Git and Github Essencials

Luis Francisco Contreras Gonzalez

Introduction to Version Control

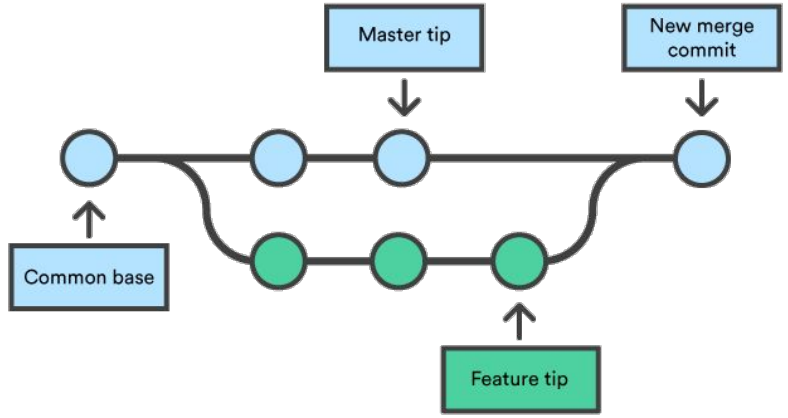
What is Version Control?

A system that records changes to files over time.

Allows you to revert to specific versions, track changes, and collaborate with others.

Why is it important?

Essential for collaboration, backup, and maintaining a history of changes.



What is Git?

Definition:

A distributed version control system created by Linus Torvalds in 2005.

Key Features:

Tracks changes to files.

Allows multiple people to work on the same project.

Branching and merging.

High performance, even with large projects.

https://git-scm.com/downloads

What is GitHub?

Definition:

A cloud-based hosting service that manages Git repositories.

Features:

Collaboration: Pull requests, code reviews.

Project management: Issues, project boards.

CI/CD: Integration with various continuous integration services.

git

- -Is a version control software
- -Is used as a command line tool and ran locally
- -Tracks code and version history
- -Allows you to work on different versions/branches of a code base
- -Helps to synchronizes different versions of the same code base (local code base, remote code base etc.)

GitHub

- -Is a web application that hosts remote git repositories
- -Owned by Microsoft
- -Is deeply integrate with git
- -Provides extra functionality on top of git
- -Mainly used by teams of 2 or more people

What is a GitHub Repository/Repo?

A GitHub repository, often called a "repo," is like a folder for your project on GitHub.

It contains all your project's files, including code, images, and documentation, along with a history of all the changes made to these files.

Repositories help you organize and collaborate on projects with others by keeping everything in one place.

main 1 branch 0 tags Go to file Add file Code

contrerasgonzalezluisfco add test memcache 7cf7a0d yesterday 34 commits

allure-results	add test memcache	yesterday
data	add test help button	yesterday
info test cases	Add compare test case	last month
pages	add test	last week
reports	add test help button	yesterday
reportsclear	add option for multiple runs	3 weeks ago
tests	add test memcache	yesterday
trig	modify readme	2 weeks ago
utils	compare output.txt	last month
.gitignore	add test for login and perfweb validation	last month
README.MD	add steps to allure compare	last week
pytest.ini	add compare trig for multiple ex	3 weeks ago
xpath_how_to.txt	modify readme	2 weeks ago

README.MD

Selenium with Python Pefweb automation

Perfweb automation with Pythong, Selenium and Pytest

http://perfweb.ssd.hursley.ibm.com

Installation Requirements

About

No description, website, or topics provided.

