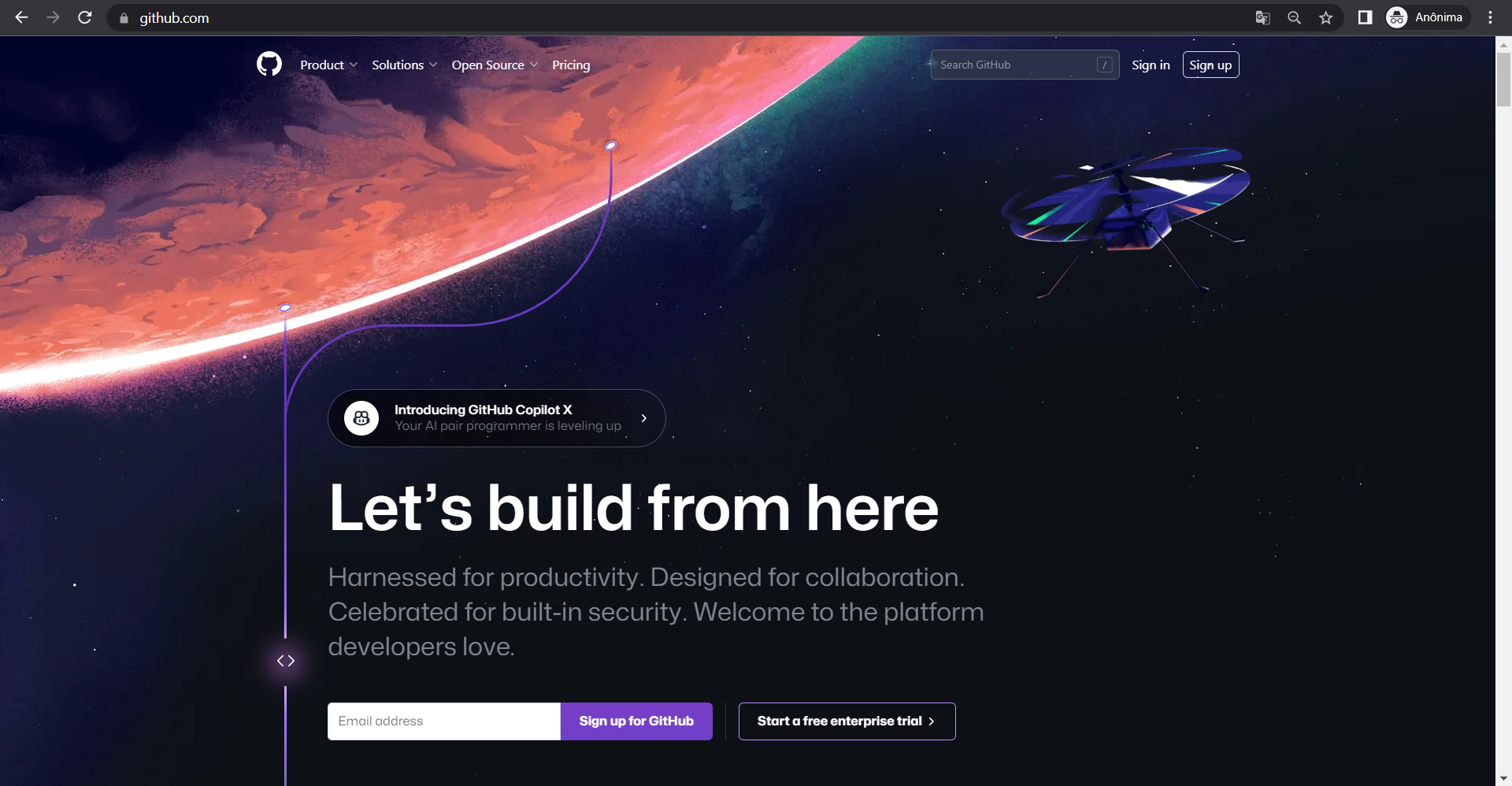
GITHUB

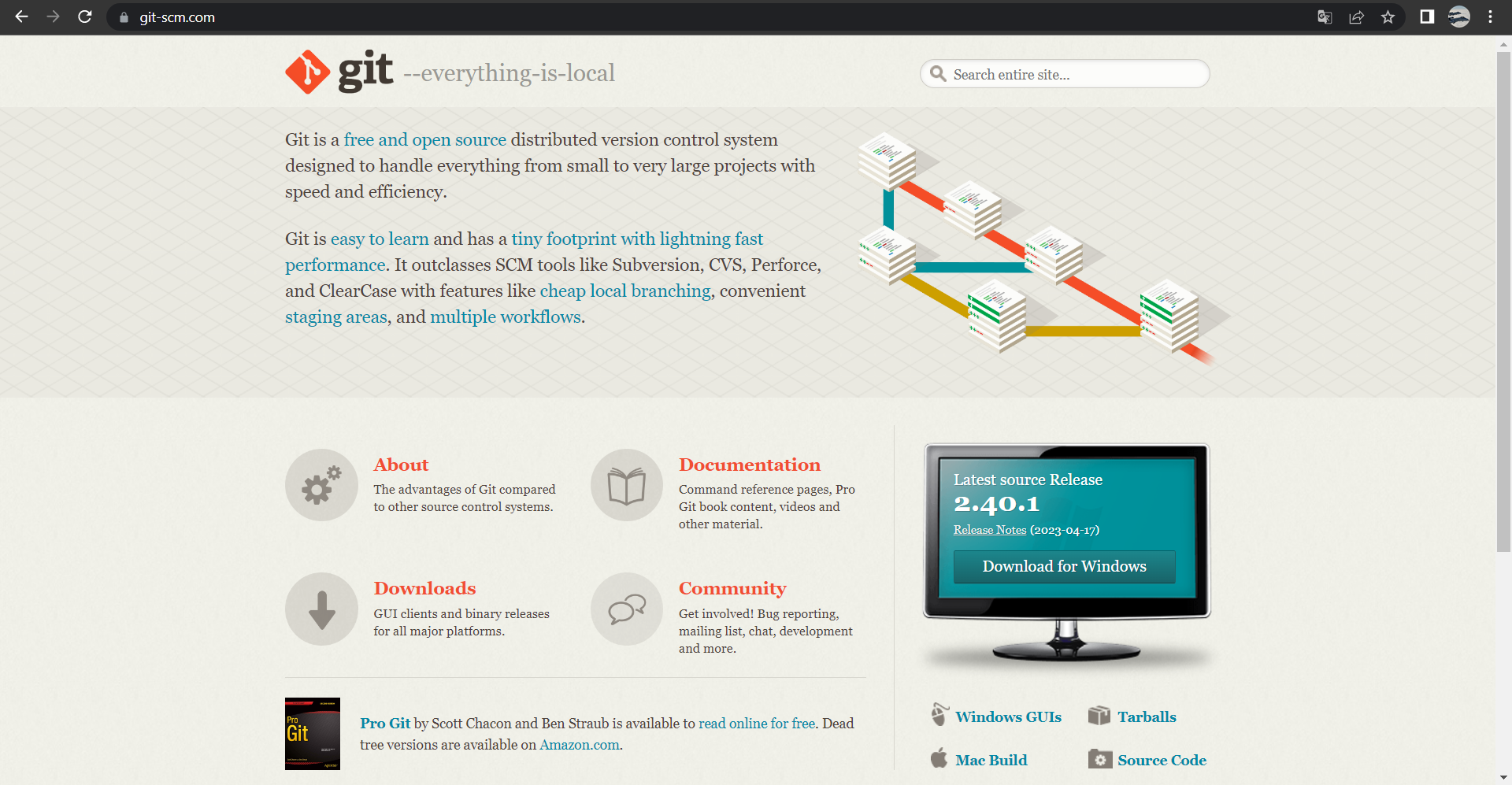
<https://github.com/>



*Passos*

* Clique em Sign Up.
  + Siga os passos para criação de uma conta.
  + Confirme seu e-mail.
* Com a sua conta online, vá em **repositórios**.
  + Para criar um novo, clique no botão **New**.
  + Configure o nome, privacidade e outras opções.
  + Clique em **Criar**.
* Com o seu repositório novo criado:
  + Entre nele e veja o **passo-a-passo** para a sincronização dele.
  + Baixe o programa **Git** (git-scm) e sincronize um repositório em uma pasta no seu sistema. (só seguir o passo-a-passo)

