Authentication / Authorization II

Sessions Controller

sessions_controller.rb

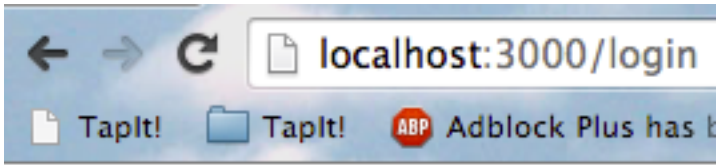
```ruby
class SessionsController < ApplicationController
  def new
    # Nothing to do here - go on to new.html.erb
    # (Login page)
  end

  def create
    name = params[:reviewer][:name]
    password = params[:reviewer][:password]

    reviewer = Reviewer.find_by name: name
    if(reviewer && reviewer.authenticate(password))
      session[:reviewer_id] = reviewer.id
      redirect_to root_path, notice: "Logged in successfully"
    else
      flash.now[:alert] = "Invalid username/password combination"
      render action: "new"
    end
  end

  def destroy
    reset_session
    redirect_to login_path, notice: "You have been logged out"
  end
end
```

views/sessions/new.html.erb

new.html.erb

```
<h1>Login</h1>

<%= form_for(:reviewer, url: sessions_path) do |f| %>

  <div class="field">
    <%= f.label :name %><br/>
    <%= f.text_field :name %>
  </div>

  <p/>

  <div class="field">
    <%= f.label :password %><br/>
    <%= f.password_field :password %>
  </div>

  <div class="actions">
    <%= f.submit "Login" %>
  </div>

<% end %>
```

Login page



Login

Name

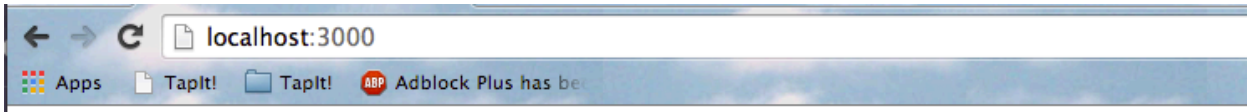
Joe

Password

Login

```
<form accept-charset="UTF-8" action="/sessions" method="post"><div
style="display:none"><input name="utf8" type="hidden"
value="&#x2713;" /><input name="authenticity_token" type="hidden"
value="rb+lfzDJkSPsCiCY414+B1kfwdvQt/BwmhXXF5XIPDw=" /></div>

  <div class="field">
    <label for="reviewer_name">Name</label><br/>
    <input id="reviewer_name" name="reviewer[name]" type="text" />
  </div>

  <p/>

  <div class="field">
    <label for="reviewer_password">Password</label><br/>
    <input id="reviewer_password" name="reviewer[password]"
type="password" />
  </div>

  <div class="actions">
    <input name="commit" type="submit" value="Login" />
  </div>

</form>
```

Logged In



Logged in successfully

Books

My way or highway (Me) Edit Remove

The Rails Way (The Other Guy) Edit Remove

The Starving Students' Cookbook (Dede Hall) Edit Remove

The Vegetarian Family Cookbook (Nava Atlas) Edit Remove

Ember.js in Action (Joachim Haagen Skeie) Edit Remove

New Book

Cookie in the browser

localhost

1 cookie

_i_reviewed_session

Name:	_i_reviewed_session
Content:	dno0dHFacUpETVNWQ2JmZlhlMnhwaXVreElicjFnM0hBdmlUR1FWUEt3Ykw5Um5kalJtaDlQK09COFFyN0JUVk9HQzA4eGcvQ0t0LytqNHBjRXRsdjVLSlNCUU1yb2Mvd0ZNdTZLRUpHQzBjR3pkeFdDeUFseW9CRmltUmk5aFRrZWpTTHgzOVZsMzAvVlhQbVBnV21jOXhoWUVMSGZFNXE3aC9CUzdTTUNZOWkxTVh4V3FWZzEzZFlickY4djN4SXA5b0ZEN0E3THlWNVVZWkpmQWhrNGdhL21Fd1pGZTVNOGplcXVDYVl2NnAvQVVZdHgzMWVSNk1lVmZjQkFOcWZKNENtMXd1RzAzMTlmR250VGN1cEE9PS0tRG9ES1ZWc0NxbzEzNGlPVTNXTU0zUT09--88f02fd8829dc1d6e25a70b5981396c76540711c
Domain:	localhost
Path:	/
Send for:	Any kind of connection
Accessible to script:	No (HttpOnly)
Created:	Sunday, July 27, 2014 12:05:55 PM

