Name: Andreu John L. Salvador	**Date Performed: Aug 26, 2023**
Course/Section: CPE232-CPE31S5	**Date Submitted: Aug 28, 2023**
Instructor: Engr. Roman Richard	**Semester and SY: 1st sem/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.

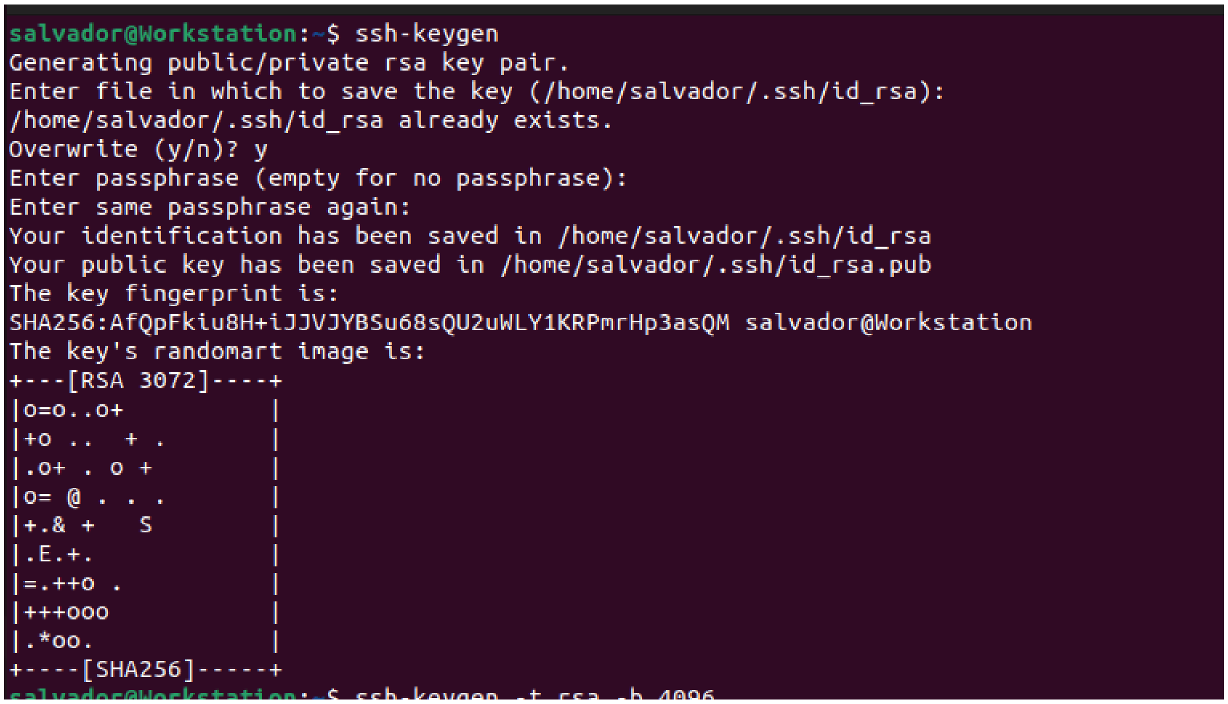
SSH introduced public key authentication as a more secure alternative to the older .rhosts authentication. It improved security by avoiding the need to have password stored in files and eliminated the possibility of a compromised server stealing the user's password.

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

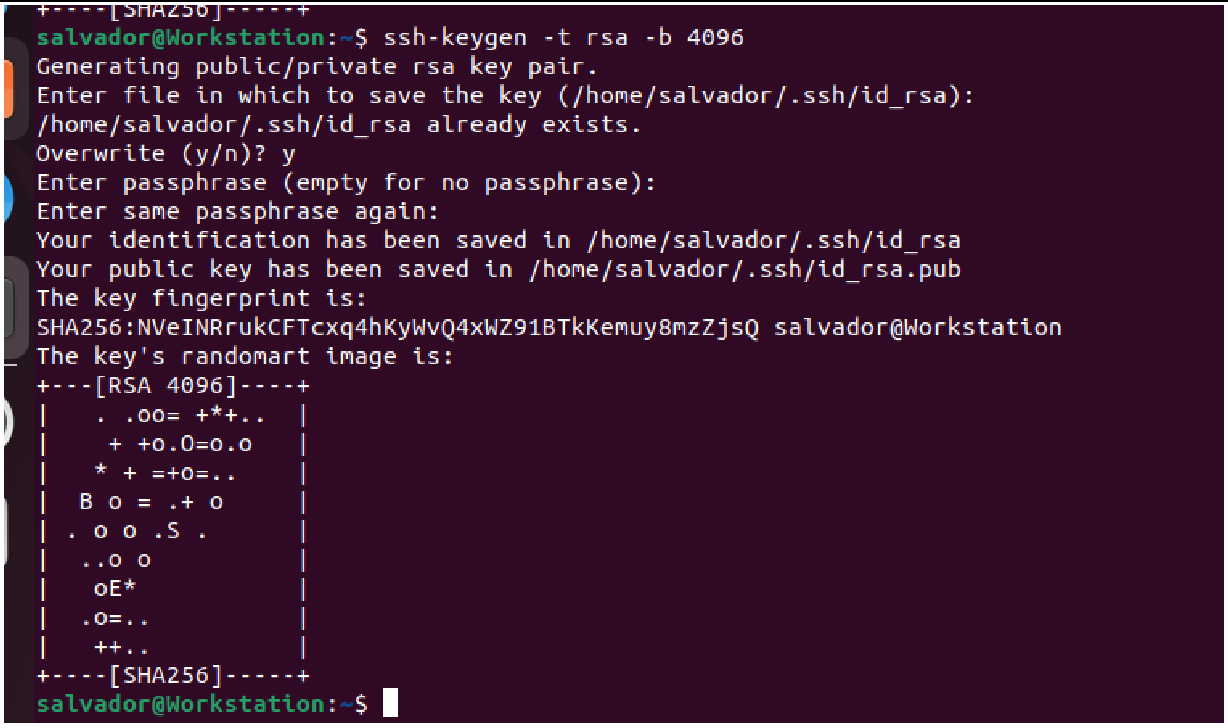
1. The simplest way to generate a key pair is to run *ssh-keygen* without arguments. In this case, it will prompt for the file in which to store keys. First,

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*.



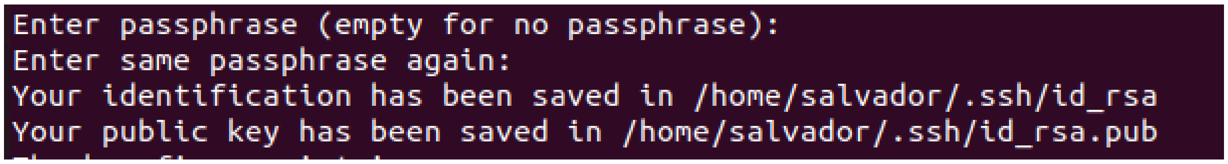
```
salvador@Workstation:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/salvador/.ssh/id_rsa):
/home/salvador/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/salvador/.ssh/id_rsa
Your public key has been saved in /home/salvador/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:AfQpFkiu8H+iJJVJYBSu68sQU2uWLY1KRPmrHp3asQM salvador@Workstation
The key's randomart image is:
+---[RSA 3072]----+
|o=o..o+          |
|+o ..  + .       |
|.o+ . o +        |
|o= @ . . .       |
|+.& +   S        |
|.E.+.            |
|=.++o .          |
|+++ooo           |
|.*oo.            |
+----[SHA256]-----+
salvador@Workstation:~$ ssh-keygen -t rsa -b 4096
```

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.



