

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

-----🙞🙜🕮🙞🙜-----



**ĐOÀN XUÂN HÙNG : 15110219**

**PHẠM MINH KẾ : 15110226**

Đề Tài:

**TÌM HIỂU LARAVEL FRAMEWORK VÀ XÂY DỰNG ỨNG DỤNG MINH HỌA**

**(WEBSITE TRẮC NGHIỆM TRỰC TUYẾN)**

**TIỂU LUẬN CHUYÊN NGÀNH KỸ SƯ CNTT**

**GIÁO VIÊN HƯỚNG DẪN**

**ThS. ĐẶNG THỊ KIM GIAO**

**KHÓA 2015 - 2019**



**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

-----🙞🙜🕮🙞🙜-----



**ĐOÀN XUÂN HÙNG : 15110219**

**PHẠM MINH KẾ : 15110226**

Đề Tài:

**TÌM HIỂU LARAVEL FRAMEWORK VÀ XÂY DỰNG ỨNG DỤNG MINH HỌA**

**(WEBSITE TRẮC NGHIỆM TRỰC TUYẾN)**

**TIỂU LUẬN CHUYÊN NGÀNH KỸ SƯ CNTT**

**GIÁO VIÊN HƯỚNG DẪN**

**ThS. ĐẶNG THỊ KIM GIAO**

**KHÓA 2015 - 2019**

**ĐH SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CNTT**

**\*\*\*\*\*\*\***

**XẢ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc lập – Tự do – Hạnh phúc**

**\*\*\*\*\*\*\***

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

Họ và tên Sinh viên 1: Đoàn Xuân Hùng MSSV 1: 15110219

Họ và tên Sinh viên 2: Phạm Minh Kế MSSV 2: 15110226

Ngành: Công Nghệ Thông Tin

Tên đề tài: Tìm hiểu Laravel framework và xây dựng ứng dụng minh họa (Website trắc nghiệm trực tuyến).

Họ và tên Giáo viên hướng dẫn: Đặng Thị Kim Giao

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện

* Lý thuyết: tìm hiểu Laravel Framework.
* Cài đặt: xây dựng website thi trắc nghiệm trực tuyến.

1. Ưu điểm

* Thực hiện đầy đủ các chức năng của đề tài.
* Có khả năng tự nghiên cứu cao, chủ động hoàn thành các công việc được giao.
* Báo cáo viết đầy đủ, rõ ràng và khoa học.

1. Khuyết điểm

* Cần bổ sung thêm một số chức năng mới có thể đưa vào ứng dụng trong thực tế.

1. Đề nghị cho bảo vệ hay không? Được bảo vệ
2. Đánh giá loại : Giỏi
3. Điểm : 9.5

*Tp. Hồ Chí Minh, ngày 25 tháng 12 năm 2018*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên)*

*Đặng Thị Kim Giao*

**XẢ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc lập – Tự do – Hạnh phúc**

**\*\*\*\*\*\*\***

**ĐH SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA CNTT**

**\*\*\*\*\*\*\***

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

Họ và tên Sinh viên 1: Đoàn Xuân Hùng MSSV 1: 15110219

Họ và tên Sinh viên 2: Phạm Minh Kế MSSV 2: 15110226

Ngành: Công Nghệ Thông Tin

Tên đề tài: Tìm hiểu Laravel framework và xây dựng ứng dụng minh họa (Website trắc nghiệm trực tuyến).

Họ và tên Giáo viên phản biện: Trương Thị Ngọc Phượng

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

1. Ưu điểm

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

1. Khuyết điểm

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

1. Đề nghị cho bảo vệ hay không?
2. Đánh giá loại :

*Tp. Hồ Chí Minh, ngày tháng năm 20*

Giáo viên phản biện

*(Ký & ghi rõ họ tên)*

1. Điểm :

**LỜI CẢM ƠN**

Nhóm em xin chân thành cám ơn khoa Công Nghệ Thông Tin, Bộ Môn Công Nghệ Phần Mềm, trường Đại học Sư Phạm Kỹ Thuật Tp.HCM đã tạo điều kiện thuận lợi cho chúng em thực hiện đề tài này.

Chúng em cũng xin được gửi lời cảm ơn chân thành nhất đến cô Đặng Thị Kim Giao, người đã tận tình chỉ bảo và hướng dẫn nhóm em thực hiện đề tài này.

Bên cạnh đó, chúng em xin gửi lời cảm ơn đến các thầy cô giảng viên của trường Đại Học Sư Phạm Kỹ Thuật Tp.HCM nói chung cũng như các thầy cô giảng viên khoa Công Nghệ Thông Tin nói riêng, những người đã giảng dạy, tạo điều kiện cho em tích lũy được những kiến thức quý báu trong những năm học qua.

Dù đã cố gắng hoàn thành đề tài tiểu luận đúng yêu cầu, nhưng do thời gian hạn hẹp và khả năng còn hạn chế nên chắc chắn sẽ không tránh khỏi thiếu sót. Chúng em mong nhận được sự thông cảm và tận tình chỉ bảo của các thầy cô và các bạn.

**Nhóm em xin chân thành cảm ơn.**

Trường ĐH Sư Phạm Kỹ Thuật TP.HCM

Khoa: CNTT

**ĐỀ CƯƠNG CHI TIẾT**

Họ và tên Sinh viên 1: Đoàn Xuân Hùng MSSV 1: 15110219

Họ và tên Sinh viên 2: Phạm Minh Kế MSSV 2: 15110226

Thời gian làm luận văn : Từ : 27/08/2018 Đến : 23/12/2018

Chuyên ngành: Công Nghệ Phần Mềm

Tên luận văn: Tìm hiểu Laravel framework và xây dựng ứng dụng minh hoạ (Website trắc nghiệm trực tuyến).

GV hướng dẫn: Đặng Thị Kim Giao

Nhiệm Vụ Của Luận Văn:

1. Tìm hiểu lý thuyết Laravel Framework
2. Xác định yêu cầu Website trắc nghiệm trực tuyến
3. Mô hình hóa yêu cầu, thiết kế cơ sở dữ liệu
4. Thiết kế giao diện
5. Viết code xử lý
6. Kiểm thử, sửa lổi
7. Viết báo cáo

MỤC LỤC

KẾ HOẠCH THỰC HIỆN

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Thời gian** | **Công việc** | **Ghi chú** |
| 1 | 27/8/2018 đến 23/09/2018  (Các tuần 2 3 4 5) | Tìm hiểu lý thuyết Laravel Framework:  Cài đặt, Cấu trúc Route, Controller, Model, Schema, View Blade, Get Pos, Query, Middleware, Authentication. |  |
| 2 | 24/9/2018 đến 30/09/2018  (Tuần 6) | Xác định yêu cầu Website trắc nghiệm trực tuyến:  - Yêu cầu chức năng.  - Yêu cầu phi chức năng. |  |
| 3 | 1/10/2018 đến 14/10/2018  (Các tuần 7, 8) | Mô hình hóa yêu cầu, thiết kế cơ sở dữ liệu:  - Lược đồ UseCase.  - Sơ đồ Sequence.  - Thiết kế database. |  |
| 4 | 15/10/2018 đến 04/11/2018  (Các tuần 9, 10, 11) | Thiết kế giao diện. |  |
| 5 | 5/11/2018 đến 2/12/2018  (Các tuần 12, 13, 14, 15) | Viết code xử lý. |  |
| 6 | 3/12/2018 đến 09/12/2018  (Tuần 16) | Kiểm thử, sửa lổi. |  |
| 7 | 10/12/2018 đến 16/12/2018  (Tuần 17) | Viết báo cáo. |  |

*Tp. Hồ Chí Minh, ngày 23 tháng 12 năm 2018*

Người viết đề cương

(Ký & ghi rõ họ tên)

Ý kiến của Giáo viên hướng dẫn

(Ký & ghi rõ họ tên)

**MỤC LỤC**

[**DANH SÁCH HÌNH ẢNH** 1](#_Toc533380831)

[**DANH SÁCH BẢNG BIỂU** 3](#_Toc533380832)

[**Phần 1: MỞ ĐẦU** 5](#_Toc533380833)

[**1.1.** **Tính cấp thiết của đề tài** 5](#_Toc533380834)

[**1.2.** **Mục đích của đề tài** 5](#_Toc533380835)

[**1.3.** **Đối tượng và phạm vi nghiên cứu** 5](#_Toc533380836)

[**1.4.** **Kết quả dự kiến đạt được** 5](#_Toc533380837)

[**Phần 2: NỘI DUNG** 6](#_Toc533380838)

[**Chương 1: GIỚI THIỆU LARAVEL FRAMEWORK** 6](#_Toc533380839)

[**1.1.** **Khái niệm** 6](#_Toc533380840)

[**1.2.** **Lịch sử phát triển của Laravel Framework** 6](#_Toc533380841)

[**1.3.** **Ưu nhược điểm của Laravel Framework** 7](#_Toc533380842)

[**1.3.1.** **Ưu điểm** 7](#_Toc533380843)

[**1.3.2.** **Nhược điểm** 8](#_Toc533380844)

[**1.4.** **Tại sao chúng ta nên sử dụng Laravel Framework** 8](#_Toc533380845)

[**1.5.** **Cấu trúc của thư mục Laravel Framework** 8](#_Toc533380846)

[**Chương 2: CÀI ĐẶT LARAVEL FRAMEWORK** 10](#_Toc533380847)

[**2.1.** **Cài đặt Composer** 10](#_Toc533380848)

[**2.2.** **Cài đặt Laravel** 10](#_Toc533380849)

[**Chương 3: ROUTE TRONG LARAVEL FRAMEWORK** 11](#_Toc533380850)

[**3.1.**  **Basic Routing** 11](#_Toc533380851)

[**3.2.**  **Route Parameters** 12](#_Toc533380852)

[**3.3.**  **Route Filters** 13](#_Toc533380853)

[**3.4.**  **Name Route** 15](#_Toc533380854)

[**3.5.**  **Route Groups** 16](#_Toc533380855)

[**3.6.**  **Route Prefixing** 17](#_Toc533380856)

[**Chương 4: VIEW TRONG LARAVEL FRAMEWORK** 17](#_Toc533380857)

[**Chương 5: CONTROLLER TRONG LARAVEL FRAMEWORK** 19](#_Toc533380858)

[**5.1.**  **Khai báo Controller trong Laravel Framework** 19](#_Toc533380859)

[**5.2.**  **Chạy Controller theo phương thức Route::get** 20](#_Toc533380860)

[**5.3.**  **Implicit Controllers – Sử dụng phương thức Route::controller** 21](#_Toc533380861)

[**5.4.**  **RESTful Resource Controllers – Sử dụng phương thức Route::resource** 22](#_Toc533380862)

[**5.5.**  **Controller với Filter** 25](#_Toc533380863)

[**Chương 6: KẾT NỐI DATABASE** 27](#_Toc533380864)

[**6.1.**  **Cấu hình kết nối cơ sở dữ liệu trong Laravel Framework** 27](#_Toc533380865)

[**6.2.**  **Database Transactions** 28](#_Toc533380866)

[**Chương 7: SỬ DỤNG QUERY BUILDER** 30](#_Toc533380867)

[**7.1.**  **Truy vấn dữ liệu** 30](#_Toc533380868)

[**7.1.1.** **Lấy tất cả dữ liệu trong bảng** 30](#_Toc533380869)

[**7.1.2.** **Truy vấn lấy 1 dòng dữ liệu từ bảng CSDL** 31](#_Toc533380870)

[**7.1.3.** **Truy vấn một cột từ một hàng đơn** 31](#_Toc533380871)

[**7.1.4.** **Truy vấn danh sách từ 1 bảng giá trị** 31](#_Toc533380872)

[**7.1.5.** **Sử dụng mệnh đề select** 32](#_Toc533380873)

[**7.1.6.** **Sử dụng từ khóa where, thêm điều kiện** 32](#_Toc533380874)

[**7.1.7.** **Sử dụng Order By, Group By, và Having** 33](#_Toc533380875)

[**7.1.8.** **Offset & Limit** 33](#_Toc533380876)

[**7.2.**  **Join bảng dữ liệu** 33](#_Toc533380877)

[**7.3.**  **Hàm trong Laravel Framework** 34](#_Toc533380878)

[**7.4.**  **Raw Expressions** 35](#_Toc533380879)

[**7.5.**  **Câu lệnh Inserts dữ liệu** 35](#_Toc533380880)

[**7.6.**  **Câu lệnh Updates dữ liệu** 36](#_Toc533380881)

[**7.7.**  **Câu lệnh Deletes dữ liệu** 36](#_Toc533380882)

[**7.8.**  **Câu lệnh Union** 36](#_Toc533380883)

[**Chương 8: SỬ DỤNG ELOQUENT ORM** 36](#_Toc533380884)

[**8.1.**  **Lấy dữ liệu** 37](#_Toc533380885)

[**8.2.**  **Thêm dữ liệu (Insert)** 38](#_Toc533380886)

[**8.3.**  **Cập nhật dữ liệu (Update)** 38](#_Toc533380887)

[**8.4.**  **Xóa dữ liệu (Delete)** 39](#_Toc533380888)

[**Chương 9: SỬ DỤNG MIGRATIONS VÀ SCHEMA BUILDER** 39](#_Toc533380889)

[**9.1.**  **Lớp Schema Builder – Thao tác với bảng CSDL** 40](#_Toc533380890)

[**9.1.1.** **Phương thức create – Tạo bảng dữ liệu** 40](#_Toc533380891)

[**9.1.2.** **Phương thức rename – Đổi tên bảng dữ liệu** 41](#_Toc533380892)

[**9.1.3.** **Phương thức drop – Xóa bảng dữ liệu** 41](#_Toc533380893)

[**9.1.4.** **Phương thức table – Cập nhật thêm cột và một bảng tồn tại** 41](#_Toc533380894)

[**9.1.5.** **Phương thức renameColumn – Đổi tên cột** 41](#_Toc533380895)

[**9.1.6.** **Phương thức dropColumn- Xóa cột** 41](#_Toc533380896)

[**9.1.7.** **Phương thức hasTable – Kiểm tra có tồn tại bảng không** 42](#_Toc533380897)

[**9.1.8.** **Phương thức hasColumn- Kiểm tra có tồn tại cột không** 42](#_Toc533380898)

[**9.1.9.** **Thêm chỉ mục vào** 42](#_Toc533380899)

[**9.1.10.**  **Sử dụng khóa ngoại** 42](#_Toc533380900)

[**9.2.**  **Migrations – Quản lý CSDL** 43](#_Toc533380901)

[**9.2.1.** **Dùng artisan tạo migration** 43](#_Toc533380902)

[**9.2.2.** **Chỉnh sửa migrations** 45](#_Toc533380903)

[**Chương 10: WEBSITE THI TRẮC NGHIỆM ONLINE** 46](#_Toc533380904)

[**10.1.** **Khảo sát hiện trạng** 46](#_Toc533380905)

[**10.2.** **Xác định yêu cầu** 46](#_Toc533380906)

[**10.2.1.**  **Yêu cầu chức năng** 46](#_Toc533380907)

[**10.2.2.**  **Yêu cầu phi chức năng** 47](#_Toc533380908)

[**10.2.3.**  **Phân tích và mô hình hóa yêu cầu** 48](#_Toc533380909)

[**10.3.** **Phân tích thiết kế cơ sở dữ liệu** 49](#_Toc533380910)

[**10.3.1.**  **Lược đồ Use Case** 49](#_Toc533380911)

[**10.3.2.**  **Mô hình hóa yêu cầu** 49](#_Toc533380912)

[**10.3.3.**  **Cơ sở dữ liệu** 55](#_Toc533380913)

[**10.3.4.**  **Lược đồ Diagram** 61](#_Toc533380914)

[**10.4.** **Thiết kế giao diện** 61](#_Toc533380915)

[**10.4.1.**  **Giao diện trang chủ** 61](#_Toc533380916)

[**10.4.2.**  **Giao diện trang đăng kí** 63](#_Toc533380917)

[**10.4.3.**  **Giao diện đăng nhập** 65](#_Toc533380918)

[**10.4.4.**  **Giao diện trang Thông tin cá nhân học viên** 65](#_Toc533380919)

[**10.4.5.**  **Giao diện trang lịch sử học tập** 67](#_Toc533380920)

[**10.4.6.**  **Giao diện danh sách đề thi theo môn học** 67](#_Toc533380921)

[**10.4.7.**  **Giao diện trang làm bài thi** 68](#_Toc533380922)

[**10.4.8.**  **Giao diện trang tài liệu** 69](#_Toc533380923)

[**10.4.9.**  **Giao diện quản lý ngân hàng câu hỏi** 70](#_Toc533380924)

[**10.4.10. Giao diện trang quản lý ngân hàng đề thi** 73](#_Toc533380925)

[**10.4.11. Giao diện trang quản lý ngân hàng tài liệu** 76](#_Toc533380926)

[**10.4.12. Giao diện trang thông báo** 77](#_Toc533380927)

[**10.4.13. Giao diện trang quản lý tin tức** 79](#_Toc533380928)

[**10.4.14. Giao diện quản lý User** 80](#_Toc533380929)

[**10.4.15. Giao diện trang quản lý môn học** 82](#_Toc533380930)

[**10.4.16. Giao diện trang phân công giảng dạy** 84](#_Toc533380931)

[**10.4.17. Giao diện trang quản lý chủ đề môn học** 85](#_Toc533380932)

[**10.5.** **Kiểm thử phần mềm** 87](#_Toc533380933)

[**Phần 4: KẾT LUẬN** 91](#_Toc533380934)

[**4.1.** **Những kết quả đạt được** 91](#_Toc533380935)

[**4.2.** **Những khó khăn khi thực hiện đề tài** 91](#_Toc533380936)

[**4.3.** **Ưu điểm** 91](#_Toc533380937)

[**4.4.** **Hạn chế** 91](#_Toc533380938)

[**4.5.** **Hướng phát triển trong tương lai** 91](#_Toc533380939)

[**TÀI LIỆU THAM KHẢO** 93](#_Toc533380940)

**DANH SÁCH HÌNH ẢNH**

[Hình 1.5. 1 Thư mục Laravel Framework 9](#_Toc533366917)

[Hình 1.5. 2 Chức năng của các thư mục chính 9](#_Toc533366918)

[Hình 1.5. 3 Chức năng cụ thể của các thư mục con trong app 10](#_Toc533366919)

[Hình 2.2. 1 Cài đặt Laravel thành công 11](#_Toc533366943)

[Hình 9.2.1. 1 Thực hiện migrate bằng command line 45](#_Toc533366952)

[Hình 9.2.1. 2 Bảng dữ liệu được tạo bằng lệnh migrate 45](#_Toc533366953)

[Hình 10.3.1. 1 Lược đồ use-case 49](#_Toc533367001)

[Hình 10.3.2. 1 Lược đồ tuần tự quản lý ngân hàng câu hỏi 50](#_Toc533367009)

[Hình 10.3.2. 2 Lược đồ tuần tự quản lý ngân hàng đề thi 50](#_Toc533367010)

[Hình 10.3.2. 3 Lược đồ tuần tự quản lý tin tức 51](#_Toc533367011)

[Hình 10.3.2. 4 Lược đồ tuần tự quản lý tài liệu 51](#_Toc533367012)

[Hình 10.3.2. 5 Lược đồ tuần tự đăng thông báo 52](#_Toc533367013)

[Hình 10.3.2. 6 Lược đồ tuần tự đổi mật khẩu 52](#_Toc533367014)

[Hình 10.3.2. 7 Lược đồ tuần tự quản lý môn học hệ thống 53](#_Toc533367015)

[Hình 10.3.2. 8 Lược đồ tuần tự quản lý chủ đề môn học 53](#_Toc533367016)

[Hình 10.3.2. 9 Lược đồ tuần tự quản lý users 54](#_Toc533367017)

[Hình 10.3.2. 10 Lược đồ tuần tự phân công giảng dạy 54](#_Toc533367018)

[Hình 10.3.2. 11 Lược đồ tuần tự làm bài thi 55](#_Toc533367019)

[Hình 10.3.4. 1 Lược đồ Diagram 61](#_Toc533367021)

[Hình 10.4.1. 1 Giao diện trang chủ 1 62](#_Toc533367070)

[Hình 10.4.1. 2 Giao diện trang chủ 2 62](#_Toc533367071)

[Hình 10.4.2. 1 Giao diện trang đăng ký 64](#_Toc533367078)

[Hình 10.4.3. 1 Giao diện trang đăng nhập 65](#_Toc533367096)

[Hình 10.4.4. 1 Giao diện thông tin cá nhân học viên 66](#_Toc533367116)

[Hình 10.4.5. 1 Giao diện trang lịch sử học tập 67](#_Toc533367121)

[Hình 10.4.6. 1 Giao diện danh sách đề thi theo môn học 68](#_Toc533367127)

[Hình 10.4.7. 1 Giao diện trang làm bài thi 69](#_Toc533367134)

[Hình 10.4.8. 1 Giao diện trang tài liệu 70](#_Toc533367153)

[Hình 10.4.9. 1 Giao diện danh sách ngân hàng câu hỏi 71](#_Toc533367159)

[Hình 10.4.9. 2 Giao diện trang thêm câu hỏi 1 71](#_Toc533367160)

[Hình 10.4.9. 3 Giao diện trang thêm câu hỏi 2 72](#_Toc533367161)

[Hình 10.4.10. 1 Giao diện trang danh sánh đề thi 73](#_Toc533367175)

[Hình 10.4.10. 2 Giao diện thêm đề thi 1 74](#_Toc533367176)

[Hình 10.4.10. 3 Giao diện thêm đề thi 2 74](#_Toc533367177)

[Hình 10.4.11. 1 Giao diện trang ngân hàng tài liệu 76](#_Toc533367180)

[Hình 10.4.11. 2 Giao diện trang thêm tài liệu 76](#_Toc533367181)

[Hình 10.4.12. 1 Giao diện trang danh sách thông báo 77](#_Toc533367185)

[Hình 10.4.12. 2 Giao diên trang thêm thông báo 78](#_Toc533367186)

[Hình 10.4.13. 1 Giao diện trang danh sách tin tức 79](#_Toc533367223)

[Hình 10.4.13. 2 Giao diện trang thêm tin tức 79](#_Toc533367224)

[Hình 10.4.14. 1 Giao diện trang danh sách User 80](#_Toc533367241)

[Hình 10.4.14. 2 Giao diên trang thêm tài khoản 81](#_Toc533367242)

[Hình 10.4.15. 1 Giao diện trang danh sách môn học 82](#_Toc533367262)

[Hình 10.4.15. 2 Giao diện trang thêm môn học 83](#_Toc533367263)

[Hình 10.4.16. 1 Giao diện trang danh sách phân công giảng dạy 84](#_Toc533367269)

[Hình 10.4.16. 2 Giao diện trang thêm phân công mới 84](#_Toc533367270)

[Hình 10.4.17. 1 Giao diện danh sách chủ đề 85](#_Toc533367273)

[Hình 10.4.17. 2 Giao diện trang thêm chủ đề mới 86](#_Toc533367274)

**DANH SÁCH BẢNG BIỂU**

[Bảng 10.2.2. 1 Yêu cầu phi chức năng 47](#_Toc533365297)

[Bảng 10.2.3. 1 Phân tích và mô hình hóa yêu cầu 48](#_Toc533365310)

[Bảng 10.3.3. 1 dbo.BangDiem 55](#_Toc533365344)

[Bảng 10.3.3. 2 dbo.CauHoi 56](#_Toc533365345)

[Bảng 10.3.3. 3 dbo.ChiTietDeThi 56](#_Toc533365346)

[Bảng 10.3.3. 4 dbo.ChuDe 57](#_Toc533365347)

[Bảng 10.3.3. 5 dbo.DeThi 57](#_Toc533365348)

[Bảng 10.3.3. 6 dbo.GiangDay 58](#_Toc533365349)

[Bảng 10.3.3. 7 dbo.Level 58](#_Toc533365350)

[Bảng 10.3.3. 8 dbo.MonHoc 58](#_Toc533365351)

[Bảng 10.3.3. 9 dbo.TaiLieu 59](#_Toc533365352)

[Bảng 10.3.3. 10 dbo.ThongBao 59](#_Toc533365353)

[Bảng 10.3.3. 11 dbo.TinTuc 60](#_Toc533365354)

[Bảng 10.3.3. 12 dbo.TypeAccount 60](#_Toc533365355)

[Bảng 10.3.3. 13 dbo.Users 60](#_Toc533365356)

[Bảng 10.4.1. 1 Xử lý giao diện trang chủ 62](#_Toc533365368)

[Bảng 10.4.2. 1 Xử lý giao diện trang đăng ký 64](#_Toc533365399)

[Bảng 10.4.3. 1 Xử lý giao diện trang đăng nhập 65](#_Toc533365426)

[Bảng 10.4.4. 1 Xử lý giao diện trang thông tin cá nhân học viên 66](#_Toc533365450)

[Bảng 10.4.5. 1 Xử lý giao diện trang lịch sử học tập 67](#_Toc533365488)

[Bảng 10.4.6. 1 Xử lý giao diện trang danh sách đề thi theo môn học 68](#_Toc533365497)

[Bảng 10.4.7. 1 Xử lý giao diện trang danh sách đề thi theo môn học 69](#_Toc533365505)

[Bảng 10.4.8. 1 Xử lý giao diện trang tài liệu 70](#_Toc533365521)

[Bảng 10.4.9. 1 Xử lý giao diện trang danh sách ngân hàng câu hỏi 72](#_Toc533365527)

[Bảng 10.4.10. 1 Xử lý giao diện trang danh sách ngân hàng đề thi 74](#_Toc533365583)

[Bảng 10.4.10. 2 Xử lý giao diện trang thêm đề thi 1 75](#_Toc533365584)

[Bảng 10.4.10. 3 Xử lý giao diện trang thêm đề thi 2 75](#_Toc533365585)

[Bảng 10.4.11. 1 Xử lý giao diện trang danh sách tài liệu 76](#_Toc533365594)

[Bảng 10.4.11. 2 Xử lý giao diện trang thêm tài liệu 77](#_Toc533365595)

[Bảng 10.4.12. 1 Xử lý giao diện trang danh sách thông báo 78](#_Toc533365643)

[Bảng 10.4.12. 2 Xử lý giao diện trang thêm thông báo 78](#_Toc533365644)

[Bảng 10.4.13. 1 Xử lý giao diện trang danh sách thông báo 79](#_Toc533365654)

[Bảng 10.4.13. 2 Xử lý giao diện trang thêm tin tức 80](#_Toc533365655)

[Bảng 10.4.14. 1 Xử lý giao diện trang danh sách user 81](#_Toc533365663)

[Bảng 10.4.14. 2 Xử lý giao diện trang thêm tài khoản 81](#_Toc533365664)

[Bảng 10.4.15. 1 Xử lý giao diện trang danh sách môn học 83](#_Toc533365670)

[Bảng 10.4.15. 2 Xử lý giao diện trang thêm môn học 83](#_Toc533365671)

[Bảng 10.4.16. 1 Xử lý giao diện trang danh sách phân công giảng dạy 84](#_Toc533365695)

[Bảng 10.4.16. 2 Xử lý giao diện trang thêm phân công mới 85](#_Toc533365696)

[Bảng 10.4.17. 1 Xử lý giao diện trang danh sách chủ đề 86](#_Toc533365703)

[Bảng 10.4.17. 2 Xử lý giao diện trang thêm chủ đề mới 86](#_Toc533365704)

[Bảng 10.5. 1 Bảng danh sách kiểm thử 87](#_Toc533365722)

**Phần 1: MỞ ĐẦU**

* 1. **Tính cấp thiết của đề tài**
* Có hơn 20 Framework mã nguồn mở PHP khác nhau, nhưng trong số đó thì Laravel Framework là phổ biến nhất.
* Nó là 1 Framework khá mới mẻ nhưng bù lại nó có "hướng dẫn sử dụng" khá đầy đủ, rõ ràng và dễ hiểu và cùng với cộng đồng phát triển rộng lớn trên toàn thế giới.
* Ứng dụng công nghệ thông tin vào dạy học và trắc nghiệm đang là hai trào lưu được nhiều người quan tâm, hưởng ứng.
* Đã có nhiều người nghiên cứu việc ứng dụng các phần mềm hỗ trợ trắc nghiệm sẵn có và chia sẻ kinh nghiệm với người dùng. Tuy nhiên, các phần mềm sẵn có thường không miễn phí, hoặc hạn chế một số tính năng nào đấy, có thể tiềm ẩn virus, quảng cáo… Việc đối mới giáo dục của nước ta trong những năm qua đã chuyển từ hình thức học và thi “tự luận” sang "trắc nghiệm” đã làm tăng hiệu quả trong học tập và giảm chi phí tổ chức các kỳ thi.

