



RAPPORT D'ACTIVITÉS DES SCÉNARIOS

Présenté en vue de compléter le
breif projet la prise en main du
Workflow GIT/GITHUB

Y O U C O D E

GIT / GITHUB WORFLOW

AMINE SAILA

Table des matières

Introduction.....	3
Scénario #1.....	4
#1 : First Step : Immersion.....	4
#2 : Second Step : La découverte.....	5
#3 : third step : Historique.....	7
#4 : Fourth Step : Excluding files	8
#5 : fifth Step : Branching and Merging.....	9
#6: sixth Step: Conflict Resolution.....	11
#7: Dans cette section nous avons amené à installer le logiciel 'p4merge'.....	13
#8: eighth Step: Challenge	14
Scénario #2.....	15
#1 : First Step : Tagging	15
#2 : Second Step : Stashing and Saving work in Progress	16
#3: third step: Voyage sur Github, Local Repo to github Repo	17
#4: Fourth Step: Mini challenge (optional)	18
#5: Fifth Step: Création d'une local copy :	19
#6: sixth step : Sending the website :.....	20
#7: seventh step : Fetch and pull :.....	24
Scénario #3.....	27
#1: First Step: Changes on Github.....	27
#2: Second Step: Branching and merging sur GITHUB.....	30
#3: Third Step: compare pull Requests.....	33
#4: fourth Step: merging en local.....	35
#5: fifth Step: The Cleaning up	38
#6: sixth step: Rebasing:	40
#7: seventh step GitHub Insights:.....	42
#8 : eighth step : Default branch and conflicts	42
#Quick Challenge: Try and Catch ()	45

Introduction

Dans le cadre de la validation des compétences de la période SAS , le brief projet est un moyen utile pour que nous validions les compétences dans leur niveau respectif N1, N2, N3 La gestion des workflows sous GIT & GITHUB sera la base de notre apprentissage en terme de la gestion de projet agile (Scrum) , en fonction d'une planification journalière des scénarios qui sont sous forme d'un Product BACKLOG

Scénario #1

#1 : First Step : Immersion

1. Définition de git

Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2.

2. Explication de fonctionnement du fichier caché (.git)

Le dossier .git contient toutes les informations nécessaires à votre projet dans le contrôle de version, ainsi que toutes les informations sur les validations, l'adresse du référentiel distant, etc. Toutes sont présentes dans ce dossier. Il contient également un journal qui stocke votre historique de validation afin que vous puissiez revenir à l'historique.

3. Sur votre bureau créez un répertoire nommée « /projects ».

```
#mkdir projects
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ mkdir projects

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ cd projects/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects
$ |
```

4. Créez un Repository Local sous le nom « /demo ».

```
#git init demo
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects
$ git init demo
Initialized empty Git repository in C:/Users/Administrateur/Downloads/CourseImplimentation/BreifProj/projects/demo/.git/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects
$ cd demo/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ |
```

5. Vérifiez le statut du répertoire « /demo ».

```
#git status
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

6. Expliquez le commentaire.

Tout simplement ce commentaire nous informe qu'il n'y a rien dans l'arbre de travail et qu'il n'y a rien à commettre

7. Créez le fichier README.md et ajoutez la ligne suivante : '#Demo project un simple fichier'.

```
#echo "#Demo project un simple fichier" >> readme.md
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ echo "#Demo project un simple fichier" >> readme.md

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ cat readme.md
"#Demo project un simple fichier"

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
```

8. Faites le staging et le committing avec un commentaire.

```
#git add readme.md

#git commit -m "Initial commit"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git add readme.md
warning: LF will be replaced by CRLF in readme.md.
The file will have its original line endings in your working directory

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git commit -m "initial commit"
[master (root-commit) 12f792c] initial commit
1 file changed, 1 insertion(+)
create mode 100644 readme.md
```

#2 : Second Step : La découverte

1. Déplacez-vous dans les fichiers de configuration « .git ».

```
#cd .git
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ cd .git
```

2. Tapez Ls -al

```
#ls -al
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo/.git (GIT_DIR!)
$ ls -al
total 13
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:09 ./
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:07 ../
-rw-r--r-- 1 Administrateur 197121 15 Nov 29 12:09 COMMIT_EDITMSG
-rw-r--r-- 1 Administrateur 197121 130 Nov 29 12:04 config
-rw-r--r-- 1 Administrateur 197121  73 Nov 29 12:04 description
-rw-r--r-- 1 Administrateur 197121  23 Nov 29 12:04 HEAD
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:04 hooks/
-rw-r--r-- 1 Administrateur 197121 137 Nov 29 12:09 index
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:04 info/
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:09 logs/
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:09 objects/
drwxr-xr-x 1 Administrateur 197121  0 Nov 29 12:04 refs/
```

3. Expliquez les clauses suivantes : HEAD, LOGS, BRANCHES

HEAD: Une tête est simplement une référence à un objet commit. Chaque tête a un nom (nom de branche ou nom de tag, etc.). Par défaut, il existe une tête dans chaque référentiel appelée maître. Un référentiel peut contenir un nombre quelconque de têtes. A tout moment, une tête est sélectionnée comme «tête actuelle». Cette tête est aliasée avec HEAD, toujours en majuscule ". Notez cette différence: une "tête" (minuscule) fait référence à l'une des têtes nommées du référentiel; "HEAD" (majuscule) fait exclusivement référence à la tête actuellement active. Cette distinction est fréquemment utilisée dans la documentation Git.

LOGS: Le log de Git est un outil formidable pour analyser l'historique des commits et resituer un contexte. Il nous permet aussi bien de suivre un projet dans sa globalité que dans ses détails : fonctionnalités, correctifs, fichiers et répertoires, auteurs, dates...

BRANCHES: Branching means you diverge from the main line of development and continue to do work without messing with that main line.

4. Retournez vers votre Repo et créez le fichier « Licence.md ».

```
#cd ..
```

```
#touch Licence.md
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo/.git (GIT_DIR!)
$ cd ..
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ touch Licence.md
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ ls
Licence.md  readme.md
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ |
```

5. Faites-le Commit

```
#git add .  
  
#git commit -m "second commit"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git add .  
  
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git commit -am "second commit"  
[master b4711c3] second commit  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 Licence.md
```

6. Affichez les fichiers traqués.

```
#git status
```

#3 : third step : Historique

1. Affichez le dernier commit sur une ligne en ajoutant l'option d'affichage de la hiérarchie de la branche, avec les commit et leur branche aussi

```
#git log - --oneline - --graph - --decorate - --all
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git log --oneline --graph --decorate --all  
* b4711c3 (HEAD -> master) second commit  
* 12f792c initial commit
```

2. Créez un alias de la commande précédente le nom de l'alias est : historique

```
#git config - --global alias.hist "log - --oneline - --graph - --decorate - --all "
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git config --global alias.hist "log --oneline --graph --decorate --all"
```

3. Affichez la liste des alias

```
#git config - --get-regexp alias
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git config --get-regexp alias  
alias.hist log --oneline --graph --decorate --all
```

4. Affichez l'historique des commit du fichier README.md avec l'alias.

```
#git hist readme.md
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git hist readme.md  
* 12f792c initial commit
```

#4 : Fourth Step : Excluding files

1. Renommer le fichier Licence.md à Licence.txt

```
#git mv Licence.md Licence.txt
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git mv Licence.md Licence.txt

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ ls
Licence.txt  readme.md
```

2. Faites le tagging avec mise à jour (ne pas faire « git add . »)

```
#git add -u
```

3. Faites-le commit.

```
#git commit -m "renamed Licence"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git add -u

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git commit -m "readme Licence"
[master de96bdc] readme Licence
1 file changed, 0 insertions(+), 0 deletions(-)
rename Licence.md => Licence.txt (100%)
```

4. Supposons que nous développons sur notre plateforme, sûrement on a des fichiers qu'on veut exclure de notre arborescence du repository Local
 - Créez un fichier nommé application.log

```
#touch application.log
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ touch application.log

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ ls
application.log  Licence.txt  readme.md
```

- Ne faites pas le tagging mais créez un fichier nommé « .gitignore »

```
#touch .gitignore
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ touch .gitignore
```

- Sur le fichier .gitignore Ajoutez la ligne suivante « *.log »

```
#echo "*.log" >> .gitignore
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ echo "*.log" >> .gitignore
```


- Faites le staging et le commit

```
#git add .  
#git commit -m "second commit"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git add .  
warning: LF will be replaced by CRLF in .gitignore.  
The file will have its original line endings in your working directory  
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git commit -m "second commit"  
[master 8d4ecc0] second commit  
1 file changed, 1 insertion(+)  
create mode 100644 .gitignore
```

- Qu'est-ce que vous constatez ?

ce que nous pouvons comprendre de cette étape est que quel que soit le fichier ou l'extension de fichier que nous avons mis dans .gitignore ne sera ni (staged) ni commitez

#5 : fifth Step : Branching and Merging

Introduction : un petit rappel, on a trois types pour faire le merge :

- The fast forward merging
- The manual
- The automatic

1. Modifiez le fichier README.md

```
#vim README.md
```

2. Ne faites pas le commit
3. Créez une branche pour la modification du fichier README.md du nom 'updates'

```
#git checkout -b updates
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)  
$ git checkout -b updates  
Switched to a new branch 'updates'
```

4. Faites le tagging et le commit en une seule ligne.

```
#git commit -am "changement sur le fichier readme.md"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (updates)
$ git commit -am "changement sur le fichier readme"
warning: LF will be replaced by CRLF in readme.md.
The file will have its original line endings in your working directory
[updates 869fb7d] changement sur le fichier readme
1 file changed, 1 insertion(+)
```

5. Affichez l'historique avec l'alias

```
#git hist
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (updates)
$ git hist
* 869fb7d (HEAD -> updates) changement sur le fichier readme
* 8d4ecc0 (master) second commit
* de96bdc readme Licence
* b4711c3 second commit
* 12f792c initial commit
```

6. Qu'est-ce que vous constatez ?

La branche actuelle est Updates et les modifications que nous avons apportées ont été validées dans cette branche

7. Ce que nous pouvons comprendre de cette étape est que lorsque nous créons une branche, vous pouvez ajouter une modification distincte de la branche principale.

8. Retournez vers la branche Master.

