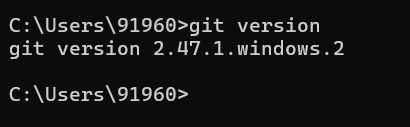
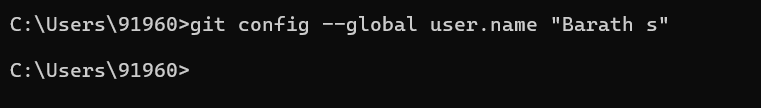
**GITHUB EXERCISE -1**

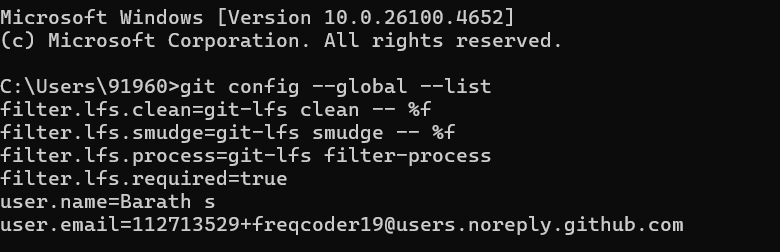
1. TO CHECK IF GIT CLIENT IS INSTALLED PROPERLY: OPEN GIT BASH SHELL AND EXECUTE

