

TRƯỜNG ĐẠI HỌC KINH TẾ - LUẬT

**KHOA HỆ THỐNG THÔNG TIN**

**THỰC TẬP TỐT NGHIỆP**

NGÀNH THƯƠNG MẠI ĐIỆN TỬ

**XÂY DỰNG PLUGIN REDMINE VỚI NGÔN NGỮ LẬP TRÌNH RUBY ON RAILS**

**Công ty:**

**Công ty Cổ phần Viễn thông FPT**

GVHD: **TS. Nguyễn Quang Phúc**

Sinh viên:

MSSV: K184111360

Họ và tên: Trần Thị Trúc Anh

Lớp: K18411

**Thành phố Hồ Chí Minh, Tháng 11/2021**

**LỜI CẢM ƠN**

Trải qua khoảng thời gian thực tập hơn hai tháng tại công ty Cổ phần Viễn thông FPT, em đã học hỏi được nhiều kiến thức, kỹ năng để giải quyết các vấn đề trong công việc và cuộc sống. Đây sẽ là một khoảng thời gian vô cùng ý nghĩa đối với em khi được trải nghiệm vị trí thực tập sinh Software Developer tại công ty này. Công việc này đã giúp em hình thành một ý thức trách nhiệm và sự chủ động giải quyết trong mọi vấn đề được đặt ra. Ngoài ra, nó còn là động lực để em tiếp tục bước tiếp trong sự nghiệp và theo đuổi mục tiêu của bản thân mình.

Trước nhất, em xin gửi lời cảm ơn các thầy cô trong khoa Hệ thống đã tận tình truyền đạt đến em các kiến thức bổ ích để em có thể vận dụng nó ngày hôm nay. Em xin chân thành cảm ơn thầy Nguyễn Quang Phúc vì những lời khuyên, kinh nghiệm của thầy và đã dành thời gian giúp em hoàn thiện được báo cáo này.

Gửi lời cảm ơn đến chị Tiên, người đã hướng dẫn cho em và cho em cơ hội tham gia vào các dự án thực tế của công ty để em hoàn thành đề tải của mình một các tốt nhất. Cảm ơn vì sự chỉ dạy tận tình của chị trong suốt khoảng thời gian thực tập.

Em cũng muốn gửi lời cảm ơn đến anh Phong, chị My ở ban Nhân sự đã hỗ trợ em hết mình, sẵn sàng giúp em hoàn thành bài báo cáo này.

Sau cùng, em xin kính chúc quý công ty, quý đồng nghiệp, quý thầy cô mạnh khỏe, luôn thành công và hạnh phúc trong công việc và cuộc sống.

Em xin chân thành cảm ơn!

Trần Thị Trúc Anh

**PHIẾU ĐÁNH GIÁ KẾT QUẢ THỰC TẬP**

MSSV: K184111360

Họ tên: TRẦN THỊ TRÚC ANH

Tên cơ quan thực tập: CHI NHÁNH CÔNG TY CỔ PHẦN VIỄN THÔNG FPT

Địa chỉ cơ quan: Lô B3, Đường Sáng Tạo, Khu E Office, Khu Chế Xuất Tân Thuận, Phường Tân Thuận Đông, Quận 7, TP.Hồ Chí Minh

Điện thoại cơ quan: 028.7300.2222

Họ tên người đại diện cơ quan: Chị Nguyễn Thị Hoài Thanh

Chức vụ: Phó Ban nhân sự Điện thoại: 028.7300.2222   
Thời gian thực tập tại Cơ quan: Từ ngày 13/09/2021 Đến ngày: 15/11/2021

*\* Đề nghị* ***Quý Cơ quan*** *đánh giá* ***bằng cách đánh dấu* X** *vào cột xếp loại các nội dung đánh giá trong bảng sau.*

*Ghi chú:*

***Loại A: 2đ****;* ***Loại B: 1,5đ***

***Loại C: 1đ****;* ***Loại D: 0,5đ***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nội dung đánh giá | Xếp loại | | | |
| **A** | **B** | **C** | **D** |
| **1. Tinh thần kỷ luật, thái độ** | | | | |
| 1. Thực hiện nội quy của cơ quan | X |  |  |  |
| 1. Chấp hành giờ giấc làm việc | X |  |  |  |
| 1. Thái độ giao tiếp với đồng nghiệp, cộng sự | X |  |  |  |
| 1. Ý thức bảo vệ tài sản | X |  |  |  |
| 1. Tích cực trong công việc |  | X |  |  |
| **2. Kỹ năng chuyên môn, nghiệp vụ** | | | | |
| 1. Đáp ứng yêu cầu công việc |  | X |  |  |
| 1. Tinh thần học hỏi, nâng cao trình độ chuyên môn |  | X |  |  |
| 1. Có sáng kiến, năng động trong công việc |  | X |  |  |
| **3. Kết quả thực tập** | | | | |
| 1. Có sản phẩm ứng dụng thực tế đem lại lợi ích cho Cơ quan |  | X |  |  |
| 1. Mức độ hoàn thành nhiệm vụ thực tập |  | X |  |  |

**PHIẾU NHẬN XÉT CƠ QUAN THỰC TẬP**

**1. Ưu điểm**

* Có tinh thần tự học và tìm hiểu công nghệ tốt
* Có tư duy logic lập trình
* Thái độ chăm chỉ, học hỏi kinh nghiệm

**2. Hạn chế**

* + Chưa có kinh nghiệm tối ưu các process.
  + Chưa thể tư duy các nghiệp vụ tương lai để sáng tạo thêm.

**3. Đề xuất, góp ý**

* Trao dồi thông tin kiến thức về công nghệ mới ở nhiều trang mạng(hướng phát triển của tương lai, các công nghệ mình yêu thích đã có những cải tiến gì...).
* Tự đặt ra nhiều câu hỏi với bản thân "những công nghệ đó có tích hợp vào dự án mình được không?", "cái đó có gì hay ho ?"...
* Tự viết các demo để hiểu rõ hơn về công nghệ vừa học, và đặt ra nhiều câu hỏi và test thử để hiểu sâu hơn về công nghệ đó.
* Cần hiểu biết về luồng chạy của một công nghệ, ưu nhược điểm của nó so với các công nghệ khác, tránh hiểu biết ở mức lý thuyết nền.

……….., ngày……..tháng…….năm………

**XÁC NHẬN CỦA CƠ QUAN**

*Ký, ghi rõ họ tên, đóng dấu*

**PHIẾU ĐÁNH GIÁ BÁO CÁO THỰC TẬP**

**CỦA GIẢNG VIÊN HƯỚNG DẪN**

MSSV: K184111360

Họ tên: Trần Thị Trúc Anh

GVHD: TS. Nguyễn Quang Phúc

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tiêu chí** | **Tiêu chí cụ thể** | **Điểm** | **Ghi chú** |
| 1 | **Hình thức báo cáo (20%)** | Trình bày (5%) |  |  |
| Kết cấu báo cáo (10%) |  |  |
| Văn phong (5%) |  |  |
| 2 | **Nội dung báo cáo (45%)** | Kỹ năng phân tích (5%) |  |  |
| Mục tiêu (10%) |  |  |
| Chuyên môn (30%) |  |  |
| 3 | **Thái độ của sinh viên (15%)** |  |  |  |
| 4 | **Doanh nghiệp đánh giá (20%)** |  |  | *GVHD quy đổi từ đánh giá của DN* |
| **TỔNG ĐIỂM** | | |  |  |

……….., ngày……..tháng…….năm………

**Giảng viên hướng dẫn**

*Ký, ghi rõ họ tên*

**PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

……….., ngày……..tháng…….năm………

**Giảng viên hướng dẫn**

*Ký, ghi rõ họ tên*

**LỊCH TRÌNH LÀM VIỆC VÀ NỘI DUNG CÔNG VIỆC**

Thời gian: từ ngày 13/09/2021 đến ngày 15/11/2021

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Thời gian | Công việc | Nơi làm việc |
| 1 | Ngày 13/09/2021 | * Làm quen với mentor và các thực tập sinh khác trong ban * Giới thiệu sơ bộ về công việc | Tại nhà |
| 2 | Ngày 15/09/2021 đến ngày 23/09/2021 | * Tham khảo tài liệu liên quan * Cài đặt Docker (nền tảng để phát triển ứng dụng) và công cụ quản lí dự án (Redmine) trên nền tảng đó | Tại nhà |
| 3 | Ngày 24/09/2019 đến ngày 30/09/2021 | * Xây dựng plugin khảo sát cho redmine (Tham khảo tài liệu) * Cài redis trên docker | Tại nhà |
| 4 | Ngày 30/9/2021 đến ngày 05/10/2021 | * Tạo ra plugin có thể ghi đè các chức năng của Issues Redmine | Tại nhà |
| 5 | Ngày 5/10/2021 đến ngày 14/10/2021 | * Config mail để gửi và nhận thông báo từ redmine * Thêm themes cho redmine * Cài đặt plugin Easy Gantt, plugin DMSF | Tại nhà |
| 6 | Ngày 15/10/2021 đến ngày 18/10/2021 | * Nhận và phân tích nghiệp vụ quản lí nhân viên | Tại nhà |
| 7 | Ngày 19/10/2021 đến ngày 28/10/2021 | * Tạo một plugin mới để quản lí nhân viên * Xây dựng view để quản lí ( hiển thị, thêm, xóa, sửa thông tin nhân viên) * Phân tích database * Tạo dữ liệu lưu về database thông qua các API có sẵn | Tại nhà |
| 8 | Ngày 28/10/2021 đến ngày 30/10/2021 | * Sửa lỗi và báo cáo đợt 1 | Tại nhà |
| 9 | Ngày 31/10/2021 đến ngày 9/11/2021 | * Xây dựng chức năng tạo, lưu các tệp đính kèm và hiển thị ảnh đại diện của nhân viên | Tại nhà |
| 10 | Ngày 9/11/2021 đến ngày 15/11/2021 | * Sửa lỗi và hoàn thiện tính năng cuối cùng * Báo cáo và bàn giao sản phẩm | Tại nhà |

Mục lục

[Chương 1. Giới thiệu tổng quan 1](#_Toc92105419)

[1.1. Tổng quan về đề tài 1](#_Toc92105420)

[1.1.1. Lý do chọn đề tài 1](#_Toc92105421)

[1.1.2. Mục tiêu ý nghĩa của đề tài 1](#_Toc92105422)

[1.1.3. Phạm vi đề tài 1](#_Toc92105423)

[1.2. Tổng quan về công ty thực tập 1](#_Toc92105424)

[1.2.1. Giới thiệu chung về công ty Cổ phần Viễn thông FPT 1](#_Toc92105425)

[1.2.2. Vai trò và nhiệm vụ của công ty 2](#_Toc92105426)

[1.2.3. Giới thiệu về phòng Hệ thống thuộc Trung tâm Hệ thống thông tin 3](#_Toc92105427)

[1.3. Phỏng vấn nhân viên tại phòng Công nghệ thông tin 3](#_Toc92105428)

[Chương 2. Cơ sở lý thuyết 5](#_Toc92105429)

[2.1. Phát triển website 5](#_Toc92105430)

[2.2. Ngôn ngữ lập trình Ruby và framework Rails 5](#_Toc92105431)

[2.3. Hệ điều hành Ubuntu 5](#_Toc92105432)

[2.4. Giới thiệu về docker 6](#_Toc92105433)

[2.5. Redmine (Phần mềm quản lý task) 6](#_Toc92105434)

[Chương 3. Các công việc triển khai tại đơn vị thực tập 7](#_Toc92105435)

[3.1. Mô tả tóm tắt vị trí công việc thực tập 7](#_Toc92105436)

[3.2. Làm việc với mã nguồn mở Redmine 7](#_Toc92105437)

[3.2.1. Cài đặt môi trường làm việc 7](#_Toc92105438)

[3.2.2. Cài đặt và khởi chạy Redmine 10](#_Toc92105439)

[3.3. Làm việc với hệ thống Redmine 12](#_Toc92105440)

[3.3.1. Cấu hình email (smtp) 12](#_Toc92105441)

[3.3.2. Thêm chủ đề (theme) cho Redmine 15](#_Toc92105442)

[3.3.3. Cài đặt các plugin cho redmine 17](#_Toc92105443)

[3.4. Xây dựng plugin cho Redmine 23](#_Toc92105444)

[3.4.1. Tạo mới plugin 23](#_Toc92105445)

[3.4.2. Tạo plugin ghi đè tính năng issues 28](#_Toc92105446)

[3.4.3. Plugin quản lí staffs 33](#_Toc92105447)

[Chương 4. Đánh giá công việc 45](#_Toc92105448)

[4.1. Kết quả thu được 45](#_Toc92105449)

[4.2. Thuận lợi 45](#_Toc92105450)

[4.3. Khó khăn 46](#_Toc92105451)

[Chương 5. Tổng kết 46](#_Toc92105452)

[5.1. Tổng kết những ưu điểm 46](#_Toc92105453)

[5.2. Tổng kết những nhược điểm 46](#_Toc92105454)

[5.3. Đề xuất những giải quyết 46](#_Toc92105455)

[5.4. Kinh nghiệm rút ra sau khi thực tập 47](#_Toc92105456)

[Tài liệu tham khảo: 48](#_Toc92105457)

Danh mục hình ảnh

[Hình 1.1. Logo FPT Telecom 2](#_Toc92105354)

[Hình 2.1. Chân dung Yukihiro Matsumoto 5](#_Toc92105355)

[Hình 3.1. Terminal tạo docker container ‘hello-word’ 8](#_Toc92105356)

[Hình 3.2. Cấu hình file Dockerfile 8](#_Toc92105357)

[Hình 3.3. Cấu hình file docker-compose yaml 9](#_Toc92105358)

[Hình 3.4. Cấu hình file docker-compose yaml 9](#_Toc92105359)

[Hình 3.5. Cấu hình file database yaml 10](#_Toc92105360)

[Hình 3.6. Danh sách các container đang chạy 11](#_Toc92105361)

[Hình 3.7. Giao diện trang chủ của Redmine 11](#_Toc92105362)

[Hình 3.8. Cấu hình email ở file configuration yaml 12](#_Toc92105363)

[Hình 3.9. Giao diện email notifications 13](#_Toc92105364)

[Hình 3.10. Giao diện email notifications 13](#_Toc92105365)

[Hình 3.11. Giao diện tạo mới User 14](#_Toc92105366)

[Hình 3.12. Terminal Action Mailer 14](#_Toc92105367)

[Hình 3.13. Nội dung mail được gửi đến User 15](#_Toc92105368)

[Hình 3.14. Thêm themes cho Redmine 16](#_Toc92105369)

[Hình 3.15. Giao diện điều chỉnh hiển thị 16](#_Toc92105370)

[Hình 3.16. Giao diện thay đổi khi sử dụng chủ đề tùy chỉnh 17](#_Toc92105371)

[Hình 3.17. Các file, folder của Easy Gantt trong plugin 18](#_Toc92105372)

[Hình 3.18. File init.rb của plugin Easy Gantt 18](#_Toc92105373)

[Hình 3.19. Thông tin plugin Easy Gantt hiển thị ở Redmine 19](#_Toc92105374)

[Hình 3.20. Giao diện của Easy Gantt 20](#_Toc92105375)

[Hình 3.21. File init.rb của plugin DMSF 21](#_Toc92105376)

[Hình 3.22. Thông tin plugin DMSF 22](#_Toc92105377)

[Hình 3.23. Giao diện plugin DMSF 22](#_Toc92105378)

[Hình 3.24. Terminal khởi tạo plugin 23](#_Toc92105379)

[Hình 3.25. File init.rb của plugin Polls 24](#_Toc92105380)

[Hình 3.26. Thông tin của plugin Polls 24](#_Toc92105381)

[Hình 3.27. Terminal tạo model Polls 25](#_Toc92105382)

[Hình 3.28. File migrate của Polls 25](#_Toc92105383)

[Hình 3.29. Phương thức vote ở model 26](#_Toc92105384)

[Hình 3.30. Khởi tạo controller Polls 26](#_Toc92105385)

[Hình 3.31. Các phương thức ở file polls\_controller.rb 27](#_Toc92105386)

[Hình 3.32. View hiển thị các cuộc thăm dò 27](#_Toc92105387)

[Hình 3.33. Định tuyến cho plugin Polls 28](#_Toc92105388)

[Hình 3.34. Giao diện của Polls 28](#_Toc92105389)

[Hình 3.35. Giao diện Issues Redmine 29](#_Toc92105390)

[Hình 3.36. Giao diện tạo mới Issues 29](#_Toc92105391)

[Hình 3.37. File show.html.erb của Issues 30](#_Toc92105392)

[Hình 3.38. Chỉnh sửa file locale en.yml 30](#_Toc92105393)

[Hình 3.39. Giao diện Show issues sau khi chỉnh sửa 30](#_Toc92105394)

[Hình 3.40. File migration add column vào Issues 31](#_Toc92105395)

[Hình 3.41. Module Issue Patch 32](#_Toc92105396)

[Hình 3.42. Khai báo Module ở file init.rb của plugin Polls 32](#_Toc92105397)

[Hình 3.43. Giao diện edit Issue 33](#_Toc92105398)

[Hình 3.44. Ngày hoàn thành issue hiển thị ở giao diện Show 33](#_Toc92105399)

[Hình 3.45. File init.rb của plugin Staffs 36](#_Toc92105400)

[Hình 3.46. Thông tin plugin Staffs 36](#_Toc92105401)

[Hình 3.47. File migration add colum cho Users 37](#_Toc92105402)

[Hình 3.48. Giao diện edit User của Redmine 38](#_Toc92105403)

[Hình 3.49. File new.html.erb của User được chia làm nhiều fieldset 39](#_Toc92105404)

[Hình 3.50. Giao diện edit User sau khi được chỉnh sửa 39](#_Toc92105405)

[Hình 3.51. Giao diện edit User sau khi được chỉnh sửa 40](#_Toc92105406)

[Hình 3.52. Phương thức new của module UserControllerPatch 41](#_Toc92105407)

[Hình 3.53. Phương thức create của module UserControllerPatch 41](#_Toc92105408)

[Hình 3.54. Phương thức user\_params 41](#_Toc92105409)

[Hình 3.55. Parameters được truyền đi 42](#_Toc92105410)

[Hình 3.56. Các plugin nội bộ của Redmine 43](#_Toc92105411)

[Hình 3.57. Module UserPatch 43](#_Toc92105412)

[Hình 3.58. Thẻ file\_field\_tag được thêm ở view 44](#_Toc92105413)

[Hình 3.59. Giao diện edit User 44](#_Toc92105414)

[Hình 3.60. Thẻ image\_tag được thêm ở view 45](#_Toc92105415)

[Hình 3.61. Ảnh đại diện được hiển thị 45](#_Toc92105416)

Danh mục bảng biểu

[Bảng 3‑1 Bảng cấu trúc dữ liệu của staffs 36](#_Toc89090491)

# **Chương 1. Giới thiệu tổng quan**

* 1. **Tổng quan về đề tài**
     1. **Lý do chọn đề tài**

Redmine là một phần mềm khá phổ biến và được sử dụng ở một số công ty công nghệ của Nhật Bản. Nó chính là một open source hoàn toàn miễn phí và ngoài các tính năng được cấp sẵn, ta hoàn toàn có thể xây dựng thêm các plugin hoặc cài các plugin có sẵn để phục vụ nhu cầu của từng công ty cũng để tăng cường hiệu quả của dự án. Bài báo cáo dưới đây sẽ nêu ra một cách chi tiết cách cài đặt và xây dựng plugin cho Redmine bằng ngôn ngữ lập trình Ruby On Rails.

