Neste módulo, você será introduzido ao Git e GitHub, ferramentas essenciais para controle de versão e colaboração em projetos de tecnologia. Aprenderemos a importância dessas ferramentas e como elas podem resolver problemas comuns enfrentados por profissionais e estudantes:

Instalação e Configuração

Iniciaremos com a instalação do Git em diferentes sistemas operacionais e a criação de um perfil no GitHub. Este passo é fundamental para garantir que você esteja preparado para utilizar essas ferramentas no seu dia a dia profissional.

Comandos Básicos do Git

Você aprenderá os comandos essenciais do Git, como `git init`, `git clone`, `git add`, `git commit`, e `git push`. Esses comandos são fundamentais para o gerenciamento de versões e colaboração em projetos.

Criação e Gerenciamento de Repositórios

Exploraremos como criar e gerenciar repositórios no GitHub, destacando a importância de publicar projetos para portfólios pessoais e trabalho em equipe. Também discutiremos a importância de um README bem estruturado.

Este módulo é um passo crucial na sua jornada para se tornar um cientista de dados competente. As habilidades adquiridas aqui serão fundamentais para colaborar eficazmente em projetos de tecnologia. Continue avançando e prepare-se para o próximo módulo!

VCS – Version Control System

- Registro das alterações realizadas nos arquivos.
- Identificação de qual envolvido no projeto foi responsável pelas alterações.
- Possibilidade de voltar para versão anterior.
- Controle unificado de todas as versões do arquivo.
- Facilidade e ganho de tempo quando é necessário acessar os arquivos de um projeto.

O QUE É O GIT?

O Git é um sistema de controle de versão distribuído amplamente utilizado para o gerenciamento de projetos de software e o rastreamento de mudanças no código-fonte durante o desenvolvimento. É um VCS!

UTILIDADES

- Controle de versionamento
- Rastreamento de alterações
- Compartilhamento de informações
- Colaboração nos projetos
- Auxilia na criação do Portfólio pessoal (no github)
- Um dos VCS mais usados do mundo.

USABILIDADE

Instalado localmente na sua máquina, com ele você pode fazer commits, criar branches, mesclar códigos e muito mais.

O QUE É O GITHUB?

O Github é a ferramenta auxiliar que nos ajuda a tornar os projetos públicos. É uma plataforma de hospedagem de código que permite que desenvolvedores armazenem, colaborem, gerenciem e compartilhem projetos

