Thiết kê chi tiết server-side cho trang FeelEvent

1PAC. INC.

Version 0.1.1
2016-09-09

##

1. MỞ ĐẦU

Đây là bảng thiết kế chi tiết server-end của trang **World School** (FeelEvent), một trang được tách ra từ Feelnote .

1.1. Chú thích về mặt kỹ thuật

- Vì các thông tin về hội thảo (event) cũng được sử dụng ở Feelnote nên tất cả các bảng (table) của cơ sở dữ liệu sẽ được cấu trúc tại Feelnote DB .
- Mặc dù giao diện của Feelnote được thiết kế theo cơ cấu SPA (Single Page Application) , nhưng trang **World School** là một trang web mở được công bố rộng rãi, nên dựa trên quan điểm SEO giao diện của nó sẽ được thiết kế theo phương thức truyền thống MPA(Multi Page Application) .

1.2. Cấu hình hệ thống

- Nginx
- Ruby (phiên bản 2.3.1)
- Ruby on Rails (phiên bản 5.0.0.1)
- MySQL (phiên bản 5.7)
- Redis

1.3. Nguyên tắc code

- Nguyên tắc code cơ bản
  - https://github.com/fortissimo1997/ruby-style-guide/blob/japanese/README.md
  - https://github.com/fortissimo1997/ruby-style-guide/blob/japanese/README.ja.md
- Các nguyên tắc code khác của Ruby (bổ sung cho nguyên tắc cơ bản ở trên)
  - https://github.com/styleguide/ruby
- Nguyên tắc code cơ bản cho Rails
  - https://github.com/bbatsov/rails-style-guide/blob/master/README.md

1.3.1. Nguyên tắc code riêng của 1PAC

Mục định nghĩa

- Không thêm dòng trống vào sau `class`.
- Không thêm dòng trống vào giữa 2 `end` cuối cùng.

Cách dùng unless, if đặt sau

Các nguyên tác tối thiểu dưới đây được đặt ra để giữ cho code vẫn có thể dễ hiểu khi có nhiều xử lý phân nhánh if.(trừ một số ngoại lệ).

- Chỉ được áp dụng if, unless đặt sau khi sử dùng `return`, `raise` để thực hiện ngưng xử lý. Ví dụ:

```
return unless @user.signed_in?
```

- Đối với trường hợp thêm xử lý hoặc đặt giá trị cho biến số và tiếp tục xử lý sau đó, thì dù cho có là 1 dòng đi nữa cũng phải dùng cấu trúc if bình thường. Ví dụ:

```
if @flow.init?
  @user.assign_attributes @flow.get_data(:form)
end
```

2. Cấu trúc thư mục

Dưới đây là thông tin về cấu trúc thư mục(directory) và tập tin (file) của application.
* Chỉ trích dẫn các directory và file chính.

```
.
├── Gemfile      # File quản lý, thiết định các gem cơ bản
├── Gemfile.lock # File quản lý, giữ phiên bản tất cả các gem sử dụng trong app
├── Rakefile     # Thiết định các task có thể chạy từ terminal
├── app/         # Thư mục chính lưu giữ application
│   ├── assets/      # Thư mục chứa các file hình ảnh và style sheets
│   ├── cells/       # Thư mục chứa có xử lý logic của gem Cells
│   ├── controllers/ # Thư mục chứa các controller của Rails
│   ├── helpers/     # Thư mục chứa các module hỗ trợ cho Rails
│   ├── mailers/     # Thư mục chứa các controller cho xử lý email
│   ├── models/      # Thư mục chứa các định nghĩa model của Rails
│   └── views/       # Thư mục chứa các view của Rails
├── config/ # Thư mục chứa các file thiết định của project, application
│   ├── application.rb # File ghi thiết định chung của application cho mọi môi trường
│   ├── boot.rb # File khởi động chính
│   ├── database.yml # File ghi thiết định cơ sở dư liệu(database)
│   ├── environment.rb # File thiết định các môi trường
│   ├── environments/ # Thư mục chứa file ghi thiết định riêng cho mỗi môi  trường
│   ├── initializers/ # Thư mục chứa các file thiết lập cơ bản ban đầu
│   ├── locales/ # Thư mục chứa các file từ điển định nghĩa chung
│   └── routes.rb # File thiết lập routing của Rails
├── config.ru # File thiết định của Racka
├── db/       # Thư mục chứa các thông tin của database
├── doc/      # Thư mục chứa các tài liệu hướng dẫn, thiết kế
├── lib/      # Thư mục chứa các libray của Rails
├── log/      # Thư mục chứa các file log
├── public/   # Thư mục gốc của web server
├── spec/     # Thư mục chứa các file code kiểm tra (test code)
├── tmp/      # Thư mục chứa các file, thông tin lưu dữ tạm thời
└── vendor/   # Thư mục chứa các library bên ngoài (ví dụ gem của rails)
```