* + 1. **Mục tiêu ý nghĩa của đề tài**
* Mục tiêu trước nhất của bài báo cáo này là xây dựng, phát triển plugin cho Redmine trên nền tảng Ruby On Rails, thông qua đó có thể phát triển một plugin quản lý nhân viên.
* Nắm được nghiệp vụ, thao tác với ứng dụng Redmine, từ đó có thể theo dõi dự án và các vấn đề xảy ra.
* Hiểu được quy trình để phát triển một tính năng với mã nguồn mở.
  + 1. **Phạm vi đề tài**

Toàn bộ công việc được thực hiện từ ngày 13/09/2021 đến ngày 15/11/2021. Với nội dung xoay việc lập trình, phát triển mã nguồn mở Redmine với ngôn ngữ Ruby và một số nội dung liên quan.

* 1. **Tổng quan về công ty thực tập**
     1. **Giới thiệu chung về công ty Cổ phần Viễn thông FPT**
        1. **Lịch sử hình thành và phát triển**

Công ty Cổ phần Viễn thông FPT (tên gọi tắt là FPT Telecom) là một trong những công ty con thuộc Tập đoàn Công nghệ FPT. Hiện đang là một trong những nhà cung cấp dịch vụ Viễn thông và Internet hàng đầu khu vực.



Hình 1.1. Logo FPT Telecom

Thành lập ngày 31/01/1997, khởi nguồn từ Trung tâm Dịch vụ Trực tuyến do 4 thành viên sáng lập cùng sản phẩm mạng Intranet đầu tiên của Việt Nam mang tên “Trí tuệ Việt Nam – TTVN”, sản phẩm được coi là đặt nền móng cho sự phát triển của Internet tại Việt Nam.

Trong suốt quá trình hoạt động, FPT Telecom luôn không ngừng nỗ lực với mục tiêu cao nhất là mang lại sự hài lòng cho khách hàng. Với hơn 24 năm hoạt động và phát triển, công ty đã gặt hái được không ít những thành tựu nổi bật. Tổng số cán bộ nhân viên của công ty đã hơn 9.500 người với 90 chi nhánh khắp cả nước.

* + - 1. **Lĩnh vực kinh doanh**
* **FPT Telecom kinh doanh ở các lĩnh vực:**
* Cung cấp hạ tầng mạng viễn thông cho dịch vụ Internet băng rộng
* Dịch vụ giá trị gia tăng trên mạng Internet, điện thoại di động
* Dịch vụ Truyền hình trả tiền
* Dịch vụ tin nhắn, dữ liệu, thông tin giải trí trên mạng điện thoại di động
* Thiết lập hạ tầng mạng và cung cấp các dịch vụ Viễn thông, Internet
* Xuất nhập khẩu thiết bị Viễn thông và Internet
* Dịch vụ Viễn thông cố định nội hạt
* Dịch vụ Viễn thông giá trị gia tăng
* Dịch vụ Viễn thông cố định đường dài trong nước
  + 1. **Vai trò và nhiệm vụ của công ty**

Với sứ mệnh tiên phong mang Internet, mang kết nối đến với người dân Việt Nam cùng mong muốn lớn lao mỗi gia đình Việt Nam đều sử dụng ít nhất một dịch vụ của Công ty, FPT Telecom đang nỗ lực thực thi Chiến lược “Mang đến trải nghiệm tuyệt vời cho khách hàng” trên cơ sở phát huy giá trị văn hóa cốt lõi “Lấy khách hàng làm trọng tâm” và nền tảng sức mạnh công nghệ FPT, từ đó tiên phong trở thành Nhà cung cấp dịch vụ số có trải nghiệm khách hàng vượt trội, tốt nhất tại Việt Nam.

Hãng không chỉ cải tiến mà còn mang đến nhiều ứng dụng thực tế mới, giúp khách hàng nâng cao hiệu quả sử dụng cũng như có thêm trải nghiệm mới. Song hướng tới phát triển các hoạt động vì cộng đồng công nghệ, từ đó xây dựng hệ sinh thái công nghệ bền vững.

* + 1. **Giới thiệu về phòng Hệ thống thuộc Trung tâm Hệ thống thông tin**

Trung tâm Hệ thống thông tin (ISC) FPT Telecom được tổ chức thành 4 khối Công nghệ, Phát triển, Vận hành và Hỗ trợ. Tại chi nhánh Tp.Hồ Chí Minh đang có 363 nhân viên, trong đó Phòng Hệ Thống (ISCHT) có 47 nhân viên đang làm việc.

* **Nhiệm vụ của trung tâm hệ thống thông tin:**
* Xây dựng, phát triển, vận hành, đảm bảo an toàn và bảo mật hệ thống thông tin.
* Xây dựng và vận hành các hệ thống kết nối với Core hệ thống thông tin của Công ty: Radius, callcenter, quản lý thiết bị, hệ thống thanh toán, hệ thống SMS và các hệ thống khác.
* Nghiên cứu và tích hợp các hệ thống bên ngoài vào Core hệ thống thông tin, phối hợp với các đơn vị khác nghiên cứu ứng dụng công nghệ trong việc tích hợp hệ thống và triển khai các dịch vụ gia tăng trên nền internet của công ty.
  1. **Phỏng vấn nhân viên tại phòng Công nghệ thông tin**
* **Thông tin người được phỏng vấn**
* Họ và tên: Đỗ Cẩm Tiên
* Chức vụ: Teamlead
* Thâm niên công tác: 3.5 năm
* Nhiệm vụ:
* Quản lý team developer phát triển các dự án opensource, PHP, Ruby
* Vận hành và phát các triển dự án quản lý, điều khiển thiết bị modem
* Vận hành và phát triển các hệ thống quản lý dự án, tasks, dashboard, staffs, KPI...
* Vận hành và phát triển Identity and Access Management tích hợp cho các hệ thống FPT Telecom
* Vận hành và phát triển Home Wifi cho 10 triệu CPE
* Cung cấp API cho các dự án mobile, web service, web client của FPT Telecom.
* **Nội dung phỏng vấn**
* Thuận lợi trong công việc:
* Môi trường thuận lợi để phát triển kỹ năng lập trình, viết tài liệu, xử lý dữ liệu lớn của một tập đoàn FPT Telecom viễn thông
* Phong cách làm việc chuyên nghiệp
* Có nhiều anh chị có kỹ năng và đạt chứng chỉ quốc tế, hỗ trợ các bạn mới phát triển
* Tiếp xúc với nhiều vị trí khác nhau, trao đổi kỹ năng và kiến thức.
* Khó khăn trong công việc:
* Áp lực trong công việc cao, đòi hỏi tính tỉ mỉ và cẩn thận
* Hệ thống lớn ảnh hướng đến nhiều khách hàng trên toàn quốc
* Đòi hỏi nhân viên phát triển kỹ năng lập trình tối ưu hệ thống hiệu quả, giữ cho hệ thống chạy ổn định
* Có kế hoạch xử lý lỗi để hạn chế ảnh hưởng đến các đội chăm sóc khách hàng trên toàn quốc.
* Các kiến thức cần thiết cho vị trí thực tập và các đề xuất để phát triển công ty
* Nắm vững các kiến thức lập trình cơ bản
* Tìm hiểu kỹ các framework, keyword mà teamlead đưa ra
* Nắm rõ các nghiệp vụ được giao, và mục đích của các chức năng để vận dụng phát triển theo hướng tối ưu nhất có thể
* Biết lắng nghe và chủ động tương tác, trao đổi với người hướng dẫn thực tập
* Phát triển các kỹ năng còn thiếu sót, tìm hiểu các công nghệ mới, tối ưu và xử lý các vấn đề mà dự án gặp phải
* Đạt các chứng chỉ quốc tế liên quan đến công việc hiện tại.

# **Chương 2. Cơ sở lý thuyết**

* 1. P**hát triển website**

Website hay còn gọi là trang web là một tập hợp các thông tin được thể hiện dưới dạng văn bản, hình ảnh, video,… thường nằm trên một tên miền và được lưu trữ trên máy chủ. Website có thể truy cập được thông qua mạng Isnternet.

Phát triển web là việc xây dựng và duy trì các trang web. Việc này có thể bao gồm phát triển một trang tĩnh đơn giản của văn bản thuần túy đến các ứng dụng web phức tạp, doanh nghiệp điện tử và dịch vụ mạng xã hội.

Các nhà phát triển web sẽ thực hiện điều này bằng cách sử dụng nhiều ngôn ngữ mã hóa khác nhau. Ngôn ngữ họ sử dụng phụ thuộc vào loại nhiệm vụ mà họ đang thực hiện trước và nền tảng mà họ đang làm việc.

Được phân thành front-end (phía người dùng) và back-end (phía máy chủ).

* 1. **Ngôn ngữ lập trình Ruby và framework Rails**

Ruby là một [ngôn ngữ lập trình](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh) [hướng đối tượng](https://vi.wikipedia.org/wiki/L%E1%BA%ADp_tr%C3%ACnh_h%C6%B0%E1%BB%9Bng_%C4%91%E1%BB%91i_t%C6%B0%E1%BB%A3ng), được thông dịch, mã nguồn mở được tạo ra bởi Yukihiro Matsumoto, người đã chọn tên của viên đá quý để gợi ý "một viên ngọc của một ngôn ngữ".



Hình 2.1. Chân dung Yukihiro Matsumoto

Rails(framework) là một khung ứng dụng web phía máy chủ được viết bằng Ruby theo Giấy phép MIT (Massachusetts Institute of Technology). Rails là một khuôn khổ model – view – controller (MVC), cung cấp các cấu trúc mặc định cho cơ sở dữ liệu, dịch vụ web và các trang web.

* 1. **Hệ điều hành Ubuntu**

Ubuntu Server là một hệ điều hành máy chủ, được phát triển bởi Canonical và các lập trình viên mã nguồn mở trên khắp thế giới, hoạt động với hầu hết mọi nền tảng phần cứng hoặc ảo hóa. Nó có thể phục vụ các trang web, chia sẻ tệp và vùng chứa, cũng như mở rộng các dịch vụ trên đám mây.

* 1. **Giới thiệu về docker**

Docker là một nền tảng, dự án mã nguồn mở viết bởi Dotcloud (Công ty A PaaS) được sinh ra để phát triển, vận chuyển và chạy các ứng dụng. Docker cho phép tách các ứng dụng khỏi cơ sở hạ tầng để phân phối phần mềm một cách nhanh chóng. Với Docker, chúng ta có thể quản lý cơ sở hạ tầng của mình giống như cách tự quản lý các ứng dụng của bản thân. Từ đó, giảm đáng kể độ trễ giữa việc viết mã và chạy mã trong sản xuất. Nói dễ hiểu hơn thì docker là một máy ảo sử dụng tài nguyên chia sẻ với hệ điều hành OS.

Docker image là ảnh của một môi trường phát triển (có thể gọi là snapshot) - một file bất biến, không thay đổi. Nó gồm các source code, libraries, dependencies, tools và các files khác cần thiết cho một ứng dụng để chạy. Có thể tìm các docker image ở cộng đồng docker hub.  Khi đã có docker image, ta có thể khởi tạo các docker container từ docker image. Docker container là máy ảo chứa môi trường phần mềm.

Dockerfile là file chứa các câu lệnh đặc tả về môi trường thực thi phần mềm. Một số câu lệnh hỗ trợ bởi docker.

Docker compose là một công cụ để xác định và chạy multi-container các ứng dụng Docker. Với Compose sử dụng tệp YAML để định cấu hình các dịch vụ của ứng dụng. Sau đó, với một lệnh duy nhất có thể khởi động tất cả các dịch vụ từ cấu hình của mình.

* 1. **Redmine (Phần mềm quản lý task)**

Redmine là một công cụ theo dõi vấn đề và quản lý dự án dựa trên web và mã nguồn mở miễn phí. Nó cho phép người dùng quản lý nhiều dự án và các tiểu dự án liên quan. Nó có tính năng wiki và diễn đàn cho từng dự án, theo dõi thời gian và kiểm soát truy cập dựa trên vai trò linh hoạt.

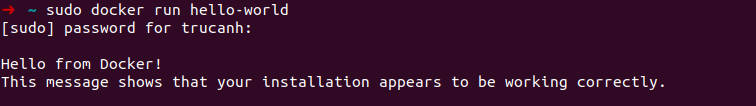
# **Chương 3. Các công việc triển khai tại đơn vị thực tập**

## **Mô tả tóm tắt vị trí công việc thực tập**

Vị trí thực tập: Thực tập sinh phát triển phần mềm tại trung tâm Hệ Thống Thông Tin, làm việc với restful API và phát triển hệ thống quản lý nhiệm vụ, nhân viên,...

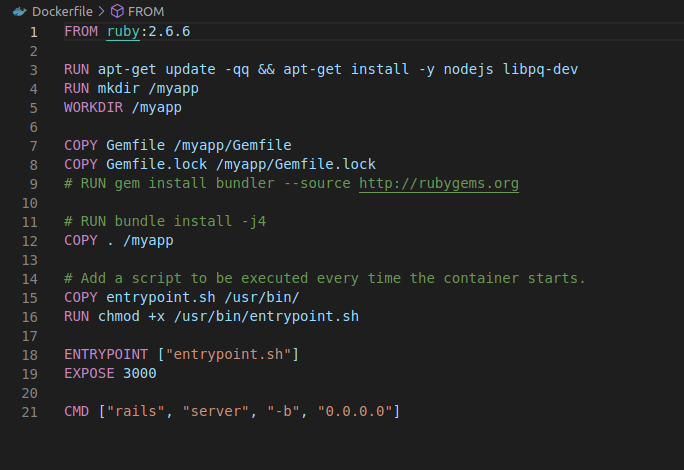
* **Các công việc thực tập:**
* Làm việc với open source Redmine:
  + Nắm rõ framework Ruby on Rails với trình soạn thảo mã Vs Code
  + Sử dụng docker trên hệ điều hành Ubuntu
  + Làm việc với hệ quản trị cơ sở dữ liệu Mysql
  + Biết sử dụng công cụ quản lí và các cách cấu hình, tùy chỉnh trên ứng dụng Redmine
  + Thay đổi, ghi đè các tính năng có sẵn hoặc tạo thêm thông qua các plugin được tạo
* Làm việc với thành viên trong công ty:
  + Tiếp nhận các nhiệm vụ được giao
  + Trao đổi khi có các thắc mắc trong công việc để hoàn thành các nhiệm vụ được giao
  + Nhận được các tài liệu liên quan đến Redmine, các nghiệp vụ liên quan đến công việc
  1. **Làm việc với mã nguồn mở Redmine**
     1. **Cài đặt môi trường làm việc**
* Cài đặt Docker trên Ubuntu:

Vào terminal của Ubuntu sử dụng câu lệnh ’sudo-apt-get update’ để tải thông tin gói từ tất cả các nguồn được cấu hình. Chạy câu lệnh ‘sudo apt-get install docker-ce docker-ce-cli containerd.io’ để cài đặt docker. Tiếp đến chạy container mang tên ‘hello-word’ để xác nhận việc cài đặt thành công.



