Name: Erebete, Jan Kenneth F.	**Date Performed: 8/15/25**
Course/Section: CPE31S4	**Date Submitted: 8/15/25**
Instructor: Engr. Robin Valenzuela	**Semester and SY: 2025 - 2026**

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

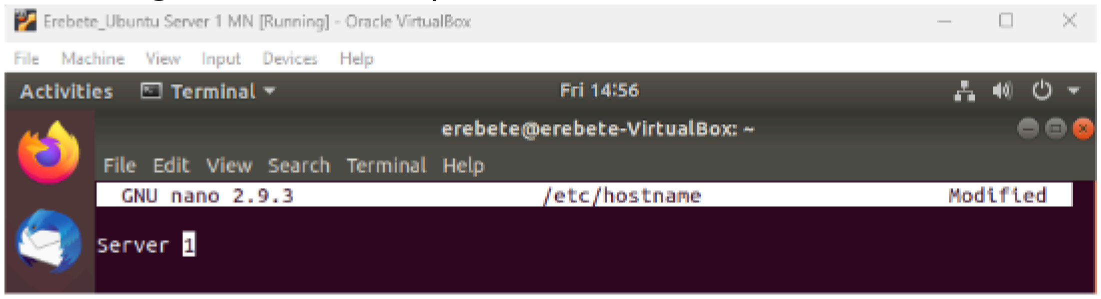
1.3 Verify connectivity

1.4 Setup Git Repository using local and remote repositories

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

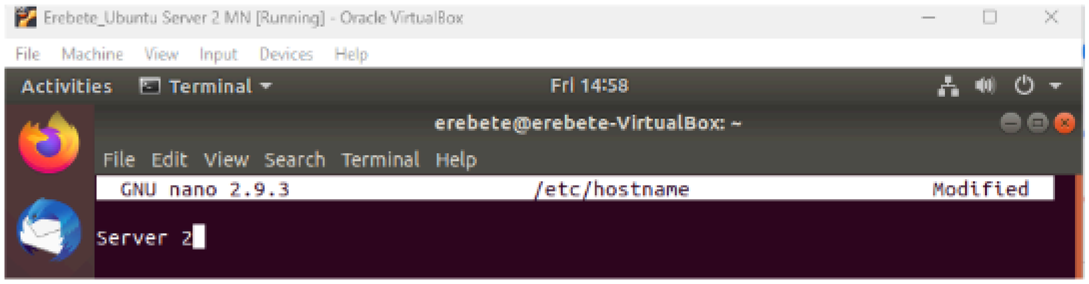
Part 1: Discussion

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


Erebete_Ubuntu Server 1 MN [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Activities Terminal Fri 14:56
erebete@erebete-VirtualBox: ~
File Edit View Search Terminal Help

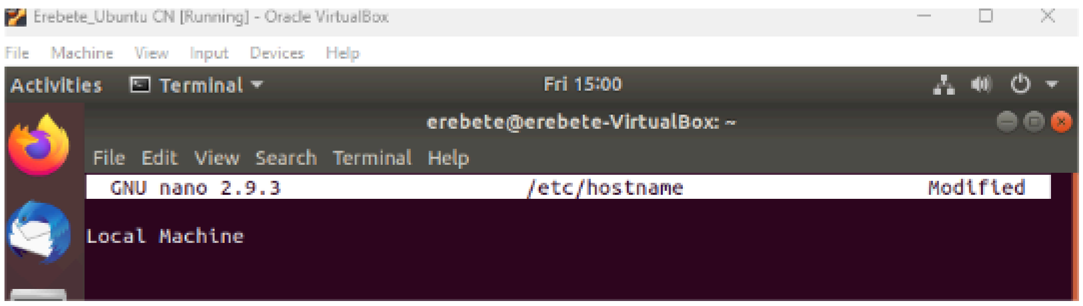
```
GNU nano 2.9.3                /etc/hostname                Modified

Server 1
```

1.2 Use server2 for Server 2


Erebete_Ubuntu Server 2 MN [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Activities Terminal Fri 14:58
erebete@erebete-VirtualBox: ~
File Edit View Search Terminal Help

```
GNU nano 2.9.3                /etc/hostname                Modified

Server 2
```

1.3 Use workstation for the Local Machine


Erebete_Ubuntu CN [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Activities Terminal Fri 15:00
erebete@erebete-VirtualBox: ~
File Edit View Search Terminal Help

```
GNU nano 2.9.3                /etc/hostname                Modified

Local Machine
```

