  ***Git***

**login to github from terminal**

**how to sign into github from terminal**

$ git config --global user.name "John Doe"

$ git config --global user.email johndoe@example.com

or to see the signed in user

git config –list

**how to login to git from terminal**

git config --global user.name "your\_username"

**how to login to git from terminal**

git config --global user.email [your\_email\_address@example.com](mailto:your_email_address@example.com)

**how to set up git for github**

$ git config --global user.name "Your name here"

$ git config --global user.email [your\_email@example.com](mailto:your_email@example.com)

**setup github password terminal**

$ git config credential.helper store

$ git push https://github.com/owner/repo.git

Username for 'https://github.com': <USERNAME>

Password for 'https://USERNAME@github.com': <PASSWORD>

# All Git Commands

**search (regex)**

git grep "regex"

**list all branches**

git branch -a

**list remote branches**

git branch -r

**checkout a branch on remote**

make sure you don't use origin

git fetch

git checkout branchName

**Create a new branch**

first create a branch

git checkout -b <branchName>

Create a new branch from an existing branch

git checkout origin/branchName -b newBranchName

Then push your new branch to the repo

git push origin <branchName>

**Create a branch from a commit**

AKA Recover a deleted branch

git checkout -b <branch> <sha>

**revert all changes in a branch. Removes staged and working directory changes.**

git reset --hard

**Resets index to former commit; replace 56e05fced with your commit code. You can use git log to get commit code**

git reset 56e05fced

**revert a file to the most recent commit**

git checkout HEAD -- /somePath/file.txt

**to discard changes in working directory**

git checkout -- <file>

**Checkout a file from another branch**

git checkout origin/branchName -- fileName.txt

**undo the last commit. Blow it out of the water.**

git reset --hard HEAD~1

**undo your last commit but leave the files from that commit staged.**

git reset --soft HEAD~1

**delete local (untracked) files**

git clean -f

**If you want to also remove directories, run**

git clean -f -d

**clean a folder**

git clean -fxd {dir\_path}

**commit a folder/file without staging it.**

git commit /folderToCommit -m 'commit msg'

**list all branches (remote & local/remote only)**

git branch -a

git branch -r

**Find out all branches a commit is on**

git branch --contains <commit>

**display log with Tree**

git log --pretty=format:"%h - %cr (%an) %s" --graph

**Merge Master into your local branch**

git fetch

git merge origin/master

a shortcut to this is. They are both the same

git pull origin master

or, if it's a busy repo.

git pull --rebase <remote name> <branch name>

**list conflicts**

git diff --name-only --diff-filter=U

grep -lr '<<<<<<<' .

**Diff a conflict**

git mergetool -t opendiff

**pull a branch , merge if conflicted use remote.**

git pull -s recursive -X theirs origin ra

**show log with merged files**

git log -m -1 --name-only

**Show the changes between two branches.**

git diff --name-status master..branchName > changelog.txt

**Recover a deleted branch**

Get the SHA of the last commit on the branch.

git checkout -b newbranchname 56e05fced

**Stashes**

**save a stash**

git stash save "My changes."

**list your saved stashes**

git stash list

**apply a stash (Where stash@{1} is the stash you want to apply.)**

git stash apply stash@{1}

**delete a branch on origin**

git push origin --delete <branchName>

**delete a branch locally**

git branch -d <branchName>

**Get all commits from a branch. For a release log, changelog etc.**

git cherry -v develop mybranch

**Revert a commit that is origin/remote**

This reverts the commit with a new commit.

