Q1. Describe the usage of the git stash command by using an example and also state the process by giving the screenshot of all the commands written in git bash.

Initially for doing any operations on git bash, configuring the system is the first step.

We can globally configure the system by user name and mail.

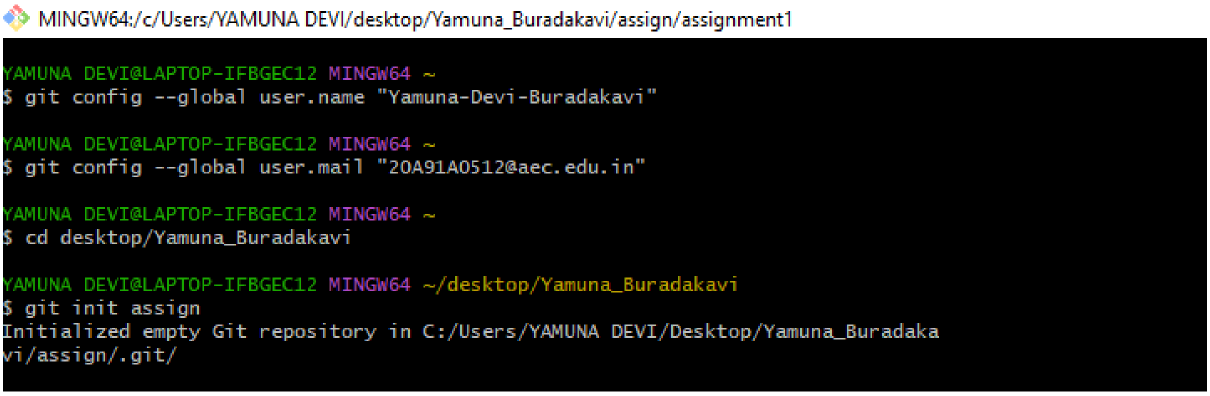
The commands are as follows:

git config –global user.name “user-name”

git config –global user.mail “email-address”

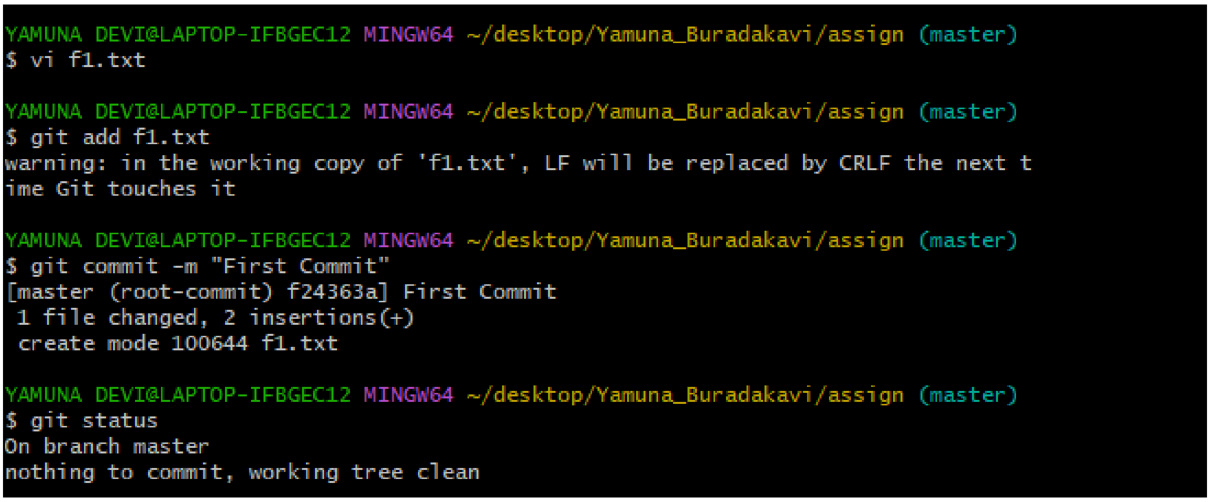
Next, we initialized an empty git repository. By default, it contains **.git** file.

The git stash command takes your uncommitted changes (both staged and unstaged), saves them away for later purposes. It uses stack data structure. Stash can be used when we are not sure about the modifications. Generally , it does not show the changes but stores them.



```
MINGW64:/c/Users/YAMUNA DEVI/desktop/Yamuna_Buradakavi/assign/assignment1

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~
$ git config --global user.name "Yamuna-Devi-Buradakavi"

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~
$ git config --global user.mail "20A91A0512@aec.edu.in"

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~
$ cd desktop/Yamuna_Buradakavi

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi
$ git init assign
Initialized empty Git repository in C:/Users/YAMUNA DEVI/Desktop/Yamuna_Buradaka
vi/assign/.git/
```

Step-1: First, we need to create files, add and commit the files.

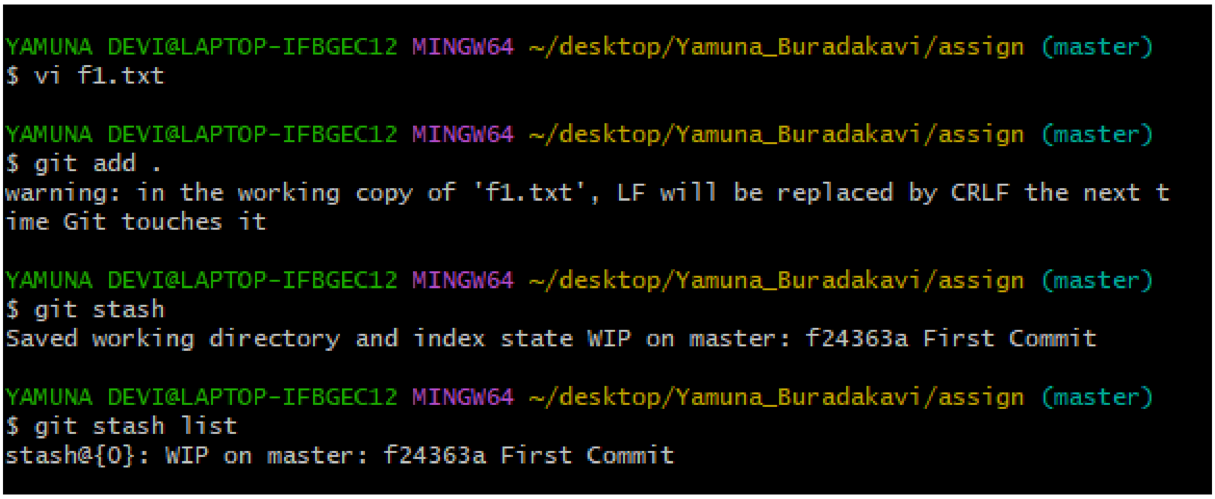


```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ vi f1.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git add f1.txt
warning: in the working copy of 'f1.txt', LF will be replaced by CRLF the next t
ime Git touches it

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git commit -m "First Commit"
[master (root-commit) f24363a] First Commit
 1 file changed, 2 insertions(+)
 create mode 100644 f1.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git status
On branch master
nothing to commit, working tree clean
```

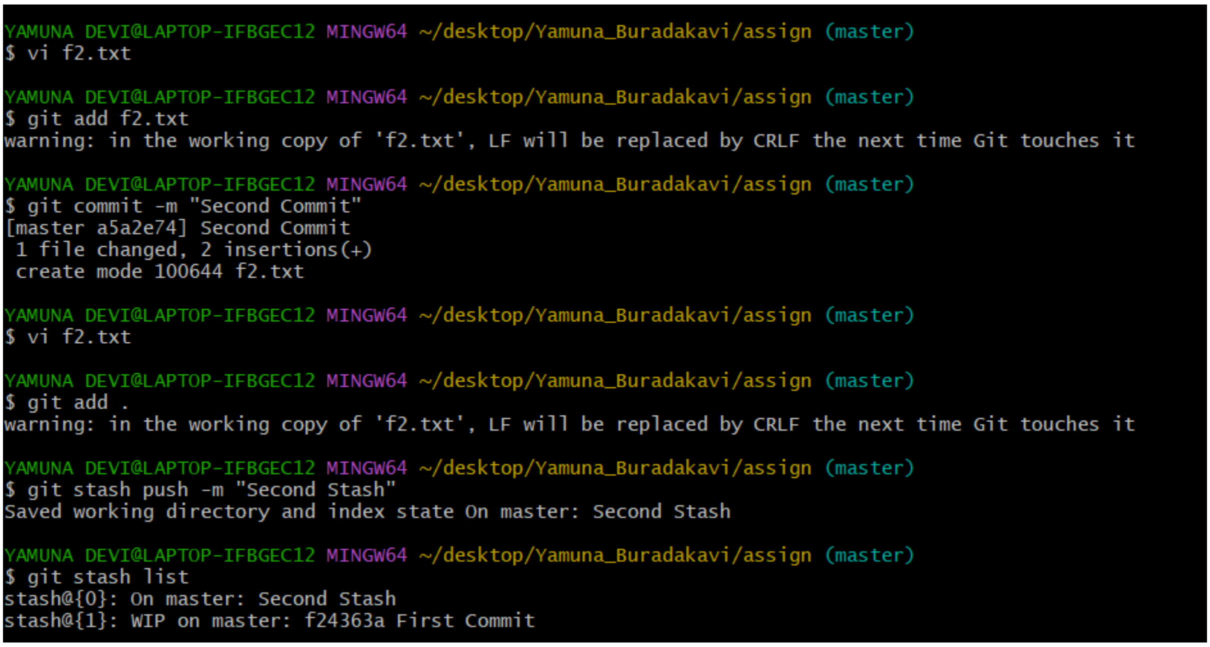
Step-2: Now, we can modify the changes in the file and add them. Now by using the command

