

INTRODUÇÃO



GIT/GITHUB

DIA 1

TÓPICOS ABORDADOS



git



- BREVE HISTÓRIA
- VISÃO GERAL
- CRIAÇÃO DE CONTA
- REPOSITÓRIO REMOTO
- BENEFÍCIOS/ESTUDANTE
- INSTALAÇÃO DO GIT
- CONFIGURAÇÃO INICIAL
- COMANDOS BÁSICOS
- EXERCÍCIOS DE FIXAÇÃO



BREVE HISTÓRIA

O que é Git/GitHub ?

- Git é um sistema de controle de versão distribuído amplamente utilizado por desenvolvedores para gerenciar e acompanhar alterações no código-fonte ao longo do tempo.
- Enquanto o GitHub é uma plataforma de hospedagem de código-fonte e controle de versão que usa Git. Com isso você pode armazenar, compartilhar e trabalhar com outros colaboradores escrevendo código.

BREVE HISTÓRIA

- **Git:**
 - Criado por Linus Torvalds
 - Criado para auxiliar no gerenciamento do desenvolvimento do kernel do Linux
 - Deu inicio ao desenvolvimento em abril de 2005
 - Primeira versão do Git é lançada em 2005
 - Em 2007, Junio Hamano se torna o mantenedor principal do Git

Obs. : Linus Torvalds iniciou o desenvolvimento do Git, com a ajuda de outros colaboradores, devido a uma necessidade urgente de substituir o sistema de controle de versão anterior (BitKeeper) que havia se tornado pago

BREVE HISTÓRIA

- **GitHub:**
 - Criado por Tom Preston-Werner, Chris Wanstrath, PJ Hyett e Scott Chacon
 - Criado com a necessidade de um sistema de controle de versão descentralizado e fácil de usar, baseado em Git
 - Em 2008, o GitHub é lançado
 - Ganhou rápida popularidade entre os desenvolvedores
 - Em 2018, o GitHub foi comprado pela Microsoft por 7,5 bilhões de dólares

Obs. : Antes do GitHub, os desenvolvedores costumavam compartilhar código através de arquivos zip ou por meio de sistemas de controle de versão centralizados.

VISÃO GERAL

O que é controle de versão?

- Controle de versão é uma metodologia que permite acompanhar e gerenciar alterações em arquivos ao longo do tempo. Com o controle de versão, você pode:
 - Comparar mudanças
 - Rastrear alterações
 - Manter a integridade dos dados
 - Documentar os desenvolvimentos
 - Trabalhar colaborativamente
 - Reverter para alterações anteriores
 - etc

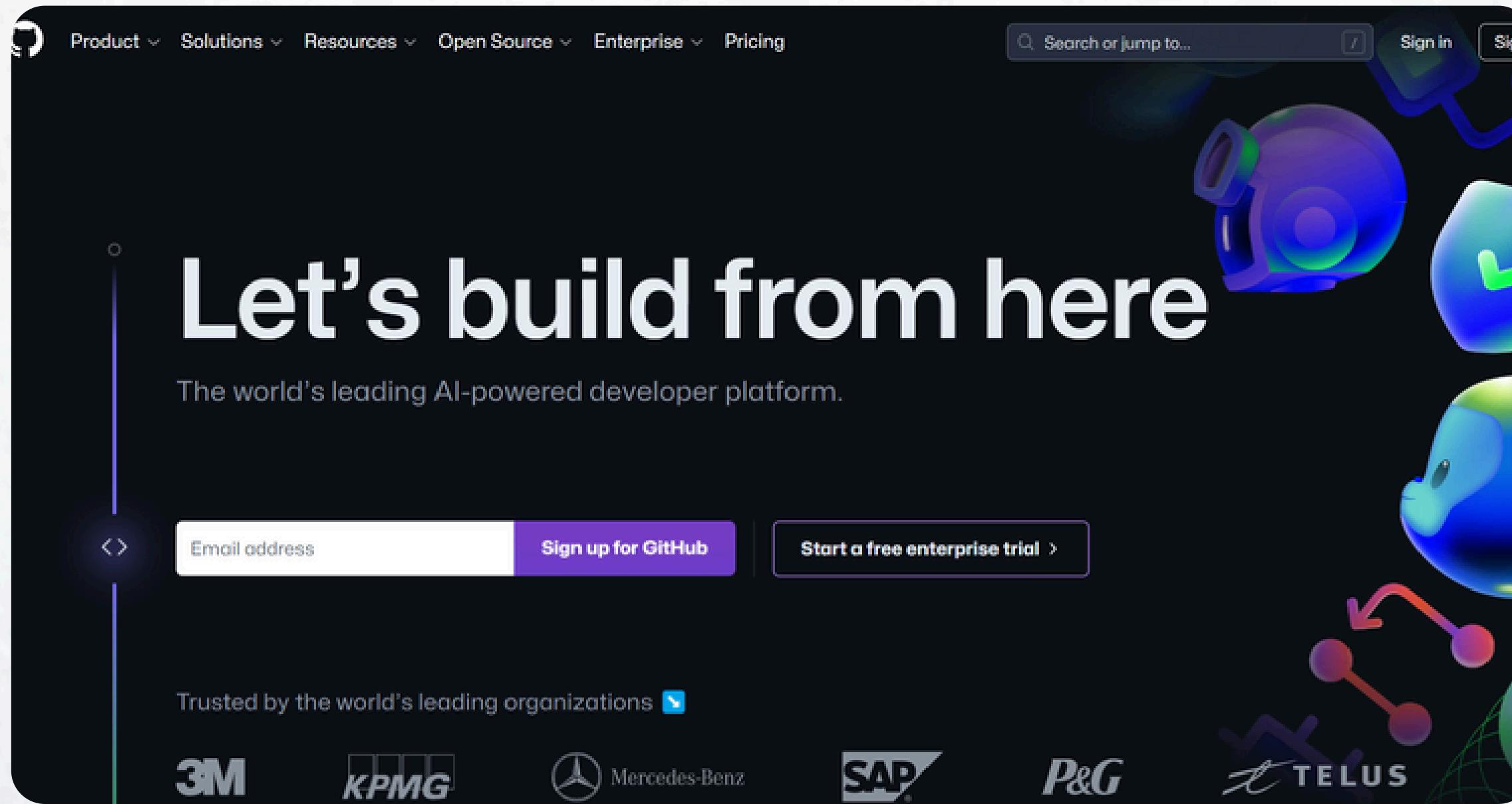
VISÃO GERAL

Por que usar o Git?

