# **PLANO DE AULA**

Aula 21 | Tempo estimado: 1 hora e 30 minutos | 9+ anos

Tipo da atividade: Offline

Ferramenta(s): Computador. Git, nodeJS e VS Code

Conteúdos

* React Router.

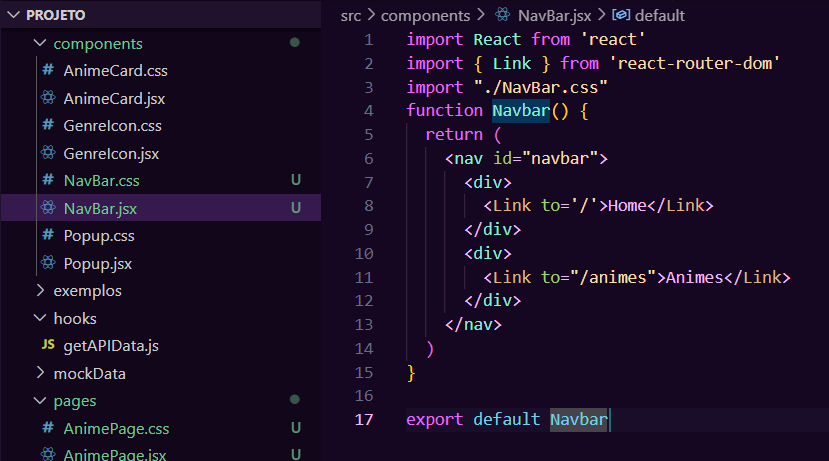
Objetivos

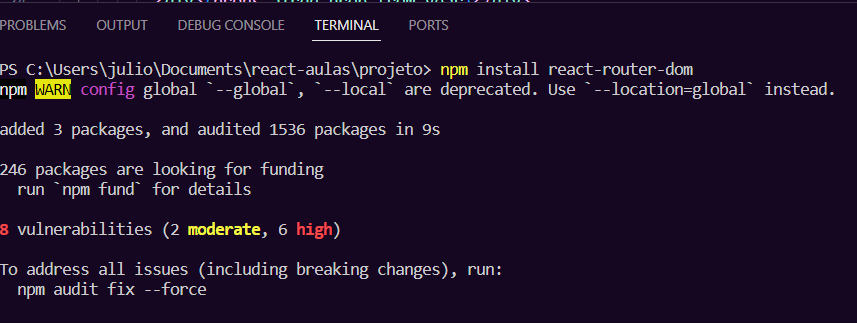
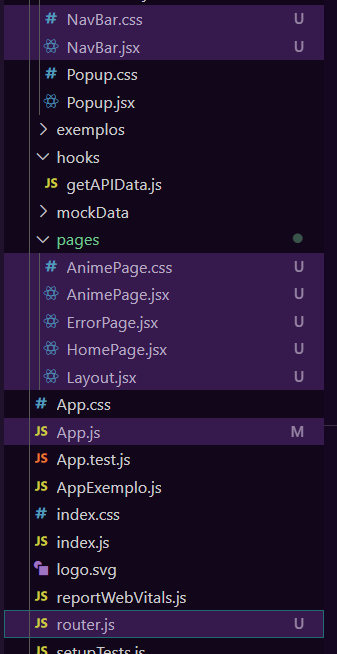
