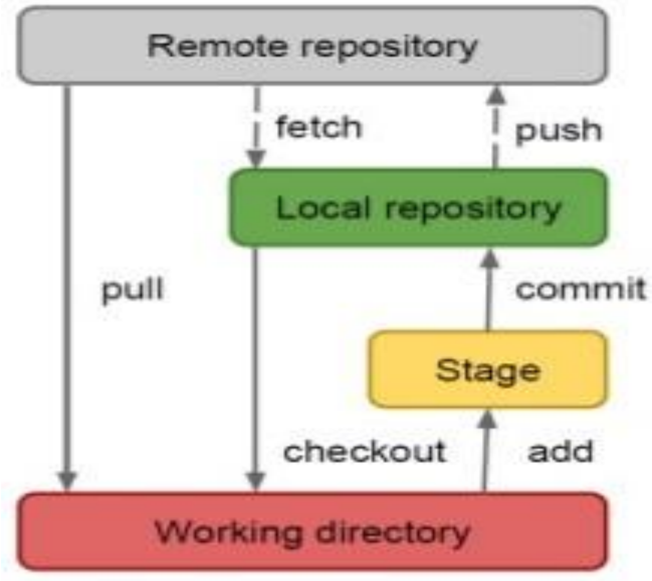
GIT

(VCS - Version Control System)

GIT is a distributed version control & source code management system with an emphasis on speed
Its commonly used for source code management (SCM)



Git Repository Technology

Repository : GIT Repo is a directory that stores all the files, folders and content needed for your project

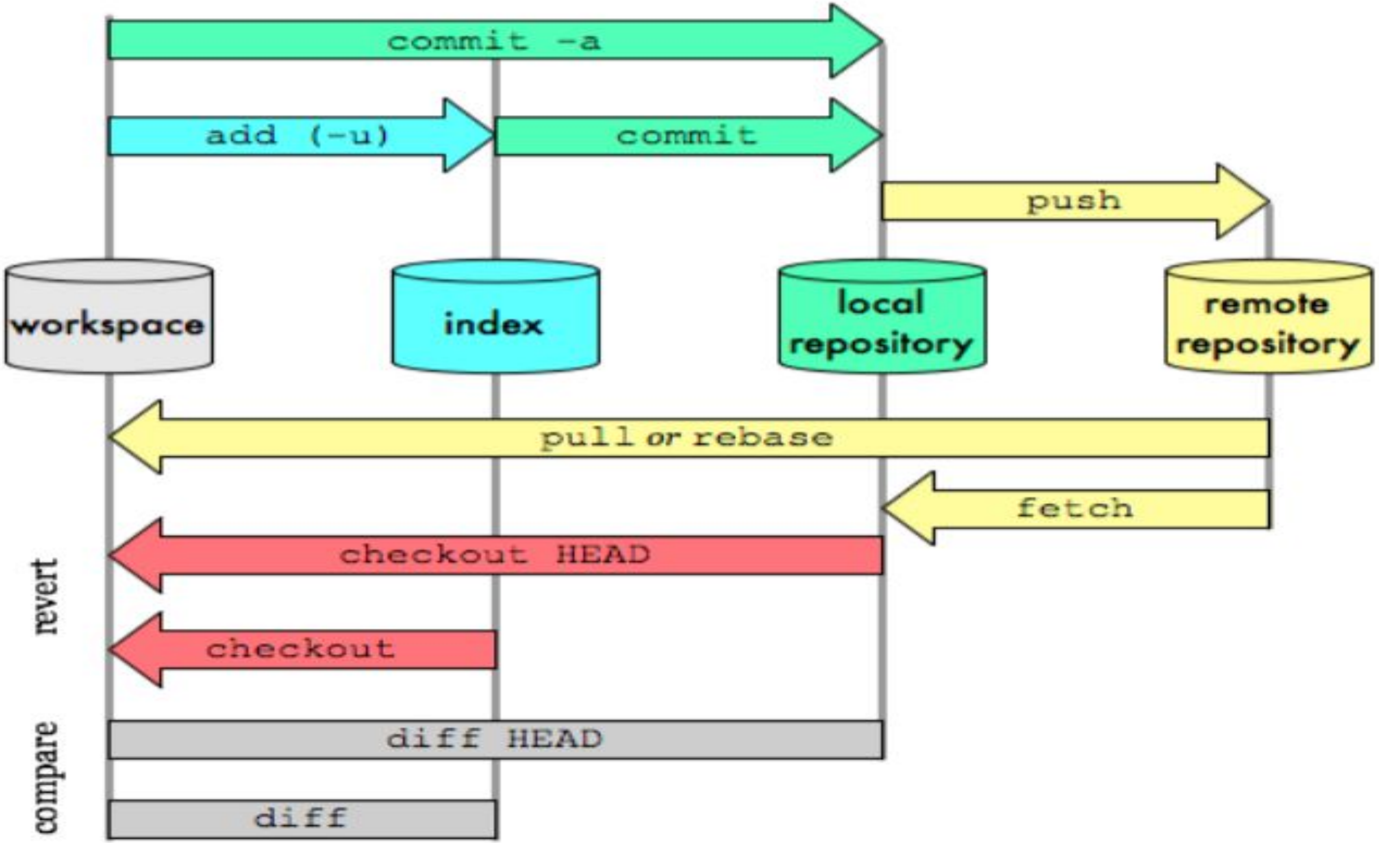
Branch : A version of the repo that diverges from the main working project. Branches can be a new version of a repository

