

LAUNCH GIT

Git has two modes of use – a **bash scripting shell** (or command line) and a **graphical user interface** (GUI).

1. To launch **Git Bash**, click on the Git bash icon to see the unix-based command line.



```
MINGW64:/c/Users/chyecheah.t
chyecheah.t@APIIT-LP-0581 MINGW64 ~ (master)
$ |
```

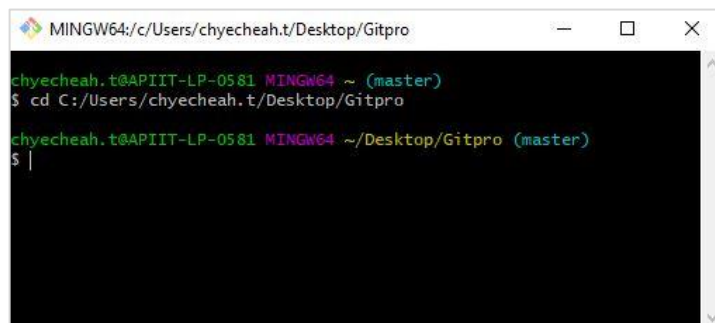
2. Configure your local Git installation to use your GitHub credentials by entering the following:

```
git config --global user.name "github_username"
git config --global user.email "email_address"
```

Note: Replace **github_username** and **email_address** with your GitHub credentials.

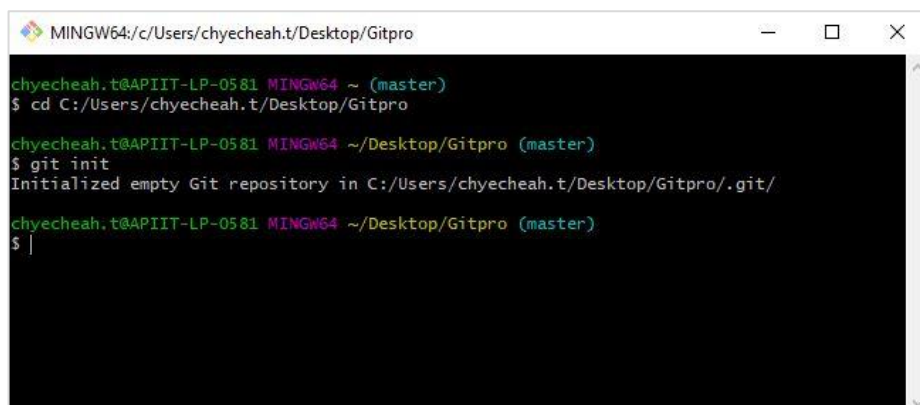
INITIALIZING A REPOSITORY

1. Create a folder (anywhere you like) to initialize Git Project.
2. Here we have created a folder “Gitpro” in the Desktop folder.



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~ (master)
$ cd C:/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

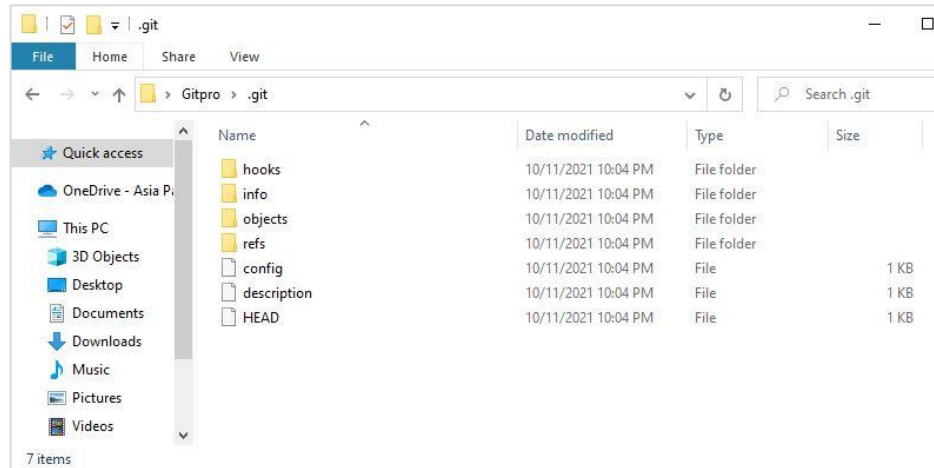
3. Use the command “**git init -b main**” to setup your home base and initialize your git. This will setup your local repository.



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~ (master)
$ cd C:/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git init
Initialized empty Git repository in C:/Users/chyecheah.t/Desktop/Gitpro/.git/
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

The **.git** folder will contain details of every single change made to the code base. All snapshots of the modifications will be recorded in this folder like a database, which makes it possible to undo the changes and rollback to the desired version of the code.

The **.git** folder is hidden to prevent accidental deletion or modification of the folder. The version history of the code base will be lost if this folder is deleted.



