PROJETO INDIVIDUAL

Módulo 3 – Estão servidos?

RESILIA | Senac

CONTEXTO

O termo “Json-server” é utilizado para descrever um servidor (pode também ser correlacionado a um garçom) que serve dados no formato Json.

Neste projeto, você irá **desenvolver um servidor json (Json-server)** com 3 ou mais rotas, a entidade que será utilizada nas rotas precisa ter 4 ou mais atributos.

E atenção, o servidor criado neste projeto será utilizado no projeto em grupo!

O QUE É PARA FAZER?

Desenvolver um servidor json (Json-server) onde irá conter 3 rotas com 4+ dados nas quais o usuário poderá realizar o GET, POST, PUT, DELETE.

COMO FAZER?

Mecânica

- O projeto consiste em desenvolver um Json-server que contenha três rotas por exemplo (tarefas, projetos e notas), onde cada rota deve conter quatro ou mais dados já pré-preenchidos (mokados).
- Para a elaboração vocês devem escolher um tema (Lembre-se que o tema que for escolhido neste projeto poderá ser o mesmo para seu projeto final).

Temas

1. Escola
2. Restaurante
3. Farmácia
4. Agência de publicidade
5. Academia
6. Garagem de veículos
7. Loja de informática
8. Pub
9. Oficina mecânica
10. Hamburgueria
11. Dentista
12. Auto-escola
13. Transportadora
14. Estúdio de tatuagem
15. Hotel
16. Imobiliária
17. Livraria
18. Posto de Combustível
19. Salão de beleza
20. Cinema

COMO FAZER?

Requisitos

- Seu Json-server deve ser capaz de inserir (POST), acessar os dados (GET), Atualizar um dado (PUT) e excluir um dado (DELETE) nas três ou mais rotas disponíveis.

Para isso seu servidor Json deve conter:

- 3 ou mais rotas;
  - Cada uma das rotas deverá conter os métodos:
    - GET
    - POST
    - PUT
    - DELETE
  - Cada rota deverá conter 4 ou mais dados.
  - O Servidor deverá estar hospedado (deploy) no render.com ou em qualquer outro serviço de hospedagem.

EXTRAS

Requisitos Extras

- Rotas personalizadas para cada verbo/ação realizada.
  - Ex: Rota criada automática = “ POST /notas/” | Rota personalizada = “POST /notas/create”.
- Ter uma rota onde tem uma lista de itens/produtos em que cada item/produto contenha um link de foto.

COMO FAZER?

Dicas

- O json-server (npm) contém 2 abordagens, uma com os comandos por exemplo: *“json-server db.json --routes routes.json”* e a outra é a que vimos em aula configurando tudo em um arquivo .js. É recomendado que para não existir problemas na hora do deploy (hospedagem) você utilize a abordagem usando um arquivo .js.
- Os dados presentes em seu Json-server são identificados como único através de seu identificador “*id*” lembre-se de adicionar um “*id*” para cada item existente dentro do *“db.json”*.

COMO FAZER?

Documentação e referências

- Sobre json-server - Documentação: <https://github.com/typicode/json-server/#json-server->
- Tutorial JSON Server: <https://www.digitalocean.com/community/tutorials/json-server>
- Como criar um Json-server? <https://www.geeksforgeeks.org/how-to-create-a-rest-api-using-json-server-npm-package/>

F.A.Q.

Posso hospedar em outro site que não seja o render.com?

Pode sim! Contanto que a hospedagem permita você realizar as requisições da sua máquina local ex: realizar um fetch em algum projeto seu.

O que seria um dado mokado?

É um dado preenchido pelo programador, ou seja, você abrir o arquivo db.json manualmente e inserir lá alguns dados manualmente. Isso se caracteriza como uma simulação, ou seja você estará simulando dados (criando dados fictícios) dentro do servidor.

Irei utilizar meu projeto como complemento no projeto final?

Como este projeto é individual e o projeto final é em grupo, quando os grupos foram montados vocês se reuniram e definiram qual projeto seria utilizado dentre os desenvolvidos pelos integrantes do grupo.

Utilizei a abordagem por comandos e está funcionando na hospedagem, devo mudar para a outra?

