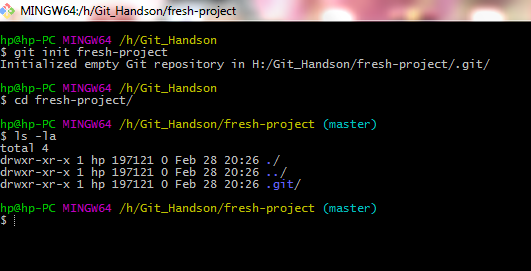
While doing the below labs, you need to take screenshot for each step, for each assignment and create word document. Upload the word document to topgear.

1. Set the global configuration file with your user name, email and editor as Notepad++. List all the properties which you just set.



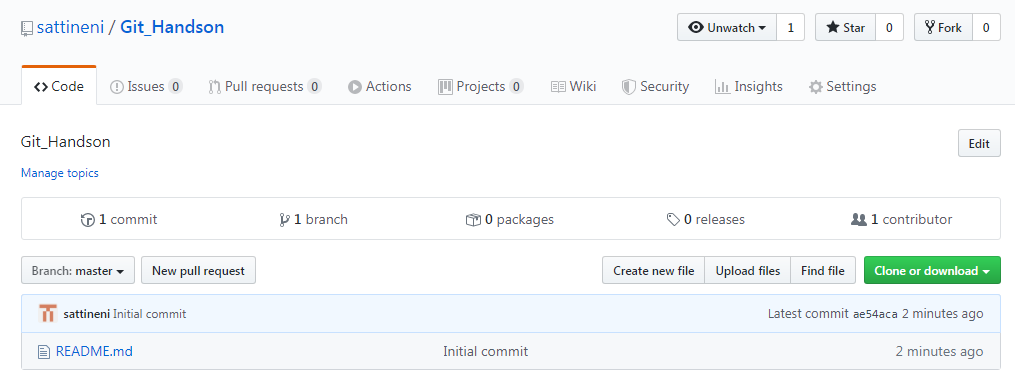
1. Make a fresh Git project

Git init fresh-project

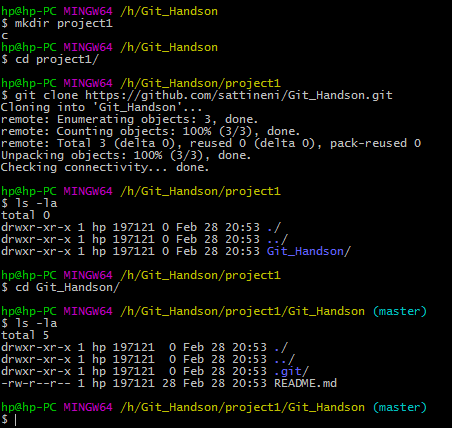


1. Create a Github account (Or use the account if already registered).Clone a project from the remote repository to your local repository

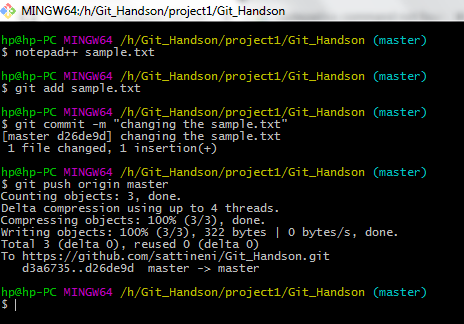
Created GitHub account and created Git\_Handson repository



