LAB # 18

Getting Familiar with Git & GitHub

Objective: Importance and deep dive related to Git and GitHub

Theory:

What are Git and GitHub?

Git is the open-source software that runs locally on your machine to track changes, while **GitHub** is a cloud-based platform that hosts Git repositories and provides collaboration tools.

Git: The Version Control System

Git is a free, open-source, distributed version control system (DVCS) created by Linus Torvalds. It operates locally on your computer via the command line or graphical user interfaces (GUIs) like Git GUI or GitHub Desktop.

- **Function:** Git's primary function is to track changes in source code and other files, allowing developers to maintain a complete history of a project.
- **Distributed Nature:** In a DVCS, every developer has a full copy of the entire project repository and its history on their local machine, enabling offline work and robust data integrity.
- **Key Features:**
    - **Commits:** Snapshots of your code at specific points in time.
    - **Branching and Merging:** Allows developers to work on separate features or fixes in isolation (branches) and then integrate (merge) those changes back into the main codebase.
    - **Speed:** Uses SHA hashes to compress and store changes efficiently.

You can download the Git software from the official Git website.

GitHub: The Collaboration Platform

GitHub is a web-based hosting service for Git repositories, owned by Microsoft since 2018. It is a "hub" where developers can store their Git projects in the cloud, manage them via a user-friendly web interface, and collaborate with others.

- **Function:** GitHub extends Git's core functionality with a suite of DevOps and project management tools, facilitating teamwork and community engagement.
- **Key Features:**
    - **Repository Hosting:** Provides a central, shared location for team members to push and pull code changes.

LAB # 18

- **Pull Requests:** A mechanism for proposing changes and facilitating code review and discussion before merging code into the main branch.
- **Issue Tracking:** Tools for creating, assigning, and tracking bugs or feature requests.
- **Team Management:** Built-in user authentication, access control, and permissions management.
- **Integrations:** A marketplace for third-party tools, including continuous integration/continuous deployment (CI/CD) pipelines via GitHub Actions.

You can create an account and explore the platform on the GitHub website.

Git vs GitHub

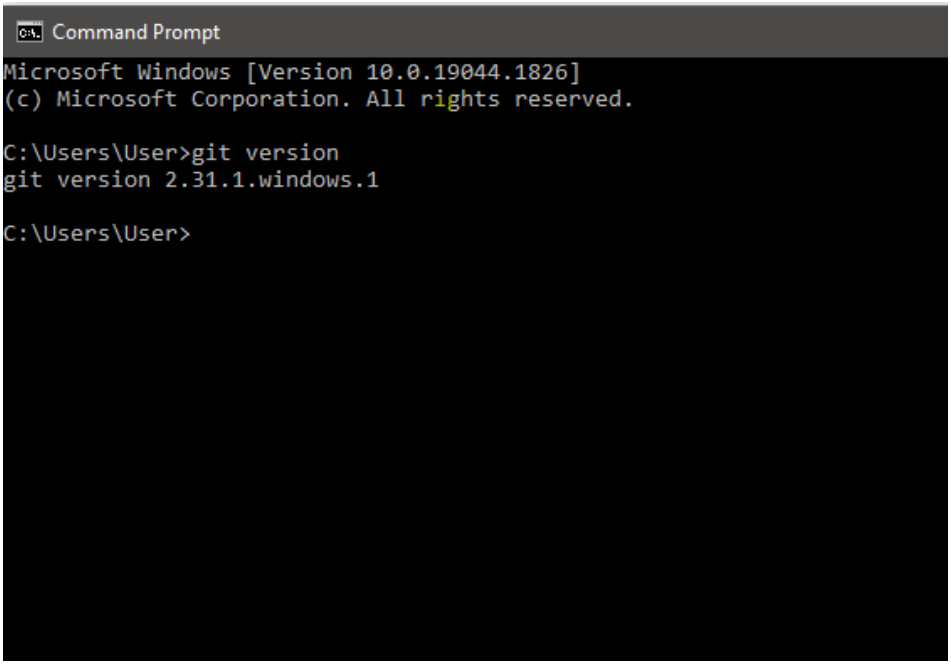
Basis of Difference	Git	GitHub
Nature	Software (tool)	Service/Platform (web-based)
Location	Installed and hosted locally	Hosted in the cloud
Interface	Primarily command-line (CLI), with GUI options	Graphical user interface (GUI) via web browser
Ownership	Maintained by the Linux Foundation	Owned by Microsoft
Primary Use	Tracking code changes and version history locally	Hosting repositories and team collaboration online

