ROUTER

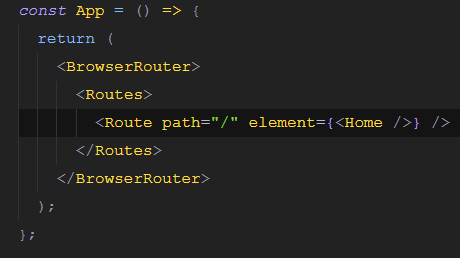
1. **Router**

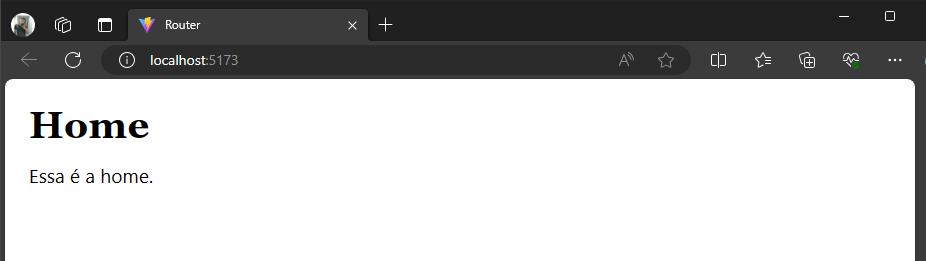
**Instalar a extensão que permite gerenciarmos rotas.**

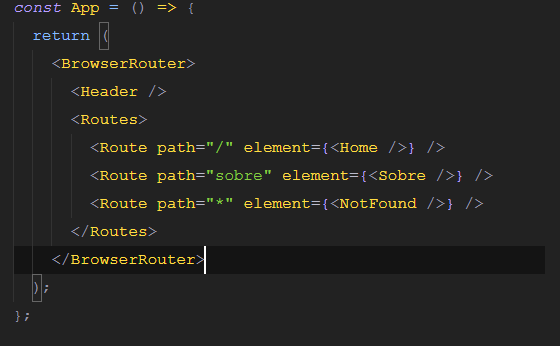
**npm install react-router-dom**

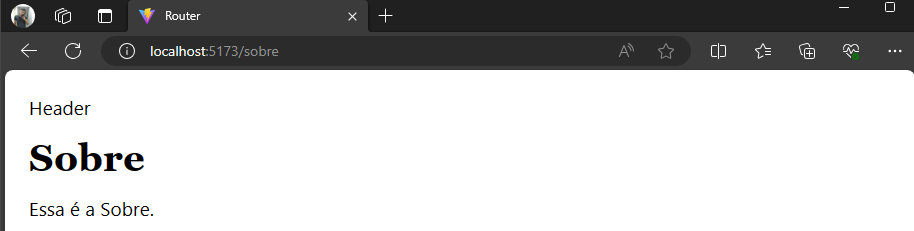
* 1. **– BrowserRouter, Routes, Route**

**BrowserRouter é o componente pai que envolve tudo que depender do react-router. O Routes define a área em que vamos colocar os nossos Route. O Route recebe um caminho em path, se esse caminho for o mesmo do URL ele irá renderizar o componente que estiver dentre de element={}.**

