



Academic Year: 2024-25

Semester: V

Class / Branch: TEIT

Subject: DevOps Lab

Name of Instructor: Ms. Sujata Oak/ Ms. Sonal Jain

Experiment No. 2

Aim: To perform installation of Git and work on local and remote Git repositories.

GIT is a Version Control System (VCS) (aka Revision Control System (RCS), Source Code Manager (SCM)). A VCS serves as a Repository (or repo) of program codes, including all the historical revisions. It records changes to files at so-called commits in a log so that you can recall any file at any commit point.

To issue a command, start a "Terminal" (for Ubuntu/Mac) or "Git Bash" (for Windows):

```
$ git <command> <arguments>
```

The commonly-used commands are:

1. **init, clone, config**: for starting a Git-managed project.
2. **add, mv, rm**: for staging file changes.
3. **commit, rebase, reset, tag**:
4. **status, log, diff, grep, show**: show status
5. **checkout, branch, merge, push, fetch, pull**

Getting Started with Local Repo

There are 2 ways to start a Git-managed project:

1. Starting your own project;
2. Cloning an existing project from a GIT host.

Git uses two stages to commit file changes:

1. "git add <file>" to stage file changes into the staging area, and
2. "git commit" to commit ALL the file changes in the staging area to the local repo.



You need to setup Git on your local machine, as follows:

To update all local package index for ubuntu:

`$sudo apt update`

```
sujata@Ubuntu:~$ sudo apt update
[sudo] password for sujata:
sujata is not in the sudoers file. This incident will be reported.
```

```
sujata@Ubuntu:~$ su root
Password:
root@Ubuntu:/home/sujata#
```

```
root@Ubuntu:/home/sujata# sudo apt update
Hit:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:3 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
312 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Step1: Download & Install:

- For Ubuntu, issue command "`sudo apt-get install git`".
- `#apt install git`

```
root@Ubuntu:/home/sujata# apt install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
```

To check version of git:

`#git --version`

Compiled by Prof. Sujata Oak



```
root@Ubuntu:/home/sujata# git --version
git version 2.25.1
```

Step 2: Customize and configure your Git Account:

```
#git config --global user.name "sujataoak799"
```

```
root@Ubuntu:/home/sujata# git config --global user.name "sujataoak799"
```

```
#git config --global user.email sujataoak2021@gmail.com
```

```
root@Ubuntu:/home/sujata# git config --global user.email "sujataoak2021@gmail.com"
root@Ubuntu:/home/sujata#
```

To List Global configuration for Git:

```
#git config --list
```

```
root@Ubuntu:/home/sujata# git config --list
user.name=sujataoak799
user.email=sujataoak2021@gmail.com
```

Step . 3

To Integrate Git account with Github:

Goto www.github.com

Sign-in to your account

Create a Repository



Create a new repository

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

Required fields are marked with an asterisk (*).

Repository template

No template ▾

Start your repository with a template repository's contents.

Owner *

 sujataoak799 ▾

Repository name *

hello_test

hello_test is available.

Great repository names are short and memorable. Need inspiration? How about **fuzzy-system** ?

Description (optional)

Setting up Git ✓

☒



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:

☒ Add a README file ✓

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

Add .gitignore

.gitignore template: None ▾

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

Choose a license

License: None ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

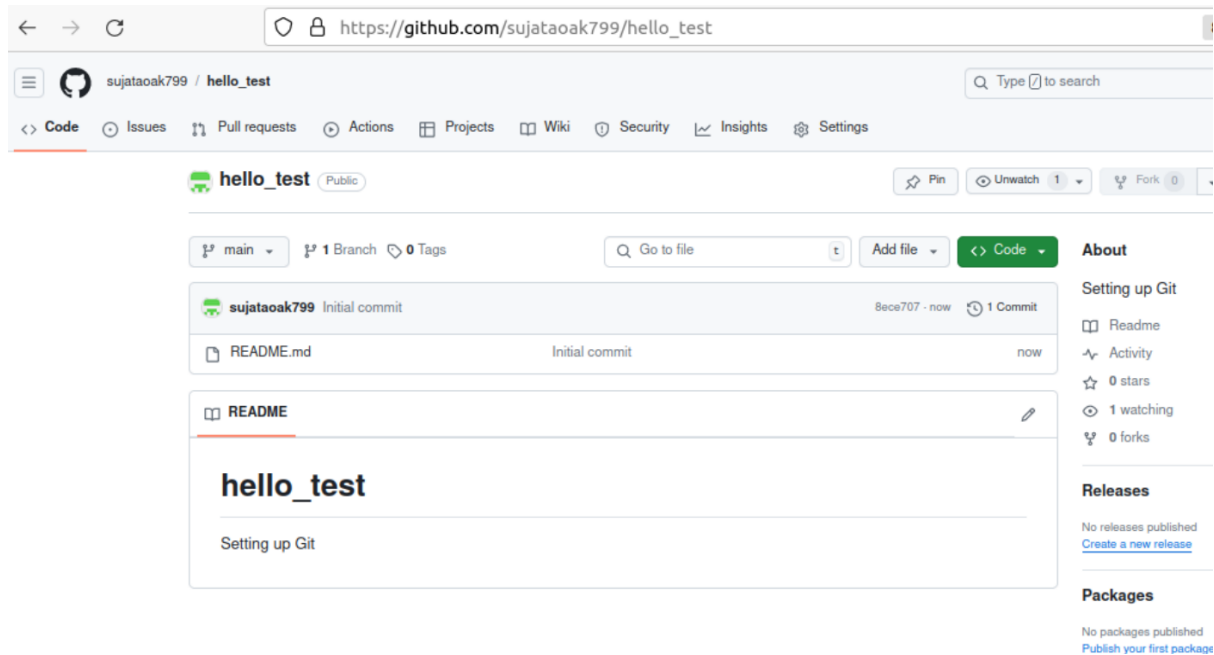
This will set **Pmain** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository



PARSHVANATH CHARITABLE TRUST'S
A. P. SHAH INSTITUTE OF TECHNOLOGY
Department of Information Technology
(NBA Accredited)



Step . 3

Now Clone the Remote Repository into your Local Repository ie; Ubuntu Operating System:
Firstly create a workspace:

```
root@Ubuntu:/home/sujata# mkdir workspace
root@Ubuntu:/home/sujata# cd workspace/
root@Ubuntu:/home/sujata/workspace#
```

- Initiate that directory to make it a git repository (.git file must be added inside that folder after initiation)

#git init

```
root@Ubuntu:/home/sujata/workspace# git init
Initialized empty Git repository in /home/sujata/workspace/.git/
```




PARSHVANATH CHARITABLE TRUST'S

A. P. SHAH INSTITUTE OF TECHNOLOGY

Department of Information Technology

(NBA Accredited)



```
root@Ubuntu:/home/sujata/workspace# ls
root@Ubuntu:/home/sujata/workspace# git clone https://github.com/sujataoak799/hello_test.git
Cloning into 'hello_test'...
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), 881 bytes | 881.00 KiB/s, done.
root@Ubuntu:/home/sujata/workspace#
```

```
root@Ubuntu:/home/sujata/workspace# ls
hello_test
```

```
root@Ubuntu:/home/sujata/workspace# cd hello_test/
root@Ubuntu:/home/sujata/workspace/hello_test# ls
README.md
root@Ubuntu:/home/sujata/workspace/hello_test# cat README.md
# hello_test
Setting up Git
```

Now Create a python file hello.py in Local Repository:

```
root@Ubuntu:/home/sujata/workspace/hello_test# touch hello.py
root@Ubuntu:/home/sujata/workspace/hello_test# ls
hello.py README.md
```

```
root@Ubuntu:/home/sujata/workspace/hello_test# gedit hello.py
```

```
# factorial of given number
def factorial(n):
    if n < 0:
        return 0
    elif n == 0 or n == 1:
```

Compiled by Prof. Sujata Oak



```
        return 1
    else:
        fact = 1
        while(n > 1):
            fact *= n
            n -= 1
        return fact

# Driver Code
num = 5
print("Factorial of",num,"is",
factorial(num))
```

Step 4: Now we will apply some git commands to add, commit and push hello.py file to remote repository: Firstly To View Uncommitted File:

#git status

```
root@Ubuntu:/home/sujata/workspace/hello_test# git status
On branch main
Your branch is up to date with 'origin/main'.