Installing Git

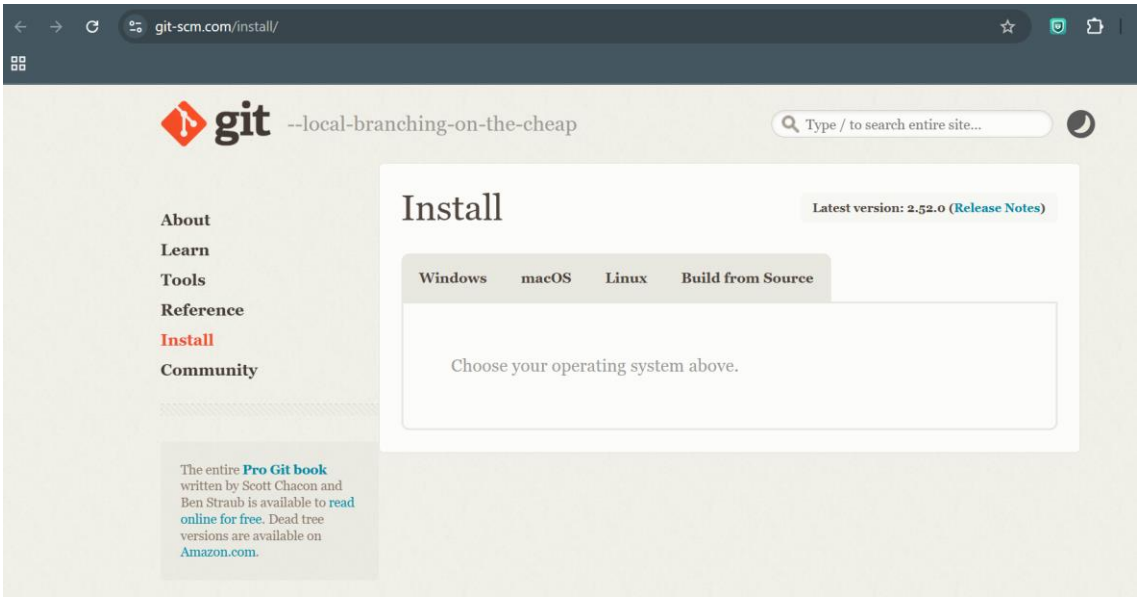
Step 1: Download Git

- **Go to https://git-scm.com/downloads**
- **Choose your OS (Windows / macOS / Linux) and download.**

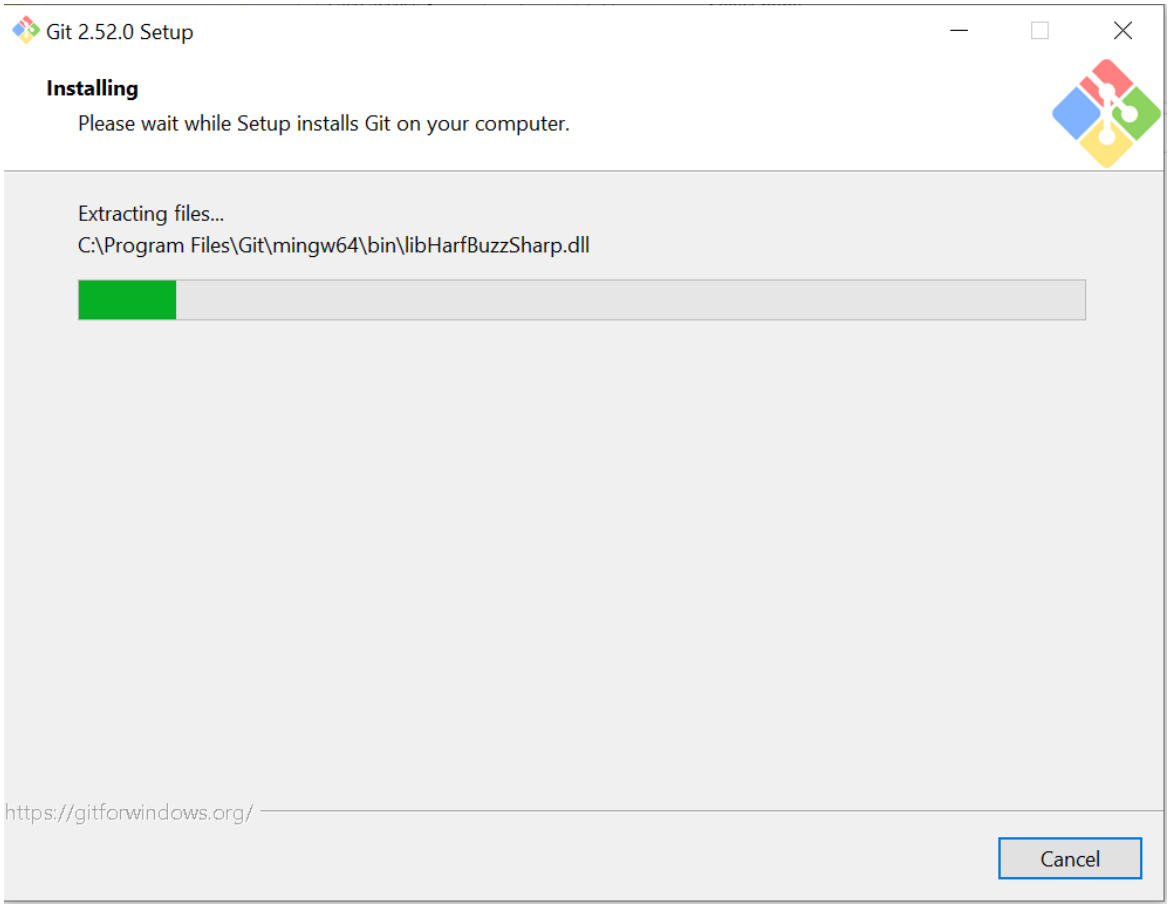
Note:
Git comes preinstalled in some Macs and Linux-based systems, but you can always check if you have Git installed in your machine by typing git version in your terminal. You can use Command Prompt to do this.



