|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Natureza da Avaliação | Valores | | Disciplina: Programação Web | Curso | Identificação da Turma | | |
| Valor | Nota | Turno | Período | Turma |
| Passo a Passo |  | - | Professor(a): Priscila Nunes | SIS INF | Noturno |  |  |

# **Aluno(a):** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Data: \_\_\_/\_\_\_/\_\_\_\_

1. **Dashboard**

O **dashboard** é a área da nossa aplicação que utilizamos para condensar informações sobre diversas partes do sistema em um único lugar, assim facilitando sua visualização.

Para desenvolver esse layout utilizaremos vários componentes do Bootstrap, como Navbar, Panel e o seu sistema de grid. Além disso criaremos um menu retrátil utilizando CSS e jQuery.

1. **Estrutura do Projeto**

A estrutura de diretórios do projeto, que é composta por **index.html** e três pastas: **css**, **img** e **js**.

O index.html é o arquivo principal do projeto, onde será montado o HTML;

A pasta **CSS** conterá os arquivos **CSS** que utilizaremos no projeto: ***bootstrap.min.css*** e ***style.css***

A pasta **IMG** conterá as imagens que utilizaremos no projeto;

A pasta **JS** conterá os arquivos **JavaScript** que utilizaremos no projeto: ***bootstrap.min.js***, ***jquery-3.2.1.min.js*** e ***script.js***.

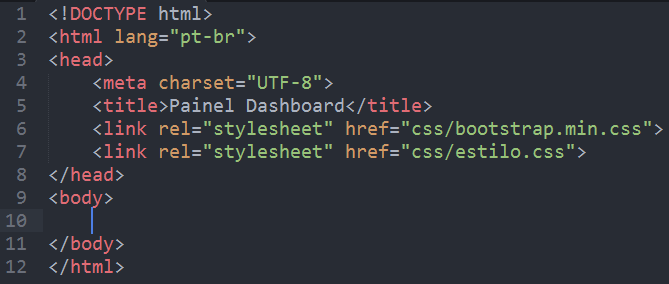
A estrutura do layout será inicialmente dividida em duas partes: topo e corpo.

|  |
| --- |
|  |

Veja que o topo do layout que é dividido em duas partes: o banner e a barra de navegação. Além disso, o corpo que é composto de cinco seções, cada uma focada em área do layout.

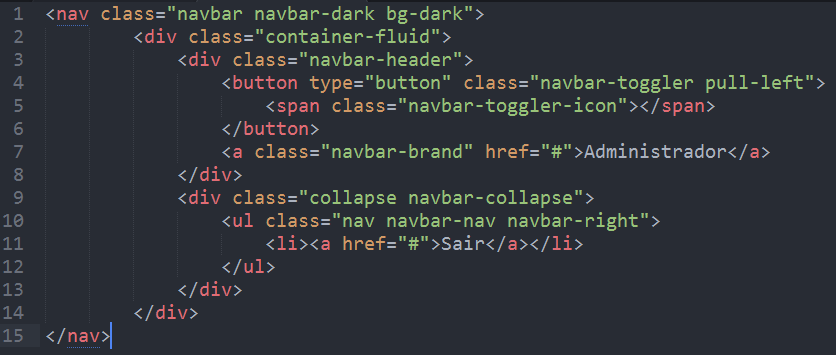
1. **Estrutura do HTML**

No cabeçalho da página referenciamos as folhas de estilos que iremos utilizar no nosso exemplo:



* **Linha 6**: inserimos a folha de estilo do Bootstrap;
* **Linha 7**: inserimos a nossa folha de estilo.