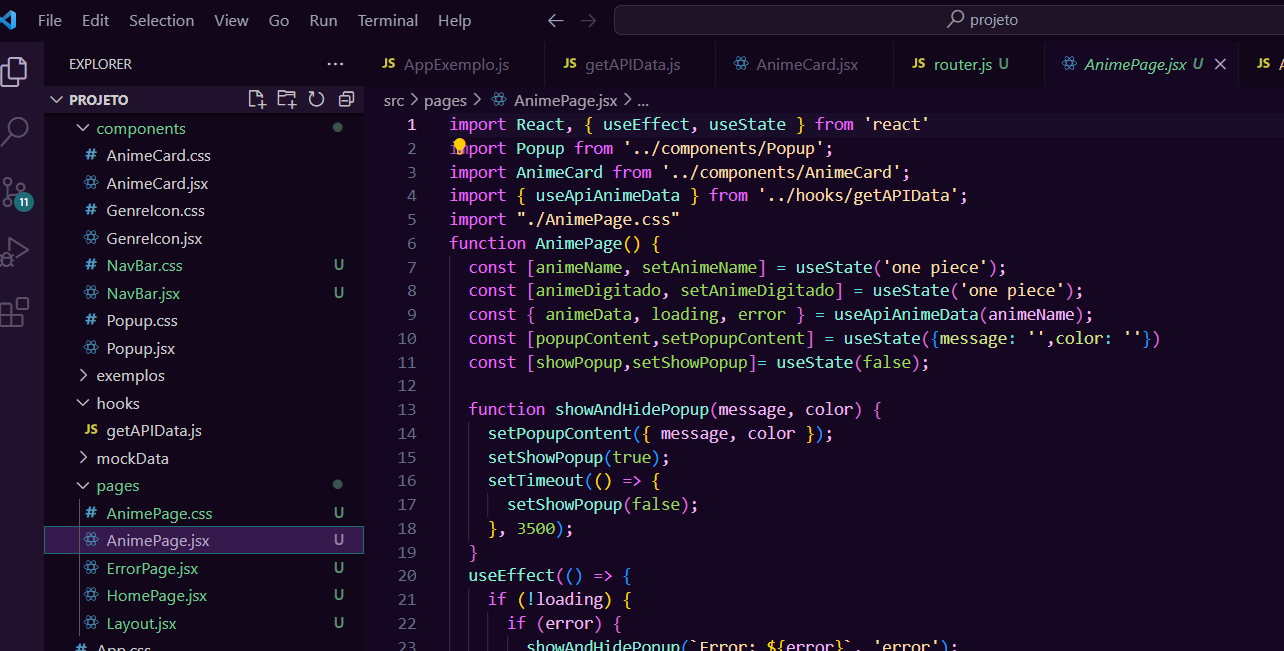
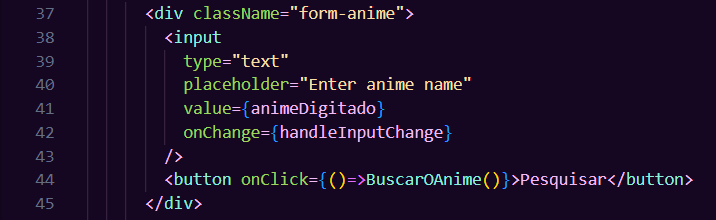
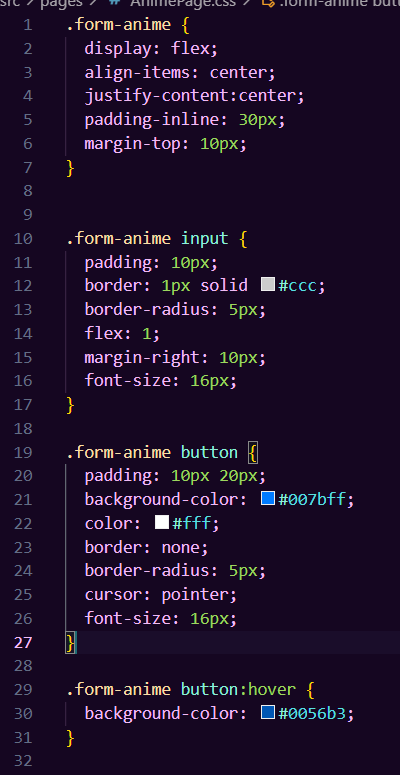
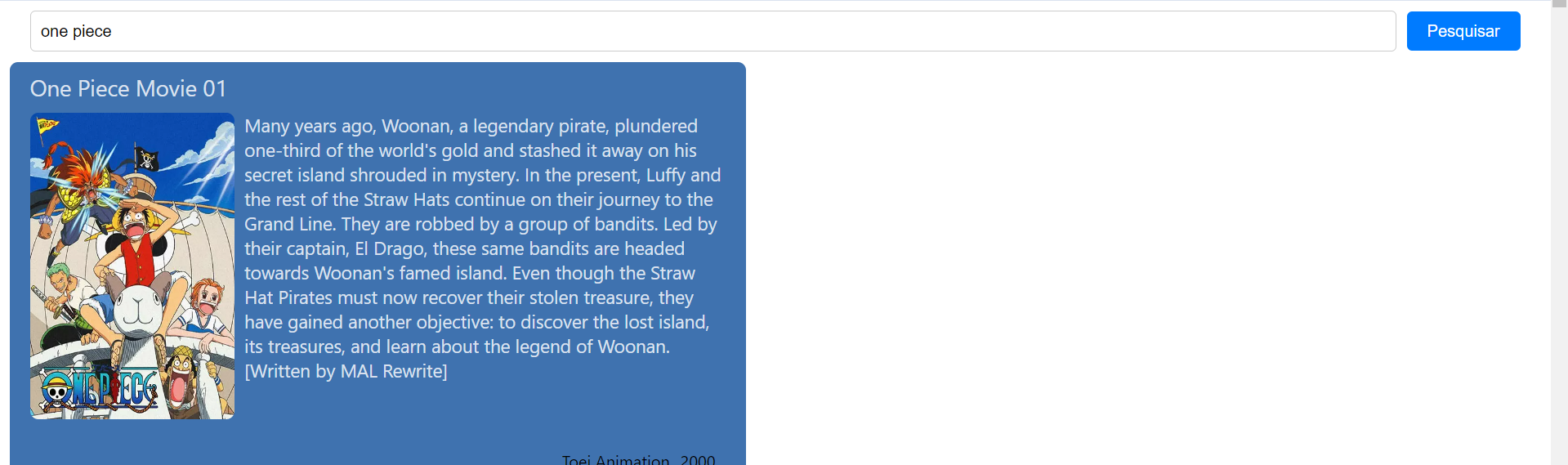
