Fetch & Axios

Client Server Architecture

request

response

Client

Server

JSON

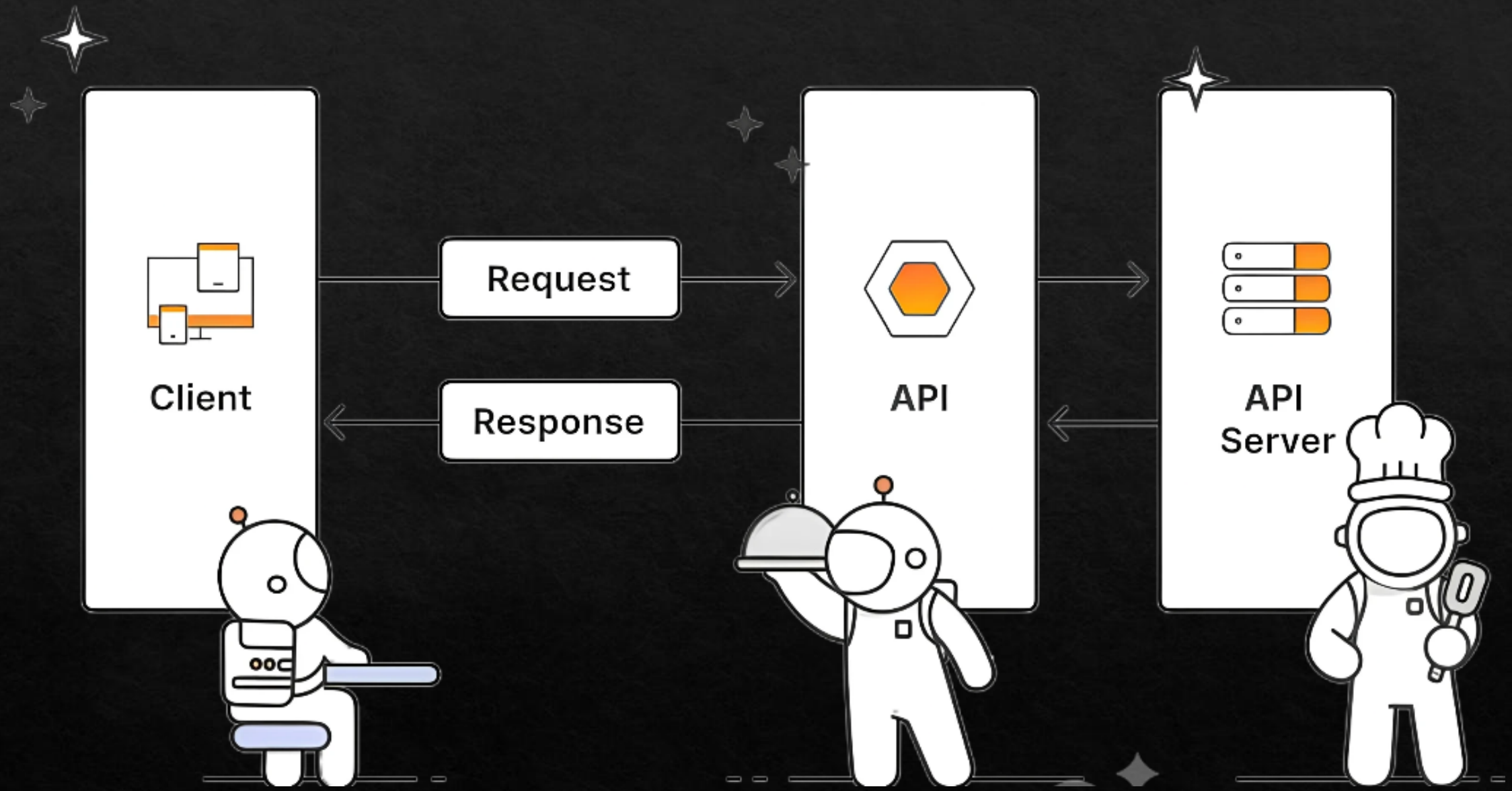
- JSON stands for JavaScript Object Notation.
- It’s a lightweight data format used to store and exchange data — like sending data between a browser and a server.

```
// JavaScript Object
const user = {
    name: "Manas",
    age: 25,
    isStudent: true
};

// JSON version (as a string)
const jsonUser = `{
  "name": "Manas",
  "age": 25,
  "isStudent": true
}`;
```

Note:

- Keys and string values are wrapped in double quotes.
- It's a string, not an object


Client
Request
Response
API
API
Server

HTTP Methods:

Tells a server what action to perform on a resource, such as creating, reading, updating, or deleting data.

