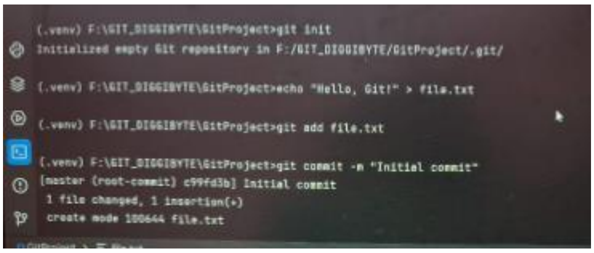
GIT ASSIGNMENT

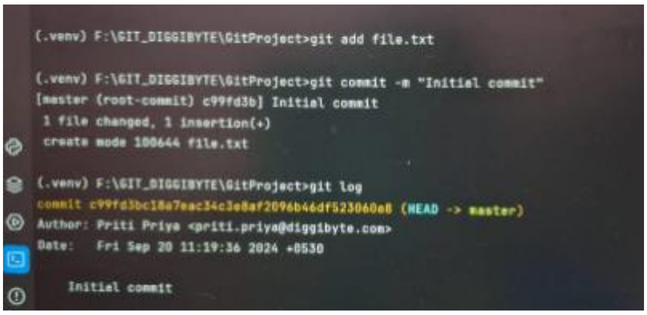
Question 1

Step 1: Create a new Git repository.



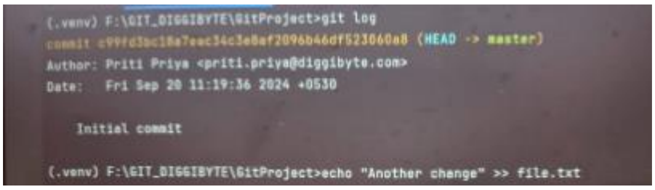
```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git init
Initialized empty Git repository in F:/GIT_DIGGIBYTE/GitProject/.git/

(.venv) F:\GIT_DIGGIBYTE\GitProject>echo "Hello, Git!" > file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject>git commit -m "Initial commit"
[master (root-commit) c99fd3b] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

Step 2: Create a file and commit changes.



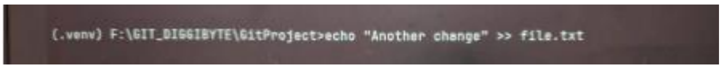
```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject>git commit -m "Initial commit"
[master (root-commit) c99fd3b] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject>git log
commit c99fd3bc18a7eac34c3e8af2096b46df523060a8 (HEAD -> master)
Author: Priti Priya <priti.priya@diggibyte.com>
Date:   Fri Sep 20 11:19:36 2024 +0530

    Initial commit
```

Step 3: View the commit history of your repository.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git log
commit c99fd3bc18a7eac34c3e8af2096b46df523060a8 (HEAD -> master)
Author: Priti Priya <priti.priya@diggibyte.com>
Date:   Fri Sep 20 11:19:36 2024 +0530

    Initial commit

(.venv) F:\GIT_DIGGIBYTE\GitProject>echo "Another change" >> file.txt
```

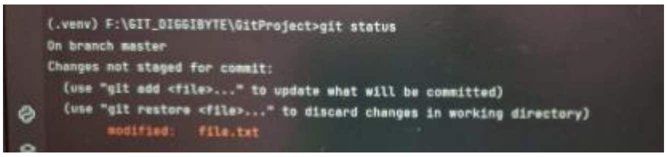
Step 4: Open the file you created earlier and make some changes to it.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject>echo "Another change" >> file.txt
```

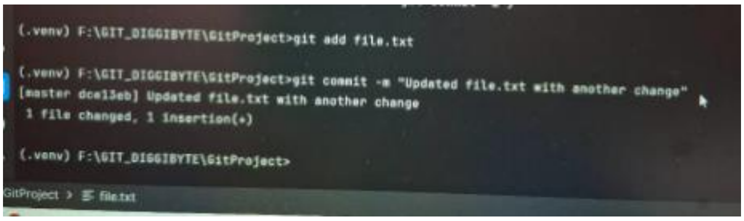
Step 5: Check the file you modified is now marked as "modified" and unstaged.

Hint (git status)

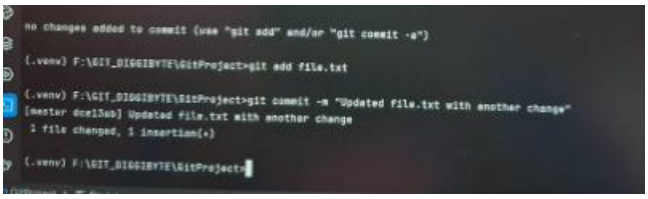


