Subject Name: **Source Code Management**

Subject Code: **24CSE0106**

Session: **2024-25**

Department: **CSE**

Submitted By:
Itishjot Singh
2410990366
G5

Submitted To:
Dr. Chetna Sharma

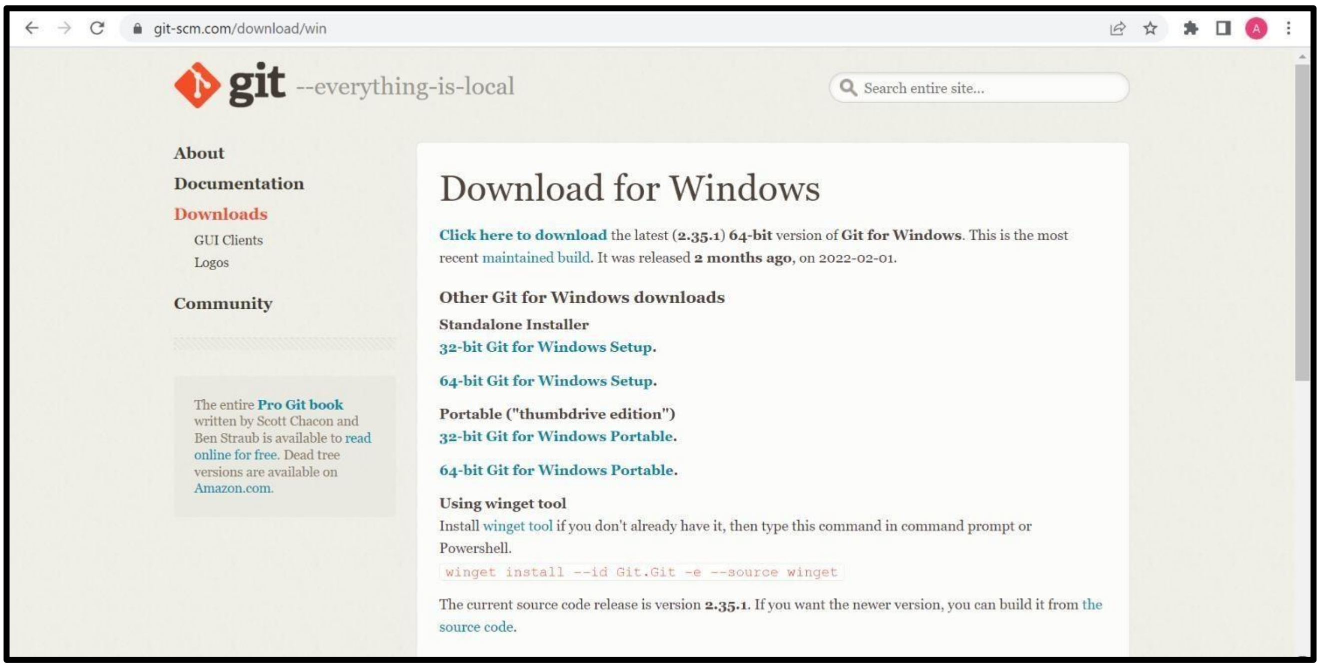
List of Programs

Task 1.1

Practical 1
Aim: To install and configure Git Client on your local system.

Theory:
Git is a distributed version control system used to track changes in source code. This practical focuses on setting up Git on your local system for effective version control.
Procedure:
○ Download Git from git-scm.com.
○ Install Git by following the setup wizard.
○ Open Git Bash and verify installation using the command: git --version.
○ Configure user details using the commands:
git config --global user.name "Your Name"
git config --global user.email "Your Email"

Snapshots of download:



The entire Pro Git book written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

