# **Introdução ao Git e ao GitHub**

* Recomendação de editor: Typora

**Para gerar a chave SSH:**

ssh-keygen -t ed25519 -C [bcesar2002@hotmail.com](mailto:bcesar2002@hotmail.com)

Em seguida:

cat id\_ed25519.pub (copiar o conteúdo exibido e colocar no GitHUb, em “<https://github.com/settings/keys>”)

Depois:

eval $(ssh-agent -s) (retorno: “Agent pid <número do processo>)

ssh-add id\_ed25519

Para clonar o repositório:

git clone <link SSH do repositório copiado do GitHub>

**Para gerar a chave Personal access tokens:**

No GitHub, ir para “<https://github.com/settings/tokens>” e clicar em “Generate new token”

* preencher o campo “Note”
* configurar o prazo de expiração
* marcar o escopo: “repo”
* clicar em “Generate token”
* copiar o token gerado e salvar em um local seguro.

**Para iniciar um repositório Git:**

git init (na pasta desejada)

ls -a (exibe arquivos ocultos)

**Para setar algumas variáveis do Git:**

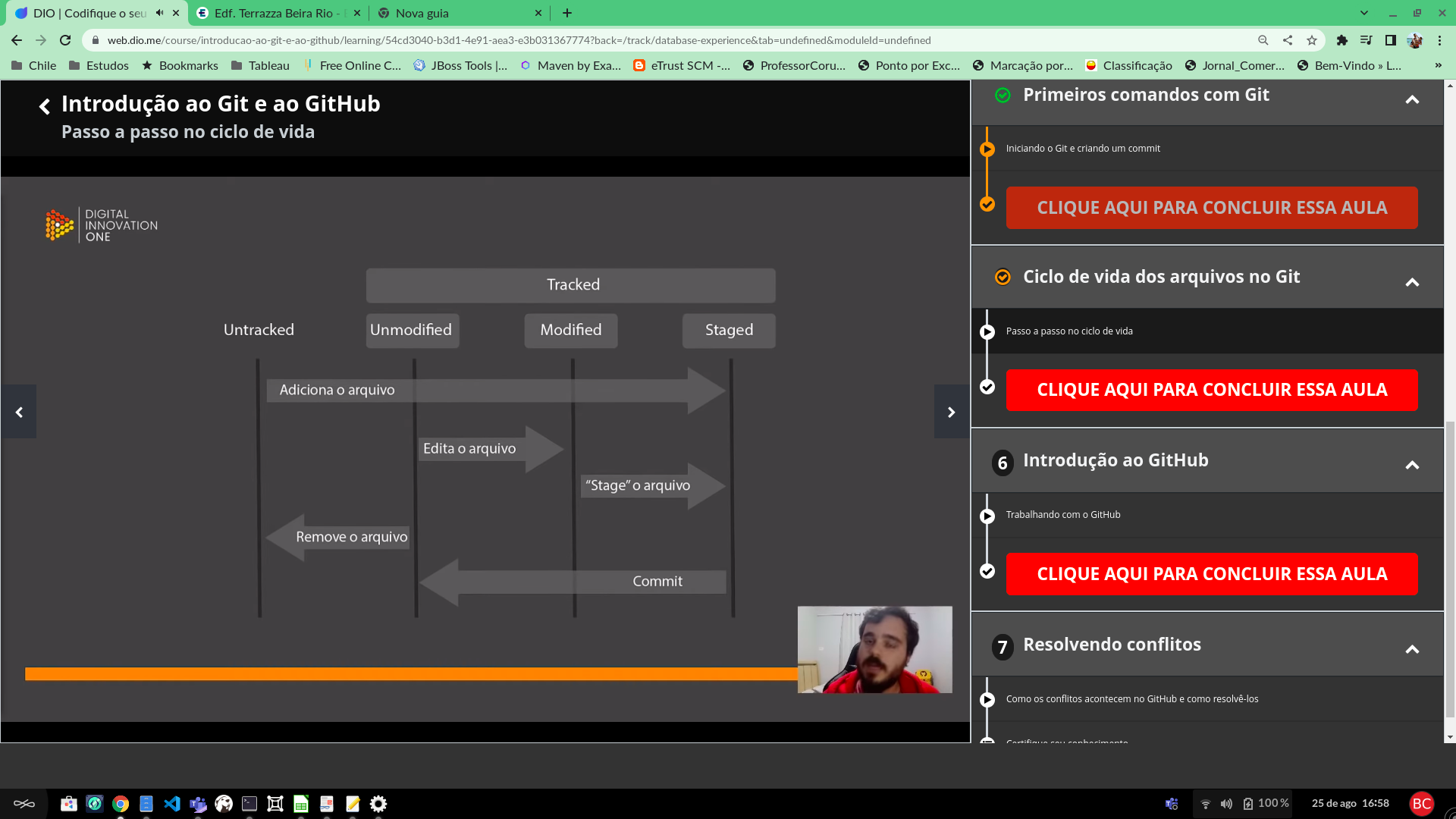
git config –global user.email “[bcesar2002@hotmal.com](mailto:bcesar2002@hotmal.com)

