

- 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`.
- Issue the command `ssh-keygen -t rsa -b 4096`. The algorithm is selected using the `-t` option and key size using the `-b` option.

```
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:KMsAgoQu8I92DF2jfhN8WGB9SYTsX8nSdsfE2e5YgBI TIPQC@Q5202-30
The key's randomart image is:
+---[RSA 4096]-----+
|..      o +o.  .o |
|=     = + o.  .o+|
|* . . = + . oo. +.|
|ooo o o.o .E* . =|
|. .* . *S. + . = |
| oo=O+ . . . . |
|. .o.              |
|                  |
+---[SHA256]-----+
TIPQC@Q5202-30 MINGW64 ~
$
```

- 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.
- 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`.

```
TIPQC@Q5202-30 MINGW64 ~
$ ls -la .ssh
total 24
drwxr-xr-x 1 TIPQC 197121    0 Aug 23 09:22 ./
drwxr-xr-x 1 TIPQC 197121    0 Aug 23 09:22 ../
-rw-r--r-- 1 TIPQC 197121 3381 Aug 23 09:31 id_rsa
-rw-r--r-- 1 TIPQC 197121  740 Aug 23 09:31 id_rsa.pub
TIPQC@Q5202-30 MINGW64 ~
$
```

Task 2: Copying the Public Key to the remote servers

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

```
TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id -i ~/.ssh/id_rsa emman@192.168.56.108
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/c/Users/TIPQC/.ssh/id_rsa.pub"
The authenticity of host '192.168.56.108 (192.168.56.108)' can't be established.
ED25519 key fingerprint is SHA256:NtY3B4MkKVV0v0RVIYwLIYkqJ0shd/a0wJAWwogf6U.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? /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 '192.168.56.108 (192.168.56.108)' can't be established.
ED25519 key fingerprint is SHA256:NtY3B4MkKVV0v0RVIYwLIYkqJ0shd/a0wJAWwogf6U.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
emman@192.168.56.108's password:
Permission denied, please try again.
emman@192.168.56.108's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'emman@192.168.56.108'"
and check to make sure that only the key(s) you wanted were added.
```

```

TIPQC@Q5202-30 MINGW64 ~
$ ssh-copy-id -i ~/.ssh/id_rsa emman@192.168.56.110
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/c/Users/TIPQC/.ssh/id_rsa.pub"
The authenticity of host '192.168.56.110 (192.168.56.110)' can't be established.
ED25519 key fingerprint is SHA256:83IVUCQ7xkRV0vTkx/YLAvQ9QP7YcJmoc2I5Fh09bRU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
emman@192.168.56.110's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'emman@192.168.56.110'"
and check to make sure that only the key(s) you wanted were added.

```

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

```

TIPQC@Q5202-30 MINGW64 ~
$ ssh emman@192.168.56.108
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.

```

```

TIPQC@Q5202-30 MINGW64 ~
$ ssh emman@192.168.56.110
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.

```

Reflections:

Answer the following:

- How will you describe the ssh-program? What does it do?
 - SSH, often called Secure Shell or Secure Socket Shell, is a network protocol that provides users, especially system administrators, with a secure way to access a computer across an unsafe network.
- How do you know that you already installed the public key to the remote servers?
 - On your local computer towards authenticating with a remote server without a password.

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*

```
MINGW64:/c/Users/TIPQC
TIPQC@Q5202-30 MINGW64 ~
$
```

