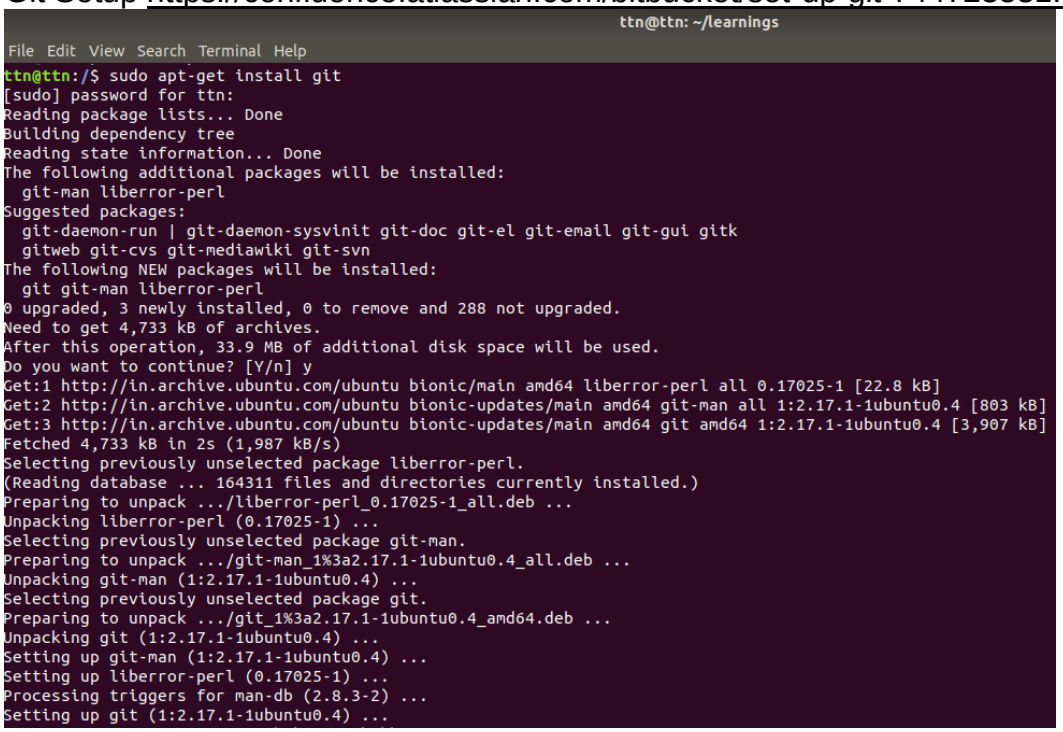
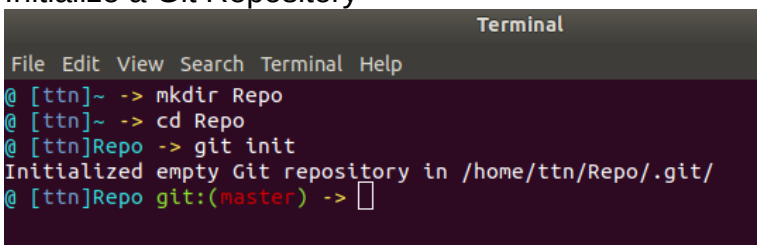
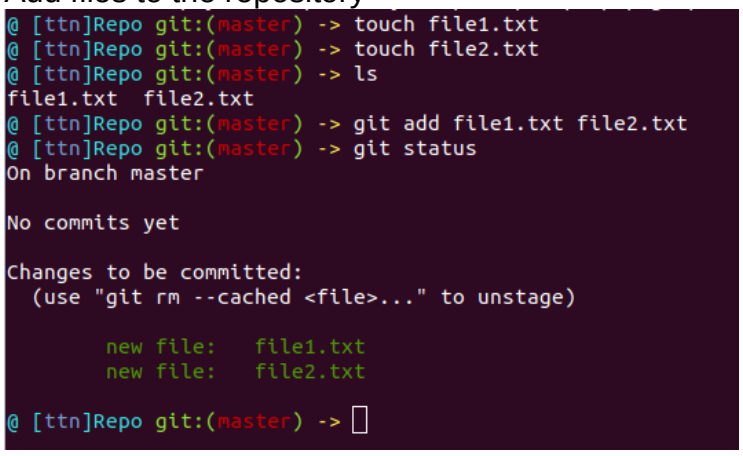
1. Git Setup https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html



2. Initialize a Git Repository



3. Add files to the repository



4. Unstage 1 file

```
@ [ttn]Repo git:(master) -> git reset file2.txt
@ [ttn]Repo git:(master) -> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   file1.txt

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

        file2.txt

@ [ttn]Repo git:(master) ->
```

5. Commit the file

```
@ [ttn]Repo git:(master) -> git commit file1.txt
[master (root-commit) 8f30ad6] Committing the first file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file1.txt
@ [ttn]Repo git:(master) -> git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        file2.txt

nothing added to commit but untracked files present (use "git add" to track)
@ [ttn]Repo git:(master) ->
```

