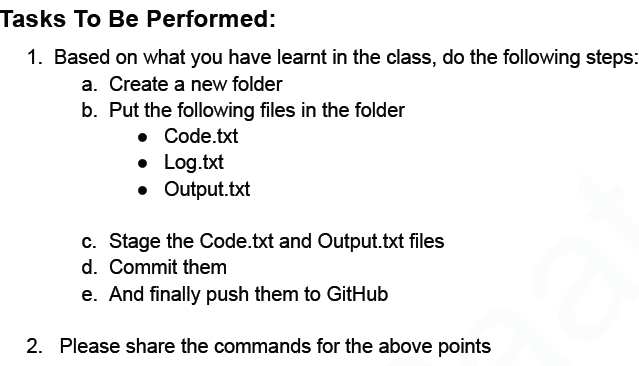
**Git Assignment**

**Task 1**

****

1. Create a New Folder :

mkdir foldername  
  
2. Put the following files in the folder

● Code.txt

● Log.txt

● Output.txt

cd foldername (To change the Working Directory)

git init (Initialize Git in the Folder)

touch code.txt log.txt output.txt

1. Stage the Code.txt and Output.txt files

git add . (Stages all the 3 Files since . means current directory, alternative you can mention the file names with space)

git status (To check if we have missed any staging of the file)

git remote add origin <url> (URL would be of the Repository you want to push the codes on)

1. Commit them

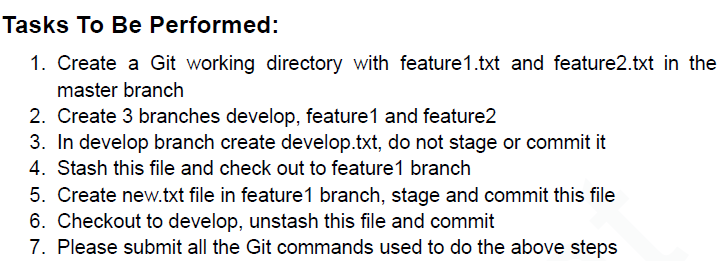
git commit -m “Commit Message” (Here, -m Option signifies the Message to be passed commiting. The message should be enclosed within the inverted commas)

1. And finally push them to GitHub

git push origin <branch> (Pushes the commited code to online repo, you have to specifiy master if default branch)

**Git Assignment**

**Task 2**

****

1. Create a Git working directory with feature1.txt and feature2.txt in the master branch

mkdir task2 (Creates Directory with name : task2)

touch feature1.txt feature2.txt

git add .

git remote add origin <repo URL>

git commit -m “Commit Message”

Unless and until you don’t commit, you cannot create or view any branch. Hence commited.

1. Create 3 branches develop, feature1 and feature2

git branch develop (Creates branch)

git branch feature1

git branch feature2

1. In develop branch create develop.txt, do not stage or commit it

git checkout develop (Changes branch from master to develop)

touch develop.txt

1. Stash this file and check out to feature1 branch

git add .

git stash push develop.txt

1. Create new.txt file in feature1 branch, stage and commit this file

touch new.txt

git add new.txt

git commit -m “Committed from Feature1”

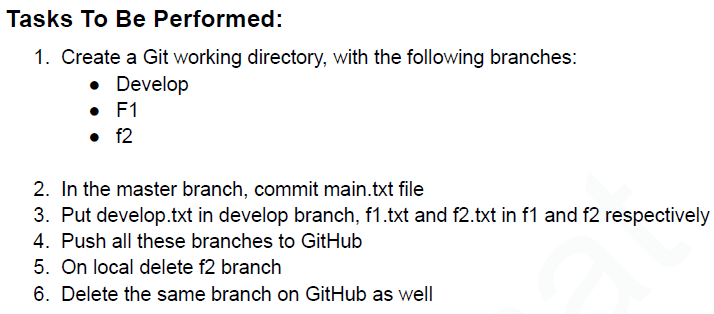
1. Checkout to develop, unstash this file and commit

git stash pop

git commit -m “Commited from Develop”

**Git Assignment**

**Task 3**

****

1. Create a Git working directory, with the following branches:

● Develop

● F1

● f2

mkdir dirname (Creates Directory)

git branch develop (Creates Branch)

git branch f1

git branch f2

1. In the master branch, commit main.txt file

touch main.txt

git add main.txt

git commit -m “Commited Main.txt” (Commits the main.txt File Locally”)

1. Put develop.txt in develop branch, f1.txt and f2.txt in f1 and f2 respectively

git checkout develop (Switches the branch to develop)

touch develop.txt

git add develop.txt

git commit -m “develop.txt committed”

git checkout f1

touch f1.txt

git add f1.txt

git commit -m “f1.txt Committed”

git checkout f2

touch f2.txt

git add f2.txt

git commit -m “f2.txt Committed”