```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file.txt
```

Step 6: Stage the changes you made to the file and commit the changes to the repository.

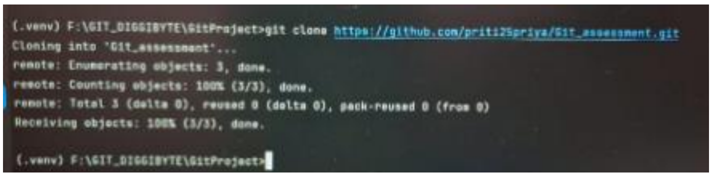


```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject>git commit -m "Updated file.txt with another change"
[master dce13eb] Updated file.txt with another change
 1 file changed, 1 insertion(+)

(.venv) F:\GIT_DIGGIBYTE\GitProject>
```



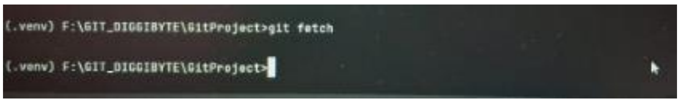
```
no changes added to commit (use "git add" and/or "git commit -a")

(.venv) F:\GIT_DIGGIBYTE\GitProject>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject>git commit -m "Updated file.txt with another change"
[master dce13eb] Updated file.txt with another change
 1 file changed, 1 insertion(+)

(.venv) F:\GIT_DIGGIBYTE\GitProject>
```

Step 7: Clone the repository you have created in GitHub.



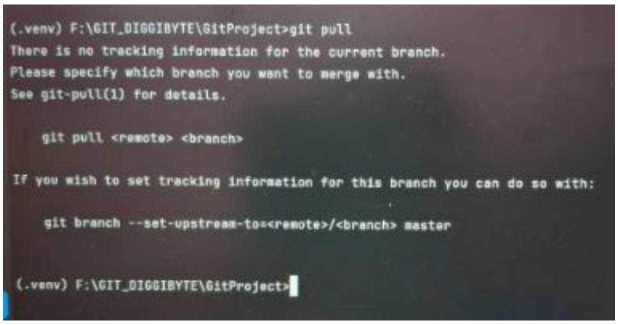
```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git clone https://github.com/priti125priya/Git_assessment.git
Cloning into 'Git_assessment'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

(.venv) F:\GIT_DIGGIBYTE\GitProject>
```

Step 8: Fetch the changes, navigate into the cloned repository using the command line, and use the command git fetch to fetch any changes that have been made to the original repository since you cloned it.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git fetch

(.venv) F:\GIT_DIGGIBYTE\GitProject>
```

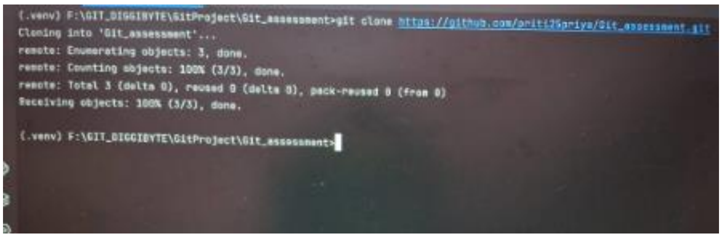
Step 9: Pull changes, merge the changes you just fetched into your local copy of the repository, and use the command git pull.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject>git pull
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=<remote>/<branch> master

