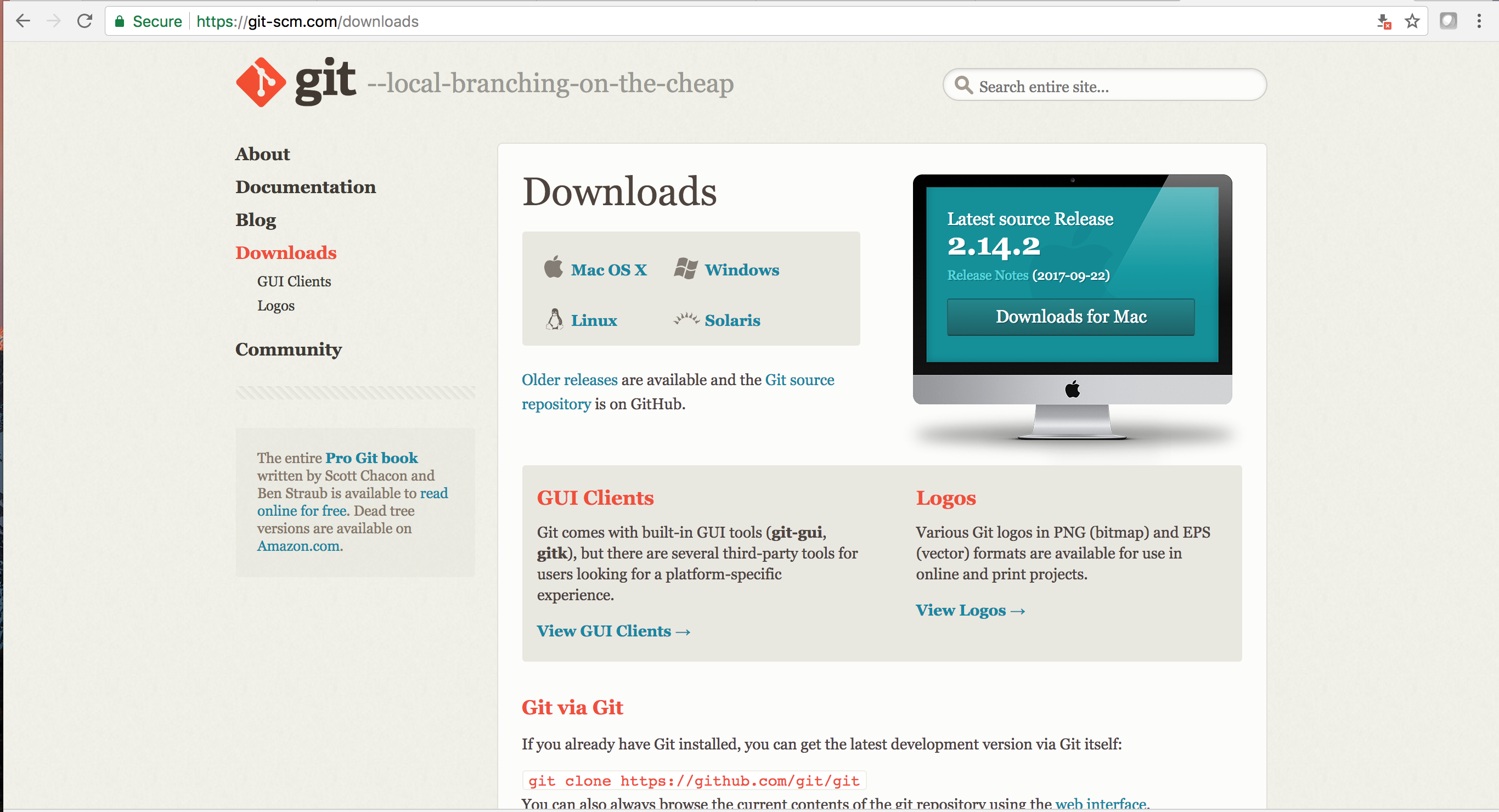
