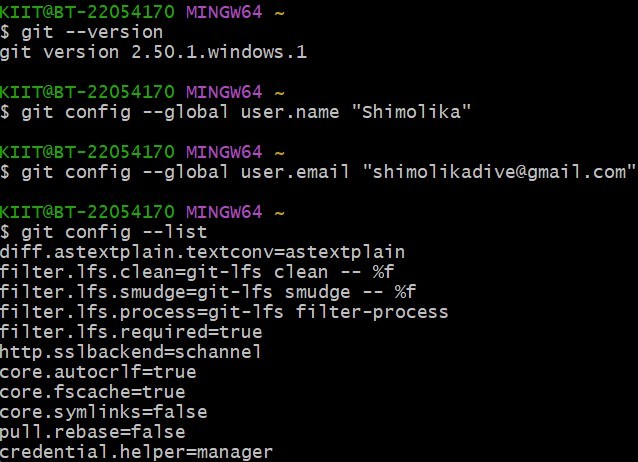
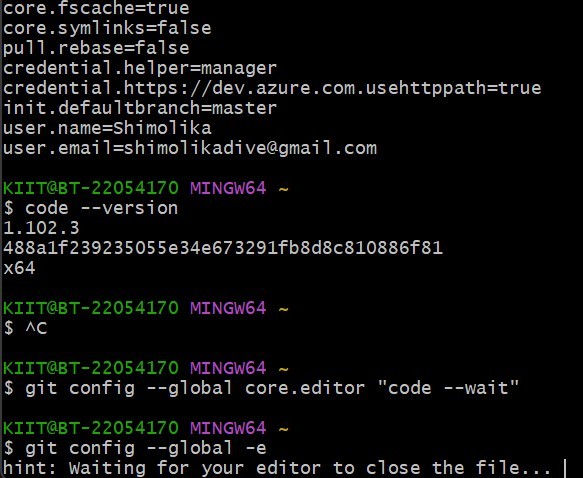
WEEK 8 MANDATORY HANDS-ON(GIT)

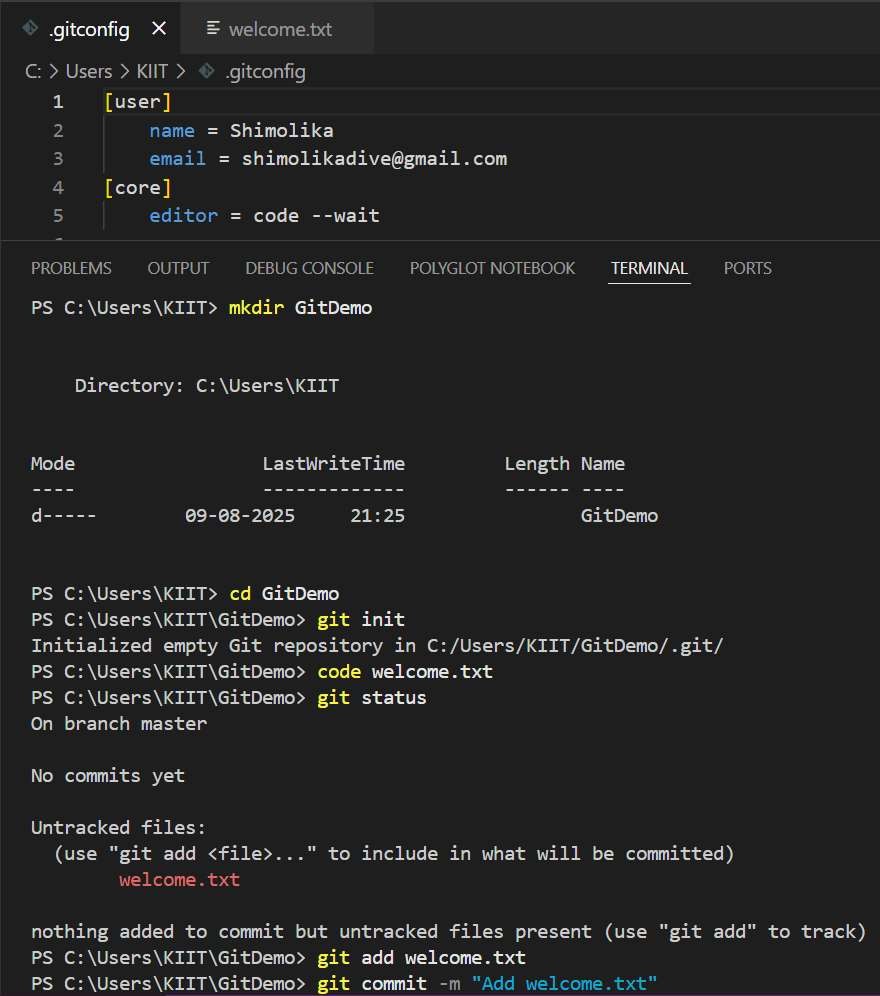
1. **Setting up the machine with Git configuration.**

