

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.

Git stash: The git stash command takes your uncommitted changes (both staged and unstaged), saves them away for later use, and then reverts them from your working copy.

```
MINGW64:/c/Users/rajav/desktop/git1

raja vemana@Raja-PC MINGW64 ~
$ cd desktop

raja vemana@Raja-PC MINGW64 ~/desktop
$ mkdir git1

raja vemana@Raja-PC MINGW64 ~/desktop
$ cd git1
```

```
MINGW64:/c/Users/rajav/desktop/git1

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git branch
* master

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git config --global user.name "Raja-vemana"

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git config --global user.email "rajavemana2002@gmail.com"

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git branch brn1

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git branch
brn1
* master
```

Create a file. For example here we created assign1.

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ vi doc1
```



```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ vi doc1
```

```
stash Assignment1
line 2 added
```

```
doc1 [unix] (11:42 16/02/2023) 3,0-1 All
```

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (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:   doc1

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

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash
warning: in the working copy of 'doc1', LF will be replaced by CRLF the next time Git touches it
Saved working directory and index state WIP on master: 05a0e4f 1line added
```

***git stash list**: Used to list out all the stashes stored in the stash stack.

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash list
stash@{0}: WIP on master: 05a0e4f 1line added
```

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ vi doc1
```



```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (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:   doc1

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

```

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ vi doc1
```

***git stash show**: By default git stash show shows the changes recorded in the **latest** stash (stash@{0}) in the --stat format.

```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash show
doc1 | 1 +
1 file changed, 1 insertion(+)

```

Syntax: `git stash show -index`

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash show 1
doc1 | 1 +
1 file changed, 1 insertion(+)
```

***git stash apply:** Used to apply the latest stash without removing the stash from the stash stack.

```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash apply
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:   doc1

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

```

```
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ vi doc1
```

[illegible]

*The changes from the latest stash i.e, the line “Some more lines” got added to the file.

*The stash was also not removed from the stack.

```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash list
stash@{0}: WIP on master: 05a0e4f 1line added
stash@{1}: WIP on master: 05a0e4f 1line added

```

*We can also use it along with index number by giving →git apply index

***git stash pop**: Used to apply the recorded changes of your latest stash on the current working branch as well as remove that stash from the stash stack.

```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash pop
error: Your local changes to the following files would be overwritten by merge:
    doc1
Please commit your changes or stash them before you merge.
Aborting
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:   doc1

no changes added to commit (use "git add" and/or "git commit -a")
The stash entry is kept in case you need it again.

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ vi doc1

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git commit -a -m "stash0 applied"
[master 3cf6e8a] stash0 applied
1 file changed, 1 insertion(+)

```

```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash pop
On branch master
nothing to commit, working tree clean
Dropped refs/stash@{0} (4f5d557f65169df83023257bee12a6238e0b74d9)

```

[illegible]

- ***git stash clear** : Used to clear all the stashes from stash stack.

```

raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash clear

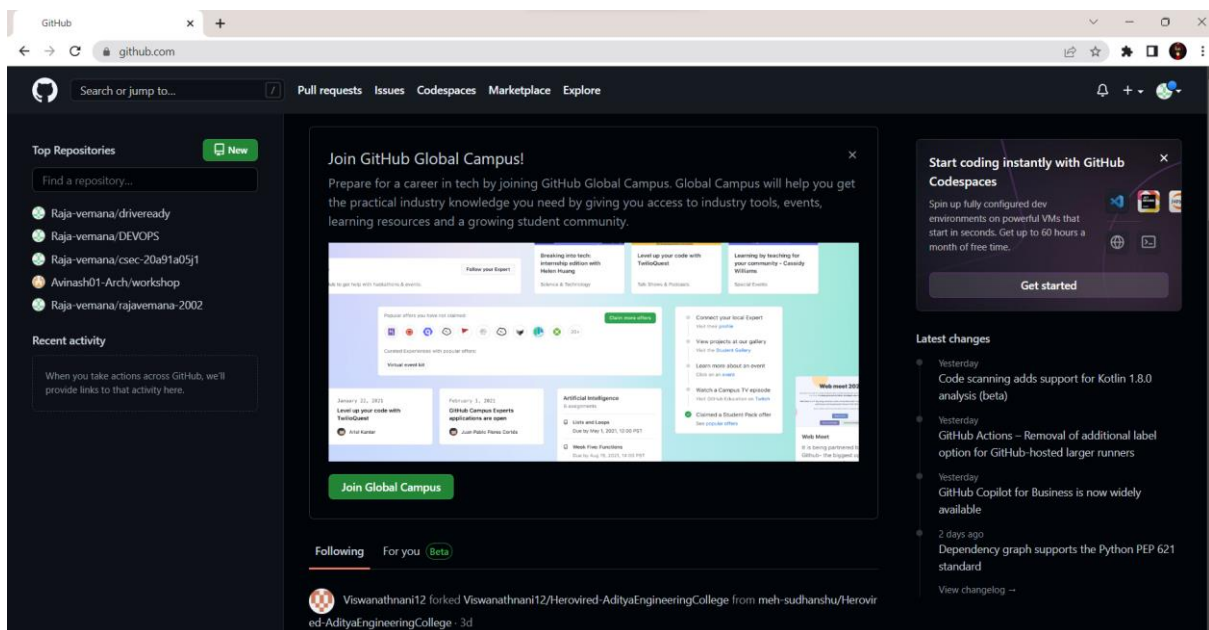
raja vemana@Raja-PC MINGW64 ~/desktop/git1 (master)
$ git stash list

```

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.

Git fetch: Git fetch is a command that allows you to download objects from remote repository but it doesn't integrate any of this new data into your working files.

