**Assignment: 2 (Git Task)**

**Task 1**

* Create a script to generate a merge conflict.
* Using Shell Script

#!/bin/bash

mkdir gittest

cd gittest

git init

touch f1

cat > f1

git add f1

git commit -m "master commit"

git branch ninja

git checkout ninja

cat > f1

git add f1

git commit -m "ninja commit"

git checkout master

cat > f1

git add f1

git commit -m "master final commit"

git merge ninja

* Using Python Script

First created a repo and created some file into it

1. import git
2. from git import Repo
3. import os.path as osp
4. rw\_dir= osp.join(osp.expanduser('~'))
5. repo\_dir = osp.join(rw\_dir, 'repo1')
6. file\_name = osp.join(repo\_dir, 'f1')
7. r = git.Repo.init(repo\_dir)
8. fo = open(file\_name, "wb")
9. fo.write( "Python is a great language.")
10. fo.close()
11. r.index.add([file\_name])
12. r.index.commit("initial commit")
13. new\_branch = r.create\_head('ninja')
14. new\_branch.checkout()
15. fo = open(file\_name, "wb")
16. fo.write( "Python is not a great language.")
17. fo.close()
18. r.index.add([file\_name])
19. r.index.commit("ninja commit")
20. r.heads.master.commit = new\_branch.commit
21. r.heads.master.checkout()
22. fo = open(file\_name, "wb")
23. fo.write( "eko is great.");
24. fo.close()
25. r.index.add([file\_name])
26. r.index.commit("final commit commit")

Run merge script to get merge conflict error

from subprocess import Popen, PIPE

from os import path

git\_command = ['/usr/bin/git', 'merge', 'ninja']

repository = path.dirname('/home/hemant/gitApp/')

git\_query = Popen(git\_command, cwd=repository, stdout=PIPE, stderr=PIPE)

(git\_status, error) = git\_query.communicate()

print git\_status

print error

* Create muliple braches and push changes for each branch into remote repo.

mkdir gitrepo

cd gitrepo

git init

echo "add content here" >> README.md

git status

git add README.md

git commit -m "first commit"

git status

git remote add origin https://github.com/hemantkhokharvce/remoteRepo.git

git push -u origin master

git branch n1

git branch n2

git branch

git checkout n1

touch f1.txt

git add f1.txt

git commit -m "added f1.txt file"

git checkout n2

touch f2.txt

git add f2.txt

git commit -m "added f2.txt file"

git checkout master

git push -u origin n1

git push -u origin n2

* Create a script to clone remote repo and check out all existing remote branch.

#!/bin/bash

mkdir clone

cd clone

git init

git clone https://github.com/hemantkhokharvce/remoteRepo.git

git remote add origin https://github.com/hemantkhokharvce/remoteRepo.git

git ls-remote

* Clone a particular folder from a remote repo.