****

****

****

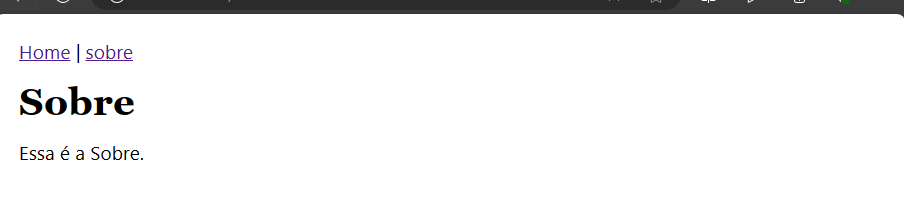
****

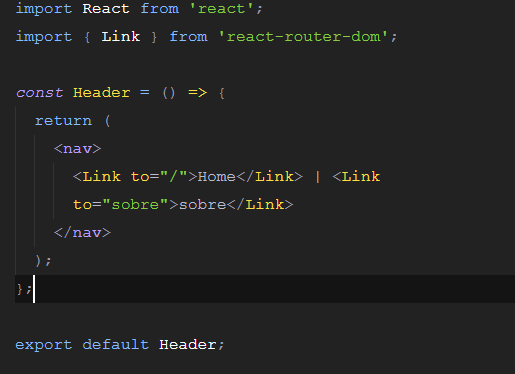
**\*Detalhes: Header fora do Routes, ou seja vai aparecer em toda a aplicação independente do url, se botamos /sobre mostra o componente de <Sobre /> e qualquer outra coisa que não seja sobre depois da barra vai para a página de NOTFOUND.**

1. **Link**

**2.1 -Link**

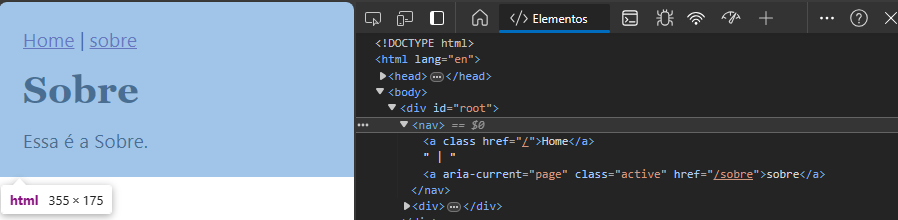
**O Link irá modificar a rota e renderizar o novo componente sem recarregar a página. Age como se fosse um <a></a>**

****

****

**2.2 -NavLink**

**NavLink faz função parecida com a de link, só que ele mostra qual Rota está ativa.**

****

* **Pode perceber que na tag a de sobre tem a class=”active”**

**2.3 -useNavigate**

**O hook useNavigate permite navegarmos programaticamente entre as rotas. Por exemplo, pode ser utilizado quando o usuário faz um login bem sucedido e enviamos o mesmo a página da sua conta.**

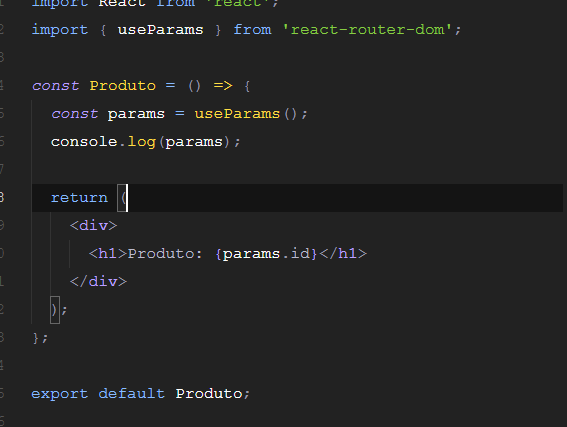
****

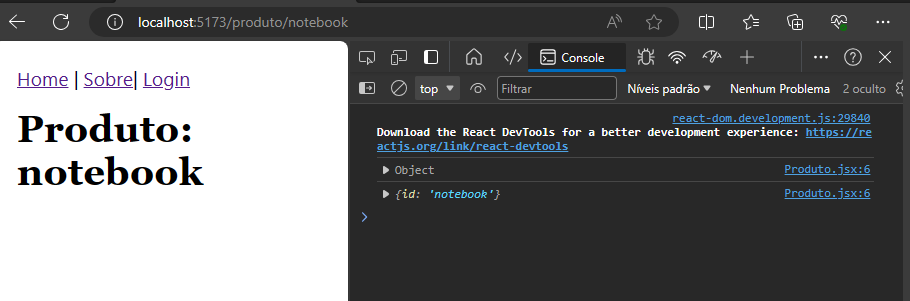
**Usando o useNavigate do router dom o que esta acontecendo nesse exemplo é que ao clicar no botão de login ativa a função de handleClick que utiliza do hook usenavigate para direcionar para a página sobre, em um exemplo real direcionaria para a página do usuário do login.**

