

LP- II : Web Application Development

Assignment 2A: Create version control account on GitHub and use Git commands to create repository and push your code to GitHub.

❖ **What is Git:**

Git is a version control system designed to help developers track changes in the projects, manage code versions, and collaborate effectively.

❖ **Key Features of git:**

1. **Version Control System:** Git keeps a detailed history of changes in code, enabling developers to revert to earlier versions, compare updates, and track contributions.
 2. **Local Repository Management:** It allows developers to store the entire project history locally, making it possible to work offline and sync changes with others later.
 3. **Branching and Merging:** Git supports the creation of branches for testing or adding new features. These branches can be merged back into the main project without disturbing the original code.
 4. **Distributed System:** Git is decentralized, meaning every developer has a complete copy of the repository, ensuring resilience and flexibility in case of system failures.
-

❖ **What is GitHub:**

- GitHub is a cloud-based platform for hosting Git repositories, designed to help developers collaborate on projects, share code, and manage version control efficiently.

❖ **Key Features of GitHub:**

1. **Repository Hosting:** GitHub provides a central location to store Git repositories online, making it easy to access, manage, and back up code from anywhere.

2. **Version Control Integration:** GitHub seamlessly integrates with Git, allowing developers to push, pull, and synchronize changes, making code management smoother and more reliable.
3. **Community and Open Source:** GitHub serves as a hub for open-source contributions, allowing developers to share projects publicly, contribute to others' work, and build a professional portfolio.
4. **Collaboration Tools:** GitHub offers features like pull requests, code reviews, and issue tracking, making teamwork more efficient and organized.
5. **GitHub Codespaces:** GitHub offers Codespaces, a cloud-based development environment that lets developers write, test, and debug code directly from their browser.

