

**Batch: D-2                      Roll No.: 16010123325**

**Experiment / assignment / tutorial No. \_1\_\_**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

**Title: Implementation of Version control System.**

**AIM:** Implementation of Version Control System (Git & GitHub)

**Objective:** Understand and utilize Git for version control.

• **Tasks:**

- Initialize a Git repository.
- Perform basic Git operations: add, commit, push, pull.
- Create and switch branches.
- Resolve merge conflicts.
- Collaborate using pull requests on GitHub

**Problem Definition:**

The goal of this assignment is to gain practical, hands-on experience with Git and GitHub—essential tools for modern software development and collaboration. Students will explore core Git operations by completing curated interactive exercises from the Git Exercises website (<https://gitexercises.fracz.com/>). They will demonstrate their understanding of Git commands, branch management, resolving merge conflicts, and contributing to remote repositories hosted on GitHub. The focus is on mastering the Git workflow, from repository initialization to managing collaborative development using pull requests.

**Resources used:**

---

**Expected OUTCOME of Experiment:**

**CO 1:** Build full stack applications in JavaScript using the MERN technologies.

**CO5:**Deploy MERN applications to cloud platforms like Heroku or AWS and collaborate using GitHub version control.

---

**Books/ Journals/ Websites referred:**

1. Shelly Powers Learning Node O' Reilly 2 nd Edition, 2016.

**Pre Lab/ Prior Concepts:**

Before beginning the experiment, students must be familiar with:

- The command line interface (CLI)
- Basic file and directory structure navigation
- Source code repositories
- Git vs GitHub
- The version control concept and its need in collaborative software development

**Methodology:**

1. Gain a deep understanding of fundamental Git operations through individual exploration.
2. Complete practical tasks provided in the Git Exercises.
3. Log each step taken along with its outcome for traceability.
4. Push all changes to a designated GitHub repository.
5. Collaborate using a shared repository by creating and managing pull requests.
6. Handle and resolve merge conflicts encountered during collaborative work.
7. Document the entire workflow with supporting screenshots, command logs, and personal reflections.

## Implementation Details:

### Exercise 1: master

**Objective:** Initialize a fresh Git repository with the default "master" branch and understand the base state.

**Git Commands Used:** start, master

**Outcome:** Successfully started the exercise with an empty repository.

#### Screenshot:

```
Preparing the exercise environment, hold on...
Exercise master started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/master
shreya@Shreyas-MacBook-Air-2 exercises % git start master
Preparing the exercise environment, hold on...
Exercise master started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/master
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the master exercise. Hold on...
Exercise: master
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/master/xu3
```

#### Steps for execution:

- Started the exercise using git start master
- Verified the repository state using git stat

**Reflection / Learning:** Gained understanding of how to start working in a clean Git repository using the default branch.

### Exercise 2: commit one file

**Objective:** Understand how to commit a newly created file to the Git repository.

#### Steps for execution:

- Created a new file
- Used git add to stage the file
- Committed using git commit

**Git Commands Used:** touch, add, commit

**Outcome:** The file was successfully committed.

#### Screenshot:

```
shreya@Shreyas-MacBook-Air-2 exercises % git start commit-one-file
Preparing the exercise environment, hold on...
Exercise commit-one-file started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/commit-one-file
shreya@Shreyas-MacBook-Air-2 exercises % git verify
You haven't made any progress on exercise commit-one-file.
Did you forget to commit your changes?
shreya@Shreyas-MacBook-Air-2 exercises % git status
On branch commit-one-file
Your branch is up to date with 'origin/commit-one-file'.

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

nothing added to commit but untracked files present (use "git add" to track)
shreya@Shreyas-MacBook-Air-2 exercises % git add A.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit -m "Adding A.txt"
[commit-one-file 6cef7cb] Adding A.txt
 1 file changed, 1 insertion(+)
 create mode 100644 A.txt
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the commit-one-file exercise. Hold on...
Exercise: commit-one-file
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/commit-one-file/xu3
```

