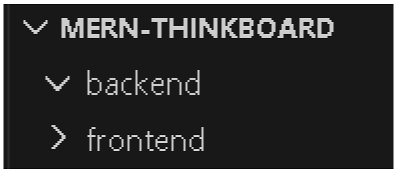
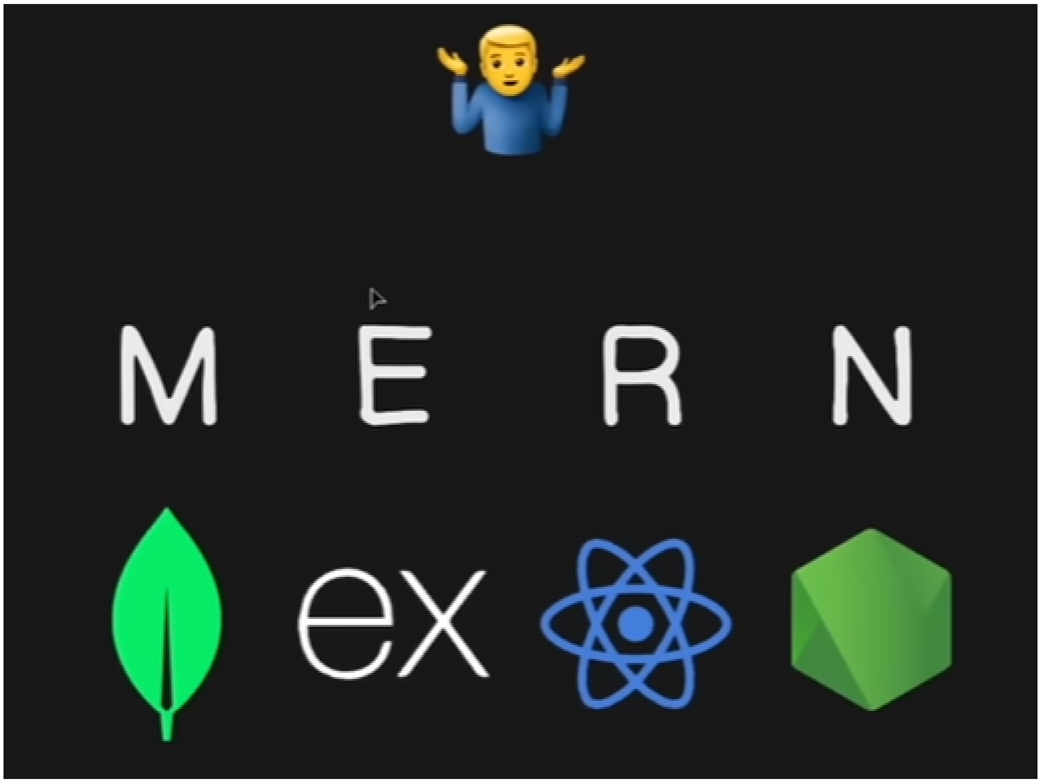
In the VS Code, Create folder structure:



```
⌄ MERN-THINKBOARD
  ⌄ backend
  › frontend
```

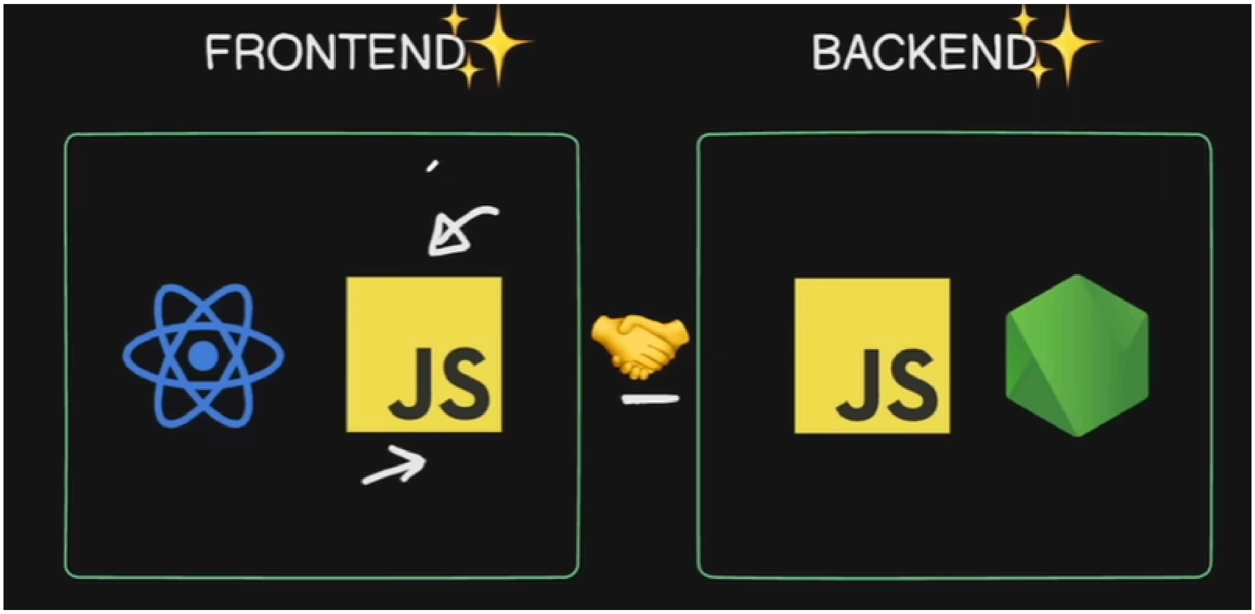
Backend is for API, once the backend completes, then go to the frontend having react application.

For theory: https://app.eraser.io/workspace/GlhY2F7ItehsjZ2z9phZ

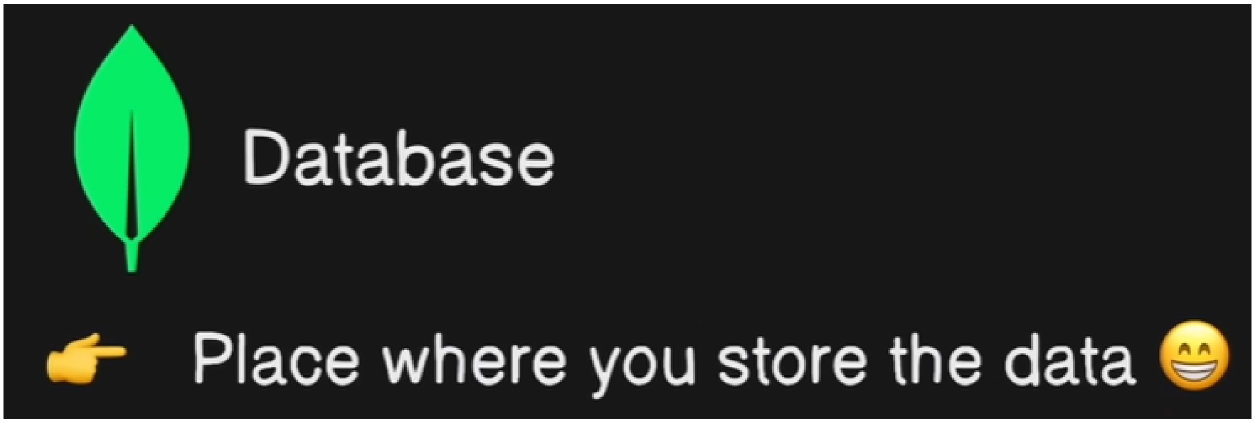


M: MongoDB, E: Express, R: React and N: Node.

