Name: Gabriel Soriano	**Date Performed: August 25, 2022**
Course/Section: CPE 232-CPE31S23	**Date Submitted: August 25, 2022**
Instructor: Engr. Taylar	**Semester and SY: 1st sem_2022-2023**

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.

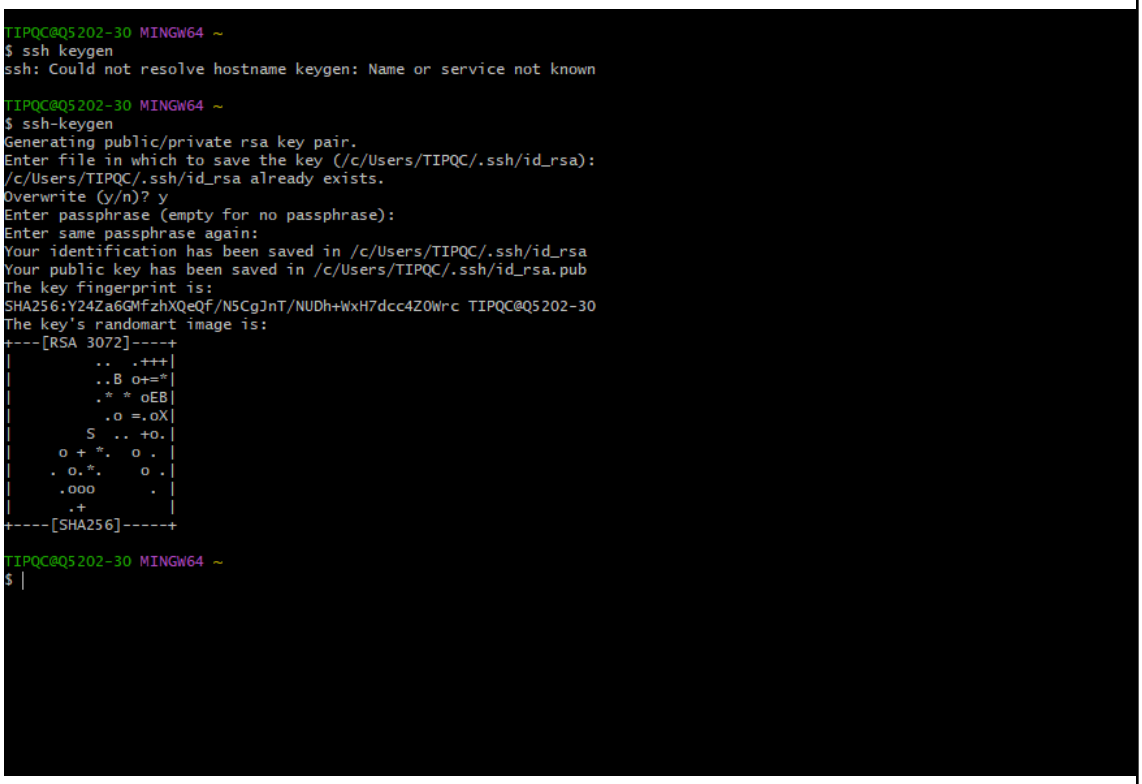
However, SSH keys are authentication credentials just like passwords. Thus, they must be managed somewhat analogously to usernames and passwords. They should have a proper termination process so that keys are removed when no longer needed.

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*.

SCREENSHOT:



```
TIPQC@Q5202-30 MINGW64 ~
$ ssh keygen
ssh: Could not resolve hostname keygen: Name or service not known

TIPQC@Q5202-30 MINGW64 ~
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/TIPQC/.ssh/id_rsa):
/c/Users/TIPQC/.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 /c/Users/TIPQC/.ssh/id_rsa
Your public key has been saved in /c/Users/TIPQC/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Y24Za6GMfzhXQeQf/N5CgJnT/NUDh+WxH7dcc4ZOWrc TIPQC@Q5202-30
The key's randomart image is:
+---[RSA 3072]----+
|          ..  .+++|
|          ..B o+=*|
|          .* * oEB|
|           .o =.oX|
|         S  .. +o.|
|      o + *.  o . |
|     . o.*.    o .|
|      .ooo      . |
|       .+         |
+----[SHA256]-----+

TIPQC@Q5202-30 MINGW64 ~
$ |
```

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.

SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/TIPQC/.ssh/id_rsa):
/c/Users/TIPQC/.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 /c/Users/TIPQC/.ssh/id_rsa
Your public key has been saved in /c/Users/TIPQC/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:XxfXmDf1yij7OU5Ob1TUI8C5uGWbpWPg3VEWu+M9Ecc TIPQC@Q5202-30
The key's randomart image is:
+---[RSA 4096]----+
|          ..o  .+|
|           o . BB|
|          . . B=E|
|         o + = =*|
|        S B O == |
|         + @ oo +|
|          + +. o.|
|           =.o. .|
|           .=o.  |
+----[SHA256]-----+

TIPQC@Q5202-30 MINGW64 ~
$ |
```

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.

SCREENSHOT:


```
TIPQC@Q5202-30 MINGW64 ~
$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/TIPQC/.ssh/id_rsa):
/c/Users/TIPQC/.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 /c/Users/TIPQC/.ssh/id_rsa
Your public key has been saved in /c/Users/TIPQC/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:XxfXmDf1yij7OU5Ob1TUI8C5uGWbpWPg3VEWu+M9Ecc TIPQC@Q5202-30
The key's randomart image is:
+---[RSA 4096]----+
|          ..o  .+|
|           o . BB|
|          . . B=E|
|         o + = =*|
|        S B O == |
|         + @ oo +|
|          + +. o.|
|           =.o. .|
|           .=o.  |
+----[SHA256]-----+

TIPQC@Q5202-30 MINGW64 ~
$ |
```


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

SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ ls -la .ssh
total 33
drwxr-xr-x 1 TIPQC 197121    0 Aug 25 08:06 ./
drwxr-xr-x 1 TIPQC 197121    0 Aug 25 07:31 ../
drwxr-xr-x 1 TIPQC 197121    0 Aug 25 08:02 DNSCache/
-rw-r--r-- 1 TIPQC 197121 3381 Aug 25 08:26 id_rsa
-rw-r--r-- 1 TIPQC 197121  740 Aug 25 08:26 id_rsa.pub
-rw-r--r-- 1 TIPQC 197121  936 Aug 25 08:06 known_hosts
-rw-r--r-- 1 TIPQC 197121  192 Aug 25 08:04 known_hosts.old
drwxr-xr-x 1 TIPQC 197121    0 Aug 23 09:03 ssh-copy-id.QRsfsSktr4/

TIPQC@Q5202-30 MINGW64 ~
$ |
```

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.

   SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id
Usage: /usr/bin/ssh-copy-id [-h|-?|-f|-n|-s] [-i [identity_file]] [-p port] [-F
alternative ssh_config file] [[-o <ssh -o options>] ...] [user@]hostname
        -f: force mode -- copy keys without trying to check if they are already
installed
        -n: dry run    -- no keys are actually copied
        -s: use sftp   -- use sftp instead of executing remote-commands. Can be
useful if the remote only allows sftp
        -h|-?: print this help

TIPQC@Q5202-30 MINGW64 ~
$ |
```

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

   SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id-i ~/.ssh/id_rsa soriano2@192.168.56.105
bash: ssh-copy-id-i: command not found

TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id -i ~/.ssh/id_rsa soriano2@192.168.56.105
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/c/Users/TIPQC/.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 prompt
ed now it is to install the new keys
soriano2@192.168.56.105's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'soriano2@192.168.56.105'"
and check to make sure that only the key(s) you wanted were added.


TIPQC@Q5202-30 MINGW64 ~
$ |
```

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.

   SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id-i ~/.ssh/id_rsa soriano2@192.168.56.105
bash: ssh-copy-id-i: command not found

TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id -i ~/.ssh/id_rsa soriano2@192.168.56.105
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/c/Users/TIPQC/.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 prompt
ed now it is to install the new keys
soriano2@192.168.56.105's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'soriano2@192.168.56.105'"
and check to make sure that only the key(s) you wanted were added.


TIPQC@Q5202-30 MINGW64 ~
$ |
```

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?

   - The connection will work only if the connection has been authenticated via using the commands that copies the id for the key to be verified and authenticated in both the bash and the ubuntu. The connection didn't ask for a password because I've already entered the password while authenticating it.

SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ ssh soriano2@192.168.56.105
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

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

0 updates can be applied immediately.

Last login: Thu Aug 25 08:08:44 2022 from 192.168.56.1
soriano2@server1-Server1:~$
```

```
TIPQC@Q5202-30 MINGW64 ~/.ssh
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQC8huulcPnVgOJ3nGuMKnqMsvRXBRME3hSMKxBB4nWG9ZisynO8pfBZIrDVMuHY5uPO1xA9zvDyXSO6FtkaeOOn
mgOps3lleqBg48tyUxjdpCds4hltyNwe5qL4SHTeBwTYr/PzDCaOgVHQ4eI/gWNsDBPuuHT5VuVEstx+2gJFoVzRl5PBckP4gVDtdeMTkdwk4bvjFnX+VDcZUZeM
kRcERimO2ZultBMfkX12gOTQWE+J4L1zuyeRdLHksnQHRs9IvMsMOtdg6Rl4osxTrYhOEN1K2gFmVmNmrkNEclaXbnviHg+Won4a7qb/1ejTXWHrPnvQOqYmWZVN
XwVfJniFYVjz8o4/95Z+5gcMY8wcBUpeuamhbvnMfKhYyY9vOsoTiS8C5febqUa3kiOji5sJqjCj28d8L6hs1N+CPWLV9awqm1NKooLmfYts+GBNXT1of4ycCFKF
V3KV87i4wdVA7WoSqzIkKVr5UlvZvkJ8fwaAntUM9A04tNN/aHStdsr7doe5aQtGq6q8KbUaSEcLh5sQqsGUxEymZ3ASnEFiyVgp3FEyMgJYUl42mSFF+OBiGfx3
sYyMmsj/pSLTV5yzmFH78OaJS/VwhP9sVkBEb9muAMELKFzKrjYQ== TIPQC@Q5202-30

TIPQC@Q5202-30 MINGW64 ~/.ssh
$ ssh soriano2@192.168.56.106
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-46-generic x86_64)

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

0 updates can be applied immediately.

Last login: Thu Aug 18 12:29:01 2022 from 192.168.56.104
soriano2@soriano2-localmachine:~$
```

Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?
   - The ssh program can be described as a very secured command program, why? because it does use both public and private keys for authentication. It can be used as a key to authenticate other computers' connections with ssh.

2. How do you know that you already installed the public key to the remote servers?
   - When I issue the command cd then go to the ssh directory, then list its content, then a file “authorized_keys” is present.