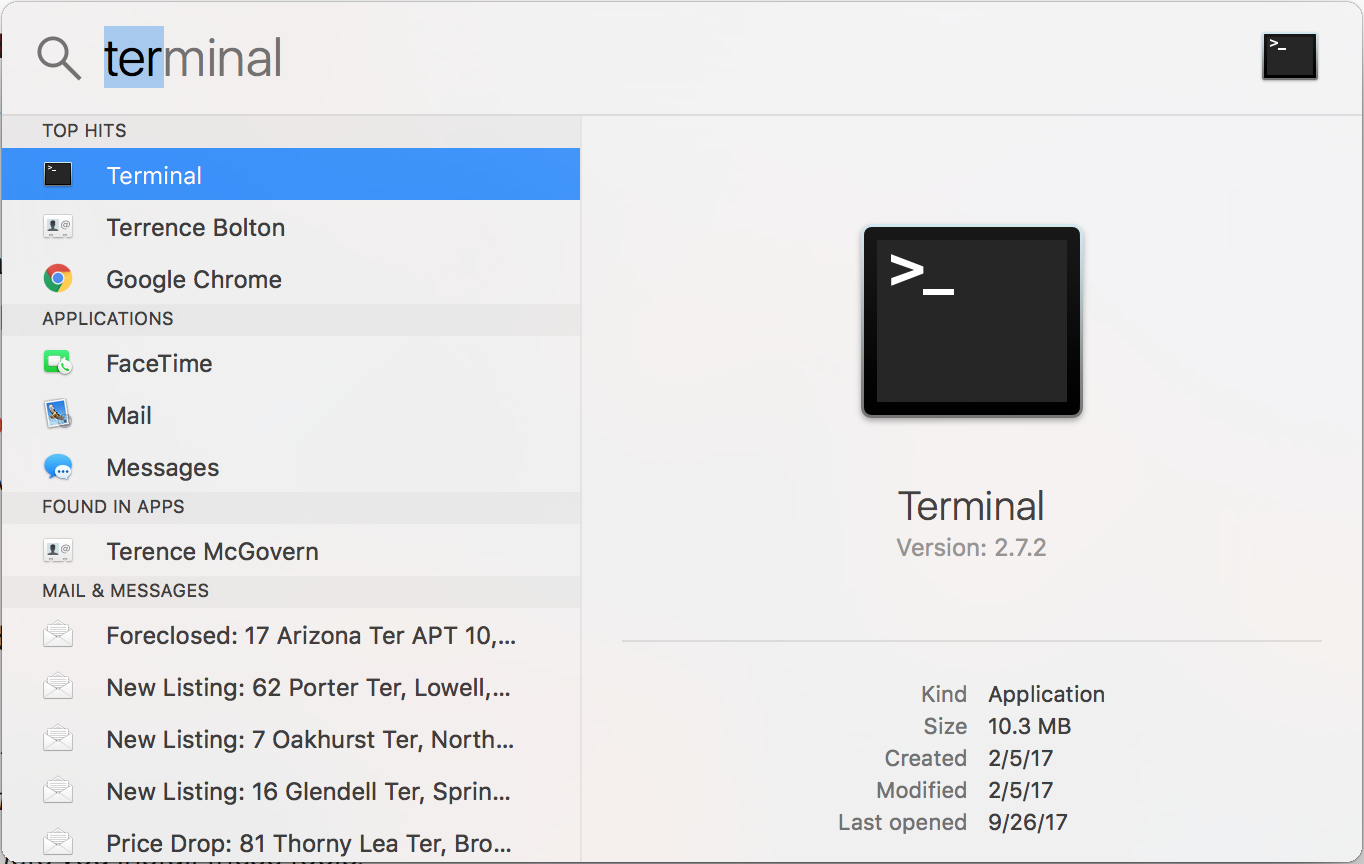
**Git Workflow**

