**Week 8 - GIT - Hands-on**

DEBANJAN KAURI(Superset ID-6362388)

**Question 1: Set Up Git & Notepad++**

Scenario:

Set up Git and integrate Notepad++ as the default editor. Create and push a Git project named GitDemo.

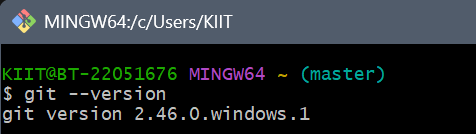
To achieve this:

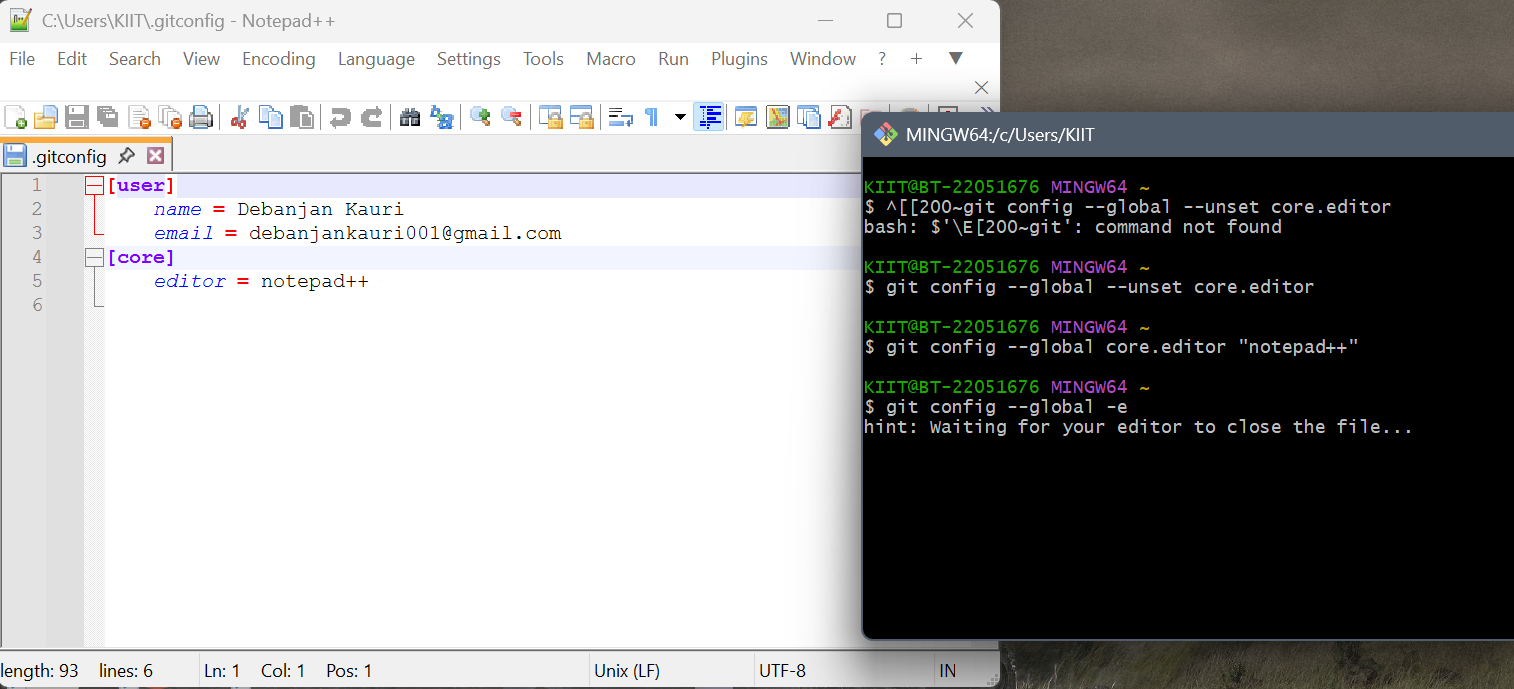
1. Git Setup & Configuration

2. Notepad++ Integration

