FSD Training

Linux

GIT

Java

UI – HTML, CSS, Javascript

Angular & React Introduction

Hibernate

Spring

Spring Boot

Cloud computing technologies

GIT:

It is a version controlling system, which helps people to collaborate their work so that at the end every user will have other users work in their machine as well as in remote machine.

Git provides an Online Remote Repository which users need to clone

List of GIT commands:

git clone: Clones the remote repository in your local machine

git push: Pushes the local repository changes to the remote repository

git pull: Pulls the remote repository changes to the local repository

git commit: Commits the changes done by the user & creates one unique id

git add: Tracks the changes in the working directory for commit

git status: Shows list of tracked & untracked changes in the working directory

Things you need are:

Git account: Create it from your personal ID

Git Bash: Install GIT in your machine so that you get GIT bash, which is a terminal

Git Branch:

Branch is like a pointer which will have the work with series of some commit ids, by default GIT maintains a default branch with a name called master/main

Steps we did

1. Created a Remote Repository
2. Cloned the Remote repository in a folder
3. Navigated to the repository in the local machine
4. Created a file and added some content
5. Used git add command
6. Used git commit command
7. Used git push command
8. Observed that Remote repository got the updated from the local repository
9. Created another folder Developer1 and cloned the remote repository
10. Created another folder Developer2 and cloned the remote repository
11. In Developer1 folder navigated to the local repository & created some files, then entered git add, git commit, git push
12. In Developer2 folder navigated to the local repository & created some files, then entered git add, git commit, git push
13. Remote rejected the push, so in the local repository we entered git pull, here if its fast forward merge then it doesn’t create a commit else it asks us to create a commit for merge

What happens if both the developers try to edit the same file & uploads to the Remote Git

Then Git doesn’t automatically merge instead users need to manually resolve the conflicts and merge the work

Why we should not work with the master branch?

In Real time there could be chance that users may push errors in master and master branch will be automatically merged without any review by any users, hence we must control it by creating a feature branch which is a copy of master branch, so that users can work with the feature branch & push the feature branch so that somebody will review that branch before merge

Feature branch

>> git branch branch-name

The above command creates feature branch

Checkout to branch

When you want to work in a different branch you need to checkout, the work you do in one branch is not updated to another branch automatically

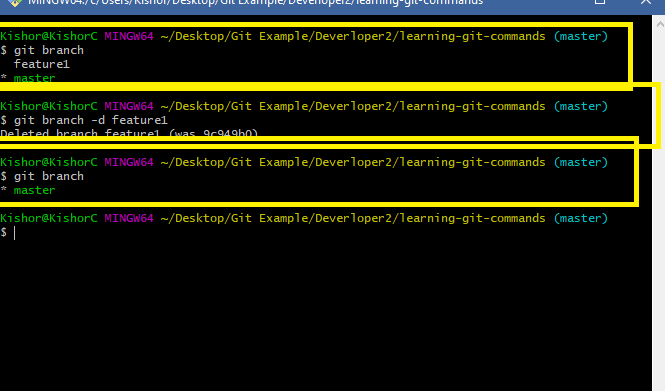
>> git checkout branch-name

Note:

1. Never work in the master branch
2. Always make commits or new changes in the Feature branch
3. Push the feature branch to the remote
4. In Remote the feature branch can be merged or it will be closed without merging

