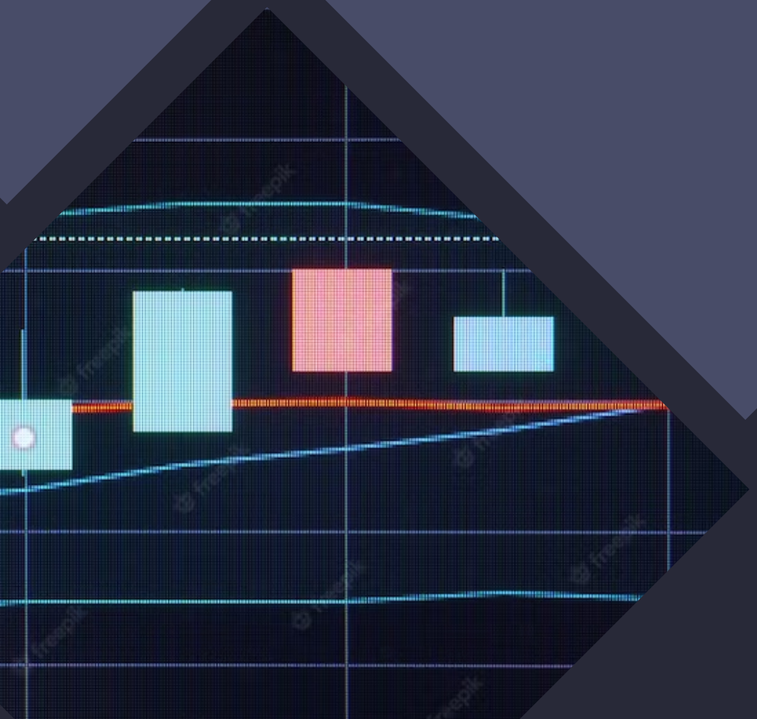
Streamlining Collaboration and Version Control: A Comprehensive Guide to Git & GitHub

Introduction

Welcome to the presentation on Streamlining Collaboration and Version Control: A Comprehensive Guide to **Git & GitHub**. In today's digital age, efficient collaboration and version control are vital for software development projects. This presentation will provide a detailed overview of Git and GitHub and how they can enhance your team's productivity.



Understanding Version Control

Version control is a system that tracks changes to files over time, allowing multiple people to collaborate on a project. **Git** is a distributed version control system that provides a robust framework for managing code. It allows developers to create branches, merge changes, and revert to previous versions. Git also provides a complete history of changes made to a project.

Introducing GitHub

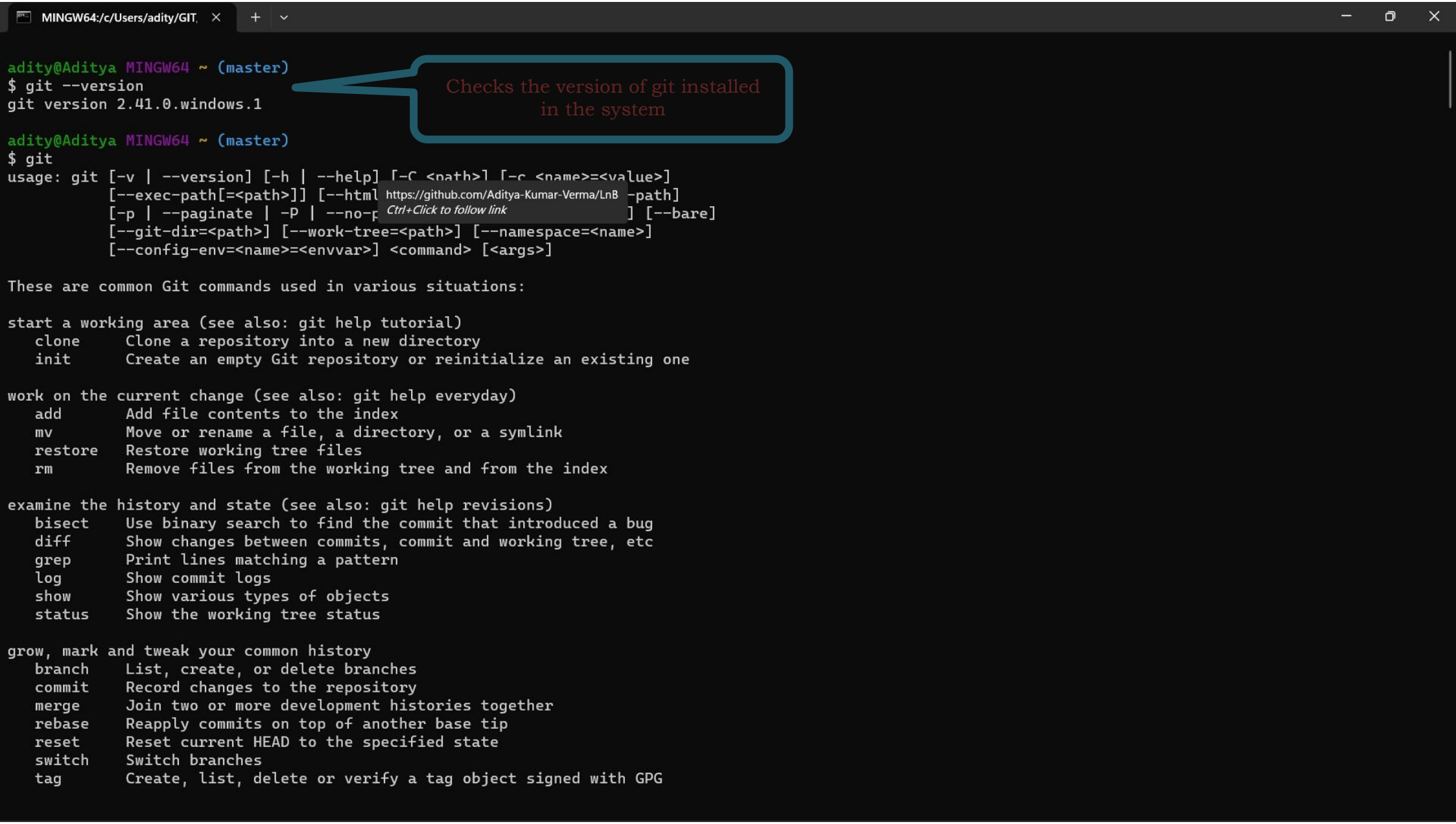
GitHub is a web-based platform that hosts Git repositories. It provides a centralized location for storing and managing code, enabling seamless collaboration among developers. With GitHub, team members can fork repositories, propose changes through pull requests, and review each other's code. GitHub also offers a range of additional features such as issue tracking and project management tools.

