

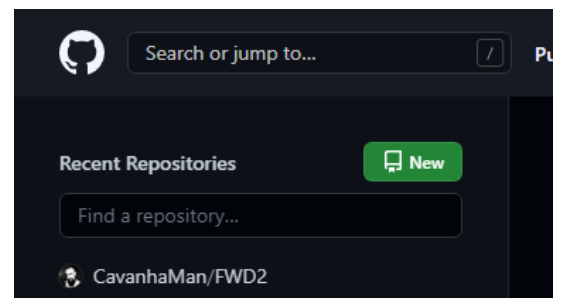
GitHub

Vamos a um passo-a-passo para iniciar o seu projeto do zero:

1) Acesso o GitHub: <https://github.com/> e clique em **sign in** para efetuar login.

2) Clique no botão “**New**” para criar um novo repositório

Repositório nada mais é do que uma pasta para um projeto – pelo próprio GitHub: “*Você pode armazenar vários projetos nos repositórios do GitHub, incluindo projetos de código aberto. Com projetos de código aberto, você pode compartilhar código para tornar o software melhor e mais confiável. Você pode usar repositórios para colaborar com outras pessoas e acompanhar seu trabalho.*”



3) **Insira um nome para o repositório** – é recomendável que o nome seja **referente ao projeto** e um nome “profissional” – lembre-se que seu projeto poderá ser configurado como “público” e poderá ser visto por outras pessoas (inclusive potenciais contratantes!). Também é recomendável que este nome seja o mesmo nome

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *

Repository name *



CavanhaMan

/

Great repository names are short and memorable. Need inspiration? How about **crispy-octo-carnival**?

utilizado no nome da pasta local onde o seu projeto se encontra. Se criar um nome diferente, poderá haver algum esquecimento e você poderá confundir os projetos.

4) **Adicione uma descrição para o seu projeto** – é opcional, vale mais como uma identificação para si mesmo ou, no caso de querer que outros vejam o seu projeto, uma descrição para aqueles que forem acessar seu projeto.

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.

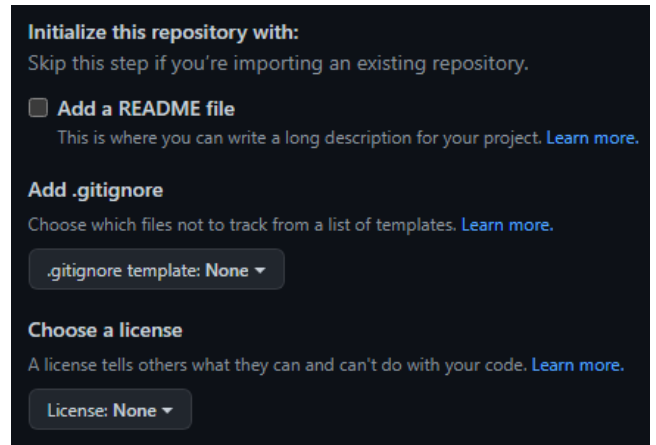


Private

You choose who can see and commit to this repository.

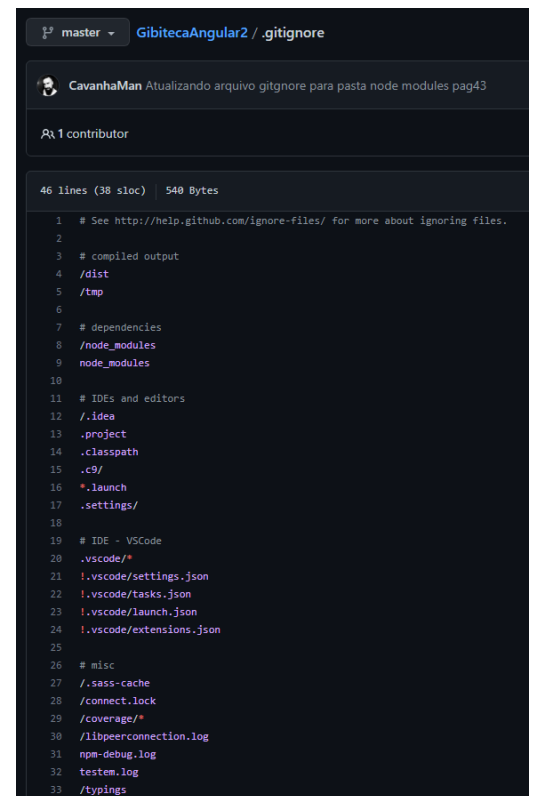
5) Defina se seu repositório será **público** ou **privado**. Os repositórios públicos podem ser acessados por todos na internet. Os repositórios privados só podem ser acessados por você, pelas pessoas com as quais você compartilha explicitamente o acesso e, para repositórios da organização, por determinados integrantes da organização.