**T**

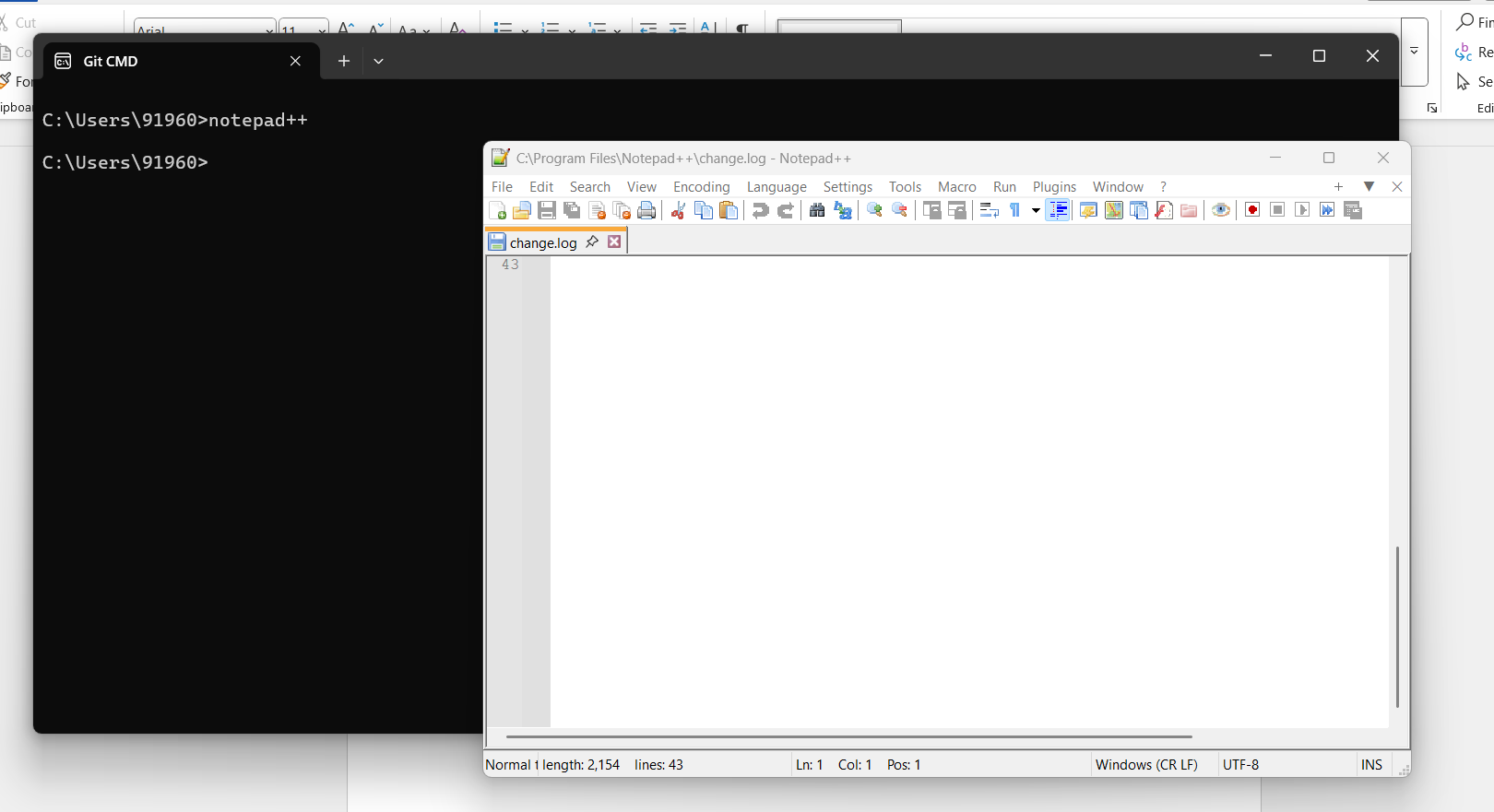
1. TO CONFIGURE USER LEVEL CONFIGURATION OF USER ID AND EMAIL ID EXECUTE



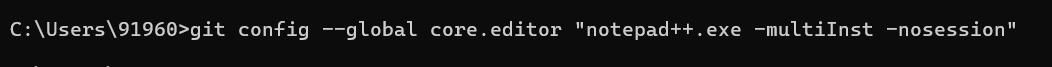
1. TO CHECK IF THE CONFIGURATION IS PROPERLY SET, EXECUTE THE FOLLOWING COMMAND.



