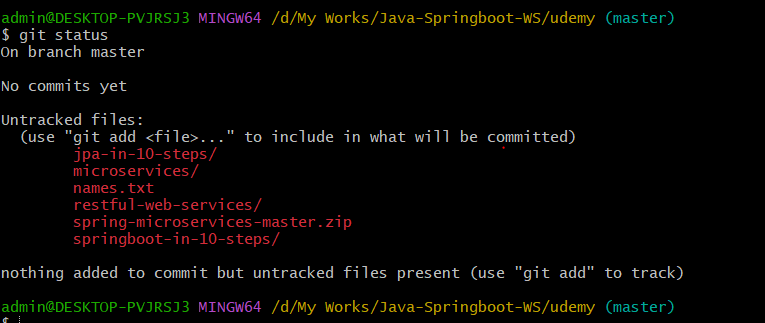
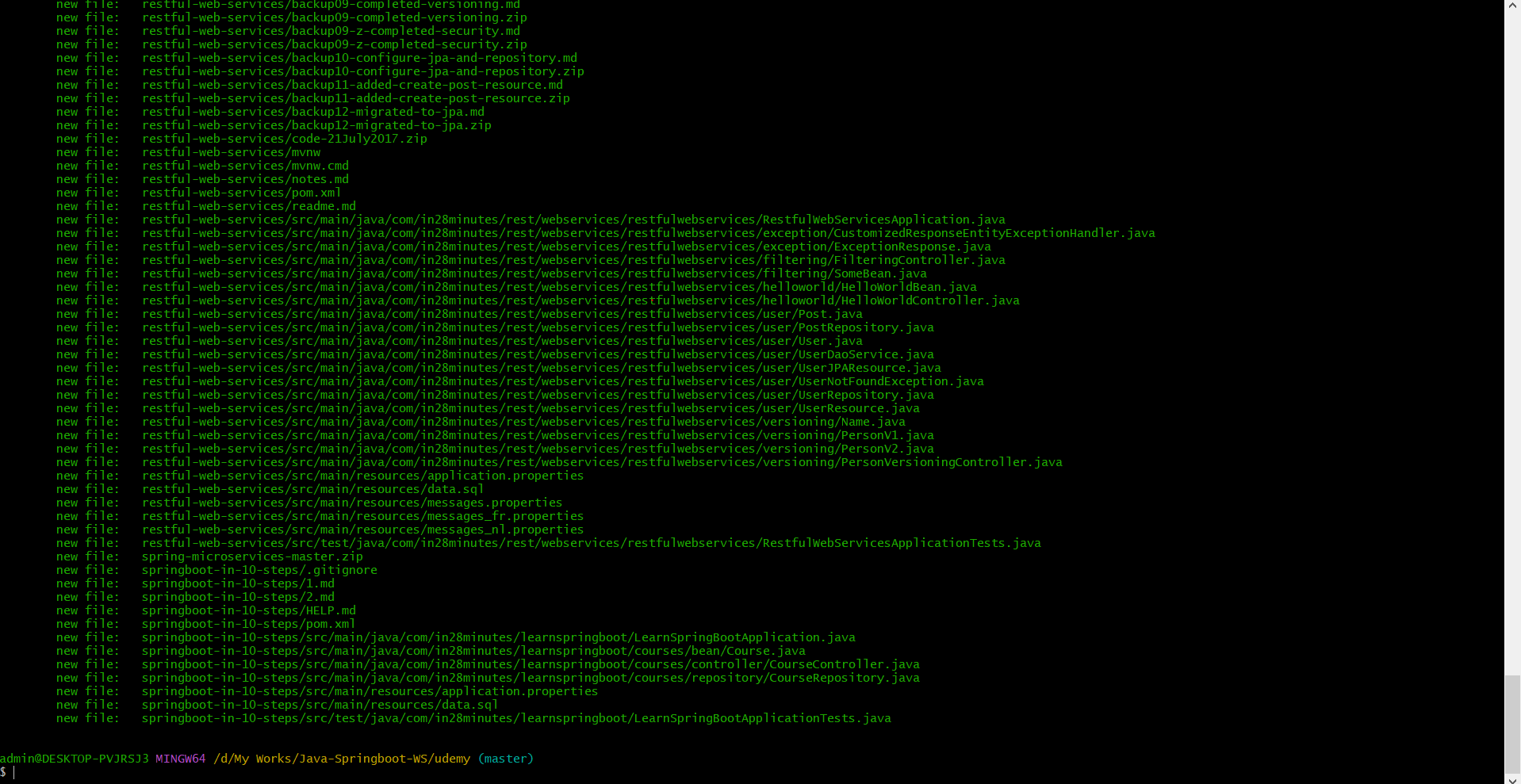
* 🡪 🡪 **Git 🡪 🡪 🡪**

