Módulo I - Introdução, Sistemas e Aplicações

Tópico 7 - Tipos de aplicação Web

StartSe

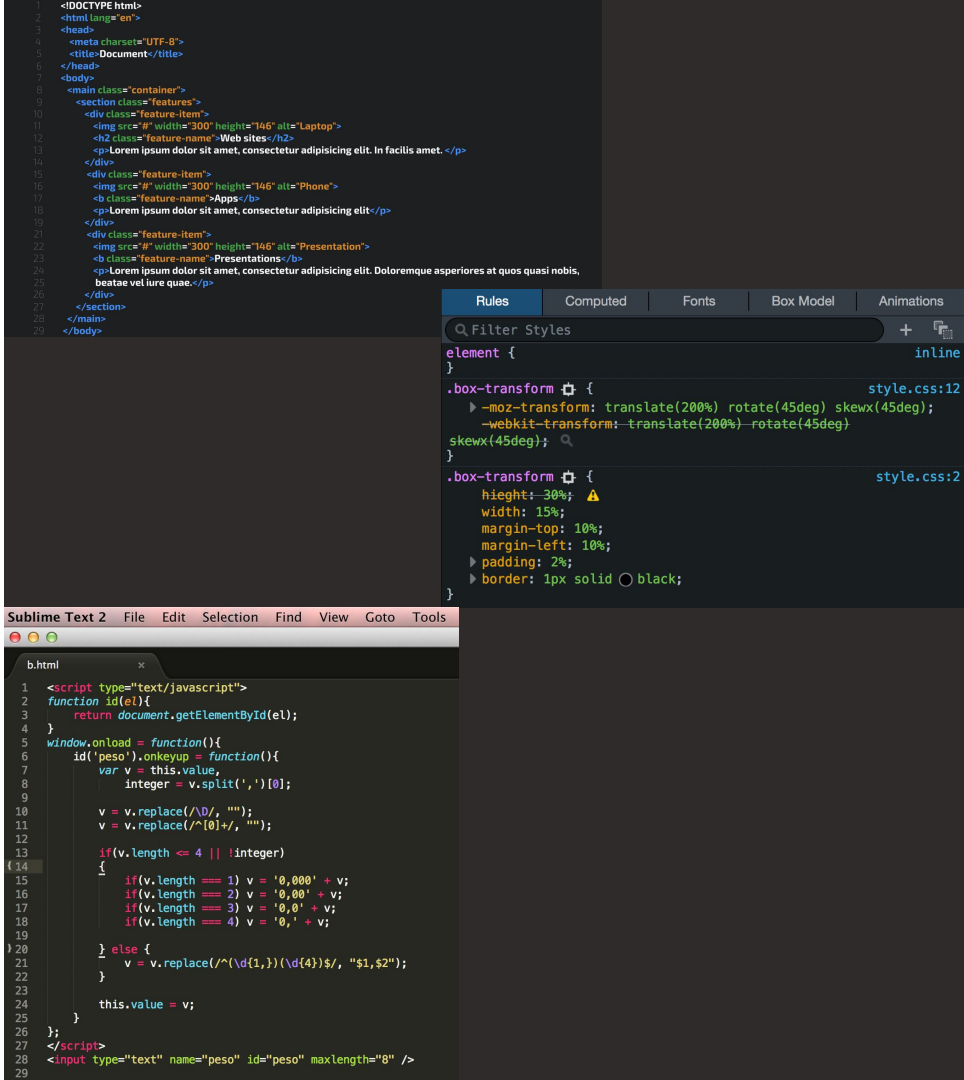
O que é uma *webpage*?

Uma webpage é um documento individual que pode ser visualizado online. Ela pode ser uma página repleta de textos, imagens ou vídeos. Uma *webpage* pode ser um formulário simples e até mesmo aparentar estar vazia, abrigando códigos que você sequer pode notar. Cada *webpage* contém uma URL específica que direciona os usuários até ela.

Um *website*, por sua vez, é uma coleção de *webpages.* Se *webpages* são páginas, o site é como se fosse o livro todo.

