

<LAB365>

**BOAS-VINDAS & FERRAMENTAS**

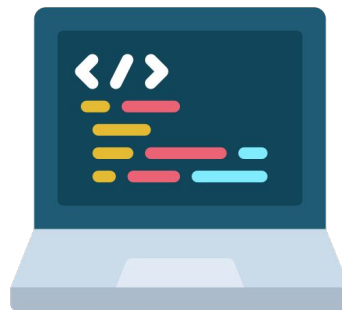
<LAB365>

**SENAI**

## AGENDA | M1S01 - A1

- Apresentação do mentor;
- Apresentação da turma;
- Ferramentas de estudo;
- Exercícios;

# APRESENTAÇÃO DO MENTOR



# RECOMENDAÇÕES

- Participar das reflexões
- Canal tira-dúvidas: Canal do Discord “stack-overflow” (dúvidas assíncronas)
- Trabalhar em equipe (boa comunicação)
- Saber ajudar e receber ajuda (dividir tarefas / delegar)
- Cada pessoa tem sua bagagem, seu tempo
- Praticar, resolver exercícios/desafios
- Tempo limitado em cada aula (acesse os conteúdos complementares)
- Links / Artigos da bibliografia
- Interaja! Use o chat
- Faça comentários
- Acrescente informações
- Me corrija
- Anote! (papel ou meio digital)

*Tire suas dúvidas*

**CHAT!**

## APRESENTAÇÕES DA TURMA

- Nome?
- Idade?
- De onde fala?
- O que o trouxe ao curso?
- Já programou antes?

*Escreva no Discord*

Poste no canal  
**hello-world**

# APRESENTAÇÃO DA TURMA

Vamos usar o Padlet!

<https://padlet.com/ottohannemann/futurodev-joinville-cin47kn7apchf4cg>



# padlet

# FERRAMENTAS



# FERRAMENTAS

- Discord (convite via e-mail) [discord.com](https://discord.com)
- E-mail (logar com e-mail de estudante) [gmail.com](https://gmail.com)
- Google Meet [meet.google.com](https://meet.google.com)
- AVA (conteúdos e links úteis) [ava.sesisenai.org.br](https://ava.sesisenai.org.br)
- Espaço do Estudante [estudante.sesisenai.org.br](https://estudante.sesisenai.org.br)
- Trello (exercícios semanais) [trello.com](https://trello.com)
- VS Code (Prettier, Live Server, Auto Rename Tag, GitLens, CSS Peek, ESLint, Material Icon Theme)  
[code.visualstudio.com/download](https://code.visualstudio.com/download)
- GitHub (criar uma conta) [github.com](https://github.com)



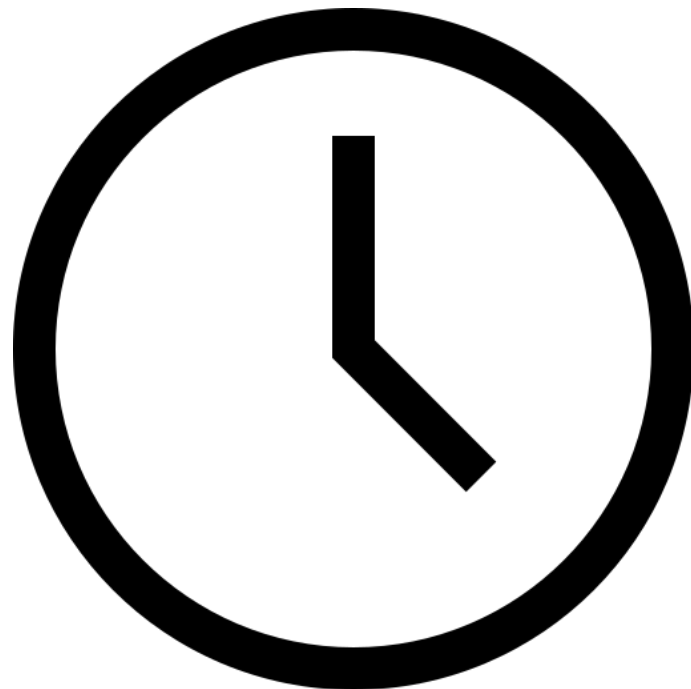
## INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

