KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

****

**BÁO CÁO MÔN HỌC**

**CÔNG NGHỆ PHẦN MỀM**

**TRIỂN KHAI GITEA**

**TRÊN MÔI TRƯỜNG DOCKER**

*Nhóm sinh viên thực hiện:*

Họ tên: Lâm Ngọc Tài

Trần Thái Hưng

Nguyễn Nhất Sang

Nguyễn Nhất Sang

*Giáo viên hướng dẫn:*

Họ tên: Nguyễn Bảo Ân

*Trà Vinh, tháng 07 năm 2023*

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

**A blue and white logo

Description automatically generated**

**BÁO CÁO MÔN HỌC**

**CÔNG NGHỆ PHẦN MỀM**

**TRIỂN KHAI GITEA**

**TRÊN MÔI TRƯỜNG DOCKER**

*Nhóm sinh viên thực hiện:*

Họ tên: Lâm Ngọc Tài

Trần Thái Hưng

Nguyễn Nhất Sang

Nguyễn Nhất Sang

*Giáo viên hướng dẫn:*

Họ tên: Nguyễn Bảo Ân

*Trà Vinh, tháng 07 năm 2023*

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

*Trà Vinh, ngày tháng năm 2023*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

# LỜI CẢM ƠN

Đầu tiên, chúng em xin bày tỏ lòng biết ơn chân thành đến tất cả các thầy cô giáo, giảng viên của Trường Đại học Trà Vinh, nhất là các thầy cô thuộc Khoa Kỹ thuật & Công nghệ, bộ môn Công nghệ thông tin, đã hỗ trợ chúng em rất nhiều để chúng em có thể hoàn thành bài báo cáo này một cách tốt nhất.

Tiếp đến, chúng em xin gửi lời tri ân đến thầy Nguyễn Bảo Ân – Giảng viên Khoa Kỹ thuật & Công nghệ, Trường Đại học Trà Vinh, đã luôn quan tâm và hướng dẫn chúng em với nhiều kiến thức bổ ích trong quá trình làm bài báo cáo.

Bài báo cáo của chúng em có thể còn nhiều thiếu sót do kiến thức và kinh nghiệm chưa đủ. Chúng em rất mong nhận được sự góp ý và chỉ bảo của các thầy cô để chúng em có thể học hỏi và hoàn thiện hơn trong những nghiên cứu sau này.

Cuối cùng, chúng em xin chúc các thầy cô luôn khỏe mạnh và hạnh phúc.

Chúng em xin trân trọng cảm ơn!

# MỤC LỤC

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 1](file:///C:\Users\ASUS\Desktop\fileCongNghePhanMen.docx#_Toc140001051)

[LỜI CẢM ƠN 2](#_Toc140001052)

[MỤC LỤC 3](#_Toc140001053)

[DANH MỤC HÌNH ẢNH 5](#_Toc140001054)

[CHƯƠNG 1. GIỚI THIỆU 6](#_Toc140001055)

[I. Tổng quan về dự án 6](#_Toc140001056)

[II. Mục tiêu bài báo cáo 6](#_Toc140001057)

[CHƯƠNG 2. PHÂN TÍCH YÊU CẦU 8](#_Toc140001058)

[I. Miêu tả yêu cầu dự án 8](#_Toc140001059)

[II. Phân tích các chức năng và tính năng cần hiện thực 8](#_Toc140001060)

[CHƯƠNG 3. THIẾT KẾ KIẾN TRÚC 10](#_Toc140001061)

[I. Xác định kiến trúc hệ thống 10](#_Toc140001062)

[CHƯƠNG 4. QUY TRÌNH PHÁT TRIỂN 11](#_Toc140001063)

[I. Sử dụng phương pháp Agile 11](#_Toc140001064)

[II. Sơ đồ quy trình phát triển 13](#_Toc140001065)

[III. Công cụ phát triển 14](#_Toc140001066)

[CHƯƠNG 5. TRIỂN KHAI 17](#_Toc140001067)

[I. Môi trường triển khai 17](#_Toc140001068)

[II. Các bước triển khai ứng dựng trên local 17](#_Toc140001069)

[III. Các bước triển khai ứng dựng trên VPS (Ubuntu) Azure 23](#_Toc140001070)

[CHƯƠNG 6. KẾT LUẬN 34](#_Toc140001071)

[I. Tổng kết kết quả đạt được 34](#_Toc140001072)

[II. Khó khăn và bài học 34](#_Toc140001073)

[III. Hướng phát triển 35](#_Toc140001074)

[IV. Kết luận 35](#_Toc140001075)

[TÀI LIỆU THAM KHẢO 36](#_Toc140001076)

# DANH MỤC HÌNH ẢNH

[Hình 1. Logo Gitea. 6](#_Toc140001465)

[Hình 2. Mô hình Client – Sever. 10](#_Toc140001466)

[Hình 3. Mô hình Agile. 11](#_Toc140001467)

[Hình 4. Quy trình Scrum. 12](#_Toc140001468)

[Hình 5. Product backlog của dự án. 13](#_Toc140001469)

[Hình 6. Sprint 1. 14](#_Toc140001470)

[Hình 7. Sprint 2. 14](#_Toc140001471)

[Hình 8. Logo Visual Studio code. 15](#_Toc140001472)

