Matt’s Bookstore

1. Rails new bookshop
2. Rails db:setup
3. Rails db:create
4. Rails s
5. Rails g model author
6. Add to author model migration file
7. t.string :name
8. t.string :url\_image
9. t.timestamps
10. rails db:migrate
11. rails s
12. rails g controller authors
13. add def index in authors controller
14. add to routes.rb
15. resources :authors
16. Create index.html.erb in views > authors

* Add identifying text inside the file

1. Add root “authors#index” to routes.rb
2. Paste css into application.css from slack
3. add to def index in authors controller
4. @authors = Author.all

20. Add to authors index.html.erb

<% @authors.each do |author| %>

<%= author.name %>

<% end %>

1. Copy data into seed.rb from channel

* Comment out the books constant section of the seed file

1. In authors index.html.erb
2. <h1>Authors</h1>
3. <div *class*="authors">
4. <% @authors.each do |author| %>
5. <div *class*="author">
6. <%= image\_tag author.url\_image, width: 64 %>
7. <h3><%= author.name %></h3>
8. </div>
9. <% end %>
10. </div>
11. Set up book model and relationship with authors.
12. Rails s
13. Rails g model book title:string url\_image:string author:references
14. Rails db:migrate
15. Add in author.rb
16. has\_many :books
17. uncomment the books constant in seed.rb
18. rails db:reset
19. rails s
20. modify authors index.html.erb

<div *class*="authors">

<% @authors.each do |author| %>

<div *class*="author">

<%= image\_tag author.url\_image, width: 64 %>

<h3><%= author.name %></h3>

<p><%= author.books.count %> books</p>

</div>

<% end %>

</div>

1. rails c
2. Author.create(name: “Jane Austen”, url\_image: <https://upload.wikimedia.org/wikipedia/commons/thumb/c/cc/CassandraAusten-JaneAusten%28c.1810%29_hires.jpg/400px-CassandraAusten-JaneAusten%28c.1810%29_hires.jpg>
3. Book.create(title: “Pride and Prejudice”, url\_image: <https://upload.wikimedia.org/wikipedia/commons/thumb/1/17/PrideAndPrejudiceTitlePage.jpg/440px-PrideAndPrejudiceTitlePage.jpg>, author\_id: 4)
4. In authors index.html.erb, update

<p><%= pluralize author.books.count, "book" %></p>

1. rails db:setup
2. rails s
3. in author index.html.erb, add at bottom

<div *class*="admin\_tools">

<h2>Administrator's Tools</h2>

<ul>

<li>Add a new author</li>

<li>Create a book</li>

</ul>

</div>

1. rails g controller books
2. books controller, add

def new

end

1. need to set up routes because there’s nothing to link to the actions in the books controller.

Add resources :books to routes.rb

1. create new.html.erb in views >books
2. in authors index.html.erb add to
3. <li><%= link\_to "Add a new book", new\_book\_path %></li>
4. In books controller, add instance variable to use in form

def new

@authors = Author.all

end

1. In books new.html.erb add

<%= form\_for :book do |f| %>

<p>

<%= f.label :title %>

<%= f.text\_field :title %>

</p>

<p>

<%= f.label :url\_image, "Image URL" %>

<%= f.text\_field :url\_image %>

</p>

<p>

<%= f.label :author\_id, "Author ID" %>

<%= f.select :author\_id, options\_from\_collection\_for\_select(@authors, :id, :name) %>

</p>

<p>

<%= f.submit "Create book" %>

</p>

<% end %>

SIDENOTE about options\_from\_collection\_for\_select

<%= f.select :author\_id, options\_from\_collection\_for\_select(@authors, :id, :name) %>

<!--

equivalent of options\_from\_collection\_for\_select above

<select>

<% @authors.each do |author| %>

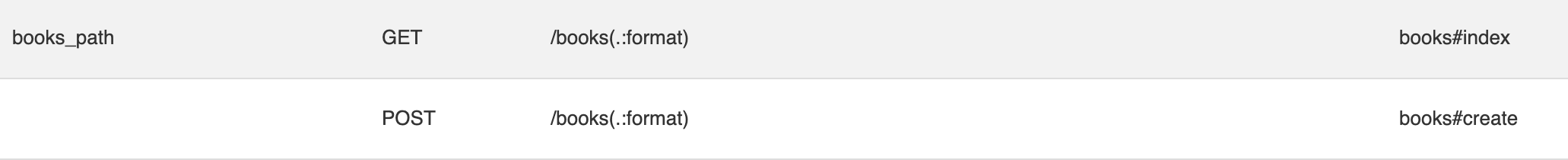
<option value ="<%= author.id %>"><%= author.name %></option>

