**API - JWT**

1. **Hướng dẫn tổ chức API trong Project**

**XHR :**

- Dùng **callback** để xử lý. Có sẵn trong trình duyệt.

- **onprogess** : Trạng thái trước khi xử lý 3.

- **onload** : Trạng thái khi đã xử lý xong 4.

- **onreadystatechage** : Theo dõi từ trạng thái 1 đến 4.

- **onerror** : Trả về khi gặp lỗi.

**Fetch :**

- Là **WebAPI** có sẵn trong trình duyệt.

- **Package fetch** để hỗ trợ cho trình duyệt cũ.

- Dùng cho các project nhỏ, đơn giản.

**Axios :**

- Sử dụng trên cả **browser** và **node.js.**

1. **HTTP API**

Có thể trả về **xml** hoặc **json**.

**HTTP :** Một giao thức qua **internet**.

**API :** Là chuẩn giao tiếp giữa **Client** và **Server**. Chỉ quan tâm đến dữ liệu.

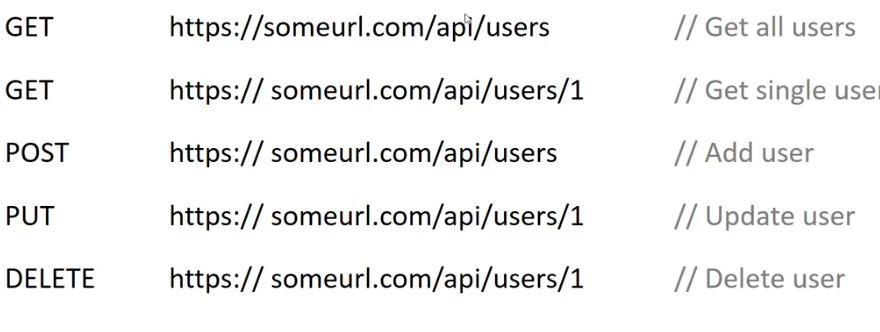
1. **REST API**

**REST API :** Sử dụng giao thức **HTTP** , là một quy chuẩn .

Những ngôn ngữ viết về **websever** đều có thể viết ra **REST API.**

- Có quy chuẩn **build** **url** thế nào cho có ý nghĩa.

- **Client** dùng **Api** do **Server** đinh nghĩa sẵn để cho **server** biết **client** mong gì.



PUT khi gửi lên thì nó thay thế toàn bộ cái cũ bằng nội dung nó gửi lên (replace the entire entity)

PATCH chỉ update cái gửi lên thôi (only update the fields that were supplied).

Nên nếu dùng PUT mà không gửi hết đầy đủ thì sẽ bị mất dữ liệu

**Chú ý :** Khi viết **REST API** server người ta sẽ không dùng cookie để làm **Authentication.**

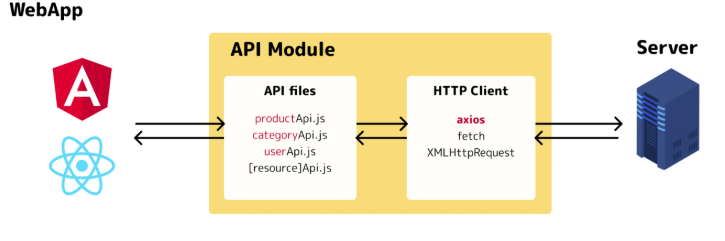
Thường dùng 1 **Header** có **name** **authentication** có chứa **token.**

1. **Trong React**

Tạo ra một **object** và sử dụng ở nhiều nơi.

Tạo 1 file chứa biến môi trường **.env**.

Tạo 1 file axiosClient tất cả các **HTTP request** phải đi qua nó.



1. Phân biệt **Authentication** và **Authorization**

**Authentication** ( xác thực ) **:**

- Xác định bạn là ai.

+ Username + password.

+ Vân tay.

+ Nhận diện gương mặt. ...

**Đặng nhập** thành công vào hệ thống **server** sẽ cho bạn 1 cái **token**.

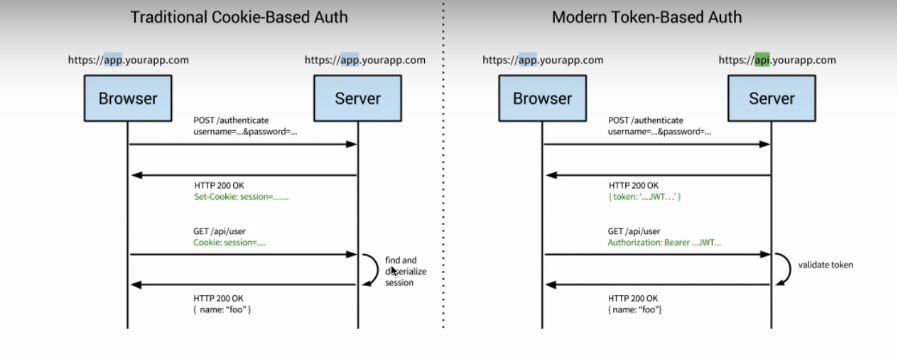
- Khi gặp lỗi nó sẽ trả về 401.

**Authorization** ( ủy quyền ) **:**

- Xác thực rồi -> bước tiếp theo phần quyền.

Các phổ biến nhất để phân quyền là **JWT( Json Web Token ).**

1. So sánh giữa **cookie** vs **token authentication**



**Cookie :**

- **Server** có **State** quản lý những **cookie, session** .

**Authentication :**

- **Server** tạo ra một cái **token 🡪** trình duyệt lưu **token** được lưu trong trình duyệt.

- Không tốn hiệu năng.( **performace** ).

- Có thể chứa data. Chứa nhiều quá thì nó nặng.

1. **sá**