Pro Git (Second Edition) – Take Note

# Chương 1: Getting Started (Bắt đầu với Git)

Mục tiêu:  
Giúp bạn làm quen với Git, hiểu về khái niệm hệ thống quản lý phiên bản (VCS), cài đặt Git và thực hiện thao tác cơ bản đầu tiên.  
  
Nội dung chính:  
1. Git là gì?  
 - Là hệ thống quản lý phiên bản phân tán (Distributed Version Control System).  
 - Mọi người trong nhóm đều có bản sao đầy đủ lịch sử thay đổi.  
 - Khác với SVN hay CVS (hệ thống tập trung).  
  
2. Lý do chọn Git:  
 - Nhanh, hiệu quả với dự án lớn.  
 - Tối ưu khi làm việc nhóm.  
 - Bảo vệ dữ liệu tốt hơn.  
  
3. Cài đặt Git:  
 - Windows: dùng installer tại git-scm.com.  
 - Linux/macOS: dùng apt, brew, hoặc yum.  
  
4. Cấu hình Git lần đầu:  
 git config --global user.name "Tên của bạn"  
 git config --global user.email "email@example.com"  
 git config --global core.editor nano  
  
5. Trợ giúp khi dùng Git:  
 - git help <lệnh>  
 - git <lệnh> --help

# Chương 2: Git Basics (Các thao tác cơ bản)

Mục tiêu:  
Hiểu cách tạo dự án Git, thêm/sửa file, commit, hoàn tác và làm việc với kho chứa từ xa (remote).  
  
Nội dung chính:  
1. Tạo dự án Git mới:  
 git init  
  
2. Sao chép (clone) một dự án:  
 git clone https://github.com/tai-khoan/ten-repo.git  
  
3. Theo dõi thay đổi:  
 git status  
 git add <tên-file>  
 git commit -m "Ghi chú thay đổi"  
  
4. Xem lịch sử:  
 git log  
 git log --oneline  
  
5. Di chuyển và xóa file:  
 git mv tenfile1 tenfile2  
 git rm tenfile  
  
6. Hoàn tác thay đổi:  
 git checkout -- <file>  
 git reset HEAD <file>  
 git revert <mã-commit>  
  
7. Làm việc với Remote:  
 git remote add origin <url>  
 git fetch, git pull, git push origin main  
  
8. Tag – gắn nhãn phiên bản:  
 git tag v1.0.0  
 git push origin v1.0.0

# Chương 3: Git Branching (Làm việc với nhánh)

Mục tiêu:  
Hiểu cách tạo, quản lý, hợp nhất nhánh – điểm mạnh nhất của Git so với các hệ thống khác.  
  
Nội dung chính:  
1. Khái niệm về nhánh:  
 - Nhánh (branch) là dòng phát triển độc lập.  
 - main là nhánh chính mặc định.  
  
2. Tạo và chuyển đổi nhánh:  
 git branch ten-nhanh  
 git checkout ten-nhanh  
 git checkout -b nhanh-moi  
  
3. Xem danh sách nhánh:  
 git branch  
 git branch -r  
  
4. Hợp nhất nhánh (merge):  
 git checkout main  
 git merge nhanh-phu  
  
5. Rebase (tái căn chỉnh nhánh):  
 git rebase main  
  
6. Xóa hoặc đổi tên nhánh:  
 git branch -d nhanh-phu  
 git branch -m nhanh-cu nhanh-moi  
  
7. Nhánh từ xa (remote branches):  
 git fetch  
 git checkout -b nhanh-moi origin/nhanh-moi

## ****Chương 4: Git trên Máy chủ (Git on the Server)****

### 1. Tạo Repository Bare trên Máy chủ

* **Bare repository** là repo **không chứa thư mục làm việc (working directory)**.
* Dùng lệnh sau để tạo một bare repo:
* mkdir -p /opt/git/project.git
* cd /opt/git/project.git
* git init --bare
* Bare repo được dùng như trung tâm để **push/pull code**, không được sửa trực tiếp trong đó.

### 2. Truy cập Git Server qua SSH

#### Bước 1: Tạo user riêng cho Git

sudo adduser git

#### Bước 2: Cấp quyền SSH cho user git

* Developer gửi **public key** cho admin.
* Admin thêm key vào /home/git/.ssh/authorized\_keys

#### Bước 3: Clone từ client

git clone git@gitserver:/opt/git/project.git

➡️ Ưu điểm: Bảo mật, linh hoạt, không cần phần mềm phụ trợ.

### 3. Chạy Git Daemon (qua giao thức git://)

* Dành cho việc clone nhanh, không yêu cầu login.
* Cấu hình:

git daemon --reuseaddr --base-path=/opt/git/ /opt/git/

* Port mặc định: **9418**
* Cần bật git-daemon-export-ok trong repo:

touch /opt/git/project.git/git-daemon-export-ok

➡️ Ưu điểm: Nhanh. Nhược điểm: Không bảo mật (vì không xác thực).

### 4. Giao thức HTTP(S)

* Cài Apache hoặc Nginx hỗ trợ Smart HTTP (được Git hỗ trợ chính thức).
* Yêu cầu cài đặt:
  + CGI hoặc FastCGI
  + git-http-backend từ git-core
* Ví dụ cấu hình Apache:

ScriptAlias /git/ /usr/lib/git-core/git-http-backend/

SetEnv GIT\_PROJECT\_ROOT /opt/git

SetEnv GIT\_HTTP\_EXPORT\_ALL

➡️ Ưu điểm: Phù hợp web + có thể dùng xác thực HTTP basic, LDAP, OAuth.