First get the commit sha.

git revert -m 1 <commit-hash>

git commit -m "Reverting the last commit which messed the repo."

git push -u origin master

**Utilities**

**Get the status on all repos in a folder**

find . -maxdepth 1 -mindepth 1 -type d -exec sh -c '(echo {} && cd {} && git status -s && echo)' \;

Save the results to a file.

find . -maxdepth 1 -mindepth 1 -type d -exec sh -c '(echo {} && cd {} && git status -s && echo)' \; > gitreport.txt

**Delete all local branches that don't exist on origin**

run git fetch -p this removes the remote references.

run git branch -vv

then run the following script

git fetch -p && for branch in `git branch -vv | grep ': gone]' | awk '{print $1}'`; do git branch -D $branch; done

**Adding project local machine to Git repository:**

## **Adding a project to GitHub with GitHub CLI**

1. In the command line, navigate to the root directory of your project.
2. Initialize the local directory as a Git repository.

git init -b main

1. To create a repository for your project on GitHub, use the gh repo create subcommand. Replace project-name with the desired name for your repository. If you want your project to belong to an organization instead of to your user account, specify the organization name and project name with organization-name/project-name.

gh repo create *project-name*

1. Pull changes from the new repository that you created. (If you created a .gitignore or LICENSE file in the previous step, this will pull those changes to your local directory.)

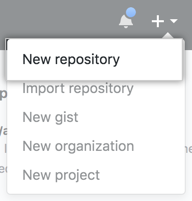
git pull --set-upstream origin main

1. Stage, commit, and push all of the files in your project.

git add . && git commit -m "initial commit" && git push

## **Adding a project to GitHub without GitHub CLI**

1. [Create a new repository](https://docs.github.com/en/articles/creating-a-new-repository) on GitHub.com. To avoid errors, do not initialize the new repository with *README*, license, or gitignore files. You can add these files after your project has been pushed to GitHub

.

1. Open Git Bash.
2. Change the current working directory to your local project.
3. Initialize the local directory as a Git repository.

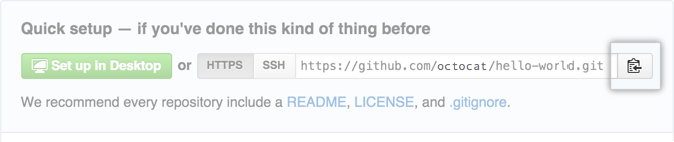
$ git init -b main

Add the files in your new local repository. This stages them for the first commit.

$ git add .

1. Commit the files that you've staged in your local repository.

$ git commit -m "First commit"

1. At the top of your repository on GitHub.com's Quick Setup page, click  to copy the remote repository URL.
2. In the Command prompt, [add the URL for the remote repository](https://docs.github.com/en/github/getting-started-with-github/managing-remote-repositories) where your local repository will be pushed.
3. $ git remote add origin  *<REMOTE\_URL>*
4. # Sets the new remote
5. $ git remote -v

# Verifies the new remote URL

1. [Push the changes](https://docs.github.com/en/github/getting-started-with-github/pushing-commits-to-a-remote-repository) in your local repository to GitHub.com.

$ git push origin main

**Importing a Git repository to local machine**

Before you start, make sure you know:

* Your GitHub username
* The clone URL for the external repository, such as https://external-host.com/user/repo.git or git://external-host.com/user/repo.git (perhaps with a user@ in front of the external-host.com domain name)

1. [Create a new repository on GitHub](https://docs.github.com/en/articles/creating-a-new-repository). You'll import your external Git repository to this new repository.
2. On the command line, make a "bare" clone of the repository using the external clone URL. This creates a full copy of the data, but without a working directory for editing files, and ensures a clean, fresh export of all the old data.

$ git clone --bare [https://external-host.com/*extuser*/*repo.git*](https://external-host.com/extuser/repo.git)

1. Push the locally cloned repository to GitHub using the "mirror" option, which ensures that all references, such as branches and tags, are copied to the imported repository.

