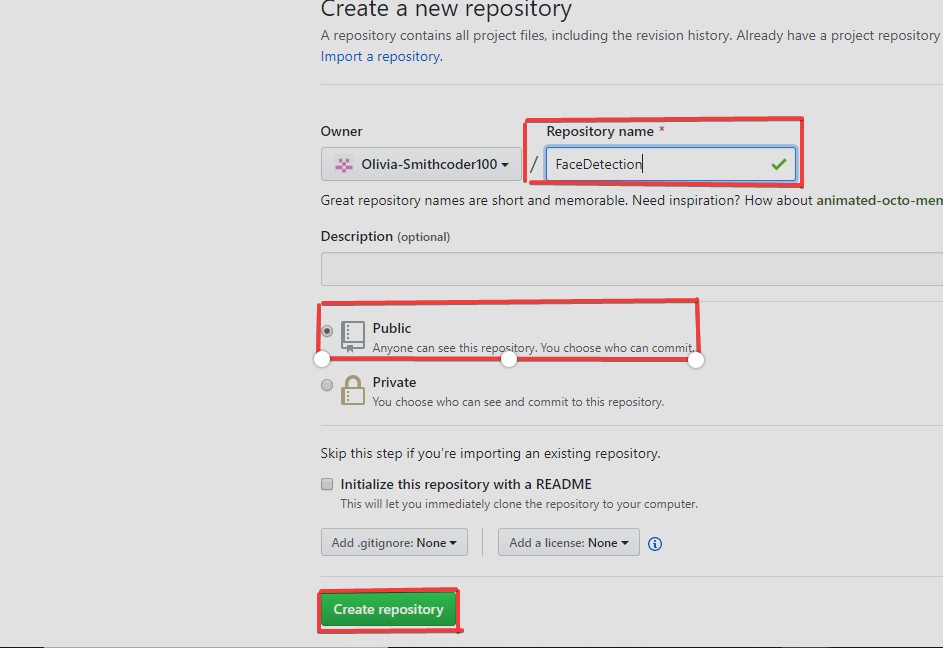
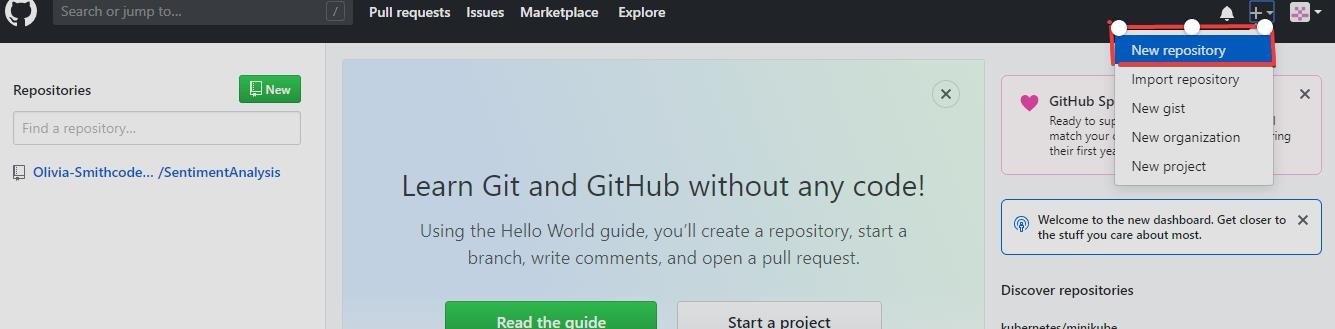
GIT Push and Pull

# Using Command line to PUSH to GitHub

## Creating a new repository

* + You need to create a new repository and click on the plus sign.
  + Fill up all the required details, i.e., repository name, description and also make the repository public this time as it is free.



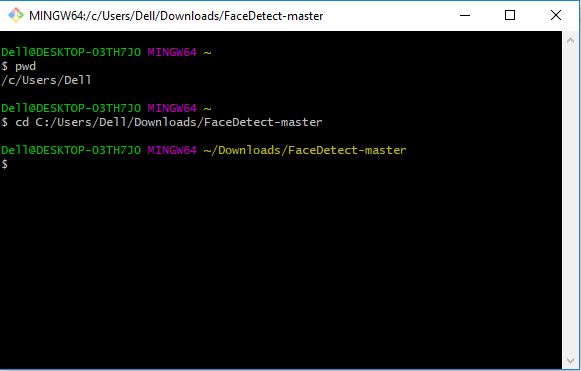
## Open your Git Bash

* + Git Bash can be downloaded in [here,](https://git-scm.com/downloads) and it is a shell used to interface with the operating system which follows the UNIX command.

## Create your local project in your desktop directed towards a current working directory

* + pwd stands for 'print working directory', which is used to print the current directory.
  + Move to the specific path in your local computer by cd 'path\_name'. The cd commands stand for 'change directory' and it is used to change to the working directory in your operating system, and to locate your file, 'path\_name', i.e.,

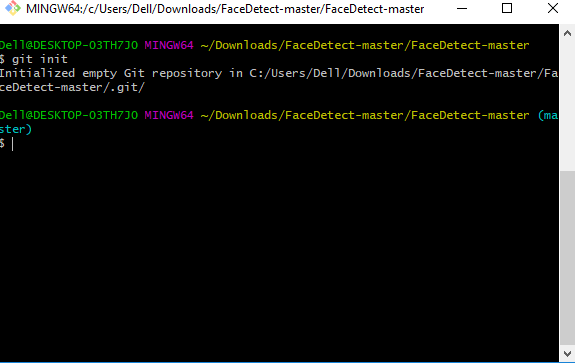
C:/Users/Dell/Downloads/FaceDetect-master needs to be given. This command can identify the required file that you are looking to work with.



## Initialize the git repository

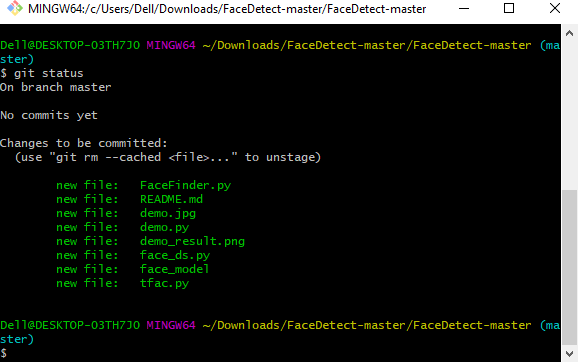
* + Use git init to initialize the repository. It is used to create a new empty repository or

directory consisting of files' with the hidden directory. '.git' is created at the top level of your project, which places all of the revision information in one place.