PUSHING FILES TO THE LOCAL REPOSITORY (WORKING DIRECTORY)

Let's add a new file and start the tracking by Git

1. Create a Text file and save it in the Gitpro directory. Run the echo command as shown below. This command echoes the text provided (This is a sample text file) to the command console. But, since the command is using the **redirection operator (>)**, this operator tells Windows to instead create a file with the text (sample.txt).

```
echo This is a sample text file > sample.txt
```

2. Now you need to add the changes into the directory. Use this command: **git add .** (this will add all the new files or modified files). You also can use **git add filename**
3. After add command, you have submit the file to the staging. But still is not in your repository. To submit the file into your repository, you should use the command **commit** :
git commit -m "some text"
 - o **-m** allow you to add comments

```
MINGW64/c:/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~ (master)
$ cd C:/Users/chyecheah.t/Desktop/Gitpro

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ echo This is a sample text file > sample.txt

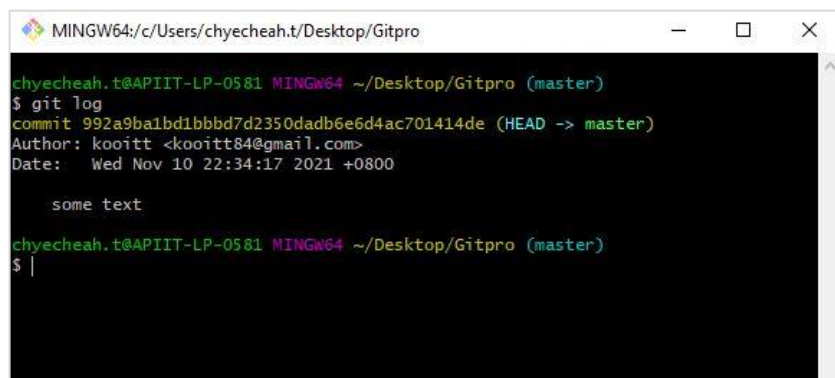
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git add .
warning: LF will be replaced by CRLF in sample.txt.
The file will have its original line endings in your working directory

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git commit -m "some text"
[master (root-commit) 992a9ba] some text
1 file changed, 1 insertion(+)
create mode 100644 sample.txt

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

The commonly accepted rules on how to write a git commit message are:

- Limit the subject line to 50 characters.
 - Capitalize only the first letter in the subject line.
 - Don't put a period at the end of the subject line.
 - Put a blank line between the subject line and the body.
 - Wrap the body at 72 characters.
 - Use the imperative mood.
 - Describe what was done and why, but not how.
4. To check what has happened: The file was committed to the local repository. Use “git log” command to see the progress. Each commit has a unique ID.

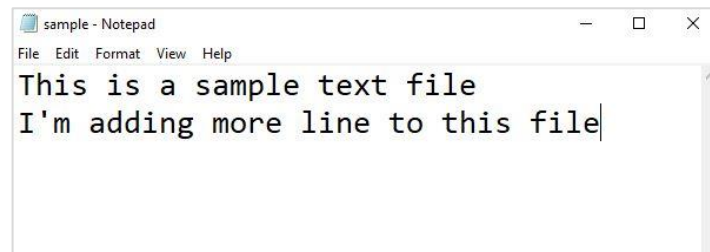


```
MINGW64:/c:/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git log
commit 992a9ba1bd1bbbd7d2350dadb6e6d4ac701414de (HEAD -> master)
Author: kooitt <kooitt84@gmail.com>
Date:   Wed Nov 10 22:34:17 2021 +0800

    some text

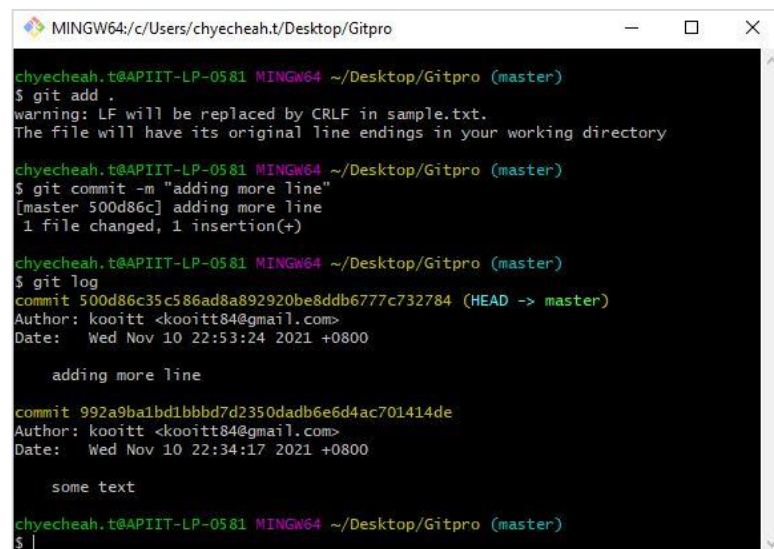
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

5. Let's add some changes in the text file:



```
sample - Notepad
File Edit Format View Help
This is a sample text file
I'm adding more line to this file|
```

