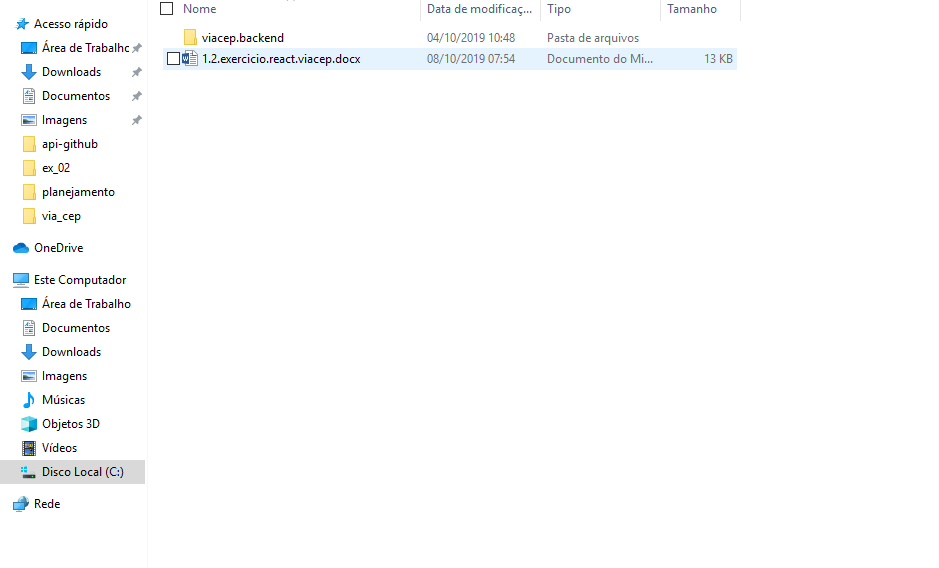
Tendo o node, NPM e o create-react-app instalados em sua máquina conseguimos iniciar nosso projeto

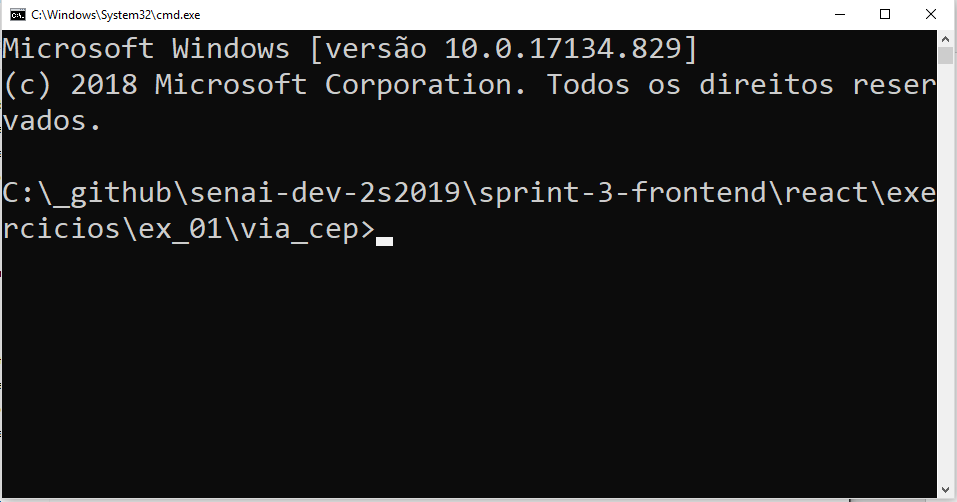
Caso você não possua algum desses, acesse o documento com o tutorial de instalação dessas ferramentas (roteiro do Gufos).

# Criação do projeto

Selecionar a pasta que deseja criar o meu projeto



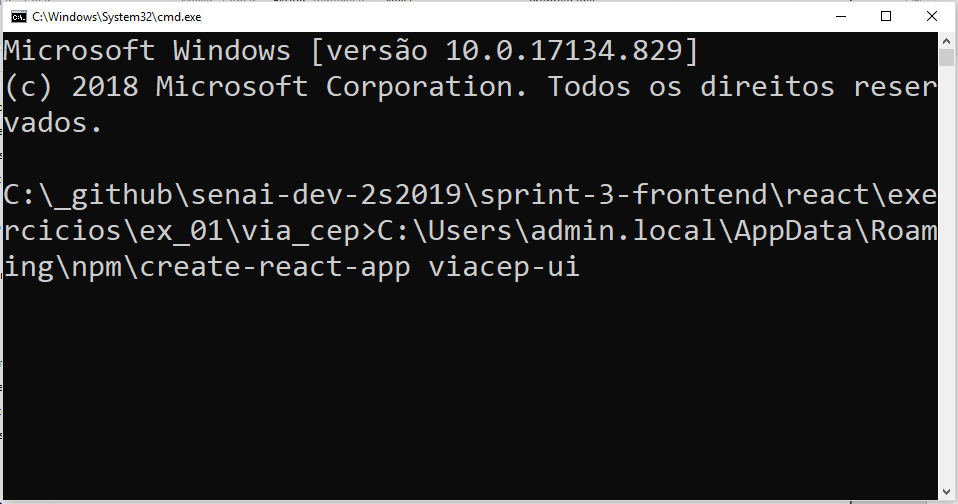
Executar cmd na pasta para abrir direto no caminho dela



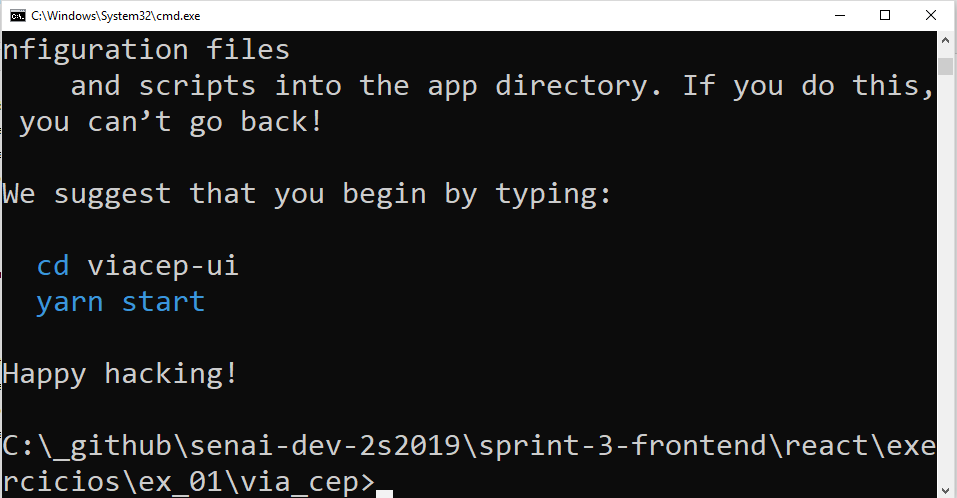
No mesmo cmd passar o caminho da pasta onde está instalado o create-react-app, normalmente ele é instalado no caminho:

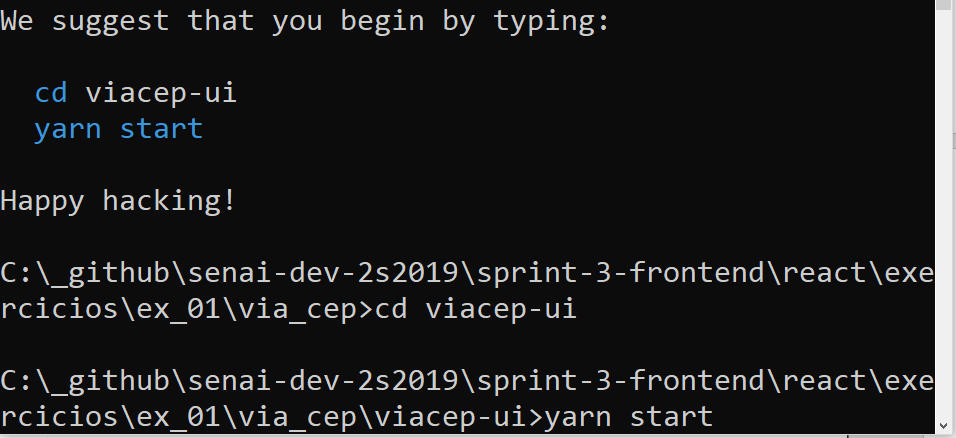
***C:\Users\admin.local\AppData\Roaming\npm\***