**Reflection / Learning:** Learnt the basic workflow of creating, staging, and committing a file.

### Exercise 3: commit one staged file

**Objective:** Practice committing only staged files, leaving unstaged files untouched.

#### Steps Taken:

- Edited multiple files
- Staged only one file using git add
- Committed the staged file

**Git Commands Used:** add, commit

**Outcome:** Only the staged file was committed.

#### Screenshot:

```
shreya@Shreyas-MacBook-Air-2 exercises % git start next
Preparing the exercise environment, hold on...
Exercise commit-one-file-staged started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/commit-one-file-staged
shreya@Shreyas-MacBook-Air-2 exercises % git status
On branch commit-one-file-staged
Your branch is up to date with 'origin/commit-one-file-staged'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   A.txt
    new file:   B.txt

shreya@Shreyas-MacBook-Air-2 exercises % git reset
shreya@Shreyas-MacBook-Air-2 exercises % git add B.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit -m "Unstage + add one file"
[commit-one-file-staged 9c2f2da] Unstage + add one file
 1 file changed, 1 insertion(+)
 create mode 100644 B.txt
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the commit-one-file-staged exercise. Hold on...
Exercise: commit-one-file-staged
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/commit-one-file-staged/xu3
```

**Reflection / Learning:** Learnt the distinction between staged and unstaged changes in Git.

### Exercise 4: ignore-them

**Objective:** Learn how to ignore certain files using .gitignore.

#### Steps Taken:

- Created .gitignore file
- Added patterns to ignore temporary files

**Git Commands Used:** echo, add, commit

**Outcome:** Ignored files no longer show up in git status.

#### Screenshot:

```
shreya@Shreyas-MacBook-Air-2 exercises % git verify

Verifying the ignore-them exercise. Hold on...
Exercise: ignore-them
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/ignore-them/xu3

Next task: chase-branch
In order to start, execute: git start next
```

**Reflection / Learning:** Ignored files no longer show up in git status.

### Exercise 5: chase-branch

**Objective:** Understand how branches point to different commits and how they move.

**Steps Taken:**

- Checked out to a branch
- Made commits and observed branch movement

**Git Commands Used:** checkout

**Outcome:** Branch moved as expected with new commits.

**Screenshot:**

**Reflection / Learning:** Understood how Git branches act as pointers to specific commits.

### Exercise 6: merge-conflict

**Objective:** Experience and resolve a merge conflict.

**Steps Taken:**

- Made conflicting changes on two branches
- Attempted to merge and resolved the conflict

**Git Commands Used:** merge

**Outcome:** Merge completed after conflict resolution

**Screenshot:**

```
git commit --amend -m "merge and resolve"

[merge-conflict 1de859d] merge and resolve
Date: Mon Jul 28 15:05:41 2025 +0530
shreya@Shreyas-MacBook-Air-2 exercises % git push --force origin merge-conflict

Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (12/12), 1.16 KiB | 1.16 MiB/s, done.
Total 12 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: (
remote: *****
remote: Exercise: merge-conflict
remote: Status: PASSED
remote: You can see the easiest known solution and further info at:
remote: https://gitexercises.fracz.com/e/merge-conflict/xu3
remote:
remote: Next task: remove-ignored
remote: In order to start, execute: git start next
remote:
remote: See your progress and instructions at:
remote: https://gitexercises.fracz.com/c/xu3
remote: *****
remote: )
remote: Remember that you can use git verify to strip disturbing output.
remote: error: hook declined to update refs/heads/merge-conflict
remote: warning: The last gc run reported the following. Please correct the root cause
remote: and remove gc.log.
remote: Automatic cleanup will not be performed until the file is removed.
remote: warning: There are too many unreachable loose objects; run 'git prune' to remove them.
remote:
To https://gitexercises.fracz.com/git/exercises.git
! [remote rejected] merge-conflict -> merge-conflict (hook declined)
error: failed to push some refs to 'https://gitexercises.fracz.com/git/exercises.git'
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the merge-conflict exercise. Hold on...
Exercise: merge-conflict
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/merge-conflict/xu3
```