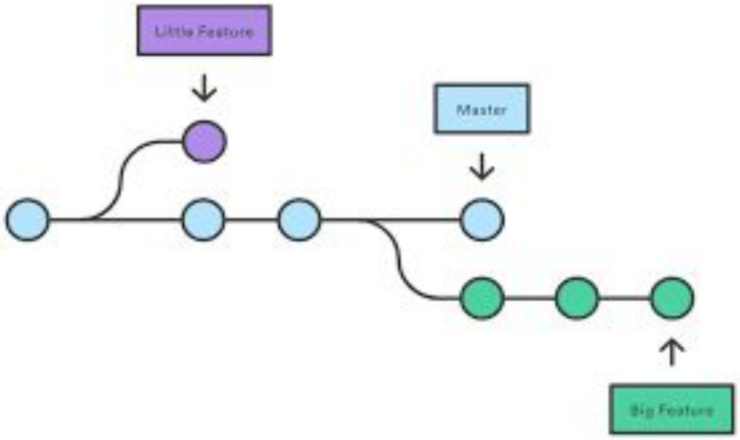
Clone : a clone is a copy of repo or action of copying a repo

Master : the primary branch of all repo’s. All committed and accepted changes should be on the master branch.

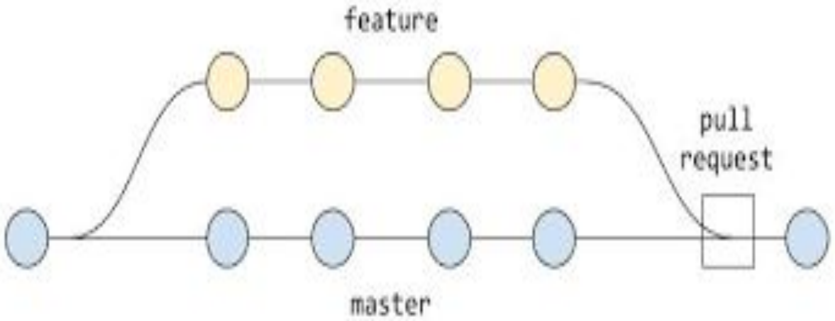
Merge : merging two branches

GIT Architecture



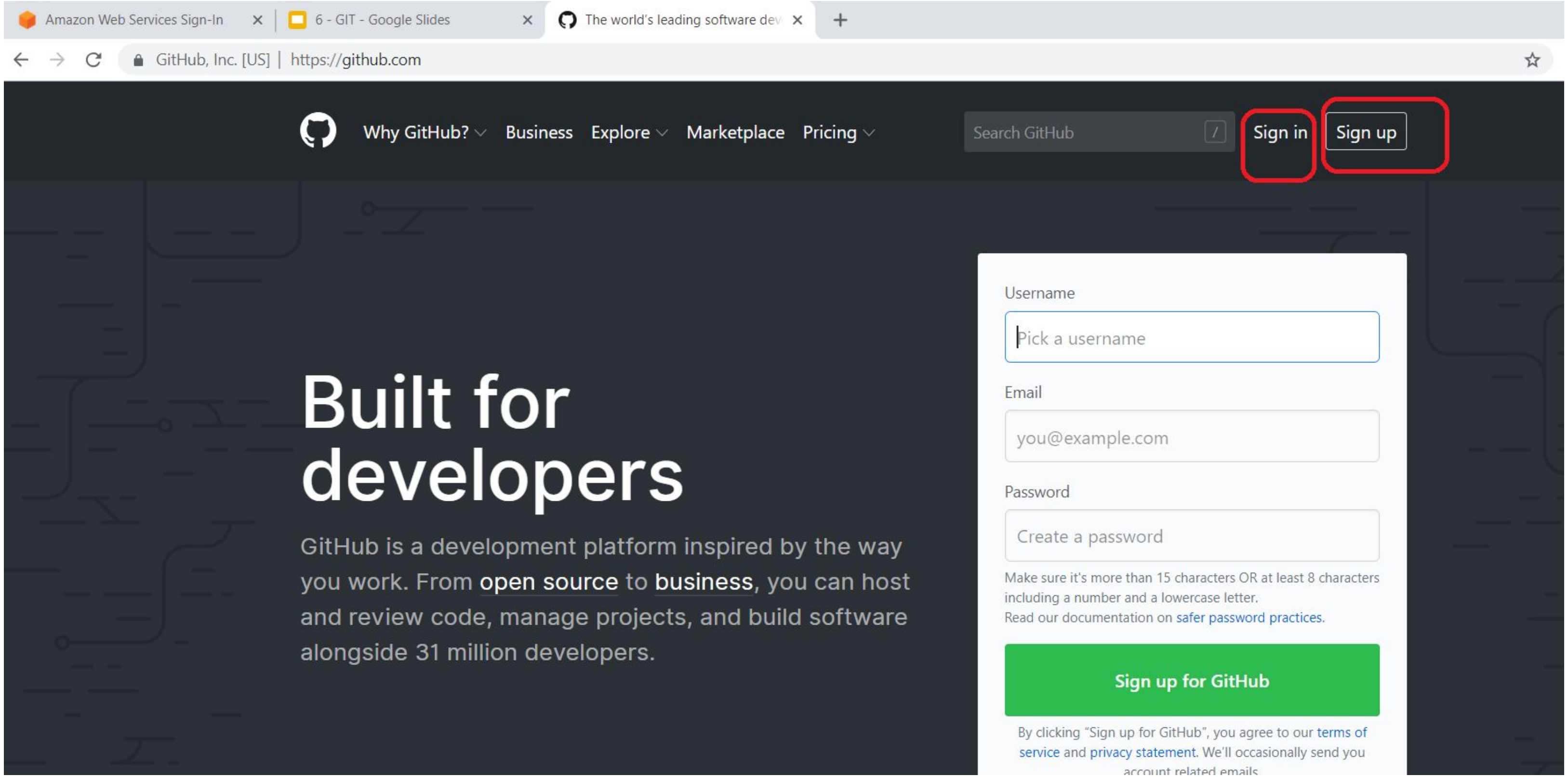


Little Feature
Master
Big Feature



feature
pull request
master

Creating an Account and Repository in GIT HUB


Amazon Web Services Sign-In
6 - GIT - Google Slides
The world's leading software dev
GitHub, Inc. [US] | https://github.com
Why GitHub?
Business
Explore
Marketplace
Pricing
Search GitHub
Sign in
Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 31 million developers.

