# Lab 1 — Setup & First Commit

Q1. Check Git installation

Code

git —version

Output:

git version 2.44.0

Q2. Set global username & email

Code

git config --global user.name "Angel Snowlita"

git config --global user.email "angel@example.com"

git config --global —list

Output:

user.name=Angel Snowlita

user.email=angel@example.com

Q3. Initialize a repository

Code

mkdir GitDemo && cd GitDemo

git init

ls -la

Output:

Initialized empty Git repository in .../GitDemo/.git/

. .. .git

Q4. Create a file and verify

Code

echo "Welcome to Git" > welcome.txt

ls

cat welcome.txt

Output:

welcome.txt

Welcome to Git

Q5. Stage & commit

Code

git status

git add welcome.txt

git commit -m "Add welcome.txt"

git status

Output:

Untracked files: welcome.txt

[main (root-commit) a1b2c3d] Add welcome.txt

1 file changed, 1 insertion(+)

create mode 100644 welcome.txt

nothing to commit, working tree clean

Q6. Add remote & push

Code

git remote add origin https://github.com/<user>/<repo>.git

git branch -M main

git push -u origin main

Output:

Enumerating objects: 3, done.

To https://github.com/<user>/<repo>.git

\* [new branch] main -> main

branch 'main' set up to track ‘origin/main'.

# Lab 2 — .gitignore

Q1. Create log files

Code

touch debug.log

mkdir -p log && touch log/app.log

Q2. Create .gitignore to ignore logs

Code

printf "\*.log\nlog/\n" > .gitignore

git status —ignored

Output:

Untracked files:

.gitignore

Ignored files:

debug.log

log/

Q3. Commit ignore rules

Code

git add .gitignore

git commit -m "Ignore .log files and log/ folder”

Output:

[main b2c3d4e] Ignore .log files and log/ folder

1 file changed, 2 insertions(+)

create mode 100644 .gitignore

# Lab 3 — Branching & Basic Merge

Q1. Create and switch to feature branch

Code

git checkout -b GitNewBranch

Output:

Switched to a new branch ‘GitNewBranch'

Q2. Add work on branch and commit

Code

echo "Feature work" > feature.txt

git add feature.txt

git commit -m "Add feature work”

Output:

[GitNewBranch c3d4e5f] Add feature work

1 file changed, 1 insertion(+)

create mode 100644 feature.txt

Q3. Show branches & diff

Code

git branch

git diff main..GitNewBranch -- feature.txt

Output:

\* GitNewBranch

main

diff --git a/feature.txt b/feature.txt

+Feature work

Q4. Merge into main

Code

git checkout main

git merge GitNewBranch --no-ff -m "Merge branch 'GitNewBranch'"

git log --oneline --graph --decorate —all

Output:

Merge made by the 'ort' strategy.

\* d4e5f6a (HEAD -> main) Merge branch 'GitNewBranch'

|\

| \* c3d4e5f (GitNewBranch) Add feature work

* a1b2c3d Add welcome.txt

Q5. Delete merged branch

Code

git branch -d GitNewBranch

Output:

Deleted branch GitNewBranch (was c3d4e5f).

# Lab 4 — Merge Conflict & Resolution

Q1. Create branch and commit a file there

Code

git checkout -b GitWork

printf "Hello from branch\n" > hello.xml

git add hello.xml

git commit -m "Add hello.xml in branch”

[GitWork e5f6a7b] Add hello.xml in branch

1 file changed, 1 insertion(+)

create mode 100644 hello.xml

Q2. Create conflicting file on main

Code

git checkout main

printf "Different content on main\n" > hello.xml

git add hello.xml

git commit -m "Add hello.xml in main”

Output:

[main f6a7b8c] Add hello.xml in main

1 file changed, 1 insertion(+)

create mode 100644 hello.xml

Q3. Attempt merge → see conflict

Code

git merge GitWork

Output:

Auto-merging hello.xml

CONFLICT (add/add): Merge conflict in hello.xml

Automatic merge failed; fix conflicts and commit the result.

Q4. Resolve conflict & commit

Code

# choose final content

printf "Resolved content\n" > hello.xml

git add hello.xml

git commit -m "Resolve merge conflict in hello.xml”

Output:

[main a7b8c9d] Resolve merge conflict in hello.xml

Q5. Ignore backup files & commit

Code

echo "\*.orig" >> .gitignore

git add .gitignore

git commit -m "Ignore backup files”

Output:

[main b8c9d0e] Ignore backup files

1 file changed, 1 insertion(+)

Q6. Clean up branch

Code

git branch -d GitWork

git log --oneline --graph —decorate

Output:

Deleted branch GitWork (was e5f6a7b).

\* b8c9d0e (HEAD -> main) Ignore backup files

\* a7b8c9d Resolve merge conflict in hello.xml

* f6a7b8c Add hello.xml in main

# Lab 5 — Sync With Remote

Q1. Check status and branches

Code

git status

git branch -a

Output:

On branch main

nothing to commit, working tree clean

\* main

remotes/origin/main

Q2. Pull & push

Code

git pull origin main

git push origin main

Output:

Already up to date.

Everything up-to-date.