



*Misons sur le capital Humain*



# UNITECH | BRH

*Université de Technologie d'Haïti / Banque de la République d'Haïti .*

---

## *Gestion des risques informatique*

*Préparer Par :*

- *Johnley-Roosevelt LORVIL*

*Le 28/03/2025*

## Partie 1

### Exercice 1 : Initialiser un dépôt Git local

**Objectif :** Créer un projet versionné localement.

**Énoncé :**

1. Crée un dossier mon-projet-git
2. Crée un fichier index.txt avec une phrase
3. Initialise Git et fais un premier commit

**L'image que voici est la résolution de l'exercice précédent :**

```
MINGW64:/c:/Users/johnley/mon-projet-git
bash: red: command not found

johnley@DESKTOP-UFG4M20 MINGW64 ~
$ mkdir mon-projet-git

johnley@DESKTOP-UFG4M20 MINGW64 ~
$ cd mon-projet-git

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git
$ echo "ceci est mon premier projet git" > index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git
$ cat index.txt
ceci est mon premier projet git

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git
$ git init
Initialized empty Git repository in C:/Users/johnley/mon-projet-git/.git/

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git add index.txt
warning: in the working copy of 'index.txt', LF will be replaced by CRLF the next time Git touches it

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git commit -m "first_one" -a
[master (root-commit) 8528236] first_one
1 file changed, 1 insertion(+)
create mode 100644 index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git status
On branch master
nothing to commit, working tree clean

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ |
```

## Exercice 2 : Travailler avec l'historique Git

**Objectif :** Modifier un fichier, suivre et annuler les changements.

**Énoncé :**

1. Modifie `index.txt`
2. Vérifie les changements
3. Annule les modifications

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ nano index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git status
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:   index.txt

no changes added to commit (use "git add" and/or "git commit -a")

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git restore index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git status
On branch master
nothing to commit, working tree clean

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ |
```

## Exercice 3 : Ajouter un fichier README.md

**Objectif :** Créer un README simple et le commiter.

**Énoncé :**

1. Crée un fichier `README.md` avec :  
md  
CopierModifier  
# Mon Projet Git  
Petit test de versionnage avec Git.
2. Ajoute et committe le fichier

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ echo "# Mon projet git">README.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ echo "Petit test de versionnage avec git">>README.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ cat readme.md
# Mon projet git
Petit test de versionnage avec git

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git add README.md
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git commit -m "second_one" -a
[master 81b70ee] second_one
1 file changed, 2 insertions(+)
create mode 100644 README.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git status
On branch master
nothing to commit, working tree clean

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ |
```

## Exercice 4 : Créer un dépôt GitHub et pousser le projet

**Objectif :** Mettre ton projet en ligne.

**Énoncé :**

1. Crée un nouveau repo sur GitHub sans README
2. Ajoute le remote et pousse ton projet

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git remote add origin git@github.com:johnleylorvil/work.git

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git push -u origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 528 bytes | 132.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To github.com:johnleylorvil/work.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ |
```

## Exercice 5 : Créer une nouvelle branche

**Objectif :** Expérimenter une fonctionnalité sans casser le code principal.

**Énoncé :**

1. Crée une branche `nouvelle-fonction`
2. Ajoute un fichier `fonction.py`
3. Merge cette branche dans `main`

**Les deux images que voici est la résolution de l'exercice précédent :**

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git branch
* master

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git branch nouvelle-fonction

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git branch
* master
  nouvelle-fonction

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (master)
$ git branch -M main

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git branch
* main
  nouvelle-fonction

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git checkout nouvelle-fonction
Switched to branch 'nouvelle-fonction'

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ echo "essaie">fonction.py

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ ls
README.md  fonction.py  index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ git add fonction.py
warning: in the working copy of 'fonction.py', LF will be replaced by CRLF the next time Git touches it

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ git status
On branch nouvelle-fonction
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   fonction.py
```

## *Missions sur le capital Humain*

```
new file:   fonction.py

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ git commit -m "troisieme"
[nouvelle-fonction 2fcb27e] troisieme
1 file changed, 1 insertion(+)
create mode 100644 fonction.py

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ git status
On branch nouvelle-fonction
nothing to commit, working tree clean

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/master'.

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git merge nouvelle-fonction
Updating 81b70ee..2fcb27e
Fast-forward
 fonction.py | 1 +
1 file changed, 1 insertion(+)
create mode 100644 fonction.py

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ ls
README.md  fonction.py  index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 333 bytes | 33.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'main' on GitHub by visiting:
remote:   https://github.com/johnleylorvil/work/pull/new/main
remote:
To github.com:johnleylorvil/work.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

## Exercice 6 : Gérer un conflit

**Objectif :** Apprendre à résoudre un conflit Git.

**Énoncé :**

1. Sur main, ajoute une ligne à index.txt
2. Sur une autre branche update-index, modifie la même ligne

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ nano index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git add index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git commit -m "main"
[main af9cad2] main
1 file changed, 2 deletions(-)

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git checkout update-index
Switched to branch 'update-index'

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (update-index)
$ nano index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (update-index)
$ git add index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (update-index)
$ git commit -m "update-index"
[update-index 21a1ae7] update-index
1 file changed, 1 insertion(+), 3 deletions(-)

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (update-index)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 3 commits.
(use "git push" to publish your local commits)

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git merge update-index
Auto-merging index.txt
CONFLICT (content): Merge conflict in index.txt
Automatic merge failed; fix conflicts and then commit the result.

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main|MERGING)
$ nano index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main|MERGING)
$ git add index.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main|MERGING)
$ |
```

## Exercice 7 : Améliorer le README.md

**Objectif :** Rédiger un README plus professionnel.

❖ **Énoncé :**

Ajoute les sections suivantes :

- Description
- Installation
- Utilisation
- Auteur

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ nano readme.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ cat readme.md
# Mon Projet Git

Petit test de versionnage avec Git.

