FSD Laboratory 01

Arin Khopkar
TY B.Tech CSE
Panel I, I2
62
1032221073

Aim: Version control with Git.
Objectives:
1. To introduce the concepts and software behind version control, using the example of Git.
2. To understand the use of 'version control' in the context of a coding project.
3. To learn Git version control with Clone, commit to, and push, pull from a git repository.

Theory:
1. What is Git? What is Version Control?

Git is a distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Created by Linus Torvalds in 2005, Git allows multiple developers to work on a project simultaneously without overwriting each other's changes. Key features of Git include:

- **Distributed Nature**: Each developer has a full copy of the project's history, enabling offline work and better collaboration.
- **Branching and Merging**: Git makes branching and merging easy, allowing developers to work on different features or fixes concurrently.
- **Performance**: Git is optimized for speed and efficiency in handling project history and changes.

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later. It is essential for collaborative software development, providing benefits such as:

- **History Tracking**: Keeps a history of changes, enabling developers to revert to previous versions if needed.
- **Collaboration**: Facilitates collaboration among multiple developers by managing changes and preventing conflicts.
- **Backup**: Acts as a backup system by keeping a history of the project.
- **Branching and Merging**: Supports parallel development by allowing developers to create branches and merge changes seamlessly.

2. How to use Git for version controlling?

Basic Git Workflow:
Configure Git:

```
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
```

Initialize a Repository:
```
git init
```
This command initializes a new Git repository in your current directory.

Clone a Repository:
```
git clone <repository_url>
```
This command creates a copy of an existing repository.

Check Repository Status:
```
git status
```
This command shows the status of changes as untracked, modified, or staged.

Stage Changes:
```
git add <file>
git add .
```
The first command stages a specific file, while the second stages all changes in the directory.

Commit Changes:
```
git commit -m "Commit message"
```
This command records the staged changes with a message.

Push Changes:
```
git push origin <branch_name>
```
This command uploads your local changes to the remote repository.

Pull Changes:
```
git pull origin <branch_name>
```
This command fetches and merges changes from the remote repository to your local repository.

Create a Branch:
```
git checkout -b <new_branch_name>
```
This command creates and switches to a new branch.

Merge a Branch:
```
git checkout <target_branch>
git merge <source_branch>
```
These commands merge changes from the source branch into the target branch.

Resolve Conflicts:

If there are merge conflicts, Git will prompt you to resolve them manually. After resolving, you can stage the resolved files and commit the changes.

Additional Useful Commands:

Log History:

```
git log
```

- Shows the commit history.

Delete a Branch:

```
git branch -d <branch_name>
```

- Deletes a local branch.

Revert a Commit:

```
git revert <commit_id>
```

- Creates a new commit that undoes the changes of a specified commit.

Reset to a Previous Commit:

```
git reset --hard <commit_id>
```

- Resets the repository to a specified commit, discarding all changes after that commit.

FAQ:
1. What is branching in Git?

Branching in Git allows you to diverge from the main line of development and continue to work separately without affecting the main project. Each branch represents an independent line of development. You can create, modify, and merge branches, enabling you to work on different features, bug fixes, or experiments in isolation. This is especially useful for collaborative projects where multiple developers are working on different parts of the project simultaneously.

Key Concepts of Branching:

- **Branch**: A branch is a movable pointer to a commit. The default branch in many Git repositories is called `master` or `main`.
- **HEAD**: HEAD is a pointer that always points to the current branch you are working on.
- **Feature Branches**: Temporary branches created to work on new features or bug fixes. They can be merged back into the main branch once the work is complete.