- Git é uma das ferramentas de controle de versão mais populares devido às suas características:
 - Melhoria na colaboração
 - Flexibilidade nos Fluxos de Trabalho
 - Comunidade e suporte
 - Usa SHA-1 para nomear e identificar objetos (arquivos, diretórios, commits), garantindo a integridade dos dados.
 - Cada desenvolvedor possui uma cópia completa do repositório, permitindo trabalho offline e sincronização quando necessário

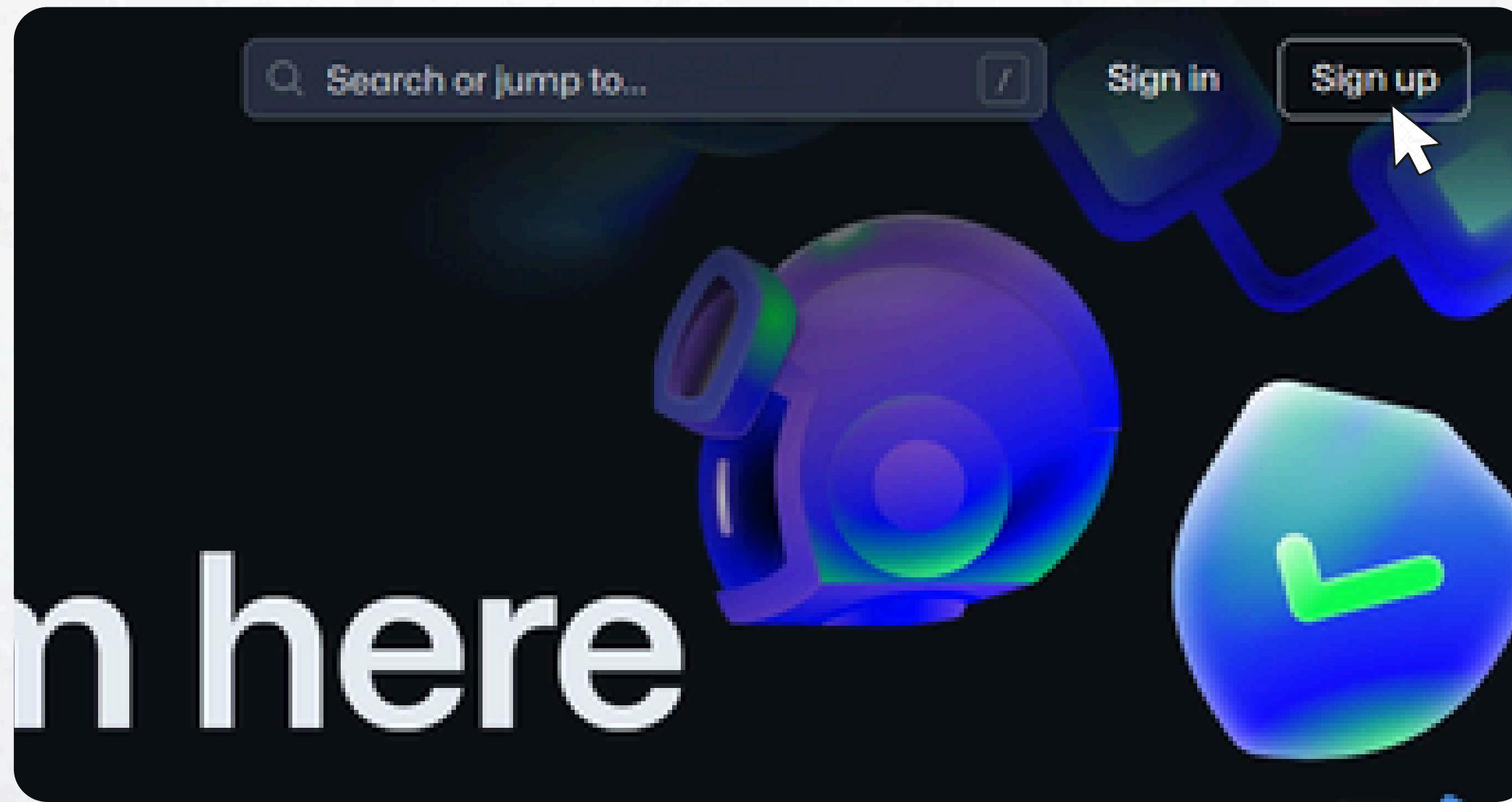
CRIAÇÃO DE CONTA

- Primeiramente vamos para o site oficial do GitHub.



CRIAÇÃO DE CONTA

- Sign in(entrar) e Sign Up(inscrever-se)



CRIAÇÃO DE CONTA

- Email acessível

Welcome to GitHub!

