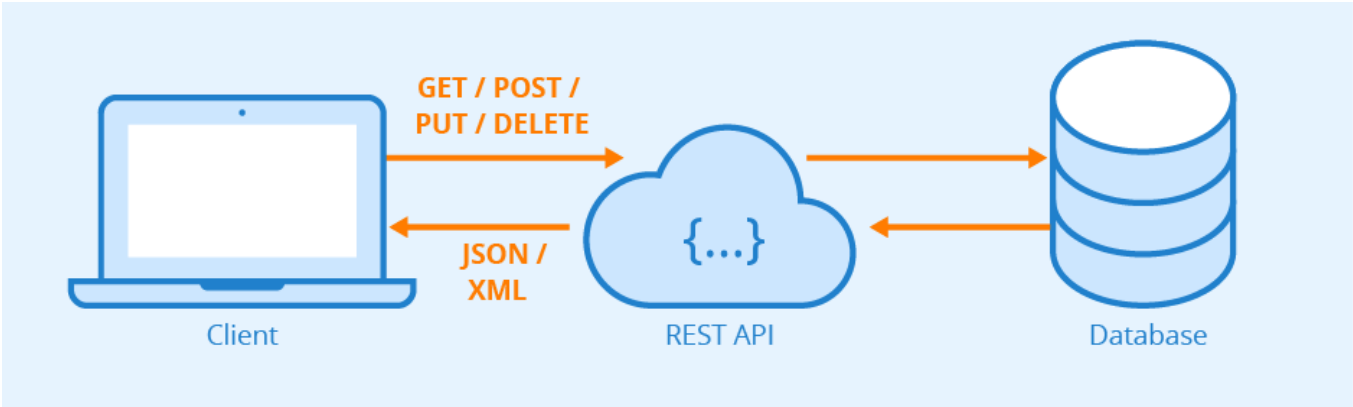
Tổ chức API trong ReactJS project như thế nào?

. API là gì?
. XHR vs Fetch vs Axios, dùng cái nào?
. Tổ chức API module như thế nào?
. Code mẫu sử dụng axios

1. API là gì?

- Đơn giản, nó là chuẩn giao tiếp giữa client và server.
- Client dùng API do server định nghĩa sẵn để nói cho server biết client mong gì.



Một vài ví dụ:

```
# Client muốn lấy 10 products đầu tiên
GET https://js-post-api.herokuapp.com/api/products?_limit=10&_page=1
```

```
# Client muốn xoá product có ID 12345
DELETE https://js-post-api.herokuapp.com/api/products/12345
```

2. XHR vs Fetch vs Axios, dùng cái nào?

Giờ thử thực hiện gọi API này bằng 3 cách xem thế nào nhé

```
GET https://js-post-api.herokuapp.com/api/products?_limit=10&_page=1
```

XHR /XMLHttpRequest/

- Dùng dạng callback.
- Hơi cũ rồi, hiện ít sử dụng trong project.

```
const xhr = new XMLHttpRequest();
const url = 'https://js-post-api.herokuapp.com/api/products?
_limit=10&_page=1'
xhr.open('GET', url);
xhr.responseType = 'json';
xhr.send();

xhr.onload = function () {
  if (xhr.status != 200) { // analyze HTTP status of the response
    alert(`Error ${xhr.status}: ${xhr.statusText}`); // e.g. 404: Not
Found
  } else { // show the result
    console.log(xhr.response)
  }
};
```

Fetch API

- Lưu ý `Fetch API` là WebAPI có sẵn trong trình duyệt.
- Còn package `https://github.com/github/fetch` chỉ là polyfill để hỗ trợ trình duyệt cũ mà thôi.
- Dùng cho các project nhỏ, đơn giản.

```
try {
  const url = 'https://js-post-api.herokuapp.com/api/products?
_limit=10&_page=1';
  const response = await fetch(url);
  const responseJSON = await response.json();
  console.log(responseJSON);
} catch (error) {
  console.log('Failed to fetch products: ', error);
}
```

Axios (recommended)