```
#git checkout master
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (updates)
$ git checkout master
Switched to branch 'master'
```

9. Affichez l'historique avec l'alias

```
#git hist
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git hist
* 869fb7d (updates) changement sur le fichier readme
* 8d4ecc0 (HEAD -> master) second commit
* de96bdc readme Licence
* b4711c3 second commit
* 12f792c initial commit
```

10. Qu'est-ce que vous constatez

Maintenant la branche actuelle est master

11. Nous voyons le dernier commit que nous avons fait dans la branche "updates" et on lui donne le nom de la branche "Updates"
12. Maintenant faites le merge.

```
#git merge updates
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git merge updates
Updating 8d4ecc0..869fb7d
Fast-forward
 README.md | 1 +
 1 file changed, 1 insertion(+)
```

#6: sixth Step: Conflict Resolution

1. Créez une branche avec le nom 'BAD'

```
#git checkout -b BAD
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git checkout -b BAD
Switched to a new branch 'BAD'
```

2. Modifiez le fichier README.md et ajoutez la ligne 'Trouble'

```
#vim README.md //ajouter line Trouble
```

3. Faites le tagging et le commit en une seule ligne

```
#git commit -am "Trouble"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (BAD)
$ git commit -am "Trouble"
[BAD 1ef98b0] Trouble
 1 file changed, 1 insertion(+)
```

4. Switchez faire la branche principale 'master'

```
#git switch master
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (BAD)
$ git switch master
Switched to branch 'master'
```

5. Maintenant modifiez le fichier README.md et ajoutez la ligne 'Troubleshooting'

```
#vim README.md //ajoutez Troubleshooting
```

6. Stagging/commiting avec commentaire 'branche bien faite'

```
#git commit -am "branche bien faite"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git commit -am "branche bien faite"
[master 862c1e1] branche bien faite
1 file changed, 1 insertion(+)
```

7. Oups!! corriger le commentaire du dernier commit

```
#git commit --amend
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git commit --amend
[master df7deea] branche est bien faite
Date: Fri Nov 29 20:29:28 2019 +0100
1 file changed, 1 insertion(+)
```

```
branche est| bien faite

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Fri Nov 29 20:29:28 2019 +0100
#
# On branch master
# Changes to be committed:
#   modified:   readme.md
#
```

8. Affichez l'historique

```
#git hist
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git hist
* df7deea (HEAD -> master) branche est bien faite
| * 1ef98b0 (BAD) Trouble
|/
* 869fb7d (updates) changement sur le fichier readme
* 8d4ecc0 second commit
* de96bdc readme Licence
* b4711c3 second commit
* 12f792c initial commit
```

9. Qu'est-ce que vous constatez ? expliquez l'objectif de ce petit scenario ?

le but de ces dernières tâches est de comprendre comment modifier le message de validation en cas d'erreur de frappe ou si vous souhaitez simplement modifier le message de validation.

10. Faites le merge de la branche 'BAD'

```
#git merge BAD
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git merge bad
Auto-merging readme.md
CONFLICT (content): Merge conflict in readme.md
Automatic merge failed; fix conflicts and then commit the result.
```

11. Expliquez le message ?

Tout Simplement nous avons un conflit de fusion qui signifie que git ne sait pas quel changement nous voulons garder. Nous devons donc résoudre ce conflit manuellement

12. Exécutez la commande suivante : cat README.md ? Expliquez ?

```
#cat README.md
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master|MERGING)
$ cat readme.md
"#Demo project un simple fichier"
this file is for information and other things
<<<<<< HEAD
Troubleshooting
=====
this the added line
>>>>>> bad
```

Nous voyons qu'il y a 2 changements et que nous devons choisir un seul changement et le valider, et ce faisant, nous avons résolu le conflit

13. Tapez la commande « git mergetool » ? qu'est-ce que vous constatez ?

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master|MERGING)
$ git mergetool

This message is displayed because 'merge.tool' is not configured.
See 'git mergetool --tool-help' or 'git help config' for more details.
'git mergetool' will now attempt to use one of the following tools:
opendiff kdiff3 tkdiff xxdiff meld tortoisemerge gvimdiff diffuse diffmerge ecmerge p4merge araxis bc codecompare smerge emerge vimdiff
Merging:
readme.md

Normal merge conflict for 'readme.md':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (vimdiff):
4 files to edit
```

Nous n'avons pas encore configuré P4merge en tant qu'outil de fusion

#7: Dans cette section nous avons amené à installer le logiciel 'p4merge'.

Lien d'installation

<https://www.perforce.com/products/helix-core-apps/merge-diff-tool-p4merge>

1. Installez p4merge sur votre PC
2. Sur git bash tapez la commande : git config --global --list

```
#git config --global --list
```

Cette commande nous montre une liste du contenu du fichier de configuration

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master|MERGING)
$ git config --global --list
core.editor=vim
user.name=Med Mouiguina
user.email=Med@gmail.com
credential.helper=cache
alias.hist=log --oneline --graph --decorate --all
```

3. Git config --global merge.tool p4merge
4. Git config --global mergetool.p4merge.path "lien d'installation" (.exe)
5. N'oubliez pas de bien configurer le prompt ? devinez comment ?

```
#git config --global difftool.prompt false
```

14. Tapez la commande git mergetool

```
#git mergetool
```

Analysez ce que vous voyez ? et expliquez la plateforme ouverte en temps réel ? Cet outil montre les deux différences qui ont été apportées au fichier et nous pouvons décider si nous souhaitons conserver les modifications ou les supprimer ou en créer de nouvelles.

#8: eighth Step: Challenge

Après avoir résolu les conflits des branches ; on a maintenant des fichiers indésirables à rejeter :

Sur le fichier .gitignore ; écrivez une clause pour rejeter les fichiers indésirables et redondants

```
#vim .gitignore
```

```
*.log
*.orig
~
~
```

On laisse que les fichiers : licence.txt Readme.md .gitignore (exemple *.log pour application.log)

On va ajouter les fichier d'extention .orig au fichier .gitignore et faire une commit

Scénario #2

#1 : First Step : Tagging

1. Déplacez-vous sur la branche Principale

```
#git switch master
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (updates)
$ git switch master
Switched to branch 'master'
```

2. Créez un TAG avec un nom V1.0 et un commentaire ' REALEASE 1.0 '

```
#git tag -a V1.0 -m "RELEASE 1.0"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git tag -a V1.0 -m "Release 1.0"
```

3. Affichez les informations sur le TAG

```
#git show --tags
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git show --tags
tag V1.0
Tagger: Med Mouiguina <Med@gmail.com>
Date:   Fri Nov 29 22:31:08 2019 +0100

Release 1.0

commit 183021121a36e8a933c58d822cb295bd11307941 (HEAD -> master, tag: V1.0)
Author: Med Mouiguina <Med@gmail.com>
Date:   Fri Nov 29 22:29:33 2019 +0100

    Updating gitignore file

diff --git a/.gitignore b/.gitignore
index 397b4a7..3598a4d 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,2 @@
 *.log
+*.orig
```

4. D'après vous un TAG il sert à faire quoi au juste ?

Les gens utilisent cette fonctionnalité pour marquer les points de publication (v1.0, v2.0, etc.). Nous avons utilisé ici les annotated tags. Ils sont stockés en tant qu'objets complets dans la base de données Git. Ils sont contrôlés; contient le nom, l'adresse e-mail et la date du tagueur; avoir un message de marquage

#2 : Second Step : Stashing and Saving work in Progress

5. Modifiez le fichier README.md , ajoutez une ligne

```
#echo "I added new line " >> README.md
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ echo "I added new line" >> readme.md
```

6. A vrai dire , l'équipe a décidé que cette tâche sera en standby et que les modifications progressent avec le timeline du projet
- Tapez la commande git Stash ?

```
#git stash
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git stash
warning: LF will be replaced by CRLF in readme.md.
The file will have its original line endings in your working directory
Saved working directory and index state WIP on master: 1830211 Updating gitignore file
```

- Expliquez le fonctionnement de la commande git Stash
Souvent, lorsque vous travaillez sur une partie de votre projet, la situation est dans un état désordonné et vous souhaitez changer de branche pour travailler sur autre chose. Le problème, c'est que vous ne voulez pas faire un commit à moitié fait pour pouvoir revenir à ce point plus tard. La réponse à ce problème est la commande git stash.
- Tapez la commande git stash list

```
#git stash list
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git stash list
stash@{0}: WIP on master: 1830211 Updating gitignore file
```

- Exécutez la commande git status ?

```
#git status
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git status
On branch master
nothing to commit, working tree clean
```

- Qu'est-ce que vous constatez ?
la cachette est créée et la modification que nous avons faite désespérée

7. Modifiez le fichier « Licence.txt », ajoutez la ligne « APACHE 2.0 »

```
#echo "APACHE 2.0" >> Licence.txt
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ echo "APACHE 2.0" >> Licence.txt

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ cat Licence.txt
APACHE 2.0
```

8. Faites staging et le commit en une seule ligne

```
#git commit -am "adding info to licence file"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git commit -am "adding info to licence file"
warning: LF will be replaced by CRLF in Licence.txt.
The file will have its original line endings in your working directory
[master d7f2157] adding info to licence file
1 file changed, 1 insertion(+)
```

9. Exécutez la commande « git stash pop » ?