*Create a new repository in github



Create a new repository

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

Owner *

Raja-vemana ▼

Repository name *

Fetch and pull ✓

Great repository names are short and descriptive. Your new repository will be created as `Fetch-and-pull`. [Learn more.](#)

Description (optional)

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▼

The screenshot shows the GitHub repository page for 'Raja-vemana / Fetch-and-pull'. The repository is public and has 1 unwatcher, 0 forks, and 0 stars. The page includes a search bar, navigation links (Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, Settings), and a 'Quick setup' section. The 'Quick setup' section provides instructions for setting up the repository on a desktop or using HTTPS/SSH, and also includes a command line setup for creating a new repository or pushing an existing one.

```
Quick setup — if you've done this kind of thing before
Set up in Desktop or HTTPS SSH https://github.com/Raja-vemana/Fetch-and-pull.git
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line
echo "# Fetch-and-pull" >> README.md
git init
git add README.md
git commit -m "First commit"
git branch -M main
git remote add origin https://github.com/Raja-vemana/Fetch-and-pull.git
git push -u origin main

...or push an existing repository from the command line
git remote add origin https://github.com/Raja-vemana/Fetch-and-pull.git
git branch -M main
git push -u origin main
```

*Clone the empty remote repository into local repository using git bash.

```

raja vemana@Raja-PC MINGW64 ~
$ cd desktop

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ git init
Reinitialized existing Git repository in C:/Users/rajav/Desktop/.

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ git config --global user.name "Raja-vemana"

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ git config --global user.email "rajavemana2002@gmail.com"

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ git clone https://github.com/Raja-vemana/Fetch-and-pull.git
Cloning into 'Fetch-and-pull'...
warning: You appear to have cloned an empty repository.

```

*Check if there are any commits present in the repository.

```

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ cd Fetch-and-pull

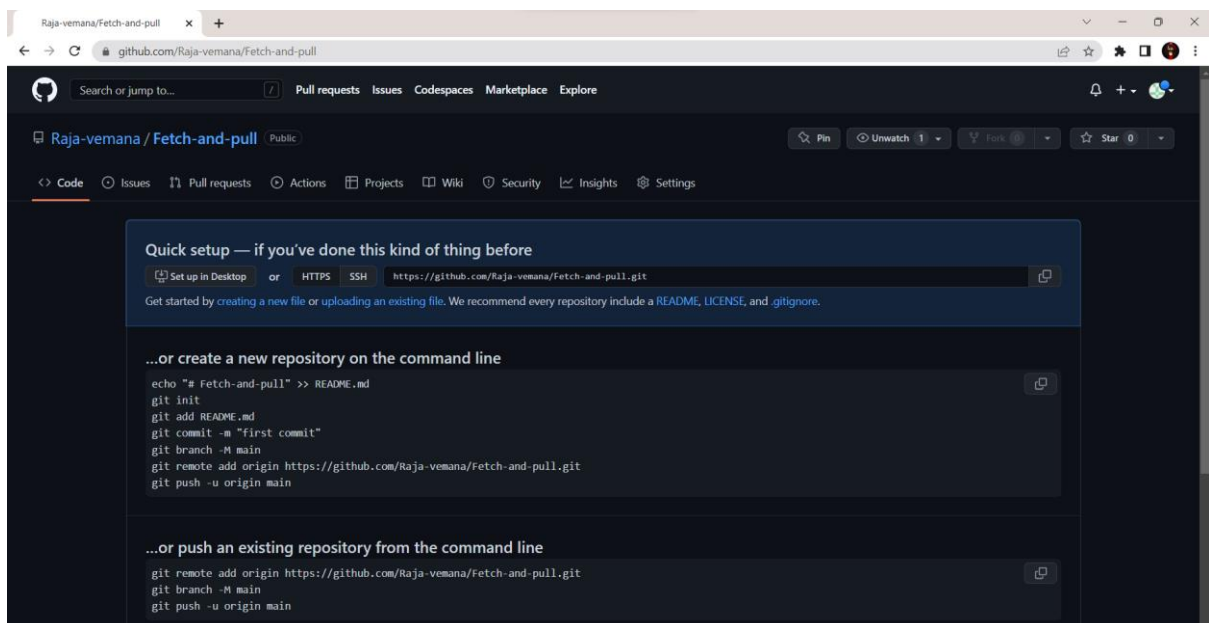
```