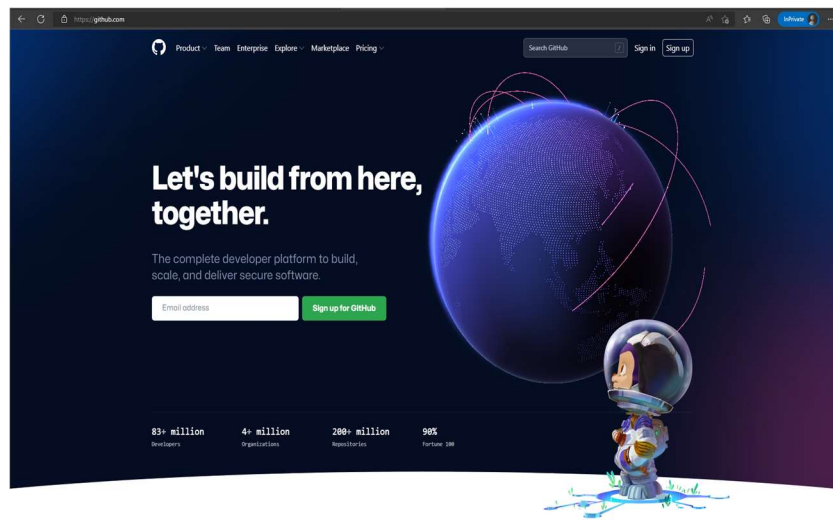
2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/bin/git*.

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

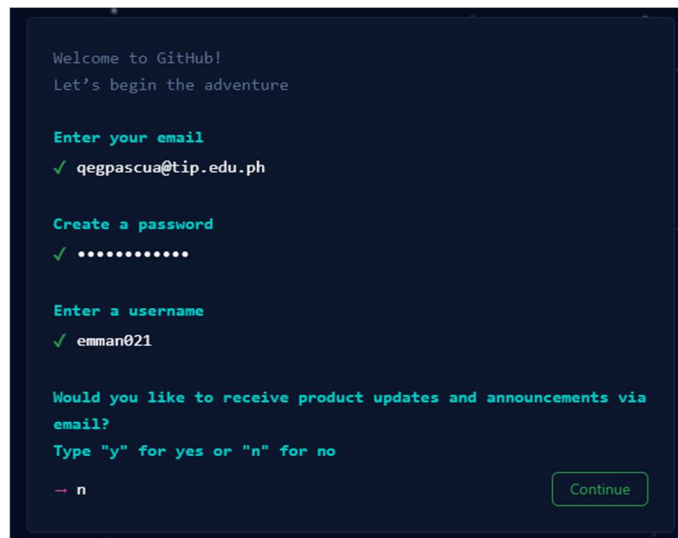
3. The version of git installed in your device is the latest. Try issuing the command *git --version* to know the version installed.

```
TIPQC@Q5202-30 MINGW64 ~
$ git --version
git version 2.37.2.windows.2
```

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.



Welcome to GitHub!
Let's begin the adventure

Enter your email
✓ qegpascua@tip.edu.ph

Create a password
✓

Enter a username
✓ emman021

Would you like to receive product updates and announcements via email?
Type "y" for yes or "n" for no
→ n

Continue

- a. Create a new repository and name it as CPE232_yourname. Check Add a README file and click Create repository.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner * Repository name *

 emman021 / CPE232_emman ✓

Great repository names are short and memorable. Need inspiration? How about [turbo-octo-broccoli](#)?

- 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 / Add new

Title

CPE232

Key

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'

