|  |  |  |
| --- | --- | --- |
| **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HCM**  **KHOA CÔNG NGHỆ THÔNG TIN**  🙞❖🙜  A blue and white logo with a book and a square cap  Description automatically generated  **TIỂU LUẬN MÔN HỌC: CÔNG NGHỆ PHẦN MỀM**  **ĐỀ TÀI**  **TÌM HIỂU CÔNG NGHỆ GITHUB**  **NHÓM: 05**    **Thành phố Hồ Chí Minh, 12 tháng 09 năm 2024**  **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HCM**  **KHOA CÔNG NGHỆ THÔNG TIN**  A blue and white logo with a book and a square cap  Description automatically generated🙞❖🙜  **ĐỀ TÀI**  **TÌM HIỂU CÔNG NGHỆ GITHUB**   |  |  | | --- | --- | | Nhóm thực hiện: Nhóm 05  Thành viên:  Trương Văn Thạch Bảo  Nguyễn Thế Duy  Lê Hoàng Hải  Nguyễn Kim Phụng  Đường Xuân Thành Tài | Giảng viên hương dẫn:  Mạnh Thiên Lý |   **Thành phố Hồ Chí Minh, 12 tháng 9 năm 2024** |

**PHÂN CÔNG CÔNG VIỆC THÀNH VIÊN NHÓM**

* **Phần làm chung:**
* Tìm hiểu về github
* **Bảng phân công:**

|  |  |  |
| --- | --- | --- |
| **Tên Thành Viên** | **Công việc** | **Hoàn thành** |
| Trương Văn Thạch Bảo | So sánh Github và kết luận | 100% |
| Nguyễn Thế Duy | Github hỗ trợ cộng đồng lập trình viên | 100% |
| Lê Hoàng Hải | Các tính năng của github | 100% |
| Nguyễn Kim Phụng | Giới thiệu, Kiến trúc và công nghệ nền tảng Github | 100% |
| Đường Xuân Thành Tài | Các ứng dụng thực tiễn của github | 100% |

**LỜI CAM ĐOAN**

Xin cam đoan đây là công trình nghiên cứu của riêng nhóm em. Các số liệu, kết quả nêu trong Tiểu luận / Đồ án là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Nhóm em xin cam đoan rằng mọi sự giúp đỡ cho việc thực hiện Tiểu luận / Đồ án này đã được cảm ơn và các thông tin trích dẫn trong Tiểu luận / Đồ án đã được chỉ rõ nguồn gốc.

**LỜI CẢM ƠN**

Để hoàn thành bài tiểu luận này, nhóm em xin được gửi lời cảm ơn chân thành và sâu sắc đến giảng viên Mạnh Thiên Lý, người đã tận tình hướng dẫn nhóm trong suốt quá trình làm đề tài. Nhờ có sự khuyên bảo nhiệt tình cùng những kiến thức quý báu thầy truyền dạy mà nhóm em mới có thể hoàn thành bài tiểu luận một cách tốt đẹp và đúng tiến độ.

Tuy nhiên với kinh nghiệm còn hạn chế, bài tiểu luận sẽ không thể tránh khỏi những thiếu sót và sai lầm, nhóm em hi vọng sẽ nhận được những phản hồi và góp ý để ngày càng hoàn thiện hơn. Từ đó có điều kiện bổ sung, nâng cao ý thức để có thể phục vụ tốt hơn công tác thực tế sau này

## Mục lục

[I. Phần Mở Đầu: 1](#_Toc7508)

[1.1. Lý do chọn đề tài: 1](#_Toc16802)

[1.2. Tổng quan về đề tài: 1](#_Toc15315)

[1.3. Mục đích nghiên cứu: 1](#_Toc2015)

[1.4. Đối tượng và phạm vi nghiên cứu: 2](#_Toc25615)

[II. Giới thiệu về Github 2](#_Toc32241)

[2.1. Khái niệm về GitHub 2](#_Toc5802)

[2.2. Vai trò của GitHub trong phát triển phần mềm 3](#_Toc1109)

[III. Kiến trúc và công nghệ nền tảng của github 4](#_Toc4866)

[3.1. Git - Công cụ quản lý mã nguồn 4](#_Toc8801)

[3.2. GitHub và Git 5](#_Toc15039)

[IV. Các tính năng chính của Github 5](#_Toc29727)

[4.1. Repository (Kho lưu trữ) 5](#_Toc32449)

[4.2. Branch và Pull Request 6](#_Toc16388)

[4.3. GitHub Issues và GitHub Projects 7](#_Toc2164)

[4.4. GitHub Actions 7](#_Toc15212)

[4.5. GitHub Pages 8](#_Toc10968)

[V. Github hỗ trợ cộng đồng lập trình viên: 9](#_Toc31755)

[5.1. GitHub và Open Source 9](#_Toc16576)

[5.1.1. GitHub như một nền tảng hỗ trợ các dự án mã nguồn mở 9](#_Toc17829)

[5.1.2. Các dự án nổi bật trên GitHub 10](#_Toc2583)

[5.2. GitHub Community và GitHub Sponsors 10](#_Toc7889)

[5.2.1. Cộng đồng GitHub và cách thức tương tác giữa các lập trình viên 10](#_Toc26661)

