Lab1 - Git

Set global config and create local repository

1. Set your global name and email

```
brahim@Training:/tmp/Lab1$ git config --global user.name "Brahim HAMDI"
brahim@Training:/tmp/Lab1$ git config --global user.email "brahim.hamdi.consult@gmail.com"
brahim@Training:/tmp/Lab1$
brahim@Training:/tmp/Lab1$ git config --list
user.name=Brahim HAMDI
user.email=brahim.hamdi.consult@gmail.com
user.name=Brahim HAMDI
user.email=brahim.hamdi.consult@gmail.com
brahim@Training:/tmp/Lab1$
```

2. Initialize GIT repository in the folder.

```
brahim@Training:/tmp/Lab1$ git init project1
astuce: Utilisation de 'master' comme nom de la branche initiale. Le nom de la branche
astuce: par défaut peut changer. Pour configurer le nom de la branche initiale
astuce: pour tous les nouveaux dépôts, et supprimer cet avertissement, lancez :
astuce:
astuce:         git config --global init.defaultBranch <nom>
astuce:
astuce: Les noms les plus utilisés à la place de 'master' sont 'main', 'trunk' et
astuce: 'development'. La branche nouvellement créée peut être rénommée avec :
astuce:
astuce:         git branch -m <nom>
Dépôt Git vide initialisé dans /tmp/Lab1/project1/.git/
brahim@Training:/tmp/Lab1$ cd project1/
brahim@Training:/tmp/Lab1/project1$ ls -a
.  ..  .git
brahim@Training:/tmp/Lab1/project1$
```

What is in .git new create folder ?

```
brahim@Training:/tmp/Lab1/project1$ tree .git
.git
├── branches
├── config
├── description
├── HEAD
├── hooks
│   ├── applypatch-msg.sample
│   ├── commit-msg.sample
│   ├── fsmonitor-watchman.sample
│   ├── post-update.sample
│   ├── pre-applypatch.sample
│   ├── pre-commit.sample
│   ├── pre-merge-commit.sample
│   ├── prepare-commit-msg.sample
│   ├── pre-push.sample
│   ├── pre-rebase.sample
│   ├── pre-receive.sample
│   ├── push-to-checkout.sample
│   └── update.sample
├── info
│   └── exclude
├── objects
│   ├── info
│   └── pack
└── refs
    ├── heads
    └── tags

9 directories, 17 files
brahim@Training:/tmp/Lab1/project1$
```

First lab

Continuing on the same repository.

1. Create the following tree :

```
hasher
└── hasher.rb
README.md
rng
└── rng.py
webui
├── files
└── webui.js
worker
└── worker.py
```

```
brahim@Training:/tmp/Lab1/project1$ mkdir hasher rng worker webui
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ touch README.md hasher/hasher.rb rng/rng.py webui/webui.js worker/worker.py
```

2. Check the state of Git repository

```
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master

Aucun commit

Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        README.md
        hasher/
        rng/
        webui/
        worker/

aucune modification ajoutée à la validation mais des fichiers non suivis sont présents (utilisez "git add" pour les suivre)
brahim@Training:/tmp/Lab1/project1$
```

3. Add README.md to « staging area ». Then check the state of the local repository again.

```
brahim@Training:/tmp/Lab1/project1$ git add README.md
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master

Aucun commit

Modifications qui seront validées :
  (utilisez "git rm --cached <fichier>..." pour désindexer)
        nouveau fichier : README.md

Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        hasher/
        rng/
        webui/
        worker/

brahim@Training:/tmp/Lab1/project1$
```

4. Add README.md to the local repository as the first commit.

```
brahim@Training:/tmp/Lab1/project1$ git commit -m "1rst commit - add README.md"
[master (commit racine) 4864c73] 1rst commit - add README.md
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 README.md
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        hasher/
        rng/
        webui/
        worker/

aucune modification ajoutée à la validation mais des fichiers non suivis sont présents (utilisez "git add" pour les suivre)
brahim@Training:/tmp/Lab1/project1$
```

5. Add all files into GIT repository **as a single commit** (without git add).

```
brahim@Training:/tmp/Lab1/project1$ git commit -am "Commit without staging"
Sur la branche master
Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        hasher/
        rng/
        webui/
        worker/

aucune modification ajoutée à la validation mais des fichiers non suivis sont présents (utilisez "git add" pour les suivre)
brahim@Training:/tmp/Lab1/project1$
```