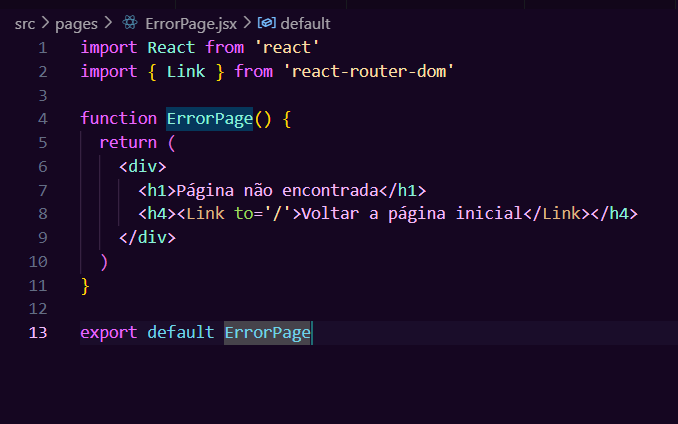
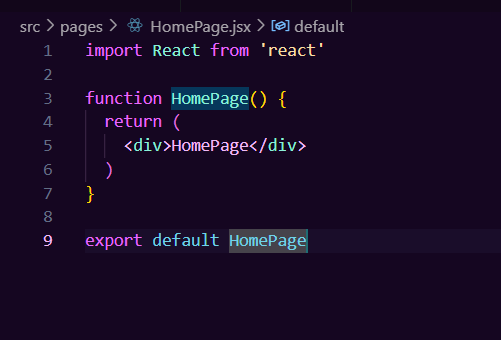
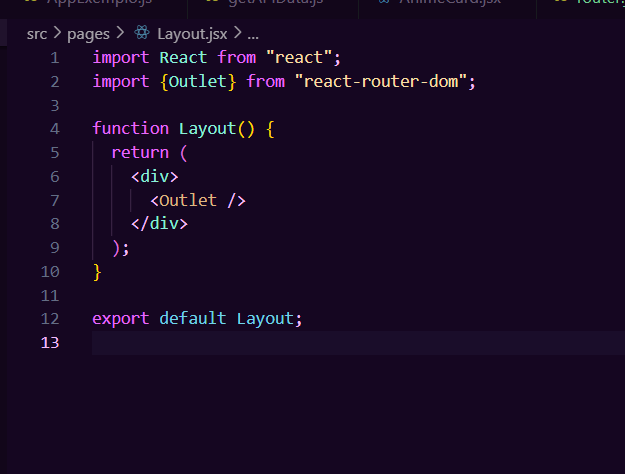
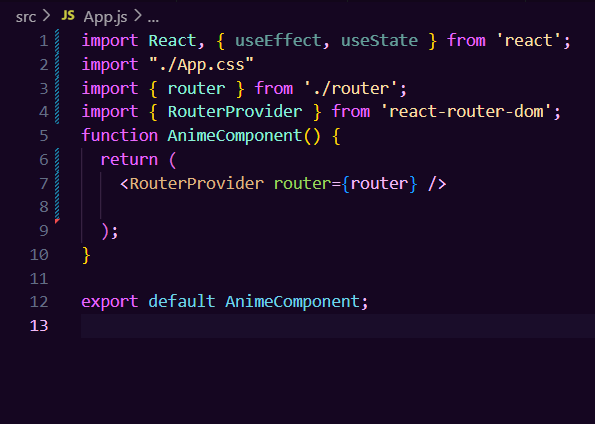
