Lesson 06
API Overview, Mock API, Axios

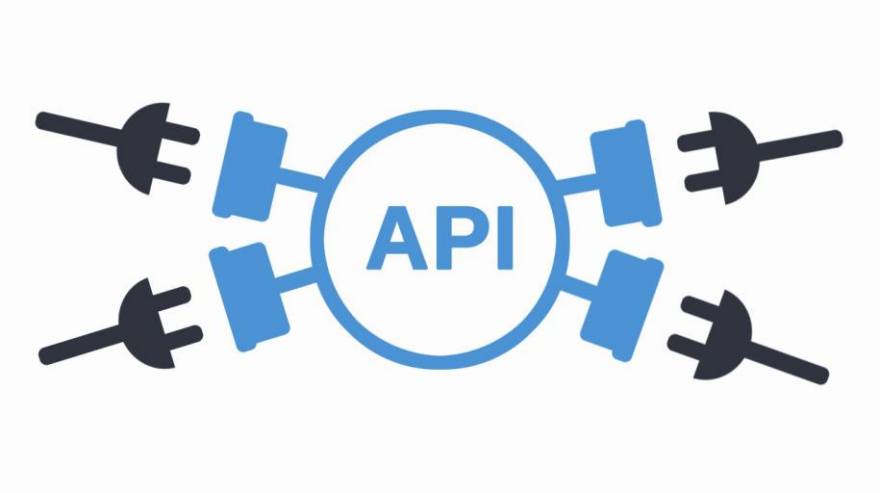
Module: ReactJS

Mục tiêu bài học

- Trình bày được API, Web API
- Trình bày được RESTful API
- Quá trình làm việc của HTTP
- Tạo Backend API với Mock API
- Gọi API từ Frontend với Axios

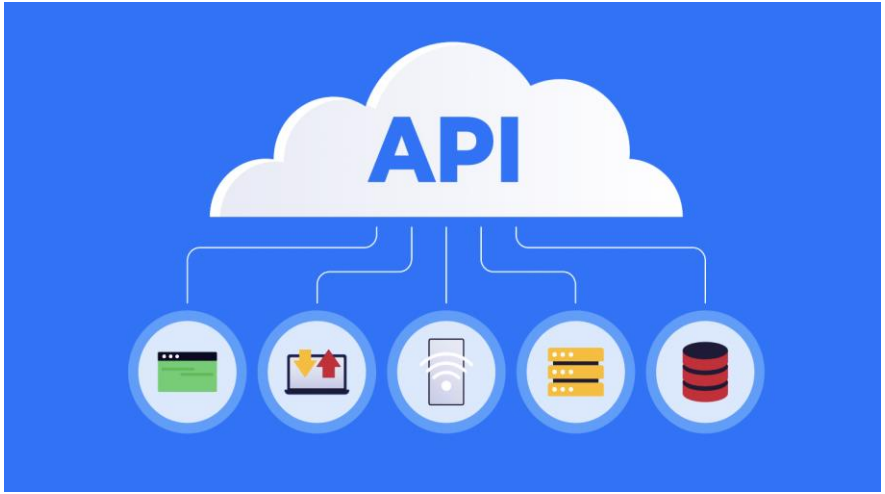
API, Web API

- **API** (Application Programming Interface): giao diện lập trình ứng dụng
- Đây là phương tiện cho hai hoặc nhiều ứng dụng trao đổi, tương tác với nhau, tạo ra tương tác giữa người dùng với ứng dụng hiệu quả và tiện lợi hơn.
- 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



API, Web API

- **API** bao gồm:
  - Web API: API của Web
  - OS API: API của hệ điều hành
  - API của Library/Framework



