

「YouCode」

Rapport D'activités des Scénarios



Réalisé par :
HAMZA ELGHANDOUR

Encadré par :
Hanane Jabane



Sommaire :

Introduction :	3
Outils de travail :	4
-la méthode scrum :	4
-Trello :	4
Git :	4
Github :	5
Réalisation de scénarios :	Erreur ! Signet non défini.
Scenario 1 :	Erreur ! Signet non défini.
Scenario 2 :	Erreur ! Signet non défini.
Scenario 3 :	Erreur ! Signet non défini.
Dictionnaire de commande :	Erreur ! Signet non défini.
Conclusion :	Erreur ! Signet non défini.



Introduction :

Dans le cadre de la validation des compétences de la période SAS, le brief projet est un moyen utile pour valider 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.

Le but est également d'apprendre à gérer un projet professionnel, de la reformulation du cahier des charges jusqu'à sa réalisation.

Vous verrez également dans ce projet une application direct des compétences acquises pendant ces trois premières semaines de formation à youcode notamment en ce qui concerne les outils git, GitHub et Trello.

Dans les ligne qui suivent nous allons détailler le développement de chacune de ces parties pour que le lecteur de ce rapport soit éclairé sur les différents étapes de ce brief projet.



Outils de travail :

-la méthode scrum :

est certainement la plus connue des méthodologies agiles. Utilisée depuis 1993, elle fournit un cadre de gestion de projet avec des rôles, des réunions, des artefacts, des règles de gestion et un cycle itératif de développement. Le cadre de travail Scrum est simple, transparent et pragmatique. Pour résumer simplement, on définit le contenu d'une itération (ou « sprint scrum ») en termes de fonctionnalités, elles sont développées, puis validées à l'issue du sprint. Un bilan du sprint écoulé est réalisé avant de continuer sur le sprint suivant.

-Trello :

Trello est un outil collaboratif gratuit, permettant de travailler en ligne et à distance sur des projets communs. Les projets sont gérés au travers de tableaux, où sont créées des listes et des cartes basées sur le principe du kanban.

Au sein des listes créées sur Trello, on insère des cartes qui définiront les différentes étapes ou process d'un projet. Ces cartes peuvent contenir de multiples types contenus ou informations, comme du texte, des images, des liens hypertextes, des pièces jointes etc...

Chaque carte peut être associée à des utilisateurs distincts s'ils sont invités sur le projet, et/ou à des informations utiles comme l'ajout d'une date, d'un code couleur...

Git :

Créé par Linus Torvalds qui a créé le noyau LINUX en 2005.

C'est un logiciel open-source pouvant être téléchargé pour les plates-formes Linux, Windows, Solaris et Mac.

Git est un outil de versioning qui permet de tracer l'évolution de votre projet et d'y apporter des modifications sereinement.

Il permet d'avoir l'historique de l'avancement et les modifications du projet en cours de création.

Ce visionner code est un outil pour collaborer et travailler en équipe.



Github :

Créé en 2008 par l'entrepreneur américain Tom Preston-Werner

C'est une plateforme web qui host les GIT permettant de créer un dépôt distant et partager les projets pour faciliter le travail collaboratif

GitHub permet d'utiliser le versioning de Git sans avoir à apprendre Git qui se gère à la ligne de commande donc à "l'ancienne".

「YouCode」

「YouCode」

Realisation de scenario :

Premier scenario :

#1 : First Step : Immersion

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop (master)
$ mkdir projects
```

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects (master)
$ cd demo/
```

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git init
Initialized empty Git repository in C:/Users/youcode/Desktop/projects/demo/.git/
```

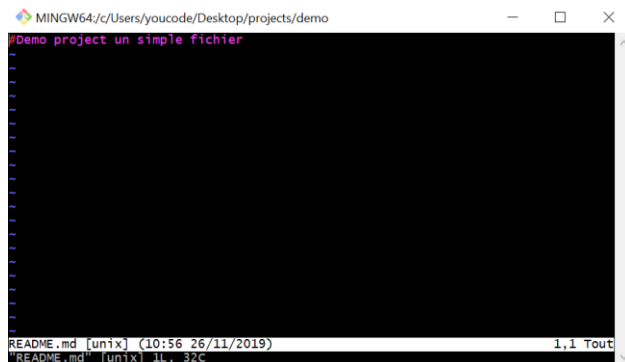
```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git status
On branch master

No commits yet

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

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ touch README.md

youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ vim README.md
```



「YouCode」

```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/demo (master)
$ git add .

Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/demo (master)
$ git commit -m "README MODIF"
[master (root-commit) d4eb55c] README MODIF
1 file changed, 1 insertion(+)
create mode 100644 README.md

Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/demo (master)
$
```

On a créé un répertoire nommée projects, un repo local sous le nom demo. Ensuite on a créé un fichier dans ce repo et on a modifier ce fichier. Et enfin le staging et le committing. Le staging area est une sorte de zone de transit où se trouve les fichiers modifiés qui sont pris en compte pour le prochain commit.

#2 : Second Step : La découverte

1) Déplacement les fichiers de configuration « .git ».

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ cd .git

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo/.git (GIT_DIR!)
```

2) Le Tape Ls -al

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo/.git (GIT_DIR!)
$ ls -al
total 17
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:47 ./
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:47 ../
-rw-r--r-- 1 youcode 197121 13 nov. 26 11:47 COMMIT_EDITMSG
-rw-r--r-- 1 youcode 197121 130 nov. 26 11:46 config
-rw-r--r-- 1 youcode 197121 73 nov. 26 11:46 description
-rw-r--r-- 1 youcode 197121 23 nov. 26 11:46 HEAD
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:46 hooks/
-rw-r--r-- 1 youcode 197121 137 nov. 26 11:47 index
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:46 info/
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:47 logs/
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:47 objects/
drwxr-xr-x 1 youcode 197121  0 nov. 26 11:46 refs/
```

3) Explication des clauses suivantes : HEAD, LOGS, BRANCHES:

- Logs: endroit contenant les commits effectués et sa commande permet de les afficher.
- Branches: endroit contenant les branches effectués
- Head : le commit en cour où se trouve votre répo. Souvent le HEAD pointe vers le derniers commit dans notre branche en cour. En gros HEAD veut dire: vers quoi mon répo pointe actuellement

4) Création du fichier "Licence.md" dans notre répo:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo/.git (GIT_DIR!)
$ cd ..

youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ touch Licence.md
```


「YouCode」

5) Faire le commit:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git add .

youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git commit -m "add licence"
[master 928f7a1] add licence
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Licence.md
```

6) Afficher les fichiers traqués:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git ls-tree -r master --name-only
Licence.md
README.md
```

Dans cette étape on a créé un fichier licence.md puis on a comité. Ensuite on doit afficher les fichiers traqués (les fichiers qui se trouve dans la staging area).