```
#git stash pop
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git stash pop
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   readme.md

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (1310c64f802cac98c8535625adb4afa2c7601d2d)
```

10. Qu'est-ce que vous constatez ?

après avoir exécuté la commande git stash pop, nous avons appliqué la modification cachée que nous avons effectuée, puis le stash a été supprimée.

#3: third step: Voyage sur Github, Local Repo to github Repo

1. Pour cette phase vous devez avoir un compte GITHUB
2. Créez un repo github public sans ajouter le fichier README.md
3. Créez le remote en https

```
#git remote add origin https://github.com/Ned-med/brief_1.git
```

4. Examinez le remote

```
#git remote -v
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git remote add origin https://github.com/Ned-med/brief_1.git

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/projects/demo (master)
$ git remote -v
origin  https://github.com/Ned-med/brief_1.git (fetch)
origin  https://github.com/Ned-med/brief_1.git (push)
```

5. Pushez le tous à travers la commande : `git push -u origin master - --tags`

```
#git push -u origin master --tags
```

- Expliquez la commande ?
Le command et pour but de pushez notre repo à github avec les tags et aussi établir une "upstream" avec remote repo.
 - Expliquez les options `-u` et `--tags` ?
-u : cette option sert à configurer une relation de branche de suivi . Pour chaque branche mise à jour ou poussée avec succès, ajoutez une référence en amont (suivi).
- --tags : envoyer tous les tags que nous avons à github
6. Sur Github Vérifiez la liste des commits , les branches , les releases et les tags

les branches n'ont pas poussé pour faire ça on doit exécuter le command suivant `git push --all`

#4: Fourth Step: Mini challenge (optional)

En examinant les types d'authentification sur GITHUB, on tombe sur l'authentification HTTPS et SSH, certes HTTPS est beaucoup plus facile à manager tandis que le SSH est plus sécurisé tandis que fiable en ce qui concerne les transferts cryptés. Le but de ce challenge est de créer une authentification SSH entre votre repo local et le repo GITHUB.

- Créez un dossier sous git nommé **.SSH**

```
#mkdir .ssh
```

- Déplacez-vous dans le dossier **.SSH**

```
#cd .ssh
```

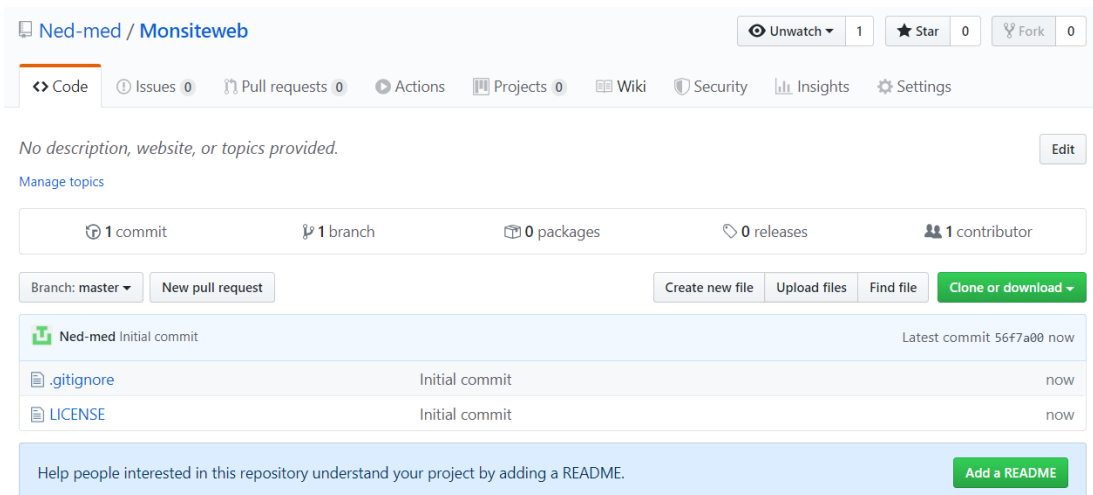
- Maintenant à vos mains : Créez une authentification sécurisé SSH entre votre repo local et votre repo distant (aide : `Ssh -keygen-t rsa -C 'email'`).

```
#ssh-keygen -t rsa -C "med@gmail.com"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/demo/.ssh (master)
$ ssh-keygen -t rsa -C "med@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Administrateur/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Administrateur/.ssh/id_rsa.
Your public key has been saved in /c/Users/Administrateur/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:nKkMUVbCoEE1agKhv8+QIF0W4s1ZxZBxkqp7zYzE9Rs med@gmail.com
The key's randomart image is:
+---[RSA 3072]-----+
|ooo=B+..          |
|o  *=+..          |
|o = .o            |
| = .o. . o        |
|o.=o.. S          |
|==*o oE.          |
|+B+= oo           |
|..o++ .           |
| . o              |
+----[SHA256]-----+
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/demo/.ssh (master)
```

#5: Fifth Step: Création d'une local copy :

1. Sur github créez un autre repo nommé (Monsiteweb)
2. Ajoutez à l'arboresence toujours sur github le fichier .gitignore et un fichier licence.txt 'APACHE 2.0 '



3. Déplacez-vous dans le répertoire projets sous GIT

```
Projets/demo# cd ..
```

4. Créez un clone github vers le local sous le nom (Monsiteweb-local)

```
#git clone https://github.com/Ned-med/Monsiteweb.git Monsiteweb-local
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ git clone https://github.com/Ned-med/Monsiteweb.git Monsiteweb-local
Cloning into 'Monsiteweb-local'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), done.
```

5. Vérifiez si le clone est créé.

```
#git clone https://github.com/Ned-med/Monsiteweb.git Monsiteweb-local
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ ls -al
total 8
drwxr-xr-x 1 Administrateur 197121 0 Nov 27 09:29 ./
drwxr-xr-x 1 Administrateur 197121 0 Nov 26 21:10 ../
drwxr-xr-x 1 Administrateur 197121 0 Nov 26 23:55 demo/
drwxr-xr-x 1 Administrateur 197121 0 Nov 27 09:29 Monsiteweb-local/
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
```

#6: sixth step : Sending the website :

1. Télécharger le site web depuis le lien suivant : <http://www.initializr.com/>
2. Sur le site telecharger un site bootstrap avec le fichier .htaccess et le fichier 404.html

2 - Fine tuning

HTML/CSS Template	HTML5 Polyfills	jQuery
<input type="radio"/> No template	<input checked="" type="radio"/> Modernizr	<input checked="" type="checkbox"/> Minified
<input type="radio"/> Mobile-first Responsive	<input type="radio"/> Just HTML5shiv	<input type="checkbox"/> Development
<input checked="" type="radio"/> Twitter Bootstrap	<input checked="" type="checkbox"/> Respond - <i>Alternatives</i>	

H5BP Optional		
<input type="checkbox"/> IE Classes	<input checked="" type="checkbox"/> Favicon	<input type="checkbox"/> Humans.txt
<input checked="" type="checkbox"/> Old browser warning	<input checked="" type="checkbox"/> Apple and Windows Icons	<input checked="" type="checkbox"/> 404 Page
<input checked="" type="checkbox"/> Google Analytics	<input type="checkbox"/> plugins.js	<input type="checkbox"/> Adobe Cross Domain
<input checked="" type="checkbox"/> .htaccess	<input type="checkbox"/> Robots.txt	

Download it!	What's inside?
---------------------	-----------------------

3. Analysez l'arborescence.

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ ls -al
total 12
drwxr-xr-x 1 Administrateur 197121 0 Nov 27 09:48 ./
drwxr-xr-x 1 Administrateur 197121 0 Nov 26 21:10 ../
drwxr-xr-x 1 Administrateur 197121 0 Nov 26 23:55 demo/
drwxr-xr-x 1 Administrateur 197121 0 Nov 27 09:48 initializr/
drwxr-xr-x 1 Administrateur 197121 0 Nov 27 09:29 Monsiteweb-local/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ cd initializr/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/initializr
$ ls
404.html          browserconfig.xml  favicon.ico  img/          js/          tile-wide.png
apple-touch-icon.png  css/              fonts/      index.html  tile.png
```

4. Expliquez ?

Nous avons téléchargé le site Web, nous avons extrait les fichiers et on a analysé l'arborescence.

5. Copiez le site télécharger dans votre repo local à travers une seule commande :

```
#cp -R ../initializr/* .
```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ cp -R ../initializr/* .
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ ls
404.html  apple-touch-icon.png  browserconfig.xml  css/  favicon.ico  fonts/  img/  index.html  js/  LICENSE  tile.png  tile-wide.png
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ |

```

6. Git status

```
#git status
```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    404.html
    apple-touch-icon.png
    browserconfig.xml
    css/
    favicon.ico
    fonts/
    index.html
    js/
    tile-wide.png
    tile.png

nothing added to commit but untracked files present (use "git add" to track)

```

7. Faites le tagging et le commit en une seule ligne

```
#git add -A && git commit -am "committing website"
```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git add -A && git commit -am "commiting website"
[master c60efa3] commiting website
22 files changed, 9594 insertions(+)
create mode 100644 404.html
create mode 100644 apple-touch-icon.png
create mode 100644 browserconfig.xml
create mode 100644 css/bootstrap-theme.css
create mode 100644 css/bootstrap-theme.min.css
create mode 100644 css/bootstrap.css
create mode 100644 css/bootstrap.min.css
create mode 100644 css/main.css
create mode 100644 favicon.ico
create mode 100644 fonts/glyphicons-halflings-regular.eot
create mode 100644 fonts/glyphicons-halflings-regular.svg
create mode 100644 fonts/glyphicons-halflings-regular.ttf
create mode 100644 fonts/glyphicons-halflings-regular.woff
create mode 100644 index.html
create mode 100644 js/main.js
create mode 100644 js/vendor/bootstrap.js
create mode 100644 js/vendor/bootstrap.min.js
create mode 100644 js/vendor/jquery-1.11.2.min.js
create mode 100644 js/vendor/modernizr-2.8.3-respons-1.4.2.min.js
create mode 100644 js/vendor/npm.js
create mode 100644 tile-wide.png
create mode 100644 tile.png

```

8. Faite le push à github

```
#git push --all
```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git push --all
git: 'credential-cache' is not a git command. See 'git --help'.
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 8 threads
Compressing objects: 100% (27/27), done.
Writing objects: 100% (28/28), 200.30 KiB | 1.15 MiB/s, done.
Total 28 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/Ned-med/Monsiteweb.git
 56f7a00..c60efa3  master -> master

```

9. Vérifiez l'existence du site local sur votre repo github

Ned-med / Monsiteweb

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. [Edit](#)

[Manage topics](#)

2 commits 1 branch 0 packages 0 releases 2 contributors Apache-2.0

Branch: master New pull request

Create new file Upload files Find file Clone or download

135579 committing website Latest commit c60efa3 27 minutes ago

css	committing website	27 minutes ago
fonts	committing website	27 minutes ago
js	committing website	27 minutes ago
.gitignore	Initial commit	2 hours ago
404.html	committing website	27 minutes ago
LICENSE	Initial commit	2 hours ago
apple-touch-icon.png	committing website	27 minutes ago
browserconfig.xml	committing website	27 minutes ago
favicon.ico	committing website	27 minutes ago
index.html	committing website	27 minutes ago
tile-wide.png	committing website	27 minutes ago
tile.png	committing website	27 minutes ago

#7: seventh step : Fetch and pull :

1. Sur github éditez le fichier Index.html, sur la balise <title> </title> ajoutez le titre , mon premier site web .

Ned-med / Monsiteweb

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

Monsiteweb / index.html [Cancel](#)

Edit file Preview changes

Spaces 4 No wrap

```

1 <!doctype html>
2 <html class="no-js" lang="">
3   <head>
4     <meta charset="utf-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6     <title>mon premier site web</title>
7     <meta name="description" content="">
8     <meta name="viewport" content="width=device-width, initial-scale=1">
9     <link rel="apple-touch-icon" href="apple-touch-icon.png">
10
11    <link rel="stylesheet" href="css/bootstrap.min.css">
12    <style>
13      body {
14        padding-top: 50px;
15        padding-bottom: 20px;
16      }
17    </style>
18    <link rel="stylesheet" href="css/bootstrap-theme.min.css">
19    <link rel="stylesheet" href="css/main.css">
20
21    <script src="js/vendor/modernizr-2.8.3-respond-1.4.2.min.js"></script>
22  </head>
23  <body>

```

2. Faites le commit sur github

Commit changes

changed the title

i've changed the title of the main page

☒ Commit directly to the `master` branch.
☐ Create a **new branch** for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

- Sur git et sur le repo local , Editez le fichier README.md

```
#echo "new line with more info" >> README.md
```

- Faites le tagging et le commit en une seule ligne

```
# git add -A && git commit -am "made changes to readme file"
```

- Qu'est-ce que vous constatez ?

nous avons apporté des modifications à la fois dans notre repo distant et local

Donc, fondamentalement, nous avons deux commits différents

Trouble shooting :

- Exécutez la commande « git fetch »

```
# git fetch
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/Ned-med/Monsiteweb
c60efa3..a7db2f8 master -> origin/master
```

- Git status

```
# git status
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git status
On branch master
Your branch and 'origin/master' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)

nothing to commit, working tree clean
```

- Expliquez ?

