

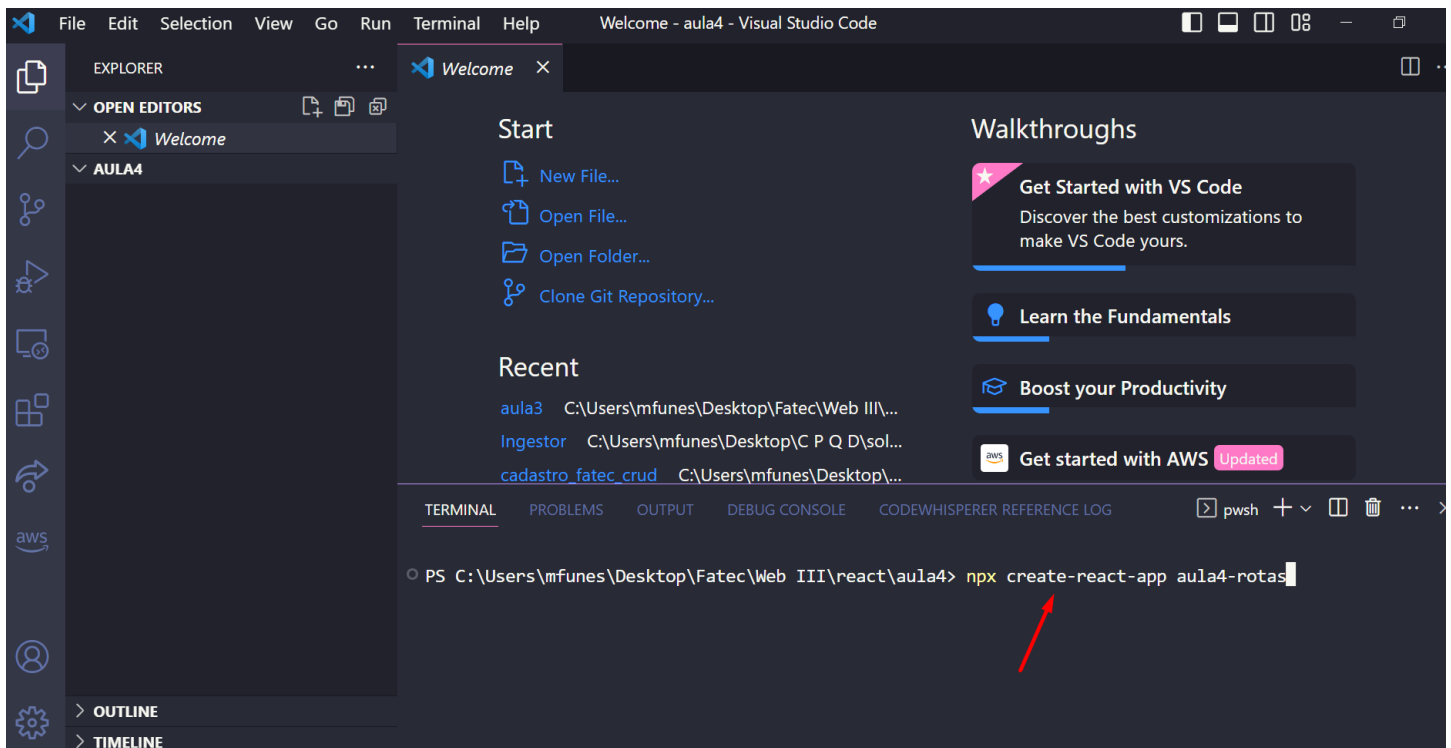
Aula 4 – Introdução ao Front-End

Objetivo da aula:

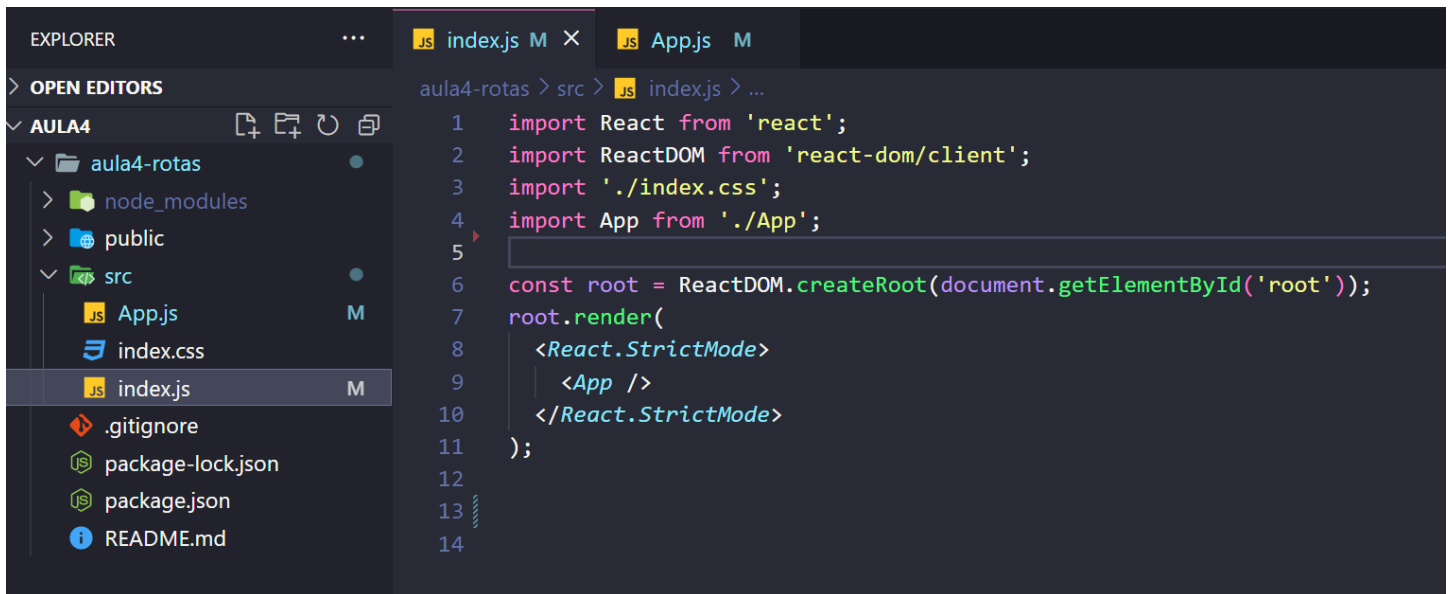
- Trabalhando com Rotas em React
- Navegação de páginas no navegador
- Trabalhando com links internos
- Header compartilhado
- Rota “Not Found”
- Parâmetros Dinâmicos em Rotas

Trabalhando com Rotas

01 – Crie um novo projeto chamado aula4-rotas.



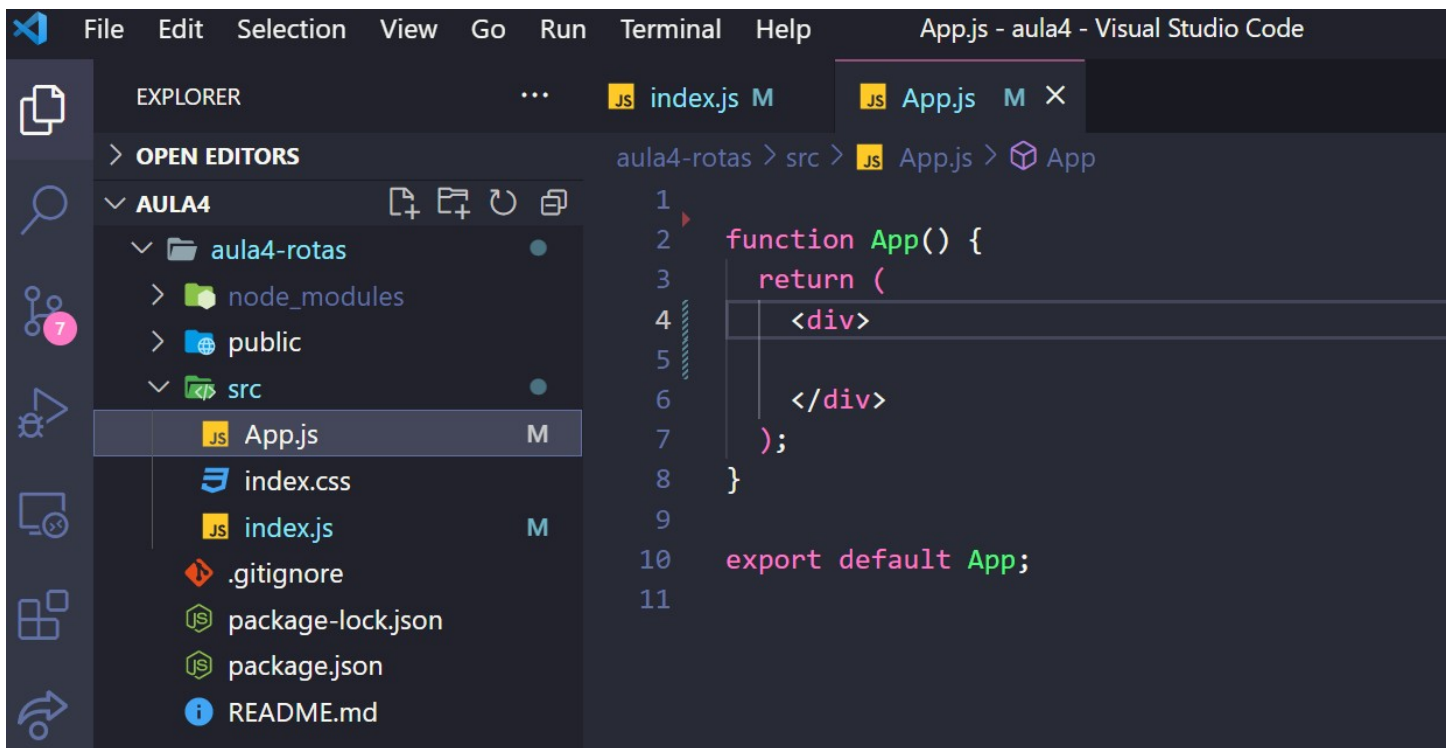
02 – Vamos deixar o projeto mais enxuto, dele alguns arquivos criados pelo React que não serão usados, deixando o projeto como abaixo. Aproveite e no index.js também retire alguns imports



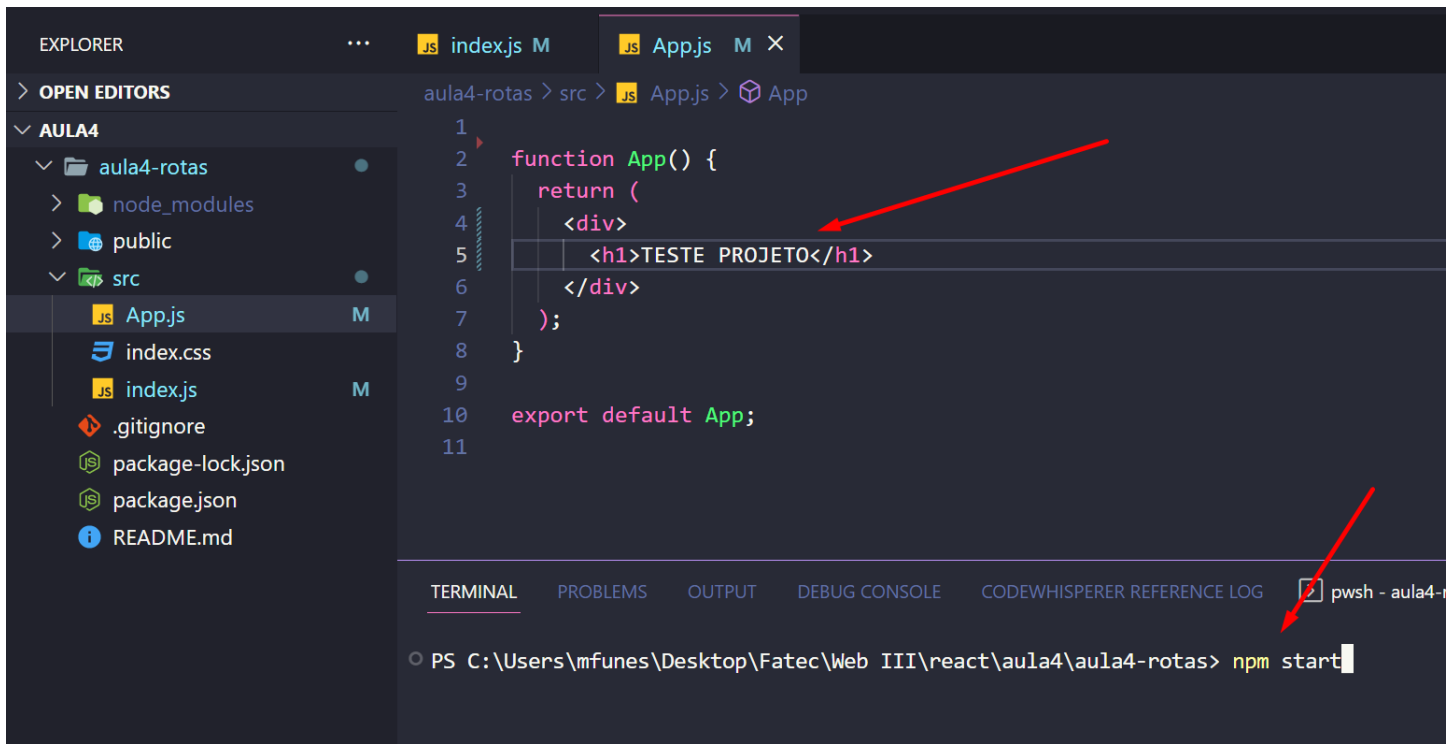
The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays the project structure for 'AULA4'. The 'src' folder is expanded, showing 'App.js', 'index.css', and 'index.js'. The 'index.js' file is selected. The main editor area shows the content of 'index.js' with the following code:

```
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5
6 const root = ReactDOM.createRoot(document.getElementById('root'));
7 root.render(
8   <React.StrictMode>
9     <App />
10  </React.StrictMode>
11 );
12
13
14
```

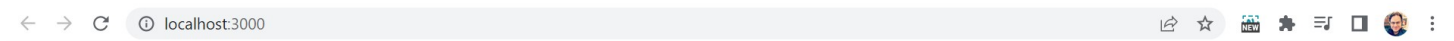
03 – No arquivo app.js deixe apenas o componet vazio. Assim nosso projeto está enxuto e pronto para ser usado:



04 – Vamos verificar se está funcionando o projeto, coloque uma tag H1 com TESTE PROJETO e dê um npm start

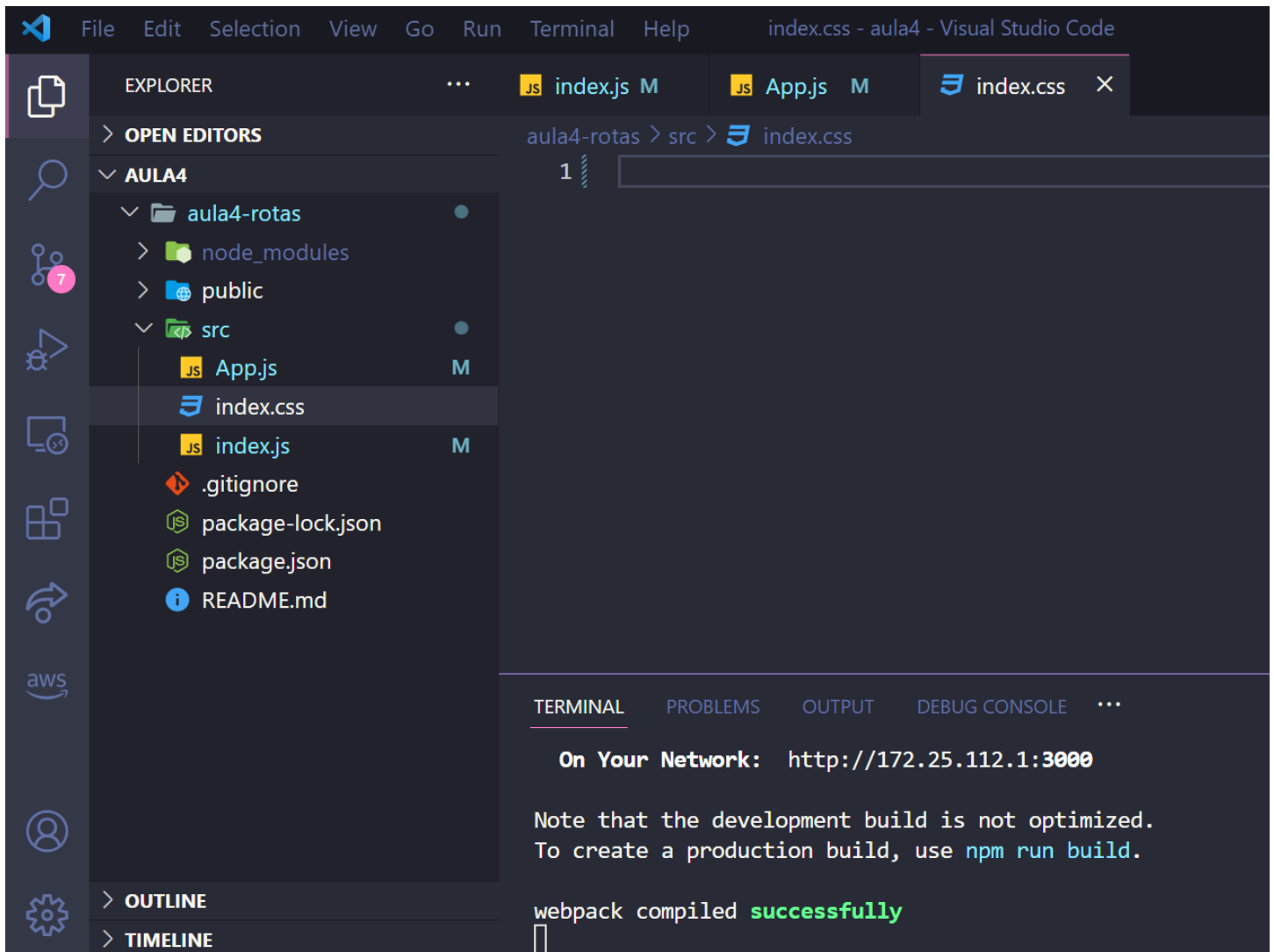


05 - Após iniciar o projeto, no navegador ele deve estar assim:



TESTE PROJETO

06 – Vamos fazer nosso index.css do zero. Limpe todo o conteúdo do arquivo



07 – Vamos fazer a configuração padrão de margin e espaçamento utilizando o * para todas as tags. Também mudar a font do body.

FileEditSelectionViewGoRunTerminalHelpindex.css - aula4 - Visual Studio Code

EXPLORER

OPEN EDITORS

AULA4

- aula4-rotas
 - node_modules
 - public
 - src
 - App.jsM
 - index.cssM
 - index.jsM
 - .gitignore
 - package-lock.json
 - package.json
 - README.md

OUTLINE

TIMELINE

index.jsM

App.jsM

index.cssM

aula4-rotas > src > index.css > body

```
1  *{
2    margin: 0;
3    padding: 0;
4    box-sizing: border-box;
5  }
6
7  body{
8    font-family: sans-serif;
9  }
```

TERMINAL

PROBLEMS

OUTPUT

DEBUG CONSOLE

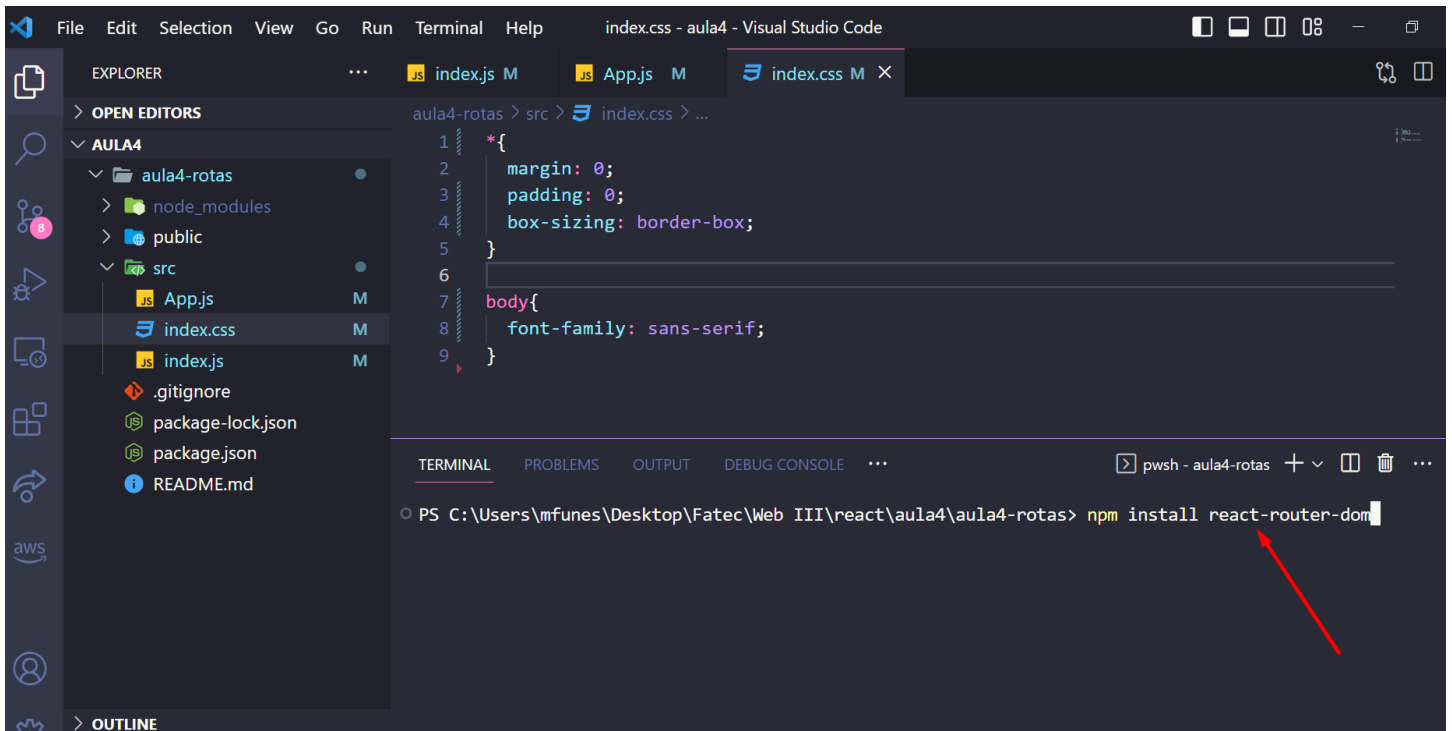
...

