



Rails Girls

INTRO TO MVC + STYLING

Stephanie Nemeth + Melanie Keatley

INTRODUCTION

- Welcome!
- Meet our host + presenters
- Intro to our App
- Intro to Model-View-Controller
- Lunch!
- Intro to CSS
- Bootstrapping our App

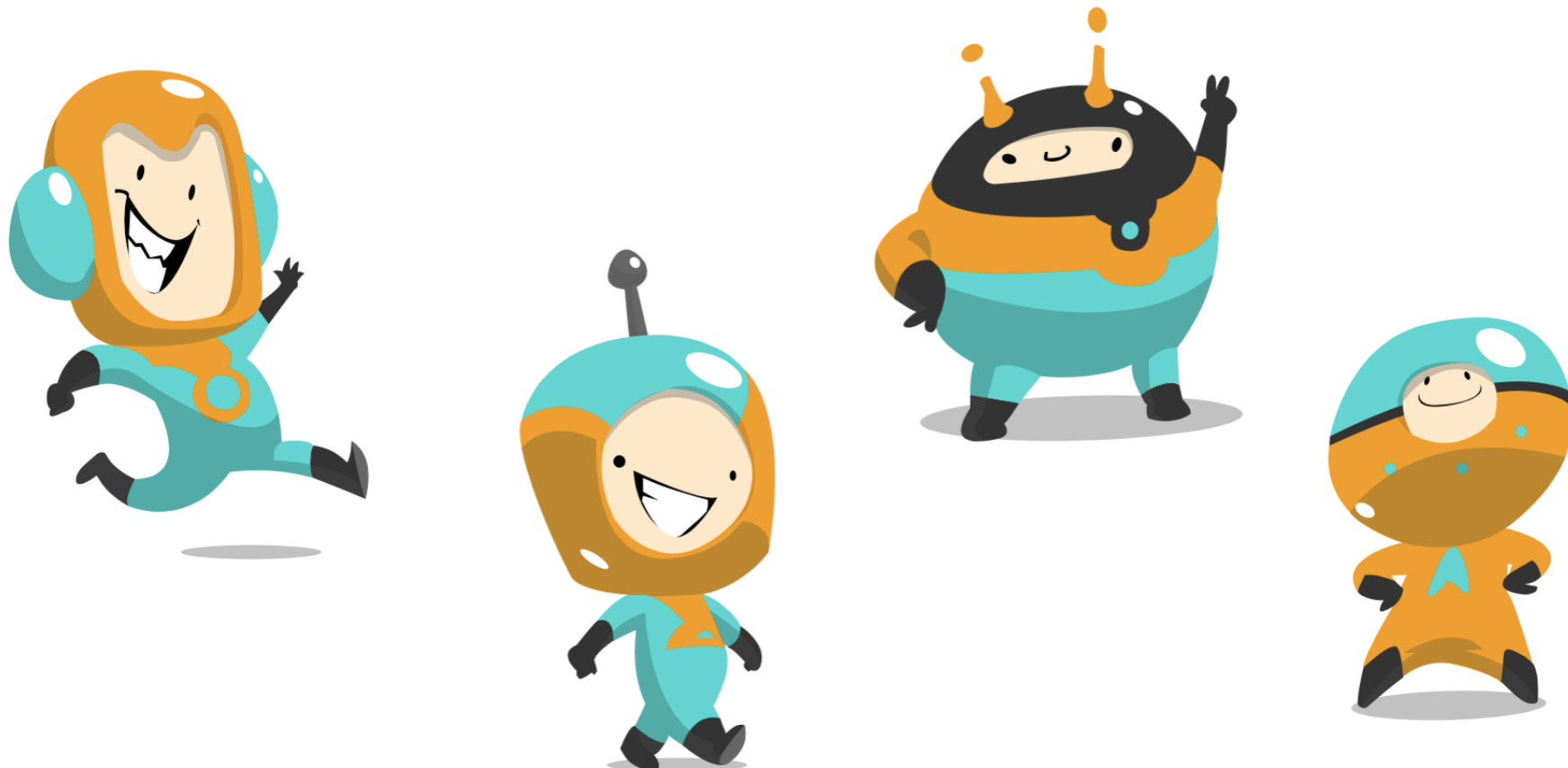


OUR HOST...

Space Babies

We:

- Create beautiful, fast and secure web applications
- Regularly offer apprenticeships
- E-mail info@spacebabies.nl to get in touch



THE PRESENTERS



Stephanie Nemeth

 @stephaniecodes



Melanie Keatley

 @Keatley

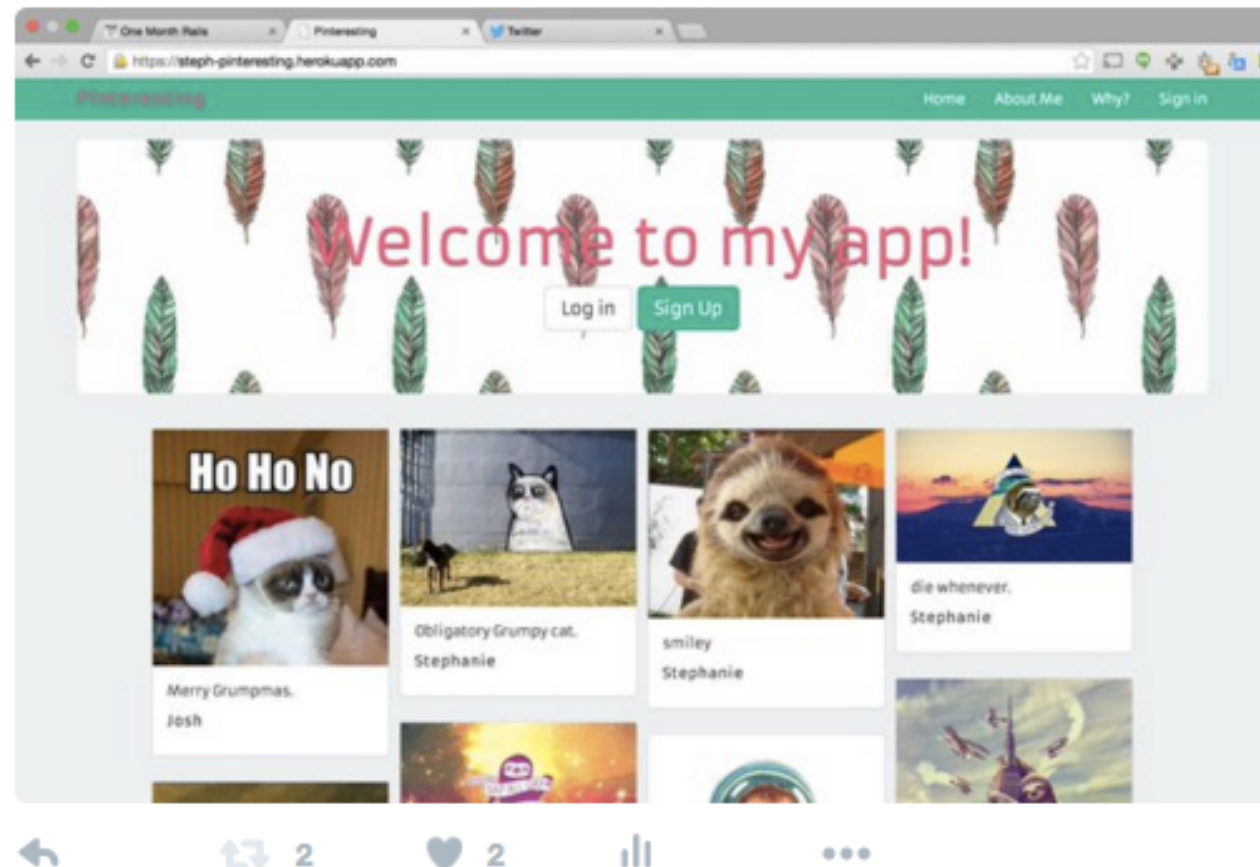
A YEAR AGO...

.....



Stephanie Nemeth @stephaniecodes · 16 Dec 2014

I just finished building my first app!
bit.ly/1zPwM7I On to the next project...
Thanks @onemonthedu!



AND NOW...



```
1 class IssueRefund
2   include Interactor
3
4   delegate :purchase, :amount, :available_amount, to: :context
5
6   before do
7     context.account = purchase.account
8     context.fail! message: "Sorry, the refund amount cannot be zero. Please try again." if amount.zero?
9     context.fail! message: "Sorry, the amount is higher than the amount available for refund. Please try again." if amount > availa
10  end
11
12  def call
13    response = hermes.post "messages", {message: "purchase/refund", parameters: { purchase_id: purchase.to_param, amount: amount.to
14    if response.success?
15      context.refund = hermes.process(response)
16      purchase.update_attributes status: "refunded"
17      PublishStateEvent.call purchase: purchase
18      Tracker::AddEventToTimeline.call({owner_uid: owner_uid, ref_id: purchase.ref_id, source: "Refund", name: "purchase_refunded_b
19    else
20      context.fail! message: "Purchase failed to be refunded"
21    end
22
23    rescue S2P::BadRequest => error
24      context.fail! message: error.message, status: :bad_request
25  end
26
27  private
```


OUR APP



- Let's first take a look at our app so far.
 - Local:
 - Open terminal & navigate into app's folder: \$ **cd railspet**
 - To start rails server: \$ **bin/rails server**
 - Then, go to <http://localhost:3000> in your browser.
 - Cloud9:
 - Login to c9.io in the browser & open your railspet workspace.
 - To start rails server: \$ **rails s -p \$PORT -b \$IP**
 - Then, refresh the preview window.