=> Nhận biết điều đó, nhóm chúng em sẻ sử dụng Laravel Framework để thực hiện việc xây dựng một website trắc nghiệm hỗ trợ cho việc học tập của các bạn học sinh, sinh viên.

**1.2. Mục đích của đề tài**

* Tìm hiểu về Laravel Framework.
* Xây dựng website demo dựa trên framework đó.

**1.3. Đối tượng và phạm vi nghiên cứu**

* Các công cụ lập trình ngôn ngữ PHP Laravel Framework.
* Các công cụ, framework, ngôn ngữ thiết kế giao diện và xây dựng website: Ajax, JavaScript, Bootstrap, HTML, CSS.
* Nghiên cứu các chức năng của Website thi trắc nghiệm online.

**1.4. Kết quả dự kiến đạt được**

* Nắm vững được kiến thức Laravel Framework và vận dụng nó.
* Xây dựng thành công Website thi trắc nghiệm online.

**Phần 2: NỘI DUNG**

**Chương 1: GIỚI THIỆU LARAVEL FRAMEWORK**

* 1. **Khái niệm**

Laravel là một nền tảng ứng dụng web mã nguồn mở được viết bằng ngôn ngữ PHP dựa theo mô hình MVC. Nó là framework khá mới mẻ, ra đời vào tháng 04-2011 và cha đẻ của nó Taylor Otwell. Việc tiếp cận và sử dụng Laravel cũng giống như các framework khác không phải là vấn đề quá khó khăn. Các nhà phát triển framework Laravel đã xây dựng hệ thống hướng dẫn sử dụng khá hoàn chỉnh và đầy đủ. Nó giúp cho các lập trình viên tiết kiệm thời gian, công sức và chi phí để xây dựng một website.

**1.2. Lịch sử phát triển của Laravel Framework**

Bản Laravel beta đầu tiên được phát hành vào ngày 9/6/2011, tiếp đó là Laravel 1 phát hành trong cùng tháng. Laravel 1 bao gồm các tính năng như xác thực, bản địa hóa, model, view, session, định tuyến và các cơ cấu khác, nhưng vẫn còn thiếu controller, điều này làm nó chưa thật sự là một MVC framework đúng nghĩa.

Laravel 2 được phát hành vào tháng 9 năm 2011, mang đến nhiều cải tiến từ tác giả và cộng đồng. Tính năng đáng kể bao gồm hỗ trợ controller, điều này thực sự biến Laravel 2 thành một MVC framework hoàn chỉnh, hỗ trợ Inversion of Control (IoC), hệ thống template Blade. Bên cạnh đó, có một nhược điểm là hỗ trợ cho các gói của nhà phát triển bên thứ 3 bị gỡ bỏ.

Laravel 3 được phát hành vào tháng 2 năm 2012, với rất nhiều tính năng mới bao gồm giao diện dòng lệnh (CLI) tên “Artisan”, hỗ trợ nhiều hơn cho hệ thống quản trị cơ sở dữ liệu, chức năng ánh xạ cơ sở dữ liệu Migration, hỗ trợ “bắt sự kiện” trong ứng dụng và hệ thống quản lý gói gọi là “Bundles”. Lượng người dùng và sự phổ biến tăng trưởng mạnh kể từ phiên bản Laravel 3.

Laravel 4, tên mã “Illuminate”, được phát hành vào tháng 5 năm 2013. Lần này thực sự là sự lột xác của Laravel framework, di chuyển và tái cấu trúc các gói hỗ trợ vào một tập được phân phối thông qua Composer, một chương trình quản lý gói thư viện phụ thuộc độc lập của PHP. Bố trí mới như vậy giúp khả năng mở rộng của Laravel 4 tốt hơn nhiều so với các phiên bản trước. Ra mắt lịch phát hành chính thức sáu tháng một phiên bản nâng cấp nhỏ. Các tính năng khác trong Laravel 4 bao gồm tạo và thêm dữ liệu mẫu (database seeding), hỗ trợ các kiểu gửi mail, và hỗ trợ “xóa mềm” (soft-delete: record bị lọc khỏi các truy vấn từ Eloquent mà không thực sự xóa hẳn khỏi DB).

Laravel 5 được phát hành trong tháng 2 năm 2015, như một kết quả thay đổi đáng kể cho việc kết thúc vòng đời nâng cấp Laravel lên 4.3. Bên cạnh một loạt tính năng mới và các cải tiến như hiện tại, Laravel 5 cũng giới thiệu cấu trúc cây thư mục nội bộ cho phát triển ứng dụng mới. Những tính năng mới của Laravel 5 bao gồm hỗ trợ lập lịch định kỳ thực hiện nhiệm vụ thông qua một gói tên là “Scheduler”, một lớp trừu tượng gọi là “Flysystem” cho phép điều khiển việc lưu trữ từ xa đơn giản như lưu trữ trên máy local - dễ thấy nhất là mặc định hỗ trợ dịch vụ Amazone S3, cải tiến quản lý assets thông qua “Elixir”, cũng như đơn giản hóa quản lý xác thực với các dịch vụ bên ngoài bằng gói “Socialite”.

Laravel 5.1 phát hành vào tháng 6 năm 2015, là bản phát hành đầu tiên nhận được hỗ trợ dài hạng (LTS) với một kết hoạch fix bug lên tới 2 năm vào hỗ trợ vá lỗi bảo mật lên tới 3 năm.

Laravel 5.3, được phát hành vào ngày 23 tháng 8 năm 2016. Các tính năng mới trong 5.3 tập trung vào việc cải thiện tốc độ phát triển bằng cách bổ sung thêm các cải tiến cho các tác vụ phổ biến. Phiên bản này có nhiều tính năng mới, như Laravel Dusk, Laravel Mix, Blade Components và Slots, Markdown Emails, Automatic Facades, Route Improvements, Higher Order Messaging cho Collections, và nhiều thứ khác nữa.

Laravel 5.5, phát hành vào ngày 30 tháng 8 năm 2017.

Laravel 5.6, phát hành vào ngày 7 tháng 2 năm 2018.

Laravel 5.7, phát hành vào ngày 4 tháng 9 năm 2018.

**1.3. Ưu nhược điểm của Laravel Framework**

**1.3.1. Ưu điểm**

Được thiết kế theo mô hình Model-View-Controller: Mô hình MVC giúp tách thành phần hiển thị giao diện (presentation) và xử lý (business logic) của một phần mềm thành những thành phần độc lập, từ đó giúp cho việc thiết kế, xử lý và bảo trì mã nguồn dễ dàng, đồng thời tăng khả năng mở rộng của phần mềm. Laravel vận dụng mô hình này trong thiết kế, giúp tách biệt các tập tin giao diện với các tập tin xử lý dữ liệu, nâng cao khả năng quản lý và dễ bảo trì. Hệ thống tài liệu (document) đã được làm lại một cách cẩn thận và rõ ràng hơn. Mỗi trang tài liệu đã được dành vô số thời gian tỉ mỉ để chăm chút cẩn thận. Điều này cũng giúp cho người dùng sử dụng nhanh chóng nắm được cách sử dụng Laravel, từ đó sẽ có nhiều người sử dụng nó hơn. Hệ thống tài liệu đi kèm với khả năng tìm kiếm real-time và auto complete. Việc quản lý layout thật sự giản đơn. Dễ dàng tích hợp các thư viện của Zend và các gói bundle (giống như moduel) vào project. Có rất nhiều gói bundle được cộng đồng mạng build sẵn và chia sẻ trên Internet. Laravel hỗ trợ nhiều cách viết để viết ngắn gọn nhất có thể và nhiều cách viết khác nhau: Như Query Builder và Eloquent.

**1.3.2. Nhược điểm**

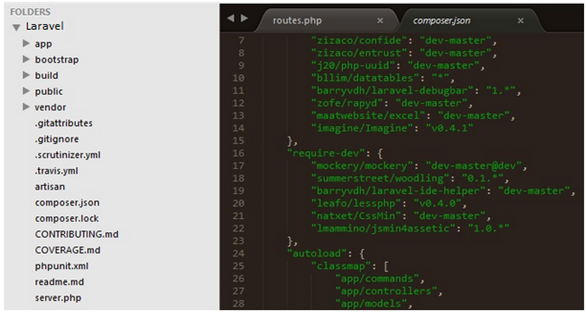
Điểm yếu của Laravel là không hỗ trợ các phiên bản PHP cũ vì thế các website có nền tảng PHP version thấp hơn khá là khó khi có ý định chuyển sang Laravel và phải luôn cập nhật các thông tin mới nhất về PHP để áp dụng vào Laravel. Phiên bản của Laravel 5.3 hiện tại chỉ hỗ trợ phiên bản 5.6 trở lên.

**1.4. Tại sao chúng ta nên sử dụng Laravel Framework**

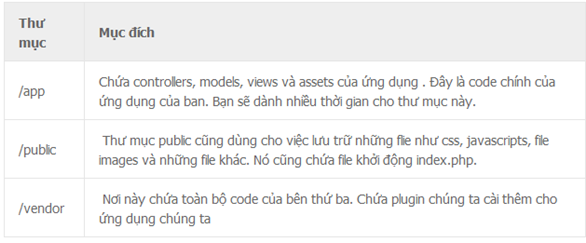
* Được thừa hưởng những ưu điểm và thế mạnh của các framework khác.
* Có số lượng người sử dụng nhiều nhất.
* Document rõ dàng, dễ học.
* Autoload theo namespace.
* Sử dụng mô hình ORM rất đơn giản khi thao tác với DB.
* Các lệnh tương tác với cơ sở dữ liệu cực kỳ ngắn gọn và thân thiện.
* Việc quản lý layout thật sự giản đơn với Balade Templating.
* Dễ dàng tích hợp các thư viện khác vào dự án, và được quản lý với Composer.
* Phần Route rất mạnh.

**1.5. Cấu trúc của thư mục Laravel Framework**

Một dự án web viết bằng Laravel có cấu trúc thư mục như sau:

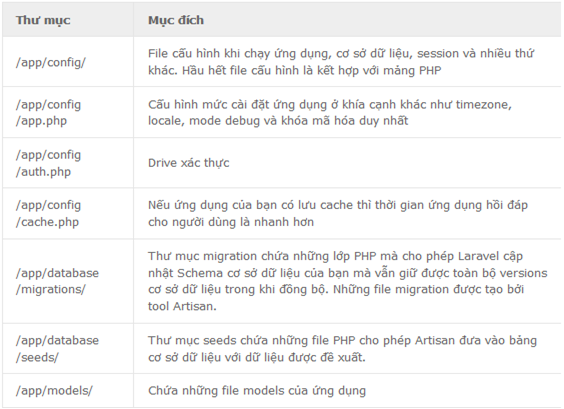


Hình 1.5. Thư mục Laravel Framework



Hình 1.5. Chức năng của các thư mục chính

Như đã đề cập ở trên, thư mục /app là nguồn chính của ứng dụng, vì vậy chúng ta cần tìm hiểu sâu về chúng:



Hình 1.5. Chức năng cụ thể của các thư mục con trong app

**Chương 2: CÀI ĐẶT LARAVEL FRAMEWORK**

**2.1. Cài đặt Composer**

Chúng ta sẽ cài đặt Laravel bằng composer nên các bạn cần cài đặt composer trước, tôi dùng window nên tôi download bản cài đặt composer của window. Bạn có thể tải composer tại địa chỉ: http://laravel.com/docs/4.2/installation

Các bạn download về và cài đặt bình thường, lưu ý ở chỗ chọn php thôi, các bạn chỉ đường dẫn đến file php.exe trong thư mục php của XAMPP (xampp/php/php.exe).

**2.2. Cài đặt Laravel**

Bản Laravel 4.2, bản này yêu cầu chạy trên PHP>=5.4, hiện tại tôi đang dùng XAMPP 5.6.3 nên cài được bản mới nhất của Laravel. Nếu các bạn muốn cài đặt bản mới nhất thì chỉ cần dùng lệnh sau:

*composer create-project laravel/laravel tên-thư-mục-laravel-của-bạn*

Còn muốn nếu install các bản thấp hơn thì dùng:

*composer create-project laravel/laravel tên-thư-mục-laravel-của-bạn 4.0.\**

Sau khi cài đặt xong, chúng ta có cấu trúc thư mục Laravel như hình 1.5.1 phía trên. Để biết được có cài Laravel thành công hay không thì chúng ta thử chạy Laravel bằng cách mở trình duyệt lên rồi gõ địa chỉ như sau: http://localhost/laravel\_demo/public/. Nếu thành công thì sẽ hiển thị hình dưới đây:



Hình 2.2. Cài đặt Laravel thành công

**Chương 3: ROUTE TRONG LARAVEL FRAMEWORK**

Mục đích của router là định tuyến đến những controller cụ thể nào từ phía request của người sử dụng. Đối với những Framework khác thì khi các bạn muốn thực hiện 1 công việc nào đó thường chúng ta cần xây dựng 1 controller để xử lý phải không nào, nhưng trong Laravel thì các bạn hoàn toàn có thể xử lý dữ liệu ngay trong phần Router. Đây chính là điểm mạnh cũng như điểm khác biệt lớn của Laravel Framework đối với các Framework khác.

**3.1. Basic Routing**

Các định tuyến trong Laravel đều được viết trong app/route.php. Cú pháp đơn giản nhất của 1 định tuyến đó là:

Route::method(‘URI’,’Function call back’)

Trong đó thì:

* URI là dạng link trên url.
* Function callback: Hàm sẽ gọi tới link URI phía trên được chạy, đây chính là nơi các bạn có thể xử lý dữ liệu.
* Method chính là các dạng phương thức cơ bản: post, get, put, delete, any.

+ POST Route: Các thao tác lấy từ form như thêm dữ liệu.

+ GET Route: Dành cho các thao tác truy cập thông thường tương đương với request cơ bản trong PHP. Ví dụ chạy 1 link trên url.

+ PUT Route: Dành cho thao tác lấy từ form nhưng là cập nhật dữ liệu.

+ DELETE Route: Dành cho thao tác hành động xóa dữ liệu.

+ ANY Route: là sự tổng hợp các thao tác ở trên.

Trong đó thì 2 dạng POST và GET sẽ được sử dụng nhiều nhất.

Ví dụ:

Các bạn mở file app/route.php và viết dòng code sau:

*Route::get(‘/san-pham’, function(){*

*echo ‘Trang sản phẩm’;*

*});*

Tiếp theo truy cập vào đường link url với đường dẫn localhost/kienthuc24h/san-pham (Trong đó thì kienthuc24h chính là thư mục dự án). Khi các bạn chạy link này lập tức trong Route sẽ kiểm tra và gọi đúng tới hàm callback. Sau khi chạy trang này các bạn sẽ nhận được thông báo như ý muốn.

Ngoài ra các bạn còn có thể tùy chỉnh rất nhiều trong phần Route này theo cú pháp:

*Route::method(‘Tên định danh’,’Tham số’);*

Ví dụ:

*Route::get(“san-pham”,”ProductController@list”);*

Với ví dụ trên có nghĩa là khi chúng ta chạy link san-pham lúc này bộ điều hướng sẽ gọi tới controller là ProducController và phương thức (action) là list.

**3.2. Route Parameters**

Trong trường hợp các bạn muốn gửi kèm theo tham số cho bộ định tuyến thì chỉ cần khai báo theo dạng {Tên tham số} và trong hàm callback chúng ta coi nó như tham số trong hàm bình thường.

Ví dụ:

*Route::get(‘/san-pham/{name}/{id}’, function($name,$id){*

*echo ‘Bạn đang xem sản phẩm’.$name.’có ID = ’.&id;*

*});*

Tiếp theo bạn truy cập vào đường link localhost/kienthuc24h/san-pham/nokia520/1 và kết quả sẽ là:

*Bạn đang xem sản phẩm nokia520 có ID = 1*

Ngoài ra các bạn hoàn toàn có thể kiểm tra dữ liệu của tham số truyền vào bằng cách sử dụng thêm phương thức where().

Ví dụ:

*Route::get(‘/san-pham/{name}/{id}’, function($name,$id){*

*echo ‘Bạn đang xem sản phẩm’.$name.’có ID = ’.&id;*

*})->where(array(‘name’=>’[a-z]+’,’id’=>’[0-9]+’));*

Ở trường hợp trên ta xét với điều kiện kiểm tra sẽ là sản phẩm (name) sẽ là các ký tự thường từ a-z còn id sẽ là các số tự nhiên từ 0-9.

**3.3. Route Filters**

Đây chính là phần bộ lọc định tuyến, nó cho phép chúng ta có thể kiểm tra trước khi cập nhật vào đường dẫn đó (URI). Ứng dụng trong thực tế là kiểm tra trạng thái đăng nhập trước khi cho phép truy cập vào 1 trang nào đó.

Filter chia ra 2 loại là before và after.

* Before: Nghĩa là request trước khi tới routing.
* After: Nghĩa là request sau khi tới routing.

Trên thực tế thì Before thường được sử dụng nhiều hơn.

Cú pháp cơ bản:

*Route::filter(“tên filter”, function(){*

*// code xử lý*

*});*

Ví dụ:

*Route:: filter(“checklogin”, function(){*

*if(session::get(‘login’)!=1){*

*return Redirect::to(“/dang-nhap”);*

*}*

*});*

Đoạn code trên mình xây dựng 1Filter có tên là checklogin đang thực hiện kiểm tra xem biến session login có bằng 1, ý định của mình ở đây là kiểm tra xem đã đăng nhập chưa. Nếu chưa đăng nhập thì lập tức chuyển về trang đăng nhập.

Tiếp theo ta có đoạn code

*Route::get(‘/them-san-pham’, array(“before”=>”checklogin”,”uses”=>*

*”ProductController@store”));*

Với đoạn code trên mình đã thêm vào khóa before chỉ tới checklogin. Chính là filter ta đã thiết lập trước đó, và đẩy nó sang controller ProductController với action là store dựa vào khóa uses.

* Như vậy, khi các bạn truy cập vào link them-san-pham thì lúc này nó sẽ kiểm tra Filter checklogin trước.
* Với bất kỳ link nào đó mà bạn muốn kiểm tra đăng nhập thì chỉ cần sử dụng “before”=>”checklogin” là được.
* Filter trong Laravel Framework còn cho phép chúng ta truyền giá trị vào dựa vào cú pháp sau:

*Route::filter(‘Tên filter’, function($route, $request,$value){*

*// code xử lý, kiểm tra dữ liệu*

*});*

+ Như các bạn thấy tham số thứ 3 $value chính là giá trị mà chúng ta sẽ truyền vào, ví dụ:

*Route::filter(“checkId”, function($route, $request,$value){*

*// Xử lý dữ liệu*

*if($value<=0)*

*{*

*echo ‘Id khong hop le’;*

*}*

*});*

*Route::get(‘/check-id’,array(“before”=>”checkId:-1”,”uses”=>”*

*DemoController@checkid“));*

+ Như vậy khi các bạn chạy link check-id lúc này sẽ gọi tới Filter checkId để kiểm tra, và giá trị mình gửi sang bên Filter đó chính là -1(checkId:-1).

Qua những ví dụ này các bạn thấy được ý nghĩa và sức mạnh của filter trong thực tế phải không nào. Với filter ta có thể tối giản code ở mức cao nhất và tái sử dụng vô cùng đơn giản.

**3.4. Name Route**

Đặt tên cho các Route nhằm sử dụng linh hoạt cho ứng dụng. Dễ dàng lấy được thông tin của Route thông qua tên này.

Ví dụ:

*Route::get(‘/san-pham/{name}/{id}’, array(‘as’=>’product’,’uses’=>ProductController@view));*

*$url= URL::route(“product”,array(‘nokia520’,’1’));*

*echo “<a href=’$url’>Chi tiết sản phẩm</a>”;*

**+** as: Là tên của route (product).

+ uses: Route này sẽ sử dụng controller (ProductController) nào và action (view) nào.

+ Thư viện URL, sử dụng phương thức route với cú pháp sau:

*URL::route(“tên route”, array(‘chứa đối số’));*

Sau khi chạy đoạn code trên chúng ta sẽ có $url=/san-pham/nokia520/1.

Tiếp đến ta xây dựng 1 controller với nội dung sau:

*<?php*

*class ProductController extends BaseController{*

*public function view($name,$id)*

*{*

*echo ‘Bạn đang xem sản phẩm’.$name.’có ID = ’.$id;*

*}*

*}*

và kết quả sẽ hiển thị ra như sau:

*Bạn đang xem sản phẩm nokia520 có ID = 1*

**3.5. Route Groups**

Route Groups sẽ giúp chúng ta có thể nhóm các Route mà cần thông qua bộ lọc định tuyến nào đó (Filter). Như với ví dụ trên chúng ta có Filter là checklogin, giả sử chúng ta có rất nhiều trang cần sử dụng bộ lọc kiểm tra đăng nhập này, không lẻ chúng ta phải viết tất cả các khóa before. Nhưng khi sử dụng Route Groups các bạn chỉ cần viết 1 lần khóa before cho nhóm đó.

Ví dụ:

*Route::group(array(“before”=>”checklogin”), function(){*

*Route::get(‘/them-san-pham’,array(“uses”=>”Product*

*Controller@store”));*

*Route::get(‘/cap-nhat-san-pham’,array(“uses”=>”Product*

*Controller@update”));*

*Route::get(‘/them-danh-muc’,array(“uses”=>”Catalog*

*Controller@store”));*

*Route::get(‘/cap-nhat-danh-muc’,array(“uses”=>”Catalog*

*Controller@update”));*

*});*

Như vậy là khi các bạn truy cập vào các link ở trên thì sẽ được thông qua Filter checklogin kiểm trước.

**3.6. Route Prefixing**

Gồm 1 số các định tuyến mà đằng trước nó có tiền tố:

Ví dụ:

*localhost/kienthuc24h/admin/them-san-pham*

*localhost/kienthuc24h/admin/cap-nhat-san-pham*

*localhost/kienthuc24h/admin/them-danh-muc*

*localhost/kienthuc24h/admin/cap-nhat-danh-muc*

Với các đường link trên thì các bạn thấy trước nó đều có tiền tố (Prefix) là admin phải không nào. Lúc này chúng ta hoàn toàn có thể cho nó vào 1 nhóm có tiền tố admin và thực hiện kiểm tra.

Ví dụ:

*Route::group(array(“prefix”=>”admin”,”before”=>”checklogin”), function(){*

*// phần sử lý nằm ở đây.*

*});*

Với ví dụ trên thì mình đang nhóm tất cả các link có tiền tố là admin, và trước khi truy cập vào những link này mình sẽ thực hiện gọi tới Filter checklogin để kiểm tra xem đã đăng nhập chưa.

**Chương 4: VIEW TRONG LARAVEL FRAMEWORK**

Như đã đề cập ở trên thì thư mục view dùng để chứa các file mẫu như html để tương tác với người sử dụng nằm trong thư mục (app/view).

Để sử dụng view ta có thể sử dụng như sau:

*return View::make("Tên",Đối số nếu có);*

Ví dụ trong app/route.php ta có đường dẫn sau:

*Route::get("name",function(){*

*return View::make("hoten"); // gọi file view có tên là: app/views/hoten.php*

*});*

*<!-- View stored in app/views/hoten.php -->*

*<html>*

*<body>*

*<h1>Hello, <?php echo $name; ?></h1>*

*</body>*

*</html>*

Sau đó ta tạo file app/views/hoten.php với nội dung là ” Tên tôi là Tính” khi chạy thử đường link http://localhost/kienthuc24h/public/name thì ta sẽ thấy kết quả.