3. Bố cục (layout), bộ phận (partial)

3.1. Layout

Sử dụng các layout có sẵn của `Rails`.
Vì đây là một web application đơn giản, nên chỉ sử dụng các layout có sẵn cơ bản của `application`

- Thư mục chứa các tập tin layout
  - app/views/layouts/
- File ghi định dạng layout chung
  - application.html.erb

3.2. Partial

Dưới đây là thông tin về các tập tin chứa partial chung như header, footer. * Thư mục chứa các partial dùng chung ** app/views/shared/

3.2.1. Các partial chính

```
app/views/shared/
  ├── _body_bottom.html.erb    # Chứa phần chung đặt ngay trước thẻ </body>
  ├── _body_top.html.erb       # Chứa phần chung đặt ngay sau thẻ <body>
  ├── _global_header.html.erb  # Chứa global header
  └── _global_footer.html.erb  # Chứa global footer
```

3.2.2. Ví dụ

```
<%= render 'shared/global_header' %>
# nội dung của `app/views/shared/_global_header.html.erb` sẽ được chèn vào

<%= render 'shared/global_footer', bar: 'baz' %>
# Trường hợp muốn đưa tham số cho partial(có thể tỉnh lược locals:)
```

4. Bộ nhớ đệm (Cells)

Nhằm giảm bớt xử lý trên database system, chương trình sẽ dùng gem `Cells` để tạo các tập tin bộ nhớ đệm.

4.1. Cells là gì ?

Đây là một Gem có thể xử lý cả logic(controller) lẫn view (partial)
Ví dụ, chúng sẽ được dùng trong các xử lý xuất các mục của side bar.

Các thông tin khác về Cells có thể xem tại link Github dưới
https://github.com/apotonick/cells

4.2. Các dùng

- Thư mục chứa các xử lý logic (controller)
  - app/cells/
- Cách đặt tên file
  - File có phần mở rộng là `rb` và phần tên có kết thúc là `_cell` (`*_cell.rb`)

4.3. Tập tin chứa view

Mặc dù trong thiết định thông thường của gem `Cells` thì tập tin chứa view sẽ được đặt tại `app/cells/`, nhưng nhằm mục đích tập trung tất cả các view về một nơi, view sẽ được đặt tại `app/views/`

5. Tips

Hướng dẫn chung về nhưng `Gem` khác.

5.1. Decorator (Draper)

Gem tạo ra một lớp Presenter(Draper) nằm giữa view và model nhằm giảm thiểu các logic gây khó hiểu không cần thiết trong model và view.

5.1.1. Ví dụ

app/decorators/user_decorator.rb

```
class UserDecorator < Draper::Decorator
  delegate_all

  def full_name
    "#{first_name} #{last_name}"
  end
end
```

app/views/me/sample.html.erb

```
<h1><%= @user.full_name %></h1>
```

Các thông tin khác của Draper có thể xem tại link Github dưới
https://github.com/drapergem/draper

6. Các lưu ý khi lập trình

Dưới đây là 2 lưu ý chính trong quá trình lập trình. * Sử dụng `RSpec` để thực hiện các kiểm tra. * Bắt buộc phải tạo `Pull Request` để có thể kiểm tra chéo code của nhau.

