

Presentation by Daniel Perez

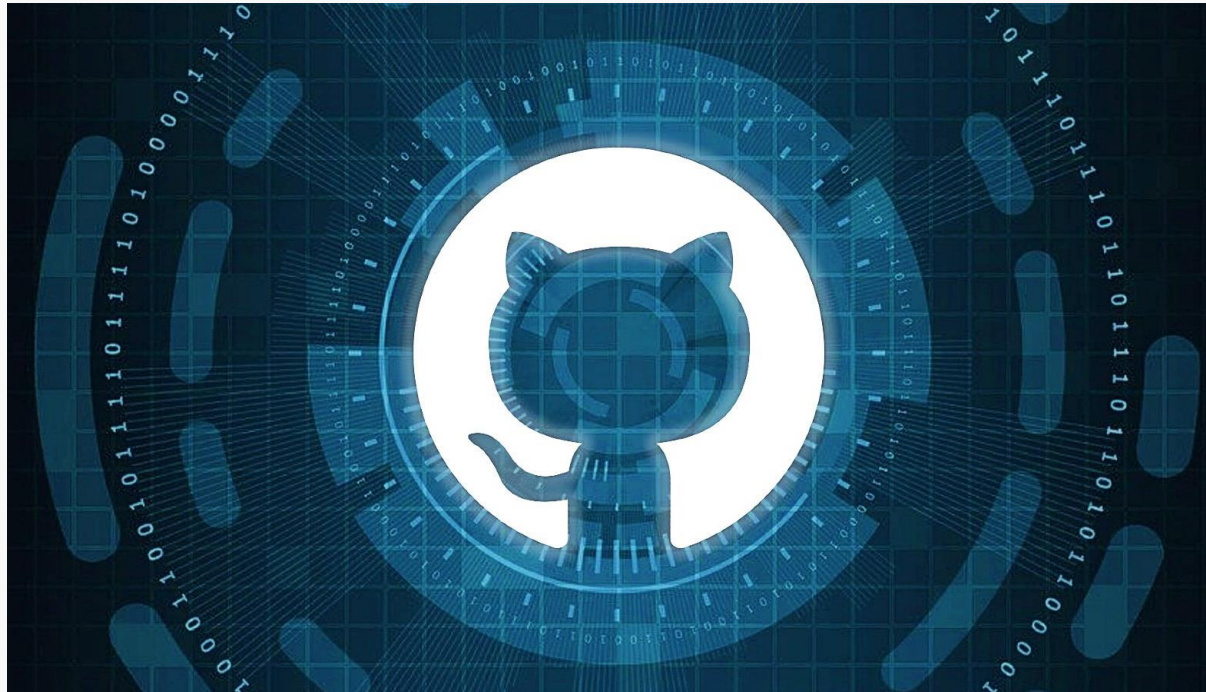
Graduate Fellow

Murty Sunak Quantitative Computing Lab

GitHub Workflow Basics

Requirements

- **GitHub desktop installed**
- **Account and Logged into GitHub**
- **Create a working directory in your computer**





- Learn basic concepts of version control systems
- Learn how to setup a git repository and organization
- Learn how to setup a project and prepare to contribute to it by cloning and branching
- Learn how to create and assign issues
- Learn about pull requests and code review



Version Control System:

Software tool that helps in recording changes made to files by keeping a track of modifications done in the code.

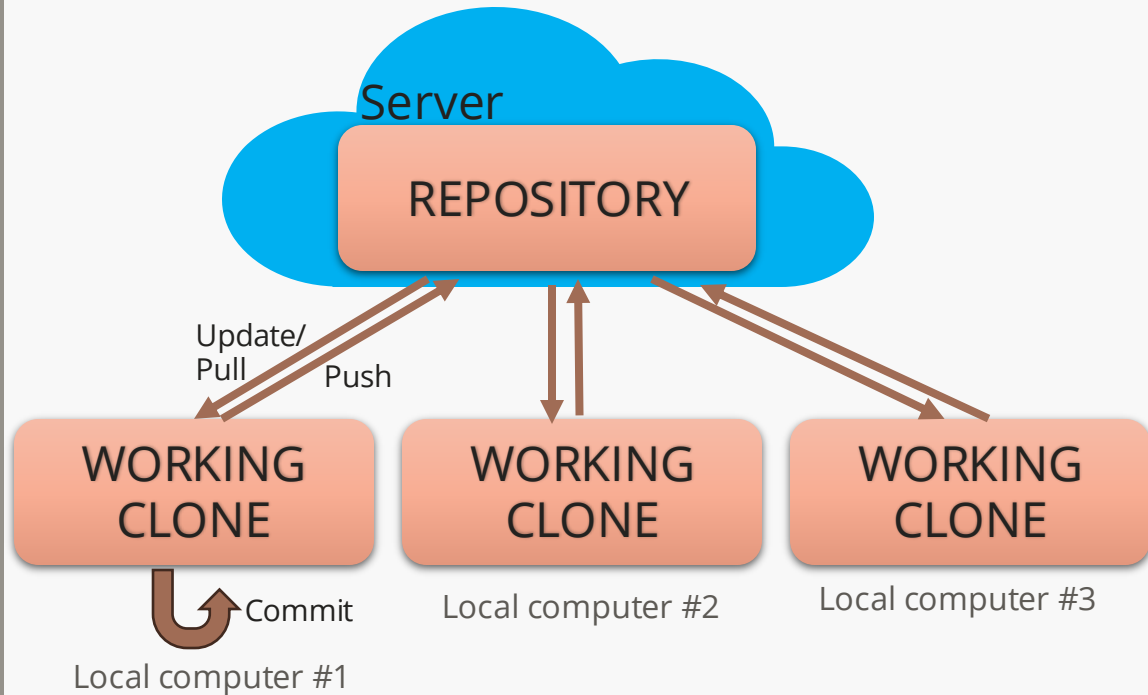
Why do we use a Version Control System (VCS)?

- Keep a complete change history of every file; the change history contains essential information such as which change was made, who made the change, when the change happened, and why the change was made.
- Let people work together independently at the same time
- Make project management easier (traceability, reversibility, backup)

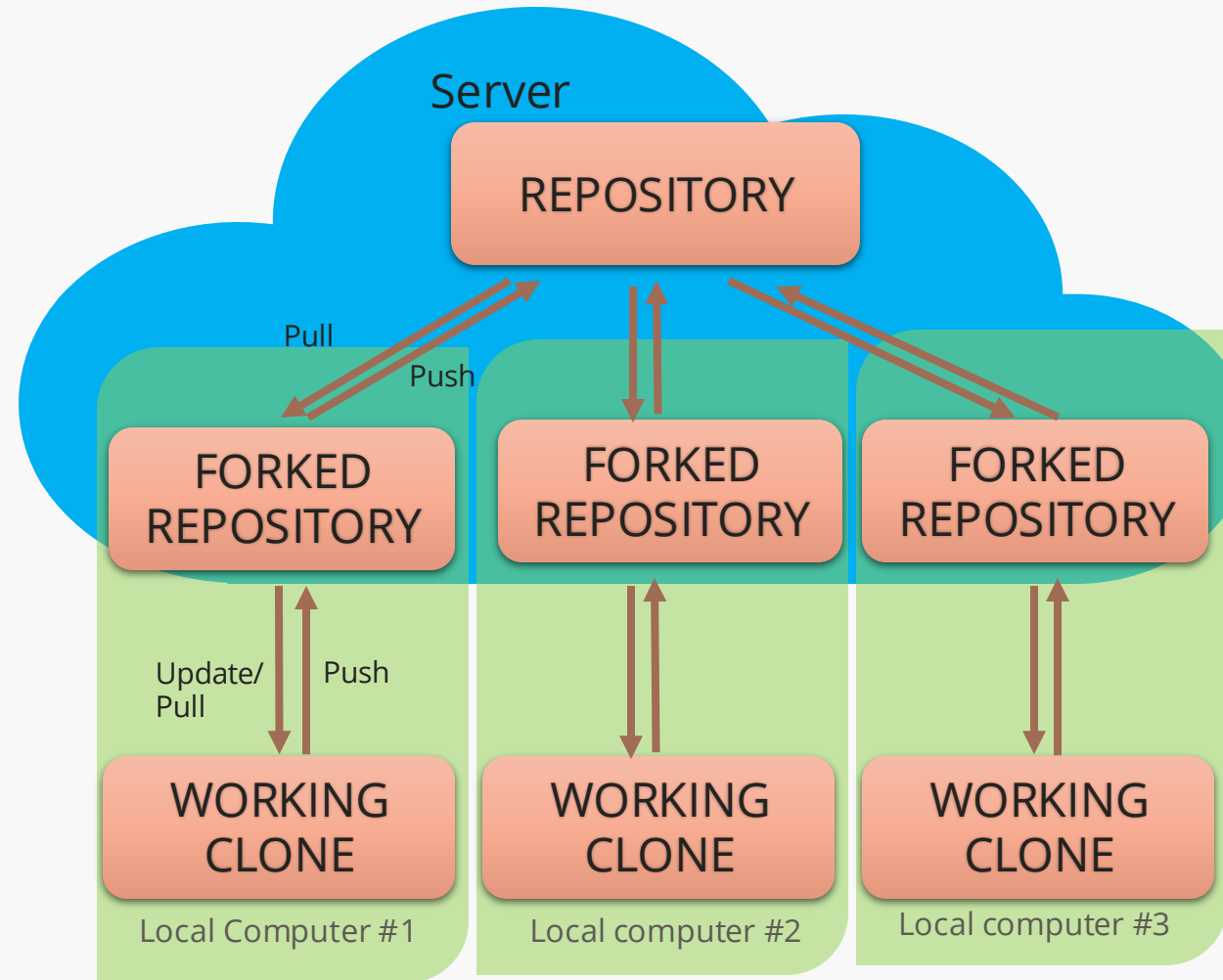
Types of VCS

- Local Version Control Systems
- Centralized Version Control Systems
- Distributed Version Control Systems

Centralized VCS



Distributed VCS



What is GitHub?

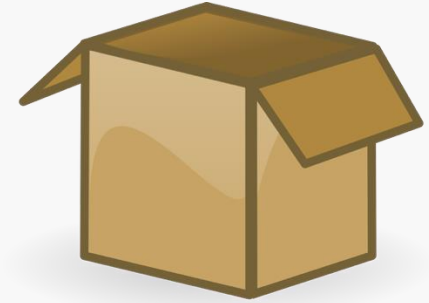
GitHub is a web-based platform used for version control, collaboration, and code repository management. It allows developers to work together on projects, track changes, merge code, and manage issues.



Key Components

Repositories

Repositories in GitHub are where your project's files and revision history are stored. They can be public or private, and each repository contains documentation, code, and related resources.



Issues

Issues in GitHub are used to track tasks, enhancements, and bugs for your projects. They facilitate communication among team members, assign tasks, and prioritize work.



Roles: Maintainer/Contributor

The Maintainer as the person who is owner and in charge of the main repository, manage a project's direction and improvement

The Contributor as the person who is accessing or helping modify the repository

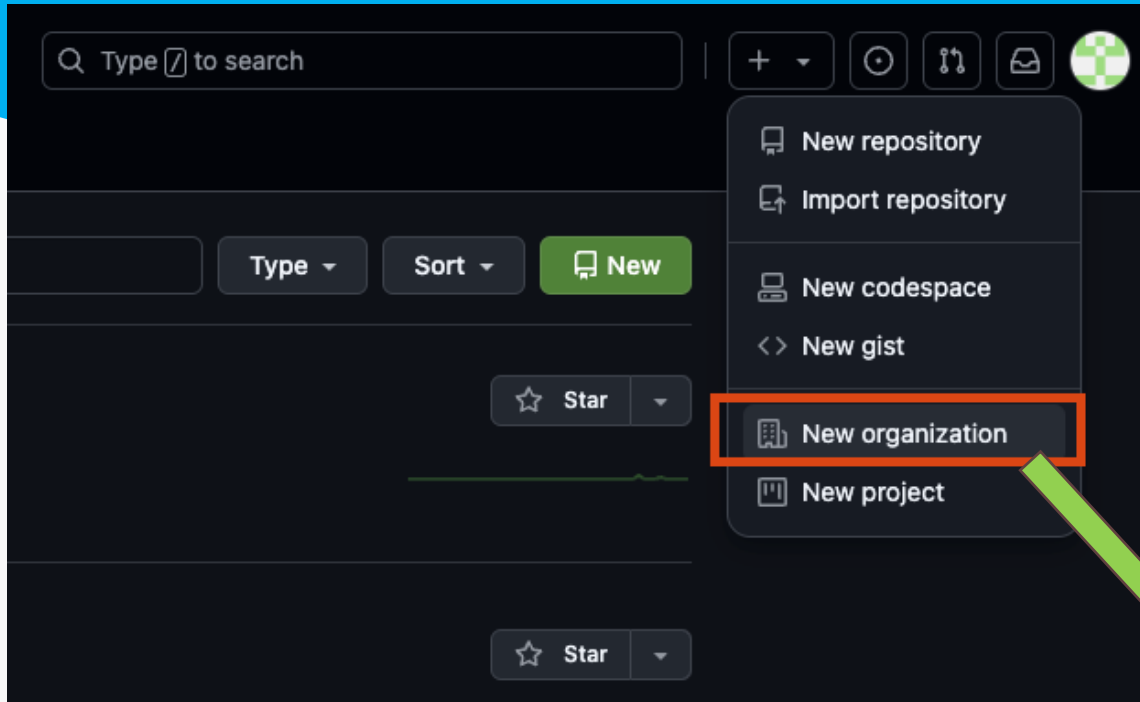
Roles: Read, Triage, Write, Admin

Prepare the project <Maintainer>



Creates an organization and a repository

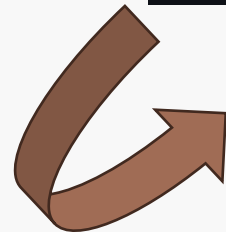
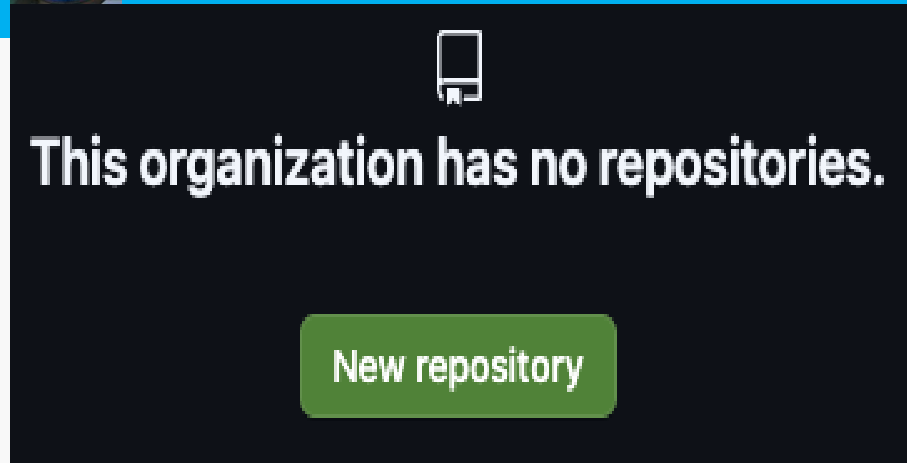
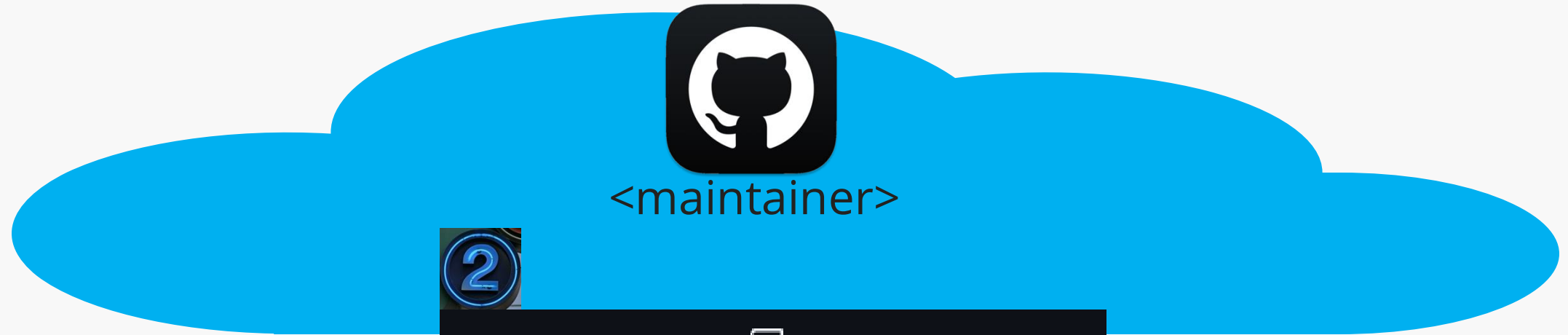
1



