**GIT COMMANDS:**

**SETUP:**

git config --global user.name “[firstname lastname]”: set a name that is identifiable.

git config --global user.email “[valid-email]”: set a mail id for identification.

git config --global color.ui auto: set automatic command line coloring for git for easy reviewing

git config --local: set local variable

git config --global: use global config file

**SETUP and INIT:**

git init : Initializing an existing directory as git repository.

git clone <url> : retrieve an entire repository from a hosted location via URL

git status : show modified files and status of the file.

git status –s: show status

git status –b: Get branch information

git status –u: show untracked file information

git status --long: show status in long format

git add <file> : add a file as it looks now to your next commit.

git add –all: add all files

git reset <file> : unstage a file while retaining the changes in working directory.

git reset –hard <commit id>:

--hard : will erase your local work if you have anything stashed.  
--mixed : Unstaging all changes but leave them in the working directory (default).  
-soft : staged and working directory are not altered

git revert HEAD:  the commit with a new commit to reverse changes.

git diff: diff of what is changed but not staged.

git diff –staged: diff of what is staged but not yet committed.

git commit -m “[descriptive message]”: commit your staged content as a new commit snapshot.

*git commit –amend* : enter commit message in nano or vim editor. Append previous commit.

git format-patch -1: create patch for recently changes.

git apply <patch file>: apply patch file

git am: applies a series of patchs.

git blame <filename>: Show file line numbers annotated with author information can also see in git log

git branch –a: list all the branches

git checkout <branch>: checkout or switch to branch

git checkout –b <branch>: create new branch and checkout to new branch

git checkout <commit id> : checkout to the commit id.

git diff –cached: show difference after adding file

git diff: show diff between local repo and git repo files.

git diff --staged # Same as 'git diff --cached' mostly used with latest version of git.

git diff e3b43d63 60fdba79 directory/path: show diff between two commit ids

git log: Show the Git log messages.

git log –oneline: Show the Git log messages in oneline

git log --graph --decorate –oneline: Show the Git log messages also displays graph

git log --author="name": displays author edited files log message

**Inspect & Compare:**

git log: show the commit history for the currently active branch.

git log branchB..branchA: show the commits on branchA that are not on branchB.

git log --follow [file]: show the commits that changed file, even across renames.

git show: **command line utility** that is used to view expanded details.

**Share & Update:**

git remote add [alias] [url]: A new connection to remote repository

git fetch [alias]: fetch down all the branches from that Git remote

git merge [alias]/[branch]: merge a remote branch into your current branch to bring it up to date

git merge branch-name: merge branch name to current branch

git merge –abort: stop the merge and return everything to its pre-merge state.

git mergetool: View file conflicts in a merge tool. Default is vimdiff

git push [alias] [branch]: Transmit local branch commits to the remote repository branch

git push master <branch>: push the files to particular branch

git push <local branch> master: push the files to local branch to remote branch

git pull: fetch and merge any commits from the tracking remote branch

git pull origin <branch>: Pull the data from remote repository to local repository

git tag: list all created tags

git tag -a <tagname> -m "Source used for April sprint demo": create a tag for given commit message

git tag -a April\_Sprint -m "Source used for April sprint demo" e3b43d63: ": create a tag for given commit message and for commit id.

**Tracking path changes:**

git rm [file]: delete the file from project and stage the removal for commit

git rm -r directory/path: delete recursive removal when given a directory name.

git mv [existing-path] [new-path]: change an existing file path and stage the move

git cp [existing-path] [new-path]: copy file from an existing file path and stage the move

git log --stat –M: show all commit logs with indication of any paths that moved

git ls-files –stage: list files staged files

git ls-files –cached: list cached files

git ls-files –modified: list modified files

git ls-files –others: list untracked files

git ls-files –deleted: list deleted files