[Hình 9. Trang chủ Azure. 23](#_Toc140001473)

[Hình 10. Cấu hình máy ảo. 23](#_Toc140001474)

[Hình 11. Cấu hình máy ảo. 24](#_Toc140001475)

[Hình 12. Thông số cấu hình máy ảo. 25](#_Toc140001476)

[Hình 13. Máy ảo tạo thành công. 25](#_Toc140001477)

[Hình 14. Thông số máy ảo. 26](#_Toc140001478)

[Hình 15. Kết nối PuTTY. 26](#_Toc140001479)

[Hình 16. Initial Cofinguration của Gitea. 29](#_Toc140001480)

[Hình 17. General Setup. 30](#_Toc140001481)

[Hình 18. Thêm key SSH. 31](#_Toc140001482)

[Hình 19. Thêm key SSH thành công. 32](#_Toc140001483)

[Hình 20. Token của key. 32](#_Toc140001484)

[Hình 21. Thực thi lệnh lấy key verify. 32](#_Toc140001485)

[Hình 22. Dán SSH signature vào Armored SSH signature. 33](#_Toc140001486)

[Hình 23. Thêm SSH key thành công 33](#_Toc140001487)

# CHƯƠNG 1. GIỚI THIỆU

## I. Tổng quan về dự án

Gitea là một phần mềm dùng để lưu trữ và quản lý mã nguồn sử dụng Git cũng như các tính năng hợp tác khác như theo dõi lỗi, đánh giá mã, tích hợp liên tục, bảng kanban, vé và wiki. Nó hỗ trợ tự lưu trữ nhưng cũng cung cấp một phiên bản miễn phí công khai. Nó là một nhánh của Gogs và được viết bằng Go. Gitea có thể được lưu trữ trên tất cả các nền tảng được hỗ trợ bởi Go bao gồm Linux, macOS và Windows. Dự án được tài trợ trên Open Collective.



Hình 1. Logo Gitea.

## II. Mục tiêu bài báo cáo

Thứ nhất, bài báo cáo sẽ tìm hiểu về quy trình triển Agile, một phương pháp quản lý dự án linh hoạt và hiệu quả. Bài báo cáo sẽ giới thiệu các nguyên tắc, giá trị và lợi ích của quy trình Agile, cũng như các công cụ và kỹ thuật hỗ trợ việc áp dụng quy trình này trong thực tế.

Thứ hai, bài báo cáo sẽ triển khai gitea bằng docker, một ứng dụng quản lý mã nguồn mở và dễ sử dụng. Bài báo cáo sẽ hướng dẫn các bước cài đặt và cấu hình gitea trên docker, cũng như các tính năng và chức năng của gitea.

Thứ ba, bài báo cáo sẽ triển khai gitea lên VPS Azure, một dịch vụ cung cấp máy chủ ảo trên đám mây. Bài báo cáo sẽ giải thích lý do chọn VPS Azure làm nền tảng cho gitea, cũng như các bước để đăng ký, tạo và kết nối với VPS Azure. Bài báo cáo sẽ cũng sẽ chỉ ra các vấn đề và khó khăn khi triển khai gitea lên VPS Azure, cũng như các giải pháp và kiến nghị để khắc phục.

# CHƯƠNG 2. PHÂN TÍCH YÊU CẦU

## I. Miêu tả yêu cầu dự án

Yêu cầu dự án là dockerize gitea, một nền tảng quản lý mã nguồn mở, và triển khai lên azure, một dịch vụ điện toán đám mây của Microsoft. Để thực hiện yêu cầu này, bạn cần có kiến thức về docker, gitea và azure. Bạn cũng cần có tài khoản azure và quyền truy cập vào máy chủ ảo. Các bước cơ bản để dockerize gitea và triển khai lên azure là:

- Cài đặt docker trên máy chủ ảo azure

- Tải xuống và chạy image gitea từ docker hub

- Cấu hình gitea để sử dụng cổng và tên miền mong muốn

- Tạo một nhóm bảo mật azure và mở cổng cho gitea

- Kiểm tra truy cập gitea từ trình duyệt.

## II. Phân tích các chức năng và tính năng cần hiện thực

Quản lý repository:

Tạo mới, xóa và sao chép repository.

Quản lý branch và commit.

Hỗ trợ merge và rebase.

Đánh dấu và gắn thẻ các phiên bản.

Giao diện người dùng:

Giao diện web thân thiện và dễ sử dụng.

Quản lý tài khoản người dùng và phân quyền.

Tạo và quản lý nhóm người dùng.

Hỗ trợ tích hợp CI/CD:

* Tích hợp công cụ CI/CD như Jenkins, GitLab CI/CD hoặc Azure DevOps.
* Cung cấp tính năng tự động xây dựng, kiểm tra và triển khai phần mềm từ repository.

Hệ thống bảo mật:

* Xác thực người dùng và quản lý phân quyền truy cập.
* Bảo vệ dữ liệu và thông tin nhạy cảm.
* Ghi lại và giám sát các hoạt động trong hệ thống.

Công cụ hợp tác:

* Hỗ trợ việc nhận xét và thảo luận mã nguồn.
* Quản lý vấn đề và yêu cầu từ người dùng.
* Tích hợp hệ thống theo dõi lỗi (bug tracking) và yêu cầu tính năng (feature requests).