Tell us about your organization

Set up your organization

Prepare the project

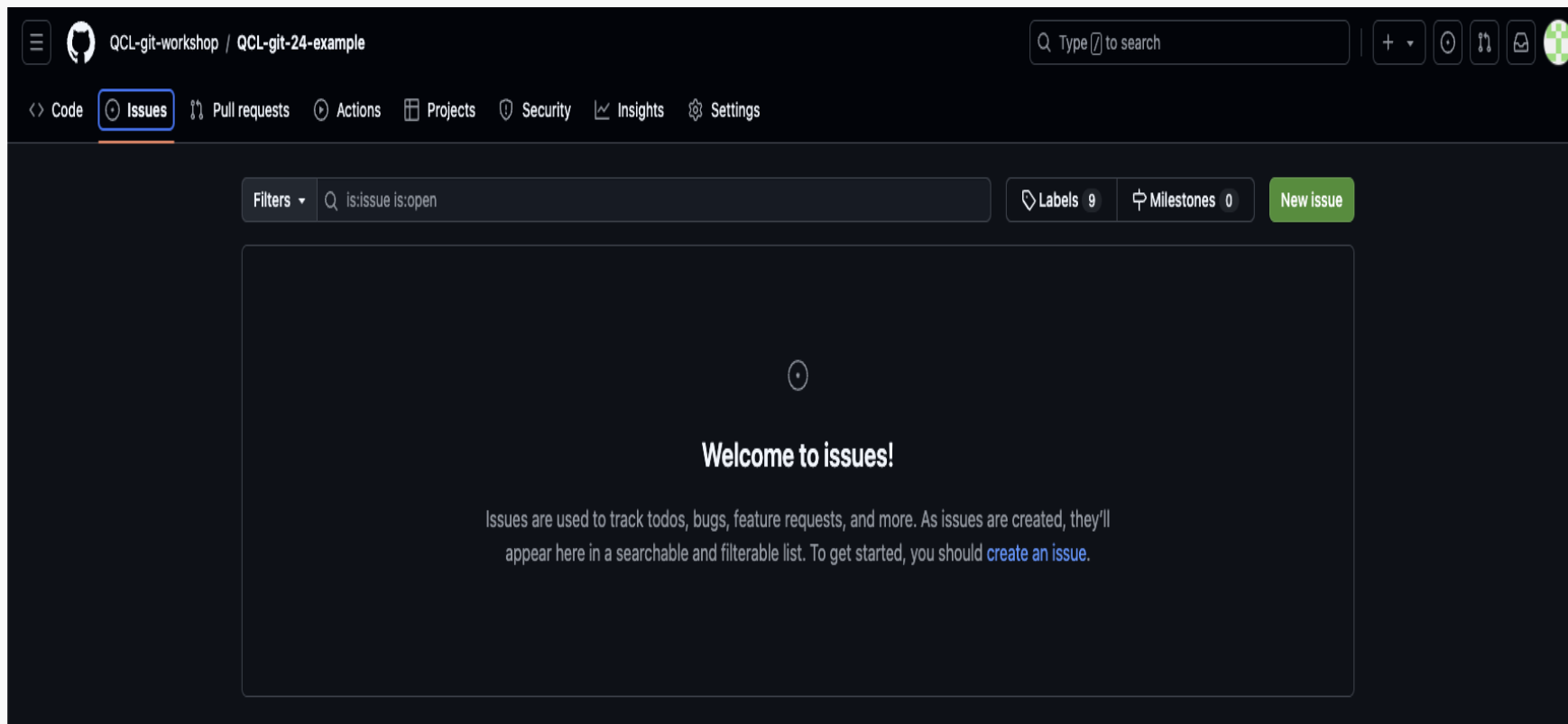


Update the readme file

Issues and collaborations <Maintainer>

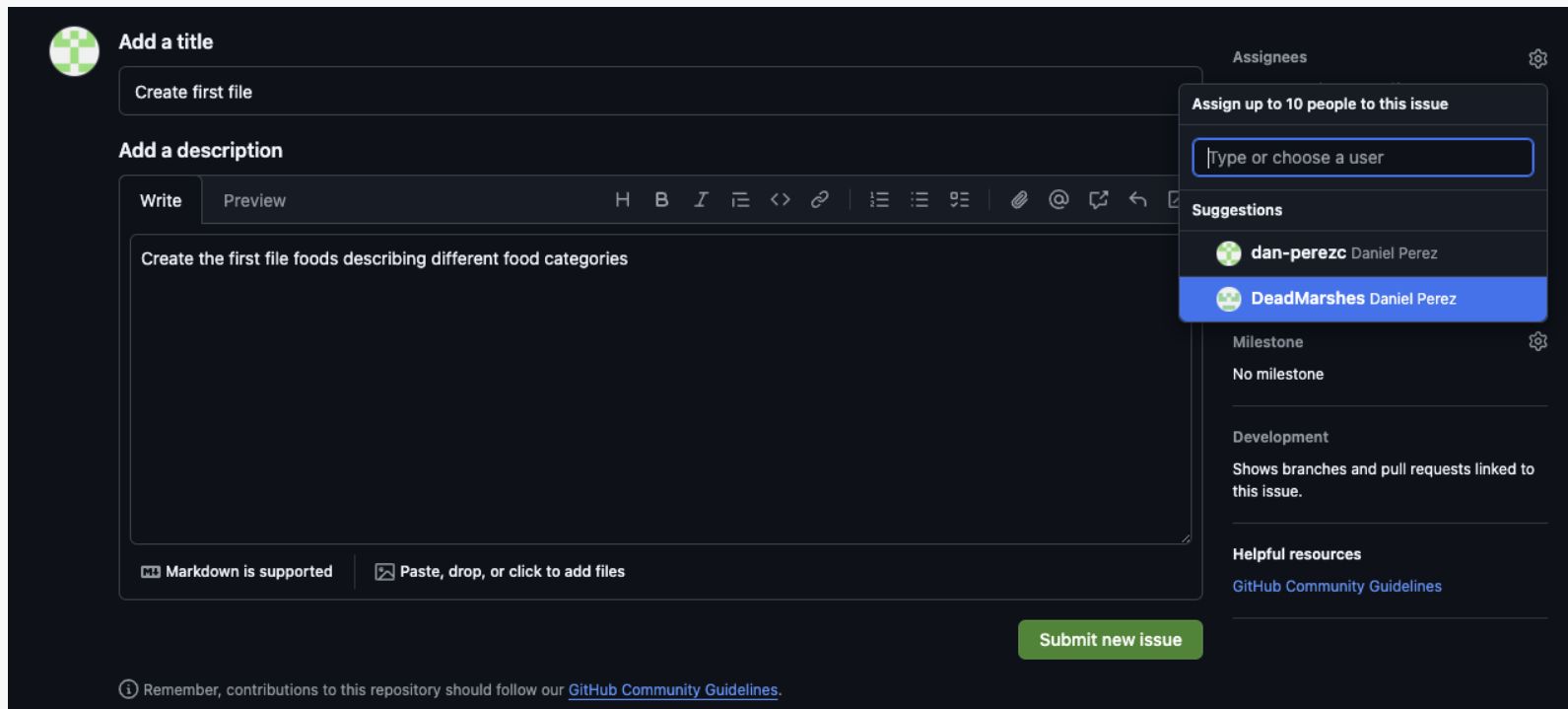
Creating an issue and then claiming it

Helps prevent developers from working on the same thing at the same time. Also, issue provides a place for the community to propose ideas, prioritize issues, size issues, clarify requirements, and verify bugs



Issues and collaborations

Create and assign an issue to your Contributor



Add a title

Create first file

Add a description

Write Preview

Create the first file foods describing different food categories

Markdown is supported | Paste, drop, or click to add files

Assignees

Assign up to 10 people to this issue

Type or choose a user

Suggestions

- dan-perezc Daniel Perez
- DeadMarshes Daniel Perez**

Milestone

No milestone

Development

Shows branches and pull requests linked to this issue.

Helpful resources

[GitHub Community Guidelines](#)

Submit new issue

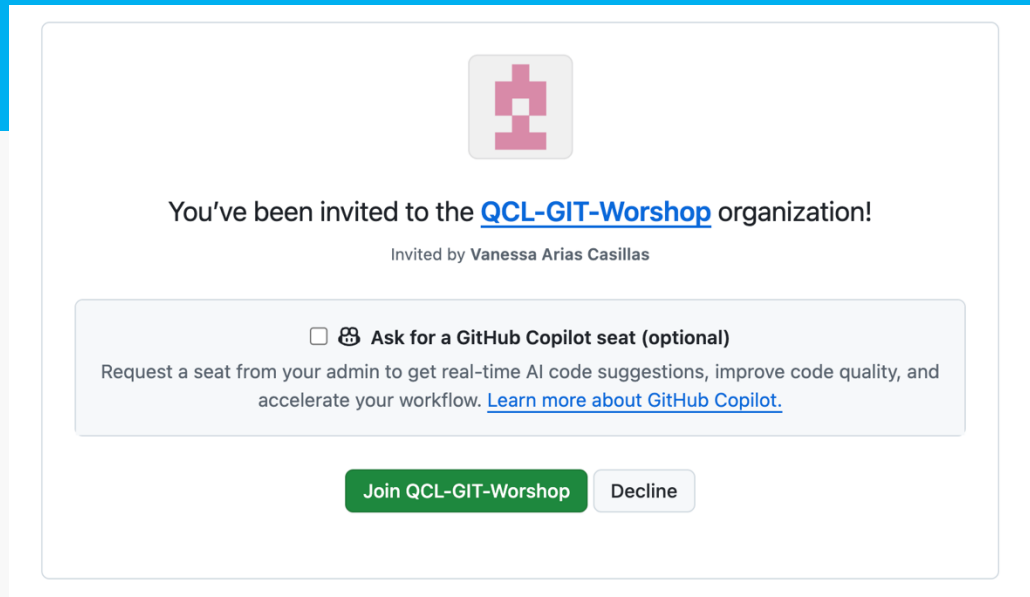
Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Join to the project < Contributor >



Receives invitation to the company or project

Either
Find the invitation in
your email, view the
invitation, and accept
the invitation.
Or
Navigate to the team's
organization, view the
invitation, and accept
the invitation.



Organizations

New organization

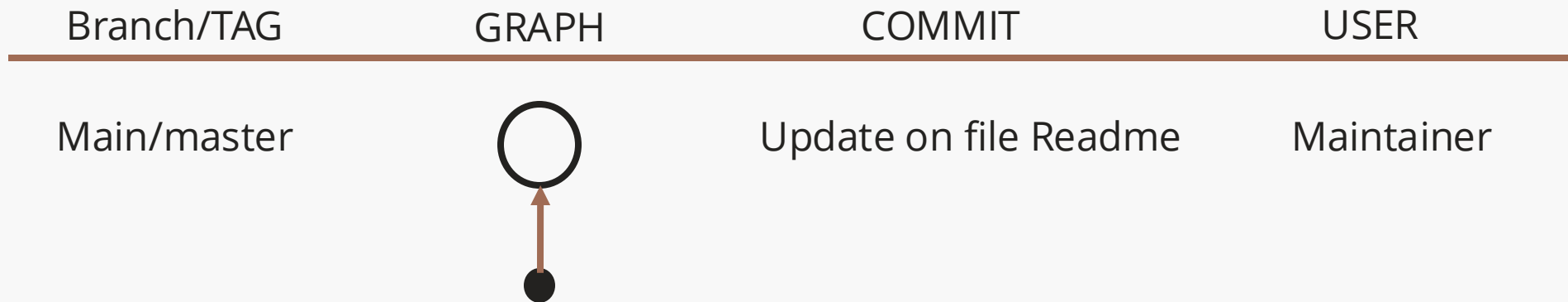
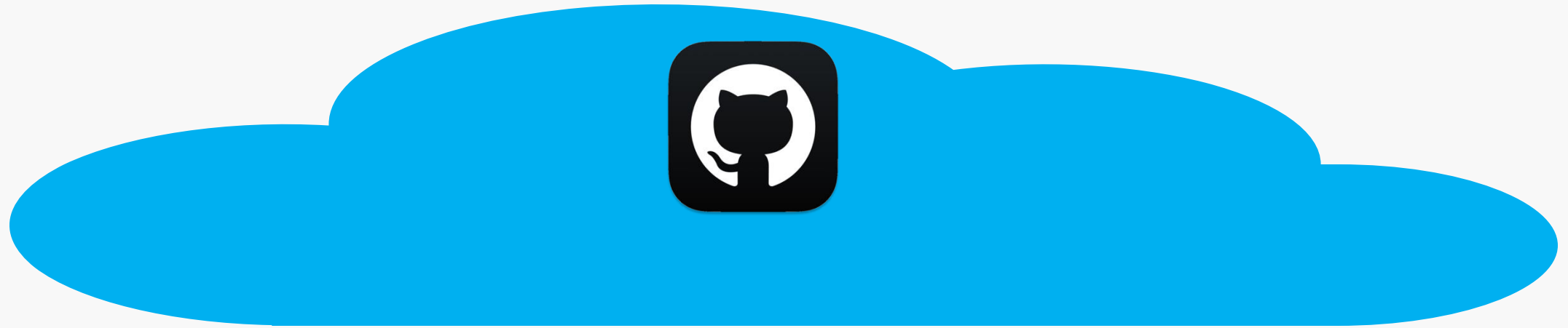
 **QCL-GIT-Workshop** Member