How to delete a branch

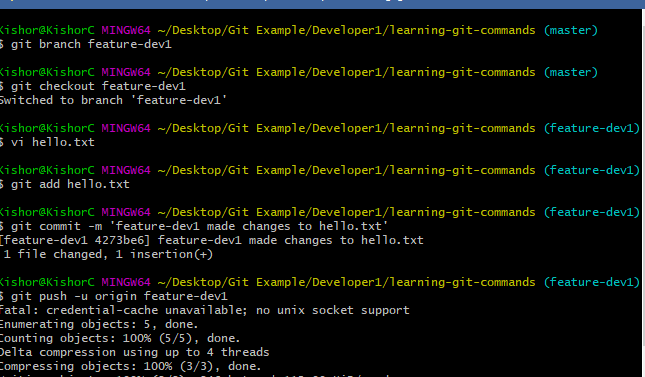
>> git branch -d branch-name



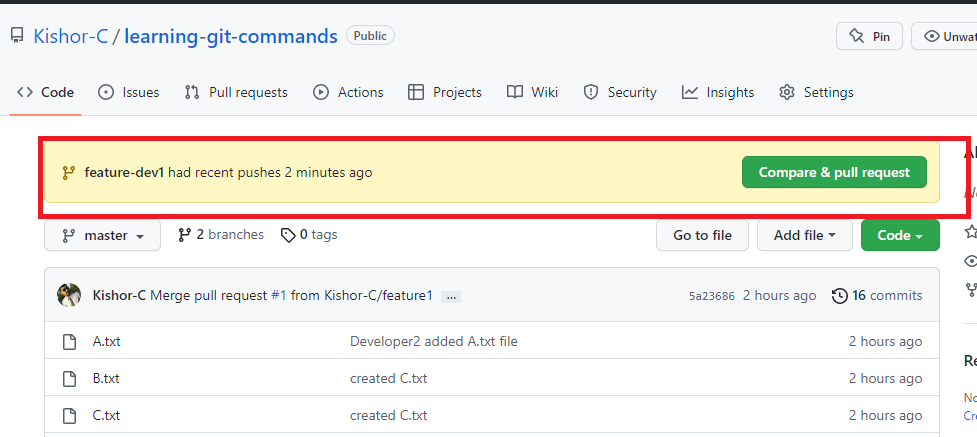
What to do when the remote can’t merge the feature branch

In the remote whenever there’s a Merge conflict you can close the pull request without merging & delete the branch, so that the user who pushed the branch should able to resolve the conflict and again push the feature branch

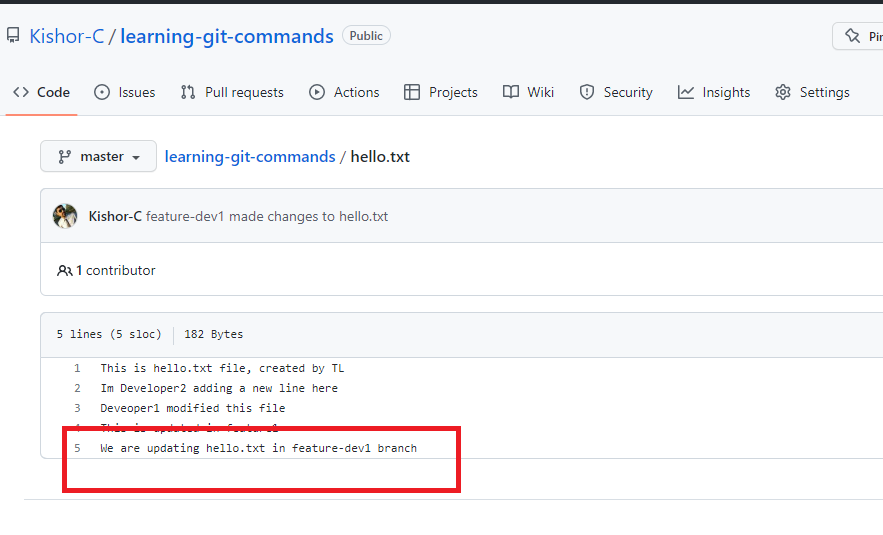
Let us create a new branch in the developer1 terminal & edit hello.txt with some content & push that new feature branch



We must see the same changes in the Remote, but we get a pull request in the Remote

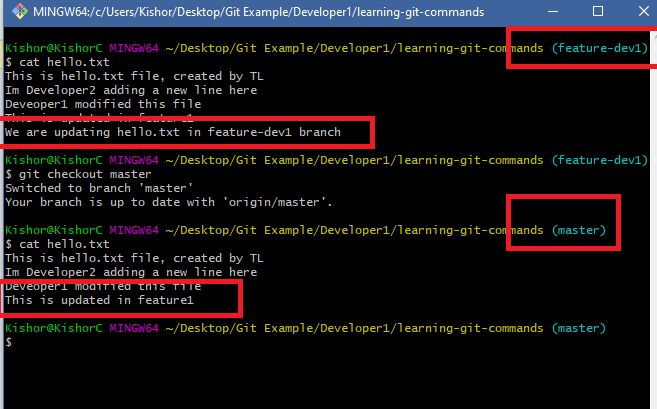


You need to click on Compare & pull request & check for merge option, if possible you can merge else you can close pull request



The highlighted part is the change done by developer1 in the feature-dev1 branch.