Username

Pick a username

Email

you@example.com

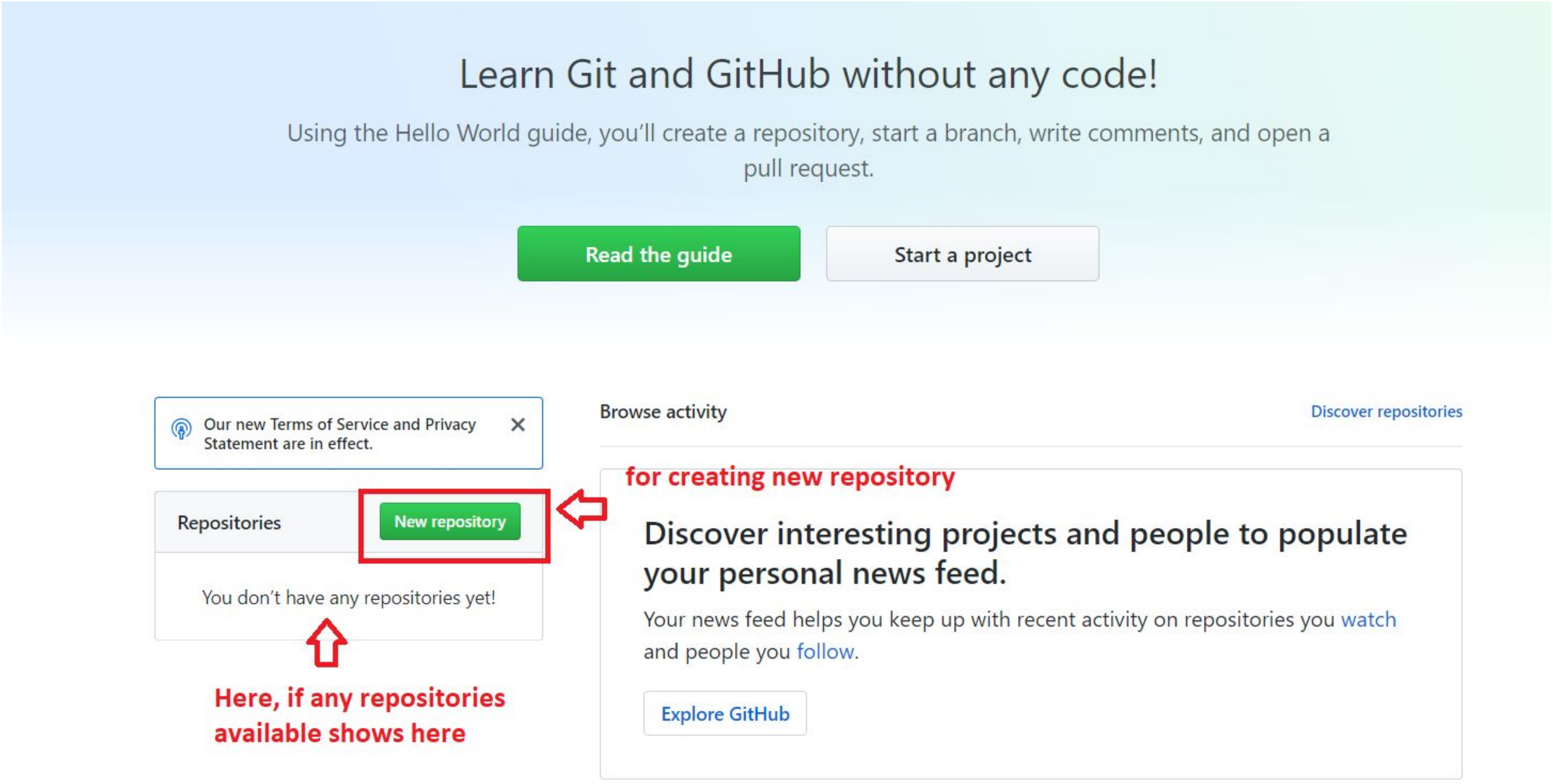
Password

Create a password

Make sure it's more than 15 characters OR at least 8 characters including a number and a lowercase letter.
Read our documentation on safer password practices.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our terms of service and privacy statement. We'll occasionally send you account related emails.



Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Our new Terms of Service and Privacy Statement are in effect.

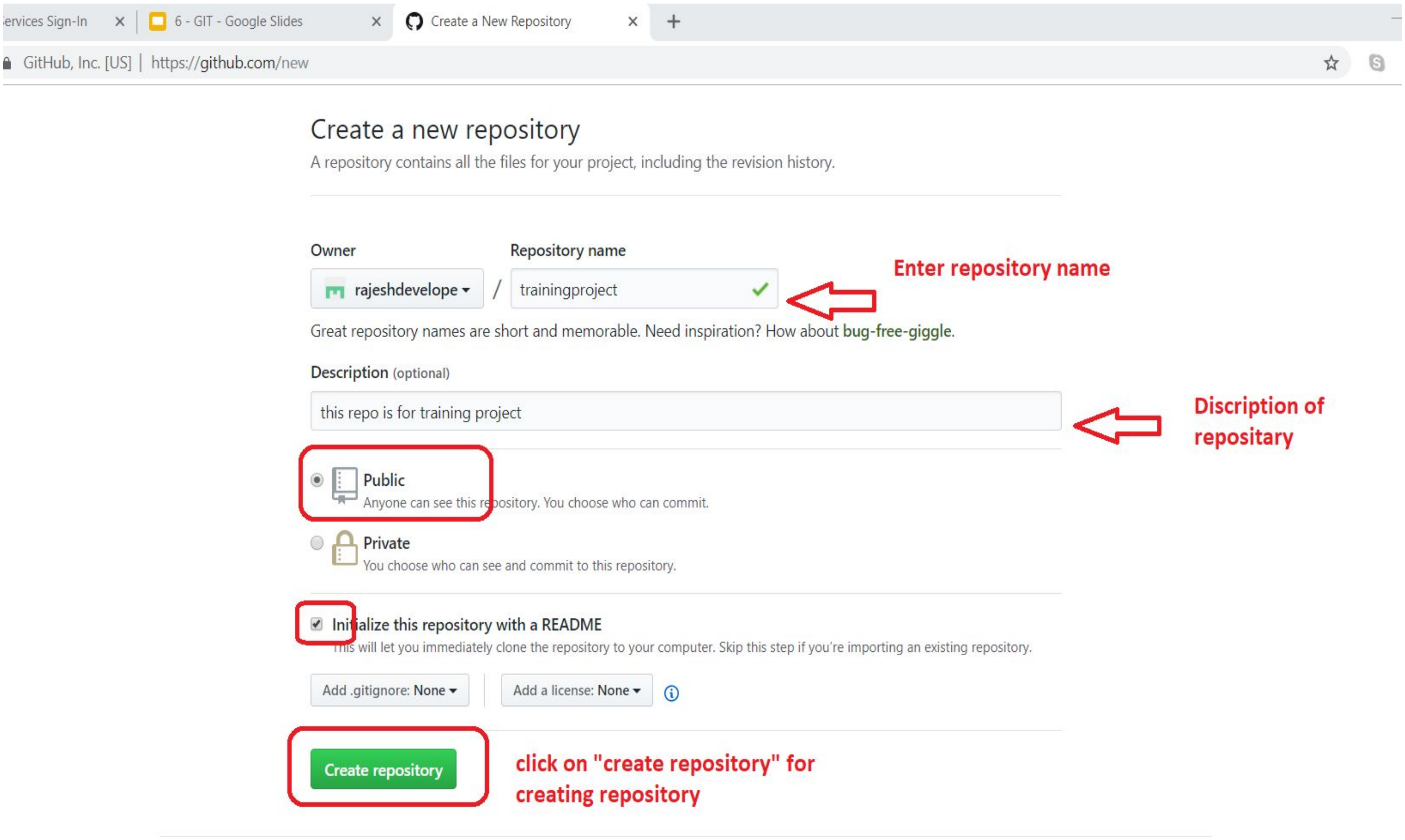
You don't have any repositories yet!

for creating new repository

Discover interesting projects and people to populate your personal news feed.