What's the problem ? How to solve it ?

```
brahim@Training:/tmp/Lab1/project1$ git add .
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Modifications qui seront validées :
  (utilisez "git restore --staged <fichier>..." pour désindexer)
        nouveau fichier : hasher/hasher.rb
        nouveau fichier : rng/rng.py
        nouveau fichier : webui/webui.js
        nouveau fichier : worker/worker.py

brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ git commit -m "2nd Commit - rest"
[master 20e1619] 2nd Commit - rest
 4 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 hasher/hasher.rb
 create mode 100644 rng/rng.py
 create mode 100644 webui/webui.js
 create mode 100644 worker/worker.py
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/project1$
```

6. Add these lines to worker/worker.py :

import logging
import os
from redis import Redis
import requests

import time

Then show the state.

```
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Modifications qui ne seront pas validées :
  (utilisez "git add <fichier>..." pour mettre à jour ce qui sera validé)
  (utilisez "git restore <fichier>..." pour annuler les modifications dans le répertoire de travail)
        modifié :         worker/worker.py

aucune modification n'a été ajoutée à la validation (utilisez "git add" ou "git commit -a")
brahim@Training:/tmp/Lab1/project1$
```

7. Commit all changes as a single commit.

```
brahim@Training:/tmp/Lab1/project1$ git commit -am "3rd commit - modif1 of worker/worker.py"
[master 893931e] 3rd commit - modif1 of worker/worker.py
 1 file changed, 5 insertions(+)
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/project1$
```

8. Show the commit history. What this command show ?

```
brahim@Training:/tmp/Lab1/project1$ git log
commit 893931e3721d6e0b2f706e60ae8c062baba6eeba (HEAD -> master)
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 10:49:05 2023 +0200

    3rd commit - modif1 of worker/worker.py

commit 20e161921544aeb1ae6894a5f5ca0e85a8c938e6
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 10:45:48 2023 +0200

    2nd Commit - rest

commit 4864c733d13e89f65cbbbd3cb349213e623e1f1c
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 10:42:42 2023 +0200

    1rst commit - add README.md
brahim@Training:/tmp/Lab1/project1$
```

9. Create a « .gitignore » and « architecture.pdf » files. Check the state of local repository.

```
brahim@Training:/tmp/Lab1/project1$ touch .gitignore architecture.pdf
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        .gitignore
        architecture.pdf

aucune modification ajoutée à la validation mais des fichiers non suivis sont présents (utilisez "git add" pour les suivre)
brahim@Training:/tmp/Lab1/project1$
```

10. Add « architecture.pdf » to « .gitignore » as first line, and then check the state of local repository again.

```
brahim@Training:/tmp/Lab1/project1$ vim .gitignore
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        .gitignore

aucune modification ajoutée à la validation mais des fichiers non suivis sont présents (utilisez "git add" pour les suivre)
brahim@Training:/tmp/Lab1/project1$
```

Commit changes.

```
brahim@Training:/tmp/Lab1/project1$ git add .gitignore
brahim@Training:/tmp/Lab1/project1$ git commit -m "Add .gitignore"
[master c474bb7] Add .gitignore
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/project1$
```

Removing a file from a repository

Continuing on the same repository:

1. Remove *worker/worker.py* with git rm command. show the state of local repository. Check how the staging area looks like.

```
brahim@Training:/tmp/Lab1/project1$ git rm worker/worker.py
rm 'worker/worker.py'
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Modifications qui seront validées :
  (utilisez "git restore --staged <fichier>..." pour désindexer)
        supprimé :        worker/worker.py

brahim@Training:/tmp/Lab1/project1$ tree
.
├── architecture.pdf
├── hasher
│   └── hasher.rb
├── README.md
├── rng
│   └── rng.py
└── webui
    └── webui.js

3 directories, 5 files
brahim@Training:/tmp/Lab1/project1$
```

Since the « worker » directory should be empty, it should disappear from disk, verify with system commad.