Readme

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Languages

Python 100.0%

LOCAL REPOSITORY

Resides on the computer of a team member

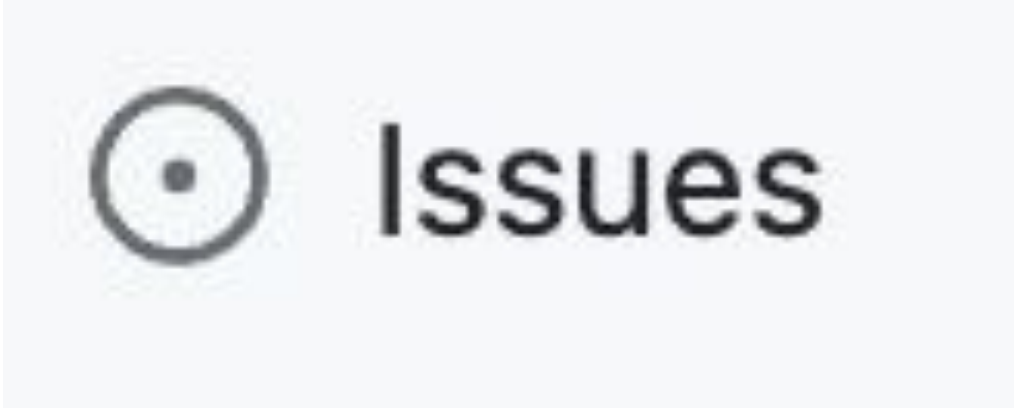
REMOTE REPOSITORY

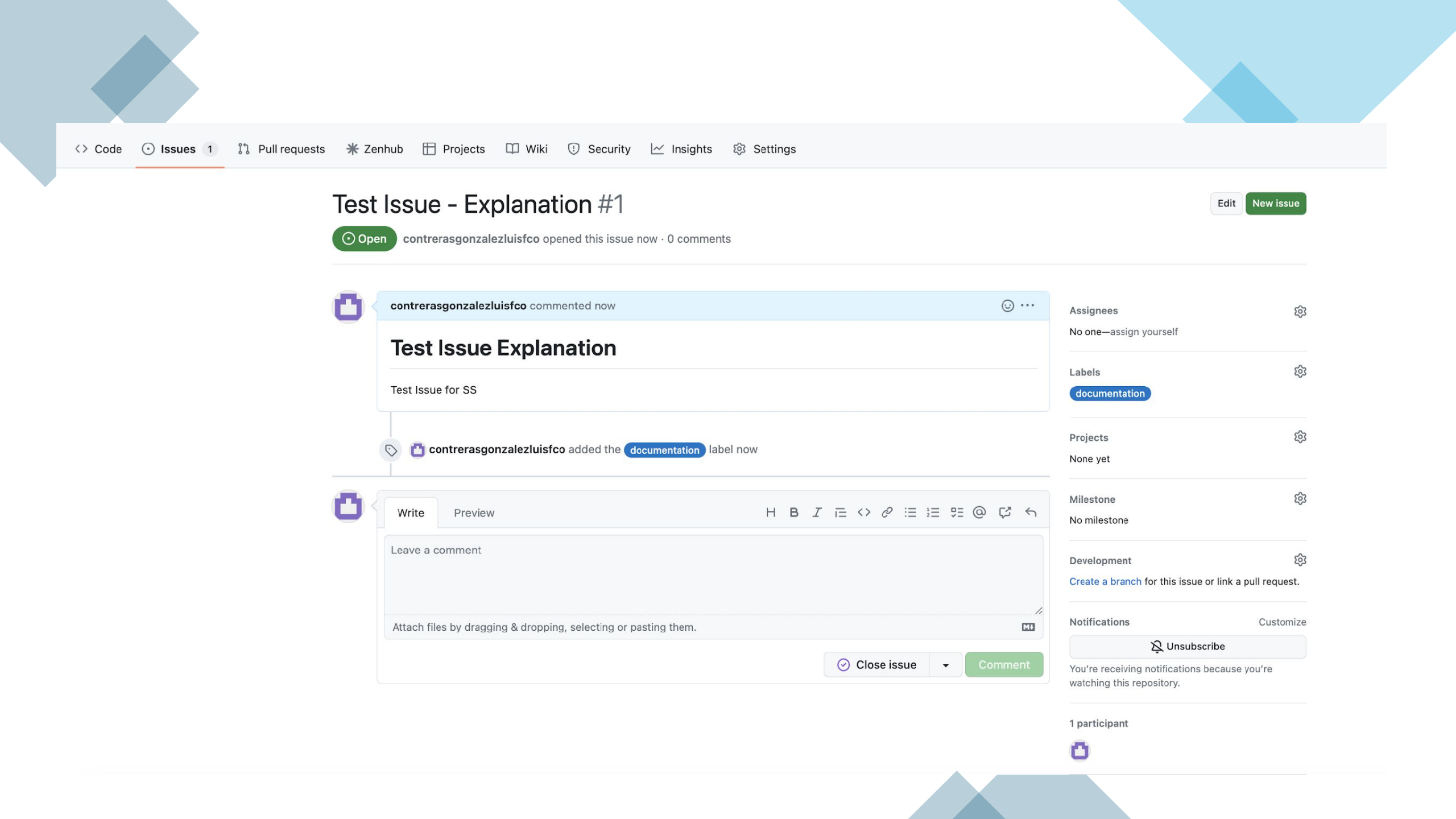
Are hosted on a server that is accessible for all team members

Issues Tab

Issues in GitHub are a way to track tasks, enhancements, bugs, or any other work that needs to be done in a project. They help teams organize and discuss their work in a structured way.

The Issues tab is where you can view, create, and manage all the issues related to a project. It's like a to-do list for the project, showing what needs to be done, who’s working on it, and what’s already completed.

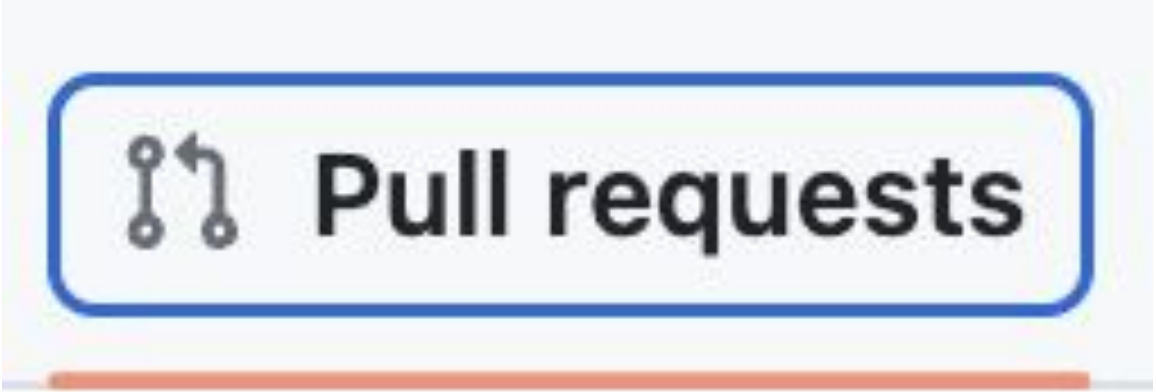



Code
Issues 1
Pull requests
Zenhub
Projects
Wiki
Security
Insights
Settings
Test Issue - Explanation #1
Edit
New issue
Open
contrerasgonzalezluisfco opened this issue now · 0 comments
contrerasgonzalezluisfco commented now
Test Issue Explanation
Test Issue for SS
contrerasgonzalezluisfco added the documentation label now
Write
Preview
Leave a comment
Attach files by dragging & dropping, selecting or pasting them.
Close issue
Comment
Assignees
No one—assign yourself
Labels
documentation
Projects
None yet
Milestone
No milestone
Development
Create a branch for this issue or link a pull request.
Notifications
Customize
Unsubscribe
You're receiving notifications because you're watching this repository.
1 participant

Pull request Tab

A Pull Request (PR) is a way to propose changes to a codebase in GitHub. It allows developers to review, discuss, and approve code changes before they are merged into the main project.

