Branching and Merging

Branching:

1. Set P4Merge as your diff and merge tool

```
$ git config --global diff.tool p4merge
```

```
$ git config --global difftool.p4merge.path "D:/Perforce/p4merge.exe"
```

```
$ git config --global merge.tool p4merge
```

```
$ git config --global mergetool.p4merge.path "D:/Perforce/p4merge.exe"
```

2. Create a new branch

```
$ git branch GitNewBranch
```

3. List all branches

```
$ git branch -a
  GitNewBranch
* main
  remotes/origin/main
```

4. Switch to the new branch

```
$ git checkout GitNewBranch
Switched to branch 'GitNewBranch'
```

5. Add new files with content

```
$ echo "This is file 1 in GitNewBranch" >> file1.txt
```

```
$ echo "Another file in GitNewBranch" >> file2.txt
```

6. Stage & commit

```
$ git add file1.txt file2.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'file2.txt', LF will be replaced by CRLF the next time Git touches it
```

```
$ git commit -m "Add file1 and file2 in GitNewBranch"
[GitNewBranch 4a1bb08] Add file1 and file2 in GitNewBranch
 2 files changed, 2 insertions(+)
 create mode 100644 file1.txt
 create mode 100644 file2.txt
```

7. Check status

```
$ git status
On branch GitNewBranch
nothing to commit, working tree clean
```

Merging:

1. Switch back to main

```
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
```

2. List differences between main and GitNewBranch (CLI view)

```
$ git diff main GitNewBranch
diff --git a/file1.txt b/file1.txt
new file mode 100644
index 0000000..f2a79f0
--- /dev/null
+++ b/file1.txt
@@ -0,0 +1 @@
+This is file 1 in GitNewBranch
diff --git a/file2.txt b/file2.txt
new file mode 100644
index 0000000..4285005
--- /dev/null
+++ b/file2.txt
@@ -0,0 +1 @@
+Another file in GitNewBranch
```

3. View visual differences using P4Merge

```
$ git difftool main GitNewBranch

Viewing (1/2): 'file1.txt'
Launch 'p4merge' [Y/n]? Y
```

4. Merge the branch into main

```
$ git merge GitNewBranch
Updating 3302d54..4a1bb08
Fast-forward
 file1.txt | 1 +
 file2.txt | 1 +
 2 files changed, 2 insertions(+)
 create mode 100644 file1.txt
 create mode 100644 file2.txt
```

5. If there are conflicts, git will open P4Merge

```
$ git mergetool
No files need merging
```

6. View merge history

```
$ git log --oneline --graph --decorate
* 4a1bb08 (HEAD -> main, GitNewBranch) Add file1 and file2 in GitNewBranch
* 3302d54 Add .gitignore to ignore .log files and log folder
* 8ebf8cf (origin/main) This is a Multi-line commit message
* 9efd41a Initial commit
```

7. Delete the merged branch

```
$ git branch -d GitNewBranch
Deleted branch GitNewBranch (was 4a1bb08).
```

8. Check the status

```
$ 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
```