Tìm kiếm và duyệt mã nguồn:

* Tìm kiếm linh hoạt trong repository.
* Xem và duyệt lịch sử thay đổi của mã nguồn.
* Hiển thị thay đổi và so sánh giữa các phiên bản.

Quản lý vấn đề và yêu cầu:

* Tạo và quản lý vấn đề và yêu cầu từ người dùng.
* Gắn kết vấn đề và yêu cầu với commit và phiên bản tương ứng.

Ghi chú và nhận xét:

* Cho phép ghi chú và nhận xét trên mã nguồn và các vấn đề.
* Hỗ trợ đánh dấu và trả lời nhận xét.

Hỗ trợ tích hợp và mở rộng:

* Cung cấp API cho phép tích hợp với các ứng dụng và dịch vụ bên ngoài.
* Hỗ trợ cấu hình mở rộng và tùy chỉnh.

Các chức năng và tính năng trên đây chỉ là một số ví dụ phổ biến trong Gitea. Tùy thuộc vào yêu cầu cụ thể và mục đích sử dụng, có thể có thêm hoặc thay đổi các chức năng và tính năng trong quá trình hiện thực Gitea.

# CHƯƠNG 3. THIẾT KẾ KIẾN TRÚC

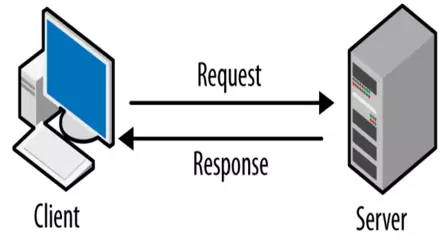
## I. Xác định kiến trúc hệ thống

Máy chủ (Server):

Máy chủ Gitea là phần chịu trách nhiệm xử lý và lưu trữ dữ liệu liên quan đến mã nguồn và quản lý các hoạt động liên quan.

Nó cung cấp API để tương tác với giao diện người dùng và các ứng dụng bên ngoài.

Máy chủ Gitea thực hiện các chức năng như quản lý người dùng, quản lý repository, quản lý phiên bản, xác thực, quyền truy cập và tích hợp CI/CD.



Hình 2. Mô hình Client – Sever.

Người dùng (Client):

Gitea cung cấp một giao diện người dùng web để người dùng tương tác với hệ thống.

Giao diện người dùng cho phép người dùng tạo, sao chép và quản lý repository, thực hiện các hoạt động version control, tạo yêu cầu và vấn đề, xem lịch sử thay đổi và thực hiện các hoạt động liên quan đến mã nguồn.

Kiến trúc hệ thống của Gitea tập trung vào cung cấp một giao diện trực quan và dễ sử dụng cho người dùng và một máy chủ phục vụ các yêu cầu và xử lý dữ liệu liên quan đến quản lý mã nguồn.

# CHƯƠNG 4. QUY TRÌNH PHÁT TRIỂN

## I. Sử dụng phương pháp Agile

### 1. Agile

Agile là một phương pháp quản lý dự án linh hoạt, được thiết kế để thích ứng với những thay đổi và yêu cầu của khách hàng. Điều đặc biệt về Agile là nó tập trung vào con người và sự tương tác hơn là quy trình và công cụ. Thay vì tuân theo một kế hoạch cố định, Agile khuyến khích sự linh hoạt và sáng tạo trong việc giải quyết vấn đề.

Một trong những đặc điểm quan trọng của Agile là sự coi trọng sản phẩm là kết quả của các chu kỳ lặp ngắn gọi là sprint. Trong mỗi sprint, nhóm phát triển liên tục nhận phản hồi từ khách hàng và cải tiến sản phẩm. Điều này giúp đảm bảo rằng sản phẩm được phát triển theo hướng đúng và đáp ứng được nhu cầu thực tế của khách hàng. A diagram of a software development process

Description automatically generated

Hình 3. Mô hình Agile.

Agile là một phương pháp quản lý dự án linh hoạt và thích ứng, có những đặc điểm chính là tập trung vào con người và sự tương tác, coi sản phẩm là kết quả của các chu kỳ lặp ngắn, khuyến khích sự hợp tác giữa các bên liên quan và đề cao sự linh hoạt và sáng tạo trong việc giải quyết vấn đề. Với các đặc tính này, Agile đã trở thành một phương pháp quản lý dự án phổ biến và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau.

### 2. Scrum

Scrum là một phương pháp quản lý dự án linh hoạt, tập trung vào việc tạo ra các sản phẩm có giá trị cao cho khách hàng. Scrum có ba đặc điểm chính đó là sự tự quản, đa năng và kích thước nhỏ của các nhóm scrum, cách tổ chức dự án thành các sprint ngắn và sử dụng các công cụ để theo dõi và cải tiến tiến độ.

Một trong những đặc điểm quan trọng của Scrum là các nhóm scrum là tự quản. Điều này có nghĩa là nhóm tự tổ chức và tự quản lý công việc của mình. Thay vì có một người quản lý dự án truyền thống, các thành viên trong nhóm scrum chịu trách nhiệm chung và làm việc cùng nhau để đạt được mục tiêu dự án. Tính đa năng của các nhóm scrum cũng quan trọng, vì mỗi thành viên đóng góp vào các khía cạnh khác nhau của dự án và có khả năng đảm nhận nhiều vai trò khác nhau.

