Name: Valencia, Mark Janssen M.	**Date Performed:** August 24, 2023
Course/Section: CPE32s6	**Date Submitted:** August 24, 2023
Instructor: Dr. Jonathan Taylar	**Semester and SY:** 2nd 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

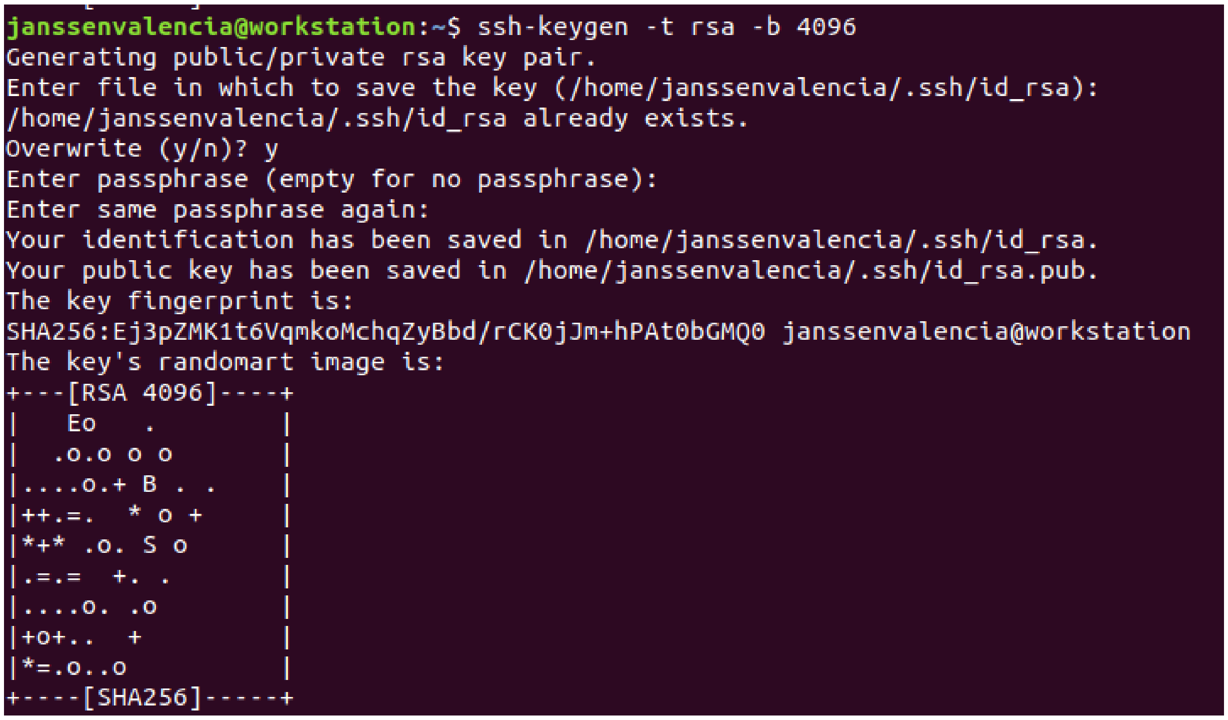
The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.

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.

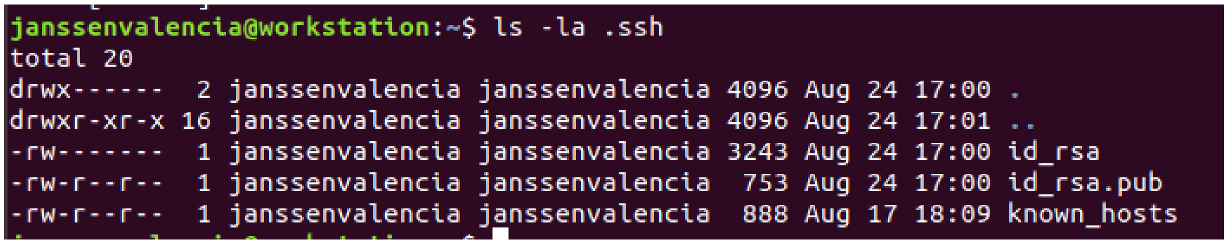
Task 1: Create an SSH Key Pair for User Authentication

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*.
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.



```
janssenvalencia@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/janssenvalencia/.ssh/id_rsa):
/home/janssenvalencia/.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/janssenvalencia/.ssh/id_rsa.
Your public key has been saved in /home/janssenvalencia/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Ej3pZMK1t6VqmkoMchqZyBbd/rCK0jJm+hPAt0bGMQ0 janssenvalencia@workstation
The key's randomart image is:
+---[RSA 4096]----+
|    Eo    .      |
|   .o.o o o      |
|....o.+ B . .    |
|++.=.   * o +    |
|*+* .o. S o      |
|.=.=  +. .       |
|....o. .o        |
|+o+..   +        |
|*=.o..o          |
+----[SHA256]-----+
```

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.