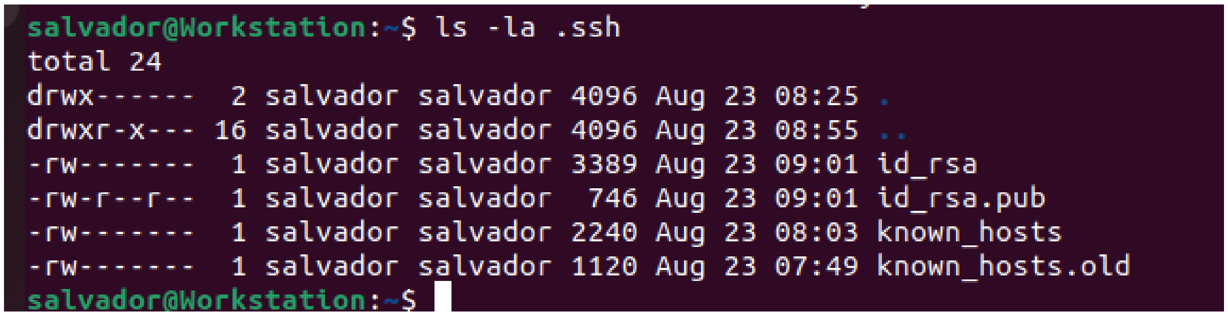
```
+----[SHA256]-----+
salvador@Workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/salvador/.ssh/id_rsa):
/home/salvador/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/salvador/.ssh/id_rsa
Your public key has been saved in /home/salvador/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:NVeINRrukCFTcxq4hKyWvQ4xWZ91BTkKemuy8mzZjsQ salvador@Workstation
The key's randomart image is:
+---[RSA 4096]----+
|   . .oo= +*+..  |
|    + +o.O=o.o   |
|   * + =+o=..    |
|  B o = .+ o     |
| . o o .S .      |
|  ..o o          |
|   oE*           |
|  .o=..          |
|   ++..          |
+----[SHA256]-----+
salvador@Workstation:~$
```

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.



```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/salvador/.ssh/id_rsa
Your public key has been saved in /home/salvador/.ssh/id_rsa.pub
```

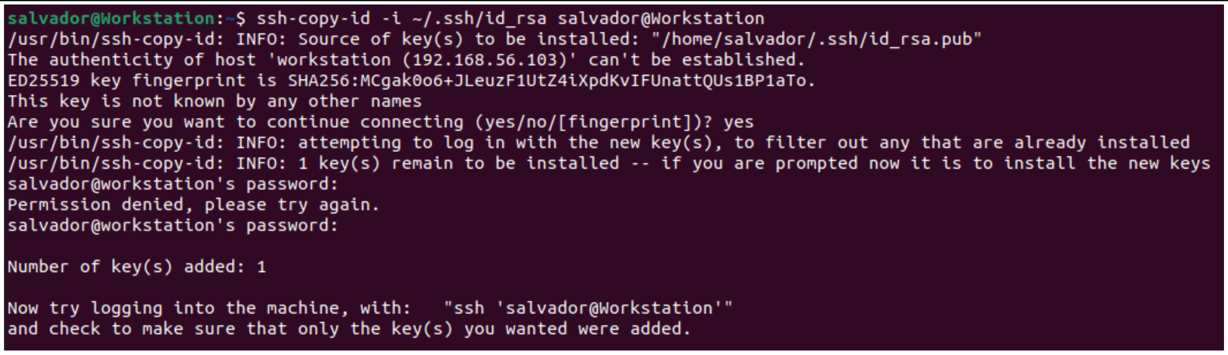
4. Verify that you have created the key by issuing the command *ls -la .ssh.* The command should show the .ssh directory containing a pair of keys. For example, id_rsa.pub and id_rsa. .



```
salvador@Workstation:~$ ls -la .ssh
total 24
drwx------  2 salvador salvador 4096 Aug 23 08:25 .
drwxr-x--- 16 salvador salvador 4096 Aug 23 08:55 ..
-rw-------  1 salvador salvador 3389 Aug 23 09:01 id_rsa
-rw-r--r--  1 salvador salvador  746 Aug 23 09:01 id_rsa.pub
-rw-------  1 salvador salvador 2240 Aug 23 08:03 known_hosts
-rw-------  1 salvador salvador 1120 Aug 23 07:49 known_hosts.old
salvador@Workstation:~$
```

Task 2: Copying the Public Key to the remote servers

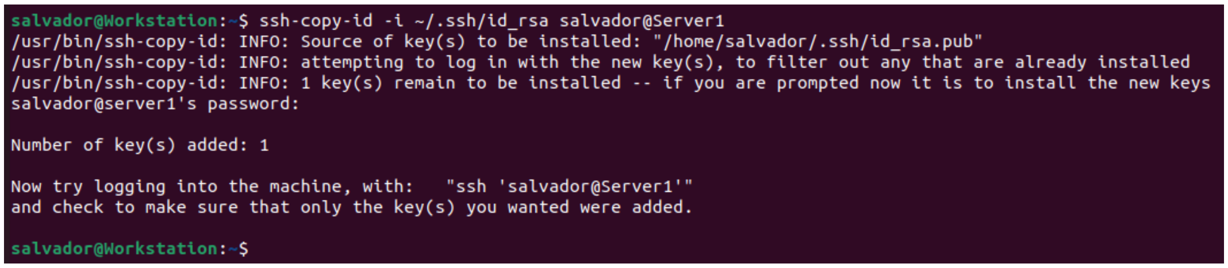
1. To use public key authentication, the public key must be copied to a server and installed in an *authorized_keys* file. This can be conveniently done using the *ssh-copy-id* tool.
2. Issue the command similar to this: *ssh-copy-id -i ~/.ssh/id_rsa user@host*



```
salvador@Workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa salvador@Workstation
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/salvador/.ssh/id_rsa.pub"
The authenticity of host 'workstation (192.168.56.103)' can't be established.
ED25519 key fingerprint is SHA256:MCgak0o6+JLeuzF1UtZ4iXpdKvIFUnattQUs1BP1aTo.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/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 prompted now it is to install the new keys
salvador@workstation's password:
Permission denied, please try again.
salvador@workstation's password:

Number of key(s) added: 1

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

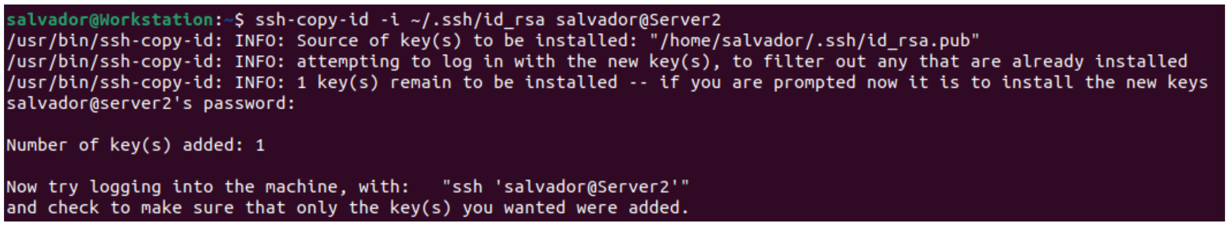
3. Once the public key has been configured on the server, the server will allow any connecting user that has the private key to log in. During the login process, the client proves possession of the private key by digitally signing the key exchange.

Server1:



```
salvador@Workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa salvador@Server1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/salvador/.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 prompted now it is to install the new keys
salvador@server1's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'salvador@Server1'"
and check to make sure that only the key(s) you wanted were added.

salvador@Workstation:~$
```

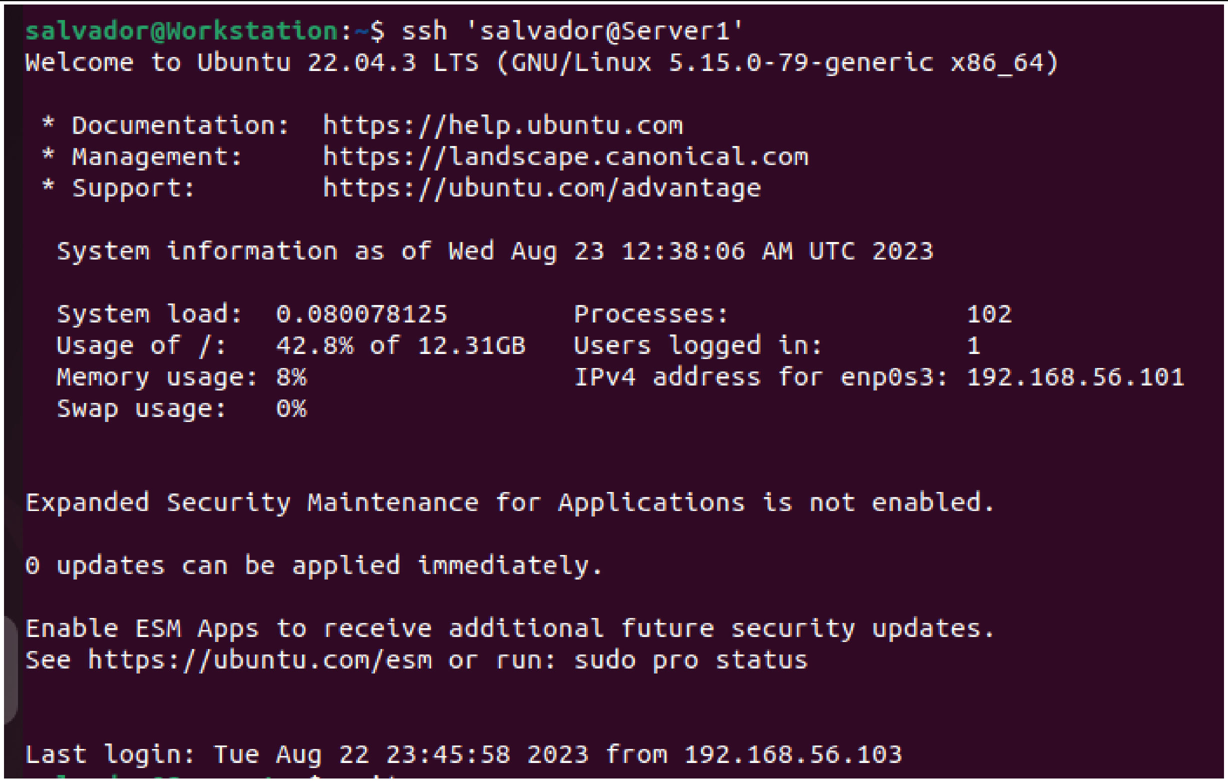
Server2:



```
salvador@Workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa salvador@Server2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/salvador/.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 prompted now it is to install the new keys
salvador@server2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'salvador@Server2'"
and check to make sure that only the key(s) you wanted were added.
```

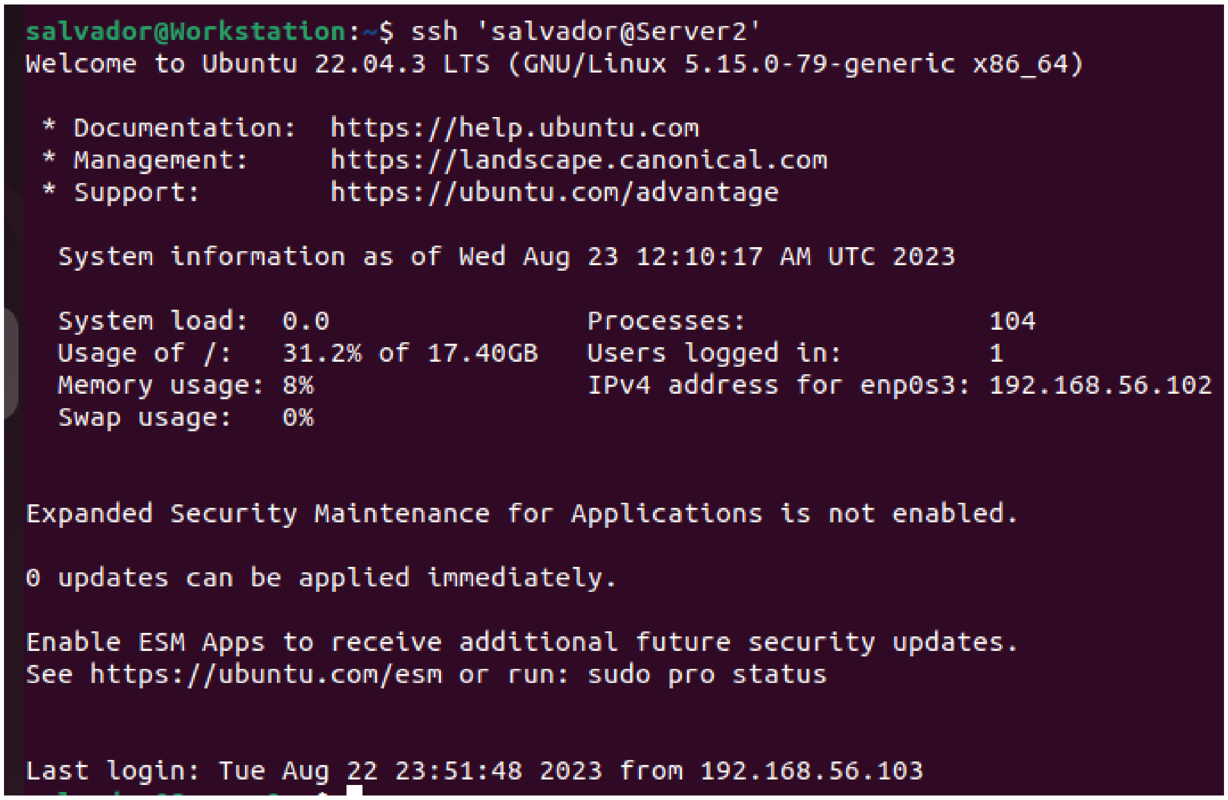
4. On the local machine, verify that you can SSH with Server 1 and Server 2. What did you notice? Did the connection ask for a password? If not, why?

Server1:



```
salvador@Workstation:~$ ssh 'salvador@Server1'
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)

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

  System information as of Wed Aug 23 12:38:06 AM UTC 2023

  System load:  0.080078125        Processes:               102
  Usage of /:   42.8% of 12.31GB   Users logged in:         1
  Memory usage: 8%                 IPv4 address for enp0s3: 192.168.56.101
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Tue Aug 22 23:45:58 2023 from 192.168.56.103
```

Server2:



```
salvador@Workstation:~$ ssh 'salvador@Server2'
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-79-generic x86_64)