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

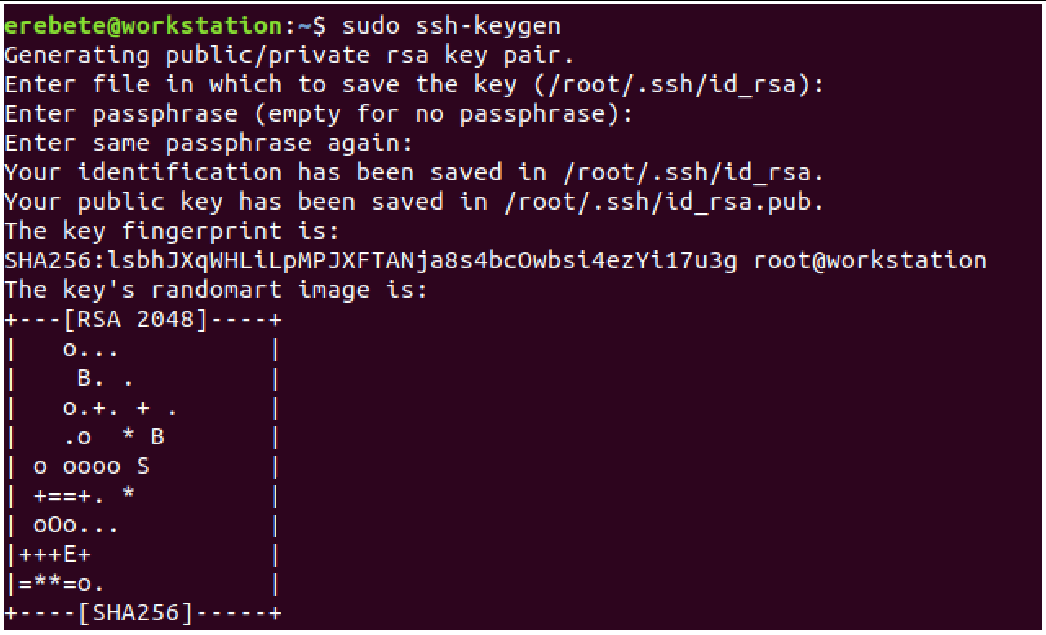
The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.

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.

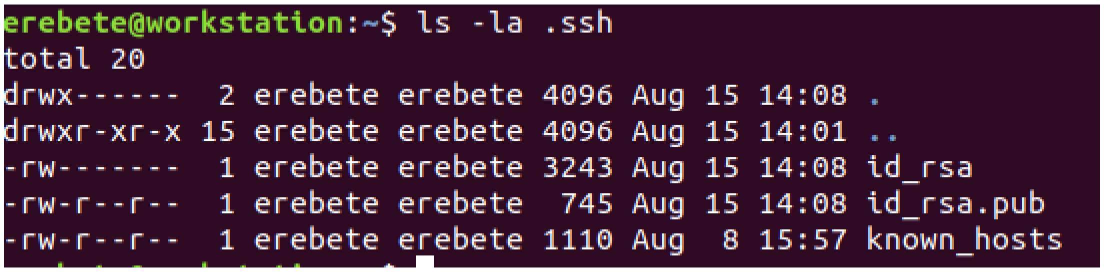
Task 1: Create an SSH Key Pair for User Authentication

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*.



```
erebete@workstation:~$ sudo ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:lsbhJXqWHLiLpMPJXFTANja8s4bcOwbsi4ezYi17u3g root@workstation
The key's randomart image is:
+---[RSA 2048]----+
|   o...          |
|    B. .         |
|   o.+. + .      |
|   .o  * B       |
| o oooo S        |
| +==+. *         |
| oOo...          |
|+++E+            |
|=**=o.           |
+----[SHA256]-----+
```

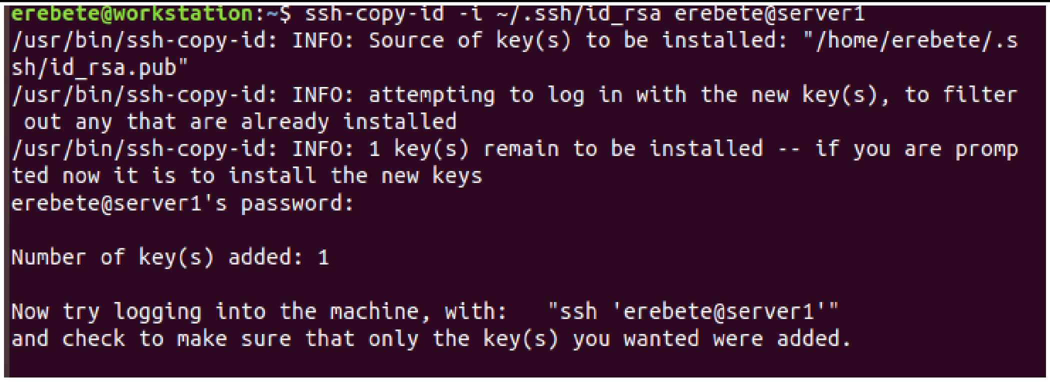
2. Issue the command *ssh-keygen -t rsa -b 4096.* The algorithm is selected using the -t option and key size using the -b option.
3. When asked for a passphrase, just press enter. The passphrase is used for encrypting the key, so that it cannot be used even if someone obtains the private key file. The 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.



```
erebete@workstation:~$ ls -la .ssh
total 20
drwx------  2 erebete erebete 4096 Aug 15 14:08 .
drwxr-xr-x 15 erebete erebete 4096 Aug 15 14:01 ..
-rw-------  1 erebete erebete 3243 Aug 15 14:08 id_rsa
-rw-r--r--  1 erebete erebete  745 Aug 15 14:08 id_rsa.pub
-rw-r--r--  1 erebete erebete 1110 Aug  8 15:57 known_hosts
```

