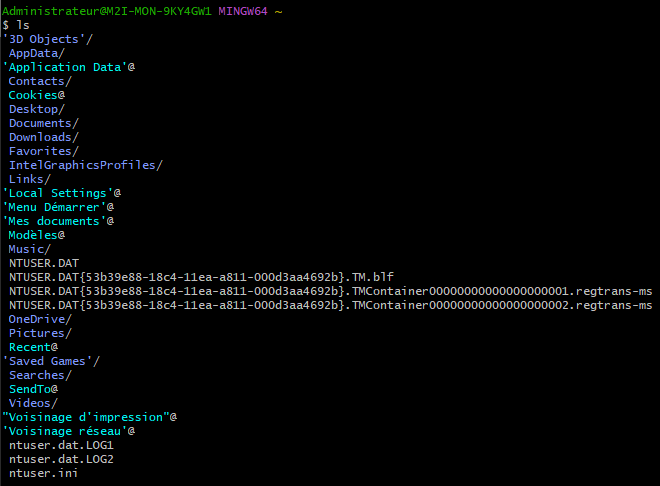
**Bases de la ligne de commande :**

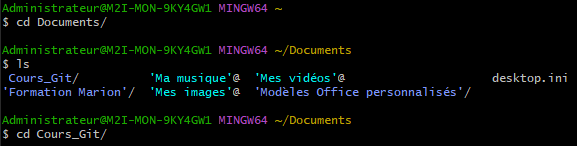
**ls** : « list » lister le contenu d’un dossier (autres dossiers, fichiers etc…)



**ls -l** : lister le contenu d’un dossier de façon plus détaillée (taille, date etc…)

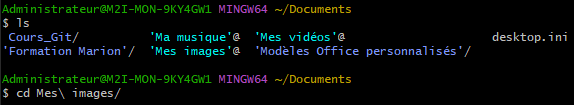
Devant les fichiers on retrouve différentes lettres parmi drwx : droits de lecture, écriture et exécution

**cd** + nom du dossier : « change directory - changer de dossier » naviguer en avançant vers le dossier



Ecrire le début du nom de dossier puis tab pour compléter le nom automatiquement

Si le nom de dossier est en plusieurs mots écrire le premier mot puis anti slash et le second mot etc..



**cd ..** : retour en arrière vers le dossier précédent

**cd ../..** : retour en arrière deux fois vers les dossiers précédents etc…

**cd ~** : retour au premier dossier principal

**clear** : effacer toutes les lignes de code écrites sur le terminal

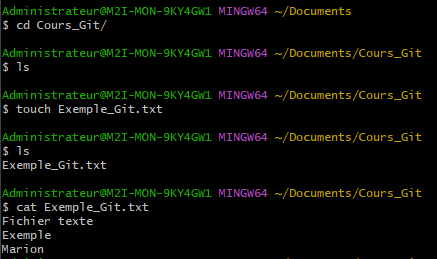
**pwd** : pour savoir exactement où je suis dans l’arbre de dossiers



**mkdir** + nom du dossier : « make directory – créer un dossier »

**touch** + nom du fichier en .txt ou .html : « créer un fichier texte vide »

**cat** + nom du fichier : visualiser le contenu d’un fichier texte



Dossier vide

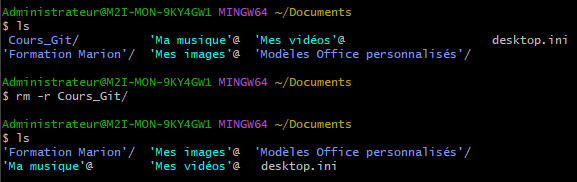
puis fichier visible

dans le dossier

après création

**rm** + nom du dossier : « remove – supprimer un dossier vide »

**rm -r** + nom du dossier : supprimer un dossier contenant des fichiers ou autres dossiers



Dossier visible

puis plus dans la liste

après suppression

**rm -d** + nom du fichier : supprimer un fichier

**Bases de git :**

Working directory :

Environnement de travail / Espace de travail

Dépôt local

Pour travailler sur un fichier encore à l’état de projet, pas encore un fichier « versionné »

Stage :

Environnement intermédiaire / Salle d’attente avant « versionnement »

Dépôt local

On ne passe pas directement de l’environnement de travail à la zone de dépôt finale du fichier

Cette zone permet de réaliser les dernières modifications

Repository :

Environnement final

Dépôt local

Cette zone contient le fichier avec les changements validés, le fichier « versionné »

GitHub :

Dépôt distant regroupant les différentes versions du fichier

**Git init** : créer un dépôt git local

**Git status** : permet de connaître le statut/l’état dans lequel se trouve les fichiers du dépôt local

« A quel endroit ils se trouvent ? » Working directory, Stage ou repository ?

Réponse « *untracked files* » : fichiers se trouvant dans le working directory (pas de suivi effectué par git)

Réponse *« change to be committed*» : fichiers se trouvent dans le stage (suivi effectué par git)

« *new file*» quand c’est la première fois que le fichier est dans stage

« *modified*» quand ce n’est pas la première fois, qu’il existe déjà des versions

**Git add .** : passer tous les fichiers du working directory au stage

**Git add + nom du fichier** : passer le fichier souhaité du working directory au stage

**Git rm --cached + nom du fichier** : repasser le fichier du stage au working directory

**Git commit -m« nom de la version »** : passer tous les fichiers du stage au repository

Plus tard quand on veut modifier un fichier pour en faire une nouvelle version

Modifier le fichier dans le working directory

Enregistrer avec Ctrl+S

Refaire les différentes manip : git add, git commit -m

**Git log** : voir l’historique des changements

**Git log --oneline** : voir l’historique des versions en une ligne

Si je suis bloqué dans le terminal (je ne peux plus écrire) appuyer sur la touche Q

**Git remote add origin + url du dépôt** : lier le dépôt local au dépôt distant

**Git remote set-url origin + url du dépôt distant**: changer l’url du dépôt distant

**Git push origin main** : envoyer les fichiers/modifications du repository (dépôt local) vers le GitHub (dépôt distant)

**Git checkout –b + nom de la branche** : créer une branche locale et se positionner dessus pour travailler isolé de la branche principale « master » pour ne pas créer de bug dessus

**Git branch** : lister les différentes branches existantes, celle qui est colorée est celle sur laquelle on travaille

**Git checkout + nom de la branche** : naviguer entre les branches pour travailler sur la bonne branche

**Git merge + nom de la branche** : combiner la branche à la branche master (en étant positionné sur le branche master)