# Exercises

## Bài tập 1: Thực hiện lại toàn bộ các chức năng đã làm ở trên lớp nếu chưa hoàn thành

## Bài tập 2: Sử dụng Laravel cơ bản để thực hiện bài toán sau đây

Tạo 1 trang CRUD cho chức năng “Chức vụ” – Roles

Bảng **roles** có các trường như sau

* id
* name
* created\_at
* updated\_at

Vào CSDL, thêm mới 1 record với **name = ‘admin’**. Đồng thời đặt giá trị cho trường **role** trong bảng **users** = NULL (để tiến hành gán sau)

Sửa migration của bảng users để tương thích với bảng **roles** như sau

* Đổi tên trường từ **role** -> **role\_id** (chú ý thay đổi kiểu dữ liệu từ ‘string’ -> bigInteger)

Xem thêm ở đây: <https://laravel.com/docs/8.x/migrations#renaming-and-dropping-tables>

Đồng thời đặt giá trị của trường **role\_id** trong bảng **users** có id là 1 với **role\_id = 1** (role admin)

**role\_id** là id của bảng **roles**. Cho rằng một người dùng sẽ chỉ có 1 role duy nhất.

Giao diện tự thiết kế (Có thể sử dụng Modal bootstrap hoặc tách riêng thành 1 form). Ví dụ giao diện mẫu như sau