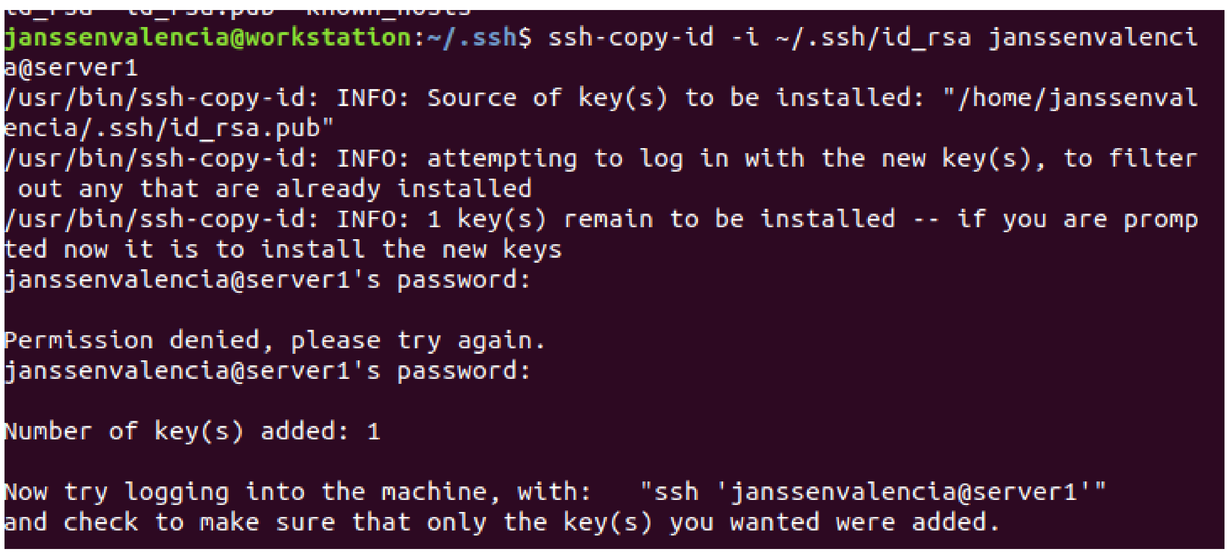


```
janssenvalencia@workstation:~$ ls -la .ssh
total 20
drwx------  2 janssenvalencia janssenvalencia 4096 Aug 24 17:00 .
drwxr-xr-x 16 janssenvalencia janssenvalencia 4096 Aug 24 17:01 ..
-rw-------  1 janssenvalencia janssenvalencia 3243 Aug 24 17:00 id_rsa
-rw-r--r--  1 janssenvalencia janssenvalencia  753 Aug 24 17:00 id_rsa.pub
-rw-r--r--  1 janssenvalencia janssenvalencia  888 Aug 17 18:09 known_hosts
```

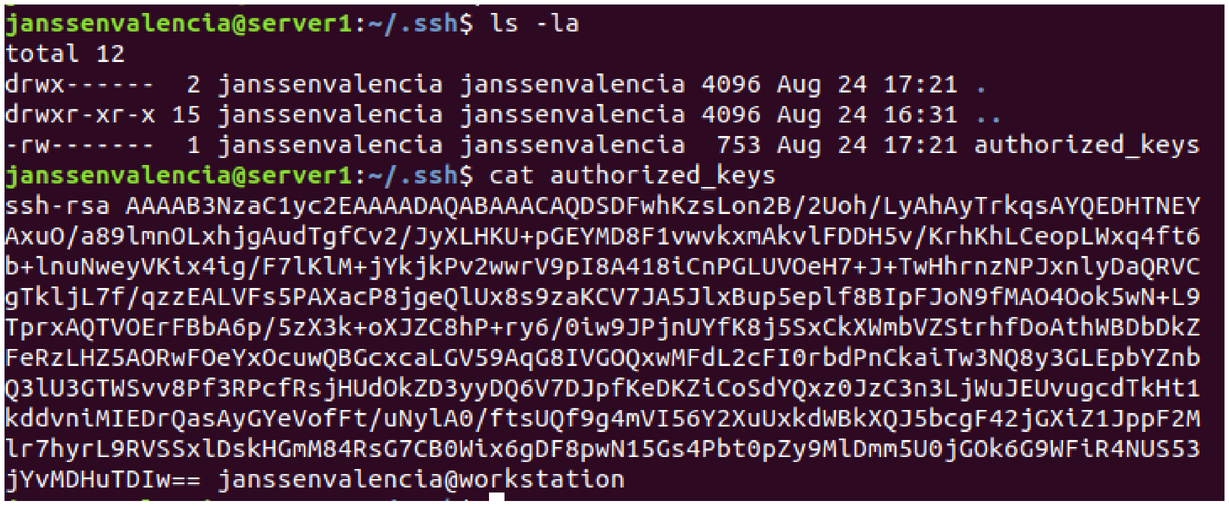
Task 2: Copying the Public Key to the remote servers

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*
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.