API, Web API

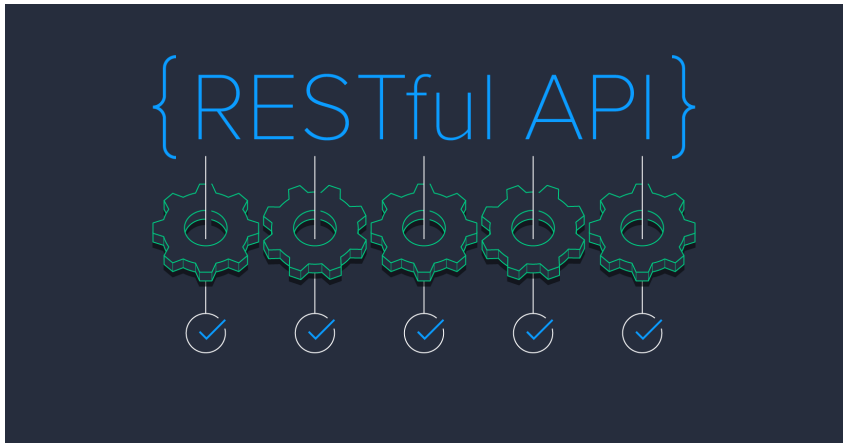
- **Tính năng của Web API**:
  - **Tự động hóa sản phẩm**: Đối với Web API, sẽ giúp người dùng có thể dễ dàng tự động quản lý được công việc.
  - **Tích hợp linh động**: API cho phép lấy nội dung ở bất kỳ Website hay ứng dụng nào đó một cách dễ dàng, khiến trải nghiệm người dùng được tăng lên.
  - **Cập nhật thông tin theo thời gian thực**: API giúp thay đổi và cập nhật những thông tin mới theo thời gian thực.

RESTful API

- **REST là gì?**
  - **REST (REpresentational State Transfer)** là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy.
  - Thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, **REST** gửi một yêu cầu HTTP như **GET, POST, PUT, DELETE**,.v.v.. đến một URL để xử lý dữ liệu

RESTful API

- **RESTful API là gì?**
  - **RESTful API** là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource.
  - **RESTful API** là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng khác nhau giao tiếp với nhau

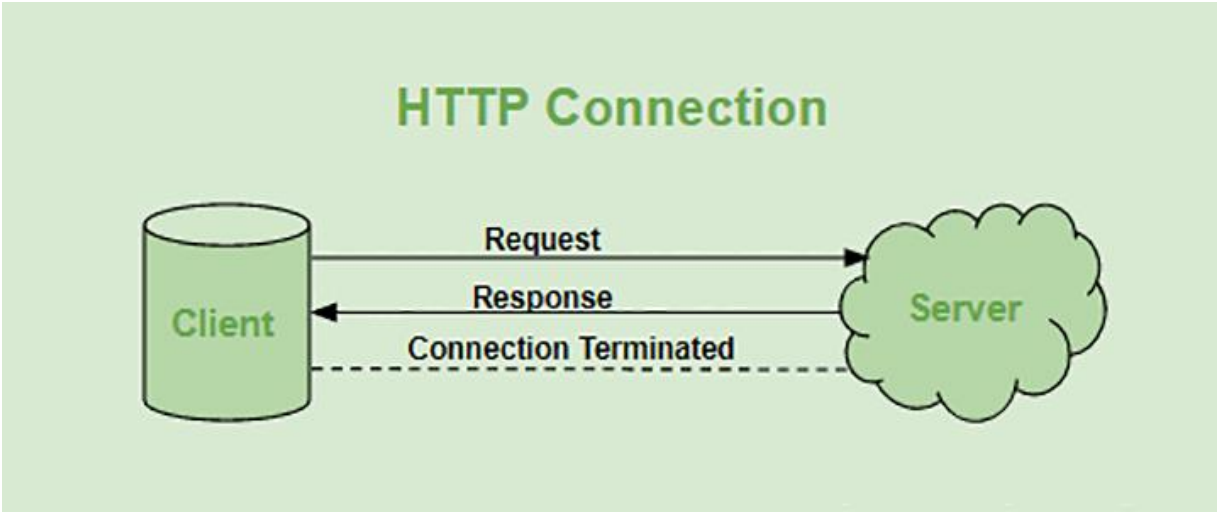


RESTful API

- **Cách thức 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)**: Xóa một Resource.
  - Những phương thức hay hoạt động này thường được gọi là **CRUD** tương ứng với các hành động **Create, Read, Update, Delete**