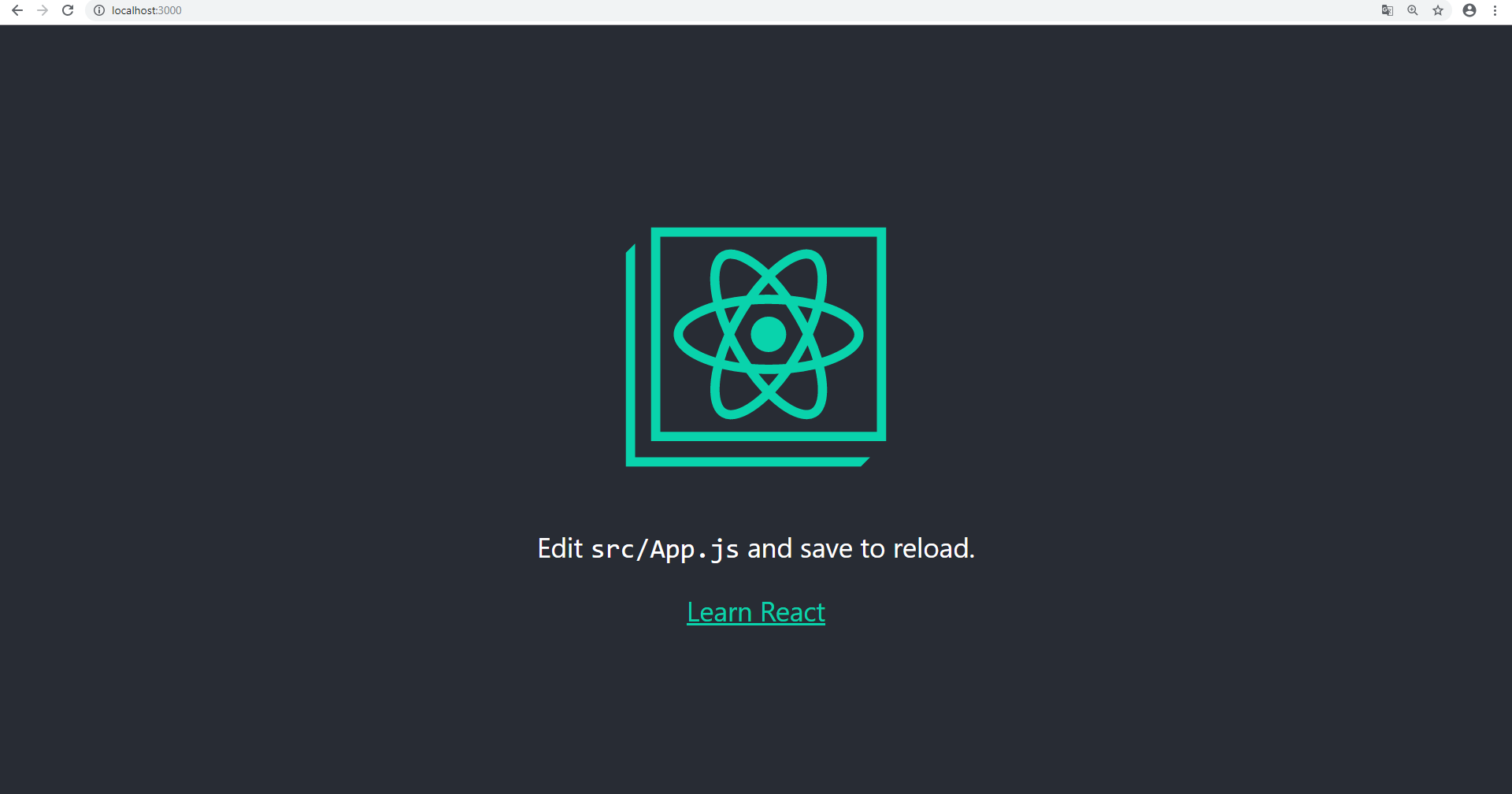
Passando o caminho, confirme se no final do npm existe uma barra invertida (“\”), e em seguida coloque create-react-app <nome-do-projeto>



Após o carregamento tendo tudo dado certo você terá uma mensagem de sucesso no seu cmd. Siga os comandos passados em azul (ou em outra cor que esteja destacada) para iniciar sua aplicação React.