Vậy nếu ta muốn truyền đối số cho view thì sao. Bạn có nhiều cách để làm điều đó.

+ Cách 1: Sử dụng đối số

*Route::get("name2",function(){*

*$data['name']= "Bui Van Tinh";*

*return View::make("hoten",$data);*

*});*

Với cách này bạn đang truyền sang view hoten.php một đối số có tên là $name. Do vậy tại hoten.php bạn có thể dùng nó như sau:

*echo "Họ tên của bạn là: $name";*

+ Cách 2: Truyền qua 1 phương thức tên with().

*Route::get("name3",function(){*

*return View::make('hoten')->with('name', 'Steve');*

*});*

Và ở view hoten.php ta hoàn toàn có thể sử dụng $name dễ dàng. Nhưng với cách này, bạn sẽ rất khó để truyền nhiều giá trị cho view.

+ Cách 3: Sử dụng hàm compact() trong PHP.

*Route::get("name4",function(){*

*$name = "Framgia";*

*return View::make("hoten",compact("name"));*

*});*

Và ở view abcd.php bạn chỉ việc sử dụng $id một cách dễ dàng.

Trong trường hợp bạn muốn thao đặt view trong thực mục cho gọn gàng. Bạn có thể sử dụng phân cách, thay vì là dấu / thì bạn sẽ dùng dấu “.”.

*Route::get("name",function(){*

*return View::make("demo.hoten");*

*});*

**Chương 5: CONTROLLER TRONG LARAVEL FRAMEWORK**

Như các bạn được biết trong Laravel cho phép chúng ta có thể xử lý dữ liệu ngay bên trong Route nhưng đối với những thao tác phức tạp và dài chúng ta lên xử lý trong Controller. Có như vậy thì ứng dụng của bạn mới trở nên linh hoạt và dễ mở rộng sau này.

**5.1. Khai báo Controller trong Laravel Framework**

Để tạo 1 controller các bạn cần vào thư mục app/controller với các quy tắc sau:

1. Tên file trùng với tên Class.
2. Kế thừa từ lớp BaseController.

Ví dụ, tạo 1 file HocphpController.php với nội dung:

*<?php*

*class HocphpController extends BaseController {*

*public function test(){*

*echo 'Tim hieu ve Controller trong Laravel tai kienthuc24h';*

*}*

*}*

**5.2. Chạy Controller theo phương thức Route::get**

Trước tiên các bạn truy cập vào file routes.php thêm nội dung sau:

Route::get('/controller-test', 'HocphpController@test');

Với đoạn code cấu hình trên ta đã khai báo 1 link controller-test sẽ gọi tới controller HocphpController và phương thức(action) là test.Như vậy khi các chạy link localhost/kienthuc24h/public/controller-test sẽ hiển thị nội dung như ý muốn:

*Tim hieu ve Controller trong Laravel tai kienthuc24*

Chúng ta sẽ đi thêm vào 1 ví dụ nữa về việc truyền tham số vào controller thế nào nhé.Trong HocphpController tạo phương thức sau:

*<?php*

*class HocphpController extends BaseController {*

*public function view($title, $id)*

*{*

*echo 'Bạn đang xem bài viết '.$title.' có ID là :'.$id;*

*}*

*}*

Và file routes.php ta cấu hình như sau:

*Route::get('/bai-viet-{title}/{id}', 'HocphpController@view');*

Và khi các bạn chạy link localhost/laravel/public/bai-viet-hocphp/1 sẽ nhận được kết quả:

*Bạn đang xem bài viết hocphp có ID là :1*

Như vậy các bạn thấy chúng ta đã gửi thành công 2 biến đó chính là $title = hocphp và $id = 1.

**5.3. Implicit Controllers – Sử dụng phương thức Route::controller**

Như các bạn thấy trong 1 controller chúng ta thường có khá nhiều các phương thức (action) phải không nào, như vậy chả lẽ mỗi lần muốn chạy 1 action nào đó lại phải vào file routes.php để thêm cấu hình sao? Như vậy cũng thật bất tiện và khó quản lý phải không nào. Trong Larvel còn cho phép chúng ta có thể chạy các phương thức trong controller 1 cách tự động dựa vào tên action đó.

* Trước tiên các bạn vào file routes.php thêm đoạn code cấu hình như sau:

*Route::controller('hocphp', 'HocphpController');*

Với phương thức Route::controller cho phép chúng ta sử dụng 2 đối số, đối số thứ nhất chính là định dạnh URI, đối số thứ 2 chính là tên lớp controller. Trong ví dụ trên mình đã định nghĩa 1 URI có tên hocphp đại diện cho lớp controller HocphpController.

Tiếp theo chúng ta cần chỉnh sửa nội dung file controller như sau:

*<?php*

*class HocphpController extends BaseController {*

*public function getTest(){*

*echo 'Tim hieu ve Controller trong Laravel tai kienthuc24h';*

*}*

*}*

Như các bạn thấy nếu chúng ta cấu hình theo phương thức Route::controller trong routes thì trong controller chúng ta cũng phải viết đúng theo các quy tắc trong Laravel như sau:

* Trước các action phải có các tiền tố là các giá trị get, post, any…các giá trị này các bạn có thể xem lại về Route phía trên và tên action phải viết hoa chữ cái đầu.
* Tên phương thức phân biệt chữ hoa và chứ thường.
* Khi muốn chạy action thì chạy theo dạng link URI/tên action viết thường.

Và lúc này chúng ta muốn chạy action test thì sẽ chạy đường link sau: locahost/kienthuc24h/public/hocphp/test, thì sẽ được kết quả như mong đợi.

**5.4. RESTful Resource Controllers – Sử dụng phương thức Route::resource**

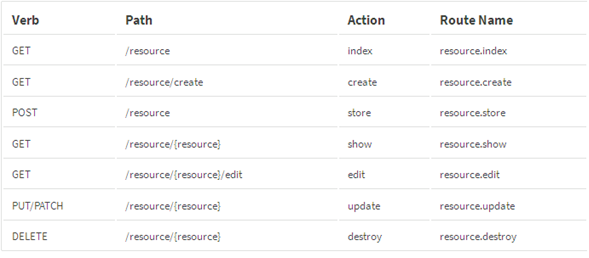
* Để lập một controller qua câu lệnh command line, thì ta thực hiện:

*php artisan controller:make HocphpController*

* Bây giờ bạn có thể xử lý một resourceful route tới controller:

*Route::resource('hocphp', 'HocphpController');*

Việc lập route đơn này sẽ xử lý nhiều route của RESTful actions trên nguồn hocphp. Dưới đây là bảng ánh xạ giữa route và action:



Hình 5.4.1: Ánh xạ giữa route và action

Và dưới đây là 1 ví dụ đơn giản trong controller HocphpController:

*<?php*

*class HocphpController extends BaseController {*

*/\*\**

*\* Hocphp Repository*

*\**

*\* @var Hocphp*

*\*/*

*protected $Hocphp;*

*/\*\**

*\* Display a listing of the resource.*

*\**

*\* @return Response*

*\*/*

*public function index()*

*{*

*}*

*\* Show the form for creating a new resource.*

*\**

*\* @return Response*

*\*/*

*public function create()*

*{*

*}*

*/\*\**

*\* Store a newly created resource in storage.*

*\**

*\* @return Response*

*\*/*

*public function store()*

*{*

*}*

*/\*\**

*\* Display the specified resource.*

*\**

*\* @param int $id*

*\* @return Response*

*\*/*

*public function show($id)*

*{*

*}*

*/\*\**

*\* Show the form for editing the specified resource.*

*\**

*\* @param int $id*

*\* @return Response*

*\*/*

*public function edit($id)*

*{*

*}*

*/\*\**

*\* Update the specified resource in storage.*

*\**

*\* @param int $id*

*\* @return Response*

*\*/*

*public function update($id)*

*{*

*}*

*/\*\**

*\* Remove the specified resource from storage.*

*\**

*\* @param int $id*

*\* @return Response*

*\*/*

*public function destroy($id)*

*{*

*}*

*}*

Như vậy những việc quản lý dữ liệu (hiển thị, thêm, sửa, xóa) các bạn có thể viết tại controller này.

Ngoài ra thì bạn có thể giới hạn được những action nào được thực thi:

*Route::resource('hocphp', 'HocphpController',*

*array('only' => array('index', 'show')));*

*Route::resource('hocphp', 'HocphpController',*

*array('except' => array('create', 'store', 'update', 'destroy')));*

**5.5. Controller với Filter**

Trong phần Route phía trên mình đã giới thiệu về filters trong Laravel.Đây chính là phần bộ lọc định tuyến, nó cho phép chúng ta có thể kiểm tra trước khi truy cập vào đường dẫn đó.Như vậy các bạn hoàn toàn có thể sử dụng filters để lọc dữ liệu với Controller.Bạn có thể khai báo trên file route như sau:

*Route::get('add', array('before' => 'auth',*

*'uses' => 'HocphpController@store'));*

* Ngoài ra các bạn còn có thể sử dụng filters ngay tại trong Controller, ví dụ:

*<?php*

*class HocphpController extends BaseController {*

*public function \_\_construct()*

*{*

*$this->beforeFilter('auth', array('except' => 'getLogin'));*

*$this->beforeFilter('csrf', array('on' => 'post'));*

*}*

*}*

* Bạn cũng có thể chỉnh định cho filter bằng cách sử dụng Closure:

*<?php*

*class HocphpController extends BaseController {*

*/\*\**

*\* Instantiate a new UserController instance.*

*\*/*

*public function \_\_construct()*

*{*

*$this->beforeFilter(function()*

*{*

*//code here*

*});*

*}*

*}*

* Nếu bạn muốn sử dụng một phương thức khác làm filter thì ban có thể thêm @ để định nghĩa filter:

*<?php*

*class HocphpController extends BaseController {*

*public function \_\_construct()*

*{*

*$this->beforeFilter('@filterRequests');*

*}*

*public function filterRequests($route, $request){*

*//code here*

*}*

*}*

*}*

* Như vậy các bạn thấy việc sử dụng Filter cho Controller có rất nhiều cách phải không nào.

**Chương 6: KẾT NỐI DATABASE**

Laravel kết nối tới cơ sở dữ liệu và chạy truy vấn cực kỳ đơn giản. Cấu hình nằm tại file app/config/database.php. Trong file này thì định nghĩa tất cả những loại kết nối cơ sở dữ liệu, tất nhiên là chúng cũng có loại cơ sở kết nối mặc định. Những ví dụ dưới đây được áp dụng cho tất cả những loại cơ sở dữ liệu mà được cung cấp trong file này.Hiện tại thì Laravel hổ trợ những hệ thống cơ sở dữ liệu sau: MySQL, Postgres, SQLite, và SQL Server.

**6.1. Cấu hình kết nối cơ sở dữ liệu trong Laravel Framework**

* Các bạn mở file app/config/database.php và tìm tới dòng cấu hình:

'mysql' => array(

'driver'    => 'mysql',

'host'      => 'localhost',

'database'  => 'laravel',

'username'  => 'root',

'password'  => '',

'charset'   => 'utf8',

'collation' => 'utf8\_unicode\_ci',

'prefix'    => '',

),

* Với đoạn cấu hình trên thì các bạn cần quan tâm cho mình 4 thông số cấu hình sau:

1. host: Cấu hình host,các bạn giữ nguyên là localhost.
2. database: Cấu hình tên cơ sở dữ liệu mà bạn đang sử dụng.
3. username: Tài khoản có quyền truy cập vào cơ sở dữ liệu.
4. password: Mật khẩu của tài khoản ở trên.

* Do mình đang sử dụng Xampp lên mặc định username = 'root' và password = ' '.

Để phục vụ cho những đoạn code test này mình sẽ tạo 1 bảng dữ liệu sau:

*CREATE TABLE IF NOT EXISTS `users` (*

*`id` int(11) NOT NULL AUTO\_INCREMENT,*

*`name` varchar(128) CHARACTER SET utf8 NOT NULL,*

*PRIMARY KEY (`id`)*

*)ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8\_bin AUTO\_INCREMENT=12 ;*

* Thực hiện chạy câu lệnh truy vấn, sau khi đã kết nối vào hệ thống:

1. Chạy câu lệnh truy vẫn dữ liệu:

*$results = DB::select('select \* from users where id = ?', array(1));*

Với phương thức select chúng ta sẽ lấy ra được dữ liệu trả về dạng mảng.

1. Chạy câu lệnh thêm dữ liệu vào cơ sở dữ liệu:

*DB::insert('insert into users (id, name) values (?, ?)', array(1, 'hocphp'));*

1. Chạy câu lệnh cập dữ liệu:

*DB::update('update users set name = "nobitacnt" where id = ?', array('1'));*

1. Chạy câu lệnh xóa dữ liệu:

*DB::delete('delete from users');*

Ghi chú: Câu lệnh insert và câu lệnh delete luôn trả về con số record bị ảnh hưởng.

1. Câu lệnh xóa bảng:

*DB::statement('drop table users');*

**6.2. Database Transactions**

Để thao tác những câu lệnh bên trong transaction cơ sở dữ liệu, bạn có thể dùng phương thức transaction:

*DB::transaction(function()*

*{*

*DB::table('users')->update(array('id' => 1));*

*DB::table('posts')->delete();*

*});*

Ghi chú: Bất kỳ câu lệnh nào cũng được đưa vào transaction, và transaction cũng sẽ thực thi rollback một cách tự động.

1. Để bắt đầu 1 transaction ta thực hiện:

*DB::beginTransaction();*

1. Bạn cũng có thể rollback qua phương thức:

*DB::rollback();*

1. Cuối cùng thì bạn cần commit nó lên:

*DB::commit();*

Khi mà bạn sử dụng nhiều hơn 1 cơ sở dữ liệu và muốn thực hiện truy vấn thì các bạn có thể sử dụng phương thức:

*DB::connection*

Ví dụ mình có cấu hình như sau, với 2 cơ sở dữ liệu:

*//cơ sở dữ liệu 1là chính*

*'mysql' => array(*

*'driver'    => 'mysql',*

*'host'      => 'localhost',*

*'database'  => 'laravel',*

*'username'  => 'root',*

*'password'  => '',*

*'charset'   => 'utf8',*

*'collation' => 'utf8\_unicode\_ci',*

*'prefix'    => '',*

*),*

*//cơ sở dữ liệu 2*

*'mysql2' => array(*

*'driver'    => 'mysql',*

*'host'      => 'localhost',*

*'database'  => 'laravel2',*

*'username'  => 'root',*

*'password'  => '',*

*'charset'   => 'utf8',*

*'collation' => 'utf8\_unicode\_ci',*

*'prefix'    => '',*

*),*

Và kết nối mặc định của chúng ta sẽ kết nối tới cơ sở dữ liệu mysql, còn trong trường hợp muốn kết nối tới cơ sở dữ liệu 2 ta sử dụng:

*$users = DB::connection('mysql2')->select('select \* from user');*

*print\_r($users);*

**Chương 7: SỬ DỤNG QUERY BUILDER**

Query Builder giúp ta trong việc thao tác cỡ sở dữ liệu một cách thuận lợi hơn, với những câu lệnh truy vấn đơn giản thân thiện để tạo và chạy những truy vấn từ cơ sở dữ liệu (CSDL). Nó thường được sử dụng để thực thi hầu hết những thao tác về CSDL trong ứng dụng của bạn, và làm việc với tất cả những CSDL được hổ trợ. Query Builder sử dụng PDO (PHP Data Objects) nhằm bảo vệ ứng dụng và tránh các lỗi về SQL injection. Query Builder xây dựng lớp DB để thực hiện các câu truy vấn. Chúng ta sẽ cùng đi tìm hiểu các phương thức quan trọng trong lớp DB này:

* Để phục vụ cho loạt ví dụ ở dưới chúng ta cùng tạo 1 bảng dữ liệu để thực hành:

*CREATE TABLE IF NOT EXISTS `users` (*

*`id` int(11) NOT NULL AUTO\_INCREMENT,*

*`name` varchar(128) CHARACTER SET utf8 NOT NULL,*

*PRIMARY KEY (`id`)*

*)ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8\_bin AUTO\_INCREMENT=12 ;*

**7.1. Truy vấn dữ liệu**

* + 1. **Lấy tất cả dữ liệu trong bảng**

*$users = DB::table('users')->get();*

*print\_r($users);*

* Và sẽ có kết quả như sau:

*Array(*

*[0] => stdClass Object*

*(*

*[id] => 12*

*[name] => nobita*

*)*

*[1] => stdClass Object*

*(*

*[[id] => 13*

*[name] => xuka*

*)*

*)*

* + 1. **Truy vấn lấy 1 dòng dữ liệu từ bảng CSDL**

*$user = DB::table('users')->where('id', '12')->first();*

*print\_r($user);*

* Và kết quả như sau:

*stdClass Object(*

*[id] => 12*

*[name] => nobita*

*)*

* + 1. **Truy vấn một cột từ một hàng đơn**

*$name = DB::table('users')->where('id', '12')->pluck('name');*

*echo $name;*

* Và kết quả sẽ trả về là: nobita
  + 1. **Truy vấn danh sách từ 1 bảng giá trị**

*$users = DB::table('users')->lists('name');*

*print\_r($users);*

* Và kết quả sẽ như sau:

*Array(*

*[0] => nobita*

*[1] => xuka*

*)*

* Nếu các bạn muốn lấy thêm nhiều trường dữ liệu thì có thể sử dụng:

*$users = DB::table('users')->lists('name','email','phone');*

*print\_r($users);*

* + 1. **Sử dụng mệnh đề select**

*$users = DB::table('users')->select('id', 'name')->get();*

*$users = DB::table('users')->distinct()->get();*

*$users = DB::table('users')->select('name as user\_name')->get();*

* + 1. **Sử dụng từ khóa where, thêm điều kiện**

*$users = DB::table('users')->where('id', '>', 100)->get();*

* Hoặc câu lệnh orWhere:

*$users = DB::table('users')*

*->where('id', '>', 100)*

*->orWhere('name', 'nobita')*

*-> get();*

* Sử dụng where Between:

*$users = DB::table('users')*

*->whereBetween('id', array(1, 100))->get();*

* Sử dụng where not between:

*$users = DB::table('users')*

*->whereNotBetween('id', array(1, 100))->get();*

* Sử dụng where trong 1 mảng hoặc không thuộc 1 mảng:

*$users = DB::table('users')*

*->whereIn('id', array(1, 2, 3))->get();*

*$users = DB::table('users');*

*->whereNotIn('id', array(1, 2, 3))->get();*

* Sử dụng where với giá trị null hoặc có giá trị trả về nhưng lại không đặt giá trị:

*$users = DB::table('users')*

*->whereNull('name')->get();*

* Đôi khi bạn cần tạo mệnh đề where phức tạp như “where exists”:

*DB::table('users')*

*->whereExists(function($query)*

*{*

*$query->select(DB::raw(1))*

*->from('orders')*

*->whereRaw('orders.user\_id = users.id');*

*})->get();*

Với câu lệnh trên sẽ sinh cho ta câu lệnh sql như sau: *select \* from users where exists (select 1 from orders where orders.user\_id = users.id );*

* Hay những mệnh đề where lồng nhau:

*DB::table('users')*

*->where('name', '=', 'nobita')*

*->orWhere(function($query)*

*{*

*$query->where('id', '>', 100)*

*->where('email', '<>', 'nobitacnt@gmail.com');*

*}) ->get();*

Với câu lệnh trên sẽ sinh cho ta câu lệnh sql như sau:

*select \* from users where name = 'nobita' or (id > 100 and email <> 'nobitacnt@gmail.com');*

* + 1. **Sử dụng Order By, Group By, và Having**

*users = DB::table('users')*

*->orderBy('name', 'desc')*

*->groupBy('count')*

*->having('count', '>', 100)*

*->get();*

* + 1. **Offset & Limit**

*$users = DB::table('users')->skip(10)->take(5)->get();*

**7.2. Join bảng dữ liệu**

* Câu lệnh Join cơ bản

*DB::table('users')*

*->join('contacts', 'users.id', '=', 'contacts.user\_id')*

*->join('orders', 'users.id', '=', 'orders.user\_id')*

*->select('users.id', 'contacts.phone', 'orders.price')*

*->get();*

* Câu lệnh join trái

*DB::table('users')*

*->leftJoin('orders', 'users.id', '=', 'orders.user\_id')*

*->get();*

* Bạn cũng có thể nâng cấp với mệnh đề join

*DB::table('users')*

*->join('orders', function($join)*

*{*

*$join->on('users.id', '=', 'orders.user\_id')->orOn(...);*

*})*

*->get();*

* Sử dụng where trong mệnh đề join

*DB::table('users')*

*->join('orders', function($join)*

*{*

*$join->on('users.id', '=', 'orders.user\_id')*

*->where('orders.user\_id', '>', 5);*

*})*

*->get();*

**7.3. Hàm trong Laravel Framework**

Trong Laravel cũng có những hàm trong SQL như:

*$users = DB::table('users')->count();*

*$price = DB::table('orders')->max('price');*

*$price = DB::table('orders')->min('price');*

*$price = DB::table('orders')->avg('price');*

*$total = DB::table('users')->sum('votes');*

**7.4. Raw Expressions**

Trong một số trường hợp ta cần sử dụng mã SQL nguyên mẫu trong loạt truy vấn của mình. Laravel cung cấp cho bạn phương thức DB::raw:

*$users = DB::table('users')*

*->select(DB::raw('count(\*) as user\_count, status'))*

*->where('status', '<>', 1)*

*->groupBy('status')*

*->get();*

Tăng hoặc giảm giá trị của một cột:

*DB::table('users')->increment('votes');*

*DB::table('users')->increment('votes', 5);*

*DB::table('users')->decrement('votes');*

*DB::table('users')->decrement('votes', 5);*

**7.5. Câu lệnh Inserts dữ liệu**

* Chèn một hàng giá trị vào 1 bảng

*DB::table('users')->insert(*

*array('email' => 'john@example.com', 'votes' => 0)*

*);*

* Nếu bảng có id tự động tăng, thì bạn dùng insertGetId để chèn

*$id = DB::table('users')->insertGetId(*

*array('email' => 'john@example.com', 'votes' => 0)*

*);*

* Thêm nhiều records vào một bảng

*DB::table('users')->insert(array(*

*array('email' => 'taylor@example.com', 'votes' => 0),*

*array('email' => 'dayle@example.com', 'votes' => 0),*

*));*

**7.6. Câu lệnh Updates dữ liệu**

*DB::table('users')*

*->where('id', 1)*

*->update(array('votes' => 1));*

**7.7. Câu lệnh Deletes dữ liệu**

* Xóa 1 record tại một bảng

*DB::table('users')->where('votes', '<', 100)->delete();*

* Xóa toàn bộ record trên bảng

*DB::table('users')->delete();*

* Xóa nội dung của bảng

*DB::table('users')->truncate();*

**7.8. Câu lệnh Union**

*$first = DB::table('users')->whereNull('first\_name');*

*$users = DB::table('users')->whereNull('last\_name')->union($first)->get();*

**Chương 8: SỬ DỤNG ELOQUENT ORM**

Như đã đề cập ở phía trên thì Eloquent ORM (Object Relational Mapping) là một phương pháp lập trình để chuyển đổi từ mô hình database sang mô hình đối tượng, đây là thế mạnh của Laravel Framework. Để sử dụng được Eloquent ORM các bạn cần chú ý tới những vấn đề đơn giản sau:

+ Tất cả các file phải được viết trong thư mục app/models.

+ Các file model phải được kế thừa từ lớp Eloquent.

Và chúng ta sẽ đi tìm hiểu các phương thức quan trọng trong lớp Eloquent này:

* Trước tiên mình sẽ tạo 1 file model có tên User.php với nội dung sau nhé:

*class User extends Eloquent {*

*protected $table = 'users';*

*}*

+ Khi model được định nghĩa là chúng ta có thể thao tác trên nó, và lớp model đều phải kế thừa từ lớp Eloquent.

+ Thuộc tính $table sẽ khai báo bảng dữ liệu mà ta sẽ thao tác.

+ Lưu ý rằng cột updated\_at và created\_at nên được tạo tự động, nếu bạn không muốn nó tạo tự động ngày giờ thì bạn có thề đưa thuộc tính $timestamps về false nhé.

**8.1. Lấy dữ liệu**

* Lấy tất cả dữ liệu trong bảng users

*$users = User::all();*

*foreach ($user as $row)*

*{*

*echo $row->name.'<br />';*

*}*

* Lấy 1 dòng dữ liệu thông qua khóa chính

*$user = User::find(1);*

*echo $user->name;*

* Nếu không lấy được dữ liệu nào trả về từ khóa chính này thì kết quả sẽ là NULL.
* Đề đăng ký xử lý lỗi thì ta phải kích hoạt ModelNotFoundException.

*use Illuminate\Database\Eloquent\ModelNotFoundException;*

*App::error(function(ModelNotFoundException $e)*

*{*

*return Response::make('Not Found', 404);*

*});*

* Truy vấn sử dụng Eloquent Models

*$users = User::where('id', '>', 100)->take(10)->get();*

*foreach ($users as $user)*

*{*

*var\_dump($user->name);*

*}*

* Bạn cũng có thể sử dụng hàm trong query builder

*$count = User::where('id', '>', 100)->count();*

* Phương thức count() sẽ trả về tống số dòng tìm thấy phù hợp với câu điều kiện (id > 100).

