Name: Buenvenida, Ken Benedict D.	**Date Performed: 08/22/2023**
Course/Section: CpE232 - CpE31S4	**Date Submitted:**
Instructor: Engr. Jonathan Taylar	**Semester and SY: 2023 - 2024 1st Semester**

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

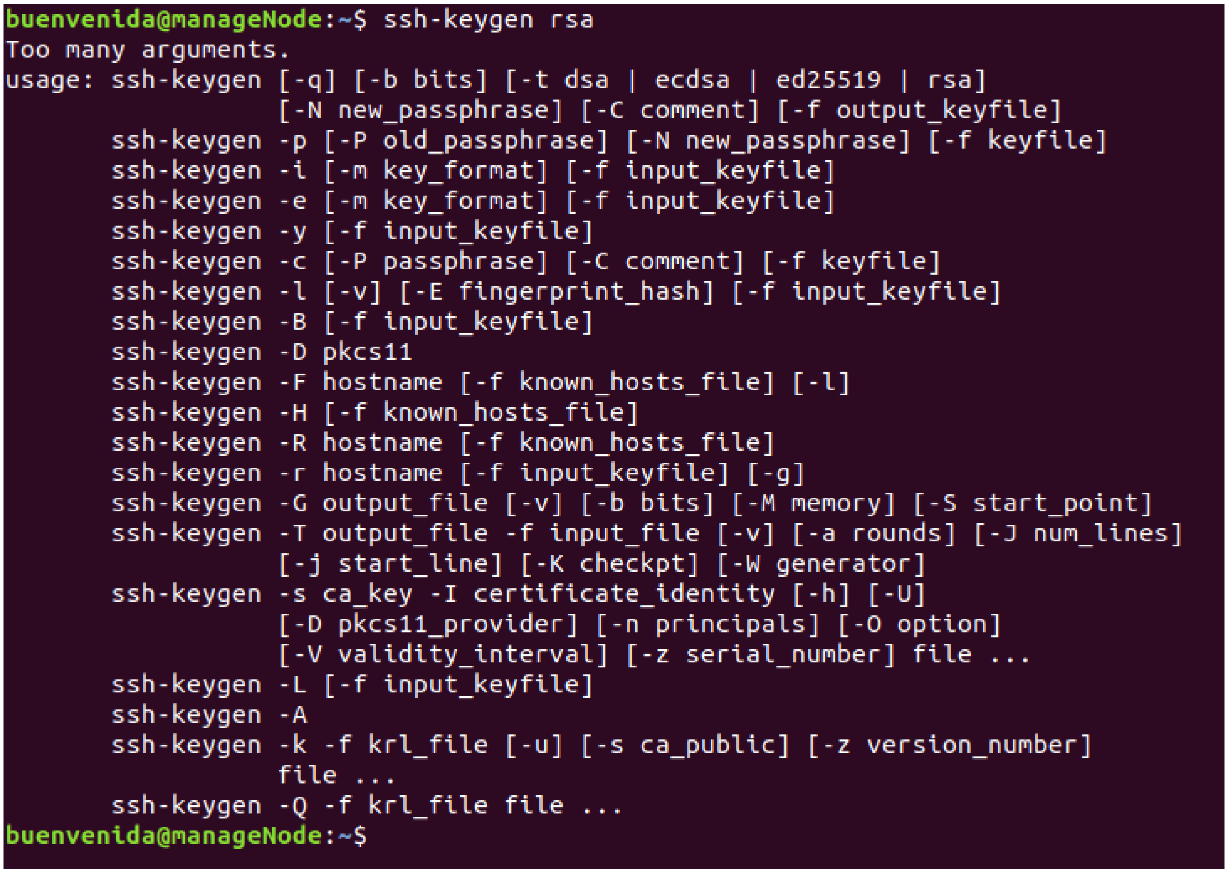
The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.

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.

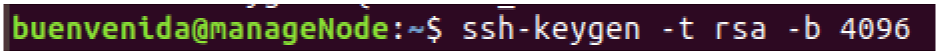
Task 1: Create an SSH Key Pair for User Authentication

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*.



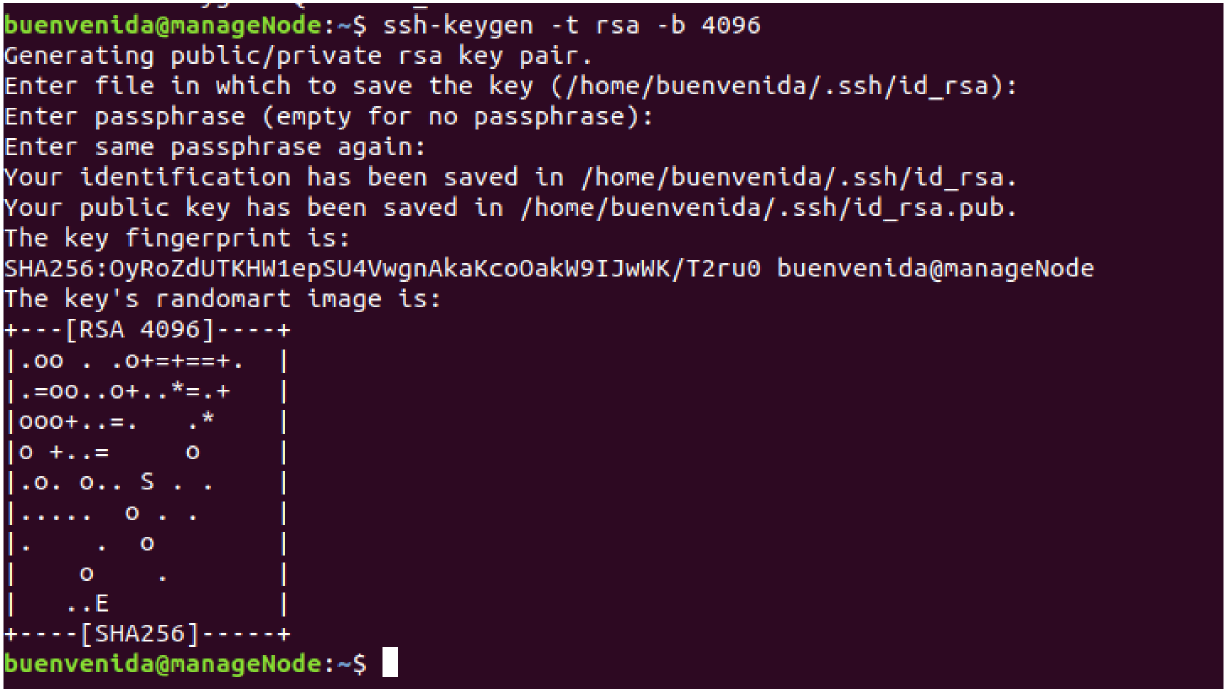
```
buenvenida@manageNode:~$ ssh-keygen rsa
Too many arguments.
usage: ssh-keygen [-q] [-b bits] [-t dsa | ecdsa | ed25519 | rsa]
                  [-N new_passphrase] [-C comment] [-f output_keyfile]
       ssh-keygen -p [-P old_passphrase] [-N new_passphrase] [-f keyfile]
       ssh-keygen -i [-m key_format] [-f input_keyfile]
       ssh-keygen -e [-m key_format] [-f input_keyfile]
       ssh-keygen -y [-f input_keyfile]
       ssh-keygen -c [-P passphrase] [-C comment] [-f keyfile]
       ssh-keygen -l [-v] [-E fingerprint_hash] [-f input_keyfile]
       ssh-keygen -B [-f input_keyfile]
       ssh-keygen -D pkcs11
       ssh-keygen -F hostname [-f known_hosts_file] [-l]
       ssh-keygen -H [-f known_hosts_file]
       ssh-keygen -R hostname [-f known_hosts_file]
       ssh-keygen -r hostname [-f input_keyfile] [-g]
       ssh-keygen -G output_file [-v] [-b bits] [-M memory] [-S start_point]
       ssh-keygen -T output_file -f input_file [-v] [-a rounds] [-J num_lines]
                  [-j start_line] [-K checkpt] [-W generator]
       ssh-keygen -s ca_key -I certificate_identity [-h] [-U]
                  [-D pkcs11_provider] [-n principals] [-O option]
                  [-V validity_interval] [-z serial_number] file ...
       ssh-keygen -L [-f input_keyfile]
       ssh-keygen -A
       ssh-keygen -k -f krl_file [-u] [-s ca_public] [-z version_number]
                  file ...
       ssh-keygen -Q -f krl_file file ...
buenvenida@manageNode:~$
```

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.