❖ **Key Difference Between Git and GitHub:**



Git

Software

Version control

Maintained by Linux

Open-Source

No user management

Locally installed

Minimal external tool
configuration

Little to no competition



GitHub

Service

Git repository hosting

Maintained by Microsoft

Free or paid membership

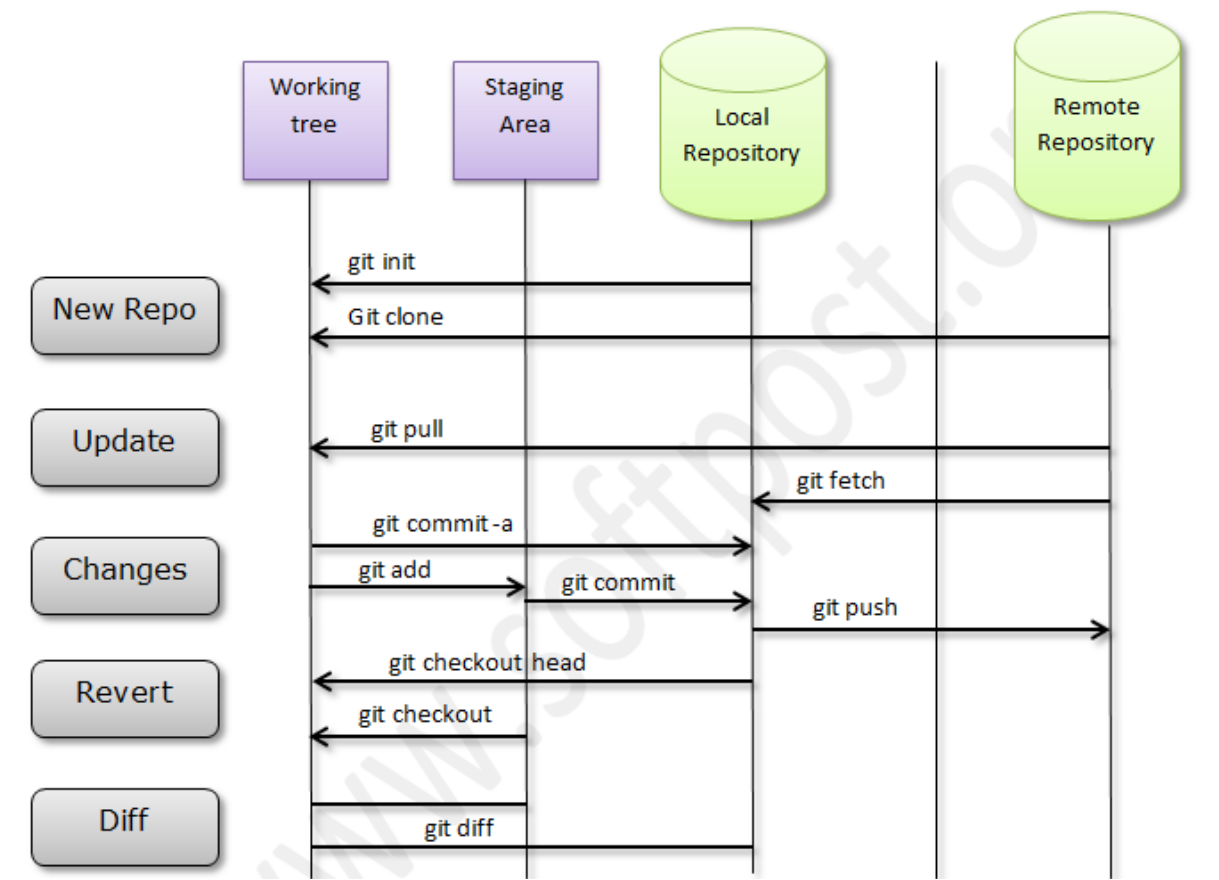
Built-in user management

Hosted on the web

Active marketplace for
tool integration

High competition

❖ Git Architecture:



❖ Components of Git Architecture:

1. Working Tree:

- The working directory is the place where you work on your project files.
- It contains the actual files and directories of the project.
- Any changes made here need to be staged and committed to be tracked by Git.

2. Staging Area:

- The staging area is an intermediate space where changes are prepared before committing.
- Files are added to the staging area using the git add command.
- It acts as a preview of what will be included in the next commit.

3. Local Repository:

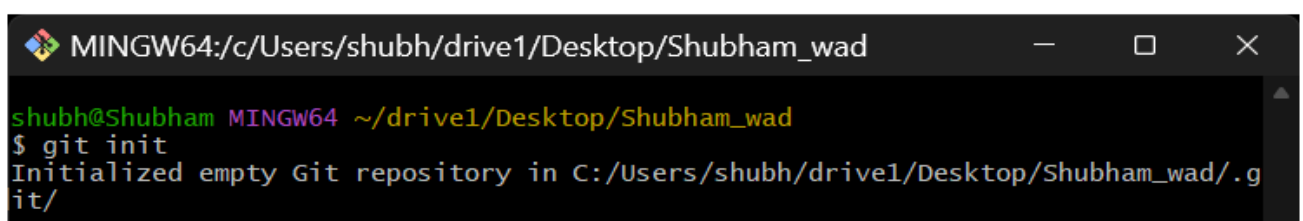
- The local repository is the .git folder located inside your project directory.
- It contains the entire history of the project, including commits, branches, and configurations.
- Changes are saved to the local repository using the git commit command.

4. Remote Repository:

- The remote repository is a version of your repository hosted on a remote server (e.g., GitHub, GitLab, Bitbucket).
- It allows multiple developers to collaborate by sharing and synchronizing code.

❖ Git Workflow Step by Step:

- **Create a Folder:** Create a new folder for your project and navigate to it.
- **Add a File:** Add any new file (e.g., index.html, README.md) to the folder
- **Open Git Bash:** Right-click inside the folder and open it with Git Bash.
- **Initialize Git:** Run git init to initialize the folder as a Git repository.



```
MINGW64:/c/Users/shubh/drive1/Desktop/Shubham_wad
shubh@Shubham MINGW64 ~/drive1/Desktop/Shubham_wad
$ git init
Initialized empty Git repository in C:/Users/shubh/drive1/Desktop/Shubham_wad/.git/
```

- **Stage Changes:** Use "git add ." to add all files in the folder to the staging area
- **Commit Changes:** Run git commit -m "Any message" to save the staged changes to the local repository. Use git status to check status of file.

```
shubh@Shubham MINGW64 ~/drive1/Desktop/Shubham_wad (master)
$ git add .

shubh@Shubham MINGW64 ~/drive1/Desktop/Shubham_wad (master)
$ git commit -m "First Commit"
[master (root-commit) 8495e87] First Commit
1 file changed, 1 insertion(+)
create mode 100644 Shubham.txt
```