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

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

The file hello.py shown in red color is untracked (it means not tracked by git till now). So to add to the git versioning : git add <filename>

```
root@Ubuntu:/home/sujata/workspace/hello_test# git add hello.py
```

```
root@Ubuntu:/home/sujata/workspace/hello_test# git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   hello.py
```



Once file is added to git, the color changes to green color. But it says changes to be committed

Step 5: To start first commit

```
git commit -m "First Python File"
```

```
root@Ubuntu:/home/sujata/workspace/hello_test# git commit -m " First Python File"
[main e19123e] First Python File
1 file changed, 10 insertions(+)
create mode 100644 hello.py
```

```
root@Ubuntu:/home/sujata/workspace/hello_test# git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Step 6: To Push your changes to Github Repository:

```
git push origin master or git push origin main
```

#git push origin main

Enter username and password

```
root@Ubuntu:/home/sujata/workspace/hello_test# git push origin main
Username for 'https://github.com': sujataoak799
Password for 'https://sujataoak799@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-ur
ls for information on currently recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/sujataoak799/hello_test.git/'
root@Ubuntu:/home/sujata/workspace/hello_test#
```

It says support for password authentication was removed on August 13, 2021.
So we need a new kind of method:

Goto your Github Account → User Profile → Settings → Developer
Settings → Personal Access Token → Token(classic) → Generate New

Compiled by Prof. Sujata Oak



token → Generate New Token(Classic)

← → ↻

GitHub Apps
OAuth Apps
Personal access tokens
Fine-grained tokens (Beta)
Tokens (classic)

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

My Token

What's this token for?

Expiration *

Custom... 13 / 07 / 2024

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo:invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input checked="" type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects

Click on Generate Token

Copy the token and paste it some location:

#git remote set-url origin https://tokenhere@github.com/user_name/repo_name

```
root@Ubuntu:/home/sujata/workspace/hello_test# git remote set-url origin https://ghp_ppTKMIcg1amhsE9erCkvBsk38vmy3e3HLLre@github.com/sujataoak799/hello_test
```

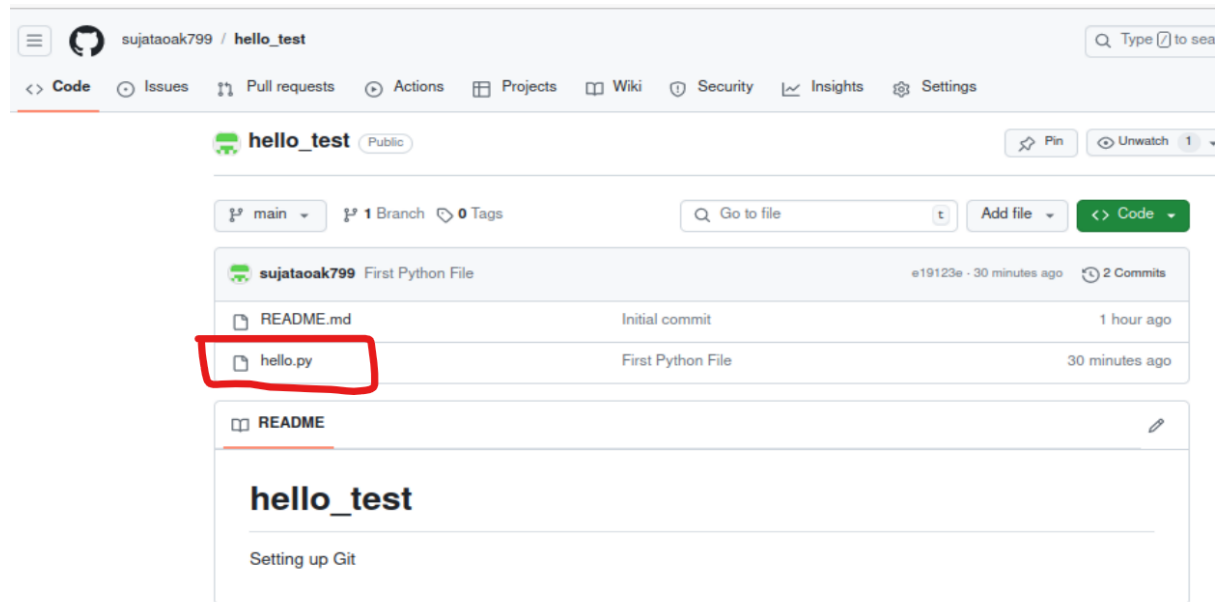
Again try to Push your changes to Github Repository:

#git push origin main