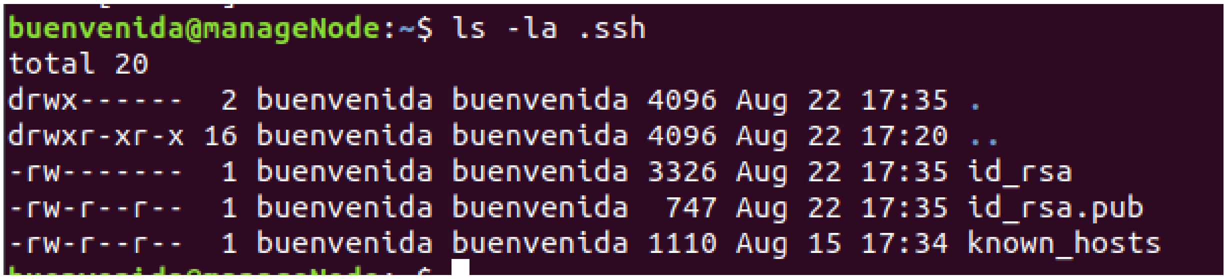
```
buenvenida@manageNode:~$ ssh-keygen -t rsa -b 4096
```

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.



```
buenvenida@manageNode:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/buenvenida/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/buenvenida/.ssh/id_rsa.
Your public key has been saved in /home/buenvenida/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:OyRoZdUTKHW1epSU4VwgnAkaKcoOakW9IJwWK/T2ru0 buenvenida@manageNode
The key's randomart image is:
+---[RSA 4096]----+
|.oo . .o+=+==+.  |
|.=oo..o+..*=.+   |
|ooo+..=.    .*   |
|o +..=      o    |
|.o. o.. S . .    |
|.....  o . .     |
|.    .  o        |
|    o    .       |
|   ..E           |
+----[SHA256]-----+
buenvenida@manageNode:~$
```

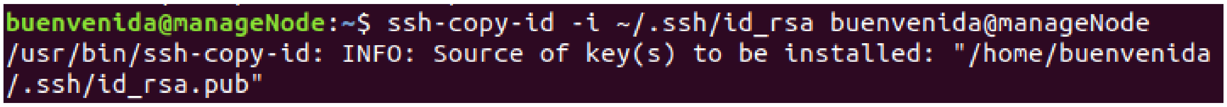
4. Verify that you have created the key by issuing the command *ls -la .ssh.* The command should show the .ssh directory containing a pair of keys. For example, id_rsa.pub and id_rsa.



```
buenvenida@manageNode:~$ ls -la .ssh
total 20
drwx------  2 buenvenida buenvenida 4096 Aug 22 17:35 .
drwxr-xr-x 16 buenvenida buenvenida 4096 Aug 22 17:20 ..
-rw-------  1 buenvenida buenvenida 3326 Aug 22 17:35 id_rsa
-rw-r--r--  1 buenvenida buenvenida  747 Aug 22 17:35 id_rsa.pub
-rw-r--r--  1 buenvenida buenvenida 1110 Aug 15 17:34 known_hosts
```

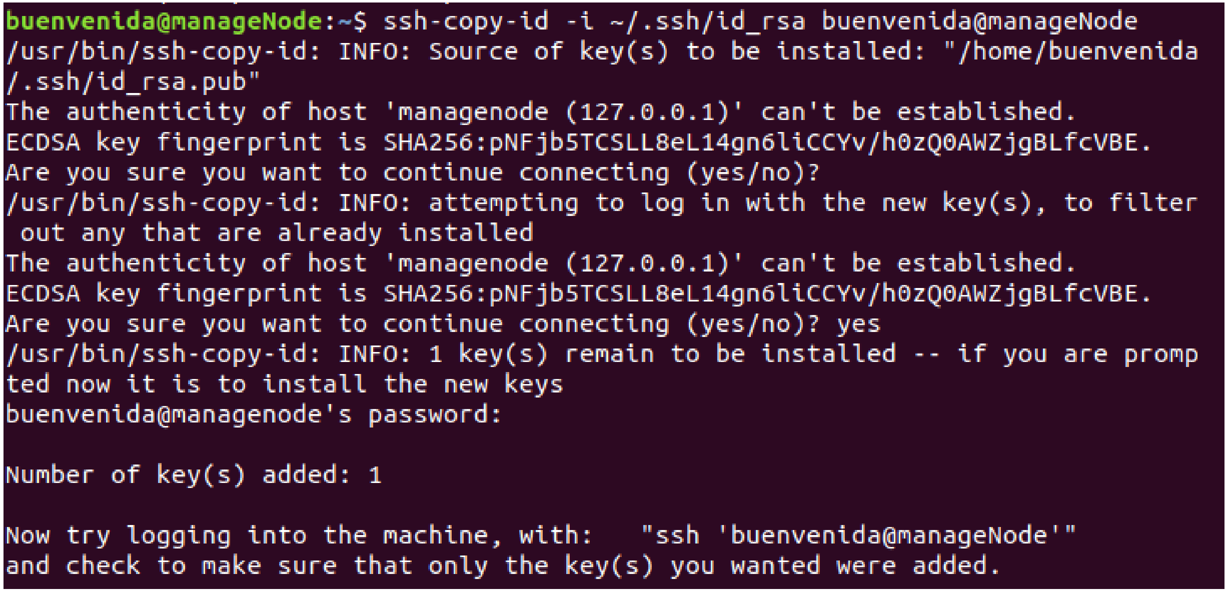
Task 2: Copying the Public Key to the remote servers

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.



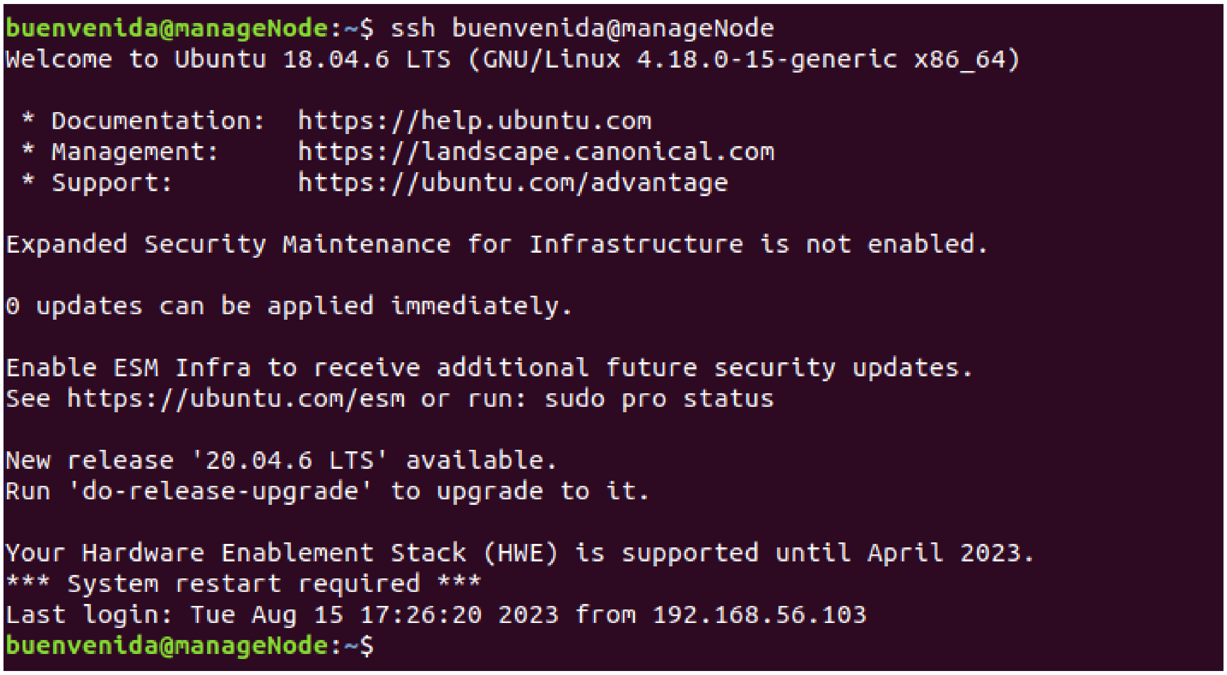
```
buenvenida@manageNode:~$ ssh-copy-id -i ~/.ssh/id_rsa buenvenida@manageNode
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/buenvenida
/.ssh/id_rsa.pub"
```

