LAB 3: ĐỒNG BỘ VÀ BẤT ĐỒNG BỘ

Bài 1: Tạo form nhập thông tin

	Form Đăng Ký	
Tên:		
An		
Email:		
anh@itc.e	du.vn	
	Gửi	
	Gửi thành công!	

Yêu cầu:

- Tạo một form với các trường: Tên, Email.
- Khi người dùng nhấn nút "Gửi", hiển thị thông báo "Đang xử lý..." và sau 2 giây hiển thị thông báo "Gửi thành công!" hoặc "Gửi thất bại!" dựa trên kết quả của Promise.

Hướng dẫn

Filie.js

File .js

- Lắng nghe sự kiện **submit** của form.
- Khi form được submit, ngăn không cho form gửi dữ liêu đi theo cách mặc định.
- Hiển thị thông báo "Đang xử lý...".
- Gọi hàm simulateServerRequest để giả lập việc gửi dữ liệu lên server.
- Hàm simulateServerRequest trả về một Promise. Sau 2 giây, Promise sẽ được giải quyết ngẫu nhiên với kết quả thành công hoặc thất bại.
- Dựa trên kết quả của Promise, hiển thị thông báo tương ứng cho người dùng.

Bài 2: Quản lý người dùng

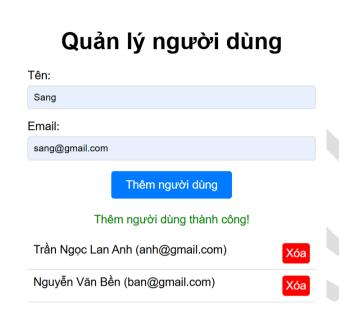
Tạo một ứng dụng quản lý người dùng với các tính năng:

- 1. Thêm người dùng mới.
- 2. Lấy danh sách người dùng từ server.
- 3. Xóa người dùng.

Yêu cầu

- 1. Tạo một form với các trường: Tên, Email và một nút "Thêm người dùng".
- 2. Khi người dùng nhấn nút "Thêm người dùng", gửi thông tin lên server giả lập và cập nhật danh sách người dùng.

- 3. Hiển thị danh sách người dùng từ server.
- 4. Cho phép xóa người dùng từ danh sách.



Hướng dẫn:

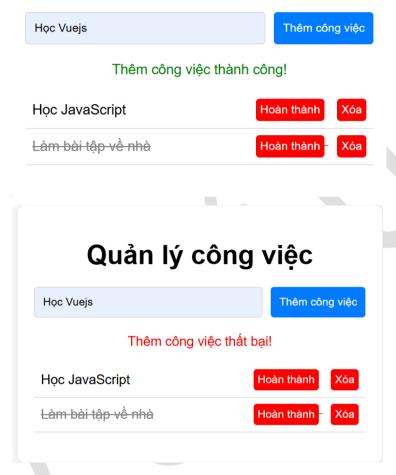
File .html

File .js

- Sử dụng sự kiện **DOMContentLoaded** để đảm bảo mã JavaScript chỉ chạy sau khi toàn bộ nội dung của trang đã được tải.
- Lắng nghe sự kiện submit của form để thêm người dùng mới.
- Sử dụng Promise để giả lập việc thêm người dùng, lấy danh sách người dùng, và xóa người dùng từ server.
- Cập nhật danh sách người dùng sau mỗi thao tác thêm hoặc xóa.

Bài 3: Quản lý công việc

Quản lý công việc



Đánh dấu hoàn thành



Xóa công việc



Tạo một ứng dụng quản lý công việc (To-Do List) với các tính năng:

- 1. Thêm công việc mới.
- 2. Lấy danh sách công việc từ server.
- 3. Đánh dấu công việc là đã hoàn thành.
- 4. Xóa công việc.

Yên cầu

- 1. Tạo một form với trường nhập công việc và một nút "Thêm công việc".
- 2. Khi người dùng nhấn nút "Thêm công việc", gửi thông tin lên server giả lập và cập nhật danh sách công việc.
- 3. Hiển thị danh sách công việc từ server.
- 4. Cho phép đánh dấu công việc là đã hoàn thành và xóa công việc.

Hướng dẫn:

File .html

File .js

- Sử dụng sự kiện DOMContentLoaded để đảm bảo mã JavaScript chỉ chạy sau khi toàn bộ nội dung của trang đã được tải.
- Lắng nghe sự kiện submit của form để thêm công việc mới.

- Sử dụng Promise để giả lập việc thêm công việc, lấy danh sách công việc, đánh dấu hoàn thành công việc, và xóa công việc từ server.
- Cập nhật danh sách công việc sau mỗi thao tác thêm, hoàn thành hoặc xóa.

Bài 4: Hệ thống quản lý người dùng

Từ bài tập 2, tạo một ứng dụng quản lý người dùng với các tính năng:

- 1. Thêm người dùng mới.
- 2. Lấy danh sách người dùng từ server giả lập.
- 3. Xóa người dùng.

