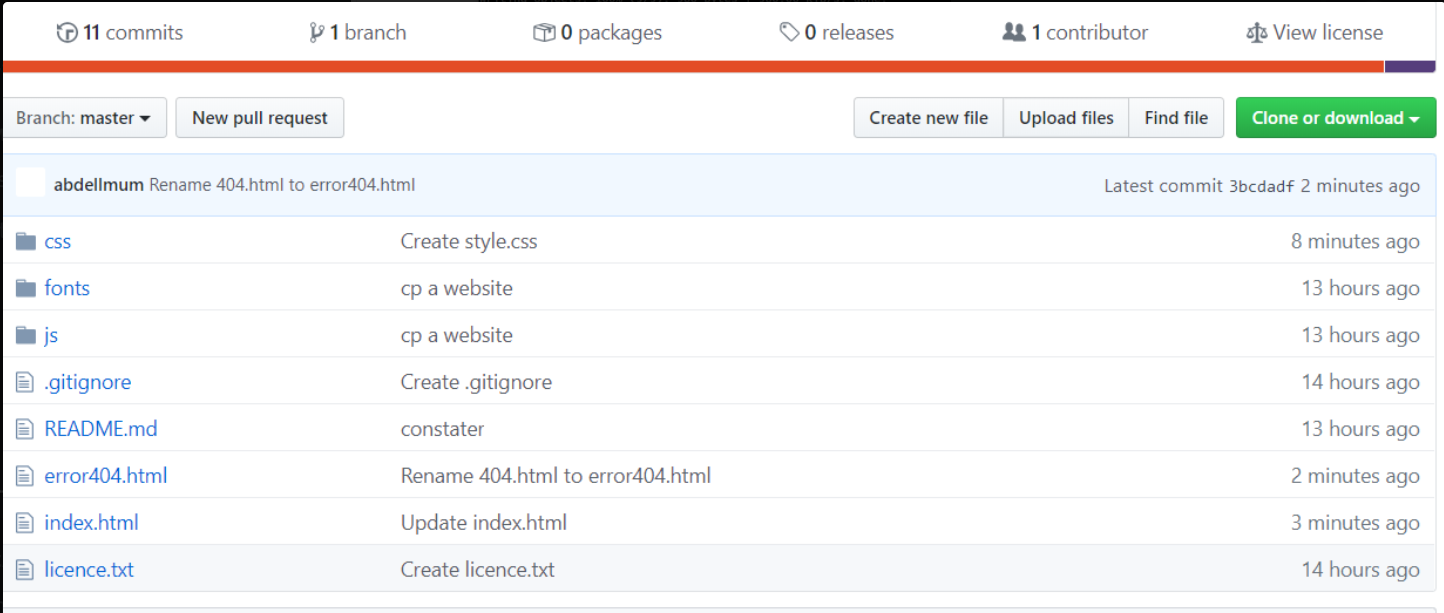
#1: First Step: Changes on Github

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

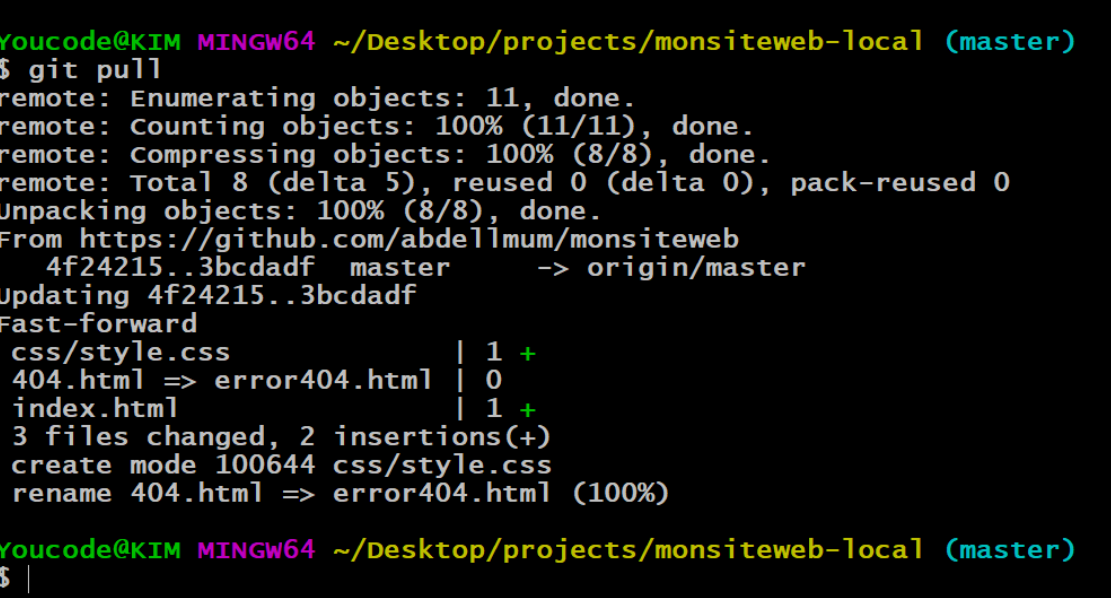
2. Sur index.html ajoutez juste après la balise <body> une balise <H1> : <h1> modification récente </h1>

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

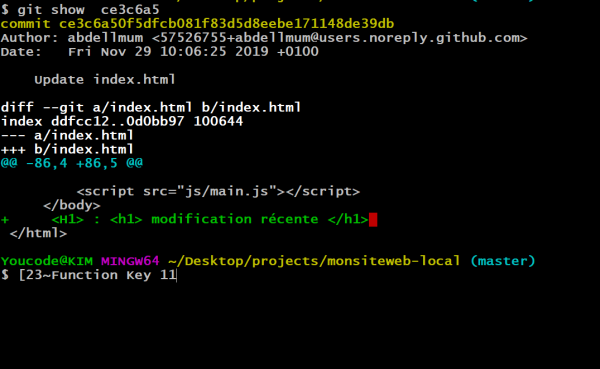


Maintenant faites le pull et fusionnez les changements distants avec le repo local

