12 Project Forking Workflow

Forking a project to your own namespace is useful if you have no write access to the project you want to contribute to. If you do have write access or can request it, we recommend working together in the same repository since it is simpler.

> The Forking Workflow is fundamentally different than other popular Git workflows. Instead of using a single server-side repository to act as the “central” codebase, it gives every developer their own server-side repository. This means that each contributor has not one, but two Git repositories: a private local one and a public server-side one. The Forking Workflow is most often seen in public open source projects.
>
> The main advantage of the Forking Workflow is that 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.

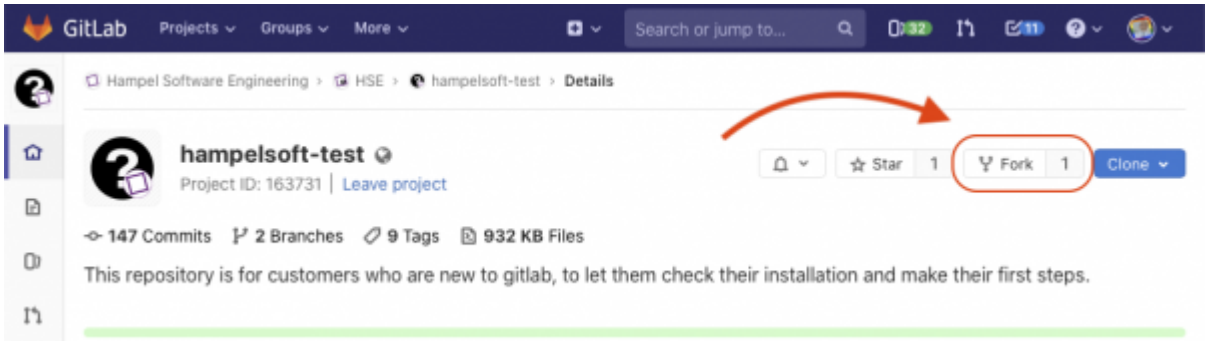
(taken from Atlassian)

“Fork” is not a Git operation — it just means you have made a copy of an existing repository and are doing new development on your copy.

GitLab Documentation

Forking a project is in most cases a two-step process.

1. Click on the fork button located in between the star and clone buttons on the project’s home page.



2. Once you do that, you’ll be presented with a screen where you can choose the namespace to fork to. Only namespaces (groups and your own namespace) where you have write access to, will be shown. Click on the namespace to create your fork there.

(taken from gitlab.com)

Reproducing a forked situation manually

You are the `user`. The guy who created the original repo is the `maintainer`.

```
# clone own private repo (this is remote 'origin')
git clone https://user@gitlab.com/user/repo.git
```

```
# add the public repo as a remote called 'upstream'
git remote add upstream https://user@gitlab.com/maintainer/repo.git
```

```
# Edit some code
git commit -a -m "this is my change"
```

```
# keep my private repo up to date with changes from public repo
git pull upstream master
```

```
# publish my changes to my private repo
git push origin feature-branch
```

...create a merge request and wait for it to be resolved...

```
# update my private repo from the public repo
git pull upstream master
```

Other Resources

- https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow
- https://reflectoring.io/github-fork-and-pull/
- https://blog.seibert-media.net/blog/2014/04/24/git-workflows-der-forking-workflow-teil-1/
- https://blog.seibert-media.net/blog/2014/04/25/git-workflows-der-forking-workflow-teil-2/