$ cd *repo.git*

$ git push --mirror https://github.com/*ghuser*/*repo.git*

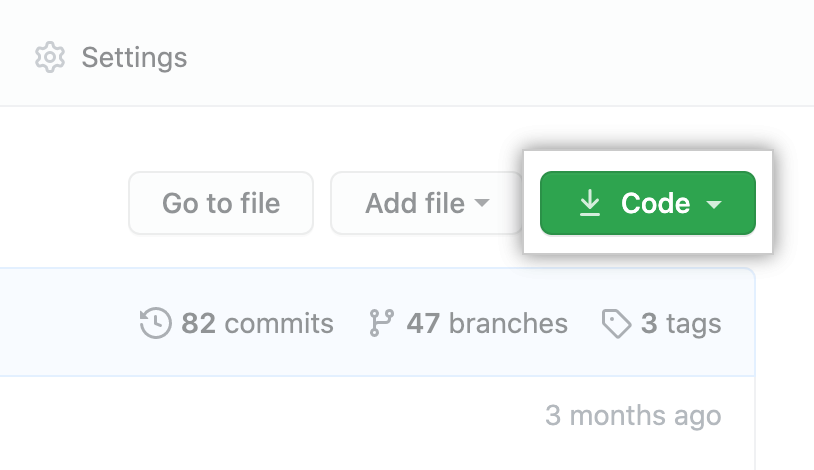
# Pushes the mirror to the new repository on GitHub.com

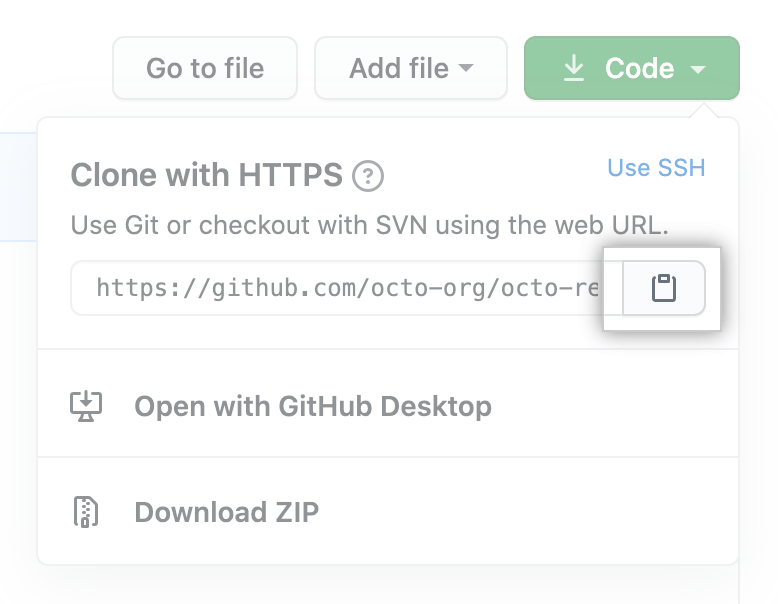
1. Remove the temporary local repository.

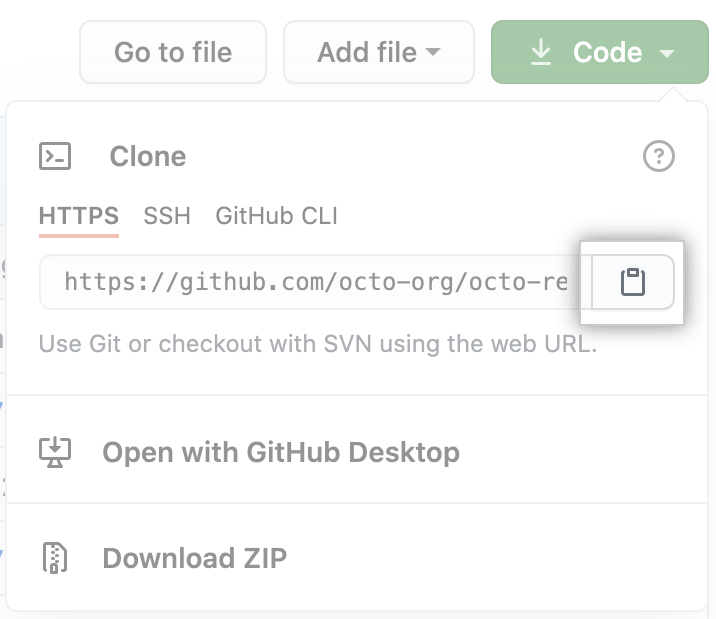
$ cd ..

