



**UNIVERSITÉ
DE TECHNOLOGIE D'HAÏTI**

53 et 57, Ave N, Turgeau, Port-au-Prince, Haïti

Email : info@unitech.edu.ht

Faculté des Sciences, de Génie et d'Architecture

Programme : Diplôme de 2^{ème} cycle en technologie de l'information

Cours : Gestion des Risques Informatiques

Objet : Devoir (Travaux Pratiques Git, GitHub)

Professeur :

Dr.-Ing. Austin Waffo Kouhoué, PhD

Présenté par :

Jonel LUBIN

Soumis le, 10 – 05– 2025

Table des matières

TP 4	1
Exercice 1 : Initialiser un dépôt Git local	1
Exercice 2 : Travailler avec l'historique Git	1
Exercice 3 : Ajouter un fichier README.md Objectif :	2
Exercice 4 : Créer un dépôt GitHub et pousser le projet Objectif :	3
Exercice 5 : Créer une nouvelle branche Objective :	5
Exercice 7 : Améliorer le README.md Objectif :	9
TP5	13
Exercice 1 : Initialiser un dépôt Git Objectif : Créer un dépôt local et ajouter un fichier	13
Exercice 2 : Suivre les modifications d'un fichier Objectif :	14
Exercice 3 : Consulter l'historique des changements	18
Exercice 4 : Annuler des changements	20

TP 4

Exercice 1 : Initialiser un dépôt Git local

Objectif : Créer un projet versionné localement

1. Création d'un dossier **mon-projet-git**

Après avoir lancé git : voici le script de cette question :

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ mkdir mon-projet-git
```

2. Création d'un fichier index.txt avec une phrase

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ touch index.txt

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

J'ouvre le fichier avec nano index.txt puis Ctrl +x , puis y suivi de Enter

```
GNU nano 8.3 index.txt
ceci est mon premier fichier versionné avec git|
```

3. Initialise Git et fais un premier commit

```
JLMsc@DESKTOP-KHTOMKE 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
```

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ git commit -m "Premier commit : ajout du fichier index.txt"
[master (root-commit) e9bbdb7] Premier commit : ajout du fichier index.txt
1 file changed, 1 insertion(+)
create mode 100644 index.txt
```

Exercice 2 : Travailler avec l'historique Git

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

1. Modifions l'index

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

```
MINGW64:/c/Users/JLMsc/mon-projet-git
GNU nano 8.3 index.txt
ceci est mon premier fichier versionné avec git
je modifie mon premier fichier|
```

2. Vérifions les changements

a) Voyons l'état du dépôt

```
$ 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")
```

b) Affichons les différences avec le dernier commit :

```
$ git diff
warning: in the working copy of 'index.txt', LF will be replaced by CRLF the next time Git touches it
diff --git a/index.txt b/index.txt
index 183b5fb..0ebb1c3 100644
--- a/index.txt
+++ b/index.txt
@@ -1,3 @@
- ceci est mon premier fichier versionné avec git
+
+ je modifie mon premier fichier
```

3. Annulons les modifications

```
$ git restore index.txt

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

Exercice 3 : Ajouter un fichier README.md Objectif :

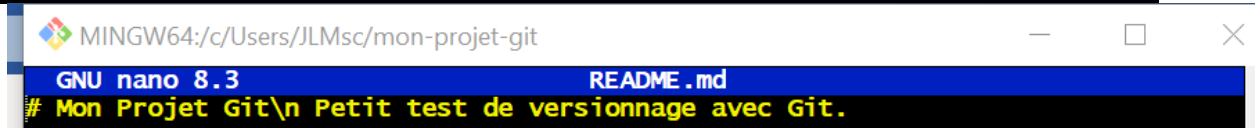
Créer un README simple et le commiter.

1. Crée un fichier README.md avec :

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ touch README.md

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ ls
README.md  index.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ nano README.md
```



2. Ajoute et committe le fichier

```
$ git add README.md
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ ls
README.md  index.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ git commit -m "Ajout du fichier README.md"
[master dd483a8] Ajout du fichier README.md
1 file changed, 1 insertion(+)
create mode 100644 README.md
```

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


Pour cette étape je lance <https://github.com/>, j'ai déjà un compte sur github et je crée un new repository, je laisse la description publique avec une description : « mon premier projet » puis je clique sur Create repository

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)


Required fields are marked with an asterisk ().*

Owner *

 JLabin-prog ▾

Repository name *

mon-projet-git


 mon-projet-git is available.