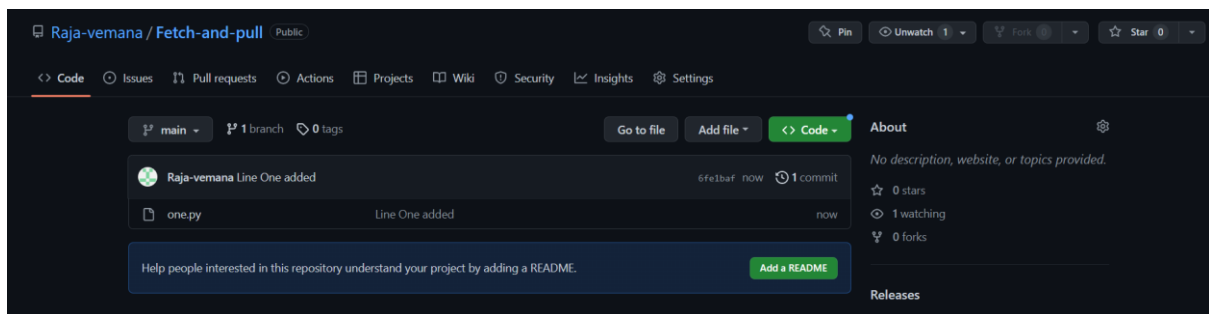
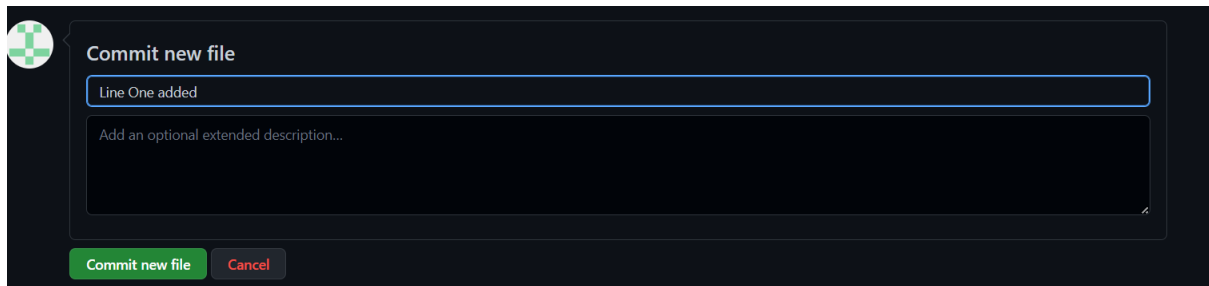
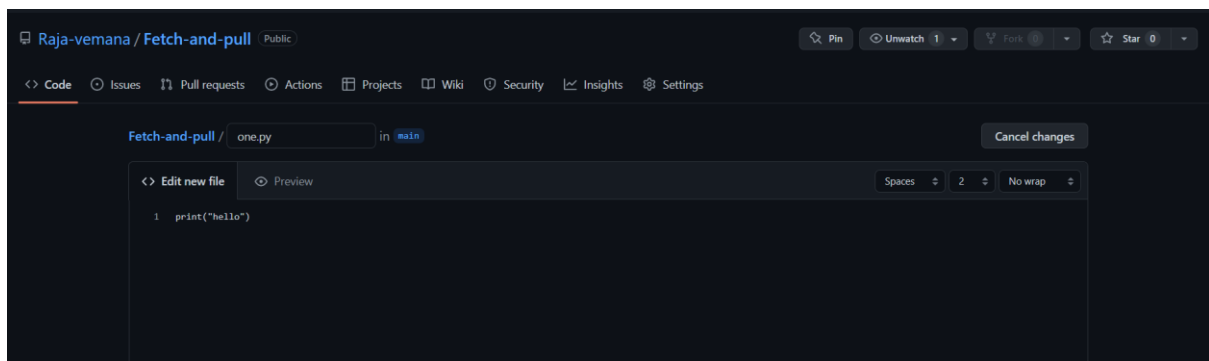
```

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline --all

```

*Create a file in the github.





```
raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline --all
```

We cannot find any commits in our local repository even if we made a commit in remote repository.

*Git fetch:

Let us fetch the repository. It will download all the changes that are made in the remote repository but doesn't merge our changes with working files.

```
raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git fetch
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 595 bytes | 28.00 KiB/s, done.
From https://github.com/Raja-vemana/Fetch-and-pull
* [new branch]      main      -> origin/main
```

*Use the git log --oneline --all command to check whether the changes have been downloaded or not.

```

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline --all
6fe1baf (origin/main) Line One added

```

*Here using git fetch the commits are not applied to the main branch.

****Git merge:** Git merging is basically to merge multiple sequences of commits, stored in multiple branches in a unified history or to be simple you can say in a single branch.

*It can be used to merge the changes that are made in the remote repository to the local repository after fetching the changes.

```

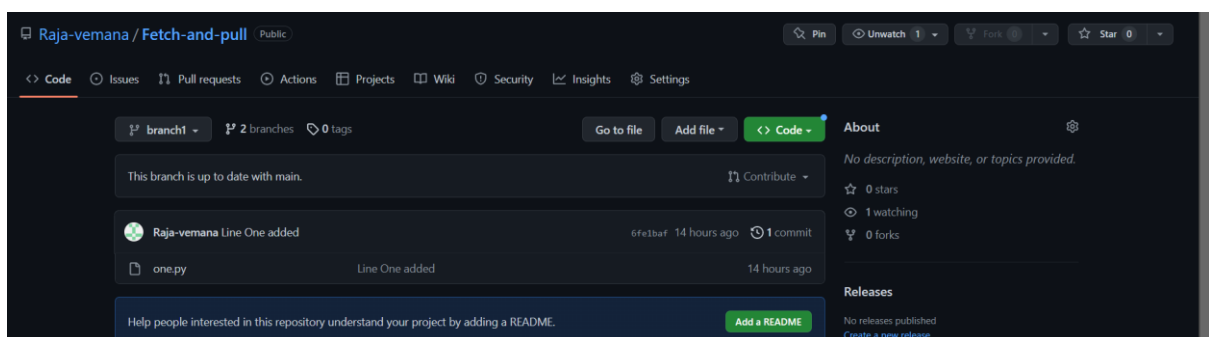
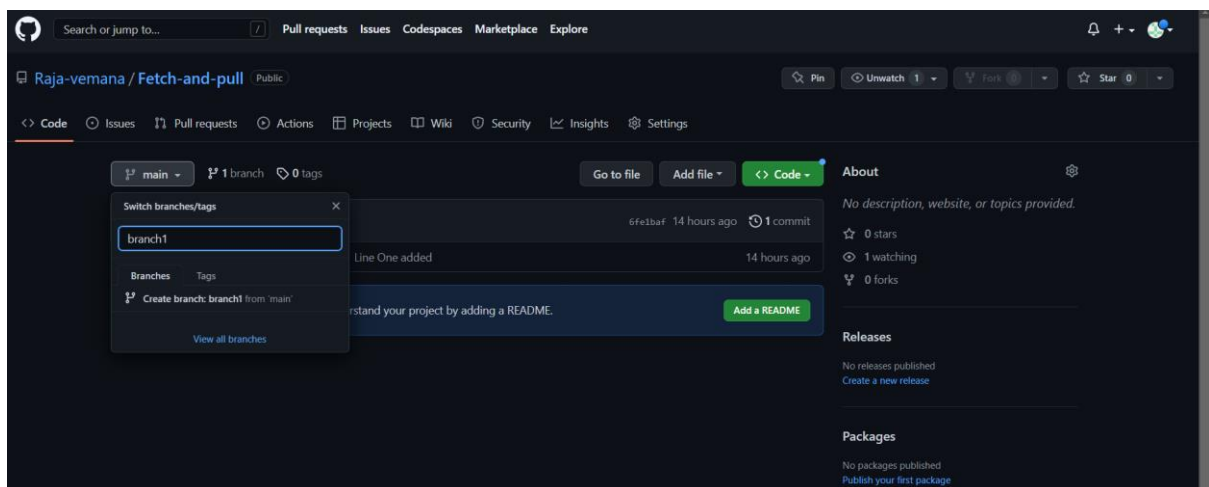
raja vemana@Raja-PC MINGW64 ~ (master)
$ cd desktop

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ cd Fetch-and-pull

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline
6fe1baf (HEAD -> main, origin/main) Line One added

