

Experiment -1.2

Install Git and creating repository.

Student Name: Shubham Kumar

Branch: CSE(DevOps)

Semester: 4th

Subject Name: Git and GitHub

UID: 22BDO10033

Section/Group: 22BCD-1/A

Date of Performance: 24/01/24

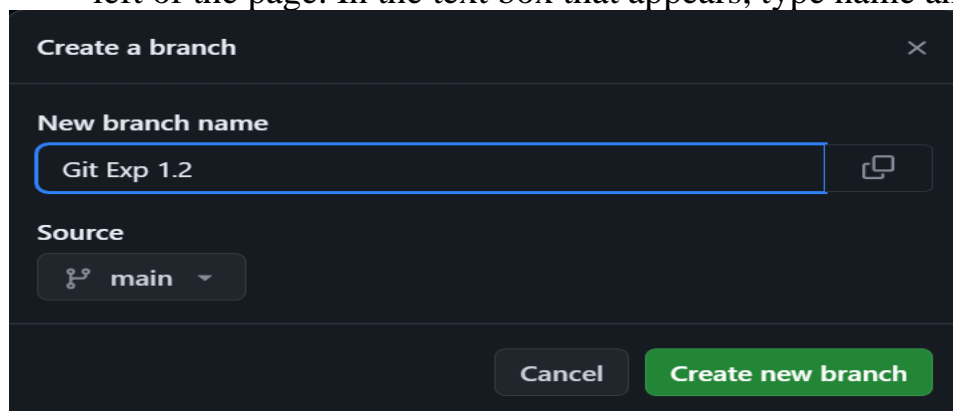
Subject Code: 22CHS-293

1. Aim/Overview of the practical: - Creating branches with GitHub and merging with the main branch.

2. Task to be done: - Create branch of a repository and make changes and merge the file into main branch.

3. Steps for experiment/practical:

1. Create a new branch: - Open GitHub repository click on the “Branch: main” button top left of the page. In the text box that appears, type name and press enter.



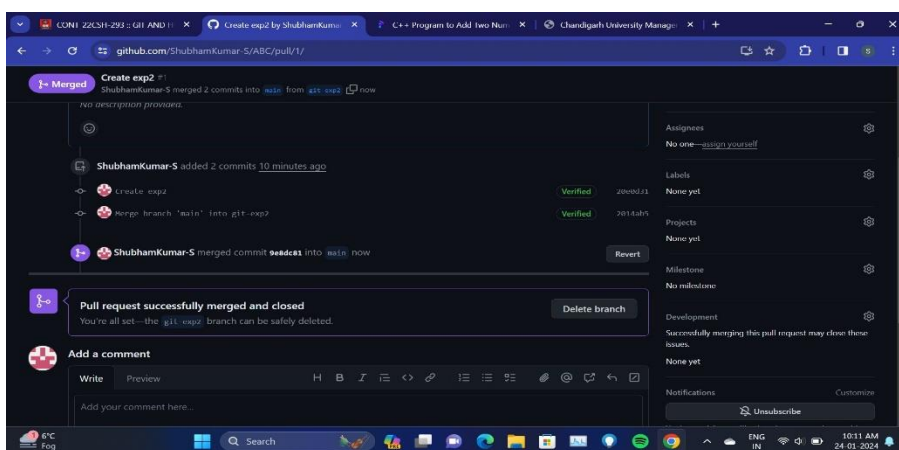
2. Edit and Commit Changes:

Make changes to your code or add new files, on down “Changes” section, Enter a commit message in the “Commit changes” section. Click on commit change.

