Thiết kê chi tiết server-side cho trang FeelEvent

1PAC. INC.

Bản hiệu chỉnh 0.2.3
2016-09-19

Mục lục

1. MỞ ĐẦU 1
2. Cấu trúc thư mục 3
3. Bố cục (layout), bộ phận (partial) 4
4. Bộ nhớ đệm (Cells) 5
5. Tips 6
6. Các lưu ý khi lập trình 8
7. A-1 Login 10
8. A-2 Logout 13
9. A-3 Đổi mật khẩu(Nhập dữ liệu) 14
10. A-3-a Đổi mật khẩu (Xử lý) 16
11. A-4 Tạo lại mật khẩu (Nhập dữ liệu) 18
12. A-4-a Tạo lại mật khẩu (Xử lý) 20
13. B-1 Top page 22
14. C-1 My Page (WIP) 24
15. D-1 Đăng kí thành viên mới 25
16. D-1-a Đăng kí thành viên mới (Xử lý) 27
17. D-2 Đăng ki thành viên mới (Đăng kí chính thức) 28

Chương 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
```

Chương 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
│   ├── serializers/      # Thư mục dùng cho gem ActiveModelSerializers
│   └── 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)
```

Chương 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:)
```

Chương 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/`

Chương 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

5.2. ActiveModelSerializers

Đây là library dùng để chuyển dạng dữ liệu của ActiveModel sang Serialize và chuyển sang JSON

5.2.1. Example

app/models/book.rb

```
class Book < ActiveRecord::Base
  belongs_to :publisher
  has_and_belongs_to_many :authors
end
```

app/serializers/book_serializer.rb

Chỉ định mục cần xuất bằng cách đặt tham số cho attributes

```
class BookSerializer < ActiveModel::Serializer
  attributes :id, :title, :price, :published_at, :publisher_id
end
```

app/controllers/books_controller.rb

