API là gì ?

API là viết tắt của Application Programming Interface (giao diện lập trình ứng dụng), là giao diện cho nhiều ứng dụng trao đổi, tương tác với nhau. Với API, các lập trình viên có thể tiếp cận, truy xuất dữ liệu từ máy chủ thể hiện chúng trên ứng dụng phần mềm hoặc website của mình một cách dễ dàng hơn.

Tìm hiểu Web

Web API: là hệ thống API được sử dụng trong các hệ thống website. Hầu hết các website đều ứng dụng đến Web API cho phép bạn kết nối, trao đổi thông tin, dữ liệu qua lại hoặc cập nhật cơ sở dữ liệu. Ví dụ: Bạn thiết kế chức năng login thông Google, Facebook, …

Mỗi khi thực hiện truy xuất thông tin, Web API sẽ trả lại dữ liệu ở dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS.

Một Web API được xây dựng tuân theo kiến trúc REST thì được gọi là RESTful API.

REST (REpresentational State Transfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Mặc dù REST có thể được sử dụng với mọi giao thức, nhưng nó thường dùng HTTP cho Web API.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource (tài nguyên – ví dụ như là tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu di động,… Nó cũng bao gồm các trạng thái tài nguyên được định dạng và truyền tải qua HTTP). RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.

Ưu điểm của RESTful API:

Giúp cho ứng dụng rõ ràng hơn

Dữ liệu được trả về với nhiều định dạng khác nhau như: xml, html, json….

Code đơn giản và ngắn gọn

Chú trọng vào tài nguyên của hệ thống

Cách hoạt động của RESTful API

REST hoạt động chủ yếu dựa vào các giao thức HTTP:

GET (SELECT): Trả về một Resource hoặc một danh sách Resource.

POST (CREATE): Tạo mới một Resource.

PUT (UPDATE): Cập nhật thông tin cho Resource.

DELETE (DELETE): Xoá một Resource.

RESTful API không sử dụng session và cookie, nó sử dụng một access\_token với mỗi request

Dữ liệu trả về của API thường có cấu trúc như sau:

*{*

*"status\_code": 200,*

*"data": [*

*{*

*"name": "AB",*

*"email": " AB@example.com",*

*"address": "Da Nang"*

*},*

*],*

*error\_messages: ""*

*}*

Những ràng buộc kiến trúc cơ bản của RESTful API là gì?

Sử dụng công cụ Mock API để mô phỏng Backend

Trong quá trình phát triển phần mềm, việc phải gửi-nhận, giao tiếp với server là điều không thể tránh khỏi. Giả sử như khi chúng ta đang code một đoạn nhận và xử lý data từ server, mỗi lần muốn thay đổi response từ server để kiểm tra code, ta lại phải vào đổi code phía backend như vậy thì rất bất tiện và tốn thời gian. Việc sử dụng Mock API sẽ giúp chúng ta xử lí vấn đề này.

Tại sao sử dụng Mock API?

Lợi ích của Mock API là:

Các nhà phát triển frontend và các nhà phát triển phụ trợ có thể làm việc song song, do đó phát triển nhanh chóng.

Giao diện người dùng có thể hoạt động như một ứng dụng độc lập trong quá trình phát triển mà không có bất kỳ phụ thuộc API phụ trợ nào.

Dễ dàng demo

Mock API có thể dễ dàng được thay thế bằng API thực khi nó đã sẵn sàng.

Cấu hình Mock API

Thư viện npm sử dụng để mock API là connect-api-mocker

connect-api-mocker có thể được sử dụng với rất nhiều Node framework như Connect, Express và BrowserSync.

Trong bài này chúng ta sẽ sử dụng Express và connect-api-mocker

Cài đặt

npm i -D express connect-api-mocker

Sử dụng với Express

Trong thư mục gốc, bạn tạo 1 folder gọi là mock-api.

Trong thư mục mock-api, tạo fie app.js với đoạn code sau:

*const express = require("express");*

*const apiMocker = require("connect-api-mocker");*

*const port = 9000;*

*const app = express();*

*app.use("/api", apiMocker("mock-api"));*

*app.listen(port);*

Tạo Mock API

HTTP là gì ?

HTTP (Hypertext Transfer Protocol) là một giao thức nằm ở tầng ứng dụng (Application layer) của tập giao thức TCP/IP, để truyền nhận dữ liệu giữa các hệ thống phân tán thông qua internet

Quá trình làm việc của HTTP sẽ diễn ra như sau

1. HTTP client thiết lập một kết nối TCP đến server. Nếu thiết lập thành công, client và server sẽ truyền nhận dữ liệu với nhau thông qua kết nối này, kết nối được thiết lập còn gọi là socket interface bao gồm các thông tin: địa chỉ IP, loại giao thức giao vận (chính là TCP), và port (mặc định là 80).
2. Sau khi kết nối thành công, client gửi một HTTP request đến server thông qua socket interface vừa được thiết lập. Trong gói tin request sẽ chứa đường dẫn yêu cầu
3. Server sẽ nhận và xử lý request từ client thông qua socket, sau đó đóng gói dữ liệu tương ứng và gửi một HTTP response về cho client. Dữ liệu trả về sẽ là một file HTML chứa các loại dữ liệu khác nhau như văn bản, hình ảnh,…
4. Server đóng kết nối TCP.
5. Client nhận được dữ liệu phản hồi từ server và đóng kết nối TCP.

Axios là một thư viện HTTP Client dựa trên Promise. Cơ bản thì nó cung cấp một API cho việc xử lý XHR (XMLHttpRequests).

Nó được sử dụng để gửi các yêu cầu HTTP, đặc biệt là các yêu cầu AJAX, từ phía máy khách cũng như phía máy chủ.