**用Ruby on Rails创建授权访问项目**

**一、创建项目**

* 输入rails new *CuisineApp*创建项目
* 输入cd *cuisineapp*进入项目子目录

**二、创建模型(Model)**

* 输入rails generate model *Cuisine*创建第一个模型*Cuisine*
* 编辑app/models/*cuisine*.rb如下：

class *Cuisine* < ActiveRecord::Base

has\_many :*recipes* #表示每个*cuisine*表可对应多个*recipe*表

end

* 编辑db/migrate/*<timestamp1>*\_create\_*cuisine*s.rb文件，其中*timestamp1*为创建*Cuisine*模型的时间戳，添加字段如下：

class Create*Cuisine*s < ActiveRecord::Migration

def change

create\_table :*cuisine*sdo |*t*|

*t*.string :*region* #添加*region*字符串字段

*t*.string :*image* #添加*image*字符串字段

*t*.timestamps null: false #生成created\_at和updated\_at字段

end

end

end

* 输入rails generate model *Recipe*创建第二个模型*Recipe*
* 编辑app/models/*recipe*.rb如下：

class *Recipe* < ActiveRecord::Base

belongs\_to :*cuisine* #表示每个*recipe*表属于单一*cuisine*表

end

* 编辑db/migrate/*<timestamp2>*\_create\_*recipe*s.rb文件，其中*timestamp2*为创建*Recipe*模型的时间戳，添加字段如下：

class Create*Recipe*s < ActiveRecord::Migration

def change

create\_table :*recipe*sdo |*t*|

*t*.string :*name* #添加*name*字符串字段

*t*.string :*image* #添加*image*字符串字段

*t*.text :*ingredients* #添加*ingredients*文本字段

*t*.text :*instructions* #添加*instructions*文本字段

*t*.references :*cuisine* #添加*cuisine*\_id外键字段用于关联*cuisine*

*t*.timestamps null: false #生成created\_at和updated\_at字段

end

end

end

* 输入rails generate model *User*创建第三个模型*User*
* 编辑app/models/*user*.rb如下：

class *User* < ActiveRecord::Base

has\_secure\_password #使用安全保存密码的方法，此方法依赖bcrypt Gem

def *editor*? #判断记录的*role*属性值是否为*editor*

self.*role* == '*editor*'

end

def *admin*? #判断记录的*role*属性值是否为*admin*

self.role == '*admin*'

end

end

* 编辑Gemfile，添加一行内容：

gem 'bcrypt', '~> 3.1.7'

* 输入bundle install安装bcrypt
* 编辑db/migrate/*<timestamp3>*\_create\_*user*s.rb文件，其中*timestamp3*为创建*User*模型的时间戳，添加字段如下：

class Create*User*s < ActiveRecord::Migration

def change

create\_table :*user*sdo |*t*|

*t*.string :*fname* #添加*fname*字符串字段

*t*.string :*lname* #添加*lname*字符串字段

*t*.string :*email* #添加*email*字符串字段

*t*.string :*password*\_digest #添加*password*\_digest字符串字段

#由bcrypt生成*password*的散列值

*t*.string :*role* #添加*role*字符串字段

*t*.timestamps null: false #生成created\_at和updated\_at字段

end

end

end

* 输入rake db:migrate以更新数据库结构
* 编辑db/seeds.rb样本数据文件如下：

*mateo* = *User*.create(*fname*: "*Mateo*", *lname*: "...", *email*: "...", *password*: "...", *password*\_confirmation: "...", *role*: "*editor*")

*julian* = *User*.create(*fname*: "*Julian*", *lname*: "...", *email*: "...", *password*: "...", *password*\_confirmation: "...")

*freida* = *User*.create(*fname*: "*Freida*", *lname*: "...", *email*: "...", *password*: "...", *password*\_confirmation: "...", *role*: "*admin*")

*c1* = *Cuisine*.create(*region*: "...", *image*: "#")

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c1*.id)

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c1*.id)

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c1*.id)

*c2* = *Cuisine*.create(*region*: "...", *image*: "#")

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c2*.id)

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c2*.id)

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c2*.id)

*c3* = *Cuisine*.create(*region*: "...", *image*: "#")

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c3*.id)

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c3*.id)

*Recipe*.create(*name*: "...", *image*: "#", *ingredients*: "...",

*instructions*: "...", *cuisine*\_id: *c3*.id)

* 输入rake db:seed将样本数据添加到数据库

**三、创建控制器(Controller)**

* 编辑app/controllers/application\_controller.rb文件，添加控制器逻辑如下：

class ApplicationController < ActionController::Base

protect\_from\_forgery with: :exception

helper\_method :*current\_user* #将*current\_user*作为辅助方法，

#供视图逻辑调用，默认Application控制器定义的方法可被其它控制器调用

def *current\_user* #返回当前会话的用户id，若尚未登录则为空值

@*current\_user* ||= *User*.find(session[:*user*\_id]) if

session[:*user*\_id]

end

def *require\_user* #将尚未登陆的用户导向登陆页面

redirect\_to *login*\_url unless *current\_user*

end

def *require\_editor*

redirect\_to root\_url unless *current\_user*.*editor*?