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

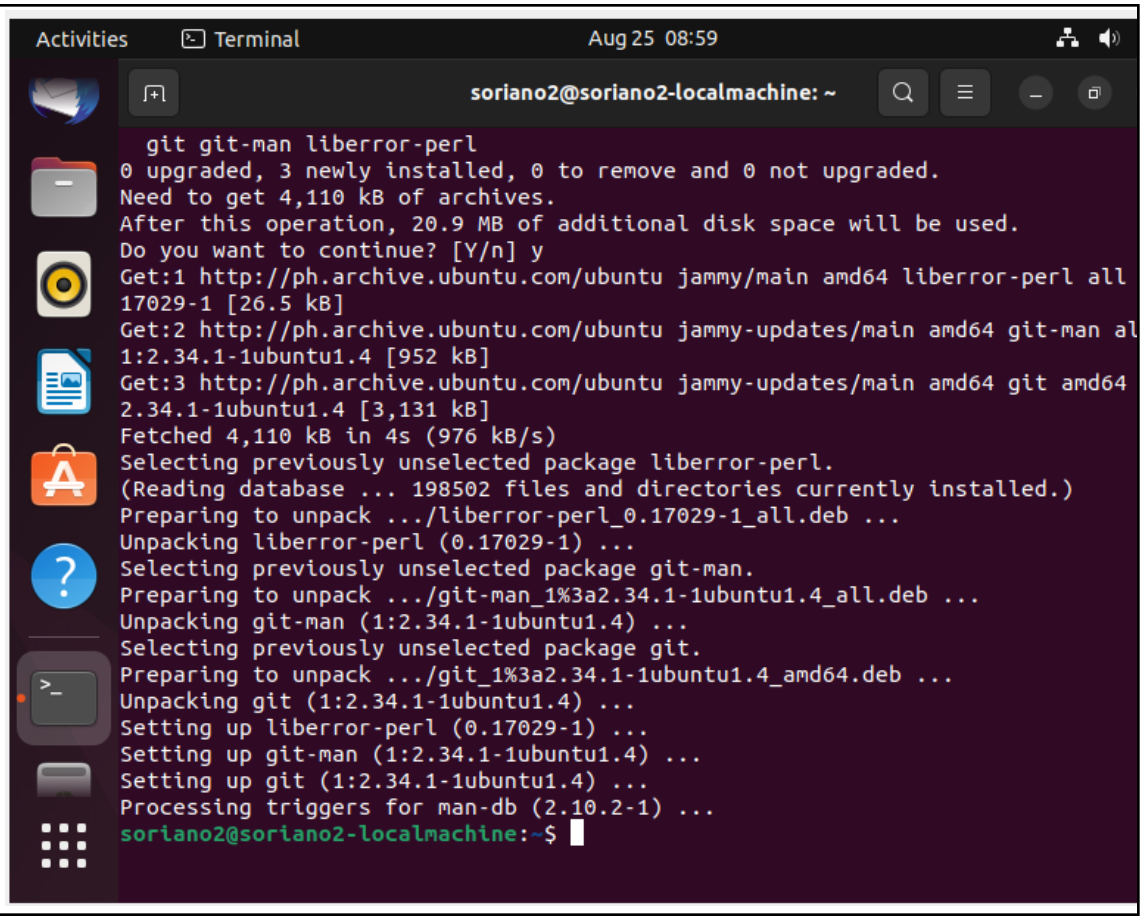
1. On the local machine, verify the version of your git using the command *which git*. If a directory of git is displayed, then you don't need to install git. Otherwise, to install git, use the following command: *sudo apt install git*

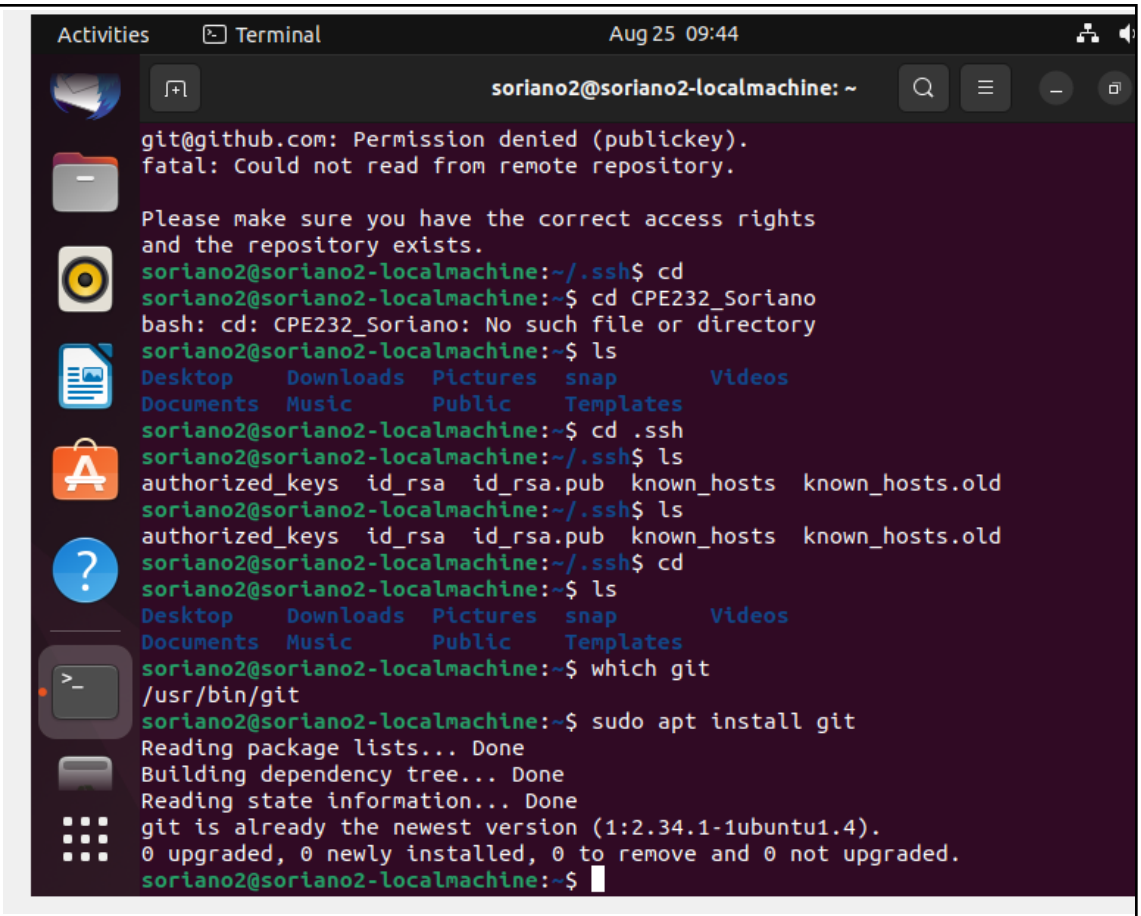
SCREENSHOT:

```
TIPQC@Q5202-30 MINGW64 ~
$ which git
/mingw64/bin/git

TIPQC@Q5202-30 MINGW64 ~
$ |
```

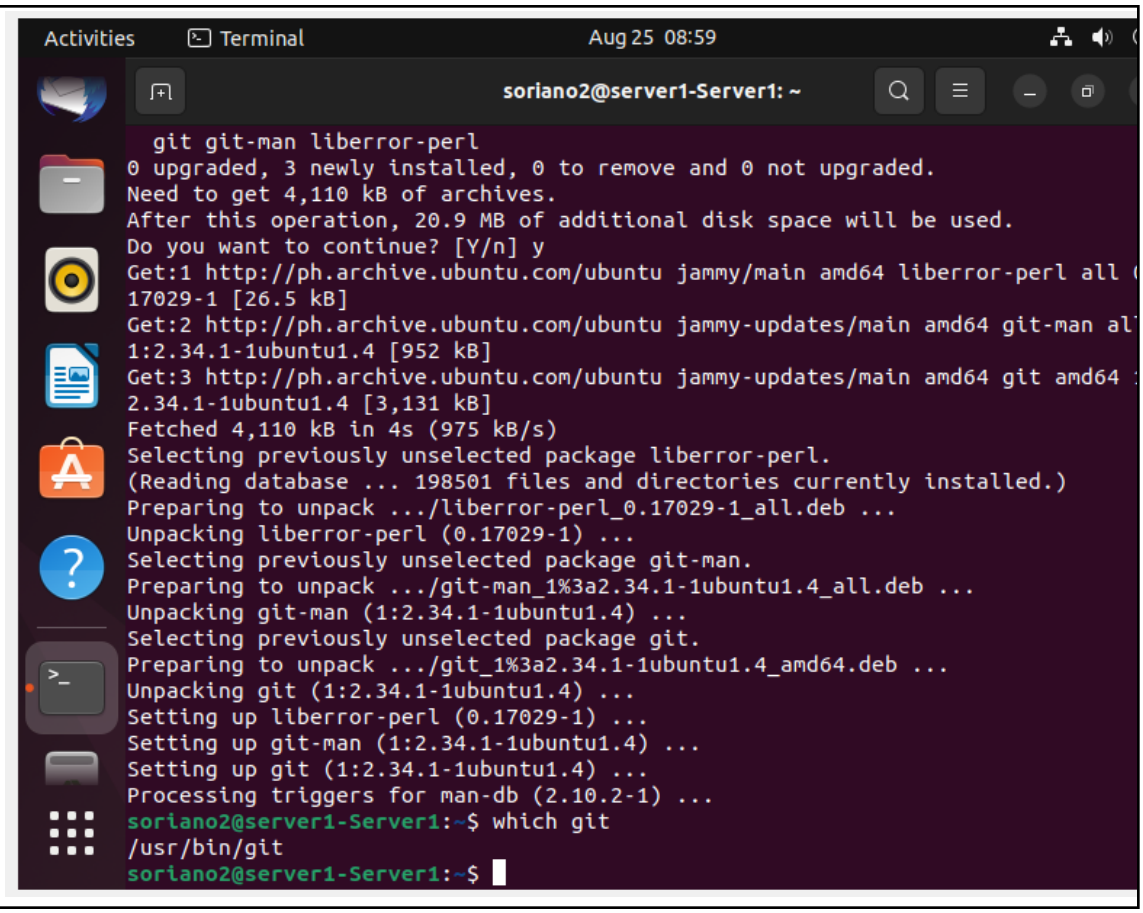
```
Activities   Terminal   Aug 25 08:58
soriano2@server1-Server1: ~
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl al
17029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man
1:2.34.1-1ubuntu1.4 [952 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd6
2.34.1-1ubuntu1.4 [3,131 kB]
Fetched 4,110 kB in 4s (975 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 198501 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.34.1-1ubuntu1.4_all.deb ...
Unpacking git-man (1:2.34.1-1ubuntu1.4) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.4_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.4) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.34.1-1ubuntu1.4) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
soriano2@server1-Server1:~$ S
```