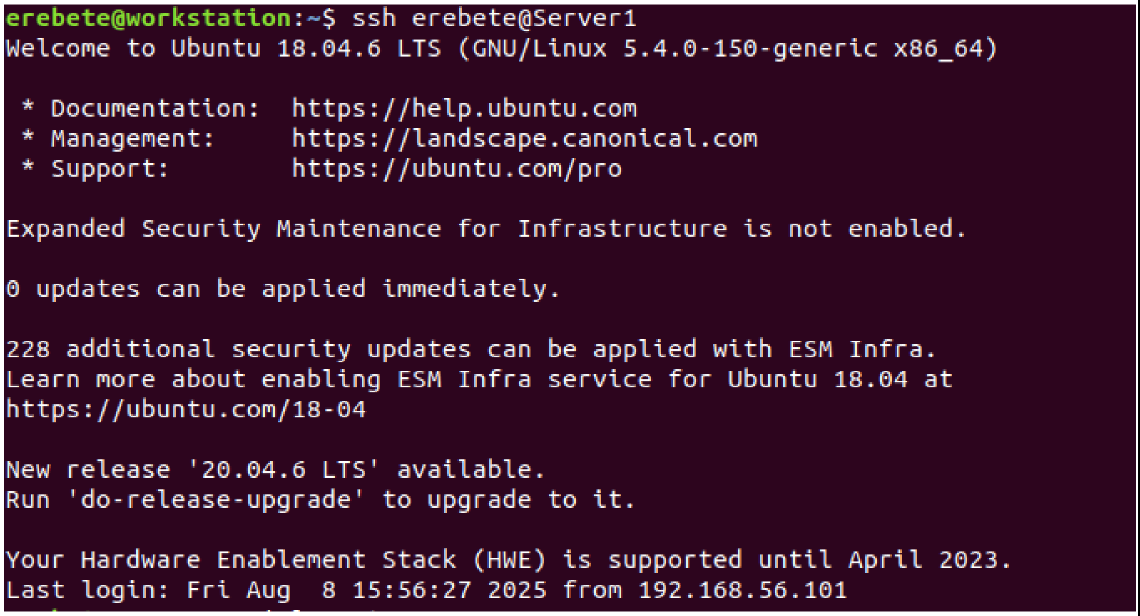
Task 2: Copying the Public Key to the remote servers

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



```
erebete@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa erebete@server1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/erebete/.s
sh/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
erebete@server1's password:

Number of key(s) added: 1

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



```
erebete@workstation:~$ ssh erebete@Server1
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