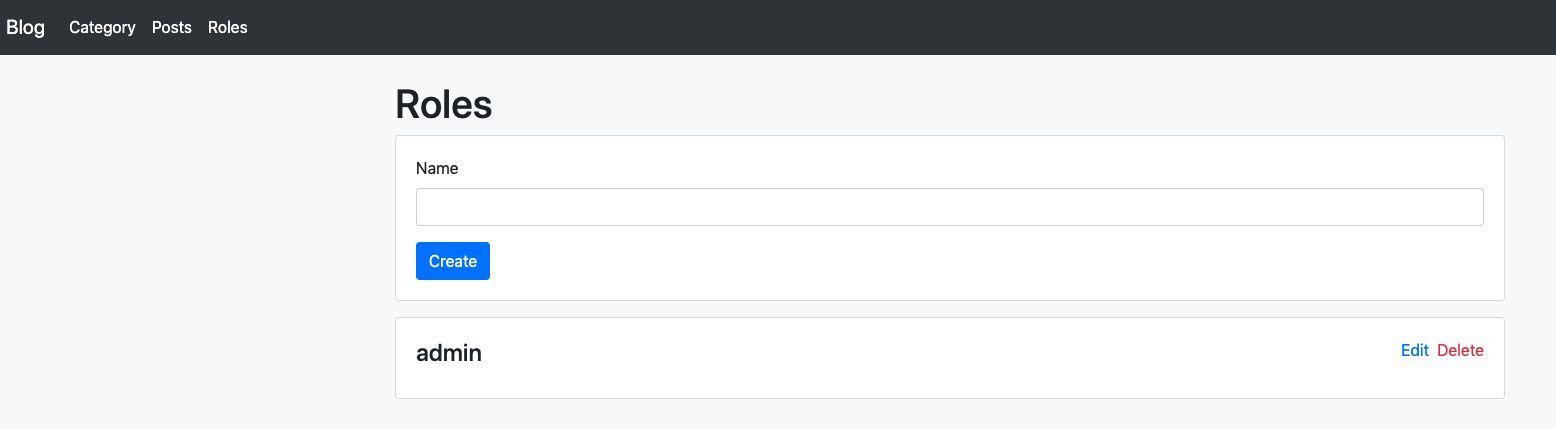


Figure : Giao diện mẫu của Roles

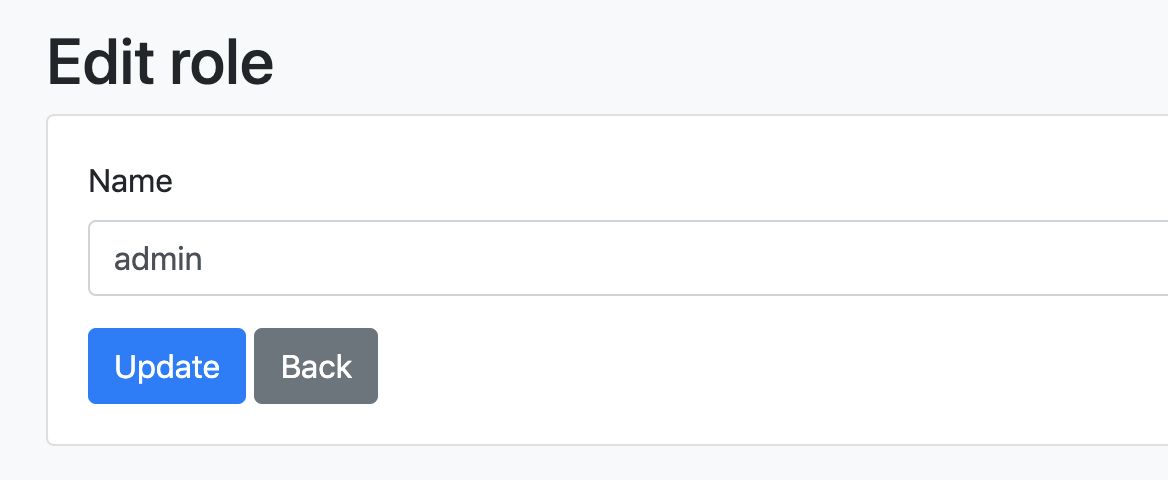


Figure : Giao diện cập nhật chức vụ

Đồng thời thêm 1 link trên navbar để hiển thị danh sách **roles** có trong hệ thống như hình trên

## Bài tập 3: Sử dụng Laravel cơ bản để thực hiện bài toán sau đây

Thực hiện chức năng gán quyền cho người dùng đã đăng kí trong hệ thống. Giao diện mẫu có thể xem ở đây

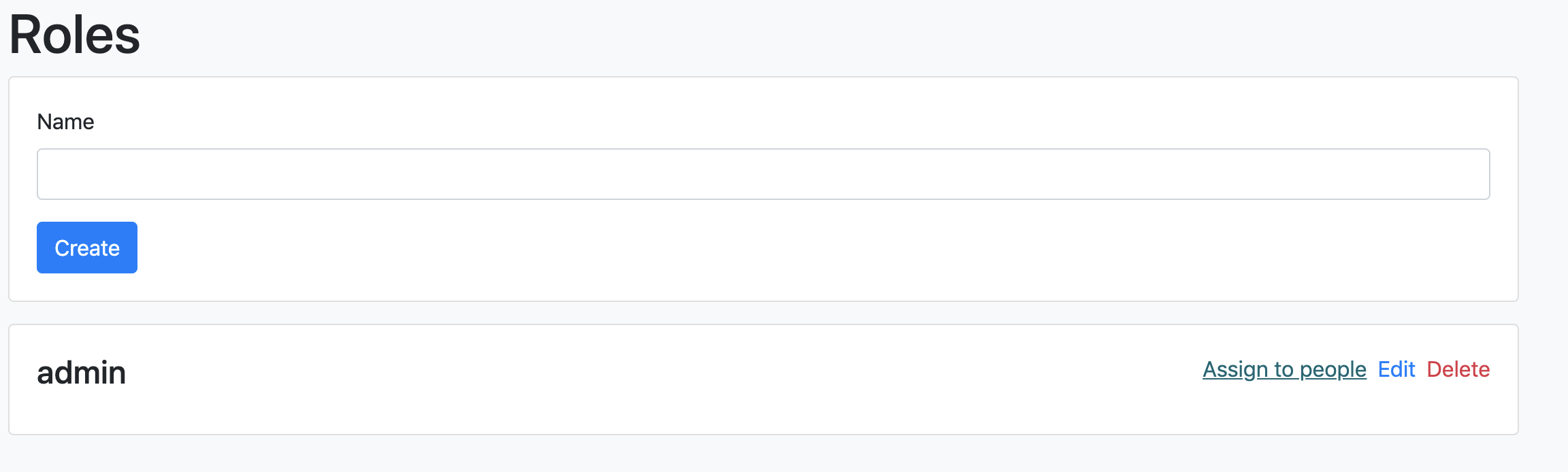


Figure : Gán quyền cho người dùng

Chỉnh sửa middleware **CheckRole** cho phép chỉ có người dùng có chức vụ là “admin”, ứng với role\_id = 1 mới được phép **thêm mới**, **sửa**, **xoá** và **xem** **Category**, **Roles**