```

*Now create a new branch in the remote repository and make a new commit.



*Modify the file one.py and make a commit.

Raja-vemana / Fetch-and-pull Public

Pin Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Fetch-and-pull / one.py in branch1 Cancel changes

Edit file Preview changes Spaces 2 No wrap

```
1 print("hello")
2 print("Line2 added")
```

Commit changes

added line2

Add an optional extended description...

☒ Commit directly to the branch1 branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

Raja-vemana / Fetch-and-pull Public

Pin Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

branch1 had recent pushes less than a minute ago Compare & pull request

branch1 2 branches 0 tags Go to file Add file Code

This branch is 1 commit ahead of main. Contribute

Raja-vemana added line2 189f657 now 2 commits

one.py added line2 now

Help people interested in this repository understand your project by adding a README. Add a README

About No description, website, or topics provided. 0 stars 1 watching 0 forks

Releases No releases published Create a new release

Packages No packages published Publish your first package

Languages

*We found that the branch1 is 1 commit ahead of the main

```

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 648 bytes | 27.00 KiB/s, done.
From https://github.com/Raja-vemana/Fetch-and-pull
* [new branch]      branch1      -> origin/branch1

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git branch
* main

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git merge origin/branch1
Updating 6fe1baf..1d0f057
Fast-forward
 one.py | 1 +
 1 file changed, 1 insertion(+)

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline
1d0f057 (HEAD -> main, origin/branch1) added line2
6fe1baf (origin/main) Line One added

```

We can see that the branch1 is merged with main branch and the commit made in the branch1 is also merged with the main branch.

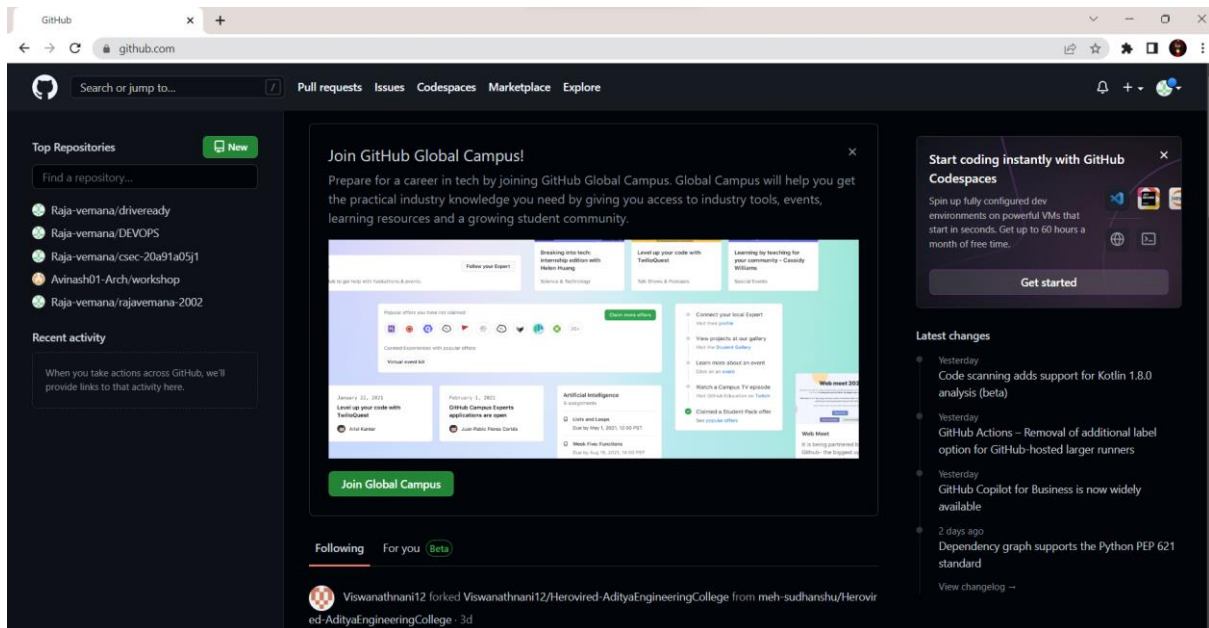
Q3. State the difference between git fetch and git pull by doing a practical example in your git bash and attach a screenshot of all the processes.

Git fetch: Git fetch is a command that allows you to download objects from remote repository but it doesn't integrate any of this new data into your working files.

Git pull: Git pull is a command that allows you to fetch from and integrate with another repository or local branch. It update your current HEAD branch with the latest changes from the remote server.

*We can say that git pull is a git fetch followed by an additional action say git merge.



*Create a new repository in github.