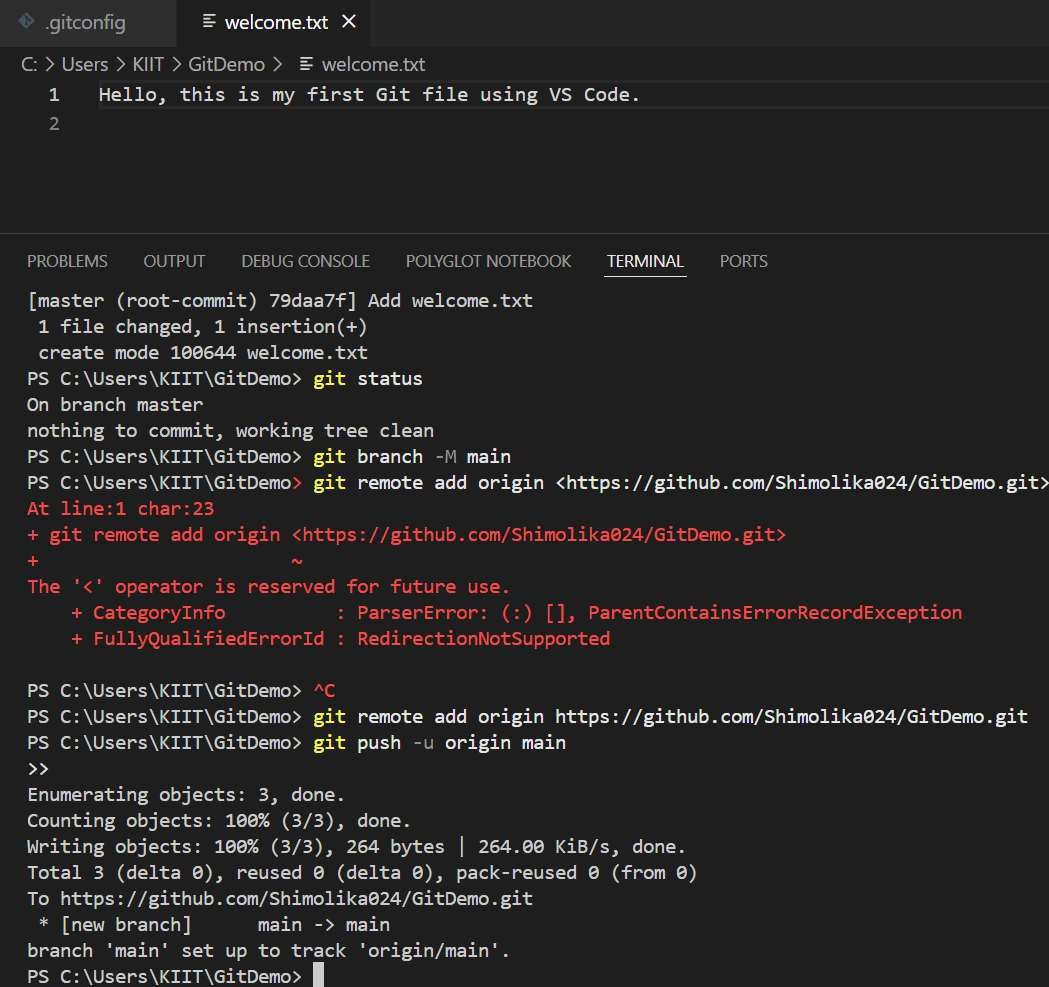
****

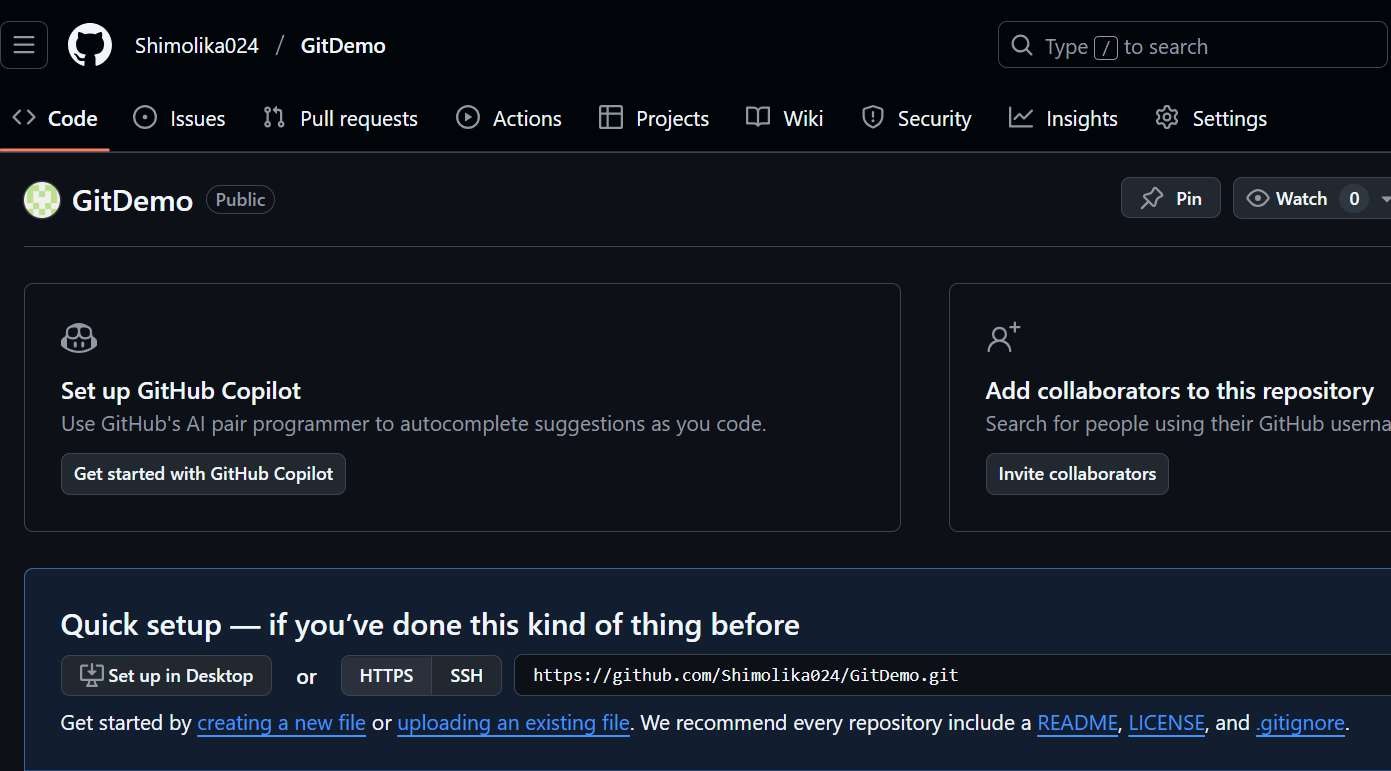
1. **To configure the editor**

****

1. **Add a file to source code repository. Check the status by executing . Make the file to be tracked by Git repository. Signup with GitLab and create a remote repository “GitDemo”.**







