SPAAM

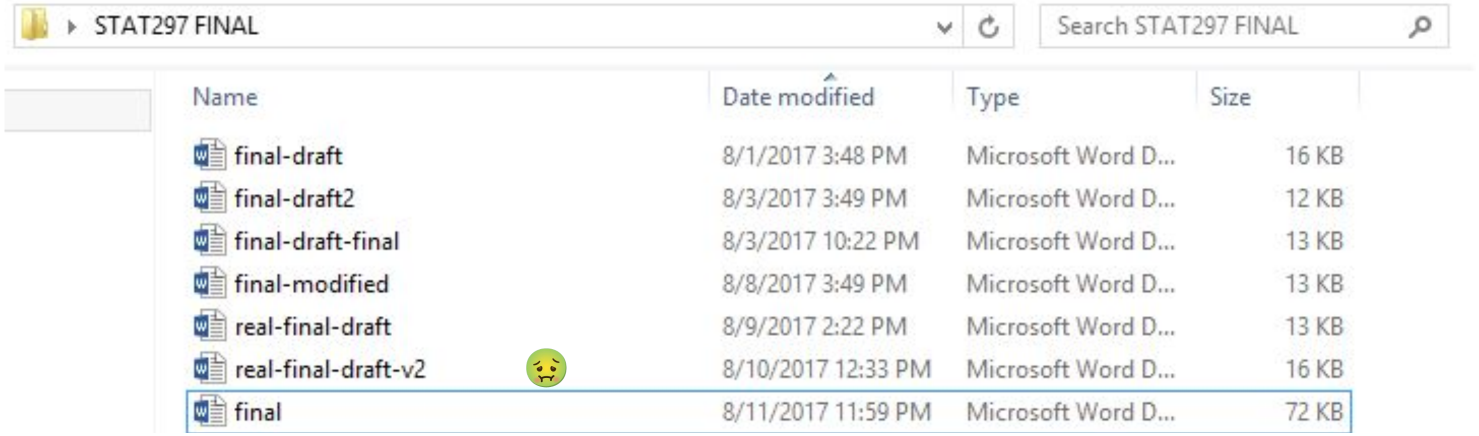
Standards, Precautions & Advances in Ancient Metagenomics

Practical 2B: Introduction to Git(Hub)

Megan Michel & James A. Fellows Yates

What is Git?

- **Version Control System (VCS)-** tracks changes to files over time



STAT297 FINAL

Search STAT297 FINAL

Name	Date modified	Type	Size
final-draft	8/1/2017 3:48 PM	Microsoft Word D...	16 KB
final-draft2	8/3/2017 3:49 PM	Microsoft Word D...	12 KB
final-draft-final	8/3/2017 10:22 PM	Microsoft Word D...	13 KB
final-modified	8/8/2017 3:49 PM	Microsoft Word D...	13 KB
real-final-draft	8/9/2017 2:22 PM	Microsoft Word D...	13 KB
real-final-draft-v2 🤢	8/10/2017 12:33 PM	Microsoft Word D...	16 KB
final	8/11/2017 11:59 PM	Microsoft Word D...	72 KB

M. Beckman, S. Guerrier, J. Lee, R. Molinari, S. Orso, I. Rudnytskyi, Eds., in *An Introduction to Statistical Programming Methods with R* (2020).

What is Git?

- **Version Control System (VCS)-** tracks changes to files over time (e.g. scripts, simple csvs, small fastas, etc.)
- Enables restoration of old versions, modifications to previous changes, tracking contributions by multiple people, etc.
- Primarily of text files, but can also do other types

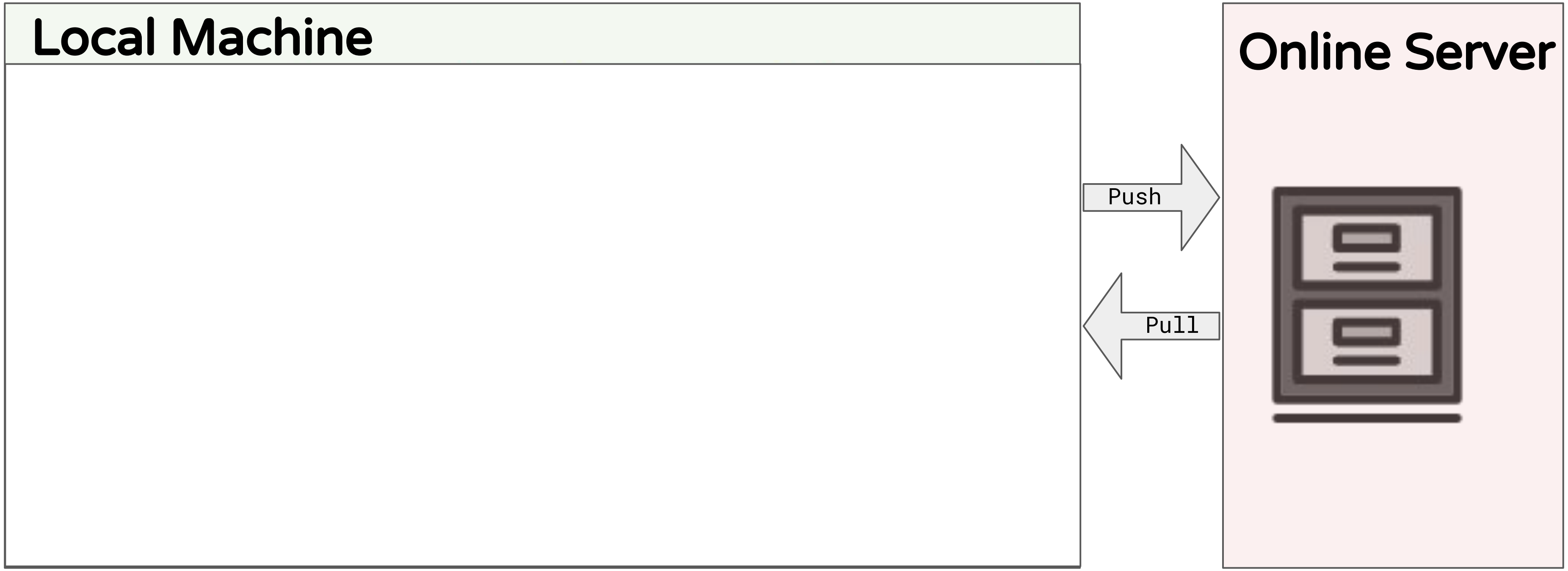
What is GitHub?

- Remote hosting service for version-controlled repositories
  - Field standard approach for sharing data, code, etc.
  - User-friendly GUI
- Similar open-source alternative: GitLab

Why use Git(Hub)

1. Keep a (deep) backup of your work
2. Revert to an old version/ modify previous changes
3. Allow multiple contributors to work simultaneously
4. Test new code before updating public version
5. Share your data, code, and results!

Git(Hub) in graphics



Modified from S. Chacon, B. Straub, *Pro Git* (Apress, Second Edition., 2022), *The Expert's Voice*.

Git(Hub) in graphics

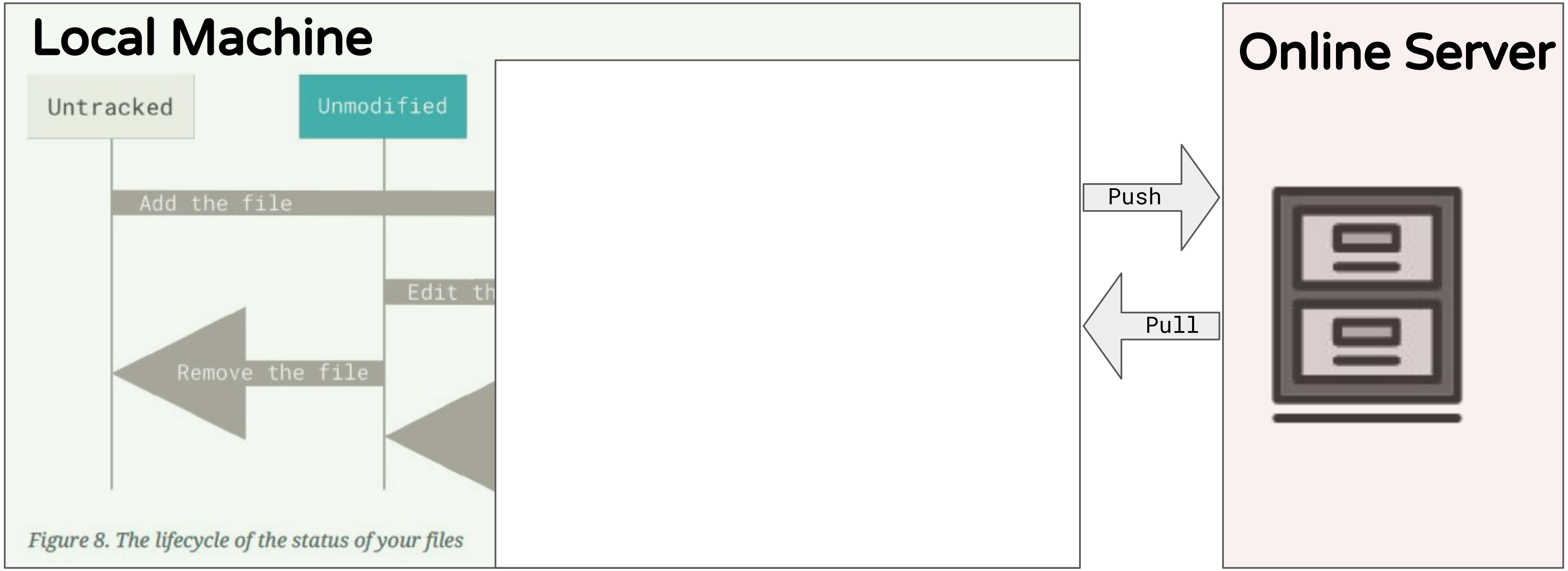
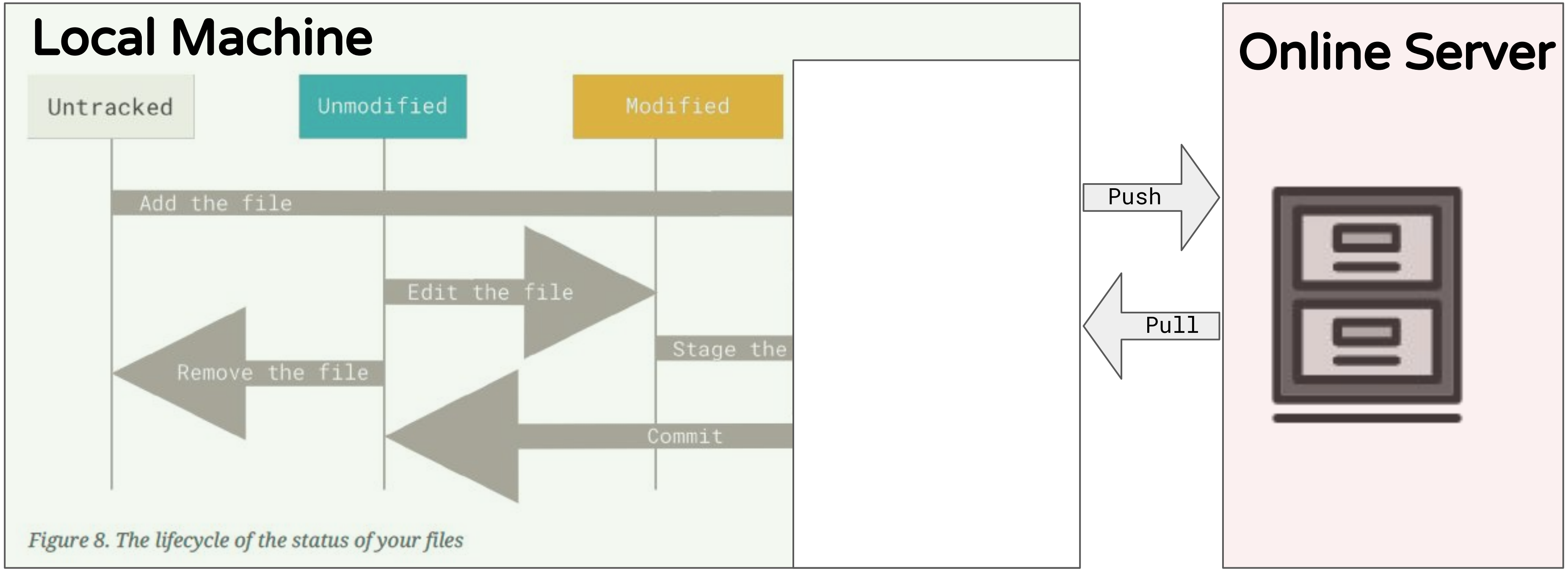


Figure 8. The lifecycle of the status of your files

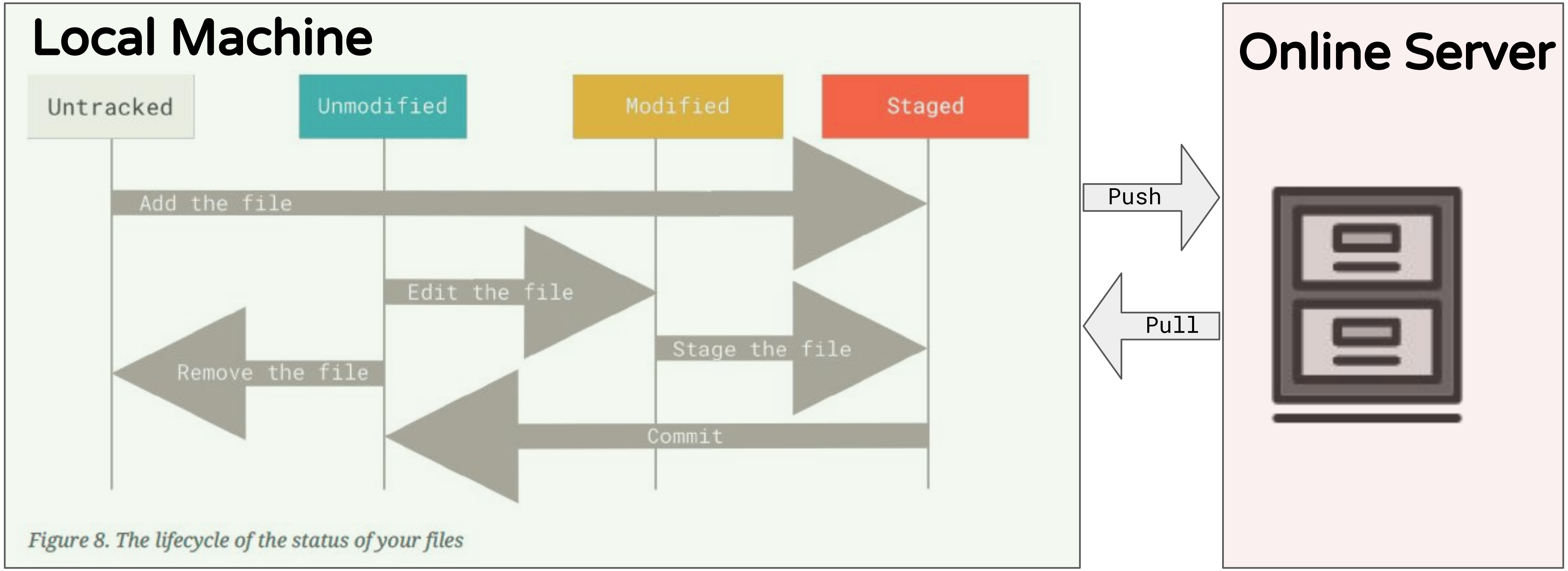
Modified from S. Chacon, B. Straub, *Pro Git* (Apress, Second Edition., 2022), *The Expert's Voice*.