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 ***

 Raja-vemana / 

Great repository names are short, lowercase, and use hyphens. Your new repository will be created as **Fetch-and-pull**. [Learn more.](#)

Description (optional)

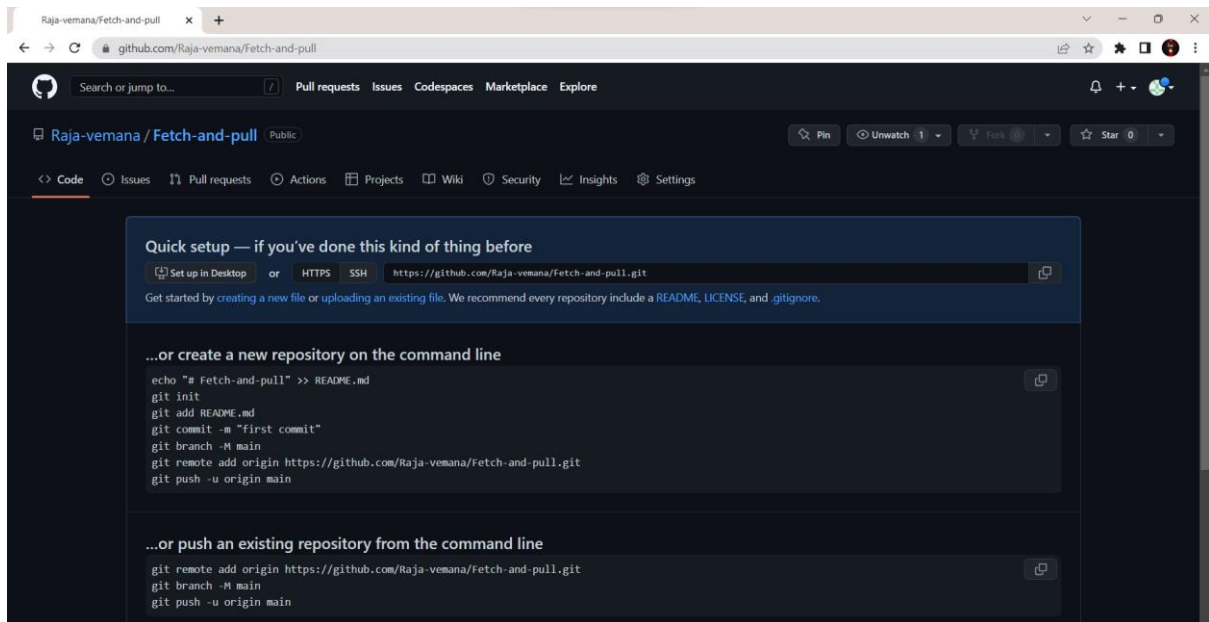
☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)