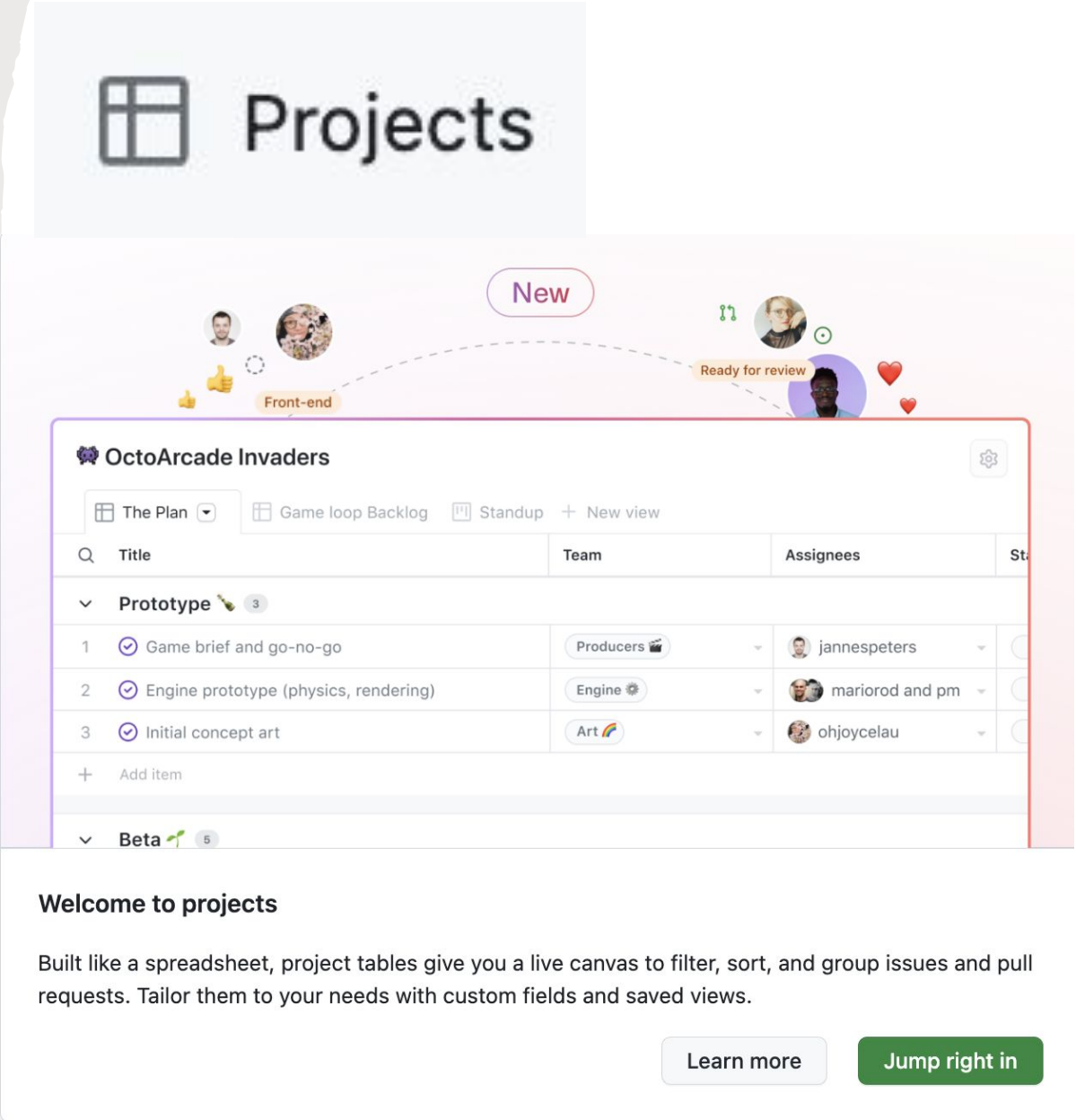
The Pull Request tab is where you can view, create, and manage all pull requests for a project. It shows proposed changes, allows for discussion and review, and tracks the status of each request.



Projects Tab

The Projects tab in GitHub is a tool that helps you organize and manage your work using boards, similar to a Kanban board. It allows you to create and track tasks, set priorities, and visualize your workflow.

In the Projects tab, you can create boards to organize issues, pull requests, and notes into customizable columns like "To Do," "In Progress," and "Done." This helps teams stay organized and ensures that everyone knows what needs to be done and what's currently being worked on.



Wiki

The Wiki is useful for:

- Project Documentation: Keeping detailed information about your project in one place.
- Collaboration: Allowing team members to contribute to and update documentation.
- Organization: Structuring information with pages and links, making it easy to find and navigate.

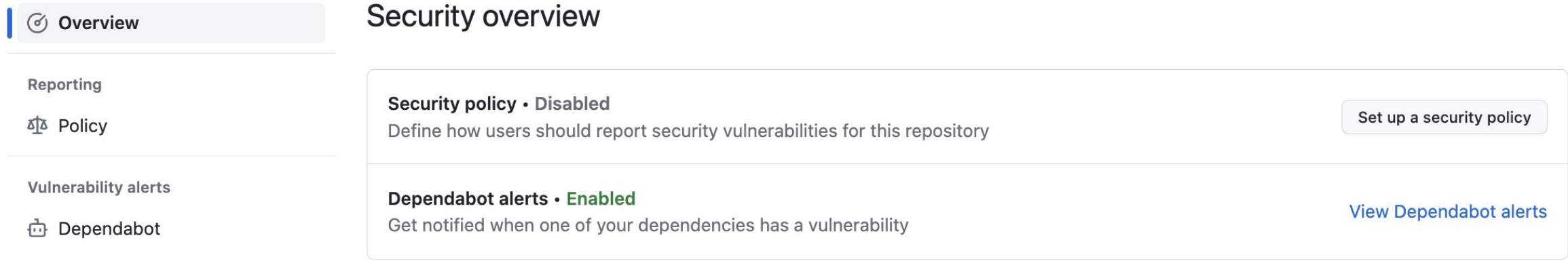
Security

The Security tab allows you to:

Track Vulnerabilities: Find and address security issues in your code or dependencies.

