Name: Yuri P. Nollan	**Date Performed: 09/08/2023**
Course/Section: CPE31S6	**Date Submitted: 09/08/2023**
Instructor: Dr. Jonathan Taylar	**Semester and SY: 1st SEM AY 2023-2024**

Activity 2: SSH Key-Based Authentication and Setting up Git

1. **Objectives:**

1.1 Configure remote and local machine to connect via SSH using a KEY instead of using a password

1.2 Create a public key and private key

1.3 Verify connectivity

1.4 Setup Git Repository using local and remote repositories

1.5 Configure and Run ad hoc commands from local machine to remote servers

Part 1: Discussion

It is assumed that you are already done with the last Activity (**Activity 1: Configure Network using Virtual Machines).** *Provide screenshots for each task*.

It is also assumed that you have VMs running that you can SSH but requires a password. Our goal is to remotely login through SSH using a key without using a password. In this activity, we create a public and a private key. The private key resides in the local machine while the public key will be pushed to remote machines. Thus, instead of using a password, the local machine can connect automatically using SSH through an authorized key.

What Is ssh-keygen?

Ssh-keygen is a tool for creating new authentication key pairs for SSH. Such key pairs are used for automating logins, single sign-on, and for authenticating hosts.

SSH Keys and Public Key Authentication

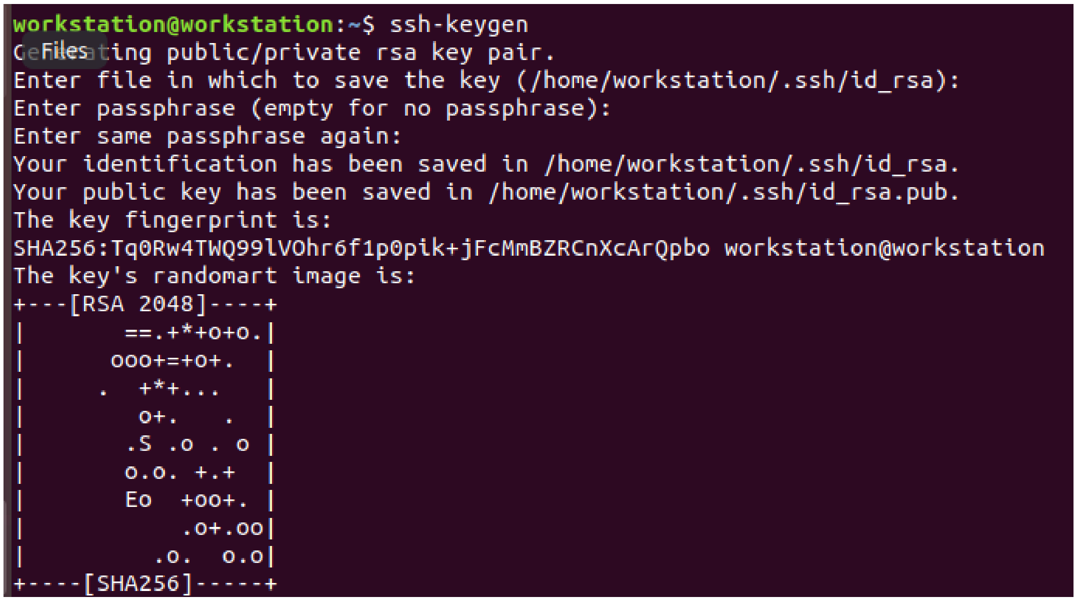
The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.

SSH introduced public key authentication as a more secure alternative to the older .rhosts authentication. It improved security by avoiding the need to have password stored in files and eliminated the possibility of a compromised server stealing the user's password.

However, SSH keys are authentication credentials just like passwords. Thus, they must be managed somewhat analogously to usernames and passwords. They should have a proper termination process so that keys are removed when no longer needed.

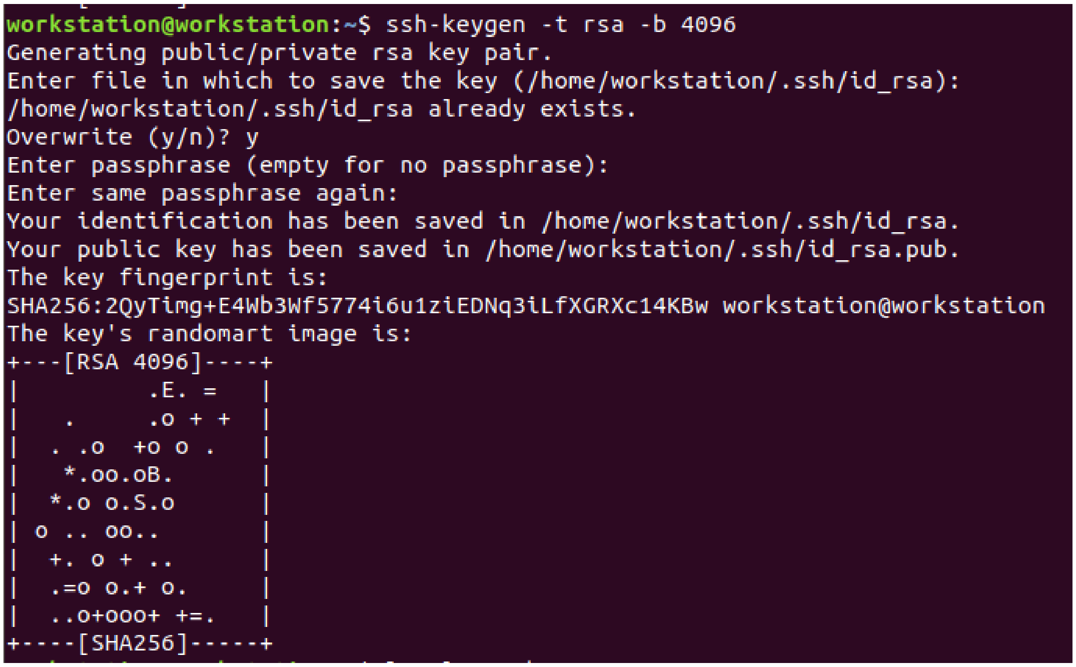
Task 1: Create an SSH Key Pair for User Authentication

1. The simplest way to generate a key pair is to run *ssh-keygen* without arguments. In this case, it will prompt for the file in which to store keys. First, the tool asked where to save the file. SSH keys for user authentication are usually stored in the users .ssh directory under the home directory. However, in enterprise environments, the location is often different. The default key file name depends on the algorithm, in this case *id_rsa* when using the default RSA algorithm. It could also be, for example, *id_dsa* or *id_ecdsa*.



