# **Hướng Dẫn Sử Dụng Git Trên Windows**

## **1. Cài Đặt Git Trên Windows**

### **1.1. Tải Git**

* Truy cập trang web chính thức của Git tại địa chỉ:<https://git-scm.com/>.
* Nhấn vào liên kết tải về để chọn phiên bản phù hợp với hệ điều hành Windows của bạn (32-bit hoặc 64-bit).

### **1.2. Cài Đặt Git**

Sau khi tải về, thực hiện các bước cài đặt như sau:

* Mở file cài đặt Git đã tải về và làm theo hướng dẫn từng bước:
  + **Select Components**: Để các tùy chọn mặc định và đảm bảo đã chọn "Git Bash Here" và "Git GUI Here".
  + **Editor Configuration**: Lựa chọn editor mặc định mà bạn muốn sử dụng với Git (ví dụ: Vim, Nano, hoặc Visual Studio Code).
  + **PATH environment**: Chọn **Git from the command line and also from 3rd-party software** nếu bạn muốn sử dụng Git trong CMD (khuyến nghị).
  + **HTTPS Transport Backend**: Giữ tùy chọn mặc định **Use the OpenSSL library**.
  + **Configuring line ending conversions**: Chọn **Checkout Windows-style, commit Unix-style line endings** để đảm bảo tính tương thích giữa các hệ điều hành.
* Nhấn **Next** để hoàn tất cài đặt.

### **1.3. Kiểm Tra Cài Đặt Git**

Sau khi cài đặt xong, mở **Git Bash** hoặc **Command Prompt** và chạy lệnh sau để kiểm tra xem Git đã cài đặt thành công chưa:

| git --version |
| --- |

Nếu Git được cài đặt thành công, bạn sẽ thấy phiên bản Git hiện tại.

## **2. Cấu Hình Git**

Trước khi sử dụng Git, bạn cần cấu hình thông tin cá nhân để nhận diện người thực hiện commit.

### **2.1. Cấu Hình Tên Người Dùng**

Mở **Git Bash** và nhập lệnh sau:

| git config --global user.name "Tên của bạn" |
| --- |

### **2.2. Cấu Hình Địa Chỉ Email**

Cấu hình địa chỉ email của bạn bằng lệnh sau:

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

**Lưu ý**: Email này phải trùng với email bạn sử dụng trên GitHub nếu bạn định đẩy mã nguồn lên GitHub.

### **2.3. Kiểm Tra Cấu Hình**

Để kiểm tra lại các thông tin cấu hình vừa nhập, sử dụng lệnh:

| git config --global --list |
| --- |

Lệnh này sẽ liệt kê các thông tin cấu hình đã thiết lập.

## **3. Các Thao Tác Cơ Bản Với Git**

### **3.1. Khởi Tạo Repository Mới**

Để khởi tạo một Git repository mới trên máy tính, bạn cần thực hiện các bước sau:

#### **Bước 1: Mở Git Bash hoặc Command Prompt**

Trước tiên, bạn cần mở Git Bash (nếu đã cài đặt Git trên Windows) hoặc Command Prompt.

#### **Bước 2: Chuyển Đến Thư Mục Dự Án**

Sử dụng lệnh cd (change directory) để điều hướng đến thư mục nơi bạn muốn khởi tạo Git repository. Nếu chưa có thư mục dự án, bạn có thể tạo thư mục mới.

**Ví dụ**:

Nếu bạn muốn làm việc trong thư mục D:\Projects\MyProject, bạn chạy lệnh:

| cd /d/Projects/MyProject |
| --- |

Nếu thư mục này chưa tồn tại, bạn có thể tạo mới nó bằng lệnh:

| mkdir /d/Projects/MyProject cd /d/Projects/MyProject |
| --- |

#### **Bước 3: Khởi Tạo Git Repository**

Sau khi điều hướng đến thư mục mong muốn, khởi tạo Git repository bằng lệnh sau:

| git init |
| --- |

**Giải thích**: Lệnh này sẽ tạo một thư mục ẩn .git trong thư mục hiện tại, giúp Git quản lý mã nguồn và theo dõi các thay đổi trong dự án.