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

  System information as of Wed Aug 23 12:10:17 AM UTC 2023

  System load:  0.0                Processes:               104
  Usage of /:   31.2% of 17.40GB   Users logged in:         1
  Memory usage: 8%                 IPv4 address for enp0s3: 192.168.56.102
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Tue Aug 22 23:51:48 2023 from 192.168.56.103
```

At the very first try you log in into that host, it asks for a password or passphrase but the second time or the next time you try to log in to that host it will not ask for a passphrase anymore, a seamless login is created. The use of the ssh that you did was to authenticate your working machine to those hosts (server1 and server2) and establish a connection because of the ssh key pair.

Reflections:
Answer the following:
1. How will you describe the ssh-program? What does it do?

SSH or Secure Shell is a network protocol that provides a secure connection between hosts. in the activity we ssh to have a seamless connection from a server to the workstation. In this way we can access the servers (server1 and server2) in our workstation with a secure connection, the option was to not ask for a password (passphrase) everytime you try to access that host. With that, the ssh-program was thoroughly demonstrated as a network protocol that lets you connect or interact with a host, use it remotely and have control over it with high security.

2. How do you know that you already installed the public key to the remote servers?

By determining if you can access that host inside your workstation. using the ssh command followed by the user and host, if the command successfully prompts and if it does not ask for the password again for the next time you try to access the server then you can verify that the public key was already installed to the remote servers.

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**).

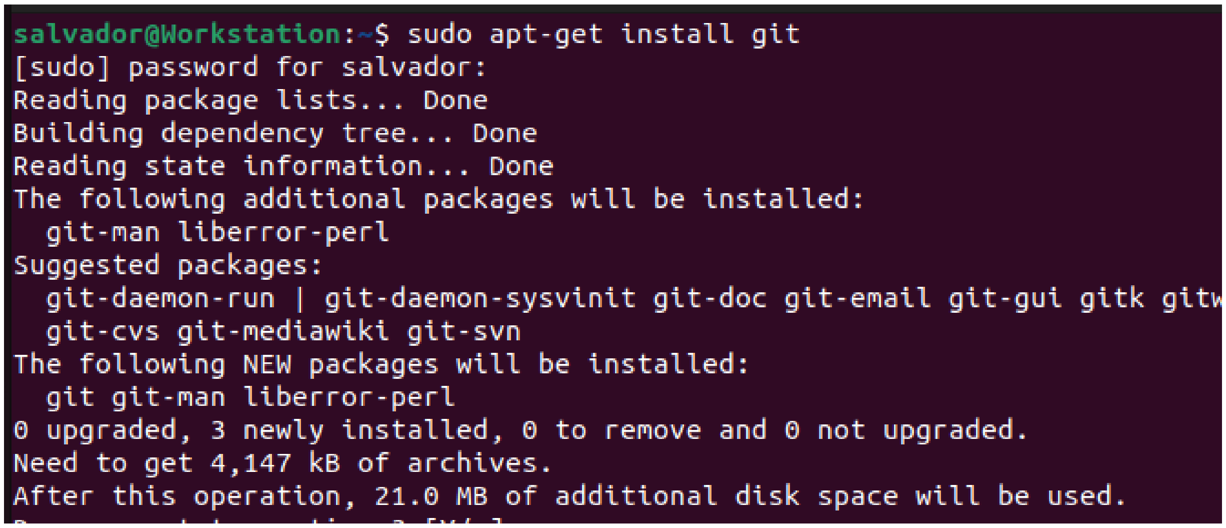
Set up Git
At the heart of GitHub is an open-source version control system (VCS) called Git. Git is responsible for everything GitHub-related that happens locally on your computer. To use Git on the command line, you'll need to download, install, and configure Git on your computer. You can also install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

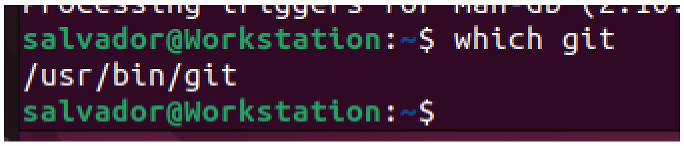
Task 3: Set up the Git Repository

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*



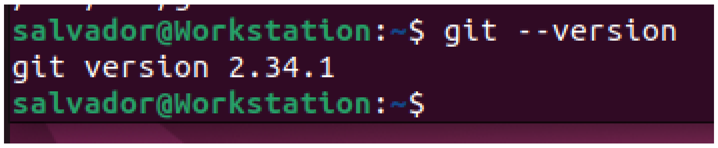
```
salvador@Workstation:~$ sudo apt-get install git
[sudo] password for salvador:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitw
  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 0 not upgraded.
Need to get 4,147 kB of archives.
After this operation, 21.0 MB of additional disk space will be used.
```

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



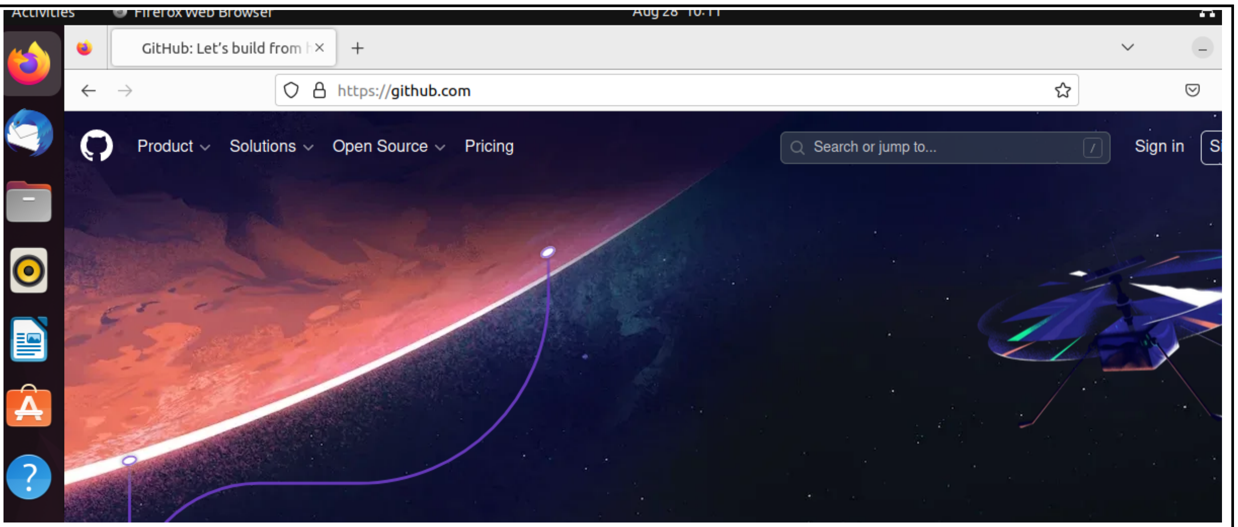
```
salvador@Workstation:~$ which git
/usr/bin/git
salvador@Workstation:~$
```

3. The version of git installed in your device is the latest. Try issuing the command *git --version* to know the version installed.

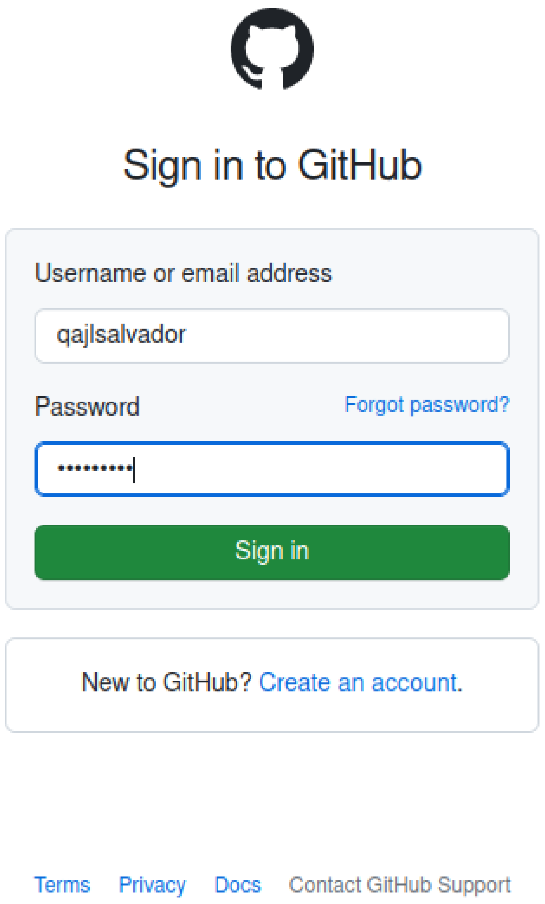