(.venv) F:\GIT_DIGGIBYTE\GitProject>
```

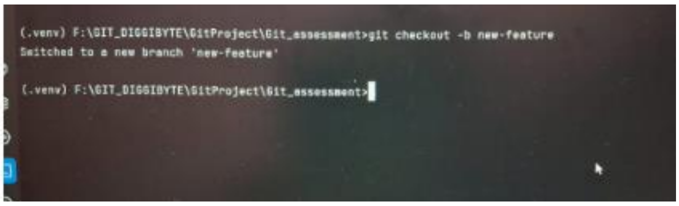
Question 2

Step 1: Clone the repository you have created in GitHub.



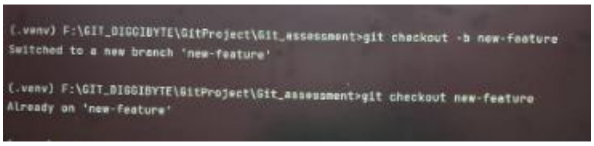
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git clone https://github.com/priti25priya/Git_assessment.git
Cloning into 'Git_assessment'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 2: Create a new branch using the command.



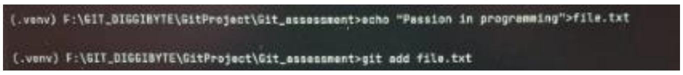
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b new-feature
Switched to a new branch 'new-feature'

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 3: Switch to the new branch.



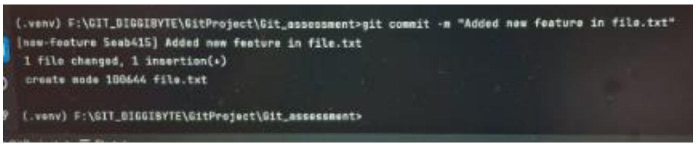
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b new-feature
Switched to a new branch 'new-feature'

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout new-feature
Already on 'new-feature'
```

Step 4: Make some changes to the code in your local copy of the repository.



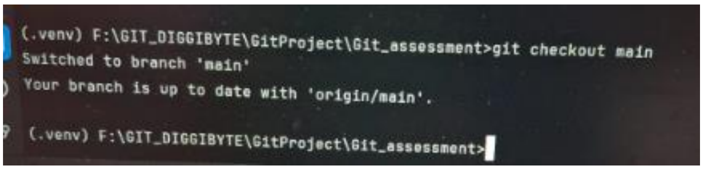
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>echo "Passion in programming">file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt
```

Step 5: Commit changes to the new branch.



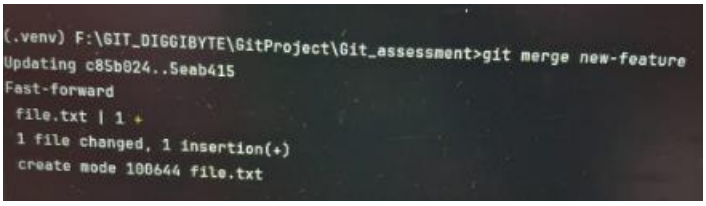
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "Added new feature in file.txt"
[new-feature 5eab415] Added new feature in file.txt
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 6: Switch back to the original branch



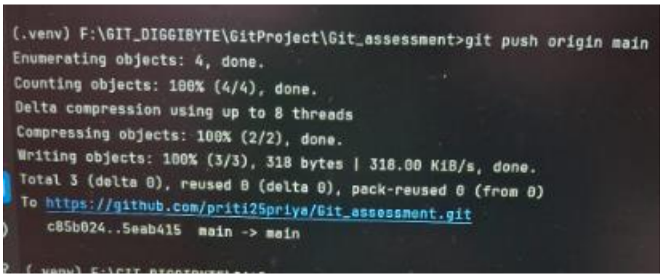
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 7: Merge the new branch.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git merge new-feature
Updating c85b024..5eab415
Fast-forward
 file.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

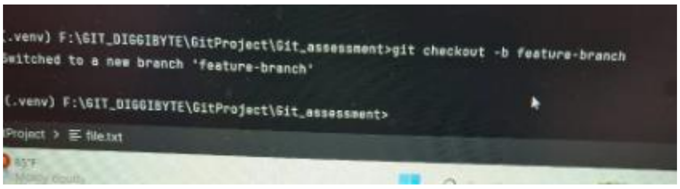
Step 8: Push changes to the original branch



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>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), 318 bytes | 318.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/priti25priya/Git_assessment.git
   c85b024..5eab415  main -> main
```

