# Git Hands-On Labs-WEEK 8

## Lab 1 — Setup and Basic Git Commands

Check Git installation:  
  
 git --version

Configure Git identity:  
  
 git config --global user.name "Your Name"  
 git config --global user.email "your.email@example.com"

Set VS Code as Git editor:  
  
 git config --global core.editor "code --wait"

Create a new local repo:  
  
 mkdir GitDemo  
 cd GitDemo  
 git init

Create and commit a file:  
  
 echo "Hello Git" > welcome.txt  
 git add welcome.txt  
 git commit -m "Add welcome.txt"

Push to remote:  
  
 git remote add origin https://github.com/<username>/GitDemo.git  
 git branch -M main  
 git push -u origin main

## Lab 2 — Using .gitignore

Create ignored files:  
  
 mkdir logs  
 touch error.log logs/debug.txt

Create .gitignore file with:  
  
 \*.log  
 logs/

Stage & commit:  
  
 git add .gitignore  
 git commit -m "Add .gitignore to ignore .log files and logs/ folder"

## Lab 3 — Branching and Merging

Create and switch to a branch:  
  
 git checkout -b GitNewBranch  
 echo "New branch content" > branch.txt  
 git add branch.txt  
 git commit -m "Add branch.txt in GitNewBranch"

Merge back to main:  
  
 git checkout main  
 git merge GitNewBranch  
 git branch -d GitNewBranch

## Lab 4 — Merge Conflict Resolution

Simulate a conflict:  
  
 git checkout -b GitWork  
 echo "Branch version" > hello.xml  
 git add hello.xml  
 git commit -m "Add hello.xml in branch"  
  
 git checkout main  
 echo "Main version" > hello.xml  
 git add hello.xml  
 git commit -m "Add hello.xml in main"

Merge (will cause conflict):  
  
 git merge GitWork

Resolve in VS Code and commit:  
  
 git add hello.xml  
 git commit -m "Resolve merge conflict in hello.xml"

## Lab 5 — Cleanup and Push

Delete merged branch:  
  
 git branch -d GitWork

Pull and push:  
  
 git pull origin main  
 git push origin main

