CS 4743
Applied Software Engineering
Spring 2015

Assignment 2: Git

This assignment involves using Git in a way that closer resembles its use in industry. You (and your partner) will execute the demoproject that I conduct in class by following my demo script. This is to acquaint you with Git branching and merging. Next, you will put your Assignment 1 project under Git version control and make some changes to it (add some features, modify a feature, etc.). But, each team member can only perform one half of the changes and the changes must occur in a predetermined sequence (see below).

Students I have asked dread merge conflicts and that is only because we don't teach you everything you need to handle them well. That's the goal of this lab. Merge conflicts are completely normal and you should never have to nuke your repo. Deciding what to keep or move in a merge involves work. But the work is tractable is if you reduce the # of merge decision makers to a single developer, i.e., the branch owner.

This is not a complex assignment but it may be difficult for those working alone or for those who become easily frustrated. This assignment builds on Assignment 1 so you really should complete it first before attempting this one. If you are working alone, you can create a new Eclipse workspace and act as if that is your partner's workspace (keep it in a totally separate folder on your computer or V drive). Then, just make the changes according to the task sequence below.

One nice thing about this assignment is that once you reach Task #2, you will freeze your completed Assignment 1 as the first commit to master. You should always be able to restart dev work from there, if necessary. If you must nuke your remote, so be it, but I may add you to the Wall of Shame.

Task (2 parts):
*** gitdemo part: aka the easy part***
1. Follow gitdemo script and save the final **gitdemo directory and local repo** in a single zip file (or Eclipse export) to submit.

*** Inventory Parts program part: aka the other easy part***
2. Put your Inventory Parts project under Git version control (create the local and remote repos, make an initial commit, and push it to the remote in the master branch). If you need me to make remote repos for you in Github, let me know.
3. One team member is designated as the master owner. Only the master owner is allowed to merge dev branches into master.
4. One team member COMPLETELY implements **AND TESTS** Change Request 1 in its own branch

5. The other team member COMPLETELY implements **AND TESTS** Change Request 2 in its own branch
6. **AFTER** both dev branches have been pushed, **the master owner merges** the dev branches into master **AND TESTS** (be sure to push the merge result back to the remote)
8. Check the remote repo to make sure all dev branches are merged
9. The other team member (the one who is not the master owner) should pull master so that it is up-to-date. Now switch the master owner to the other team member.
10. One team member COMPLETELY implements **AND TESTS** Change Request 3 in its own branch
11. The other team member COMPLETELY implements **AND TESTS** Change Request 4 in its own branch
12. **AFTER** both dev branches have been pushed, **the master owner merges** the dev branches into master **AND TESTS** (be sure to push the merge result back to the remote)
13. Check the remote repo to make sure all dev branches are merged
14. Zip up the full project folder and local git repo in a single zip file (or Eclipse export) to submit.

Change Requests:

1. Implement an ID field in Part (and all necessary MVC components)

    Required
    unsigned int or UUID (you get to choose)
    Must be unique
    Automatically assigned and incrementing (first part that is added should have ID 1)
    *** Some ways to implement are using an auto-incrementing static member or UUID

2. Implement a Unit of Quantity field in Part (and all necessary MVC components)

    Required, default to “Unknown” but cannot be saved as “Unknown”
    Must show on the Parts List ALONG WITH Quantity
    Values are in set {“Linear Feet”, “Pieces”, “Unknown”}

3. Implement an External Part # field in Part (and all necessary MVC components)

    Optional
    Values are alphanumeric and symbols, max length 50

4. Implement a Location field in Part (and all necessary MVC components)

    Required, default to “Unknown” but cannot be saved as “Unknown”
    Values are in set {“Facility 1 Warehouse 1”, “Facility 1 Warehouse 2”, “Facility 2”, “Unknown”}

Note: all fields must be visible in the Part Detail and all except ID must be editable. Drop-down lists are recommend for the value-restricted fields.

Deliverables:

1. Zip up your finished gitdemo directory, including your local repo (the .git folder)

2. Zip everything up into a single zip file and submit it to Blackboard.
*** Be sure to identify team members for Assignment 2 in either your project's main class in a comment OR you can add the comment when submitting.
*** Both team member's should submit the same project separately

Rubric:

20 pts	gitdemo submitted and git graph looks right
10 pts	Change Request 1 implemented correctly
10 pts	Change Request 1 dev branch merged correctly
10 pts	Change Request 2 implemented correctly
10 pts	Change Request 2 dev branch merged correctly
10 pts	Change Request 3 implemented correctly
10 pts	Change Request 3 dev branch merged correctly
10 pts	Change Request 4 implemented correctly
10 pts	Change Request 4 dev branch merged correctly