Hình 3.1. Terminal tạo docker container ‘hello-word’

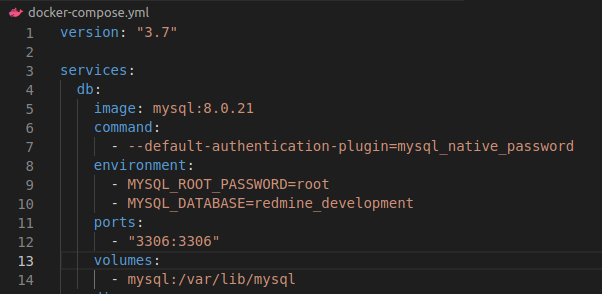
Sau khi tải về open source của redmine, tạo docker image thông qua dockerfile. Trong Dockerfile, ta tiến hành cài đặt các gói cần thiết cho việc cài ruby, mysql và rails.



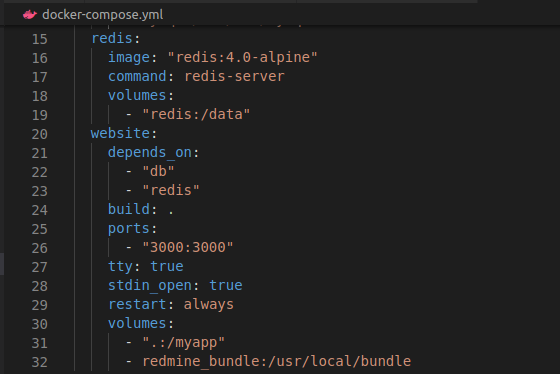
Hình 3.2. Cấu hình file Dockerfile

Môi trường sẽ được dựng trên môi trường đặc tả sau lệnh ’FROM’, ở đây là môi trường ruby 2.6.6. Các câu lệnh tiếp theo trong file lần lượt mô tả các package sẽ được cài đặt vào môi trường. Cuối cùng là câu lệnh thực thi khi container được dựng lệnh/khởi động.

Cấu hình các dịch vụ cần khởi chạy trong file ‘docker-compose.yml’. Khai báo MySQL server, Redis server và website trong file docker-compose này.



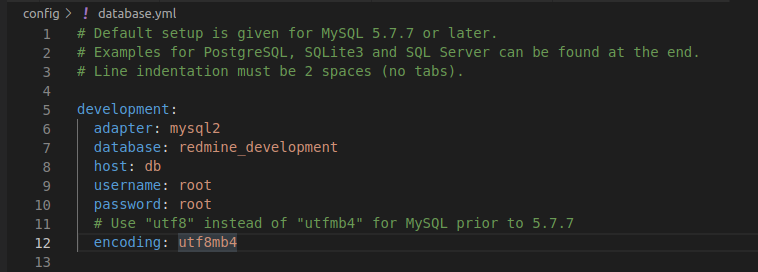
Hình 3.3. Cấu hình file docker-compose yaml



Hình 3.4. Cấu hình file docker-compose yaml

* **Nội dung file compose:**
* Sử dụng docker version ‘3.7’
* Về service Mysql:
* Image sử dụng là mysql
* Sử dụng cổng 3306 để export dữ liệu ra bên ngoài và cũng chính là cổng được sử dụng bên trong container
* Volumes chính là thư mục gắn kết(mount folder) bên ngoài với bên trong container như phần sử dụng command. Đường dẫn '/var/lib/mysql' là folder để lưu trữ dữ liệu bên ngoài container.
* Về Redis:
* Image được sử dụng là redis:4.0-alpine
* Volumns được mount tới thư mục ‘redis:/data’.
* Về website:
* Sử dụng cổng 3000 làm đầu ra
* Dữ liệu lưu tại thư mục ‘/usr/local/bundle’*.*

Vào file ‘database.yml’ của Redmine app để cấu hình lại host thành ‘db’, cập nhật username và password.



Hình 3.5. Cấu hình file database yaml

* + 1. **Cài đặt và khởi chạy Redmine**

Sau khi đã hoàn thành các bước cấu hình trên ta tiến hành build ứng dụng Redmine trên Docker.

Di chuyển đến thư mục chứa app Redmine bằng câu lệnh ’cd <folder>’, và thực hiện các bước dưới đây.

Bước 1: Tạo docker compose

docker-compose build

Bước 2: Cài đặt thư viện của Rails

docker-compose run website bunlde install

Bước 3: Khởi tạo database

docker-compose run website bin/rake db:create

Bước 4: Chạy migration để tạo dữ liệu bảng và cột

docker-compose run website bin/rake db:migrate

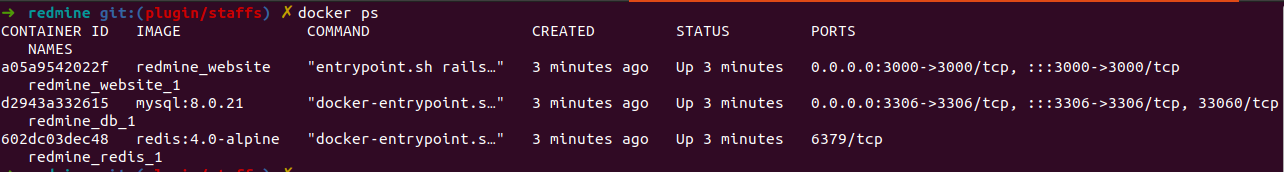
Bước 5: Tạo các dữ liệu mặc định

docker-compose run website bin/rails redmine:load\_default\_data

Bước 6: Mở các container để khởi chạy website

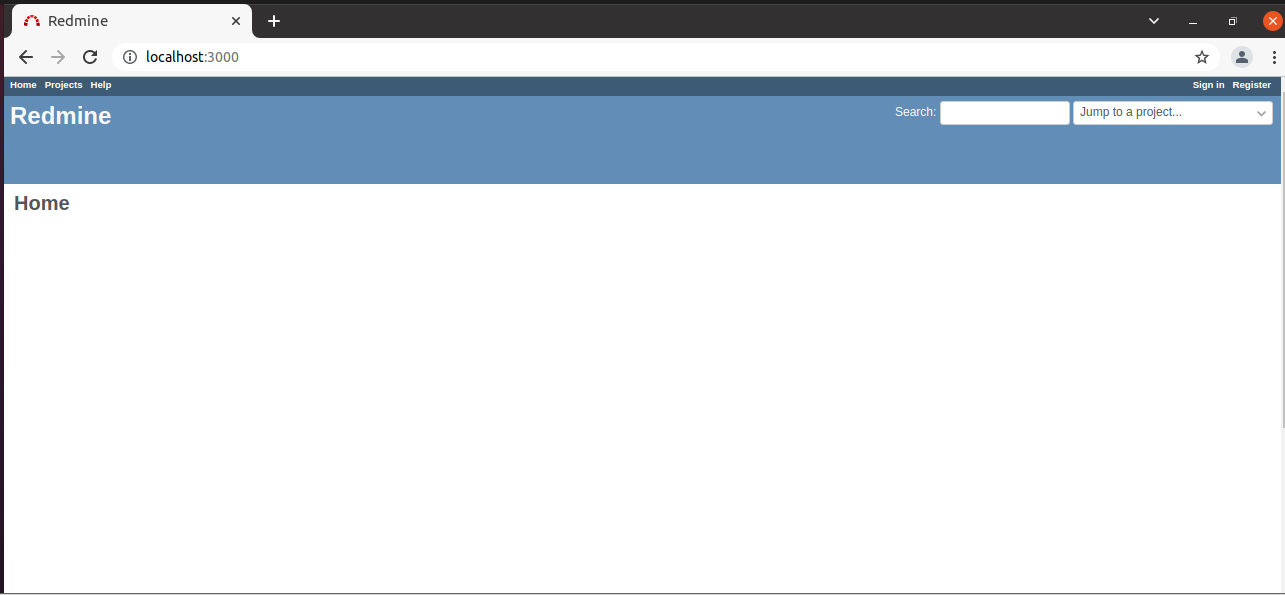
docker-compose up

Khởi chạy các container với câu lệnh docker-compose up. Để kiểm tra danh sách các container đang chạy với câu lệnh ’docker ps’.



Hình 3.6. Danh sách các container đang chạy

Sau khi đã chạy xong các lệnh trên , ta có thể chạy Redmine app với MySQL bằng Docker . Vào trình duyệt và gõ *‘http://localhost:3000/’* để kiểm tra. Từ đó ta thấy được giao diện trang chủ của Redmine.



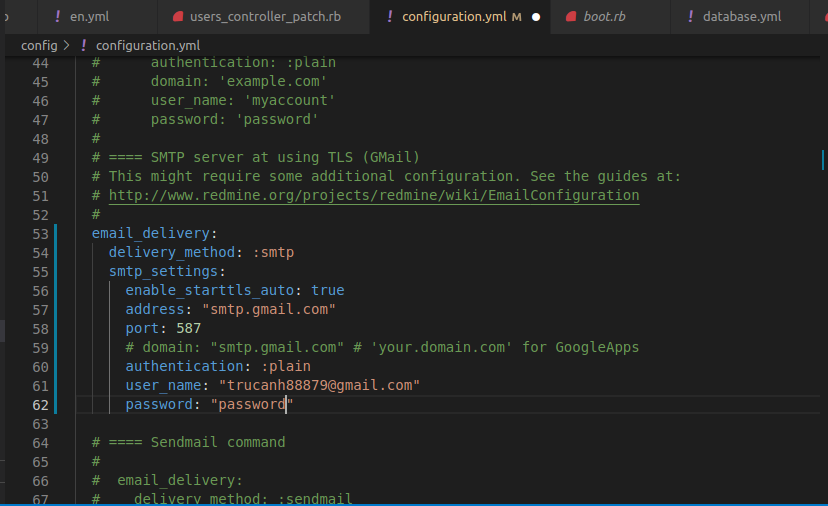
Hình 3.7. Giao diện trang chủ của Redmine

## **Làm việc với hệ thống Redmine**

### **Cấu hình email (smtp)**

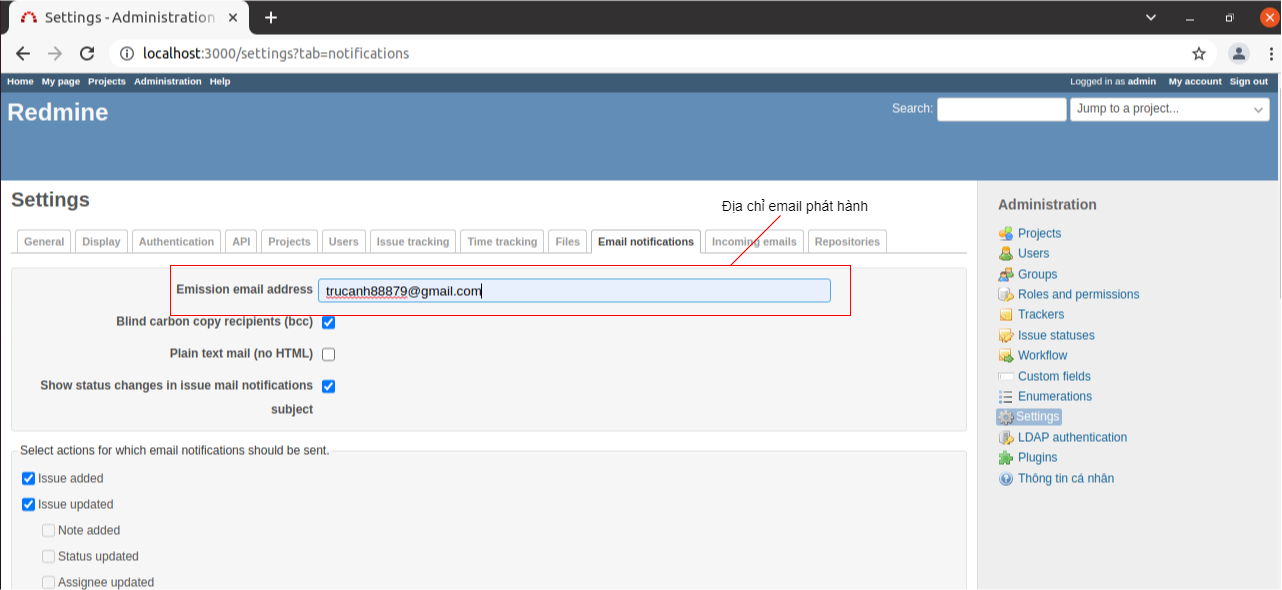
Action Mailer của Rails cho phép gửi email từ ứng dụng Redmine bằng cách sử dụng các Class và view.

* **Cách cấu hình action mailer trong Redmine:**
* Vào file ‘configuration.yml’  thay đổi một số cấu hình mặc định của Redmine bằng cấu hình mới
* Với delivery\_method, cấu hình smtp được chỉ định với từ khóa smtp\_setting



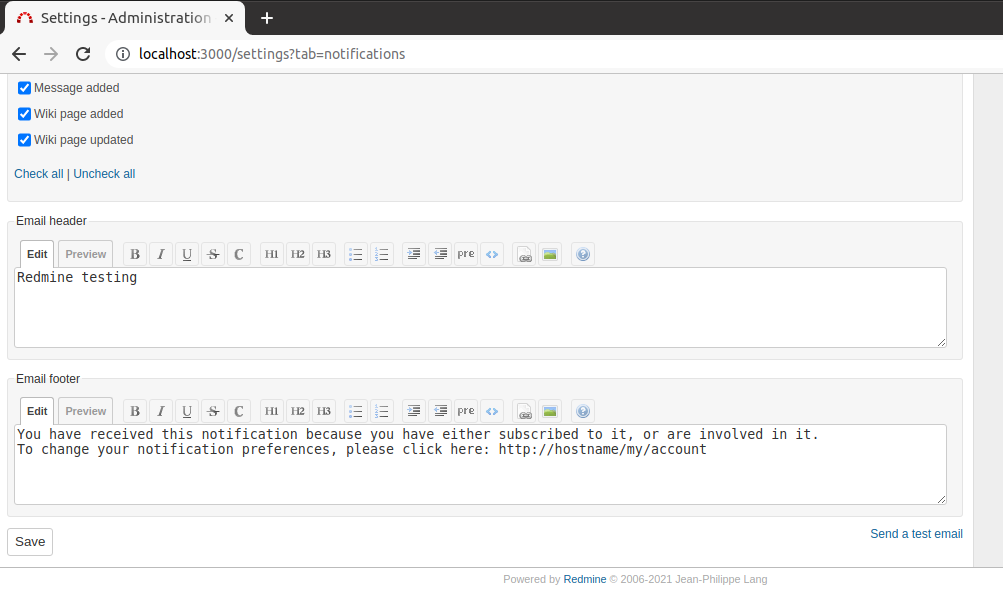
Hình 3.8. Cấu hình email ở file configuration yaml

Ngoài ra, để thay đổi định dạng cho email có thể vào website Redmine, chọn **Administration ⏵ Setting ⏵ Email notifications** để tùy chỉnh.



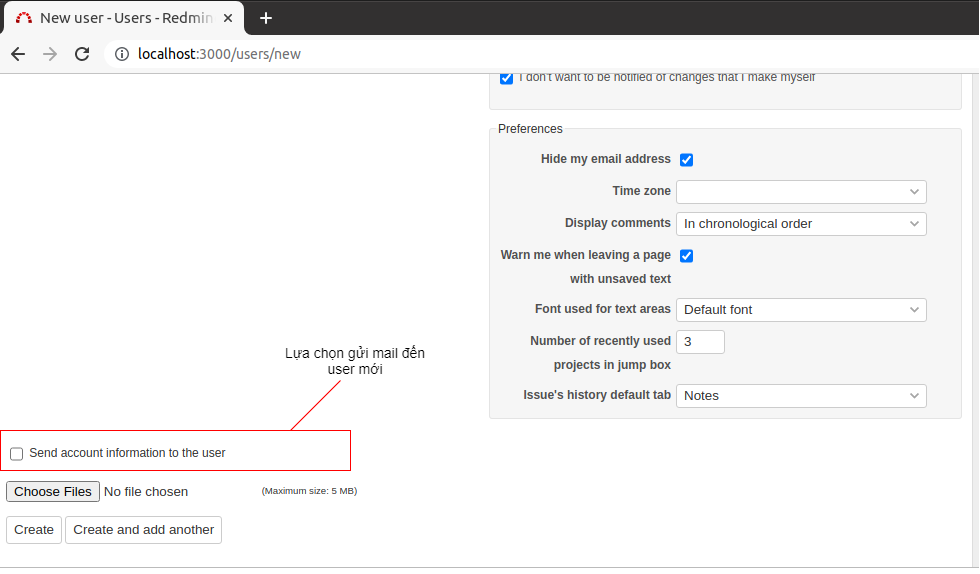
Hình 3.9. Giao diện email notifications

Thông qua tùy chỉnh email cho phép chúng ta thay đổi bố cục của phần đầu cuối của email sẽ được gửi đi.