Git(Hub) in graphics



Modified from S. Chacon, B. Straub, *Pro Git* (Apress, Second Edition., 2022), *The Expert's Voice*.

Git(Hub) in graphics



Modified from S. Chacon, B. Straub, *Pro Git* (Apress, Second Edition., 2022), *The Expert's Voice*.

SSH setup

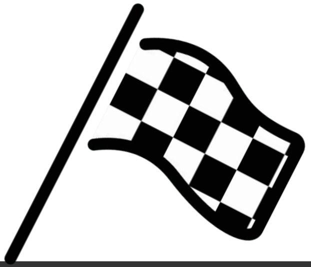
How to never type a password again!

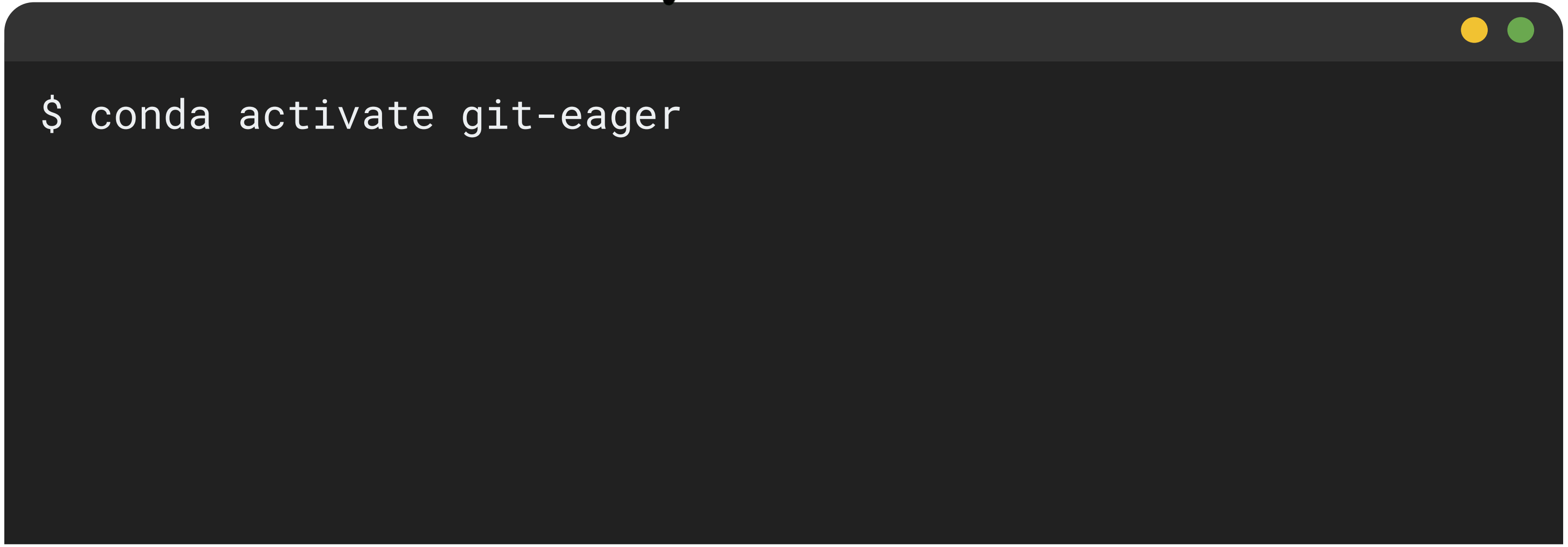
SSH keys

- Replaces password with file containing a cryptographic ultra-secure password
- Very common for working with servers
- We will generate a public key (shared with github) and a private key (kept on your computer)
    - Behind the scenes: the public and private keys are compared to authenticate
- SSH key files can be reused (e.g. copied to a new laptop)

Start your terminals!

Time for the practical portion





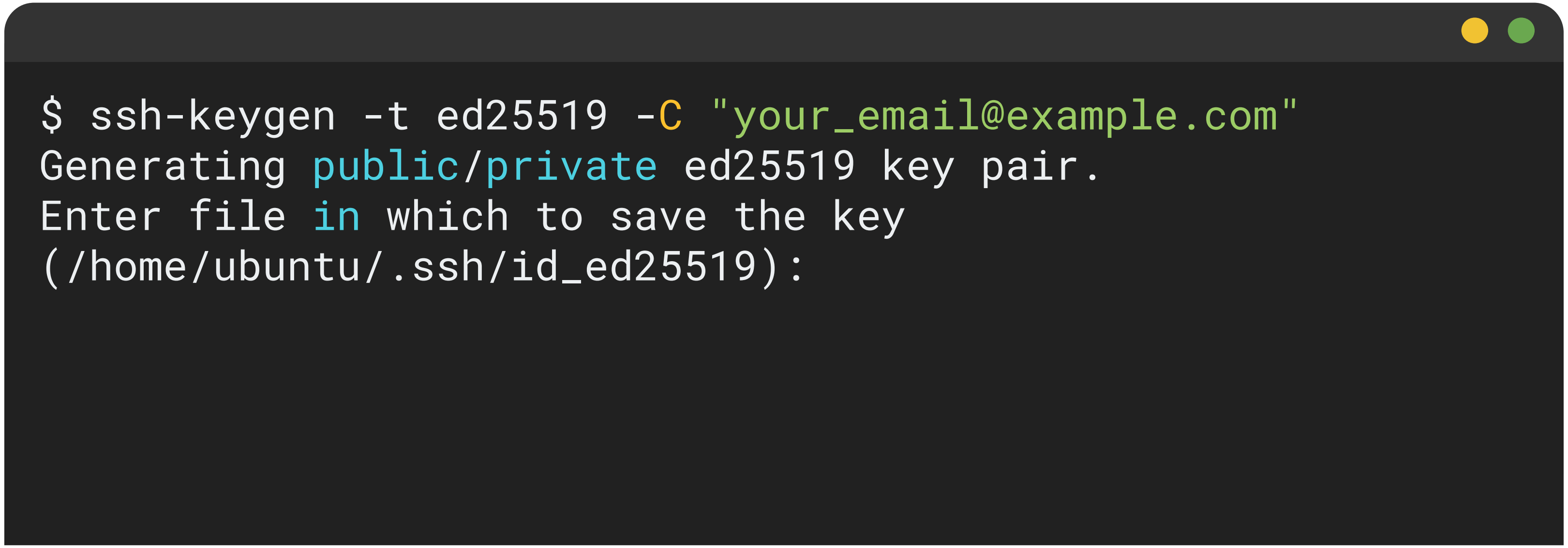
```
$ conda activate git-eager
```

SSH key Setup

Make a new key with your email address:

```
$ ssh-keygen -t ed25519 -C "your_email@example.com"
```

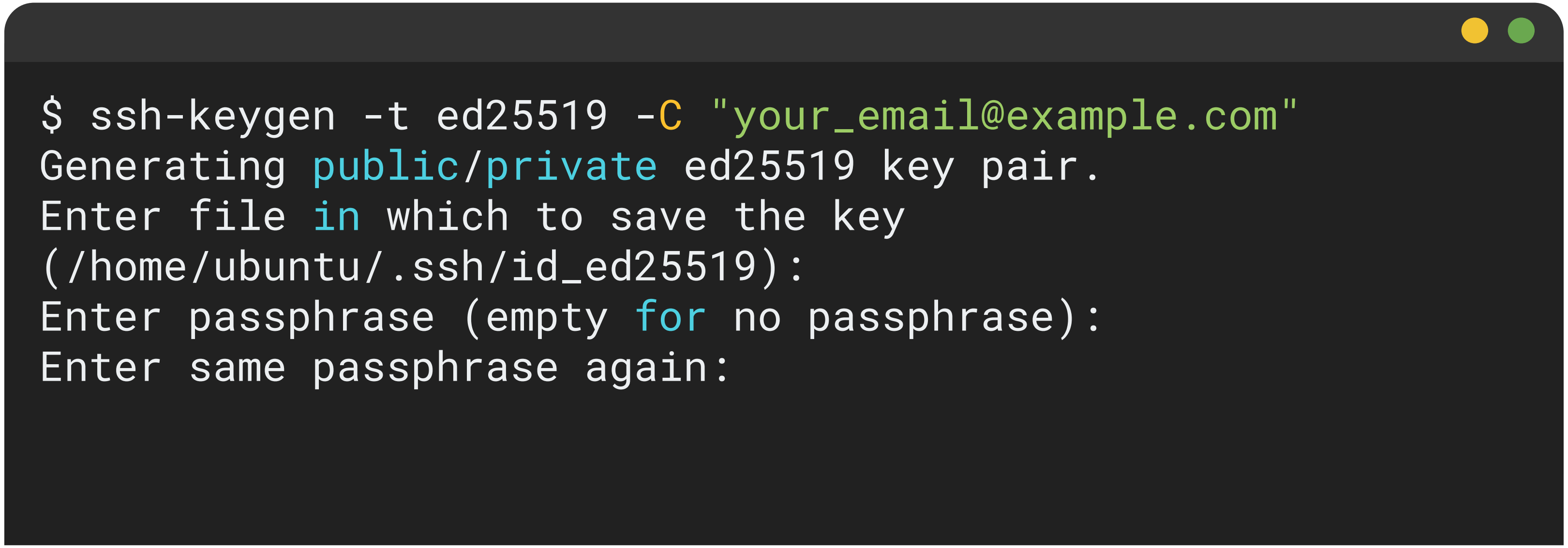
Press enter (default file location)



```
$ ssh-keygen -t ed25519 -C "your_email@example.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key
(/home/ubuntu/.ssh/id_ed25519):
```

SSH key Setup

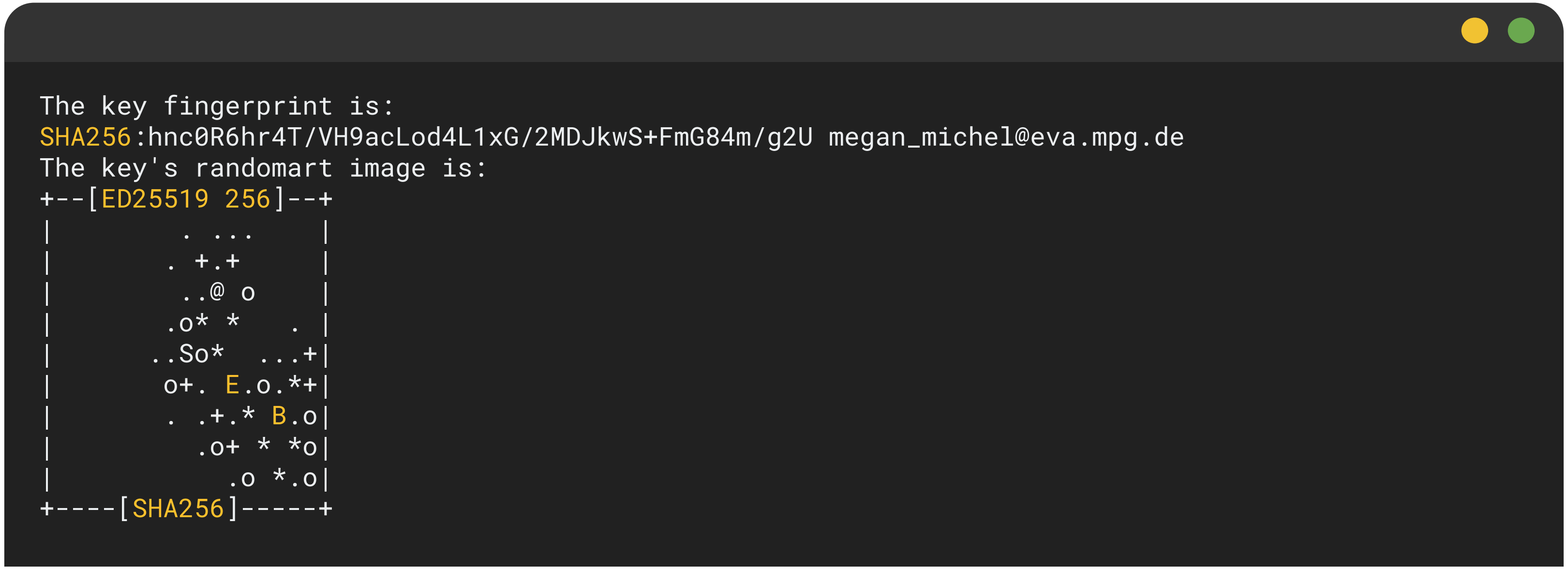
Recommended to skip passphrase entry



```
$ ssh-keygen -t ed25519 -C "your_email@example.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key
(/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

SSH key Setup

Should something like this:



```
The key fingerprint is:
SHA256:hnc0R6hr4T/VH9acLod4L1xG/2MDJkwS+FmG84m/g2U megan_michel@eva.mpg.de
The key's randomart image is:
+--[ED25519 256]--+
|        . ...    |
|       . +.+     |
|        ..@ o    |
|       .o* *   . |
|      ..So*  ...+|
|       o+. E.o.*+|
|       . .+.* B.o|
|         .o+ * *o|
|           .o *.o|
+----[SHA256]-----+
```

SSH key Setup

Check for *id_ed25519** and *id_ed25519*.pub* files

```
$ cd ~/.ssh/
$ ls id*
```

SSH key Setup

1. Check ssh program is running

```
$ eval "$(ssh-agent -s)"
  Agent pid 59566
```

2. Give SSH your key to record

```
$ ssh-add ~/.ssh/id_ed25519

Identity added: /home/ubuntu/.ssh/id_ed25519 (megan_michel@eva.mpg.de)
```

SSH key Setup

On Github:

1. Settings > SSH & GPG Keys > New SSH Key
2. Give a name of key (title: “spaam-summer-school”)
3. Paste PUBLIC key

```
$ cat ~/.ssh/id_ed25519.pub # manually copy paste
```

4. Press Add SSH key

SSH key Setup

Check that it worked!