6) Escolha se vai iniciar o seu repositório com um arquivo “**leiam**” adicionando um **README** file – novamente, caso seja um projeto público, é interessante deixar mais informações sobre o projeto para aqueles que foram vê-lo. Caso seu objetivo seja criar um portfólio para empresas, é muito recomendável essa prática, explicando o que foi feito no projeto, qual o seu objetivo e etc. Se for um repositório profissional para um projeto de uma contratante, pode haver relatórios que deverão ser inseridos neste arquivo, conforme for orientado.



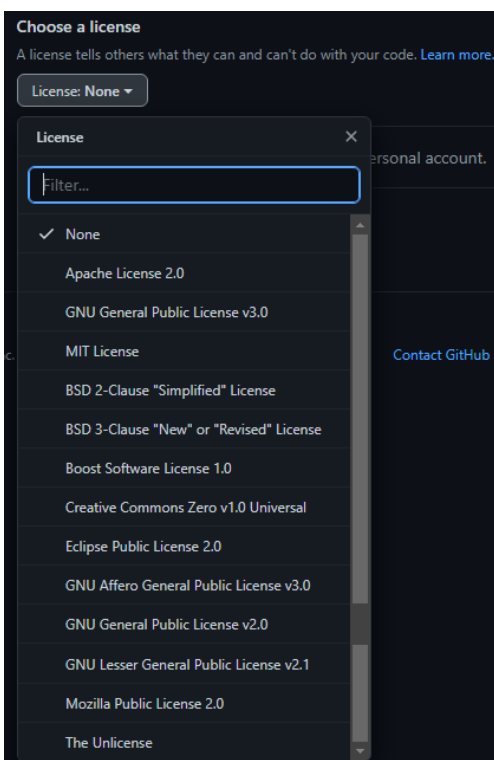
7) Adicionar um arquivo “**.gitignore**”. Dependendo da plataforma utilizada, muitos arquivos de sistema poderão ser gerados ou mesmo arquivo que não são úteis para o projeto, por exemplo, arquivos baixados pelo bootstrap para o caso de projetos de webdesign devem ser ignorados pois podem ser acessados online de forma mais rápida e local (Content Delivery Network – CDN), além de serem “gerados” toda vez que for necessário através dos comandos npm. Para isso, as pastas ou arquivos que deverão ser ignorado pelo github, deverão estar dentro do arquivo **.gitignore**. Ou seja, quando você for fazer os **commits**, estes arquivos não serão enviados para o seu repositório online.

Outro caso específico é que poderemos usar chaves ou senhas para acessar determinados serviços através do nosso projeto, como por exemplo, acessando APIs que tenham controle de acesso e estas chaves não podem ser commitadas no projeto, pois são particulares e intransferíveis na maioria das vezes, neste caso, as chaves ou senhas deverão ser colocadas em arquivos separados que serão lidos durante a execução do projeto (isso pro caso de não serem



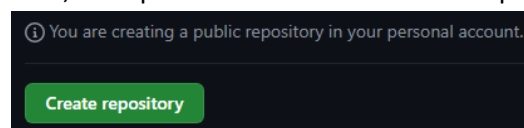
criptografadas). Neste caso, estes arquivos também deverão ser incluídos no arquivo **.gitignore**.

Um exemplo de um arquivo **.gitignore** em um projeto pode ser visto na imagem ao lado.