OUR APP

- Our app should look something like this....



A screenshot of a Cloud9 IDE environment. The top bar shows the 'Cloud9' logo and various menu items like File, Edit, Find, View, Goto, Run, Tools, Window, Support, Preview, and Run. The left sidebar displays a file explorer for the 'railspet' project, showing folders like app, bin, config, db, lib, log, public, spring, test, tmp, vendor, and files like config.ru, Gemfile, Gemfile.lock, RackMultiPart20151207-7, Rakefile, and README.md. The main editor area shows a web browser with the URL 'https://railspet-traumverloren.c9users.io/'. The browser displays a page titled 'My Pets' with a table of pet information. The table has columns for 'Name', 'Max weight', 'Foods Eaten', and actions. Two pets are listed: 'wilbur' (600 kg) and 'dasher' (50 kg). Below the table is a link for 'New Pet'. The bottom panel shows a terminal with logs for a GET request to '/pets', including database queries and rendering times.

Cloud9 File Edit Find View Goto Run Tools Window Support Preview Run

[P] /README.md [B] https://railspet-traumverloren.c9users.io/ routes.rb index.html.erb pets_controller.rb

https://railspet-traumverloren.c9users.io/ Browser

My Pets

	Name	Max weight	Foods Eaten	
	wilbur	600 kg	%	Show Edit Delete
	dasher	50 kg	%	Show Edit Delete

[New Pet](#)

bash - "traumverlore" x ruby - "traumverlore" x

```
Food Load (0.2ms) SELECT "foods".* FROM "foods" WHERE "foods"."pet_id" = ? [{"pet_id", 3}]
Rendered pets/show.html.erb within layouts/application (18.8ms)
Completed 200 OK in 130ms (Views: 127.4ms | ActiveRecord: 0.5ms)

Started GET "/pets" for 83.162.47.219 at 2015-12-07 16:43:34 +0000
Cannot render console from 83.162.47.219! Allowed networks: 127.0.0.1, ::1, 127.0.0.0/127.255.255.255
Processing by PetsController#index as HTML
Pet Load (0.4ms) SELECT "pets".* FROM "pets"
Food Load (0.2ms) SELECT "foods".* FROM "foods" WHERE "foods"."pet_id" = ? [{"pet_id", 1}]
Food Load (0.1ms) SELECT "foods".* FROM "foods" WHERE "foods"."pet_id" = ? [{"pet_id", 3}]
Rendered pets/index.html.erb within layouts/application (9.7ms)
Completed 200 OK in 81ms (Views: 78.7ms | ActiveRecord: 0.7ms)
```


OUR APP

- Let's take a few minutes to add a couple pets!
- Try out the new, show, edit, delete actions.
- Watch the rails server window as you navigate pages/actions.



INTRO TO MODEL - VIEW - CONTROLLER



- Ruby on Rails is a MVC framework.

- Remember:

Ruby is a programming language.

Rails is a framework that extends Ruby to build web apps.

- MVC is a pattern/architecture that Rails follows to organize our code.

M→ Model

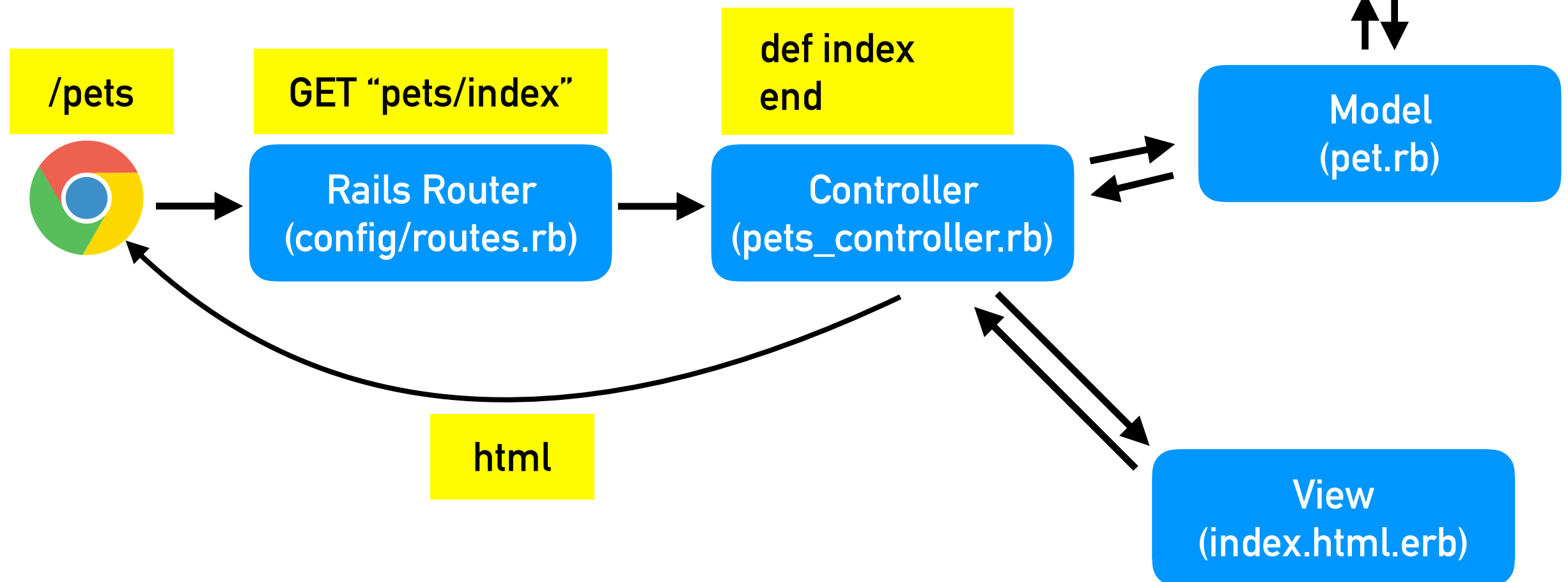
V→ View

C→ Controller

MVC OVERVIEW

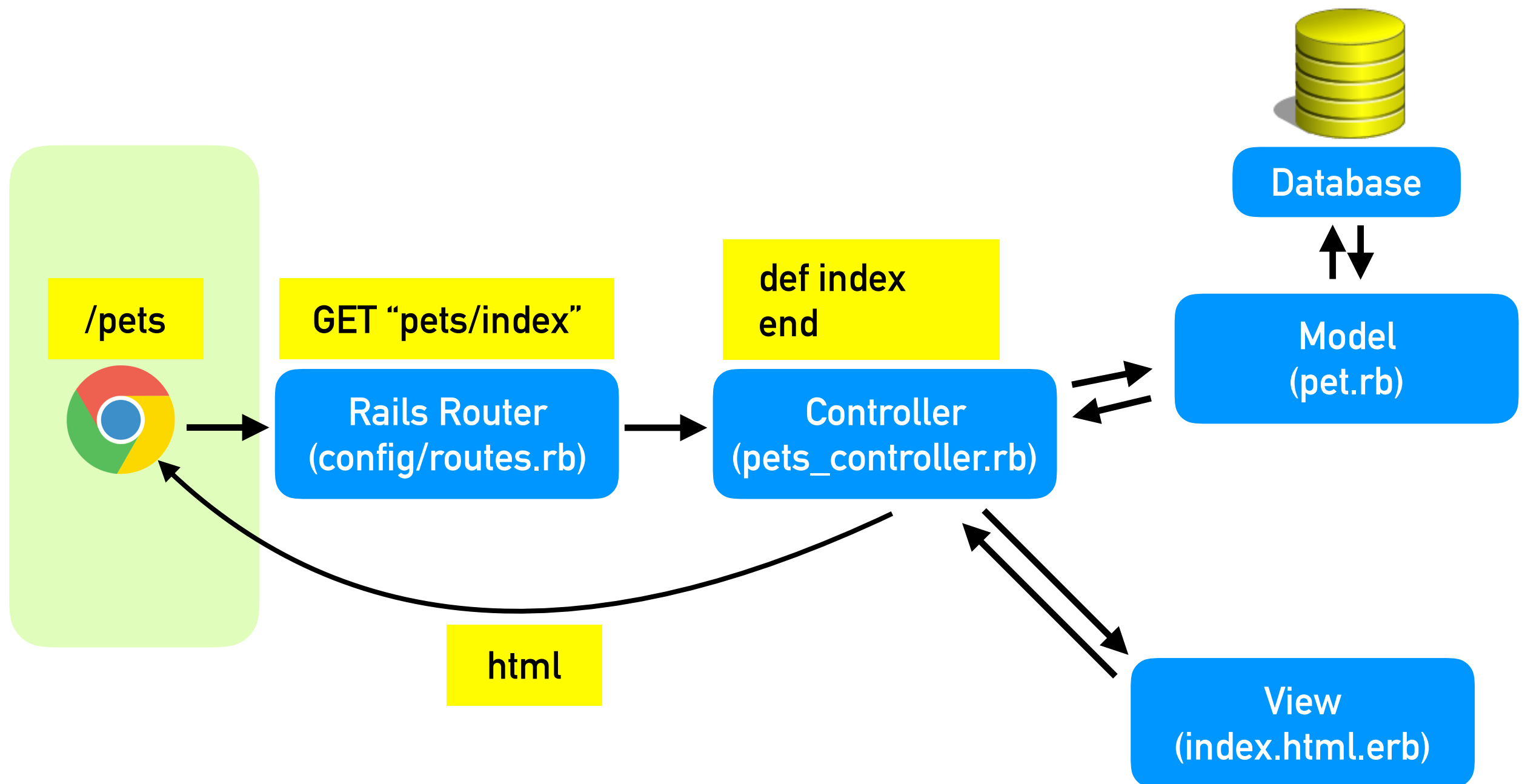
- Before we dive further into MVC....