Add SSH 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.

```
TIPQC@Q5202-30 MINGW64 ~  
$ cat .ssh/id_rsa.pub  
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQACXsZmh1wB4Nza080xi3p39NiSrX0J1tUhrPAhlqhOxM4kCJILPpMiw0TEFHbIR6O9gRn9bng7TxPMIqziN6xRn6TeUCWCEp0EQ+8WEvMk6wkNBhKf6S9bVb1k1INRv3mY3GmVmx5XnRZ50j8VMNRb  
pootyjhtPrYK+hdxa/ezk4v510Y1a/73+W9qP/cgr2g3ITCIGrTqmWBrdDhKeETvESYSCmj6q+pncxM7952TMeL/g7G8mJ30f4V1x2wBRPDTm8etGeMk/OnvmQ/HRAF6Tfn5FTVJdMrt1N4SnRfsjN7xqTjSJe67vYeGWl3Z8aBQ7utlyf3kmeiDmzH  
H7YcevCNDN5wEhfwelA4nr573h1623j14ARS+4WvqGYJxy1WWIP0rgB04ecEbMQWKiOIA16FhvT8WZlk+xiYPh/KpxHsKNz8Sj2reooA14EP53rt/HNTN9GFU5aVT4OqbMRFDxg4xNH03H3bsSMKfLa97FAwXQHas97i99Cr8ihhXSgbd/wsHJrkSju  
04XgNgBTJwR/v24vs1NVF+Ia6fInKhgpiAtfXMYhe4yIJQaUn3eDOUih72QQfVL/KRgqCnUiLduqAZ4bkyMZ8rwJ2KCKfTeTs7hRoou00+yeHZ29/bkuxaGKurGwwD+MKdhoaBr5G+ZHTdkU9dm6a1xw== TIPQC@Q5202-30  
  
TIPQC@Q5202-30 MINGW64 ~  
$ |
```

SSH keys / Add new

Title

CPE232

Key

```
AAAAB3NzaC1yc2EAAAADAQABAAQACXsZmh1wB4Nza080xi3p39NiSrX0J1tUhrPAhlqhOxM4kCJILPpMiw0TEFHbIR  
6O9gRn9bng7TxPMIqziN6xRn6TeUCWCEp0EQ+8WEvMk6wkNBhKf6S9bVb1k1INRv3mY3GmVmx5XnRZ50j8VMNRb  
RbpootyjhtPrYK+hdxa/ezk4v510Y1a/73+W9qP/cgr2g3ITCIGrTqmWBrdDhKeETvESYSCmj6q+pncxM7952TMeL/g7G  
BmJJO4Vlx2wBRPDTm8etGeMk/OnvmQ/HRAF6Tfn5FTVJdMrt1N4SnRfsjN7xqTjSJe67vYeGWl3Z8aBQ7utlyf3kmeiDmz  
HH7YcevCNDN5wEhfwelA4nr573h1623j14ARS+4WvqGYJxy1WWIP0rgB04ecEbMQWKiOIA16FhvT8WZlk+xiYPh/KpxH  
sKNz8Sj2reooA14EP53rt/HNTN9GFU5aVT4OqbMRFDxg4xNH03H3bsSMKfLa97FAwXQHas97i99Cr8ihhXSgbd/wsHJr  
kSju04XgNgBTJwR/v24vs1NVF+Ia6fInKhgpiAtfXMYhe4yIJQaUn3eDOUih72QQfVL/KRgqCnUiLduqAZ4bkyMZ8rwJ2KCK  
fTeTs7hRoou00+yeHZ29/bkuxaGKurGwwD+MKdhoaBr5G+ZHTdkU9dm6a1xw== TIPQC@Q5202-30
```

Add SSH key

- 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.

The screenshot shows the GitHub interface for a repository named 'CPE302_yourname' by user 'jvtaylor-cpe'. The repository is public and has 1 watch, 0 stars, and 0 forks. The 'Code' dropdown menu is open, displaying three options: 'Clone' (with a yellow circle highlighting the 'SSH' option), 'Download ZIP', and 'GitHub CLI'. The 'Clone' section shows the SSH link: `git@github.com:jvtaylor-cpe/CPE302_yourname`. Below the link, it states 'Use a password-protected SSH key.' and 'Download ZIP'. The repository's README is visible in the background, showing the title 'CPE302_yourname'.

- f. Issue the command `git clone` followed by the copied link. For example, `git clone git@github.com:emman021/CPE232_emmans.git`. When prompted to continue connecting, type yes and press enter.

```
TIPQC@Q5202-30 MINGW64 ~
$ git clone git@github.com:emman021/CPE232_emmans.git
Cloning into 'CPE232_emmans'...
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.

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

- g. 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`.

