### ***Manual Git e GitHub***

O Git é um **sistema de controle de versão dos códigos**. Ele serve para acompanhar as mudanças no código de um projeto, e facilita o trabalho em equipe.

### ***Como funciona:***

No git **temos um repositório**, onde o projeto e suas versões ficam armazenados

Nese repositório, **fazemos Commits que são mudanças com uma mensagem explicando o que foi feito**

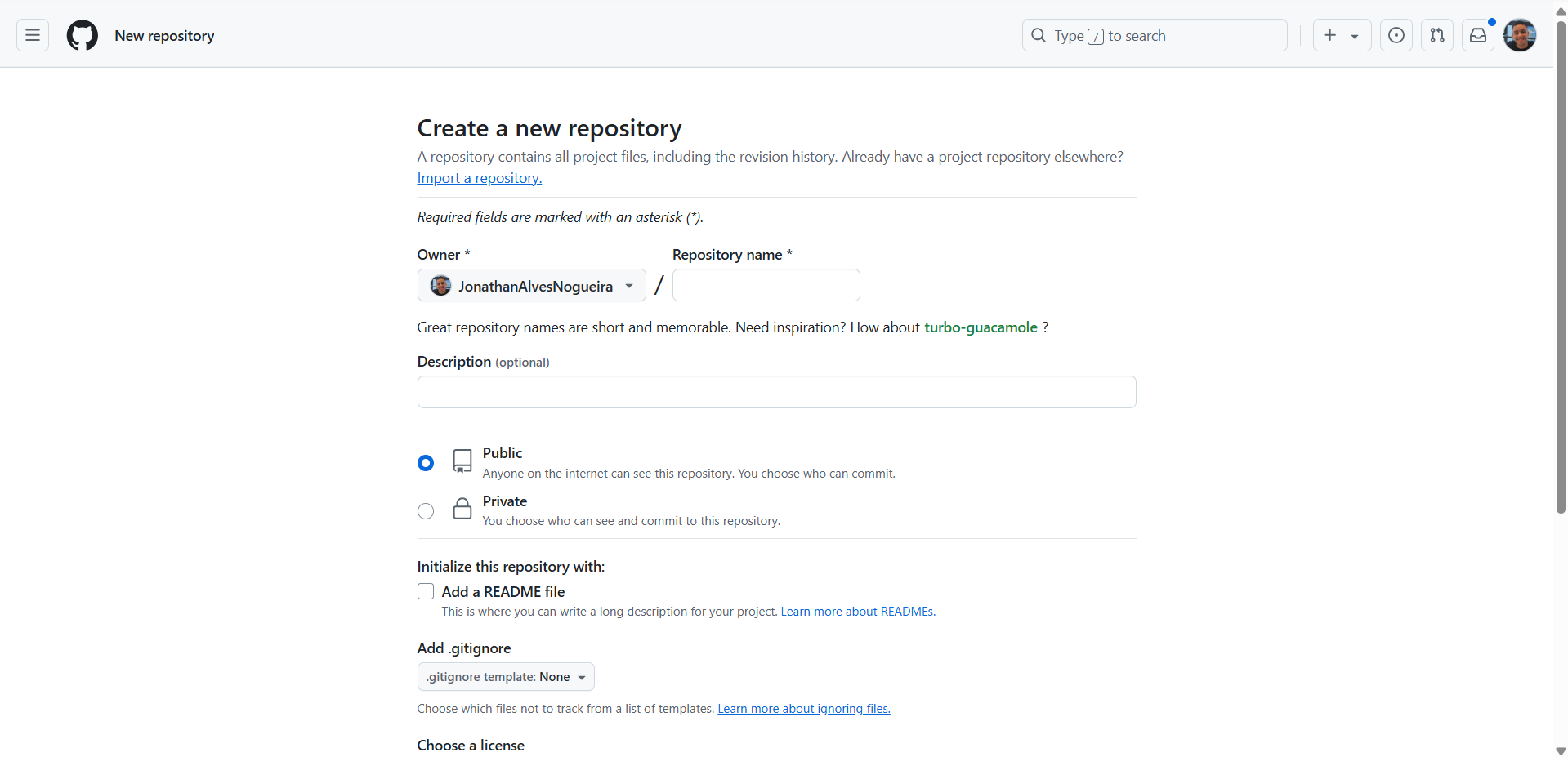
Nesse repositório, **também temos Branch, que são versões paralelas para testar ou mexer em algo dos códigos.**

