

LAB 5. REACT: COMPONENTS, PROPS, RENDERING & STYLING CONTENT

Thời lượng: 4 tiết

A. Mục tiêu

Bài thực hành này được thực hiện sau khi sinh viên hoàn tất các yêu cầu trong bài Lab

4. Sau khi hoàn thành bài thực hành này, sinh viên cần nắm:

- Làm quen các câu lệnh tạo và chạy một trang web bằng React, cài đặt các gói (packages).
- Tạo các thành phần (components) có trong website Tips & Tricks Blog và sử dụng các tham số (props) trong một thành phần.
- Sử dụng React Router để tùy biến định tuyến (route) đến các thành phần khác nhau.
- Làm quen với các thành phần có trong React Bootstrap và Font Awesome.
- Tạo các trang cho người đọc và tùy chỉnh giao diện website.
- Hiển thị dữ liệu từ API.

Yêu cầu: Sinh viên tự làm phần “B. Hướng dẫn thực hành” ở nhà và nộp lên hệ thống LMS. Tại phòng Lab, sinh viên làm phần “C. Bài tập thực hành” dựa trên dự án đã hoàn thành ở phần B.

B. Hướng dẫn thực hành

1. Tạo dự án React

React là một thư viện hoặc framework JavaScript cho phép xây dựng các giao diện người dùng. Ứng dụng mà React tạo có dạng là SPA (single-page application) cho phép chúng ta tạo các thành phần có thể tái sử dụng. Thư viện React được tạo bởi Facebook. Thay vì thao tác trực tiếp đến DOM (Document Object Model) của trình duyệt, React tạo một DOM ảo trong bộ nhớ - nơi mà tất cả thao tác cần thiết được thực hiện trước khi tạo các thay đổi trong DOM của trình duyệt.

React chỉ thay đổi những gì cần được thay đổi, điều đó có nghĩa là không cần phải tải lại toàn bộ các thành phần có trên trang web khi nội dung của một thành phần bị thay đổi.

1.1. Cài đặt Node.js và tạo dự án React

Truy cập vào website <https://nodejs.org/en/> tải về phiên bản LTS và cài đặt:

Node.js® is an open-source, cross-platform JavaScript runtime environment.

Download for Windows (x64)

18.15.0 LTS

Recommended For Most Users

19.7.0 Current

Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

For information about supported releases, see the [release schedule](#).

Hình 1

Tại thư mục gốc của dự án, mở terminal, gõ lệnh sau để tạo một ứng dụng React:

```
npx create-react-app tat-blog
```

Chờ cho quá trình tạo hoàn tất:

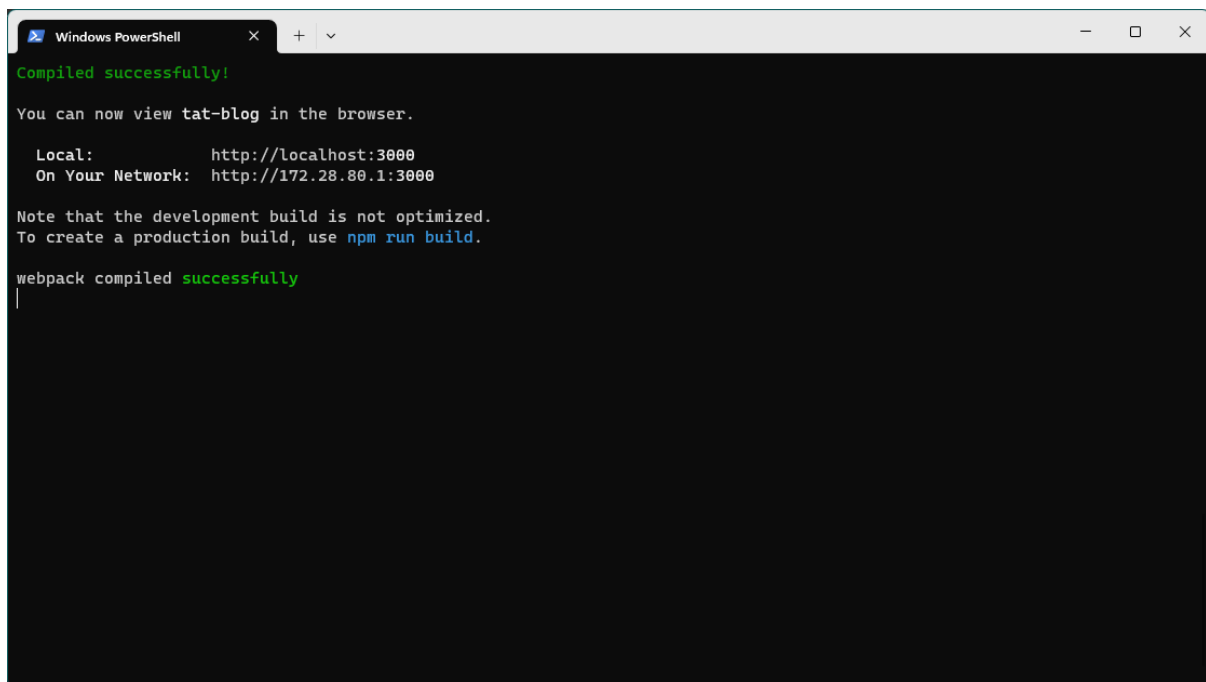
```
PS D:\GitRepos> npx create-react-app tat-blog
Creating a new React app in D:\GitRepos\tat-blog.
Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...
[Progress bar] | idealTree:domexception: sill fetch manifest browser-process-hrtime@^1.0.0
```

Hình 2

Sau khi đã tạo ứng dụng React có tên là tat-blog, thay đổi đường dẫn hiện tại của terminal vào thư mục tat-blog và chạy ứng dụng:

```
cd tat-blog
npm start
```