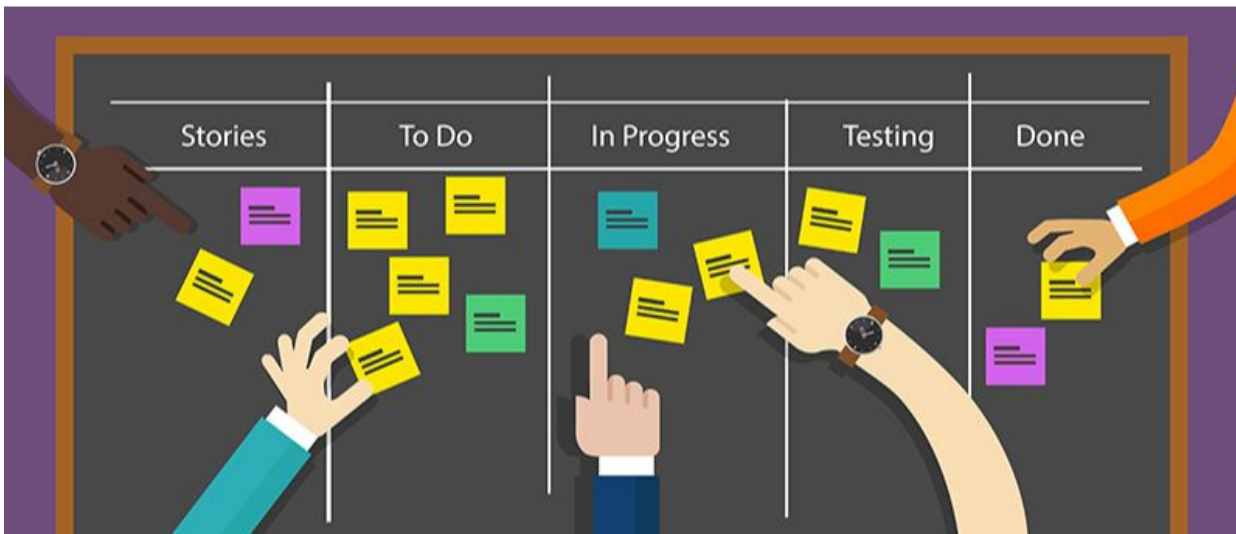
Nos vemos em 20 minutos.

**Início:** 20:20

**Retorno:** 20:40



# EXERCÍCIOS | KANBAN | TRELLO



# KANBAN

- Kanban é um sistema de controle de estoque e fluxo de peças desenvolvido pela Toyota por volta de 1950/60;
- Foi adaptado para diversas áreas, inclusive desenvolvimento de software, onde é muito utilizado para Gerenciamento de Projetos;
- É um sistema visual de gestão de trabalho, que busca conduzir cada tarefa por um fluxo pré definido de trabalho.

Um **fluxo de trabalho** que busca indicar (e limitar) o trabalho em andamento. O foco do Kanban é priorizar a produtividade e a organização das entregas com objetivo de proporcionar um trabalho mais transparente e direcionado.

# KANBAN

O kanban tem três principais funções:

- **Gerenciar o fluxo de trabalho** e facilitar a visualização da dimensão do que está sendo produzido e em que ritmo está sendo produzido;
- **Equilibrar os processos** que vêm antes e depois, para que uma atividade não seja interrompida pela falta de uma outra que deveria ter sido entregue anteriormente;
- **Limitar a quantidade** de trabalho que deve ser realizada pela equipe, respeitando a capacidade produtiva.

# KANBAN

As três partes principais do Kanban são: **cartão**, **coluna** e **quadro**

- **Cartão:**

- É a **menor parte** do kanban;
- Trata-se de uma **tarefa ou ação** que precisa ser tomada para que o resultado final seja entregue;
- Geralmente são diferenciados por um **sistema de cores**;
- Cada vez que uma atividade muda de **status**, basta mover o cartão para a outra **coluna** que indica o estado atual.

- **Quadro:**

- O quadro nada mais é do que o **kanban** como um todo, organizado em **colunas** e **cartões**.
- Cada quadro é um kanban e uma única equipe pode trabalhar com vários quadros simultaneamente.

# KANBAN

- **Priorização de tarefas:**

- Com a adequação de cores, ordem ou colunas, é possível destacar uma tarefa para demonstrar a importância da mesma.

- **Aumento da produtividade:**

- Considerando que o Kanban permite ver futuras tarefas, não há tempo ocioso. Também, se especifica o que deve ser feito, não sendo necessário o retrabalho.