Invitation expires in 7 days

Accept

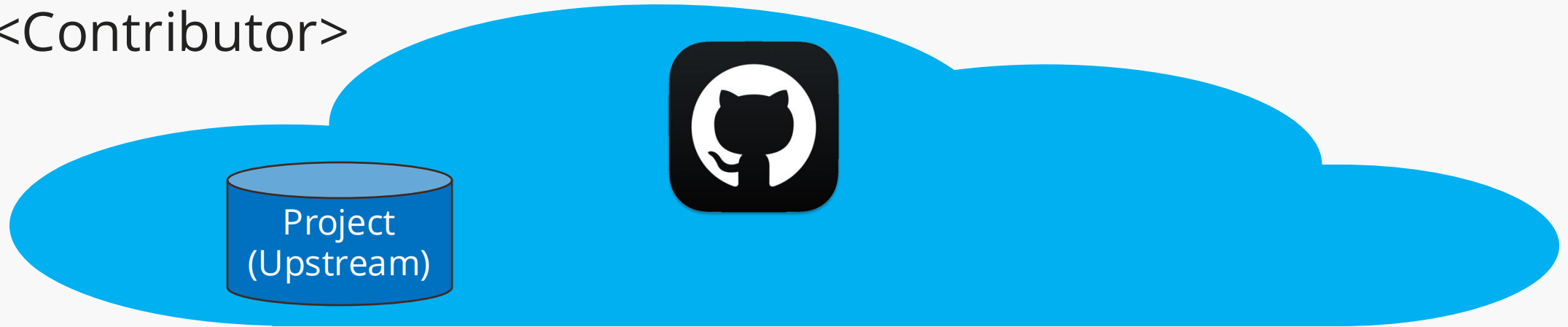
Decline

Project tracker



Prepare to work on a project

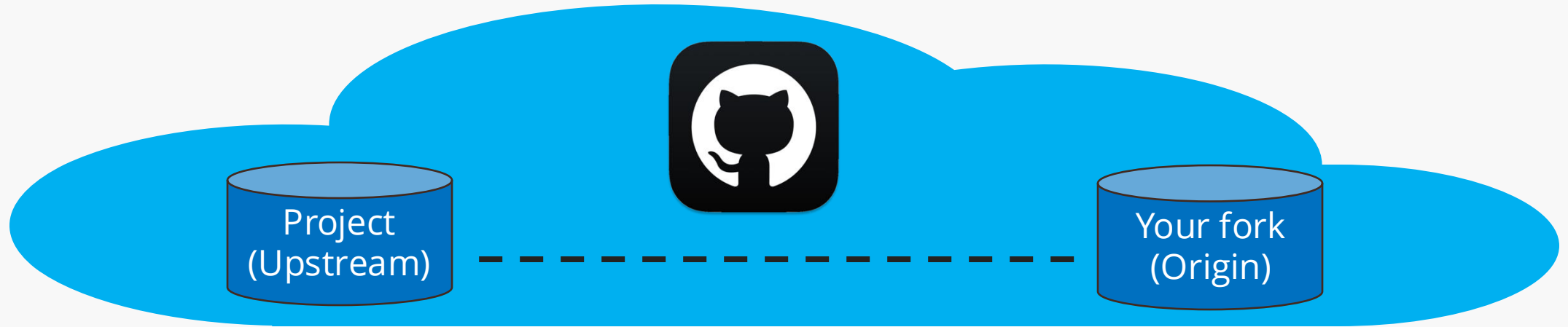
<Contributor>



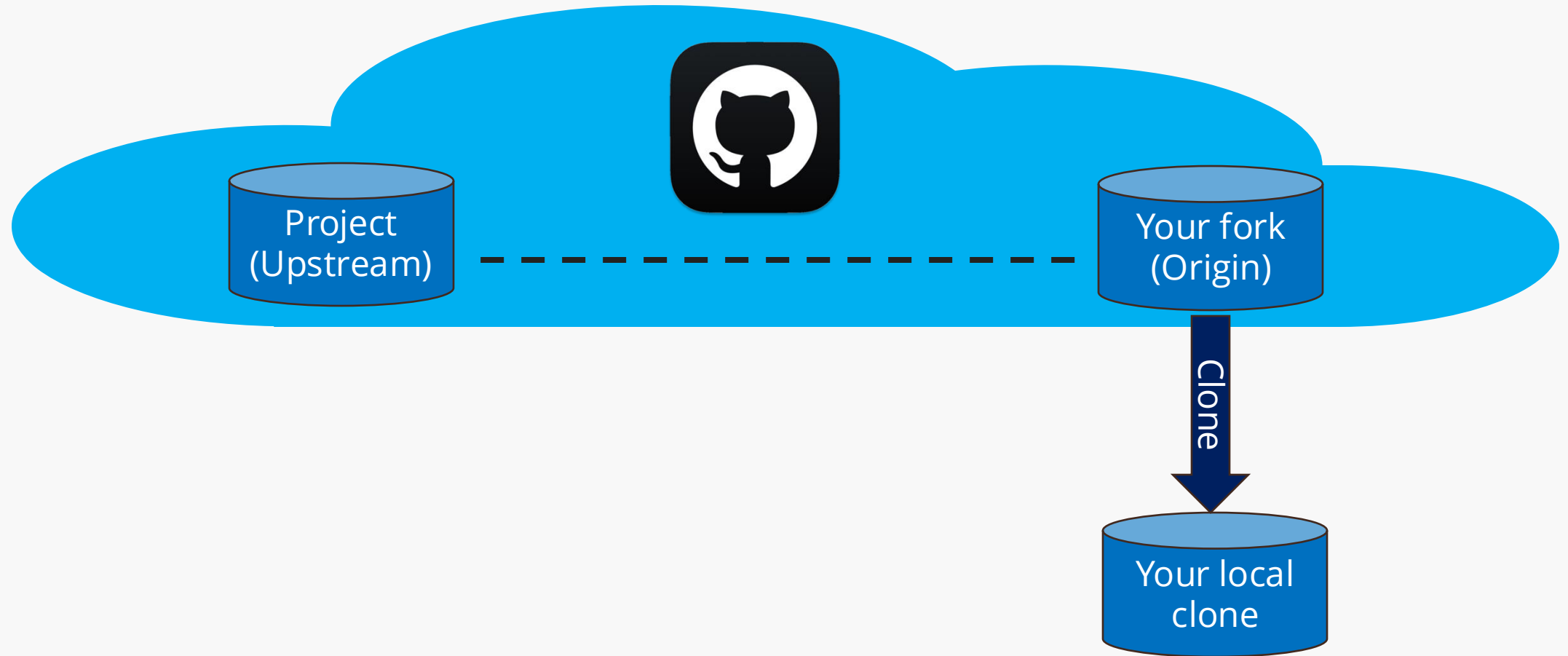
Prepare to work on a project



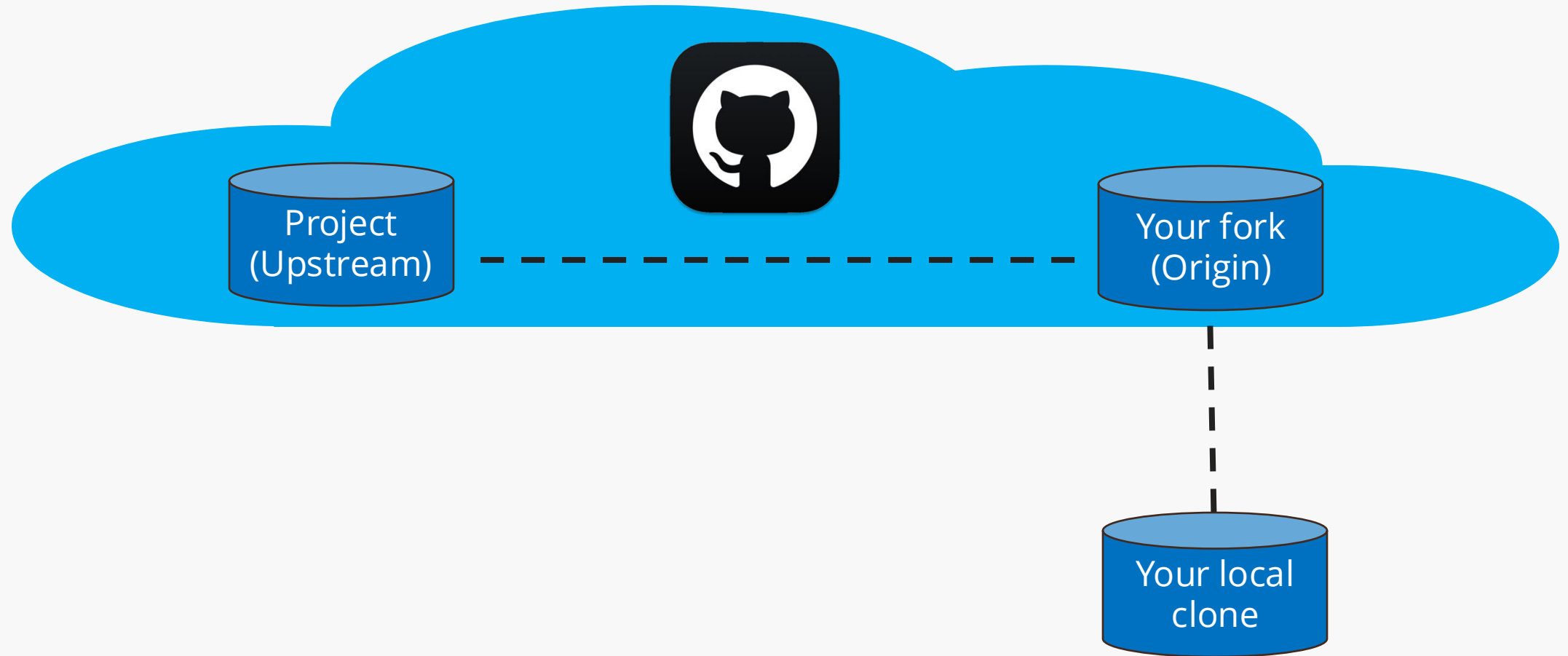
Prepare to work on a project



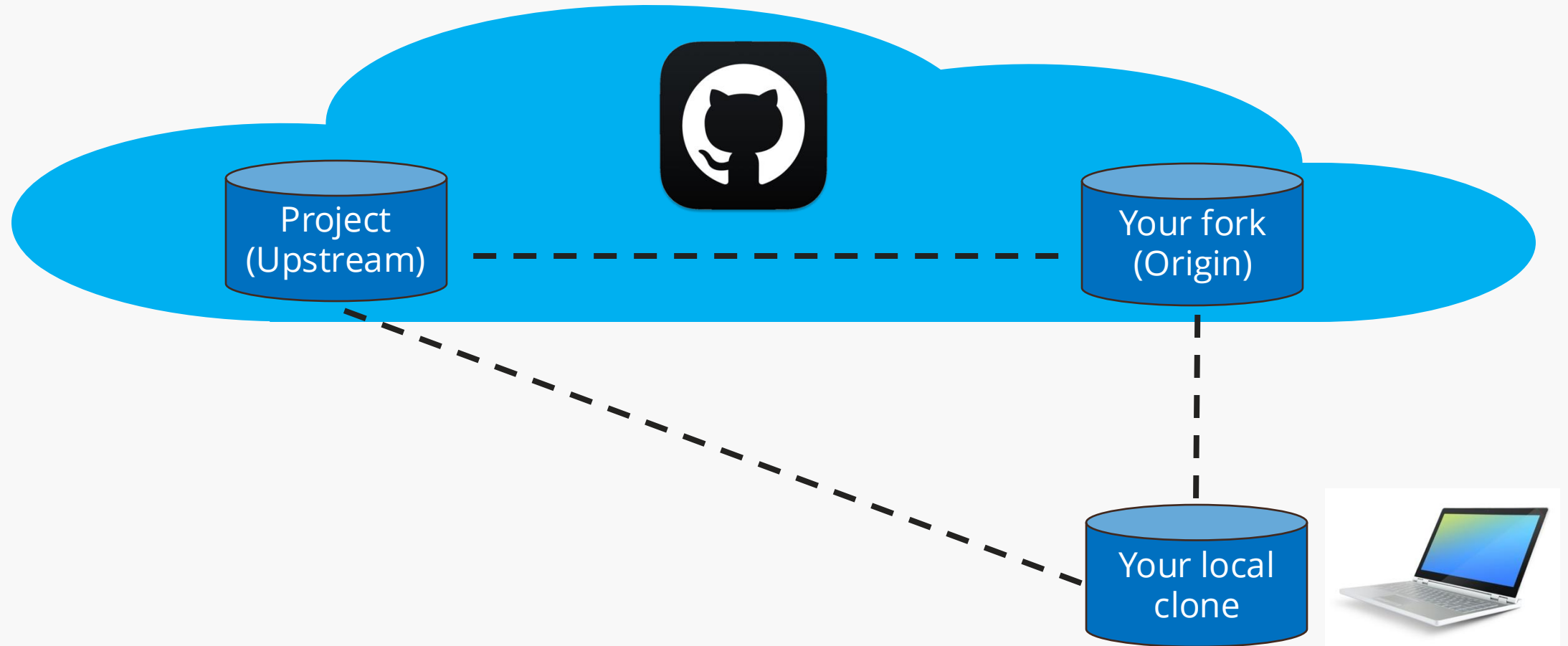
Prepare to work on a project



Prepare to work on a project



Prepare to work on a project



Activity! <Contributor>



Create new fork and clone the repo

The screenshot shows the GitHub interface for the repository 'QCL-git-24-example'. At the top, there's a navigation bar with 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Security', 'Insights', and 'Settings'. Below this, a message states: 'You now have admin access to the QCL-git-workshop/QCL-git-24-example repository.' The repository name 'QCL-git-24-example' is displayed with a 'Public' badge. To the right, there are buttons for 'Watch 0', 'Fork 0', and 'Star 0'. The 'Fork' button is highlighted with a blue border, and its dropdown menu is open, showing 'Existing forks' and the message 'You don't have any forks of this repository.' with a '+ Create a new fork' option. Below the repository name, there's a section for the 'main' branch, showing '1 Branch' and '0 Tags'. A commit by 'dan-perezc' is listed, titled 'Update README.md', with a commit hash 'ca6c087' and a timestamp '31 minutes ago'. The commit details show '2 Commits' and a file 'README.md' with the message 'Update README.md' and a timestamp '31 minutes ago'.

Activity!