Visualisez et télécharger les fichiers distants sur GITBASH 4. Maintenant faites le pull et fusionnez les changements distants avec le repo local



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



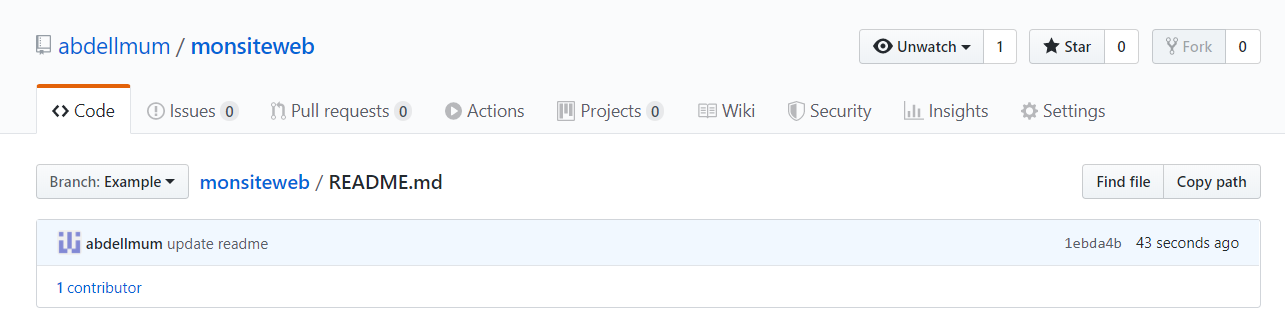
#2: Second Step: Branching and merging sur GITHUB

1. Repo distant :

⇒ Sur GitHUB créez une branche du nom ‘Example’

⇒ Sur la branche ‘Example’, modifiez le fichier READM.md et faites-le commit (toujours sur GITHUB)

⇒ Sur github et sur le menu ‘branch’ ; Expliquez ce que vous voyez comme message ?

