ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

Task 3 - Tìm hiểu về backend web

Tp. Hồ Chí Minh, Tháng 10/2022

Mục lục

1 API

1.1 API là gì

API là viết tắt của "Application Programming Interface" (Giao diện Lập trình Ứng dụng) là một tập hợp các quy tắc và giao thức cho phép các phần mềm hoặc ứng dụng khác tương tác với nhau. API cho phép các phần mềm khác nhau trao đổi thông tin và chức năng một cách dễ dàng và đồng nhất.

API hoạt động bằng cách xác định các cách mà các phần mềm khác nhau có thể gọi và sử dụng các chức năng của một ứng dụng hoặc dịch vụ cụ thể. API thường bao gồm một tập hợp các hàm, quy tắc, và định dạng dữ liệu mà các phần mềm khác có thể sử dụng để truy cập và tương tác với một hệ thống hoặc dịch vụ cụ thể.

Ví dụ phổ biến về API bao gồm các API web, cho phép các ứng dụng web truy cập và tương tác với dữ liệu từ các dịch vụ web khác nhau như các dịch vụ mạng xã hội, dịch vụ thanh toán trực tuyến, hoặc dịch vụ đám mây lưu trữ. Các ứng dụng di động cũng thường sử dụng API để tương tác với các tính năng và dữ liệu trên các nền tảng khác nhau.

API đóng vai trò quan trọng trong việc kết nối các hệ thống và ứng dụng khác nhau để tạo ra các sản phẩm và dịch vụ phù hợp với nhu cầu của người dùng.

1.2 API hoạt động như thế nào

Kiến trúc API thường được giải thích dưới dạng máy chủ và máy khách. Ứng dụng gửi yêu cầu được gọi là máy khách, còn ứng dụng gửi phản hồi được gọi là máy chủ. Như vậy, trong ví dụ về thời tiết, cơ sở dữ liệu của cơ quan thời tiết là máy chủ còn ứng dụng di động là máy khách.

API REST là loại API phổ biến và linh hoạt nhất trên web hiện nay. Máy khách gửi yêu cầu đến máy chủ dưới dạng dữ liệu. Máy chủ dùng dữ liệu đầu vào từ máy khách này để bắt đầu các hàm nội bộ và trả lại dữ liệu đầu ra cho máy khách. Hãy cùng xem xét API REST chi tiết hơn ở bên dưới.

1.3 API REST

1.3.1 API REST là gì

REST là từ viết tắt của Chuyển trạng thái đại diện. REST xác định một tập hợp các hàm như GET, PUT, DELETE, v.v. mà máy khách có thể dùng để truy cập vào dữ liệu của máy chủ. Máy khách và máy chủ trao đổi dữ liệu qua giao thức HTTP.

Tính năng chính của API REST là tính không trạng thái. Tính không trạng trái nghĩa là máy chủ không lưu dữ liệu của máy khách giữa các yêu cầu. Các yêu cầu mà máy khách gửi cho máy chủ tương tự như URL mà bạn nhập vào trình duyệt

để truy cập vào trang web. Phản hồi từ máy chủ là dữ liệu thuần chứ không được kết xuất thành đồ họa như thường thấy trên trang web.

1.3.2 Lợi ích của API REST

API REST mang lại 4 lợi ích chính:

1. Tích hợp: API được sử dụng để tích hợp ứng dụng mới với hệ thống phần mềm hiện tại. Điều này làm tăng tốc độ phát triển vì không cần phải viết lại từng chức năng từ đầu. Bạn có thể sử dụng API để tận dụng mã hiện có.

2. Đổi mới: Rất nhiều lĩnh vực có thể thay đổi khi một ứng dụng mới ra mắt. Doanh nghiệp cần khẩn trương phản ứng và hỗ trợ việc triển khai nhanh chóng các dịch vụ đổi mới. Họ có thể thực hiện việc này bằng cách thực hiện các thay đổi ở cấp độ API mà không cần phải viết lại toàn bộ mã.

3. Mở rộng: API mang lại cơ hội độc đáo cho các doanh nghiệp để đáp ứng nhu cầu khách hàng của họ trên những nền tảng khác nhau. Ví dụ: API bản đồ cho phép tích hợp thông tin bản đồ qua các trang web, nền tảng Android, iOS, v.v. Mọi doanh nghiệp đều có thể cung cấp quyền truy cập tương tự vào cơ sở dữ liệu nội bộ của họ bằng API miễn phí hoặc trả phí.