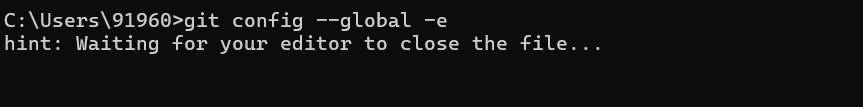
1. INTEGRATE NOTEPAD++.EXE TO GIT AND MAKE IT A DEFAULT EDITOR

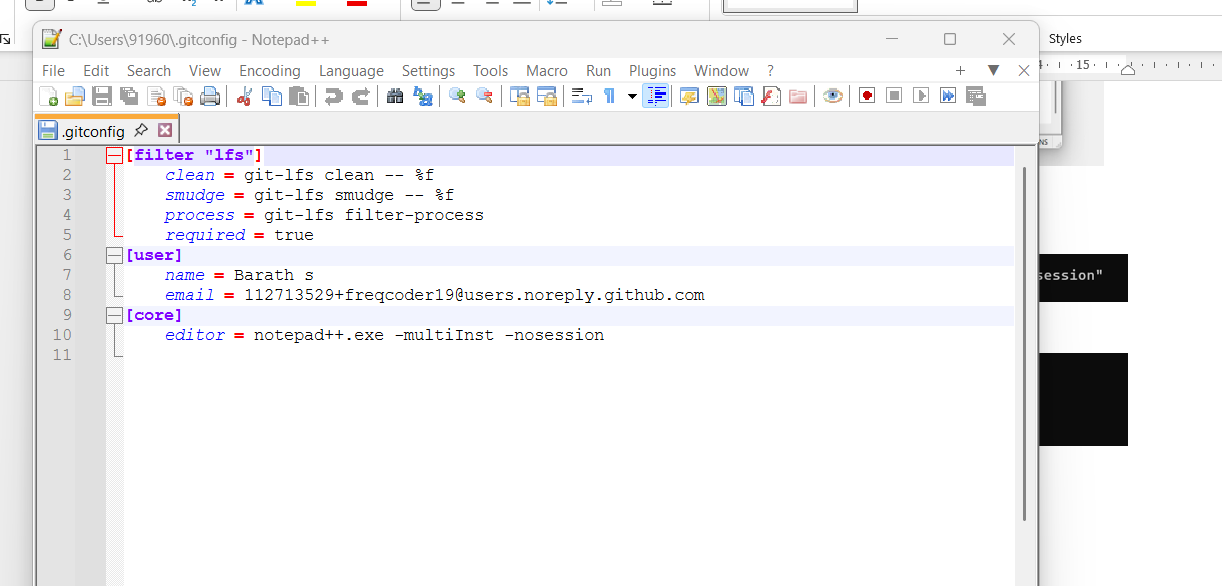
****

1. TO CONFIGURE THE EDITOR, EXECUTE THE COMMAND

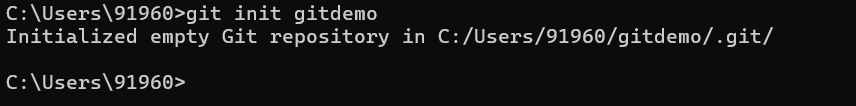
****

1. TO VERIFY IF NOTEPAD++ IS THE DEFAULT EDITOR, EXECUTE THE COMMAND

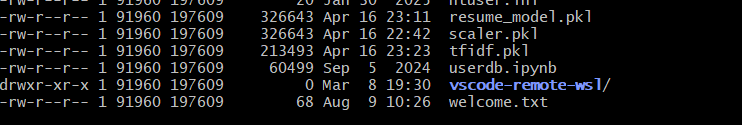
****

****

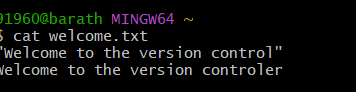
1. ADD A FILE TO SOURCE CODE REPOSITORY

****

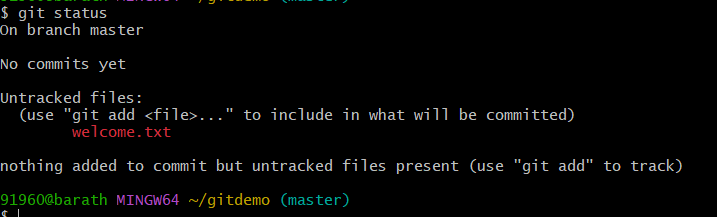
1. TO VERIFY THE CONTENT, EXECUTE THE COMMAND

****

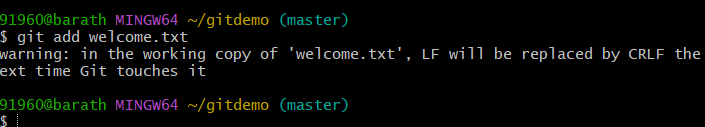
1. TO VERIFY THE CONTENT, EXECUTE THE COMMAND

