**WEEK 8 – ASSIGNMENT**

**GIT**

**Exercise 1: Login to GitLab and create a “GitDemo” project:**

//install git check version

git –version

//user details

git config --global user.name "github"

git config --global user.email [gitdemoemail@example.com](mailto:gitdemoemail@example.com)

//check for notepad

notepad++

//create and push a repository

mkdir GitDemo

cd GitDemo

//initialize git

git init

//create a file

echo "Welcome to Git Demo!" > welcome.txt

//check status

git status

//add to staging

git add welcome.txt

//save the file

git commit

//link local repo to remote

git remote add origin <https://github.com/username/GitDemo.git>

//pull from remote

git pull origin master

//push local repo to remote

git push origin master

**Exercise 2: Implement git ignore command to ignore unwanted files and folders Prerequisites**:

//create unwanted files

cd GitDemo

//create log folder

echo "This is a log file" > debug.log

mkdir log

echo "Log content" > log/app.log

//create .gitignore

notepad++ .gitignore

//verify

git status

//stage and commit

git add .gitignore

git commit -m "Add .gitignore to ignore log files and log folder"

//push the changes

git push origin master

//testing

echo "New log data" > test.log

git status

**Exercise 3: Construct a branch, do some changes in the branch, and merge it with master (or trunk):**

//create a new branch

git branch GitNewBranch

//list all branches

git branch -a

//switch to new branch

git checkout GitNewBranch

//add files

echo "This is a new branch file" > branchfile.txt

//stage and commit

git add branchfile.txt

git commit -m "Add file in GitNewBranch"

//status

git status

//swutch back to master

git checkout master

//view difference

git diff master GitNewBranch

//merge

git merge GitNewBranch

//merge history

git log --oneline --graph –decorate

//status

git status

//push

git push origin GitNewBranch

**Exercise 4: Implement conflict resolution when multiple users are updating the trunk (or master) in such a way that it results into a conflict with the branch’s modification.**

//verify master is clean

git checkout master

git status

//create a branch

git checkout -b GitWork

echo "<message>Hello from branch</message>" > hello.xml

//update

echo "<update>Branch edit</update>" >> hello.xml

git status

//commit

git add hello.xml

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

//switch back to master

git checkout master

//add files

echo "<message>Hello from master</message>" > hello.xml

//commit changes

git add hello.xml

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

//commit history

git log --oneline --graph --decorate –all

//differences

git diff master GitWork

//merge

git merge GitWork

//list

git branch

//delete merged branch

git branch -d GitWork

//final log

git log --oneline --graph –decorate

**Exercise 5: Execute steps involving clean up and push back to remote Git.**

//verify master is clean

git checkout master

git status

//list all branches

git branch -a

//pull changes

git pull origin master

//push changes

git push origin master