[5.2.2. Chương trình GitHub Sponsors hỗ trợ các lập trình viên mã nguồn mở 11](#_Toc11413)

[5.3. Hợp tác và làm việc nhóm trên GitHub 11](#_Toc6445)

[5.3.1 Các công cụ hỗ trợ hợp tác (code review, project management) 11](#_Toc5147)

[5.3.2 Quản lý phiên bản và các lợi ích trong cộng tác dự án 12](#_Toc21986)

[VI. Các ứng dụng thực tiễn của Github 13](#_Toc23245)

[6.1. Tại sao nên chọn GitHub cho doanh nghiệp? 13](#_Toc1913)

[6.1.1. Lý do doanh nghiệp sử dụng GitHub 13](#_Toc16542)

[6.1.2. Cách doanh nghiệp sử dụng GitHub 13](#_Toc17257)

[6.2. Githud trong học tập 14](#_Toc11591)

[6.3. Tích hợp GitHub với các công cụ CI/CD, IDE và các công cụ quản lý dự án. 14](#_Toc25116)

[6.3.1. Tích hợp với các công cụ CI/CD: 14](#_Toc20673)

[6.3.2. Tích hợp với IDE: 15](#_Toc28691)

[6.3.3. Tích hợp với các công cụ quản lý dự án: 15](#_Toc22601)

[VII. So sánh Github với các nền tảng tương tự: 15](#_Toc30035)

[7.1. So sánh GitHub với GitLab 15](#_Toc25765)

[7.2. So sánh GitHub với Bitbucket 15](#_Toc11539)

[7.3. Ưu và nhược điểm của GitHub so với các đối thủ cạnh tranh 16](#_Toc5004)

[VIII. Kết luận 16](#_Toc22532)

[TÀI LIỆU THAM KHẢO 17](#_Toc15714)

[1. Github là gì: 18](#_Toc26025)

[2. Kiến trúc và công nghệ github: 18](#_Toc20894)

[3. Các tính năng của github: 18](#_Toc21910)

[4. Github hỗ trợ cộng đồng lập trình viên: 18](#_Toc20372)

[5. Các ứng dụng thực tiễn của github 18](#_Toc30616)

# Phần Mở Đầu:

## Lý do chọn đề tài:

* Trong thời đại công nghệ phát triển như hiện nay, các công cụ hỗ trợ phát triển phần mềm đã trở thành một phần không thể thiếu, đặc biệt là trong việc quản lý mã nguồn và cộng tác làm việc nhóm. GitHub nổi lên như một nền tảng hàng đầu, không chỉ giúp quản lý mã nguồn hiệu quả mà còn tạo điều kiện thuận lợi cho việc chia sẻ và phát triển các dự án mã nguồn mở. Việc tìm hiểu sâu hơn về GitHub sẽ giúp chúng ta hiểu rõ hơn về quy trình phát triển phần mềm hiện đại và cách thức mà các lập trình viên trên khắp thế giới hợp tác với nhau. Do đó, đề tài này được lựa chọn để nghiên cứu vai trò, tính năng và ứng dụng của GitHub trong thực tế.

## Tổng quan về đề tài:

* GitHub là một nền tảng dựa trên hệ thống kiểm soát phiên bản Git, cho phép lưu trữ mã nguồn và cộng tác trong phát triển phần mềm. Với sự ra đời vào năm 2008, GitHub đã nhanh chóng trở thành công cụ không thể thiếu đối với các lập trình viên, doanh nghiệp, và đặc biệt là cộng đồng mã nguồn mở. Ngoài việc lưu trữ mã nguồn, GitHub còn cung cấp nhiều tính năng hữu ích như quản lý dự án, theo dõi lỗi, và tự động hoá quy trình kiểm thử, triển khai. Bài nghiên cứu sẽ tập trung vào phân tích cơ chế hoạt động của GitHub, những tính năng nổi bật và ứng dụng thực tế của nền tảng này trong các dự án phần mềm.

## Mục đích nghiên cứu:

**Mục tiêu chính của nghiên cứu này là:**

* Tìm hiểu sâu hơn về cách GitHub hoạt động và quản lý các dự án phần mềm.
* Phân tích các tính năng chính của GitHub như quản lý kho mã nguồn, Pull Request, Issue Tracking, và CI/CD.
* Khám phá lợi ích mà GitHub mang lại cho các dự án mã nguồn mở và doanh nghiệp.
* Đánh giá những thách thức và hạn chế khi sử dụng GitHub, đặc biệt đối với người mới bắt đầu.

## Đối tượng và phạm vi nghiên cứu:

* **Đối tượng nghiên cứu**: Nền tảng GitHub và các tính năng hỗ trợ quản lý và phát triển dự án phần mềm.
* **Phạm vi nghiên cứu**: Bài nghiên cứu sẽ tập trung vào việc khám phá các tính năng quan trọng của GitHub, các ứng dụng trong thực tế, và tác động của GitHub đối với cộng đồng lập trình viên. Nghiên cứu sẽ không đi sâu vào các hệ thống kiểm soát phiên bản khác như GitLab hay Bitbucket, mà chủ yếu tập trung vào GitHub và hệ thống Git.