</select>

-->

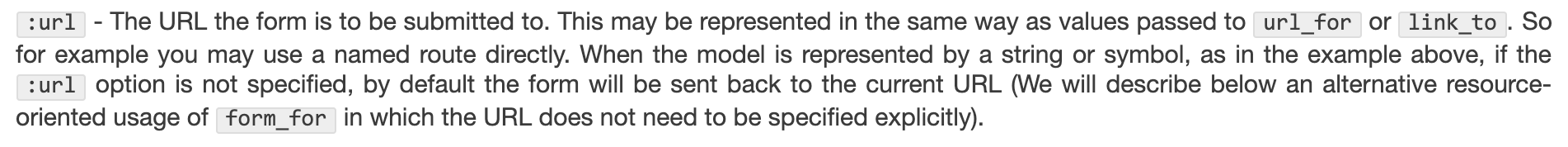
58. When you submit this form, it will fail because the route is not properly set up for the submit.

We look in the rails routes and find to create a new book on the system, the route is books\_path with the method of POST

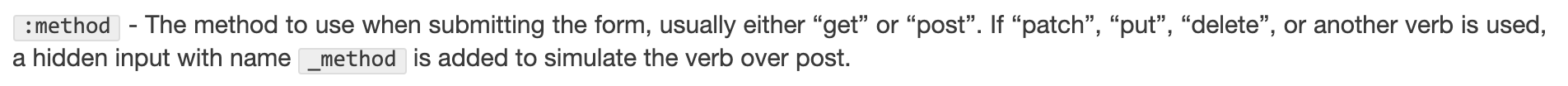


Then we look in api ruby rails docs for form\_for to find the syntax conventions

:url is for books\_path



:method is for the POST



We modify the form in books new.html.erb to

<%= form\_for :book, url: books\_path, method: :post do |f| %>

1. On books controller, add def create to submit the data to model

def create

# Create an entry on book model

# Book.create(title: something, url\_image: something, author\_id: something)

# these values are passed in from params as params[:book][:title]

# the equivalent

# title = params[:book][:title]

# url\_image = params[:book][:url\_image]

# author\_id = params[:book][:author\_id]

# Book.create(title: title, url\_image: url\_image, author\_id: author\_id)

Book.create(title: params[:book][:title], url\_image: params[:book][:url\_image], author\_id: params[:book][:author\_id])

# after it is created, go back to the home page

redirect\_to authors\_path

end

1. Modify def create in books controller for a more secure way of

def create

# params to expect a key of book which will include title, url\_image, author\_id

# secure against sql injection

book\_params = params.require(:book).permit(:title, :url\_image, :author\_id)

Book.create(book\_params)

# after it is created, go back to the home page

redirect\_to authors\_path # /authors

end

this is how scaffold will do it with private def book\_params

* book\_params can be used in edit method

1. In Books controllers, Edit def create and create a new method def book\_params for future use
2. def create
3. Book.create(book\_params)
4. # after it is created, go back to the home page
5. redirect\_to authors\_path # /authors
6. end

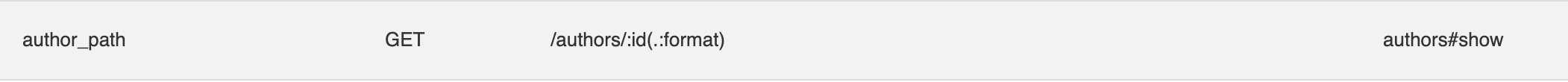
private

def book\_params

params.require(:book).permit(:title, :url\_image, :author\_id)

end

1. In layouts > application.html.erb
2. <body>
3. <h1>Amazon Bookstore</h1>
4. <%= yield %>
5. <footer>
6. <ul>
7. <li><%= link\_to “Home”, root\_path %></li>
8. <li>Privacy Policy</li>
9. <li>Contact Page</li>
10. </ul>
11. </footer>
12. </body>
13. Change authors index.html.erb book link to go to showing everything to do with the particular author



Note it takes an id

<p><%= link\_to pluralize(author.books.count, "book"), author\_path(author.id) %></p>

1. In authors controller, add def show
2. Create views > authors > show.html.erb

<div *class*="author">

<%= image\_tag @author.url\_image, width: 64 %>

<h2><%= @author.name %></h2>

</div>

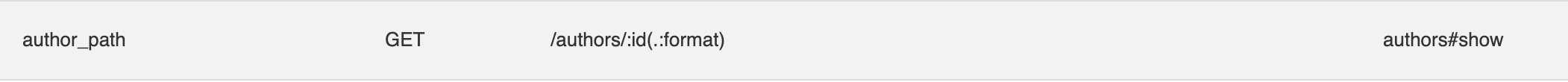
1. To create @author in views > authors >show.html.erb, create it in the def show of authors controller
2. Update def show of authors show.html.erb

def show

@author = Author.find(params[:id])

end

params[:id] here is derived from the id requirement in /authors/1 or author\_path