2. Undo the file deletion prior to commit.

```
brahim@Training:/tmp/Lab1/project1$ git restore --staged worker/worker.py
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Modifications qui ne seront pas validées :
  (utilisez "git add/rm <fichier>..." pour mettre à jour ce qui sera validé)
  (utilisez "git restore <fichier>..." pour annuler les modifications dans le répertoire de travail)
        supprimé :        worker/worker.py

aucune modification n'a été ajoutée à la validation (utilisez "git add" ou "git commit -a")
brahim@Training:/tmp/Lab1/project1$ tree
.
├── architecture.pdf
├── hasher
│   └── hasher.rb
├── README.md
├── rng
│   └── rng.py
└── webui
    └── webui.js

3 directories, 5 files
brahim@Training:/tmp/Lab1/project1$
```

Is the file in the workdir ? How to resove problem ?

```
brahim@Training:/tmp/Lab1/project1$ git checkout HEAD worker/worker.py
1 chemin mis à jour depuis cc9e871
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ tree
.
├── architecture.pdf
├── hasher
│   └── hasher.rb
├── README.md
├── rng
│   └── rng.py
├── webui
│   └── webui.js
└── worker
    └── worker.py

4 directories, 6 files
brahim@Training:/tmp/Lab1/project1$
```

3. Remake command 1, and then commit changes

```
brahim@Training:/tmp/Lab1/project1$ git rm worker/worker.py
rm 'worker/worker.py'
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
Modifications qui seront validées :
  (utilisez "git restore --staged <fichier>..." pour désindexer)
        supprimé :        worker/worker.py

brahim@Training:/tmp/Lab1/project1$ git commit -m "remove worker/worker.py"
[master d8d2c68] remove worker/worker.py
 1 file changed, 5 deletions(-)
 delete mode 100644 worker/worker.py
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/project1$
```

4. Revert to last commit (last version).

```
brahim@Training:/tmp/Lab1/project1$ tree
.
├── architecture.pdf
├── hasher
│   └── hasher.rb
├── README.md
├── rng
│   └── rng.py
└── webui
    └── webui.js

3 directories, 5 files
brahim@Training:/tmp/Lab1/project1$ git revert HEAD
[master 4911e47] Revert "remove worker/worker.py"
 1 file changed, 5 insertions(+)
 create mode 100644 worker/worker.py
brahim@Training:/tmp/Lab1/project1$ tree
.
├── architecture.pdf
├── hasher
│   └── hasher.rb
├── README.md
├── rng
│   └── rng.py
├── webui
│   └── webui.js
└── worker
    └── worker.py

4 directories, 6 files
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche master
rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/project1$
```

Working with branches

Continuing on the same repository:

1. List branches.

```
brahim@Training:/tmp/Lab1/project1$ git branch
* master
brahim@Training:/tmp/Lab1/project1$
```

   asterisk marking currently active branch.

2. Create a new branch *my_aple_app.*List branches.

```
brahim@Training:/tmp/Lab1/project1$ git branch my_aple_app
brahim@Training:/tmp/Lab1/project1$ git branch
* master
  my_aple_app
brahim@Training:/tmp/Lab1/project1$
```

3. Rename the new branche to *my_apple_app.*

```
brahim@Training:/tmp/Lab1/project1$ git branch  -m my_aple_app my_apple_app
brahim@Training:/tmp/Lab1/project1$ git branch
* master
  my_apple_app
brahim@Training:/tmp/Lab1/project1$
```

4. delete the branch.

```
brahim@Training:/tmp/Lab1/project1$ git branch -d my_apple_app
Branche my_apple_app supprimée (précédemment 4911e47).
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ git branch
* master
brahim@Training:/tmp/Lab1/project1$
```

5. Create and switch to new branch my_apple_app in one command.

```
brahim@Training:/tmp/Lab1/project1$ git checkout -b my_apple_app
Basculement sur la nouvelle branche 'my_apple_app'
brahim@Training:/tmp/Lab1/project1$ git branch
  master
* my_apple_app
brahim@Training:/tmp/Lab1/project1$
```

6. Implement OS X version and commit it

```
brahim@Training:/tmp/Lab1/project1$ mkdir osx
brahim@Training:/tmp/Lab1/project1$ touch osx/osx.py
brahim@Training:/tmp/Lab1/project1$ git branch
  master
* my_apple_app
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche my_apple_app
Fichiers non suivis:
  (utilisez "git add <fichier>..." pour inclure dans ce qui sera validé)
        osx/

aucune modification ajoutée à la validation mais des fichiers non suivis sont présents (utilisez "git add" pour les suivre)
brahim@Training:/tmp/Lab1/project1$ git add .
brahim@Training:/tmp/Lab1/project1$ git status
Sur la branche my_apple_app
Modifications qui seront validées :
  (utilisez "git restore --staged <fichier>..." pour désindexer)
        nouveau fichier : osx/osx.py

brahim@Training:/tmp/Lab1/project1$ git commit -m "add osx version"
[my_apple_app 2ba0a59] add osx version
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 osx/osx.py
brahim@Training:/tmp/Lab1/project1$ git log --oneline --all
2ba0a59 (HEAD -> my_apple_app) add osx version
4911e47 (master) Revert "remove worker/worker.py"
d8d2c68 remove worker/worker.py
c474bb7 Add .gitignore
893931e 3rd commit - modif1 of worker/worker.py
20e1619 2nd Commit - rest
4864c73 1rst commit - add README.md
brahim@Training:/tmp/Lab1/project1$
```