**E por fim temos o merge, depois de testar em uma Branch seu código, a ideia principal e colocar esse código, no repositório final, para isso serve o Merge, ele une as duas versões.**

### ***Git é diferente de GITHUB***

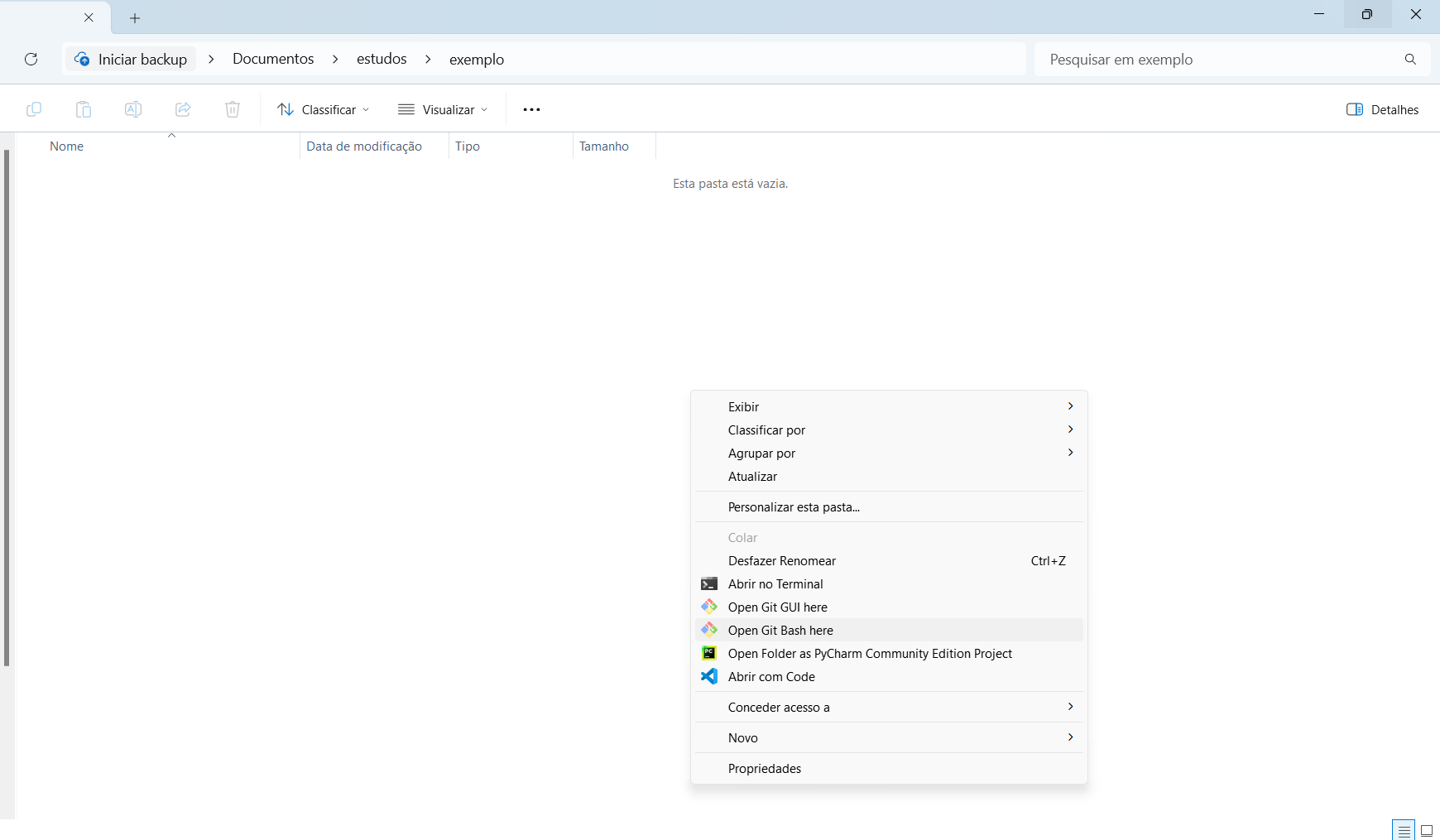
GitHub é um serviço para gerenciar projetos, e códigos.