Git stash (or) git stash push -m “message”

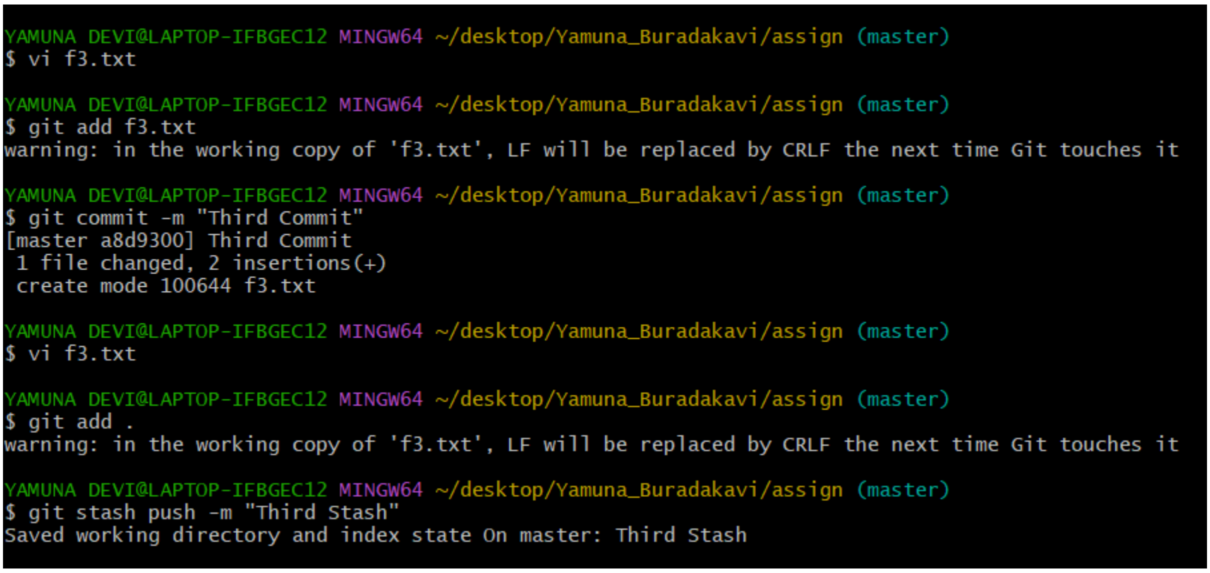
We can store the changes in a stack data structured stash. In the same way, two more stashes have been done.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ vi f1.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git add .
warning: in the working copy of 'f1.txt', LF will be replaced by CRLF the next t
ime Git touches it

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash
Saved working directory and index state WIP on master: f24363a First Commit

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash list
stash@{0}: WIP on master: f24363a First Commit
```



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ vi f2.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git add f2.txt
warning: in the working copy of 'f2.txt', LF will be replaced by CRLF the next time Git touches it

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git commit -m "Second Commit"
[master a5a2e74] Second Commit
 1 file changed, 2 insertions(+)
 create mode 100644 f2.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ vi f2.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git add .
warning: in the working copy of 'f2.txt', LF will be replaced by CRLF the next time Git touches it

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash push -m "Second Stash"
Saved working directory and index state On master: Second Stash

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash list
stash@{0}: On master: Second Stash
stash@{1}: WIP on master: f24363a First Commit
```



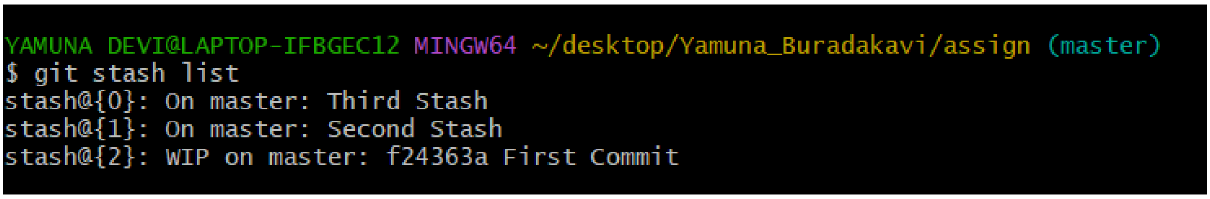
```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ vi f3.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git add f3.txt
warning: in the working copy of 'f3.txt', LF will be replaced by CRLF the next time Git touches it

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git commit -m "Third Commit"
[master a8d9300] Third Commit
 1 file changed, 2 insertions(+)
 create mode 100644 f3.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ vi f3.txt

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git add .
warning: in the working copy of 'f3.txt', LF will be replaced by CRLF the next time Git touches it

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash push -m "Third Stash"
Saved working directory and index state On master: Third Stash
```

Stash list:

The list of stashes will be displayed with index. The most recent stash will occupy the top index i.e., 0.

The stashes saved with messages will also be displayed.

Git stash list

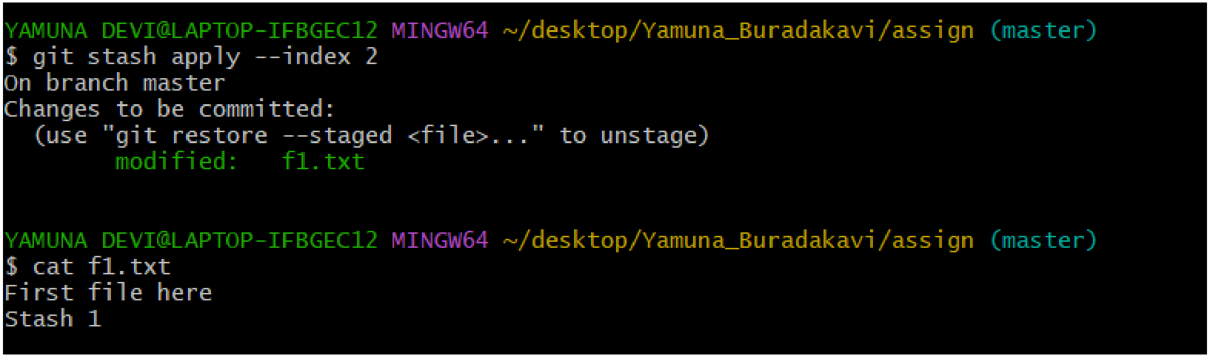


```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash list
stash@{0}: On master: Third Stash
stash@{1}: On master: Second Stash
stash@{2}: WIP on master: f24363a First Commit
```

Stash apply:

It re-applies the changes of the most recent by using the git stash command. If multiple stash are present, git stash apply command followed by stash index to apply the changes of that particular commit.