<Contributor>



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.

Required fields are marked with an asterisk (*).

Owner *



DeadMarshes ▾



Repository name *

QCL-git-24-example

✔ QCL-git-24-example is available.

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

Description (optional)

Origin

☒ Copy the `main` branch only

Contribute back to QCL-git-workshop/QCL-git-24-example by adding your own branch. [Learn more.](#)

You are creating a fork in your personal account.

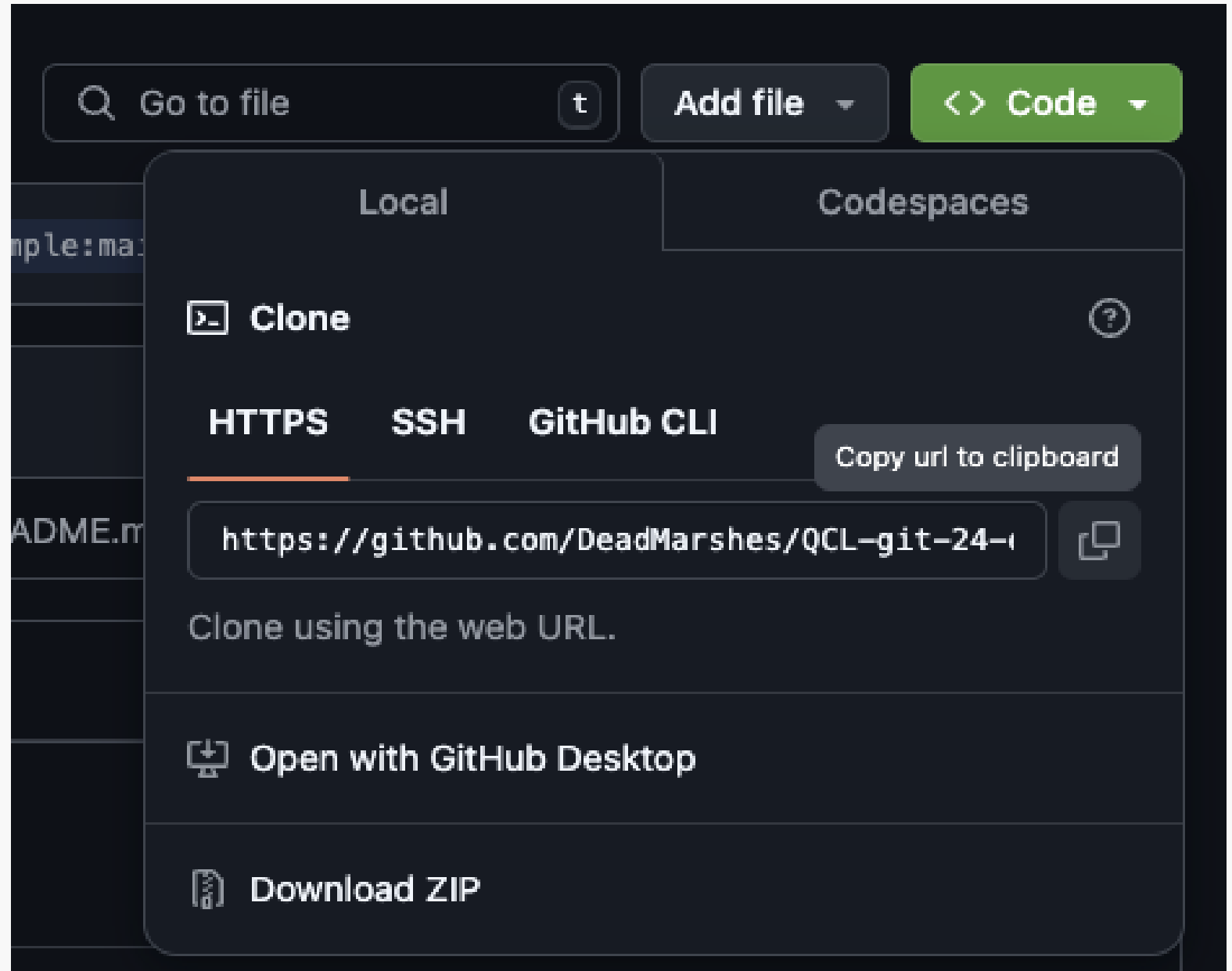
Create fork

Activity!

<Contributor>

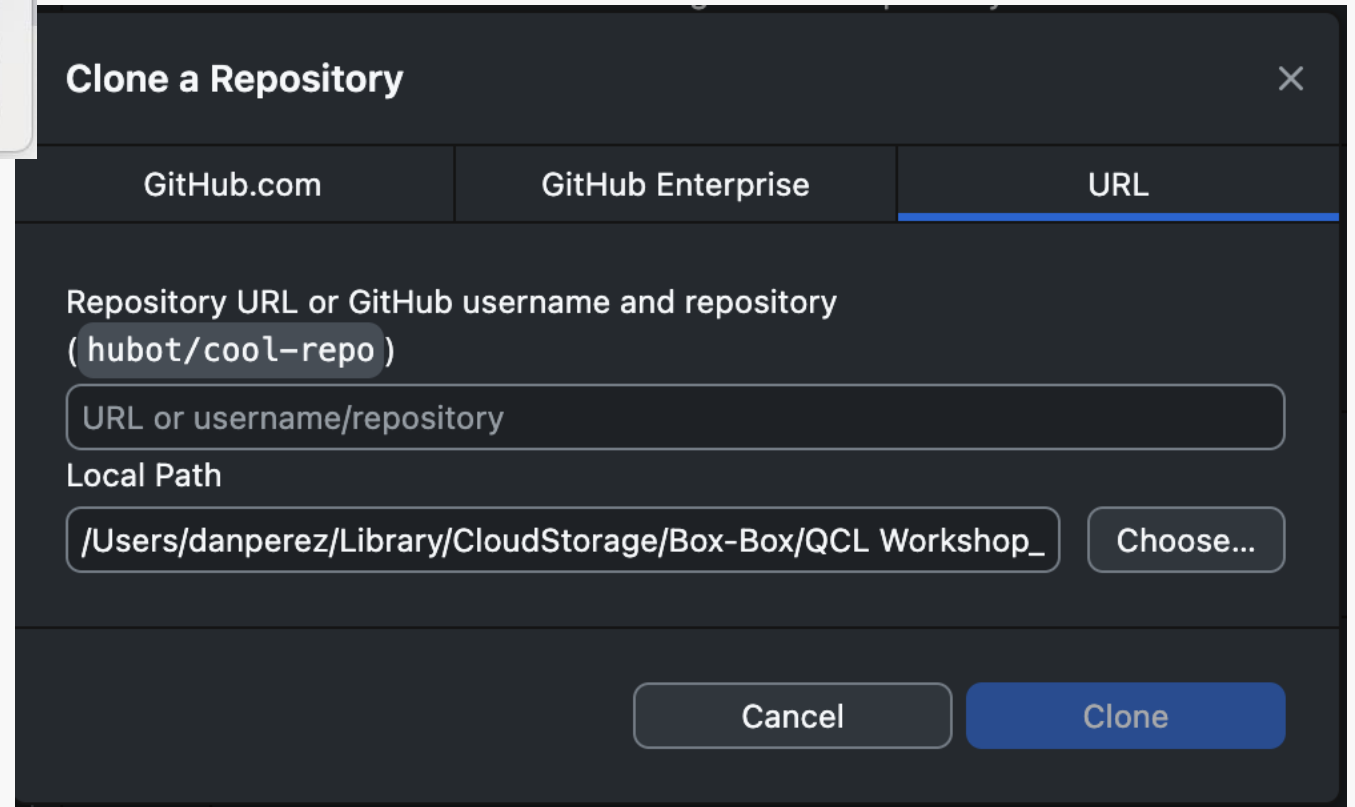
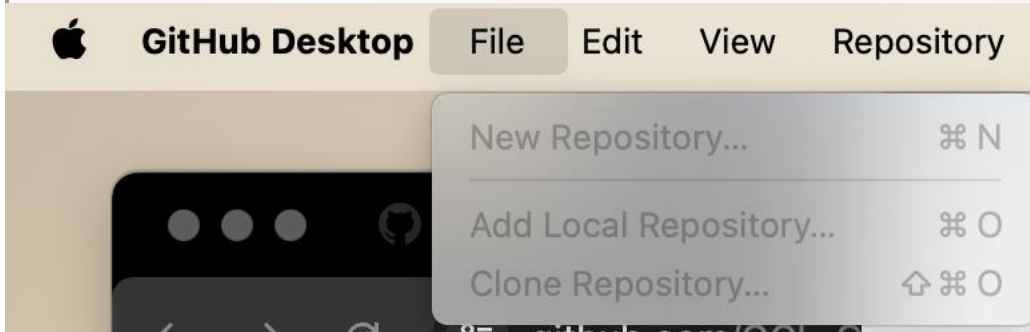


Make sure you are in your fork's page



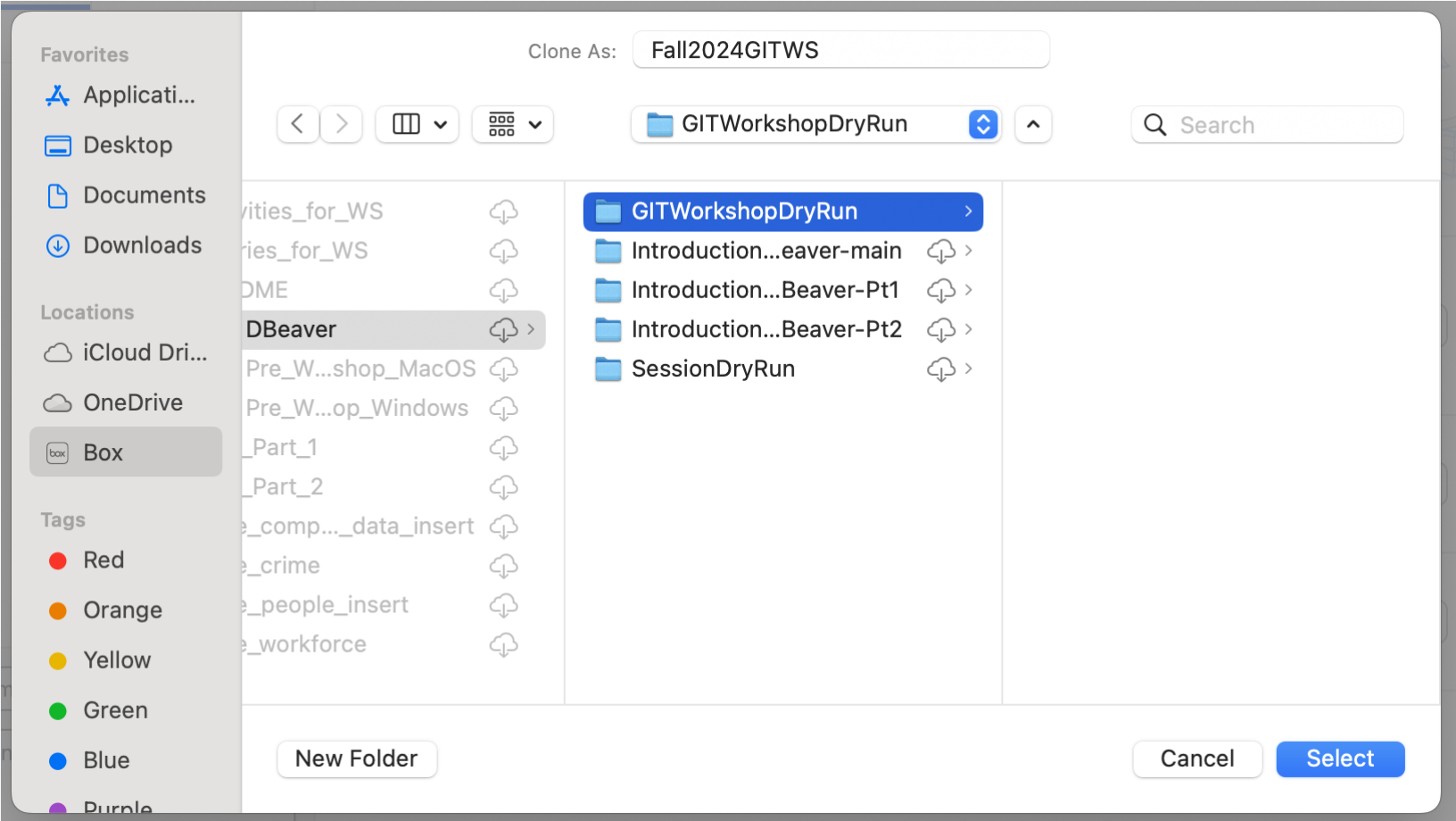
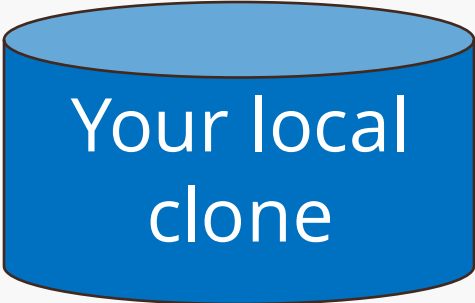
Activity!