Server 1

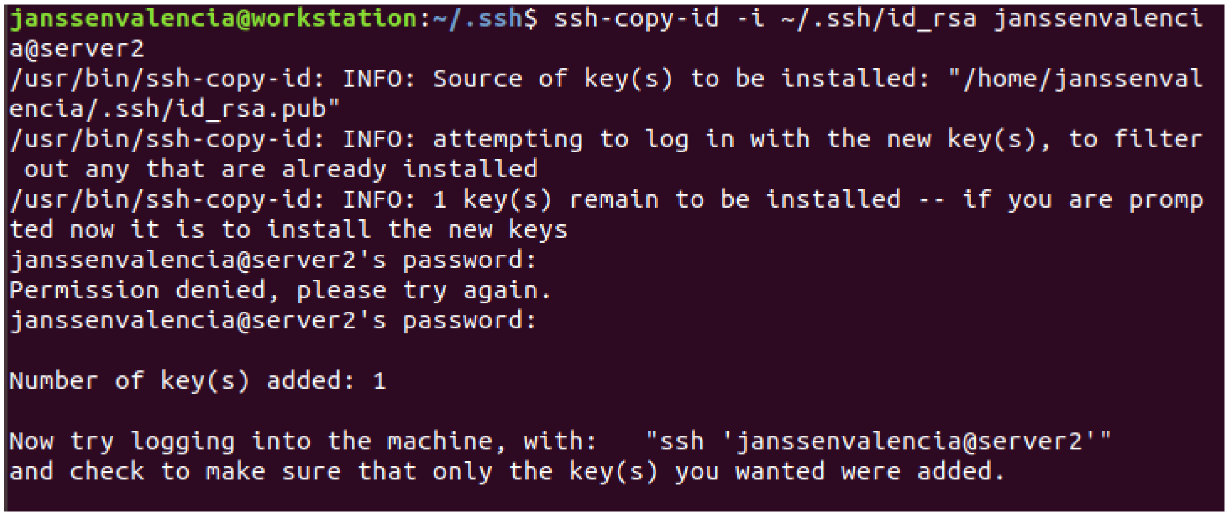


```
janssenvalencia@workstation:~/.ssh$ ssh-copy-id -i ~/.ssh/id_rsa janssenvalenci
a@server1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/janssenval
encia/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
 out any that are already 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
janssenvalencia@server1's password:

Permission denied, please try again.
janssenvalencia@server1's password:

Number of key(s) added: 1

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

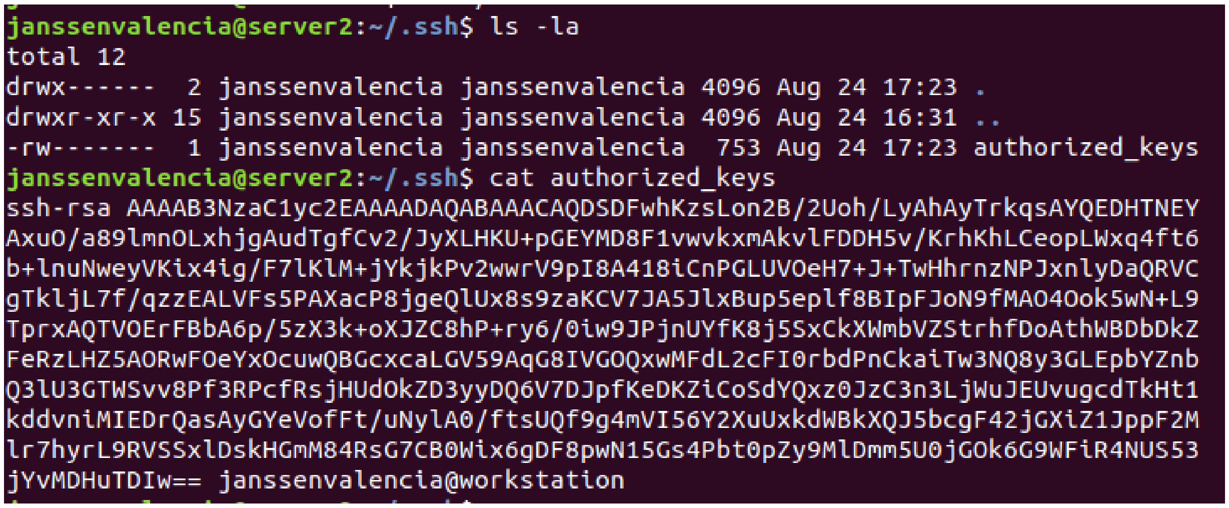


```
janssenvalencia@server1:~/.ssh$ ls -la
total 12
drwx------  2 janssenvalencia janssenvalencia 4096 Aug 24 17:21 .
drwxr-xr-x 15 janssenvalencia janssenvalencia 4096 Aug 24 16:31 ..
-rw-------  1 janssenvalencia janssenvalencia  753 Aug 24 17:21 authorized_keys
janssenvalencia@server1:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDSDFwhKzsLon2B/2Uoh/LyAhAyTrkqsAYQEDHTNEY
AxuO/a89lmnOLxhjgAudTgfCv2/JyXLHKU+pGEYMD8F1vwvkxmAkvlFDDH5v/KrhKhLCeopLWxq4ft6
b+lnuNweyVKix4ig/F7lKlM+jYkjkPv2wwrV9pI8A418iCnPGLUVOeH7+J+TwHhrnzNPJxnlyDaQRVC
gTkljL7f/qzzEALVFs5PAXacP8jgeQlUx8s9zaKCV7JA5JlxBup5eplf8BIpFJoN9fMAO4Ook5wN+L9
TprxAQTVOErFBbA6p/5zX3k+oXJZC8hP+ry6/0iw9JPjnUYfK8j5SxCkXWmbVZStrhfDoAthWBDbDkZ
FeRzLHZ5AORwFOeYxOcuwQBGcxcaLGV59AqG8IVGOQxwMFdL2cFI0rbdPnCkaiTw3NQ8y3GLEpbYZnb
Q3lU3GTWSvv8Pf3RPcfRsjHUdOkZD3yyDQ6V7DJpfKeDKZiCoSdYQxz0JzC3n3LjWuJEUvugcdTkHt1
kddvniMIEDrQasAyGYeVofFt/uNylA0/ftsUQf9g4mVI56Y2XuUxkdWBkXQJ5bcgF42jGXiZ1JppF2M
lr7hyrL9RVSSxlDskHGmM84RsG7CB0Wix6gDF8pwN15Gs4Pbt0pZy9MlDmm5U0jGOk6G9WFiR4NUS53
jYvMDHuTDIw== janssenvalencia@workstation
```

Server 2



```
janssenvalencia@workstation:~/.ssh$ ssh-copy-id -i ~/.ssh/id_rsa janssenvalenci
a@server2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/janssenval
encia/.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
janssenvalencia@server2's password:
Permission denied, please try again.
janssenvalencia@server2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'janssenvalencia@server2'"
and check to make sure that only the key(s) you wanted were added.
```