- **Redução de custos:**

- O aumento da produtividade já traz redução de custos e ainda é possível direcionar melhor a equipe com a **eficiência e a facilidade** que a metodologia Kanban mostra.

- **Autonomia:**

- É fácil olhar para o quadro e entender o **status das entregas** e também o que precisa ser feito, isso **estimula a autonomia** da equipe de trabalho já que eles podem **verificar sozinhos o andamento das entregas**.

# KANBAN



Fonte: [O que é Kanban: Guia completo \(atualizado 2025\)](#)

<LAB365>

<LAB365>

**SENAI**



<LAB365>

**INTERNET & TECNOLOGIA**

<LAB365>



## AGENDA | M1S01 - A2

- Criação e história da Internet;
- Arquitetura cliente-servidor;
- Caminhos dentro da TI;

# INTERNET

*"Uma coleção mundial de redes interconectadas que usam o protocolo TCP/IP para comunicação, permitindo que milhões de dispositivos troquem dados e recursos de forma descentralizada."*

**Andrew S. Tanenbaum**, renomado cientista da computação e autor de referência em redes de computadores

Para entendermos um pouco melhor toda essa parte do desenvolvimento web temos de iniciar entendendo um pouco sobre o que é a internet e como ela se formou.

## INTERNET: História - Década de 60

Durante a Guerra Fria, os EUA buscavam uma rede de comunicação descentralizada que sobrevivesse a ataques nucleares.

Em 1969 a ARPANET foi criada pela ARPA (Agência de Projetos de Pesquisa Avançada) do Departamento de Defesa dos EUA.

- Primeira conexão: UCLA (Universidade da Califórnia, Los Angeles) e SRI (Stanford Research Institute).
- Primeira mensagem: "LOGIN" (a rede caiu após as primeiras duas letras).

# INTERNET: História - Década de 70

A década de 70 é marcada pela criação dos protocolos e expansão da internet inicial.

- 1971: Ray Tomlinson envia o primeiro e-mail e introduz o uso do "@" para separar o nome do usuário do servidor.
- 1973: Primeira conexão internacional da ARPANET com a Noruega e o Reino Unido.
- 1974: Vint Cerf e Bob Kahn desenvolvem o TCP/IP, o protocolo que permitiria a comunicação entre redes diferentes.

# INTERNET: História - Década de 80

Nos anos 80 temos o início da transição para uma rede mundial, em um formato que estamos mais acostumados.

- 1983: Adoção oficial do TCP/IP pela ARPANET, marcando o nascimento da "internet" como a conhecemos.
- 1984: Introdução do DNS (Domain Name System), que substituiu os endereços IP numéricos por nomes de domínio (ex: .com, .org).
- 1989: Tim Berners-Lee, no CERN, propõe a criação da World Wide Web (WWW).

# INTERNET: História - Década de 90

Nos anos 90 a internet já está bem estabelecida aos moldes atuais e temos a popularização da mesma.

- 1991: Tim Berners-Lee lança a WWW, combinando hipertexto, TCP/IP e DNS para criar uma rede de documentos interconectados.
- 1995: A internet é aberta para uso comercial, e empresas como Amazon e eBay começam a operar.
- 1998: Google é fundado, revolucionando a busca na web.
- 1999: Wi-Fi é padronizado, permitindo conexões sem fio.

# INTERNET: História - 2000+

- Anos 2000:
  - Facebook (2004) e YouTube (2005) revolucionam redes sociais e vídeos.
  - iPhone (2007) impulsiona a internet móvel.
  - Computação em nuvem (AWS, 2008) ganha força.
- Anos 2010:
  - Internet das Coisas (IoT) conecta dispositivos inteligentes.
  - Instagram e WhatsApp dominam as redes sociais.
  - 5G é lançado (2019), prometendo velocidade e baixa latência.
- Anos 2020:
  - Avanços em IA, realidade virtual e aumentada.
  - Expansão do 5G e pesquisas com 6G.
  - Desafios: privacidade, segurança e inclusão digital.