Great repository names are short and memorable. Need inspiration? How about [ideal-octo-chainsaw](#) ?

Description (optional)

mon premier projet

☒ Public

 Private

Anyone on the internet can see this repository. You choose who can commit.



2. Ajoute le remote et pousse ton projet

Je copie l'Url depuis le github puis colle dans git

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ git remote add origin https://github.com/JLabin-prog/mon-projet-git.git
```

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (master)
$ git branch -M main
```


```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git puush -u origin main
git: 'puush' is not a git command. See 'git --help'.
```

The most similar command is
push

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git push -u origin main
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), 591 bytes | 295.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/JLubin-prog/mon-projet-git.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

Voici le résultat sur github



The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with 'main' selected, '1 Branch', '0 Tags', a search bar, and a 'Code' button. Below this, the repository name 'mon premier projet' is displayed, along with '2 Commits'. The commit history table lists two commits: 'Ajout du fichier README.md' (28 minutes ago) and 'Premier commit : ajout du fichier index.txt' (53 minutes ago). The 'README' file is selected and its content is displayed: 'Mon Projet Git\nPetit test de versionnage avec Git.' On the right sidebar, there are links for 'Readme', 'Activity', '0 stars', '1 watching', and '0 forks'. At the bottom of the sidebar, there's a 'Releases' section indicating 'No releases published' with a link to 'Create a new release'.

Exercice 5 : Créer une nouvelle branche Objective :

Expérimenter une fonctionnalité sans casser le code principal. Énoncé :

1. Crée une branche nouvelle-fonction

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git branch nouvelle-fonction

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

Basculons sur cette branche avec la commande git Checkout nouvelle-fonction

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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ |
```

2. Ajout d'un fichier fonction.py

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ touch fonction.py

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ nano fonction.py
```

Il faut faire le transfert vers le dépôt distant

```
README.md  fonction.py  index.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (nouvelle-fonction)
$ git push -u origin nouvelle-fonction
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), 342 bytes | 342.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'nouvelle-fonction' on GitHub by visiting:
remote:   https://github.com/JLubin-prog/mon-projet-git/pull/new/nouvelle-fonction
remote:
To https://github.com/JLubin-prog/mon-projet-git.git
 * [new branch]      nouvelle-fonction -> nouvelle-fonction
branch 'nouvelle-fonction' set up to track 'origin/nouvelle-fonction'.
```

3. Merge cette branche dans main

on doit retourner dans la branche main

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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git merge nouvelle-fonction
Updating dd483a8..38fdf61
Fast-forward
 fonction.py | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 fonction.py

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$
```

Voici le résultat sur github.com

The screenshot shows the GitHub repository page for 'mon-projet-git'. The repository is public and has 2 branches and 0 tags. The main branch is selected. The repository description is 'mon premier projet'. The file list shows 'README.md', 'fonction.py', and 'index.txt'. The commit history shows a merge pull request from 'nouvelle-fonction' to 'main'.

Exercice 6 : Gérer un conflit Objectif :

Apprendre à résoudre un conflit Git. Énoncé :

- 1- Sur main, ajoute une ligne à index.txt

Comme on est déjà dans git je vais ajouter une ligne à index.txt

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

The screenshot shows the nano text editor window. The title bar indicates the file is 'index.txt' and it has been modified. The content of the file is 'ceci est mon premier fichier versionné avec git' followed by a new line 'Ligne ajoutée'.


