



<> Code

! Issues

🔗 Pull requests 55

▶ Actions

📁 Projects

📖 Wiki

🛡️ Security

🔗 master ▼



albertosca Update index.html ...

✓ on 7 Apr 2020 ⌚ 7

[View code](#)

README.md



Boas vindas ao repositório do projeto de Carrinho de Compras!

Você já usa o GitHub diariamente para desenvolver os exercícios, certo? Agora, para desenvolver os projetos, você deverá seguir as instruções a seguir. Fique atento a cada passo, e se tiver qualquer dúvida, nos envie por *Slack!* #vqv 🚀

Aqui você vai encontrar os detalhes de como estruturar o desenvolvimento do seu projeto a partir desse repositório, utilizando uma branch específica e um *Pull Request* para colocar seus códigos.

Instruções para entregar seu projeto:

ANTES DE COMEÇAR A DESENVOLVER:

1. Clone o repositório

- `git clone https://github.com/tryber/sd-03-block10-project-shopping-cart.git`

- Entre na pasta do repositório que você acabou de clonar:

- `cd sd-03-block10-project-shopping-cart`

2. Crie uma branch a partir da branch `master`

- Verifique que você está na branch `master`
 - Exemplo: `git branch`
- Se não estiver, mude para a branch `master`
 - Exemplo: `git checkout master`
- Agora, crie uma branch onde você vai guardar os `commits` do seu projeto
 - Você deve criar uma branch no seguinte formato: `nome-de-usuario-nome-do-projeto`
 - Exemplo: `git checkout -b joaozinho-project-shopping-cart`

4. Quando fizer mudanças, adicione-as ao `stage` do Git e faça um `commit`

- Verifique que as mudanças ainda não estão no `stage`
 - Exemplo: `git status` (devem aparecer listados os novos arquivos em vermelho)
- Adicione o novo arquivo ao `stage` do Git
 - Exemplo:
 - `git add .` (adicionando todas as mudanças - *que estavam em vermelho* - ao stage do Git)
 - `git status` (devem aparecer listados os arquivos em verde)
- Faça o `commit` inicial
 - Exemplo:
 - `git commit -m 'iniciando o projeto. VAMOS COM TUDO :rocket:'` (fazendo o primeiro commit)
 - `git status` (deve aparecer uma mensagem tipo *nothing to commit*)

5. Adicione a sua branch com o novo `commit` ao repositório remoto

- Usando o exemplo anterior: `git push -u origin joaozinho-project-shopping-cart`

6. Crie um novo `Pull Request` (PR)

- Vá até a página de *Pull Requests* do [repositório no GitHub](#)
- Clique no botão verde "New pull request"
- Clique na caixa de seleção "Compare" e escolha a sua branch **com atenção**
- Clique no botão verde "Create pull request"

- Adicione uma descrição para o *Pull Request*, um título claro que o identifique, e clique no botão verde "*Create pull request*"
- **Não se preocupe em preencher mais nada por enquanto!**
- Volte até a [página de Pull Requests do repositório](#) e confira que o seu *Pull Request* está criado

Entregáveis

Para entregar o seu projeto você deverá criar um Pull Request neste repositório. Este Pull Request deverá conter os arquivos `index.html`, `style.css` e `script.js`, que conterão seu código HTML, CSS e JavaScript, respectivamente. Você pode adicionar outros arquivos se julgar necessário. ⚠ É importante que seus arquivos tenham exatamente estes nomes! ⚠

Você pode adicionar outros arquivos se julgar necessário. Qualquer dúvida, procure a monitoria. Lembre-se que você pode consultar nosso conteúdo sobre Git & GitHub sempre que quiser!

Requisitos do projeto

A seguir, estão listados todos os requisitos do projeto. Leia-os atentamente e siga à risca o que for pedido. Em particular, **atente-se para os nomes de ids que alguns elementos de seu projeto devem possuir**. O não cumprimento de um requisito, total ou parcialmente, impactará em sua avaliação.

Os requisitos do seu projeto são avaliados automaticamente, sendo utilizada a resolução `1366 x 768` (1366 pixels de largura por 768 pixels de altura). Logo, recomenda-se desenvolver seu projeto usando a mesma resolução, via instalação [deste plugin](#) do `Chrome` para facilitar a configuração dessa resolução.