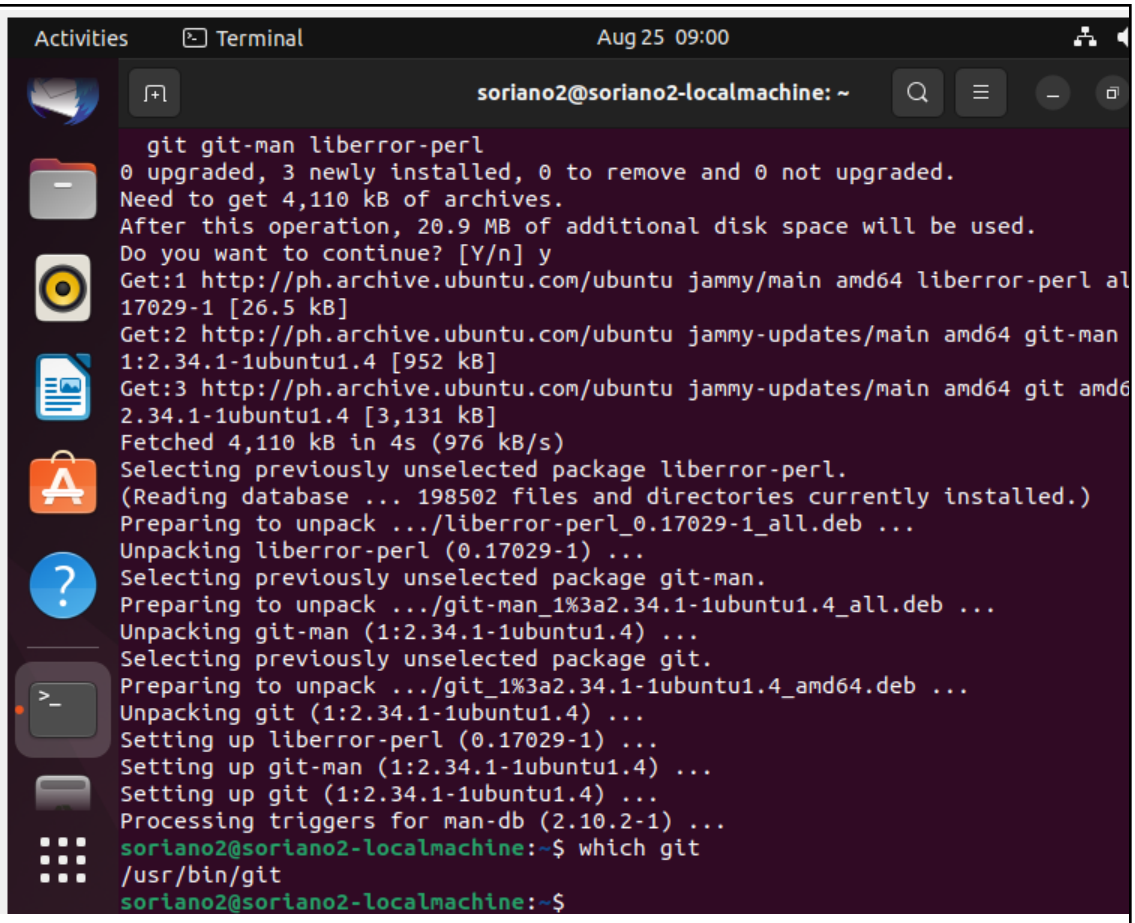

Activities
Terminal
Aug 25 08:59
soriano2@soriano2-localmachine: ~
git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all
17029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man al
1:2.34.1-1ubuntu1.4 [952 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd64
2.34.1-1ubuntu1.4 [3,131 kB]
Fetched 4,110 kB in 4s (976 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 198502 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.34.1-1ubuntu1.4_all.deb ...
Unpacking git-man (1:2.34.1-1ubuntu1.4) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.4_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.4) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.34.1-1ubuntu1.4) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
soriano2@soriano2-localmachine:~$

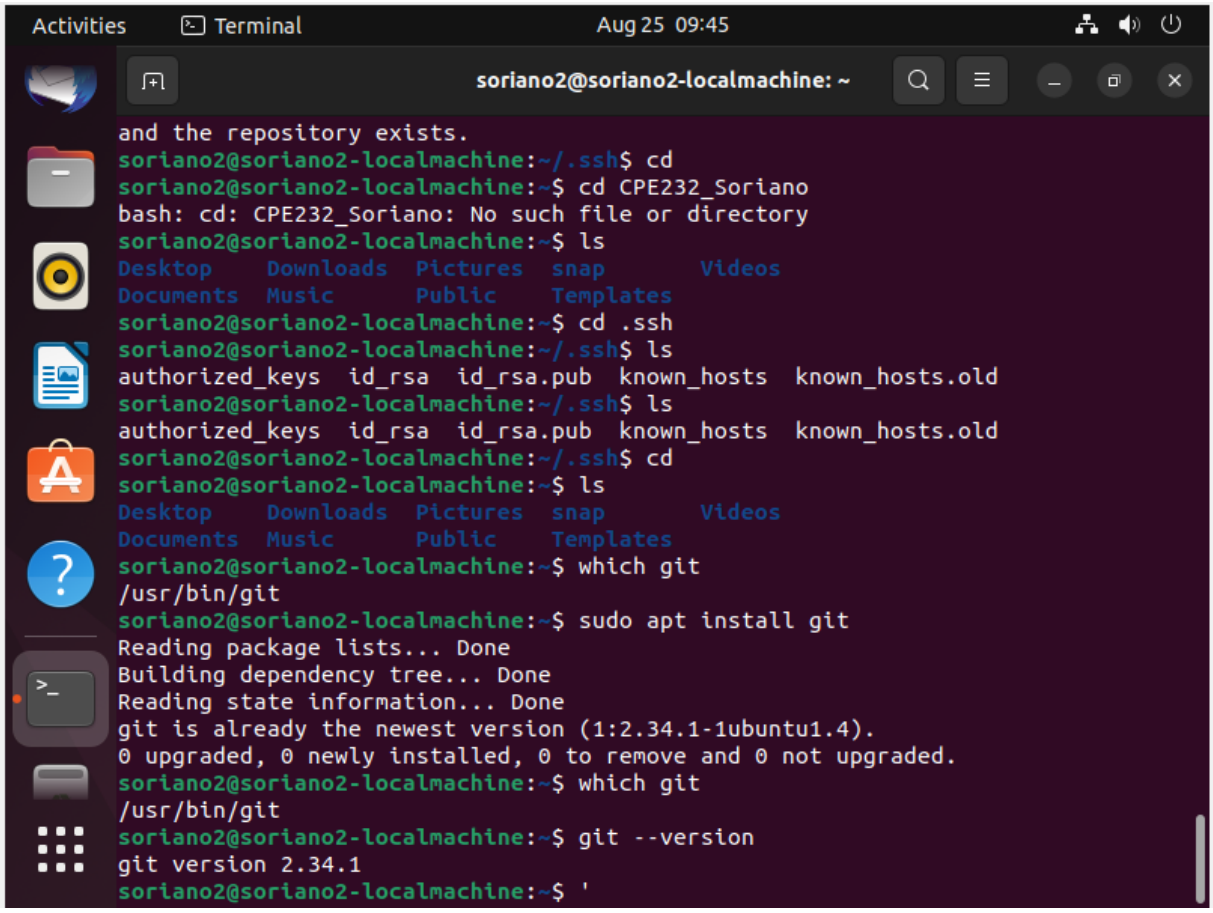


2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/bin/git.*

SCREENSHOT:

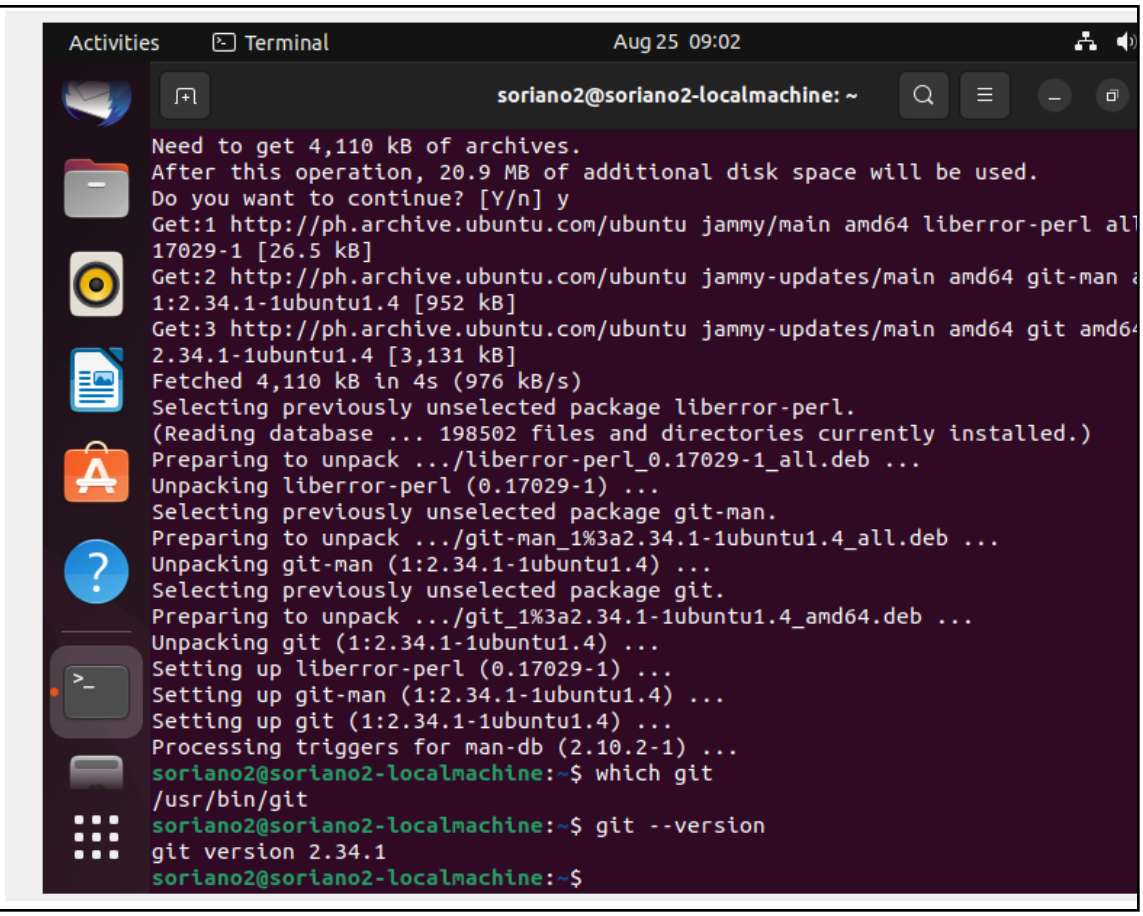

Activities
Terminal
Aug 25 08:59
soriano2@server1-Server1: ~
git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all (
17029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man al
1:2.34.1-1ubuntu1.4 [952 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd64
2.34.1-1ubuntu1.4 [3,131 kB]
Fetched 4,110 kB in 4s (975 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 198501 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.34.1-1ubuntu1.4_all.deb ...
Unpacking git-man (1:2.34.1-1ubuntu1.4) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.4_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.4) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.34.1-1ubuntu1.4) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
soriano2@server1-Server1:~$ which git
/usr/bin/git
soriano2@server1-Server1:~$