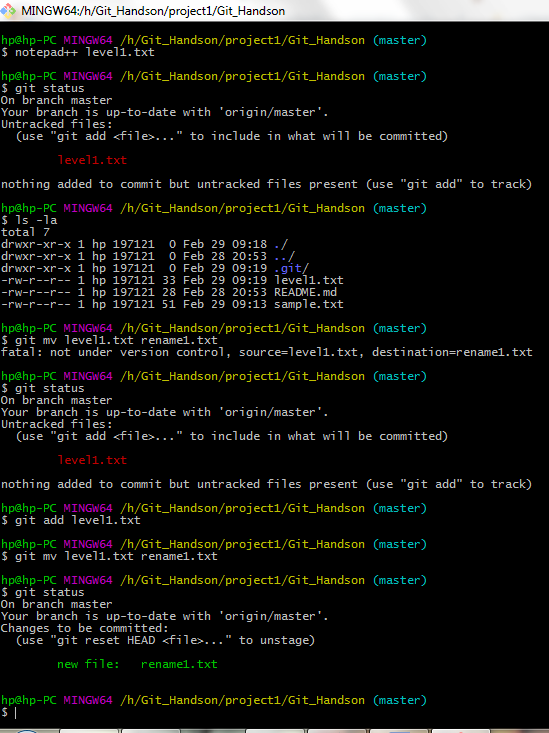
Git clone <https://github.com/sattineni/Git_Handson.git>

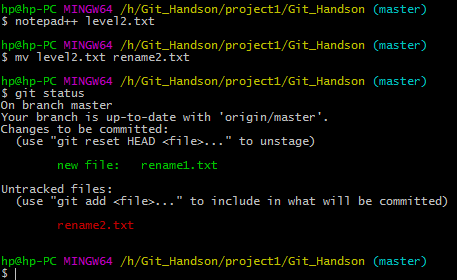


1. Push the project created in assignment 2 to the remote repository.

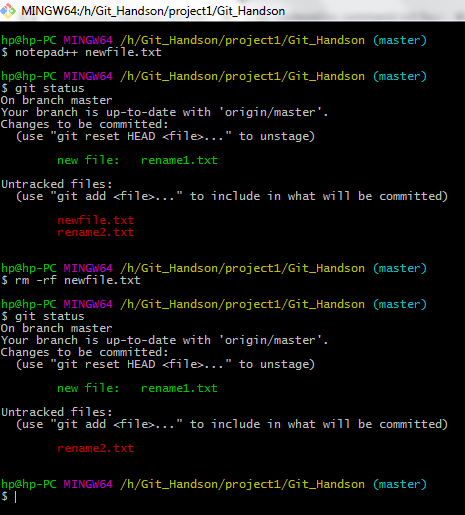


1. Use the different ways of renaming and moving files

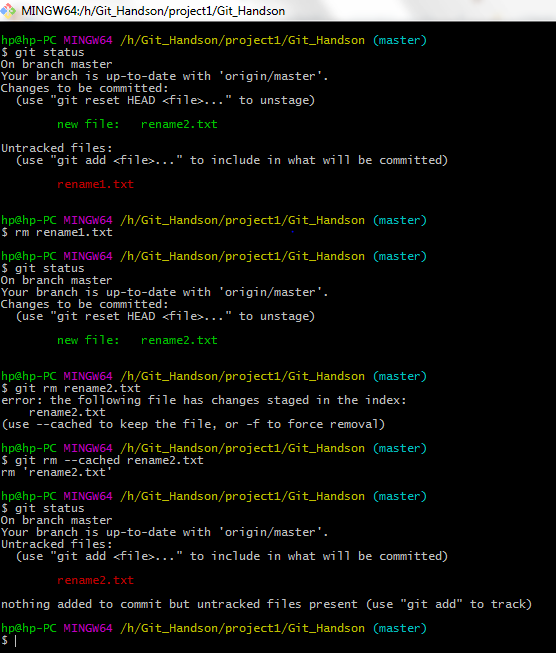


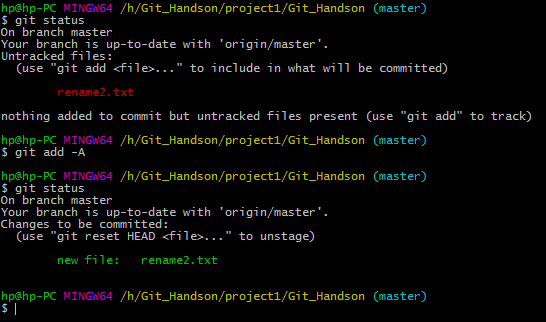


1. You just created a new file, but then you decided that the file is to be removed. How do you delete this untracked file.



