Reflection on Git Branching, Merge Conflict, and Resolution

Reflection on Git Branching, Merge Conflict, and Resolution

This project was an individual exercise designed to help me practice real-world version control workflows using Git and GitHub. The objective was to create a small but functional project, develop features on separate branches, intentionally trigger a merge conflict, and then resolve it manually. Through this assignment, I gained practical experience with branching, merging, conflict handling, and communicating clearly through commits and screenshots.

1. What Caused the Merge Conflict

The merge conflict in my project occurred because two different branches modified the exact same line of code inside index.html — specifically the main <h1> heading of the landing page.

- On feature-a, I changed the heading to: “QuickStart — Launch beautiful pages fast (Feature A)” - On feature-b, I changed the same heading to: “QuickStart - Rapid pages for your portfolio (Feature B)”

Git can automatically merge changes as long as they affect different lines or regions of a file. However, because both branches changed the same line, Git was unable to determine which version to keep. As a result, when I attempted to merge feature-b into main (after merging feature-a cleanly), Git correctly raised a conflict.

2. How I Resolved the Conflict

When the conflict occurred, Git paused the merge and marked the conflicting section inside index.html. Opening the file in VS Code showed the conflict clearly with Git’s special markers:

<<<<<<< HEAD (Feature A version) ======= (Feature B version) >>>>>>> feature-b

VS Code also displayed helpful merge options such as Accept Current Change, Accept Incoming Change, and Accept Both Changes.

To resolve the conflict: 1. I clicked “Accept Both Changes” to bring both versions into the file. 2. I manually cleaned the duplicated lines. 3. I created a final merged heading that represented both features clearly: QuickStart — Rapid, beautiful pages (merged A + B) 4. I saved the file and staged the resolution: git add index.html 5. I committed the resolved state: git commit -m "Resolve merge conflict: merged Feature A and Feature B" 6. Finally, I pushed the merge result to GitHub: git push origin main

3. What I Learned About Distributed Version Control

This assignment helped me understand several important concepts in Git:

a. The importance of branching Working on separate branches allowed me to isolate features and experiment without affecting the main codebase. I learned how branching improves collaboration and keeps work organised.

b. Why conflicts happen I learned that merge conflicts are not errors—they are just Git asking for clarification when automatic merging is ambiguous. Most conflicts occur when two developers change the same line or area of a file.

c. How to resolve conflicts professionally Before this project, merge conflicts felt intimidating. Now I understand: - how to read conflict markers - how to use VS Code’s merge tools - how to clean up combined changes - the importance of staging and committing resolved files

d. The value of clear commit messages Consistent and meaningful commit messages helped track the history of changes.

e. GitHub as a collaborative tool Pushing to GitHub, viewing branches online, and confirming the final merged code showed me how distributed teams work together through shared repositories.

f. Confidence with essential Git commands During this assignment, I practiced and now understand: - git init - git branch / git checkout - git add / git commit - git push / git pull - git merge - how to interpret Git merge output - how to fix unmerged changes

4. Conclusion

Overall, this assignment gave me a strong practical understanding of Git workflows, branching strategies, merge conflicts, and manual conflict resolution. I now feel more confident in using Git for projects, collaborating with others, and handling situations where automatic merging fails. This exercise closely mirrors what happens in real software development teams and has helped me strengthen my version control skills in a meaningful way.