****

1. CHECK THE STATUS BY EXECUTING

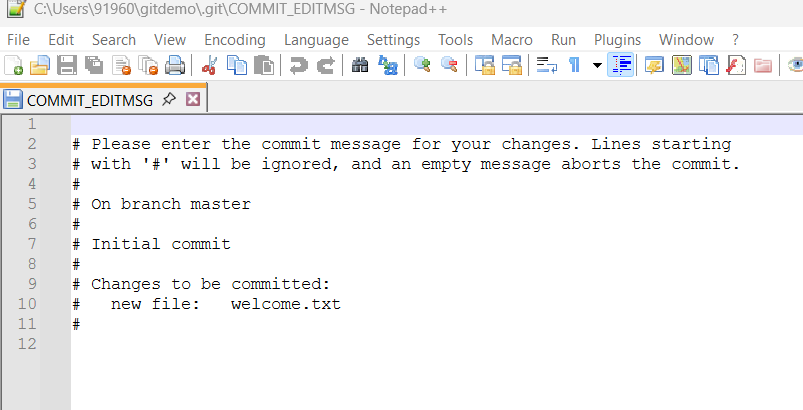
****

1. TO MAKE THE FILE TO BE TRACKED BY GIT REPOSITORY, EXECUTE THE COMMAND

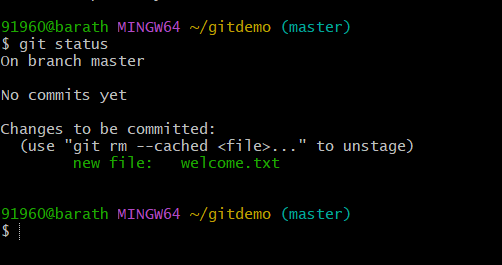
****

1. TO ADD MULTI LINE COMMENTS, WE ARE OPENING DEFAULT EDITOR TO COMMENT. EXECUTE THE COMMAND

****

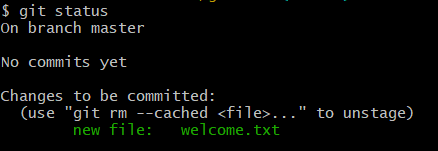
****

1. TO CHECK IF LOCAL AND “WORKING DIRECTORY” GIT REPOSITORY ARE SAME, EXECUTE GIT STATUS

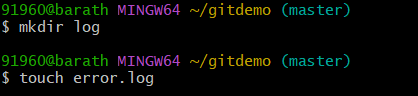
****

**GITHUB EXERCISE -2**

1. First we need to ensure the git demo directory is present



1. Then we need to create a log folder and log file



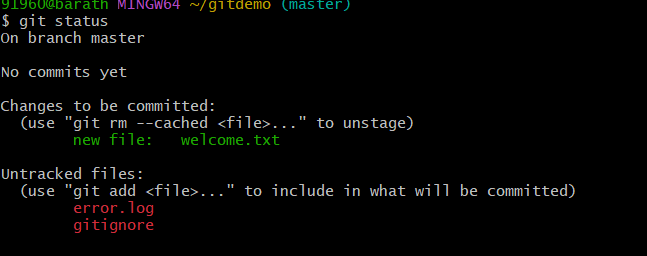
1. We need to create an gitignore file



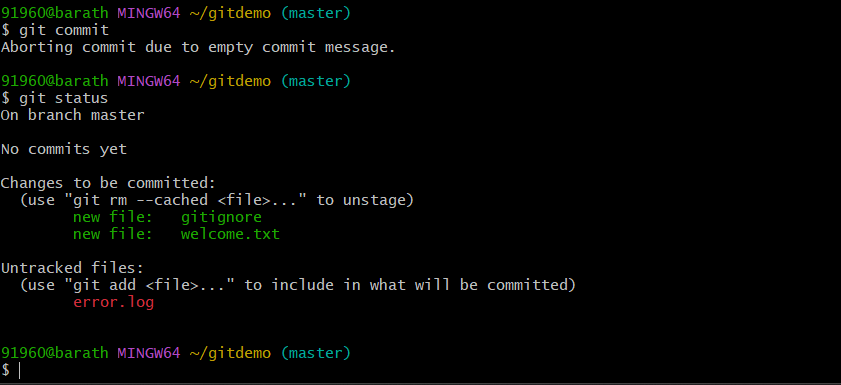
1. We need to type and save below two commands