```
$ ssh -T git@github.com
Hi meganemichel! You've successfully authenticated, but GitHub does not
provide shell access.
```

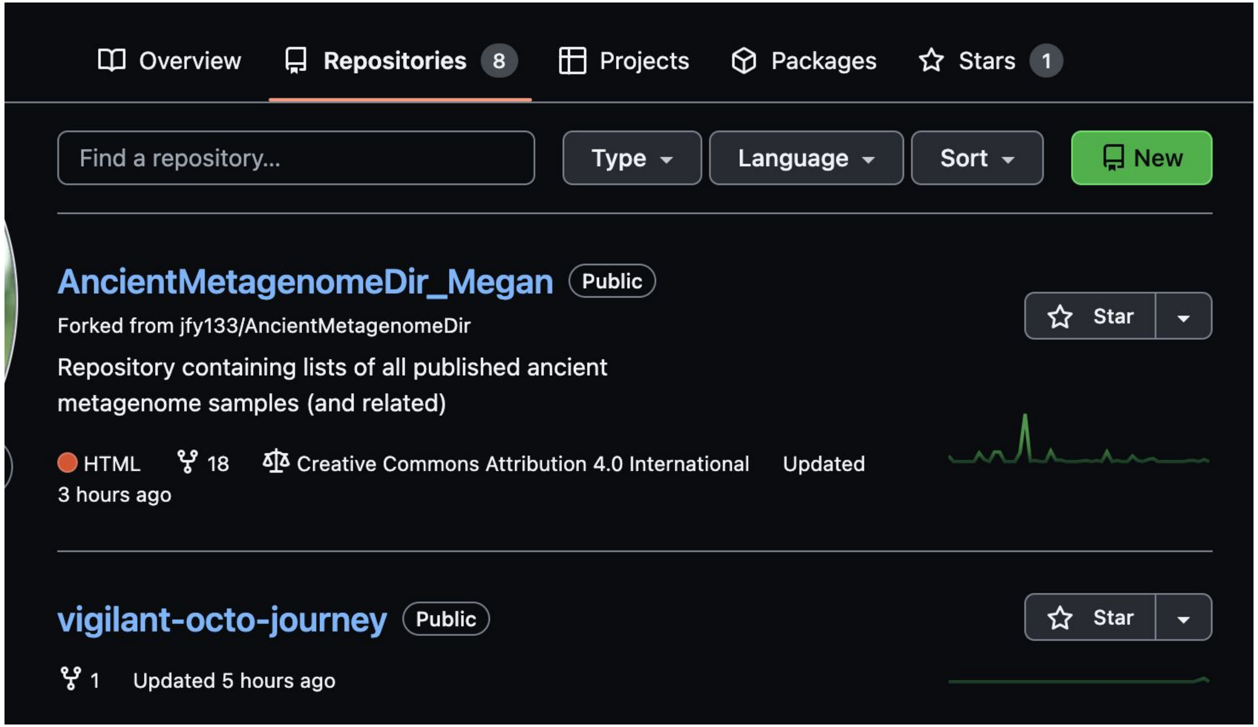
⚠️ If you see an error message like this, say yes

```
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ECDSA key fingerprint is SHA256:p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

The only 6 commands

you really need to know

Make your own repository



AncientMetagenomeDir_Megan Public

Forked from jfy133/AncientMetagenomeDir

Repository containing lists of all published ancient metagenome samples (and related)

HTML 18 Creative Commons Attribution 4.0 International Updated 3 hours ago

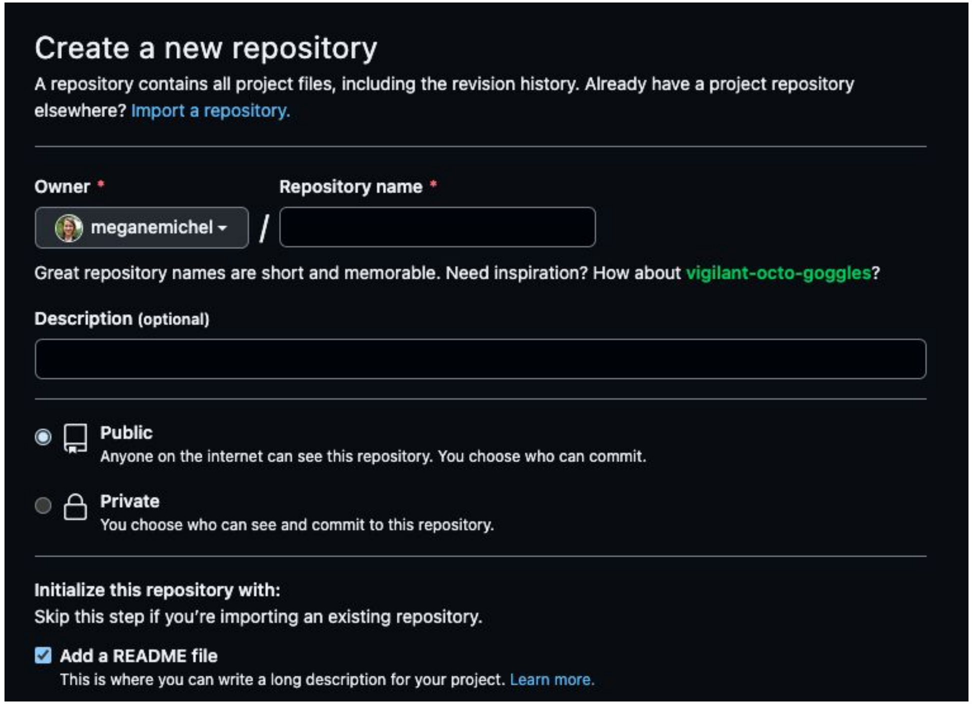
vigilant-octo-journey Public

1 Updated 5 hours ago

Make your own repository

1. Name your repository
2. Select **Add a README file**
3. Choose **Create Repository**

⚠️ For the remainder of this section, replace 'vigilant-octo-journey' with the name of your new repo



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * meganemichel / Repository name *

Great repository names are short and memorable. Need inspiration? How about vigilant-octo-goggles?

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.

Change directories

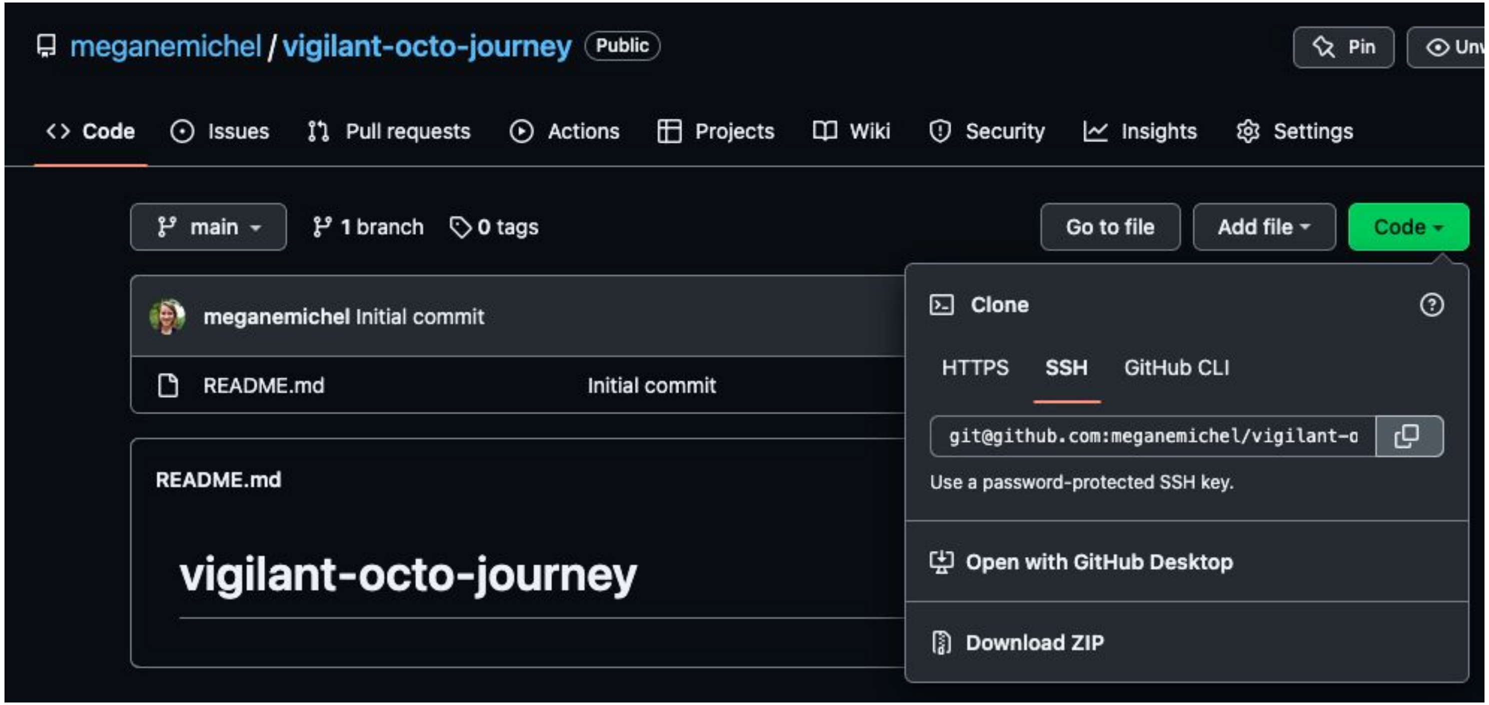
We will save our data in the following directory:

```
$ cd /vol/volume/2b-introduction-to-github/
```

git clone

git clone

Clone a `remote` (online) repository to your `local` machine



meganemichel Initial commit

README.md Initial commit

README.md

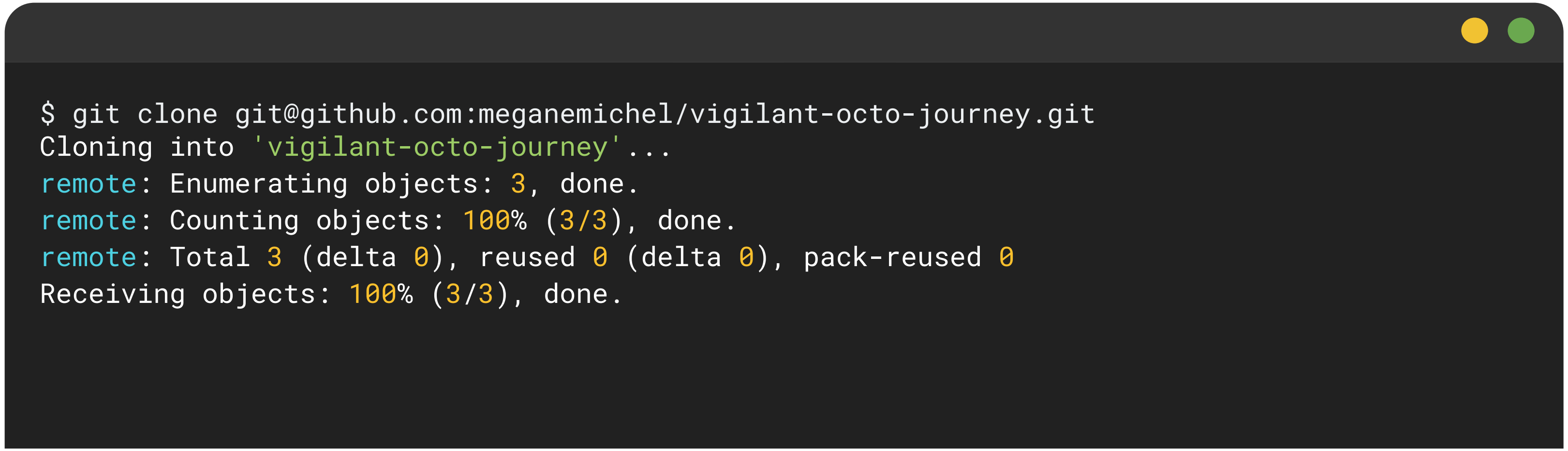
vigilant-octo-journey

```
git@github.com:meganemichel/vigilant-o
```

Use a password-protected SSH key.

git clone

Clone a `remote` (online) repository to your `local` machine

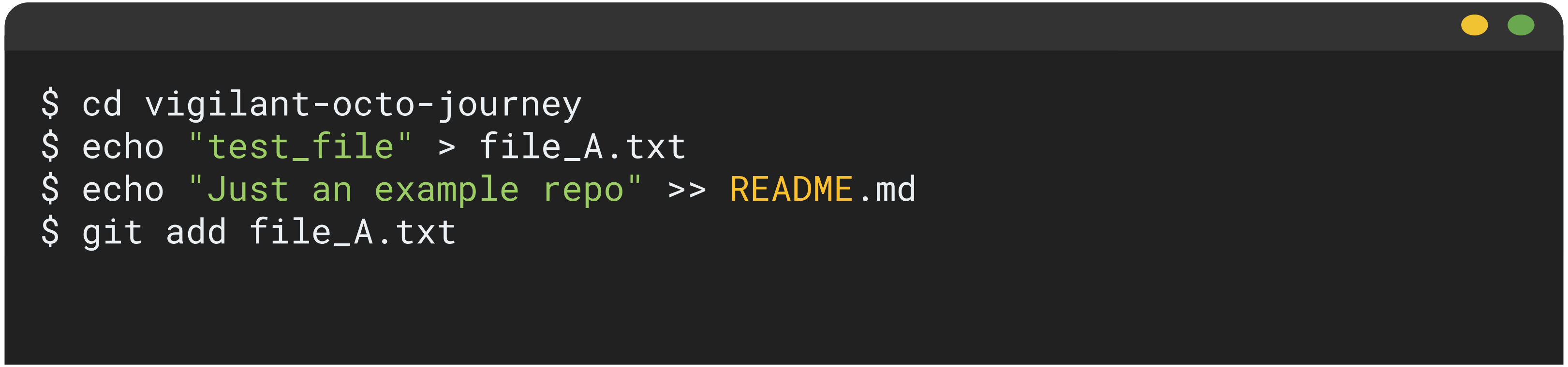


```
$ git clone git@github.com:meganemichel/vigilant-octo-journey.git
Cloning into 'vigilant-octo-journey'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

git add

git add <file>

Add a new or modified file into a ‘staging’ area (ready to send to the repository) on your `local` machine



```
$ cd vigilant-octo-journey
$ echo "test_file" > file_A.txt
$ echo "Just an example repo" >> README.md
$ git add file_A.txt
```

git status

git status

Check what files are `locally` changed, staged, etc.