Clone your repo



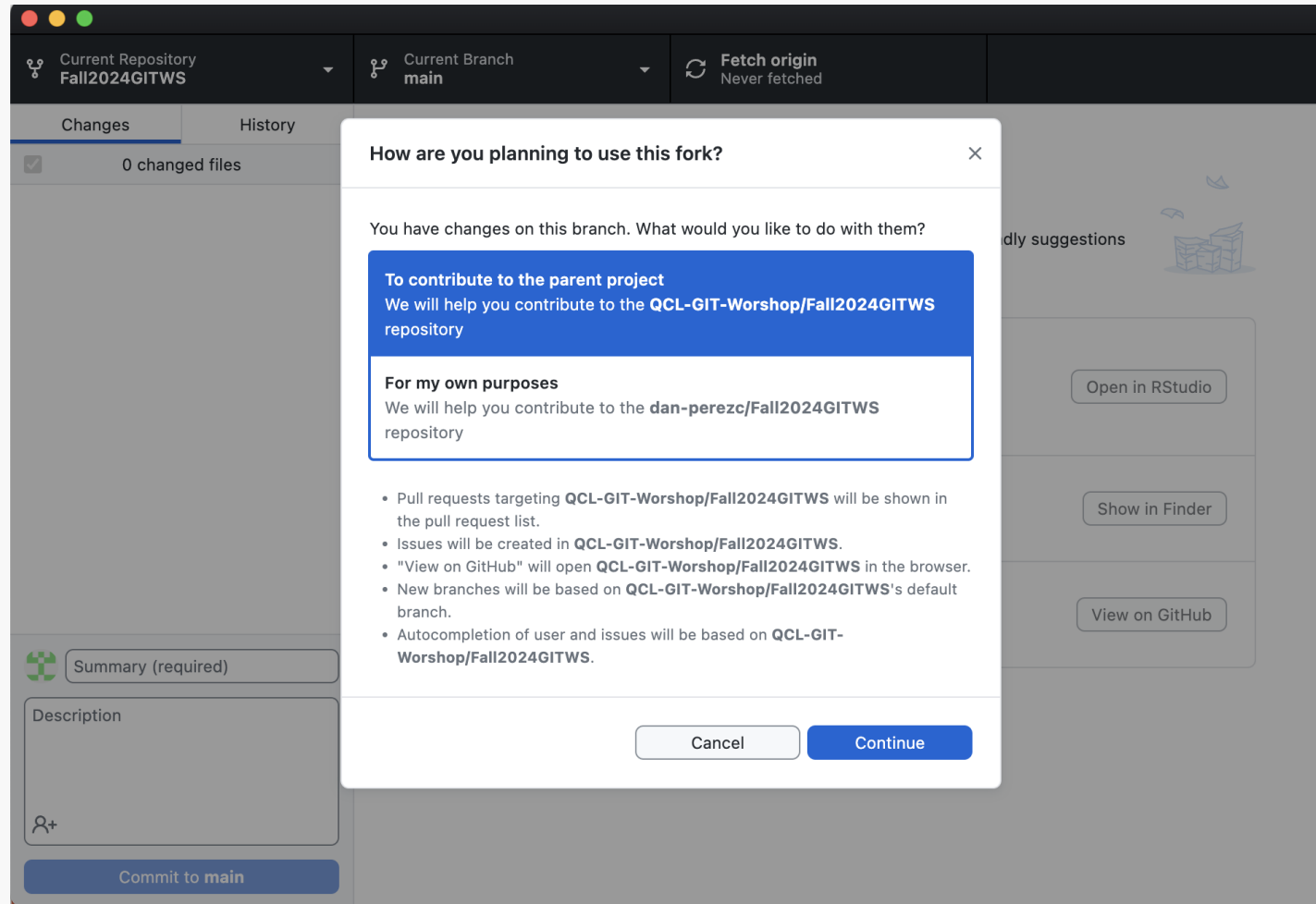
Activity!

Clone your repo



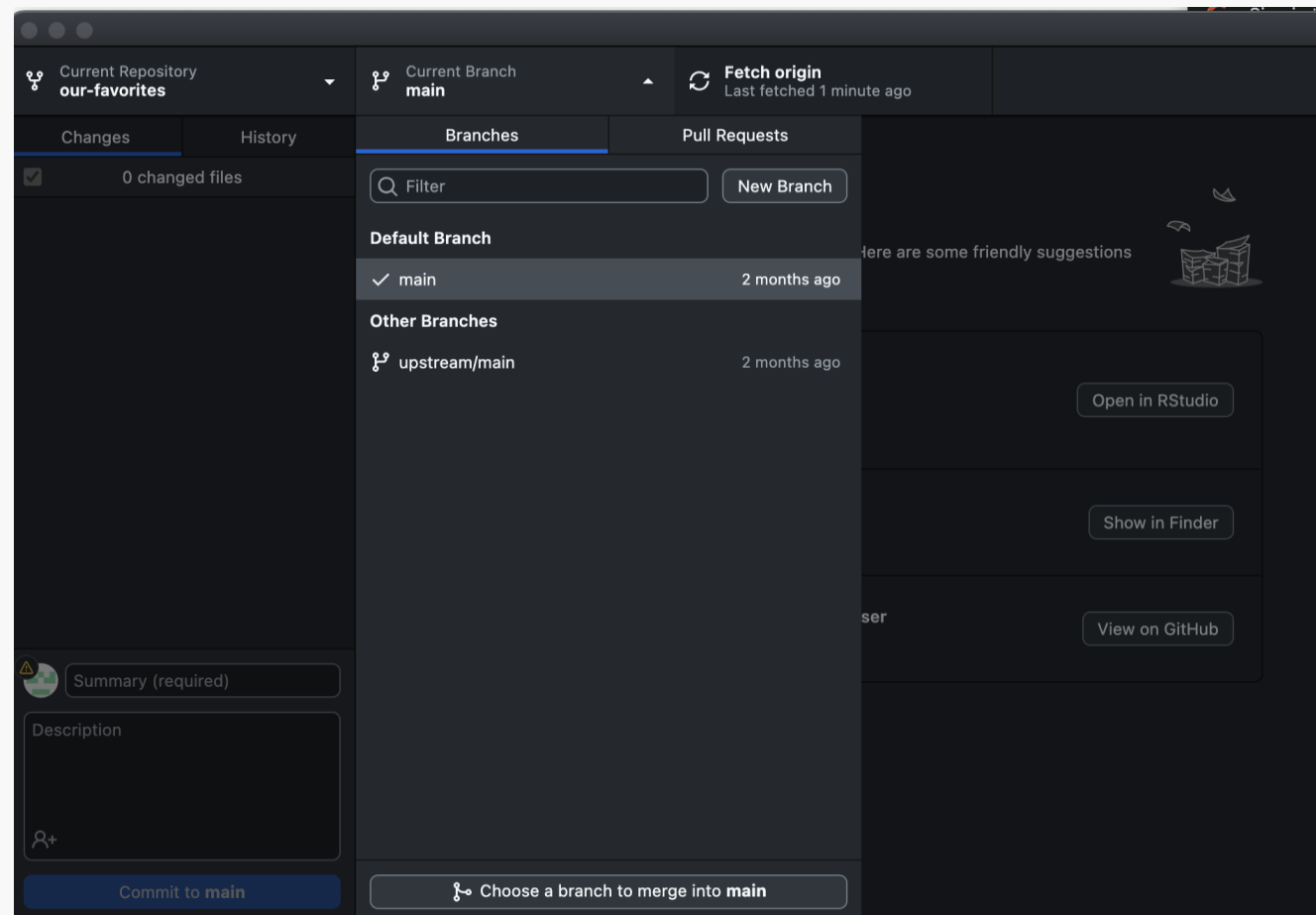
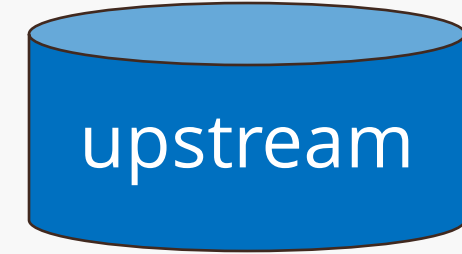
Activity!

Clone your repo



Activity!

Verify Branches, upstream remote

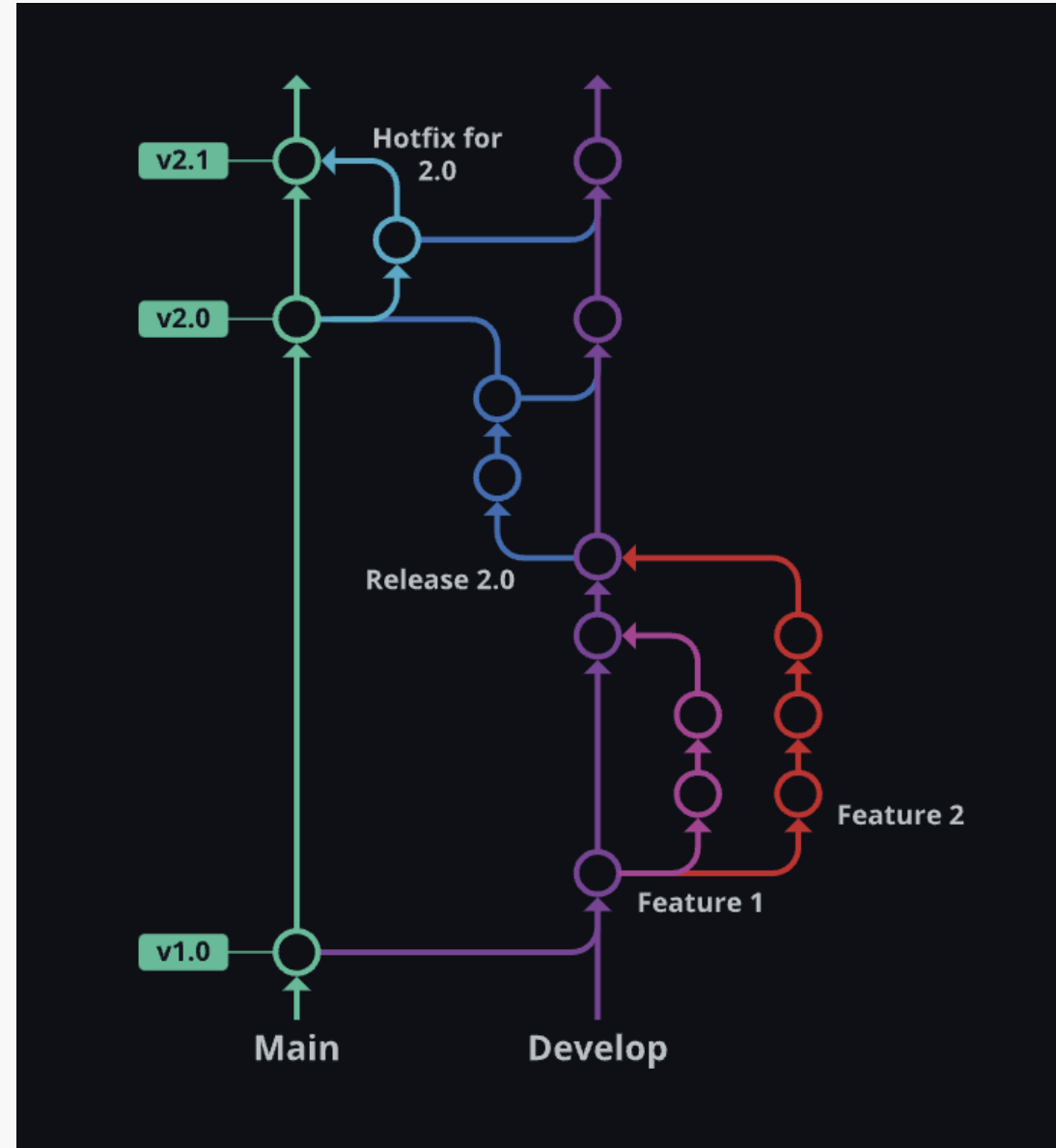


Branching Strategy

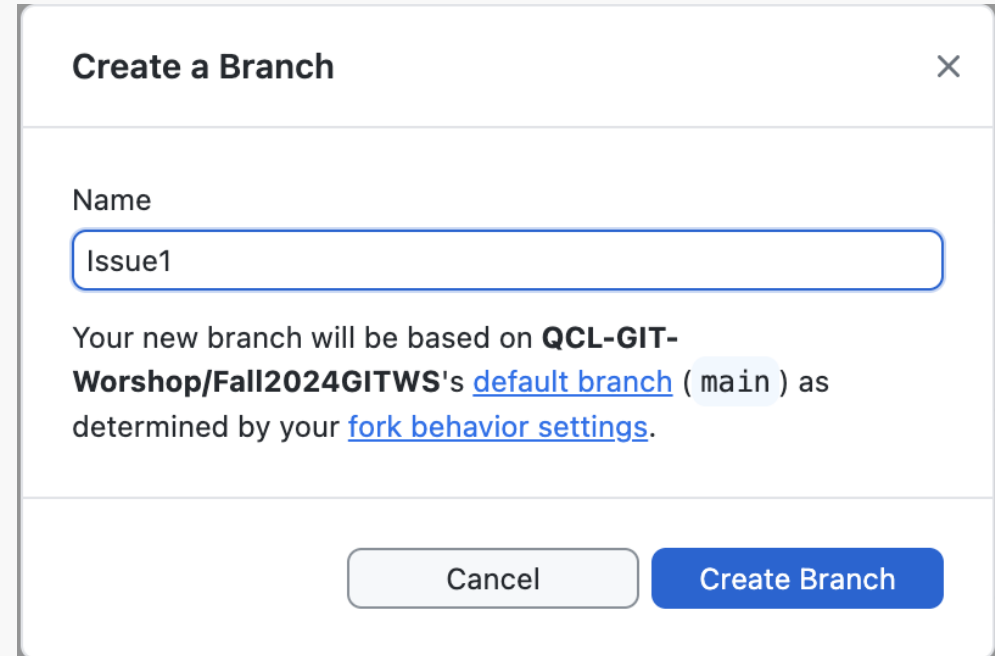
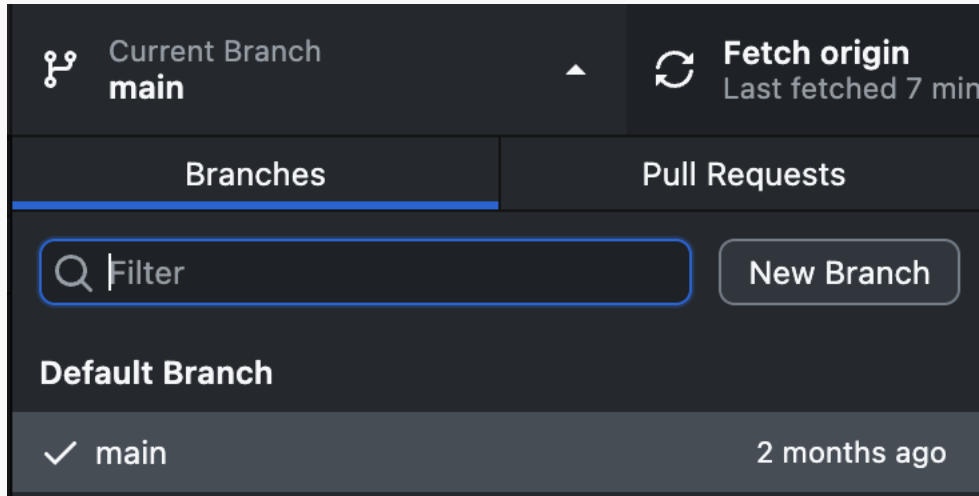
Branching allows developers to work on features or fixes in isolation without affecting the main codebase. It enables parallel development, experimentation, and the creation of feature branches.

Best Practices

- Creating descriptive branch names, keeping branches short-lived
- merging changes frequently to avoid conflicts
- Having a branching strategy that aligns with the project's needs.