It's OK not to understand this stuff!



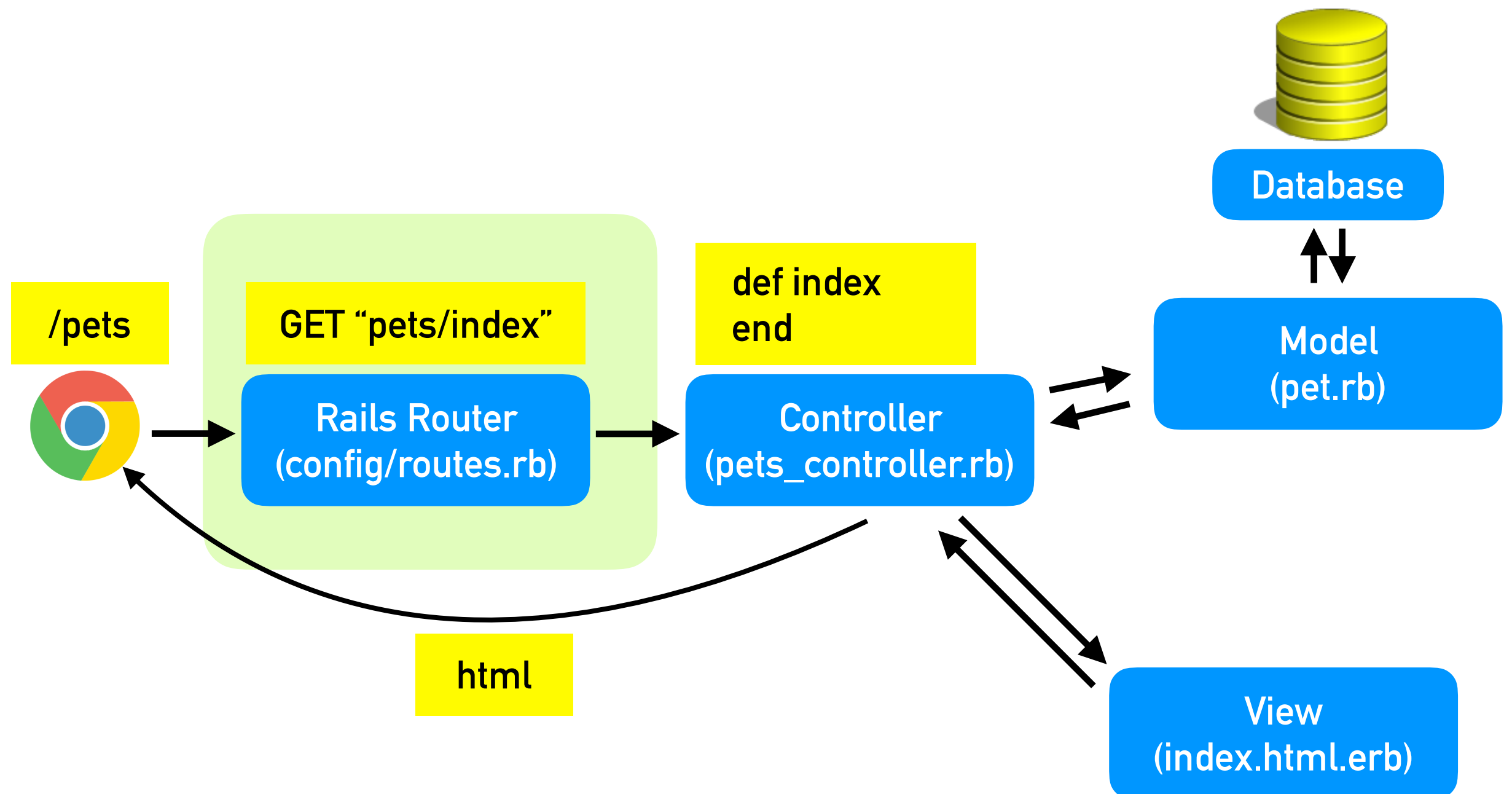
BROWSER REQUEST

- The browser makes a request: localhost:3000/pets/
- The web server (WEBrick) receives the request.



ROUTES

- It uses our `config/routes.rb` file to find out which controller to use.



ROUTES



- It uses our `config/routes.rb` file to find out which controller to use.

Rails Server:

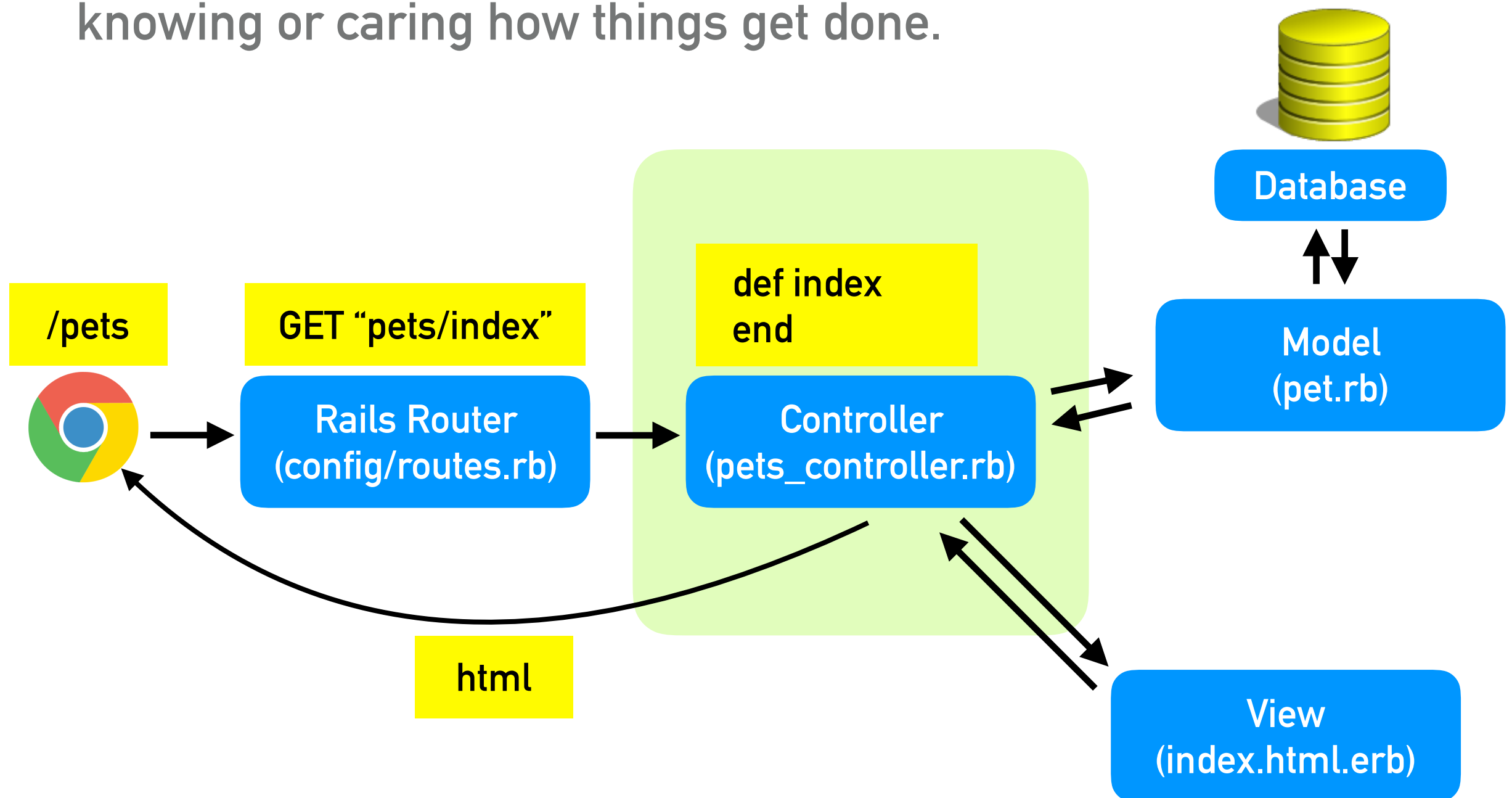
```
Started GET "/pets" for ::1 at 2015-12-05 20:32:38 +0100
Processing by PetsController#index as HTML
  Pet Load (0.1ms)  SELECT "pets".* FROM "pets"
  Food Load (0.1ms)  SELECT "foods".* FROM "foods" WHERE "foods"."pet_id" = ?  [["pet_id", 4]]
  Rendered pets/index.html.erb within layouts/application (1.8ms)
Completed 200 OK in 21ms (Views: 19.9ms | ActiveRecord: 0.2ms)
```