**Ví dụ kết quả**:

| Initialized empty Git repository in D:/Projects/MyProject/.git/ |
| --- |

### **3.2. Thêm File Vào Git**

Sau khi khởi tạo repository, bạn cần thêm các file vào vùng chờ (staging area) để Git quản lý và theo dõi.

#### **Bước 1: Tạo File Mới (Tùy Chọn)**

Nếu chưa có file nào trong thư mục, bạn có thể tạo file mới để thử nghiệm.

**Ví dụ**: Bạn có thể tạo file README.md bằng lệnh sau trong Git Bash:

| echo "# My Project" > README.md |
| --- |

Lệnh này sẽ tạo ra file README.md với nội dung dòng "# My Project".

#### **Bước 2: Thêm File Vào Git**

Để thêm một file cụ thể vào Git, sử dụng lệnh:

| git add README.md |
| --- |

Nếu bạn muốn thêm tất cả các file trong thư mục vào vùng chờ, sử dụng lệnh:  
bash  
**Giải thích**:

* git add README.md: Thêm file README.md vào vùng chờ.
* git add .: Thêm tất cả các file và thư mục (bao gồm cả các file đã chỉnh sửa và file mới tạo) trong thư mục hiện tại vào vùng chờ.

**Ví dụ**:

| git add README.md |
| --- |

### **3.3. Tạo Commit**

Sau khi đã thêm file vào vùng chờ, bạn cần tạo một commit để lưu lại trạng thái hiện tại của dự án.

#### **Bước 1: Tạo Commit**

Sử dụng lệnh sau để tạo một commit với thông điệp mô tả thay đổi:

| git commit -m "Initial commit" |
| --- |

**Giải thích**:

* -m: Tham số này chỉ ra rằng bạn đang thêm một thông điệp commit ngay sau nó.
* "Initial commit": Đây là thông điệp mô tả cho commit này, thông thường commit đầu tiên trong một dự án sẽ có thông điệp là "Initial commit".

**Ví dụ kết quả**:

| [main (root-commit) 5a2c1a4] Initial commit  1 file changed, 1 insertion(+)  create mode 100644 README.md |
| --- |

**Giải thích kết quả**:

* main: Nhánh mà commit đang được tạo (nhánh chính mặc định).
* root-commit: Đây là commit đầu tiên trong repository.
* 5a2c1a4: Mã hash của commit, Git sử dụng mã này để theo dõi commit.
* 1 file changed, 1 insertion(+): Commit này đã thay đổi một file (README.md) và thêm một dòng vào file này.

### **Ví Dụ Cụ Thể Cho Toàn Bộ Quá Trình**

Giả sử bạn muốn khởi tạo một dự án mới trong thư mục D:\Projects\MyProject, thêm file README.md vào dự án và tạo commit đầu tiên:

1. **Tạo thư mục dự án và chuyển đến thư mục đó**:

| mkdir /d/Projects/MyProject cd /d/Projects/MyProject |
| --- |

1. **Khởi tạo Git repository:**

| git init |
| --- |

1. **Tạo file README.md**:

| echo "# My Project" > README.md |
| --- |

1. **Thêm file vào Git**:

| git add README.md |
| --- |

1. **Tạo commit với thông điệp "Initial commit"**:

| git commit -m "Initial commit" |
| --- |

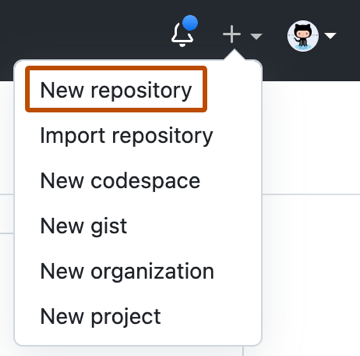
## **4. Kết Nối Với GitHub**

### **4.1. Tạo Repository Trên GitHub**

#### **Bước 1: Đăng nhập vào GitHub**

* Truy cập trang web [GitHub](https://github.com/) và đăng nhập vào tài khoản của bạn.

#### **Bước 2: Tạo repository mới**

1. Sau khi đăng nhập, bạn sẽ được chuyển hướng đến trang chủ GitHub.
2. Ở góc phải phía trên màn hình, nhấn vào biểu tượng dấu **+**, sau đó chọn **New repository**.  
   