Let's begin the adventure

Enter your email*



Continue

CRIAÇÃO DE CONTA

- Senha

Create a password*



Continue

- Nome do usuário

Enter a username*



Continue

CRIAÇÃO DE CONTA

- Verificação de robo

Verify your account

Protegendo sua conta

Resolva este enigma para sabermos que você é uma pessoa de verdade

Verificar



Áudio

CRIAÇÃO DE CONTA

- Verificação de código

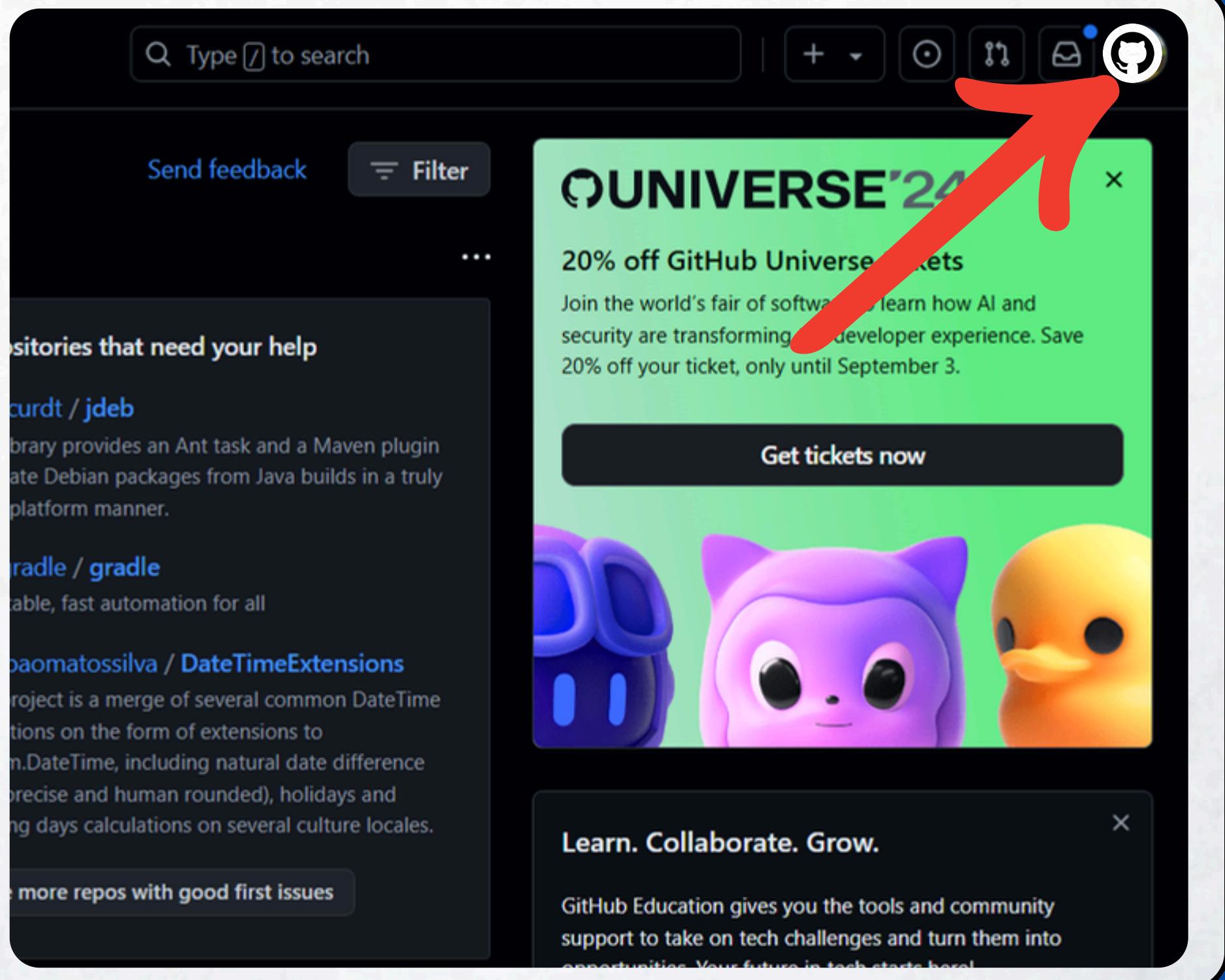
You're almost done!