```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git add index.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git commit -m "Ajour d'une ligne dasn index.txt sur main"
[main bc9e834] Ajour d'une ligne dasn index.txt sur main
1 file changed, 2 insertions(+)
```

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
\$ |

2- Sur une autre branche update-index, modifie la même ligne2
Je crée un conflit git en modifiant la même ligne sur une autre branche

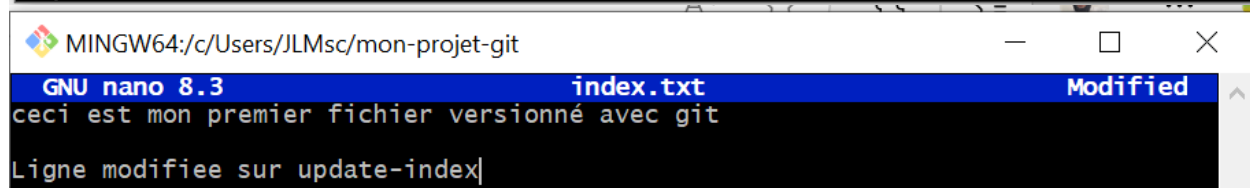
```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git checkout -b update-index
Switched to a new branch 'update-index'

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
$ |
```

Modifier la même ligne que sur main dans index.txt

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
$ nano index.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
$ |
```



MINGW64:/c/Users/JLMsc/mon-projet-git

GNU nano 8.3	index.txt	Modified
ceci est mon premier fichier versionné avec git		
Ligne modifiée sur update-index		

- Commit la modification

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
$ git add index.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
$ git commit -m "modification de la meme ligne dans index.txt sur update-index"
[update-index 834e1a3] modification de la meme ligne dans index.txt sur update-i
ndex
1 file changed, 1 insertion(+), 1 deletion(-)
```

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
\$ |

- Fusionner avec main pour provoquer un conflit

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (update-index)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

- Lancer la fusion :

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git merge update-index
Updating bc9e834..834e1a3
Fast-forward
 index.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

En résumé, git trouve un conflit dans index.txt, car la même ligne a été modifiée dans deux branches différentes

Pour vérifier le conflit on fait git status

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git satus
git: 'satus' is not a git command. See 'git --help'.

The most similar command is
  status
```



```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git commit -m "Ajout de la section Description dans ReadMe.md"
[main ffc5fce] Ajout de la section Description dans ReadMe.md
1 file changed, 3 insertions(+)

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

• Installation

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git clone https://github.com/JLubin-prog/mon-projet-git.git
Cloning into 'mon-projet-git'...
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 10 (delta 0), reused 9 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (10/10), done.

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ cd mon-projet-git

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git/mon-projet-git (main)
$ |
```

• Utilisation

- Nous allons Utiliser

1. git status

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 5 commits.
(use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
mon-projet-git/

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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/mon-projet-git (main)
$ |
```

2. git log

```
Ajout de la section Description dans ReadMe.md
commit 7af5955b9204e4613a4b85808df863a9083b0ad0
Author: JLubin <ing.jlubin@gmail.com>
Date:   Fri May 9 23:00:40 2025 -0400

Ajour de la section Description dans READ.md
commit 834e1a316029403bd26725e1341a6afbf7de17a1 (update-index)
Author: JLubin <ing.jlubin@gmail.com>
Date:   Fri May 9 22:17:44 2025 -0400

modification de la meme ligne dans index.txt sur update-index
commit bc9e834d1110f526791de65422ae1d9d50d1e213
Author: JLubin <ing.jlubin@gmail.com>
Date:   Fri May 9 22:06:52 2025 -0400

Ajour d'une ligne dasn index.txt sur main
commit 4a87f548ba63a640920ae73ae0960cf9cf46e8d2 (origin/main)
Merge: dd483a8 38fdf61
Author: JLubin <jonelubin10@gmail.com>
Date:   Fri May 9 21:56:23 2025 -0400