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

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

228 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04

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: Fri Aug  8 15:56:27 2025 from 192.168.56.101
```

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?

    - If you copied right the public key it won't ask for a password but if you didn't it will ask because the server authenticates you using your private key locally matching the public key installed in the server's ~/.ssh/authorized_keys file. This makes authentication automatic and secure without password prompts.

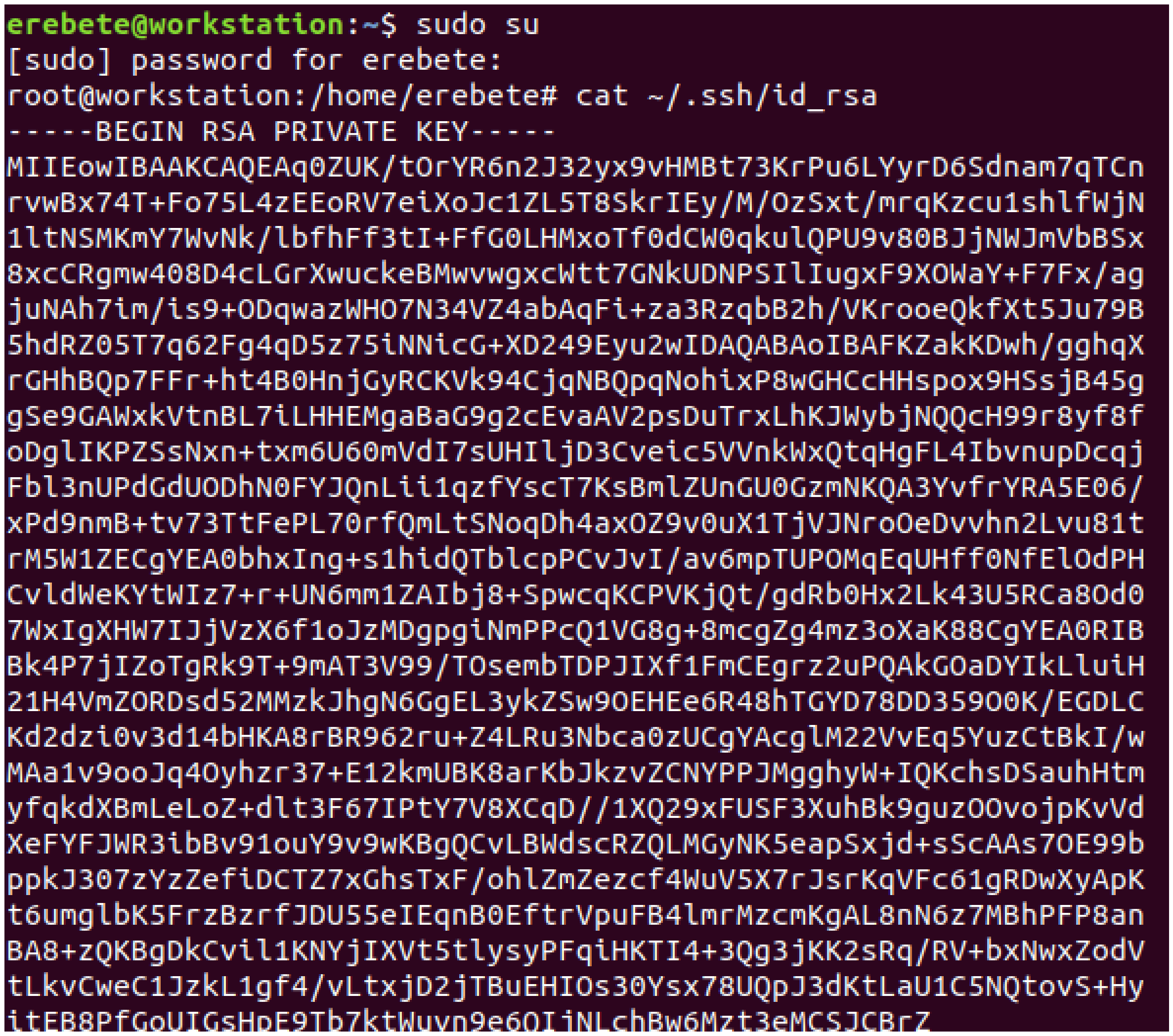
Reflections:

Answer the following:

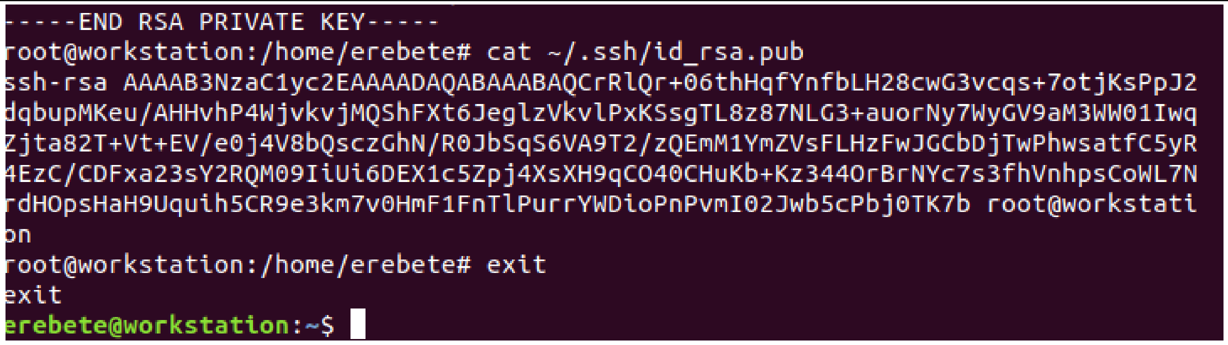
1. How will you describe the ssh-program? What does it do?
    - The SSH is a secure tool that lets you remotely connect and control another computer safely, even over insecure networks.

2. How do you know that you already installed the public key to the remote servers?
    - By using the SSH into the remote server If your public key is correctly installed on the remote server and matches your private key locally, you should log in without being prompted for a password.

Part 2: Discussion