# ARQUITETURA CLIENTE-SERVIDOR

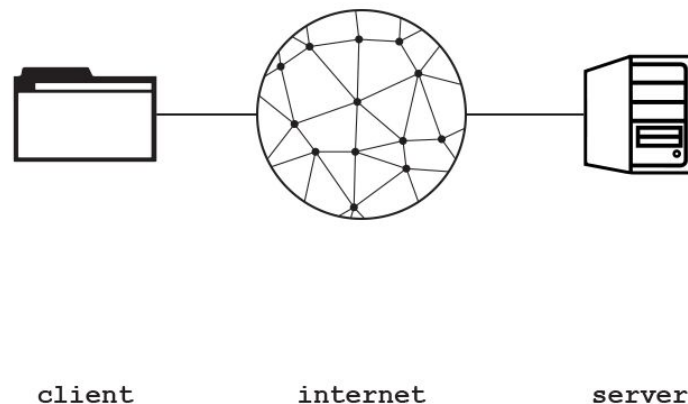
Um servidor é um recurso dentro de um sistema computacional, capaz de processar aplicações, armazenar dados e prestar serviços, enquanto o cliente é um computador que solicita esses serviços e recursos. Em uma infraestrutura de TI, ambos formam a arquitetura conhecida como “cliente-servidor”.

Uma estrutura cliente-servidor é um modelo de arquitetura de rede em que um programa de computador (cliente), solicita um serviço ou recurso de outro programa de computador (servidor), recebendo uma resposta com as informações solicitadas por meio de protocolos de rede em infraestruturas de TI.

# ARQUITETURA CLIENTE-SERVIDOR

A comunicação cliente-servidor é baseada em troca de mensagens, onde o cliente envia requisições e o servidor responde com as informações requisitadas.

A arquitetura cliente-servidor pode ser implementada de diferentes maneiras, como em aplicações web, jogos online, sistemas de gerenciamento de bancos de dados, entre outros.



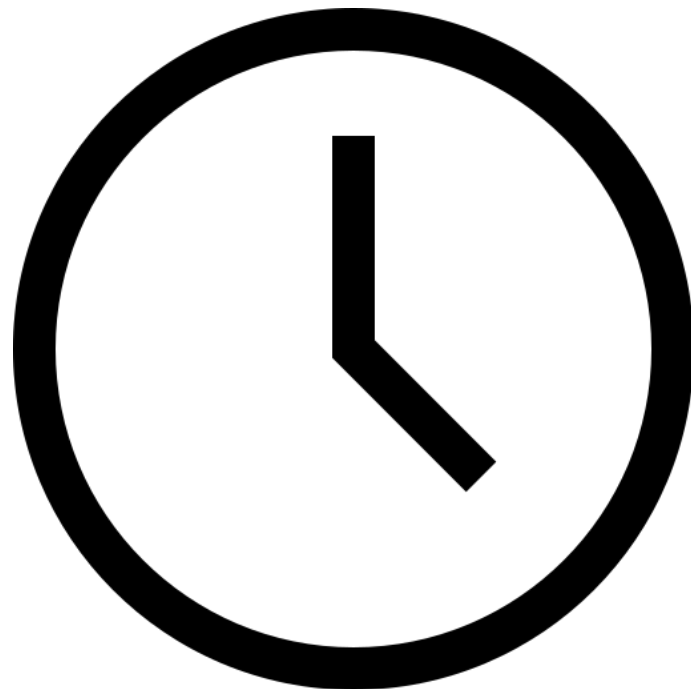
## INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

**Início:** 20:20

**Retorno:** 20:40



# TECNOLOGIAS E CAMINHOS NA TI

A área de Tecnologia da Informação (TI) é um dos pilares da transformação digital que vivemos hoje, e seu impacto vai muito além dos códigos e algoritmos. Enquanto o desenvolvimento de software é uma das carreiras mais visíveis e demandadas, a TI oferece um leque diversificado de caminhos para quem deseja trabalhar com tecnologia, inovação e resolução de problemas.

Agora vamos falar um pouco sobre as diversas áreas de atuação dentro da TI e sobre quais tecnologias são utilizadas dentro de cada área. Obviamente passaremos superficialmente em cada área, pois somente dentro do desenvolvimento de sistemas temos uma infinidade de frameworks e linguagens que poderiam ser citadas.

# DESENVOLVIMENTO DE SOFTWARE

O que faz: Criação de aplicativos, sistemas e softwares para diversas plataformas (web, mobile, desktop).

Caminhos:

- Front-end;
- Back-end;
- Full-stack;
- Mobile.

Carreira: Desenvolvedor Analista, Engenheiro/Arquiteto de Software, Tech Lead.