4. Dễ duy trì: API đóng vai trò là cổng giữa hai hệ thống. Mỗi hệ thống đều phải thực hiện các thay đổi nội bộ để API không bị tác động. Bằng cách này, mọi sự thay đổi về mã trong tương lai do một bên thực hiện sẽ không tác động đến bên còn lại.

1.3.3 Bảo mật API REST

Mọi API đều phải được bảo mật bằng phương thức xác thực và giám sát đầy đủ. Có 2 cách chính để bảo mật cho API REST:

1. Token xác thực: Những token này được sử dụng để cho phép người dùng thực hiện lệnh gọi API. Token xác thực kiểm tra xem thông tin nhận dạng người dùng nhập có chính xác không và họ có quyền truy cập lệnh gọi API cụ thể đó không. Ví dụ: khi bạn đăng nhập vào máy chủ email, máy khách email của bạn sẽ dùng token xác thực để bảo mật hoạt động truy cập.

2. Khóa API: Khóa API xác thực chương trình hoặc ứng dụng thực hiện lệnh gọi API. Các khóa này nhận dạng ứng dụng và đảm bảo khóa có quyền truy cập cần thiết để thực hiện lệnh gọi API cụ thể. Khóa API không bảo mật như token nhưng chúng cho phép giám sát API để thu thập dữ liệu về việc sử dụng. Bạn có thể nhận thấy những chuỗi ký tự và chữ số dài trong URL trình duyệt khi bạn truy cập các trang web khác nhau. Chuỗi này là một khóa API mà trang web sử dụng để thực hiện lệnh gọi API nội bộ.

1.4 Điểm cuối API

Điểm cuối API là điểm tiếp xúc cuối cùng trong hệ thống giao tiếp của API. Những điểm cuối này bao gồm URL máy chủ, dịch vụ và những địa điểm kỹ thuật

số cụ thể khác, từ đây thông tin được gửi đi và tiếp nhận giữa các hệ thống. Điểm cuối API rất quan trọng đối với doanh nghiệp vì 2 lý do chính:

1. Bảo mật: Điểm cuối API khiến hệ thống dễ bị tấn công. Việc giám sát API để ngăn tình trạng lạm dụng là rất quan trọng.

2. Hiệu năng: Điểm cuối API, nhất là những điểm cuối có lưu lượng truy cập cao, có thể gây ra tình trạng nghẽn mạng và ảnh hưởng đến hiệu năng hệ thống.

1.5 Sử dụng một API

Để sử dụng một API, bạn cần thực hiện các bước cơ bản sau:

Chọn API phù hợp: Đầu tiên, bạn cần chọn API mà bạn muốn sử dụng. Điều này có thể là API cung cấp dịch vụ, dữ liệu hoặc chức năng cụ thể mà bạn cần để phát triển hoặc tích hợp vào ứng dụng của bạn.

Đăng ký và nhận khóa API (API Key): Đối với nhiều API, bạn sẽ cần đăng ký và nhận một khóa API hoặc thông tin xác thực khác để có quyền truy cập. Khóa API này giúp dịch vụ API xác định bạn là một người dùng hợp pháp và có quyền truy cập.

Xác định điểm cuối (API Endpoint): Tìm hiểu cách API được tổ chức và xác định điểm cuối cụ thể mà bạn muốn tương tác. Điều này thường bao gồm URL của điểm cuối và cách bạn nên gửi yêu cầu HTTP đến đó.

Tạo yêu cầu HTTP: Sử dụng ngôn ngữ lập trình hoặc công cụ tương tác với API, bạn sẽ tạo yêu cầu HTTP để gửi đến điểm cuối API. Điều này bao gồm việc xác định phương thức HTTP (GET, POST, PUT, DELETE), tham số yêu cầu (nếu cần), và dữ liệu yêu cầu (nếu áp dụng).

Gửi yêu cầu và xử lý phản hồi: Gửi yêu cầu HTTP đến điểm cuối API bằng cách sử dụng khóa API và thông tin xác thực (nếu cần). API sẽ xử lý yêu cầu của bạn và trả về phản hồi, thường là dữ liệu trong định dạng JSON hoặc XML. Bạn sau đó có thể xử lý phản hồi để sử dụng trong ứng dụng của bạn.