Nous avons récupéré nos commits de github et nous pouvons voir que nous avons deux changements différents dans chaque repo

4. Git pull

```
# git pull
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git pull
Merge made by the 'recursive' strategy.
 index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

5. Git push

```
# git push
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git push
git: 'credential-cache' is not a git command. See 'git --help'.
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 569 bytes | 569.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/Ned-med/Monsiteweb.git
 a7db2f8..6b9d899 master -> master
```

6. Verifiez les commit sur github

The screenshot shows the GitHub repository page for 'Ned-med / Monsiteweb'. The repository has 1 watch, 0 stars, and 0 forks. The main navigation bar includes links for Code, Issues (0), Pull requests (0), Actions, Projects (0), Wiki, Security, Insights, and Settings. The current branch is 'master'. The commit history for November 27, 2019, is displayed, showing five commits:

- Merge branch 'master' of https://github.com/Ned-med/Monsiteweb** (135579 committed 2 minutes ago) with commit hash 6b9d899.
- more changes to readme.md** (135579 committed 1 hour ago) with commit hash 71417c9.
- changed the title** (135579 committed 1 hour ago) with commit hash a7db2f8, marked as 'Verified'.
- committing website** (135579 committed 3 hours ago) with commit hash c60efa3.
- Initial commit** (135579 committed 4 hours ago) with commit hash 56f7a00, marked as 'Verified'.

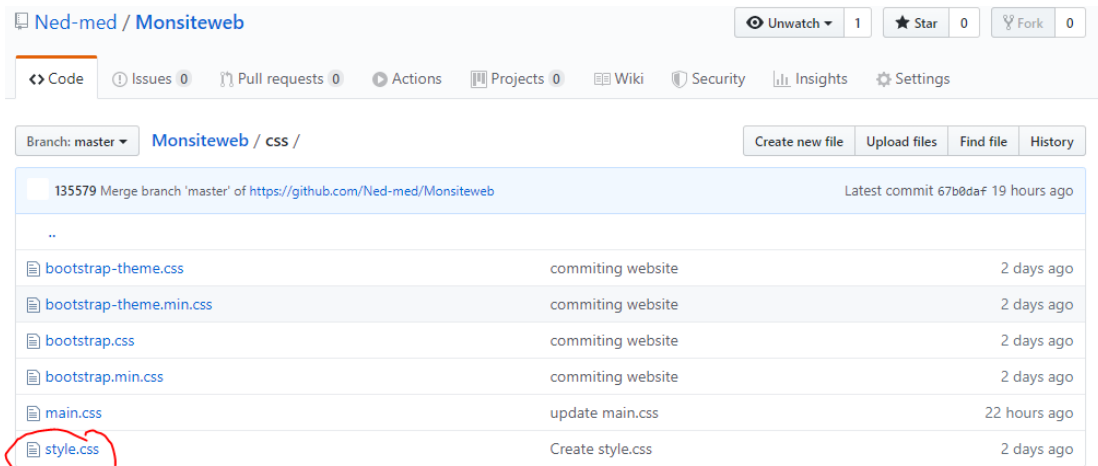
L'objectif de cette phase : c'est lorsqu'il y'a deux changements sur le repo local et sur le repo GitHub, donc les démarches décrites c'est pour la gestion de ce conflit

Scénario #3

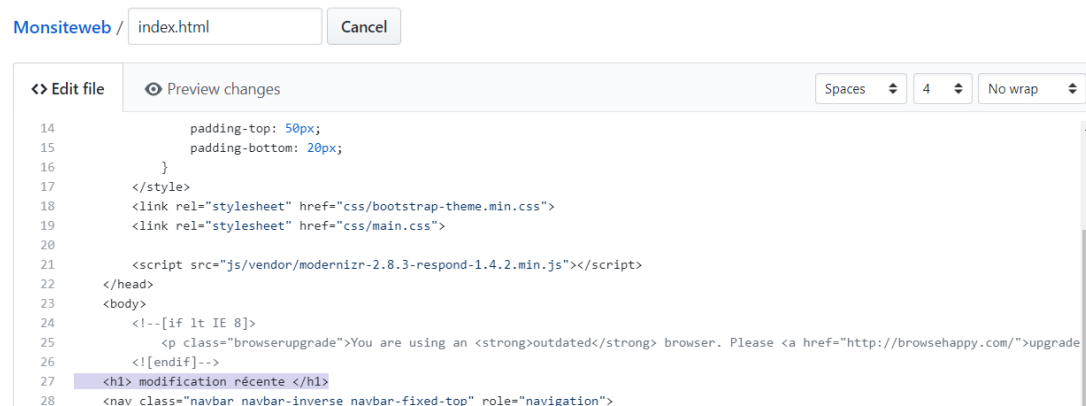
#1: First Step: Changes on Github

Sur votre compte Github : faites les modifications suivantes :

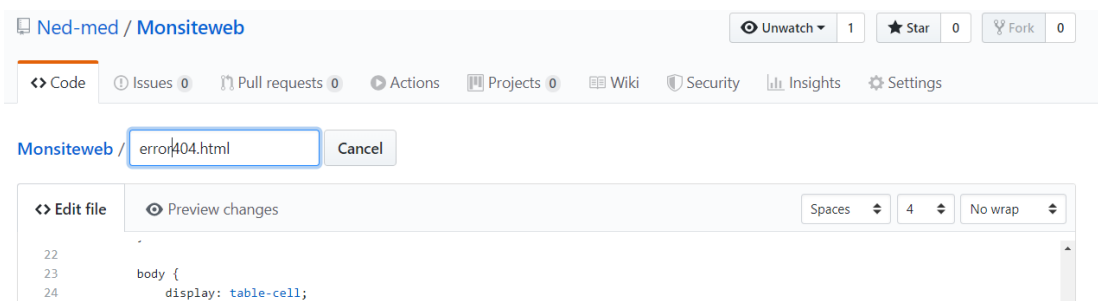
1. Sur le dossier CSS ajoutez un autre fichier nommé style.css



2. Sur index.html ajoutez juste après la balise <body> une balise <H1> : <h1> modification récente </h1>

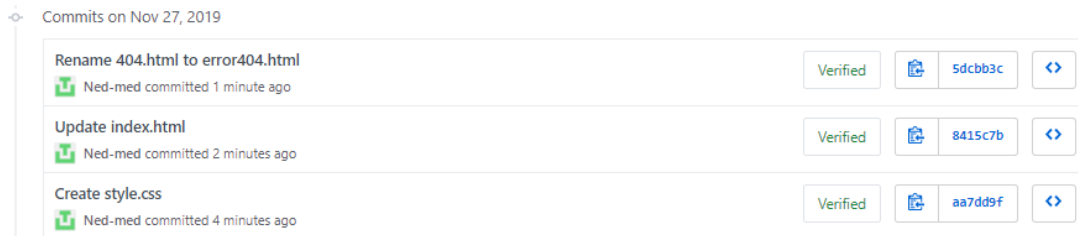


3. Renommer le fichier 404.html en error404.html.



Maintenant la synchronisation avec votre repository local, les changements effectués doivent impérativement apparaître sur votre repo LOCAL :

4. Sur gitHub vérifiez la liste des commits .



5. Sur GITBASH, vérifiez avec « git status »

```
# git status
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

6. Visualisez et télécharger les fichiers distants sur GITBASH

```
# git fetch
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git fetch
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 8 (delta 5), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (8/8), done.
From https://github.com/Ned-med/Monsiteweb
 6b9d899..5dcbb3c master -> origin/master
```

7. Maintenant faites le pull et fusionnez les changements distants avec le repo local

```
# git merge
```

8. Vérifiez votre repo Local

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git hist
* 5dcbb3c (origin/master, origin/HEAD) Rename 404.html to error404.html
* 8415c7b Update index.html
* aa7dd9f Create style.css
* 6b9d899 (HEAD -> master) Merge branch 'master' of https://github.com/Ned-med/Monsiteweb
|
| * a7db2f8 changed the title
| * | 71417c9 more changes to readme.md
|/
* c60efa3 committing website
* 56f7a00 Initial commit