Check your assignment and Create a branch



Create a branch

No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



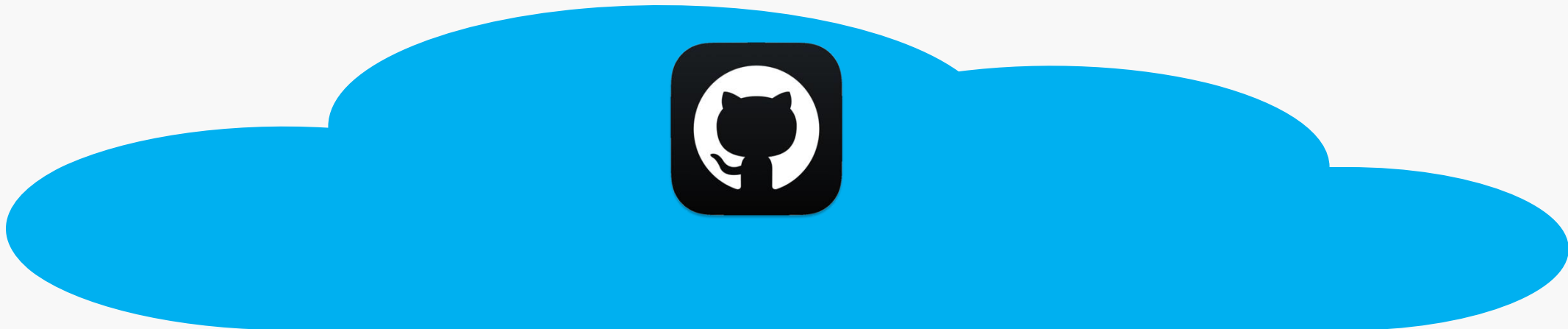
Publish your branch

The current branch (Issue1) hasn't been published to the remote yet. By publishing it to GitHub you can share it, open a pull request, and collaborate with others.

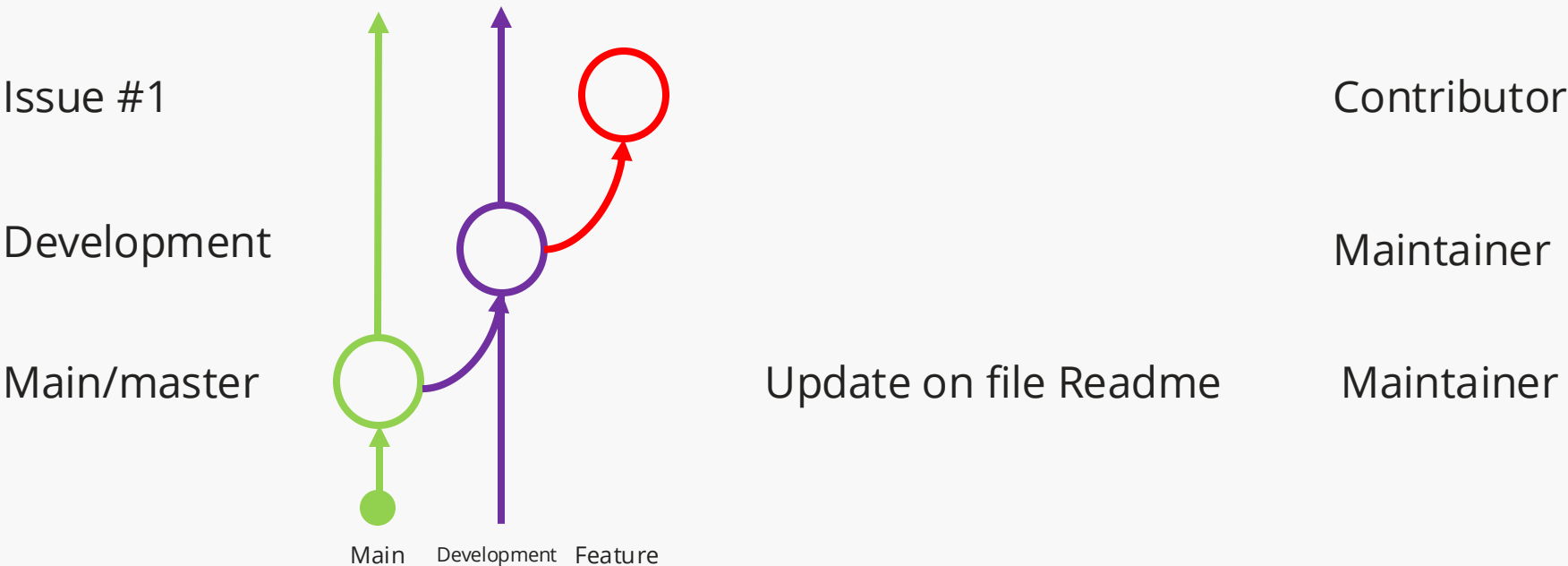
Publish branch

Always available in the toolbar or ⌘ P

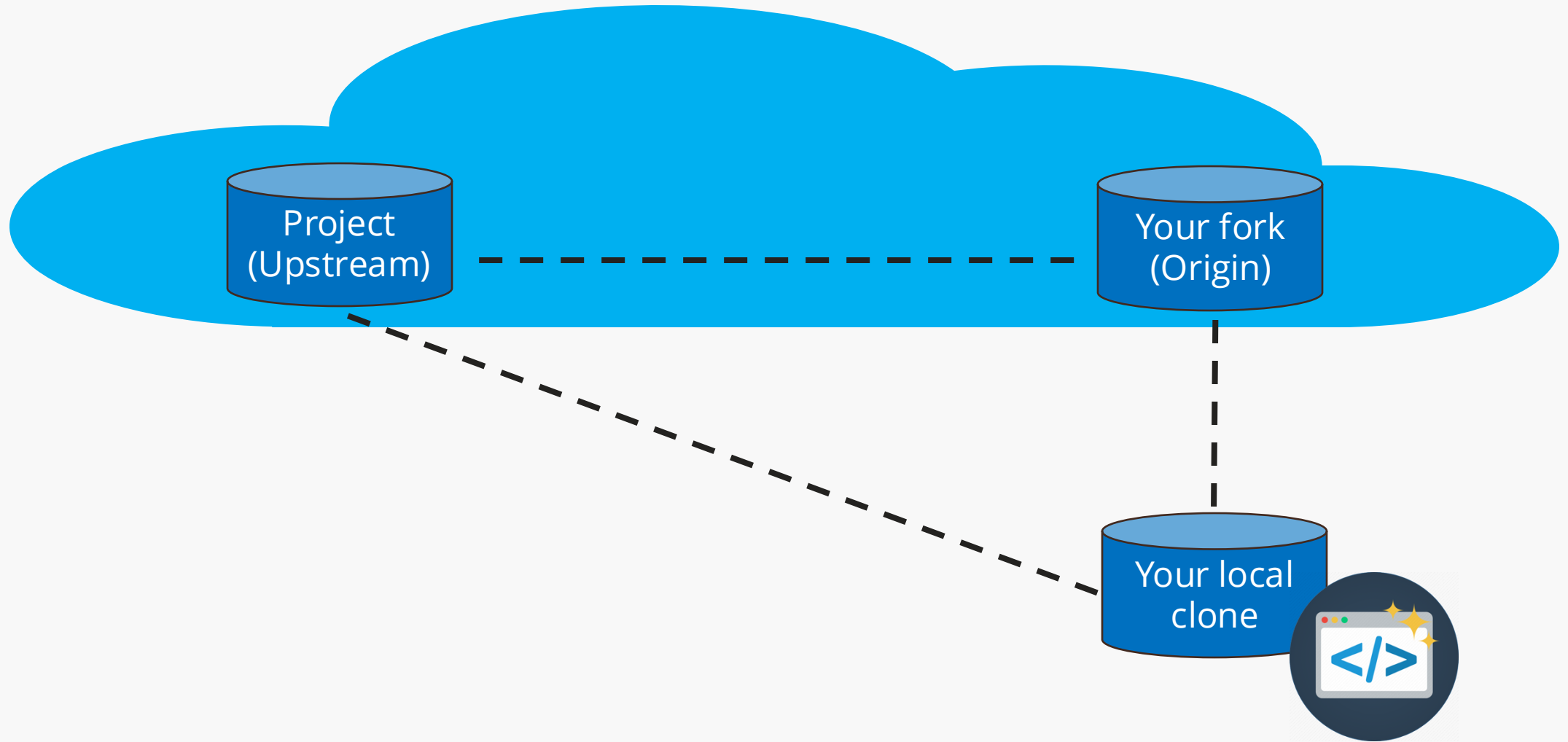
Project tracker



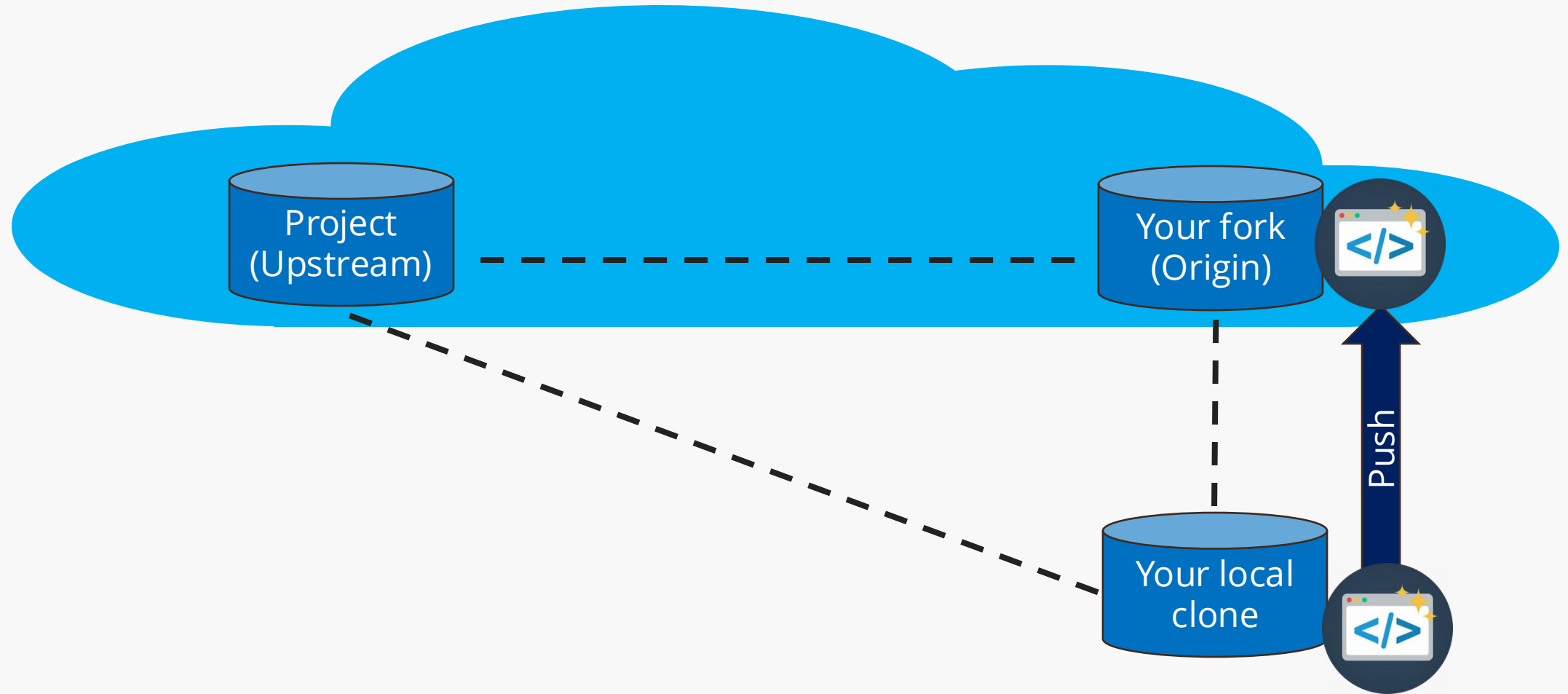
| Branch/TAG | GRAPH | COMMIT | USER |
|-------------|-------|-----------------------|-------------|
| Issue #1 | | | Contributor |
| Development | | | Maintainer |
| Main/master | | Update on file Readme | Maintainer |



