1. Git Setup https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html

   sudo apt-get install git

2. Initialize a Git Repository

   git init

3. Add files to the repository

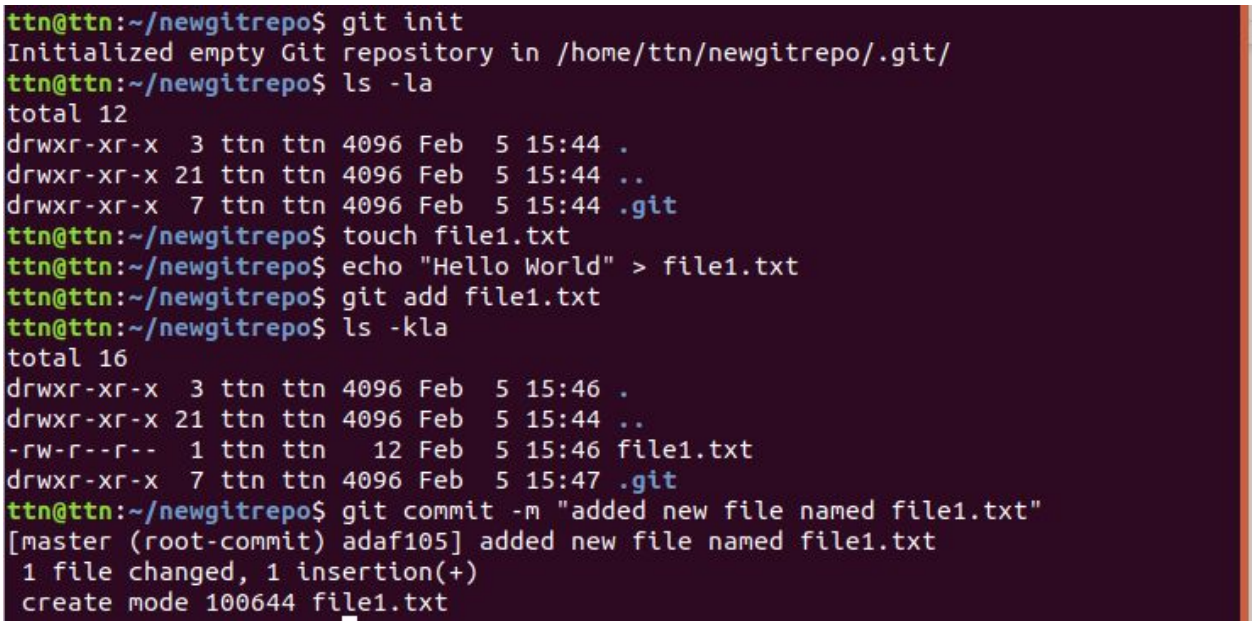
   git add deepak

4. Unstage 1 file

   git reset HEAD deepak

5. Commit the file

   git commit -m “added the file to my repository”



6. Add a remote

   git remote add origin git@github.com:tripathideepak1997/Project1.git

```
ttn@ttn:~/newgitrepo$ git remote add origin git@github.com:tripathideepak1997/Pr
oject1.git
ttn@ttn:~/newgitrepo$ git remote -v
origin  git@github.com:tripathideepak1997/Project1.git (fetch)
origin  git@github.com:tripathideepak1997/Project1.git (push)
```

7. Undo changes to a particular file

```
ttn@ttn:~/newgitrepo$ rm -f file1.txt
ttn@ttn:~/newgitrepo$ ls
ttn@ttn:~/newgitrepo$ ls -la
total 12
drwxr-xr-x  3 ttn ttn 4096 Feb  5 16:01 .
drwxr-xr-x 21 ttn ttn 4096 Feb  5 15:44 ..
drwxr-xr-x  8 ttn ttn 4096 Feb  5 15:55 .git
ttn@ttn:~/newgitrepo$ git checkout file1.txt
ttn@ttn:~/newgitrepo$ ls -la
total 16
drwxr-xr-x  3 ttn ttn 4096 Feb  5 16:01 .
drwxr-xr-x 21 ttn ttn 4096 Feb  5 15:44 ..
-rw-r--r--  1 ttn ttn   12 Feb  5 16:01 file1.txt
drwxr-xr-x  8 ttn ttn 4096 Feb  5 16:01 .git
```

