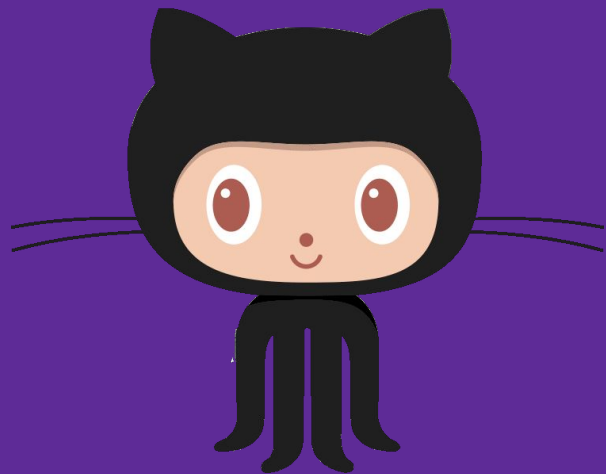
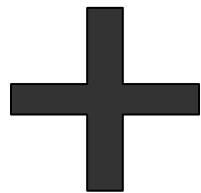


An Introduction to GitHub

Dalton Breno Costa





GEMINI

Global Emergency Medicine
Innovation & Implementation
Research Lab

NOT SO LONG AGO.
IN A GALAXY CLOSE BY...

HEY GEORGE
WHAT'S UP?

I ACCIDENTILY
DELETED ANOTHER
PAGE OF MY
MANUSCRIPT...

NOT THIS STUPID
'SUN BATTLE' THING
AGAIN...

IT'S NOT STUPID!
- YOUR STUPID!

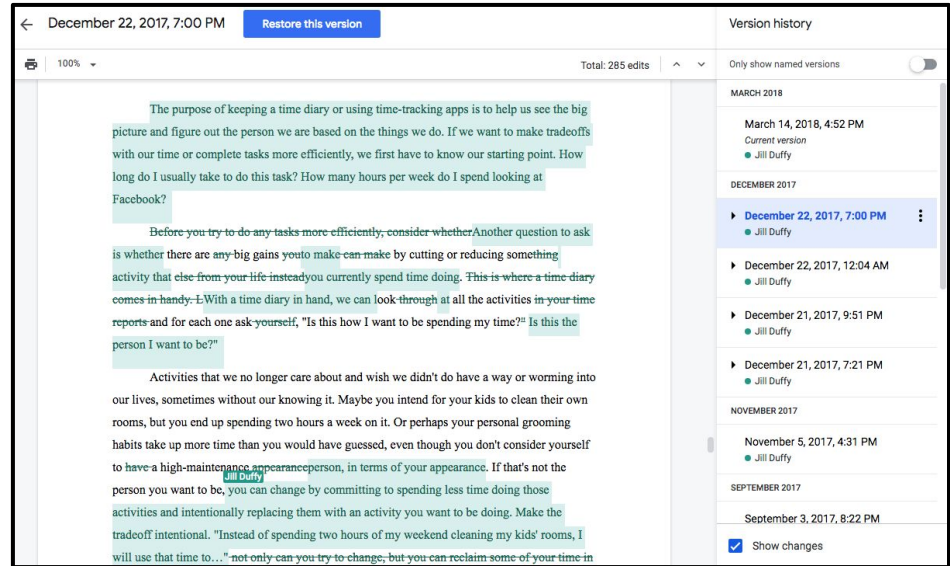
OH WELL...
YOU HAD IT ALL
UNDER VERSION
CONTROL RIGHT?

VERSION CON-WHAT?

UGH...

Version Control System

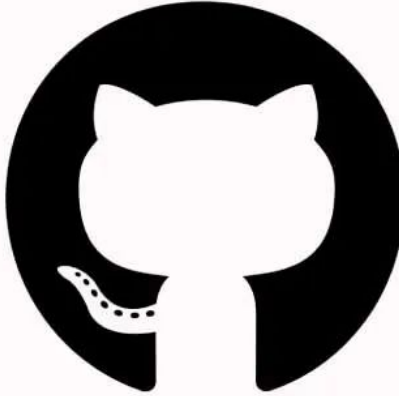
Version Control System allows developers to work on the same project by tracking and logging changes without any conflict with other iteration. When a developer change or modify a piece of the code, a version control system give other developers power to review and even restore the earlier version of the document.