Collaborating with Git & GitHub

When collaborating with Git and GitHub, developers can work on different features or bug fixes concurrently by creating separate **branches**. Once the work is completed, changes can be merged back into the main branch. Developers can also review and discuss code changes through **pull requests**. This streamlined collaboration process ensures that everyone is on the same page and reduces conflicts.

Managing Versions with Git

With Git, developers can easily manage different versions of a project. They can create **tags** to mark important milestones or releases, making it easy to revert to a specific version if needed. Git also allows for **branching** and **merging**, enabling parallel development and efficient collaboration. These version control capabilities ensure that projects are organized and changes are tracked effectively.


MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~ (master)
$ git --version
git version 2.41.0.windows.1
Checks the version of git installed in the system
adity@Aditya MINGW64 ~ (master)
$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
[--exec-path[=<path>]] [--html -path]
[-p | --paginate | -P | --no-p] [--bare]
[--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
[--config-env=<name>=<envvar>] <command> [<args>]
https://github.com/Aditya-Kumar-Verma/LnB
Ctrl+Click to follow link
These are common Git commands used in various situations:
start a working area (see also: git help tutorial)
clone Clone a repository into a new directory
init Create an empty Git repository or reinitialize an existing one
work on the current change (see also: git help everyday)
add Add file contents to the index
mv Move or rename a file, a directory, or a symlink
restore Restore working tree files
rm Remove files from the working tree and from the index
examine the history and state (see also: git help revisions)
bisect Use binary search to find the commit that introduced a bug
diff Show changes between commits, commit and working tree, etc
grep Print lines matching a pattern
log Show commit logs
show Show various types of objects
status Show the working tree status
grow, mark and tweak your common history
branch List, create, or delete branches
commit Record changes to the repository
merge Join two or more development histories together
rebase Reapply commits on top of another base tip
reset Reset current HEAD to the specified state
switch Switch branches
tag Create, list, delete or verify a tag object signed with GPG

MINGW64:/c/Users/adity/GIT

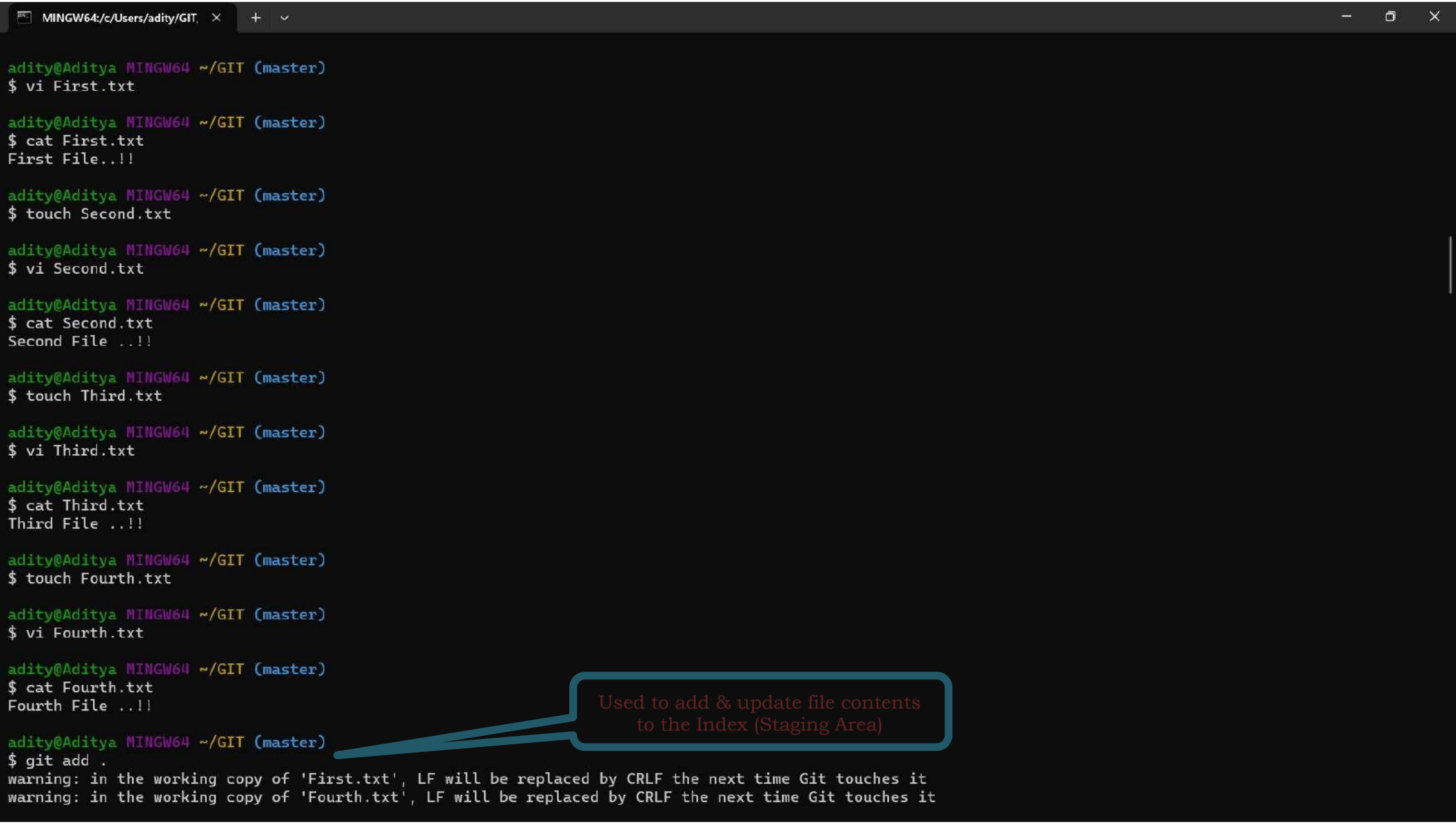
```
collaborate (see also: git help workflows)
   fetch     Download objects and refs from another repository
   pull      Fetch from and integrate with another repository or a local branch
   push      Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.

adity@Aditya MINGW64 ~ (master)
$ git config
usage: git config [<options>]

Config file location
    --global              use global config file
    --system              use system config file
    --local               use repository config file
    --worktree            use per-worktree config file
    -f, --file <file>     use given config file
    --blob <blob-id>      read config from given blob object

Action
    --get                 get value: name [value-pattern]
    --get-all             get all values: key [value-pattern]
    --get-regexp          get values for regexp: name-regex [value-pattern]
    --get-urlmatch        get value specific for the URL: section[.var] URL
    --replace-all         replace all matching variables: name value [value-pattern]
    --add                 add a new variable: name value
    --unset               remove a variable: name [value-pattern]
    --unset-all           remove all matches: name [value-pattern]
    --rename-section      rename section: old-name new-name
    --remove-section      remove a section: name
    -l, --list            list all
    --fixed-value         use string equality when comparing values to 'value-pattern'
    -e, --edit            open an editor
    --get-color           find the color configured: slot [default]
    --get-colorbool       find the color setting: slot [stdout-is-tty]

Type
    -t, --type <type>     value is given this type
```