Scrum chia dự án thành các sprint, các đợt phát triển ngắn kéo dài từ 2 đến 4 tuần. Mỗi sprint tập trung vào việc phát triển một phần của sản phẩm hoặc một tính năng cụ thể. Các sprint giúp tăng tốc độ phát triển và đưa ra các sản phẩm có giá trị ngay từ đầu. Khi mỗi sprint kết thúc, nhóm scrum tổ chức cuộc họp để xem xét kết quả, nhận phản hồi từ khách hàng và điều chỉnh kế hoạch cho sprint tiếp theo.

A diagram of a scrum process

Description automatically generated

Hình 4. Quy trình Scrum.

Tóm lại, Scrum là một phương pháp quản lý dự án linh hoạt, tập trung vào tạo ra các sản phẩm có giá trị cao cho khách hàng. Các đặc điểm chính của Scrum bao gồm sự tự quản, đa năng và kích thước nhỏ của các nhóm scrum, sự chia dự án thành các sprint ngắn và sử dụng các công cụ để theo dõi tiến độ, giải quyết vấn đề và cải tiến liên tục. Với cách tiếp cận này, Scrum đã trở thành một phương pháp phổ biến trong quản lý dự án và đóng góp vào sự thành công của nhiều dự án phát triển phần mềm.

## II. Sơ đồ quy trình phát triển

### Product backlog

Product backlog của dự án triển khai Gitea là danh sách các yêu cầu chức năng, phi chức năng và kỹ thuật trong quá trình phát triển phần mềm. Product backlog được quản lý bởi product owner, người có trách nhiệm xác định mức độ ưu tiên, phạm vi và thời gian của các yêu cầu. Product backlog là một công cụ linh hoạt, có thể thay đổi theo nhu cầu của dự án.

A screenshot of a progress bar

Description automatically generated

Hình 5. Product backlog của dự án.

Các Product backlog đã hoàn thành tất cả.

### Sprint

Một cách để quản lý dự án một cách hiệu quả là phân rã dự án thành các Sprint. Sprint là một khoảng thời gian nhất định, thường là từ một đến bốn tuần, trong đó một nhóm làm việc cùng nhau để hoàn thành một tập hợp các nhiệm vụ được xác định trước. Mục tiêu của mỗi Sprint là tạo ra một sản phẩm hoặc dịch vụ có giá trị, có thể kiểm tra và có thể sử dụng. Phân rã dự án thành các Sprint giúp nhóm làm việc linh hoạt, nhanh chóng và phản hồi với các thay đổi trong yêu cầu.

A screenshot of a computer

Description automatically generated

Hình 6. Sprint 1.

A screenshot of a computer

Description automatically generated

Hình 7. Sprint 2.

## III. Công cụ phát triển

### Công cụ lập trình Visual Studio code (VS Code)

Công cụ lập trình VS Code là một ứng dụng mã nguồn mở được phát triển bởi Microsoft. VS Code hỗ trợ nhiều ngôn ngữ lập trình, hệ điều hành và nền tảng. VS Code cung cấp các tính năng như chỉnh sửa, gỡ lỗi, kiểm tra, tương tác với Git và phát triển web. VS Code là một công cụ lập trình hiệu quả và linh hoạt cho các nhà phát triển.

A blue ribbon with a cross

Description automatically generated

Hình 8. Logo Visual Studio code.

### Cơ sở dữ liệu

Gitea là một ứng dụng quản lý mã nguồn mở, cho phép tạo và lưu trữ các dự án của mình trên máy chủ riêng. Gitea có thể kết nối với nhiều loại hệ quản trị cơ sở dữ liệu SQL, bao gồm MySQL, PostgreSQL, SQLite và TiDB. Chúng ta có thể cấu hình Gitea để sử dụng hệ quản trị cơ sở dữ liệu phù hợp với yêu cầu và tài nguyên của môi trường làm việc của mình. Gitea cung cấp một giao diện người dùng thân thiện và nhiều tính năng hỗ trợ quá trình phát triển phần mềm.

### Ngôn ngữ và công nghệ

Gitea được xây dựng bằng ngôn ngữ Go (Golang). Go là một ngôn ngữ lập trình hiệu năng cao được phát triển bởi Google, nổi tiếng với việc xử lý đa luồng hiệu quả và cú pháp dễ đọc.

Gitea là một nền tảng mã nguồn mở cho việc lưu trữ, quản lý và phân phối các dự án phần mềm. Gitea sử dụng nhiều công nghệ khác nhau để cung cấp các tính năng như giao diện web, hỗ trợ Git, tích hợp liên tục, bảo mật và khả năng mở rộng. Một số công nghệ được sử dụng trong Gitea bao gồm:

- Ngôn ngữ lập trình Go: Đây là ngôn ngữ chính được sử dụng để viết mã nguồn của Gitea. Go là một ngôn ngữ lập trình đa năng, hiệu suất cao và dễ học.

- Thư viện web framework Gin: Đây là một thư viện web framework được viết bằng Go, giúp Gitea xây dựng giao diện web nhanh chóng và dễ dàng.