Git



GitHub



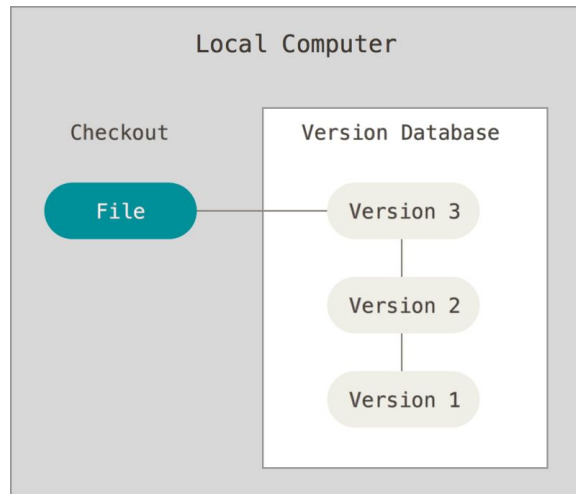
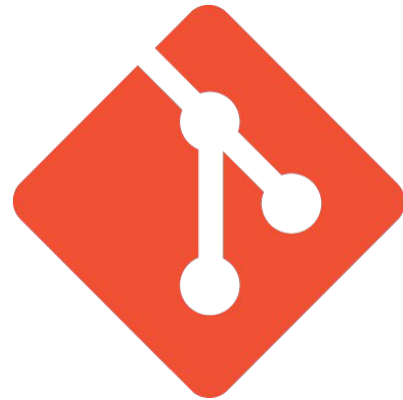
Gitlab



Git

- Git is free, fast and the most popular distributed version control system for storing and tracking changes in the source codes of projects;
- Allows you to create Git Repositories;
- Works locally.

<https://git-scm.com/>

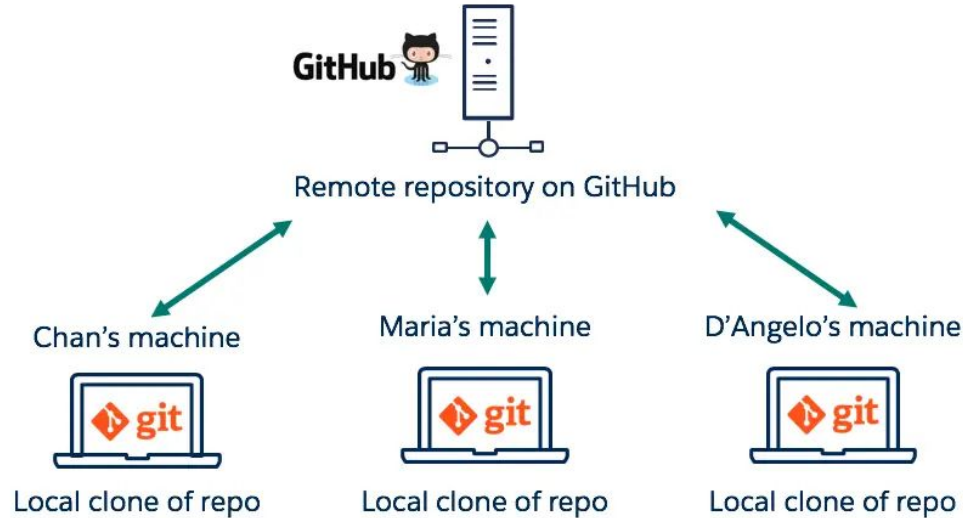


GitHub



GitHub is a cloud-based hosting service that allows developers to manage Git repositories.




