**Reflection / Learning:** Understood how Git branches act as pointers to specific commits.

### Exercise 7: save-your-work

**Objective:** Learn to use git stash to temporarily save and restore work.

**Steps Taken:**

- Made uncommitted changes
- Stashed changes using git stash
- Restored them later using git stash pop

**Git Commands Used:** stash, stash pop

**Outcome:** Changes were successfully stashed and restored.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git start save-your-work
Preparing the exercise environment, hold on...
Exercise save-your-work started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/save-your-work
shreya@Shreyas-MacBook-Air-2 exercises % git stash
Saved working directory and index state WIP on save-your-work: 55c720f Excellent version with a bug
shreya@Shreyas-MacBook-Air-2 exercises % git commit -am "remove bug"
[save-your-work 38b64bb] remove bug
1 file changed, 1 deletion(-)
shreya@Shreyas-MacBook-Air-2 exercises % git stash apply
Auto-merging bug.txt
On branch save-your-work
Your branch is ahead of 'origin/save-your-work' by 2 commits.
(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:   bug.txt
        modified:   program.txt

no changes added to commit (use "git add" and/or "git commit -a")
shreya@Shreyas-MacBook-Air-2 exercises % git commit -am "finish"
[save-your-work c046cee] finish
2 files changed, 7 insertions(+), 2 deletions(-)
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the save-your-work exercise. Hold on...
Exercise: save-your-work
Status: PASSED
```

**Exercise 8: change-branch-history**

**Objective:** Modify a branch's history using git rebase or git commit --amend.

**Steps Taken:**

- Used git rebase -i to edit commit history
- Modified messages and merged commits

**Git Commands Used:** rebase, commit

**Outcome:** Branch history was successfully edited.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git start change-branch-history
Preparing the exercise environment, hold on...
Exercise change-branch-history started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/change-branch-history
shreya@Shreyas-MacBook-Air-2 exercises % git checkout change-branch-history
Already on 'change-branch-history'
Your branch is ahead of 'origin/change-branch-history' by 2 commits.
(use "git push" to publish your local commits)
shreya@Shreyas-MacBook-Air-2 exercises % git rebase hot-bugfix
Successfully rebased and updated refs/heads/change-branch-history.
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the change-branch-history exercise. Hold on...
Exercise: change-branch-history
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/change-branch-history/xu3
```

**Reflection / Learning:** Gained experience in rewriting history carefully.

**Exercise 9: remove-ignored**

**Objective:** Clean the working directory of ignored files.

**Steps Taken:**

- Checked ignored files using git status
- Removed them using Git clean

**Git Commands Used:** clean

**Outcome:** Ignored files removed from the working directory.

**Screenshot:**

```

● shreya@Shreyas-MacBook-Air-2 exercises % git start next
Preparing the exercise environment, hold on...
Exercise remove-ignored started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/remove-ignored
● shreya@Shreyas-MacBook-Air-2 exercises % git rm --cached ignored.txt
rm 'ignored.txt'
● shreya@Shreyas-MacBook-Air-2 exercises % git commit -am "untrack ignored.txt"
[remove-ignored 5dd5ea9] untrack ignored.txt
1 file changed, 1 deletion(-)
delete mode 100644 ignored.txt
● shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the remove-ignored exercise. Hold on...
Exercise: remove-ignored
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/remove-ignored/xu3

Next task: case-sensitive-filename
In order to start, execute: git start next

See your progress and instructions at:
https://gitexercises.fracz.com/c/xu3
○ shreya@Shreyas-MacBook-Air-2 exercises %

```

**Reflection / Learning:** Learned how to clean ignored files efficiently from the workspace.

## Exercise 10: case-sensitive-file

**Objective:** Handle and commit files with only case differences in their names.

**Steps Taken:**

- Created files with case-only name differences
- Observed behavior on case-sensitive and insensitive systems