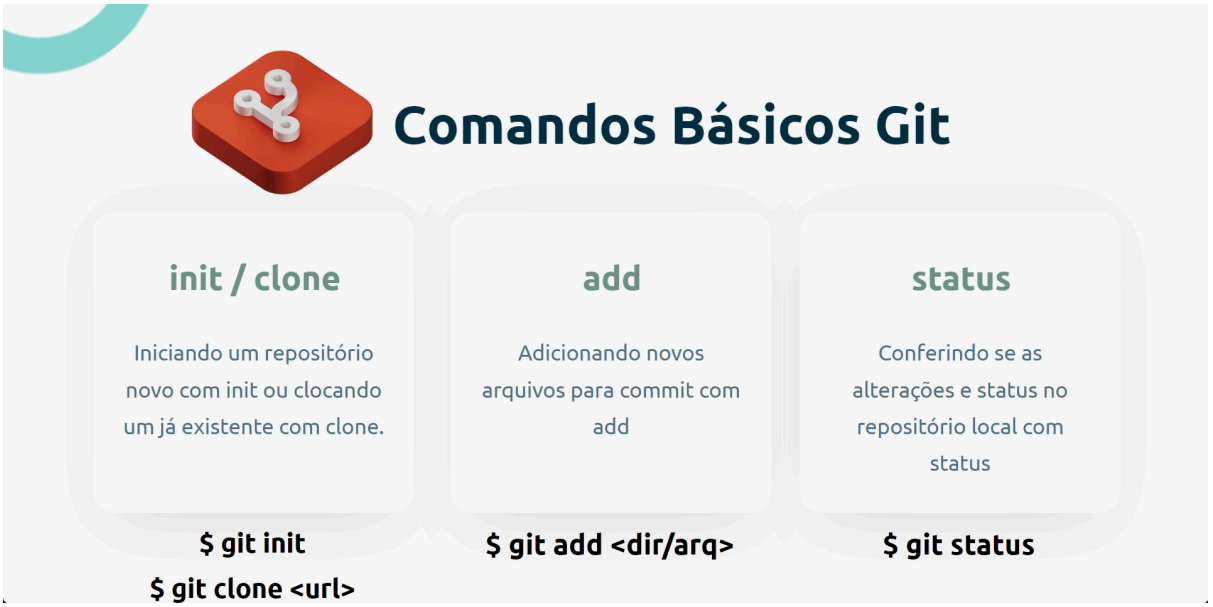
UTILIDADES

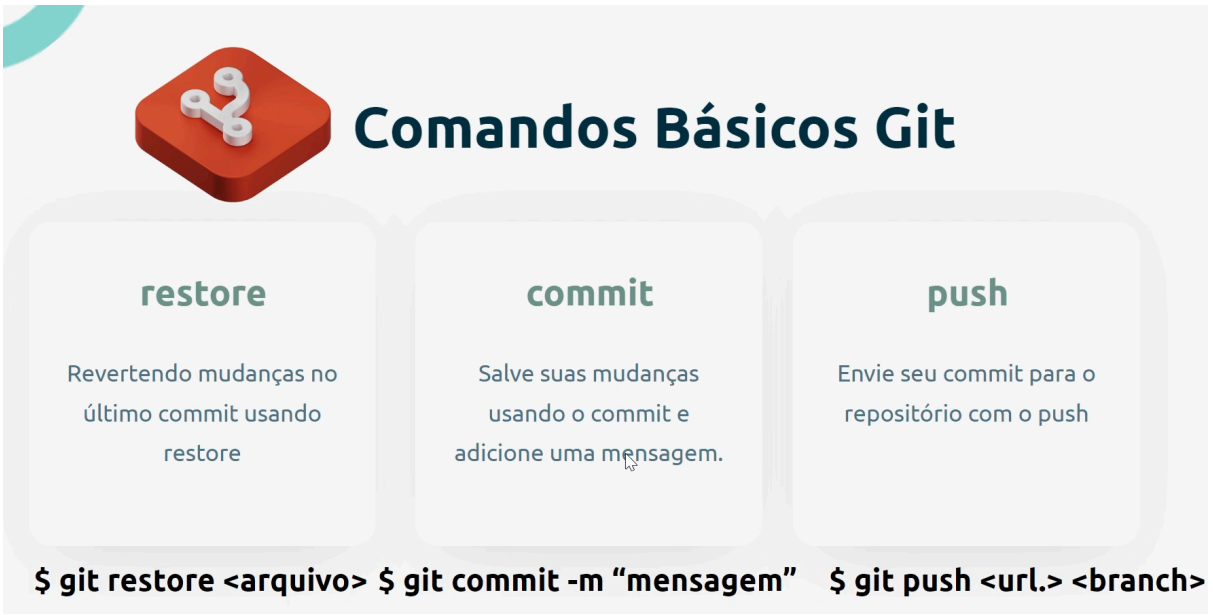
- Hospedagem de repositórios
- Colaboração entre projetos
- Integração com diversas ferramentas
- Comunidade e Descoberta
- Usado como portfólio

USABILIDADE

É um site utilizado de forma online, sem hospedagem local.

meu perfil no github https://github.com/jn036


Comandos Básicos Git
init / clone
Iniciando um repositório novo com init ou clocando um já existente com clone.
add
Adicionando novos arquivos para commit com add
status
Conferindo se as alterações e status no repositório local com status
$ git init
$ git clone <url>
$ git add <dir/arq>
$ git status


Comandos Básicos Git
restore
Revertendo mudanças no último commit usando restore
commit
Salve suas mudanças usando o commit e adicione uma mensagem.
push
Envie seu commit para o repositório com o push
$ git restore <arquivo>
$ git commit -m “mensagem”
$ git push <url.> <branch>

Passo a Passo

1. Criar repositório do github
2. Subir um arquivo manualmente, sem a utilizazção de códigos nesse repositório
3. Subir um arquivo localmente, do github para a máquina
4. Subir um arquivo localmente da máquina para o github

READ ME

É um arquivo comumente encontrado em projetos de TI, com um texto simples contendo uma breve descrição do projeto.