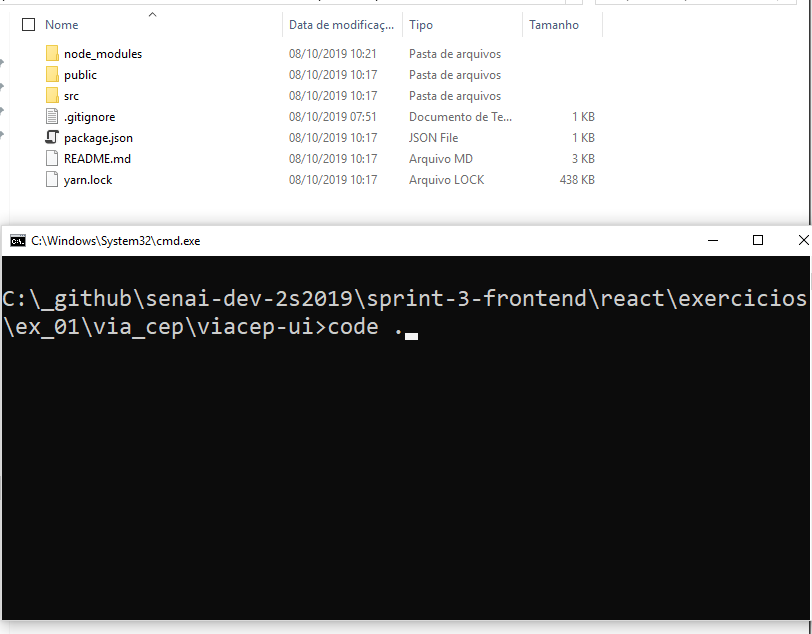




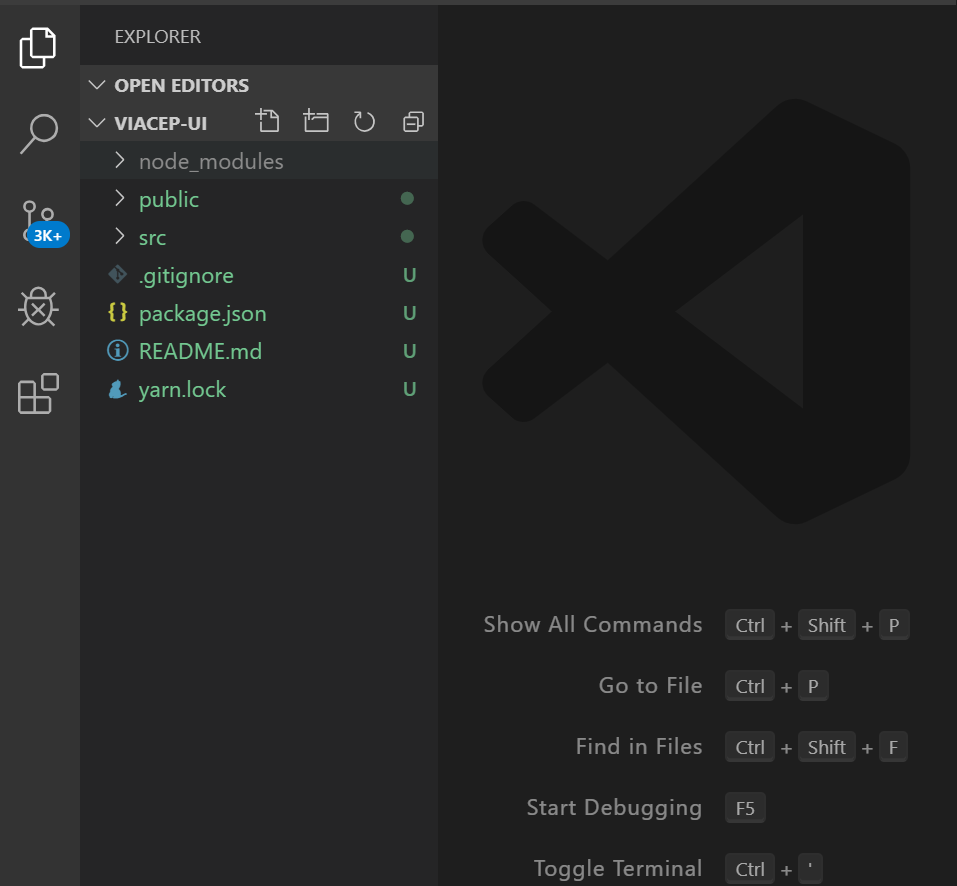


# Organização de pastas

Abrindo o CMD na raiz do meu projeto, digito “code .” para abrir o meu projeto no VS Code.



Teremos nossa estrutura inicial de projeto



Perguntas a serem feitas que podem ajudar no inicio de um projeto React:

- O que minha aplicação vai fazer?

- Quantas páginas precisarei ter?

- Sabendo minhas páginas, precisarei de componentes?

- Farei requisições da minha aplicação? Quando?

Sabendo responder essas informações conseguimos ter noção do que precisaremos fazer e por onde começamos.

Normalmente, um layout nos da boa noção e responde todas essas perguntas.

Respondendo a essas perguntas:

- O que minha aplicação vai fazer?

**Vai exibir para meu usuário as informações do CEP digitado pelo mesmo.**

- Quantas páginas precisarei ter?

**Apenas uma, que será onde o usuário digitará o CEP**

- Sabendo minhas páginas, precisarei de componentes?

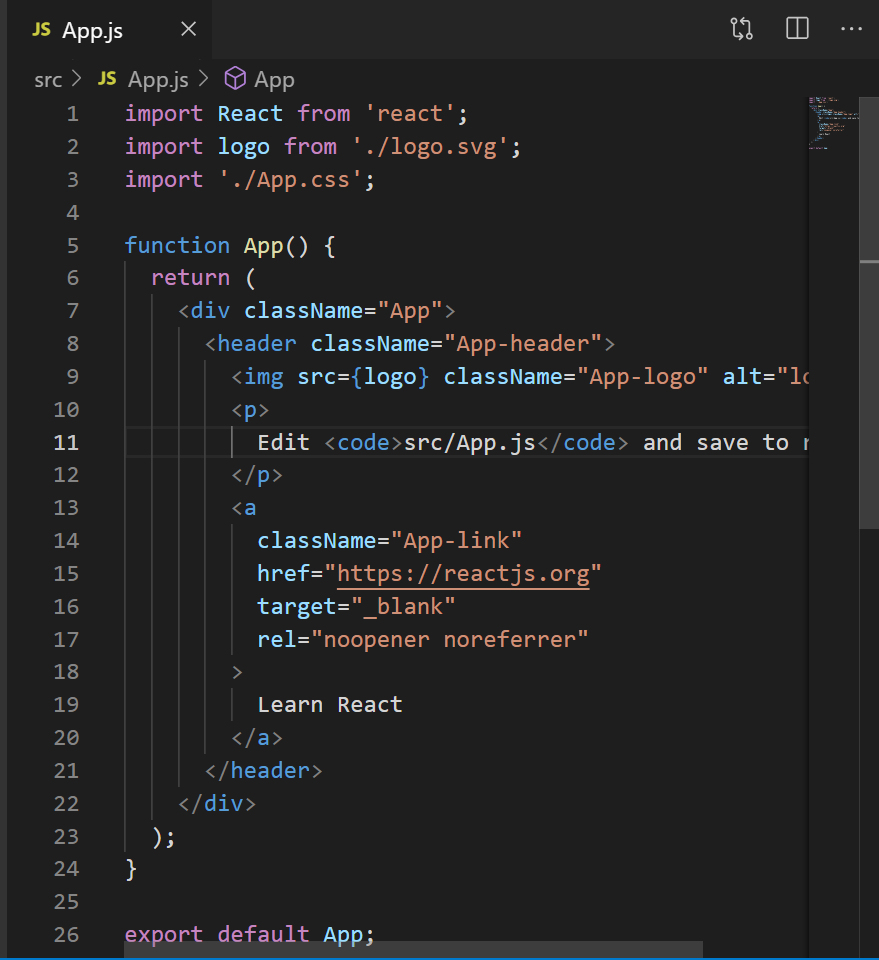
**Um componente input e uma tabela que serão projetadas as informações obtidas.**

- Farei requisições da minha aplicação? Quando?

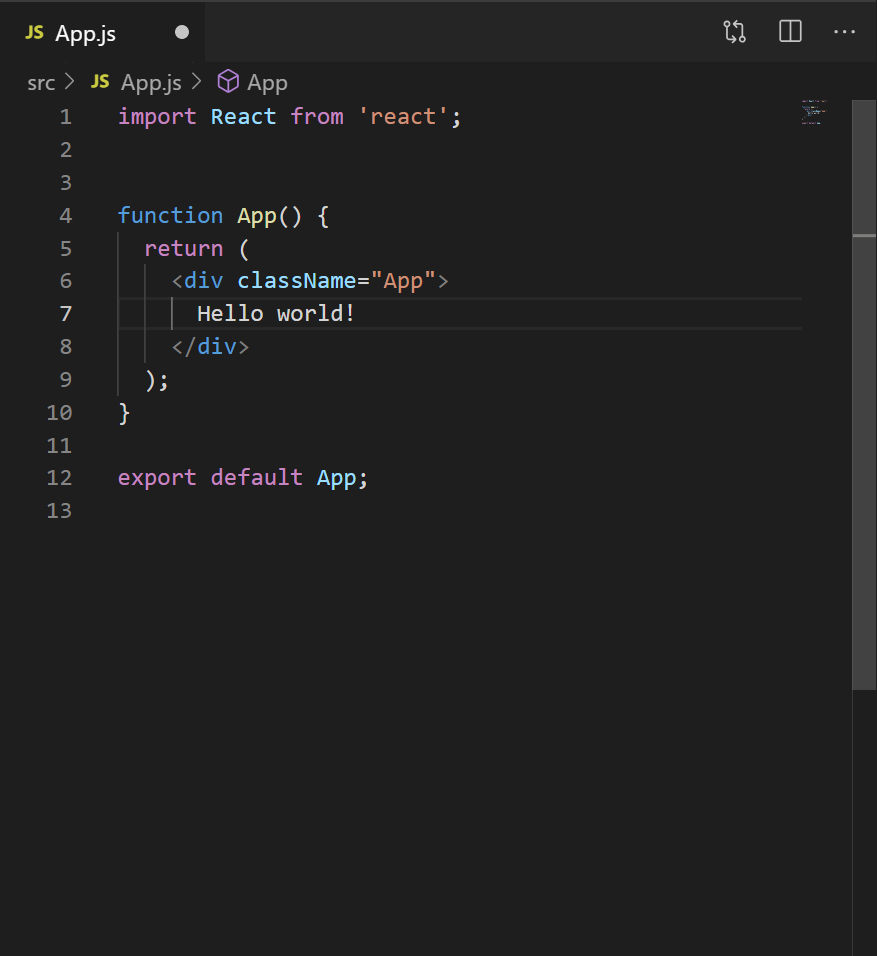
**Sim, para a API do viacep quando o usuário clicar no botão, tendo digitado o CEP no campo de texto.**

Repare que algumas dessas respostas ainda ficam em aberto, gerando outras dúvidas, não nos aprofundaremos nesse conceito aqui, mas quanto mais vocês conseguirem detalhar essas respostas num ponto que não gerem mais dúvidas, mais fácil será seu trabalho, seja no frontend ou no backend.