7. Switch back to master and verify if new changes in workdir.

```
brahim@Training:/tmp/Lab1/project1$ git checkout master
Basculement sur la branche 'master'
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ tree
.
├── architecture.pdf
├── hasher
│   └── hasher.rb
├── README.md
├── rng
│   └── rng.py
├── webui
│   └── webui.js
└── worker
    └── worker.py

4 directories, 6 files
brahim@Training:/tmp/Lab1/project1$ git log --oneline
4911e47 (HEAD -> master) Revert "remove worker/worker.py"
d8d2c68 remove worker/worker.py
c474bb7 Add .gitignore
893931e 3rd commit - modif1 of worker/worker.py
20e1619 2nd Commit - rest
4864c73 1rst commit - add README.md
brahim@Training:/tmp/Lab1/project1$
```

Reviewing the repository history

On the same local repository :

1. Show all commit history (one per line).

```
brahim@Training:/tmp/Lab1/project1$ git log --oneline --all
2ba0a59 (my_apple_app) add osx version
4911e47 (HEAD -> master) Revert "remove worker/worker.py"
d8d2c68 remove worker/worker.py
c474bb7 Add .gitignore
893931e 3rd commit - modif1 of worker/worker.py
20e1619 2nd Commit - rest
4864c73 1rst commit - add README.md
brahim@Training:/tmp/Lab1/project1$
```

2. Show commit history of *my_apple_app* branch

```
brahim@Training:/tmp/Lab1/project1$ git log --oneline my_apple_app
2ba0a59 (my_apple_app) add osx version
4911e47 (HEAD -> master) Revert "remove worker/worker.py"
d8d2c68 remove worker/worker.py
c474bb7 Add .gitignore
893931e 3rd commit - modif1 of worker/worker.py
20e1619 2nd Commit - rest
4864c73 1rst commit - add README.md
brahim@Training:/tmp/Lab1/project1$
```

3. Show commit history of *worker/worker.py* file.

```
brahim@Training:/tmp/Lab1/project1$ git log --oneline --all worker/worker.py
4911e47 (HEAD -> master) Revert "remove worker/worker.py"
d8d2c68 remove worker/worker.py
893931e 3rd commit - modif1 of worker/worker.py
20e1619 2nd Commit - rest
brahim@Training:/tmp/Lab1/project1$
```

Merging branches and conflict detecting (cloned repository)

1. Quit the Projet1 workdir, then clone this remote repository : *https://github.com/DevTrainings/test_merge_conflict.git*
   Change to *test_merge_conflict* directory. How many branches in this repository ?

```
brahim@Training:/tmp/Lab1$ git clone https://github.com/DevTrainings/test_merge_conflict.git
Clonage dans 'test_merge_conflict'...
remote: Enumerating objects: 23, done.
remote: Total 23 (delta 0), reused 0 (delta 0), pack-reused 23
Réception d'objets: 100% (23/23), 7.09 Kio | 7.09 Mio/s, fait.
Résolution des deltas: 100% (1/1), fait.
brahim@Training:/tmp/Lab1$ cd test_merge_conflict/
brahim@Training:/tmp/Lab1/test_merge_conflict$ ls -a
.  ..  file  .git  readme.md
brahim@Training:/tmp/Lab1/test_merge_conflict$ git branch
* master
brahim@Training:/tmp/Lab1/test_merge_conflict$
brahim@Training:/tmp/Lab1/test_merge_conflict$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/bar
  remotes/origin/cherry-pick
  remotes/origin/foo
  remotes/origin/master
brahim@Training:/tmp/Lab1/test_merge_conflict$
```

```
brahim@Training:/tmp/Lab1/test_merge_conflict$ git log  --all --oneline
c5d665f (HEAD -> master, origin/master, origin/HEAD) fixed readme a bit
4909911 (origin/cherry-pick) merge conflict in file
e7f0e30 added file for cherry_pick
286a273 added readme
c5c3da1 (origin/foo) file2 for demonstration of cherry-pick
bfb5647 (origin/bar) bar
c361f2e foo
25f7328 file
brahim@Training:/tmp/Lab1/test_merge_conflict$
brahim@Training:/tmp/Lab1/test_merge_conflict$
brahim@Training:/tmp/Lab1/test_merge_conflict$ git log  --all --oneline --graph
* c5d665f (HEAD -> master, origin/master, origin/HEAD) fixed readme a bit
| * 4909911 (origin/cherry-pick) merge conflict in file
| * e7f0e30 added file for cherry_pick
|/
* 286a273 added readme
| * c5c3da1 (origin/foo) file2 for demonstration of cherry-pick
| * c361f2e foo
|/
| * bfb5647 (origin/bar) bar
|/
* 25f7328 file
brahim@Training:/tmp/Lab1/test_merge_conflict$
```