Merge pull request #1 from JLubin-prog/nouvelle-fonction

ajout du fichier fonction.py avec Bonjour
commit 38fdf61d613853dc6ce698b1a0d836a2f657fa3d (origin/nouvelle-fonction, nouvelle-fonction)
Author: JLubin <ing.jlubin@gmail.com>
Date:   Fri May 9 21:32:36 2025 -0400

ajout du fichier fonction.py avec Bonjour
commit dd483a85bddb3909255a301cb051206b840a31ab
Author: JLubin <ing.jlubin@gmail.com>
Date:   Fri May 9 20:49:54 2025 -0400

Ajout du fichier README.md
commit e9bbdb7ecc1506ce18d5df6c61fb192e99dc00a9
Author: JLubin <ing.jlubin@gmail.com>
Date:   Fri May 9 20:24:59 2025 -0400
```

3. git show

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ git show
commit 67a6e9a97959e11fe718172e6b8928f18d9efcd4 (HEAD -> master)
Author: JLibin <ing.jlibin@gmail.com>
Date: Sat May 10 20:25:39 2025 -0400

    Initial commit avec README.md

diff --git a/README.md b/README.md
deleted file mode 100644
index 721641a..0000000
--- a/README.md
+++ /dev/null
@@ -1,0,0 @@
-Mon dépôt Git

JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ |
```

- Auteur

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ nano README.md
```



The screenshot shows the nano text editor interface. The title bar indicates the file path is MINGW64:/c/Users/JLMsc and the file being edited is README.md. The editor content shows the file is being edited by 'Auteur' and contains the following text:

```
## Auteur
-Nom ; Jone1 LUBIN
-GitHub : JLibin-Prog
```

A la fin, j'effectue des commandes comme git add , git commit

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ git commit -m "Ajout des infos de l'auteur dans le README.md"
[master 2d30b0f] Ajout des infos de l'auteur dans le README.md
1 file changed, 3 insertions(+)
create mode 100644 README.md
```

TP5

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

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ mkdir ex1_init_repo

JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ |
```

Ensuite, je rentre dans le répertoire puis je l'initialise

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~ (master)
$ cd ex1_init_repo

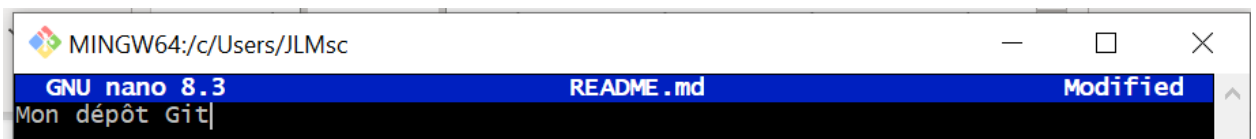
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git init
Initialized empty Git repository in C:/Users/JLMsc/ex1_init_repo/.git/
```

- Ajouter un fichier README.md contenant une description

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ touch README.md

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ nano README.md

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```



J'ajoute le fichier au suivi Git puis je valide les modifications avec un message de commit

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git commit -m "Initial commit avec README.md"
[master (root-commit) ff0d3f0] Initial commit avec README.md
1 file changed, 1 insertion(+)
create mode 100644 README.md

