Git Push and Pull

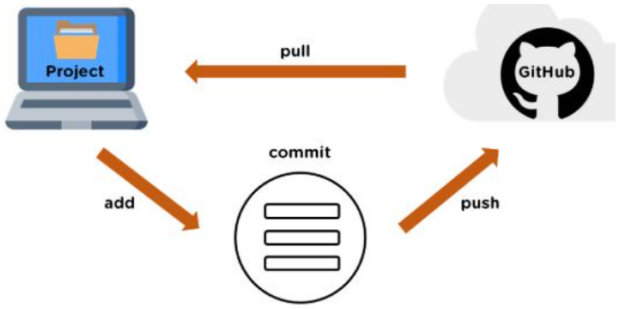
Topics Covered:

- Git Push.
- Steps involved in git push operation.
- Git Pull.
- Steps involved in git pull operation.

Topics in Detail:

Git Push Command

- The **Git Push Command** transfers the commits from local repository to remote repository.
- After making all the modifications, **push command** will help to **share** the modifications with the **remote** team members.



Steps involved in git push operation:

We can create repositories in two ways

- Using HTTPS
- Using SSH

Creating repositories using HTTPS

- Open Git Bash and configure it with username and email ID.

```
$ git config –global user.name “username”

$ git config –global user.email “mailid”

$ git config --list
```

```
[redacted]-S9VIU5AR MINGW64 ~ (master)
$ git config --global user.name "[redacted]"

[redacted]-S9VIU5AR MINGW64 ~ (master)
$ git config --global user.email [redacted]@gmail.com

[redacted]-S9VIU5AR MINGW64 ~ (master)
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.email=[redacted]@gmail.com
user.name=[redacted]
```

- Check the current working repository.

```
$ pwd
```

```
[redacted]-S9VIU5AR MINGW64 ~ (master)
$ pwd
/c/Users/FATHIMA
```

- In the working directory, create a repository

```
$ mkdir Git_Demo
$ cd Git_Demo
$ pwd
```

- Now, the Git_Demo repository will be empty. So, let's create a folder in the working repository.

```
$ mkdir FirstRepo
$ cd FirstRepo
$ pwd
```

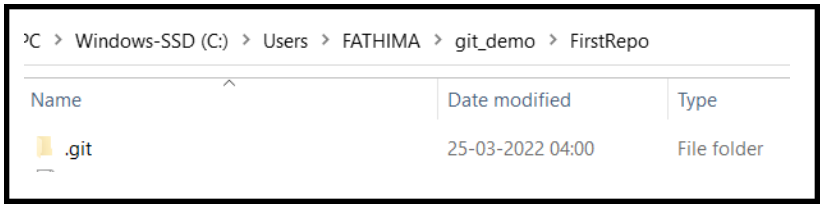
```
LAPTOP-S9VIU5AR MINGW64 ~ (master)
$ mkdir git_demo

APTOP-S9VIU5AR MINGW64 ~ (master)
$ cd git_demo

LAPTOP-S9VIU5AR MINGW64 ~/git_demo (master)
$ pwd
/c/Users/FATHIMA/git_demo

PTOP-S9VIU5AR MINGW64 ~/git_demo (master)
$ mkdir FirstRepo

LAPTOP-S9VIU5AR MINGW64 ~/git_demo (master)
$ cd FirstRepo

LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ pwd
/c/Users/FATHIMA/git_demo/FirstRepo
```

- Initialize a repository to the created folder.

```
$ git init
```

```
APTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ git init
Initialized empty Git repository in C:/Users/FATHIMA/git_demo/FirstRepo/.git/
```

  - When a git repository is created for the first time, it will create a **branch, and** it is called **master**.
  - If you navigate to the folder, there will be a hidden **.git** folder. This folder is created when we **initialize** the repository.
  - Inside that .git folder there will be several directories and configurations.