#3 : third step : Historique

1) L'affichage du dernier commit sur une ligne en ajoute l'option d'affichage de la hiérarchie de la branche, avec les commits et leur branche aussi

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git log -1 --oneline
c8aa1be (HEAD -> master) add licence.md
```

2) Création un alias de la commande précédente le nom de l'alias est : historique

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git config --global alias.historique 'log -1 --oneline'

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git historique
c8aa1be (HEAD -> master) add licence.md
```

3) Affichage de la liste des alias

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git config --get-regexp alias
alias.historique log -1 --oneline
```

4) Affichage de l'historique des commit du fichier README.md avec l'alias.

「YouCode」

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git historique 40bd39
40bd396 README modif
```

Git permet la définition des alias pour chaque commande afin d'éviter de taper l'intégralité de la commande, c'est un moyen qui facilite le travail sous git. Cette technique peut aussi être utile pour créer des commandes qui nous manquent.

Dans cette partie et pour se familiariser avec l'alias, on a créé un alias appelé « historique ». Ensuite on affiche la liste des alias et l'historique de commit du fichier readme.md avec l'alias.

#4 : Fourth Step : Excluding files

- 1) Renommer le fichier Licence.md à Licence.txt:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ mv Licence.md Licence.txt
```

- 2) Faire le staging avec mise à jour (ne pas faire « git add . »)

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git add Licence.txt
```

- 3) Faire le commit:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git commit -m 'change name licence'
[master 99c830c] change name licence
1 file changed, 0 insertions(+), 0 deletions(-)
rename Licence.md => Licence.txt (100%)
```

- 4) Créer un fichier nommé application.log:

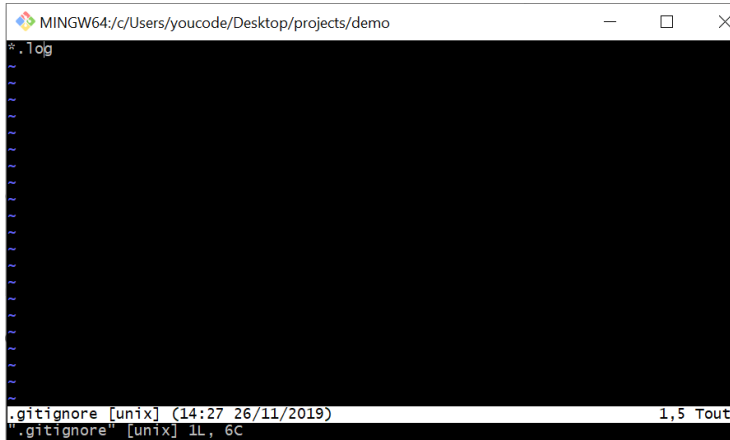
```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/demo (master)
$ touch application.log
```

- 5) Sans staging, créer un fichier nommé « .gitignore » :

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ touch .gitignore
```

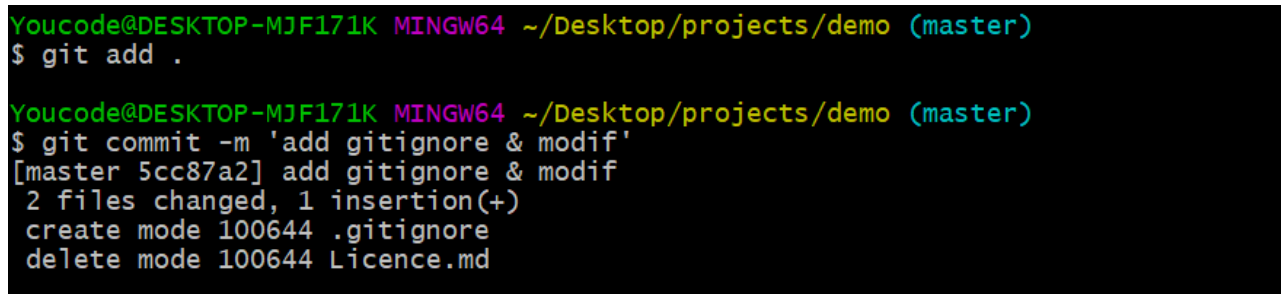
「YouCode」

6) Sur le fichier .gitignore Ajouter la ligne suivante « *.log » :



The screenshot shows a terminal window titled 'MINGW64: c:/Users/youcode/Desktop/projects/demo'. The file '.gitignore' is open, displaying the content '*.log'. The status bar at the bottom indicates '.gitignore [unix] (14:27 26/11/2019)' and '1,5 Tout'.

7) Faire le staging et le commit :



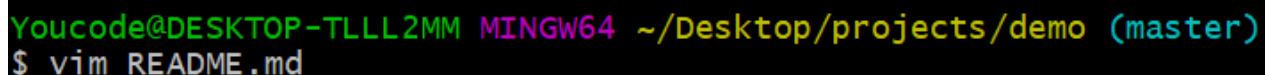
```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/demo (master)
$ git add .

Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/demo (master)
$ git commit -m 'add gitignore & modif'
[master 5cc87a2] add gitignore & modif
2 files changed, 1 insertion(+)
create mode 100644 .gitignore
delete mode 100644 Licence.md
```

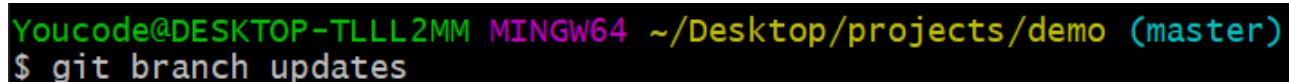
Si on veut exclure certains fichier ou répertoire du suivi par git on doit créer un fichier « .gitignore ».

Lorsqu'un fichier ou un répertoire est ignoré il ne sera pas suivi par git, signale par des commandes telles git status ou git diff ni mis en scene avec des commandes telles que git add.

