**Install GIT & make sure it is added into PATH.**

**Section 0 -Use GIT as local VCS. Steps to follow:**

1. **Create a directory ‘project\_dir’ & cd to ‘project\_dir’.**

mkdir project\_dir

cd project\_dir

1. **Initialize git version database. (git init)**

git init

C:/Users/BSAISREE/project\_dir/.git/

1. **Create a new file index.html.**

touch index.html

1. **Check the git status. You should find index.html as untracked file.**

git status

1. **Stage the index.html file.**

git add index.html

1. **Commit index.html**

git commit -m “first change”

1. **Make few changes in index.html & create a new file info.txt file.**

vi index.html

a tab is opened

click on I for insertion

make the changes that you want

then click on esc and type :wq!

touch info.txt

1. **Check git status. You should find index.html & info.txt as untracked files.**

git status

1. **Configure GIT to ignore all txt files.**

touch .gitignore

a file is created namely .gitignore on local repository

move info.txt to .gitignore

git add .gitignore

the file info.txt is ignored. Other files can be seen

1. **Again check the git status. You should find only index.html as untracked file.**

git status

1. **State & commit index.html**

git add index.html

git commit -m “second change”

1. **Log all your comments so far.**

git log

1. **Make some changes in index.html.**

vi index.html

a tab is opened

click on I for insertion

make the changes that you want

then click on esc and type :wq!

1. **Revert the change made in the previous step using git command.**

git checkout –index.html

1. **Again change index.html.**

vi index.html

a tab is opened

click on I for insertion

make the changes that you want

then click on esc and type :wq!

1. **Stage index.html**

git add index.html

1. **Revert back the last stage.**

git rm –cached index.html

1. **Rename ‘add’ command to ‘my-add’.**

git mv add my-add

git status

1. **Using my\_add command Stage index.html again & commit the changes.**

git add .

git commit -m “changes ”

1. **Revert the last commit.**

git reset –soft HEAD~1

*GIT Branching*

**Objective: Commit HTML, CSS & JavaScript assignments into GIT.**

**SECTION-1 (HTML assignments) - Steps to follow:**

1. **First take a backup of your assignments & projects. This is required because due to incorrect GIT operation you may lose your files.**

cp -r myrepo backup\_copy

1. **Create an empty directory ‘Assignments’ & cd to ‘Assignments’.**

mkdir Assignments

cd Assignments

1. **Create a file README.txt inside ‘Assignments’ & write few lines about the contents of ‘Assignments’ folder.**

touch README.txt

vi Assignments

a tab is opened

click on I for insertion

write few lines that you want

then click on esc and type :wq!

1. **Commit README.txt file.**

git commit -m “Adding new file README.txt”

1. **Now create a new branch ‘html-assignments’.**

git branch ‘html-assignments’

1. **Switch to ‘html-assignments’ branch.**

git checkout html-assignments

1. **Copy all HTML assignments inside ‘Assignments’ folder.**

Created 2 new files namely one.html and two.html inside Assignments folder

1. **Commit HTML assignments into ‘html-assignments’ branch.**

git add one.html

git add two.html

git commit -m “committing one and two in branch”

1. **Make minor changes into few files belonging to ‘html-assignments’ branch.**

vi one.html

a tab is opened

click on I for insertion

make changes that you want

then click on esc and type :wq!

1. **Commit those changed files.**

git add one.html

git commit -m “changes in one”

1. **Switch to master branch.**

git checkout master

1. **Make minor changes into README.txt file & commit those changes into master.**

vi README.txt

a tab is opened

click on I for insertion

make changes that you want

then click on esc and type :wq!

git add README.txt

git commit -m “changes in README.txt”

1. **Again switch to ‘html-assignments’ branch.**

git checkout html-assignments

1. **Make minor changes into few files belonging to ‘html-assignments’ branch.**

vi two.html

a tab is opened

click on I for insertion

make changes that you want

then click on esc and type :wq!

1. **Commit those changes.**

git add two.html

git commit -m “changes in two”

1. **Switch to master.**

git checkout master

1. **Merge ‘html-assignments’ branch into master. Confirm all html assignments are shown in master.**

git merge html-assignments

git branch -merge

git diff master..html-assignments

1. **Finally delete the ‘html-assignments’ branch.**

git branch -d html-assignments

**SECTION-2 - (CSS assignments) Steps to follow:**

1. **Create a new branch ‘css-assignments’.**

git branch css-assignments

1. **Switch to ‘css-assignments’ branch.**

git checkout css-assignments

1. **Copy all CSS assignments inside ‘Assignments’ folder.**

Created 2 new files namely one.css and two.css in master branch

1. **Commit CSS assignments into ‘css-assignments’ branch.**

git add one.css

git add two.css

git commit -m “Committing one and two in branch”

1. Make minor changes into README.txt file on line 1 belonging to ‘css-assignments’ branch.

vi README.txt

a tab is opened

click on I for insertion

make changes in line 1

then click on esc and type :wq!