```
erebete@workstation:~$ sudo su
[sudo] password for erebete:
root@workstation:/home/erebete# cat ~/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAq0ZUK/tOrYR6n2J32yx9vHMBt73KrPu6LYyrD6Sdnam7qTCn
rvwBx74T+Fo75L4zEEoRV7eiXoJc1ZL5T8SkrIEy/M/OzSxt/mrqKzcu1shlfWjN
1ltNSMKmY7WvNk/lbfhFf3tI+FfG0LHMxoTf0dCW0qkulQPU9v80BJjNWJmVbBSx
8xcCRgmw408D4cLGrXwuckeBMwvwgxcWtt7GNkUDNPSIlIugxF9XOWaY+F7Fx/ag
juNAh7im/is9+ODqwazWHO7N34VZ4abAqFi+za3RzqbB2h/VKrooeQkfXt5Ju79B
5hdRZ05T7q62Fg4qD5z75iNNicG+XD249Eyu2wIDAQABAoIBAFKZakKDwh/gghqX
rGHhBQp7FFr+ht4B0HnjGyRCKVk94CjqNBQpqNohixP8wGHCcHHspox9HSsjB45g
gSe9GAWxkVtnBL7iLHHEMgaBaG9g2cEvaAV2psDuTrxLhKJWybjNQQcH99r8yf8f
oDglIKPZSsNxn+txm6U60mVdI7sUHIljD3Cveic5VVnkWxQtqHgFL4IbvnupDcqj
Fbl3nUPdGdUODhN0FYJQnLii1qzfYscT7KsBmlZUnGU0GzmNKQA3YvfrYRA5E06/
xPd9nmB+tv73TtFePL70rfQmLtSNoqDh4axOZ9v0uX1TjVJNroOeDvvhn2Lvu81t
rM5W1ZECgYEA0bhxIng+s1hidQTblcpPCvJvI/av6mpTUPOMqEqUHff0NfElOdPH
CvldWeKYtWIz7+r+UN6mm1ZAIbj8+SpwcqKCPVKjQt/gdRb0Hx2Lk43U5RCa8Od0
7WxIgXHW7IJjVzX6f1oJzMDgpgiNmPPcQ1VG8g+8mcgZg4mz3oXaK88CgYEA0RIB
Bk4P7jIZoTgRk9T+9mAT3V99/TOsembTDPJIXf1FmCEgrz2uPQAkGOaDYIkLluiH
21H4VmZORDsd52MMzkJhgN6GgEL3ykZSw9OEHEe6R48hTGYD78DD359O0K/EGDLC
Kd2dzi0v3d14bHKA8rBR962ru+Z4LRu3Nbca0zUCgYAcglM22VvEq5YuzCtBkI/w
MAa1v9ooJq4Oyhzr37+E12kmUBK8arKbJkzvZCNYPPJMgghyW+IQKchsDSauhHtm
yfqkdXBmLeLoZ+dlt3F67IPtY7V8XCqD//1XQ29xFUSF3XuhBk9guzOOvojpKvVd
XeFYFJWR3ibBv91ouY9v9wKBgQCvLBWdscRZQLMGyNK5eapSxjd+sScAAs7OE99b
ppkJ307zYzZefiDCTZ7xGhsTxF/ohlZmZezcf4WuV5X7rJsrKqVFc61gRDwXyApK
t6umglbK5FrzBzrfJDU55eIEqnB0EftrVpuFB4lmrMzcmKgAL8nN6z7MBhPFP8an
BA8+zQKBgDkCvil1KNYjIXVt5tlysyPFqiHKTI4+3Qg3jKK2sRq/RV+bxNwxZodV
tLkvCweC1JzkL1gf4/vLtxjD2jTBuEHIOs30Ysx78UQpJ3dKtLaU1C5NQtovS+Hy
itEB8PfGoUIGsHpE9Tb7ktWuvn9e6OIjNLchBw6Mzt3eMCSJCBrZ
```



```
-----END RSA PRIVATE KEY-----
root@workstation:/home/erebete# cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCrRlQr+06thHqfYnfbLH28cwG3vcqs+7otjKsPpJ2
dqbupMKeu/AHHvhP4WjvkvjMQShFXt6JeglzVkvlPxKSsgTL8z87NLG3+auorNy7WyGV9aM3WW01Iwq
Zjta82T+Vt+EV/e0j4V8bQsczGhN/R0JbSqS6VA9T2/zQEmM1YmZVsFLHzFwJGCbDjTwPhwsatfC5yR
4EzC/CDFxa23sY2RQM09IiUi6DEX1c5Zpj4XsXH9qCO40CHuKb+Kz344OrBrNYc7s3fhVnhpsCoWL7N
rdHOpsHaH9Uquih5CR9e3km7v0HmF1FnTlPurrYWDioPnPvmI02Jwb5cPbj0TK7b root@workstati
on
root@workstation:/home/erebete# exit
exit
erebete@workstation:~$
```

Provide screenshots for each task.

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

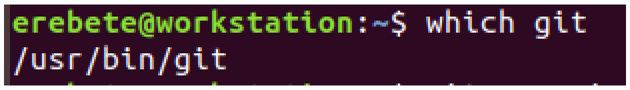
Set up Git

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:

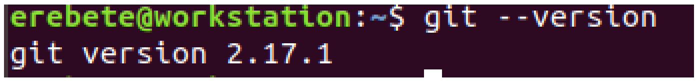
- Creating a repository
- Forking a repository
- Managing files
- Being social

Task 3: Set up the Git Repository

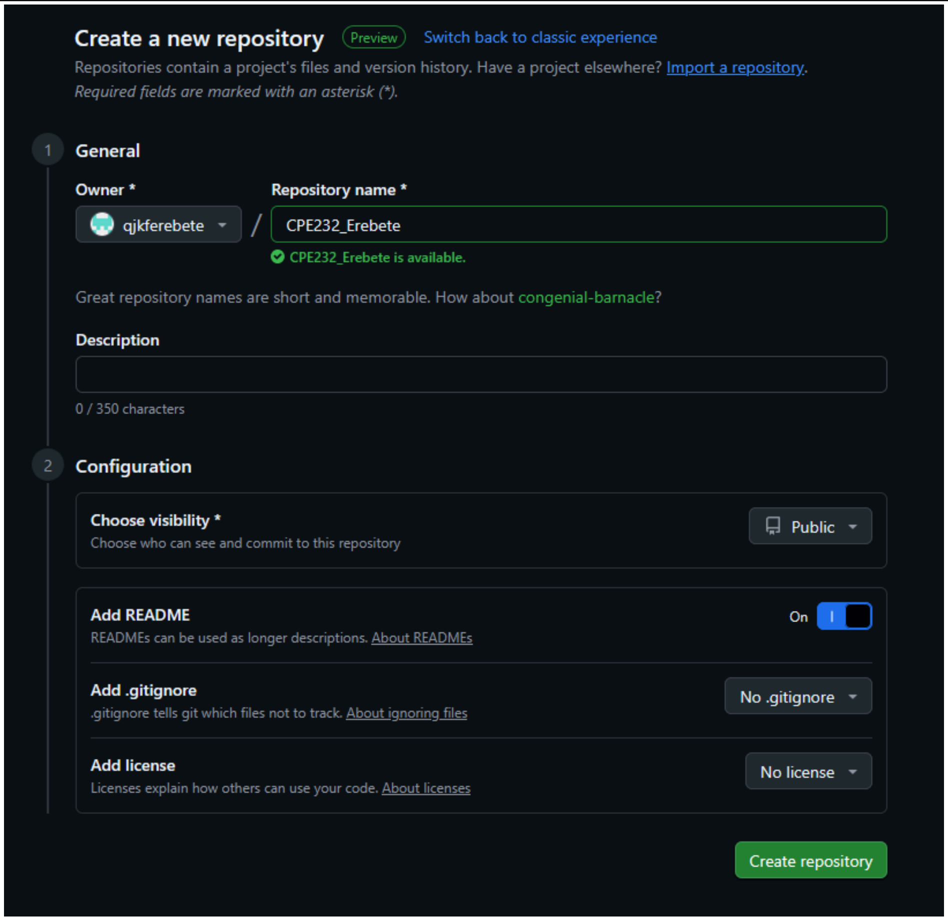
1. On the local machine, verify the version of your git using the command *which git.* If a directory of git is displayed, then you don't need to install git. Otherwise, to install git, use the following command: *sudo apt install git*
2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/bin/git.*

   ```
   erebete@workstation:~$ 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.

