1. CSS phù hợp cho trang describeContainer

<div class="container">

<img src="image.jpg" alt="Ảnh">

<div class="info">

<h2>Tiêu đề</h2>

<p>Thông tin mô tả.</p>

</div>

</div>

.container {

display: flex;

}

.container img {

max-width: 50%; /\* hoặc bất kỳ chiều rộng nào bạn muốn \*/

}

.info {

max-width: 50%; /\* hoặc tùy chỉnh chiều rộng \*/

}

/\* Media query cho màn hình nhỏ \*/

@media (max-width: 768px) {

.container {

flex-direction: column; /\* Đặt các phần tử theo cột \*/

}

.container img {

max-width: 100%; /\* Ảnh chiếm toàn bộ chiều rộng \*/

height: auto; /\* Đảm bảo tỉ lệ ảnh đúng \*/

}

.info {

max-width: 100%; /\* Thông tin chiếm toàn bộ chiều rộng \*/

}

}

1. Thanh search khi gõ hiện lên thông tin sách

**html**

<nav>

<input type="text" id="search" placeholder="Tìm kiếm sách...">

<div id="suggestions" class="suggestions"></div>

</nav>

**css**  
.suggestions {

border: 1px solid #ccc;

max-height: 200px;

overflow-y: auto;

display: none; /\* Ẩn ban đầu \*/

}

.suggestion-item {

padding: 10px;

cursor: pointer;

}

.suggestion-item:hover {

background-color: #f0f0f0;

}

**Js**

const searchInput = document.getElementById('search');

const suggestionsDiv = document.getElementById('suggestions');

// Hàm debounce

function debounce(func, delay) {

let timeoutId;

return function(...args) {

clearTimeout(timeoutId);

timeoutId = setTimeout(() => {

func.apply(this, args);

}, delay);

};

}

// Hàm gọi API

async function fetchBooks(query) {

try {

const response = await fetch(`https://api.example.com/books?search=${query}`);

const data = await response.json();

displaySuggestions(data);

} catch (error) {

console.error('Error fetching books:', error);

}

}

// Hàm hiển thị gợi ý

function displaySuggestions(books) {

suggestionsDiv.innerHTML = ''; // Xóa các gợi ý cũ

if (books.length > 0) {

suggestionsDiv.style.display = 'block'; // Hiện danh sách gợi ý

books.forEach(book => {

const suggestionItem = document.createElement('div');

suggestionItem.className = 'suggestion-item';

suggestionItem.textContent = book.title;

suggestionItem.onclick = () => {

window.location.href = `book-details.html?id=${book.id}`;

};

suggestionsDiv.appendChild(suggestionItem);

});

} else {

suggestionsDiv.style.display = 'none'; // Ẩn nếu không có gợi ý

}

}

// Hàm tìm kiếm có debounce

const debouncedFetchBooks = debounce(fetchBooks, 500);

searchInput.addEventListener('input', function() {

const query = this.value.trim();

if (query) {

debouncedFetchBooks(query); // Gọi hàm tìm kiếm đã debounce

} else {

suggestionsDiv.style.display = 'none'; // Ẩn nếu ô tìm kiếm trống

}

});

1. Chế độ trình chiếu

<html lang="vi">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>PDF in HTML</title>

</head>

<body>

<iframe src="https://media.geeksforgeeks.org/wp-content/cdn-uploads/20210101201653/PDF.pdf"

width="800" height="500" **allowfullscreen**></iframe>

</body>

</html>

1. Dropdown khi ấn vào

import React, { useState } from 'react';

import Dropdown from 'react-bootstrap/Dropdown';

import 'bootstrap/dist/css/bootstrap.min.css';

import './App.css'; // Nhớ thêm file CSS của bạn

const categories = [

'tl1', 'tl2', 'tl3', 'tl4', 'tl5', 'tl6',

'tl7', 'tl8', 'tl9', 'tl10', 'tl11', 'tl12',

'tl13', 'tl14', 'tl15'

];

const CategoryDropdown = () => {

return (

<Dropdown>

<Dropdown.Toggle variant="success" id="dropdown-basic">

Chọn thể loại

</Dropdown.Toggle>

<Dropdown.Menu className= “'seven-columns”>

{categories.map((category, index) => (

<Dropdown.Item key={index} className="category-item">

{category}

</Dropdown.Item>

))}

</Dropdown.Menu>

</Dropdown>

);

};

export default CategoryDropdown;

**css**

.category-item {

padding: 5px;

}

.seven-columns {

display: grid;

grid-template-columns: repeat(7, 1fr);

}

1. Phân trang

import React, { useState, useEffect } from 'react';

const BookList = () => {

const [books, setBooks] = useState([]);

const [currentPage, setCurrentPage] = useState(1);

const booksPerPage = 10;

useEffect(() => {

const fetchBooks = async () => {

const response = await fetch('API\_URL'); // Thay thế 'API\_URL' bằng URL thực tế

const data = await response.json();

setBooks(data);

};

fetchBooks();

}, []);

// Tính toán sách cần hiển thị dựa trên trang hiện tại

const indexOfLastBook = currentPage \* booksPerPage;

const indexOfFirstBook = indexOfLastBook - booksPerPage;

const currentBooks = books.slice(indexOfFirstBook, indexOfLastBook);

// Chuyển trang

const totalPages = Math.ceil(books.length / booksPerPage);

return (

<div>

<h1>Danh sách sách</h1>

<ul>

{currentBooks.map((book) => (

<li key={book.id}>

{book.name} - Lượt xem: {book.views}

</li>

))}

</ul>

<div>

<button

onClick={() => setCurrentPage(currentPage - 1)}

disabled={currentPage === 1}

>

Trang trước

</button>

<span> Trang {currentPage} / {totalPages} </span>

<button

onClick={() => setCurrentPage(currentPage + 1)}

disabled={currentPage === totalPages}

>

Trang sau

</button>

</div>

</div>

);

};

export default BookList;

1. Sort

import React, { useState, useEffect } from 'react';

const BookList = () => {

const [books, setBooks] = useState([]);

const [sortOrder, setSortOrder] = useState('name'); // 'name' hoặc 'views'

useEffect(() => {

const fetchBooks = async () => {

const response = await fetch('API\_URL'); // Thay thế 'API\_URL' bằng URL thực tế

const data = await response.json();

setBooks(data);

};

fetchBooks();

}, []);

const handleSortChange = (event) => {

setSortOrder(event.target.value);

};

const sortedBooks = [...books].sort((a, b) => {

if (sortOrder === 'name') {

return a.name.localeCompare(b.name);

} else if (sortOrder === 'views') {

return b.views - a.views; // Sắp xếp theo lượt xem giảm dần

}

return 0;

});

return (

<div>

<h1>Danh sách sách</h1>

<select onChange={handleSortChange} value={sortOrder}>

<option value="name">Sắp xếp theo tên</option>

<option value="views">Sắp xếp theo lượt xem giảm dần</option>

</select>

<ul>

{sortedBooks.map((book) => (

<li key={book.id}>

{book.name} - Lượt xem: {book.views}

</li>

))}

</ul>

</div>

);

};

export default BookList;

1. Tăng lượt xem
2. Lượt tải sách về