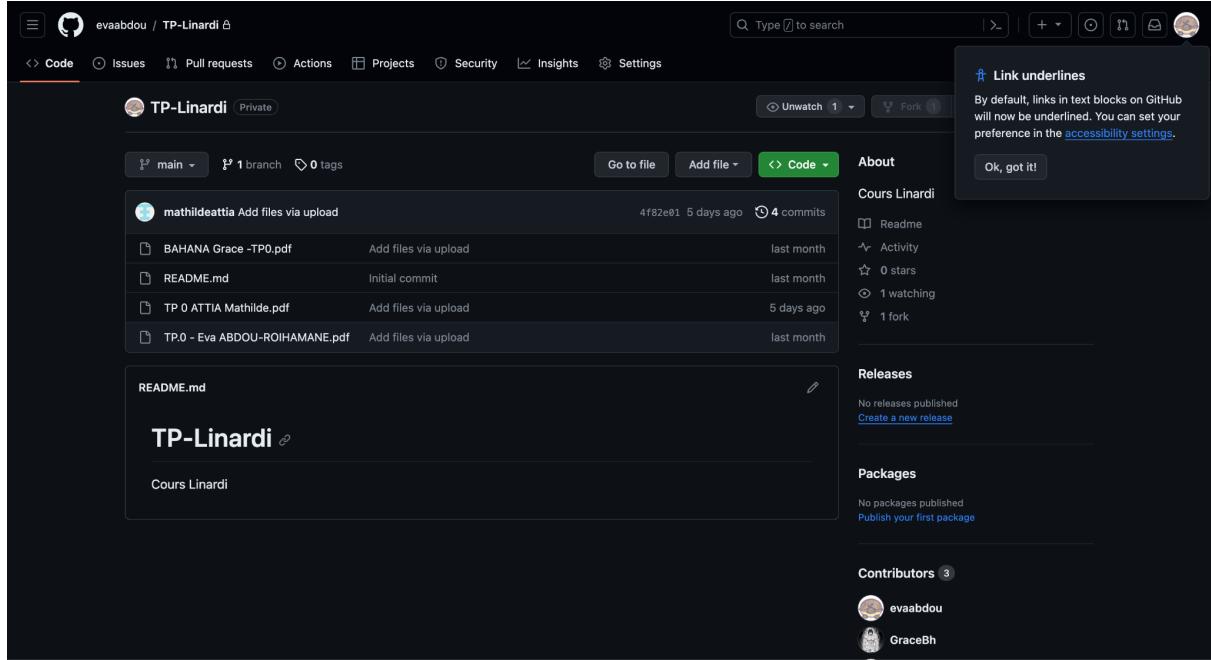


Compte rendu

TP 1

Dans le cadre de ce compte rendu, je vais vous expliquer comment j'ai créé un commit sur GitHub.



Étape 1 : Accès au répertoire local

1.1. Tout d'abord, j'ai ouvert mon terminal, qui est le lieu où je peux interagir avec Git. Sur mon Mac, j'ai utilisé le Terminal.

1.2. Ensuite, j'ai navigué vers le répertoire local de mon projet. Dans mon cas, le projet est dans un dossier nommé "mon-projet" sur mon bureau. Pour y accéder, j'ai saisi la commande suivante : `cd ~/Desktop/mon-projet`.

Étape 2 : Vérification de l'état du dépôt

2.1. Pour m'assurer que je travaille dans un état propre, j'ai utilisé la commande `git status`. Cela m'a permis de voir la liste des fichiers modifiés, non suivis et des informations sur la branche actuelle.

Étape 3 : Ajout des fichiers au suivi de Git

3.1. Pour ajouter les fichiers que je voulais inclure dans le commit, j'ai utilisé la commande `git add`. Par exemple, pour ajouter un fichier nommé "mon-fichier.txt", j'ai saisi : `git add mon-fichier.txt`.

3.2. Si je voulais ajouter tous les fichiers modifiés en une seule fois, j'aurais utilisé `git add .`.

Étape 4 : Création d'un commit

4.1. Pour créer un commit, j'ai utilisé la commande `git commit -m "Message descriptif"`. J'ai remplacé "Message descriptif" par un message qui décrit les modifications que j'ai apportées dans ce commit. Par exemple, `git commit -m "Ajout de nouvelles fonctionnalités"`.

Étape 5 : Envoi du commit sur GitHub

5.1. Pour envoyer mon commit vers mon dépôt GitHub, j'ai utilisé la commande `git push origin nom_de_branche`. Dans mon cas, j'ai travaillé sur la branche principale, généralement appelée "main" ou "master", donc j'ai utilisé `git push origin main`.

5.2. Il est important de s'assurer que je suis connecté à mon compte GitHub et que j'ai les autorisations nécessaires pour pousser vers ce dépôt.

En suivant ces étapes, j'ai réussi à créer un commit sur GitHub. Les commits jouent un rôle fondamental dans la gestion de versions de mon code source, ce qui facilite le suivi des modifications, la collaboration avec d'autres développeurs et la maintenance efficace de mon projet.