#### **Bước 3: Điền thông tin cần thiết**

* **Repository name**: Nhập tên cho repository (ví dụ: MyProject). Đây sẽ là tên bạn sử dụng để liên kết với repository cục bộ sau này.
* **Description**: (Không bắt buộc) Bạn có thể nhập mô tả ngắn gọn về repository này.
* **Public/Private**: Chọn chế độ hiển thị:
  + **Public**: Repository của bạn sẽ có thể được mọi người nhìn thấy.
  + **Private**: Chỉ bạn và những người bạn mời mới có thể truy cập.
* Bạn có thể bỏ qua các tùy chọn như Initialize this repository with a README, .gitignore, hoặc LICENSE vì bạn sẽ đẩy mã nguồn từ máy tính cục bộ sau.

#### **Bước 4: Nhấn nút "Create Repository"**

Sau khi điền xong thông tin, nhấn vào nút **Create Repository** để tạo repository mới.

### **4.2. Liên Kết Repository Cục Bộ Với GitHub**

#### **Bước 1: Tìm URL của repository trên GitHub**

* Sau khi tạo repository thành công, bạn sẽ thấy một URL dạng https://github.com/tên\_người\_dùng/tên\_repository.git ở phần hướng dẫn trên trang GitHub.

**Ví dụ**: Nếu tên người dùng của bạn là user123 và tên repository là MyProject, URL sẽ là:

| https://github.com/user123/MyProject.git |
| --- |

#### **Bước 2: Liên kết repository cục bộ với GitHub**

* Mở **Git Bash** hoặc **Command Prompt** trên máy tính của bạn.
* Chuyển đến thư mục chứa repository cục bộ của bạn (nơi bạn đã tạo repository trong các bước trước đó).

**Ví dụ**:

| cd /d/Projects/MyProject |
| --- |

Thêm GitHub repository làm remote cho repository cục bộ:

| git remote add origin https://github.com/user123/MyProject.git |
| --- |

**Giải thích**:

* git remote add origin: Lệnh này dùng để thêm một remote có tên là origin cho repository của bạn. origin thường là tên mặc định cho remote của GitHub repository.
* https://github.com/user123/MyProject.git: Đây là URL của repository GitHub mà bạn đã tạo.

**Lưu ý**: Thay thế user123 bằng tên người dùng GitHub của bạn và MyProject bằng tên repository thực tế của bạn trên GitHub.

### **4.3. Đẩy Mã Nguồn Lên GitHub**

#### **Bước 1: Kiểm tra nhánh hiện tại**

Trước khi đẩy mã nguồn lên GitHub, bạn cần đảm bảo rằng mình đang làm việc trên nhánh chính (main). Bạn có thể kiểm tra nhánh hiện tại bằng lệnh:

| git branch |
| --- |

Nếu bạn chưa ở trên nhánh main, chuyển sang nhánh này bằng lệnh:

| git checkout main |
| --- |

#### **Bước 2: Đẩy mã nguồn lên GitHub**

Sử dụng lệnh sau để đẩy các commit từ repository cục bộ lên repository GitHub:

| git push -u origin main |
| --- |

**Giải thích**:

* git push: Lệnh này dùng để đẩy các commit từ repository cục bộ lên remote repository.
* -u origin main: Đẩy nhánh main của repository cục bộ lên remote origin (đã thiết lập ở bước trên) và đặt origin/main làm nhánh mặc định cho các lần push tiếp theo.

#### **Ví dụ kết quả:**

| Enumerating objects: 5, done. Counting objects: 100% (5/5), done. Delta compression using up to 4 threads Compressing objects: 100% (3/3), done. Writing objects: 100% (5/5), 428 bytes | 214.00 KiB/s, done. Total 5 (delta 0), reused 0 (delta 0) To https://github.com/user123/MyProject.git  \* [new branch] main -> main Branch 'main' set up to track remote branch 'main' from 'origin'. |
| --- |

**Giải thích kết quả**:

* Enumerating objects: Git đang đếm số lượng file cần đẩy lên.
* Compressing objects: Git nén các file để giảm dung lượng dữ liệu truyền đi.
* Writing objects: Các file đang được đẩy lên GitHub.
* To https://github.com/user123/MyProject.git: Đây là URL repository mà mã nguồn của bạn đã được đẩy lên.
* Branch 'main' set up to track remote branch 'main' from 'origin': Nhánh main của bạn đã được thiết lập để theo dõi nhánh main trên origin (GitHub).