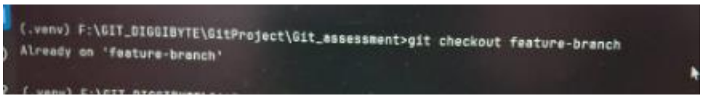
Question 3

Step 1: Create a feature branch.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b feature-branch
Switched to a new branch 'feature-branch'

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

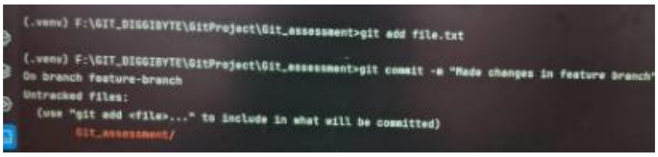
Step 2: Switch to the new branch.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout feature-branch
Already on 'feature-branch'
```

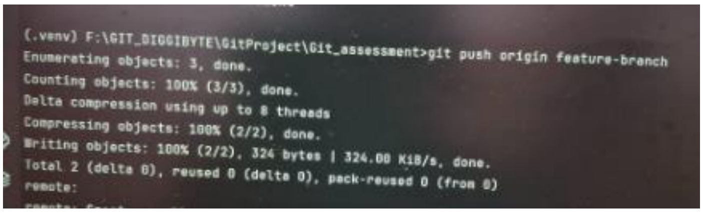
Step 3: open the file and make some changes to it.

Step 4: Add and commit the changes to the new branch.



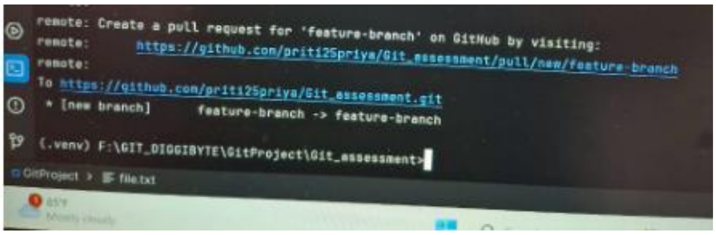
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "Made changes in feature branch"
On branch feature-branch
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Git_assessment/
```

Step 5: Push the changes to the new feature branch.

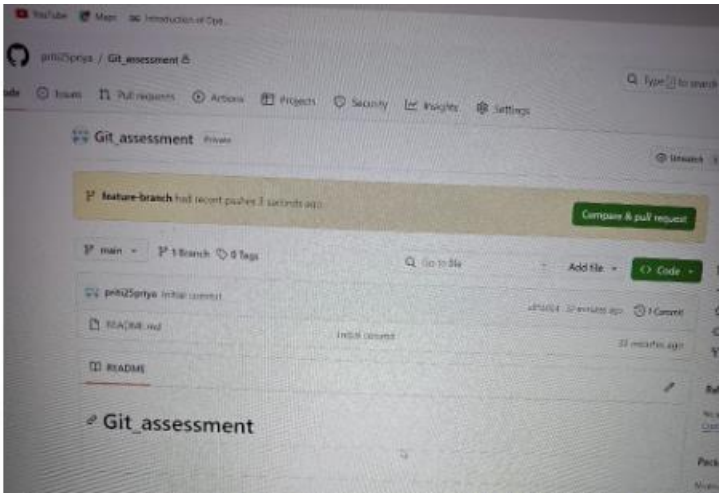


```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git push origin feature-branch
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 324 bytes | 324.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
```