8. Push changes to Github

   git push origin master

```
ttn@ttn:~/newgitrepo$ touch file2.txt
ttn@ttn:~/newgitrepo$ echo "def" > file2.txt
ttn@ttn:~/newgitrepo$ git add file2.txt
ttn@ttn:~/newgitrepo$ git commit -m "adding file2.txt"
[master fa69746] adding file2.txt
 1 file changed, 1 insertion(+)
 create mode 100644 file2.txt
ttn@ttn:~/newgitrepo$ git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 283 bytes | 283.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:tripathideepak1997/Project1.git
   adaf105..fa69746  master -> master
```

9. Clone the repository

```
ttn@ttn:~$ git clone git@github.com:tripathideepak1997/Project1.git Project2
Cloning into 'Project2'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
ttn@ttn:~$ cd Project2
ttn@ttn:~/Project2$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

10. Add changes to one of the copies and pull the changes in the other.

```
ttn@ttn:~/Project1$ ls
file1.txt  file2.txt  xyz
ttn@ttn:~/Project1$ git pull origin master
Warning: Permanently added the RSA host key for IP address '192.30.253.112' to t
he list of known hosts.
From github.com:tripathideepak1997/Project1
 * branch            master     -> FETCH_HEAD
Already up to date.
ttn@ttn:~/Project1$ cat file2.txt
def
def
```

11. Check differences between a file and its staged version

```
ttn@ttn:~/Project1$ echo "someone trying to edit" >file3.txt
ttn@ttn:~/Project1$ git add file3.txt
ttn@ttn:~/Project1$ git diff --cached
diff --git a/file3.txt b/file3.txt
index 363ef61..be67541 100644
--- a/file3.txt
+++ b/file3.txt
@@ -1 +1 @@
-some
+someone trying to edit
```

12. Ignore a few files to be checked in

```
ttn@ttn: ~/Project1
File  Edit  View  Search  Terminal  Help
ttn@ttn:~/Project1$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   file3.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        3.txt
        file
        file4.txt

ttn@ttn:~/Project1$ vim .gitignore
ttn@ttn:~/Project1$ vim .gitignore
ttn@ttn:~/Project1$ git add .gitignore
ttn@ttn:~/Project1$ git commit -m "Added the file that ignores 3.txt file file4.txt"
[master 4967bbf] Added the file that ignores 3.txt file file4.txt
 2 files changed, 4 insertions(+), 1 deletion(-)
 create mode 100644 .gitignore
ttn@ttn:~/Project1$ git push origin master
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 436 bytes | 436.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To github.com:tripathideepak1997/Project1.git
   575fdcc..4967bbf  master -> master
ttn@ttn:~/Project1$ git status
On branch master
Your branch is up to date with 'origin/master'.
```

13. Create a new branch.

```
ttn@ttn:~/Project1$ git branch test
ttn@ttn:~/Project1$ git branch
* master
  test
ttn@ttn:~/Project1$ git checkout <test>
bash: syntax error near unexpected token `newline'
ttn@ttn:~/Project1$ git checkout test
Switched to branch 'test'
ttn@ttn:~/Project1$ git branch
  master
* test
```

14. Diverge them with commits

```
ttn@ttn:~/Project1$ git checkout test
Switched to branch 'test'
ttn@ttn:~/Project1$ echo "Hello World " > newfile
ttn@ttn:~/Project1$ git add newfile
ttn@ttn:~/Project1$ git commit -m "new file added with some message"
[test 0e34094] new file added with some message
 1 file changed, 1 insertion(+)
 create mode 100644 newfile
ttn@ttn:~/Project1$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
ttn@ttn:~/Project1$ git checkout -b new_dir
Switched to a new branch 'new_dir'
ttn@ttn:~/Project1$ echo "helloworld to all " > newfile
ttn@ttn:~/Project1$ git add newfile
ttn@ttn:~/Project1$ git commit -m "message changed in newfile"
[new_dir 277cf75] message changed in newfile
 1 file changed, 1 insertion(+)
 create mode 100644 newfile
ttn@ttn:~/Project1$
```

15. Edit the same file at the same line on both branches and commit