TP2

Déploiement d'un Site HTML sur GitHub et Optimisation avec PageSpeed Insights

Dans ce compte rendu, je vais expliquer les étapes que j'ai suivies pour déployer un site HTML sur GitHub. Ensuite, j'expliquerai comment j'ai utilisé PageSpeed Insights pour optimiser le site en termes de performances et d'efficacité.

Étape 1 : Téléchargement et Création de la Repository GitHub

1.1. J'ai téléchargé la ressource "siteWeb_TP2.zip" qui contenait le squelette du site à partir du dossier "Supports de cours" sur TEAMS.

1.2. Ensuite, je me suis rendu sur GitHub (<https://github.com>) et me suis connecté à mon compte GitHub.

1.3. Dans mon tableau de bord GitHub, j'ai cliqué sur le bouton "New" pour créer un nouveau dépôt (repository).

1.4. J'ai donné un nom à mon dépôt, ajouté une description, choisi la visibilité. Ensuite, j'ai cliqué sur "Create repository" pour créer le dépôt.

(J'ai utilisé Sourcetree pour cloner mon dépôt GitHub sur mon ordinateur local. Pour ce faire, j'ai suivi ces étapes :

- a. J'ai ouvert Sourcetree et cliqué sur "Clone/New" pour créer un nouveau clone.
- b. J'ai saisi l'URL de mon dépôt GitHub dans le champ approprié.
- c. J'ai choisi un emplacement sur mon ordinateur où je voulais cloner le dépôt.
- d. J'ai cliqué sur "Clone" pour lancer le processus de clonage.)

Étape 2 : Ajout du Squelette du Site et Déploiement

Après avoir configuré GitHub Pages dans les paramètres de mon dépôt, voici comment j'ai pu accéder au site web que j'ai déployé :

Je me suis rendu sur la page d'accueil de mon dépôt GitHub en utilisant l'URL spécifique de mon dépôt.

Ensuite, dans la barre de navigation de mon dépôt, j'ai cherché l'onglet "Settings" (Paramètres).

J'ai fait défiler la page jusqu'à la section "GitHub Pages".

Dans cette section, j'ai trouvé l'option pour sélectionner la source de mon site. J'ai choisi la branche "main" (ou "master") comme source.

Après avoir fait ma sélection, j'ai cliqué sur le bouton "Save" pour enregistrer mes modifications.

Une fois les paramètres enregistrés, un message m'a informé que GitHub Pages était en cours de déploiement.

J'ai attendu quelques minutes, le temps que le site se déploie.

Lorsque le déploiement a été terminé, j'ai vu un message indiquant que mon site était publié. Il contenait également l'URL du site, généralement sous la forme <https://mon-nom-utilisateur.github.io/mon-depot>.

J'ai copié cette URL et je l'ai collée dans la barre d'adresse de mon navigateur web.

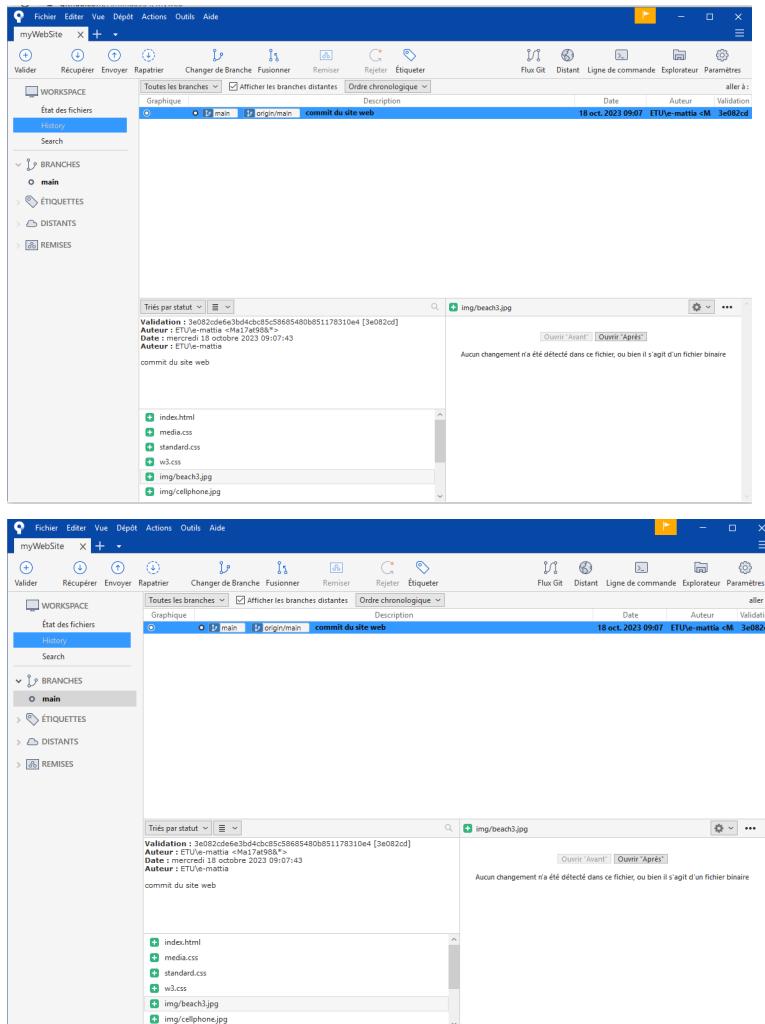
J'ai appuyé sur "Entrée" pour accéder à mon site web déployé, que je peux désormais partager avec d'autres utilisateurs.

