Lab: (Command Line) Pull Requests II

Estimated time: 25 minutes

> *Note: This lab assumes that you are using a command line. If you would prefer to use Sourcetree, there are separate instructions.*

In this lab, you will:

1. Fork a remote repository.
2. Synchronize a forked repository using Bitbucket.
3. Create a multi-repository pull request.
4. Merge a multi-repository pull request.

1: Fork a remote repository.

1. Create a remote repository that we will consider to be the "upstream" repository. Do this by logging into Bitbucket and creating a repository named `projectj`. You can have Bitbucket create the README commit if you would like.

2. If you didn't create commit above, create and commit a `README.md` file containing the text `# PROJECTJ README #` with a commit message of "Initial commit".

3. **Fork** your `projectj` repository. Do this by clicking the + and selecting `Fork this repository`. Name the fork `projectjfork`. After creating the fork, you should see `projectjfork` in your list of Bitbucket repositories.

   > *Congratulations, you have forked a remote repository.*

2: Synchronize a forked repository using Bitbucket.

1. In the upstream repository (`projectj`), update `README.md` and create a commit. You can do this directly in Bitbucket or from your local client, pushing the changes to `projectj`.

2. In Bitbucket, navigate to `projectjfork`.

3. Select the **Source** tab. View the Repository details in the upper right. You may need to click on the icon in the upper right (or press]) to expand the repository details, then click > to show the dropdown.

4. Click the **Sync (1 commit behind)** link. Accept the default merge message and click `Sync`.

5. Click the `Commits` link. Notice that a merge commit was created in your fork.

   > *Congratulations, you have synchronized a repository using Bitbucket.*

3: Create a multi-repository pull request.

1. Using the command line, clone `projectjfork` to create a local repository for the forked repository.

2. Create a branch named `feature1` off of the `master` branch. This will be the branch that is part of the pull request.

3. Create a commit on the `feature1` branch containing a file named `fileA.txt` with the line "feature 1" as the content of the file.

4. Push the `feature1` branch to the remote `projectjfork` repository.

5. In Bitbucket, navigate to the `projectjfork` repository. Click on `Commits` and `Branches` to verify that your `feature1` branch and commit are on the remote repository.

6. Click on `Pull requests`. Click `Create pull request`.

7. On the `Create a pull request` page, select `feature1` on the left and `master` on the right. Modify the information if you would like and select `Create pull request`. You should see that your pull request was created.

   > *Congratulations, you have created a multi-repository pull request.*

4: Merge a multi-repository pull request.

1. In Bitbucket, navigate to `projectj`. This is the upstream repository. Click on `Pull requests`. You should see the pull request from your fork.

2. Click on the link to view the pull request.

3. Click the `Merge` button. Accept the default commit message and merge strategy (merge commit). Click `Merge`. You should see that the `feature1` branch is now merged. Click on `Commits` and verify that the work of feature 1 is now in your commit graph.

4. In Bitbucket, navigate to the `projectjfork` repository. Select the **Source** tab. Because of the merge commit upstream, this repository is 2 commits behind. Click the `Sync` button on the right (in `Repository details`). Accept the default commit message and click `Sync`.

5. Because the work of the `feature1` branch is merged, you can delete the `feature1` branch label in the forked remote repository. You should find the `feature1` branch under `Branches > Merged branches`.

6. Using the command line, delete the local `projectjfork` repository/directory and create a new clone of the same name (you may need to delete the existing local folder first). View the commit graph and verify that the merge commit from upstream is present. Now all of your repositories should be synchronized.

   > *Note: Deleting the local repository and recreating it using `clone` is a choice. Because we knew that all of our local work has been pushed to the remote repository, we can delete the local repository and start over using a clone. We are then assured that the remote and local repositories are synchronized.*

7. You will not use the `projectj` or `projectjfork` repositories in future labs. You can delete them.

> *Congratulations, you have merged a multi-repository pull request and completed this lab.*