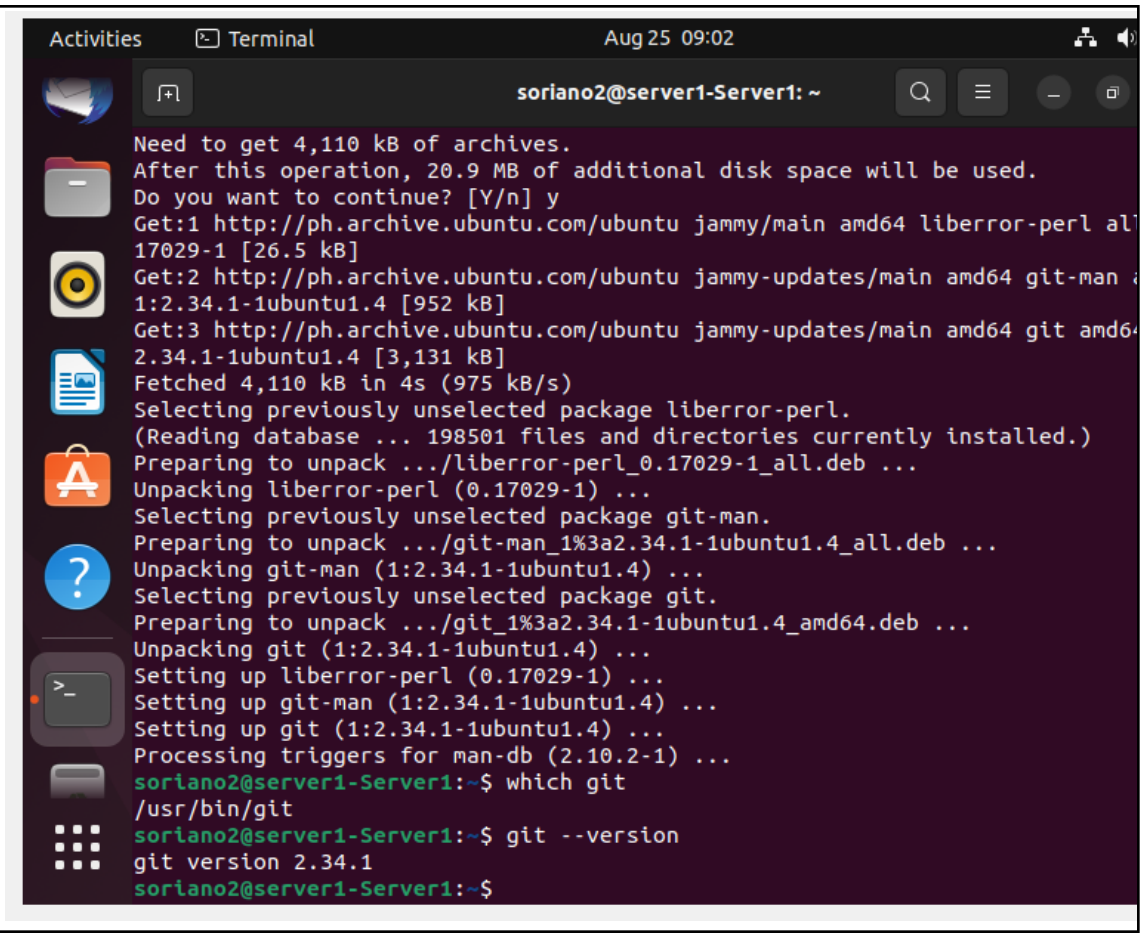

Activities
Terminal
Aug 25 09:00
soriano2@soriano2-localmachine: ~
git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl al
17029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man
1:2.34.1-1ubuntu1.4 [952 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd6
2.34.1-1ubuntu1.4 [3,131 kB]
Fetched 4,110 kB in 4s (976 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 198502 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.34.1-1ubuntu1.4_all.deb ...
Unpacking git-man (1:2.34.1-1ubuntu1.4) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.4_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.4) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.34.1-1ubuntu1.4) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
soriano2@soriano2-localmachine:~$ which git
/usr/bin/git
soriano2@soriano2-localmachine:~$

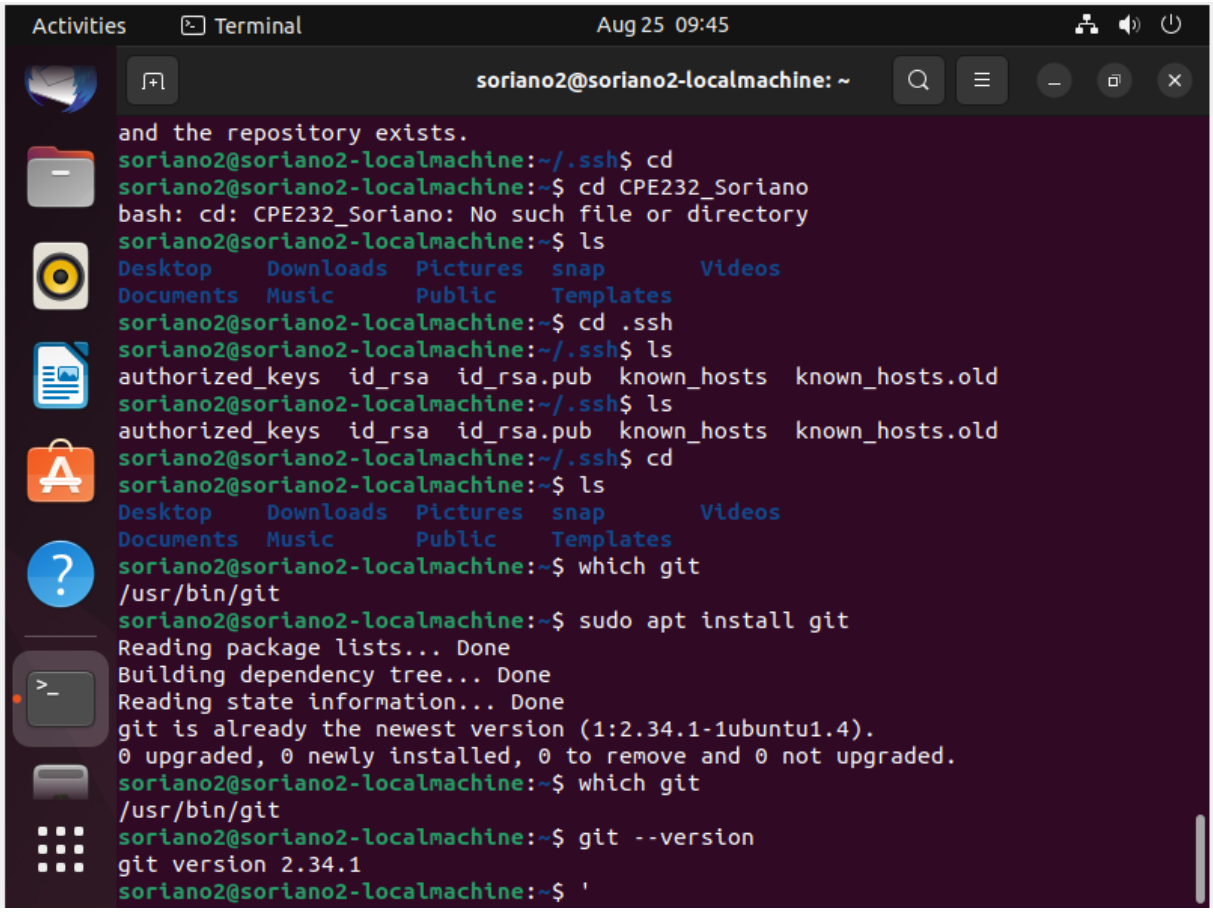


3. The version of git installed in your device is the latest. Try issuing the command *git --version* to know the version installed.

   SCREENSHOT:

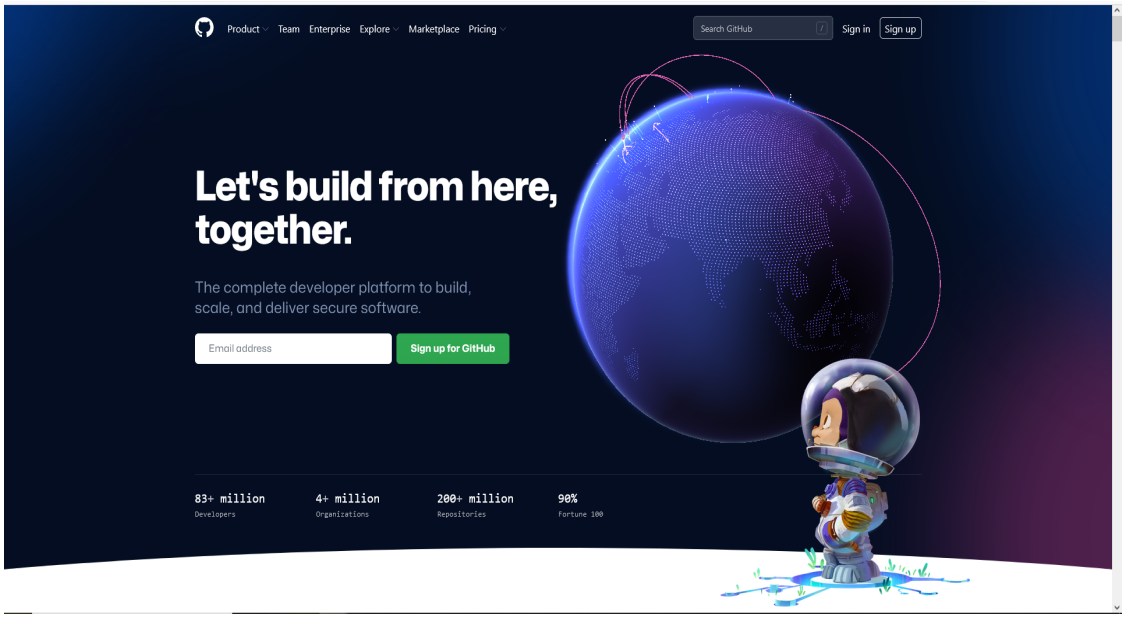

Activities
Terminal
Aug 25 09:02
soriano2@soriano2-localmachine: ~
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all
17029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man
1:2.34.1-1ubuntu1.4 [952 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd6
2.34.1-1ubuntu1.4 [3,131 kB]
Fetched 4,110 kB in 4s (976 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 198502 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.34.1-1ubuntu1.4_all.deb ...
Unpacking git-man (1:2.34.1-1ubuntu1.4) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.4_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.4) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.34.1-1ubuntu1.4) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
soriano2@soriano2-localmachine:~$ which git
/usr/bin/git
soriano2@soriano2-localmachine:~$ git --version
git version 2.34.1
soriano2@soriano2-localmachine:~$