We sent a launch code to teste@empresa.com

→ Enter code*

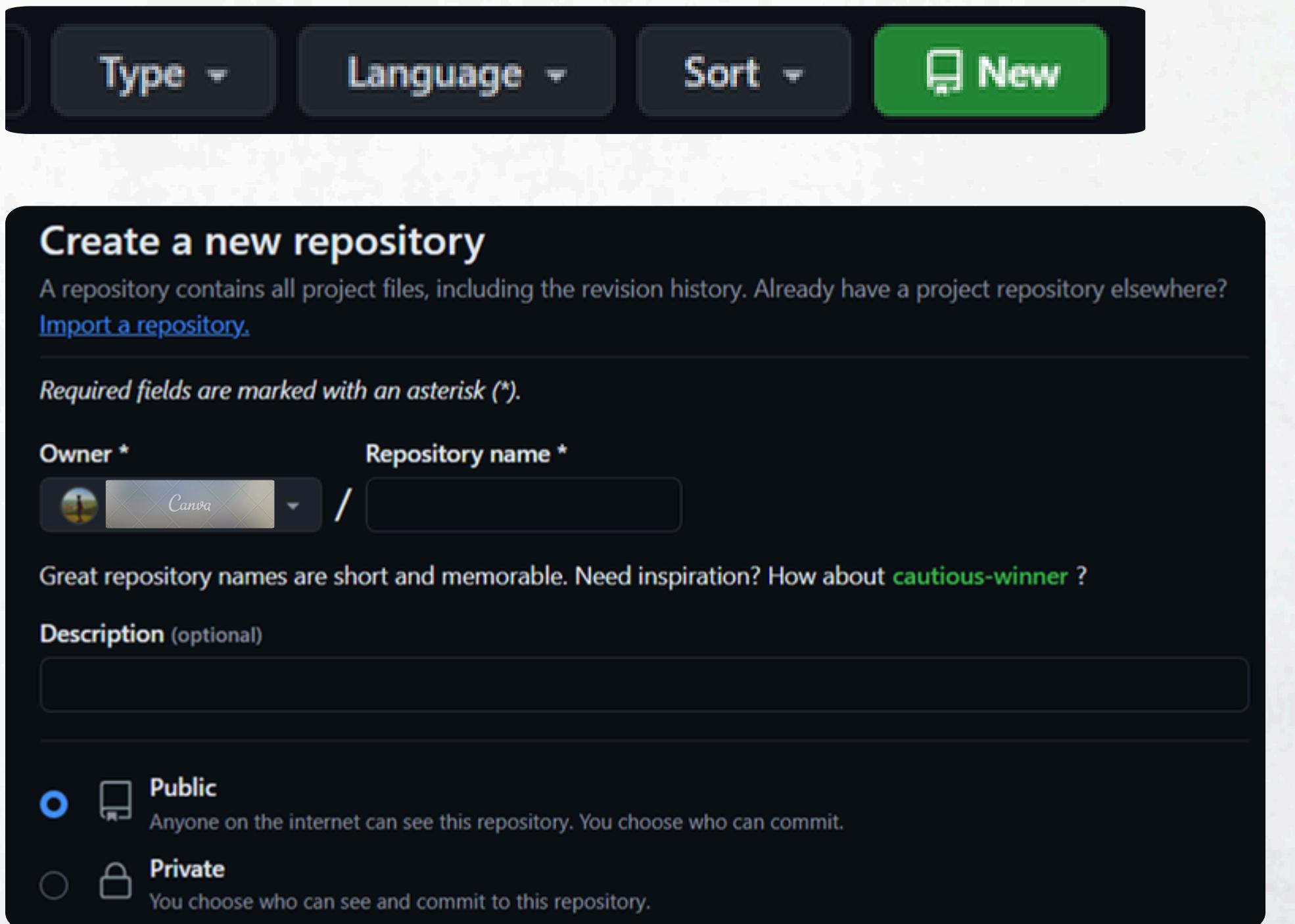
REPOSITÓRIO REMOTO

- Clique no ícone do seu perfil e logo em seguida em **Your repositories(Seus repositórios)**



REPOSITÓRIO REMOTO

- Clique em **New(Novo)**
- Logo em seguida aparecerá essa tela ao lado -->
- Coloque o nome do repositório como:
Git/GitHub-LinguagemC
- Deixe o repositório como público