```
root@Ubuntu:/home/sujata/workspace/hello_test# git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 449 bytes | 449.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/sujataoak799/hello_test
8ece707..e19123e main -> main
```



This time it is successfully implemented.
Goto Remote Repository and see the file hello.py



Step7 : To see the logs in online like username, email -id, date, time of creation.

`git log`

or

`git log --online`

```
root@Ubuntu:/home/sujata/workspace/hello_test# git log
commit e19123efabe5c27c0ad82109b2cc1e14be02074d (HEAD -> main, origin/main, origin/HEAD)
Author: sujataoak799 <sujataoak2021@gmail.com>
Date: Tue Jul 9 23:36:03 2024 +0530

    First Python File

commit 8ece7077e5801bfceaae96270dff81d3f4a4efdb
Author: sujataoak799 <79905110+sujataoak799@users.noreply.github.com>
Date: Tue Jul 9 22:51:50 2024 +0530

    Initial commit
```

Step 8: To show repository id and other detail



#git show

```
root@Ubuntu:/home/sujata/workspace/hello_test# git show
commit e19123efabe5c27c0ad82109b2cc1e14be02074d (HEAD -> main, origin/main, origin/HEAD)
Author: sujataoak799 <sujataoak2021@gmail.com>
Date: Tue Jul 9 23:36:03 2024 +0530

    First Python File

diff --git a/hello.py b/hello.py
new file mode 100644
index 00000000..50d5ba4
--- /dev/null
+++ b/hello.py
@@ -0,0 +1,10 @@
+ num = int(input("Enter a number: "))
+ factorial = 1
+ if num < 0:
+     print(" Factorial does not exist for negative numbers")
+ elif num == 0:
+     print("The factorial of 0 is 1")
+ else:
+     for i in range(1,num + 1):
+         factorial = factorial*i
+     print("The factorial of",num,"is",factorial)
```

Step 09: To see the difference in the content of file between first and second commit.

#git diff



PARSHVANATH CHARITABLE TRUST'S

A. P. SHAH INSTITUTE OF TECHNOLOGY

Department of Information Technology

(NBA Accredited)



```
root@Ubuntu:/home/sujata/workspace/hello_test# git diff 8ece707 e19123e
diff --git a/hello.py b/hello.py
new file mode 100644
index 0000000..50d5ba4
--- /dev/null
+++ b/hello.py
@@ -0,0 +1,10 @@
+ num = int(input("Enter a number: "))
+ factorial = 1
+ if num < 0:
+     print(" Factorial does not exist for negative numbers")
+ elif num == 0:
+     print("The factorial of 0 is 1")
+ else:
+     for i in range(1,num + 1):
+         factorial = factorial*i
+     print("The factorial of",num,"is",factorial)
```

Step 10: Creating a latter commit and reverting back to see the initial/original content

```
git revert<secondcommitID>
```

```
GNU nano 4.8 /home/sujata/workspace/hello_test/.git/C
Revert "Initial commit"

This reverts commit 8ece7077e5801bfceaae96270dff81d3f4a4efdb.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
# Your branch is up to date with 'origin/main'.
#
# Changes to be committed:
#   deleted:   README.md
#
Reverted Initial commit
```

Save it

Compiled by Prof. Sujata Oak

Department of Information Technology | APSIT



PARSHVANATH CHARITABLE TRUST'S

A. P. SHAH INSTITUTE OF TECHNOLOGY

Department of Information Technology

(NBA Accredited)



```
root@Ubuntu:/home/sujata/workspace/hello_test# git revert 8ece707
Removing README.md
[main f1bdb1c] Revert "Initial commit"
1 file changed, 2 deletions(-)
delete mode 100644 README.md
root@Ubuntu:/home/sujata/workspace/hello_test#
```

Conclusion:

In this experiment, we understood the use case of Version Control System, its benefits in real time scenario which provides a application of reverting the changes when people are in working in a collaborating environment. Different commands were used for the same such as revert(by using its id),diff for displaying the changes between the initial and latter texts.