On Your Network: http://172.25.112.1:3000

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

webpack compiled **successfully**

08 – Para trabalhar com rotas, precisamos instalar uma biblioteca do React. Pare a execução do projeto (CTRL + C) e depois instale o react-router-dom

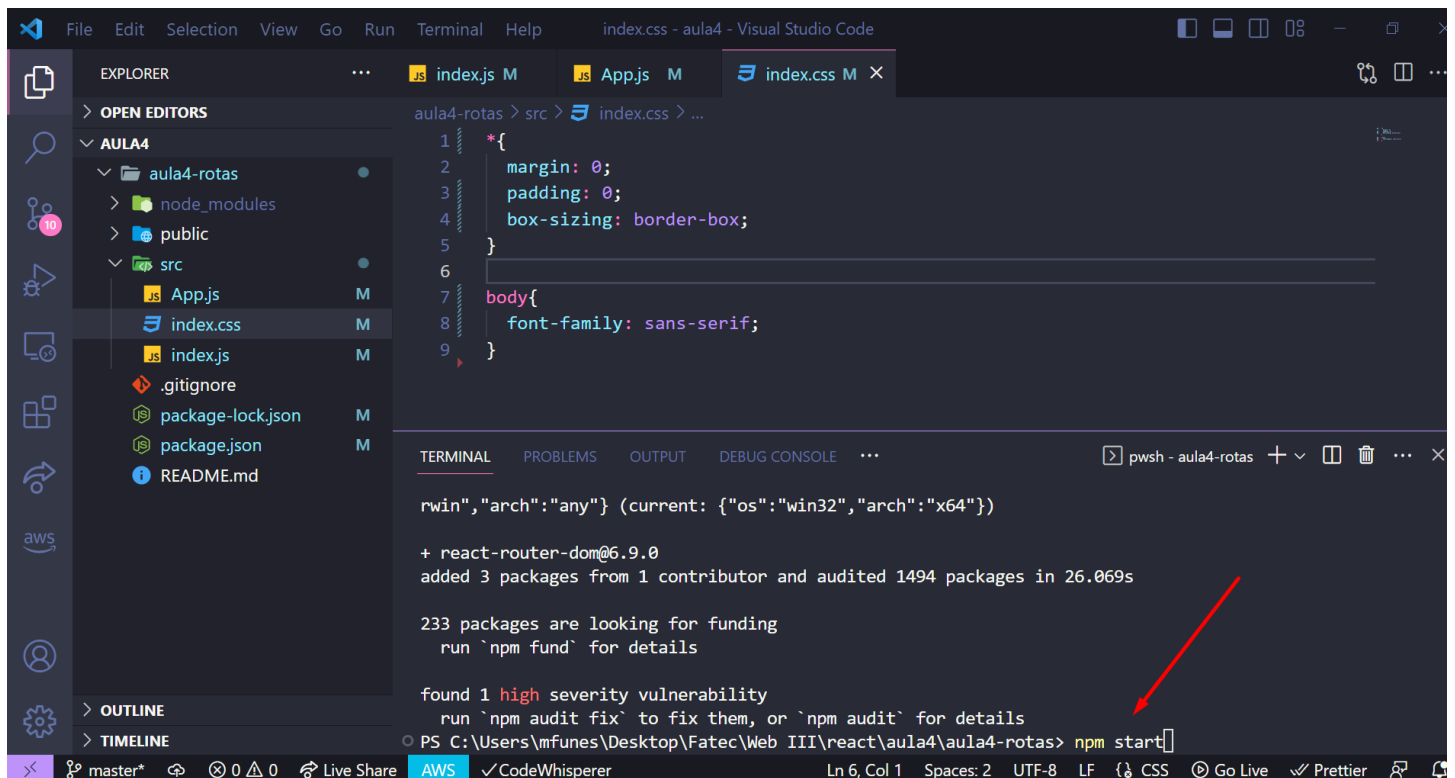


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying the project structure. The main editor area shows the `index.css` file with the following CSS code:

```
1  *{
2    margin: 0;
3    padding: 0;
4    box-sizing: border-box;
5  }
6
7  body{
8    font-family: sans-serif;
9  }
```

The bottom panel shows the Terminal with the command `npm install react-router-dom` being executed. A red arrow points to the `react-router-dom` text in the command. The terminal prompt indicates the current directory is `C:\Users\mfunes\Desktop\Fatec\Web III\react\aula4\aula4-rotas`.

09 – Após a instalação, execute o projeto novamente com npm start



The screenshot shows the Visual Studio Code interface with the Explorer view on the left and the Terminal view at the bottom. The Explorer view shows the project structure for 'aula4-rotas', including 'node_modules', 'public', and 'src'. The 'src' folder contains 'App.js', 'index.css', and 'index.js'. The 'index.css' file is open in the editor, showing a reset CSS and a body font-family. The Terminal view shows the output of the 'npm start' command, which includes the installation of 'react-router-dom@6.9.0' and a warning about a high severity vulnerability. A red arrow points to the 'npm start' command in the terminal.

```
1 *{
2   margin: 0;
3   padding: 0;
4   box-sizing: border-box;
5 }
6
7 body{
8   font-family: sans-serif;
9 }
```

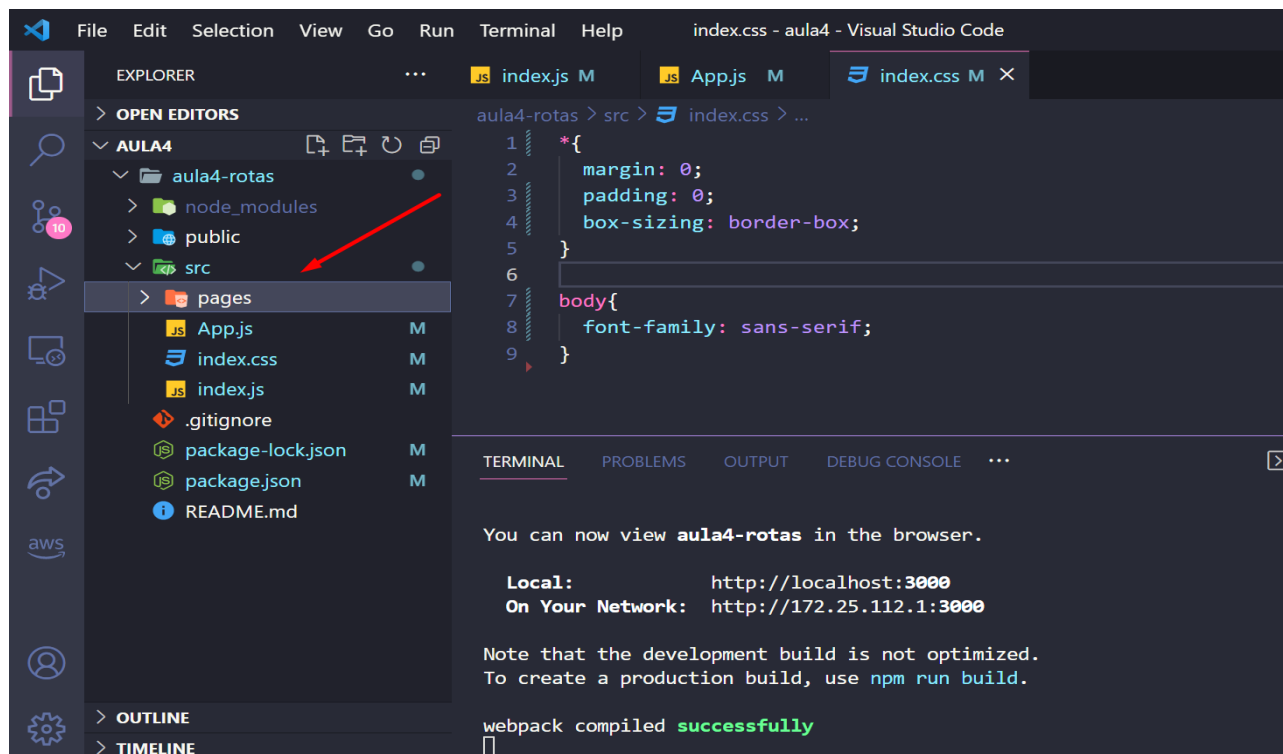
```
Terminal
rwin", "arch": "any"} (current: {"os": "win32", "arch": "x64"})

+ react-router-dom@6.9.0
added 3 packages from 1 contributor and audited 1494 packages in 26.069s

233 packages are looking for funding
  run `npm fund` for details

found 1 high severity vulnerability
run `npm audit fix` to fix them, or `npm audit` for details
PS C:\Users\mfunes\Desktop\Fatec\Web III\react\aula4\aula4-rotas> npm start
```

10 – Como qualquer site, nós queremos poder criar diversas páginas e permitir que o usuário navegue entre ela. Dê um botão direito na pasta SRC e crie uma nova pasta chamada pages.



The screenshot shows the Visual Studio Code interface with the Explorer view on the left and the Terminal view at the bottom. The Explorer view shows the project structure for 'aula4-rotas', including 'node_modules', 'public', and 'src'. The 'src' folder now contains a new 'pages' folder, 'App.js', 'index.css', and 'index.js'. The 'index.css' file is open in the editor, showing a reset CSS and a body font-family. The Terminal view shows the output of the 'npm start' command, which includes the installation of 'react-router-dom@6.9.0' and a warning about a high severity vulnerability. A red arrow points to the 'pages' folder in the Explorer view.

```
1 *{
2   margin: 0;
3   padding: 0;
4   box-sizing: border-box;
5 }
6
7 body{
8   font-family: sans-serif;
9 }
```

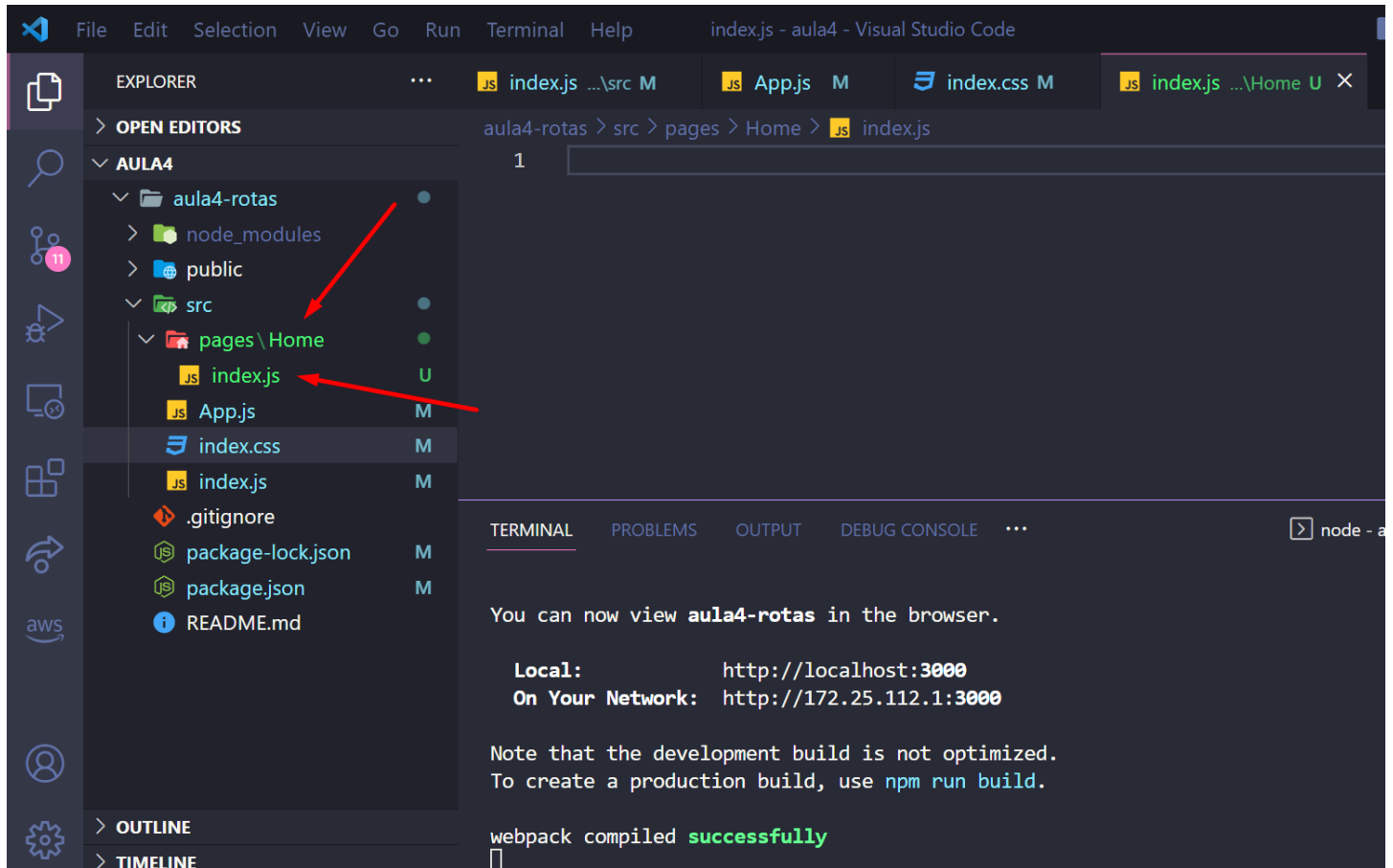
```
Terminal
You can now view aula4-rotas in the browser.

Local:      http://localhost:3000
On Your Network: http://172.25.112.1:3000

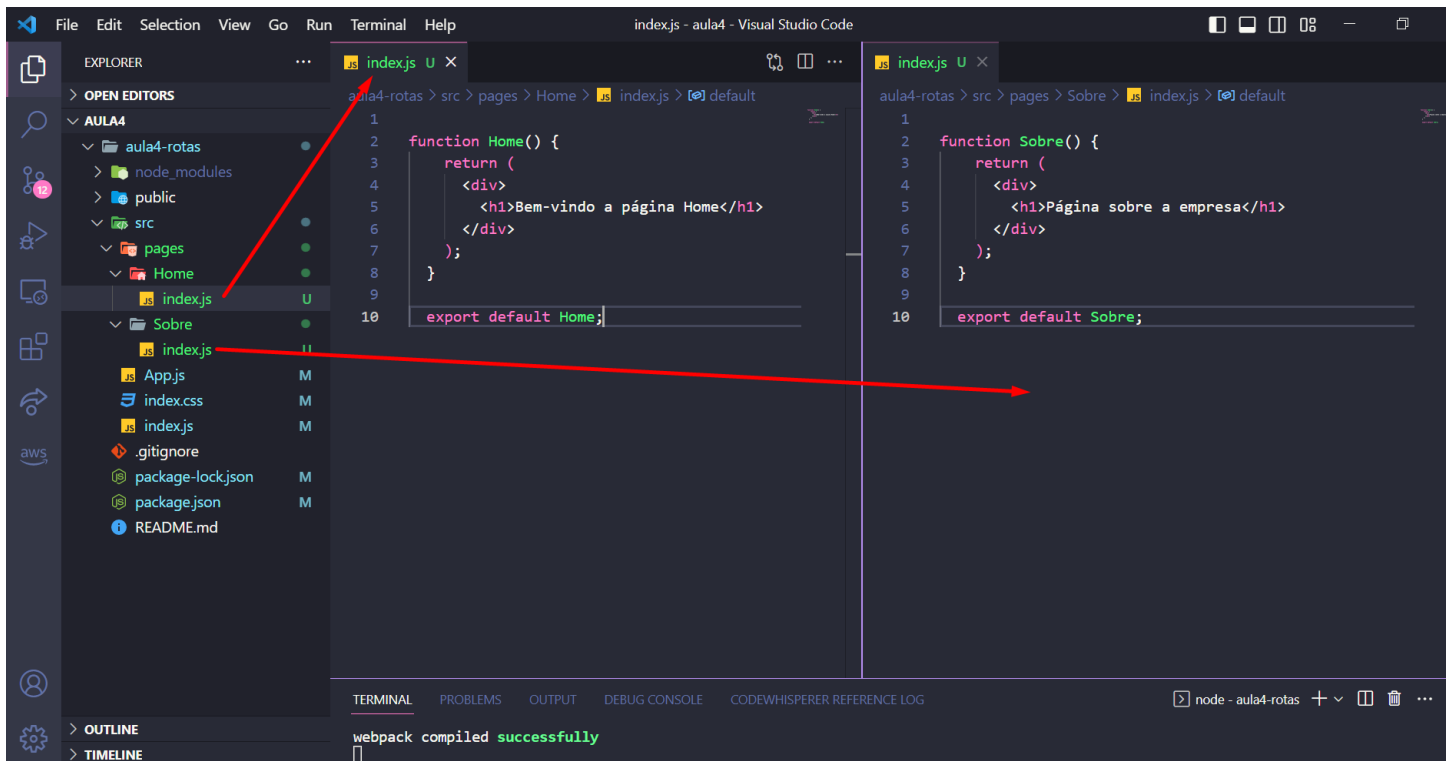
Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

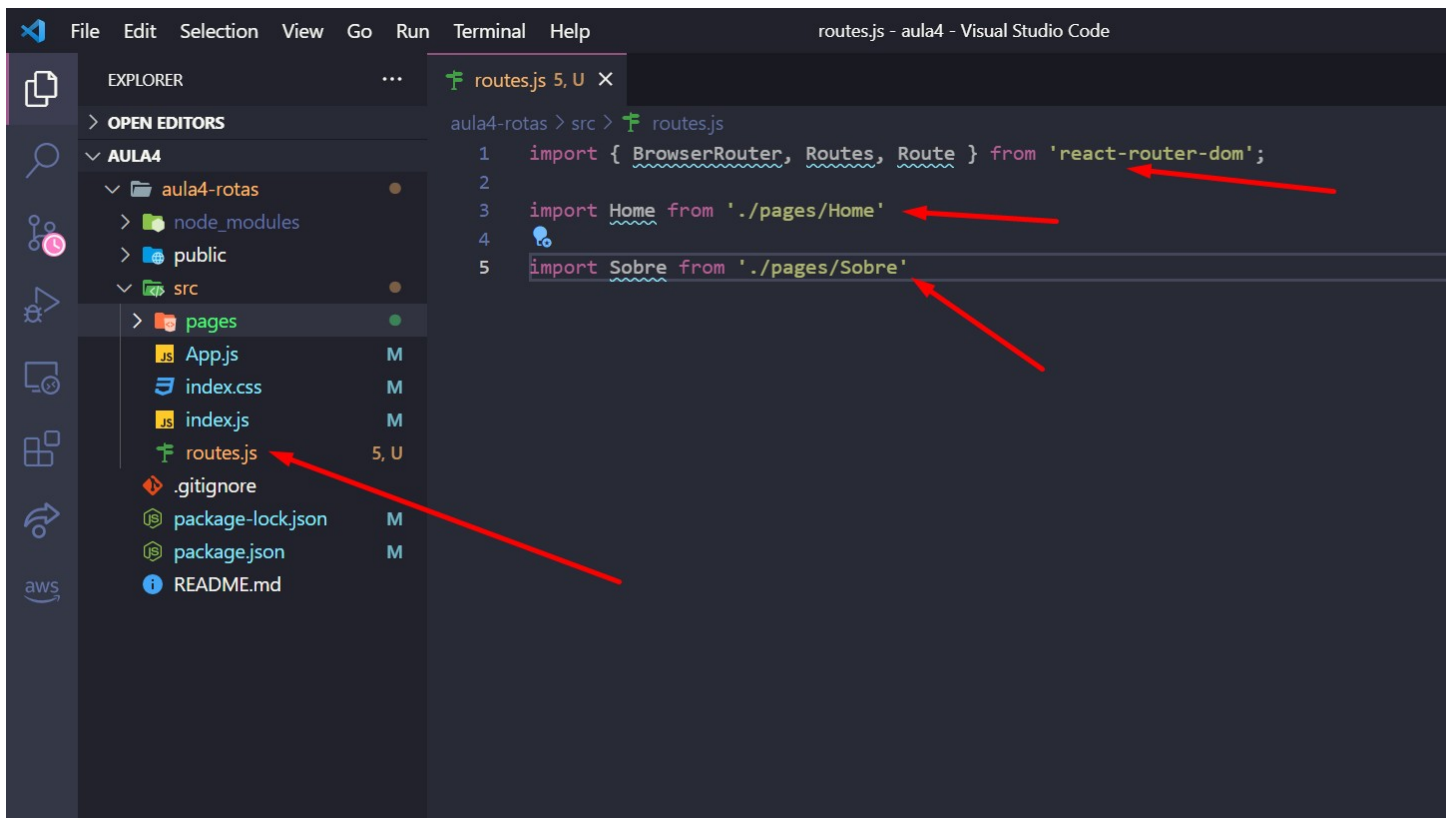
11 – Dentro de pages crie outra pasta chamada Home (utilize o H maiúsculo pois todo componente tem a primeira letra maiúscula). Ele ficará parecendo que não tem uma pasta dentro da outra, isso é normal, ficará assim até criarmos outra pasta.



12 – Crie outra pasta chamada Sobre e dentro dela um outro arquivo index.js. Também mude esse index conforme abaixo é demonstrado:



13 – Crie um arquivo chamado router.js dentro de SRC e dentro dele vamos configurar nossas rotas. Faça a importação dos componentes básicos do react-router-dom. Vamos importar o Home e Sobre, veja que não precisamos colocar os arquivos index.js, basta chamar a pasta que ele automaticamente irá chamar o index.js, apenas não coloque '/' no final do import.



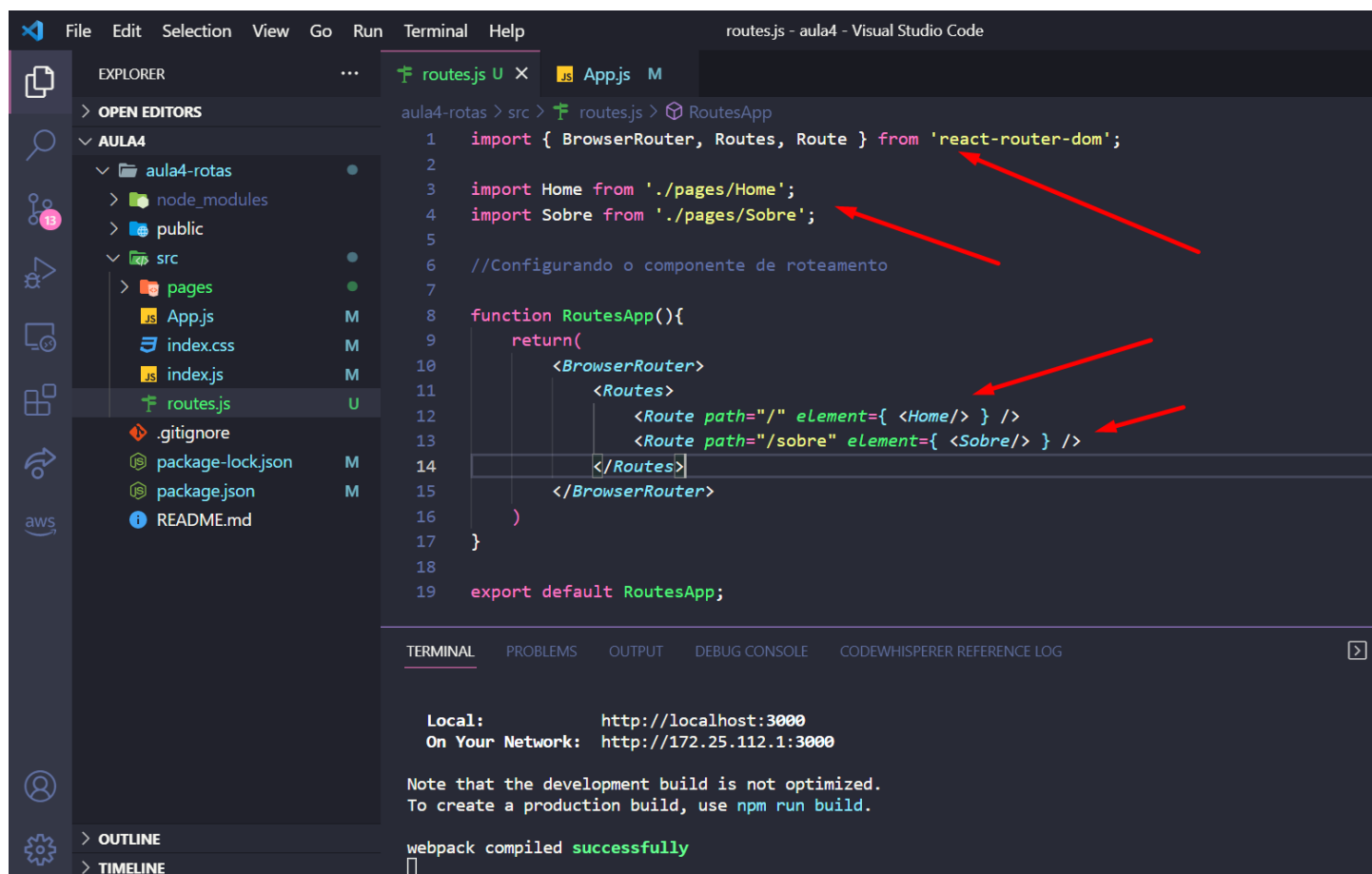
The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Editor window on the right. The Explorer sidebar shows the project structure for 'aula4', with the 'pages' folder selected under 'src'. The Editor window shows the content of 'routes.js' with the following code:

```
1 import { BrowserRouter, Routes, Route } from 'react-router-dom';  
2  
3 import Home from './pages/Home'  
4  
5 import Sobre from './pages/Sobre'
```

Red arrows point to the following elements:

- The 'pages' folder in the Explorer sidebar.
- The 'react-router-dom' package in the first import statement.
- The './pages/Home' path in the second import statement.
- The './pages/Sobre' path in the fifth import statement.

14 – Agora vamos de fato configurar as rotas, isso vai permitir que ao usuário clicar em algum link ou digitar o endereço na barra de navegação o React saiba pra onde levar o usuário através da rota correta. Veja que o Home colocamos apenas no path '/' pois queremos que Home sempre seja exibida por padrão.



```
File Edit Selection View Go Run Terminal Help
routes.js - aula4 - Visual Studio Code

EXPLORER
> OPEN EDITORS
  aula4-rotas
  > node_modules
  > public
  > src
    > pages
      App.js M
      index.css M
      index.js M
      routes.js U
    .gitignore
    package-lock.json M
    package.json M
    README.md

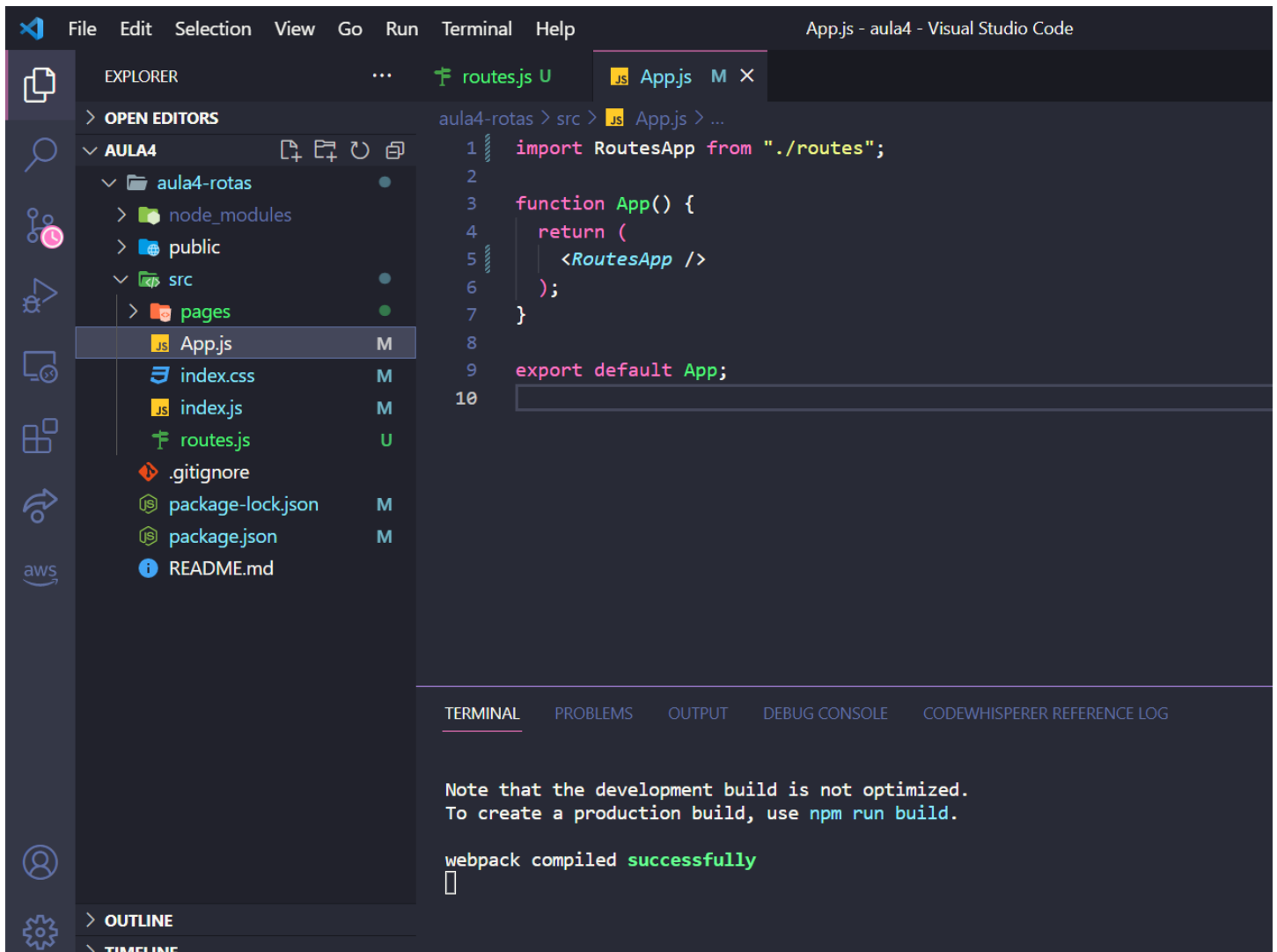
routes.js
1 import { BrowserRouter, Routes, Route } from 'react-router-dom';
2
3 import Home from './pages/Home';
4 import Sobre from './pages/Sobre';
5
6 //Configurando o componente de roteamento
7
8 function RoutesApp(){
9   return(
10     <BrowserRouter>
11       <Routes>
12         <Route path="/" element={ <Home/> } />
13         <Route path="/sobre" element={ <Sobre/> } />
14       </Routes>
15     </BrowserRouter>
16   )
17 }
18
19 export default RoutesApp;

TERMINAL
Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

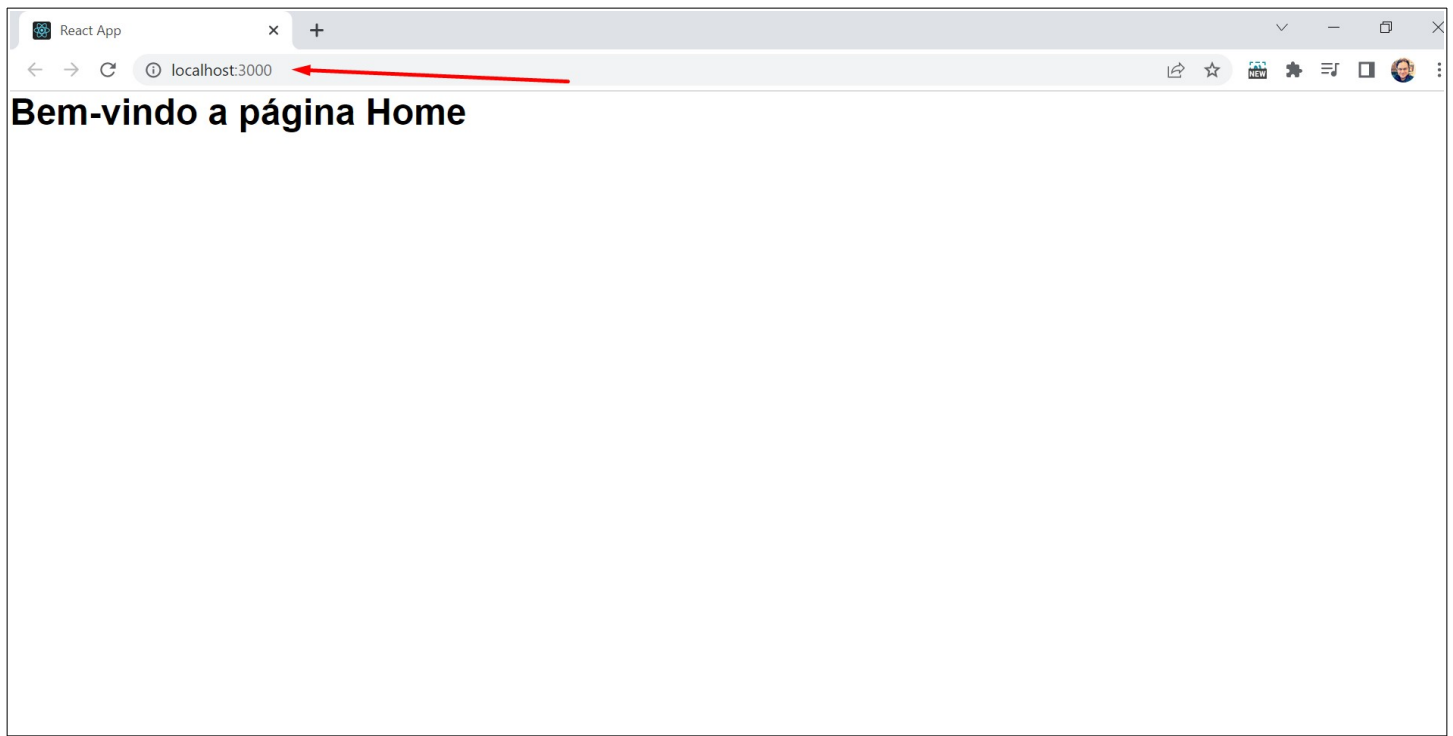
Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

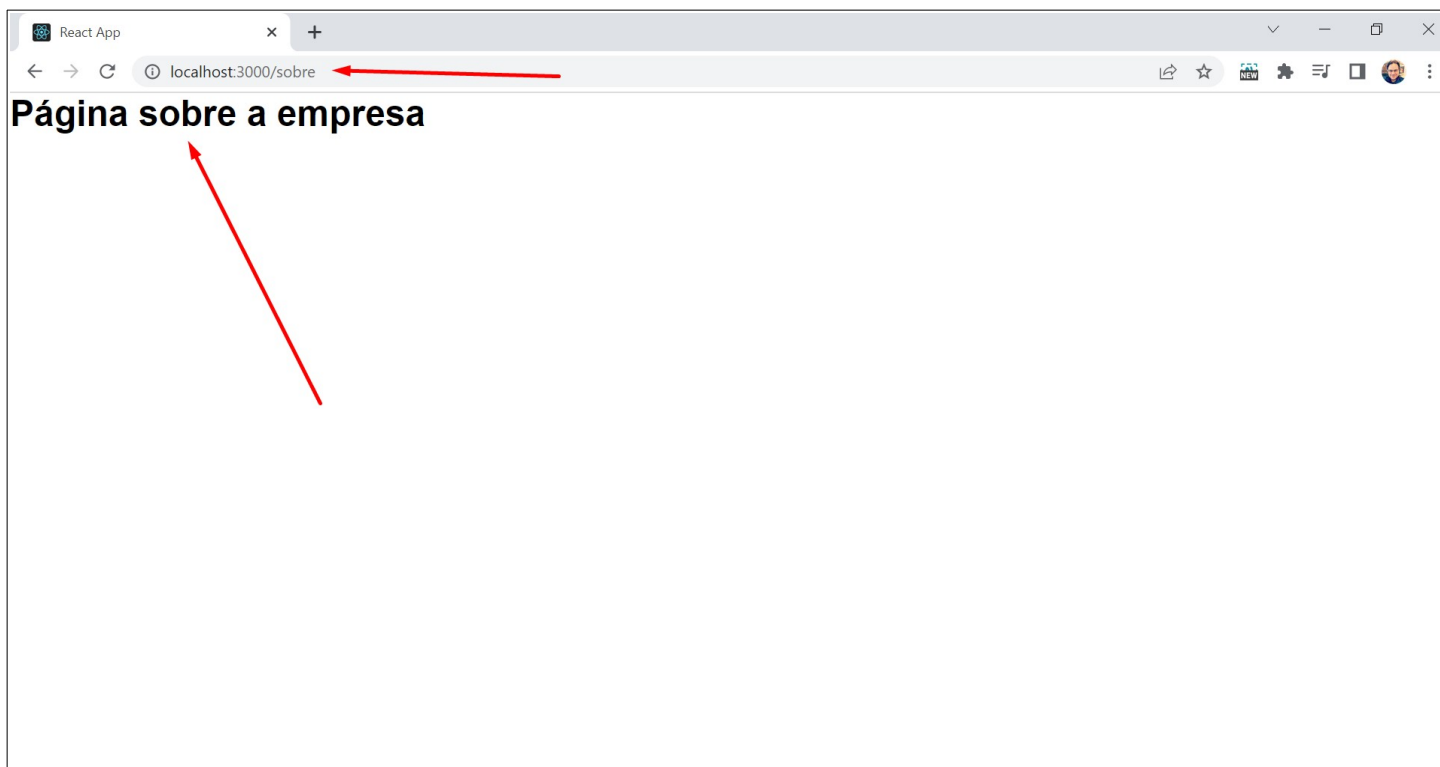
15 – Para funcionar esse roteamento, precisamos chama-lo para ser executado no App.js, ele é o primeiro componente que é renderizado ao entrar no site. Importamos o RoutesApp criado anteriormente, e queremos que ele renderize o componente RoutesApp ao usuário acessar o site da nossa aplicação.



16 – Acesse a aplicação no navegador. Veja que automaticamente ele já entrou em Home, nem precisamos passar /Home no final da página para isso.

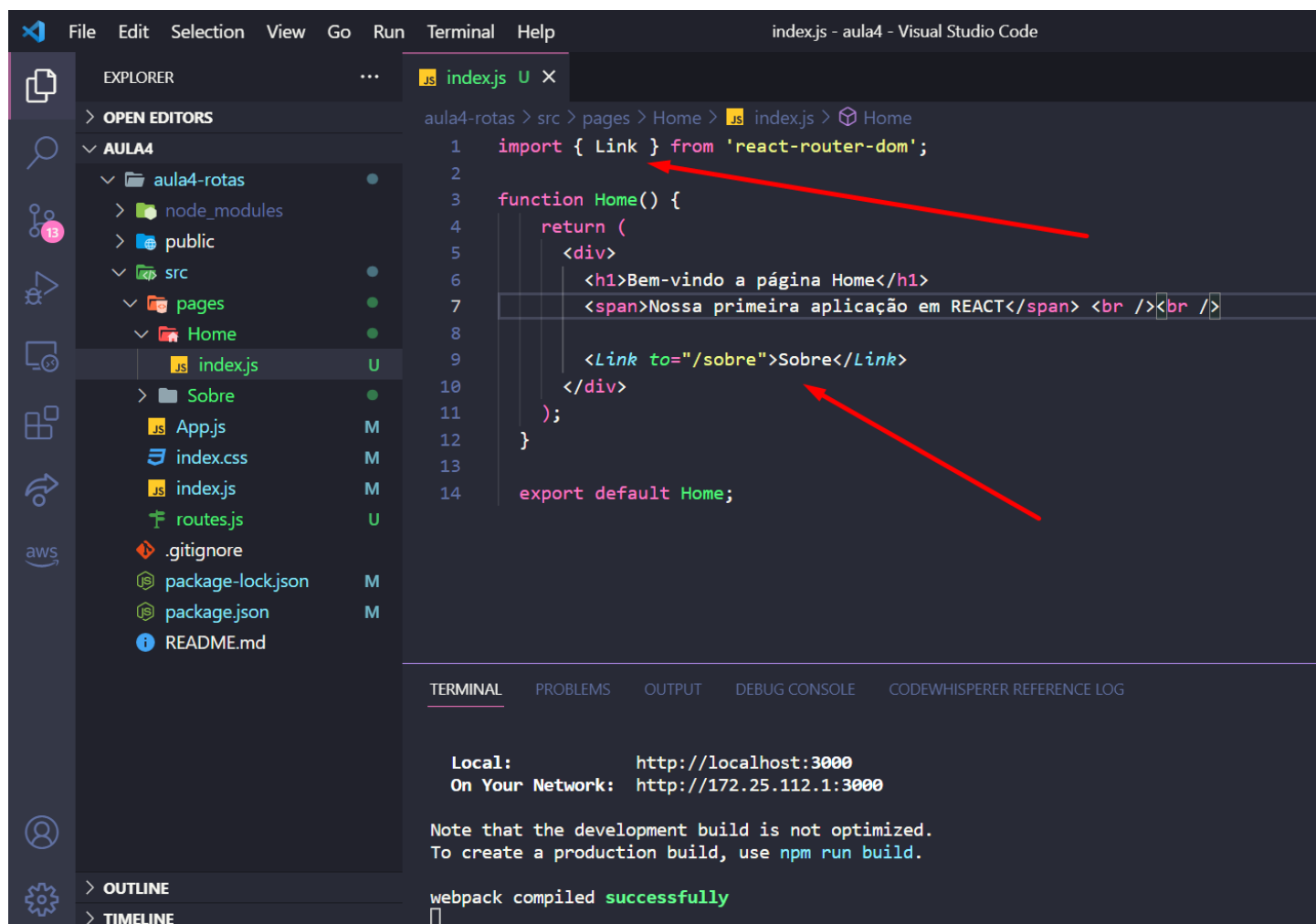


17 – Agora coloque /sobre no final do endereço. Ele irá para a rota que configuramos anteriormente.