## Add the file to the new local repository

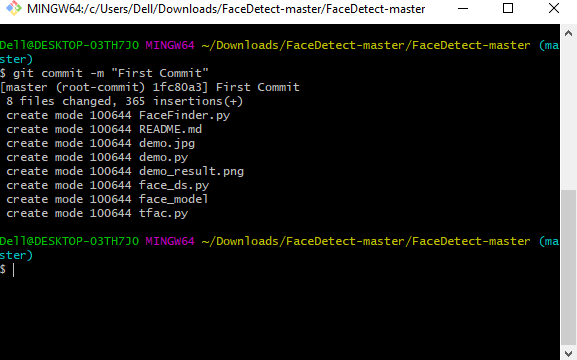
* + Use git add . in your bash to add all the files to the given folder.
  + Use git status in your bash to view all the files which are going to be staged to the first commit.



## Commit the files staged in your local repository by writing a commit message

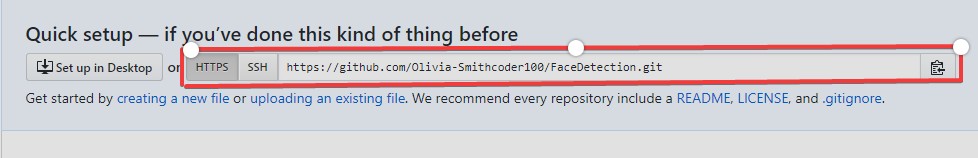
* + You can create a commit message by git commit -m 'your message', which adds the change to the local repository.
  + git commit uses '-m' as a flag for a message to set the commits with the content where the full description is included, and a message is written in an imperative sentence up to 50 characters long and defining "what was changed", and "why was the change

made".



## Copy your remote repository's URL from GitHub

* + The HTTPS or URL is copied from the given GitHub account, which is the place of the remote repository.



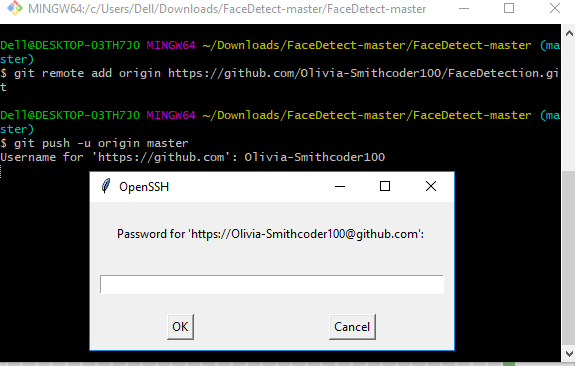
## Add the URL copied, which is your remote repository to where your local content from your repository is pushed

* + git remote add origin 'your\_url\_name'
  + In the above code, The 'origin' is the remote name, and the remote URL is

"<https://github.com/Olivia-Smithcoder100/FaceDetection.git>". You can see the remote as GitHub in this case, and GitHub provides the URL for adding to the remote repository.

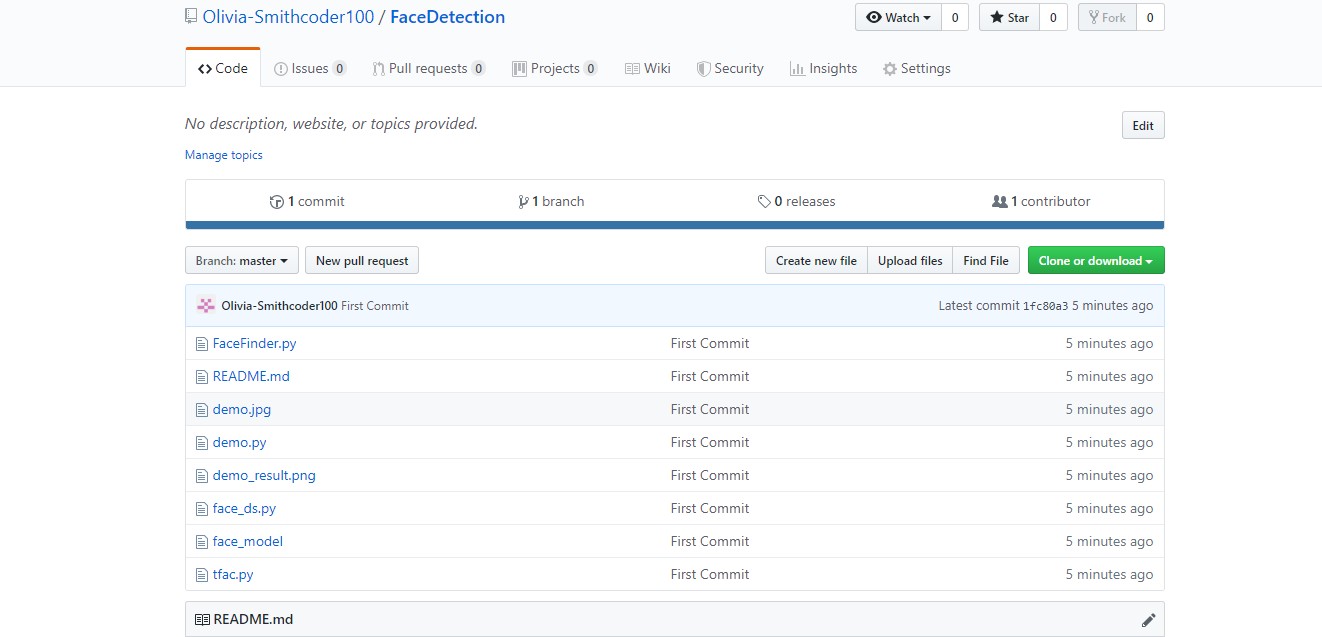
## Push the code in your local repository to GitHub

* + git push -u origin master is used for pushing local content to GitHub.
  + In the code, the origin is your default remote repository name and '-u' flag is upstream, which is equivalent to '-set-upstream.' and the master is the branch, name.upstream is the repository that we have cloned the project.
  + Fill in your GitHub username and password.



## View your files in your repository hosted on GitHub

* + You can finally see the file hosted on GitHub.



# PULL Request

## If you make a change in a repository, GIT PULL can allow others to view the changes. It is used to acknowledge the change that you've made to the repository that you're working on. Or also called a target repository.

The simple command to PULL from a branch is:

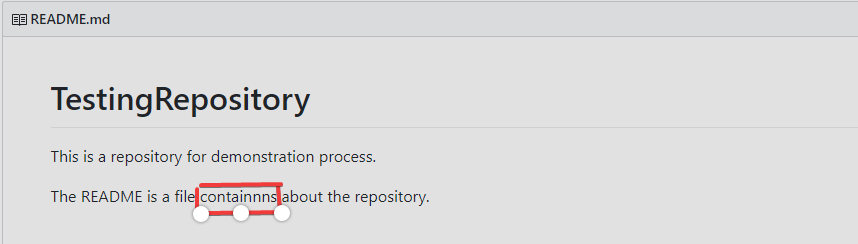
git pull 'remote\_name' 'branch\_name'.