- **Descrição:** Breve explicação do projeto contendo seus objetivos, ferramentas utilizadas.
- **Instruções:** Se seu projeto exige alguma instalação para rodar, é descrito nesse arquivo.
- **Contribuição:** Se seu projeto está aberto para contribuição, isso precisa estar explícito no ReadMe.
- **Créditos:** Indicação de todos os contribuidores do projeto.
- **Licença:** É sempre importante adicionar a legenda da licença do MIT.

READ ME – EXEMPLO

```
Projeto de Análise de Dados de Vendas
Este é um projeto de análise de dados de vendas que visa analisar e visualizar os dados de vendas de uma empresa durante o último ano.
O projeto utiliza técnicas de ciência de dados e visualização de dados para extrair insights valiosos e apresentá-los de forma clara e compreensível.

Estrutura do Projeto
data/: Contém os conjuntos de dados de vendas.
notebooks/: Contém os notebooks Jupyter utilizados na análise de dados.
scripts/: Contém scripts Python para processamento de dados e visualização.
results/: Contém os resultados da análise, como gráficos e relatórios.

Os resultados da análise estão disponíveis em results/ e incluem os seguintes artefatos:

relatorio_analise.pdf: Relatório detalhado da análise de dados.
graficos/: Pasta contendo gráficos gerados durante a análise.
Como Contribuir


Se você deseja contribuir para este projeto, por favor, siga estas etapas:
Crie uma nova branch com sua funcionalidade ou correção de bug (git checkout -b feature/nova-feature).
Faça commit das suas alterações (git commit -am 'Adiciona nova feature').
Faça push para a branch (git push origin feature/nova-feature).
Crie um novo Pull Request.

Créditos: Úrsula Martins - Contato: ursula****@gmail.com

Licença
Este projeto está licenciado sob a Licença MIT.
```

Controle de versão

Um sistema que registra alterações em um arquivo ou conjunto de arquivos ao longo do tempo, permitindo que você retorne a versões específicas mais tarde. É essencial para o desenvolvimento colaborativo de software.

Git

Um sistema de controle de versão distribuído, amplamente utilizado para rastrear mudanças no código-fonte durante o desenvolvimento de software. Ele permite que múltiplos desenvolvedores trabalhem em paralelo.

GitHub

Uma plataforma de hospedagem de código-fonte com controle de versão usando o Git. Oferece funcionalidades de colaboração, como revisão de código, gerenciamento de projetos e integração contínua.

README

Um arquivo de texto que fornece informações sobre outros arquivos em um diretório ou arquivo de projeto de software. Geralmente inclui instruções de instalação, uso e contribuição.

Repositório

Um local centralizado onde os dados são armazenados e gerenciados. No contexto do Git e GitHub, refere-se a um diretório ou espaço de armazenamento onde seu projeto reside.

Utilização de Controle de Versão

É fundamental utilizar o Git para controle de versão em projetos de ciência de dados. Isso permite rastrear alterações no código e nos dados, facilitando a colaboração e a reversão de mudanças quando necessário.

Criação de Repositórios no GitHub

Criar repositórios no GitHub para armazenar e compartilhar projetos de ciência de dados é uma prática recomendada. Isso não só ajuda na organização, mas também serve como um portfólio profissional acessível a recrutadores e colegas.

Documentação com README

Incluir um arquivo README bem estruturado em cada repositório é crucial. Ele deve fornecer uma visão geral do projeto, instruções de instalação, detalhes de contribuição e informações de licença, garantindo clareza e eficácia na comunicação.

Personalização do Perfil no GitHub

Personalizar o perfil no GitHub com informações pessoais e profissionais, além de projetos relevantes, pode aumentar a visibilidade e atratividade para recrutadores. Um perfil bem configurado é essencial para se destacar em processos seletivos.