### **5. Clone Repository Từ GitHub Về Máy Tính**

#### **Bước 1: Mở Git Bash hoặc Command Prompt**

Trước tiên, bạn cần mở **Git Bash** (nếu bạn đã cài đặt Git trên Windows) hoặc **Command Prompt**.

#### **Bước 2: Lấy URL của Repository Từ GitHub**

Để clone một repository từ GitHub về máy tính, bạn cần lấy URL của repository mà bạn muốn sao chép. Để làm điều này:

1. Truy cập trang GitHub của repository bạn muốn clone.
2. Ở góc trên bên phải của repository, nhấn vào nút **Code**.

Bạn sẽ thấy một trường URL với định dạng tương tự như:

| https://github.com/tên\_người\_dùng/tên\_repository.git |
| --- |

Ví dụ: Nếu bạn muốn clone một repository tên là MyProject từ người dùng user123, URL sẽ là:

| https://github.com/user123/MyProject.git |
| --- |

#### **Bước 3: Clone Repository**

Sử dụng lệnh git clone với URL của repository để sao chép repository về máy tính của bạn.

**Cú pháp**:

| git clone https://github.com/tên\_người\_dùng/tên\_repository.git |
| --- |

**Ví dụ**: Nếu bạn muốn clone repository MyProject từ người dùng user123, lệnh sẽ như sau:

| git clone https://github.com/user123/MyProject.git |
| --- |

**Giải thích**:

* git clone: Lệnh này dùng để sao chép (clone) toàn bộ repository từ GitHub về máy tính.
* https://github.com/user123/MyProject.git: Đây là URL của repository mà bạn muốn clone.

**Ví dụ kết quả**:

| Cloning into 'MyProject'... remote: Enumerating objects: 10, done. remote: Counting objects: 100% (10/10), done. remote: Compressing objects: 100% (7/7), done. Receiving objects: 100% (10/10), 5.20 KiB | 2.60 MiB/s, done. |
| --- |

**Giải thích kết quả**:

* Cloning into 'MyProject': Git đang sao chép repository MyProject về máy tính của bạn.
* Enumerating objects: Git đang đếm các đối tượng (file, thư mục, commit) cần được sao chép.
* Receiving objects: Quá trình tải xuống các đối tượng từ GitHub về máy tính của bạn đã hoàn thành.

#### **Bước 4: Di Chuyển Vào Thư Mục Vừa Clone**

Sau khi repository đã được sao chép về máy tính, bạn sẽ thấy một thư mục mới được tạo ra với tên của repository (trong ví dụ này là MyProject). Để bắt đầu làm việc với mã nguồn trong repository này, bạn cần di chuyển vào thư mục vừa được clone.

Sử dụng lệnh cd để di chuyển vào thư mục:

**Cú pháp**:

| cd tên\_repository |
| --- |

**Ví dụ**:

| cd MyProject |
| --- |

**Giải thích**:

* cd: Lệnh này được sử dụng để thay đổi thư mục làm việc hiện tại.
* MyProject: Tên của thư mục chứa repository vừa clone.

#### **Bước 5: Bắt Đầu Làm Việc Với Mã Nguồn Cục Bộ**

Bây giờ bạn đã di chuyển vào thư mục chứa repository vừa clone, bạn có thể thực hiện các hành động như chỉnh sửa mã nguồn, tạo commit, và đẩy các thay đổi trở lại GitHub.

**Ví dụ các bước tiếp theo**:

1. **Chỉnh sửa mã nguồn**: Sử dụng editor bạn thích (Visual Studio Code, Sublime Text, hoặc bất kỳ trình soạn thảo nào) để chỉnh sửa mã nguồn trong repository.
2. **Thêm và commit thay đổi**: Sau khi chỉnh sửa, bạn có thể thêm các thay đổi vào Git bằng lệnh git add và tạo commit bằng lệnh git commit.