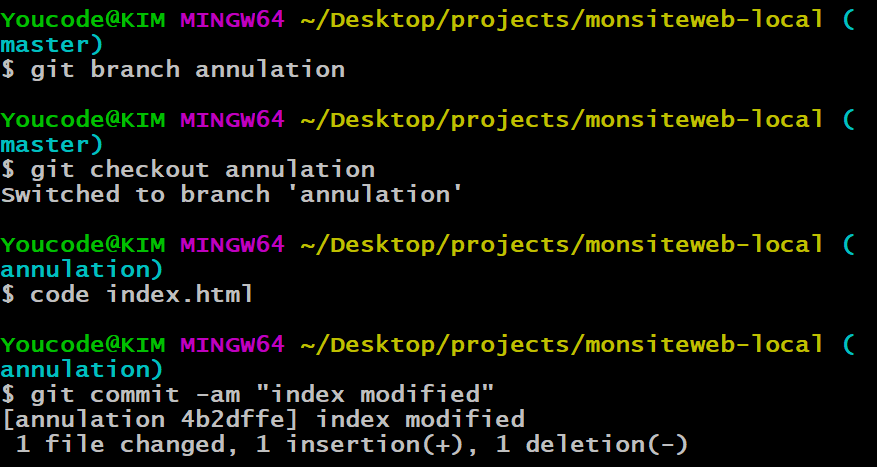


2. Repo local :

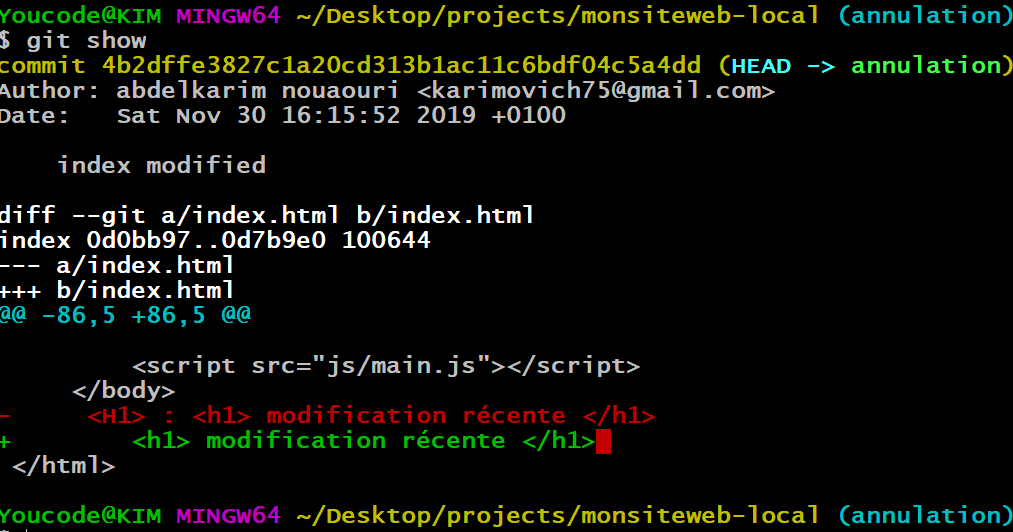
⇒ Sur votre repo local, créez une branche nommé ‘annulation’.

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

⇒ Faites-le commit et le stagging en une seule ligne.