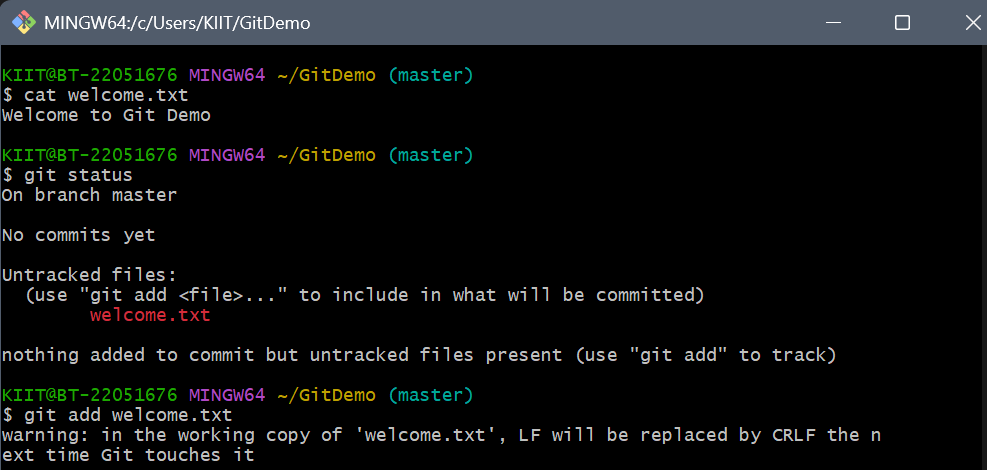
3. Create & Manage Git Project

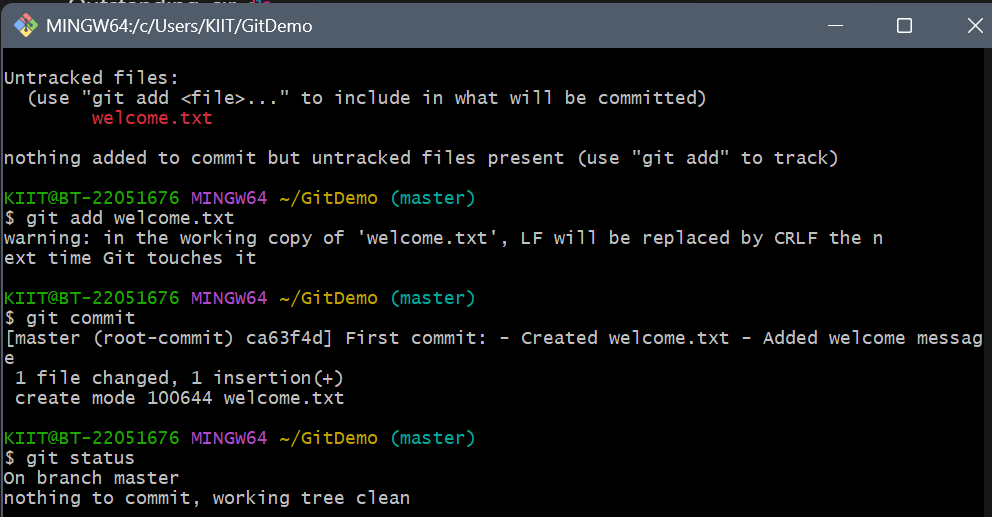
4. Connect to GitLab



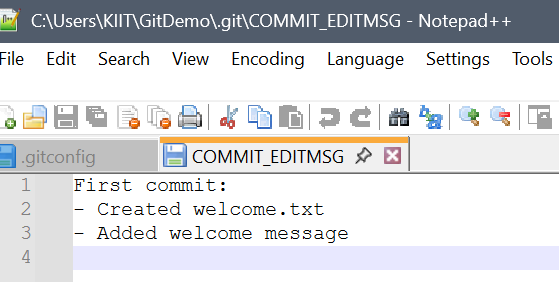


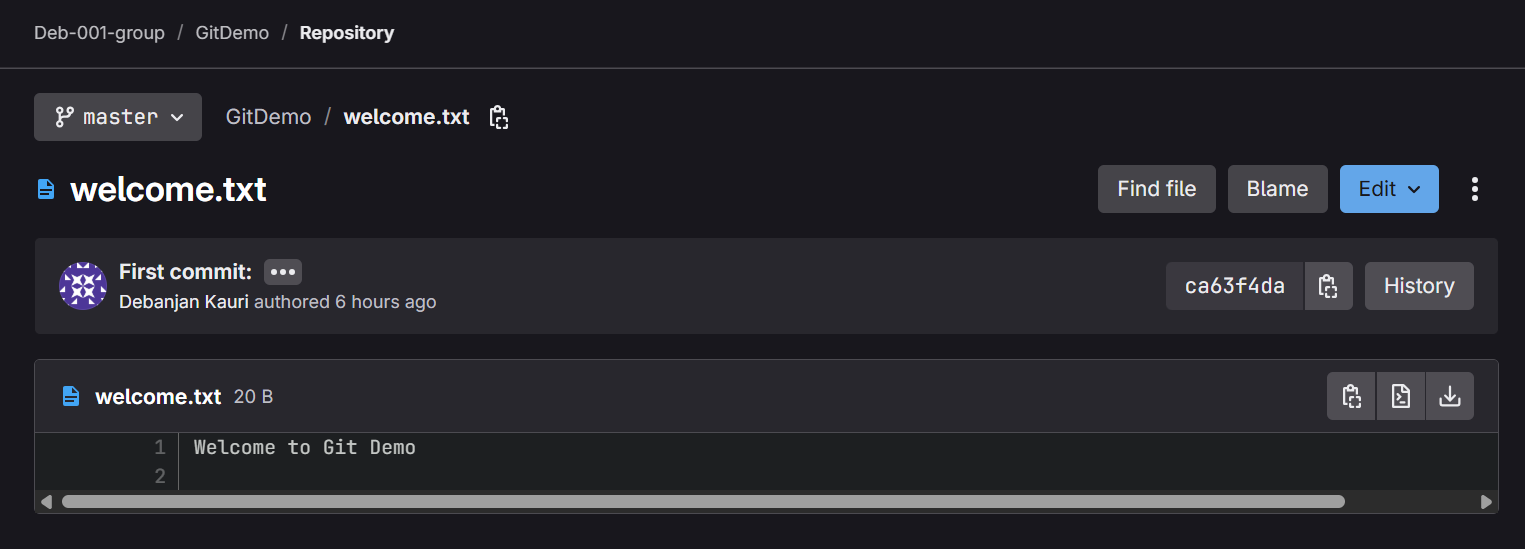












**Question 2: Git .gitignore lab**

Scenario:

Ignore .log files and log/ folder from being tracked by Git using .gitignore.