Step 6: Create a pull request.



```
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:      https://github.com/priti25priya/Git_assessment/pull/new/feature-branch
remote:
To https://github.com/priti25priya/Git_assessment.git
 * [new branch]      feature-branch -> feature-branch

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```



priti25priya / Git_assessment

Git_assessment Private

feature-branch had recent pushes 3 seconds ago

Compare & pull request

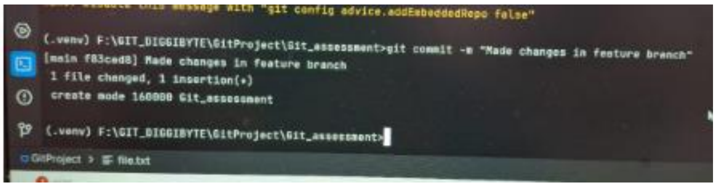
main 1 Branch 0 Tags

priti25priya Initial commit

README.md Initial commit

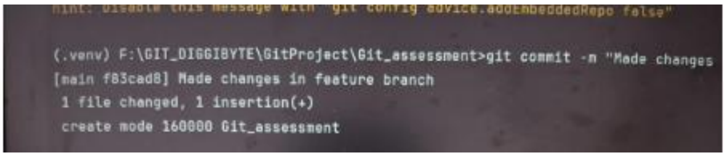
Git_assessment

Step 6: As another user in the master branch make some changes to the same file.



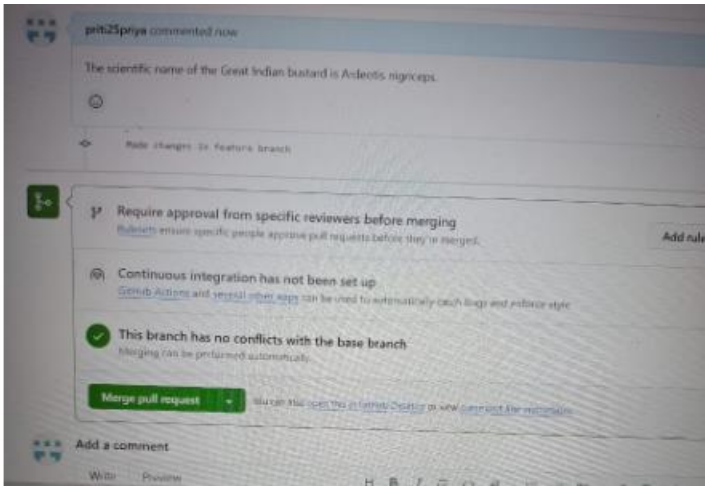
```
hint: Disable this message with "git config advice.addEmbeddedRepo false"

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "Made changes in feature branch"
[main f83ced8] Made changes in feature branch
 1 file changed, 1 insertion(+)
 create mode 160000 Git_assessment

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 7: Add and commit the changes to the master branch.



```
hint: Disable this message with "git config advice.addEmbeddedRepo false"

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "Made changes
[main f83cad8] Made changes in feature branch
 1 file changed, 1 insertion(+)
 create mode 160000 Git_assessment
```

Step 8: Push the changes to the master branch.



priti25priya commented now

The scientific name of the Great Indian bustard is Ardeotis nigriceps.

Made changes in feature branch

Require approval from specific reviewers before merging

Rulesets ensure specific people approve pull requests before they're merged.

Continuous integration has not been set up

GitHub Actions and several other apps can be used to automatically catch bugs and enforce style

This branch has no conflicts with the base branch

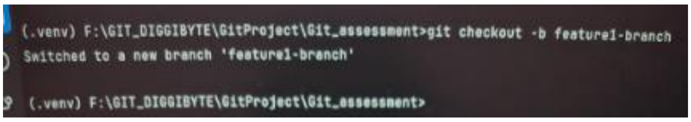
Merging can be performed automatically.

Note: There will be a conflict in the pull request, how do we resolve it??

Hint: git rebase

Question 4

Step 1: Step 1: Create a feature branch.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b feature1-branch
Switched to a new branch 'feature1-branch'

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 2: Switch to the new branch.

open the file and make some changes to it.

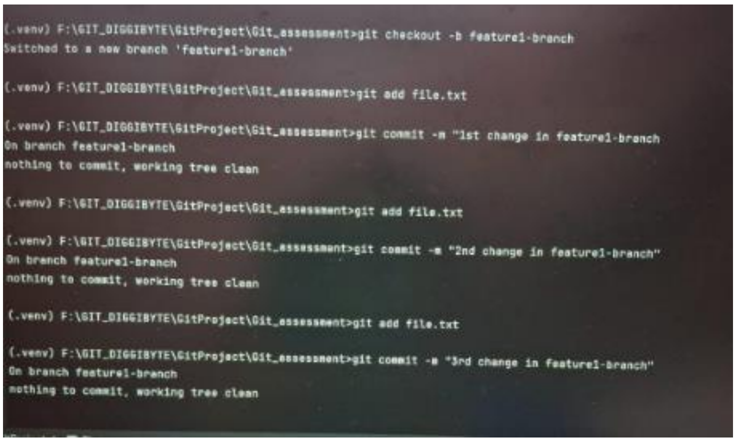
Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

