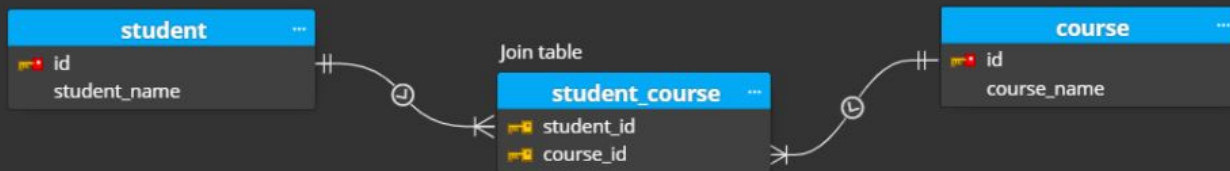


FullStack #WebDevelopment Bootcamp

Course Week 5

Relationships: Recap

Many to many relationship



One to many relationship



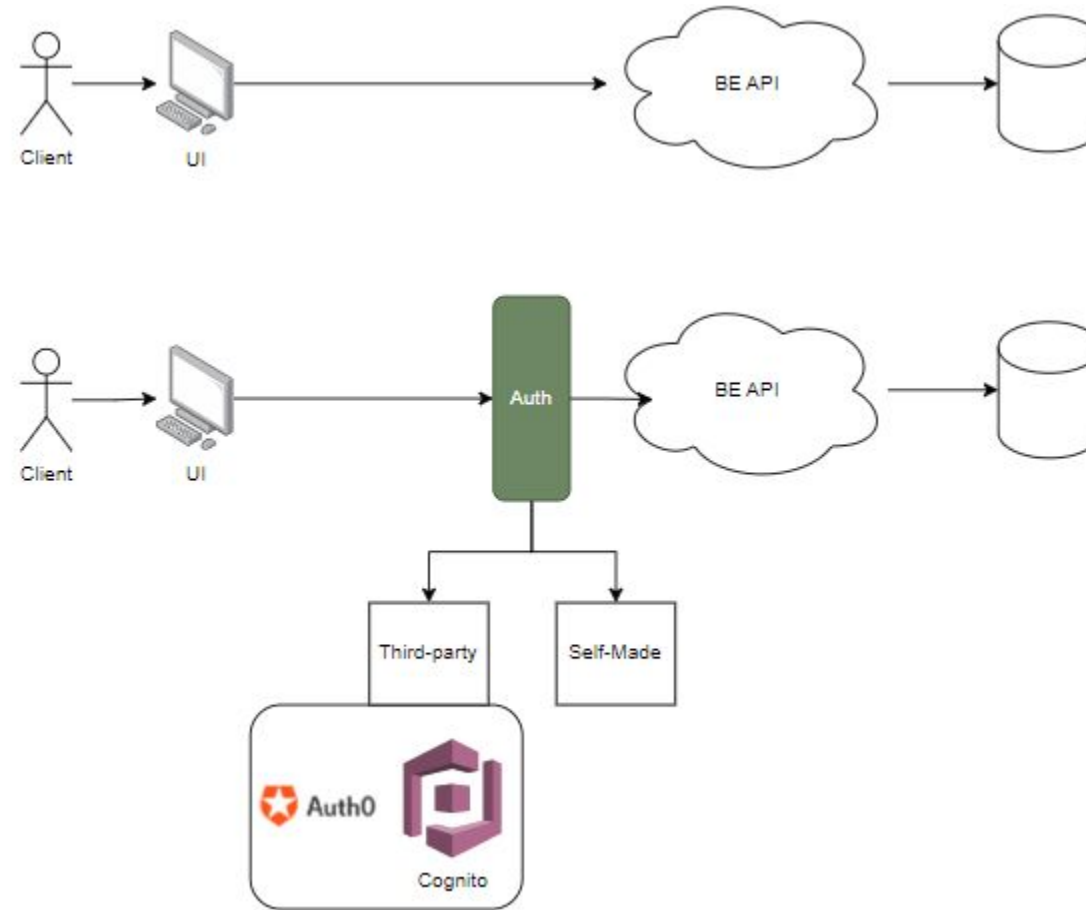
One to one relationship



Authentication is a term that refers to the process of proving that some fact or some document is genuine. In computer science, this term is typically associated with proving a user's identity. Usually, a user proves their identity by providing their credentials, that is, an agreed piece of information shared between the user and the system.