1. Save the file check status

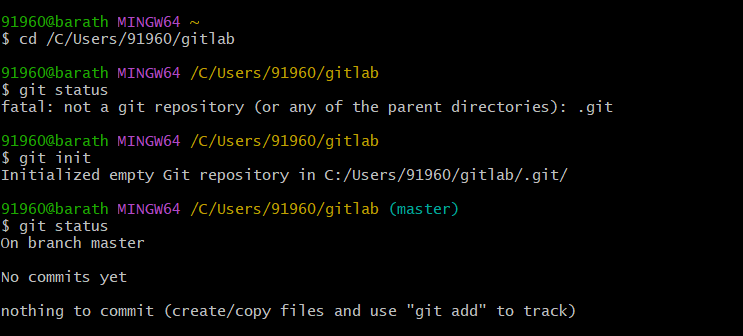


1. Now we add the gitignore file add add the log values to repository and mention clearly that these values to be ignore

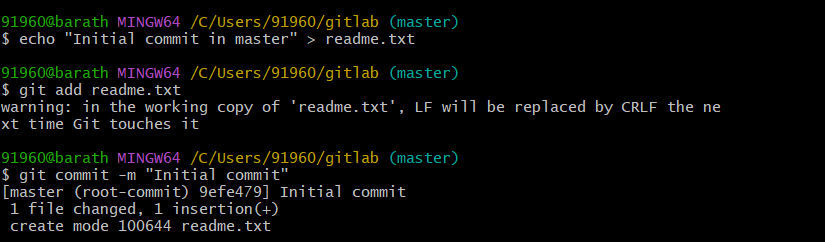


**GITHUB EXERCISE -3**

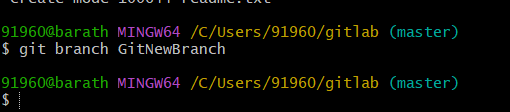
1 ) we create a new repository



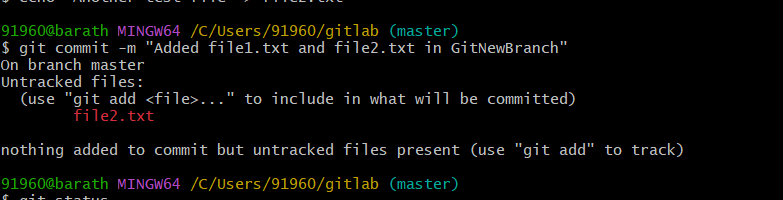
2) We need atleast one file or folder to create a branch in the repository we can’t do it in a empty repository so we add some files for branching purpose



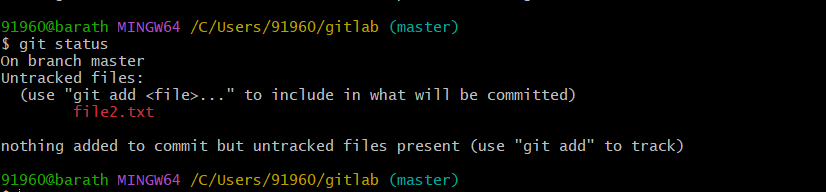
3) create a new branch

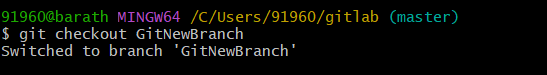


4) changing the file to new branch

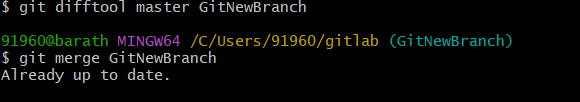


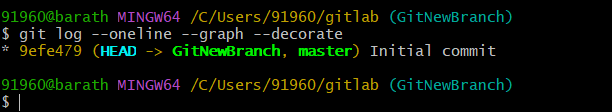
5) checking the git status after branching

