# **PLANO DE AULA**

Aula 13 | Tempo estimado: 1 hora e 30 minutos | Web Starter

Tipo da atividade: Offline

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

Conteúdos

* CRUD e assincronismo.

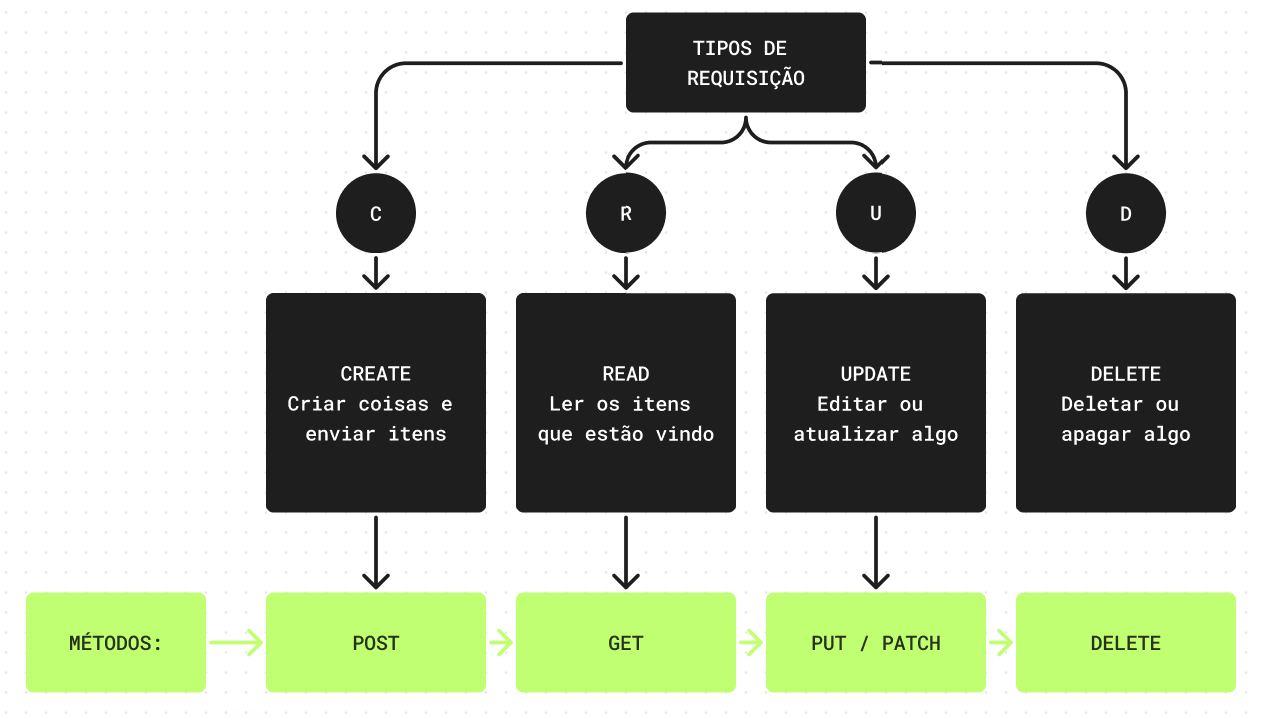
Objetivos

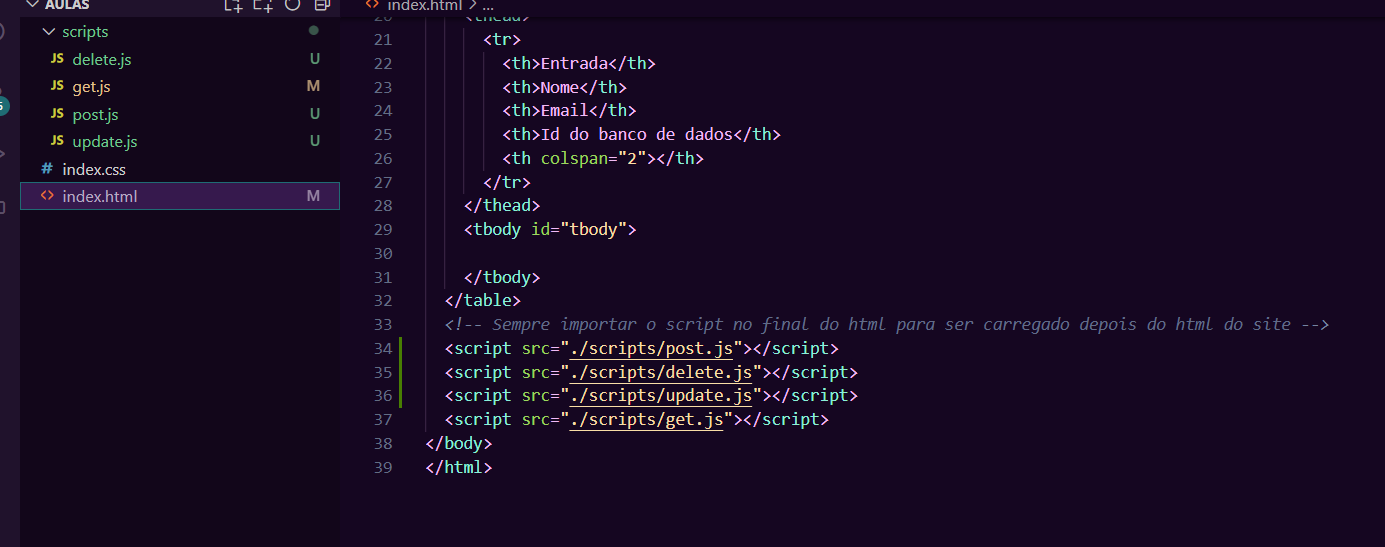
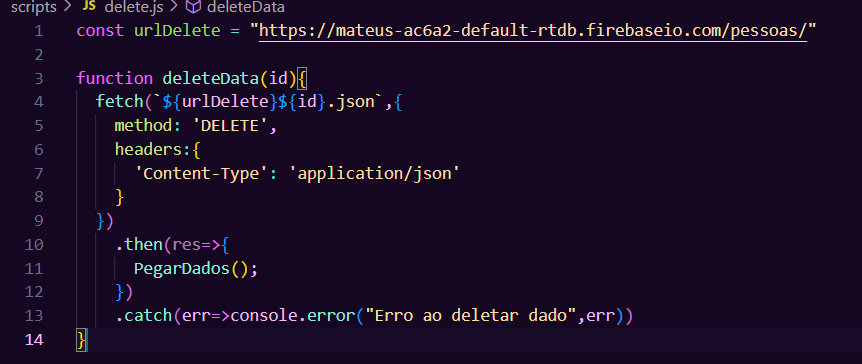
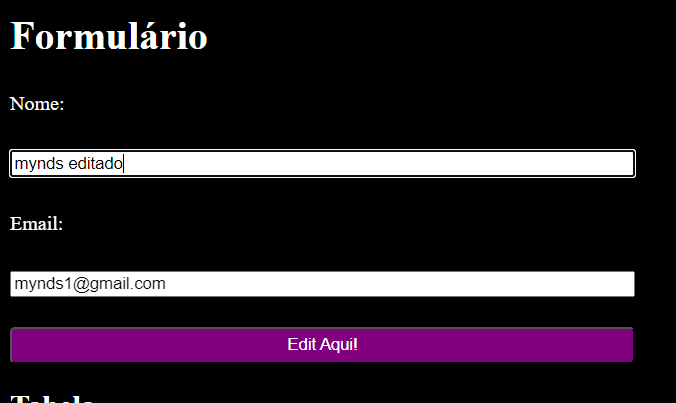
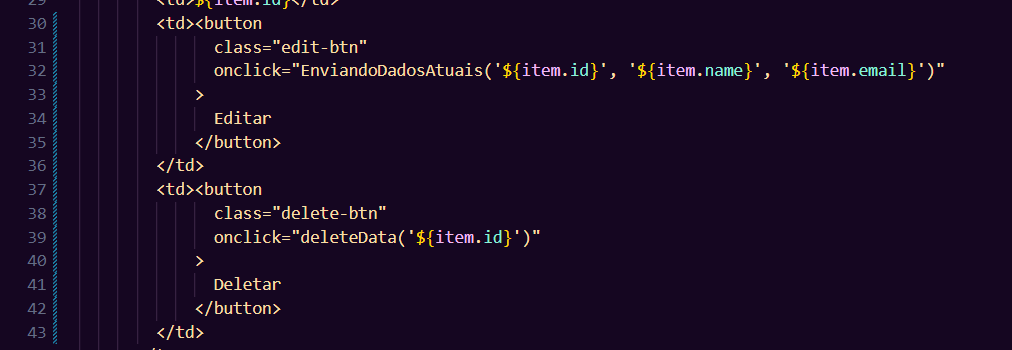