MongoDB is the database, Express is the web framework, React is frontend library, nodeJs is the runtime.

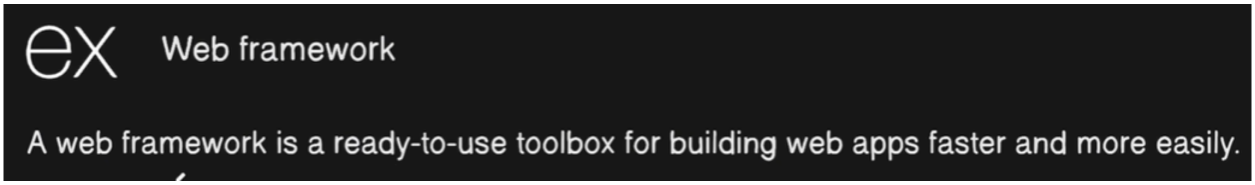


Learn javascript in both frontend and backend.
In frontend, we use react. In the backend, we use node.



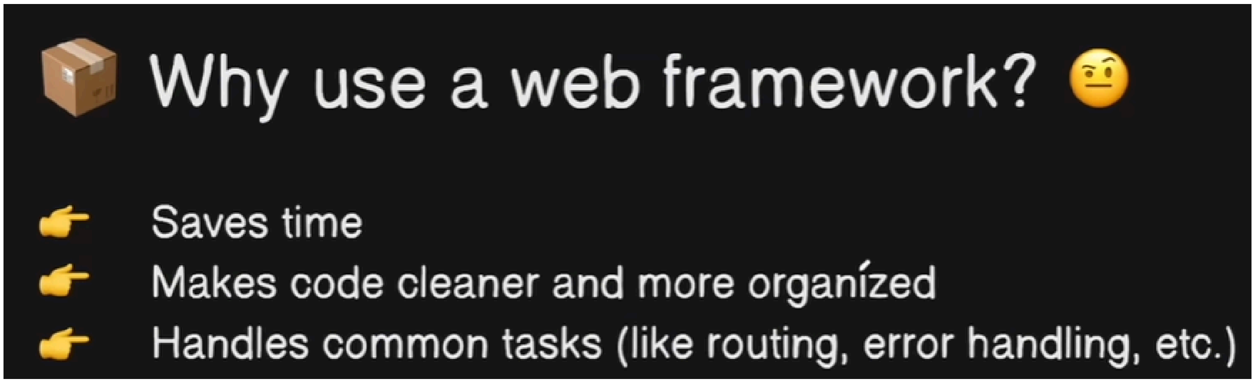
Database

👉 Place where you store the data 😁



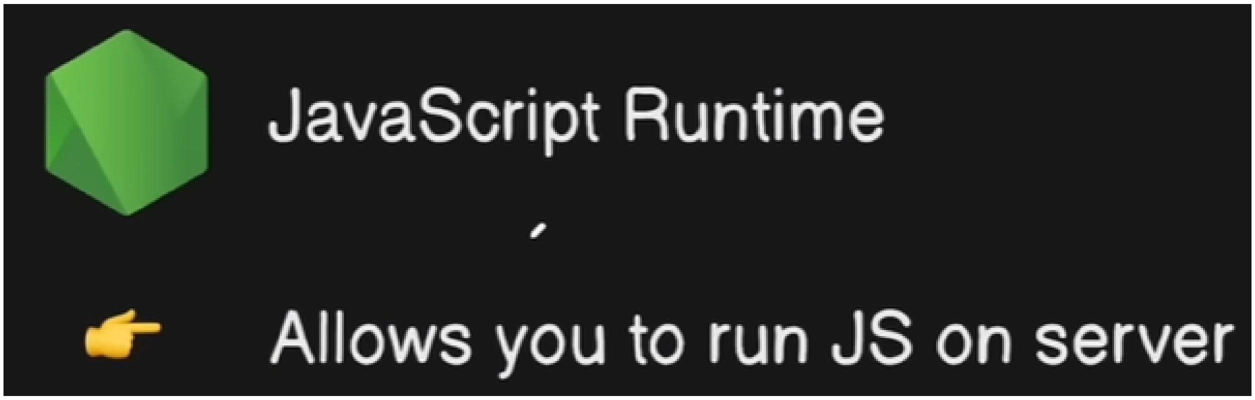
Web framework

A web framework is a ready-to-use toolbox for building web apps faster and more easily.



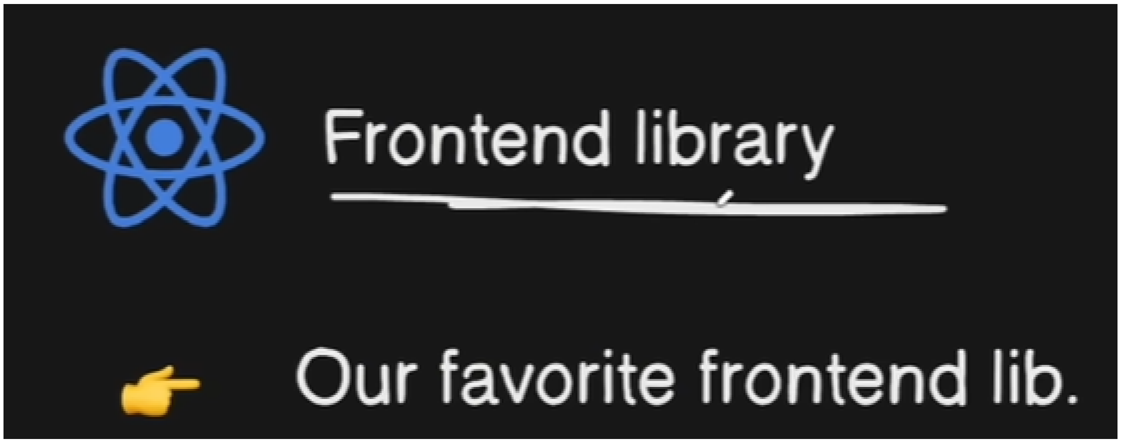
📦 Why use a web framework? 🤨

- 👉 Saves time
- 👉 Makes code cleaner and more organized
- 👉 Handles common tasks (like routing, error handling, etc.)