**Installing GIT on Desktop**

* Get access to the Github repository
* Install Git bash on desktop



* Open bash terminal (For Windows: open git bash terminal, For Mac: just open terminal)



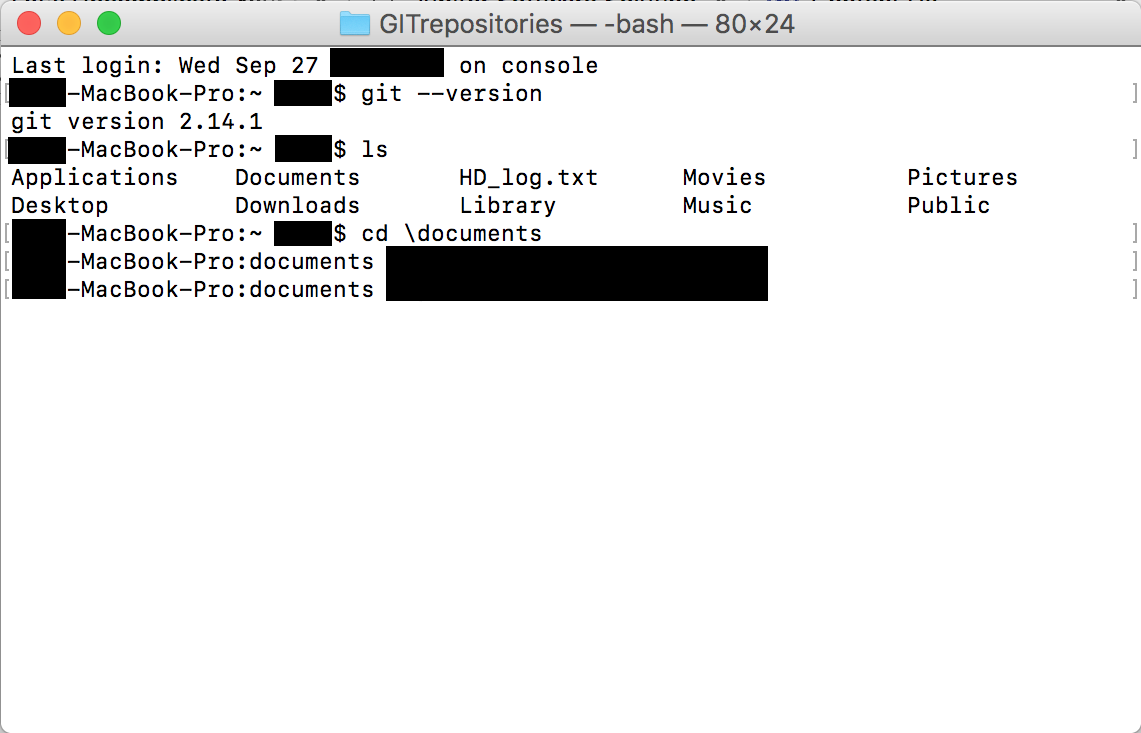
**Inside the Terminal…**

Check whether git is installed or not

* **Command:** git --version

Change directory to documents

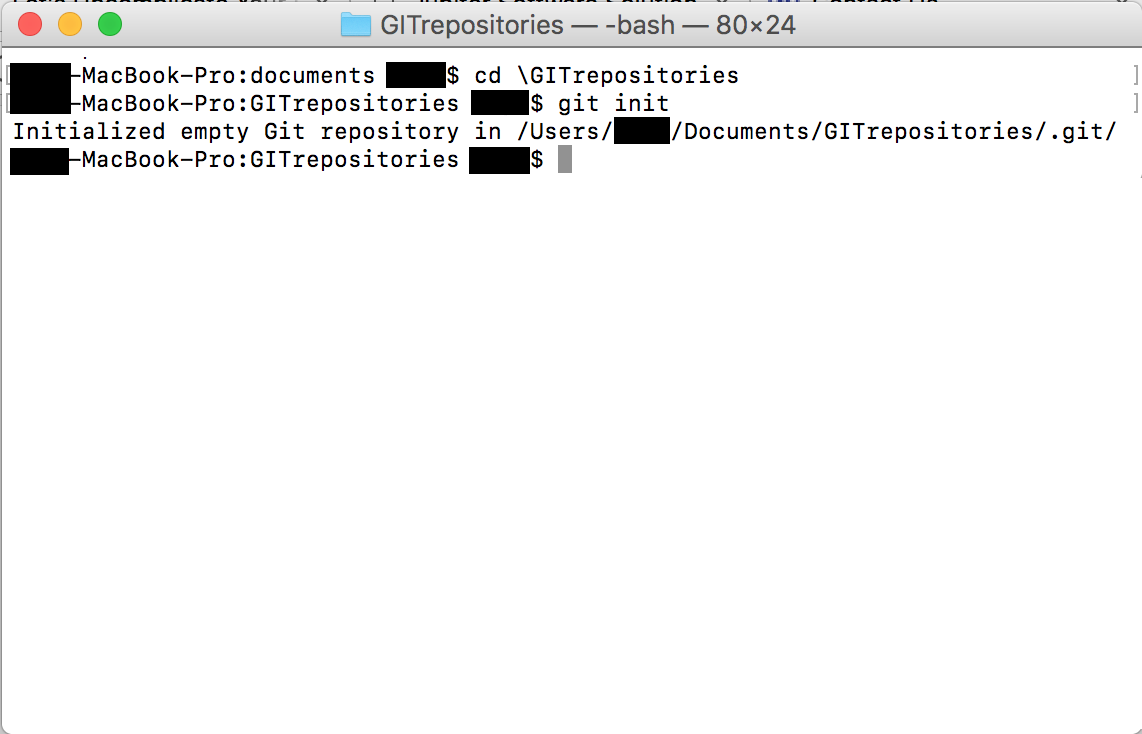
* **Command:** cd \documents



Create a sub-folder if you wish to. If you do create a folder inside ‘Documents’, a folder to store all your GIT repositories (in below example User created a folder called GITrepositories inside documents folder), then use cd command to change path into that sub-folder. And once you are inside a desired folder…

Initiate git repository (if using for the first time)

* **Command:** git init
* This command creates a git repository in the current folder. However, by default this folder is considered as the master.



Generating a SSH key

* **Command:** ssh-keygen -t rsa -b 4096 -C "email-registered-on-github"
* While creating SSH
* Do not give any name to the file. Just keep hitting enter.
* You may or may not assign a passphrase. If you do assign remember it.

Copying the RSA key

* **Command:** clip < ~/.ssh/id\_rsa.pub
* Enter this key in online github portal SSH key tab under account settings. Provide PC name and add the SSH key.

Cloning the repository to local machine/desktop

* **Command:** git clone git@github.com:colaberry/Filtered.AI.git

Change directory to the cloned repository

* **Command:** cd Filtered.AI

Pull the latest version from the master source on github

* **Command:** git pull

Create a branch within the master git repository

* **Command:** git checkout -b sathwik-test

\*\*\* Perform required Operations/edits to the code or documents \*\*\*

After making changes you can check status of the repository to show and review all the changes made

* **Command:** git status

Add the file/files that has/have been changed

* **Command:** git add <filename> (Eg. git add README.md)

Commit the changes in remote source to online github repository

* **Command:** git commit –m “Updates Readme\*\*”

Push the changes to the master repository to which you are a collaborator

* **Command:** git push --set-upstream origin sathwik-test