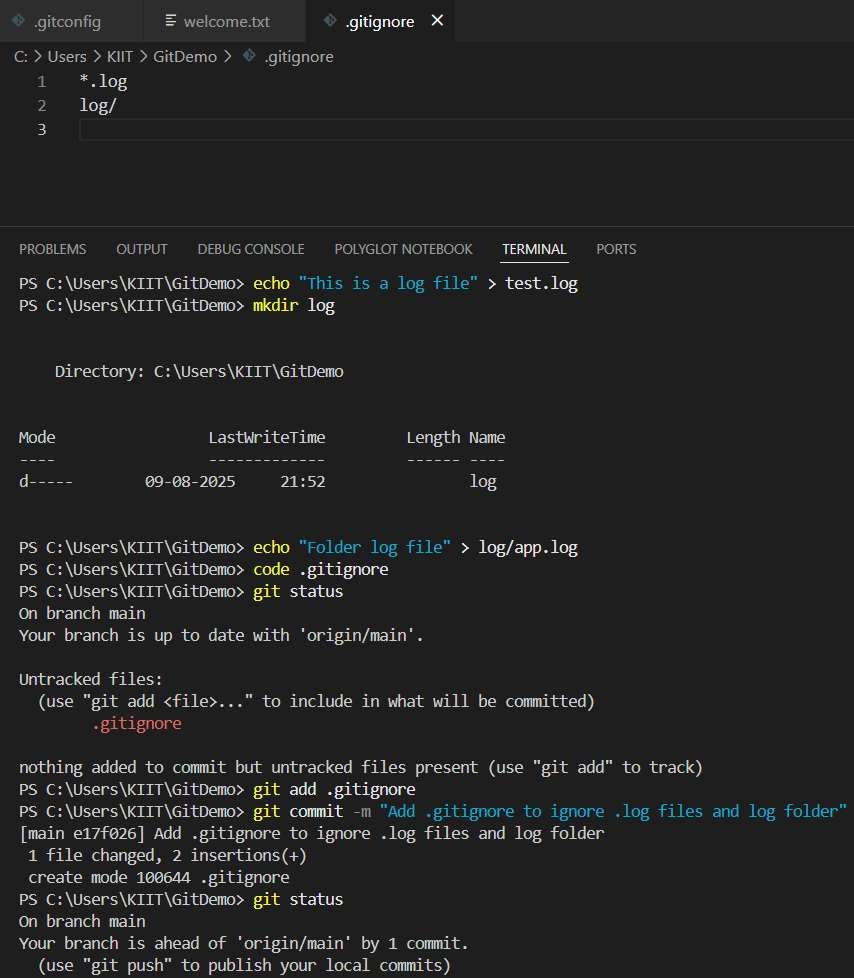
**Question:**

**Explain git ignore. Explain how to ignore unwanted files using git ignore.**

.gitignore is a configuration file in Git used to specify which files or directories should be ignored by version control, meaning Git will not track or commit them. This is useful for excluding temporary files, build artifacts, logs, environment configuration files, or any other files that shouldn’t be shared in the repository. To ignore unwanted files, you simply create a .gitignore file in the project’s root directory and list the file names, directory names, or patterns (like \*.log for all log files) that you want Git to ignore. Once added, Git will skip tracking those files unless they were already committed earlier.

Creating a “.log” file and a log folder in the working directory of Git. Update the .gitignore file in such a way that on committing, these files (.log extensions and log folders) are ignored.

Verifying if the git status reflects the same about working directory, local repository and git repository.



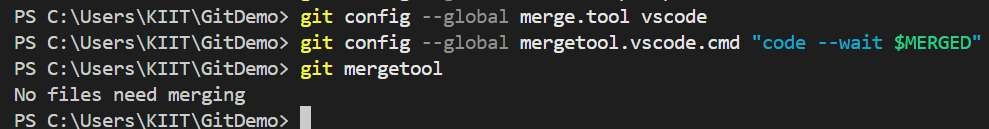
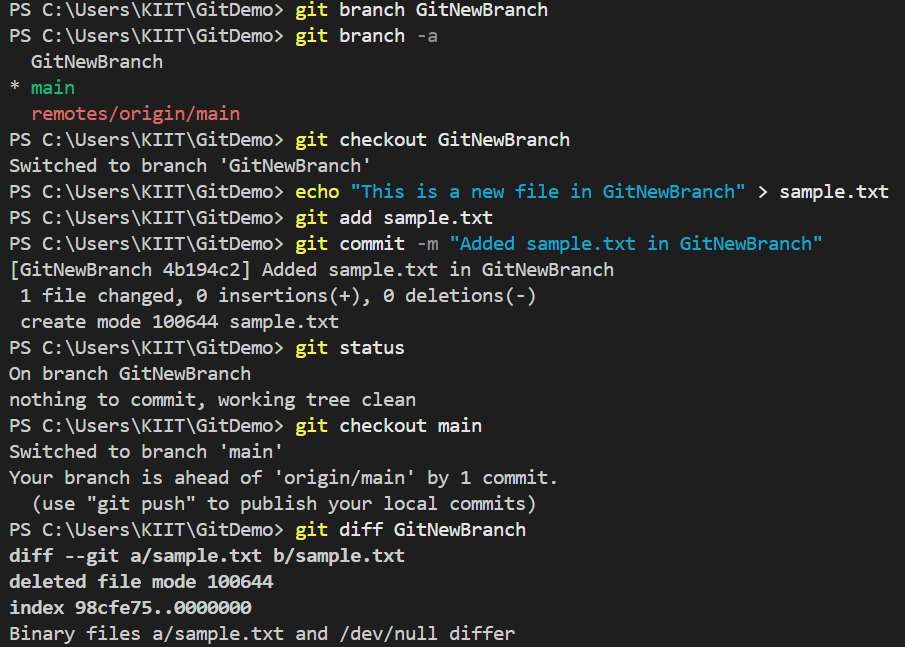
**Question:**

