

NAME: SANA TARIQ

CLASS: BSE-5B

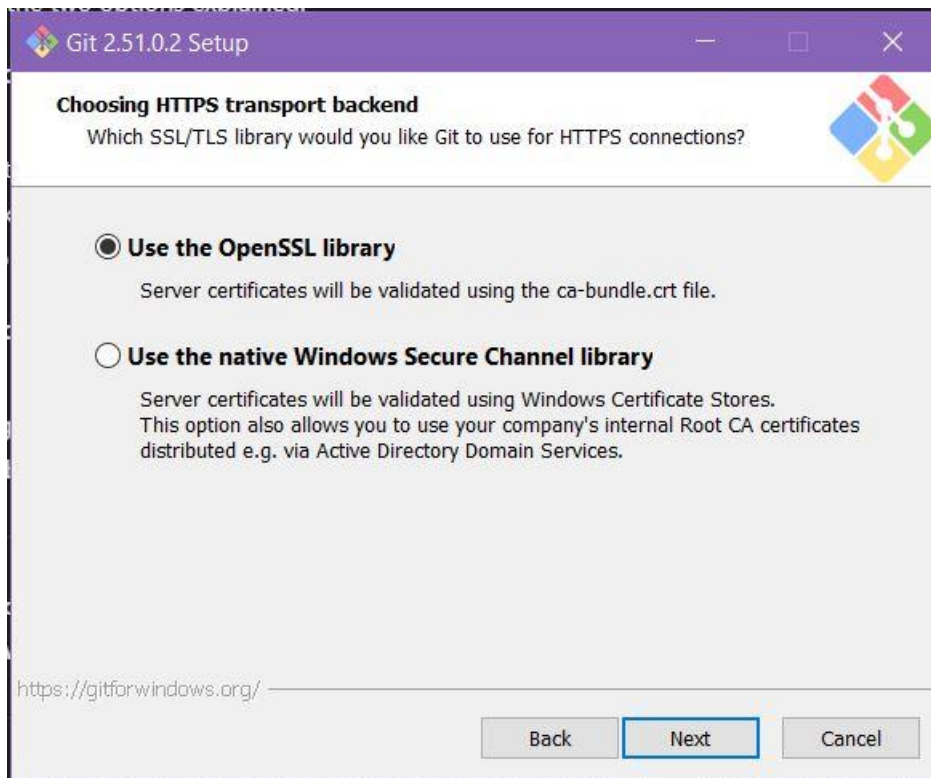
REG.NO: 058

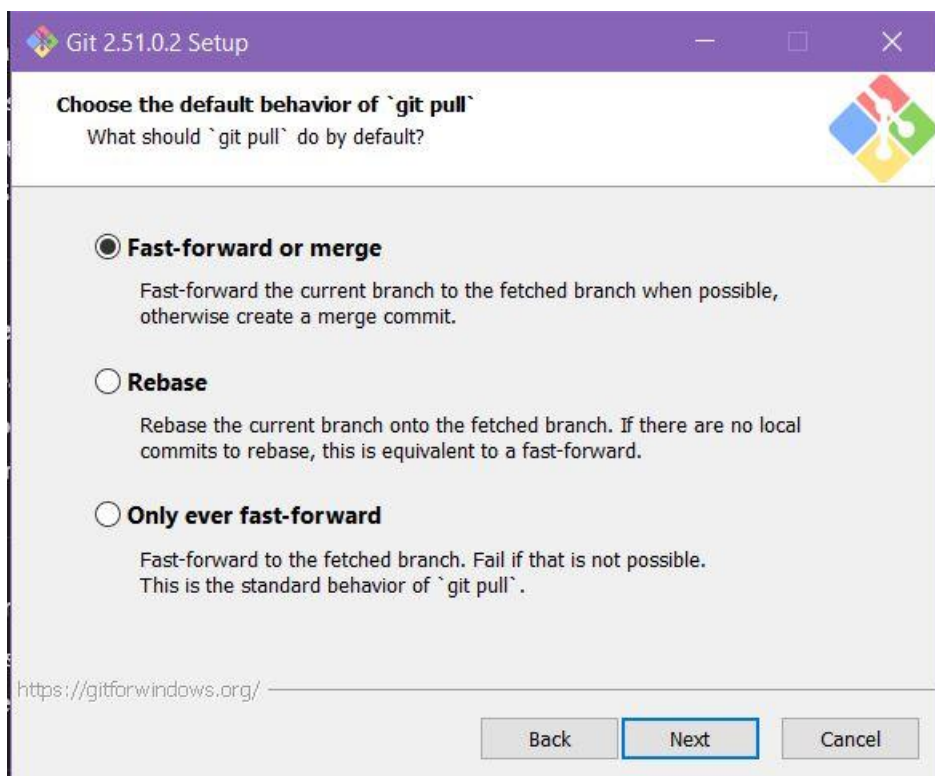
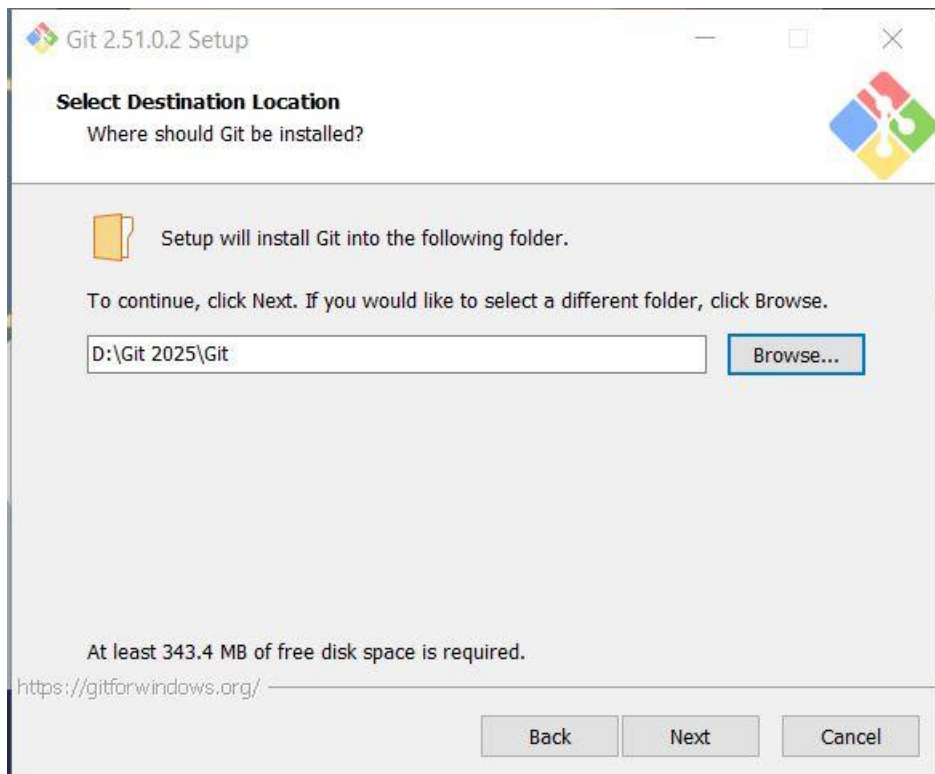
SUBJECT: CLOUD COMPUTING

