Forking

Fork

- It creates two git repository one private and one public.
- It gives each developer its own server-side repository.
- Forking Workflow is most often seen in public open-source projects.
- Contributions can be integrated without the need for everybody to push to a single central repository.
- Developers push to their own server-side repositories, and only the project maintainer can push to the official repository. This allows the maintainer to accept commits from any developer without giving them write access to the official codebase.

- When a new developer wants to start working on the project, they do not directly clone the official repository.
- Instead, they fork the official repository to create a copy of it on the server.
- This new copy serves as their personal public repository—no other developers are allowed to push to it, but they can pull changes from it.
- After they have created their server-side copy, the developer performs a git clone to get a copy of it onto their local machine. This serves as their private development environment, just like in the other workflows.
- When they're ready to publish a local commit, they push the commit to their own public repository—not the official one. Then, they file a pull request with the main repository, which lets the project maintainer know that an update is ready to be integrated.

Basic flow of fork

- Fork the repository: Click on the fork button on GitHub of your repository. This will create a new repository duplicating the original project.
- Clone your fork: It clone the repository to your local machine.

git clone https://github.com/your-username/forked-repo.git

- Make changes: After cloning the repository, you can navigate into the directory, create branches, and make changes.
- Push changes: After committing your changes, push them back to your forked repository:
- Create a pull request: If you want the original project to incorporate your changes, you can submit a pull request from your forked repository. The maintainers of the original repository can review your changes and decide whether to merge them.

Git hook

- A **Git hook** is a script that Git automatically executes before or after certain events, such as committing, merging, or pushing changes.
- Git hooks are a way to customize Git’s behavior and automate workflows like running tests, formatting code, or enforcing commit message policies.
- There are two groups of these hooks:
- Client-side
- Server-side.