2. The file `file` was changed in both branches `bar` and `foo` and we want to get those changes back into the master.
   What's the content of `file` in master, foo and bar branches ?

Branche	**master**	**bar**	**foo**
Contenu du ***file***	. . . . .	. **bar** . . bar	. **foo** . foo .

3. Merge *bar* to *master*. What the content of `file` in master branch now ?

```
brahim@Training:/tmp/Lab1/test_merge_conflict$ git checkout master
Basculement sur la branche 'master'
Votre branche est à jour avec 'origin/master'.
brahim@Training:/tmp/Lab1/test_merge_conflict$ git merge bar
Merge made by the 'ort' strategy.
 file | 4 ++--
 1 file changed, 2 insertions(+), 2 deletions(-)
brahim@Training:/tmp/Lab1/test_merge_conflict$ cat file
.
bar
.
.
bar
brahim@Training:/tmp/Lab1/test_merge_conflict$
```

4. Do the same with `foo`. What happened ? Is the merge completed ?

```
brahim@Training:/tmp/Lab1/test_merge_conflict$ git merge foo
Fusion automatique de file
CONFLIT (contenu) : Conflit de fusion dans file
La fusion automatique a échoué ; réglez les conflits et validez le résultat.
brahim@Training:/tmp/Lab1/test_merge_conflict$ cat file
.
<<<<<<< HEAD
bar
.
.
bar
=======
foo
.
foo
.
>>>>>>> foo
brahim@Training:/tmp/Lab1/test_merge_conflict$
```

5. Resolve manually the conflict then commit chages. Is merge foo ok ?

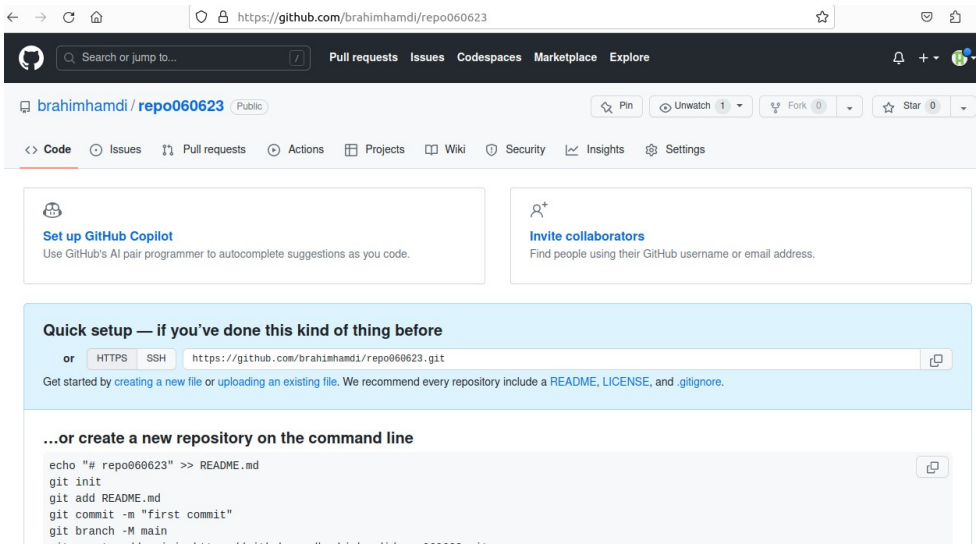
```
brahim@Training:/tmp/Lab1/test_merge_conflict$ vim file
brahim@Training:/tmp/Lab1/test_merge_conflict$ cat file
.
bar
.
foo
bar
brahim@Training:/tmp/Lab1/test_merge_conflict$ git commit -am "merge - resolv conflict !"
[master 33d43b1] merge - resolv conflict !
brahim@Training:/tmp/Lab1/test_merge_conflict$ git log  --all --oneline --graph
*   33d43b1 (HEAD -> master) merge - resolv conflict !
|\
| * c5c3da1 (origin/foo, foo) file2 for demonstration of cherry-pick
| * c361f2e foo
* |   5894235 Merge branch 'bar'
|\ \
| * | bfb5647 (origin/bar, bar) bar
| |/
* | c5d665f (origin/master, origin/HEAD) fixed readme a bit
| | * 4909911 (origin/cherry-pick, cherry-pick) merge conflict in file
| | * e7f0e30 added file for cherry_pick
| |/
|/|
* | 286a273 added readme
|/
* 25f7328 file
brahim@Training:/tmp/Lab1/test_merge_conflict$
```