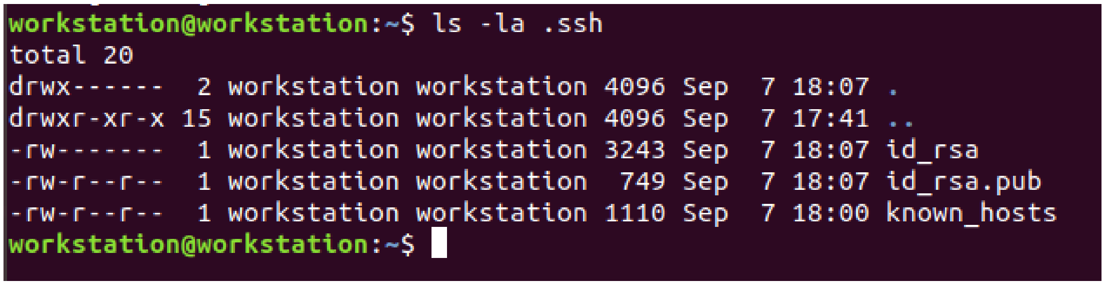
```
workstation@workstation:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/workstation/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/workstation/.ssh/id_rsa.
Your public key has been saved in /home/workstation/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Tq0Rw4TWQ99lVOhr6f1p0pik+jFcMmBZRCnXcArQpbo workstation@workstation
The key's randomart image is:
+---[RSA 2048]----+
|       ==.+*+o+o.|
|      ooo+=+o+.  |
|     .  +*+...   |
|        o+.   .  |
|       .S .o . o |
|       o.o. +.+  |
|       Eo  +oo+. |
|           .o+.oo|
|         .o.  o.o|
+----[SHA256]-----+
```

2. Issue the command *ssh-keygen -t rsa -b 4096.* The algorithm is selected using the -t option and key size using the -b option.



```
workstation@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/workstation/.ssh/id_rsa):
/home/workstation/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/workstation/.ssh/id_rsa.
Your public key has been saved in /home/workstation/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:2QyTimg+E4Wb3Wf5774i6u1ziEDNq3iLfXGRXc14KBw workstation@workstation
The key's randomart image is:
+---[RSA 4096]----+
|         .E. =   |
|   .     .o + +  |
|  . .o  +o o .   |
|   *.oo.oB.      |
|  *.o o.S.o      |
| o .. oo..       |
|  +. o + ..      |
|  .=o o.+ o.     |
|  ..o+ooo+ +=.   |
+----[SHA256]-----+
```

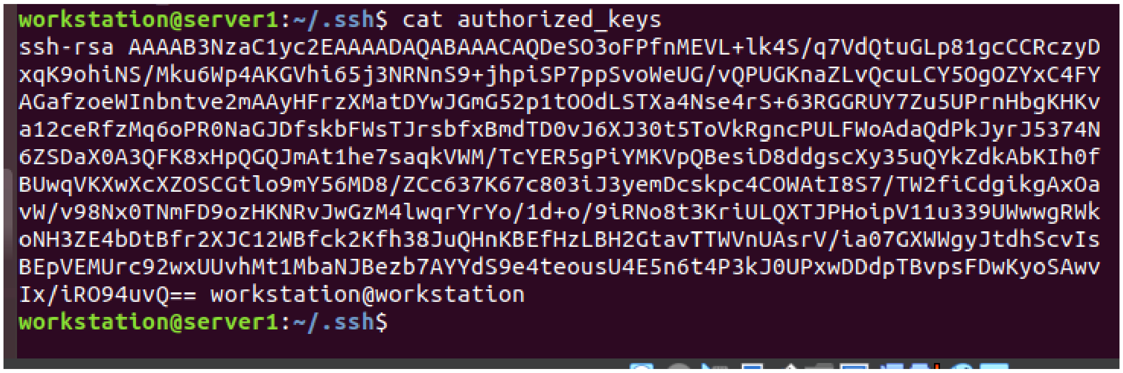
3. When asked for a passphrase, just press enter. The passphrase is used for encrypting the key, so that it cannot be used even if someone obtains the private key file. The passphrase should be cryptographically strong.
4. Verify that you have created the key by issuing the command *ls -la .ssh.* The command should show the .ssh directory containing a pair of keys. For example, id_rsa.pub and id_rsa.



```
workstation@workstation:~$ ls -la .ssh
total 20
drwx------  2 workstation workstation 4096 Sep  7 18:07 .
drwxr-xr-x 15 workstation workstation 4096 Sep  7 17:41 ..
-rw-------  1 workstation workstation 3243 Sep  7 18:07 id_rsa
-rw-r--r--  1 workstation workstation  749 Sep  7 18:07 id_rsa.pub
-rw-r--r--  1 workstation workstation 1110 Sep  7 18:00 known_hosts
workstation@workstation:~$ 
```