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



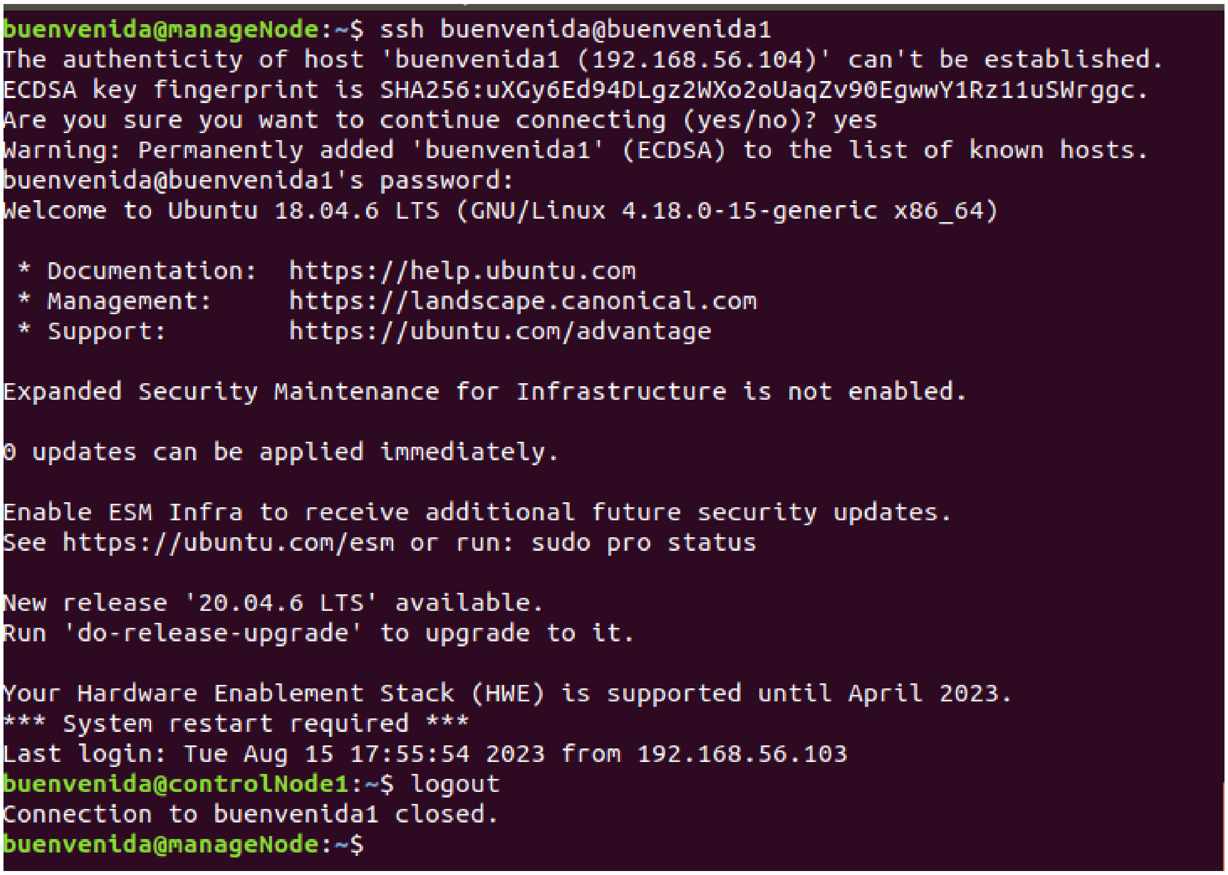
```
buenvenida@manageNode:~$ ssh-copy-id -i ~/.ssh/id_rsa buenvenida@manageNode
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/buenvenida
/.ssh/id_rsa.pub"
The authenticity of host 'managenode (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:pNFjb5TCSLL8eL14gn6liCCYv/h0zQ0AWZjgBLfcVBE.
Are you sure you want to continue connecting (yes/no)?
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
 out any that are already installed
The authenticity of host 'managenode (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:pNFjb5TCSLL8eL14gn6liCCYv/h0zQ0AWZjgBLfcVBE.
Are you sure you want to continue connecting (yes/no)? yes
/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
buenvenida@managenode's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'buenvenida@manageNode'"
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.



```
buenvenida@manageNode:~$ ssh buenvenida@manageNode
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.18.0-15-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

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

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.
*** System restart required ***
Last login: Tue Aug 15 17:26:20 2023 from 192.168.56.103
buenvenida@manageNode:~$
```

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?



```
buenvenida@manageNode:~$ ssh buenvenida@buenvenida1
The authenticity of host 'buenvenida1 (192.168.56.104)' can't be established.
ECDSA key fingerprint is SHA256:uXGy6Ed94DLgz2WXo2oUaqZv90EgwwY1Rz11uSWrggc.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'buenvenida1' (ECDSA) to the list of known hosts.
buenvenida@buenvenida1's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.18.0-15-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