## The git pull command is a combination of git fetch which fetches the recent commits in the local repository and git merge, which will merge the branch from a remote to a local branch also 'remote\_name' is the repository name and 'branch\_name' is the name of the specific branch.

You"ll be looking at two different ways on how to use the PULL request.

# PULL Request through Command Line

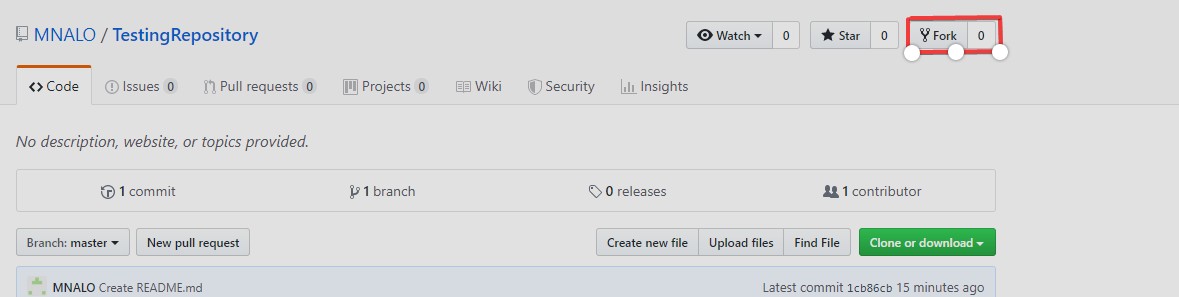
## You can see the README files below which contains a typo. The README file has the word "contain" misspelled as "containnns". The owner of this repository is MNALO, and Olivia is the collaborator. She will solve the error and submit a PULL Request You'll see the process for making a PULL Request through a particular example given below.



In the file above, you can see a typo in the word "containnns".

## Fork the Repository

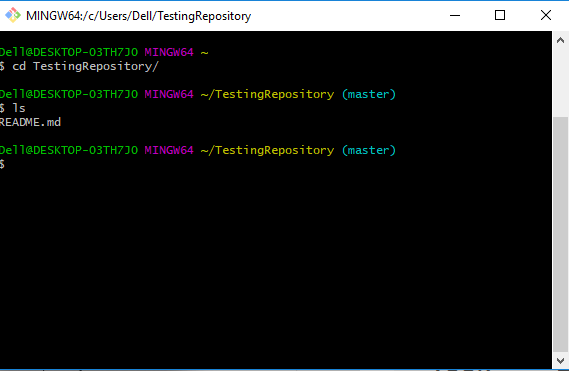
* + "The "Fork" is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project."[(Source)](https://docs.github.com/en/get-started/quickstart/fork-a-repo)



## Open your bash in your computer

* + You need to move to the required path or folder by using the cd command, and the

content can be viewed by using the ls command, which will list all of the present files in the directory and in our case you can see the 'README.md' is present.



## Make a new branch

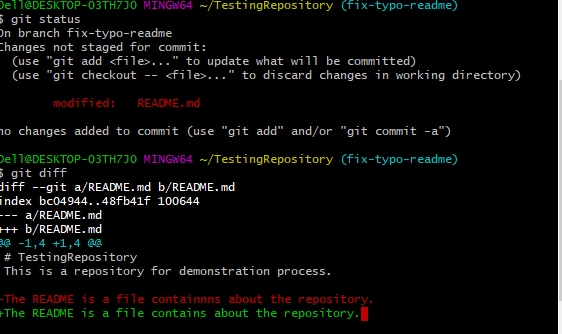
* + You can create a new branch by using the git checkout -b 'branch\_name'. In the above code, '-b' flag is used to create a new branch, and 'branch\_name' is used to give the

branch a specific name, and with checkout, the branch is switched to the newly created branch.

Make a new branch.

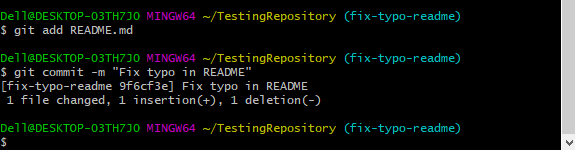
## Make a change by using vim from bash or direct replacement from the original README file

o You can change the word "containnns" to "contains" in the README file, and the changes with the current status can be viewed by using the following command.



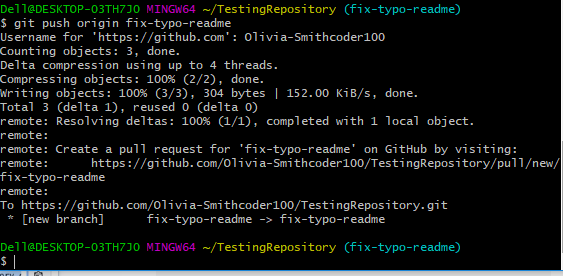
## Adding and Committing a file to the repository

* You need to add and commit by the following commands.



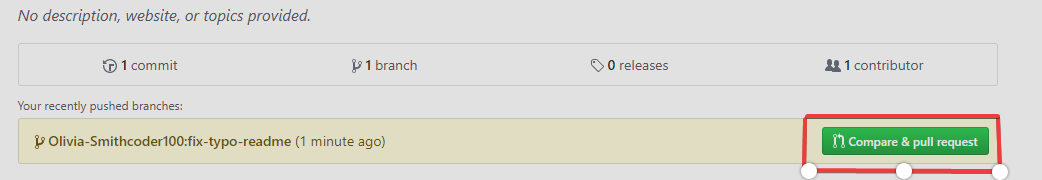
## Push the repository to the GitHub

* You need to push the content by git push origin 'branch\_name'
* In the above code, the origin is the remote repository, and 'branch\_name' is the required branch that you need to upload your local content.