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

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

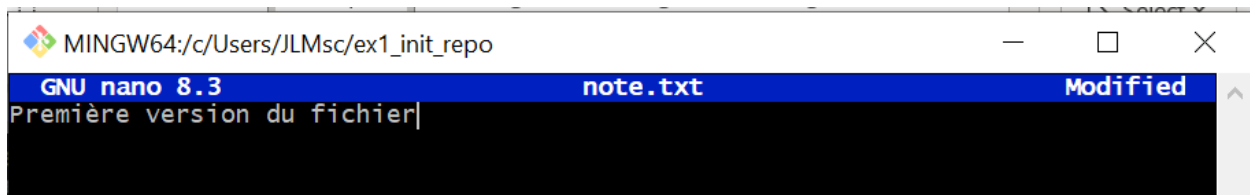
Observer l'état d'un fichier modifié

- Modifier plusieurs fois note.txt

1. Créer un fichier note.txt

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ touch note.txt

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



2. Ajouter et valider ce fichier dans Git

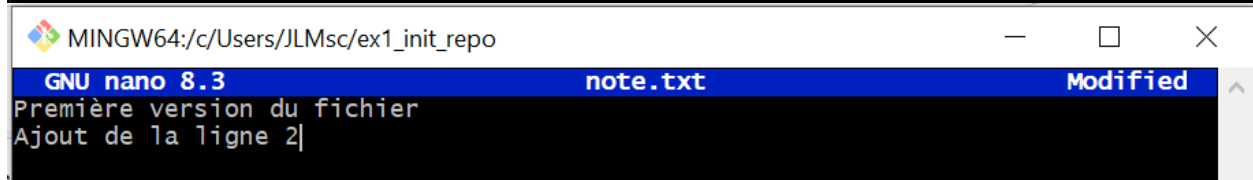
```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git add note.txt
warning: in the working copy of 'note.txt', LF will be replaced by CRLF the next time Git touches it

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git commit -m "Ajout initial de note.txt"
[master d6d2e3b] Ajout initial de note.txt
1 file changed, 1 insertion(+)
create mode 100644 note.txt

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```


3. Modifier le fichier une première fois

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



The screenshot shows a terminal window with the command prompt 'JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)' and the command '\$ nano note.txt'. Below the terminal, a window titled 'MINGW64:/c/Users/JLMsc/ex1_init_repo' displays the nano text editor. The editor's status bar indicates 'GNU nano 8.3', the filename 'note.txt', and the state 'Modified'. The editor's content shows two lines of text: 'Première version du fichier' and 'Ajout de la ligne 2|', where the cursor is at the end of the second line.

4. Vérifier l'état du fichier

```
JLMsc@DESKTOP-KHTOMKE 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

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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

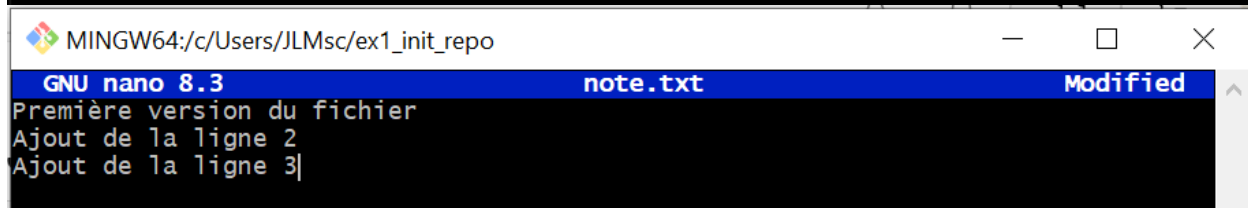
5. Voir les différences avec la version précédente

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git diff note.txt
warning: in the working copy of 'note.txt', LF will be replaced by CRLF the next
time Git touches it
diff --git a/note.txt b/note.txt
index 5ffb9b6..518bf25 100644
--- a/note.txt
+++ b/note.txt
@@ -1,1,2 @@
 Première version du fichier
+Ajout de la ligne 2

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

6. Ajouter une deuxième modification

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



MINGW64:/c/Users/JLMsc/ex1_init_repo

GNU nano 8.3	note.txt	Modified
Première version du fichier		
Ajout de la ligne 2		
Ajout de la ligne 3		

7. Observer à nouveau les changements

```
JLMsc@DESKTOP-KHTOMKE 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

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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git diff note.txt
warning: in the working copy of 'note.txt', LF will be replaced by CRLF the next
time Git touches it
diff --git a/note.txt b/note.txt
index 5ffb9b6..0e7db3a 100644
--- a/note.txt
+++ b/note.txt
@@ -1,3 @@
 Première version du fichier
+Ajout de la ligne 2
+Ajout de la ligne 3

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

8. Enregistrer toutes les modifications dans un nouveau commit

```
JLMsc@DESKTOP-KHTOMKE 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

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git commit -m "Ajouts de lignes dans note.txt"
[master 4af3bb0] Ajouts de lignes dans note.txt
1 file changed, 2 insertions(+)