Terminal: \$ bin/rake routes

```
[→ railspet git:(master) bin/rake routes
      Prefix Verb   URI Pattern                                  Controller#Action
root GET          /                                              pets#index
pet_foods GET        /pets/:pet_id/foods/:format                foods#index
          POST       /pets/:pet_id/foods/:format                foods#create
new_pet_food GET        /pets/:pet_id/foods/new/:format            foods#new
edit_pet_food GET        /pets/:pet_id/foods/:id/edit/:format        foods#edit
pet_food GET        /pets/:pet_id/foods/:id/:format             foods#show
          PATCH     /pets/:pet_id/foods/:id/:format             foods#update
          PUT        /pets/:pet_id/foods/:id/:format             foods#update
          DELETE    /pets/:pet_id/foods/:id/:format             foods#destroy
pets GET         /pets/:format                               pets#index
          POST       /pets/:format                               pets#create
new_pet GET        /pets/new/:format                           pets#new
edit_pet GET        /pets/:id/edit/:format                       pets#edit
pet GET          /pets/:id/:format                           pets#show
          PATCH     /pets/:id/:format                           pets#update
          PUT        /pets/:id/:format                           pets#update
          DELETE    /pets/:id/:format                           pets#destroy
```

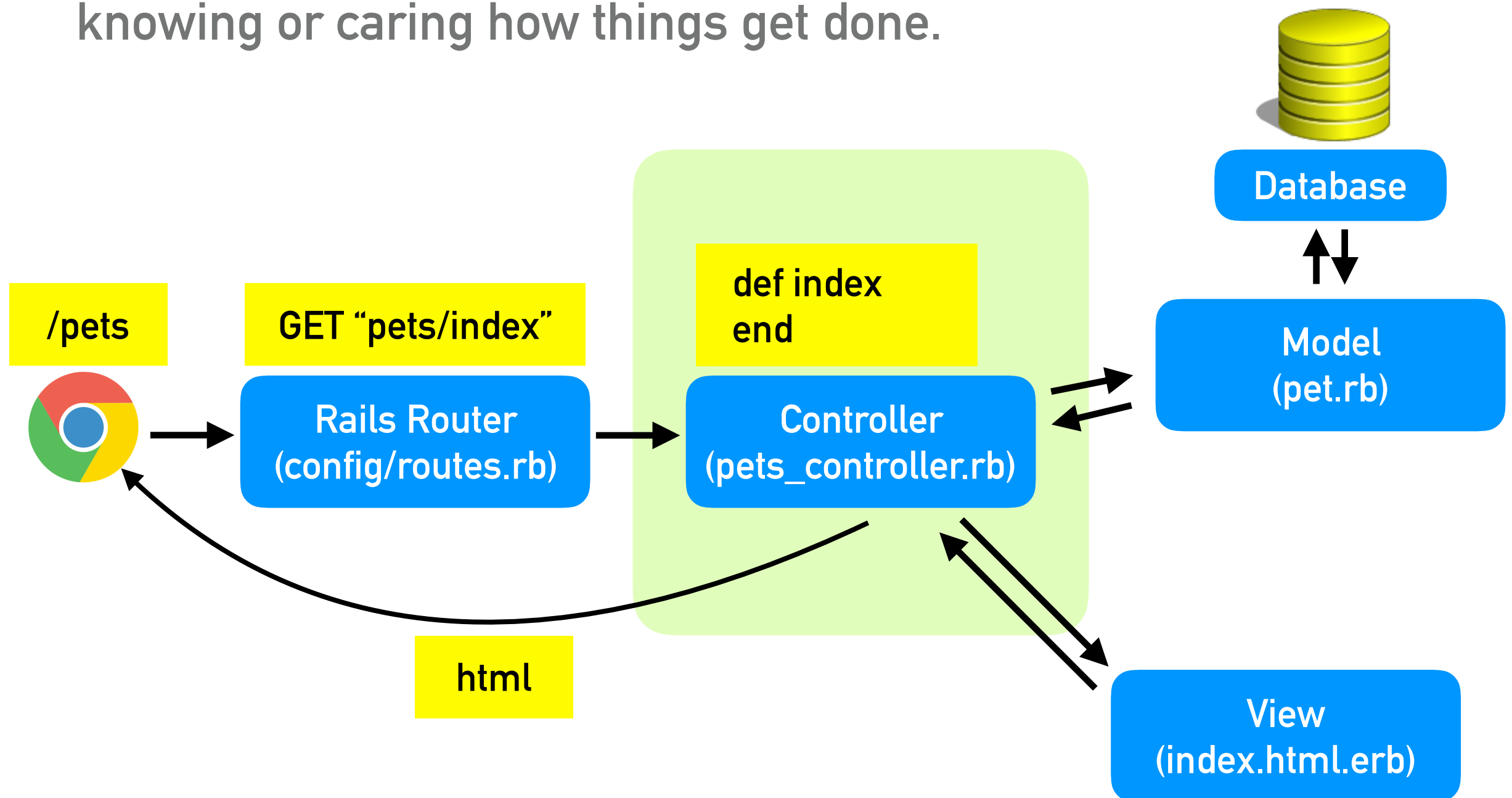

CONTROLLER

- Request ends up at the controller.
- Controllers are like the `managers`. They give orders without knowing or caring how things get done.



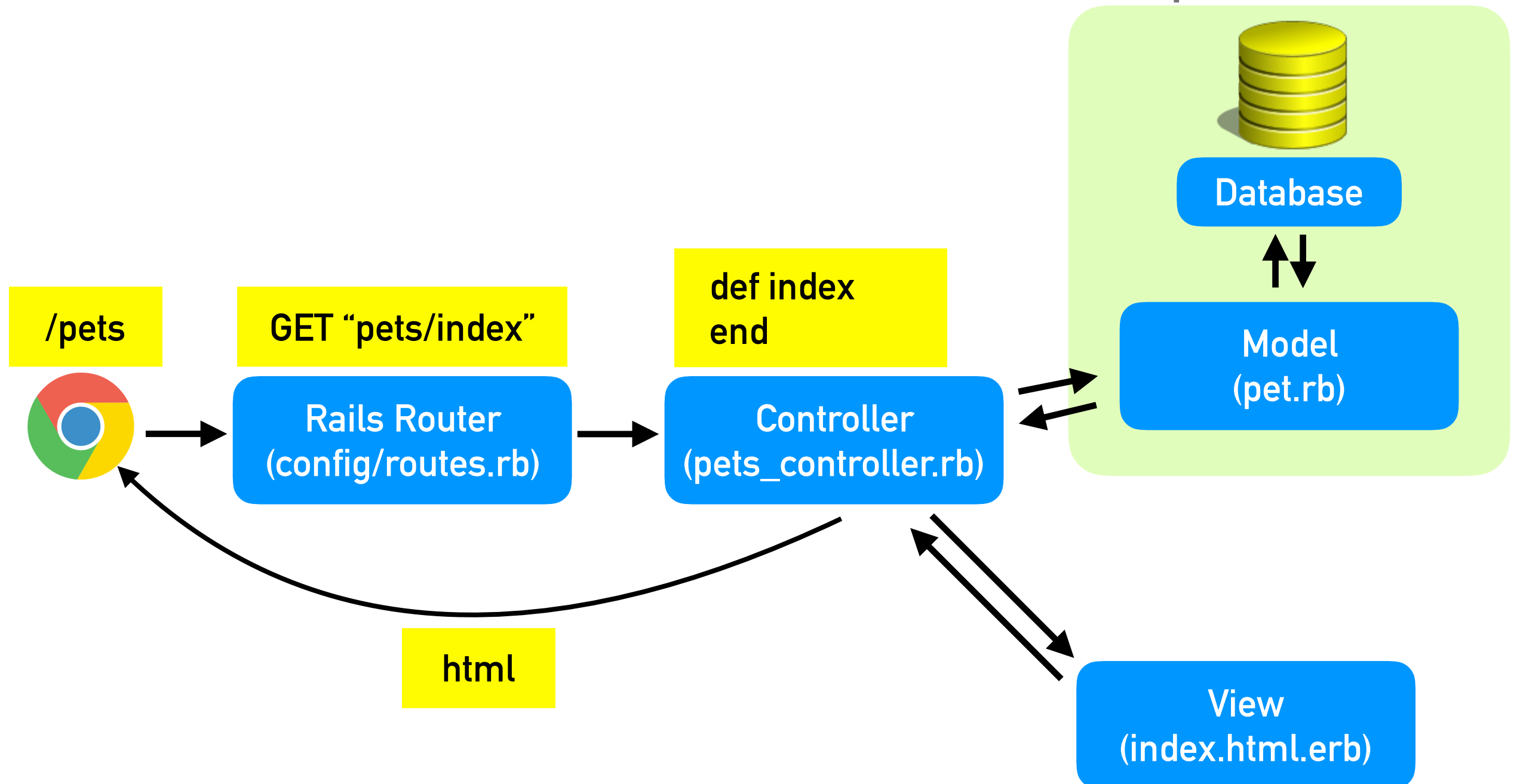
CONTROLLER

- Request ends up at the controller.
- Controllers are like the `managers`. They give orders without knowing or caring how things get done.