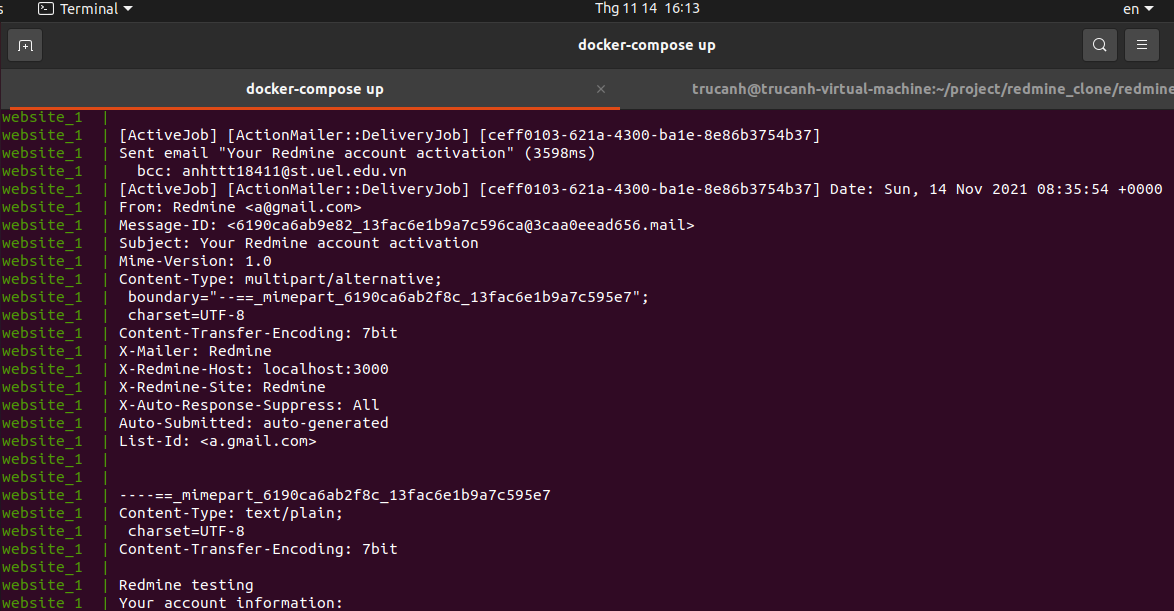
Hình 3.10. Giao diện email notifications

Việc cấu hình email cho phép Redmine gửi email đến các user trong khi tạo mới hay cập nhật các vấn đề cần giải quyết... Ví dụ trong trường hợp tạo mới một user, khi admin lựa chọn gửi thông báo đến user, ngay lập tức Redmine sẽ gửi đến email của user đã được lưu trước đó.

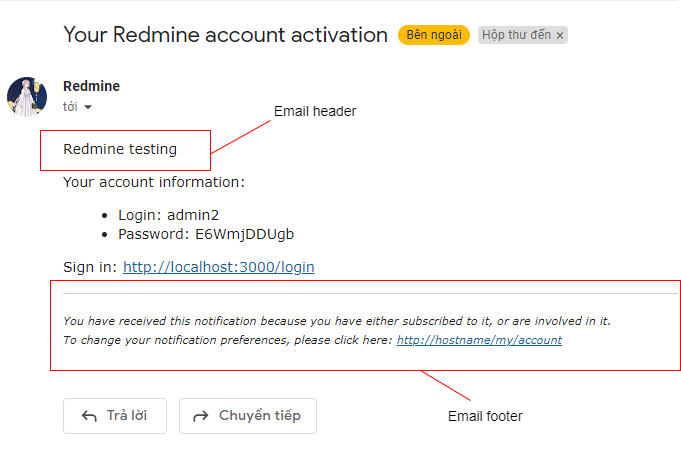


Hình 3.11. Giao diện tạo mới User

ActionMailer sẽ được gọi đến khi ta nhấn nút Create để tạo mới User. Nó chứa toàn bộ thông tin của Email được gửi đi. Gồm thông tin người nhận, ngày gửi đi, chủ đề,…



Hình 3.12. Terminal Action Mailer

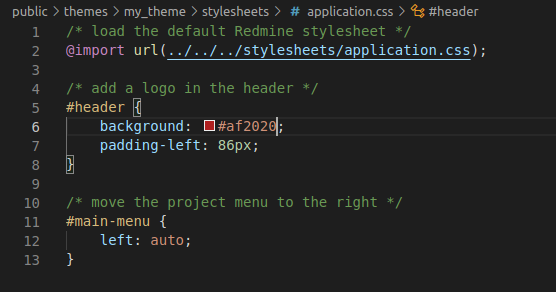


Hình 3.13. Nội dung mail được gửi đến User

### **Thêm chủ đề (theme) cho Redmine**

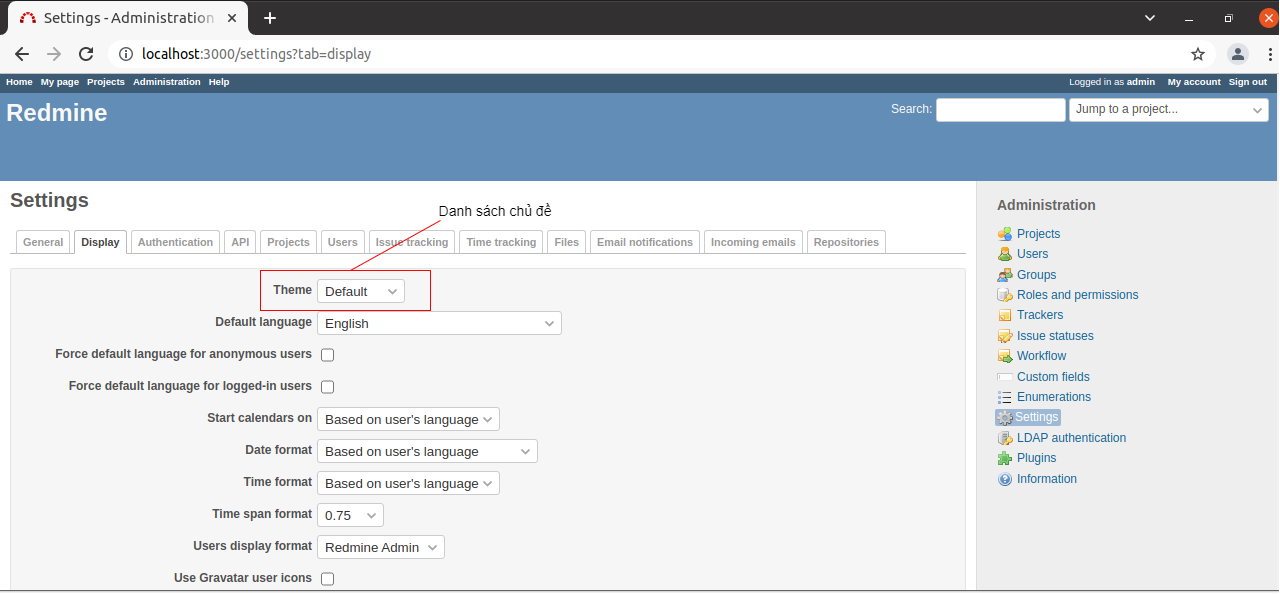
Redmine cung cấp hỗ trợ cơ bản cho các theme. Có thể ghi đè theme bởi các biểu định kiểu web (stylesheets) với file application.css và thêm một số javascript tùy chỉnh.

* Tạo một thư mục trong public/themes. Tên thư mục sẽ được sử dụng làm tên chủ đề. Ví dụ như ‘public/themes/my\_theme’
* Trong thư mục my\_theme đó tạo application.css tùy chỉnh và lưu nó trong một thư mục có tên là stylesheets ‘public/themes/my\_theme/stylesheets/application.css’
* Từ đó, chúng ta có thể tạo ra các stylesheet tùy chỉnh để ghi đè các thuộc tính trước đó.



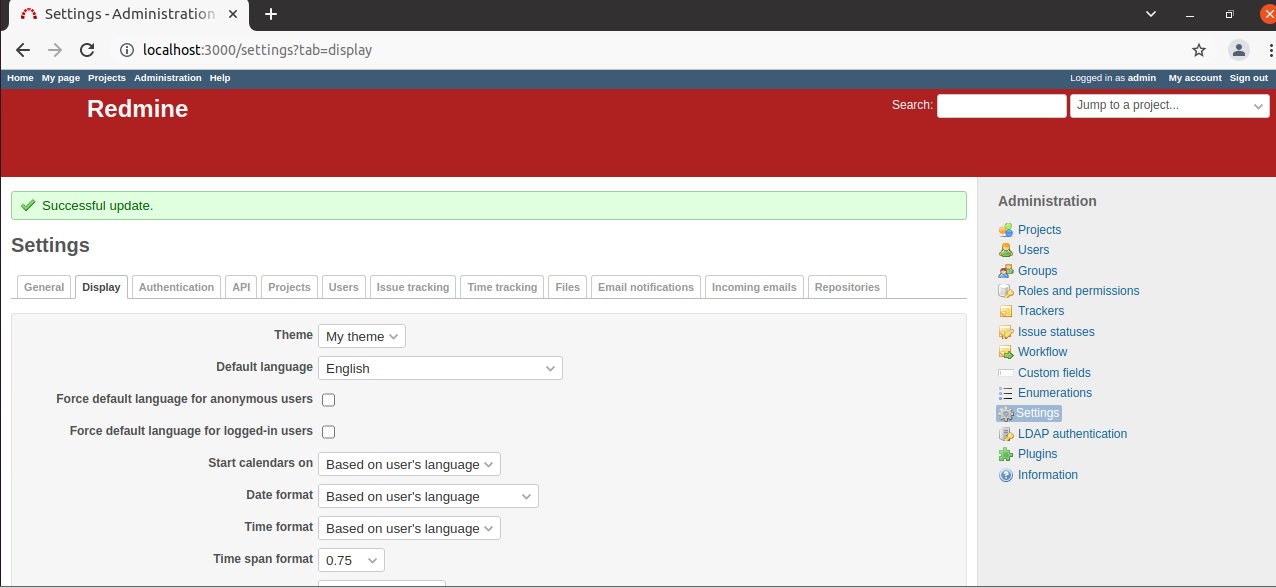
Hình 3.14. Thêm themes cho Redmine

Áp dụng chủ đề: đi đến **Administration⏵Setting⏵Display** và chọn chủ đề mới tạo trong danh sách thả xuống "Theme".



Hình 3.15. Giao diện điều chỉnh hiển thị

Lưu cài đặt và Redmine bây giờ sẽ được hiển thị bằng cách sử dụng chủ đề tùy chỉnh.



Hình 3.16. Giao diện thay đổi khi sử dụng chủ đề tùy chỉnh

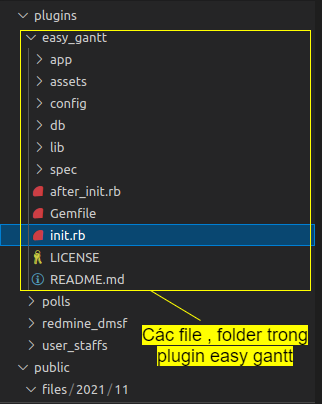
### **Cài đặt các plugin cho redmine**

Chúng ta có thể tìm kiếm các plugin Redmine và các nhà phát triển plugin cũng có thể đăng ký plugin của riêng họ bằng tài khoản Redmine. Thông tin chung về các plugin và cách cài đặt vào Redmine có thể tìm thấy tại [*Plugins Directory*](https://www.redmine.org/plugins).

#### **Plugin Easy Gantt**

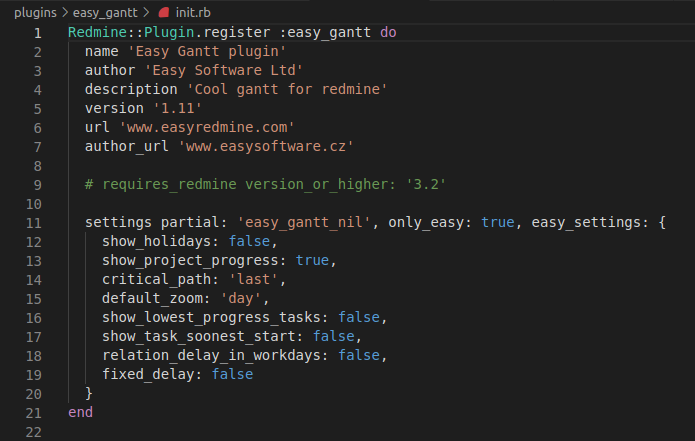
Mục đích: Easy Gantt giúp các nhà quản lý có thể quản lý dự án với biểu đồ Gantt. Thông qua biểu đồ Gantt chúng ta có thể hình dung được dự án để có thể lập kế hoạch, xác định các giải pháp và giải quyết các tác vụ dễ dàng hơn.

* **Cài đặt plugin Easy Gantt:**



Hình 3.17. Các file, folder của Easy Gantt trong plugin

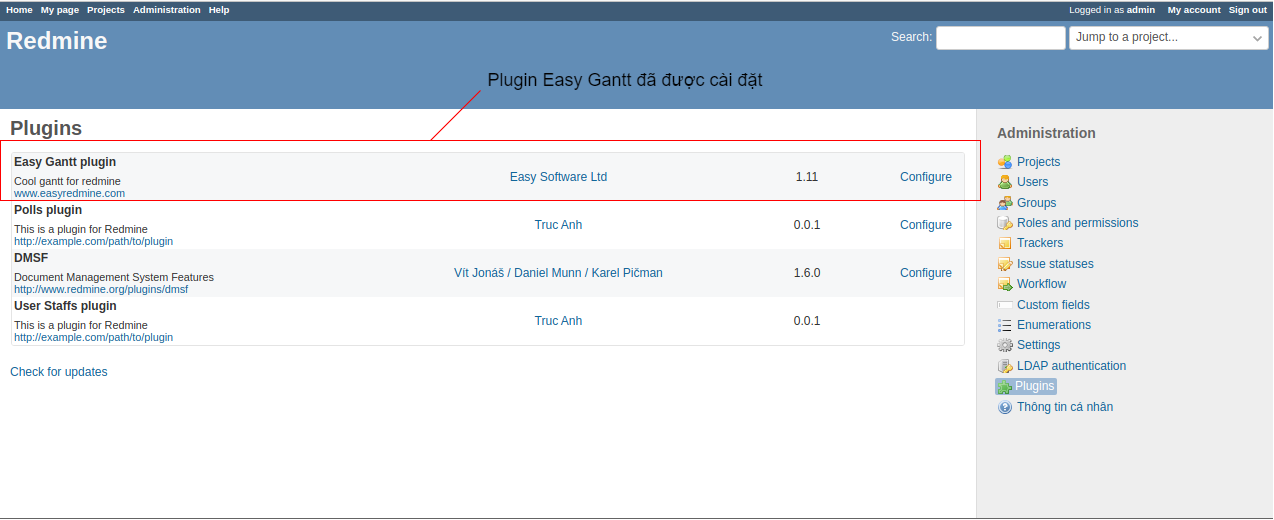
Thông tin của Plugin sẽ được cấu hình tại file ‘init.rb’ trong thư mục của easy\_gantt.



Hình 3.18. File init.rb của plugin Easy Gantt

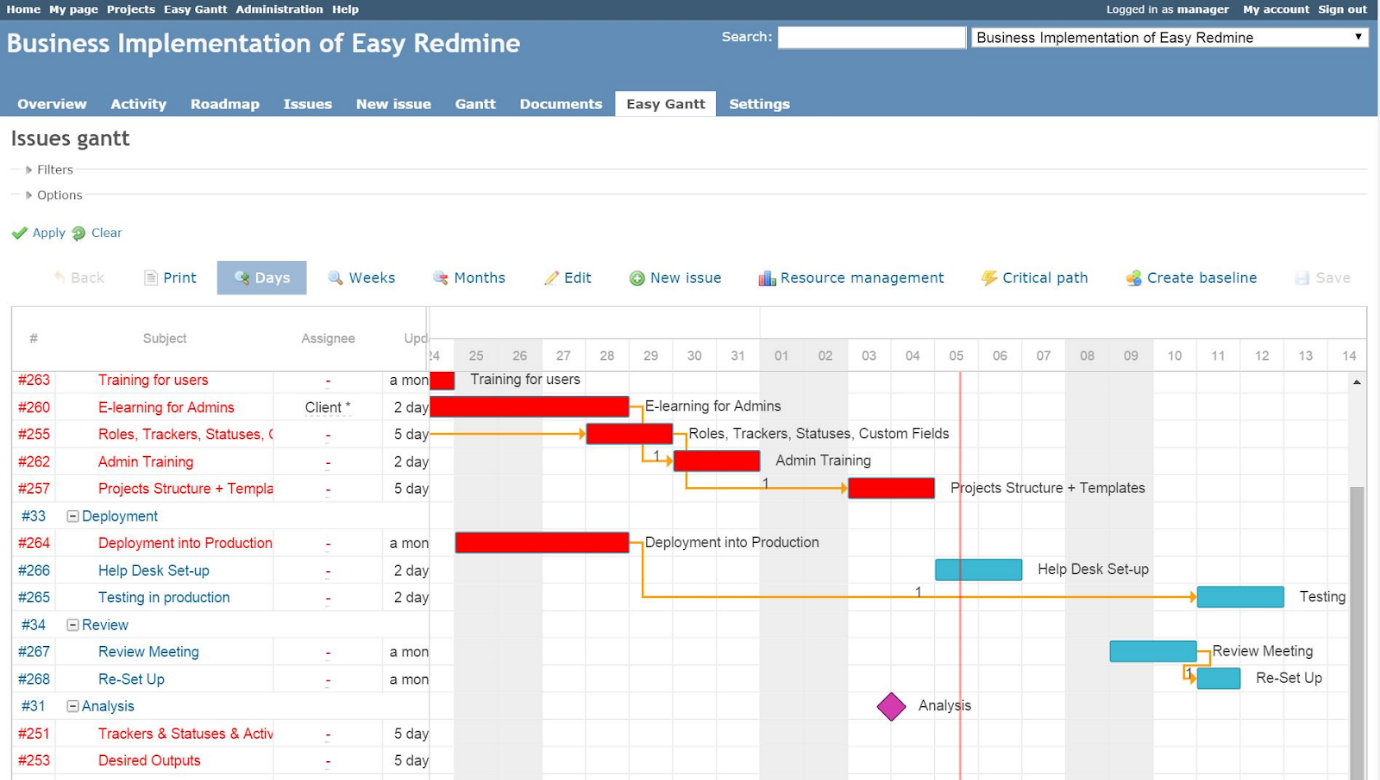
Để plugin có thể được hoạt động, trước tiên phải tạo các bảng dữ liệu được định nghĩa tại thư mục ‘db’. Migration là một tính năng quan trọng của Active Record cho phép người dùng chỉnh sửa cấu trúc dữ liệu mà không cần vào trong database. Để tạo migration cho plugin Easy Gantt chạy câu lệnh ‘docker-compose run website bundle exec rake redmine:plugins:migrate’.