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
```

9. Sur votre repo Distant GITHUB, laissez un commentaire sur les fichiers que vous avez modifiez

Rename 404.html to error404.html

master

Ned-med committed 13 minutes ago

Verified

1 parent 8415c7b commit 5dcbb3c5a1682d887605590f5c2a5a6bdf12ad9e

Showing 1 changed file with 0 additions and 0 deletions.

0 404.html → error404.html

File renamed without changes.

1 comment on commit 5dcbb3c

Lock conversation

Ned-med commented on 5dcbb3c now

Author Owner + ...

I modified the name of this file

Ned-med / Monsiteweb

Unwatch 1 Star 0 Fork 0

<> Code

Issues 0

Pull requests 0

Actions

Projects 0

Wiki

Security

Insights

Settings

Update index.html

master

Ned-med committed 16 minutes ago

Verified

1 parent aa7dd9f commit 8415c7bd747e8b2537f775ef88a9b28c906c92e0

Showing 1 changed file with 1 addition and 0 deletions.

1 index.html

@@ -24,6 +24,7 @@

24 24 <!--[if lt IE 8]>

25 25 <p class="browserupgrade">You are using an outdated browser. Please <a href="http://browsehappy.com/"

26 26 <![endif]>-->

27 + <h1> modification récente </h1>

27 28 <nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">

28 29 <div class="container">

29 30 <div class="navbar-header">

1 comment on commit 8415c7b

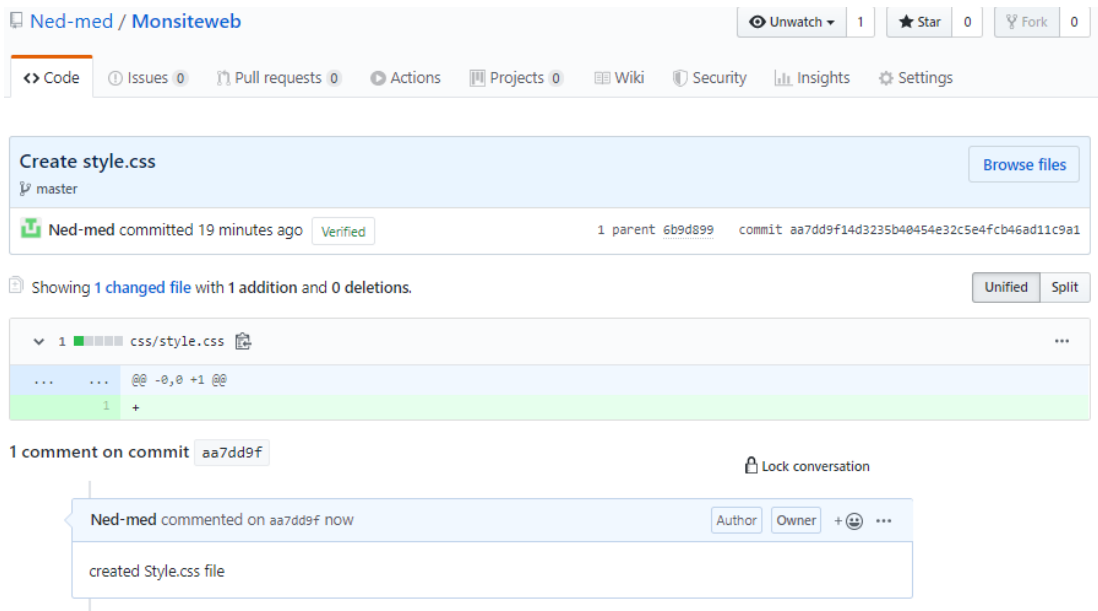
Lock conversation

Ned-med commented on 8415c7b now

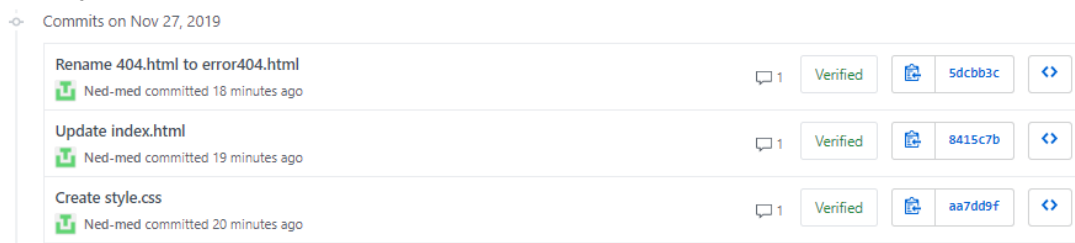
Author Owner + ...

updated index.html

29



10. En regardant la liste des commits.

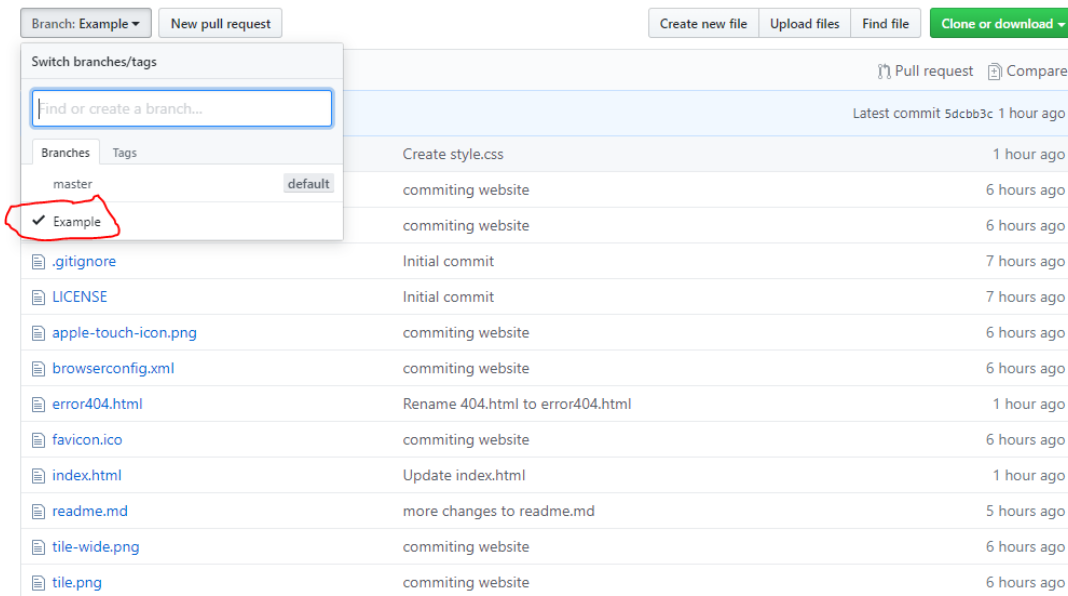


11. Sur GITBASH affichez les informations du commit de l'ajout de la balise <h1></h1>

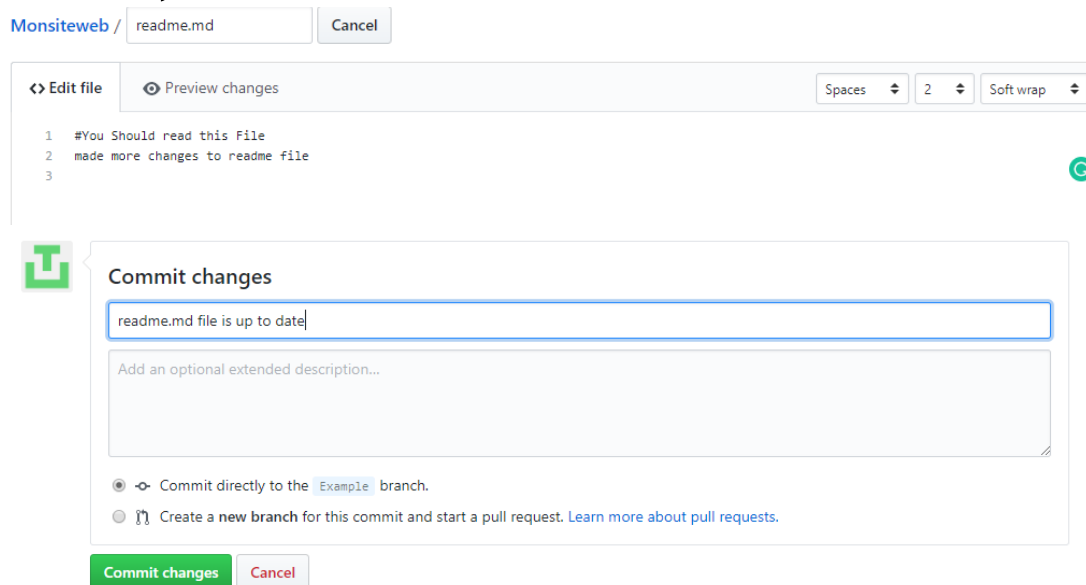


#2: Second Step: Branching and merging sur GITHUB

1. Repo distant :
 - Sur GitHub créez une branche du nom 'Example'



- Sur la branche 'Example', modifiez le fichier README.md et faites-le commit (toujours sur GITHUB)



- Sur GitHub et sur le menu 'Branch' ; Expliquez ce que vous voyez comme message ? Dans le menu des branches nous avons toutes les branches que nous avons et dans l'entrée, nous pouvons soit rechercher une branche, soit en créer une nouvelle
2. Repo local :
- Sur votre repo local, créez une branche nommé 'annulation'.

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git checkout -b annulation
Switched to a new branch 'annulation'

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (annulation)
```

- Sur votre repo local éditez le fichier « index.html » et supprimez la balise H1.

```
<!-- Main jumbotron for a primary marketing message or call to action -->
<div class="jumbotron">
  <div class="container">
    <h1>Hello, world!</h1>
    <p>This is a template for a simple marketing or informational website. It includes a large callout called a jumbotron and three supporting pieces of content. Use it as a starting point to create something more unique.</p>
    <p><a class="btn btn-primary btn-lg" href="#" role="button">Learn more &raquo;</a></p>
  </div>
</div>
```

index.html [dos] (13:10 27/11/2019) 55,9 49%

1 more line; before #1 14 seconds ago

- Faites-le commit et le tagging en une seule ligne.