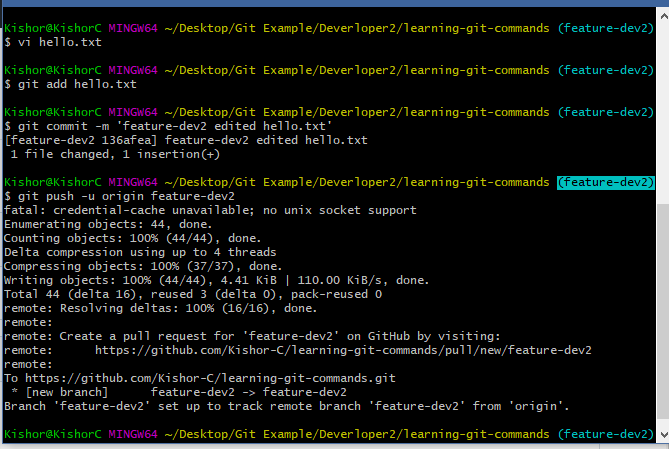
Note: Even in Developer1 terminal we have a master branch which is also not having this change



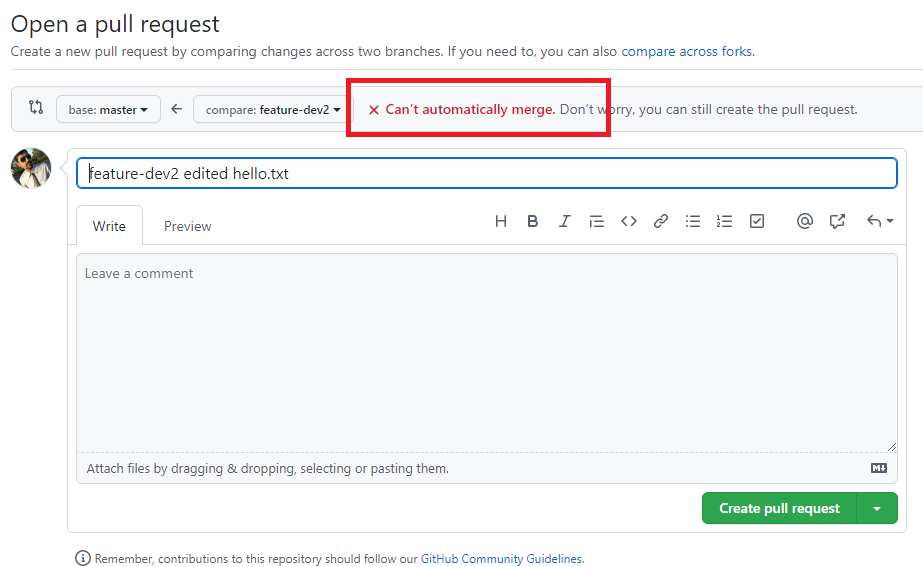
There are two ways in Developer1 Local repository to get master branch up to date changes of feature branch

1. use git merge in local repository in the master branch(git merge feature-dev1): This is not recommended as it gets only local repository changes, but it doesn’t get any remote repository changes
2. use git pull in local repository so that it pulls all the updates from remote repository: This is appropriate because it always gets changes done by other users from the remote

What happens if Developer2 terminal make changes to the ‘hello.txt’ without updating the local master



Here the Developer2 pushed feature-dev2 hence there was no rejection at the remote, however when the feature branch is tried to merge with remote master you may get conflict because the feature branch is not having other user work.



Here we can’t merge hence we can create pull request & close it so that the Developer2 will manually resolve, he must follow below steps

1. Pull the remote master to local master
2. Checkout to feature branch
3. Merge local master with feature branch if conflict resolve

Steps:

>> git checkout master

>> git pull

>> git checkout feature-dev2

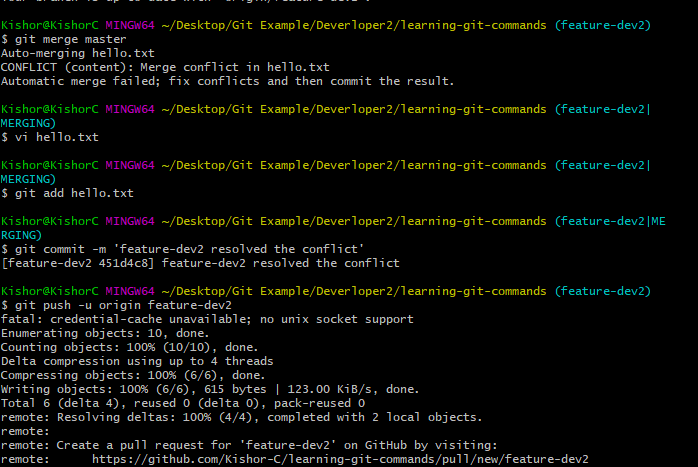
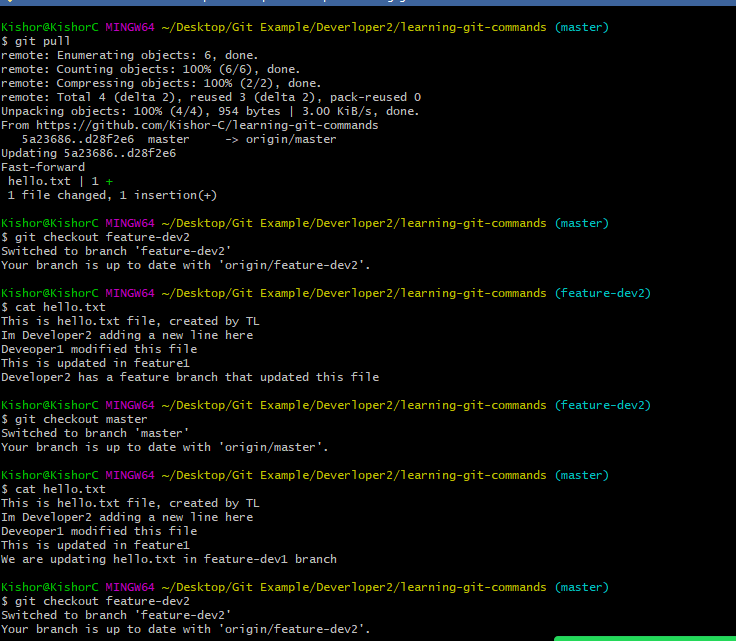
>> git merge master

# resolve the conflict when you get auto-merge failed

>> git add hello.txt

>> git commit -m ‘some message’

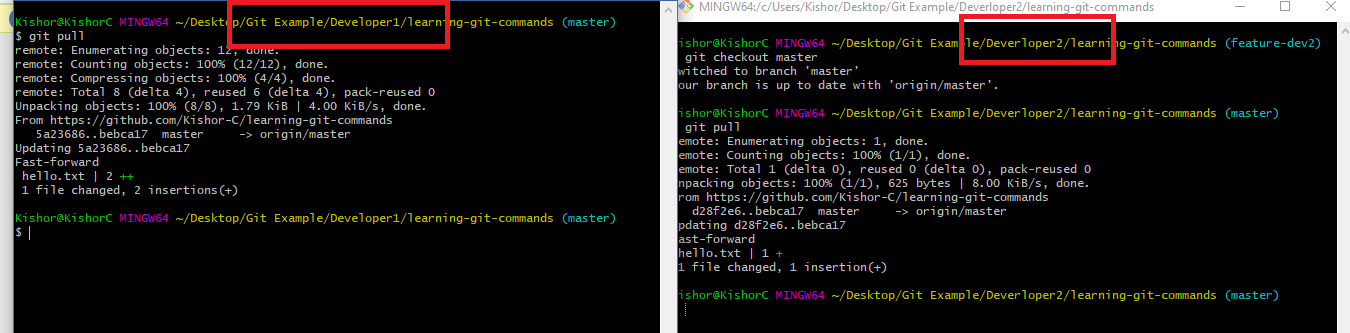
>> git push -u origin feature-dev2



Now in Remote there wouldn’t be any conflict to merge this feature branch

Final step:

Pull remote in local master in both the developers terminal to keep up to date.



