Some commonly commands which use in git:

1. Git init: create a new git repository.
2. Git clone: clone git repository to your local machine.
3. Git status: display the state of the working directory and the staged snapshot.
4. Git add <filename>: to add all the files
5. Git commit -m ‘some msg’: to commit all the changes from local to remote repository.
6. Git remote add origin master <server>: add project to server.
7. Git push -u origin master: push all changes to GitHub/Bitbucket server.
8. Git pull: pull all changes from repository to local.
9. Git branch ‘branch-name’: create new branch
10. Git checkout branch-name: pointer starts to point branch-name branch and leave master branch.
11. Git checkout -b branch-name: create new branch and switch to it in one step.
12. Git checkout merge: before merging any branch to master branch first enter into master branch
13. Git merge branch-name: merge branch with master branch and now pointer points to master branch
14. Git branch -d branch-name: delete branch
15. Git log: display history, like all commits and who did the commits.
16. Git diff: changes b/w two branches or changes b/w files in local.
17. Git fetch origin master: fetch latest history from the server
18. Git reset --hard origin/master: undo recent commit from local.
19. Git stash: save code on a stack for temporary so that user can safely switch the branch and move back to it and commit its code.
20. Git stash pop: remove changes from stack and place the temporary code in working directory.
21. Git stash list: display list of stash file
22. Git rm filename: remove file from repository and local both.
23. Git rm --cached filename: remove file from repository, not from local.

There are some scenarios which we generally observe in GitHub

**Scenario-1: Create a new repository in git ui and clone to your laptop.**

**Scenario-2a: create a development branch in local and push to server**

**Scenario-2b: create a development branch in ui and clone in local**

**Scenario-3: Import a Project which is already present in git**

**Scenario-4: make some change in local and revert it using get revert.**

**Scenario-5: Make a simple change in your project using git ui. Now pull this change to your local and observe the result. (Import others changes into your project when your project have no changes)**

**Scenario-6 Make a change in project using git ui and also make small change in your local in different line and committing both changes now pull repo changes in your local and observe the auto merge done by git. (Import others changes into your project when their change and your change is not in same line )**

**Scenario-7 Make a small change in project using GitHub ui and make a small change in local in same line and commit both changes and face your first merge conflicts. Now resolve this conflict by overwriting repo changes with your local changes. (Import others changes into your project conflicting changes)**

**Scenario-8 - Make a small change in project using GitHub ui and make a small change in local in same line and commit both changes and face your first merge conflicts. Now resolve this conflict by discarding your local change and accept repo changes**

**Scenario-9: make some change in local dev branch and push it and raise a pull request**

**Scenario-10: make some change in local dev branch and push it and raise a pull request this time review it and add some review comments and then merge it**

**Scenario-11: make a commit In local master and now discard it/ revert it , bring it back to same state as master**