[Skip to content](https://chatgpt.com/c/6838907c-e120-800c-a61a-cb8f89f9d6de" \l "main)

**Chat history**

[New chat](https://chatgpt.com/)

Search chats

Ctrl K

[Library](https://chatgpt.com/library)

[GPTs](https://chatgpt.com/gpts)

Today

[Frontend API and Storage](https://chatgpt.com/c/6838907c-e120-800c-a61a-cb8f89f9d6de)

[PostgreSQL vs MySQL Comparison](https://chatgpt.com/c/68388fb4-ad9c-800c-be4c-a85025f05e05)

[Node.js Interview Questions](https://chatgpt.com/c/68388edc-8f00-800c-87a1-1d21624b3a33)

[Fibonacci Recursive JS Code](https://chatgpt.com/c/68388e7f-7f74-800c-aaad-5bb25447b6f7)

[Frontend Accessibility Interview Qs](https://chatgpt.com/c/68384b52-8964-800c-ae0d-e5e0ebd89628)

[React JWT Authentication](https://chatgpt.com/c/683826bc-7c20-800c-a69d-9fd3fe5e54d0)

[Usage of event.preventDefault()](https://chatgpt.com/c/68382233-5c20-800c-abee-b1f1cc4dd2c2)

[Interview Follow-up on WhatsApp](https://chatgpt.com/c/683804cd-d2d4-800c-be00-0c5bda1fe01e)

[Remove Duplicates Array](https://chatgpt.com/c/68377f28-c4a4-800c-9954-fa6624650f2d)

[JavaScript bind method](https://chatgpt.com/c/68377593-ab74-800c-8933-eb3a7c6a879b)

Yesterday

[Palindrome Substring Approach Review](https://chatgpt.com/c/68373ed5-e0d8-800c-bfa6-de6cfc109c77)

[jQuery Interview Questions](https://chatgpt.com/c/68373516-f1b4-800c-b91a-eaaed3d0df12)

[Frontend bottleneck solution](https://chatgpt.com/c/68372e77-3a00-800c-9f2b-d856aad14c9a)

[Prototype aur Inheritance Notes](https://chatgpt.com/c/6836ffdd-1774-800c-bc67-f076148f2b72)

[Interview Reschedule Request](https://chatgpt.com/c/6836f0a3-6b04-800c-a00d-3edd25092ef1)

[Maximum Product Explanation](https://chatgpt.com/c/683635a0-2308-800c-89d6-6e91e4a3f0bf)

Previous 7 Days

[TwoSum Function Error](https://chatgpt.com/c/6834cb37-cb84-800c-982f-df867caf78c6)

[Frontend Dev Concepts Study](https://chatgpt.com/c/6835b0d4-95b4-800c-a95b-3fdd764d2297)

[Frontend Developer Job Email](https://chatgpt.com/c/683592c7-8be8-800c-a120-f20105d28cbb)

[Cold email response draft](https://chatgpt.com/c/68341092-b050-800c-82d1-6c489bde80f1)

[MERN Stack Contract Opportunity](https://chatgpt.com/c/68333f32-1ec4-800c-be7d-f69e4cb1491a)

[CV banane ki request](https://chatgpt.com/c/6832f198-e8e0-800c-a959-d5fa1663d59b)

[GovFinder.us SRS Analysis](https://chatgpt.com/c/6832edab-e6ac-800c-9c43-c2e7060e047e)

[App Pilot SRS Document](https://chatgpt.com/c/6832eccb-8490-800c-b04a-4d97323f41f0)

[Frontend Developer CV Skills](https://chatgpt.com/c/6832df3b-8624-800c-b38b-842dfb0608f0)

[MERN Stack Developer Response](https://chatgpt.com/c/6832e7f3-2ff4-800c-a407-fedd758e7c7a)

[Rest Break Addition](https://chatgpt.com/c/6832dbc0-049c-800c-87a1-cf5a4de3b90a)

[Technical Interview Discussion](https://chatgpt.com/c/6832d267-f658-800c-b4d8-7e636c8f568b)

Upgrade plan

More access to the best models

ChatGPT

Share

User

**You said:**

frontend k lye api related kia questions hosakte hain? HTTP k hi batana cookies , loca; storage session storage bhi network tab me jo dekhte hain wo sab as a frontend deveoper kia kia knowledge honi chahye

**ChatGPT said:**

As a **frontend developer**, API integration and browser communication (like HTTP requests, cookies, and local/session storage) are essential areas of knowledge. Here's a complete breakdown of **important HTTP, browser storage, and network-related concepts** you should know and the **interview questions** that can be asked from them.