Download for Windows

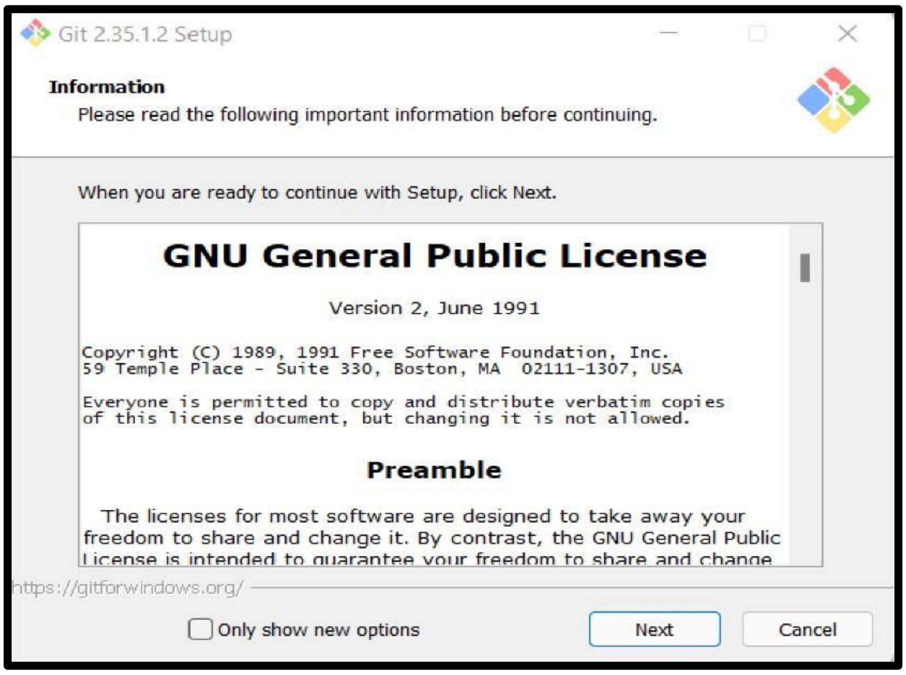
Click here to download the latest (2.35.1) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 2 months ago, on 2022-02-01.

Other Git for Windows downloads
Standalone Installer
32-bit Git for Windows Setup.
64-bit Git for Windows Setup.
Portable ("thumbdrive edition")
32-bit Git for Windows Portable.
64-bit Git for Windows Portable.
Using winget tool
Install winget tool if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version 2.35.1. If you want the newer version, you can build it from the source code.

Opted for “64-bit Git for Windows Setup”



Git 2.35.1.2 Setup

Information

Please read the following important information before continuing.

When you are ready to continue with Setup, click Next.