Git stash apply –index {}

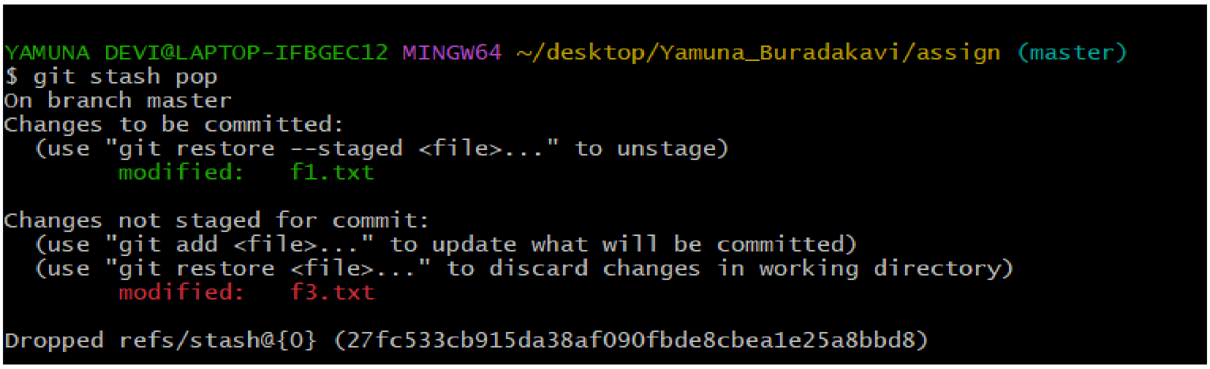


```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash apply --index 2
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   f1.txt


YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ cat f1.txt
First file here
Stash 1
```

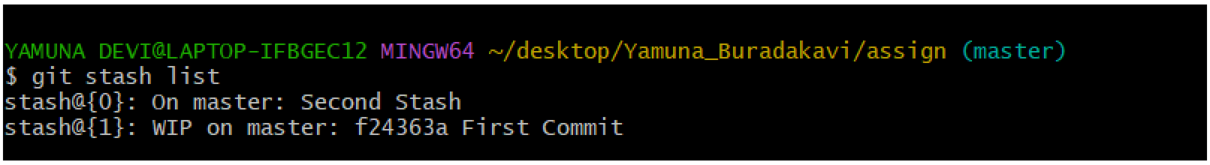
Git Stash pop:

The git stash pop command deletes the stash from the stack after it is applied. It is similar to apply but it returns the top most stash.

Git stash pop



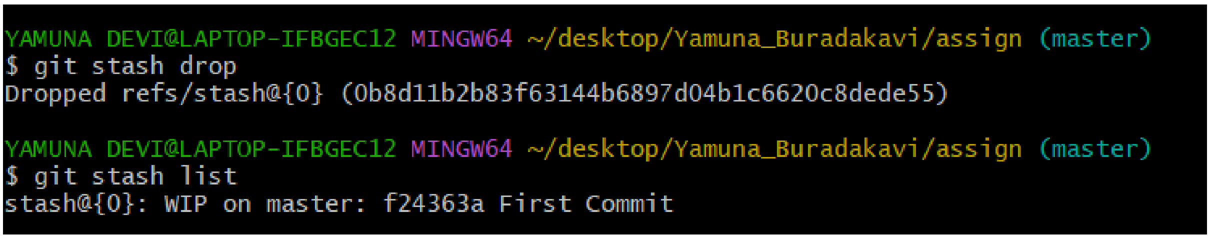
```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash pop
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   f1.txt

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:   f3.txt

Dropped refs/stash@{0} (27fc533cb915da38af090fbde8cbea1e25a8bbd8)
```



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash list
stash@{0}: On master: Second Stash
stash@{1}: WIP on master: f24363a First Commit
```

Stash Drop:

The **git stash drop** command is used to delete a stash from the stack. It deletes the top most stash.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash drop
Dropped refs/stash@{0} (0b8d11b2b83f63144b6897d04b1c6620c8dede55)

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash list
stash@{0}: WIP on master: f24363a First Commit
```

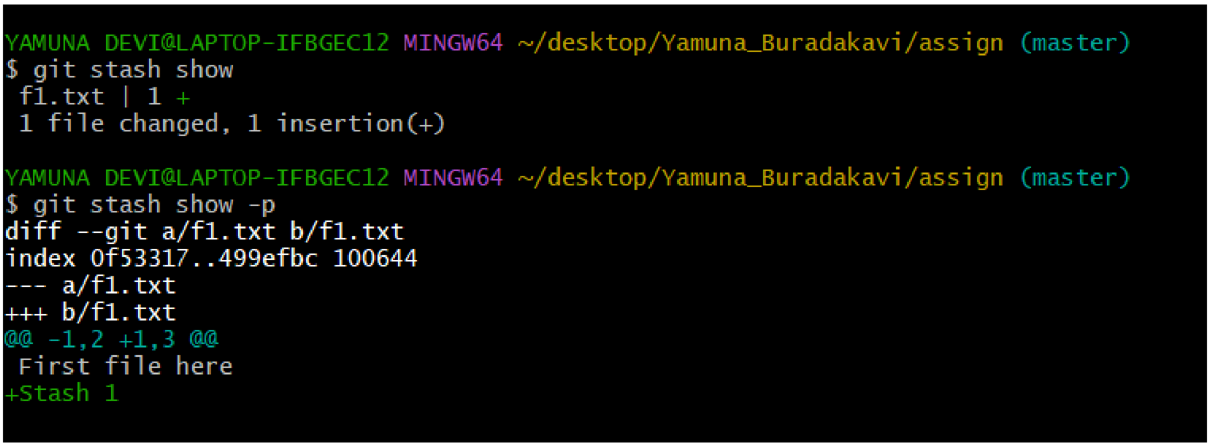
Stash Show:

The command shows the file that is stashed and changes made on those files. Here, it shows the file insertions and the changes made in them.

Git stash show

The changes made in the file are displayed with the command

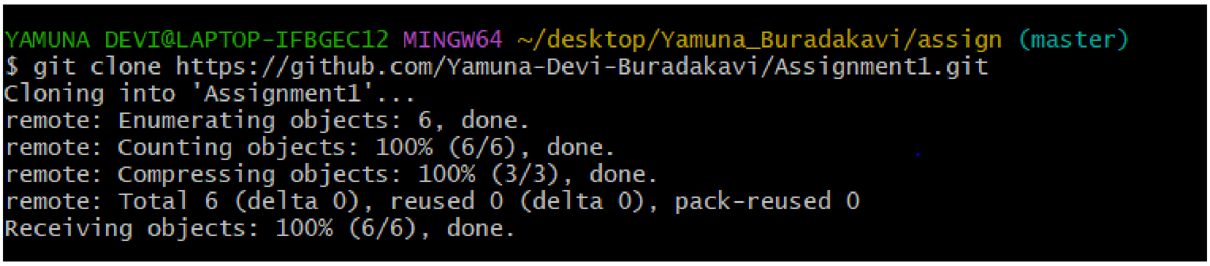
Git stash show -p



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash show
 f1.txt | 1 +
 1 file changed, 1 insertion(+)

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git stash show -p
diff --git a/f1.txt b/f1.txt
index 0f53317..499efbc 100644
--- a/f1.txt
+++ b/f1.txt
@@ -1,2 +1,3 @@
 First file here
+Stash 1
```

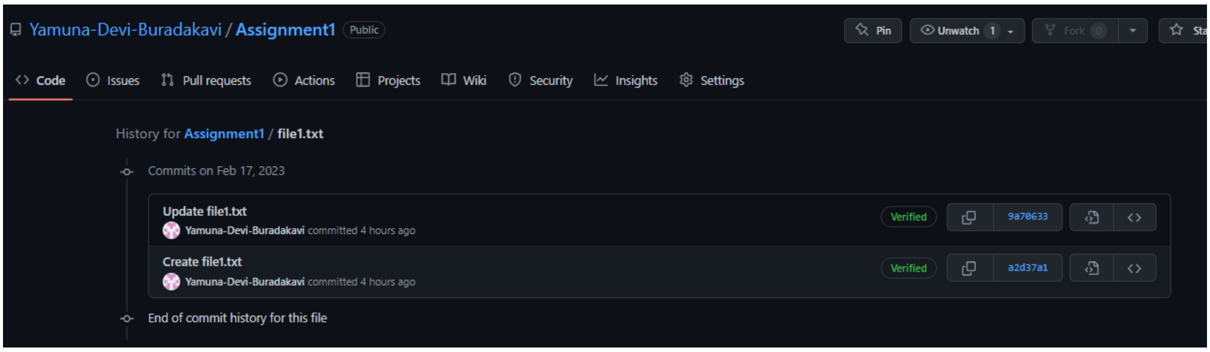
Q2. By using a sample example of your choice, use the git fetch command and also use the git merge command and describe the whole process through a screenshot with all the commands and their output in git bash.

Initially, we clone the repository from github to git using **git clone "url" .**The url can be copied from code in github repository.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ git clone https://github.com/Yamuna-Devi-Buradakavi/Assignment1.git
Cloning into 'Assignment1'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
```

Then, change the directory to the cloned repository. Make the changes in github repo file1.txt