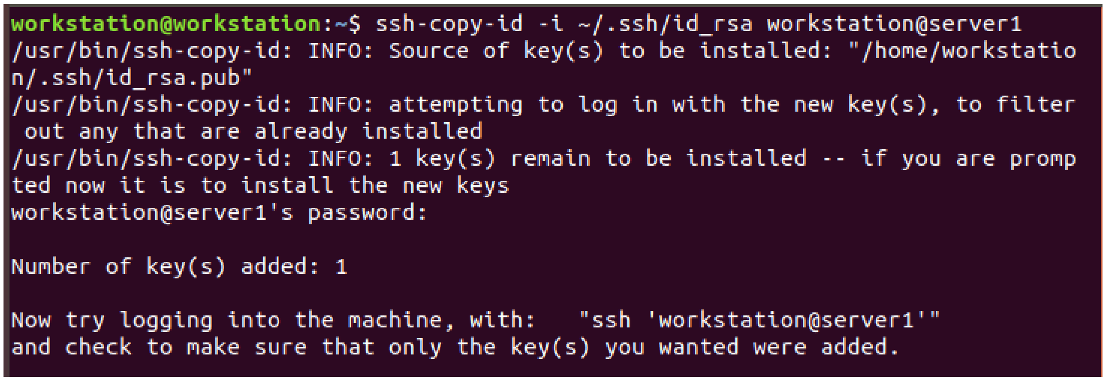
Task 2: Copying the Public Key to the remote servers

1. To use public key authentication, the public key must be copied to a server and installed in an *authorized_keys* file. This can be conveniently done using the *ssh-copy-id* tool.



```
workstation@server1:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDeSO3oFPfnMEVL+lk4S/q7VdQtuGLp81gcCCRczyD
xqK9ohiNS/Mku6Wp4AKGVhi65j3NRNnS9+jhpiSP7ppSvoWeUG/vQPUGKnaZLvQcuLCY5OgOZYxC4FY
AGafzoeWInbntve2mAAyHFrzXMatDYwJGmG52p1tOOdLSTXa4Nse4rS+63RGGRUY7Zu5UPrnHbgKHKv
a12ceRfzMq6oPR0NaGJDfskbFWsTJrsbfxBmdTD0vJ6XJ30t5ToVkRgncPULFWoAdaQdPkJyrJ5374N
6ZSDaX0A3QFK8xHpQGQJmAt1he7saqkVWM/TcYER5gPiYMKVpQBesiD8ddgscXy35uQYkZdkAbKIh0f
BUwqVKXwXcXZOSCGtlo9mY56MD8/ZCc637K67c803iJ3yemDcskpc4COWAtI8S7/TW2fiCdgikgAxOa
vW/v98Nx0TNmFD9ozHKNRvJwGzM4lwqrYrYo/1d+o/9iRNo8t3KriULQXTJPHoipV11u339UWwwgRWk
oNH3ZE4bDtBfr2XJC12WBfck2Kfh38JuQHnKBEfHzLBH2GtavTTWVnUAsrV/ia07GXWWgyJtdhScvIs
BEpVEMUrc92wxUUvhMt1MbaNJBezb7AYYdS9e4teousU4E5n6t4P3kJ0UPxwDDdpTBvpsFDwKyoSAwv
Ix/iRO94uvQ== workstation@workstation
workstation@server1:~/.ssh$
```

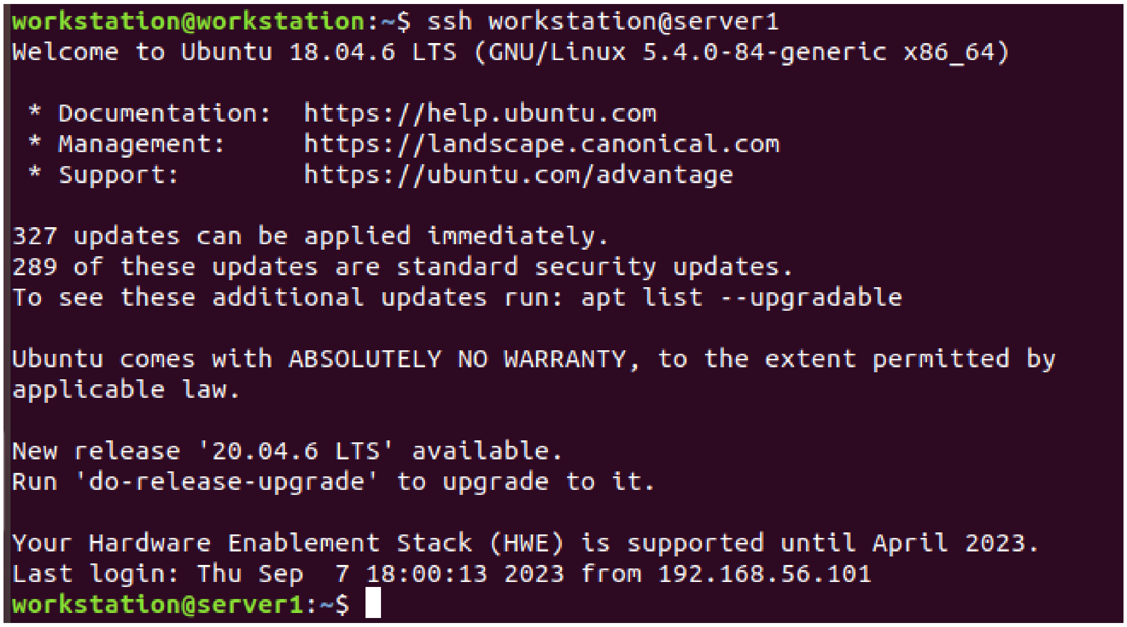
2. Issue the command similar to this: *ssh-copy-id -i ~/.ssh/id_rsa user@host*



```
workstation@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa workstation@server1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/workstatio
n/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
 out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are promp
ted now it is to install the new keys
workstation@server1's password: 

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'workstation@server1'"
and check to make sure that only the key(s) you wanted were added.
```

3. Once the public key has been configured on the server, the server will allow any connecting user that has the private key to log in. During the login process, the client proves possession of the private key by digitally signing the key exchange.

4. On the local machine, verify that you can SSH with Server 1 and Server 2. What did you notice? Did the connection ask for a password? If not, why?



```
workstation@workstation:~$ ssh workstation@server1
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-84-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

327 updates can be applied immediately.
289 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Thu Sep  7 18:00:13 2023 from 192.168.56.101
workstation@server1:~$ 
```

Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?
   - For me, ssh-program is like a secure program and a useful tool that is used to connect and control other computers to transfer files, and access different kinds of locked files or services on the internet.
2. How do you know that you already installed the public key to the remote servers?
   - You will know that you already installed the public key to the remote server if the functions you called was reflected to the remote server.