2. How to create and merge branches in Git? Write the commands used.

Creating a Branch:

To create a new branch, you use the `git branch` command followed by the name of the new branch.

```
git branch <branch_name>
```

Alternatively, you can create and switch to a new branch immediately using the `git checkout -b` command:

```
git checkout -b <branch_name>
```

To switch to a different branch, use the `git checkout` command:

```
git checkout <branch_name>
```

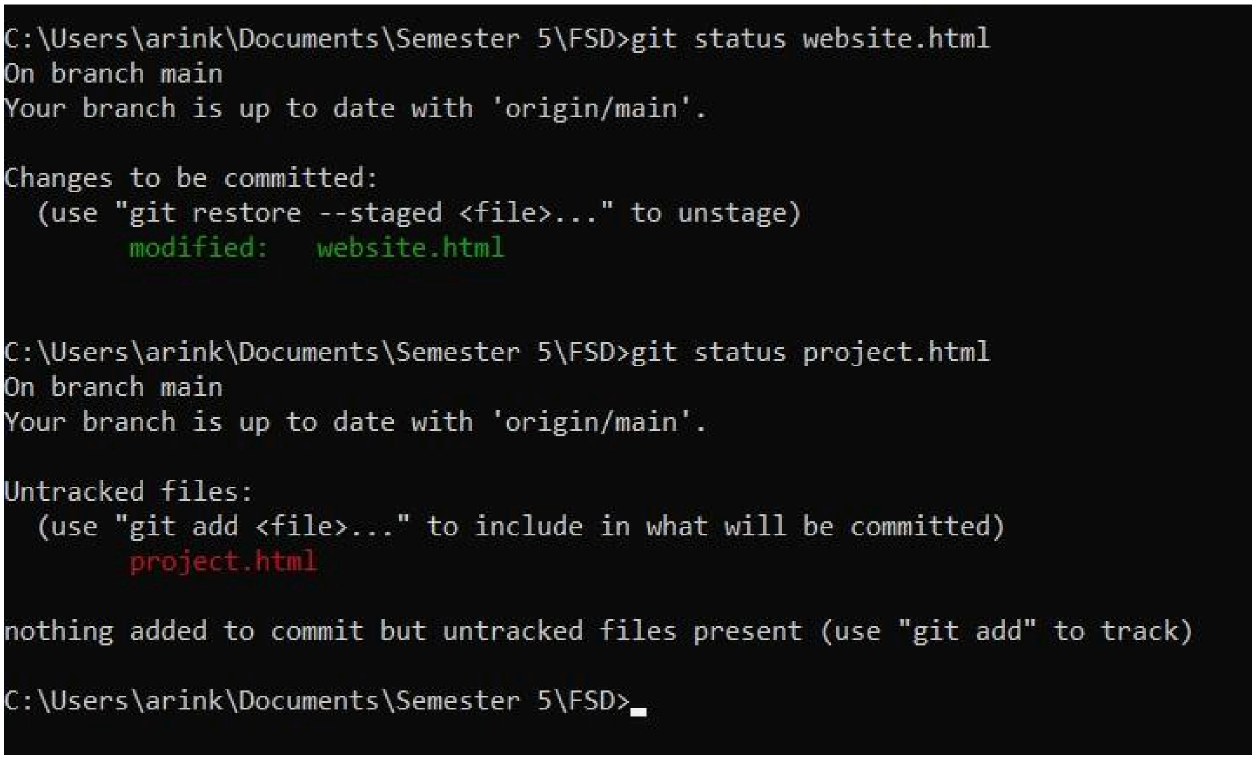
Starting from Git 2.23, you can use the `git switch` command to switch branches, which is more intuitive:

```
git switch <branch_name>
```

