

Development with GIT

Step 1: Sign into bitbucket

Step 2: Fork the project repository


Find the project's repository on https://bitbucket.org/Citytech_global , and then "fork" it by clicking the **Fork this repository** link in the upper right corner:

City Tech / Citytech-gitlearn


Citytech-gitflow



Invite Clone ...

Here's where you'll find this repository's source files. To give your users an idea of what they'll find here, [add a description to your repository](#)

 master ▾

Files ▾ Filter files 🔍

 /

Name	Size	Last commit	Message
 .gitignore	624 B	55 minutes ago	Initial commit
 README.md	565 B	55 minutes ago	Initial commit

Fork this repository

Compare branches or tags

Add file



Manage notifications

Download repository

This proceeds to create a copy of the project repository in your bitbucket account. In the page below, you will see that you are now creating a repository in your account. Click "**Fork repository**" to fork.

Fork Citytech_global / Citytech-gitflow

Workspace  Me User 

Project*  my-gitflow 


Name* Citytech-gitflow

Access level ☒ Private repository

▼ Advanced settings

Description

It's encouraged to write a little about why you are forking.

Forking No forks 

Fork repository Cancel

Now you will see the repository in your account.

Me User / my-gitflow

Citytech-gitflow

Invite Clone ...

Here's where you'll find this repository's source files. To give your users an idea of what they'll find here, [add a description to your repository](#).

master Files Filter files

Name	Size	Last commit	Message
.gitignore	624 B	5 hours ago	Initial commit
README.md	565 B	5 hours ago	Initial commit

Repository details

Last updated 4 hours ago

Open pull requests 0 Branches 2

Watchers 1 Forks 0

Version control system Git Language Java

Access level Admin

Fork of [Citytech_global/citytech-gitflow](#)

Step 3: Clone your fork

From your repository, click the “Clone” button at the top right.

Me User / my-gitflow

Citytech-gitflow

Invite Clone ...

Here's where you'll find this repository's source files. To give your users an idea of what they'll find here, [add a description to your repository](#).

Clone this repository

HTTPS

```
git clone https://meuser@bitbucket.org/meuser/citytech-gitflow.git
```



Sourcetree is a free Git client for macOS.

Clone in Sourcetree

VS Code is a source-code editor developed by Microsoft.

Clone in VS Code

Close

Using Git on your local machine, clone your fork using the URL you just copied:

git clone URL_OF_FORK

For example, I used

git clone <https://meuser@bitbucket.org/meuser/citytech-gitflow.git>

```
meuser@citytech workspace % git clone https://meuser@bitbucket.org/meuser/citytech-gitflow.git
Cloning into 'citytech-gitflow'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 974 bytes | 243.00 KiB/s, done.
```

Cloning copies the repository files (and commit history) from bitbucket to your local machine. The repository will be downloaded into a subdirectory of your working directory, and the subdirectory will have the same name as the repository.

Step 4: Navigate to your local repository

Since the clone was downloaded into a subdirectory of your working directory, you can navigate to it using: **cd NAME_OF_REPOSITORY**

For example, I used **cd citytech-gitflow**

```
meuser@citytech workspace % ls
citytech-gitflow
meuser@citytech workspace % cd citytech-gitflow
meuser@citytech citytech-gitflow %
```

Step 5: Check that your fork is the “origin” remote

You are going to be synchronizing your local repository with both the project repository and your fork. The URLs that point to these repositories are called “remotes”. More specifically, the project repository is called the “upstream” remote, and your fork is called the “origin” remote.

When you cloned your fork, that should have automatically set your fork as the “origin” remote. Use **git remote -v** to show your current remotes. You should see the URL of your fork (which you copied in step 3) next to the word “origin”.