« Git show » pour vérifier vos changements

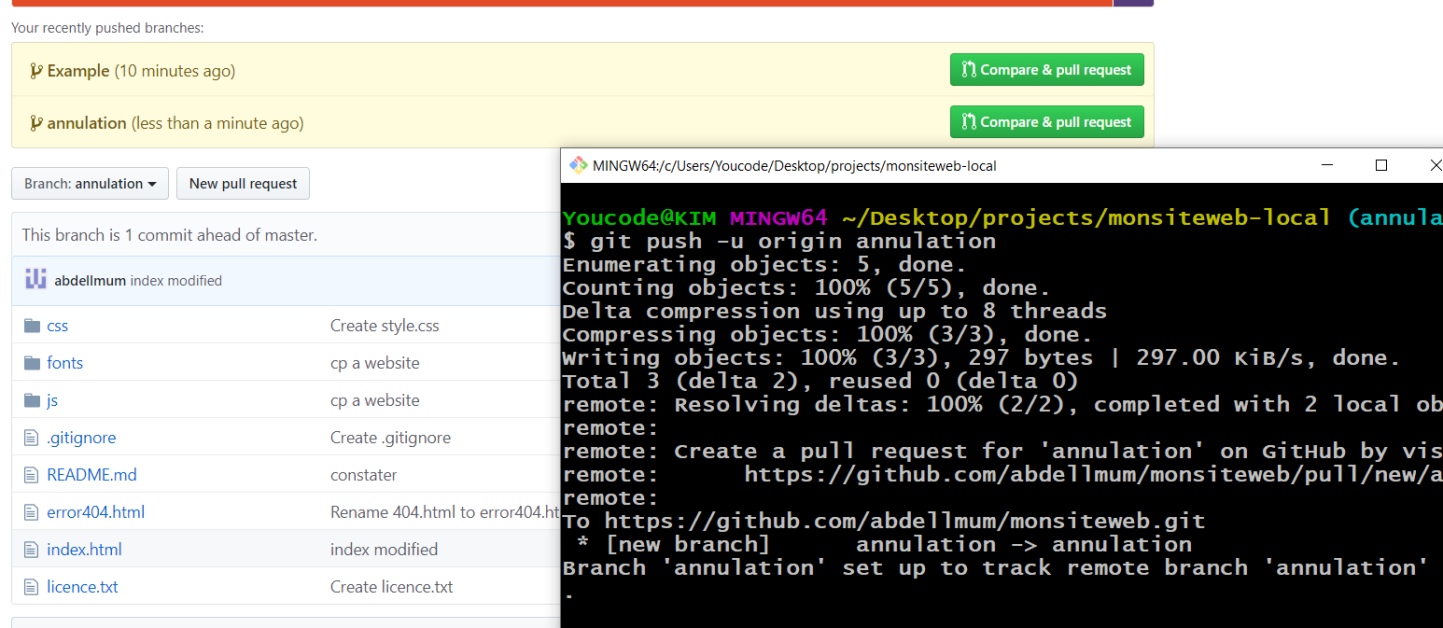


3. Vérification :

⇒ Sur GITBASH Pushez les changements que vous avez sur la branche ‘annulation’ à votre repo distant

⇒ Maintenant vérifiez votre repo distant.

⇒ Qu’est-ce que vous constatez?