```
TIPQC@Q5202-30 MINGW64 ~
$ cd CPE232_emmans

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ ls
README.md

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$
```

- h. Use the following commands to personalize your git.

- i.
- `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`

```
TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ git config --global user.name emman021

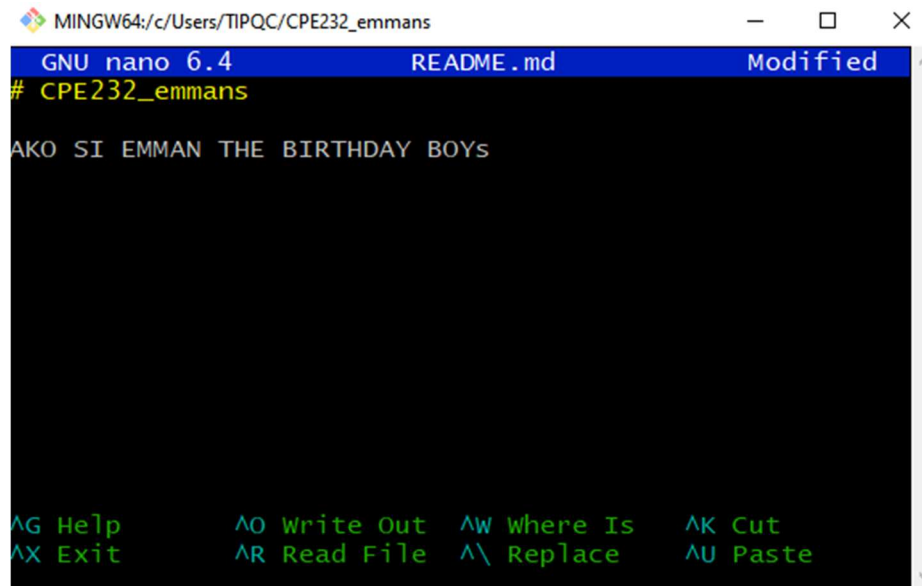
TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ git config --global user.email qegpascua@tip.edu.ph

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ cat ~/.gitconfig
[user]
    name = emman021
    email = qegpascua@tip.edu.ph

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$
```

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

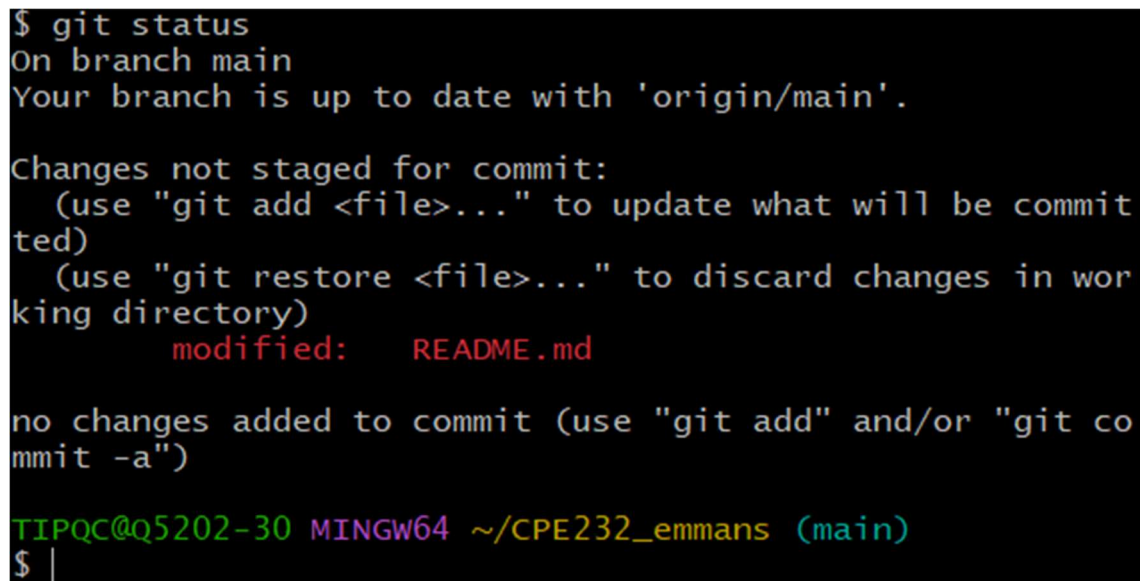