**Ví dụ**:

| git add . git commit -m "Cập nhật tính năng mới" |
| --- |

1. **Đẩy các thay đổi lên GitHub**: Sau khi đã commit các thay đổi, bạn có thể đẩy chúng trở lại GitHub bằng lệnh:

| git push origin main |
| --- |

### **6. Các Lệnh Git Cơ Bản Khác**

### **6.1. Pull Thay Đổi Từ GitHub**

#### **Mô tả:**

Nếu repository trên GitHub đã có thay đổi từ những thành viên khác hoặc từ những lần cập nhật trước và bạn muốn đồng bộ các thay đổi này vào repository cục bộ của mình, bạn cần sử dụng lệnh git pull.

#### **Cú pháp:**

| git pull origin main |
| --- |

#### **Giải thích:**

* git pull: Lệnh này sẽ lấy các thay đổi từ repository trên GitHub và hợp (merge) các thay đổi đó vào bản cục bộ của bạn.
* origin: Đây là tên mặc định của remote repository trên GitHub.
* main: Nhánh chính mà bạn muốn đồng bộ với repository cục bộ.

#### **Ví dụ kết quả:**

| remote: Enumerating objects: 5, done. remote: Counting objects: 100% (5/5), done. remote: Compressing objects: 100% (3/3), done. Unpacking objects: 100% (5/5), done. From https://github.com/user123/MyProject  \* branch main -> FETCH\_HEAD Updating a1b2c3d..d4e5f6g Fast-forward  README.md | 2 ++  1 file changed, 2 insertions(+) |
| --- |

**Giải thích kết quả**:

* remote: Enumerating objects: Git đang kiểm tra các thay đổi từ repository trên GitHub.
* Updating a1b2c3d..d4e5f6g: Git đang cập nhật mã nguồn của bạn từ commit a1b2c3d đến commit d4e5f6g.
* 1 file changed, 2 insertions(+): 1 file đã được thay đổi, và có 2 dòng mới được thêm vào.

### **6.2. Tạo Nhánh Mới**

#### **Mô tả:**

Trong quá trình phát triển phần mềm, bạn thường cần tạo một nhánh mới để thêm tính năng mới hoặc sửa lỗi mà không ảnh hưởng đến mã nguồn chính trên nhánh main.

#### **Cú pháp:**

| git checkout -b tên\_nhánh\_mới |
| --- |

#### **Giải thích:**

* git checkout -b: Lệnh này vừa tạo một nhánh mới và ngay lập tức chuyển sang nhánh đó.
* tên\_nhánh\_mới: Đây là tên của nhánh mà bạn muốn tạo, ví dụ như feature\_login hoặc bugfix\_ui.

#### **Ví dụ:**

| git checkout -b feature\_login |
| --- |

**Giải thích**:

* Git sẽ tạo một nhánh mới có tên feature\_login và chuyển sang nhánh này để bạn có thể bắt đầu phát triển tính năng mới.

**Ví dụ kết quả**:

| Switched to a new branch 'feature\_login' |
| --- |

### **6.3. Hợp Nhánh (Merge)**

#### **Mô tả:**

Khi bạn hoàn thành công việc trên nhánh mới (ví dụ: nhánh feature\_login), bạn cần hợp (merge) nhánh này vào nhánh chính (main) để tích hợp các thay đổi mới vào dự án.

#### **Cú pháp:**

| git checkout main git merge tên\_nhánh |
| --- |

#### **Giải thích:**

* git checkout main: Chuyển về nhánh chính (main) để chuẩn bị hợp nhánh.
* git merge tên\_nhánh: Hợp các thay đổi từ nhánh tên\_nhánh vào nhánh chính (main).

#### **Ví dụ:**

Giả sử bạn đã hoàn thành công việc trên nhánh feature\_login và muốn hợp nó vào nhánh main:

| git checkout main git merge feature\_login |
| --- |

**Ví dụ kết quả**:

| Updating d4e5f6g..h7i8j9k Fast-forward  login.js | 10 ++++++++++  1 file changed, 10 insertions(+) |
| --- |

**Giải thích kết quả**:

* Fast-forward: Git đã hợp nhánh feature\_login vào nhánh main mà không cần giải quyết xung đột.
* login.js | 10 ++++++++++: File login.js đã được thay đổi với 10 dòng mới được thêm vào.