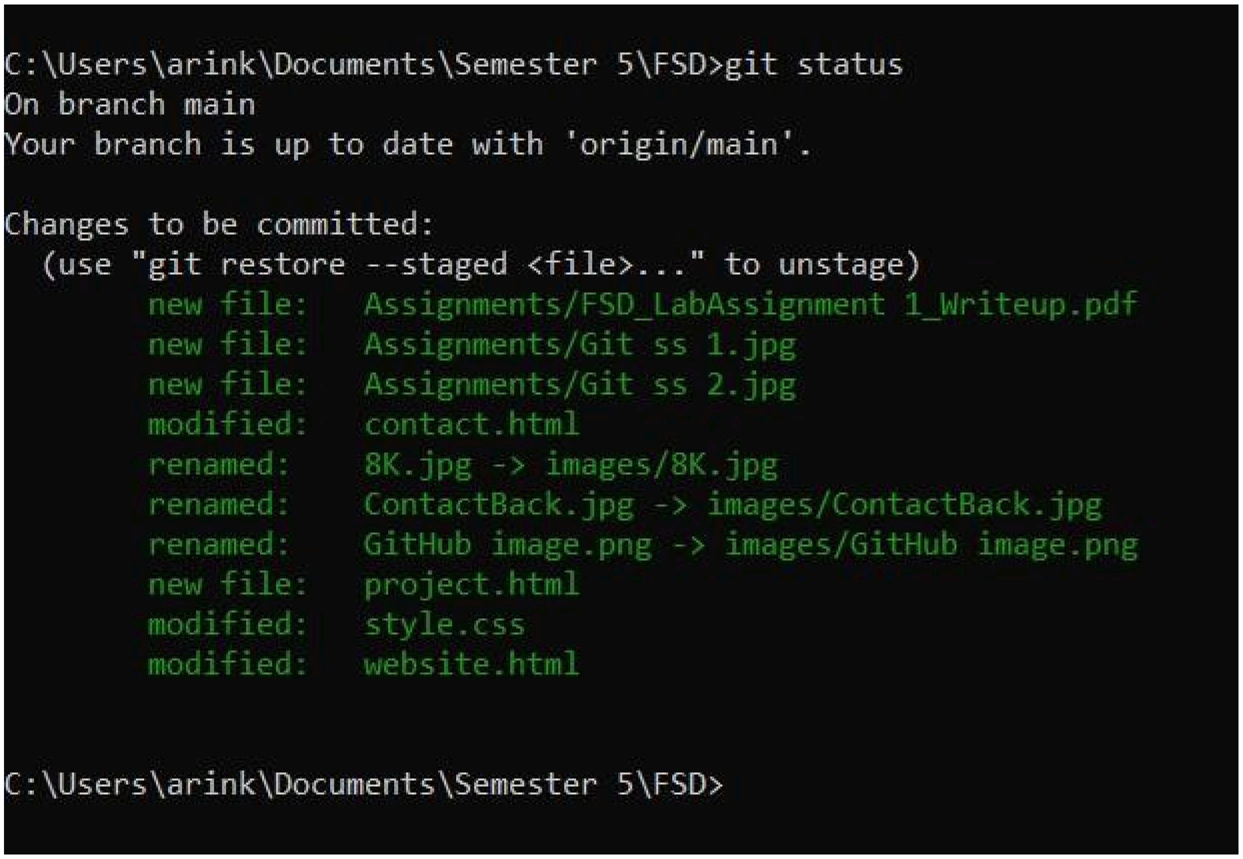
To merge changes from one branch into another, you first switch to the branch you want to merge into, then use the `git merge` command.

```
git checkout <target_branch>

git merge <source_branch>
```

