**1. Cài đặt Git và GitHub**

**Bước 1: Cài đặt Git**

1. Truy cập trang web chính thức của Git tại <https://git-scm.com/>.
2. Tải và cài đặt Git cho hệ điều hành của bạn (Windows, macOS, hoặc Linux).
3. Sau khi cài đặt, mở **Command Prompt** (Windows) hoặc **Terminal** (macOS/Linux) và gõ lệnh sau để kiểm tra:

git --version

Nếu Git đã được cài đặt thành công, bạn sẽ thấy phiên bản Git hiển thị.

**Bước 2: Cài đặt tài khoản GitHub**

1. Truy cập trang <https://github.com/> và tạo tài khoản GitHub nếu bạn chưa có.
2. Sau khi đăng ký tài khoản, bạn sẽ sử dụng GitHub để lưu trữ và chia sẻ mã nguồn.

**2. Cấu hình Git**

Sau khi cài đặt Git, bạn cần cấu hình thông tin cá nhân để Git biết ai là người thực hiện các thay đổi.

Mở Terminal (hoặc Command Prompt trên Windows) và thực hiện các lệnh sau để cấu hình tên và email:

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

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

**3. Khởi tạo một dự án Git và kết nối với GitHub**

**Bước 1: Tạo thư mục dự án**

Trước tiên, bạn cần tạo một thư mục trong máy tính để chứa dự án của mình.

1. Mở Terminal (hoặc Command Prompt) và di chuyển đến nơi bạn muốn tạo thư mục dự án.
2. Tạo thư mục mới và di chuyển vào thư mục đó:

mkdir my-project

cd my-project

**Bước 2: Khởi tạo Git trong thư mục**

1. Trong thư mục my-project, chạy lệnh sau để khởi tạo Git:

git init

Lệnh này sẽ tạo một thư mục .git trong thư mục của bạn, biến nó thành một repository Git.

**Bước 3: Tạo một repository trên GitHub**

1. Truy cập [GitHub](https://github.com/) và đăng nhập vào tài khoản của bạn.
2. Nhấn vào biểu tượng **+** ở góc trên bên phải và chọn **New repository**.
3. Điền tên repository (ví dụ: my-project), chọn chế độ **Public** hoặc **Private**, và nhấn **Create repository**.

**Bước 4: Kết nối repository local với GitHub**

Sau khi tạo repository trên GitHub, bạn cần kết nối repository local của mình với GitHub.

1. Copy URL của repository GitHub (ví dụ: https://github.com/username/my-project.git).
2. Trong thư mục my-project trên máy tính của bạn, kết nối Git local với repository GitHub:

git remote add origin https://github.com/username/my-project.git

**Bước 5: Thêm và commit file vào repository**

1. Tạo một file trong thư mục my-project (ví dụ: tạo file index.html hoặc bất kỳ file nào bạn muốn).
2. Thêm file vào Git:

git add index.html

Lệnh git add thêm file vào "staging area", chuẩn bị để commit.

1. Commit các thay đổi:

git commit -m "Initial commit"

Lệnh git commit -m lưu lại các thay đổi với một thông điệp mô tả.

**Bước 6: Đẩy mã nguồn lên GitHub**

1. Đẩy các thay đổi lên repository GitHub:

git push -u origin master

* + Lệnh git push gửi các thay đổi từ repository local lên GitHub.
  + -u đảm bảo bạn không cần phải chỉ định lại origin master trong các lần push sau.

1. Sau khi thực hiện lệnh này, bạn sẽ thấy các file của mình đã được tải lên repository trên GitHub.

**4. Các thao tác Git cơ bản**

**Bước 1: Kiểm tra trạng thái dự án**

Để kiểm tra trạng thái của các file trong dự án (có thay đổi gì chưa), bạn sử dụng lệnh:

git status

**Bước 2: Thêm và commit thay đổi**

1. Sau khi chỉnh sửa hoặc thêm file, sử dụng git add để thêm các file thay đổi:

git add .

Lệnh trên sẽ thêm tất cả các thay đổi trong thư mục.