# Giới thiệu về Github

## Khái niệm về GitHub

**\*GitHub là gì?**

GitHub là một nền tảng trực tuyến nổi bật cho việc lưu trữ và quản lý mã nguồn, dựa trên hệ thống quản lý mã nguồn phân tán Git. Đây là một công cụ quan trọng dành cho các nhà phát triển phần mềm, giúp họ lưu trữ, quản lý, và theo dõi các phiên bản mã nguồn của các dự án phần mềm một cách hiệu quả. Ngoài việc cung cấp không gian lưu trữ mã nguồn, GitHub còn tích hợp nhiều tính năng khác như theo dõi lỗi, quản lý dự án, đánh giá mã nguồn (code review), và cộng tác trong nhóm. GitHub không chỉ là nơi để các nhà phát triển lưu trữ mã nguồn mà còn là môi trường cộng tác mạnh mẽ cho hàng triệu dự án phần mềm trên khắp thế giới, từ các dự án mã nguồn mở cho đến các dự án thương mại.

**\*Lịch sử ra đời và phát triển của GitHub**

GitHub được ra mắt lần đầu tiên vào năm 2008 bởi Tom Preston-Werner, Chris Wanstrath, PJ Hyett và Scott Chacon, với mục tiêu cung cấp một nền tảng giúp các nhà phát triển phần mềm dễ dàng sử dụng Git, một hệ thống quản lý mã nguồn phân tán mạnh mẽ. Git, do Linus Torvalds – người sáng lập Linux – phát triển, đã trở thành tiêu chuẩn trong việc quản lý mã nguồn, nhưng vào thời điểm đó, việc sử dụng Git trực tiếp không phải là điều đơn giản đối với mọi người. GitHub đã giải quyết vấn đề này bằng cách cung cấp giao diện người dùng thân thiện, dễ sử dụng và tích hợp nhiều tính năng giúp quản lý các dự án phần mềm trở nên dễ dàng hơn.

Kể từ khi ra mắt, GitHub đã không ngừng phát triển và trở thành một trong những nền tảng quản lý mã nguồn phổ biến nhất trên thế giới. Vào năm 2018, GitHub đã được Microsoft mua lại với giá 7,5 tỷ USD, một sự kiện gây chú ý lớn trong cộng đồng công nghệ. Sau khi về tay Microsoft, GitHub tiếp tục phát triển và tích hợp thêm nhiều tính năng và công cụ mạnh mẽ để hỗ trợ nhà phát triển trong việc quản lý dự án và phát triển phần mềm.

## Vai trò của GitHub trong phát triển phần mềm

Trong lĩnh vực phát triển phần mềm, GitHub đóng vai trò cực kỳ quan trọng. Nó không chỉ là nơi lưu trữ mã nguồn mà còn là một công cụ mạnh mẽ giúp các nhóm phát triển phần mềm hợp tác và quản lý dự án hiệu quả hơn. Dưới đây là một số vai trò chính của GitHub:

**Quản lý mã nguồn:**GitHub giúp các nhà phát triển lưu trữ và quản lý mã nguồn một cách dễ dàng. Họ có thể tạo ra các kho lưu trữ (repositories), lưu trữ mã nguồn của từng phiên bản, và theo dõi sự thay đổi của mã theo thời gian. Nhờ việc sử dụng Git, GitHub giúp các nhà phát triển dễ dàng theo dõi sự thay đổi của từng dòng mã, từ đó giúp đảm bảo tính toàn vẹn và nhất quán của dự án.

**Hợp tác và chia sẻ:**GitHub tạo ra môi trường cộng tác trực tuyến, nơi các nhà phát triển từ khắp nơi trên thế giới có thể làm việc cùng nhau trên cùng một dự án. Tính năng "pull request" giúp các thành viên trong nhóm đề xuất thay đổi mã nguồn, đồng thời các thành viên khác có thể kiểm tra và đánh giá mã trước khi tích hợp vào dự án chính.

**Quản lý dự án:** GitHub cung cấp các công cụ giúp theo dõi tiến trình của dự án phần mềm, quản lý các yêu cầu công việc và lỗi (issues), và phân chia công việc cho các thành viên trong nhóm. Các tính năng như bảng Kanban và timeline giúp các nhà quản lý dự án dễ dàng theo dõi tiến độ và điều phối công việc.

**Cộng đồng mã nguồn mở:** GitHub là nơi mà các dự án mã nguồn mở có thể phát triển và thu hút cộng đồng tham gia. Các nhà phát triển trên toàn thế giới có thể xem, sao chép (fork), và đóng góp cho các dự án mã nguồn mở, từ đó thúc đẩy sự sáng tạo và phát triển công nghệ mới.

# Kiến trúc và công nghệ nền tảng của github

## Git - Công cụ quản lý mã nguồn

**\*Git là gì?**

Git là một hệ thống quản lý mã nguồn phân tán (Distributed Version Control System - DVCS) do Linus Torvalds, người sáng lập hệ điều hành Linux, phát triển vào năm 2005. Nó được thiết kế để giúp các nhà phát triển phần mềm quản lý các phiên bản mã nguồn của dự án một cách hiệu quả, đặc biệt trong các dự án có nhiều nhà phát triển tham gia. Git cho phép theo dõi các thay đổi của mã nguồn theo thời gian, giúp các nhà phát triển có thể quay lại các phiên bản trước, so sánh sự khác biệt giữa các phiên bản và hợp nhất (merge) các thay đổi từ nhiều nhánh (branch) khác nhau.

