RUBY ON RAILS

SST Hackathon Training Instruction guide

Contents

[1. Introduction 2](#_Toc371669231)

[2. An Introduction to Nitrous.io 3](#_Toc371669232)

[3. An Introduction to Heroku 6](#_Toc371669233)

[4. Code Monkey Events 7](#_Toc371669234)

[5. Setting Up 11](#_Toc371669235)

[6. Create an “About Us” Page 13](#_Toc371669236)

[7. Adding Page Header and Footer 14](#_Toc371669237)

[8. Allowing Users To Signup 17](#_Toc371669238)

[9. Adding Name and Avatar to the User 20](#_Toc371669239)

[10. Creating Events 22](#_Toc371669240)

[11. Beautifying Events List 26](#_Toc371669241)

[12. Creating Photos 30](#_Toc371669242)

[13. Beautifying individual event page 37](#_Toc371669243)

[14. Allowing Enthusiasts to Follow an Event 40](#_Toc371669244)

[15. Creating Comment 44](#_Toc371669245)

[16. Beautifying individual photo page 48](#_Toc371669246)

[17. Sending Email Notifications 51](#_Toc371669247)

[18. Publishing to heroku 53](#_Toc371669248)

# Introduction

* In this course, we will learn how to build ***a web application***
* You need to have background knowledge in ***html and css*** for this course
* This document is the **instruction guide**, so it will not contain explanations
* We will use ***Nitrous.io*** as the development environment
* The programming language that we will use is called ***Ruby***
* We don’t want to build the application from scratch. So, we will use ***Ruby on Rails (Rails)*** framework. It gives us tools to build web apps without writing a lot of codes

## Resources

* If you are new to programming or Ruby, try out [http://tryruby.org](http://tryruby.org/)
* If you want to learn HTML & CSS, check out <http://freecourses.tutsplus.com/30-days-to-learn-html-and-css/>
* You can experiment HTML & CSS online at <http://cssdesk.com/>

# An Introduction to Nitrous.io

* Nitrous.io is an online development environment.
* It allows us to write programs and run (deploy) them from the web browser.

## Sign up

* Visit [www.nitrous.io](http://www.nitrous.io) and sign up for a free account.



Figure 1: Nitrous.io

## Create a new box

* Log in to Nitrous.io
* Create a new Development Box or “box” as shown in Figure 2.
* Choose “Ruby/Rails” box.
* Pick “Southeast Asia” as the region.
* Enter a unique name for the box.
* Click “Create Box” (we need to wait a bit, while Nitous.io provisions our box)

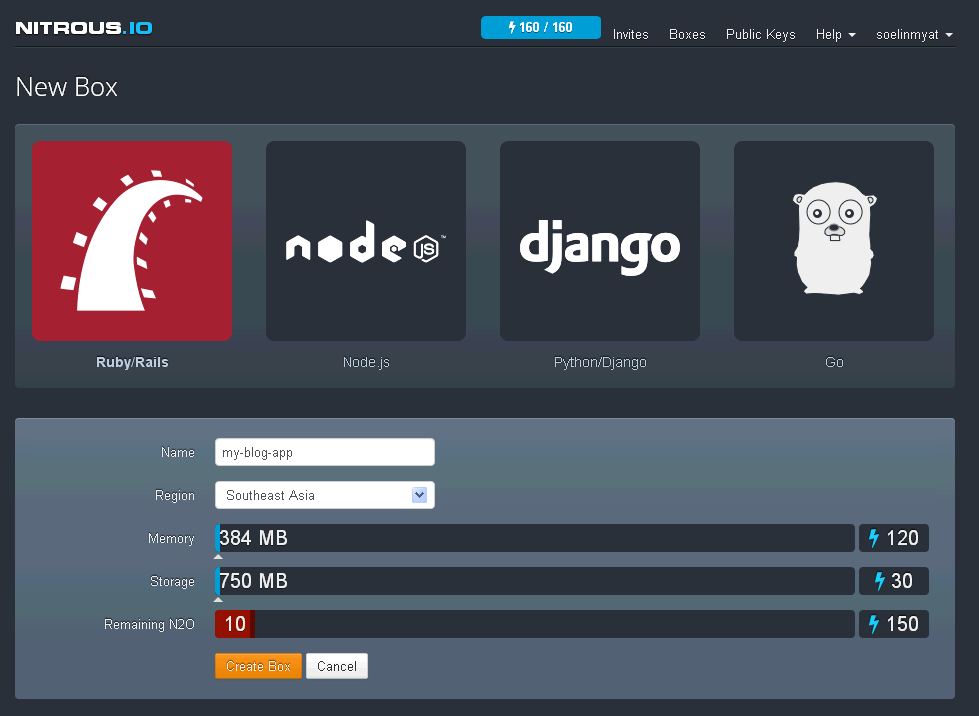


Figure 2: Create a new box in Nitrous.io

After creating the box, we will see the page as shown in Figure 3. This page is called Integrated Development Environment (IDE).

IDE has three main parts. We have labeled them A, B and C.

* Part A shows the ***folder structure*** of our box.
* Part B is the ***code editor*** where we write codes.
* Part C is the ***console***