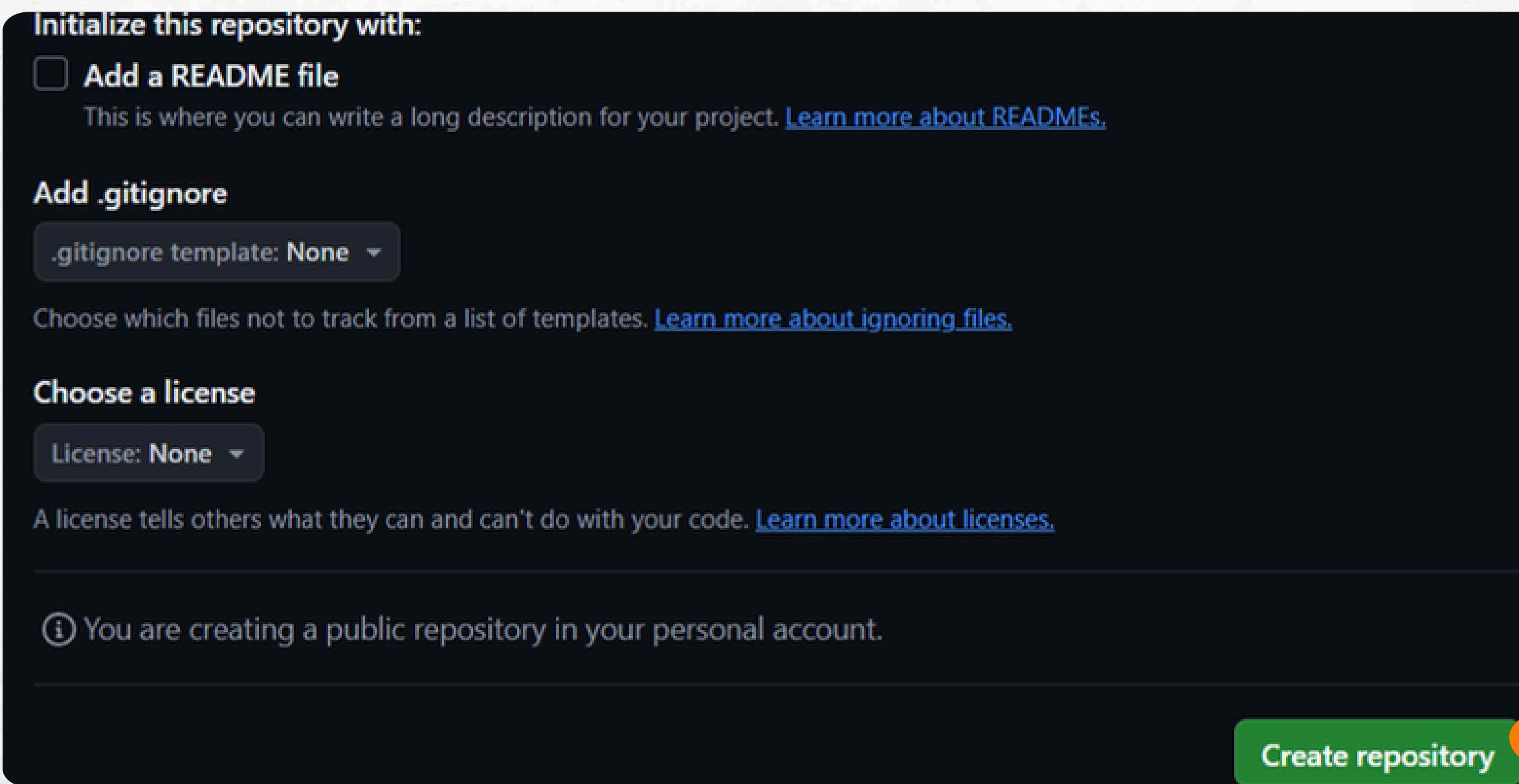


The screenshot shows the GitHub interface for creating a new repository. At the top, there are four buttons: 'Type' (dropdown), 'Language' (dropdown), 'Sort' (dropdown), and a green 'New' button with a smartphone icon. Below these, the main form has the following fields:

- Create a new repository**: A heading with a subtext: "A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)"
- Owner ***: A dropdown menu showing "Canva".
- Repository name ***: An input field containing the placeholder text "Cautious-winner".
- Description (optional)**: A large text area with a placeholder text: "Great repository names are short and memorable. Need inspiration? How about [cautious-winner](#) ?"
- Public**: A selected radio button option. Description: "Anyone on the internet can see this repository. You choose who can commit."
- Private**: An unselected radio button option. Description: "You choose who can see and commit to this repository."

REPOSITÓRIO REMOTO

- Clique em **Create repository (Criar repositório)**



BENEFÍCIOS PARA ESTUDANTES

- O GitHub oferece gratuitamente uma variedade de ferramentas e serviços enquanto você for estudante vinculado a uma instituição de ensino.
- O GitHub Student Developer Pack é um dos principais benefícios deste programa. Entre os benefícios oferecidos, destacam-se:
 - Repositórios privados ilimitados
 - GitHub Copilot
 - GitHub Pro
 - Acesso a plataformas de aprendizado
 - Créditos em serviços de nuvem
 - Ferramentas de desenvolvimento
 - etc

INSTALAÇÃO DO GIT

- Para **Windows**, você pode baixar o instalador do Git a partir do site oficial do Git . Ou caso prefira é possível instalá-lo pelo PowerShell utilizando o seguinte comando: winget install –id Git.Git –e –source winget
- No **MacOs**, o Git pode ser instalado utilizando um dos seguintes gerenciadores de pacotes:
 - Homebrew (brew install git)
 - MacPorts (sudo port install git)Também está disponível um instalador para **MacOs**
- A instalação no Linux pode ser feita utilizando o gerenciador de pacotes da sua distribuição (por exemplo, sudo apt-get install git no Ubuntu).

CONFIGURAÇÃO INICIAL

- 1 - Depois de instalar o Git, entre no **Terminal** e configure suas credenciais (**nome de usuário** e **email**):
 - **git config --global user.name "seu nome de usuário"**
 - **git config --global user.email "seuemail@exemplo.com"**
- 2 - Logo após verifique se você já tem uma chave SSH:
 - **ls -al ~/.ssh**

Se você ver arquivos como **id_rsa.pub** ou **id_ed25519.pub**, você já tem uma chave SSH.

CONFIGURAÇÃO INICIAL

- 3 – Se não tiver uma chave SSH, gere uma nova:
 - **ssh-keygen -t ed25519 -C "seu_email@exemplo.com"**

Se o seu sistema não suporta '**ed25519**', use '**rsa**':

- **ssh-keygen -t rsa -b 4096 -C "seu_email@exemplo.com"**

Pressione Enter para aceitar o local padrão do arquivo. Digite uma senha segura quando solicitado.

- 4 – inicie o ssh-agent em segundo plano:
 - Linux/MacOs: **eval "\$(ssh-agent -s)"**
 - Windows (GitBash): **eval \$(ssh-agent -s)**

CONFIGURAÇÃO INICIAL

- 5 - Adicione sua chave SSH ao ssh-agent:

- **ssh-add ~/.ssh/id_ed25519**

Ou, se você usou rsa:

- **ssh-add ~/.ssh/id_rsa**

CONFIGURAÇÃO INICIAL

- 6 – Copie a chave SSH para sua área de transferência:
 - **Linux:**
 - **xclip -selection clipboard < ~/.ssh/id_ed25519.pub**
 - Ou, se você usou rsa:
 - **xclip -selection clipboard < ~/.ssh/id_rsa.pub**

Se xclip não estiver instalado, use:

- **cat ~/.ssh/id_ed25519.pub**

Ou, se você usou rsa:

- **cat ~/.ssh/id_rsa.pub**

E copie manualmente a saída.

CONFIGURAÇÃO INICIAL

- MacOs:
 - **pbcopy < ~/.ssh/id_ed25519.pub**
- Ou, se você usou rsa:
 - **pbcopy < ~/.ssh/id_rsa.pub**

CONFIGURAÇÃO INICIAL

- Windows (PowerShell):

- **Get-Content ~/.ssh/id_ed25519.pub | Set-Clipboard**

Ou, se você usou rsa:

- **Get-Content ~/.ssh/id_rsa.pub | Set-Clipboard**

Ou no Command Prompt:

- **type %userprofile% \ .ssh \ id_ed25519.pub | clip**

Ou, se você usou rsa:

- **type %userprofile% \ .ssh \ id_rsa.pub | clip**

CONFIGURAÇÃO INICIAL

- 7 – Adicione a chave SSH à sua conta do GitHub:
 - Vá para GitHub.com e faça login.
 - Clique na sua foto de perfil e selecione "Settings".
 - No menu lateral esquerdo, clique em "SSH and GPG keys".
 - Clique em "New SSH key" ou "Add SSH key".
 - Cole sua chave no campo "Key".
 - Dê um título descritivo à sua chave no campo "Title".
 - Selecione a opção "Authentication Key" em Key Type.
 - Clique em "Add SSH key".