Não é obrigatório é somente uma recomendação que utilize a abordagem por arquivo .js já que é mais customizável para se adequar a hospedagens.

MÃO NA MASSA

Momento **1** - Início

Comece identificando o que você precisará fazer para realizar o projeto como um todo e organize em lista!

Defina as prioridades de cada afazer de sua lista (algo útil é a matriz de Eisenhower).

Utilitário para o projeto

Como sugestão você pode começar com as atividades:
- Criar a pasta do projeto
- Iniciar o git na pasta Instalar o json-server no projeto com o npm

Lembre-se de sempre reservar de 5 a 10 minutos para realizar uma reflexão/reordenação dos itens já concluídos e os próximos a serem concluídos.

Salve e Commite todas as alterações realizadas no projeto.

MÃO NA MASSA

Momento **2** - Desenvolvimento

Continue progredindo nas tarefas que foram mapeadas, nesse momento você pode seguir realizando:

- Criar um .gitignore (gitignore.io) para projetos node
- Criar um arquivo .js e adicionar a alteração no git e commitar
- Criar o repositório no github/Conectar com o projeto local
- Realizar o “git push”

Lembre-se de sempre reservar de 5 a 10 minutos para realizar uma reflexão/reordenação dos itens já concluídos e os próximos a serem concluídos.

Salve e Commite todas as alterações realizadas no projeto.

MÃO NA MASSA

Momento 3 - Ajustes

Neste momento restam os ajustes necessários no arquivo db.json

Realize os ajustes criando os exemplos e preparando tudo para o funcionamento das rotas que estão configuradas.

Lembre-se de sempre reservar de 5 a 10 minutos para realizar uma reflexão/reordenação dos itens já concluídos e os próximos a serem concluídos.

MÃO NA MASSA

Momento 4 - Final de projeto

Hora de revisar os últimos detalhes!

Realize os últimos testes e verificações nessa etapa, nesse momento vamos preparar tudo para a publicação do projeto.

Lembre-se de testar todas as rotas para ver se tudo está funcionando.

Visualize o projeto depois do deploy para garantir que esteja tudo funcionando conforme o planejado e realize os ajustes necessários em caso de erro.

Envie o link do projeto pela plataforma!

RUBRICA

Conteúdo	Habilidades
Organização do Código	1. O projeto está organizado em pastas 2. Os arquivos JS/JSON estão separados em arquivos diferentes 3. Os arquivos JS/JSON então indentados e formatados. 4. O projeto não apresenta erros ao ser executado. 5. O projeto não apresenta erros ao ser executado no servidor de hospedagem(escolhido e hospedado pelo aluno(a, e)).
JavaScript /Json	1. O código executa corretamente. 2. Não são apresentados erros de sintaxe. 3. Foi utilizado um Array para cada rota e um Object para cada dado. 4. Os dados nos arquivos Json estão estruturados e contendo um identificador (id, _id) 5. O package.json contém o script de “start”.
Funcionalidades	1. O projeto contém 3 ou mais rotas. 2. Cada rota contém os métodos (GET, POST, PUT, DELETE). 3. Cada rota contém 4 ou mais dados mokados. 4. Todos os dados contém um identificador (id, _id) 5. O projeto foi hospedado e está funcionando na hospedagem.

RUBRICA

Conteúdo	Habilidades
Resolução de Problemas	1. O projeto possui todas as rotas pedidas (3 ou mais) 2. As rotas possuem todos os métodos pedidos. 3. Ao inserir um dado, o json-server atribui o identificador automático seguindo o padrão definido no projeto (id ou _id). 4. Não houve nenhum erro de cors ao testar o projeto hospedado. 5. Os dados inseridos ficam salvos ao atualizar o site da hospedagem.
Git/GitHub	1. Entregou o link do repositório no Github. 2. O código está completo/funcionando no Github. 3. Além do código, foi colocado um arquivo READ.me explicando do que se trata e como pode ser executado o projeto. 4. O código foi enviado em commits por etapas. 5. As descrições dos commits/PRs estão bem redigidas e apresentam bem as mudanças realizadas

Até a próxima e
#confianoprocesso

RESILIA | Senac