**Git Commands Used:** add, commit

**Outcome:** Git tracked both files depending on the system's case sensitivity.

**Screenshot:**

```

shreya@Shreyas-MacBook-Air-2 exercises % git start next
Preparing the exercise environment, hold on...
Exercise case-sensitive-filename started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/case-sensitive-filename
shreya@Shreyas-MacBook-Air-2 exercises % git reset HEAD^
shreya@Shreyas-MacBook-Air-2 exercises % mv File.txt file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit -m "lowercase filename"
[case-sensitive-filename 4897dfa] lowercase filename
1 file changed, 1 insertion(+)
create mode 100644 file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the case-sensitive-filename exercise. Hold on...
Exercise: case-sensitive-filename
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/case-sensitive-filename/xu3

Next task: fix-typo
In order to start, execute: git start next

See your progress and instructions at:
https://gitexercises.fracz.com/c/xu3
shreya@Shreyas-MacBook-Air-2 exercises %

```

**Reflection / Learning:** Git is case-sensitive but local file systems may not be; important for collaboration.

## Exercise 11: fix-typo

**Objective:** Fix a typo in a previous commit.

**Steps Taken:**

- Fixed the typo in the file
- Amended the last commit

**Git Commands Used:** commit --amend

**Outcome:** Commit was updated with corrected content.

**Screenshot:**

```

shreya@Shreyas-MacBook-Air-2 exercises % git start next
Preparing the exercise environment, hold on...
Exercise fix-typo started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/fix-typo
shreya@Shreyas-MacBook-Air-2 exercises % nano file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit -a --amend
[fix-typo edd5e67] Add Hello world
Date: Mon Jul 28 15:47:05 2025 +0530
1 file changed, 1 insertion(+)
create mode 100644 file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add .
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the fix-typo exercise. Hold on...
Exercise: fix-typo
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/fix-typo/xu3