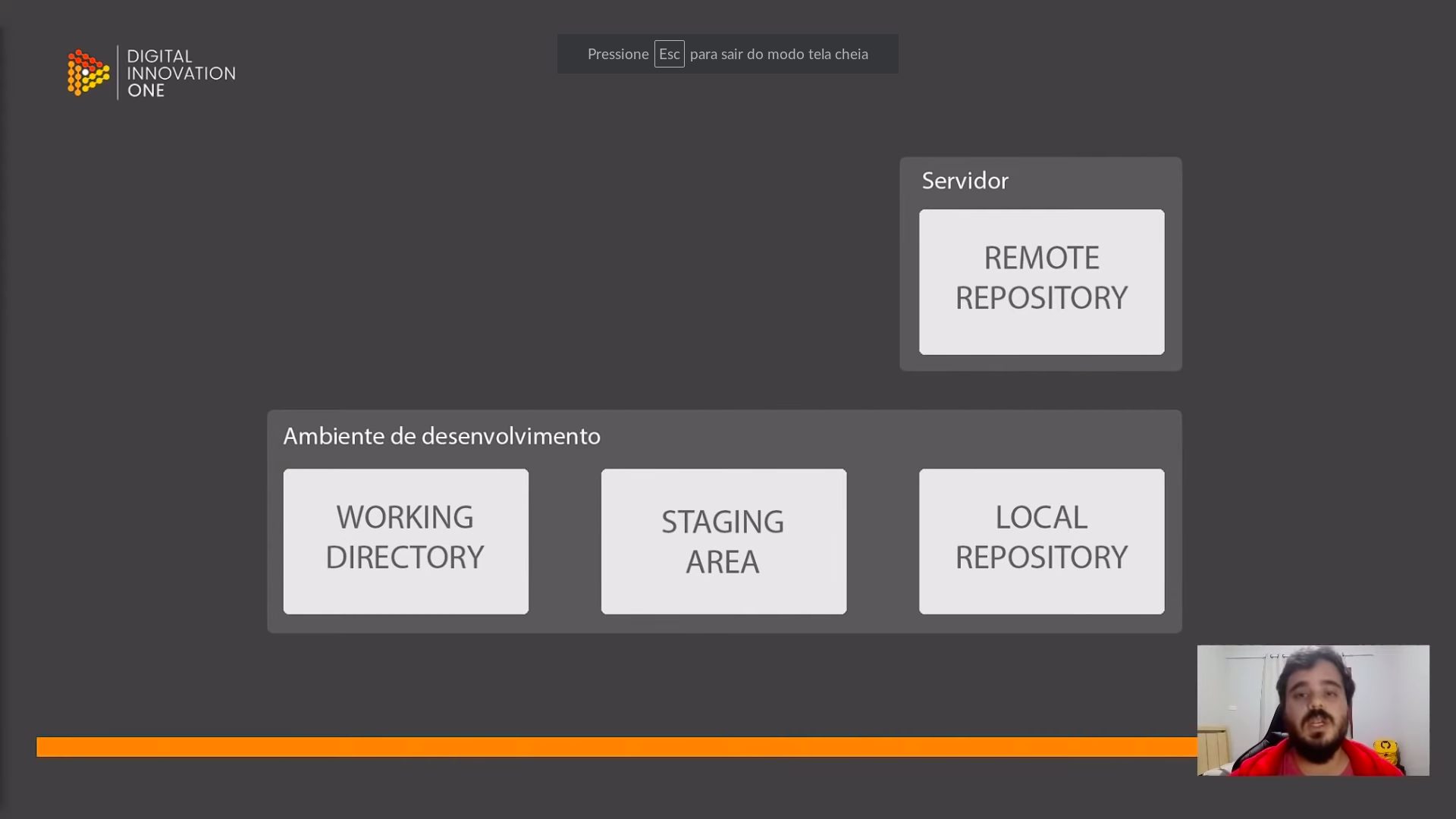
git config –global user.name “bcesar2002”