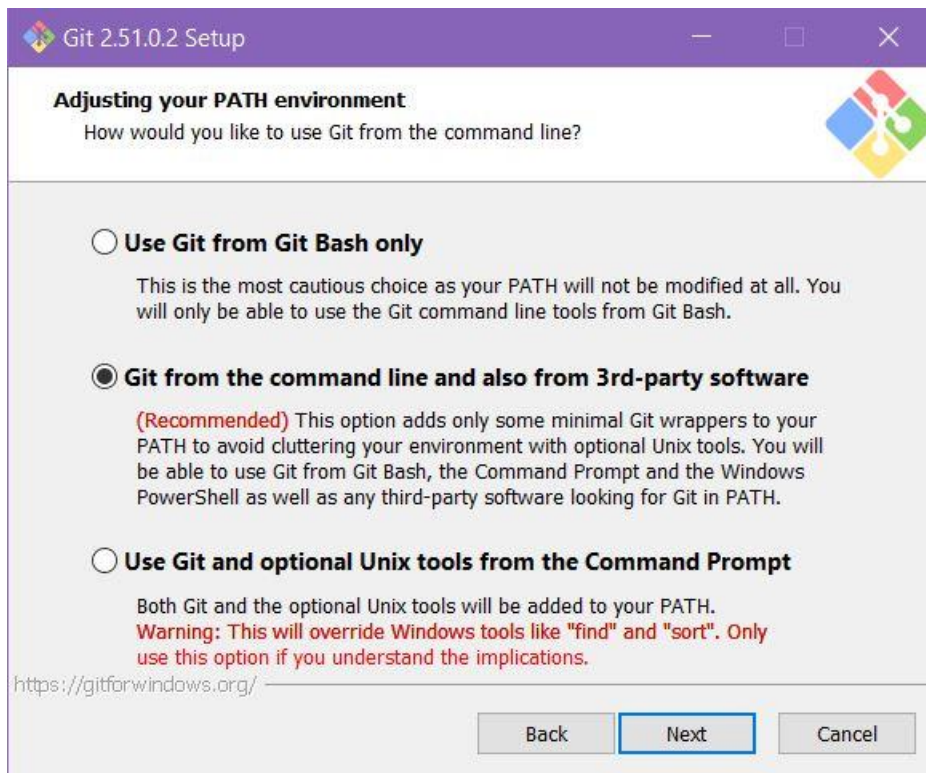
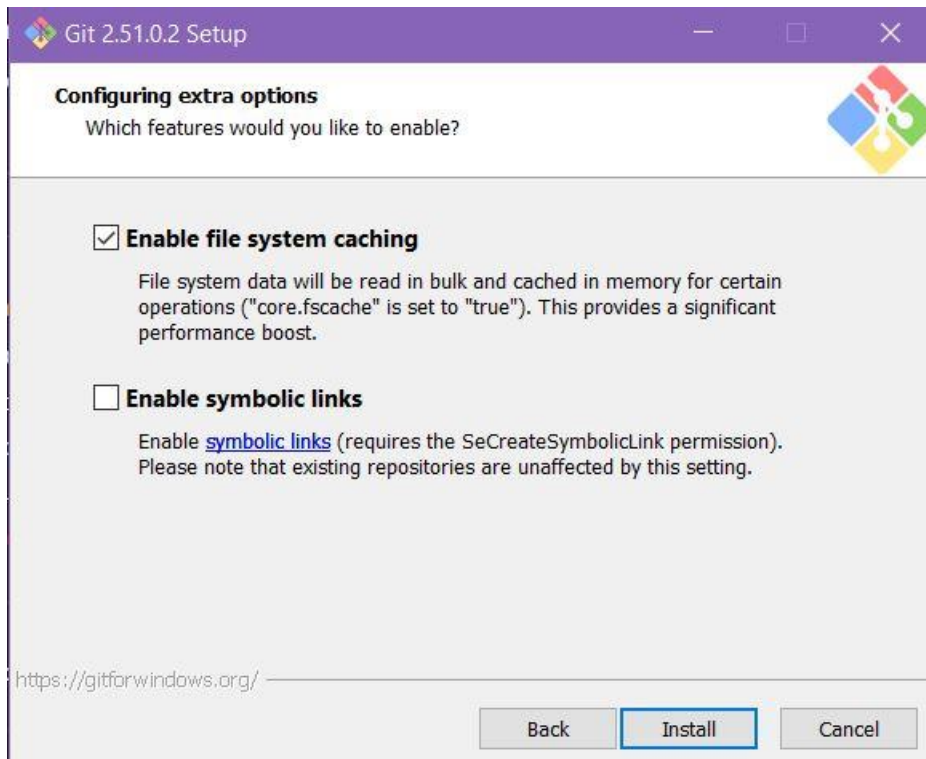
LAB#02:

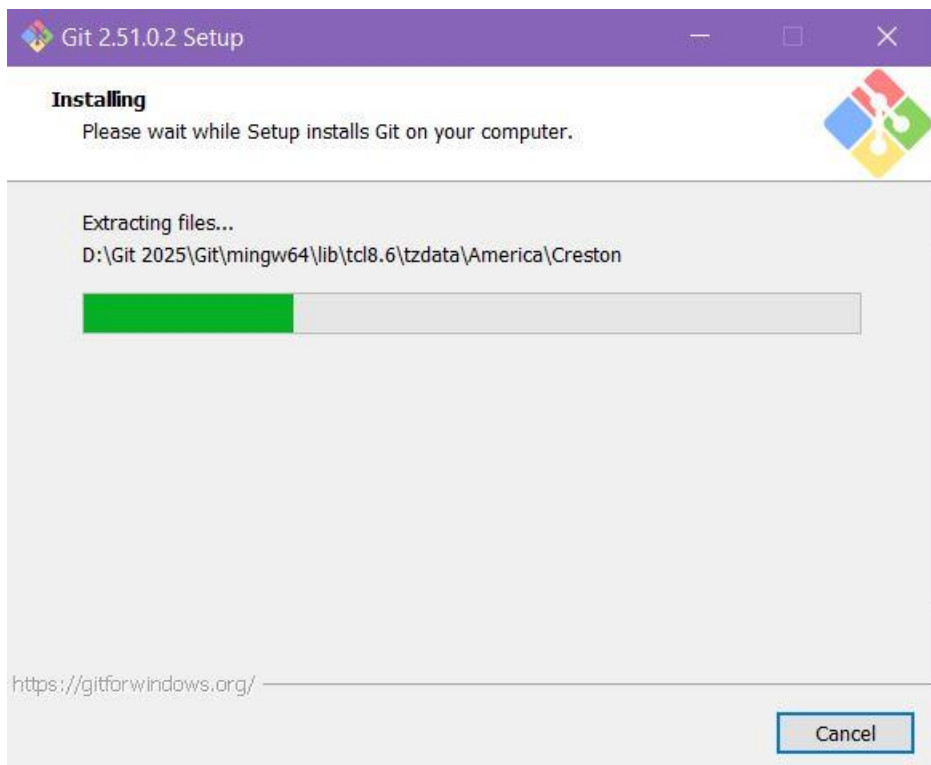
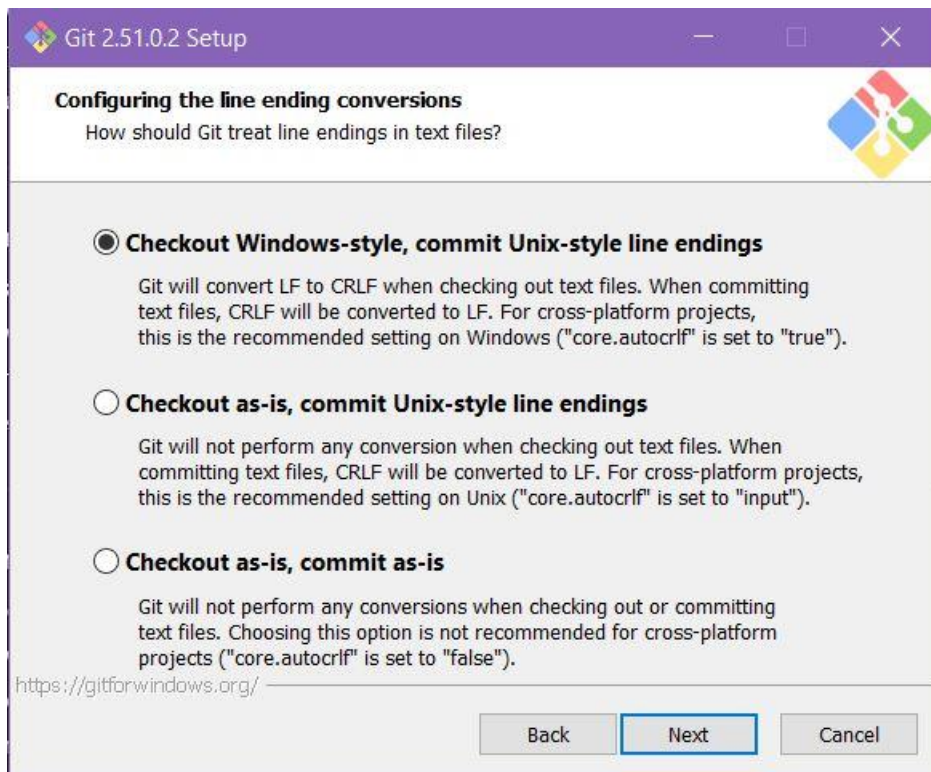
TASK#01:

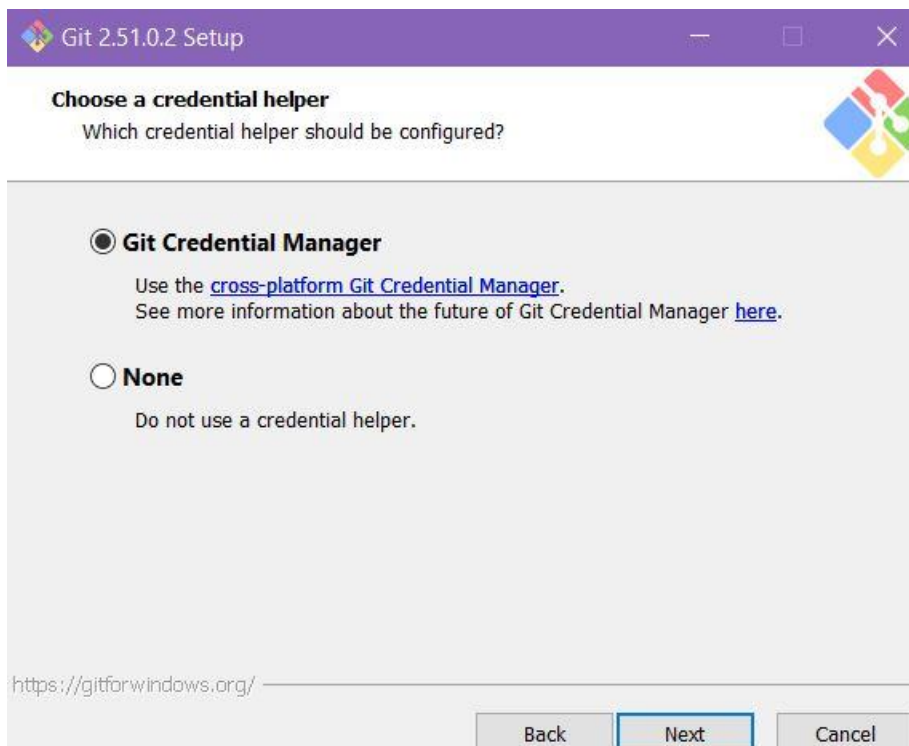
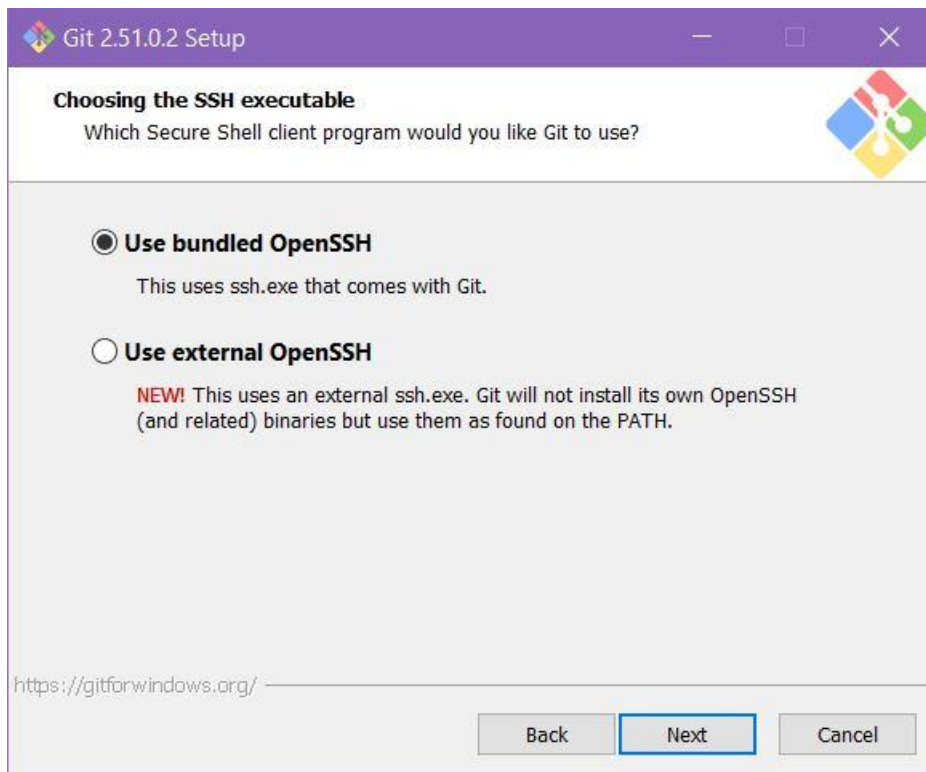
Create a new private repository named Lab2 on GitHub.