E para a criar o topo utilizaremos o componente **navbar** do **Bootstrap**, com o seguinte código dentro do **<body>**:

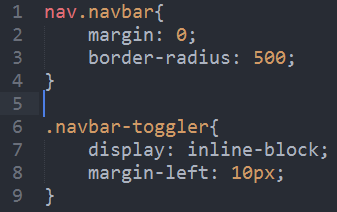


* **Linha 1**: utilizamos o elemento <nav> com a classe navbar, para criar o componente navbar. Ainda nessa linha adicionamos classe navbar-dark para definir o tema da navbar, onde navbar-dark é um tema escuro e navbar-light o tema padrão;
* **Linha 2**: utilizamos a classe container-fluid para criar uma área responsiva e que ocupe a tela inteira;
* **Linha 3**: adicionamos uma <div> com a classe navbar-header, que serve para guardar links, dropdown, formulários de busca e etc;
* **Linha 4**: criamos o botão que utilizamos para abrir o menu lateral. Ele possui a classe navbar-toggler, que define a aparência do botão e a utilizaremos para capturar o evento de clique. Ainda nessa linha, adicionamos a classe pull-left, que coloca o elemento no lado esquerdo do elemento navbar-header;
* **Linha 7**: adicionamos um link com a classe .navbar-brand que estiliza o título de nossa aplicação;
* **Linhas 09 a 13**: adicionamos uma <div> com a classe navbar-collapse, que serve para ocultar o elemento em resoluções de dispositivos mobile.
* **Linha 10**: criamos uma lista de links. A classe nav serve como base de uma lista de links em uma barra de navegação e a classe navbar-nav define a aparência da lista na navbar. E por último, a classe ml-auto coloca o elemento na ponta direita da navbar.

No final do **<body>**, referenciamos os scripts que iremos utilizar no nosso exemplo:

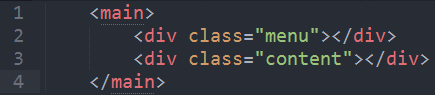


No arquivo estilo.css, iremos adaptar a aparência da navbar ao nosso layout, adicionando o bloco abaixo:



* **Linhas 1 a 4:** removemos a margem padrão da navbar e as bordas arredondadas.
* **Linhas 5 a 8:** deixamos o botão do menu visível e adicionamos uma margem de 10px a esquerda do botão.

**Abaixo da tag </nav> adicionamos uma <div> com a classe menu, que será o menu, e uma outra <div> com a classe content onde construiremos o conteúdo do dashboard.**



* **Linha 1**: O elemento **<main>** define o conteúdo principal dentro do [<body>](https://developer.mozilla.org/pt-BR/docs/Web/HTML/Element/body) em seu documento ou aplicação. Entende-se como conteúdo principal aquele relacionado diretamente com o tópico central da página ou com a funcionalidade central da aplicação;

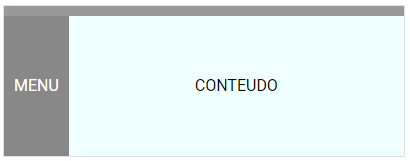
**Utilizando o bloco abaixo, iremos criar a estrutura do menu utilizando o elemento <ul> para formar uma lista de links:**

****

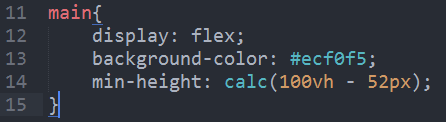
* **Linha 4:** utilizamos a classe visible-xs no elemento <li> contendo o link sair, essa classe serve para exibir o elemento somente em resoluções de dispositivos mobile. Assim, quando o link sair da navbar for ocultado, o link sair do menu será exibido.
* **Linha 5:** utilizamos a classe active para marcar no menu o link a qual essa página pertence.

**Arquivo estilo.css**

Aqui iremos definir a aparência da tag main e das classes content e menu, definindo o tamanho das áreas ocupadas pelo **menu** e pelo **conteúdo** no dashboard. Assim como o comportamento quando o menu estiver **aberto ou fechado**:

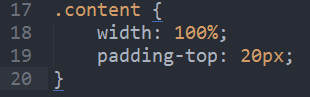
****

Adicione o código abaixo ao seu css:



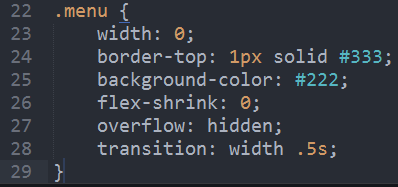
* **Linha 12:** utilizamos a propriedade display: flex, que fará com que o espaço interno do main seja dividido pela quantidade de elementos dentro dele;
* **Linha 13:** definimos uma cor de fundo utilizando a propriedade background-color;
* **Linha 14:** calculamos uma altura mínima para o elemento de 100vh - 52px, onde 100vh significa 100 por cento da altura da tela e 52px é a altura da navbar. Assim, o elemento com a classe main sempre irá preencher o restante da tela disponível.