6. Add a remote

```
@ [ttn]Repo git:(master) -> git remote add origin https://github.com/srishh/Repo
@ [ttn]Repo git:(master) -> git remote -v
origin  https://github.com/srishh/Repo (fetch)
origin  https://github.com/srishh/Repo (push)
@ [ttn]Repo git:(master) ->
```

7. Undo changes to a particular file

```
@ [ttn]Repo git:(master) -> git add file2.txt
@ [ttn]Repo git:(master) -> git reset --hard
HEAD is now at 8f30ad6 Committing the first file
@ [ttn]Repo git:(master) -> git status
On branch master
nothing to commit, working tree clean
@ [ttn]Repo git:(master) ->
```

8. Push changes to Github

```
@ [ttn]Repo git:(master) -> git push origin master
Username for 'https://github.com': srishh
Password for 'https://srishh@github.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 226 bytes | 226.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/srishh/Repo
 * [new branch]      master -> master
```

9. Clone the repository

```
@ [ttn]~ ->cd Cloned
@ [ttn]Cloned -> git clone git@github.com:srishh/Repo.git
Cloning into 'Repo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
Receiving objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
@ [ttn]Cloned -> ls
Repo
@ [ttn]Cloned -> cd Repo
@ [ttn]Repo git:(master) -> ls
file1.txt
@ [ttn]Repo git:(master) ->
```

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

```
@ [ttn]Repo git:(master) -> nano files.txt
@ [ttn]Repo git:(master) -> git add files.txt
@ [ttn]Repo git:(master) -> git commit
[master dac361c] Commited in the cloned repo
 1 file changed, 1 insertion(+)
 create mode 100644 files.txt
@ [ttn]Repo git:(master) -> 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), 305 bytes | 305.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:srishh/Repo.git
   8f30ad6..dac361c  master -> master
@ [ttn]Repo git:(master) -> cd ~/
@ [ttn]~ -> cd Repo
@ [ttn]Repo git:(master) -> git pull origin master
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
Unpacking objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
From https://github.com/srishh/Repo
 * branch            master     -> FETCH_HEAD
   8f30ad6..dac361c  master     -> origin/master
Updating 8f30ad6..dac361c
Fast-forward
 files.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 files.txt
@ [ttn]Repo git:(master) -> git status
On branch master
nothing to commit, working tree clean
@ [ttn]Repo git:(master) -> ls
file1.txt  files.txt
```

11. Check differences between a file and its staged version

```
@ [ttn]Repo git:(master) -> nano files.txt
@ [ttn]Repo git:(master) -> git diff
diff --git a/files.txt b/files.txt
index 6434b13..48c8357 100644
--- a/files.txt
+++ b/files.txt
@@ -1 +1,3 @@
 This is a new file.
+
+It has been changed.
```

12. Ignore a few files to be checked in

```
@ [ttn]Repo git:(master) -> touch .gitignore
@ [ttn]Repo git:(master) -> git rm --cached files.txt
rm 'files.txt'
@ [ttn]Repo git:(master) -> git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    files.txt

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

        files.txt

@ [ttn]Repo git:(master) ->
```

13. Create a new branch.

```
                                                    Terminal
File  Edit  View  Search  Terminal  Help
@ [ttn]Repo git:(master) -> git checkout -b practice
D       files.txt
Switched to a new branch 'practice'
@ [ttn]Repo git:(practice) ->
```

14. Diverge them with commits

```
@ [ttn]Repo git:(practice) -> nano file1.txt
@ [ttn]Repo git:(practice) -> git add file1.txt
@ [ttn]Repo git:(practice) -> git commit
[practice 2ddc6a0] Committed in practice branch
 1 file changed, 1 insertion(+), 1 deletion(-)
@ [ttn]Repo git:(practice) -> git checkout master
Switched to branch 'master'
@ [ttn]Repo git:(master) -> nano file1.txt
@ [ttn]Repo git:(master) -> git add file1.txt
@ [ttn]Repo git:(master) -> git commit
[master ec87aa9] Committed with master branch
 1 file changed, 1 insertion(+)
@ [ttn]Repo git:(master) ->
```

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

```
@ [ttn]Repo git:(practice) -> nano file1.txt
@ [ttn]Repo git:(practice) -> git add file1.txt
@ [ttn]Repo git:(practice) -> git commit
[practice 2ddc6a0] Committed in practice branch
 1 file changed, 1 insertion(+), 1 deletion(-)
@ [ttn]Repo git:(practice) -> git checkout master
Switched to branch 'master'
@ [ttn]Repo git:(master) -> nano file1.txt
@ [ttn]Repo git:(master) -> git add file1.txt
@ [ttn]Repo git:(master) -> git commit
[master ec87aa9] Committed with master branch
 1 file changed, 1 insertion(+)
@ [ttn]Repo git:(master) ->
```

16. Try merging and resolve merge conflicts

```
@ [ttn]Repo git:(master) -> git merge practice
Removing files.txt
Auto-merging file1.txt
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.
@ [ttn]Repo git:(master|MERGING) -> nano file1.txt
@ [ttn]Repo git:(master|MERGING) -> git add file1.txt
@ [ttn]Repo git:(master|MERGING) -> git commit
[master 62f42c8] Merge branch 'practice'
@ [ttn]Repo git:(master) ->
```