Source: <https://auth0.com/intro-to-iam/what-is-authentication>

Authentication: Layer



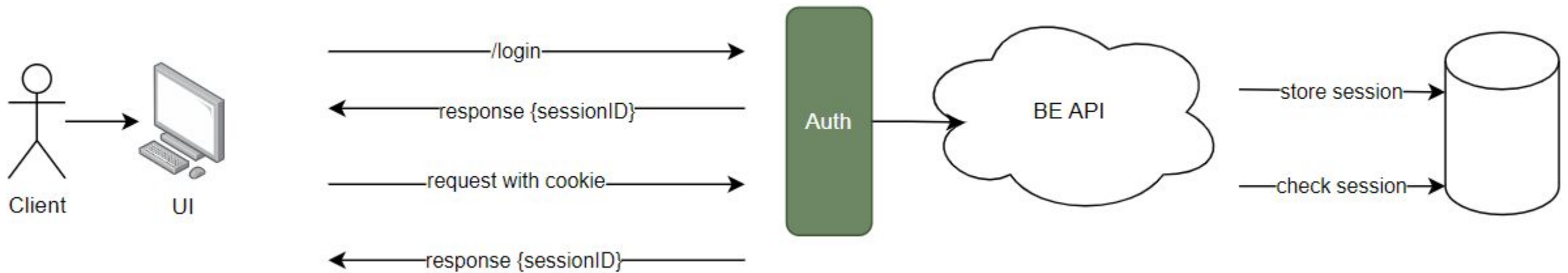
Authentication: Self-Made vs Third-Party

Self-Made

- You secure your user data (e.g.: passwords, etc) and you are not a security expert
- You must re-implement the wheel for different type of authentication (Single-Sign-On, IDP, etc)
- You must take care of scalability

Third-Party

- They secure the user data (maintained by security experts)
- Offers libraries for different platforms
- Scalability: designed for many users
- Offers integration with various other vendors (Google, Facebook, etc.)



This is a **stateful** authentication (the user authentication information is stored on the server)
The storing of the sessions represents a bottleneck in a distributed system

Authentication: Session Login Response Sample



× Headers Preview Response Cookies Timing

▼ General General

Request URL: `http://fr.localhost.com:8082/auth`
Request Method: `OPTIONS`
Status Code: ● 200
Remote Address: `127.0.0.1:8082`

▼ Response Headers view source

Access-Control-Allow-Headers: `x-requested-with, authorization, Content-Type, Authorization, Access-Control-Allow-Headers, credential, X-XSRF-TOKEN, Cookie, Set-Cookie, withCredentials, x-frame-options`
Access-Control-Allow-Methods: `POST, GET, OPTIONS, DELETE`
Access-Control-Allow-Origin: `*`
Access-Control-Max-Age: `3600`
Content-Length: `0`
Date: `Tue, 18 Oct 2016 10:37:45 GMT`
Set-Cookie: `JSESSIONID=C367245309E4E80606066FDCFBEE43;path=/
Set-Cookie: JSESSIONID=C367245309E4E80606066FDCFBEE43`

▼ Request Headers view source

Accept: `/*/*`
Accept-Encoding: `gzip, deflate, sdch`
Accept-Language: `fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4`
Access-Control-Request-Headers: `content-type`
Access-Control-Request-Method: `POST`
Cache-Control: `no-cache`
Connection: `keep-alive`
Host: `fr.localhost.com:8082`
Origin: `http://localhost:4200`
Pragma: `no-cache`