6. You do the changes in the file and then you need to add and commit them into your local repository. Use the below commands:



```
MINGW64:/c:/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git add .
warning: LF will be replaced by CRLF in sample.txt.
The file will have its original line endings in your working directory

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git commit -m "adding more line"
[master 500d86c] adding more line
1 file changed, 1 insertion(+)

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git log
commit 500d86c35c586ad8a892920be8ddb6777c732784 (HEAD -> master)
Author: kooitt <kooitt84@gmail.com>
Date:   Wed Nov 10 22:53:24 2021 +0800

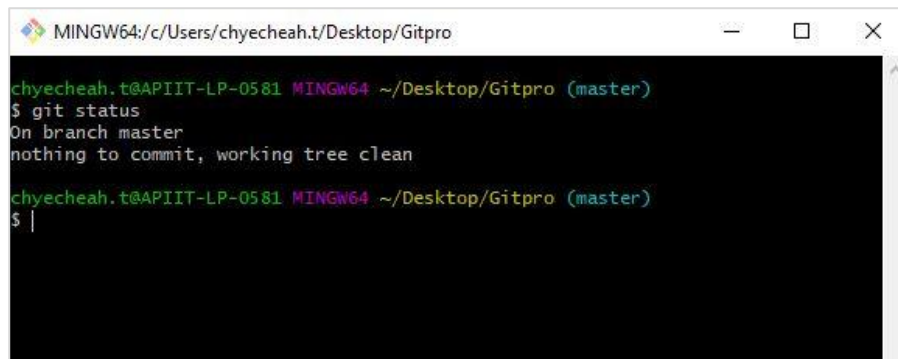
    adding more line

commit 992a9ba1bd1bbbd7d2350dadb6e6d4ac701414de
Author: kooitt <kooitt84@gmail.com>
Date:   Wed Nov 10 22:34:17 2021 +0800

    some text

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

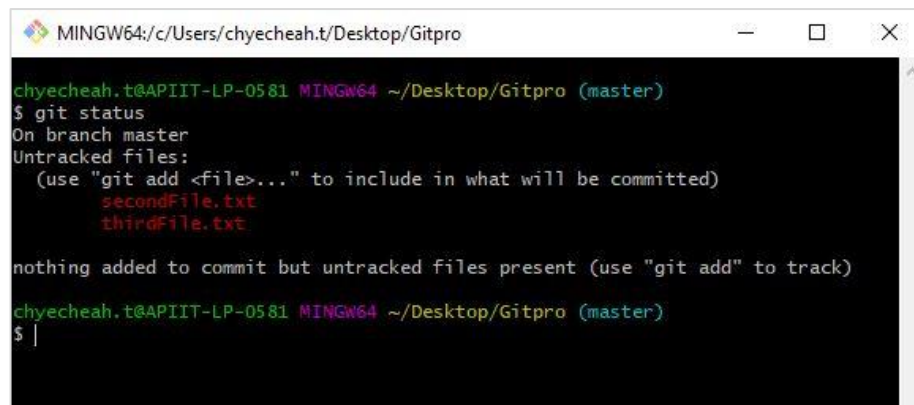
7. Use the command "git status". **Git status:** will report back to us the difference between our working directory, staging index and our repository



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
nothing to commit, working tree clean

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

8. We add 2 new files into our directory (Gitpro). They are secondFile.txt and thirdFile.txt. Now we use again the "git status".

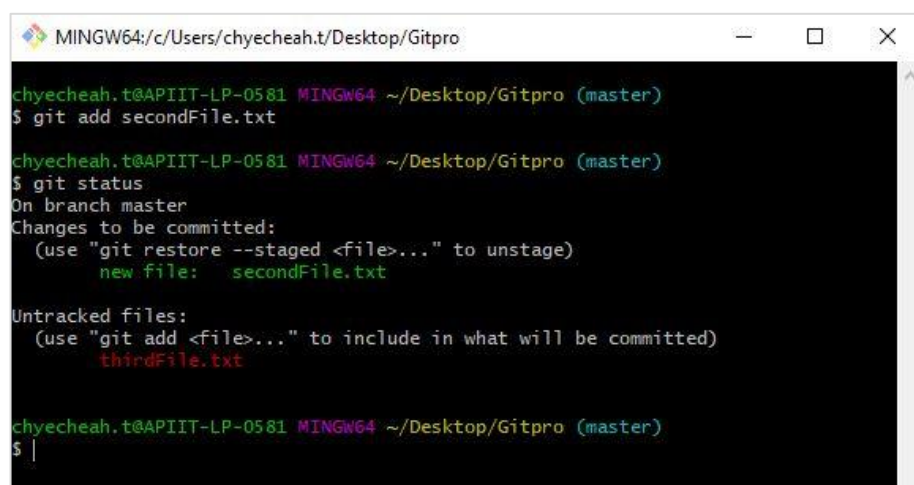


```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        secondFile.txt
        thirdFile.txt

nothing added to commit but untracked files present (use "git add" to track)

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

9. Untracked files: it means Git is detecting some new files in working directory. Add the second file and then use "git status".