### **6.4. Xem Lịch Sử Commit**

#### **Mô tả:**

Để xem danh sách các commit trước đây trong dự án của bạn, bạn sử dụng lệnh git log. Lệnh này giúp bạn theo dõi lịch sử các thay đổi và xem các commit được thực hiện bởi ai và khi nào.

#### **Cú pháp:**

| git log |
| --- |

#### **Ví dụ kết quả:**

| commit h7i8j9kdfc2b345678a1d9cf3a5d6fbc213def45 (HEAD -> main) Author: user123 <user123@example.com> Date: Thu Oct 10 12:34:56 2024 +0700   Thêm tính năng đăng nhập  commit d4e5f6g789h1j23b4cd5e6f7g8h9i01234abcd56 Author: user123 <user123@example.com> Date: Wed Oct 9 10:20:30 2024 +0700   Cập nhật giao diện người dùng  commit a1b2c3d4567e89f0a1b2c3d456789abcd123ef45 Author: user123 <user123@example.com> Date: Tue Oct 8 09:15:20 2024 +0700   Initial commit |
| --- |

**Giải thích kết quả**:

* Mỗi commit được xác định bởi một mã hash duy nhất (ví dụ: h7i8j9kdfc2b345678a1d9cf3a5d6fbc213def45).
* Author: Tác giả thực hiện commit.
* Date: Ngày và giờ commit được tạo ra.
* Thêm tính năng đăng nhập: Thông điệp commit mô tả thay đổi.

### **7. Xóa Repository Cục Bộ**

#### **Mô tả:**

Nếu bạn không còn cần repository cục bộ trên máy tính của mình hoặc muốn xóa hoàn toàn thư mục chứa repository, bạn có thể thực hiện điều này bằng cách xóa thư mục tương ứng. Việc xóa repository cục bộ **không ảnh hưởng** đến repository từ xa (GitHub).

### **Cách 1: Xóa Repository Bằng Giao Diện Người Dùng (File Explorer)**

#### **Bước 1: Mở File Explorer**

* Mở **File Explorer** (hoặc **Windows Explorer**) trên máy tính của bạn.

#### **Bước 2: Tìm đến Thư Mục Chứa Repository**

* Điều hướng đến thư mục mà bạn đã lưu repository cục bộ.

**Ví dụ**: Nếu bạn đã lưu repository cục bộ trong thư mục D:\Projects\MyProject, hãy mở thư mục D:\Projects và tìm thư mục MyProject.

#### **Bước 3: Xóa Toàn Bộ Thư Mục**

* Nhấp chuột phải vào thư mục chứa repository (ví dụ: MyProject) và chọn **Delete** (Xóa).
* Một hộp thoại xác nhận sẽ hiện lên, yêu cầu bạn xác nhận xóa thư mục. Nhấp **Yes** để xóa.

### **Cách 2: Xóa Repository Bằng Dòng Lệnh Trong Git Bash hoặc Command Prompt**

#### **Bước 1: Mở Git Bash hoặc Command Prompt**

* Mở **Git Bash** (hoặc **Command Prompt**) trên máy tính của bạn.

#### **Bước 2: Sử Dụng Lệnh rm -rf Để Xóa Thư Mục**

Sử dụng lệnh rm -rf để xóa toàn bộ thư mục chứa repository và tất cả các file bên trong thư mục đó.

#### **Cú pháp:**

| rm -rf đường\_dẫn\_tới\_thư\_mục\_repository |
| --- |

#### **Ví dụ:**

Nếu repository của bạn nằm trong thư mục D:\Projects\MyProject, bạn sử dụng lệnh sau:

| rm -rf /d/Projects/MyProject |
| --- |

**Giải thích**:

* rm: Lệnh xóa file hoặc thư mục.
* -rf: Hai tham số này được sử dụng để xóa thư mục một cách "đệ quy" (r) và bỏ qua cảnh báo nếu thư mục có chứa file hoặc thư mục con (f).
* /d/Projects/MyProject: Đường dẫn đầy đủ đến thư mục chứa repository cục bộ mà bạn muốn xóa.

#### **Kết quả:**

Lệnh trên sẽ xóa hoàn toàn thư mục MyProject và tất cả nội dung bên trong nó khỏi máy tính của bạn.