**✅ 1. HTTP Concepts for Frontend Developers**

**🔹 Must-Know Topics**

* HTTP methods: GET, POST, PUT, DELETE, PATCH
* Status codes: 200, 201, 204, 400, 401, 403, 404, 500
* RESTful APIs
* Headers (Content-Type, Authorization, etc.)
* JSON payloads
* CORS
* Fetch vs Axios
* Async/await with API calls

**❓Possible Interview Questions**

1. What is the difference between GET and POST?
2. What is a 401 vs 403 status code?
3. How do you send a token in an API request?
4. What is CORS, and how does it affect frontend development?
5. How do you handle API errors in the frontend?
6. What’s the difference between PUT and PATCH?
7. What are headers? Can you give examples of commonly used ones?
8. What is the purpose of the Content-Type header?
9. Difference between REST and RESTful APIs?

**✅ 2. Cookies, Local Storage, Session Storage**

**🔹 Must-Know Concepts**

| **Feature** | **Cookies** | **Local Storage** | **Session Storage** |
| --- | --- | --- | --- |
| Size Limit | ~4 KB | ~5–10 MB | ~5–10 MB |
| Expiry | Set manually | Never (until cleared) | End of session |
| Accessible by JS | Yes | Yes | Yes |
| Sent with requests | Yes (auto) | No | No |
| Use Case | Auth/session mgmt | Storing user prefs | Temporary data |

**❓Possible Interview Questions**

1. Difference between cookies, local storage, and session storage?
2. When should you use local storage over cookies?
3. Are cookies accessible through JavaScript?
4. Can you store JWT tokens in local storage?
5. How do you store and retrieve data from local/session storage in JavaScript?
6. What are the security concerns with storing sensitive data in local storage?

**✅ 3. Network Tab in Developer Tools (Chrome DevTools)**

**🔹 What You Should Know**

* How to inspect API calls
* Status codes
* Request and Response headers
* Payloads (request body)
* Preview and response tabs
* Timing tab (TTFB, DNS, etc.)
* Initiator (which file triggered the request)
* Throttling (simulate slow networks)

**❓Possible Interview Questions**

1. How do you debug a failed API call?
2. What does 304 Not Modified mean?
3. How can you check what data is being sent to the server?
4. What is the use of the “Preview” tab vs “Response” tab?
5. What’s the difference between Initiator and Timing tabs?

**✅ 4. Frontend Developer Expectations w.r.t. API and HTTP**

You should be able to:

* Integrate REST APIs using fetch or axios
* Understand and debug API requests using DevTools
* Handle loading, error, and success states in UI
* Securely store tokens or session data
* Deal with authentication and authorization flows (like storing JWTs and refreshing tokens)
* Optimize API calls (debounce search inputs, cancel previous requests)
* Understand caching and 304 responses

Would you like me to create **MCQs or coding scenarios** for these topics to practice? Or do you want **notes with real examples (e.g., fetch/axios)** for each?

**You said:**

sabko samjhao with examples

**ChatGPT said:**

Perfect! Let’s go through everything step-by-step with **clear explanations and examples** so you understand each concept well as a **frontend developer**.

**✅ 1. HTTP Basics for Frontend Developers**

**🔸 HTTP Methods**

| **Method** | **Use Case** | **Example** |
| --- | --- | --- |
| GET | Get data from server | Get user list |
| POST | Send new data to server | Create new user |
| PUT | Replace existing data | Replace whole user object |
| PATCH | Update partial data | Update user's email only |
| DELETE | Remove data | Delete user |

**Example:**

js

CopyEdit

*// GET request* fetch("https://api.example.com/users") .then(res => res.json()) .then(data => console.log(data)); *// POST request* fetch("https://api.example.com/users", { method: "POST", headers: { "Content-Type": "application/json" }, body: JSON.stringify({ name: "Ali", email: "ali@example.com" }) });

**🔸 Status Codes**

| **Code** | **Meaning** | **Example** |
| --- | --- | --- |
| 200 | OK | Data fetched |
| 201 | Created | New user created |
| 204 | No Content | Deleted, no response body |
| 400 | Bad Request | Wrong input |
| 401 | Unauthorized | No valid token |
| 403 | Forbidden | Access denied |
| 404 | Not Found | Wrong URL |
| 500 | Server Error | Server crashed |

**Frontend Example (handling error):**

js

CopyEdit

fetch("/api/user") .then(res => { if (!res.ok) throw new Error("Something went wrong"); return res.json(); }) .then(data => console.log(data)) .catch(err => console.error(err.message));

**🔸 Headers Example**

js

CopyEdit

fetch("/api/user", { headers: { "Content-Type": "application/json", "Authorization": "Bearer your\_jwt\_token" } });

**🔸 CORS (Cross-Origin Resource Sharing)**

* **Problem**: Your frontend runs on localhost:3000 but backend is on localhost:5000.
* **Browser blocks requests** unless backend allows it using proper headers.