Para criar um repositório. Acesse o GitHub, logue na sua conta, e clique repositório e depois new

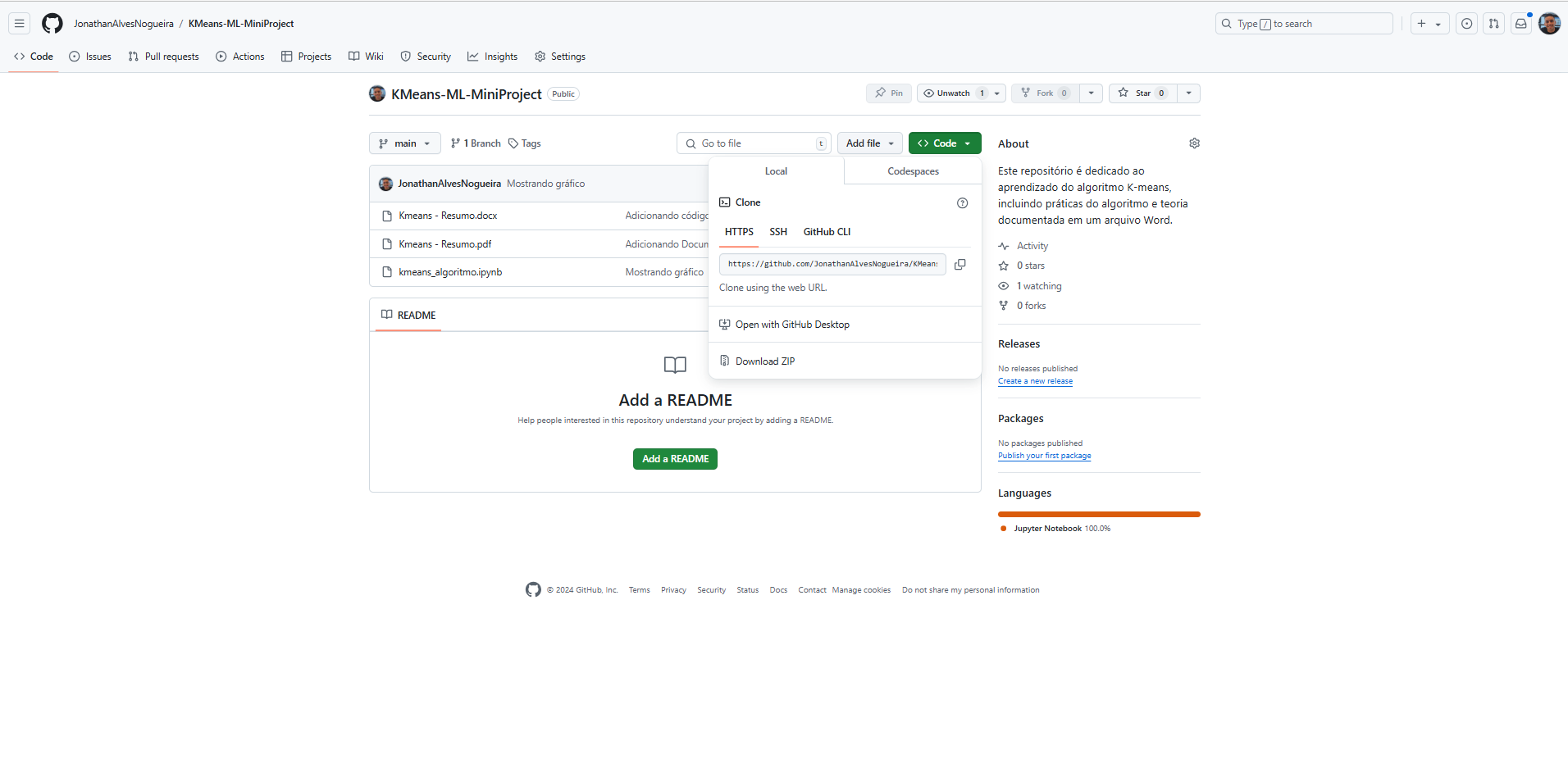
Chegando nessa parte, escreva o nome do seu repositório, e uma breve descrição de qual a finalidade desse projeto.