<https://git-scm.com/download/win> --> download

1. Cd “full path” like cd “D:\My Works\Java-Springboot-WS\udemy”
2. $ git init to Initialized empty Git repository in D:/My Works/Java-Springboot-WS/udemy/.git/
3. **ls -a to show hidden file** & ls to show all files in directory except hidden
4. touch names.txt – to create any empty file
5. git status 

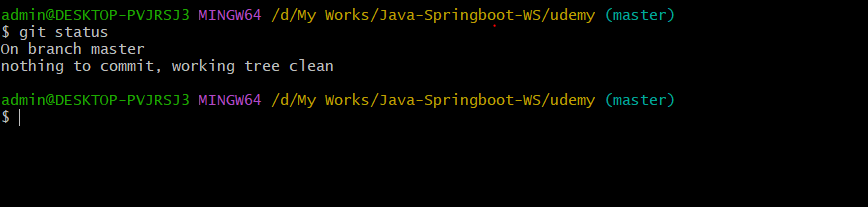
**untracked files:**- Git sees a file you didn't have in the previous snapshot (commit), and which hasn't yet been staged;

1. **git add .** (**git space dot**) 🡪 . means everything in this current directory will be staged(It tells Git that you want to include updates to a particular file in the next commit.) -- for individual files git add names.txt
2. **git status**

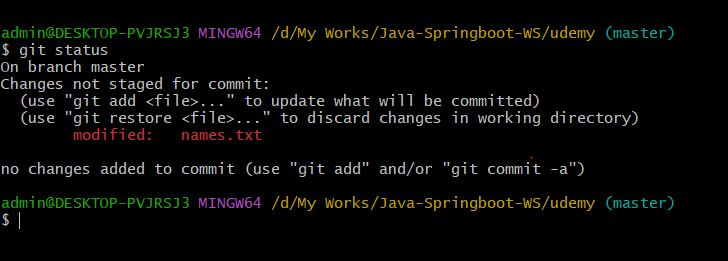


Green – added to history

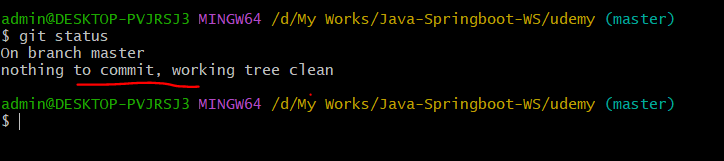
1. $ git commit -m "many files are added --->" 🡪 m means message (The git commit command **captures a snapshot of the project's currently staged changes**. )
2. Git status