<https://github.com/>



GitLab

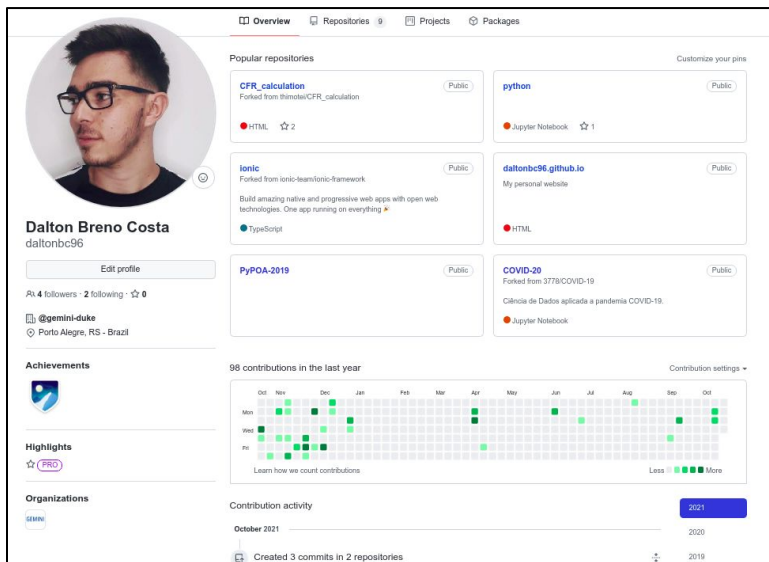


- GitLab is a repository manager that provides featured services like custom workflows, DevOps score, value stream mapping, monitoring, and so on;
- GitLab offers some similar features like GitHub, but GitLab is more than Github in terms of collaborative environment and functionalities.

 Manage	 Plan	 Create	 Verify	 Package	 Secure	 Release	 Configure	 Monitor	 Defend
									
									

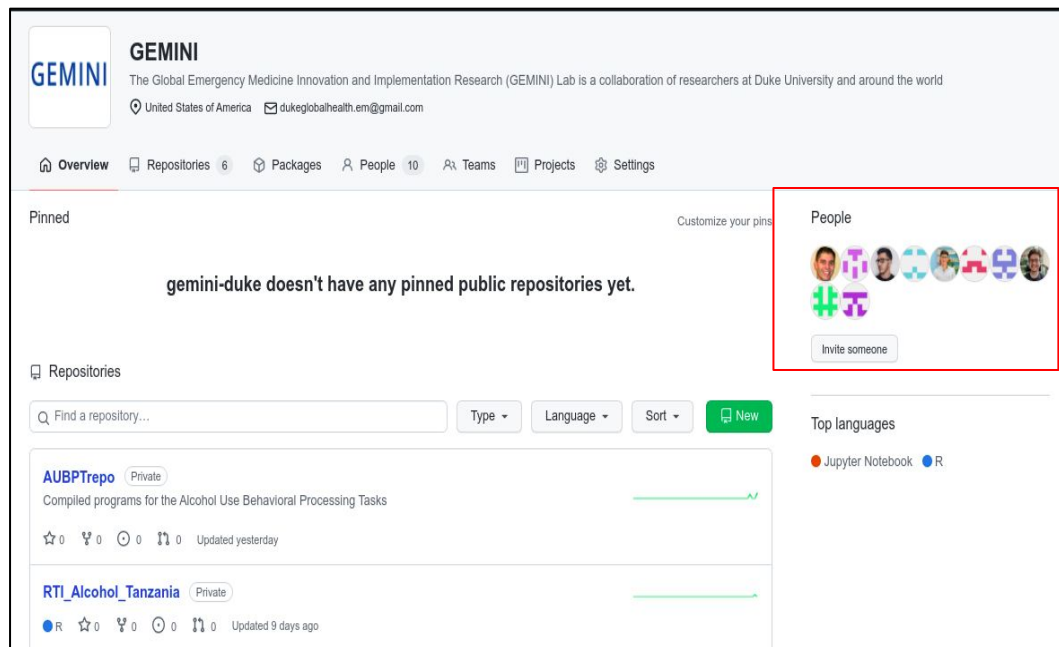
Types of Accounts

Personal user



The screenshot shows a personal GitHub profile for Dalton Breno Costa (daltonbc96). The profile includes a bio, a location (Porto Alegre, RS - Brazil), and a list of achievements. The 'Popular repositories' section displays several public repositories, including 'CFR_calculation', 'python', 'ionic', 'daltonbc96.github.io', 'PyPOA-2019', and 'COVID-20'. A contribution graph is visible, showing activity over the last year. The 'Organizations' section shows the user is part of the 'GEMINI' organization.