- Hệ quản trị cơ sở dữ liệu SQLite, MySQL và PostgreSQL: Đây là các hệ quản trị cơ sở dữ liệu phổ biến, cho phép Gitea lưu trữ và truy vấn các dữ liệu liên quan đến các dự án, người dùng và cài đặt.

- Hệ thống kiểm soát phiên bản Git: Đây là một hệ thống kiểm soát phiên bản phân tán, cho phép Gitea quản lý và phân phối các mã nguồn của các dự án.

- Công cụ tích hợp liên tục Drone: Đây là một công cụ tích hợp liên tục được viết bằng Go, cho phép Gitea tự động kiểm tra, xây dựng và triển khai các mã nguồn của các dự án.

### 4. Công cụ hỗ trợ cho SCM, collaborative working

Gitea là một nền tảng quản lý mã nguồn mở, dựa trên Git, cho phép các nhóm làm việc cộng tác trên các dự án phần mềm. Gitea cung cấp các công cụ hỗ trợ cho SCM (Software Configuration Management), như khả năng tạo và quản lý các nhánh, thẻ, yêu cầu kéo, vấn đề và wiki. Gitea cũng hỗ trợ collaborative working, bằng cách cho phép các thành viên của nhóm giao tiếp, phản hồi và đánh giá lẫn nhau qua các tính năng như bình luận, thông báo và biểu đồ. Gitea là một giải pháp hiệu quả và an toàn để quản lý mã nguồn trong môi trường làm việc nhóm.

# CHƯƠNG 5. TRIỂN KHAI

## I. Môi trường triển khai

### 1. Docker

Docker là một nền tảng phần mềm cho phép bạn xây dựng, chạy và quản lý các ứng dụng bằng cách sử dụng các container. Container là các đơn vị phần mềm độc lập, nhẹ và di động, có thể chạy trên bất kỳ môi trường nào. Docker giúp bạn tận dụng các lợi ích của container hóa, như khả năng mở rộng, tính linh hoạt, hiệu suất cao và bảo mật tốt hơn. Docker cung cấp cho bạn các công cụ và dịch vụ để quản lý vòng đời của các ứng dụng container, từ việc phát triển, kiểm thử, triển khai cho đến vận hành.

### 2. VPS Azure

VPS Azure là một dịch vụ cho phép bạn tạo và quản lý các máy ảo trên nền tảng đám mây của Microsoft. VPS Azure cung cấp nhiều lựa chọn về kích thước, hệ điều hành, bộ nhớ và băng thông của máy ảo, phù hợp với nhu cầu và ngân sách của bạn. VPS Azure còn có các tính năng bảo mật, sao lưu, khôi phục và mở rộng linh hoạt, giúp bạn tận dụng tối đa khả năng của đám mây.

### 3. Google Cloud platform

Google Cloud platform là một bộ sưu tập các dịch vụ điện toán đám mây do Google cung cấp, bao gồm các dịch vụ lưu trữ, tính toán, mạng, big data, trí tuệ nhân tạo, bảo mật và quản lý. Google Cloud platform cho phép các doanh nghiệp và nhà phát triển xây dựng, triển khai và vận hành các ứng dụng trên cơ sở hạ tầng của Google, với khả năng mở rộng linh hoạt, hiệu suất cao và chi phí thấp. Google Cloud platform cũng tích hợp với các sản phẩm và công nghệ khác của Google, như Google Workspace, Google Maps, Google Analytics và TensorFlow.

## II. Các bước triển khai ứng dựng trên local

Cài đặt git, compose, Docker Compose(linux) hoặc Docker Desktop(window)

**Bước 1:** Clone ứng dụng từ github <https://github.com/go-gitea/gitea> .

**Bước 2:** CD và thư mục vừa clone về.

**Bước 3:** Thực thi lệnh “docker build -t gitea .”.

**Bước 4:** Tạo file nginx.conf.txt

Như sau:

server {

server\_name <domain hoặc ip pulic>;

root /var/www/html;

location / {

proxy\_pass http://localhost:3000

proxy\_set\_header HOST $host;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

}

**Bước 5:** Tạo file docker-compose.yml

Nội dung file docker-compose: <https://github.com/lntaivn/gitea_docker-compose/blob/main/docker-compose.yml> (ngày tạo 08/07/2023).

version: "3"

networks:

gitea:

external: false

services:

server:

image: gitea

container\_name: gitea

environment:

- USER\_UID=1000

- USER\_GID=1000

- GITEA\_\_database\_\_DB\_TYPE=mysql

- GITEA\_\_database\_\_HOST=db:3306

- GITEA\_\_database\_\_NAME=gitea

- GITEA\_\_database\_\_USER=gitea

- GITEA\_\_database\_\_PASSWD=gitea

restart: always

networks:

- gitea

volumes:

- ./gitea:/data

- /etc/timezone:/etc/timezone:ro

- /etc/localtime:/etc/localtime:ro

ports:

- "3000:3000"

- "2222:22"

depends\_on:

- db

Trong đoạn code trên:

**Network:**

Một mạng có tên "gitea" được định nghĩa và được sử dụng trong các dịch vụ khác trong tệp docker-compose.