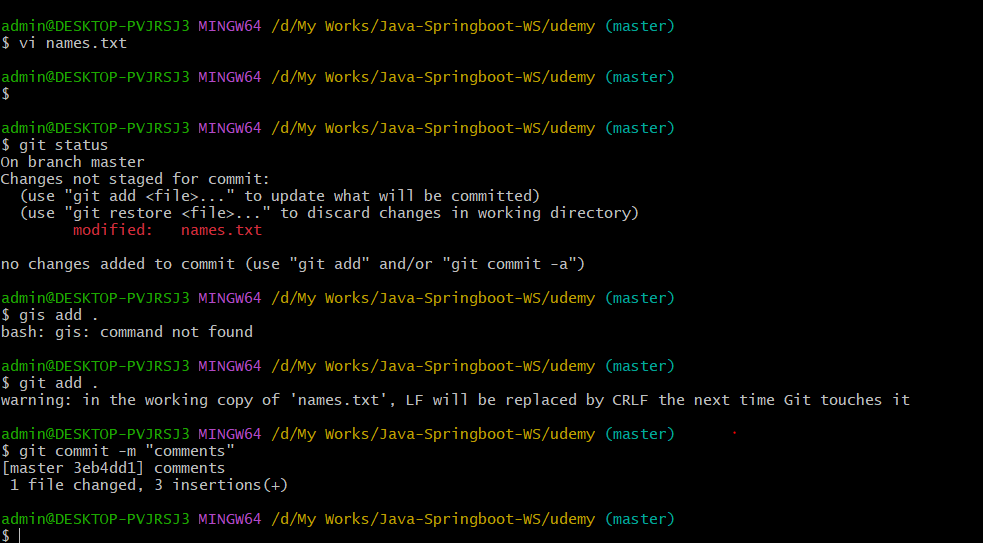


1. Vi names.txt – open file in vi editor
2. Press a and type some words
3. escape then :wq! – to save changes in file
4. Cat names.txt – to display contents of files
5. Git status

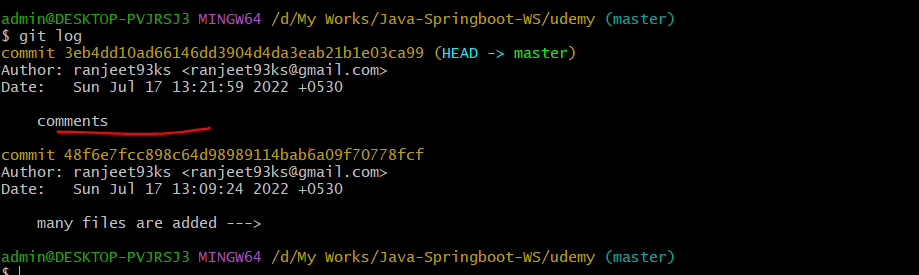


1. $ git restore names.txt – to discard changes else git add names.txt and git commit -m “comment”
2. Git status