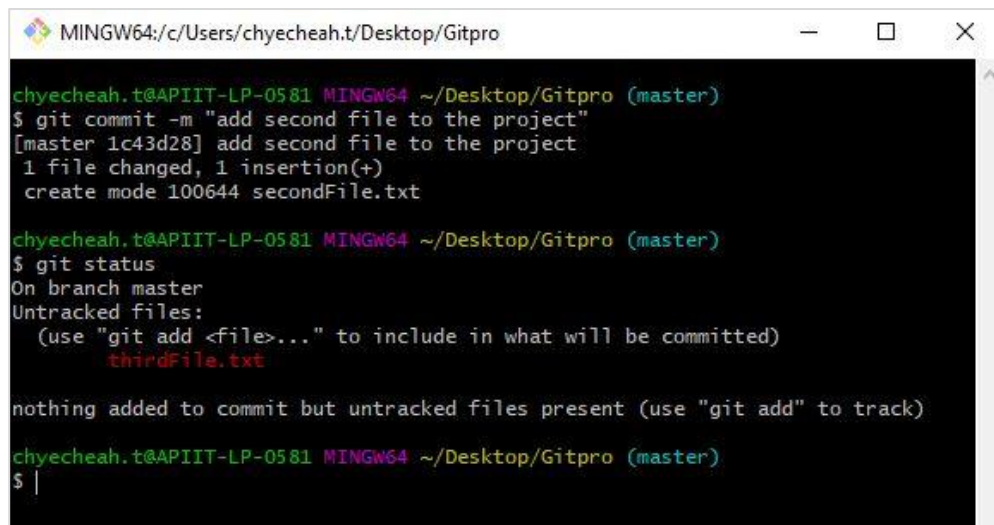
```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git add secondFile.txt

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   secondFile.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        thirdFile.txt

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

10. Commit the second file. And then use "git status".



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro

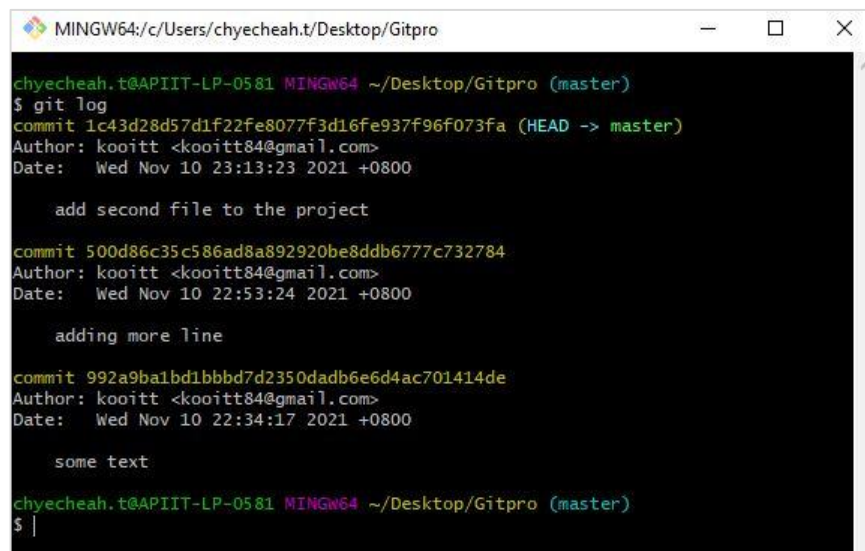
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git commit -m "add second file to the project"
[master 1c43d28] add second file to the project
1 file changed, 1 insertion(+)
create mode 100644 secondFile.txt

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        thirdFile.txt

nothing added to commit but untracked files present (use "git add" to track)

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

11. Now you can use git log.



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git log
commit 1c43d28d57d1f22fe8077f3d16fe937f96f073fa (HEAD -> master)
Author: kooitt <kooitt84@gmail.com>
Date:   Wed Nov 10 23:13:23 2021 +0800

    add second file to the project

commit 500d86c35c586ad8a892920be8ddb6777c732784
Author: kooitt <kooitt84@gmail.com>
Date:   Wed Nov 10 22:53:24 2021 +0800

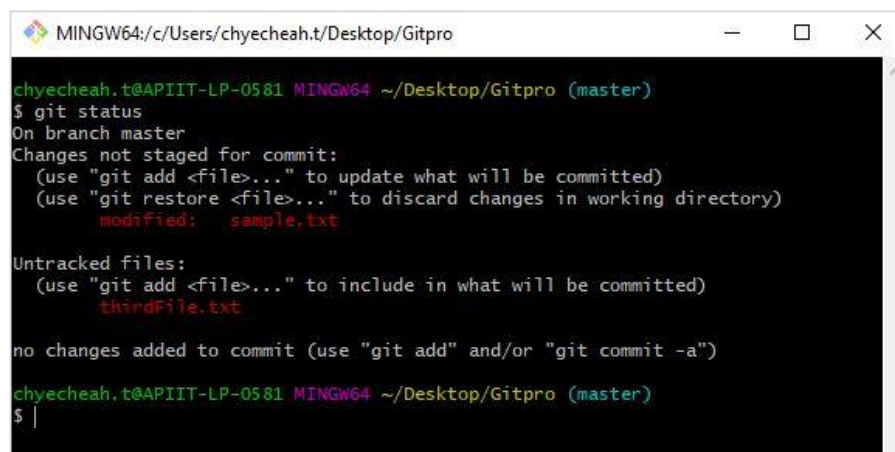
    adding more line

commit 992a9ba1bd1bbbd7d2350dadb6e6d4ac701414de
Author: kooitt <kooitt84@gmail.com>
Date:   Wed Nov 10 22:34:17 2021 +0800

    some text

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

12. Modify some files. Here we are modifying the sample.txt. Then we type this command: git status



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro

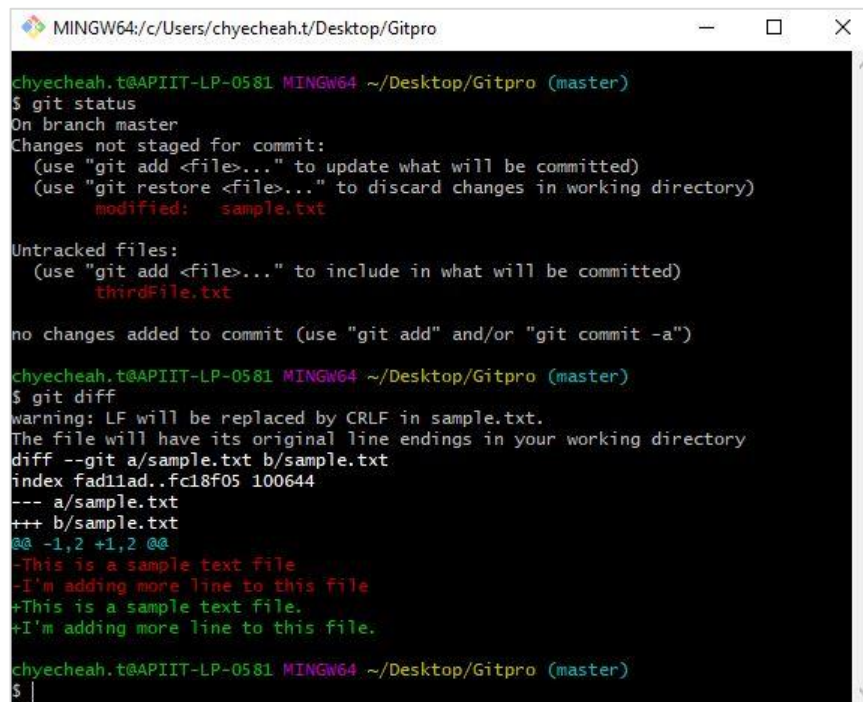
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   sample.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        thirdFile.txt

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

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```


13. To see what changes have been made, you can use this command: **git diff**. It compares the two files from working and repository directory and show the changes. The file with - - - is the one in the repository. The file with +++ is the one in the new version (working). **Git diff** only look at the file in working directory.