Your news feed helps you keep up with recent activity on repositories you watch and people you follow.

Here, if any repositories available shows here



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: rajeshdevelope / Repository name: trainingproject ✓

Enter repository name

Great repository names are short and memorable. Need inspiration? How about bug-free-giggle.

Description (optional): this repo is for training project

Discription of repositary

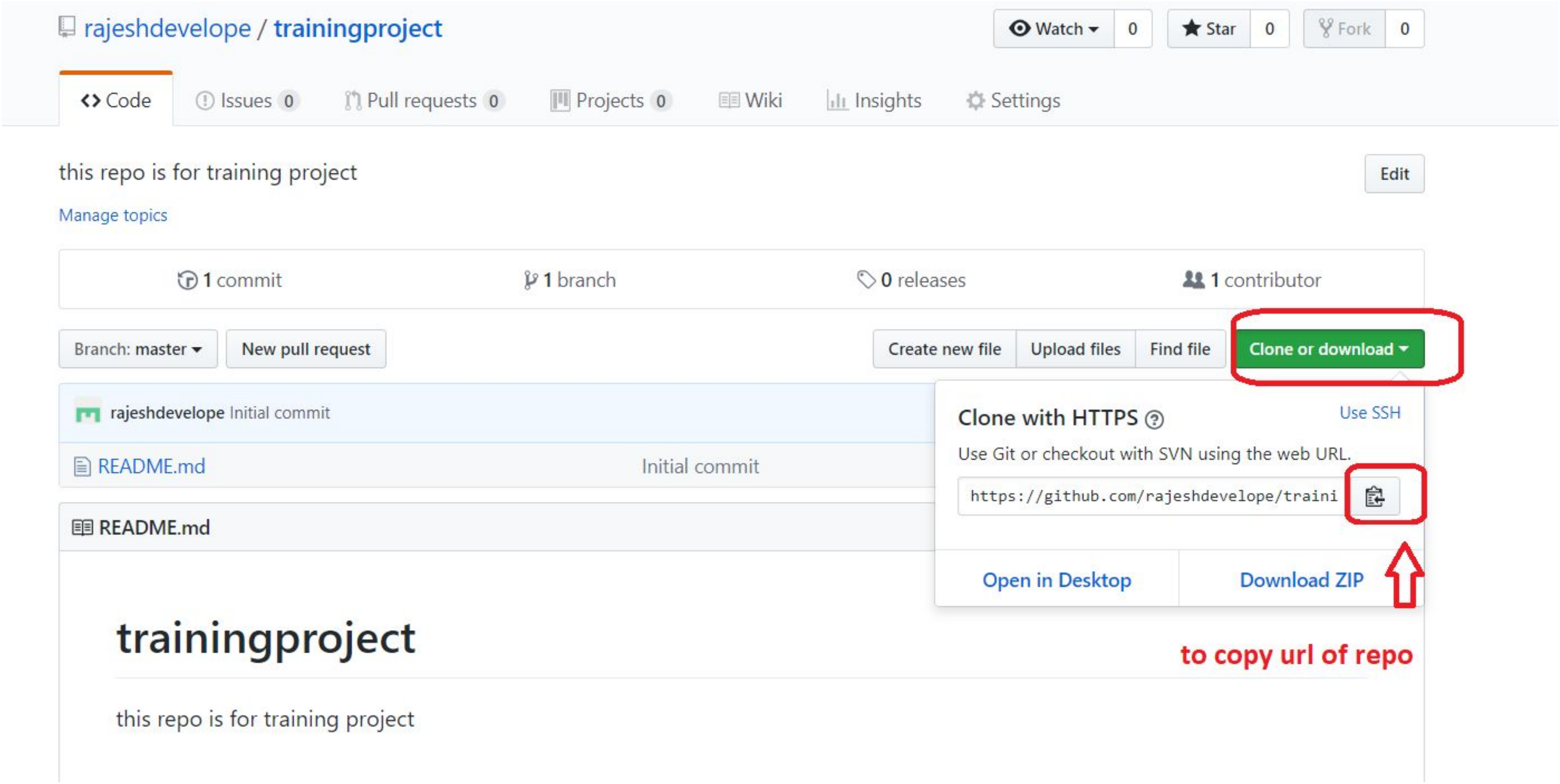
● Public
Anyone can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