Webpack sẽ build ứng dụng và chạy ở đường dẫn: <http://localhost:3000>:



```
Windows PowerShell
Compiled successfully!

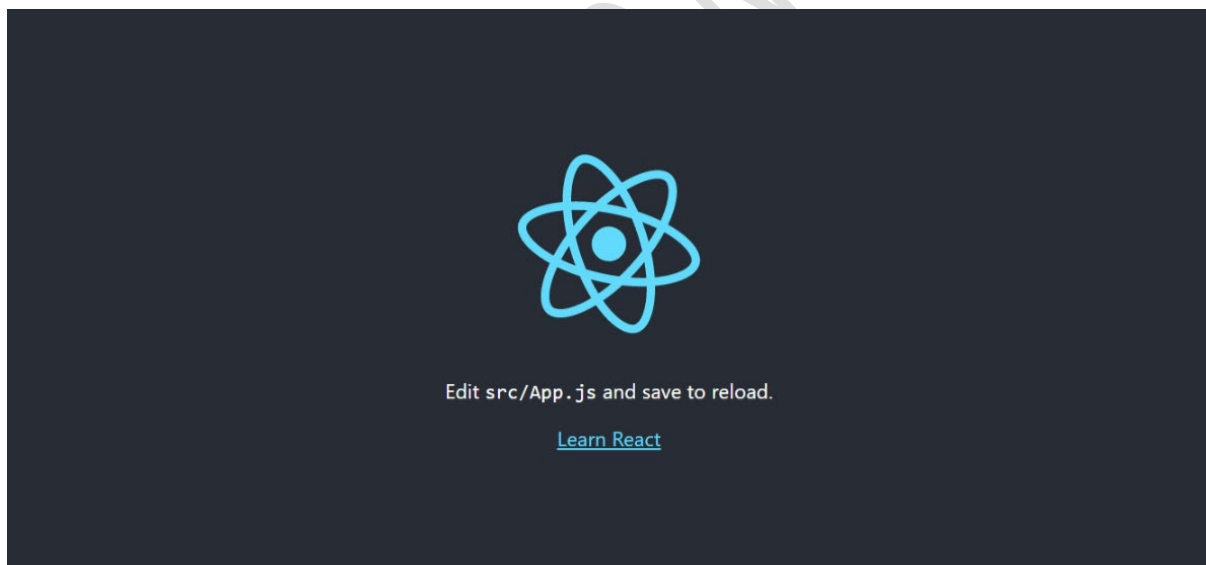
You can now view tat-blog in the browser.

Local:      http://localhost:3000
On Your Network:  http://172.28.80.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

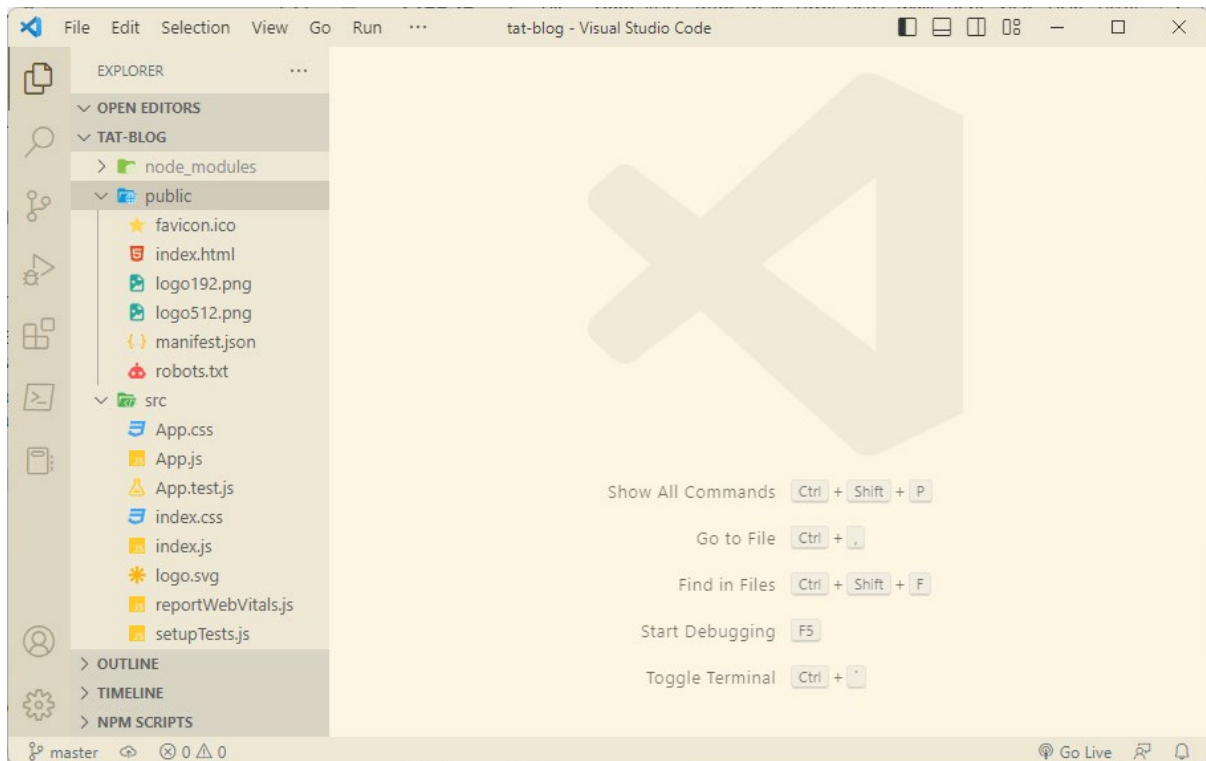
Hình 3



Hình 4

Để dừng, nhấn tổ hợp phím **Ctrl + C**, nhấn **y** và **Enter**.

Từ phần này trở về sau, trình soạn thảo mã nguồn được sử dụng cho lập trình ứng dụng web React này là Visual Studio Code. Hãy mở thư mục dự án bằng Visual Studio Code:



Hình 5

Một dự án React có cấu trúc thông thường như sau:

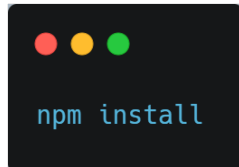
- Thư mục src/ chứa các tập tin mã nguồn chính của dự án:
 - index.js là tập tin script điểm vào, thực thi đầu tiên. Hàm `ReactDOM.render()` hiển thị mã HTML được chỉ định bên trong phần tử có id là root.

```
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Hình 6

- app.js là tập tin thành phần (component) chính của ứng dụng
- *.css được nhập bởi các tập tin JavaScript
- Thư mục public/ chứa các tập tin tĩnh:
 - Hình ảnh
 - index.html

- package.json và package-lock.json quản lý các phụ thuộc (dependencies) của dự án
- Thư mục node_modules/ chứa các gói (packages) của bên thứ 3 đã được liệt kê ở tập tin package.json. Thư mục này được tạo lại bằng cách chạy lệnh:



Lệnh trên sẽ cài đặt tất cả các gói có trong package.json.

1.2. Cài đặt các gói React Bootstrap, React Router và Font Awesome

Tại terminal đã ở đường dẫn của dự án, gõ lệnh sau:



Trong đó:

- react-bootstrap thay thế Bootstrap JavaScript với mỗi thành phần của Bootstrap được xây dựng lại như một thành phần của React mà không cần phụ thuộc vào jQuery.
- react-router-dom cho phép chúng ta thiết lập các định tuyến đến các thành phần React khác nhau.
- fontawesome-svg-core là gói lõi bao gồm tất cả mọi thứ để các biểu tượng hoạt động.
- free-solid-svg-icons và free-regular-svg-icons là hai gói biểu tượng miễn phí.
- react-fontawesome là thành phần Font Awesome React.