```
MINGW64/c/Users/chyecheah.t/Desktop/Gitpro

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   sample.txt

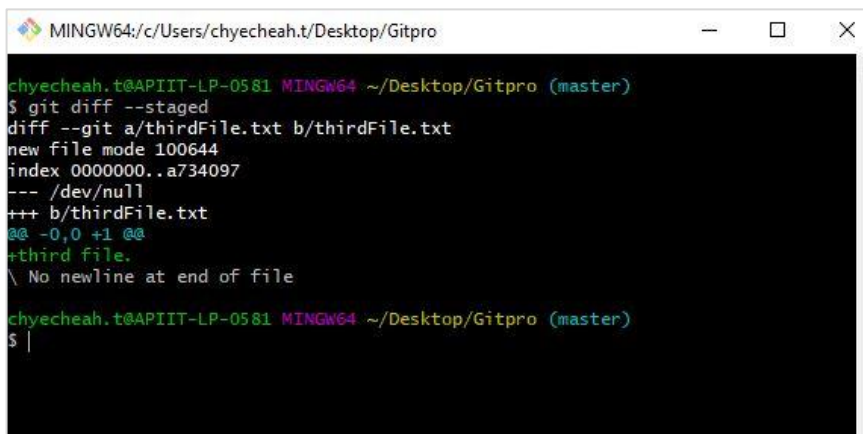
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        thirdFile.txt

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

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git diff
warning: LF will be replaced by CRLF in sample.txt.
The file will have its original line endings in your working directory
diff --git a/sample.txt b/sample.txt
index fad11ad..fc18f05 100644
--- a/sample.txt
+++ b/sample.txt
@@ -1,2 +1,2 @@
-This is a sample text file
-I'm adding more line to this file
+This is a sample text file.
+I'm adding more line to this file.