**Explain branching and merging Explain about creating a branch request in GitLab. Explain about creating a merge request in GitLab.**

Branching in Git allows you to create separate lines of development within the same repository, enabling you to work on new features or fixes without affecting the main codebase. Merging is the process of integrating changes from one branch into another, combining their histories. In GitLab, creating a branch request typically means pushing a new branch to the remote repository so it can be reviewed or used for further work. A merge request (MR) in GitLab is a formal way to propose merging changes from one branch into another (usually from a feature branch to main or develop), allowing team members to review code, discuss changes, and ensure quality before integration.

**Constructing a branch, making some changes in the branch, and merging it with master**

**(or trunk).**

****

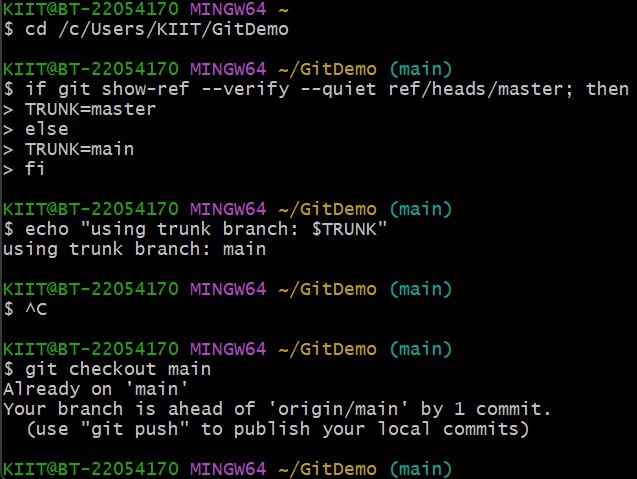
**Question:**

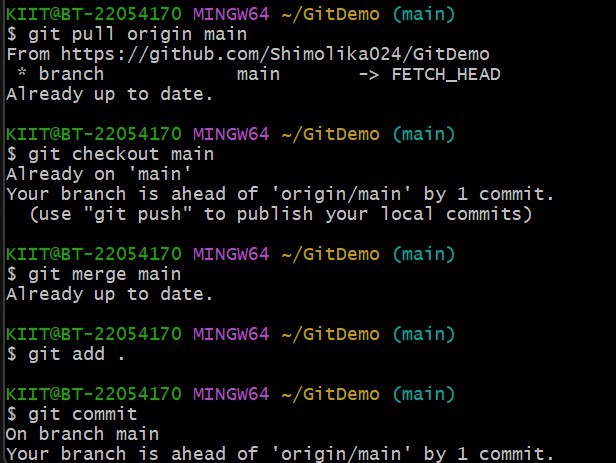
**Explain how to resolve the conflict during merge.**

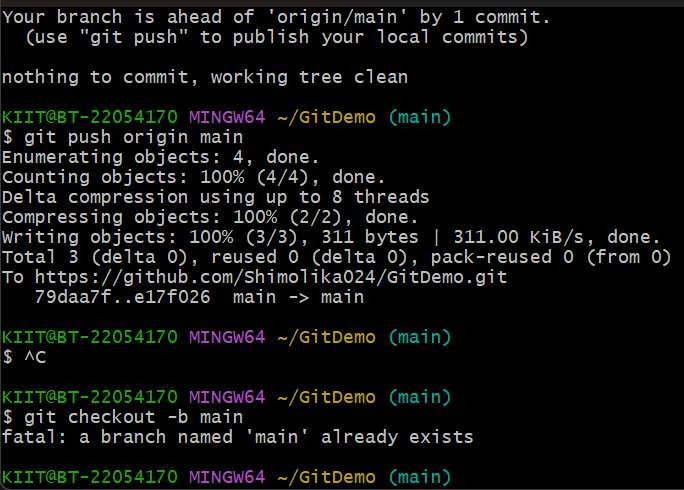
When a merge conflict occurs in Git, it means that changes in two branches affect the same part of a file and Git cannot automatically decide which version to keep. To resolve it, open the conflicted file, look for conflict markers (<<<<<<<, =======, >>>>>>>), and manually edit the content to keep the correct or combined changes. After saving the file, mark the conflict as resolved using git add

