***Commande GIT :***

1. **git --version :**

* montre version de git installer dans votre pc.

1. **git config --global user.name "zakaria saidi":**

* **--global** : signifie que cette configuration sera appliquée à tous les dépôts Git de ton ordinateur.
* **user.name** : est la clé qui indique le nom d'utilisateur.
* **zakaria saidi** : est la valeur de cette clé, donc le nom qui sera affiché pour tous tes commits.

1. **git config --global user.email "saidizakaria121@gmail.com":**

* **user.email** :est la clé de configuration pour définir l’adresse e-mail de l’utilisateur.
* **saidizakaria121@gmail.com** : est l’adresse e-mail que tu associes à tes commits.

1. **git init :**

* est utilisée pour initialiser un nouveau dépôt Git dans un projet. En l'exécutant, Git crée un dossier caché nommé .git . Git sait maintenant qu'il doit surveiller le dossier sur lequel vous l'avez lancé.Git crée un dossier caché pour suivre les modifications

1. **git status :**

* est utilisée pour afficher l'état actuel de votre dépôt Git.

1. **git add <nom\_du\_fichier>** :

* Cette commande ajoute un fichier spécifique à l'index (ou "staging area") pour le préparer à être validé (committed).
* Si vous voulez ajouter tous les fichiers modifiés, vous pouvez utiliser git add . ou git add --all.

1. **git commit -m "Message de commit"** :

* **commit** enregistre les modifications dans l'historique du projet.
* **-m** permet d'ajouter un message de commit. Ce message doit être clair et décrire les modifications effectuées pour ce commit

:

1. **git log** :

* Affiche l'historique des commits du dépôt avec des détails comme l'ID du commit, l'auteur, la date et le message du commit.
* Utilisez **git log --oneline** pour un affichage simplifié de l'historique, avec un commit par ligne.

1. **git clone <url>** :

* **Clone** un dépôt Git distant dans un répertoire local. C'est utilisé pour copier un projet déjà existant depuis un dépôt distant (par exemple, GitHub).

1. **git remote add origin <url>** :

* **remote add** ajoute un dépôt distant au projet.
* **origin** est le nom donné au dépôt distant (conventionnellement, on utilise "origin" pour le dépôt principal).
* **<url>** est le lien vers le dépôt distant.

1. **git push origin <branche>** :

* Envoie les commits de la branche spécifiée au dépôt distant.
* **origin** est le dépôt distant, et **<branche>** est la branche que vous voulez envoyer, par exemple main ou master.

1. **git pull origin <branche>** :

* Récupère et fusionne les modifications de la branche spécifiée depuis le dépôt distant vers la branche actuelle.
* **pull** est souvent utilisé pour synchroniser votre dépôt local avec les derniers changements effectués sur le dépôt distant.

1. **git branch :**

* Affiche toutes les branches existantes dans le dépôt, avec un astérisque **\*** indiquant la branche active.
* Utilisez **git branch <nom\_branche>** pour créer une nouvelle branche sans changer la branche actuelle.

1. **git checkout <nom\_branche>** :

* Permet de basculer entre les branches existantes.
* Utilisez **git checkout -b <nom\_branche>** pour créer une nouvelle branche et y basculer directement.

1. **git merge <nom\_branche>** :
   * Fusionne la branche spécifiée avec la branche active.
   * Par exemple, si vous êtes sur la branche main et que vous exécutez git merge feature, Git fusionnera les changements de la branche feature dans main.
2. **git stash** :

* Enregistre temporairement les modifications non validées pour pouvoir revenir à une branche propre.
* Utilisez **git stash pop** pour récupérer les changements enregistrés.

1. **git reset <options> <commit>** :

* Annule les commits en fonction de l'option spécifiée.
* Options :
  + **--soft** conserve les modifications dans l'index (staging area).
  + **--mixed** conserve les modifications dans le répertoire de travail.
  + **--hard** annule les modifications définitivement, les supprimant du répertoire de travail.

1. **git revert <commit> :**

* Crée un nouveau commit qui annule les modifications du commit spécifié, sans modifier l'historique du projet.

1. **git diff** :

* Affiche les différences entre le répertoire de travail et l'index.
* Utilisez **git diff <branche1> <branche2>** pour voir les différences entre deux branches.