Part 2: Discussion

Provide screenshots for each task.

It is assumed that you are done with the last activity (**Activity 2: SSH Key-Based Authentication**).

Set up Git

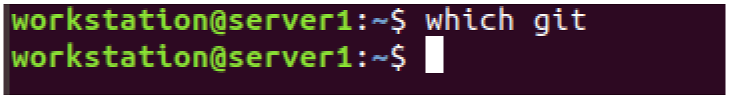
At the heart of GitHub is an open-source version control system (VCS) called Git. Git is responsible for everything GitHub-related that happens locally on your computer. To use Git on the command line, you'll need to download, install, and configure Git on your computer. You can also install GitHub CLI to use GitHub from the command line.

If you don't need to work with files locally, GitHub lets you complete many Git-related actions directly in the browser, including:

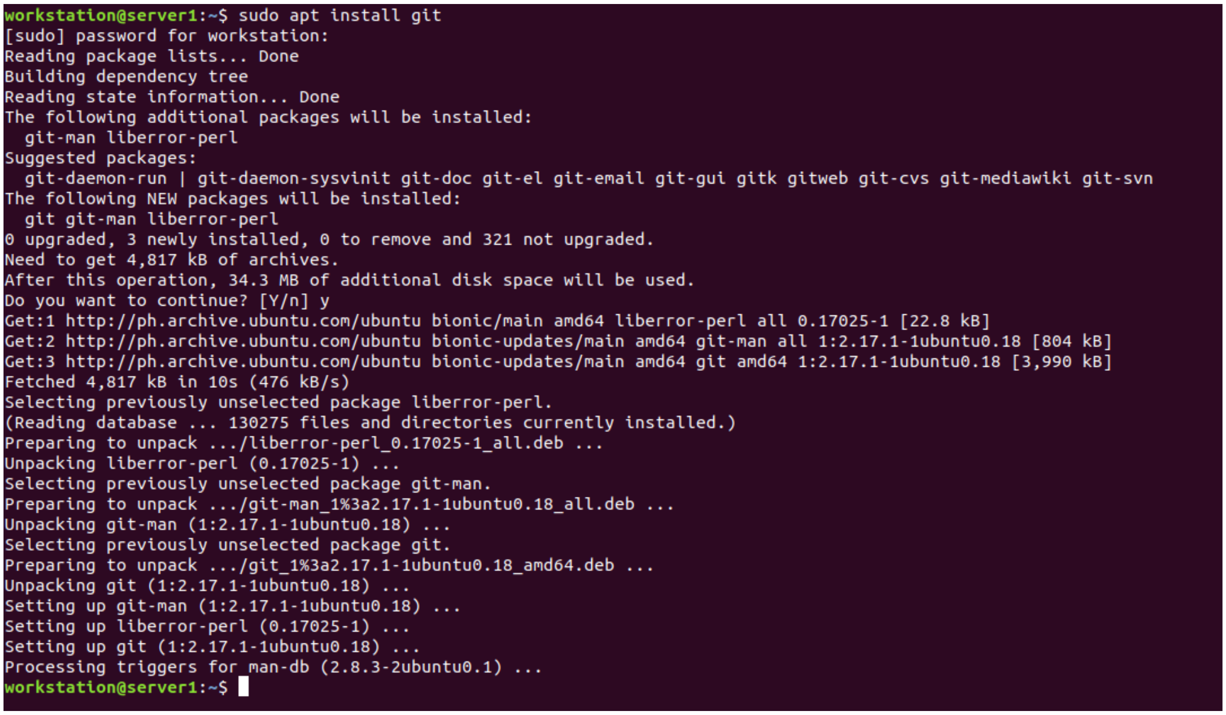
- Creating a repository
- Forking a repository
- Managing files
- Being social

Task 3: Set up the Git Repository

1. On the local machine, verify the version of your git using the command *which git.* If a directory of git is displayed, then you don’t need to install git. Otherwise, to install git, use the following command: *sudo apt install git*

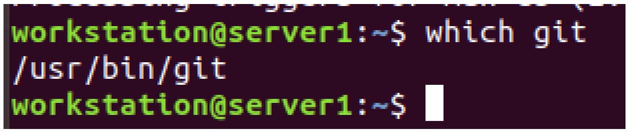


```
workstation@server1:~$ which git
workstation@server1:~$ 
```



```
workstation@server1:~$ sudo apt install git
[sudo] password for workstation:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 321 not upgraded.
Need to get 4,817 kB of archives.
After this operation, 34.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu bionic/main amd64 liberror-perl all 0.17025-1 [22.8 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu bionic-updates/main amd64 git-man all 1:2.17.1-1ubuntu0.18 [804 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu bionic-updates/main amd64 git amd64 1:2.17.1-1ubuntu0.18 [3,990 kB]
Fetched 4,817 kB in 10s (476 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 130275 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17025-1_all.deb ...
Unpacking liberror-perl (0.17025-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.17.1-1ubuntu0.18_all.deb ...
Unpacking git-man (1:2.17.1-1ubuntu0.18) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.17.1-1ubuntu0.18_amd64.deb ...
Unpacking git (1:2.17.1-1ubuntu0.18) ...
Setting up git-man (1:2.17.1-1ubuntu0.18) ...
Setting up liberror-perl (0.17025-1) ...
Setting up git (1:2.17.1-1ubuntu0.18) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
workstation@server1:~$ 
```