#5 : fifth Step : Branching and Merging



```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ vim README.md
```



```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git branch updates
```

「YouCode」

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (updates)
$ git historique
* 8105365 (HEAD -> updates) change readme
* d90174b (master) create gitignore
* fd29e97 renommer licence
* b41878b create licence
* 6ed2ee1 create readme
```

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git historique
* 8105365 (updates) change readme
* d90174b (HEAD -> master) create gitignore
* fd29e97 renommer licence
* b41878b create licence
* 6ed2ee1 create readme
```

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git merge updates
Updating d90174b..8105365
Fast-forward
 README.md | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Ce qui permet de faire le « versionning » c'est le système de branche. Lorsque on veut essayer de nouvelle future on crée une branche pour diverger de la ligne principale de développement, si les modifications marchent bien ou peut les garder sinon on peut les abandonner

Lorsque on crée une branche, ce n'est qu'une copie du repo master, il contient le même historique de la branche principale.

Le merging permet le fusionnement entre la branche principale et celle qui contient les changements quand veut ajouter au projet

#6 sixth Step : Conflict Resolution

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ vim README.md
```

「YouCode」

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git checkout BAD
Switched to branch 'BAD'
M       README.md
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (BAD)
$ git commit -am 'modif readme 2'
[BAD 4a59673] modif readme 2
1 file changed, 2 insertions(+), 1 deletion(-)
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (BAD)
$ git checkout master
Switched to branch 'master'
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ vim README.md
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ cat README.md
#Demo project un simple fichier modifié
Troubleshooting
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git commit -am 'branche bien faite'
[master 50418e6] branche bien faite
1 file changed, 2 insertions(+), 1 deletion(-)
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git commit --amend
[master f630fe7] Modif README 3
Date: Tue Nov 26 16:07:57 2019 +0100
1 file changed, 2 insertions(+), 1 deletion(-)
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git historique
f630fe7 (HEAD -> master) Modif README 3
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/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.
```

「YouCode」

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master|MERGING)
$ cat README.md
#Demo project un simple fichier modifié
<<<<<<< HEAD
Troubleshooting
=====
Trouble
>>>>>>> BAD
```

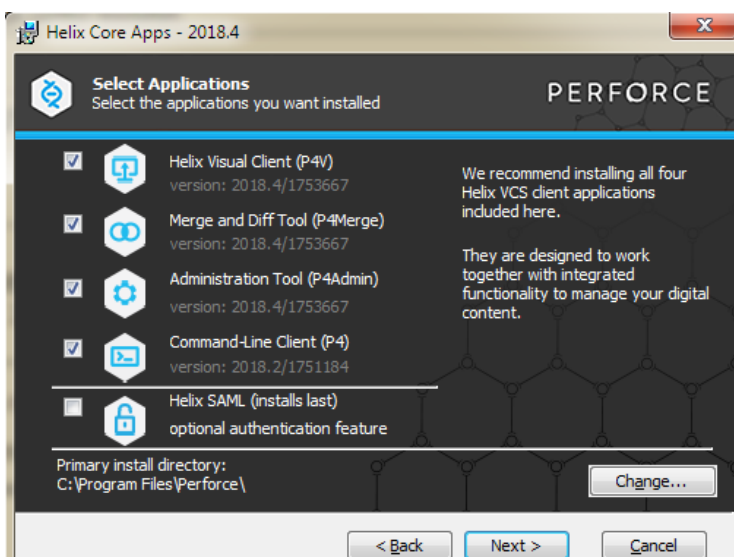
```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master|MERGING)
$ git mergetool
Merging:
README.md

Normal merge conflict for 'README.md':
  {local}: modified file
  {remote}: modified file
The merge tool p4merge is not available as 'C:\Program Files\Perforce\p4merge.exe'
```

Lorsque on fait deux modification sur le même fichier et sur la même ligne, la première sur la branche principale et la deuxième sur une autre branche (BAD), on constate qu'il est impossible de faire le fusionnement (merge).

#7: seventh Step: merge tools

1) L'Installation de p4merge sur notre Pc :



「YouCode」

1) Git config --global merge.tool p4merge

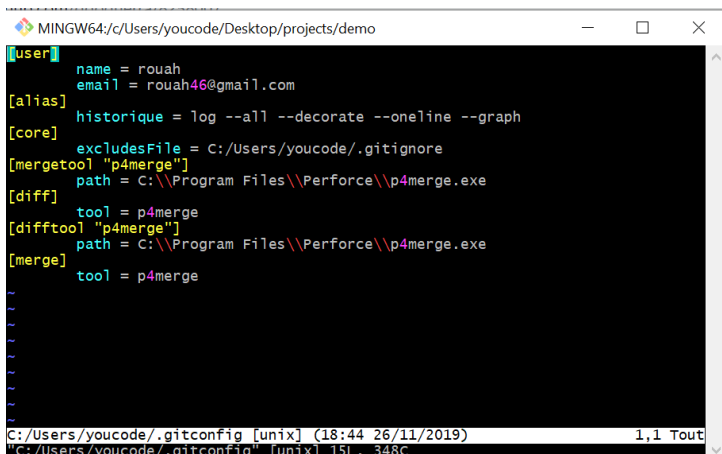
```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master|MERGING)
$ git config --global merge.tool p4merge
```

2) Git config --global mergetool.p4merge.path "lien d'installation" (.exe)

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master|MERGING)
$ git config --global mergetool.p4merge.path "C:\Program Files\Perforce\p4merge.exe"
```

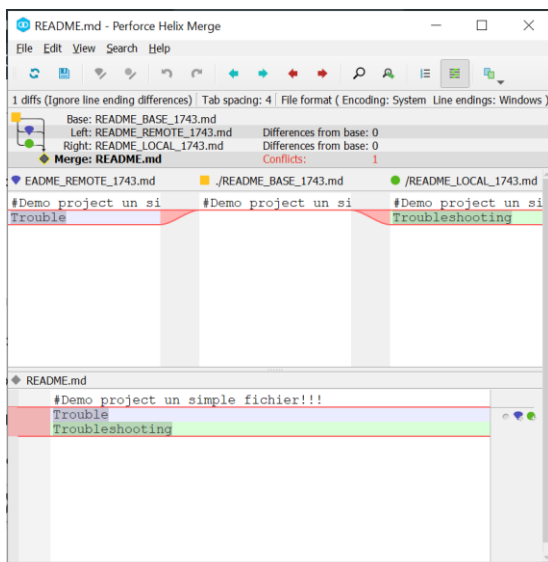
3) Configuration du prompt :