To achieve this:

1. In your Git working directory, create a file with .log extension and a folder named log/.

2. Open or create a .gitignore file in the root directory

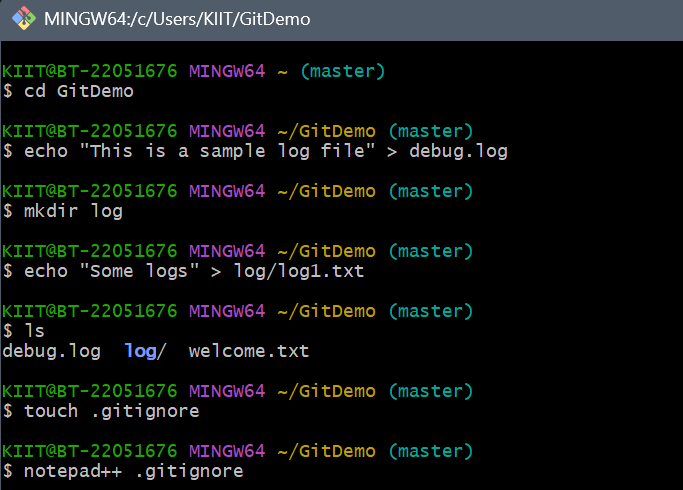
3. Add the following lines:

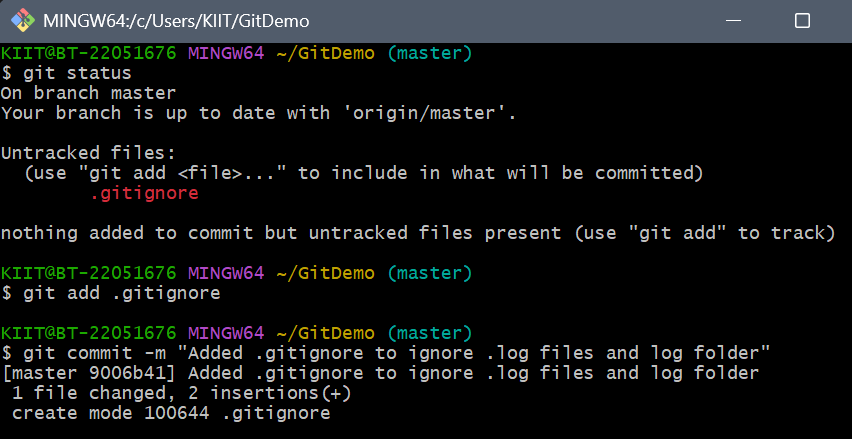
\*.log

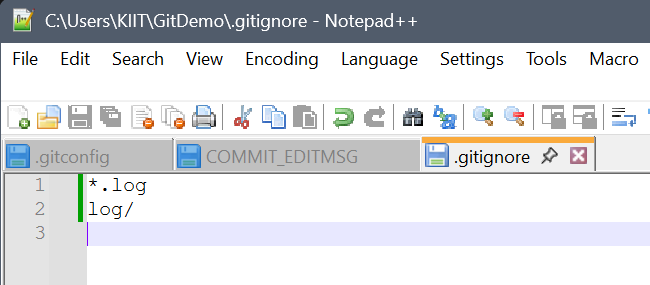
log/

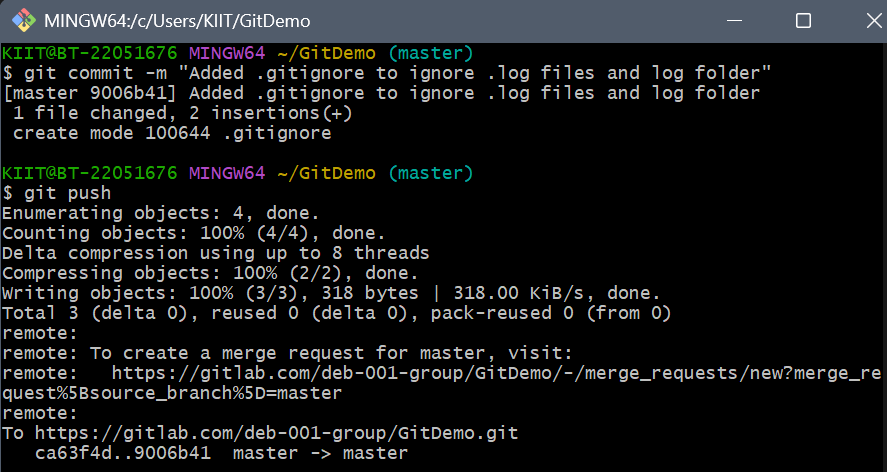
4. Save and close .gitignore.

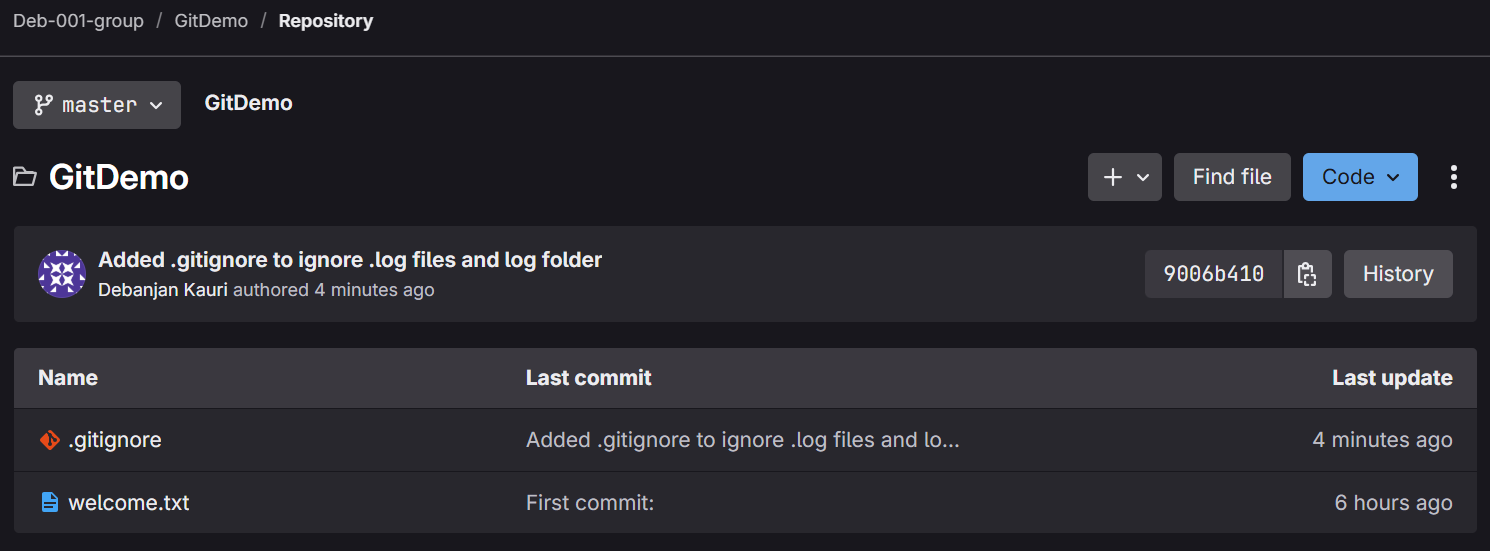
5. Run git status to verify.











