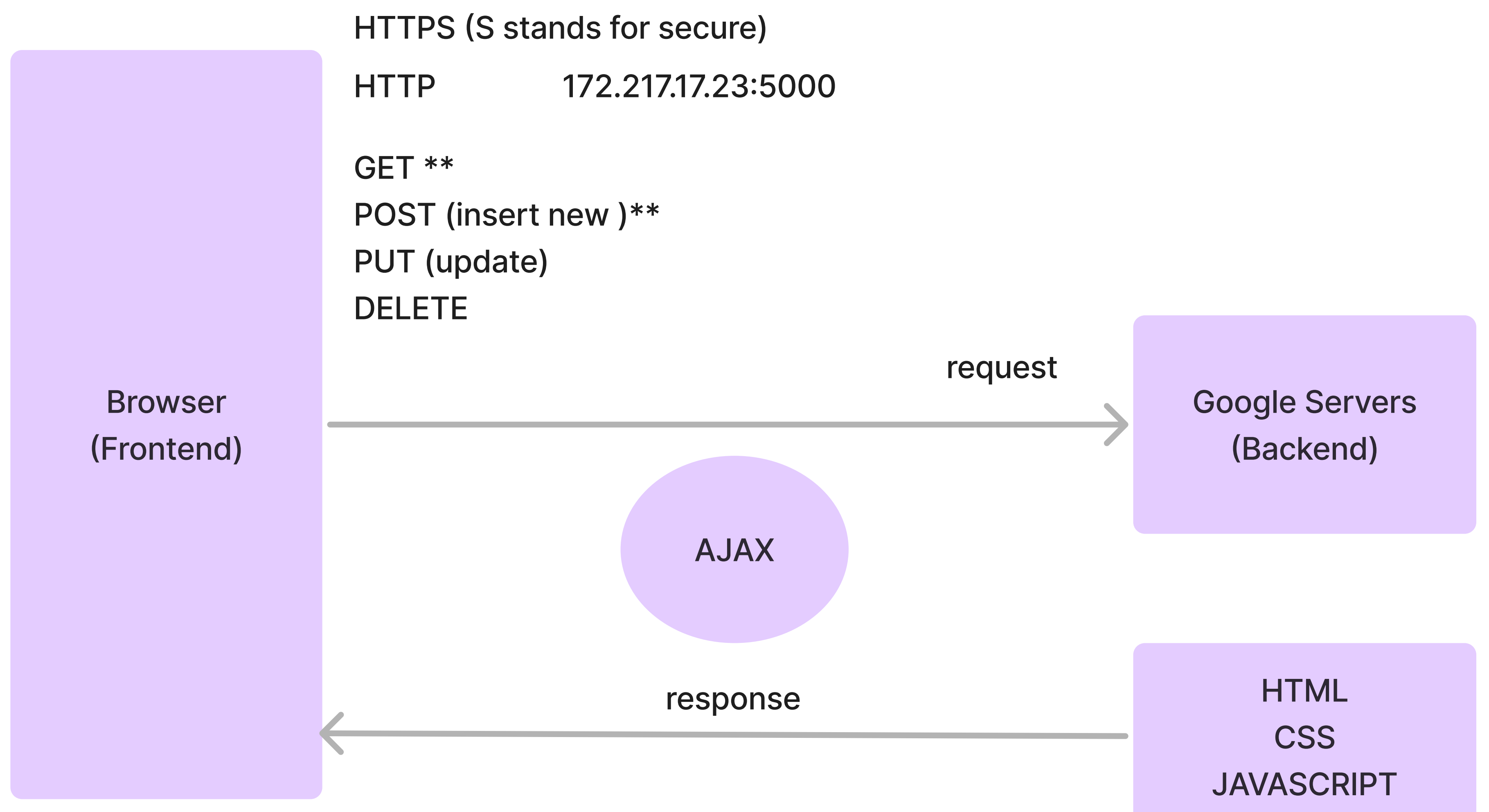


HTTPS (S stands for secure)

HTTP 172.217.17.23:5000

GET **
POST (insert new)**
PUT (update)
DELETE

Browser
(Frontend)

request

Google Servers
(Backend)

AJAX

response

HTML
CSS
JAVASCRIPT

Network Security
CORS - Cross Origin Resource Sharing

frontend (localhost:3000) - example.com
frontend2 (localhost:4000) - fakeExample.com
backend (localhost:5000)

backend, explicitly configure it to be only accessible to the frontend
if and only if its domain is localhost:3000

assets = any files related to images
documents = HTML, JS, CSS, JSON

HTTP Status Codes
1XX
2XX **
3XX **
4XX **
5XX

2XX - ALL GOOD
3XX - NOTHING CHANGED
4XX - WE SCREWED UP

JSON (JavaScript Object Notation)