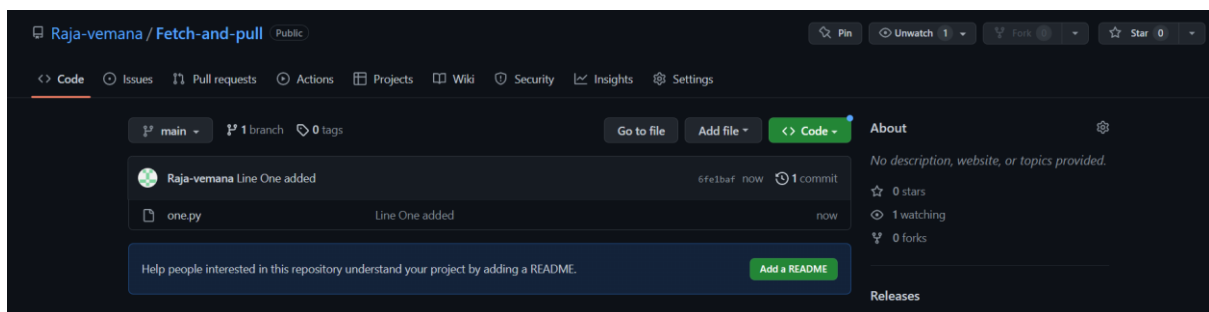
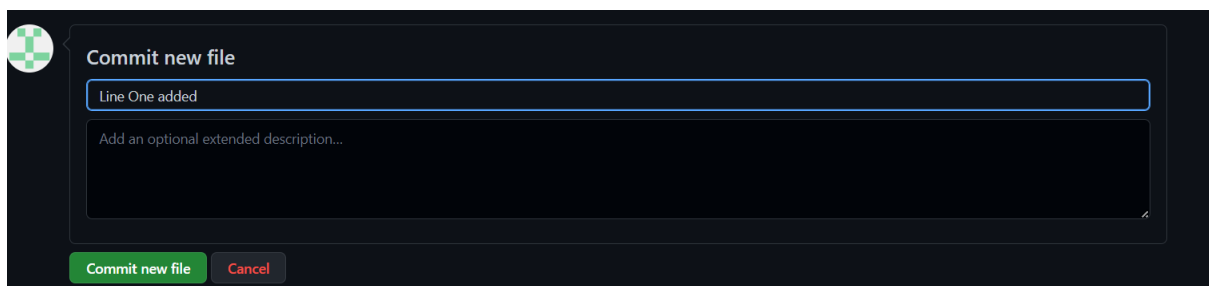
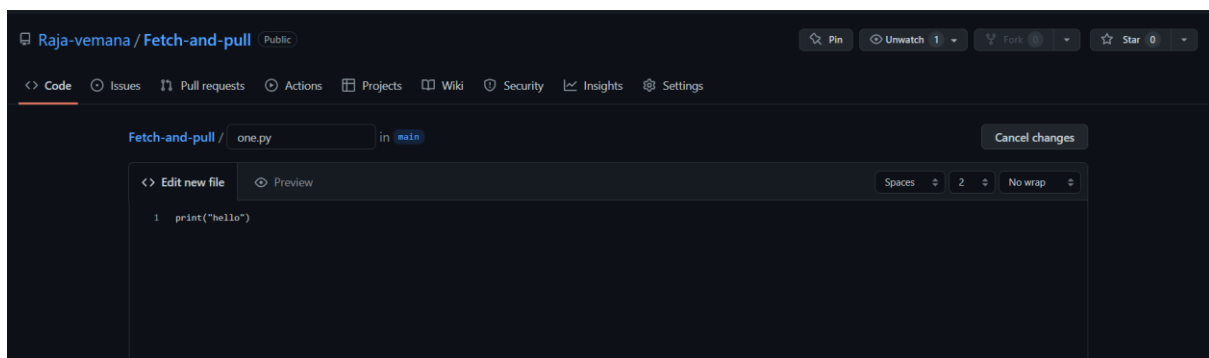
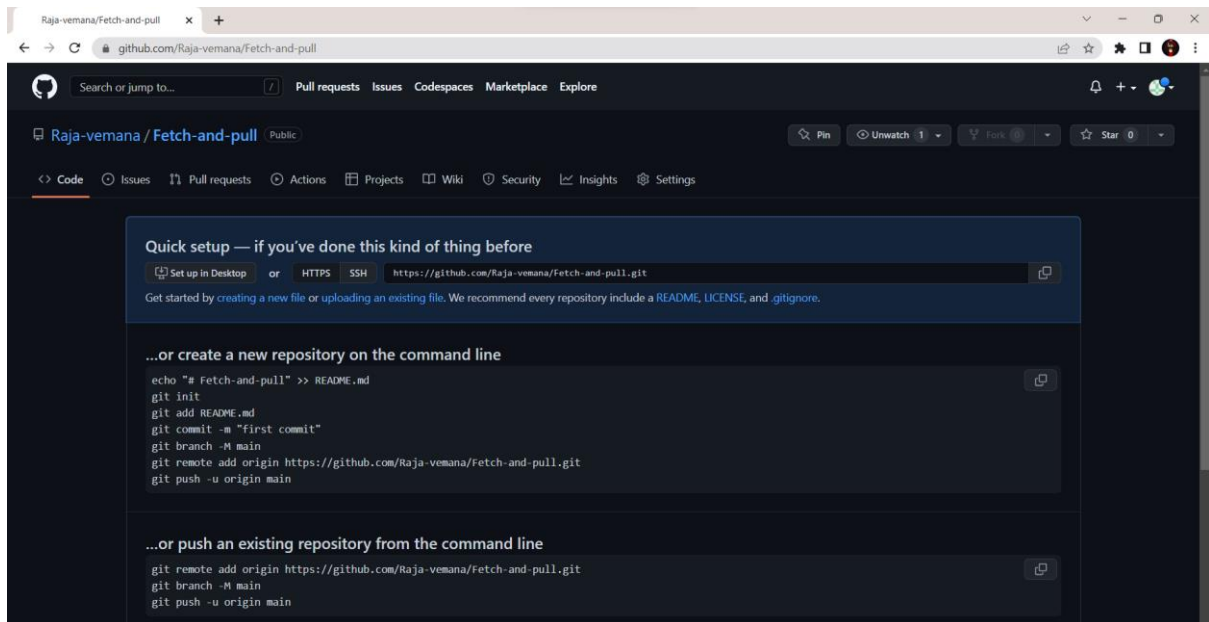
*Clone the empty remote repository into local repository using git bash.

```
raja vemana@Raja-PC MINGW64 ~  
$ cd desktop  
  
raja vemana@Raja-PC MINGW64 ~/desktop (master)  
$ git init  
Reinitialized existing Git repository in C:/Users/rajav/Desktop/.  
  
raja vemana@Raja-PC MINGW64 ~/desktop (master)  
$ git config --global user.name "Raja-vemana"  
  
raja vemana@Raja-PC MINGW64 ~/desktop (master)  
$ git config --global user.email "rajavemana2002@gmail.com"  
  
raja vemana@Raja-PC MINGW64 ~/desktop (master)  
$ git clone https://github.com/Raja-vemana/Fetch-and-pull.git  
Cloning into 'Fetch-and-pull'...  
warning: You appear to have cloned an empty repository.
```

*Check if there are any commits present in the repository.

```
raja vemana@Raja-PC MINGW64 ~/desktop (master)  
$ cd Fetch-and-pull  
  
raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)  
$ git log --oneline --all
```

*Create a file in the github.



We cannot find any commits in our local repository even if we made a commit in remote repository.

*Git fetch:

Let us fetch the repository. It will download all the changes that are made in the remote repository but doesn't merge our changes with working files.

```
raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git fetch
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 595 bytes | 28.00 KiB/s, done.
From https://github.com/Raja-vemana/Fetch-and-pull
* [new branch]      main      -> origin/main
```

*Use the git log --oneline --all command to check whether the changes have been downloaded or not.

```
raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline --all
6fe1baf (origin/main) Line One added
```

*Here using git fetch the commits are not applied to the main branch.

***Git pull:** It downloads and also applies the changes to the working files.

```
raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git pull

raja vemana@Raja-PC MINGW64 ~/desktop/Fetch-and-pull (main)
$ git log --oneline --all
6fe1baf (HEAD -> main, origin/main) Line One added
```

Here the HEAD->main is present which indicates that the changes have been applied to the present branch.

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:

The Awk is a powerful scripting language used for **text scripting**. It searches and replaces the texts and sorts, validates, and indexes the database. It performs various actions on a file like searching a specified text and more.

```

raja vemana@Raja-PC MINGW64 ~ (master)
$ awk '{ print "printing a line using AWK command"}'

printing a line using AWK command

printing a line using AWK command

printing a line using AWK command

raja vemana@Raja-PC MINGW64 ~ (master)
$ cd desktop

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ vi student

```

Name	Marks
Raja	98
Rushi	89
Saranya	96
Eswar	63
Ram	73

```

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ awk '{print}' student
Name      Marks
Raja      98
Rushi     89
Saranya   96
Eswar     63
Ram       73

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ awk '/96/ {print}' student
Saranya 96

```

```

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ awk '{print $1}' student
Name
Raja
Rushi
Saranya
Eswar
Ram

```

*The above command will print column1

***\$NF**: Used to display the last field of the file.

```

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ awk '{print $NF}' student
Marks
98
89
96
63
73

```

Steps to follow bash scripting:

Step1) create the file with extension .sh.