**Question 3: Branching & Merging Git task**

Scenario:

You're building a new feature separately to avoid breaking the main (master) code. After testing, you'll merge it back and delete the branch.

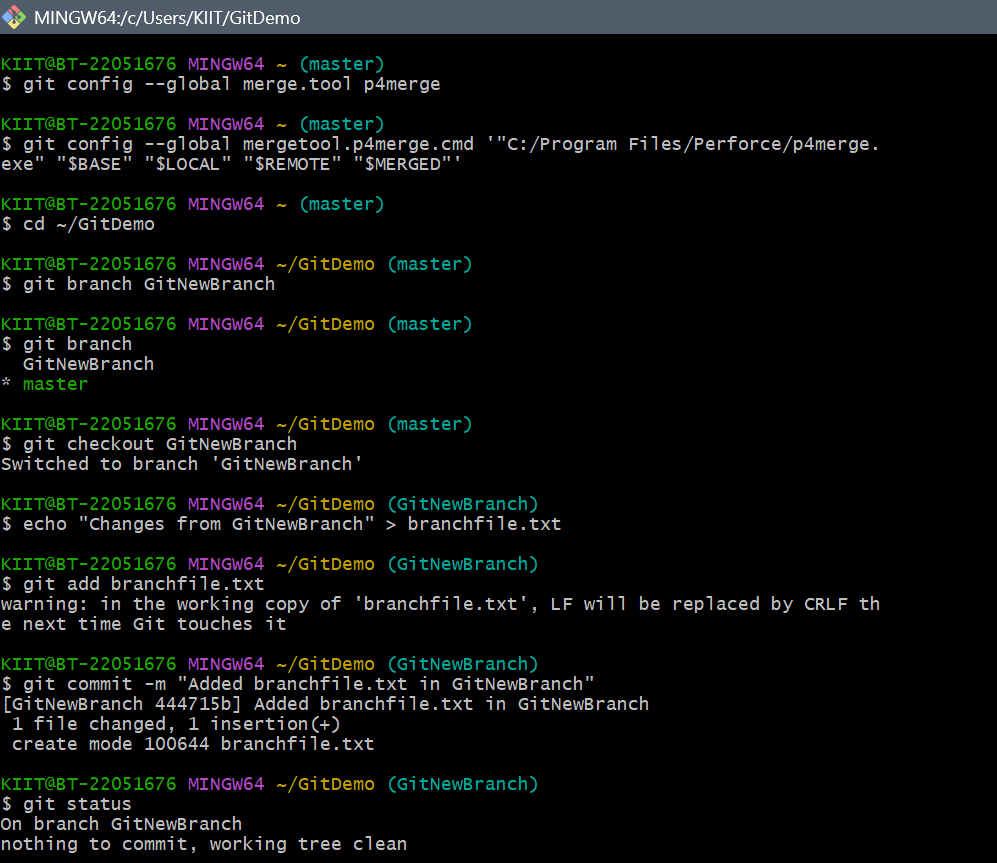
To achieve this:

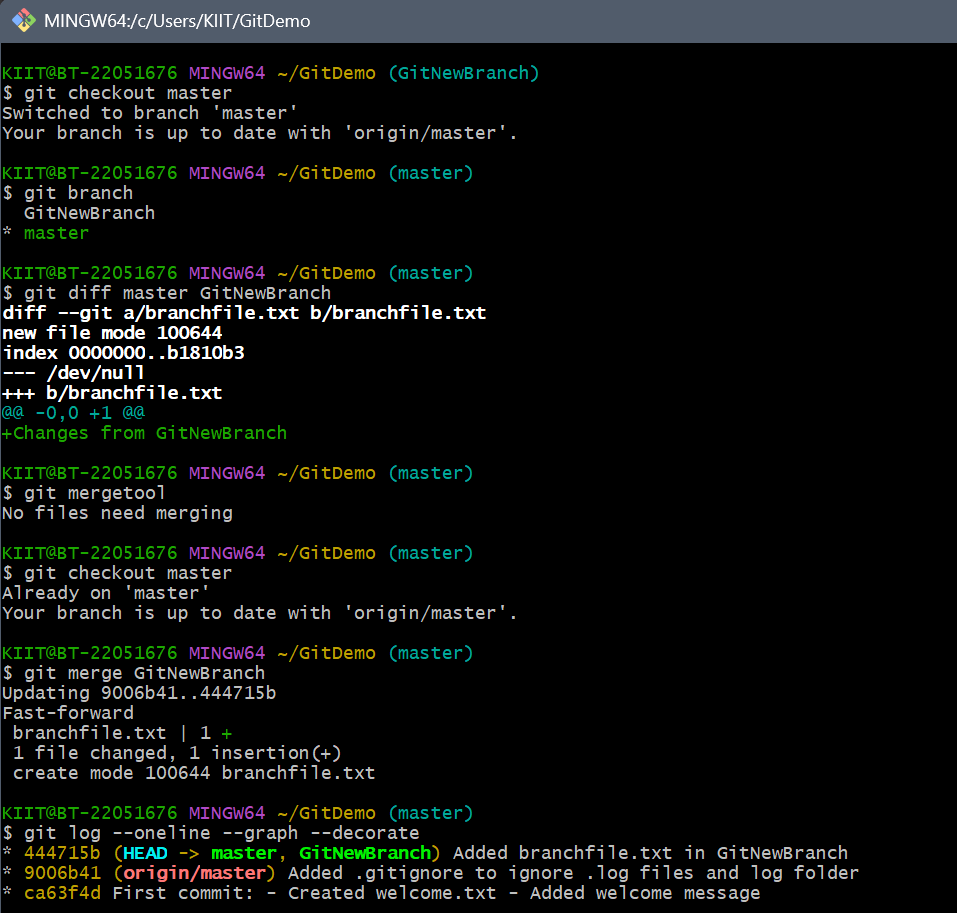
1. Branching:

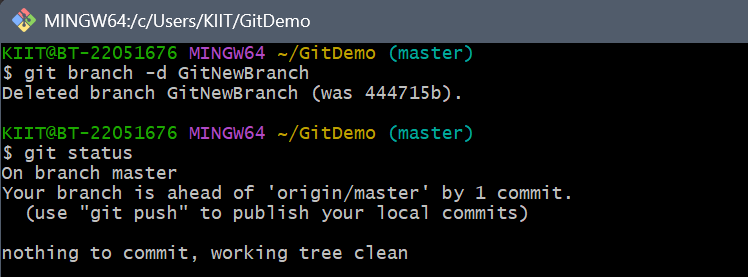
* Create Branch & List branches (local + remote)
* Switch to new branch
* Add files + content → commit & Check status

2. Merging:

* Switch to master
* Show CLI diff (branch vs master)
* Show visual diff (with P4Merge)
* Merge branch into master & View merge log
* Delete branch post-merge







**Question 4: Conflict Resolution in Git**

Scenario:

Two developers edit hello.xml in parallel on master and GitWork.On merging, Git finds conflicting changes and prompts for resolution.

To achieve this:

1. Create branch GitWork, add & commit hello.xml.

2. Switch to master, add different content in hello.xml, commit.

3. Merge GitWork into master → conflict occurs.

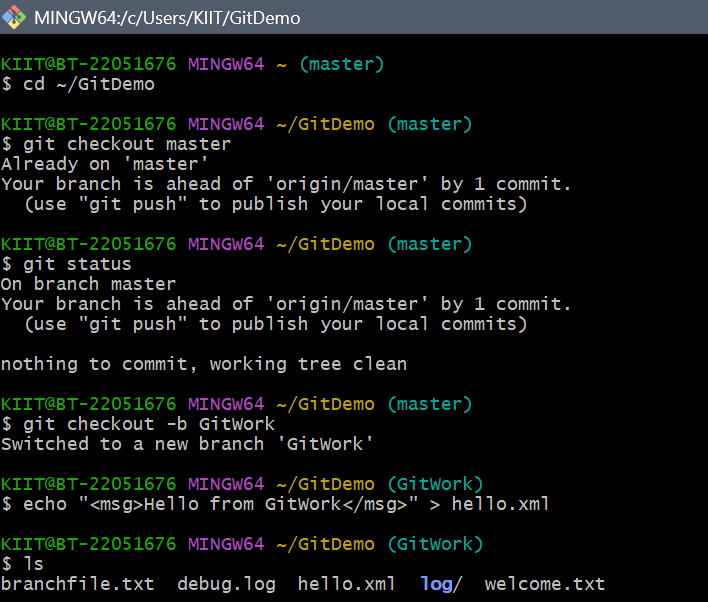
4. Use git diff or git mergetool to view conflict.

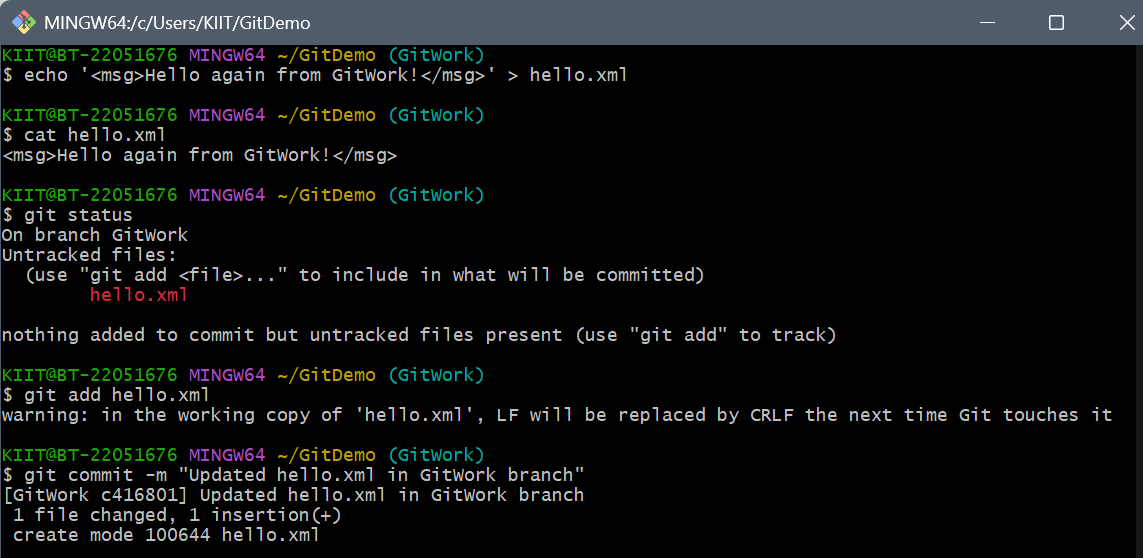
5. Manually edit and resolve hello.xml.