Sau khi chạy câu lệnh ‘docker-compose up’ để khởi chạy các container, vào website trong **Administration⏵Plugins**  có thể thấy các thông số cấu hình tại file ‘init.rb’ hiển thị trên giao diện này.



Hình 3.19. Thông tin plugin Easy Gantt hiển thị ở Redmine

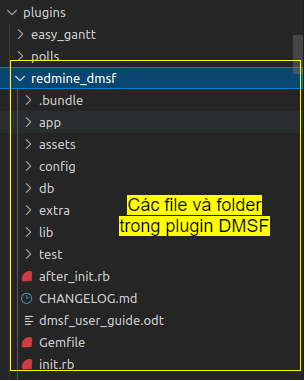
Để sử dụng plugin Easy Gantt vào Projects và chọn dự án có thể sử dụng được Easy Gantt.



Hình 3.20. Giao diện của Easy Gantt

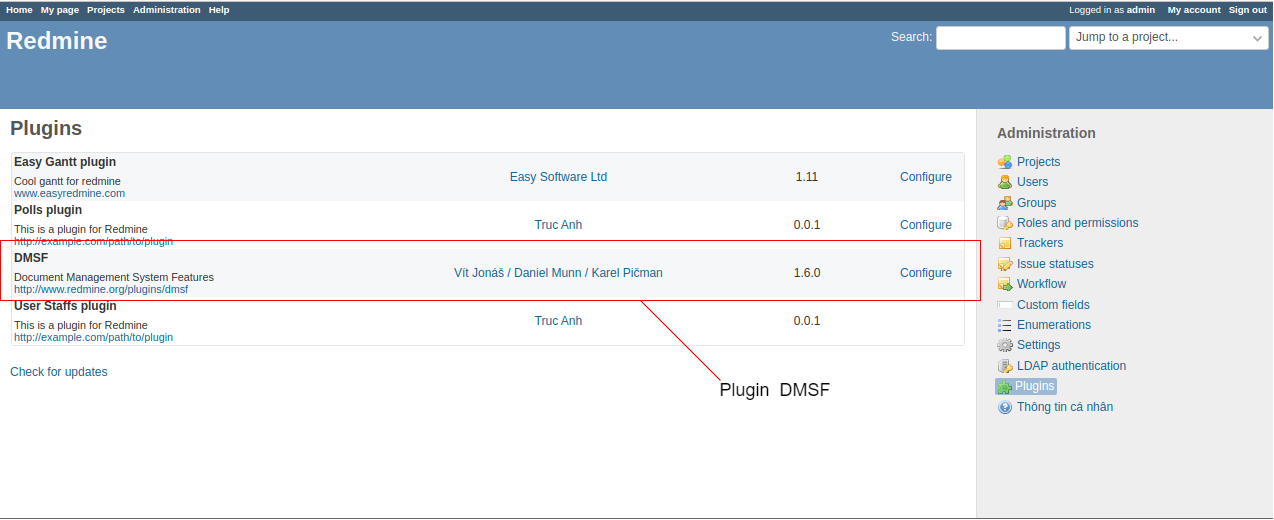
#### **Plugin DMSF(Document Management System Features)**

* **Mục đích, tính năng:**
* Cấu trúc thư mục
* Phiên bản tài liệu/lịch sử sửa đổi
* Thông báo qua email cho các thư mục và/hoặc tài liệu
* Khóa tài liệu
* Nhiều (kéo/thả tùy thuộc vào trình duyệt) tải lên/tải xuống
* Tải xuống nhiều lần qua zip
* Tài liệu trực tiếp hoặc liên kết tài liệu gửi qua email
* Quy trình phê duyệt tài liệu có thể định cấu hình
* Kiểm tra quyền truy cập tài liệu
* Tích hợp với nguồn cấp dữ liệu hoạt động của Redmine
* Macro Wiki để liên kết nội dung nhanh chóng
* Chức năng WebDAV đọc/ghi đầy đủ
* Tìm kiếm toàn văn nội dung tài liệu tùy chọn
* Các liên kết tượng trưng của tài liệu và thư mục
* Trường tùy chỉnh tài liệu
* Thùng rác
* Rest API
* **Cài đặt plugin DMSF:**



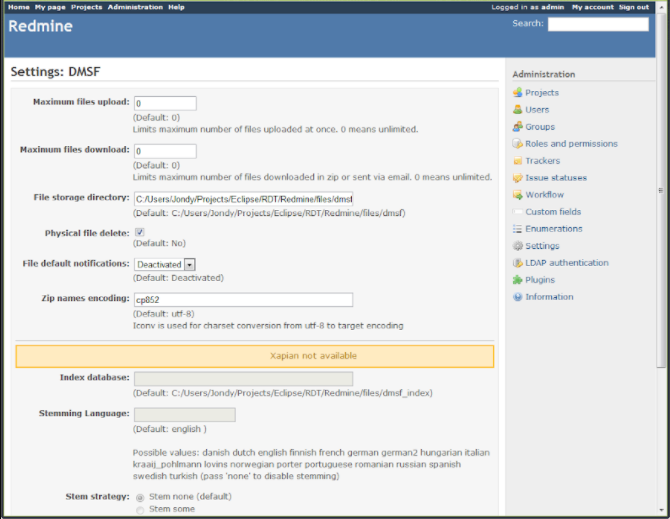
Hình 3.21. File init.rb của plugin DMSF

Chạy migration cho plugin DMSF với câu lệnh ‘docker-compose run website bundle exec rake redmine:plugins:migrate’.Vào website kiểm tra plugin DMSF đã được cài đặt.



Hình 3.22. Thông tin plugin DMSF

Sử dụng plugin DMSF vào **Project⏵DMSF**



Hình 3.23. Giao diện plugin DMSF

## **Xây dựng plugin cho Redmine**

### **Tạo mới plugin**

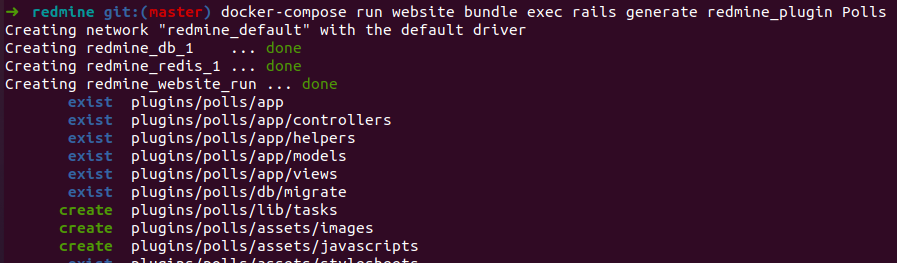
Tạo một plugin mới có thể được thực hiện bằng cách sử dụng trình tạo plugin Redmine.

Cú pháp cho trình tạo này là:

bundle exec rails generate redmine\_plugin <plugin\_name>

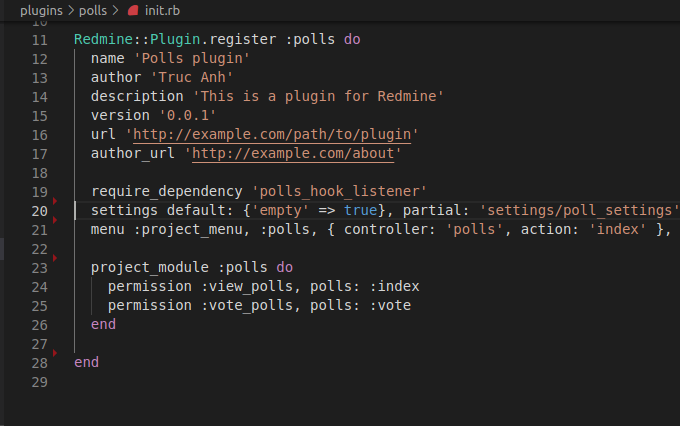
Di chuyển vào thư mục redmine, sau đó thực hiện lệnh sau:

docker-compose run website bundle exec rails generate redmine\_plugin Polls



Hình 3.24. Terminal khởi tạo plugin

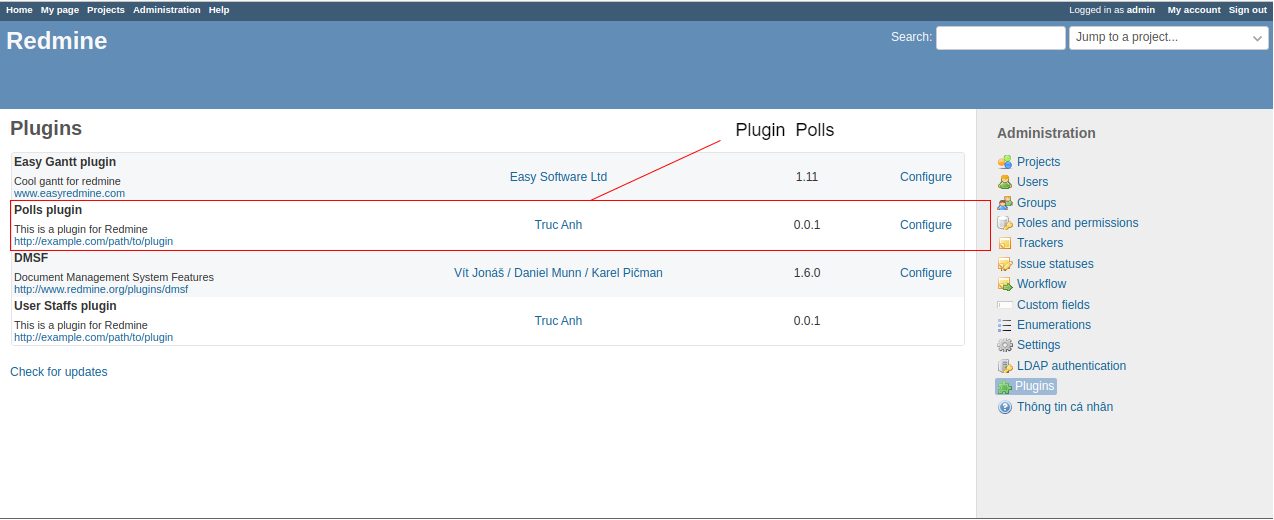
Cấu trúc plugin được tạo trong ‘plugins/polls’. Chỉnh sửa ‘plugins/polls/ init.rb’ để điều chỉnh thông tin plugin (tên, tác giả, mô tả và phiên bản).



Hình 3.25. File init.rb của plugin Polls

Sau đó khởi động lại ứng dụng và trỏ trình duyệt của ta đến *‘http: // localhost: 3000/admin/plugins’*.

Sau khi đăng nhập, ta sẽ thấy plugin mới của mình trong danh sách plugin:



Hình 3.26. Thông tin của plugin Polls

Bất kỳ thay đổi nào đối với tệp init.rb của plugin đều yêu cầu khởi động lại ứng dụng vì nó không được tải lại theo mỗi yêu cầu.

Bất cứ ứng dụng Rails nào cũng có 3 phần model, view và controller, Redmine cũng không ngoại lệ.

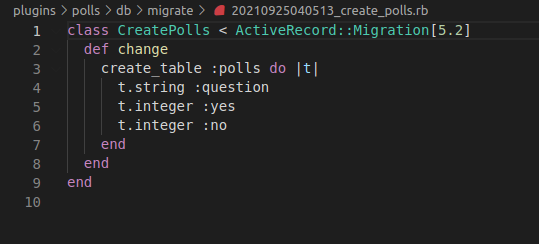
Ở bước tiếp theo chúng ta cần khởi tạo model. Hiện tại plugin không lưu trữ bất cứ thứ gì. Ta tiến hành tạo model khảo sát ý kiến  ​​đơn giản cho plugin Polls. Cú pháp là:

bundle exec rails generate redmine\_plugin\_model <plugin\_name> <model\_name> [field[:type][:index] field[:type][:index] ...]

https://lh4.googleusercontent.com/3I4c7AnoXulELShGU4lxDSuiN5-hR4fIj2lnS0GdZGhHZRbJeMn_XqRzoinQ8YzYWO6qFtehocBIH1ta28JmVm665ugjGwMPl9s2RpwSP-EsiP_2CwDd0lPtCQaYVCu8YYDjf3UJ

Hình 3.27. Terminal tạo model Polls

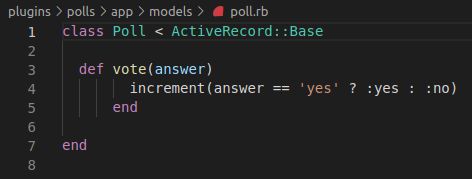
Điều này tạo ra một Poll Model ​​và file migration ứng ‘20210925040513\_create\_ polls’ trong ‘plugin/polls/db/migrate’.



Hình 3.28. File migrate của Polls

Say khi điều chỉnh file migrate (ví dụ: thay đổi giá trị mặc định ...) sau đó chạy migrate cơ sở dữ liệu bằng lệnh  ‘docker-compose run website bundle exec rake redmine:plugins:migrate’.

Chỉnh sửa ‘plugins/polls/app/models/poll.rb’ trong thư mục plugin của ta để thêm phương thức #vote sẽ được gọi từ controller.



Hình 3.29. Phương thức vote ở model

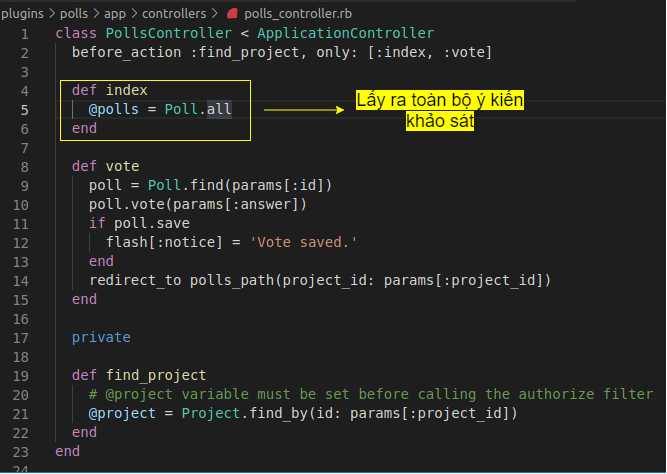
Ở thời điểm hiện tại, plugin không làm được gì cả. Vì vậy, ta cần tạo một controller cho plugin Polls. Redmine cho phép chúng ta có thể sử dụng trình tạo controller để thực hiện điều đó. Cú pháp là:

bundle exec rails generate redmine\_plugin\_controller <plugin\_name> <controller\_name> <actions>

https://lh6.googleusercontent.com/uOS2mGA1NXkFORcFoWGVJRbfzmfWiQryeH_lerZeMkmueBEsonCoxpBgzOovqOt5xLIZa50sqUVaM4iPU0c704aCohNaKfp1cMgpiFYG9eZGhrvwBes_R3osUnRcwCeNIerMIDhB

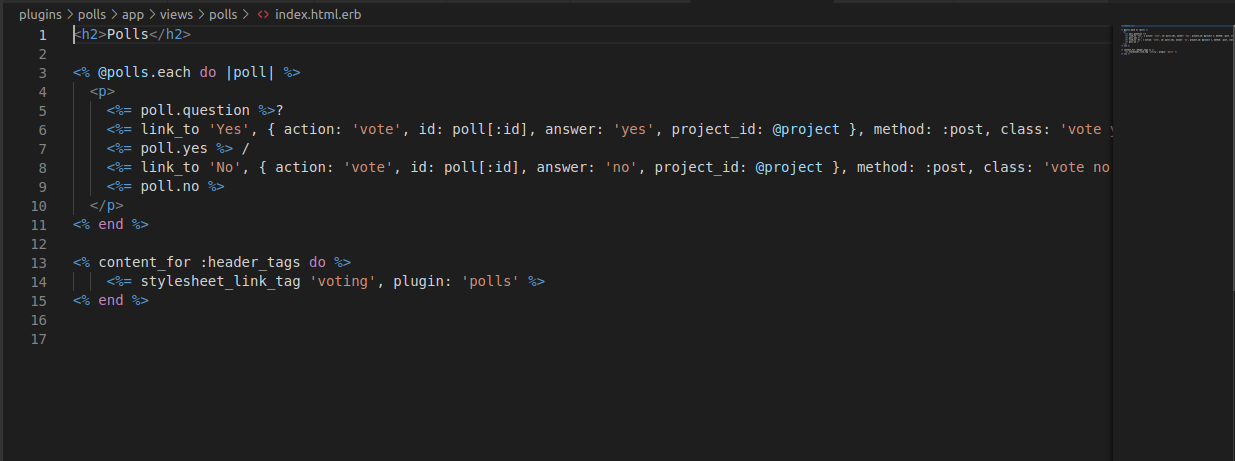
Hình 3.30. Khởi tạo controller Polls

Controller PollsController với 2 hành động (#index và #vote) được tạo. Vào file ‘plugins/polls/app/controllers/polls\_controller.rb’ để chỉnh sửa 2 hành động này.