```
janssenvalencia@server2:~/.ssh$ ls -la
total 12
drwx------  2 janssenvalencia janssenvalencia 4096 Aug 24 17:23 .
drwxr-xr-x 15 janssenvalencia janssenvalencia 4096 Aug 24 16:31 ..
-rw-------  1 janssenvalencia janssenvalencia  753 Aug 24 17:23 authorized_keys
janssenvalencia@server2:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDSDFwhKzsLon2B/2Uoh/LyAhAyTrkqsAYQEDHTNEY
AxuO/a89lmnOLxhjgAudTgfCv2/JyXLHKU+pGEYMD8F1vwvkxmAkvlFDDH5v/KrhKhLCeopLWxq4ft6
b+lnuNweyVKix4ig/F7lKlM+jYkjkPv2wwrV9pI8A418iCnPGLUVOeH7+J+TwHhrnzNPJxnlyDaQRVC
gTkljL7f/qzzEALVFs5PAXacP8jgeQlUx8s9zaKCV7JA5JlxBup5eplf8BIpFJoN9fMAO4Ook5wN+L9
TprxAQTVOErFBbA6p/5zX3k+oXJZC8hP+ry6/0iw9JPjnUYfK8j5SxCkXWmbVZStrhfDoAthWBDbDkZ
FeRzLHZ5AORwFOeYxOcuwQBGcxcaLGV59AqG8IVGOQxwMFdL2cFI0rbdPnCkaiTw3NQ8y3GLEpbYZnb
Q3lU3GTWSvv8Pf3RPcfRsjHUdOkZD3yyDQ6V7DJpfKeDKZiCoSdYQxz0JzC3n3LjWuJEUvugcdTkHt1
kddvniMIEDrQasAyGYeVofFt/uNylA0/ftsUQf9g4mVI56Y2XuUxkdWBkXQJ5bcgF42jGXiZ1JppF2M
lr7hyrL9RVSSxlDskHGmM84RsG7CB0Wix6gDF8pwN15Gs4Pbt0pZy9MlDmm5U0jGOk6G9WFiR4NUS53
jYvMDHuTDIw== janssenvalencia@workstation
```

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?

    - **After copying the ssh keys on both server 1 and server 2, when connecting to both of those sever from the workstation it does not ask for a password however when using one of the servers and connecting to the other it does ask for a password since they do not have the key to the current machine that is in used to connect. Aside from this, I have used the workstation to access the .ssh location of both of the servers and check for the authorized_keys key.**

Reflections:
Answer the following:
1. How will you describe the ssh-program? What does it do?
   - **I found the SSH program useful since it allows you to connect to a specific machine remotely and execute programs without being physically near the machine itself Securely using an encrypted connection.**
2. How do you know that you already installed the public key to the remote servers?
   - **You will be notified when the key is copied after executing the copy command on the workstation, To double check if the public key is copied on the remote servers you can easily check the directory if they content the file "*Authorized_keys*"**

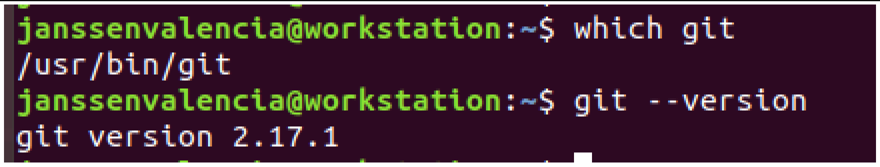