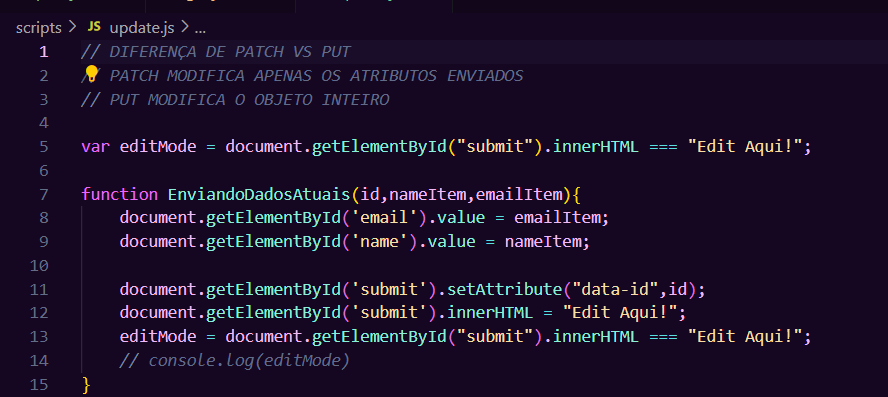
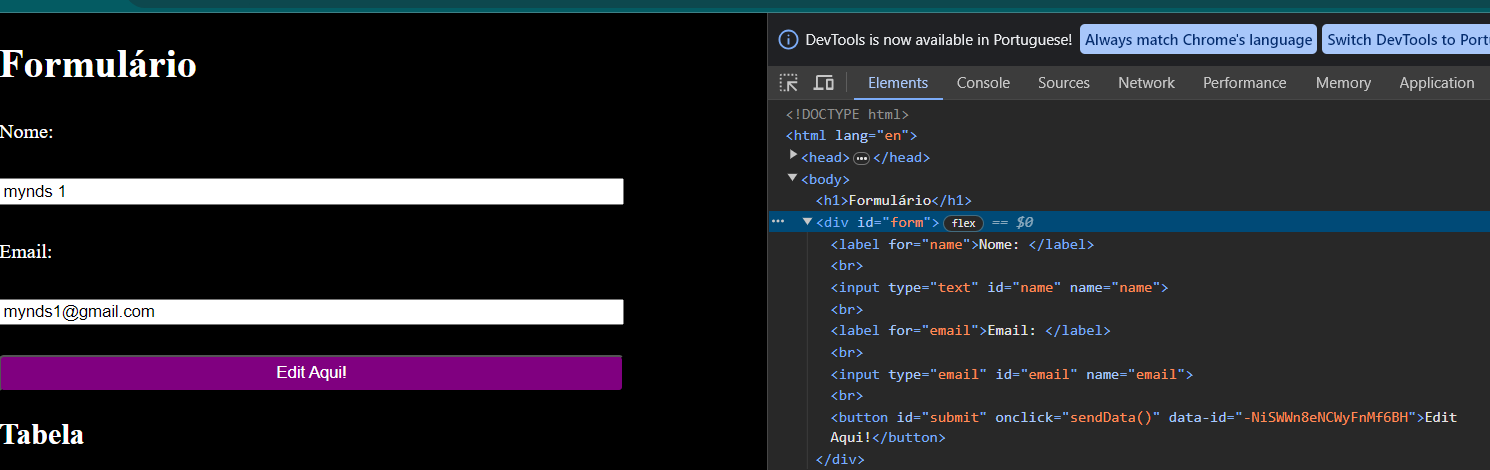
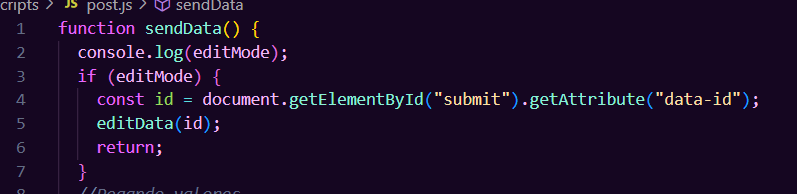
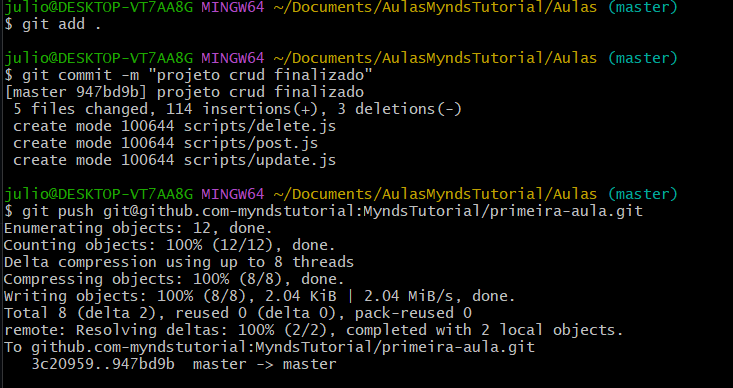
* Ensinar sobre o que é CRUD e como sincronizar os dados.