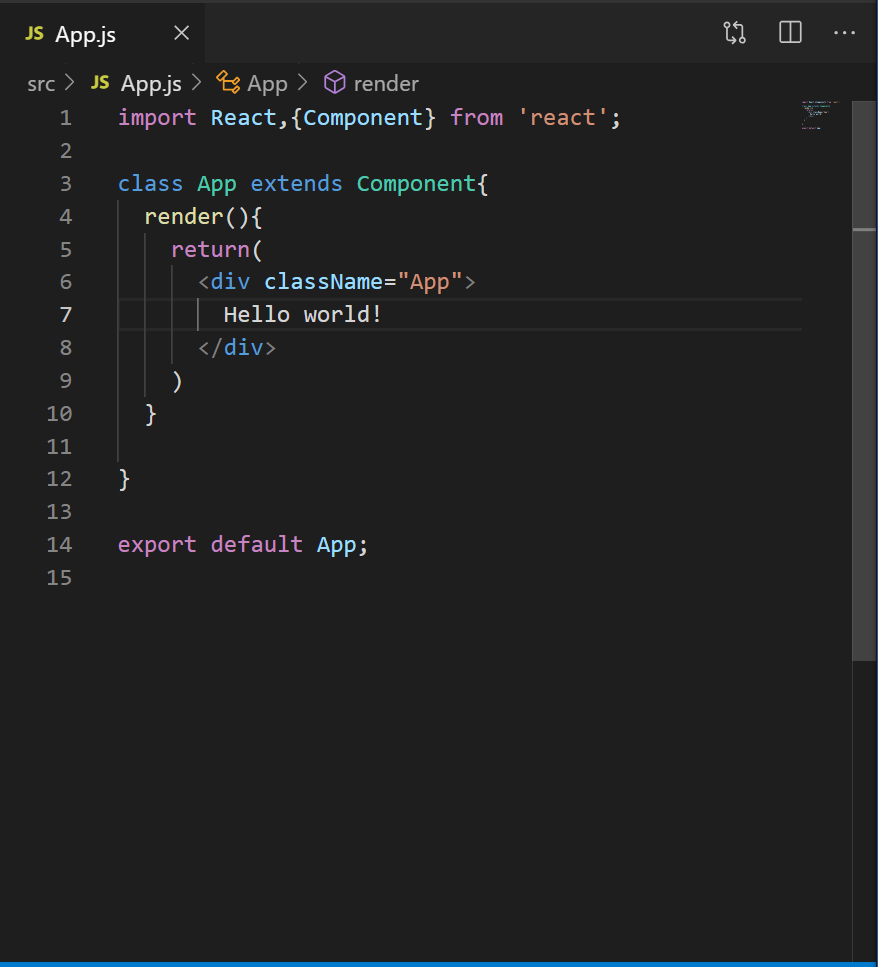
Tendo em mente as respostas obtidas posso construir minha página inicial toda em meu App.js



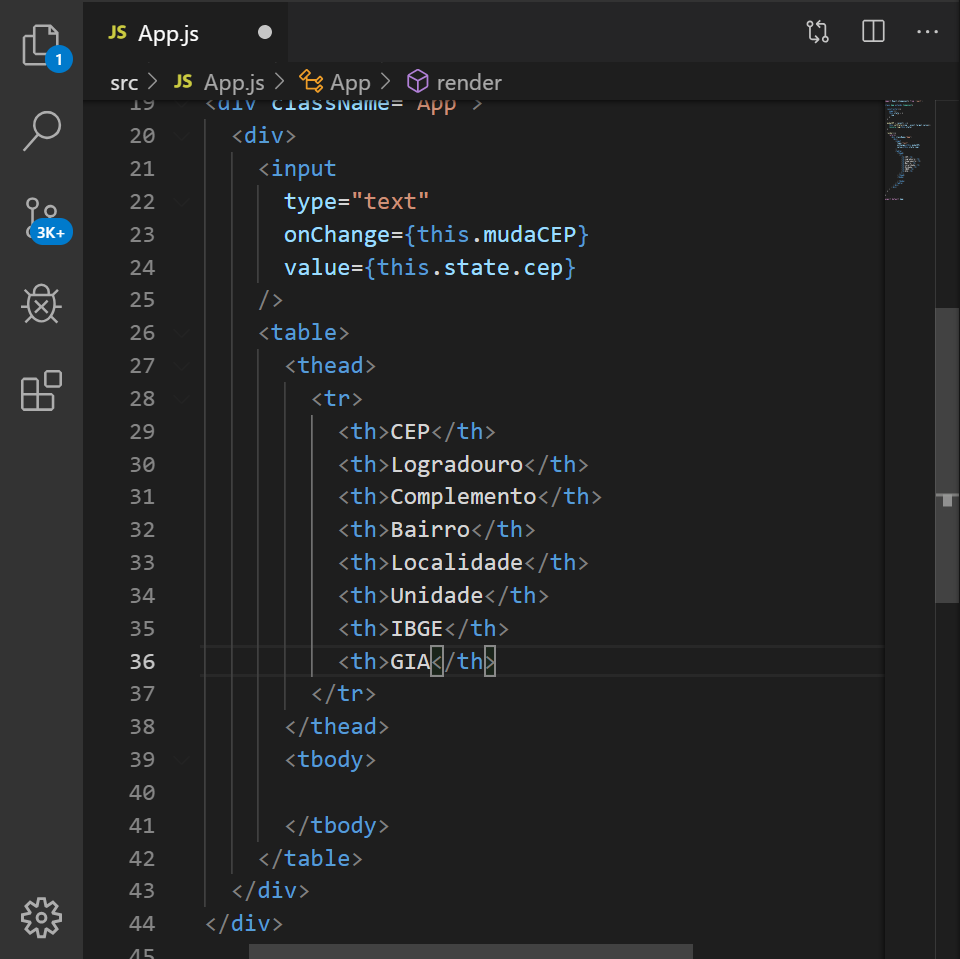
Apagamos o conteúdo que ele possui e seus imports mantendo apenas o import do próprio React.

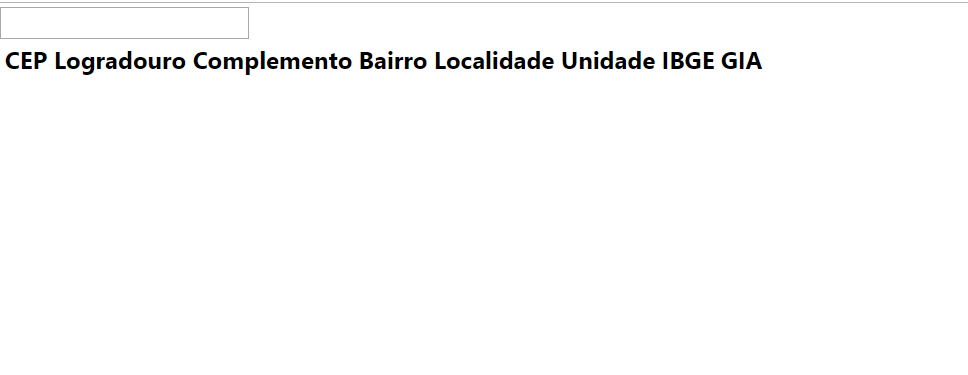


Trocaremos o componente App de função para class com sua respectiva estrutura.