1. **useParams**

**3.1- Rota Dinâmica**

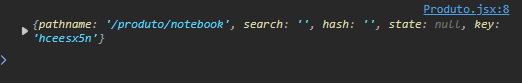
**O uso de :nome irá definir uma rota dinâmica, onde o nome poderá ser utilizado como uma variável dentro do componente. Serve para buscarmos rotas dinâmicas como, ex: produto/notebook ou produto/smartphone.**

****

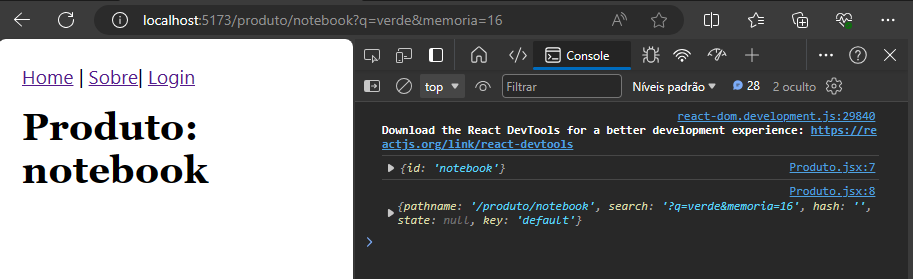
****

**Usa isso para fazer um fetch passando uma url especifica, e no final da url passa o parâmetro exato do produto que quer puxar.**

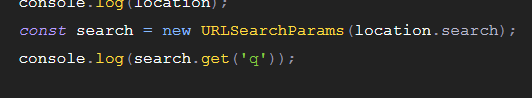
**3.2- useLocation**

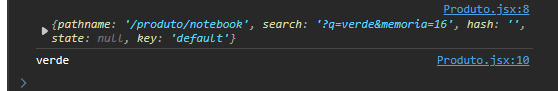
****

**Trás informações da rota, importante é o search, que é basicamente o que vem depois da (?) em uma página de busca por exemplo.**

****

**Exemplo de uso:**

****

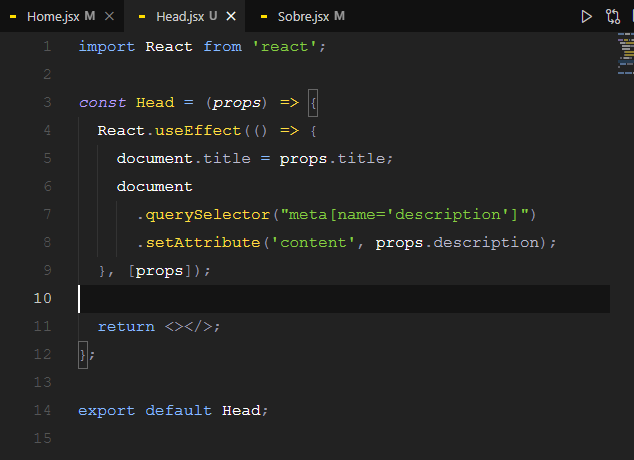
****

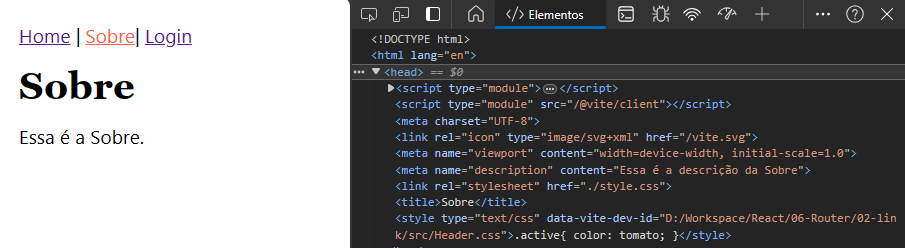
1. **Nested Routes(rotas aninhadas)**

**Utilizamos nestd routes quando precisamos de rotas dentro de rotas. Como por exemplo: perfil/editar e perfil/certificados e etc. Utilize o \\* para definir que existem outras rotas dentro.**