```
ttn@ttn:~/Project1$ git branches
git: 'branches' is not a git command. See 'git --help'.
ttn@ttn:~/Project1$ git branch
  master
* new_dir
  test
ttn@ttn:~/Project1$ echo "Hello World for the first time" > new_file
ttn@ttn:~/Project1$ git add new_file
ttn@ttn:~/Project1$ git commit
Aborting commit due to empty commit message.
ttn@ttn:~/Project1$ git commit -m "File edited with some changes in it "
[new_dir a236973] File edited with some changes in it
 1 file changed, 1 insertion(+)
 create mode 100644 new_file
ttn@ttn:~/Project1$ git checkout test
Switched to branch 'test'
ttn@ttn:~/Project1$ echo "This time adding line from test branch to the new_file" > new_file
ttn@ttn:~/Project1$ git add new_file
ttn@ttn:~/Project1$ git commit -m "The file is changed at the same line as previous commit"
[test e9697c9] The file is changed at the same line as previous commit
 1 file changed, 1 insertion(+)
 create mode 100644 new_file
ttn@ttn:~/Project1$
```

16. Try merging and resolve merge conflicts

```
ttn@ttn:~/Project1$ git checkout test
Switched to branch 'test'
ttn@ttn:~/Project1$ ls
3.txt  file  file1.txt  file2.txt  file3.txt  file4.txt  newfile  new_file  xyz
ttn@ttn:~/Project1$ echo "Hello to test" > new_file
ttn@ttn:~/Project1$ cat new_file
Hello to test
ttn@ttn:~/Project1$ git add new_file
ttn@ttn:~/Project1$ git commit -m "added to new_file in test"
[test 8bf79f7] added to new_file in test
 1 file changed, 1 insertion(+), 1 deletion(-)
ttn@ttn:~/Project1$ git branch
  master
  new_dir
* test
ttn@ttn:~/Project1$ git checkout new_dir
Switched to branch 'new_dir'
ttn@ttn:~/Project1$ echo "Hello to new_dir" > new_file
ttn@ttn:~/Project1$ git add new_file
ttn@ttn:~/Project1$ git commit -m "added to new_file in new_dir"
[new_dir ddcab94] added to new_file in new_dir
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
ttn@ttn:~/Project1$ git merge new_dir
Already up to date.
ttn@ttn:~/Project1$ git merge test
Auto-merging newfile
CONFLICT (add/add): Merge conflict in newfile
Auto-merging new_file
CONFLICT (add/add): Merge conflict in new_file
Automatic merge failed; fix conflicts and then commit the result.
ttn@ttn:~/Project1$ cat new_file
<<<<<<< HEAD
Hello to new_dir
=======
Hello to test
>>>>>>> test
ttn@ttn:~/Project1$ nano new_file
ttn@ttn:~/Project1$ cat new_file
<<<<<<< HEAD
Hello to new_dir
=======
Hello to test
hello this time after merging conflict
>>>>>>> test
ttn@ttn:~/Project1$ git add .
ttn@ttn:~/Project1$ git commit -m "Conflicting message resolved"
[new_dir cd32b7f] Conflicting message resolved
ttn@ttn:~/Project1$
```

17. Stash the changes and pop them

```
ttn@ttn:~/Project1$ ls
3.txt  file  file1.txt  file2.txt  file3.txt  file4.txt  newfile  new_file  xyz
ttn@ttn:~/Project1$ echo "Checking stash " > new_file
ttn@ttn:~/Project1$ git status
On branch new_dir
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   new_file

no changes added to commit (use "git add" and/or "git commit -a")
ttn@ttn:~/Project1$ git add .
ttn@ttn:~/Project1$ git stash
Saved working directory and index state WIP on new_dir: cd32b7f Conflicting message resolved
ttn@ttn:~/Project1$ git stash list
stash@{0}: WIP on new_dir: cd32b7f Conflicting message resolved
ttn@ttn:~/Project1$ git stash pop
On branch new_dir
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   new_file

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (8af6974305b531eddb898c3ecc9b2652dd0ee776)
ttn@ttn:~/Project1$
```

18. Add the following code to your .bashrc file : color_prompt="yes"

```
parse_git_branch() {
git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
```

```
        }
        if [ "$color_prompt" = yes ]; then
        PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\]
$(parse_git_branch)\[\033[00m\]\$ '
        else
        PS1='\u@\h:\W $(parse_git_branch)\$ '
        fi
        unset color_prompt force_color_prompt
```

```
ttn@ttn:~/Project1$ nano ~/.bashrc
ttn@ttn:~/Project1$ . ~/.bashrc
bash: alias: deepak: not found
bash: alias: =: not found
bash: alias: touch /tmp/aliastesting: not found
ttn@ttn:Project1 (new_dir)$ git checkout master
```