Estratégias e atividades

* Continuar o site da semana passada, criando agora uma tabela para ver as pessoas.
* O que é um banco de dados.
* Aprender sobre requisições.
* Entender o famoso CRUD.

1. Hoje vamos entender sobre as ações que é possível ter em um site. Existe o CRUD, onde são as 4 interações básicas do banco de dados:



1. Vamos criar os arquivos de *update delete* e *post* importando eles no index.html:  
   
2. Arquivo de *post*: nele vamos criar um objeto com os valores do *input* e enviar para o banco de dados. Depois disso, vamos formatar a resposta com o *.then()* e caso dê errado, ele irá para o *.catch()*. Formatando a resposta caso dê certo, vamos chamar a função *pegarDados* e atualizar a tabela:  
   
3. O *delete* tem uma URL um pouco diferente, pois o *.json* dela fica depois do ID do dado no banco que vamos criar. Caso dê certo, reiniciamos o “pegar dados”, caso dê errado, vamos colocar no console:  
   
4. Agora o editar funciona de dois modos. Existe duas “situações no editar”:
   1. A primeira é quando clicamos no botão de editar de algo:  
      
   2. A segunda é quando de fato enviamos para o banco clicando no botão de editar lá do formulário:  
      
5. Vamos entender o que acontece quando clicamos no botão editar de uma linha da tabela:
   1. Dentro do *get.js* se você reparar existe um novo formato de arquivos nos botões onde chamamos as funções do nosso código:
   2. A função *EnviandoDadosAtuais* é responsável lá no *update.js* por colocar o ID do banco dentro da tag do botão. E colocar os dados no *input* que quer editar:   
        
      Fazendo com que o formulário se altere, e dentro do botão exista uma meta-tag que salva o ID do banco:  
      
   3. Agora, quando editarmos, após clicar no botão “Edit Aqui!”, ele irá chamar a função *post* que verificará se está no *editMode*, caso esteja, ele irá chamar a função de *editar*:  
      
   4. E a função de editar irá trocar os dados no banco substituindo o objeto pelo novo e se der certo recarregará a tabela:  
      
6. 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:  
  

Tarefas

* Tentar clonar o projeto em casa e editar, seja o CSS, ou adicionar mais campos no formulário.