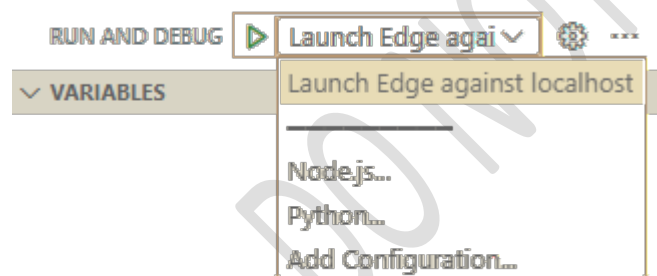
1.3. Debug trong React

Lưu ý: Phần này hướng dẫn debug một website React trong trình duyệt Edge, chi tiết hướng dẫn vui lòng xem thêm tại https://code.visualstudio.com/docs/nodejs/reactjs-tutorial#_debugging-react. Mặc định cổng mà React sử dụng là 3000.

Ở thư mục gốc của dự án, tạo thư mục `.vscode` và bên trong là tập tin `launch.json`:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "type": "msedge",
      "request": "launch",
      "name": "Launch Edge against localhost",
      "url": "http://localhost:3000",
      "webRoot": "${workspaceFolder}"
    }
  ]
}
```

Tiến hành chạy server như bình thường bằng lệnh `npm start`. Sau đó nhấn F5 hoặc vào công cụ **Run and Debug (Ctrl + Shift + D)** nhấn vào mũi tên màu xanh nếu như **Launch Edge against localhost** được chọn:



Hình 7

Một cửa sổ trình duyệt Edge được mở lên và trong Visual Studio Code xuất hiện thanh

Debug: . Bây giờ chúng ta có thể đặt các breakpoints vào vị trí mong muốn và debug.

2. Tạo Header, Footer và Sidebar

Thành phần (Components) là những đoạn code độc lập và tái sử dụng. Chúng được sử dụng giống như là các hàm JavaScript nhưng làm việc độc lập và trả về HTML.

Thành phần bao gồm hai loại: Class và Function. Bài lab này tập trung vào thành phần Function.

Thành phần lớp (Class component)

Thành phần hàm (Function component)

```
class Car extends React.Component {    function Car() {
  render() {                          return <h2>Hi, I am a Car!</h2>;
    return <h2>Hi, I am a          }
    Car!</h2>;
  }
}
```

Lưu ý: Tên của các thành phần hoặc các tập tin NÊN được bắt đầu bởi một ký tự hoa. Các thành phần cũng NÊN được tách thành các tập tin riêng.

2.1. Tạo Navbar

Trong thư mục src, tạo mới thư mục Components và các tập tin trong thư mục Components như sau:

- Navbar.js
- Sidebar.js
- Footer.js

Đây là các thành phần quan trọng tạo nên bố cục của hoàn chỉnh của một trang web thông thường. Trong src/Components/Navbar.js thêm nội dung như sau:

```
import React from 'react';
import { Navbar as Nb, Nav } from 'react-bootstrap';
import {
  Link
} from 'react-router-dom';

const Navbar = () => {
  return (
    <Nb collapseOnSelect expand='sm' bg='white' variant='light'
    className='border-bottom shadow'>
      <div className='container-fluid'>
        <Nb.Brand href='/'>Tips & Tricks</Nb.Brand>
        <Nb.Toggle aria-controls='responsive-navbar-nav' />
        <Nb.Collapse id='responsive-navbar-nav' className='d-sm-inline-flex
        justify-content-between'>
          <Nav className='mr-auto flex-grow-1'>
            <Nav.Item>
              <Link to='/' className='nav-link text-dark'>
                Trang chủ
              </Link>
            </Nav.Item>
            <Nav.Item>
              <Link to='/blog/about' className='nav-link text-dark'>
                Giới thiệu
              </Link>
            </Nav.Item>
          </Nav>
        </Nb.Collapse>
      </div>
    </Nb>
  );
}
```

```
        <Nav.Item>
          <Link to='/blog/contact' className='nav-link text-dark'>
            Liên hệ
          </Link>
        </Nav.Item>
        <Nav.Item>
          <Link to='/blog/rss' className='nav-link text-dark'>
            RSS Feed
          </Link>
        </Nav.Item>
      </Nav>
    </Nb.Collapse>
  </div>
</Nb>
)
}
```

```
export default Navbar;
```

Dòng `export default Navbar;` cho phép sử dụng Navbar trong tập tin JavaScript khác.

Trong tập tin `src/index.js`, bổ sung dòng sau:

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

Dòng trên sẽ thêm các CSS có trong tập tin `bootstrap.min.css` vào trình duyệt.

Lưu ý: Sinh viên có thể tự tạo các tập tin .css và áp dụng cho các thành phần cụ thể bằng cách sử dụng `import`

Trong tập tin `src/App.js` thay thế bởi nội dung như sau:

```
import './App.css';
import Navbar from './Components/Navbar';
import {
  BrowserRouter as Router,
  Routes,
  Route,
} from 'react-router-dom';

function App() {
  return (
    <div>
      <Router>
        <Navbar />
      </Router>
    </div>
  );
}

export default App;
```


Kết quả:

[Tips & Tricks](#) [Trang chủ](#) [Giới thiệu](#) [Liên hệ](#) [RSS Feed](#)

Hình 8

2.2. Tạo Sidebar

Trong src/Components/Sidebar.js thêm nội dung như sau:

```
import React from 'react';

const Sidebar = () => {
  return (
    <div className='pt-4 ps-2'>
      <h1>
        Tìm kiếm bài viết
      </h1>
      <h1>
        Các chủ đề
      </h1>
      <h1>
        Bài viết nổi bật
      </h1>
      <h1>
        Đăng ký nhận tin mới
      </h1>
      <h1>
        Tag cloud
      </h1>
    </div>
  )
}
```

```
export default Sidebar;
```

Gọi Sidebar trong App.js:

```
...
import Sidebar from './Components/Sidebar';
...
return (
  <div>
    <Router>
      <Navbar />
      <div className='container-fluid'>
        <div className='row'>
          <div className='col-9'>

            </div>
            <div className='col-3 border-start'>
              <Sidebar />
            </div>
          </div>
        </div>
      </Router>
    </div>
  );
```

2.3. Tạo Footer

Trong src/Components/Footer.js thêm nội dung như sau:

```
import React from 'react';

const Footer = () => {
  return (
    <footer className='border-top footer text-muted'>
      <div className='container-fluid text-center'>
        &copy; 2023 - Tips & Tricks Blog
      </div>
    </footer>
  )
}
```

`export default Footer;`

Gọi Footer trong App.js:

```
...
import Footer from './Components/Footer';
...

