Git Tutorial

MD. Abdullah All Mahmud and Jubayer Ahamed

Git is a free and open-source distributed version control system that tracks changes to files and allows for collaboration on projects. It's a fundamental tool for software development, enabling users to manage different versions of their code, track who made which changes and revert to previous states if needed. Git works on your computer, but you also use it with online services like GitHub, GitLab, or Bitbucket to share your work with others.
Suppose a university software team is developing a **Library Management System** to manage books, track student borrowing, and calculate fines. The project has modules like Book Registration, Student Information, and Fine Calculation. Since new features are added frequently, proper version control is needed to keep track of changes and return to a stable version if required.
The team is using Git locally. They initialize the project with Git Bash, add and commit files with clear messages after every change. For example, if the Book Registration module is updated, they commit it with a note explaining the change. This helps maintain a clear project history and supports teamwork during development.

Core Concepts:

- Repository: A directory where Git tracks all project files and their history.
- Clone: Creating a local copy of a remote repository.
- Commit: Saving a snapshot of changes to the repository with a descriptive message.
- Branch: Creating separate lines of development to work on features or bug fixes independently.
- Merge: Combining changes from different branches into one.
- Push: Uploading local commits to a remote repository.
- Pull: Fetching changes from a remote repository and integrating them into the local repository.
- Staging Area: A temporary area where you stage changes before committing them.
- Remote Repository: A centralized repository hosted online (e.g., on GitHub, GitLab, Bitbucket).

Basic Git Commands:

- git init: Initializes a new Git repository in the current directory.
- git push: Uploads local commits to a remote repository.
- git pull: Fetches changes from a remote repository and merges them.
- git commit -m "Your commit message": Saves staged changes to the repository.
- git branch: Lists, creates, or deletes branches.
- git clone <repository_url>: Clones a remote repository to your local machine.
- git status: Shows the current state of the working directory and staging area.
- git add <file>: Stages a file for the next commit.
- git add . or git add -A: Stages all changes in the current directory.
- git checkout <branch_name>: Switches to a different branch.
- git merge <branch_name>: Merges changes from a specified branch into the current branch.
- git log: Displays the commit history.
- git diff: Shows the differences between files.

Workflow Example:

1. Initialize: Start a new project or clone an existing one.
2. Add Files: Use git add to stage files you've modified or created.
3. Commit Changes: Use git commit to save the staged changes with a descriptive message.
4. Push Changes: Use git push to upload your local commits to a remote repository.
5. Collaborate: Use branches, pull requests, and merges to work with others.

Follow the steps below to practice the basic Git commands on your computer:

1. Go to the location in your computer from where you want to push a file or folder.

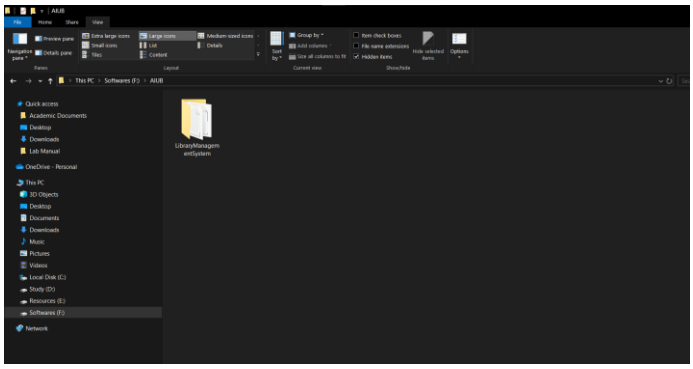

Figure 1

2. Open Git Bash

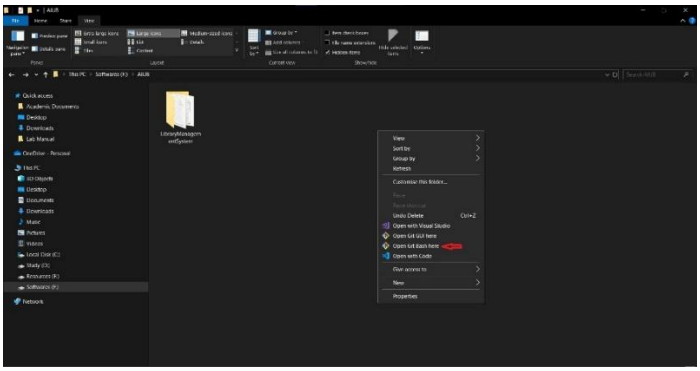

Figure 2

3. Configure User Info (only first time per computer)

git config --global user.name "Your Name"

git config --global user.email "your_email@example.com"

