Git Hands-On Report

# Hands-On 1: Git Setup, Notepad++ Integration, and File Tracking

Objectives:

- Understand basic Git commands (git init, git status, git add, git commit, git push, git pull)

- Setup Git configuration on a local machine

- Integrate Notepad++ as the default Git editor

- Add files to a Git repository

Step-by-Step:

1. Check Git version:  
 git --version

2. Configure global Git user info:  
 git config --global user.name "Your Name"  
 git config --global user.email "you@example.com"

3. Verify config:  
 git config --list

4. Add Notepad++ to system path, test by:  
 notepad++

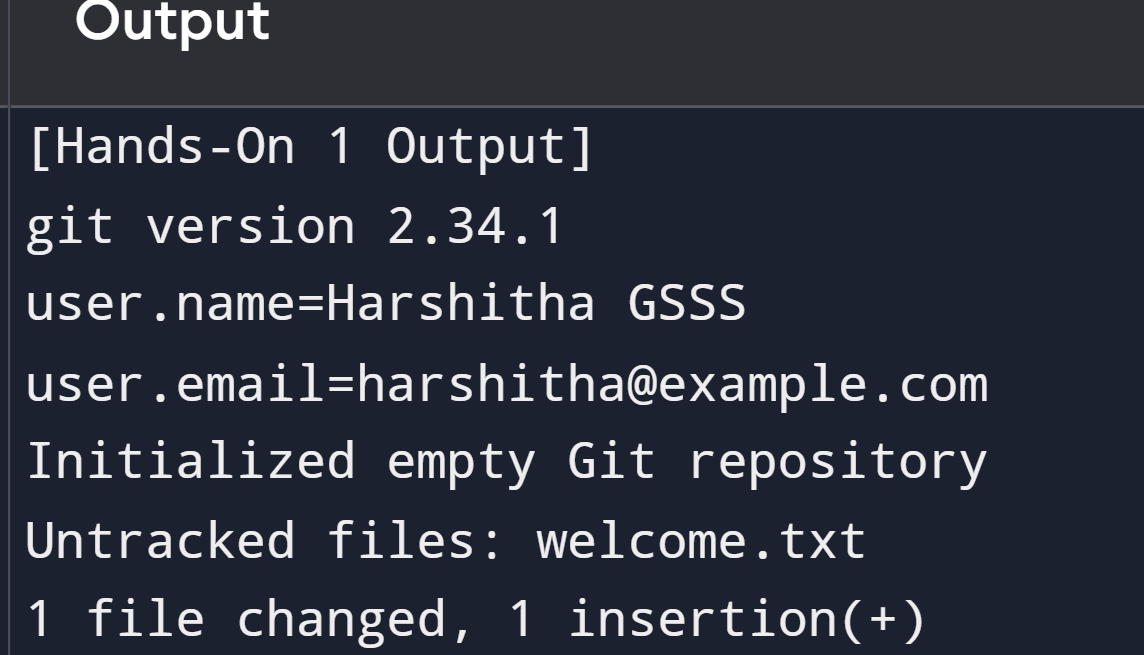
5. Set Notepad++ as default editor:  
 git config --global core.editor "notepad++ -multiInst -notabbar -nosession -noPlugin"

6. Verify editor:  
 git config --global -e

7. Create GitDemo repo:  
 mkdir GitDemo && cd GitDemo && git init

8. Create file:  
 echo "Welcome to Git" > welcome.txt

9. Add and commit file:  
 git add welcome.txt  
 git commit  
output:



# Hands-On 2: Git Ignore Usage

Objectives:

- Understand .gitignore and its use

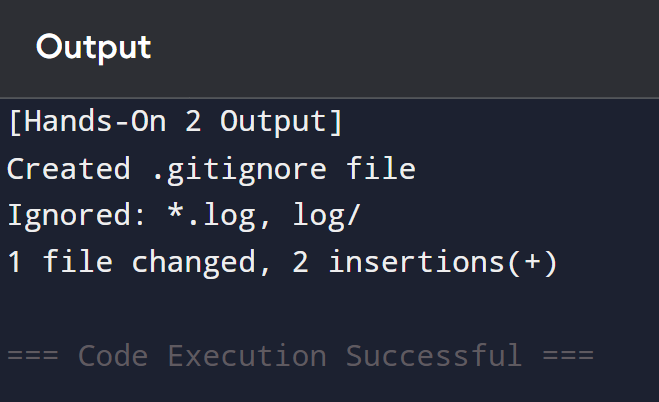
Step-by-Step:

1. Create files/folders:  
 touch error.log  
 mkdir log

2. Add to .gitignore:  
 echo "\*.log" > .gitignore  
 echo "log/" >> .gitignore

3. Check status:  
 git status

4. Add and commit .gitignore:  
 git add .gitignore  
 git commit -m "Ignore log files and folders"



# Hands-On 3: Branching and Merging

Objectives:

- Create branches, commit changes, merge with master

Step-by-Step:

1. Create and switch to branch:  
 git branch GitNewBranch  
 git checkout GitNewBranch

2. Add file and commit:  
 echo "This is a feature branch" > branch.txt  
 git add branch.txt  
 git commit -m "Feature added"

3. Switch to master:  
 git checkout master

4. Merge branch:  
 git merge GitNewBranch

5. View graph:  
 git log --oneline --graph --decorate

6. Delete branch:  
 git branch -d GitNewBranch

A screenshot of a computer program

AI-generated content may be incorrect.

# Hands-On 4: Conflict Resolution

Objectives:

- Understand and resolve Git conflicts

Step-by-Step:

1. Create and modify hello.xml in GitWork:  
 git checkout -b GitWork  
 echo "Branch version" > hello.xml  
 git add hello.xml  
 git commit -m "Branch changes"

2. Modify hello.xml in master:  
 git checkout master  
 echo "Master version" > hello.xml  
 git add hello.xml  
 git commit -m "Master changes"

3. Merge GitWork into master (conflict occurs):  
 git merge GitWork

4. Resolve manually or using tool:  
 git mergetool  
 git add hello.xml  
 git commit -m "Resolved conflict"

5. Ignore backup files:  
 echo "\*.orig" >> .gitignore  
 git add .gitignore  
 git commit -m "Ignore backup"

6. Delete GitWork:  
 git branch -d GitWork  
A screenshot of a computer

AI-generated content may be incorrect.

# Hands-On 5: Cleanup and Push to Remote

Objectives:

- Clean and push to remote Git

Step-by-Step:

1. Ensure clean master:  
 git status

2. List branches:  
 git branch -a

3. Pull from remote:  
 git pull origin master

4. Push changes:  
 git push origin master  
