**CHƯƠNG I: GIỚI THIỆU**

1. **QUẢN LÝ PHIÊN BẢN SỬ DỤNG GIT**
   1. **Giới thiệu về quản lý phiên bản sử dụng Git**

Quản lý phiên bản là một khía cạnh quan trọng trong quá trình phát triển phần mềm. Git là một trong những công cụ quản lý phiên bản phổ biến nhất hiện nay, được sử dụng rộng rãi trong cộng đồng phát triển phần mềm. Git được phát triển bởi Linus Torvalds vào năm 2005 như là một công cụ quản lý mã nguồn cho dự án Linux. Sau đó, nó đã trở thành một công cụ quản lý phiên bản độc lập và có thể được sử dụng cho các dự án phần mềm khác.

* 1. **Tầm quan trọng của Git trong quản lý phiên bản**

Git giúp cho quá trình quản lý phiên bản trở nên dễ dàng và hiệu quả hơn. Nó cho phép các nhà phát triển phần mềm theo dõi, quản lý và kiểm soát các thay đổi của mã nguồn một cách chặt chẽ, đồng thời cho phép các phiên bản khác nhau của sản phẩm tồn tại đồng thời.

Git là một trong những Hệ thống Quản lý Phiên bản Phân tán, vốn được phát triển nhằm quản lý mã nguồn (source code) hữu hiệu của Linux.

Trên Git, có thể lưu trạng thái của file khi có nhu cầu dưới dạng lịch sử cập nhật. Vì thế, có thể đưa file đã chỉnh sửa một lần về trạng thái cũ hay có thể hiển thị sự khác biệt ở nơi chỉnh sửa.

Thêm nữa, khi định ghi đè (overwrite) lên file mới nhất đã chỉnh sửa của người khác bằng file đã chỉnh sửa dựa trên file cũ, thì khi đăng (upload) lên server sẽ hiện ra cảnh cáo. Vì thế, sẽ không xảy ra thất bại về việc đã ghi đè lên nội dung chỉnh sửa của người khác mà không hề hay biết.

1. **GIT**
   1. **Sơ lược về Git**

Git là một đại diện tiêu biểu của hệ thống quản lý phiên bản phân tán DVCS mã nguồn mở, một trong những dạng hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay.

Git có những đặc điểm sau:

* Có hướng tiếp cận mới so với các hệ thống Source Control khác như SVN hay CVS truyền thống.
* Có nhiệm vụ theo dõi những thay đổi, chỉnh sửa trong Source Code của người dùng vào mội thời điểm và đồng bộ những Source Code do họ chỉnh sửa lên Server cùng đồng nghiệp.

Ngoài ra, Git còn có thể chạy trên nhiều hệ điều hành khác nhau như Linux, Windows, MacOS,…

* 1. **Lịch sử phát triển Git**

Git được tạo ra bởi Linus Torvalds vào năm 2005. Lúc đó, dự án Linux đang sử dụng một hệ thống quản lý phiên bản tên là BitKeeper để quản lý mã nguồn của mình. Tuy nhiên, khi quyền sử dụng BitKeeper của dự án Linux bị thu hồi, Linus Torvalds quyết định tạo ra một hệ thống quản lý phiên bản của riêng mình.

Ban đầu, Git được sử dụng cho việc phát triển và quản lý mã nguồn của dự án Linux. Tuy nhiên, sau đó Git đã được sử dụng rộng rãi trong cộng đồng phát triển phần mềm và trở thành một trong những hệ thống quản lý phiên bản phổ biến nhất trên thế giới.

Hiện nay, Git được sử dụng trong các dự án phát triển phần mềm lớn như Android, jQuery, Ruby on Rails, và nhiều dự án phần mềm mã nguồn mở khác trên toàn cầu. Git cũng được tích hợp vào nhiều dịch vụ lưu trữ mã nguồn như GitHub, GitLab, Bitbucket, để giúp các nhà phát triển làm việc và quản lý mã nguồn một cách dễ dàng.

* 1. **Ưu điểm và nhược điểm của Git**

Dựa vào những đặc tính và sự phổ biến của Git đối với người dùng, nhất là lập trình viên, ta sẽ đưa ra những lợi ích khi sử dụng Git. Tuy nhiên, một công cụ hay phần mềm nào cũng đều có lợi và có hại. Vì vậy chúng ta sẽ tìm hiểu ưu và nhược điểm của nó.

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| * Hệ thống quản lý phiên bản phân tán (DVCS) với tính năng nhánh và định nghĩa linh hoạt. * Tốc độ xử lý nhanh hơn so với nhiều hệ thống quản lý phiên bản khác. * Khả năng xử lý các xung đột đến từ sự thay đổi dữ liệu giữa các nhánh một cách tốt hơn. * Tính bảo mật cao với các tính năng chống phá hoại, sao chép dự phòng và mã hóa. * Sử dụng dễ dàng và hỗ trợ đa nền tảng. | * Học cú pháp Git ban đầu có thể khó khăn đối với người mới bắt đầu. * Quản lý phiên bản phân tán có thể dẫn đến việc tăng kích thước repository nhanh hơn so với hệ thống quản lý phiên bản tập trung. * Không có tính năng giải quyết xung đột tự động hoàn toàn, yêu cầu sự can thiệp của người dùng. * Không thể xóa các commit trước đó một cách dễ dàng mà không ảnh hưởng đến lịch sử phiên bản. * Có thể gặp phải các vấn đề liên quan đến hiệu suất nếu repository quá lớn hoặc có quá nhiều branch. |