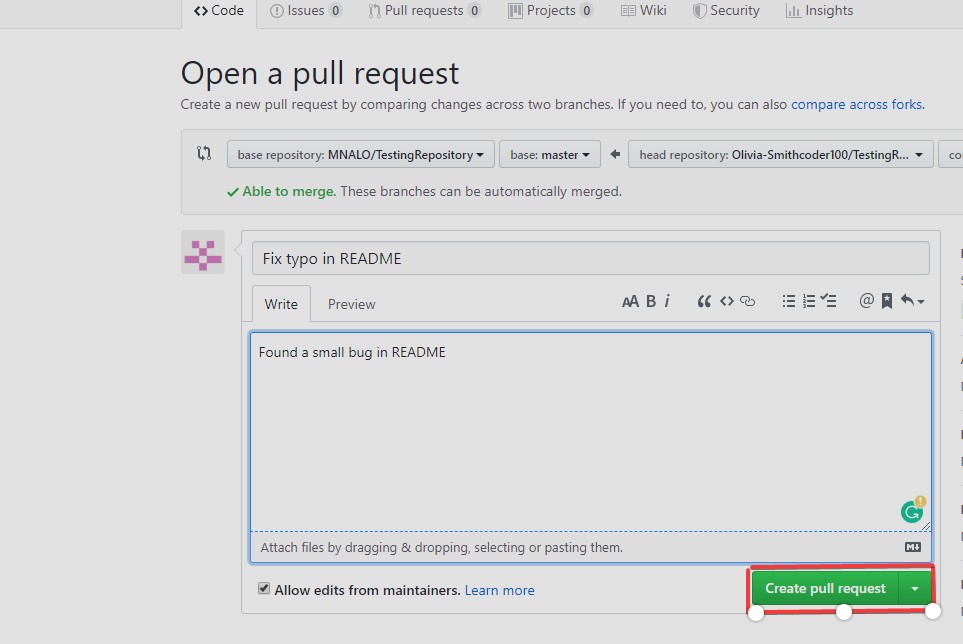
## PULL request for a specific branch on GitHub

* You can move to your repository in GitHub and see that there is a new branch.
* You can now move to step 8, but there is a need for a local repository update with the upstream repository, read this detailed blog on [How To Create a Pull](https://www.digitalocean.com/community/tutorials/how-to-create-a-pull-request-on-github) [Request on GitHub](https://www.digitalocean.com/community/tutorials/how-to-create-a-pull-request-on-github)
* Alternatively, you can do git pull-request in the command line and complete the PULL Request to GitHub, where it will force push your current branch to a remote repository.



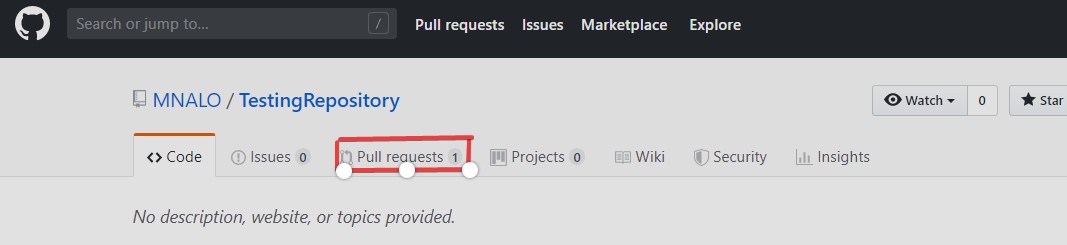
## Open a Pull request

* You need to click the button on "Create pull request," to finish the action.

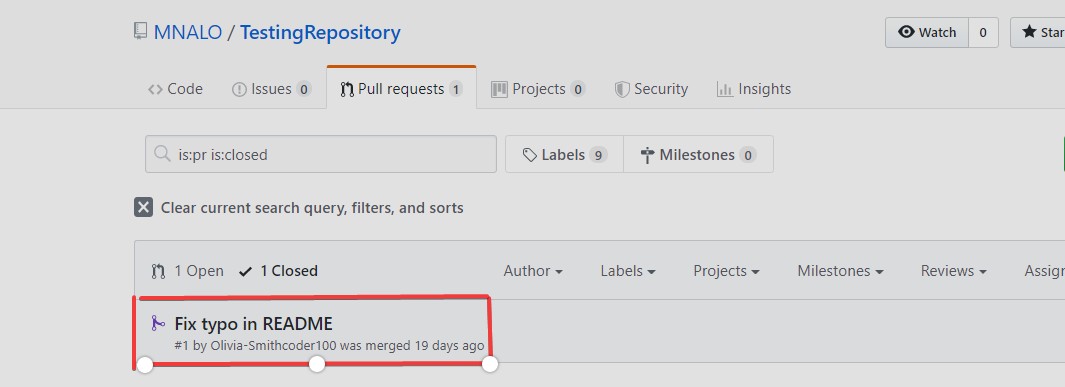


# Deleting a Branch after the PULL Request is Merged

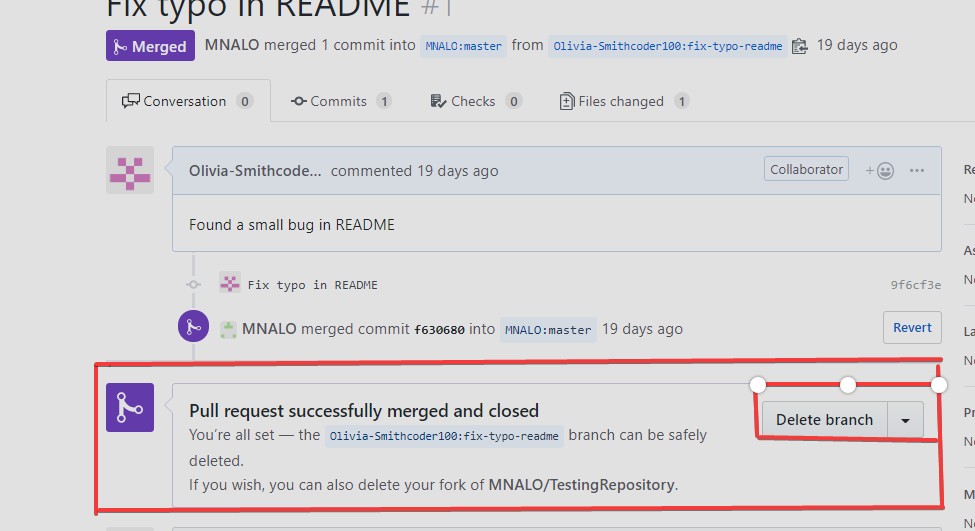
* You need to move to the main page of the repository and click "Pull requests".



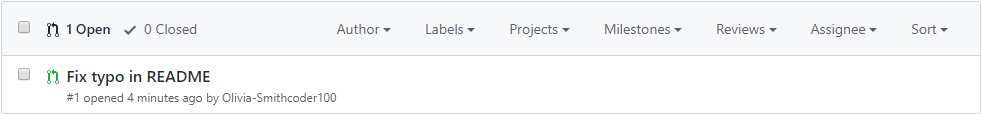
* + You need to click 'Closed' to see the lists of all the PULL Requests that you've made, but there is only one at the moment which needs to be selected. It is the one related to your branch that you want to delete.



