Problem Statement 4:

Tagging and Versioning Create tags to mark specific versions or releases of the project. Explore the difference between lightweight tags and annotated tags. Rollback to a previous tagged version to simulate version control in action.

Solution:

Tagging:

Tagging refers to creating a named reference (tag) for a specific commit. Tags are used to mark specific points in Git history, such as software releases or important milestones. They provide a way to easily refer to and retrieve a particular version of the code.

Versioning:

"Tagging" is a way to capture a point in the version history as a reference point, often marking a specific release or a significant commit. It's a way to assign a meaningful name or label to a specific commit, making it easier to refer to that particular state in the future. Tags in Git are like bookmarks or labels that point to specific points in Git history.

Create tags to mark specific versions

```
C:\Users\DELL\simple website>git add .

C:\Users\DELL\simple website>git commit -m "second"
[master 84d3434] second
 1 file changed, 41 insertions(+)
 create mode 100644 src/second.html

C:\Users\DELL\simple website>git tag v0.1

C:\Users\DELL\simple website>git push origin v0.1
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 743 bytes | 92.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kavinvelavan/simple-website.git
 * [new tag]         v0.1 -> v0.1
```

```
C:\Users\DELL\simple website>git tag v0.2

C:\Users\DELL\simple website>git push origin v0.2
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 698 bytes | 58.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/kavinvelavan/simple-website.git
 * [new tag]         v0.2 -> v0.2
```

Lightweight Tags:

- Lightweight tags are essentially just a reference to a specific commit. They are created using the git tag command without the -a, -s, or -m options.
- They are simple pointers to a specific commit and only contain the commit checksum (SHA-1 hash).
- Lightweight tags don't store any additional information, such as the tagger's name, email, or a message.
- Example of creating a lightweight tag:

    git tag v1.0

Annotated Tags:

- Annotated tags, on the other hand, are more feature-rich. They store additional information like the tagger's name, email, date, and a tagging message.
- They are created using the -a, -s, or -m options with the git tag command.
- Annotated tags are useful for creating more detailed release notes and documentation about a specific version.
- Example of creating an annotated tag:

    git tag -a v1.0 -m "Release version 1.0"

- Annotated tags are generally preferred for creating official release points in a project because they provide more context and information.

Rollback to a previous tagged version to simulate version control in action

```
C:\Users\DELL\simple website>git tag
v0.1
v0.2

C:\Users\DELL\simple website>git checkout v0.1
Note: switching to 'v0.1'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 84d3434 second
```