```
$ git status
On branch main
Your branch is up-to-date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   file_A.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md
```

⚠️ Try to add the README.md as well and check the status again

git commit

git commit -m "<change_description>"

Package or ‘save’ the changes into a single ‘commit’ with a message of the changes.

- Each commit has a unique ‘hash’ ID
- Will not change: stored ‘forever’ in git history

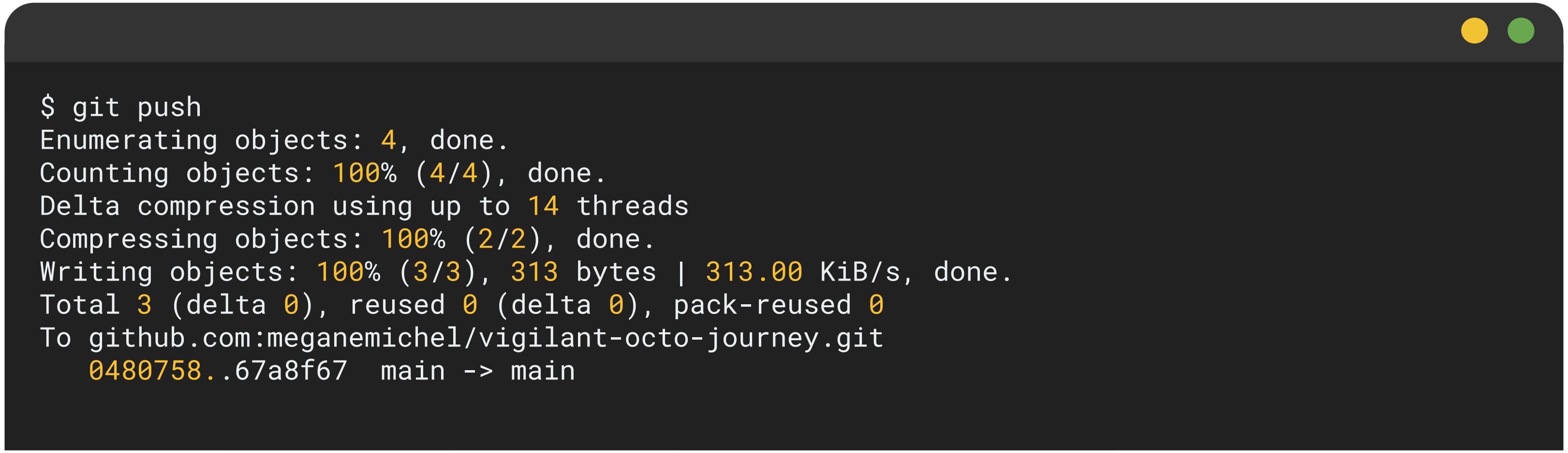
```
$ git commit -m "Add example file"
[main c58ac5f] Add example file
 2 files changed, 2 insertions(+)
 create mode 100644 file_A.txt
```

⚠️ You may see a warning that your name and email were configured automatically. You can safely ignore that for now.

git push

git push

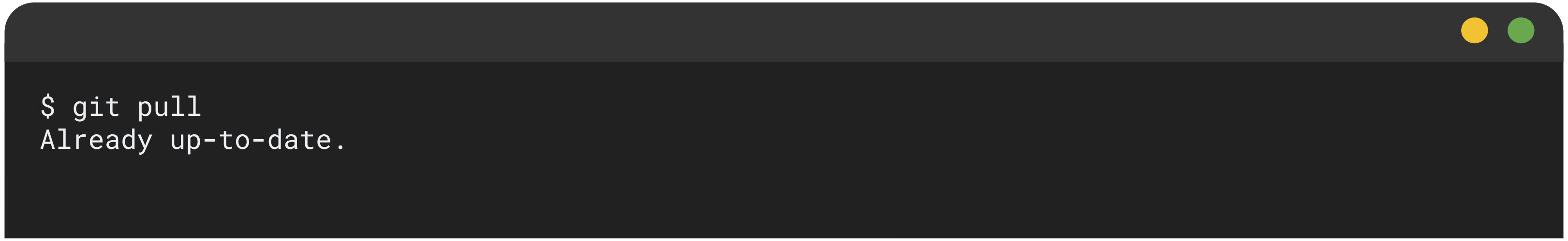
Push (upload) your `local` commit to the `remote` (online) repository



```
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 14 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 313 bytes | 313.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:meganemichel/vigilant-octo-journey.git
   0480758..67a8f67  main -> main
```

git pull

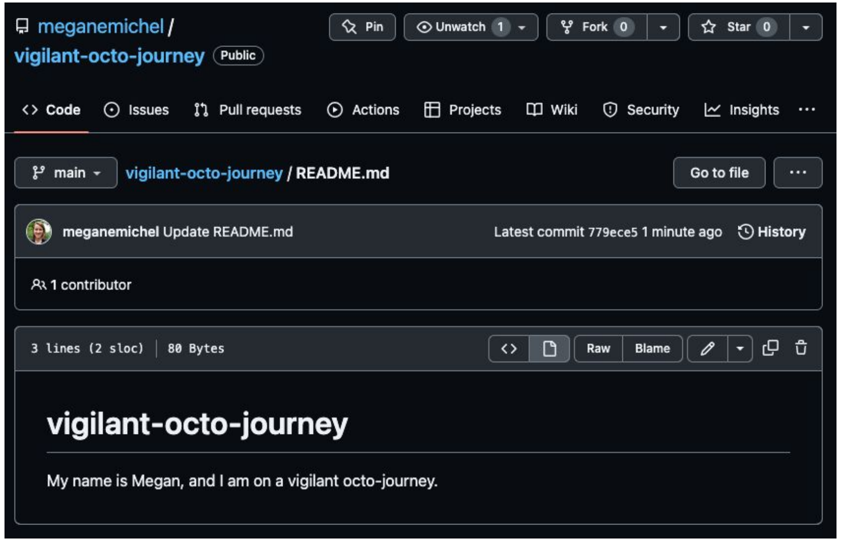
Pull (download) any commits from the `remote` to your `local` repository clone



```
$ git pull
Already up-to-date.
```

git pull

Pull (download) any commits from the `remote` to your `local` repository clone



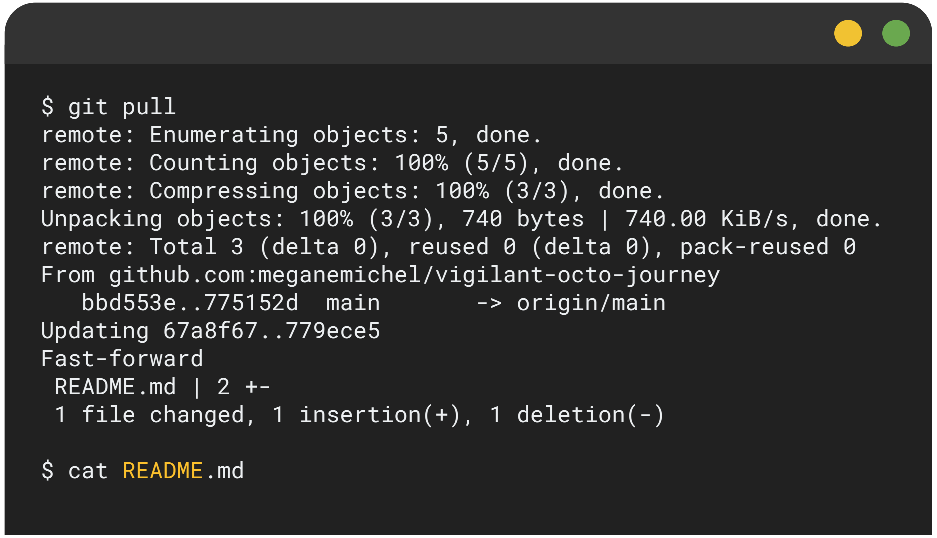
meganemichel Update README.md   Latest commit 779ece5 1 minute ago

1 contributor

3 lines (2 sloc) | 80 Bytes

vigilant-octo-journey

My name is Megan, and I am on a vigilant octo-journey.



```
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
Unpacking objects: 100% (3/3), 740 bytes | 740.00 KiB/s, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
From github.com:meganemichel/vigilant-octo-journey
   bbd553e..775152d  main       -> origin/main
Updating 67a8f67..779ece5
Fast-forward
 README.md | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

$ cat README.md
```

Summary: the 6 commands

1. git clone
2. git add
3. git status
4. git commit
5. git push
6. git pull

Working collaboratively

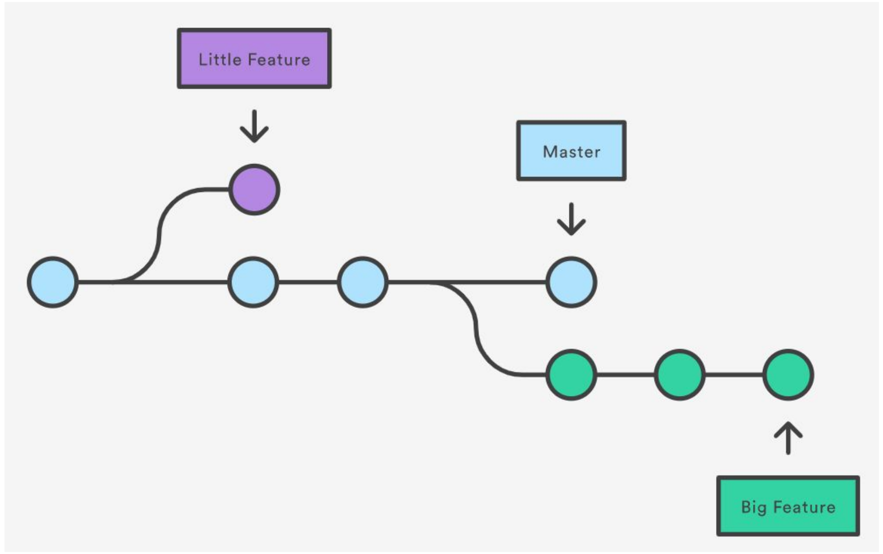
i.e. make changes without interfering with other people's changes?

Branch

Branch

Copy of main repository, within the repository

- Edit and prototype without breaking main files
- e.g. development branches
- Recommended for small-team projects

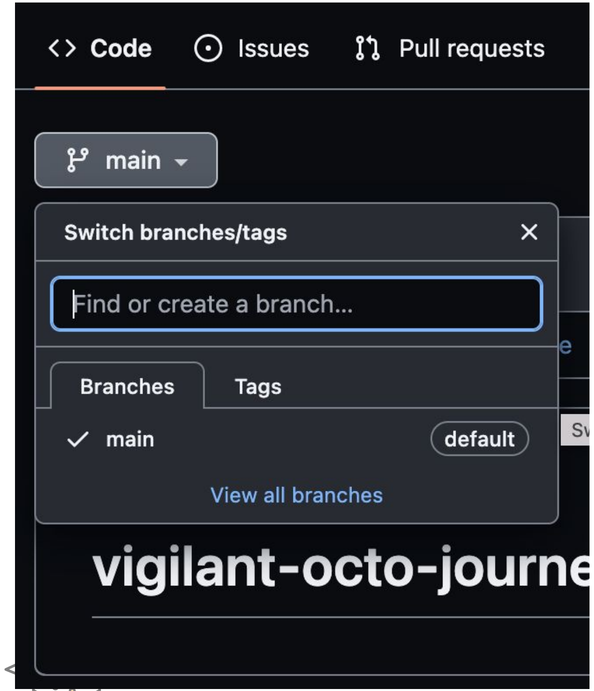


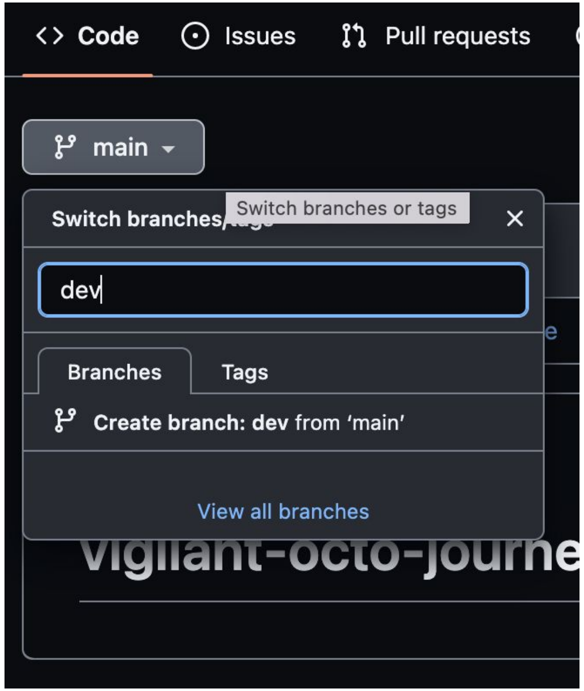
https://east.fm/refcards/git/branch/using-branches.html

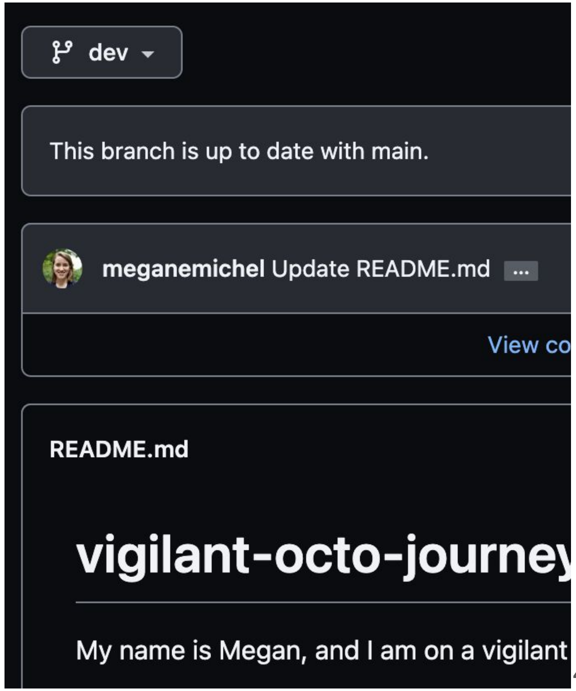
git switch

git switch

Making a new branch on GitHub







This branch is up to date with main.

meganemichel Update README.md

My name is Megan, and I am on a vigilant

git switch

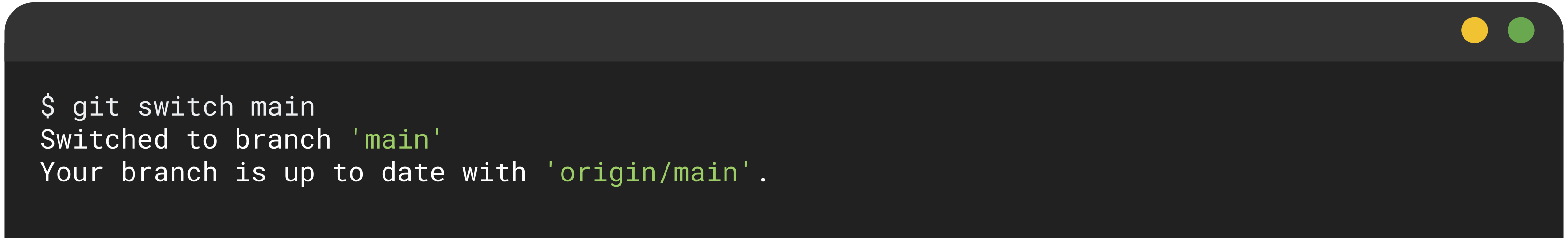
To make a new branch on command line

