pwd : present working directory

ls : content of current directory

cd : change directory

**working with SSH keys :**

SSH keys are necessary to connect with the github server

There are two types of ssh keys , one is public and one is private:

public key is denoted by file\_name.pub and privatekey is denoted by file\_name

**\* To check is SSH key was already created :**

open git bash

ls -al ~/.ssh :::: this will list all the files realted to .ssh

NOte : If you don't have a supported public and private key pair,

or don't wish to use any that are available,

generate a new SSH key.

If no ssh key was found we generate a new key using :

ssh-keygen -t rsa -b 4096 -C "tanveer.ahmed@brillio.com"

This above code will promt a messasge to enter file name :

enter the name for the ssh key : example : testkey

You will be asked to input a pass phrase , ignore this for now .

two files will be generated after the process of generation is done :

testkey.pub and testkey

You put the public key in new ssh key in settings of the github

The private key must be inserted in the ssh-agent via git-bash cli

steps for ssh private key :

eval "$(ssh-agent -s)" : this evaluates if the ssh-agent is present or not

[Generating a new SSH key and adding it to the ssh-agent - GitHub Docs](https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent)

some knowledge about ssh keys :

public key : it is stored in the github main site .

private key : it is stored in your laptop

while making a request , your laptop will send this private key and try to match it

with the public key if matched the changes will take place

* ALL your private key and public keys are strored in users/.ssh folder

use “cat ~/.ssh/keyname “ to output the content of the files

to add the private key into the ssh-agent use the following command :

**IMPORTANT GIT COMMANDS**

* git config –global user.name “[name]” ->sets author name  
  git config –global user.email “[email address]” ->sets author email id
* git init [repository name] ->start new repository
* git clone [url] ->obtain a repository from an existing URL.
* git add [file] ->adds a file to the staging area.
* git commit -m “[ Type in the commit message]” ->records or snapshots the file permanently in the version history.  
  git commit -a ->commits any files you’ve changed since then.&commits any files you’ve added
* git diff ->shows the file differences which are not yet staged.  
  git diff –staged ->differences between the files in the staging area and the latest version present.  
  git diff [first branch] [second branch] ->differences between the two branches mentioned.
* git reset [file] ->unstages the file, but it preserves the file contents.  
  git reset [commit] ->undoes all the commits after the specified commit and preserves the changes locally.  
  git reset –hard [commit] ->discards all history and goes back to the specified commit.
* git status ->command lists all the files that have to be committed.

9 ) git rm [file] ->deletes the file from your working directory and stages the deletion.

* git log ->used to list the version history for the current branch.  
  git log –follow[file] ->lists version history for a file, including the renaming of files also.
* git show [commit] ->shows the metadata and content changes of the specified commit.
* git tag [commitID] ->used to give tags to the specified commit.
* git branch ->lists all the local branches in the current repository.  
  git branch [branch name] -> creates a new branch.  
  git branch -d [branch name] -> deletes the feature branch.
* git checkout [branch name] -> used to switch from one branch to another  
  git checkout -b [branch name] ->creates a new branch and also switches to it.
* git merge [branch name] ->merges the specified branch’s history into the current branch.
* git remote add [variable name] [Remote Server Link] ->used to connect your local repository to the remote server.
* git push [variable name] master ->sends the committed changes of master branch to your remote repository.  
  git push [variable name] [branch] ->sends the branch commits to your remote repository.  
  git push –all [variable name] ->pushes all branches to your remote repository.  
  git push [variable name] :[branch name] ->deletes a branch on your remote repository.
* git pull [Repository Link] ->fetches and merges changes on the remote server to your working directory.
* git stash save ->stores all the modified tracked files.
* git stash pop ->restores the most recently stashed files.  
  git stash list ->lists all stashed changesets.  
  git stash drop ->discards the most recently stashed changeset.
* git checkout -b [name of the branch] -> to create a new branch  
  git checkout [name of the branch] -> to navigate into the named branch  
  git checkout -d [name of the branch] -> to delete the branch  
  git merge [branch name] -> to merge the branch to current branch in working directory

git branch -M <branchname > : changes the branch name for the main branch

git remote -v : shows you all the remote conneections in deatils