end

def *require\_admin*

redirect\_to root\_url unless *current\_user*.*admin*?

end

end

* 输入rails generate controller *Cuisines*创建控制器*Cuisines*
* 编辑app/controllers/*cuisines*\_controller.rb文件，添加控制器逻辑如下：

class *Cuisines*Controller < ApplicationController

def *index* # 根据ruby标准定义，*index*代表显示所有记录信息

@*cuisines* = *Cuisine*.all #将模型从数据库取得的所有数据传给视图

end

def *show* # 根据ruby标准定义，*show*代表显示某条记录信息

@*cuisine* = *Cuisine*.find(params[:id])

#查找:id对应的数据并传给视图

@*recipes* = @*cuisine*.*recipes*

# 查找@*cuisine*记录对应的所有*recipe*数据并传给视图

end

end

* 输入rails generate controller *Recipes*创建控制器*Recipes*
* 编辑app/controllers/*recipes*\_controller.rb文件，添加控制器逻辑如下：

class *Recipe*sController < ApplicationController

before\_action

:*require\_user*, only: [:*show*, :*edit*, :*update*, :*destroy*]

before\_action :*require\_editor*, only: [:*edit*, :*update*]

before\_action :*require\_admin*, only: [:*destroy*]

def *show*

@*recipe* = *Recipe*.find(params[:id])

end

def *edit*

@*recipe* = *Recipe*.find(params[:id])

end

def *update*

@*recipe* = *Recipe*.find(params[:id])

if @*recipe*.update(recipe\_params)

redirect\_to @*recipe*

else

render '*edit*'

end

end

def *destroy*

@*recipe* = *Recipe*.find(params[:id])

@*recipe*.destroy

redirect\_to root\_url

end

private

def *recipe*\_params

params.require(:*recipe*).permit(

:*name*, :*ingredients*, :*instructions*)

end

end

* 输入rails generate controller *Users*创建控制器*Users*
* 编辑app/controllers/*users*\_controller.rb文件，添加控制器逻辑如下：

class *Users*Controller < ApplicationController

def *new*

@*user* = *User*.new

end

def *create*

@*user* = *User*.new(*user*\_params)

if @*user*.save

session[:*user*\_id] = @*user*.id #创建新会话

redirect\_to root\_url

else

redirect\_to *signup*\_url

end

end

private

def *user*\_params

params.require(:*user*).permit(:*name*, :*email*, :*password*,

:*password*\_confirmation)

end

end

* 输入rails generate controller *Sessions*创建控制器*Sessions*
* 编辑app/controllers/*sessions*\_controller.rb文件，添加控制器逻辑如下：

class *Sessions*Controller < ApplicationController

def *new* #根据ruby标准定义，*new*代表返回HTML输入表单用于创建新记录

end

def *create* #根据ruby标准定义，*create*代表创建新记录

@*user* = *User*.find\_by\_email(params[:*session*][:*email*])

if @*user* && @*user*.authenticate(params[:*session*][:*password*])

session[:*user*\_id] = @*user*.id #创建新会话

redirect\_to root\_url

else

redirect\_to *login*\_url

end

end

def *destroy* #根据ruby标准定义，*destroy*代表删除记录

session[:*user*\_id] = nil #移除当前会话

redirect\_to root\_url

end

end

**四、创建路由(Routes)**

* 编辑config/routes.rb文件，添加路由规则如下：

Rails.application.routes.draw do

root '*cuisines*#*index*'

get '*signup*' => '*users*#*new*'

get '*login*' => '*sessions*#*new*'

post '*login*' => '*sessions*#*create*'

delete '*logout*' => '*sessions*#*destroy*'

resources :*users*

resources :*cuisines*, only: [:*index*, :*show*]

resources :*recipes*, only: [:*show*, :*edit*, :*update*, :*destroy*]