Trabalhando com links internos

18 – Agora como podemos fazer com que um link interno também utilize nossas rotas. Vá até o arquivo index.js dentro da pasta Home, lá importe o Link pertencente ao react-router-dom. Dentro da nossa página coloque chame o <Link> e configure com o parâmetro to="/sobre" que ao clicar nesse link ele deve rotear o usuário para a página sobre.



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal panels. The Explorer panel on the left shows the project structure with the file `index.js` selected in the `Home` directory. The Editor panel displays the code for `index.js`, which imports `Link` from `react-router-dom` and renders a link to the `/sobre` route. The Terminal panel at the bottom shows the development server running on `http://localhost:3000` and a successful webpack compilation.

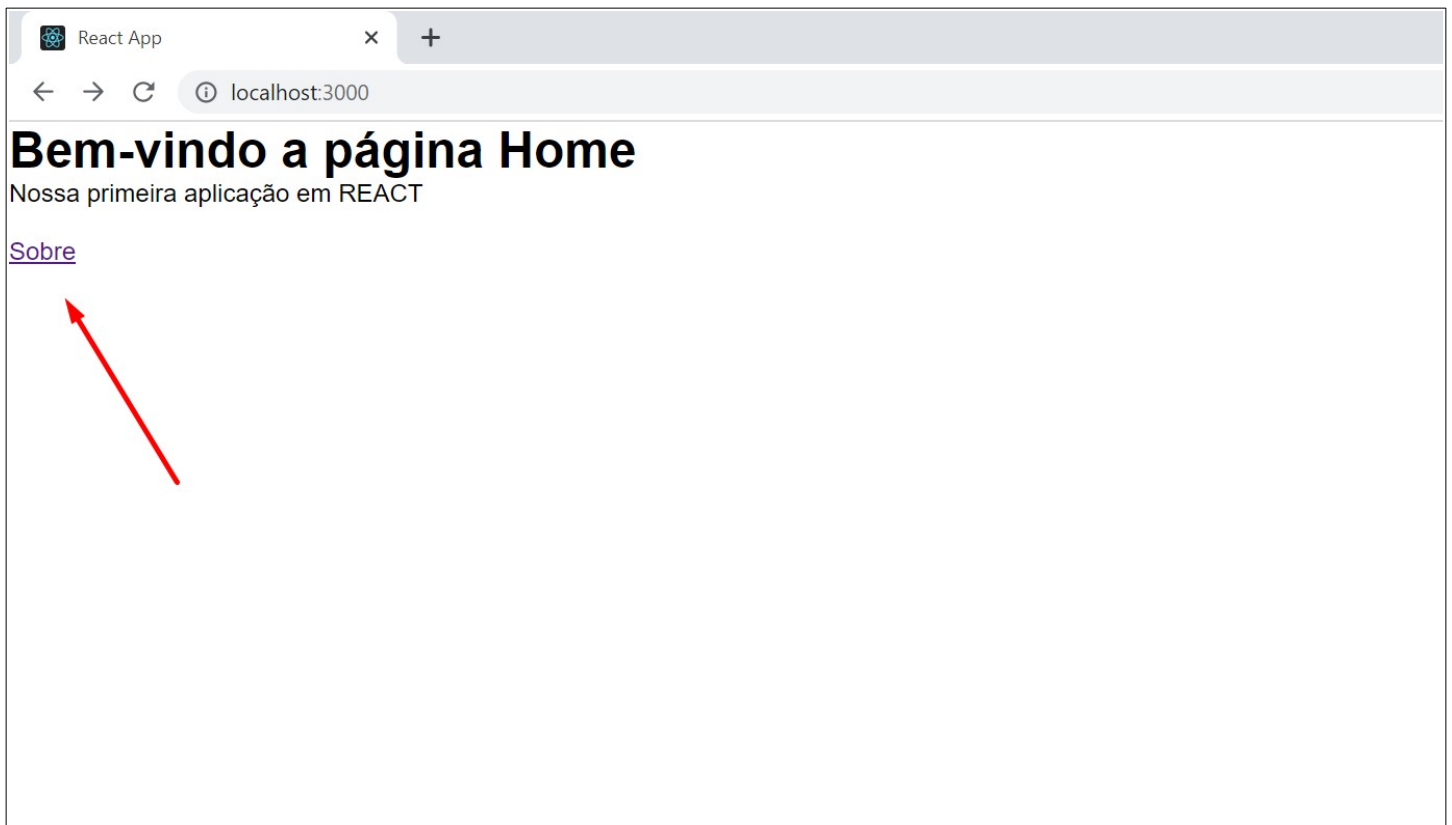
```
1 import { Link } from 'react-router-dom';
2
3 function Home() {
4   return (
5     <div>
6       <h1>Bem-vindo a página Home</h1>
7       <span>Nossa primeira aplicação em REACT</span> <br />
8       <Link to="/sobre">Sobre</Link>
9     </div>
10  );
11 }
12
13 export default Home;
```

Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

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

webpack compiled successfully

19 – Agora ao clicar no link ele deverá nos levar a página correta.



20 – Tente agora fazer com que o usuário estando na página sobre consiga voltar, por um link, a página home. (caso precise a resposta está na próxima página).

21 – Nosso index.js da página Sobre ficará dessa forma.

FileEditSelectionViewGoRunTerminalHelpindex.js - aula4 - Visual Studio Code

EXPLORER

> OPEN EDITORS

AULA4

aula4-rotas

node_modules

public

src

pages

Home

index.js

Sobre

index.js

App.js

index.css

index.js

routes.js

.gitignore

package-lock.json

package.json

README.md

index.js ...\Home U

index.js ...\Sobre U X

aula4-rotas > src > pages > Sobre > index.js > Sobre

1import { Link } from 'react-router-dom';

2

3function Sobre() {

4return (

5<div>

6<h1>Página sobre a empresa</h1>

7

8<Link to='/'>Página Home</Link>

9</div>

10);

11}

12

13export default Sobre;

TERMINAL

PROBLEMS

OUTPUT

DEBUG CONSOLE

CODEWHISPERER REFERENCE LOG

Local: http://localhost:3000

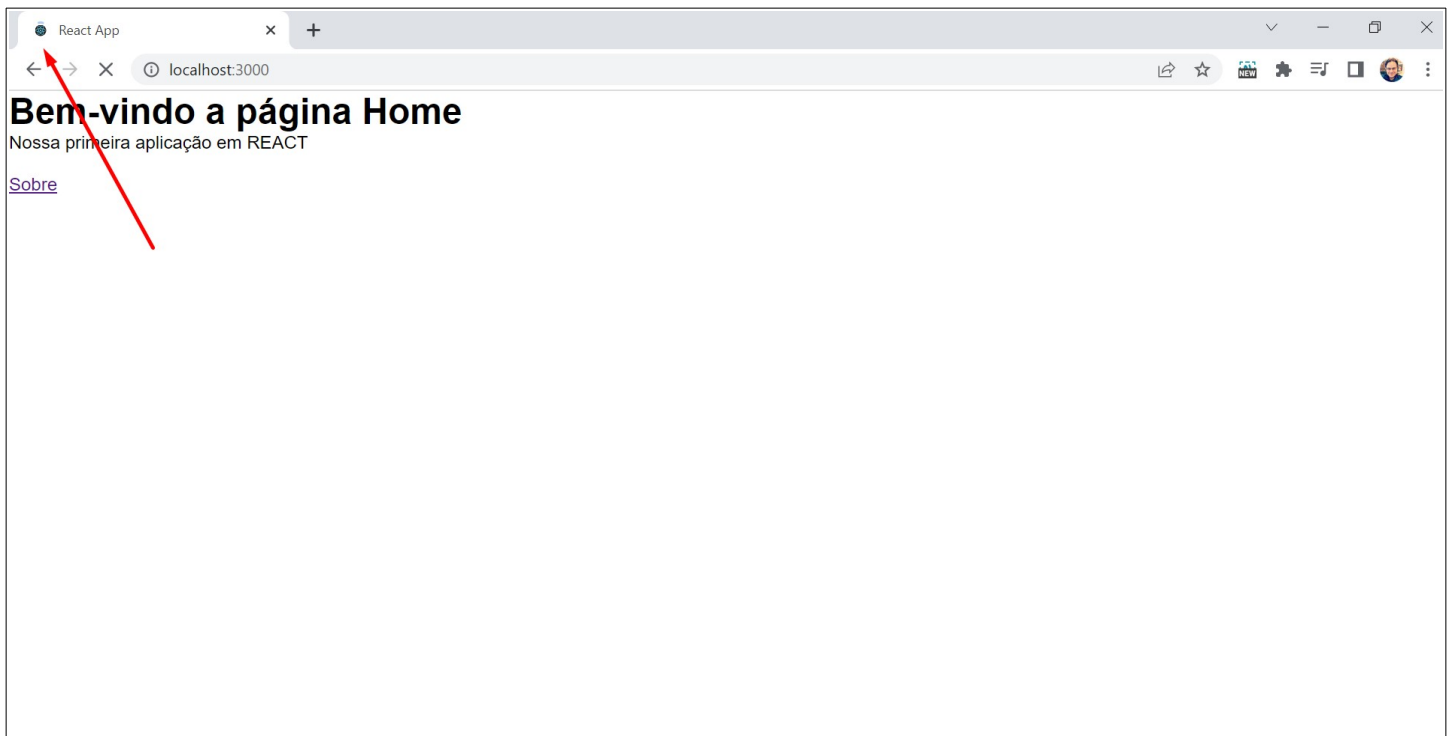
On Your Network: http://172.25.112.1:3000

Note that the development build is not optimized.

To create a production build, use `npm run build`.

webpack compiled successfully

22 – Vá até a página no navegador e dê um F5, veja que o icon da página ele recarrega, indicando que todos os componente da página foram atualizado. Agora clique nos links que criamos, veja que o icon não fica mais recarregando. Isso quer dizer que ao trocar de página nossa aplicação recarrega inteira como se tivesse acabado de atualizar. Isso dá mais performance pois a transição de páginas é instantânea.

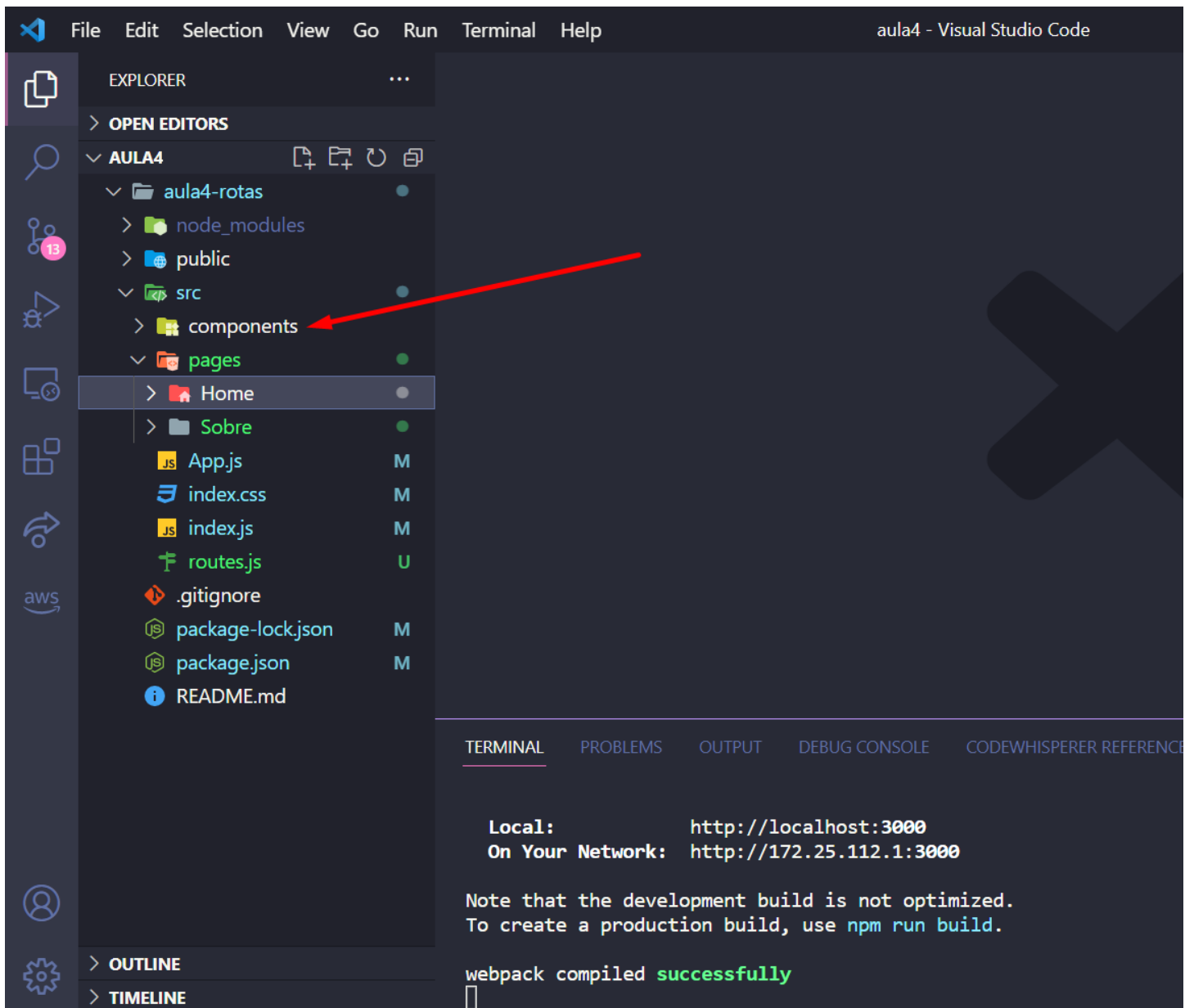


(opcional) Caso queira fixar melhor o conhecimento visto até aqui, crie uma nova página chamada contato e faça com que seja possível acessar e voltar para essa nova página estando na página Home.

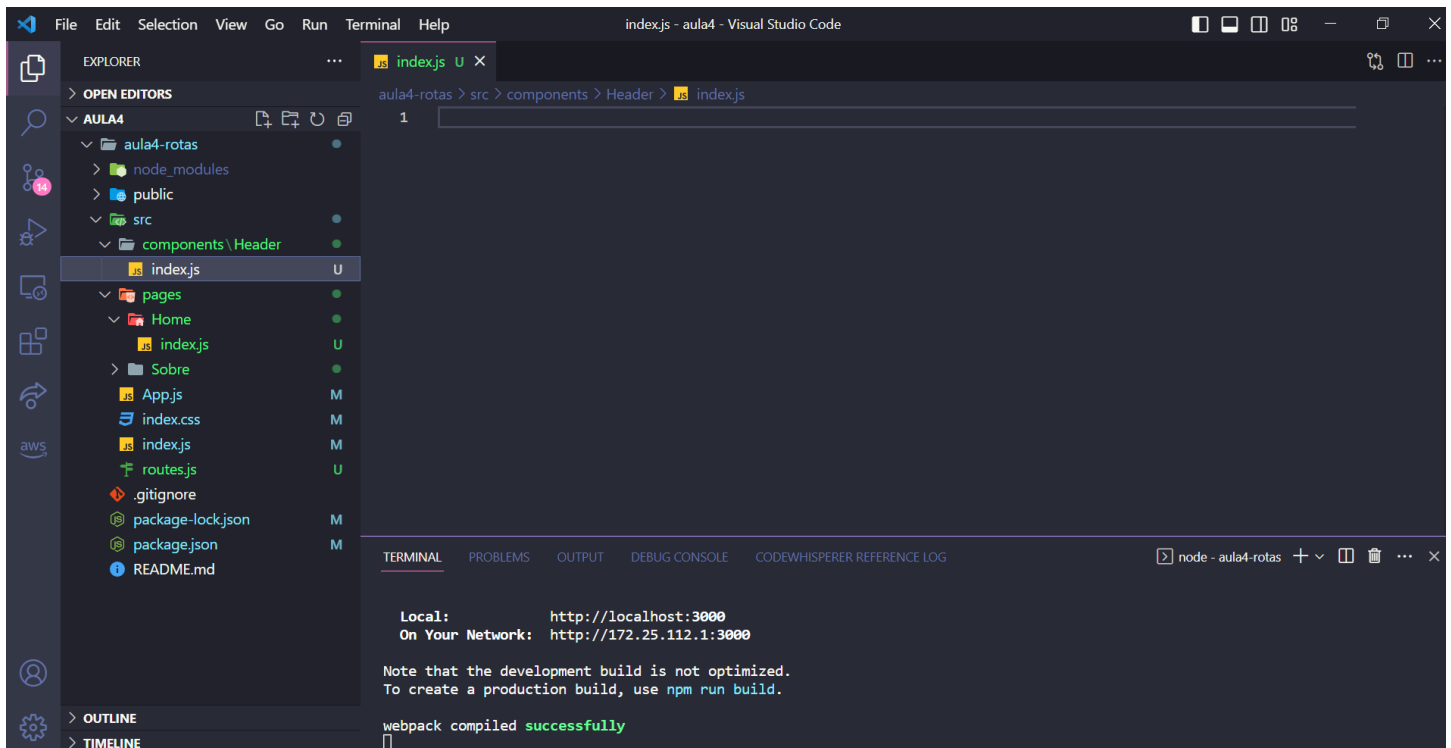
Header comum entre páginas

Em muitos casos um mesmo componente está presente em todas as páginas, um bom exemplo é o Header que pode ser o mesmo independente da página que estamos. Veremos agora como lidar com casos onde um mesmo componente não muda de página para página.

22 – Dentro de SRC crie uma nova pasta chamada **components**



23 – Dentro de components crie uma pasta chamada Header (componente tem como com primeira letra maiúscula) e dentro da pasta Header crie um index.js.