JavaScript Runtime

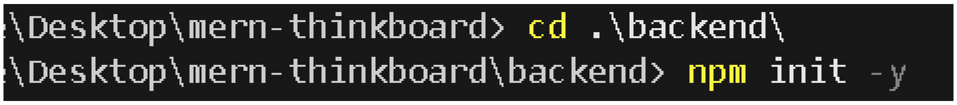
👉 Allows you to run JS on server



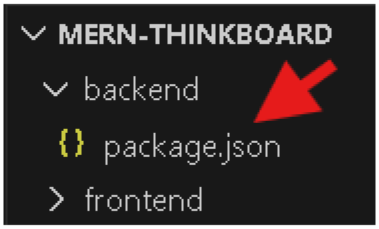
Frontend library

👉 Our favorite frontend lib.

Visit backend folder: and create package.json(using: npm init -y)



```
\Desktop\mern-thinkboard> cd .\backend\
\Desktop\mern-thinkboard\backend> npm init -y
```



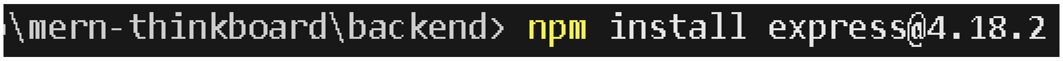
Where we can install different packages.

To clear terminal use: cls or clear

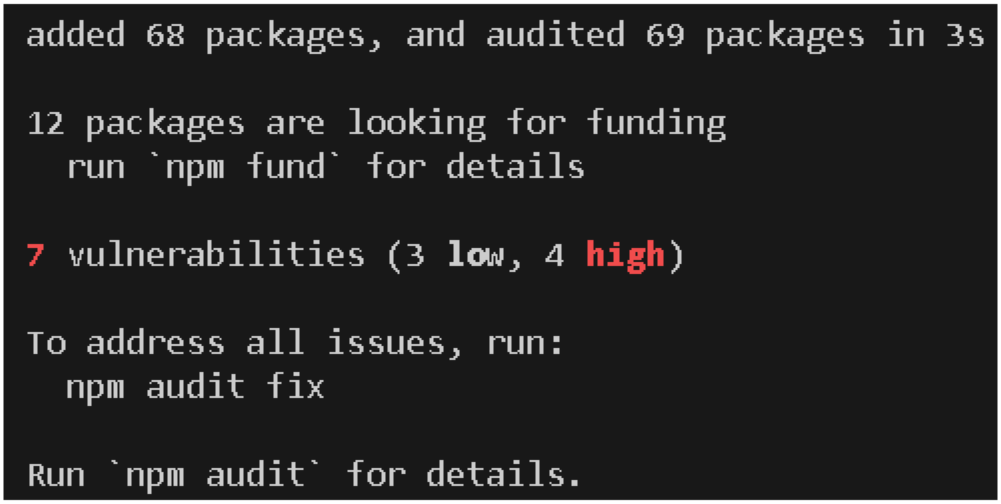
Install packages:

1. Express: with version 4.18.2

Use the command: **npm install express@4.18.2**

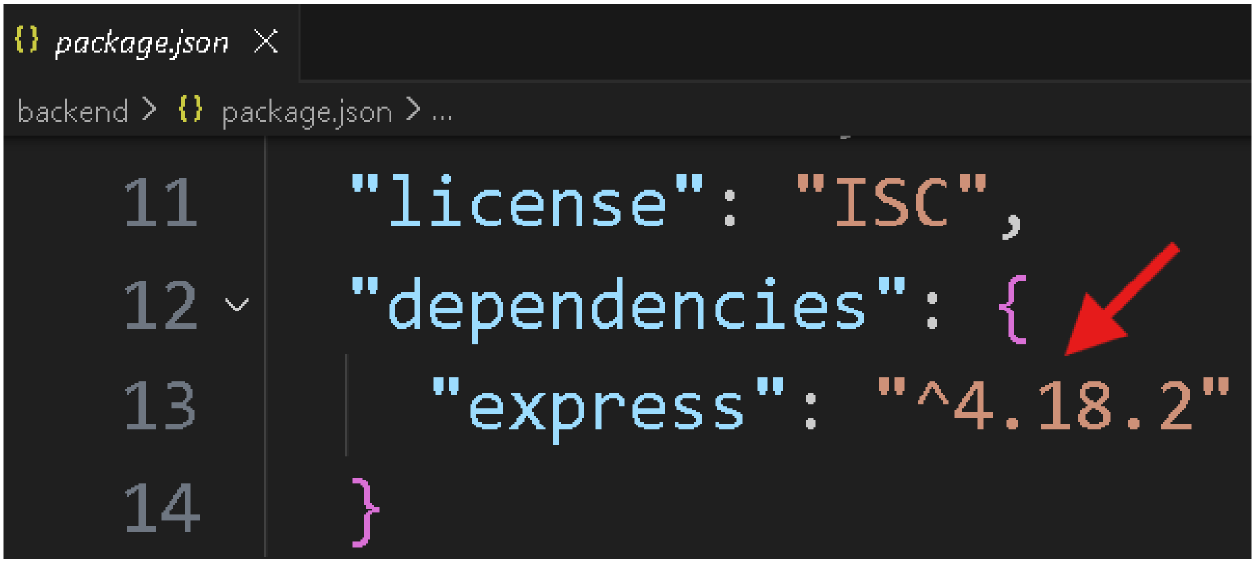


```
\mern-thinkboard\backend> npm install express@4.18.2
```



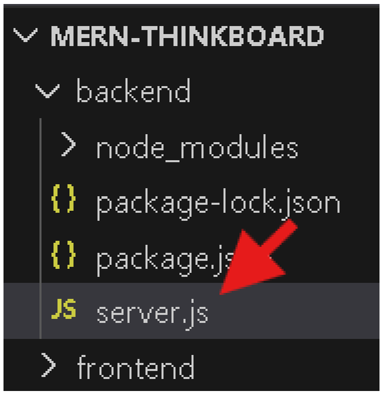
```
added 68 packages, and audited 69 packages in 3s

12 packages are looking for funding
  run `npm fund` for details

7 vulnerabilities (3 low, 4 high)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
```

Adds in the package.json:



```
  "license": "ISC",
  "dependencies": {
    "express": "^4.18.2"
  }
```

Create a server.js file(within backend folder):



```
MERN-THINKBOARD
  backend
    node_modules
    package-lock.json
    package.json
    server.js
  frontend
```

Code in server.js:

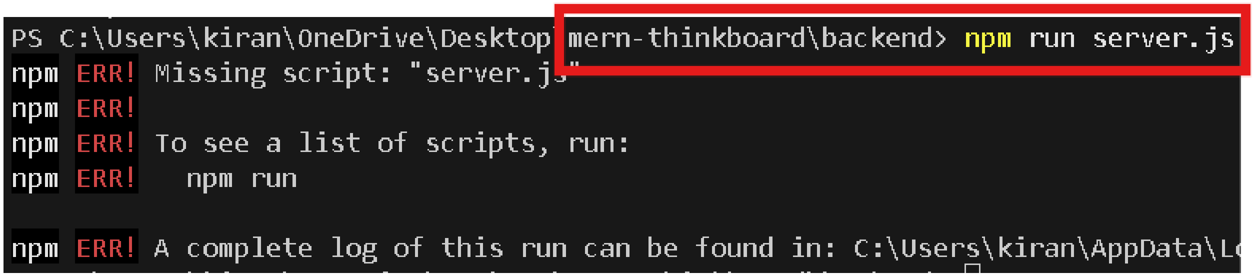
import express from "express"