Montaremos a estrutura JSX da página. Nossa tabela será baseada nas chaves que estaremos recebendo de nosso JSON.

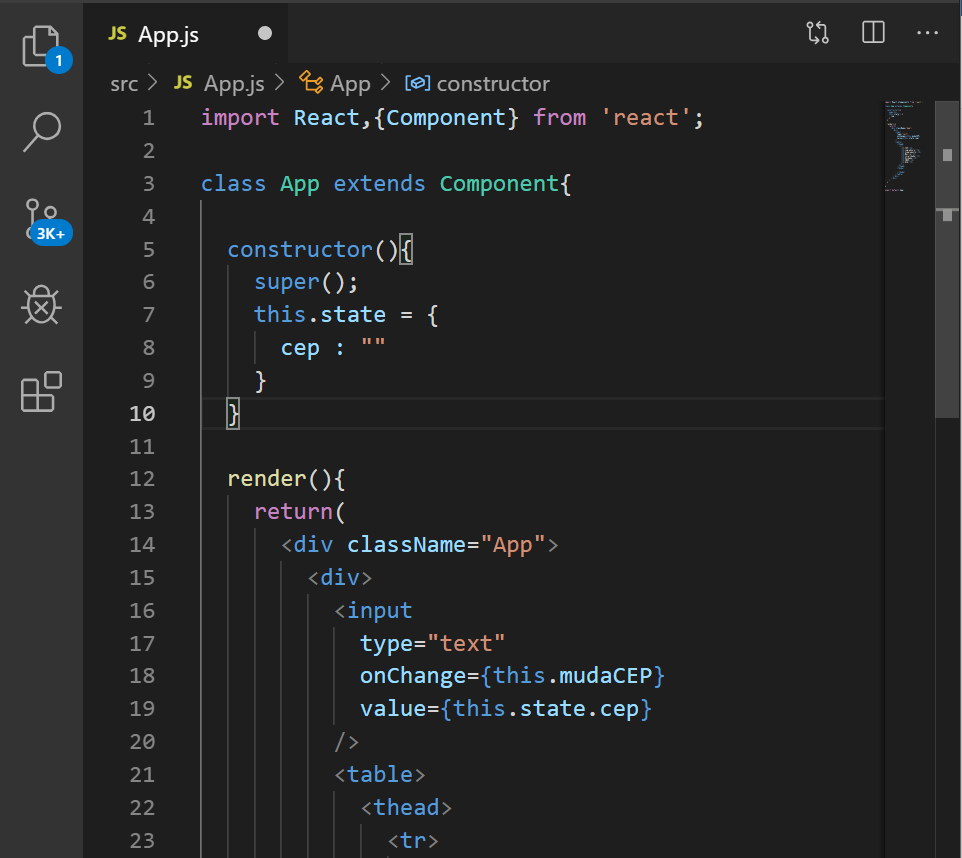




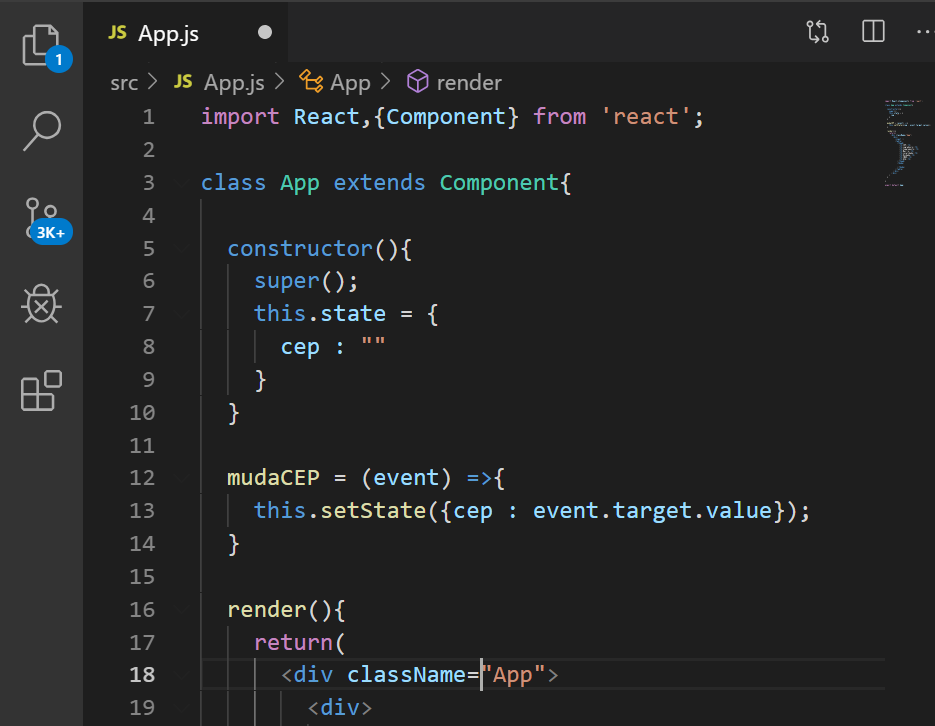
Capturar o valor do input

Essa tarefa torna-se a prioritária por precisarmos da informação imputada pelo usuário para depois fazer a requisição.

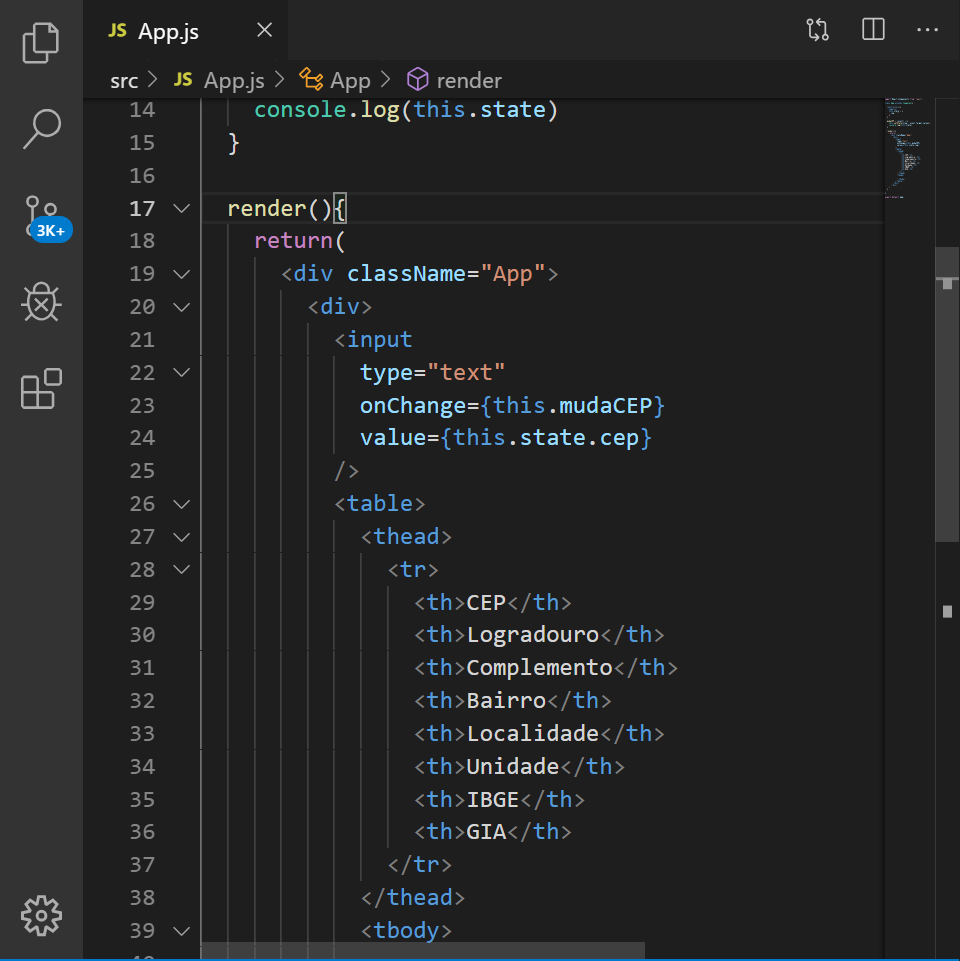
Sendo assim criamos nossa estrutura de state, para armazenar o value do input