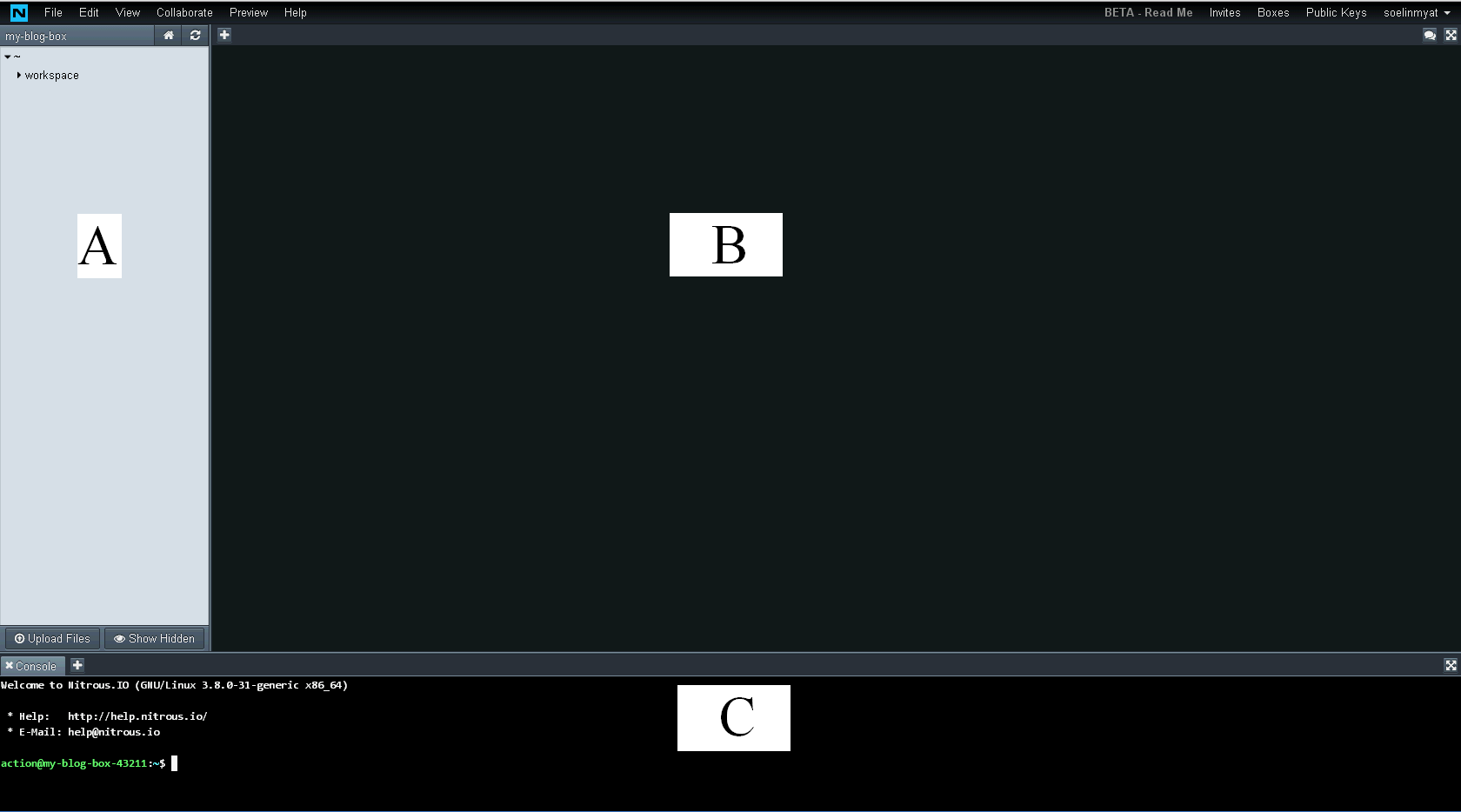


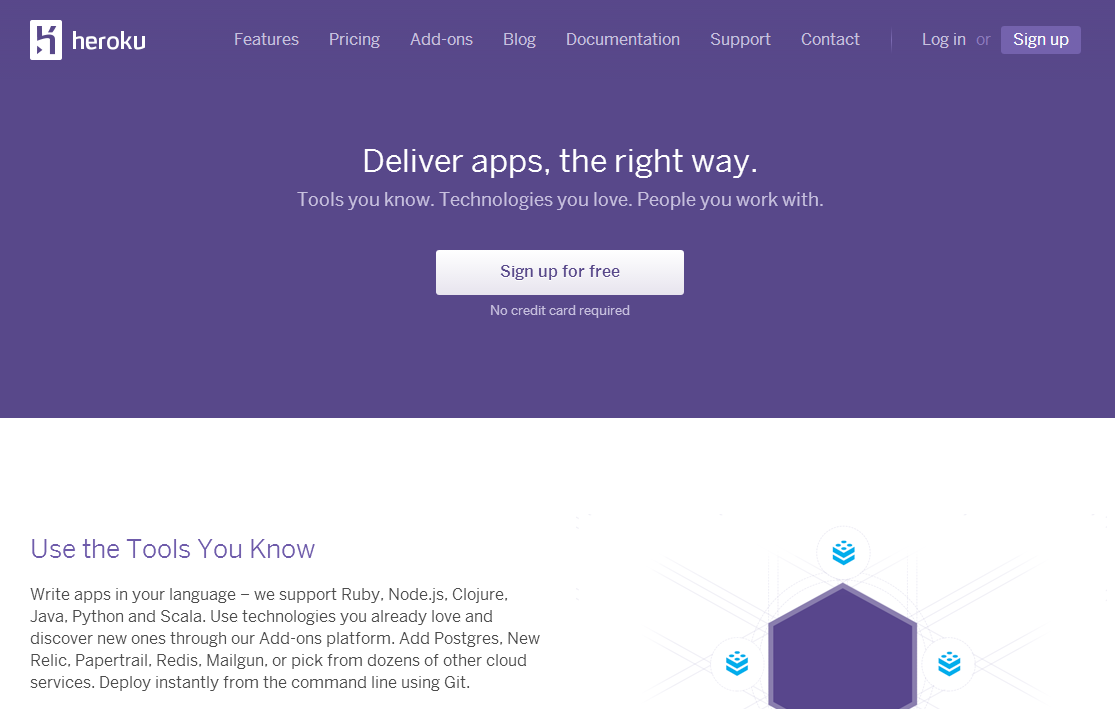
Figure 3: Nitrous.io Development Environment

# An Introduction to Heroku

* Heroku is a service which allows us to host web apps
* They offer free tier, so we can host small apps for free
* In this course, we will use Heroku to host our app

## Sign Up

* Visit <https://www.heroku.com/> and sign up for a free account.
* We will explain more about how to use it in a later section



# Code Monkey Events

## What is it?

* a website where coding enthusiasts can follow and share photos from coding events

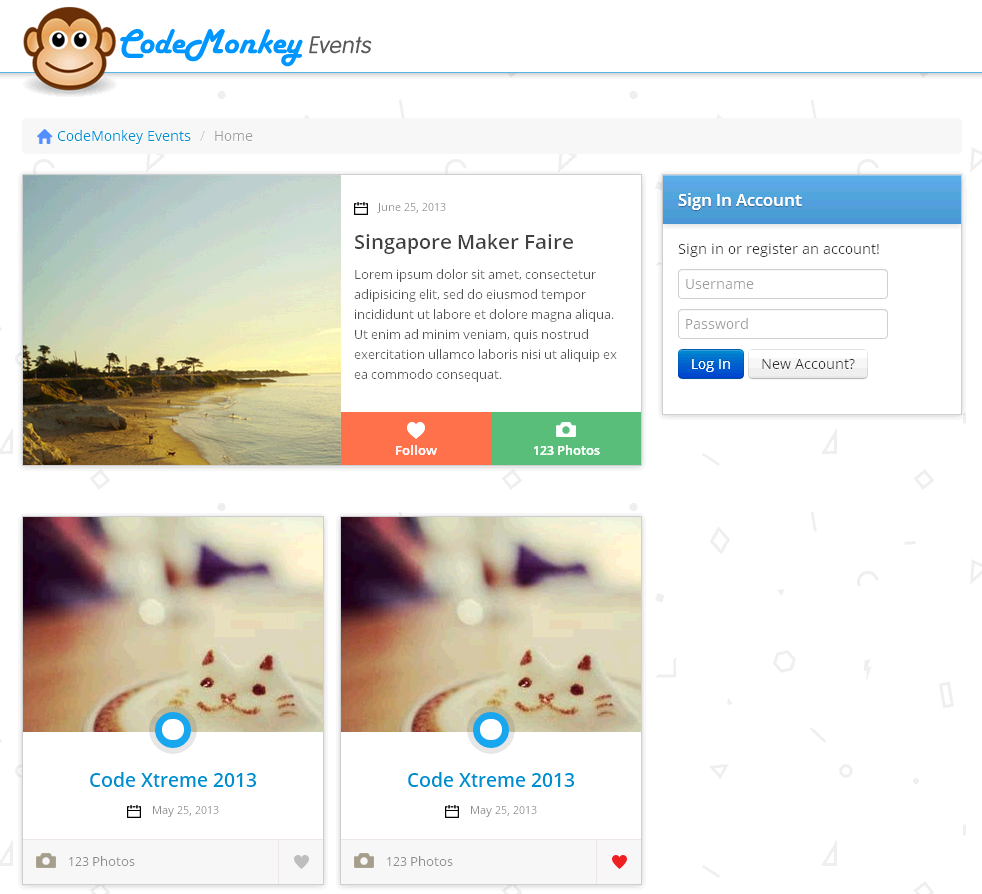


Figure 4: Home Page of Code Monkey Events

There are two types of users for ***Code Monkey Events***

* Organizers
  + create and update information about coding events
* Enthusiasts
  + upload photos to events that they have attended
  + “follow” events so that they get an email notification for new photos
  + Interact with others by commenting on photos

## How can coding enthusiasts use Code Monkey?

### Following events

* create a free account for ***Code Monkey Events***
* start following events by clicking on the “Follow” button
* Figure 5 shows the home page for the user, Guan Hong
* the latest events that he is following are shown under “My Events”

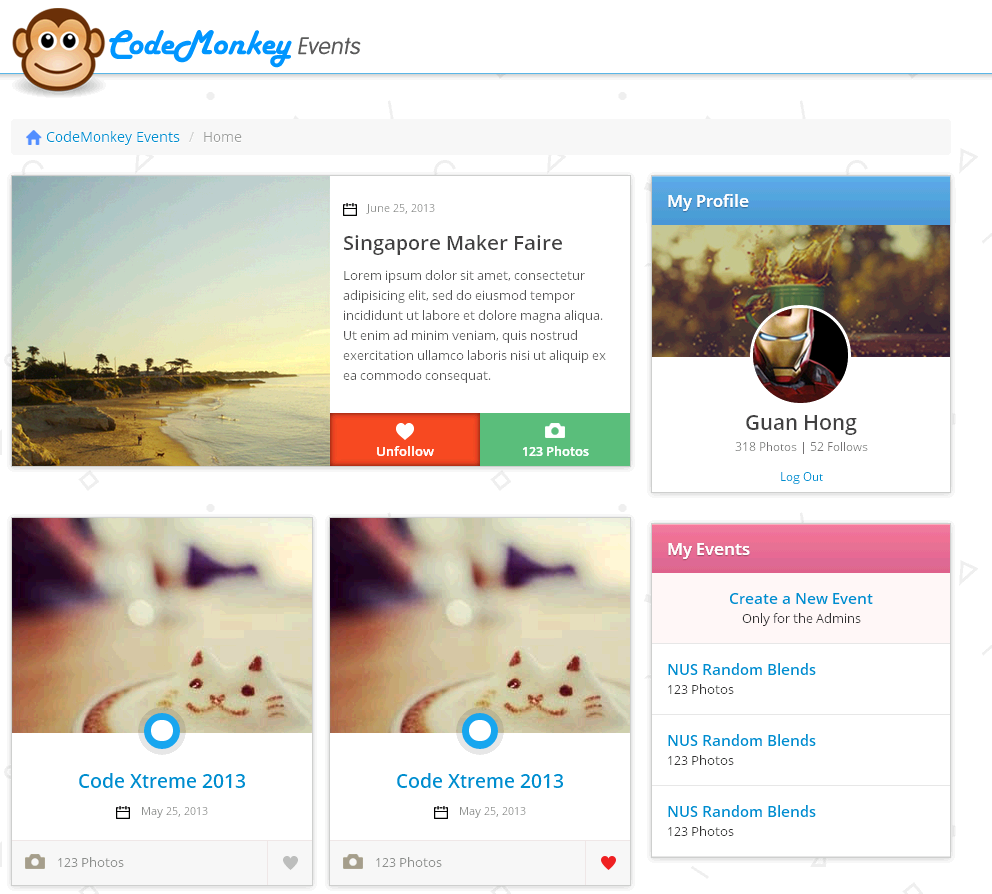


Figure 5: Home page for Guan Hong

### Browsing and uploading photos

* browse through photos for a particular event
* upload the photos
* Figure 6 shows the photo gallery view of an event