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

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.
*** System restart required ***
Last login: Tue Aug 15 17:55:54 2023 from 192.168.56.103
buenvenida@controlNode1:~$ logout
Connection to buenvenida1 closed.
buenvenida@manageNode:~$
```

Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?
2. How do you know that you already installed the public key to the remote servers?

Part 2: Discussion

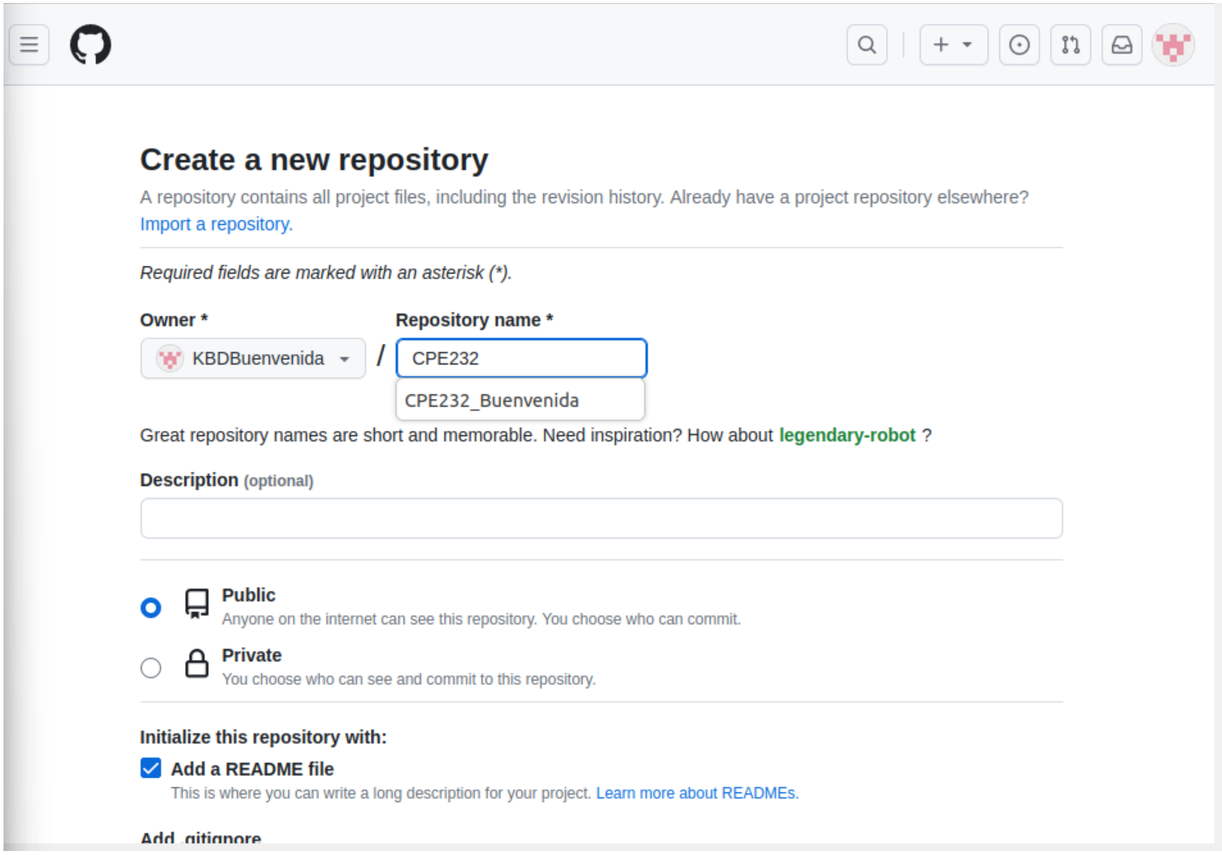
Provide screenshots for each task.

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



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 * KBDBuenvenida / **Repository name *** CPE232

CPE232_Buenvenida

Great repository names are short and memorable. Need inspiration? How about **legendary-robot** ?

Description (optional)

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

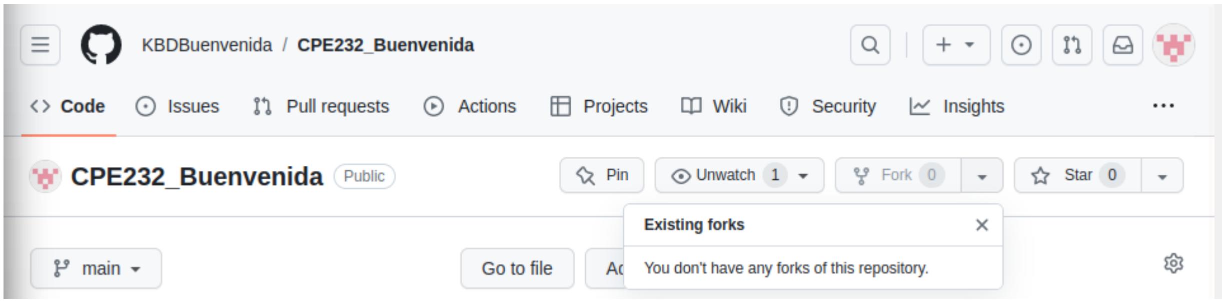
○ **Private**
You choose who can see and commit to this repository.

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

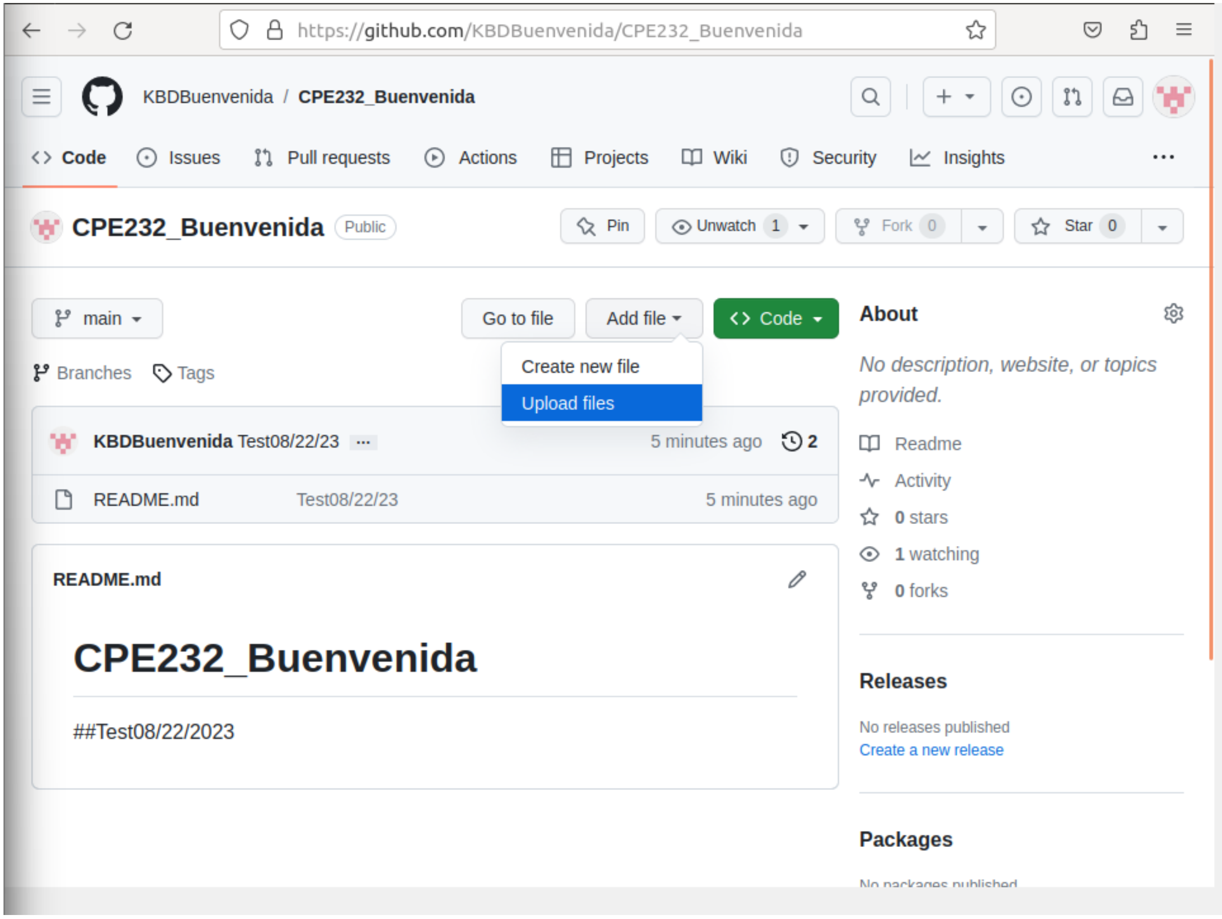
- Forking a repository



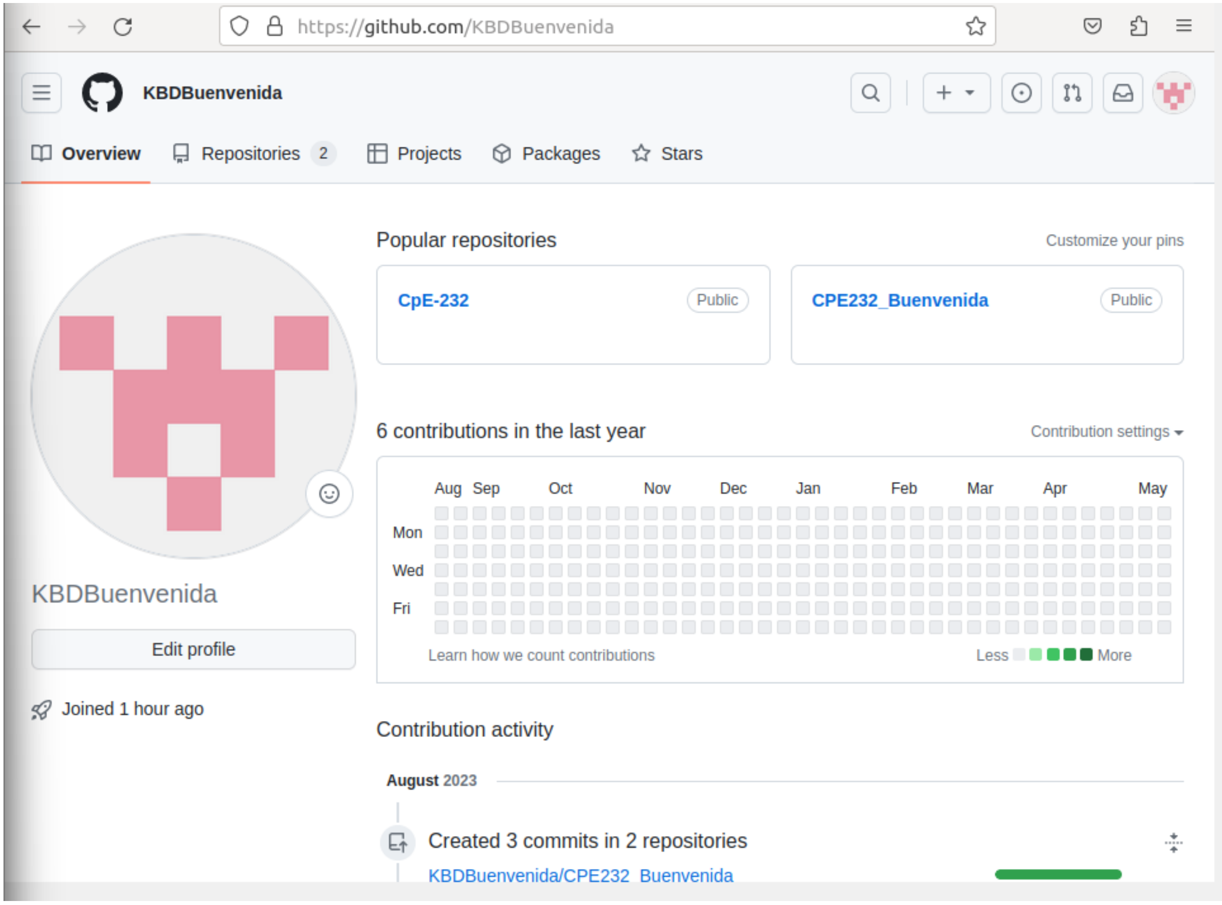
Existing forks

You don't have any forks of this repository.

- Managing files



- Being social



https://github.com/KBDBuenvenida

KBDBuenvenida

Overview | Repositories 2 | Projects | Packages | Stars

Popular repositories — Customize your pins

CpE-232 — Public

CPE232_Buenvenida — Public

KBDBuenvenida

Edit profile

Joined 1 hour ago

6 contributions in the last year — Contribution settings

Contribution activity

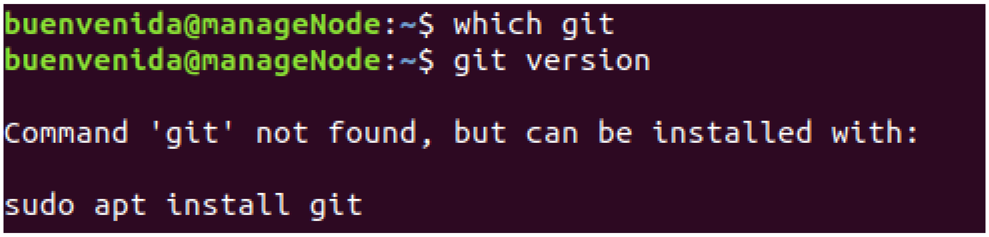
August 2023

Created 3 commits in 2 repositories

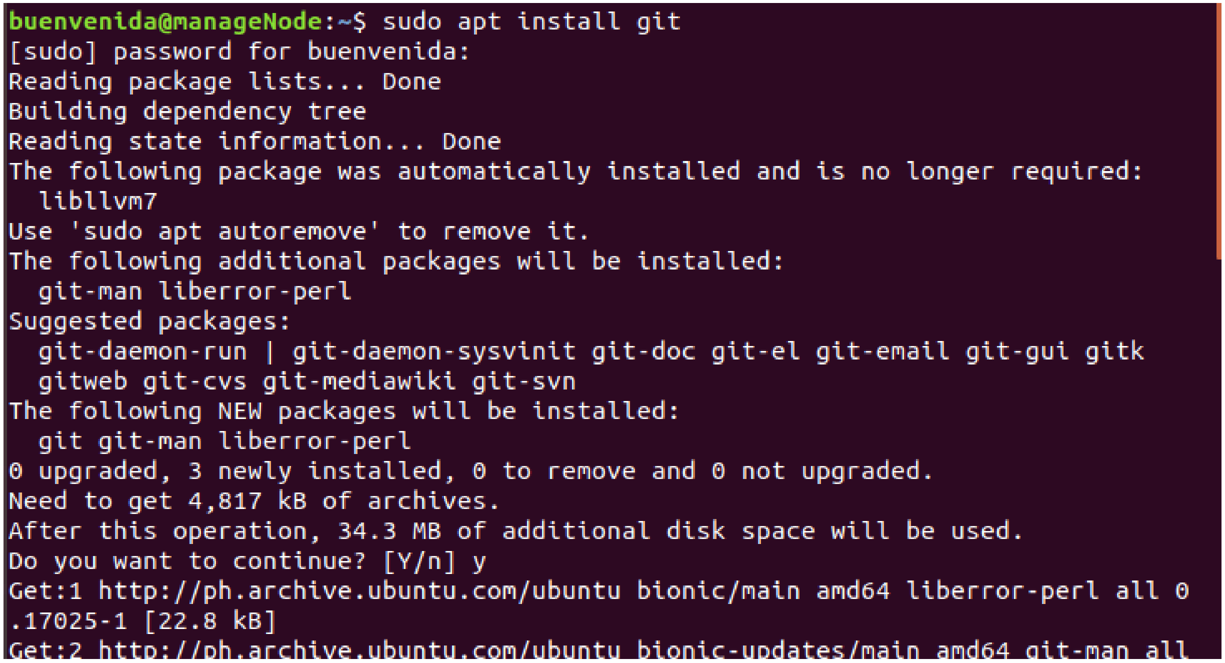
KBDBuenvenida/CPE232_Buenvenida

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*

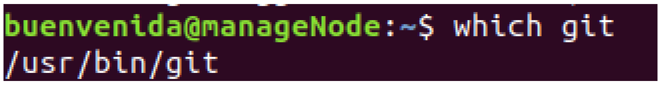