No bloco baixo iremos definir a aparência do **conteúdo** do dashboard:



* **Linha 18**: definimos a largura do elemento para utilizar toda a largura disponível;
* **Linha 19**: acionamos um padding-top de 20px para criar um espaçamento interno no elemento.

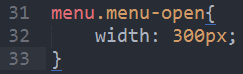
**Para estilizar o menu do dashboard usaremos o seguinte código:**



* **Linha 2**3: definimos a largura inicial do menu com o valor 0;
* **Linhas 24 e 25:** definimos a aparência base do menu;
* **Linha 26:** utilizamos a propriedade flex-shrink com o valor 0 para impedir que o container flex, o main, sobreponha o tamanho mínimo definido pelo elemento;
* **Linha 27:** utilizamos a propriedade overflow:hidden para que só seja exibida a parte do conteúdo que esteja dentro da área visível do menu;
* **Linha 28:** adicionamos a propriedade transition: width .5s, permitindo que a transição do valor da propriedade width ocorra suavemente no período de 0.5 segundos.

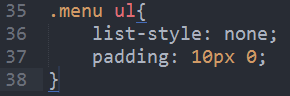
**No bloco abaixo estamos definindo a aparência do menu quando ele estiver aberto:**

Os arquivos possuem dependência um do outro, então devem ser carregados nesta exata ordem.



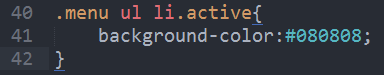
* **Linhas 32:** definimos a largura do menu para 300px, exibindo o conteúdo do menu quando ele tiver a classe menu-open.

**Estamos definindo o estilo da lista de links utilizando o bloco abaixo:**



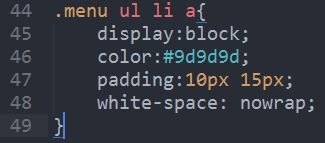
* **Linhas 36 e 37:** sobrepomos a aparência padrão da lista, removendo os bullets e modificando o padding.

**E para identificar quando um item da lista de links estiver ativo, utilizamos o bloco abaixo:**



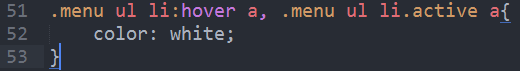
* **Linha 41:** definimos um background-color com a cor em hexadecimal #080808, realçando o item da lista com a classe active.

**Para definir a**aparência padrão**dos links da lista, utilizaremos o código abaixo:**



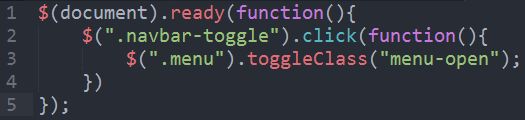
* **Linhas 45 a 48:** definimos aparência de cada link do menu;
* **Linha 48:** utilizamos a propriedade white-space: nowrap para evitar que as palavras se quebrem durante a animação de abrir ou fechar do menu.

**E por último iremos destacar o link do menu quando o cursor do mouse estiver sobre o item da lista ou**ativo**:**



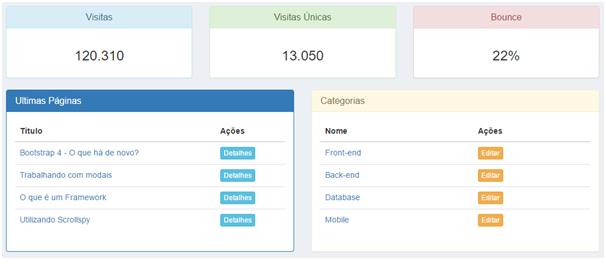
* No bloco acima definimos uma cor para o link quando passarmos o cursor do mouse sobre um item do menu ou quando ele estiver com a classe active.