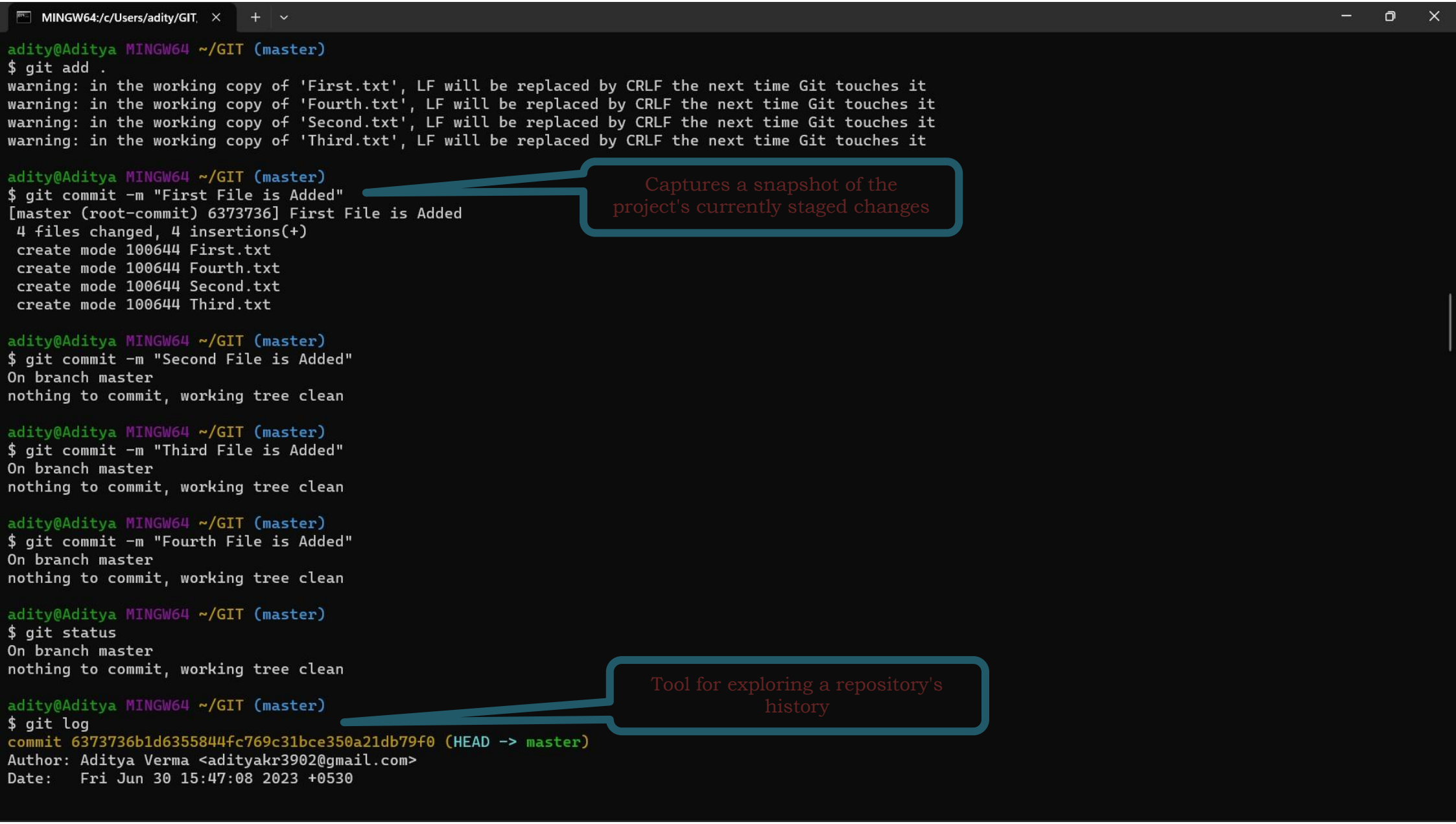
MINGW64:/c/Users/adity/GIT

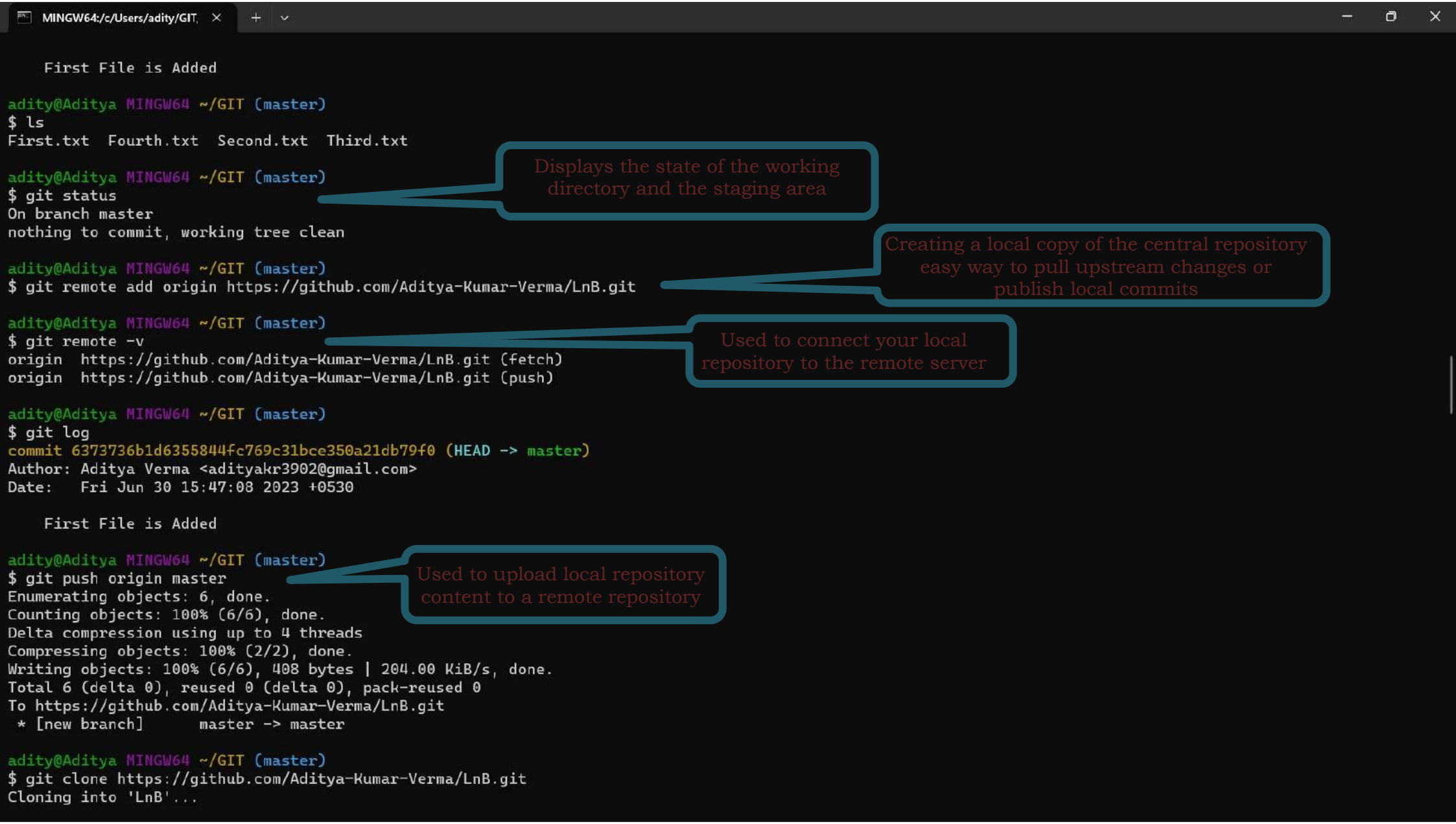
```
Type
    -t, --type <type>     value is given this type
    --bool                value is "true" or "false"
    --int                 value is decimal number
    --bool-or-int         value is --bool or --int
    --bool-or-str         value is --bool or string
    --path                value is a path (file or directory name)
    --expiry-date         value is an expiry date

Other
    -z, --null            terminate values with NUL byte
    --name-only           show variable names only
    --includes            respect include directives on lookup