1. Push all these branches to GitHub

git push origin develop (Pushes the branch develop to remote repo)

git push origin f1

git push origin f2

1. On local delete f2 branch

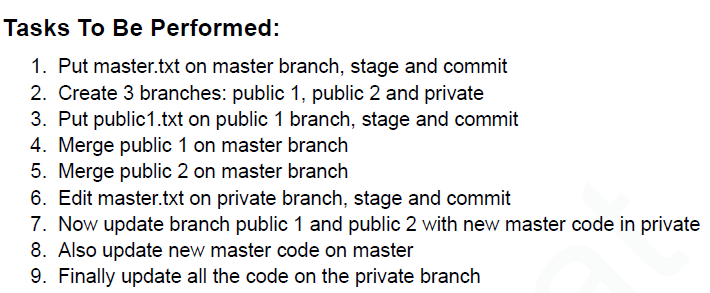
git branch -D f2 (Deletes branch f2 Locally)

1. Delete the same branch on GitHub as well

git push origin --delete f2 (Deletes branch f2 on Git Remote Repo)

**Git Assignment**

**Task 4**

****

1. Put master.txt on master branch, stage and commit

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git init

Reinitialized existing Git repository in C:/Users/adnan/Tasks/.git/

umesh@iceweasel MINGW64 ~/Tasks (master)

$ echo "Dummy Master Text" >> master.txt

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git add .

warning: in the working copy of 'master.txt', LF will be replaced by CRLF the next time Git touches it

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git commit -m "master.txt committed"

[master 50f6b26] master.txt committed

1 file changed, 1 insertion(+)

create mode 100644 master.txt

1. Create 3 branches: public 1, public 2 and private

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch public1

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch public2

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch private

umesh@iceweasel MINGW64 ~/Tasks (master)

$

1. Put public1.txt on public 1 branch, stage and commit

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git checkout public1

Switched to branch 'public1'

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ echo "//Dummy text on public1.txt" >> public1.txt

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ git add public1.txt

warning: in the working copy of 'public1.txt', LF will be replaced by CRLF the next time Git touches it

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ git commit -m "Public1.txt Committed!"

[public1 f67dae0] Public1.txt Committed!

1 file changed, 1 insertion(+)

create mode 100644 public1.txt

umesh@iceweasel MINGW64 ~/Tasks (public1)

$

1. Merge public 1 on master branch

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ git checkout master

Switched to branch 'master'

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git merge public1

Updating 50f6b26..f67dae0

Fast-forward

public1.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 public1.txt

1. Merge public 2 on master branch

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git merge public2

Already up to date.

1. Edit master.txt on private branch, stage and commit

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git checkout private

Switched to branch 'private'

umesh@iceweasel MINGW64 ~/Tasks (private)

$ cat master.txt

Dummy Master Text

umesh@iceweasel MINGW64 ~/Tasks (private)

$ echo "Edited on Private Branch!" >> master.txt

umesh@iceweasel MINGW64 ~/Tasks (private)

$ cat master.txt

Dummy Master Text

Edited on Private Branch!

umesh@iceweasel MINGW64 ~/Tasks (private)

$ git add master.txt

warning: in the working copy of 'master.txt', LF will be replaced by CRLF the next time Git touches it

umesh@iceweasel MINGW64 ~/Tasks (private)

$ git commit -m "Committed Edited Master.txt from Private Branch!"

[private 8dda544] Committed Edited Master.txt from Private Branch!

1 file changed, 1 insertion(+)

umesh@iceweasel MINGW64 ~/Tasks (private)

$

1. Now update branch public 1 and public 2 with new master code in private

umesh@iceweasel MINGW64 ~/Tasks (private)

$ git checkout public1

Switched to branch 'public1'

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ git merge private

Merge made by the 'ort' strategy.

master.txt | 1 +

1 file changed, 1 insertion(+)

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ cat master.txt

Dummy Master Text

Edited on Private Branch!

umesh@iceweasel MINGW64 ~/Tasks (public1)

$ git checkout public2

Switched to branch 'public2'

umesh@iceweasel MINGW64 ~/Tasks (public2)

$ git merge private

Updating 50f6b26..8dda544

Fast-forward

master.txt | 1 +

1 file changed, 1 insertion(+)

umesh@iceweasel MINGW64 ~/Tasks (public2)

$ cat master.txt

Dummy Master Text

Edited on Private Branch!

umesh@iceweasel MINGW64 ~/Tasks (public2)

$

1. Also update new master code on master

umesh@iceweasel MINGW64 ~/Tasks (public2)

$ git checkout master

Switched to branch 'master'

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git merge private

Merge made by the 'ort' strategy.

master.txt | 1 +

1 file changed, 1 insertion(+)

umesh@iceweasel MINGW64 ~/Tasks (master)

$ cat master.txt

Dummy Master Text

Edited on Private Branch!

umesh@iceweasel MINGW64 ~/Tasks (master)

$

1. Finally update all the code on the private branch

umesh@iceweasel MINGW64 ~/Tasks (private)