**8.2. Thêm dữ liệu (Insert)**

* Gọi phương thức save:

*$user = new User;*

*$user->name = 'Nobita';*

*$user->save();*

* Sử dụng phương thức Create:

*//Create a new user in the database...*

*$user = User::create(array('name' => 'John'));*

*//Retrieve the user by the attributes, or create it if it doesn't exist...*

*$user = User::firstOrCreate(array('name' => 'John'));*

*//Retrieve the user by the attributes, or instantiate a new instance...*

*$user = User::firstOrNew(array('name' => 'John'));*

**8.3. Cập nhật dữ liệu (Update)**

* Sử dụng phương thức Save

*$user = User::find(1);*

*$user->name = 'Nobita';*

*$user->save();*

* Sử dụng phương thức Push

*$user->push();*

* Bạn có thể chạy câu lệnh update để truy vấn model theo điều kiện

*$affectedRows =User::where('id','>',100)->update(array('status' => 2));*

**8.4. Xóa dữ liệu (Delete)**

* Gọi phương thức delete dựa vào khóa chính của dữ liệu

*$user = User::find(1);*

*$user->delete();*

* Xóa bằng khóa

*User::destroy(1);*

*User::destroy(array(1, 2, 3));*

*User::destroy(1, 2, 3);*

* Xóa bằng kiểu query

*$affectedRows = User::where('id', '>', 100)->delete();*

**Chương 9: SỬ DỤNG MIGRATIONS VÀ SCHEMA BUILDER**

Thông thường khi chúng ta muốn tạo ra 1 cơ sở dữ liệu và các bảng dữ liệu chúng ta hay vào trong PhpMyadmin để thực hiện thủ công phải không nào. Nhưng đối với cách làm này thì nhiều lúc sẽ làm cho chúng ta mất khá nhiều thời gian trong việc tạo cũng như chỉnh sửa dữ liệu. Laravel cung cấp cho chúng ta 1 giải pháp rất tuyệt vời đó chính Migrations và Schema Builder. Từ khi làm việc với Laravel mình đã chuyển sang sử dụng cặp đôi Migrations và Schema Builder để thực hiện thiết kế và quản lý cơ sở dữ liệu (csdl).

Đầu tiên các bạn cần cấu hình các thông số kết nối cơ sở dữ liệu trong file /app/config/database.php, tên csdl của mình là: laravel\_demo.

*'mysql' => array(*

*'driver'    => 'mysql',*

*'host'      => 'localhost',*

*'database'  => 'laravel\_demo',*

*'username'  => 'forge',*

*'password'  => '',*

*'charset'   => 'utf8',*

*'collation' => 'utf8\_unicode\_ci',*

*'prefix'    => '',*

*),*

**9.1. Lớp Schema Builder – Thao tác với bảng CSDL**

Lớp Schema trong Laravel cung cấp cơ chế thao tác với bảng CSDL. Nó làm việc với tất cả CSDL mà Laravel hổ trợ, và thông qua những hàm API của hệ thống.

**9.1.1. Phương thức create – Tạo bảng dữ liệu**

*Schema::create('users', function($table){*

*$table->increments('id');*

*$table->string('email')->unique();*

*$table->string('password', 50);*

*$table->rememberToken();*

*$table->string('name');*

*$table->string('phone', 15);*

*$table->timestamps();*

*});*

Như các bạn thấy đối số đầu tiên trong phương thức create chính là tên bảng dữ liệu mà ta muốn tạo. Trong đối số thứ 2 chính là nơi ta thực hiện khai báo các cột dữ liệu trong bảng này.

Mình xin giải thích 1 chút về các lệnh hay sử dụng ở trên:

+ increments: Tạo khóa chính của bảng dữ liệu,và khóa này tự tăng.

+ string: Tạo cột dạng chuỗi (giống kiểu varchar).

+ rememberToken: Tạo 1 columns tên là remember\_token trong table users, nó sẽ dùng để lưu token khi người dùng đăng nhập sử dụng ghi nhớ lần nhau – dạng VARCHAR(100).

+ timestamps: Lệnh này sẽ tạo 2 columns là created\_at và update\_at trong table users, nó là 2 trường mặc định để lưu thời gian tạo và sửa của model. Nếu bạn không tạo 2 trường này, mặc định khi insert 1 bản ghi Laravel 4 sẽ báo lỗi. Nếu bạn vẫn không muốn sử dụng 2 cột này, thì bạn vào model của bảng này và thêm vào: public Stimestamps=fale; nó sẽ không mặc định thêm 2 dữ liệu này vào bản ghi nữa.

Ngoài ra còn rất nhiều lệnh khác được sử dụng khi tạo bảng dữ liệu, các bạn có thể vào trang chủ laravel để xem thêm.

**9.1.2. Phương thức rename – Đổi tên bảng dữ liệu**

*Schema::rename($from, $to);*

Ví dụ: đổi tên bảng từ users sang user ta sử dụng.

*Schema::rename('users', 'user');*

**9.1.3. Phương thức drop – Xóa bảng dữ liệu**

*Schema::drop('users');*

*Schema::dropIfExists('users');*

**9.1.4. Phương thức table – Cập nhật thêm cột và một bảng tồn tại**

*Schema::table('users', function($table)*

*{*

*$table->string('address');*

*});*

Với ví dụ trên ta đang thực hiện thêm 1 cột address vào bảng users.

**9.1.5. Phương thức renameColumn – Đổi tên cột**

Đổi tên cột address thành address1

*Schema::table('users', function($table)*

*{*

*$table->renameColumn('address', 'address1');*

*});*

**9.1.6. Phương thức dropColumn - Xóa cột**

*Schema::table('users', function($table)*

*{*

*$table->dropColumn('address');*

*});*

Hoặc các bạn cũng có thể xóa nhiều cột dữ liệu, mỗi cột cách nhau bằng dấu “,”:

*Schema::table('users', function($table)*

*{*

*$table->dropColumn('address','phone','email');*

*});*

**9.1.7. Phương thức hasTable – Kiểm tra có tồn tại bảng không**

*if (Schema::hasTable('users'))*

*{*

*//code here*

*}*

**9.1.8. Phương thức hasColumn - Kiểm tra có tồn tại cột không**

*if (Schema::hasColumn('users', 'email'))*

*{*

*//code here*

*}*

Với đối số thứ nhất là tên bảng, đối số thứ 2 là tên cột muốn kiếm tra.

**9.1.9. Thêm chỉ mục vào**

*$table->string('email')->unique();*

**9.1.10. Sử dụng khóa ngoại**

*$table->integer('user\_id')->unsigned();*

*$table->foreign('user\_id')->references('id')->on('users');*

* Bạn cũng có thể cho “onDelete” và “onUpdate” trên ràng buộc của bạn:

*$table->foreign('user\_id')*

*->references('id')->on('users')*

*->onDelete('cascade');*

* Đề xóa khóa ngoại, bạn sử dụng phương thức dropForeign:

*$table->dropForeign('posts\_user\_id\_foreign');*

* Ngoài ra lớp còn hỗ trợ cho chúng ta rất nhiều các phương thức để làm việc với CSDL, các bạn có thể vào trang chủ laravel để xem thêm về cách sử dụng các phương thức này.

**9.2. Migrations – Quản lý CSDL**

Migrations là một kiểu điều khiển database, nó cho phép một nhóm có thể làm việc với CSDL một cách hiệu quả hơn. Migrations đi kèm với Schema Builder để thiết kế hệ thống của bạn một cách dễ dàng.

**9.2.1. Dùng artisan tạo migration**

* Để tạo migration chúng ta sẽ thao tác trên màn hình cmd.

Ví dụ:

*php artisan migrate:make create\_users\_table*

Sau khi chạy dòng lệnh trên hệ thống sẽ tự động tạo cho ta 1 file bên trong /app/database/ migratons theo định dạng: *yyyy\_mm\_dd\_hhmmss\_create\_users\_table.*

* Và nội dung trong file này sẽ như sau:

*<?php*

*use Illuminate\Database\Schema\Blueprint;*

*use Illuminate\Database\Migrations\Migration;*

*class CreateUsersTable extends Migration {*

/\*\*

\* *Run the migrations.*

*\**

*\*@return void*

*\*/*

*public function up()*

*{*

*}*

*/\*\**

*\* Reverse the migrations.*

*\**

*\* @return void*

*\*/*

*public function down()*

*{*

*}*

*}*

* Mặc định nó sẽ tạo cho chúng ta 1 lớp được kế thừa từ lớp Migration, và nó có 2 phương thức là up() và down(), function up dùng để thực thi migrate và down dùng để đảo ngược migrate.Hai quá trình này phải hoàn toàn ngược nhau. Ví dụ ở up() bạn create table users thì ở down() bạn drop table users, không làm gì hơn cả.
* Giờ sẽ tới bước chúng ta thực hiện tạo bảng dữ liệu users bên trong phương thức up() bằng các phương thức bên trong lớp Schema được hướng dẫn như phần:

*public function up()*

*{*

*Schema::create('users', function($table){*

*$table->increments('id');*

*$table->string('email')->unique();*

*$table->string('password', 50);*

*$table->rememberToken();*

*$table->string('name');*

*$table->string('phone', 15);*

*$table->timestamps();*

*});*

*}*

* Tiếp tục, ở down() bạn chỉ thêm lệnh xóa bảng đó đi:

*public function down()*

*{*

*Schema::drop('users');*

*}*

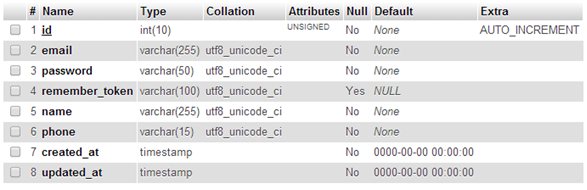
* Như vậy Việc thiết kế đã xong, bây giờ chúng ta sẽ sử dụng migrations để thực thi tập tin này bằng lệnh:

*php artisan migrate*



Hình 9.2.1. Thực hiện migrate bằng command line

* Bây giờ trong CSDL đã có table users với cái trường mình mong muốn. Bạn có thể tạo nhiều table cùng 1 lúc chứ không cần làm từng table 1. Đồng thời hệ thống sẽ tự động tạo thêm 1 bảng migrations nữa.



Hình 9.2.1. Bảng dữ liệu được tạo bằng lệnh migrate

**9.2.2. Chỉnh sửa migrations**

Ở đây chúng ta có thể thêm 1 cột dữ liệu hoặc chỉnh sửa 1 cột dữ liệu nào đó thì chúng ta có 2 cách:

+ Cách 1: các bạn tạo thêm 1 migrations mới, và sử dụng như ở trên, với cách này thì sẽ tạo ra khá nhiều file trong migrations khi chỉnh sửa nhiều.

+ Cách 2: Viết trực tiếp các chỉnh sửa hoặc thêm vào file migrations cũ sau đó chạy lệnh trên cửa sổ cmd: *php artisan migrate:refresh*

Và nó sẽ hỏi bạn 3 lần: có chạy lệnh này không? Có đồng ý chạy function down() ở các file không? Có đồng ý chạy function up() ở các file không? Lệnh này, nó sẽ làm lần lượt chạy function down() ở tất cả các migrate (file nằm trong app/database/migrations), sau đó chạy function up() ở tất cả các file.

Với 2 cách trên, mỗi cách có 1 ưu và nhược điểm khác nhau các bạn thấy cách nào phù hợp thì làm nhé.

**Chương 10: WEBSITE THI TRẮC NGHIỆM ONLINE**

**10.1. Khảo sát hiện trạng**

Cùng với sự bùng nổ của công nghệ thông tin và truyền thông, đào tạo trực tuyến (ĐTTT) đang là một loại hình cần được nhân rộng nhằm tạo cơ hội cho người học có thể tự học, tự nghiên cứu và trau dồi kiến thức.

Ứng dụng Công nghệ thông tin trong dạy và học đang là một trong những xu thế phát triển nhằm nâng cao chất lượng đào tạo ở bậc học Phổ thông và Đại học. So với các phương pháp giáo dục truyền thống, mô hình này là một bước phát triển vượt bậc của xã hội loài người, bởi nó tận dụng tối đa sức mạnh của công nghệ thông tin vào sự nghiệp giáo dục và đào tạo, bởi những lợi ích mà nó đem lại rất nhiều và hiệu quả như sau: g**iảm chi phí**, tự định hướng, tự điều chỉnh, linh hoạt, dễ tiếp cận và thuận tiện.

**10.2. Xác định yêu cầu**

**10.2.1. Yêu cầu chức năng**

* Yêu cầu chức năng nghiệp vụ
* Yêu cầu chức năng hệ thống
* Admin:
* Quản lý Giảng viên, Học viên của hệ thống.
* Quản lý môn học hệ thống.
* Quản lý chủ đề từng môn học.
* Phân công giảng dạy cho giảng viên.
* Đổi mật khẩu.
* Giảng viên:
* Quản lý ngân hàng câu hỏi theo môn học.
* Quản lý ngân hàng đề thì theo môn học.
* Quản lý tin tức, tài liệu.
* Đăng thông báo.
* Đổi mật khẩu.
* Guest:
* Xem tin tức, thông báo hệ thông đăng tải.
* Đăng ký trở thành học viên hệ thống.
* Học Viên:
* Thực hiện các bài kiểm tra trắc nghiệm của hệ thống.
* Xem lịch sử học tập, thông tin cá nhân.
* Tra cứu đề thi.

**10.2.2. Yêu cầu phi chức năng**

Bảng 10.2.2. Yêu cầu phi chức năng

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Nội dung | Tiêu chuẩn | Mô tả chi tiết |
| 1 | Tốc độ xử lý quá trình tìm kiếm đề thi nhanh. | Hiệu quả |  |
| 2 | Tốc độ load câu hỏi khi thực hiện đề thi phãi nhanh chóng ngay lập tức. | Hiệu quả | Tối đa 2s phải có kết quả tìm kiếm. |
| 3 | Giao diện thân thiện dễ sử dụng. | Tiện dụng | Giao diện phãi rõ rang và dễ dùng, không cần qua hướng dẫn chit tiết.  Người dùng ở mọi lứa tuổi đểu có thể dễ dàng sử dụng. |
| 4 | Dễ dàng thiết kế thêm chức năng mới. | Tiến hóa | Các doanh nghiệp có thể yêu cầu thêm tính năng hữu ích cho chương trình. |
| 5 | Dễ thao tác. | Tiện dụng | Các thao tác nhập xuất chỉnh sửa đơn giản, gần gũi với người dùng, quản lý. |
| 6 | Tin tức đăng tải, nội dung câu hỏi phãi chính xác, không sai lệch | Đúng đắn |  |

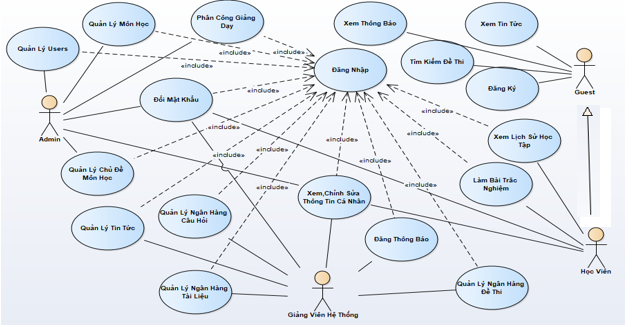
**10.2.3. Phân tích và mô hình hóa yêu cầu**

Bảng 10.2.3. Phân tích và mô hình hóa yêu cầu

|  |  |
| --- | --- |
| Tên tác nhân | Công việc/ Vai trò |
| Admin | * Quản lý User hệ thống( thêm, xóa, sửa giảng viên, học viên) * Xem thông tin cá nhân * Quản lý môn học hệ thống * Quản lý chủ đề môn học * Phân công giảng dạy * Đổi mật khẩu * Đăng xuất khỏi hệ thống |
| Giảng viên | * Tạo câu hỏi theo môn học * Tạo đề thi theo môn học từ ngân hàng câu hỏi * Đăng tài liệu bài giảng * Đăng thông báo * Đăng tin tức * Tra cứu thông tin cá nhân * Đổi mật khẩu * Đăng xuất khỏi hệ thống |
| Học Viên | * Tra cứu đề thi * Làm bài thi * Tra cứu điểm * Thông tin cá nhân * Xem thông báo, tài liệu bài giảng,tin tức * Đổi mật khẩu * Đăng xuất khỏi hệ thống |
| Guset | * Đăng ký để trở thành học viên hệ thống * Xem tin tức,thông báo, top điểm cao, tìm kiếm đề thi |

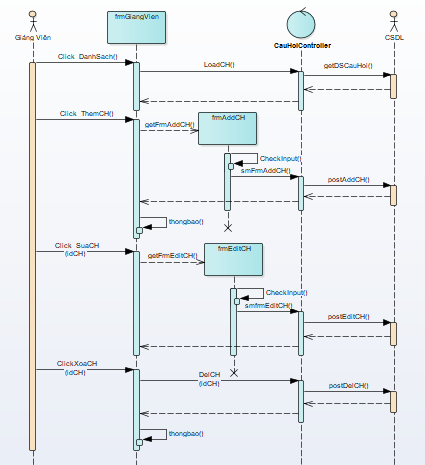
**10.3. Phân tích thiết kế cơ sở dữ liệu**

**10.3.1. Lược đồ Use Case**

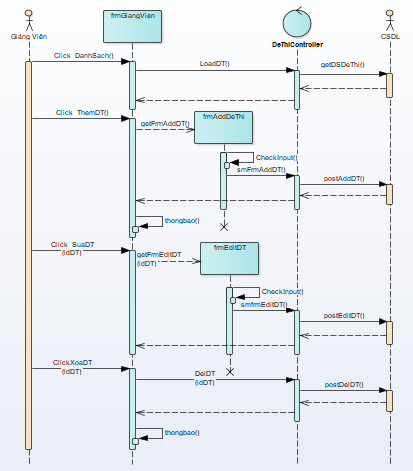


Hình 10.3.1. Lược đồ use-case

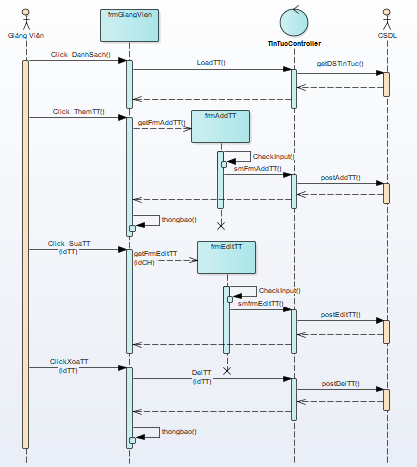
**10.3.2. Mô hình hóa yêu cầu**



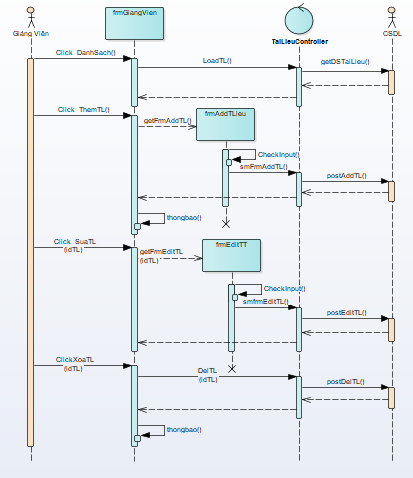
Hình 10.3.2. Lược đồ tuần tự quản lý ngân hàng câu hỏi



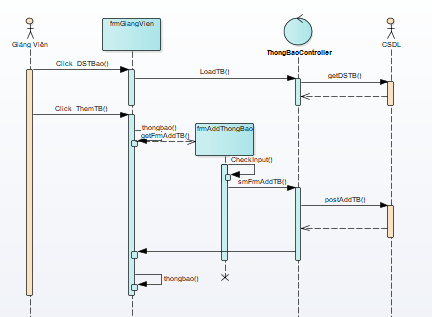
Hình 10.3.2. Lược đồ tuần tự quản lý ngân hàng đề thi



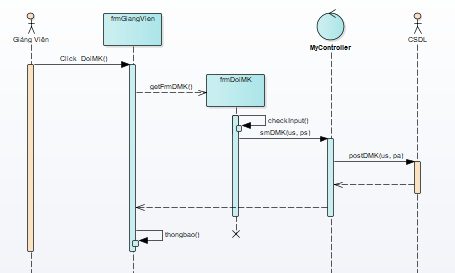
Hình 10.3.2. Lược đồ tuần tự quản lý tin tức



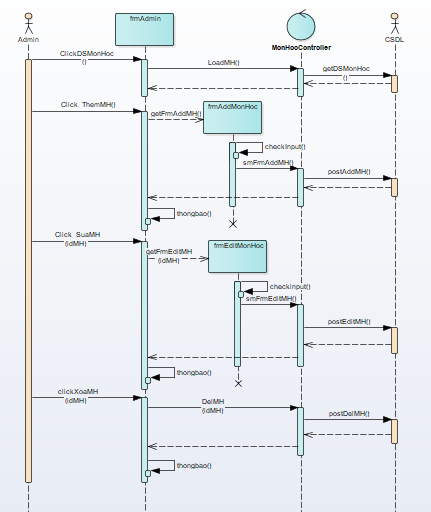
Hình 10.3.2. Lược đồ tuần tự quản lý tài liệu



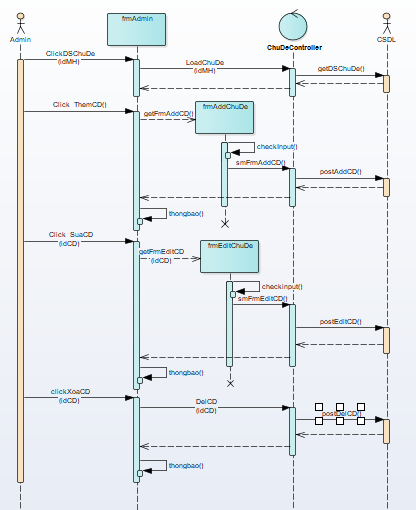
Hình 10.3.2. Lược đồ tuần tự đăng thông báo



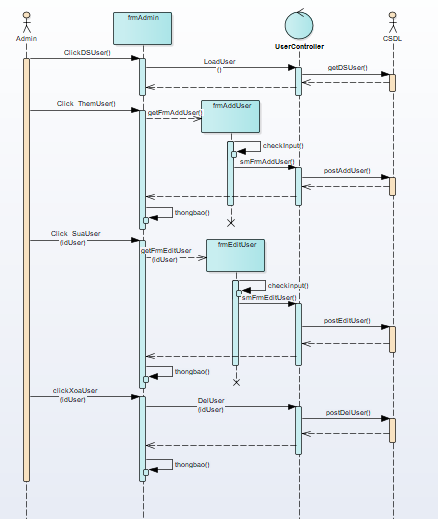
Hình 10.3.2. Lược đồ tuần tự đổi mật khẩu



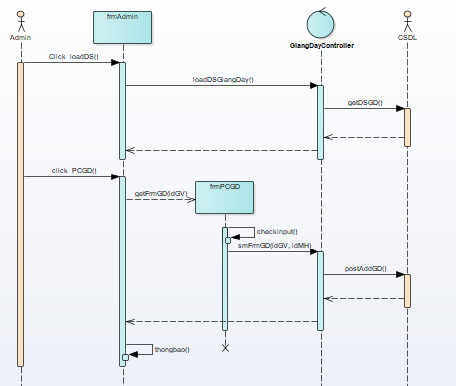
Hình 10.3.2. Lược đồ tuần tự quản lý môn học hệ thống



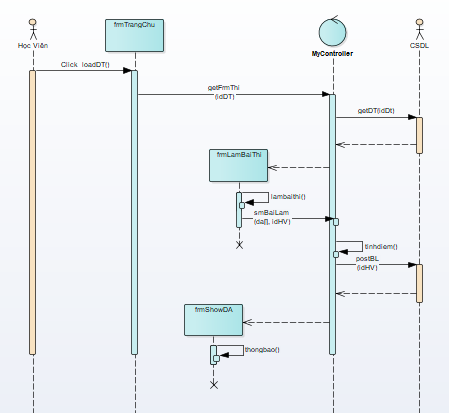
Hình 10.3.2. Lược đồ tuần tự quản lý chủ đề môn học



Hình 10.3.2. Lược đồ tuần tự quản lý users



Hình 10.3.2. Lược đồ tuần tự phân công giảng dạy



Hình 10.3.2. Lược đồ tuần tự làm bài thi

**10.3.3. Cơ sở dữ liệu**

* dbo.BangDiem

Lưu thông tin lịch sử học tập của học viên qua các lần làm bài.

Bảng 10.3.3. dbo.BangDiem

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idBangDiem | id mỗi lần thi | int | 10 ký tự | Khóa chính |
| 2 | TenDeThi | Tên đề thi | varchar() | 200 ký tự |  |
| 3 | idHocVien | Mã học viên | int | 10 ký tự |  |
| 4 | DiemSo | Điểm số lần thi | float | 6 ký tự |  |
| 5 | NgayLamBai | Ngày Làm Bài | timestamp |  |  |

* dbo.CauHoi

Lưu thông tin chi tiết của mỗi câu hỏi.

Bảng 10.3.3. dbo.CauHoi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idCauHoi | Mã câu hỏi | int | 10 ký tự | Khóa chính |
| 2 | idMonHoc | Mã môn học | int | 10 ký tự | Khóa ngoại |
| 3 | idLevel | Mã cấp độ | int | 10 ký tự | Khóa ngoại |
| 4 | idChuDe | Mã chủ đề | int | 10 ký tự | Khóa ngoại |
| 5 | NoiDung | Nội dung | varchar() | 500 ký tự |  |
| 6 | PhuongAnA | Phương án A | varchar() | 200 ký tự |  |
| 7 | PhuongAnB | Phương án B | varchar() | 200 ký tự |  |
| 8 | PhuongAnC | Phương án C | varchar() | 200 ký tự |  |
| 9 | PhuongAnD | Phương án D | varchar() | 200 ký tự |  |
| 10 | DapAn | Đáp án | varchar() | 200 ký tự |  |
| 11 | urlHinh | Tên hình ảnh câu hỏi | varchar() | 200 ký tự |  |
| 12 | NgayDuocTao | Ngày được tạo | date |  |  |
| 13 | idNguoiTao | Mã người tạo | int | 10 ký tự | Khóa ngoại |
| 14 | TrangThai | Trạng thái | int | 10 ký tự |  |
| 15 | ChuThich | Chú thích | varchar() | 500 ký tự |  |

* dbo.ChiTietDeThi

Lưu thông tin chi về những câu hỏi thuộc mỗi đề thi.