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master|MERGING)
$ git config --global --edit
```



```
MINGW64/c:/Users/youcode/Desktop/projects/demo
[user]
name = rouah
email = rouah46@gmail.com
[alias]
historique = log --all --decorate --oneline --graph
[core]
excludesFile = C:/Users/youcode/.gitignore
[mergetool "p4merge"]
path = C:\Program Files\Perforce\p4merge.exe
[diff]
tool = p4merge
[difftool "p4merge"]
path = C:\Program Files\Perforce\p4merge.exe
[merge]
tool = p4merge
~
~
~
~
C:/Users/youcode/.gitconfig [unix] (18:44 26/11/2019) 1,1 Tout
"C:/Users/youcode/.gitconfig" [unix] 15L, 348C
```

4) Taper la commande git mergetool :





5) Analyser et expliquer la plateforme ouverte en temps réel :

On obtient la plateforme ci dessus qui affiche en haut à gauche la modification apportée à la branche master, la partie en haut à droite représente la branche BAD. Celle du milieu est celle où devrait se trouver le merge des deux modifications.

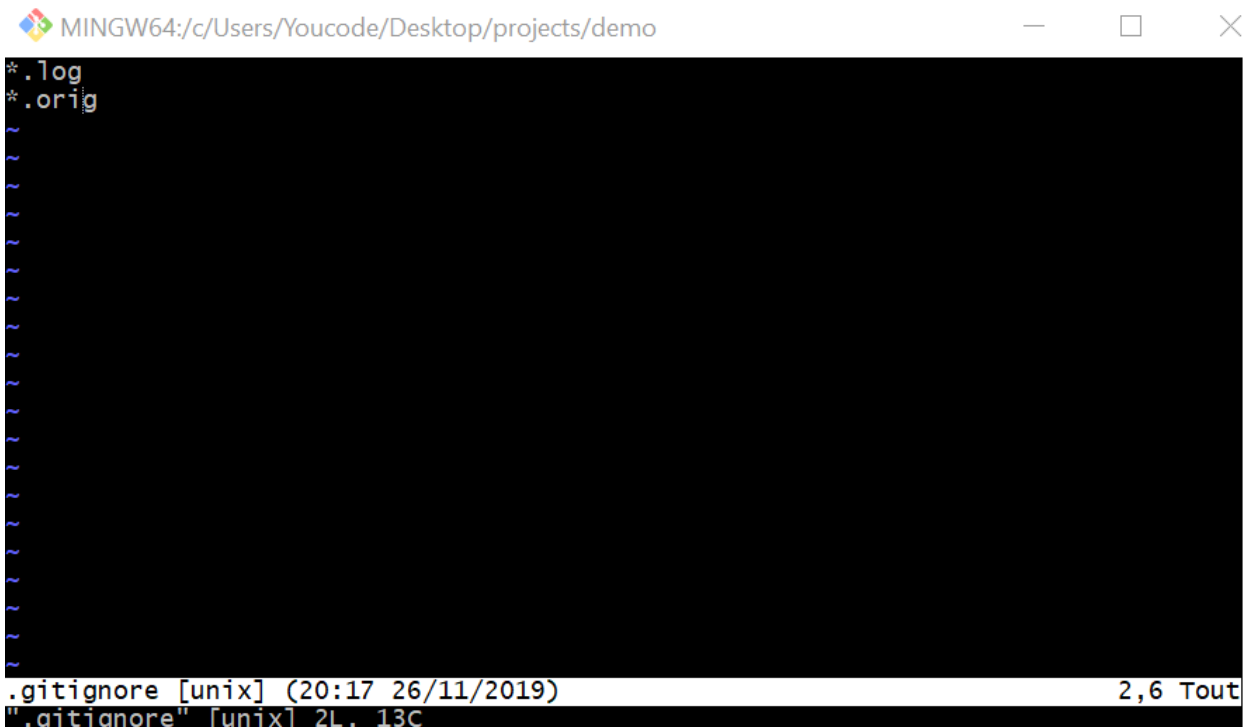
La partie du bas et celle qu'on peut modifier manuellement pour résoudre le conflit du merge.

#8: eighth Step: Challenge

1) Sur le fichier **.gitignore** ; écrire une clause pour rejeter les fichiers indésirables et redondants

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master|MERGING)
$ vim .gitignore
```





Scénario #2

#1 : First Step : Tagging

1. Se déplacer sur la branche Principale:
2. Créer un TAG avec un nom V1.0 et un commentaire ' REALEASE 1.0 '

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git tag -a v1.0 -m 'REALEASE 1.0'
```

3. Afficher les informations sur le TAG:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git show v1.0
tag v1.0
Tagger: rouah <rouah46@gmail.com>
Date:   Wed Nov 27 10:59:12 2019 +0100

REALEASE 1.0

commit 37de517c88345990c307a9e87f176a94b4257236 (HEAD -> master, tag: v1.0)
Merge: 13e1bff 4009b2f
Author: rouah <rouah46@gmail.com>
Date:   Wed Nov 27 09:44:59 2019 +0100

4

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

Tag : Git donne la possibilité d'étiqueter un certain état dans l'historique comme important. Généralement, les gens utilisent cette fonctionnalité pour marquer les états de publication (v1.0 et ainsi de suite). Dans cette section, nous avons comment lister les différentes étiquettes comment créer de nouvelles étiquettes et les différents types d'étiquettes.

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

- 1) Modifier le fichier README.md et ajouter une ligne:

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
vim README.md
```



2) Tapez la commande git Stash ?

```
youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git stash
Saved working directory and index state WIP on master: dd35bd6 modif licence
```

3) Taper la commande git stash list

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git stash list
stash@{0}: WIP on master: 37de517 4
```

4) Exécuter la commande git status

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git status
On branch master
nothing to commit, working tree clean
```

5) Conclusion:

6) Modifier le fichier « Licence.txt » et ajouter la ligne « APACHE 2.0 »

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ vim Licence.txt
```

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/projects/demo (master)
$ git commit -am 'modif licence'
warning: LF will be replaced by CRLF in Licence.txt.
The file will have its original line endings in your working directory
[master 7689cd3] modif licence
1 file changed, 1 insertion(+)
```

```
youcode@DESKTOP-KVTCEUL MINGW64 ~/Desktop/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} (b5f16eef6b7539dd7086d360bbe8c0ecb64f0310)
```

「YouCode」

Stach : git stach est une commande qui permet de mettre de cote des modifications.

il récupéré les **modifications en cours** et les enregistrer dans un **conteneur** qu'on appellera un **stash**.

