Version Control Day 1 Assingment

Mohamed Emary

March 4, 2025

1 Git Lab 1

1.1 Basic Initialization

Scenario: You are starting a new project called `CalculatorApp`. Initialize a new Git repository in the project directory and set up a `.gitignore` file to ignore any temporary files ending with `.tmp`.

Questions:

- What command will you use to initialize a Git repository?
- How will you create a `.gitignore` file to exclude `.tmp` files?

```
mkdir CalculatorApp
cd CalculatorApp
git init
touch .gitignore
echo "*.tmp" > .gitignore
```

1.2 Tracking Changes

Scenario: You created two files: `index.html` and `style.css`. Make Git track these files and commit them with the message `Initial commit with HTML and CSS files.`

Questions:

- What commands will you use to add the files to the staging area?
- How will you commit the changes?
- How will you retrieve the stashed changes later?

```
git add index.html style.css

git commit -m "Initial commit with HTML and CSS files"

git stash pop
```

1.3 Branching

Scenario: You need to add a new feature to your project but don't want to disrupt the main branch. Create a branch named `feature-login`.

Questions:

- How will you create a new branch?
- How will you switch to the newly created branch?

```
git branch feature-login
git checkout feature-login
```

1.4 Merging

Scenario: After completing the `feature-login` branch, merge it back into the main branch.

Questions:

- What command will you use to merge the `feature-login` branch into the main branch?
- What will you do if there are merge conflicts?

```
git checkout main

git merge feature-login

# If merge conflicts occur:
git status                    # Check conflicted files
git diff                     # View conflicts

# Manually resolve conflicts in editor
git add <resolved-files>     # Stage resolved files
git commit -m "Merge feature-login into main"
```

1.5 Undoing Changes

Scenario: You accidentally added a file called debug.log to the staging area, but you don't want to include it in the commit.

Questions:

- How will you remove the file from the staging area without deleting it from your working directory?

```
git restore --staged debug.log
```

1.6 Viewing History

Scenario: You want to see the commit history of the project to review past changes.

Questions:

- What command will you use to view the commit history?
- How can you display detailed information about a specific commit?

```
git log

git show <commit-hash>
```

1.7 Reverting a Commit

Scenario: A recent commit introduced a bug, and you want to revert it without removing the commit from history.

Questions:

- How will you identify the commit to revert?
- What command will you use to create a new commit that undoes the changes?

```
git log --oneline

git revert <commit-hash>
```

1.8 Stashing

Scenario: You made some changes to script.js but need to switch branches without committing your changes. Use Git's stashing feature to save your work temporarily.

Questions:

- What command will you use to stash your changes?

```
# Basic stash
git stash

# Stash with message
git stash save "WIP: script.js changes"
```