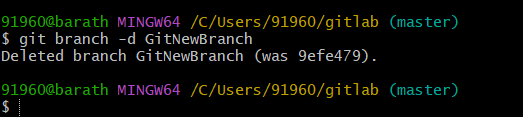




**MERGING THE BRANCHES**

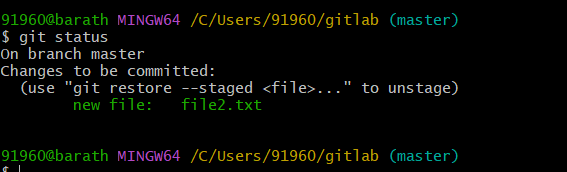




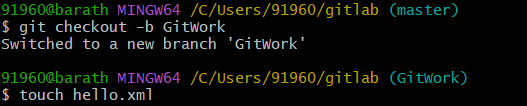
**Deleting a branch**

**GITHUB EXERCISE -4**

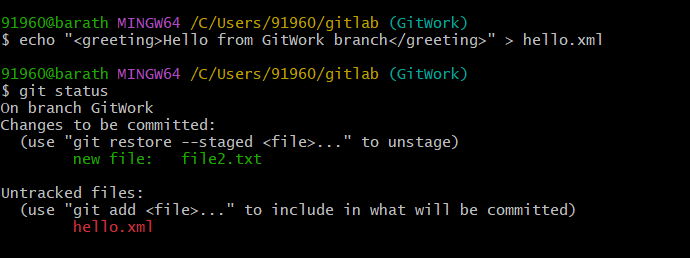
1. Verify if master is in clean state.



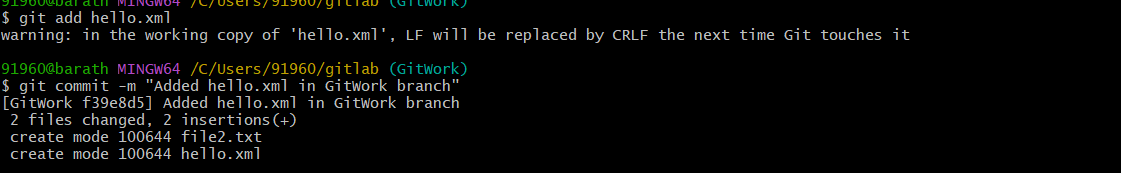
1. Create a branch **“GitWork”.** Add a file “hello.xml”.



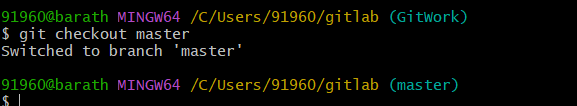
1. Update the content of “hello.xml” and observe the status



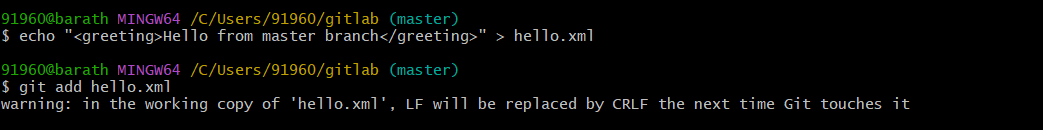
1. git commit -m "Added hello.xml in GitWork branch"



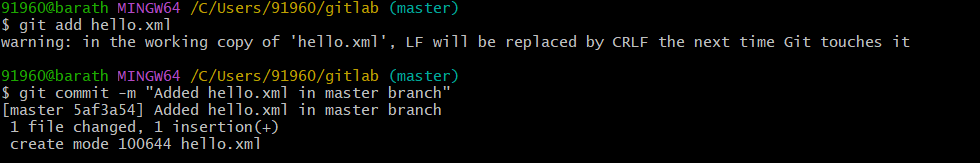
1. switch to master



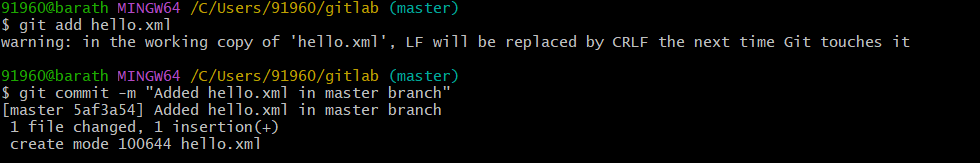
1. Add a file **“hello.xml”** to the master and add some different content than previous.