Crio uma função para receber o valor do input envia esse valor recebido para meu state na chave CEP



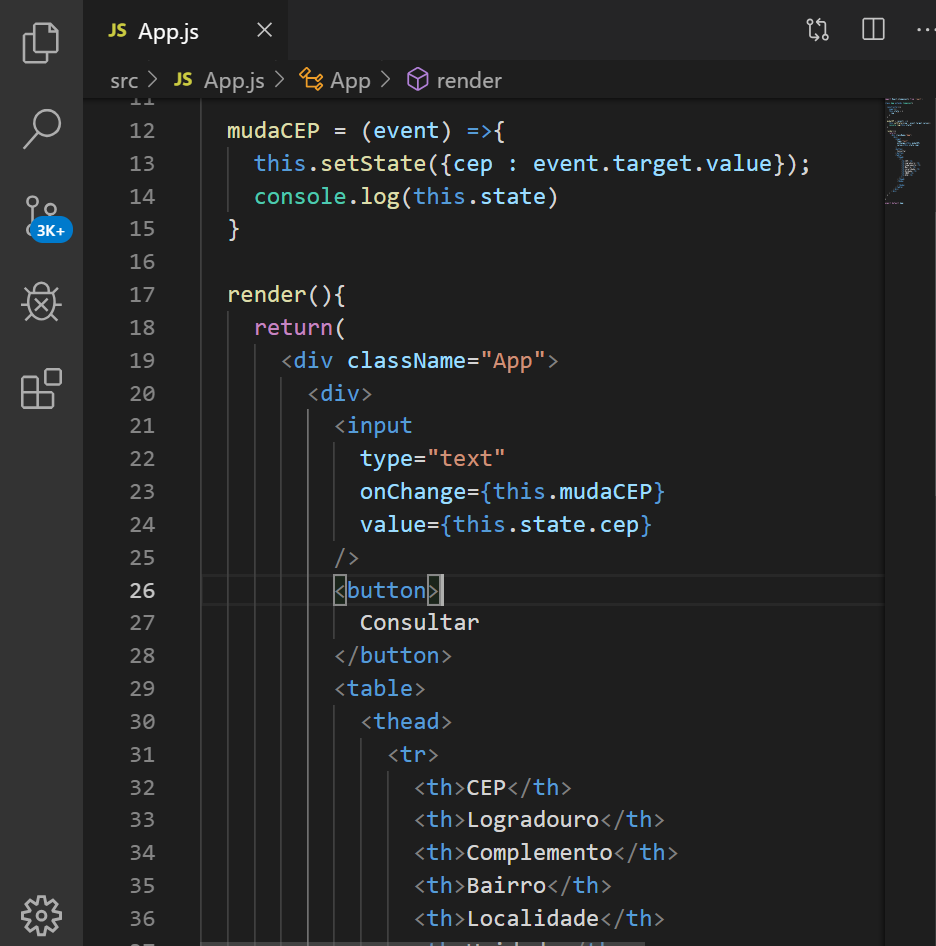
Vinculo o value do input com meu state cep e define o onChange para acionar nossa função mudaCEP



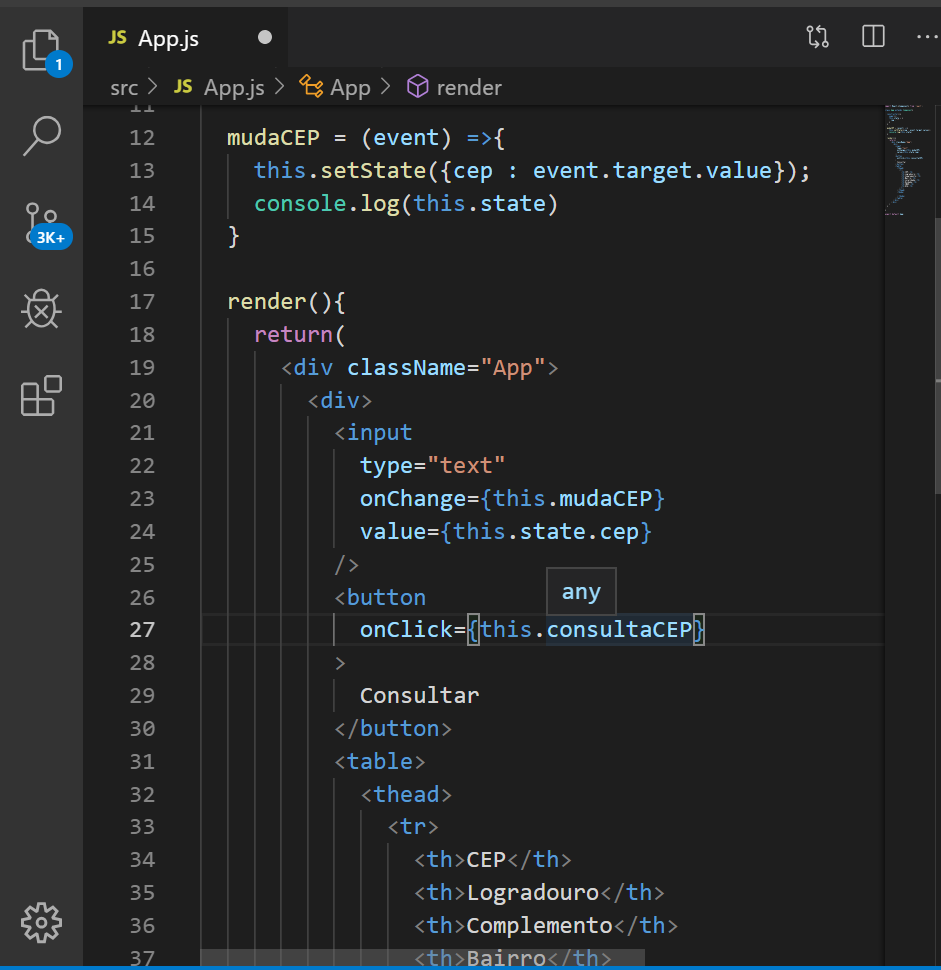
# Consulta a partir do valor digitado

Queremos agora consultar o CEP a partir do valor armazenado no state, quando o usuário clicar no botão consultar