**Services:**

Dịch vụ "server" được định nghĩa. Nó sử dụng image gitea, đồng thời có tên container là "gitea".

Các biến môi trường được thiết lập để cấu hình ứng dụng Gitea để sử dụng MySQL làm cơ sở dữ liệu. Các biến môi trường này bao gồm thông tin về MySQL như HOST, NAME, USER, PASSWD.

Dịch vụ này sẽ được khởi động lại luôn luôn (restart: always), nó cũng được gắn vào mạng "gitea" và có các khối lưu trữ (volumes) để lưu trữ dữ liệu Gitea và cấu hình thời gian (timezone).

Nó cũng mở các cổng 3000 và 2222 để truy cập vào ứng dụng Gitea.

Cuối cùng, nó phụ thuộc vào dịch vụ "db" (MySQL) thông qua depends\_on.

services:

server:

...

db:

image: mysql:8

restart: always

environment:

- MYSQL\_ROOT\_PASSWORD=gitea

- MYSQL\_USER=gitea

- MYSQL\_PASSWORD=gitea

- MYSQL\_DATABASE=gitea

networks:

- gitea

volumes:

- ./mysql:/var/lib/mysql

ports:

- "3306:3306"

**db:**

Dịch vụ "db" sử dụng image MySQL phiên bản 8 (mysql:8) và được khởi động lại luôn luôn. Các biến môi trường được thiết lập để đặt mật khẩu và tên người dùng cho MySQL, cùng với tên cơ sở dữ liệu (gitea).

Dịch vụ này cũng được gắn vào mạng "gitea" và có một khối lưu trữ để lưu trữ dữ liệu MySQL.

Cuối cùng, cổng 3306 được mở để cho phép kết nối đến cơ sở dữ liệu MySQL từ bên ngoài.

services:

server:

...

db:

...

nginx:

image: nginx

ports:

- "8080:80"

volumes:

- ./nginx.conf.txt:/etc/nginx/conf.d/default.conf

depends\_on:

- server

phpmyadmin:

image: phpmyadmin/phpmyadmin

container\_name: phpmyadmin

environment:

- PMA\_HOST=db

- PMA\_PORT=3306

- MYSQL\_ROOT\_PASSWORD=gitea

restart: always

networks:

- gitea

ports:

- "8000:80"

**nginx:**

Dịch vụ "nginx" sử dụng image nginx trên Docker hub và mở cổng 8080 để truy cập vào server Nginx từ bên ngoài.

Một khối lưu trữ được sử dụng để cung cấp cấu hình nginx thông qua tệp "nginx.conf.txt".

Dịch vụ này phụ thuộc vào dịch vụ "server" (Gitea) thông qua depends\_on.

**phpmyadmin:**

Dịch vụ "phpmyadmin" sử dụng image phpMyAdmin và có tên container là "phpmyadmin".

Các biến môi trường được thiết lập để chỉ định máy chủ MySQL (PMA\_HOST) và cổng (PMA\_PORT) để kết nối đến.

Nó cũng sử dụng mật khẩu gitea (MYSQL\_ROOT\_PASSWORD) cho quyền root của MySQL.

Dịch vụ này được khởi động lại luôn luôn, gắn vào mạng "gitea" và mở cổng 8000 để truy cập vào phpMyAdmin từ bên ngoài.

Trên cơ bản, tệp docker-compose này sử dụng các dịch vụ Docker để triển khai một môi trường ứng dụng Gitea hoàn chỉnh với MySQL, Nginx và phpMyAdmin.

## III. Các bước triển khai ứng dựng trên VPS (Ubuntu) Azure

**Bước 1**: đăng nhập Azuretạo máy ảo

**A screenshot of a computer

Description automatically generated**

Hình 9. Trang chủ Azure.

Nhấn vào Create → nhấn vào Azure virtual machine.

A screenshot of a computer

Description automatically generated

Hình 10. Cấu hình máy ảo.

Cài đặt các thông số máy ảo và chọn dịch vụ

- Thiết lập tên máy ảo - mạng máy ảo.

- Chọn dịch vụ ubuntu Server 22.04 LTS.

- Chọn dịch vụ dành riêng cho tài khoản free (free services eligible).

A screenshot of a computer screen

Description automatically generated

Hình 11. Cấu hình máy ảo.

Lựa chọn đăng nhập bằng Password

Điền tên đăng nhập - mật khẩu đăng nhập

Lựa chọn ports hỗ trợ cho HTTP, SSH, HTTPS

Nhấn Next

A screenshot of a computer

Description automatically generated

Hình 12. Thông số cấu hình máy ảo.

Thiết lập các thông số thành công nhấn Create

A screenshot of a computer

Description automatically generated

Hình 13. Máy ảo tạo thành công.

Tạo thành công máy ảo - nhấn vào Go to resource

A screenshot of a computer

Description automatically generated

Hình 14. Thông số máy ảo.

Giao diện khi máy ảo được tạo thành công.

Nhìn thấy được địa chỉ ip public của máy ảo của chúng ta là 20.48.61.133

**Bước 2:** Tiến hành kết nối từ xa với máy ảo sử dụng putty.

A computer screen shot of a computer

Description automatically generated

Hình 15. Kết nối PuTTY.

Đăng nhập vào putty nhập tên và mật khẩu đã tạo.