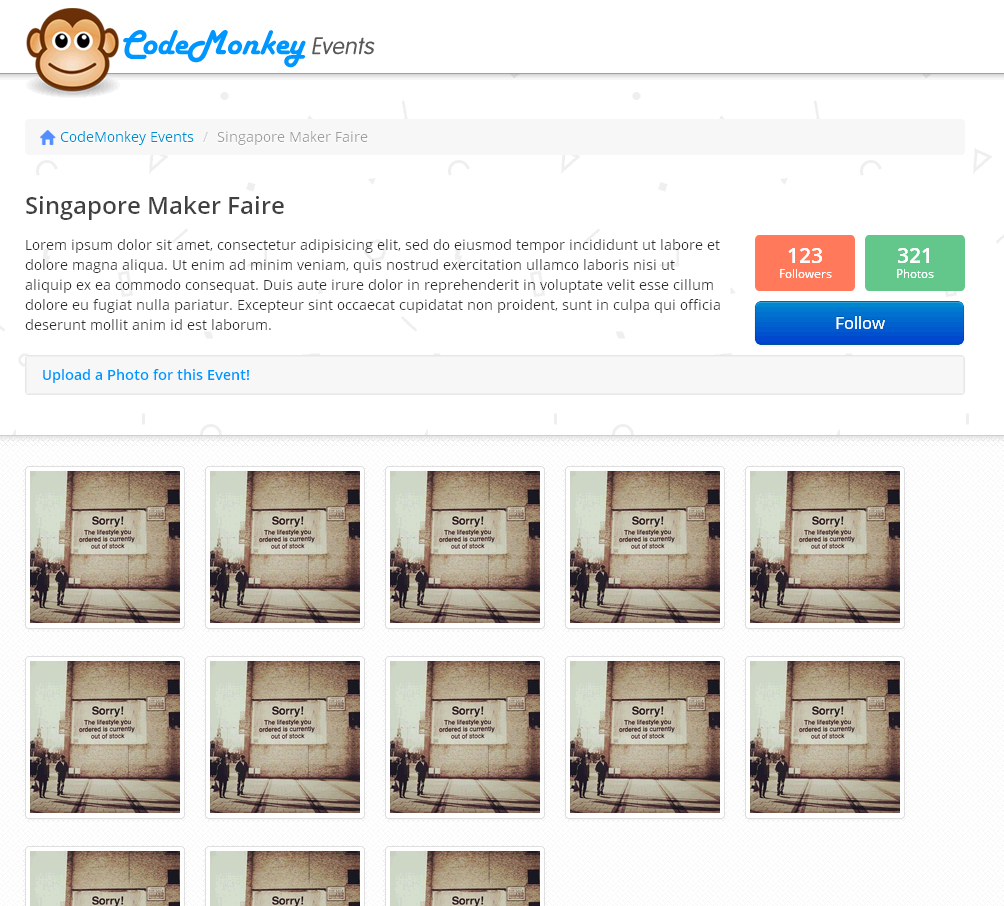


Figure 6: Photo gallery of an event, Singapore Maker Fair

### Commenting on photos

* Comment on each photo.
* Users are required to log in
* Figure 7 shows a single-photo page with comments.

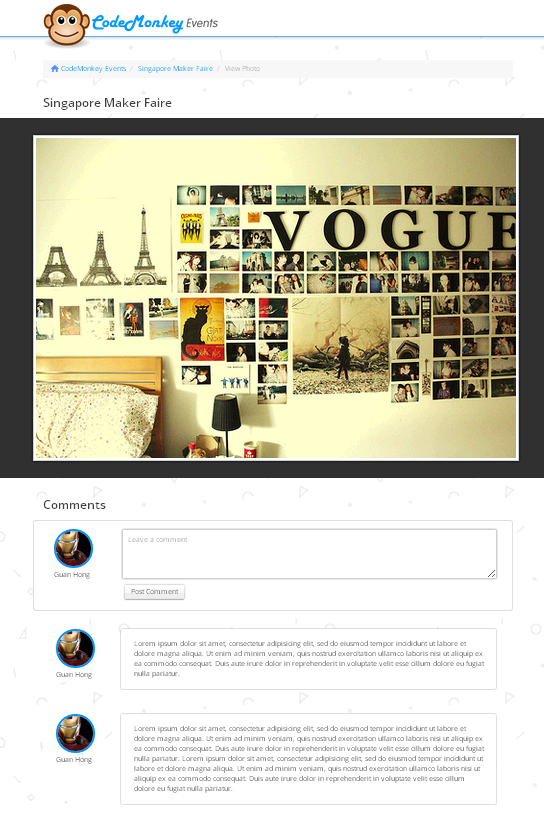


Figure 7: A single-photo page with comments

# Setting Up

* We will do necessary setup for ***Code Monkey Events*** in this section.

## Learning Points

* Creating a new Rails application
* Starting the server from console
* Previewing in Nitrous.io
* Installing additional gems
* Adding css files
* Adding images

## Instructions

Note: ***Italic words*** are commands to be entered into **console** in the Nitrous.io IDE.

* Create a new project called “codemonkey”

***rails new codemonkey***

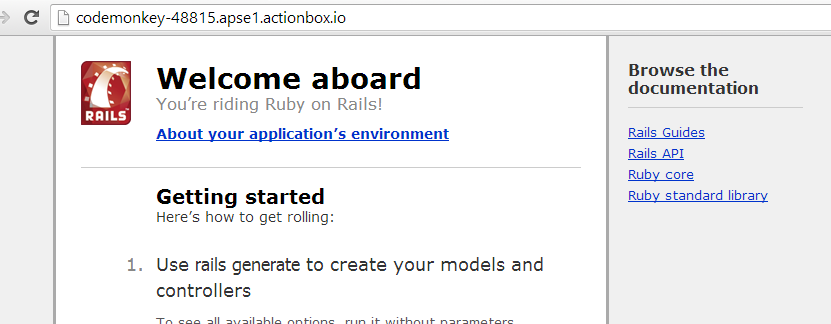
* Go into the folder “*codemonkey*”

***cd codemonkey***

* Run the server

***rails server***

* Preview the site
  + Click on “Preview” at the top navigation bar in the Nitrous.io IDE and choose “Port 3000”
  + You will see the following page. Congrats! Your first Rails a is running.



* Stop the server - Press Ctrl+C in the console
* Install ***simple\_form*** gem
  + Add the following line into “Gemfile” file

*gem ‘simple\_form’*

* + Download the gem

***bundle install***

* + Install the gem

***rails generate simple\_form:install***

Note: A gem gives us special “abilities” or “power” to develop complex functionalities with ease. E.g. **simple\_form** gem makes creating web forms easy for us.

* Add “Bootstrap”
  + Download “Bootstrap” from <http://getbootstrap.com/2.3.2/>
  + Extract the zip file
  + Upload bootstrap.min.js to “app/assets/javascripts” folder
    - Right click “app/assets/javascripts” folder in the folder structure in Nitrous.io
    - Click “Upload Files to…”
    - Drag and drop the file and click “Upload”
  + Upload bootstrap.min.css and bootstrap-responsive.min.css to “app/assets/stylesheets” folder

Note: Bootstrap is a CSS library which helps us with beautifying the look of the app.

* Add images
  + We provide a few files together with this document
  + Upload the given images in the folder “codemonkey/images” to “app/assets/images” folder
* Add custom css file
  + Move the given “style.css.scss” from in the folder “codemonkey/stylesheets” to the folder “app/assets/stylesheets”

Note: We will be using these images and css files in developing **Code Monkey Events**. Since the focus of this course is on Rails, we will not be explaining how html and css works.

## Resources

* Refer to <https://github.com/plataformatec/simple_form> for more info on Simple form.
* Refer to <http://getbootstrap.com/2.3.2/> for more info on Bootstrap

# Create an “About Us” Page

* We will create an “About us” page for Code Monkey.

## Learning points

* Generating a controller
* Customizing the view

## Instructions

* Create a controller “InfoController” with a function “index”

***rails generate controller Info index***

A few files will be created. The important files are

“info\_controller.rb” under “app/controllers” folder

“index.html.erb” under “app/views/info” folder

* *Open “index.html.erb” under “app/views/info” folder and replace the existing codes with the following*

<div class=**"container"**>

<h2>**About us**</h2>

<p>

**Code Monkey is a place for coding enthusiasts to meet, interact and share photos of the coding events.**

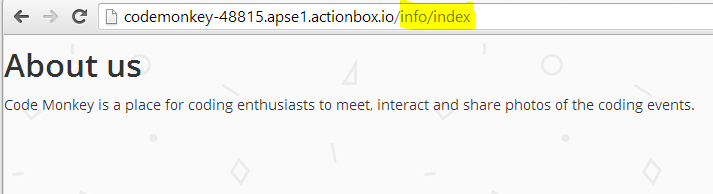
</p>

</div>

* Run the server

***rails server***

* Preview the page
  + Click on “Preview” at the top navigation bar in the Nitrous.io IDE and choose “Port 3000”
  + Add “info/index” to the end of the url and click Enter. You will see the following. You have created “About us” page for **Code Monkey Events**!



# Adding Page Header and Footer

* We will add header and footer to the page

## Learning Points

* Customizing the layout

## Instructions

* In “application.html.erb” file under “app/views/layouts”, replace

**<%=** yield **%>**

with

**<%=** render "layouts/header" **%>**

**<%=** render "layouts/notice" **%>**

**<%=** yield **%>**

**<%=** render "layouts/footer" **%>**

Note: Changing “application.html.erb” file affects all the pages on the app by default. E.g. We have add “header”, “notice” and “footer” sections above. Every page on the app will now have those sections by default.

* Create a new file “\_header.html.erb” under “app/views/layouts” and add the following code
  + To create a new file, right click on “app/views/layouts” folder, choose “New File” and enter the new file name

<!-- header -->

<div class=**"container-fluid topHeader"**>

<div class=**"row-fluid"**>

<div class=**"container"**>

<div class=**"span12"**>

<a href=**"/"** class=**"logo"**>

**<%= image\_tag('sitelogo.png', :class => "logo") %>** </a>

</div>

</div>

</div>

</div>

Note: The erb files starting with “\_” such as “\_header.html.erb” are called partial. You can render (load) partials by using “render” as we have done so in “application.html.erb” above.