$ rm -rf *repo.git*

# Cloning a repository

1. On GitHub.com, navigate to the main page of the repository.
2. Above the list of files, click  **Code**.
3. To clone the repository using HTTPS, under "Clone with HTTPS", click . To clone the repository using an SSH key, including a certificate issued by your organization's SSH certificate authority, click **Use SSH**, then click . To clone a repository using GitHub CLI, click **Use GitHub CLI**, then click .





1. Open Git Bash.
2. Change the current working directory to the location where you want the cloned directory.
3. Type git clone, and then paste the URL you copied earlier.

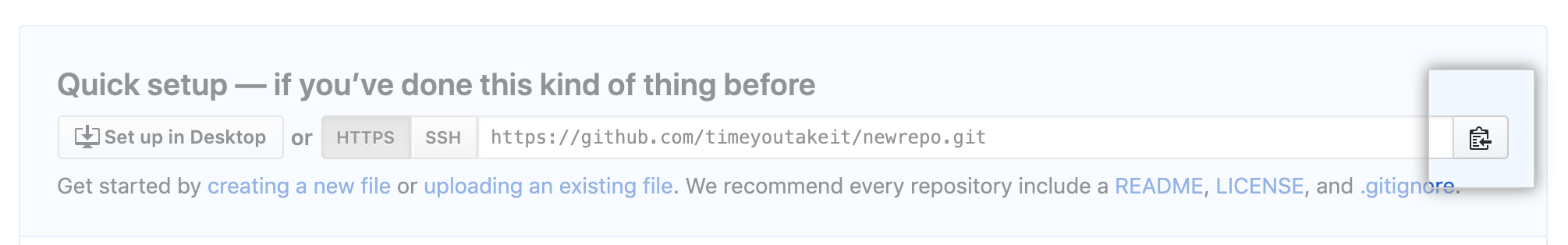
$ git clone <https://github.com/YOUR-USERNAME/YOUR-REPOSITORY>

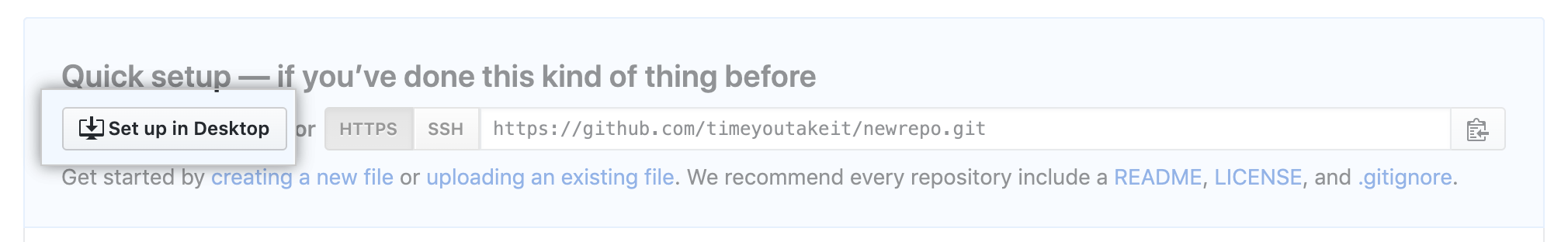
1. Press **Enter** to create your local clone.

$ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY

## Cloning an empty repository

An empty repository contains no files. It's often made if you don't initialize the repository with a README when creating it.

1. On GitHub.com, navigate to the main page of the repository.
2. To clone your repository using the command line using HTTPS, under "Quick setup", click . To clone the repository using an SSH key, including a certificate issued by your organization's SSH certificate authority, click **SSH**, then click .

Alternatively, to clone your repository in Desktop, click  **Set up in Desktop** and follow the prompts to complete the clone.

1. Open Git Bash.
2. Change the current working directory to the location where you want the cloned directory.
3. Type git clone, and then paste the URL you copied earlier.