6.1. Sử dụng RSpec để kiểm tra trong lập trình

Web Application này không sử dụng `Minitest` của `Rails` mà sẽ dùng `RSpec` để thực hiện kiểm tra code trong lập trình. * FactoryGirl **Mặc dù Rails có chức năng** `Fixture` **để tạo test data, nhưng application lần này sẽ sử dụng** `FactoryGirl` **.** * **Faker** Một `Gem` chuyên dùng để tạo các dữ liệu ảo trong test.

6.1.1. Các loại TEST

- Test đơn thể độc lập
  - Thực hiện kiểm tra các method của Model, Decorator, Helper
- Test dành cho controller
  - Dùng kiểm tra cả quá trình tạo view, ngăn ngừa lỗi gọi method không được định nghĩa, hoặc gây lỗi trong view.

Sách tham khảo

> Everyday Rails - RSpec###Rails#####
> https://leanpub.com/everydayrailsrspec-jp/read

6.2. Kiểm tra code chéo dựa trên Pull-Request

Thực hiện tạo branch và chia nhỏ các commit theo mức hợp lý. Thực hiện kiểm tra kỹ trên máy tính của mình trước khi đưa lên Github tạo Pull-Request và nhờ người khác kiểm tra chéo.
* Về cơ bản, chỉ được thực hiện `Merge` sau khi **đã được kiểm tra chéo và có comment đồng ý.**

6.2.1. Các kiểm tra tối thiểu trước khi nhờ kiểm tra chéo

Thực hiện tất cả các điều dưới đây trước khi nhờ kiểm tra chéo.

- Kiểm tra xem có code không phù hợp với nội dung của branch không.

- Message trong mỗi commit có đơn giản, dễ hiểu không.
- Đã thực hiện tinh chỉnh, tối giản code chưa (refactoring)
- Method có thiết định public, private phù hợp hay chưa.
- Code có thực sự dê đọc và dễ hiểu không
- Tên của Class/Method/biến số có phù hợp không
- Có trùng lặp với các code đã có sẵn hay không
- Có gây ảnh hưởng tới các code đã có sẵn không
- Có các comment tại vị trí cần thiết hay chưa
- Đã xoá hết các code dùng trong debbug chưa
- Có thể viết code một cách khác nào tốt hơn hay không
- Tên file có hợp lý chưa
- Các file không cần thiết có bị commit chung hay không
- Có chứa lỗi đánh máy không
- Có tuân thủ đúng các qui tắc lập trình không
- Validate trong model có phù hợp chưa
- Có vấn đề gì về mặt bảo mật không
- Code có đi lệch khỏi các qui tắc của framework không
- Có chứa các sử lý mà có thể thay thế bằng library có sẵn không
- Có gây các tiềm ẩn gây nặng xử lý, hoặc nguy hiểm về bảo mật trong tương lai không
- Có cần thực hiện xử lý transaction cho cơ sở dữ liệu không
- Có cần sử dụng bộ nhớ đệm để giảm thiểu xử lý không
- Có chứa các SQL không cần thiết không
- Code kiểm tra có chứa đủ các trường hợp hay chưa
- Chương trình có chạy đúng tại nhưng điều kiện đặc biệt không
- Xử lý chặn lỗi (error handling) có được thực hiện phù hợp, đầy đủ chưa

7. A-1 Login

7.1. Thông số cơ bản

Method, URL	GET|POST /signin
Controller, Action	sessions#signin
Tên route	new_user_session
View	devise/sessions/new.html.erb

7.2. Request

Parameter	Chi tiết	Bắt buộc	Ghi chú
event_user[email]	Email	Có	
event_user[password]	Mật khẩu	Có	
event_user[remember_me]	Giữ thông tin login		