   ```
   erebete@workstation:~$ git --version
   git version 2.17.1
   ```
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

Repositories contain a project's files and version history. Have a project elsewhere? Import a repository.
Required fields are marked with an asterisk (*).

1 General

Owner *: qjkferebete / Repository name *: CPE232_Erebete

CPE232_Erebete is available.

Great repository names are short and memorable. How about congenial-barnacle?

Description

0 / 350 characters

2 Configuration

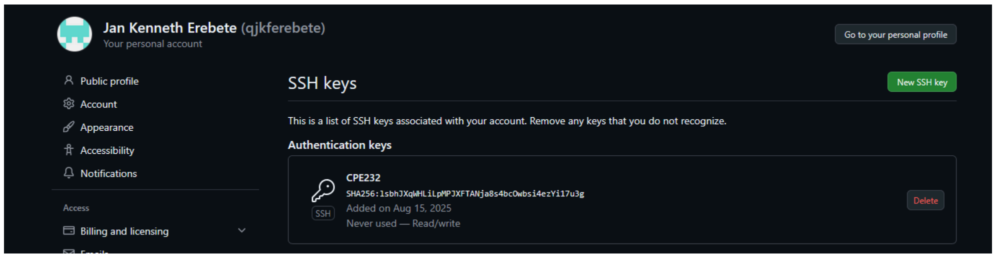
Choose visibility *
Choose who can see and commit to this repository
Public

Add README
READMEs can be used as longer descriptions. About READMEs
On

Add .gitignore
.gitignore tells git which files not to track. About ignoring files
No .gitignore

Add license
Licenses explain how others can use your code. About licenses
No license

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.
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.