Set Security Policies: Define rules and guidelines for how security issues are handled.

Monitor Activity: Get alerts and reports on any potential security risks in your project.



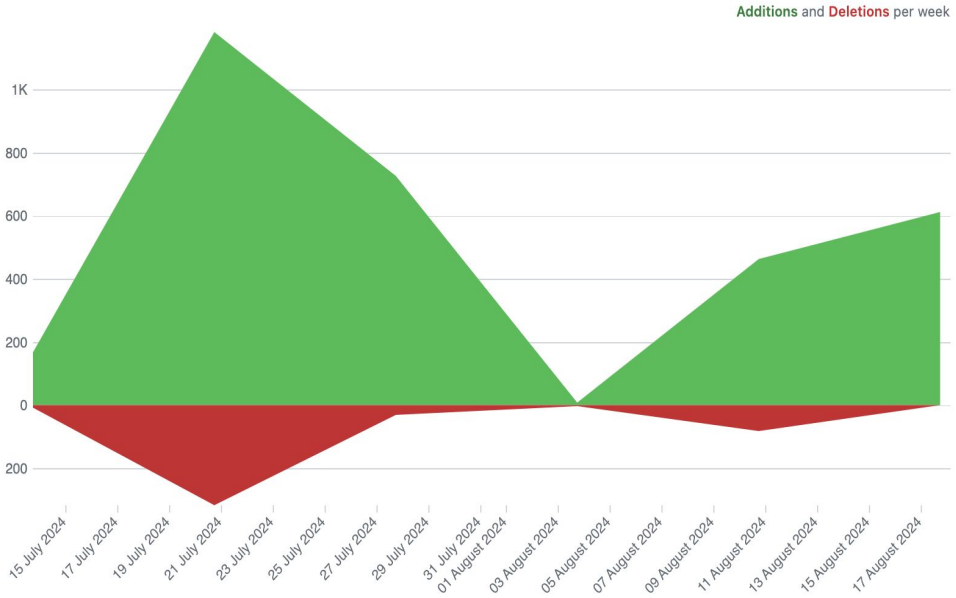
Insights Tab

The Insights tab provides:

- Contributions: See how often people are contributing to the project, including commits, pull requests, and issues.
- Traffic: Track the number of views and clones of your repository.
- Dependency Graph: View the dependencies your project relies on and identify potential security risks.
- Community: Monitor community activity, including issue reporting, discussions, and pull requests.

- Pulse
- Contributors
- Community
- Commits
- Code frequency
- Dependency graph
- Network
- Forks

Code frequency over the history of **contrerasgonzalezluisfco/perfweb_automation**



- Pulse
- Contributors
- Community
- Commits
- Code frequency
- Dependency graph
- Network
- Forks

August 14, 2024 – August 21, 2024

Period: 1 week

Overview

0 Active pull requests

1 Active issue

0 Merged pull requests	**0** Open pull requests	**0** Closed issues	**1** New issue

Excluding merges, **1 author** has pushed **8 commits** to main and **8 commits** to all branches. On main, **76 files** have changed and there have been **612 additions** and **0 deletions**.