Locking down the app

- We can have a `before_action` in the `ApplicationController` (from which all the other controllers inherit) that will make you login if you are not yet logged in
- But if everything is blocked off – how will we get to the login page? Hmm…
- Controllers can override `before_action` with `skip_before_action`

application_controller.rb

application_controller.rb

```ruby
class ApplicationController < ActionController::Base
  # Prevent CSRF attacks by raising an exception.
  # For APIs, you may want to use :null_session instead.
  protect_from_forgery with: :exception

  before_action :ensure_login

  protected
    def ensure_login
      redirect_to login_path unless session[:reviewer_id]
    end
end
```

sessions_controller.rb

sessions_controller.rb

```ruby
class SessionsController < ApplicationController
  skip_before_action :ensure_login,
    only: [:new, :create, :destroy]

  def new
  end

  def create
  end

  def destroy
    reset_session
    redirect_to login_path, notice: "You have been logged out"
  end
end
```

Logging out

- Let's add `logged_in?` and `current_user` methods to `ApplicationController` and make them available as helper methods to all controllers and views via `helper_method`
- (Adding them to `ApplicationHelper` would only make them available to the views, but not to controllers)
- Then, we can add logic to `application.html.erb` for logging out

application_controller.rb

application_controller.rb

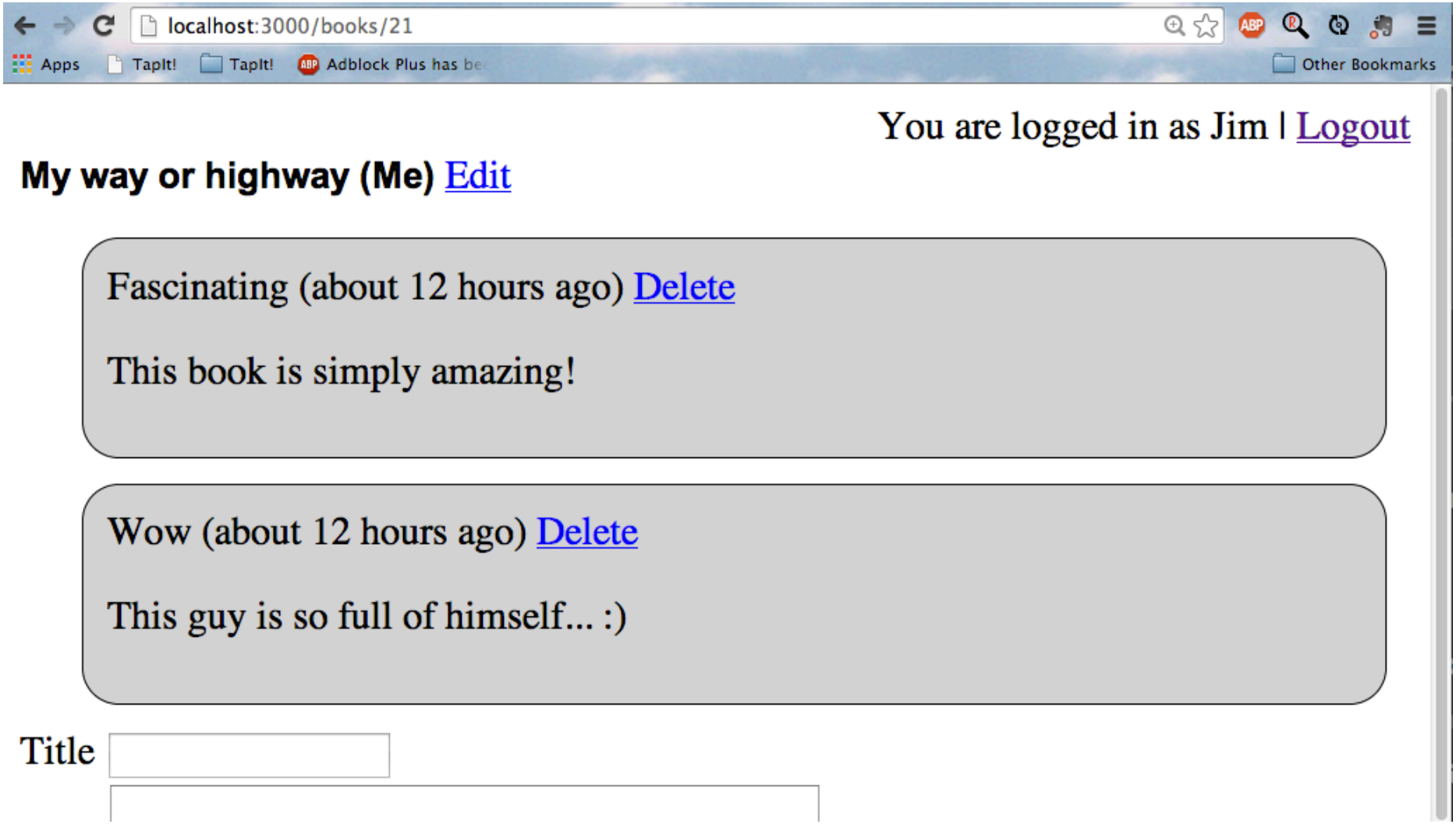
```ruby
class ApplicationController < ActionController::Base
  protect_from_forgery with: :exception

  before_action :ensure_login

  protected
    def ensure_login
      redirect_to login_path unless logged_in?
    end

    def logged_in?
      session[:reviewer_id]
    end

    def current_user
      @current_user ||= Reviewer.find(session[:reviewer_id])
    end

    helper_method :logged_in?, :current_user
end
```