# CIÊNCIA DE DADOS E INTELIGÊNCIA ARTIFICIAL

O que faz: Analisa grandes volumes de dados para extrair insights e criar modelos preditivos.

Caminhos:

- Data Science;
- Machine Learning;
- Big Data.

Carreira: Cientista de Dados, Engenheiro de Machine Learning, Analista de BI.



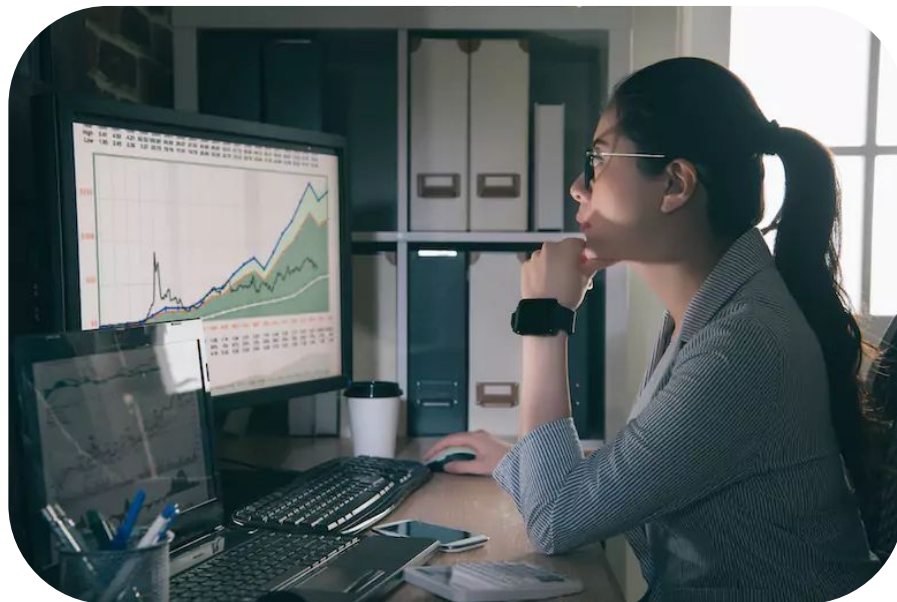
# SEGURANÇA DA INFORMAÇÃO

O que faz: Protege sistemas, redes e dados contra ameaças cibernéticas.

Caminhos:

- Pentest (Testes de Invasão);
- Segurança de Redes;
- Governança de Dados.

Carreira: Analista de Segurança, Ethical Hacker, CISO (Chief Information Security Officer).



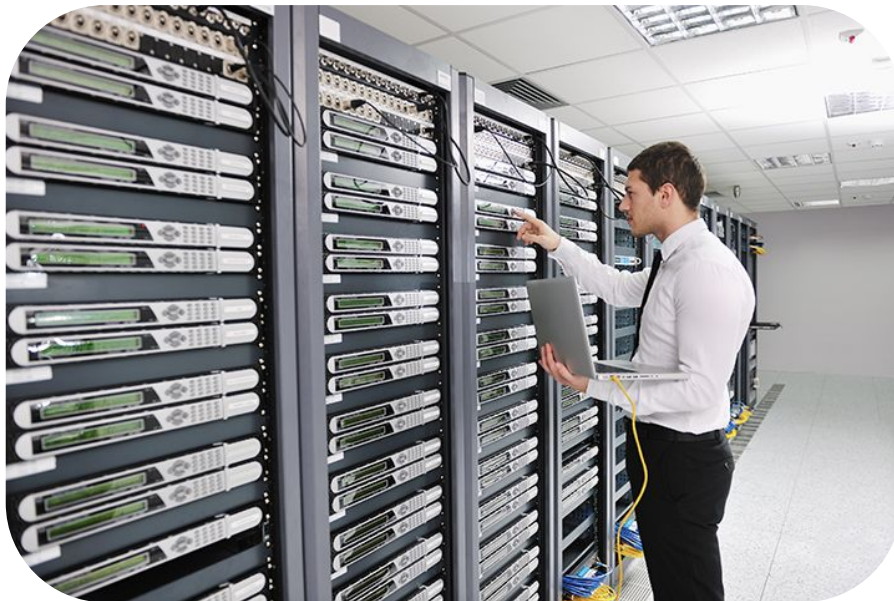
# INFRAESTRUTURA E REDES

O que faz: Gerencia servidores, redes e sistemas que sustentam a operação de empresas.