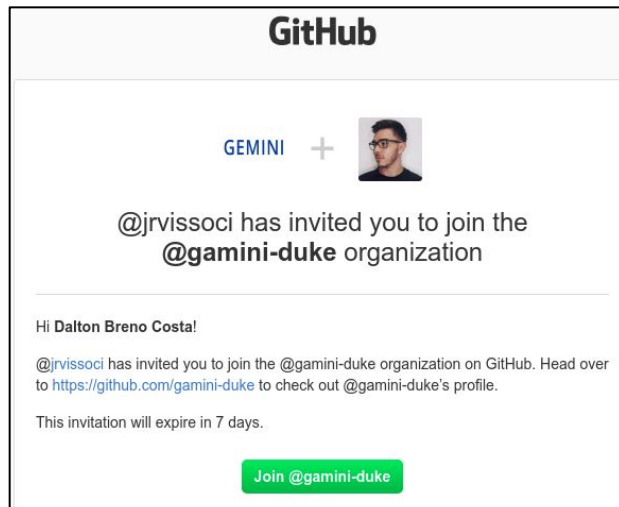
Organization



The screenshot shows the GitHub profile for the 'GEMINI' organization. The profile includes a bio, a location (United States of America), and a list of repositories. The 'Pinned' section displays a message: 'gemini-duke doesn't have any pinned public repositories yet.' The 'People' section shows a list of team members, with a red box highlighting the 'People' section. The 'Repositories' section shows a list of repositories, including 'AUBPTrepo' and 'RTI_Alcohol_Tanzania'.

How to access Gemini account on GitHub?

- You can access it using the address: <https://github.com/gemini-duke>
- If you want to add or read private files, you need to be added to the organization's team on Github;
- In order to be added to the team you need to provide your username and you will receive an invitation in your email.



How to install GitHub on my computer?

1. Create a personal user account on GitHub: <https://github.com/>
2. Install Git on your computer: <https://git-scm.com/downloads>
 - a. Tutorial teaching how to install Git on your computer: <https://phoenixnap.com/kb/how-to-install-git-windows>
- 3 - On Git terminal (Git Bash) set global user:

```
$ git config --global user.name "daltonbc96"  
$ git config --global user.email dalton.bc96@gmail.com
```

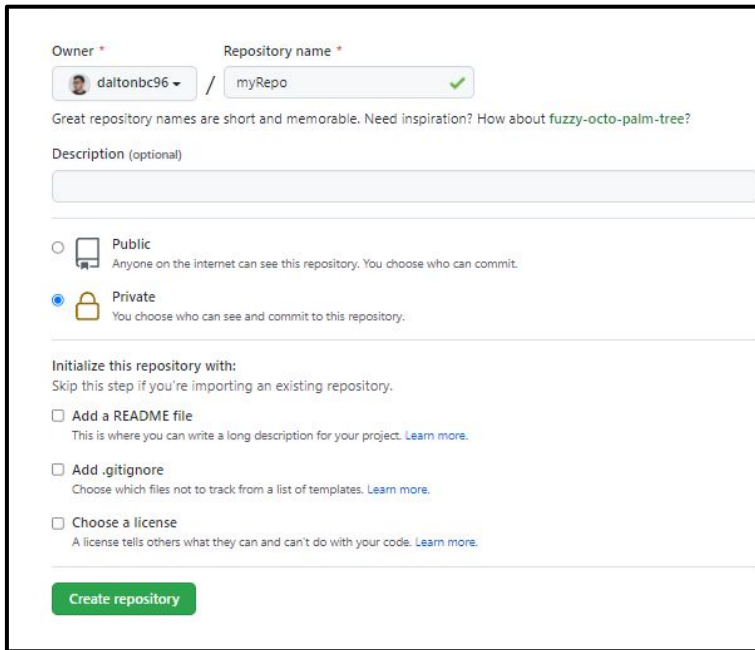
Summary of main Git commands:

<https://training.github.com/downloads/github-git-cheat-sheet.pdf>

Creating a repository


Step by step how to create a new repository on GitHub:

<https://docs.github.com/en/get-started/quickstart/create-a-repo>



The screenshot shows the GitHub 'Create repository' form. At the top, there are two input fields: 'Owner' with a dropdown menu showing 'daltonbc96' and a profile picture, and 'Repository name' with the text 'myRepo' and a green checkmark. Below these fields is a line of text: 'Great repository names are short and memorable. Need inspiration? How about [fuzzy-octo-palm-tree?](#)'. Underneath is a 'Description (optional)' text area. The next section is for visibility, with two radio buttons: 'Public' (unselected) and 'Private' (selected). Below the radio buttons is a section titled 'Initialize this repository with:' followed by the instruction 'Skip this step if you're importing an existing repository.' There are three checkboxes: 'Add a README file' (unselected), 'Add .gitignore' (unselected), and 'Choose a license' (unselected). Each checkbox has a brief description and a 'Learn more' link. At the bottom of the form is a green button labeled 'Create repository'.