Git stach pop permet de réintégrer les donnes du dernier stach , par défaut il va le supprimer une fois réintégrer .

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ git push -u origin master --tags
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 8 threads
Compressing objects: 100% (20/20), done.
Writing objects: 100% (28/28), 2.43 KiB | 311.00 KiB/s, done.
Total 28 (delta 6), reused 0 (delta 0)
remote: Resolving deltas: 100% (6/6), done.
To https://github.com/SaraMarhoum/Brief-scenario-2.git
 * [new branch]      master -> master
 * [new tag]         v1.0 -> v1.0
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

dans cette étape on a crée un repo sur github, puis on a « pusher » le contenu du repo local vers github, la raison de cette étape et de savoir lié entre le repo local et le repo a distance et de savoir les différentes commande qui permet la coordination entre ses deux ou



#4 : Fourth Step : Mini challenge (optionnel)

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.

1) Vérifier l'existence d'une clé ssh

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ ls -al ~/.ssh
ls: cannot access '/c/Users/Youcode/.ssh': No such file or directory
```

2) Générer une nouvelle clé SSH et l'ajouter à ssh-agent

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ ssh-keygen -t rsa -b 4096 -C "sara_maroum@hotmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Youcode/.ssh/id_rsa):
Created directory '/c/Users/Youcode/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Youcode/.ssh/id_rsa.
Your public key has been saved in /c/Users/Youcode/.ssh/id_rsa.pub.
The key fingerprint is:
...
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ eval $(ssh-agent -s)
Agent pid 1461

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/Youcode/.ssh/id_rsa (sara_maroum@hotmail.com)
```



```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ ls -al ~/.ssh
total 20
drwxr-xr-x 1 Youcode 197121 0 nov. 27 22:31 ./
drwxr-xr-x 1 Youcode 197121 0 nov. 27 22:30 ../
-rw-r--r-- 1 Youcode 197121 3389 nov. 27 22:31 id_rsa
-rw-r--r-- 1 Youcode 197121 749 nov. 27 22:31 id_rsa.pub
```

3) Ajouter une nouvelle clé SSH au compte GitHub

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ clip < ~/.ssh/id_rsa.pub
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ vim ~/.ssh/id_rsa.pub
```

4) Tester la connexion SSH

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/demo (master)
$ ssh -T git@github.com
The authenticity of host 'github.com (xxx)' can't be established.
RSA key fingerprint is xxx.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com,xxx' (RSA) to the list of known hosts.
Hi SaraMarhoum! You've successfully authenticated, but GitHub does not provide
shell access.
```

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

Copie locale (clone) : on a commencé par créer un repo à distance sur github ensuite on se déplace dans le répertoire projet sous git et on crée une copie .

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop (master)
$ cd projects
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects (master)
```

```
Youcode@DESKTOP-4BC5EH MINGW64 ~/Desktop/projects (master)
$ git clone https://github.com/khadija1998/monsitweb.git Monsitweb-local
Cloning into 'Monsitweb-local'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (7/7), done.
```

「YouCode」

Vérifier si le clone est créé

```
Youcode@DESKTOP-4BCSEH MINGW64 ~/Desktop/projects (master)
$ ls
demo/  Monsiteweb-local/
```



#6: sixth step : Sending the website :

1) Télécharger le site web

1 - Pre-configuration

Classic H5BP
[Docs](#) [Demo](#)

Responsive
[Docs](#) [Demo](#)

Bootstrap
[Docs](#) [Demo](#)

2) Sur le site telecharger un site bootstrap avec le fichier **.htaccess** et le fichier 404.html

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

H5BP Optional

<input type="checkbox"/> IE Classes	<input type="checkbox"/> Favicon	<input type="checkbox"/> Humans.txt
<input type="checkbox"/> Old browser warning	<input type="checkbox"/> Apple and Windows Icons	<input checked="" type="checkbox"/> 404 Page
<input 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) Analyser l'arborescence.

「YouCode」

PC > Bureau > initializr



Rechercher dans : initializr

<input type="checkbox"/> Nom	Modifié le	Type	Taille
 css	27/11/2019 16:06	Dossier de fichiers	
 img	27/11/2019 13:06	Dossier de fichiers	
 js	27/11/2019 16:06	Dossier de fichiers	
 .editorconfig	27/11/2019 13:06	Fichier EDITORCONFIG	1 Ko
 .gitattributes	27/11/2019 13:06	Document texte	1 Ko
 .gitignore	27/11/2019 13:06	Document texte	1 Ko
 .htaccess	27/11/2019 13:06	Fichier HTACCESS	39 Ko
 404	27/11/2019 13:06	Firefox Document	2 Ko
 index	27/11/2019 13:06	Firefox Document	1 Ko

「YouCode」

- 4) Copier le site télécharger dans le repo local à travers une seule commande :

```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ cp -R ~/DESKTOP/initializr/* .
```

- 5) Git status

```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/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
    css/
    fonts/
    index.html
    js/

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

- 6) Faire le staging et le commit en une seule ligne

```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git commit -am 'monsitweb'
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  404.html
  css/
  fonts/
  index.html
  js/

nothing added to commit but untracked files present
```

- 7) Faire le push à github

```
Youcode@DESKTOP-MJF171K MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git push origin master
Everything up-to-date
```

L'objet est de savoir comment copier des dossiers sur les repos locaux. On a commencé par copier le site télécharger dans le repo local à travers une seule commande, juste après on envoie les modifications au repo à distance sur GitHub.