Part 2: Discussion

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

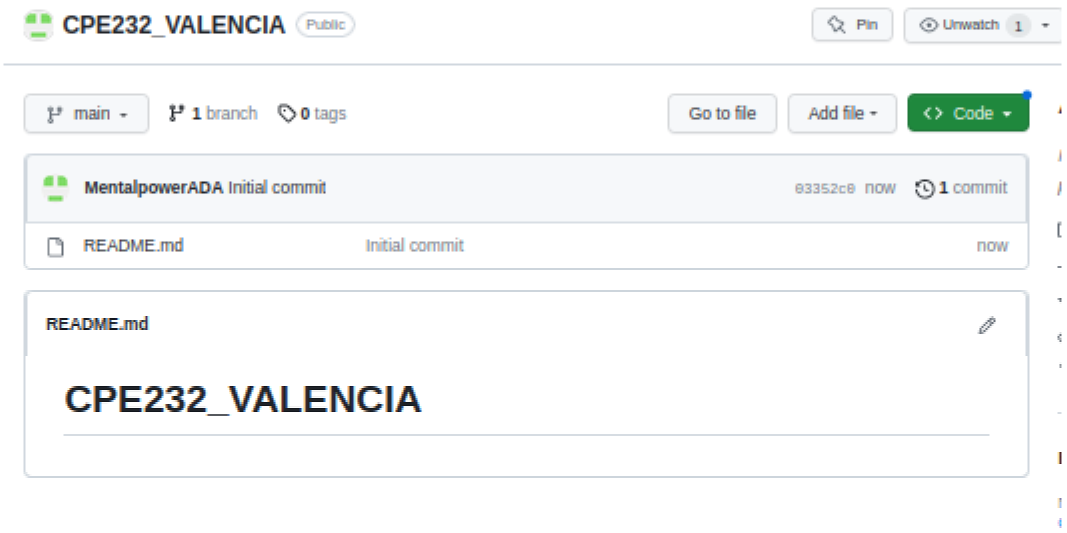
Task 3: Set up the Git Repository
1. On the local machine, verify the version of your git using the command *which git.* If a directory of git is displayed, then you don't need to install git. Otherwise, to install git, use the following command: *sudo apt install git*
2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/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.



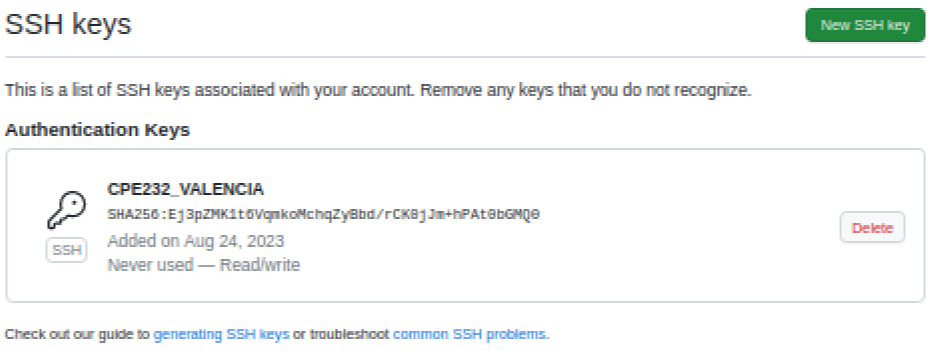
```
janssenvalencia@workstation:~$ which git
/usr/bin/git
janssenvalencia@workstation:~$ git --version
git version 2.17.1
```

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

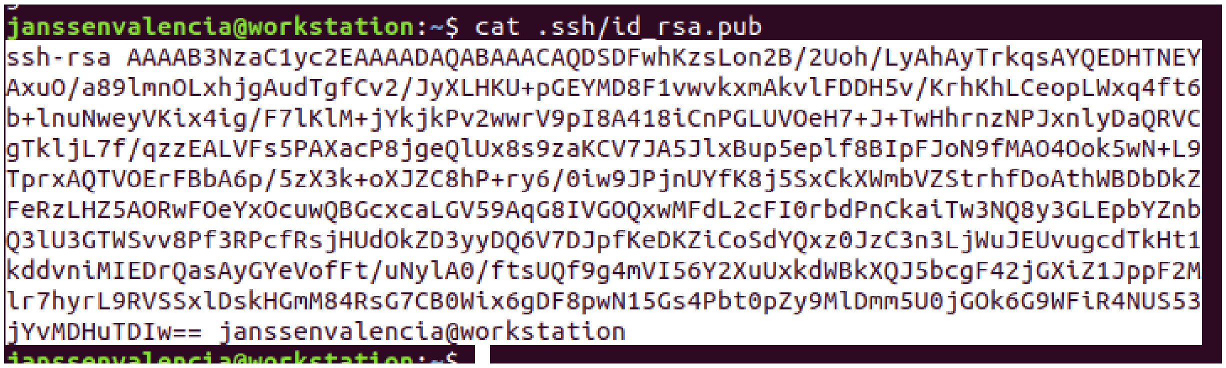
5. Sign up in case you don't have an account yet. Otherwise, login to your GitHub account.
   a. Create a new repository and name it as CPE232_yourname. Check Add a README file and click Create repository.



   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.

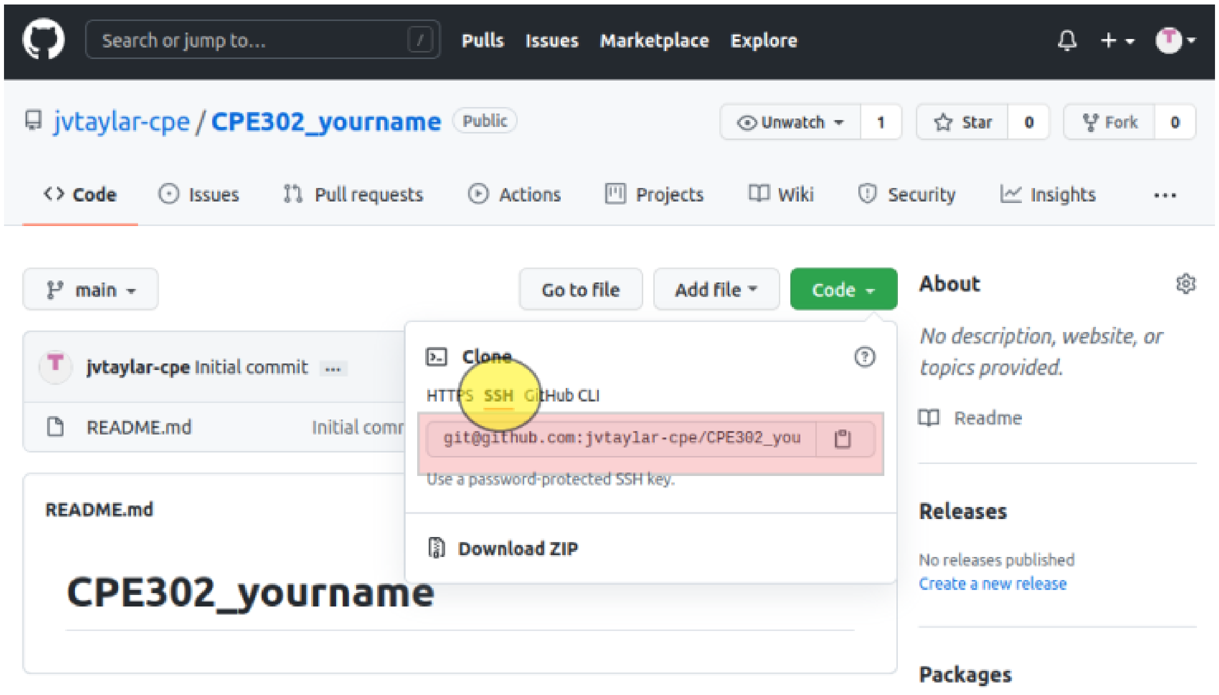

SSH keys
New SSH key
This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.
Authentication Keys
CPE232_VALENCIA
SHA256:Ej3pZMK1t6VqmkoMchqZyBbd/rCK8jJm+hPAt0bGMQ0
Added on Aug 24, 2023
Never used — Read/write
SSH
Delete
Check out our guide to generating SSH 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.



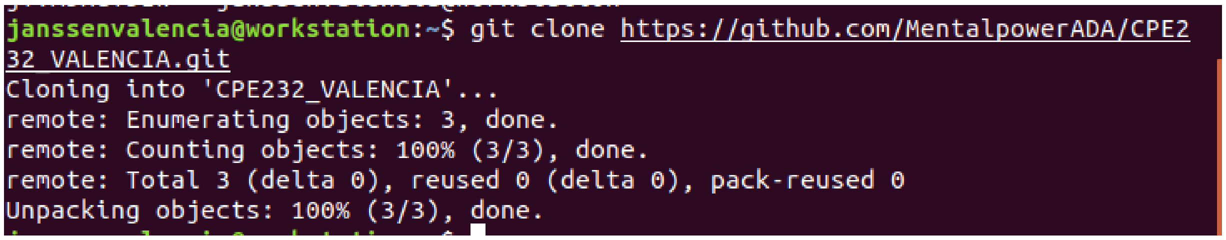
```
janssenvalencia@workstation:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDSDFwhKzsLon2B/2Uoh/LyAhAyTrkqsAYQEDHTNEY
AxuO/a89lmnOLxhjgAudTgfCv2/JyXLHKU+pGEYMD8F1vwvkxmAkvlFDDH5v/KrhKhLCeopLWxq4ft6
b+lnuNweyVKix4ig/F7lKlM+jYkjkPv2wwrV9pI8A418iCnPGLUVOeH7+J+TwHhrnzNPJxnlyDaQRVC
gTkljL7f/qzzEALVFs5PAXacP8jgeQlUx8s9zaKCV7JA5JlxBup5eplf8BIpFJoN9fMAO4Ook5wN+L9
TprxAQTVOErFBbA6p/5zX3k+oXJZC8hP+ry6/0iw9JPjnUYfK8j5SxCkXWmbVZStrhfDoAthWBDbDkZ
FeRzLHZ5AORwFOeYxOcuwQBGcxcaLGV59AqG8IVGOQxwMFdL2cFI0rbdPnCkaiTw3NQ8y3GLEpbYZnb
Q3lU3GTWSvv8Pf3RPcfRsjHUdOkZD3yyDQ6V7DJpfKeDKZiCoSdYQxz0JzC3n3LjWuJEUvugcdTkHt1
kddvniMIEDrQasAyGYeVofFt/uNylA0/ftsUQf9g4mVI56Y2XuUxkdWBkXQJ5bcgF42jGXiZ1JppF2M
lr7hyrL9RVSSxlDskHGmM84RsG7CB0Wix6gDF8pwN15Gs4Pbt0pZy9MlDmm5U0jGOk6G9WFiR4NUS53
jYvMDHuTDIw== janssenvalencia@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.