### 5. Tích hợp GitWeb (giao diện web cho Git)

* Giao diện đơn giản để xem commit, branch, diff…
* Cài trên Debian/Ubuntu:

sudo apt-get install gitweb

* Cần cấu hình Apache hoặc CGI để chạy gitweb.cgi.

### 6. Quản lý nhiều người dùng

#### Cách 1: Shared SSH User (git)

* Dễ dùng nhưng khó kiểm soát từng người.

#### Cách 2: SSH với nhiều tài khoản

* Mỗi người có 1 user riêng.
* Dùng git-shell làm shell mặc định:

chsh -s $(which git-shell) username

→ Chặn user không truy cập shell bình thường.

#### Cách 3: Sử dụng Gitolite hoặc Gitosis

* Hệ thống quản lý user, quyền truy cập (read/write), branch, tag dễ dàng hơn.

### 7. So sánh giao thức

| **Giao thức** | **Bảo mật** | **Yêu cầu setup** | **Dễ dùng** | **Tốc độ** |
| --- | --- | --- | --- | --- |
| SSH | ✅ | Trung bình | Trung bình | Tốt |
| Git Daemon | ❌ | Dễ | Dễ | Rất nhanh |
| HTTP/HTTPS | ✅ | Khó hơn | Dễ dùng cho người mới | Trung bình |
| HTTPS + OAuth | ✅✅ | Khó | Hiện đại | Trung bình |

### Tổng kết

* Bạn có thể **tự triển khai Git Server** nếu không muốn dùng dịch vụ bên thứ ba.
* Có nhiều giao thức phù hợp với từng nhu cầu:
  + **SSH** cho nội bộ bảo mật.
  + **HTTP(S)** cho tổ chức lớn, tích hợp web.
  + **Git Daemon** cho tốc độ, không bảo mật.

Chương 5: Git phân tán (Distributed Git)

Git là một hệ thống quản lý mã nguồn phân tán (Distributed Version Control System – DVCS), nghĩa là mỗi lập trình viên có một bản sao đầy đủ của repository – bao gồm tất cả commit, nhánh, tag, v.v.

1. Các mô hình làm việc phân tán (Distributed Workflows)

a. Centralized Workflow (Kiểu tập trung)

Mọi người cùng push/pull lên một repository trung tâm (giống SVN).

Thường dùng khi team nhỏ, hoặc chuyển từ SVN sang.

Ví dụ:

bash

Copy code

git clone git@server:project.git

git commit

git push origin main

b. Feature Branch Workflow

Mỗi tính năng (feature) có một nhánh riêng.

Merge vào main qua Pull Request (hoặc Merge Request).

Hỗ trợ kiểm tra code, CI/CD, rollback dễ hơn.

Quy trình:

bash

Copy code

git checkout -b new-feature

git push origin new-feature

c. Gitflow Workflow (mô hình chuyên nghiệp)

Dùng nhánh: main, develop, feature/\*, release/\*, hotfix/\*.

Rất phổ biến trong các dự án lớn (Jira, CI/CD).

rust

Copy code

main -> production

develop -> development

feature/\* -> tính năng mới

release/\* -> chuẩn bị phát hành

hotfix/\* -> sửa lỗi gấp

d. Forking Workflow (mô hình GitHub)

Mỗi người fork repo chính về tài khoản riêng.

Push vào repo riêng → tạo Pull Request để merge vào repo chính.

Mô hình phù hợp cộng đồng mã nguồn mở (Open Source).

2. Remote Repository (Kho từ xa)

Mỗi remote là một bản sao của repo ở đâu đó (GitHub, GitLab, máy chủ).

Mặc định tên remote là origin.

bash

Copy code

git remote add myfriend git@github.com:friend/project.git

git fetch myfriend

git checkout -b review-branch myfriend/feature-abc

3. Theo dõi và đóng góp vào dự án người khác

Dùng git remote add để kết nối với repo của người khác.

Dùng fetch để lấy branch, sau đó merge hoặc rebase về repo của bạn.

4. Làm việc với nhiều remote

Ví dụ: bạn làm việc trên origin, nhưng muốn lấy code từ upstream.

bash

Copy code

git remote add upstream git@github.com:original/project.git

git fetch upstream

5. Tạo Pull Request (hoặc Merge Request)

Sau khi push feature lên GitHub/GitLab:

Vào trang web → bấm "New Pull Request"

Người review → chấp nhận/đề nghị sửa → merge

Chương 6: GitHub

GitHub là dịch vụ Git Hosting phổ biến nhất, cho phép bạn lưu trữ, chia sẻ và cộng tác phát triển phần mềm. Ngoài ra còn hỗ trợ:

Issues, Wiki, Pull Request, CI/CD, Action, Project board...

🔧 1. Tạo tài khoản và repo

Đăng ký tại github.com

Tạo repository mới (Public hoặc Private)

Clone về máy:

bash

Copy code

git clone https://github.com/username/project.git

🔐 2. Xác thực GitHub

a. HTTPS + Token

Bắt buộc dùng Personal Access Token (PAT) thay vì mật khẩu.

Tạo tại: Settings > Developer settings > Tokens

b. SSH

bash

Copy code

ssh-keygen -t rsa -b 4096 -C "email@example.com"

Add ~/.ssh/id\_rsa.pub vào GitHub: Settings > SSH and GPG keys

3. Fork và Pull Request

Bấm “Fork” để tạo bản sao về tài khoản bạn.

Clone repo đó → sửa → push

Trên GitHub: “New Pull Request” → chọn nhánh → gửi yêu cầu

4. Quản lý tổ chức và nhóm

Tạo Organization: Quản lý theo công ty/nhóm

Phân quyền: Admin, Write, Read

Tạo Team: Tách nhóm theo chức năng (Dev, QA, Manager…)