**\*Sự khác biệt giữa Git và GitHub**

Mặc dù GitHub và Git có mối liên hệ mật thiết với nhau, nhưng chúng không hoàn toàn giống nhau. Git là công cụ quản lý mã nguồn cục bộ (local) trên máy tính của từng nhà phát triển, trong khi GitHub là dịch vụ trực tuyến giúp lưu trữ các kho mã nguồn Git và cung cấp môi trường cộng tác giữa các nhà phát triển. Nói cách khác, Git là công cụ giúp quản lý mã nguồn trên máy tính, còn GitHub là nền tảng để chia sẻ và hợp tác các kho mã nguồn đó trên internet.

**\*Các chức năng cơ bản của Git**

* **Commit**: Ghi lại trạng thái hiện tại của mã nguồn vào lịch sử, tạo ra một điểm mốc mà người dùng có thể quay lại nếu cần.
* **Branch**: Tạo nhánh mới để phát triển tính năng riêng mà không ảnh hưởng đến nhánh chính của dự án.
* **Merge**: Kết hợp các thay đổi từ các nhánh khác nhau vào một nhánh chung.
* **Clone**:Sao chép một kho lưu trữ Git từ máy chủ về máy cục bộ để làm việc.

## GitHub và Git

GitHub sử dụng Git làm công cụ chính để quản lý mã nguồn. Khi các nhà phát triển tạo ra một kho lưu trữ (repository) trên GitHub, họ thực chất đang sử dụng Git để quản lý các phiên bản mã nguồn của dự án đó. GitHub cung cấp giao diện người dùng web để thực hiện các thao tác cơ bản của Git như commit, branch, merge, và pull request. Ngoài ra, GitHub còn cung cấp các tính năng nâng cao để quản lý dự án, như tích hợp với các công cụ CI/CD, quản lý tài liệu và giao tiếp với các thành viên trong nhóm.

GitHub còn cho phép tích hợp Git với nhiều dịch vụ khác. Ví dụ, các công cụ kiểm tra mã tự động (CI tools) như Travis CI hay CircleCI có thể tích hợp với GitHub để kiểm tra mã nguồn ngay khi có một pull request mới. GitHub cũng tích hợp với các công cụ theo dõi công việc như Jira hoặc Trello để quản lý dự án dễ dàng hơn.

Trong tổng thể, GitHub đã trở thành một công cụ không thể thiếu trong quá trình phát triển phần mềm hiện đại, giúp các nhà phát triển hợp tác dễ dàng và quản lý mã nguồn hiệu quả hơn.

# Các tính năng chính của Github

## Repository (Kho lưu trữ)

**\*Cách tạo và quản lý repository:**

- Tạo repository: Để tạo một mới repository, bạn có thể nhấp vào nút `+` ở góc trên bên phải của trang chủ và chọn `New repository`. Điền tên repository, mô tả (tùy chọn), chọn giữa public hoặc private, và cấu hình các tùy chọn khác như initial commit, .gitignore, và license.

- Quản lý repository: Bạn có thể quản lý các file trong repository bằng cách sử dụng giao diện web hoặc qua command line (git). Có thể thêm, sửa đổi, xóa file, tạo branch mới, và merge branch.

**\*Khác biệt giữa public và private repository:**

- Public: Mọi người có thể xem và clone repository. Thích hợp cho dự án mã nguồn mở hoặc cho mục đích chia sẻ kiến thức.

- Private: Chỉ những người được mời có thể xem và contribute vào repository. Thích hợp cho dự án nhóm, dự án công ty, hoặc bất kỳ dự án nào yêu cầu bảo mật.

## Branch và Pull Request

**\*Tạo branch và quản lý các nhánh của dự án:**

- Tạo branch: Sử dụng lệnh `git branch <branch-name>` hoặc qua giao diện web. Branch giúp phát triển các tính năng mới mà không ảnh hưởng đến branch chính (master/main).

- Quản lý branch: Có thể xem, sửa đổi, xóa branch. Sử dụng `git checkout <branch-name>` để chuyển sang một branch cụ thể.

Pull Request: Ý nghĩa và quy trình review mã nguồn:

- Ý nghĩa: Pull Request (PR) là một yêu cầu để merge các thay đổi từ một branch vào một branch khác, thường là từ branch phát triển vào branch chính.

- Quy trình review: Người gửi PR sẽ mời các thành viên khác trong nhóm review mã nguồn. Các reviewer sẽ kiểm tra code, đưa ra ý kiến, và có thể yêu cầu thay đổi trước khi merge. Khi tất cả các phản hồi được giải quyết, PR có thể được merge bởi người quản trị hoặc owner của branch nhận.

## GitHub Issues và GitHub Projects

**Quản lý nhiệm vụ và lỗi (issue):**

- Tạo và quản lý issue: Issues giúp theo dõi các vấn đề, yêu cầu, hoặc bugs trong dự án. Có thể tạo issue mới, gán cho người phụ trách, thêm nhãn, và theo dõi tiến trình giải quyết.

**Sử dụng GitHub Projects để quản lý tiến độ và công việc nhóm:**

- Tạo và quản lý project: GitHub Projects cho phép tạo các bảng quản lý dự án, giúp tổ chức công việc, phân công nhiệm vụ, và theo dõi tiến độ. Có thể sử dụng các board Kanban, Scrum, hoặc board tùy chỉnh.