- We are going to create two notepads and try committing them one by one
- First Notepad

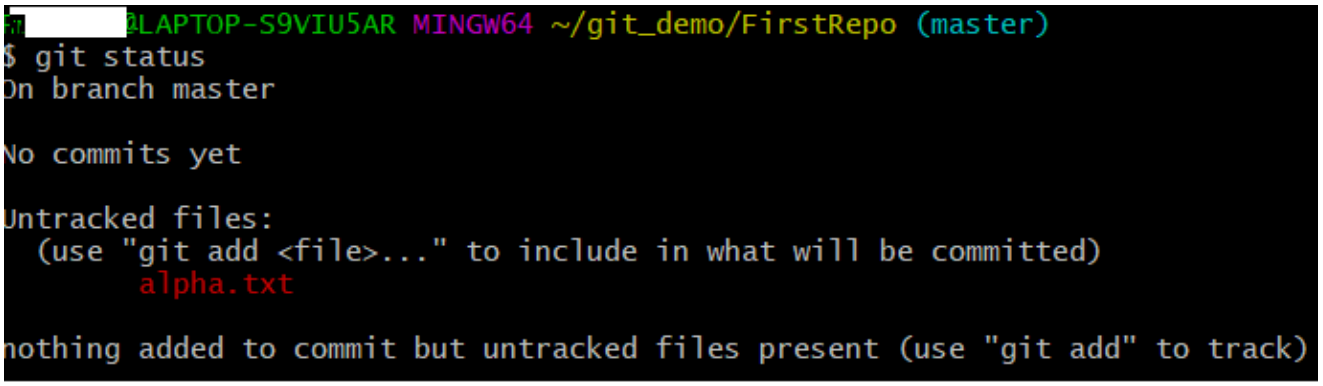
```
$ touch alpha.txt
$ notepad alpha.txt
```

Now a notepad opens on the screen where we can type anything. At last save and close the notepad.

- Check the status of the file created using the following command.

```
$ git status
```

  - **No commits yet** - So far no file is committed in the repository.
  - **Untracked files** names are highlighted in **red** color.



- To **track** untracked files, use the **add** command.
- We can specify the **exact file name** or the following command.

```
$ git add .
```

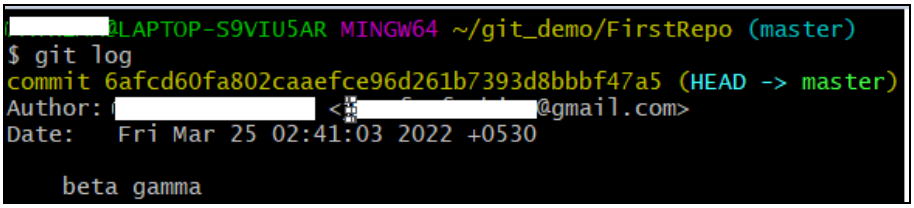
- **add .** command will add all the files which are modified and also untracked.
- If we try checking the status after the **add** command, the **red** color file names change to **green**.

- Then, commit the file.

```
$ git commit -m “alpha”
```

- Check the status again, the work repository will clean.
- Since there was only one file and that was committed already in the previous step, it shows **nothing to commit** status.
- To check all the information regarding the commits made, use the following command

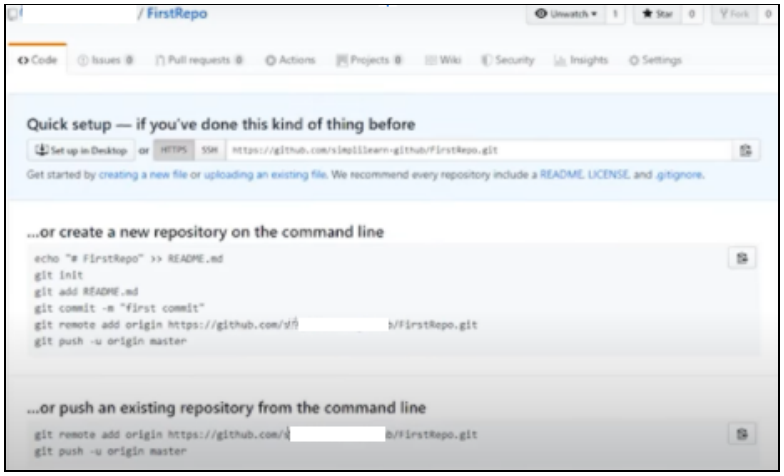
```
$ git log
```



Displays

- Commit ID
- Author Name
- E-Mail ID
- Date
- Time

- Now let’s push the alpha.txt on GitHub. Open your GitHub account and create a new repository.
- Copy the “git remote add origin” URL



- Paste the copied URL in the Git Bash.

```
@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ git remote add origin https://github.com/[redacted]/FirstRepo.git
```

`$ git remote -v`

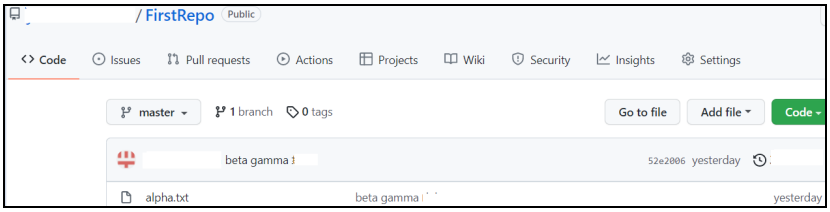
```
@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ git remote -v
origin  https://github.com/[redacted]/FirstRepo.git (fetch)
origin  https://github.com/[redacted]/FirstRepo.git (push)
```

- Let's now push the content to the repository.