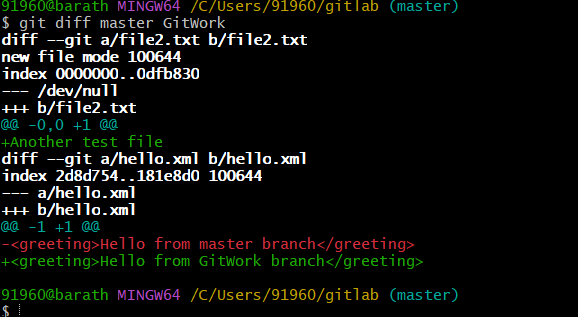
1. Observe the log by executing **“git log –oneline –graph –decorate –all”**



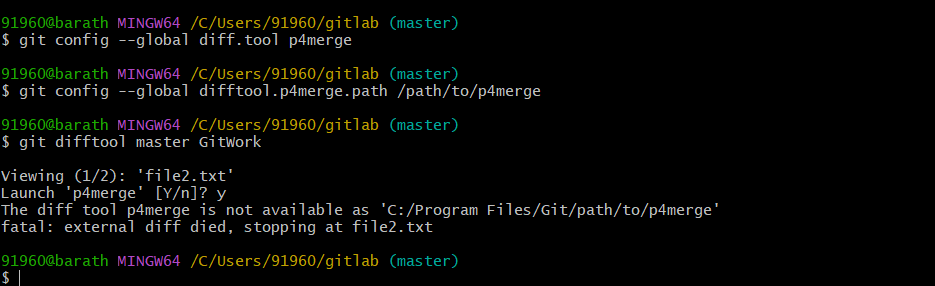
1. Observe the log by executing **“git log –oneline –graph –decorate –all”**



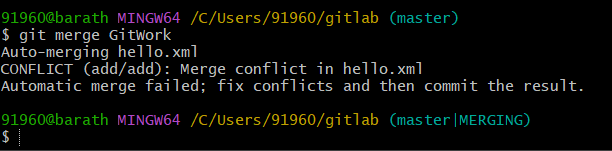
1. Check the differences with Git diff tool



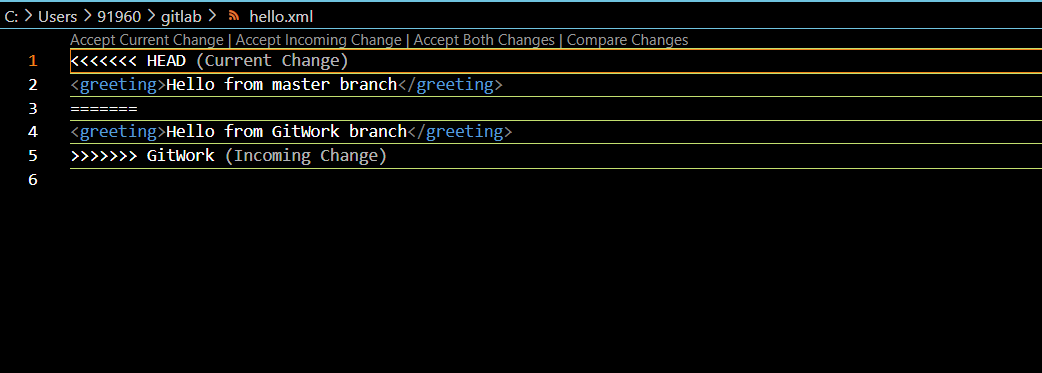
1. For better visualization, use P4Merge tool to list out all the differences between master and branch



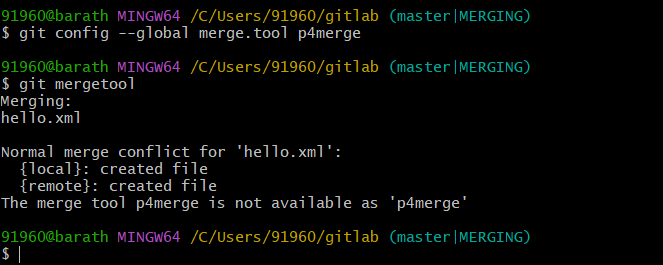
1. Merge the bran to the master



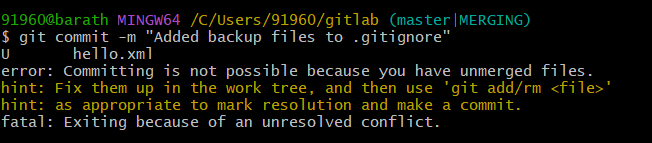
1. Observe the git mark up.