CONFIGURAÇÃO INICIAL

- 8 – Teste sua conexão:
 - **ssh -T git@github.com**

Você pode ver um aviso sobre a autenticidade do host. Digite 'yes' para continuar.

- Se tudo estiver configurado corretamente, você verá uma mensagem de boas-vindas do GitHub.
- **Nota:** Para Windows, é recomendável usar Git Bash para executar comandos Unix-like.

CONFIGURAÇÃO INICIAL

Alternativa ao SSH

- Personal Access Token (PAT)
- O Personal Access Token (PAT) clássico do GitHub é uma forma de autenticação que permite que usuários accessem a API do GitHub e realizem operações em suas contas sem precisar usar a senha diretamente

CONFIGURAÇÃO INICIAL

Configuração no GitHub

- Acesse as configurações da sua conta do GitHub.
- Acesse as configurações de desenvolvedor.
- Em seguida, acesse **Personal Access Token** no menu lateral e clique em **Tokens (classic)**.
- Clique em **Generate new token > Generate new token (classic)** no canto superior direito.
- Em "Note", escreva o propósito do token.
- Defina o tempo de vida do token em **Expiration**.
- Em **Select scopes**, marque a opção **repo**.
- Por fim, clique em **Create token** e copie o token gerado.

CONFIGURAÇÃO INICIAL

Configuração no Git

- Abra o terminal
- Use o seguinte comando para armazenar as credenciais: **git config --global credential.helper cache**
 - Isso armazenará suas credenciais na memória por um tempo.
- Na próxima vez que você realizar uma operação que exija autenticação (como `git push` ou `git clone`), insira seu nome de usuário do GitHub e, em vez de sua senha, insira o PAT.

CONFIGURAÇÃO INICIAL

Configuração no Git

- Outra forma é configurar diretamente o repositório. Se você deseja que o Git use o PAT apenas para um repositório específico, pode configurar as credenciais diretamente na URL do repositório:
 - **git clone https://<username>:<token>@github.com/<username>/<repository>.git**
- Substitua `<username>`, `<token>` (seu PAT) e `<repository>` pelos valores apropriados.

COMANDOS BÁSICOS

Terminal

- O Command Line (CL), também conhecido como linha de comando ou terminal, é uma interface textual que permite aos usuários interagir com o sistema operacional executando comandos. É uma ferramenta poderosa para realizar uma ampla gama de tarefas, desde navegação de arquivos até a administração do sistema.
- A seguir veremos alguns comandos essenciais do CL tanto no **Windows** como no **Unix/Linux**

COMANDOS BÁSICOS

Windows

- **dir** – Lista o conteúdo de um diretório.
- **cd** – Altera o diretório de trabalho atual.
- **cls** – Limpa a tela do terminal.
- **copy** – Copia arquivos de um lugar para outro.
- **move** – Move ou renomeia arquivos ou diretórios.
- **del** – Remove arquivos.
- **mkdir** – Cria um novo diretório.
- **rmdir** – Remove diretórios vazios.
- **type** – Mostra o conteúdo de um arquivo.
- **find** – Procura por uma string de texto em um arquivo ou arquivos.
- **ren** – Renomeia arquivos ou diretórios

COMANDOS BÁSICOS

Unix/Linux

- **ls** – Lista o conteúdo de um diretório.
- **cd** – Altera o diretório de trabalho atual.
- **pwd** – Mostra o diretório de trabalho atual.
- **cp** – Copia arquivos ou diretórios.
- **mv** – Move ou renomeia arquivos ou diretórios.
- **rm** – Remove arquivos ou diretórios.
- **mkdir** – Cria um novo diretório.
- **rmdir** – Remove diretórios vazios.
- **touch** – Cria um novo arquivo vazio ou atualiza o timestamp de um arquivo existente.
- **cat** – Mostra o conteúdo de um arquivo.
- **find** – Procura por arquivos em um diretório.

EXERCÍCIOS DE FIXAÇÃO

Exercício 1

- Utilizando o CL do seu computador crie uma pasta chamda "**Workshop**" no diretório **Documentos**, em seguida crie o arquivo **fatorial.py** dentro do diretório **Workshop**.

EXERCÍCIOS

Resolução

- 1. Abra o terminal do seu computador.
- 2. Utilize o comando dir no Windows e ls para Unix/Linux para listar os arquivos e pastas no diretório atual.
- 3. Com o comando cd Documentos entre na pasta Documentos.
- 4. Utilize o comando mkdir Workshop para criar a pasta Workshop dentro de Documentos.
- 5. Entre na pasta que acabou de criar.
- 6. Por fim crie o arquivo fatorial.py utilizando type nul > fatorial.py no Windows e touch fatorial.py no Linux.

COMANDOS BÁSICOS

Git

- **git init** : Inicializa um novo repositório Git.
- **git clone url do repositório** : Clona um repositório existente.
- **git status** : Mostra o status das alterações no repositório.
- **git add nome do arquivo** : Adiciona um arquivo ao índice (staging area).
- **git commit -m "mensagem"** : Confirma as alterações adicionadas com uma mensagem descritiva.
- **git push** : Envia os commits locais para um repositório remoto.
- **git pull** : Atualiza o repositório local com as alterações do repositório remoto.

COMANDOS BÁSICOS

Git

- **git branch** : Lista, cria ou deleta branches.
- **git checkout nome da outra branch** : Alterna entre branches.
- **git merge nome da outra branch** : Mescla uma branch com a branch atual.
- **git rm nome do arquivo ou diretório** : Remove um arquivo ou diretório.