Đăng nhập thành công.

**Bước 3:** Built dự án gitea

Tiến hành cài các đặt các gối cần thiết để chạy ssh, docker, docker-compose, git, nginx

Tạo Git user:

sudo adduser --system --shell /bin/bash --gecos 'Git Version Control' --group --disabled-password --home /home/git git

**Bước 4**: Cấu hình nginx và docker để chạy dự án

Tạo thư mục để chứa dự án:

mkdir gitea

cd gitea

**Nginx:**

**- Truy cập HTTP và HTTPS:**

sudo ufw allow "Nginx Full"

**- Cấu hình Nginx:**

sudo nano /etc/nginx/sites-available/gitea

**Thêm cấu hình vào**

server {

# Lắng nghe yêu cầu về tên miền/địa chỉ IP của bạn.

server\_name 20.48.61.133;

root /var/www/html;

location / {

proxy\_pass http://localhost:3000  
 proxy\_set\_header HOST $host;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

}

}

**Khởi động lại Nginx:**

sudo ln -s /etc/nginx/sites-available/gitea /etc/nginx/sites-enabled/gitea

sudo nginx -t

sudo systemctl restart nginx

**Docker:**

- Tạo file docker-compose.yml:

nano docker-compose.yml

version: "3"

networks:

gitea:

external: false

services:

server:

image: gitea/gitea:latest

container\_name: gitea

environment:

- USER\_UID=1000

- USER\_GID=1000

restart: always

networks:

- gitea

volumes:

- ./gitea:/data

- /etc/timezone:/etc/timezone:ro

- /etc/localtime:/etc/localtime:ro

ports:

- "3000:3000"

- "2222:22"

volumes:

gitea: {}

**Chạy docker:**

docker-compose up -d

**Bước 5:** Kiểm nghiệm kết quả

A screenshot of a computer

Description automatically generated

Hình 16. Initial Cofinguration của Gitea.

Bước 6: Tạo SSH cho gitea

A screenshot of a computer

Description automatically generated

Hình 17. General Setup.

Trong SSH server port thay đổi 22 thành 2222.

Đối với Window mở Git Bash, đối với Ubuntu mở cmd.

Thực thi lệnh: ssh-keygen -t rsa -b 4096 -C "git@20.48.61.133:2222"

           Kết quả:

Generating public/private rsa key pair.

#xuất hiện dòng bên dưới nhấn enter dòng đầu là “đường dẫn chứa key”

Enter file in which to save the key (/c/Users/<tên đăng nhập>/.ssh/id\_rsa):

(Enter file in which to save the key (/home/<tên đăng nhập>/.ssh/): )

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Sau khi enter xong sẽ hiển thị The key's randomart image

Đối với Window:

Tiếp tục thực thi: cd "/c/Users/<tên đăng nhập>/.ssh”

ls -l

Đối với Ubuntu:

Nếu không vào được ssh sudo chmod 700 ~/.ssh

Di chuyển tới nơi chưa key:

cd /home/hung/.ssh/

Kết quả:

-rw-r--r-- 1 Admin 197121 3389 Jul 8 19:41 id\_rsa

-rw-r--r-- 1 Admin 197121 747 Jul 8 19:41 id\_rsa.pub

Thực thi lệnh: cat id\_rsa.pub

Kết quả xuất hiện như sau: ssh-rsa mã base64

vào http://20.48.61.133/user/settings/keys

nhấn “add key ssh”

Đặt tên key

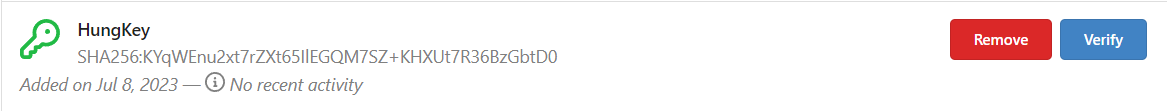
Sao chép và dán key public vào content

A screenshot of a computer

Description automatically generated

Hình 18. Thêm key SSH.

Nhấn Add key



Hình 19. Thêm key SSH thành công.

Nhấn verify

Sau khi nhấn hiện ra

A screenshot of a computer

Description automatically generated

Hình 20. Token của key.

Thấy dòng lệnh echo coppy nó

Đổi /path\_to\_your\_privkey thành đường dẫn đến nơi chứa “key private”

A screenshot of a computer

Description automatically generated

Hình 21. Thực thi lệnh lấy key verify.

nhấn enter key verify xuất hiện nó thường có cấu trúc như bên dưới

-----BEGIN SSH SIGNATURE-----

Mã hash: chữ ký SSH

-----END SSH SIGNATURE-----

copy hết rồi vào lại web dán vào

A screenshot of a computer

Description automatically generated

Hình 22. Dán SSH signature vào Armored SSH signature.

Nhấn verify



Hình 23. Thêm SSH key thành công

Bước 7: Thuê domain “lntaivn.com” để trỏ vào “<http://20.48.61.133>”.