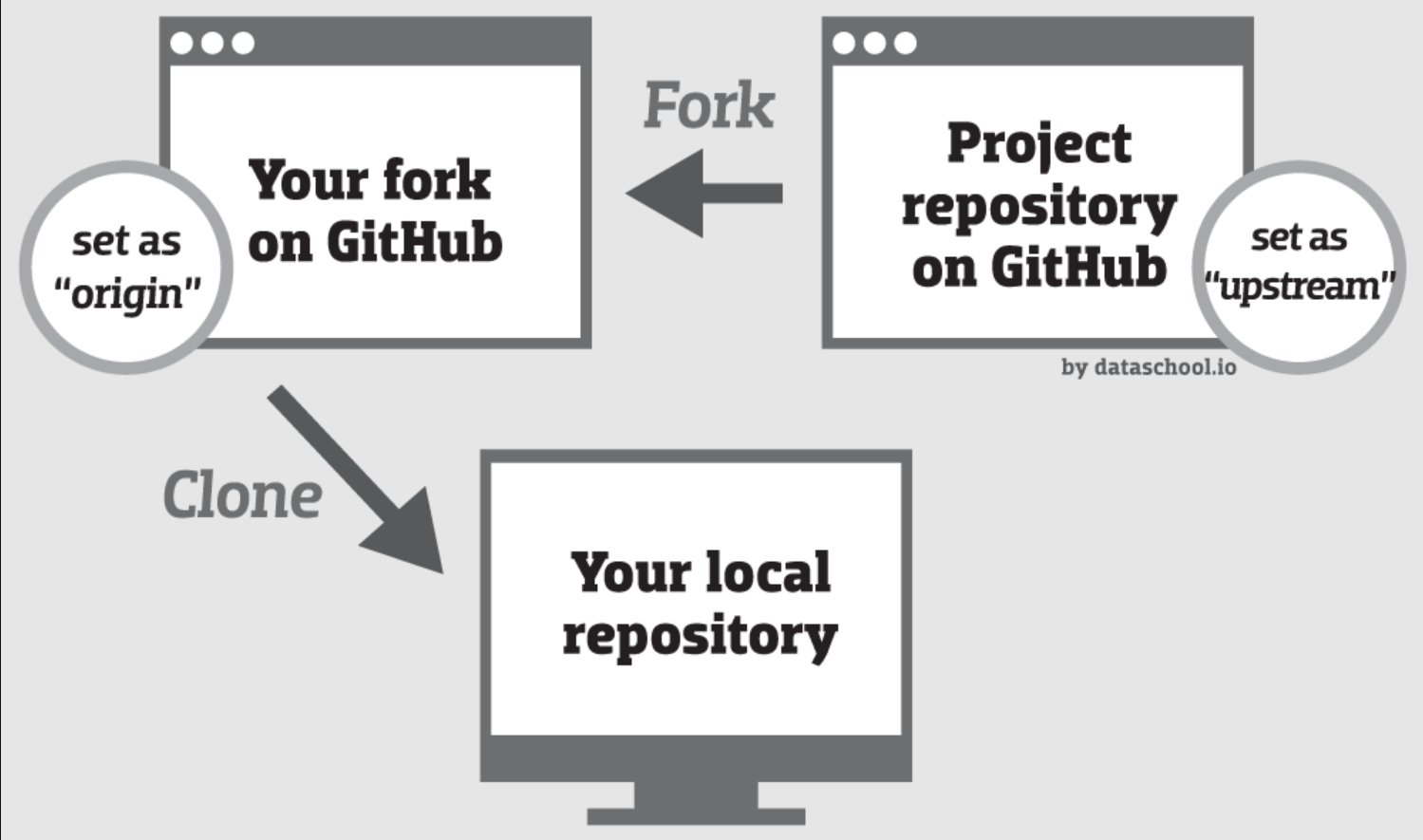
$ git clone <https://github.com/YOUR-USERNAME/YOUR-REPOSITORY>