- Written in .json file

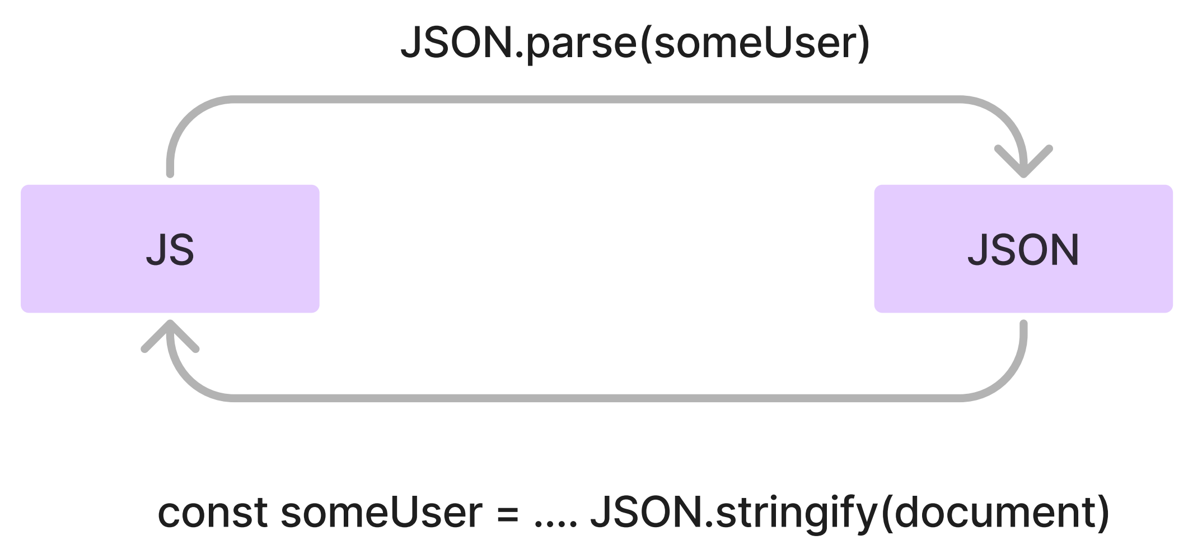
someUser.json file

```
[
{
  “firstName: “John”;
  “lastName”: “Smith”;
},
{
  “firstName: “John”;
  “lastName”: “Smith”;
},
{
  “firstName: “John”;
  “lastName”: “Smith”;
},
{
  “firstName: “John”;
  “lastName”: “Smith”;
}
]
```

JavaScript Object

- Written in .js file

```
const someUser = {
  firstName = “John”,
  lastName = “Smith”,
}
```