Collaborative working on remote repository

On the Project1 repository :

1. Create an account on Github.

   Create a new empty repository on your remote GitHub account (example : remote_repo06062023).



2. Setup the remote repository for the local repository/

```
brahim@Training:/tmp/Lab1/test_merge_conflict$ cd ../project1/
brahim@Training:/tmp/Lab1/project1$ git remote -v
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ git remote add origin https://github.com/brahimhamdi/repo060623.git
brahim@Training:/tmp/Lab1/project1$ git remote -v
origin  https://github.com/brahimhamdi/repo060623.git (fetch)
origin  https://github.com/brahimhamdi/repo060623.git (push)
brahim@Training:/tmp/Lab1/project1$
```

3. Publish your entire repository to the server.

```
brahim@Training:/tmp/Lab1/project1$ git push origin master
Username for 'https://github.com': brahimhamdi
Password for 'https://brahimhamdi@github.com':
Énumération des objets: 19, fait.
Décompte des objets: 100% (19/19), fait.
Compression par delta en utilisant jusqu'à 8 fils d'exécution
Compression des objets: 100% (11/11), fait.
Écriture des objets: 100% (19/19), 1.63 Kio | 833.00 Kio/s, fait.
Total 19 (delta 4), réutilisés 0 (delta 0), réutilisés du pack 0
remote: Resolving deltas: 100% (4/4), done.
To https://github.com/brahimhamdi/repo060623.git
 * [new branch]      master -> master
brahim@Training:/tmp/Lab1/project1$ git branch -a
* master
  my_apple_app
  remotes/origin/master
brahim@Training:/tmp/Lab1/project1$
```

4. Quit Project1 repository, then clone remote repository in « collaborator » folder.

```
brahim@Training:/tmp/Lab1$ git clone https://github.com/brahimhamdi/repo060623 collaborator
Clonage dans 'collaborator'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 22 (delta 5), reused 22 (delta 5), pack-reused 0
Réception d'objets: 100% (22/22), fait.
Résolution des deltas: 100% (5/5), fait.
brahim@Training:/tmp/Lab1$ cd collaborator/
brahim@Training:/tmp/Lab1/collaborator$ ls -a
.  ..  .git  .gitignore  hasher  README.md  rng  webui  worker
brahim@Training:/tmp/Lab1/collaborator$
brahim@Training:/tmp/Lab1/collaborator$
```

5. Set your local name and email (différent from the global).

```
brahim@Training:/tmp/Lab1/collaborator$ git config --local user.name "collaborator"
brahim@Training:/tmp/Lab1/collaborator$ git config --local user.email "collaborator@gmail.com"
brahim@Training:/tmp/Lab1/collaborator$ git config --list
user.name=Brahim HAMDI
user.email=brahim.hamdi.consult@gmail.com
user.name=Brahim HAMDI
user.email=brahim.hamdi.consult@gmail.com
core.editor=vim
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
remote.origin.url=https://github.com/brahimhamdi/repo060623
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.master.remote=origin
branch.master.merge=refs/heads/master
user.name=collaborator
user.email=collaborator@gmail.com
brahim@Training:/tmp/Lab1/collaborator$ cat .git/config
[core]
        repositoryformatversion = 0
        filemode = true
        bare = false
        logallrefupdates = true
[remote "origin"]
        url = https://github.com/brahimhamdi/repo060623
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
        remote = origin
        merge = refs/heads/master
[user]
        name = collaborator
        email = collaborator@gmail.com
brahim@Training:/tmp/Lab1/collaborator$
```