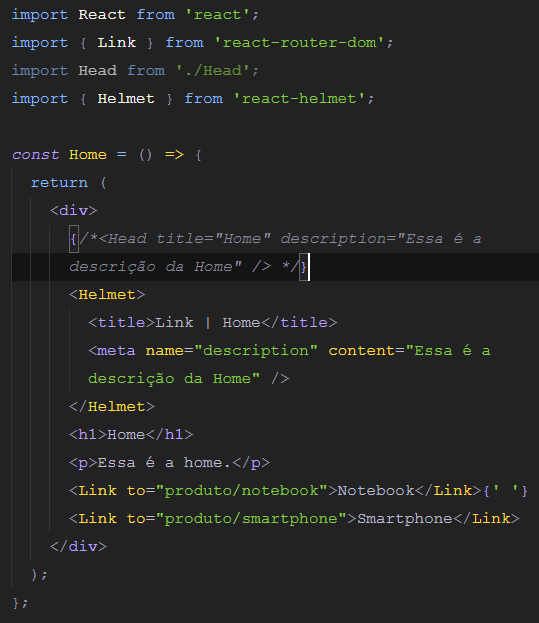
1. **Head**

**No react não temos acesso direto as tags e informações do Head. Porém com o uso de rotas é essencial realizar a mudança do título e descrição para cada rota. Podemos fazer isso através de um componente ou custom Hook. Head.js**

****

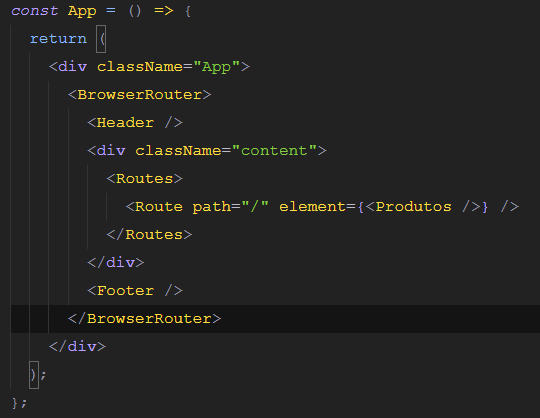
****

**Também tem como fazer isso através de uma dependência, chamada React-helmet.**

****

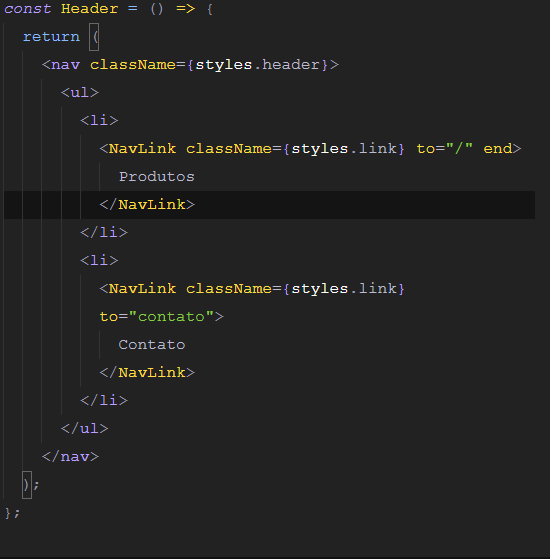
**6-Projeto**

**1º Passo:**

****

**Utilizar React-router-dom pra definir o escopo principal da aplicação com Header e Footer fora dos routes, e com a página principal colocada dentro do routes com path= ‘/’.**