* Create a new file “\_footer.html.erb” under “app/views/layouts” and add the following code

<!-- footer -->

<div class=**"container-fluid footer"**>

<div class=**"row-fluid"**>

<div class=**"container"**>

<div class=**"span12"**><a href=**"/info/index"**>**© Copyright 2013 CodeMonkey Events.**</a></div>

</div>

</div>

</div>

* Create a new file “\_notice.html.erb” under “app/views/layouts” and add the following code

<!-- alert -->

**<%** **if** flash[:alert] **%>**

<div class=**"container"**>

<div class=**"row"**>

<div class=**"span12"**>

<p class=**"alert alert-error"**>**<%=** flash[:alert] **%>**</p>

</div>

</div>

</div>

**<%** **end** **%>**

<!-- notice -->

**<%** **if** flash[:notice] **%>**

<div class=**"container"**>

<div class=**"row"**>

<div class=**"span12"**>

<p class=**"alert"**>**<%=** flash[:notice] **%>**</p>

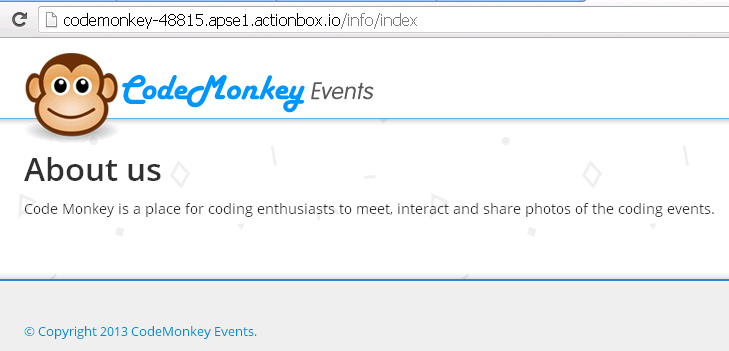
</div>

</div>

</div>

**<%** **end** **%>**

* You have successfully added the header and footer for every page on **Code Monkey Events**. Preview the “About us” page again.



# Allowing Users To Signup

* We will allow users to sign up

## Learning points

* Using “devise” gem for user signup and login
* Using “rolify” gem to assign roles to users
* Using “cancan” gem for controlling user actions based on their roles
* Migrating database
* Adding root url

## Instructions

* Add the following to GEMFILE

gem 'devise'

gem 'cancan'

gem 'rolify'

* Download gems

***bundle install***

* Install devise

***rails generate devise:install***

* Create User

***rails generate devise User***

* Create devise views

***rails generate devise:views***

* Create Ability

***rails generate cancan:ability***

* Create Role

***rails generate rolify:role***

* Update database

***rake db:migrate***

Note: The “devise” gem helps us with authentication of users such as sign up, sing in and sign out. Check <https://github.com/plataformatec/devise> for more info.

Note: The “rolify” gem helps us with assigning roles to users. Check <https://github.com/EppO/rolify> for more info.

Note: The “cancan” gem helps us with restricting the actions that a user can do in the app. Check <https://github.com/ryanb/cancan> for more info.

* Add the following code to the file “config/environments/development.rb”

config**.**action\_mailer**.**default\_url\_options **=** **{** :host **=>** 'localhost:3000' **}**

* Set “About us” page as root url
  + In “routes.rb” file under “config” folder, add the following code

root "info#index"

* Wrap the codes in “new.html.erb” file under “app/views/devise/sessions” with a “div” block and give it a class “container”

<div class=**"container"**>

**.**

**.**

**. existing codes**

</div>

* Wrap the codes in “new.html.erb” file under “app/views/devise/registrations” with a “div” block and give it a class “container”
* Wrap the codes in “edit.html.erb” file under “app/views/devise/registrations” with a “div” block and give it a class “container”
* Add the following code to the end of the file *“index.html.erb” under “app/views/info” folder*

**<%** if user\_signed\_in? **%>**

**Signed in as <%=** current\_user.email **%>.**<br/>

**<%=** link\_to 'Edit', edit\_user\_registration\_path **%>**<br/>

**<%=** link\_to 'Sign out', destroy\_user\_session\_path, :method => :delete **%>**

**<%** else **%>**

**<%=** link\_to 'Register', new\_user\_registration\_path **%> or <%=** link\_to 'Sign in', new\_user\_session\_path **%>**

**<%** end **%>**

* Add the role of new user as “enthusiast” by default
  + Add the following code to “user.rb” file under “app/models” folder
  + Make sure to add the code before the last “end” statement

after\_create :assign\_default\_role

private

**def** **assign\_default\_role**

add\_role**(**:enthusiast**)** if **self.**roles**.**blank?

**end**

* Run the server and review “About us” page again
  + You can now sign up, log in and log out! Try it out.
* Stop the server by pressing Ctrl+C in the console.

# Adding Name and Avatar to the User

* We will now add name and avatar to a user. An avatar is an image.

## Learning points

* Using “paperclip” gem for uploading photo
* Adding additional information to devise sign up process

Note: The “paperclip” gem helps us with image uploading. Check <https://github.com/thoughtbot/paperclip> for more info.

## Instructions

* Add the following to GEMFILE

gem 'paperclip'

* Download the gem

bundle install

* Add name

rails generate migration AddNameToUsers name:string

* Add avatar

rails generate paperclip User avatar

* Update database

rake db:migrate

* Add the following code to “user.rb” under “app/models” folder before the statement “private”

validates\_presence\_of :avatar**,** :name

has\_attached\_file :avatar**,** styles: **{**

thumb: '70x70#'**,**

medium: '95x'

**}**

Note: “validates\_presence\_of :avatar**,** :name” makes it compulsory for all users to have name and avatar. If they don’t add name or upload avatar, they will not be able to successfully create an account.

* Add following code to “application\_controller.rb” under “app/controllers” folder before the last “end” statement

before\_filter :configure\_permitted\_parameters**,** **if:** :devise\_controller**?**

protected

**def** **configure\_permitted\_parameters**

devise\_parameter\_sanitizer**.**for**(**:sign\_up**)** **<<** **[**:avatar**,** :name**]**

devise\_parameter\_sanitizer**.**for**(**:account\_update**)** **<<** **[**:avatar**,** :name**]**

**end**

* In “new.html.erb” under “app/views/devise/registrations” folder, add

**<%=** f.input :name, :required => true, :autofocus => true **%>**

**<%=** f.input :avatar, :required => true, :autofocus => true **%>**

after

**<%=** f.input :password\_confirmation, :required => true **%>**

* In “edit.html.erb” under “app/views/devise/registrations” folder, add

**<%=** f.input :name, :required => true, :autofocus => true **%>**

**<%=** f.input :avatar, :required => true, :autofocus => true **%>**

after

**<%=** f.input :email, :required => true **%>**

* Run the server and preview the site
  + You can now add name and upload an image as your avatar when you sign up
* When we host our app on Heroku, we will need to store images using Amazon S3 service. We will need to add an additional gem and settings. We will go through how to do it during the training.

# Creating Events

* We will create pages related to events

## Learning points

* Using ***generate scaffold***command
* Displaying an image using “image\_tag”

## Instructions

* Scaffold Event

***rails generate scaffold Event user\_id:integer title:string date:date description:text main\_image:attachment***

* Empty “scaffolds.css.scss” file under “app/assets/stylesheets” folder
* Update database

***rake db:migrate***

* Set the root url to the index page of events
  + In “routes.rb” file under “config” folder, replace

root ‘info#index’ with

***root 'events#index'***

* In “routes.rb” file under “config” folder, add

resources 'events'

* *In “event.rb” file under “app/models” folder, add*

belongs\_to: :user

validates\_presence\_of :title**,** :date**,** :main\_image

has\_attached\_file :main\_image**,** styles: **{**

thumb: '160x160>'**,**

square: '200x200#'**,**

medium: '300x300>'

**}**

* *In “user.rb” file under “app/models” folder, add the following before “private” statement*

has\_many :events

* Wrap the codes in “edit.html.erb” file under “app/views/events” with a “div” block and give it a class “container”
* Wrap the codes in “new.html.erb” file under “app/views/events” with a “div” block and give it a class “container”
* Wrap the codes in “show.html.erb” file under “app/views/events” with a “div” block and give it a class “container”
* *In “\_form.html.erb” file in the “app/views/events” folder, replace*

**<**%= f.input :user\_id %>

*with*

**<**%= f.input :user\_id, :as =**>** :hidden**,** :input\_html **=>** **{** :value **=>** current\_user**.**id **}** %>

* *In “events\_controller.rb” file in the “app/controllers” folder, add the following code at the second line*

before\_filter :authenticate\_user**!,** except: **[**:index**,** :show**]**

before\_filter :new\_event**,** :only **=>** **[**:new**,** :create**]**

load\_and\_authorize\_resource

* *In “events\_controller.rb” file in the “app/controllers” folder, add the following function after “private” statement*

**def** **new\_event**

@event **=** Event**.**new

@event**.**user\_id **=** current\_user**.**id

**end**

* *In “show.html.erb” file in the “app/views/events” folder, replace*

**<**%= @event.main\_image %>

with

**<%=** image\_tag @event.main\_image(:medium) **%>**

* *In “index.html.erb” file under “app/views/events” folder, replace*

<td>**<%=** event.main\_image **%>**</td>

with

<td>**<%=** image\_tag event.main\_image(:thumb) **%>**</td>

* *In “ability.rb” file in the “app/models” folder, add the following code inside “initialize” method as shown below*

**def** **initialize(**user**)**

user **||=** User**.**new # guest user (not logged in)

can **[**:show**,** :index**],** Event

**if** user**.**has\_role? :organizer

can :manage**,** Event**,** :user\_id **=>** user**.**id

**end**

**end**

Note: “ability.rb” is used for “cancan” gem. It specifies the actions that a particular user can do within the app. For example, in the above “ability.rb” file, it is specified that everyone can see “show” and “index” pages of Event but only users with role “organizer” can manage (create, edit, delete) events.