Exemplo de utilização no mercado de trabalho:

Colaboração em Projetos de Ciência de Dados: Equipes de ciência de dados em empresas utilizam Git e GitHub para colaborar em projetos, permitindo que múltiplos membros trabalhem simultaneamente em diferentes partes do projeto sem conflitos.
Portfólio Profissional: Cientistas de dados usam GitHub para criar portfólios online, onde podem demonstrar suas habilidades e projetos para potenciais empregadores, aumentando suas chances em processos seletivos.

Exemplo de código básico para iniciar um repositório Git:

```
# Inicializa um novo repositório Git
git init

# Adiciona todos os arquivos ao repositório
git add .

# Faz o commit das mudanças com uma mensagem
git commit -m "Initial commit"

# Conecta o repositório local a um repositório remoto no GitHub
git remote add origin https://github.com/usuario/repo.git

# Envia as mudanças para o repositório remoto
git push -u origin master
```

Vamos discutir o código passo-a-passo:

`git init`: Inicializa um novo repositório Git no diretório atual.
`git add .`: Adiciona todos os arquivos do diretório atual ao repositório.
`git commit -m "Initial commit"`: Faz o commit das mudanças com a mensagem "Initial commit".
`git remote add origin`: Conecta o repositório local a um repositório remoto no GitHub.
`git push -u origin master`: Envia as mudanças para o repositório remoto na branch master.

Aprenda a identificar e corrigir os erros mais comuns e torne seu código mais eficiente e confiável.

01

Esquecer de inicializar um repositório Git.

Descrição: Iniciantes frequentemente esquecem de inicializar um repositório Git antes de começar a trabalhar em um projeto, o que impede o uso de comandos Git subsequentes.
Tipo de Erro: Erro de configuração.

Exemplo:

```
# Tentativa de adicionar arquivos sem inicializar o repositório
git add .
```

Correção:

```
# Inicialize o repositório antes de adicionar arquivos
git init
git add .
```

Passo 1: Navegue até o diretório do projeto no terminal.
Passo 2: Execute o comando `git init` para inicializar o repositório.
Passo 3: Agora, adicione arquivos com `git add`.

02

Não configurar o nome de usuário e email no Git.

Descrição: Antes de fazer commits, é necessário configurar o nome de usuário e email no Git. Esquecer de fazer isso pode resultar em commits sem identificação adequada.
Tipo de Erro: Erro de configuração.

Exemplo:

```
# Tentativa de commit sem configuração de usuário
git commit -m "Initial commit"
```

Correção:

```
# Configure o nome de usuário e email antes de fazer commits
git config --global user.name "Seu Nome"
git config --global user.email "seuemail@exemplo.com"
git commit -m "Initial commit"
```

Passo 1: Execute `git config --global user.name "Seu Nome"` para definir seu nome.
Passo 2: Execute `git config --global user.email "seuemail@exemplo.com"` para definir seu email.
Passo 3: Agora, você pode fazer commits com identificação adequada.

03

Confundir `git pull` com `git fetch`.

Descrição: Iniciantes podem confundir os comandos `git pull` e `git fetch`, levando a conflitos de mesclagem inesperados.
Tipo de Erro: Erro lógico.

Exemplo:

```
# Uso de git pull sem querer mesclar mudanças
git pull origin main
```

Correção:

```
# Use git fetch para apenas buscar mudanças
git fetch origin
git merge origin/main
```

Passo 1: Use `git fetch` para buscar mudanças sem mesclar.
Passo 2: Revise as mudanças buscadas.
Passo 3: Use `git merge` para mesclar as mudanças quando estiver pronto.

04

Ignorar arquivos importantes no .gitignore.

Descrição: Iniciantes podem acidentalmente adicionar arquivos importantes ao `.gitignore`, impedindo que sejam rastreados pelo Git.
Tipo de Erro: Erro de configuração.

Exemplo:

```
# .gitignore com arquivos importantes
*.env
*.config
```

Correção:

```
# Revise o .gitignore para garantir que apenas arquivos
irrelevantes sejam ignorados
*.log
*.tmp
```

Passo 1: Revise o arquivo `.gitignore` regularmente.
Passo 2: Remova entradas que não deveriam ser ignoradas.
Passo 3: Adicione apenas arquivos que não devem ser rastreados.