// Create express app

const app = express()

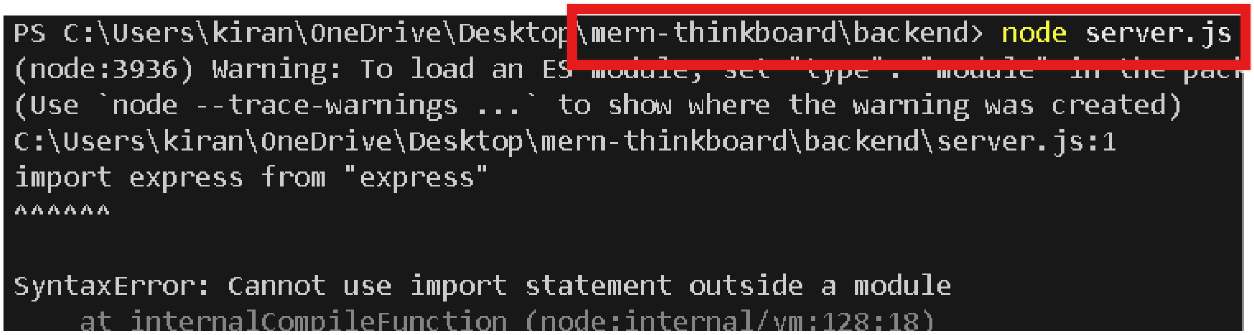
// Connects at port 5001, then prints the statement
app.listen(5001, () => {
    console.log("Server started on PORT: 5001...")
})

To run the server: npm run server.js



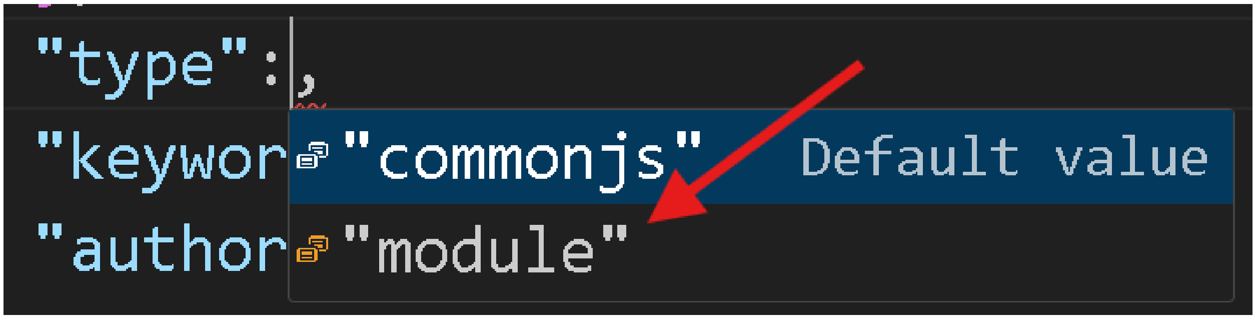
```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run server.js
npm ERR! Missing script: "server.js"
npm ERR!
npm ERR! To see a list of scripts, run:
npm ERR!   npm run

npm ERR! A complete log of this run can be found in: C:\Users\kiran\AppData\Lo
```

Or use command: **node server.js**



```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> node server.js
(node:3936) Warning: To load an ES module, set "type": "module" in the pack
(Use `node --trace-warnings ...` to show where the warning was created)
C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend\server.js:1
import express from "express"
^^^^^^


SyntaxError: Cannot use import statement outside a module
    at internalCompileFunction (node:internal/vm:128:18)
```

To get rid of error.

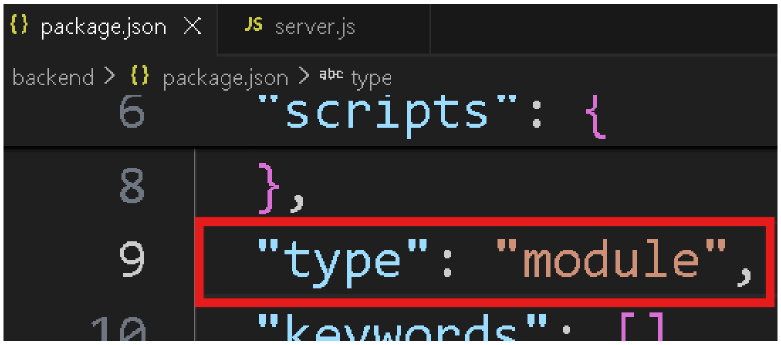


```
"type": ,
"keywor "commonjs"    Default value
"author "module"
```

Default value is commonjs:

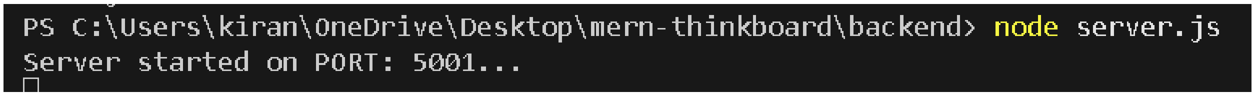
Then need to import like **Eg:** const express = require(“express”)

But in this case we use module, so need to make into module type.



```
  "scripts": {

  },
  "type": "module",
  "keywords": []
```

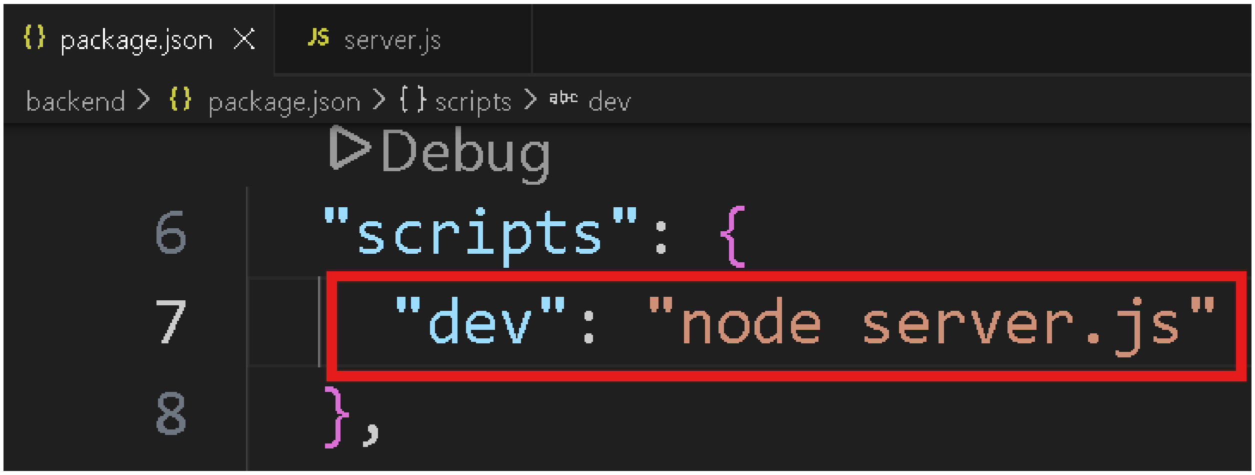
Now run the code again(using **node server.js**):



```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> node server.js
Server started on PORT: 5001...
```