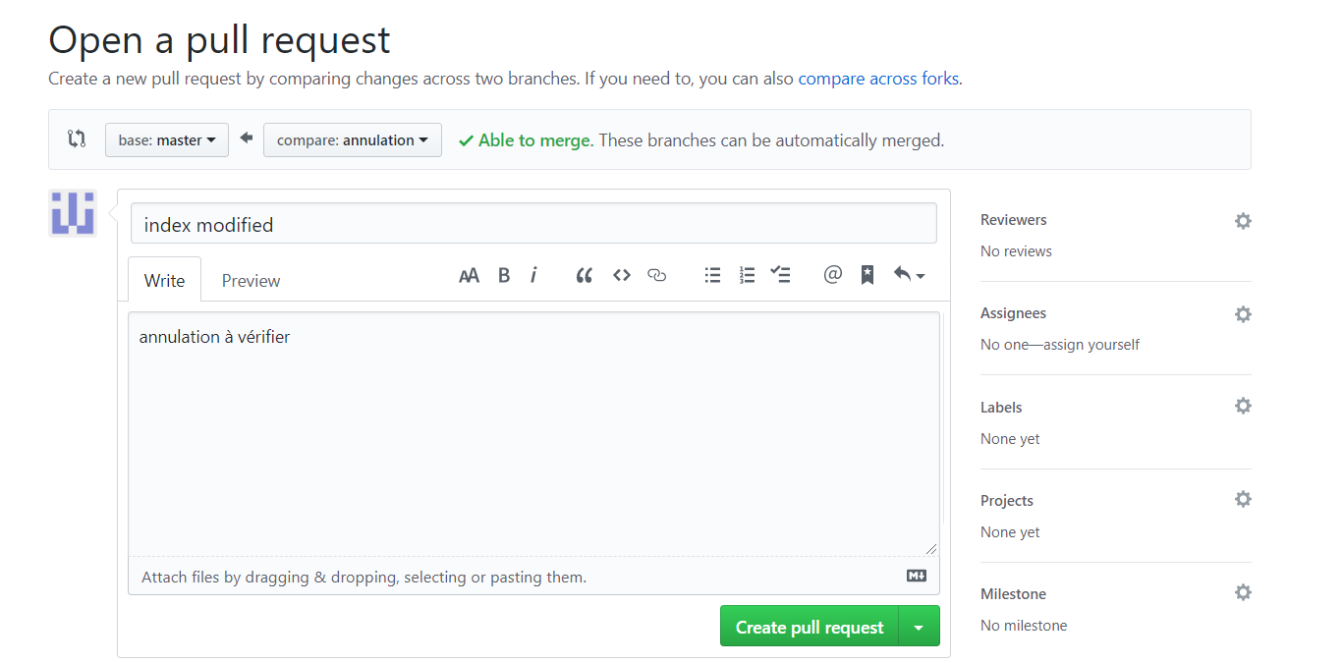


#3: Third Step: compare pull Requests

⇒ Sur votre repo Distant, un message indiquant qu’une branche a été ajoutez avec un bouton « compare and pull request »

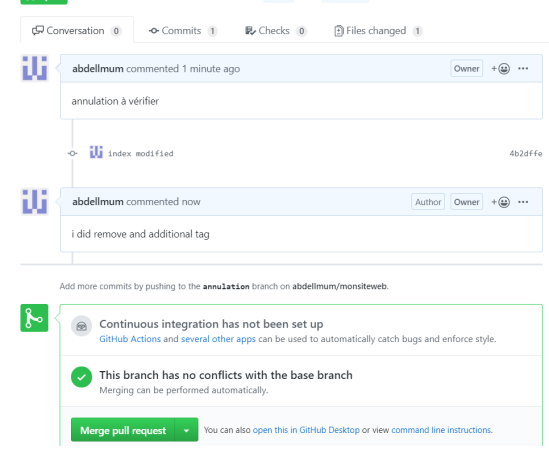
⇒ Cliquez sur le bouton ? analysez le rendu de la page ? qu’est-ce que vous constatez ?

⇒ Créez un pull request avec le commentaire ‘annulation à vérifier’



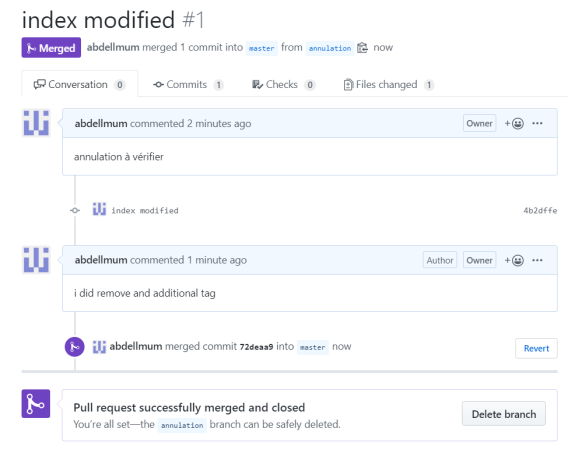
⇒ Une page s’ouvre, explorez la page ? qu’est-ce que vous constatez ?

⇒ Sur l’onglet ‘conversation’ vous pouvez aussi laisser des commentaires à la personne qui demande pull request, laisser un commentaire.



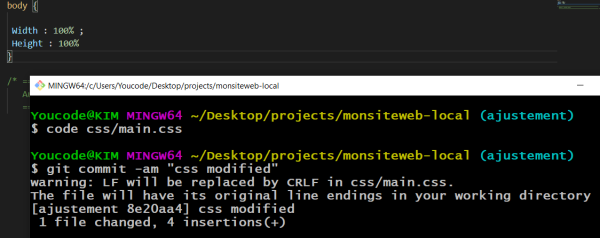
⇒ Maintenant si tout va bien ; aucun message d’erreur effectuez le merging ?