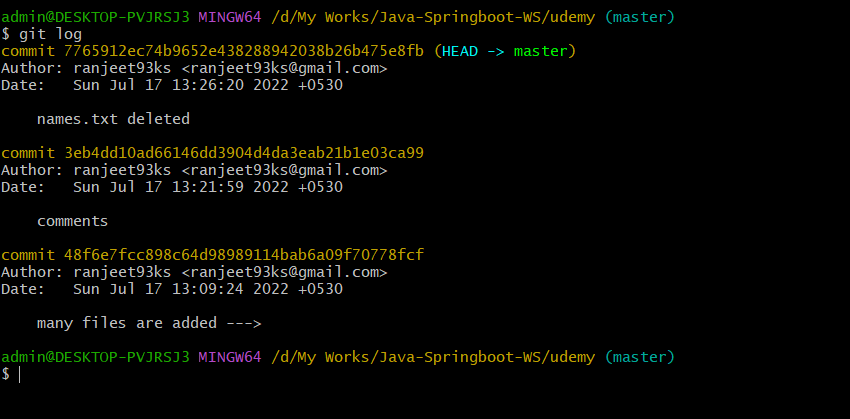




1. Git log : to check history of all changes :-

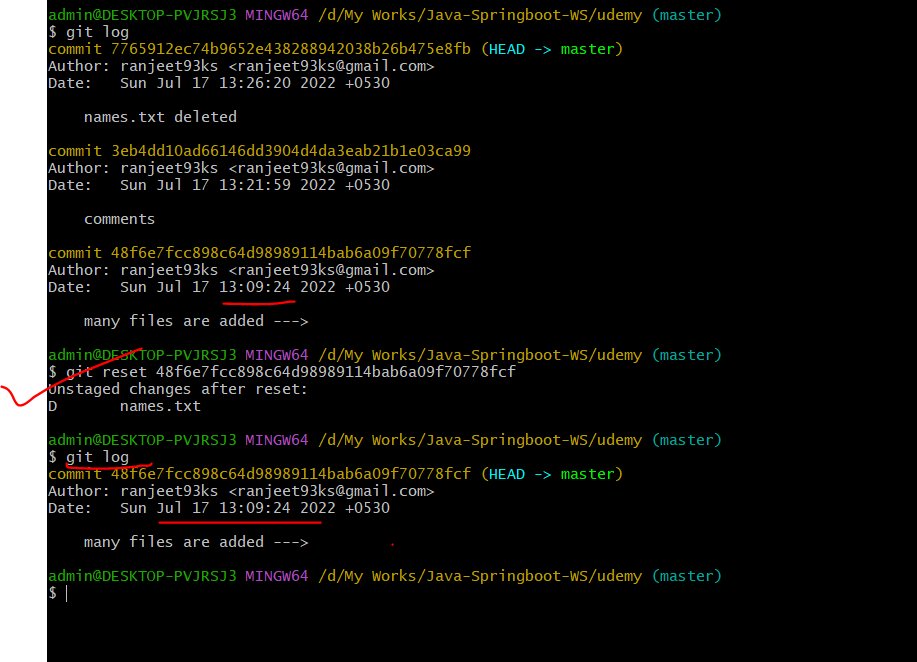


1. rm names.txt to delete file
2. git add . & git commit -m “deleted”
3. git log

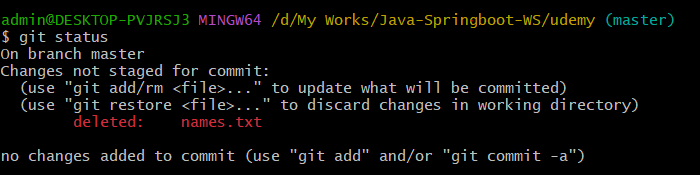


So, total 3 commits till now with commit id mentioned in above pic. Now. Let say we wrongly changes the file n commit and so I want to restore the changes back to previous 1. So, we need to take commit id just before that

1. git reset 48f6e7fcc898c64d98989114bab6a09f70778fcf – **all commit before it – will be removed**

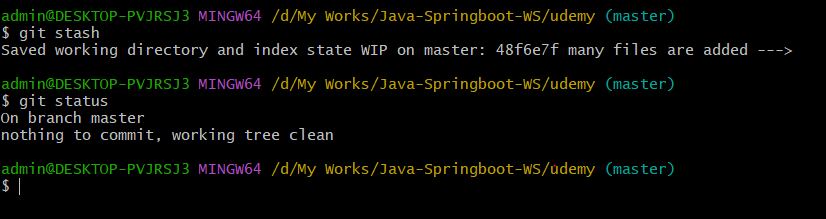


1. **git status**

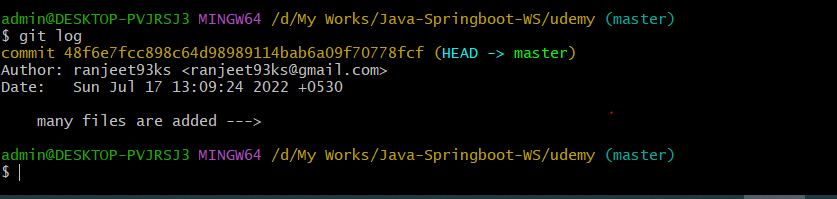


**Discarded changes are unstaged, but now we want to remove it from history but we don’t want to changes these files for that git stash**

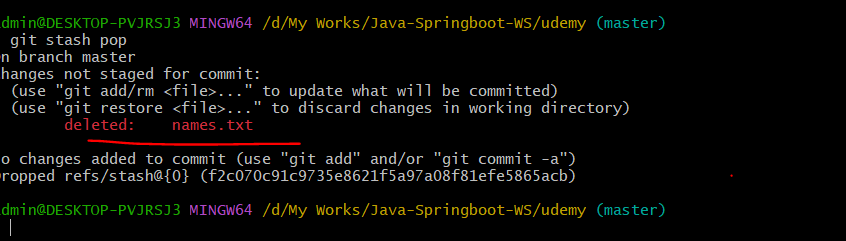
1. **git stash :--** Saved working directory and index state WIP on master: 48f6e7f many files are added --->
2. **git status**

