Introduction to GitHub Desktop

Why use version control?

- Have you ever found yourself making multiple versions of a document?

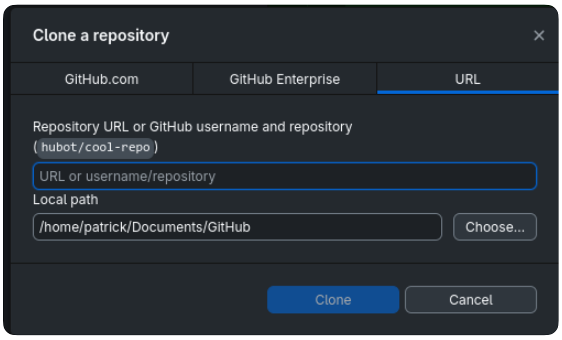
```
file_version1.docx
file_version2.docx
file_this_isthe_last_version.docx
file_this_is_really_the_last_one_for_sure.docx
```

- As you might have experienced, it is easy to lose track of changes made across versions.
- Moreover, it is easy to accidentally make changes to the wrong one
- These challenges are amplified when working on collaborative projects, where multiple people are editing the same file
  - Highly prone to version conflicts
- Version control allows you to make reversible changes to a common set of text-based files
  - Collaborators can track changes, either adopting themselves or merging them with their own
  - Can revert project to any state in its history without limit*
  - Can store and make changes locally without internet access, sharing when it's convenient for you

* Github has a 100mb file size limit

Cloning a Repository

- Can be found under File > Clone Repository.
- This creates a local copy of a repository, whether it's your own or somebody elses.



Repository URL or GitHub username and repository (`hubot/cool-repo`)

URL or username/repository

Local path

/home/patrick/Documents/GitHub

Screenshot_2026-02-08_16-26-59.png

In this case, you can clone a copy of our class repository by inputting the user name and repository name:

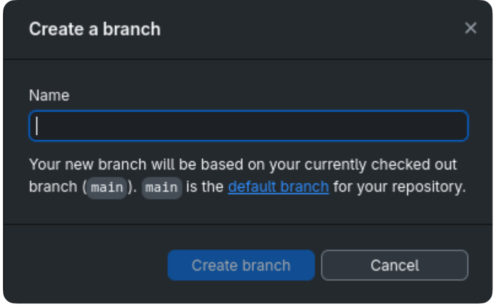
```
pchest/Data-Storytelling-2026
```

Pick a location on your computer that you will remember to store the repository.

NOTE: You can have a repository cloned at multiple locations on your device! They will only synchronize when you pull.

Creating a new branch

- Branches allow you to have multiple parallel versions of your code
- Very helpful if you are making local changes to a repository you don't own! (Hint hint)
- Go to Repository > Create Branch



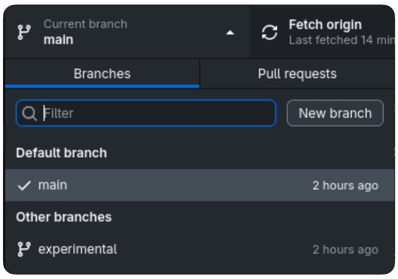
Create a branch

Name

Your new branch will be based on your currently checked out branch (`main`). `main` is the default branch for your repository.

Screenshot_2026-02-08_17-25-46.png

You can switch between branches using the top bar



Screenshot_2026-02-08_18-02-05.png

Pulling changes

If there are changes made on the main repository that you want to bring to your local folder, switch to the clean "main" branch and click "Fetch origin" in the top right panel.

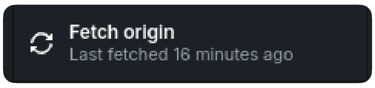
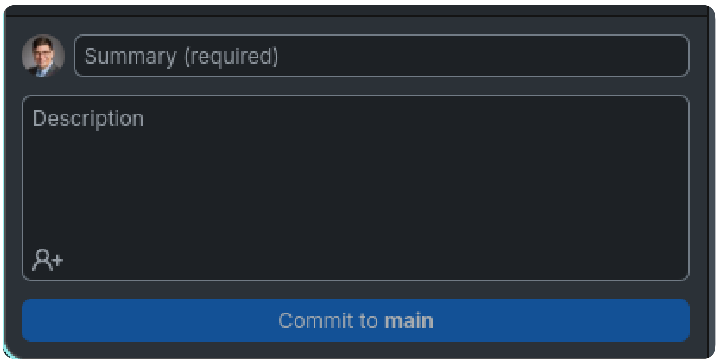


image-112.png

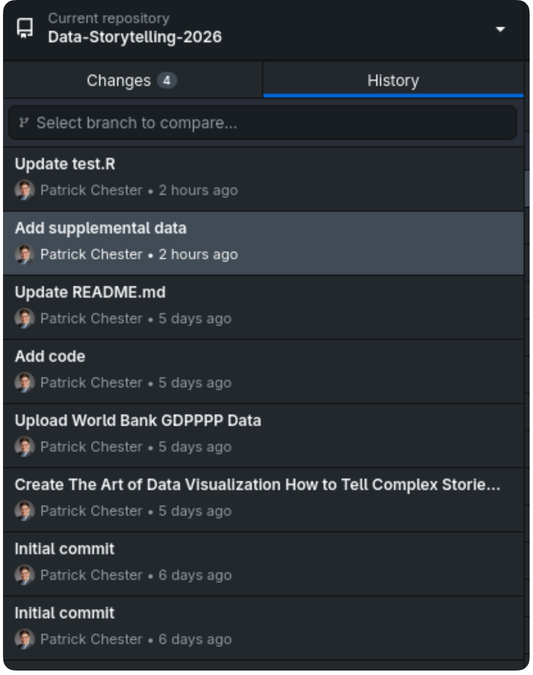
Making a commit

- Commits are a way of making specific, well-documented, and reversible changes to a project.
- First, let's make a change to `test.R`
  - For example, replacing "here()" with your repository's working directory
- Then go down to the lower left panel and input a short and (optionally) long description of the change.
  - This is important, as it helps you keep track of what changes you've made to your projects and why you made them
  - It also is very helpful when describing changes you've made to code/text to collaborators who are working on a common project



Screenshot_2026-02-08_17-54-40.png

You can check your commit history using the "History" tab on the left panel.



Current repository
Data-Storytelling-2026

Update test.R
Patrick Chester • 2 hours ago

Add supplemental data
Patrick Chester • 2 hours ago

Update README.md
Patrick Chester • 5 days ago

Add code
Patrick Chester • 5 days ago

Upload World Bank GDPPPP Data
Patrick Chester • 5 days ago

Create The Art of Data Visualization How to Tell Complex Storie...
Patrick Chester • 5 days ago

Initial commit
Patrick Chester • 6 days ago

Initial commit
Patrick Chester • 6 days ago

Screenshot_2026-02-08_18-05-40.png

Pushing Commits

- Once you have made local commits, you can push them to the global repository
- This will ensure that any other devices or collaborators can access the changes you made to