```
buenvenida@manageNode:~$ which git
buenvenida@manageNode:~$ git version

Command 'git' not found, but can be installed with:

sudo apt install git
```



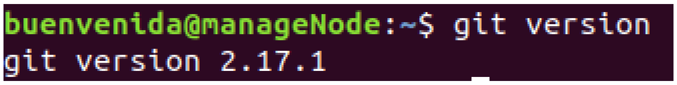
```
buenvenida@manageNode:~$ sudo apt install git
[sudo] password for buenvenida:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
The 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 0 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
```

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



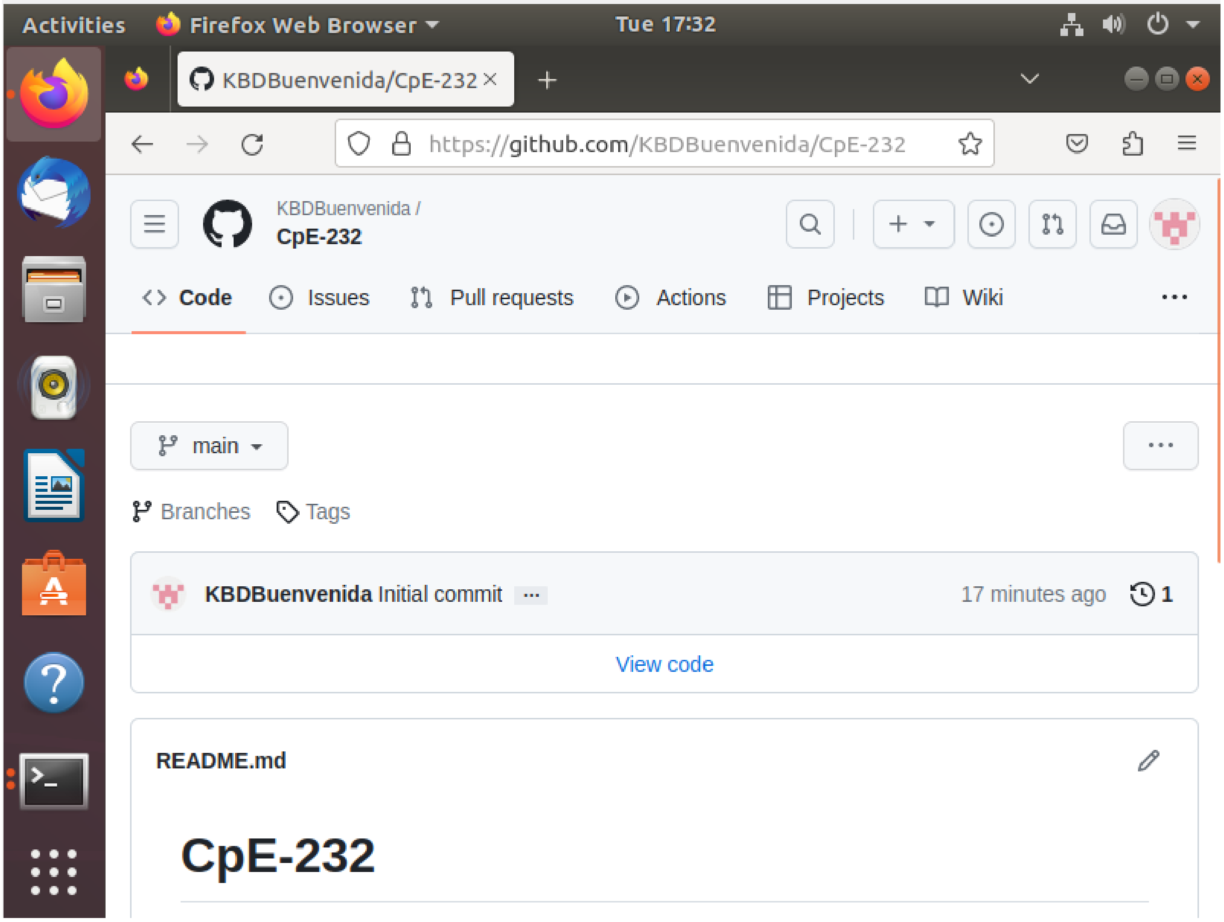
```
buenvenida@manageNode:~$ which git
/usr/bin/git
```

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