7.3. Chú thích đặc biệt cho (Devise)

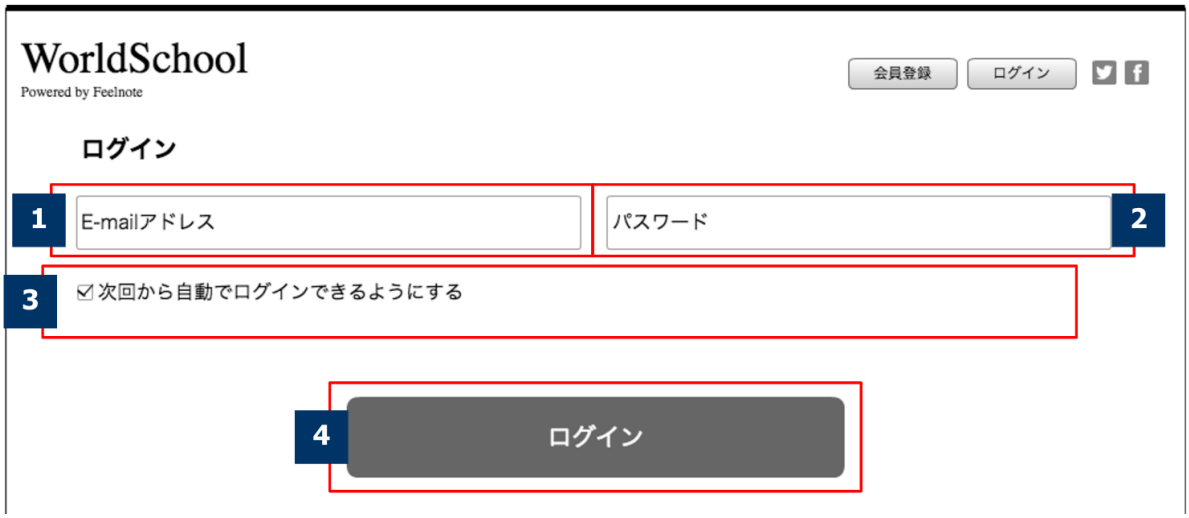
Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

> Có thể xem thông tin của Devise tại link Github dưới
> https://github.com/plataformatec/devise

7.3.1. Thông số thiết định Devise

- Tên table chứa thông `event_users`
- Tên cột chưa thông tin id đăng nhập `email`
- Tên cột chứa mật khẩu `password (encrypted_password)`

7.4. Chi tiết



1. Email
    - Xuất dưới dạng tag là input với hạng mục là `email`
2. Mật khẩu
    - Xuất dưới dạng tag là input với hạng mục là `password`
    - Nếu xuất hiện màn hình lỗi sau ấn login, không chứa nội dung đã nhập trước đó
3. Lưu lại login
    - Xuất dưới dạng tag là input với hạng mục là `checkbox`
    - Giá trị mặc định là không được chọn
4. Nút Login
    - Trường hợp **Login thành công**
        - Sử dụng `Devise` tạo session
        - Chuyển về trang `B-1`
    - Trường hợp **Login thất bại**
        - Trả lại màn hình này và xuất các thông báo lỗi cần thiết.

8. A-2 Logout

8.1. Thông số cơ bản

Method, URL	DELETE /signout
Controller,Action	sessions#destroy
Tên route	destroy_user_session
View	Không có

8.2. Request

- Không có

8.3. Chú thích đặc biệt cho (Devise)

Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

Có thể xem thông tin của Devise tại link Github dưới
https://github.com/plataformatec/devise

8.3.1. Thông số thiết định Devise

- Tên table chứa thông `event_users`
- Tên cột chưa thông tin id đăng nhập `email`
- Tên cột chứa mật khẩu `password (encrypted_password)`

8.4. Chi tiết

1. Chi tiết chung
    - Sử dụng `Devise` cho sử lý ngừng sesion
    - Chuyển về trang `B-1`

9. A-3 Đổi mật khẩu(Nhập dữ liệu)

9.1. Thông số cơ bản

Method, URL	GET /password/new
Controller, Action	devise/password#new
Tên route	new_user_password
View	devise/password/new.html.erb

9.2. Request

- Không có

9.3. Chú thích đặc biệt cho (Devise)

Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

Có thể xem thông tin của Devise tại link Github dưới
https://github.com/plataformatec/devise

9.3.1. Thông số thiết định Devise

- Tên table chứa thông `event_users`
- Tên cột chưa thông tin id đăng nhập `email`
- Tên cột chứa mật khẩu `password (encrypted_password)`

9.4. Chi tiết

1. Địa chỉ email
    - Xuất dưới dạng tag input với hạng mục là `email`
2. Nút gửi đi
    - Gửi thông tin đến `A-3-a` bằng method POST.