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$
```

14. To see and compare the changes of the file in stage directory: you should use "git diff -- staged".

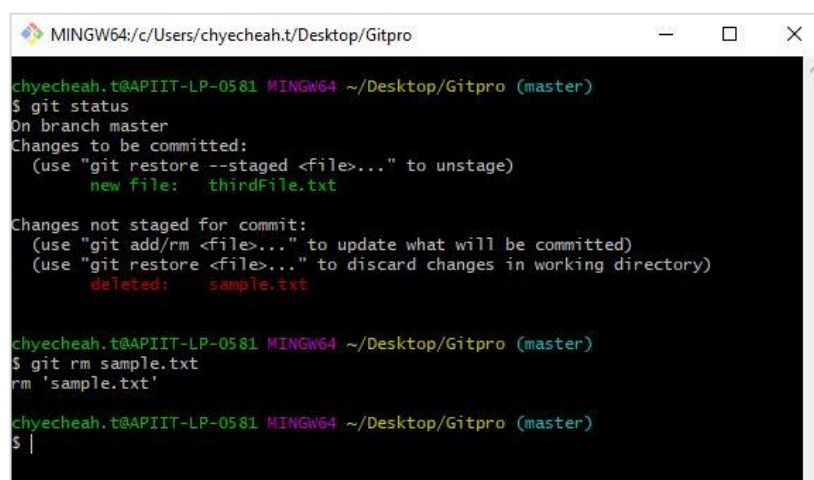


```
MINGW64/c/Users/chyecheah.t/Desktop/Gitpro

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git diff --staged
diff --git a/thirdFile.txt b/thirdFile.txt
new file mode 100644
index 0000000..a734097
--- /dev/null
+++ b/thirdFile.txt
@@ -0,0 +1 @@
+third file.
\ No newline at end of file

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$
```

15. To delete the files from repository. We need to use: **git rm filename**



```
MINGW64/c/Users/chyecheah.t/Desktop/Gitpro

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   thirdFile.txt

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    sample.txt

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git rm sample.txt
rm 'sample.txt'