```
salvador@Workstation:~$ git --version
git version 2.34.1
salvador@Workstation:~$
```

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.



Sign in to GitHub

Username or email address

qajlsalvador

Password

Forgot password?

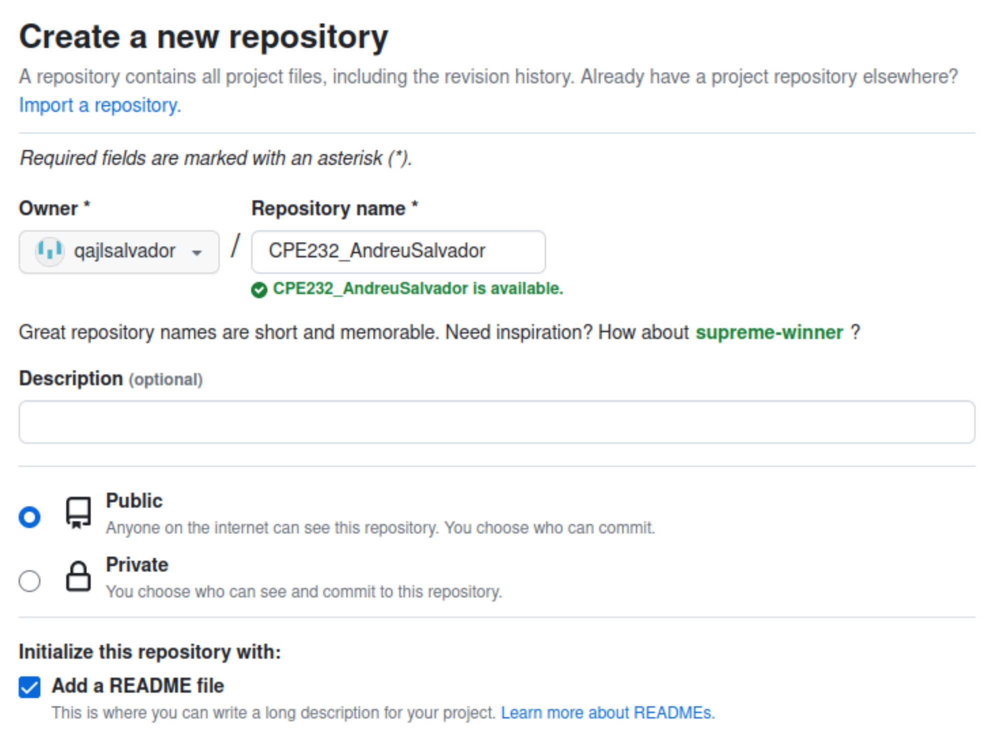
••••••••••

Sign in

New to GitHub? Create an account.

Terms Privacy Docs Contact GitHub Support

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 * qajlsalvador / **Repository name *** CPE232_AndreuSalvador

CPE232_AndreuSalvador is available.

Great repository names are short and memorable. Need inspiration? How about **supreme-winner** ?

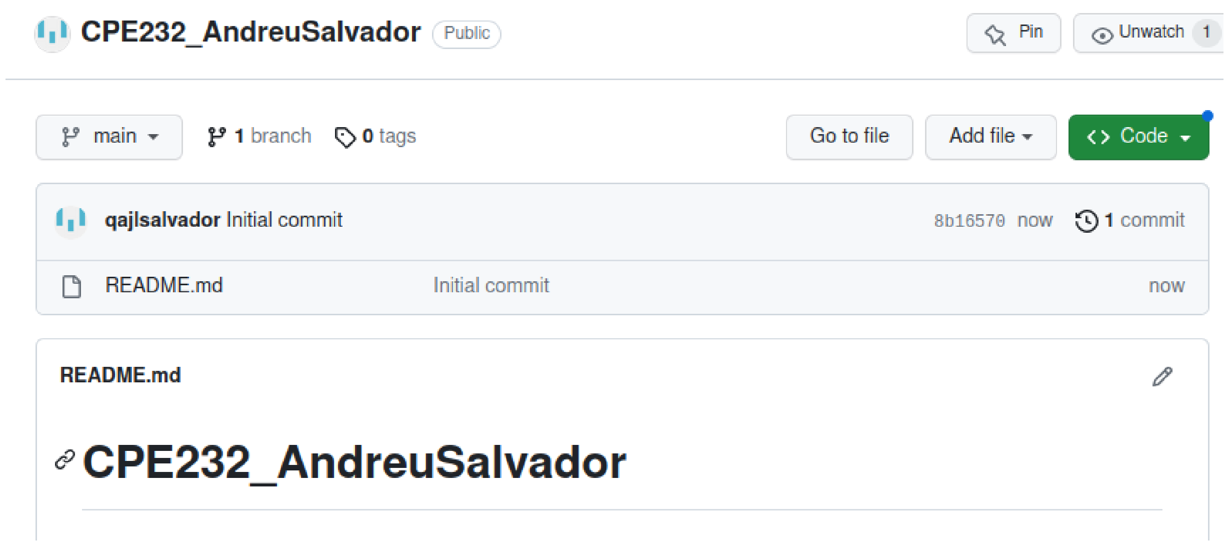
Description (optional)

[✓ **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.

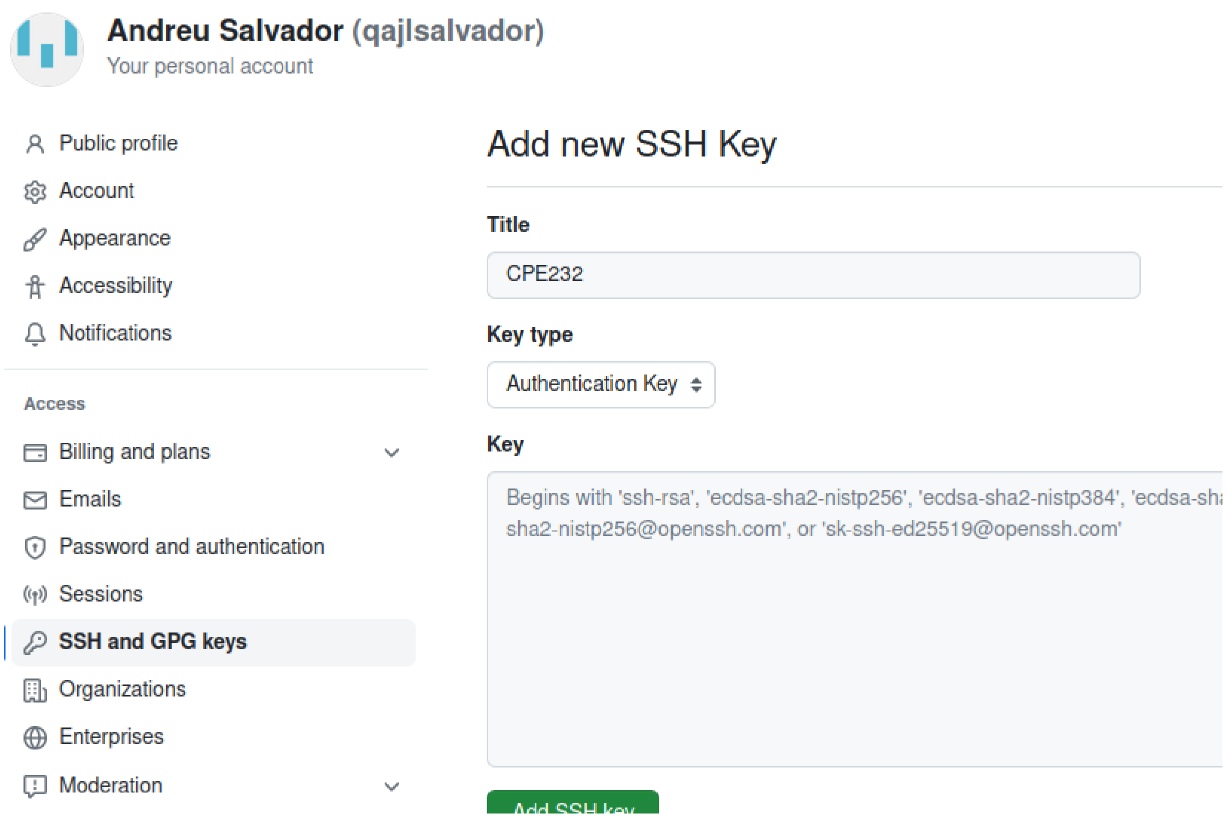

CPE232_AndreuSalvador Public

qajlsalvador Initial commit	8b16570 now	1 commit
README.md	Initial commit	now

README.md

CPE232_AndreuSalvador

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.



Add new SSH Key

Title

CPE232

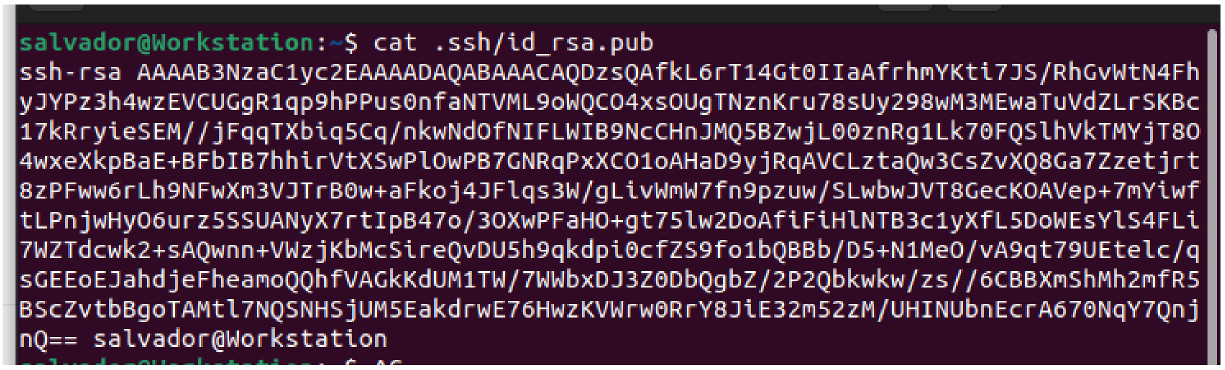
Key type

Authentication Key

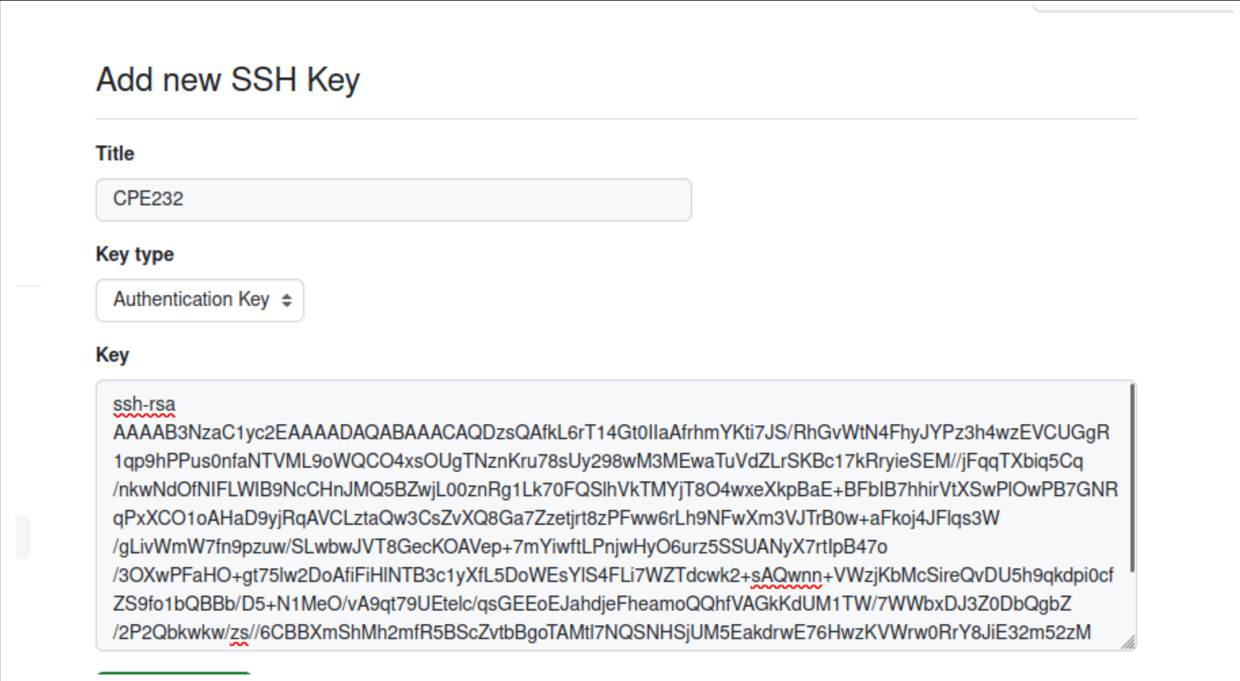
Key

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sh
sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'

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.