10. A-3-a Đổi mật khẩu (Xử lý)

10.1. Thông số cơ bản

Method,URL	POST /password
Controller, Action	devise/password#create
Tên route	user_password
View	devise/password/new_complete.html.erb
View của email	devise/mailer/reset_password_instructions.html.erb

10.2. Request

Parameter	Chi tiết	Bắt buộc	Ghi chú
event_user[email]	Email	Có	

10.3. Chú thích đặc biệt cho (Devise)

Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

Có thể xem thông tin của Devise tại link Github dưới
https://github.com/plataformatec/devise

10.3.1. Thông số thiết định Devise

- Tên table chứa thông `event_users`
- Tên cột chưa thông tin id đăng nhập `email`
- Tên cột chứa mật khẩu `password (encrypted_password)`

10.4. Chi tiết

- Trường hợp thành công (Validate thành công)
  - Dùng `Devise` thực hiện các sử lý chuẩn bị cho thay đổi mật khẩu
    - Xử lý gửi email đi có sẵn trong `Devise`
    - Thay đổi mẫu chung của email cho phù hợp
- Trường hợp thất bại(Lỗi không qua được validate)
  - Chuyển về màn hinh nhập `A-3` và hiện thông số lỗi cần thiết

11. A-4 Tạo lại mật khẩu (Nhập dữ liệu)

11.1. Thông số cơ bản

Method, URL	GET /password/edit
Controller,Action	devise/password#edit
Tên route	edit_user_password
View	devise/password/edit.html.erb

11.2. Request

Parameter	Chi tiết	Bắt buộc	Ghi chú
reset_password_token	Chuỗi nhận dạng (Token)	Có	

11.3. Chú thích đặc biệt cho (Devise)

Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

Có thể xem thông tin của Devise tại link Github dưới
https://github.com/plataformatec/devise

11.3.1. Thông số thiết định Devise

- Tên table chứa thông `event_users`
- Tên cột chưa thông tin id đăng nhập `email`
- Tên cột chứa mật khẩu `password (encrypted_password)`

11.4. Chi tiết

1. Chi tiết chung
    - Sử dụng `Devise` để kiểm tra token
2. Mật khẩu

- Xuất dưới dạng tag input vơi hạng mục là `password`
- Không chứa giá trị khi bị trả lại màn hình lỗi

3. Mật khẩu(Xác nhận)

- Xuất dưới dạng tag input vơi hạng mục là `password`
- Không chứa giá trị khi bị trả lại màn hình lỗi

12. A-4-a Tạo lại mật khẩu (Xử lý)

12.1. Thông số cơ bản

Method,URL	PATCH /password
Controller, Action	devise/password#update
Tên route	(default)
View	Không có

12.2. Request

Parameter	Chi tiết	Bắt buộc	Ghi chú
event_user[password]	Mật khẩu	Có	
event_user[password_confirmation]	Xác nhận mật khẩu	Có	

12.3. Chú thích đặc biệt cho (Devise)

Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

Có thể xem thông tin của Devise tại link Github dưới
https://github.com/plataformatec/devise

12.3.1. Thông số thiết định Devise

- Tên table chứa thông `event_users`
- Tên cột chưa thông tin id đăng nhập `email`
- Tên cột chứa mật khẩu `password (encrypted_password)`

12.4. Chi tiết

- Trường hợp thành công (Validate thành công)
  - Sử dụng `Devise` để xử lý chuẩn bị tạo lại mật khẩu
    - Xử lý gửi email cũng có sẵn trong `Devise`
    - Chỉnh sửa mẫu chung cho phù hợp nội dung
- Trường hợp thất bại (Lỗi không qua validate)
  - Chuyển về màn hình `A-3` và hiển thị thông tin lỗi cần thiết.

#include::src/backend_site/vn/C-1.adoc[]

#include::src/backend_site/vn/D-1.adoc[]