Bảng 10.3.3. dbo.ChiTietDeThi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idDeThi | Mã đề thi | int | 10 ký tự | Khóa chính |
| 2 | idCauHoi | Mã câu hỏi | int | 10 ký tự | Khóa chính |
| 3 | DapAn | Đáp án của câu hỏi | varchar() | 10 ký tự |  |

* dbo.ChuDe

Lưu thông tin các chủ đề của các môn học.

Bảng 10.3.3. dbo.ChuDe

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idChuDe | Mã chủ đề | int | 10 ký tự | Khóa chính |
| 2 | TenChuDe | Tên chủ đề | varchar() | 20 ký tự |  |
| 3 | idMonHoc | Mã môn học | int | 10 ký tự | Khóa ngoại |

* dbo.DeThi

Lưu thông tin chi tiết đề thi hệ thống.

Bảng 10.3.3. dbo.DeThi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idDeThi | Mã đề thi | int | 10 ký tự | Khóa chính |
| 2 | idMonHoc | Mã môn học | int | 10 ký tự | Khóa ngoại |
| 3 | TenDeThi | Tên đề thi | varchar() | 1000 ký tự |  |
| 4 | ThoiGian | Thời gian làm bài | int | 10 ký tự |  |
| 5 | SoLuongCau | Số lượng câu | int | 10 ký tự |  |
| 6 | SoLanThi | Số lần đã thi | int | 10 ký tự |  |
| 7 | NgayDuocTao | Ngày được tạo | date |  |  |
| 8 | idNguoiTao | Mã người tạo | int | 10 ký tự | Khóa ngoại |
| 9 | TrangThai | Trạng thái | int | 1. ý tự |  |

* dbo.GiangDay

Lưu thông tin các môn học do giảng viên hệ thống đảm nhiêm do admin phân công.

Bảng 10.3.3. dbo.GiangDay

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idGiangVienHT | Mã giảng viên | int | 10 ký tự | Khóa chính |
| 2 | idMonHoc | Mã môn học | int | 10 ký tự | Khóa chính |

* dbo.Level

Lưu thông tin các cấp độ của một câu hỏi.

Bảng 10.3.3. dbo.Level

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idLevel | Mã cấp độ | int | 10 ký tự | Khóa chính |
| 2 | NameLevel | Tên cấp độ | varchar() | 20 ký tự |  |

* dbo.MonHoc

Lưu thông tin chi tiết của môn học trong hệ thống

Bảng 10.3.3. dbo.MonHoc

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idMonHoc | Mã môn học | int | 10 ký tự | Khóa chính |
| 2 | TenMonHoc | Tên môn học | varchar() | 20 ký tự |  |

* dbo.TaiLieu

Lưu thông tin tài liệu giáo viên đăng tải.

Bảng 10.3.3. dbo.TaiLieu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idTaiLieu | Mã tài liệu | int | 10 ký tự | Khóa chính |
| 2 | TenTaiLieu | Tên tài liệu | varchar() | 200 ký tự |  |
| 3 | Type | Loại tài liệu | varchar() | 20 ký tự |  |
| 4 | urlFile | Link tài liệu | varchar() | 20 ký tự |  |
| 5 | idMonHoc | Mã môn học | Int | 10 ký tự | Khóa ngoại |
| 6 | idGiangVien | Mã giảng viên | Int | 10 ký tự | Khóa ngoại |
| 7 | NgayDang | Ngày đăng | date |  |  |

* dbo.ThongBao

Lưu thông tin chi tiết của các thông báo mà giảng viên đăng tải.

Bảng 10.3.3. dbo.ThongBao

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idThongBao | Mã thông báo | int | 10 ký tự | Khóa chính |
| 2 | TieuDe | Tiêu đề thông báo | varchar() | 200 ký tự |  |
| 3 | NoiDung | Nội dung | varchar() | 500 ký tự |  |
| 4 | idNguoiGui | Mã người đăng | int | 10 ký tự | Khóa ngoại |
| 5 | NgayDang | Ngày Đăng | timestamp |  |  |

* dbo.TinTuc

Lưu thông tin chi tiết của các tin tức được đăng tải.

Bảng 10.3.3. dbo.TinTuc

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idTinTuc | Mã tin tức | int | 10 ký tự | Khóa chính |
| 2 | TieuDe | Tiêu đề tin tức | varchar() | 200 ký tự |  |
| 3 | TomTat | Tóm tắt | varchar() | 500 ký tự |  |
| 4 | NoiDung | Nội dung | varchar() | 500 ký tự |  |
| 5 | NgayTao | Ngày Đăng | timestamp |  |  |
| 6 | AnHien | Ẩn hiện | int | 10 ký tự |  |
| 7 | idGiangVien | Mã người đăng | iint | 10 ký tự | Khóa ngoại |
| 8 | urlHinh | Link hình ảnh | varchar() | 20 ký tự |  |

* dbo.TypeAccount

Lưu thông tin các loại tài khoản.

Bảng 10.3.3. dbo.TypeAccount

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | idtype | Mã loại tài khoản | int | 10 ký tự | Khóa chính |
| 2 | tentype | Tên loại tài khoản | varchar() | 20 ký tự |  |

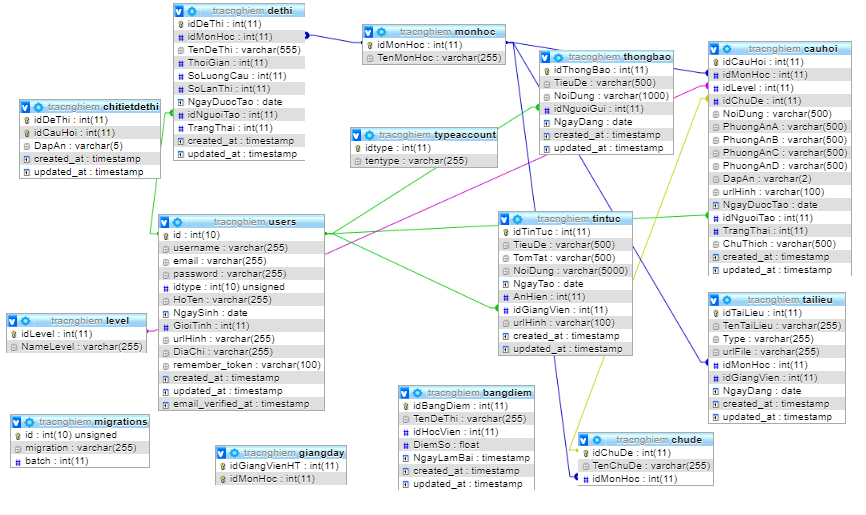
* dbo.Users

Lưu thông tin chi tiết các tài khoản người dùng.

Bảng 10.3.3. dbo.Users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Tên Thuộc Tính** | **Mô Tả** | **Kiểu dữ liệu** | **Kích thước** | **Ghi chú** |
| 1 | Id | Mã tài khoản | int | 10 ký tự | Khóa chính |
| 2 | Username | Tên đăng nhập | varchar() | 20 ký tự |  |
| 3 | Email | Email | varchar() | 50 ký tự |  |
| 4 | Password | Mật khẩu | varchar() | 200 ký tự |  |
| 5 | Idtype | Mã loại tài khoản | int | 10 ký tự | Khóa ngoại |
| 6 | HoTen | Họ tên | varchar() | 50 ký tự |  |
| 7 | NgaySinh | Ngày sinh | date |  |  |
| 8 | GioiTinh | Giới tính | int | 10 ký tự |  |
| 9 | urlHinh | Link hình đại diện | varchar() | 20 ký tự |  |
| 10 | DiaChi | Địa chỉ | varchar() | 200 ký tự |  |

**10.3.4. Lược đồ Diagram**



Hình 10.3.4. Lược đồ Diagram

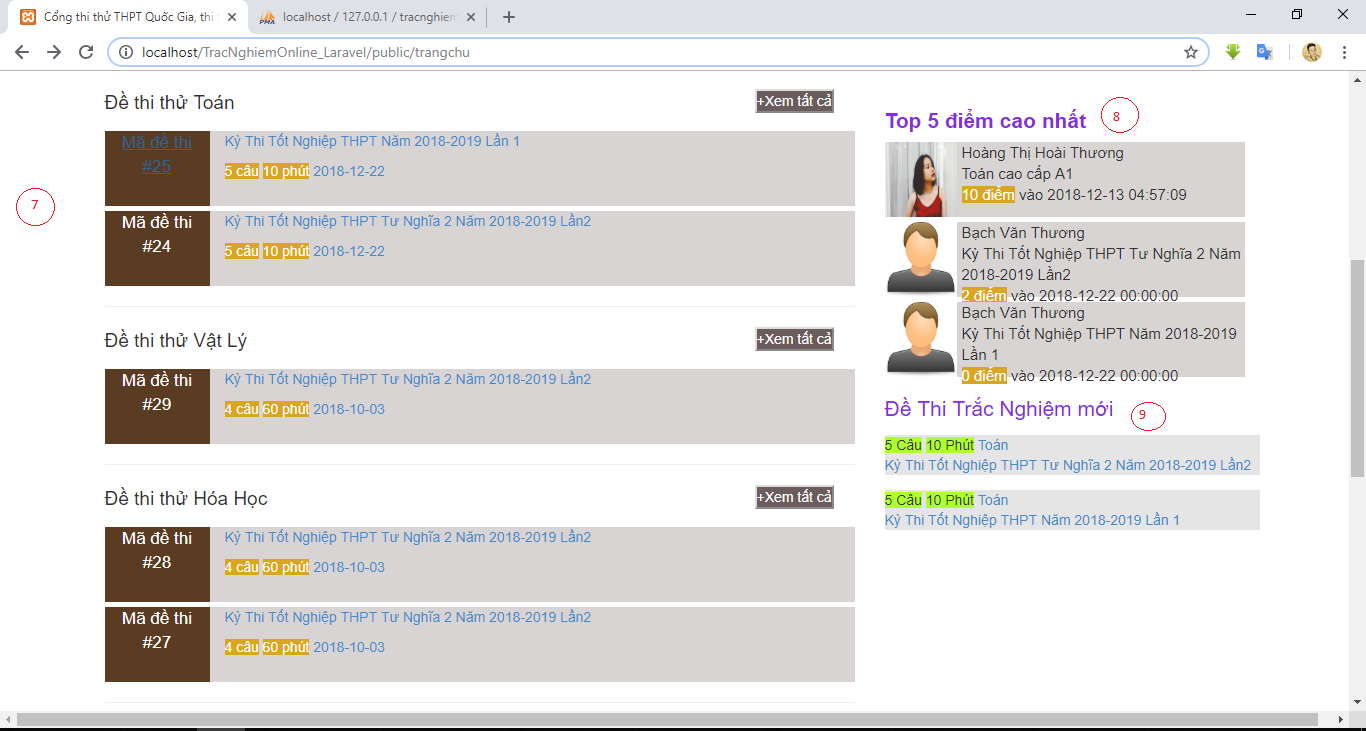
**10.4. Thiết kế giao diện**

**10.4.1. Giao diện trang chủ**

* Giao diện



Hình 10.4.1. Giao diện trang chủ 1



Hình 10.4.1. Giao diện trang chủ 2

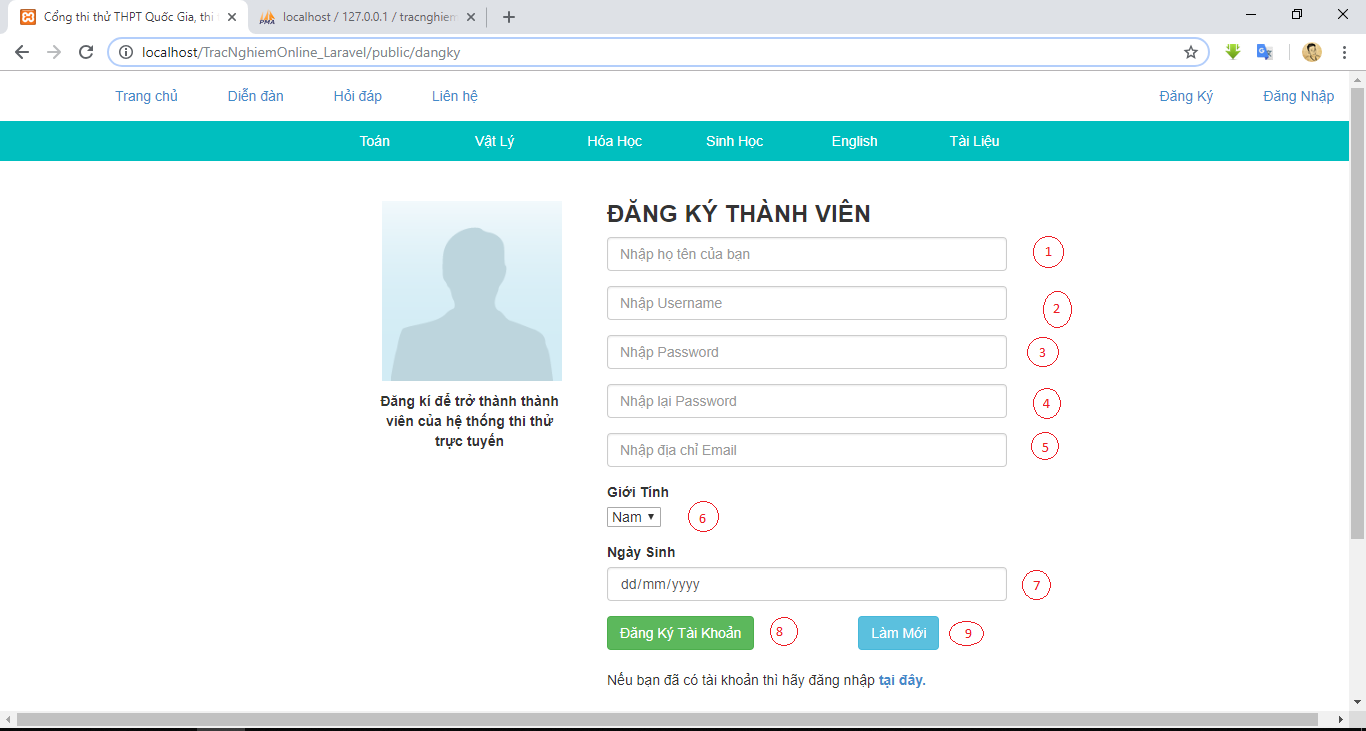
* Danh sách xử lý

Bảng 10.4.1. Xử lý giao diện trang chủ

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quay về trang chủ, giải đáp, liên hệ, diễn đàn | Click “Trang Chủ”, “Giải đáp “, “Liên hệ”, “diễn đàn” |  |
| 2 | Gọi trang đăng ký | Click chọn “Đăng Ký” |  |
| 3 | Gọi trang đăng nhập | Click chọn “Đăng Nhập” |  |
| 4 | Gọi đến trang chưa danh sách đề thi theo môn học mà người dùng chọn, và trang tài liệu | Click chọn môn học tương ứng trên menu chính hay tài liệu |  |
| 5 | Gọi đến trang tin tức hiển thị nội dung tin tức | Click chọn link tin tức |  |
| 6 | Gọi đến trang thông báo hiển thị nội dung thông báo | Click chọn link thông báo |  |
| 7 | Hiển thị một số đề thi theo từng môn học, sẽ gọi đến trang làm bài thi khi người dùng click | Click chọn vào tên đề thi |  |
| 8 | Hiên thị top 5 điểm cao |  |  |
| 9 | Hiển thị một số đề thi mới, sẽ gọi đến trang làm bài thi | Click chọn link tên đề thi |  |

**10.4.2. Giao diện trang đăng kí**

* Giao diện



Hình 10.4.2. Giao diện trang đăng ký

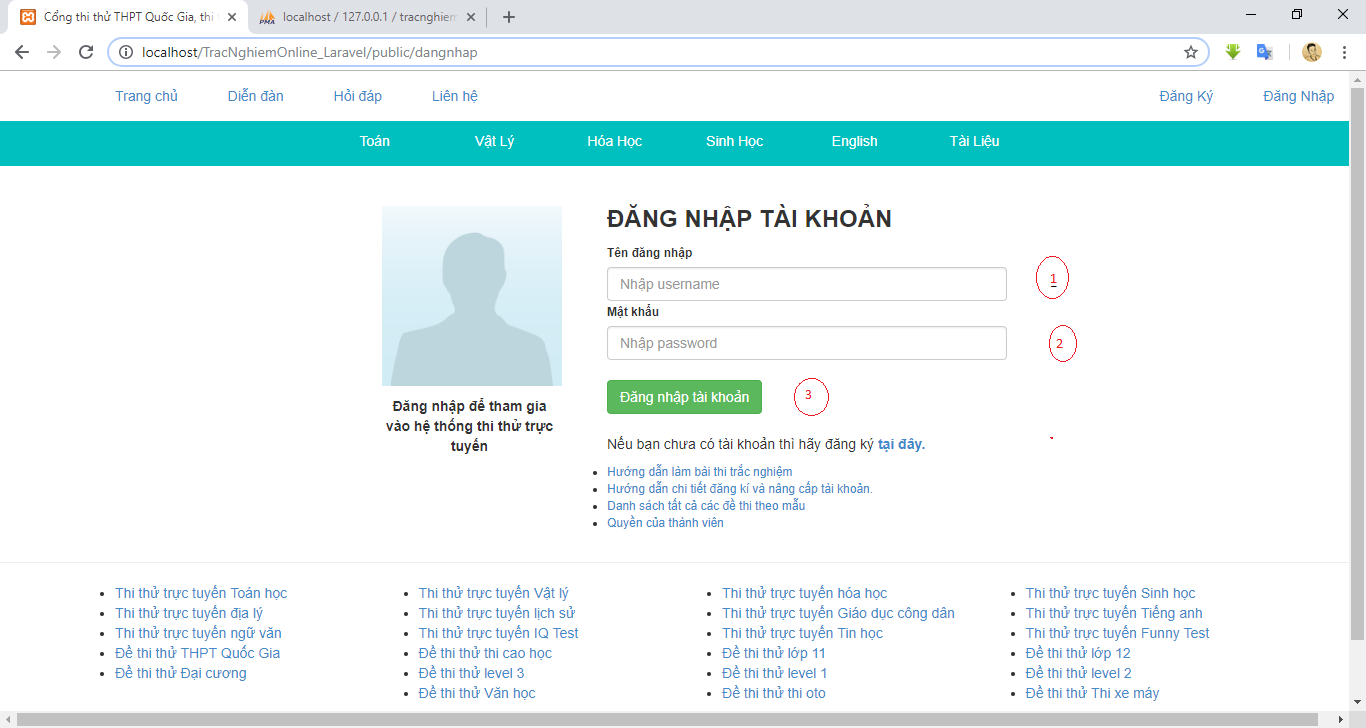
* Danh sách xử lý

Bảng 10.4.2. Xử lý giao diện trang đăng ký

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox để nhập họ tên học viên |  |  |
| 2 | Textbox để nhập username đăng nhập |  |  |
| 3 | Textbox để nhập Password |  |  |
| 4 | Textbox để nhập lại Password |  |  |
| 5 | Textbox để nhập gmail |  |  |
| 6 | Combobox chọn giới tính |  |  |
| 7 | Datetimepicker chọn ngày sinh |  |  |
| 8 | Button submit đăng ký | Các mục (1)(2)(3)(4)(5)(6)(7) nhập đầy đủ |  |
| 9 | Button làm mới |  | Clear các textbox |

**10.4.3. Giao diện đăng nhập**

* Giao diện



Hình 10.4.3. Giao diện trang đăng nhập

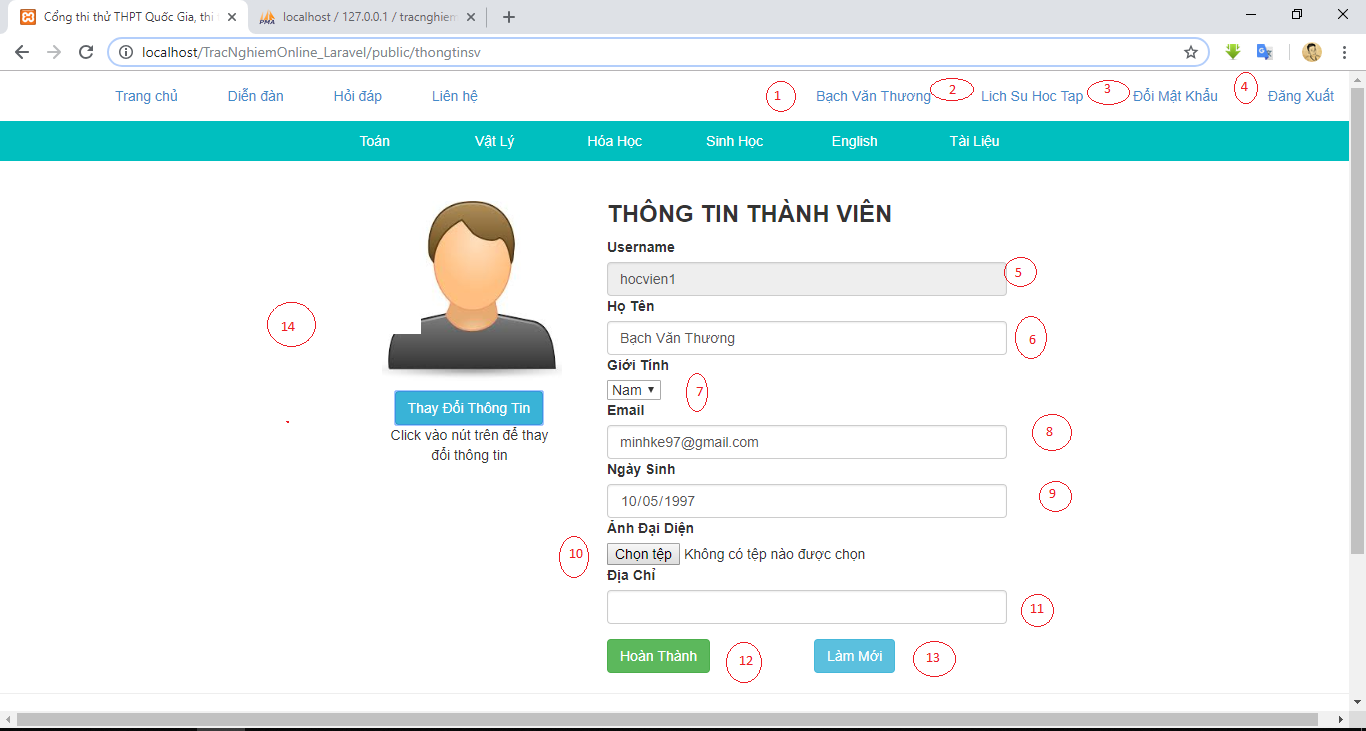
* Danh sách xử lý

Bảng 10.4.3. Xử lý giao diện trang đăng nhập

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox để nhập username đăng nhập |  |  |
| 2 | Textbox để nhập Password |  |  |
| 3 | Button submit đăng ký | Các mục (1)(2) nhập đầy đủ |  |

**10.4.4. Giao diện trang Thông tin cá nhân học viên**

* Giao diện



Hình 10.4.4. Giao diện thông tin cá nhân học viên

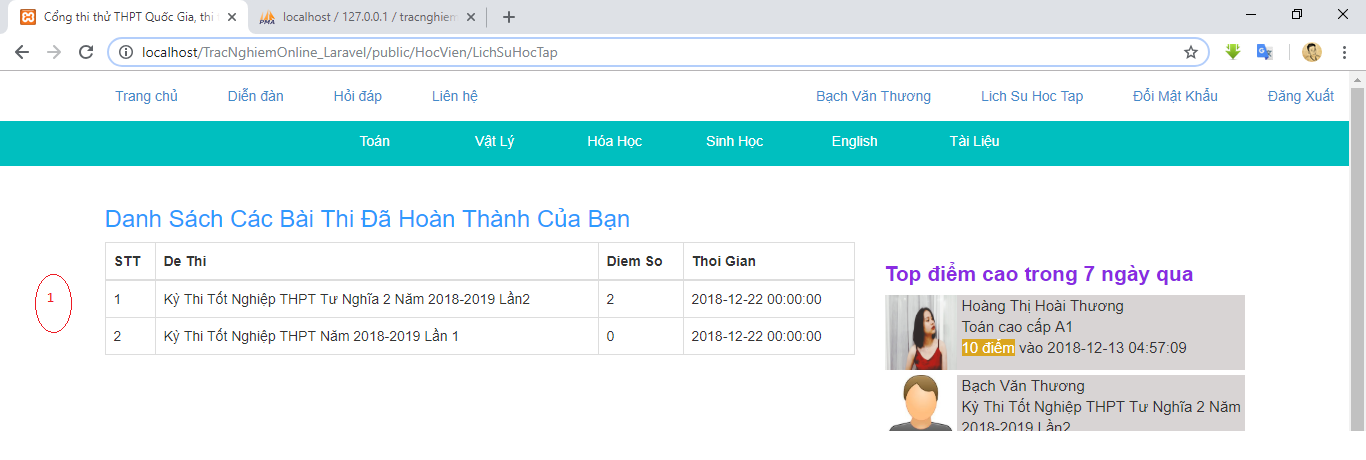
* Danh sách xử lý

Bảng 10.4.4. Xử lý giao diện trang thông tin cá nhân học viên

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Label hiển thị tên học viên | Sau khi đăng nhập thành công |  |
| 2 | Gọi trang lịch sử học tập | Sau khi đăng nhập thành công |  |
| 3 | Gọi trang đổi mật khẩu | Sau khi đăng nhập thành công |  |
| 4 | Đăng xuất tài khoản | Sau khi đăng nhập thành công |  |
| 5 | Textbox hiển thi username |  |  |
| 6 | Textbox hiển thi tên học viên |  |  |
| 7 | Combobox hiển thị giới tính |  |  |
| 8 | Textbox hiển thi gmail |  |  |
| 9 | Textbox hiển thi ngày sinh |  |  |
| 10 | Input file thay đổi ảnh đại diện |  |  |
| 11 | Textbox hiển thi địa chỉ |  |  |
| 12 | Button submit | Các mục (6)(7)(8)(9)(11) không được để trống |  |
| 13 | Button làm mới |  | Clear các textbox |
| 14 | Pictuerbox hiển thị hình ảnh đại diện |  |  |