Hình 3.31. Các phương thức ở file polls\_controller.rb

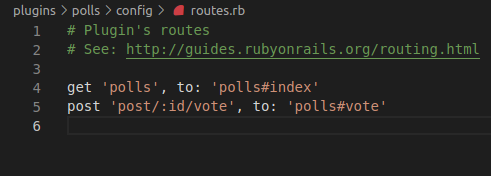
Sau đó, chỉnh sửa ‘plugins/polls/app/views/polls/index.html.erb’ sẽ hiển thị các cuộc thăm dò hiện có:



Hình 3.32. View hiển thị các cuộc thăm dò

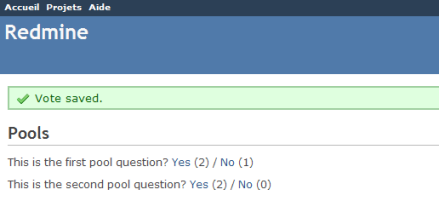
Routes đảm nhiệm vai trò cầu nối giao tiếp giữa thao tác người dùng với website. Khi HTTP request được gửi đến server, routes cho biết nó sẽ chạy ở controller nào với action nào.

Redmine không cung cấp các kí tự đại diện mặc định của route (': controller /: action /: id'). Các plugin phải khai báo các route thích hợp trong file ‘config/routes.rb’. Vì vậy, để thêm 2 route cho 2 hành động ta vào chỉnh sửa file ‘plugins/polls/config/route.rb’.



Hình 3.33. Định tuyến cho plugin Polls

Bây giờ, khởi động lại ứng dụng và trỏ trình duyệt và đến *‘*[*http://localhost:3000/ polls*](http://localhost:3000/%20%20%20%20%20polls)*’*. Chúng ta sẽ thấy 2 cuộc thăm dò và ta có thể bỏ phiếu cho chúng.

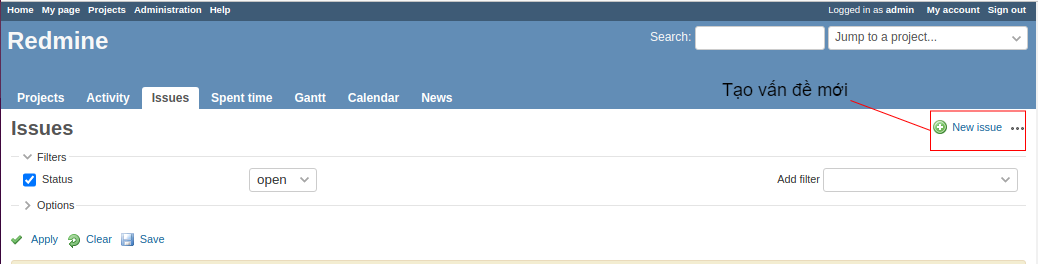


Hình 3.34. Giao diện của Polls

### **Tạo plugin ghi đè tính năng issues**

Issues (vấn đề) là một trong những tính năng phổ biến nhất trong Redmine. Một vấn đề có thể được thêm vào một dự án cụ thể. Người dùng có thể tạo vấn đề trong dự án nếu được cấp quyền (được quản lý bởi quản trị viên tại **Admistration⏵Roles and permissions**).

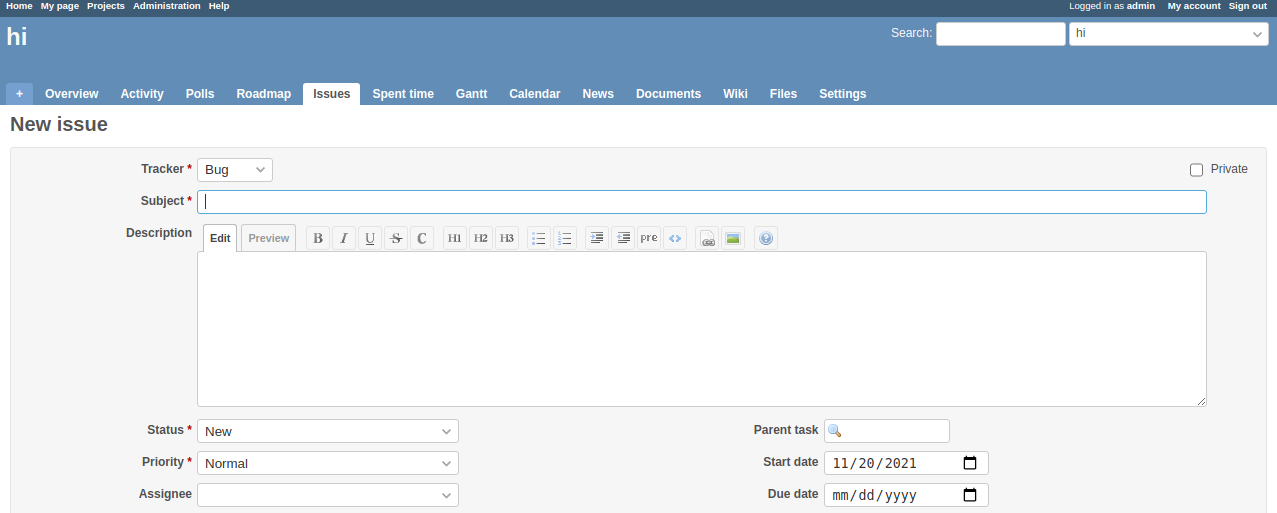
Để tạo một vấn đề mới, hãy chuyển đến Project. Chọn nhanh bằng cách click vào Issues.



Hình 3.35. Giao diện Issues Redmine

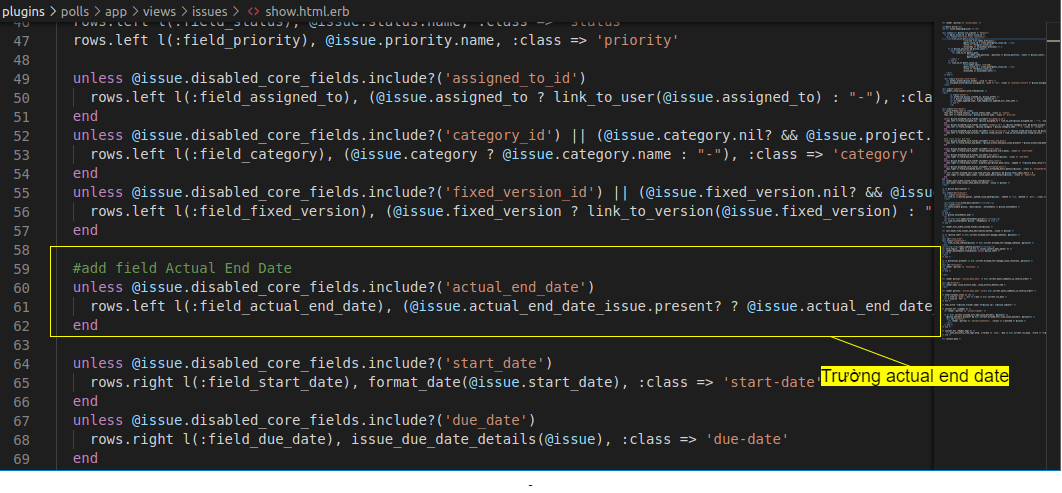
Hoặc có thể tạo issue thông qua các dự án.

* **Yêu cầu đặt ra**: thêm trường ‘actual end date’ cho Issues. Khi task đã được giải quyết thời gian sẽ được ghi nhận tại thời điểm đó. Trường ‘actual end date’ không thể thay đổi khi nhiệm vụ hoàn thành.
* **Thực hiện:**



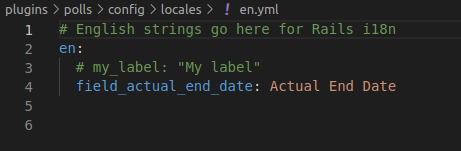
Hình 3.36. Giao diện tạo mới Issues

Việc ghi đè trong plugin giúp các nhà phát triển dễ quản lí mã nguồn và tiện lợi hơn trong việc bảo trì và sửa chữa. Ở plugin Polls ta tạo file ‘../views/issues/show.html.erb’ để có thể ghi đè giao diện Show của Issues. Ta thêm thuộc tính ‘actual\_end\_date\_issue’ cho đối tượng issue đã được tạo ra. Với điều kiện thuộc tính đó có tồn tại ở một đối tượng issue thì ngày kết thúc vấn đề sẽ được hiển thị ở giao diện Show.



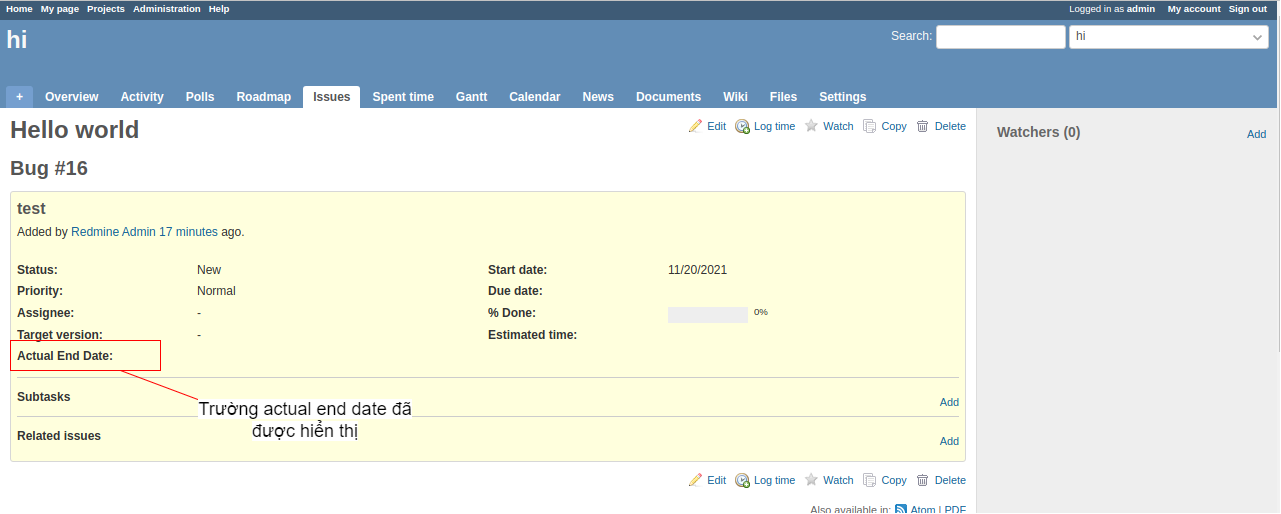
Hình 3.37. File show.html.erb của Issues

Để định nghĩa nhãn ‘field\_actual\_end\_date’ vào ‘locales/en.yml’ để cấu hình.



Hình 3.38. Chỉnh sửa file locale en.yml

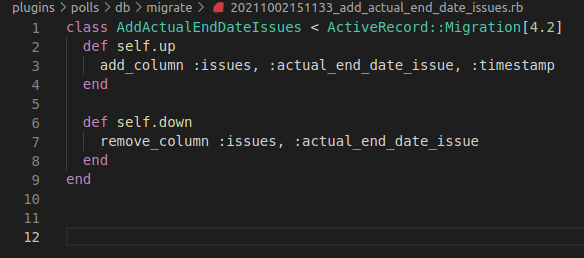
Khởi động lại ứng dụng tại màn hình hiển thị vấn đề trường actual end date đã được thêm vào.



Hình 3.39. Giao diện Show issues sau khi chỉnh sửa

* Các bước mà Redmine hiển thị ra view
  + Rails tiến hành tải Bootstrap và toàn bộ framework của nó
  + Rails bắt đầu tải mã trong plugin
  + Rails tìm thấy một thư mục view trong ‘../vendor/plugins/polls/app/views’ và chờ nó vào đường dẫn đến view.
  + Rails tải tất cả các plugin khác
  + Sau đó, Rails tải ứng dụng từ ‘../app’
  + Rails hoàn tất quá trình tải và phục vụ các yêu cầu
  + Yêu cầu gửi đến và view được hiển thị
  + Rails tìm kiếm một mẫu phù hợp và tải mẫu của plugin trong khi chờ đường dẫn của view
  + Rails trả về view cho plugin

Để có thể lưu trữ ngày hoàn thành vấn đề, cần phải tạo migration cho issues tạo cột ‘actual\_end\_date\_issue’ với câu lệnh ‘docker-compose run website bundle exec rails generate redmine\_plugin\_migration polls add\_actual\_end\_date\_issues ‘.



Hình 3.40. File migration add column vào Issues

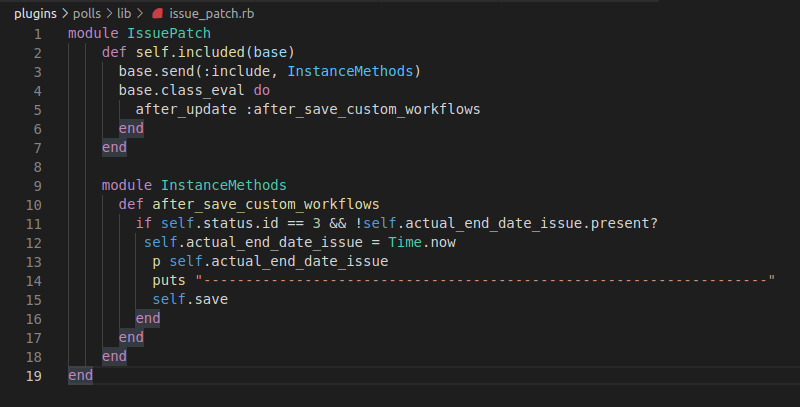
Thêm cột ‘actual\_end\_date\_issue’ trong bảng issues với kiểu dữ liệu là timestamp. Chạy migration để tạo cột trong database.

* Mở rộng Redmine Core, thay vì ghi đè Controller và Model, ta nên:

- Thêm các phương thức mới vào một Model/Controller

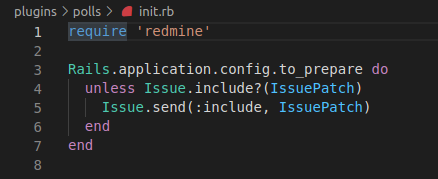
- Đóng gói một phương thức hiện có.

Để giải quyết vấn đề hiện tại ta sử dụng phương pháp thêm phương thức vào model. Tiến hành thêm một phương thức mới vào Issue có tên là ‘after\_save\_custom\_workflows’ và cũng khai báo một mối quan hệ.



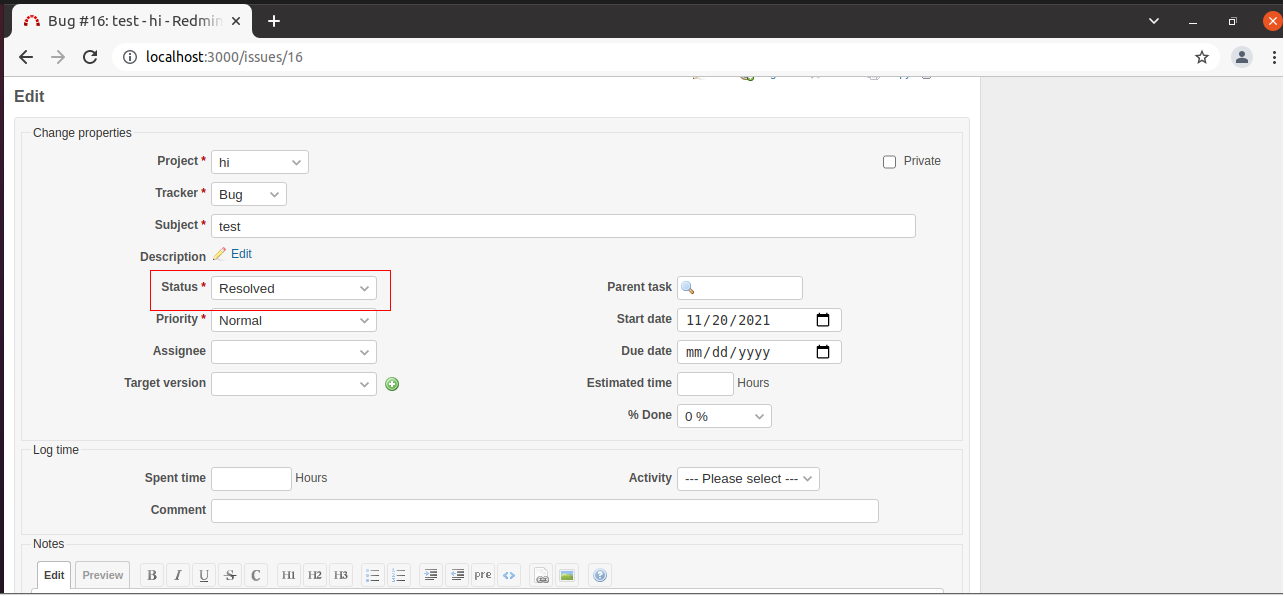
Hình 3.41. Module Issue Patch

Module IssuePatch vẫn chưa thể sử dụng. Tại ‘file init.rb’ phải thực hiện khai báo để đính kèm module IssuePatch.