mkdir git-completion

cd git-completion

git init

git config core.sparsecheckout true

echo folder >> .git/info/sparse-checkout

git remote add -f origin <https://github.com/hemantkhokharvce/remoteRepo.git>

git pull origin master

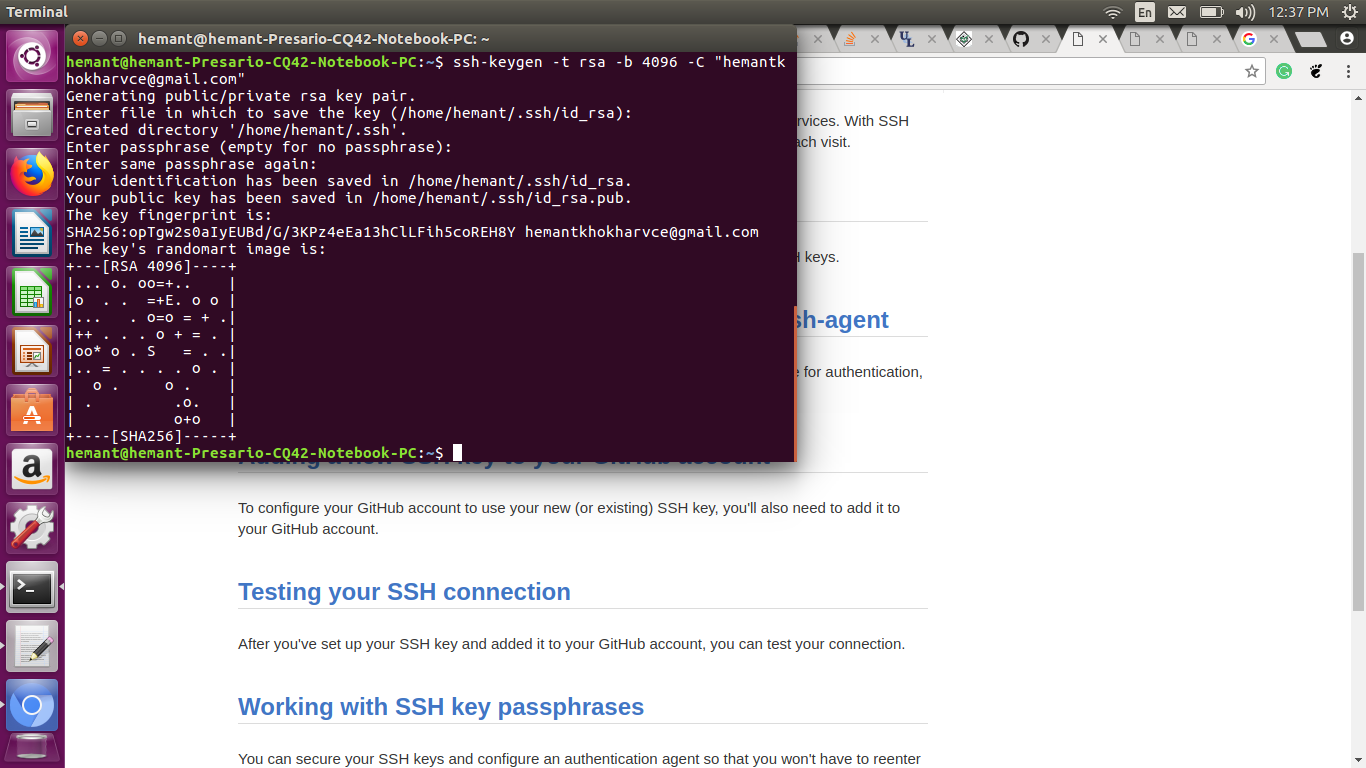
**Task 2**

* Use both https and ssh protocal to clone your remote repo.

1. Checked if any ssh key exist

ls -al ~/.ssh

1. Generate a new ssh key using



1. Copy this key and pasted into github -> settings -> ssh and gpg keys -> new ssh key -> paste your ssh key here
2. Clone using ssh

git clone [git@github.com:hemantkhokharvce/remoteRepo.git](mailto:git@github.com:hemantkhokharvce/remoteRepo.git)

* Change your working copy into a clonable remote repo

1. on remote machine firstly we create a directory with .git extention. Then initialize it with the command

git init --bare.

A bare repository created with git init --bare is for **sharing**.

1. **Command perform on remote machine side**  
   cd /var/local/git  
   mkdir myrepo.git  
   cd myrepo.git  
   git init --bare  
     
   **On local machine side (user side)**  
   git remote add origin ssh://user@host/var/local/git/myrepo.git  
   git push -u origin master

* Use file system protocal in both local and remote mode(clone from another machine using file protocal) to clone your repo.

**git clone ftp://var/local**

* Ignore backup, swp and pyc file from being commited.

