**Stage 4**

-**GitHub:**

\*O projeto do nodejs está aberto no GitHub.

\*Você pode tornar-se contribuidor de ferramentas importantes e ganhar experiência.

\*Deixe toda a linguagem do seu perfil e seus repositórios em inglês.

-**API:**

**\***Ludico:restaurante lasanha e peixe grelhado, o garçom vem até a mesa, anota as informações do pedido. Vai caminhar até a cozinha e levar o pedido até uma pessoa que lê os pedidos para o cozinheiro. Ele prepara a comida e deixa no local de saída para o garçom pegar e entregar.

\*Técnica: compra online de roupa, coloca no carrinho e faz o checkout/pagamento com cartão de outra empresa. Os dados são inseridos no mesmo site.

Mesa – site, Garçom – API, bloco de notas – documento JSON.

O site acessa a API, manda o JSON com as informações, a cozinha é o sistema do Mercado Pago. Processa os dados de pagamento e manda a resposta.

O site não tem acesso ao sistema do Mercado Pago, a API vem do próprio Mercado Pago e permite a comunicação entre os dois.

\*O GitHub possui uma API e nós podemos acessar e pedir dados.

+Nós vamos acessar as informações públicas do seu perfil.

Consumir API -> Sistema ir na API e fazer transação de dado, pessoa não dona. A proprietária da API, fornece dados.

Discover -> Curso de API com node.

\*api.github.com

Ao entrar em uma rota (links) de dentro da API, você vê um resultado parecido com object

JSON -> resposta da API, pode ser em JSON ou URL.

APIs podem ter conteúdo privado ou público.

user-url

“A student focused on web development: JavaScript, HTML, CSS, Git.”

-**JSON** (melhor amigo):

\*É uma forma de documento simples, leve para transacionar informações. Muito usado em APIs.

\*Sintaxe: escrito como object. Texto

Descrição gerada automaticamente

\*Fetch: comando, vai até a URL da API e traz as informações do JSON.

\*promise: recurso do JavaScript para pegar respostas de algo.

.then() -> entra aqui se o fetch funcionar.

\*Transforme em json.

-Vamos deixar o site carregar as informações e depois acessar os elementos no DOM com o JS. Precisamos que o site esteja carregado antes de acessar o DOM. Por isso JS vai ficar no final do body.

-Use nomes de funções e variáveis mais explicativas. No mercado você trabalha com outras pessoas, elas precisam entender o código.

-**Arrow functions**:

Forma contraída da função (igual no DOM).

Função anônima, sem nome.

1.Parâmetro => {

code

}

2.(param, param2) => {

code

}

3.() => {

code

}

-**GitHub Pages:**

-Senha: #teste