Figure 3

Check if it is saved correctly:

git config --global --list

Figure 4

4. Create a New Project Folder

mkdir LibraryManagementSystem

cd LibraryManagementSystem

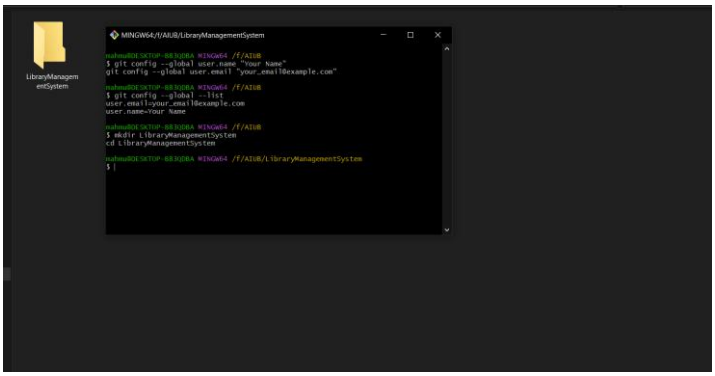

Figure 5

5. Initialize Git in This Folder

git init

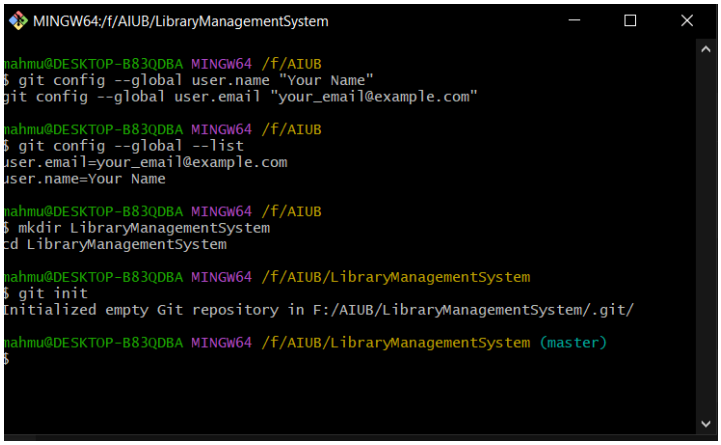

Figure 6

Check current status:

git status

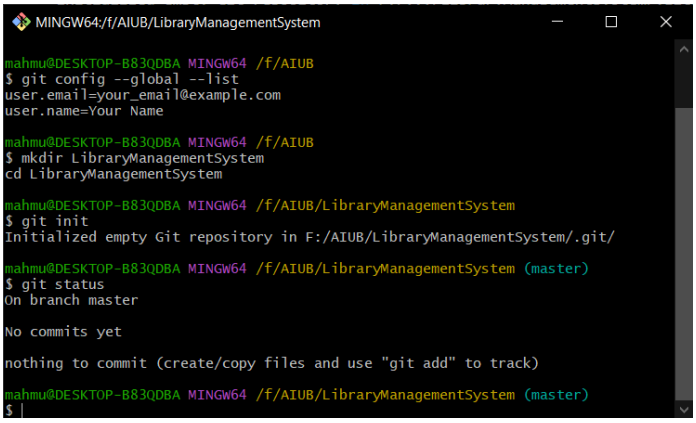

Figure 7

6. Create the First File

echo "Library Management System - Initial Version" > README.txt

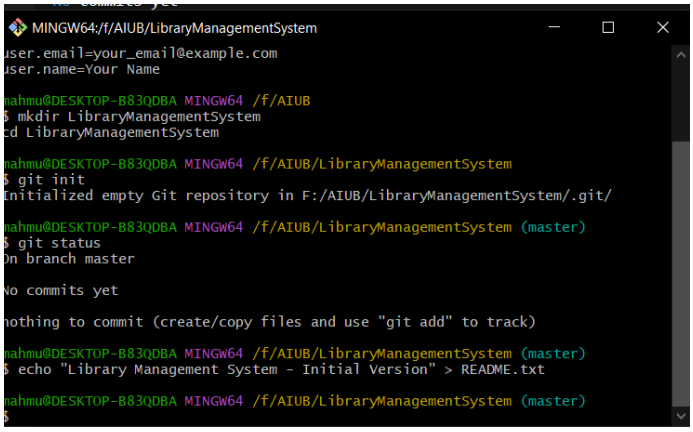

Figure 8

7. Add the File to Staging

git add README.txt

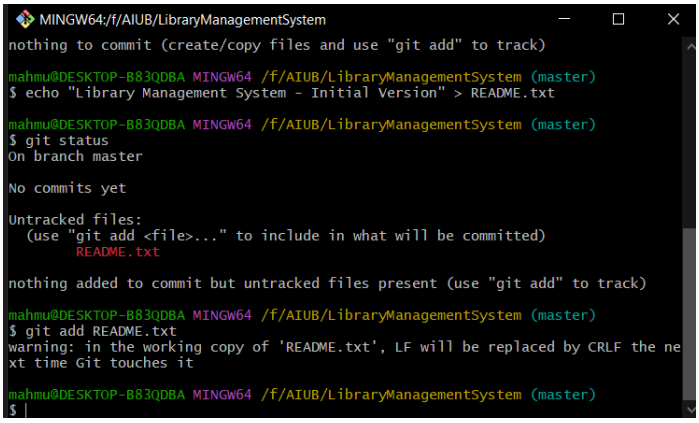

Figure 9

8. Commit the First Change

git commit -m "Initial commit: Added project description"