History for Assignment1 / file1.txt

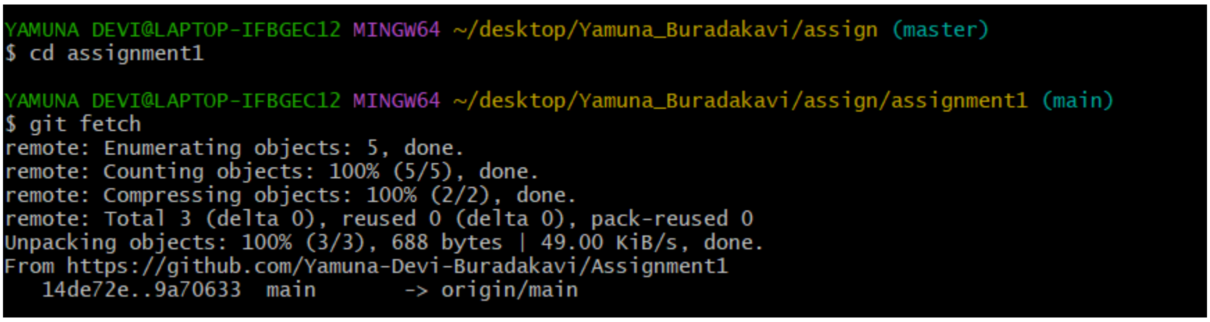
Commits on Feb 17, 2023

Update file1.txt
Yamuna-Devi-Buradakavi committed 4 hours ago
Verified 9a70633

Create file1.txt
Yamuna-Devi-Buradakavi committed 4 hours ago
Verified a2d37a1

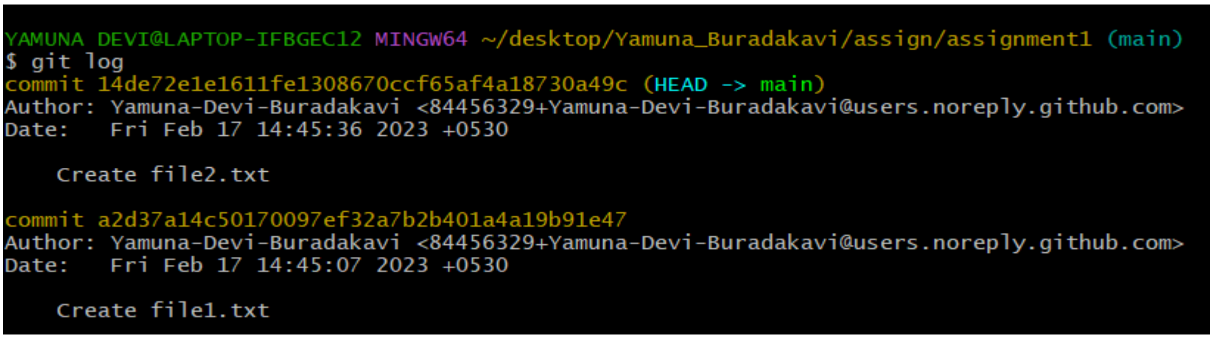
End of commit history for this file

Now, Fetch it. Git fetch downloads all the commits, objects from the cloned repository but it does not affect the current working one.



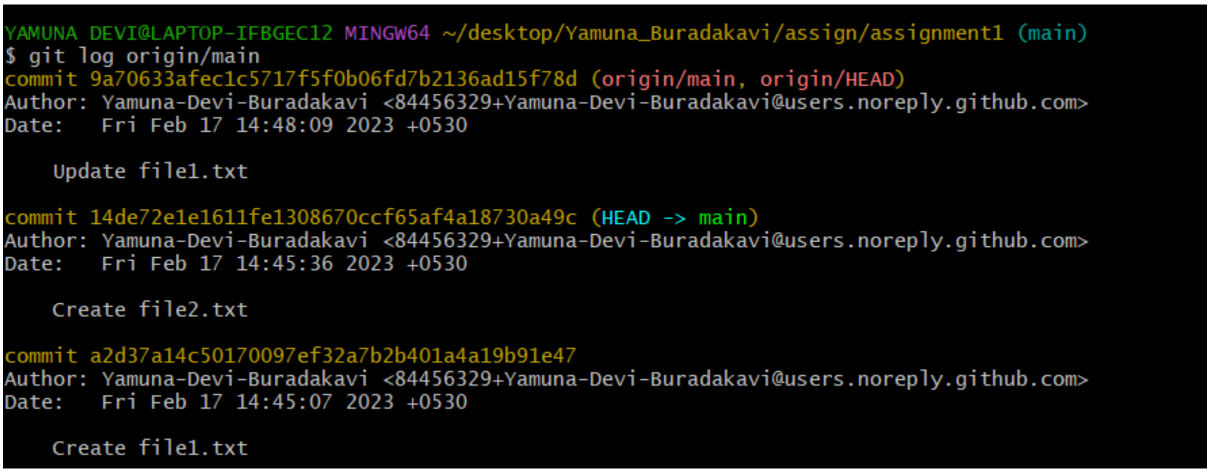
```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ cd assignment1

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 688 bytes | 49.00 KiB/s, done.
From https://github.com/Yamuna-Devi-Buradakavi/Assignment1
   14de72e..9a70633  main       -> origin/main
```

Here, we can observe that they are only two commits just for creating files.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git log
commit 14de72e1e1611fe1308670ccf65af4a18730a49c (HEAD -> main)
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:36 2023 +0530

    Create file2.txt

commit a2d37a14c50170097ef32a7b2b401a4a19b91e47
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:07 2023 +0530

    Create file1.txt
```

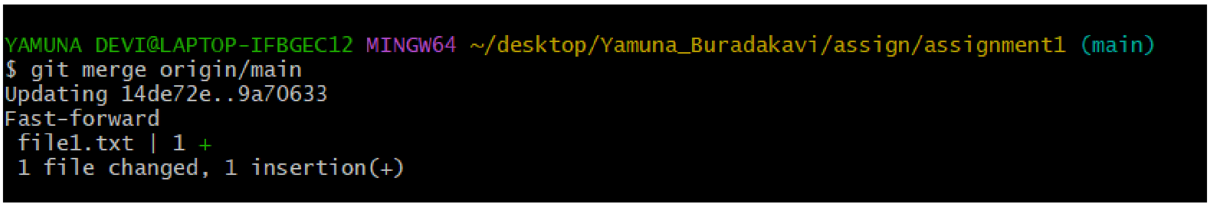
By using **git log origin/main**, we can observe the commits are three.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git log origin/main
commit 9a70633afec1c5717f5f0b06fd7b2136ad15f78d (origin/main, origin/HEAD)
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:48:09 2023 +0530

    Update file1.txt

commit 14de72e1e1611fe1308670ccf65af4a18730a49c (HEAD -> main)
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:36 2023 +0530

    Create file2.txt

commit a2d37a14c50170097ef32a7b2b401a4a19b91e47
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:07 2023 +0530

    Create file1.txt
```

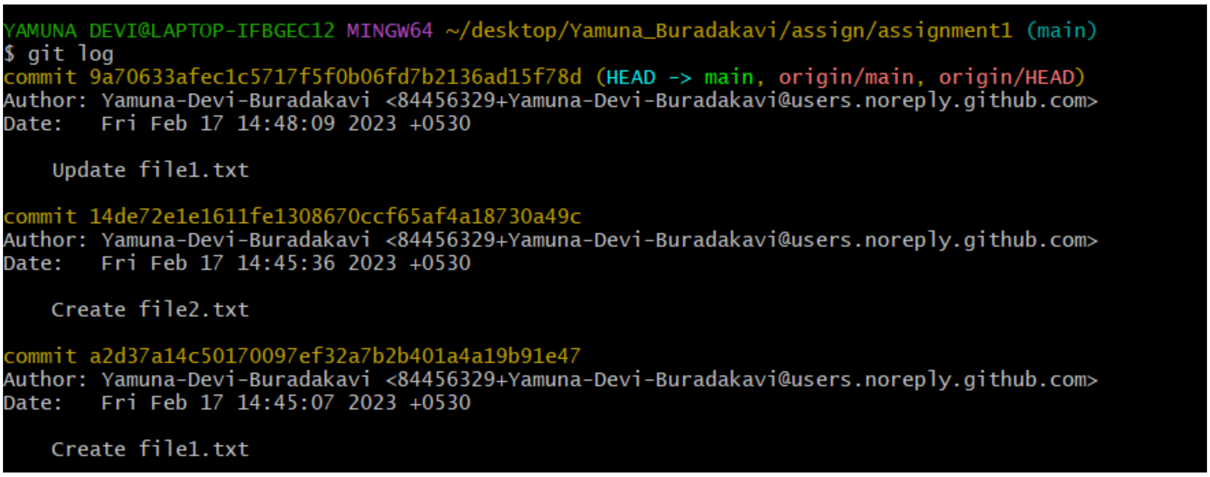
Git Merge:

This command will merge the specifically mentioned commit to a specifically mentioned branch by passing in the branch name in commit