Step 2)open the shell and write the script.

Step 3)save the code and run the code .

To run the run a code

Syntax: bash filename.sh

```

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ vi prime.sh

```

```

for((i=2;i<=20;))
do
    for((j=i-1;j>=2;))
    do
        if [ `expr $i % $j` -ne 0 ] ; then
            prime=1
        else
            prime=0
            break
        fi
        j=`expr $j - 1`
    done
    if [ $prime -eq 1 ] ; then
        echo $i
    fi
    i=`expr $i + 1`
done
~
~
~
~
~
prime.sh[+] [unix] (05:29 01/01/1970) 17,5 All
-- INSERT --

```

```

raja vemana@Raja-PC MINGW64 ~/desktop (master)
$ bash prime.sh
prime.sh: line 13: [: -eq: unary operator expected
3
5
7
11
13
17
19

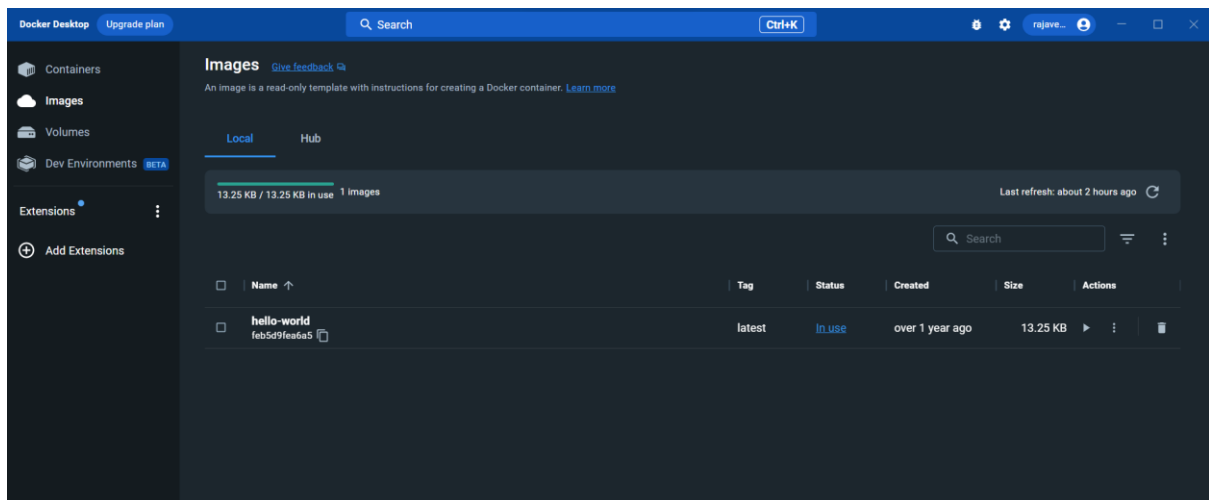
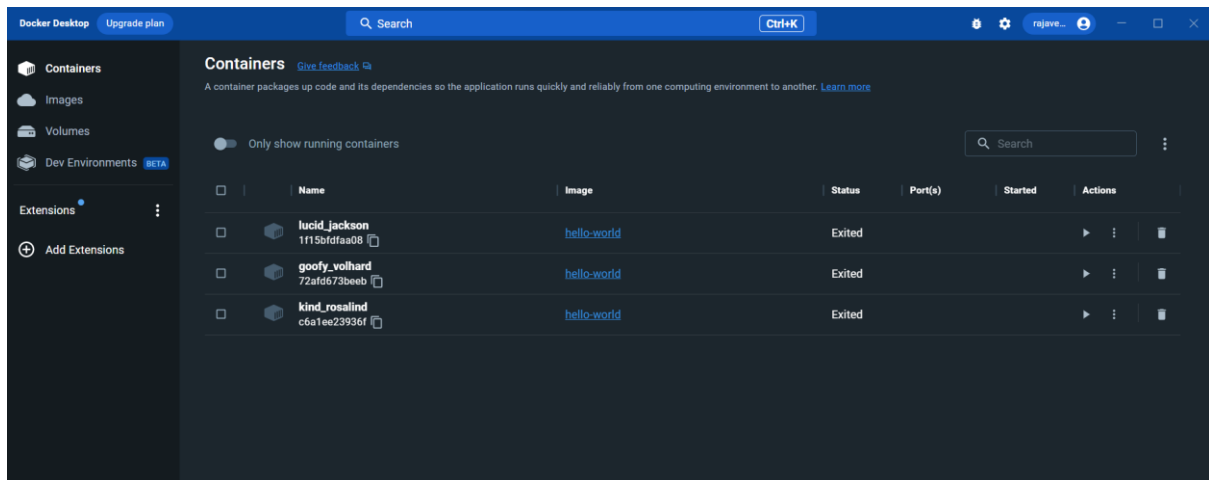
```

Q5. Set up a container and run a Ubuntu operating system. For this purpose, you can make use of the docker hub and run the container in interactive mode. All the processes pertaining to this should be provided in a screenshot for grading.

Image: Images are used to create containers. It uses a private container registry to share container images within the enterprise and also use public container registry to share container images with whole world.

Container: Containers are used to hold the entire package that is needed to run the application. We can say that the image is a template and the container is a copy of the template.

*These are the containers present in the docker desktop.



*For setting up a container and run the ubuntu os,

→First we need to download the image of Ubuntu from docker hub using the command **docker pull ubuntu** .