views/layouts/application.html.erb

application.html.erb

```
<!DOCTYPE html>
<html>
<head>
  <title>IReviewed</title>
  <%= stylesheet_link_tag    'application',
    media: 'all', 'data-turbolinks-track' => true %>
  <%= javascript_include_tag 'application',
    'data-turbolinks-track' => true %>
  <%= csrf_meta_tags %>
</head>
<body>

<% if logged_in? %>
  <div style='float: right;'>
    You are logged in as <%= current_user.name %> |
    <%= link_to "Logout", logout_path, data: {method: :delete} %>
  </div>
<% end %>

<% flash.each do |key, value| %>
  <%= content_tag :p, value, id: key %>
<% end %>

<%= yield %>

</body>
</html>
```

views/layouts/application.html.erb



Authorization

- We have implemented basic *Authentication*, but this still does nothing for our *Authorization*
- Anybody who logs into the system can edit anyone else's books and notes?!
- **SOLUTION:** We can go back to the `BooksController` and scope things down based on the `current_user`

Authorization – index, new, create

```ruby
def index
  @books = current_user.books.all
end

def new
  @book = current_user.books.new
end

def create
  @book = current_user.books.new(book_params)

  respond_to do |format|
  end
end
```

Authorization – the other actions

```
def show
end

def edit
end

def update
end

def destroy
end

private
  #  before_action :set_book, only: [:show, :edit, :update, :destroy]
  def set_book
    @book = current_user.books.find(params[:id])
  end
```

Authorization

Books

You are logged in as Jim | Logout

The Rails Way (The Other Guy) Edit Remove

The Starving Students' Cookbook (Dede Hall) Edit Remove

The Vegetarian Family Cookbook (Nava Atlas) Edit Remove

New Book

Authorization

Books

You are logged in as Joe | Logout

My way or highway (Me) Edit Remove

Ember.js in Action (Joachim Haagen Skeie) Edit Remove

New Book