```
buenvenida@manageNode:~$ git version
git version 2.17.1
```

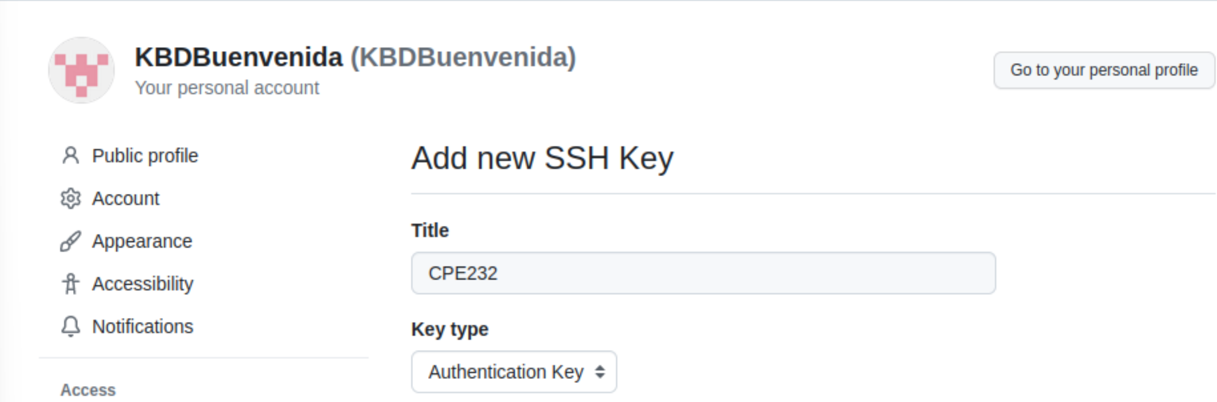
4. Using the browser in the local machine, go to www.github.com.



KBDBuenvenida Initial commit

CpE-232

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.
    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.

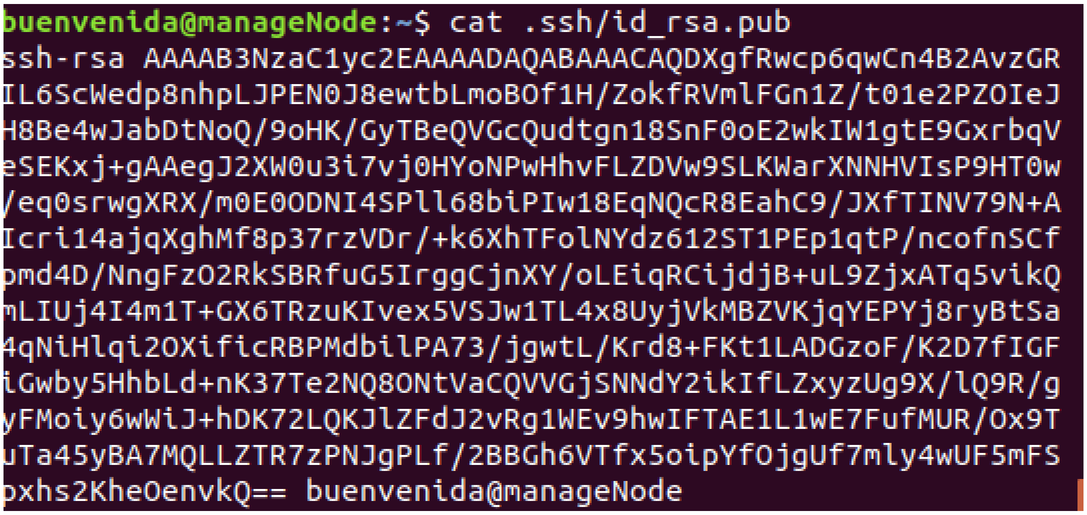


Add new SSH Key

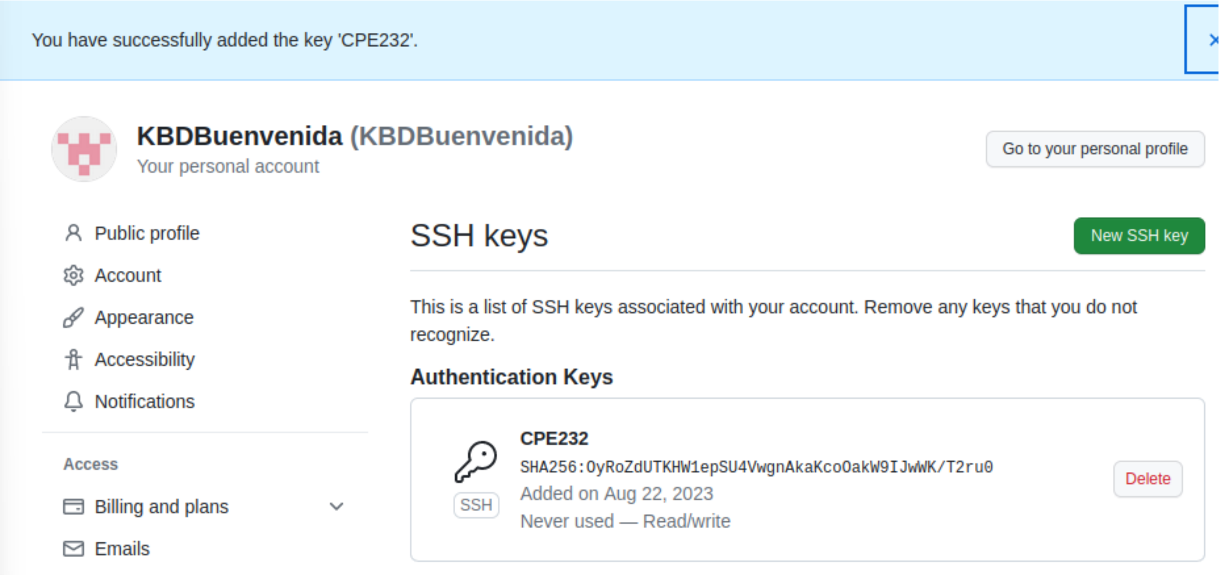
Title: CPE232

Key type: Authentication Key

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.