Caminhos:

- Administração de Redes;
- Cloud Computing;
- DevOps.

Carreira: Administrador de Redes, Engenheiro de Cloud, Especialista em DevOps.





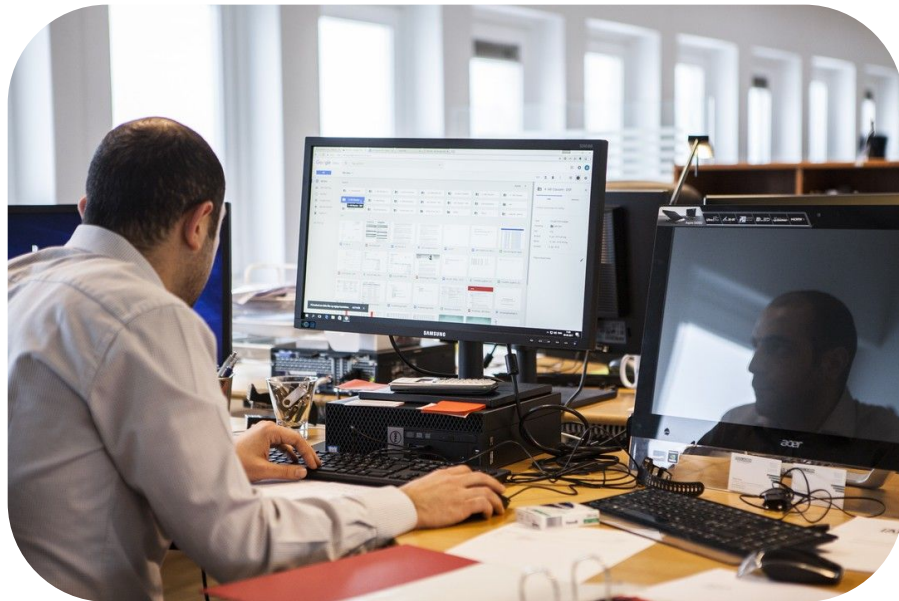
# BANCO DE DADOS

O que faz: Projeta, implementa e gerencia bancos de dados para armazenar e organizar informações.

Caminhos:

- SQL;
- NoSQL;
- Data Engineering.

Carreira: Administrador de Banco de Dados (DBA), Engenheiro de Dados.



# UX/UI DESIGN

O que faz: Cria experiências e interfaces intuitivas e atraentes para os usuários.

Caminhos:

- UX;
- UI.

Carreira: Designer de UX/UI, Product Designer.



<LAB365>

<LAB365>

**SENAI**

# <LAB365>

**ESTUDANDO TI E CONFIGURAÇÃO DE AMBIENTE**

## AGENDA | M1S01 - A3

- Perfis de aprendizado;
- Carreira de desenvolvedor;
- Iniciando no Front-end;
- Configuração de ambiente;

## PERFIS DE APRENDIZADO

O que nos diferencia é nossa singularidade e isso se aplica das mais diversas formas, incluindo como aprendemos.



# PERFIS DE APRENDIZADO



# CARREIRA DE DESENVOLVEDOR

Dentro da Carreira de TI, um dos fatores que sempre é um diferencial são os conhecimentos do Profissional.

Adquirimos conhecimento por meio de:

- Experiência;
- Estudos/Qualificações.



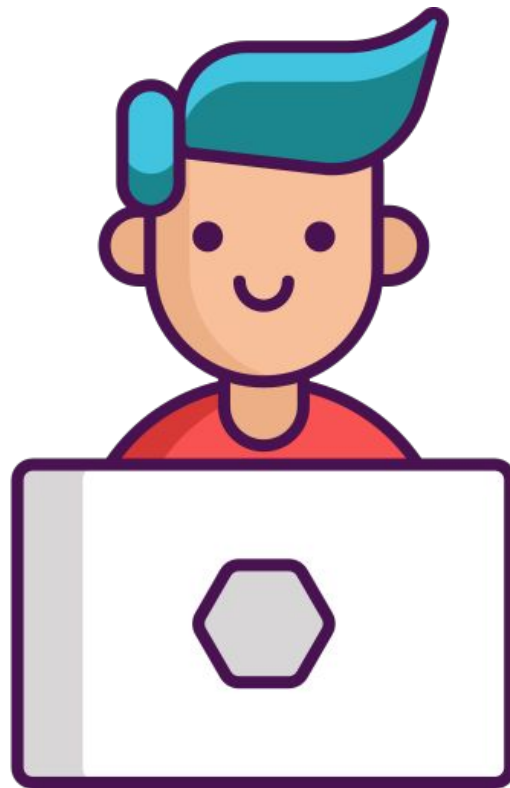


## CARREIRA DE DESENVOLVEDOR - EXPERIÊNCIA

A carreira de programação geralmente é dividida em níveis que refletem a experiência, habilidades e responsabilidades de cada profissional.