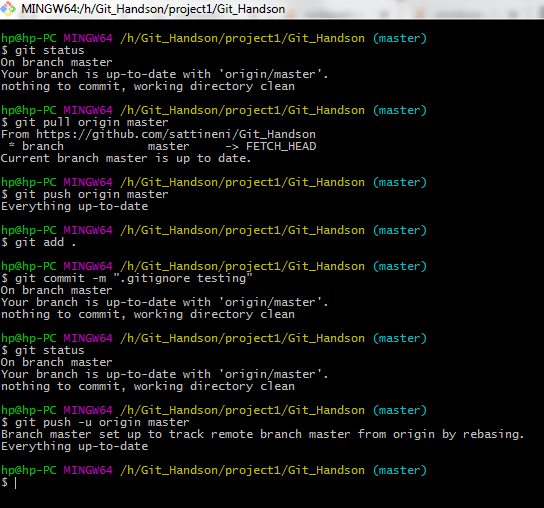
1. Demonstrate the following:
   1. delete of a tracked file
   2. backing out staged deletion
   3. recursive deletion

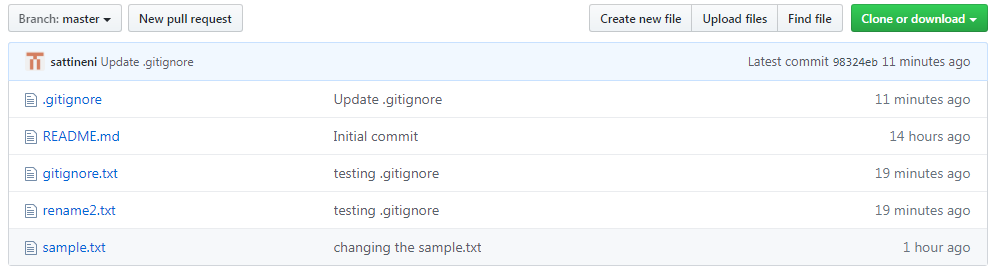




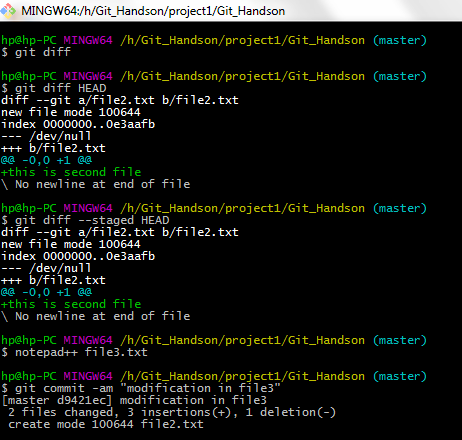
1. You do not want to push certain folders/files of your project;how do you manage this?

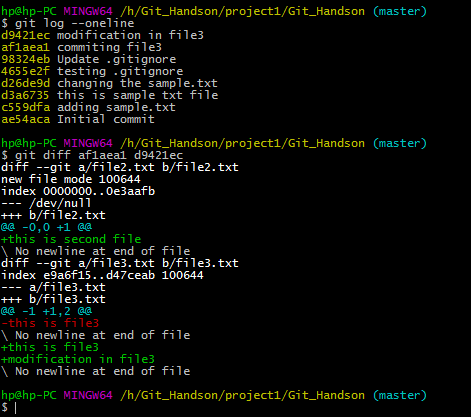
By using .gitignore we can ignore unwanted files and folders.