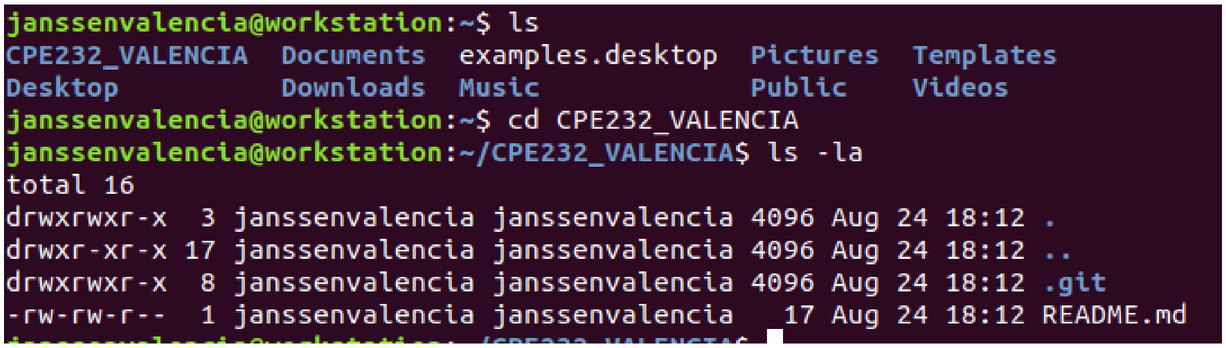
```
git@github.com:jvtaylar-cpe/CPE302_you
```