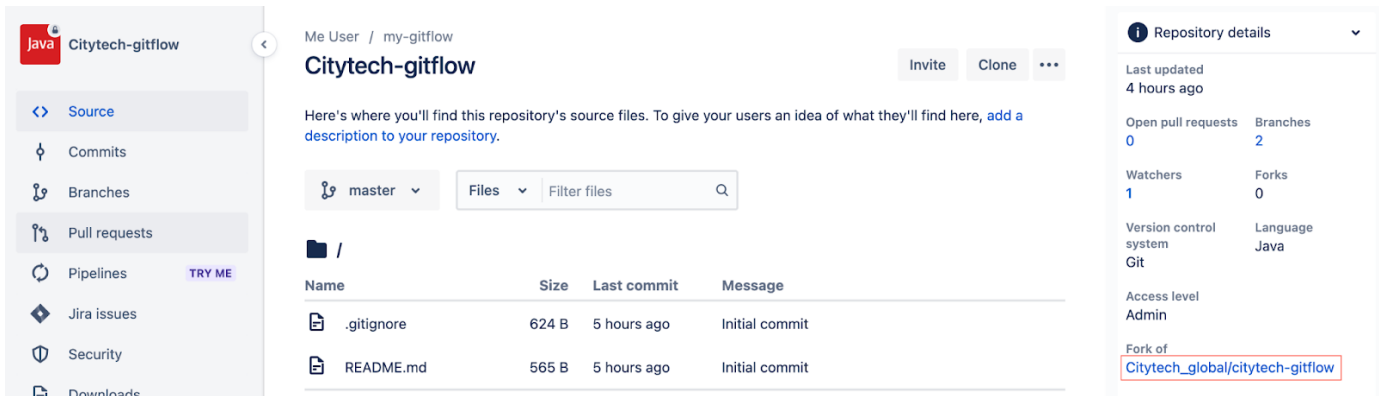
```
meuser@citytech citytech-gitflow % git remote -v
origin https://meuser@bitbucket.org/meuser/citytech-gitflow.git (fetch)
origin https://meuser@bitbucket.org/meuser/citytech-gitflow.git (push)
meuser@citytech citytech-gitflow %
```

If you don't see an “origin” remote, you can add it using:

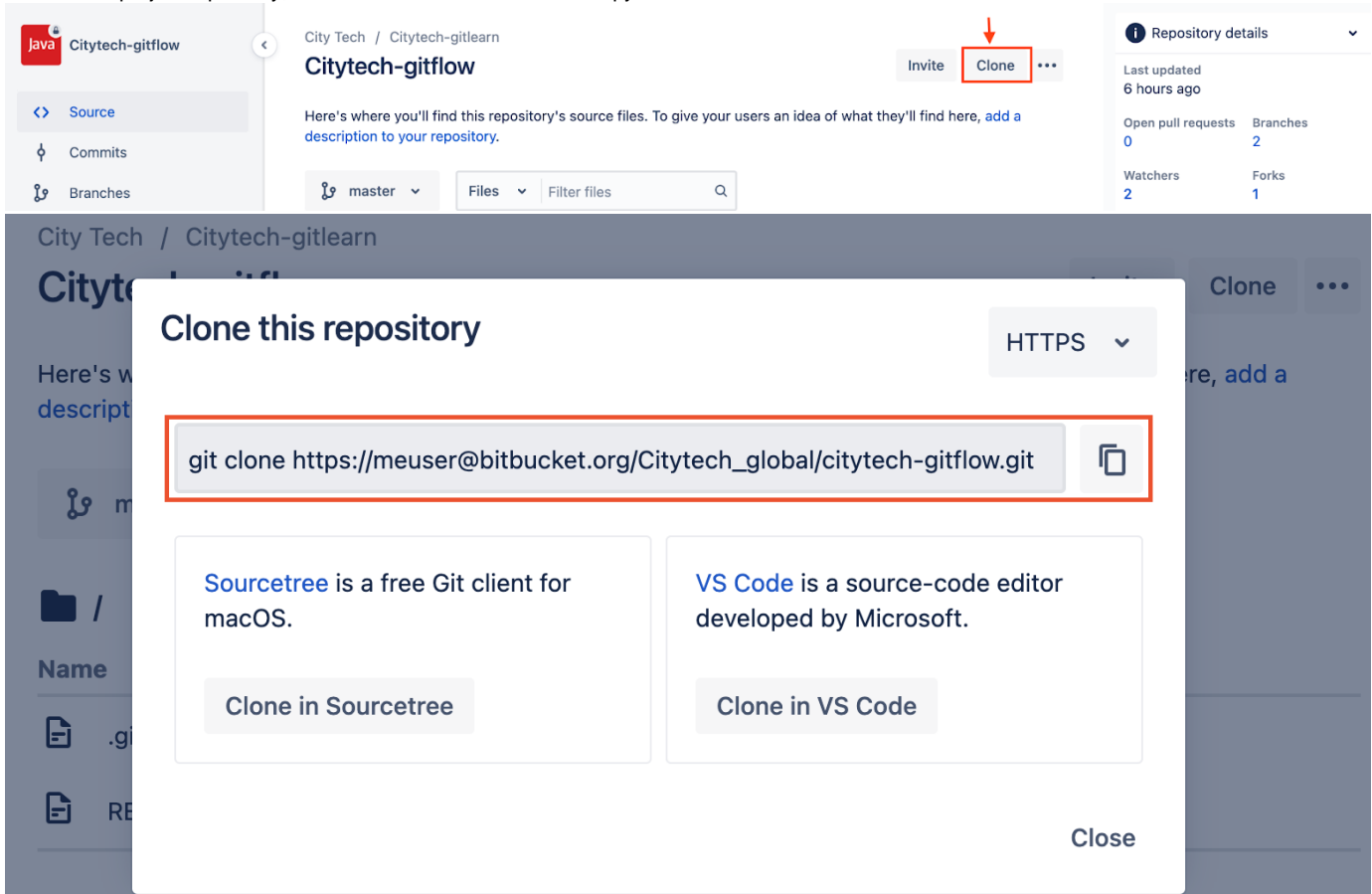
git remote add origin URL_OF_FORK

Step 6: Add the project repository as the “upstream” remote

Go to your fork on bitbucket, and click the “Fork of” link to return to the project repository:



While in the project repository, click the Clone button and then copy the HTTPS URL:



Add the project repository as the "upstream" remote using:

git remote add upstream URL_OF_PROJECT

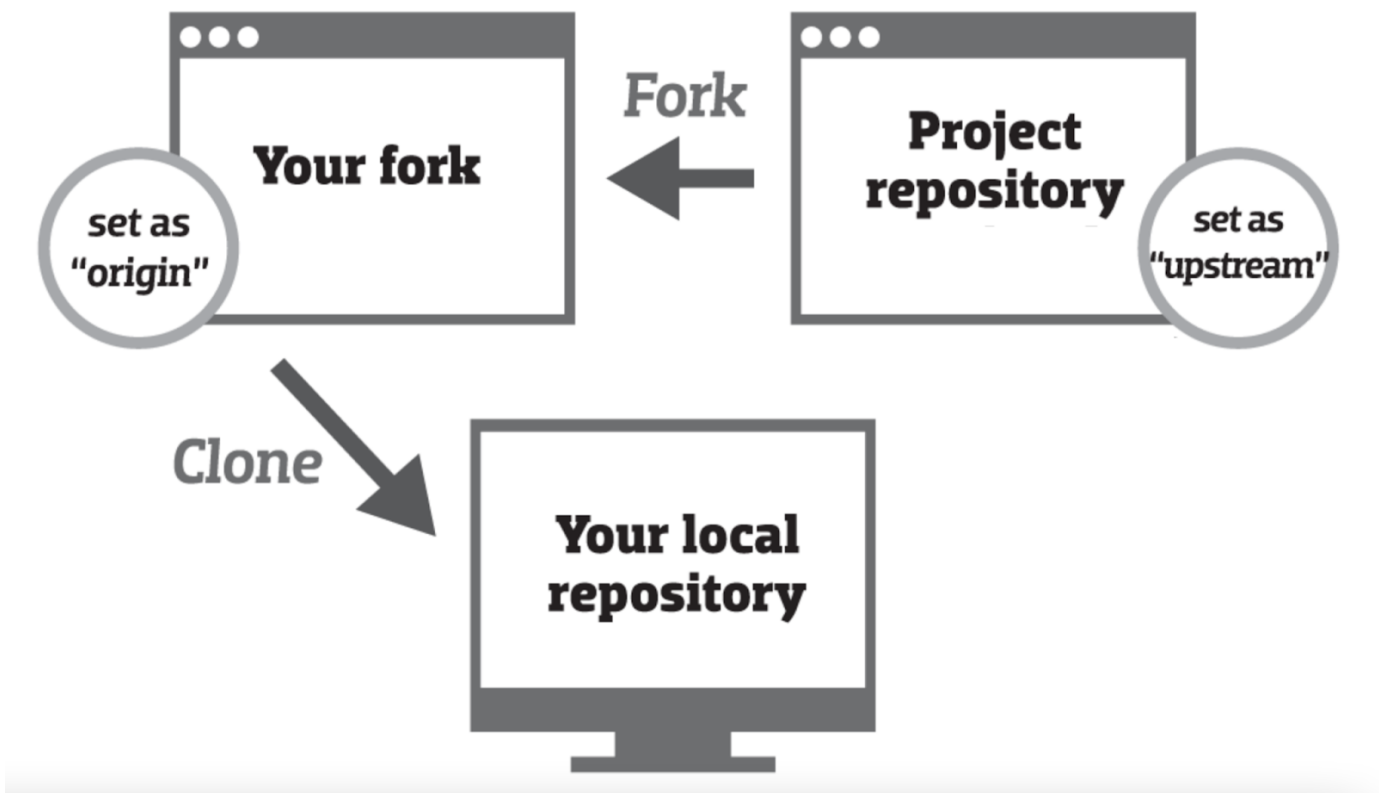
For example, I used

git remote add upstream https://meuser@bitbucket.org/Citytech_global/citytech-gitflow.git

Use git remote -v to check that you now have two remotes: an origin that points to your fork, and an upstream that points to the project repository.

```
meuser@citytech citytech-gitflow % git remote add upstream https://meuser@bitbucket.org/Citytech_global/citytech-gitflow.git
meuser@citytech citytech-gitflow % git remote -v
origin https://meuser@bitbucket.org/meuser/citytech-gitflow.git (fetch)
origin https://meuser@bitbucket.org/meuser/citytech-gitflow.git (push)
upstream https://meuser@bitbucket.org/Citytech_global/citytech-gitflow.git (fetch)
upstream https://meuser@bitbucket.org/Citytech_global/citytech-gitflow.git (push)
meuser@citytech citytech-gitflow %
```

This diagram summarizes the entire setup process (steps 1 through 6):



Step 7: Pull the latest changes from upstream into your local repository

Before you start making any changes to your local files, it's a good practice to first synchronize your local repository with the project repository. Use **git pull upstream master** to "pull" any changes from the "master" branch of the "upstream" into your local repository. (If the project repository uses "dev" instead of "master" for its default branch, then you would use **git pull upstream dev** instead.)

If you forked and cloned the project repository just a few minutes ago, it's very unlikely there will be any changes, in which case Git will report that your local repository is "already up to date". But if there are any changes, they will automatically be merged into your local repository.

Step 8: Create a new branch

Rather than making changes to the project's "master" branch, it's a good practice to instead create your own branch. This creates an environment for your work that is isolated from the master branch.

Use **git checkout -b BRANCH_NAME** to create a new branch and then immediately switch to it. The name of the branch should briefly describe what you are working on, and should not contain any spaces.

For example, I used **git checkout -b feature/PRJ-1** because I am going to work on the feature PRJ-1.

Use **git branch** to show your local branches. You should see your new branch as well as "master", and your new branch should have an asterisk next to it to indicate that it's "checked out" (meaning that you're working in it).

```
meuser@citytech citytech-gitflow % git checkout -b feature/PRJ-1
Switched to a new branch 'feature/PRJ-1'
meuser@citytechcitytech-gitflow % git branch
* feature/PRJ-1
  master
meuser@citytech citytech-gitflow %
```

Step 9: Make changes in your local repository

Use a text editor or IDE to make the changes you planned to the files in your local repository. Because you checked out a branch in the previous step, any edits you make will only affect that branch.

Step 10: Commit your changes

After you make a set of changes, use **git add -A** to stage your changes and **git commit -m "DESCRIPTION OF CHANGES"** to commit them.

For example, I used **git commit -m "PRJ-1 prepare add login form"** for one of my commits.

If you are making multiple sets of changes, it's a good practice to make a commit after each set.

Step 11: Push your changes to your fork

When you are done making all of your changes, upload these changes to your fork using **git push origin BRANCH_NAME**. This "pushes" your changes to the "**BRANCH_NAME**" branch of the "origin" (which is your fork on bitbucket).