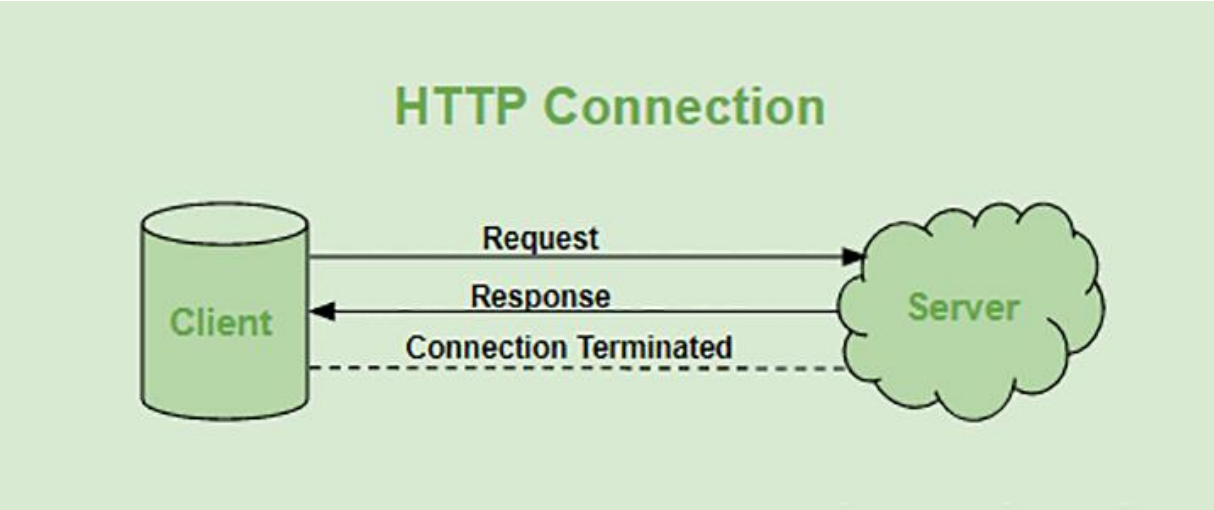
HTTP

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



HTTP

- **HTTP là gì?**
  - **HTTP client** thiết lập một kết nối **TCP** đến **server**. **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: địa chỉ **IP**, **TCP**, và **port** (80)