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$
```

16. To rename the files, we need to use: **git mv second-file.txt secondary-file.txt**

SHORTCUTS:

1. Try some short cut:
 - o `git log --oneline`
2. To commit several files together with one command, you can type:
 - o `git commit -am "Msg"` (-a will commit all the files together).
3. To know what have been modify :
 - o `git status -s`

Summary

`git init <Project>` : Project New repository (Working Directory)

`git add .` (name of the file) : add the files to the staging

`git commit -m "message"` : snapshot (send to local repository)

`Git status` (status of files)

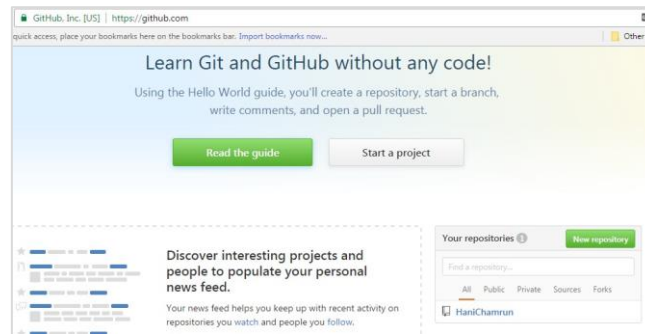
`Git log` (commit history)

`Git diff` (view differences)

`Git diff -- staged` (differences in staging)

REMOTE REPOSITORY (GIT HUB)

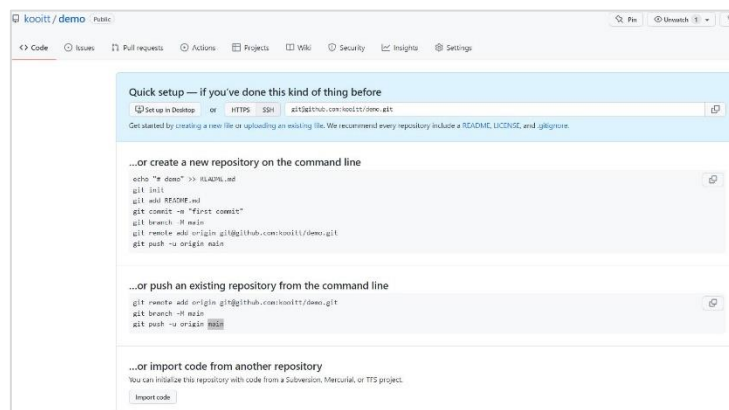
1. Browse to the official GitHub website: [www.github.com](https://github.com) to setup Git Hub account.



2. In the upper right corner, next to your avatar or identicon, click + and then select New repository. Select **Initialize this repository with a README.**

A screenshot of the "Create a new repository" form on GitHub. The form has a title "Create a new repository" and a subtitle "A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository." Below this, there are two input fields: "Owner" with a dropdown menu showing "kooitt" and "Repository name" with a text input showing "demo". A message below these fields says "Great repository names are short and memorable. Need inspiration? How about cautious-octo-fiesta?". There's a "Description (optional)" text area. Below that, there are two radio buttons for "Public" (selected) and "Private". A section titled "Initialize this repository with:" follows, with a note "Skip this step if you're importing an existing repository." and three checkboxes: "Add a README file", "Add .gitignore", and "Choose a license". A blue information box at the bottom says "You are creating a public repository in your personal account." and a green "Create repository" button is at the bottom.

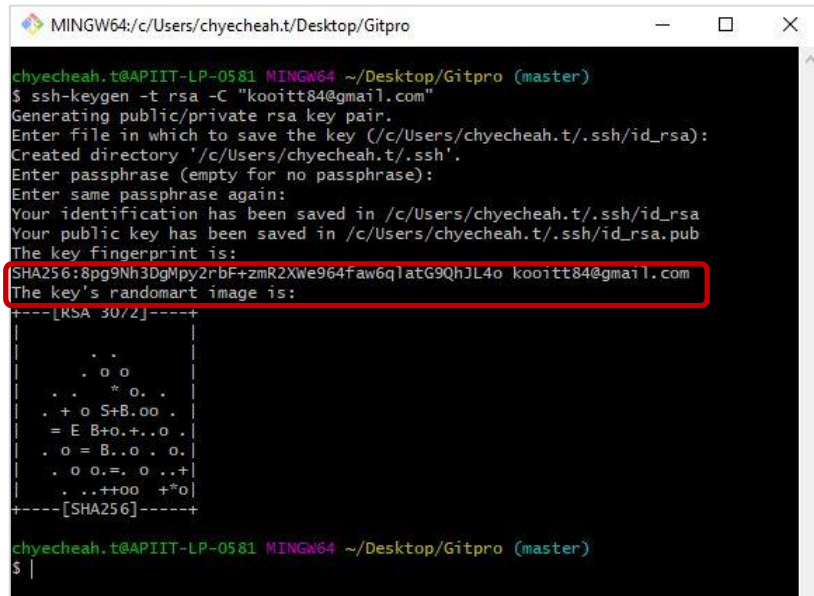
3. This is how an empty repository looks like:



Git workshop

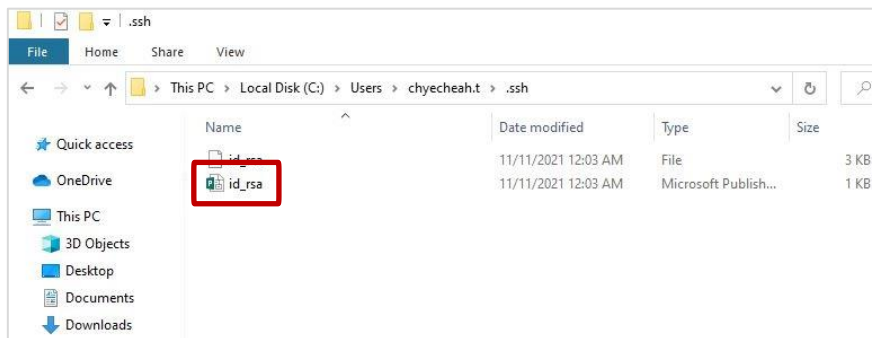
4. Now we have to create SSH Key. Type: `ssh-keygen -t rsa -C "your email"`

- `-t` will show the type of SSH (here is RSA)
- `-C` will show the comments
- Press Enter, then enter your password

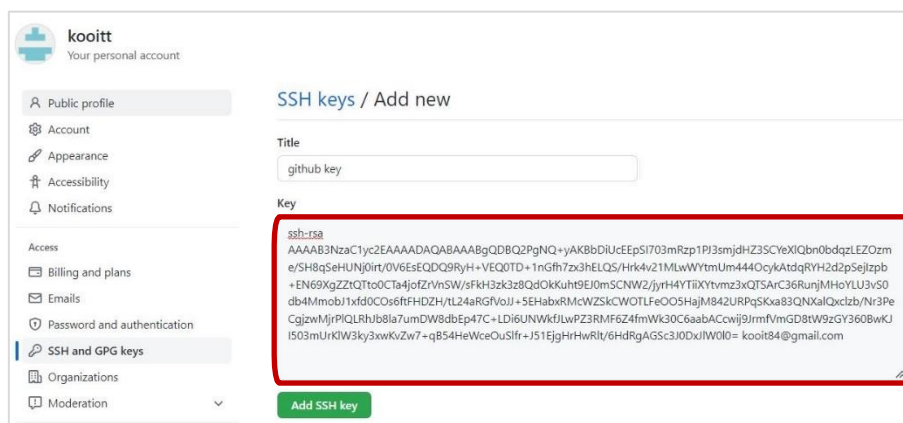