24 – Crie uma função chamada Header:

FileEditSelectionViewGoRunTerminalHelp

index.js - aula4 - Visual Studio Code

EXPLORER

OPEN EDITORS

AULA4

- aula4-rotas
 - node_modules
 - public
 - src
 - components\Header
 - index.jsU
 - pages
 - Home
 - index.jsU
 - Sobre
 - App.jsM
 - index.cssM
 - index.jsM
 - routes.jsU
 - .gitignore
 - package-lock.jsonM
 - package.jsonM
 - README.md

OUTLINE

TIMELINE

index.jsU

aula4-rotas > src > components > Header > index.js > Header

```
1
2 function Header(){
3   return(
4     <header>
5       <h2>Fatec - Franca 2023</h2>
6     </header>
7   )
8 }
9
10 export default Header;
```

TERMINAL

PROBLEMS

OUTPUT

DEBUG CONSOLE

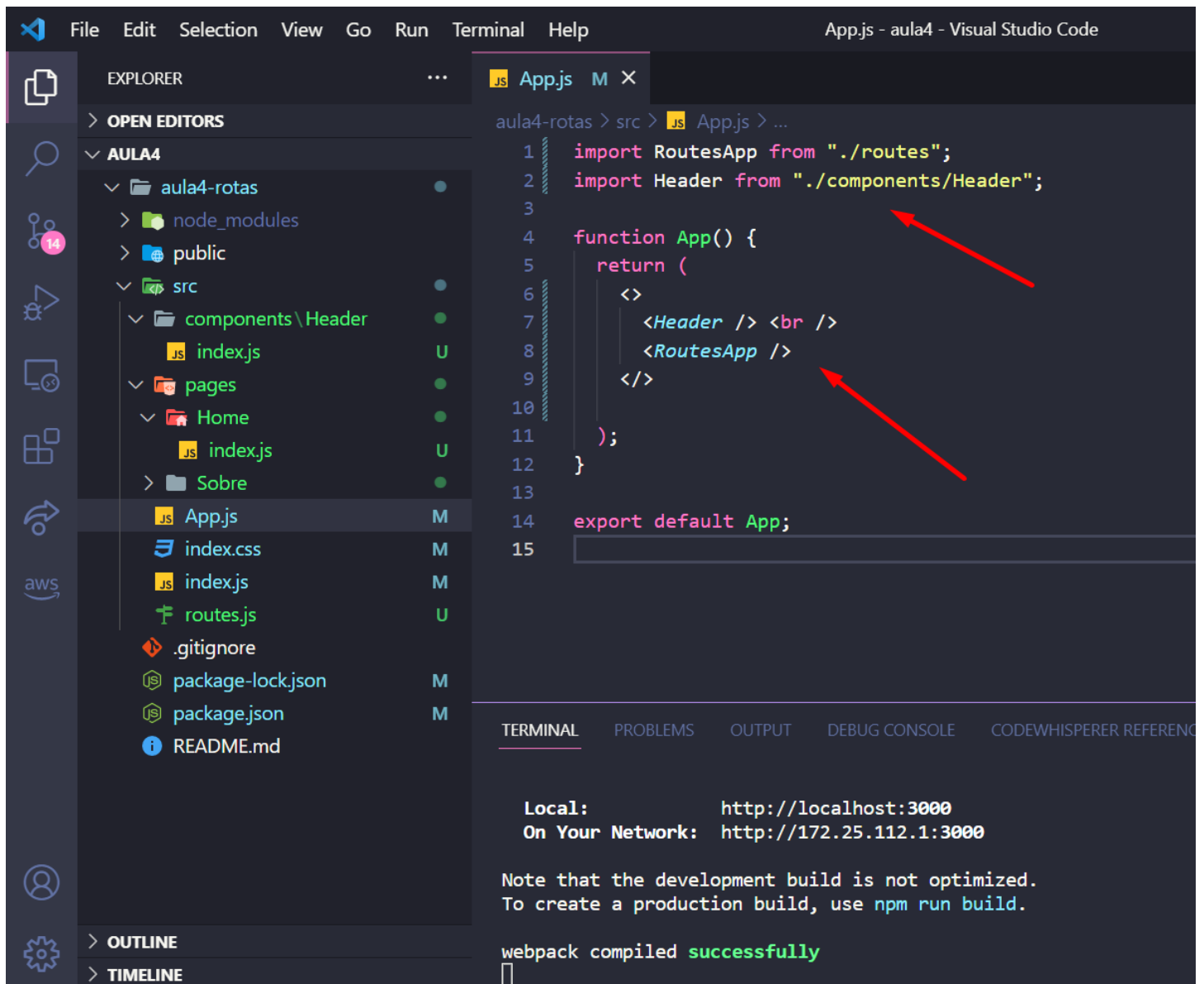
CODEWHISPERER: REFERENCE LOG

Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

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

webpack compiled **successfully**

25 – Como queremos que nosso Header fique fixo no topo do site independente da página, dentro de App.js (componente esse que renderiza nossa aplicação), faça o import do Header e dentro de App crie uma tag vazia `<> </>`, essa tag tem a mesma lógica de uma div, porém, sem afetar em nada dentro da nossa página. Agora chame primeiro o componente `<Header />` e depois `<RoutesApp />`.



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal panels. The Explorer panel on the left shows the project structure for 'aula4', with 'App.js' selected in the 'src' directory. The Editor panel in the center displays the code for 'App.js', which imports 'RoutesApp' and 'Header' from their respective files. The 'App' function returns a JSX element consisting of an empty tag, followed by the 'Header' component, and then the 'RoutesApp' component. Two red arrows point to the 'Header' and 'RoutesApp' components in the JSX. The Terminal panel at the bottom shows the development server running on localhost:3000 and a successful webpack compilation.

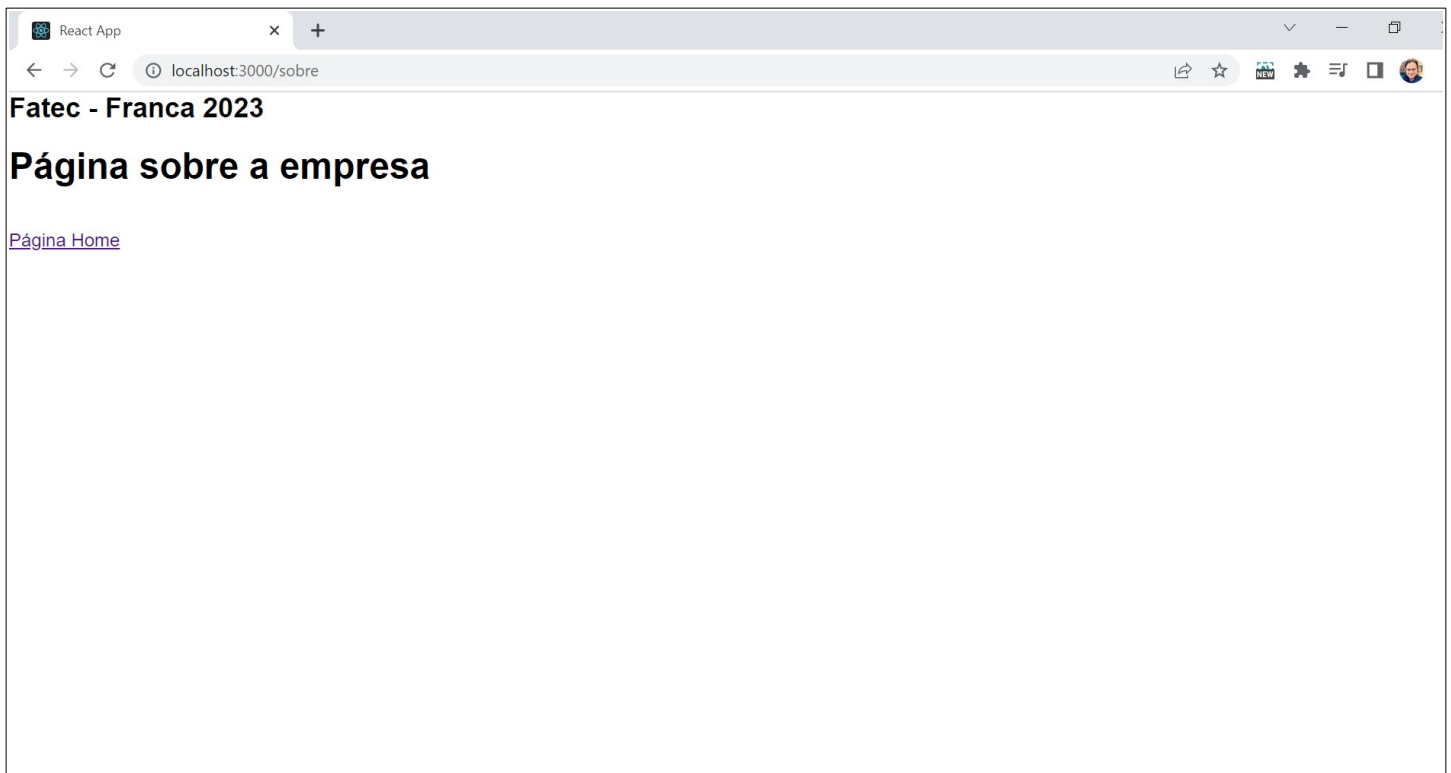
```
1 import RoutesApp from "../routes";
2 import Header from "../components/Header";
3
4 function App() {
5   return (
6     <>
7       <Header /> <br />
8       <RoutesApp />
9     </>
10  );
11 }
12
13 export default App;
```

Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

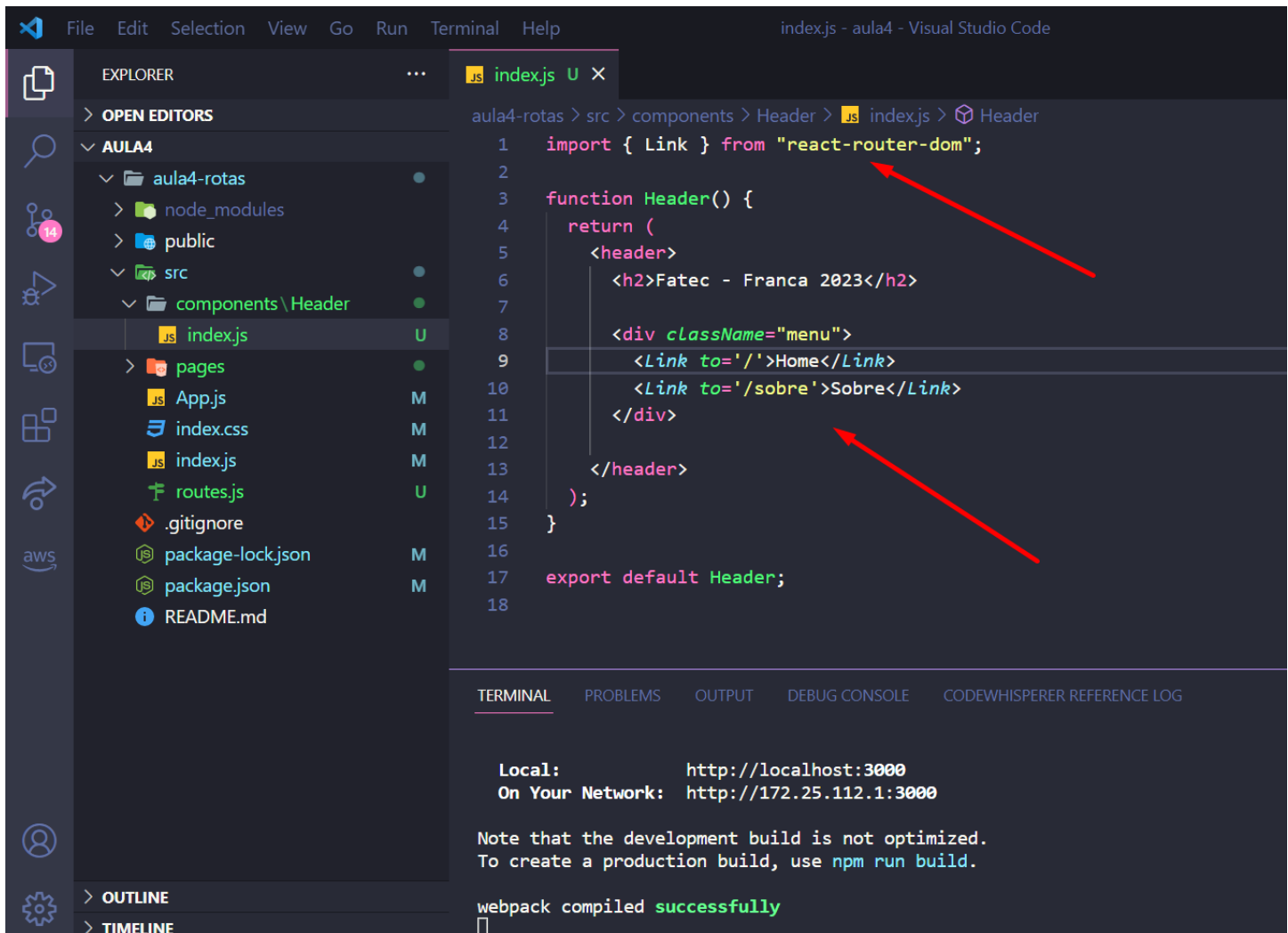
Note that the development build is not optimized.
To create a production build, use `npm run build`.

webpack compiled **successfully**

26 – Agora, no navegador, veja que nosso Header está fixo mesmo clicando nos links de mudança de páginas.



27 – Vamos agora fazer com que o links fiquem no nosso Header, como se fosse um menu. Vá até index.js do nosso Header e import o react-router-dom. Vamos criar uma nova div para colocar nosso menu, crie um className="menu" para depois estilizarmos esse menu. Dentro dessa div coloque os links para nossas páginas criadas Home e Sobre.



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal panels. The Explorer panel on the left shows the project structure with the file `index.js` selected in the `components/Header` directory. The Editor panel in the center displays the code for `index.js`, which imports `Link` from `react-router-dom` and defines a `Header` function. The function returns a JSX element containing a header with the text "Fatec - Franca 2023" and a `div` with the class `menu` containing two `Link` components for "Home" and "Sobre". Two red arrows point to the `react-router-dom` import and the `className="menu"` attribute. The Terminal panel at the bottom shows the local and network URLs for the development server and a message indicating that webpack compiled successfully.

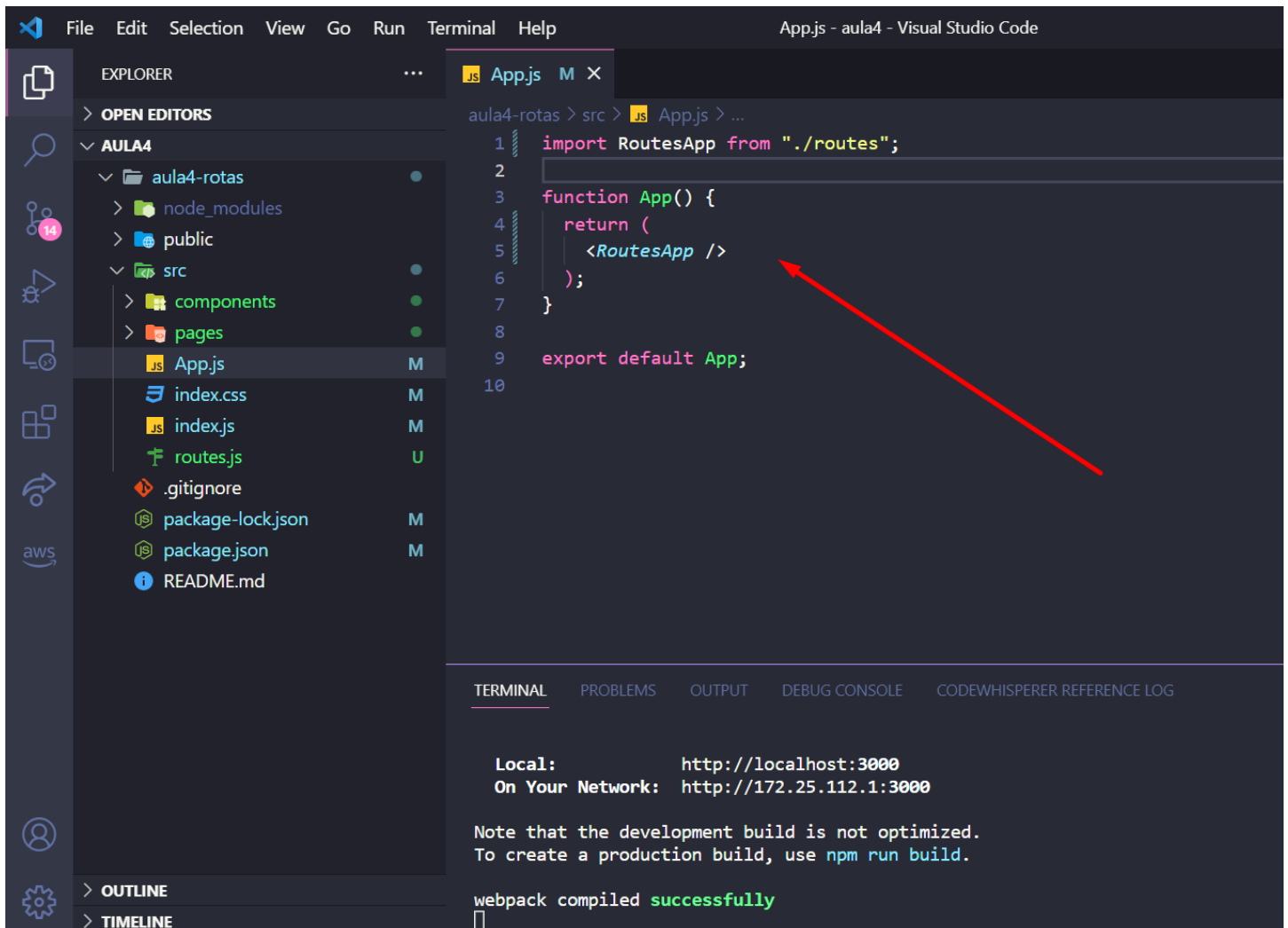
```
1 import { Link } from "react-router-dom";
2
3 function Header() {
4   return (
5     <header>
6       <h2>Fatec - Franca 2023</h2>
7
8       <div className="menu">
9         <Link to="/">Home</Link>
10        <Link to="/sobre">Sobre</Link>
11      </div>
12    </header>
13  );
14 }
15
16 export default Header;
```

Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

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

webpack compiled successfully

28 – Observe que nossa aplicação irá indicar erro no navegador, isso acontece devido nosso componente Header não conhecer nosso componente RoutesApp, ele não sabe como o comportamento das rotas deve acontecer. Para corrigir isso, vá até o App.js e retire o Header do app.



29 – Vá até router.js e vamos adicionar o Header diretamente em rotas:

Visual Studio Code interface showing the file explorer on the left, the editor in the center, and the terminal at the bottom.

File Explorer (Left):

- aula4-rotas
 - node_modules
 - public
 - src
 - components
 - pages
 - App.js
 - index.css
 - index.js
 - routes.js
 - .gitignore
 - package-lock.json
 - package.json
 - README.md

Editor (Center):

```
1 import { BrowserRouter, Routes, Route } from 'react-router-dom';
2
3 import Home from './pages/Home';
4 import Sobre from './pages/Sobre';
5
6 import Header from './components/Header';
7
8 //Configurando o componente de roteamento
9 function RoutesApp(){
10   return(
11     <BrowserRouter>
12       <Header />
13       <Routes>
14         <Route path="/" element={ <Home/> } />
15         <Route path="/sobre" element={ <Sobre/> } />
16       </Routes>
17     </BrowserRouter>
18   )
19 }
20
21 export default RoutesApp;
```