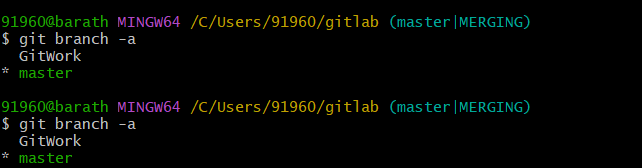


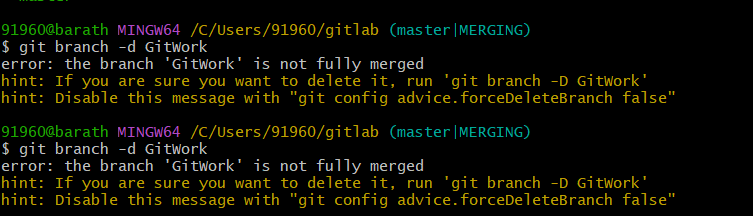
1. Use 3-way merge tool to resolve the conflict

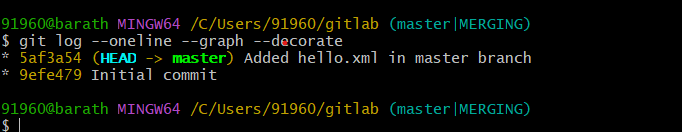


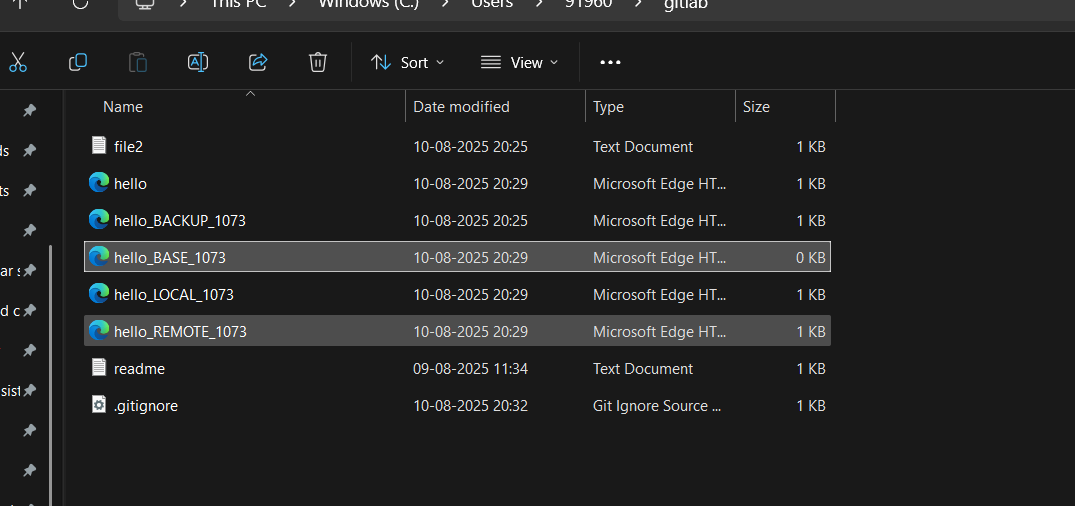
1. Commit the changes to the master, once done with conflict



2. 

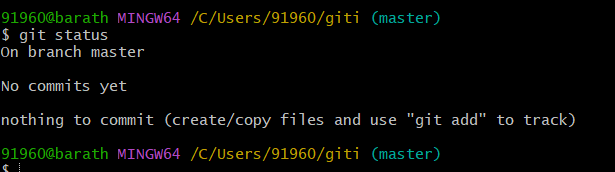




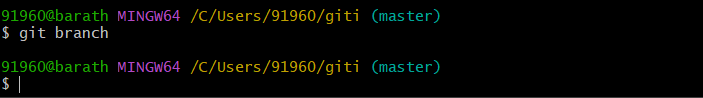


**GITHUB EXERCISE -5**

1. **Check status**

****

1. List out all the available branches.

****