## GitHub Actions

**Khái niệm và cách hoạt động của GitHub Actions:**

- GitHub Actions: Là một công cụ tự động hóa CI/CD (Continuous Integration/Continuous Deployment) cho GitHub. Cho phép tạo các workflow để tự động chạy các bước kiểm thử, build, và deploy code mỗi khi có thay đổi được push vào repository.

**Tự động hóa quy trình CI/CD:**

- Cấu hình workflow: Các workflow được định nghĩa bằng YAML trong file `.github/workflows/`. Có thể sử dụng các template sẵn có hoặc tạo custom workflow phù hợp với nhu cầu dự án.

- Ví dụ: Một workflow cơ bản có thể bao gồm:

```yaml

name: CI

on: [push, pull\_request]

jobs:

build:

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v2

- name: Set up JDK 1.8

uses: actions/setup-java@v1

with:

java-version: 1.8

- name: Build with Maven

run: mvn clean install

- name: Run tests

run: mvn test

```

## GitHub Pages

**Tạo và triển khai website tĩnh từ GitHub repository:**

- **Cách tạo GitHub Pages**:

1. Chuyển đến repository muốn sử dụng GitHub Pages.

2. Trong Settings, tìm mục "GitHub Pages" và chọn nguồn (source) là `main` hoặc `master` branch (tùy thuộc vào cách bạn quản lý branch).

3. Chọn một địa chỉ URL cho trang web của bạn.

4. Lưu thay đổi.

- Triển khai website: Sau khi cấu hình xong, GitHub sẽ tự động build và deploy website tĩnh từ các file trong branch đã chọn. Website sẽ sẵn sàng truy cập tại địa chỉ URL đã chỉ định.

- Hỗ trợ các loại website: GitHub Pages hỗ trợ các loại website tĩnh như HTML, CSS, JavaScript, Jekyll (một framework cho blog), và nhiều hơn nữa.

# Github hỗ trợ cộng đồng lập trình viên:

## GitHub và Open Source

GitHub đã trở thành một trong những nền tảng quan trọng nhất cho các dự án mã nguồn mở trên thế giới. Với hàng triệu kho lưu trữ công khai, GitHub không chỉ là nơi lưu trữ mã nguồn, mà còn là nơi tạo ra sự kết nối giữa các lập trình viên từ khắp nơi trên thế giới.

### GitHub như một nền tảng hỗ trợ các dự án mã nguồn mở

Một trong những yếu tố quan trọng giúp GitHub trở thành trung tâm của cộng đồng mã nguồn mở là tính năng dễ sử dụng và tích hợp chặt chẽ với Git. GitHub cho phép các lập trình viên từ khắp nơi cùng tham gia vào một dự án, đóng góp ý tưởng và mã nguồn, giúp tăng tốc độ phát triển phần mềm. Các dự án mã nguồn mở thường là nơi mà những lập trình viên học hỏi và phát triển kỹ năng của mình thông qua việc cộng tác và review code từ các chuyên gia khác.

GitHub cung cấp rất nhiều công cụ hỗ trợ cho các dự án mã nguồn mở như GitHub Actions (tự động hoá quy trình làm việc), Issues (quản lý nhiệm vụ), và Discussions (diễn đàn trao đổi ý kiến), giúp tạo ra môi trường thuận lợi cho việc phát triển các dự án mã nguồn mở. Nhờ những công cụ này, các nhóm phát triển có thể làm việc một cách hiệu quả và minh bạch, đồng thời thu hút thêm sự tham gia từ cộng đồng.

### Các dự án nổi bật trên GitHub

Trên GitHub, có rất nhiều dự án mã nguồn mở nổi bật đã được cộng đồng lập trình viên đóng góp và sử dụng rộng rãi. Một số dự án điển hình có thể kể đến như:

+ **Linux Kernel:** Là dự án mã nguồn mở nổi bật nhất, cung cấp nền tảng hệ điều hành cho hàng triệu thiết bị trên toàn cầu.

+ **TensorFlow:** Một thư viện machine learning mạnh mẽ, được phát triển bởi Google và được sử dụng rộng rãi trong các ứng dụng AI.

+ **React**: Thư viện JavaScript phát triển giao diện người dùng, được Facebook phát triển và hiện nay đã trở thành công cụ phổ biến cho các nhà phát triển web.

Những dự án này không chỉ giúp thúc đẩy công nghệ phát triển mà còn chứng minh sức mạnh của mô hình mã nguồn mở và cách GitHub đóng góp vào việc phát triển chúng.

## GitHub Community và GitHub Sponsors

GitHub không chỉ là một công cụ kỹ thuật, mà còn là một nền tảng xã hội, nơi mà các lập trình viên có thể kết nối và học hỏi lẫn nhau. Thông qua cộng đồng GitHub, các lập trình viên có cơ hội tương tác và hỗ trợ nhau, tạo ra một môi trường phát triển phần mềm phong phú.