```
$ git switch -c new-feature
Switched to a new branch 'new-feature'
```

(Old command: `git checkout -b`*)*

git switch

Go back to the `main` branch



```
$ git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

git switch

⚠️ Commit changes to save to branch!

```
$ git switch new-feature

$ echo "What is a vigilant-octo-journey anyway?" >> README.md

$ git status
On branch new-feature
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

$ git switch main
M    README.md
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

git add/commit/push

```
$ git switch new_feature
$ git add README.md
$ git commit -m "Update README"
$ git push
```

⚠️ If branch created on CLI

```
$ git push --set-upstream origin new-feature
```

To also push the new-feature branch on GitHub (first time only)

Code peer review

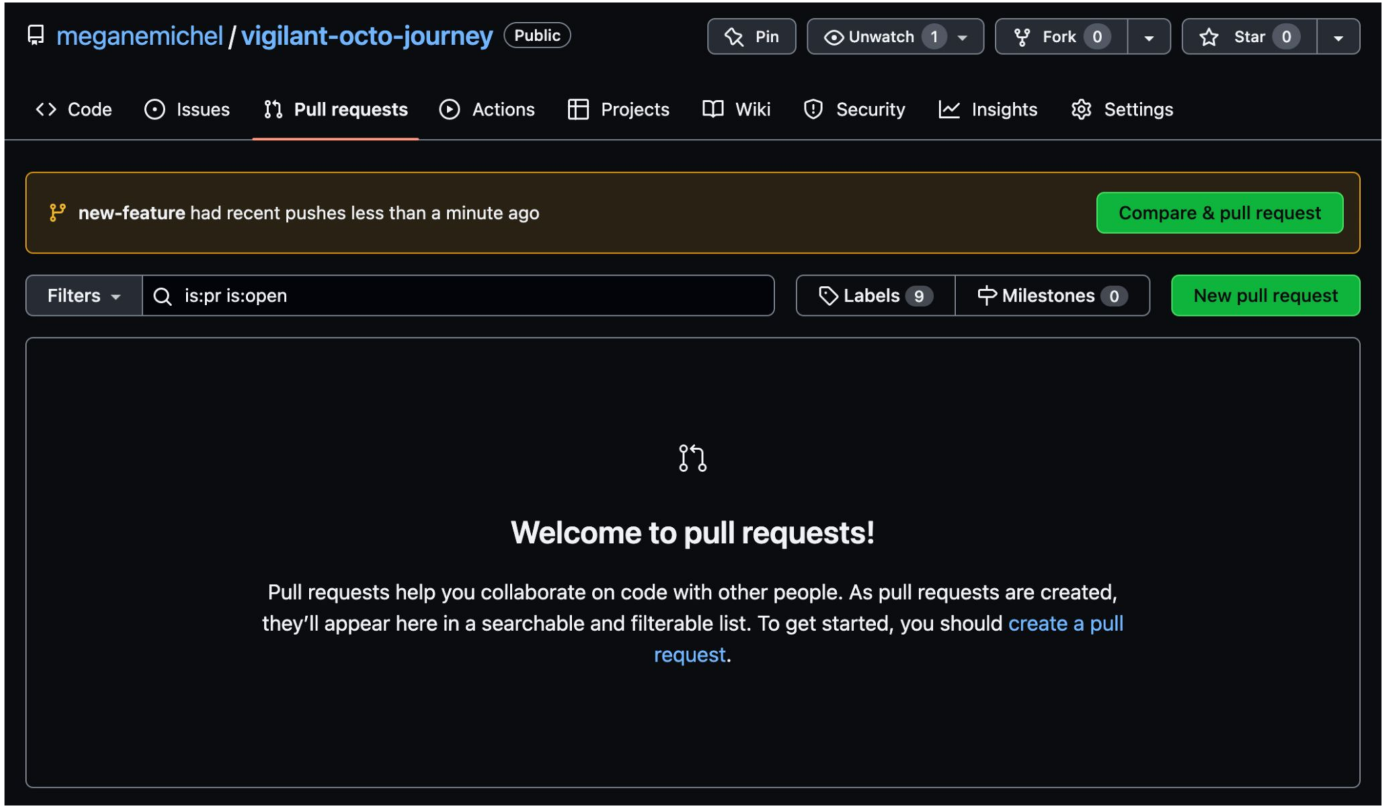
Pull Request

Pull Request

Pull request: propose changes to a branch from another branch

- a.k.a PRs
- e.g., **request** to pull your changes from dev into main branch
- Allows others to comment and make suggestions before merging into main branch

Pull Request

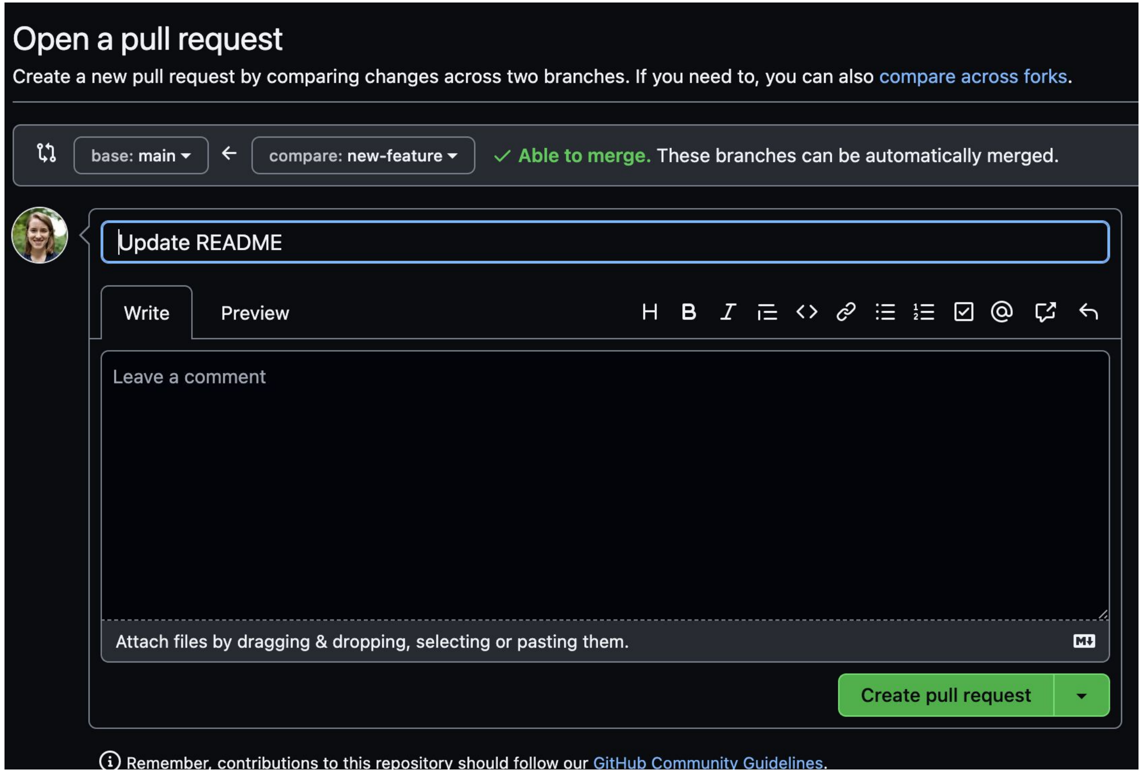


new-feature had recent pushes less than a minute ago

Welcome to pull requests!

Pull requests help you collaborate on code with other people. As pull requests are created, they'll appear here in a searchable and filterable list. To get started, you should create a pull request.

Pull Request


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: main ← compare: new-feature ✓ Able to merge. These branches can be automatically merged.

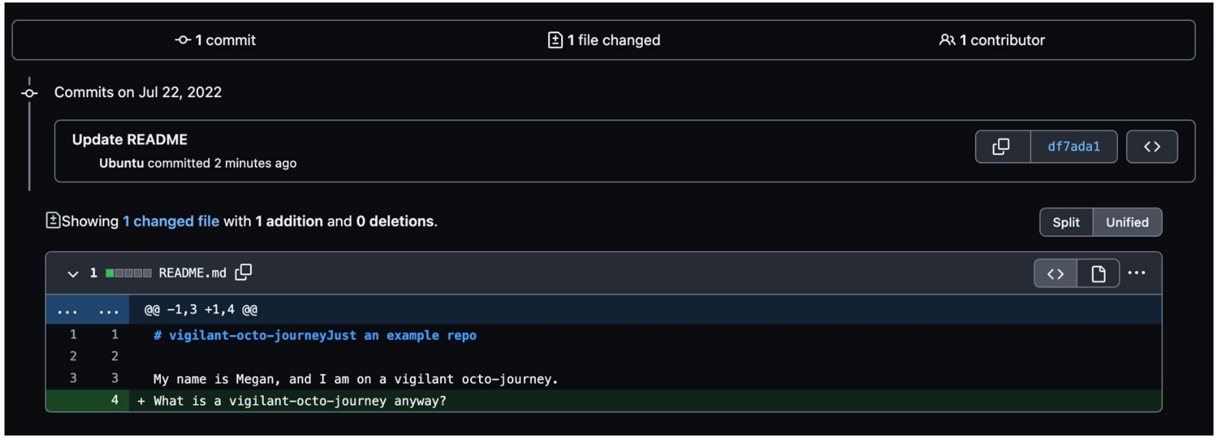
Update README

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Pull Request


1 commit 1 file changed 1 contributor

Commits on Jul 22, 2022

Update README
Ubuntu committed 2 minutes ago df7ada1

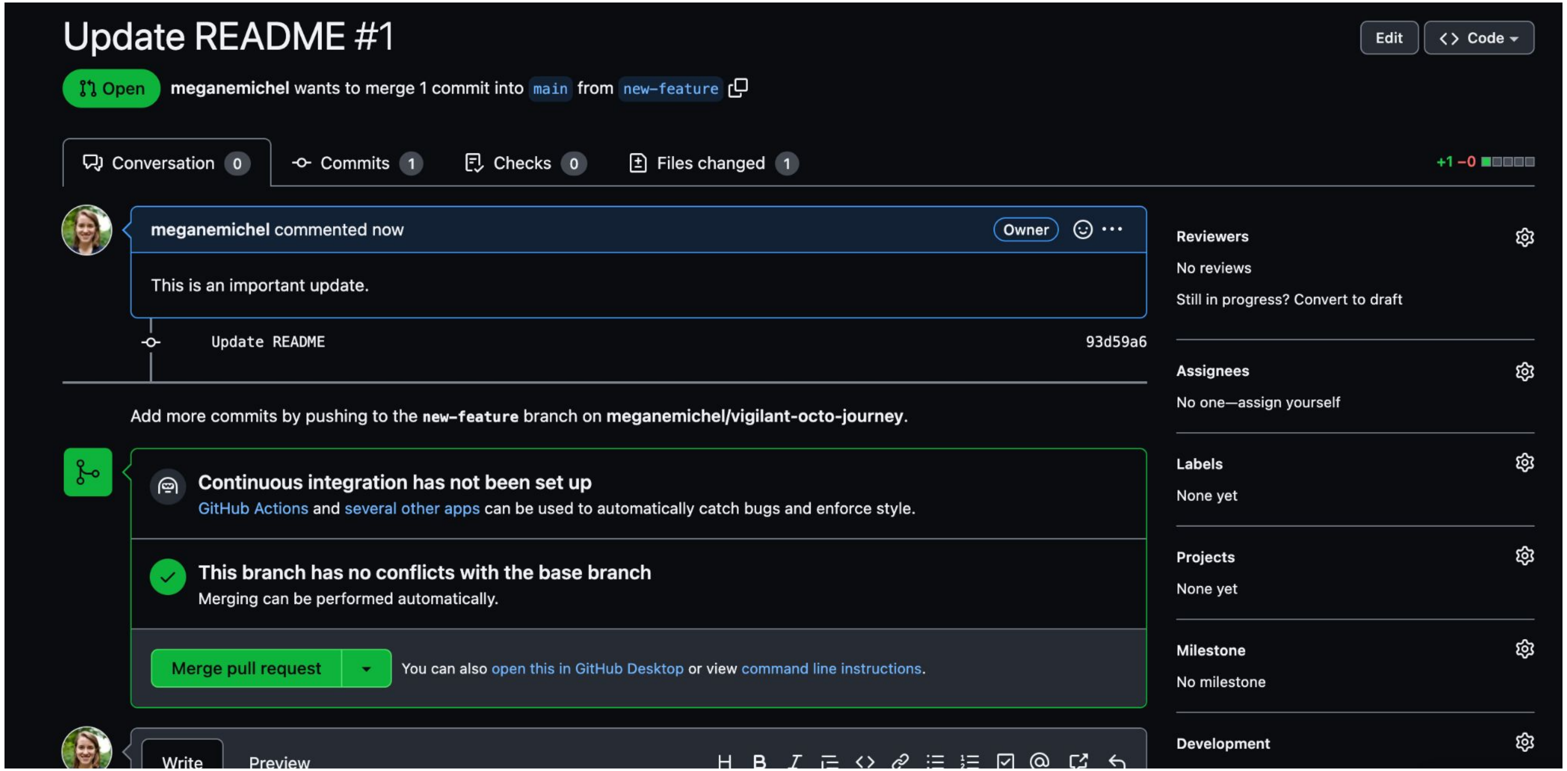
Showing **1 changed file** with **1 addition** and **0 deletions**. Split Unified

1 README.md