GNU General Public License

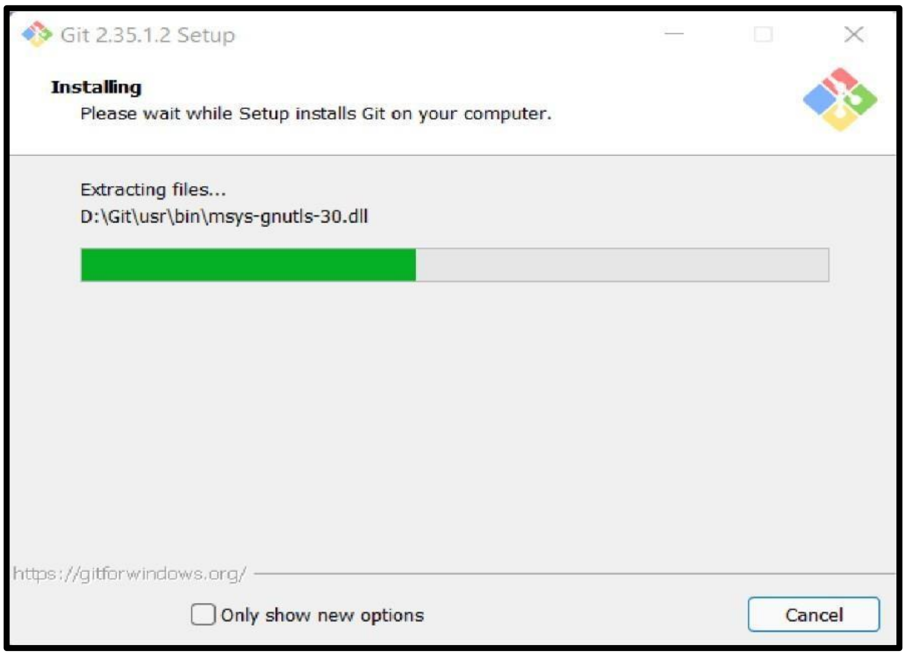
Version 2, June 1991

```
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA  02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change

Git Setup



Git 2.35.1.2 Setup

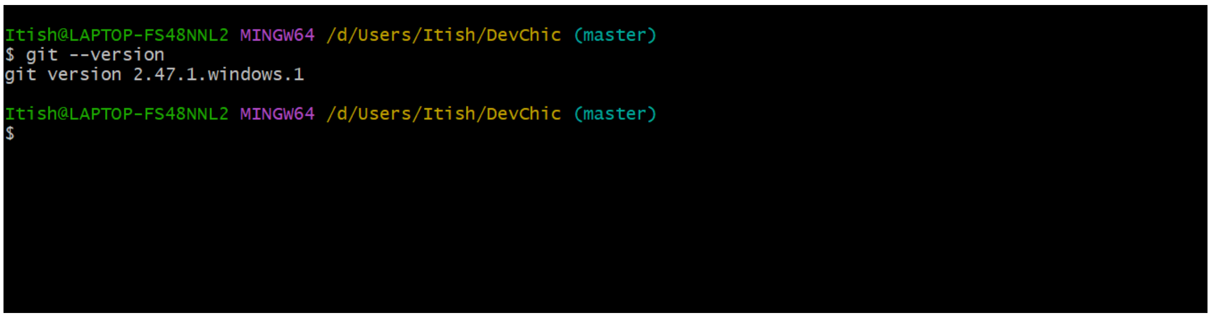
Installing

Please wait while Setup installs Git on your computer.

Extracting files...

D:\Git\usr\bin\msys-gnutls-30.dll

Git Installation



```
Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git --version
git version 2.47.1.windows.1