```
def index
  @books = Book.all
  respond_to do |format|
    format.html
    format.json { render json: @books }
  end
end
```

Ngoài các chức năng đưa ra ở trên, còn có những chức năng thay đổi nội dung tuỳ vào điều kiện khác hoặc dựa trên Asscociation. Nếu cần thiết có thể xem tại trang chủ của Gem.

> Xem chi tiết về ActiveModelSerializers tại đường dẫn Github bên dưới
> https://github.com/rails-api/active_model_serializers

Chương 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によるRailsテスト入門
> 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

Chương 7. A-1 Login

7.1. Thông số cơ bản

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

7.2. Request

- Chi tiết ở bảng định nghĩa FORM bên dưới.

7.3. Chú thích đặc biệt cho (Devise)

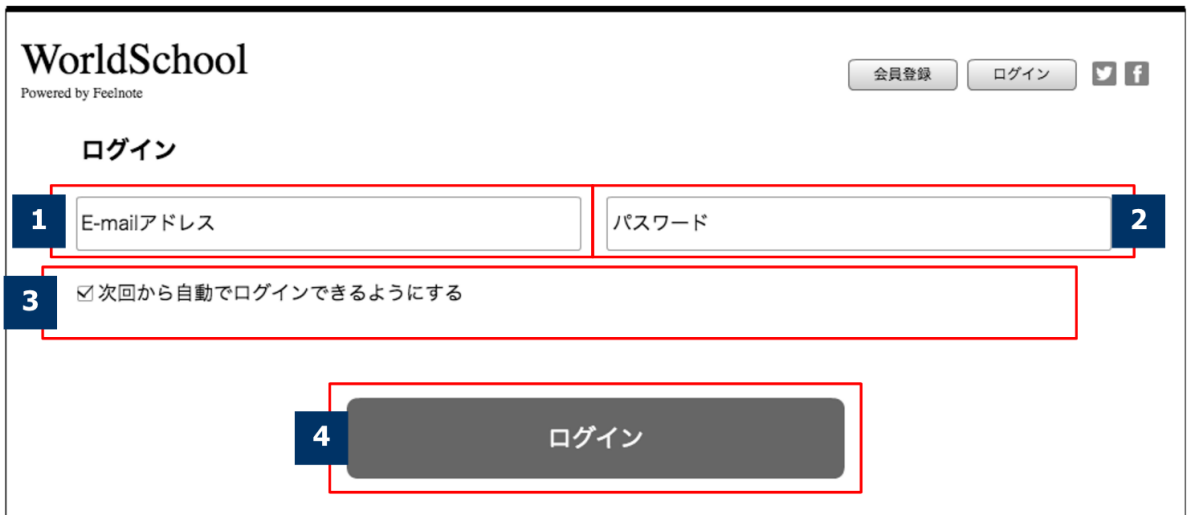
Trang web này sử dụng gem `Devise` để thực hiện các thao tác liên quan đến quản lý User và xử lý đăng nhập

> 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.

7.5. Định nghĩa FORM

ID	Tên hiển thị/Tên field ở form	Hạng mục của form	Bắt buộc	Ghi chú
01	メールアドレス user[email]	email	○	
02	パスワード user[password]	password	○	
03	ログイン情報の保持 user[remember_me]	checkbox		

Chương 8. A-2 Logout

8.1. Thông số cơ bản

Method, URL	DELETE /signout
Controller,Action	users/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`

Chương 9. A-3 Đổi mật khẩu(Nhập dữ liệu)

9.1. Thông số cơ bản

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

9.2. Request

- Chi tiết ở bảng định nghĩa FORM bên dưới

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.

9.5. Định nghĩa FORM

ID	Tên hiển thị/Tên field ở form	Hạng mục của form	Bắt buộc	Ghi chú
01	メールアドレス user[email]	email	○	

Chương 10. A-3-a Đổi mật khẩu (Xử lý)

10.1. Thông số cơ bản

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

10.2. Request

- Xem ở bảng định nghĩa FORM `A-3`

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

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

11.1. Thông số cơ bản

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

11.2. Request

- Xem ở bảng định nghĩa FORM bên dưới

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

11.5. Định nghĩa FORM

ID	Tên hiển thị/Tên field ở form	Hạng mục của form	Bắt buộc	Ghi chú
01	パスワード user[password]	password	○	
02	パスワード user[password_confirmation]	password	○	

Chương 12. A-4-a Tạo lại mật khẩu (Xử lý)

12.1. Thông số cơ bản

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

12.2. Request

- Xem ở bản định nghĩa FORM `A-4`

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-4` và hiển thị thông tin lỗi cần thiết.

Chương 13. B-1 Top page

13.1. Thông số cơ bản

Method, URL	GET /
Controller, Action	events#index
Tên route	root
View	events/index.html.erb

13.2. Request

- Chi tiết ở trong bảng đỉnh nghĩa FORM bên dưới.

13.3. Chi tiết

1. Các thành phần của FORM.
    - Xuất các thành phần của FORM được ghi trong bảng định nghĩa FORM
2. Nút `Hiển thị thêm` (tiếng Nhật `もっと見る`)
    - Gán URL của trang hiển thị thêm vào `data-url`
    - Nếu không có trang kế tiếp thì không hiển thị nút này.
3. Kết quả tìm kiếm
    - Xử lý hiển thị của lần tìm kiếm đầu tiên sẽ do Rails thực hiện
    - Nếu không có parameter đính kèm thì xử lý tìm kiếm sẽ thực hiện hiện tìm kiếm sau khi xoá (reset) các điều kiện.

13.4. Định nghĩa FORM

ID	Tên hiển thị/Tên field ở form	Hạng mục của form	Bắt buộc	Ghi chú
01	キーワード keyword_ids	checkbox		
02	種別 event_types	checkbox		
03	開催場所 held_area	select(single)		
04	開催開始日 held_started_on	date		・**format:** YYYY-MM-DD
05	開催終了日 held_ended_on	date		・**format:** YYYY-MM-DD
06	参加費 entry_fee_type	select(single)		・**pattern:** (free:無料, pay:有料)
07	対象年齢 qualifying_age_id	select(single)		
08	フリーワード word	text		

Chương 14. C-1 My Page (WIP)

14.1. Thông số cơ bản

Method, URL	GET /me
Controller, Action	me#show
Tên route	me
View	me/show.html.erb

14.2. Request

- Không có

14.3. Chi tiết

1. TODO

Chương 15. D-1 Đăng kí thành viên mới

15.1. Thông số cơ bản

Method, URL	GET /users/signup
Controller, Acition	users/registrations#new
Tên route	new_user_registration
View	users/registrations/new.html.erb

15.2. Request

- Xem chi tiết ở bảng định nghĩa FORM bên dưới

15.3. 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

15.3.1. 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)`

15.4. Chi tiết

1. Form
    - Xuất các thành phần ghi trong định nghĩa Form
2. Nút `Gửi đi` (tiếng Nhật `送信`)
    - Gửi bằng method `POST` đến `D-1-a`

15.5. Định nghĩa FORM

ID	Tên hiển thị/Tên field ở form	Hạng mục của form	Bắt buộc	Ghi chú
01	対象年齢 user[qualifying_age_id]	select(single)	○	
02	メールアドレス user[email]	email	○	
03	パスワード user[password]	password	○	
04	パスワード(確認用) user[password_confirmation]	password	○	
05	キーワード user[keyword_ids]	checkbox		
06	種別 user[event_types]	checkbox		
07	ニュースレター受け取りフラグ user[subscribe_newsletter]	checkbox		

Chương 16. D-1-a Đăng kí thành viên mới (Xử lý)

16.1. Thông số cơ bản

Method, URL	POST /users
Controller, Action	users/registrations#create
Tên route	user_registration
View	users/registrations/new_complete.html.erb
Mail view	users/mailer/confirmation_instructions.html.erb

16.2. Requests

- Xem tại bảng định nghĩa FORM `D-1`

16.3. Chi tiết

- Khi xử lý thành công (kiểm tra dữ liệu nhập, validate, không bị lỗi)
  - Dùng `Devise` trong xử lý đăng kí tạm thời.

    # Xử lý gửi mail cũng dùng chức năng có sẵn của `Devise`

    # Tạo các thông tin sau cho User

      - Lứa tuổi đối tượng(event_qualifying_ages_event_users)
      - Keyword(event_users_keywords)
      - Hình thức event(event_types_event_users)
      - Hiển thị màn hình thông báo đã thành công trong đăng kí tạm thời
- Khi có lỗi(dữ liệu nhập có lỗi, validate không thành công)
  - Chuyển về màn hình nhập dữ liệu `D-1` và hiển thị các thông tin lỗi cần thiết

Chương 17. D-2 Đăng ki thành viên mới (Đăng kí chính thức)

17.1. Thông số cơ bản

Method, URL	GET /users/confirmation
Controller, Action	users/confirmations#show
Tên route	new_user_confirmation
View	Không có

17.2. Requests

Parameters	Chi tiết	Bắt buộc	Ghi chú
confirmation_token	Token	Có	

17.3. 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

17.3.1. 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)`

17.4. Chi tiết

- Khi xử lý thành công (kiểm tra dữ liệu nhập, validate, không bị lỗi)
  - Dùng `Devise` để xử lý đăng kí chính thức
  - Chuyển sang trang `A-1` sau khi hoàn tất đăng kí
- Khi có lỗi (dữ liệu nhập có lỗi, validate không thành công)
  - Hiện thị lỗi tại màn hình `users/confirmations/new.html.erb`