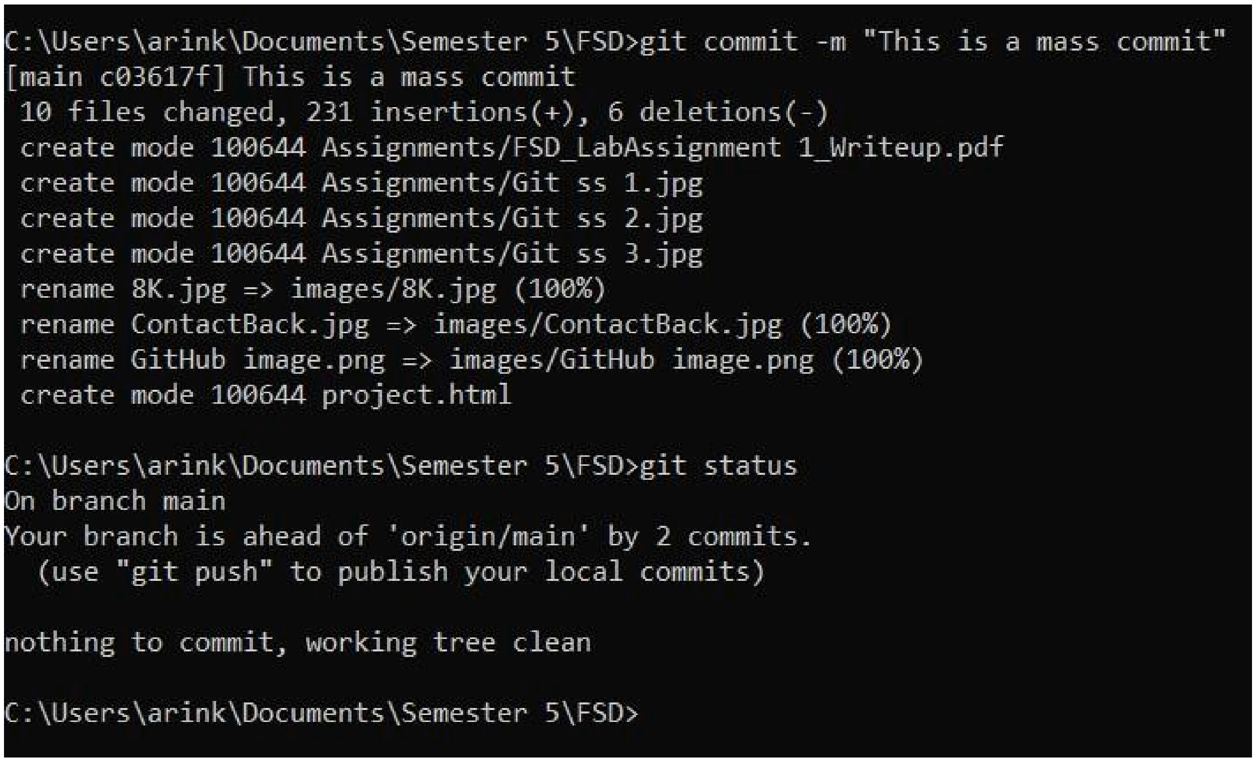
Output: Screenshots of the output to be attached.



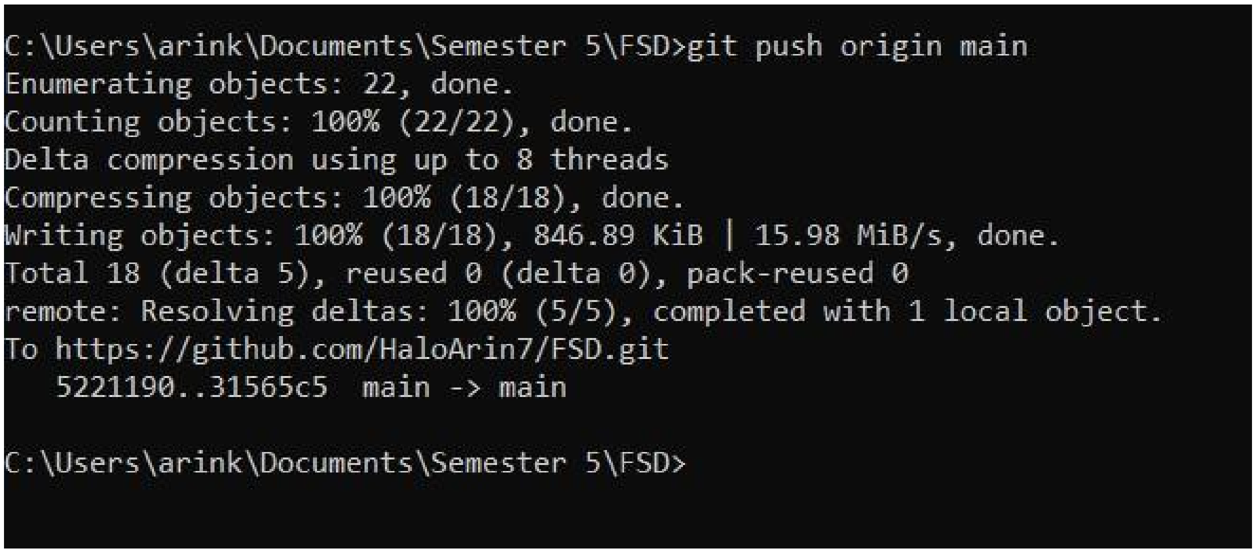
```
C:\Users\arink\Documents\Semester 5\FSD>git status website.html
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   website.html


C:\Users\arink\Documents\Semester 5\FSD>git status project.html
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)
        project.html

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

C:\Users\arink\Documents\Semester 5\FSD>
```