$ git merge master

Updating 8dda544..c7c9389

Fast-forward

public1.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 public1.txt

umesh@iceweasel MINGW64 ~/Tasks (private)

$ git merge public1

Merge made by the 'ort' strategy.

umesh@iceweasel MINGW64 ~/Tasks (private)

$ git merge public2

Already up to date.

umesh@iceweasel MINGW64 ~/Tasks (private)

$ ls

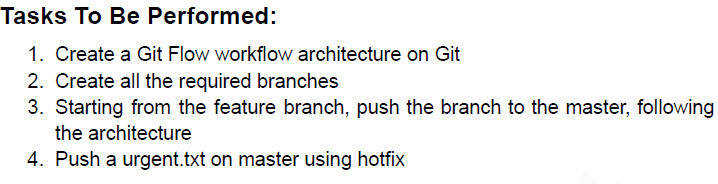
master.txt public1.txt

umesh@iceweasel MINGW64 ~/Tasks (private)

$

**Git Assignment**

**Task 5**

****

1. Create a Git Flow workflow architecture on Git

Git Flow is a branching model that defines a set of branches with specific purposes. It typically consists of the following branches:

master: Represents production-ready code.

develop: Contains the latest features and bug fixes for the next release.

feature: Feature branches where you develop new features.

hotfix: Branches to fix critical issues in production.

release: Branches to prepare for a new release.

1. Create all the required branches

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch develop

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch feature

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch hotfix

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch release

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git branch

develop

feature

hotfix

\* master

release

3. Starting from the feature branch, push the branch to the master, following

the architecture

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git checkout feature

Switched to branch 'feature'

umesh@iceweasel MINGW64 ~/Tasks (feature)

$ echo "Feature 1 Added!" >> feature1.txt

umesh@iceweasel MINGW64 ~/Tasks (feature)

$ git add feature1.txt

warning: in the working copy of 'feature1.txt', LF will be replaced by CRLF the next time Git touches it

umesh@iceweasel MINGW64 ~/Tasks (feature)

$ git commit -m "Feature1.txt Committed!"

[feature 0331d94] Feature1.txt Committed!

1 file changed, 1 insertion(+)

create mode 100644 feature1.txt #Feature is UP and Ready to test in dev

umesh@iceweasel MINGW64 ~/Tasks (feature)

$ git checkout develop

Switched to branch 'develop'

umesh@iceweasel MINGW64 ~/Tasks (develop)

$ git merge feature

Updating 318c591..0331d94

Fast-forward

feature1.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 feature1.txt

umesh@iceweasel MINGW64 ~/Tasks (develop)

$ git push origin develop #Once tested, push it to remote repo

Enumerating objects: 16, done.

Counting objects: 100% (16/16), done.

Delta compression using up to 8 threads

Compressing objects: 100% (8/8), done.

Writing objects: 100% (15/15), 1.40 KiB | 717.00 KiB/s, done.

Total 15 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (1/1), done.

remote:

remote: Create a pull request for 'develop' on GitHub by visiting:

remote: https://github.com/umeshkuduwa/GitAssignment/pull/new/develop

remote:

To https://github.com/umeshkuduwa/GitAssignment.git

\* [new branch] develop -> develop

umesh@iceweasel MINGW64 ~/Tasks (develop)

$ git checkout master

Switched to branch 'master'

g

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git merge develop #Merge develop into master (represents Release)

Updating 318c591..0331d94

Fast-forward

feature1.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 feature1.txt

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git push origin master #Release

Total 0 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/umeshkuduwa/GitAssignment.git

7d29834..0331d94 master -> master

4. Push a urgent.txt on master using hotfix

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git checkout hotfix

Switched to branch 'hotfix'

umesh@iceweasel MINGW64 ~/Tasks (hotfix)

$ echo "Urgent FIX!" >> urgent.txt

umesh@iceweasel MINGW64 ~/Tasks (hotfix)

$ git add urgent.txt

warning: in the working copy of 'urgent.txt', LF will be replaced by CRLF the next time Git touches it

umesh@iceweasel MINGW64 ~/Tasks (hotfix)

$ git commit -m "Urgent Fix Added!"

[hotfix 0458c2e] Urgent Fix Added!

1 file changed, 1 insertion(+)

create mode 100644 urgent.txt

umesh@iceweasel MINGW64 ~/Tasks (hotfix)

$ git checkout master

Switched to branch 'master'

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git merge hotfix

Updating 99ad508..0458c2e

Fast-forward

urgent.txt | 1 +

1 file changed, 1 insertion(+)

create mode 100644 urgent.txt

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git status

On branch master

nothing to commit, working tree clean

umesh@iceweasel MINGW64 ~/Tasks (master)

$ git push origin master

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 8 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 294 bytes | 294.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/umeshkuduwa/GitAssignment.git

99ad508..0458c2e master -> master

umesh@iceweasel MINGW64 ~/Tasks (master)

$