```
@@ -1,3 +1,4 @@
  # vigilant-octo-journeyJust an example repo

  My name is Megan, and I am on a vigilant octo-journey.
+ What is a vigilant-octo-journey anyway?
```

Pull Request


Update README #1

Open meganemichel wants to merge 1 commit into main from new-feature

Conversation 0 · Commits 1 · Checks 0 · Files changed 1 · +1 −0

meganemichel commented now — Owner

This is an important update.

Update README 93d59a6

Add more commits by pushing to the **new-feature** branch on **meganemichel/vigilant-octo-journey**.

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

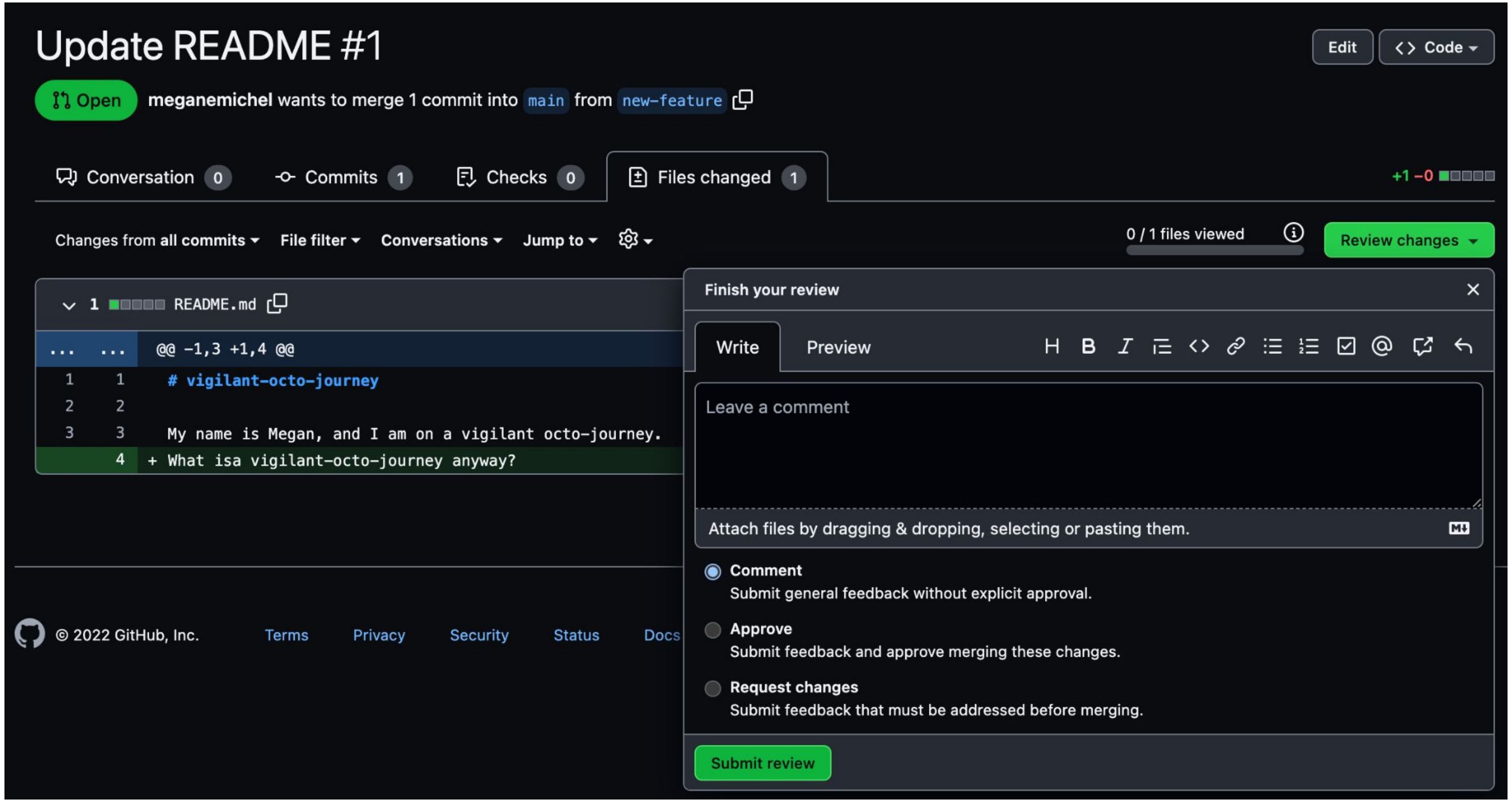
This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request — You can also open this in GitHub Desktop or view command line instructions.

Write · Preview

Reviewers: No reviews. Still in progress? Convert to draft
Assignees: No one—assign yourself
Labels: None yet
Projects: None yet
Milestone: No milestone
Development

Pull Request



Update README #1

meganemichel wants to merge 1 commit into `main` from `new-feature`

README.md