    --show-origin         show origin of config (file, standard input, blob, command line)
    --show-scope          show scope of config (worktree, local, global, system, command)
    --default <value>     with --get, use default value when missing entry


adity@Aditya MINGW64 ~ (master)
$ mkdir GIT

adity@Aditya MINGW64 ~ (master)
$ cd GIT

adity@Aditya MINGW64 ~/GIT (master)
$ git init
Initialized empty Git repository in C:/Users/adity/GIT/.git/

adity@Aditya MINGW64 ~/GIT (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

adity@Aditya MINGW64 ~/GIT (master)
$ ls

adity@Aditya MINGW64 ~/GIT (master)
$ touch First.txt
```

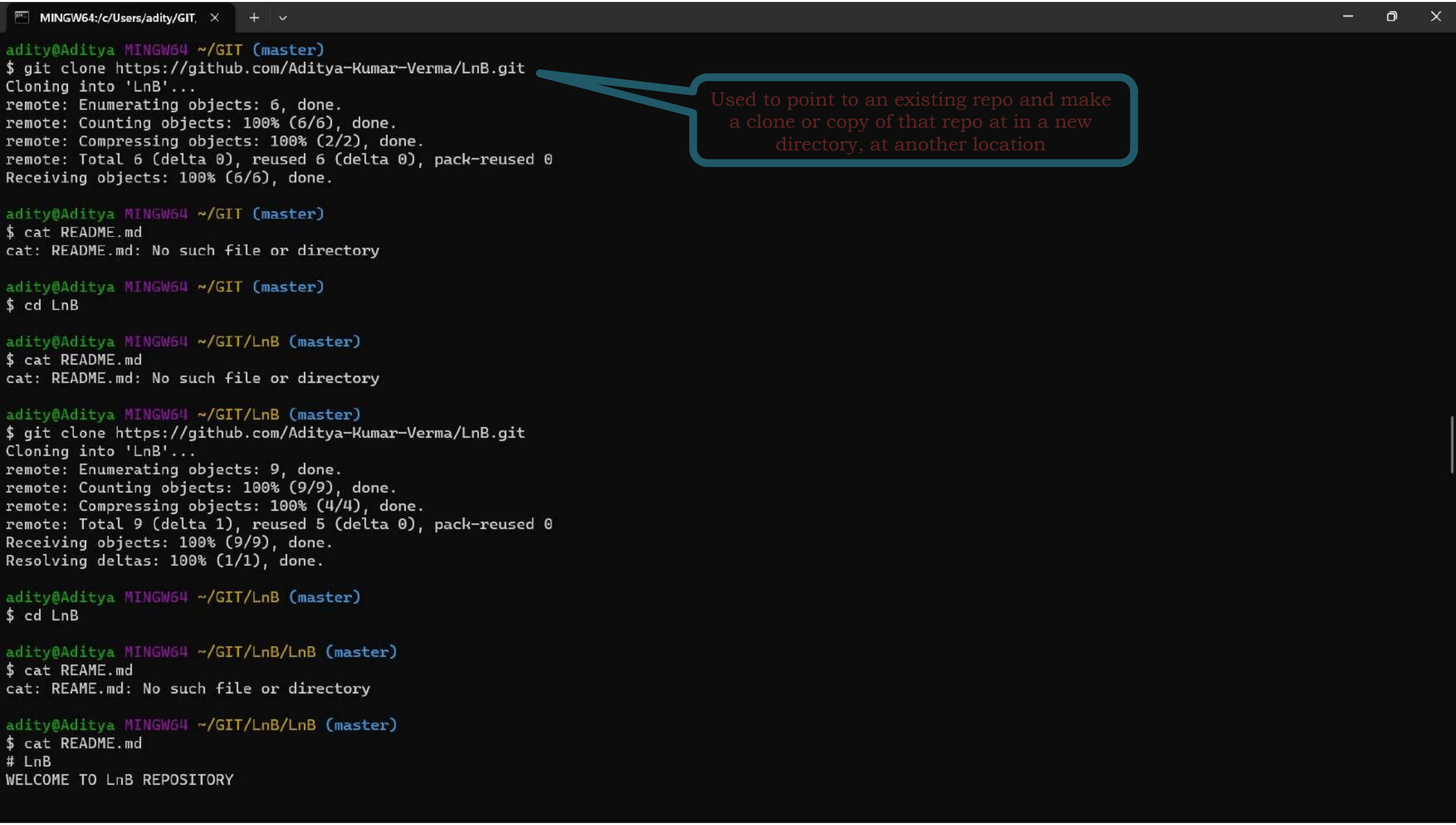
Used to create a plain new directory to work upon