<Router>
  <Navbar />
  <div className='container-fluid'>
    <div className='row'>
      <div className='col-9'>

        </div>
```

```
<div className='col-3 border-start'>
  <Sidebar />
</div>
</div>
</div>
<Footer />
```

...

Kết quả:



Hình 9

3. Tạo trang Giới thiệu, Liên hệ, RSS Feed

Trong thư mục src, tạo mới thư mục Pages và các tập tin trong thư mục Pages như sau:

- Index.js
- About.js
- Contact.js
- Rss.js

Thư mục này sẽ lưu những trang có trong web. Trong tập tin src/Pages/Index.js, thêm nội dung như sau:

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

const Index = () => {
  useEffect(() => {
    document.title = 'Trang chủ';
  }, []);

  return (
    <h1>
```

```
    Đây là trang chủ  
  </h1>  
);  
}
```

```
export default Index;
```

`useEffect` là một hook cho phép thực hiện các tác dụng phụ (side effects) trong thành phần, đó có thể là: lấy dữ liệu, thay đổi DOM và bộ hẹn giờ,... `useEffect` có dạng như sau: `useEffect(<function>, <dependency>)` trong đó đối số thứ hai là tùy chọn. Sử dụng mảng trống để hàm `useEffect` kiểm soát chỉ chạy trong kết xuất đầu tiên.

Lưu ý: Nếu ứng dụng React trong quá trình phát triển thì kể cả với mảng trống, `useEffect` sẽ chạy thêm 1 lần nữa¹.

```
useEffect(() => {  
  }, [prop, state]);
```

Ở đoạn mã trên, hàm `useEffect` sẽ chạy vào lần đầu kết xuất và bất kỳ thời điểm nào mà các giá trị phụ thuộc (`prop`, `state`) thay đổi.

Thực hiện tương tự trong các tập tin `About.js`, `Contact.js` và `RSS.js`.

Trong `src/App.js`, thêm các dòng sau:

```
...  
import Index from './Pages/Index';  
...  
<div className='col-9'>  
  <Index />  
</div>  
...
```

Kết quả:

¹ Nguồn: <https://react.dev/reference/react/useEffect#examples-dependencies>,
<https://react.dev/learn/synchronizing-with-effects#how-to-handle-the-effect-firing-twice-in-development>

[Tips & Tricks](#) [Trang chủ](#) [Giới thiệu](#) [Liên hệ](#) [RSS Feed](#)**Đây là trang chủ****Tìm kiếm bài viết****Các chủ đề****Bài viết nổi bật****Đăng ký nhận tin mới****Tag cloud**

© 2023 - Tips & Tricks Blog

Hình 10

Bố cục chính của trang web sẽ bao gồm Navbar, Sidebar và MainContent. Tạo tập tin mới src/Pages/Layout.js với nội dung như sau:

```
import { Outlet } from 'react-router-dom';

const Layout = () => {
  return (
    <>
      <Outlet />
    </>
  );
};
```

```
export default Layout;
```

Chỉnh sửa tập tin src/App.js như sau:

```
import './App.css';
import Navbar from './Components/Navbar';
import Sidebar from './Components/Sidebar';
import Footer from './Components/Footer';
import Layout from './Pages/Layout';
import Index from './Pages/Index';
import About from './Pages/About';
import Contact from './Pages/Contact';
import RSS from './Pages/RSS';
import {
  BrowserRouter as Router,
  Routes,
  Route,
} from 'react-router-dom';
```

```
function App() {  
  return (  
    <div>  
      <Router>  
        <Navbar />  
        <div className='container-fluid'>  
          <div className='row'>  
            <div className='col-9'>  
              <Routes>  
                <Route path='/' element={<Layout />}>  
                  <Route path='/' element={<Index />} />  
                  <Route path='blog' element={<Index />} />  
                  <Route path='blog/Contact' element={<Contact />} />  
                  <Route path='blog/About' element={<About />} />  
                  <Route path='blog/RSS' element={<RSS />} />  
                </Route>  
              </Routes>  
            </div>  
            <div className='col-3 border-start'>  
              <Sidebar />  
            </div>  
          </div>  
        <Footer />  
      </Router>  
    </div>  
  );  
}
```

```
export default App;
```

Kết quả khi truy cập vào một menu nào đó thì nội dung của trang web thay đổi theo:

Tips & Tricks Trang chủ Giới thiệu Liên hệ RSS Feed

Đây là trang giới thiệu

Tìm kiếm bài
viết

Các chủ đề

Bài viết nổi bật

Đăng ký nhận
tin mới

Tag cloud

© 2023 - Tips & Tricks Blog

Hình 11

[Tips & Tricks](#) [Trang chủ](#) [Giới thiệu](#) [Liên hệ](#) [RSS Feed](#)

Đây là trang RSS Feed

[Tìm kiếm bài viết](#)[Các chủ đề](#)[Bài viết nổi bật](#)[Đăng ký nhận tin mới](#)[Tag cloud](#)

© 2023 - Tips & Tricks Blog

Hình 12

4. Tạo TagList và PostItem

4.1. Tạo TagList

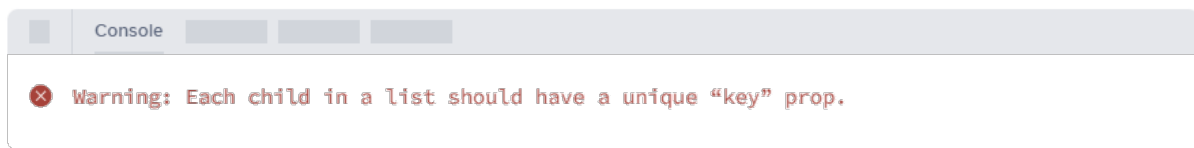
TagList dùng để hiển thị danh sách các thẻ (tags) của một bài viết (post). Bây giờ tạo tập tin `src/Components/TagList.js` với nội dung sau:

```
import { Link } from 'react-router-dom';

const TagList = ({ tagList }) => {
  if (tagList && Array.isArray(tagList) && tagList.length > 0)
    return (
      <>
        {tagList.map((item, index) => {
          return (
            <Link to={`/blog/tag?slug=${item.name}`}
              title={item.name}
              className='btn btn-sm btn-outline-secondary me-1'
              key={index}>
              {item.name}
            </Link>
          );
        })}
      </>
    );
  else
    return (
      <></>
    );
};

export default TagList;
```

Lưu ý: Nếu không thêm thuộc tính key vào thẻ, React sẽ hiển thị cảnh báo sau ở console:



Hình 13

4.2. Tạo PostItem

PostItem là thành phần hiển thị thông tin cơ bản của một bài viết để hiển thị trên trang chủ của trình duyệt. Nó bao gồm hình ảnh, tên, tác giả, chủ đề, mô tả, danh sách thẻ (tag list) và nút **Xem chi tiết**.