Xử lý lỗi và xử lý ngoại lệ (Exception Handling): Luôn luôn cần xử lý lỗi và xử lý ngoại lệ khi sử dụng API. API có thể trả về mã lỗi hoặc thông báo lỗi khi xảy ra vấn đề, và bạn cần kiểm tra và xử lý chúng một cách thích hợp trong mã của bạn.

Thực hiện tích hợp API vào ứng dụng của bạn: Sau khi bạn đã thành công trong việc gửi yêu cầu và xử lý phản hồi từ API, bạn có thể tích hợp dữ liệu hoặc chức năng từ API này vào ứng dụng của bạn, hiển thị thông tin cho người dùng hoặc sử dụng nó cho mục đích khác.

Lưu ý rằng mỗi API có thể có cách sử dụng và tài liệu hướng dẫn riêng, vì vậy luôn luôn đọc tài liệu cụ thể của API mà bạn đang sử dụng để biết cách tương tác chính xác.

2 Database

2.1 Định nghĩa

Cơ sở dữ liệu (Database) là một tập hợp có cấu trúc của dữ liệu được tổ chức và lưu trữ trong một hệ thống để có thể truy cập và quản lý dễ dàng. Cơ sở dữ liệu cho phép lưu trữ thông tin, dự liệu liên quan và thực hiện các thao tác truy vấn, cập nhật và xử lý dữ liệu một cách hiệu quả.

2.2 Các loại cơ sở dữ liệu phổ biến

Dưới đây là mô tả chi tiết hơn về từng loại cơ sở dữ liệu phổ biến:

- Cơ sở dữ liệu quan hệ (Relational Database):
  - Đặc điểm chính: Cơ sở dữ liệu quan hệ sử dụng bảng để tổ chức dữ liệu thành các hàng và cột, với các ràng buộc quan hệ giữa các bảng.
  - Ngôn ngữ truy vấn: Sử dụng SQL (Structured Query Language) để truy vấn và thao tác dữ liệu.
  - Ưu điểm:
    * Độ tin cậy cao với các giao dịch ACID (Atomicity, Consistency, Isolation, Durability).
    * Mô hình dữ liệu có cấu trúc giúp đảm bảo tính nhất quán. Hỗ trợ truy vấn phức tạp và biểu đồ quan hệ phức tạp.
  - Nhược điểm:
    * Khó mở rộng theo chiều ngang (scalability) so với một số hệ thống NoSQL.
    * Cần thời gian và công sức để thiết kế cơ sở dữ liệu quan hệ.
- Cơ sở dữ liệu không quan hệ (NoSQL Database):
  - Đặc điểm chính: NoSQL là một danh mục chung cho các hệ thống lưu trữ dữ liệu không tuân thủ mô hình cơ sở dữ liệu quan hệ truyền thống.
  - Loại NoSQL Database:
    * Cơ sở dữ liệu cột gia đình (Column-family): Lưu trữ dữ liệu trong các gia đình cột thay vì hàng, phù hợp cho việc đọc và ghi dữ liệu lớn với tốc độ cao. Ví dụ: Apache Cassandra, HBase.
    * Cơ sở dữ liệu tài liệu (Document): Lưu trữ dữ liệu trong các tài liệu JSON hoặc BSON, phù hợp cho dự án có nhu cầu dữ liệu linh hoạt. Ví dụ: MongoDB, CouchDB.
    * Cơ sở dữ liệu key-value: Lưu trữ dữ liệu dưới dạng cặp key-value, thích hợp cho các tác vụ đơn giản và hiệu suất cao. Ví dụ: Redis, Amazon DynamoDB.

  - Ưu điểm:
    * Khả năng mở rộng tốt, phù hợp cho các dự án có tải công việc cao hoặc dữ liệu linh hoạt.
    * Hỗ trợ hiệu suất cao cho đọc và ghi dữ liệu.
  - Nhược điểm:
    * Thiếu tính nhất quán và các giao dịch ACID có thể không được hỗ trợ.
    * Khó truy vấn dữ liệu phức tạp liên quan đến mối quan hệ giữa dữ liệu.