**2º Passo:**

****

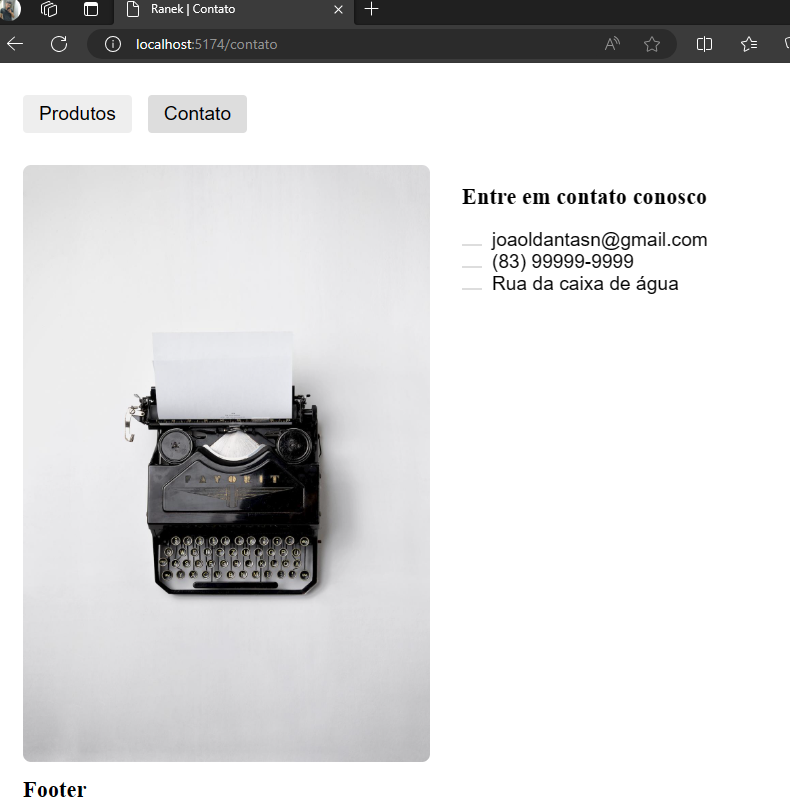
**Adiciona-se NavLink no Header, para facilitar a navegação, o NavLink funciona como o <a></a> no html e diferencia-se do <Link></Link> pois mostra a página ativa no momento.**

****

**ps: O Header foi estilizado**

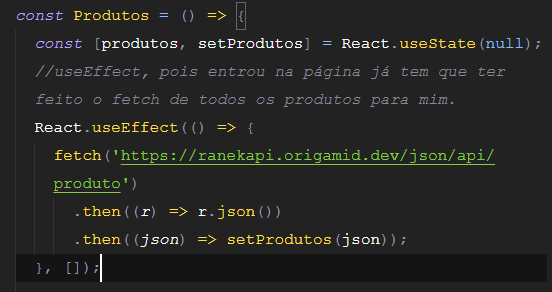
**3º Passo:**

**Nesse exemplo em especifico foi a estilização da página de Contato, juntamente com a adição do Head na sua página.**

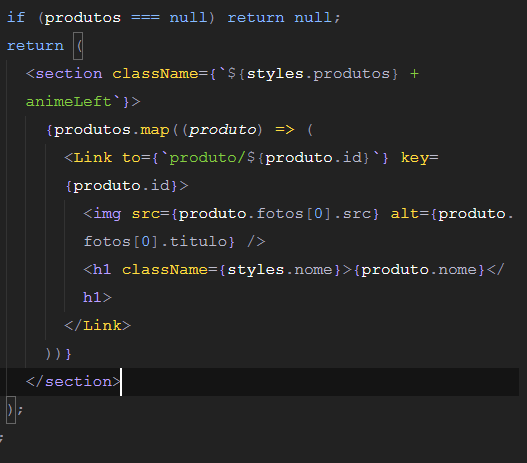
****

**4º Passo:**

**Montar a página inicial que é a de produtos, nela vai ter um fetch de produtos. Utilizando-se o useEffect, pois quando entrarmos na página os elementos já devem estar carregados nela**