1. mkdir .gitignore
2. $ git config –global core.excludefile ~/.gitignore
3. cd .gitignore
4. cat > .gitignore

\*.swp

\*.pyc

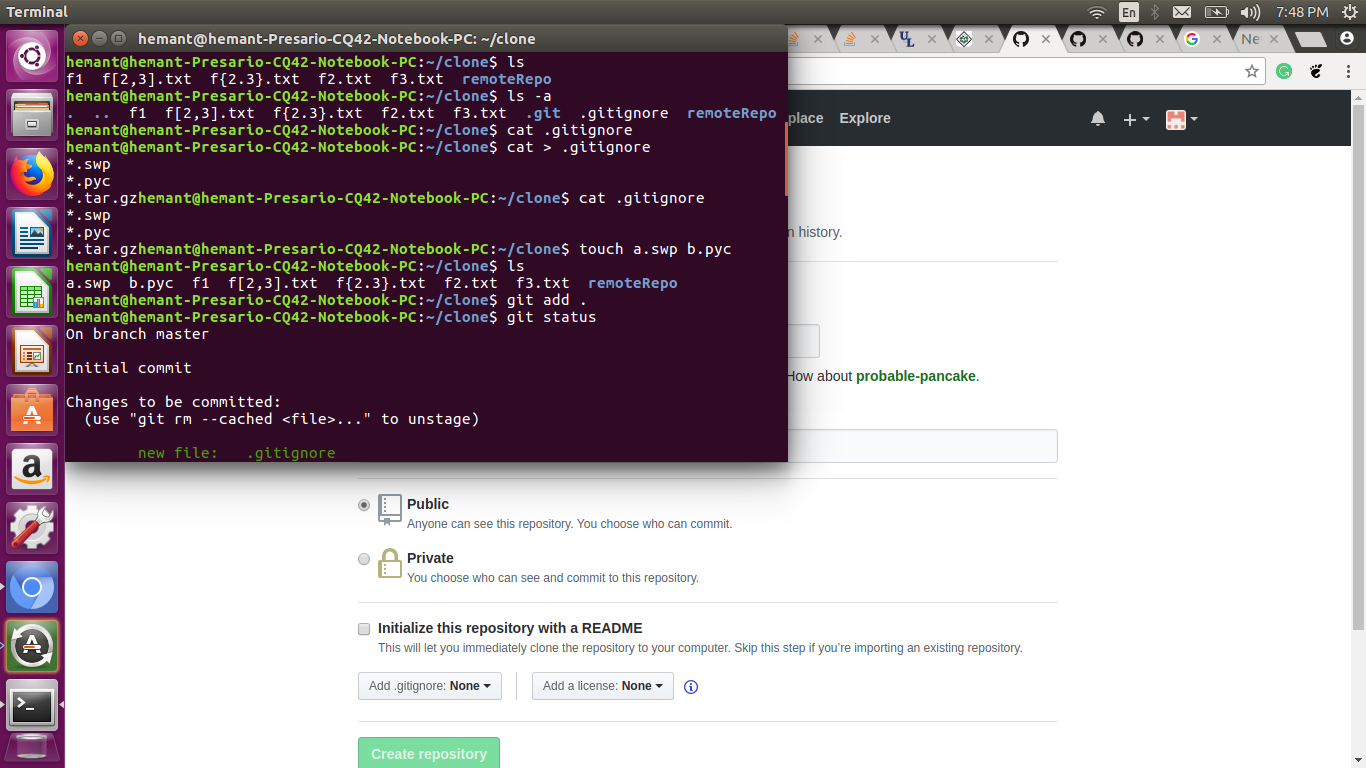
\*.tar.gz

5. touch a.swp b.pyc

6. git add .

7. git status

8. git commit -m “does not commit file with extensions .swp , .pyc, .tar.gz ”



**Task3**

* Add remote name **parent** for [ot-training/jenkins](https://github.com/ot-training/jenkins.git) to your own repo.

git remote add parent https://github.com/ot-training/jenkins.git