## 📖 Description
Ce projet est un exercice simple pour apprendre à utiliser Git et GitHub.
Il couvre les bases des commandes Git telles que `add`, `commit`, `merge`, `branch` et `push`.

## 🛠 Installation
1. **Clone le dépôt :**
   ```bash
   git clone <URL_DU_REPOSITORY>

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git add readme.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git commit -m "readme_pro" -a
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it
[main 799d598] readme_pro
 2 files changed, 15 insertions(+), 3 deletions(-)

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ git push -u origin main
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 4 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (13/13), 1.41 KiB | 207.00 KiB/s, done.
Total 13 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To github.com:johnleylorvil/work.git
 2fcb27e..799d598  main -> main
branch 'main' set up to track 'origin/main'.

johnley@DESKTOP-UFG4M20 MINGW64 ~/mon-projet-git (main)
$ |
```



## Partie 2

### Travaux Pratiques GIT et GITHUB

#### Exercice 1 : Initialiser un dépôt Git

**Objectif :** Créer un dépôt local et ajouter un fichier

- Créer un dossier ex1\_init\_repo
- Ajouter un fichier README.md contenant une description

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~  
$ mkdir ex1_init_repo  
  
johnley@DESKTOP-UFG4M20 MINGW64 ~  
$ cd ex1_init_repo  
  
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo  
$ git init  
Initialized empty Git repository in C:/Users/johnley/ex1_init_repo/.git/  
  
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)  
$ nano readme.md  
  
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)  
$ cat readme.md  
# Exercice 1 - Initialiser un dépôt Git  
  
Ceci est un projet pour apprendre à initialiser un dépôt Git et ajouter un fichier.  
  
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)  
$ ls  
readme.md  
  
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)  
$ |
```

#### Exercice 2 : Suivre les modifications d'un fichier

**Objectif :** Observer l'état d'un fichier modifié

- ☐ Modifier plusieurs fois note.txt

Les deux images que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ nano note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ cat note.txt
Ceci est la première version de note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git add note.txt
warning: in the working copy of 'note.txt', LF will be replaced by CRLF the next time Git touches it

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   note.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    readme.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git commit -m "first_add" -a
[master (root-commit) 466b57a] first_add
 1 file changed, 1 insertion(+)
 create mode 100644 note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    readme.md

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

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ nano note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git status
On branch master
Changes not staged for commit:
```

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git status
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:   note.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    readme.md

no changes added to commit (use "git add" and/or "git commit -a")

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ |
```

### Exercice 3 : Consulter l'historique des changements

**Objectif :** Utiliser log, show, blame

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git log
commit dffe1cb9fe7d068f36b8b9eb555575c3ab3fd4de (HEAD -> master)
Author: johnley <109400230+johnleylorvil@users.noreply.github.com>
Date: Sat May 10 22:03:50 2025 -0400

    second_add

commit 466b57aab0cf26ff1f115b56ff69cb5acce692cb
Author: johnley <109400230+johnleylorvil@users.noreply.github.com>
Date: Sat May 10 21:57:30 2025 -0400

    first_add

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git show 466b57aab0cf26ff1f115b56ff69cb5acce692cb
commit 466b57aab0cf26ff1f115b56ff69cb5acce692cb
Author: johnley <109400230+johnleylorvil@users.noreply.github.com>
Date: Sat May 10 21:57:30 2025 -0400

    first_add

diff --git a/note.txt b/note.txt
new file mode 100644
index 0000000..0227b0e
--- /dev/null
+++ b/note.txt
@@ -0,0 +1 @@
+Ceci est la première version de note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git blame note.txt
^466b57a (johnley 2025-05-10 21:57:30 -0400 1) Ceci est la première version de note.txt
dffe1cb9 (johnley 2025-05-10 22:03:50 -0400 2) version 2

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ |
```

## Exercice 4 : Annuler des changements

Annuler avant le commit

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ nano note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ cat note.txt
Ceci est la première version de note.txt
version 2
ligne temporaire

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git status
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:   note.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        readme.md

no changes added to commit (use "git add" and/or "git commit -a")

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ git checkout -- note.txt

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ cat note.txt
Ceci est la première version de note.txt
version 2

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (master)
$ |
```

## Exercice 5 : Lier un dépôt local à GitHub

Push vers GitHub

L'image que voici est la résolution de l'exercice précédent :

```
johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ git remote -v

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ git remote add origin git@github.com:johnleylorvil/ex1_init_repo.git

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        readme.md

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

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ git add readme.md
warning: in the working copy of 'readme.md', LF will be replaced by CRLF the next time Git touches it

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ git commit -m "Readme" -a
[main ce69ca2] Readme
 1 file changed, 3 insertions(+)
 create mode 100644 readme.md

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ git push -u origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 852 bytes | 38.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To github.com:johnleylorvil/ex1_init_repo.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

johnley@DESKTOP-UFG4M20 MINGW64 ~/ex1_init_repo (main)
$ |
```