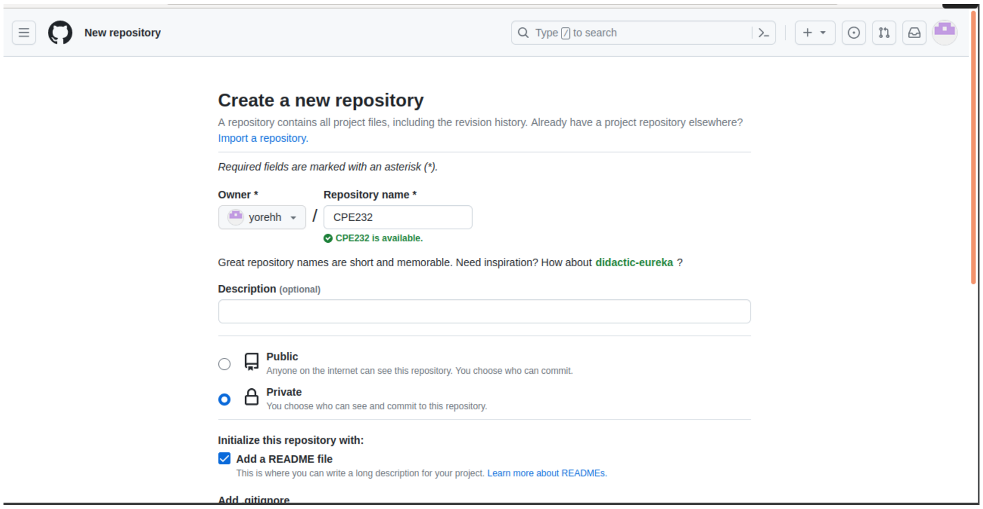
2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/bin/git.*



```
workstation@server1:~$ which git
/usr/bin/git
workstation@server1:~$ 
```

3. The version of git installed in your device is the latest. Try issuing the command *git --version* to know the version installed.
4. Using the browser in the local machine, go to www.github.com.
5. Sign up in case you don’t have an account yet. Otherwise, login to your GitHub account.

a. Create a new repository and name it as CPE232_yourname. Check Add a README file and click Create repository.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * / Repository name *

yorehh / CPE232

CPE232 is available.

Great repository names are short and memorable. Need inspiration? How about **didactic-eureka** ?

Description (optional)

○ **Public**
Anyone on the internet can see this repository. You choose who can commit.

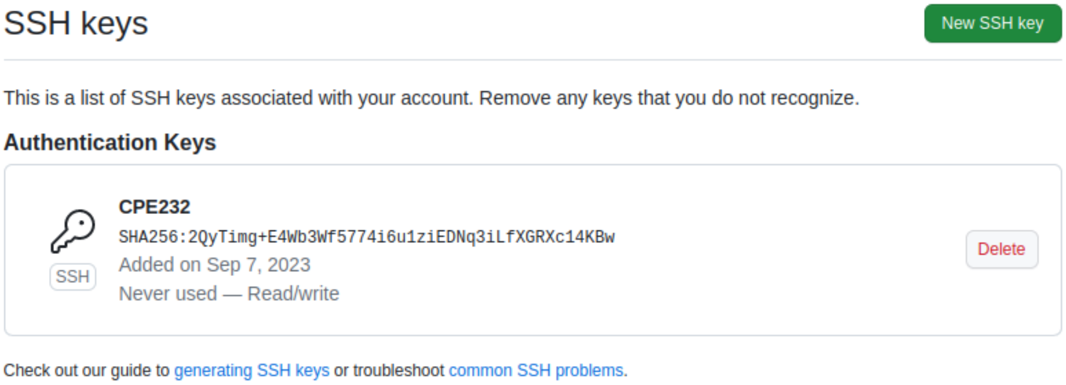
● **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☑ **Add a README file**
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

b. Create a new SSH key on GitHub. Go your profile's setting and click SSH and GPG keys. If there is an existing key, make sure to delete it. To create a new SSH keys, click New SSH Key. Write CPE232 key as the title of the key.



SSH keys

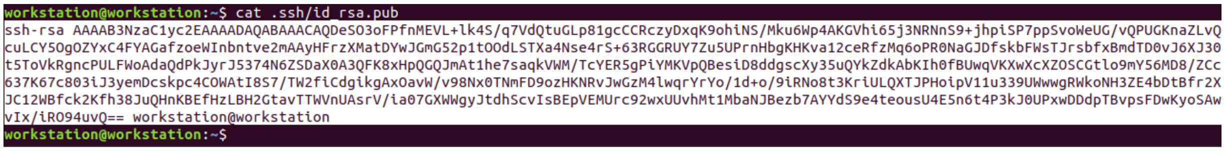
This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

CPE232
SHA256:2QyTimg+E4Wb3Wf5774i6u1ziEDNq3iLfXGRXc14KBw
Added on Sep 7, 2023
Never used — Read/write

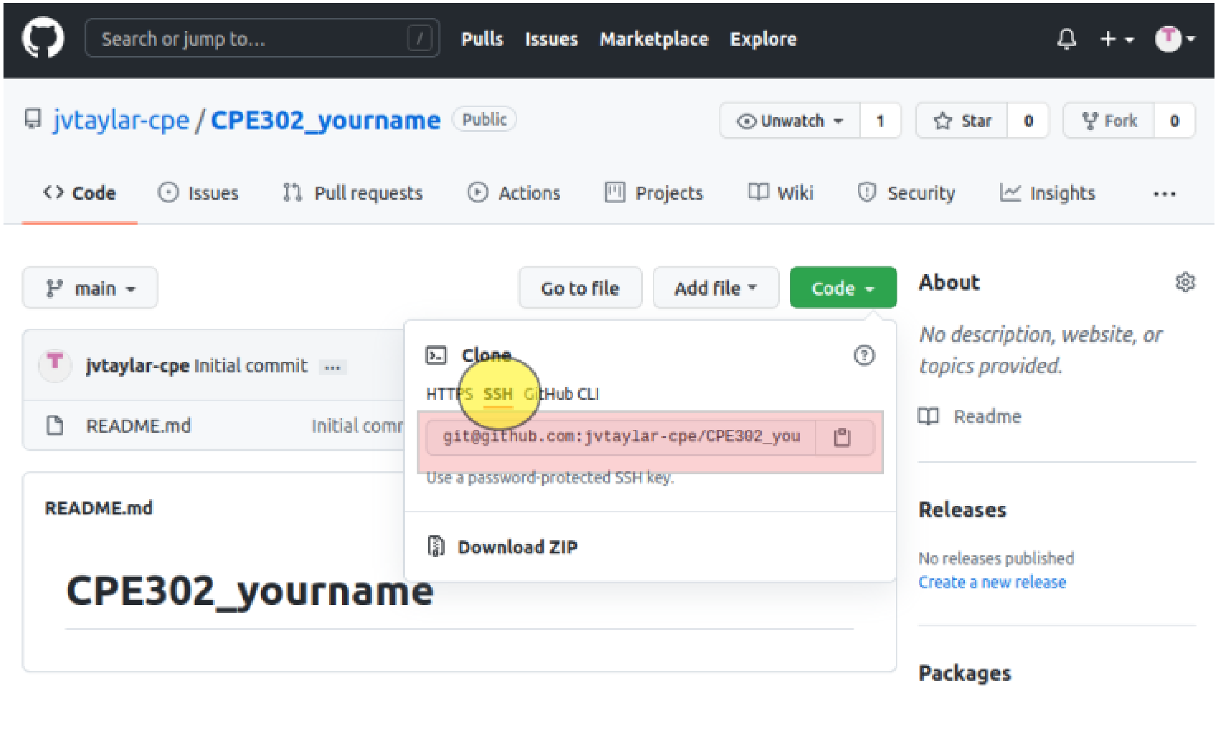
Check out our guide to generating SSH keys or troubleshoot common SSH problems.