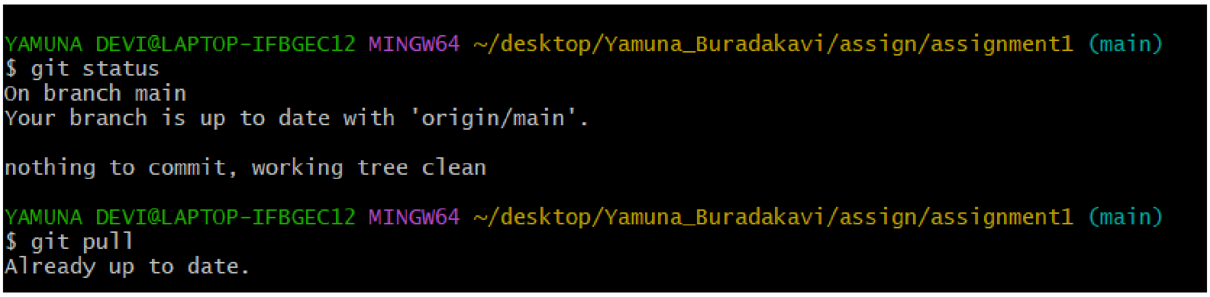


```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git merge origin/main
Updating 14de72e..9a70633
Fast-forward
 file1.txt | 1 +
 1 file changed, 1 insertion(+)
```

Now, we can observe all the commits here including the modifications.



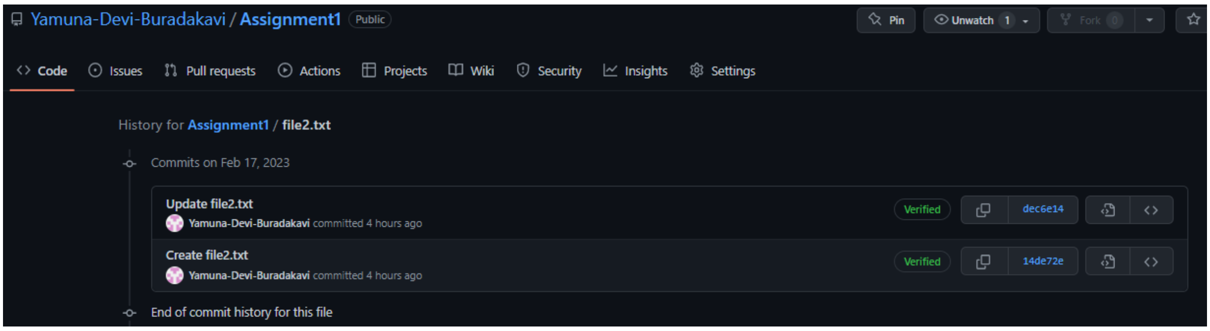
```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git log
commit 9a70633afec1c5717f5f0b06fd7b2136ad15f78d (HEAD -> main, origin/main, origin/HEAD)
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:48:09 2023 +0530

    Update file1.txt

commit 14de72e1e1611fe1308670ccf65af4a18730a49c
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:36 2023 +0530

    Create file2.txt

commit a2d37a14c50170097ef32a7b2b401a4a19b91e47
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:07 2023 +0530

    Create file1.txt
```



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git pull
Already up to date.
```

Q3. State the difference between git fetch and git pull by doing a practical example in your git bash and attach a screenshot of all the processes.

Git fetch downloads all the commits, objects from the cloned repository but it does not effect the current working one. Here we changed another file with modifications but fetch does not represent it.



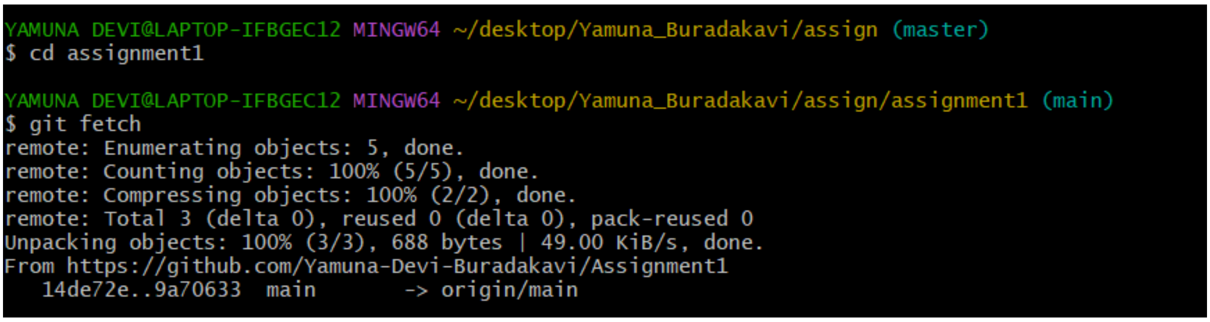
History for Assignment1 / file2.txt

Commits on Feb 17, 2023

Update file2.txt
Yamuna-Devi-Buradakavi committed 4 hours ago
Verified dec6e14

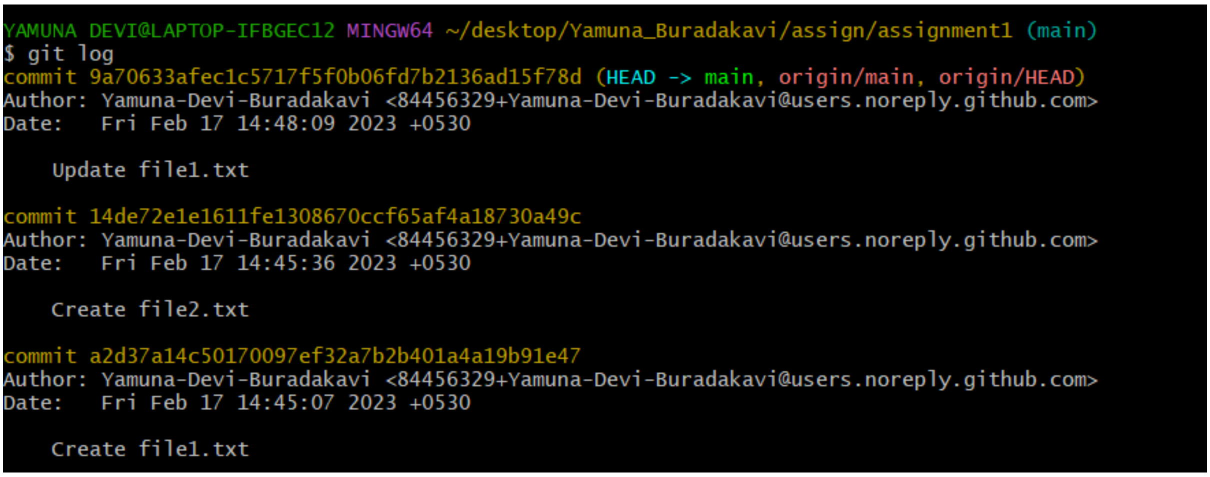
Create file2.txt
Yamuna-Devi-Buradakavi committed 4 hours ago
Verified 14de72e

End of commit history for this file



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ cd assignment1

YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 688 bytes | 49.00 KiB/s, done.
From https://github.com/Yamuna-Devi-Buradakavi/Assignment1
   14de72e..9a70633  main       -> origin/main
```

The log here does not represent the fourth commit.

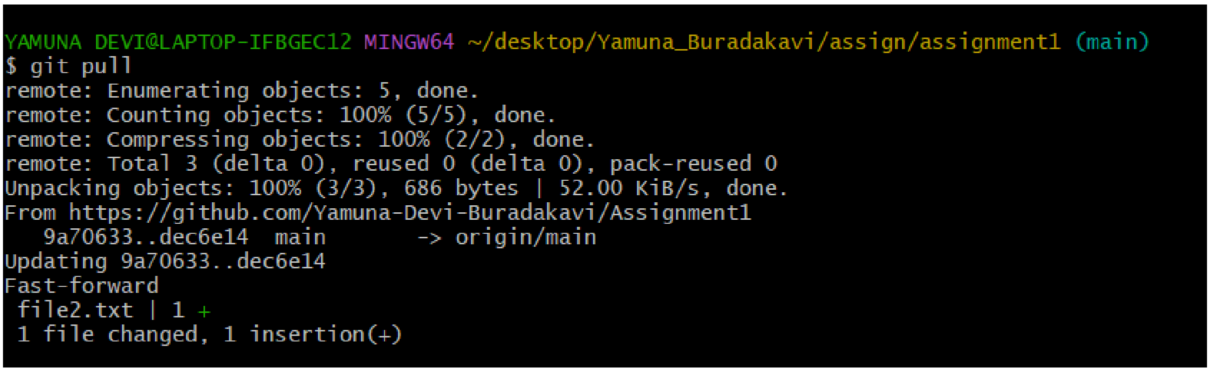


```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git log
commit 9a70633afec1c5717f5f0b06fd7b2136ad15f78d (HEAD -> main, origin/main, origin/HEAD)
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:48:09 2023 +0530

    Update file1.txt

commit 14de72e1e1611fe1308670ccf65af4a18730a49c
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:36 2023 +0530

    Create file2.txt

commit a2d37a14c50170097ef32a7b2b401a4a19b91e47
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:07 2023 +0530

    Create file1.txt
```