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$
```

<u>Git Bash version</u>

Practical 2

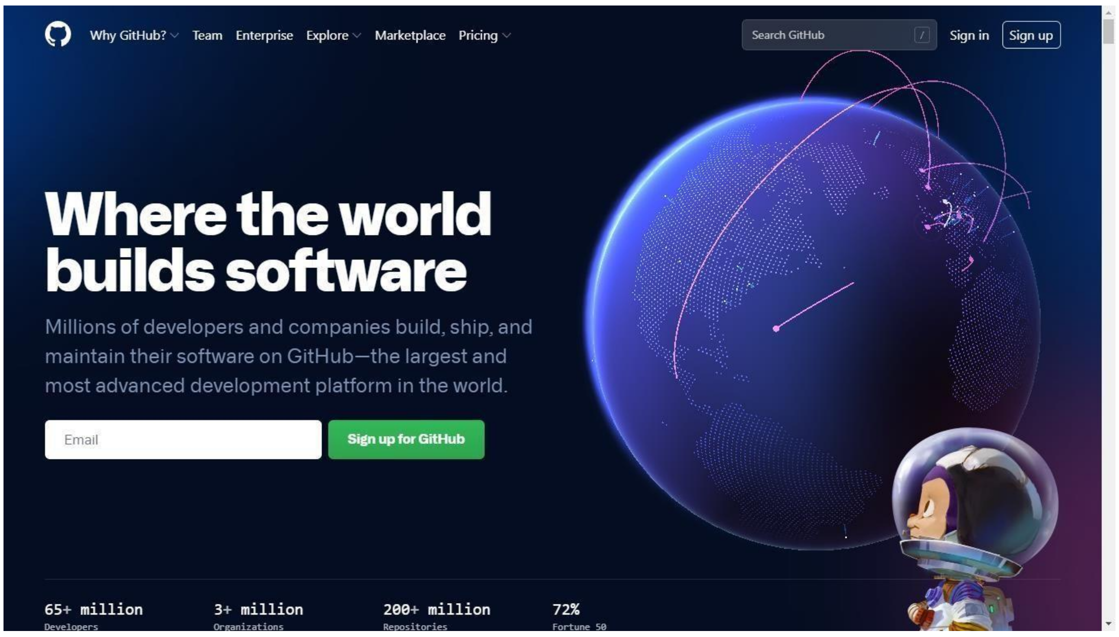
Aim: Setting up GitHub Account

Theory:
GitHub: GitHub is a website and cloud-based service (client) that helps an individual or developers to store and manage their code. We can also track as well as control changes to our or public code.

Advantages of GitHub: GitHub has a user-friendly interface and is easy to use. We can connect the git-hub and git but using some commands shown below in figure 001. Without GitHub we cannot use Git because it generally requires a host and if we are working for a project, we need to share it will our team members, which can only be done by making a repository. Additionally, anyone can sign up and host a public code repository for free, which makes GitHub especially popular with open-source projects.

Procedure:
To make an account on GitHub, we search for GitHub on our browser or visit https://github.com/signup. Then, we will enter our mail ID and create a username and password for a GitHub account.

Snapshots:

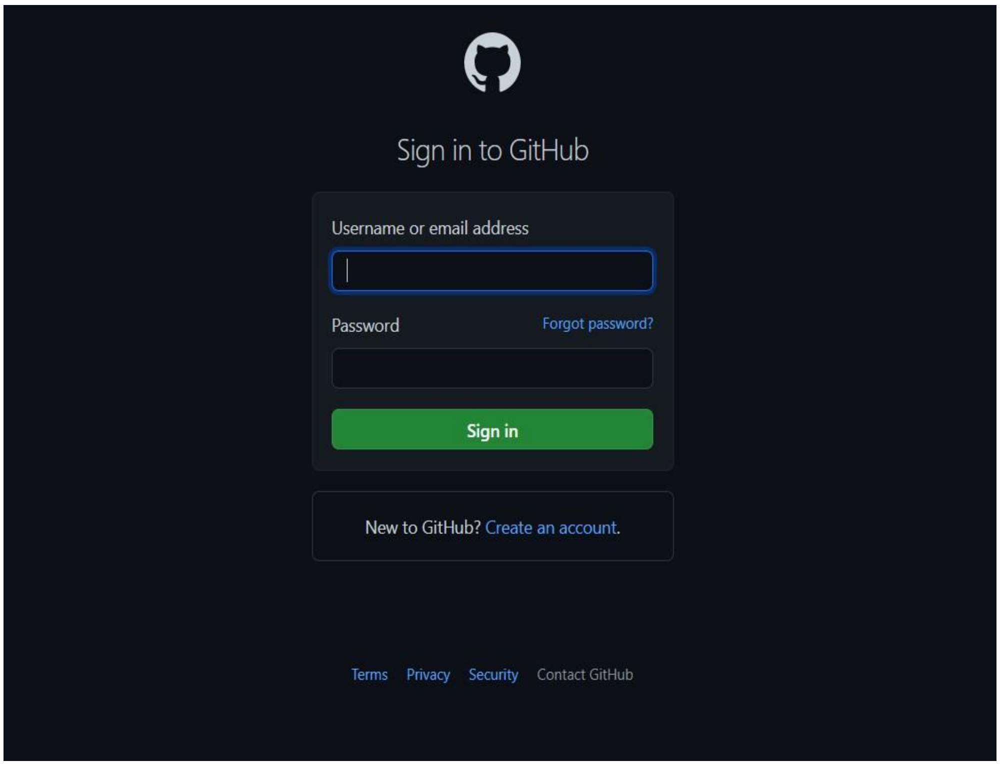


Where the world builds software

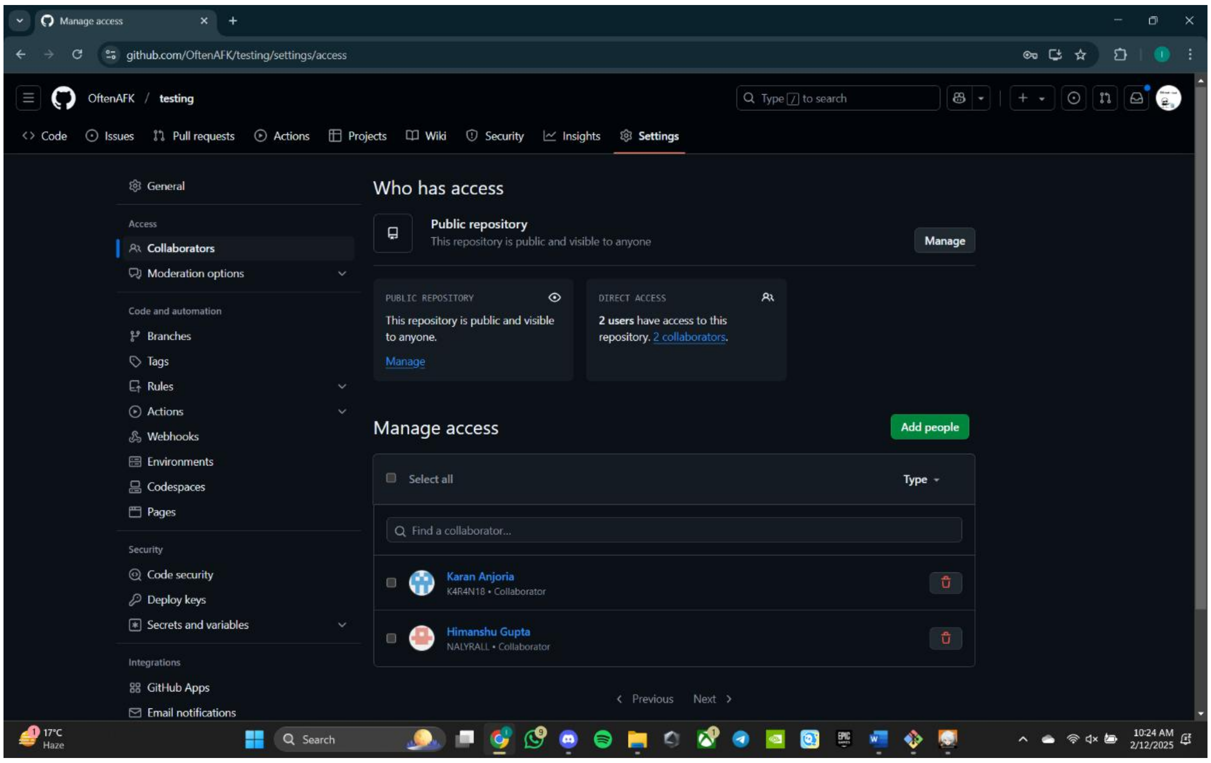
Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.

After visiting the link this type of interface will appear, if you already have an account, you can sign in and if not, you can create.

GitHub Login:



Adding Collaborators:



Who has access

Public repository
This repository is public and visible to anyone

PUBLIC REPOSITORY
This repository is public and visible to anyone.
Manage

DIRECT ACCESS
2 users have access to this repository. 2 collaborators.

Manage access

Karan Anjoria
K4R4N18 • Collaborator

Himanshu Gupta
NALYRALL • Collaborator

Practical 3