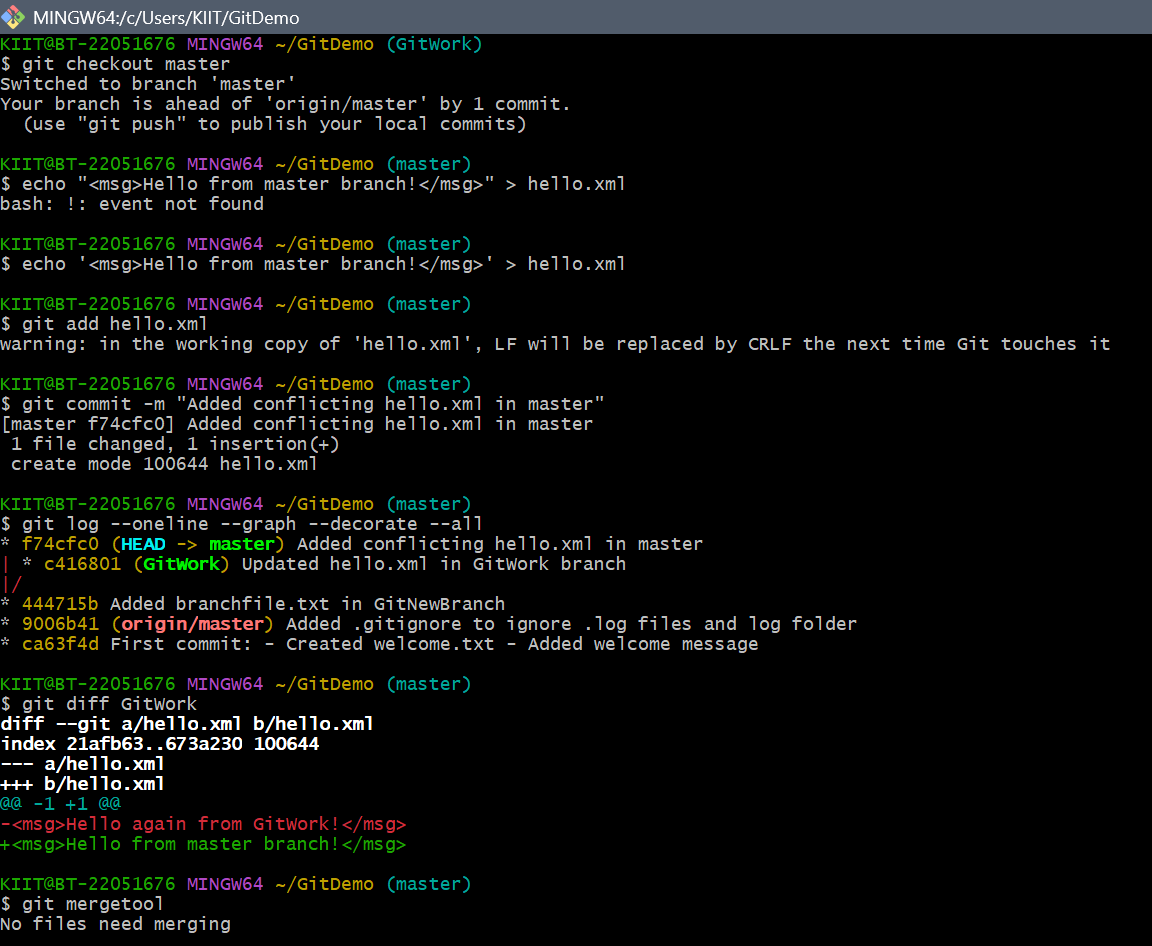
6. Add resolved file, commit changes.

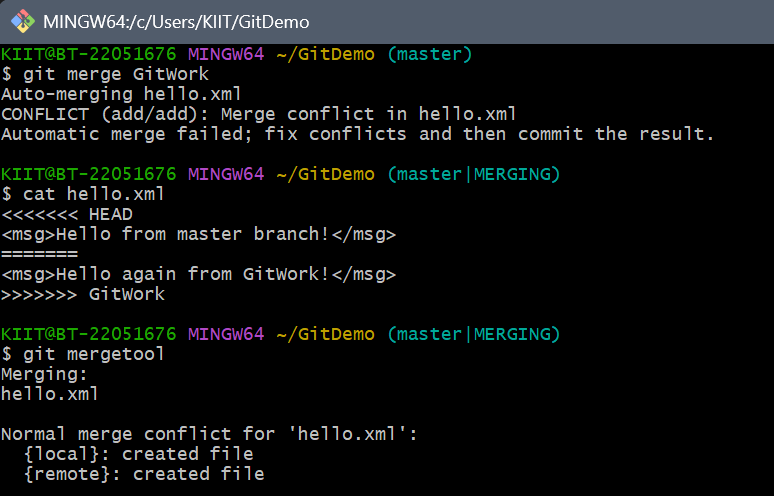
7. Add .gitignore for backup files, commit.