Owner * Repository name *

 daltonbc96 / myRepo ✓

Great repository names are short and memorable. Need inspiration? How about [fuzzy-octo-palm-tree?](#)

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.

Initialize this repository with:
Skip this step if you're importing an existing repository.

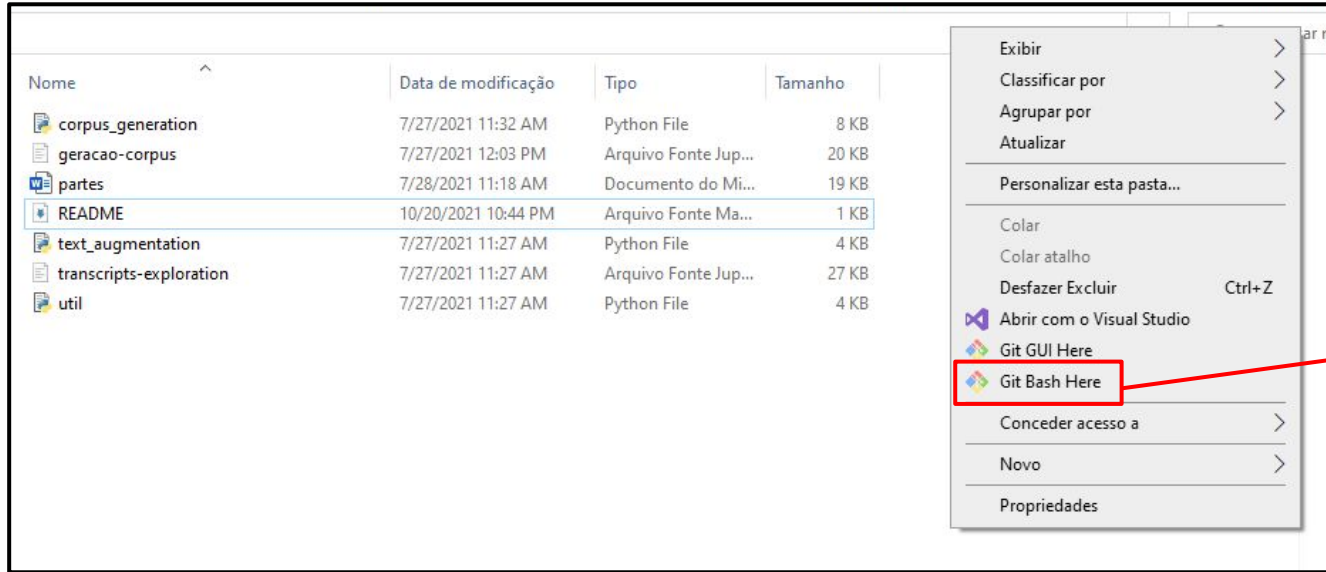
☐ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository



Create a folder on your computer with your files



Use the right mouse button to open Git Bash in the location of your files

Connecting my remote repository with my local repository

Quick setup — if you've done this kind of thing before

 Set up in Desktop or ☐ HTTPS ☐ SSH 

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a `README`, `LICENSE`, and `.gitignore`.

...or create a new repository on the command line

```
echo "# myRepo" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/daltonbc96/myRepo.git
git push -u origin main
```

After including the README file, add the other files you need to upload to the repository, for example:

```
$ add file.csv
$ add file2.py
```

