , how do you handle versioning ? Do you prefer to create a new API each time or maintain backward compatibility with older versions?

There are four common ways to version a REST API.

Versioning through URI Path
Versioning through query parameters
Versioning through custom headers
Versioning through content negotiation

How can you ensure that your API is scalable when the number of users increases significantly ?

Make sure that apply principles of scalable api design which is

Statelessness
Loose Coupling
Vertical & Horizontal Scaling
Resource-Oriented Design
Asynchronous Operations
Strategic Caching
Efficient Database Usage
API Rate Limiting

If you have real-time features like chat or live updates, do you think WebSocket are the ideal solution? Or are there other options like Server-Sent Events (SSE) ?

WebSocket A computer communications protocol, providing full-duplex communication channels over a single TCP connection.

SSE (Server-Sent Events) A standard allowing a web page to get updates from a server.

WebSocket: Ideal for interactive applications requiring real-time data in both directions (e.g., online gaming, collaborative tools).
SSE: More suited for applications where the server updates the client in real-time without needing client input (e.g., stock tickers, news feeds).
WebSocket: Best suited for applications requiring full duplex, real-time communication and interactive features.

SSE: A good choice for simpler applications where the server needs to update the client in real-time, but client interaction is limited or unidirectional.

When creating an API how do you create clear documentation ? what do you use-Swagger , Postman , or something else?

Postman

What is an API Gateway , and how can it help you manage and rout requests to a set of microservices?

An API gateway is an API management tool that sits between a client and a collection of backend services.
The API Gateway intelligently routes requests to the appropriate microservice based on the request path, method, or other criteria, reducing the need for clients to know the details of the backend architecture

How do you test your API to ensure it operates efficiently? Do you use unit tests or integration tests?

Unit Tests
Focus: Test individual components or functions of the API in isolation.
Purpose: Ensure that each function behaves as expected and handles edge cases correctly.
Tools: Frameworks like Jest, Mocha, or JUnit can be used to write unit tests.

Integration Tests
Focus: Test the interactions between different components or services, including database interactions and external APIs.
Purpose: Ensure that the API works correctly when integrated with other components, validating end-to-end functionality.
Tools: Use tools like Postman, Supertest, or RestAssured to perform integration tests.