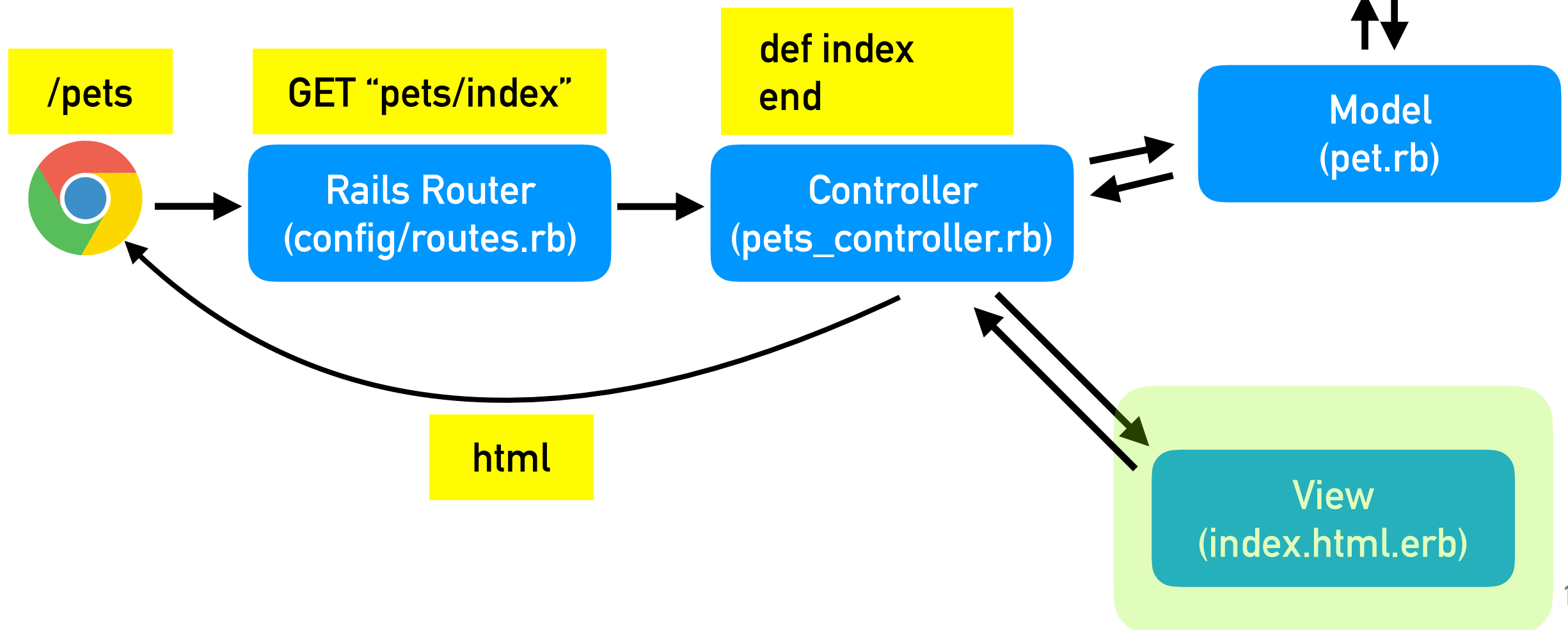
MODEL

- Model do the heavy lifting of the app.
- It talks to the database and retrieves the records requested.



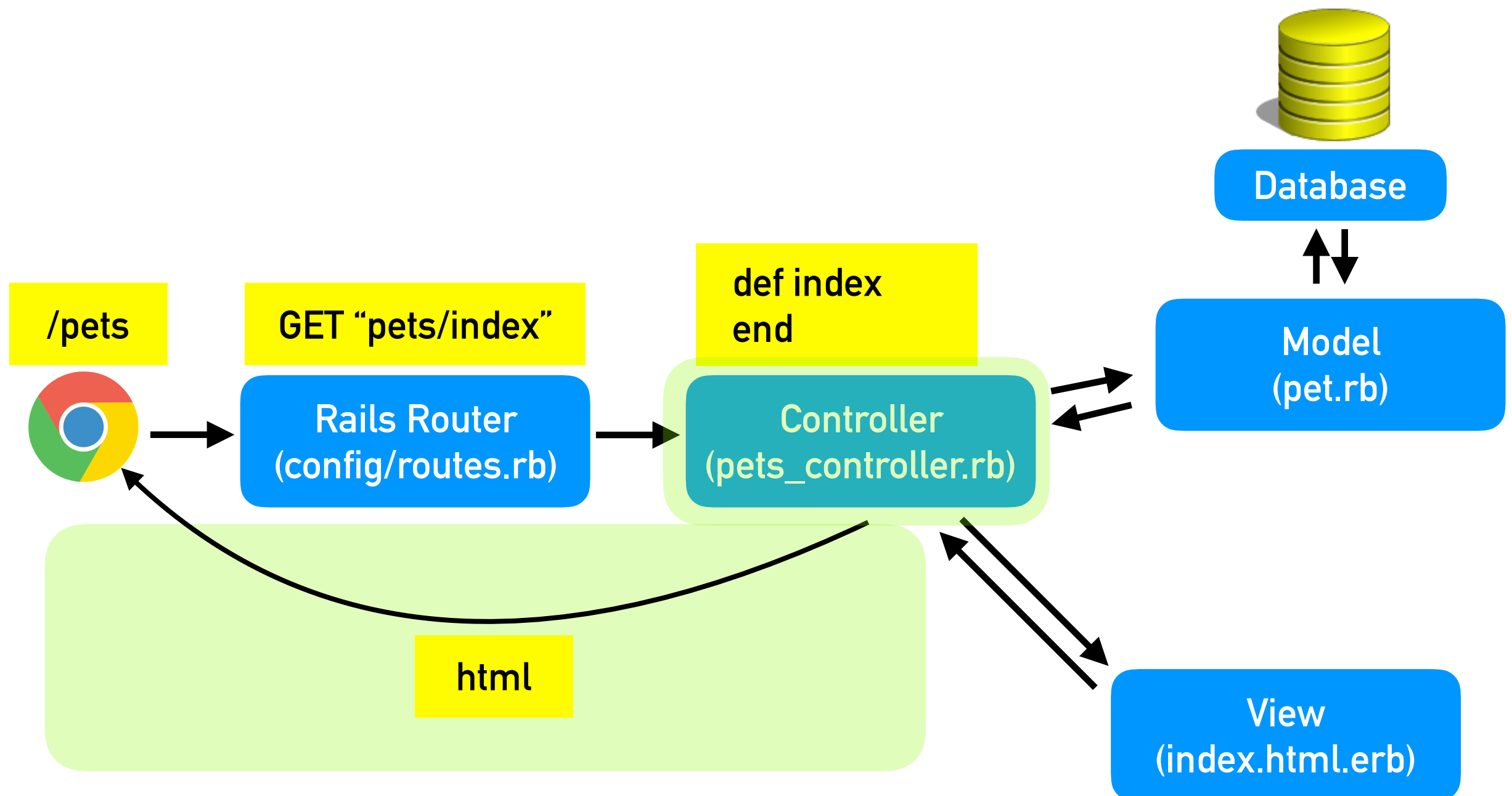
VIEW

- Views are what the user sees.
- They are the puppets reading back what the controller gives them.



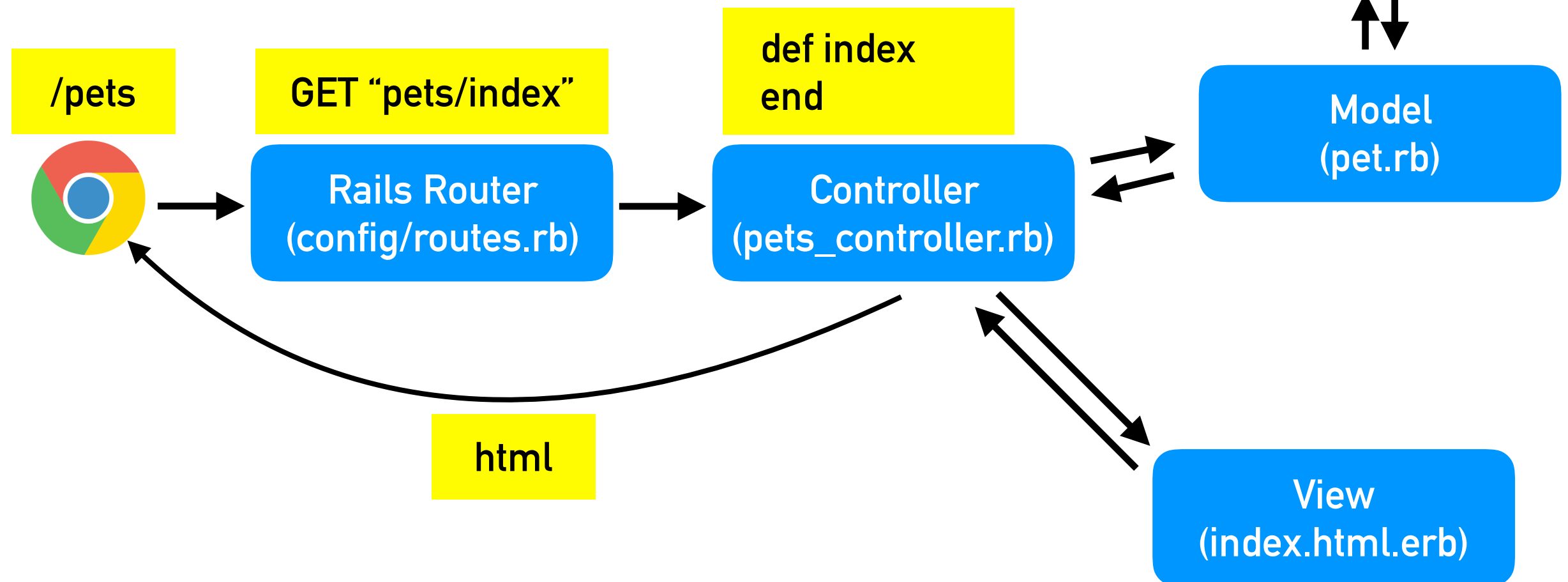
BACK TO THE CONTROLLER

- Finally the controller returns the response to the user.