- **Create a GitHub Repository:** On GitHub, create a new repository but do not add a README. Copy the repository URL(HTTPS with .git at the end).
- **Generate Personal Access Token:** Go to GitHub Settings > Developer Settings > Personal Access Tokens and generate a token with the required scopes (check all in this case). Copy it and store in a secure place.
- **Add Remote Repository:** Adding a remote repository means linking your local Git repository to a repository hosted on a remote platform i.e. GitHub. Follow the structure of the URL as: `https://@github.com/username/repository-name.git`

```
shubh@Shubham MINGW64 ~/drive1/Desktop/Shubham_wad (master)
$ git remote add origin https://ghp_0XJFNjShmcyJKRxfoBELxrDePtU7tx4F6Cx5@github.com/Mr-Shubham-Palde/Git_test

shubh@Shubham MINGW64 ~/drive1/Desktop/Shubham_wad (master)
$ git push -u origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/Mr-Shubham-Palde/Git_test'

shubh@Shubham MINGW64 ~/drive1/Desktop/Shubham_wad (master)
$ git push https://ghp_0XJFNjShmcyJKRxfoBELxrDePtU7tx4F6Cx5@github.com/Mr-Shubham-Palde/Git_test
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 249 bytes | 249.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/Mr-Shubham-Palde/Git_test/pull/new/master
remote:
To https://github.com/Mr-Shubham-Palde/Git_test
 * [new branch]      master -> master
```

- **Perform Initial Push using PAT:** Above screenshot also shows the initial push from the local repository to the remote repository, done using PAT. After successful execution of the command, file should be visible on the GitHub repository as well, along with its respective commit message.
- **Upstream option:** Use `git push --set-upstream` . What it does:
 - Links the local master branch to the remote master branch in the origin remote repository.

- After this, you can simply run `git push` or `git pull` without specifying the branch name.
 - In case of any error of authentication, perform following commands:
 1. `git config --global user.name "GitHubUsername"`
 2. `git config --global user.email "GitHubEmail"` These commands should resolve any issues faced while authentication. Ideal to perform them before using the Personal Access Token to set the remote repository.
-

◆ Git Clone & Git Pull:

1. Git Clone: `git clone` is used to create a copy of a remote repository on your local machine. It also sets up the remote origin automatically, allowing you to easily push and pull changes from the remote repository.

❖ Syntax: `git clone <repository - url>`

2. Git Pull: `git pull` fetches updates from the specified remote repository (e.g., origin) and the specified branch. It then merges the changes into your current local branch, keeping your local project up to date.

❖ Syntax: `git pull <remote> <branch>`

Example of Git Clone & Pull:

Step 1: Create a Repository and Add a File on GitHub:

1. Navigate to GitHub and create a new repository.
2. Add a file (e.g., `example.txt`) to the repository and save changes.

Step 2: Clone the Repository:

1. Create any new folder and open Git Bash from that location.
2. Run the `git clone` command.
3. This will create a local copy of the repository in a new folder named after the repository.

```
MINGW64:/c/Users/shubh/drive1/Desktop/copy
shubh@Shubham MINGW64 ~/drive1/Desktop/copy
$ git clone https://github.com/Mr-Shubham-Palde/Git_test.git
Cloning into 'Git_test'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (8/8), 46.72 KiB | 388.00 KiB/s, done.
shubh@Shubham MINGW64 ~/drive1/Desktop/copy
$
```

Step 3: Edit the file on GitHub

1. Open the repository on GitHub and edit the file you added (e.g., example.txt).
2. Save the changes directly on GitHub.

Step 4: Navigate to the Cloned Repository on Your Local Machine

1. Use the cd command to navigate to the folder where the repository was cloned:

```
shubh@Shubham MINGW64 ~/drive1/Desktop/copy
$ cd Git_test

shubh@Shubham MINGW64 ~/drive1/Desktop/copy/Git_test (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 998 bytes | 142.00 KiB/s, done.
From https://github.com/Mr-Shubham-Palde/Git_test
  8800353..5f3fd4e  main       -> origin/main
Updating 8800353..5f3fd4e
Fast-forward
  Shubham.txt | 3 ++-
  1 file changed, 2 insertions(+), 1 deletion(-)
```

Step 5: Pull the Changes from GitHub

1. Run the git pull command to retrieve the latest changes made on GitHub (as shown above).
2. This will fetch and merge the updated file (e.g., Fun_wad.txt) into your local repository.