Activities
Terminal
Aug 25 09:02
soriano2@server1-Server1: ~
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all
17029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man
1:2.34.1-1ubuntu1.4 [952 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd6
2.34.1-1ubuntu1.4 [3,131 kB]
Fetched 4,110 kB in 4s (975 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 198501 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17029-1_all.deb ...
Unpacking liberror-perl (0.17029-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.34.1-1ubuntu1.4_all.deb ...
Unpacking git-man (1:2.34.1-1ubuntu1.4) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.4_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.4) ...
Setting up liberror-perl (0.17029-1) ...
Setting up git-man (1:2.34.1-1ubuntu1.4) ...
Setting up git (1:2.34.1-1ubuntu1.4) ...
Processing triggers for man-db (2.10.2-1) ...
soriano2@server1-Server1:~$ which git
/usr/bin/git
soriano2@server1-Server1:~$ git --version
git version 2.34.1
soriano2@server1-Server1:~$



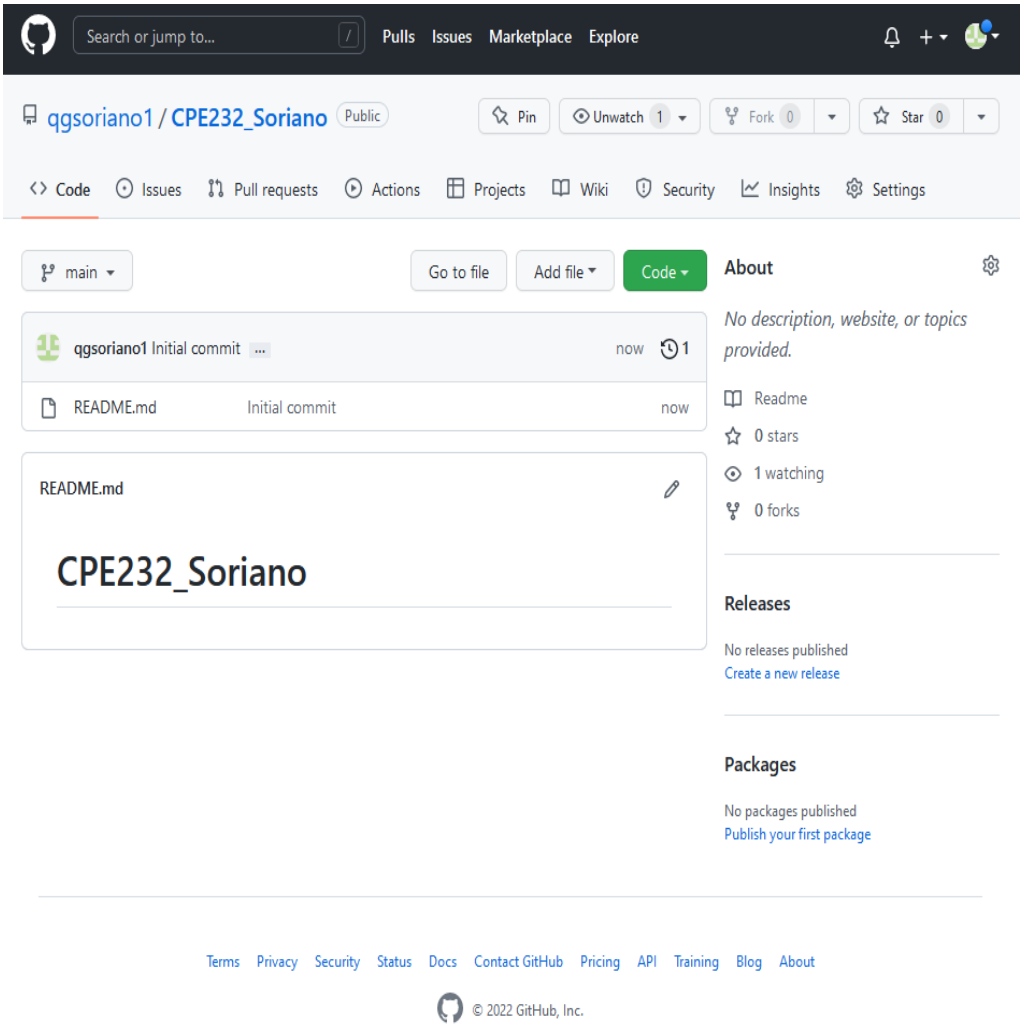
4. Using the browser in the local machine, go to www.github.com.

SCREENSHOT:



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.

SCREENSHOT:

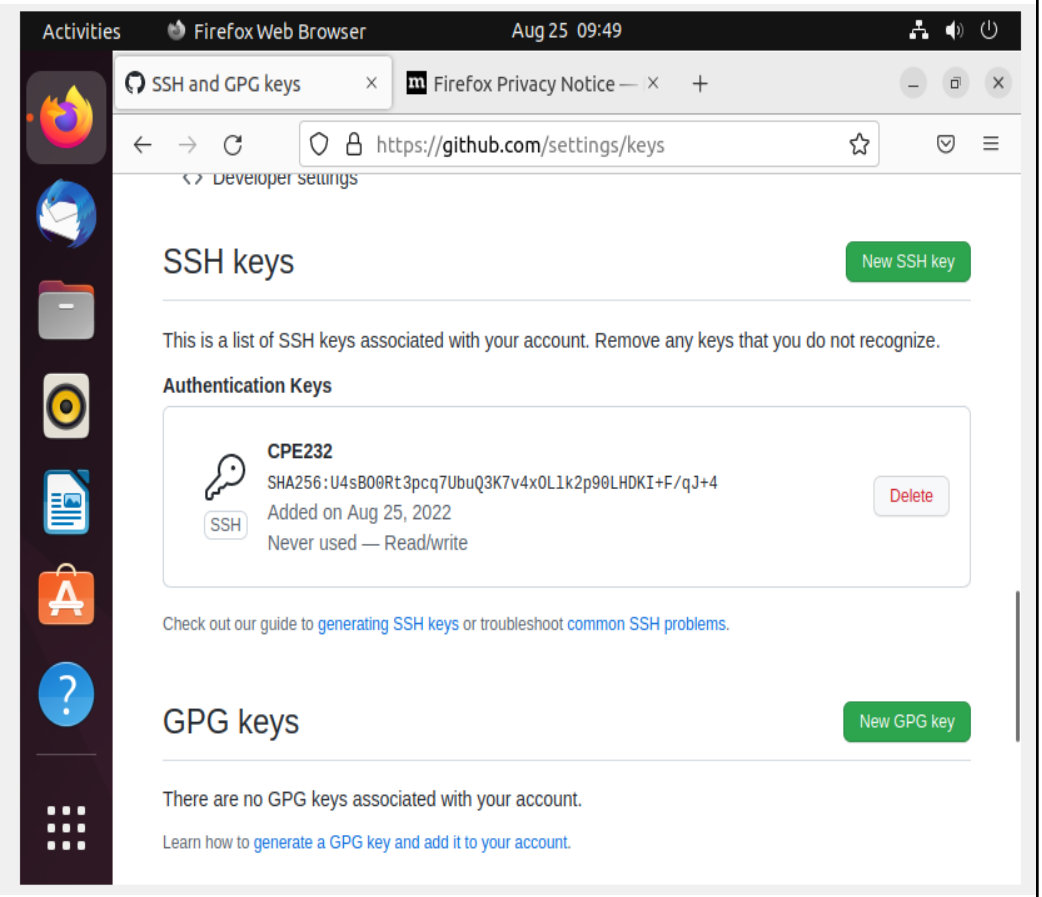


    b. Create a new SSH key on GitHub. Go your profile's setting and click SSH and GPG keys. If there is an existing key, make sure to delete it. To

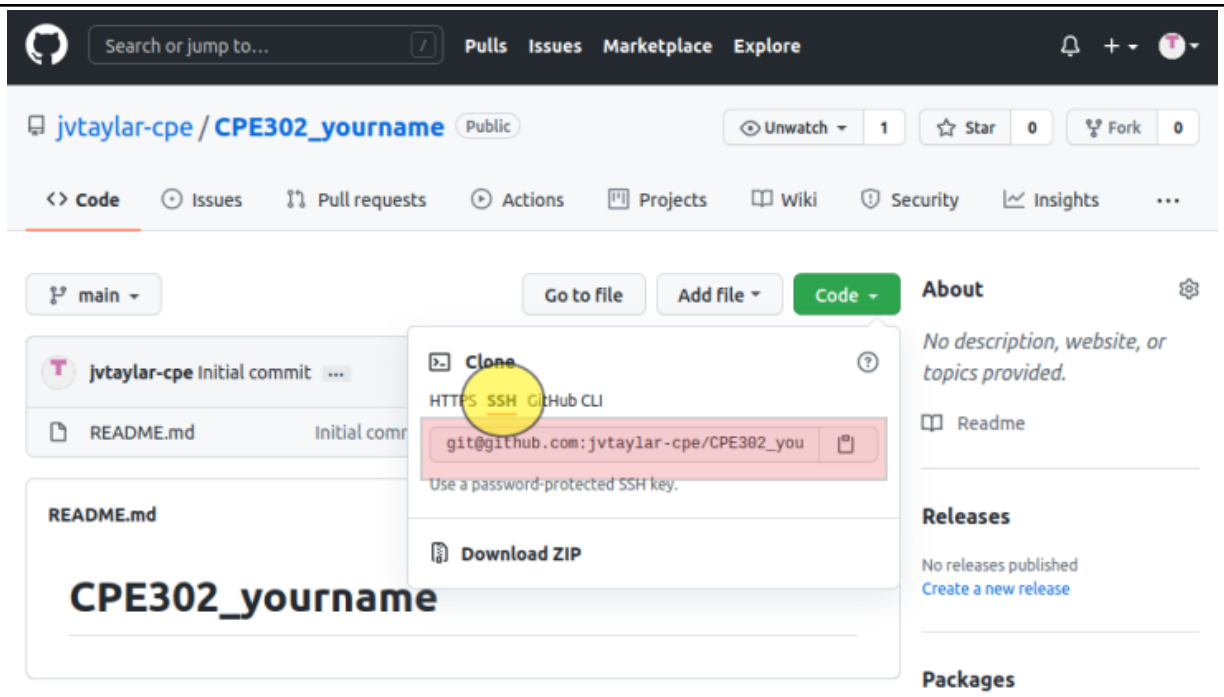
create a new SSH keys, click New SSH Key. Write CPE232 key as the title of the key.

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.