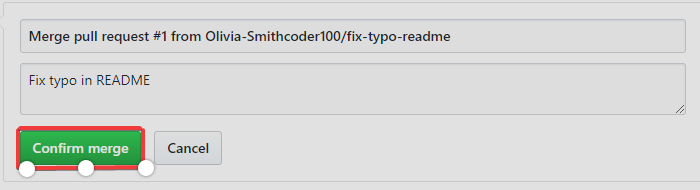
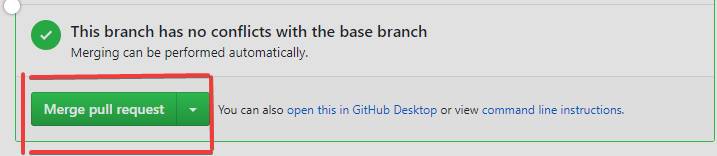
* + - You can now click 'Delete branch' to complete the action.



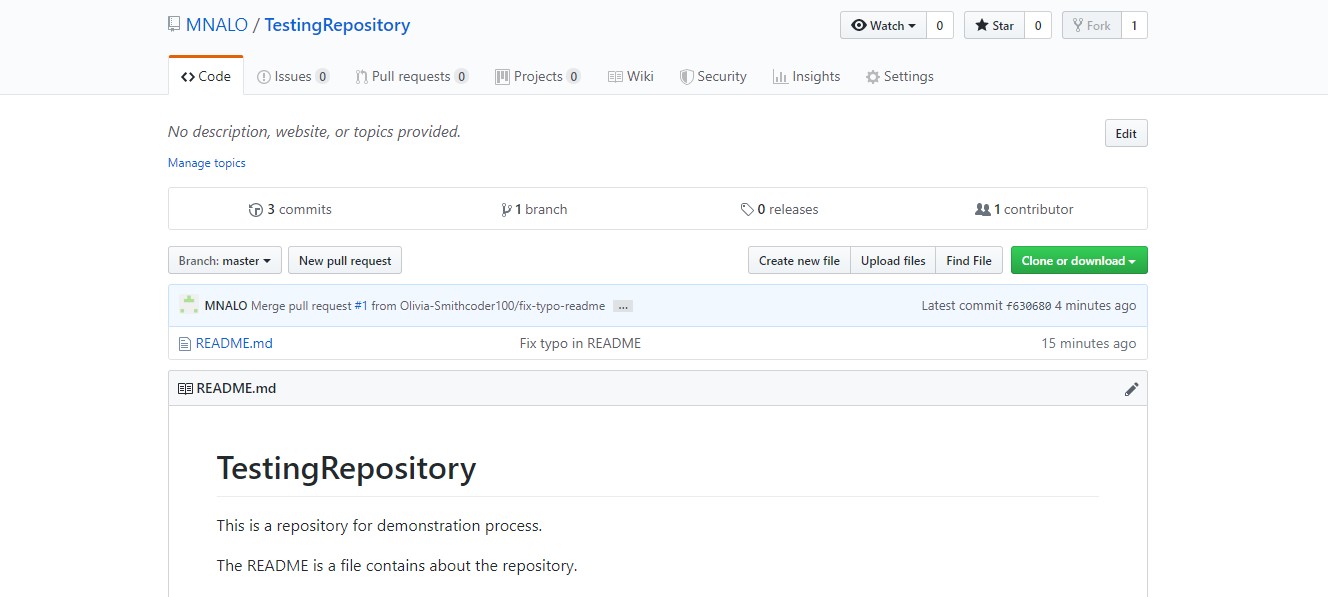
The owner of the repository can view all the commits, pull request, etc., made by collaborators and others. The changes made by someone can be significant, quick fixes for a bug, errors, etc., and are added to the project.



The owner now clicks "Merge pull request". Also, he/she will click "Confirm merge" through the following process.



The last change made to the README.md file with a corrected typo is below.



**Creating and deleting branches within your repository**

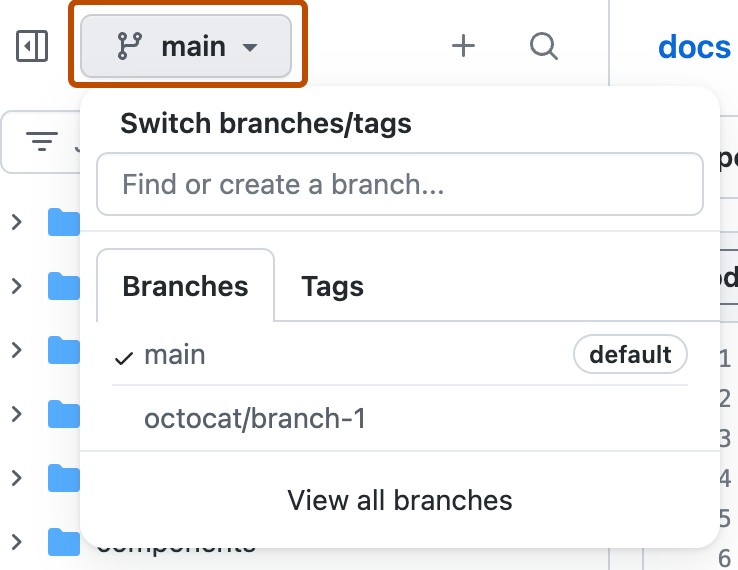
# [Creating a branch](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-and-deleting-branches-within-your-repository#creating-a-branch)

You can create a branch in different ways on GitHub.

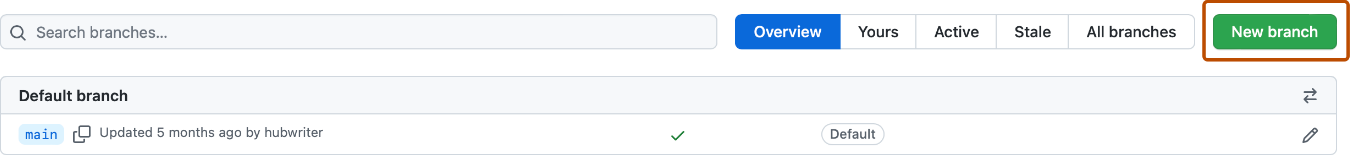
**Note:** You can only create a branch in a repository to which you have push access.

[Creating a branch via the branches overview](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#creating-a-branch-via-the-branches-overview)

1. On GitHub.com, navigate to the main page of the repository.
2. From the file tree view on the left, select the
3. branch dropdown menu, then click **View all branches**. You can also find the branch dropdown menu at the top of the integrated file editor.



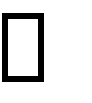
1. Click **New branch**.