LAB # 18



Step 02 Follow the necessary installer guide until installation is complete.



LAB # 18

Step 3: Verify Installation - Open the command prompt and type git version to verify that Git was successfully installed.

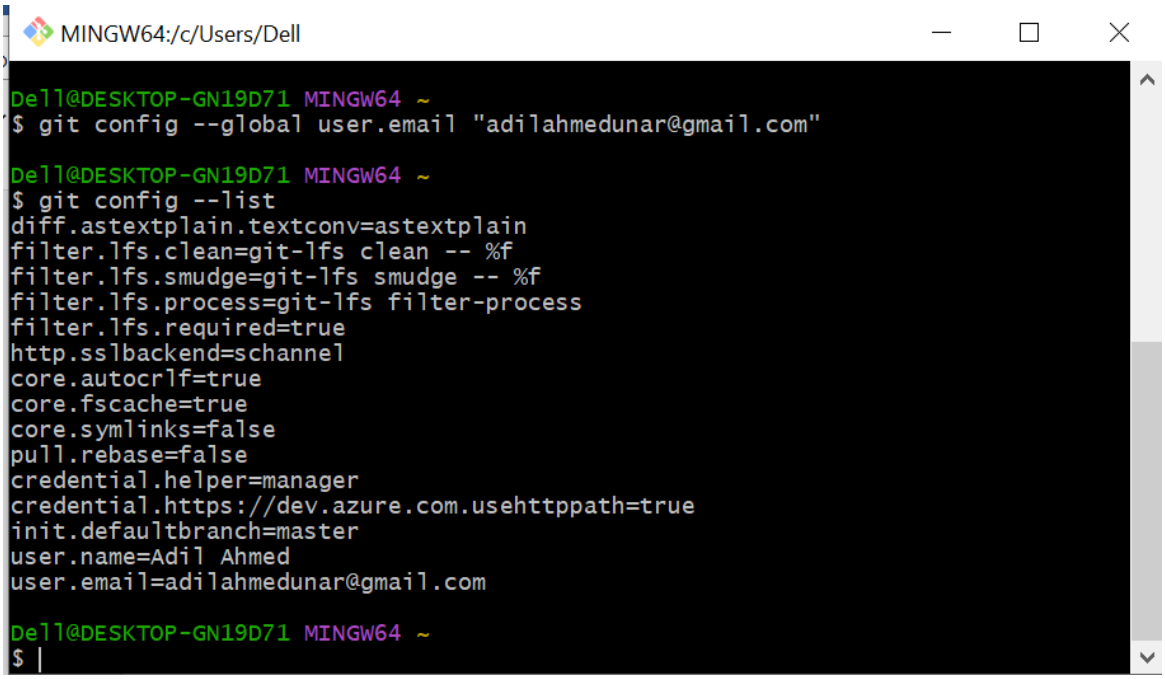
Command: git –version

Step 4: Configuring Git and Verify

Set your identity so Git knows who is making changes:

Commands:

git config --global user.name "Your Name"
git config --global user.email "youremail@example.com**"** then
git config –list



LAB # 18

Step 5: Creating a Local Repository

a. Create a project folder:
   - **mkdir** "Project Folder Name"
   - **cd** "Project Folder Name"

```
Dell@DESKTOP-GN19D71 MINGW64 ~
$ mkdir Lab18Project

Dell@DESKTOP-GN19D71 MINGW64 ~
$ cd Lab18Project

Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project
$
```

b. Initialize Git
   - **git init**

```
Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project
$ git init
Initialized empty Git repository in C:/Users/Dell/Lab18Project/.git/

Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$
```

Note: Follow the location and verify the folder.

c. Check Status
   - **git status**

```
Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

d. Create and commit a file
   - **echo "Hello Git!" > README.md**
   - **git add README.md**
   - **git commit -m "First commit"**
   - **git log**

```
Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$ echo "Hello Git!" > README.md

Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$ git add README.md
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$ git commit -m "First Commit"
[master (root-commit) f055a52] First Commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$ git log
commit f055a52c6a925035b74a65de2908c3ad679199f0 (HEAD -> master)
Author: Adil Ahmed <adilahmedunar@gmail.com>
Date:   Sun Jan 4 13:26:19 2026 +0500

    First Commit

Dell@DESKTOP-GN19D71 MINGW64 ~/Lab18Project (master)
$
```

LAB # 18

Task # 01

Install Git on your system and verify the installation. Configure your Git username and email. Create a local repository, add a file, and make your first commit. Paste screenshots of Git version, configuration, and commit history in your lab file.