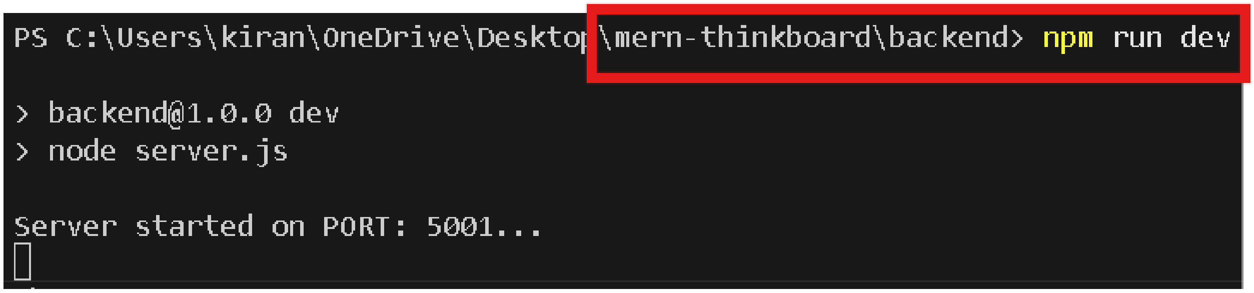
Instead of running using **node** [server.js]() and want to run using **npm run dev**

In package.json:



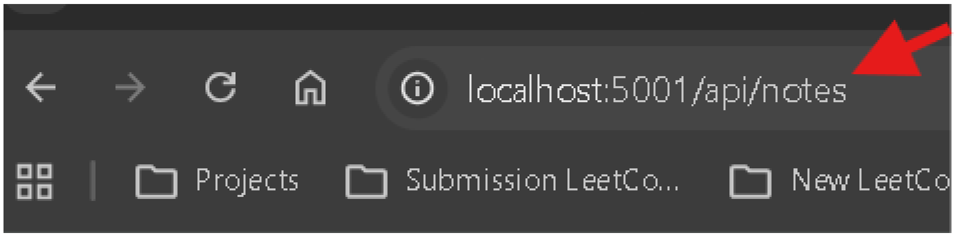
```
"scripts": {
  "dev": "node server.js"
},
```

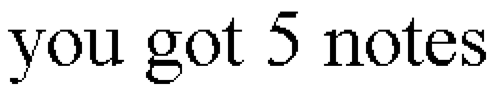
In terminal:



```
PS C:\Users\kiran\OneDrive\Desktop\mern-thinkboard\backend> npm run dev

> backend@1.0.0 dev
> node server.js

Server started on PORT: 5001...
```

Then in the browser: Search([http://localhost:5001/api/notes]())





you got 5 notes

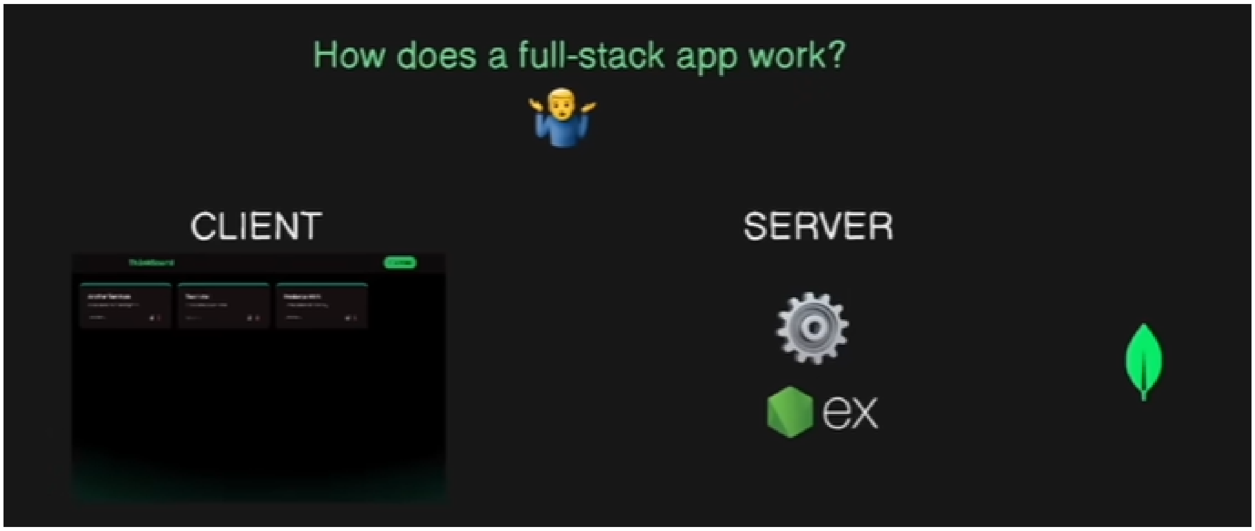
Finally code in [server.js]():

```
import express from "express"
const app = express()

app.get("/api/notes", (req, res) => {
   res.send("you got 5 notes")
})
```

```
app.listen(5001, () => {
    console.log("Server started on PORT: 5001...")
})
```

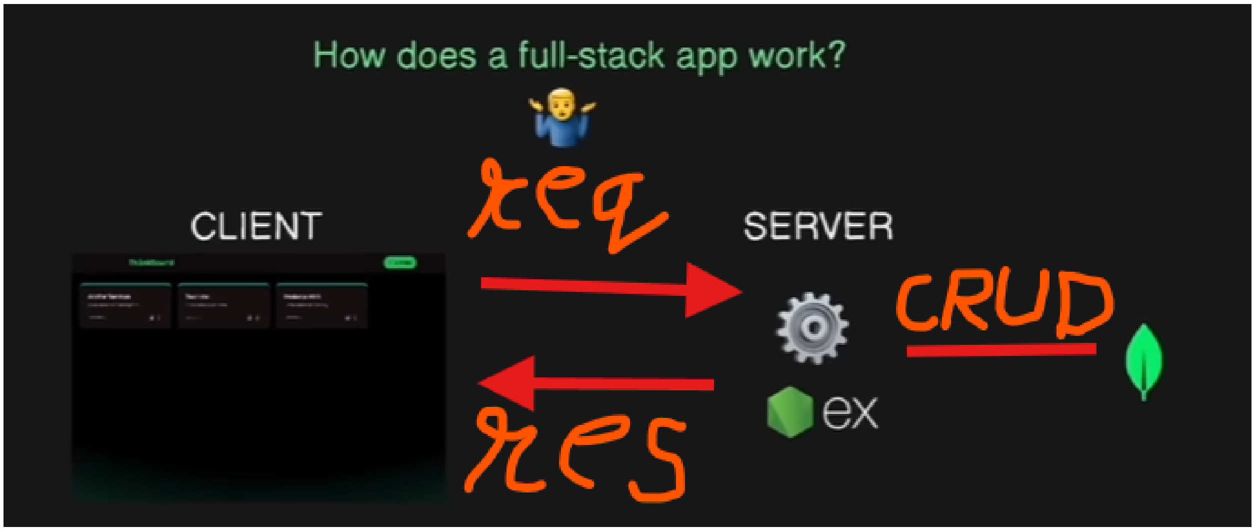
Where we have build our very first api. Where listening to **get request.**
res.send for testing purposes.