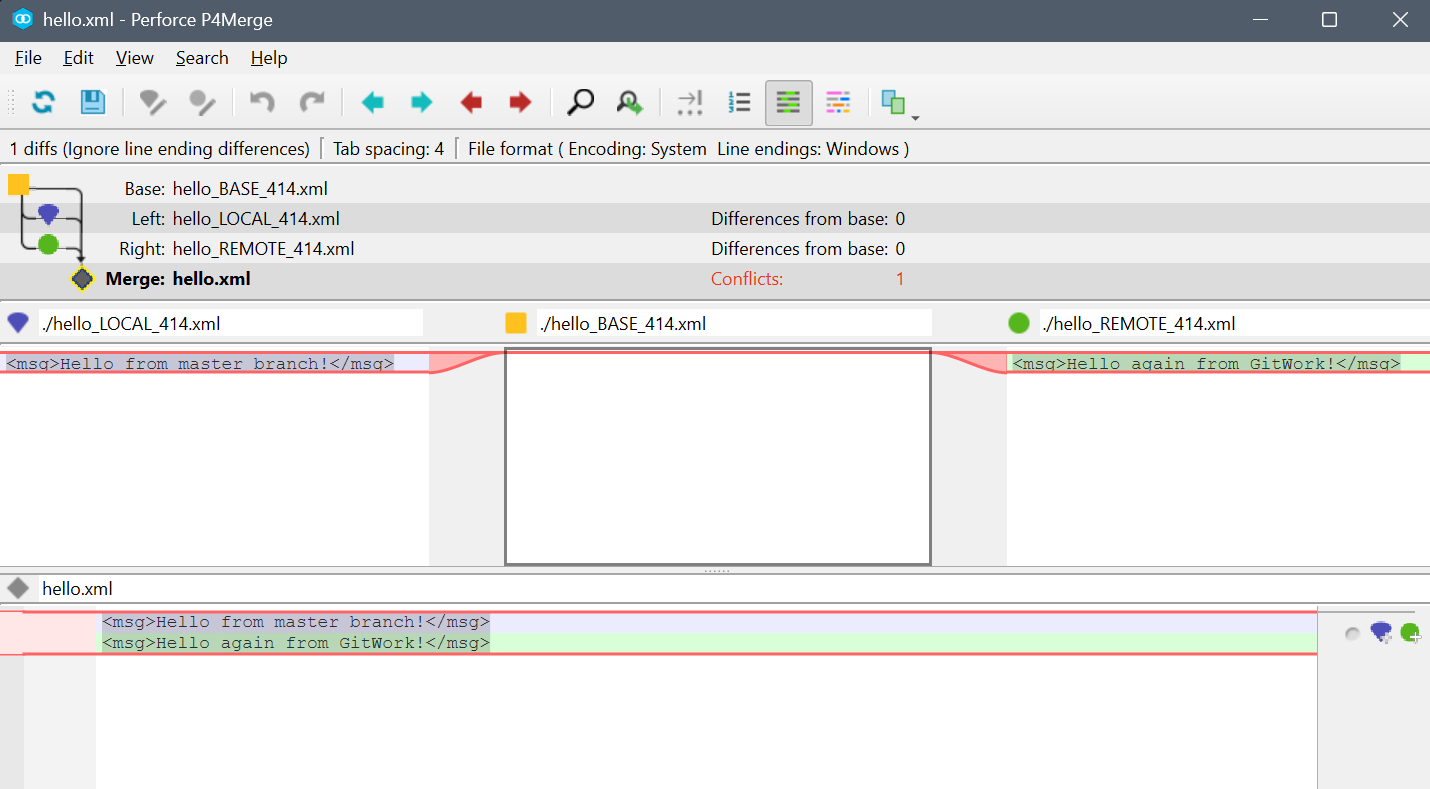
8. Delete GitWork, verify with git log --oneline --graph.