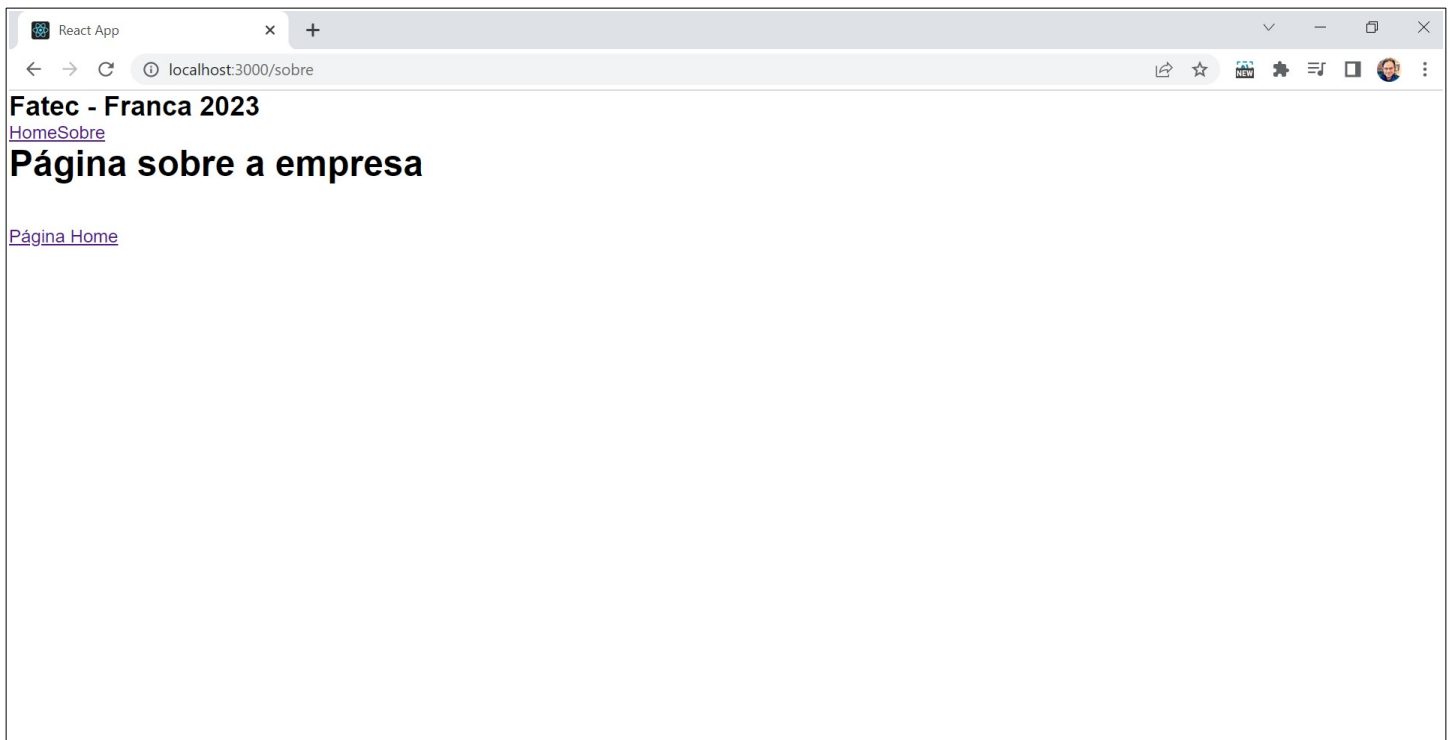
Terminal (Bottom):

```
On Your Network: http://172.25.112.1:3000

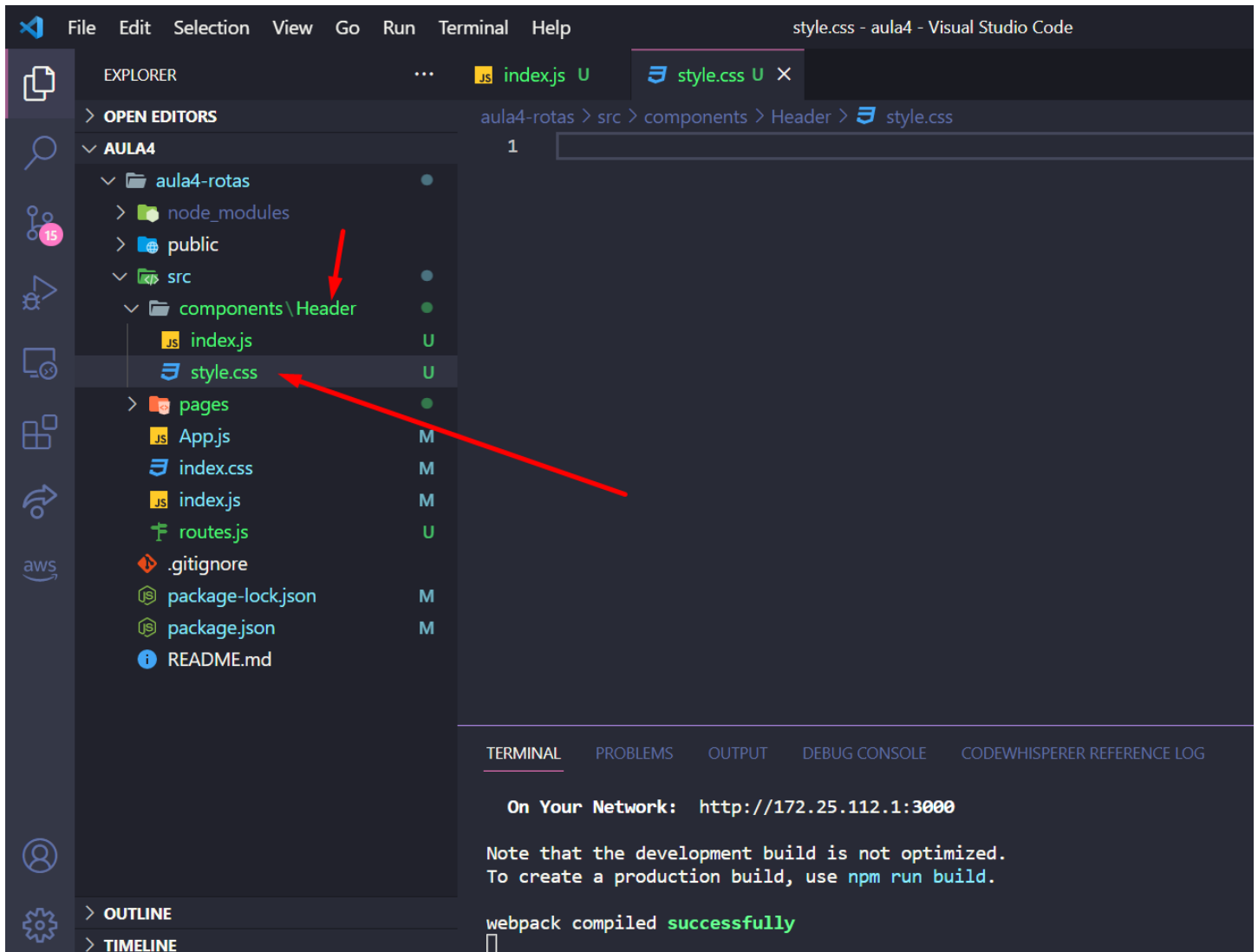
Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

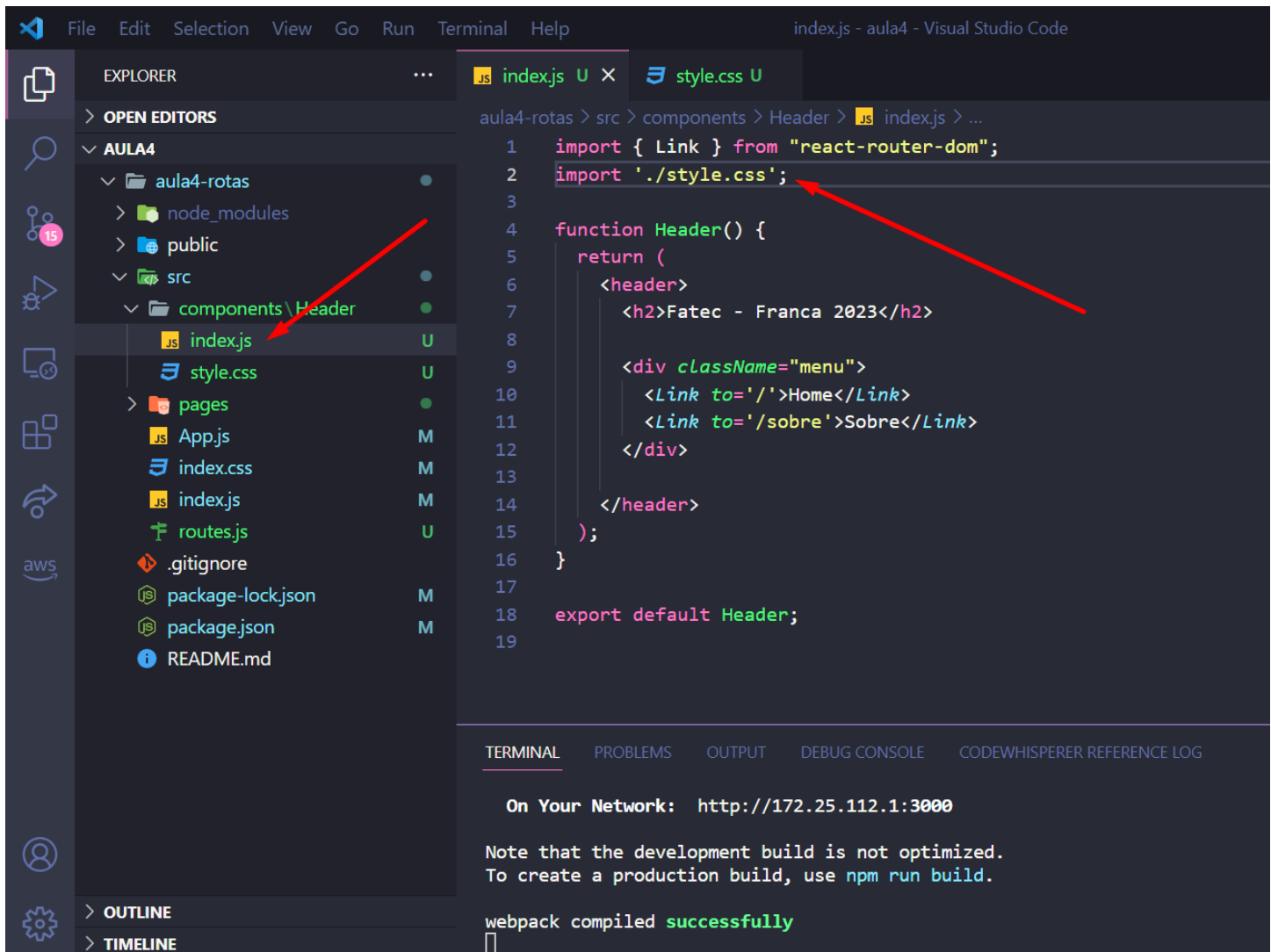
30 – Agora é possível utilizar os links dentro do nosso Header. Como estão grudados e bem feito, bora estilizar esse Header e Menu



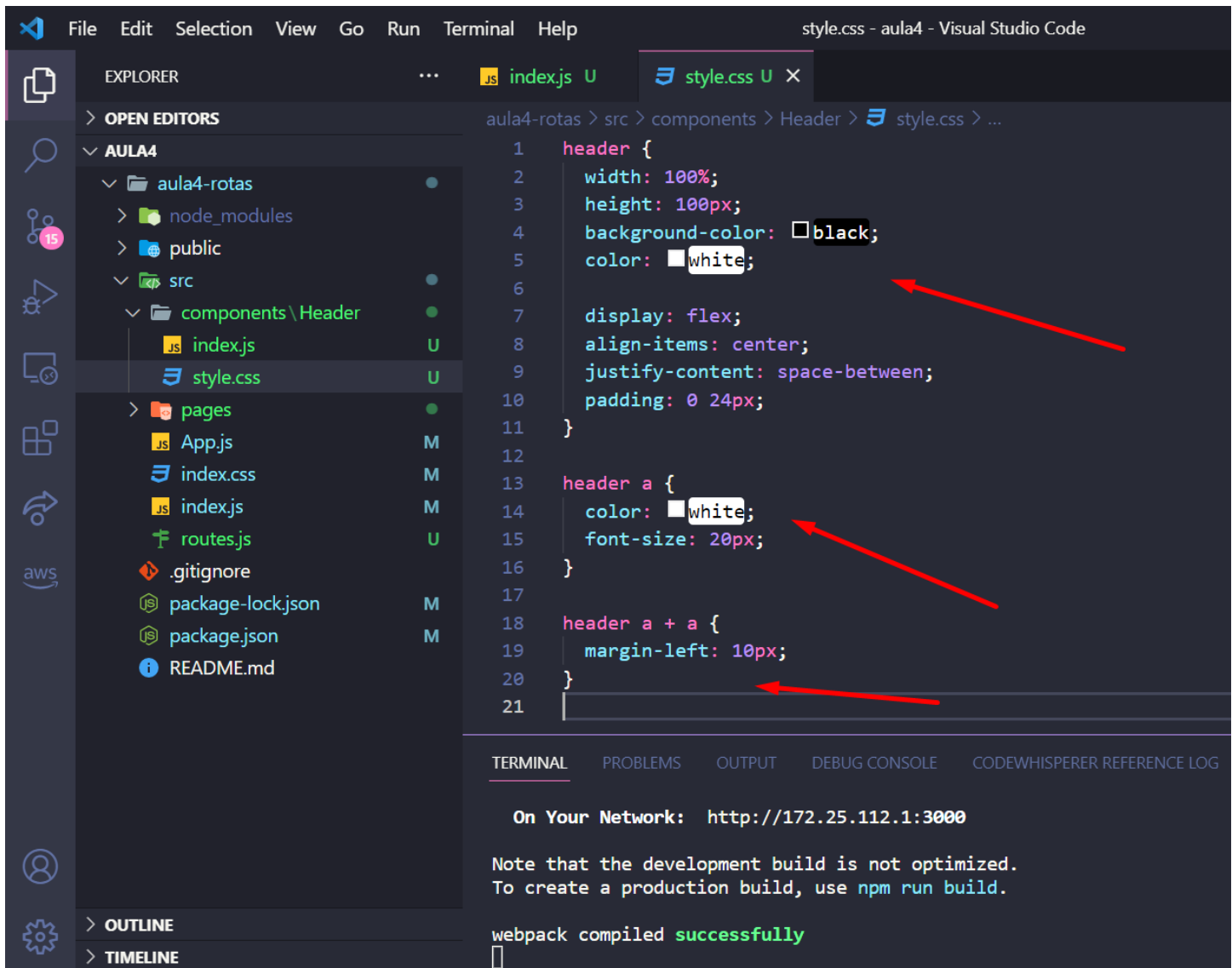
31 – Dentro da pasta Header, crie um arquivo chamado style.css:



32 – Importe o style.css e acabamos de criar dentro de index.js, dentro da pasta Header.



33 – Vamos fazer o seguinte estilo. Ao colocar header a estamos dizendo que queremos configurar a tag a, lembrando que no HTML <a> é a tag basica de um link. Ao colocar a + a dizemos que se a página tiver mais de uma tag a, ou seja, mais de um link queremos que a margem entre essa tags a seja de 10px.



The screenshot shows the Visual Studio Code interface with the Explorer, Editor, and Terminal panels. The Explorer panel on the left shows the project structure for 'aula4', with the 'components/Header' folder selected. The Editor panel in the center displays the 'style.css' file with the following CSS code:

```
1 header {
2   width: 100%;
3   height: 100px;
4   background-color: black;
5   color: white;
6
7   display: flex;
8   align-items: center;
9   justify-content: space-between;
10  padding: 0 24px;
11 }
12
13 header a {
14   color: white;
15   font-size: 20px;
16 }
17
18 header a + a {
19   margin-left: 10px;
20 }
21
```

Three red arrows point to specific lines of code: the first arrow points to line 4 ('background-color: black;'), the second arrow points to line 14 ('color: white;'), and the third arrow points to line 19 ('margin-left: 10px;').

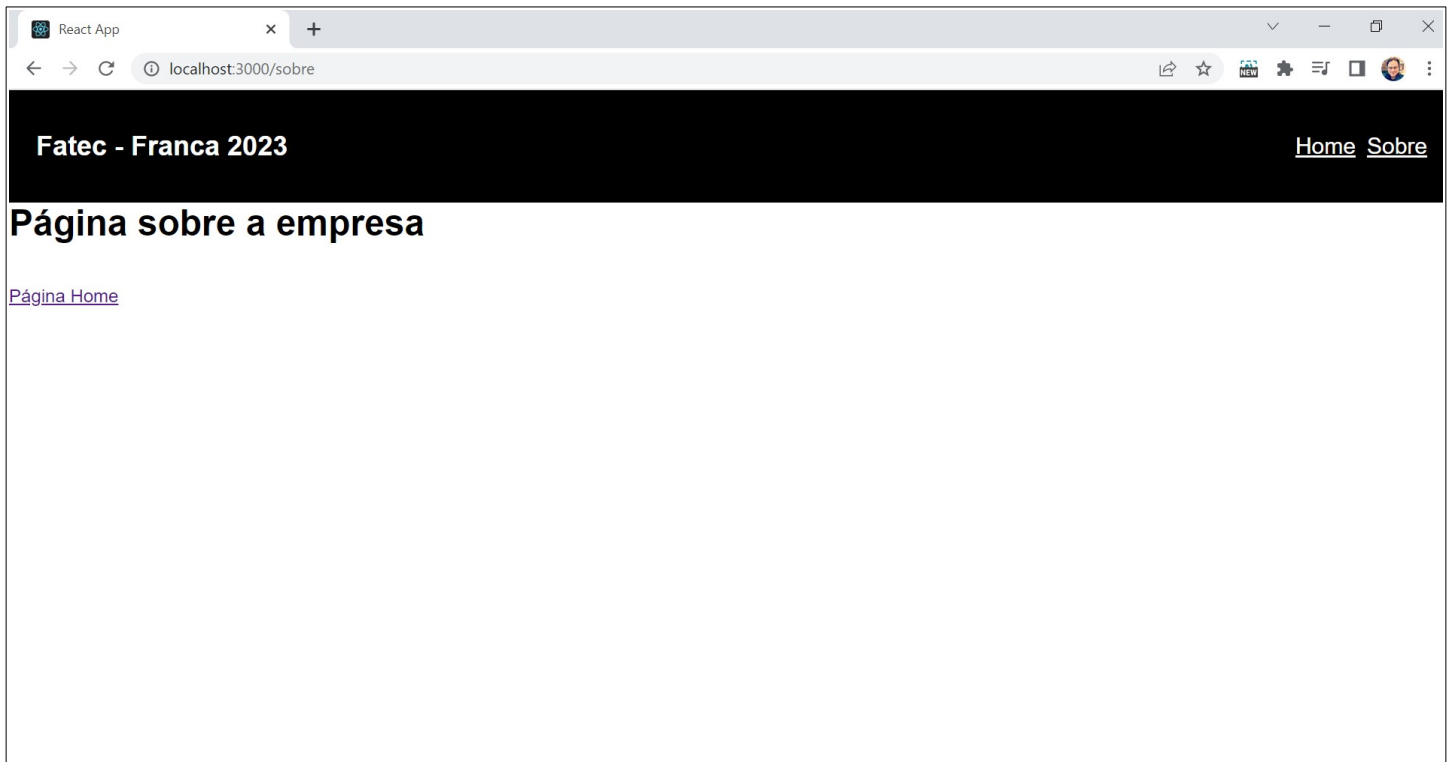
The Terminal panel at the bottom shows the following output:

```
On Your Network: http://172.25.112.1:3000

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

webpack compiled successfully
```

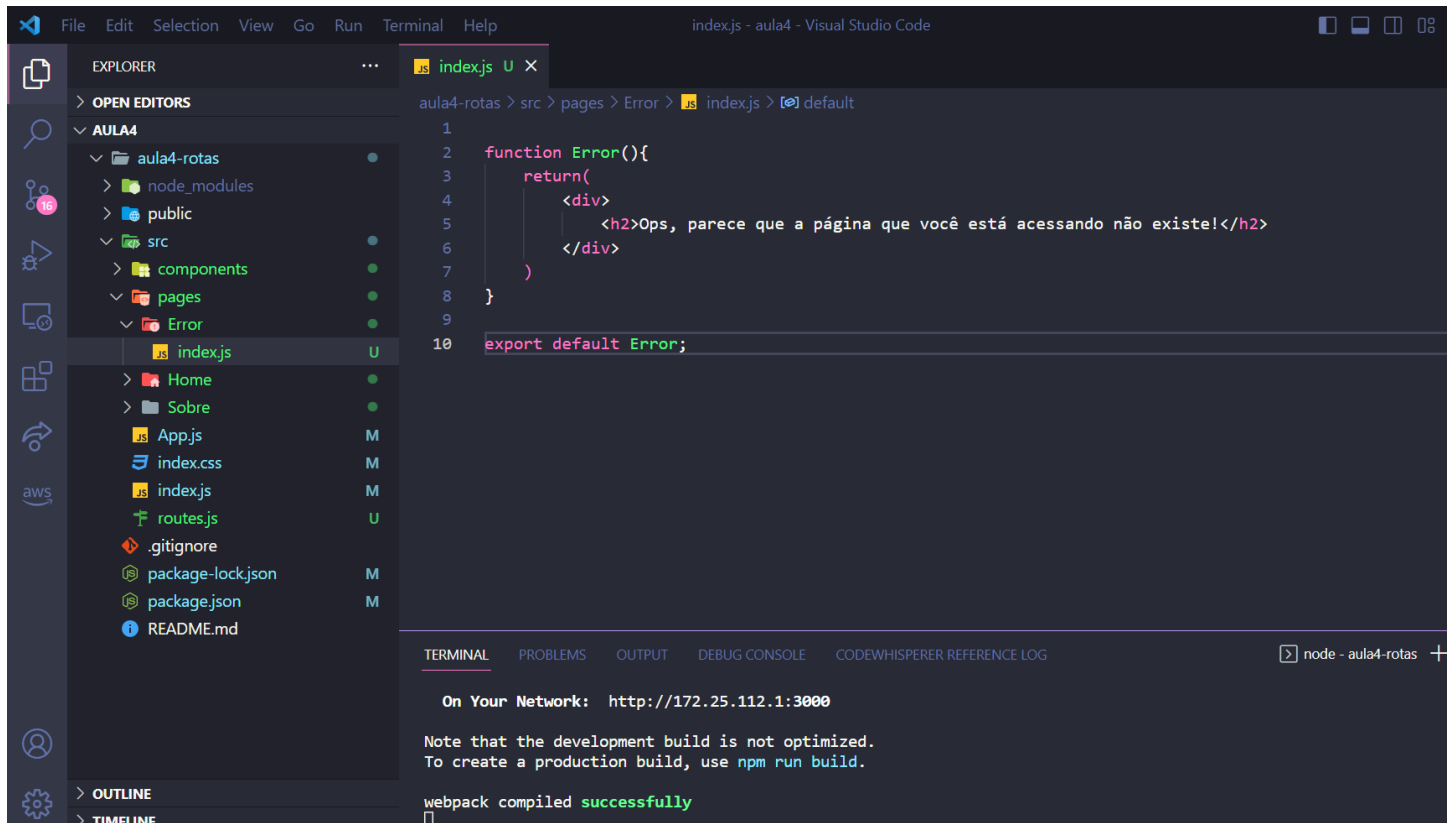
34 – Sua página deverá estar como abaixo é demonstrado.



Rota 'Not Found'

E quando o usuário tenta acessar uma página que não existe. Vamos agora configurar o comportamento de página não encontrada.

35 – Crie uma pasta chamada Error dentro da pasta pages. Dentro da pasta Error crie um arquivo index.js, dentro desse index uma função que retorna uma frase genérica de página não encontrada.