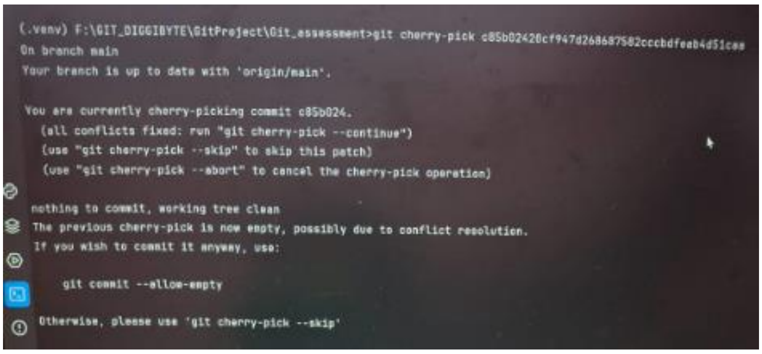
open the same file and make some changes to it.

Add and commit the changes to the new branch.



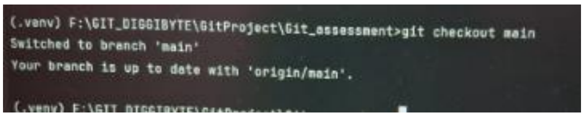
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b feature1-branch
Switched to a new branch 'feature1-branch'

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "1st change in feature1-branch
On branch feature1-branch
nothing to commit, working tree clean

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "2nd change in feature1-branch"
On branch feature1-branch
nothing to commit, working tree clean

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "3rd change in feature1-branch"
On branch feature1-branch
nothing to commit, working tree clean
```

Step 3: Identify the commit or commits that you want to "cherry-pick"(Note the hash of the commit or commits that you want to "cherry-pick")



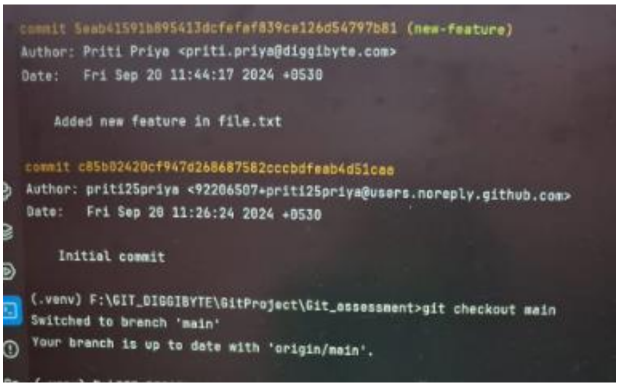
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git cherry-pick c85b02420cf947d268687582cccbdfeab4d51cea
On branch main
Your branch is up to date with 'origin/main'.

You are currently cherry-picking commit c85b024.
  (all conflicts fixed: run "git cherry-pick --continue")
  (use "git cherry-pick --skip" to skip this patch)
  (use "git cherry-pick --abort" to cancel the cherry-pick operation)

nothing to commit, working tree clean
The previous cherry-pick is now empty, possibly due to conflict resolution.
If you wish to commit it anyway, use:

    git commit --allow-empty

Otherwise, please use 'git cherry-pick --skip'
```

Step 4: Use the "git checkout" command to switch to the branch where you want to apply the changes.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

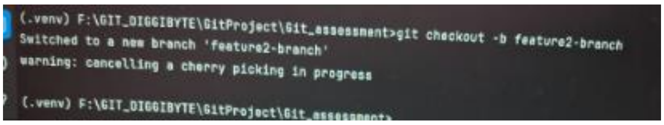
Step 5: Use the "git cherry-pick" command followed by the commit hash(es) that you want to apply.



```
commit 5eab41591b895413dcfefaf839ce126d54797b81 (new-feature)
Author: Priti Priya <priti.priya@diggibyte.com>
Date:   Fri Sep 20 11:44:17 2024 +0530

    Added new feature in file.txt

commit c85b02420cf947d268687582cccbdfeab4d51cae
Author: priti25priya <92206507+priti25priya@users.noreply.github.com>
Date:   Fri Sep 20 11:26:24 2024 +0530

    Initial commit

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Question 5

Step 1: Step 1: Create a feature branch.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b feature2-branch
Switched to a new branch 'feature2-branch'
warning: cancelling a cherry picking in progress

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>
```

Step 2: Switch to the new branch.

open the file and make some changes to it.

