7. COMUNICAÇÃO ENTRE FRONT-END E BACK-END

COMUNICAÇÃO ENTRE FRONT-END E BACK-END

Uma Aplicação Web sempre é composta por dois "blocos": o front-end e o back-end. Podemos classificar nossas tecnologias, arquiteturas, etc baseado nessas duas separações. É comum ouvir dizer também que o beck-end é Server Side, e o front-end Client Side.

O que é back-end?
São todas as tecnologias que rodam em um "Servidor", uma ou mais máquinas.

E o front-end?
Ele é tudo que roda do lado cliente, ou seja, o que o usuário final vê.

Referência: https://devpleno.com/aplicacao-web

COMUNICAÇÃO ENTRE FRONT-END E BACK-END

Para a parte do cliente, geralmente temos o Navegador (browser), ele faz o Request para o servidor ou tecnologia back-end. (Apenas esclarecendo, o request é o front-end pedindo alguma coisa ao back-end, por exemplo, uma página, uma imagem, por aí vai).

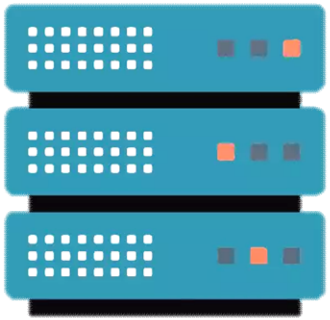
A primeira comunicação sempre é feita do Front-end para o Beck-end. Por exemplo, ao entrar no site seu navegador faz uma requisição à máquina que é o servidor, com isso o server irá mandar uma resposta (response) para o navegador, que é nosso front-end. Essa comunicação é chamada de request-response, pois tudo é baseado nisso, entre a requisição do front, resposta do server.

Referência: https://devpleno.com/aplicacao-web

COMUNICAÇÃO ENTRE FRONT-END E BACK-END

Request

Response



Client

Server

Referência: https://devpleno.com/aplicacao-web

QUAIS TECNOLOGIAS TEMOS PARA CADA TIPO?

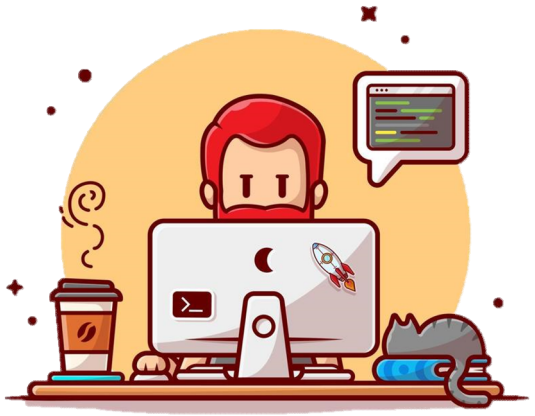
Algumas tecnologias **front-end** são: HTML, Javascript, CSS. Temos tembem alguns assets como imagem e PDF, eles são alguns arquivos que precisamos para funcionar o front-end. Antigamente existia o Flash e o Java Applet, mas hoje em dia estão em desuso por causa do HTML5.

Já do lado **back-end** temos N tecnologias, tais como: PHP, ASP, JAVA, NodeJS, C#, Python, C++, etc. Além disso, podemos também ter banco de dados.

Referência: https://devpleno.com/aplicacao-web

QUAIS TECNOLOGIAS TEMOS PARA CADA TIPO?

O lado **back-end**, é o lado que, se não for padronizado a WEB, continuará normal, mas já o lado do **front-end** tivemos que ter uma padronização, por esse motivo temos um número reduzido de tecnologias, ela é padronizada pelo W3C, já que ali está a maior parte dos usuários e diversidade de dispositivos que vão usar o **client side** para acessar o **server side**. Já o Server side tem a função de devolver algo padronizado para o cliente.



Referência: https://devpleno.com/aplicacao-web

O que é Express?

O QUE É EXPRESS?

Express é um framework para Node.js que facilita a criação de servidores web. Ele não é uma linguagem nova — ele usa JavaScript + Node, mas deixa a criação de APIs e servidores mais rápida e organizada.

✓ **Com o Express você pode:**

- Criar rotas (GET, POST, PUT, DELETE, etc.);
- Receber e enviar dados (JSON, formulários);
- Conectar com banco de dados;
- Criar APIs para um site ou aplicativo;
- Gerenciar middlewares (autenticação, validação, etc.).

COMO INSTALAR O EXPRESS.JS

1. **Primeiro, tenha o Node.js instalado**

Se ainda não instalou o Node.js, você precisa baixar: https://nodejs.org/Instalando o Node.js, você também instala o npm (gerenciador de pacotes).

2. Criar uma pasta para seu projeto

No terminal/cmd/powershell:

- **mkdir meu-projeto**
- **cd meu-projeto**

COMO INSTALAR O EXPRESS.JS

3. Iniciar um projeto Node.js

Dentro da pasta, digite: **npm init –y**

4. Instalar o Express

Agora, instale o Express com: **npm install express**

EXEMPLO RÁPIDO DE SERVIDOR COM EXPRESS

5. Crie um arquivo chamado index.js:

```
const express = require('express');
const app = express();
const port = 3000;

app.get('/', (req, res) => {
  res.send('Olá, mundo!');
});

app.listen(port, () => {
  console.log(`Servidor rodando em http://localhost:${port}`);
});
```

EXEMPLO RÁPIDO DE SERVIDOR COM EXPRESS

6. Rode o servidor

node index.js

Abra o navegador e acesse **http://localhost:3000** — você verá a mensagem "Olá, mundo!".

Listagem 1. Percorrendo árvores

EXEMPLO RÁPIDO DE SERVIDOR COM EXPRESS

`const express = require('express');` → Aqui você está importando o módulo express, que você instalou no seu projeto (npm install express). Assim você pode usar todas as funções do Express.

`const app = express();` → Você está criando uma aplicação Express. O app é a sua aplicação (o seu servidor).

`const port = 3000;` → Define que o servidor vai rodar na porta 3000. (Quando você abrir no navegador: http://localhost:3000)

EXEMPLO RÁPIDO DE SERVIDOR COM EXPRESS

```
app.get('/', (req, res) => {
  res.send('Olá, mundo!');
});
```

Aqui você criou uma rota:

app.get('/') → Quando alguém acessar o endereço principal (/), ou seja, http://localhost:3000
(req, res) → req é o pedido que o navegador faz, res é a resposta que o servidor dá
res.send('Olá, mundo!') → O servidor responde com a mensagem "Olá, mundo!".

EXEMPLO RÁPIDO DE SERVIDOR COM EXPRESS

```
app.listen(port, () => {
  console.log(`Servidor rodando em http://localhost:${port}`);
});
```

Essa linha liga o servidor.

app.listen(port, ...) faz o seu servidor começar a escutar na porta que você escolheu.

Quando o servidor ligar, ele mostra no terminal:
Servidor rodando em http://localhost:3000

EXEMPLO RÁPIDO DE SERVIDOR COM EXPRESS

Resumido:

const express = require('express') Importa o Express

const app = express() Cria o servidor

const port = 3000 Define a porta

app.get('/', (req, res) => {...}) Cria uma rota que envia "Olá, mundo!“

app.listen(port, () => {...}) Inicia o servidor