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (annulation)
$ git commit -am "annulation de h1"
[annulation 73d3187] annulation de h1
1 file changed, 1 deletion(-)
```

- « Git show » pour vérifier vos changements.

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (annulation)
$ git show
commit 73d318779bd5307c768c38d53d963e24640adc75 (HEAD -> annulation, origin/annulation)
Author: Med Mouiguina <Med@gmail.com>
Date: Thu Nov 28 09:42:34 2019 +0100

    annulation de h1

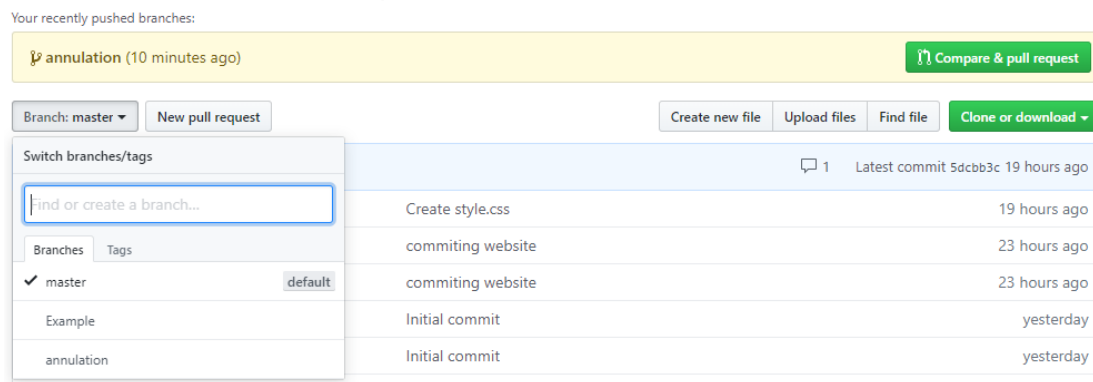
diff --git a/index.html b/index.html
index 1e75214..48af3c5 100644
--- a/index.html
+++ b/index.html
@@ -52,7 +52,6 @@
     <!-- Main jumbotron for a primary marketing message or call to action -->
     <div class="jumbotron">
       <div class="container">
-        <h1>Hello, world!</h1>
         <p>This is a template for a simple marketing or informational website. It includes a large callout called a jumbotron and three supporting pieces of content. Use it as a starting point to create something more unique.</p>
         <p><a class="btn btn-primary btn-lg" href="#" role="button">Learn more &raquo;</a></p>
       </div>
```

3. Vérification :

- Sur GITBASH Pushez les changements que vous avez sur la branche 'annulation' à votre repo distant

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (annulation)
$ git push origin annulation
git: 'credential-cache' is not a git command. See 'git --help'.
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 289 bytes | 72.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'annulation' on GitHub by visiting:
remote:   https://github.com/Ned-med/Monsiteweb/pull/new/annulation
remote:
To https://github.com/Ned-med/Monsiteweb.git
 * [new branch]      annulation -> annulation
```

- Maintenant vérifiez votre repo distant.



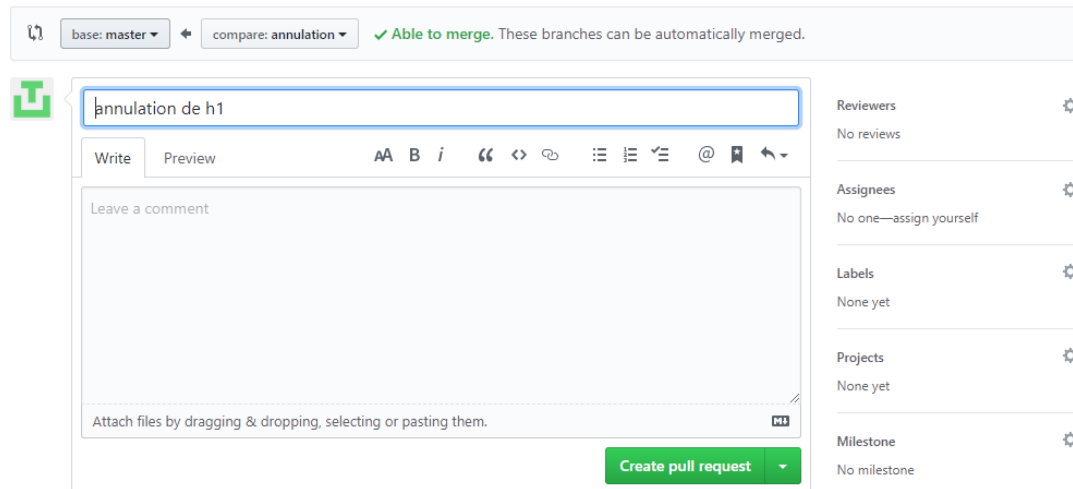
- Qu'est-ce que vous constatez.
Nous avons apporté des modifications à la branche "annulation" et nous avons transféré notre nouvelle branche vers le dépôt distant. il ne nous reste plus qu'à comparer dans GitHub et finaliser la demande de « pull »

#3: Third Step: compare pull Requests

1. Sur votre repo Distant, un message indiquant qu'une branche a été ajoutée avec un bouton « compare and pull request »
2. Cliquez sur le bouton ? analysez le rendu de la page ? qu'est-ce que vous constatez ?

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).



base: master compare: annulation ✓ Able to merge. These branches can be automatically merged.

annulation de h1

Write Preview

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Reviewers: No reviews

Assignees: No one—assign yourself

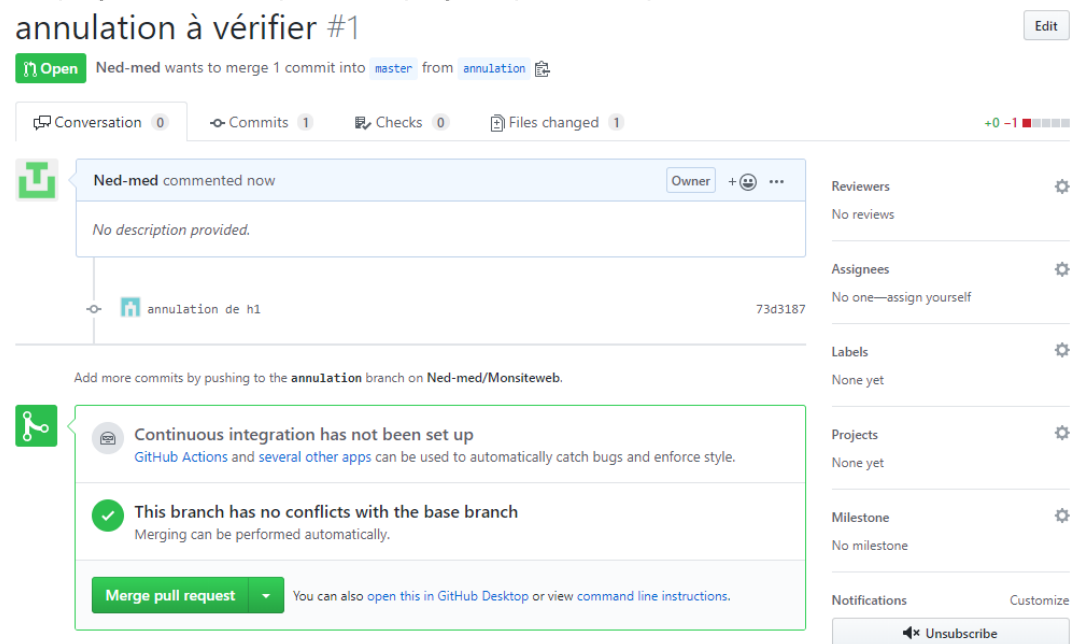
Labels: None yet

Projects: None yet

Milestone: No milestone

Nous avons la possibilité de créer une nouvelle demande de tirage en comparant les changements sur deux branches.

3. Créez un pull request avec le commentaire 'annulation à vérifier'
4. Une page s'ouvre, explorez la page ? qu'est-ce que vous constatez ?



annulation à vérifier #1

Open Ned-med wants to merge 1 commit into master from annulation

Conversation 0 Commits 1 Checks 0 Files changed 1 +0 -1

Ned-med commented now

No description provided.

annulation de h1 73d3187

Add more commits by pushing to the annulation branch on Ned-med/Monsiteweb.

Continuous integration has not been set up

GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request

You can also open this in GitHub Desktop or view command line instructions.

Reviewers: No reviews

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

Notifications: Customize

Unsubscribe

GitHub nous dit que cette branche n'a aucun conflit avec la branche de base La fusion peut être effectuée automatiquement

5. Sur l'onglet 'conversation' vous pouvez aussi laisser des commentaires à la personne qui demande le pull request ; laissez un commentaire ?
6. Maintenant si tout va bien ; aucun message d'erreur effectuez le merging ?
7. Supprimez la branche 'annulation' une fois tout est OK.

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git push origin :annulation
git: 'credential-cache' is not a git command. See 'git --help'.
To https://github.com/Ned-med/Monsiteweb.git
- [deleted]          annulation

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git branch -D annulation
Deleted branch annulation (was 73d3187).

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)

```

#4: fourth Step: merging en local

Pour finaliser l'objectif de cette étape, on va créer une branche nommé 'ajustement', ou on va modifier le fichier 'main.css' (css/main.css) en ajoutant la ligne suivante :

```

body {
  Width : 100%;
  Height : 100%
}

```

1. Faites maintenant le nécessaire pour 'pushez' les modifications sur GITHUB.

```

# git checkout -b ajustement
#vim css/main.css
#git commit -am "update main.css"
#git push ajustement

```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git checkout -b ajustement
Switched to a new branch 'ajustement'

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (ajustement)
$ vim css/main.css

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (ajustement)
$ git commit -am "update main.css"
[ajustement 7a4808d] update main.css
1 file changed, 5 insertions(+)

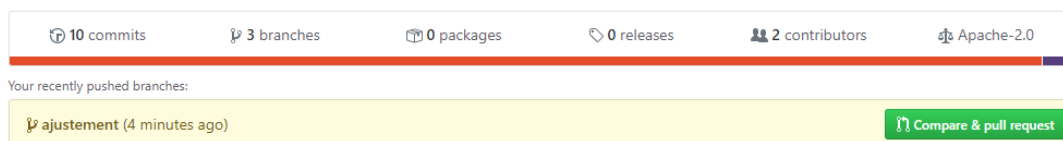
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git fetch

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git push origin ajustement
git: 'credential-cache' is not a git command. See 'git --help'.
Everything up-to-date

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)

```

2. Sur votre GITHUB, un pull request est demandé !



3. Maintenant, essayez de faire le merge en local ! @ vos mains !

```

# git merge ajustement

```

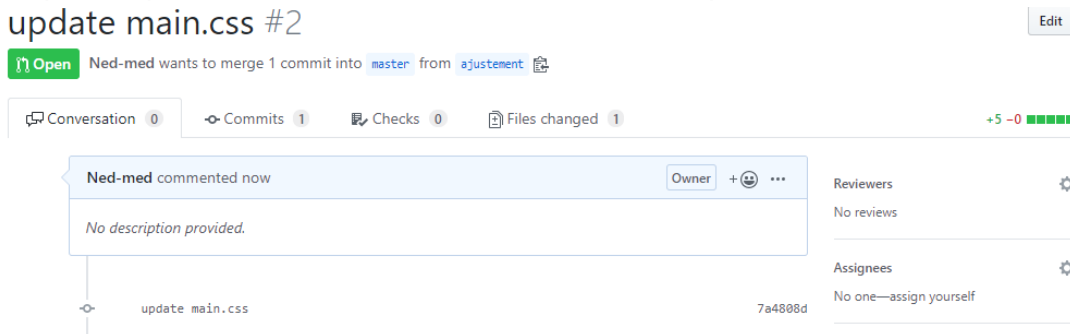
```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git merge ajustement
Updating 6b9d899..7a4808d
Fast-forward
 css/main.css | 5 +++++
 1 file changed, 5 insertions(+)
```

- Sur GITBASH, Vérifiez dans quelle branche vous êtes maintenant ! surement sur la branche 'ajustement'

```
# git branch
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (ajustement)
$ git branch
* ajustement
master
```

- Un pull Request est demandé sur GITHUB ! N'oubliez pas !
update main.css #2



- Sur GITBASH, revenez vers la branche master ! Expliquez le rendu de la commande ?