c. On the local machine's terminal, issue the command cat .ssh/id_rsa.pub and copy the public key. Paste it on the GitHub key and press Add SSH key.

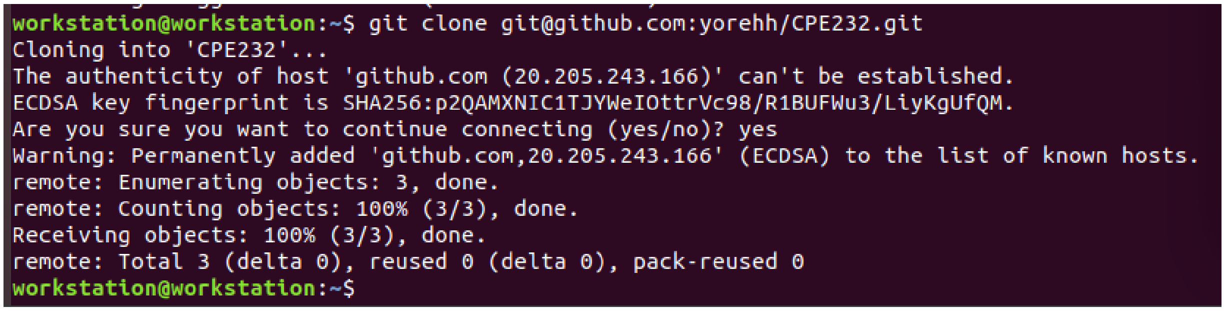


```
workstation@workstation:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDeSO3oFPfnMEVL+lk4S/q7VdQtuGLp81gcCCRczyDxqK9ohiNS/Mku6Wp4AKGVhi65j3NRNnS9+jhpiSP7ppSvoWeUG/vQPUGKnaZLvQ
cuLCY5OgOZYxC4FYAGafzoeWInbntve2mAAyHFrzXMatDYwJGmG52p1tOOdLSTXa4Nse4rS+63RGGRUY7Zu5UPrnHbgKHKva12ceRfzMq6oPR0NaGJDfskbFWsTJrsbfxBmdTD0vJ6XJ30
t5ToVkRgncPULFWoAdaQdPkJyrJ5374N6ZSDaX0A3QFK8xHpQGQJmAt1he7saqkVWM/TcYER5gPiYMKVpQBesiD8ddgscXy35uQYkZdkAbKIh0fBUwqVKXwXcXZOSCGtlo9mY56MD8/ZCc
637K67c803iJ3yemDcskpc4COWAtI8S7/TW2fiCdgikgAxOavW/v98Nx0TNmFD9ozHKNRvJwGzM4lwqrYrYo/1d+o/9iRNo8t3KriULQXTJPHoipV11u339UWwwgRWkoNH3ZE4bDtBfr2X
JC12WBfck2Kfh38JuQHnKBEfHzLBH2GtavTTWVnUAsrV/ia07GXWWgyJtdhScvIsBEpVEMUrc92wxUUvhMt1MbaNJBezb7AYYdS9e4teousU4E5n6t4P3kJ0UPxwDDdpTBvpsFDwKyoSAw
vIx/iRO94uvQ== workstation@workstation
workstation@workstation:~$
```

d. Clone the repository that you created. In doing this, you need to get the link from GitHub. Browse to your repository as shown below. Click on the Code drop down menu. Select SSH and copy the link.

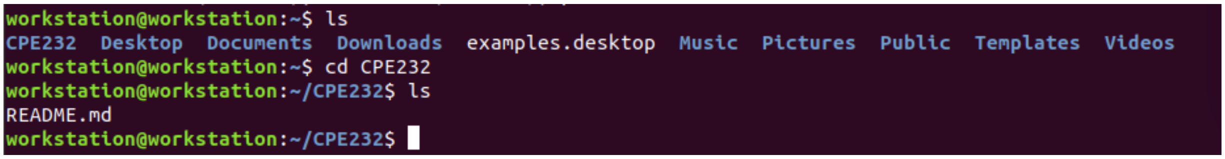


e. Issue the command git clone followed by the copied link. For example, *git clone git@github.com:jvtaylar-cpe/CPE232_yourname.git*. When prompted to continue connecting, type yes and press enter.



```
workstation@workstation:~$ git clone git@github.com:yorehh/CPE232.git
Cloning into 'CPE232'...
The authenticity of host 'github.com (20.205.243.166)' can't be established.
ECDSA key fingerprint is SHA256:p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'github.com,20.205.243.166' (ECDSA) to the list of known hosts.
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
Receiving objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
workstation@workstation:~$
```

f. To verify that you have cloned the GitHub repository, issue the command *ls*. Observe that you have the CPE232_yourname in the list of your directories. Use CD command to go to that directory and LS command to see the file README.md.