For example, I used **git push origin feature/PRJ-1**

This terminal snapshot below summarizes the entire process (steps 9 through 11):

```
meuser@citytech citytech-gitflow % ls
README.md
meuser@citytech citytech-gitflow % vi login.html
meuser@citytech citytech-gitflow % ls
README.md      login.html
meuser@citytech citytech-gitflow % git add -A
meuser@citytech citytech-gitflow % git commit -m "PRJ-1 prepare add login form"
[feature/PRJ-1 78ad81c] PRJ-1 prepare add login form
Committer: Me User <meuser@citytech.local>

1 file changed, 2 insertions(+)
create mode 100644 login.html
meuser@citytech citytech-gitflow % git push origin feature/PRJ-1
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 337 bytes | 337.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create pull request for feature/PRJ-1:
remote:   https://bitbucket.org/meuser/citytech-gitflow/pull-requests/new?source=feature/PRJ-1&t=1
remote:
To https://bitbucket.org/meuser/citytech-gitflow.git
 * [new branch]      feature/PRJ-1 -> feature/PRJ-1
meuser@citytech citytech-gitflow %
```

Step 12: Begin the pull request

Return to your fork on bitbucket, and refresh the **"Branches"** page. You will see your recently pushed branch:

Me User / my-gitflow / Citytech-gitflow

Branches

Search branches Active branches Branch type

Branch	Behind	Ahead	Updated	Pull request	Builds	Actions
master MAIN DEVELOPMENT			7 hours ago			...
feature/PRJ-1	0	1	2 minutes ago	Create		...

Click the blue **Create** button to begin the pull request.

Step 13: Create the pull request

When opening a "pull request", you are making a "request" that the project repository "pull" changes from your fork. You will see that the project repository is listed at the left as the **base repository**, and your fork is listed at the right as the **head repository**:

Me User / my-gitflow / Citytech-gitflow / Pull requests

Create a pull request

Base Repository

meuser / Citytech-gitflow
Created 5 hours ago, updated 23 minutes ago

feature/PRJ-1

Head Repository

Citytech_global/citytech-gitflow
dev

Title PRJ-1 prepare add login form

Description

Attachments [Browse to upload](#)

Reviewers ReviewerUser

Delete branch ☐ Delete feature/PRJ-1 after the pull request is merged

[Create pull request](#)

Diff [Commits](#)

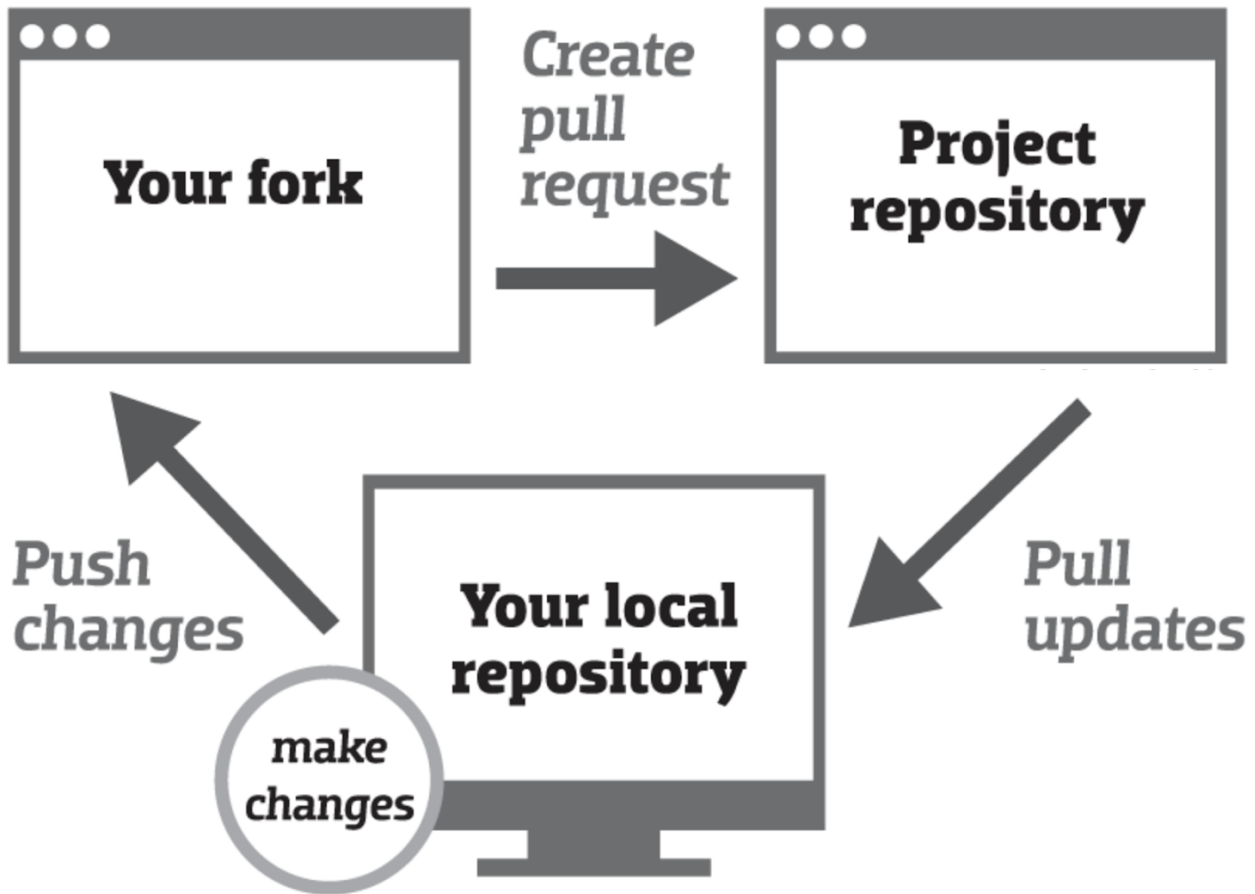
Author	Commit	Message	Date	Builds
Me User	78ad81c	PRJ-1 prepare add login form	24 minutes ago	