#7: seventh step : Fetch and pull

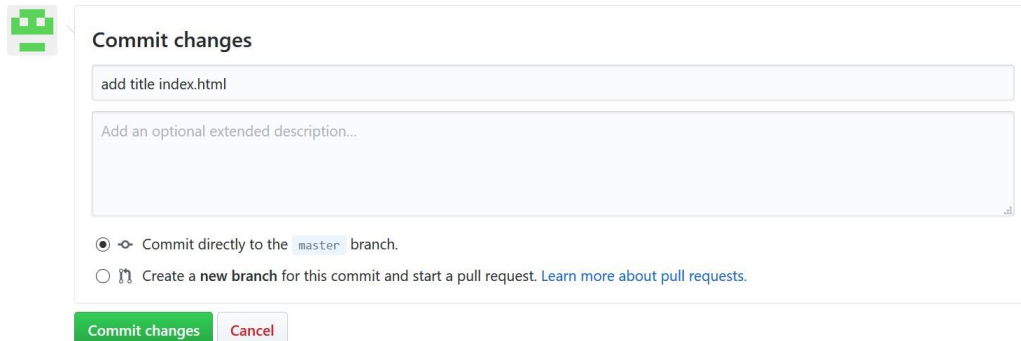
- 1) Sur Github éditer le fichier **Index.html**, sur la balise `<title>` `</title>` et ajouter le titre , mon premier site web .

```
91 lines (83 sloc) | 4.13 KB
Raw Blame History

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 sit web</title>
7     <meta name="description" content="">
8     <meta name="viewport" content="width=device-width, initial-scale=1">
9
10    <link rel="stylesheet" href="css/bootstrap.min.css">
11    <style>
12      body {
13        padding-top: 50px;
14        padding-bottom: 20px;
15      }
16    </style>
17    <link rel="stylesheet" href="css/bootstrap-theme.min.css">
18    <link rel="stylesheet" href="css/main.css">
19
20    <!--[if lt IE 9]>
21      <script src="js/vendor/html5-3.6-respons-1.4.2.min.js"></script>
22    <![endif]-->
23  </head>
24  <body>
25    <nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">
26      <div class="container">
```

「YouCode」

2) Faire le commit sur Github



3) Editer le fichier README.md, faire le staging et le commit en une seule ligne

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ vim README.md

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git commit -am 'edit README.md'
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory
[master b7a0c47] edit README.md
1 file changed, 2 insertions(+), 1 deletion(-)

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$
```

Trouble shooting :

1) Exécuter la commande « git fetch »

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/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/SaraMarhoum/Monsiteweb
05a1183..31981c0 master -> origin/master
```



2) Git status

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/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
```

「YouCode」

3) Git pull pour puller et merger

[illegible]

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git pull
Merge made by the 'recursive' strategy.
 index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

4) Git push

```
Youcode@DESKTOP-TLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git push origin master
Enumerating objects: 9, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 590 bytes | 98.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/SaraMarhoum/Monsiteweb.git
  31981c0..ae35257  master -> master
```

5) Vérifier les commit sur github

Scénario #3


#1: First Step: Changes on Github

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

```
25 lines (21 sloc) | 891 Bytes
Raw Blame History

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
10     <link rel="stylesheet" href="css/normalize.min.css">
11     <link rel="stylesheet" href="css/main.css">
12
13     <!--[if lt IE 9]>
14       <script src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.min.js"></script>
15       <script>window.html5 || document.write('<script src="js/vendor/html5shiv.js"></script>')</script>
16     <![endif]>
17   </head>
18   <body>
19     <h1> modification récente </h1>
20
21     <p>Hello world! This is HTML5 Boilerplate.</p>
22
23     <script src="js/main.js"></script>
24   </body>
25 </html>
```

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

 [error404.html](#) Renommer 404 to error404 4 hours ago

1. Sur gitHub vérifier la liste des commits .

2. Sur GITBASH, vérifier avec « git status »

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

3. Visualiser et télécharger les fichiers distants sur GITBASH

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/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 4), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (8/8), done.
From https://github.com/SaraMarhoum/Monsiteweb
 1b2e705..2149260  master    -> origin/master
```

4. Maintenant faire le pull et fusionner les changements distants avec le repo local

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git pull
Updating 1b2e705..2149260
Fast-forward
 css/style.css      | 1 +
 404.html => error404.html | 0
 index.html         | 1 +
 3 files changed, 2 insertions(+)
 create mode 100644 css/style.css
 rename 404.html => error404.html (100%)
```

5. Vérifiez le repo Local

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

6. Laissez un commentaire sur les fichiers que nous avons modifié sur GitHub en regardant la liste des commits.

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git log c0f275
commit c0f275b20f21f96e8eea392feb25324b7e9bb2ef
Author: SaraMarhoum <57989282+SaraMarhoum@users.noreply.github.com>
Date: Thu Nov 28 16:10:48 2019 +0100


    add balise h1 ds html
```

#2: Second Step: Branching and merging sur GITHUB

Monsiteweb / README.md Cancel

<> Edit file Preview changes Spaces 2 Soft wrap

```
1 # Monsiteweb
2 edit file
3 2nd file edit (sc 3)
```



Commit changes

3th senario readme edit

Add an optional extended description...

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

Commit changes Cancel

a) Sur github et sur le menu 'branch' ; expliquer le message:

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

Overview Yours Active Stale All branches Search branches...

Default branch

master Updated 5 hours ago by SaraMarhoum Default Change default branch

Your branches

Example Updated 3 minutes ago by SaraMarhoum 0 | 1 New pull request

Active branches

Example Updated 3 minutes ago by SaraMarhoum 0 | 1 New pull request

On observe un message décrivant le temps de la dernière mise à jour sur la branche et par qui.

2. Repo local :

- Sur le repo local, créer une branche nommé 'annulation'.

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch annulation

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch --list
annulation
* master
```

- Sur le repo local éditer le fichier « index.html » et supprimer la balise H1.

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git checkout annulation
Switched to branch 'annulation'

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (annulation)
$
```

MINGW64:/c/Users/Youcode/Desktop/projects/Monsiteweb-local

```
<title>mon premier site web </title>
<meta name="description" content="">
<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="css/normalize.min.css">
<link rel="stylesheet" href="css/main.css">

<!--[if lt IE 9]>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.min
.js"></script>
  <script>window.html5 || document.write('<script src="js/vendor/html5shiv.js"></s
cript>')</script>
<![endif]-->
</head>
<body>
  <h1> modifica

  <p>Hello world! This is HTML5 Boilerplate.</p>

  <script src="js/main.js"></script>
</body>
</html>
index.html[+] [dos] (16:25 28/11/2019) 19,22 Bas
-- INSERTION --
```

- c) Faire le commit et le staging en une seule ligne.