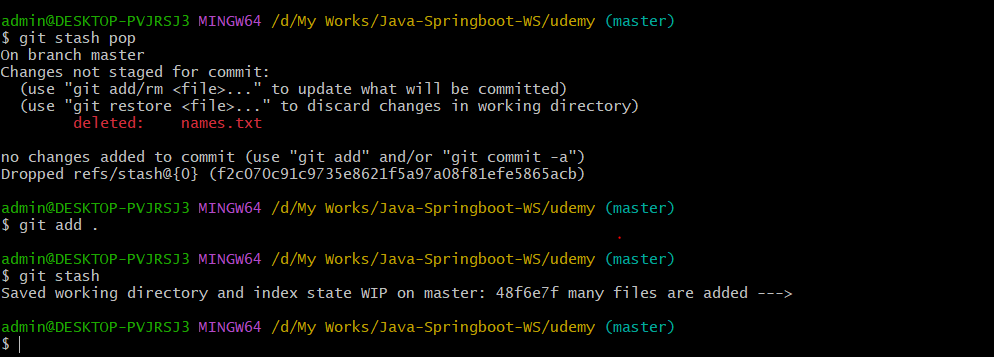


1. **git log :-- now my project folder looks like 1st commit**



1. **git stash pop : - files back to staged area**

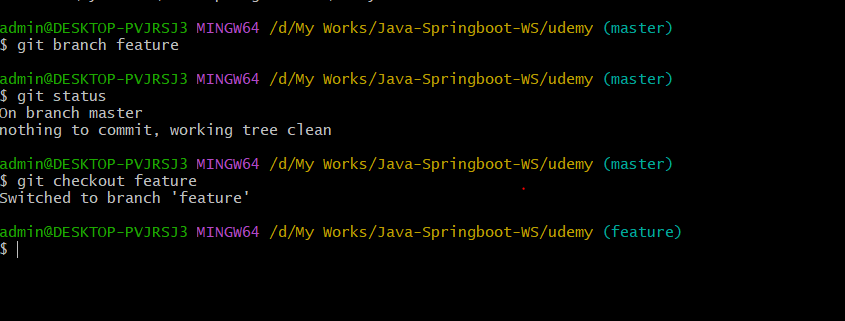


1. **Again discard** 
2. **git stash clear – to remove from seperate places as well**

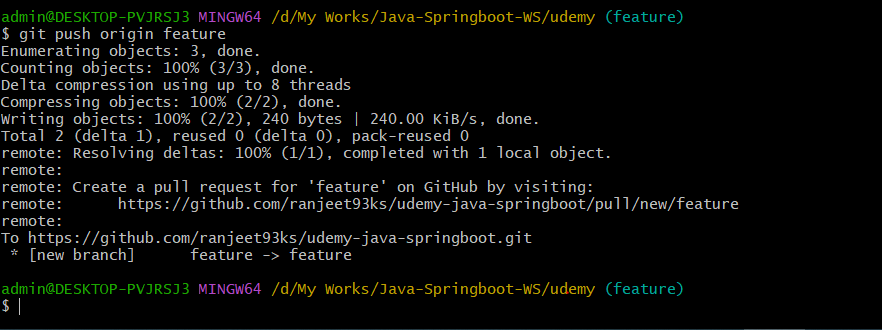
------------All about git--------------------🡪>>>>>>