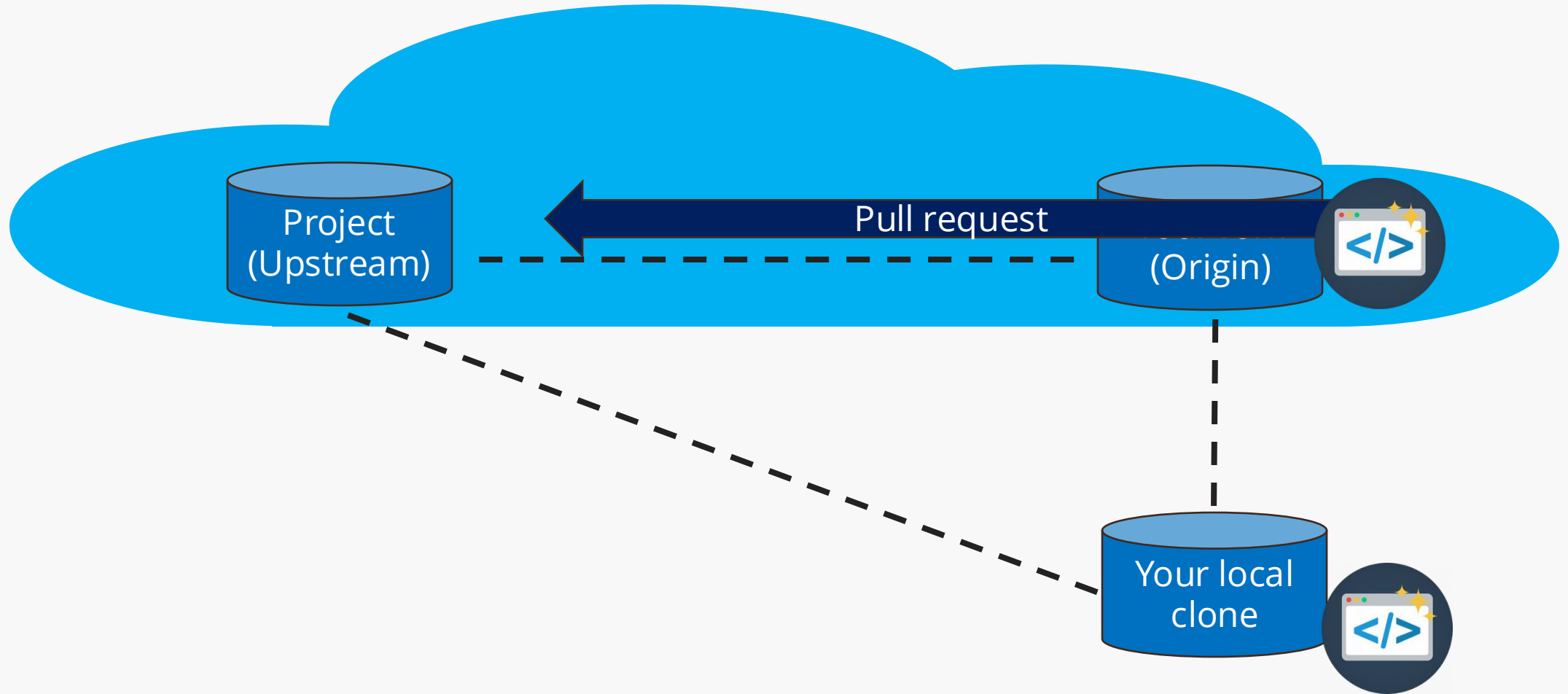
Contribute a change



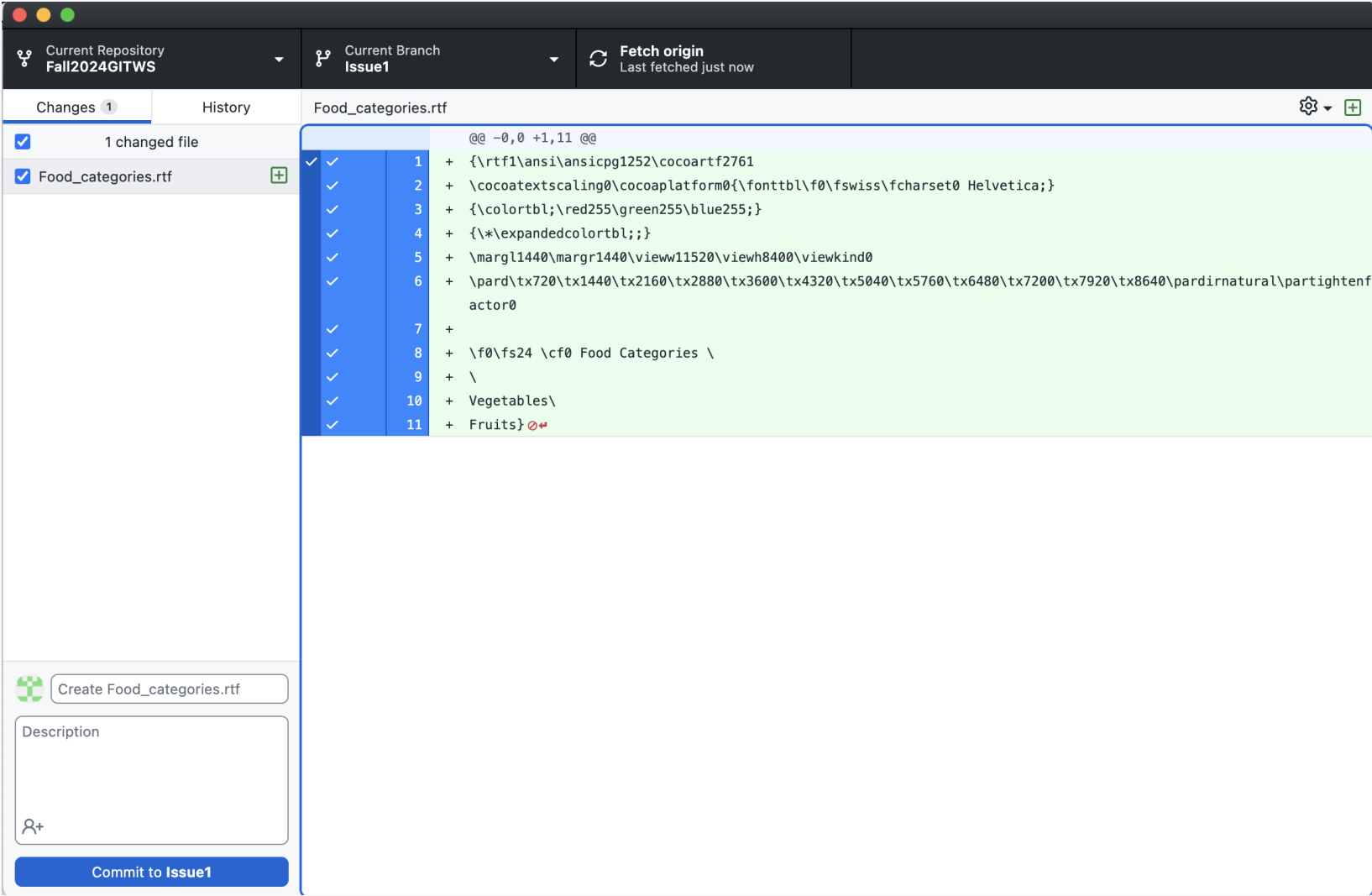
Contribute a change



Contribute a change





Activity! Work on your assigned issue



Activity! Push your changes to origin (Forked repo)

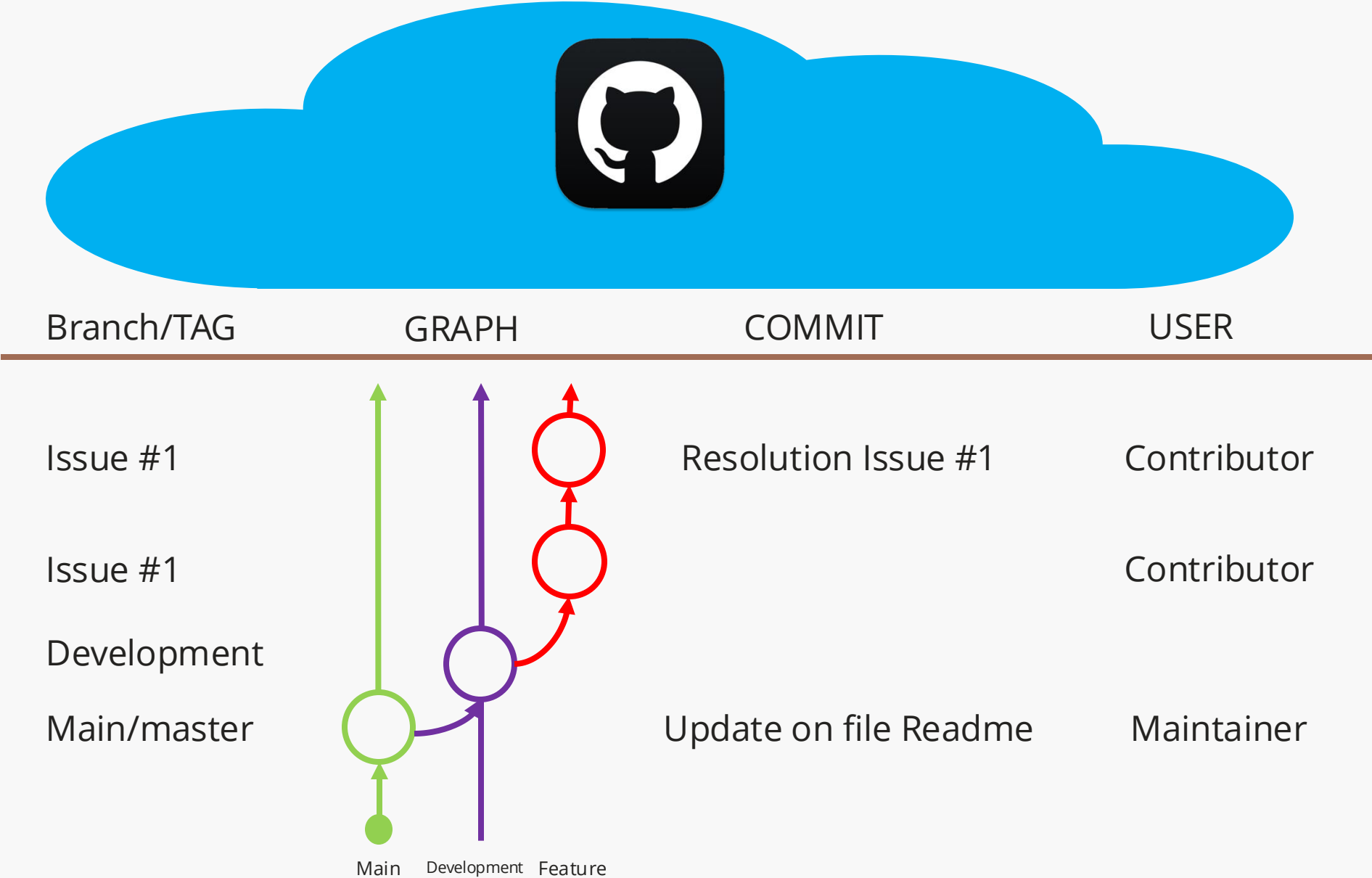
Push commits to the origin remote

You have 1 local commit waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or  

Push origin

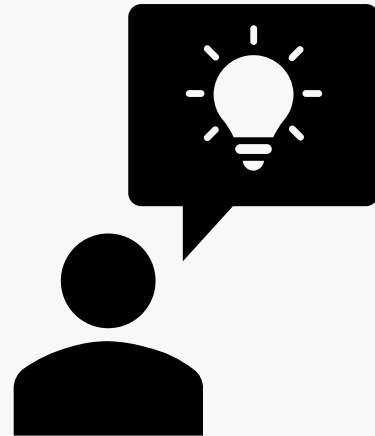
Project tracker



Pull Requests

Purpose of Pull Requests

Pull requests are used to propose changes, discuss modifications, and review code before merging it into the main branch. They allow for collaboration, feedback, and ensuring code quality.

A green rectangular button with rounded corners and a subtle gradient. The text "New pull request" is written in white, with "pull" in a lighter shade. A white hand cursor icon is pointing at the bottom right corner of the button.

New pull request

Creation Process

When creating a pull request, developers provide context about the changes, request reviews from team members, run automated tests, and ensure that the code meets the project's standards before merging.

Collaboration and Code Review

Facilitating Collaboration

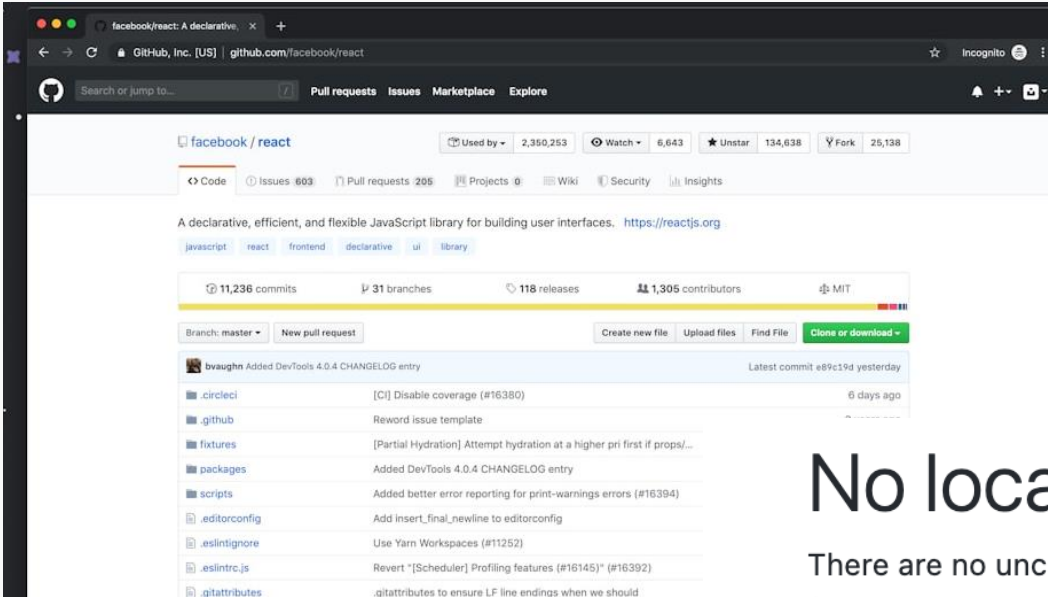
GitHub facilitates collaboration by providing tools for code review, discussions, and pull request comments. It fosters teamwork, knowledge sharing, and error detection through code reviews.



Code Review Process

Code reviews in GitHub involve team members reviewing code changes, providing feedback, suggesting improvements, and ensuring code quality. They play a vital role in maintaining code consistency and quality standards.

Activity! Create Pull Requests



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.



Preview the Pull Request from your current branch
The current branch (Issue1) is already published to GitHub.
Preview the changes this pull request will have before proposing your changes.

Branch menu or ⌘ ⌥ P

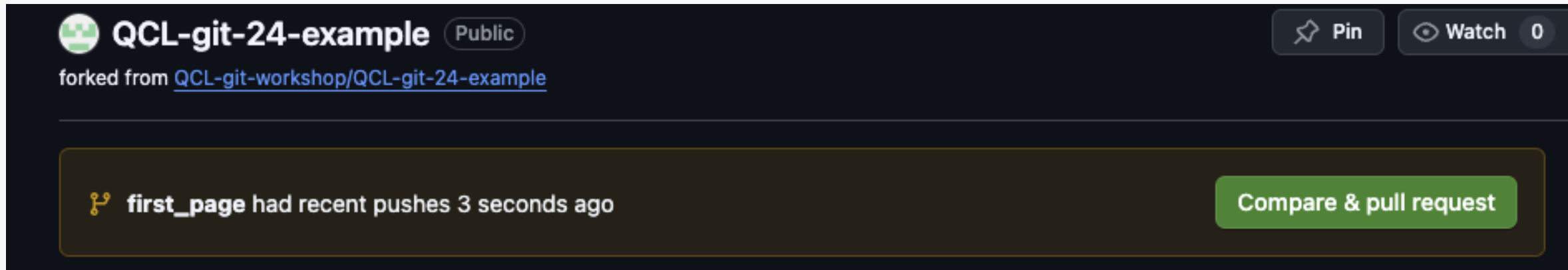
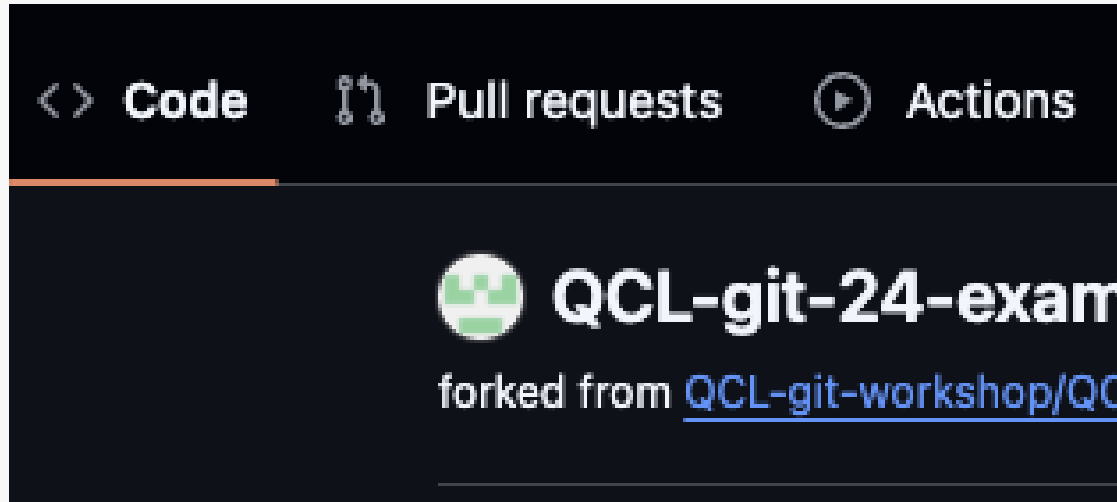
Preview Pull Request ▾

✓ Preview Pull Request

Create Pull Request

Open the repository in your external editor

Activity! Pull Requests



Pull Requests

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](#). [Learn more about diff comparisons here](#).



base repository: QCL-git-workshop/QCL-git-2...

base: main



head repository: DeadMarshes/QCL-git-24-ex...

compare: first_page

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



Add a title

Start first_page

Add a description

Write

Preview

H

B

I

≡

<>

🔗

≡

≡

≡

📎

@

🗨

←

📄

Add your description here...