☑ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None ▾ | Add a license: None ▾

click on "create repository" for creating repository



this repo is for training project

Use Git or checkout with SVN using the web URL.

```
https://github.com/rajeshdevelope/traini
```

trainingproject

this repo is for training project

to copy url of repo

Install GIT on Ubuntu

Step 1 : Installation

```
 #apt-get update
 #apt-get install git-core   -y    ----- for installing git client
 #git --version       ---- for checking git version
```

Step 2 : Configuration

```
 #git config --global user.name rajeshdevops      ------ configuring user name
 #git config --global user.email rajeshdeveloper99@gmail.com   --- configuring mail id (master mail id)
#cat .gitconfig (or) #git config --list
```

Step 3 : Create GIT Repository

```
  #mkdir /repos    -- creating repos directors
 #cd /repos        -- changing to repos dir
 #git init    ---- initializing GIT
 #ls -a
#git clone <repo url>
```

Step 4 : Working with GIT Repository

```
 #cd <repo name>

 #echo “Welcome to GIT” >> README.md --- creating README.md file and writing “Welcome to GIT” in it

 #git status ----  it show git status as what are changes done on branch

 #git add <file name>  -- to add file to cache/stage/index area

    #git status

 #git rm --cached <filename>    ---- to remove file from cache/stage/index area

#git commit -m “<comments>”    ------- to move all changes from stage area to local repo

#git commit -a -m “<comments>”   ----- to add to cache and commit to local repo at a time

#git push -u origin master ----- to push changes to central repo

#git status -s   ----- to list all changes in short format

#git log   ------ to show all git comments

#git log -p ---- to show all git commits with code changes
```

#git log --oneline ---> to show logs in one line with commit id

#git log --since=01-01-2019 --until=08-01-2019 ---> to show logs between dates

To work with Branches : always branch will be created from latest commit ID only

#git branch ---> to list all branches

#git branch <branch name> ---> to create branch

#git checkout <branch name> ---> to change / switch branch from current branch

#vi index.html ---> adding file in <branch name> branch

#git commit -a -m “index.html added”

#git push origin <branch name>

Check in browser git hub

#git rm <file name> ---> to remove file name which are associated with git

#git mv <file name> /mydir/<file name> ---> for moving file

#git log <file name> ---> to show all logs related to file

#git log --help ---> to get help

#git log --oneline

#git diff 33eeee ..345dddd ---> to show difference between two commits

Working with tag : tag is a alias name for commit ID

#git tag <tag name> <commit id> ---> creating tag name for commit id

#git tag --list ---> to show all tags names

#git diff <tag name> <tag name> ----> to show difference between tag

```
#git checkout master ----> to change from current branch to master branch

#git merge <branch name> ---> to merge <branch name> to master branch

#git push -u origin master ----> to push merge to central repo

#git branch -d <branch name>  ---> to delete a branch from local repo only

#git branch -D <branch name> ---> to delete a branch from local without merging

#git push origin --delete <branch name> ---> to delete branch from central (check in web)

#git diff ---> to list differences

#history ----> gives all the list of command that we executed after login

#git log

#git show c2233……..dfdfdf ----> to show about that particular commit
```

Git Merge & Rebase

Merge : Merge takes all the changes in one branch and merges them into another branch in one commit

Base : denotes origin of branch

Git Rebase : as its name suggests, rebase exists to change the “base” of a branch, which means its origin commit.

#git rebase <branch name> ---> to change the base of branch

(you should be in same branch, which you want to merge