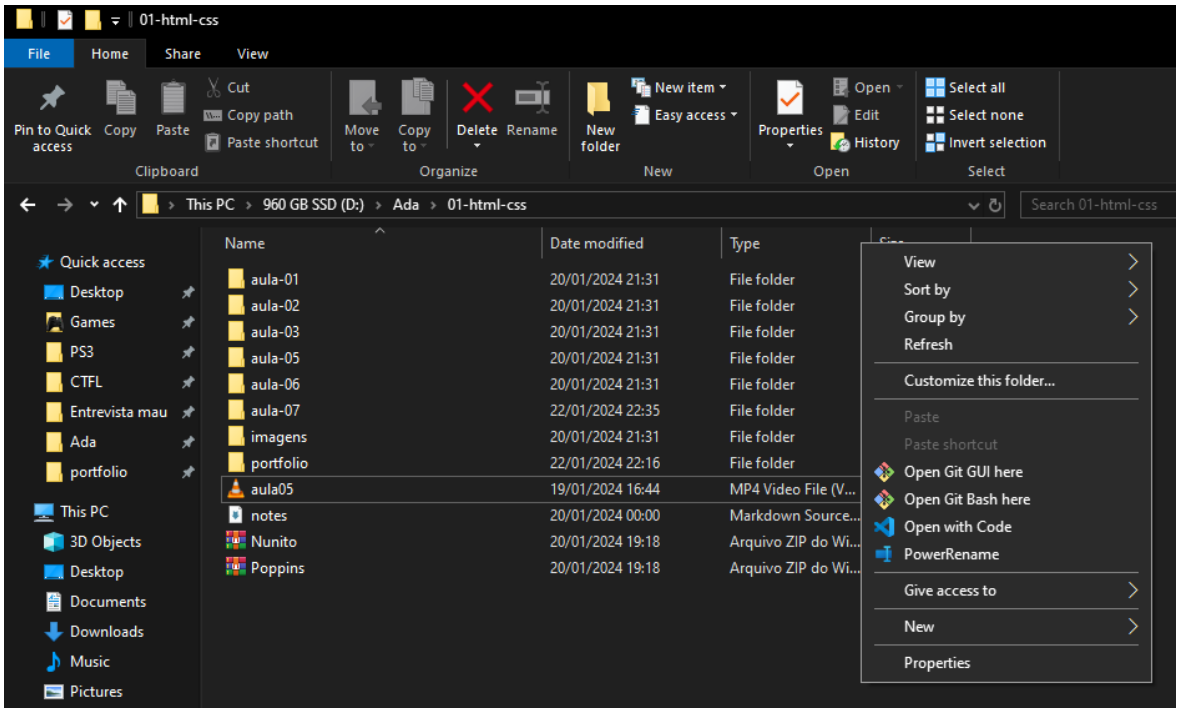
COMMITS PARA A BRANCH DEV!

Repositório do projeto: https://github.com/mcalheiro/front-end-projeto-em-grupo

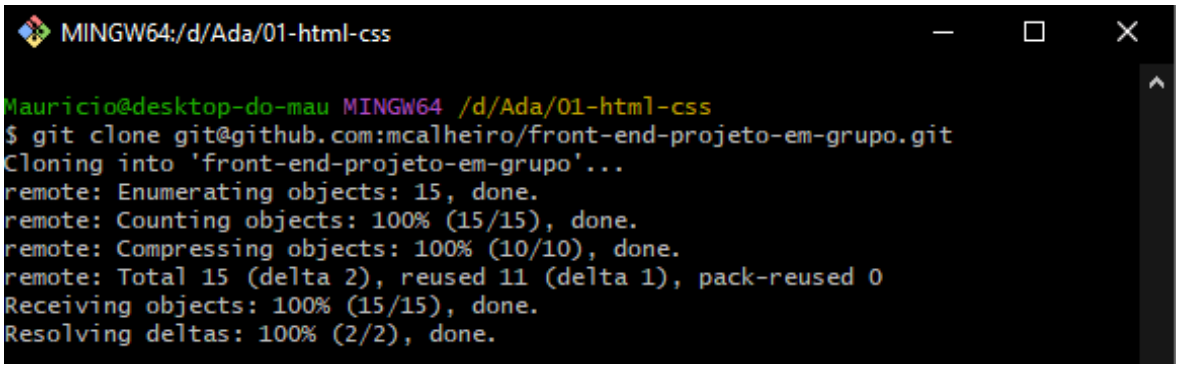
Vá até a pasta onde deseja clonar o repositório abra um terminal. Pode ser o CMD, git bash ou o que preferir. Aqui no exemplo. Clonarei dentro da pasta “01-html-css”.



No terminal (git bash, CMD, ou que preferir...) faça o seguinte comando:

```
git clone git@github.com:mcalheiro/front-end-projeto-em-grupo.git
```

Estou usando o git bash por preferência, mas pode até ser o terminal do vscode. Se tudo der você verá uma mensagem similar a esta e será criada uma pasta chamada “front-end-projeto-em-grupo”, que é o nome do repositório lá no github.