Você tem liberdade para adicionar novos comportamentos ao seu projeto, seja na forma de aperfeiçoamentos em requisitos propostos ou novas funcionalidades, **desde que tais comportamentos adicionais não conflitem com os requisitos propostos**. Em outras palavras, você pode fazer mais do que for pedido, mas nunca menos. Contudo, tenha em mente que **nada além do que for pedido nos requisitos será avaliado**. Esta é uma oportunidade de você exercitar sua criatividade e experimentar com os conhecimentos adquiridos.

Nesse projeto vocês farão um **carrinho de compras** totalmente dinâmico! E o melhor: consumindo dados diretamente de uma **API**! Isso mesmo. Da sigla em inglês *Application Programming Interface*, uma API é um ponto de contato na internet com determinado serviço. Através de **requisições HTTP** a essa API é possível interagir com ela da forma como quem a criou planejou. Aqui usaremos a API do Mercado Livre para buscarmos produtos à venda.

O [manual da API do Mercado Livre](#) contém muitas informações sobre ela. Utilizaremos alguns dos *endpoints*, e a forma de uso está detalhada no primeiro requisito. Este projeto tem como objetivo:

- Revisar seu conhecimento acerca de JavaScript, CSS e HTML;
- Checar o seu conhecimento acerca de JavaScript assíncrono através do uso da API do mercado livre.

Lembre-se de testar, usando Jest, quaisquer funções que criar para comportar a lógica da sua aplicação. Isso é um treinamento muito importante para o próximo trabalho!

Seu projeto só será avaliado se estiver passando pelos *checks* do **CodeClimate** e do **TravisCI**.

1. Listagem de produtos

Você deve criar uma listagem de produtos que devem ser consultados através da API do Mercado Livre.

Você deve utilizar o *endpoint*:

```
"https://api.mercadolibre.com/sites/MLB/search?q=$QUERY"
```

onde `$QUERY` deve ser o valor da sua busca. Para este trabalho, a busca deve ser o termo `computador`.

O retorno desse *endpoint* será algo no formato json. Por exemplo, se for pesquisado "computador":

```
{
  "site_id": "MLB",
  "query": "computador",
  "paging": {
    "total": 406861,
    "offset": 0,
    "limit": 50,
```

```
    "primary_results": 1001
  },
  "results": [
    {
      "id": "MLB1341925291",
      "site_id": "MLB",
      "title": "Processador Intel Core I5-9400f 6 Núcleos 128 Gb",
      "seller": {
        "id": 385471334,
        "permalink": null,
        "power_seller_status": null,
        "car_dealer": false,
        "real_estate_agency": false,
        "tags": []
      },
      "price": 899,
      "currency_id": "BRL",
      "available_quantity": 1,
      "sold_quantity": 0,
      "buying_mode": "buy_it_now",
      "listing_type_id": "gold_pro",
      "stop_time": "2039-10-10T04:00:00.000Z",
      "condition": "new",
      "permalink": "https://www.mercadolivre.com.br/processador-intel-c",
      "thumbnail": "http://mlb-s2-p.mlstatic.com/813265-MLA32241773956_",
      "accepts_mercadopago": true,
      "installments": {
        "quantity": 12,
        "amount": 74.92,
        "rate": 0,
        "currency_id": "BRL"
      },
      "address": {
        "state_id": "BR-SP",
        "state_name": "São Paulo",
        "city_id": "BR-SP-27",
        "city_name": "São José dos Campos"
      },
      "shipping": {
        "free_shipping": true,
        "mode": "me2",
        "tags": [
          "fulfillment",
          "mandatory_free_shipping"
        ],
        "logistic_type": "fulfillment",
        "store_pick_up": false
      },
      "seller_address": {
        "id": "",
        "comment": ""
      }
    }
  ]
}
```

```
"address_line": "",
"zip_code": "",
"country": {
  "id": "BR",
  "name": "Brasil"
},
"state": {
  "id": "BR-SP",
  "name": "São Paulo"
},
"city": {
  "id": "BR-SP-27",
  "name": "São José dos Campos"
},
"latitude": "",
"longitude": ""
},
"attributes": [
  {
    "source": 1,
    "id": "ALPHANUMERIC_MODEL",
    "value_id": "6382478",
    "value_struct": null,
    "values": [
      {
        "name": "BX80684I59400F",
        "struct": null,
        "source": 1,
        "id": "6382478"
      }
    ],
    "attribute_group_id": "OTHERS",
    "name": "Modelo alfanumérico",
    "value_name": "BX80684I59400F",
    "attribute_group_name": "Outros"
  },
  {
    "id": "BRAND",
    "value_struct": null,
    "attribute_group_name": "Outros",
    "attribute_group_id": "OTHERS",
    "source": 1,
    "name": "Marca",
    "value_id": "15617",
    "value_name": "Intel",
    "values": [
      {
        "id": "15617",
        "name": "Intel",
        "struct": null,
        "source": 1
      }
    ]
  }
]
```

```

    }
  ]
},
{
  "name": "Condição do item",
  "value_id": "2230284",
  "attribute_group_id": "OTHERS",
  "attribute_group_name": "Outros",
  "source": 1,
  "id": "ITEM_CONDITION",
  "value_name": "Novo",
  "value_struct": null,
  "values": [
    {
      "id": "2230284",
      "name": "Novo",
      "struct": null,
      "source": 1
    }
  ]
},
{
  "id": "LINE",
  "value_name": "Core i5",
  "attribute_group_id": "OTHERS",
  "attribute_group_name": "Outros",
  "name": "Linha",
  "value_id": "7769178",
  "value_struct": null,
  "values": [
    {
      "id": "7769178",
      "name": "Core i5",
      "struct": null,
      "source": 1
    }
  ],
  "source": 1
},
{
  "id": "MODEL",
  "value_struct": null,
  "values": [
    {
      "id": "6637008",
      "name": "i5-9400F",
      "struct": null,
      "source": 1
    }
  ],
  "attribute_group_id": "OTHERS",