Aim: To merge two branches within a Git repository**.**

Theory:
Merging branches in Git allows you to combine changes from one branch into another. It is a fundamental process in collaborative workflows, ensuring all contributions are integrated into a single codebase.

Procedure:

1. Create a new branch and switch to it:
```
git checkout -b new-branch
```

2. Make changes to a file in the new branch and commit them:
```
echo "New content" > file.txt
git add file.txt
git commit -m "Add changes in new branch"
```

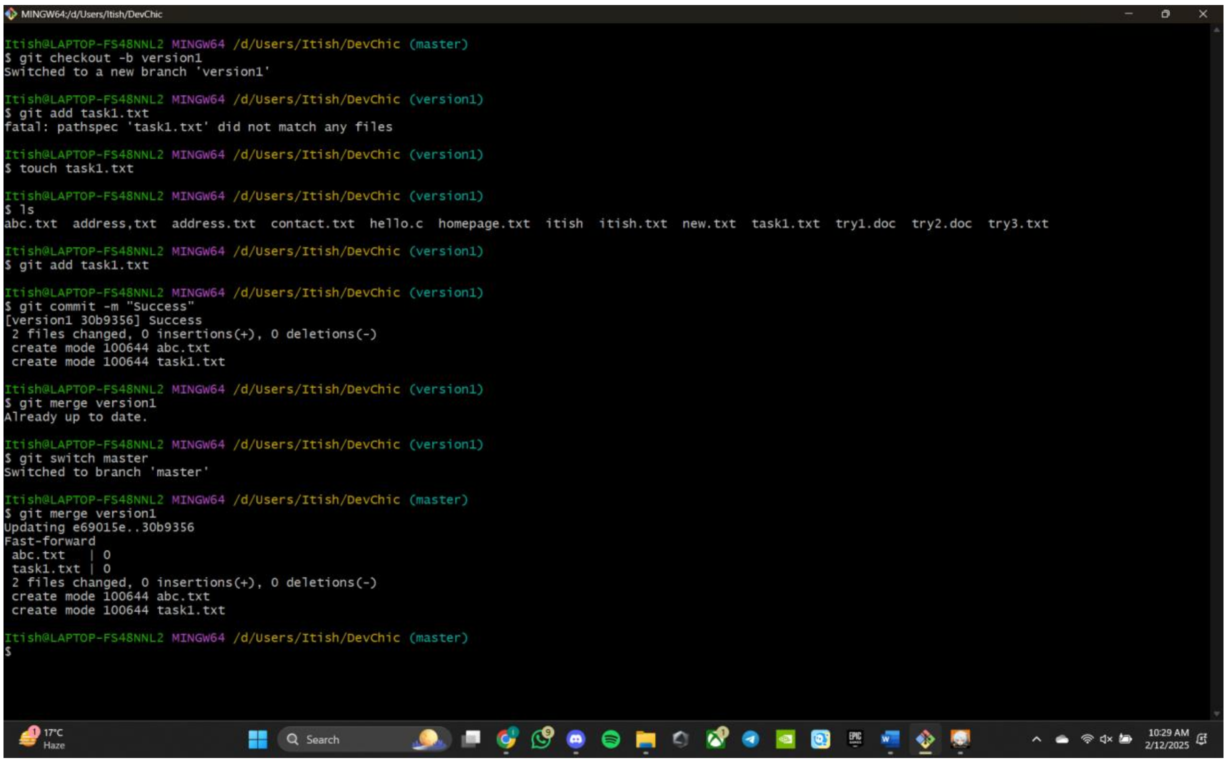
3. Switch back to the main branch:
```
git checkout main
```

4. Modify another file in the main branch and commit the changes:
```
echo "Main branch changes" > another-file.txt
git add another-file.txt
git commit -m "Modify file in main branch"
```

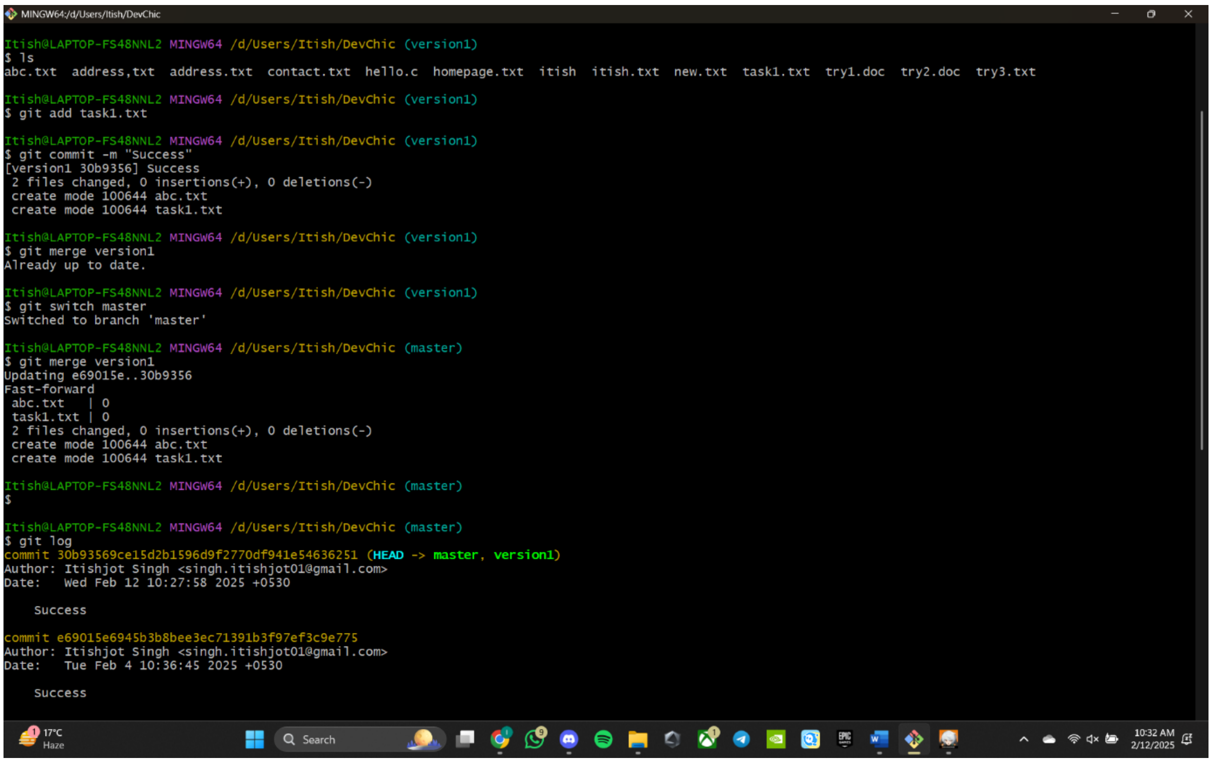
5. Merge the new branch into the main branch:
```
Git merge new-branch
```

