

# RESTFUL API TRONG REACT

Giang viên: Thuận Nguyễn

Email: [nguyenthuankma@gmail.com](mailto:nguyenthuankma@gmail.com)

Phone: 0369.821.997





# TASK 01

**RESTFUL API TỔNG QUAN**

# RESTful API là gì?

**RESTful API là một kiểu kiến trúc dịch vụ web dựa trên giao thức HTTP. Nó sử dụng các phương thức HTTP như GET, POST, PUT, DELETE để thao tác với tài nguyên.**

- **GET:** Lấy dữ liệu từ máy chủ.
- **POST:** Gửi dữ liệu mới đến máy chủ.
- **PUT:** Cập nhật dữ liệu hiện có trên máy chủ.
- **DELETE:** Xóa dữ liệu từ máy chủ.



# TASK 02

**RESTFUL API THỰC HÀNH**

# 1. Lấy dữ liệu từ API với request get ?

```
useEffect(() => {
  const fetchUsers = async () => {
    try {
      const response = await axios.get('https://jsonplaceholder.typicode.com/users');
      setUsers(response.data);
      setLoading(false);
    } catch (error) {
      console.error('Error fetching users:', error);
      setLoading(false);
    }
  };
  fetchUsers();
}, []);
```

Lấy dữ liệu từ API

Lấy dữ liệu từ Database thông qua API

## 2. Gửi dữ liệu với POST request ?

```
const [name, setName] = useState('');
const [email, setEmail] = useState('');

const handleSubmit = async (e) => {
  e.preventDefault();

  try {
    const response = await axios.post(
      'https://jsonplaceholder.typicode.com/users', API URL
      |
      {
        name,
        email Dữ liệu gửi lên
      }
    );

    console.log('User added:', response.data);
  } catch (error) {
    console.error('Error adding user:', error);
  }
};
```

Gửi dữ liệu mới từ form lên Database thông qua API

### 3. Cập nhật dữ liệu với PUT request ?

```
const [name, setName] = useState('');
const [email, setEmail] = useState('');

const handleUpdate = async (e) => {
  e.preventDefault();
  try {
    const response = await axios.put( Cập nhật dữ liệu qua PUT request
      `https://jsonplaceholder.typicode.com/users/${userId}`, API URL
      {
        name,
        email Dữ liệu mới gửi lên
      }
    );
    console.log('User updated:', response.data);
  } catch (error) {
    console.error('Error updating user:', error);
  }
};
```

chỉnh sửa dữ liệu mới từ form lên Database thông qua API

## 4. Xóa dữ liệu với DELETE request ?

```
// src/components/DeleteUser.js
import React from 'react';
import axios from 'axios';

const DeleteUser = ({ userId }) => {
  const handleDelete = async () => {
    try {
      const response = await axios.delete(`https://jsonplaceholder.typicode.com/users/${userId}`);
      console.log('User deleted:', response.data);
    } catch (error) {
      console.error('Error deleting user:', error);
    }
  };
  return (
    <button onClick={handleDelete}>Delete User</button>
  );
};

export default DeleteUser;
```