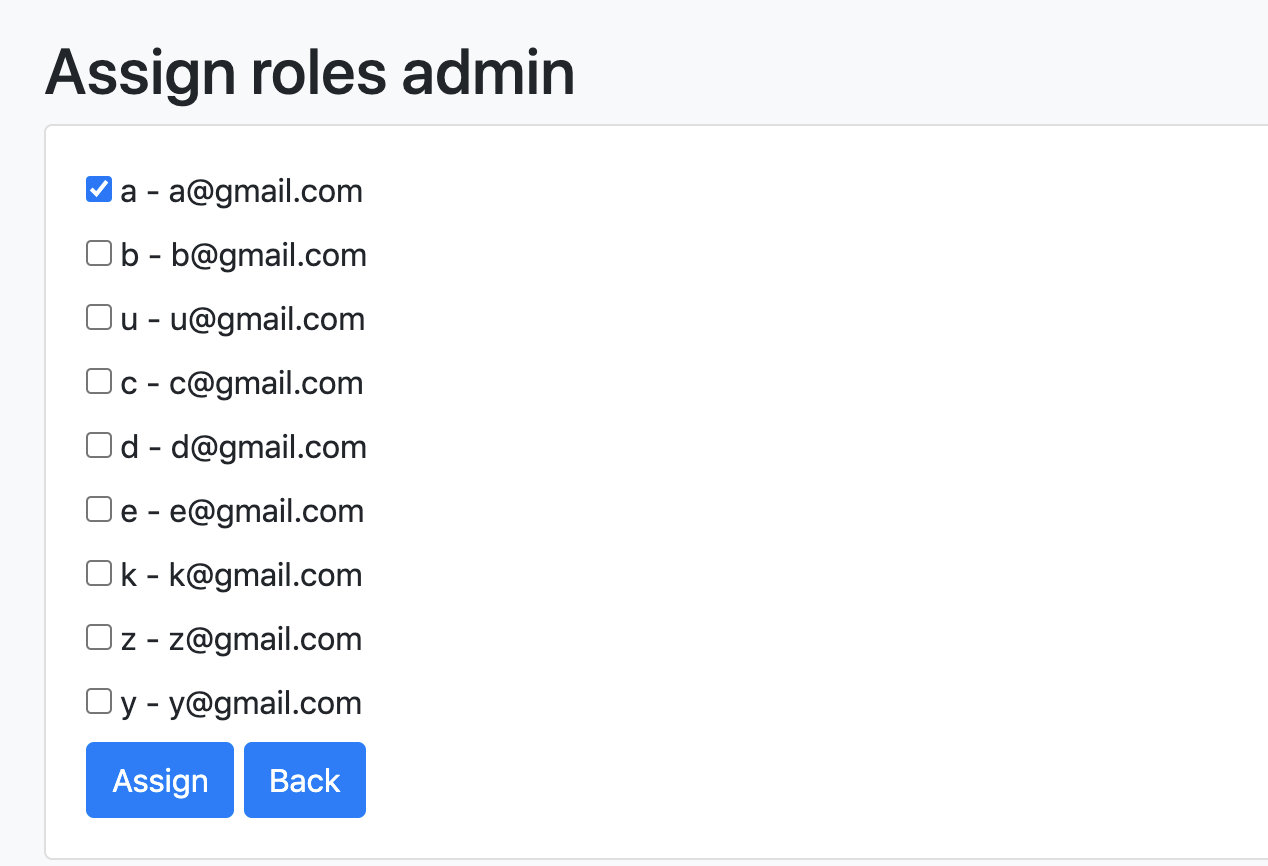


Figure : Giao diện Gán quyền admin cho người dùng

Chỉ những người dùng được check sẽ có quyền admin, còn tất cả những người dùng còn lại thì sẽ phải **bỏ quyền admin nếu có.**

Lưu ý: Để update tất cả record trong CSDL thì dùng hàm query() như sau



Sau khi cập nhật chức vụ cho người dùng thì sẽ tiến hành gửi mail cho người dùng được gán, sử dụng Laravel Notification. Nội dung trong email như sau