end

**五、创建视图(View)**

* 编辑app/views/layouts/application.html.erb网页模板文件，在body元素中添加显示逻辑如下：

<div class="header">

<div class="container">

<div class="nav">

<% if *current\_user* %>

<!-- 若用户已登录，显示用户电邮地址及登出链接 -->

<ul>

<li><%= *current\_user*.*email* %></li>

<li>

<%= link\_to "*Logout*", *logout*\_path, method:"delete" %>

</li>

</ul>

<% else %>

<!-- 若用户尚未登陆，显示登陆链接和注册链接 -->

<ul>

<li><%= link\_to "*Login*", *login*\_path %></li>

<li><%= link\_to "*Signup*", *signup*\_path %></li>

</ul>

<% end %>

</div>

<div class="logo">

</div>

</div>

</div>

* 新建app/views/*cuisines*/*index*.html.erb网页模板文件，添加显示逻辑如下：

<div class="*cuisines*">

<div class="container">

<% @*cuisines*.each do |*c*| %>

<div class="*cuisine*">

<h2><%= *c*.*region* %></h2>

<img class="arrow" src="#" width="80" height="40">

<p><%= link\_to "...", *cuisine*\_path(*c*) %></p>

<%= image\_tag *c*.*image* %>

</div>

<% end %>

</div>

</div>

* 新建app/views/*cuisines*/*show*.html.erb网页模板文件，添加显示逻辑如下：

<div class="main">

<h1 class="*cuisine*"><%= @*cuisine*.*region* %>...</h1>

<div class="container">

<div class="*recipes*">

<% @*recipes*.each do |*r*|%>

<div class="*recipe*">

<h2 class="*recipe*-*name*"><%= *r*.*name* %></h2>

<img class="arrow" src="#" width="80" height="40">

<p class="*recipe*-link">

<%= link\_to "...", *recipe*\_path(*r*) %>

</p>

<%= image\_tag *r*.*image* %>

</div>

<% end %>

</div>

</div>

</div>

* 新建app/views/*recipes*/*show*.html.erb网页模板文件，添加显示逻辑如下：

<div class="*recipe*-*show*">

<div class="container">

<h2><%= @*recipe*.*name* %></h2>

<img class="arrow" src="#" width="80" height="40">

<div class="img\_container">

<%= image\_tag @*recipe*.*image* %>

</div>

<div class="*recipe*-info">

<p class="*ingredients*"><%= @*recipe*.*ingredients* %></p>

<p class="*instructions*"><%= @*recipe*.*instructions* %></p>

</div>

<% if *current\_user* && *current\_user*.*editor*? %>

<p class="*recipe*-*edit*">

<%= link\_to "...", *edit\_recipe*\_path(@*recipe*.id) %>

</p>

<% end %>

<% if *current\_user* && *current\_user*.*admin*? %>

<p class="*recipe*-*delete*">

<%= link\_to "...", *recipe*\_path(@*recipe*),

method: "delete" %>

</p>

<% end %>

</div>

</div>

* 新建app/views/*recipes*/*edit*.html.erb网页模板文件，添加显示逻辑如下：

<div class="*recipe*-*edit*">

<div class="container">

<div class="img\_container">

<%= image\_tag @*recipe*.*image* %>

</div>

<%= form\_for(@*recipe*) do |*f*| %>

<%= *f*.text\_field :*name* %>

<%= *f*.text\_area :*ingredients* %>

<%= *f*.text\_area :*instructions* %>

<%= *f*.submit "Update", :class => "form-btn" %>

<% end %>

</div>

</div>

* 新建app/views/*users*/*new*.html.erb网页模板文件，添加显示逻辑如下：

<div class="*signup*">

<div class="container">

<%= form\_for :*user*, url: '/*users*' do |*f*| %>

<%= *f*.text\_field :*fname*, :placeholder => "*First Name*" %>

<%= *f*.text\_field :*lname*, :placeholder => "*Last Name*" %>

<%= *f*.email\_field :*email*, :placeholder => "*Email*" %>

<%= *f*.password\_field :*password*,

:placeholder => "*Password*" %>

<%= *f*.submit "*Signup*", :class => "form-btn" %>

<% end %>

</div>

</div>

* 新建app/views/*sessions*/*new*.html.erb网页模板文件，添加显示逻辑如下：

<div class="*login*">

<div class="container">

<%= form\_for(:*session*, url: *login*\_path) do |*f*| %>

<%= *f*.email\_field :*email*, :placeholder => "*Email*" %>

<%= *f*.password\_field :*password*,

:placeholder => "*Password*" %>

<%= *f*.submit "*Login*", :class => "form-btn" %>

<% end %>

</div>

</div>

* 编辑app/assets/stylesheets/application.css样式表文件如下：

[见cuisineapp.css文件]

**六、调试**

* 输入rails console进入控制台
* 输入*User*.find\_by(*fname*: "*Mateo*").*editor*? 验证*Mateo*用户为*editor*
* 输入*User*.find\_by(*fname*: "*Freida*").*admin*? 验证*Freida*用户为*admin*
* 输入exit退出控制台
* 输入rails server启动服务器
* 在浏览器地址栏输入 [http://localhost:3000/*login*](http://localhost:3000/login) 并以普通用户*Julian*（电邮地址：*julian*@*email*.com；密码：*Julian1*）登陆后查看效果
* 在浏览器地址栏输入 [http://localhost:3000/*login*](http://localhost:3000/login) 并以*editor*用户*Mateo*（电邮地址：*mateo*@*email*.com；密码：*Mateo1*）登陆后查看效果
* 在浏览器地址栏输入 [http://localhost:3000/*login*](http://localhost:3000/login) 并以*admin*用户*Freida*（电邮地址：*freida*@*email*.com；密码：*Freida1*）登陆后查看效果