Com a estrutura e a aparência do menu pronta, agora precisamos implementar a ação de abrir e fechar o menu. Para isso utilizaremos o jQuery para capturar o evento de clique do botão na navbar.



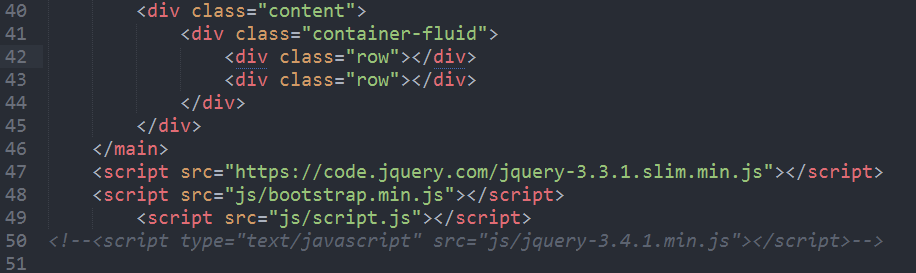
* **Linha 1**: esperamos que o documento esteja totalmente carregado utilizando a função ready do **jQuery**;
* **Linha 2**: utilizamos a função click para esperar o evento de clique no elemento com a classe .navbar-toggler, no caso o botão do **menu**, seja disparado;
* **Linha 3**: após o evento de clique ser disparado, utilizamos a função toggleClass para **adicionar** ou **remover** a classe .menu-open do menu.

**Por fim, construiremos os elementos do**dashboard**. Esse será composto por cinco**panels**utilizando o sistema de grid do**Bootstrap**, separando-os em colunas e linhas, como mostra a Figura 1.**



**Figura 1**. Sistema de grid do Bootstrap

Dentro do content,iremos utilizar o sistema de grid do **Bootstrap**, criaremos **duas linhas** onde iremos inserir os **panels** do dashboard:



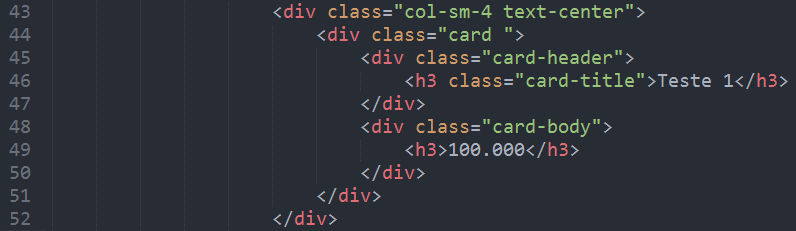
* **Linhas 41 a 44**: adicionamos um elemento com a classe container-fluid, e dentro dele dois elementos com a classe row, definindo duas linhas no sistema de grid do Bootstrap.

Para utilizar o sistema de grid do **Bootstrap** é necessário utilizar um elemento com a classe container ou container-fluid. No nosso dashboard iremos utilizar a classe container-fluid que usa toda a largura do elemento disponível, como mostra a **Figura 2**.

****

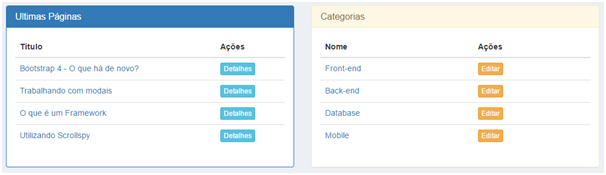
**Figura 2**. Uso da classe container-fluid

Dentro da **primeira linha** adicionamos os três primeiros **panels** do dashboard, e cada **panel** irá seguir o seguinte formato:



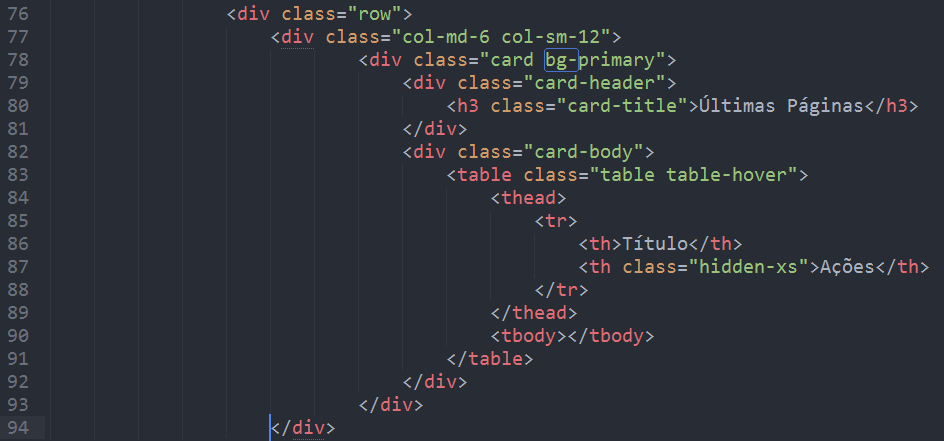
* **Linha 1**: adicionamos uma div com a classe col-sm-4, que no sistema de grid do **Bootstrap** significa que este elemento irá utilizar **4 das 12 colunas** disponíveis, enquanto a tela for **maior ou igual a 768px**. Caso a tela seja menor, ele ocupará o espaço das **12 coluna**s. E com a classe text-center alinhamos o texto ao centro do elemento;
* **Linha 2**: criamos uma **<div>** com a classe card;
* **Linha 3**: criamos o cabeçalho do **card** com a classe .card-header;
* **Linha 4**: definimos o título do **card** utilizando um **<h3>** com a classe card-title;
* **Linha 6**: criamos o corpo do **panel** com a classe card-body;
* Os outros **cards** utilizam essa mesma estrutura, só trocando o título e body.

Dentro da **segunda linha** adicionamos os dois últimos **cards** do dashboard, como mostra a **Figura 3**.



**Figura 3**. Dois últimos panels

Cada um dos **cards** possuem a mesma estrutura do bloco abaixo:



* **Linha 1**: adicionamos uma **<div>** com a classe col-md-6, que no sistema de grid do **Bootstrap** significa que este elemento irá utilizar **6 das 12 colunas** disponíveis, enquanto a tela for **maior ou igual a 992px**. Caso a tela seja menor, ele ocupará o espaço das **12 colunas**;
* **Linha 7**: criamos uma tabela com a classe table, que define sua aparência básica, modificando o padding e deixando somente as **divisões horizontais**. Ainda nessa linha adicionamos a classe .table-hover, que destaca a linha da tabela quando passamos o mouse por cima;
* **Linha 11**: adicionamos a classe .hidden-xs para ocultar a coluna em resoluções de dispositivos mobile.

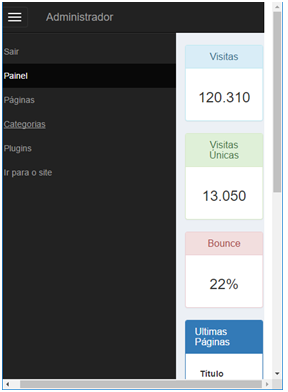
Dentro do tbody adicionamos as linhas da tabela, seguindo esse formato:



* **Linha 5**: adicionamos a classe hidden-xs para ocultar a coluna em resoluções de dispositivos mobile;
* **Linha 6**: criamos um link com a classe btn, assim dando a aparência de um botão, com a classe btn-xs diminuímos o tamanho dele, e por último, definimos uma cor com a classe btn-info.

O outro **panel** segue a mesma estrutura, modificando apenas a classe de contexto.

As colunas do sistema de grid do **Bootstrap** possuem um tamanho mínimo, então ao reduzir o tamanho da tela e abrir o menu, o **conteúdo** será empurrado para o lado criando um **scroll horizontal** na página, como mostra a **Figura 4**.



**Figura 4**. Abertura do Menu

Para resolver isso, basta adicionar a propriedade overflow: hidden ao bloco main no arquivo **estilo.css**.