Obs.: O comando **rm** é muito poderoso e perigoso se usado incorretamente, especialmente com as opções **-r** e **-f**. Uma vez que um arquivo ou diretório é removido com **rm**, ele não pode ser recuperado usando comandos normais (exceto com ferramentas de recuperação de dados).

EXERCÍCIOS DE FIXAÇÃO

Exercício 2

- Abra o VsCode e logo em seguida abra o pasta "**Workshop**" (Criada no exercício 1)
- Entre no arquivo **fatorial.py** e digite o seguinte código abaixo:

```
1  def factorial(n):
2
3      if n == 0:
4          return 1
5      else:
6          return n * factorial(n - 1)
7
8  if __name__ == "__main__":
9      # Testes básicos
10     print(factorial(5)) # Saída esperada: 120
11
12
```

EXERCÍCIOS DE FIXAÇÃO

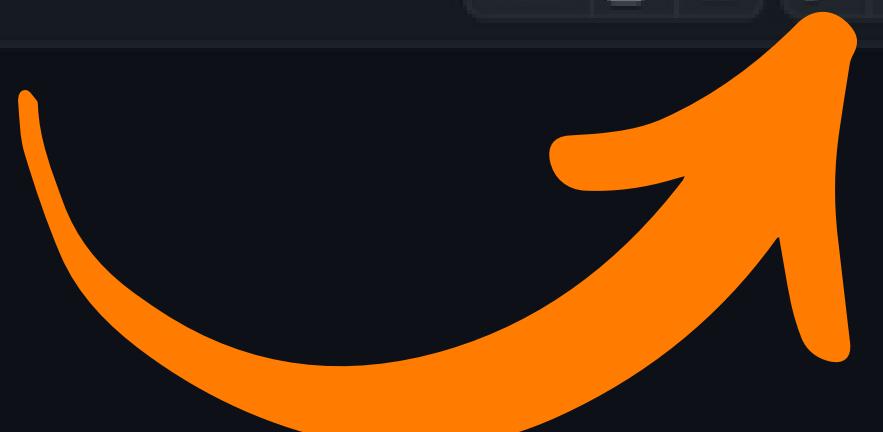
Exercício 2

- Logo após digitar o código, abra o terminal e digite os seguintes comandos:
 - **git init**
 - **git add nome do arquivo** ou **git add .**
 - **git commit -m "mensagem"**
 - **git remote add origin URL do repositório remoto**
 - **git push -u origin main**
- Verifique se no seu diretório remoto foi atualizado!

EXERCÍCIOS DE FIXAÇÃO

Exercício 3

- Vá até no seus repositórios, clique no repositório "**Git/GitHub-LinguagemC**". Dentro do repositório, clique no arquivo "**fatorial.py**", e logo em seguida no lápis.



A screenshot of a GitHub code viewer window. The window has a dark theme. At the top, there are tabs for "Code" (which is selected), "Blame", and "Raw". Below the tabs, it shows "34 lines (23 loc) · 732 Bytes". On the right side, there are icons for "Raw", copy, download, edit, and refresh. The main area displays a Python script with the following content:

```
1 ...
2
3     Analise:
4         Calcular o valor da Hipotenusa através dos valores dos catetos.
5     Definição:
6         Com os valores dos catetos, fazemos os cálculos para achar a hipotenusa.
7
8 ...
9
10    def main():
11        ...
```

EXERCÍCIOS DE FIXAÇÃO

Exercício 3

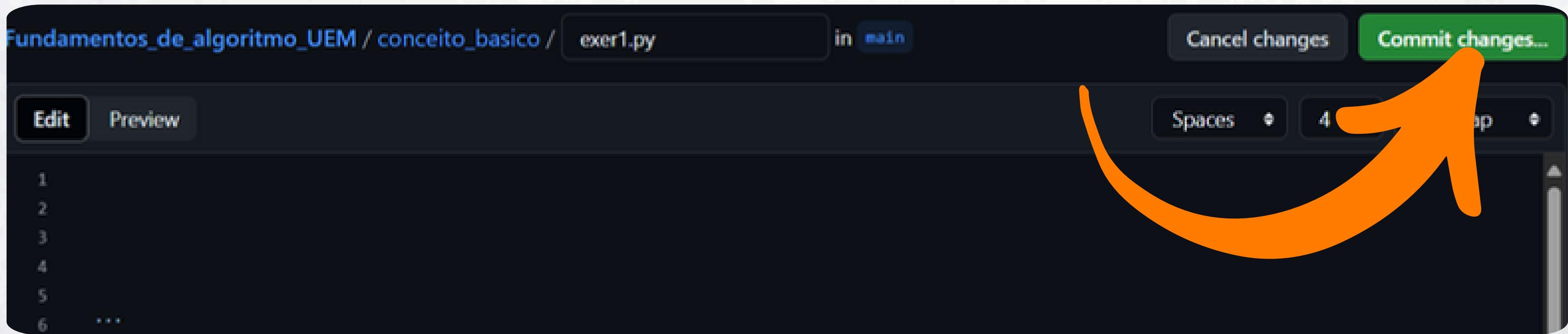
- Agora faça as seguintes alterações no repositório remoto: acrescente a especificação da função de acordo com a imagem abaixo.

```
1  def factorial(n):
2      """
3          Retorna o fatorial de um número n.
4
5          Parâmetros:
6          n (int): Um número inteiro não negativo.
7
8          Retorna:
9          int: O fatorial de n.
10         """
11     if n == 0:
12         return 1
13     else:
14         return n * factorial(n - 1)
15
16 if __name__ == "__main__":
17     # Testes básicos
18     print(factorial(5)) # Saída esperada: 120
```

