Notes, 11-13-14

Steps:

1.) Rails new soccer

2.) Cd in to that new folder created—create new Models (Models are always singular)

3.) rails g model Player name:string jersey:integer team:references

4.) rails g model Team name:string country:string

5.) go to your sublime:

>Models > team.rb

add:

class Team < ActiveRecord::Base

has\_many :players

end

6.) >Config > routes.rb

add: (to do the FULL CRUD!)

resources :teams

7.) go back to Terminal after you added that one line into your routes.rb file…and enter: --look at what it creates automatically!

Sandis-MacBook-Air:soccer sandi$ rake routes

Prefix Verb URI Pattern Controller#Action

teams GET /teams(.:format) teams#index

POST /teams(.:format) teams#create

new\_team GET /teams/new(.:format) teams#new

edit\_team GET /teams/:id/edit(.:format) teams#edit

team GET /teams/:id(.:format) teams#show

PATCH /teams/:id(.:format) teams#update

PUT /teams/:id(.:format) teams#update

DELETE /teams/:id(.:format) teams#destroy

8.) >Config > routes.rb

add:

resources :teams do

resources :players

end

9.) Now, we need to rake routes a second time b/c we changed the resources—and we see a lot more routes created.

Sandis-MacBook-Air:soccer sandi$ rake routes

Prefix Verb URI Pattern Controller#Action

team\_players GET /teams/:team\_id/players(.:format) players#index

POST /teams/:team\_id/players(.:format) players#create

new\_team\_player GET /teams/:team\_id/players/new(.:format) players#new

edit\_team\_player GET /teams/:team\_id/players/:id/edit(.:format) players#edit

team\_player GET /teams/:team\_id/players/:id(.:format) players#show

PATCH /teams/:team\_id/players/:id(.:format) players#update

PUT /teams/:team\_id/players/:id(.:format) players#update

DELETE /teams/:team\_id/players/:id(.:format) players#destroy

teams GET /teams(.:format) teams#index

POST /teams(.:format) teams#create

new\_team GET /teams/new(.:format) teams#new

edit\_team GET /teams/:id/edit(.:format) teams#edit

team GET /teams/:id(.:format) teams#show

PATCH /teams/:id(.:format) teams#update

PUT /teams/:id(.:format) teams#update

DELETE /teams/:id(.:format) teams#destroy

10.) rails g controller Teams (\*\* controllers are always plural)

11.) rails g controller Players

12.) If you don’t add it on the command line above…you will have to then to into the >App>Controller files to add this to both players\_controllers.rb and teams\_controllers.rb files: (assuming you want to do the full CRUD)

def index

end

def create

end

def new

end

def edit

end

def show

end

def update

end

def destroy

end

13.) REST—representational state transfer

When you see the Controller#Action section from rake routes, that list is the restful routes.

14.) went to : google.com and entered “ruby form helpers”

<http://guides.rubyonrails.org/form_helpers.html>

Dealing with Basic Forms:

<%= form\_tag do %>

  Form contents

<% end %>

Other ones as see on this website:

<%= form\_tag("/search", method: "get") do %>

  <%= label\_tag(:q, "Search for:") %>

  <%= text\_field\_tag(:q) %>

  <%= submit\_tag("Search") %>

<% end %>

15.) Now we go back to sublime, to create a new team

Look at the list seen from rake routes on terminal:

new\_team GET /teams/new(.:format) teams#new

--we are creating a new team with this new form.

--the form knows what models we are trying to update with that rails helper tag below. –this is essentially a SMART form!

>App>Views>Teams> created a new.html.erb file added:

<%= form\_for @team do |f| %>

<%= f.text\_field(:name) %>

<%= f.text\_field(:country) %>

<%= submit\_tag("Submit New Team") %>

<% end %>

>Ap>Controllers>teams\_controllers.rb

def new

@team = Team.new

end

16.) after firing up the rails server we enter: localhost:3000/teams/new

If we enter data into those fields, it will need to post, so seeing what’s available from rake routes—it will inform us on which step to do next on the teams\_controller.rb file.

