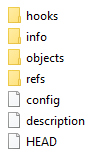
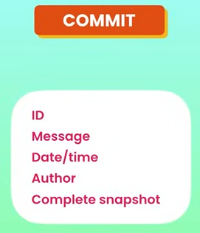
Use VS Code, instead of Gitbash.   
Open VS Code in required location/directory or navigate to one. From the top ribbon  or bottom left  , bring up the terminal. To bring up powershell , command prompt, or git bash click the + icon  and select bash. You can now type git commands normally here within VS code.

Within Terminal of VS Code :   
To create a new directory/folder: **mkdir Name\_of\_Folder**  
To go into (change) a directory/folder **within** current working directory: **cd Name\_of\_folder**  
Put the name of folder within inverted commas if there is whitespace in the name: **cd “Computer Science”**Change drive: **cd D:**  
List files: **ls**List **all** files including hidden ones, will show the .git file : **ls -a**  
To **clone** or download the exact files/repo from remote Github Server onto our local machine/laptop/pc: **git clone “—https link from git website”**  
After cloning , on our local machine we may make some changes to existing files(**modify**) or create new files(untracked by git). To check whether the files on my machines is different from that in Git server: **git status**  
**status** could be – **Untracked, modified, staged, unmodified.   
Untracked files** do not have any snapshot taken, so git has no idea about it.  
**Staged files** have their snapshot taken, but not yet committed to git- ready to be committed. The new or modified files in our working directory is to be added to the Git staging area: **git add “filename”**  
Even when a file has been deleted from our working directory (say file1.txt), the changes have to be added to the staging area: **git add file1.txt**Add all the files together: **git add .**  
  
This **does not add** the changes and files **to the remote Github server**. We need to push the changes onto Github server, so there is a local copy as well as Git server copy.   
Autocomplete with Tab  
Clear screen with clear

To create your own repository, navigate to a folder/directory where you want to create this repo. Open VS Code here (right click, open with code), use the terminal: **git init**This creates a new empty repository on your local machine. The terminal output will read something like this:   
  
Note this creates a new hidden folder/sub-directory called **.git.** Type **ls -a** to list this. View the contents of this folder through windows (view hidden files), looks like this. Its contents are all implementation details by git. Our original folder won’t be a git repository anymore if we delete this git directory (hence hidden).

********To create some new file and wrtie to it through terminal(standard UNIX command at cmd): **echo hello > filename.txt**To remove files (not git command, standard UNIX command): **rm filename.txt**  
(Filename with extension must be written)Git specifically provides a **staging area/index** for reviewing before finally commiting to the git sub-directory on our local machine.  
Once the files have been **committed** i.e, a **permanent snapshot** of those files are taken **within our git directory**, those *files still remain behind in the staging area.*  
Before commiting, one might unstage a particular file and commit that as part of next snapshot.   
*Every commit* has an ID, Message, Date\Time stamp, Author and a **complete snapshot of our complete project upto that point of time.** This way it is easier to restore our project to an earlier stage, no need to calculate changes and add them up. Git internally uses compression and other techniques(like not storing duplicate content) to do this, otherwise soon the size will blow up.