* + 1. **Cộng** đồng GitHub và cách thức tương tác giữa các lập trình viên

GitHub Community là một mạng lưới các lập trình viên từ khắp nơi trên thế giới. Cộng đồng này cho phép lập trình viên chia sẻ kiến thức, hỏi đáp về các vấn đề kỹ thuật, và cùng thảo luận về các dự án. Các tính năng như **Issues** và **Pull Requests** không chỉ hỗ trợ quản lý công việc mà còn là nơi các lập trình viên thảo luận về những cải tiến, lỗi, và cách thức phát triển phần mềm.

- Thông qua việc review code và thảo luận trực tiếp trên nền tảng, các lập trình viên có thể học hỏi lẫn nhau, tiếp nhận ý kiến từ những chuyên gia và cộng tác hiệu quả hơn. Điều này đặc biệt có giá trị trong các dự án mã nguồn mở, nơi mà các thành viên đến từ nhiều quốc gia và có múi giờ khác nhau.

* + 1. Chương trình GitHub Sponsors hỗ trợ các lập trình viên mã nguồn mở

GitHub Sponsors là một chương trình do GitHub khởi xướng, nhằm hỗ trợ tài chính cho các lập trình viên đóng góp vào các dự án mã nguồn mở. Đây là một cách để cộng đồng hoặc các công ty có thể tài trợ cho các nhà phát triển, giúp họ tiếp tục cống hiến cho các dự án mà không phải lo lắng về vấn đề kinh tế.

Chương trình này cho phép lập trình viên nhận được khoản tài trợ trực tiếp từ người dùng hoặc tổ chức thông qua GitHub. Nhờ vậy, nhiều nhà phát triển có thể dành toàn thời gian cho các dự án mã nguồn mở, từ đó cải thiện chất lượng và tốc độ phát triển của phần mềm.

## Hợp tác và làm việc nhóm trên GitHub

GitHub cung cấp nhiều công cụ mạnh mẽ để hỗ trợ việc hợp tác và làm việc nhóm giữa các lập trình viên, đặc biệt trong các dự án quy mô lớn và phân tán. Một số công cụ: Respository (kho lưu trữ), Pull Requests, Branching (Tạo nhánh), Code Review (Đánh giá mã),Issues (Theo dõi vấn đề),GitHub Projects, GitHub Pages

# Các công cụ hỗ trợ hợp tác (code review, project management)

GitHub tích hợp nhiều tính năng hỗ trợ việc hợp tác giữa các thành viên trong một nhóm phát triển. Một số công cụ nổi bật có thể kể đến:

+ **Code Review**: GitHub cho phép lập trình viên gửi **Pull Requests** khi hoàn thành một phần công việc. Các thành viên khác trong nhóm có thể review code, đưa ra nhận xét hoặc yêu cầu chỉnh sửa trước khi mã nguồn được hợp nhất vào nhánh chính. Điều này giúp đảm bảo chất lượng mã nguồn và tránh được các lỗi tiềm ẩn.

+ **Project Management**: GitHub cung cấp **GitHub Projects**, một công cụ quản lý dự án trực quan cho phép các nhóm theo dõi tiến độ công việc, phân công nhiệm vụ và quản lý các yêu cầu phát triển. Nó hoạt động giống như một bảng Kanban, cho phép nhóm phát triển theo dõi trạng thái công việc một cách hiệu quả.

+ Ngoài ra, còn có **Continuous Integration/Continuous Deployment (CI/CD):** Thông qua GitHub Actions, các nhóm có thể thiết lập quy trình tự động kiểm tra và triển khai phần mềm, giúp giảm thiểu rủi ro và tăng hiệu quả làm việc.

# Quản lý phiên bản và các lợi ích trong cộng tác dự án

GitHub dựa trên Git,, giúp theo dõi các thay đổi mã nguồn trong suốt quá trình phát triển dự án. Dưới đây là tóm tắt các lợi ích trong việc quản lý phiên bản và cộng tác dự án trên GitHub:

**+ Theo dõi lịch sử thay đổi**: Mọi thay đổi trong mã nguồn được ghi lại dưới dạng phiên bản, giúp dễ dàng theo dõi ai đã làm gì, khi nào và tại sao. Điều này giúp khôi phục lại mã từ các phiên bản trước nếu cần.

**+ Hỗ trợ cộng tác**: Nhiều thành viên có thể làm việc trên cùng một dự án mà không xung đột. GitHub cho phép làm việc độc lập trên các nhánh (branch), sau đó hợp nhất lại thông qua pull request.

**+ Hợp nhất mã dễ dàng (Merge)**: Khi một thành viên hoàn thành công việc trên nhánh của mình, họ có thể tạo pull request để yêu cầu hợp nhất (merge) với nhánh chính. Quá trình này cho phép xem xét và thảo luận trước khi tích hợp.

**+ Phân chia công việc**: Nhóm có thể chia nhỏ các nhiệm vụ hoặc tính năng trên các nhánh khác nhau, cho phép nhiều người làm việc đồng thời mà không ảnh hưởng đến mã chính.

**+ Giảm rủi ro lỗi**: Quản lý phiên bản cho phép phát hiện lỗi nhanh chóng bằng cách kiểm tra sự khác biệt giữa các phiên bản và khôi phục lại phiên bản ổn định trước đó nếu xảy ra sự cố.

**+ Đánh giá mã (Code review)**: GitHub tích hợp tính năng code review để nhóm có thể thảo luận và đánh giá mã trước khi nó được đưa vào nhánh chính, đảm bảo chất lượng và tính nhất quán.