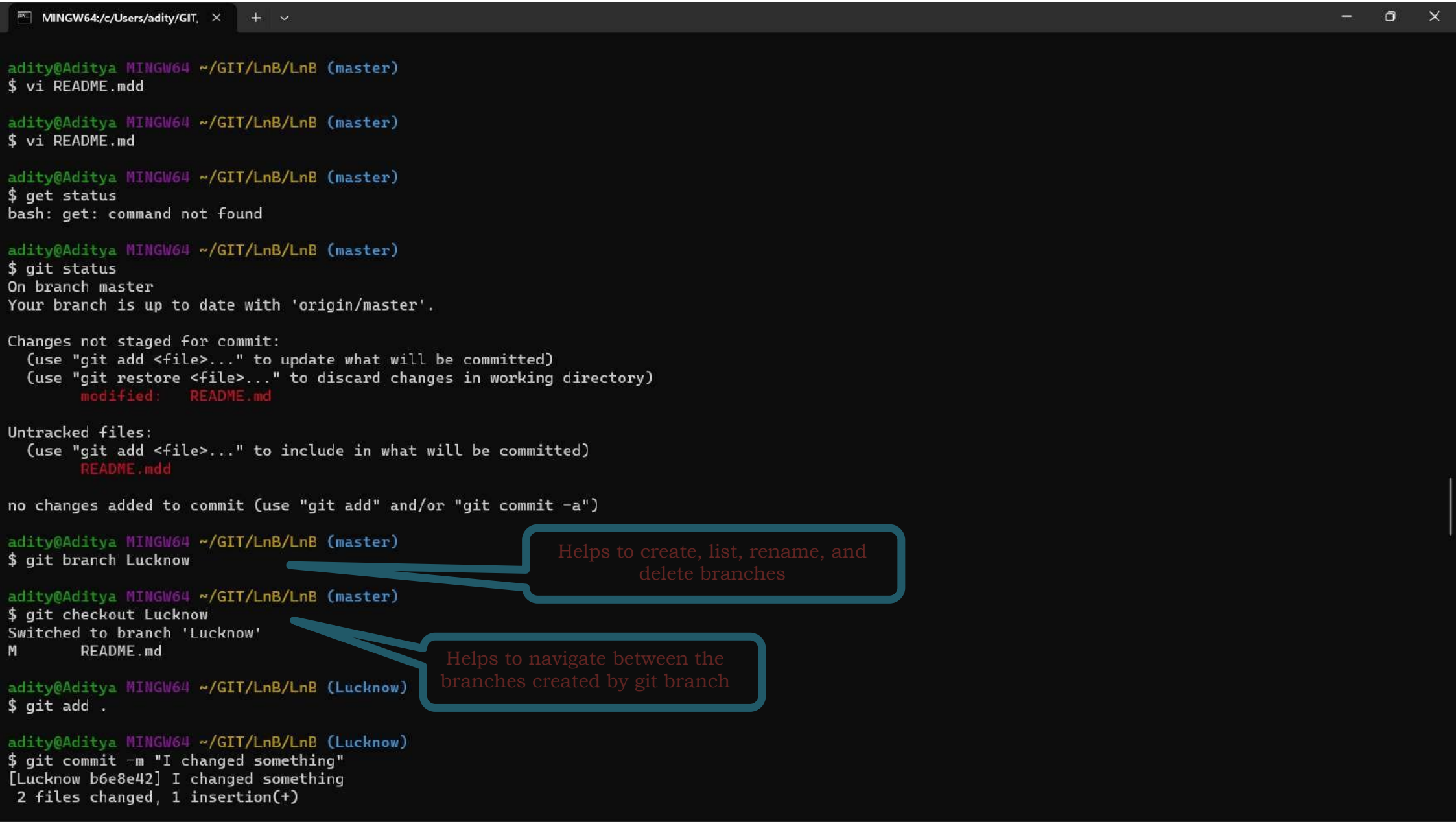
It creates a .git directory inside the plain directory to keep history of files