Embora os títulos e estruturas possam variar entre empresas, os níveis mais comuns são:

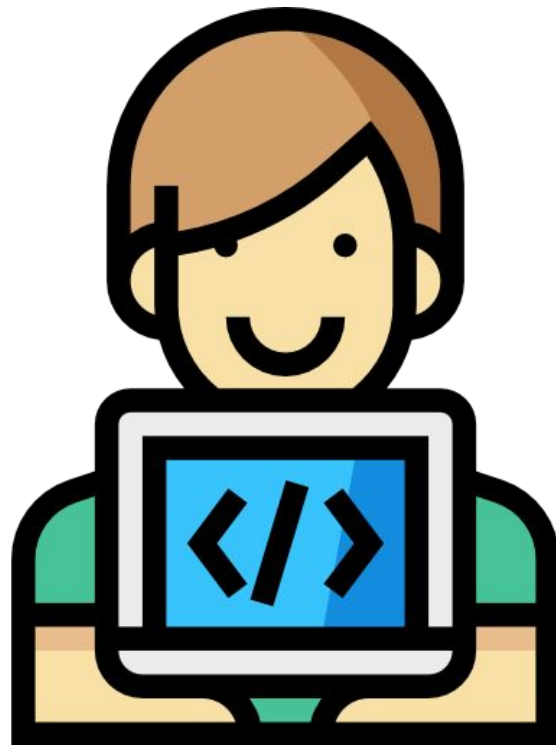
- Estágio/Trainee (0-1 anos);
- Júnior (0-2 anos);
- Pleno (2-5 anos);
- Sênior (5+ anos);
- Líder técnico (7+ anos).



# CARREIRA DE DESENVOLVEDOR - QUALIFICAÇÃO

Quando falamos de qualificações temos alguns caminhos a seguir.

- Qualificação formal:
  - Curso técnico;
  - Graduação/Pós-Graduação.
- Cursos
  - Cursos Livres/Profissionalizantes;
  - Certificações.
- Prática
  - Projetos pessoais;
  - Portfólio.



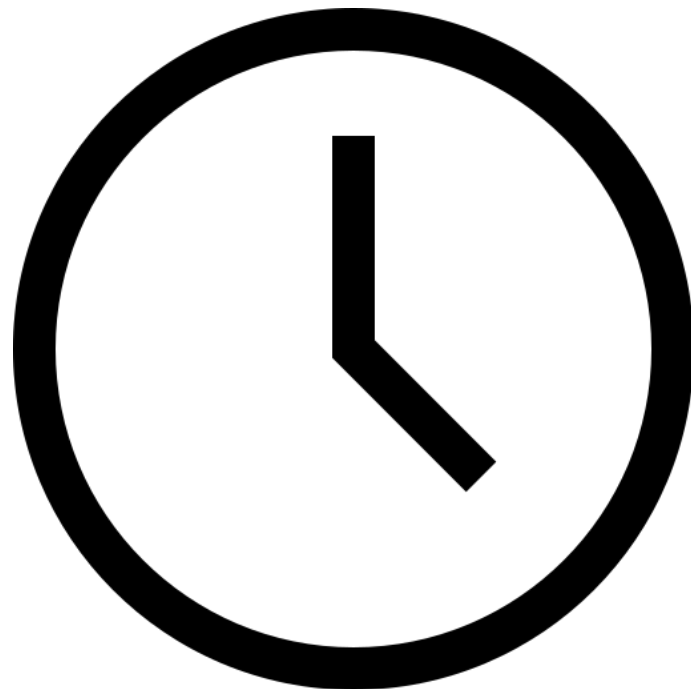
## INTERVALO!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

**Início:** 20:30

**Retorno:** 20:50



# INICIANDO NO FRONT-END

Quando falamos de desenvolvimentos no front-end, podemos resumir que estamos falando da implementação de uma interface gráfica. No ambiente web isso pode ser traduzido como uma página de internet.