```
File Edit Selection View Go Run Terminal Help
index.js - aula4 - Visual Studio Code

EXPLORER
OPEN EDITORS
AULA4
  aula4-rotas
    node_modules
    public
    src
      components
      pages
      Error
        index.js
      Home
      Sobre
      App.js
      index.css
      index.js
      routes.js
      .gitignore
      package-lock.json
      package.json
      README.md

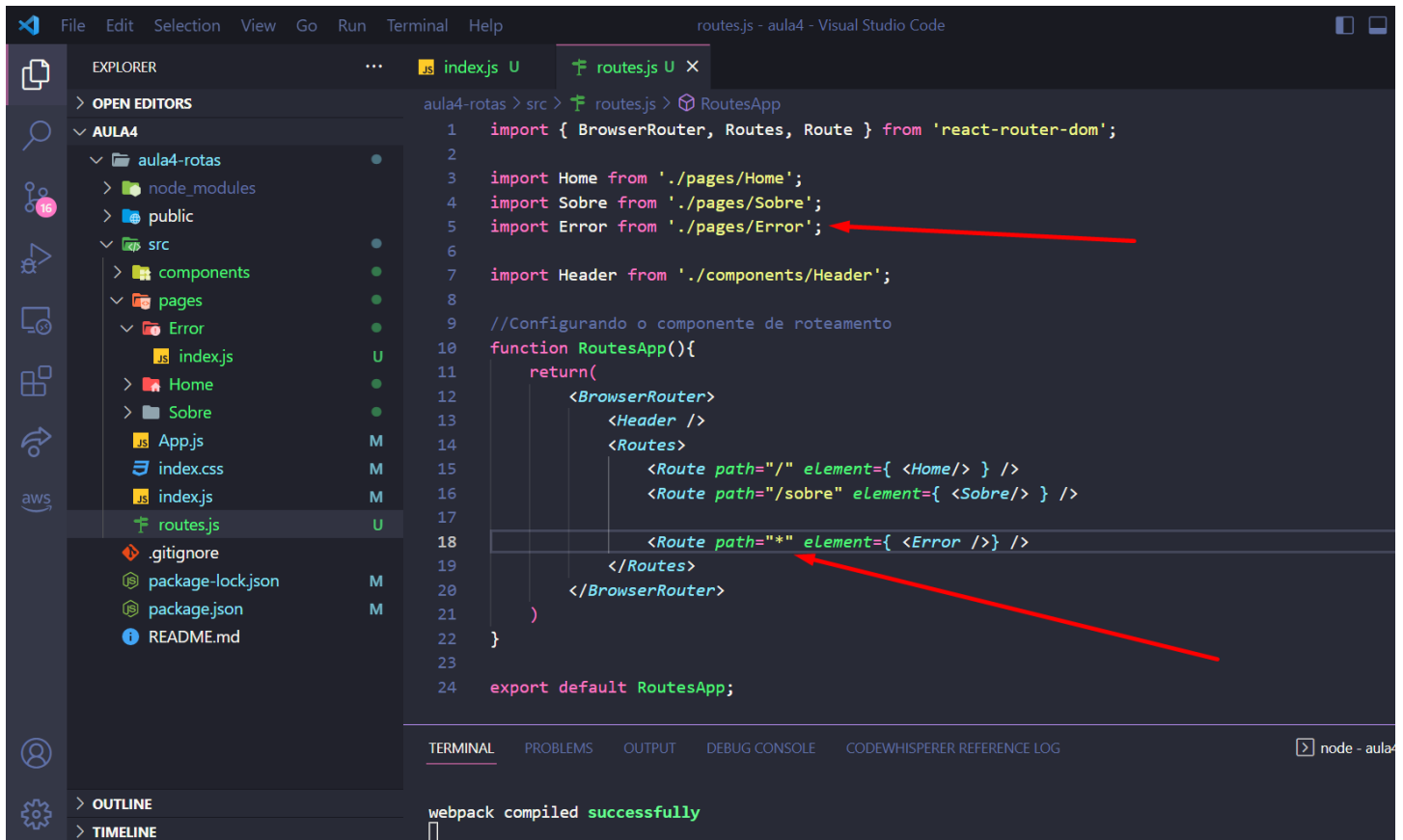
index.js U x
aula4-rotas > src > pages > Error > index.js > default
1
2 function Error(){
3   return(
4     <div>
5       <h2>Ops, parece que a página que você está acessando não existe!</h2>
6     </div>
7   )
8 }
9
10 export default Error;
```

On Your Network: <http://172.25.112.1:3000>

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

webpack compiled **successfully**

36 – Agora vamos configurar a rota para essa página. Faça a importação do Error e dentro da função RouteApp chame esse componente Error. Como queremos que qualquer página que não existe caia nessa mensagem de erro, coloque * no path

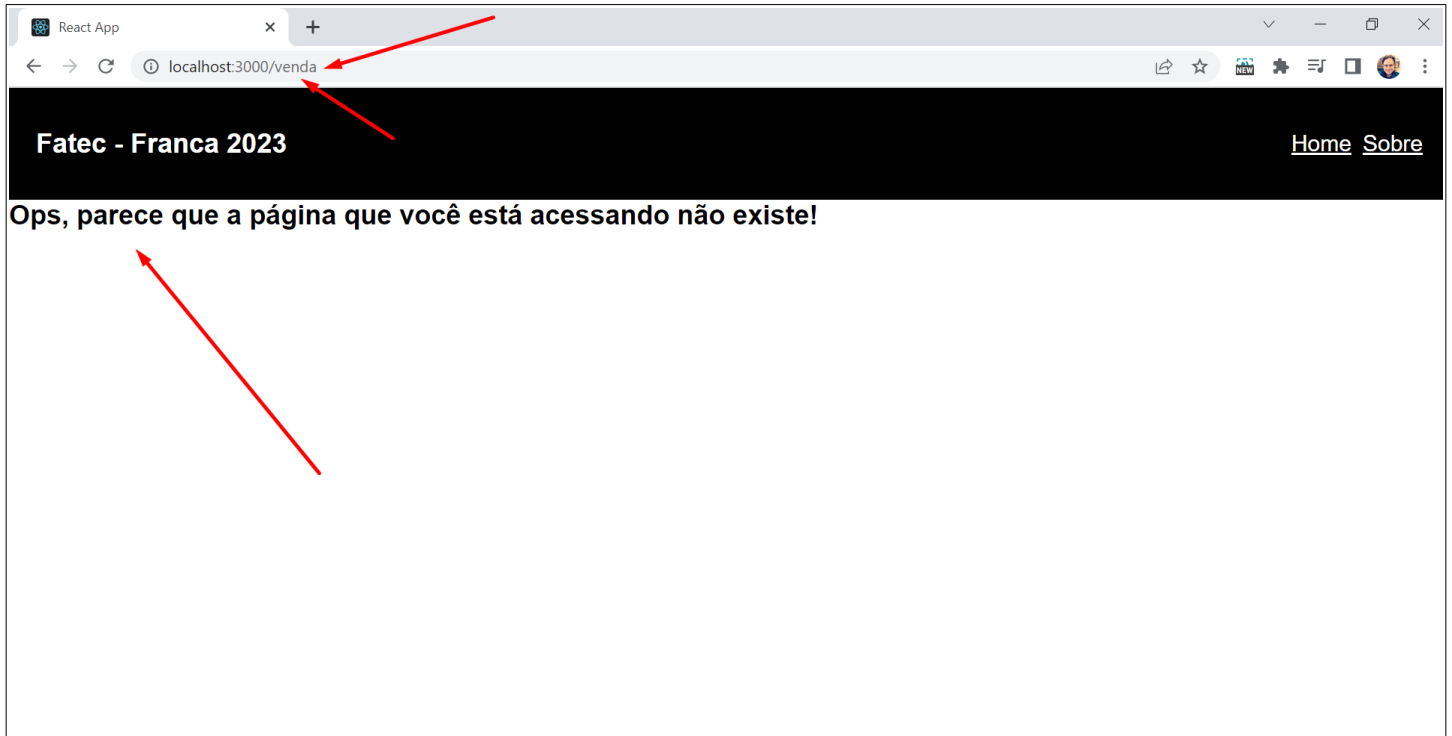


The screenshot shows the Visual Studio Code editor with the file explorer on the left and the code editor in the center. The file explorer shows the project structure for 'aula4-rotas', including a 'src' directory with 'pages' and 'components' subdirectories. The 'pages' directory contains 'Error', 'Home', and 'Sobre' files. The 'components' directory contains 'Header'. The code editor shows the 'routes.js' file with the following code:

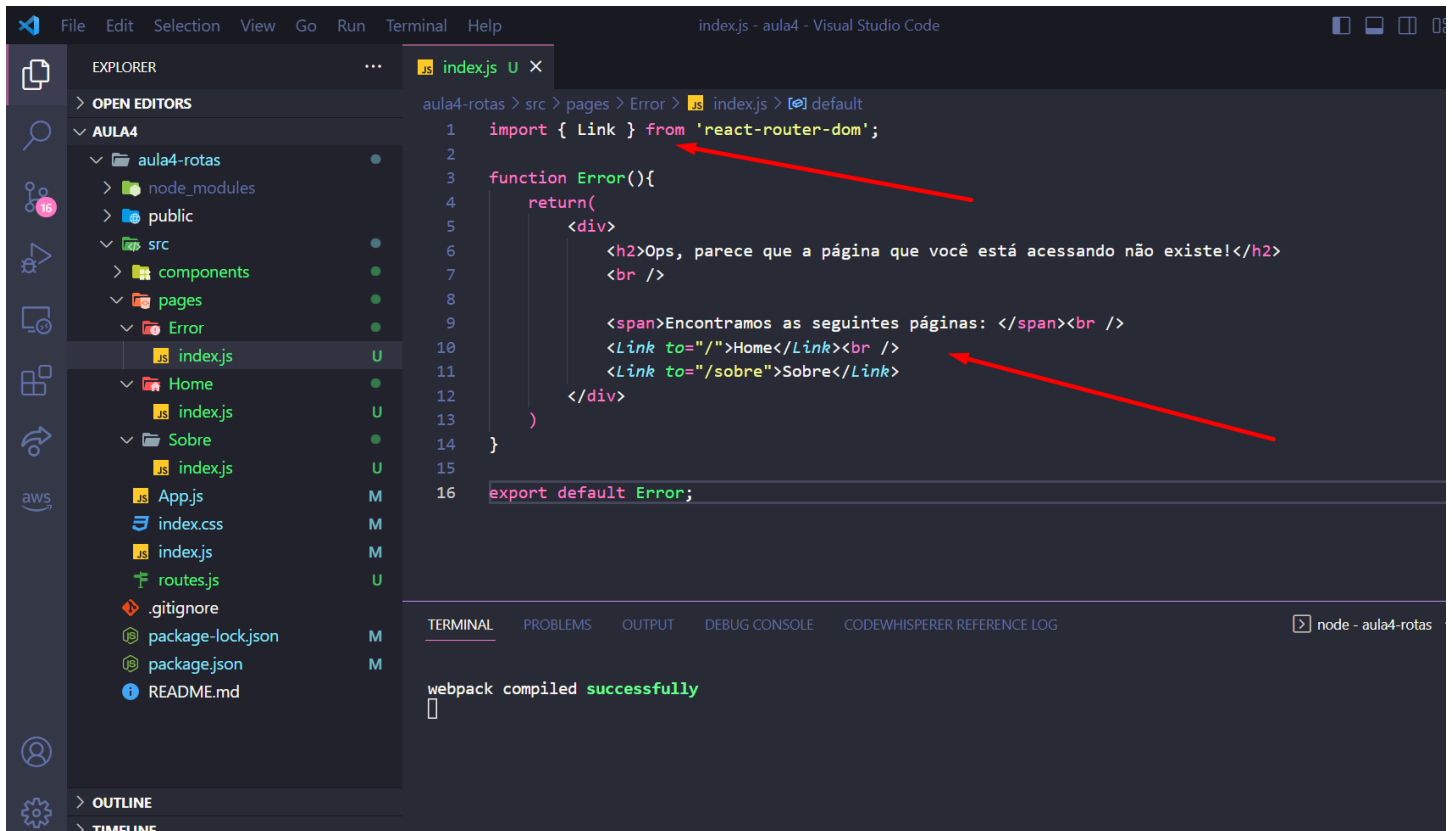
```
1 import { BrowserRouter, Routes, Route } from 'react-router-dom';
2
3 import Home from './pages/Home';
4 import Sobre from './pages/Sobre';
5 import Error from './pages/Error';
6
7 import Header from './components/Header';
8
9 //Configurando o componente de roteamento
10 function RoutesApp(){
11   return(
12     <BrowserRouter>
13       <Header />
14       <Routes>
15         <Route path="/" element={ <Home/> } />
16         <Route path="/sobre" element={ <Sobre/> } />
17         <Route path="*" element={ <Error /> } />
18       </Routes>
19     </BrowserRouter>
20   )
21 }
22
23
24 export default RoutesApp;
```

Two red arrows point to the import of the 'Error' component on line 5 and the catch-all route configuration on line 17. The terminal at the bottom shows the message 'webpack compiled successfully'.

37 – Agora no navegador, coloque /venda. Como essa pagina não existe nosso ‘Not Found’ entra em ação informando o usuário que essa página não existe.



38 – Podemos adicionar link a nossa página de ‘Not Found’ para melhorar a experiência do usuário.



```
File Edit Selection View Go Run Terminal Help index.js - aula4 - Visual Studio Code

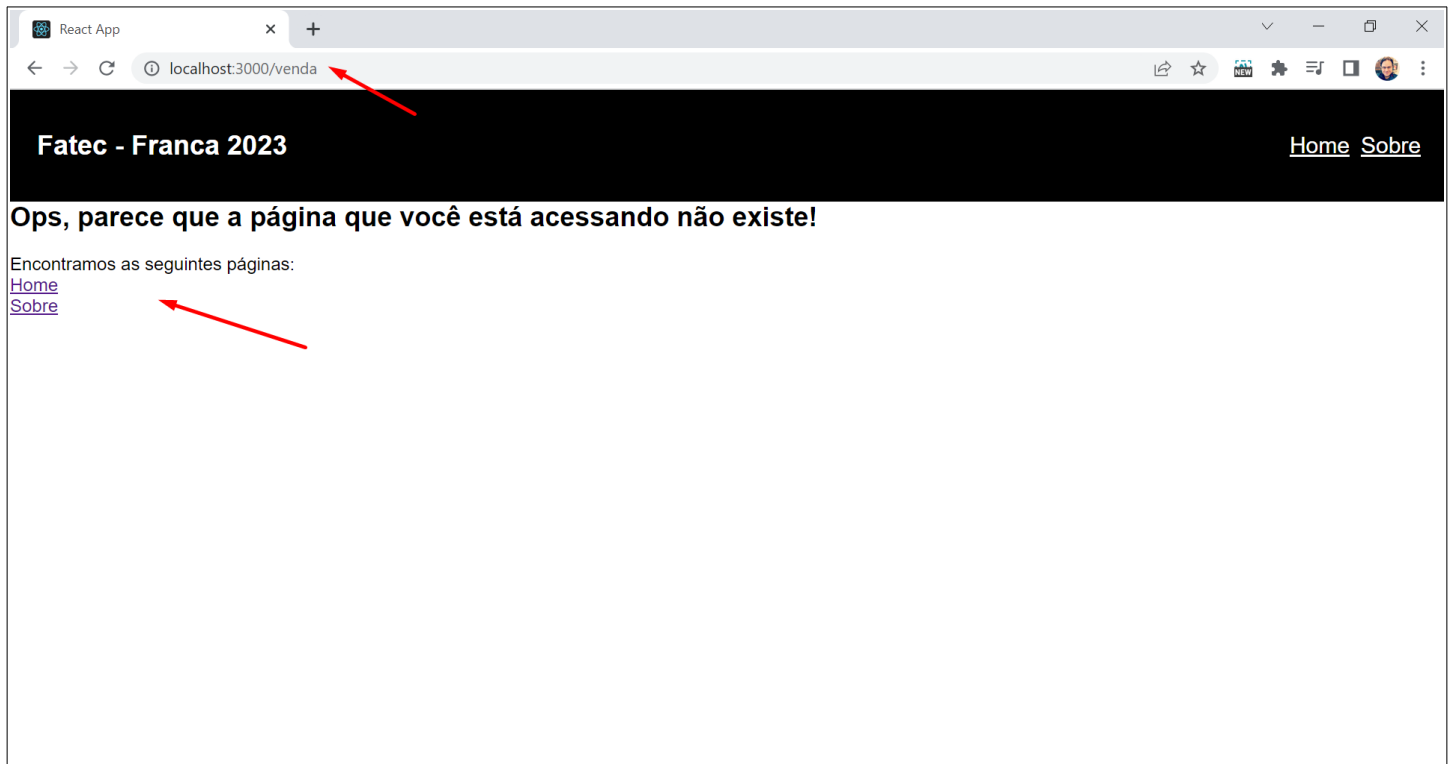
EXPLORER
OPEN EDITORS
AULA4
  aula4-rotas
    node_modules
    public
    src
      components
      pages
      Error
        index.js
      Home
        index.js
      Sobre
        index.js
      App.js
      index.css
      index.js
      routes.js
      .gitignore
      package-lock.json
      package.json
      README.md

OUTLINE
TIMELINE

aula4-rotas > src > pages > Error > index.js > default
1 import { Link } from 'react-router-dom';
2
3 function Error(){
4   return(
5     <div>
6       <h2>Ops, parece que a página que você está acessando não existe!</h2>
7       <br />
8       <span>Encontramos as seguintes páginas: </span><br />
9       <Link to="/">Home</Link><br />
10      <Link to="/sobre">Sobre</Link>
11    </div>
12  )
13 }
14
15 export default Error;
```

webpack compiled successfully

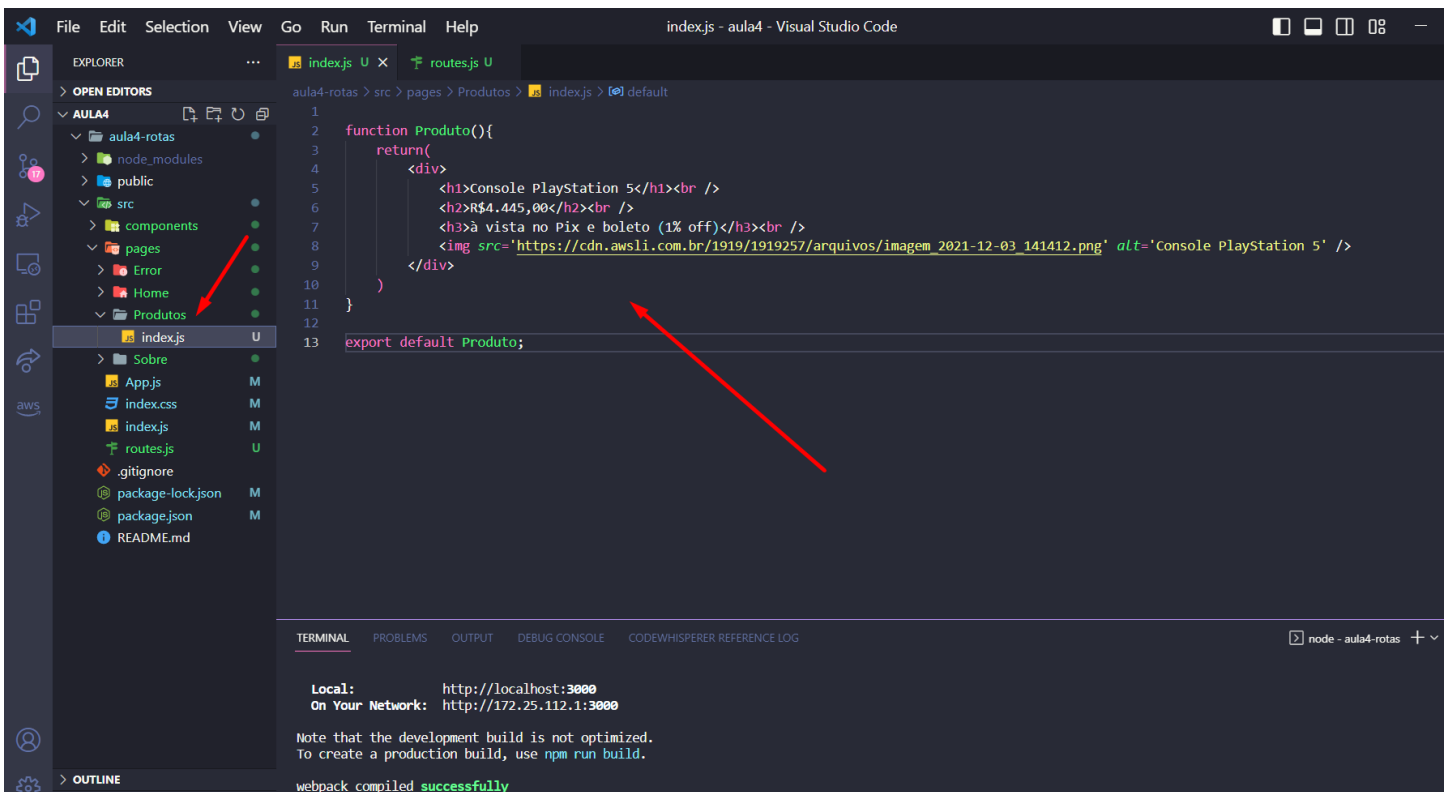
39 – Sua página deverá estar assim:



Parâmetros Dinâmicos em Rotas

Assim como em um e-commerce, muitas vezes uma página de produtos pode ter outras várias páginas que demonstram produtos, porém, todas elas estão dentro da página produtos. Vamos agora deixar mais dinâmico a interação das rotas.

40 – Dentro de pages crie uma pasta chamada Produtos, dentro de Produtos um novo arquivo index.js. Crie a função Produto que irá retornar dados de um produto. Caso queira adicionar uma imagem, basta copiar o endereço de uma imagem da web e colar dentro da tag :