```
salvador@Workstation:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDzsQAfkL6rT14Gt0IIaAfrhmYKti7JS/RhGvWtN4Fh
yJYPz3h4wzEVCUGgR1qp9hPPus0nfaNTVML9oWQCO4xsOUgTNznKru78sUy298wM3MEwaTuVdZLrSKBc
17kRryieSEM//jFqqTXbiq5Cq/nkwNdOfNIFLWIB9NcCHnJMQ5BZwjL00znRg1Lk70FQSlhVkTMYjT8O
4wxeXkpBaE+BFbIB7hhirVtXSwPlOwPB7GNRqPxXCO1oAHaD9yjRqAVCLztaQw3CsZvXQ8Ga7Zzetjrt
8zPFww6rLh9NFwXm3VJTrB0w+aFkoj4JFlqs3W/gLivWmW7fn9pzuw/SLwbwJVT8GecKOAVep+7mYiwf
tLPnjwHyO6urz5SSUANyX7rtIpB47o/3OXwPFaHO+gt75lw2DoAfiFiHlNTB3c1yXfL5DoWEsYlS4FLi
7WZTdcwk2+sAQwnn+VWzjKbMcSireQvDU5h9qkdpi0cfZS9fo1bQBBb/D5+N1MeO/vA9qt79UEtelc/q
sGEEoEJahdjeFheamoQQhfVAGkKdUM1TW/7WWbxDJ3Z0DbQgbZ/2P2Qbkwkw/zs//6CBBXmShMh2mfR5
BScZvtbBgoTAMtl7NQSNHSjUM5EakdrwE76HwzKVWrw0RrY8JiE32m52zM/UHINUbnEcrA670NqY7Qnj
nQ== salvador@Workstation
```



Add new SSH Key

Title

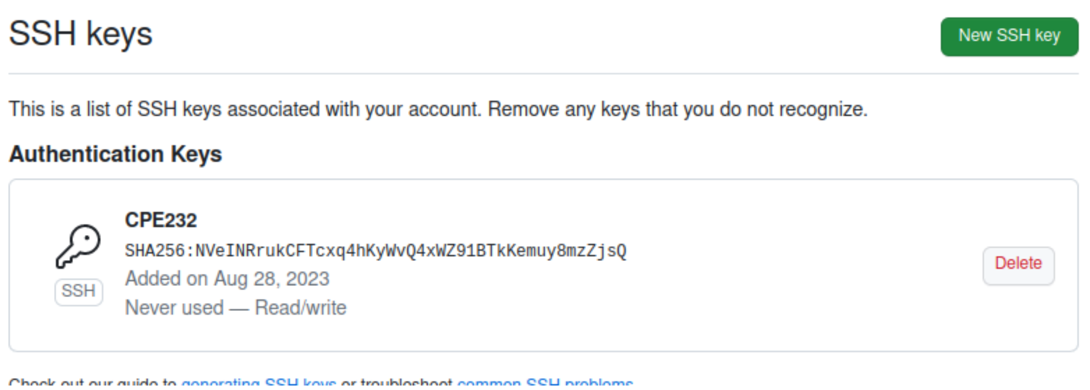
CPE232

Key type

Authentication Key

Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDzsQAfkL6rT14Gt0IlaAfrhmYKti7JS/RhGvWtN4FhyJYPz3h4wzEVCUGgR
1qp9hPPus0nfaNTVML9oWQCO4xsOUgTNznKru78sUy298wM3MEwaTuVdZLrSKBc17kRryieSEM//jFqqTXbiq5Cq
/nkwNdOfNIFLWIB9NcCHnJMQ5BZwjL00znRg1Lk70FQSlhVkTMYjT8O4wxeXkpBaE+BFblB7hhirVtXSwPlOwPB7GNR
qPxXCO1oAHaD9yjRqAVCLztaQw3CsZvXQ8Ga7Zzetjrt8zPFww6rLh9NFwXm3VJTrB0w+aFkoj4JFlqs3W
/gLivWmW7fn9pzuw/SLwbwJVT8GecKOAVep+7mYiwftLPnjwHyO6urz5SSUANyX7rtlpB47o
/3OXwPFaHO+gt75lw2DoAfiFiHlNTB3c1yXfL5DoWEsYlS4FLi7WZTdcwk2+sAQwnn+VWzjKbMcSireQvDU5h9qkdpi0cf
ZS9fo1bQBBb/D5+N1MeO/vA9qt79UEtelc/qsGEEoEJahdjeFheamoQQhfVAGkKdUM1TW/7WWbxDJ3Z0DbQgbZ
/2P2Qbkwkw/zs//6CBBXmShMh2mfR5BScZvtbBgoTAMtl7NQSNHSjUM5EakdrwE76HwzKVWrw0RrY8JiE32m52zM
```



SSH keys

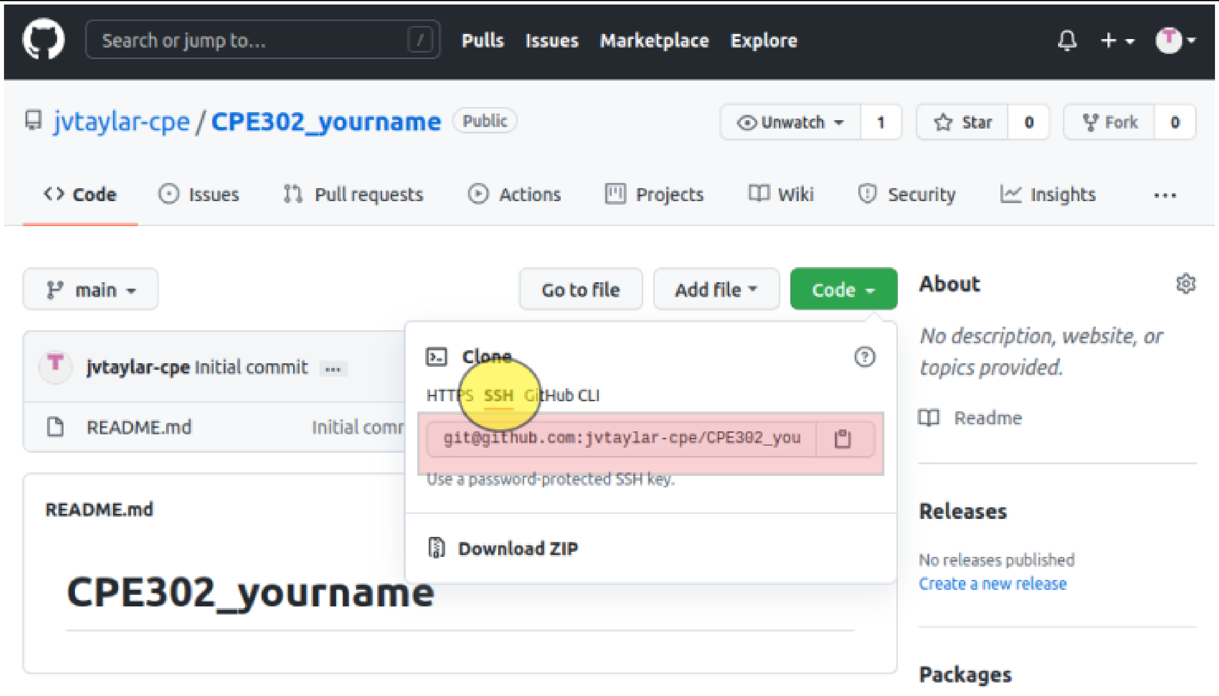
This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