Trong thư mục public, tạo một thư mục có tên là images dùng để lưu trữ các hình ảnh tĩnh trên web server. Tìm hình ảnh trên mạng và lưu vào đường dẫn public/images với tên tập tin là **image_1.jpg**. Đây là hình ảnh mặc định cho PostItem nếu như thuộc tính imageUrl không được chỉ định giá trị.

Lưu ý: Hãy kiểm tra một vài hình ảnh lưu vào thư mục public/images.

Tạo một thư mục và tập tin mới src/Utils/Utils.js:

```
export function isEmptyOrSpaces(str) {  
  return str === null || (typeof str === 'string' && str.match(/^\s*$/) !==  
  null);  
}
```

Tạo tập tin src/Components/PostItem.js với nội dung sau:

```
import TagList from './TagList';  
import Card from 'react-bootstrap/Card';  
import { Link } from 'react-router-dom';  
import { isEmptyOrSpaces } from '../Utils/Utils'  
  
const PostList = ({ postItem }) => {  
  
  let imageUrl = isEmptyOrSpaces(postItem.imageUrl)  
    ? process.env.PUBLIC_URL + '/images/image_1.jpg'  
    : `${postItem.imageUrl}`;  
  
  let postedDate = new Date(postItem.postedDate);  
  
  return (  
    <article className='blog-entry mb-4'>  
      <Card>  
        <div className='row g-0'>  
          <div className='col-md-4'>
```



```

        <Card.Img variant='top' src={imageUrl} alt={postItem.title} />
      </div>
      <div className='col-md-8'>
        <Card.Body>
          <Card.Title>{postItem.title}</Card.Title>
          <Card.Text>
            <small className='text-muted'>Tác giả:</small>
            <span className='text-primary m-1'>
              {postItem.author.fullName}
            </span>
            <small className='text-muted'>Chủ đề:</small>
            <span className='text-primary m-1'>
              {postItem.category.name}
            </span>
          </Card.Text>
          <Card.Text>
            {postItem.shortDescription}
          </Card.Text>
          <div className='tag-list'>
            <TagList tagList={postItem.tags} />
          </div>
          <div className='text-end'>
            <Link
              to={` /blog/post?year=${postedDate.getFullYear()}&month=${postedDate.getMonth(
                )}&day=${postedDate.getDay()}&slug=${postItem.urlSlug}`}
              className='btn btn-primary'
              title={postItem.title}>
              Xem chi tiết
            </Link>
          </div>
        </Card.Body>
      </div>
    </div>
  </Card>
</article >
);
};

```

export default PostList;

Để hiển thị PostItem, trong tập tin src/Pages/Index.js:

```

import React, { useEffect, useState } from 'react';
import PostItem from '../Components/PostItem';

const Index = () => {
  const [postList, setPostList] = useState([]);

  useEffect(() => {
    document.title = 'Trang chủ';

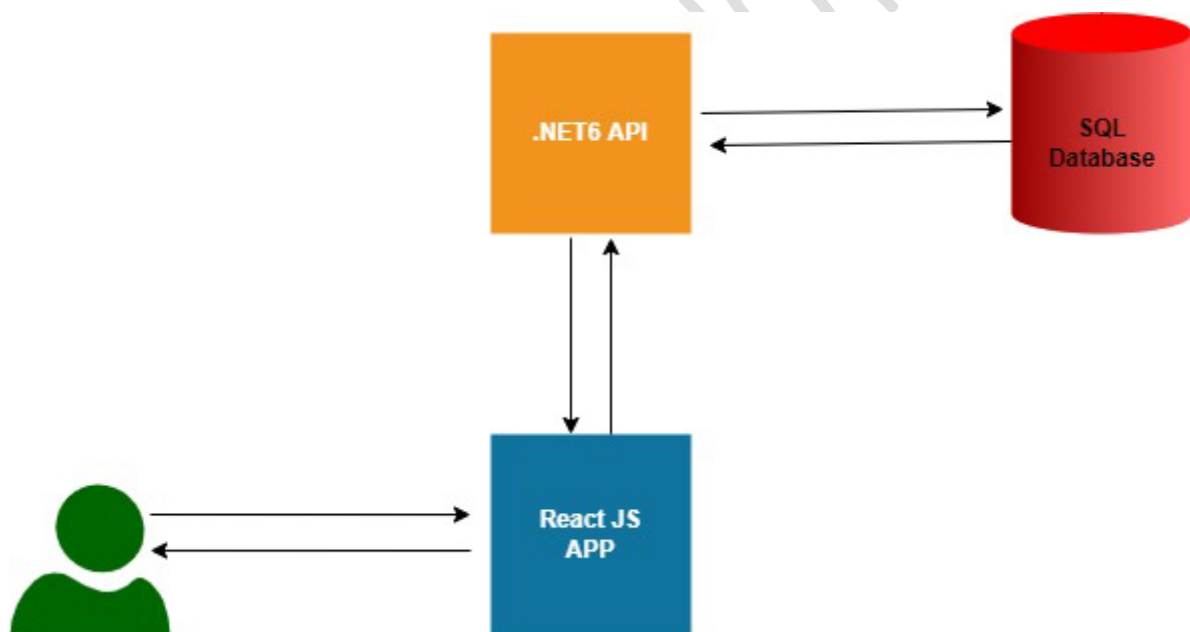
```

```
    }, []);

    if (postList.length > 0)
      return (
        <div className='p-4'>
          {postList.map(item => {
            return (
              <PostItem postItem={item} />
            );
          })}
        </div>
      );
    else return (
      <></>
    );
  }

  export default Index;
```

5. Hiển thị dữ liệu từ API



Hình 14

Tóm tắt quá trình giao tiếp:

- Người dùng truy cập vào website, một yêu cầu gửi đến máy chủ web React để các tập tin JavaScript tải về và chạy trên trình duyệt.
- Vì React là một SPA, do đó phụ thuộc vào API để hiển thị dữ liệu.
- API chạy ở máy chủ, lấy dữ liệu từ cơ sở dữ liệu và gửi JSON về React.

Axios là một thư viện HTTP Client dựa trên Promise dành cho node.js và trình duyệt.