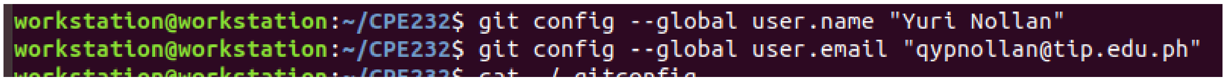


```
workstation@workstation:~$ ls
CPE232  Desktop  Documents  Downloads  examples.desktop  Music  Pictures  Public  Templates  Videos
workstation@workstation:~$ cd CPE232
workstation@workstation:~/CPE232$ ls
README.md
workstation@workstation:~/CPE232$ 
```

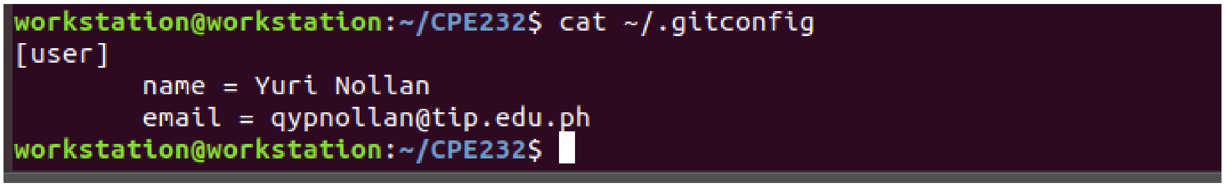
g. Use the following commands to personalize your git.

- *git config --global user.name “Your Name”*
- *git config --global user.email yourname@email.com*

- Verify that you have personalized the config file using the command *cat ~/.gitconfig*



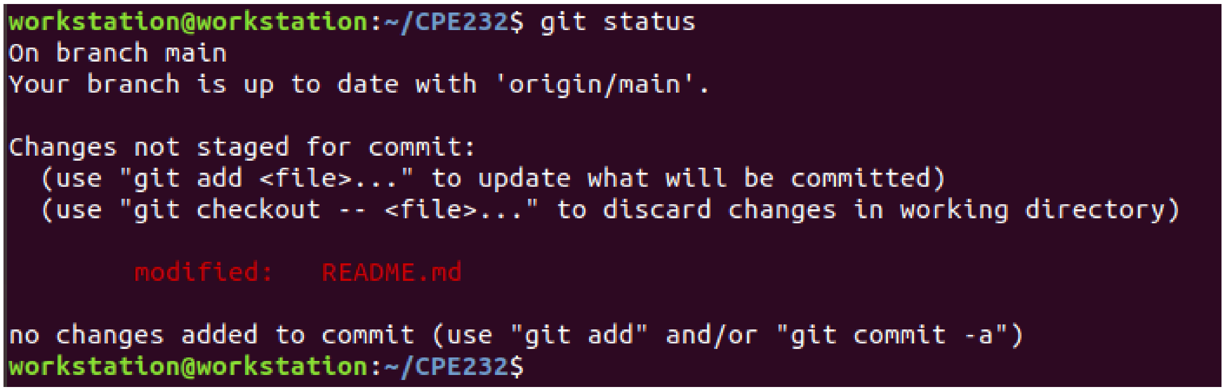
```
workstation@workstation:~/CPE232$ git config --global user.name "Yuri Nollan"
workstation@workstation:~/CPE232$ git config --global user.email "qypnollan@tip.edu.ph"
workstation@workstation:~/CPE232$ cat ~/.gitconfig
```



```
workstation@workstation:~/CPE232$ cat ~/.gitconfig
[user]
        name = Yuri Nollan
        email = qypnollan@tip.edu.ph
workstation@workstation:~/CPE232$ 
```

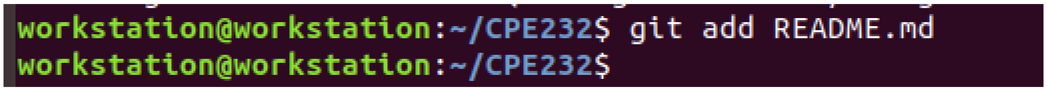
h. Edit the README.md file using nano command. Provide any information on the markdown file pertaining to the repository you created. Make sure to write out or save the file and exit.

i. Use the *git status* command to display the state of the working directory and the staging area. This command shows which changes have been staged, which haven't, and which files aren't being tracked by Git. Status output does not show any information regarding the committed project history. What is the result of issuing this command?



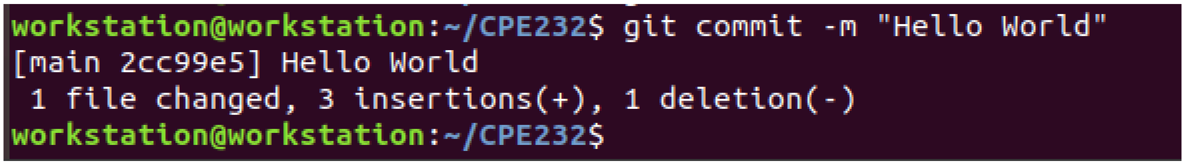
```
workstation@workstation:~/CPE232$ git status
On branch main
Your branch is up to date with 'origin/main'.

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:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
workstation@workstation:~/CPE232$
```

j. Use the command *git add README.md* to add the file into the staging area.



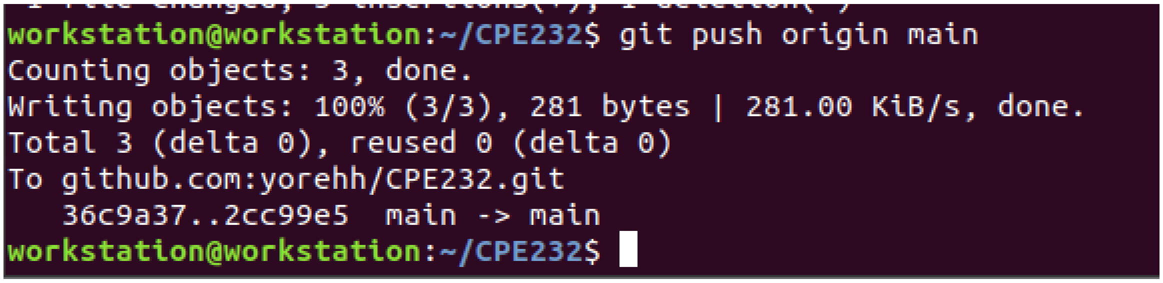
```
workstation@workstation:~/CPE232$ git add README.md
workstation@workstation:~/CPE232$
```

k. Use the *git commit -m "your message"* to create a snapshot of the staged changes along the timeline of the Git projects history. The use of this command is required to select the changes that will be staged for the next commit.