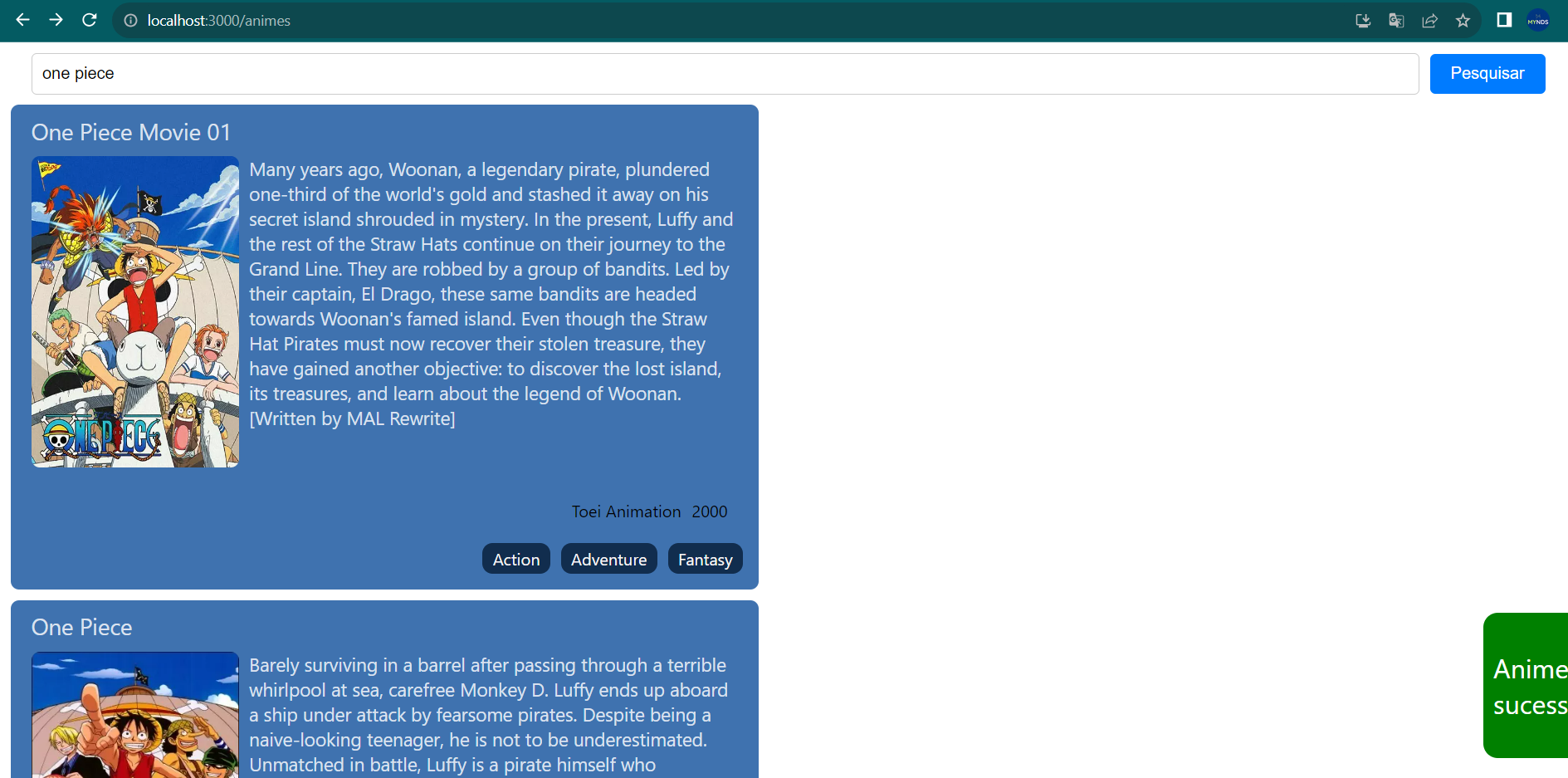
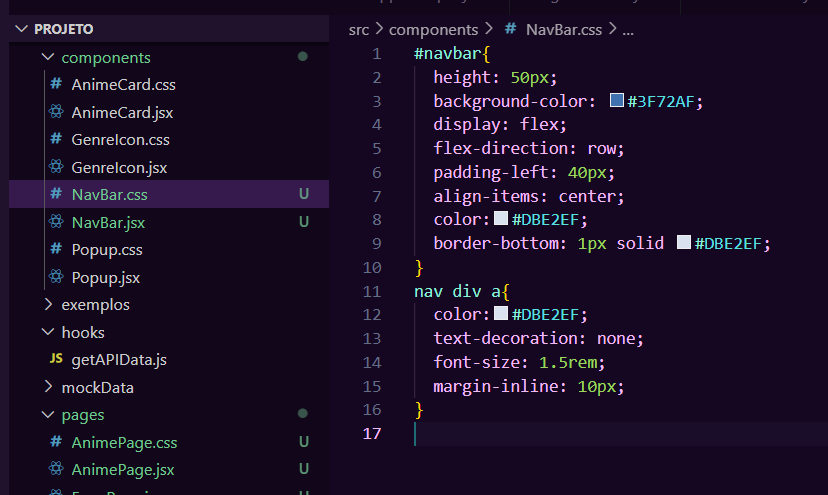
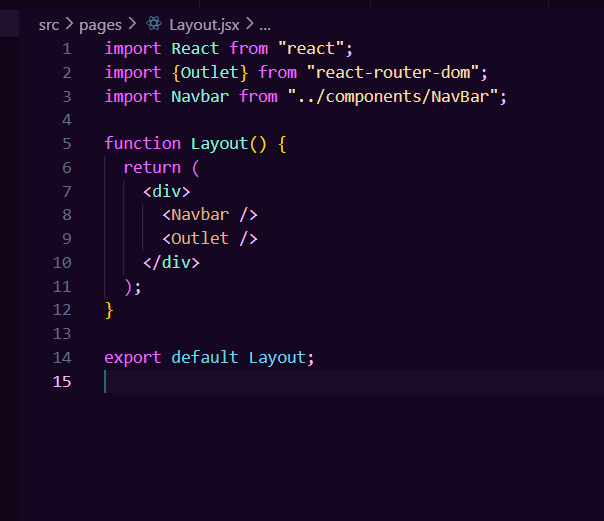
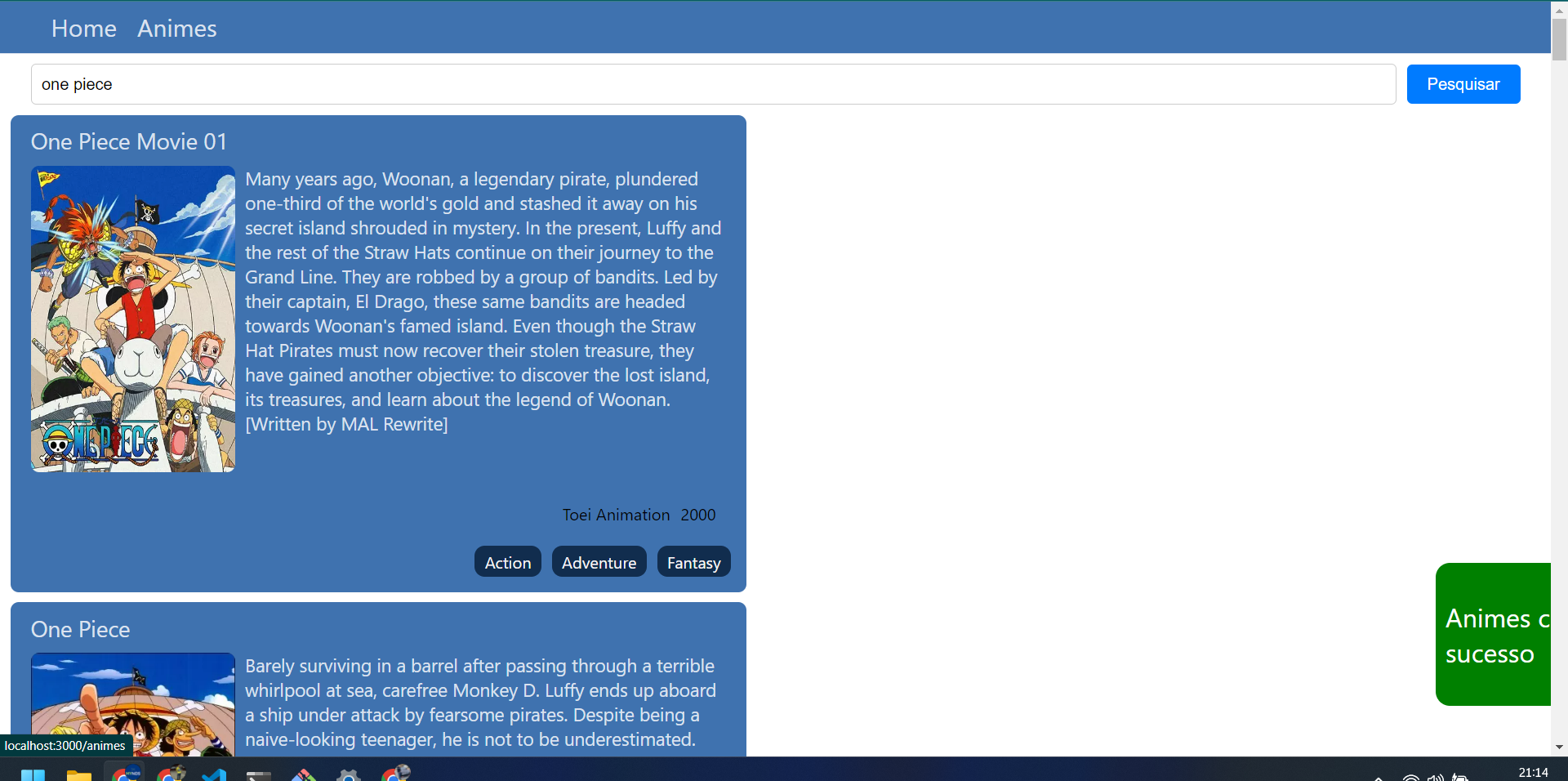