```
File Edit Selection View Go Run Terminal Help
index.js - aula4 - Visual Studio Code

EXPLORER
OPEN EDITORS
aula4-rotas
  node_modules
  public
  src
    components
    pages
      index.js
      routes.js
    .gitignore
    package-lock.json
    package.json
    README.md

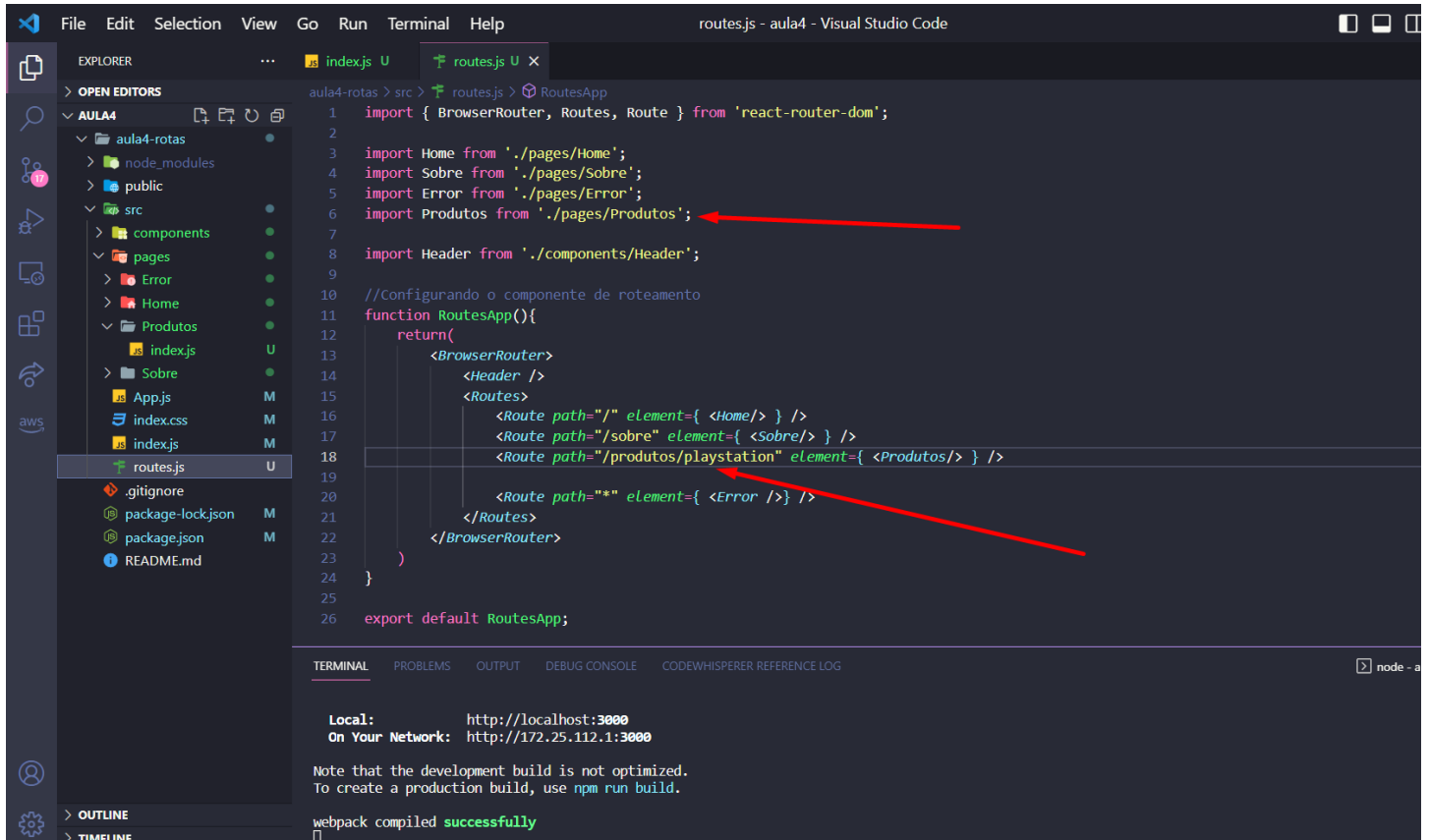
index.js
1
2 function Produto(){
3   return(
4     <div>
5       <h1>Console PlayStation 5</h1><br />
6       <h2>R$4.445,00</h2><br />
7       <h3>à vista no Pix e boleto (1% off)</h3><br />
8       <img src='https://cdn.awsli.com.br/1919/1919257/arquivos/imagem_2021-12-03_141412.png' alt='Console PlayStation 5' />
9     </div>
10  )
11 }
12
13 export default Produto;
```

Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

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

webpack compiled successfully

41 – Como temos uma nova página precisamos configurar sua nova rota. Veja que como queremos que apareça o nome do produto nessa rota, vamos chama-la de produtos/playstation.



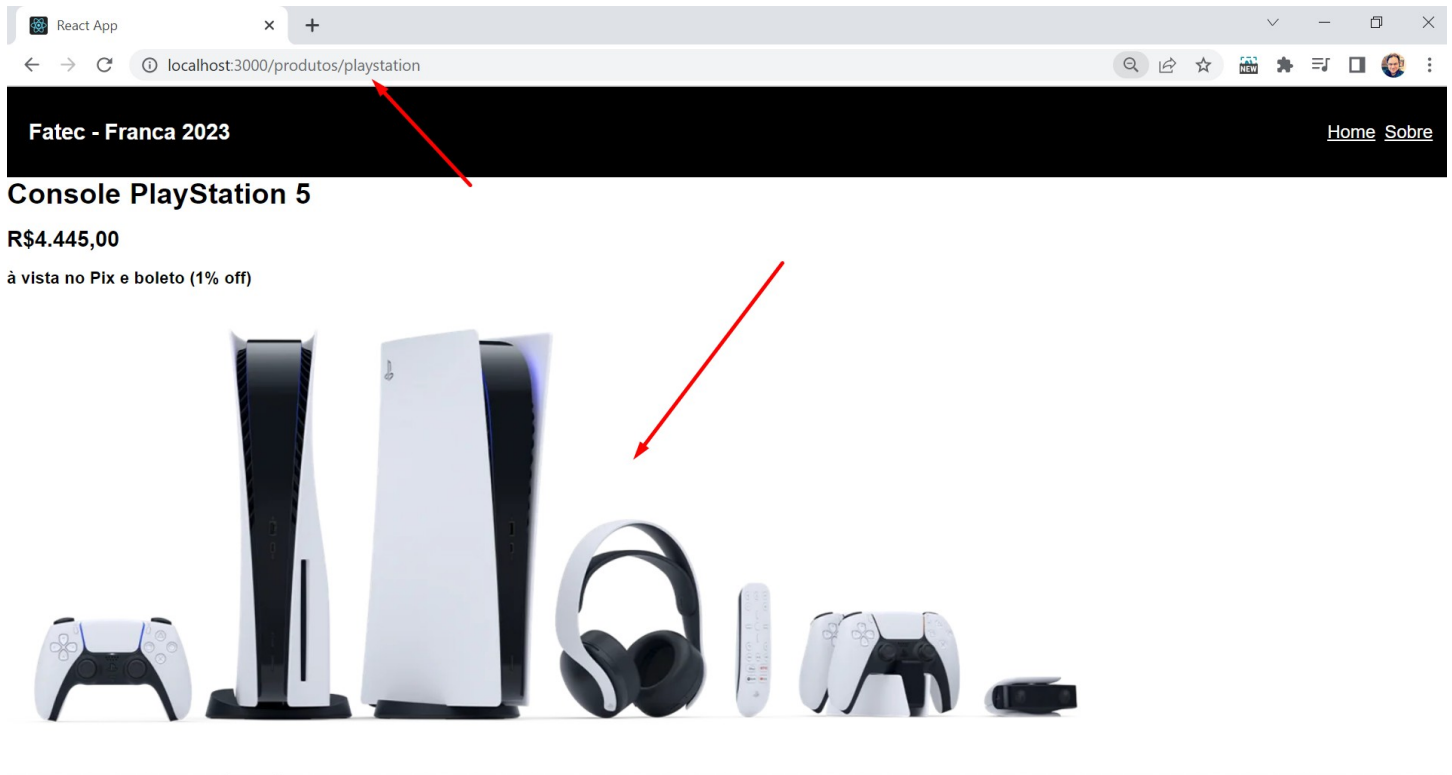
```
1 import { BrowserRouter, Routes, Route } from 'react-router-dom';
2
3 import Home from './pages/Home';
4 import Sobre from './pages/Sobre';
5 import Error from './pages/Error';
6 import Produtos from './pages/Produtos';
7
8 import Header from './components/Header';
9
10 //Configurando o componente de roteamento
11 function RoutesApp(){
12   return(
13     <BrowserRouter>
14       <Header />
15       <Routes>
16         <Route path="/" element={ <Home/> } />
17         <Route path="/sobre" element={ <Sobre/> } />
18         <Route path="/produtos/playstation" element={ <Produtos/> } />
19       </Routes>
20     </BrowserRouter>
21   )
22 }
23
24 export default RoutesApp;
```

Local: http://localhost:3000
On Your Network: http://172.25.112.1:3000

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

webpack compiled **successfully**

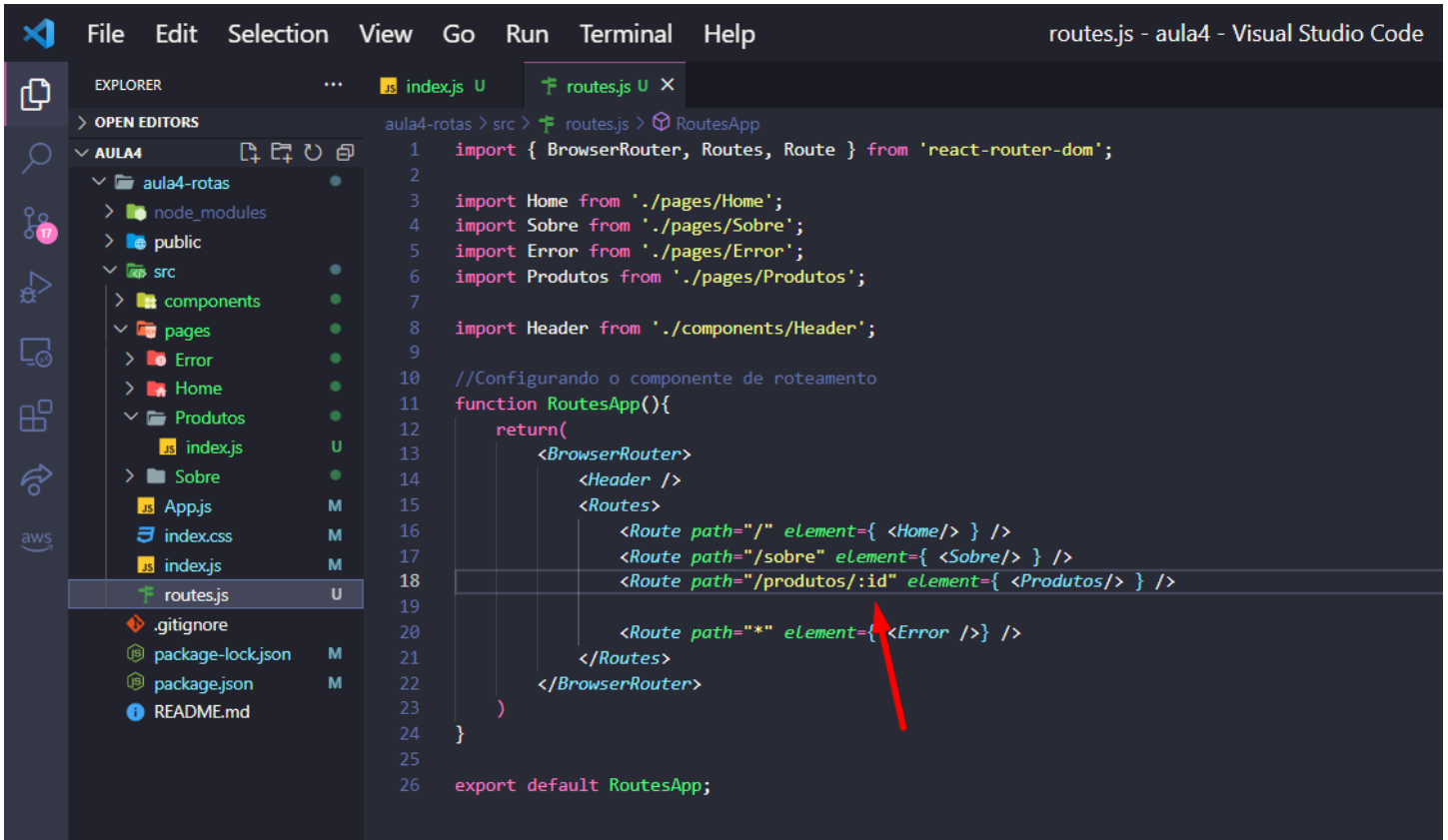
42 – Agora ao digitar *produtos/playstation* nosso produto é carregado:



43 – Agora, uma loja tem muitos produtos. Seria muito trabalhoso criar mais de 1000 rotas para exibir cada um dos produtos. Por isso podemos rotear de forma dinâmica. Vá até o index.js dentro de Produtos. Vamos importar o **useParams**. Depois crie uma const chamada id, depois 2 ifs, se o id for igual 'playstation' ele irá retornar algo e se for 'xbox' irá retornar outra informação.

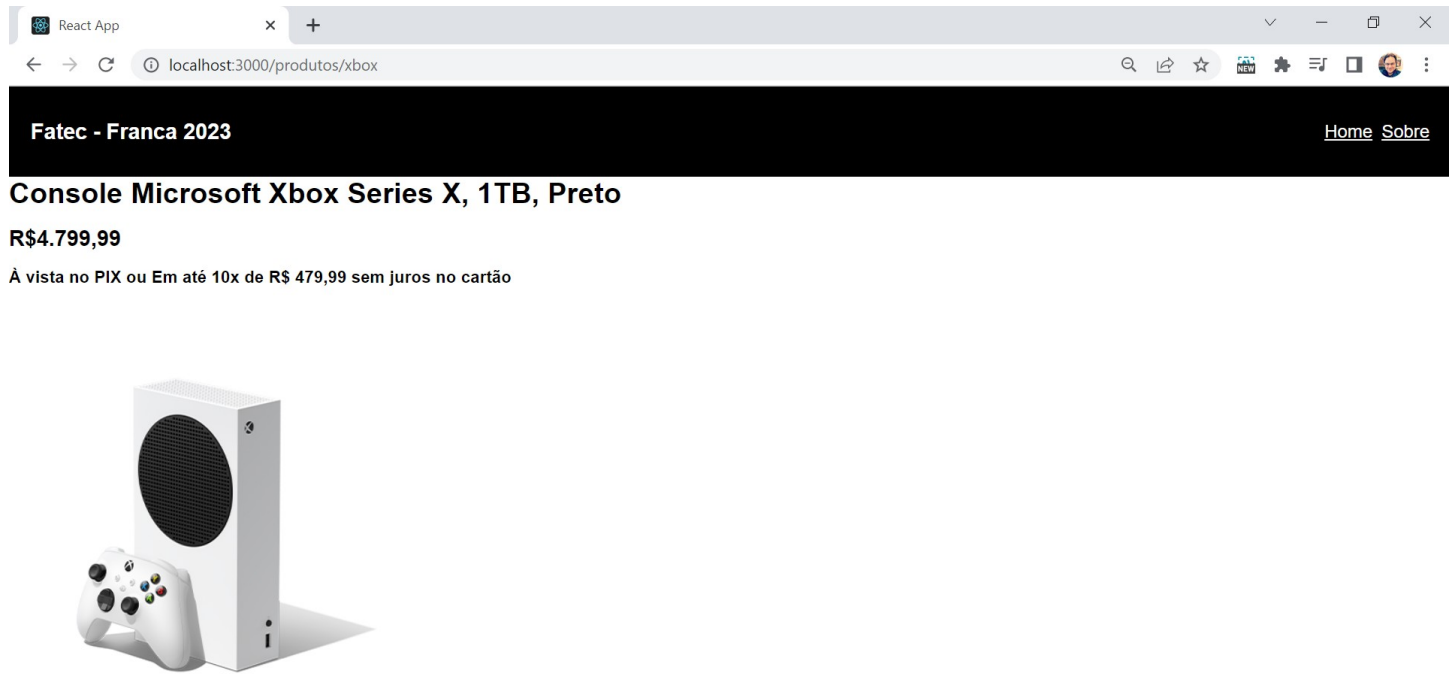
```
index.js U X routes.js U
aula4-rotas > src > pages > Produtos > index.js > ...
1 import { useParams } from 'react-router-dom';
2
3 function Produto(){
4
5     const { id } = useParams();
6
7     if(id === 'playstation'){
8         return(
9             <div>
10                 <h1>Console PlayStation 5</h1><br />
11                 <h2>R$4.445,00</h2><br />
12                 <h3>à vista no Pix e boleto (1% off)</h3><br />
13                 <img src='https://cdn.awsli.com.br/1919/1919257/arquivos/imagem_2021-12-03_141412.png' alt='Console PlayStation 5' />
14             </div>
15         )
16     }
17
18     if(id === 'xbox'){
19         return(
20             <div>
21                 <h1>Console Microsoft Xbox Series X, 1TB, Preto</h1><br />
22                 <h2>R$4.799,99</h2><br />
23                 <h3>À vista no PIX ou Em até 10x de R$ 479,99 sem juros no cartão</h3><br />
24                 <img src='https://assets.xboxservices.com/assets/cb/f8/cbf83b9d-bf30-4e36-96ad-fef4293ff944.png?n=XBX-CMP-L-Console_Large-D_02.png' alt='Console Xbox X' />
25             </div>
26         )
27     }
28
29 }
30
31
32 export default Produto;
```

44 – Agora volte em routes.js, passe o parametro :id para a rota produtos. Agora quando o usuário digitar *produtos/xbox*. O id receberá essa informação e irá retornar a página do produto corretamente.



```
1 import { BrowserRouter, Routes, Route } from 'react-router-dom';
2
3 import Home from './pages/Home';
4 import Sobre from './pages/Sobre';
5 import Error from './pages/Error';
6 import Produtos from './pages/Produtos';
7
8 import Header from './components/Header';
9
10 //Configurando o componente de roteamento
11 function RoutesApp(){
12   return(
13     <BrowserRouter>
14       <Header />
15       <Routes>
16         <Route path="/" element={ <Home/> } />
17         <Route path="/sobre" element={ <Sobre/> } />
18         <Route path="/produtos/:id" element={ <Produtos/> } />
19       </Routes>
20     </BrowserRouter>
21   )
22 }
23
24 export default RoutesApp;
```

45 – Vá até o navegador e digite /produtos/xbox:



Ainda sim não é a melhor opção se tivessemos mais de 1000 produtos, o ideal nesse caso seria utilizar APIs e montar o nome do produto, seu preço e imagem de forma dinâmica.

Exercícios

Utilizando o projeto que realizamos nessa aula:

- 1 – Crie uma página chamada **contato**, a rota dela será `/informacoes/contato`.
- 2 – Ao usuário digitar no endereço '<http://localhost:3000/informacoes/contato>' deverá aparecer as informações de contato, desse modo:



The image shows a Google Maps interface with a sidebar on the right containing contact information for Fatec Franca. The map on the left shows the location of Fatec Franca in Franca, SP, with a red pin and a search box. The sidebar on the right contains the following information:

Faculdade de Tecnologia de Franca • Fatec Franca
CNPJ/MF: 62.823.257/0109-10
Rua Irênio Grecco nº 4580
Vila Imperador
Franca/SP
14405-191
Telefone: (16) 3702-3204
Fax: (16) 3702-2854

E-mail: secretaria@fatecfranca.edu.br

Sempre verifique a pasta de Spam/Lixo eletrônico ao buscar respostas de e-mails da Fatec Franca

Linhas de ônibus urbano

- J08 (Vl. Imperador) – ponto na R. Irênio Grecco
- C01 (Circular 01), C02 (Circular 02) e LDN (Linha Direta Norte) – ponto na Av. Orlando Domplieri
- J09 (Vl. Imperador via Id. Planalto) – ponto na R. Realindo Jacinto Mendonça

Fonte: Empresa São José

Pesquise meios de colocar o Google Maps dentro da página.

- 3 – Crie uma nova rota para a página **contato**. Essas rotas devem estar presentes apenas no menu do Header.
- 5 – Crie 2 novos produtos, um chamado Atari e outro Super Nintendo. Coloque preço, imagens e etc.
- 6 – Faça com que o usuário ao acessar '<http://localhost:3000/produtos/atari>' veja o produto de modo dinâmico.
- 7 – Crie uma nova página chamada **Loja** e coloque o link dos 4 produtos dentro dessa página.
- 8 – Configure a rota da página Loja para '<http://localhost:3000/loja>' para ser acessada tanto da Home quanto do Header.