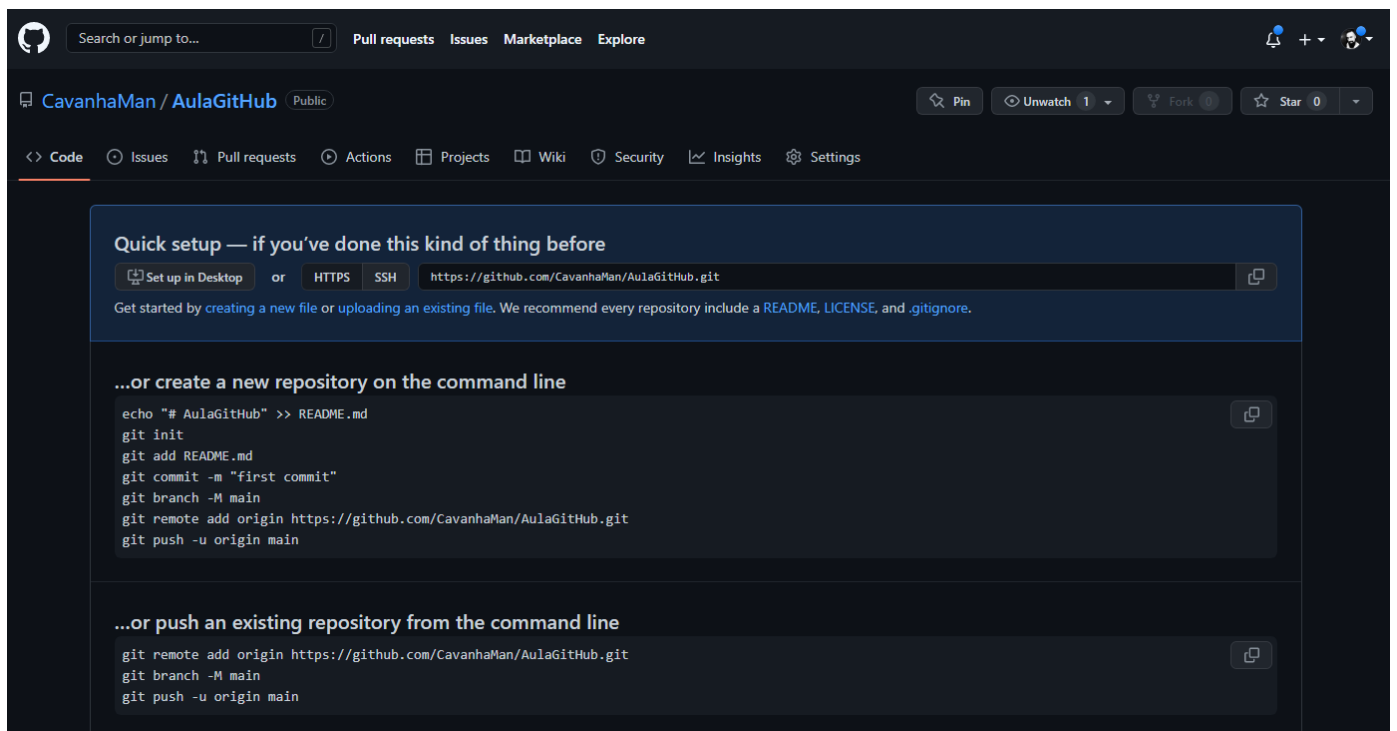


8) Você também pode escolher uma **licença** para o uso do repositório, ou seja, se o seu projeto for público, ele poderá ser compartilhado sob uma licença. Por exemplo, a GNU – *General Public License* – é a designação da licença de software para software idealizada por Richard Matthew Stallman em 1989, no âmbito do projeto GNU da *Free Software Foundation*.

9) Depois de tudo, você pode finalmente criar o seu repositório!



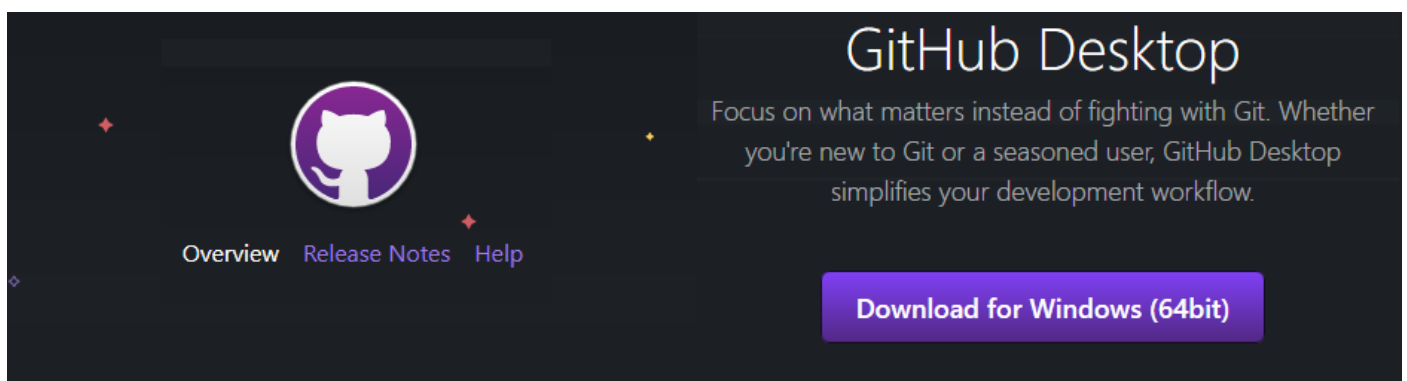
Depois de criado, você verá uma tela assim:



No topo vemos o nome do repositório e alguns comandos e opções. Temos também o caminho do repositório para acesso via HTTPS OU SSH: <https://github.com/CavanhaMan/AulaGitHub.git>

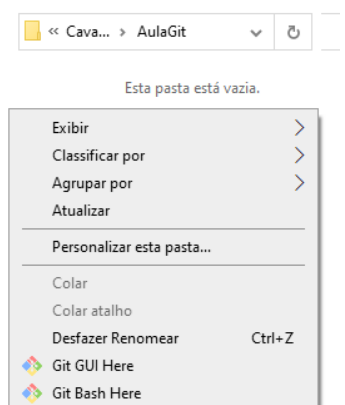
E também temos uma série de comandos que poderão ser usados para iniciar a nossa pasta local do projeto. Vamos agora a esta parte!

- 10) Primeiramente, vamos instalar o **GitHub Desktop** no nosso computador! Para isso, acesse site: <https://desktop.github.com/> e faça o download e instalação (recomenda-se reiniciar o computador depois disso).



- 11) Após a instalação, vamos criar a pasta do projeto na nossa máquina. Escolha o local e crie a pasta com o nome que você deseja – conforme recomendado, é ideal que seja o mesmo nome do repositório criado online.

- 12) Dentro da pasta criada, clique com o botão direito do mouse e clique na opção **“Git Bash Here”**. Será aberta uma tela preta semelhante ao Prompt do DOS (ou ao Shell do Linux). É nesta tela que iremos inserir os comandos.



- 13) Se for a primeira vez que está usando o git bash na sua máquina, será necessária uma configuração básica da primeira vez. No prompt, digite os seguintes comandos:

```
git config --global user.name "seunomegit"
git config --global user.email "seuemail@email.com.br"
```

Lembrando que o nome de usuário e o email usados devem ser os mesmos informados no site do GitHub. Este usuário e email serão utilizados para todos os commits dados a partir desta máquina! Cuidado ao inserir isso em máquinas públicas pois o próximo usuário poderá dar commit em seus próprios projetos usando seu usuário sem querer – principalmente se a chave de acesso também for inserida (o que, é claro, não é recomendável).

Se você ainda não tem uma chave gerada no GitHub, será necessária a geração para a autenticação:

13.1) Abra Git Bash.

13.2) Cole o texto abaixo no prompt, substituindo o endereço de e-mail pelo seu GitHub:

```
ssh-keygen -t ed25519 -C "seuemail@email.com.br"
```

Observação: Se você estiver usando um sistema legado que não é compatível com o algoritmo Ed25519, use:

```
ssh-keygen -t rsa -b 4096 -C "seuemail@email.com.br"
```

Isto cria uma nova chave SSH, usando o nome de e-mail fornecido como uma etiqueta.

13.3) Quando aparecer a solicitação *"Enter a file in which to save the key"* (Insira um arquivo no qual salvar a chave), pressione Enter. O local padrão do arquivo será aceito.

```
Enter a file in which to save the key (/c/Users/you/.ssh/id_algorithm):[Press enter]
```

13.4) Digite uma frase secreta segura no prompt:

```
Enter passphrase (empty for no passphrase): [Type a passphrase]
```

```
Enter same passphrase again: [Type passphrase again]
```

Em caso de dúvida, acesse o manual pelo link: <https://docs.github.com/pt/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>

- 14) Agora podemos colar os comandos que estão no site do git:

Inicia a pasta do repositório:

```
git init
```

Cria um arquivo readme no projeto (opcional):

```
git add README.md
```

Cria a conexão da sua pasta local com o seu repositório online:

```
git remote add origin https://github.com/CavanhaMan/AulaGitHub.git
```

Cria o “ramo principal” do seu projeto:

```
git branch -M main
```

Adiciona todos os arquivos no seu projeto:

```
git add .
```

Cria o seu primeiro commit:

```
git commit -m "Primeiro commit do meu projeto"
```

Envie os arquivos:

```
git push -u origin main
```

Se pedir sua chave, você deverá colar a chave SSH gerada.