* *Add “organizer” role to an existing account*

***rails console***

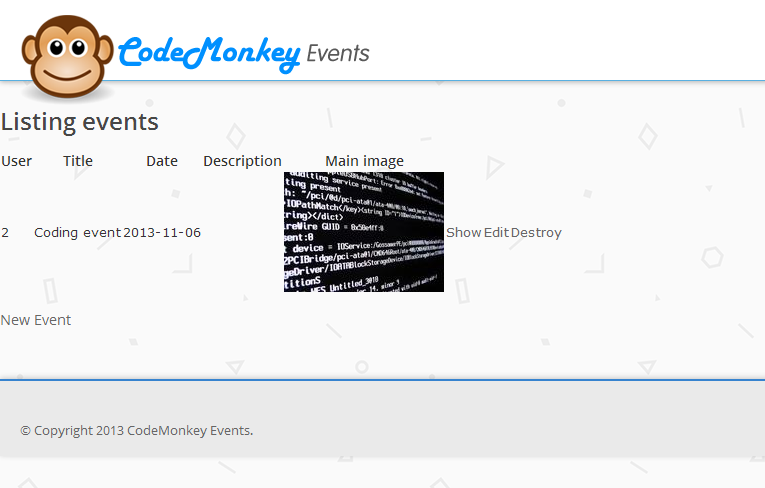
***user = User.find\_by\_email(“david@gmail.com”)***

***user.add\_role “organizer”***

***exit***

*We have added “organizer” role to the user with email* [*david@gmail.com*](mailto:david@gmail.com)*. Replace it with the email address of the user to whom you want to add “organizer” role.*

* Run server
* Preview and log in with the organizer user
  + You will be able to create and views events now
  + If you log in as the enthusiast user, you will get an error when you try to create events



# Beautifying Events List

* We will beautify the index page of Event. It will be the main page of ***Code Monkey Events*** as shown in the image below.



## Instructions

* Replace the code in “index.html.erb” under “app/views/events” with the following.

**<**div **class=**"container"**>**

**<div** **class=**"row"**>**

**<!--** **events** column **-->**

**<**div **class=**"span8"**>**

**<**% if @events**.length** **>** 0 %>

<div class="row">

**<**div **class=**"span8"**>**

**<!--** **featured** event **-->**

**<**%= render "events/featured\_preview", :event =**>** @events**[**0**]** %>

<!-- other events -->

**<**div id**=**"eventsWrapper"**>**

**<**ul **class=**"row otherEvents"**>**

**<**% 1.upto(@events.length-1) **{** **|i|** %>

<li class="span4 eventContainer">

**<**%= render "events/normal\_preview", :event =**>** @events**[**i**]** %>

</li>

**<**% } %>

</ul>

**<**/div>

</div**>**

**<**/div>

<% end %>

</div**>**

**<!--** user information column **-->**

**<**div **class=**"span4"**>**

**<**%= render "events/\_profile", :user => current\_user %>

**<**/div>

</div**>**

**<**/div>

* Create a new file “\_featured\_preview.html.erb” in “app/views/events” folder and place the following codes.

<div class="featuredEvent">

<div class="span4 featuredEventImageContainer">

<img src="**<%=** event.main\_image.url **%>"**>

</div>

<div class=**"span4 featuredEventContent"**>

<div class=**"featuredEventInner"**>

<div class=**"eventDate"**>

**<%= image\_tag('calendar.png') %>**

**<%=** event.date.to\_formatted\_s(:long) **%>**

</div>

<h1>**<%=** event.title **%>**</h1>

<p>**<%=** truncate(event.description, :length => 300) **%>**</p>

</div>

<a href=**"#"** id=**"btn-featuredFollow"**>

<div class=**"span2 featuredEventButton featuredfollowButton"**>

<p class=**"btn-label"**>**Follow**</p>

</div>

</a>

<a href=**"<%=** event\_url(event) **%>"**>

<div class=**"span2 featuredEventButton featuredPhotoButton"**>

<p class=**"btn-label"**>**0 Photos**</p>

</div>

</a>

</div>

</div>

* Create a new file “\_normal\_preview.html.erb” in “app/views/events” folder and place the following codes.

<div class="eventPicture">

<img src="**<%=** event.main\_image.url **%>"**>

</div>

<div class=**"eventContDec"**>

<div class=**"eventContDecInner"**></div>

</div>

<div class=**"eventContent"**>

<h1>

<a href=**"<%=** event\_url(event) **%>"**>**<%=** event.title **%>**</a>

</h1>

<div class=**"eventDate"**>

**<%= image\_tag('calendar.png') %>**

**<%=** event.date.to\_formatted\_s(:long) **%>**

</div>

</div>

<div class=**"eventFooter"**>

<div class=**"eventTotalPhotos"**>**0 Photos**</div>

<div id=**"btn-follow"** class=**"eventFollowButtonContainer"**>

</div>

</div>

* Create a new file “\_profile.html.erb” in “app/views/events” folder and place the following codes.

<div class="profileHeader sideContainerHeader">

**My Profile**

</div>

<div class="profileBackground">

<div class="profilePicture">

<img src="**<%=** current\_user.avatar.url(:thumb) **%>"** class=**"img-circle"**>

</div>

</div>

<div class=**"profileInfo"**>

<h1>**<%=** current\_user.name **%>**</h1>

<p>**0 Photo | 0 Follows**</p>

<small class=**"profileInfoLinks"**>**<%=** link\_to("Log Out", destroy\_user\_session\_path, :method => :delete) **%>**</small>

</div>

* Create a new file “\_login.html.erb” in “app/views/events” folder and place the following codes.

<div class="profileHeader sideContainerHeader">

**Sign In Account**

</div>

<div class="loginInfo">

<p>**Sign in or register an account!**</p>

**<%=** simple\_form\_for(:user, :url => user\_session\_path) do |f| **%>**

<div class=**"form-inputs"**>

**<%=** f.input :email, :required => false, :autofocus => true **%>**

**<%=** f.input :password, :required => false **%>**

**<%=** f.input :remember\_me, :as => :boolean, :label => false, :inline\_label => true **%>**

**<%=** f.button :submit, "Sign in", :class => "btn btn-primary" **%>**

**<%=** link\_to "Sign up", new\_user\_registration\_path, :class => "btn" **%>**<br />

</div>

**<%** end **%>**

</div>

* Run the server and preview ☺

# Creating Photos

* We will create pages related to photos

### Learning point

* Using nested routes
* One to many relation

### Instructions

* Scaffold Photo

***rails generate scaffold Photo user\_id:integer event\_id:integer name:string image:attachment***

Enter “n” if there is any question regarding “Overwrite”

* Update database

***rake db:migrate***

* In the “routes.rb” under “config” folder, replace

resources :events

With

resources :events **do**

resources :photos

**end**

* Add the following line to photo.rb under “app/models” folder

belongs\_to :event

belongs\_to :user

* Add the following line to event.rb under “app/ models” folder

has\_many :photos, :dependent => :destroy

* Add the following line to user.rb under “app/ models” folder

has\_many :photos

* Add following codes to “photo.rb” file under “app/views/” folder

has\_attached\_file :image**,** styles: **{**

thumb: '160x160>'**,**

square: '200x200#'**,**

medium: '300x300>'

**}**

* *In “photos\_controller.rb” file in the “app/controllers” folder, add the following code at the second line*

before\_filter :authenticate\_user**!,** except: **[**:index**,** :show**]**

before\_filter :new\_photo**,** :only **=>** **[**:new**,** :create**]**

load\_and\_authorize\_resource

* *In “photos\_controller.rb” file in the “app/controllers” folder, add the following function after “private” statement*

**def** **new\_photo**

@photo **=** Photo**.**new

@photo**.**user\_id **=** current\_user**.**id

**end**

* In the “photos\_controller”, add the following code to the beginning of “index”, “new”, “edit”, “show” and “update” functions

@event **=** Event**.**find**(**params**[**:event\_id**])**

* In the “photos\_controller” file, replace “create” function with the following code

**def** **create**

@event **=** Event**.**find**(**params**[**:event\_id**])**

@photo **=** @event**.**photos**.**new**(**photo\_params**)**

respond\_to **do** **|**format**|**

**if** @photo**.**save

format**.**html **{** redirect\_to event\_photo\_path**(**@event**,** @photo**),** notice: 'Photo was successfully created.' **}**

format**.**json **{** render action: 'show'**,** status: :created**,** location: @photo **}**

**else**

format**.**html **{** render action: 'new' **}**

format**.**json **{** render json: @photo**.**errors**,** status: :unprocessable\_entity **}**

**end**

**end**

**end**

* In “photos\_controller”, in “index” function, replace

@photos = Photo.all

with

@photos = @event.photos.all

* In the “photo\_controller”, in the “update” function, replace

format**.**html **{** redirect\_to @photo**,** notice: 'Photo was successfully updated.' **}**

with

format**.**html **{** redirect\_to event\_photo\_path**(**@event**,** @photo**),** notice: 'Photo was successfully updated.' **}**