Summary of Git steps to be followed when working in the project

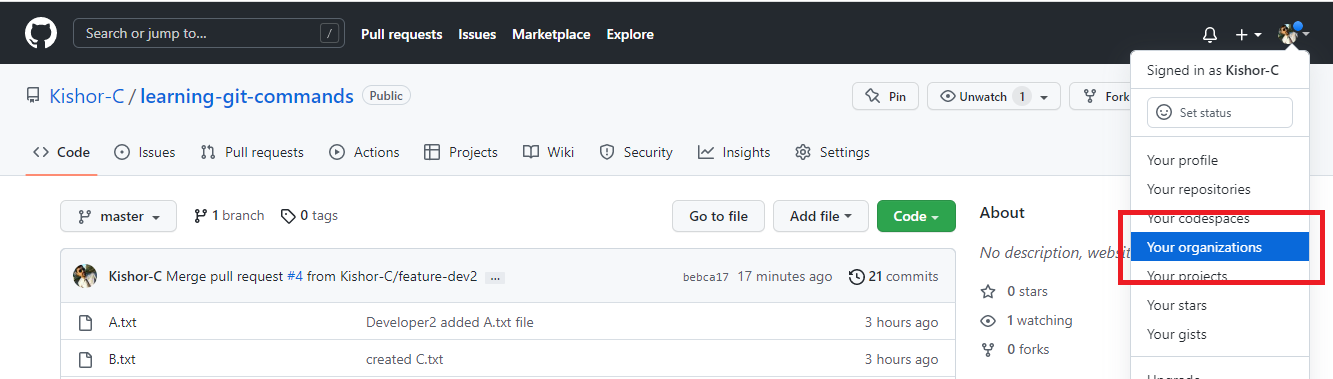
1. Git clone the remote repository if you don’t have local repository
2. Always make sure that you don’t work in master/main branch
3. Create feature branch to make any changes
4. Push the feature branch to the remote
5. If Pull request failed to merge then update the local master with remote master first using ‘git pull’
6. Once git pull updates the local master, checkout to feature branch and merge local master branch with feature branch using ‘git merge master’
7. You may get conflict while merging hence you can resolve it by editing the file having conflict.
8. Once conflict resolved commit & push the feature branch to the Remote repository for merge, but before that you can use git pull to ensure you have up to date changes

Git Organization:

It is a feature in the GIT which makes people to work in a team to collaborate their work, you need to add members here so that only members in the organization can work with the repository

Here someone must add the members by mentioning the mail-ids of their Git account so that each member will accept the invitation over the mail

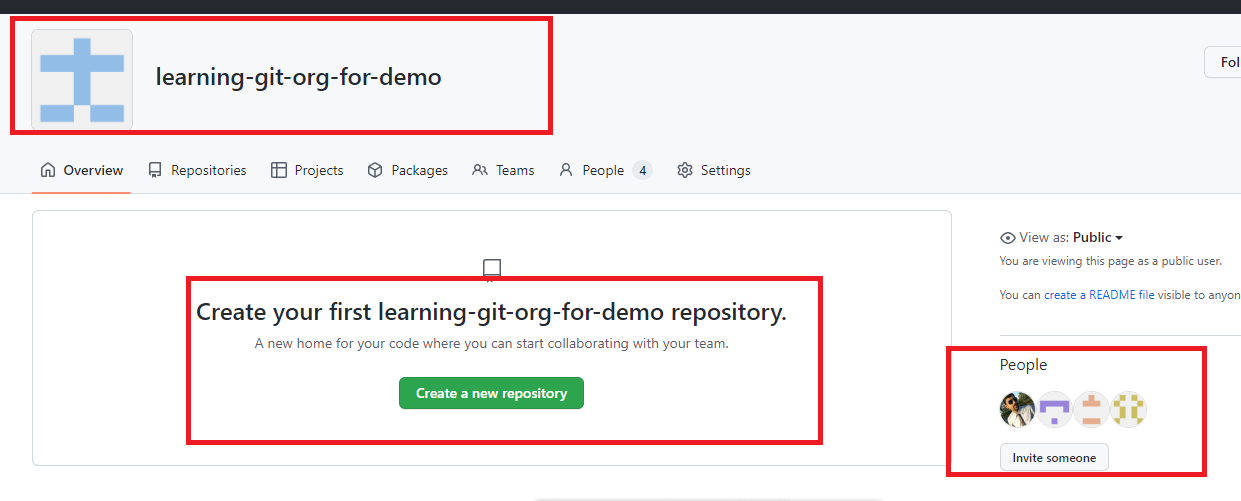
How to create Organization



Steps

1. You can create a new organization & add members.
2. You can pick Free plan
3. Enter organization name & your git mail id to create organization
4. Add members by entering mail ids
5. Members need to accept invitation
6. You can make all the members owner to enable them do push/pull task

You must able to see members in the organization



Activity

1. Try out all the GIT commands taught in the Session
2. Create an organization (any one person in the team), add the members, one person in the team must create the Repository & all the members must clone it
3. Each member can create a text file with their name in the feature branch & push that feature branch to the organization repository
4. Someone in the team must take care of merging the pull request