⇒ Supprimez la branche ‘annulation’ une fois tout est OK

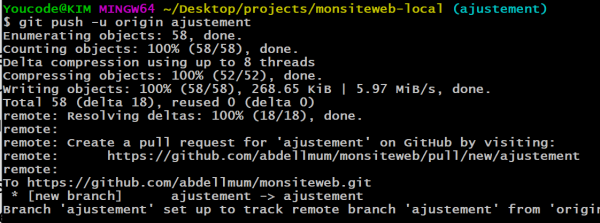


#4: fourth Step: merging en local

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 :



Faites maintenant le nécessaire pour ‘pushez’ les modifications sur GITHUB



⇒ Sur votre GITHUB, un pull request est demandé !

⇒ 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’

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 ?

4. Maintenant demandez le pull ?

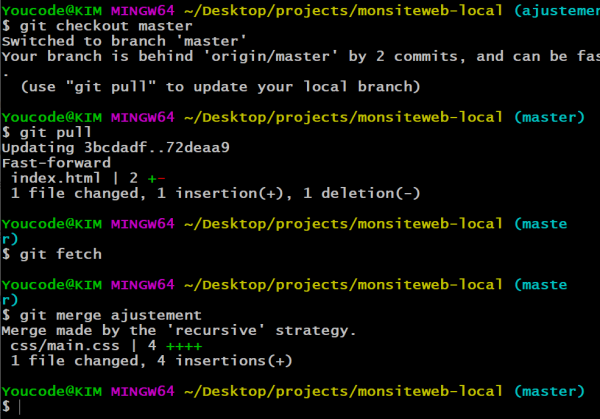
5. Une fois la commande du pull est exécutée, à votre avis quel type de merge on aura besoin

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

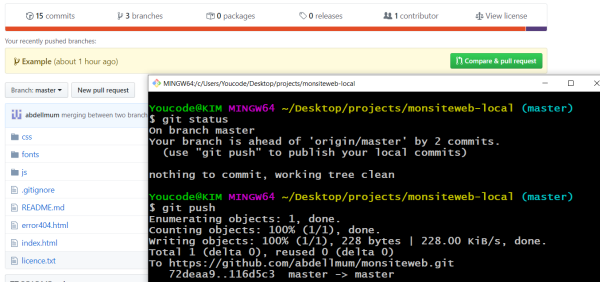
délivré ? à quoi sert cette fenêtre ?



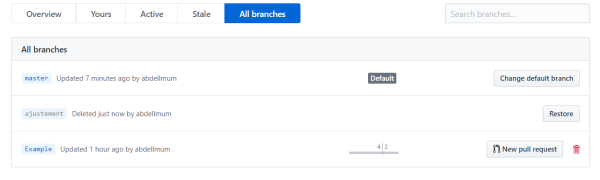
8. Git status

9. Si tout est OK ! Pushez les changements ?

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



Sur GITHUB supprimez les branches que vous avez créé.

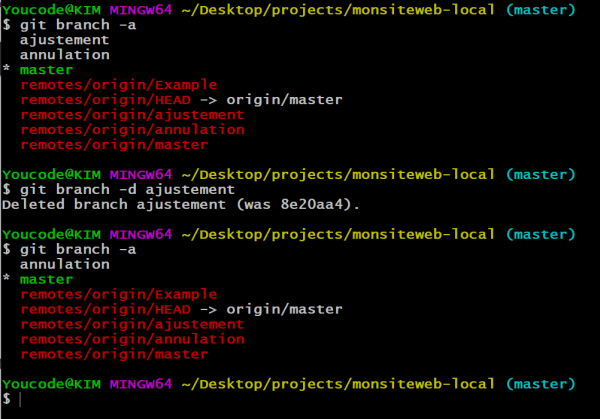


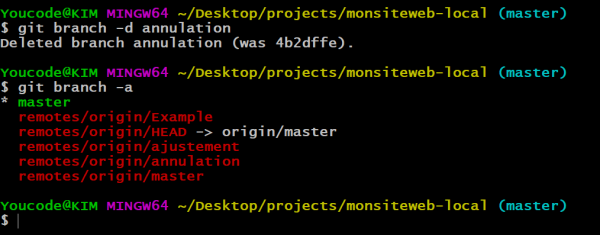
Sur le local on doit apporter les changements qu’on a fait sur GITHUB :