1. Commit thay đổi với một thông điệp mô tả:

git commit -m "Mô tả thay đổi"

**Bước 3: Đẩy thay đổi lên GitHub**

Để đồng bộ các thay đổi từ local lên GitHub, bạn sử dụng lệnh git push:

git push

**Bước 4: Lấy các thay đổi từ GitHub về máy tính (Pull)**

Để đồng bộ các thay đổi từ repository trên GitHub về máy tính của bạn, sử dụng lệnh git pull:

git pull origin master

**Bước 5: Kiểm tra lịch sử commit**

Để xem lịch sử commit của dự án, sử dụng lệnh:

git log

**5. Các thao tác Git nâng cao**

**Bước 1: Tạo nhánh (Branch) mới**

Để tạo một nhánh mới và chuyển sang nhánh đó:

git checkout -b my-branch

Điều này sẽ tạo một nhánh mới tên là my-branch và chuyển bạn sang nhánh đó.

**Bước 2: Chuyển đổi giữa các nhánh**

Để chuyển sang một nhánh khác:

git checkout my-branch

**Bước 3: Gộp (Merge) nhánh**

Khi bạn hoàn thành công việc trên một nhánh, bạn có thể gộp các thay đổi vào nhánh chính (thường là master hoặc main):

1. Đầu tiên, chuyển sang nhánh chính (master hoặc main):

git checkout master

1. Gộp nhánh my-branch vào nhánh chính:

git merge my-branch

**Bước 4: Xử lý xung đột (Conflict)**

Nếu có xung đột khi gộp nhánh (merge), Git sẽ yêu cầu bạn giải quyết vấn đề. Bạn cần mở các file bị xung đột, sửa lại nội dung, sau đó commit lại.

**6. Đồng bộ giữa Local và GitHub**

**Bước 1: Đẩy thay đổi lên GitHub**

Để đẩy mọi thay đổi từ máy tính lên GitHub:

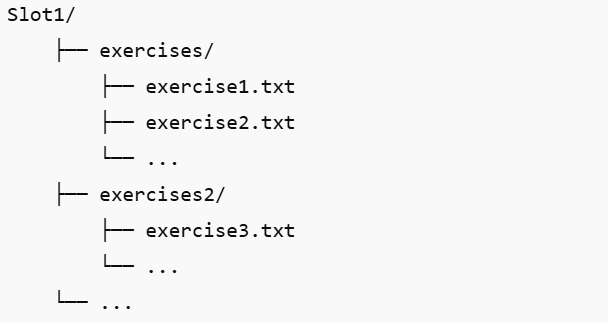
git push origin master

**Bước 2: Lấy thay đổi từ GitHub về máy tính**

Nếu có thay đổi trên GitHub mà bạn chưa có trong máy tính:

git pull origin master

Ví dụ:



Dưới đây là phiên bản chỉnh sửa của hướng dẫn trước, áp dụng phương pháp phân nhánh trong Git để làm việc với các nhánh **features**, **develop**, và **main**. Chúng ta sẽ tạo các nhánh feature để phát triển tính năng, sau đó gộp vào nhánh develop để thử nghiệm, và cuối cùng gộp vào nhánh main khi tất cả các tính năng đã hoàn thiện và sẵn sàng cho môi trường sản xuất.

**1. Tạo repository trên GitHub**

1. **Tạo repository mới trên GitHub**:
   * Truy cập <https://github.com> và đăng nhập vào tài khoản của bạn.
   * Nhấn vào biểu tượng **+** ở góc trên bên phải và chọn **New repository**.
   * Đặt tên repository, ví dụ Slot1, và chọn chế độ **Public** hoặc **Private**.
   * Nhấn **Create repository**.
2. **Lưu ý**: Khi tạo repository trên GitHub, GitHub sẽ cung cấp một URL (ví dụ: https://github.com/username/Slot1.git) để bạn có thể liên kết repository này với repository local của bạn.

**2. Cài đặt Git và cấu hình**

1. **Cài đặt Git** (nếu bạn chưa cài):
   * Tải và cài đặt Git từ <https://git-scm.com>.
   * Sau khi cài đặt, mở **Terminal** (hoặc **Command Prompt** trên Windows) và chạy lệnh sau để kiểm tra:

git --version

1. **Cấu hình Git (nếu chưa cấu hình)**:  
   Bạn cần cấu hình tên và email của bạn trong Git:

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

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

**3. Khởi tạo Git trong thư mục Slot1**

1. Mở **Terminal** (hoặc **Command Prompt**) và chuyển đến thư mục Slot1:

cd /đường/dẫn/tới/Slot1

1. **Khởi tạo Git trong thư mục Slot1**:  
   Chạy lệnh sau để khởi tạo một repository Git mới trong thư mục Slot1:

git init

1. **Kiểm tra trạng thái của Git**:  
   Kiểm tra xem Git đã nhận diện thư mục này là một repository chưa:

git status

**4. Tạo nhánh main, develop và feature**

**Bước 1: Tạo và cấu hình các nhánh**

1. **Tạo nhánh main** (nếu chưa có):  
   Nhánh main thường đã có sẵn khi bạn khởi tạo một repository Git mới, nhưng nếu bạn muốn tạo lại nó, bạn có thể thực hiện lệnh:

git checkout -b main

1. **Tạo nhánh develop từ main**:  
   Khi bạn đã ở trên nhánh main, tạo nhánh develop:

git checkout -b develop

Lúc này, bạn đã có hai nhánh:

* + main: Nhánh chính, ổn định.
  + develop: Nhánh phát triển.