Jan Kenneth Erebete (qjkferebete)
Your personal account

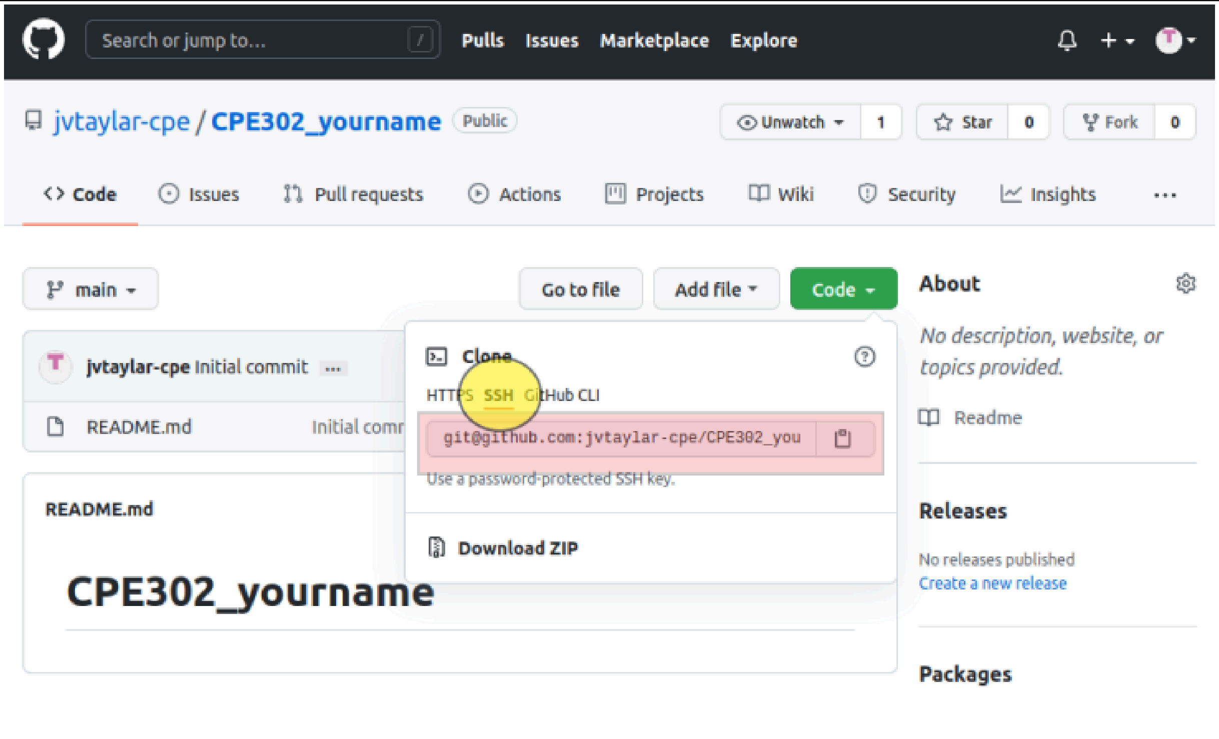
SSH keys

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

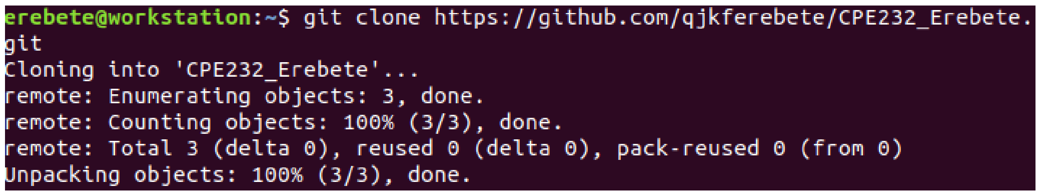
Authentication keys

CPE232
SHA256:lsbhJXqWHLiLpMPJXFTANja8s4bcOwbsi4ezYi17u3g
Added on Aug 15, 2025
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.

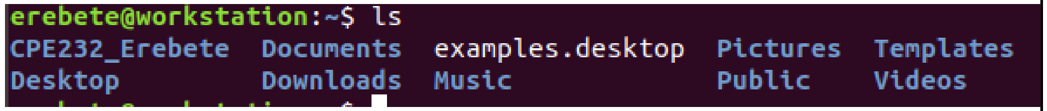


e. Issue the command git clone followed by the copied link. For example, *git clone git@github.com:jvtaylar-cpe/CPE232_yourname.git*. When prompted to continue connecting, type yes and press enter.



```
erebete@workstation:~$ git clone https://github.com/qjkferebete/CPE232_Erebete.
git
Cloning into 'CPE232_Erebete'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), done.
```

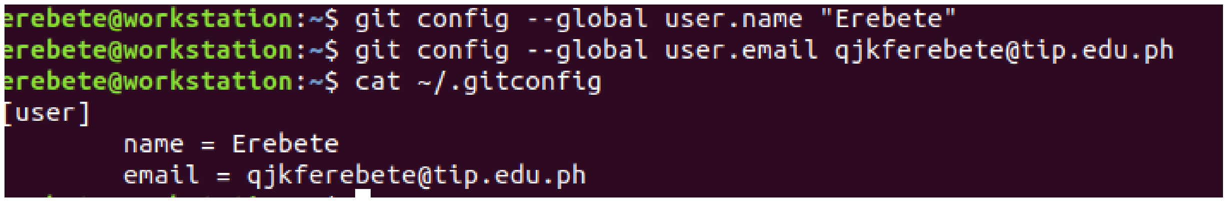
f. To verify that you have cloned the GitHub repository, issue the command *ls*. Observe that you have the CPE232_yourname in the list of your directories. Use CD command to go to that directory and LS command to see the file README.md.



```
erebete@workstation:~$ ls
CPE232_Erebete  Documents  examples.desktop  Pictures  Templates
Desktop         Downloads  Music             Public    Videos
```

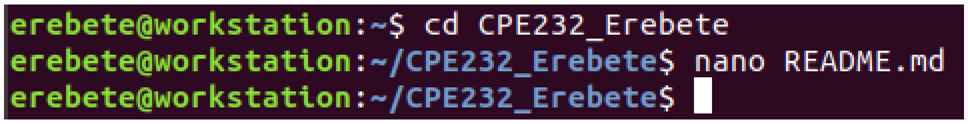
g. Use the following commands to personalize your git.

- *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*

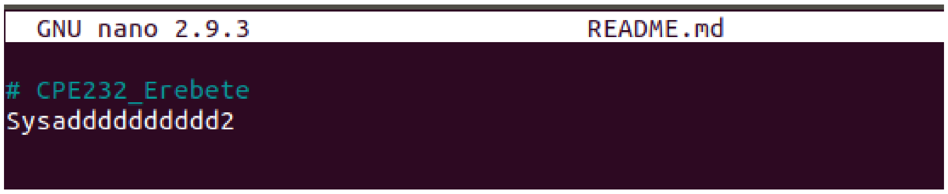


```
erebete@workstation:~$ git config --global user.name "Erebete"
erebete@workstation:~$ git config --global user.email qjkferebete@tip.edu.ph
erebete@workstation:~$ cat ~/.gitconfig
[user]
        name = Erebete
        email = qjkferebete@tip.edu.ph
```

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.