```
@@ -1,3 +1,4 @@
 # vigilant-octo-journey

 My name is Megan, and I am on a vigilant octo-journey.
+ What isa vigilant-octo-journey anyway?
```

Finish your review

Leave a comment

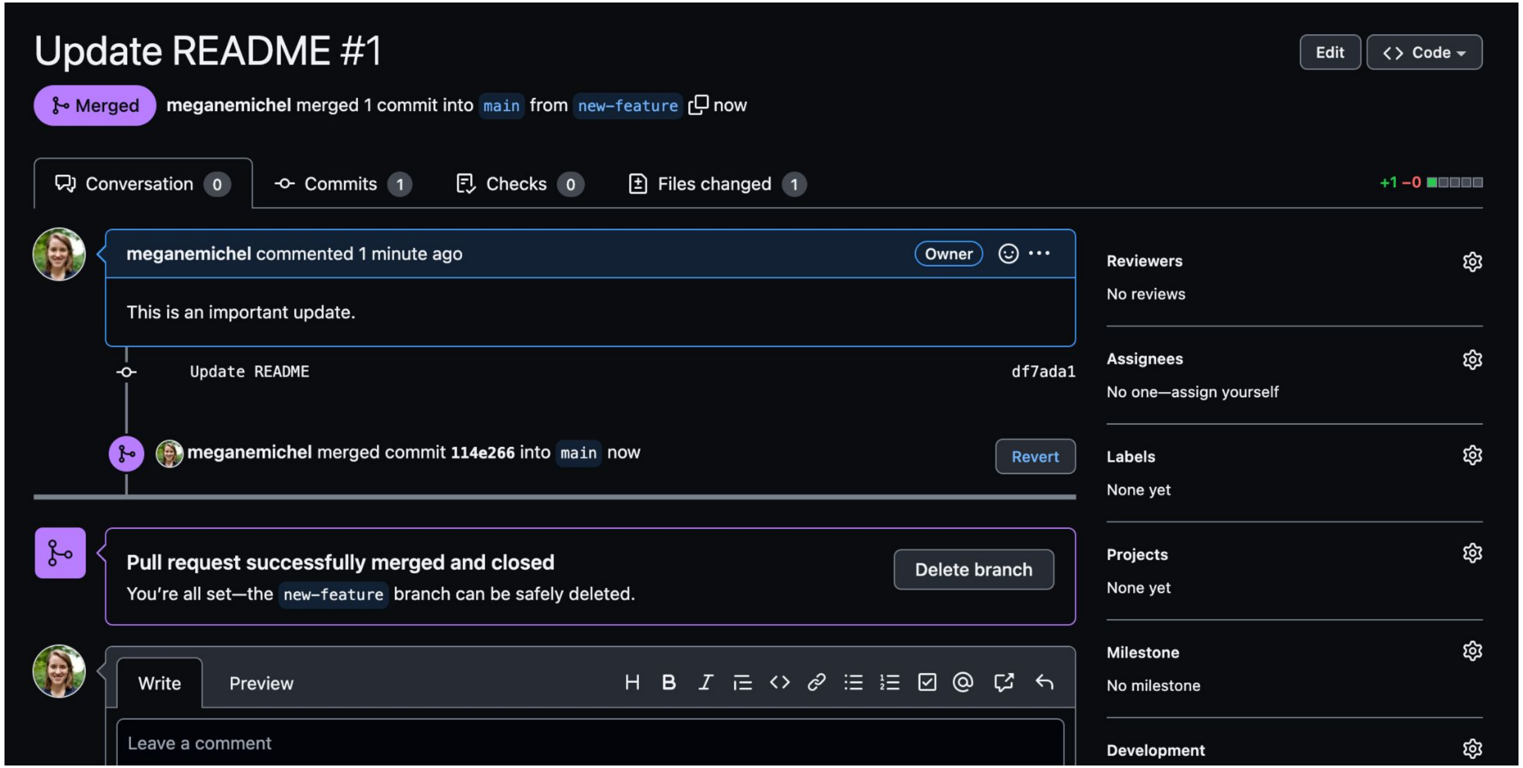
Attach files by dragging & dropping, selecting or pasting them.

[✓ Comment] Submit general feedback without explicit approval.

Approve — Submit feedback and approve merging these changes.

Request changes — Submit feedback that must be addressed before merging.

Merged Pull Request



Update README #1

Merged meganemichel merged 1 commit into `main` from `new-feature` now

meganemichel commented 1 minute ago

This is an important update.

`Update README` df7ada1

meganemichel merged commit 114e266 into `main` now

Pull request successfully merged and closed

You're all set—the `new-feature` branch can be safely deleted.

Reviewers: No reviews

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

Development

Changes After Review

How to implement requested changes?

- Push the changes to your branch!
- The PR automatically tracks changes to your branch
- Request re-review!

What if I disagree

with the requested changes?

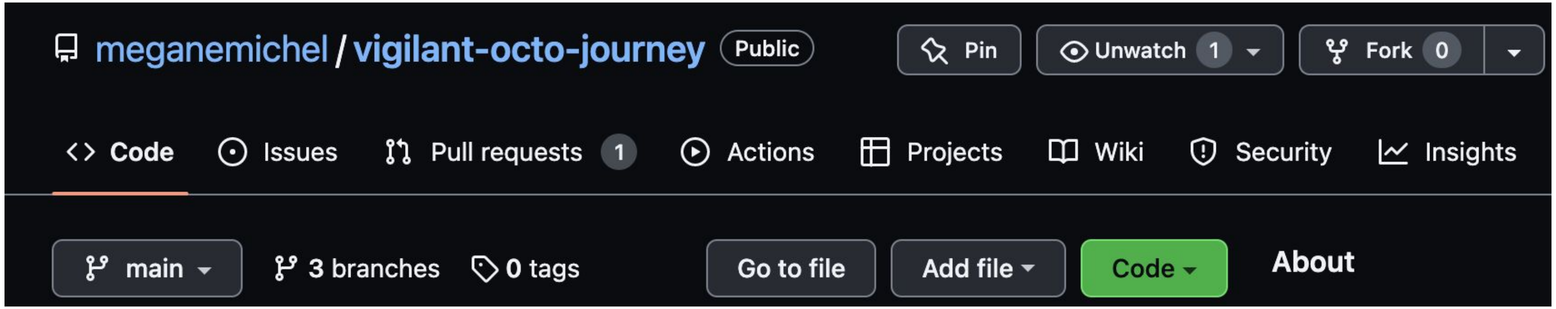
(or working with *big* teams)

Fork

Fork

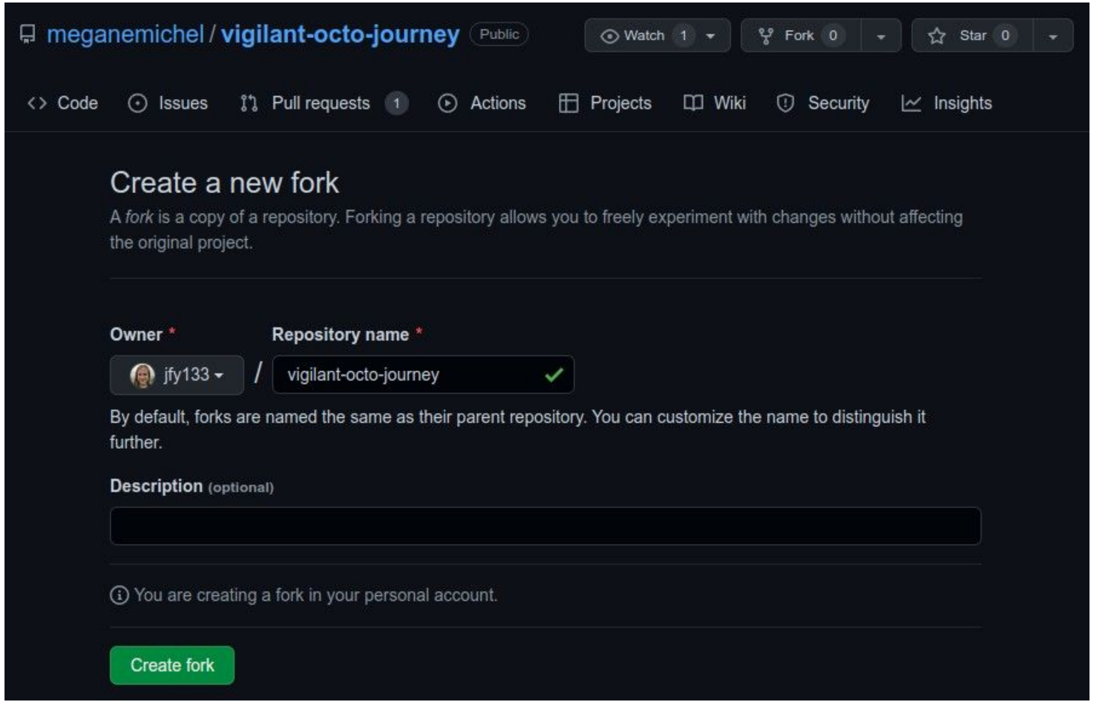
Independent copy of main repository, outside the repository

- Where you can edit without breaking files
- For 'going your own way'
- Provides extra protection - recommended for large-team projects



Making a fork

https://github.com/meganemichel/vigilant-octo-journey



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Owner * jfy133 / Repository name * vigilant-octo-journey

By default, forks are named the same as their parent repository. You can customize the name to distinguish it further.

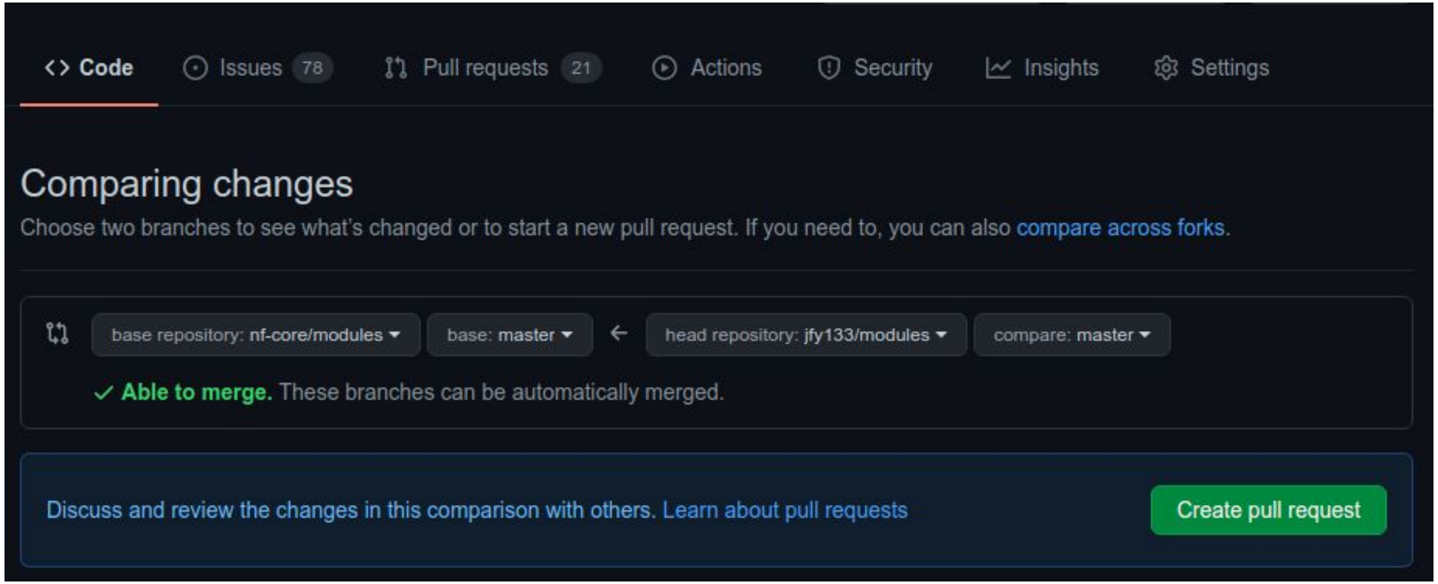
Description (optional)

You are creating a fork in your personal account.

Create fork

Pull request

Also pull request to and from forks!



Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks.

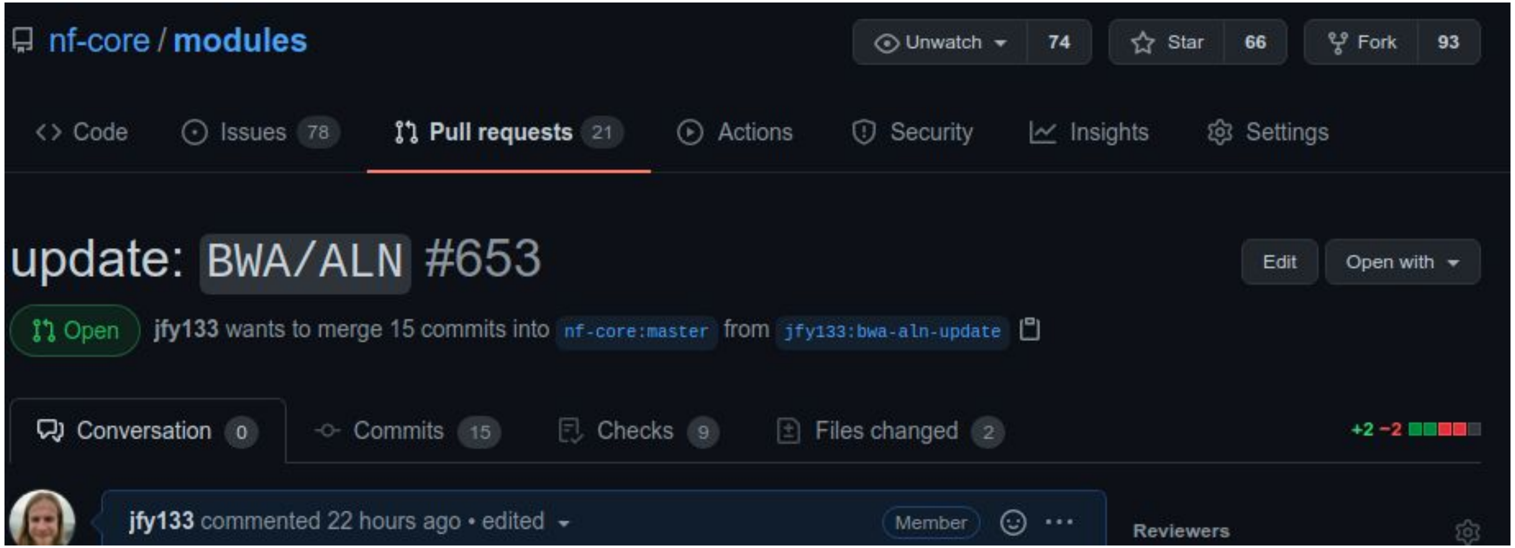
base repository: nf-core/modules ▾ base: master ▾ ← head repository: jfy133/modules ▾ compare: master ▾

✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. Learn about pull requests

Pull request

Also pull request into forks!



nf-core / modules

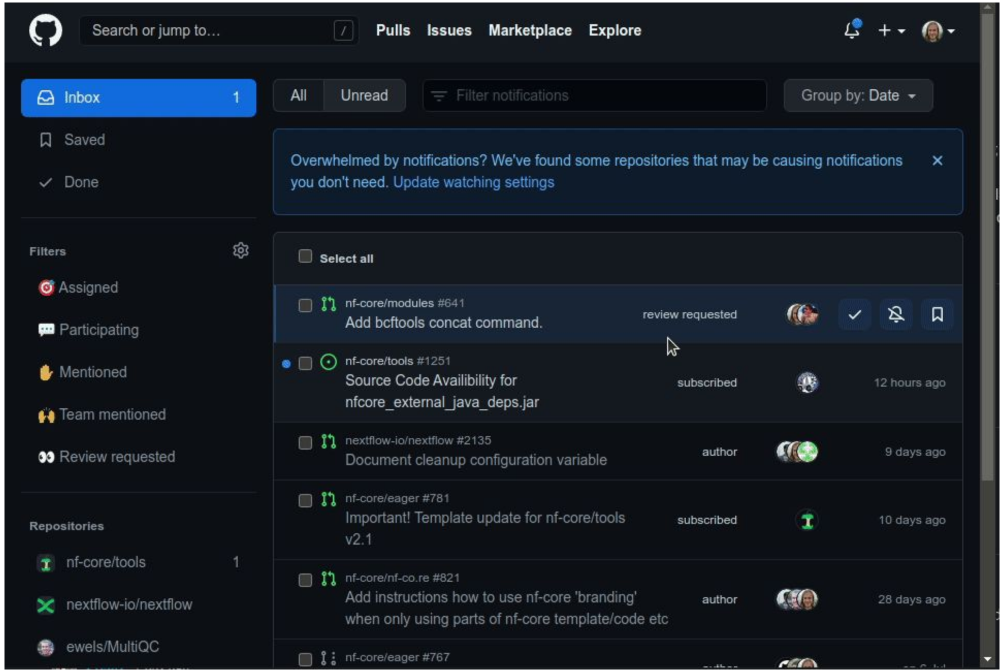
update: BWA/ALN #653

Open jfy133 wants to merge 15 commits into nf-core:master from jfy133:bwa-aln-update

jfy133 commented 22 hours ago • edited

Reviewing PRs

Pull request tab ➡️ Blue Button ➡️ Review Changes



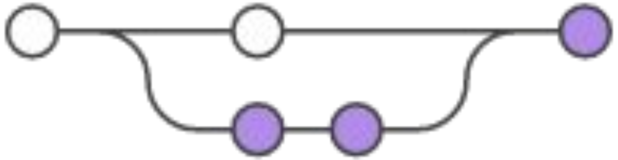
Overwhelmed by notifications? We've found some repositories that may be causing notifications you don't need. Update watching settings

nf-core/modules #641 Add bcftools concat command.	review requested	
nf-core/tools #1251 Source Code Availibility for nfcore_external_java_deps.jar	subscribed	12 hours ago
nextflow-io/nextflow #2135 Document cleanup configuration variable	author	9 days ago
nf-core/eager #781 Important! Template update for nf-core/tools v2.1	subscribed	10 days ago
nf-core/nf-co.re #821 Add instructions how to use nf-core 'branding' when only using parts of nf-core template/code etc	author	28 days ago
nf-core/eager #767		

Merge

Approve and Merge

Once collaborator approves changes, you can merge your branch into the main



Summary and Practical Task: Branch and Fork

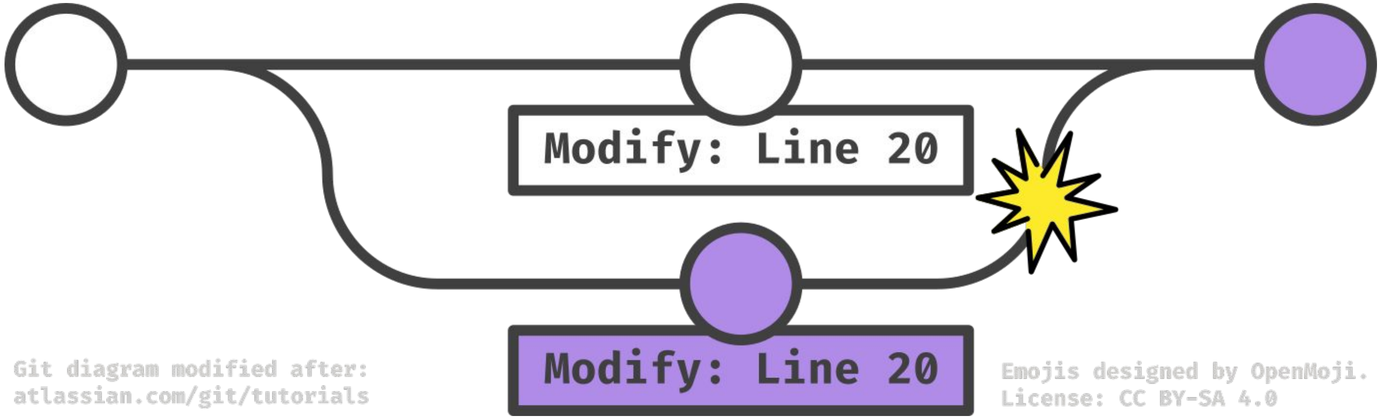
1. Fork `github.com/meganemichel/vigilant-octo-journey`
2. git clone
3. Make new branch with git checkout -c
4. Update README
5. Push changes to your fork
6. Make a PR back into `meganemichel/vigilant-octo-journey`

Merge Conflicts

And how to resolve them

Conflict!

But what if same file or lines of file been modified in both `main` and your branch!?



Git diagram modified after: atlassian.com/git/tutorials

Emojis designed by OpenMoji. License: CC BY-SA 4.0

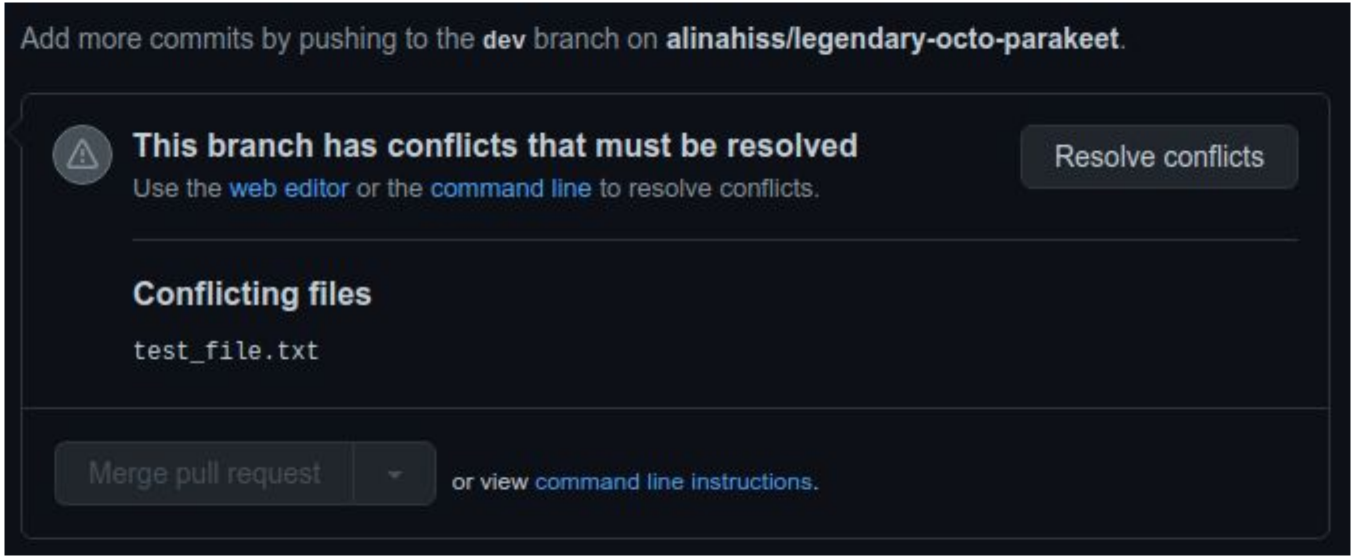
Merge Conflict

When `git` doesn't know which is the 'correct' line or file

- two people have changed the same lines in a file
- one developer deleted a file while another developer was modifying it

How to know when you have a conflict

On GitHub:


Add more commits by pushing to the **dev** branch on **alinahiss/legendary-octo-parakeet**.

This branch has conflicts that must be resolved

Use the web editor or the command line to resolve conflicts.

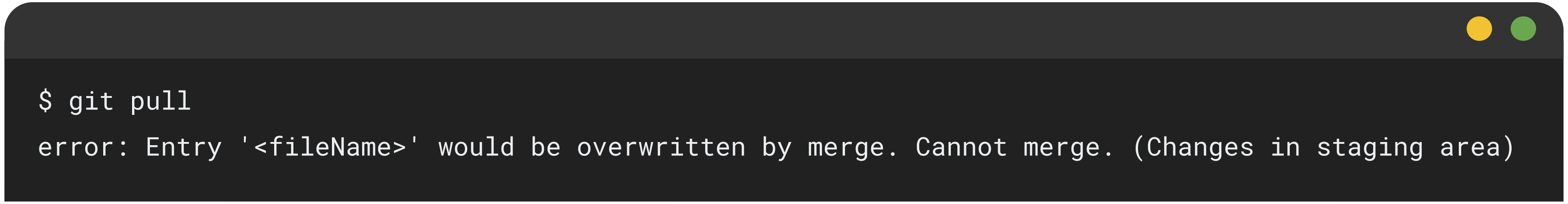
Conflicting files

`test_file.txt`

or view command line instructions.

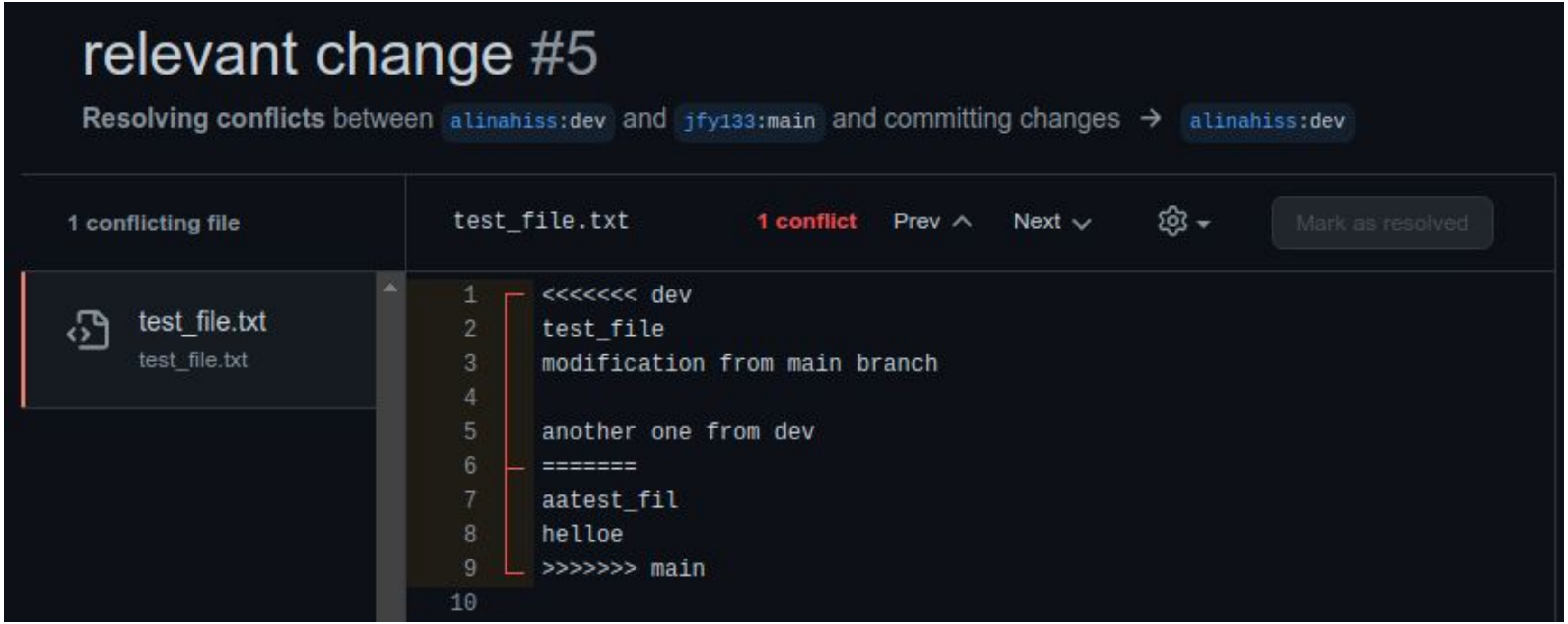
How to know when you have a conflict

On the command line