EXERCÍCIOS DE FIXAÇÃO

Exercício 3

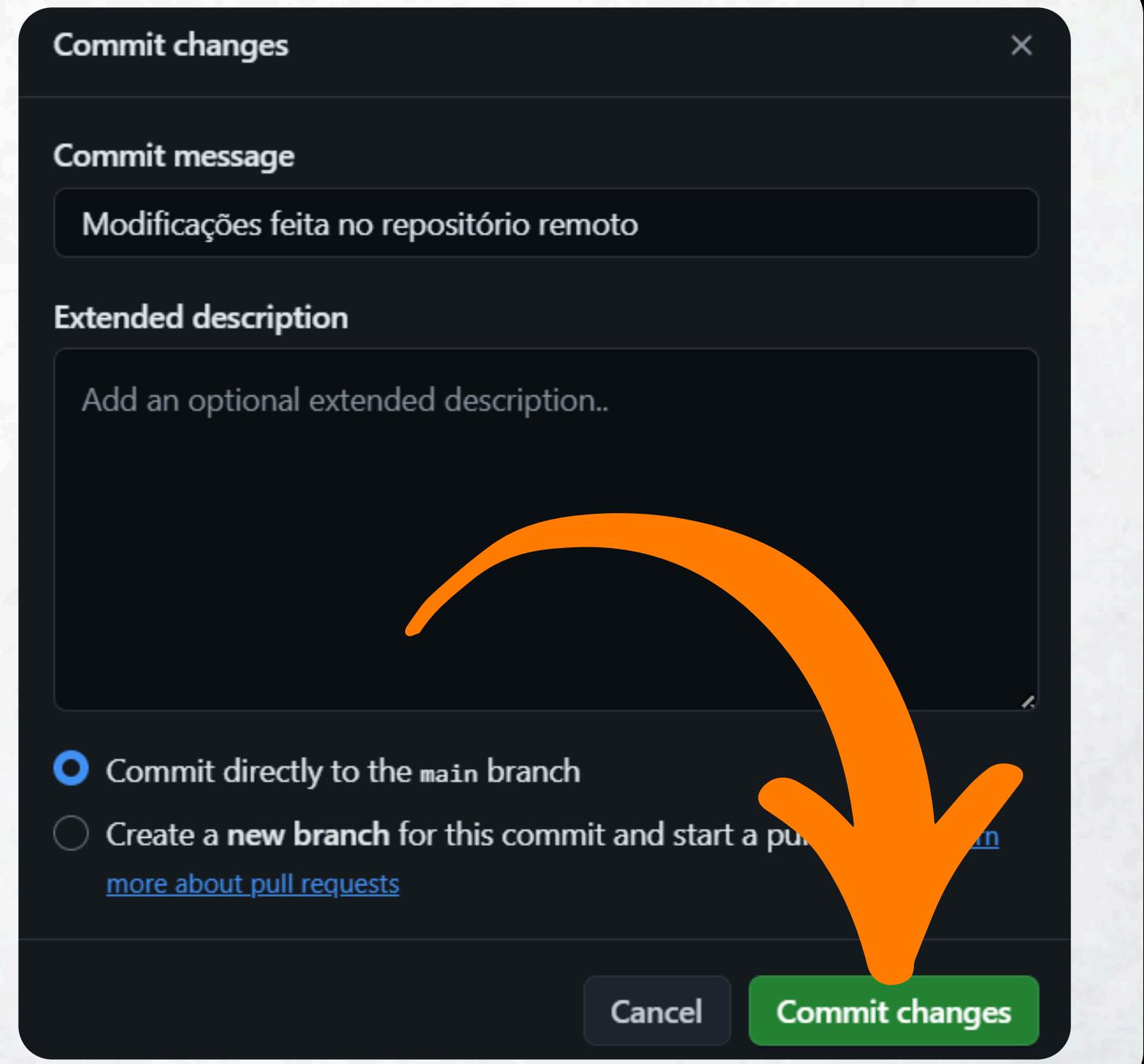
- Após ter feito as alterações no diretório remoto, clique em **Commit changes...**



EXERCÍCIOS DE FIXAÇÃO

Exercício 3

- Logo em seguida aparecerá essa tela.
- Abaixo de **Commit message** escreva a seguinte mensagem:
“Modificações feita no repositório remoto”
- Clique em **Commit changes**
- Após isso, vá para o VsCode, abra o terminal e digite **git pull**, e veja a mágica acontecer!

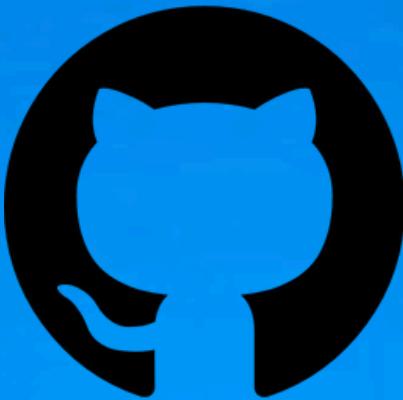


EXERCÍCIOS DE FIXAÇÃO

Exercício 4

- Utilizando o CL do seu computador crie outra pasta chamada "**CloneWorkshop**" no diretório **Documentos**, em seguida pelo CL do computador vá até aonde a pasta "**CloneWorkshop**" está localizada e digite **git clone URL do repositório**.
- Vá até no seu repositório remoto ("**Git/GitHub-LinguagemC**") e copia a **URL do repositório**
- Terminado essa etapa, abra outro VsCode e abra a pasta **CloneWorkshop**. Dentro da pasta terá que ter o **fatorial.py**.

CONGRATULATIONS



MUITO OBRIGADO!