How does a full-stack app works?

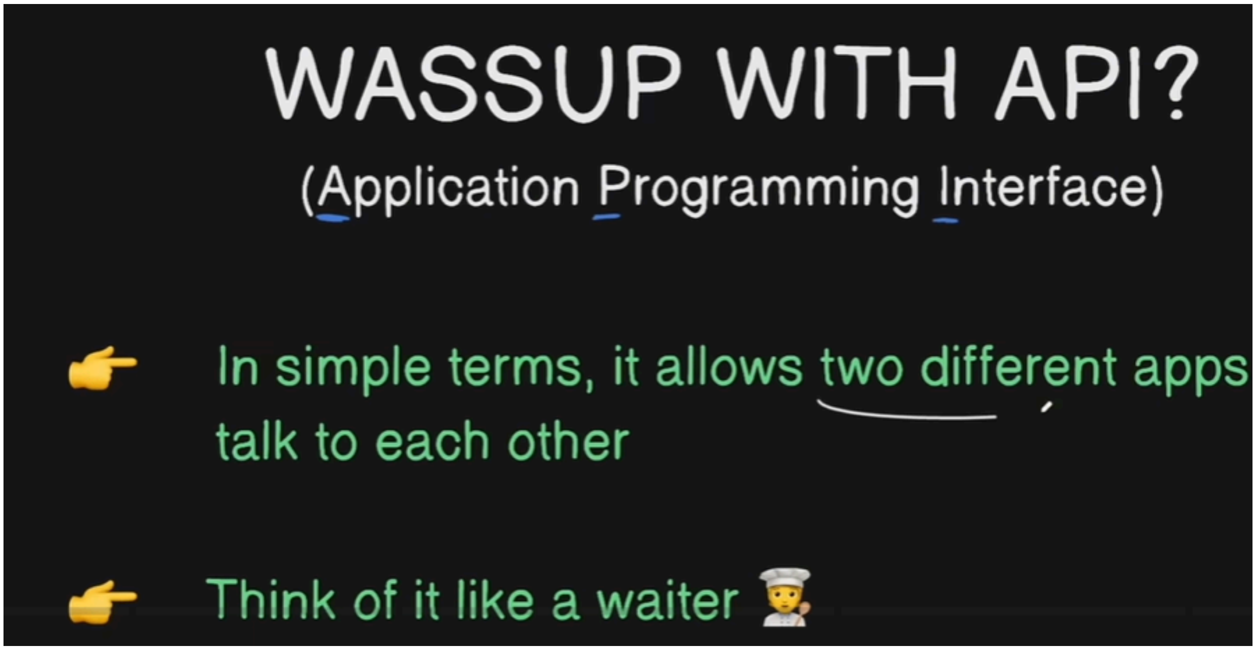


How does a full-stack app work?

Client is the frontend, server is the backend, database to store data.



How does a full-stack app work?