* In the “\_form.html.erb” file under “app/views/photos” folder, replace

**<%=** simple\_form\_for(@photo) **do** |f| **%>**

with

**<%=** simple\_form\_for([@event, @photo]) **do** |f| **%>**

* In the “\_form.html.erb” file under “app/views/photos” folder, replace

**<%=** f.input :user\_id **%>**

with

**<%=** f.input :user\_id, :**as** => :hidden, :input\_html => { :value => current\_user.id } **%>**

* In the “\_form.html.erb” file under “app/views/photos” folder, replace

**<%= f.input :event\_id %>**

with

**<%=** f.input :event\_id, :**as** => :hidden, :input\_html => { :value => @event.id } **%>**

* In the “new.html.erb” file under “app/views/photos”, replace

**<%=** link\_to 'Back', photos\_path %>

with

**<%=** link\_to 'Back', event\_photos\_path(@event) %>

* In the “show.html.erb” file under “app/views/photos”, replace

**<%=** link\_to 'Edit', edit\_photo\_path(@photo) %>

**<%=** link\_to 'Back', photos\_path %>

with

**<%=** link\_to 'Edit', edit\_event\_photo\_path(@event, @photo) %> |

**<%=** link\_to 'Back', event\_photos\_path(@event) %>

* In the “edit.html.erb” file under “app/views/photos”, replace

**<%=** link\_to 'Show', @photo %> |

**<%=** link\_to 'Back', photos\_path %>

with

**<%=** link\_to 'Show', event\_photo\_path(@event, @photo) %> |

**<%=** link\_to 'Back', event\_photos\_path(@event) %>

* In the “show.html.erb” file under “app/views/photos”, replace

**<%=** @photo.image **%>**

with

**<%=** image\_tag @photo.image(:medium) **%>**

* In the “index.html.erb” file under “app/views/photos”, replace

<td>**<%=** link\_to 'Show', photo %></td>

<td>**<%=** link\_to 'Edit', edit\_photo\_path(photo) %></td>

<td>**<%=** link\_to 'Destroy', photo, method: :delete, data: { confirm: 'Are you sure?' } %></td>

With

<td>**<%=** link\_to 'Show', event\_photo\_path(@event, photo) %></td>

<td>**<%=** link\_to 'Edit', edit\_event\_photo\_path(@event, photo) %></td>

<td>**<%=** link\_to 'Destroy', event\_photo\_path(@event, photo), method: :delete, data: { confirm: 'Are you sure?' } %></td>

* In the “index.html.erb” file under “app/views/photos”, replace

**<**%= link\_to 'New Photo', new\_photo\_path %>

with

**<**%= link\_to 'New Photo', new\_event\_photo\_path(@event) %>

* In the “index.html.erb” file under “app/views/photos”, replace

<td>**<%=** photo.image **%>**</td>

With

<td>**<%=** image\_tag photo.image(:thumb) **%>**</td>

* Wrap the codes in “new.html.erb”, “show.html.erb”, “index.html.erb” and “edit.html.erb” files under “app/views/photos” with a “div” block and give it a class “container”
* *In “ability.rb” file in the “app/models” folder, update “initialize” method as shown below*

**class** **Ability**

include CanCan**::**Ability

**def** **initialize(**user**)**

user **||=** User**.**new # guest user (not logged in)

can **[**:show**,** :index**],** Event

can **[**:show**,** :index**],** Photo

**if** user**.**has\_role? :organizer

can :manage**,** Event**,** :user\_id **=>** user**.**id

**end**

**if** user**.**has\_role? :enthusiast

can :manage**,** Photo**,** :user\_id **=>** user**.**id

**end**

**end**

**end**

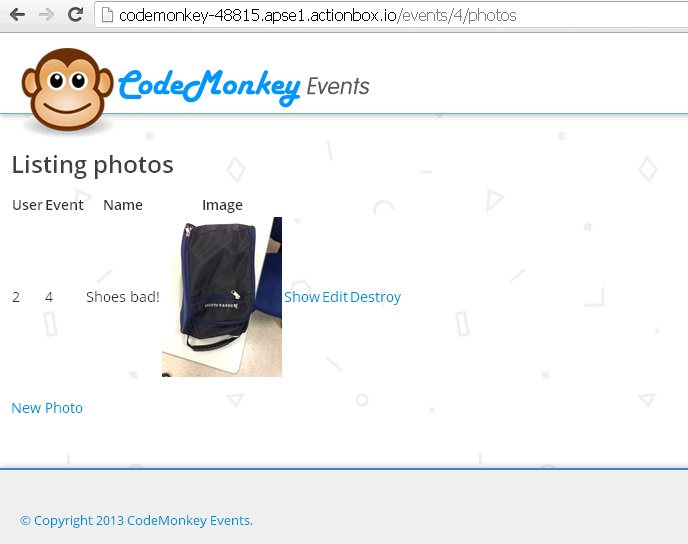
* In “\_featured\_preview.html.erb” and “\_normal\_preview.html.erb” under “app/views/events”, replace “0 Photos” with following

**<%=** pluralize(event.photos.count, "Photo") **%>**

* In “\_profile.html.erb” under “app/views/events”, replace “0 Photos with following

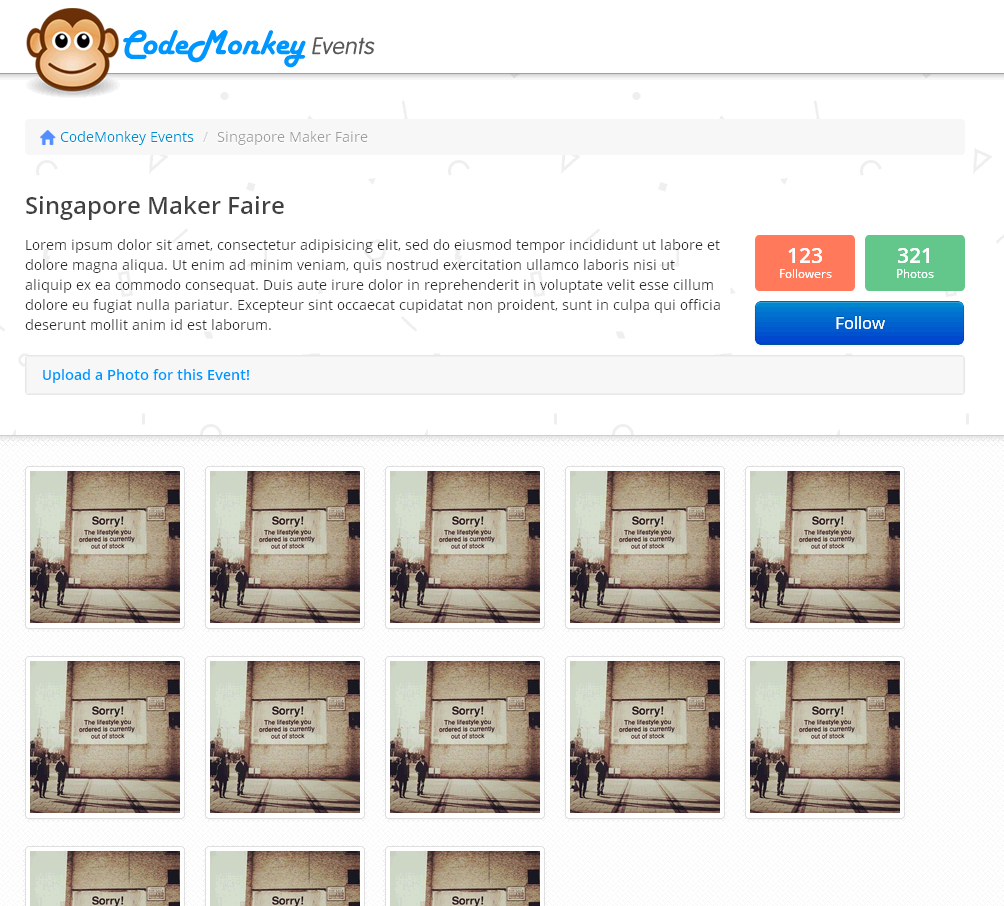
**<%=** pluralize(current\_user.photos.count, "Photo") **%>**

* Start the server, preview and go the “show” page of an event. Add “/photos” to the end of the url. E.g. <http://codemonkey-48815.apse1.actionbox.io/events/2/photos>
  + Try it out and play around!



# Beautifying individual event page

* In **Code Monkey Event**, we will not use the “index” page of Photo. Instead we will show the photos under the “show” page of Event.
* In this section, we will beautify the “show” page of Event as shown below.



### Instructions

* Replace the code in “show.html.erb” file under “app/views/events” folder with the following

**<**div **class=**"container"**>**

**<!--** **breadcrumb** **-->**

**<**div **class=**"row"**>**

**<div** **class=**"span12"**>**

**<**%= render "events/breadcrumb", :event =**>** @event %>

</div>

**<**/div>

<%= render "events/event\_info", :event => @event %>

</div>

<div class="**container-**fluid gallery">

<div class="container">

<div class="row">

<div class="span12">

<ul class="thumbnails">

<% @event.photos.each do |photo| %>

<li class="span3">

<%= render "events**/**photo\_preview", :photo => photo %>

</li>

<% end %>

</ul>

</div>

</div>

</div>

</div>

* Create a new file “\_breadcrumb.html.erb” in “app/views/events” folder and place the following codes.

**<**ul **class=**"breadcrumb"**>**

**<li>**

**<**a title**=**"Return to Home" href**=**"/"**><%= image\_tag('icon-home.png') %><**/a>

</li**>**

**<**li**>**

**<**a href**=**"/"**>**CodeMonkey Events**<**/a>

<span class="divider">/**<**/span>

</li**>**

**<**li **class=**"active"**><**%= event.title %></li>

</ul>

* Create a new file “\_event\_info.html.erb” in “app/views/events” folder and place the following codes.