Authentication: Session Request Sample

Request URL: http://localhost:8080/hello
Request Method: GET
Status Code: 🟢 200
Remote Address: [::1]:8080
Referrer Policy: no-referrer-when-downgrade

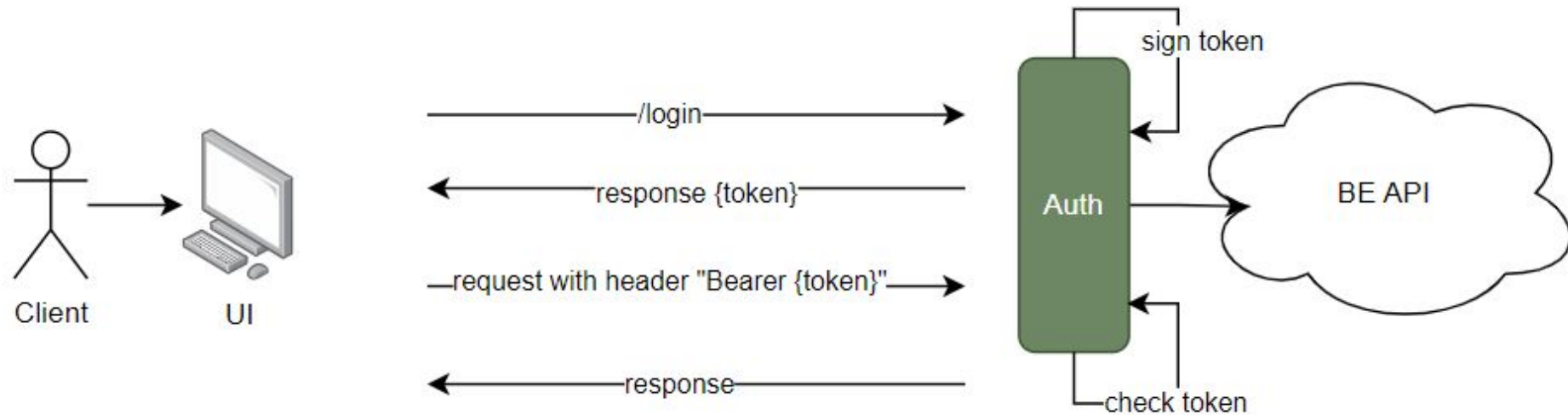
▼ Response Headers [view source](#)

Connection: keep-alive
Content-Length: 0
Date: Wed, 13 May 2020 03:12:48 GMT
Keep-Alive: timeout=60

▼ Request Headers [view source](#)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,zh-TW;q=0.8
Cache-Control: no-cache
Connection: keep-alive
Cookie: JSESSIONID=A3ACED21BAFF98584864286DEEE6F83A
Host: localhost:8080

Authentication: Token



This is a **stateless** authentication (nothing relating to the user authentication information is stored)

Used more for distributed systems

Token lifetime is usually very short (2h), that is why it is accompanied by a “refresh_token”



HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

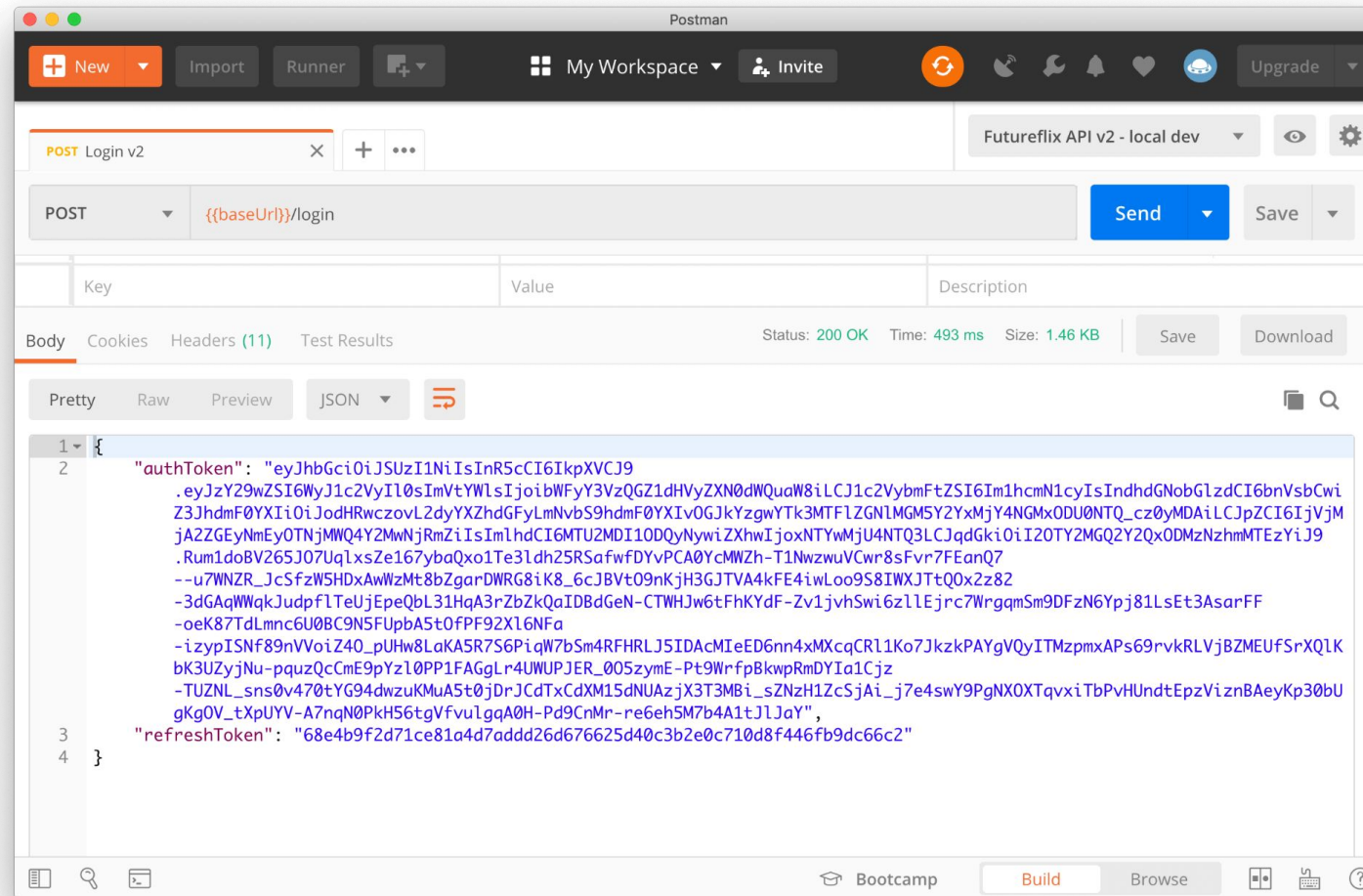
PAYLOAD: DATA

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

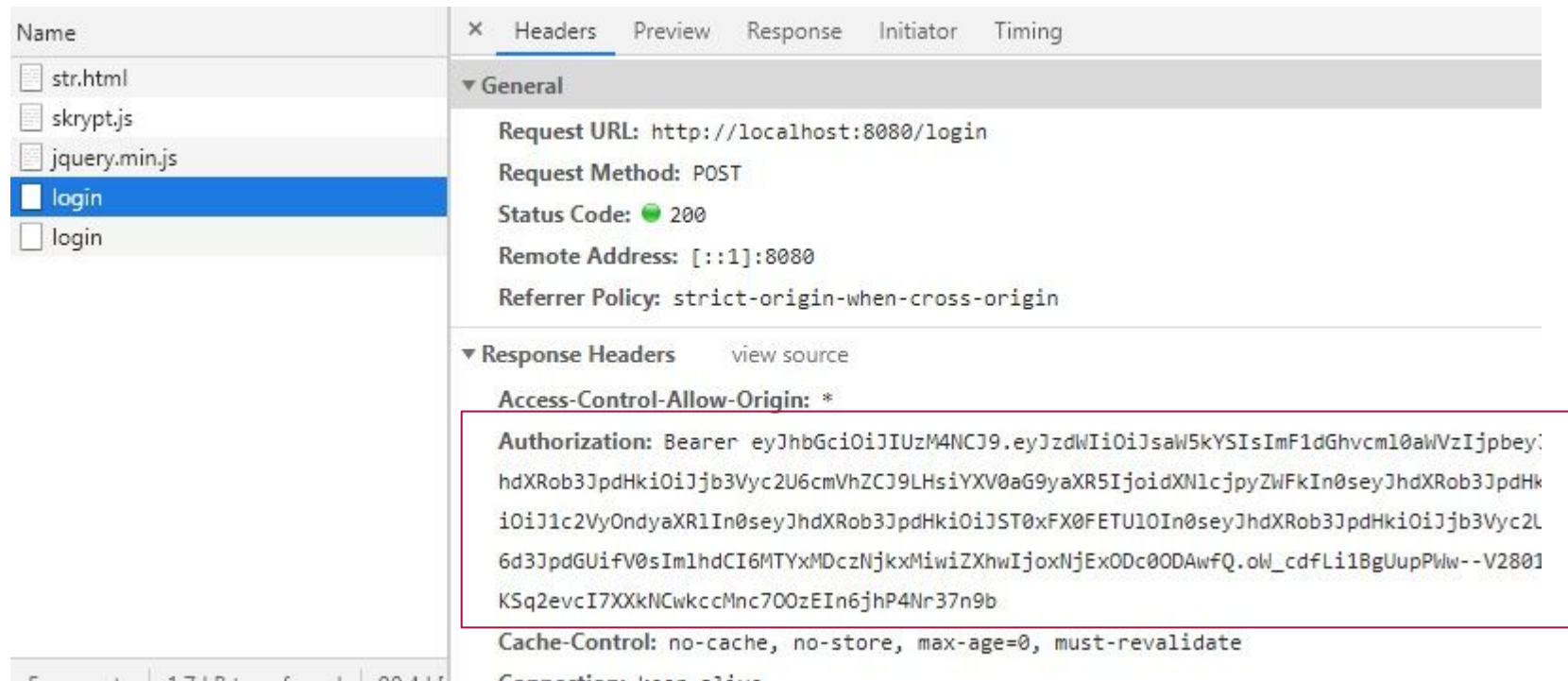
VERIFY SIGNATURE

```
HMACSHA256(
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    your-256-bit-secret
) ☐ secret base64 encoded
```

Authentication: Token Response Sample



Authentication: Token Request Sample



The screenshot displays the 'Headers' tab of a web browser's developer tools. The left pane shows a list of resources, with 'login' selected. The right pane shows the details of the selected resource, which is a POST request to 'http://localhost:8080/login'. The status code is 200 (OK). The 'Response Headers' section is expanded, and the 'Authorization' header is highlighted with a red box. The 'Authorization' header value is a Bearer token: 'eyJhbGciOiJIUzI1IiwiaW5kYSIsImF1dGhvcml0awVzIjpbe...'. The 'Access-Control-Allow-Origin' header is also visible, set to '*'. The 'Cache-Control' header is 'no-cache, no-store, max-age=0, must-revalidate'.

Name	Value
Request URL	http://localhost:8080/login
Request Method	POST
Status Code	200
Remote Address	[::1]:8080
Referrer Policy	strict-origin-when-cross-origin
Access-Control-Allow-Origin	*
Authorization	Bearer eyJhbGciOiJIUzI1IiwiaW5kYSIsImF1dGhvcml0awVzIjpbe...
Cache-Control	no-cache, no-store, max-age=0, must-revalidate



Guards have a **single responsibility**. They determine whether a given request will be handled by the route handler or not, depending on certain conditions (like permissions, roles, ACLs, etc.)