1 Issue opened by **1** person

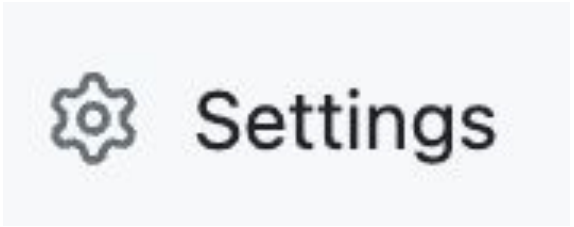
Test Issue - Explanation

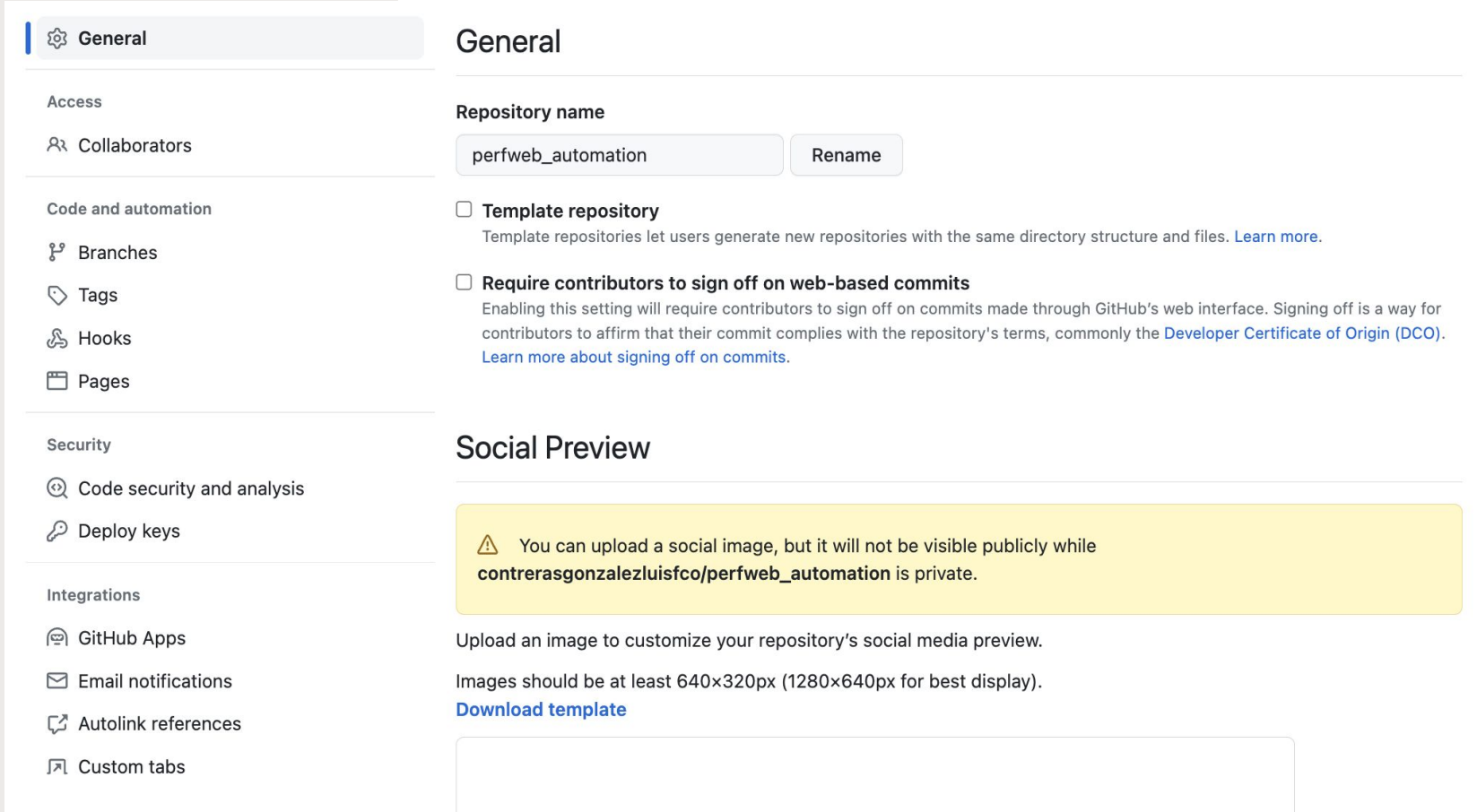
#1 opened 2 hours ago

Settings Tab

The Settings tab in GitHub is where you manage all the configuration options for your repository. It allows you to customize how your project works, who can access it, and other important details.

- Manage Access: Control who can see and contribute to your project.
- Configure Features: Enable or disable features like issues, wikis, and GitHub Pages.
- Set Repository Details: Update the repository name, description, and default branch.
- Manage Webhooks and Integrations: Connect your repository to other tools and services.





Basic Git Workflow

Clone a Repository:

git clone <repository-url>

Make Changes:

Edit files as needed.

Stage Changes:

git add <file> (Stages specific files).

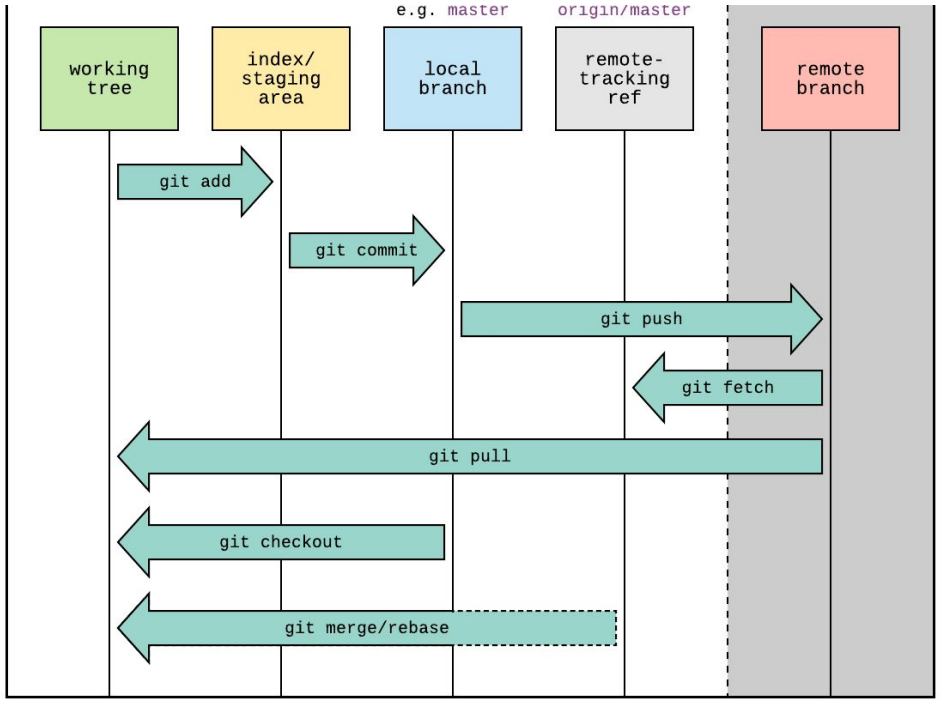
git add . (Stages all changes in the current directory).

Commit Changes:

git commit -m "commit message" (Commits staged changes).