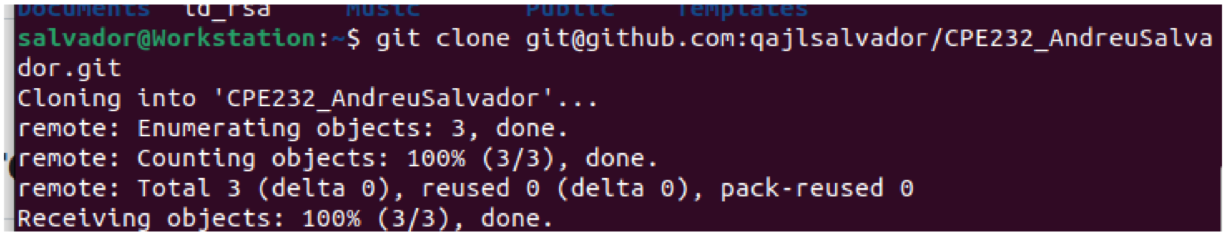
CPE232
SHA256:NVeINRrukCFTcxq4hKyWvQ4xWZ91BTkKemuy8mzZjsQ
Added on Aug 28, 2023
Never used — Read/write

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

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.

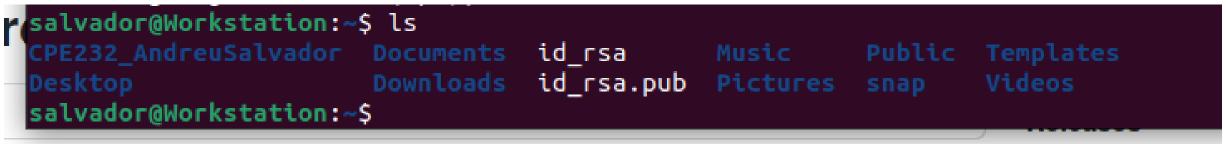


e. Issue the command git clone followed by the copied link. For example, *git clone git@github.com:jvtaylar-cpe/CPE232_yourname.git*. When prompted to continue connecting, type yes and press enter.

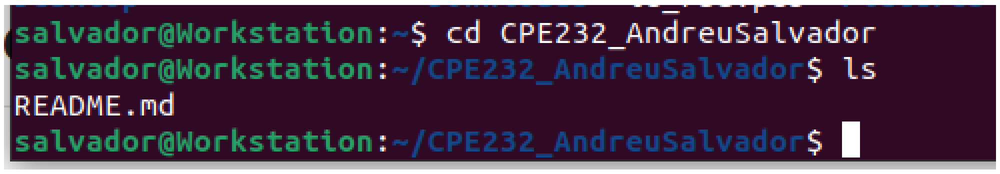


```
salvador@Workstation:~$ git clone git@github.com:qajlsalvador/CPE232_AndreuSalva
dor.git
Cloning into 'CPE232_AndreuSalvador'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

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.



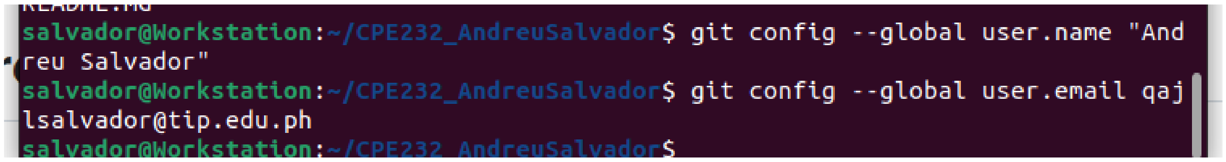
```
salvador@Workstation:~$ ls
CPE232_AndreuSalvador  Documents  id_rsa      Music     Public  Templates
Desktop                Downloads  id_rsa.pub  Pictures  snap    Videos
salvador@Workstation:~$
```



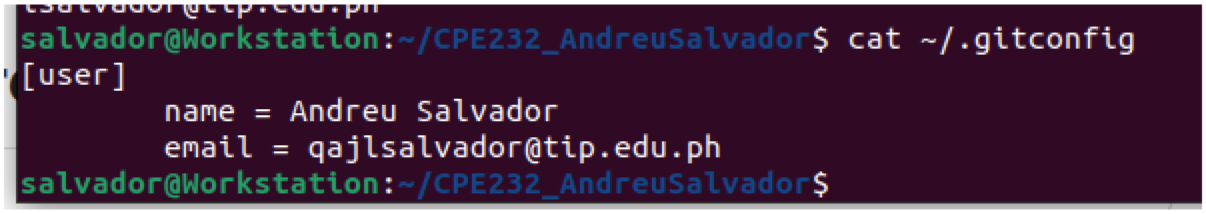
```
salvador@Workstation:~$ cd CPE232_AndreuSalvador
salvador@Workstation:~/CPE232_AndreuSalvador$ ls
README.md
salvador@Workstation:~/CPE232_AndreuSalvador$
```

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*

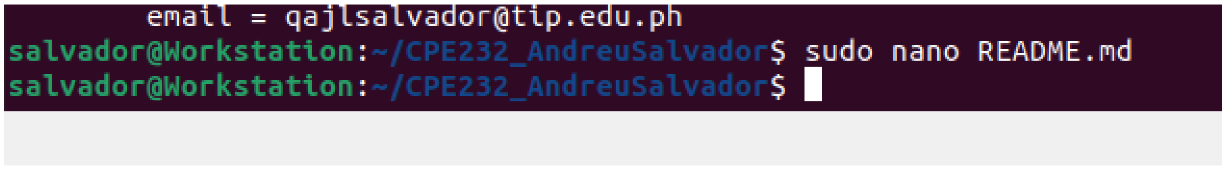


```
salvador@Workstation:~/CPE232_AndreuSalvador$ git config --global user.name "And
reu Salvador"
salvador@Workstation:~/CPE232_AndreuSalvador$ git config --global user.email qaj
lsalvador@tip.edu.ph
salvador@Workstation:~/CPE232_AndreuSalvador$
```

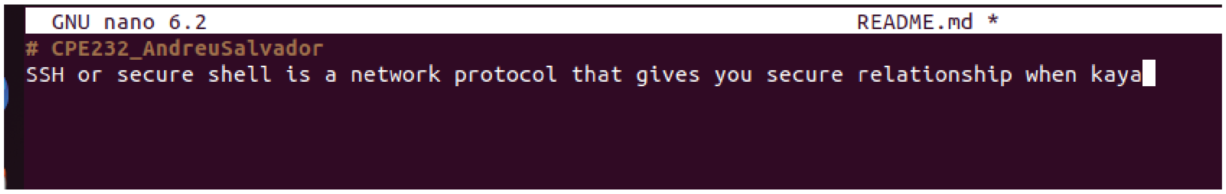


```
salvador@Workstation:~/CPE232_AndreuSalvador$ cat ~/.gitconfig
[user]
        name = Andreu Salvador
        email = qajlsalvador@tip.edu.ph
salvador@Workstation:~/CPE232_AndreuSalvador$
```

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.

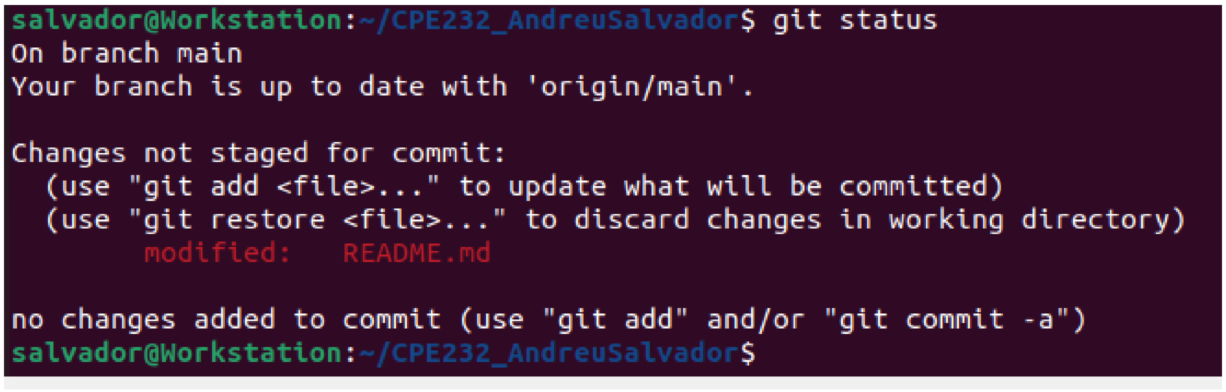