**Após isso, precisaremos linkar esse projeto criado com sua máquina. Para isso abra a ferramenta Git Bash dentro de uma pasta do seu computador de sua preferência**, e siga os seguintes passos:

Clique com o botão direito e depois selecione a opção Git Bash Here

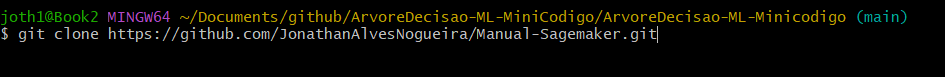


* **Copie a URL do repositório no GitHub (botão Code > HTTPS).**



* Abra o terminal e digite:

***git clone <URL\_DO\_REPOSITORIO>***



* Pressione **Enter** e o repositório será baixado para seu computador.

Com isso feito, é possível começar a usar o GitHub para armazenar seus códigos.

O Git possui alguns comandos principais que precisamos ter em mente para uma plena utilização, sendo eles:

***Add, Commit, Pull, Push, checkout, Merge.***

Abaixo deixo a explicação de todos os comandos.

### ***Comandos GIT***

### **git add**

* **O que faz:** Prepara os arquivos para serem confirmados no próximo commit.

**Uso:**

git add . # Adiciona todos os arquivos modificados

### **git commit**

* **O que faz:** Cria uma versão das mudanças e salva no histórico do repositório.

**Uso:**

git commit -m "Mensagem descritiva do commit"

**Descrição:** Registra as alterações que estavam na *staging area* com uma mensagem.

### **git log**

* **O que faz:** Exibe o histórico de commits.

**Uso:**

git log

git log --oneline # Histórico resumido

* **Descrição:** Permite navegar pelos commits anteriores e ver detalhes como autor e data.

### **git checkout**

* **O que faz:** Troca de branch ou restaura versões de arquivos.

**Uso:**

git checkout <nome-da-branch> # Mudar para outra branch  
git checkout <arquivo> # Restaurar um arquivo modificado

**Descrição:** Permite navegar entre branches ou desfazer modificações em arquivos.

### **Git pull**

* **O que faz:** Baixa e integra as mudanças do repositório remoto para o local.

**Uso:**

git pull

**Descrição:** Sincroniza o repositório local com as atualizações remotas (download + merge).

### **git push**

* **O que faz:** Envia commits locais para o repositório remoto.

**Uso:**

git push origin <branch>

**Descrição:** Transfere as alterações feitas localmente para o servidor remoto.

### **git merge**

* **O que faz:** Combina o histórico de duas branches.

**Uso:**

git merge <nome-da-branch>

**Descrição:** Integra as mudanças de uma branch na branch atual.