Homework/Project Submission with Git and Bitbucket

NGUYEN Thi Thu Trang, trangntt@soict.hust.edu.vn

1. **TIMELINE & DEADLINE**
    a. Homework:
        i. *12AM of the day before the next lesson of the assigned week*
        ii. *E.g. If the assignment was made on Monday (17th of September, 2018), you need to submit your results before 12AM on next Sunday (19th of September, 2018).*
    b. Project:
        i. Final submission: 14th of December (Week 16)
        ii. Final Presentation: 17th, 24th of December (Week 17, 18)
2. **REQUIREMENTS**
    a. Each homework/project, create a folder in the repository of your group. Put all resources of the homework/project to this folder. The folder can be one of the followings:
        i. VersionControl
        ii. RequirementAnalysis
        iii. ArchitureDesign
        iv. DetailDesign
        v. Programming
        vi. UnitTest
        vii. Project
    b. Each member creates their own branch (his/her name). Each member pushes his/her result to his/her branch. Before the deadline (at least 30 minutes), the team leader (or members) merges it to the master to a full version of a resource.
    c. **readme.md:** In each homework/project, you must have a **readme.md** file. This file can be created by a member, but should be updated by all the members for their assignment status.
        i. Description of the homework/project
        ii. Assignment of each member (WHO is the doer for WHAT tasks), status of assignment (e.g. 0%-30%-100%).
        iii. Reviewer for each assignment in your group: WHO reviews WHAT for WHOM?
    d. **review.txt**: This file contains the reviews for the assignment, including (i) WHO reviews WHAT for WHO; (ii) Any Errors, Comments, Proposals for Corrections
        i. All members need to complete and push their results to their branches at least 48 hours before the deadline.
        ii. The reviewer checks and submits their review at least 24 hours before the deadline so that the doer can correct/update his/her result.
        iii. The doer need to commit the final version to his/her branch, and then push to the remote repository at least 1 hour before the deadline.
3. **VERSION CONTROL with Git and Bitbucket**
    a. Using Bitbucket
        i. Each member creates an account in bitbucket.org

ii. One member creates a repository naming “YourClass-YourGroupNo.” (TKXDPM.20181-XY), for example TKXDPM.20181-05, then adds all your member accounts to that repository, plus the account “trangntt-student”. If you don’t follow this naming convention, your repository will be ignored.

b. Using IDE
   i. Install IDE e.g. Eclipse or Netbean
   ii. Install Git (and/or Source Tree, eGit for Eclipse…)

c. Using Git commands
   i. Learn Git commands in https://git-scm.com/book/en/v2
   ii. Practice with your homework and your team.

Some common scenarios and commands:

* Clone a repository, create a branch:

```
cd <local_working_folder>
git clone <repository_url>
cd <repository_name> /* i.e. YourClass-YourGroupNo. */
git checkout -b <your_branch> /* create and switch to your new branch */
```

Then you can make any changes such as create a new file, modify an existing file, delete a file…

* Commit and push after changing some resources:

```
cd <local_working_folder/repository_name>
git add -A /* -A: for all operations or . for without deletions in the current folder */
git status /* check status of the stage */
git commit -m "Your comment for the update" /* commit with comment*/
git push origin <your_branch> /* push from local repo to remote repo */
```

Go to the Bitbucket to *Create merge request*, then *Approve merge request*.

* Pull without conflicts

```
git stash /* run this command if you want to ignore your current change */
git pull origin <your_branch>
```

* Pull and resolve conflicts

After you commit and push to the remote repository, if there is any conflict when creating merge request, you need to resolve conflicts.

```
git checkout master
git pull origin master
git checkout <your_branch>
git rebase master
git add -A
git status
git rebase –continue
```

You need to resolve conflicts by checking the resources that have conflicts. There are both versions in the resources so that you can observe and make decisions, e.g.

```
<<<<<<< HEAD /* master branch in the remote repository */
      suggestor.setClientId(clientID);
```

```
      suggestion = suggestor.translate(input).replaceAll("-", " ");
=======  /* your branch in the local repository */
      suggestion = suggestor.translate(input);
>>>>>>> Your comment for the update
```

Then continue, force to push the merged version to the remote repository
`git push origin <your_branch> -f`

d. Note that you should practice following activities during your team working with Eclipse and Bitbucket
   i. Create Team (same as the repository)
   ii. Create Branch for each member in your group
   iii. Pull changes / resolve conflict from a remote repository
   iv. Create a branch and make change
   v. Commit, merge your branch
   vi. Push your change to Bitbucket

4. REFERENCES

a. Git tutorial:
   i. https://git-scm.com/docs/gittutorial
   ii. https://www.atlassian.com/git/tutorials
   iii. https://www.youtube.com/watch?v=HVsySz-h9r4&list=PL-osiE80TeTuRUfjRe54Eea17-YfnOOAx

b. Bitbucket tutorial:
   i. https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud
   ii. https://bitbucket.org/tutorials/bucket-o-sand