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

1. Initialize git version database. (git init)

=git init

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 “committed index.html file”

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

= 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

(\*.txt file)

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

=git status

1. State & commit index.html

=Git status

git add index.html

1. Log all your comments so far.

=git log

1. Make some changes in index.html.

==

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

=git checkout index.html

1. Again change index.html.

==

1. Stage index.html

Git add index.html

1. Revert back the last stage.

Git checkout index.html

1. Rename ‘add’ command to ‘my-add’.

=git config --global alias.my-add add

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

=git my\_add index.html

Git commit -m “committed the changes”

1. Revert the last commit.

=git revert committed the changes

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

= mkdir Assigments

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

= touch README.txt

1. Commit README.txt file.

=Git add README.txt

Git commit -m “committed README.txt”

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

Git branch ‘html-assignments’

1. Switch to ‘html-assignments’ branch.

Git checkout html-assigments

1. Copy all HTML assignments inside ‘Assignments’ folder.
2. Commit HTML assignments into ‘html-assignments’ branch.

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

==

1. Commit those changed files.

Git Commit -m “commited”

1. Switch to master branch.

Git checkout Master

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

==

Git commit -m “files change”

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

Git checkout html-assigments

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

==

1. Commit those changes.

Git commit -m “changed to ”

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

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

Git rm html-assignments

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

1. Create a new branch ‘css-assignments’.
2. Switch to ‘css-assignments’ branch.
3. Copy all CSS assignments inside ‘Assignments’ folder.
4. Commit CSS assignments into ‘css-assignments’ branch.
5. Make minor changes into README.txt file on line 1 belonging to ‘css-assignments’ branch.
6. Commit those changed files.
7. Switch to master branch.
8. Make minor changes into README.txt file on line 3 & commit those changes into master.
9. Again switch to ‘css-assignments’ branch.
10. Make minor changes into few files belonging to ‘css-assignments’ branch.
11. Commit those changes.
12. Switch to master.
13. Merge ‘css-assignments’ branch into master. Confirm all css assignments are shown in master.
14. Finally delete the ‘css-assignments’ branch.

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

1. Create a new branch ‘js-assignments’.
2. Switch to ‘js-assignments’ branch.
3. Copy all JavaScript assignments inside ‘Assignments’ folder.
4. Commit JavaScript assignments into ‘js-assignments’ branch.
5. Make minor changes into README.txt file on line 1 belonging to ‘js-assignments’ branch.
6. Commit those changed files.
7. Switch to master branch.
8. Make minor changes into README.txt file on line 1 & commit those changes into master.
9. Again switch to ‘js-assignments’ branch.
10. Make minor changes into few files belonging to ‘js-assignments’ branch.
11. Commit those changes.
12. Switch to master.
13. Merge ‘js-assignments’ branch into master. Confirm all JavaScript assignments are shown in master.
14. Finally delete the ‘js-assignments’ branch.

*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.

A:search github.com and register using credentials

1. Login on into github account.

A: provide Registered credentials

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

A: Select NEW-> Repository Name-> select public option -> click on create repository

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

Unstaged changes – put in stage changes – write commit msg – commit and push option-select folder - destination git repository window is open click on preview-push to branch in remote window is open click on preview- push confirmation window will open click on push option- pushed to file window will open click on close .

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.

A: elipse: - open eclipse – click on files new - java project – proect name (SampleProj)- click on files – new – class- create a class with index.java then follow the steps 40, 41, 42, from section 3

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

A: go to repository – click on settings – manage access – password- add people – add email id of other team mates- give them access – after accepting invite – they are able to perform changes.

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.

* Go to that project which is present on repository – click on edit (icon pencil)- do some changes

1. Commit & push the ‘SampleProj’ project.

A : at the end there is one option commit changes click on that – write some comment and commit changes .