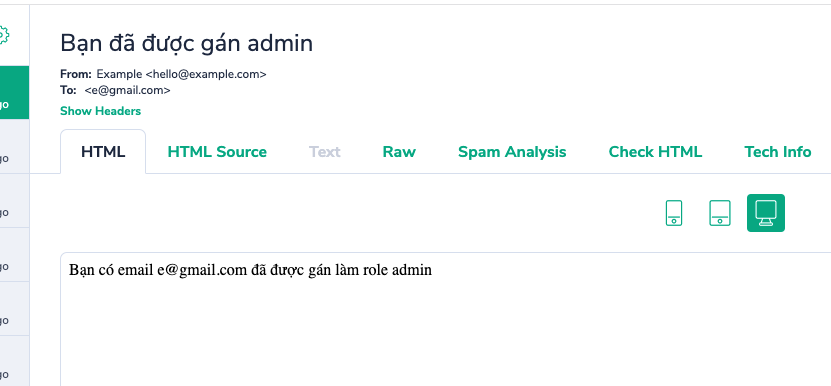


Figure : Gán admin cho người dùng với email xác định

Lưu ý: Trong trường hợp bị như hình dưới đây

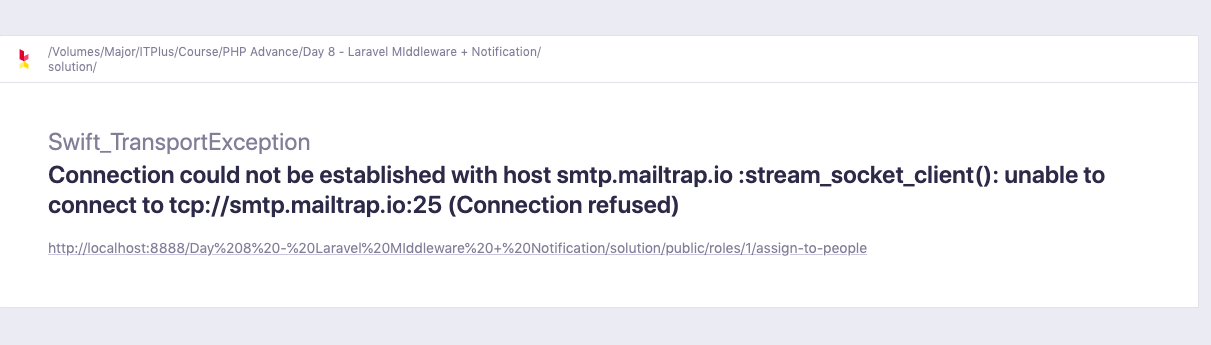


Figure : Lỗi MailTrap không thể gửi mail

Thì hãy đổi cổng **.env** từ 25 -> 465 như sau

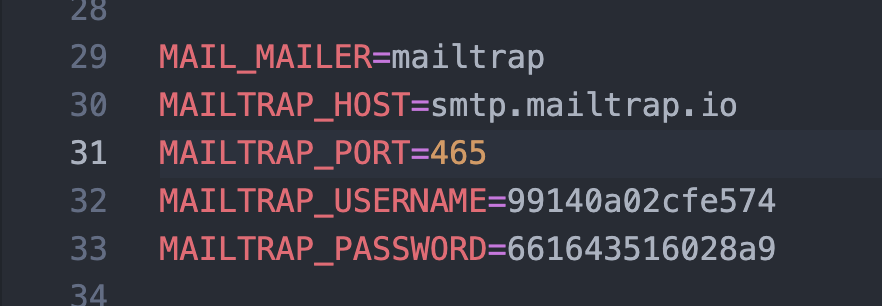


Figure : Đổi công 465 của MailTrap

Phát triển chức năng “**Post**”, tách chức năng thêm mới và sửa Post thành 1 trang riêng biệt, không sử dụng modal. Với nội dung bài Post, sử dụng WYSIWYG editor – ví dụ như sử dụng CKEditor để tiến hành chỉnh sửa/thêm mới nội dung bài Post. Giao diện như sau