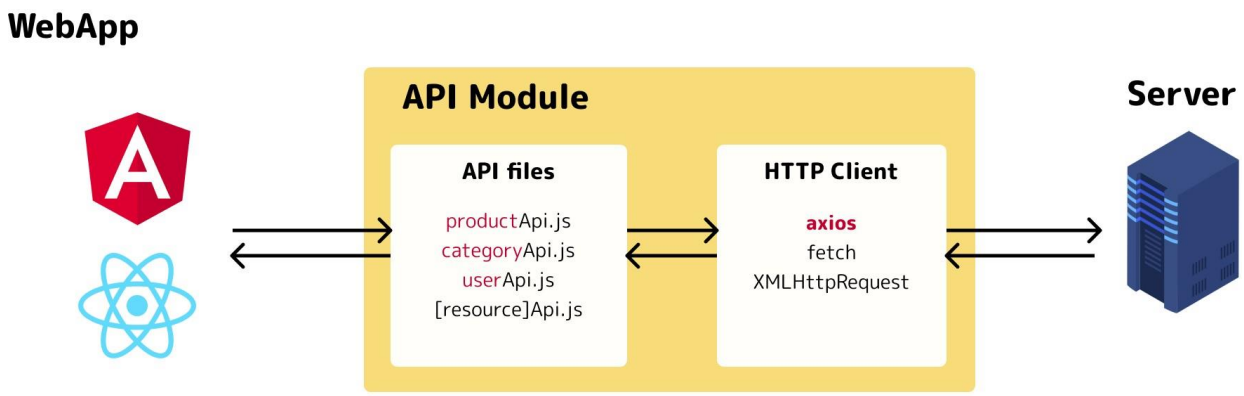
- Sử dụng được cả trên `browser` và `node.js`
- Inteceptors.
- Cancel requests.
- Auto transform JSON data.

```
# Install axios package
npm install --save axios
```

```
try {
  const url = 'https://js-post-api.herokuapp.com/api/products?
_limit=10&_page=1';
  const response = await axios.get(url);
  console.log(response);
} catch (error) {
  console.log('Failed to fetch products: ', error);
}
```

3. Tổ chức API module như thế nào?

- Thiết lập một http client và đảm bảo tất cả các http requests đều phải đi qua nó, nhằm mục đích xử lý những tác vụ chung như:
  - Thêm common headers: content-type, ...
  - Attach thêm token và xử lý expired token.
  - Xử lý lỗi chung.
- Khuyến khích sử dụng `axios` trong project thực tế.




```
src
|__api
|  |__axiosClient.js : http client for our website
|  |__productApi.js : all apis of product resources
|  |__categoryApi.js
|  |__userApi.js
|  |__...
|
|__components
|__features
|__...
|__App.js
```

- Mỗi một file API sẽ bao gồm những API liên quan tới resource đó. VD như `productApi.js` sẽ chứa tất cả APIs của `product`.

```
GET /products # Lấy ds products
GET /products?categoryId=123&page=1 # Lọc products với params
GET /products/:productId # Lấy product by ID
POST /products # Tạo một product mới
PATCH /products/:productId # Cập nhật product có ID là :productId
DELETE /products/:productId # Xoá product có ID là :productId
```

4. Code mẫu sử dụng axios

Setup file .env

```
REACT_APP_API_URL=https://js-post-api.herokuapp.com/api
```

Setup axiosClient.js

```
// api/axiosClient.js
import axios from 'axios';
import queryString from 'query-string';

// Set up default config for http requests here
// Please have a look at here `https://github.com/axios/axios#request-
config` for the full list of configs
const axiosClient = axios.create({
  baseURL: process.env.REACT_APP_API_URL,
  headers: {
    'content-type': 'application/json',
  },
  paramsSerializer: params => queryString.stringify(params),
});

axiosClient.interceptors.request.use(async (config) => {
  // Handle token here ...
  return config;
})

axiosClient.interceptors.response.use((response) => {
  if (response && response.data) {
    return response.data;
  }

  return response;
}, (error) => {
  // Handle errors
  throw error;
});

export default axiosClient;
```

Setup productApi.js

```
// api/productApi.js
class ProductApi {
  getAll = (params) => {
    const url = '/products';
    return axiosClient.get(url, { params });
  };
}

const productApi = new ProductApi();
export default productApi;
```

Sử dụng productApi trong ReactJS component

```
function App() {
  const [productList, setProductList] = useState([]);

  useEffect(() => {
    const fetchProductList = async () => {
      try {
        const params = { _page: 1, _limit: 10 };
        const response = await productApi.getAll(params);
        console.log('Fetch products successfully: ', response);

        setProductList(response.data);
      } catch (error) {
        console.log('Failed to fetch product list: ', error);
      }
    }

    fetchProductList();
  }, []);

  return <ProductList productList={productList} />;
}
```

 Túm lại

- Nên tổ chức APIs để handle APIs tốt hơn, dễ dàng hơn.
- Khuyến khích sử dụng thư viện `axios` trong project thực tế.
- Lưu ý việc tạo ra 1 object duy nhất để sử dụng cho các file APIs.

 Link tham khảo

- XMLHttpRequest
- XMLHttpRequest from Javascript.info
- Using XMLHttpRequest
- Fetch API
- Fetch API Polyfill
- Axios
- Make http request with axios