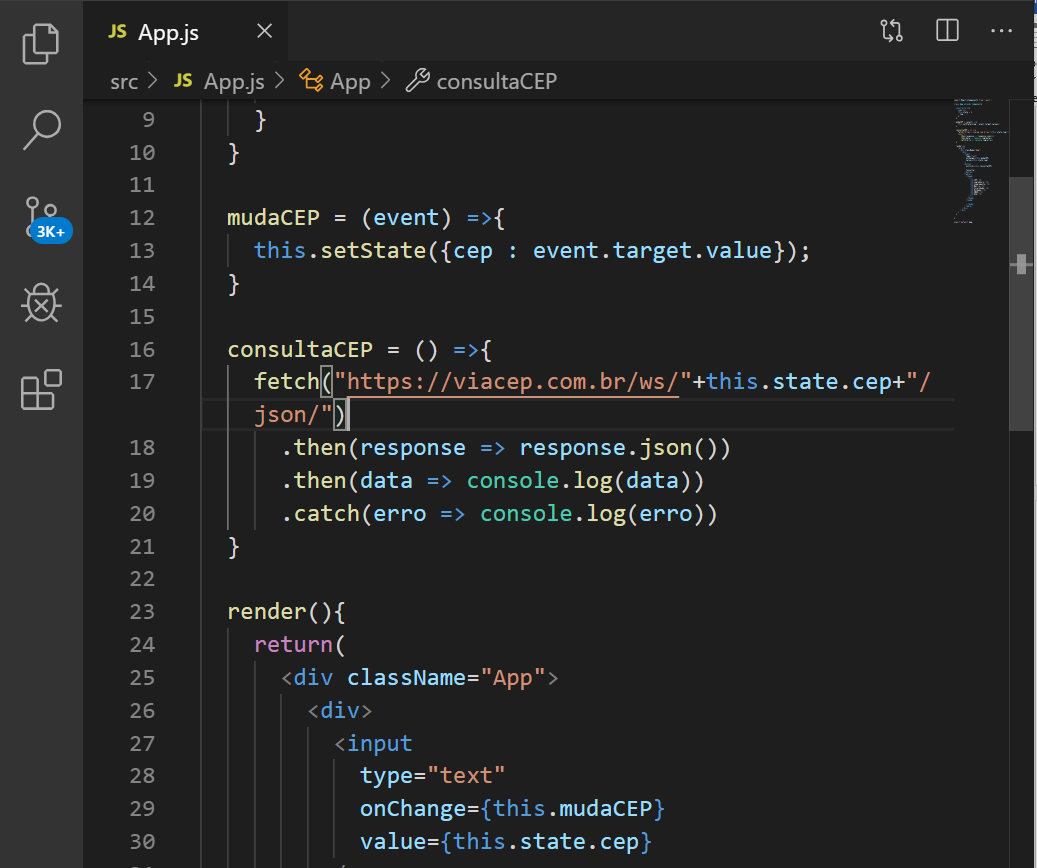
Precisaremos incluir então do lado do nosso input um botão



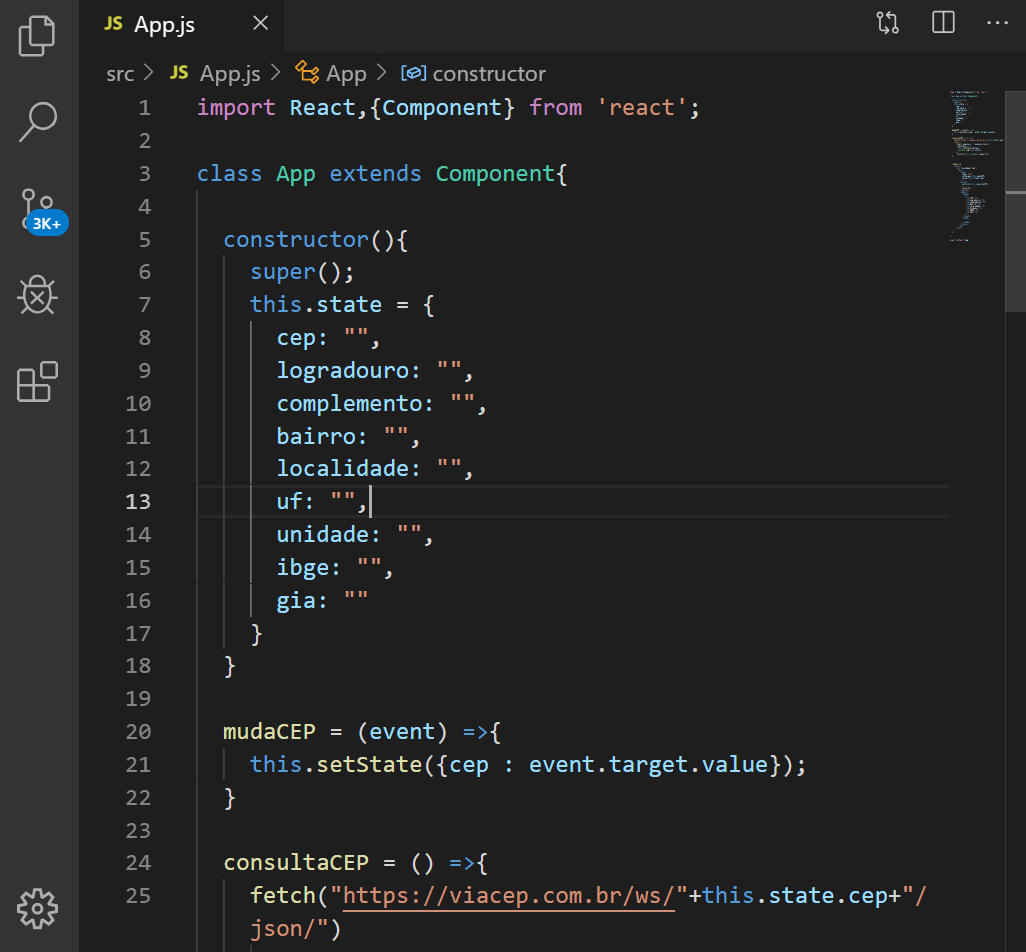
Nele iremos acionar o evento consultar ao click



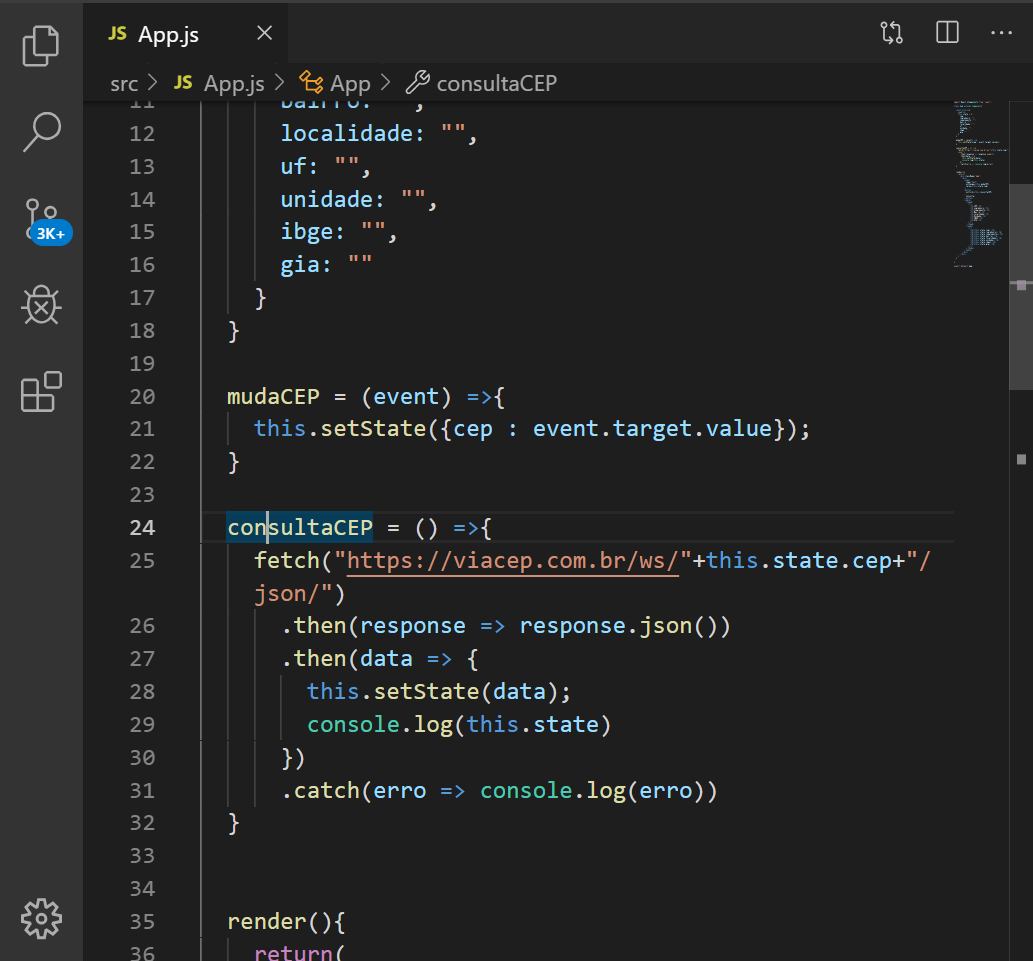
Nessa função realizaremos a requisição usando o valor do CEP armazenado em nosso state de forma dinâmica.



