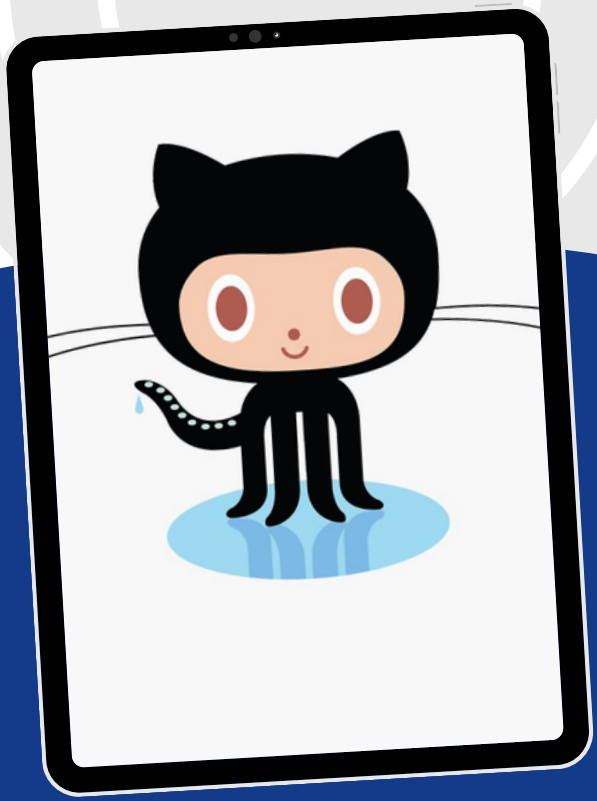
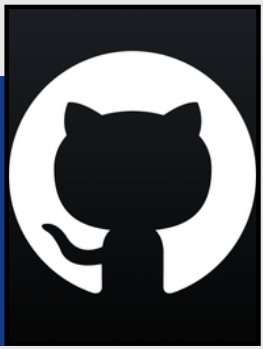
E-BOOK
INTRODUÇÃO
GIT E GITHUB

Desvende os Segredos do Git e GitHub



UFMA
Universidade Federal do Maranhão

CAPÍTULO 1: INSTALAÇÃO E CONFIGURAÇÃO



INSTALANDO O GIT

Windows

1. Acesse o site oficial do Git: https://git-scm.com/download/win
2. Baixe o instalador e execute-o.
3. Siga os passos do assistente de instalação, mantendo as configurações padrão, a menos que você tenha um motivo específico para alterá-las.

macOS

1. Abra o Terminal.
2. Execute o comando: **brew install git**

Linux (Ubuntu)

1. Abra o Terminal.
2. Execute o comando: **sudo apt-get update**
3. Em seguida, execute: **sudo apt-get install git**

CONFIGURANDO O GIT

Depois de instalar o Git, é importante configurá-lo com suas informações.

$ git config --global user.name "Seu Nome"

$ git config --global user.email "seu-email@example.com"

Essas informações serão associadas aos seus commits.

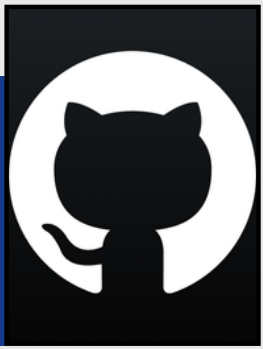
VERIFICANDO A INSTALAÇÃO

Para verificar se o Git foi instalado corretamente, abra o terminal e execute:

$ git --version

Isso deve exibir a versão do Git que foi instalada.

CAPÍTULO 2: CONCEITOS BÁSICOS DO GIT



INICIALIZANDO UM REPOSITÓRIO GIT

Para começar a rastrear as mudanças em um projeto, você precisa inicializar um repositório Git.

```
$ cd seu-diretorio-de-projeto
$ git init
```

Isso cria um novo repositório Git no seu diretório de projeto.