```
MINGW64:/c:/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ ssh-keygen -t rsa -C "kooitt84@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c:/Users/chyecheah.t/.ssh/id_rsa):
Created directory '/c:/Users/chyecheah.t/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c:/Users/chyecheah.t/.ssh/id_rsa
Your public key has been saved in /c:/Users/chyecheah.t/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:8pg9Nh3DgMpy2rbF+zmR2XWe964faw6qlatG9QhJL4o kooitt84@gmail.com
The key's randomart image is:
+---[RSA 3072]---+
  .   .   .
  .   .   .
  .   .   .
  .   .   .
  .   .   .
  .   .   .
  .   .   .
  .   .   .
  .   .   .
  .   .   .
+---[SHA256]-----+
```

5. Now go to the directory that SSH has been created. Here we have the private key and public key.

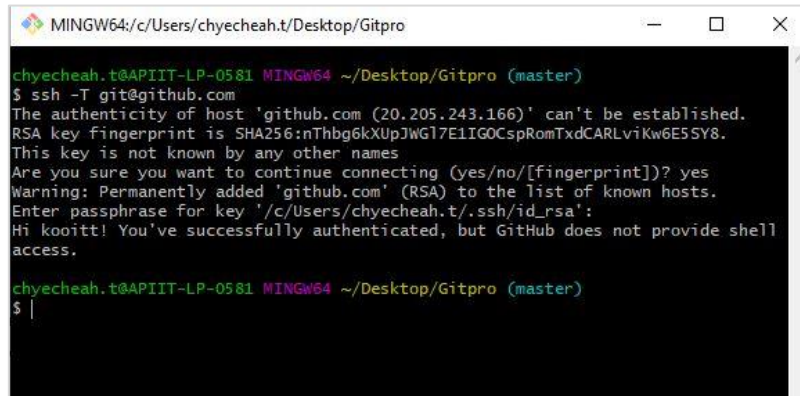


6. Open the public key. Copy the text inside and paste it to the below part.



Git workshop

- Now to test that the SSH key is working. Type: `ssh -T git@github.com`
- Then enter **yes**, and then type your password





```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ ssh -T git@github.com
The authenticity of host 'github.com (20.205.243.166)' can't be established.
RSA key fingerprint is SHA256:nThbg6kXUpJWG17E1IGOCspRomTxdCARLviKw6E5SY8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (RSA) to the list of known hosts.
Enter passphrase for key '/c/Users/chyecheah.t/.ssh/id_rsa':
Hi kooitt! You've successfully authenticated, but GitHub does not provide shell
access.

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

- We can test our files from local repository to the remote. Copy the SSH address and paste it into git bash as below:

Quick setup — if you've done this kind of thing before

 Set up in Desktop or ☐ HTTPS ☒ SSH `git@github.com:kooitt/demo.git` 

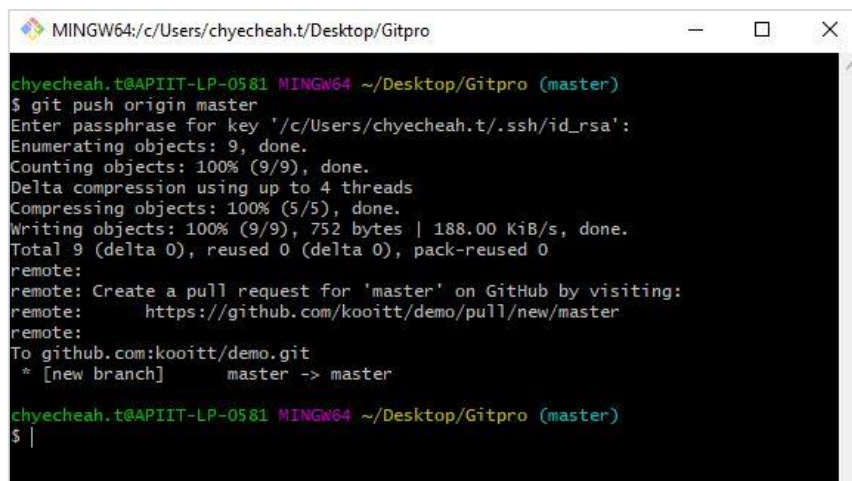
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.



```
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git remote add origin git@github.com:kooitt/demo.git

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

- Now you can use push to send the files to the remote repository: **git push origin main**



```
MINGW64:/c/Users/chyecheah.t/Desktop/Gitpro
chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ git push origin master
Enter passphrase for key '/c/Users/chyecheah.t/.ssh/id_rsa':
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (9/9), 752 bytes | 188.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/kooitt/demo/pull/new/master
remote:
To github.com:kooitt/demo.git
 * [new branch]      master -> master

chyecheah.t@APIIT-LP-0581 MINGW64 ~/Desktop/Gitpro (master)
$ |
```

- Now go to the git hub website and refresh

Reference

Git installation: <https://phoenixnap.com/kb/how-to-install-git-windows>

How to write a Git commit message: <https://www.theserverside.com/video/Follow-these-git-commit-message-guidelines>