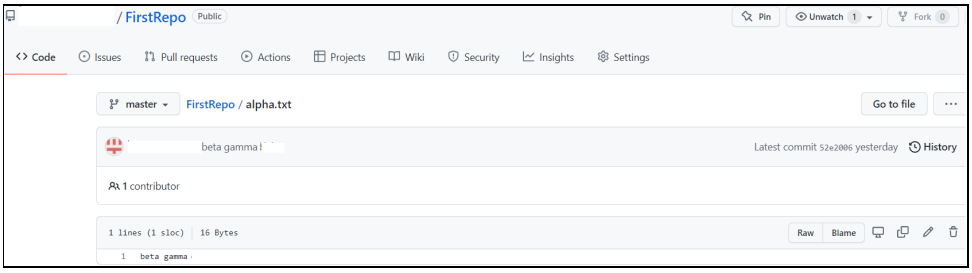
`$ git push -u origin master`

```
@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ git push -u origin master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (7/7), 492 bytes | 246.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/[redacted]/FirstRepo.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

- The repository is created on the server and the content is pushed to that repository.
- This **repository** links both the masters in the local repository and the server.
- Now to view the commits, refresh the GitHub page.
- Each commit has a **HashID** and details about it.



- Open the notepads and check the contents inside.



Creating Repositories using SSH

- **Checking for existing SSH Keys**
  - To Check whether there are any SSH Keys existing, use the following command

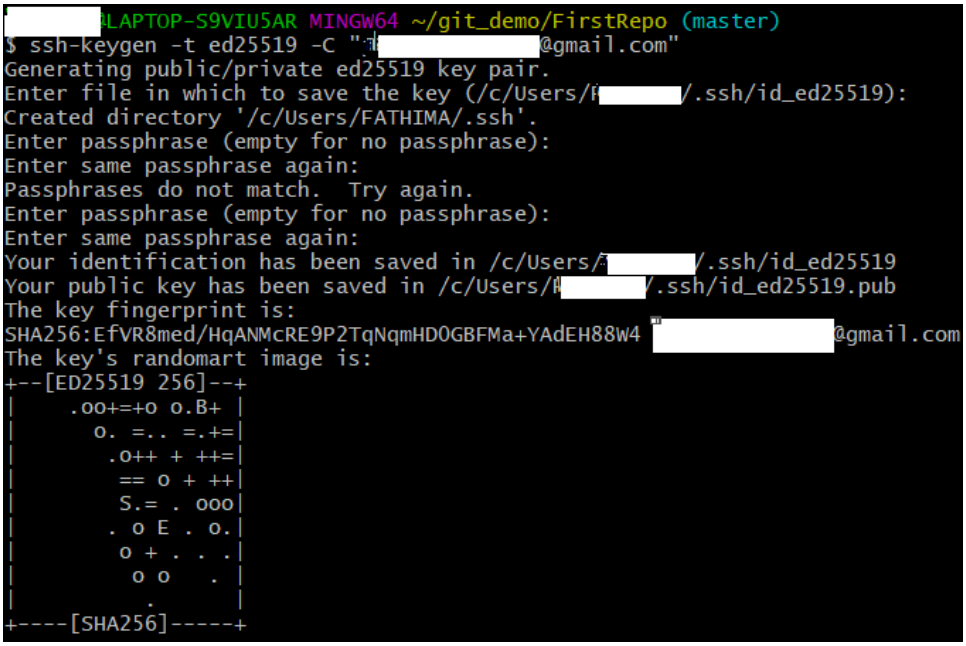
```
$ ls -al ~/.ssh
```

  - If the SSH keys exist already, it will listed as follows