MVC OVERVIEW

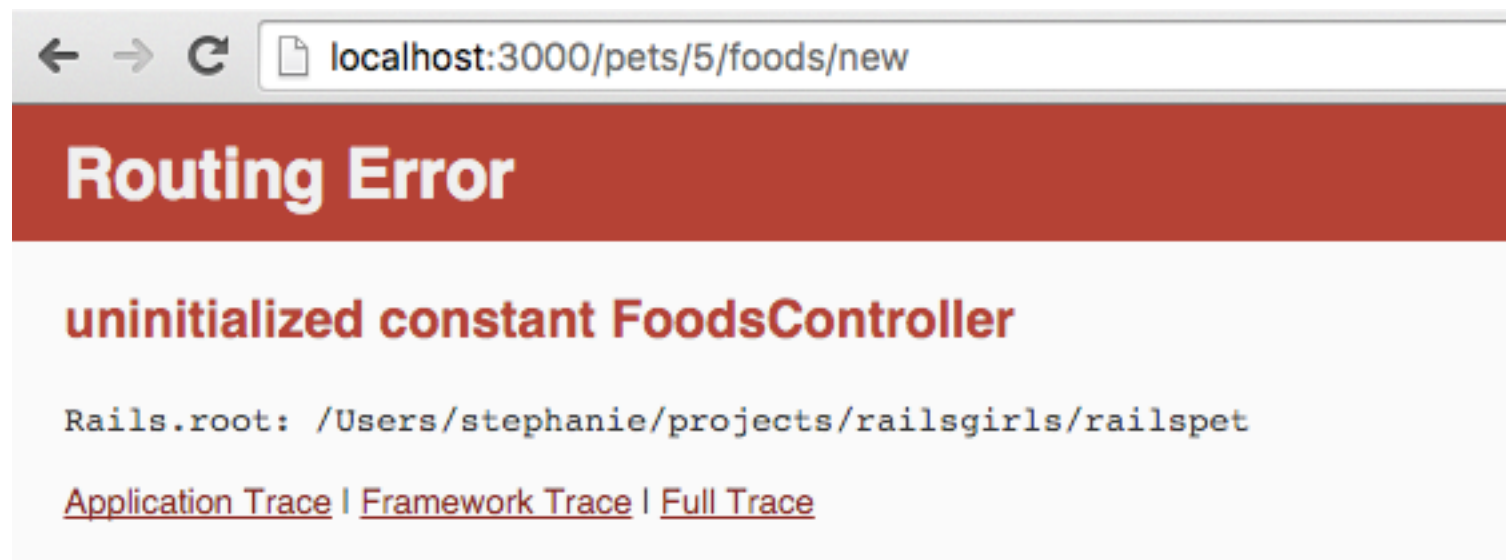
- Whew! Got all that?
- Don't sweat it if you don't.
- Let's practice in our Railspet app!



PRACTICE TIME!



- You may have noticed that not all of our app is working...



- Currently we can't feed our pets! oh nos!
- We are missing:
 - food_controller.rb
 - new & create actions in the controller
 - and a view: new.html.erb

def new
end

def
create

LET'S WRITE SOME CODE

- First, let's create a `foods_controller.rb` file
- Then, add new & create actions so we can add a food!

```
1 class FoodsController < ApplicationController
2
3   def new
4     @pet = Pet.find(params[:pet_id])
5     @food = Food.new
6   end
7
8   def create
9     @pet = Pet.find(params[:pet_id])
10    @food = @pet.foods.new(food_params)
11
12    if @food.save
13      redirect_to @pet
14    else
15      render 'new'
16    end
17  end
18
19  # Never trust parameters from the scary internet, only allow the white list through.
20  def food_params
21    params.require(:food).permit(:name, :calories)
22  end
23 end
24
```

LET'S WRITE SOME CODE



- Let's try to feed our pet again....
- Whoops! Still not working yet.

localhost:3000/pets/5/foods/new

Template is missing

Missing template foods/new, application/new with {:locale=>[:en], :formats=>[:html], :variants=>[], :handlers=>[:erb, :builder, :raw, :ruby, :coffee, :jbuilder]}. Searched in: *
"/Users/stephanie/projects/railsgirls/railspet/app/views"

Extracted source (around line #46):

```
44
45   def find(*args)
46     find_all(*args).first || raise(MissingTemplate.new(self, *args))
47   end
48
49   def find_all(path, prefixes = [], *args)
```

Rails.root: /Users/stephanie/projects/railsgirls/railspet

[Application Trace](#) | [Framework Trace](#) | [Full Trace](#)

actionview (4.2.4) lib/action_view/path_set.rb:46:in `find'
actionview (4.2.4) lib/action_view/lookup_context.rb:121:in `find'

LET'S WRITE SOME CODE

- Next, let's create a new.html.erb file in app/views/foods



```
railspet
├── app
│   ├── assets
│   ├── controllers
│   │   ├── application_controller.rb
│   │   ├── foods_controller.rb
│   │   └── pets_controller.rb
│   ├── helpers
│   ├── mailers
│   ├── models
│   ├── uploaders
│   └── views
│       └── foods
│           └── new.html.erb
├── layouts
├── pets
│   ├── edit.html.erb
│   ├── index.html.erb
│   ├── new.html.erb
│   └── show.html.erb
├── bin
├── config
│   ├── environments
│   ├── initializers
│   └── locales
```

```
1 <h1>New food</h1>
2
3 <%= form_for([@pet, @food]) do |f| %>
4   <% if @food.errors.any? %>
5     <div id="error_explanation">
6       <h2><%= pluralize(@food.errors.count, "error") %> prohibited this food from being saved:</h2>
7
8       <ul>
9         <% @food.errors.full_messages.each do |msg| %>
10          <li><%= msg %></li>
11        <% end %>
12      </ul>
13    </div>
14  <% end %>
15
16  <div class="field">
17    <%= f.label :name %><br>
18    <%= f.text_field :name %>
19  </div>
20  <div class="field">
21    <%= f.label :calories %><br>
22    <%= f.number_field :calories %>
23  </div>
24  <div class="actions">
25    <%= f.submit %>
26  </div>
27 <% end %>
28
29 <%= link_to 'Back', pet_path(@pet) %>
```

LET'S WRITE SOME CODE

- Let's try to feed our pet again....
- Success!



Pet name: wilbur



Picture:

Max weight: 11

Pet age: 0 days old

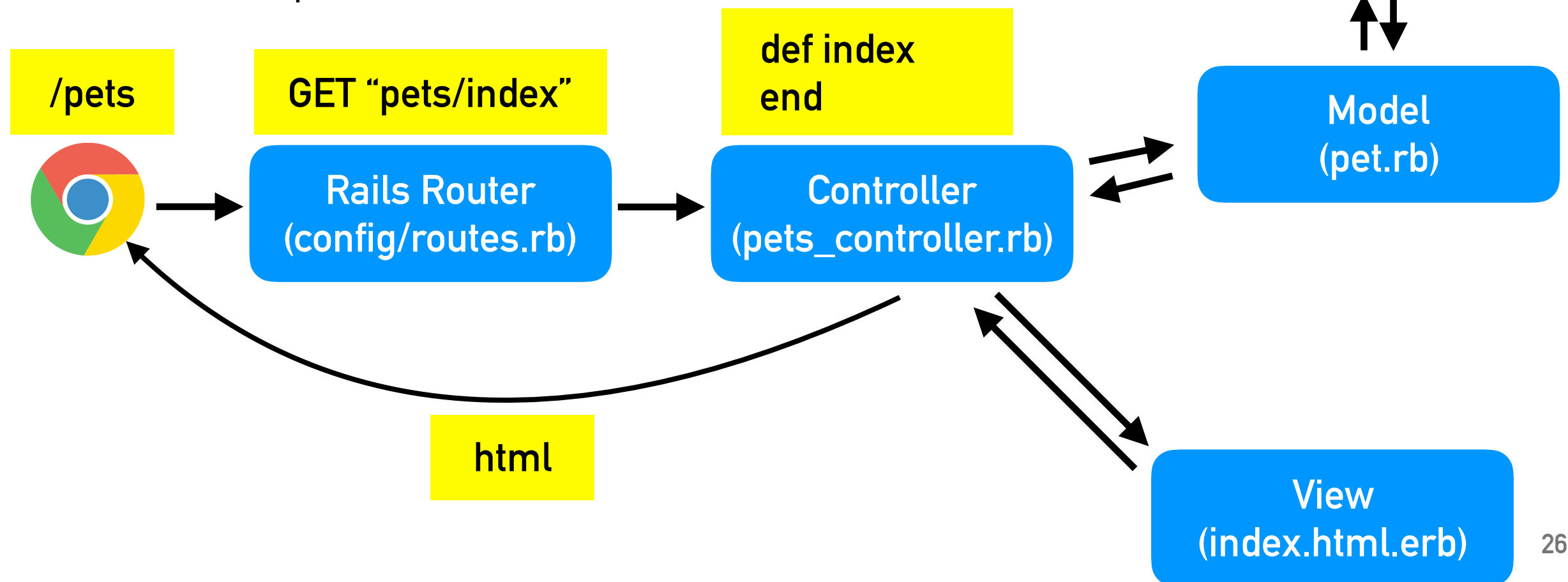
Eaten foods: chips

Pet full score: 2090% full

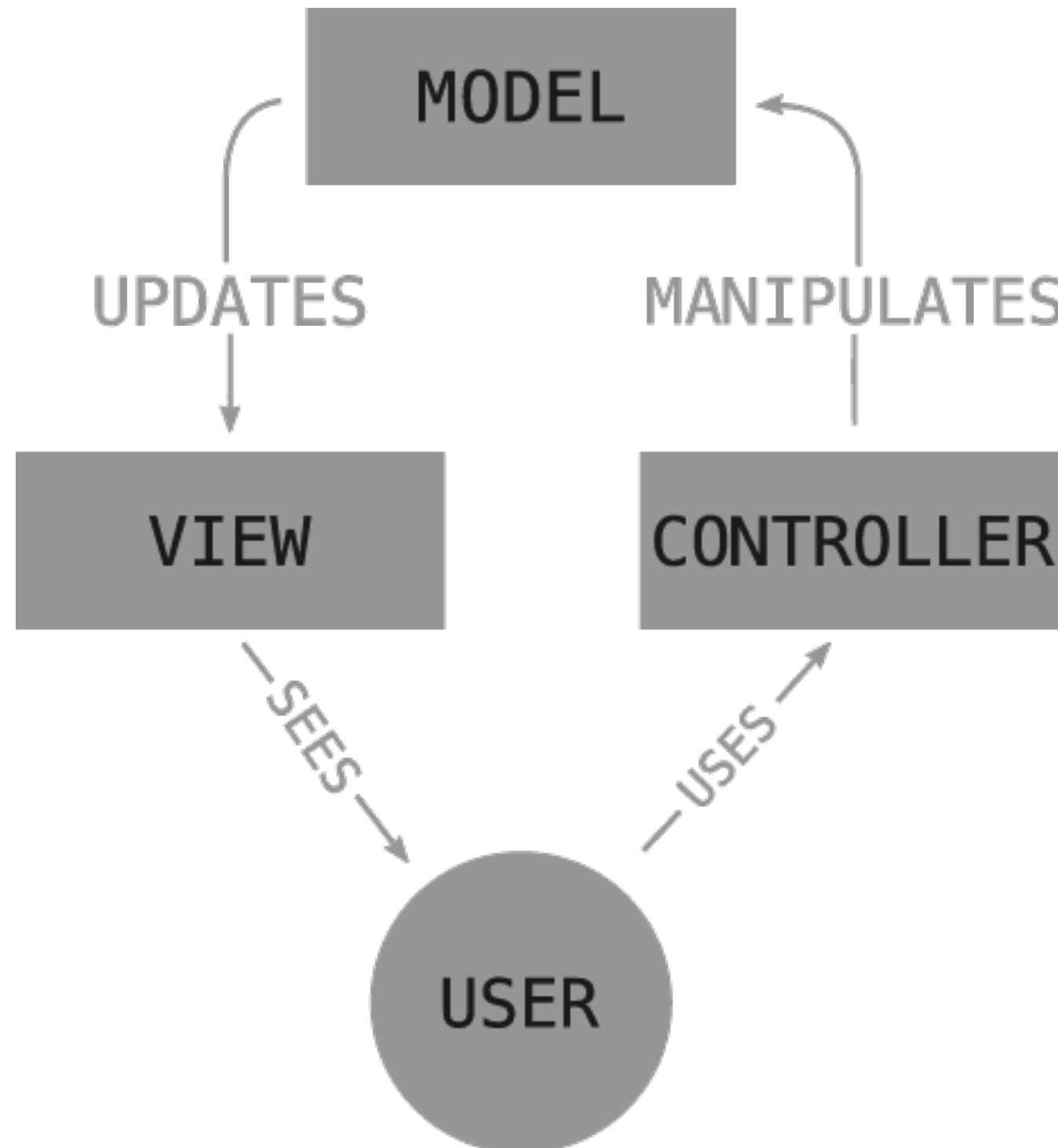
[Edit](#) | [Back](#) | [Feed your pet](#)

MVC RECAP

1. The browser issues a request for the /pets URL.
2. Rails routes /pets to the index action in the Pets controller.
3. The index action asks the Pet model to retrieve all pets (Pet.all).
4. The Pet model pulls all the pets from the database.
5. The Pet model returns the list of pets to the controller.
6. The controller captures the pets in the @pets variable, which is passed to the index view.
7. The view uses embedded Ruby to render the page as HTML.
8. The controller passes the HTML back to the browser.



MVC RECAP



INTRO TO CSS

- CSS = Cascading Style Sheet
- First, quick a HTML review...



HTML REVIEW

- Basic Format:



```
rails pet
├── app
│   ├── assets
│   ├── controllers
│   ├── helpers
│   ├── mailers
│   ├── models
│   ├── uploaders
│   └── views
│       ├── foods
│       └── layouts
│           └── application.html.erb
├── pets
├── bin
└── config
```

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Rails pet</title>
5   <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track' => true %>
6   <%= javascript_include_tag 'application', 'data-turbolinks-track' => true %>
7   <%= csrf_meta_tags %>
8 </head>
9 <body>
10
11 <%= yield %>
12
13 </body>
14 </html>
15
```

HTML ELEMENTS



```
1  <div></div>
2  <h1></h1> to <h6></h6>
3  <p></p>
4  <img src=""/>
5  <a href="#"></a>
6  <br>
7
8  <ul>
9      <li></li>
10     <li></li>
11 </ul>
```

Full list on:

<http://www.w3schools.com/tags>

HTML ATTRIBUTES



```
1 <div class="greeting"></div>
2 <h1 id="test"></h1> to <h6></h6>
3 <p></p>
4 
5 <a href="url"></a>|
```

Full list on:

<https://developer.mozilla.org/en-US/docs/Web/HTML/Attributes>

CSS BASIC FORMAT



- Element
- Id (#id)
- Class (.class)

```
1 <h1 id="greeting">Hi There!</h1>
2
3
4 <div class="about-me">
5   <p>
6     Blah blah blah
7   </p>
8 </div>
```

```
#greeting {
  color: blue;
}

.about-me {
  padding: 20px;
}

p {
  font-size: 24px;
}
```


MORE CSS STYLING



```
1  div {  
2    height: 50px;  
3    width: 100px;  
4    display: inline-block;  
5    float: left;  
6    background-color: red;  
7    color: blue;  
8    margin: 10px 2px 5px 3px;  
9    padding: 5px 10px;  
10 }
```

OUR RAILSPET APP- CSS IN ACTION

- Make a new file: `custom.css` in app/assets/stylesheets



```
rails pet
├── app
│   ├── assets
│   │   ├── images
│   │   ├── javascripts
│   │   └── stylesheets
│   │       ├── application.css
│   │       └── custom.css
│   ├── controllers
│   ├── helpers
│   ├── mailers
│   ├── models
│   ├── uploaders
│   └── views
│       ├── foods
│       ├── layouts
│       │   ├── application.html.erb
│       │   └── pets
│       │       ├── edit.html.erb
│       │       ├── index.html.erb
│       │       ├── new.html.erb
│       │       └── show.html.erb
│       ├── bin
│       ├── config
│       ├── db
│       ├── lib
│       ├── log
│       └── public
```