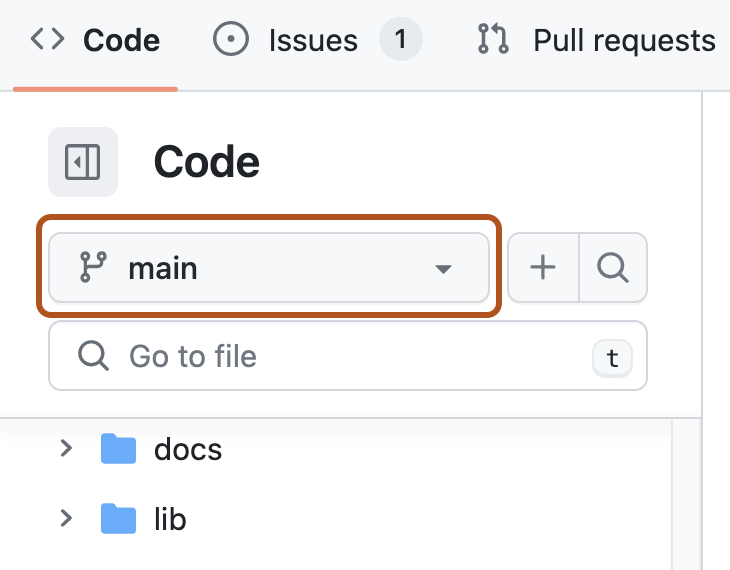


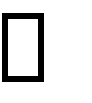
1. Under "Branch name", type a name for the branch.
2. Under "Branch source", choose a source for your branch.
   * If your repository is a fork, select the repository dropdown menu and click your fork or the upstream repository.
   * Select the branch dropdown menu and click a branch.
3. Click **Create branch**.

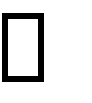
[Creating a branch using the branch dropdown](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-and-deleting-branches-within-your-repository#creating-a-branch-using-the-branch-dropdown)

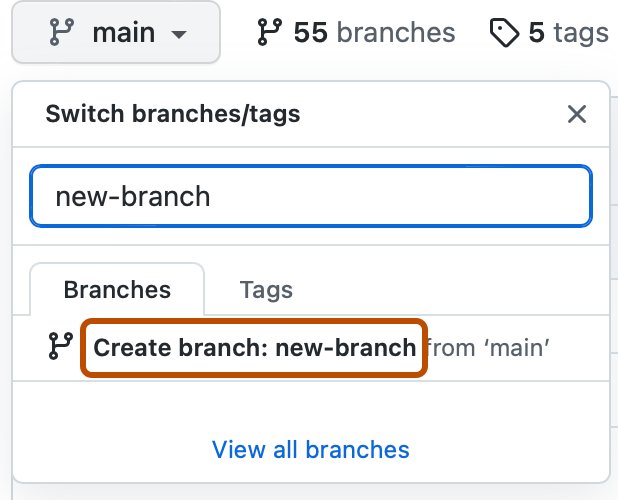
1. On GitHub.com, navigate to the main page of the repository.
2. Select the

branch dropdown menu, in the file tree view or at the top of the integrated file editor.



Optionally, if you want to create the new branch from a branch other than the default branch of the repository, click another branch, then select the branch dropdown menu again.

In the "Find or create a branch..." text field, type a unique name for your new branch, then click **Create branch**.



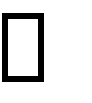
# [Deleting a branch](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-and-deleting-branches-within-your-repository#deleting-a-branch)

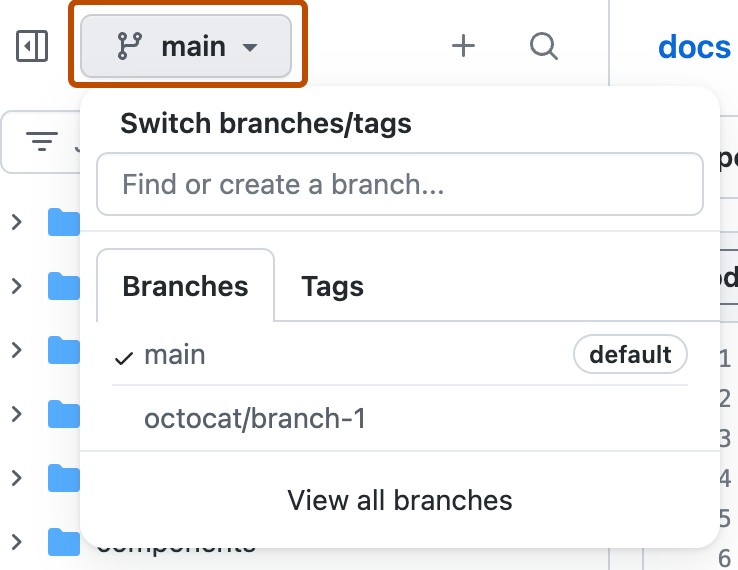
You can have head branches automatically deleted after pull requests are merged in your repository. For more information, see "[Managing the automatic deletion of branches](https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/configuring-pull-request-merges/managing-the-automatic-deletion-of-branches)."

**Note:** If the branch you want to delete is the repository's default branch, you must choose a new default branch before deleting the branch. For more information, see "[Changing the default](https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-branches-in-your-repository/changing-the-default-branch) [branch](https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-branches-in-your-repository/changing-the-default-branch)."

If the branch you want to delete is associated with an open pull request, you must merge or close the pull request before deleting the branch. For more information, see "[Merging a pull](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/merging-a-pull-request) [request](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/merging-a-pull-request)" or "[Closing a pull request](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/closing-a-pull-request)."

1. On GitHub.com, navigate to the main page of the repository.
2. From the file tree view on the left, select the

branch dropdown menu, then click **View all branches**. You can also find the branch dropdown menu at the top of the integrated file editor.



## Next to the branch that you want to delete, click

Screenshot of a branch in the branch list. A trash icon is highlighted with an orange outline.

1. If the branch is associated with at least one open pull request, deleting the branch will close the pull requests. Read the warning, then click **Delete**.

If you delete a head branch after its pull request has been merged, GitHub checks for any open pull requests in the same repository that specify the deleted branch as their base branch. GitHub automatically updates any such pull requests, changing their base branch to the merged pull request's base branch.