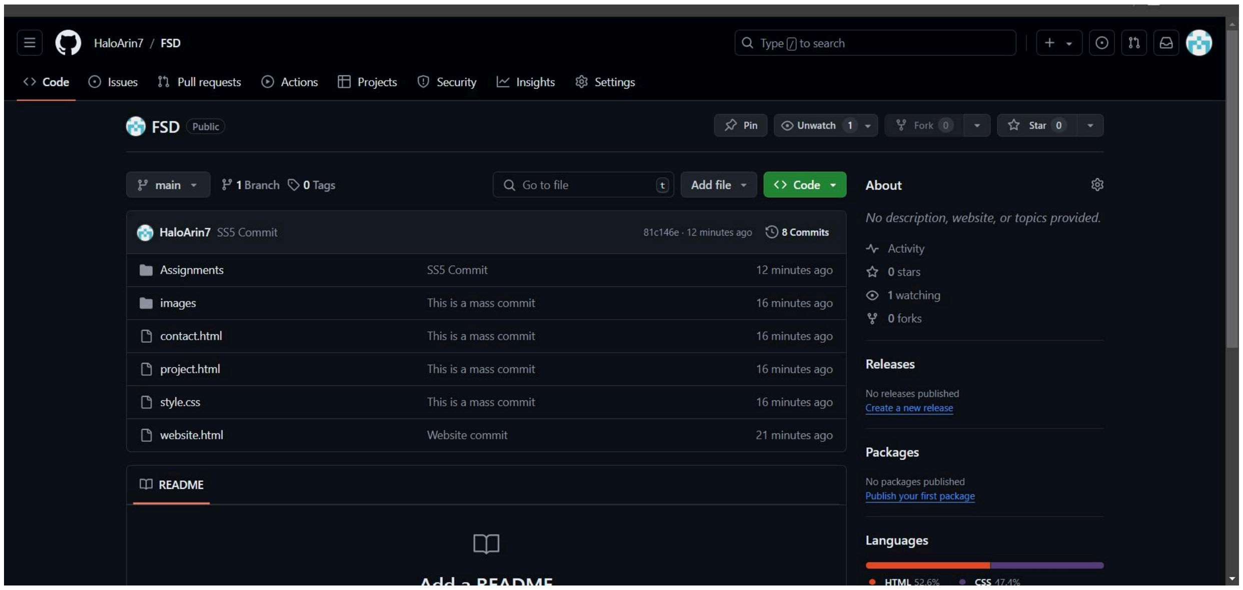
```
C:\Users\arink\Documents\Semester 5\FSD>git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Assignments/FSD_LabAssignment 1_Writeup.pdf
        new file:   Assignments/Git ss 1.jpg
        new file:   Assignments/Git ss 2.jpg
        modified:   contact.html
        renamed:    8K.jpg -> images/8K.jpg
        renamed:    ContactBack.jpg -> images/ContactBack.jpg
        renamed:    GitHub image.png -> images/GitHub image.png
        new file:   project.html
        modified:   style.css
        modified:   website.html


C:\Users\arink\Documents\Semester 5\FSD>
```



```
C:\Users\arink\Documents\Semester 5\FSD>git commit -m "This is a mass commit"
[main c03617f] This is a mass commit
 10 files changed, 231 insertions(+), 6 deletions(-)
 create mode 100644 Assignments/FSD_LabAssignment 1_Writeup.pdf
 create mode 100644 Assignments/Git ss 1.jpg
 create mode 100644 Assignments/Git ss 2.jpg
 create mode 100644 Assignments/Git ss 3.jpg
 rename 8K.jpg => images/8K.jpg (100%)
 rename ContactBack.jpg => images/ContactBack.jpg (100%)
 rename GitHub image.png => images/GitHub image.png (100%)
 create mode 100644 project.html

C:\Users\arink\Documents\Semester 5\FSD>git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

C:\Users\arink\Documents\Semester 5\FSD>
```



```
C:\Users\arink\Documents\Semester 5\FSD>git push origin main
Enumerating objects: 22, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (18/18), 846.89 KiB | 15.98 MiB/s, done.
Total 18 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 1 local object.
To https://github.com/HaloArin7/FSD.git
   5221190..31565c5  main -> main

C:\Users\arink\Documents\Semester 5\FSD>
```