Git Pull:

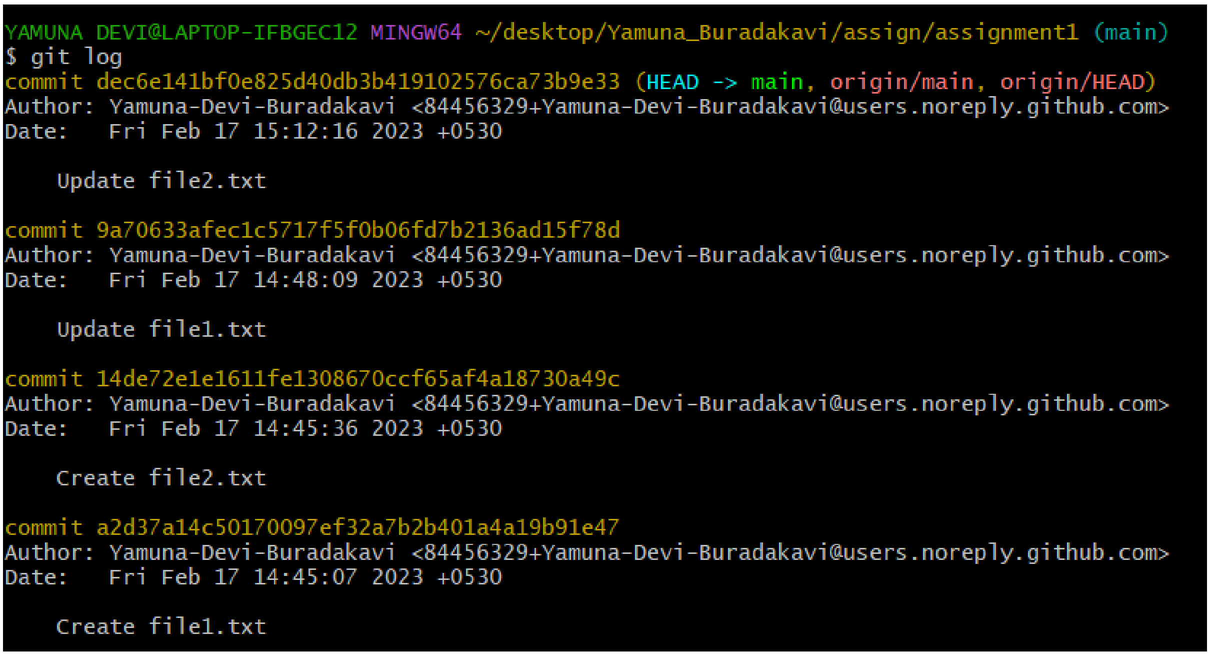
The git pull command is used to fetch and merge content from a remote repository and immediately update the local repository to match the changes and commits to the remote repo.

Git pull



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 686 bytes | 52.00 KiB/s, done.
From https://github.com/Yamuna-Devi-Buradakavi/Assignment1
   9a70633..dec6e14  main       -> origin/main
Updating 9a70633..dec6e14
Fast-forward
 file2.txt | 1 +
 1 file changed, 1 insertion(+)
```

After pull, we can directly see the changes in git log . Hence, pull is almost equivalent to fetch and merge. That’s the major difference between pull and fetch.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ git log
commit dec6e141bf0e825d40db3b419102576ca73b9e33 (HEAD -> main, origin/main, origin/HEAD)
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 15:12:16 2023 +0530

    Update file2.txt

commit 9a70633afec1c5717f5f0b06fd7b2136ad15f78d
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:48:09 2023 +0530

    Update file1.txt

commit 14de72e1e1611fe1308670ccf65af4a18730a49c
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:36 2023 +0530

    Create file2.txt

commit a2d37a14c50170097ef32a7b2b401a4a19b91e47
Author: Yamuna-Devi-Buradakavi <84456329+Yamuna-Devi-Buradakavi@users.noreply.github.com>
Date:   Fri Feb 17 14:45:07 2023 +0530

    Create file1.txt
```

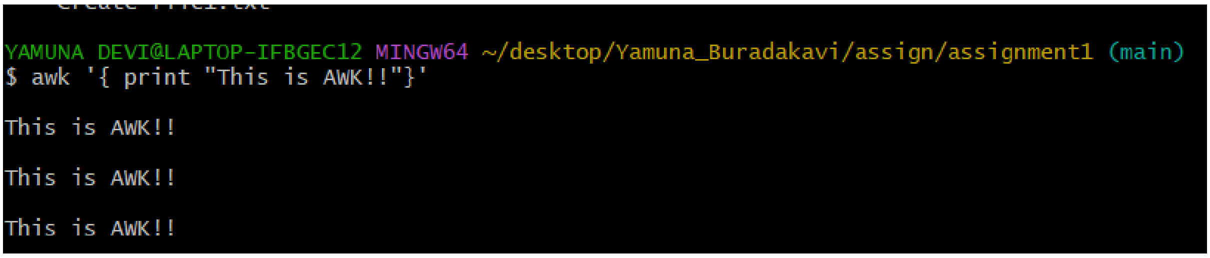
Q4. Try to find out about the awk command and use it while reading a file created by yourself. Also, make a bash script file and try to find out the prime number from the range 1 to 20.

The whole process should be carried out and by using the history command, give the screenshot of all the processes being carried out.

AWK

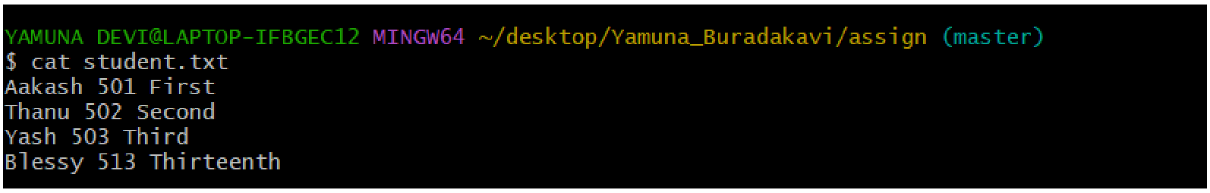
Awk enables a programmer to write small and helpful programming code in statements by which we can find similar patterns and when found ,take necessary actions.

Awk '{ print "message"}'



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign/assignment1 (main)
$ awk '{ print "This is AWK!!"}'

This is AWK!!

This is AWK!!

This is AWK!!
```

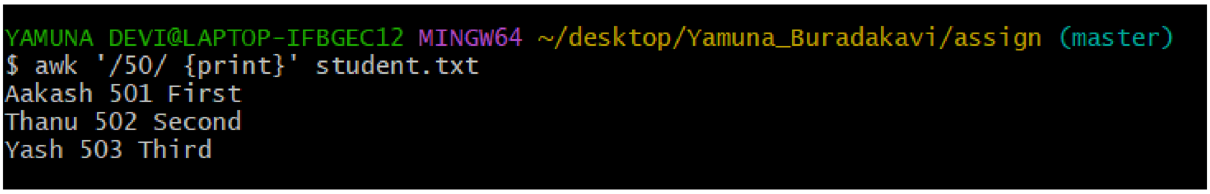
For example, create a file student.txt and display it



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ cat student.txt
Aakash 501 First
Thanu 502 Second
Yash 503 Third
Blessy 513 Thirteenth
```

In here, we can find the similar patterns in the data of the file.