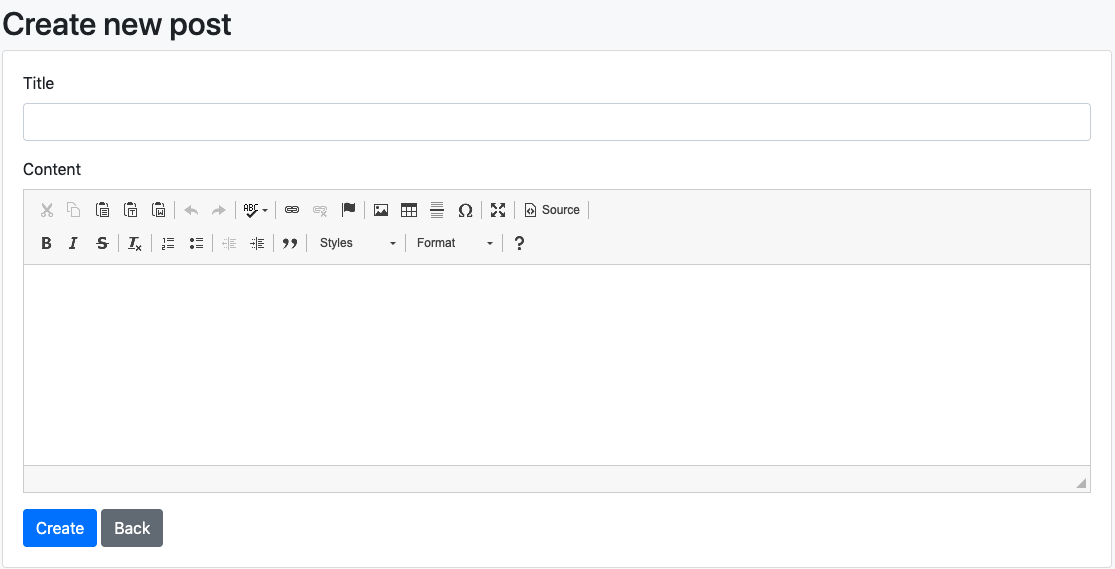


Figure : Giao diện khi có CKEditor

Giao diện WYSIWYG là giao diện cho phép người dùng tuỳ biến Editor thành fully-functional editor, giống MS Word.

Tuy nhiên cần chú ý: Sau khi sử dụng rich text editor thì hiển thị ra ngoài sẽ bị thế này

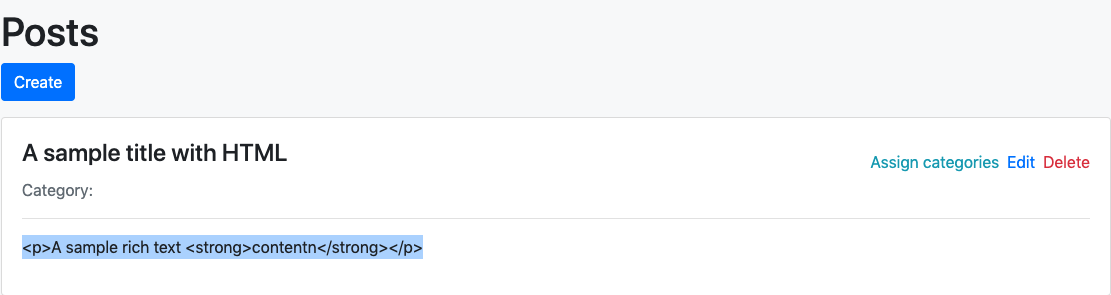


Figure : HTML chưa được escape

Để hiển thị bài Post với đầy đủ tính năng, thay vì sử dụng {{ $variable }} như truyền thống thì sẽ sử dụng {!! $variable !!} để hiển thị HTML



Figure : Hiển thị HTML

Kết quả là hiển thị Post được format

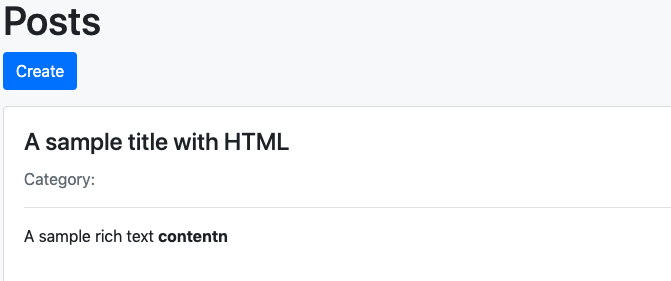


Figure : Post với HTML editor

Hướng dẫn thêm CKEditor vào trong bài Post: <https://cdn.ckeditor.com/>



Figure : Demo nhúng CKEditor4

CKEditor sẽ replace trường trong HTML với name là **‘content’**

**Nhiệm vụ cuối:** Phát triển chức năng xem chi tiết nội dung của bài **Post**.