```
$ git pull
error: Entry '<fileName>' would be overwritten by merge. Cannot merge. (Changes in staging area)
```

Resolving a simple conflict (on github)



relevant change #5

Resolving conflicts between `alinahiss:dev` and `jfy133:main` and committing changes → `alinahiss:dev`

1 conflicting file

test_file.txt
test_file.txt

test_file.txt — **1 conflict** — Prev — Next — Mark as resolved

```
<<<<<<< dev
test_file
modification from main branch

another one from dev
=======
aatest_fil
helloe
>>>>>>> main
```

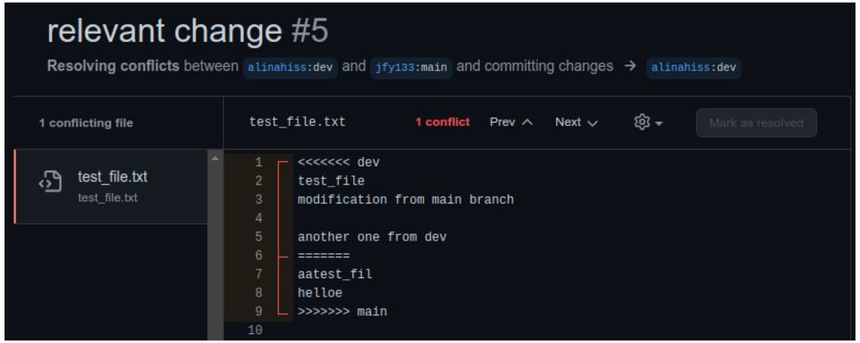
e.g. we both modified line 1. Is `aatest_fil` or `test_file` correct?

What a merge conflict looks like

- Incoming change (the later change - e.g. from your remote)
    - **`<<<<<<< dev`**
- Separator
    - **`=======`**
- Current change (what you currently have on your local)
    - **`>>>>>>> main`**

Resolving a simple conflict

Delete which you think is 'right'!



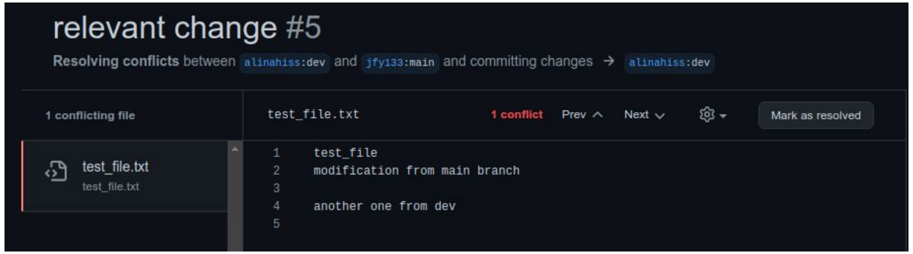
relevant change #5

Resolving conflicts between alinahiss:dev and jfy133:main and committing changes → alinahiss:dev

```
<<<<<<< dev
test_file
modification from main branch

another one from dev
=======
aatest_fil
helloe
>>>>>>> main

```



relevant change #5

Resolving conflicts between alinahiss:dev and jfy133:main and committing changes → alinahiss:dev

```
test_file
modification from main branch

another one from dev

```

⚠️ make sure to delete all conflict markers!

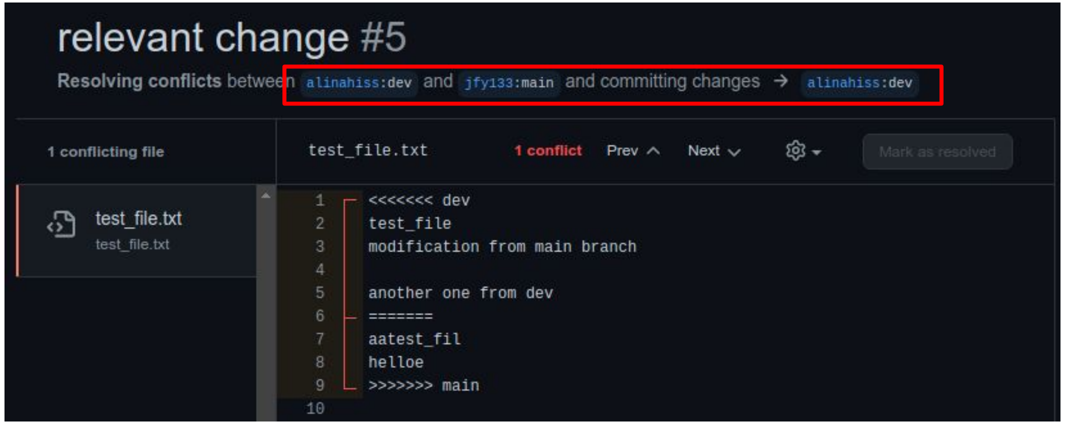
Resolving a simple conflict

- Press ‘Mark as resolved’
- Updates if PR `dev` ➡️ `main`, changes merged into `dev`
- Now can merge PR 😃🎉

Simple conflict

⚠️ check WHERE resolved changes being committed to!

- Resolving conflict when updating `dev` from `main`
- Commit will be onto `main` ⬅️ NOT recommended!
- If unsure: make a separate branch (GitHub will ask)



relevant change #5

Resolving conflicts between `alinahiss:dev` and `jfy133:main` and committing changes → `alinahiss:dev`

1 conflicting file

test_file.txt
test_file.txt

test_file.txt 1 conflict Prev Next Mark as resolved

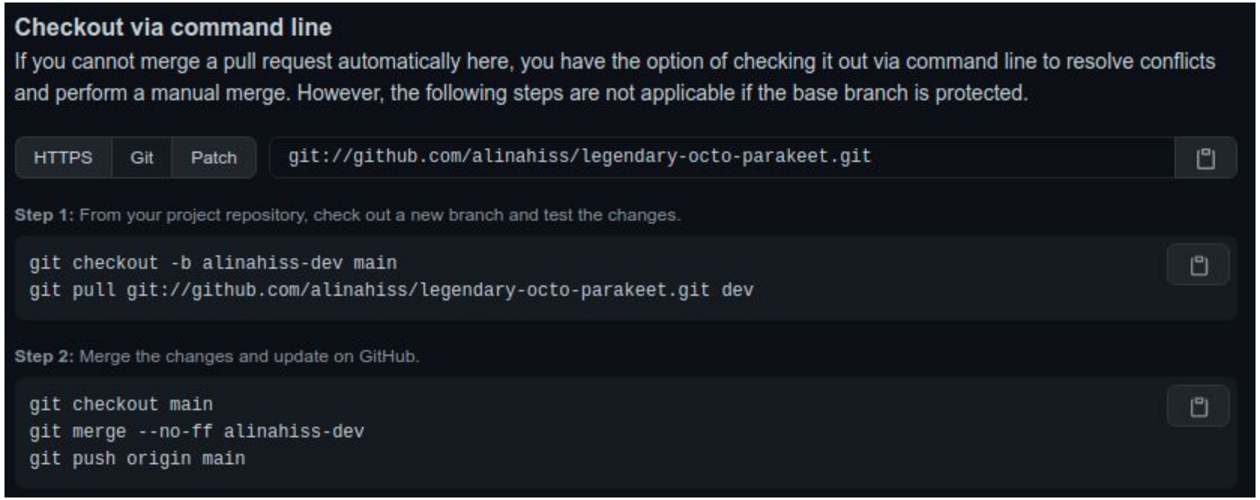
```
<<<<<<< dev
test_file
modification from main branch

another one from dev
=======
aatest_fil
helloe
>>>>>>> main

```

Where to 'resolve a conflict'

- Simple: on GitHub (e.g. couple lines)
- Complex: locally on your machine with your IDE



Checkout via command line

If you cannot merge a pull request automatically here, you have the option of checking it out via command line to resolve conflicts and perform a manual merge. However, the following steps are not applicable if the base branch is protected.

```
git://github.com/alinahiss/legendary-octo-parakeet.git
```

Step 1: From your project repository, check out a new branch and test the changes.

```
git checkout -b alinahiss-dev main
git pull git://github.com/alinahiss/legendary-octo-parakeet.git dev
```

Step 2: Merge the changes and update on GitHub.

```
git checkout main
git merge --no-ff alinahiss-dev
git push origin main
```

Tricky: out of scope for this workshop

Summary: Merge conflicts

- Merge conflict: same line modified on two branches/forks
- Indicated with
  - `>>> <BRANCH>`
  - `===`
  - `<<< <BRANCH>`
- When conflict is:
  - Simple: on GitHub
  - Complex: locally (follow instructions)

Merging

If changed files on `main` not changed on your branch, simply

```
$ git fetch upstream
$ # or git pull
```