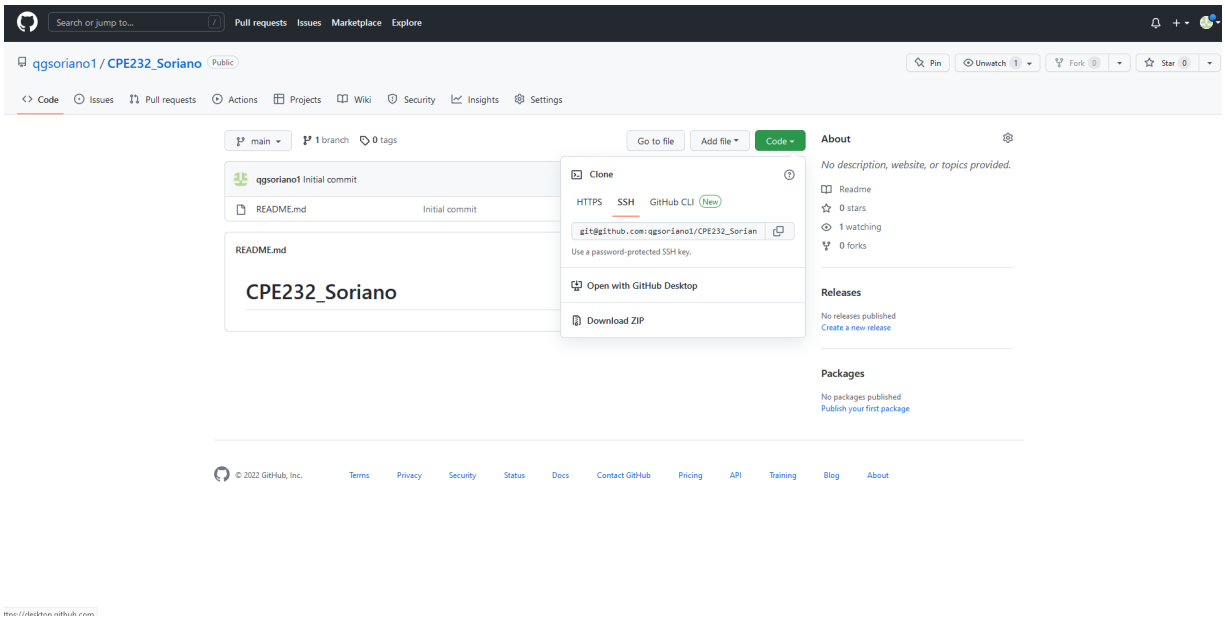
SCREENSHOT:



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.

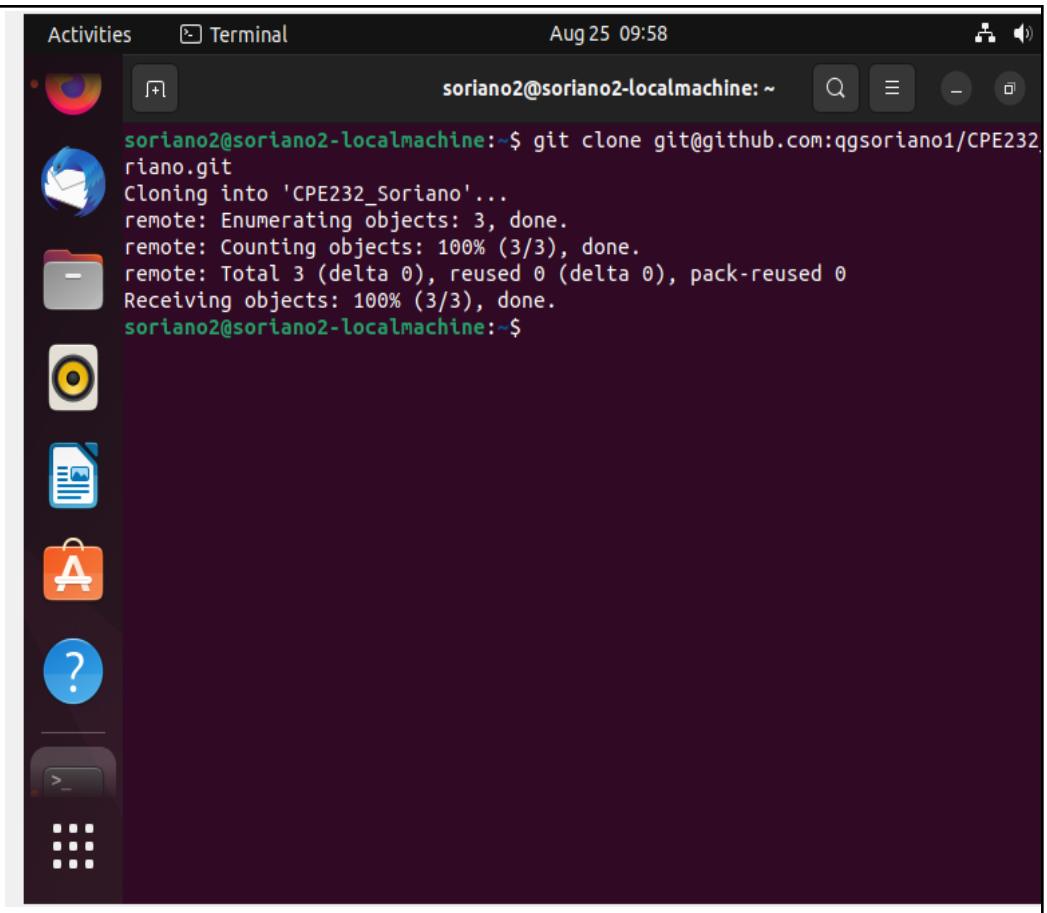


SCREENSHOT:



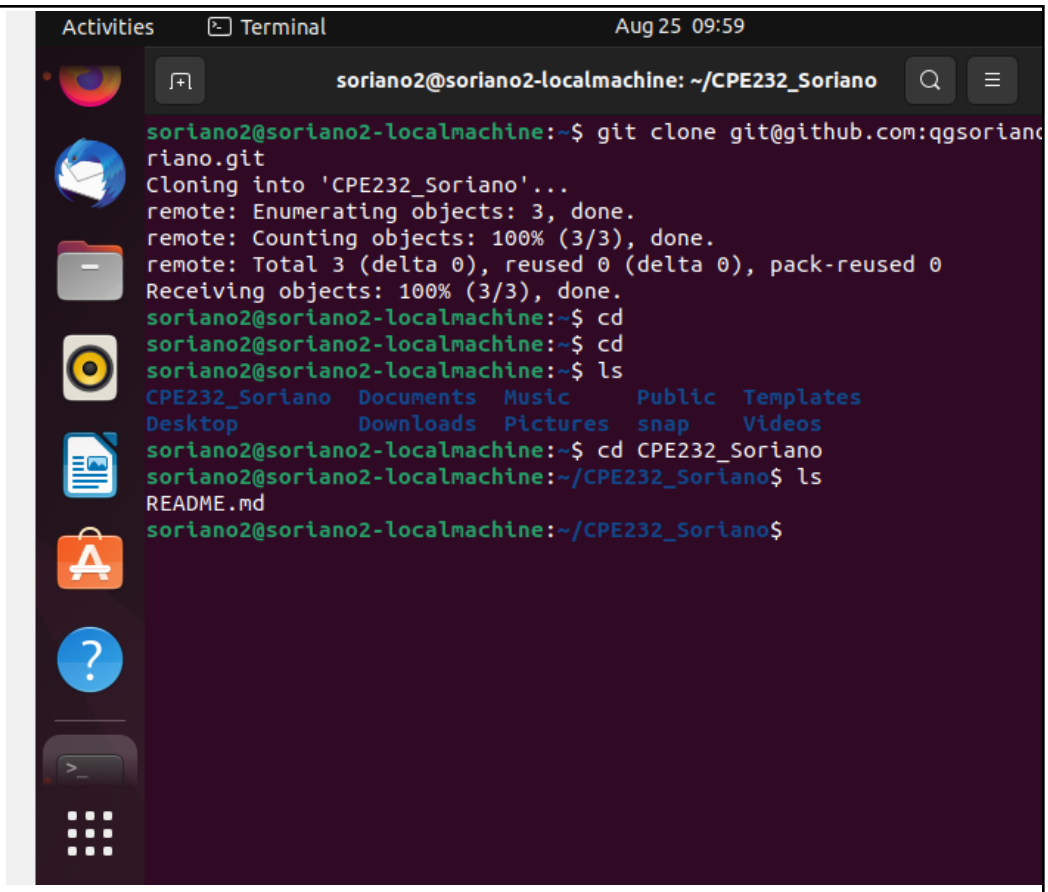
e. Issue the command git clone followed by the copied link. For example, *git clone* [*git@github.com:jvtaylar-cpe/CPE232_yourname.git*](). When prompted to continue connecting, type yes and press enter.

SCREENSHOT:



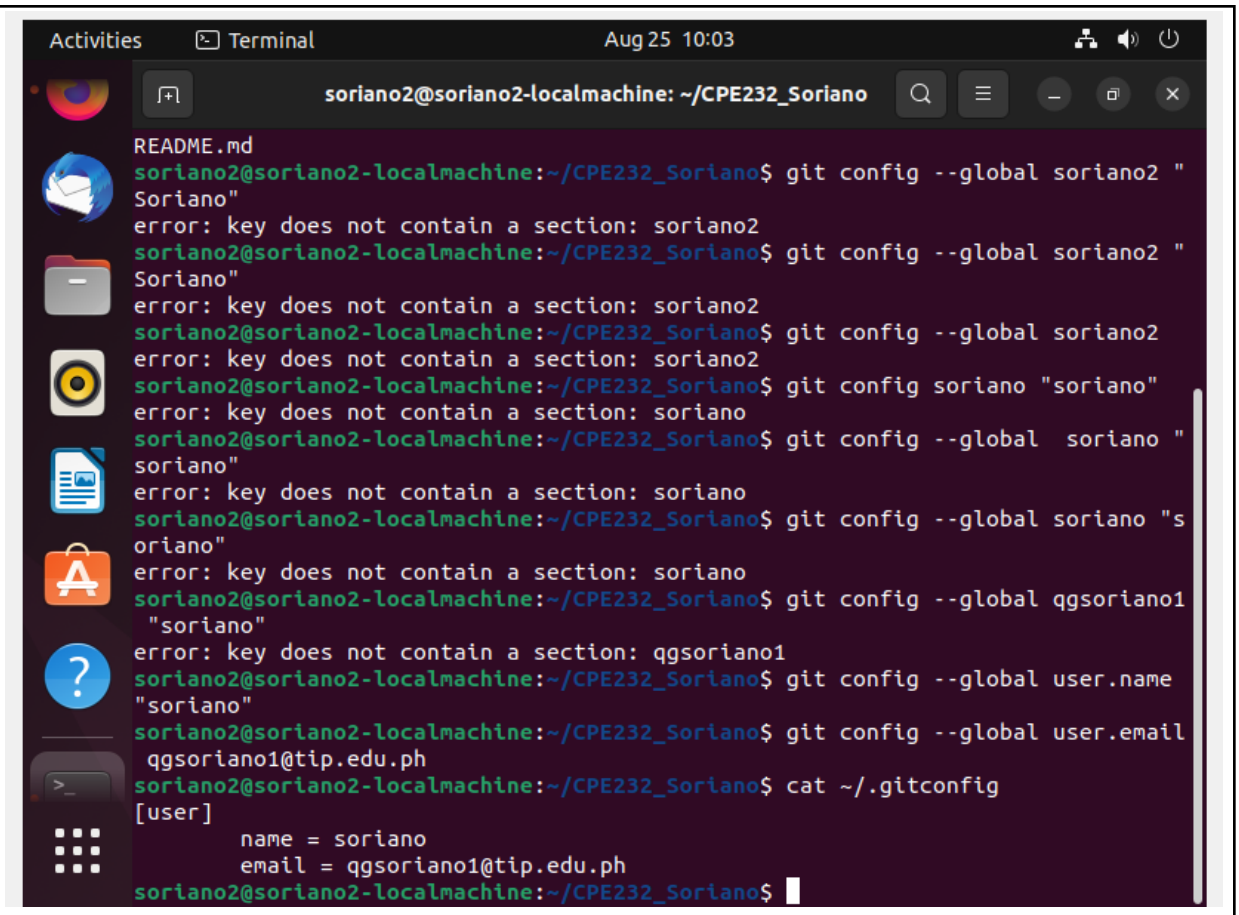
f. To verify that you have cloned the GitHub repository, issue the command *ls*. Observe that you have the CPE232_yourname in the list of your directories. Use CD command to go to that directory and LS command to see the file README.md.