```

```

        "name": "Modelo",
        "value_id": "6637008",
        "value_name": "i5-9400F",
        "attribute_group_name": "Outros",
        "source": 1
    },
    ],
    "differential_pricing": {
        "id": 33580182
    },
    "original_price": null,
    "category_id": "MLB1693",
    "official_store_id": null,
    "catalog_product_id": "MLB13953199",
    "tags": [
        "brand_verified",
        "good_quality_picture",
        "good_quality_thumbnail",
        "immediate_payment",
        "cart_eligible"
    ],
    "catalog_listing": true
},
]
}

```

A lista de produtos que devem ser exibidos é o *array* `results` no `JSON` acima.

Você **deve** utilizar a função `createProductItemElement(product)` para criar os componentes *HTML* referentes a um produto.

Adicione o elemento retornado da função `createProductItemElement(product)` como filho do elemento `<section class="items">`.

Obs: as variáveis `sku`, no código fornecido, se referem aos campos `id` retornados pela API.

2. Adicione o produto ao carrinho de compras

Cada produto na página *HTML* possui um botão com o nome `Adicionar ao carrinho!`.

Ao clicar nesse botão você deve realizar uma requisição para o *endpoint*:

```
"https://api.mercadolibre.com/items/$ItemID"
```

onde `$ItemID` deve ser o valor `id` do item selecionado.

Quando colocado o id `MLB1341706310` retorno desse *endpoint* será algo no formato:

```
{
  "id": "MLB1341706310",
  "site_id": "MLB",
  "title": "Processador Amd Ryzen 5 2600 6 Núcleos 64 Gb",
  "subtitle": null,
  "seller_id": 245718870,
  "category_id": "MLB1693",
  "official_store_id": 1929,
  "price": 879,
  "base_price": 879,
  "original_price": null,
  "currency_id": "BRL",
  "initial_quantity": 0,
  "available_quantity": 0,
  "sold_quantity": 0,
  "...",
  "warranty": "Garantia de fábrica: 3 anos",
  "catalog_product_id": "MLB9196241",
  "domain_id": "MLB-COMPUTER_PROCESSORS",
  "parent_item_id": null,
  "differential_pricing": null,
  "deal_ids": [],
  "automatic_relist": false,
  "date_created": "2019-10-15T18:13:00.000Z",
  "last_updated": "2019-12-20T18:06:54.000Z",
  "health": null,
  "catalog_listing": true
}
```

Preste atenção que o JSON deve conter apenas **um** item.