Before submitting the pull request, you first need to describe the changes you made (rather than asking the project maintainers to figure them out on their own). You should write a descriptive title for your pull request, and then include more details in the body of the pull request. If there are any related **JIRA** issues, make sure to mention those by number.

Below the pull request form, you will see a list of the commits you made in your branch, as well as the "diffs" for all of the files you changed.

If everything looks good, click the **Create pull request** button.

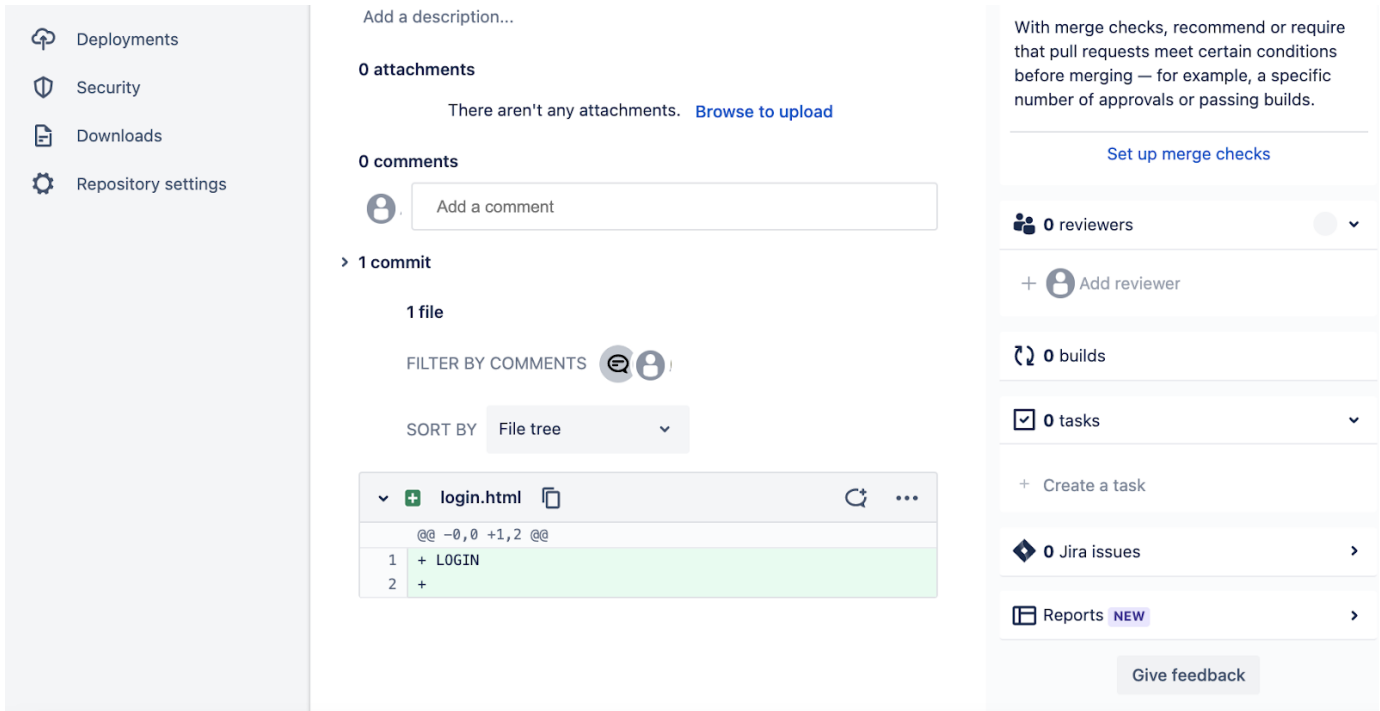
This diagram summarizes the entire pull request process (steps 7 through 13):



