Lab1: Tools for collaborative software development

Git & GitHub

**Goal:** After this lab session, the students should be able to use Git and GitHub for their teamwork in this course and other projects. Students should know about the Gitflow workflow and how to collaborate with other developers in their projects. Students should also have a basic idea about the Education Pathways code base.

**Plan**: Students watch the [recording on introduction to GitFlow](https://utoronto-my.sharepoint.com/:v:/g/personal/shurui_zhou_utoronto_ca/Edx2AzANZ5pKtkOOLcoidPQBoCqJrfWay7DHiWyMf4GCtA?e=OUwI6C) (slides can be found [here](https://q.utoronto.ca/files/22055175?wrap=1) on Quercus), then Q&A.

**Requirement & Evaluation**: Students should follow the 8 Activities. Once you are done, please submit your GitHub repo url in Assignments/Lab1 on Quercus. Your grade is depending on the completeness of the task.

**Submission:**

Submit the link to your lab1 repository created in Activity1 on Quercus. Remember to make sure that all the repositories created in this lab are public.

Refer to [this README file](https://github.com/ECE444-2022Fall/Lab1-sample-submission) as an example of what your lab1 repository should look like. If you do not know how to insert images on GitHub, please follow what the sample repo does.

**References:**

* [Introducing Gitflow](https://datasift.github.io/gitflow/IntroducingGitFlow.html)
* [Github docs](https://docs.github.com/en/get-started/quickstart/hello-world)
* [Understanding Git merge & Git rebase](https://medium.com/mindorks/understanding-git-merge-git-rebase-88e2afd42671)
* [Git rebase](https://www.atlassian.com/git/tutorials/rewriting-history/git-rebase)
* [Git rebase docmentation](https://git-scm.com/docs/git-rebase)

**Activity 0: Sign up**

* Create a [GitHub](https://github.com/) account.

**Activity 1 (1 point): Creating a repo in your own GitHub account and committing files**

* Create an empty public repository named ECE444-F2022-Lab1 (referred as **Lab1 repo** later).
* Clone the repository to your machine.
* Create a README.md file with your name.
* Add and commit your changes. Push the them to GitHub.
* Add a **screenshot** for the commit on GitHub in the README file. Commit and push the changes to README.

**Activity 2 (1 point): Branching and merging**

* Create a new branch called “develop” in your local repository.

git checkout -b develop

* Push the new branch to GitHub.

git push -u origin develop

* Create a python program that, when run on the “develop” branch, will print ‘Hello World’ to the terminal, name this file “helloworld.py”. Commit the change.
* Switch back to the “main” branch.
* Merge the changes from the “develop” branch into the “main” branch with “git merge development”.
* Add a **screenshot** for the output of the merge command on the “main” branch. Commit and push the changes.

**Activity 3 (1 point): Issues, pull requests and merge conflicts**

* On GitHub, create a new issue on your repo. The issue should be about adding how many years you have been at UofT.
* On the main branch, modify the helloworld.py program by adding your name to the print statement. Commit and push the changes.
* On the develop branch (do not pull or merge any changes), modify the helloworld.py program by adding how many years you have been at UofT to the print statement. Commit and push the changes.
* On GitHub, create a pull request to merge the changes from the develop to the main branch. You should get an error saying "Can’t automatically merge". If you do not get the error, let the lab TA know (or post it on Piazza if you are not in a lab session).
* Despite the error, you should have been able to create the pull request. Assign the issue you created to the pull request by commenting on the pull request the number of the issue (i.e., #n, where n is the number of the issue you created).
* Follow the instructions on GitHub to fix the conflict.
* Commit and push the new change of resolving the conflicts.
* Add a **screenshots** of the successful merge into your README file on the main branch. Commit and push the change.
* On GitHub, close the pull request.

**Activity 4 (1 point): Adding collaborators and pulling changes**

* Add one of your teammates [or find a partner in the lab] as a collaborator on your repo.
* Clone your teammate’s repo.
* On your teammate’s repo, make a change to the helloworld.py program. Commit and push the changes.
* On your own repo, pull the changes and check the changes that your teammate made.
* Add the **link** to your teammates’ repo and **screenshots** for the commits in the last 2 steps into your README file on the main branch. Commit and push the changes.

**Activity 5 (1 point): Practice git rebase.**

* Switch to the “develop” branch.
* Create a new branch named “rebase”, randomly modify(/add/delete) some files and add 2 new commits (e.g., c1, c2).
* Switch back to the “develop” branch. Add 2 new commits to the ‘develop’ branch (e.g., c3, c4)
* Move c1, c2 after c4 using *git rebase*. If you are not sure how to accomplish, please check the documentations provided in **References** above. If you get an error for merge conflict in this step, follow the instructions to solve the error.
* Push all your changes including the “rebase” branch to GitHub. Add a few **screenshots** of the commands (along with their outputs) you used for rebase in the README file on the ‘main’ branch. Commit and push the changes.