**Xóa dữ liệu**

**API của user cần xóa**

Xóa dữ liệu thông qua API với DELETE request



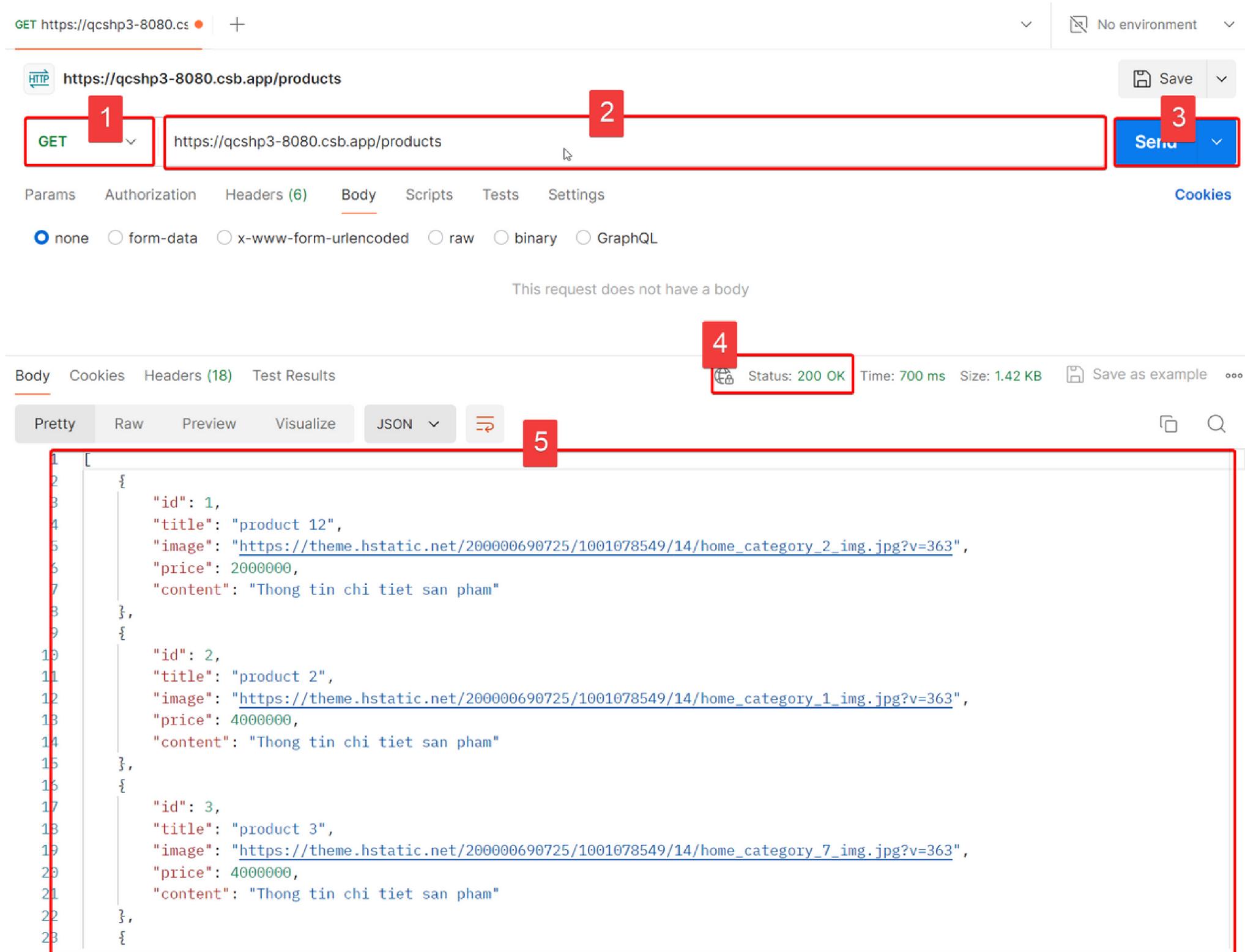
# TASK 03

**TEST API VỚI POSTMAN**

# 1. Postman là gì ?

- Postman là một **App Extensions**, cho phép làm việc với các API, nhất là REST, giúp ích rất nhiều cho việc testing.
- Hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, DELETE, OPTIONS, HEAD ...)
- Postman cho phép lưu lại các lần sử dụng. Sử dụng cho cá nhân hoặc team lớn.