# How to Create a Branch in Git

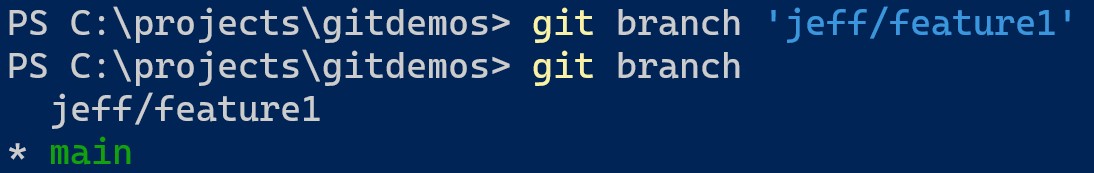
Enough theory, let’s create some branches! These examples will be using PowerShell 7 on a

Windows 10 system; however, you can use any terminal that supports Git commands.

## Option 1: Creating a Branch

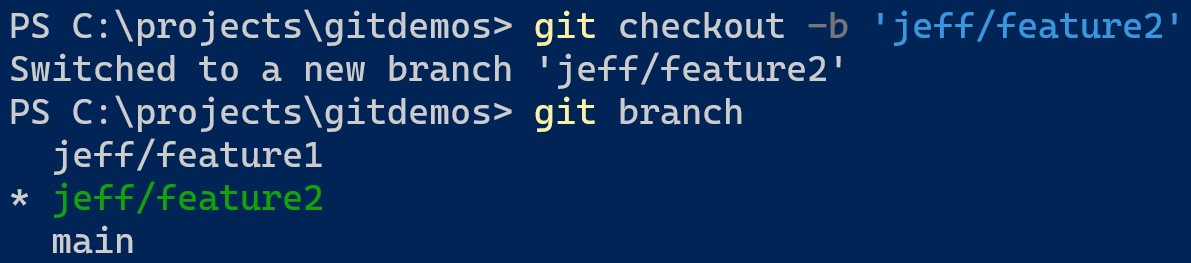
To create a branch, use the git branchcommand followed by the name of the branch. After making the branch, usegit branchagain to view available branches.

Notice that creating a branch this way does not automatically switch to the new branch. Git uses an asterisk and a different colored font to identify which branch is active. This designation represents the HEAD pointer showing which branch is active.

git branch <branch name> git branch

## Option 2: Creating a Branch using Checkout

If you want to create a branch and checkout the branch simultaneously, use the git checkoutcommand. The switch -b specifies the name of the branch. Note that after command completion, Git has moved HEAD to the new branch.

git checkout -b <branch name> git branch

## Option 3: Creating a Branch from a Commit

You can create a branch from a previous commit on an existing branch. Remember, a commit is just a snapshot in time of the files in a repository. You create a branch from a commit if you want to work on a specific snapshot of the files.

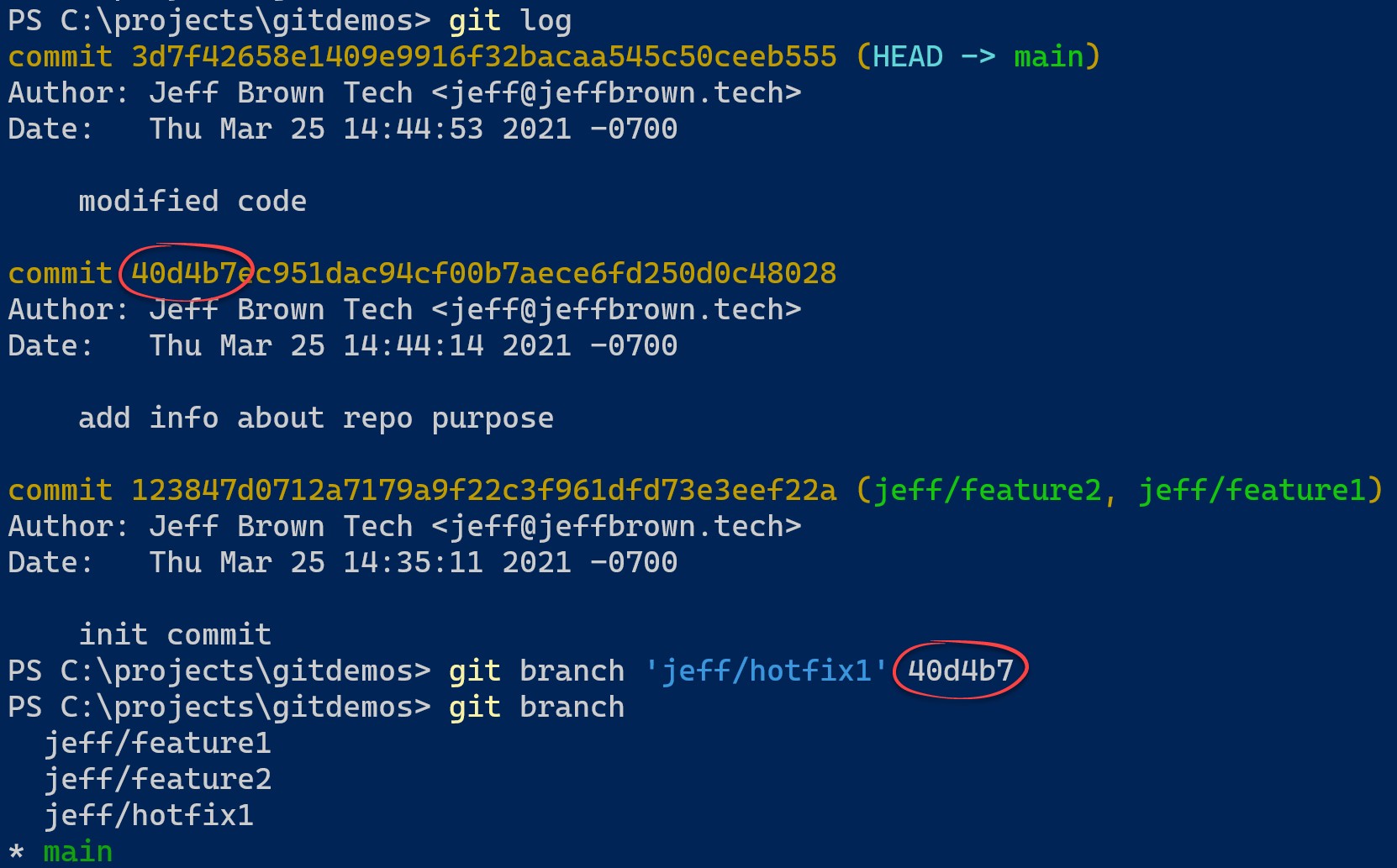
Before creating the branch, you need the SHA-1 identifier of the commit. To find the identifier, use the git logcommand to view previous commits. Each commit will have a complete SHA-1 hash as the identifier. However, you only need the first few characters to identify the commit.

Next, use the same git branchcommand from Option 1 but append the commit identifier at the end. This example is using **40b4d7** from the second commit as the identifier.