**+ Tính minh bạch và trách nhiệm**: Mọi hoạt động trên kho lưu trữ đều công khai và có thể theo dõi, giúp tăng tính minh bạch và đảm bảo trách nhiệm của từng thành viên.

* Tóm lại, GitHub giúp việc quản lý phiên bản trở nên dễ dàng và hiệu quả, đặc biệt khi làm việc nhóm, giúp tối ưu hóa sự cộng tác, theo dõi thay đổi và đảm bảo chất lượng mã nguồn.

# Các ứng dụng thực tiễn của Github

## Tại sao nên chọn GitHub cho doanh nghiệp?

### Lý do doanh nghiệp sử dụng GitHub

* Nền tảng quen thuộc: Giúp giảm thời gian đào tạo, thu hút nhân tài và tận dụng cộng đồng hơn 100 triệu nhà phát triển.
* Tích hợp đám mây: GitHub là nền tảng hoàn chỉnh để xây dựng và cung cấp phần mềm an toàn, tích hợp các công cụ DevOps, thay thế các giải pháp tại chỗ khó bảo trì.
* Tính năng hiện đại: Thường xuyên cập nhật với các công cụ như GitHub Copilot và GitHub Actions.

### Cách doanh nghiệp sử dụng GitHub

* Cộng tác: Theo dõi công việc, lập kế hoạch và đánh giá mã nguồn thông qua các vấn đề, dự án, và yêu cầu hợp nhất.
* Năng suất:Sử dụng GitHub Copilot để nhận gợi ý mã bằng AI và phát triển nhanh chóng với GitHub Codespaces.
* Tự động hóa: Sử dụng GitHub Actions để tự động hóa quy trình triển khai, lưu trữ phần mềm với GitHub Packages.
* Bảo mật:Bảo mật mã nguồn, cập nhật phụ thuộc với Dependabot và theo dõi tình hình bảo mật trên các kho lưu trữ

## Githud trong học tập

* Thể hiện khả năng lập trình: Chia sẻ mã nguồn trên GitHub cho phép sinh viên thể hiện kỹ năng lập trình của mình, ngay cả khi chưa có nhiều kinh nghiệm làm việc chuyên nghiệp.
* Làm nổi bật kỹ năng hợp tác: GitHub hỗ trợ hợp tác, giúp sinh viên chứng minh khả năng làm việc nhóm bằng cách đóng góp vào các dự án.
* Thể hiện cam kết: Một hồ sơ GitHub hoạt động tích cực phản ánh sự cam kết với việc học hỏi và phát triển liên tục trong lĩnh vực phát triển phần mềm.
* Kỹ năng giải quyết vấn đề: Mã nguồn chia sẻ trên GitHub thể hiện khả năng giải quyết vấn đề, tư duy phản biện và cách tiếp cận các thách thức.
* Lợi thế trên thị trường việc làm: Kỹ năng sử dụng GitHub có giá trị trên thị trường việc làm, giúp sinh viên trở nên hấp dẫn hơn trong mắt nhà tuyển dụng.
* Đóng góp mã nguồn mở: Đây là nền tảng quan trọng cho các dự án mã nguồn mở, cho phép sinh viên xây dựng danh tiếng và học hỏi từ người khác.
* Kiểm soát phiên bản: GitHub rất quan trọng để kiểm soát phiên bản, giúp theo dõi thay đổi, hợp tác và khôi phục từ lỗi.
* Hợp tác: GitHub được sử dụng rộng rãi cho hợp tác, cho phép làm việc nhóm hiệu quả trong các dự án mã nguồn.

## Tích hợp GitHub với các công cụ CI/CD, IDE và các công cụ quản lý dự án.

### Tích hợp với các công cụ CI/CD:

* GitHub Actions: Là giải pháp CI/CD tích hợp sẵn trên GitHub, cho phép định nghĩa các workflow trong repository. Workflow này có thể bao gồm việc build, test, và triển khai tự động mỗi khi có thay đổi được đẩy lên GitHub.
* Jenkins: Là công cụ CI/CD mã nguồn mở có thể tích hợp với GitHub thông qua webhook, cho phép tự động build và triển khai mỗi khi có thay đổi từ GitHub.

### Tích hợp với IDE:

* GitHub có thể tích hợp với nhiều IDE phổ biến như Visual Studio Code, IntelliJ IDEA, hay Eclipse thông qua các plugin hoặc extension, giúp quản lý repository, đẩy mã, và thực hiện kiểm thử trực tiếp từ IDE.

### Tích hợp với các công cụ quản lý dự án:

* Jira: Tích hợp GitHub với Jira giúp đồng bộ các vấn đề GitHub với các vấn đề trên Jira, cho phép quản lý và theo dõi tiến độ dự án trên cả hai nền tảng.
* Trello: Tích hợp với Trello giúp liên kết các vấn đề từ GitHub với các thẻ Trello, tạo cái nhìn trực quan về tiến độ dự án và cải thiện sự phối hợp trong nhóm.

# So sánh Github với các nền tảng tương tự:

## So sánh GitHub với GitLab

\*GitHub:

* Cộng đồng lớn và nhiều người dùng, thuận lợi cho việc cộng tác mã nguồn mở.
* Tích hợp mạnh mẽ với các dịch vụ và công cụ bên ngoài.
* GitHub Actions cung cấp tự động hóa quy trình làm việc.
* GitHub Pages cho phép triển khai website tĩnh dễ dàng.