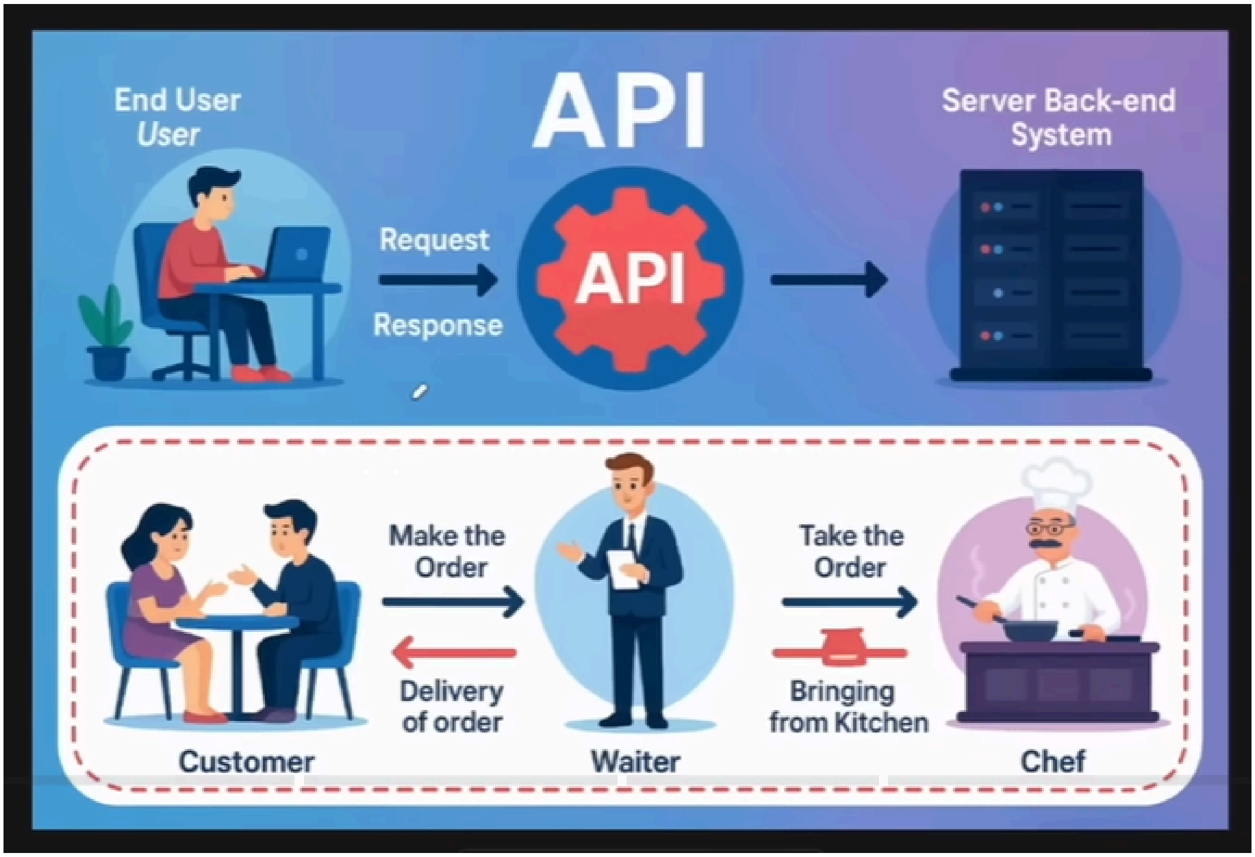


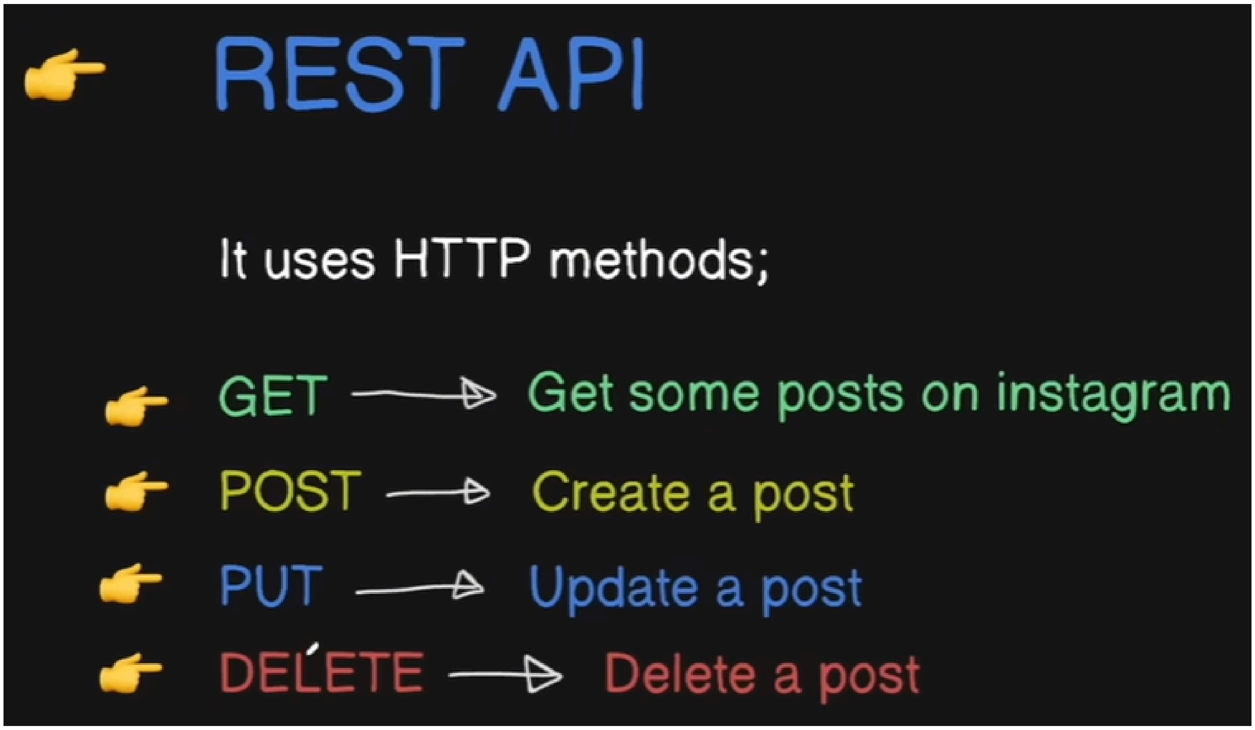
WASSUP WITH API?

(Application Programming Interface)

- 👉 In simple terms, it allows two different apps talk to each other

- 👉 Think of it like a waiter 🧑‍🍳

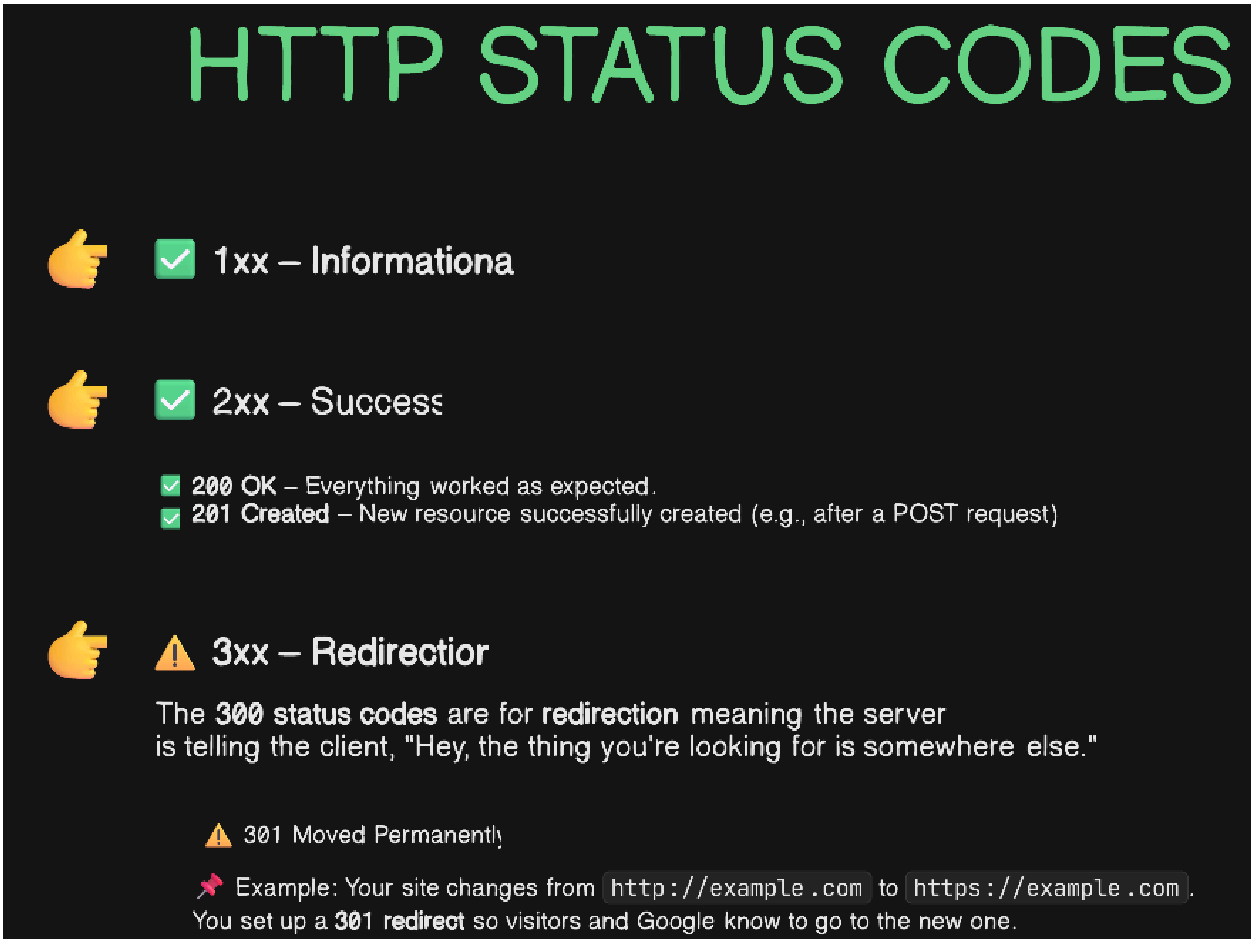




👉 REST API

It uses HTTP methods;