⚠ Na primeira vez que fizer isso pode aparecer uma mensagem para confirmar fingerprint. Digite “yes”.

Agora, no terminal, abra a pasta do repositório:

```
cd front-end-projeto-em-grupo
```

Note que ao final da linha que indica o diretório atual tem (main), indicando que você está em um repositório do git na *branch* “main”.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css
$ cd front-end-projeto-em-grupo/

Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (main)
$
```

Vamos deixar essa *branch* apenas para o projeto concluído. Então, mude para a *branch* “dev”:

```
git checkout dev
```

Note que agora, no final da linha que indica o diretório, tem (dev), indicando que você está na *branch* “dev”.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (main)
$ git checkout dev
Switched to a new branch 'dev'
branch 'dev' set up to track 'origin/dev'.

Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$
```

A partir de agora, troquei para o terminal do vscode (só pq quis), o processo continua e ainda funcionaria em qualquer outro terminal. Após mudar para a *branch* “dev”, atualize o conteúdo (vai que alguém fez um *commit* logo após você clonar). Para isso, use o comando a seguir:

```
git pull
```

Se for preciso, sua *branch* local será sincronizada com a remota. No meu caso, não havia mudanças e recebi a seguinte mensagem:

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$ git pull
Already up to date.

Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$
```

Agora que você está na *branch* “dev” e com tudo sincronizado, crie sua própria *branch* para poder fazer alterações. Esta *branch* nova pode ter o nome que desejar e será uma cópia da *branch* “dev” neste ponto. O comando a seguir cria a *branch* “exemplo” e já muda para ela:

```
git checkout -b exemplo
```

Note que a linha do endereço termina com (exemplo), o nome da nova *branch*.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$ git checkout -b exemplo
Switched to a new branch 'exemplo'

Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (exemplo)
$
```

Faça as alterações necessárias na *branch* de trabalho (neste caso, exemplo). Eu alterei o README.md apenas para efeitos demonstrativos. É possível verificar o que foi modificado com o comando git status.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (exemplo)
$ git status
On branch exemplo
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

Adicione o que quiser enviar, com o comando git add <NOME_DO_ARQUIVO>. É possível adicionar diversos arquivos de uma vez. Basta passar os nomes como parâmetros separados por espaço.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (exemplo)
$ git add README.md
```

git add README.md

Após adicionar, basta “commitar”, com o comando git commit -m “MENSAGEM REPRESENTATIVA DA MUDANÇA”.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (exemplo)
$ git commit -m "Editing README"
[exemplo 4480b3c] Editing README
 1 file changed, 1 insertion(+), 1 deletion(-)
```

git commit -m “Editing README”

Agora, que o *commit* está feito, volte para a *branch* dev com git checkout dev. Perceba que a linha do diretório termina em (dev).

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (exemplo)
$ git checkout dev
Switched to branch 'dev'
Your branch is up to date with 'origin/dev'.

Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
```

Para trazer as mudanças feitas na *branch* de trabalho para a dev, faça um merge: git merge BRANCH_QUE_QUERO_MESCLAR

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$ git merge exemplo
Updating 80b760f..4480b3c
Fast-forward
 README.md | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

git merge exemplo

Se desejar, pode verificar o estado da *branch* dev após o merge. Note que, nesse caso, a branch “dev” local está um *commit* adiantado da “origin/dev”, que é a *branch* “dev” remota que, hospedada no github.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$ git status
On branch dev
Your branch is ahead of 'origin/dev' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

git status

Para mais detalhes, é possível verificar com o comando git log um histórico de commits. Perceba que o *commit* “Editing README” que foi feito lá na *branch* “exemplo” foi trazido para a *branch* “dev” ao fazer merge.

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$ git log
commit 4480b3cc3869b412814128a4d2ad9a5dc7da6db7 (HEAD -> dev, exemplo)
Author: Mauricio Calheiro <mauriciocalheiro95@gmail.com>
Date:   Thu Jan 25 22:12:09 2024 +0000

    Editing README

commit 80b760f62a30f781ed9a3d6dbf9a8cd89964537d (origin/dev, mauricio)
Author: Mauricio Calheiro <mauriciocalheiro95@gmail.com>
Date:   Thu Jan 25 21:19:59 2024 +0000

    Adding assets directory

commit 2f6414a1525613694f121f7677c505fda45de7de
Author: Mauricio Calheiro <mauriciocalheiro95@gmail.com>
Date:   Thu Jan 25 21:16:36 2024 +0000

    Initial file structure
```