Yêu cầu

- 1. Tạo một form với trường nhập tên người dùng và một nút "Thêm người dùng".
- 2. Khi người dùng nhấn nút "Thêm người dùng", gửi thông tin lên server giả lập và cập nhật danh sách người dùng.
- 3. Hiển thị danh sách người dùng từ server.
- 4. Cho phép xóa người dùng.
- 5. Sử dụng async và await thực hiện bài tập trên

Hướng dẫn:

File .js

- Sử dụng async và await để làm việc với các tác vụ bất đồng bộ một cách dễ dàng và trực quan.
- Sử dụng sự kiện DOMContentLoaded để đảm bảo mã JavaScript chỉ chạy sau khi toàn bộ nội dung của trang đã được tải.
- Lắng nghe sự kiện submit của form để thêm người dùng mới.
- Sử dụng các hàm async để giả lập việc thêm người dùng, lấy danh sách người dùng, và xóa người dùng từ server.
- Cập nhật danh sách người dùng sau mỗi thao tác thêm hoặc xóa.

Bài 5: Lấy thông tin thời tiết từ API

Weather App



Ho Chi Minh City

Temperature: 27.26°C

Humidity: 83%

Weather: few clouds

Yên cầu:

- Tạo một form để nhập tên thành phố.
- Gửi yêu cầu tới API thời tiết và nhận dữ liệu thời tiết của thành phố.
- Hiển thị thông tin thời tiết trên giao diện.

Lưu ý: nên nhập tên tiếng việt không dấu

Hướng dẫn

- B1: Vào trang https://home.openweathermap.org/ để đăng ký tài khoản
- B2: Sau khi đăng ký xong vào mail xác nhận
- B3: Vào menu API lấy key



File .html

File .js

- Sử dụng async và await để làm việc với API bất đồng bộ.
- Lắng nghe sự kiện submit của form để lấy tên thành phố và gọi hàm getWeather để lấy dữ liệu thời tiết.
- Hàm getWeather gửi yêu cầu tới API OpenWeatherMap và trả về dữ liệu thời tiết.

Hàm displayWeather hiển thị thông tin thời tiết trên giao diện.

Bài 6: Quản lý sản phẩm

Yêu cầu:

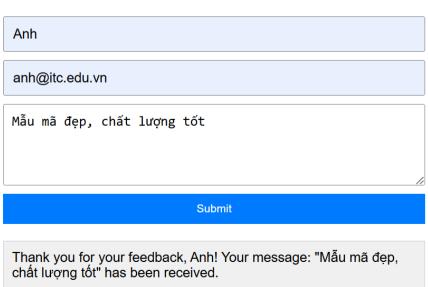
 Người dùng có thể thêm sản phẩm vào giỏ hàng, và khi nhấn vào nút "Mua hàng", thông tin giỏ hàng sẽ được gửi đến máy chủ giả lập (mock server).

Chú ý

 Đây là ví dụ sử dụng jsonplaceholder.typicode.com như một máy chủ giả lập (mock server) để thực hiện yêu cầu POST. Trong thực tế, bạn cần thay thế URL bằng API endpoint thực tế

Bài 7: Tạo Form phản hồi người dùng

Người Dùng Feedback



Yêu cầu:

- Tạo một form cho phép người dùng gửi phản hồi.
- Sử dụng fetch API để gửi dữ liệu đến server giả lập và hiển thị thông báo phản hồi từ server.

Hướng dẫn:

File .html

File .js

- **async function**: Hàm xử lý sự kiện submit được định nghĩa là async để có thể sử dụng await bên trong nó.
- await fetch: Sử dụng await để đợi kết quả của lệnh fetch. Điều này giúp mã nguồn dễ đọc và xử lý lỗi dễ dàng hơn.
- **response.ok**: Kiểm tra xem phản hồi từ server có thành công không. Nếu không, ném ra lỗi.
- await response.json(): Chờ cho đến khi dữ liệu JSON được parse xong trước khi tiếp tục.

Bài 8: Ứng dụng quản lý tiền tệ

Currency Converter



Yêu cầu:

- **Mô tả**: Xây dựng ứng dụng theo dõi và chuyển đổi tỷ giá tiền tệ giữa các loại tiền tệ khác nhau.
- **API sử dụng**: Sử dụng API từ các dịch vụ như ExchangeRate-API hoặc Open Exchange Rates để lấy tỷ giá tiền tệ và thực hiện chuyển đổi.

Hướng dẫn:

File .html

```
<body>
<div class="container">
   <h1>Currency Converter</h1>
    <input type="number" id="amount" placeholder="Amount" required>
    <select id="fromCurrency">
       <option value="USD">USD</option>
       <option value="EUR">EUR</option>
       <option value="GBP">GBP</option>
       <option value="VND">VND</option>
       <!-- Add more currencies as needed -->
    </select>
    <select id="toCurrency">
       <option value="USD">USD</option>
       <option value="EUR">EUR</option>
       <option value="GBP">GBP</option>
       <option value="VND">VND</option>
       <!-- Add more currencies as needed -->
    </select>
   <button onclick="convertCurrency()">Convert</button>
    <div id="result" class="result"></div>
</div>
<script src="currency.js"></script>
```