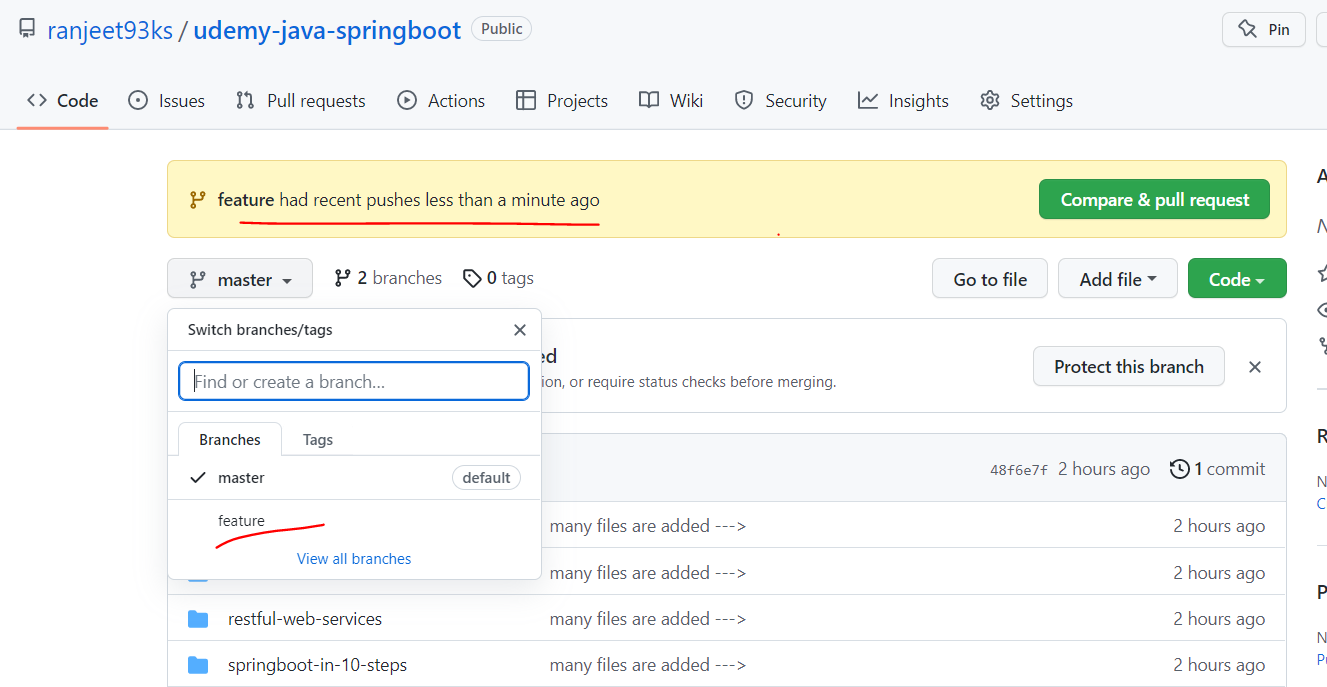
* **🡪 Github.com 🡪 🡪**

1. Visit github.com and access your profile after signin or signup
2. Click on “+” and create new repository
3. Name anything “udmey-java-springboot” keep it public and Create repository
4. Now git remote add origin <https://github.com/ranjeet93ks/udemy-java-springboot.git> -- url of newly created repo (origin is alias for this url – for all url default alias is origin)
5. Git push origin master (to push our changes to this url) – master is default branch
6. If they will for authorization to github account –approve it –all changes will be pushed to that repository
7. Refresh the url – we will get all changes
8. Branching -🡪 git branch feature (feature branch will be created)
9. Git checkout feature :- Select feature branch – now all changes will go to feature branch



1. Now, add some files using touch feature.txt – git add . n git commit -m “feature.txt added”
2. git merge feature : to add all changes to master – no feature branch will create in url – only changes will be added in master branch
3. **git push origin feature** :-- all changes will now reflect in url **feature branch .**





1. **how to work on someone’s else code** 🡪 fork the repository to your account and using new url clone the repository to ur local and do whatever changes u want.

**Git clone url , import in ide n do changes**

Note: -**Upsteam url** is source url from whi=ere u have forked the project

**git remote add upstream** “upsteamed url”

origin is forked one

**git remote -v origin** “forked url”

Note:- 1 pull request means 1 branch. So if u working on 10 features of an application, create new branch so that diff pull request for diff features ,otherwise all of commits will be added to same pull request and it will be difficult for reviewer to review ur changes and approve ur changes to be merged to main.

Even we can raise pull request from feature branch to be merged to main/master branch of same repo and after approval changes file will be added to master branch.

1. git push origin feature -f (forced update), if we reset any commit, cuz all commits are interlinked
2. **fetch upsteam** :-- get latest changes to fork url cuz after u forked there seems someone committed to upsteam.
3. To take latest changes to already existing project locally :-  
   cd “project”   
   git pull origin master