- Cơ sở dữ liệu đồ thị (Graph Database):
  - Đặc điểm chính: Cơ sở dữ liệu đồ thị làm việc với dữ liệu dưới dạng đỉnh (node) và mối quan hệ (edge) giữa các đỉnh. Nó được sử dụng để lưu trữ và truy vấn dữ liệu đồ thị.
  - Ưu điểm:
    * Hiệu quả cho các truy vấn liên quan đến mối quan hệ và phân tích đồ thị.
    * Thích hợp cho các ứng dụng xã hội, mạng lưới, phân tích mạng và khám phá dữ liệu.
  - Ví dụ DBMS: Neo4j, Amazon Neptune.

3 JSON

JSON là viết tắt của "JavaScript Object Notation" (Phiên bản ghi chép đối tượng JavaScript). Đây là một định dạng dữ liệu phổ biến được sử dụng để truyền tải và lưu trữ thông tin dưới dạng văn bản. JSON rất linh hoạt và được hỗ trợ trong nhiều ngôn ngữ lập trình khác nhau.

Dưới đây là một ví dụ đơn giản về JSON:

```
{
  "tên": "John Doe",
  "tuổi": 30,
  "địa chỉ": {
    "đường": "123 Main Street",
    "thành phố": "Anytown",
    "quận": "Somewhere"
  },
  "sở thích": ["đọc sách", "xem phim",
}
```

Trong ví dụ này, chúng ta có một đối tượng JSON mô tả thông tin về một người (tên, tuổi, địa chỉ, sở thích). JSON sử dụng cặp khóa-giá trị để biểu thị dữ liệu. Ví dụ này chỉ là một ví dụ đơn giản, và JSON có thể chứa thông tin phức tạp hơn và có thể được nhúng vào trong nhau để tạo thành cấu trúc dữ liệu phức tạp hơn.

4 Các sự kiện chính trong JavaScript

- Click (click): Sự kiện này xảy ra khi người dùng nhấp chuột lên một phần tử trên trang web.
- Double Click (dblclick): Sự kiện này xảy ra khi người dùng thực hiện một đợt nhấp đôi (double click) lên một phần tử.
- Mouseover (mouseover) và Mouseout (mouseout): Sự kiện mouseover xảy ra khi con trỏ chuột đi vào một phần tử, trong khi sự kiện mouseout xảy ra khi con trỏ chuột rời khỏi phần tử.
- Change (change): Sự kiện này thường được sử dụng trong các phần tử nhập liệu như ô text hoặc ô chọn (dropdown) và xảy ra khi giá trị của phần tử thay đổi.
- Submit (submit): Sự kiện này xảy ra khi người dùng gửi một biểu mẫu (form).
- Keydown (keydown), Keypress (keypress), và Keyup (keyup): Các sự kiện này xảy ra khi người dùng nhấn một phím trên bàn phím. Sự kiện keydown xảy ra khi phím được nhấn xuống, keypress xảy ra khi phím được giữ xuống và sự kiện keyup xảy ra khi phím được thả ra.
- Load (load): Sự kiện này xảy ra khi trang web hoàn thành tải.
- Unload (unload): Sự kiện này xảy ra khi trang web sắp được thoát.
- Resize (resize): Sự kiện này xảy ra khi cửa sổ trình duyệt thay đổi kích thước.
- Scroll (scroll): Sự kiện này xảy ra khi người dùng cuộn thanh trượt (scroll) của trang web.
- Focus (focus) và Blur (blur): Sự kiện focus xảy ra khi một phần tử nhận được sự tập trung, trong khi sự kiện blur xảy ra khi một phần tử mất sự tập trung.
- DOMContentLoaded: Sự kiện này xảy ra khi DOM của trang web đã được tải hoàn toàn.
- Error (error): Sự kiện này xảy ra khi có lỗi xảy ra trong quá trình tải trang hoặc tải tài nguyên (hình ảnh, tệp script, v.v.).
- Beforeunload (beforeunload): Sự kiện này xảy ra trước khi người dùng rời khỏi trang web và thường được sử dụng để xác nhận nếu họ thật sự muốn rời khỏi trang.
- Touch Events: Sự kiện này xảy ra khi người dùng sử dụng thiết bị cảm ứng (ví dụ: chạm, vuốt).

5 Tóm tắt đường đi của dữ liệu

Khi người dùng thao tác trên web. Một sự kiện sẽ xảy ra với tương tác đó. Dữ liệu sẽ được gửi đến database. Từ đó, các board mạch sẽ truy cập database lấy dữ liệu đó để xử lí, thao tác phía phần cứng.