O elemento principal de uma página web é um ou mais arquivos de texto escritos na linguagem HTML (Hypertext Markup Language).

Muitas páginas web também usam código JavaScript para comportamento dinâmico e código CSS (Cascading Style Sheets) para semântica de apresentação. Imagens, vídeos e outros arquivos multimídia também são frequentemente incorporados em páginas web.



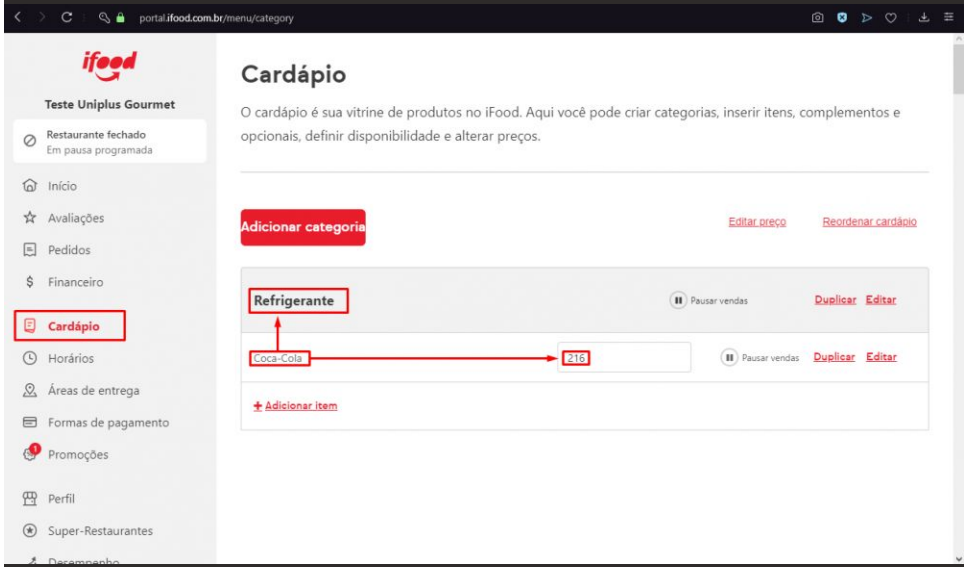
Qual é a diferença entre um site estático e dinâmico?

Um site estático é aquele que **não** conta com ferramentas de gerenciamento para alterações, mudanças ou reformulações realmente significativas de conteúdo, sem alteração do código fonte.

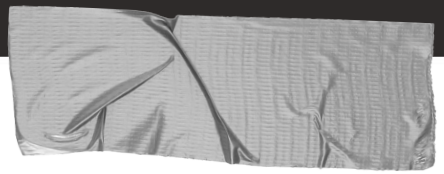
Isso significa que, caso você deseje realizar esse tipo de modificação em um *website* estático, precisará de um profissional que tenha conhecimento de linguagens como CSS, JavaScript, PHP ou HTML.

Um site dinâmico é o contrário do estático. Ele permite que sejam realizadas constantes **alterações** de conteúdo, como a criação de páginas, mas sem a necessidade de alteração no código fonte por parte de um profissional da área.

Também chamado de **site gerenciável**, o site dinâmico conta com um sistema de gerenciamento integrado, possibilitando a realização de alterações por meio de um **CMS** (Content Management System), como o WordPress.



O que é uma aplicação Web?

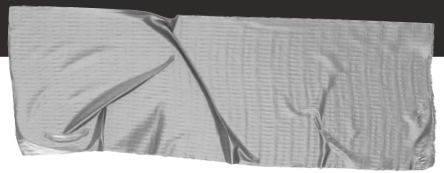


Aplicação Web:

A aplicação web diz respeito a uma solução que é executada diretamente no navegador, não sendo preciso realizar uma instalação na máquina do usuário. Pode-se, também, utilizar como definição: *tudo aquilo que é processado em um servidor terceiro*.

As plataformas de e-commerce, os apps do google como o *docs* e o *meets*, e as redes sociais são alguns dos exemplos de aplicações web.