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (annulation)
$ git commit -am 'html modif in annulation branch'
```

- d) « Git show » pour vérifier nos changements.

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (annulation)
$ git show
commit bec829ed7a5d4680849a0e68a81f68034c358552 (HEAD -> annulation)
Author: sara <sara_maroum@hotmail.com>
Date: Thu Nov 28 20:56:55 2019 +0100

    modif html annulation branch

diff --git a/index.html b/index.html
index 37e548c..a8d2f2f 100644
--- a/index.html
+++ b/index.html
@@ -16,7 +16,7 @@
     <![endif]-->
   </head>
   <body>
-    <h1> modification récente </h1>
+
   <p>Hello world! This is HTML5 Boilerplate.</p>
```

3. Vérification :

- a) Sur GITBASH Pusher les changements que nous avons sur la branche 'annulation' à notre repo distant:

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (annulation)
$ git push origin annulation
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), 295 bytes | 98.00 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/SaraMarhoum/Monsiteweb/pull/new/annulation
remote:
To https://github.com/SaraMarhoum/Monsiteweb.git
 * [new branch]      annulation -> annulation
```

- b) Maintenant vérifier le repo distant.

[Code](#)
[Issues 0](#)
[Pull requests 0](#)
[Actions](#)
[Projects 0](#)
[Wiki](#)
[Security](#)
[Insights](#)
[Settings](#)

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

[Manage topics](#)

8 commits
3 branches
0 packages
0 releases
1 contributor
Apache-2.0

Your recently pushed branches:

Example (26 minutes ago)
 [Compare & pull request](#)

annulation (1 minute ago)
 [Compare & pull request](#)

Default branch

master Updated 5 hours ago by SaraMarhoum
 Default
Change default branch

Your branches

annulation Updated 8 minutes ago by SaraMarhoum	0 1	New pull request	Delete
Example Updated 26 minutes ago by SaraMarhoum	0 1	New pull request	Delete

La création de la branche et la modification apporté au fichier apparaissent tous les deux sur le depo distant.

#3: Third Step: compare pull Requests

Sur notre repo Distant, un message indiquant qu'une branche a été ajouté avec un bouton « compare and pull request »

Conclusion:

On trouve sur cette

Cette page nous offre l'option de réaliser une demande Pull, un merge des branches et affiche la modification réalisée dans le branche créée.


Créez un pull request avec le commentaire 'annulation à vérifier'

Une page s'ouvre, explorez la page ?

qu'est-ce que vous constatez ?

Sur l'onglet 'conversation' vous pouvez aussi laisser des commentaires à la personne qui demande le pull request ; laissez un commentaire ?

Maintenant si tout va bien ; aucun message d'erreur effectuez le merging ?



Merge pull request #1 from SaraMarhoum/annulation


Annulation à vérifier

[Confirm merge](#) [Cancel](#)



Write Preview

AA B i “ <> 🔗 ☰ ☷ ✓ @ 📌 ↶





Pull request successfully merged and closed

You're all set—the `annulation` branch can be safely deleted.

[Delete branch](#)

Supprimez la branche 'annulation' une fois tout est OK.





 SaraMarhoum deleted the `annulation` branch now

[Restore branch](#)

Annulation à vérifier #1

[Edit](#)

 Open SaraMarhoum wants to merge 1 commit into `master` from `annulation` 



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



SaraMarhoum commented 5 minutes ago

Owner +😊 ...


No description provided.


  modif html annulation branch

bec829e

Add more commits by pushing to the `annulation` branch on SaraMarhoum/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](#).



Write Preview

AA B i “ <>     @  

Leave a comment


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

 Close pull request

Comment

Reviewers 

No reviews

Assignees 

No one—assign yourself

Labels 

None yet


Projects 

None yet

Milestone 

No milestone


Notifications [Customize](#)

 Unsubscribe

You're receiving notifications because you're watching this repository.

1 participant



 Lock conversation

#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% }
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch ajustement

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git checkout ajustement
Switched to branch 'ajustement'

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (ajustement)
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (ajustement)
$ vim main.css
```

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (ajustement)
$ git commit -am 'modif main css in ajustment'
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (ajustement)
$ git show
commit d9eb687f91df322dcbbf119c5abb83e3991bec95 (HEAD -> ajustement)
Author: sara <sara_maroum@hotmail.com>
Date: Thu Nov 28 23:23:03 2019 +0100
```

```
    modif main css in ajustment


diff --git a/css/main.css b/css/main.css
index 95751ea..f4382c2 100644
--- a/css/main.css
+++ b/css/main.css
@@ -195,4 +195,9 @@ textarea {
    h3 {
        page-break-after: avoid;
    }
-}
\ No newline at end of file
+
+    background: #b3d4fc;
+
+    body {          width : 100% ;      Height : 100% }
+
+}
```


```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (ajustement)
$
```

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (ajustement)
$ git push origin ajustement
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 8 threads
Compressing objects: 100% (33/33), done.
Writing objects: 100% (37/37), 15.09 KiB | 376.00 KiB/s, done.
Total 37 (delta 12), reused 0 (delta 0)
remote: Resolving deltas: 100% (12/12), done.
remote:
remote: Create a pull request for 'ajustement' on GitHub by visiting:
remote:   https://github.com/SaraMarhoum/Monsiteweb/pull/new/ajustement
remote:
To https://github.com/SaraMarhoum/Monsiteweb.git
 * [new branch]      ajustement -> ajustement
```

Sur votre GITHUB, un pull request est demandé !

Your recently pushed branches:

 ajustement (2 minutes ago)

 Compare & pull request

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

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

yes

2. Un pull Request est demandé sur GITHUB ! N'oubliez pas !

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (ajustement)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (master)
$
```

explication... la branche ajustement est a jour avec la branche master ?

4. Maintenant demandez le pull ?

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (master)
$ git pull https://github.com/SaraMarhoum/Monsiteweb.git
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), done.
From https://github.com/SaraMarhoum/Monsiteweb
* branch          HEAD       -> FETCH_HEAD
Updating 2149260..a3cff8c
Fast-forward
 index.html | 2 + -
 1 file changed, 1 insertion(+), 1 deletion(-)
```

5. Une fois la commande du pull est exécutée, à votre avis quel type de merge on aura besoin pour faire un merge sans conflits
???

6. Faites le merge à travers GITBASH

7. Une fenêtre s'ouvre après l'exécution de la commande du merge ; ce qui signifie le message délivré ? à quoi sert cette fenêtre ?