```

**Reflection / Learning:** Learnt how to fix recent commits without creating new ones.

### Exercise 12: forged-date

**Objective:** Simulate setting a commit to a specific historical date.

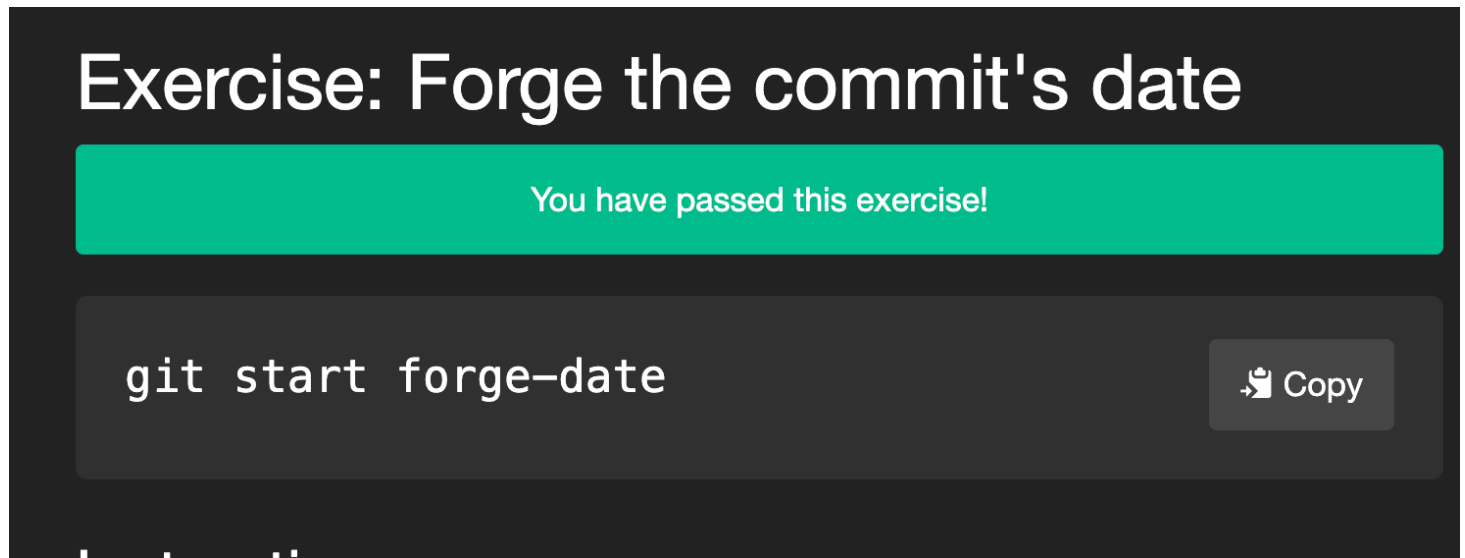
**Steps Taken:**

- Configured environment variable GIT\_COMMITTER\_DATE
- Created commit with custom date

**Git Commands Used:** committer

**Outcome:** Commit was successfully updated with forged date.

**Screenshot:**



**Reflection / Learning:** Useful for backdating historical commits when importing old code.

### Exercise 13: fix-old-typo

**Objective:** Fix a typo that exists in an older commit and update the commit history.

**Steps Taken:**

- Identified the commit containing the typo
- Used git rebase -i to edit that commit

- Corrected the typo and amended the commit

**Git Commands Used:** rebase, commit --amend

**Outcome:** Old commit was corrected and history updated.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git rebase -i HEAD~^
Stopped at d64f5d1: Add Hello world
You can amend the commit now, with

    git commit --amend

Once you are satisfied with your changes, run

    git rebase --continue
shreya@Shreyas-MacBook-Air-2 exercises % nano file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit --amend
[detached HEAD 2237011] Add Hello world
Date: Mon Jul 28 18:28:00 2025 +0530
1 file changed, 1 insertion(+)
create mode 100644 file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git rebase --continue
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not apply 443bb99... Further work on Hello world
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit; run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply 443bb99... Further work on Hello world
shreya@Shreyas-MacBook-Air-2 exercises % nano file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git rebase --continue
[detached HEAD f92dbc4] Further work on Hello world
1 file changed, 1 insertion(+)
Successfully rebased and updated refs/heads/fix-old-typo.
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the fix-old-typo exercise. Hold on...
Exercise: fix-old-typo
Status: PASSED
```

**Reflection / Learning:** Fixed historical commits using interactive rebase.

## Exercise 14: commit-lost

**Objective:** Recover a lost commit using the reflog.

**Steps Taken:**

- Used git reflog to find lost commit reference
- Checked it out and reapplied changes if necessary

**Git Commands Used:** reflog

**Outcome:** Recovered and restored lost commit data.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git start next
Preparing the exercise environment, hold on...
Exercise fix-typo started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/fix-typo
shreya@Shreyas-MacBook-Air-2 exercises % nano file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit -a --amend
[fix-typo edd5e67] Add Hello world
Date: Mon Jul 28 15:47:05 2025 +0530
1 file changed, 1 insertion(+)
create mode 100644 file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add .
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the fix-typo exercise. Hold on...
Exercise: fix-typo
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/fix-typo/xu3
```

**Reflection / Learning:** Reflog is a powerful tool to rescue mistakenly deleted or unreachable commits.

## Exercise 15: split-commit

**Objective:** Split a single commit into multiple meaningful commits.

**Steps Taken:**

- Used git rebase -i with edit

- Reset and staged changes in parts
- Committed each part separately

**Git Commands Used:** rebase, reset, add, commit

**Outcome:** One large commit was broken into two smaller, descriptive commits.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git rebase -i HEAD^^
Stopped at d64fc5d... Add Hello world
You can amend the commit now, with

    git commit --amend

Once you are satisfied with your changes, run

    git rebase --continue
shreya@Shreyas-MacBook-Air-2 exercises % nano file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit --amend

[detached HEAD 2237811] Add Hello world
Date: Mon Jul 28 18:28:00 2025 +0530
1 file changed, 1 insertion(+)
create mode 100644 file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git rebase --continue

Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not apply 443bb99... Further work on Hello world
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git rebase --abort".
hint: Disable this message with "git config advice.mergeConflict false"
Could not apply 443bb99... Further work on Hello world
shreya@Shreyas-MacBook-Air-2 exercises % nano file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git add file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git rebase --continue