```
MINGW64:/c/Users/TIPQC/CPE232_emmans
GNU nano 6.4 README.md Modified
# CPE232_emmans

AKO SI EMMAN THE BIRTHDAY BOYS

^G Help      ^O Write Out ^W Where Is  ^K Cut
^X Exit      ^R Read File ^\ Replace   ^U Paste
```

- k. 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. The status output does not show any information regarding the committed project history. What is the result of issuing this command?



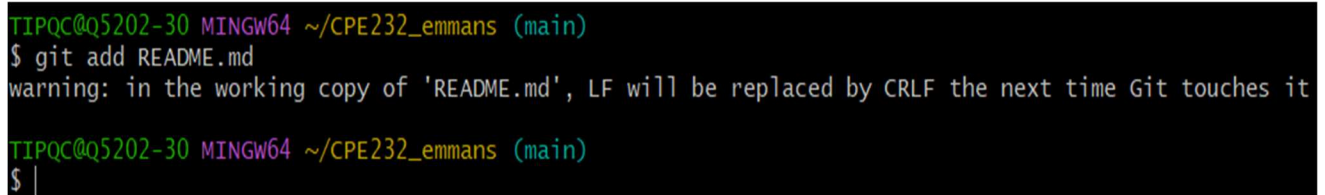
```
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be commit
  ted)
  (use "git restore <file>..." to discard changes in wor
  king directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git co
mmit -a")

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ |
```

- l. Use the command `git add README.md` to add the file into the staging area.



```
TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ git add README.md
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ |
```


- m. 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.

```
TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ git commit -m "First Commit"
[main 7776d99] First Commit
1 file changed, 3 insertions(+), 1 deletion(-)




TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$
```

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


```
TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 292 bytes | 292.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:emman021/CPE232_emmans.git
d919f6a..7776d99 main -> main


TIPQC@Q5202-30 MINGW64 ~/CPE232_emmans (main)
$
```


- o. On the GitHub repository, verify that the changes have been made to README.md by refreshing the page. Describe the README.md file. You can notice how long 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.

 main ▾  1 branch  0 tags

[Go to file](#) [Add file ▾](#) [Code ▾](#)


 emman021 First Commit

7776d99 2 minutes ago  2 commits

 README.md

First Commit

2 minutes ago

README.md 

CPE232_emmans

AKO SI EMMAN THE BIRTHDAY BOYS

Reflections:

Answer the following:

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

- Ansible operates by connecting to your nodes and sending them little programs known as "Ansible modules." These programs were created as resource models for the ideal system state. After that, Ansible runs these modules (by default over SSH) and then deletes them.

4. How important is the inventory file?

- The inventory file defines the hosts and groups of hosts upon which commands, modules, and tasks in a playbook operate. The file can be in one of many formats depending on your Ansible environment and plugins.

Conclusions/Learnings:

- So, I'm satisfied that I finished this activity, and I can say that it's a bit difficult because, first of all, I'm slow on the PC I'm using, and then it slows down, even more, when it's open at the same time, like the virtual box and the Ubuntu OS. We have also included the tabs in Google Chrome, so it is difficult for me to do it. But even so, I was happy with what I did because, through bash, I was able to transfer a message to GitHub, and I also learned how to clone by copying their link, and finally, I also learned that you first need to properly configure each server and the workstation because it will be a big problem if they are not connected.

Honor Pledge:

"I affirm that I shall not give or receive any unauthorized help on this hands-on activity and that all work shall be my own. "