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.

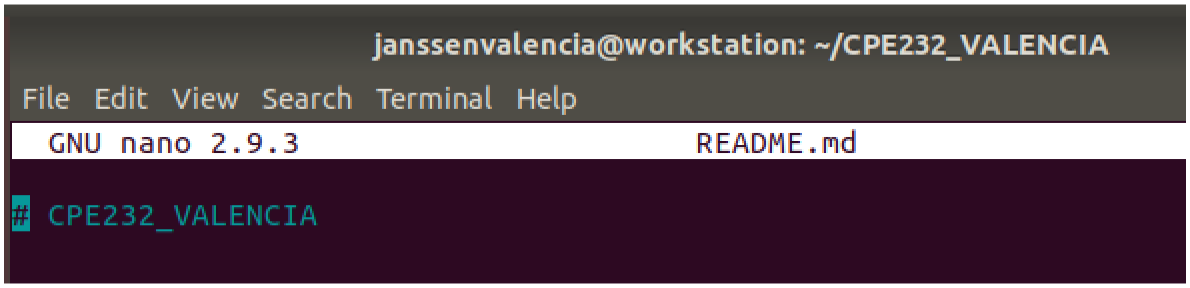


```
janssenvalencia@workstation:~$ git clone https://github.com/MentalpowerADA/CPE2
32_VALENCIA.git
Cloning into 'CPE232_VALENCIA'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (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.



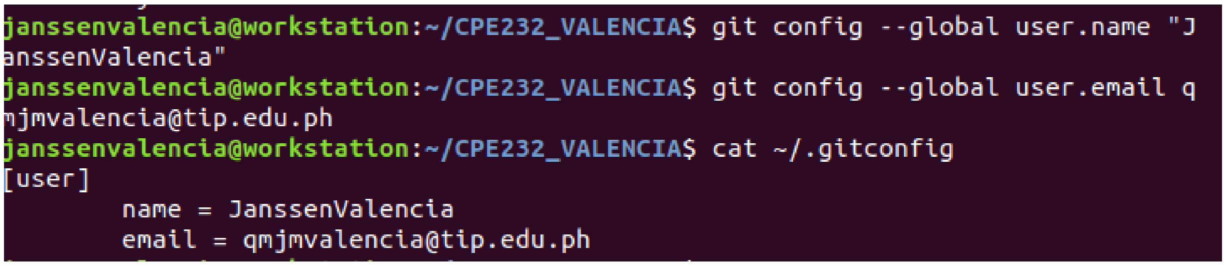
```
janssenvalencia@workstation:~$ ls
CPE232_VALENCIA  Documents  examples.desktop  Pictures  Templates
Desktop          Downloads  Music             Public    Videos
janssenvalencia@workstation:~$ cd CPE232_VALENCIA
janssenvalencia@workstation:~/CPE232_VALENCIA$ ls -la
total 16
drwxrwxr-x  3 janssenvalencia janssenvalencia 4096 Aug 24 18:12 .
drwxr-xr-x 17 janssenvalencia janssenvalencia 4096 Aug 24 18:12 ..
drwxrwxr-x  8 janssenvalencia janssenvalencia 4096 Aug 24 18:12 .git
-rw-rw-r--  1 janssenvalencia janssenvalencia   17 Aug 24 18:12 README.md
```


janssenvalencia@workstation: ~/CPE232_VALENCIA
File Edit View Search Terminal Help

```
  GNU nano 2.9.3                          README.md

# CPE232_VALENCIA
```

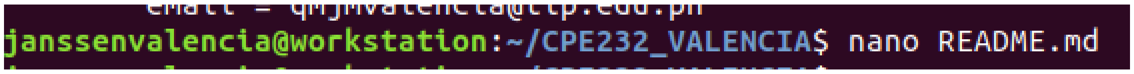
g. Use the following commands to personalize your git.

- *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*



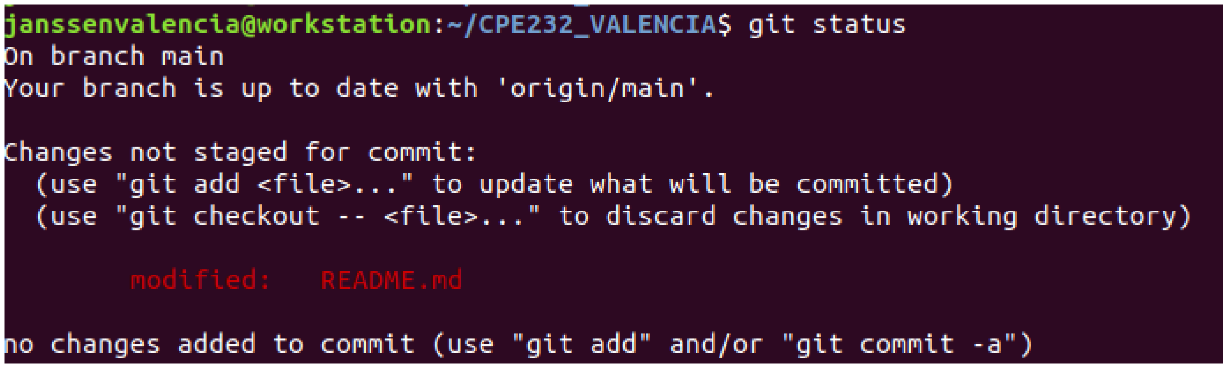
```
janssenvalencia@workstation:~/CPE232_VALENCIA$ git config --global user.name "J
anssenValencia"
janssenvalencia@workstation:~/CPE232_VALENCIA$ git config --global user.email q
mjmvalencia@tip.edu.ph
janssenvalencia@workstation:~/CPE232_VALENCIA$ cat ~/.gitconfig
[user]
        name = JanssenValencia
        email = qmjmvalencia@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.



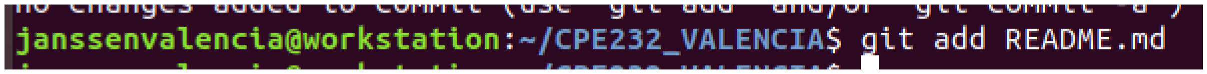
```
janssenvalencia@workstation:~/CPE232_VALENCIA$ nano README.md
```

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?