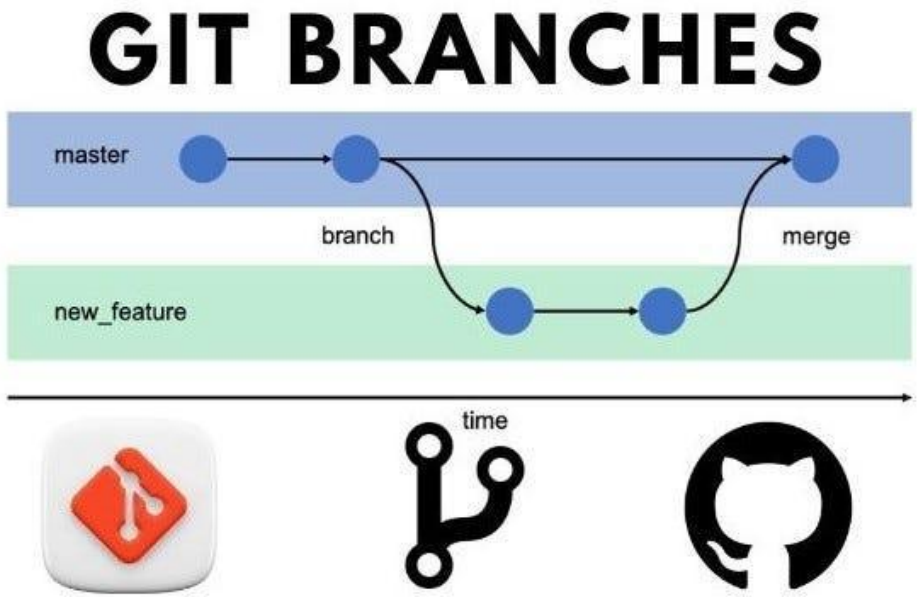
Push Changes:

git push (Uploads local changes to the remote repository).



Git Branches

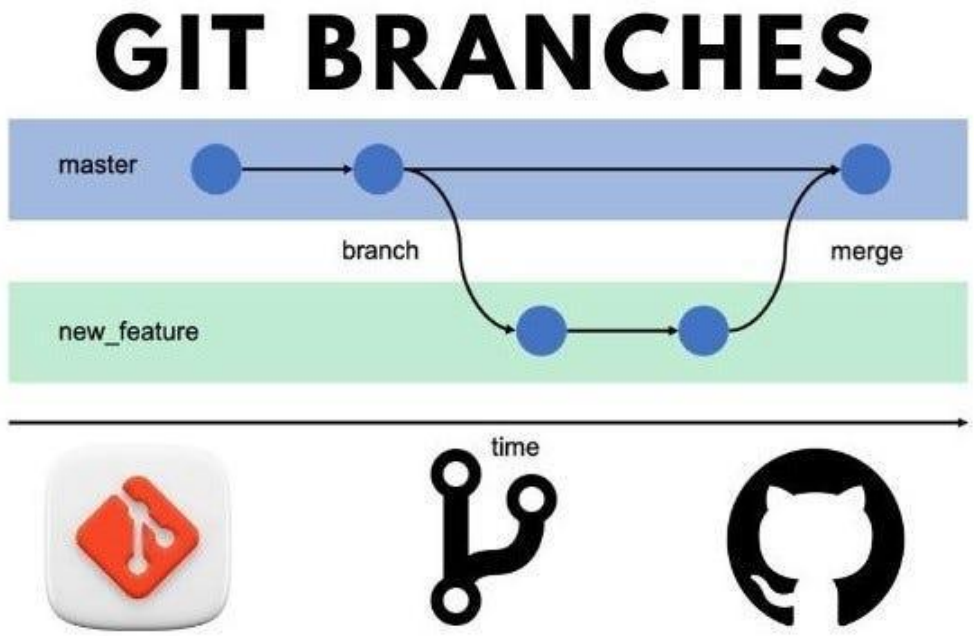
A Git branch is like a separate line of development in your project. Imagine a tree with different branches—each branch represents a different version of your project where you can make changes without affecting the main codebase.