**<**div **class=**"row"**>**

**<div** **class=**"span12"**>**

**<h1><**%= event.title %></h1>

</div>

</div>

<div class="row"**>**

**<**div **class=**"span9"**>**

**<p><**%= event.description %></p>

</div>

<div class="span3"**>**

**<**div **class=**"eventStatus followersStatus pull-left"**>**

**<div** **class=**"eventStatusInfo"**>**

**<h2>**0**<**/h2>Followers

</div**>**

**<**/div>

<div class="eventStatus photosStatus pull-left">

<div class="eventStatusInfo">

<h2><%= event.photos.count %></h2**>**

**<**%= "Photo".pluralize(event.photos.count) %>

</div>

</div>

<button onclick="#" **class=**"btn btn-primary btn-large followButton"**>** **Follow** **<**/button>

</div**>**

**<**/div>

<div class="row uploadPhoto">

<div class="span12">

<%= link\_to "Upload photo", new\_event\_photo\_path(event), :class => "btn btn-default" %>

</div**>**

**<**/div>

* Create a new file “\_photo\_preview.html.erb” in “app/views/events” folder and place the following codes.

**<**div **class=**"thumbnail"**>**

**<a** href**=**"<%= event\_photo\_url(@event, photo) %>"**>**

**<**img data**-**src**=**"holder.js/160x160" alt**=**"" src**=**"<%= photo.image.url(:medium) %>"**>**

**<**/a>

<div class="photoInfo">0 Comments</div**>**

**<**/div>

* In the “show.html.erb” file under “app/views/photos”, replace

**<**%= link\_to 'Back', event\_photos\_path(@event) %>

with

**<**%= link\_to 'Back', event\_path(@event) %>

* Start the server, preview and try it out.

# Allowing Enthusiasts to Follow an Event

* We will create functionalities to allow enthusiasts to follow an event

## Learning Point

* Using “acts\_as\_follower“ gem

## Instructions

* Add “acts\_as\_follower” gem to the Gemfile and download it by using bundle command. (If you are not sure how to do, refer to the previous sections)
* Run the generator

***rails generate acts\_as\_follower***

* Update database by calling rake db:migrate command
* Add “acts\_as\_followable” at the second line of “Event.rb” under ‘app/models” folder
* Add “acts\_as\_follower” at the second line of “User.rb” under ‘app/models” folder
* Add the following to the “events\_controller.rb” under “app/controllers” folder before “private” statement

**def** **follow**

@event **=** Event**.**find**(**params**[**:event\_id**])**

current\_user**.**follow**(** @event**)**

redirect\_to :back

**end**

**def** **unfollow**

@event **=** Event**.**find**(**params**[**:event\_id**])**

current\_user**.**stop\_following**(**@event**)**

redirect\_to :back

**end**

* Change the routes for Event in the “routes.rb” under “config” folder as show below

resources :events **do**

get :follow

get :unfollow

resources :photos **do**

resources :comments

**end**

**end**

* Create “\_my\_events.erb” under “app/views/events” folder and add the following lines

<div class=**"myEventsHeader sideContainerHeader"**>

**My Events**

</div>

<ul class=**"myEventsList"**>

**<%** **if** can? :create, **Event** **%>**

<li class=**"adminOption"**>

<h1><a href=**"<%=** new\_event\_path **%>"** role=**"button"** data-toggle=**"modal"**>**Create a New Event**</a></h1>

</li>

**<%** **end** **%>**

**<%** current\_user.all\_following.each **do** |**event**| **%>**

<li>

<h1><a href=**"<%=** event\_path(**event**) **%>"**>**<%=** event.title **%>**</a></h1>

<p>**<%=** pluralize(event.photos.count, "Photos") **%>**</p>

</li>

**<%** **end** **%>**

**<%** **if** current\_user.follow\_count == 0 **%>**

<li>

**No Events**

</li>

**<%** **end** **%>**

</ul>

* In “\_freatured\_preview.html.erb” under “app/views/events”, replace

<a href=**"#"** id=**"btn-featuredFollow"**>

<div class=**"span2 featuredEventButton featuredfollowButton"**>

<p class=**"btn-label"**>**Follow**</p>

</div>

</a>

with

**<%** **if** current\_user && current\_user.following?(**event**) **%>**

<a href=**"<%=** event\_unfollow\_path(**event**) **%>"** id=**"btn-featuredFollow"**>

<div class=**"span2 featuredEventButton featuredfollowButton featuredFollowingButton"**>

<p class=**"btn-label"**>**Unfollow**</p>

</div>

</a>

**<%** **else** **%>**

<a href=**"<%=** event\_follow\_path(**event**) **%>"** id=**"btn-featuredFollow"**>

<div class=**"span2 featuredEventButton featuredfollowButton"**>

<p class=**"btn-label"**>**Follow**</p>

</div>

</a>

**<%** **end** **%>**

* In “\_normal\_preview.html.erb” under “app/views/events”, replace

<a href=**""**><div id=**"btn-follow"** class=**"eventFollowButtonContainer"**>

</div></a>

with

**<%** **if** current\_user && current\_user.following?(**event**) **%>**

<a href=**"<%=** event\_unfollow\_path(**event**) **%>"**><div class=**"eventFollowButtonContainer followingEventButton"**></div></a>

**<%** **else** **%>**

<a href=**"<%=** event\_follow\_path(**event**) **%>"**><div id=**"btn-follow"** class=**"eventFollowButtonContainer"**></div></a>

**<%** **end** **%>**

* In “\_profile.html.erb” under “app/views/events”, replace “0 Follows” with

**<%= pluralize(current\_user.follow\_count , "Following") %>**

* In “\_event\_info.html.erb” under “app/views/events”, replace

**<h2>0</h2>Followers**

with

<h2>**<%=** event.followers\_count **%>**</h2>**<%=** "Follower".pluralize(event.followers\_count) **%>**

* In “\_event\_info.html.erb” under “app/views/events”, replace

<button onclick=**"#"** class=**"btn btn-primary btn-large followButton"**> **Follow** </button>

with

**<%** **if** current\_user && current\_user.following?(**event**) **%>**

<a href=**"<%=** event\_unfollow\_path(**event**) **%>"** id=**"btn-featuredFollow"**>

<button onclick=**"#"** class=**"btn btn-info btn-large followButton"**> **Unfollow** </button>

</a>

**<%** **else** **%>**

<a href=**"<%=** event\_follow\_path(**event**) **%>"** id=**"btn-featuredFollow"**>

<button onclick=**"#"** class=**"btn btn-primary btn-large followButton"**> **Follow** </button>

</a>

**<%** **end** **%>**

* Update “Ability.rb” file to allow “follow” and “unfollow” actions for enthusiasts. If you are not sure how to do, refer to the previous sections.
* Start the server and preview. ☺

# Creating Comment

* We will create pages related to comments
* For comments, we will allow users to only create but not to edit and delete.

## Learning point

* Using double nested resources

### Instructions

* Scaffold Comment

***rails generate scaffold Comment message:text photo\_id:integer user\_id:integer***

* Update database

***rake db:migrate***

* Add “belongs\_to” statements to “comment.rb” file under “app/models” folder in a similar way to what we have done for Photo. You need to add two belongs\_to statement as a comment belongs to both a user and a photo.
* Add “has\_many” statements to “photo.rb” and “user.rb” files under “app/models” folder in a similar way to what we have done for Photo.
* In the “routes.rb” under “config” folder, replace

resources :events **do**

resources :photos

**end**

with

resources :events **do**

resources :photos **do**

resources :comments

**end**

**end**

* In “\_form.html.erb” file under “app/views/comments” folder replace

**<%=** simple\_form\_for(@comment) **do** |f| **%>**

with

**<%=** simple\_form\_for([@**event**, @photo, @comment]) **do** |f| **%>**

* In the “\_form.html.erb” file under “app/views/photos” folder, replace

**<%=** f.input :user\_id **%>**

with

**<%=** f.input :user\_id, :**as** => :hidden, :input\_html => { :value => current\_user.id } **%>**

* In the “\_form.html.erb” file under “app/views/photos” folder, replace

**<%= f.input :photo\_id %>**

with

**<%=** f.input :photo\_id, :**as** => :hidden, :input\_html => { :value => @photo.id } **%>**

* In the “comments\_controller.rb” file, add the following code to the beginning of “index”, “new”, “edit”, “show”, “update” and “create” functions.

@event **=** Event**.**find**(**params**[**:event\_id**])**

@photo **=** Photo**.**find**(**params**[**:photo\_id**])**

* In the “comments\_controller.rb” file, edit “create” and “update” functions in a similar way to what we have done for Photo.
* *In “*comments *\_controller.rb” file in the “app/controllers” folder, add the following code at the second line*

before\_filter :authenticate\_user**!,** except: **[**:index**,** :show**]**

before\_filter :new\_comment**,** :only **=>** **[**:new**,** :create**]**

load\_and\_authorize\_resource

* *In “*comments *\_controller.rb” file in the “app/controllers” folder, add the following function after “private” statement*

**def** **new\_comment**

@comment **=** Comment**.**new

@comment**.**user\_id **=** current\_user**.**id

**end**

* *In “ability.rb” file in the “app/models” folder, update “initialize” method as shown below*

**class** **Ability**

include CanCan**::**Ability

**def** **initialize(**user**)**

user **||=** User**.**new # guest user (not logged in)

can **[**:show**,** :index**],** Event

can **[**:show**,** :index**],** Photo

**if** user**.**has\_role? :organizer

can :manage**,** Event**,** :user\_id **=>** user**.**id

**end**

**if** user**.**has\_role? :enthusiast

can :manage**,** Photo**,** :user\_id **=>** user**.**id

can :new**,** Comment

**end**

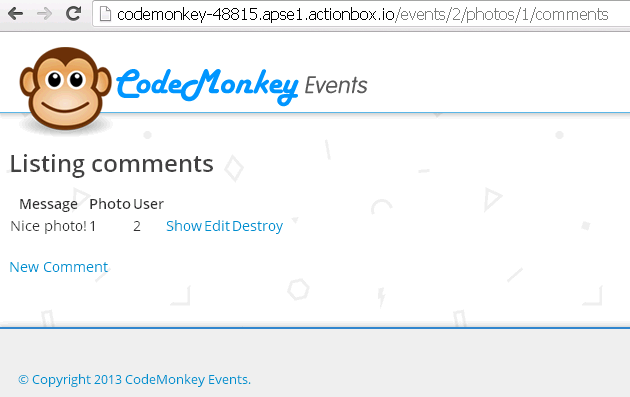
**end**

**end**

* Wrap the codes in “new.html.erb”, “show.html.erb”, “index.html.erb” and “edit.html.erb” files under “app/views/comments” with a “div” block and give it a class “container”
* Update all the “link\_to” lines in the files under “app/views/comments” folder in a similar way to what we have done for Photo.