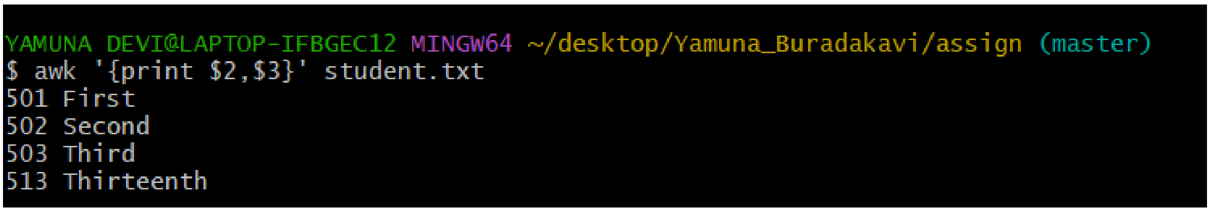
Now, let's print the statements accordingly with the patterns.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ awk '/50/ {print}' student.txt
Aakash 501 First
Thanu 502 Second
Yash 503 Third
```

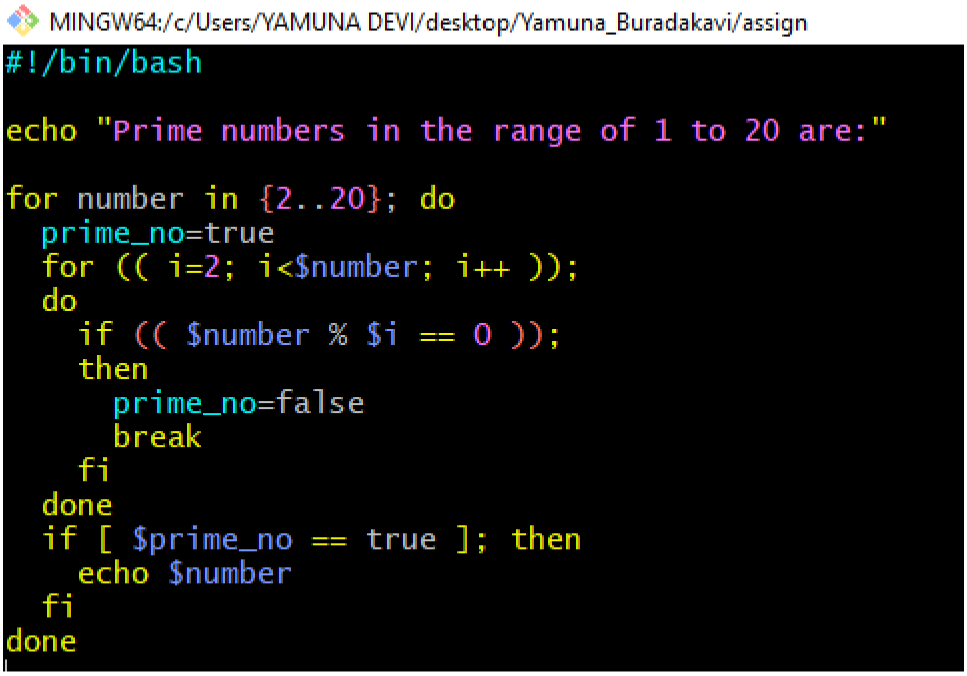
Here, Only three of the students have the pattern 50.

Next, we can print them like columns.



```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ awk '{print $2,$3}' student.txt
501 First
502 Second
503 Third
513 Thirteenth
```

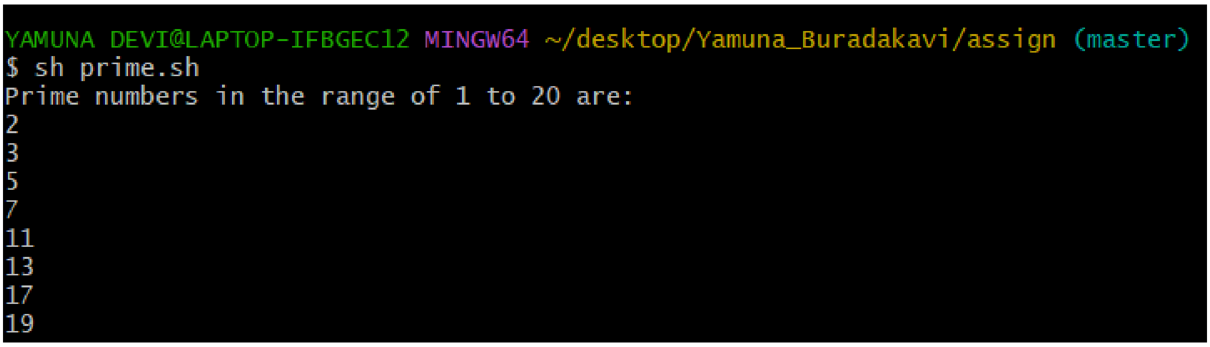
Bash script file for finding prime numbers from 1 to 20:



MINGW64:/c/Users/YAMUNA DEVI/desktop/Yamuna_Buradakavi/assign

```
#!/bin/bash

echo "Prime numbers in the range of 1 to 20 are:"

for number in {2..20}; do
  prime_no=true
  for (( i=2; i<$number; i++ ));
  do
    if (( $number % $i == 0 ));
    then
      prime_no=false
      break
    fi
  done
  if [ $prime_no == true ]; then
    echo $number
  fi
done
```

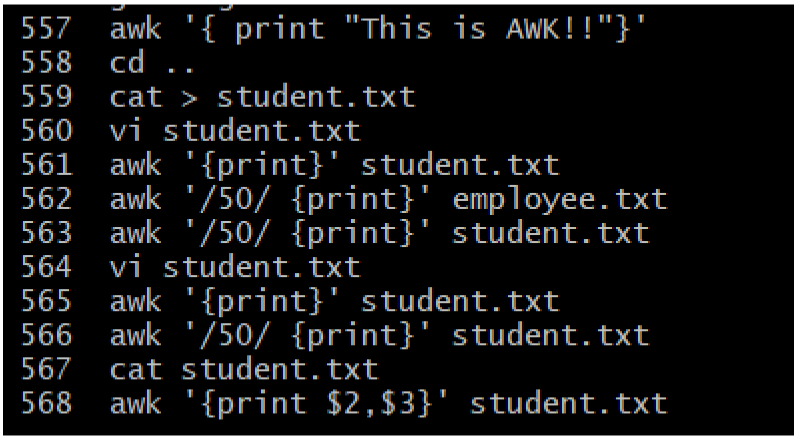
Output:



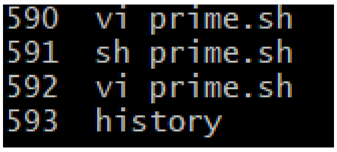
```
YAMUNA DEVI@LAPTOP-IFBGEC12 MINGW64 ~/desktop/Yamuna_Buradakavi/assign (master)
$ sh prime.sh
Prime numbers in the range of 1 to 20 are:
2
3
5
7
11
13
17
19
```

History:

This command gives us the list of commands we used.



```
 557  awk '{ print "This is AWK!!"}'
 558  cd ..
 559  cat > student.txt
 560  vi student.txt
 561  awk '{print}' student.txt
 562  awk '/50/ {print}' employee.txt
 563  awk '/50/ {print}' student.txt
 564  vi student.txt
 565  awk '{print}' student.txt
 566  awk '/50/ {print}' student.txt
 567  cat student.txt
 568  awk '{print $2,$3}' student.txt
```