```
[redacted]LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ ls -al ~/.ssh
total 10
drwxr-xr-x 1 [redacted] 197121   0 Mar 25 03:12 ./
drwxr-xr-x 1 [redacted] 197121   0 Mar 25 03:08 ../
-rw-r--r-- 1 [redacted] 197121 464 Mar 25 03:12 id_ed25519
-rw-r--r-- 1 [redacted] 197121 106 Mar 25 03:12 id_ed25519.pub
```

    We can jump to **copy the public SSH Key** step.
  - Or If we get **No such file or directory** as the result of running the above command, we need to create a new SSH Key.
- **Generate a new set of SSH Keys**
  - To generate new SSH Keys, use the following command

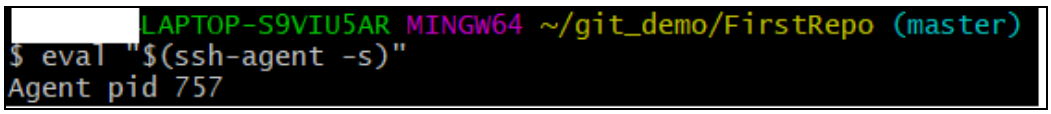
```
$ ssh-keygen  -t ed25519 -C your@email.com
```



- SSH Keys are generated as a set of **public and private keys**. One should **never reveal the Private key** and should **only use the Public Key** for GitHub Authentication.

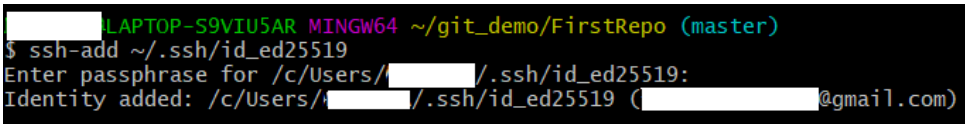
- **Adding SSH Keys to ssh-agent**
  - To make sure whether the ssh-agent is running or not, use the following command

```
$ eval "$(ssh-agent -s)"
```



  - To add the private key to ssh-agent, use the following command

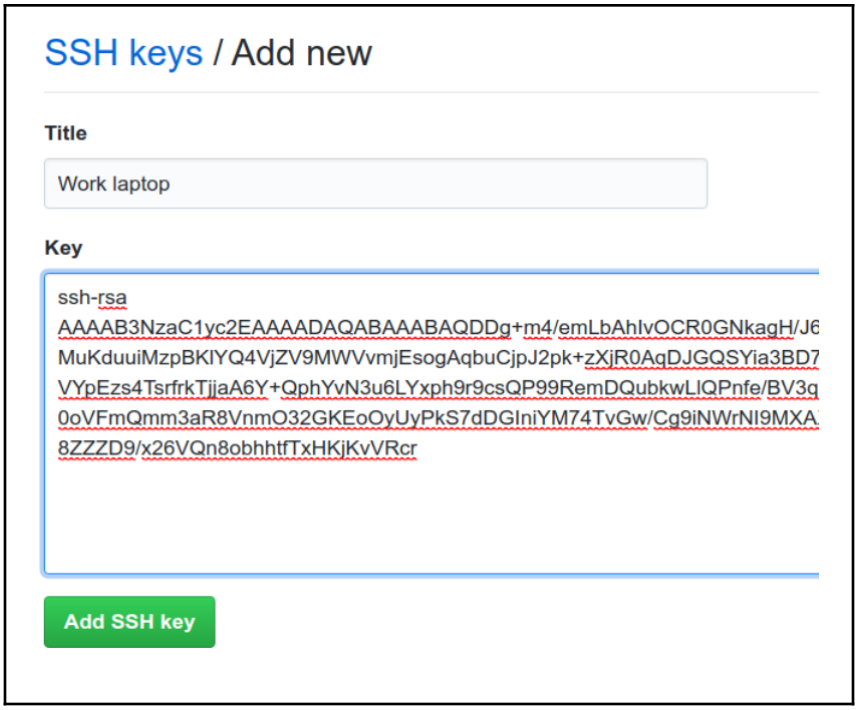
```
$ ssh-add ~/.ssh/id_ed25519
```



- **Copy the public SSH key**
  - To copy the public SSH Key, use the following command

```
$ clip < ~/.ssh/id_ed25519.pub
```

- **Add the Public SSH Key to GitHub**
  - Go to GitHub **settings** page and click the "**New SSH Key**" button
  - Enter any relative title and paste the public key



- Finally, test the authentication with the below command.

$ ssh -T git@github.com

```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ ssh -T git@github.com
The authenticity of host 'github.com (13.234.210.38)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
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' (ED25519) to the list of known hosts.
Hi [redacted]! You've successfully authenticated, but GitHub does not provid
e shell access.
```

- To create a repository follow the same set of steps just like HTTPS except in the place of "**git remote add origin**" URL command.