Lembrando que a sua pasta local não pode estar vazia – ou seja, deve haver algum arquivo lá para ser “commitado”.

Se der algum erro, tente forçar o diretório a ser alterado para main:

```
git checkout -b main
```

(antigamente era usado “master” e pode haver algum erro)

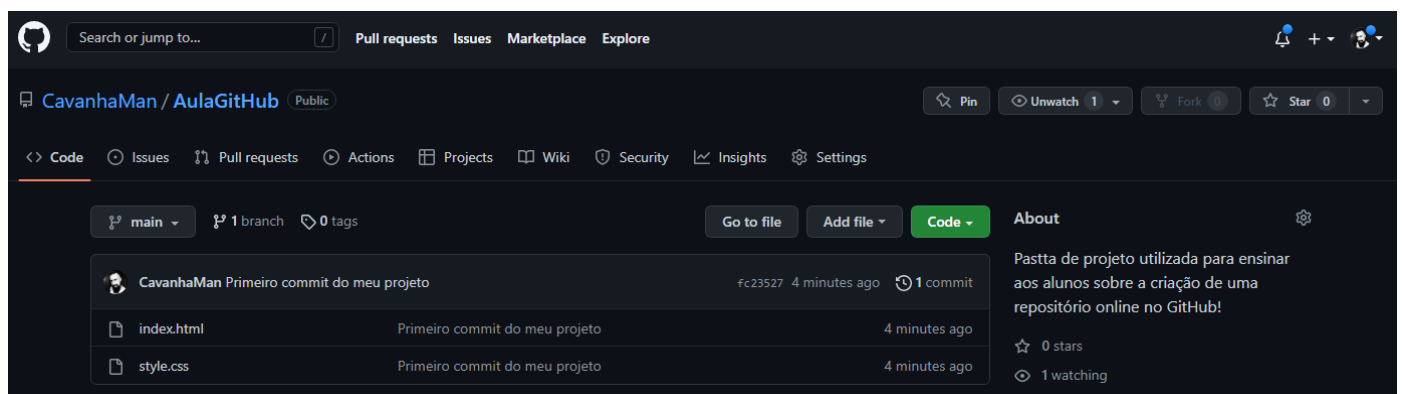
Se tudo der certo, teremos o resultado:

```
MINGW64/d/Cavanha/AulaGit
cavan@DarkTower MINGW64 /d/Cavanha/AulaGit (main)
$ git add .

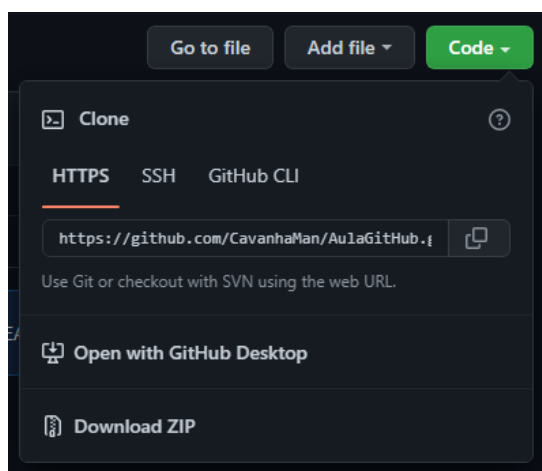
cavan@DarkTower MINGW64 /d/Cavanha/AulaGit (main)
$ git commit -m "Primeiro commit do meu projeto"
[main (root-commit) fc23527] Primeiro commit do meu projeto
2 files changed, 141 insertions(+)
create mode 100644 index.html
create mode 100644 style.css

cavan@DarkTower MINGW64 /d/Cavanha/AulaGit (main)
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.64 KiB | 838.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/CavanhaMan/AulaGitHub.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

E ao atualizarmos no site, teremos:



Se quiser compartilhar com alguém o seu repositório, basta enviar o link que foi usado no passo 14, ou ir clicar no botão **code...**



...e copiar o link mostrado em HTTPS.

Se quiser, ainda pode fazer Download de um arquivo zip com todo o conteúdo do seu repositório.

Ou **clonar** o repositório para a sua máquina! (o que também funciona com repositórios de terceiros!)