ADICIONANDO E COMMITANDO ARQUIVOS

Depois de fazer alterações nos arquivos do seu projeto, você pode adicionar essas mudanças ao "índice" e fazer um commit.

```
$ git add nome-do-arquivo
$ git commit -m "Mensagem de commit"
```

O primeiro comando adiciona as mudanças ao índice e o segundo comando faz um commit dessas mudanças com uma mensagem descritiva.

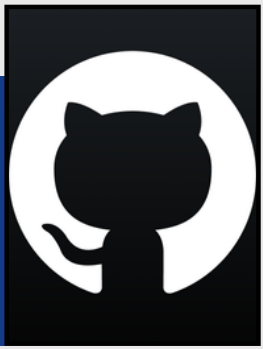
VISUALIZANDO O HISTÓRICO DE COMMITS

Você pode visualizar o histórico de commits para ver todas as mudanças que foram feitas no seu projeto.

```
$ git log
```

Isso irá mostrar uma lista de commits, incluindo o autor, a data e a mensagem do commit.

CAPÍTULO 3: TRABALHANDO COM RAMIFICAÇÕES (BRANCHES)



CRIANDO E ALTERNANDO ENTRE BRANCHES

Branches permitem que você trabalhe em diferentes versões do seu projeto simultaneamente. Para criar e alternar entre branches:

```
$ git branch nome-da-branch  # Cria uma nova branch
$ git checkout nome-da-branch  # Alterna para a nova branch
```

FUNDINDO BRANCHES (MERGE)

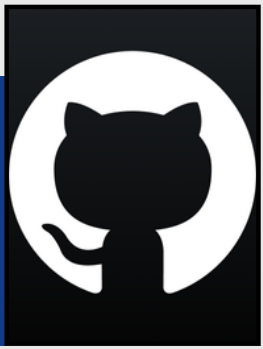
Após fazer alterações em uma branch e estar satisfeito com o resultado, você pode fundi-la de volta à branch principal.

```
$ git checkout branch-principal
$ git merge nome-da-branch
```

RESOLVENDO CONFLITOS DE MERGE

Se houver conflitos durante o merge, o Git irá indicá-los. Você precisará resolver os conflitos manualmente.

CAPÍTULO 4: TRABALHANDO COM REPOSITÓRIOS REMOTOS



CLONANDO UM REPOSITÓRIO EXISTENTE

Para obter uma cópia de um repositório existente do GitHub:

$ git clone url-do-repositorio

ADICIONANDO UM REPOSITÓRIO REMOTO

Para adicionar um repositório remoto ao seu projeto local:

$ git remote add nome-do-remoto url-do-remoto

EMPURRANDO (PUSH) E PUXANDO (PULL) ALTERAÇÕES

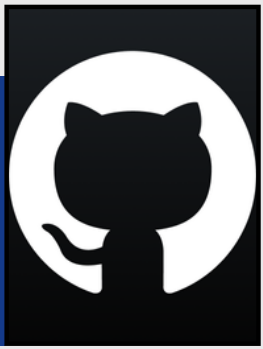
Para enviar suas alterações para o repositório remoto:

$ git push nome-do-remoto nome-da-branch

Para obter as alterações do repositório remoto:

$ git pull nome-do-remoto nome-da-branch

CAPÍTULO 5: TRABALHANDO COM FORKS E PULL REQUESTS NO GITHUB



FORKING DE UM PROJETO NO GITHUB

- Acesse o repositório no GitHub.
- Clique no botão "Fork" no canto superior direito da página. Isso criará uma cópia do repositório em sua própria conta.

CLONANDO E ATUALIZANDO O FORK LOCALMENTE

- Clone o fork para o seu ambiente local:

```
$ git clone url-do-seu-fork
```

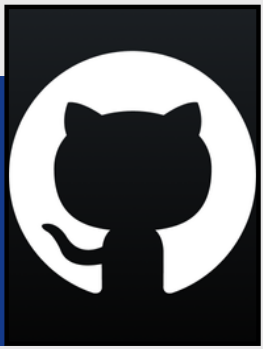
Mantenha seu fork atualizado com o repositório original:

```
$ git remote add upstream url-do-repositorio-original
$ git fetch upstream
$ git checkout main
$ git merge upstream/main
$ git push origin main
```

CRIANDO E GERENCIANDO PULL REQUESTS

- Faça suas alterações em uma branch específica no seu fork.
- Abra o seu fork no GitHub e clique no botão "Pull Request".
- Escolha a branch de origem e a branch de destino.
- Adicione uma descrição e clique em "Create Pull Request".