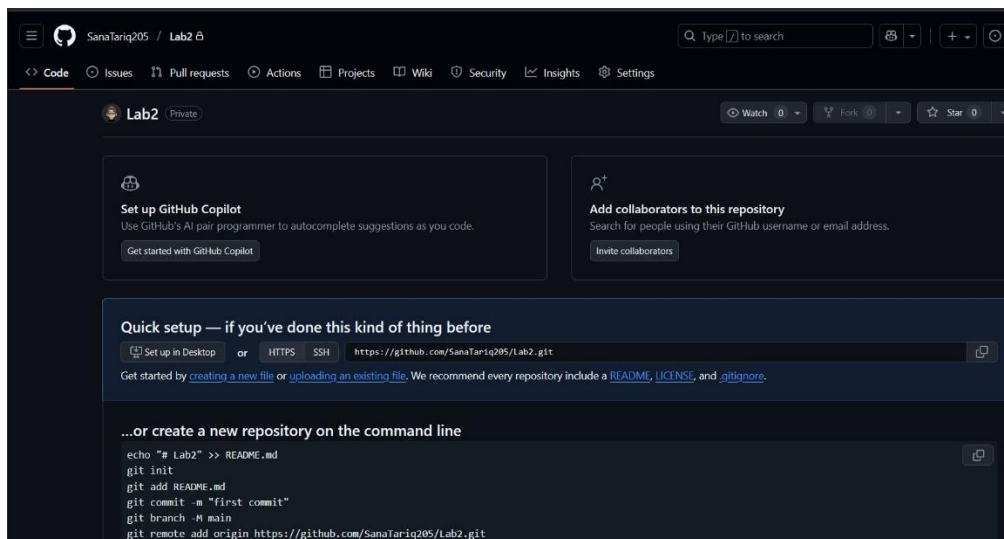
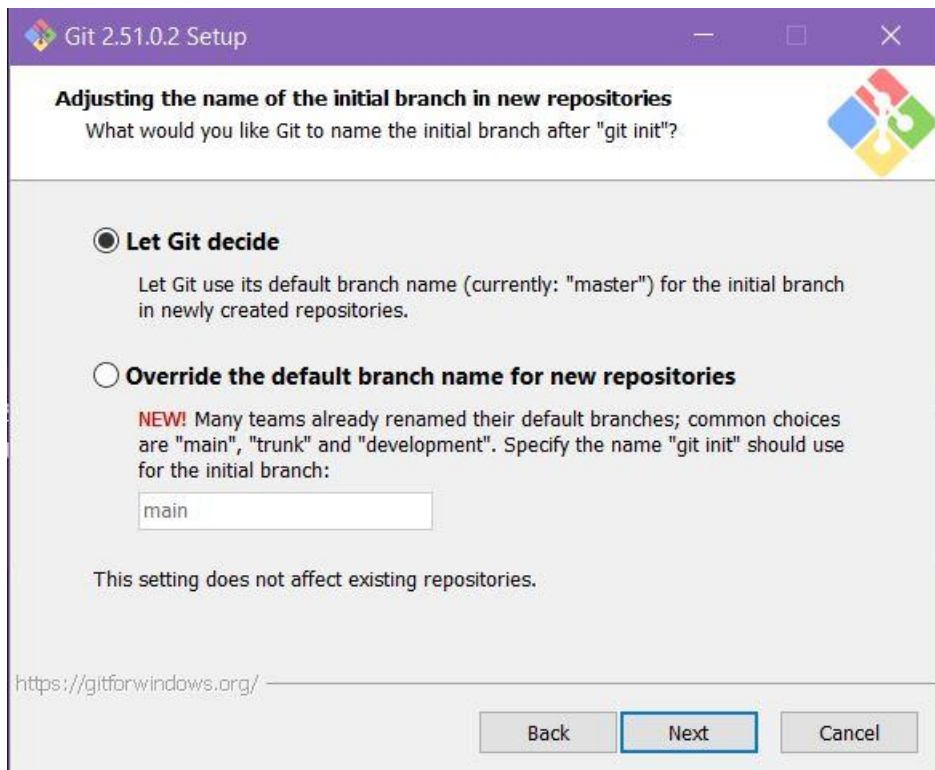












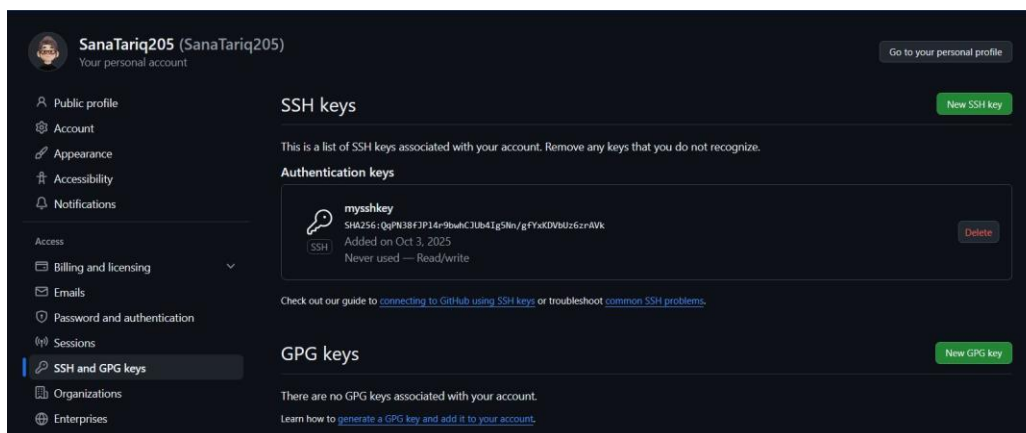
TASK#02:

1. Generate a new SSH key using PowerShell: `ssh-keygen -t ed25519 -C "your_email@example.com"`
2. Add your SSH public key to GitHub (Settings > SSH and GPG keys).
3. Clone your Lab2 repo using SSH. `git clone git@github.com:Lab2.git`


```
MINGW64/c/Users/Sana

Sana@DESKTOP-00ECUUI MINGW64 ~
$ ssh-keygen -t ed25519 -C "23-22411-058@se.fjwu.edu.pk"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/Sana/.ssh/id_ed25519):
Enter passphrase for "/c/Users/Sana/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Sana/.ssh/id_ed25519
Your public key has been saved in /c/Users/Sana/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:QqPN38fJP14r9bwhCJU4Ig5Nn/gfYxKDVbUz6zrAVk 23-22411-058@se.fjwu.edu.pk
The key's randomart image is:
+--[ED25519 256]--+
  .+.
  o + ...
  *o= . E+
  .=*.* * o+
  . ++SO +.
  .OO.++...
  .. .O*O.+
  ++...+
  .OOO...
+-----[SHA256]-----

Sana@DESKTOP-00ECUUI MINGW64 ~
```



```
Sana@DESKTOP-00ECUUI MINGW64 ~
$ git clone git@github.com:SanaTariq205/Lab2.git
Cloning into 'Lab2'...
The authenticity of host 'github.com (20.207.73.82)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
warning: You appear to have cloned an empty repository.
```

TASK#03:

1. Set up your Git identity (this ensures all commits are linked to you): `git config --global user.name "Your Name"` `git config --global user.email "your_email@example.com"`
2. Verify your configuration: `git config --list`

```
Sana@DESKTOP-00ECUUI MINGW64 ~  
$ git config --global user.name "Sana Tariq"  
  
Sana@DESKTOP-00ECUUI MINGW64 ~  
$ git config --global user.email "23-22411-058@se.fjwu.edu.pk"
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~  
$ git config --list  
diff.astextplain.textconv=astextplain  
filter.lfs.clean=git-lfs clean -- %f  
filter.lfs.smudge=git-lfs smudge -- %f  
filter.lfs.process=git-lfs filter-process  
filter.lfs.required=true  
http.sslbackend=schannel  
core.autocrlf=true  
core.fscache=true  
core.symlinks=false  
core.editor="C:\\Program Files\\Notepad++\\notepad++.exe" -multiInst -notabbar -nosession -noPlugin  
pull.rebase=false  
init.defaultbranch=main  
user.name=Sana Tariq  
user.email=23-22411-058@se.fjwu.edu.pk
```

TASK#04:

Navigate into your cloned repository folder. Show hidden files and locate the .git directory. Explore what's inside using: `ls -a .git`

```
Sana@DESKTOP-00ECUUI MINGW64 ~  
$ cd Lab2  
  
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)  
$ ls -a  
./ ../ .git/  
  
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)  
$ ls -a .git  
./ ../ config description HEAD hooks/ info/ objects/ refs/  
  
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)  
$
```

TASK#05:

1. Delete the existing .git folder from your cloned repo using Git Bash: `rm -rf .git`
2. Re-initialize the local git repository: `git init`
3. Add a file named README.md and commit it: `echo "# Lab2 Git Practice" > README.md`
`git add README.md`
`git commit -m "Initial commit"`
4. Connect your local repo to GitHub and push: `git remote add origin git@github.com:/Lab2.git`
`git push -u origin main`

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)  
$ rm -rf .git
```



```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2
$ git init
Initialized empty Git repository in C:/Users/Sana/Lab2/.git/
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ echo "# Lab2 Git Practice" > README.md

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git add README.md
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git commit -m "Initial commit"
[main (root-commit) 97f9bad] Initial commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git remote add origin git@github.com:SanaTariq205/Lab2.git

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git branch -M main

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 242 bytes | 242.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:SanaTariq205/Lab2.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

TASK#06:

1. Create a new file notes.txt and write a note. Check status: git status

Stage and commit: git add notes.txt git commit -m "Add notes.txt"

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ echo "This is my first note." > notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    notes.txt

nothing added to commit but untracked files present (use "git add" to track)
```

```

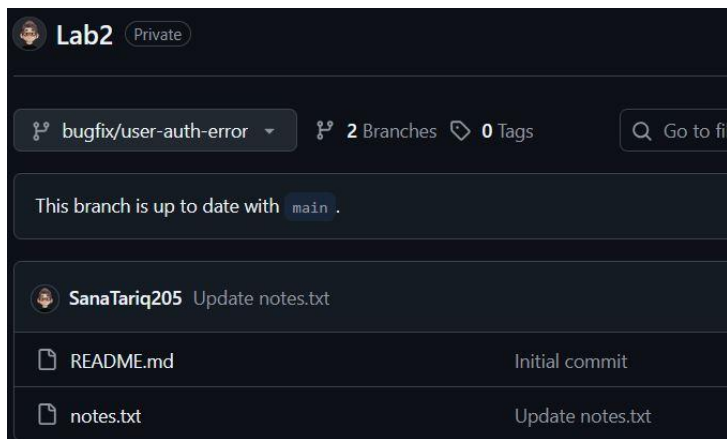
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git add notes.txt
warning: in the working copy of 'notes.txt', LF will be replaced by CRLF the next time Git touches it

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git commit -m "Add notes.txt"
[main b8d3275] Add notes.txt
1 file changed, 1 insertion(+)
create mode 100644 notes.txt

```

TASK#07:

On GitHub (web interface), create a branch named bugfix/user-auth-error. Pull the branch to your local repository to sync. git pull origin bugfix/user-auth-error



```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git pull origin bugfix/user-auth-error
From github.com:SanaTariq205/Lab2
* branch          bugfix/user-auth-error -> FETCH_HEAD
* [new branch]     bugfix/user-auth-error -> origin/bugfix/user-auth-error
Already up to date.

