

# Programação Avançada – AULA git

Matheus Moresco Engenharia de Software - 5º Período 2025/01



## Introdução ao Git

- O que é o git?
- Pra que serve?
- Como usar?



## O que é Git?

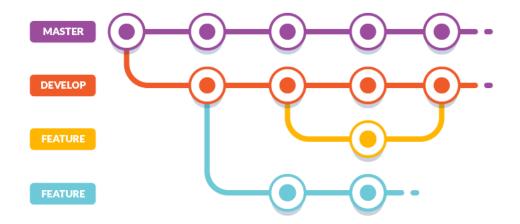
- Sistema de controle de versão distribuído.
- Criado por Linus Torvalds em 2005.
- Permite rastrear modificações no código-fonte.
- Facilita o trabalho colaborativo.





## Por que usar Git?

- Controle de versão e histórico de modificações.
- Colaboração eficiente em equipes.
- Possibilidade de reverter alterações.
- Suporte para branch e merge.





### **Principais Conceitos**

- Repositório: Local onde o código é armazenado.
- Commit: Registro de modificações.
- Branch: Linha de desenvolvimento paralela.
- Merge: Combina alterações de branches diferentes.
- Clone: Cópia de um repositório remoto.
- Push/Pull: Enviar e receber alterações de um repositório remoto.



### Instalando o Git

- Baixar e instalar via site oficial.
- Verificar instalação: git --version
- Configurar usuário:
  - git config --global user.name "Seu Nome"
  - git config --global user.email "seuemail@example.com"



## Criando um Repositório

- Criar um repositório local:
  - git init meu\_projeto
- Clonar um repositório remoto:
  - git clone https://github.com/usuario/repositorio.git



## Arquivo .gitignore

- O .gitignore é um arquivo que diz ao Git quais arquivos ou pastas ele deve ignorar ou seja, não rastrear ou versionar.
- Exemplos de itens comuns em um .gitignore:
  - Arquivos de configuração locais (.env, settings\_local.py)
  - Dependências (como a pasta node\_modules/, venv/, \_\_pycache\_\_/)
  - Arquivos temporários (\*.log, \*.tmp, caches)Builds e binários (/dist, /build, \*.exe)
- A sintaxe é bem simples:
  - Um nome de arquivo ou pasta (node\_modules/) ignora tudo que estiver com esse nome.
  - Um \* serve como curinga (\*.log ignora todos os arquivos .log).
  - Um / indica que é no nível raiz.



## Arquivo README

- O README.md é um dos arquivos mais importantes de um repositório Git. Ele serve como uma porta de entrada para quem acessa o projeto. Em geral, ele contém informações como:
  - Nome e descrição do projeto: O que o projeto faz? Para quem é?
  - Instruções de instalação: Como clonar, instalar dependências, rodar o projeto.
  - Como usar: Exemplos de uso, prints de tela, endpoints de API, etc.
  - Tecnologias utilizadas: Frameworks, bibliotecas, linguagens.
  - Como contribuir: Orientações para quem quiser ajudar no projeto.
  - Licença: Informações de direitos autorais.



## Repositórios Remotos

Distribuidores como **GitHub**, **GitLab**, **Bitbucket** e outros são **plataformas de hospedagem de repositórios Git**. Eles funcionam como **servidores remotos** para seus projetos e oferecem uma série de funcionalidades extras além do simples armazenamento de código.

### Esses serviços permitem que você:

- Armazene o código-fonte na nuvem
- Colabore com outras pessoas (controle de permissões, pull requests, revisão de código)
- Automatize processos (CI/CD integração e entrega contínua)
- Acompanhe problemas e tarefas (issues, boards)
- Documente seu projeto (wikis, páginas)
- Hospede sites estáticos (GitHub Pages, GitLab Pages)









### Ciclo Básico do Git

### **Adicionar arquivos:**

• git add arquivo.txt

#### **Fazer commit:**

• git commit -m "Mensagem descritiva"

### **Enviar alterações (push):**

• git push origin main

### **Atualizar repositório (pull):**

• git pull origin main



### Trabalhando com Branches

#### Criar uma nova branch:

• git branch nova\_branch

### Trocar para a nova branch:

• git checkout nova\_branch

### Unir alterações ao branch principal:

• git merge nova\_branch



### Resolvendo Conflito

- Ocorrem quando duas alterações impactam a mesma linha de código.
- Editar manualmente o arquivo afetado.
- Adicionar e fazer commit novamente:
  - git add arquivo.txt
  - git commit -m "Resolvendo conflito"



### Conclusão

- Git é uma ferramenta poderosa para controle de versão.
- Ajuda no rastreamento de modificações e colaboração.
- Pratique os comandos principais para dominar o uso do Git.



## Exemplo Prático

- Criar um repositório no GitHub e conectá-lo ao repositório local:
  - git remote add origin https://github.com/usuario/repositorio.git
  - git push -u origin main
- Fazer um commit
- Abrir uma nova branch e fazer alterações e depois dar um merge



### Exercício

- Criar uma conta no github
- Criar um repositório
- Clonar o repositório na sua máquina local
- Fazer alterações no arquivo de readme
- Subir as alterações para o repositório git