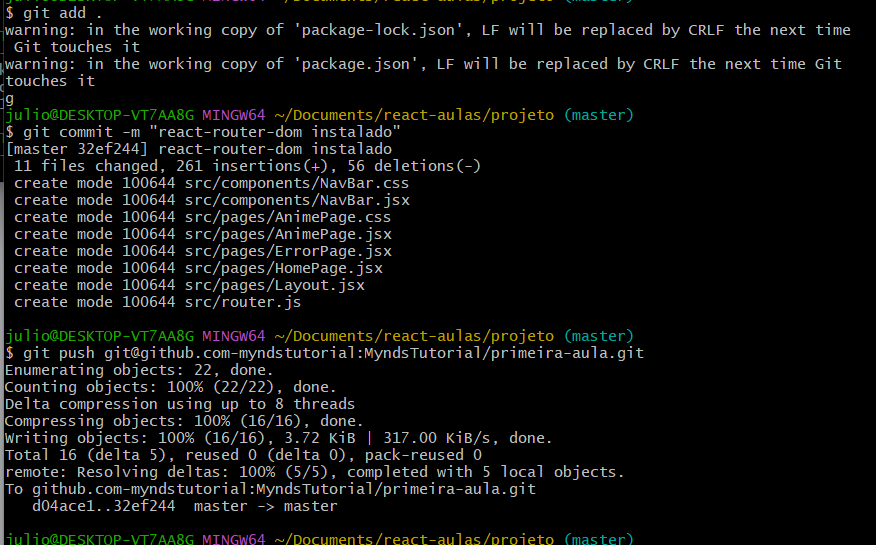
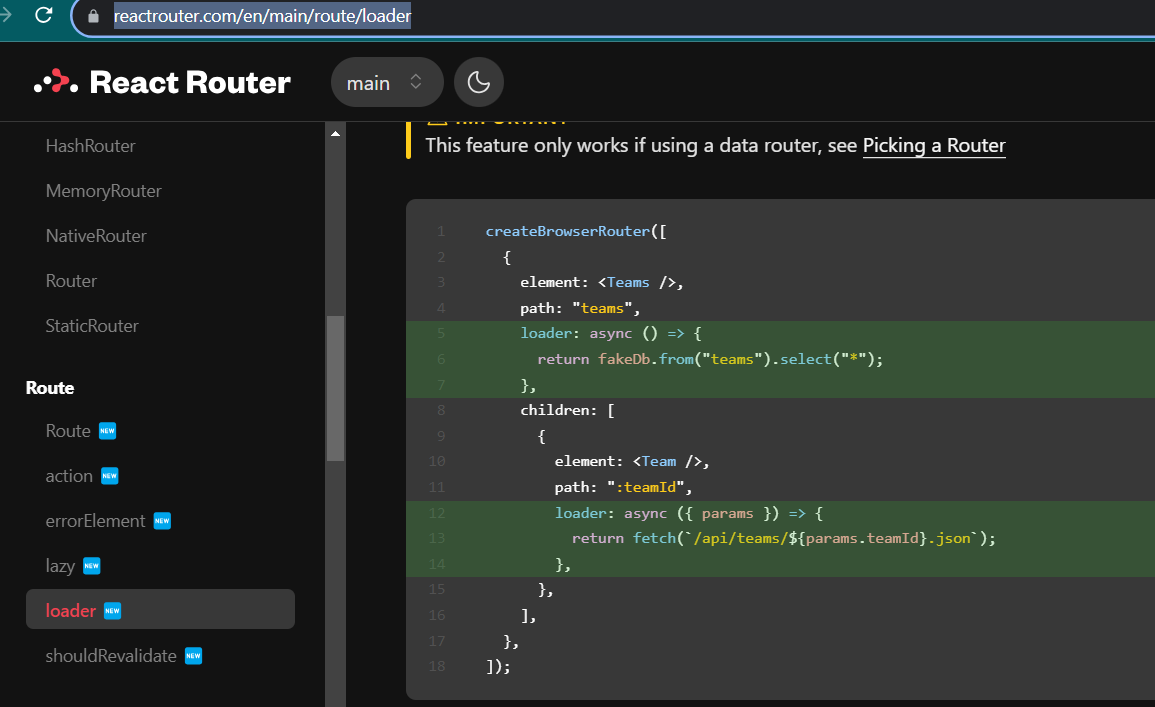
* Ensinar sobre React;
* Aprender a instalar bibliotecas;
* Configurar um *package.json*.