HaloArin7 SS5 Commit 81c146e · 12 minutes ago 8 Commits

Assignments	SS5 Commit	12 minutes ago
images	This is a mass commit	16 minutes ago
contact.html	This is a mass commit	16 minutes ago
project.html	This is a mass commit	16 minutes ago
style.css	This is a mass commit	16 minutes ago
website.html	Website commit	21 minutes ago

README

Add a README

About

No description, website, or topics provided.

Activity
0 stars
1 watching
0 forks

Releases

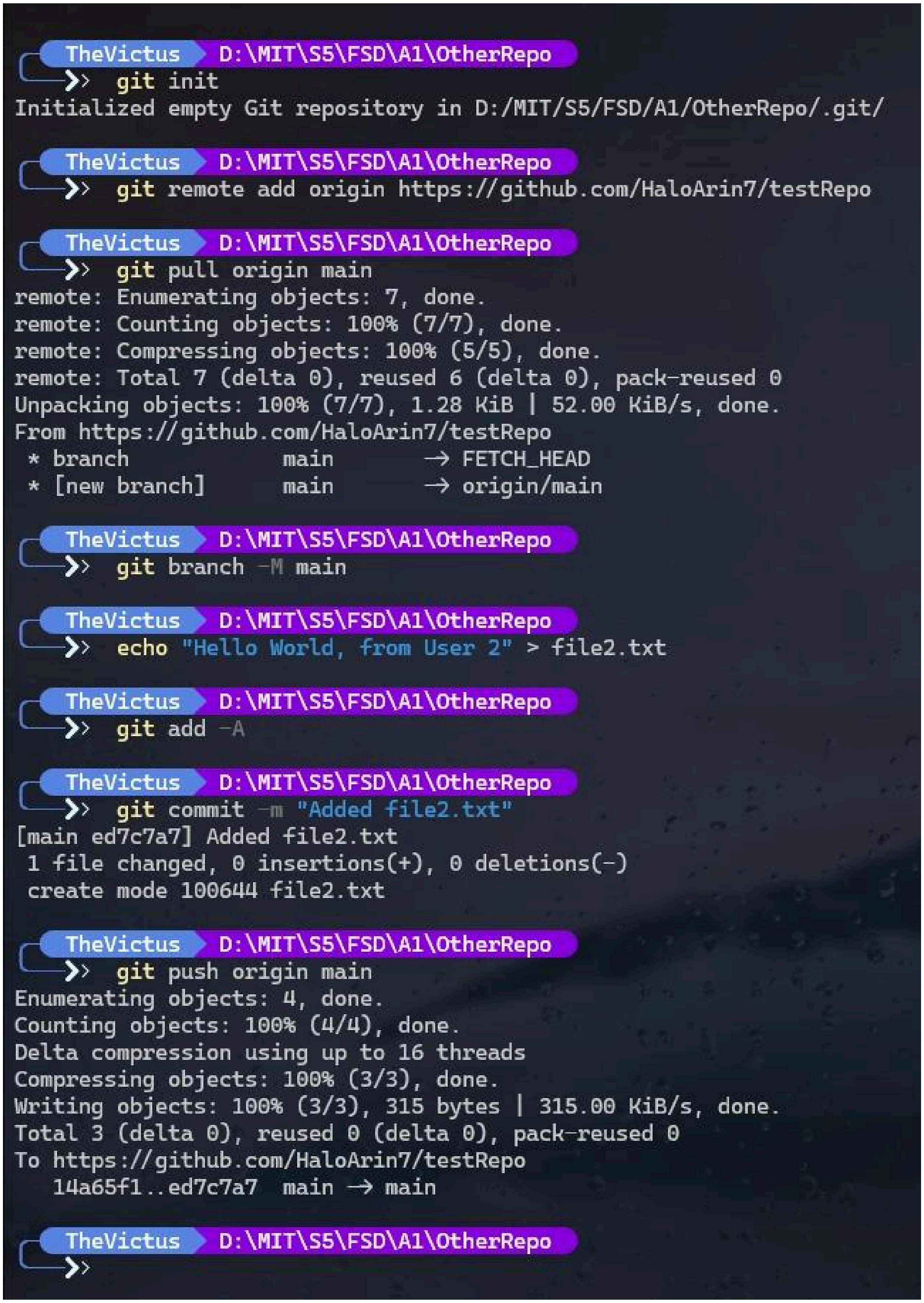
No releases published
Create a new release