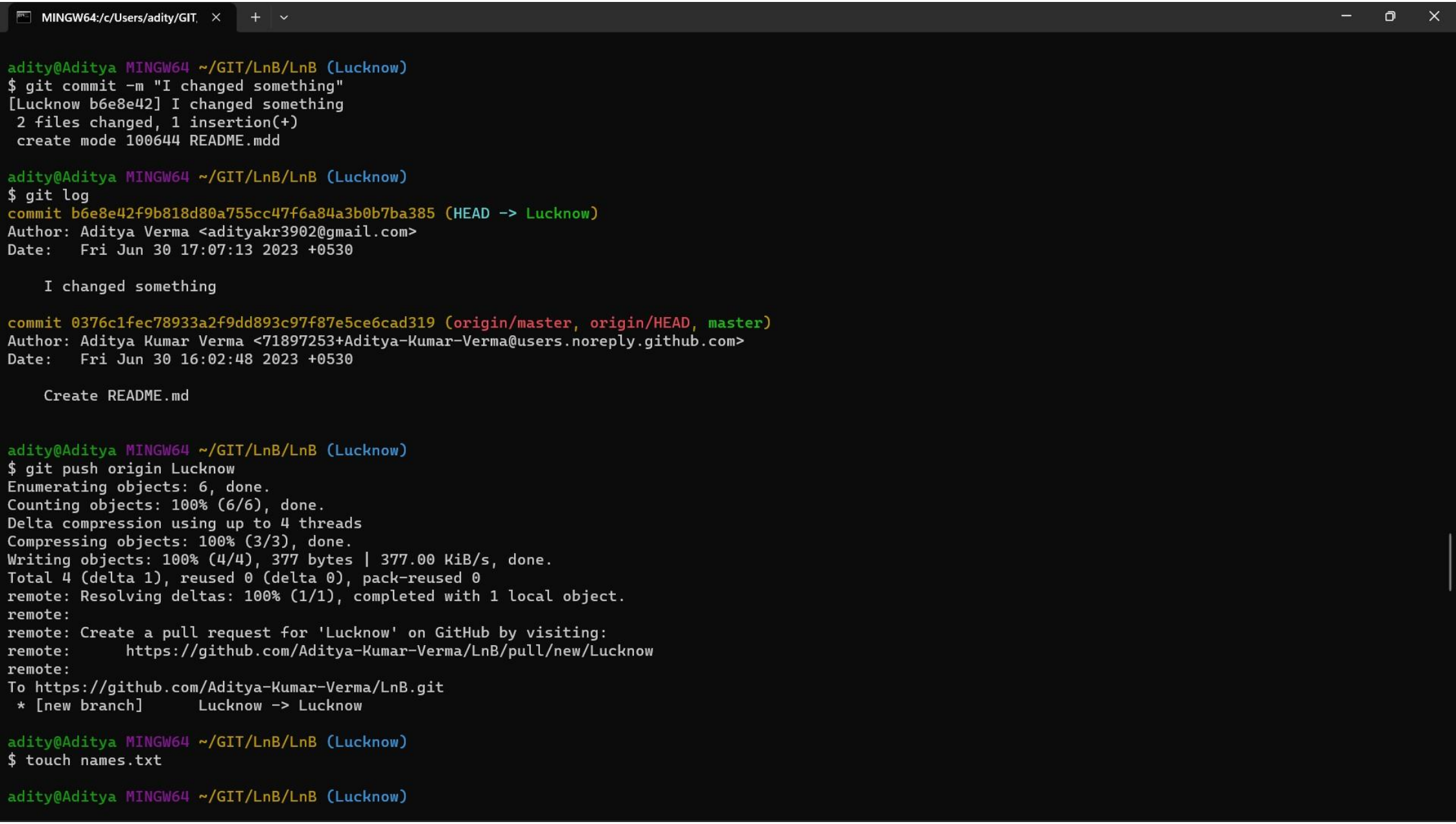

MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT (master)
$ vi First.txt
adity@Aditya MINGW64 ~/GIT (master)
$ cat First.txt
First File..!!
adity@Aditya MINGW64 ~/GIT (master)
$ touch Second.txt
adity@Aditya MINGW64 ~/GIT (master)
$ vi Second.txt
adity@Aditya MINGW64 ~/GIT (master)
$ cat Second.txt
Second File ..!!
adity@Aditya MINGW64 ~/GIT (master)
$ touch Third.txt
adity@Aditya MINGW64 ~/GIT (master)
$ vi Third.txt
adity@Aditya MINGW64 ~/GIT (master)
$ cat Third.txt
Third File ..!!
adity@Aditya MINGW64 ~/GIT (master)
$ touch Fourth.txt
adity@Aditya MINGW64 ~/GIT (master)
$ vi Fourth.txt
adity@Aditya MINGW64 ~/GIT (master)
$ cat Fourth.txt
Fourth File ..!!
adity@Aditya MINGW64 ~/GIT (master)
$ git add .
warning: in the working copy of 'First.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Fourth.txt', LF will be replaced by CRLF the next time Git touches it
Used to add & update file contents
to the Index (Staging Area)