**10.4.5. Giao diện trang lịch sử học tập**

* Giao diện



Hình 10.4.5. Giao diện trang lịch sử học tập

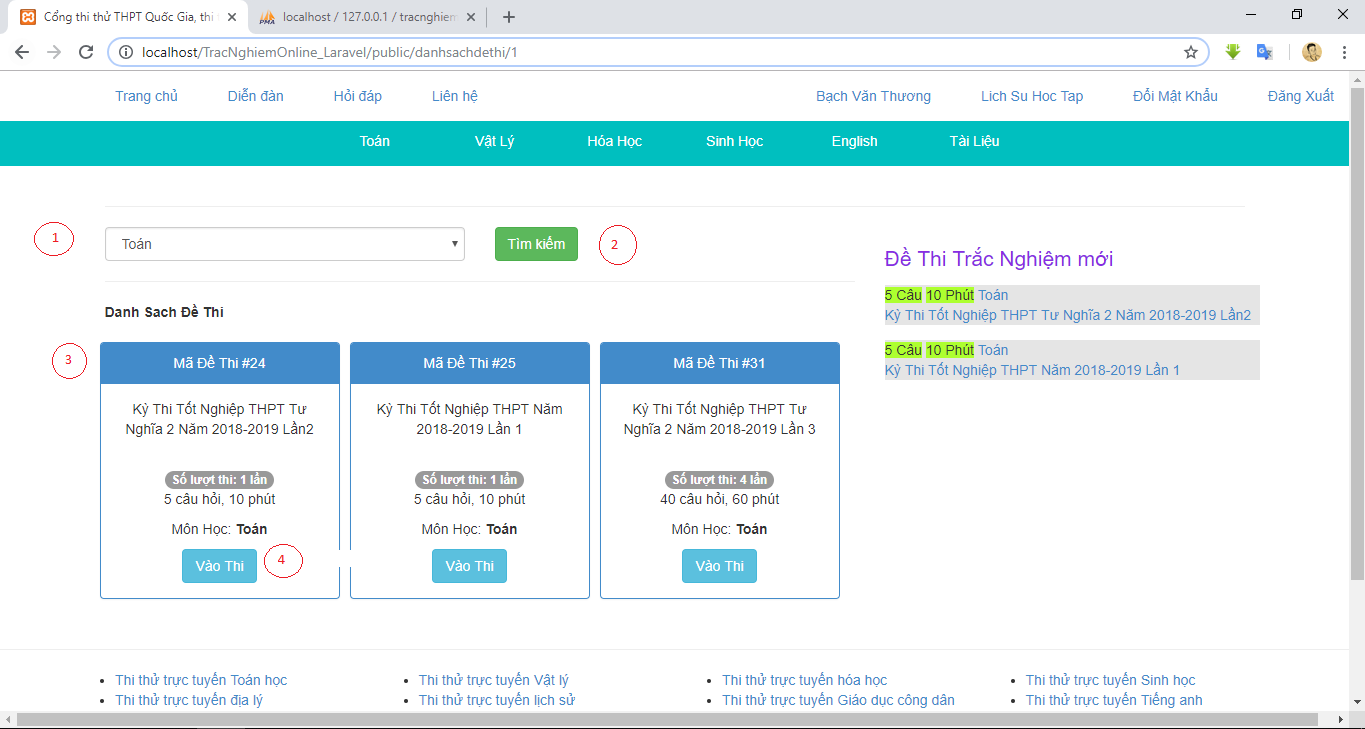
* Danh sách xử lý

Bảng 10.4.5. Xử lý giao diện trang lịch sử học tập

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Table hiển thị bảng điểm |  |  |

**10.4.6. Giao diện danh sách đề thi theo môn học**

* Giao diên



Hình 10.4.6. Giao diện danh sách đề thi theo môn học

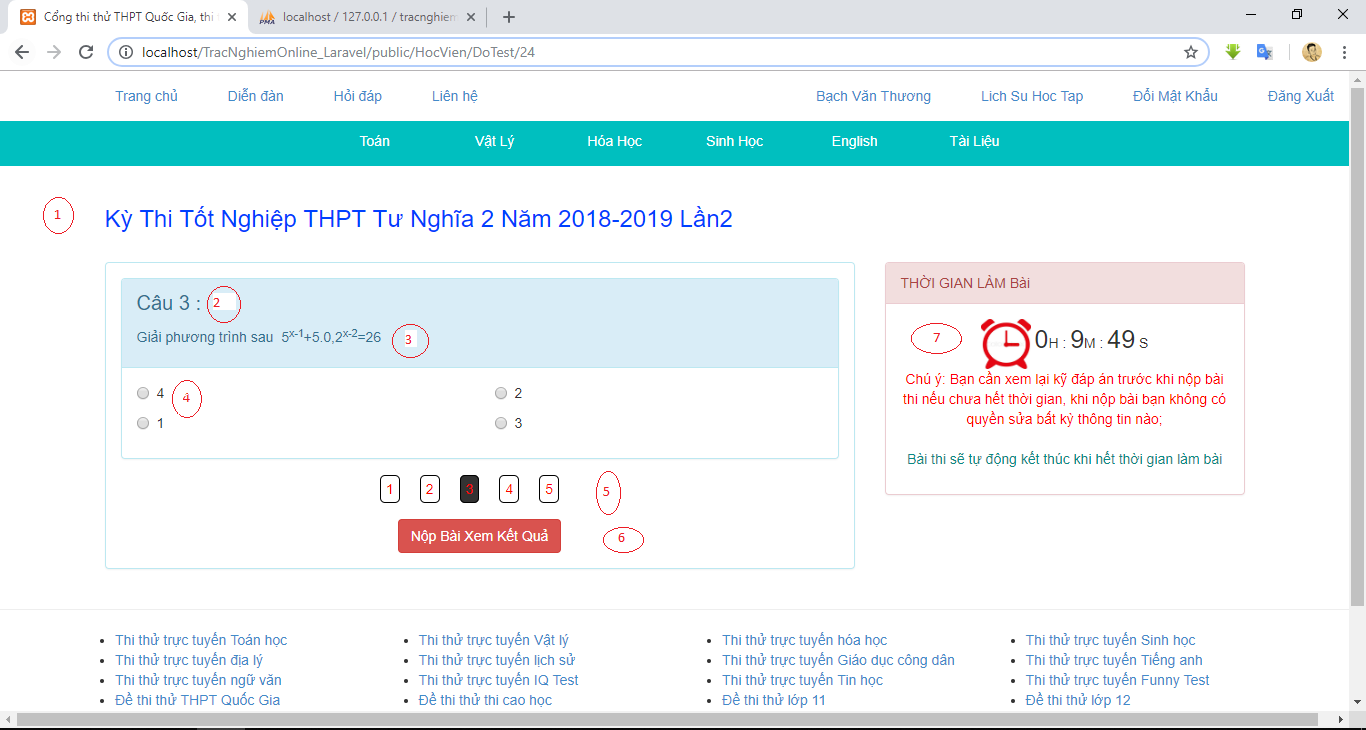
* Danh sách xử lý

Bảng 10.4.6. Xử lý giao diện trang danh sách đề thi theo môn học

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Input select chọn môn học để tìm kiếm |  |  |
| 2 | Button submit tìm kiếm |  |  |
| 3 | Box hiển thị thông tin đề thi |  |  |
| 4 | Button vào thi | Click chọn |  |

**10.4.7. Giao diện trang làm bài thi**

* Giao diên



Hình 10.4.7. Giao diện trang làm bài thi

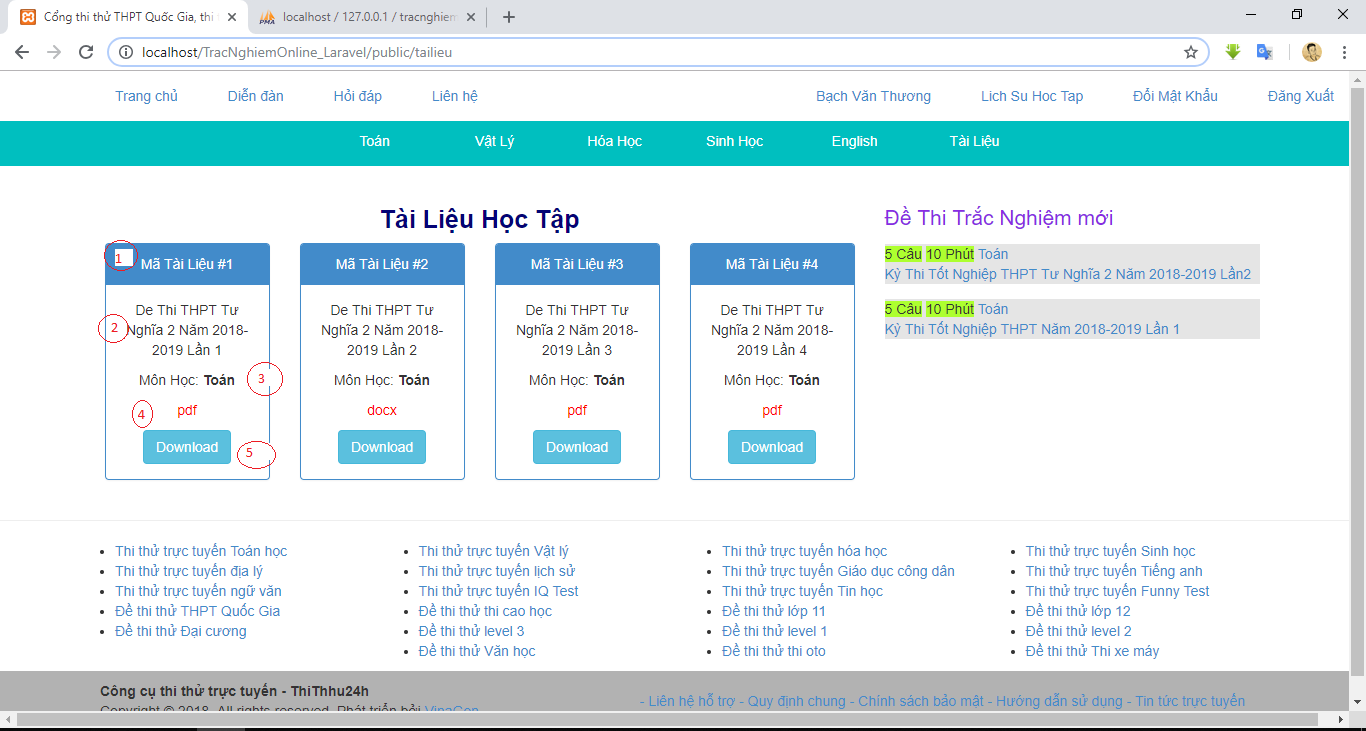
* Danh sách xử lý

Bảng 10.4.7. Xử lý giao diện trang danh sách đề thi theo môn học

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Label hiển thị tên đề thi |  |  |
| 2 | Label hiên thị thứ tự câu hỏi |  |  |
| 3 | Label thể hiên nội dung câu hỏi |  |  |
| 4 | Radiobox để lựa chọn phương án |  |  |
| 5 | Button phân trang câu hỏi |  |  |
| 6 | Button submit nộp bài |  |  |
| 7 | Hiển thị thời gian làm bài |  |  |

**10.4.8. Giao diện trang tài liệu**

* Giao diện



Hình 10.4.8. Giao diện trang tài liệu

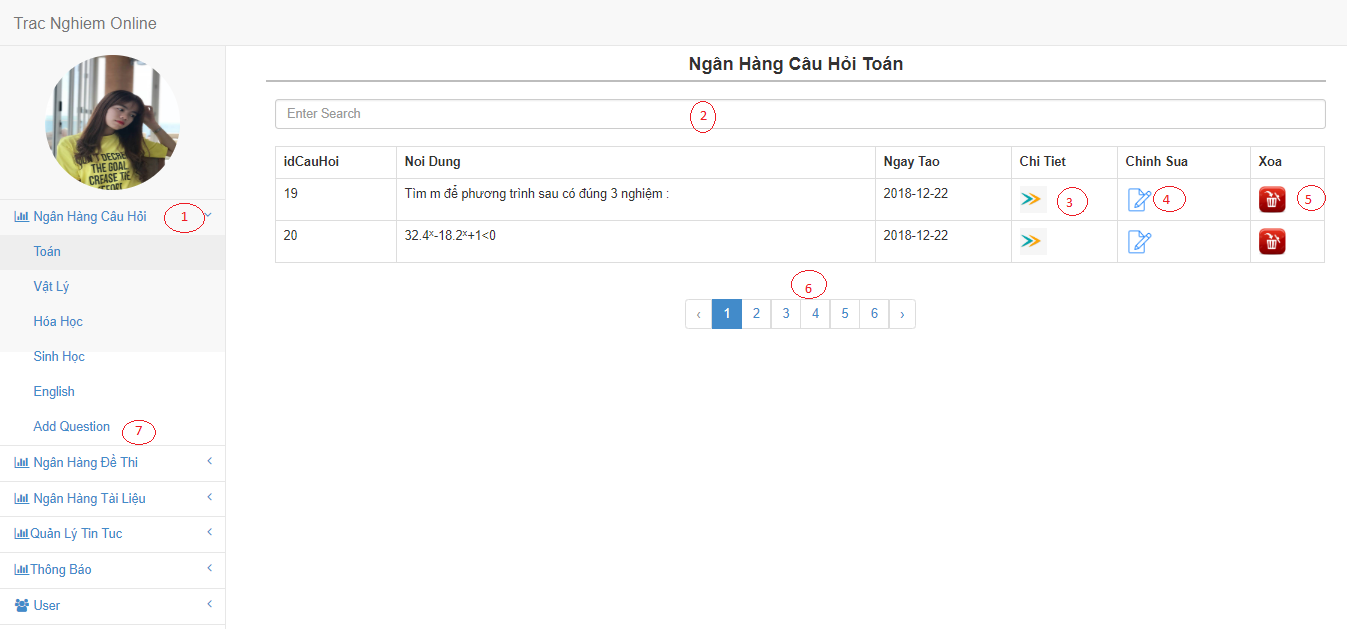
* Danh sách xử lý

Bảng 10.4.8. Xử lý giao diện trang tài liệu

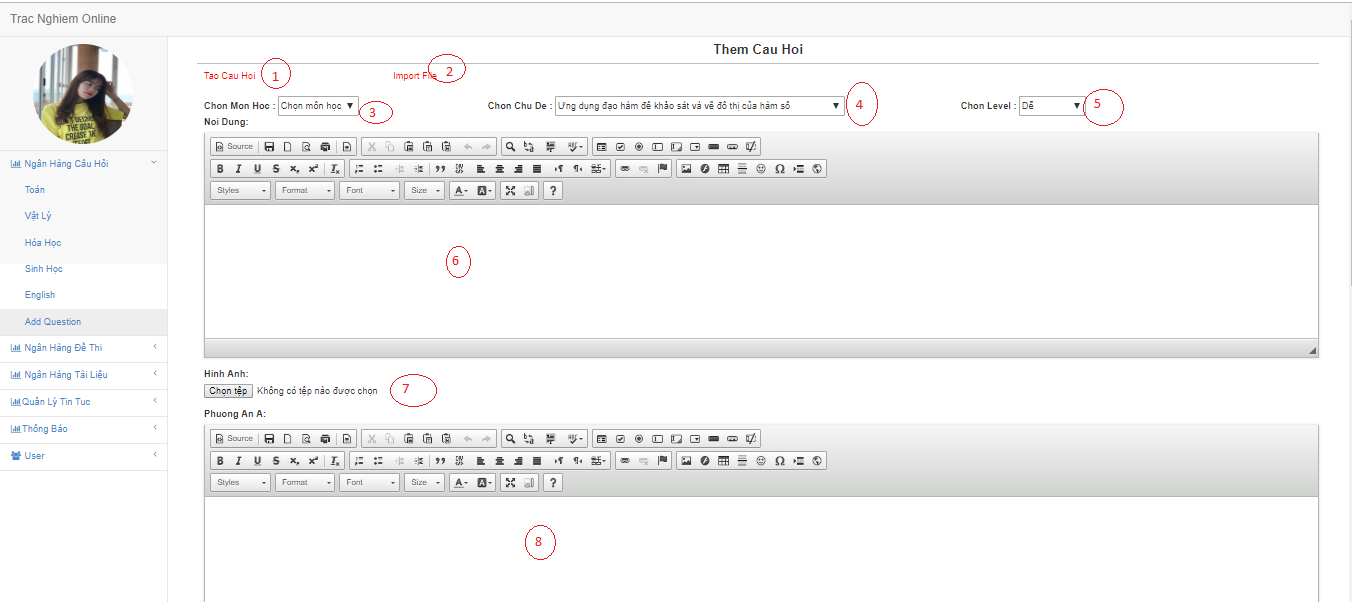
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Box thể hiện thông tin tài liệu |  |  |
| 2 | Label hiển thị tên tài liệu |  |  |
| 3 | Label thể hiện tên môn học |  |  |
| 4 | Label thể hiên loại tài liệu |  |  |
| 5 | Button download | Click chọn |  |

**10.4.9. Giao diện quản lý ngân hàng câu hỏi**

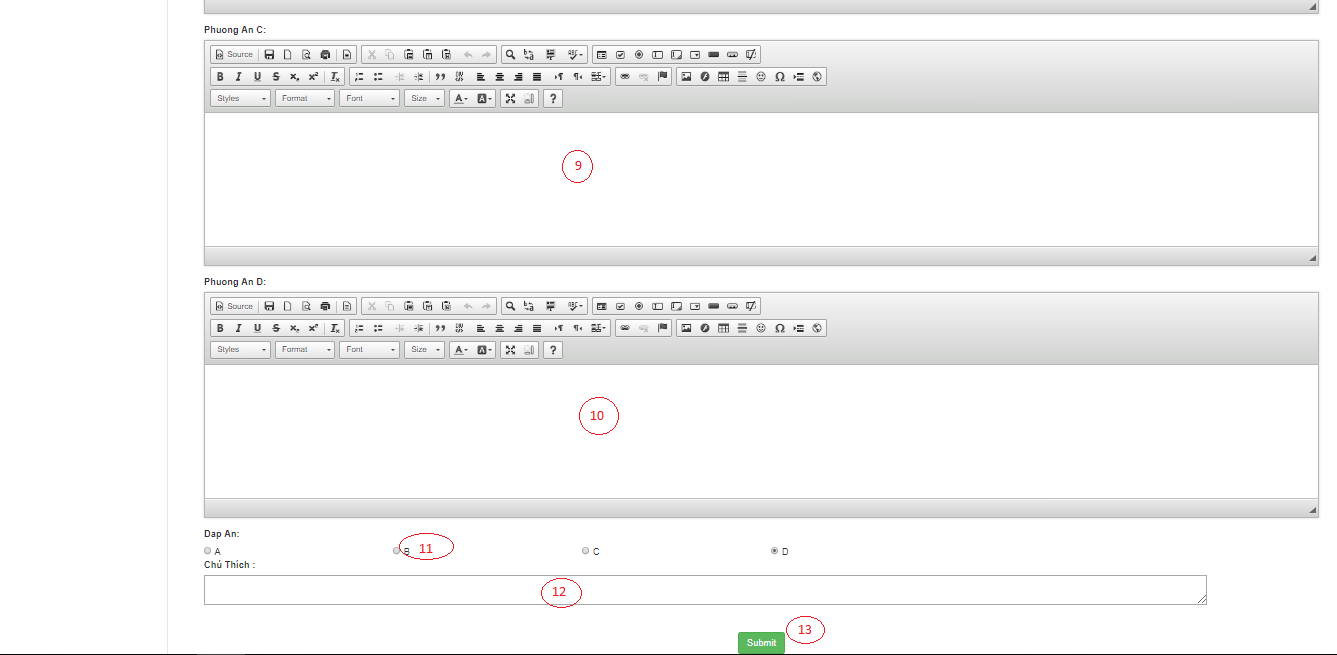
* Giao diện



Hình 10.4.9. Giao diện danh sách ngân hàng câu hỏi



Hình 10.4.9. Giao diện trang thêm câu hỏi 1



Hình 10.4.9. Giao diện trang thêm câu hỏi 2

* Danh sách xử lý

Bảng 10.4.9. Xử lý giao diện trang danh sách ngân hàng câu hỏi

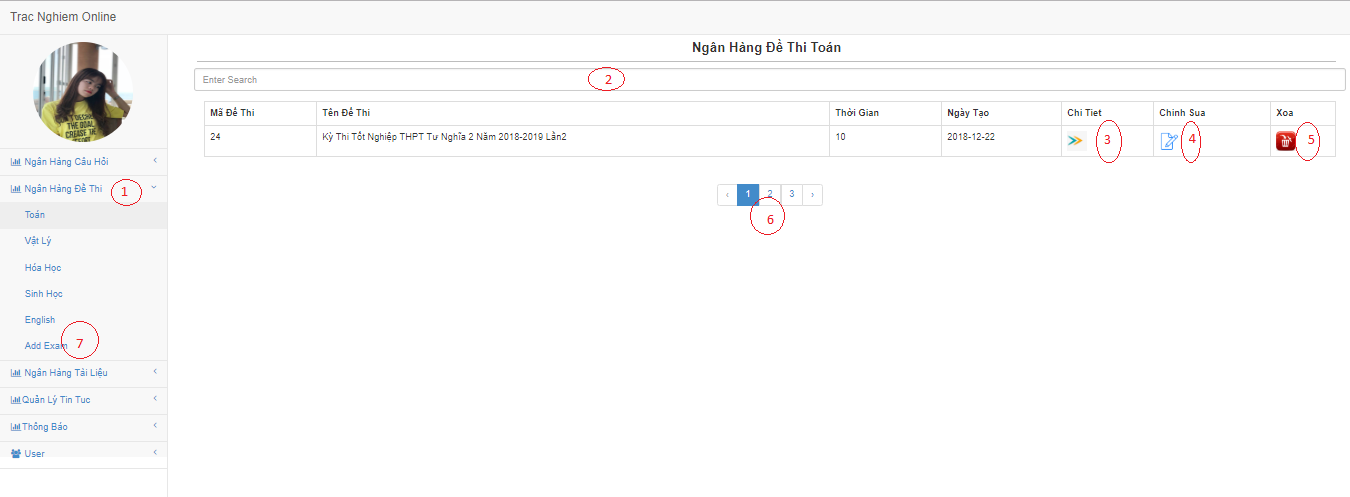
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quản lý ngân hàng câu hỏi | Click chọn | Gọi đến trang quản lý ngân hàng câu hỏi |
| 2 | Textbox search | Nhập id câu hỏi |  |
| 3 | Link xem chi tiết | Click chọn |  |
| 4 | Link chỉnh sửa | Click chọn |  |
| 5 | Link xóa | Click chọn |  |
| 6 | Button phân trang | Click chọn |  |
| 7 | Link thêm câu hỏi | Click chọn |  |

Bảng 10.4.9. Xử lý giao diện trang thêm câu hỏi 1 và 2

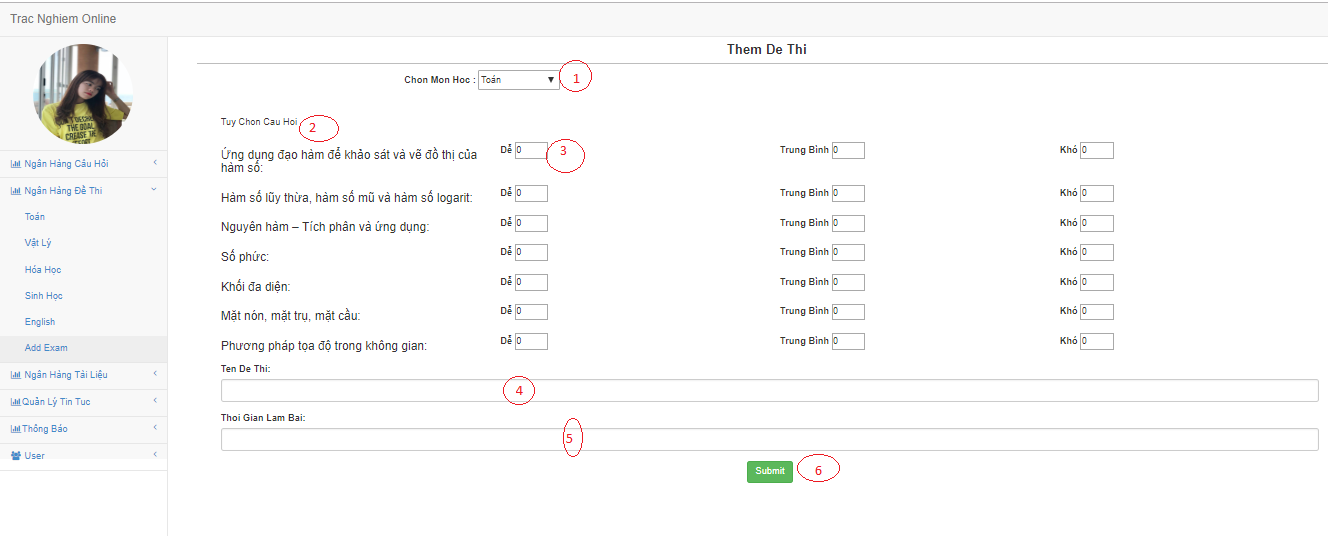
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Link tạo câu hỏi nhập tay |  |  |
| 2 | Link import câu hỏi |  |  |
| 3 | Input chọn môn học |  |  |
| 4 | Input chọn chủ đề môn học |  |  |
| 5 | Input chọn level |  |  |
| 6 | Textbox nhập nội dung câu hỏi |  |  |
| 7 | Input file chọn hình ảnh |  |  |
| 8 | Text nhập phương án A |  |  |
| 9 | Text nhập phương án B |  |  |
| 10 | Text nhập phương án C |  |  |
| 11 | Text nhập phương án D |  |  |
| 12 | Textbox nhập chú thích |  |  |
| 13 | Button submit |  |  |