```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ git remote add origin https://github.com/[redacted]/FirstRepo.git
```

- Replace that with the "**git remote set-url origin**" URL command.

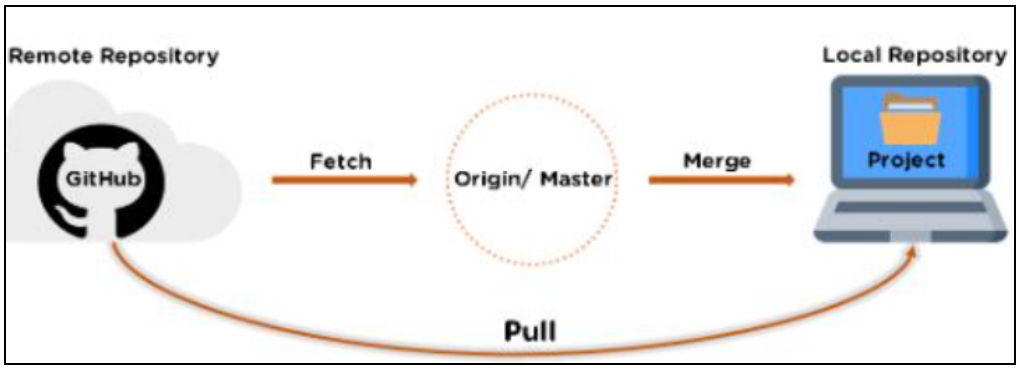
```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/FirstRepo (master)
$ git remote set-url origin git@github.com:[redacted]/FirstRepo.git
```

- Followed by the commands

```
$ git add .
$ git commit -m "Commit a file"
$ git push
```

Git Pull Request

- The **Git Pull Command** is used to **fetch** and **merge** the changes in code from the remote repository to the local repository.
- It is a combination of two commands.
    - **Git Fetch** - Downloads content from the remote repository.
    - **Git Merge** - Combines multiple commits into a single branch.



- In the working directory, create a repository

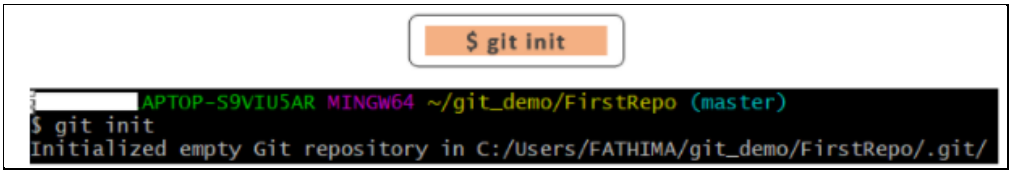
```
$ mkdir Git_Demo
$ cd Git_Demo
$ pwd
```

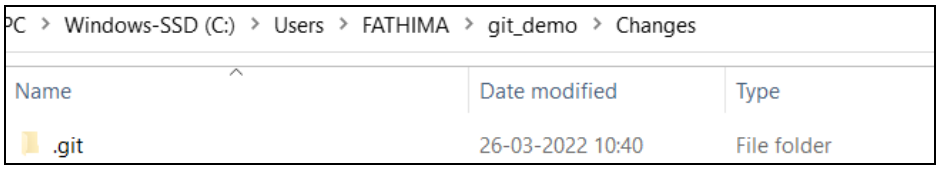
- Now, the Git_Demo repository will be empty. So, Let's create a folder in the working repository.

```
$ mkdir Changes
$ cd Changes
$ pwd
```

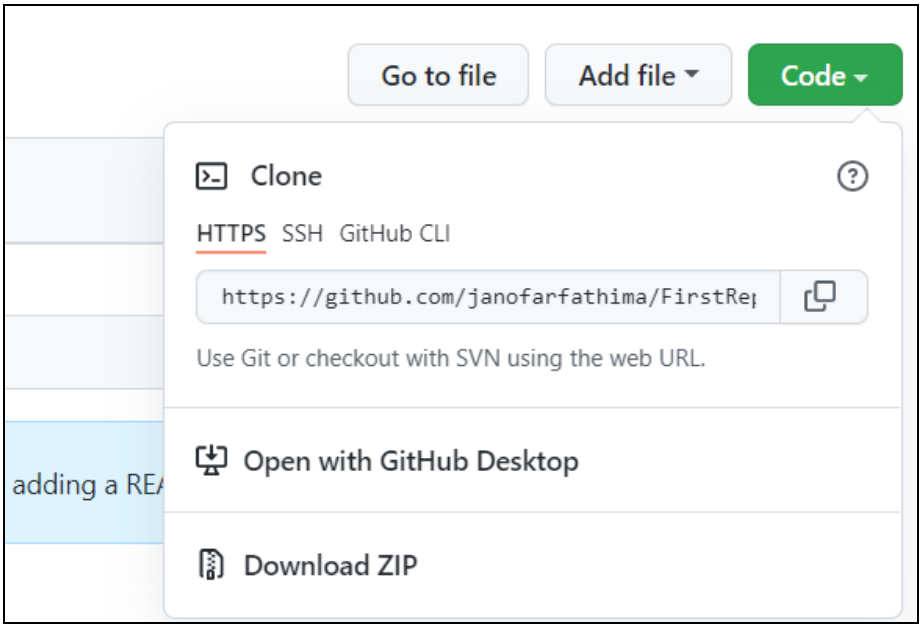
```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~ (master)
$ mkdir git_demo

