Git for Team. A scenario.

Writted by Hong Le Nguyen, last update 07.2025
Code base on Git: https://github.com/hong1234/gitForTeam1024.git

Admin

#task_1: create new branch "feature_dao" on server by admin

```
// git branch
git checkout staging
git checkout -b feature_dao

git add .
git commit -m "new feature branch feature_dao  by admin"

git push -u origin feature_dao
```

#task_2: create new branch "feature_service" on server by admin

```
git checkout staging
git checkout -b feature_service

git add .
git commit -m "new feature branch feature_service by admin"

git push -u origin feature_service
```

#task_4: merge feature branch "feature_dao" into staging branch

```
git fetch origin

git checkout feature_dao
git pull --rebase

git checkout staging
git merge --no-ff feature_dao
```

#task_4b: merge feature branch "feature_service" into staging branch

```
git checkout feature_service
git pull --rebase

git checkout staging
git merge --no-ff feature_service
```

#task_5: push staging to server

```
git checkout staging
git push origin staging
```

#task_6: delete branches

```
git branch -d feature_dao
git push --delete origin feature_dao

git branch -d feature_service
git push --delete origin feature_service
```

Dev1

#task_3: Dev1 makes some code

```
// get the remote branch "feature_dao"
git fetch origin

git checkout feature_dao
// branch 'feature_dao' set up to track 'origin/feature_dao'
// Switched to a new branch 'feature_dao'

// add or change some code here …

// git add src/Dao/UserDao.php
// git commit -m "add Dao/UserDao.php by dev-1"

git add .
git commit -m "updated by dev-1"

// Synchronize with remote branch if need before push
// git pull --rebase

// git push origin feature_dao
git push
```

#task_7: staging branch update

```
git fetch origin

git checkout staging
git pull --rebase origin staging
```

#task_8: delete "feature_dao" branch

```
git branch -d feature_dao
```

Delete remote tracking branch in Git ---

```
git branch --delete --remotes origin/feature_dao
```

However*, if the branch has already been deleted from the GitHub server*, a simpler approach is to call the git fetch command with the prune option. The prune option removes any remote tracking branch in your local repository that points to a remote branch that has been deleted on the server.

```
git fetch origin –prune
```

Dev2a

#task_3b: Dev2a makes some code

```
// get the remote branch "feature_service"
git fetch origin

git checkout feature_service
// branch 'feature_service' set up to track 'origin/feature_service'.
// Switched to a new branch 'feature_service'

// add or change some code here …

// git add src/Service/UserService.php
// git commit -m "add Service/UserService.php by dev-2a"

git add .
git commit -m "updated by dev-2a"

// Synchronize with remote branch if need before push
// git pull --rebase

// git push origin feature_dao
git push
```

#task_7b: staging branch update

```
git fetch origin

git checkout staging
git pull --rebase origin staging
```

#task_8b: delete "feature_service" branch

```
git branch -d feature_service
```

Delete remote tracking branch in Git ---

```
git fetch origin –prune
```

Dev2b

#task_3b: Dev2b makes some code

```
// get the remote branch "feature_service"
git fetch origin

git checkout feature_service
// branch 'feature_service' set up to track 'origin/feature_service'
// switched to a new branch 'feature_service'

// add or change some code here …
```

```
        // git add src/Service/BookService.php
        // git commit -m "add Service/BookService.php by dev-2b"

        git add .
        git commit -m "updated by dev-2b"

        // Synchronize with remote branch if need before push
        // git pull --rebase

        // git push origin feature_dao
        git push

#task_7b: staging branch update

        git fetch origin
        git checkout staging
        git pull --rebase origin staging

#task_8b: delete "feature_service" branch
        git branch -d feature_service

Delete remote tracking branch in Git ---
        git fetch origin –prune
```