Hint: You will need use both @event and @photo in the link\_to statement.

* Start the server, preview and go to the “show” page of Photo. And then add “/comments” at the end of the url. E.g. <http://codemonkey-48815.apse1.actionbox.io/events/2/photos/1/comments>



# Beautifying individual photo page

* In **Code Monkey Events**, we will not use any pages of Comment. Instead, they will be shown under “show” page of Photo
* We will beautify the “show” page of Photo as below

### Instructions

* Create a new file “\_breadcrumb\_preview.html.erb” in “app/views/photos” folder and place the following codes.

<ul class=**"breadcrumb"**>

<li><a title=**"Return to Home"** href=**"/"**>**<%= image\_tag('icon-home.png') %>**</a></li>

<li><a href=**"/"**>**CodeMonkey Events**</a><span class=**"divider"**>**/**</span></li>

<li class=**"active"**><a href=**"<%=** event\_path(**event**) **%>"**>**<%=** event.title **%>**</a><span class=**"divider"**>**/**</span></li>

<li class=**"active"**>**<%=** photo.name **%>**</li>

</ul>

* Create a new file “\_comment\_new\_preview.html.erb” in “app/views/photos” folder and place the following codes.

**<%** **if** user\_signed\_in? **%>**

<div class=**"span2"**>

<div class=**"span2 comment-profile"**>

<img src=**"<%=** current\_user.avatar.url(:thumb) **%>"** class=**"img-circle comment-avatar"** />

<div class=**"comment-username"**>**<%=** current\_user.name **%>**</div>

</div>

</div>

<div class=**"span9"**>

**<%=** simple\_form\_for [@**event**, @photo, Comment.new] **do** |f| **%>**

**<%=** f.input :photo\_id, :**as** => :hidden, :input\_html => { :value => @photo.id } **%>**

**<%=** f.input :user\_id, :**as** => :hidden, :input\_html => { :value => current\_user.id } **%>**

**<%=** f.input :message, :label => **false** **%>**

**<%=** f.button :submit **%>**

**<%** **end** **%>**

</div>

**<%** **else** **%>**

<div class=**"span12"**><p>**<%=** link\_to "Sign in", new\_user\_session\_path **%> to comment.**</p></div>

**<%** **end** **%>**

* Create a new file “\_comment\_show\_preview.html.erb” in “app/views/photos” folder and place the following codes.

<div class=**"span2 comment-profile"**>

<img src=**"<%=** comment.user.avatar.url(:thumb) **%>"** class=**"img-circle comment-avatar"** />

<div class=**"comment-username"**>**<%=** comment.user.name **%>**</div>

</div>

<div class=**"span9 comment-block"**>**<%=** comment.message **%>**</div>

* In “create” function of “comments\_controller.rb”, replace

format**.**html **{** redirect\_to event\_photo\_comment\_path**(**@event**,** @photo**,** @comment**),** notice: 'Photo was successfully created.' **}**

with

**format.html { redirect\_to event\_photo\_path(@event, @photo), notice: 'Photo was successfully created.' }**

* Replace the code in “show.html.erb” file under “app/views/photos” with the following

<div class=**"container content"**>

<div class=**"row"**>

<div class=**"span12"**>

**<%=** render "photos/breadcrumb", :photo => @photo, :**event** => @**event** **%>**

</div>

</div>

<div class=**"row"**>

<div class=**"span12"**><h1 class=**"galleryTitle"**>**<%=** @photo.name **%>**</h1></div>

</div>

</div>

<div class=**"container-fluid photoDisplay"**>

<div class=**"row-fluid"**>

<div class=**"container"**>

<div class=**"row"**>

<div class=**"span12"**>

<img src=**"<%=** @photo.image.url **%>"** title=**"<%=** @photo.name**%>"** class=**"img-polaroid"** />

</div>

</div>

</div>

</div>

</div>

<a name=**"comments"**></a>

<div class=**"container content"**>

<div class=**"row"**>

<div class=**"span12"**>

<h1>**Comments**</h1>

</div>

</div>

<div class=**"row comment-write"**>

**<%=** render "photos/comment\_new" **%>**

</div>

<div class=**"row"**>

<div class=**"span12"**>

<ul class=**"comments"**>

**<%** @photo.comments.each **do** |comment| **%>**

<li class=**"row"**>

**<%=** render "photos/comment\_show", :comment => comment **%>**

</li>

**<%** **end** **%>**

</ul>

</div>

</div>

</div>

# Sending Email Notifications

* We will send notification email to all the enthusiasts who are following an even when there is a new photo to the event

### Instructions

* Create Mailer

rails generate mailer notification\_mailer photo\_notification

* In “notification\_mailer.rb” under “app/mailers” folder, replace the photo\_notification function with the following

**def** **photo\_notification(**user**,** event**)**

@user **=** user

@event **=** event

mail to: user**.**email**,** subject: "New photo notification"

**end**

* Replace the code in the “photo\_notification.text.erb” file under “app/views/notification\_mailer” folder with the following

Hi **<**%= @user.name %>,

There is a new photo in <%= @event**.**title %>

<%= event\_url(@event) %>

* Create a new file “photo\_notification.html.erb” file under “app/views/notification\_mailer” folder and add the following

<p>**Hi <%=** @user.name **%>,**</p>

<p>**There is a new photo in <%=** @event.title **%>.**</p>

<p>**<%=** link\_to "View photos", event\_url(@event) **%>**</p>

* In “photos\_controller.html.erb” file under “app/controller” folder, update create function as below

**def** **create**

@event **=** Event**.**find**(**params**[**:event\_id**])**

@photo **=** @event**.**photos**.**new**(**photo\_params**)**

respond\_to **do** **|**format**|**

**if** @photo**.**save

@event**.**followers**.**each **do** **|**follower**|**

NotificationMailer**.**photo\_notification**(**follower**,** @event**).**deliver unless follower**.**id **==** current\_user**.**id

**end**

format**.**html **{** redirect\_to event\_photo\_path**(**@event**,** @photo**),** notice: 'Photo was successfully created.' **}**

format**.**json **{** render action: 'show'**,** status: :created**,** location: @photo **}**

**else**

format**.**html **{** render action: 'new' **}**

format**.**json **{** render json: @photo**.**errors**,** status: :unprocessable\_entity **}**

**end**

**end**

**end**

* In “development.rb” file under “config” folder, add the following before the last “end” statement

# Don't care if the mailer can't send

config**.**action\_mailer**.**raise\_delivery\_errors **=** **false**

# Change mail delvery to either :smtp, :sendmail, :file, :test

config**.**action\_mailer**.**delivery\_method **=** :smtp

config**.**action\_mailer**.**smtp\_settings **=** **{**

address: "smtp.gmail.com"**,**

port: 587**,**

domain: "code-monkey-events.com"**,**

authentication: "plain"**,**

enable\_starttls\_auto: **true,**

user\_name: **ENV[**"GMAIL\_USERNAME"**],**

password: **ENV[**"GMAIL\_PASSWORD"**]**

**}**

# Specify what domain to use for mailer URLs

config**.**action\_mailer**.**default\_url\_options **=** **{**host: "localhost:3000"**}**

* Set gmail information. Don’t use your personal account. Sign up a new account and use

echo “export ENV[‘GMAIL\_USERNAME’]=’your gmail @gmail.com’”

echo “export ENV[‘GMAIL\_PASSWORD’]=’your password’”

* Restart the server and try it out. It should work. ☺
* If you want to send emails on Heroku, we need to do additional configuration. We will cover how to do it during the training.

# Publishing to heroku

* We will host the app on Heroku
* You need an heroku account. You can register for free at <https://www.heroku.com/>

## Learning Points

* Hosting an application on Heroku

## Instructions

* In your Gemfile, replace

*gem 'sqlite3'*

with

*gem 'sqlite3', group: [:development]*

* In your Gemfile, add the following

*group :production do*

*gem 'pg', '0.15.1'*

*gem 'rails\_12factor'*

*end*

* In “production.rb” under “config/environments” folder set

config.serve\_static\_assets = true

* Create a git repo

***git init***

* Set up email and name

***git config --global user.email “your email here”***

***git config --global user.name “your name here”***

* First commit

***git add .***

***git commit -m “Any message here”***

* Set up public key

***heroku keys:add***

You will be prompted to log in to your heroku account. You can replace “code-monkey-events” with any name that you want as long as it is not already taken.

* Create a heroku app

***heroku apps:create code-monkey-events***

***git push heroku master***

* Upgrate database

***heroku run rake db:migrate***

* Congratulation! Your app is live now. In your browser, go to the url “your\_app\_name.herokuapp.com” and see your app in action.