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git branch -a
* main
  remotes/origin/bugfix/user-auth-error
  remotes/origin/main

```

TASK#08:

1. Create a branch named feature/db-connection using Git Bash: git checkout -b feature/db-connection 2. Push the branch to the remote repository: git push origin feature/db-connection

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git checkout -b feature/db-connection
Switched to a new branch 'feature/db-connection'

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature/db-connection)
$

```

TASK#09:

1. Create and switch to a branch feature-1:

git checkout -b feature-1

2. Modify main.py (add a function) and commit. git add main.py git commit -m "Add new function to main.py"

Switch back to main and merge: git checkout main git merge feature-1

4. Push all branches: git push origin main git push origin feature-1

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature/db-connection)
$ git checkout -b feature-1
Switched to a new branch 'feature-1'

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-1)
$
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-1)
$ echo "def hello(): print('Hello from feature-1')" > main.py

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-1)
$ git add main.py
warning: in the working copy of 'main.py', LF will be replaced by CRLF the next time Git touches it

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-1)
$ git commit -m "Add new function to main.py"
[feature-1 45dc317] Add new function to main.py
1 file changed, 1 insertion(+)
create mode 100644 main.py
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-1)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

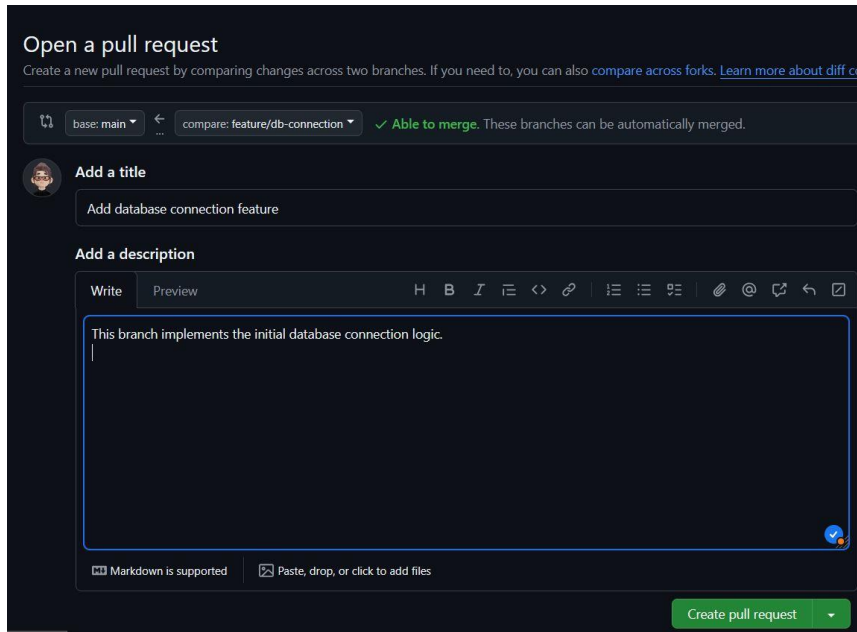
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git merge feature-1
Updating 84166f7..45dc317
Fast-forward
 main.py | 1 +
1 file changed, 1 insertion(+)
create mode 100644 main.py
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 371 bytes | 123.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:SanaTariq205/Lab2.git
84166f7..45dc317 main -> main

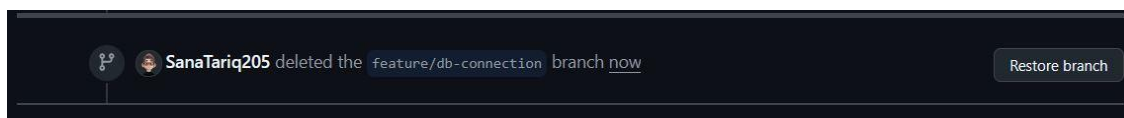
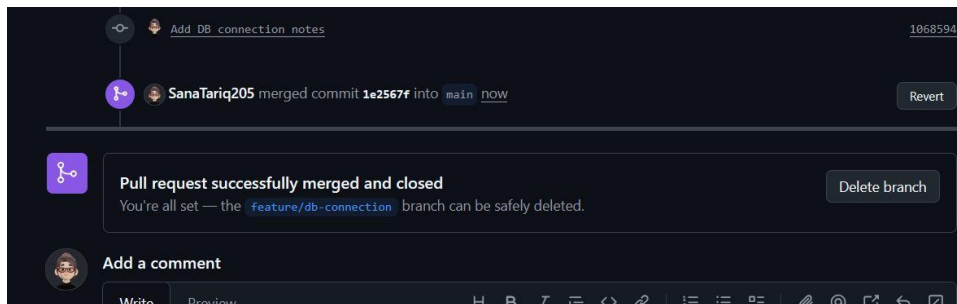
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git push origin feature-1
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-1' on GitHub by visiting:
remote:   https://github.com/SanaTariq205/Lab2/pull/new/feature-1
remote:
To github.com:SanaTariq205/Lab2.git
* [new branch]   feature-1 -> feature-1
```

TASK#10:

On GitHub, create a Pull Request from the branch `feature/db-connection` to `main`. Review the Pull Request and merge it using the GitHub GUI. After merging, delete the `feature/db-connection` branch using the GitHub GUI.



The screenshot shows the GitHub 'Open a pull request' interface. At the top, it says 'Open a pull request' and 'Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff comparison](#)'. Below this, there are two dropdown menus: 'base: main' and 'compare: feature/db-connection'. A green checkmark indicates 'Able to merge. These branches can be automatically merged.' The 'Add a title' section has a text input field with the placeholder 'Add database connection feature'. The 'Add a description' section has a rich text editor with a 'Write' tab selected. The description text reads: 'This branch implements the initial database connection logic.' At the bottom right, there is a green 'Create pull request' button.



TASK#11:

Create the following branches to simulate a professional branching strategy: `develop` staging `feature/*` `bugfix/*` Example workflow: Developers work on `feature/*` and `bugfix/*` branches. Merge into `develop` after completion. `develop` is merged into `staging` for testing. Finally, `staging` is merged into `main` (production).

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (staging)
$ git branch -a
develop
feature-1
feature/db-connection
feature/user-login
main
* staging
remotes/origin/HEAD -> origin/main
remotes/origin/bugfix/user-auth-error
remotes/origin/develop
remotes/origin/feature-1
remotes/origin/feature/db-connection
remotes/origin/main
remotes/origin/staging