```
# git checkout master
```

Ces commandes nous permettent de basculer entre les branches et dans cette situation, la branche vers laquelle nous voulons basculer est la branche principale .

- Maintenant demandez le pull ?

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git pull
Merge made by the 'recursive' strategy.
 css/style.css | 1 +
 404.html => error404.html | 0
 index.html | 2 +-
 3 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 css/style.css
 rename 404.html => error404.html (100%)
```

- Une fois la commande du pull est exécutée, à votre avis quel type de merge on aura besoin pour faire un merge sans conflits
Fast-Forward Merge or automatique merge
- Faites le merge à travers GITBASH

```
# git merge
```

- Une fenêtre s'ouvre après l'exécution de la commande du merge ; ce qui signifie le message délivré ? à quoi sert cette fenêtre ?
La fenêtre qui s'est ouverte est simplement pour nous de commenter ou de décrire la fusion qui se produit

- Git status

```
# git status
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

- Si tout est OK ! Poussez les changements ?

```
# git push
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git push
git: 'credential-cache' is not a git command. See 'git --help'.
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 388 bytes | 194.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/Ned-med/Monsiteweb.git
  2545998..67b0daf  master -> master
```

- Si tout est OK ! sur GITHUB le pull sera vérifier ? explorez le pull pour visualiser les changements ?

4. Sur GITHUB supprimez les branches que vous avez créé.

Your branches		
ajustement	Deleted just now by Ned-med	Restore
Example	Deleted just now by Ned-med	Restore
Active branches		
ajustement	Deleted just now by Ned-med	Restore
Example	Deleted just now by Ned-med	Restore

5. Sur le local on doit apporter les changements qu'on a fait sur GITHUB :

- Tapez la commande git branch -a ? expliquez le message ?

```
# git branch -a
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git branch -a
ajustement
* master
remotes/origin/Example
remotes/origin/HEAD -> origin/master
remotes/origin/ajustement
remotes/origin/master
```

Répertorie les branches de suivi à distance et les branches locales. Combiner avec --list pour correspondre aux motifs optionnels

- Sur GITBASH supprimez les branches que vous avez créé.

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local/css (master)
$ git branch -d ajustement
Deleted branch ajustement (was 7a4808d).
```

- Maintenant ! Tapez la commande git branch -a ? qu'est-ce que vous constatez ?
Maintenant nous voyons que la branche "ajustement" a été supprimée
- Ce que nous devons faire maintenant, est de faire appellez la suppression faites sur GITHUB et l'appliquez sur le local ! Trouvez la commande et exécutez ?

```
# git push origin :<branch-name>
```

- Tapez la commande git branch -a ? qu'est-ce que vous constatez ?
Nous voyons que les branches ont été supprimées à la fois dans le dépôt distant et local

#5: fifth Step: The Cleaning up

Dans cette phase je vous laisse découvrir l'objectif ?

1. Sur GITHUB Créez une branche nommée 'updatelicense', éditez le fichier Licence et changer Apache 2.0 par Apache 3.0

Branch: updatelicense Monsiteweb / LICENSE Find file Copy path

	Permissions	Limitations	Conditions
<p>Ned-med/Monsiteweb is licensed under the Apache License 2.0</p> <p>A permissive license whose main conditions require preservation of copyright and license notices. Contributors provide an express grant of patent rights. Licensed works, modifications, and larger works may be distributed under different terms and without source code.</p> <p>This is not legal advice. Learn more about repository licenses.</p>	<ul style="list-style-type: none"> ✓ Commercial use ✓ Modification ✓ Distribution ✓ Patent use ✓ Private use 	<ul style="list-style-type: none"> ✗ Trademark use ✗ Liability ✗ Warranty 	<ul style="list-style-type: none"> ① License and copyright notice ① State changes

Ned-med Update From V2.0 to V3.0 de8b938 now

2 contributors

201 lines (169 sloc) | 11.1 KB Raw Blame History

```
1
2           Apache License
3           Apache Version 3.0, January 2004
4           http://www.apache.org/licenses/
```

2. Sur GITBASH faites un pull global : git pull --all ;

```
# git pull --all
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git pull --all
Fetching origin
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/Ned-med/Monsiteweb
* [new branch]      updatelicense -> origin/updatelicense
Already up to date.
```

3. Maintenant faites le merge de la branche 'updatelicense'.

```
# git merge origin/updatelicense
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git merge origin/updatelicense
Updating 299dcab..de8b938
Fast-forward
 LICENSE | 4 ++--
 1 file changed, 2 insertions(+), 2 deletions(-)
```

4. Exécutez Git push !

```
# git push
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git push
git: 'credential-cache' is not a git command. See 'git --help'.
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/Ned-med/Monsiteweb.git
 299dcab..de8b938  master -> master
```

5. Git branch -d 'updatelicense'

```
# git branch -d updatelicense
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git branch -d updatelicense
error: branch 'updatelicense' not found.
```

6. Exécutez la commande git branch -a ? qu'est-ce que vous constatez sur le remote et aussi sur GTHUB?

```
# git branch -a
```

la branche n'a pas été supprimée du dépôt distant, et elle n'existe pas dans le local

7. Maintenant sur GITBASH, trouvez la commande qui va nous permettre de supprimer la branche depuis le remote et ainsi de pousser les changements de la suppression simultanément ? Vérifiez vos Repo (local et distant)

```
#git push origin :updatelicense
```

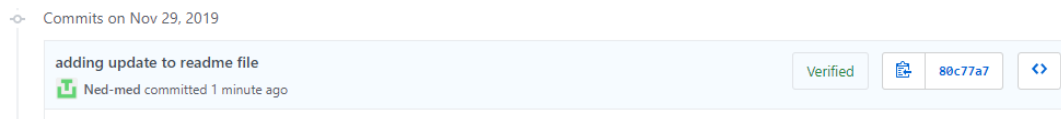
```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git branch -d updatelicense
error: branch 'updatelicense' not found.
```

8. Pourriez-vous expliquer l'objectif de cette phase ?

Le but de cette étape est de comprendre comment mieux manipuler les branches et surtout comment supprimer une branche distante du terminal Gitbash

#6: sixth step: Rebasing:

1. Faites une veille sur REBASE de 30 min intergroupe, visitez le lien suivant :
<https://www.grafikart.fr/tutoriels/amend-rebase-588>
2. Sur GITHUB, Editez le fichier README.md, ajoutez la ligne 'updates' ! n'oubliez pas le commit.



3. Sur GITBASH Editez le fichier index.html supprimez les lignes suivantes :

```
<!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8 lt-
ie7" lang=""> <![endif]-->
<!--[if IE 7]>         <html class="no-js lt-ie9 lt-
ie8" lang=""> <![endif]-->
<!--[if IE 8]>         <html class="no-js lt-ie9" lang=""> <![endif]-->
<!--[if gt IE 8]><!-->
```

4. Faites le tagging et le commit en une seule ligne

```
# git commit -am "updating index.html"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git commit -am "updating index.html"
[master 5649187] updating index.html
1 file changed, 4 deletions(-)
```

5. Maintenant apportez les changements faites sur le Repo distant : fetch

```
# git fetch
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/Ned-med/Monsiteweb
 de8b938..80c77a7  master    -> origin/master
```


6. Git status ? lisez les commentaires

```
# git status
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git status
On branch master
Your branch and 'origin/master' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)

nothing to commit, working tree clean
```

7. Maintenant si vous comprenez le commentaire, on a besoin de faire un rebase : git pull --rebase

```
# git pull --rebase
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git pull --rebase
First, rewinding head to replay your work on top of it...
Applying: updating index.html
```

8. Exécutez l'alias que vous avez créé dans le scénario #1 (historique) ? qu'est-ce que vous constatez sur l'arborescence des branches ?

```
# git hist
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb-local (master)
$ git hist
* e69c05c (HEAD -> master) updating index.html
* 80c77a7 (origin/master, origin/HEAD) adding update to readme file
* de8b938 Update From V2.0 to V3.0
* 299dcab update apache version to 3.0
* 67b0daf Merge branch 'master' of https://github.com/Ned-med/Monsiteweb
|
| * 2545998 Merge pull request #1 from Ned-med/annulation
|
| * 73d3187 annulation de h1
| * 7a4808d (origin/ajustement) update main.css
|
| * 75eab25 (origin/Example) readme.md file is up to date
|
| * 5dcbb3c Rename 404.html to error404.html
| * 8415c7b Update index.html
| * aa7dd9f Create style.css
|
| * 6b9d899 Merge branch 'master' of https://github.com/Ned-med/Monsiteweb
|
| * a7db2f8 changed the title
| * 71417c9 more changes to readme.md
|
| * c60efa3 committing website
| * 56f7a00 Initial commit
```

Après avoir exécuté la commande --rebase, nous voyons que le commit que nous avons fait au local vient après celui que nous avons fait dans github

9. Vérifiez que tous les changements que vous avez faits sont OK !

10. Expliquez l'utilité du REBASE ?

Rebase est un moyen d'intégrer les changements d'une branche à une autre. Rebase compresse tous les changements en un seul «patch». Ensuite, il intègre le patch sur la branche cible.

Contrairement à la fusion, le rebase aplatit l'historique, car il transfère le travail terminé d'une branche à une autre. Dans le processus, l'historique non désiré est éliminé.

Les "rebasements" indiquent comment les modifications doivent être transférées du sommet de la hiérarchie vers le bas, et les fusions indiquent comment elles remontent vers le haut.