Step 14: Review the pull request

You have now created a pull request, which is stored in the project's repository (not in your fork of the repository). It's a good idea to read through what you wrote, as well as clicking on the Commits tab and the Files changed tab to review the contents of your pull request:

The screenshot shows the GitHub pull request page for 'Citytech-gitflow'. The left sidebar contains navigation links: Source, Commits, Branches, Pull requests (selected), and Pipelines. The main content area shows the pull request title 'PRJ-1 prepare add login form', the source branch 'meuser/citytech-gitflow:feat...', and the target branch 'Citytech_global/citytech-gitf...'. It includes an 'OPEN' status, creation and update timestamps, and buttons for 'Edit', 'Approve', 'Merge', and 'Settings'. The description field is visible. On the right, the 'Details' tab is active, showing '0 checks' and a diagram indicating that merge checks are not yet set up in the target branch.



If you realize that you left out some important details, you can click the **Edit** button in the upper left corner to edit your pull request description.

Step 15: Add more commits to your pull request

You can continue to add more commits to your pull request even after opening it. For example, the reviewer may ask you to make some changes, or you may just think of a change that you forgot to include:

Start by returning to your local repository, and use **git branch** to see which branch is currently checked out. If you are currently in the master branch (rather than the branch you created), then use **git checkout BRANCH_NAME** to switch. For example, I used **git checkout feature/PRJ-1**.

Then, you should repeat steps 9 through 11: make changes, commit them, and push them to your fork.

Finally, return to your open pull request on bitbucket and refresh the page. You will see that your new commits have automatically been added to the pull request:

Step 16: Delete your branch from your fork

If the reviewers accept your pull request (congratulations!), they will merge your proposed changes into the project's main branch and close your pull request.

You will be given the option to delete your branch from your fork, since it's no longer of any use.

Pull request has been merged.

Click the Delete branch button.

Step 17: Delete your branch from your local repository

You should also delete the branch you created from your local repository, so that you don't accidentally start working in it the next time you want to make a new changes to this project.

First, switch to the master branch: `git checkout master`.

Then, delete the branch you created: **`git branch -D BRANCH_NAME`**. For example, I used **`git branch -D feature/PRJ-1`**.

Step 18: Synchronize your fork with the project repository

At this point, your fork is out of sync with the project repository's main branch.

To get it back in sync, you should first use Git to pull the latest changes from "upstream" (the project repository) into your local repository:

`git pull upstream master`.

Then, push those changes from your local repository to the "origin" (your fork):

`git push origin master`.

If you return to your fork on bitbucket, you will see that the master branch is "even" with the project repository's master branch:

Fork is even with the project repository.

This step is not strictly necessary, since you will pull changes from upstream before you make your next contribution to this project (step 7). However, this step is useful if you are going to clone your fork from another machine.

Congratulations!

Congratulations on making your changes available on the project!

References :

<https://www.dataschool.io/how-to-contribute-on-github/>

<https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/collaborating-on-repositories-with-code-quality-features/about-status-checks>