```
git merge new-branch
```

Snapshots:



```
Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git checkout -b version1
Switched to a new branch 'version1'

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git add task1.txt
fatal: pathspec 'task1.txt' did not match any files

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ touch task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ ls
abc.txt  address,txt  address.txt  contact.txt  hello.c  homepage.txt  itish  itish.txt  new.txt  task1.txt  try1.doc  try2.doc  try3.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git add task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git commit -m "Success"
[version1 30b9356] Success
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 abc.txt
 create mode 100644 task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git merge version1
Already up to date.

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git switch master
Switched to branch 'master'

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git merge version1
Updating e69015e..30b9356
Fast-forward
 abc.txt   | 0
 task1.txt | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 abc.txt
 create mode 100644 task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$
```



```
Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ ls
abc.txt  address,txt  address.txt  contact.txt  hello.c  homepage.txt  itish  itish.txt  new.txt  task1.txt  try1.doc  try2.doc  try3.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git add task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git commit -m "Success"
[version1 30b9356] Success
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 abc.txt
 create mode 100644 task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git merge version1
Already up to date.

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (version1)
$ git switch master
Switched to branch 'master'

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git merge version1
Updating e69015e..30b9356
Fast-forward
 abc.txt   | 0
 task1.txt | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 abc.txt
 create mode 100644 task1.txt

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git log
commit 30b93569ce15d2b1596d9f2770df941e54636251 (HEAD -> master, version1)
Author: Itishjot Singh <singh.itishjot01@gmail.com>
Date:   Wed Feb 12 10:27:58 2025 +0530

    Success

commit e69015e6945b3b8bee3ec71391b3f97ef3c9e775
Author: Itishjot Singh <singh.itishjot01@gmail.com>
Date:   Tue Feb 4 10:36:45 2025 +0530