1. **Tạo nhánh feature/feature-1**:  
   Nhánh feature/feature-1 là nơi bạn phát triển các tính năng. Giả sử bạn muốn tạo một tính năng mới:

git checkout -b feature/feature-1

**Bước 2: Thêm file và commit trên nhánh feature**

Giả sử bạn có một số file trong thư mục Slot1, bạn sẽ làm việc trên nhánh feature/feature-1.

1. **Thêm các file vào Git**:  
   Sau khi tạo các file trong thư mục Slot1 (ví dụ: exercises/exercise1.txt, exercises/exercise2.txt), bạn cần thêm các file này vào Git:

git add .

1. **Commit các thay đổi**:  
   Sau khi thêm các file vào staging area, bạn cần commit chúng:

git commit -m "Add exercises to feature-1"

**5. Gộp nhánh feature vào develop**

Khi tính năng trên nhánh feature/feature-1 đã hoàn thành và được kiểm tra, bạn sẽ chuyển sang nhánh develop và gộp các thay đổi từ nhánh feature/feature-1 vào.

1. **Chuyển sang nhánh develop**:

git checkout develop

1. **Gộp nhánh feature/feature-1 vào develop**:

git merge feature/feature-1

1. **Xử lý xung đột nếu có**:  
   Trong trường hợp có xung đột, Git sẽ thông báo bạn cần phải giải quyết xung đột trước khi tiếp tục. Bạn có thể kiểm tra trạng thái xung đột bằng lệnh:

git status

Sau khi giải quyết xung đột, hãy thêm lại các file đã sửa và commit:

git add <file\_xung\_dot>

git commit -m "Fix merge conflict between feature/feature-1 and develop"

**6. Đẩy thay đổi lên GitHub**

1. **Đẩy nhánh develop lên GitHub**:  
   Sau khi hoàn thành việc gộp nhánh feature vào nhánh develop, bạn cần đẩy nhánh develop lên GitHub:

git push origin develop

1. **Kiểm tra trên GitHub**:  
   Truy cập repository GitHub của bạn và kiểm tra xem các thay đổi đã được đồng bộ lên GitHub chưa.

**7. Gộp nhánh develop vào main khi hoàn thành**

Khi tất cả tính năng đã được phát triển và thử nghiệm trong nhánh develop, bạn sẽ gộp nhánh develop vào nhánh main để chuẩn bị triển khai sản phẩm.

1. **Chuyển sang nhánh main**:

git checkout main

1. **Gộp nhánh develop vào main**:

git merge develop

1. **Đẩy nhánh main lên GitHub**:  
   Sau khi gộp nhánh develop vào main, đẩy nhánh main lên GitHub:

git push origin main

**8. Dọn dẹp nhánh**

Sau khi hoàn tất việc gộp nhánh và đồng bộ lên GitHub, bạn có thể xóa nhánh feature và develop nếu không cần thiết nữa.

1. **Xóa nhánh feature/feature-1 trên local**:

git branch -d feature/feature-1

1. **Xóa nhánh feature/feature-1 trên GitHub**:

git push origin --delete feature/feature-1

1. **Xóa nhánh develop nếu không cần nữa**:

git branch -d develop

git push origin --delete develop

**Tóm tắt quy trình**

1. **Tạo các nhánh**:
   * main: Nhánh chính, ổn định.
   * develop: Nhánh phát triển chung.
   * feature: Nhánh tính năng riêng biệt cho từng tính năng.
2. **Làm việc với các nhánh**:
   * Tạo và phát triển tính năng trên nhánh feature.
   * Gộp nhánh feature vào nhánh develop.
   * Giải quyết xung đột khi cần.
   * Cuối cùng, gộp develop vào main khi tính năng đã hoàn thành.
3. **Đẩy và đồng bộ với GitHub**:
   * Sau khi gộp nhánh, đẩy các thay đổi lên GitHub để đồng bộ hóa.