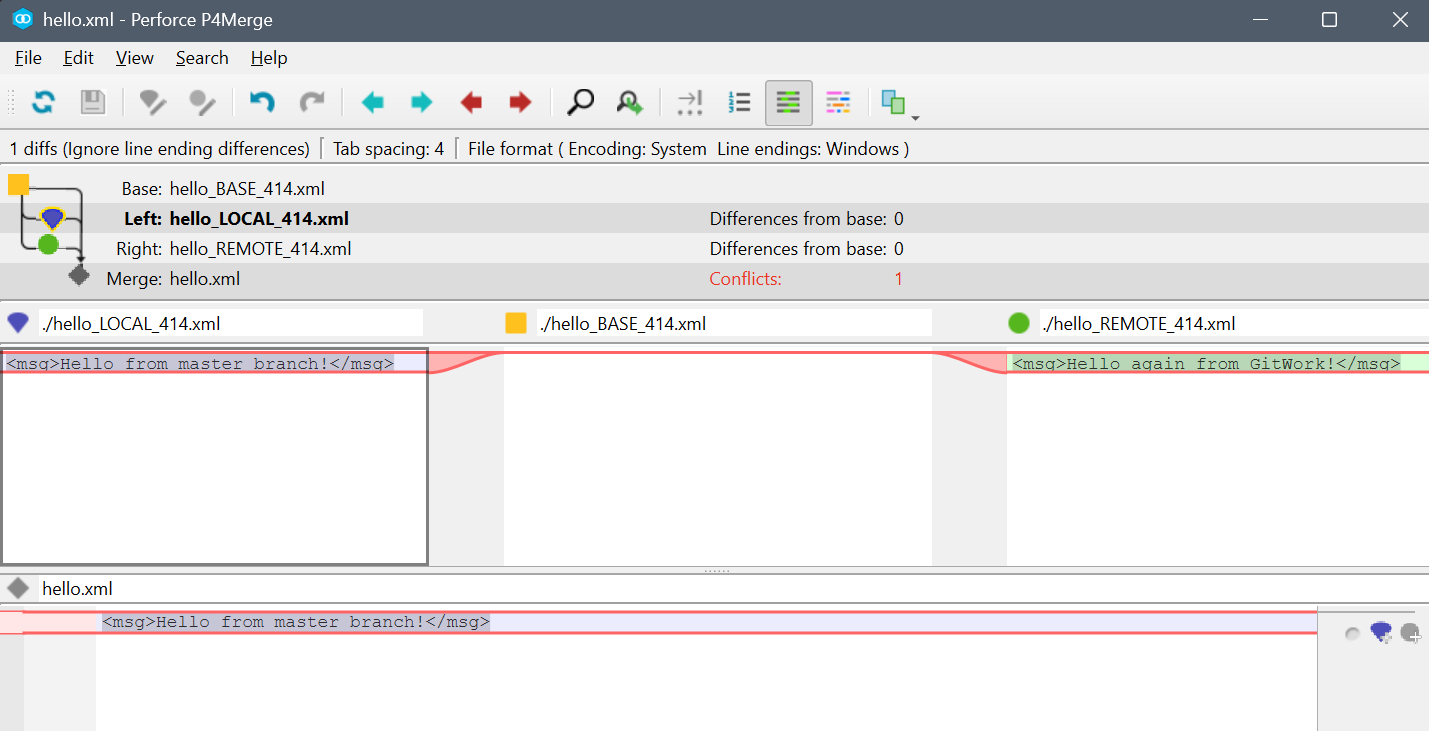


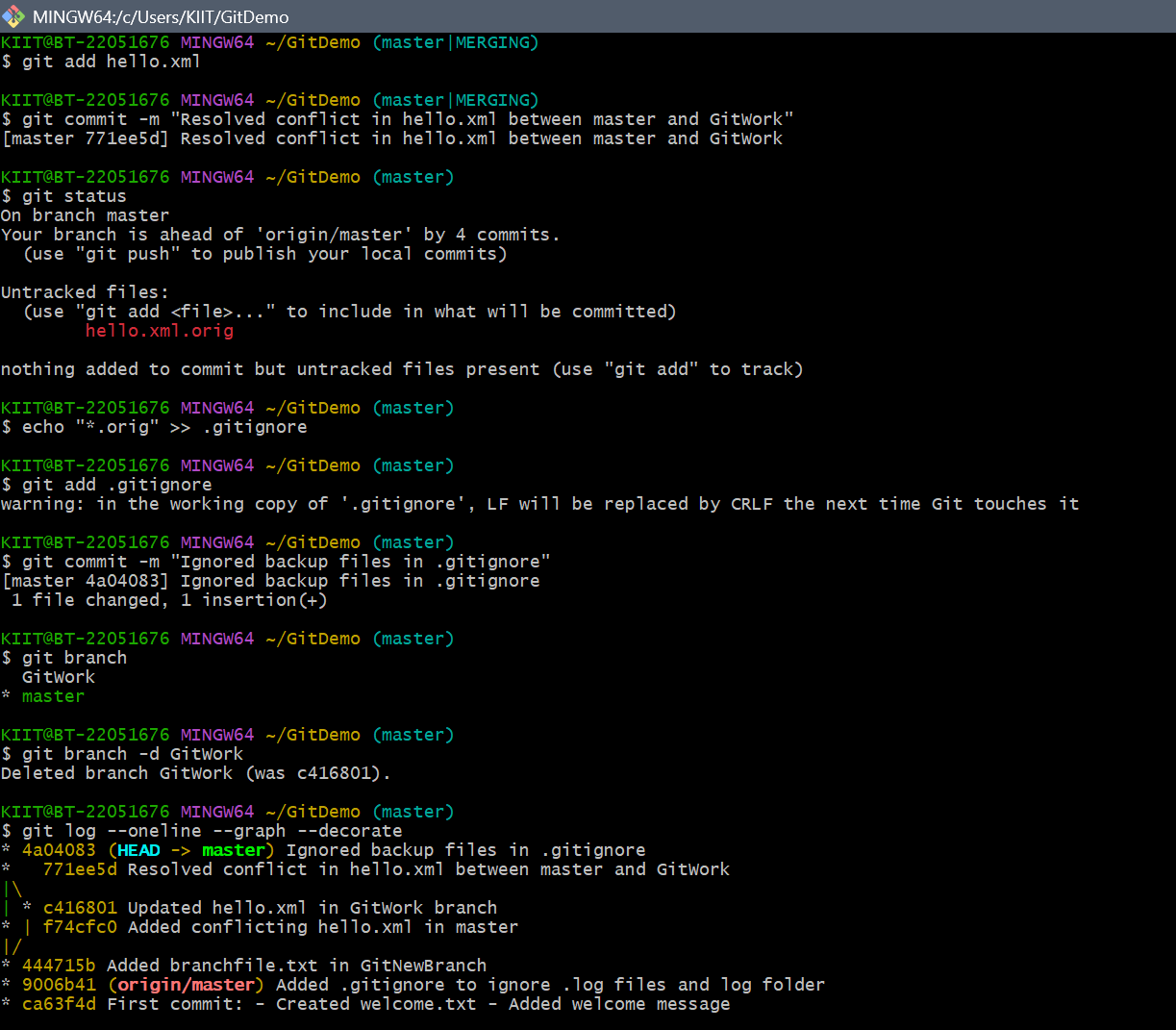












**Question 5: Clean Up and Push to Remote Repository**

Scenario:

Sync local repository with remote after making final changes, ensuring a clean and up-to-date main branch.

To achieve this:

1. Check the clean working state.

2. View available branches.

3. Pull remote updates.

4. Push local changes.

5. Verify sync.