Nesse contexto precisamos de uma ferramenta para escrever nosso código.

Essa ferramenta pode ser um arquivo de texto, por exemplo podemos escrever códigos no bloco de notas, mas o mais usual são ferramentas que chamados de IDEs.

# INICIANDO NO FRONT-END

Falando de IDEs, normalmente em cada tecnologia temos várias opções. Existem ferramentas online (que podemos acessar do navegador), mas o mais comum é baixarmos um programa mais completo e instalamos em nossos computadores.

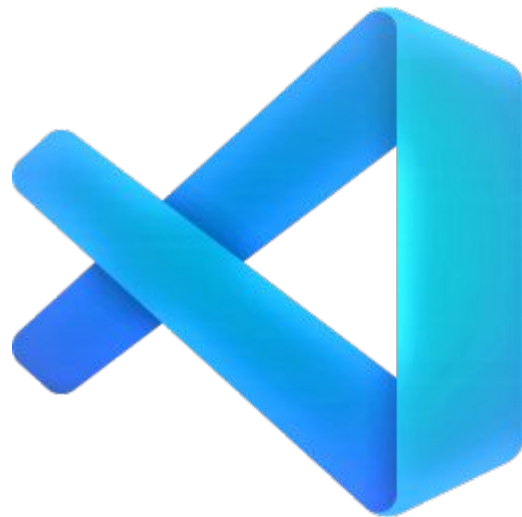
Um ponto bem importante é que a maioria das IDEs são feitas pensando no sistemas operacionais de computadores e não de dispositivos móveis. Logo é aconselhável que usem um computador com SO Windows, Linux o Mac.

# VISUAL STUDIO CODE

A ferramenta que usaremos para o desenvolvimento front-end será o Visual Studio Code.


O Visual Studio Code (ou VSCode) é um editor de código-fonte desenvolvido pela Microsoft. Ele é leve, altamente personalizável e projetado para ser uma ferramenta eficiente para desenvolvimento de software.

O VSCode é multiplataforma, ou seja, funciona em sistemas operacionais como Windows, macOS e Linux.




# VSCODE & EXTENSÕES


Agora dentro do VSCode, podemos instalar várias extensões, isso nos permite otimizar ele para trabalhar com várias linguagens de programação diferentes (como C#, Python e Java) até itens visuais que darão ícones mais receptivos aos arquivos ou ajudarão na visualização da indentação do código.




**Live Server**  
Ritwick Dey | 60,395,984 | ★★★★★ (497)  
Launch a development local Server with live reload fea...  
Disable Uninstall Auto Update



**Prettier - Code formatter**  
Prettier | prettier.io | 54,561,428 | ★★★★★ (476)  
Code formatter using prettier  
Disable Uninstall Auto Update




**ESLint**  
Microsoft | microsoft.com | 41,266,118 | ★★★★★ (241)  
Integrates ESLint JavaScript into VS Code.  
Install Auto Update  
★ This extension is recommended based on the files you recently opened.



**CSS Peek**  
Pranay Prakash | 6,761,167 | ★★★★★ (87)  
Allow peeking to css ID and class strings as definitions from html files to...  
Install Auto Update



**Auto Rename Tag**  
Jun Han | 21,228,525 | ★★★★★ (205)  
Auto rename paired HTML/XML tag  
Install Auto Update



**Material Icon Theme**  
Philipp Kief | 27,926,502 | ★★★★★ (365) | Sponsor  
Material Design Icons for Visual Studio Code  
Set File Icon Theme Disable Uninstall Auto Update

## AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique [aqui](#) ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.





<LAB365>

# ATRIBUTOS

- **Atributos são valores definidos dentro de uma classes**, e cada objetos irá ser responsável por preencher esse valores.
- Uma classe deve ter atributos esses atributos vão ser preenchidos em cada objeto. Sendo assim, nós podemos ter um classe de pessoas e um objeto André, ou conhecido, ou qualquer outra pessoa que podemos criar. Vamos ver isso na prática

```
public static void main(String[] args) {  
    Pessoa pessoa = new Pessoa();  
  
    pessoa.nome = "André";  
    pessoa.idade = 22;  
    pessoa.paisOrigem = "Brasil";  
  
}
```