**Fix**: Backend should allow CORS:

http

CopyEdit

Access-Control-Allow-Origin: \*

**✅ 2. Cookies, Local Storage, Session Storage**

**🔸 Cookies**

* Small data sent with **every HTTP request**.
* Used for **auth sessions**.
* Stored like:

js

CopyEdit

document.cookie = "username=Ali; expires=Fri, 5 June 2025 12:00:00 UTC";

**🔸 Local Storage**

* Persistent storage in browser.
* Data stays even after tab/browser close.

js

CopyEdit

localStorage.setItem("theme", "dark"); console.log(localStorage.getItem("theme")); *// "dark"*

**🔸 Session Storage**

* Same API as localStorage.
* Data goes away when the tab closes.

js

CopyEdit

sessionStorage.setItem("step", "2"); console.log(sessionStorage.getItem("step")); *// "2"*

**❗ Don’t store sensitive data (e.g., tokens) in localStorage — better to use HTTP-only cookies for security.**

**✅ 3. Network Tab in Developer Tools**

Open browser → Right click → Inspect → Network tab

**🔸 What to Look For**

| **Tab** | **Use** |
| --- | --- |
| Name | Lists API endpoints |
| Status | HTTP codes |
| Method | GET, POST etc. |
| Headers | Request & response headers |
| Payload | Data sent |
| Response | Data received |
| Preview | Formatted response |
| Timing | Load time |

**🔸 Example**

You call:

js

CopyEdit

fetch("https://jsonplaceholder.typicode.com/users");

In Network tab:

* **Name**: /users
* **Status**: 200
* **Preview**: shows user list
* **Timing**: how long the request took

**✅ 4. Frontend Developer Must-Know Skills for APIs**

You should be able to:

| **Task** | **Example** |
| --- | --- |
| Make API calls | fetch, axios |
| Handle states | Loading, error, success |
| Use async/await | await fetch() |
| Send auth tokens | Authorization: Bearer |
| Store data locally | localStorage or cookies |
| Debug requests | Chrome DevTools > Network tab |
| Optimize API usage | Debounce search, cache data |

**✅ Bonus: Axios Example**

js

CopyEdit

import axios from "axios"; axios.get("/api/profile", { headers: { Authorization: `Bearer ${token}` } }) .then(res => console.log(res.data)) .catch(err => console.error(err));

Would you like me to create:

* 🧠 A mini quiz to test these?
* 🧪 A hands-on coding project (e.g., weather app using API)?
* 📄 Cheat sheet for revision?

Let me know your preference.

**You said:**

"Content-Type": "application/json" iska kia matlab hota ha

**ChatGPT said:**

Great question! ✅

**🧾**"Content-Type": "application/json"**— What does it mean?**

This header tells the **server** what **type of data** is being sent in the **body of your request**.

**🔸 Simple Explanation:**

It means:

"Hey server! I'm sending you data in **JSON format**."

**🔸 Why is it important?**

If you don’t tell the server the content type:

* The server might not understand how to **parse the request body**.
* It may throw errors like **400 Bad Request**.

**🔸 Example Without**Content-Type

js

CopyEdit

fetch('/api/users', { method: "POST", body: JSON.stringify({ name: "Ali" }) *// You sent JSON* *// BUT you didn't tell the server it's JSON!* });

❌ The server might expect form data or something else and reject your request.

**✅ Correct Usage:**

js

CopyEdit

fetch('/api/users', { method: "POST", headers: { "Content-Type": "application/json" *// You tell the server it's JSON* }, body: JSON.stringify({ name: "Ali" }) *// And you're sending JSON* });

**🔸 Analogy:**

Imagine you're sending a parcel.

* Content-Type is like the **label** on the parcel:
  + 📦 JSON = "This parcel contains books"
  + 📦 Form = "This parcel contains clothes"

The receiver (server) uses this label to know **how to open and read the data**.

**🔸 Common Content-Types**

| **Content-Type** | **Meaning** |
| --- | --- |
| application/json | Sending JSON data |
| application/x-www-form-urlencoded | Sending form data (like <form>) |
| multipart/form-data | Used for file uploads |
| text/plain | Plain text |

Let me know if you want a visual diagram for this or practice questions to test your understanding.

Top of Form

Tools

Bottom of Form

ChatGPT can make mistakes. Check important info.



Ask In Chat