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (master)
$ git branch -a
  ajustement
  annulation
* master
remotes/origin/HEAD -> origin/master
remotes/origin/ajustement
remotes/origin/annulation
remotes/origin/master
```

explication..... la suppression exécutée sur Github mais branches toujours présentes sur Git ?

2. Sur GITBASH supprimez les branches que vous avez créé.

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (master)
$ git branch -d ajustement annulation
Deleted branch ajustement (was d9eb687).
Deleted branch annulation (was bec829e).
```

3. Maintenant ! Tapez la commande git branch -a ? qu'est-ce que vous constatez ?

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local/css (master)
$ git branch -a
* master
remotes/origin/HEAD -> origin/master
remotes/origin/ajustement
remotes/origin/annulation
remotes/origin/master
```

les branches ont été enfin supprimées suite à la commande sur git

4. 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 ?

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git pull origin master
From https://github.com/SaraMarhoum/Monsiteweb
* branch      master      -> FETCH_HEAD
Already up to date.
```

5. Tapez la commande git branch -a ? qu'est-ce que vous constatez ?


????

#5: fifth Step: The Cleaning up

Dans cette phase je vous laisse découvrir l'objectif ?

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

Your recently pushed branches:

 updatellicence (less than a minute ago) [Compare & pull request](#)

Branch: updatellicence ▾ [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download ▾](#)

Sur GITBASH faites un pull global : `git pull --all` ;

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git pull --all
Fetching origin
Already up to date.
```

Maintenant faites le merge de la branche 'updatellicence'. Exécutez Git push !
Git branch -d 'updatellicence'

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git merge updatellicence
Already up to date.

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git add .

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git commit -m 'merge branch'
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git push origin master
Everything up-to-date

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch -d 'updatellicence'
Deleted branch updatellicence (was 3e78011).
```

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/ajustement
  remotes/origin/annulation
  remotes/origin/master
  remotes/origin/updatelicense
```

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 ?

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git push origin master -d updatelicense
To https://github.com/SaraMarhoum/Monsiteweb.git
- [deleted]          updatelicense
! [remote rejected] master (refusing to delete the current branch: refs/heads/master)
error: failed to push some refs to 'https://github.com/SaraMarhoum/Monsiteweb.git'
```

Vérifiez vos Repo (local et distant)

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch --list
* master
```

Default branch		
master	Updated 2 days ago by SaraMarhoum	Default
		Change default branch

Pourriez-vous expliquer l'objectif de cette phase ?

??KLpok

#6: sixth step: Rebasing:

Sur GITHUB, Editez le fichier README.md, ajoutez la ligne 'updates' ! n'oubliez pas le commit

```

<> index.html X
<> index.html > html.no-js > body > h1
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
10         <link rel="stylesheet" href="css/normalize.min.css">
11         <link rel="stylesheet" href="css/main.css">
12
13     </head>
14     <body>
15
16     <h1>This is a random title</h1>
17         <p>Hello world! This is HTML5 Boilerplate.</p>
18
19         <script src="js/main.js"></script>
20     </body>
21 </html>
22

```

Faites le staging et le commit en une seule ligne

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)

\$ git commit -am 'update index.html'

Maintenant apportez les changements faites sur le Repo distant : fetch

```

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git fetch origin master
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/SaraMarhoum/Monsiteweb
* branch          master       -> FETCH_HEAD
   3bb76d8..7992851 master     -> origin/master

```

Git status ? lisez les commentaires

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/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
```

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git pull --rebase
First, rewinding head to replay your work on top of it...
Applying: update index.html
```

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

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git historique
0b42ca7 (HEAD -> master) update index.html
```

Vérifiez que tous les changements que vous avez faits sont OK ?

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

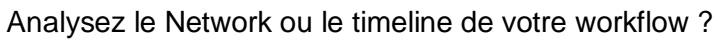
nothing to commit, working tree clean
```

Expliquez l'utilité du REBASE ?

#7: seventh step GitHub Insights:

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

Period: 1 week ▾



.....

#8 : eighth step : Default branch and conflicts

Sur github créez une branche du nom "Dev"

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

Default branch

The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

Dev ▼ Update

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

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

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 ?

;.....

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

Sur GIT, supprimez récursivement et localement votre repo website

```
Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ ls
css/  error404.html  img/  index.html  js/  LICENSE  Monsiteweb/  README.md

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ rm -rf Monsiteweb/

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ ls
css/  error404.html  img/  index.html  js/  LICENSE  README.md
```

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

```

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git clone https://github.com/SaraMarhoum/Monsiteweb.git
Cloning into 'Monsiteweb'...
remote: Enumerating objects: 52, done.
remote: Counting objects: 100% (52/52), done.
remote: Compressing objects: 100% (33/33), done.
remote: Total 52 (delta 19), reused 41 (delta 14), pack-reused 0
Unpacking objects: 100% (52/52), done.

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ ls
css/  error404.html  img/  index.html  js/  LICENSE  Monsiteweb/  README.md

```

Vérifiez si le contenu est récupéré

Le contenu supprimé à bien été récupéré

Vérifiez les branches et ou pointe le pointeur : HEAD

```

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (Dev)
$ git branch -a
* Dev
master
remotes/origin/Dev
remotes/origin/HEAD -> origin/master
remotes/origin/ajustement
remotes/origin/annulation
remotes/origin/master

```

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

```

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (Dev)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch -a

```

Toutefois la branche par défaut est :

```

Youcode@DESKTOP-TLLL2MM MINGW64 ~/Desktop/projects/Monsiteweb-local (master)
$ git branch -a
Dev
* master
remotes/origin/Dev
remotes/origin/HEAD -> origin/master
remotes/origin/ajustement
remotes/origin/annulation
remotes/origin/master

```


Conclusion :

Ce brief projet n'a pas été aussi difficile du fait que ce dernier a pour objectif la familiarisation avec les commandes git qui est un outil incontournable pour tout développeur.

S'il y'a un point qu'il fallait réussir tout de même c'est le travail en groupe et le respect de délai de livraison du projet.

Grace a trello on a pu geré ce projet et deviser les taches .

Nous avons durant la réalisation de ce projet appliqué directement les connaissances acquises pendant ce mois de formation a youcode. Nous avons fait beaucoup de recherche pour atteindre nos objectifs.

Ce projet a été vraiment passionnant et je suis fière de cette realisation.