SCREENSHOT:



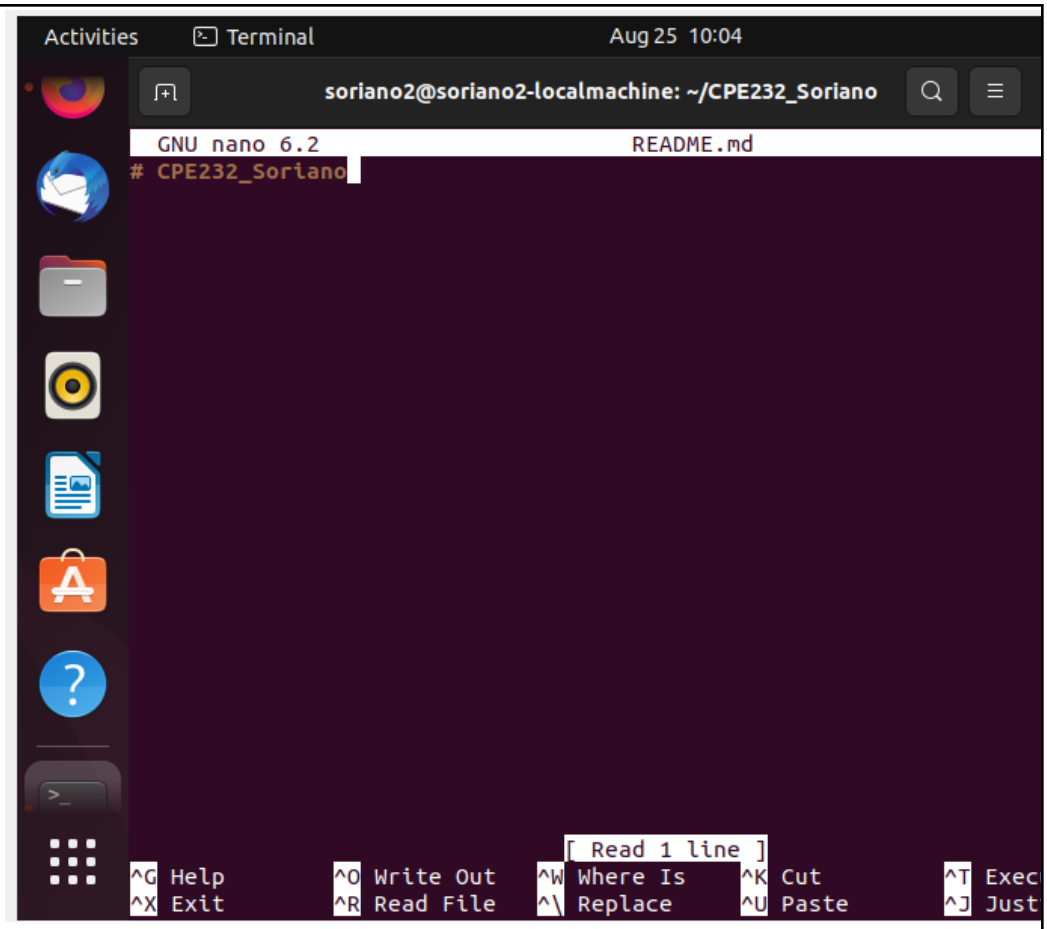
g. Use the following commands to personalize your git.
   - *git config --global user.name "Your Name"*
   - *git config --global user.email yourname@email.com*
   - Verify that you have personalized the config file using the command *cat ~/.gitconfig*

SCREENSHOT:



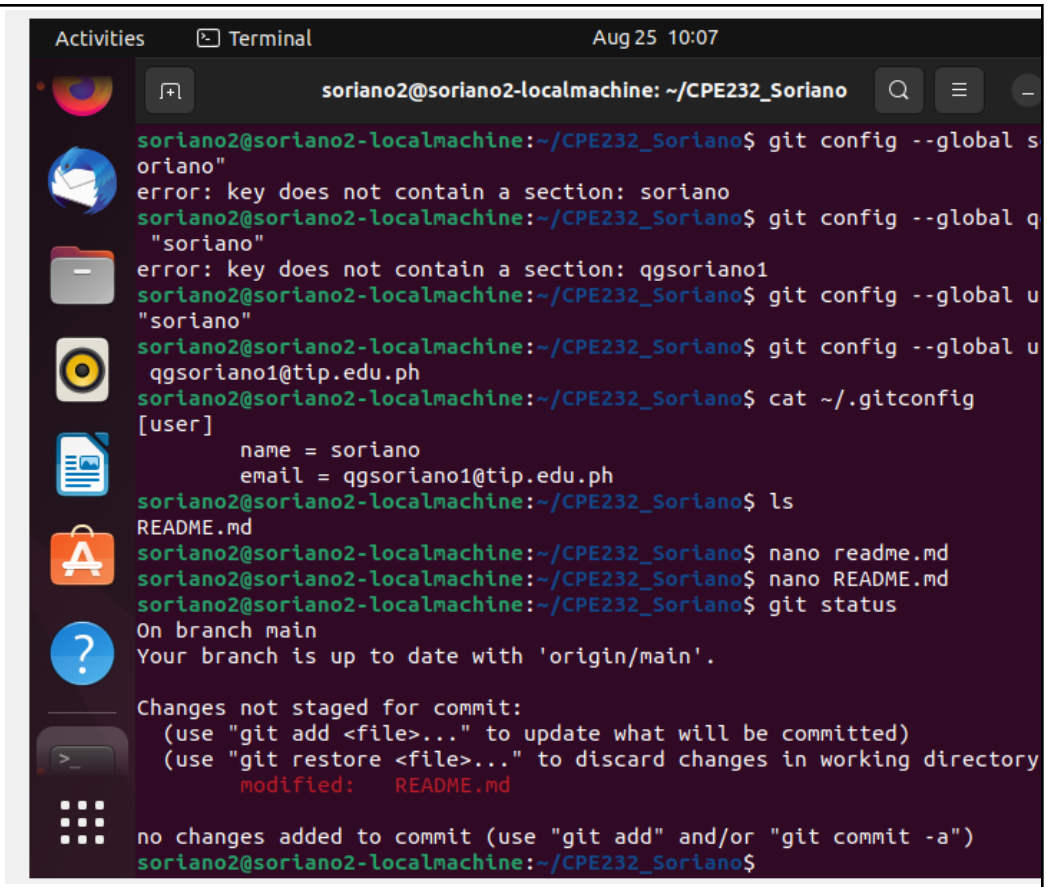
h. Edit the README.md file using nano command. Provide any information on the markdown file pertaining to the repository you created. Make sure to write out or save the file and exit.

SCREENSHOT:



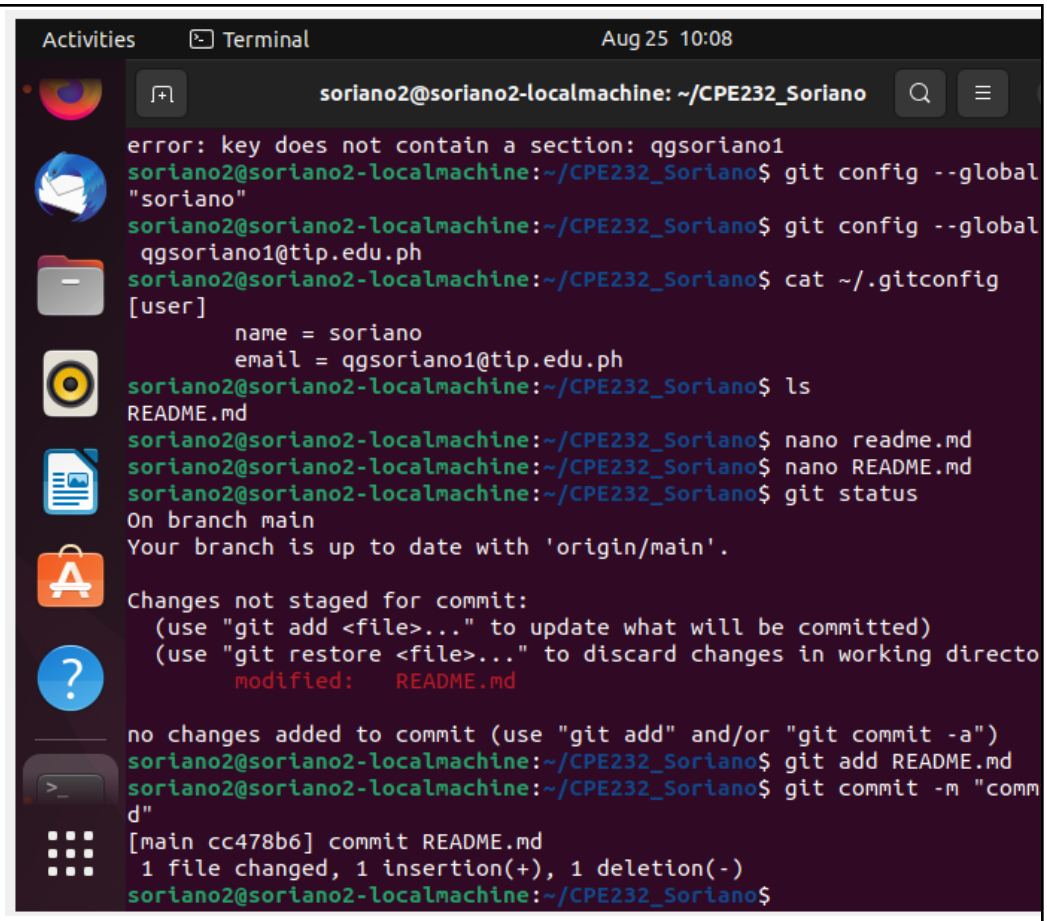
i. Use the *git status* command to display the state of the working directory and the staging area. This command shows which changes have been staged, which haven't, and which files aren't being tracked by Git. Status output does not show any information regarding the committed project history. What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

SCREENSHOT:



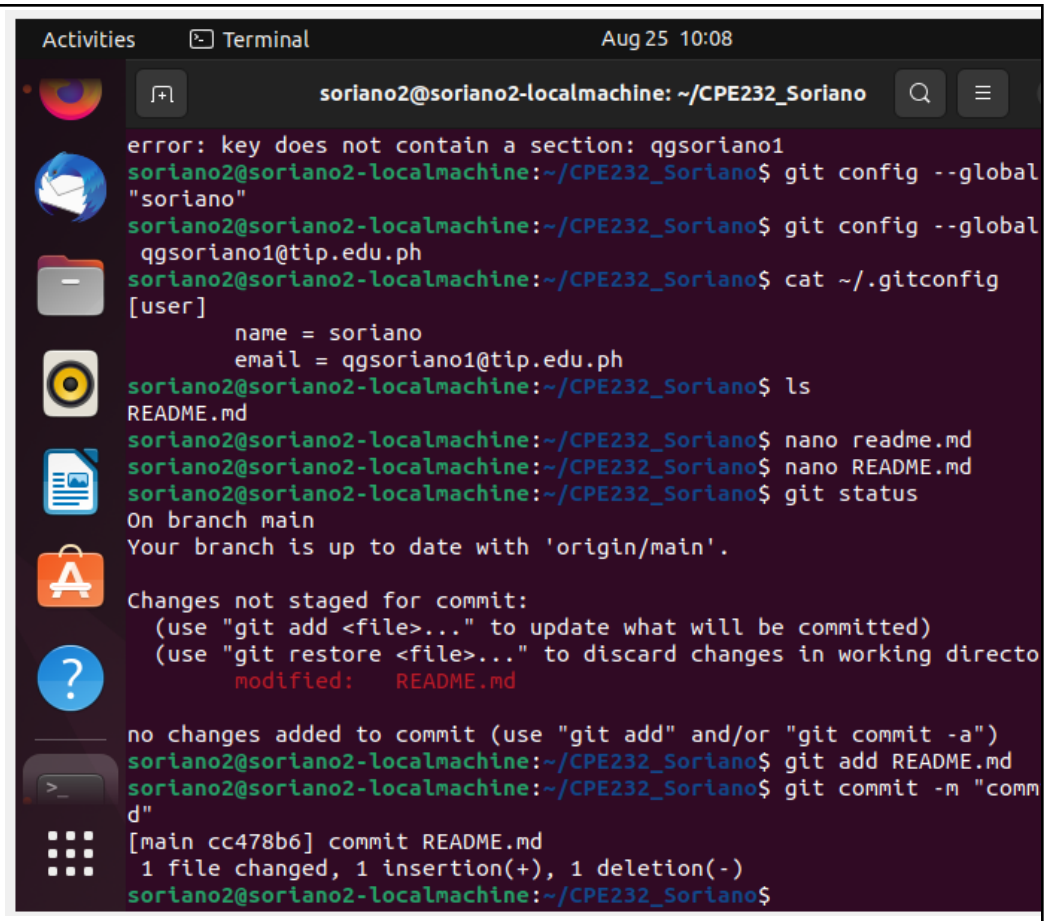
j. Use the command *git add README.md* to add the file into the staging area.

SCREENSHOT:



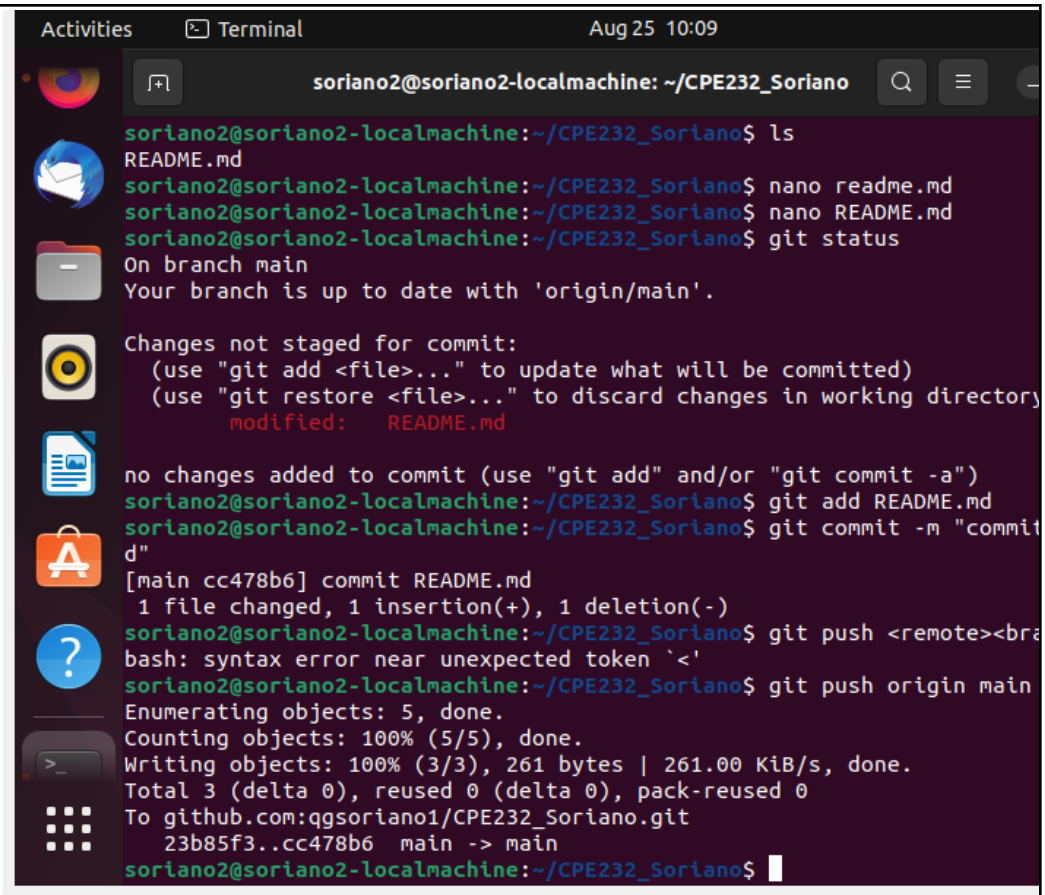
k. Use the *git commit -m "your message"* to create a snapshot of the staged changes along the timeline of the Git projects history. The use of this command is required to select the changes that will be staged for the next commit.

SCREENSHOT:



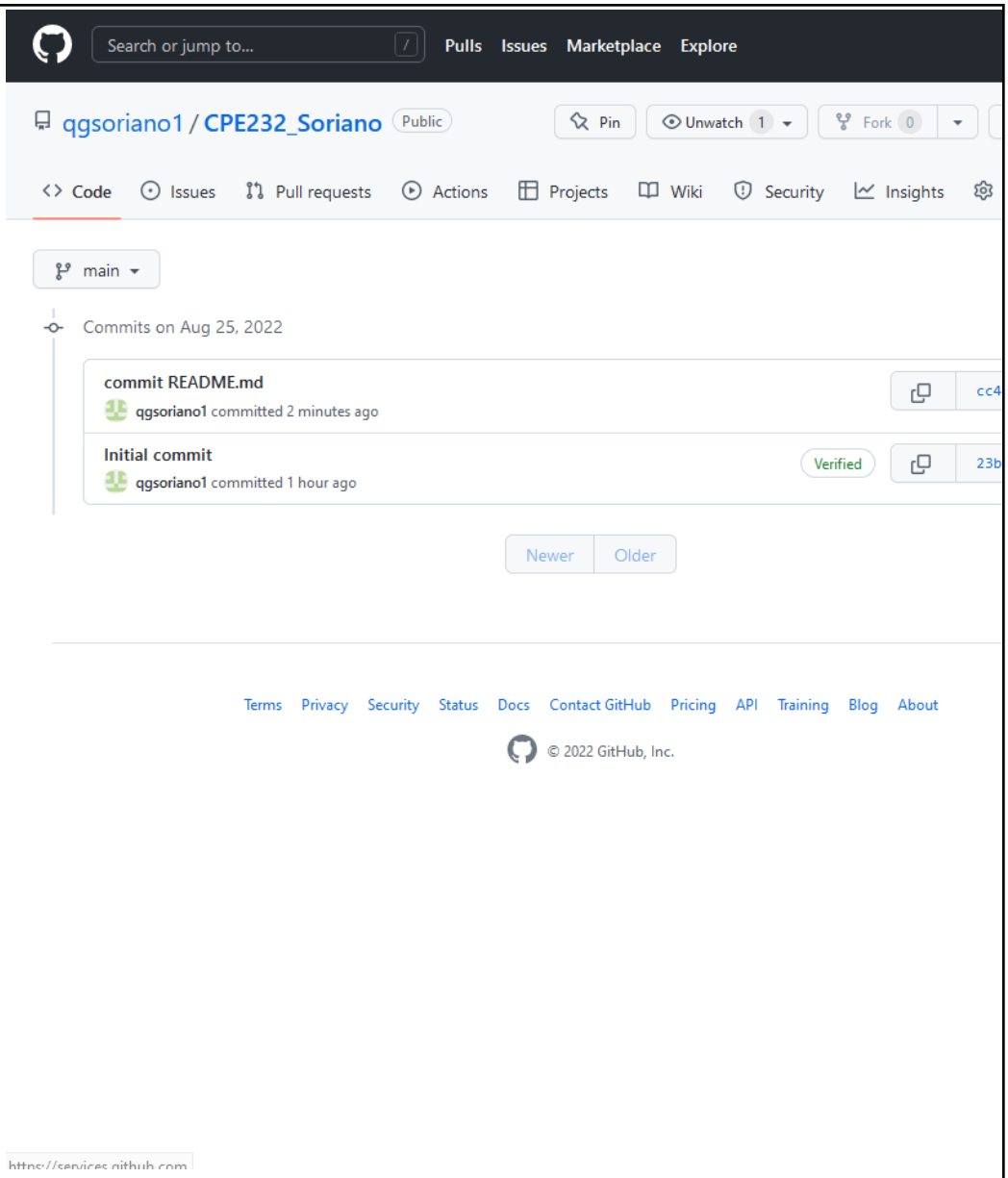
I. Use the command *git push <remote><branch>* to upload the local repository content to GitHub repository. Pushing means to transfer commits from the local repository to the remote repository. As an example, you may issue *git push origin main.*

SCREENSHOT:



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.

SCREENSHOT:



Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
   - **I already have done the authorization and the verifying of the connections through the user name of different PCs and different IP addresses. I also have executed the git clone command which does literally clone the one in the browser, the git clone url from your repository. And then checked the history of the editing and comment editing and text editing after the nano command in the local machine, and then check the history online through the browser in github.com**

4. How important is the inventory file?

   - **A great inventory will help us track all of our history and change in real time.**

Conclusions/Learnings:

- **After this activity, I have just enhanced my skills in doing and performing the git commands, both in the bash, and the local machine's CLI or SSH. By doing this, I have made myself more familiar with the different execution, syntax, and commands that was shown and performed in this activity.**