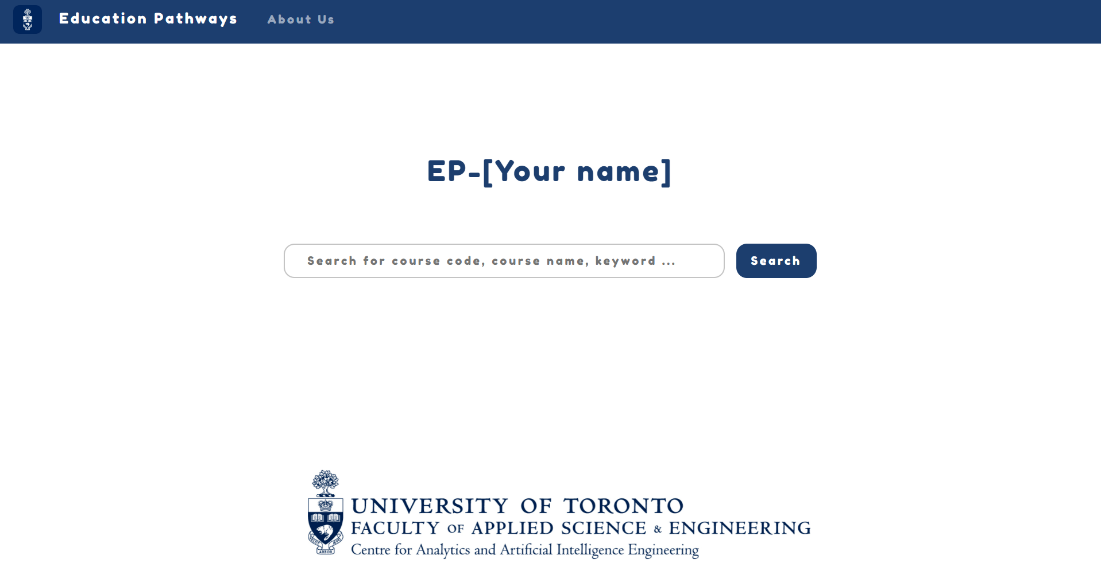
**Activity 6 (1 points): Set up the** [**Education Pathways**](https://github.com/ECE444-2022Fall/Assignment_1_starter_template) **Repo on GitHub**

* Create an empty public repository, name format: ECE444-F2022-EP (referred as **EP repo** later)
* Clone the [Education Pathways repo](https://github.com/ECE444-2022Fall/Assignment_1_starter_template) to your local machine. (If you haven’t done this in Assignment 1 )
* Modify the readme file by adding your name and mention to the top of it: this repo is a clone of <https://github.com/ECE444-2022Fall/Assignment_1_starter_template>. (Because your repo is public on GitHub, and mentioning the source of the code is a good practice of reusing open source code). Commit your changes.
* Push the EP repo on your local machine to the EP repo in your own GitHub account. (Here you may have to make some changes to your remote branch to push the repo to your own GitHub repo.)
* Add the **link** to your EP repo in the README file of your Lab1 repo (ECE444-F2022-Lab1) created in Activity 1.

Please try to spend some time understanding the Education Pathways code base before completing the following 2 activities.

**Activity 7 (1 points): Modify the title of the webpage**

* Create a new branch called “develop” in the repo and switch to it.
* Create another branch called “feature-change-title” from “develop” and switch to it.
* [In development mode](https://github.com/ECE444-2022Fall/Assignment_1_starter_template#2-to-run-the-app-in-development), Modify the code base to make the central title “Education Pathways” changed into “EP-[Your name]”.
* Rebase the change into the develop branch. Add the **screenshots** of your rebase command and the updated home page with the new title in the README file of your Lab1 repo on the main branch.



**Activity 8 (2 Point) Add a new tab to the navigation bar**

* Create a feature branch named “feature-add-a-tab” from the develop branch
* Try to add a new tab in the navigation bar called “ECE444-Lab1”
* Create a new page for the tab and add your name to the new page. (See the screenshot below.) Commit the change.



* Rebase the change into the develop branch. Add the **screenshots** of your rebase command and the new page for the new tab in the README file of your Lab1 repo on the main branch.

**Activity 9 (1pt)**: Commit and push the final version of your README file of your Lab1 repo with all the screenshots and links as required above.