Cài đặt axios bằng lệnh sau:

```
npm install axios
```

Tạo thư mục src/Services và tập tin BlogRepository.js trong thư mục Services với nội dung như sau:

```
import axios from 'axios';

export async function getPosts(keyword = '', pageSize = 10, pageNumber = 1,
sortColumn = '', sortOrder = '') {
  try {
    const response = await
    axios.get(`https://localhost:7085/api/posts?keyword=${keyword}&PageSize=${pa
geSize}&PageNumber=${pageNumber}&SortColumn=${sortColumn}&SortOrder=${sortOr
der}`);
    const data = response.data;
    if (data.isSuccess)
      return data.result;
    else
      return null;
  } catch (error) {
    console.log('Error', error.message);
    return null;
  }
}
```

Lưu ý: Tập tin BlogRepository.js trên chứa lời gọi API của các thành phần và trang.

Gọi phương thức trên trong tập tin src/Pages/Index.js để tự động lấy dữ liệu và hiển thị:

```
import React, { useEffect, useState } from 'react';
import PostItem from '../Components/PostItem';
import { getPosts } from '../Services/BlogRepository';

const Index = () => {
  const [postList, setPostList] = useState([]);

  useEffect(() => {
    document.title = 'Trang chủ';

    getPosts().then(data => {
      if (data)
        setPostList(data.items);
      else
        setPostList([]);
    })
  }, []);
```

```

if (postList.length > 0)
  return (
    <div className='p-4'>
      {postList.map((item, index) => {
        return (
          <PostItem postItem={item} key={index} />
        );
      })}
    </div>
  );
else return (
  <></>
);
}

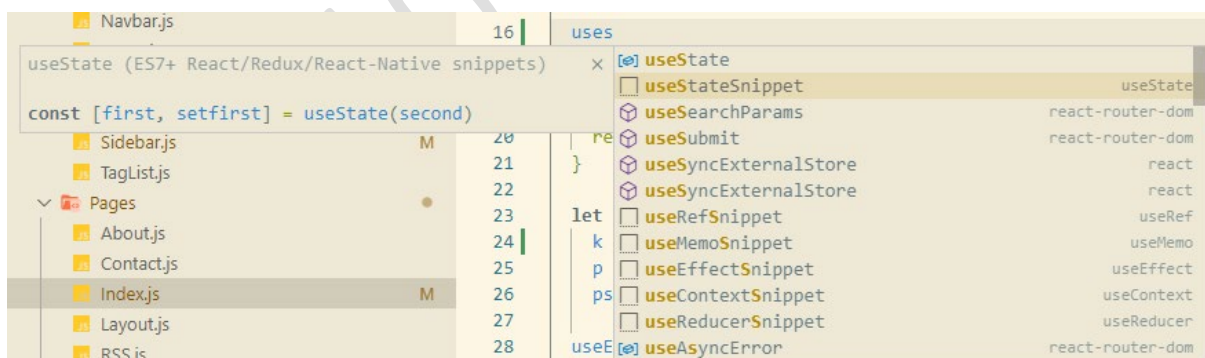
```

`export default Index;`

Ở đoạn mã trên sử dụng một hook có tên là `useState`. `useState` cho phép chúng ta theo dõi trạng thái trong một thành phần hàm. Khởi tạo trạng thái của một thành phần bằng cách gọi `useState`. `useState` chấp nhận một trạng thái khởi tạo và trả về hai giá trị:

- Trạng thái hiện tại
- Hàm cập nhật trạng thái

Lưu ý: Sử dụng snippet để tạo nhanh một trạng thái.



Hình 15

Kết quả:

Tips & Tricks Trang chủ Giới thiệu Liên hệ RSS Feed



JWT creation and validation in Python using Authlib

Tác giả: Jason Mouth Chủ đề: Domain Driven Design

Authlib is a Python library that provides various OAuth, OpenID Connect, and JWT functionality. Authlib is my preferred library for JWT functionality, as it is one of the better Python implementations for JWT best practices, designed with OAuth and OpenID Connect in mind.

Bootstrap

Tailwind CSS

Xem chi tiết



C# Code Rules

Tác giả: Kathy Smith Chủ đề: Practices

The C# Compiler's name is Roslyn. Roslyn has a very large set of analyzers to check the quality of your code, but you must turn these analyzers on before they start doing anything. This post gives you some quick information on why it's important to turn these analyzers on in your C# projects, how to do that, and how to configure them.

Tìm kiếm bài viết
Các chủ đề
Bài viết nổi bật
Đăng ký nhận tin mới
Tag cloud

Hình 16

6. Tạo Pager, SearchForm, CategoriesWidget

6.1. Tạo Pager

Trong thư mục src/Components tạo tập tin Pager.js với nội dung như sau:

```
import { Link } from 'react-router-dom';
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
import { faArrowLeft, faArrowRight } from '@fortawesome/free-solid-svg-icons'
import Button from 'react-bootstrap/Button';

const Pager = ({ postquery, metadata }) => {

  let pageCount = metadata.pageCount,
      hasNextPage = metadata.hasNextPage,
      hasPreviousPage = metadata.hasPreviousPage,
      pageNumber = metadata.pageNumber,
      pageSize = metadata.pageSize,
      actionName = '', slug = '',
      keyword = postquery.keyword ?? '';

  if (pageCount > 1) {
    return (
      <div className='text-center my-4'>
        {hasPreviousPage
          ? <Link
              to={` /blog/${actionName}?slug=${slug}&k=${keyword}
                &p=${pageNumber - 1}&ps=${pageSize}`}
              className='btn btn-info'>
                <FontAwesomeIcon icon={faArrowLeft} />
                &nbsp;Trang trước
```

```

    </Link>
    : <Button variant='outline-secondary' disabled>
      <FontAwesomeIcon icon={faArrowLeft} />
      &nbsp;Trang trước
    </Button>
  }
  {hasNextPage
    ? <Link
      to={` /blog/${actionName}?slug=${slug}&k=${keyword}
&p=${pageNumber + 1}&ps=${pageSize}`}
      className='btn btn-info ms-1'>
        Trang sau&nbsp;
        <FontAwesomeIcon icon={faArrowRight} />
      </Link>
      : <Button variant='outline-secondary' className='ms-1' disabled>
        Trang sau&nbsp;
        <FontAwesomeIcon icon={faArrowRight} />
      </Button>
    }
  </div>
);
}
return (
  <Link>
  </Link>
);
}

```

export default Pager;

Sau đó cập nhật lại tập tin src/Pages/Index.js:

```

import React, { useEffect, useState } from 'react';
import { useLocation } from 'react-router-dom';
import PostItem from '../Components/PostItem';
import Pager from '../Components/Pager';
import { getPosts } from '../Services/BlogRepository';

const Index = () => {
  const [postList, setPostList] = useState([]);
  const [metadata, setMetadata] = useState({});

  function useQuery() {
    const { search } = useLocation();
    return React.useMemo(() => new URLSearchParams(search), [search]);
  }

  let query = useQuery(),
    k = query.get('k') ?? '',
    p = query.get('p') ?? 1,