Packages

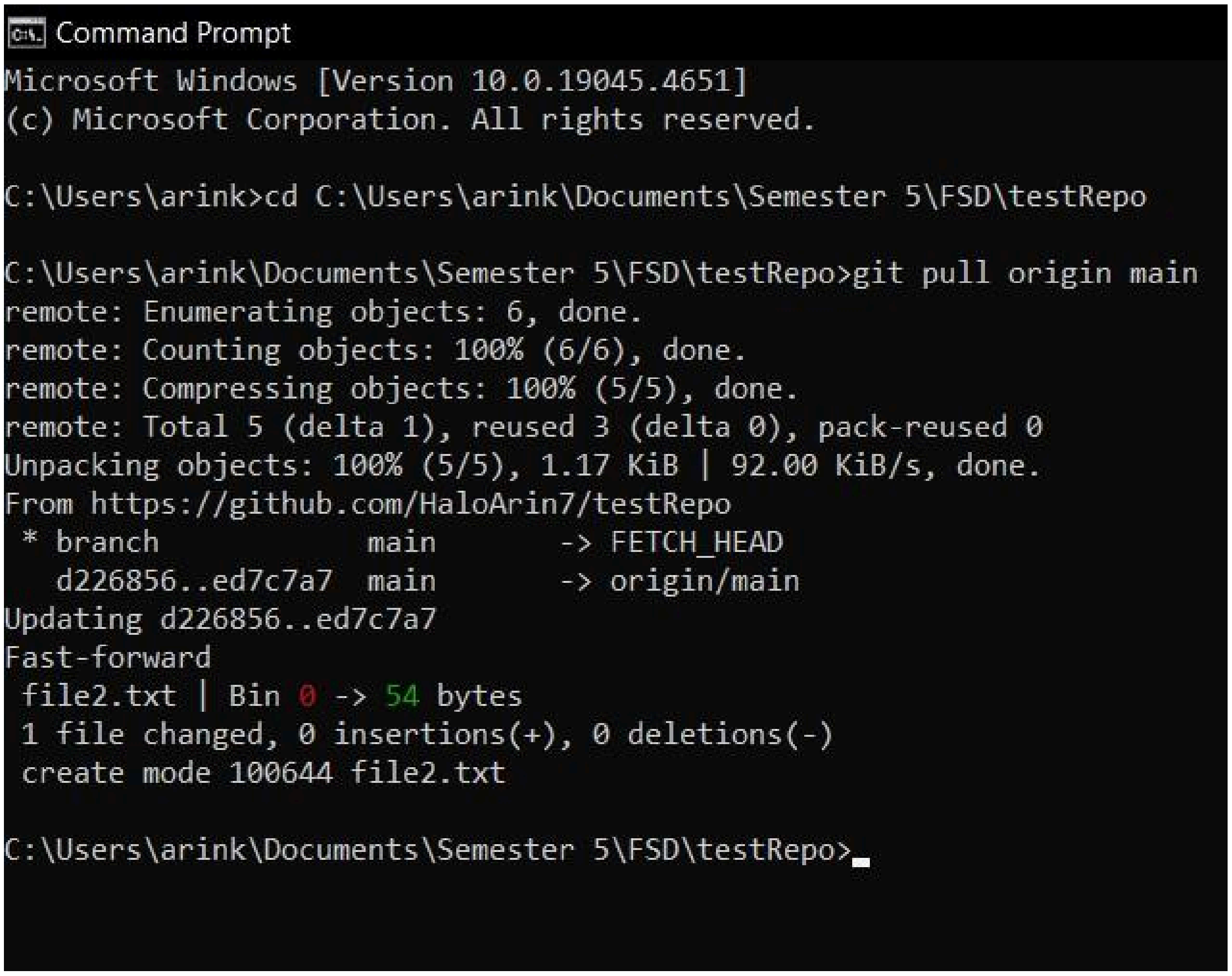
No packages published
Publish your first package

Languages

HTML 52.6% CSS 47.4%



```
 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git init
Initialized empty Git repository in D:/MIT/S5/FSD/A1/OtherRepo/.git/

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git remote add origin https://github.com/HaloArin7/testRepo

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git pull origin main
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 7 (delta 0), reused 6 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), 1.28 KiB | 52.00 KiB/s, done.
From https://github.com/HaloArin7/testRepo
 * branch            main       -> FETCH_HEAD
 * [new branch]      main       -> origin/main

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git branch -M main

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> echo "Hello World, from User 2" > file2.txt

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git add -A

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git commit -m "Added file2.txt"
[main ed7c7a7] Added file2.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file2.txt

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >> git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 315 bytes | 315.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/HaloArin7/testRepo
   14a65f1..ed7c7a7  main -> main

 TheVictus  D:\MIT\S5\FSD\A1\OtherRepo
 >>
```



```
Microsoft Windows [Version 10.0.19045.4651]
(c) Microsoft Corporation. All rights reserved.

C:\Users\arink>cd C:\Users\arink\Documents\Semester 5\FSD\testRepo

C:\Users\arink\Documents\Semester 5\FSD\testRepo>git pull origin main
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 1), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 1.17 KiB | 92.00 KiB/s, done.
From https://github.com/HaloArin7/testRepo
 * branch            main       -> FETCH_HEAD
   d226856..ed7c7a7  main       -> origin/main
Updating d226856..ed7c7a7
Fast-forward
 file2.txt | Bin 0 -> 54 bytes
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file2.txt

C:\Users\arink\Documents\Semester 5\FSD\testRepo>
```

Problem Statement:

Create a public git repository for your team and submit the repo URL as a solution to this assignment, Learn Git concept of Local and Remote Repository, Push, Pull, Merge and Branch.