Como se desenvolve aplicações Web?



Front-end

Front-end significa ‘parte dianteira’, ou seja, a porta de frente de uma aplicação web. É a parte responsável por “dar vida” à interface.

Podemos definir *front-end* como a parte visual de um site, desenvolve o código que será renderizada nos aspectos visuais e interativos de uma página.

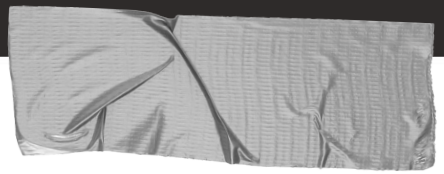
HTML, CSS, JavaScript (React, Angular), Dart (Flutter), Swift e XML são exemplos de linguagens utilizadas para a construção visual de páginas e aplicativos.

Um código simples de front-end se parece com isso ->

Esta é uma página HTML, observe as tags de abertura e fechamento que denotam que elemento da página que está sendo criado.

body: corpo
div: divisão
h1: header 1 / título, cabeçalho
ol: ordered list / lista ordenada
li: list item / item de lista

```
<html>
  <body>
    <div>
      <h1>The truth about elks and bears.</h1>
      <ol>
        <li>An elk is a smart</li>
        <li>...and cunning animal!</li>
      </ol>
      <ol>
        <li>A bear is a majestic</li>
        <li>...and dangerous animal!</li>
      </ol>
    </div>
  </body>
</html>
```



Back-end

Back-end, ou 'parte traseira' é a parte de "trás" de uma aplicação Web. Ela é a responsável, em termos gerais, pela implementação da **regra de negócio.**

Ela é o 'motor' de uma aplicação web: ela que verifica, por exemplo, se a senha bate com o usuário e efetua o login em um site. Também é no back-end que se faz a comunicação com o banco de dados, para buscar ou armazenar informações, etc.

Quando falamos de back-end em desenvolvimento web, nos deparamos com várias linguagens, como *Go*, *Clojure*, *C#*, *PHP*, *Java*, *Python*, *Ruby*, entre outras.

Um código simples de back-end se parece com isso ->

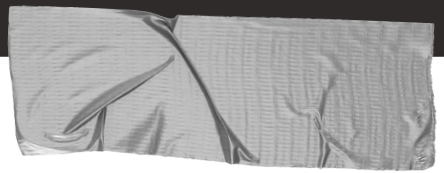
Este é um módulo em TypeScript, que define o que será feito se algo for postado em '/current'.

Parece ser uma aplicação web de algum jogo, em que no endpoint da API '*/current*' temos o jogador selecionado. O método está definindo a ação de trocar o jogador selecionado. Assim, ela recebe como parâmetros da requisição o identificador do jogador, o seu ID.

Com ele, ele verifica se há um jogador com esse ID. Se não há o ID, ele imprime no console que não há jogador com tal ID. Havendo o ID, ele faz a troca do jogador selecionado e retorna o novo jogador. Se houver algum tipo de erro, ele captura o erro e imprime no console.

```
@Post("/current")
@UndefinedResultCode(404)
no references found for changeCurrentPlayer
async changeCurrentPlayer(@Req() req: Request, @BodyParam("id") playerId: number) {
  const player = await this.playerService.findPlayerByIdEager(playerId);

  if (!player) {
    console.log(`Player with id ${playerId} was not even found`);
  }

  try {
    await this.playerService.changeCurrentPlayer(req.user, player);
  } catch (err) {
    if (err instanceof LogicError) {
      return;
    }

    console.log(err.stack);

    throw new HttpError(500, "Something gone wrong");
  }

  return player;
}
```

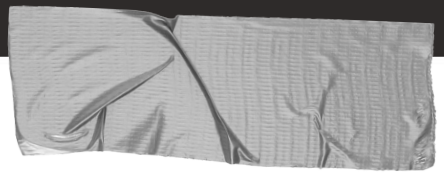
Como front-end e back-end se comunicam?