```
buenvenida@manageNode:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDXgfRwcp6qwCn4B2AvzGR
IL6ScWedp8nhpLJPEN0J8ewtbLmoBOf1H/ZokfRVmlFGn1Z/t01e2PZOIeJ
H8Be4wJabDtNoQ/9oHK/GyTBeQVGcQudtgn18SnF0oE2wkIW1gtE9GxrbqV
eSEKxj+gAAegJ2XW0u3i7vj0HYoNPwHhvFLZDVw9SLKWarXNNHVIsP9HT0w
/eq0srwgXRX/m0E0ODNI4SPll68biPIw18EqNQcR8EahC9/JXfTINV79N+A
Icri14ajqXghMf8p37rzVDr/+k6XhTFolNYdz612ST1PEp1qtP/ncofnSCf
pmd4D/NngFzO2RkSBRfuG5IrggCjnXY/oLEiqRCijdjB+uL9ZjxATq5vikQ
mLIUj4I4m1T+GX6TRzuKIvex5VSJw1TL4x8UyjVkMBZVKjqYEPYj8ryBtSa
4qNiHlqi2OXificRBPMdbilPA73/jgwtL/Krd8+FKt1LADGzoF/K2D7fIGF
iGwby5HhbLd+nK37Te2NQ8ONtVaCQVVGjSNNdY2ikIfLZxyzUg9X/lQ9R/g
yFMoiy6wWiJ+hDK72LQKJlZFdJ2vRg1WEv9hwIFTAE1L1wE7FufMUR/Ox9T
uTa45yBA7MQLLZTR7zPNJgPLf/2BBGh6VTfx5oipYfOjgUf7mly4wUF5mFS
pxhs2KheOenvkQ== buenvenida@manageNode
```



You have successfully added the key 'CPE232'.

KBDBuenvenida (KBDBuenvenida)
Your personal account

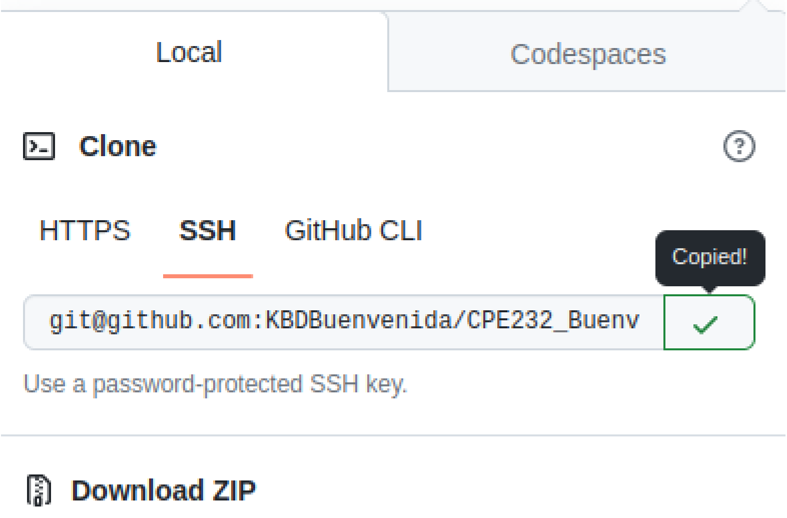
SSH keys

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

Authentication Keys

CPE232
SHA256:OyRoZdUTKHW1epSU4VwgnAkaKcoOakW9IJwWK/T2ru0
Added on Aug 22, 2023
Never used — Read/write

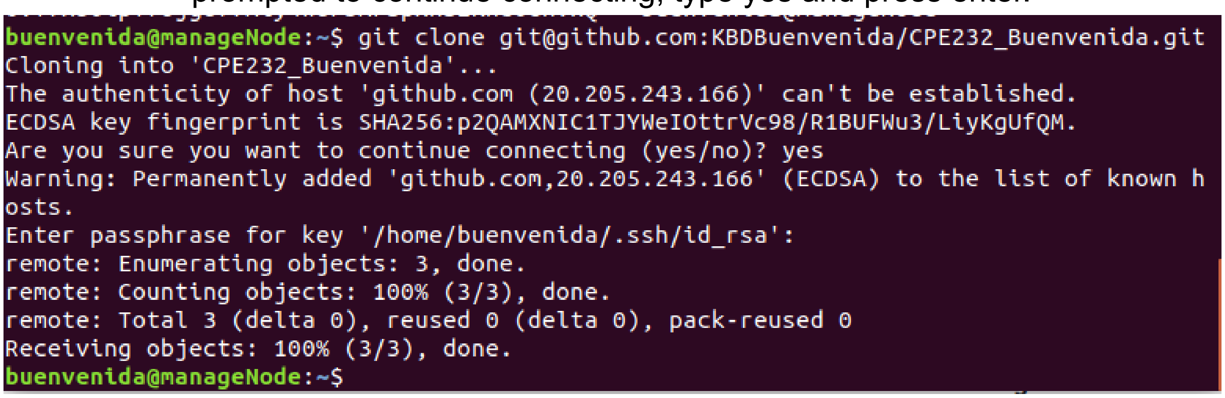
d. Clone the repository that you created. In doing this, you need to get the link from GitHub. Browse to your repository as shown below. Click on the Code drop down menu. Select SSH and copy the link.



```
git@github.com:KBDBuenvenida/CPE232_Buenv
```

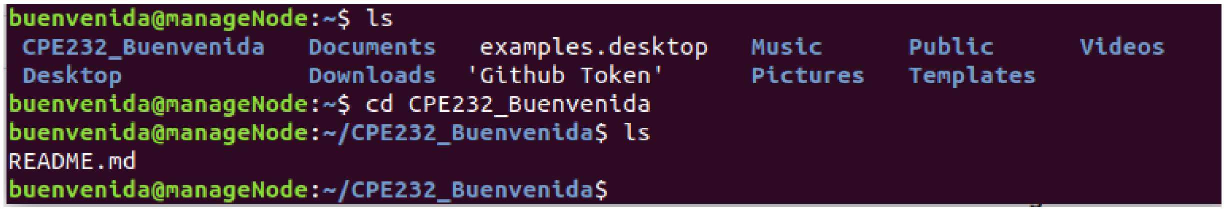
Use a password-protected SSH key.

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.



```
buenvenida@manageNode:~$ git clone git@github.com:KBDBuenvenida/CPE232_Buenvenida.git
Cloning into 'CPE232_Buenvenida'...
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 h
osts.
Enter passphrase for key '/home/buenvenida/.ssh/id_rsa':
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.
buenvenida@manageNode:~$
```

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.

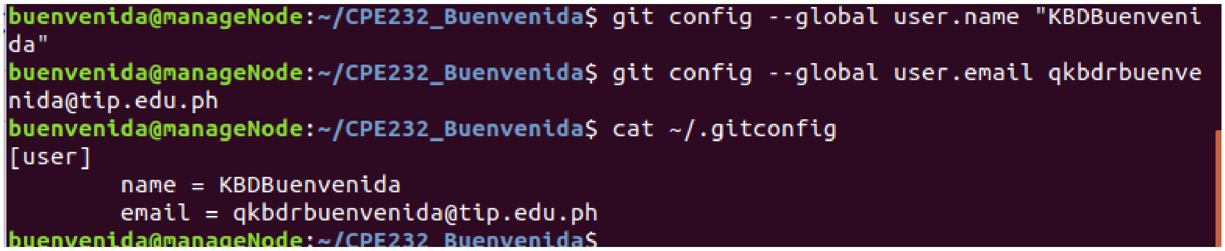


```
buenvenida@manageNode:~$ ls
 CPE232_Buenvenida    Documents    examples.desktop   Music      Public     Videos
 Desktop              Downloads   'Github Token'      Pictures   Templates
buenvenida@manageNode:~$ cd CPE232_Buenvenida
buenvenida@manageNode:~/CPE232_Buenvenida$ ls
README.md
buenvenida@manageNode:~/CPE232_Buenvenida$
```

g. Use the following commands to personalize your git.

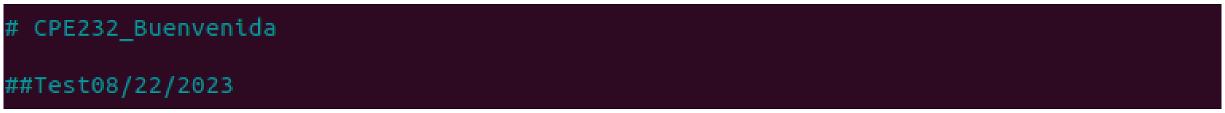
- *git config --global user.name “Your Name”*
- *git config --global user.email yourname@email.com*

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



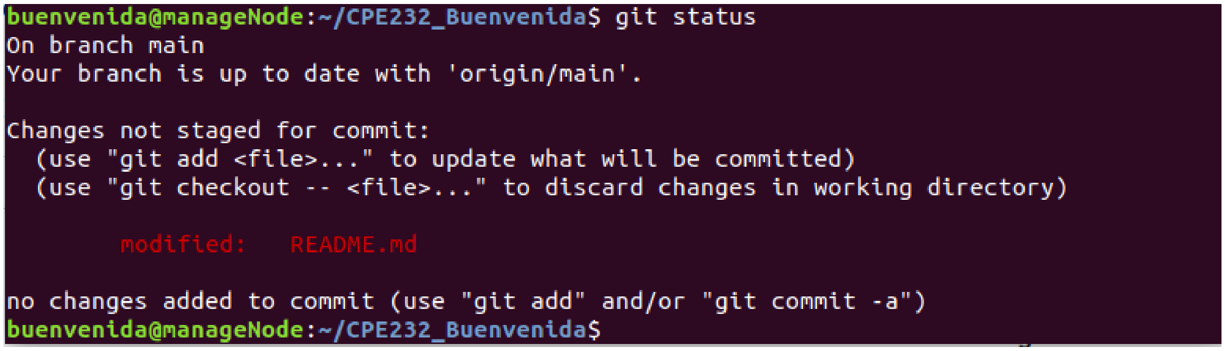
```
buenvenida@manageNode:~/CPE232_Buenvenida$ git config --global user.name "KBDBuenveni
da"
buenvenida@manageNode:~/CPE232_Buenvenida$ git config --global user.email qkbdrbuenve
nida@tip.edu.ph
buenvenida@manageNode:~/CPE232_Buenvenida$ cat ~/.gitconfig
[user]
        name = KBDBuenvenida
        email = qkbdrbuenvenida@tip.edu.ph
buenvenida@manageNode:~/CPE232_Buenvenida$
```

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.



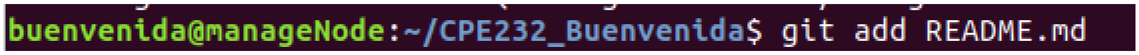
```
# CPE232_Buenvenida