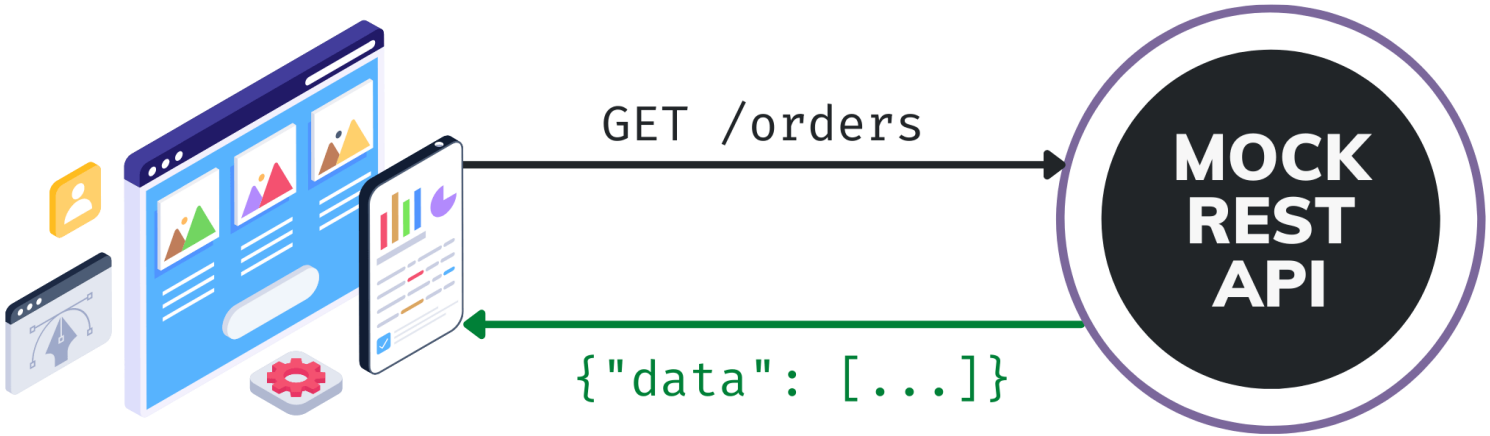


HTTP

- **HTTP là gì?**
  - **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,…
  - **Server** đóng kết nối **TCP**.
  - **Client** nhận được dữ liệu phản hồi từ **server** và đóng kết nối **TCP**

Mock API

- **Tại sao sử dụng Mock API?**
  - Các nhà phát triển **Frontend** và các nhà phát triển **Backend** có thể làm việc song song, do đó phát triển phần mềm trở nên nhanh chóng hơn.
  - Các nhà phát triển **Frontend** có thể bắt đầu với các **API mô phỏng** mà không cần chuyên sâu các kỹ năng của **Backend**.



Mock API

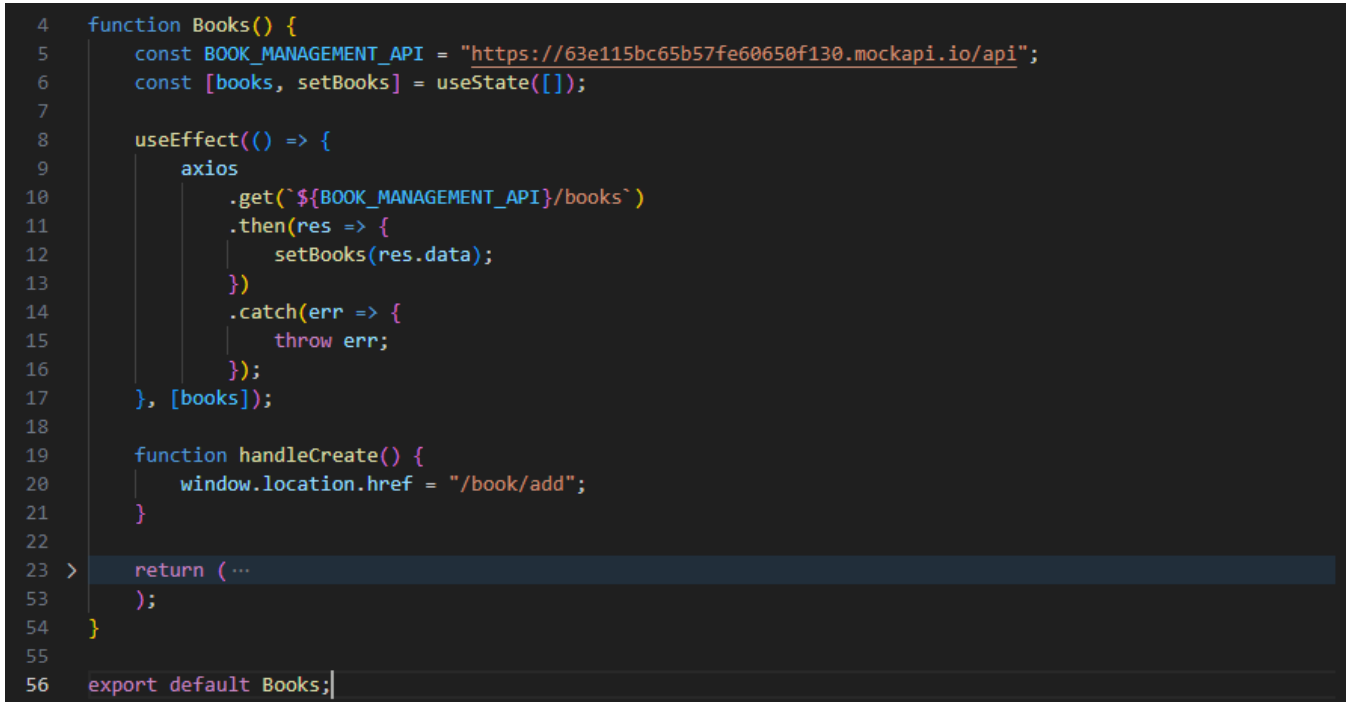
- **Cách tạo Mock API:**
  - Sử dụng **mockapi.io** để tạo sẵn các bộ **API** phía **Backend:**