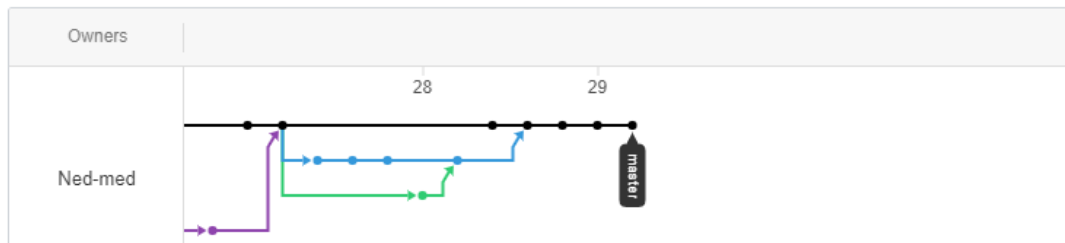
#7: seventh step GitHub Insights:

Sur Github vous avez un menu INSIGHTS ; Explorez votre activité à travers insights :

1. Analysez le Network ou le timeline de votre workflow ?

Network graph

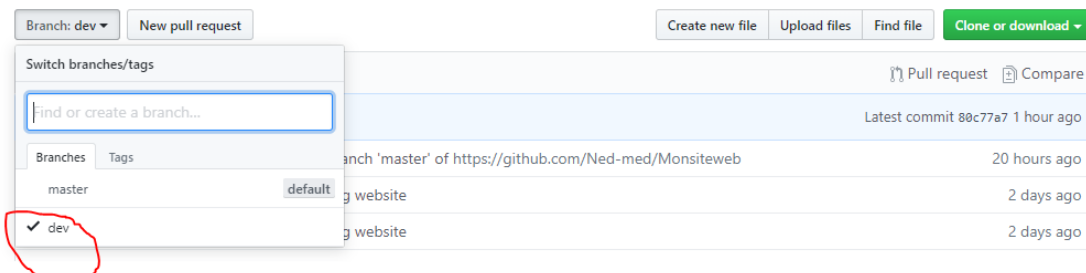
Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.



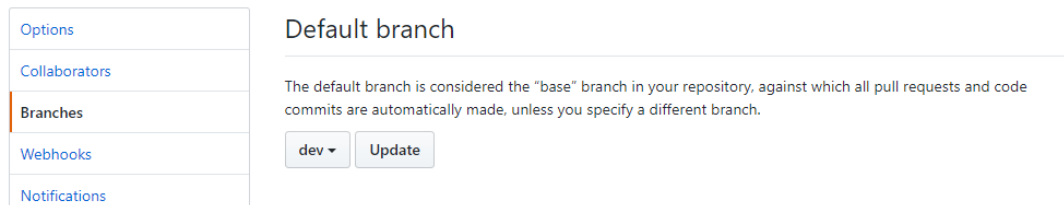
2. Discutez entre vous équipe et formalisez ce que vous voyez sur le Network en une production Timeline sur Tableau à marqueur ou trello : imaginez aussi que chaque membre du groupe à contribuer dans une tâche du Timeline Network (Like a Sprint planning) .

#8 : eight step : Default branch and conflicts

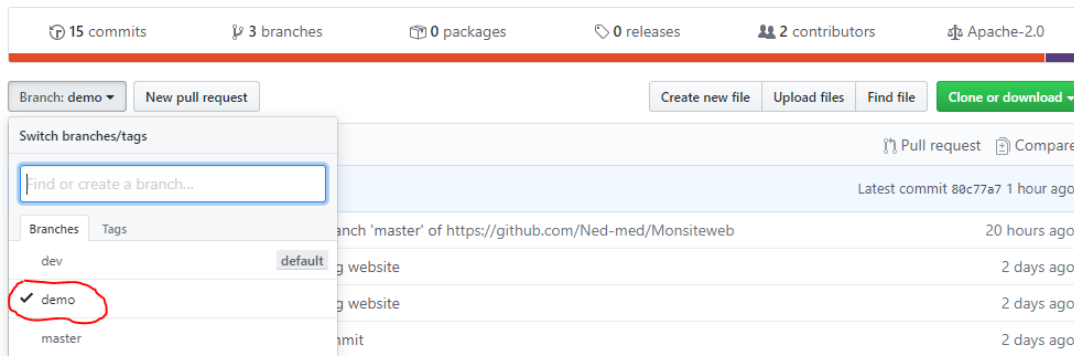
3. Sur github créez une branche du nom "Dev"



4. La Branche Dev deviendra la branche par défaut ; sur le menu « settings » mettez la branche Dev en mode par défaut



5. Créez une autre branche « demo », on va la considérer comme « feature ».

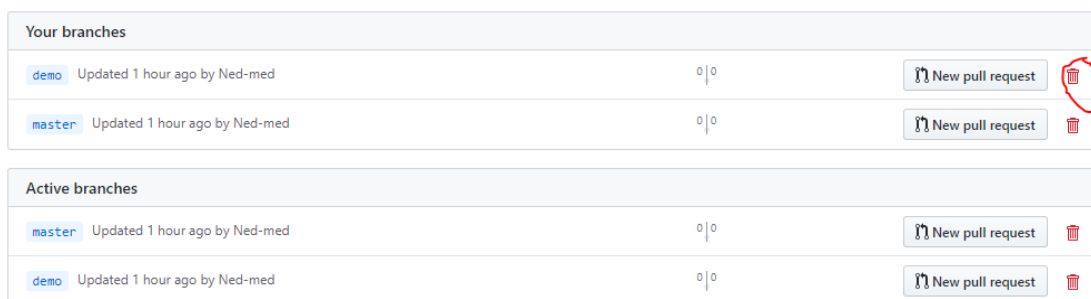


6. Avant de faire un pull Request sur le branche demo , Editez le fichier « README.md » , Ajoutez une ligne

7. Revenez à la page d'accueil de votre repo ? à votre avis on a besoin dans ce cas de faire un merge ou bien un rebase ? pourquoi ?

Dans ce cas on a besoin de faire un rebase car ce dernière compresse tous les changement sur une seule patch et après intégrer le dans le branch destinataire et de plus il élimine l'historique non désiré

8. Ne faites aucun merge ni Rebase mais plutôt supprimez la branche feature depuis GITHUB



9. Sur GIT, supprimez récursivement et localement votre repo website

```
# rm -rf Monsiteweb-local
```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ rm -rf Monsiteweb-local/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ ls
demo/  initializr/

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ |

```

10. Maintenant Clonez via https ou SSH si vous l'avez réussi

```
# git clone https://github.com/Ned-med/Monsiteweb.git
```

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj
$ git clone https://github.com/Ned-med/Monsiteweb.git
Cloning into 'Monsiteweb'...
remote: Enumerating objects: 70, done.
remote: Counting objects: 100% (70/70), done.
remote: Compressing objects: 100% (54/54), done.
remote: Total 70 (delta 27), reused 45 (delta 13), pack-reused 0
Unpacking objects: 100% (70/70), done.

```

11. Vérifiez si le contenu est récupéré

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ ls
apple-touch-icon.png  css/          favicon.ico  index.html  LICENSE  tile.png
browserconfig.xml    error404.html fonts/       js/         readme.md  tile-wide.png

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)

```

12. Vérifiez les branches et ou pointe le pointeur : HEAD

```

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ git hist
* 80c77a7 (HEAD -> dev, origin/master, origin/dev, origin/HEAD) adding update to readme file
* de8b938 Update From V2.0 to V3.0
* 299dcab update apache version to 3.0
* 67b0daf Merge branch 'master' of https://github.com/Ned-med/Monsiteweb
|
| * 2545998 Merge pull request #1 from Ned-med/annulation
| |
| | * 73d3187 annulation de h1
| | * 5dcbb3c Rename 404.html to error404.html
| | * 8415c7b Update index.html
| | * aa7dd9f Create style.css
| |
| |
| * 7a4808d update main.css
| |
| |
| * 6b9d899 Merge branch 'master' of https://github.com/Ned-med/Monsiteweb
| |
| |
| * a7db2f8 changed the title
| * 71417c9 more changes to readme.md
| |
| |
| * c60efa3 committing website
| * 56f7a00 Initial commit
|
|

```

13. Maintenant on a besoin de la branche master comme base : git checkout master

```
#git checkout master
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ git checkout master
Switched to a new branch 'master'
Branch 'master' set up to track remote branch 'master' from 'origin'.

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (master)
$ |
```

14. Toutefois la branche par défaut est : dev

#Quick Challenge: Try and Catch ()

On va imaginer le mini Scénario suivant :

1. Sur GitHub, on va éditez le fichier Readme.md

Branch: dev ▼ Monsiteweb / readme.md Find file Copy path

Ned-med Update readme.md 9481b38 now

2 contributors

3 lines (3 sloc) 60 Bytes Raw Blame History

#Made More Changes made more changes to readme file Updates

2. Localement on est sur la branche DEV.
3. On a fait des changements sur le REPO distant, surement on a besoin d'un fetch

```
#git fetch
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/Ned-med/Monsiteweb
 80c77a7..9481b38 dev -> origin/dev
```

4. On va provoquer un sous-entendu, on va éditez le fichier README.md localement en sachant qu'on a déjà modifié le fichier sur le repo distant.

```
#echo "##my local changes to readme" >> readme.md
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ echo "##my local changes to readme" >> readme.md

Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ git status
On branch dev
Your branch is behind 'origin/dev' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   readme.md

no changes added to commit (use "git add" and/or "git commit -a")
```

5. On va faire le commit

```
#git commit -am "adding one line to readme"
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ git commit -am "adding one line to readme"
warning: LF will be replaced by CRLF in readme.md.
The file will have its original line endings in your working directory
[dev 65d6d76] adding one line to readme
1 file changed, 1 insertion(+)
```

6. Normalement, un des membres de l'équipe demande un pull : git pull

```
#git pull
```

```
Administrateur@MEDMG MINGW64 ~/Downloads/CourseImplimentation/BreifProj/Monsiteweb (dev)
$ git pull
Auto-merging readme.md
Merge made by the 'recursive' strategy.
 readme.md | 1 +
1 file changed, 1 insertion(+)
```

7. Un message s'affiche indiquant que GIT n'arrive pas à traquer les informations et qu'il y'a un problème du merging, vu que le merge automatique n'est pas autorisé
Nous n'avons pas apporté de modifications sur la même ligne du fichier readme.md, c'est pourquoi au lieu d'avoir un conflit, nous avons eu une fusion automatique avec une stratégie récursive.

```
#Made More Changes
made more changes to readme file
Updates
##my local changes to readme
```

Conclusion

En conclusion, Git est un outil performant et simple d'utilisation, qui facilitera grandement le travail en équipe au sein de votre groupe durant le développement de votre site web. Les principales difficultés liées à l'utilisation de Git viennent de la résolution des éventuels conflits entre vos modifications. Ainsi que le site GitHub permet de se servir plus facilement des fonctionnalités de git.

Ce Brief nous a permis de mieux comprendre le flux de travail de git et de GitHub. Sur le chemin de la réalisation des scénarios, nous avons rencontré quelques problèmes mineurs, mais nous les avons surmontés en utilisant des documents git et en nous expliquant les choses les uns aux autres.