```
590  vi prime.sh
591  sh prime.sh
592  vi prime.sh
593  history
```

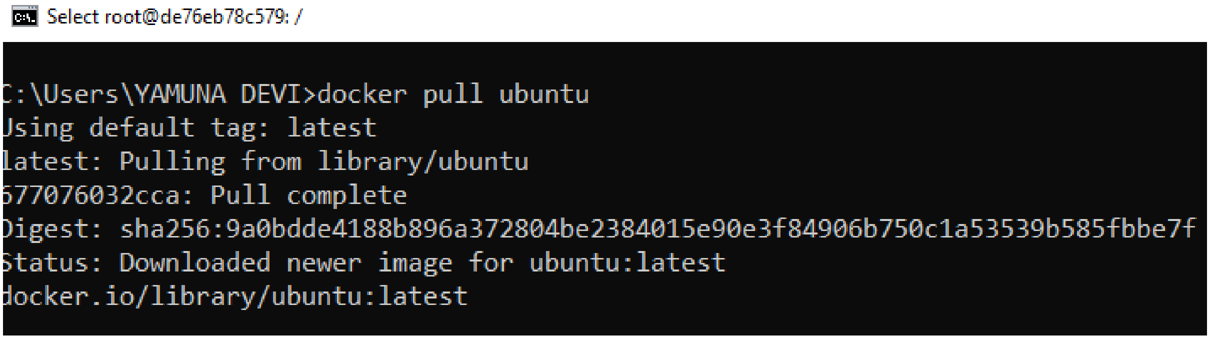
Q5. Set up a container and run an Ubuntu operating system. For this purpose, you can make use of the docker hub and run the container in interactive mode.

The definition of Docker is an open-source centralized platform designed to create, deploy, and run applications. It is based on the process of containerization. It involves the components like docker host, registry, servers etc.

It also includes containers and images.

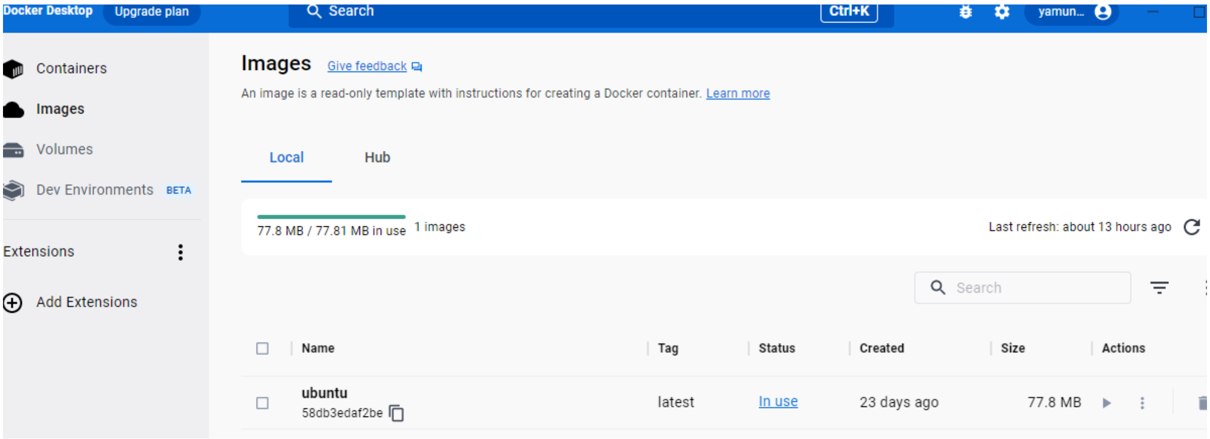
We can download the image ubuntu by this command

Docker pull ubuntu



Select root@de76eb78c579: /

```
C:\Users\YAMUNA DEVI>docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
677076032cca: Pull complete
Digest: sha256:9a0bdde4188b896a372804be2384015e90e3f84906b750c1a53539b585fbbe7f
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```



Images

An image is a read-only template with instructions for creating a Docker container. Learn more

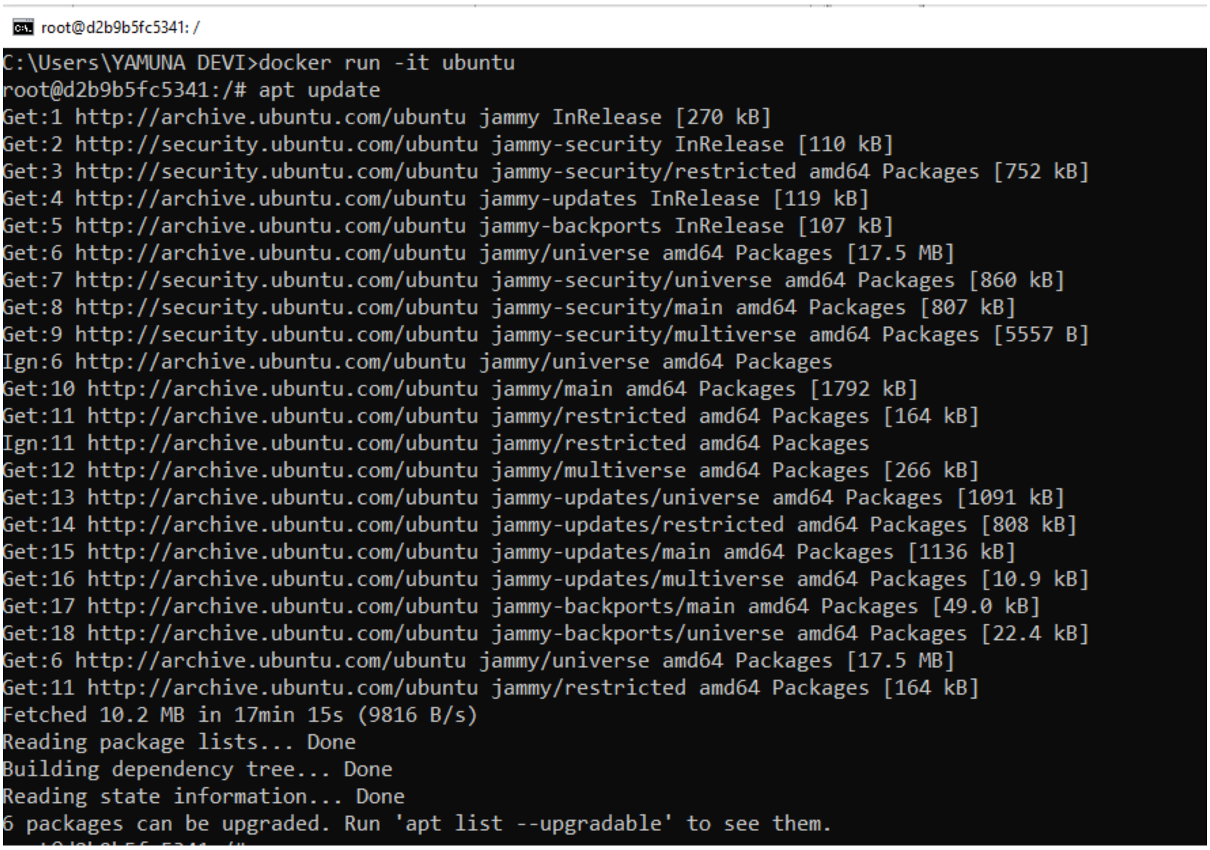
77.8 MB / 77.81 MB in use 1 images

Last refresh: about 13 hours ago

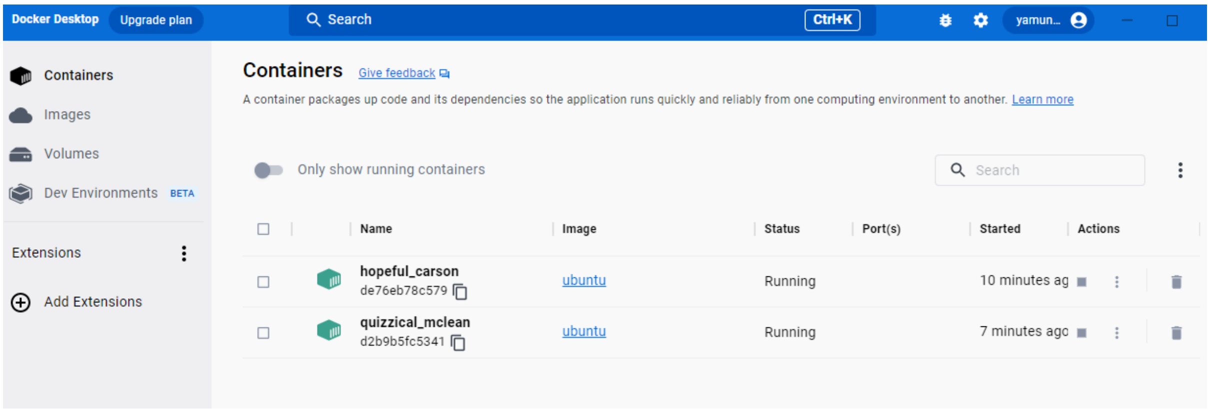
□	Name	Tag	Status	Created	Size	Actions
□	ubuntu 58db3edaf2be	latest	In use	23 days ago	77.8 MB	▶ ⋮

We can run the image by the command

Docker run -it ubuntu



```
C:\Users\YAMUNA DEVI>docker run -it ubuntu
root@d2b9b5fc5341:/# apt update
Get:1 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [752 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [107 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [17.5 MB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [860 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [807 kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [5557 B]
Ign:6 http://archive.ubuntu.com/ubuntu jammy/universe amd64 Packages
Get:10 http://archive.ubuntu.com/ubuntu jammy/main amd64 Packages [1792 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages [164 kB]
Ign:11 http://archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages
Get:12 http://archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [266 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1091 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [808 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1136 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [10.9 kB]
Get:17 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [49.0 kB]
Get:18 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [22.4 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [17.5 MB]
Get:11 http://archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages [164 kB]
Fetched 10.2 MB in 17min 15s (9816 B/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
6 packages can be upgraded. Run 'apt list --upgradable' to see them.
```



Containers

Give feedback

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

Only show running containers

Search

□	Name	Image	Status	Port(s)	Started	Actions
□	hopeful_carson de76eb78c579	ubuntu	Running		10 minutes ag	
□	quizzical_mclean d2b9b5fc5341	ubuntu	Running		7 minutes ago	