05

Não criar um README adequado para o projeto.

Descrição: Um erro comum é não criar um arquivo README ou criar um que não forneça informações suficientes sobre o projeto.
Tipo de Erro: Erro de documentação.

Exemplo:

```
# Projeto Exemplo
```

Correção:

```
# Projeto Exemplo

## Descrição
Este projeto faz XYZ...

## Instalação
1. Clone o repositório.
2. Instale as dependências.

## Uso
Execute o seguinte comando...

## Contribuição
Siga as diretrizes de contribuição.

## Licença
Este projeto está sob a licença XYZ.
```

Passo 1: Inclua uma descrição clara do projeto.
Passo 2: Adicione instruções de instalação e uso.
Passo 3: Forneça informações sobre contribuição e licença.

Se você não encontrar o erro específico nesta lista, consulte a documentação oficial do Git e GitHub para obter mais informações, ou pergunte aos tutores do curso.

O que é o Git e por que é importante?

O Git é um sistema de controle de versão amplamente utilizado que permite o rastreamento de mudanças no código-fonte e facilita a colaboração entre membros de uma equipe. Ele é importante porque resolve problemas comuns, como a perda de versões anteriores de arquivos e a desorganização em ambientes colaborativos.

Como o GitHub complementa o Git?

O GitHub é uma plataforma online que complementa o Git ao permitir o armazenamento e compartilhamento de projetos. Ele também serve como um portfólio profissional, onde desenvolvedores podem exibir seus projetos e colaborar com outros.

Quais são os passos iniciais para começar a usar o Git e o GitHub?

Os passos iniciais incluem a instalação do Git em seu sistema operacional e a criação de um perfil no GitHub. Esses passos são fundamentais para começar a gerenciar projetos e colaborar com outros desenvolvedores.

Por que é importante personalizar o perfil no GitHub?

Personalizar o perfil no GitHub é importante porque torna sua apresentação mais atraente para recrutadores. Incluir informações pessoais e profissionais, além de personalizar o perfil com códigos, pode ajudar a destacar suas habilidades e projetos.

Quais são alguns dos comandos básicos do Git que devo conhecer?

Alguns dos comandos básicos do Git que você deve conhecer incluem `git init`, `git clone`, `git add`, `git commit`, e `git push`. Esses comandos são essenciais para iniciar e gerenciar repositórios, adicionar mudanças, e enviar atualizações para o GitHub.

Qual é a importância do arquivo README em um projeto?

O arquivo README é importante porque fornece uma visão geral do projeto, instruções de instalação, detalhes de contribuição, créditos e informações de licença. Um README bem estruturado é essencial para garantir a clareza e eficácia na comunicação do projeto.

Como posso criar e gerenciar repositórios no GitHub?

Você pode criar e gerenciar repositórios no GitHub através da interface da plataforma. Isso envolve a criação de um novo repositório, a clonagem para seu ambiente local, e o uso de comandos Git para gerenciar o conteúdo e colaborar com outros.

Por que publicar projetos no GitHub é importante para meu portfólio?

Publicar projetos no GitHub é importante para seu portfólio porque demonstra suas habilidades e experiência em desenvolvimento de software. Isso pode ser um diferencial em processos seletivos, mostrando aos recrutadores seu trabalho e capacidade de colaborar em projetos.

Como o Git e o GitHub ajudam na colaboração em equipe?

O Git e o GitHub ajudam na colaboração em equipe ao permitir que vários desenvolvedores trabalhem no mesmo projeto simultaneamente, rastreando mudanças e gerenciando versões de forma eficiente. Isso facilita a comunicação e coordenação entre membros da equipe.

Qual é a importância de ter um perfil bem configurado no GitHub?

Ter um perfil bem configurado no GitHub é importante porque pode ajudar a se destacar em processos seletivos. Um perfil atraente e bem organizado demonstra profissionalismo e pode atrair a atenção de recrutadores e potenciais empregadores.