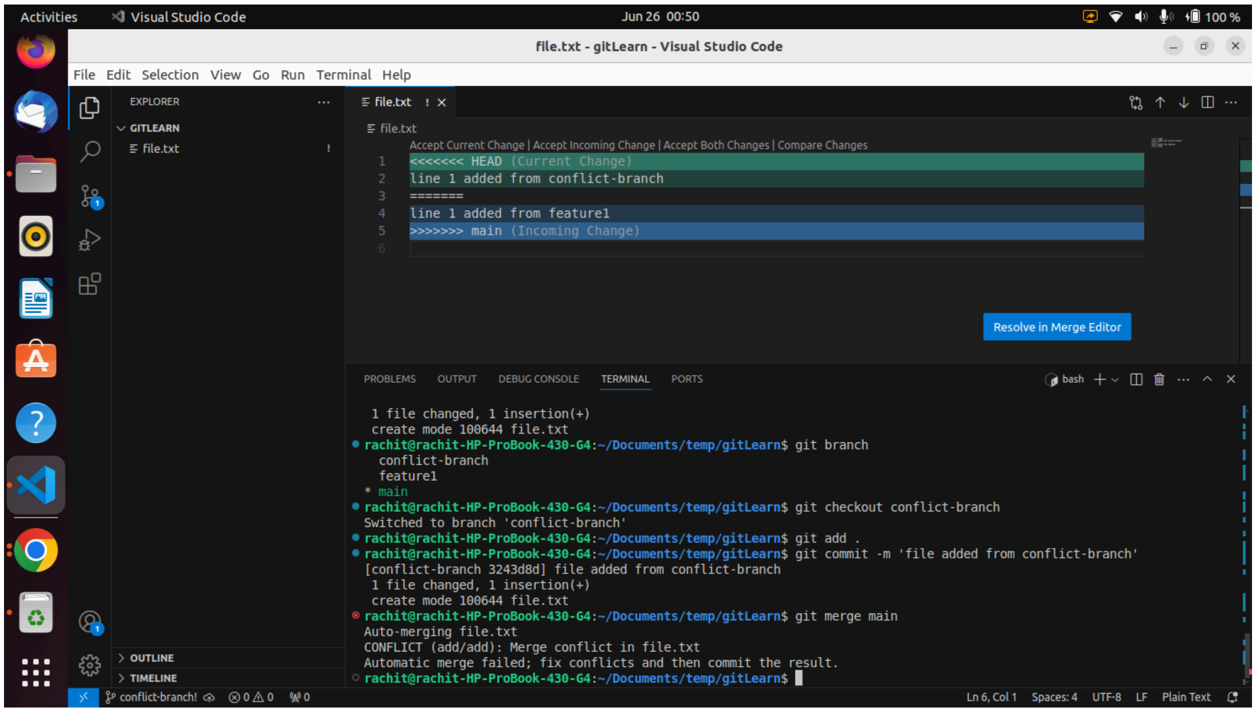
1. git config --global user.name "dineshchintu"
   git config --global user.email "musuku.dinesh@practo.com"
2. git branch feature-branch
   git checkout feature-branch
3. git branch
4. git checkout main
   git branch -d feature-branch
5. git reset --hard
6. git branch conflict-branch
7. git branch feature1
8. git checkout feature1
   touch file.txt
   git add .
   git commit -m ‘file added’
9. git checkout main
   git merge feature1
10. git checkout conflict-branch
    file is changed
    git add .
    git commit -m ‘file added from conflict-branch’
11. git merge main


Activities
Visual Studio Code
Jun 26 00:50
100 %
file.txt - gitLearn - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
GITLEARN
file.txt
OUTLINE
TIMELINE
file.txt
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
Resolve in Merge Editor
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
conflict-branch!
Ln 6, Col 1 Spaces: 4 UTF-8 LF Plain Text

```
<<<<<<< HEAD (Current Change)
line 1 added from conflict-branch
=======
line 1 added from feature1
>>>>>>> main (Incoming Change)
```

```
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
rachit@rachit-HP-ProBook-430-G4:~/Documents/temp/gitLearn$ git branch
  conflict-branch
  feature1
* main
rachit@rachit-HP-ProBook-430-G4:~/Documents/temp/gitLearn$ git checkout conflict-branch
Switched to branch 'conflict-branch'
rachit@rachit-HP-ProBook-430-G4:~/Documents/temp/gitLearn$ git add .
rachit@rachit-HP-ProBook-430-G4:~/Documents/temp/gitLearn$ git commit -m 'file added from conflict-branch'
[conflict-branch 3243d8d] file added from conflict-branch
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
rachit@rachit-HP-ProBook-430-G4:~/Documents/temp/gitLearn$ git merge main
Auto-merging file.txt
CONFLICT (add/add): Merge conflict in file.txt
Automatic merge failed; fix conflicts and then commit the result.
rachit@rachit-HP-ProBook-430-G4:~/Documents/temp/gitLearn$ 
```

12. Merge conflicts resolved.
13. git remote add origin https://github.com/dineshchintu/gitLearn.git
14. Done
15. Done
16. Done
17. git config --global alias.gitlol "log --oneline"
18. cd .git/hooks
    touch pre-commit
    chmod +x pre-commit
    dummy scripts added in pre-commit

19. git stash
    git checkout <other-branch>

20. git status
    git log -- <file-path>
    git checkout <commit-hash>^ -- <file-path>

21. git add <file-path>
    git commit --amend --no-edit
22. git reset --hard HEAD
23. git log(to find the hashId of the particular commit)
    git show <commit-hash>
24. git commit --amend -m "New commit message"
25. git checkout colleague-branch
26. git rebase -i HEAD~n(Git will prompt you to edit the combined commit message.we can modify the message as desired to summarize the changes.)
27. git status
    git reset HEAD <file-path>
28. Add “
    *.yml
    config/
    ‘
    To the .gitignore file
    git add .gitignore
    git commit -m "Add .gitignore to exclude .yml files and config directory"
29. git diff --name-only HEAD^ HEAD
30. Using git fetch followed by git rebase allows us to update our local branch with the latest changes from the remote repository without creating a merge commit
31. Look through the git reflog output to find the commit hash where the branch was deleted
    git checkout -b <branch-name> <commit-hash>
32. git clean -fd

33. git log <feature-branch>
    git checkout main
    git cherry-pick <commit-hash>
34. git log
    git checkout correct-branch
    git cherry-pick <commit-hash>
35. git cherry-pick <start-commit>^..<end-commit>
36. git clone <repository-url>
    cd <repository-name>
    git checkout -b <branch-name> origin/<branch-name>
37. git add .
    git commit -m "Your commit message here"
    git push origin main
38. git checkout -b new-feature
    git push origin new-feature
39. git clone <repository-url>

    cd <repository-name>
    git log
40. git log
    git rebase -i <commit-hash>^
    pick <commit-hash> Commit message
    drop <commit-hash> Commit message
    git push origin <branch-name> --force
41. git push origin --delete <branch-name>
42.
43.
44.
45.
46.
47.
48.

49.
50.
51.
52.
53.
54.
55.
56.
57.

58.
59.
60.
61.
62.
63.
64.
65.
66.
67.
68.
69.
70.
71.
72.
73.
74.
75.
76.
77.
78.
79.
80.

81.
82.
83.
84.
85.
86.
87.
88.
89.
90.
91.

92.
93.
94.
95.
96.