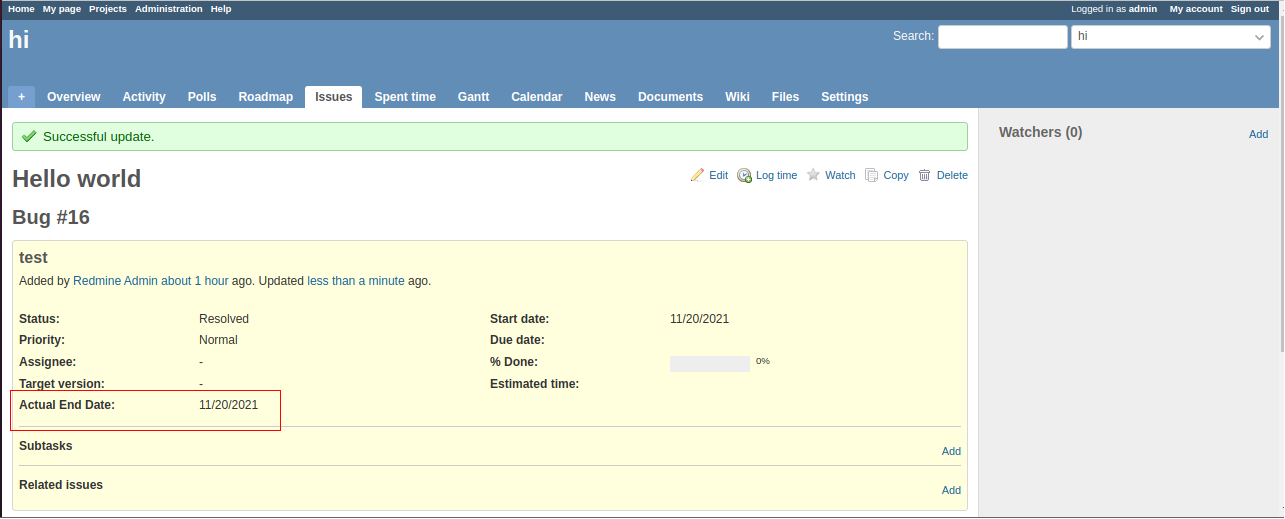
Hình 3.42. Khai báo Module ở file init.rb của plugin Polls

Restart ứng dụng. Vào trình edit issue chuyển status của issue từ New → Resolved



Hình 3.43. Giao diện edit Issue

Nhấn ‘Submit’ lưu chỉnh sửa. Redmine sẽ trả về màn hình của Issue đã được chỉnh sửa. Ngày hoàn thành issue đã được ghi nhận tại thời điểm đó.



Hình 3.44. Ngày hoàn thành issue hiển thị ở giao diện Show

### **Plugin quản lí staffs**

* **Yêu cầu đặt ra:** tạo tính năng quản lý staffs thông qua tính năng quản lý user của Redmine.
* **Phân tích Databse của Staffs:**

Bảng User đã được tạo bới Redmine, thế nên ta chỉ cần thêm một số cột cần thiết cho Staff.

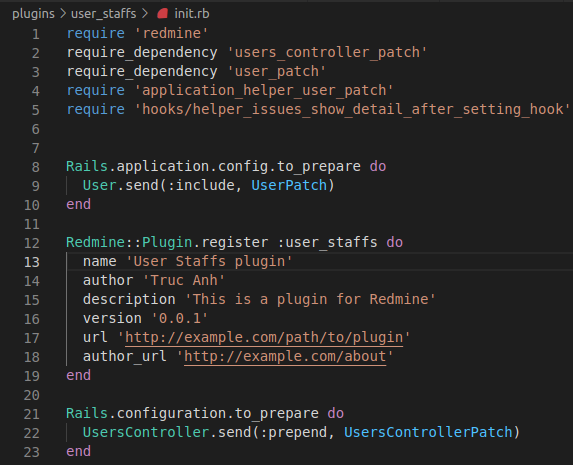
|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Mô tả |
| another\_email | varchar(50) NULL | Mail khác |
| phone | varchar(50) NULL | Số điện thoại |
| birthday | date NULL | Ngày sinh |
| sex | boolean NULL | Giới tính |
| active\_kpi | boolean NULL | Active KPI |
| active\_bug | boolean NULL | Active bug |
| team\_leader | INT(11) NULL | Id user team leader |
| type\_staff | INT(11) NULL | Loại staff |
| hardskill | varchar(255) NULL | Kỹ năng cứng |
| softskill | varchar(255) NULL | Kỹ năng mềm |
| achievement | varchar(255) NULL | Thành tựu |
| start\_date\_company | date NULL | Ngày bắt đầu vào công ty |
| start\_date\_contract | date NULL | Ngày bắt đầu hợp đồng |
| due\_date\_contract | date NULL | Ngày kết thúc hợp đồng |
| start\_date\_off | date NULL | Ngày bắt đầu nghỉ |
| end\_date\_of | date NULL | Ngày kết thúc nghỉ |
| place\_birth | varchar(50) NULL | Nơi sinh |
| permanent\_address | varchar(255) NULL | Thường trú |
| tempory\_address | varchar(255) NULL | Tạm trú |
| identity\_card | varchar(50) NULL | CMND |
| identity\_date | date NULL | Ngày cấp CMND |
| identity\_by | varchar(50) NULL | Nơi cấp CMND |
| ethnic | varchar(50) NULL | Dân tộc |
| contact | varchar(50) NULL | Người liên hệ (tên và số điện thoại) |
| note | varchar(255) NULL | Ghi chú |
| department\_id | INT(11) NULL | Id phòng ban |
| center\_id | INT(11) NULL | Id trung tâm |
| job\_positions\_id | INT(11) NULL | Id chức danh |
| location\_id | INT(11) NULL | Nơi làm việc (HN hoặc HCM) |
| contract\_id | INT(11) NULL | Id loại hợp đồng |
| work\_id | INT(11) NULL | Id trạng thái làm việc |

Bảng 3‑1 Bảng cấu trúc dữ liệu của staffs

* **Thực hiện:**

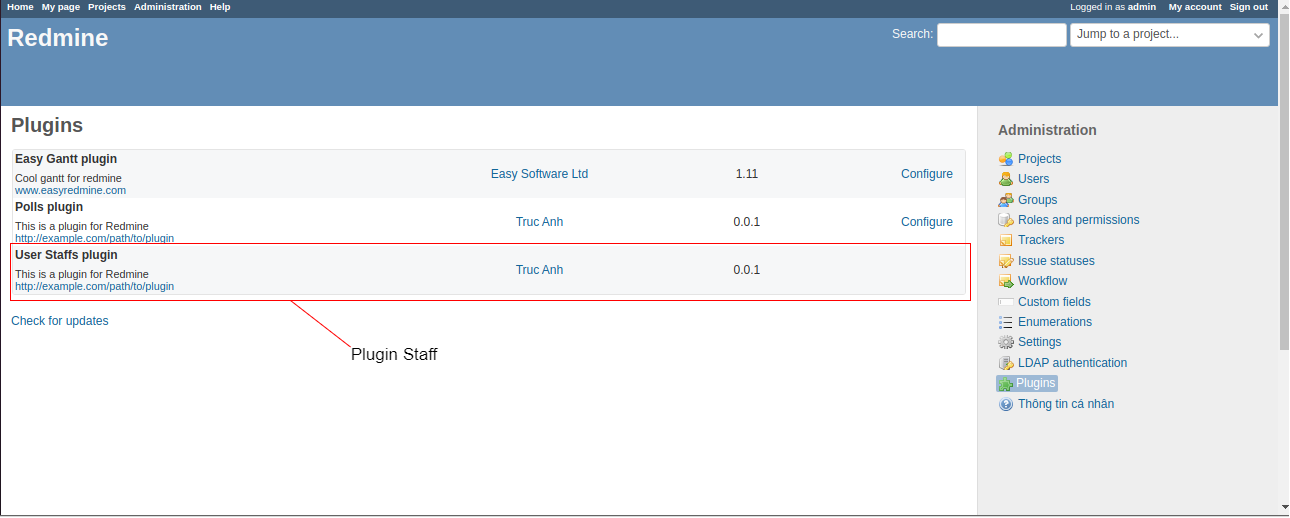
Ta tiến hành tạo một plugin mới có thể được thực hiện bằng cách sử dụng trình tạo plugin Redmine. Di chuyển vào thư mục redmine, sau đó thực hiện lệnh sau: ‘docker-compose run website bundle exec rails generate redmine\_plugin user\_staffs’.

Cấu trúc plugin được tạo trong ‘plugins/user\_staffs’. Chỉnh sửa ‘plugins/user\_staffs/init.rb’ để điều chỉnh thông tin plugin (tên, tác giả, mô tả và phiên bản).



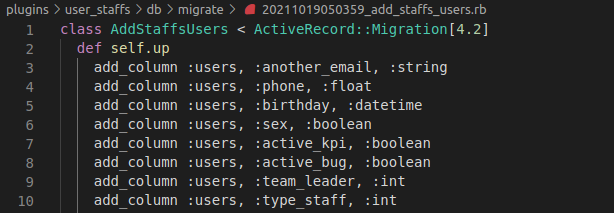
Hình 3.45. File init.rb của plugin Staffs

Sau đó khởi động lại ứng dụng với câu lệnh ‘docker-compose up’ và trỏ trình duyệt của ta đến *‘http: // localhost: 3000/admin/plugins’*. Đăng nhập, ta sẽ thấy plugin mới của mình trong danh sách plugin.



Hình 3.46. Thông tin plugin Staffs

Như phân tích database ở trên ta tiến hành tạo migration để thêm các cột cần thiết cho nhân viên với câu lệnh ‘docker-compose run website bundle exec rails generate redmine\_plugin\_migration user\_staff add\_staffs\_users’. Từ câu lệnh trên sinh ra một file tại folder migration có tên là ‘20211019050359\_add\_staffs\_users.rb’.

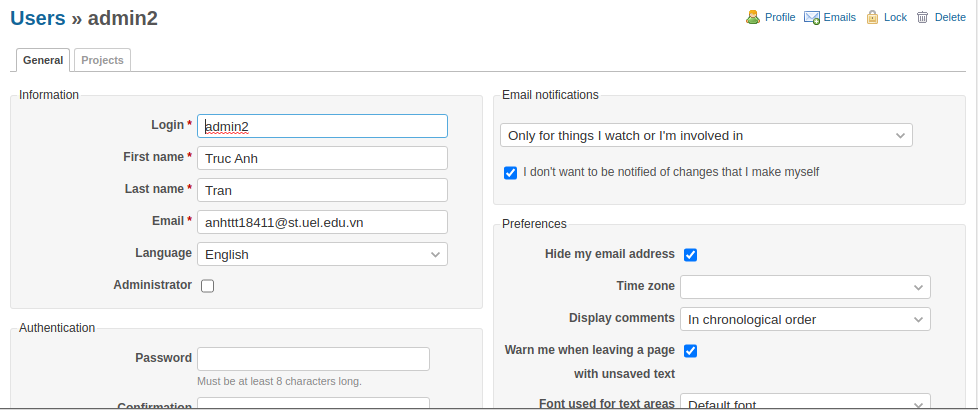


Hình 3.47. File migration add colum cho Users

Chạy migration để các cột có thể được lưu vào database. Dữ liệu đã có thể lưu về, ta tiếp tục các bước tiếp theo, đó là thêm các field cho tính năng quản lý user để từ đó ta có thể sử dụng nó với mục đích quản lí nhân viên.

Ta tiến hành xây dựng giao diện tạo mới nhân viên, các thao tác với dữ liệu đều thông qua form và các thành phần của nó. Rails helper cung cấp cho ta một phương thức là form\_with cho phép tạo ra đoạn mã html nhằm mục đích xây dựng thẻ ‘<form></form>’. Từ biến được đặt tên có thể sử dụng các phương thức FormHelper để tạo ra các thẻ input, checkbox,...Ví dụ tạo thẻ input text:

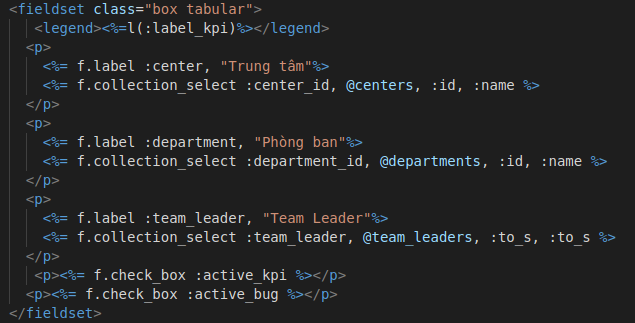
<%= f.text\_field :mail, :required => true %>



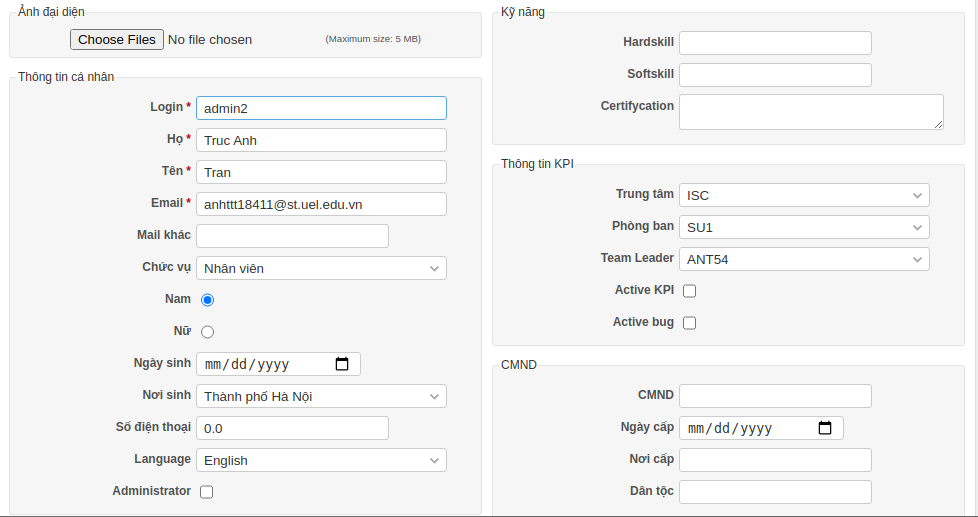
Hình 3.48. Giao diện edit User của Redmine

Redmine đã cung cấp cho ta tính năng quản lí user với các thuộc tính cơ bản đó là login, first name, last name, email, password,... Để vào trình quản lí user ta có thể vào **Administration⏵Users**. Lần lượt tạo field cho các thuộc tính đã thêm trong Active Record của user.

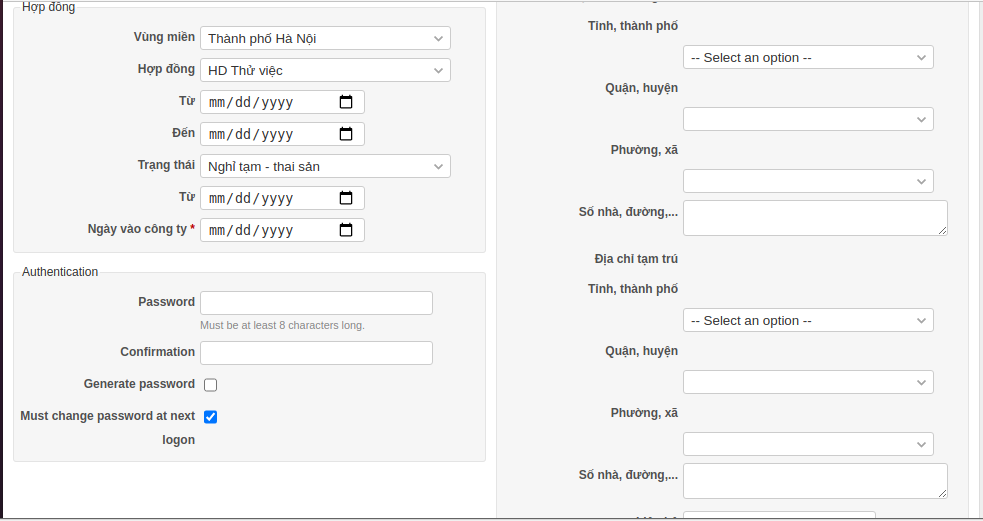
Ta chia ra các nhóm ảnh đại diện, thông tin cá nhân, hợp đồng, kỹ năng, thông tin KPI, CMND và xác thực. Nhóm thông tin cá nhân thêm các field giới tính, ngày sinh, chức danh, số điện thoại,... Nhóm hợp đồng ta thêm field thông tin hợp đồng, ngày vào công ty,... Nhóm kỹ năng có các kỹ năng cứng, mềm và các thành tựu đạt được. Ở nhóm thông tin KPI sẽ có thông tin trưởng nhóm,… Nhóm chứng minh nhân dân có số chứng minh, ngày cấp, nơi cấp, tạm trú và thường trú thêm vào đó là các thông tin liên hệ và ghi chú.