- GET: Get/fetch existing data.
- POST: Add new data.
- PUT: Modify existing data. (Replaces the entire resource)
- PATCH: Modify existing data. (update specific fields without affecting others)
- DELETE: Remove existing data.

What is Fetch?

- Browser ka built-in function
- No installation
- Returns Promise
- Needs to convert JSON into readable JavaScript Ojbect
- More code, less features

Basic Example

```
useEffect(() => {
  fetch("https://jsonplaceholder.typicode.com/users")
    .then(res => res.json())
    .then(data => setUsers(data))
    .catch(err => console.log(err));
}, []);
```

What is Axios?

- ❑ External library
- ❑ Clean syntax
- ❑ Automatic JSON parsing
- ❑ Better error handling
- ❑ Interceptors (request/response)
- ❑ Timeouts inside
- ❑ Upload/Download progress support

Installation

```
// installation
npm install axios
```

Basic Example

```
useEffect(() => {
  axios.get("https://jsonplaceholder.typicode.com/users")
    .then(res => setUsers(res.data))
    .catch(err => console.log(err));
}, []);
```

Fetch vs Axios

Feature	Fetch	Axios
Built-in	✔ Yes	✘ No (install)
JSON parse	✘ Manually	✔ Automatic
Error handling	Weak	Strong
Upload progress	No	Yes
Interceptors	No	Yes
Timeout	Manual	Built-in
Syntax	Long	Short

Axios

Axios is a popular JavaScript library for making HTTP requests from both browsers and Node.js, simplifying communication with servers to fetch or send data

How to install Axios:

- Using CDN:

```
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
```

- Using npm:

```
npm install axios
```

Axios Configuration

These are the available config options for making requests.

```
const config = {
   url: './users',
   method: 'get', //default
   baseURL: 'https://some-domain.com/api/',
   params: {
     id: 1
   },
   data: {
     name: 'Manas',
     age: 21,
   },
   responseType: 'json',
   timeout: 5000,
   headers: {
     'Content-Type': 'application/json',
     Accept: 'application/json',
   }
 }
```

```
axios(config)
```

Note:

- ❑ timeout is by default 0 which means no time limit.
- ❑ some headers are also by default set, so you don't need to mention such headers. (you can see in browser network tab)

Only the url is required. Requests will default to GET if method is not specified.

Different Response Types

responseType	Used For	responseType
json	API responses (default)	json
text	Plain string / HTML	text
blob	File download	blob (used for images, pdf's, videos, zip files)
arraybuffer	Low-level binary	arraybuffer (Audio processing, Video stream buffering, Custom binary protocols)
document	XML/HTML	document
stream	Node.js streaming	stream

Shorthand Request Methods

- axios.request(config)
- axios.get(url, config)
- axios.delete(url, config)
- axios.head(url, config)
- axios.options(url, config)
- axios.post(url, data, config)
- axios.put(url, data, config)
- axios.patch(url, data, config)

Axios
Advanced Features

Interceptors

Request ke beech me kuch inject karna.

```
axios.interceptors.request.use(config => {
  config.headers.Authorization = `Bearer ${localStorage.getItem("token")}`;
  return config;
});
```

```
axios.interceptors.response.use(
  res => res,
  err => {
    console.log("Global Error:", err);
    return Promise.reject(err);
  }
);
```

Timeouts

```
axios.get(url, { timeout: 5000 })
  .catch(err => console.log("Timeout Error"));
```

File Upload Progress

```
axios.post(url, formData, {
  onUploadProgress: (progress) => {
    console.log((progress.loaded / progress.total) * 100);
  }
});
```

Best Practice

Using Axios instance

```
const api = axios.create({
  baseURL: "https://jsonplaceholder.typicode.com"
});

const fetchUsers = async () => {
  const { data } = await api.get("/users");
  setUsers(data);
};
```

You can mention baseURL, default headers, timeout, auth token and common configuration, etc

Instance vs Interceptor

Feature	Axios Instance	Interceptor
Purpose	Reusable axios config	Auto-run logic before request or after response
Handles	baseURL, timeout, headers, etc.	token inject, errors, logging, loader, etc.
Runs	Only when calling	Automatically on every request/response
Without it	Duplicate config in every request	Duplicate logic in every request

Thank You