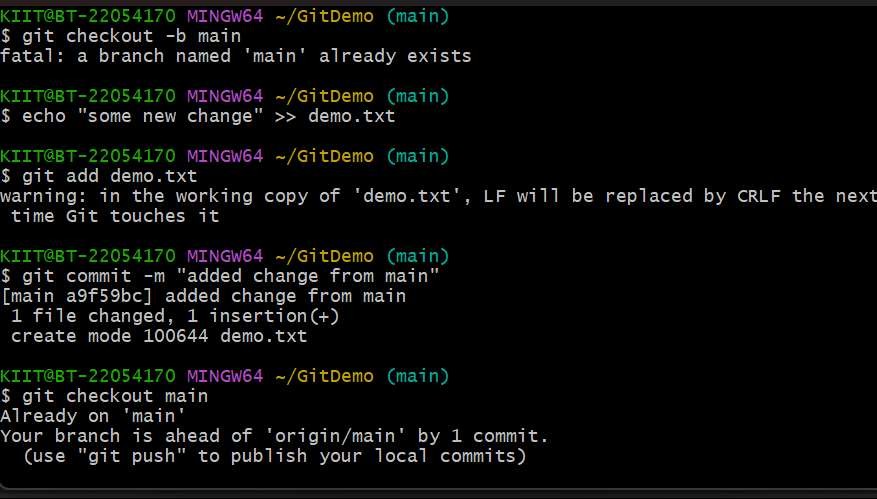
<filename> and then complete the merge with git commit.

**Implementing conflict resolution when multiple users are updating the trunk (or master) in such a way that it results into a conflict with the branch’s modification.**



****



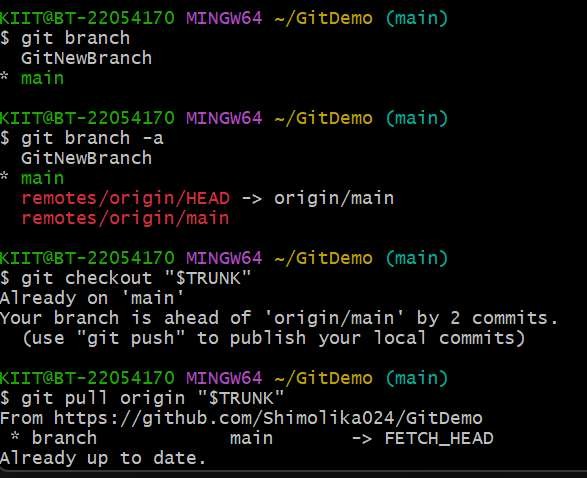
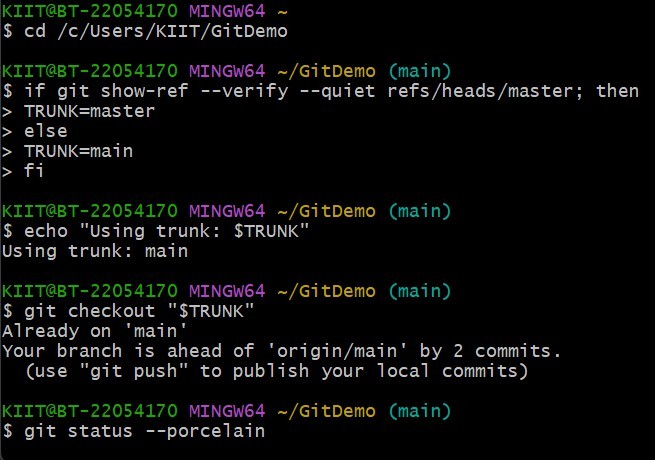
****

**Question:**

**Explain how to clean up and push back to remote Git.**

Cleaning up in Git involves removing unnecessary files, branches, or commits to keep the repository organized and efficient. This can include deleting untracked files with git clean, removing unused local branches with git branch -d, and squashing or rebasing commits to simplify history. Once cleanup is complete, changes are staged (git add), committed (git commit), and then pushed back to the remote repository using git push to ensure the cleaned-up state is reflected for all collaborators.

**Execute steps involving clean up and push back to remote Git.**

****