[detached HEAD f92dbc4] Further work on Hello world
1 file changed, 1 insertion(+)
Successfully rebased and updated refs/heads/fix-old-typo.
shreya@Shreyas-MacBook-Air-2 exercises % git verify

Verifying the fix-old-typo exercise. Hold on...
Exercise: fix-old-typo
Status: PASSED
```

**Reflection / Learning:** Helped improve commit granularity for better code tracking.

## Exercise 16: too-many-commits

**Objective:** Squash multiple small commits into one.

**Steps Taken:**

- Initiated an interactive rebase
- Used squash or fixup to combine commits

**Git Commands Used:** squash

**Outcome:** Multiple commits reduced to one cleaner commit.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git start
Preparing the exercise environment, hold on...
Exercise too-many-commits started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/too-many-commits
shreya@Shreyas-MacBook-Air-2 exercises % git rebase -i HEAD~4

error: Your local changes to the following files would be overwritten by checkout:
    file.txt
Please commit your changes or stash them before you switch branches.
Aborting
error: could not detach HEAD
shreya@Shreyas-MacBook-Air-2 exercises % git restore file.txt
shreya@Shreyas-MacBook-Air-2 exercises % git rebase -i HEAD~4

[detached HEAD fdb1614] Add file.txt
Date: Mon Jul 28 20:16:56 2025 +0530
1 file changed, 2 insertions(+)
create mode 100644 file.txt
Successfully rebased and updated refs/heads/too-many-commits.
shreya@Shreyas-MacBook-Air-2 exercises % git verify

Verifying the too-many-commits exercise. Hold on...
Exercise: too-many-commits
Status: PASSED
```

**Reflection / Learning:** Multiple commits reduced to one cleaner commit.

## Exercise 17: executable

**Objective:** Track file permission changes, specifically executable bit.

**Steps Taken:**

- Changed file permission using `chmod +x`
- Committed the permission change

**Git Commands Used:** `chmod`, `add`, `commit`

**Outcome:** File became executable, and change tracked in Git.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git start next
Preparing the exercise environment, hold on...
Exercise executable started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/executable
shreya@Shreyas-MacBook-Air-2 exercises % git update-index --chmod=+x script.sh
shreya@Shreyas-MacBook-Air-2 exercises % git commit -m "make executable"
[executable a2f3c80] make executable
1 file changed, 0 insertions(+), 0 deletions(-)
mode change 100644 => 100755 script.sh
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the executable exercise. Hold on...
Exercise: executable
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/executable/xu3
```

**Reflection / Learning:** Learnt how Git handles file metadata like permissions.

## Exercise 18: commit-parts

**Objective:** Make partial commits from modified files using `git add -p`.

**Steps Taken:**

- Used `git add -p` to interactively stage only some changes
- Committed staged changes

**Git Commands Used:** `add`, `commit`

**Outcome:** Committed only selected chunks of file modifications.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git add --patch file.txt
diff --git a/file.txt b/file.txt
index 0c668f4..b04d86b 100644
--- a/file.txt
+++ b/file.txt
@@ -1,4 +1,9 @@
-I forgot to add file header.
+This is a program
+And this is new feature done in task 1.
+It lasts for many lines as task 1 was big.
+It is supposed to work.
-It works!
+It works!
+This is not related, it is task 2.
+It is quite brilliant, actually.
+Task 1 is finished.
(1/1) Stage this hunk [y,n,q,a,d,s,e,p,?]? s
Split into 4 hunks.
@@ -1 +1,2 @@
-I forgot to add file header.
+This is a program
(1/4) Stage this hunk [y,n,q,a,d,j,j,g,/,e,p,?]? n
@@ -1,2 +2,4 @@
+This is a program
+And this is new feature done in task 1.
+It lasts for many lines as task 1 was big.
+It is supposed to work.
(2/4) Stage this hunk [y,n,q,a,d,k,j,j,g,/,e,p,?]? y
@@ -2,3 +3,4 @@
-It is supposed to work.
+It works!
+This is not related, it is task 2.
+It is quite brilliant, actually.
(3/4) Stage this hunk [y,n,q,a,d,k,j,j,g,/,e,p,?]? n
@@ -4 +4,2 @@
+It is quite brilliant, actually.
+Task 1 is finished.
(4/4) Stage this hunk [y,n,q,a,d,k,g,/,e,p,?]? y
shreya@Shreyas-MacBook-Air-2 exercises % git commit -m "task1"
[commit-parts 1dd8725] task1
1 file changed, 3 insertions(+)
shreya@Shreyas-MacBook-Air-2 exercises % git commit -am "task2"
[commit-parts ff1e7c2] task2
1 file changed, 3 insertions(+), 1 deletion(-)
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the commit-parts exercise. Hold on...
Exercise: commit-parts
Status: PASSED
You can see the easiest known solution and further info at:
```