CAPÍTULO 6: GERENCIANDO VERSIONAMENTO DE CÓDIGO COM TAGS



CRIANDO E LISTANDO TAGS

- Para criar uma tag:

```
$ git tag nome-da-tag
```

- Para listar tags:

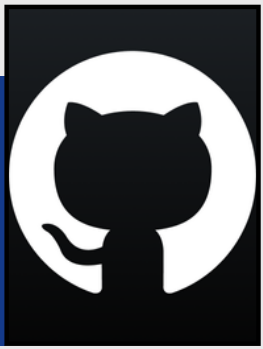
```
$ git tag
```

CHECKOUT DE UMA TAG

Para fazer checkout para uma tag específica:

```
$ git checkout nome-da-tag
```

CAPÍTULO 7: OUTROS COMANDOS ÚTEIS DO GIT



IGNORANDO ARQUIVOS COM .GITIGNORE

Crie um arquivo .gitignore no diretório raiz do seu projeto e liste os arquivos/diretórios que você deseja ignorar.

DESFAZENDO MUDANÇAS COM GIT RESET E GIT REVERT

- git reset:

  $ git reset HEAD~1 *# Desfaz o último commit mantendo as mudanças nos arquivos*

- git revert:

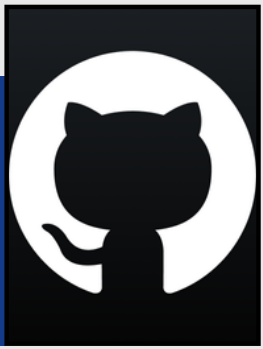
  $ git revert HEAD *# Cria um novo commit que desfaz o último*

VISUALIZANDO DIFERENÇAS COM GIT DIFF

Para visualizar as diferenças entre o seu código atual e o último commit:

$ git diff HEAD

CAPÍTULO 8: BOAS PRÁTICAS E DICAS PARA COLABORAÇÃO



MELHORES PRÁTICAS AO TRABALHAR EM EQUIPE

- Mantenha commits pequenos e focados em uma única funcionalidade.
- Descreva suas mudanças de forma clara e concisa em mensagens de commit.
- Use branches para desenvolver funcionalidades novas ou corrigir bugs.
- Faça pull regularmente para manter-se atualizado com o projeto.

ESTRATÉGIAS PARA ORGANIZAR COMMITS

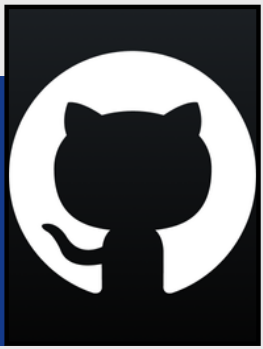
- Use commits atômicos para mudanças pequenas e específicas.
- Combine commits relacionados usando git rebase -i HEAD~n.

USANDO ISSUES E LABELS NO GITHUB

- Crie issues para rastrear tarefas, melhorias ou bugs.

- Aplique labels para categorizar e priorizar as issues.
- Referencie issues nos commits para manter o histórico associado.

CONCLUSÃO

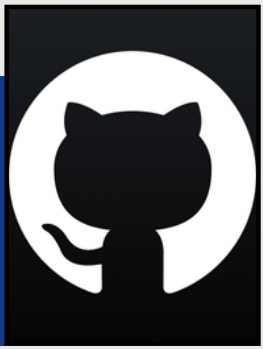


Parabéns! Agora você tem uma compreensão sólida dos conceitos fundamentais do Git e GitHub, bem como das boas práticas ao trabalhar com controle de versão e colaboração em projetos.

Lembre-se de que a prática e a experiência são essenciais para se tornar um profissional habilidoso em Git e GitHub. Continue explorando, colaborando em projetos e aprimorando suas habilidades.