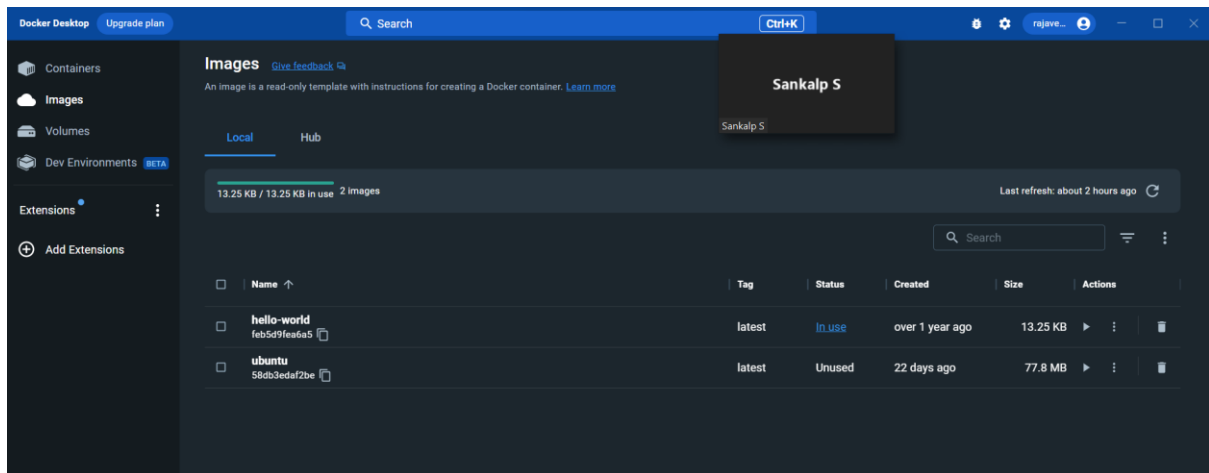
→To create a container and execute the image use the command **docker run -it ubuntu** .

→To get an idea about the available update use **apt update** command.

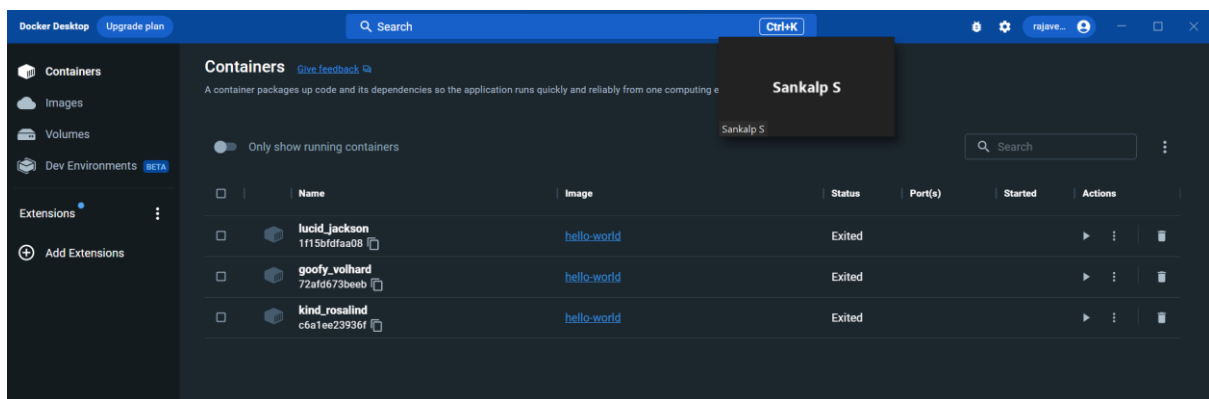
**Download the ubuntu OS image from the docker hub.

```

raja vemana@Raja-PC MINGW64 ~ (master)
$ docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
677076032cca: Pulling fs layer
677076032cca: Download complete
677076032cca: Pull complete
Digest: sha256:9a0bdd4e4188b896a372804be2384015e90e3f84906b750c1a53539b585fbbe7f
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
  
```



*An image ubuntu got downloaded but a container is not created.



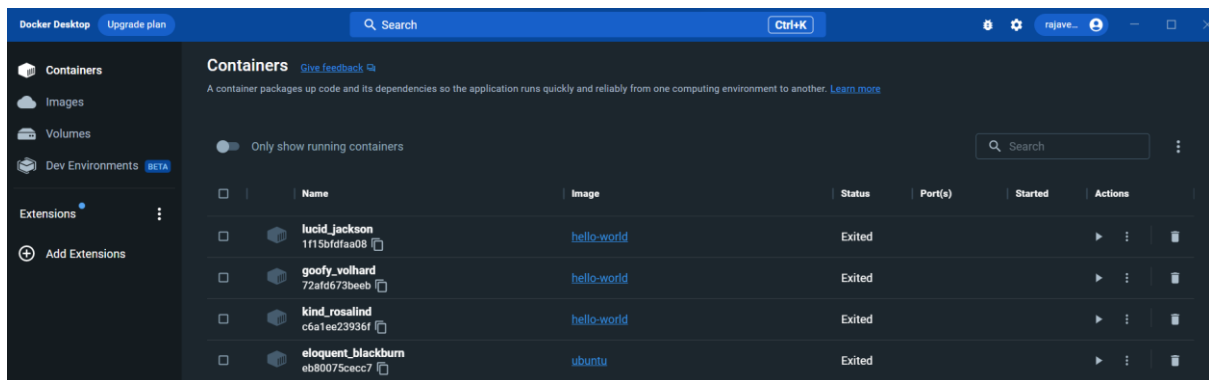
*Now run the ubuntu image which has been downloaded.

```
raja vemana@Raja-PC MINGW64 ~ (master)
$ docker run ubuntu
```

```

C:\Users\rajav>docker run -it ubuntu
root@67dbf50a1171:/# apt update
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy InRelease [270 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [5557 B]
Get:4 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [807 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [107 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [17.5 MB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [860 kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [752 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy/main amd64 Packages [1792 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [266 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/restricted amd64 Packages [164 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1136 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [808 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1091 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]
Fetched 25.8 MB in 38s (679 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
5 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@67dbf50a1171:/#

```



**Now a container got created for the ubuntu image.