##Test08/22/2023
```

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?



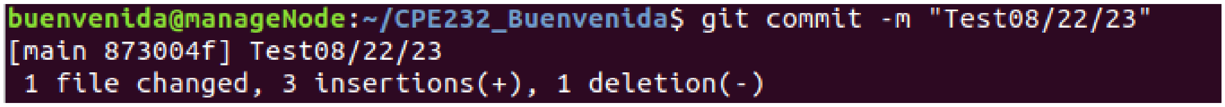
```
buenvenida@manageNode:~/CPE232_Buenvenida$ git status
On branch main
Your branch is up to date with 'origin/main'.

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")
buenvenida@manageNode:~/CPE232_Buenvenida$
```

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



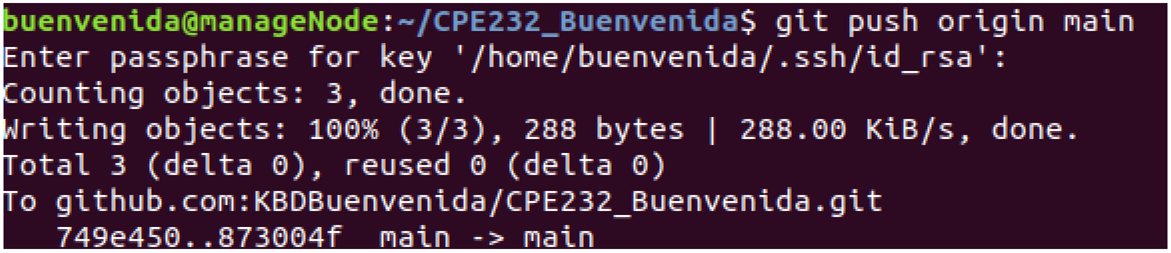
```
buenvenida@manageNode:~/CPE232_Buenvenida$ git add README.md
```

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.



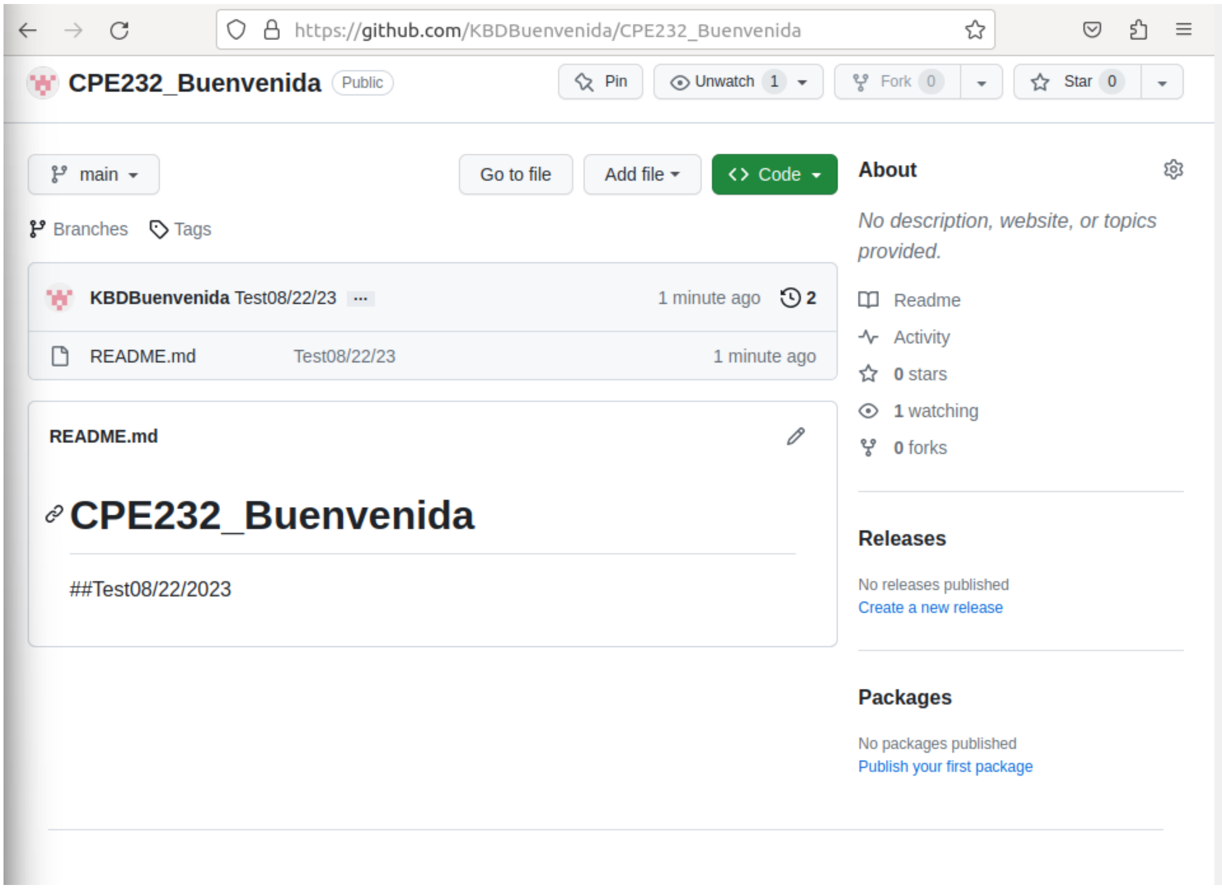
```
buenvenida@manageNode:~/CPE232_Buenvenida$ git commit -m "Test08/22/23"
[main 873004f] Test08/22/23
 1 file changed, 3 insertions(+), 1 deletion(-)
```

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.*



```
buenvenida@manageNode:~/CPE232_Buenvenida$ git push origin main
Enter passphrase for key '/home/buenvenida/.ssh/id_rsa':
Counting objects: 3, done.
Writing objects: 100% (3/3), 288 bytes | 288.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:KBDBuenvenida/CPE232_Buenvenida.git
   749e450..873004f  main -> main
```

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.



https://github.com/KBDBuenvenida/CPE232_Buenvenida

CPE232_Buenvenida Public

KBDBuenvenida Test08/22/23 ... 1 minute ago 2

README.md Test08/22/23 1 minute ago

README.md

CPE232_Buenvenida

##Test08/22/2023

About

No description, website, or topics provided.

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Reflections:
Answer the following:

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

Security and compliance: By setting firewalls, turning on/off services, and making sure servers adhere to strict security standards, Ansible may assist enforce security rules and compliance standards.

4. How important is the inventory file?

   Dynamic Inventory: Ansible is also capable of using dynamic inventory sources, which are scripts or plugins that create the inventory on the fly using a variety of data sources, such as APIs from cloud service providers or other inventory management programs.

Conclusions/Learnings:

In conclusion, setting up Git and using SSH key-based authentication are crucial for improving security and facilitating version control for software development projects.

Password-based authentication can be replaced with SSH key-based authentication, which is both more convenient and safe. SSH keys reduce the need to send passwords over the network by using asymmetric encryption, lowering the possibility of password interception and unwanted access. Only those with the corresponding private key can access remote servers or services thanks to the private and public key pair.

Key-based authentication's benefits are increased when Git is configured with SSH. Git, a popular version control program, is dependent on safe communication between local and remote repositories. Developers can securely push and pull code updates without disclosing sensitive credentials by enabling Git to use SSH keys. By doing this, unauthorized access to repositories is avoided, and important source code is shielded from security threats.