**10.4.10. Giao diện trang quản lý ngân hàng đề thi**

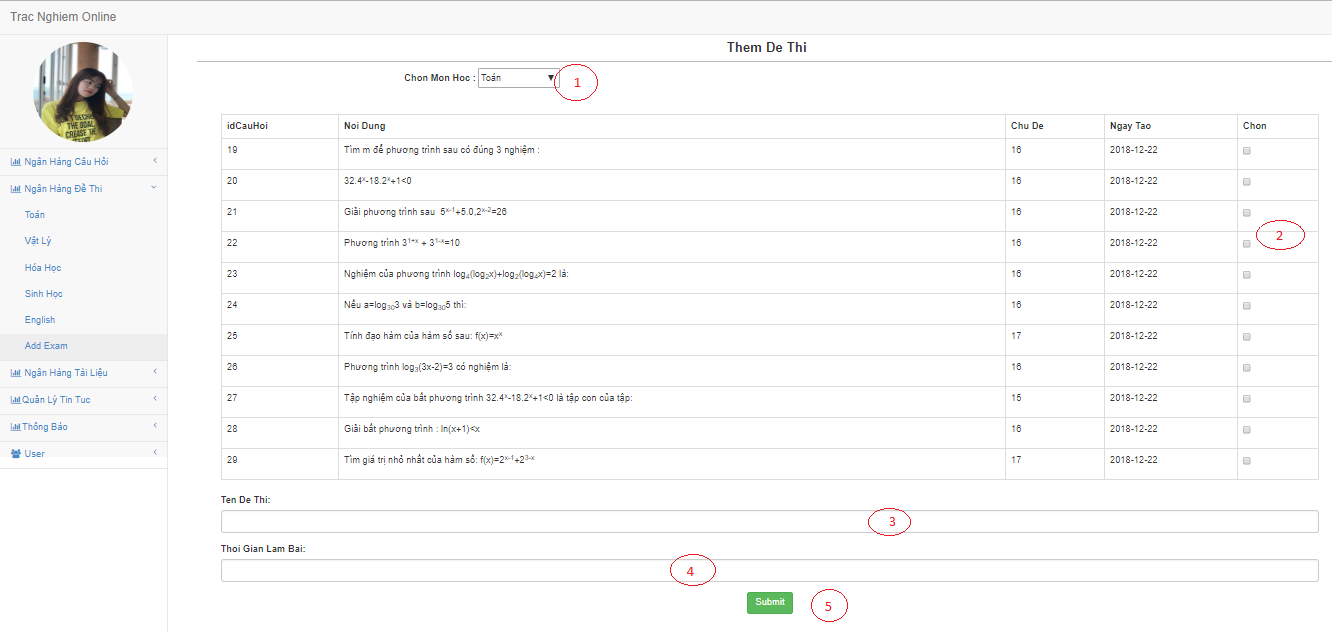
* Giao diện



Hình 10.4.10. Giao diện trang danh sánh đề thi



Hình 10.4.10. Giao diện thêm đề thi 1



Hình 10.4.10. Giao diện thêm đề thi 2

* Danh sách xử lý

Bảng 10.4.10. Xử lý giao diện trang danh sách ngân hàng đề thi

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quản lý ngân hàng đề thi | Click chọn | Gọi đến trang quản lý ngân hàng đề thi |
| 2 | Textbox search | Nhập id đề thi |  |
| 3 | Link xem chi tiết | Click chọn |  |
| 4 | Link chỉnh sửa | Click chọn |  |
| 5 | Link xóa | Click chọn |  |
| 6 | Button phân trang | Click chọn |  |
| 7 | Link thêm đề thi | Click chọn |  |

Bảng 10.4.10. Xử lý giao diện trang thêm đề thi 1

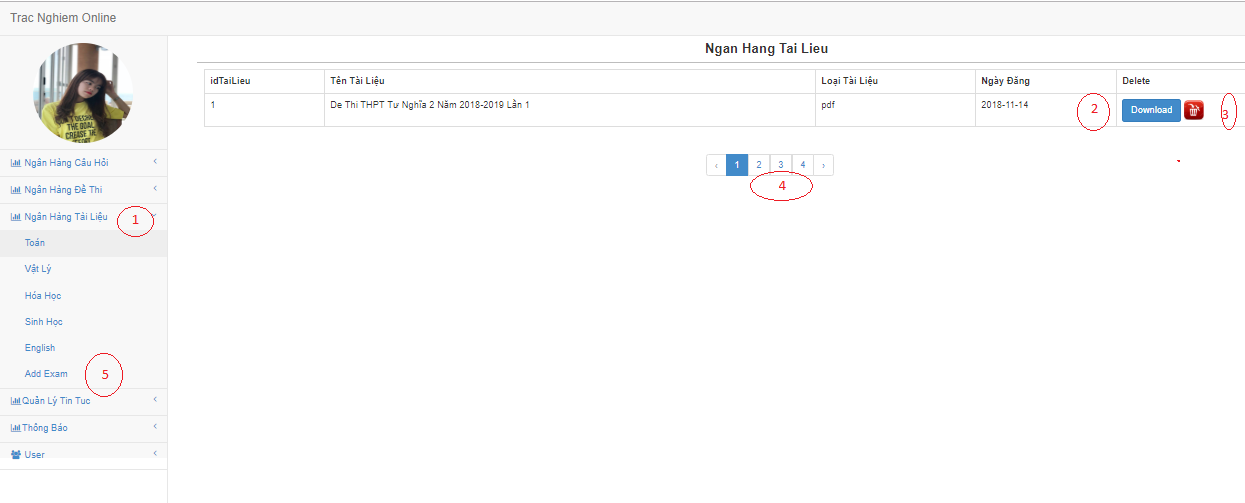
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Input select chọn môn học |  |  |
| 2 | Link tùy chọn câu hỏi |  |  |
| 3 | Input nhập số lượng câu hỏi mỗi chủ đề |  |  |
| 5 | Textbox nhập tên đề thi |  |  |
| 6 | Textbox nhập thời gian làm bài |  |  |
| 7 | Button Submit |  |  |

Bảng 10.4.10. Xử lý giao diện trang thêm đề thi 2

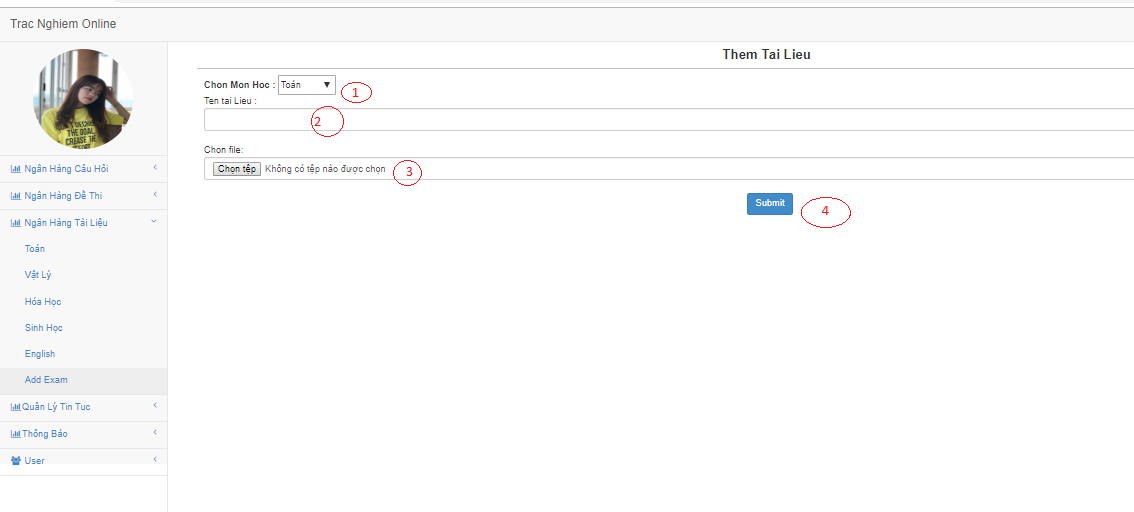
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Input select chọn môn học |  |  |
| 2 | Check box chọn câu hỏi |  |  |
| 3 | Textbox nhập tên đề thi |  |  |
| 4 | Textbox nhập thời gian làm bài |  |  |
| 5 | Button Submit | Các mục (1)(2) (3)(4) không được bỏ trống |  |

**10.4.11. Giao diện trang quản lý ngân hàng tài liệu**

* Giao diện



Hình 10.4.11. Giao diện trang ngân hàng tài liệu



Hình 10.4.11. Giao diện trang thêm tài liệu

* Danh sách xử lý

Bảng 10.4.11. Xử lý giao diện trang danh sách tài liệu

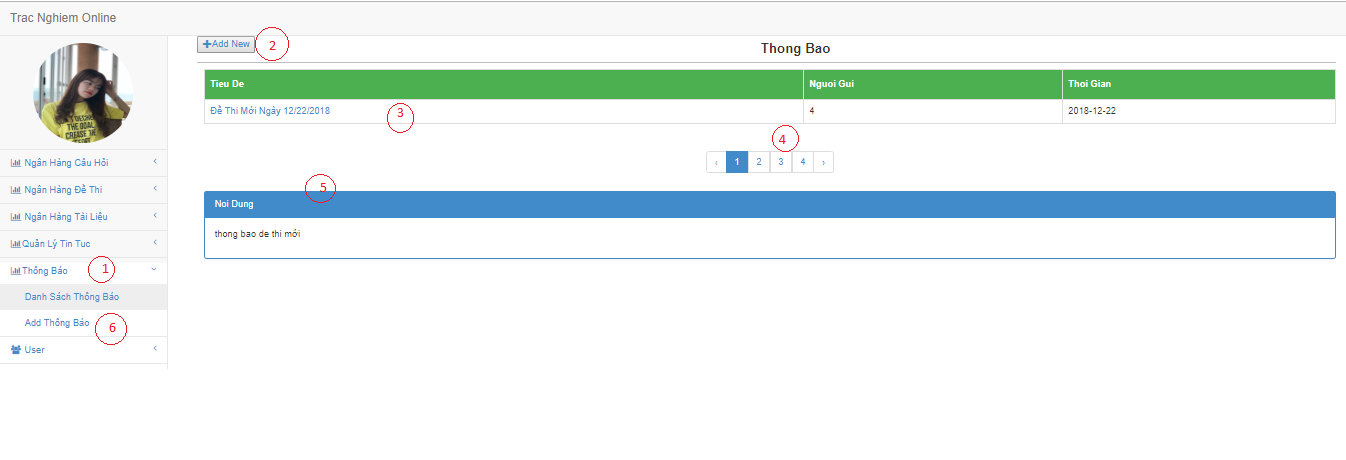
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quản lý ngân hàng tài liệu | Click chọn | Gọi đến trang quản lý ngân hàng tài liệu |
| 2 | Button download | Click chọn |  |
| 3 | Button xóa | Click chọn |  |
| 4 | Button phân trang | Click chọn |  |

Bảng 10.4.11. Xử lý giao diện trang thêm tài liệu

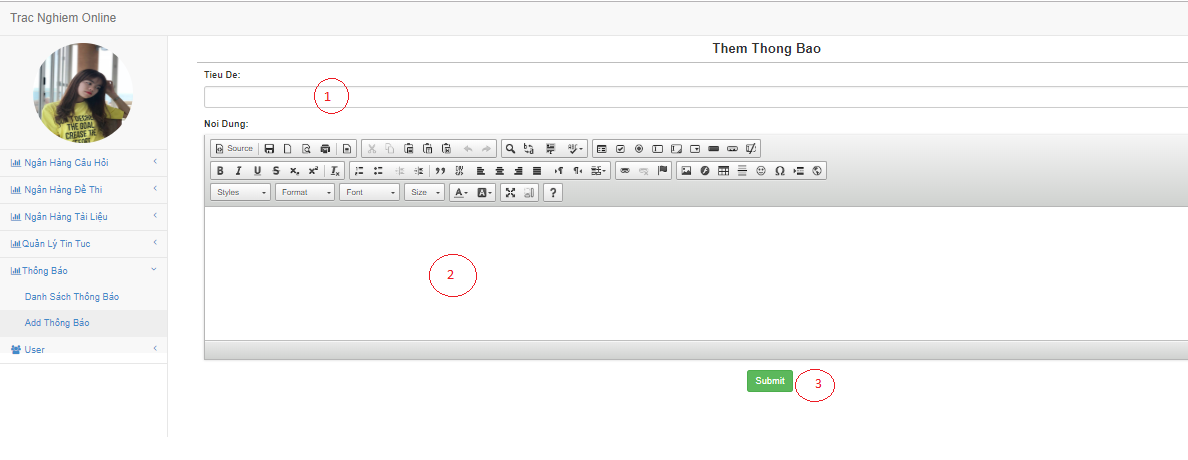
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Input select chọn môn học |  |  |
| 2 | Text box nhập tên tài liệu |  |  |
| 3 | Input file chọn file tài liệu |  |  |
| 4 | Button Submit | Các mục (1) (2) (3) không được bỏ trống |  |

**10.4.12. Giao diện trang thông báo**

* Giao diện



Hình 10.4.12. Giao diện trang danh sách thông báo



Hình 10.4.12. Giao diện trang thêm thông báo

* Danh sách xử lý

Bảng 10.4.12. Xử lý giao diện trang danh sách thông báo

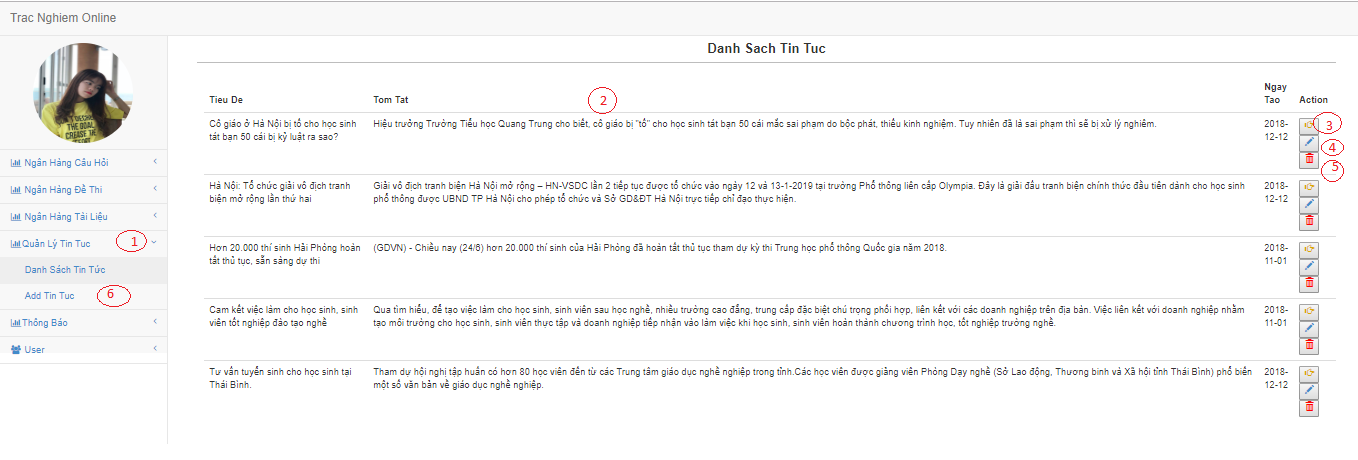
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Xem thông báo | Click chọn | Gọi đến trang danh sách thông báo |
| 2 | Button thêm thông báo mới | Click chọn |  |
| 3 | Table hiển thị tiêu đề các thông báo |  |  |
| 4 | Button phân trang | Click chọn |  |
| 5 | Div hiển thị nội dung thông báo |  |  |

Bảng 10.4.12. Xử lý giao diện trang thêm thông báo

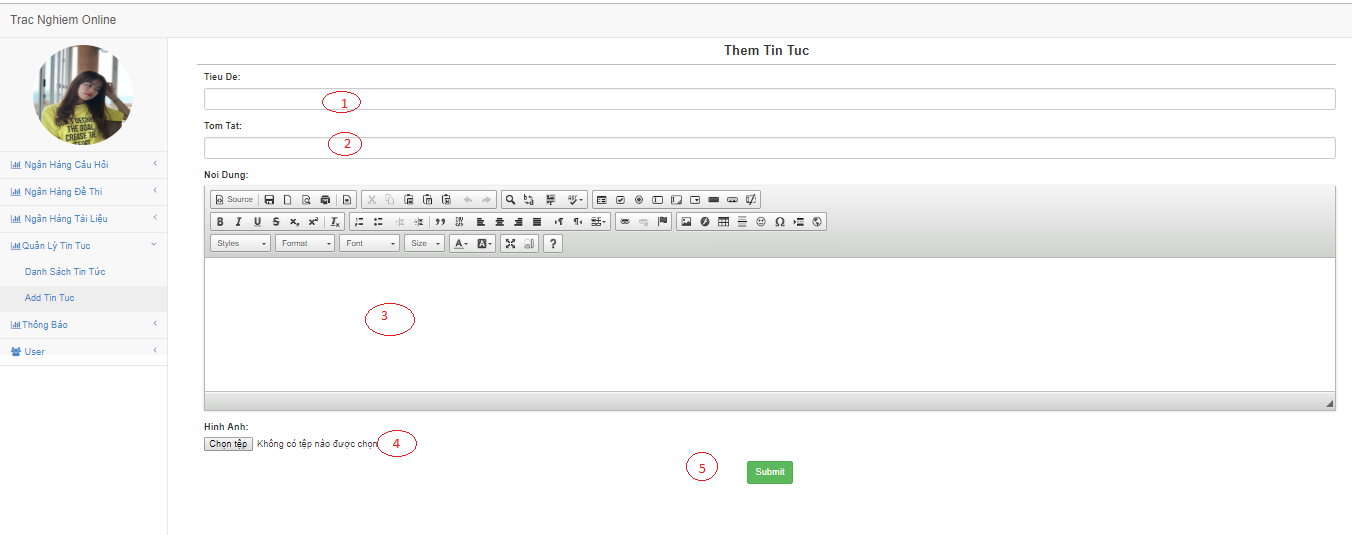
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox nhập tiêu đề |  |  |
| 2 | Text box nhập nội dung thông báo |  |  |
| 3 | Button Submit | Các mục (1) (2) không được bỏ trống |  |

**10.4.13. Giao diện trang quản lý tin tức**

* Giao diện



Hình 10.4.13. Giao diện trang danh sách tin tức



Hình 10.4.13. Giao diện trang thêm tin tức

* Danh sách xử lý

Bảng 10.4.13. Xử lý giao diện trang danh sách thông báo

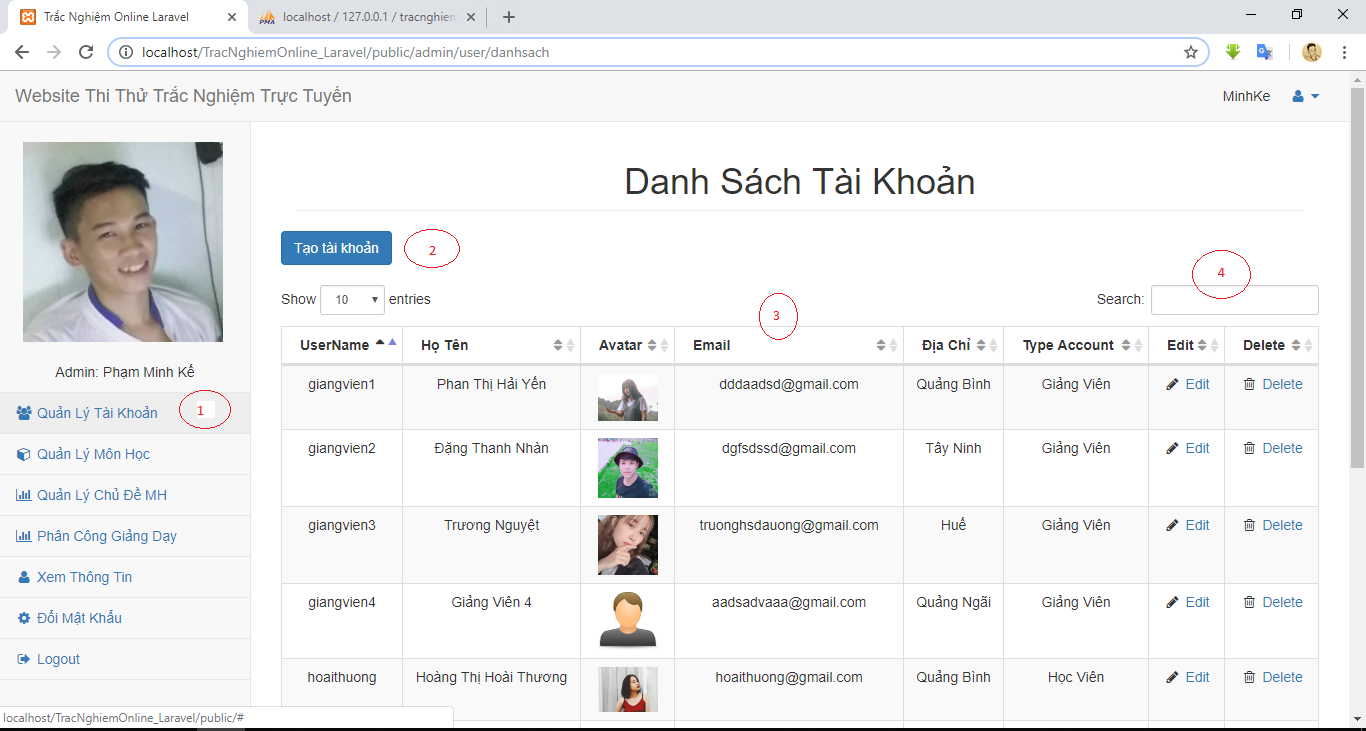
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Ngân hàng tin tức | Click chọn | Gọi đến trang danh sách tin tức |
| 2 | Table hiển thị tiêu đề tin tức | Click chọn |  |
| 3 | Button xem chi tiết tin tức | Click chọn |  |
| 4 | Button sửa tin tức | Click chọn |  |
| 5 | Button xóa tin tức | Click chọn |  |

Bảng 10.4.13. Xử lý giao diện trang thêm tin tức

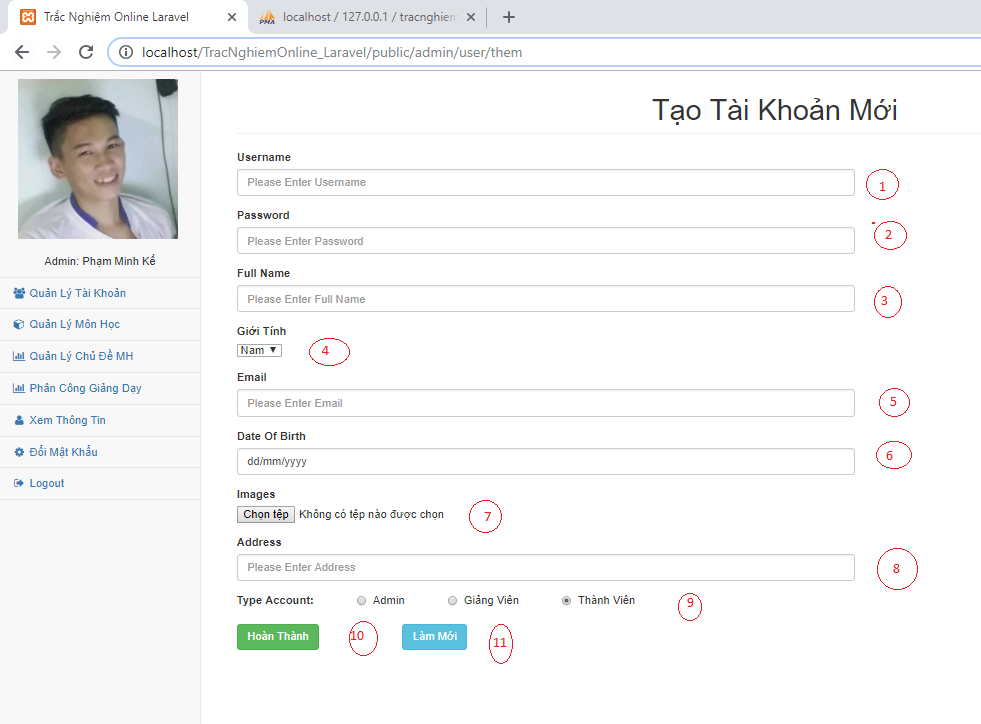
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox nhập tiêu đề |  |  |
| 2 | Textbox nhập tóm tắt |  |  |
| 3 | Text box nhập nội dung thông báo |  |  |
| 4 | Input file thêm hình ảnh |  |  |
| 4 | Button Submit | Các mục (1) (2) (3) không được để trống và nhập đúng độ dài |  |