Hình 3.49. File new.html.erb của User được chia làm nhiều fieldset



Hình 3.50. Giao diện edit User sau khi được chỉnh sửa



Hình 3.51. Giao diện edit User sau khi được chỉnh sửa

Dữ liệu vẫn chưa thể được lưu, lựa chọn cách tạo thêm phương thức cho UsersController với module UsersControllerPatch. Để đính kèm module, vào file ‘init.rb’ của folder ‘plugins/user\_staffs’ khai báo với cú pháp:

require\_dependency 'users\_controller\_patch'

Thêm vào đó:

Rails.configuration.to\_prepare do

UsersController.send(:prepend, UsersControllerPatch)

end

Ta tạo thêm một file ‘lib/users\_controller\_patch.rb’. Để UsersController sử dụng module này, chỉ cần thêm dòng lệnh Ruby

require\_dependency 'users\_controller'

Cú pháp của module:

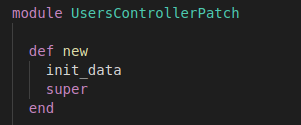
module Identifier

statement1

statement2

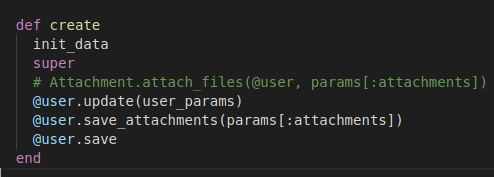
...........

end



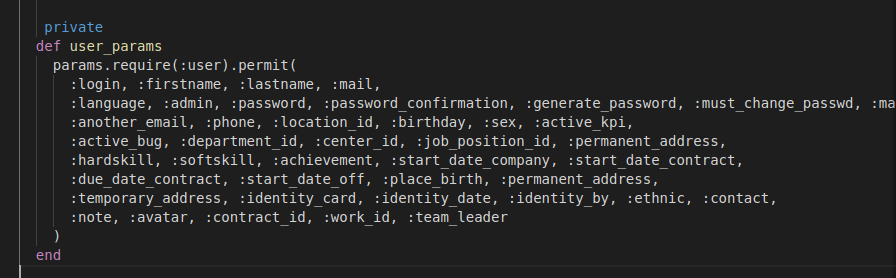
Hình 3.52. Phương thức new của module UserControllerPatch

Tạo ra phương thức new trong module. Khi ta gọi super trong Ruby tức là nó sẽ gọi đến một phương thức cùng tên ở lớp cha, ở đây lớp cha tức là UsersController với phương phức mà gọi super. Và từ đó nó sẽ kế thừa toàn bộ phương thức đó ở lớp cha. Ở giao diện tạo mới user khi nhấn nào nút create, form sẽ tự động submit lên server với HTTP request là post, dựa trên định tuyến chạy đến UsersController với action là create.



Hình 3.53. Phương thức create của module UserControllerPatch

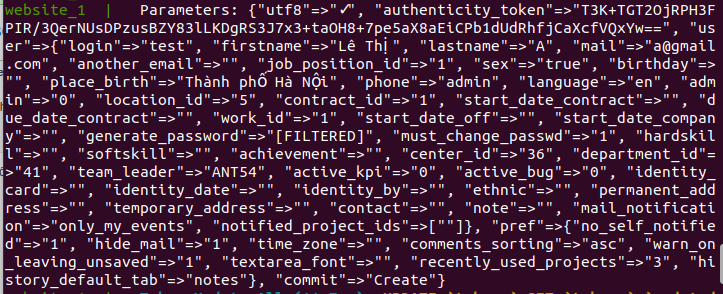
Khai báo phương thức private ta thêm phương thức có tên user\_params, khai báo các thuộc tính của user với Strong parameter. Strong parameters giống như 1 bộ lọc các giá trị được phép thay đổi và chỉ lưu các giá trị được cho phép.



Hình 3.54. Phương thức user\_params

* require(:user) bắt buộc trong params phải có key là user.
* permit(:login, :firstname, :lastname...) trong key user chỉ cho phép các thuộc tính được khai báo.

Khi nhấn nút create tạo user mới một parameters sẽ được truyền đi dưới dạng hash có key và value. Các thuộc tính chỉ được phép lưu khi khai báo Strong parameters.

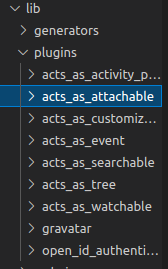


Hình 3.55. Parameters được truyền đi

Một tiện ích mở rộng rất phổ biến đã được triển khai trong Redmine là khả năng đính kèm tệp. Vì ta đang làm việc trên plugin, ảnh đại diện có thể được hiển thị bằng cách cho phép các tệp bên ngoài được đính kèm.

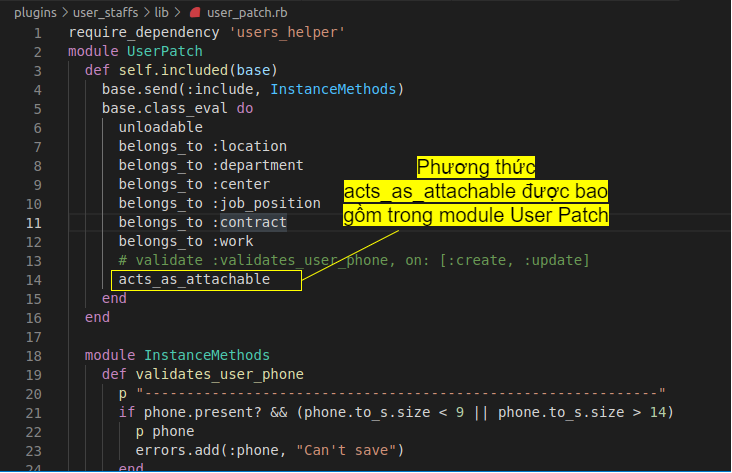
Redmine đã triển khai một số plugin nội bộ, nằm dưới ‘/path/to/redmine /lib/plugin’.

Các plugin này vẫn tuân theo các quy tắc của Rails. Một mô hình phổ biến trong các plugin là thêm một phương thức có tên là ‘act\_as\_something’ vào các model, điều này ngụ ý rằng một phương thức cấp lớp sẽ được bao gồm và được đặt tên giống như plugin.



Hình 3.56. Các plugin nội bộ của Redmine

Bằng cách bao gồm ‘act\_as\_attachable’ trong lớp của User, một liên kết 1-n được thiết lập và số lượng các phương thức thể hiện được tự động đưa vào lớp.



Hình 3.57. Module UserPatch

Cho phép các tệp đính kèm được hiển thị ở View ta sử dụng phương pháp xây dựng biểu mẫu bằng FormBuilder.

<div class="box">

<p>

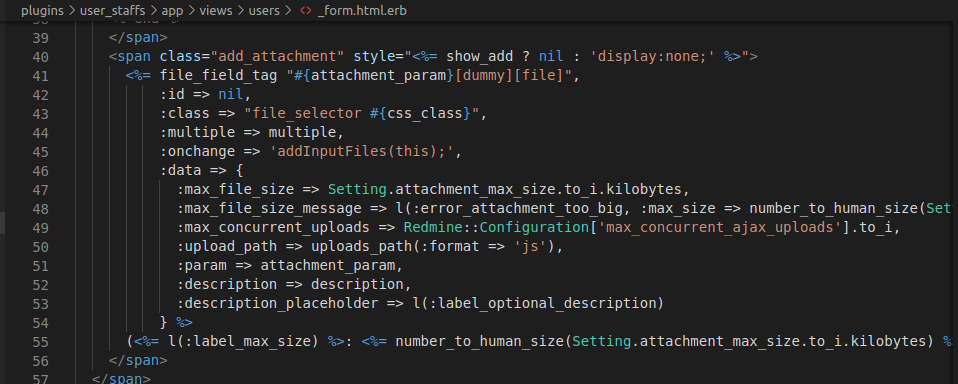
<label><%=l(:label\_attachment\_plural)%></label>

<%= render :partial => 'attachments/form' %>

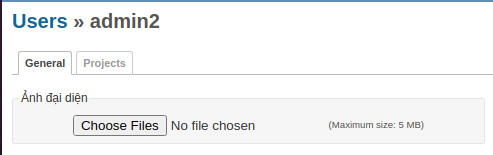
</p>

</div>

View Redmine sẽ cho phép các tệp được tải lên không đồng bộ và được gắn vào Model của User. Field đính kèm hiển thị thông qua file\_field\_tag.

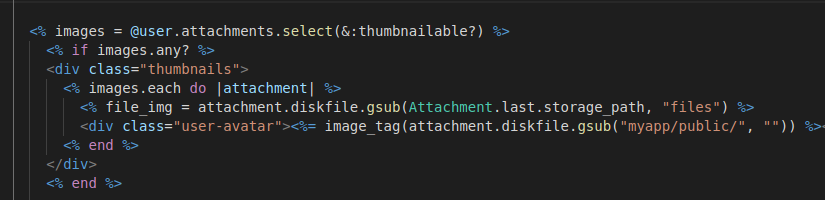


Hình 3.58. Thẻ file\_field\_tag được thêm ở view

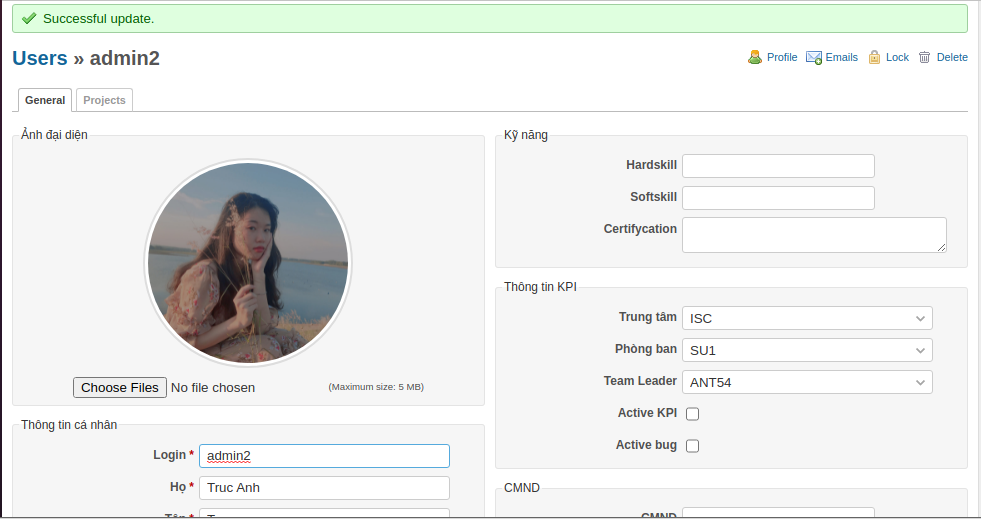


Hình 3.59. Giao diện edit User

Lưu attachment ta dùng phương thức có sẵn trong plugin ‘acts\_at\_attachable’ là hàm ‘save\_attachment’ với đối số truyền vào là ‘params[:attachment]’. Hiển thị ảnh đại diện ta dùng thẻ ‘image\_tag’ của Rails, thông qua thuộc tính diskfile của đối tượng attachment lấy ra tên của ảnh cần hiển thị.



Hình 3.60. Thẻ image\_tag được thêm ở view



Hình 3.61. Ảnh đại diện được hiển thị

**Chương 4. Đánh giá công việc**

* 1. **Kết quả thu được**
* Xây dựng được tính năng quản lý Staff
* Biết thêm các nghiệp vụ quản lý của Redmine
* Có thêm kiến thức về docker và redis, một phần củng cố kiến thức nền tảng về lập trình Ruby On Rails
* Học hỏi được quy trình làm việc, môi trường làm việc chuyên nghiệp ở công ty.
  1. **Thuận lợi**
* Đã có kiến thức nền tảng về lập trình Ruby on rails và Github
* Công ty tạo điều kiện để nhân viên học hỏi, người hướng dẫn giàu kinh nghiệm, tận tâm hỗ trợ khi gặp khó khăn, cung cấp đầy đủ các tài liệu liên quan
* Sự học hỏi không ngừng để giải quyết các nhiệm vụ được giao
* Nhận được sự hỗ trợ nhanh chóng từ các phòng ban khi có nghiệp vụ liên quan.
  1. **Khó khăn**
* Không có nhiều kinh nghiệm thực chiến, tác phong làm việc chuyên nghiệp.
* Dịch bệnh diễn biến phức tạp, sự hạn chế của các thiết bị viễn thông làm giảm hiệu quả công việc, không có sự trao đổi trực tiếp
* Thiếu kiến thức chuyên môn, nền tảng chưa vững vàng, cần sự trau dồi học hỏi không ngừng
* Vẫn còn nhiều sai sót chưa được khắc phục, chưa thực hiện được tất cả yêu cầu được giao do
* Việc đọc hiểu, tìm kiếm tài liệu hạn chế dẫn đến việc chậm trễ trong công việc.

**Chương 5. Tổng kết**

* 1. **Tổng kết những ưu điểm**
* Công ty làm việc chuyên nghiệp có quy trình nghiệp vụ rõ ràng
* Công việc được phân công riêng biệt cho từng phòng ban, mỗi phòng ban đều đảm nhận một nhiệm vụ khác nhau nhưng vẫn gắn kết với nhau để hoàn thành nhiệm vụ
* Sẵn sàng lắng nghe ý kiến nhân viên, mong muốn nhân viên gắn bó lâu dài với công ty, tạo môi trường làm việc thoải mái
* Chú trọng trong việc xây dựng các nền tảng xã hội, trao đổi nội bộ góp phần phát triển các cộng đồng chung ngày một lớn mạnh.
  1. **Tổng kết những nhược điểm**
* Áp lực công việc cao khi làm việc với một hệ thống lớn
* Quy mô lớn, số lượng lớn nhân viên gây khó khăn trong công tác quản lí nhân viên
* Chi phí đầu tư ban đầu cao, công việc bị đình trệ khi hệ thống gặp vấn đề.
  1. **Đề xuất những giải quyết**
* Nắm bắt công nghệ kịp thời, áp dụng kỹ thuật khoa học tiến để có thể tăng năng suất lao động, giảm giá thành sản phẩm, phục vụ khách hàng tốt hơn
* Giảm bớt khối lượng công việc cho nhân viên nhưng vẫn giữ nguyên được năng suất, tăng hiệu quả làm việc.
  1. **Kinh nghiệm rút ra sau khi thực tập**

Sau những ngay thực tập ở vị trí Software Developer, người thực hiện báo cáo đã có thể xây dựng cho mình một tác phong làm việc chuyên nghiệp hơn. Ngoài ra còn trau dồi cho mình được rất nhiều kiến thức, đặc biệt là các kiến thức chuyên môn.

Về kiến thức chuyên môn, người thực hiện báo cáo đã có thêm một số kiến thức về Docker, cách xây dựng một plugin Redmine trên nền tảng Rails, cách sử dụng và triển khai các dự án trên Redmine. Thêm vào đó là khả năng tìm kiếm tài liệu, đọc hiểu Tiếng Anh dần được cải thiện. Từ đó, kiến thức nền tảng về lập trình Ruby On Rails cũng được gia tăng đáng kể.

Bên cạnh các kiên thức chuyên môn là các kỹ năng mềm. Người báo cáo có thể cải thiện các kỹ năng giao tiếp, chú trọng các từ khóa khi trao đổi với người hướng dẫn để có thể hoàn thành nhiệm vụ một các nhanh chóng.

**Tài liệu tham khảo:**

[**[1]**](https://news.timviec.com.vn/cong-ty-co-phan-fpt-la-gi-gioi-thieu-lich-su-hinh-thanh-va-phat-trien-65160.html) Thanh Hằng, [Giới thiệu về FPT](https://news.timviec.com.vn/cong-ty-co-phan-fpt-la-gi-gioi-thieu-lich-su-hinh-thanh-va-phat-trien-65160.html)

[**[2]**](https://www.redmine.org/projects/redmine/wiki/plugin_tutorial) Jean-Philippe Lang, [Plugin Tutorial](https://www.redmine.org/projects/redmine/wiki/plugin_tutorial)

[**[3]**](https://docs.docker.com/get-started/) Docker Inc., [Docker Docs](https://docs.docker.com/get-started/)

[**[4]**](https://guides.rubyonrails.org/index.html) David Heinemeier Hansson, [Rails Guides](https://guides.rubyonrails.org/index.html)

[**[5]**](https://vi.wikipedia.org/wiki/Trang_Ch%C3%ADnh) Wikimedia Foudation, [Wikipedia](https://vi.wikipedia.org/wiki/Trang_Ch%C3%ADnh)

[**[6]**](https://chungta.vn/) Tập đoàn FPT, [Trang nội bộ của Tập đoàn FPT](https://chungta.vn/)

[**[7]**](https://www.redmine.org/)  Jean-Philippe Lang, [Redmine](https://www.redmine.org/)

**[8]** Alex Bevilacqua, Sách Redmine Plugin Extension and Development, Packt Publishing Ltd, 2014