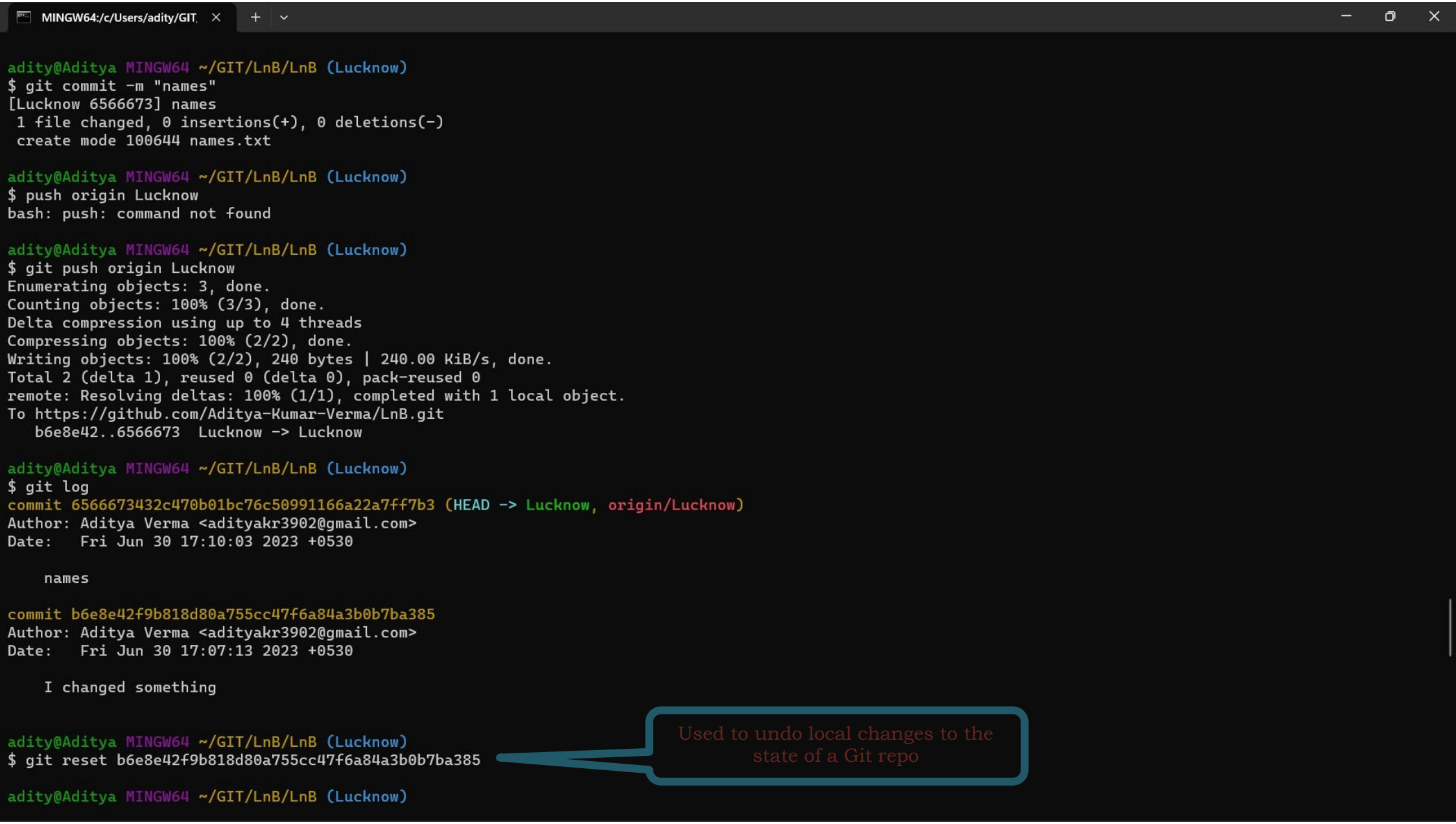

MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT (master)
$ git add .
warning: in the working copy of 'First.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Fourth.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Second.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'Third.txt', LF will be replaced by CRLF the next time Git touches it
adity@Aditya MINGW64 ~/GIT (master)
$ git commit -m "First File is Added"
[master (root-commit) 6373736] First File is Added
4 files changed, 4 insertions(+)
create mode 100644 First.txt
create mode 100644 Fourth.txt
create mode 100644 Second.txt
create mode 100644 Third.txt
Captures a snapshot of the project's currently staged changes
adity@Aditya MINGW64 ~/GIT (master)
$ git commit -m "Second File is Added"
On branch master
nothing to commit, working tree clean
adity@Aditya MINGW64 ~/GIT (master)
$ git commit -m "Third File is Added"
On branch master
nothing to commit, working tree clean
adity@Aditya MINGW64 ~/GIT (master)
$ git commit -m "Fourth File is Added"
On branch master
nothing to commit, working tree clean
adity@Aditya MINGW64 ~/GIT (master)
$ git status
On branch master
nothing to commit, working tree clean
adity@Aditya MINGW64 ~/GIT (master)
$ git log
Tool for exploring a repository's history
commit 6373736b1d6355844fc769c31bce350a21db79f0 (HEAD -> master)
Author: Aditya Verma <adityakr3902@gmail.com>
Date: Fri Jun 30 15:47:08 2023 +0530