## 2. Làm việc với request GET ?



1. Thiết lập request HTTP của bạn là GET

2. Trong trường URL yêu cầu, nhập vào link

3. Kích nút Send

4. message là 200 lấy dữ liệu thành công

5. Sẽ hiển thị kết quả sản phẩm trong phần Body của bạn.

### 3. Làm việc với request POST ?



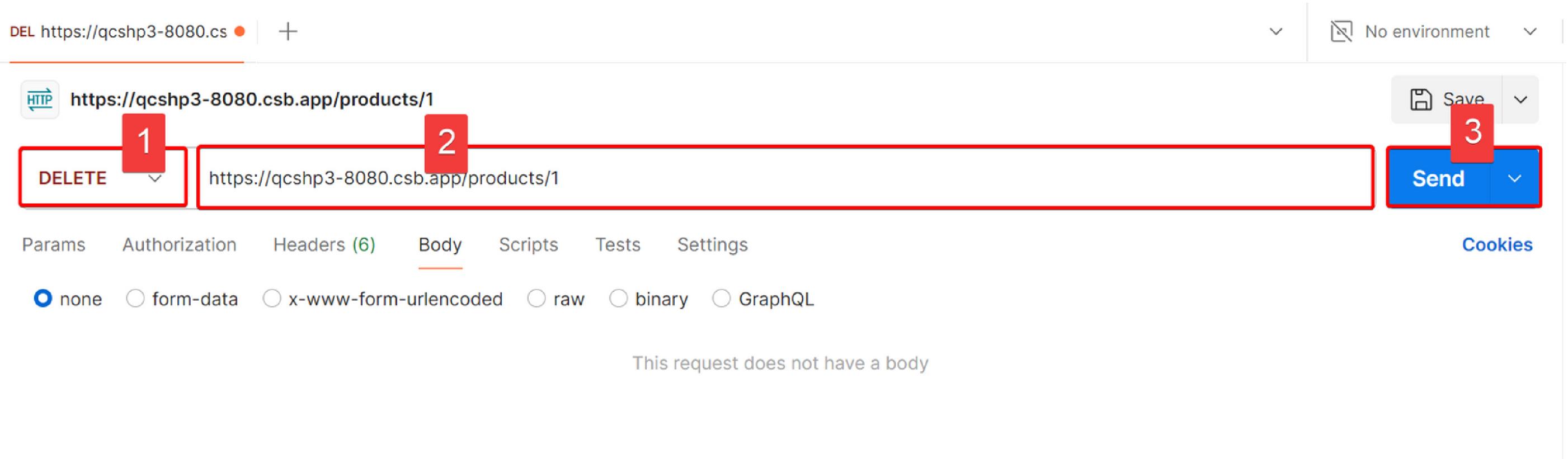
Gửi dữ liệu mới từ form lên Database thông qua API

## 4. Làm việc với request PUT ?



chỉnh sửa dữ liệu mới từ form lên Database thông qua API

## 4. Làm việc với request DELETE ?



Xóa dữ liệu thông qua API với DELETE request



# TASK 04

**CÁC THƯ VIỆN QUẢN LÝ  
SERVER STATE - CALL API**

# Các thư viện quản lý state từ server (server state) và gọi API thường dùng ?

- 01** **React Query:** React Query là một thư viện mạnh mẽ giúp quản lý state từ server, gọi API và caching dữ liệu.
- 02** **Redux Toolkit Query:** Redux Toolkit Query là một phần của Redux Toolkit, giúp quản lý API request và caching dữ liệu.
- 03** **SWR:** SWR là một thư viện React hook để lấy dữ liệu từ API và caching dữ liệu, được phát triển bởi Vercel.
- 04** **Apollo Client:** Apollo Client là một thư viện mạnh mẽ để quản lý state từ server, đặc biệt là khi bạn sử dụng GraphQL API.



# TASK 05

## SỬ DỤNG REACT-QUERY

# 1. Tổng quan về React-Query ?

- React Query (hiện tại là @tanstack/react-query) là một thư viện mạnh mẽ giúp quản lý state từ server và gọi API trong các ứng dụng React
- Nó cung cấp các công cụ để fetching, caching, đồng bộ hóa và cập nhật dữ liệu máy chủ một cách dễ dàng.
- Cài đặt @tanstack/react-query trong dự án của mình:

```
npm install @tanstack/react-query
```

# Thực hành React-Query cho dự án thực tế ?

*Truy cập link sau:*

<https://tanstack.com/query/latest/docs/framework/react/overview>