**Reflection / Learning:** `git add -p` enables highly controlled commits, useful for clean history.

## Exercise 19: pick-your-features

**Objective:** Selectively cherry-pick commits from another branch.

**Steps Taken:**

- Identified commits using git log
- Cherry-picked specific commits

**Git Commands Used:** cherry-pick

**Outcome:** Only the required changes were added to current branch.

**Screenshot:**

```
shreya@Shreyas-MacBook-Air-2 exercises % git cherry-pick feature-a
[pick-your-features ce168f6] Implement Feature A
Date: Wed Jul 30 22:02:50 2025 +0530
1 file changed, 1 insertion(+)
shreya@Shreyas-MacBook-Air-2 exercises % git cherry-pick feature-b
Auto-merging program.txt
[pick-your-features 9516f3f] Implement Feature B
Date: Wed Jul 30 22:02:50 2025 +0530
1 file changed, 1 insertion(+)
shreya@Shreyas-MacBook-Air-2 exercises % git merge --squash feature-c
Auto-merging program.txt
CONFLICT (content): Merge conflict in program.txt
Squash commit -- not updating HEAD
Automatic merge failed; fix conflicts and then commit the result.
shreya@Shreyas-MacBook-Air-2 exercises % nano program.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit -am "Complete Feature C"
[pick-your-features 90d19a0] Complete Feature C
1 file changed, 2 insertions(+)
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the pick-your-features exercise. Hold on...
Exercise: pick-your-features
Status: PASSED
```

**Reflection / Learning:** Cherry-pick helps apply individual changes without merging entire branches.

## Exercise 20: rebase-complex

**Objective:** Resolve conflicts during a complex rebase operation.

**Steps Taken:**

- Started interactive rebase
- Fixed conflicts at each step
- Continued rebase until complete

**Git Commands Used:** rebase -i ,--continue

**Outcome:** Branch history rewritten with conflicts resolved at each step.

### Screenshot:

```
● shreya@Shreyas-MacBook-Air-2 exercises % git start rebase-complex
Preparing the exercise environment, hold on...
Exercise rebase-complex started!
Read the README.md for instructions or view them in browser:
http://gitexercises.fracz.com/e/rebase-complex
● shreya@Shreyas-MacBook-Air-2 exercises % git rebase --onto your-master issue-555 rebase-complex
Successfully rebased and updated refs/heads/rebase-complex.
● shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the rebase-complex exercise. Hold on...
Exercise: rebase-complex
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/rebase-complex/xu3
```

**Reflection / Learning:** Complex rebasing for understanding Git internals and conflict handling.

### Exercise 21: invalid-order

**Objective:** Correct the chronological order of commits.

**Steps Taken:**

- Rearranged commits using git rebase -i
- Verified correct logical ordering

**Git Commands Used:** rebase -i

**Outcome:** Commits were ordered to reflect intended sequence.

### Screenshot:

```
● shreya@Shreyas-MacBook-Air-2 exercises % git rebase -i HEAD~4
Successfully rebased and updated refs/heads/invalid-order.
● shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the invalid-order exercise. Hold on...
Exercise: invalid-order
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/invalid-order/xu3
```