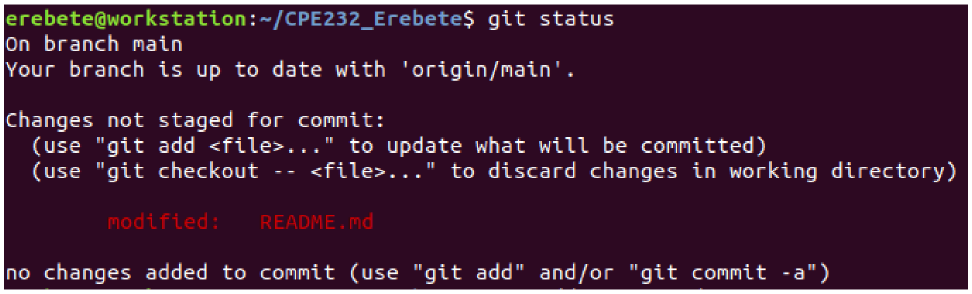


```
erebete@workstation:~$ cd CPE232_Erebete
erebete@workstation:~/CPE232_Erebete$ nano README.md
erebete@workstation:~/CPE232_Erebete$
```



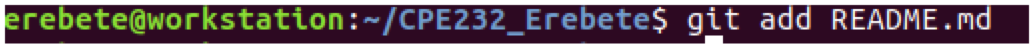
```
  GNU nano 2.9.3                        README.md

# CPE232_Erebete
Sysaddddddddddd2
```

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?



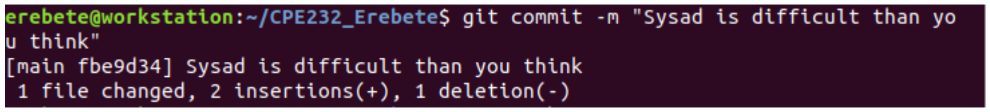
```
erebete@workstation:~/CPE232_Erebete$ git status
On branch main
Your branch is up to date with 'origin/main'.

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")
```

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



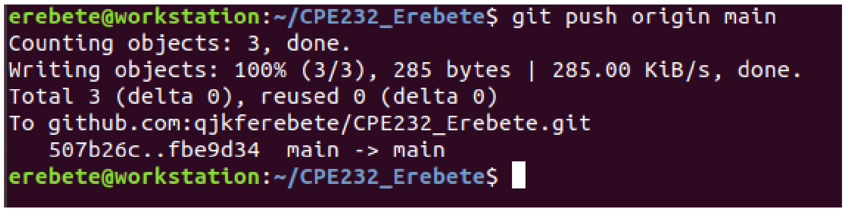
```
erebete@workstation:~/CPE232_Erebete$ git add README.md
```

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.



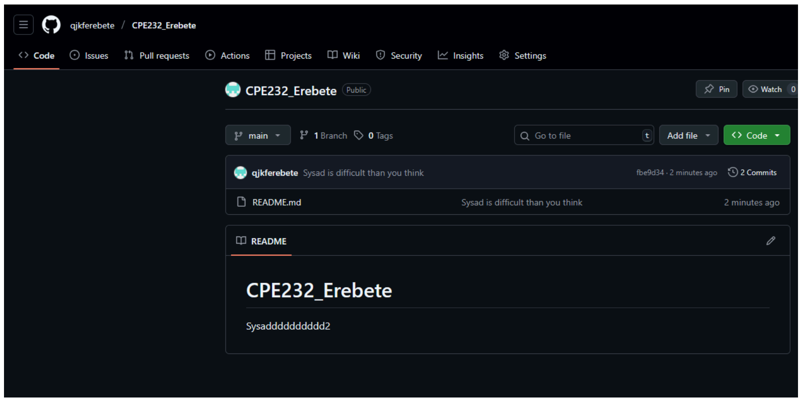
```
erebete@workstation:~/CPE232_Erebete$ git commit -m "Sysad is difficult than yo
u think"
[main fbe9d34] Sysad is difficult than you think
 1 file changed, 2 insertions(+), 1 deletion(-)
```

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.*



```
erebete@workstation:~/CPE232_Erebete$ git push origin main
Counting objects: 3, done.
Writing objects: 100% (3/3), 285 bytes | 285.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:qjkferebete/CPE232_Erebete.git
   507b26c..fbe9d34  main -> main
erebete@workstation:~/CPE232_Erebete$ 
```

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.



CPE232_Erebete Public

1 Branch 0 Tags

qjkferebete Sysad is difficult than you think fbe9d34 · 2 minutes ago 2 Commits

README.md Sysad is difficult than you think 2 minutes ago

README

CPE232_Erebete

Sysaddddddddd2

Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
   - I just followed the instruction and sometimes not familiar on the command that have been using and try to figure out how they works
4. How important is the inventory file?
   - For me it is important because if you get confused or forget something you can easily find the file that you created before that's why it is important to us.

Conclusions/Learnings:

- **Overall it was fun and stressful because of some new commands that im not familiar with and figured out how they works as a command and how do we used them.**