Pulling the changes from branch

* **Command:** git pull

Rebasing

* **Command:** git rebase –i <master>

(Do this while you are on the branch which you want to rebase with the master)

Within the vi editor

You may have many comments based on the actions. If you have multiple comments it may be messy so to change the comments we do some editing. Comments are displayed at the top of the vi editor. Pick is one of the keywords. If you want to squash the second comment, click before pick (not anywhere else) and delete pick. Then press “I” key to insert new commands. Type in “s” or “squash”. Check spaces as well. Then once press Esc to get into normal mode and then type in “:wq” to quit

To check differences between this branch and other

* **Command:** git diff <branchname>

(do this while being on master)

To merge contents of a master to another branch or one branch to another branch

* **Command:** git merge –squash sathwikv-test

To check on which branch you are type in the following command

* **Command:** git branch

To check activity log

* **Command:** git log

To check status

* **Command:** git status

To add all files to the git repository

* **Command:** git add .

To delete files in the repository

* **Command:** git rm labs/German\_Credit\_KY.ipynb

**Untracked files**

When you have untracked files in the repository, they are not added to the commit. It may become tricky as the git control doesn’t allow you to re-do the file addition step (git add) for the same file again. So how to deal with untracked files:

Viewing untracked files

* **Command:** git clean –n

Deleting all untracked files

* **Command:** git clean –f

Adding all untracked files

* **Command:** git add --all

**How to delete branches**

If a branch has been created and pushed onto remote repository also, then the branch has to be deleted from both the local machine and the remote location as well. The process to delete the branches is as follows:

Deleting a branch from local machine: (Note that you cannot delete a branch while you are on that branch, always switch to master before deleting a branch. Use git checkout command to switch to master branch)

* **Command:** git branch -d feature/login

Deleting a branch from remote

* **Command:** git push origin --delete feature/login