Or use `$ git add .` to add all files at once

...or push an existing repository from the command line

```
git remote add origin https://github.com/daltonbc96/myRepo.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Use `$ git status` to check added files

Repository created

The screenshot shows the GitHub interface for a new repository. The repository is named 'myRep' and is set to 'Private'. It has 1 branch (main) and 0 tags. The repository was created 1 minute ago with commit c5df798. The file list shows a README.md and several Python and Jupyter Notebook files, all from the first commit. The README content is 'myRep'.

Search or jump to... / Pull requests Issues Marketplace Explore

daltonbc96 / myRep Private Unwatch 1 Star

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

daltonbc96 first commit		c5df798 1 minute ago 1 commit
README.md	first commit	1 minute ago
corpus_generation.py	first commit	1 minute ago
geracao-corpus.ipynb	first commit	1 minute ago
partes.docx	first commit	1 minute ago
text_augmentation.py	first commit	1 minute ago
transcripts-exploration.ipynb	first commit	1 minute ago
util.py	first commit	1 minute ago

README.md

myRep

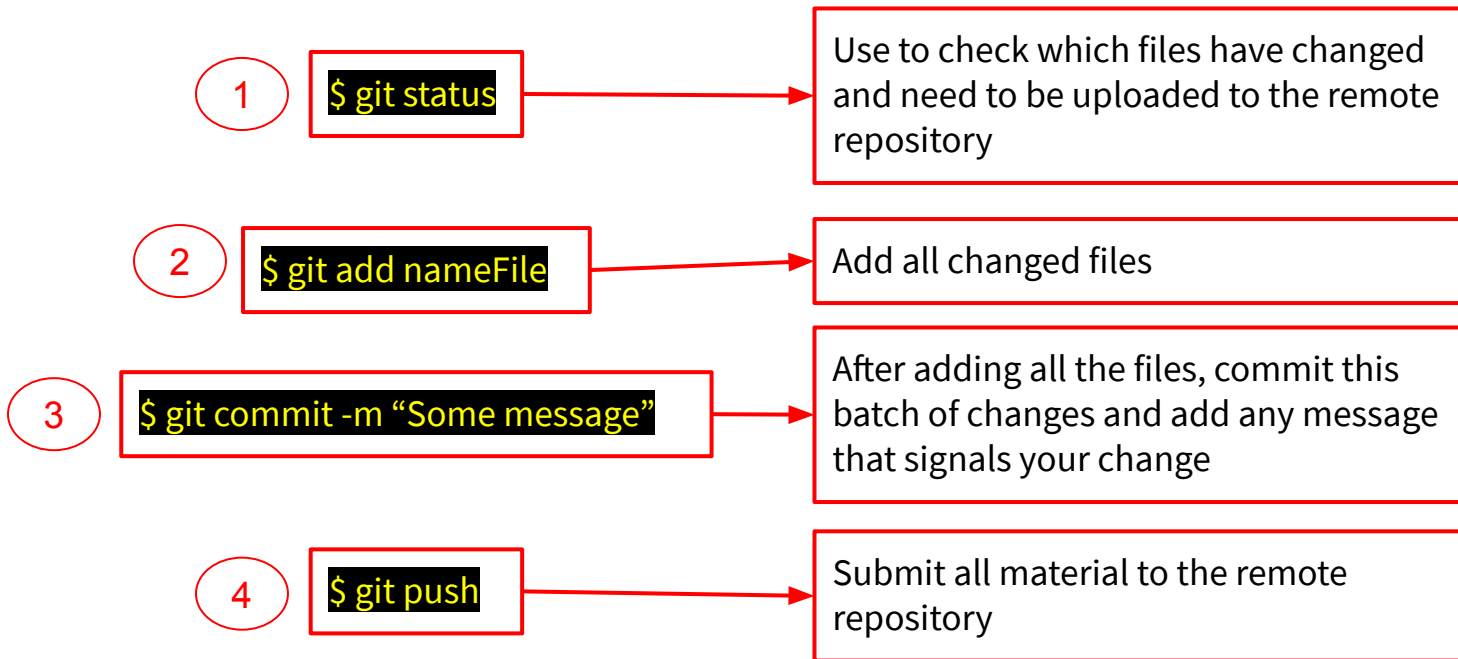
About No description, website, or topics provided. Readme

Releases No releases published Create a new release

Packages No packages published Publish your first package

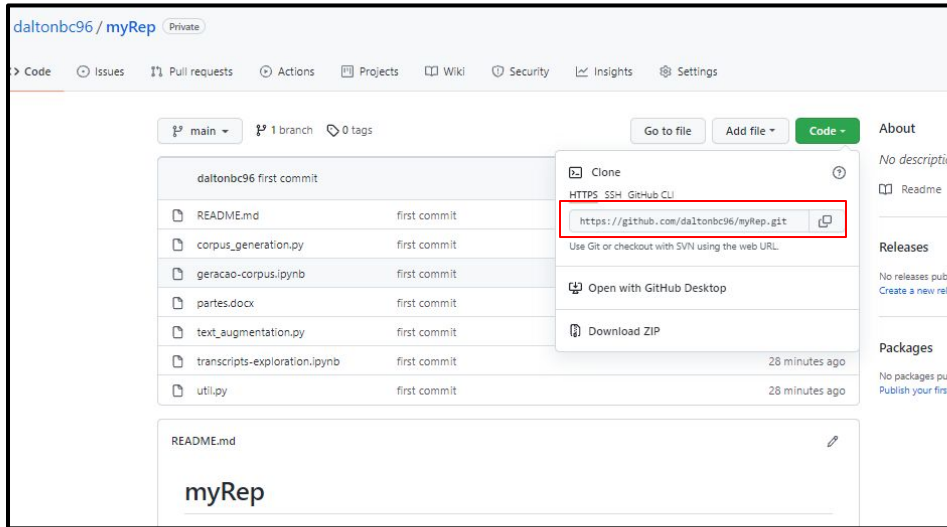
Updated a repository

- When changing the files that are in the local folder you need to update the files in the remote repository:



Clone a repository

- When you want to download material that is only in the remote repository on your computer, you should clone the material on your computer. You can update it as shown previously.

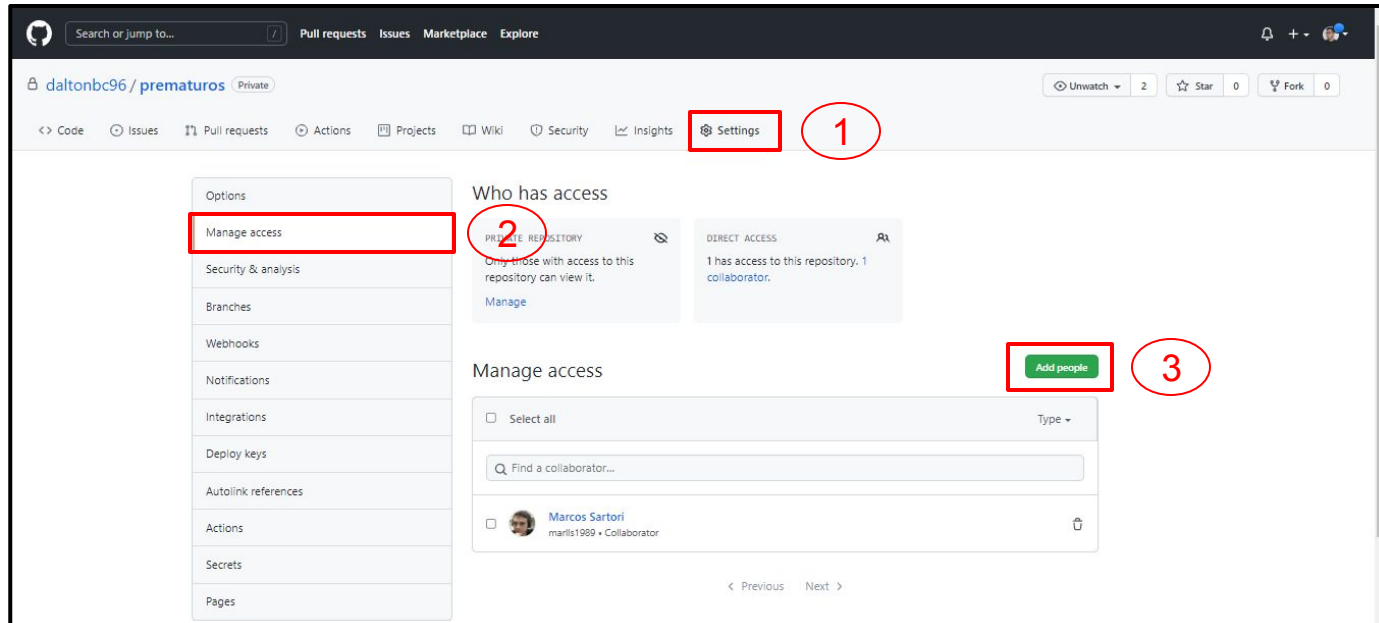


Open your Git Bash in the location you want to save the files and type the command:

```
$ git clone https://github.com/daltonbc96/myRep.git
```

Working with a team

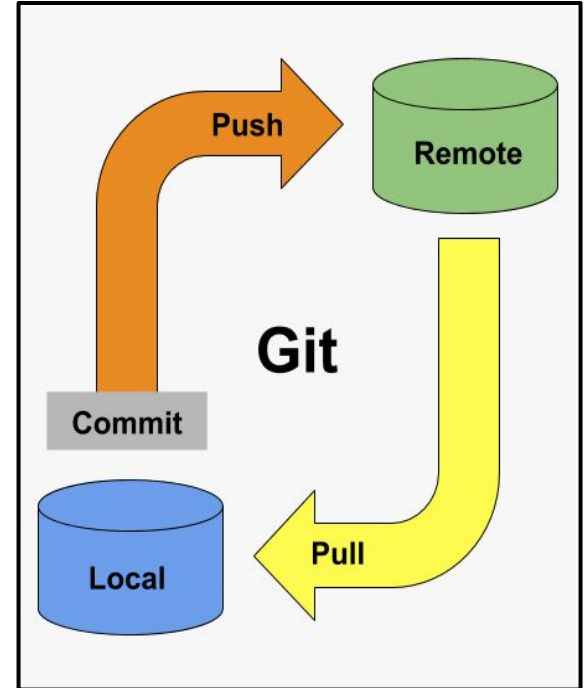
If you need to work with more people, you can add people to your repository and invite them to edit.



Updating a local repository

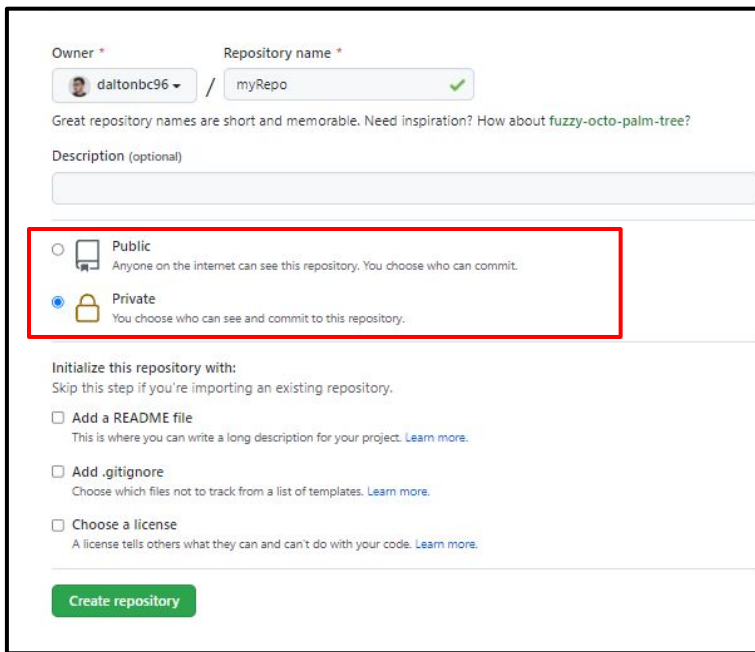
- If you are working in a team your colleagues can perform updates in the GitHub repository. **So you need to update the files with the current version from the remote repository before you start any editing.** It is important that the local and remote files are aligned, otherwise you will get conflicting files.
- Go into Git Bash and the address of your local repository and type the command:

```
$ git pull
```



Public or private?

- When your repository is **public**, everyone on the Internet can view your repository.
- When your repository is **private** only authorized people can access your repository.



The screenshot shows the GitHub repository creation interface. At the top, there are fields for 'Owner' (daltonbc96) and 'Repository name' (myRepo). Below these is a hint: 'Great repository names are short and memorable. Need inspiration? How about fuzzy-octo-palm-tree?'. A 'Description (optional)' text area follows. The 'Visibility' section has two radio buttons: 'Public' (unselected) and 'Private' (selected). The 'Private' option is highlighted with a red rectangle. Below this, the 'Initialize this repository with:' section offers three checkboxes: 'Add a README file', 'Add .gitignore', and 'Choose a license'. At the bottom is a green 'Create repository' button.

Owner * / Repository name *

Great repository names are short and memorable. Need inspiration? How about [fuzzy-octo-palm-tree?](#)

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.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

In GEMINI, we usually put the repositories in **private**. In public there are only materials that are finalized.

Final considerations

- Put in your repository only the essential stuff that will be changed constantly, such as: scripts and support material;
- Never put databases in GitHub, the ideal place for databases is the Duke Box.





Thanks!
dbc31@duke.edu