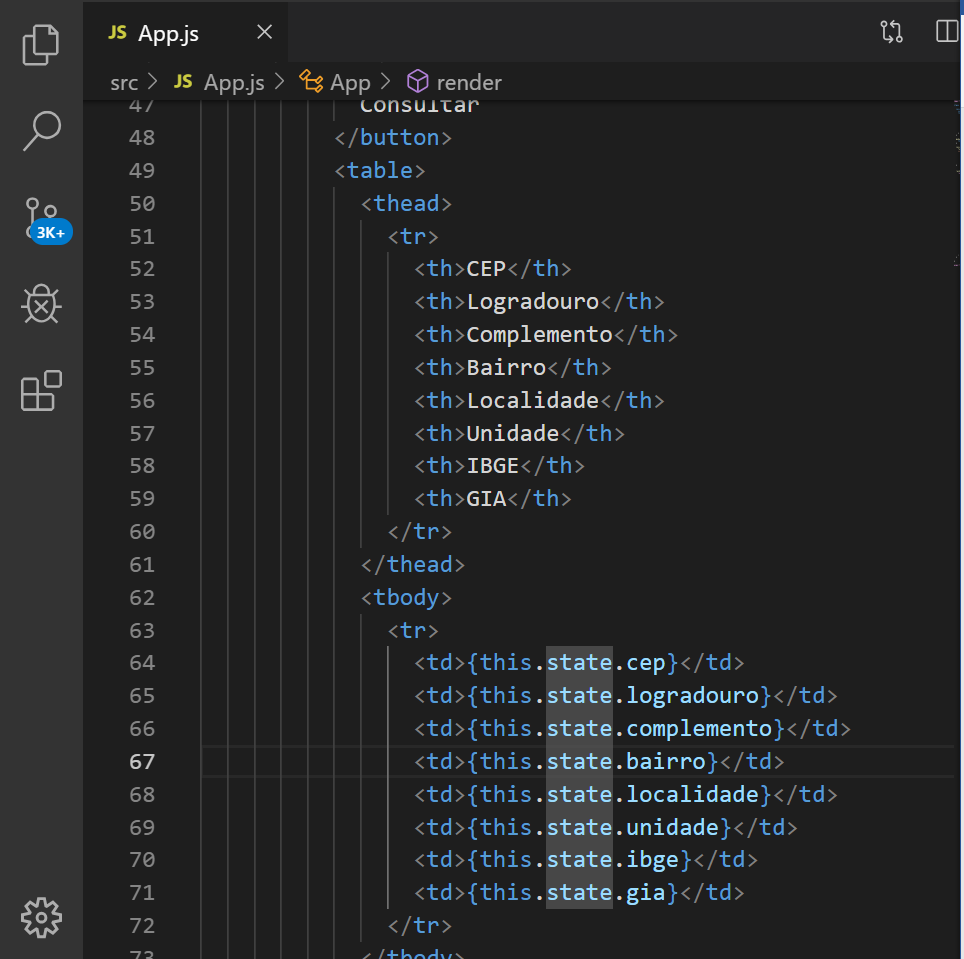
Sabendo qual meu retorno por meio do console.log() preparo meu state com todas as chaves que o json de resposta irá possuir

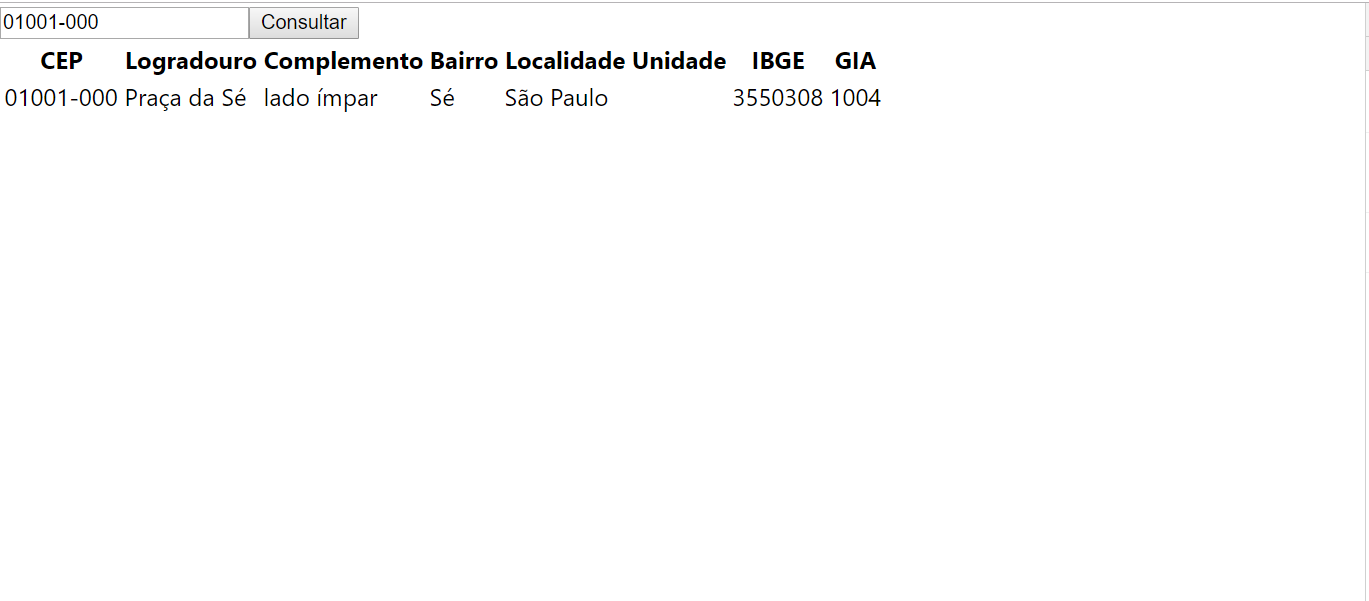


Dessa forma no 2º .then da requisição podemos realizar um setState() passando o nosso objeto data. As informações desse objeto irão preecher cada uma das chaves que existem no meu state.



Dentro de nosso tbody, criamos uma linha com 8 colunas (tds) na mesma ordem das th anteriormente definidas, contendo os valores das chaves existentes no nosso state





Dessa forma temos a estrutura da nossa aplicação viacep.