****

**Utilizamos o UseState para salvar os elementos carregados do fetch em algum local e para assim o dispor eles na tela:**

****

**Atribuimos tanto o img quanto o h1 dentro do Link pois ao clicar neles, você será redirecionado para a página especifica do Produto, mas antes temos que mudar no App.**

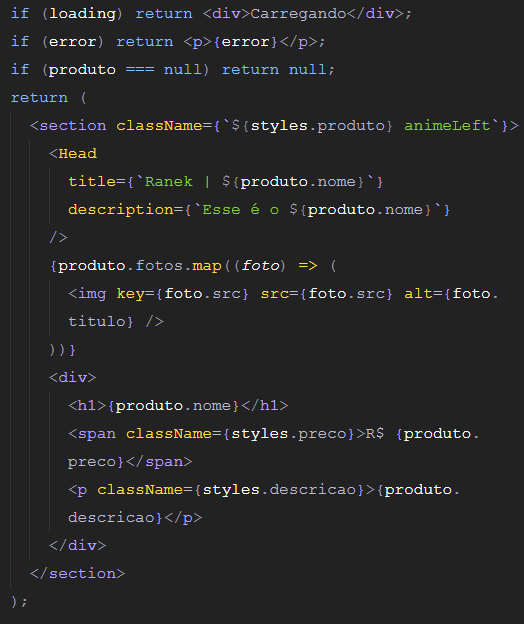
****

**5º Passo:**

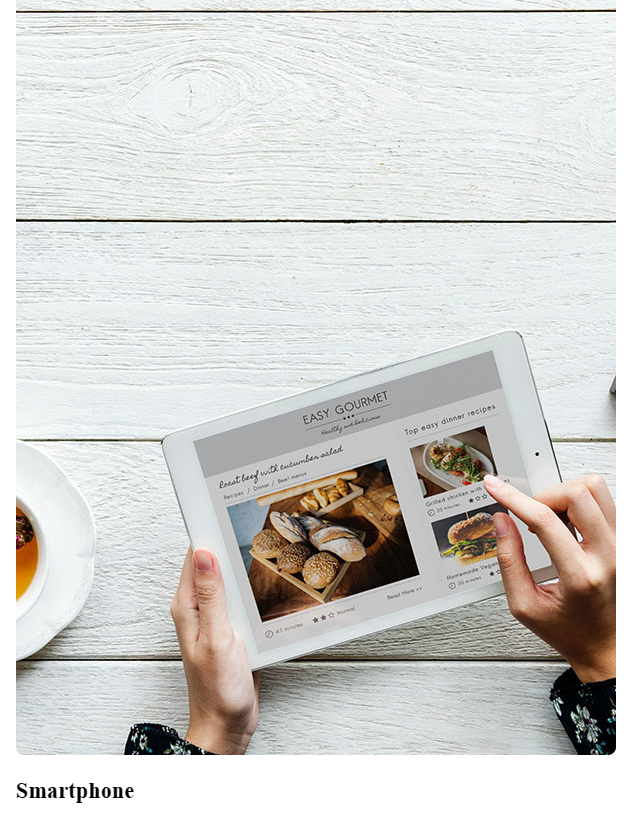
****

**A mesma questão de produtos, usa o useEffect, dessa vez fiz com async await utilizando loading e error tb. Importante ressaltar o uso do useParams que vai pegar o id pra procurar na lista de produtos, o id seria o id do produto que vai no url:**

****

****

**Usando o Head também, para mudar o nome da página de produto, resultado final da página sem o css:**

****

**6º Passo:**

**Editar o css de produto e footer**