    Success
```

Practical 4

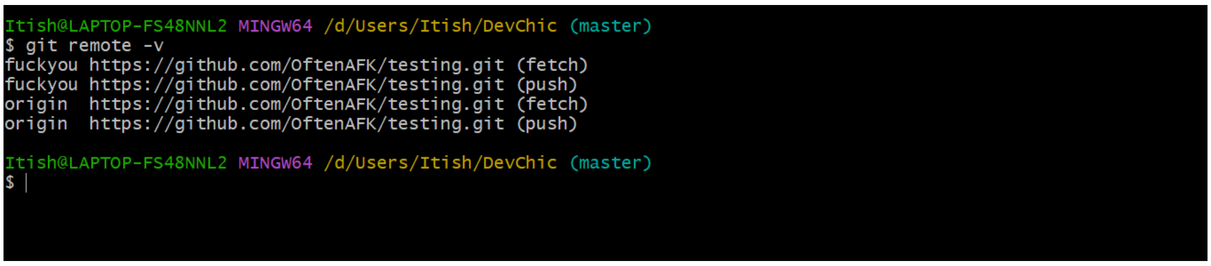
Aim:
To demonstrate push and pull operations in Git.

Theory:
Push transfers committed changes from the local repository to the remote repository, while pull retrieves updates from the remote repository.

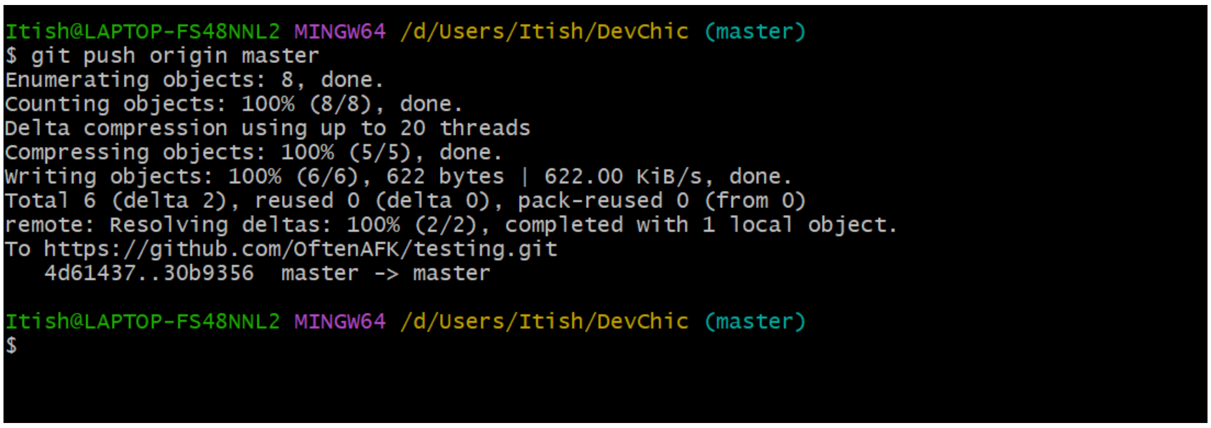
Procedure:
○ Make changes in the local repository and commit them.
○ Push the changes to the remote repository using git push.
○ Make changes directly on the remote repository (e.g., via GitHub interface).
○ Pull the changes to the local repository using git pull.

Tasks:
Provide screenshots of the push and pull operations.
Include the updated commit log.

Screenshots



```
Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git remote -v
fuckyou https://github.com/OftenAFK/testing.git (fetch)
fuckyou https://github.com/OftenAFK/testing.git (push)
origin  https://github.com/OftenAFK/testing.git (fetch)
origin  https://github.com/OftenAFK/testing.git (push)

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ |
```



```
Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$ git push origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 20 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 622 bytes | 622.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/OftenAFK/testing.git
   4d61437..30b9356  master -> master

Itish@LAPTOP-FS48NNL2 MINGW64 /d/Users/Itish/DevChic (master)
$
```