Chỉnh sửa nginx:

sudo nano /etc/nginx/sites-available/gitea

Thay đổi: server\_name lntaivn.com;

sudo systemctl restart nginx

Truy cập đường link: <http://lntaivn.com/>

(ngày truy cập được 10/07/2023)

# CHƯƠNG 6. KẾT LUẬN

## I. Tổng kết kết quả đạt được

Đề tài triển khai Gitea là một dự án nhằm mục đích tạo ra một hệ thống quản lý mã nguồn mở dựa trên Git. Để đạt được mục tiêu này, đề tài đã thực hiện các công việc sau:

- Triển khai Gitea trên docker, một nền tảng cho phép chạy các ứng dụng trong các môi trường độc lập và linh hoạt. Điều này giúp tăng khả năng di chuyển, tái sử dụng và mở rộng của Gitea.

- Triển khai Gitea lên VPS Azure, một dịch vụ cung cấp máy chủ ảo trên đám mây. Điều này giúp tăng khả năng truy cập, bảo mật và ổn định của Gitea.

Như vậy, đề tài đã hoàn thành các kết quả đạt được theo kế hoạch và đóng góp cho việc phát triển mã nguồn mở trong cộng đồng.

## II. Khó khăn và bài học

Đề tài triển khai Gitea lên docker, VPS Azure là một đề tài thú vị và hữu ích cho những ai muốn sử dụng một hệ thống quản lý mã nguồn mở và linh hoạt. Tuy nhiên, trong quá trình thực hiện đề tài, chúng tôi cũng gặp phải nhiều khó khăn và bài học rút ra được. Một số khó khăn và bài học chính là:

- Khó khăn trong việc cài đặt và cấu hình docker trên VPS Azure. Chúng tôi phải tìm hiểu về các lệnh và tham số của docker, cũng như cách sử dụng docker-compose để quản lý các container. Chúng tôi cũng phải chú ý đến các vấn đề về bảo mật, như cách tạo và sử dụng ssh key, cách mở và đóng các port trên firewall, cách thiết lập SSL cho Gitea.

- Khó khăn trong việc tích hợp Gitea với các công cụ khác, như GitLab CI/CD, Jenkins, SonarQube. Chúng tôi phải nghiên cứu về các API và webhook của Gitea, cũng như cách cấu hình các công cụ để kết nối với Gitea. Chúng tôi cũng phải đảm bảo rằng các công cụ có thể truy cập được đến Gitea qua mạng nội bộ hoặc internet.

- Bài học rút ra được là chúng tôi nên lên kế hoạch và thiết kế chi tiết trước khi triển khai đề tài, bao gồm việc xác định mục tiêu, yêu cầu, phạm vi, thời gian. Chúng tôi cũng nên kiểm tra kỹ lưỡng các bước thực hiện và kiểm tra chất lượng của sản phẩm sau khi triển khai. Ngoài ra, chúng tôi cũng nên học hỏi từ kinh nghiệm của những người đã làm đề tài tương tự trước đó, và sử dụng các tài liệu và hướng dẫn có sẵn trên internet.

## III. Hướng phát triển

Đề tài triển khai gitea docker trên VPS Azure là một đề tài có nhiều hướng phát triển tiềm năng trong tương lai. Sau khi triển khai thành công, đề tài có thể được mở rộng để áp dụng cho nhiều loại dịch vụ khác nhau, như web hosting, Cloud Computing, Machine Learning, v.v. Đề tài cũng có thể được cải tiến để tăng hiệu năng, bảo mật, tính linh hoạt và dễ sử dụng của Gitea docker. Một số hướng phát triển cụ thể có thể kể đến như sau:

- Tối ưu hóa quá trình cài đặt và cấu hình gitea docker trên VPS Azure bằng cách sử dụng các công cụ tự động hóa như Ansible, Terraform, v.v.

- Tích hợp Gitea docker với các dịch vụ khác của Azure, như Azure DevOps, Azure Active Directory, Azure Storage, v.v. để tận dụng các tính năng và lợi ích của nền tảng đám mây.

- Nâng cấp phiên bản Gitea docker để hỗ trợ các tính năng mới nhất của Gitea, như code review, pull request, issue tracker, v.v.

- Thêm các tính năng bổ sung cho Gitea docker, như backup và restore dữ liệu, phân quyền người dùng, thiết lập webhook, v.v.

- Tăng cường bảo mật cho Gitea docker bằng cách áp dụng các biện pháp như mã hóa SSL/TLS, xác thực hai yếu tố, firewall, v.v.

## IV. Kết luận

Đề tài này đã trình bày cách triển khai Gitea, một hệ thống quản lý mã nguồn mở, lên docker, một nền tảng ảo hóa ứng dụng, và VPS Azure, một dịch vụ cung cấp máy chủ ảo trên đám mây. Qua đó, người dùng có thể tạo và quản lý các kho lưu trữ mã nguồn của mình một cách dễ dàng và an toàn. Đồng thời, việc sử dụng docker và VPS Azure giúp tăng tính linh hoạt, tiết kiệm chi phí và thời gian cho việc triển khai và vận hành hệ thống. Đề tài cũng đã đưa ra một số kiến nghị để cải thiện hiệu năng và bảo mật của hệ thống Gitea trên docker và VPS Azure trong tương lai.

# TÀI LIỆU THAM KHẢO

<https://www.techaddressed.com/tutorials/add-verify-ssh-keys-gitea/>

<https://docs.gitea.com/>

<https://github.com/go-gitea/gitea>