APIs

Uma **API** *(Application Programming Interface)* pode ser definida como um conjunto de padrões que permite a construção de aplicativos, onde ele conecta aplicações, podendo ser utilizada nos mais variados tipos de negócios.

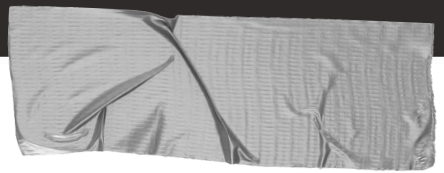
Com a API você tem uma interface para que um sistema se comunique com outro sistema, compartilhando suas ações e ferramentas. A comunicação é feita através de vários códigos, definindo comportamentos específicos.



API Endpoints

Um endpoint é basicamente o que um serviço expõe e esse serviço pode ser acessado por uma aplicação, por isso muitas vezes acaba sendo confundido com uma API.

Um endpoint contém três principais características: Address (onde o serviço está hospedado), Binding (como o serviço pode ser acessado) e Contract (o que tem no serviço).

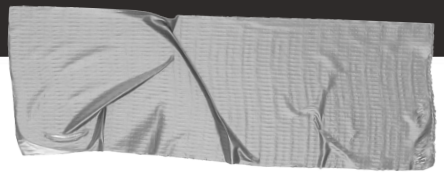


JSON

O formato JSON (JavaScript Object Notation) é, como o nome sugere, uma forma de notação de objetos JavaScript de modo que eles possam ser representados de uma forma comum a diversas linguagens.

Além disso, uma ideia que está fortemente enraizada neste formato é que ele seja facilmente trafegado entre aplicações em quaisquer protocolos, inclusive o HTTP.

Portanto, a principal diferença entre um objeto JavaScript padrão e um JSON é o fato do JSON ser na realidade: um texto.



Integração

A integração é como se comumente chama o ato de conectar o *back-end* ao *front-end*.

Geralmente, o back-end é estruturado em uma API, em que cada endpoint expõe um serviço necessário para a aplicação.

O front-end consome dessa API as informações que serão exibidas na página. Para isso, ela faz a requisição (passando parâmetros ou não) em um dos endpoints. As informações vêm em formato JSON. A partir disso, o front consegue mapear os dados e exibi-los na tela.

https://virtserver.swaggerhub.com/TradeshowDemos/SamplePetstoreAPI/1.0.0

Show O

pet Everything about your Pets

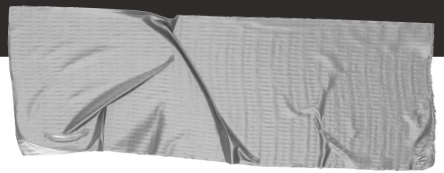
Find out more: http://swagger

POST	/pet	Add a new pet to the store
PUT	/pet	Update an existing pet
GET	/pet/findByStatus	Finds Pets by status
GET	~~/pet/findByTags~~	Finds Pets by tags
GET	/pet/{petId}	Find pet by ID
POST	/pet/{petId}	Updates a pet in the store with form data
DELETE	/pet/{petId}	Deletes a pet
POST	/pet/{petId}/uploadImage	uploads an image

store Access to Petstore orders

```
{
  “firstName”: “Jonathan”,
  “lastName”: “Freeman”,
  “loginCount”: 4,
  “isWriter”: true,
  “worksWith”: [“Spantree Technology Group”, “InfoWorld”],
  “pets”: [
    {
      “name”: “Lilly”,
      “type”: “Raccoon”
    }
  ]
}
```

Qual a diferença entre aplicações Web tradicionais e SPAs?



Aplicações SPAs

SPA significa *Single Page Application*, ou Aplicação de Página Única. Mas não é um site de uma só página como o nome sugere. É, na verdade, uma forma de site com múltiplas páginas que carrega apenas uma vez.

Em vez de um carregamento por página (uma requisição por página), como é uma aplicação Web convencional, um SPA carrega toda a aplicação apenas uma vez, demorando mais no início, mas tendo mais fluidez na troca de páginas.