```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (staging)
$ git log --oneline --graph --all --decorate
* 33ab573 (origin/main, origin/HEAD, main) Merge pull request #4 from SanaTariq205/staging
|
| * 6e09887 (HEAD -> staging, origin/staging) Merge pull request #3 from SanaTariq205/develop
|/
| * babae8a (origin/develop, develop) Merge pull request #2 from SanaTariq205/bugfix/fix-typo
|/
| * f44d4c6 (origin/bugfix/fix-typo, bugfix/fix-typo) Fix typo in notes.txt
|/
| * 1e2567f (feature/user-login) Merge pull request #1 from SanaTariq205/feature/db-connection
|/
| * 1068594 (origin/feature/db-connection, feature/db-connection) Add DB connection notes
| * | 45dc317 (origin/feature-1, feature-1) Add new function to main.py
|/
| * 84166f7 (origin/bugfix/user-auth-error) Update notes.txt
| * b8d3275 Add notes.txt
| * 97f9bad Initial commit

```

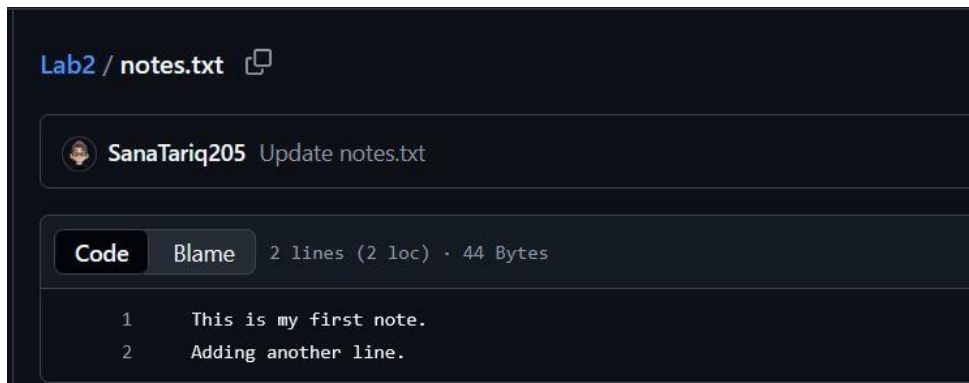
TASK#12:

Create a Pull Request (PR) / Merge Request (MR): From a feature branch into main. Add a clear title and description summarizing the changes.

The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with 'main' selected, '1 Branch', '0 Tags', a search bar, and buttons for 'Add file' and 'Code'. Below this, the commit history is displayed as a table:

| Commit Hash | Message | Time |
|-------------|------------------|----------------|
| 84166f7 | Update notes.txt | 4 minutes ago |
| b8d3275 | Add notes.txt | 16 minutes ago |
| 97f9bad | Initial commit | 16 minutes ago |

Below the commit history, there's a section for the README file, which is titled 'Lab2 Git Practice'.



```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git log --oneline
84166f7 (HEAD -> main, origin/main) Update notes.txt
b8d3275 Add notes.txt
97f9bad Initial commit
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature/db-connection)
$ git push origin feature/db-connection
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature/db-connection' on GitHub by visiting:
remote:   https://github.com/SanaTariq205/Lab2/pull/new/feature/db-connection
remote:
To github.com:SanaTariq205/Lab2.git
 * [new branch]      feature/db-connection -> feature/db-connection
```

TASK#13:

1. Delete Remote Branch After Merge: Use GitHub UI

| Branch | Updated | Check status | Behind | Ahead | Pull request |
|--------------------|--------------|--------------|--------|---------|--------------|
| main | 13 hours ago | | | Default | |
| four branches | | | | | |
| Branch | Updated | Check status | Behind | Ahead | Pull request |
| develop | 12 hours ago | | 2 | 4 | |
| staging | 13 hours ago | | 1 | 0 | #4 |
| bugfix/fix-type | Deleted now | | 1 | 0 | #2 |
| feature-1 | Deleted now | | | 6 | 0 |
| bugfix/patch-error | Deleted now | | | 7 | 0 |

EXAM EVALUATION QUESTIONS:

Q 1: Advanced Branching & Merge Verification

Create a new branch in your repository.

Make a small change in a file and commit it.

Merge this branch back into the main branch.

Show the history of commits in a way that verifies the merge.

Q 2: Multi-Stage Workflow Simulation Set up a branching workflow with three branches: main, develop, and staging.

Create a feature branch from develop, make changes, and commit them.

Merge the feature branch into develop.

Merge develop into staging.

Merge staging into main.

Provide proof that each stage contains the updated changes before it reaches main.

Q 3: Collaboration & Conflict Resolution Work with a collaborator: both contributors should modify the same file but in separate branches.

Attempt to merge the branches and capture the conflict.

Resolve the conflict so that both contributions are preserved.

Provide evidence that the final version of the file contains both collaborators' changes.

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git checkout -b exam-branch
Switched to a new branch 'exam-branch'
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (exam-branch)
$ notepad notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (exam-branch)
$ git status
On branch exam-branch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   notes.txt

no changes added to commit (use "git add" and/or "git commit -a")

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (exam-branch)
$ git add notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (exam-branch)
$ git commit -m "Exam Q1: small change"
[exam-branch ac5c83d] Exam Q1: small change
1 file changed, 1 insertion(+)
```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (exam-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git merge exam-branch
Updating 33ab573..ac5c83d
Fast-forward
 notes.txt | 1 +
 1 file changed, 1 insertion(+)

```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git log --oneline --graph --all
* ac5c83d (HEAD -> main, exam-branch) Exam Q1: small change
* 33ab573 (origin/main, origin/HEAD) Merge pull request #4 from SanaTariq205/staging
| \
| * 6e09887 (origin/staging, staging) Merge pull request #3 from SanaTariq205/develop
| /
| * bad8147 (origin/develop, develop) Merge pull request #6 from SanaTariq205/feature/add-readme
| | \
| | * 9e095ef (origin/feature/add-readme) Add test line for PR
| | * c89d22a Merge pull request #5 from SanaTariq205/feature/add-readme
| | /
| | /
| | * 095c14b Add note for code review workflow
| | /
| * babae8a Merge pull request #2 from SanaTariq205/bugfix/fix-typo
| /
| * f44d4c6 (origin/bugfix/fix-typo) Fix typo in notes.txt
| /
| * 1e2567f Merge pull request #1 from SanaTariq205/feature/db-connection
| \
| * 1068594 (origin/feature/db-connection) Add DB connection notes
| * | 45dc317 (origin/feature-1) Add new function to main.py
| /
| * 84166f7 (origin/bugfix/user-auth-error) Update notes.txt
* b8d3275 Add notes.txt
* 97f9bad Initial commit

```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-x)
$ notepad notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-x)
$ git add notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-x)
$ git commit -m "Q2: feature-x changes"
[feature-x d484554] Q2: feature-x changes
 1 file changed, 1 insertion(+)

```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-x)
$ git checkout develop
Switched to branch 'develop'

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (develop)
$ git merge feature-x
Updating bad8147..d484554
Fast-forward
 notes.txt | 1 +
 1 file changed, 1 insertion(+)

```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (staging)
$ git checkout staging
Already on 'staging'

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (staging)
$ git merge develop
Already up to date.
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (feature-1)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git merge feature-1
Updating 84166f7..45dc317
Fast-forward
 main.py | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 main.py
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (conflict_Sana)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git merge conflict_rabeea
merge: conflict_rabeea - not something we can merge