Thư viện Axios

- **Axios** là một thư viện **HTTP Client** dựa trên **Promise**. Cơ bản thì nó cung cấp một **API** cho việc xử lý **XHR** (XMLHttpRequests)**:**
  - Tạo **XMLHttpRequests** từ trình duyệt
  - Thực hiện các http **request** từ **node.js**
  - Hỗ trợ **Promise API**
  - Chặn **request** và **response**
  - Chuyển đổi dữ liệu **request**, **response**
  - Hủy **requests**
  - Tự động chuyển đổi về dữ liệu **JSON**
  - Hỗ trợ phía client để chống lại **CSRF**

Sử dụng Axios

- Viết hàm sử dụng axios gọi **API** lấy danh sách các đầu sách từ **MockAPI:**



```
4   function Books() {
5       const BOOK_MANAGEMENT_API = "https://63e115bc65b57fe60650f130.mockapi.io/api";
6       const [books, setBooks] = useState([]);
7
8       useEffect(() => {
9           axios
10              .get(`${BOOK_MANAGEMENT_API}/books`)
11              .then(res => {
12                  setBooks(res.data);
13              })
14              .catch(err => {
15                  throw err;
16              });
17      }, [books]);
18
19      function handleCreate() {
20          window.location.href = "/book/add";
21      }
22
23  >   return (···
53      );
54  }
55
56  export default Books;
```

Sử dụng Axios

- Viết hàm sử dụng axios gọi **API** lấy thông tin chi tiết đầu sách từ **MockAPI:**

```
const BOOK_MANAGEMENT_API = "https://63e115bc65b57fe60650f130.mockapi.io/api";
const { bookId } = useParams();
const [book, setBook] = useState([]);

useEffect(() => {
  if (bookId) {
    axios
      .get(`${BOOK_MANAGEMENT_API}/books/${bookId}`)
      .then(res => {
          setBook(res.data);
      })
      .catch(err => {
        throw err;
      });
  }
}, [bookId]);
```

Sử dụng Axios

- Viết hàm sử dụng axios gọi **API** thêm mới một đầu sách vào **MockAPI:**

```
function handleSubmit() {
  axios
    .post(`${BOOK_MANAGEMENT_API}/books`, book)
    .then(res => {
      alert(
        `${isCreate ? "Create" : "Edit"} book ${JSON.stringify(
          res.data
        )} successfully!!!`
      );
      window.location.href = "/";
    })
    .catch(err => {
      throw err;
    });
}
```

Sử dụng Axios

- Viết hàm sử dụng axios gọi **API** sửa thông tin một đầu sách từ **MockAPI:**

```
31  function handleSubmit() {
32    axios
33      .put(`${BOOK_MANAGEMENT_API}/books/${bookId}`, book)
34      .then(res => {
35        alert(
36          `${isCreate ? "Create" : "Edit"} book ${JSON.stringify(
37            res.data
38          )} successfully!!!`
39        );
40        window.location.href = "/";
41      })
42      .catch(err => {
43        throw err;
44      });
45  }
```

Sử dụng Axios

- Viết hàm sử dụng axios gọi **API** sửa thông tin một đầu sách từ **MockAPI:**

```
function removeBook() {
  if (bookId) {
    axios
    .delete(`${BOOK_MANAGEMENT_API}/books/${bookId}`)
    .then(res => {
      alert(
        `Remove book ${JSON.stringify(
          res.data
        )} successfully!!!`
      );
      window.location.href = "/";
    })
    .catch(err => {
      throw err;
    });
  }
}
```

Tóm tắt bài học

- Trình bày được API, Web API
- Trình bày được RESTful API
- Quá trình làm việc của HTTP
- Tạo Backend API với Mock API
- Gọi API từ Frontend với Axios