1. **Commit those changed files.**

git add README.txt

git commit -m ”Changes in README.txt”

1. **Switch to master branch.**

git checkout master

1. **Make minor changes into README.txt file on line 3 & commit those changes into master.**

vi README.txt

a tab is opened

click on I for insertion

make changes in line 3

then click on esc and type :wq!

git add README.txt

git commit -m “other change in README.txt”

1. **Again switch to ‘css-assignments’ branch.**

git checkout css-assignments

1. **Make minor changes into few files belonging to ‘css-assignments’ branch.**

vi two.css

a tab is opened

click on I for insertion

make changes that you want

then click on esc and type :wq!

1. **Commit those changes.**

git add two.css

git commit -m “change in two.css”

1. **Switch to master.**

git checkout master

1. **Merge ‘css-assignments’ branch into master. Confirm all css assignments are shown in master.**

git merge css-assignmets

git diff master..css-assignments

1. **Finally delete the ‘css-assignments’ branch.**

git branch -d css-assignments

**SECTION-3 - (JavaScript assignments) Steps to follow:**

1. **Create a new branch ‘js-assignments’.**

git branch js-assignments

1. **Switch to ‘js-assignments’ branch.**

git checkout js\_assignments

1. **Copy all JavaScript assignments inside ‘Assignments’ folder.**

Created 2 files namely one.js and two .js in the master branch

1. **Commit JavaScript assignments into ‘js-assignments’ branch.**

git add one.js

git add two.js

git commit -m “committing one and two in branch”

1. **Make minor changes into README.txt file on line 1 belonging to ‘js-assignments’ branch.**

vi README.txt

a tab is opened

click on I for insertion

make changes in line 1

then click on esc and type :wq!

1. **Commit those changed files.**

git add README.txt

git commit -m “changes in README.txt”

1. **Switch to master branch.**

git checkout master

1. **Make minor changes into README.txt file on line 1 & commit those changes into master.**

vi README.txt

a tab is opened

click on I for insertion

make changes in line 1

then click on esc and type :wq!

git add README.txt

git commit -m “changes in README.txt”

1. **Again switch to ‘js-assignments’ branch.**

git checkout js-assignments

1. **Make minor changes into few files belonging to ‘js-assignments’ branch.**

vi two.js

a tab is opened

click on I for insertion

make changes that you want

then click on esc and type :wq!

1. **Commit those changes.**

git add assign2.js

git commit -m “Edited assign2.js”

1. **Switch to master.**

git checkout master

1. **Merge ‘js-assignments’ branch into master. Confirm all JavaScript assignments are shown in master.**

git merge js-assignments

git diff master..js-assignments

1. **Finally delete the ‘js-assignments’ branch.**

git branch -d js-assignments

*GIT Remoting*

**Objective: Pushing source code into GITHUB & collaborate team members.**

**SECTION-3 (Pushing assignments to remote repository) - Steps to follow:**

1. **Create a github account if you do not have already.**

Saisreeja30

1. **Login on into github account.**

Done

1. **Create new public repository ‘freshersbatch-oct16’.**

Freshersbatch-dec21

1. **Commit & push any sample file to this repository under ‘Assignments’ directory.**

git remote add origin <https://github.com/saisreeja30/freshersbatch_dec21.git>

git push origin master

**SECTION-4 (Pushing source code to remote repository using Eclipse GIT plugin) - Steps to follow:**

1. **One developer from project team will create eclipse projects ‘SampleProj’ & add sample source code files. Then commit all files through eclipse GIT plugin.**

One of our team member downloaded eclipse and created a project ‘SampleProj’ and added source code files named as sp1.html, sp2.css and sp3.js. She committed the code in the eclipse and pushed it in her GitHub account.

1. **Collaborate other team members with your github account so that they can also modify the committed files.**

She collaborated with us by adding our GitHub accounts in the project. I was able to make changes in the files >> Adding and Deleting lines >> Committing them.

1. **Other developers from same team will checkout all files from remote repository. This might get conflicts since certain files fail to merge. In such case, merge it manually.**
2. **Commit & push the ‘SampleProj’ project.**

After committing I was able to push the changes that I made by clicking on the “Commit” button. Other collaborators were able to see the changes that I made and also I was able to see the changes that they made after committing and pushing the files.