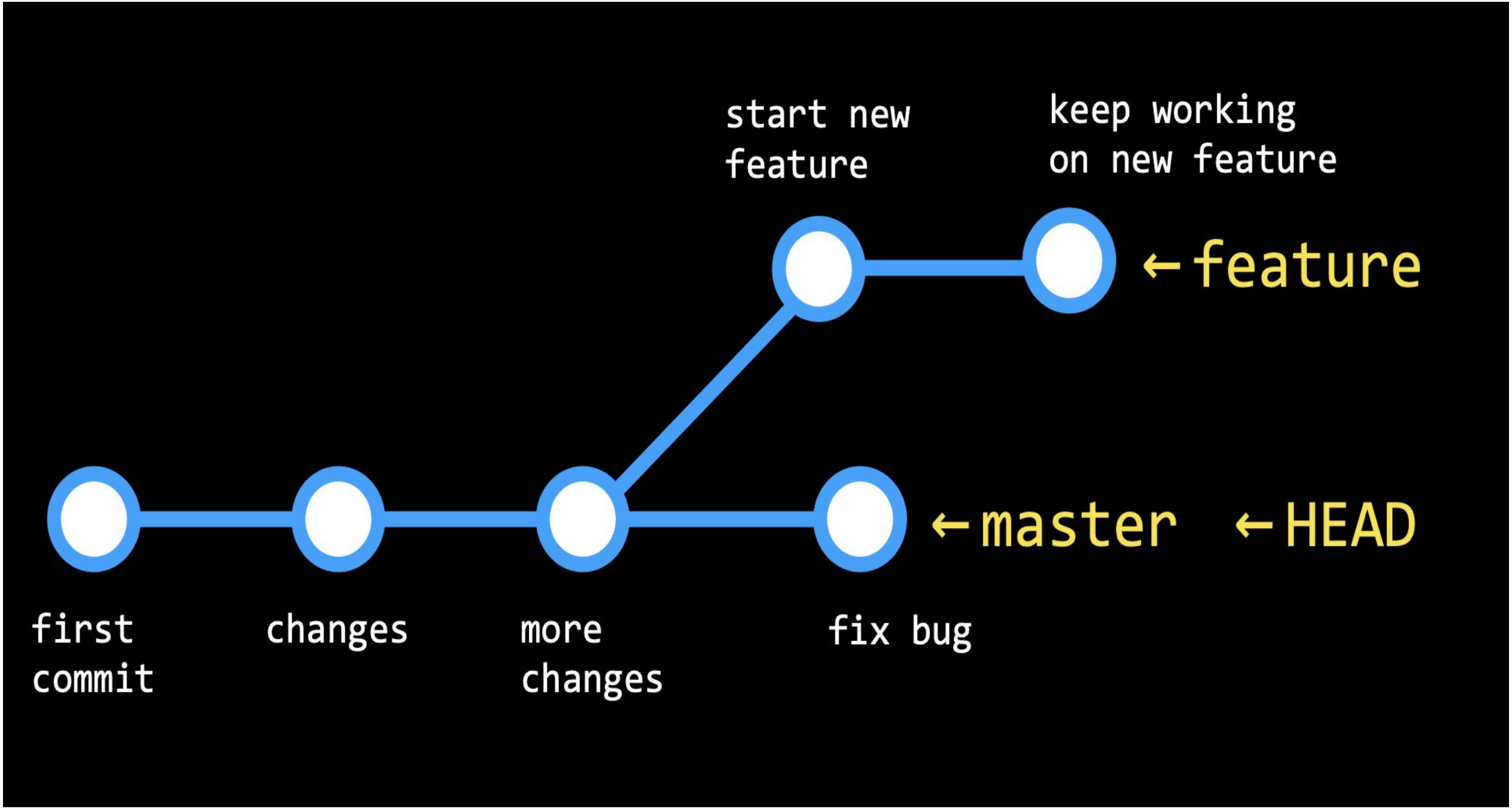


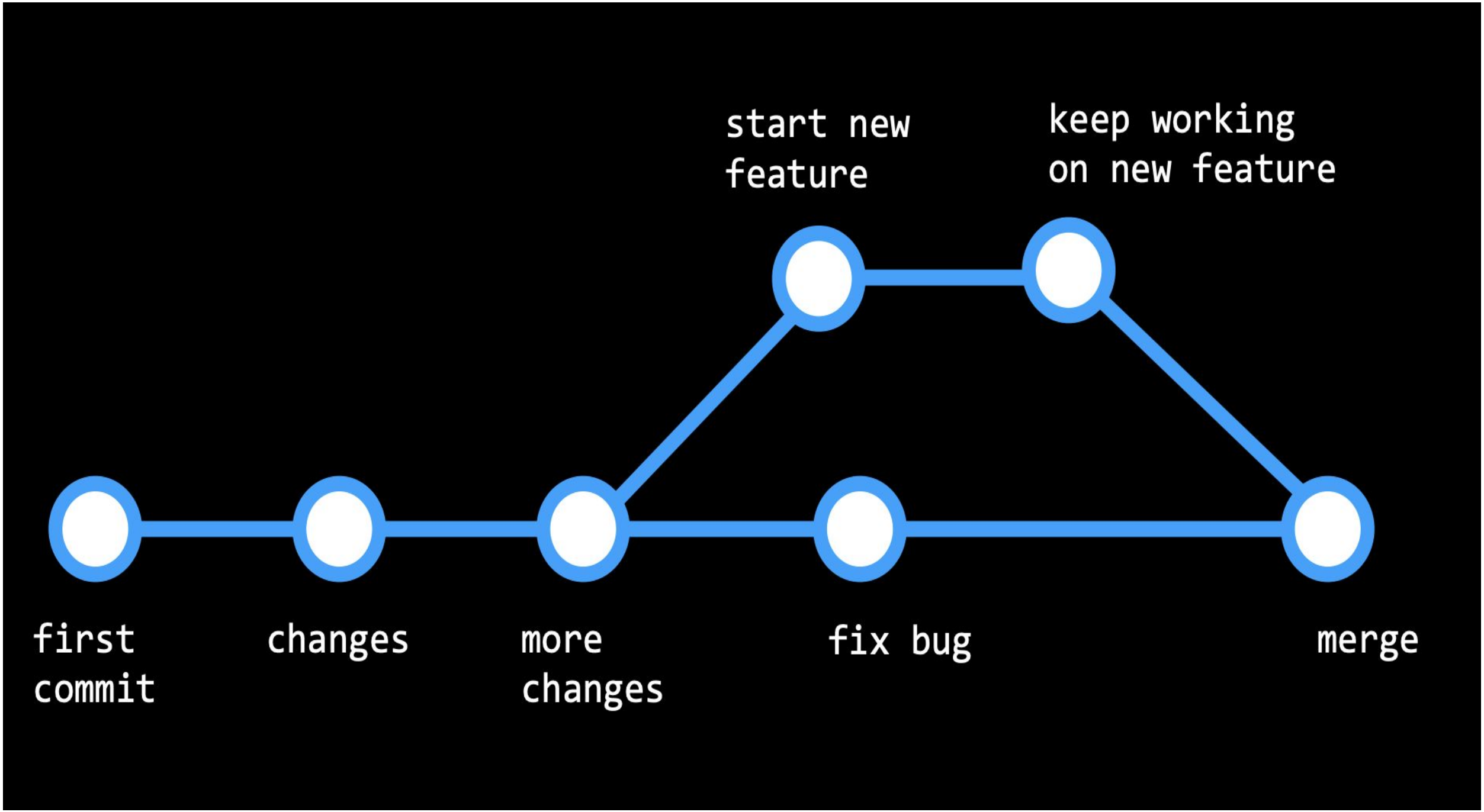
Why Branches ?

Branches let you:

- Work on new features or fix bugs without disturbing the main project.
- Test changes safely before merging them back into the main codebase.
- Collaborate with others on different parts of the project simultaneously.




start new feature
keep working on new feature
←feature
←master
←HEAD
first commit
changes
more changes
fix bug


start new feature
keep working on new feature
first commit
changes
more changes
fix bug
merge

Common Git Commands

Status:

git status (Shows the status of changes).