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git merge conflict_Sana
Updating 935a75d..27cf10e
Fast-forward
 notes.txt | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)
```

```
Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (collab)
$ git commit -m "Added Sana's Fun Fact"
[collab 935a75d] Added Sana's Fun Fact
 1 file changed, 1 insertion(+)

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (collab)
$ git push origin collab
ssh: Could not resolve hostname github.com: Name or service not known
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (collab)
$ git push origin collab
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 323 bytes | 323.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:SanaTariq205/Lab2.git
 7351ddd..935a75d collab -> collab
```



```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (conflict_Sana)
$ notepad notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (conflict_Sana)
$ git add notes.txt

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (conflict_Sana)
$ git commit -m "Sana changed line"
[conflict_Sana 27cf10e] Sana changed line
1 file changed, 2 insertions(+), 1 deletion(-)

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (conflict_Sana)
$ git push origin conflict_Sana
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 311 bytes | 311.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'conflict_Sana' on GitHub by visiting:
remote:   https://github.com/SanaTariq205/Lab2/pull/new/conflict_Sana
remote:
To github.com:SanaTariq205/Lab2.git
 * [new branch]      conflict_Sana -> conflict_Sana

```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git add notes.txt
warning: in the working copy of 'notes.txt', LF will be replaced by CRLF the next time Git touches it

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git commit -m "Add notes.txt"
[main b8d3275] Add notes.txt
1 file changed, 1 insertion(+)
create mode 100644 notes.txt

```

```

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   notes.txt

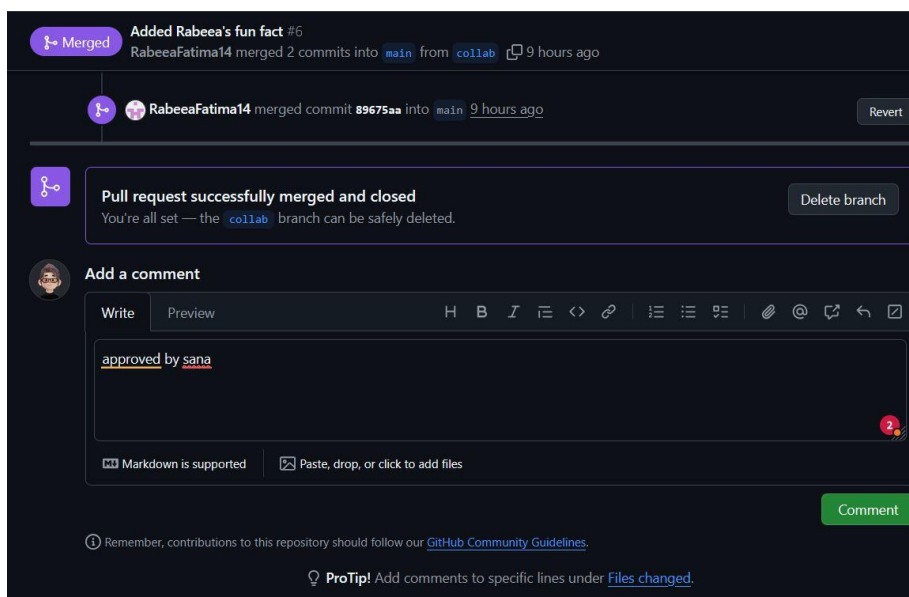
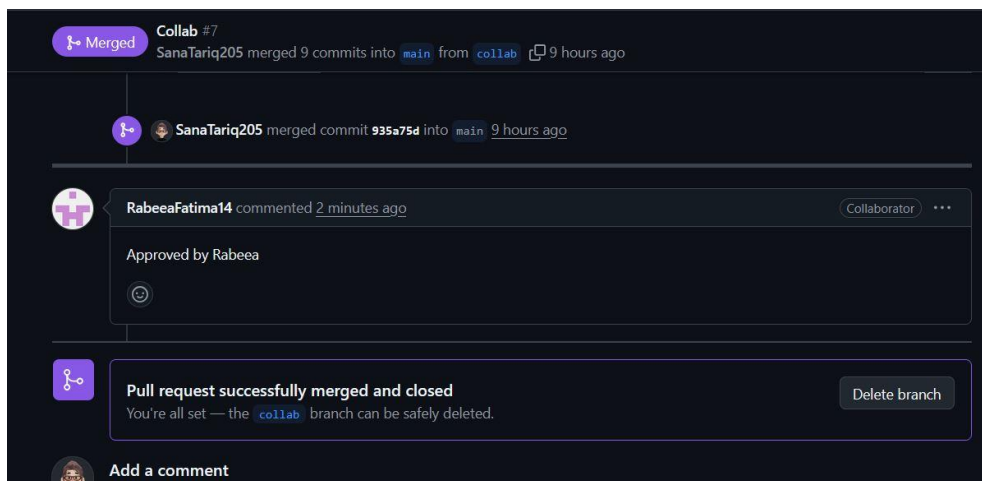
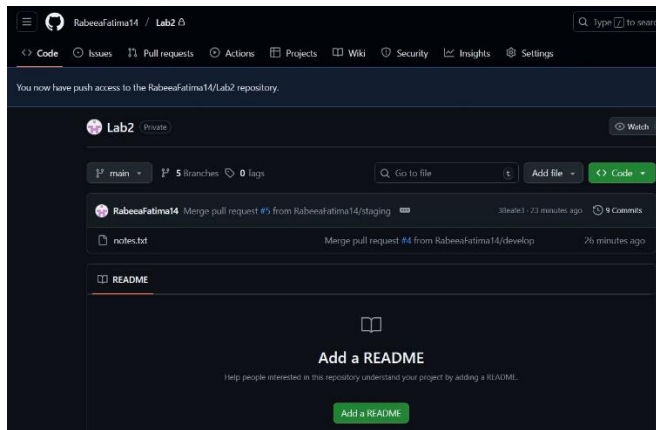
no changes added to commit (use "git add" and/or "git commit -a")

Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git add notes.txt
warning: in the working copy of 'notes.txt', LF will be replaced by CRLF the next time Git touches it


Sana@DESKTOP-00ECUUI MINGW64 ~/Lab2 (main)
$ git commit -m "Update notes.txt"
[main 84166f7] Update notes.txt
1 file changed, 1 insertion(+)

```


BONUS TASK:



Make SanaTariq205/Lab2 public



SanaTariq205/Lab2

☆ 0 stars 👁 0 watchers


- The code will be visible to everyone who can visit <https://github.com>
- Anyone can fork your repository.
- All push rulesets will be disabled.
- Your changes will be published as activity.
- Actions history and logs will be visible to everyone.


I have read and understand these effects

Reviewers

No reviews

Assignees

 RabeeaFatima14

 SanaTariq205