Você **deve** utilizar a função `createCartItemElement()` para criar os componentes *HTML* referentes a um item do carrinho.

Adicione o elemento retornado da função `createCartItemElement(product)` como filho do elemento `<ol class="cart__items">`.

3. Remova o item do carrinho de compras ao clicar nele

Ao clicar no **produto no carrinho de compra**, ele deve ser removido da lista. Para isso, uma função (já existente) chamada `cartItemClickListener(event)` deve ser implementada com a lógica necessária para realizar a remoção.

4. Carregue o carrinho de compras através do LocalStorage ao iniciar a página

Ao carregar a página, o estado atual do carrinho de compras deve ser carregado do **LocalStorage**. Para que isso funcione, o carrinho de compras deve ser salvo no **LocalStorage**, ou seja, todas as **adições** e **remoções** devem ser abordadas para que a lista atual seja salva.

5. Some o valor total dos itens do carrinho de compras de forma assíncrona

Cada vez que se adicionar um item ao carrinho de compras será necessário somar seus valores e apresentá-los na página principal do projeto. Não queremos que essa soma, no entanto, impacte no carregamento da página. Devemos, portanto, fazer essa soma de forma *assíncrona*. Use `async/await` para fazer isso. O elemento que tem como filho o preço total dos itens do carrinho deve ter, **obrigatoriamente**, a classe `total-price`.



6. Botão para limpar carrinho de compras

Crie um botão para remover todos os itens do carrinho de compras. Ele deve, **obrigatoriamente**, ter a classe `empty-cart`.

7. Adicionar um texto de "loading" durante uma requisição à API

Uma requisição à API gasta um tempo e durante ele, ficamos sem saber se está tudo certo ou se algo deu errado. Normalmente é utilizada alguma forma para mostrar que a requisição está em andamento. Mostre a palavra "loading..." em alguma lugar da página **apenas durante** a requisição à API. O elemento mostrado durante o carregamento da página deve, **obrigatoriamente**, ter a classe `loading`.

DURANTE O DESENVOLVIMENTO

-  **PULL REQUESTS COM ISSUES NO CODE CLIMATE NÃO SERÃO AVALIADAS, ATENTE-SE PARA RESOLVÊ-LAS ANTES DE FINALIZAR O DESENVOLVIMENTO!** 
- Faça `commits` das alterações que você fizer no código regularmente
- Lembre-se de sempre após um (ou alguns) `commits` atualizar o repositório remoto
- Os comandos que você utilizará com mais frequência são:

- i. `git status` (para verificar o que está em vermelho - fora do stage - e o que está em verde - no stage)
- ii. `git add` (para adicionar arquivos ao stage do Git)
- iii. `git commit` (para criar um commit com os arquivos que estão no stage do Git)
- iv. `git push -u nome-da-branch` (para enviar o commit para o repositório remoto na primeira vez que fizer o `push` de uma nova branch)
- v. `git push` (para enviar o commit para o repositório remoto após o passo anterior)

DEPOIS DE TERMINAR O DESENVOLVIMENTO (OPCIONAL)

Para "entregar" seu projeto, siga os passos a seguir:

- Vá até a página **DO SEU Pull Request**, adicione a label de "code-review" e marque seus colegas
 - No menu à direita, clique no link "**Labels**" e escolha a label **code-review**
 - No menu à direita, clique no link "**Assignees**" e escolha **o seu usuário**
 - No menu à direita, clique no link "**Reviewers**" e digite `students`, selecione o time `tryber/students-sd-03`

Se ainda houver alguma dúvida sobre como entregar seu projeto, [aqui tem um video explicativo](#).

⚠ Lembre-se que garantir que todas as *issues* comentadas pelo CodeClimate estão resolvidas! ⚠

REVISANDO UM PULL REQUEST



À medida que você e as outras pessoas que estudam na Trybe forem entregando os projetos, vocês receberão um alerta via Slack para também fazer a revisão dos Pull Requests dos seus colegas. Fiquem atentos às mensagens do "Pull Reminders" no Slack!

Use o material que você já viu sobre [Code Review](#) para te ajudar a revisar os projetos que chegaram para você.

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Contributors 2



jeanpsv Jean Paulo Silva Vasconcelos



albertosca

Languages