\*GitLab:

* Cung cấp cả phiên bản cài đặt tự chủ (self-hosted) và dịch vụ đám mây.
* Tự động hóa CI/CD được tích hợp sẵn và không cần cài đặt thêm.
* Quản lý vòng đời phần mềm toàn diện từ lập kế hoạch đến theo dõi, CI/CD và giám sát.
* Cung cấp tính năng DevOps tích hợp.

## So sánh GitHub với Bitbucket

\*GitHub:

* + - Phổ biến trong cộng đồng mã nguồn mở.
    - Giao diện người dùng trực quan và dễ sử dụng.
    - Mạng xã hội dành cho lập trình viên với tính năng theo dõi, sao chép (star) và fork.

\*Bitbucket:

* + Tích hợp mạnh mẽ với các sản phẩm của Atlassian như Jira và Trello.
  + Hỗ trợ cả Git và Mercurial.
  + Cung cấp kế hoạch miễn phí cho các nhóm nhỏ với tính năng CI/CD thông qua Bitbucket Pipelines.
  + Có mô hình giá cho các nhóm và doanh nghiệp lớn.

## Ưu và nhược điểm của GitHub so với các đối thủ cạnh tranh

\*Ưu điểm:

* Cộng đồng lớn nhất, có lợi cho việc tìm kiếm và cộng tác.
* Giao diện người dùng thân thiện và dễ sử dụng.
* Tích hợp mạnh với nhiều công cụ và dịch vụ bên ngoài.
* GitHub Actions và GitHub Pages là các tính năng mạnh mẽ cho tự động hóa và triển khai web.

\*Nhược điểm:

* Các kế hoạch miễn phí có giới hạn về tính năng so với phiên bản trả phí.
* Các kế hoạch trả phí có thể đắt đỏ so với các đối thủ cung cấp nhiều tính năng tương tự.
* Không có phiên bản tự chủ (self-hosted) như GitLab.

# Kết luận

**Tóm tắt những điểm chính đã được thảo luận**

- GitHub là một nền tảng phổ biến dành cho quản lý mã nguồn và cộng tác dự án, với các tính năng như repository, branch và pull request, issues và projects, GitHub Actions, và GitHub Pages.

- GitHub được so sánh với GitLab và Bitbucket dựa trên các tiêu chí như tính năng tự động hóa, quản lý vòng đời phần mềm, cộng đồng người dùng, tích hợp với các công cụ khác, và mô hình giá cả.

- Mỗi nền tảng có ưu và nhược điểm riêng, và lựa chọn tốt nhất phụ thuộc vào nhu cầu cụ thể của dự án và đội ngũ phát triển. GitHub nổi bật nhờ cộng đồng lớn và giao diện người dùng thuận tiện, trong khi GitLab và Bitbucket cung cấp tính năng CI/CD tích hợp và kế hoạch giá có lợi cho các nhóm và doanh nghiệp.

Khi lựa chọn nền tảng, các đội ngũ và tổ chức cần xem xét kỹ lưỡng các yếu tố như mức độ hỗ trợ tự động hóa, tính năng cộng tác, chi phí, và tính tương thích với công cụ hiện có để đạt được hiệu quả làm việc tối ưu.

**TÀI LIỆU THAM KHẢO**

1. Github là gì:

<https://topdev.vn/blog/github-la-gi/> [Truy cập ngày 12/09/2024]

1. Kiến trúc và công nghệ github:

<https://github.com/hocchudong/thuctap032016/blob/master/ThaiPH/OpenStack/ThaiPH_baocaotimhieucloudcomputingvaopenstack.md> [Truy cập ngày 12/09/2024]

1. Các tính năng của github:

<https://blog.haposoft.com/github-va-cac-chuc-nang-chinh/> [Truy cập ngày 12/09/2024]

1. Github hỗ trợ cộng đồng lập trình viên:

<https://wiki.matbao.net/github-la-gi-top-7-loi-ich-github-mang-lai-cho-lap-trinh-vien/#6-loi-ich-cua-github-doi-voi-cac-lap-trinh-vien>[Truy cập ngày 12/09/2024]

<https://mytour.vn/vi/blog/bai-viet/du-an-nha-tai-tro-cua-github-gio-cho-phep-nguoi-dung-ung-ho-cac-du-an-ma-nguon-mo-wired.html>[Truy cập ngày 12/09/2024]

1. Các ứng dụng thực tiễn của github
   1. Tại sao nên chọn GitHub cho doanh nghiệp?

<https://docs.github.com/en/enterprise-cloud@latest/admin/overview/about-github-for-enterprises>

* 1. Githud leand

<https://www.linkedin.com/pulse/why-essential-cse-student-learn-github-adib-reza#:~:text=Collaboration%3A%20GitHub%20is%20a%20popular,contribute%20to%20open%2Dsource%20projects>

* 1. Tích hợp GitHub với các công cụ CI/CD, IDE và các công cụ quản lý dự án.

<https://dev.to/pratik_kale/integrating-github-with-other-tools-5g7g?utm_source=zalo&utm_medium=zalo&utm_campaign=zalo>