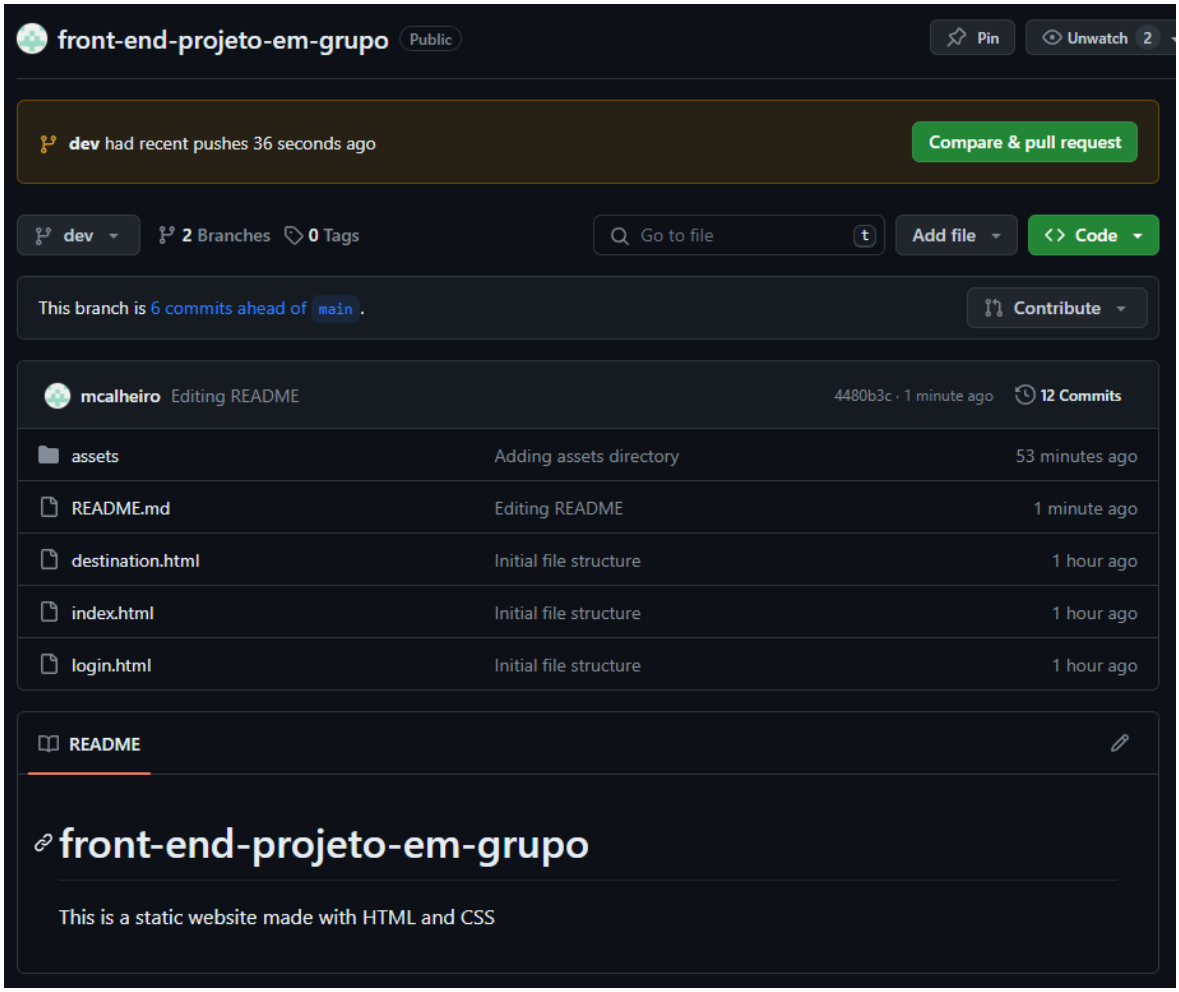
git log

Por fim, faça um push para enviar do repositório local para o remoto:

```
Mauricio@desktop-do-mau MINGW64 /d/Ada/01-html-css/front-end-projeto-em-grupo (dev)
$ git push origin dev
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 307 bytes | 307.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:mcalheiro/front-end-projeto-em-grupo.git
   80b760f..4480b3c  dev -> dev
```

git push origin dev

Se não houver conflitos, você verá a mensagem acima e (se desejar) pode visitar o repositório e perceberá a notificação de que a *branch* “dev” teve pushes recentes.



Agora, faça um pull na *branch* dev novamente, e a partir dela crie outra *branch* para trabalhar. Idealmente, crie uma *branch* para cada nova mudança que for implementar, sempre a partir da *branch* "dev" mais atualizada.

⚠ E SE HOUVER CONFLITOS? ⚠

Chama os amigos no grupo.

⚠ E SE DER QUALQUER OUTRO ERRO? ⚠

Chama os amigos no grupo.

⚠ PREFIRO USAR A INTERFACE DO GITHUB DESKTOP E TENHO DÚVIDAS ⚠

Chama o João Marcos, ele sabe bem.

Instale o git: https://git-scm.com/download/win

Faça sua chave SSH: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Adicione a chave no github: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account