## BAIXANDO O GIT



*Instalação*

* Baixe o programa na sessão de downloads
* Execute o instalador e faça a instalação padrão
* Após terminar, verifique se o **Git Bash** está instalado

*Criando um repositório*

* No **Git Bash**, ande pelas pastas do sistema até o local desejado
* Crie um novo repositório no GitHub
* Siga o passo-a-passo para a criação e sincronização
  + **Comandos:**

git init

git add --all

É necessário um arquivo dentro da pasta

git branch -M main

git commit -m *“Mensagem sobre esse commit.”*

git remote add origin https://github.com/username/new\_repo

git push -u origin main

# COMO FUNCIONA O GIT & GITHUB

Existem várias funções e ferramentas na plataforma, mas as mais comuns serão descritas abaixo:

## REPOSITÓRIOS

Repositórios são locais onde arquivos são acompanhados e *versionados*, ou seja, é mantido uma documentação das alterações feitas em códigos, arquivos de imagem, som, entre outros.

É **necessário** um repositório para subir arquivos para o GitHub e ter uma documentação compartilhada.

*Criando um repositório:*

git init

git branch -M main

git add --all

* + É necessário um arquivo dentro da pasta
  + Use um arquivo README.md para mostrar informações no GitHub

*Determinando um usuário* e *e-mail*

git config --{ *local | global* } user.name *“seu-nome”*

git config --{ *local | global* } user.email *“seu-email”*

## MANDANDO PARA O REPOSITÓRIO REMOTO

Para enviar ao repositório remoto é necessário ter alterações *“commitadas”* e diferentes do branch que está no ambiente remoto.

*Enviando arquivos para o remoto*

git push { *origin “branch”* }

// Usar --set-upstream e o nome da branch (sem ser a main) para criar uma branch nova

*Sincronizando arquivos locais com o remoto*

git fetch { *origin “branch”* } // Apenas procura o repositório remoto

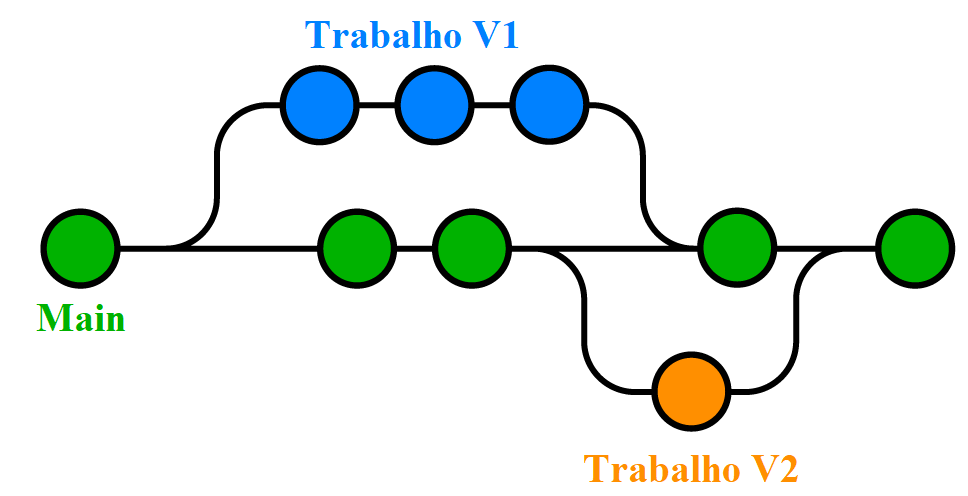
git pull { *origin “branch”* } // Puxa os arquivos remotos para a local

git clone { *“link-do-repositório-remoto”* }

**Nota:** *Local* é o usuário somente daquela pasta ou usuário do seu sistema. Já o *Global* é uma configuração para qualquer pasta e qualquer usuário daquele computador. O uso do *global* **não é indicado**.