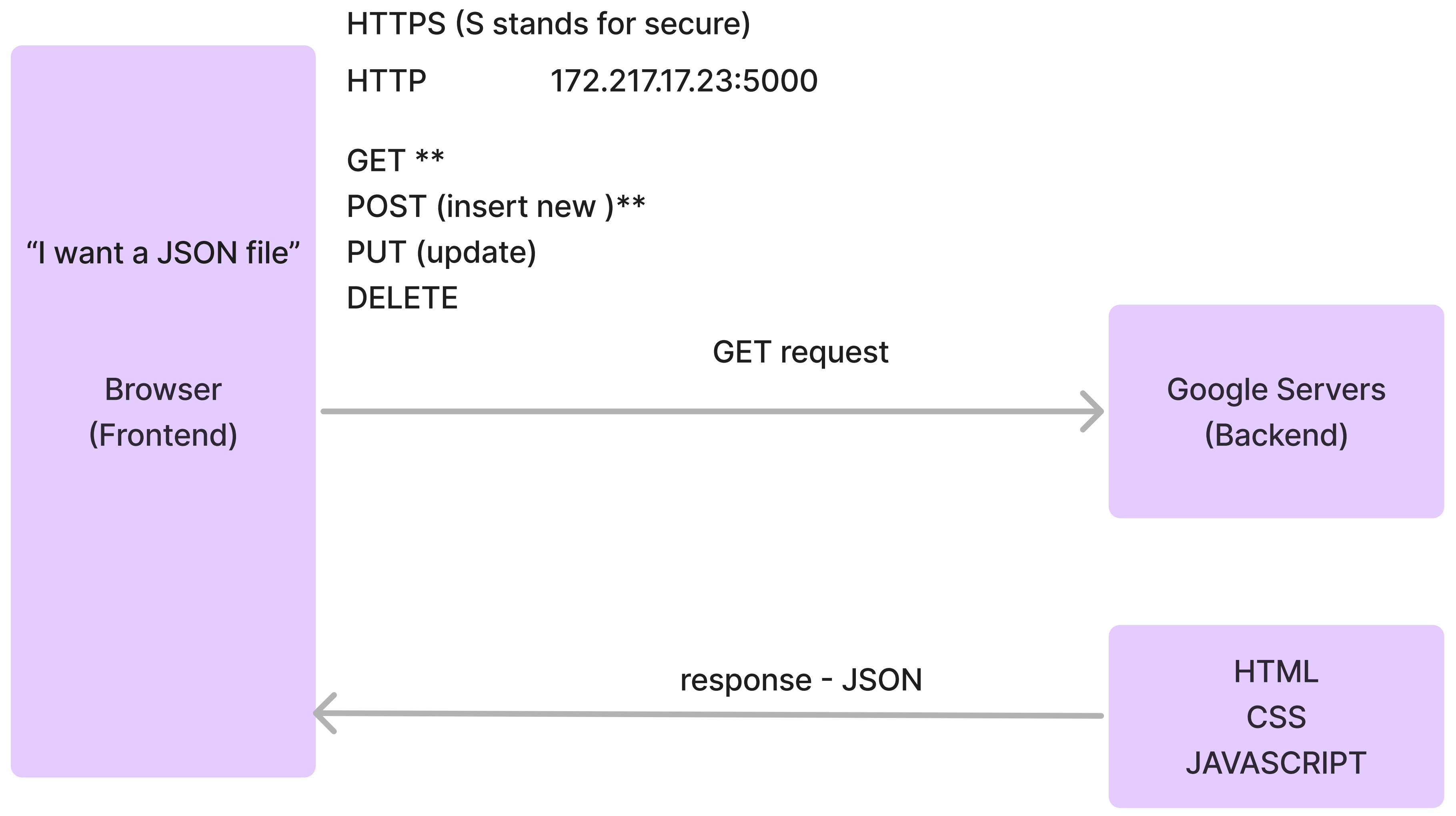


JSON.parse(someUser)

JS

JSON

const someUser = ... JSON.stringify(document)



HTTPS (S stands for secure)

HTTP 172.217.17.23:5000

GET **
POST (insert new)**
PUT (update)
DELETE

“I want a JSON file”

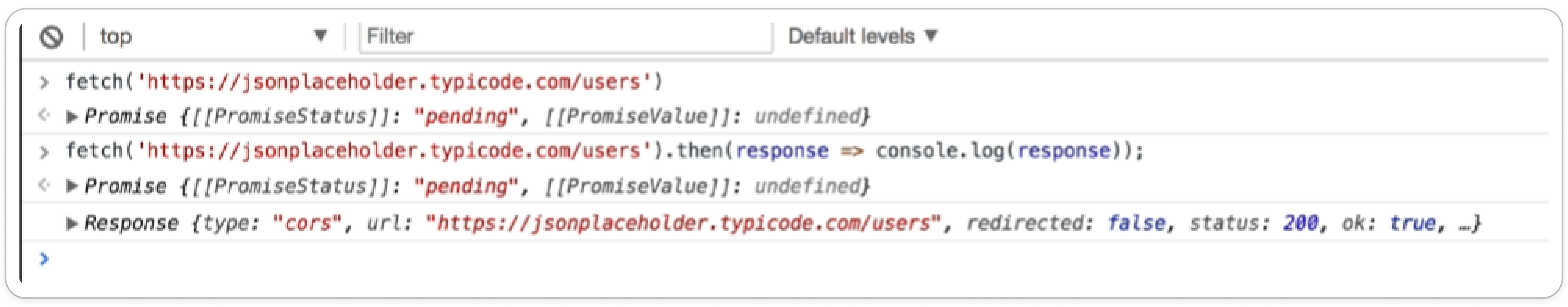
Browser
(Frontend)

GET request

Google Servers
(Backend)

response - JSON

HTML
CSS
JAVASCRIPT



```
> fetch('https://jsonplaceholder.typicode.com/users')
<· ▶ Promise {[[PromiseStatus]]: "pending", [[PromiseValue]]: undefined}
> fetch('https://jsonplaceholder.typicode.com/users').then(response => console.log(response));
<· ▶ Promise {[[PromiseStatus]]: "pending", [[PromiseValue]]: undefined}
   ▶ Response {type: "cors", url: "https://jsonplaceholder.typicode.com/users", redirected: false, status: 200, ok: true, …}
>
```