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

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

3. Maintenant ! Tapez la commande git bash –a ? qu’est-ce que vous constatez ?





#5: fifth Step: The Cleaning up

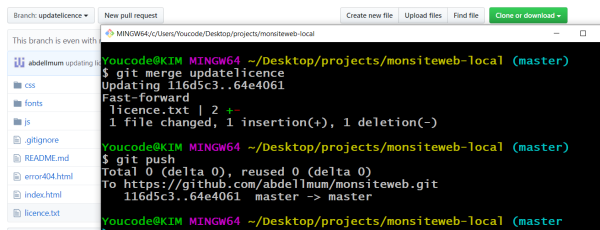
Dans cette phase je vous laisse découvrir l’objectif ?

⇒ Sur GITHUB Créez une branche nommée ‘updatelicence, éditez le fichier Licence et changer Apache 2.0 par Apache 3.0

⇒ Sur GITBASH faites un pull global : git pull –-all ;

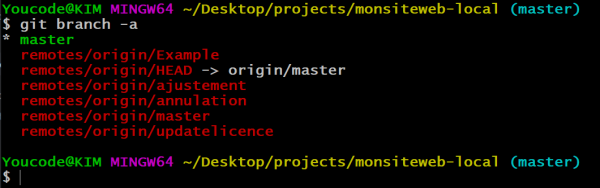
⇒ Maintenant faites le merge de la branche ‘updatelicence’.

⇒ Executez Git push



Git branch –d ‘updatelicence’

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

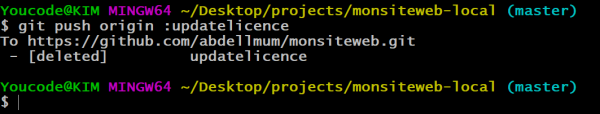


Challenge:

Maintenant sur GITBASH, trouvez la commande qui va nous permettre de supprimer la branche depuis le remote et ainsi de pushez les changements de la suppression simultanément ?

Vérifiez vos Repo (local et distant)

Pourriez-vous expliquez l’objectif de cette phase ?



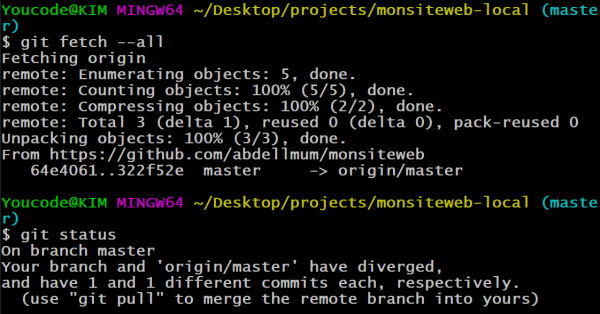
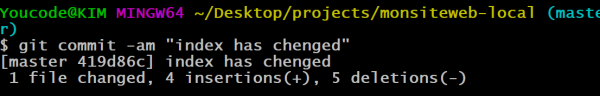
#6: sixth step: Rebasing

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

⇒ Sur GITBASH Editez le fichier index.html supprimez les lignes Et modifiez le titre du site : choisissez le titre que vous vouliez Faites le stagging et le commit en une seule ligne