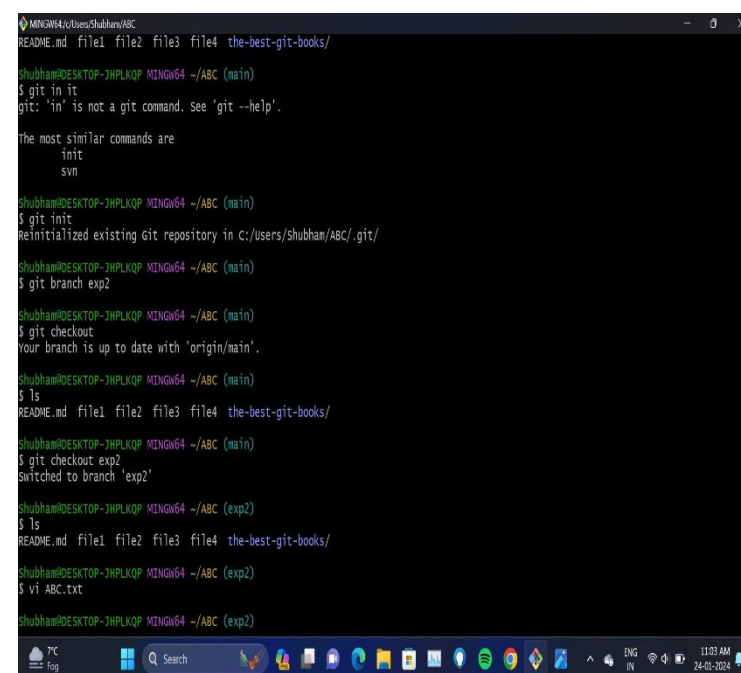
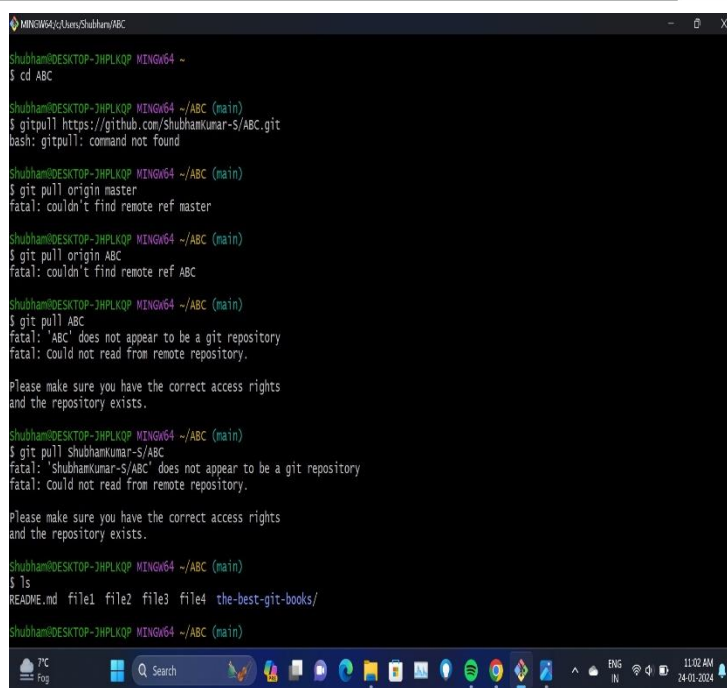
3. Create a pull request: - After committed changes, GitHub will display a message with a “Compare & pull request” button. Click on that button.

4. Merge the pull request: - After the pull request is reviewed and approved, you can merge it. Click on the "Merge pull request" button, confirm the merge.

5. Confirm the Merge: - Once the pull request is merged.



6. Delete the Branch (optional): - After merging, GitHub will give you the option to delete the branch. Choose to delete the branch if you no longer need it.



```
MINGW64/c/Users/Shubham/ABC
-f, --[no-]force          allow adding otherwise ignored files
-u, --[no-]update         update tracked files
--[no-]renormalize        renormalize EOL of tracked files (implies -u)
-N, --[no-]intent-to-add  record only the fact that the path will be added later
-A, --[no-]all            add changes from all tracked and untracked files
--[no-]ignore-removal     ignore paths removed in the working tree (same as --no-all)
--[no-]refresh            don't add, only refresh the index
--[no-]ignore-errors      just skip files which cannot be added because of errors
--[no-]ignore-missing     check if - even missing - files are ignored in dry run
--[no-]sparse             allow updating entries outside of the sparse-checkout cone
--[no-]chmod (+|-)x       override the executable bit of the listed files
--[no-]pathspecc-from-file read pathspec from file
--[no-]pathspecc-file-nul with --pathspecc-from-file, pathspecc elements are separated with NUL character

Shubham@DESKTOP-JHPLKQP MINGW64 ~/ABC (exp2)
$ git add ABC.txt

Shubham@DESKTOP-JHPLKQP MINGW64 ~/ABC (exp2)
$ git commit -m "hello"
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'Shubham@DESKTOP-JHPLKQP.(none)')
```

```
MINGW64/c/Users/Shubham/ABC
core.fsccache=true
core.symlinks=false
core.fsmonitor=true
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
core.editor="C:\Users\Shubham\AppData\Local\Programs\Microsoft VS Code\bin\code" --wait
user.name=sShubhamkumar63@gmail.com
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.origin.url=https://github.com/ShubhamKumar-S/ABC.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.main.remote=origin
branch.main.merge=refs/heads/main

Shubham@DESKTOP-JHPLKQP MINGW64 ~/ABC (exp2)
$ git push origin
fatal: The current branch exp2 has no upstream branch.
To push the current branch and set the remote as upstream, use

  git push --set-upstream origin exp2

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

Shubham@DESKTOP-JHPLKQP MINGW64 ~/ABC (exp2)
$ git checkout ABC
error: pathspecc 'ABC' did not match any file(s) known to git

Shubham@DESKTOP-JHPLKQP MINGW64 ~/ABC (exp2)
$ git check|
```

Learning outcomes (What I have learnt):

1. Understanding Git Bash Commands
2. Use of GitHub Repository URL
3. Creating a Local Copy using Git bash.
4. Navigating Directories.
5. Connecting Local and Remote Repositories

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			