```
        email = qajlsalvador@tip.edu.ph
salvador@Workstation:~/CPE232_AndreuSalvador$ sudo nano README.md
salvador@Workstation:~/CPE232_AndreuSalvador$
```



```
  GNU nano 6.2                                              README.md *
# CPE232_AndreuSalvador
SSH or secure shell is a network protocol that gives you secure relationship when kaya
```

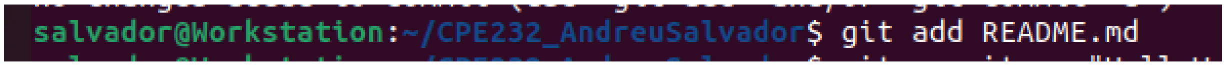
i. Use the *git status* command to display the state of the working directory and the staging area. This command shows which changes have been staged, which haven't, and which files aren't being tracked by Git. Status output does not show any information regarding the committed project history. What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



```
salvador@Workstation:~/CPE232_AndreuSalvador$ 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 restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
salvador@Workstation:~/CPE232_AndreuSalvador$
```

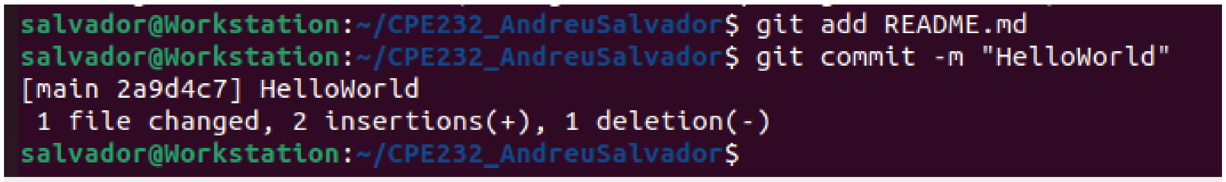
Using this command, you can check the "status" of a local repository, it shows the name of the branch, the current status of the branch, changes that occurred or been staged or commitment.

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



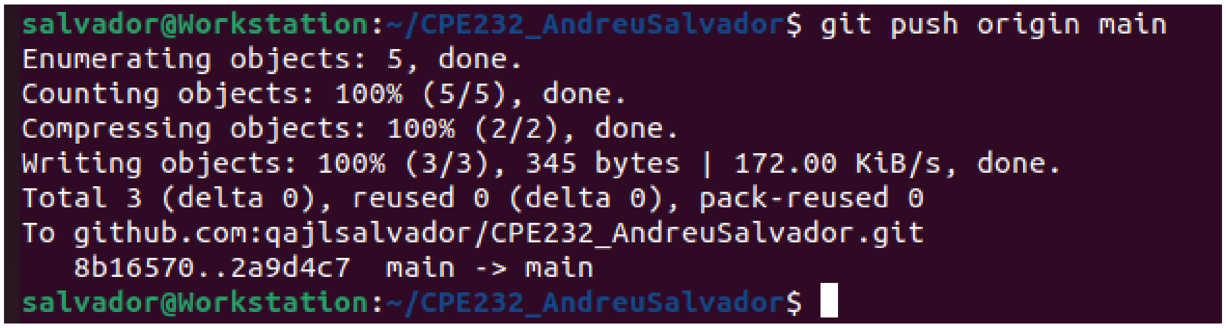
```
salvador@Workstation:~/CPE232_AndreuSalvador$ git add README.md
```

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.



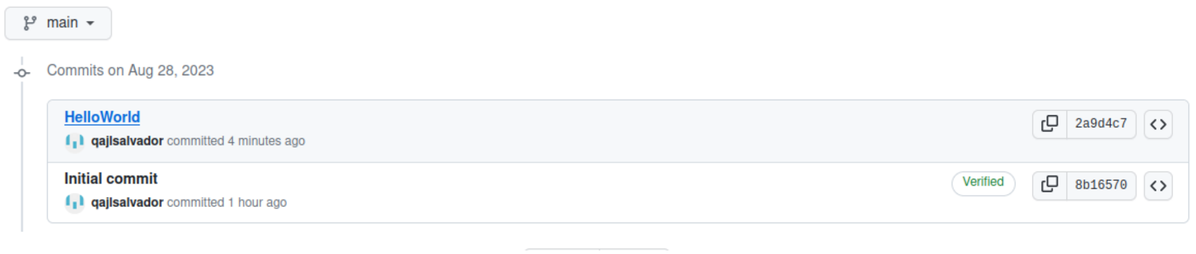
```
salvador@Workstation:~/CPE232_AndreuSalvador$ git add README.md
salvador@Workstation:~/CPE232_AndreuSalvador$ git commit -m "HelloWorld"
[main 2a9d4c7] HelloWorld
 1 file changed, 2 insertions(+), 1 deletion(-)
salvador@Workstation:~/CPE232_AndreuSalvador$
```

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.*



```
salvador@Workstation:~/CPE232_AndreuSalvador$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 345 bytes | 172.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:qajlsalvador/CPE232_AndreuSalvador.git
   8b16570..2a9d4c7  main -> main
salvador@Workstation:~/CPE232_AndreuSalvador$
```

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 how long the last commit is. 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.

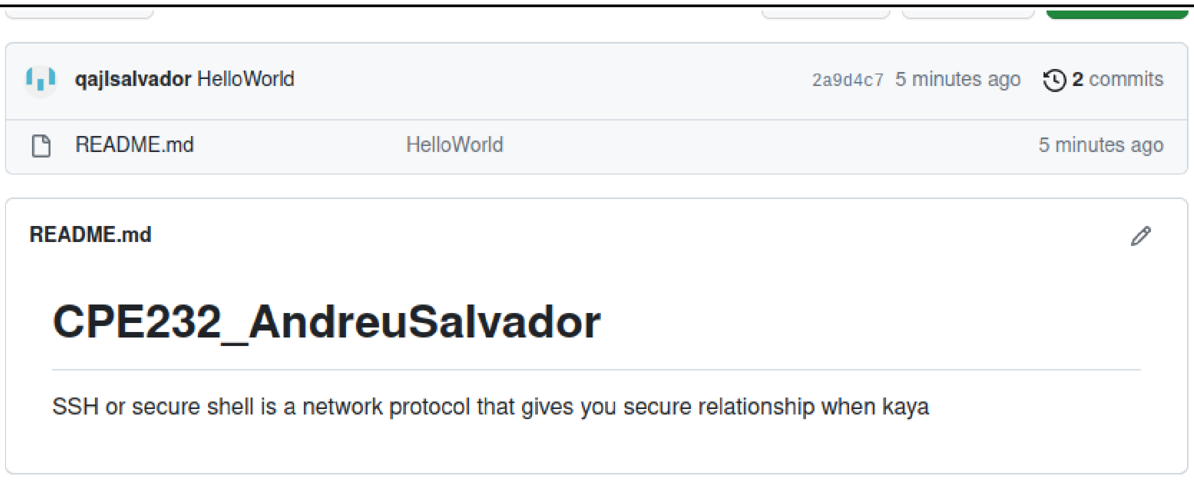


main

Commits on Aug 28, 2023

HelloWorld
qajlsalvador committed 4 minutes ago — 2a9d4c7

Initial commit
qajlsalvador committed 1 hour ago — Verified — 8b16570



qajlsalvador HelloWorld 2a9d4c7 5 minutes ago 2 commits

README.md HelloWorld 5 minutes ago

README.md

CPE232_AndreuSalvador

SSH or secure shell is a network protocol that gives you secure relationship when kaya

There is a status that says “2 commits” and when the commitment was done which is 5 minutes ago.

Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?

So far throughout this activity, we’ve connected the workstation to the server as well as created a github account and also connected that to the workstation which we can edit its contents using commands.

4. How important is the inventory file?

The Inventory file is important to not just for clear and neat management of the files, it allows you to access different keys in which where these are located to. it plays an important part in using the ansible commands as it is where you can find the needed content or information for this command.

Conclusions/Learnings:

Throughout the experience of working with the activity, I've established a ssh connection, a secured connection with seamless log in for the server and host . not only that, but the fundamentals on setting up a git repository was also included in the activity which is something that will be used in the future activities in this course. Overall, this activity was able to demonstrate how a ssh connection is made and what are the needed contents or the terms that are important in creating the ssh connection and the git repository.