**Nota 2:** Os comandos irão ser executados no local atual do seu terminal, ou seja, troque para a pasta desejada **antes de executar os comandos**.

## BRANCHES & MERGING



Uma branch é uma cópia do repositório original (Normalmente da branch **main** ou **master**), e podem ser usadas para proteger o código que está sendo colocado em produção, efetivamente é possível criar ambientes de teste antes de oficializar aquela alteração.

**Merging** é a ação de alterar a branch *main* com as novas alterações feitas na branch alternativa.

*Como criar uma branch*

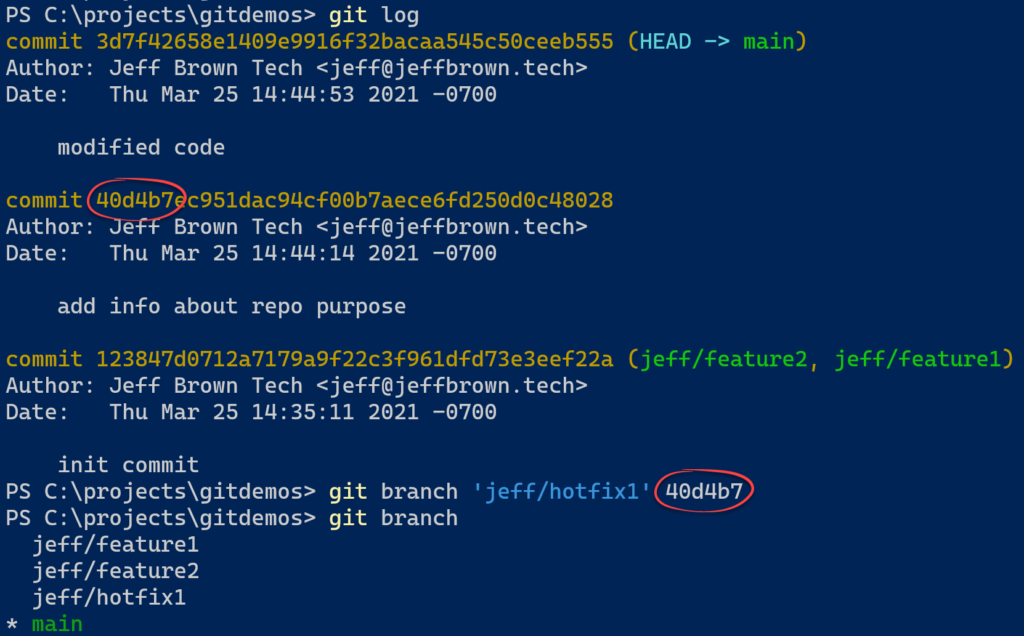
git branch *“Nome-da-branch”*

git checkout -b *“Nome-da-branch”*

* É possível criar uma branch referenciando o código SHA-1 de um commit, igual na imagem abaixo.

git log

git branch *“nome-da-branch”* {*código sha-1*}



*Como trocar de branch*

git checkout *“branch”*

O **merge** é feito *puxando os arquivos* de outra branch para a que você está, ou seja, dentro da *main* você irá puxar outra branch.

*Fazendo merge em repositórios locais*

git checkout *main*

git merge *“branch-para-unir”*

git commit -m *“branch merge”*

*Criando branches em repositórios remotos*

git push --set-upstream *origin “branch”*

Criar um **pull request** no site do GitHub

*Fazendo merge da main para uma branch alternativa*

git checkout *“branch-alternativa”*

git merge *main*