Part 5: Tags Feature

Research Summary

Git tags are named references that point to specific commits, serving as permanent markers in the repository's history. Unlike branches, which are mutable pointers that move as new commits are added, tags are immutable references that remain fixed to a particular commit. While commits represent snapshots of the project at specific points in time, tags are simply human-readable labels that make it easier to reference important commits. Tags are commonly used to mark release points (v1.0.0, v2.0.0), milestones in development, or any significant state worth remembering. In Git's storage mechanism, lightweight tags (the simplest form) are stored as files in `.git/refs/tags/` containing just the commit hash they point to, creating a direct relationship between the tag name and the commit without additional metadata.

Design and Implementation Approach

The implementation follows Git's lightweight tag model, storing tags as simple file references in '.git/refs/tags/' directory. Each tag is a text file named by the tag's name, and containing the commit hash it references. The repository layer provides four core operations: create_tag which validates uniqueness and resolves references to commit hashes, delete_tag which removes tag files, tags() which lists all tag names, and tag_exists for existence checking. The create_tag method accepts a commit reference parameter that defaults to 'HEAD', allowing users to tag either the current commit or any specific commit by hash. Error handling follows the existing pattern: CLI input validation and repository logic enforcement (ValueError for empty names, RepositoryError for duplicates or invalid states).

link for implementation branch: https://github.com/Shahaf19/asp-caf-assignment/tree/task4