Se precisar de mais alguma coisa ou tiver outras dúvidas, não hesite em entrar em contato. Desejo-lhe muito sucesso em sua jornada com Git e GitHub!

APÊNDICES



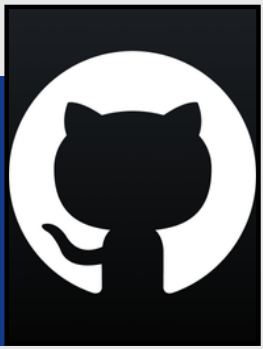
FERRAMENTAS E RECURSOS ÚTEIS

- Site Oficial do Git
- GitHub Learning Lab
- Pro Git Book

GLOSSÁRIO DE TERMOS

- Repositório: Local onde o código de um projeto é armazenado.
- Branch: Uma linha de desenvolvimento separada no repositório.
- Commit: Um registro de uma mudança no repositório.
- Fork: Uma cópia de um repositório em sua própria conta.
- Pull Request: Uma solicitação para incorporar alterações em um repositório.

SOBRE O AUTOR



Olá! Nós somos Hudson Costa Diniz e Fernando Castro Sousa, ambos estudantes de Bacharelado em Ciência e Tecnologia na Universidade Federal do Maranhão (UFMA). Atualmente, estamos focados em explorar o emocionante campo da tecnologia, com ênfase em práticas DevOps.

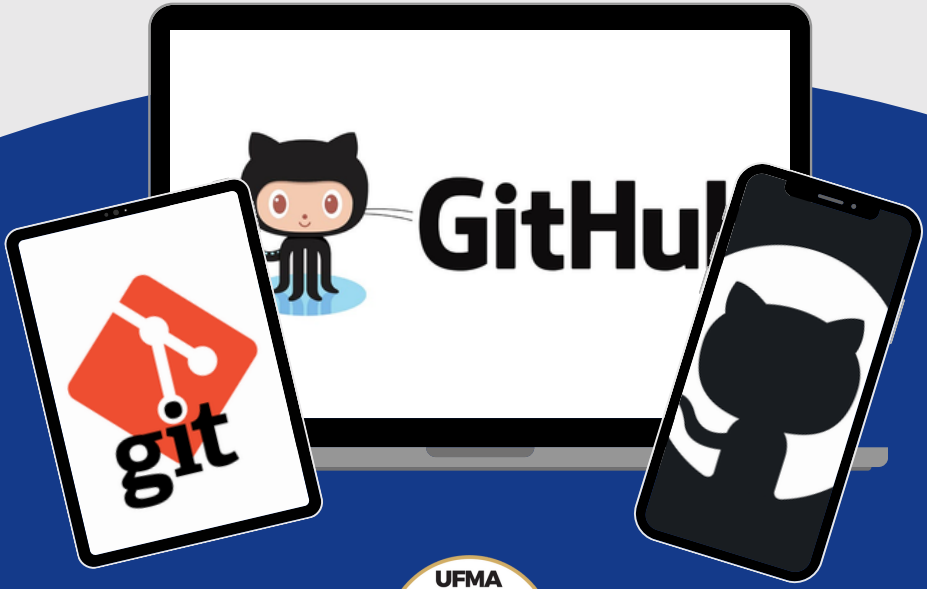
Este eBook sobre "Introdução ao Git e GitHub" é o resultado da nossa dedicação em compartilhar conhecimentos e tornar a complexidade do controle de versão acessível a todos, desde iniciantes até desenvolvedores experientes. A contribuição de ambos, foi fundamental para o sucesso deste projeto.

Fernando Castro Sousa
Hudson Costa Diniz
Estudante de Bacharelado em Ciência e Tecnologia
Universidade Federal do Maranhão (UFMA)

OBRIGADO!

"Que este eBook tenha sido uma fonte valiosa de conhecimento para você no universo do Git e GitHub. Continue explorando, colaborando e criando. Desejo-lhe muito êxito em sua jornada de desenvolvimento. Muito obrigado por ler este eBook!"

-Hudson costa



UFMA
Universidade Federal do Maranhão