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

Exercice 3 : Consulter l'historique des changements

Objectif : Utiliser log, show, blame

1. Consulter l'historique des commits avec `git log`

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git log
commit 4af3bb09057c6d0fe0b45f2549bee1a4706cb6cc (HEAD -> master)
Author: JLubin <ing.jlubin@gmail.com>
Date: Sat May 10 21:03:21 2025 -0400

    Ajouts de lignes dans note.txt

commit d6d2e3b01e811fd862f90d56760ebf7ca922696d
Author: JLubin <ing.jlubin@gmail.com>
Date: Sat May 10 20:50:44 2025 -0400

    Ajout initial de note.txt

commit ff0d3f01dd3da70a3eb14537dc2083a8745a15e5
Author: JLubin <ing.jlubin@gmail.com>
Date: Sat May 10 20:43:47 2025 -0400

    Initial commit avec README.md

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

La commande affiche la liste des commits avec :

- l'ID du commit (hash),
 - l'auteur,
 - la date,
 - et le message du commit.
- Pour rendre l'affichage plus compact

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git log --oneline
4af3bb0 (HEAD -> master) Ajouts de lignes dans note.txt
d6d2e3b Ajout initial de note.txt
ff0d3f0 Initial commit avec README.md

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

2. Afficher le détail d'un commit avec git show

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git show
commit 4af3bb09057c6d0fe0b45f2549bee1a4706cb6cc (HEAD -> master)
Author: JLubin <ing.jlubin@gmail.com>
Date: Sat May 10 21:03:21 2025 -0400

    Ajouts de lignes dans note.txt

diff --git a/note.txt b/note.txt
index 5ffb9b6..0e7db3a 100644
--- a/note.txt
+++ b/note.txt
@@ -1,3 @@
 Première version du fichier
+Ajout de la ligne 2
+Ajout de la ligne 3

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

Cela montre :

- les fichiers modifiés,
- les différences ligne par ligne (diff),
- et les métadonnées du commit.

3. Voir qui a modifié chaque ligne d'un fichier avec git blame

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git blame note.txt
d6d2e3b0 (JLubin 2025-05-10 20:50:44 -0400 1) Première version du fichier
4af3bb09 (JLubin 2025-05-10 21:03:21 -0400 2) Ajout de la ligne 2
4af3bb09 (JLubin 2025-05-10 21:03:21 -0400 3) Ajout de la ligne 3

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ |
```

Cela affiche, ligne par ligne :

- l'auteur,
- la date,
- et le commit à l'origine de chaque ligne.

Exercice 4 : Annuler des changements

Annuler avant le commit

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git restore note.txt
```

Exercice 5 : Lier un dépôt local à GitHub

Push vers GitHub


1. Je crée un repository sur Github Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *

 JLabrin-prog ▾

Repository name *

/ ex1_init_repo

✓ ex1_init_repo is available.

Great repository names are short and memorable. Need inspiration? How about [friendly-computing-machine](#) ?

Description (optional)

Initialiser repository



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:



Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None ▾





A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

Create repository

2. Je copie l'url puis je retourne sur GitBash

Quick setup — if you've done this kind of thing before

 Set up in Desktop or  HTTPS  SSH 

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

3. Se lier au dépôt distant

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git remote add origin https://github.com/JLubin-prog/ex1_init_repo.git
```

4. Renommer la branche courante

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (master)
$ git branch -M main
```

5. Transfert vers le dépôt distant

```
JLMsc@DESKTOP-KHTOMKE 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), 835 bytes | 417.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/JLubin-prog/ex1_init_repo.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (main)
$ |
```

6. Vérifier que tout est bien lié

```
JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (main)
$ git remote -v
origin https://github.com/JLubin-prog/ex1_init_repo.git (fetch)
origin https://github.com/JLubin-prog/ex1_init_repo.git (push)

JLMsc@DESKTOP-KHTOMKE MINGW64 ~/ex1_init_repo (main)
$ |
```