Note the HEAD designator is on the main branch, which is the active branch. The other branches *jeff/feature1* and *jeff/feature2* point to the same commit when you created them earlier. Both point to the same snapshot as each branch has not had additional commits made to each one since creation.

git log

git branch <branch name> <identifier>



## Option 4: Creating a Branch from Another Branch

If you use branches dedicated to hotfixes or features, you create branches from these other branches to work on the item. Creating a branch from another branch is no different from creating from the main branch. You just need to specify the name of the other branch as the starting point. This example shows creating the *feature4* branch from the develop branch.

git checkout -b feature4 develop

# Merge branches in GitHub

Now, let’s discuss how to merge GitHub branches. It’s relatively simple to do:

1. Navigate to your repository, then find and click the **Pull requests** button. It should be in between the Issues and Actions buttons. You’ll see a summary of all pull requests you have pending.
2. Navigate to and click on the pull request you’d like to merge into the main branch.
3. Now, you have a few choices for initiating the merge, depending on your repository’s merge options:

### Merge every commit into your main branch:

To do this, click **Merge pull request**. If this button does not appear, open the dropdown menu and click on **Create a merge commit**.

### Combine the commits into one large commit:

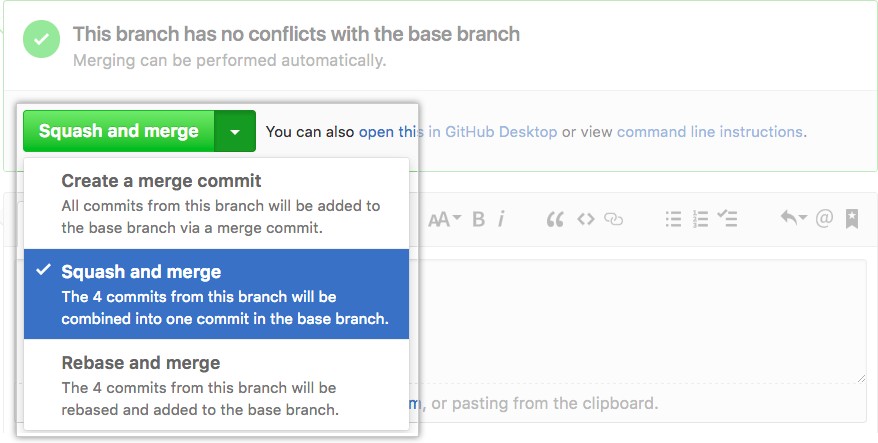
You can “squash” your commits together by clicking the dropdown menu, then

choosing **Squash and merge** and clicking the new **Squash and merge** button. Doing this helps you create a cleaner, more streamlined Git history for your repository.

### Rebase each commit onto the main branch:

To do this, click the dropdown menu, choose **Rebase and merge,** and click the new **Rebase and merge** button. This also creates a cleaner project history.

1. You can now leave a comment if you’d like, or you can accept the default message GitHub provides you.
2. Scroll below the commit message field to choose a Git author email address.
3. Click on **Confirm merge**, **Confirm squash and merge**, or **Confirm rebase and merge**, depending on which option you choose in Step 3.
4. If the merge occurs successfully, GitHub will display a note stating so. This helps confirm that you’re good to go.
5. If you’d like, you can now delete the branch by clicking **Delete branch** to keep things nice and neat. Alternatively, you can set it up so that the branch deletes automatically once you complete a merge to save some time.



The merge button

Merging Branches in a Local Repository

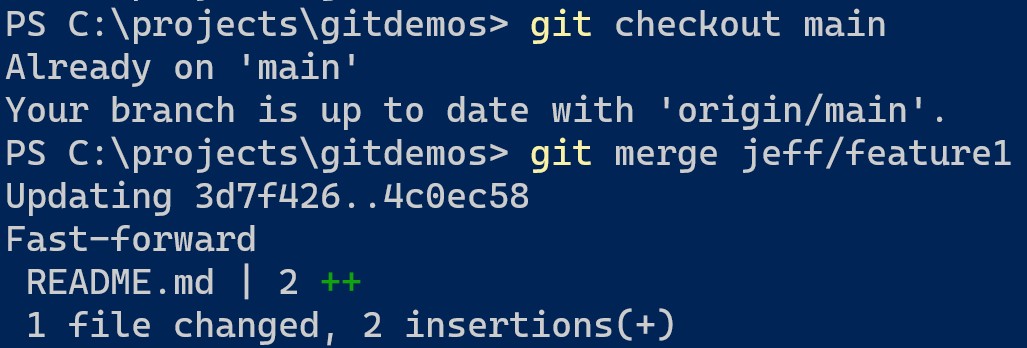
To merge branches locally, use git checkoutto switch to the branch you want to merge into. This branch is typically the *main* branch. Next, use git mergeand specify the name of the other

branch to bring into this branch. This example merges the *jeff/feature1* branch into the *main*

branch. Note that this is a **fast-forward** merge.

git checkout main

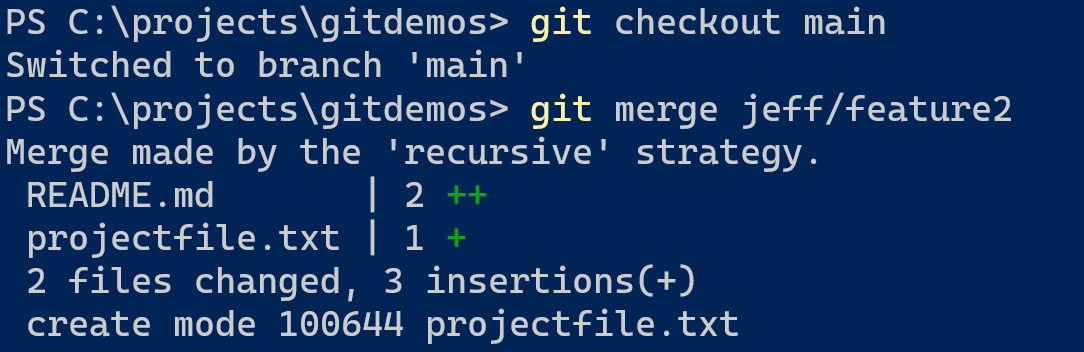
git merge jeff/feature1



Work continues on the main and other branches, so they no longer share a common commit history. Now a developer wants to merge the *jeff/feature2* branch into the main branch. Instead, Git performs a three-way (or recursive) merge commit.

git checkout main

git merge jeff/feature2



## Merging Main into a Branch

While you are working on your branch, other developers may update the *main* branch with their branch. This action means your branch is now out of date of the *main* branch and missing content. You can merge the *main* branch into your branch by checking out your branch and using the same git merge command.

git checkout <branch name> git merge main