**Para adicionar um arquivo ao repositório Git:**

git add \* (ou .)

git commit -m “commit inicial”





Para mover um arquivo entre pastas: mv <arquivo> <destino>

Ciclo do Git:

WORKING DIRECTORY – (git add .) → STAGING AREA – git commit -m → REPOSITORY LOCAL

Para listar as configurações do seu Git: git config –list

Para modificar configurações do Git: git config –global –unset <nome da variável> (ex.: user.mail)

Envia para o repositório remoto (GitHub, por exemplo):

git remote add origin <link SSL/HTTPS do GitHub> (ex.: git@github.com:bcesar2002/livro-receitas.git

git remote -v (lista repositórios remotos)

git push origin master/main

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Colocando os códigos e práticas com Git e GitHub

- Dentro do git.sh estou colocando os códigos e algumas explicações.

## Entendendo o funcionamento do Git

### Sha1

Secure Hash Algorith (Algorítmo de hash seguro) - Algorítmo que faz o encriptamento de um arquivo, a encriptação gera um conjunto de 40 caracteres onde cada conjunto é único, e serve como identificação. Representa um arquivo de forma única.

Comando que gera um sha1 para o arquivo:



### Objetos Internos do Git (Responsáveis pelo versionamento)

Objetos são constituídos por metadados.

#### Blobs (bolhas)

Blobs são a composição básica dos objetos.

Metadados:

- Tipo: blob

- Tamanho do arquivo

- \0

- Conteúdo do blob

- sha1 próprio

#### Trees (Árvores)

São objetos que armazenam blobs e outras trees.

Metadados:

- Tipo: tree

- \0

- Aponta para blobs e trees que possuem o seu sha1.

- Guarda os nomes dos arquivos.

- sha1 próprio

#### Commit

Junta os blobs e trees e criam um projeto. E permite a vizualização da linha do tempo de alteração.

Metadados:

- Tipo: commit

- Aponta para uma tree que aponta para outras trees e blobs

- Aponta para o commit anterior a ele

- Aponta para um Autor

- Aponta para uma Mensagem (Que deve descrever as alterações realizadas)

- Possui um timestamp (Carimbo de tempo)

- sha1 próprio

## Status dos arquivos no Git

### Untracked

Arquivo que o git ainda não tem ciencia.

### Tracked

Arquivos que o git tem ciencia da sua existência. Possuem três estados.

#### Unmodified

Arquivos que não sofreram modificação.

#### Modified

Arquivos que sofreram alguma modificação.

#### Staged

Estado que os arquivo Modified e Untracked ficam, esperando ser comitados.