**10.4.14. Giao diện quản lý User**

* Giao diện



Hình 10.4.14. Giao diện trang danh sách User



Hình 10.4.14. Giao diên trang thêm tài khoản

* Danh sách xử lý

Bảng 10.4.14. Xử lý giao diện trang danh sách user

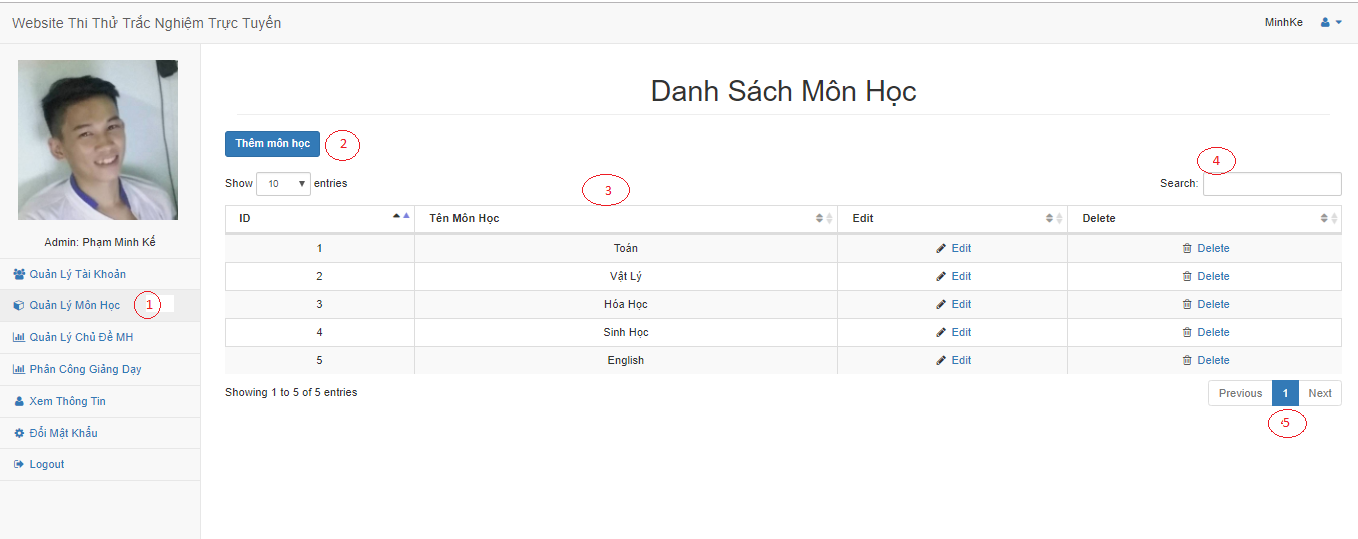
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quản lý tài khoản | Click chọn | Gọi đến trang quản lý tài khoản |
| 2 | Button thêm tài khoản mới | Click chọn |  |
| 3 | Table hiển thị danh sách tài khoản |  |  |
| 4 | Textbox search | Nhập username để tìm kiếm |  |

Bảng 10.4.14. Xử lý giao diện trang thêm tài khoản

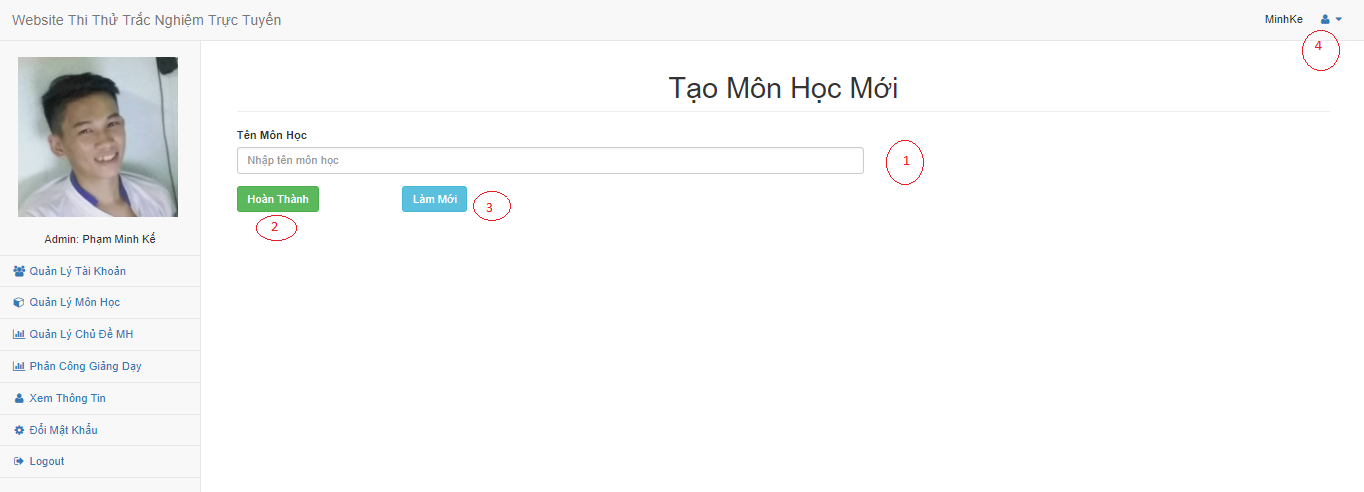
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox nhập username |  |  |
| 2 | Textbox nhập password |  |  |
| 3 | Textbox nhập họ tên |  |  |
| 4 | Comboboc chọn giới tính |  |  |
| 5 | Textbox nhập Email |  |  |
| 6 | Textbox nhập ngày sinh |  |  |
| 7 | Input file chọn ảnh đại diện |  |  |
| 8 | Textbox nhập địa chỉ |  |  |
| 9 | Radiobox phân quyền |  |  |
| 10 | Button submit | Các mục (1) (2) (3) (4) (5) (6) (7) (8) (9) không được bỏ trống và nhập đúng kiểu dữ liệu |  |
| 11 | Button làm mới |  | Clear các textbox |

**10.4.15. Giao diện trang quản lý môn học**

* Giao diện



Hình 10.4.15. Giao diện trang danh sách môn học



Hình 10.4.15. Giao diện trang thêm môn học

* Danh sách xử lý

Bảng 10.4.15. Xử lý giao diện trang danh sách môn học

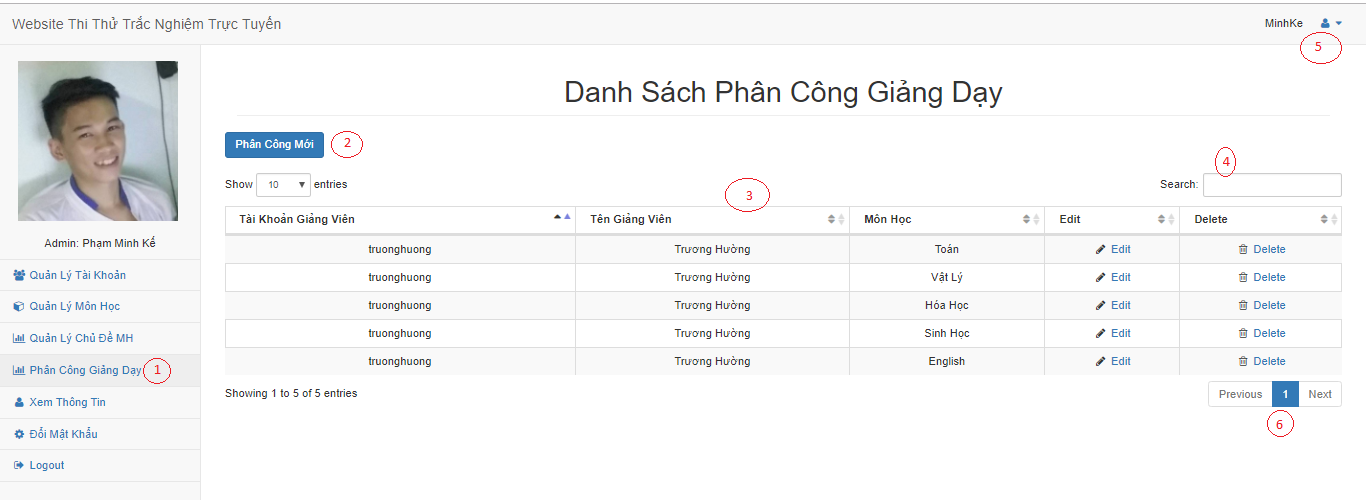
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quản lý mô học | Click chọn | Gọi đến trang quản lý môn học |
| 2 | Button thêm môn học mới | Click chọn |  |
| 3 | Table hiển thị danh sách môn học |  |  |
| 4 | Textbox search | Nhập tên môn học để tìm kiếm |  |
| 5 | Button phân trang |  |  |

Bảng 10.4.15. Xử lý giao diện trang thêm môn học

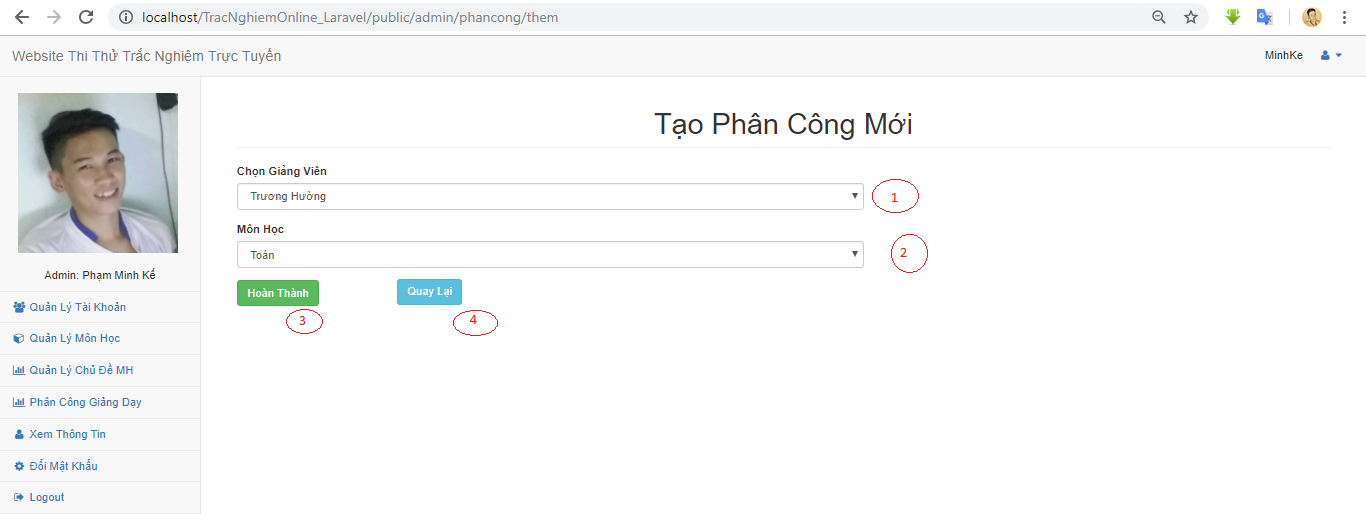
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox nhập tên môn học |  |  |
| 2 | Button submit hoàn thành | Mục (1) không được bỏ trống và nhập đúng kiểu dữ liệu |  |
| 3 | Button làm mới |  | Clear các textbox |
| 4 | Button logout | Click chọn |  |

**10.4.16. Giao diện trang phân công giảng dạy**

* Giao diện



Hình 10.4.16. Giao diện trang danh sách phân công giảng dạy



Hình 10.4.16. Giao diện trang thêm phân công mới

* Danh sách xử lý

Bảng 10.4.16. Xử lý giao diện trang danh sách phân công giảng dạy

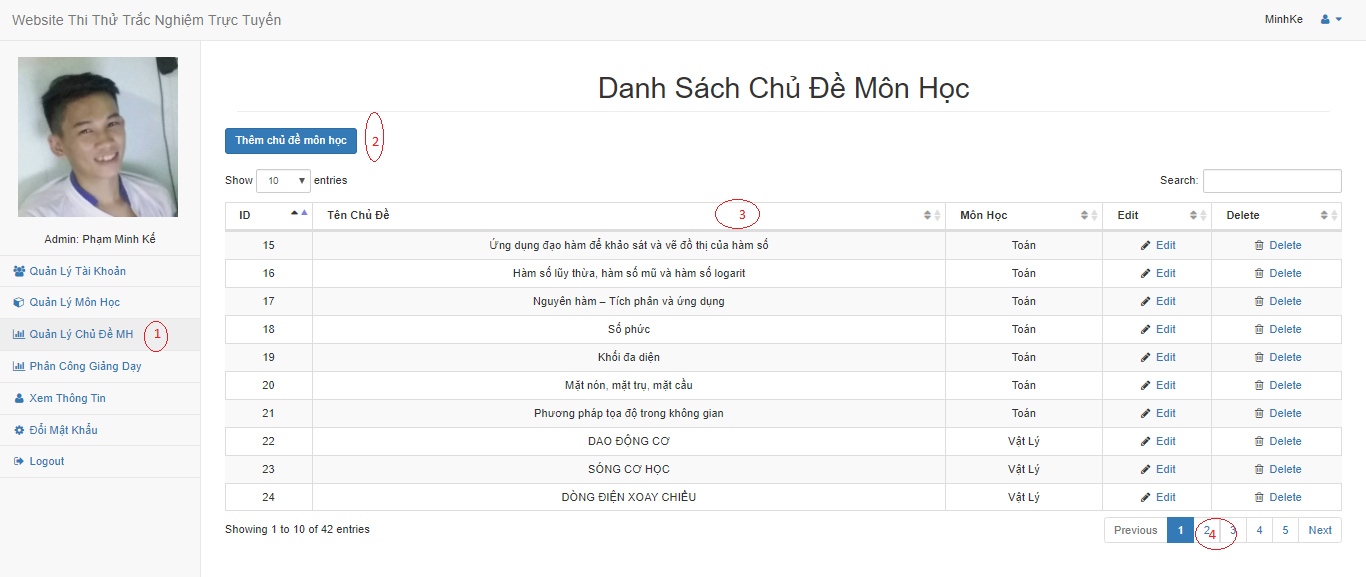
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Phân công giảng dạy | Click chọn | Gọi đến trang phân công giảng dạy |
| 2 | Button thêm phân công mới | Click chọn |  |
| 3 | Table hiển thị danh sách phân công |  |  |
| 4 | Textbox search | Nhập id Giảng viên để tìm kiếm |  |
| 5 | Button phân trang |  |  |

Bảng 10.4.16. Xử lý giao diện trang thêm phân công mới

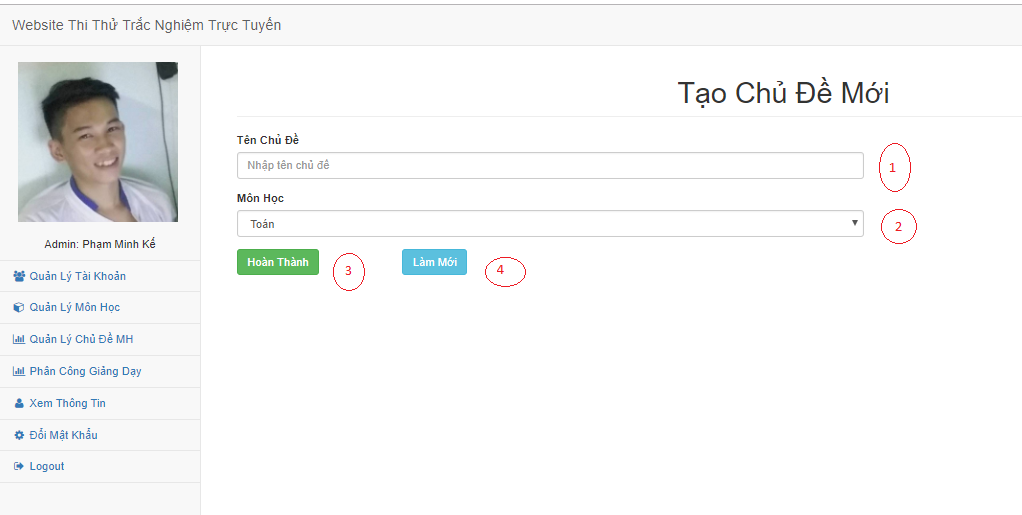
|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Input select chọn giáo viên | Click chọn |  |
| 2 | Input select chọn môn học | Click chọn |  |
| 3 | Button submit |  |  |
| 4 | Button quay lại |  |  |

**10.4.17. Giao diện trang quản lý chủ đề môn học**

* Giao diện



Hình 10.4.17. Giao diện danh sách chủ đề



Hình 10.4.17. Giao diện trang thêm chủ đề mới

* Danh sách xử lý

Bảng 10.4.17. Xử lý giao diện trang danh sách chủ đề

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Quản lý chủ đề | Click chọn | Gọi đến trang quản lý chủ đề |
| 2 | Button thêm chủ đề mới | Click chọn |  |
| 3 | Table hiển thị danh sách chủ đề các môn học |  |  |
| 4 | Textbox search | Nhập tên môn học để tìm kiếm |  |
| 5 | Button phân trang |  |  |

Bảng 10.4.17. Xử lý giao diện trang thêm chủ đề mới

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên Xử Lý** | **Điều kiện gọi thực hiện** | **Ghi chú** |
| 1 | Textbox tên chủ đề |  |  |
| 2 | Input select chọn môn học | Click chọn |  |
| 3 | Button submit |  |  |
| 4 | Button quay lại |  |  |

**10.5. Kiểm thử phần mềm**

Bảng 10.5. Bảng danh sách kiểm thử

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case description** | **Test steps** | **Expected Result** | **Status** |
| Kiểm tra giao diện hiển thị trang chủ | 1. Truy cập vào hệ thống.  2. Kiểm tra giao diện hiển thị. | 1. Hệ thống tồn tại.  2. Giao diện có hiển thị menu môn học  3. Hiển thị đề thi theo môn học  4.Hiển thị tin tức,thông báo  5.Top điểm cao, đề thi mới | PASS |
| Kiểm tra khi Guest đăng ký | 1. Truy cập trang đăng ký  2. Đăng ký | 1. Textbox nhập thông tin nhập được  2. Bắt lỗi khi nhập sai hoặc bỏ trống  3.Sau khi đăng ký đăng nhập hệ thống thành công | PASS |
| Kiểm tra đăng nhập | 1.truy cập trang đăng nhập  2.Đăng nhập | 1.Bắt lỗi khi đăng nhập sai  2.Đăng nhập thành công với tài khoản đúng | PASS |
| Kiểm tra đổi mật khẩu | 1.Đăng nhập vào hệ thống  2.Truy cập trang đổi mật khẩu  3.Kiểm tra đổi mật khẩu bằng các cách nhập sai | 1.Bắt lỗi khi nhập thiếu dữ liệu  2. Bắt lỗi khi nhập sai mật khẩu cũ  3. Bắt lỗi khi nhập mật khẩu mới không trùng nhau  4. Đổi mật khẩu khi nhập đúng | PASS |
| Kiểm tra tìm kiếm đề thi theo môn học | 1. Kiểm tra vào môn học trên menu  2. Hiển thị dánh sách các đề thi theo môn học | Hiển thị dánh sách các đề thi theo môn học | PASS |
| Kiểm tra làm bài thi | 1.Click vào bắt đầu làm bài thi  2. Thực hiện bài thi  3. Kiểm tra phân trang các câu hỏi  4.Có tự động nộp bài khi hết thời gian | 1.Bài thi có hiển thị  2. Các câu hỏi được phân trang  3.Tự động nộp bài khi hết thời gian | PASS |
| Kiểm tra show đáp án | 1.Kiểm tra có show đáp án sau khi nộp bài  2.Kiểm tra có hiển thị câu đúng câu sai  3. Kiểm tra có hiển thị điểm | 1. Có show đáp án chú thích  2.Có show đáp án và phương án  3.Tính đúng điểm | PASS |
| Kiểm tra quản lý tài khoản | 1.Đăng nhập vào tài khoản admin  2.Kiểm tra có hiển thị danh sách user  3.Kiểm tra hiển thị thông tin user khi chọn xem chi tiết  4.Kiểm tra hiện thị form thêm ,chỉnh sửa user chính xác | 1.Có hiển thị danh sách User  2.Hiển thị chi tiết thông tin user  3.Bắt lỗi khi thêm hoặc chỉnh sửa user thiếu thông tin hoặc nhập sai dữ liệu  4.Xóa đúng user | PASS |
| Kiểm tra quản lý môn học | 1.Đăng nhập vào tài khoản admin  2.Kiểm tra có hiển thị danh sách môn học  3.Kiểm tra hiển thị thông tin môn học khi chọn xem chi tiết  4.Kiểm tra hiện thị form thêm ,chỉnh sửa môn học chính xác | 1.Có hiển thị danh sách môn học  2.Hiển thị chi tiết thông tin môn học  3.Bắt lỗi khi thêm hoặc chỉnh sửa môn học thiếu thông tin hoặc nhập sai dữ liệu  4.Xóa đúng user | PASS |
| Kiểm tra phân công giảng dạy | 1.Đăng nhập vào tài khoản admin  2.Kiểm tra có hiển thị danh sách phân công giảng dạy  3.Kiểm tra hiển thị thông tin phân công giảng dạy khi chọn xem chi tiết  4.Kiểm tra hiện thị form thêm ,chỉnh sửa phân công chính xác | 1.Có hiển thị danh sách phân công giảng dạy  2.Hiển thị chi tiết thông tin giảng dạy  3.Bắt lỗi khi thêm hoặc chỉnh sửa môn học trùng  4.Xóa phân công giảng dạy thành công | PASS |
| Kiểm tra quản lý ngân hàng câu hỏi | 1.Đăng nhập vào tài khoản Giảng Viên  2.Kiểm tra có hiển thị danh sách câu hỏi theo môn học  3.Kiểm tra hiển thị thông tin câu hỏi khi chọn xem chi tiết  4.Kiểm tra hiện thị form thêm ,chỉnh sửa câu hỏi chính xác | 1.Có hiển thị danh sách câu hỏi theo từng môn học  2.Bắt lỗi khi thêm ,chỉnh sửa câu hỏi  3. Bắt lỗi khi import file không đúng  4.Thêm ,chỉnh sửa, xóa câu hỏi được chính xác | PASS |
| Kiểm tra quản lý ngân hàng đề thi | 1.Đăng nhập vào tài khoản Giảng Viên  2.Kiểm tra có hiển thị danh sách đề thi theo môn học  3.Kiểm tra hiển thị chi tiết đề thi khi chọn xem chi tiết  4.Kiểm tra hiện thị form thêm ,chỉnh sửa đề thi chính xác | 1.Có hiển thị danh sách đề thi theo từng môn học  2.Bắt lỗi khi thêm ,chỉnh sửa đề thi khi nhập thiếu hay sai dữ liệu,độ dài  3. Có thể tạo đề thi theo hai cách tự động hay tùy chọn  4.Thêm ,chỉnh sửa, xóa đề thi được chính xác | PASS |
| Kiểm tra ngân hàng tài liệu | 1.Đăng nhập vào tài khoản Giảng Viên  2.Kiểm tra có hiển thị danh sách tài liệu theo môn học  3Kiểm tra hiện thị form thêm ,tài liệu | 1.Hiển thị danh sách tài liệu theo môn học  2.Có thể download tài liệu  3.Có thể xóa tài liệu  4. Băt lỗi khi thêm tài liệu như nhập thiếu sai thông tin, file không đúng quy định | PASS |
| Kiểm tra xem đăng thông báo | 1.Đăng nhập vào tài khoản Giảng Viên  2.Kiểm tra có hiển thị danh sách thông báo  3Kiểm tra hiện thị form thêm thông báo | 1.Hiển thị danh sách thông báo  2.Xem thông báo được  3.Bắt lỗi khi nhập sai dự liệu,độ dài khi thêm thông báo | PASS |

**Phần 4: KẾT LUẬN**

**4.1. Những kết quả đạt được**

* Có một kiến thức mới về PHP Framework.
* Hiểu cách thao tác với cơ sở dữ liệu theo phương pháp lập trình hướng đối tượng mà Laravel Framework cung cấp.
* Hiểu cách thức hoạt động của một trang web động.
* Hơn nữa việc nghiên cứu tìm hiểu một công nghệ mới cũng được củng cố, cải thiện kỹ năng thu thập nhu cầu của người dùng, phân tích chức năng nghiệp vụ, cách xây dựng ứng dụng từ đầu đến cuối.

**4.2.** **Những khó khăn khi thực hiện đề tài**

* Thiếu kinh nghiệm cho project 🡪 project còn sơ sài, giao diện chưa đẹp mắt.
* Trình độ Tiếng Anh còn thấp 🡪 việc đọc tài liệu Tiếng Anh bị hạn chế.

**4.3.** **Ưu điểm**

* Là website nên không cần phải cài đặt tốn bộ nhớ.
* Website giúp cho học sinh, sinh viên rèn
* Giao diện đơn giản giúp người dùng dễ dàng sử dụng trong những lần đầu tiên.
* Hạn chế việc lưu trữ trên giấy tờ, việc lưu trữ thông tin trở nên gọn nhẹ hơn.

**4.4. Hạn chế**

Dù nhóm đã cố gắng tìm hiểu và phát triển website một cách tốt nhất, nhưng do giới hạn về thời gian cũng như khả năng bản thân còn hạn chế, nên không tránh khỏi thiếu sót, khuyết điểm:

* Giao diện chưa được đẹp mắt, còn sơ sài.
* Thuật toán chưa được tối ưu hóa.
* Ở những nơi không có Internet thì không thể truy cập.
* Chưa làm chức năng khôi phục mật khẩu khi quên.

**4.5.** **Hướng phát triển trong tương lai**

Mặc dù còn tồn tại những hạn chế bên trên, nhưng với sự quyết tâm của nhóm, website sẽ được khắc phục và tiếp tục nâng cấp, cải tiến nhằm tiến tới sự hoàn hảo hơn và được sử dụng rộng rãi trong môi trường giảng đường. Giảm bớt sự lằng nhằng trong việc đăng ký và mất mát thông tin.

**TÀI LIỆU THAM KHẢO**

[1] <https://laravel.com/>

[2] <https://www.tutorialspoit.com/laravel/>

[3] [http://webfaver.com/php-coding/laravel-5/tut-laravel-5-0-laravel-la-gi-gioi-thieu-laravel- framework.html](http://webfaver.com/php-coding/laravel-5/tut-laravel-5-0-laravel-la-gi-gioi-thieu-laravel-%20%20%20framework.html)

[4] <https://cmay.vn/laravel-la-gi-uu-diem-cua-laravel-framework/>

[5] <https://toidicode.com/query-builder-trong-laravel-11.html>

[6] <https://www.codehub.vn/Xac-Thuc-Nguoi-Dung-Trong-Laravel>

[7] <https://toidicode.com/controller-trong-laravel-10.html>

[8] <https://toidicode.com/model-trong-laravel-12.html>

[9] <https://vietpro.net.vn/lap-trinh-laravel>