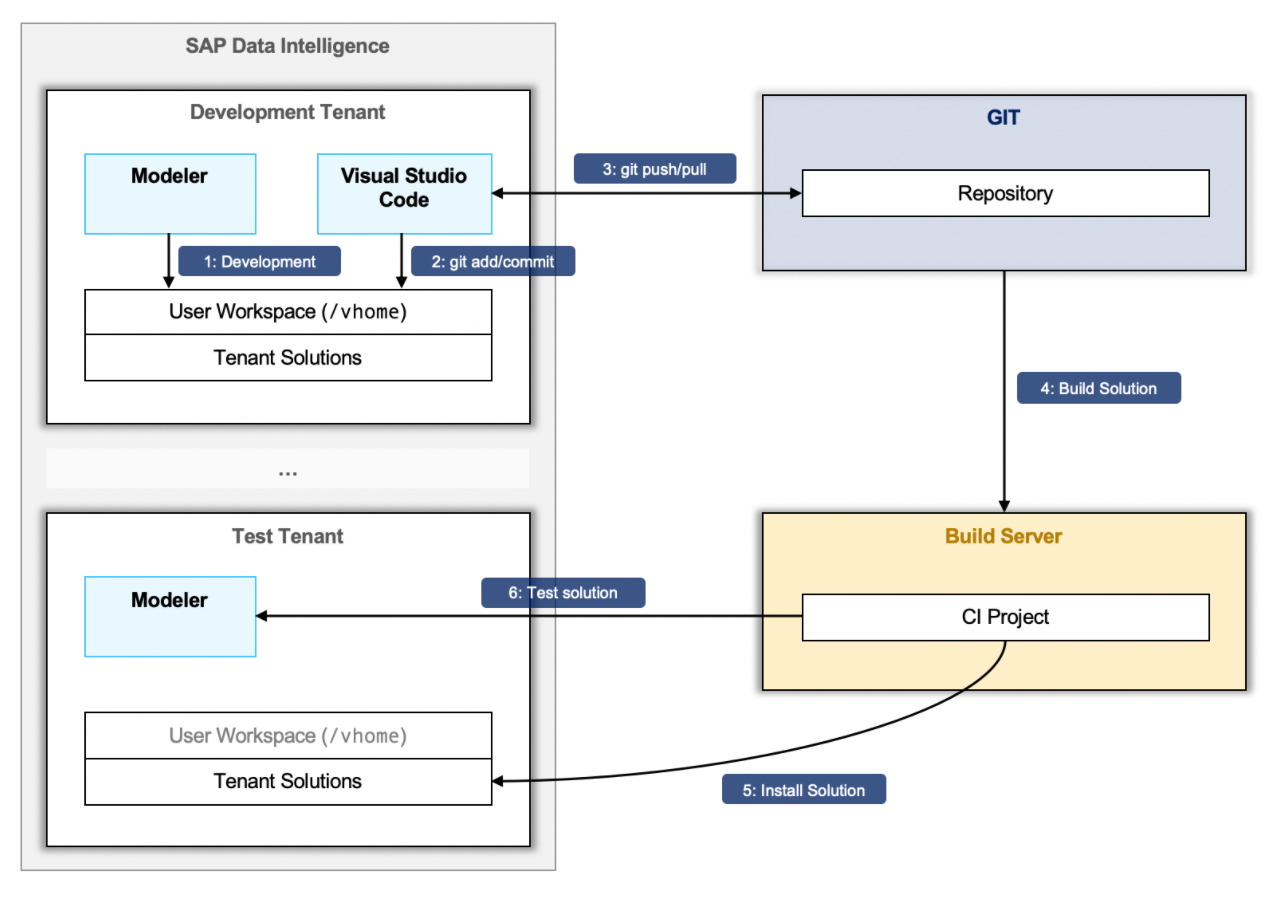
1. Press **Enter** to create your local clone.