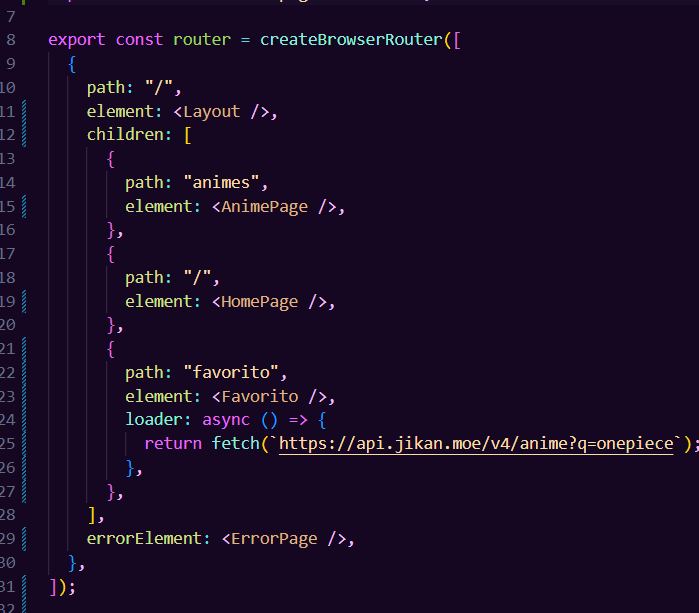
Estratégias e atividades

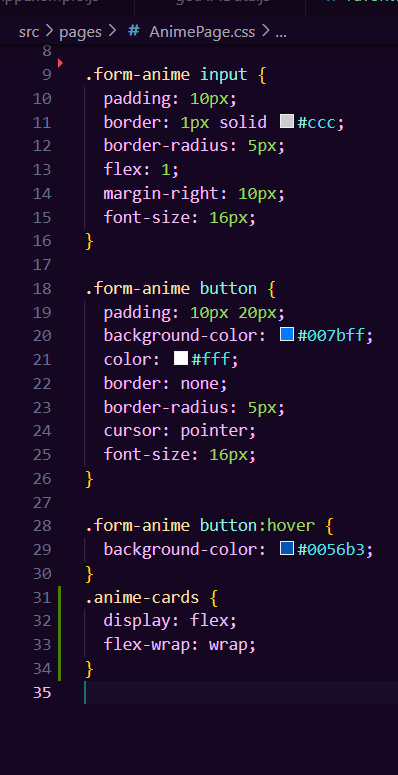
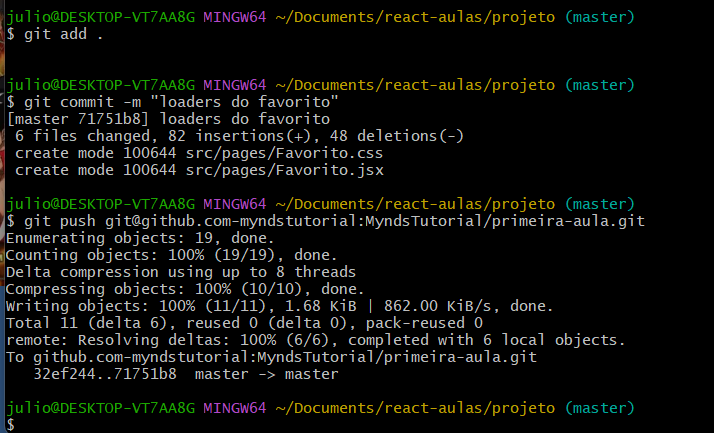
* Criar um App que consome API.