6. Add these line to *hasher/hasher.rb* :

```
require 'digest'
require 'sinatra'
require 'socket'
```

```
brahim@Training:/tmp/Lab1/collaborator$ vim hasher/hasher.rb
brahim@Training:/tmp/Lab1/collaborator$ git status
Sur la branche master
Votre branche est à jour avec 'origin/master'.

Modifications qui ne seront pas validées :
  (utilisez "git add <fichier>..." pour mettre à jour ce qui sera validé)
  (utilisez "git restore <fichier>..." pour annuler les modifications dans le répertoire de travail)
        modifié :         hasher/hasher.rb

aucune modification n'a été ajoutée à la validation (utilisez "git add" ou "git commit -a")
brahim@Training:/tmp/Lab1/collaborator$
```

7. Commit changes, then push to remote repository.

```
brahim@Training:/tmp/Lab1/collaborator$ git commit -am "modif1 of hasher.rb by collaborator"
[master fef7a34] modif1 of hasher.rb by collaborator
 1 file changed, 3 insertions(+)
brahim@Training:/tmp/Lab1/collaborator$ git status
Sur la branche master
Votre branche est en avance sur 'origin/master' de 1 commit.
  (utilisez "git push" pour publier vos commits locaux)

rien à valider, la copie de travail est propre
brahim@Training:/tmp/Lab1/collaborator$
brahim@Training:/tmp/Lab1/collaborator$ git log -2
commit fef7a34b482cb532cafb1a9fad396c91aacefa6b (HEAD -> master)
Author: collaborator <collaborator@gmail.com>
Date:   Tue Jun 6 12:50:32 2023 +0200

    modif1 of hasher.rb by collaborator

commit 4911e47e303393fc0770eaa588bb2b5b1084f51b (origin/master, origin/HEAD)
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 11:19:23 2023 +0200

    Revert "remove worker/worker.py"

    This reverts commit d8d2c6843a4a7c46914dae4738dca7e99559df46.
brahim@Training:/tmp/Lab1/collaborator$
```

```
brahim@Training:/tmp/Lab1/collaborator$ git push origin master
Username for 'https://github.com': brahimhamdi
Password for 'https://brahimhamdi@github.com':
Énumération des objets: 7, fait.
Décompte des objets: 100% (7/7), fait.
Compression par delta en utilisant jusqu'à 8 fils d'exécution
Compression des objets: 100% (3/3), fait.
Écriture des objets: 100% (4/4), 349 octets | 349.00 Kio/s, fait.
Total 4 (delta 1), réutilisés 0 (delta 0), réutilisés du pack 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/brahimhamdi/repo060623
   4911e47..fef7a34  master -> master
brahim@Training:/tmp/Lab1/collaborator$
```

8. On the *Project1* repository :

pull the changes from remote repository.

```
commit 4911e47e303393fc0770eaa588bb2b5b1084f51b (HEAD -> master)
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 11:19:23 2023 +0200

    Revert "remove worker/worker.py"

    This reverts commit d8d2c6843a4a7c46914dae4738dca7e99559df46.

commit d8d2c6843a4a7c46914dae4738dca7e99559df46
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 11:17:06 2023 +0200

    remove worker/worker.py
brahim@Training:/tmp/Lab1/project1$
brahim@Training:/tmp/Lab1/project1$ git pull origin master
Depuis https://github.com/brahimhamdi/repo060623
 * branch            master     -> FETCH_HEAD
Mise à jour 4911e47..fef7a34
Fast-forward
 hasher/hasher.rb | 3 +++
 1 file changed, 3 insertions(+)
brahim@Training:/tmp/Lab1/project1$ git log -2
commit fef7a34b482cb532cafb1a9fad396c91aacefa6b (HEAD -> master, origin/master)
Author: collaborator <collaborator@gmail.com>
Date:   Tue Jun 6 12:50:32 2023 +0200

    modif1 of hasher.rb by collaborator

commit 4911e47e303393fc0770eaa588bb2b5b1084f51b
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jun 6 11:19:23 2023 +0200

    Revert "remove worker/worker.py"

    This reverts commit d8d2c6843a4a7c46914dae4738dca7e99559df46.
brahim@Training:/tmp/Lab1/project1$
```