## Ciclo de vida dos arquivos no git

--- Untracked --- Unmodifie --- Modified --- Staged ---

O------------ Adiciona Arquivo ------------->

O-Editar Arquivo->

O---"Staged"--->

<-Remover Arquivo-O

<----------Commit---------------O

1. Arquivos que estão como Untracked são adicionados ao status de Staged, quando se utiliza o comando git add.

2. Arquivos Unmodified quando sofrem qualquer alteração ficam com o Status de Modified.

3. Arquivos Modified são adicionados ao status Staged, quando utiliza-se o comando git add.

4. Arquivos Unmodified quando removidos, tornam-se Untracked.

5. Arquivos na camada de Staged, aguardam ser comitados, com o comando git commit eles são comitados e se tornam Unmodified.

## Ambiente de Desenvolvimento

Ambiente de desenvolvimento é formado por toda a parte que fica na máquina.

--- Diretório de trabalho --- Staging Área --- Repositório Local (Git) ---

O--------git add-------->

O---------git commit------>

--- Untracked/Modified ------> Staged ----------> Unmodified --------------

1. Arquivos criados e/ou modificados no Diretório de trabalho vão para a área de Staging.

2. Os arquivos na Staging Área possuem as mudificações que "esperam" para ser commitadas.

3. No repositório local ficam apenas os commits.

--- Repositório Local (Git) ---- git push -----> Repositório Remoto (GitHub) ---

## Chaves ssh

### SSH

Conexão segura entre duas máquinas com uma chave pública e chave privada.

Para gerar as chaves no terminal entre o acesso remoto e o local:

- ssh-keygen -t ed25519 -C "e-mail"

- eval $(ssh-agent -s)

- ssh-add arquivo-do-id

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#!/bin/bash

# Comando para o repositório local

openssl sha1 nome\_do\_arquivo # Guarda o versionamento e cria um código sha1

git init # Inicia o repositório dentro de um diretório local

git add . # Adiciona todos os arquivos untracked ou modified para a área de staged

git add \* # Adiciona todos os arquivos untracked ou modified para a área de staged

git add nome\_do\_arquivo # Adiciona um arquivo para a área de staged

git commit -m 'Mensagem do Commit' # Cria o commit com todos os arquivos na área de Staged deixando eles com o status Unmodified

# Um commit deixa as alterações rastreáveis.

# O -m pede um comentário sobre as mudanças do commit.

git merge # Precisa juntar alterações de uma determinada branch com a master.

git status # Mostra as informações de status do repositório local, Untracked, Unmodified, Modified, Staged

git log # Mostra as alterações que o projeto vem sofrendo

# Mostra o hash dos commits, mostra a branch, o autor, data e a mensagem do commit.

git branch nome\_do\_branch # Cria um novo ramo, o git init cria um branch master, o git branch cria ramificações.

# Os branchs servem para partes diferentes dentro do projeto.

# Visualiza todas as branchs

git config --global user.email 'e-mail' # Atualiza o e-mail do usuário no GIT

git config --global user.name username\_do\_repositorio\_remoto # Atualiza o username do usuário no GIT

git config --list # Lista todas as configurações do Git

git config --global --unset user.email # Apaga configuraçãos de e-mail Git

git config --global --unset user.name # Apaga configurações de user do Git

# Comandos para gerar as chaves SSH - passo a passo

ssh-keygen -t ed25519 -C 'e-mail'

eval $(ssh-agent -s) # Cria um agente

ssh-add arquivo\_do\_id # Associa a chave privada

# Comandos do acesso remoto

git remote add origin link\_do\_repositório\_remoto # Cria uma conexão entre o repositório local e o remoto

git remote - v # Mostra as conexões remotas que o repositório local tem

git remote --verbose # Mostra as conexões remotas que o repositório local tem

# Documentação: https://git-scm.com/docs

git push # Publica as alterações, os commits, do repositório local para o repos>

git push origin master # Publicação feita pela branch master

git pull # Atualiza o repositório local conforme o repositório remoto.

git pull oringin master # Atualizar o repositório a partir do branch master

git clone link.git # Clona um projeto do repositório remoto