Log:

git log (Displays commit history).

Branching:

git branch (Lists branches).

git checkout -b <branch-name> (Creates a new branch and switches to it).

Merging:

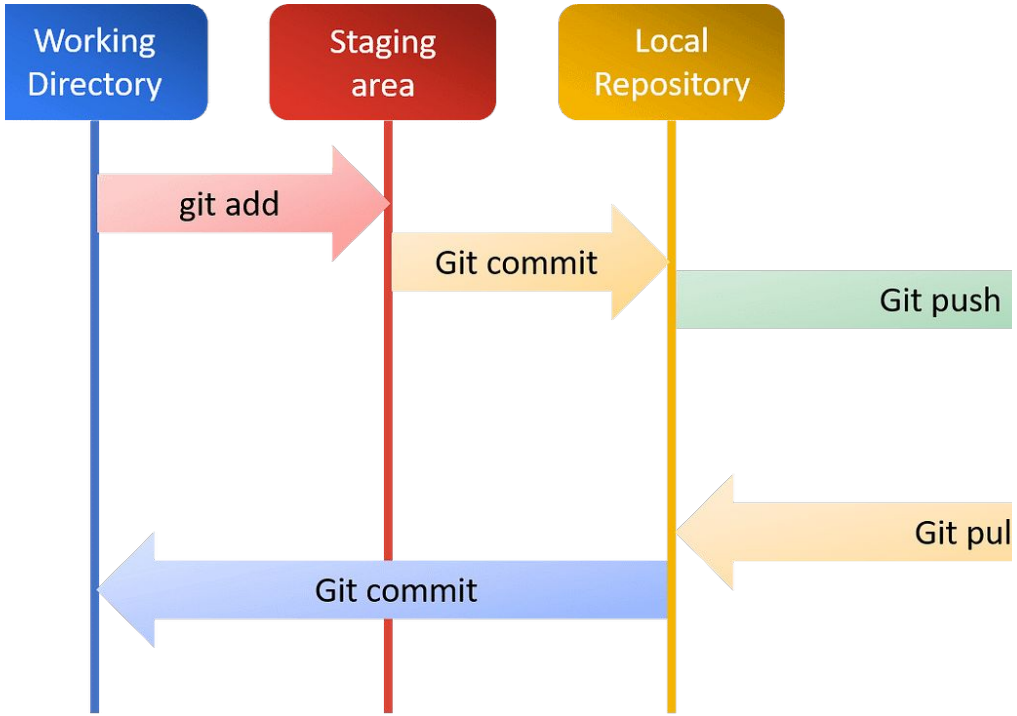
git merge <branch-name> (Merges changes from one branch into another).

Pulling:

git pull (Fetches and merges changes from the remote repository).

Git fetch:

git fetch (Downloads updates from the remote repository without modifying your local files)



Basic GitHub Workflows

Forking a Repository:

Create your own copy of someone else's project.

Creating a Pull Request:

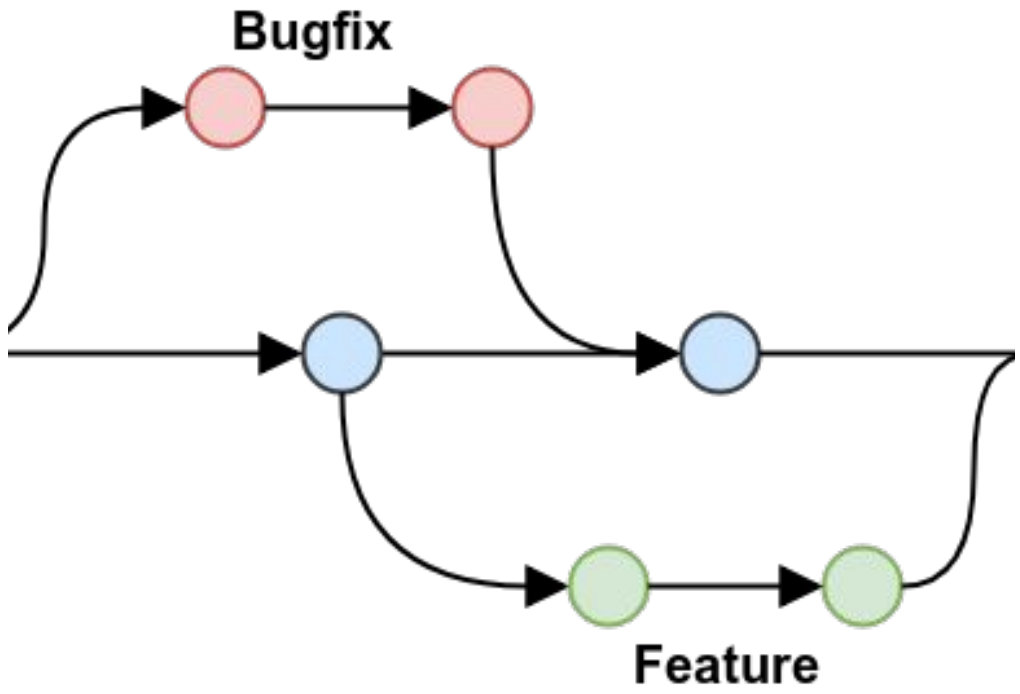
Propose changes to a project by submitting a pull request.

Code Reviews:

Collaborators can review and comment on your changes.

Merging Pull Requests:

Once approved, pull requests can be merged into the main branch.



Best Practices

Commit Often:

Small, frequent commits with clear messages make it easier to track changes.

Use Branches:

Keep your main branch clean by creating feature or bug-fix branches.

Collaborate Effectively:

Regularly pull changes from the main branch to keep your branch up to date.

Review Code Thoroughly:

Engage in code reviews to ensure quality and knowledge sharing.

.