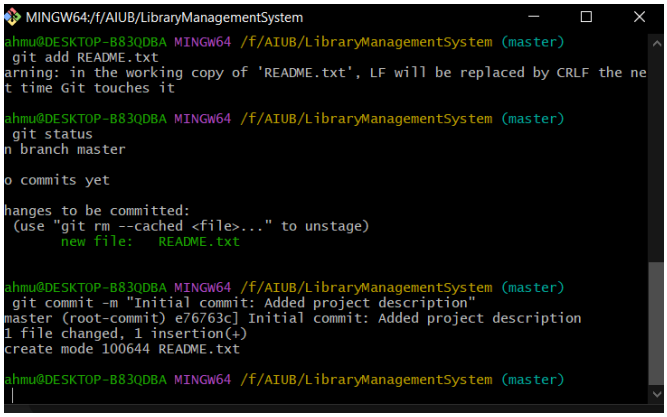

Figure 10

9. Modify the File and Commit Again

- Edit the README.txt file (add some text manually in Notepad, e.g., “Book Registration Module details”).
- Check differences:

  git diff

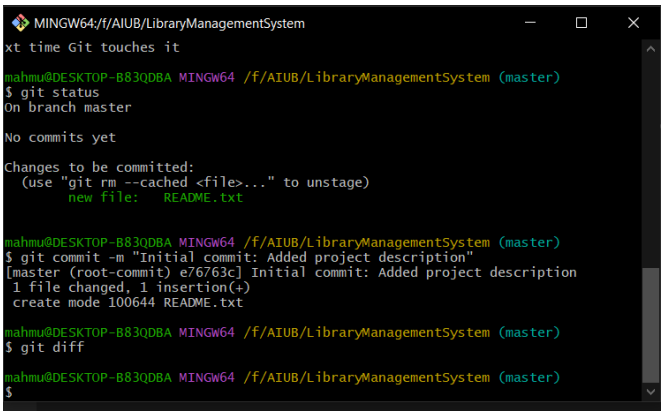

Figure 11

- Stage and commit:

  git add README.txt

  git commit -m "Added Book Registration Module details"

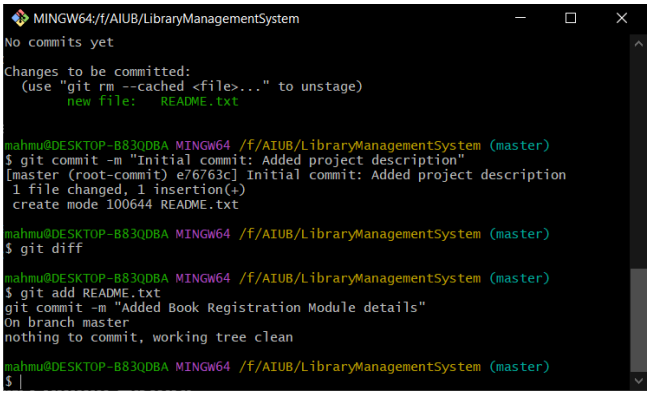

Figure 12

10. **View Commit History**

git log

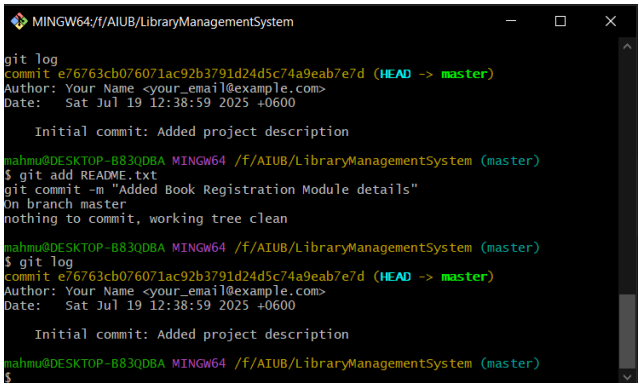

Figure 13

11. **Add Another File**

echo "Student Information Module details" > student_module.txt

git add student_module.txt

git commit -m "Added Student Information Module"

git log

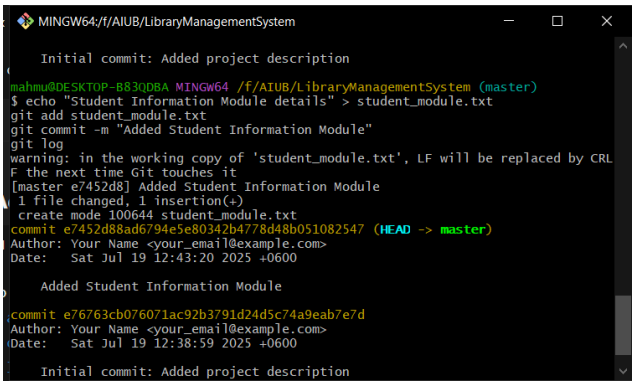

Figure 14

Lab Task:

Create a GitHub repository named GitLabTask . It must contain:

- README.txt file
- Upload the file using git commands
- At least one commit

Then, share the link of your GitHub repository.

References:

1. https://www.geeksforgeeks.org/git/git-tutorial/
2. https://www.w3schools.com/git/
3. https://git-scm.com/docs/gittutorial
4. https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository
5. https://www.atlassian.com/git