Dentro do Arquivo App.js crie os botoes para as requisiçoes pedidas

```
return (
<div>
     <button onClick={handleClick}>Testar Rota</button>
     <button onClick={handleParamClick}>Testar Rota com Parâmetro</button>
<button onClick={handleQueryClick}>Testar Rota com Query</button>
<div>
       {message && <p>{message}</p>} {/* Renderizar a mensagem recebida do backend, se
existir */}       </div>
   </div>
  );
```

Primeiro botao - Verificar se a rota executou com sucesso (Primeiro botao), voce vai clicar e vai chamar

```
          //Requisiçao DENTRO de App.js

 const handleClick = async () => {
try {
     const response = await axios.get('http://localhost:(NUMERO DA ROTA DA API EM
C#)/teste');//Armazena o resultado da requisicao, endereço da funçao na API/nome
     setMessage(response.data); //Exibe a mensagem armazenada na requisicao feita anteriormente
   } catch (error) {
     console.error(error); //Retorna no console se der erro e qual o erro
   }
  };


  //Função DENTRO de program.cs

  app.MapGet("/teste", () => //Funçao chamada teste que retorna que foi efetuado com sucesso a
execuçao da rota da api, app.MapGet pega uma informaçao
  {
    return Results.Ok("Rota executou com sucesso!"); //Aqui dentro voce efetua a operaçao e retorna
o resultado, no caso vai retornar que a rota foi executada   });
```

Segundo Botao - parâmetro nessa URL de requisição seja usado no back para gerar uma resposta diferente. Por exemplo, como fazer com que /teste/10 chamado a partir da sua
aplicação front-end a partir de um botão ou link retorne do back-end o texto "Rota executou com sucesso recebendo o valor 10!"

```
          //Requisiçao DENTRO DO App.js
            const handleQueryClick = async () => {
          try {
          const response = await axios.get('http://localhost:(NUMERO DA ROTA DA API EM
C#)/teste/10'); //mesmo esquema, a diferença é que aqui vai ser mandado parametro, ou seja valor.
E isso vai ser recebido la na funçao e retornar o resultado
```

setMessage(response.data); //Exibe a mensagem armazenada na requisicao feita anteriormente } catch (error) {
console.error(error);
}
};

//funçao DENTRO DE program.cs
app.MapGet("/teste/{valor:int}", (int valor) => //chamada da funçao recebendo o valor e indicando que vai passar um valor do tipo int na string
{ return Results.Ok($"Rota executou com sucesso recebendo o valor {valor}!"); //retorno da String com o resultado gerado });

Terceiro Botao - Como fazer com que um valor passado na query nessa URL de requisição seja usado no back para gerar uma resposta diferente. Por exemplo, como fazer com que /teste?valor=10&quantidade=3 chamado a partir da sua aplicação front-end a partir de um botão ou link retorne do back-end o texto "Rota executou com sucesso recebendo o valor 10 e quantidade 3!"

//Requisiçao DENTRO DO App.js

const handleQueryClick = async () => {
try { const response = await axios.get('http://localhost:(NUMERO DA ROTA DA API EM C#)teste/query? valor=10&quantidade=3'); //mesmo esquema novamente, a diferença é que aqui vai ser mandado 2 parametros, como valor e quantidade. E isso vai ser recebido la na funçao e retornar o resultado setMessage(response.data); //Exibe a mensagem armazenada na requisicao feita anteriormente } catch (error) {
console.error(error);
}
};

//funçao DENTRO DE program.cs
app.MapGet("/teste/query", (int valor, int quantidade) => //chamada da funçao recebendo os valores e indicando que vai passar valores do tipo int na string
{ return Results.Ok($"Rota executou com sucesso recebendo o valor {valor} e quantidade {quantidade}!"); //retorno da String com o resultado gerado });