```

```
ps = query.get('ps') ?? 10;

useEffect(() => {
  document.title = 'Trang chủ';

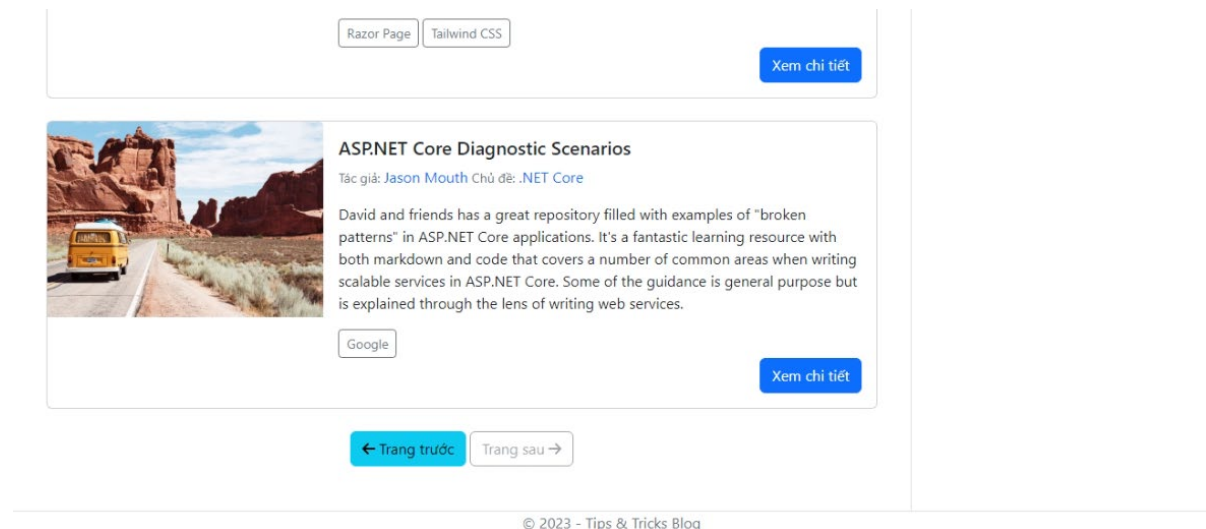
  getPosts(k, ps, p).then(data => {
    if (data) {
      setPostList(data.items);
      setMetadata(data.metadata);
    }
    else
      setPostList([]);
  })
}, [k, p, ps]);

useEffect(() => {
  window.scrollTo(0, 0);
}, [postList]);

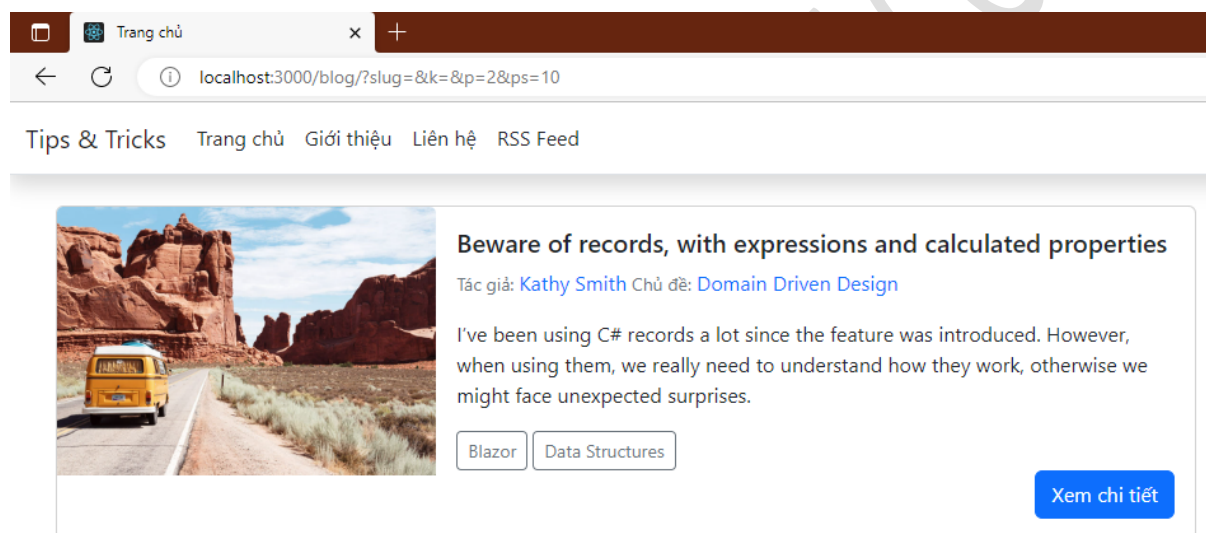
if (postList.length > 0)
  return (
    <div className='p-4'>
      {postList.map((item, index) => {
        return (
          <PostItem postItem={item} key={index} />
        );
      })}
      <Pager postquery={{ 'keyword': k }} metadata={metadata} />
    </div>
  );
else return (
  <></>
);
}
```

`export default Index;`

Kết quả trên web sẽ hiển thị nút điều hướng sang trang trước đó hoặc trang tiếp theo nếu như dữ liệu được phân trang:



Hình 17



Hình 18

6.2. Tạo SearchForm

Trong thư mục src/Components tạo tập tin SearchForm.js với nội dung như sau:

```
import { useState } from 'react';
import Form from 'react-bootstrap/Form';
import Button from 'react-bootstrap/Button';
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
import { faSearch } from '@fortawesome/free-solid-svg-icons';

const SearchForm = () => {
  const keyword = useRef('');

  const handleSubmit = (e) => {
    e.preventDefault();
    window.location = `/blog?k=${keyword.current.value}`;
  };
}
```



```

    });

    return (
      <div className='mb-4'>
        <Form method='get' onSubmit={handleSubmit}>
          <Form.Group className='input-group mb-3'>
            <Form.Control
              type='text'
              name='k'
              ref={keyword}
              aria-label='Enter keyword'
              aria-describedby='btnSearchPost'
              placeholder='Enter keyword' />
            <Button
              id='btnSearchPost'
              variant='outline-secondary'
              type='submit'>
              <FontAwesomeIcon icon={faSearch} />
            </Button>
          </Form.Group>
        </Form>
      </div>
    );
  }
}

```

```
export default SearchForm;
```

Sửa lại nội dung trong tập tin src/Components/Sidebar.js như sau:

```

...
    <div className='pt-4 ps-2'>
      <SeachForm />

      <h1>
        Các chủ đề
      </h1>
    </div>
  ...