⇒ Maintenant apportez les changements faites sur le Repo distant : fetch

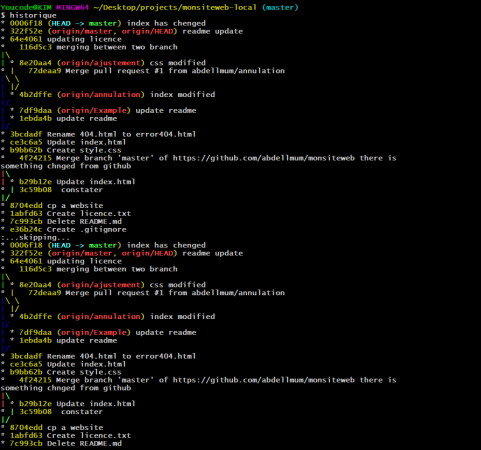
⇒ Git status ? lisez les commentaires



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

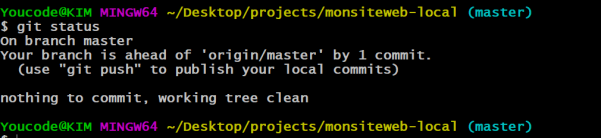
⇒ Exécutez l’alias que vous avez créé dans le scénario #1 (historique) ? qu’est-ce que vous

⇒ constatez sur l’arborescence des branches ?



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

⇒ Expliquez l’utilité du REBASE ?

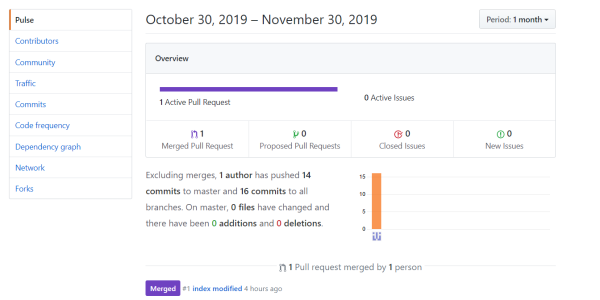


#7: seventh step GitHub Insights:

Sur Github vous avez un menu INSIGHTS ; Explorez votre activité à travers insights ⇒ Analysez le Network ou le timeline de votre workflow ?

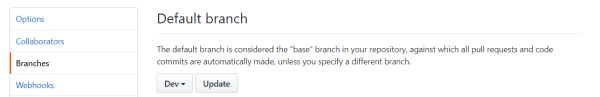
⇒ Discutez entre vous équipe et formalisez ce que vous voyez sur le Network en une production

⇒ Timeline sur Tableau à marqueur ou trello : imaginez aussi que chaque membre du groupe à contribuer dans une tache du Timeline Network ( Like a Sprint planning)



#8 : eight step : Default branch and conflicts

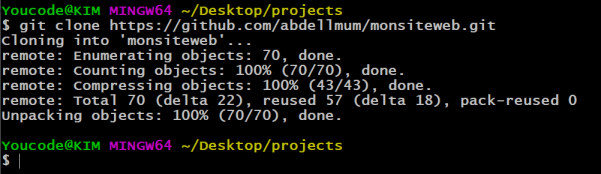
⇒ Sur github créez une branche du nom “Dev”

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

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

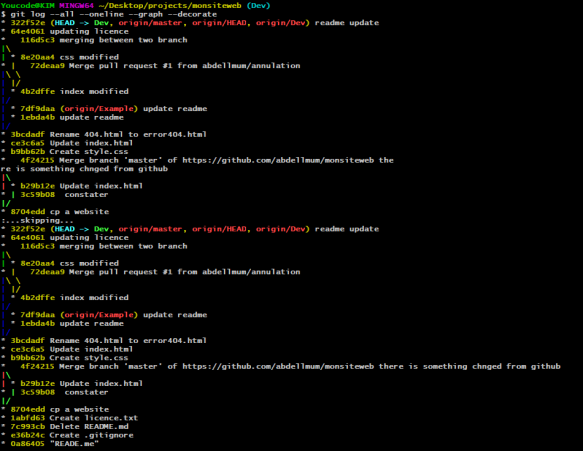
⇒ Sur GIT, supprimez récursivement et localement votre repo website

⇒ Maintenant Clonez via https ou SSH si vous l’avez réussi



⇒ Vérifiez si le contenu est récupéré

⇒ Vérifiez les branches et ou pointe le pointeur : HEAD



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

⇒ Toutefois la branche par défaut est