**Reflection / Learning:** Proper commit ordering improves clarity and traceability in version history.

### Exercise 22: find-swearwords

**Objective:** Detect and remove inappropriate words from commit messages or code.

**Steps Taken:**

- Reviewed commit history
- Searched for inappropriate content
- Amended or rebased to correct it

**Git Commands Used:** log, rebase, commit --amend

**Outcome:** Commit history cleaned up for professionalism.

### Screenshot:

```
shreya@Shreyas-MacBook-Air-2 exercises % git log -p -1
commit 20e0ce341d4410ce8a8972de35b407c17289387e (HEAD)
Author: SM006 <shreyam1005@gmail.com>
Date:   Wed Jul 30 22:10:37 2025 +0530

    Add word #94

diff --git a/words.txt b/words.txt
index 6f798f8..79df695 100644
--- a/words.txt
+++ b/words.txt
@@ -90,3 +90,5 @@ risus

    feugiat.

+shit
+
shreya@Shreyas-MacBook-Air-2 exercises % git add words.txt
shreya@Shreyas-MacBook-Air-2 exercises % git commit --amend
[detached HEAD 20f437b] Add word #94
Date:   Wed Jul 30 22:10:37 2025 +0530
1 file changed, 2 insertions(+)
shreya@Shreyas-MacBook-Air-2 exercises % git rebase --continue
Successfully rebased and updated refs/heads/find-swearwords.
shreya@Shreyas-MacBook-Air-2 exercises % git verify
Verifying the find-swearwords exercise. Hold on...
Exercise: find-swearwords
Status: PASSED
```

**Reflection / Learning:** Maintaining clean, respectful commit history is crucial in professional environments.

### Exercise 23: find-bug

**Objective:** Locate the commit that introduced a bug using git bisect.

**Steps Taken:**

- Started bisect process with good/bad commits
- Used git bisect run or tested manually

**Git Commands Used:** bisect start, bisect bad, bisect good, bisect reset

**Outcome:** Pinpointed the exact commit where the bug was introduced.

### Screenshot:

```
shreya@Shreyas-MacBook-Air-2 exercises % git verify
You need to use git verify <exercise-name> in detached HEAD
shreya@Shreyas-MacBook-Air-2 exercises % git verify find-bug
Verifying the find-bug exercise. Hold on...
Exercise: find-bug
Status: PASSED
You can see the easiest known solution and further info at:
https://gitexercises.fracz.com/e/find-bug/xu3

Congratulations! You have done all exercises!

See your progress and instructions at:
https://gitexercises.fracz.com/c/xu3
shreya@Shreyas-MacBook-Air-2 exercises %
```

**Reflection / Learning:** Powerful debugging tool that saves time when identifying regressions.

# ShreyansTatiya

Passed 23 exercises

23 / 23

## Congratulations!

You managed to pass all exercises!

You Are Mighty!

Please find below a random kitten generated specially for you.



find-bug	✓
2 hours ago	
find-bug	✓
2 hours ago	
find-swearwords	✓
2 hours ago	
pick-your-features	✓
2 hours ago	
invalid-order	✓
2 hours ago	
invalid-order	✗
2 hours ago	
invalid-order	✗
2 hours ago	
invalid-order	✗

**Steps for execution:**

1. Visit the website: <https://gitexercises.fracz.com>
2. Configure Git username and email
3. clone the Git Exercises environment or follow the instructions provided to start exercises using the git start <exercise-name> command (if installed locally).
4. For each exercise:
  - a. Read the task and goal
  - b. Perform operations in the terminal
  - c. Use appropriate Git commands (add, commit, branch, merge, etc.)
  - d. Resolve any errors/conflicts
5. Git Verify

**Conclusion:**

In this experiment, I explored key Git concepts such as committing, branching, rebasing, conflict resolution, and collaboration through GitHub. By working through practical scenarios, I developed a stronger grasp of version control workflows. I now feel more confident using Git for both personal projects and collaborative development. The hands on exercises helped me understand not just the commands, but also the logic and purpose behind each one.