```

Lưu ý: Sử dụng useRef không kết xuất lại (re-render) thành phần trong khi đó useState thì có. Chúng ta có thể sử dụng useState hoặc useRef nếu muốn lưu trữ dữ liệu sau khi kết xuất. Cơ chế cập nhật của useState là bất đồng bộ (asynchronous) còn của useRef là đồng bộ (synchronous)².

6.3. Tạo CategoriesWidget

Trong thư mục src/Services tạo tập tin Widgets.js với nội dung như sau:

```
import axios from 'axios';
```

² Nguồn: <https://stackoverflow.com/questions/56455887/react-usestate-or-userref>

```
export async function getCategories() {
  try {
    const response = await
    axios.get('https://localhost:7085/api/categories');
    const data = response.data;
    if (data.isSuccess)
      return data.result;
    else
      return null;
  } catch (error) {
    console.log('Error', error.message);
    return null;
  }
}
```

Lưu ý: Tập tin Widgets.js chứa các hàm gọi API dành cho các widget ở sidebar. Nếu máy chủ API chưa có API này hoặc cần tham số thì điều chỉnh lại mã nguồn cho phù hợp.

Trong thư mục src/Components tạo tập tin CategoriesWidget.js với nội dung như sau:

```
import { useState, useEffect } from 'react';
import ListGroup from 'react-bootstrap/ListGroup';
import { Link } from 'react-router-dom';
import { getCategories } from '../Services/Widgets';

const CategoriesWidget = () => {
  const [categoryList, setCategoryList] = useState([]);

  useEffect(() => {
    getCategories().then(data => {
      if (data)
        setCategoryList(data);
      else
        setCategoryList([]);
    });
  }, []);

  return (
    <div className='mb-4'>
      <h3 className='text-success mb-2'>
        Các chủ đề
      </h3>
      {categoryList.length > 0 &&
        <ListGroup>
          {categoryList.map((item, index) => {
            return (
              <ListGroup.Item key={index}>
                <Link to={`/blog/category?slug=${item.urlSlug}`}`>

```

```

        title={item.description}
        key={index}>
        {item.name}
        <span>&nbsp;({item.postCount})</span>
      </Link>
    </ListGroup.Item>
  );
  }}}
</ListGroup>
}
</div>
);
}

```

export default CategoriesWidget;
 Thêm CategoriesWidget vào Sidebar.js:

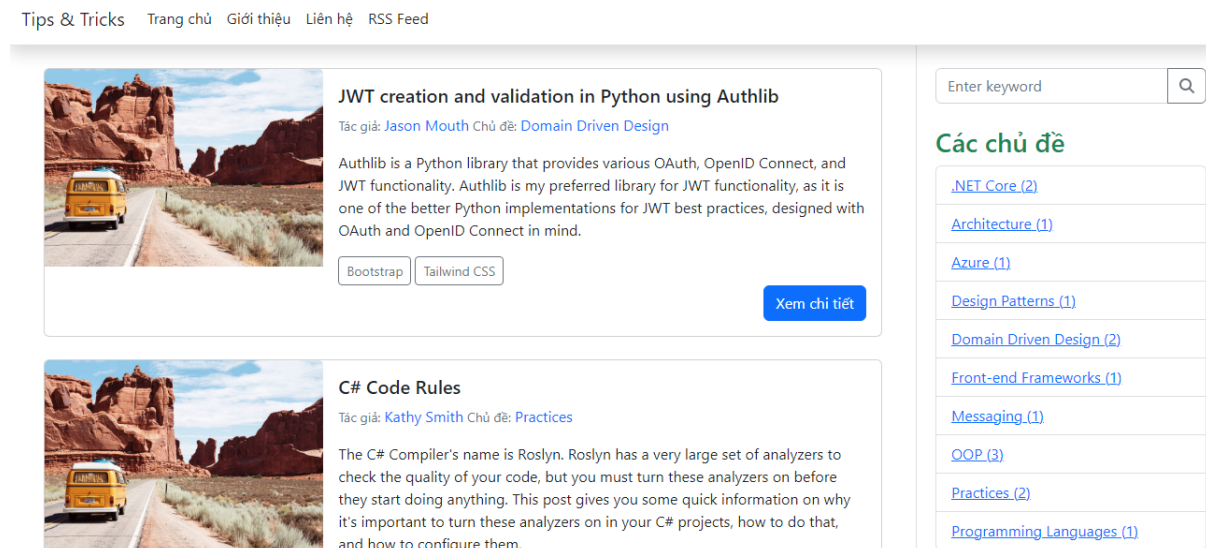
```

...
<div className='pt-4 ps-2'>
  <SearchForm />

  <CategoriesWidget />
...

```

Kết quả:



Hình 19

C. Bài tập thực hành

Lưu ý: Nội dung phần **C. Bài tập thực hành** này tương tự bài Lab 2 nhưng thực hiện trên dự án web React ở phần **B. Hướng dẫn thực hành** đã làm ở nhà.

1. Thay đổi nội dung trong `src/Components/PostItem.js` để hiển thị tiêu đề bài viết, tên tác giả và tên chuyên mục dưới dạng liên kết sau (chỉ tạo liên kết):
 - <https://localhost:3000/blog/author/kathy-smith>
 - <https://localhost:3000/blog/category/domain-driven-design>
 - <https://localhost:3000/blog/post/2023/2/1/blog-post-title>
2. Tạo các thành phần cho sidebar:
 - **FeaturedPosts:** Hiển thị TOP 3 bài viết được xem nhiều nhất. Người dùng có thể click chuột để xem chi tiết.
 - **RandomPosts:** Hiển thị TOP 5 bài viết ngẫu nhiên. Người dùng có thể click chuột để xem chi tiết.
 - **TagCloud:** Hiển thị danh sách các thẻ (tag). Khi người dùng click chuột vào thẻ nào thì hiển thị danh sách bài viết chứa thẻ đó.
 - **BestAuthors:** Hiển thị TOP 4 tác giả có nhiều bài viết nhất. Khi người dùng click chuột vào tên tác giả, hiển thị danh sách bài viết của tác giả đó.
 - **Archives:** Hiển thị danh sách 12 tháng gần nhất và số lượng bài viết trong mỗi tháng dưới dạng các liên kết. Khi người dùng click chuột vào tháng nào thì hiển thị danh sách bài viết được đăng trong tháng đó. Định dạng: November 2022 (5), February 2023 (11), ...
3. Tạo giao diện đăng ký nhận thông báo khi có bài viết mới
4. Tạo giao diện thảo luận, đánh giá bài viết
5. Tạo giao diện cho trang Liên hệ
6. Thêm chức năng chia sẻ bài viết lên mạng xã hội
7. Tùy chỉnh blog đẹp hơn.

D. Tài liệu tham khảo

[1] "React Tutorial," W3schools, <https://www.w3schools.com/react/default.asp>

--- HẾT ---