17. Stash the changes and pop them

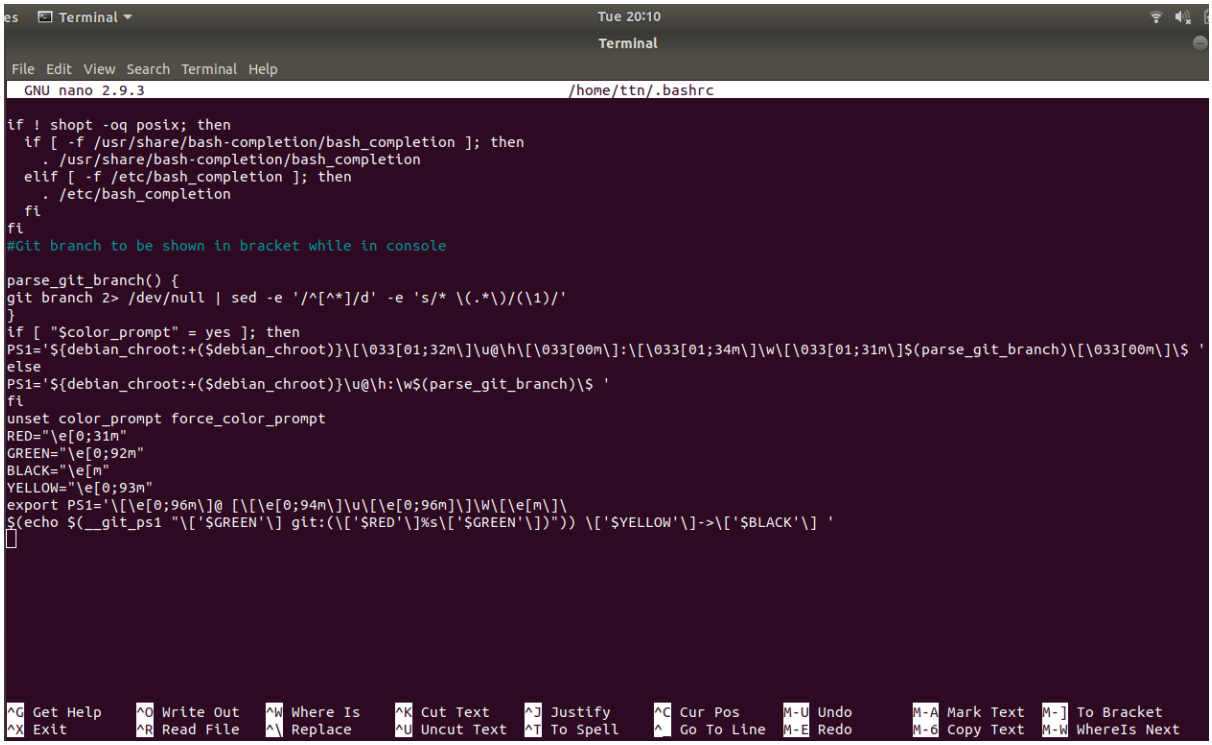
```
@ [ttn]Repo git:(master) -> nano stashfile.txt
@ [ttn]Repo git:(master) -> git add stashfile.txt
@ [ttn]Repo git:(master) -> git stash
Saved working directory and index state WIP on master: 62f42c8 Merge branch 'pra
ctice'
@ [ttn]Repo git:(master) -> git status
On branch master
nothing to commit, working tree clean
@ [ttn]Repo git:(master) -> git stash pop
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   stashfile.txt

Dropped refs/stash@{0} (b39920c790f550cce63c9d8e8581774440894dd3)
@ [ttn]Repo git:(master) ->
```

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
```


es Terminal
Tue 20:10
Terminal
File Edit View Search Terminal Help
GNU nano 2.9.3 /home/ttn/.bashrc
if ! shopt -oq posix; then
if [-f /usr/share/bash-completion/bash_completion]; then
. /usr/share/bash-completion/bash_completion
elif [-f /etc/bash_completion]; then
. /etc/bash_completion
fi
fi
#Git branch to be shown in bracket while in console
parse_git_branch() {
git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
}
if ["$color_prompt" = yes]; then
PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[01;31m\]$(parse_git_branch)\[\033[00m\]\$ '
else
PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w$(parse_git_branch)\$ '
fi
unset color_prompt force_color_prompt
RED="\e[0;31m"
GREEN="\e[0;92m"
BLACK="\e[m"
YELLOW="\e[0;93m"
export PS1='\[\e[0;96m\]@ [\[\e[0;94m\]\u\[\e[0;96m]\]\W\[\e[m\]\
$(echo $(__git_ps1 "\['$GREEN'\] git:(\['$RED'\]%s\['$GREEN'\])")) \['$YELLOW'\]->\['$BLACK'\] '
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos M-U Undo M-A Mark Text M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line M-E Redo M-6 Copy Text M-W WhereIs Next