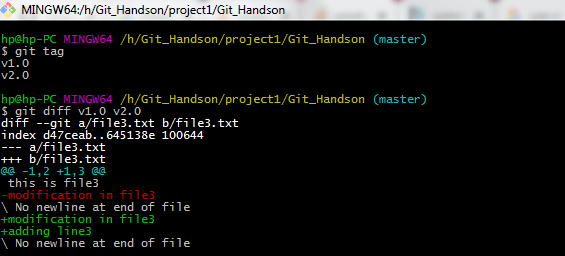


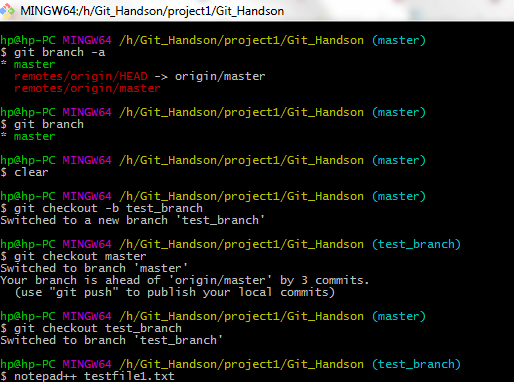


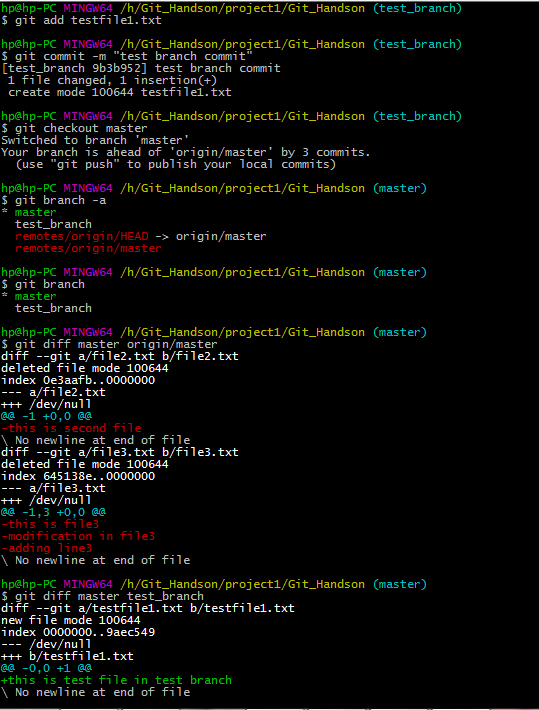
1. Create a branch called “test”. Make some changes in the master branch. Let there be some changes in the working directory and some in the staging area. Make some changes in the test branch as well. Issue the command to show the differences for
   1. Working directory vs Staging area
   2. Working directory vs Local Repository
   3. Staging area vs Local Repository
   4. Between two commits
   5. Between two tags
   6. Local vs Remote Repository
   7. Master branch vs test branch



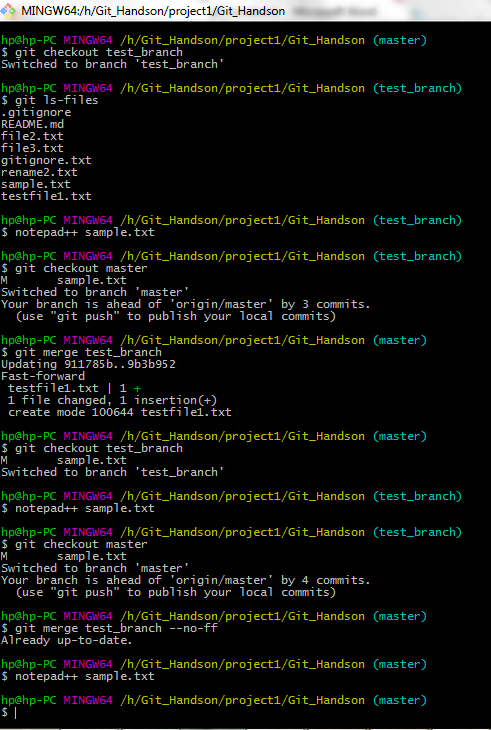








1. Merge the changes from test branch to master branch.
   1. FastForward merge
   2. Disabling FastForward merge
   3. What is the difference between option a and option b

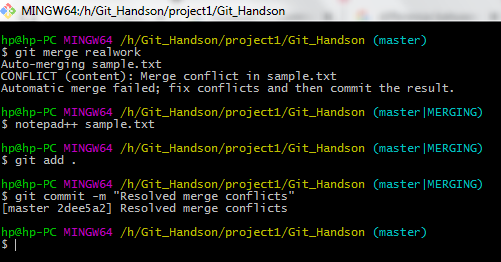


A fast-forward merge can occur when there is a linear path from the current branch tip to the target branch. Instead of “actually” merging the branches, all Git has to do to integrate the histories is move (i.e., “fast forward”) the current branch tip up to the target branch tip. This effectively combines the histories, since all of the commits reachable from the target branch are now available through the current one.