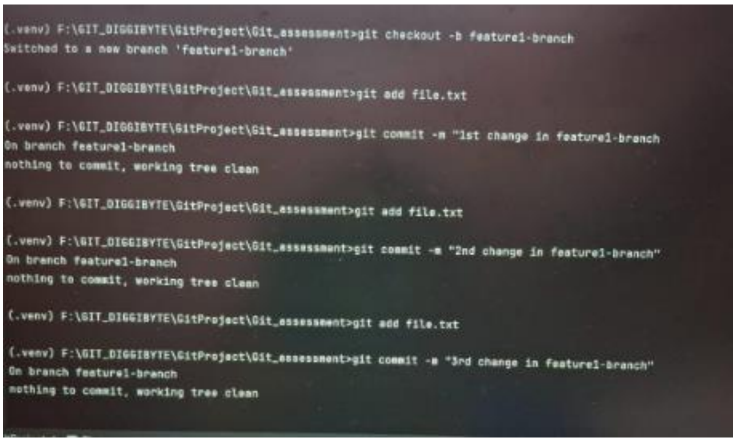
Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

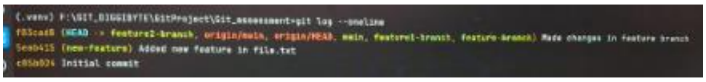
open the same file and make some changes to it.

Add and commit the changes to the new branch.



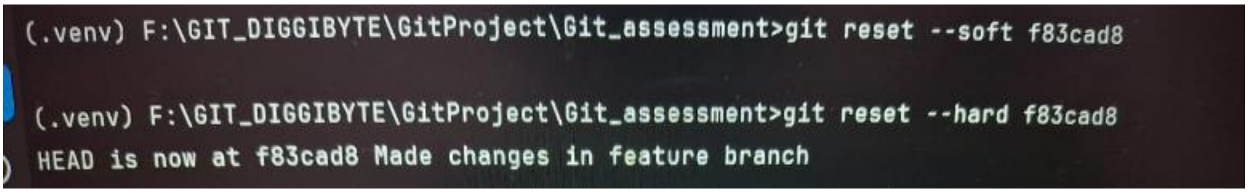
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git checkout -b feature1-branch
Switched to a new branch 'feature1-branch'

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "1st change in feature1-branch
On branch feature1-branch
nothing to commit, working tree clean

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "2nd change in feature1-branch"
On branch feature1-branch
nothing to commit, working tree clean

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git add file.txt

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git commit -m "3rd change in feature1-branch"
On branch feature1-branch
nothing to commit, working tree clean
```

Step 3: Use the "git log" command to view the commit history and identify the commit to which you want to reset.



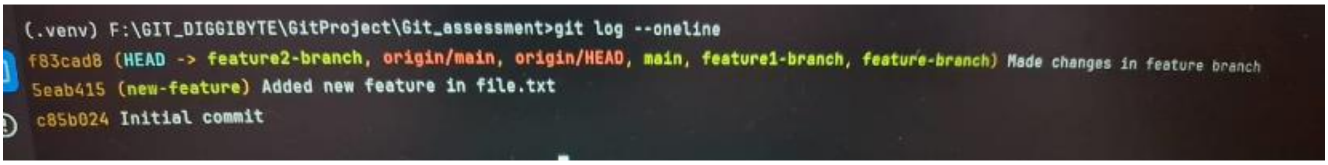
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git log --oneline
f83cad8 (HEAD -> feature2-branch, origin/main, origin/HEAD, main, feature1-branch, feature-branch) Made changes in feature branch
5eab415 (new-feature) Added new feature in file.txt
c85b024 Initial commit
```