POST /teams(.:format) teams#create

def create

Team.create(name: params[:team][:name], country: params[:team][:country])redirect\_to "/"

end

17.) to check if they got submitted, go to rails console in terminal

enter: Team.all

from -e:1:in `<main>'irb(main):002:0> Team.all

**Team Load (2.3ms)** **SELECT "teams".\* FROM "teams"**

=> #<ActiveRecord::Relation [#<Team id: 1, name: "USA Team", country: "USA", created\_at: "2014-11-13 18:50:42", updated\_at: "2014-11-13 18:50:42">]>

In the server, when submit another team from the form –we see this occurring:

Started POST "/teams" for 127.0.0.1 at 2014-11-13 10:53:58 -0800

Processing by TeamsController#create as HTML

Parameters: {"utf8"=>"✓", "authenticity\_token"=>"Ym0KgWa/4dFGGW0OpsUV/OREFx2BOdQx1sNNHpvHu0I=", "team"=>{"name"=>"Manchester", "country"=>"England"}, "commit"=>"Submit New Team"}

**(0.2ms)** **begin transaction**

**SQL (0.5ms)** INSERT INTO "teams" ("country", "created\_at", "name", "updated\_at") VALUES (?, ?, ?, ?) [["country", "England"], ["created\_at", "2014-11-13 18:53:58.507658"], ["name", "Manchester"], ["updated\_at", "2014-11-13 18:53:58.507658"]]

**(0.9ms)** **commit transaction**

Redirected to http://localhost:3000/

Completed 302 Found in 13ms (ActiveRecord: 1.6ms)

(?, ?, ?, ?) is called “parameter binding”, it’s used to prevent SQL insertion

To add a level of security to our app (industry standard)

We will go into teams\_controller.rb and change it to reflect this on the def create and create a Private method as well: (called “Strong Parameters)- you don’t want to expose certain fields in your model for updates. Without this it will expose a way into our application—strong parameters prevents ways into our application.

def create

Team.create(params)

redirect\_to "/"

end

private

def team\_params

params.require(:team).permit(:name, :country)

end

18.) To Edit the data for teams (update)

edit\_team GET /teams/:id/edit(.:format) teams#edit

PUT /teams/:id(.:format) teams#update (put goes into your views (edit.html.erb file)

Add to: teams\_controller.rb:

def edit

@team = Team.find(params[:id])

Add to edit.html.erb

<%= form\_for @team, method: "put" do|f| %>

19.) to Update the Edit for Teams:

def update

team = Team.find(params[:id])

team.update(team\_params)

redirect\_to "/"

You could see it on the server to make sure the update went through:

**(0.2ms)** **begin transaction**

**SQL (0.5ms)** UPDATE "teams" SET "name" = ?, "updated\_at" = ? WHERE "teams"."id" = 1 [["name", "USA "], ["updated\_at", "2014-11-13 19:51:49.468281"]]

**(0.8ms)** **commit transaction**

Redirected to http://localhost:3000/

Completed 302 Found in 59ms (ActiveRecord: 2.3ms)

20.) To show all the teams, here is the route from rake route:

teams GET /teams(.:format) teams#index

Config> routes.rb file: root "teams#index"

Controllers>teams\_controllers.rb file:

def index

@team = Team.all

render "index"

Created a new index.html.erb file (Views>Teams)

<% @team.each do|t| %>

<p> <%= t.name %>, <%=t.country %></p>

<% end %>

21.) Now do the same for a new team player for a specific team:

http://localhost:3000/teams/1/players/new

new\_team\_player GET /teams/:team\_id/players/new(.:format) players#new

In the player\_controller.rb

def new

@player = Player.new

end

1/

def create

# first we have to find the team, then push in our new player

team = Team.find(params[:team\_id])

player = Player.create(player\_params)

team.players << player

redirect\_to "/"

end

In the new.html.erb file for the players

<%= form\_for @player, url: team\_players\_path(params[:team\_id]) do |f| %>

<%= f.text\_field(:name, placeholder: "Player Name") %>

<%= f.text\_field(:jersey, placeholder: "Jersey Number") %>

<%= submit\_tag("Submit New Player") %>

<% end %>

When successful, you will see this on the server:

**(0.2ms)** **begin transaction**

**SQL (1.1ms)** INSERT INTO "players" ("created\_at", "jersey", "name", "updated\_at") VALUES (?, ?, ?, ?) [["created\_at", "2014-11-13 20:37:55.227927"], ["jersey", 1], ["name", "bob"], ["updated\_at", "2014-11-13 20:37:55.227927"]]

**(0.8ms)** **commit transaction**

**(0.1ms)** begin transaction

**SQL (0.6ms)** **UPDATE "players" SET "team\_id" = ?, "updated\_at" = ? WHERE "players"."id" = 1** [["team\_id", 1], ["updated\_at", "2014-11-13 20:37:55.446561"]]

**(0.9ms)** commit transaction

Redirected to http://localhost:3000/

Completed 302 Found in 273ms (ActiveRecord: 5.6ms)

Alternatively, a different syntax:

In the new.html.erb:

<%= form\_for [@team, @player] do |f| %>

In the Players\_controller.rb—we added a new @team line (see below.)

def new

@team = Team.find(params[:team\_id])

To refactor this one that is 3 database calls to ONE database call instead:

def create

# first we have to find the team, then push in our new player

team = Team.find(params[:team\_id])

player = Player.create(player\_params)

team.players << player

so to refactor, we are just doing one create statement, instead of finding, and shoveling--this one is a refactor of three lines.

player = Player.create(player\_params.merge(team\_id: params[:team\_id]))

22.) In order to edit:

http://localhost:3000/teams/1/players/2/edit

players\_controller.rb

def edit

@team = Team.find(params[:team\_id])

@player = Player.find(params[:id])

end

edit.html.rb

<%= form\_for [@team, @player] do |f| %>

<%= f.text\_field(:name) %>

<%= f.text\_field(:jersey) %>

<%= submit\_tag("Edit Player") %>

<% end %>

23.) In order to update:

players\_controller.rb

def update

player = Player.find(params[:id])

player.update(player\_params)

redirect\_to "/"

edit.html.rb

<%= form\_for [@team, @player], method: "put" do |f| %>

we went into player\_controller.rb to refactor some code and created a few “before\_filter” –allowed us to remove /refactor a bunch of repeat code

before\_filter :assign\_team, only: [:new, :edit]

before\_filter :assign\_player, only: [:edit, :update]

then we defined it in the private methods section

def assign\_team

@team = Team.find(params[:team\_id])

end

def assign\_player

@player = Player.find(params[:id])

end

23.) In order to delete players:

players\_controller.rb

def destroy

@player.destroy

redirect\_to "/"

end

edit.html.rb (we created another form for the delete—this created a delete button on the edit form)

<%= form\_for [@team, @player], method: "delete" do |f| %>

<%= submit\_tag("Delete Player") %>

<% end %>

In order to delete teams:

teams\_controller.rb

def destroy

team = Team.find(params[:id])

team.destroy

redirect\_to "/"

end

Now, to show ALL the players and teams on the index.html.erb form (teams side), upon render on root (‘/’), we changed it to reflect:

>Apps>Teams>index.html.erb—made the changes to loop through the players and create new links to link to the edit page (for editing and deleting)

<% @team.each do|t| %>

<div>

<h1> <%= t.name %></h1>

<a href="/teams/<%= t.id%>/edit">Edit Team</a>

<h2> <%=t.country %></h2>

<% t.players.each do |p| %>

<h3> <%= p.name %> <%= p.jersey %>

<a href="/teams/<%= t.id%>/players/<%=p.id%>/edit">Edit Player</a>

</h3>

<% end %>

</div>

<% end %>

http://localhost:3000/rails/info/routes

this a browser based list of routes that is interactive.

Notes from board:

When you do this with resources : resources, it will create in your new Routes

***Routes:***

-root

- get -> new team form

-get -> new player form

-post -> new team

-post -> new player

-put -> edit team

-put -> edit player

-delete -> player

-delete -> team

post -> /players

post -> /teams/:id/players

put -> /teams/:id/players/:id

***Player (model)***

-name

-jersey

-team\_id

***Team (model)***

-name

-country