- 👉 GET → Get some posts on instagram
- 👉 POST → Create a post
- 👉 PUT → Update a post
- 👉 DELETE → Delete a post



HTTP STATUS CODES

👉 ✅ 1xx – Informationa

👉 ✅ 2xx – Success

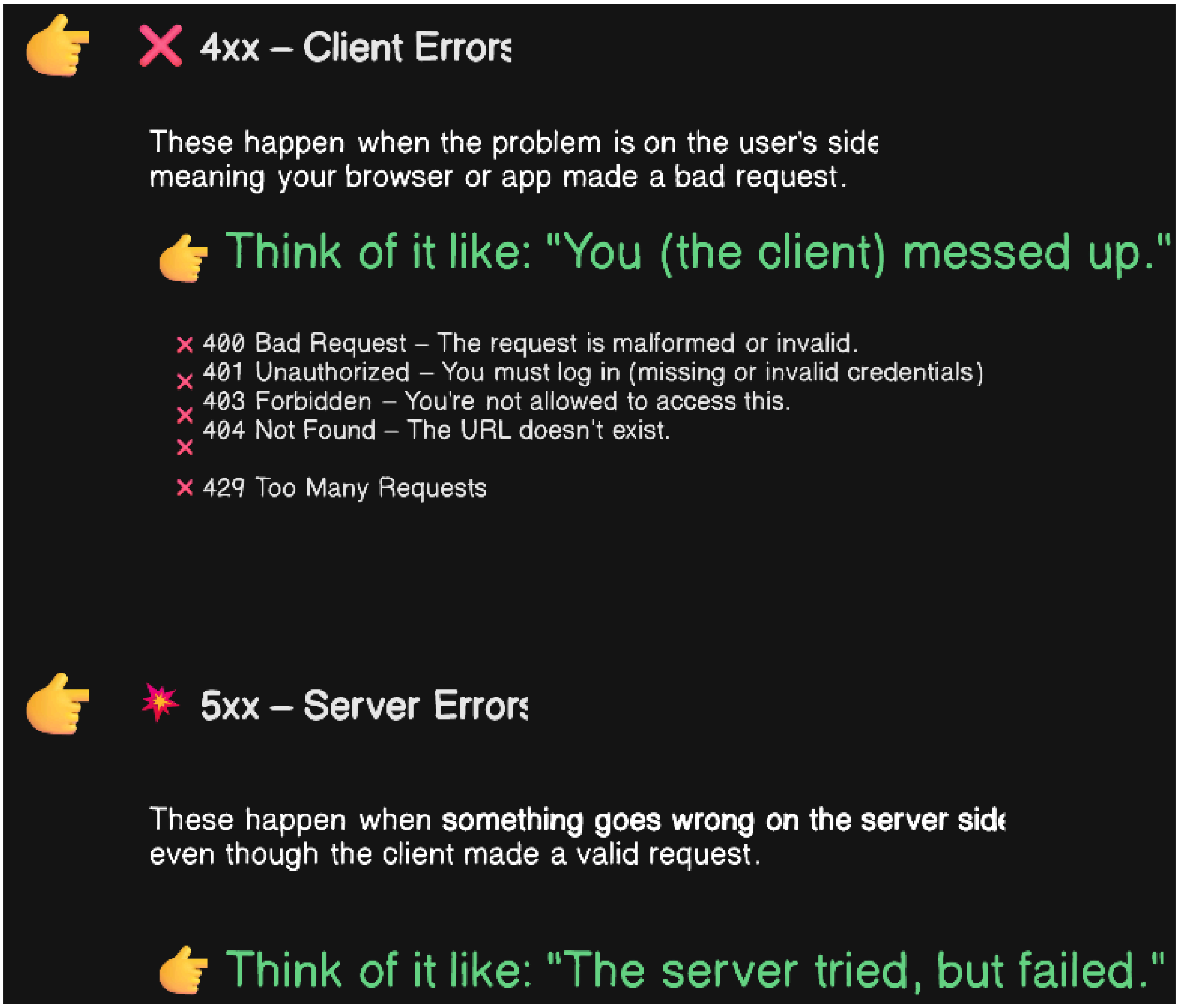
✅ **200 OK** – Everything worked as expected.
✅ **201 Created** – New resource successfully created (e.g., after a POST request)

👉 ⚠️ 3xx – Redirectior

The **300 status codes** are for **redirection** meaning the server
is telling the client, "Hey, the thing you're looking for is somewhere else."

⚠️ 301 Moved Permanentl

📌 Example: Your site changes from `http://example.com` to `https://example.com`.
You set up a **301 redirect** so visitors and Google know to go to the new one.



👉 ❌ 4xx – Client Errors

These happen when the problem is on the user's side
meaning your browser or app made a bad request.

👉 Think of it like: "You (the client) messed up."

- ✕ 400 Bad Request – The request is malformed or invalid.
- ✕ 401 Unauthorized – You must log in (missing or invalid credentials)
- ✕ 403 Forbidden – You're not allowed to access this.
- ✕ 404 Not Found – The URL doesn't exist.
- ✕
- ✕ 429 Too Many Requests

👉 💥 5xx – Server Errors

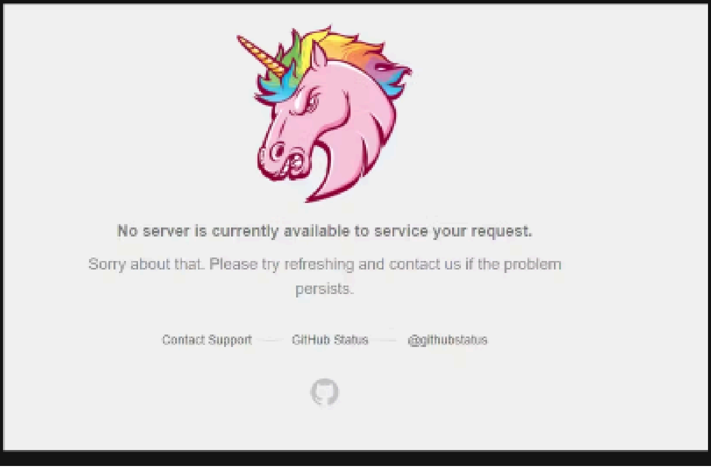
These happen when **something goes wrong on the server side**
even though the client made a valid request.

👉 Think of it like: "The server tried, but failed."

💥 **500 Internal Server Error** – Something broke on the server

💥 **503 Service Unavailable** – Server is temporarily overloaded or down

503 - Github 😵‍💫



No server is currently available to service your request.

Sorry about that. Please try refreshing and contact us if the problem persists.

Contact Support — GitHub Status — @githubstatus