Step 4: Use the "git reset" command followed by the desired reset type and the commit hash



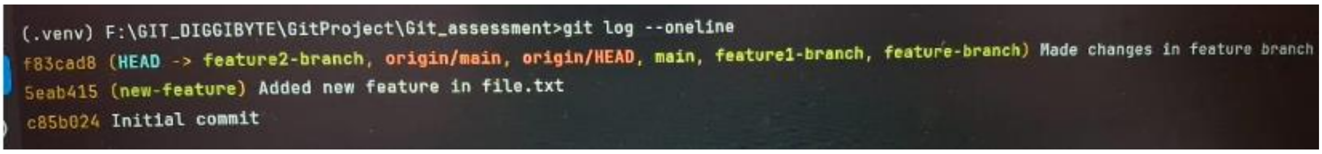
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git reset --soft f83cad8

(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git reset --hard f83cad8
HEAD is now at f83cad8 Made changes in feature branch
```

Step 5: Verify that the reset was successful by using the "git log" command again.



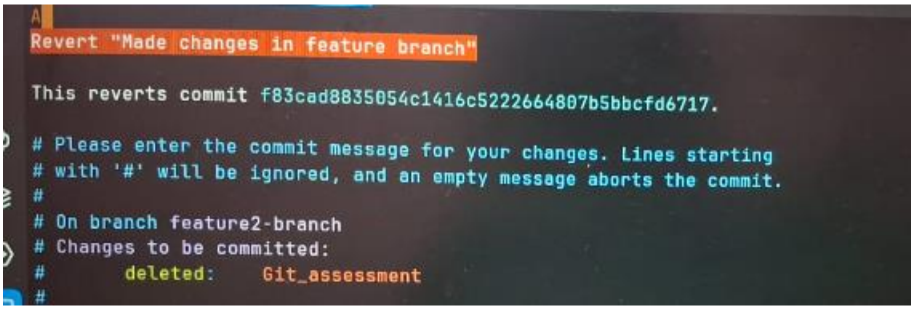
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git log --oneline
f83cad8 (HEAD -> feature2-branch, origin/main, origin/HEAD, main, feature1-branch, feature-branch) Made changes in feature branch
5eab415 (new-feature) Added new feature in file.txt
c85b024 Initial commit
```

Step 6: Use the "git log" command to view the commit history and identify the commit that you want to reverse.



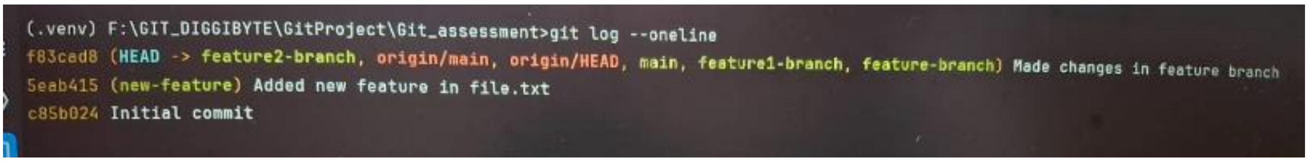
```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git log --oneline
f83cad8 (HEAD -> feature2-branch, origin/main, origin/HEAD, main, feature1-branch, feature-branch) Made changes in feature branch
5eab415 (new-feature) Added new feature in file.txt
c85b024 Initial commit
```

Step 7: Use the "git revert" command followed by the commit hash or reference to which you want to revert. (Hint: git revert <commit hash>)



```
A
Revert "Made changes in feature branch"

This reverts commit f83cad8835054c1416c5222664807b5bbcfd6717.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch feature2-branch
# Changes to be committed:
#       deleted:    Git_assessment
#
```

Step 8: Verify that the revert was successful by using the "git log" command again.



```
(.venv) F:\GIT_DIGGIBYTE\GitProject\Git_assessment>git log --oneline
f83cad8 (HEAD -> feature2-branch, origin/main, origin/HEAD, main, feature1-branch, feature-branch) Made changes in feature branch
5eab415 (new-feature) Added new feature in file.txt
c85b024 Initial commit
```

Note: Identify the difference between git log after git reset and git r evert.