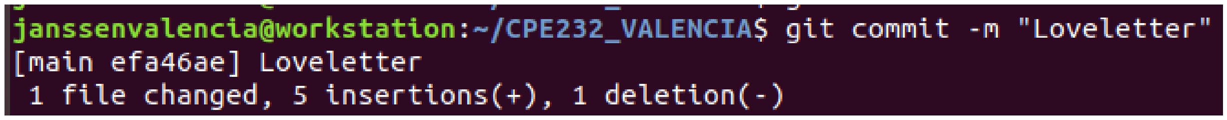
```
janssenvalencia@workstation:~/CPE232_VALENCIA$ git status
On branch main
Your branch is up to date with 'origin/main'.

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.



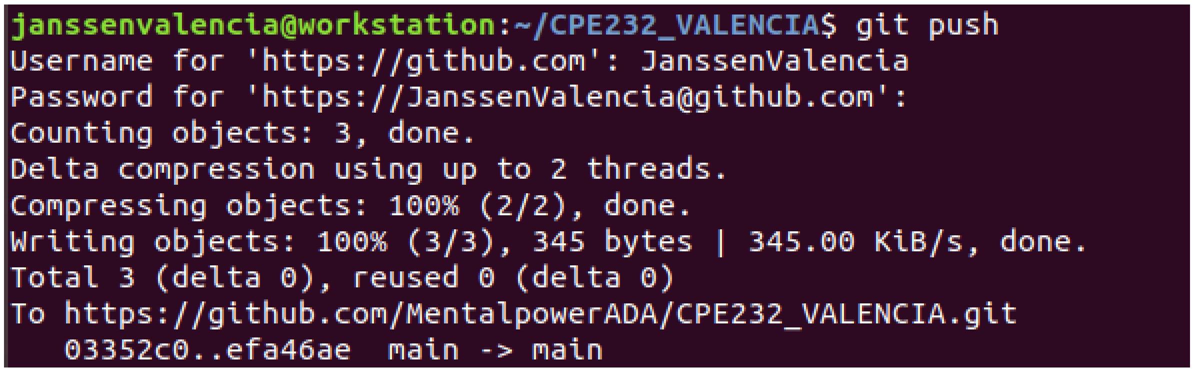
```
janssenvalencia@workstation:~/CPE232_VALENCIA$ git add README.md
```

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.



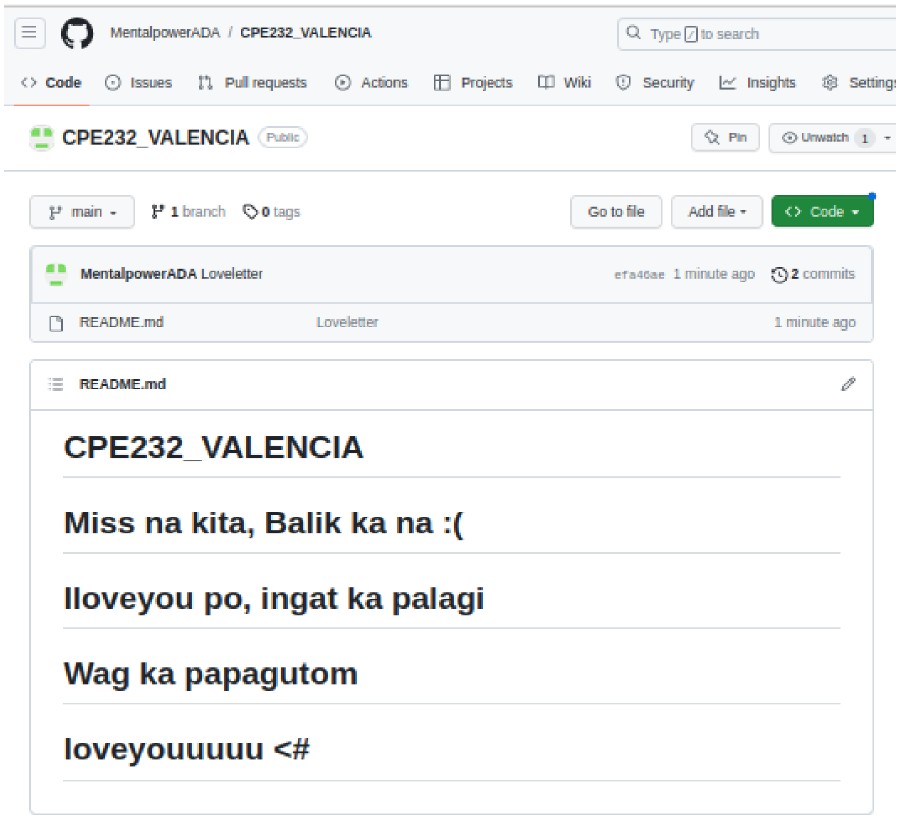
```
janssenvalencia@workstation:~/CPE232_VALENCIA$ git commit -m "Loveletter"
[main efa46ae] Loveletter
 1 file changed, 5 insertions(+), 1 deletion(-)
```

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.*



```
janssenvalencia@workstation:~/CPE232_VALENCIA$ git push
Username for 'https://github.com': JanssenValencia
Password for 'https://JanssenValencia@github.com':
Counting objects: 3, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 345 bytes | 345.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/MentalpowerADA/CPE232_VALENCIA.git
   03352c0..efa46ae  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.



CPE232_VALENCIA

MentalpowerADA Loveletter   efa46ae 1 minute ago   2 commits

README.md   Loveletter   1 minute ago

README.md

CPE232_VALENCIA

Miss na kita, Balik ka na :(

Iloveyou po, ingat ka palagi

Wag ka papagutom

loveyouuuuu <#

Reflections:
Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
   - **We have created an SSH key and copied it to both of the servers 1 and 2 for us to be able to access those servers remotely and easily without entering the password each time we connect to the servers. We also have configured and linked our own GitHub accounts to our local workstation and have configured and demonstrated how it works.**

Conclusions/Learnings:

- **In this hands-on activity we have connected to both of the remote servers with the use of SSH keys that we have generated and copied to both of the virtual servers, We also have created our very first git account and as well as demonstrated, installed and used it on our workstations. With these new skills and knowledge that I have learned while performing this hands-on activity, we can surely be indeed confident as well as competent as a soon-to-be systems administrators and future computer engineers.**

"*I Valencia, Mark Janssen Affirm that I will not give or receive any unauthorized help on this activity and that all work shall be my own*"