**. What is a REST API?**

**Answer:**  
A **REST API** is a way for two computers to communicate over HTTP (like the web). It follows rules (REST principles) to fetch or modify data using standard methods like GET, POST, PUT, DELETE.

**2. What are the key principles of REST?**

**Answer:**

* **Stateless:** No client data is stored on the server between requests.
* **Client-Server:** Separation between frontend (client) and backend (server).
* **Uniform Interface:** Uses standard HTTP methods (GET, POST, etc.).
* **Cacheable:** Responses can be cached for better performance.
* **Layered System:** Requests can pass through multiple layers (like proxies).

**3. What are HTTP methods in REST?**

**Answer:**

* GET → Fetch data (e.g., load a user’s profile).
* POST → Create new data (e.g., add a new user).
* PUT → Update existing data (e.g., edit a user’s details).
* DELETE → Remove data (e.g., delete a user).
* PATCH → Partially update data (e.g., change only a user’s email).

**4. What is an endpoint in REST API?**

**Answer:**  
An **endpoint** is a URL where an API can be accessed (e.g., https://api.example.com/users).

**5. What is the difference between**PUT**and**POST**?**

**Answer:**

* POST → **Creates** new data (e.g., POST /users creates a user).
* PUT → **Updates or replaces** existing data (e.g., PUT /users/1 updates user with ID 1).

**6. What is the difference between**PUT**and**PATCH**?**

**Answer:**

* PUT → Replaces **entire** data.
* PATCH → Updates **only specific fields**.

**Example:**

* PUT /users/1 → Updates **all** user details.
* PATCH /users/1 → Updates **only** the email.

**7. What are status codes in REST API?**

**Answer:**  
HTTP status codes indicate if a request succeeded or failed:

* 200 OK → Success.
* 201 Created → Resource created (after POST).
* 400 Bad Request → Invalid input.
* 401 Unauthorized → Not logged in.
* 404 Not Found → Resource doesn’t exist.
* 500 Server Error → Backend crashed.

**8. What is JSON? Why is it used in REST APIs?**

**Answer:**

* **JSON (JavaScript Object Notation)** is a lightweight data format (like a dictionary in Python or object in JS).
* It’s **easy to read** and works across programming languages.

**Example:**

json

Copy

{

"name": "John",

"age": 25

}

**9. What is CRUD in REST API?**

**Answer:**  
CRUD stands for:

* **C**reate → POST
* **R**ead → GET
* **U**pdate → PUT/PATCH
* **D**elete → DELETE

**Example (User API):**

* POST /users → Create user
* GET /users/1 → Read user with ID 1
* PUT /users/1 → Update user
* DELETE /users/1 → Delete user

**10. What is authentication in REST API?**

**Answer:**  
Authentication verifies **who is making the request**. Common methods:

* **API Keys** (e.g., ?api\_key=123).
* **JWT (Tokens)** (e.g., Authorization: Bearer <token>).
* **OAuth** (used by Google/Facebook login).

**11. What is pagination in API?**

**Answer:**  
Pagination splits large data into smaller chunks (pages).

**Example:**

plaintext

Copy

GET /users?page=1&limit=10

(Returns first 10 users.)

**12. What is query parameter vs path parameter?**

**Answer:**

* **Path Parameter** → Part of the URL (e.g., /users/1 where 1 is the user ID).
* **Query Parameter** → Added after ? (e.g., /users?name=John).

**13. What is the difference between REST and GraphQL?**

**Answer:**

| **REST** | **GraphQL** |
| --- | --- |
| Fixed endpoints (e.g., /users) | Single endpoint (/graphql) |
| Returns all data (over-fetching) | Returns only requested data |
| Uses HTTP methods (GET, POST) | Uses queries (query { users { name } }) |

**14. How do you test a REST API?**

**Answer:**

* **Postman** → GUI tool for API testing.
* **cURL** → Command-line tool.
* **Swagger/OpenAPI** → Documentation + testing.

**Example (cURL):**

bash

Copy

curl -X GET https://api.example.com/users

**15. What is rate limiting in API?**

**Answer:**  
Rate limiting **blocks too many requests** from a single user (to prevent abuse).

**Example:**

* "100 requests per hour per user."