📄 Markdown is supported

📎 Paste, drop, or click to add files

☒ Allow edits by maintainers ?

Create pull request



Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Reviewers



No reviews

Assignees



No one—[assign yourself](#)

Labels



None yet

Projects



None yet

Milestone



No milestone

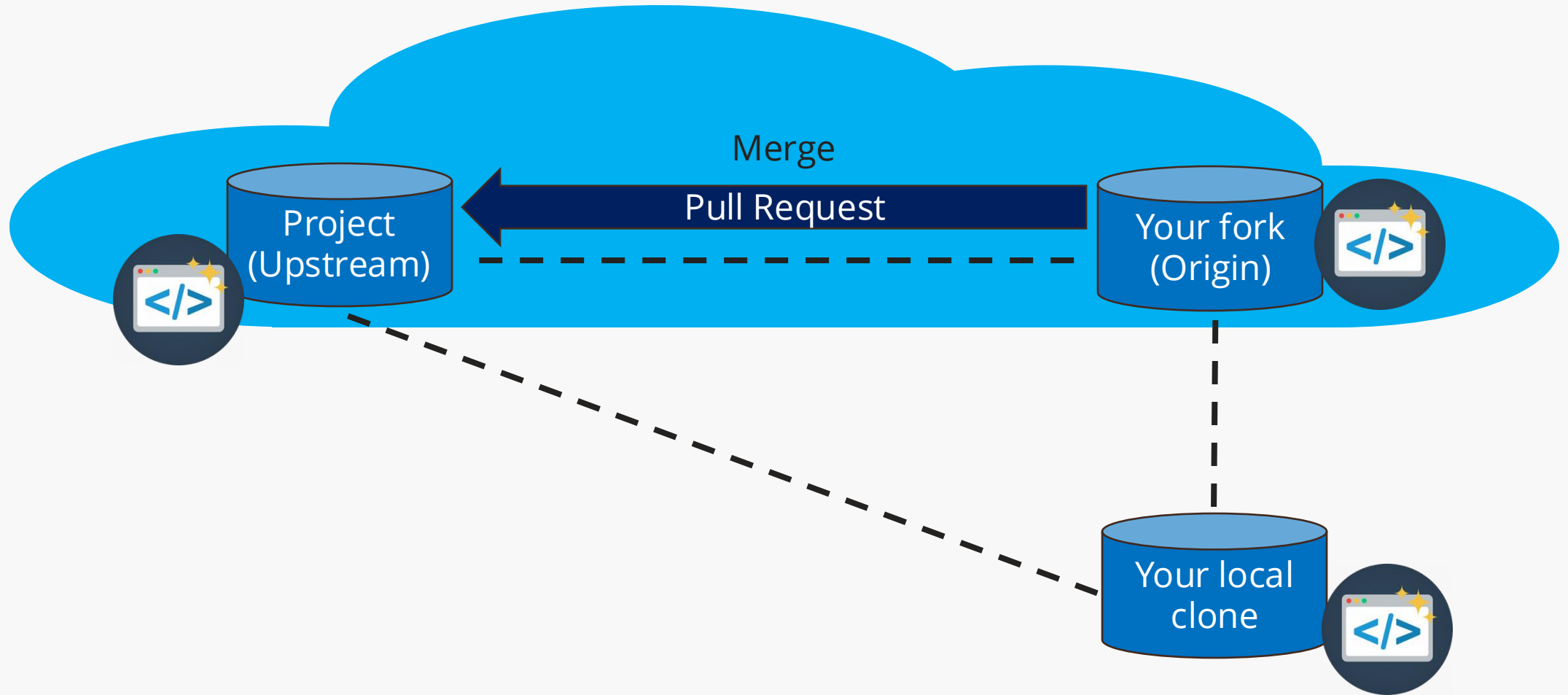
Development

Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

[GitHub Community Guidelines](#)

Integrate a change



Maintainer reviews the PR



Looks good!

Accept the PR by merging it into master: choose "squash and merge" strategy for now and click **Merge pull request** and then **Confirm**.

Merge pull request

You can also [open this pull request in a web browser](#)

✓ Create a merge commit

All commits from this branch will be added to the base branch via a merge commit.

Squash and merge

The 9 commits from this branch will be combined into one commit in the base branch.

Rebase and merge

The 9 commits from this branch will be rebased and added to the base branch.

Start first_page #2

Merged


dan-perezc merged 2 commits into QCL-git-workshop:main

Maintainer close the issue



Create first file #1


Open dan-perezc opened this issue 7 hours ago · 0 comments




dan-perezc commented 7 hours ago

Create the first file foods describing different food categories

Member ...




dan-perezc assigned DeadMarshes 7 hours ago



DeadMarshes mentioned this issue 6 hours ago

Start first_page #2











Merged





Add a comment

Write


Preview

H B I          

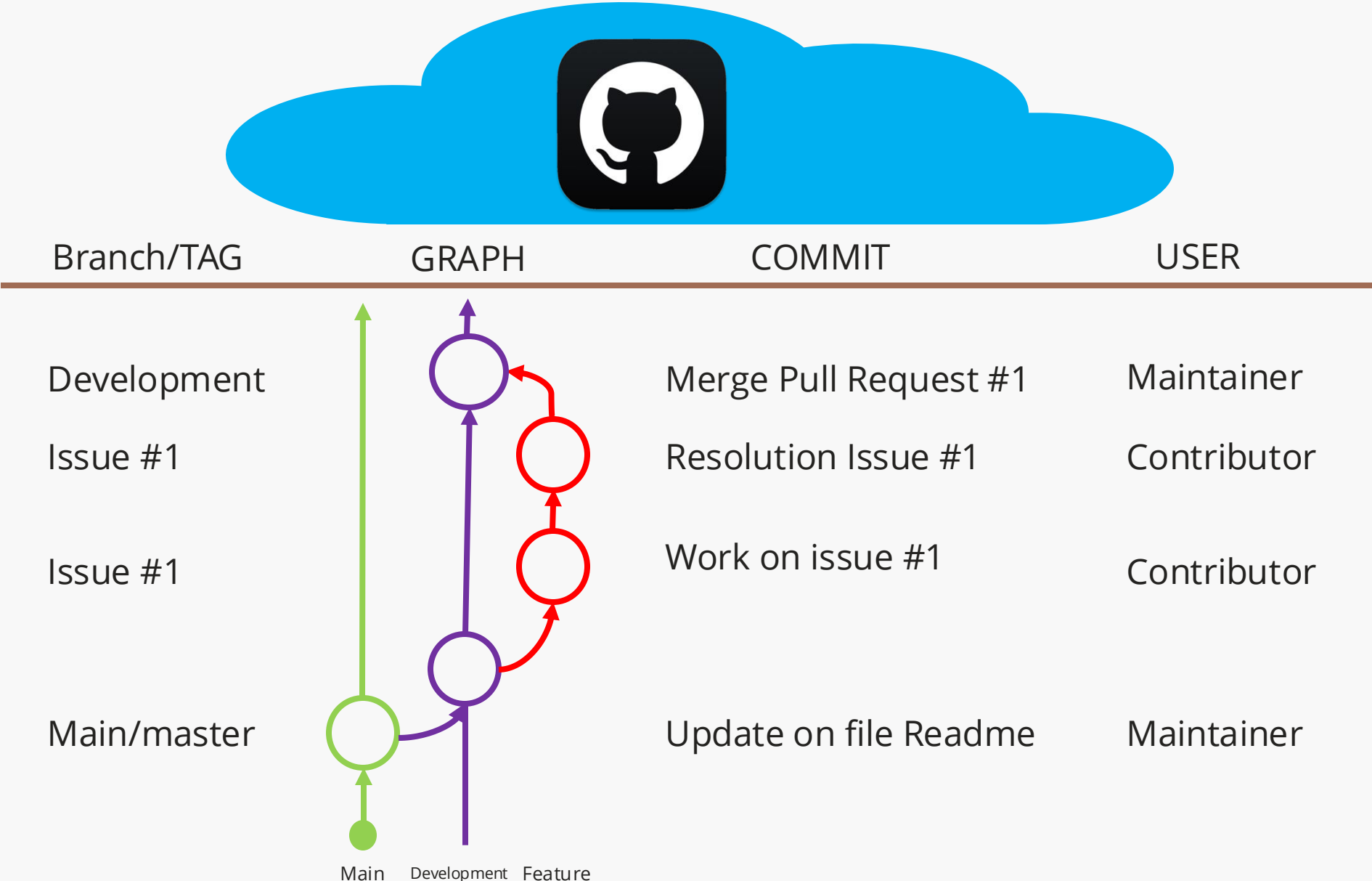
Add your comment here...

 Markdown is supported  Paste, drop, or click to add files

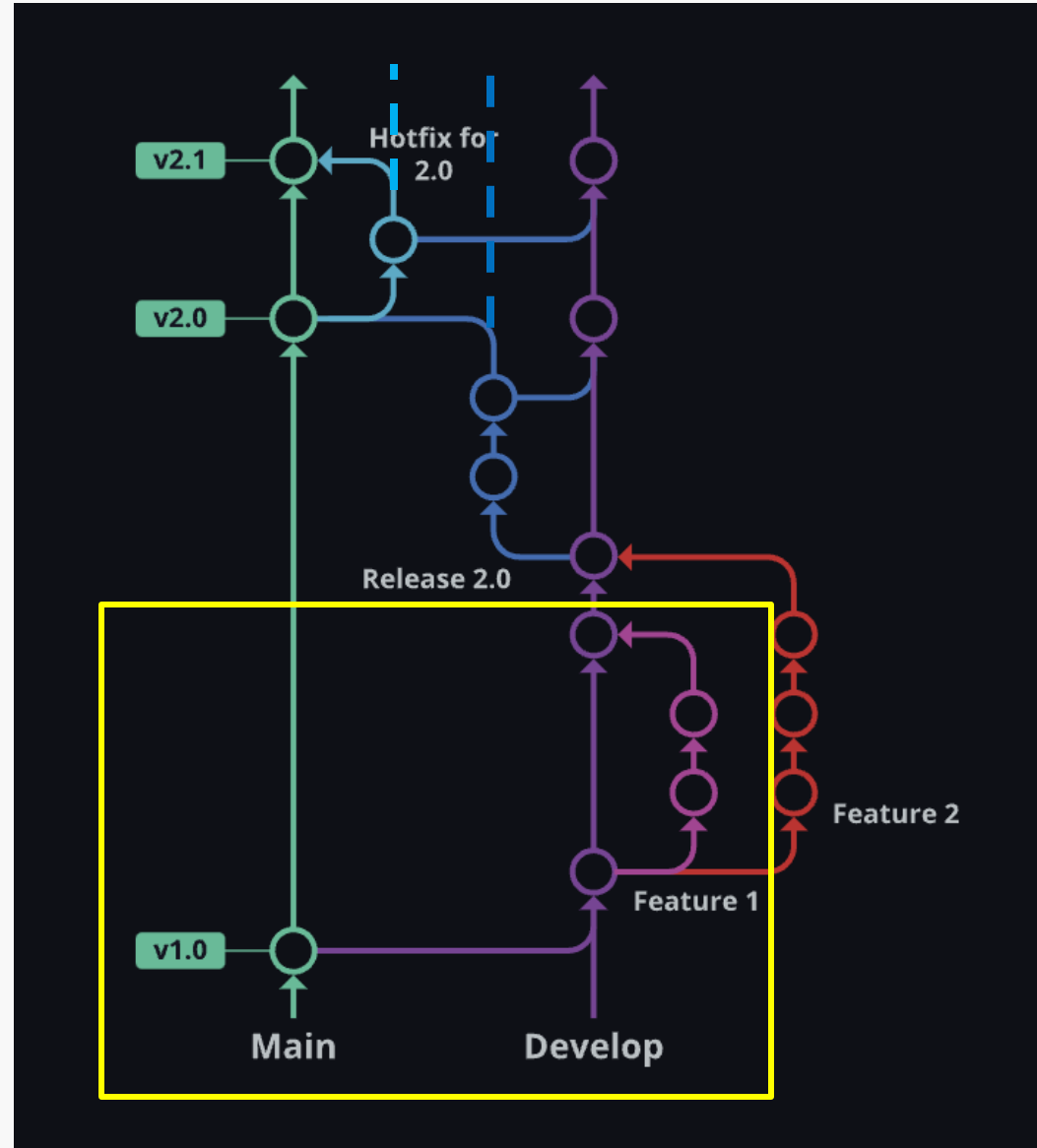
Close issue Comment

 Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Project tracker



Branching Strategy



Let's cleanup!

Stand on main branch and pull upstream

Delete feature branch and push the change to origin

Workflow Best Practices

| Best Practice | Description |
|------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Branch Protection | Enable branch protection rules to prevent direct commits to critical branches and ensure changes go through code review. |
| Continuous Integration | Integrate CI tools to automatically test and build code changes, ensuring the project's stability and quality. |
| Documentation | Maintain detailed documentation in repositories to enhance project accessibility, onboarding, and knowledge sharing. |
| Code Reviews | Encourage regular code reviews to catch bugs early, share knowledge, and maintain code quality standards. |
| Issue Tracking | Effectively use GitHub issues to track tasks, bugs, and enhancements, facilitating project management and collaboration. |

Cheat sheet Git code

SETUP & INIT Configuring user information, initializing and cloning repositories

git init initialize an existing directory as a Git repository

git clone [url] retrieve an entire repository from a hosted location via URL

SHARE & UPDATE Retrieving updates from another repository and updating local repos

git remote add [alias] [url] add a git URL as an alias

git fetch [alias] fetch down all the branches from that Git remote

git merge [alias]/[branch] merge a remote branch into your current branch to bring it up to date

git push [alias] [branch] Transmit local branch commits to the remote repository branch

git pull fetch and merge any commits from the tracking remote branch

STAGE & SNAPSHOT Working with snapshots and the Git staging area

git status show modified files in working directory, staged for your next commit

git add [file] add a file as it looks now to your next commit (stage)

git reset [file] un-stage a file while retaining the changes in working directory

git diff diff of what is changed but not staged

git diff --staged diff of what is staged but not yet committed

git commit -m "[descriptive message]" commit your staged content as a new commit snapshot

USER

git config user.name

BRANCH

git branch list all branches

git branch my-feature create new branch

git checkout branch change branch

git merge my-feature (step on main and merge feature branch)

git branch -d my-feature Delete branch

Additional Info



<https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-and-deleting-branches-within-your-repository>

<https://www.geeksforgeeks.org/version-control-systems/>

https://www.geeksforgeeks.org/ultimate-guide-git-github/?ref=gcse_ind

<https://www.gitkraken.com/learn/git/git-flow>

<https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>