1. We can use @author.books to get to the books by the author because of the has\_many relationship in author model

In authors show.html.erb, add

<div *class*="author">

<%= image\_tag @author.url\_image, width: 64 %>

<h2><%= @author.name %></h2>

<p>There are <%= pluralize @author.books.count, "book" %> written by <strong><%= @author.name %></strong></p>

<ul>

<% @author.books.each do |book| %>

<li><%= book.title %></li>

<% end %>

</ul>

</div>

1. Add “add a book” link to authors show.html.erb



<div>

<%= link\_to "Add a book by #{@author.name}", new\_book\_path %>

</div>

1. When we click through that add a book link, it should default to the same author. The URL should also be **/author/id/books/new** and not **books/new. This is a nested route to specify constraints how resources fit together.**

**(books belong to authors)**

On routes.rb change to

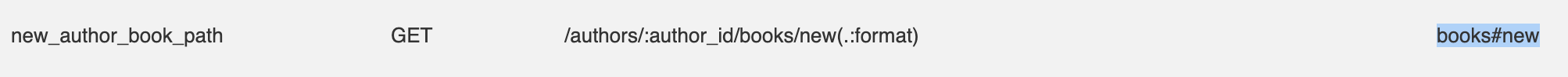
resources :authors do

resources :books

end

This changes all the routes, so all the route paths on links need to change.

1. For add a new book link in authors show.html.erb



Update to

<div>

<%= link\_to "Add a book by #{@author.name}", new\_author\_book\_path(@author.id) %>

</div>

More errors ensue

1. in new.html.erb for book

change

<%= form\_for :book, url: books\_path, method: :post do |f| %>

To

<%= form\_for [:author, @book] do |f| %>

Active record object book belongs to an author

1. define @book in books controller
2. def new
3. # instance variable accessible in the view
4. @authors = Author.all
5. # instantiating a new object from book and assigning it to @book so allow the new book form to use it
6. @book = Book.new(author\_id: params[:author\_id])
7. end
8. in books new.html.erb, modify this line to make the drop down list show the relevant author clicked through

<%= f.select :author\_id, options\_from\_collection\_for\_select(@authors, :id, :name, @book.author\_id) %>

1. Because of nested routes, new\_book\_path no longer exists. We can create it again by enabling both nested and un-nested (root-level) routes to create a new book

New\_author\_book\_path requires an author id argument which ties creating every new book to an author



Create a book without specifying an author

Rails.application.routes.draw do

# For details on the DSL available within this file, see http://guides.rubyonrails.org/routing.html

resources :authors do

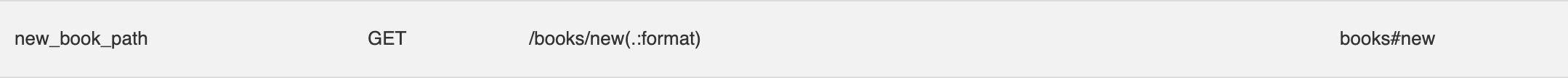
resources :books

end

resources :books, only: [:new]

root to: "authors#index"

end



1. Remove the add new book link in authors index.html.erb

Delete

<li><%= link\_to "Add a new book", new\_book\_path %></li>

1. Go to authors index.html.erb, add to the top
2. <form *action*="<%= authors\_path %>">
3. <label>Search by Author:
4. <input *type*="text" *name*="search">
5. </label>
6. <button *type*="submit">Search</button>
7. </form>
8. Mock up search

Rails c

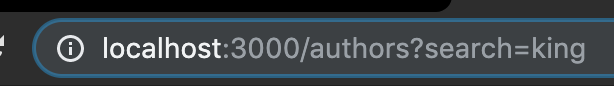
Search.upcase = ‘king’

Author.where(“upper(name) LIKE ?”, “%#{search}%”) 🡪 for postgresql

? is a placeholder for search term

1. Change the form action on authors index.html.erb to return a authors path

<form *action*="<%= authors\_path %>">



1. In authors controller
2. def index
3. # search term is whatever that is inputted in the form
4. @search\_term = params[:search]
5. if @search\_term.blank?
6. @authors = Author.all
7. else
8. @authors = Author.where("upper(name) LIKE ?", "%#{@search\_term.upcase}%")
9. end
10. end

to make a variable usable in the view, make it an instance variable

1. Added under near search form on authors index.html.erb

<% if @search\_term %>

Found <%= pluralize @authors.count, "authors" %> matching <em><%= @search\_term %></em>

<% end %>

1. Modify the search form on authors index.html.erb so that search term persists in the text field

<input *type*="text" *name*="search" *value*="<%= @search\_term %>">