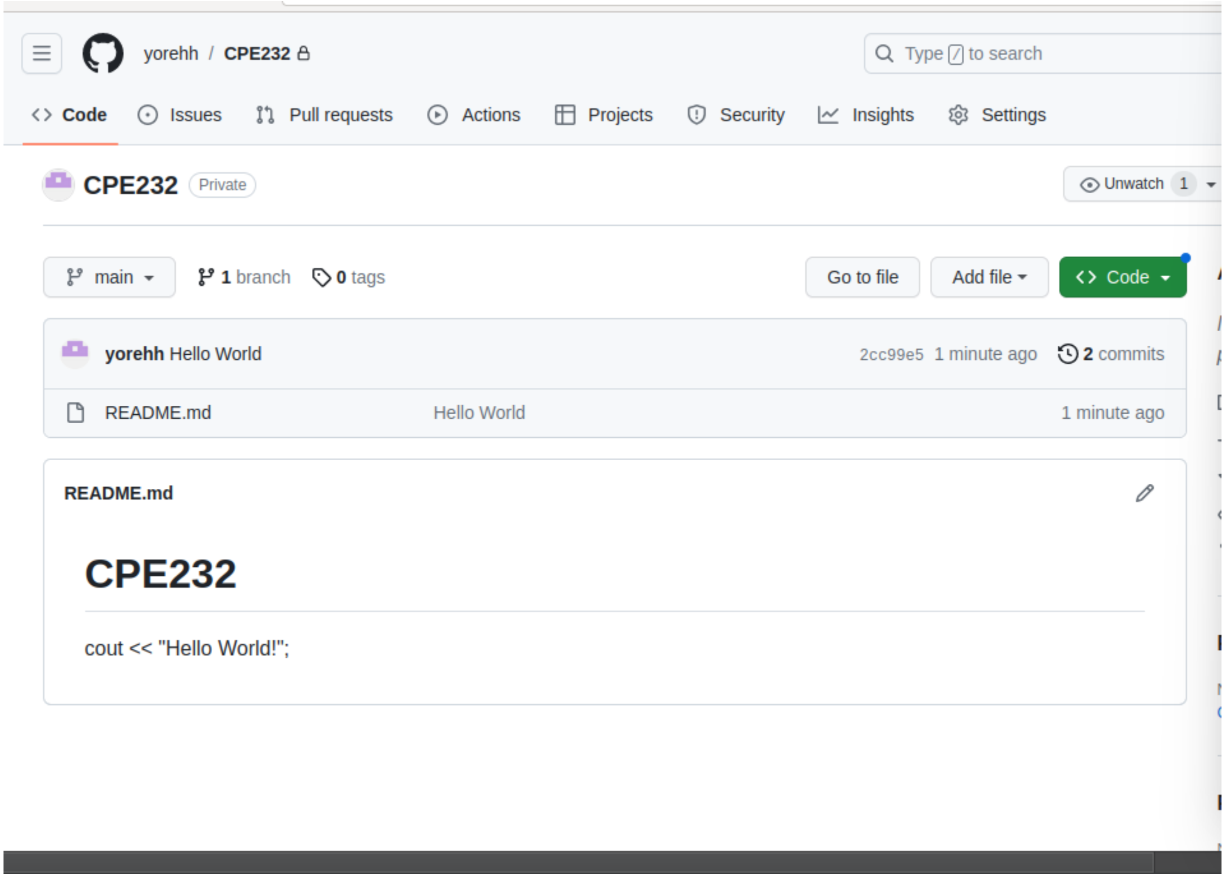
```
workstation@workstation:~/CPE232$ git commit -m "Hello World"
[main 2cc99e5] Hello World
 1 file changed, 3 insertions(+), 1 deletion(-)
workstation@workstation:~/CPE232$
```

l. Use the command *git push <remote><branch>* to upload the local repository content to GitHub repository. Pushing means to transfer commits from the local repository to the remote repository. As an example, you may issue *git push origin main.*



```
 1 file changed, 3 insertions(+), 1 deletion(-)
workstation@workstation:~/CPE232$ git push origin main
Counting objects: 3, done.
Writing objects: 100% (3/3), 281 bytes | 281.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:yorehh/CPE232.git
   36c9a37..2cc99e5  main -> main
workstation@workstation:~/CPE232$ 
```

m. On the GitHub repository, verify that the changes have been made to README.md by refreshing the page. Describe the README.md file. You can notice the how long was the last commit. It should be some minutes ago and the message you typed on the git commit command should be there. Also, the README.md file should have been edited according to the text you wrote.



yorehh Hello World 2cc99e5 1 minute ago 2 commits

README.md Hello World 1 minute ago

README.md

CPE232

cout << "Hello World!";

Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
    - So far, the sort of things that we have don to the remote servers using ansible commands are adding a readme.md file, and trying to connect the remote servers or double - checking if the remote server are working by inputting a message in the remote server.
4. How important is the inventory file?
    - Inventory file is very important to store all of the assets and resources within a certain circle or server.

Conclusions/Learnings:

In this activity, I have learned how to configure remote and local machine to connect via SSH using a KEY instead of using a password by finding the key of the local machine. I was also able to create a public key and private key. I was also able to verify connectivity from the remote and local machine. I was also able to set up a Git Repository using local and remote repositories from the local machine. and lastly, I was able to configure and run commands from the local machine to the remote servers. Overall, this activity has helped me enhance my knowledge in SSH key and the connectivity from a local machine to other remote servers.