```
1 h1 {
2   text-decoration: underline;
3   color: red;
4 }
5
```

```
1 <h1>My Pets</h1>
2
3 <table>
4   <thead>
5     <tr>
6       <th></th>
7       <th>Name</th>
8       <th>Max weight</th>
9       <th>Foods Eaten</th>
10      <th colspan="3"></th>
11    </tr>
12  </thead>
13
14  <tbody>
15    <% @pets.each do |pet| %>
16      <tr>
17        <td><%= image_tag(pet.picture_url, width: 200) if pet.picture.present? %></td>
18        <td><%= pet.name %></td>
19        <td><%= pet.max_weight %> kg</td>
20        <td><%= pet.eaten_foods %></td>
21        <td><%= link_to 'Show', pet %></td>
22        <td><%= link_to 'Edit', edit_pet_path(pet) %></td>
23        <td><%= link_to 'Delete', pet, method: :delete, data: { confirm: 'Are you sure?' } %></td>
24      </tr>
25    <% end %>
26  </tbody>
27 </table>
28
29 <br>
30
31 <%= link_to 'New Pet', new_pet_path %>
```

CSS - TRY IT YOURSELF!



- You can add classes and ids to your views.
- And then add CSS to the ``app/assets/stylesheets/custom.css`` file.

CSS Properties: <http://www.w3schools.com/cssref/default.asp>

BOOTSTRAPPING OUR APP

- Bootstrap is a front-end framework from Twitter that makes it easy to add nice web design and user interface elements to an application.
- It's worth noting that using Bootstrap automatically makes our application's design responsive, ensuring that it looks good across a wide range of devices.
- More Info: <http://getbootstrap.com/>



ADD THE BOOTSTRAP GEM



- Add gem 'bootstrap-sass' to your Gemfile.
- In the terminal, \$ bundle install

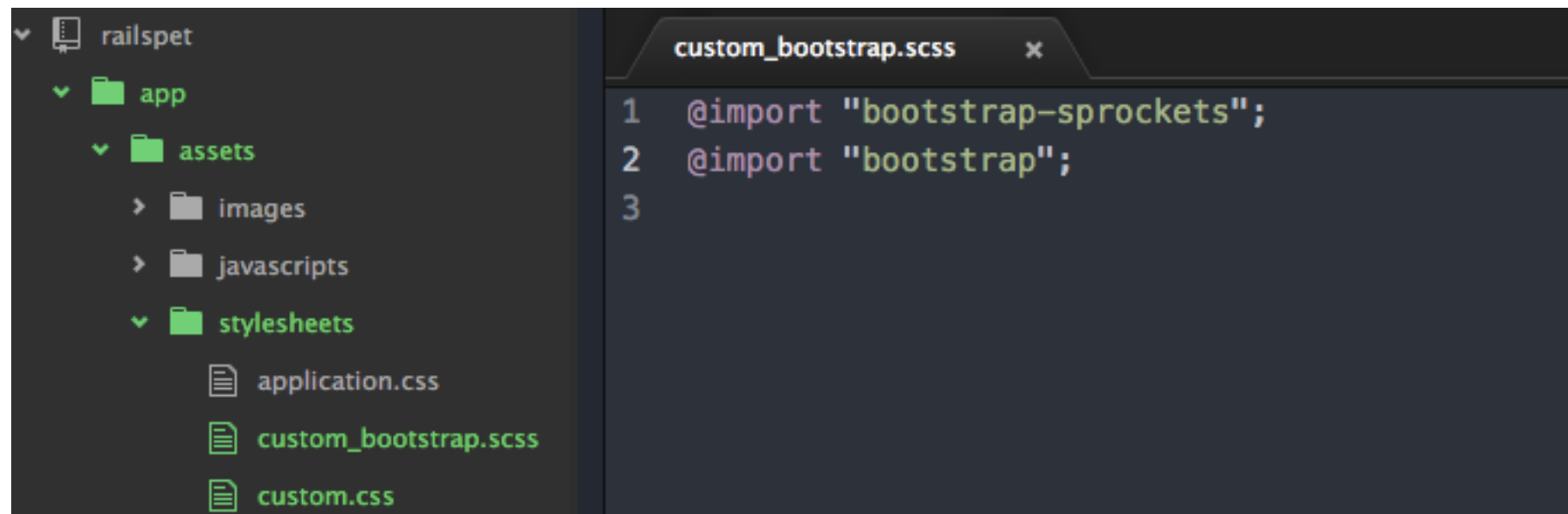
```
Gemfile x
1  source 'https://rubygems.org'
2
3  gem 'rails', '4.2.4'
4  gem 'sqlite3'
5  gem 'sass-rails', '~> 5.0'
6  gem 'uglifier', '>= 1.3.0'
7  gem 'coffee-rails', '~> 4.1.0'
8  gem 'jquery-rails'
9  gem 'turbolinks'
10 gem 'jbuilder', '~> 2.0'
11 gem 'sdoc', '~> 0.4.0', group: :doc
12
13 gem 'carrierwave'
14
15 # Styling
16 gem 'bootstrap-sass'
17
```

UNDERSTAND THE APPLICATION.CSS FILE



app/assets/stylesheets/application.css

- Takes all the other files in your /stylesheets directory and combines them for when you run your app.
- Let's create a new stylesheet file: `custom_bootstrap.scss`



BOOTSTRAP MAGIC!



- Bootstrap comes pre-built with things like buttons, navigation bars, icons, that you can use to make your web app look really nice, really fast!
- First, let's clean up the view a little, by adding a container class in our `views/layouts/application.html.erb` file:

```
application.html.erb x
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>Railspet</title>
5    <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track' => true %>
6    <%= javascript_include_tag 'application', 'data-turbolinks-track' => true %>
7    <%= csrf_meta_tags %>
8  </head>
9  <body>
10    <div class="container">
11      <%= yield %>
12    </div>
13
14  </body>
15  </html>
```




MORE BOOTSTRAP IMPROVEMENTS



- Our table on the pets#index page could use some style too.

← → ↻ 📄 localhost:3000/pets
Reload this page

My Pets

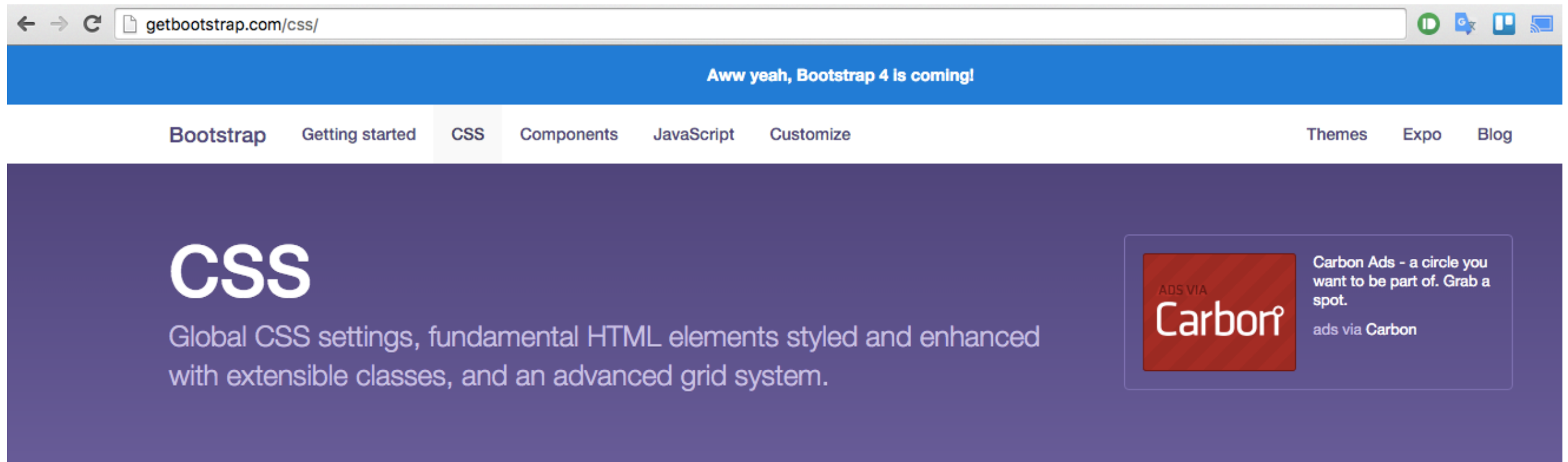
	Name	Max weight	Foods Eaten	
	Wilbur	11 kg	chips	Show Edit Delete
	Dasher	35 kg	bananas	Show Edit Delete

[New Pet](#)

MORE BOOTSTRAP IMPROVEMENTS



- Check out: <http://getbootstrap.com/css/> for guides to CSS.

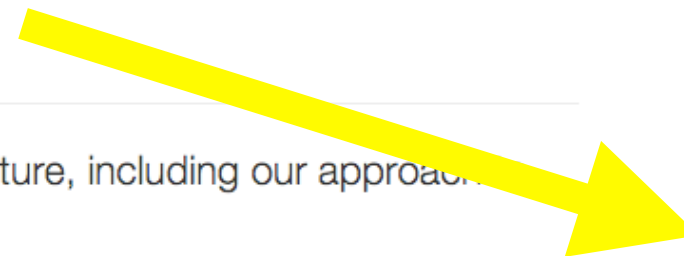


Overview

Get the lowdown on the key pieces of Bootstrap's infrastructure, including our approach to better, faster, stronger web development.

HTML5 doctype

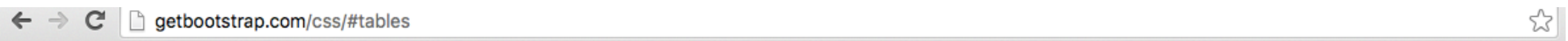
Overview
Grid system
Typography
Code
Tables
Forms
Buttons



BOOTSTRAP - TABLE CSS



- Check out: <http://getbootstrap.com/css/> for guides to CSS.



Tables

Basic example

For basic styling—light padding and only horizontal dividers—add the base class `.table` to any `<table>`. It may seem super redundant, but given the widespread use of tables for other plugins like calendars and date pickers, we've opted to isolate our custom table styles.

EXAMPLE

Optional table caption.

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

```
<table class="table">
  ...
</table>
```

Copy

- Overview
- Grid system
- Typography
- Code
- Tables**
 - Basic example
 - Striped rows
 - Bordered table
 - Hover rows
 - Condensed table
 - Contextual classes
 - Responsive tables
- Forms
- Buttons
- Images
- Helper classes
- Responsive utilities
- Using Less
- Using Sass
- Back to top
- Preview theme

BOOTSTRAP - TABLE CSS

- Add the class to the table element in the `index.html.erb` file.





```
1 <h1>My Pets</h1>
2
3
4 <table class="table table-hover">
5   <thead>
6     <tr>
7       <th></th>
8       <th>Name</th>
9       <th>Max weight</th>
10      <th>Foods Eaten</th>
11      <th colspan="3"></th>
12    </tr>
13  </thead>
14
15  <tbody>
16    <% @pets.each do |pet| %>
17      <tr>
18        <td><%= image_tag(pet.picture_url, width: 200) if pet.picture.present? %></td>
19        <td><%= pet.name %></td>
20        <td><%= pet.max_weight %> kg</td>
21        <td><%= pet.eaten_foods %></td>
22        <td><%= link_to 'Show', pet %></td>
23        <td><%= link_to 'Edit', edit_pet_path(pet) %></td>
24        <td><%= link_to 'Delete', pet, method: :delete, data: { confirm: 'Are you sure?' } %></td>
25      </tr>
26    <% end %>
27  </tbody>
28 </table>
```

VOILA! BOOTSTRAP MAGIC!



My Pets

	Name	Max weight	Foods Eaten			
	wilbur	11 kg	chips	Show	Edit	Delete
	Dasher	35 kg	bananas	Show	Edit	Delete

[New Pet](#)

BOOTSTRAP - PRACTICE TIME

- Check out: <http://getbootstrap.com/css/>
- Play around and explore bootstrap and css yourself!



THAT'S IT! YOU MADE IT!

- Hope you've had fun and learned some stuff!
- Stay in contact! Questions? Suggestions?
- Thanks for coming!