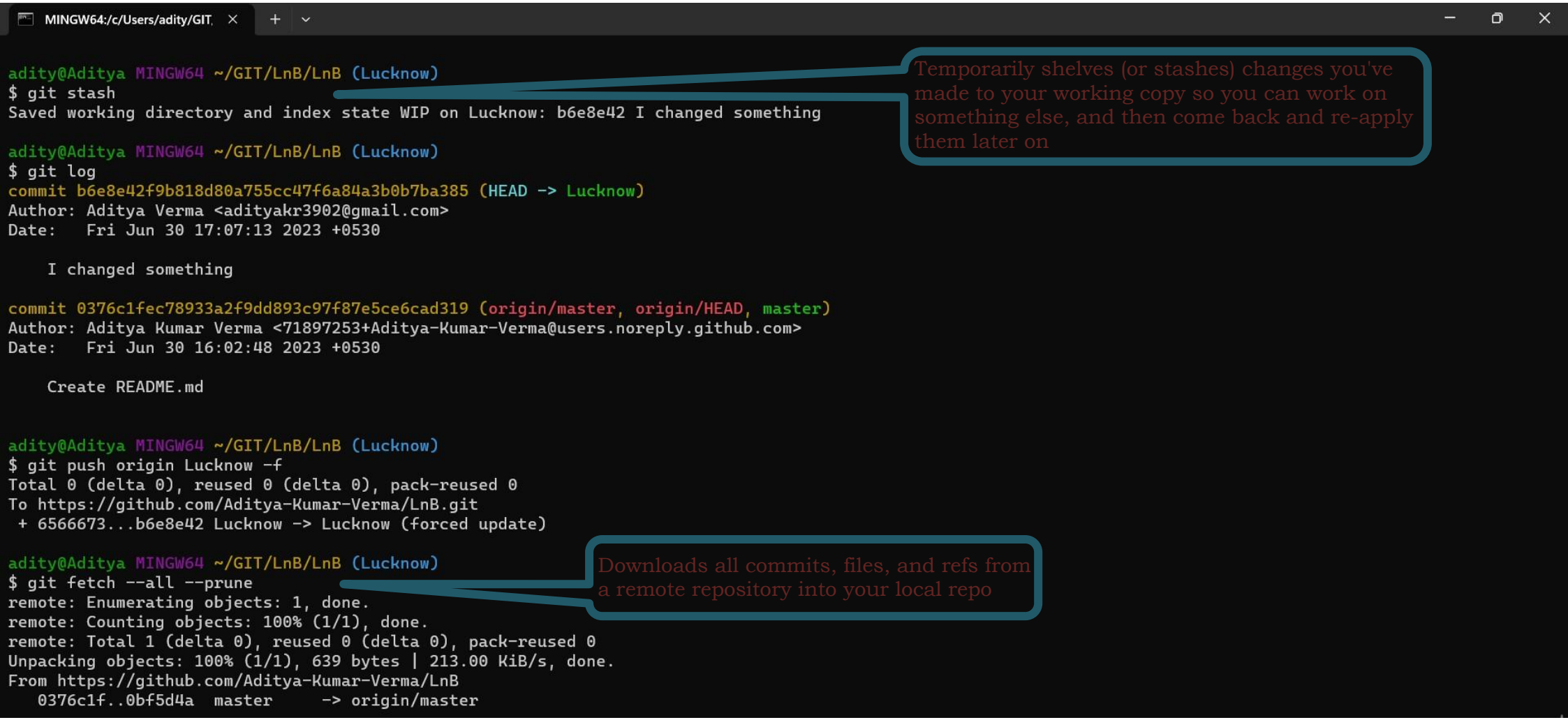

MINGW64:/c/Users/adity/GIT
First File is Added
adity@Aditya MINGW64 ~/GIT (master)
$ ls
First.txt Fourth.txt Second.txt Third.txt
adity@Aditya MINGW64 ~/GIT (master)
$ git status
On branch master
nothing to commit, working tree clean
Displays the state of the working directory and the staging area
adity@Aditya MINGW64 ~/GIT (master)
$ git remote add origin https://github.com/Aditya-Kumar-Verma/LnB.git
Creating a local copy of the central repository easy way to pull upstream changes or publish local commits
adity@Aditya MINGW64 ~/GIT (master)
$ git remote -v
origin https://github.com/Aditya-Kumar-Verma/LnB.git (fetch)
origin https://github.com/Aditya-Kumar-Verma/LnB.git (push)
Used to connect your local repository to the remote server
adity@Aditya MINGW64 ~/GIT (master)
$ git log
commit 6373736b1d6355844fc769c31bce350a21db79f0 (HEAD -> master)
Author: Aditya Verma <adityakr3902@gmail.com>
Date: Fri Jun 30 15:47:08 2023 +0530
First File is Added
adity@Aditya MINGW64 ~/GIT (master)
$ git push origin master
Used to upload local repository content to a remote repository
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 408 bytes | 204.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Aditya-Kumar-Verma/LnB.git
* [new branch] master -> master
adity@Aditya MINGW64 ~/GIT (master)
$ git clone https://github.com/Aditya-Kumar-Verma/LnB.git
Cloning into 'LnB'...


MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT (master)
$ git clone https://github.com/Aditya-Kumar-Verma/LnB.git
Cloning into 'LnB'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
adity@Aditya MINGW64 ~/GIT (master)
$ cat README.md
cat: README.md: No such file or directory
adity@Aditya MINGW64 ~/GIT (master)
$ cd LnB
adity@Aditya MINGW64 ~/GIT/LnB (master)
$ cat README.md
cat: README.md: No such file or directory
adity@Aditya MINGW64 ~/GIT/LnB (master)
$ git clone https://github.com/Aditya-Kumar-Verma/LnB.git
Cloning into 'LnB'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 9 (delta 1), reused 5 (delta 0), pack-reused 0
Receiving objects: 100% (9/9), done.
Resolving deltas: 100% (1/1), done.
adity@Aditya MINGW64 ~/GIT/LnB (master)
$ cd LnB
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ cat REAME.md
cat: REAME.md: No such file or directory
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ cat README.md
# LnB
WELCOME TO LnB REPOSITORY
Used to point to an existing repo and make a clone or copy of that repo at in a new directory, at another location


MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ vi README.mdd
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ vi README.md
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ get status
bash: get: command not found
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ git status
On branch master
Your branch is up to date with 'origin/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: README.md
Untracked files:
(use "git add <file>..." to include in what will be committed)
README.mdd
no changes added to commit (use "git add" and/or "git commit -a")
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ git branch Lucknow
Helps to create, list, rename, and delete branches
adity@Aditya MINGW64 ~/GIT/LnB/LnB (master)
$ git checkout Lucknow
Switched to branch 'Lucknow'
M README.md
Helps to navigate between the branches created by git branch
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git add .
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git commit -m "I changed something"
[Lucknow b6e8e42] I changed something
2 files changed, 1 insertion(+)


MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git commit -m "I changed something"
[Lucknow b6e8e42] I changed something
2 files changed, 1 insertion(+)
create mode 100644 README.mdd
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git log
commit b6e8e42f9b818d80a755cc47f6a84a3b0b7ba385 (HEAD -> Lucknow)
Author: Aditya Verma <adityakr3902@gmail.com>
Date: Fri Jun 30 17:07:13 2023 +0530
I changed something
commit 0376c1fec78933a2f9dd893c97f87e5ce6cad319 (origin/master, origin/HEAD, master)
Author: Aditya Kumar Verma <71897253+Aditya-Kumar-Verma@users.noreply.github.com>
Date: Fri Jun 30 16:02:48 2023 +0530
Create README.md
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git push origin Lucknow
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 377 bytes | 377.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'Lucknow' on GitHub by visiting:
remote: https://github.com/Aditya-Kumar-Verma/LnB/pull/new/Lucknow
remote:
To https://github.com/Aditya-Kumar-Verma/LnB.git
* [new branch] Lucknow -> Lucknow
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ touch names.txt
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)


MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git commit -m "names"
[Lucknow 6566673] names
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 names.txt
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ push origin Lucknow
bash: push: command not found
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git push origin Lucknow
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 240 bytes | 240.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Aditya-Kumar-Verma/LnB.git
b6e8e42..6566673 Lucknow -> Lucknow
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git log
commit 6566673432c470b01bc76c50991166a22a7ff7b3 (HEAD -> Lucknow, origin/Lucknow)
Author: Aditya Verma <adityakr3902@gmail.com>
Date: Fri Jun 30 17:10:03 2023 +0530
names
commit b6e8e42f9b818d80a755cc47f6a84a3b0b7ba385
Author: Aditya Verma <adityakr3902@gmail.com>
Date: Fri Jun 30 17:07:13 2023 +0530
I changed something
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git reset b6e8e42f9b818d80a755cc47f6a84a3b0b7ba385
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
Used to undo local changes to the state of a Git repo


MINGW64:/c/Users/adity/GIT
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git stash
Saved working directory and index state WIP on Lucknow: b6e8e42 I changed something
Temporarily shelves (or stashes) changes you've made to your working copy so you can work on something else, and then come back and re-apply them later on
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git log
commit b6e8e42f9b818d80a755cc47f6a84a3b0b7ba385 (HEAD -> Lucknow)
Author: Aditya Verma <adityakr3902@gmail.com>
Date: Fri Jun 30 17:07:13 2023 +0530
I changed something
commit 0376c1fec78933a2f9dd893c97f87e5ce6cad319 (origin/master, origin/HEAD, master)
Author: Aditya Kumar Verma <71897253+Aditya-Kumar-Verma@users.noreply.github.com>
Date: Fri Jun 30 16:02:48 2023 +0530
Create README.md
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git push origin Lucknow -f
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Aditya-Kumar-Verma/LnB.git
+ 6566673...b6e8e42 Lucknow -> Lucknow (forced update)
adity@Aditya MINGW64 ~/GIT/LnB/LnB (Lucknow)
$ git fetch --all --prune
Downloads all commits, files, and refs from a remote repository into your local repo
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (1/1), 639 bytes | 213.00 KiB/s, done.
From https://github.com/Aditya-Kumar-Verma/LnB
0376c1f..0bf5d4a master -> origin/master

Conclusion

In conclusion, Git and GitHub are powerful tools for streamlining collaboration and version control in software development projects. By leveraging Git's distributed version control system and GitHub's web-based platform, teams can enhance productivity, track changes effectively, and ensure seamless collaboration. Embracing Git and GitHub can significantly improve the efficiency and success of your development projects.