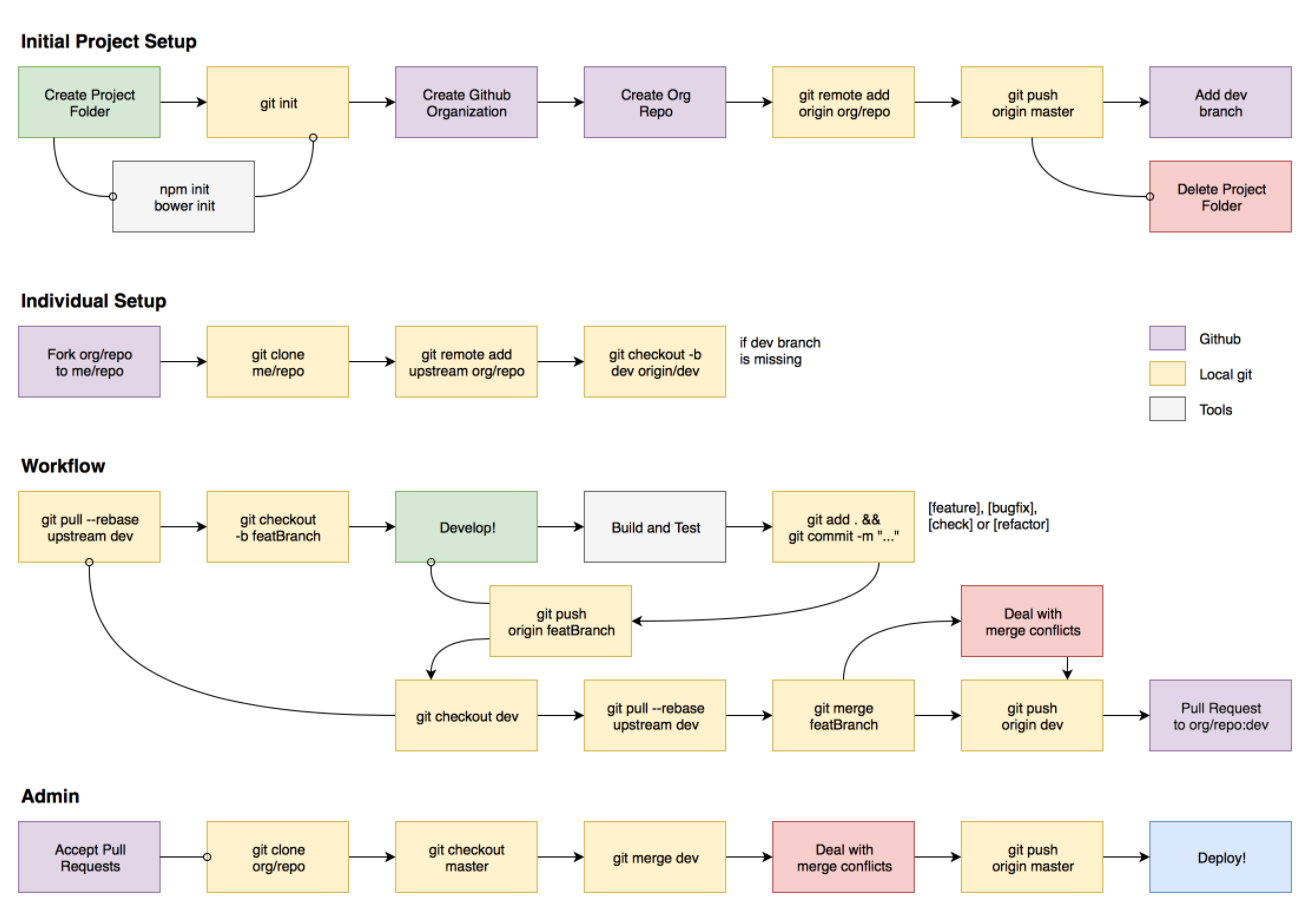
$ git clone <https://github.com/YOUR-USERNAME/YOUR-REPOSITORY>

**Chart:**

***Git Flow:***

******

******



# Links:

***Github link:***

<https://github.com/>

***Clone and use Git repository in Visual Studio Code:***

<https://docs.microsoft.com/en-us/azure/developer/javascript/how-to/with-visual-studio-code/clone-github-repository>

***Clone and use Git repository in Visual Studio :***

<https://docs.microsoft.com/en-us/visualstudio/get-started/tutorial-open-project-from-repo?view=vs>

***Top 20 Command should know:***

<https://www.freecodecamp.org/news/10-important-git-commands-that-every-developer-should-know/>