In the event that we require a merge commit during a fast forward merge for record keeping purposes we can execute git merge with the --no-ff option.

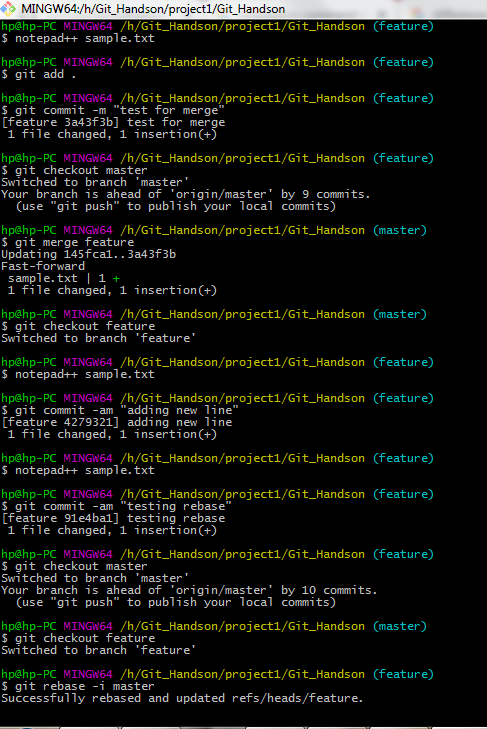
**git merge test\_branch --no-ff**

1. Create a merge conflict situation. Resolve the conflict and merge the changes between the branches.



1. What is the difference between merge and rebase, demonstrate with the eg.

Both **merge and rebase** can be used to **combine** two branches. **Merge** command just unify our work **with a** commit without changing history. While **rebase** apply feature branch changes on top **of** master branch and change the history. If we prefer to have clean history, then we can use **rebase.**



1. With an example, demonstrate fetch, clone and pull. What is the usecase for these operations. Are they same, different? Explain

**Fetch:**

We download changes to our local branch from origin through fetch. Fetch asks the remote repo for all commits that others have made but we don't have on our local repo. Fetch downloads these commits and adds them to the local repository.

**git fetch origin master**

**Clone:**

The "clone" command downloads an existing Git repository to our local computer.

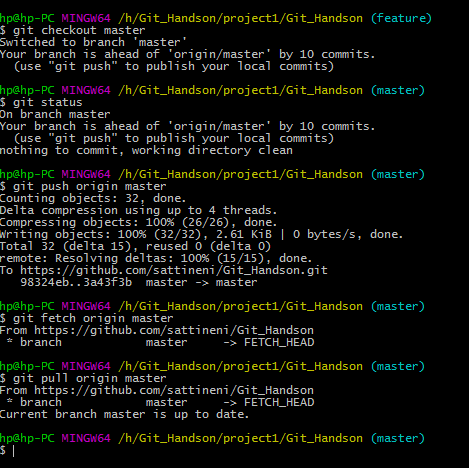
**git clone** [**https://github.com/sattineni/Git\_Handson.git**](https://github.com/sattineni/Git_Handson.git)

**git clone** [**git@github.com:sattineni/Git\_Handson.git**](mailto:git@github.com:sattineni/Git_Handson.git)

**Pull:**

The Pull command update our local branch with the origin/main branch i.e. actually what it does is combination of Git Fetch and Git merge one after another. But this may Cause Conflicts to occur, so it’s recommended to use Git Pull with a clean copy

**git pull = git fetch + git merge**



1. Create a new repository in Github, with a README file. While pushing to the remote repository, if the remote branch is ahead of the local repository (new file is added in remote repository, which is not there in local repository) and pull is failing, how do you solve this problem?