[redacted]@LAPTOP-S9VIU5AR MINGW64 ~ (master)
$ cd git_demo

[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo (master)
$ pwd
/c/Users/[redacted]/git_demo

[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo (master)
$ mkdir Changes

[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo (master)
$ cd Changes

[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/Changes (master)
$ pwd
/c/Users/[redacted]/git_demo/Changes
```

- The Changes folder is empty. So let's initialize a repository to that.





- Now let's pull files from the remote repository. In GitHub, go to the repository which we want to pull and then click on **Code** option and copy the **URL** to **clone.**

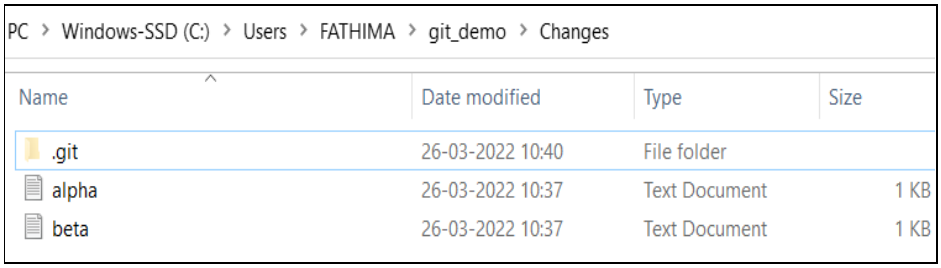


- In GitBash, type the pull command followed by the copied URL.

```
$ git pull copied URL
```

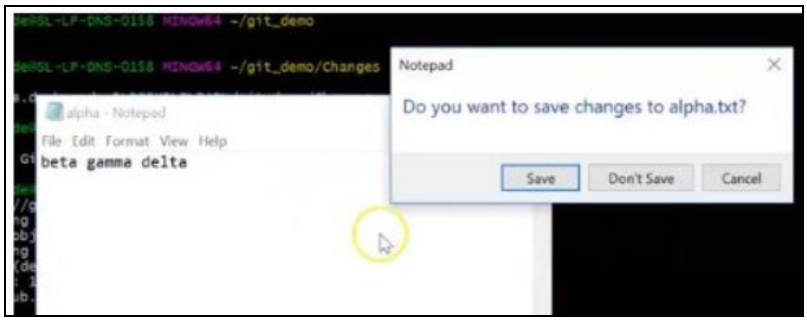
```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/Changes (master)
$ git pull https://github.com/[redacted]/FirstRepo.git
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 10 (delta 0), reused 10 (delta 0), pack-reused 0
Unpacking objects: 100% (10/10), 727 bytes | 22.00 KiB/s, done.
From https://github.com/[redacted]/FirstRepo
 * branch            HEAD       -> FETCH_HEAD
```

- The contents from the remote repository have been pulled to the local repository.



- Open the notepad from GitBash and make changes.

```
C:/windows/notepad alpha

C:/windows/notepad beta
```



- Let's check the status

```
$ git status
```

- Status shows the untracked files and those that are not committed. Untracked files are in red color.

```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/Changes (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:   alpha.txt
        modified:   beta.txt

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

- To track the files ***$ git add .*** command is used and later commit the files.

```
[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/Changes (master)
$ git add .

[redacted]@LAPTOP-S9VIU5AR MINGW64 ~/git_demo/Changes (master)
$ git commit -m "changed"
[master c1a53b1] changed
 2 files changed, 2 insertions(+), 2 deletions(-)
```