1. Hoje vamos instalar a biblioteca do *react-router-dom*, que irá permitir ter várias páginas.



1. Instalar a biblioteca:  
   
2. Primeiro de tudo, precisamos entender o que são rotas. No HTML puro, nós temos que criar códigos HTML inteiros e isso deixa o nosso site mais pesado, pois tem que recarregar totalmente os scripts, então nós criaremos páginas diferentes usando o mesmo App.js, e isso faz com que o projeto seja incrível. Então vamos começar entendendo os arquivos que vamos criar:  
   
3. Vamos começar a entender o que de fato foi alterado. Todo o nosso código do App.js foi para o *AnimePage.jsx*.
4. Agora, devemos editar aquele *input* para dar a ele uma qualidade melhor:  
     
   E o CSS:  
   
5. E o resultado será esse:  
   
6. Vamos criar a página de *ErrorPage.jsx*:  
   
7. A HomePage.jsx:  
   
8. O *Layout.jsx* é o arquivo “principal”, onde podemos colocar tudo que for ser repetido, como *navbar, popups*, etc.  
   
9. Dentro do *router.js*, vamos criar as configurações das rotas:  
   
10. E para utilizá-lo é só importar no App.js:  
    
11. Mas repare que agora se você for no *localhost:3000* algo mudou:  
    
12. Estamos na página inicial, porém não temos como ir até a página de animes, se digitarmos lá em cima localhost:3000/animes, veremos que automaticamente irá carregar a página:  
    
13. Para podermos transitar entre as telas, vamos criar uma navbar que será responsável por nos permitir trocar as telas:  
      
    
14. E agora para finalizar, vamos importá-la na tela geral, onde será responsável por todas as páginas:  
    O outlet significa o componente que vai ser renderizado, que no caso é a nossa página configurada lá no *router.js*.  
      
    
15. Agora nosso projeto nos permite transitar entre as telas:  
    
16. Para finalizar, vamos subir o código no Github.
17. Criar um App que consome API.
18. Agora vamos aprender como funciona ler a documentação da biblioteca.
19. Vamos até a documentação da biblioteca e procurar sobre *Loaders*:  
    <https://reactrouter.com/en/main/route/loader>  
    
20. Como podemos ver, nós conseguimos renderizar uma tela, enquanto ela carrega chamar uma função para garantir que tem algo que estará lá quando iniciarmos. Vamos criar um componente chamado *<Favorito />* em pages, e com isto vamos criar um *Loader* para ele:



1. Agora, vamos criar a rota na navbar para poder ir até o favorito:  
   
2. Vamos colocar o CSS para poder fazer os cards irem um ao lado do outro:  
   
3. Agora, como ficou o *Favorito.jsx*:  
   
4. Agora, nos próximos minutos os alunos vão procurar pela documentação e ferramentas que podem ser úteis na construção do projeto, e tentar implementar dentro da *HomePage* do *site.Dicas*:
   1. redirect;
   2. userNavigation;
   3. useRoutes;
   4. useParams.
5. Para finalizar, vamos subir o código no Github.

Recursos

* Git Chrome e VS Code.

Observação

* Sempre que você quiser testar algum exemplo em JavaScript, você pode digitar um código direto no navegador como por exemplo:



* Professor: por se tratar de uma API de animes, ela é como se fosse o Google. Muito cuidado com o que o aluno for pesquisar! Ele tem acesso a internet pois é uma aula de web 🙂
* No começo da aula você pode abrir o site onde mostra as especificações do react-router-dom: <https://reactrouter.com/en/main/start/tutorial>

Tarefas

* Sem tarefa.