C'est une partie importante du processus de déploiement, car elle montre comment accéder au site web une fois qu'il a été hébergé sur GitHub Pages. J'ai inclus ces étapes dans mon compte rendu pour une description complète de la procédure.

2.1. Après avoir créé la repository, j'ai suivi les instructions fournies par GitHub pour ajouter le squelette du site au dépôt. Généralement, cela implique d'ajouter les fichiers du site

(HTML, CSS, JavaScript, etc.) dans le dépôt en utilisant l'interface GitHub ou en utilisant des commandes Git via Git Bash ou le Terminal.

2.2. Une fois que j'ai ajouté les fichiers au dépôt, j'ai utilisé l'outil de déploiement intégré de GitHub, qui peut être configuré dans les paramètres du dépôt. J'ai choisi GitHub Pages comme option de déploiement, qui permet d'héberger le site directement depuis le dépôt.



GitHub Pages

Access

Collaborators

Moderation options

Code and automation

- Branches**
- Tags**
- Rules**
- Actions**
- Webhooks**
- Environments**
- Codespaces**
- Pages**

Build and deployment

Source

Deploy from a branch

Branch

Your GitHub Pages site is currently being built from the **main** branch. [Learn more about configuring the publishing source for your site.](#)

main / (root) **Save**

Learn how to [add a Jekyll theme](#) to your site.

Security

Code security and analysis

Deploy keys

Secrets and variables

Custom domain

Custom domain

Custom domains allow you to serve your site from a domain other than [aminaa954.github.io](#). [Learn more about configuring custom domains.](#)

Email notifications

Integrations

GitHub Apps

Email notifications

SSL

HTTP

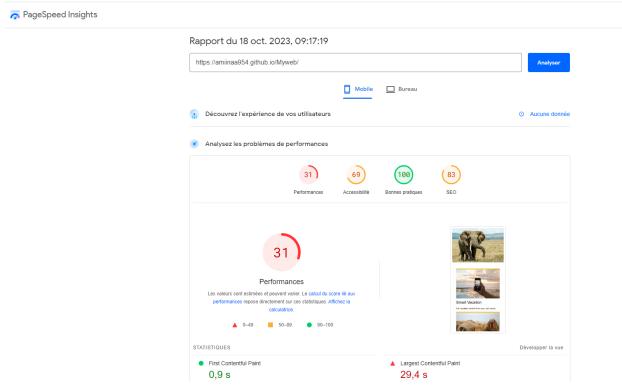
The screenshot shows a Microsoft Edge browser window with the address bar set to aminaa954.github.io/Myweb/. The main content area displays a high-resolution photograph of an African elephant standing in a dry, grassy savanna with hills in the background. A small Google Translate overlay is visible in the top right corner of the image. The browser's taskbar at the bottom shows several other open tabs, including 'Visseuse de l...', 'Sourceset Free Gi...', 'Pages', 'W3CSS Template', 'Export en développement', 'CAS-Central Auth...', 'PageSpeed Insights', 'Document partagé...', and 'Splendid18102023...'. The system tray at the bottom right indicates the date as 18/10/2023 and the time as 09:08.

Étape 3 : Optimisation du Site avec PageSpeed Insights

3.1. Pour optimiser le site, j'ai utilisé PageSpeed Insights de Google. J'ai accédé à l'outil en ouvrant un navigateur web et en recherchant "PageSpeed Insights" dans un moteur de recherche.

3.2. J'ai saisi l'URL du site que j'ai déployé sur GitHub dans l'outil PageSpeed Insights et j'ai attendu que l'analyse se termine.

3.3. L'outil PageSpeed Insights m'a fourni des suggestions et des rapports sur les aspects suivants :



Performance globale du site : J'ai noté les scores attribués par l'outil et les domaines nécessitant une amélioration.

Réduction des requêtes HTTP : J'ai suivi les recommandations de l'outil pour réduire le nombre de requêtes HTTP en combinant des fichiers CSS et JavaScript, en réduisant la taille des images, etc.

Nettoyage du code : J'ai examiné mon code HTML et CSS pour le rendre plus propre, en supprimant les éléments inutiles et en optimisant les sélecteurs CSS.

En suivant les étapes ci-dessus, j'ai réussi à déployer un site HTML sur GitHub et à l'optimiser avec PageSpeed Insights. L'optimisation frontend a permis d'améliorer la vitesse de chargement du site, offrant ainsi une meilleure expérience utilisateur. Ce compte rendu résume le travail effectué lors de ce TP.