



# Authentication / Authorization I

# What happened to the users?

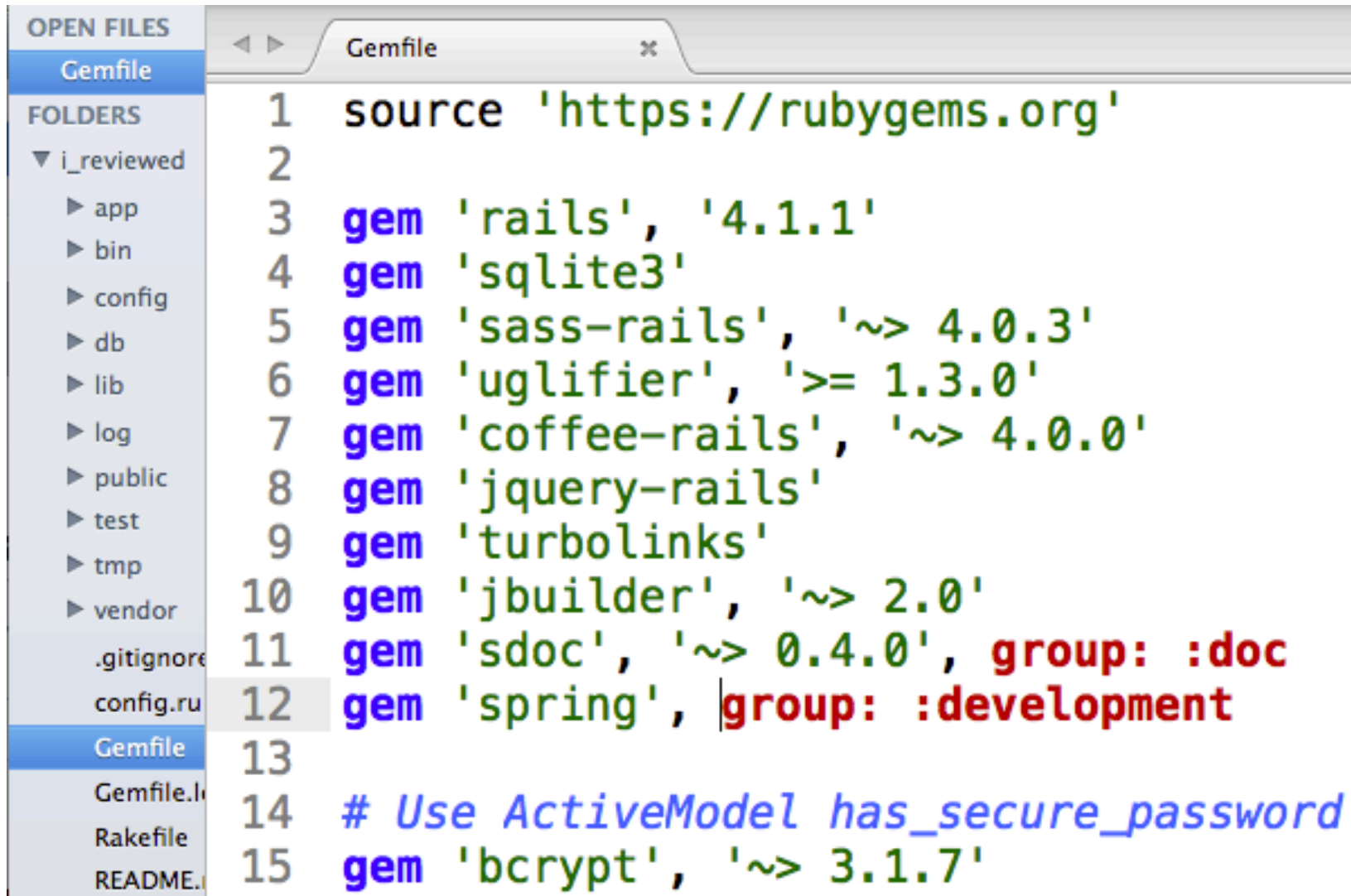
- The books/notes stuff is working pretty nicely, but everybody can see and modify everybody else's books and notes?
- We need the 2 security 'A's:
  - Authentication
    - Do we know who the user is and are his **credentials valid**?
  - Authorization
    - Provide access only to the books/notes a particular logged-in user is **authorized** to modify/see

# Authentication Summary

has\_secure\_password to the rescue

1. Enable (uncomment) bcrypt-ruby (Gemfile)
  - Run `$bundle (install)`
2. Make sure `password_digest` is a column
3. Account for `password` (not `password_digest`) inside strong parameters list in the model if you plan to use mass assignment when creating users
4. No need for `password` column in the table (virtual attribute)

# bcrypt-ruby

`$bundle`

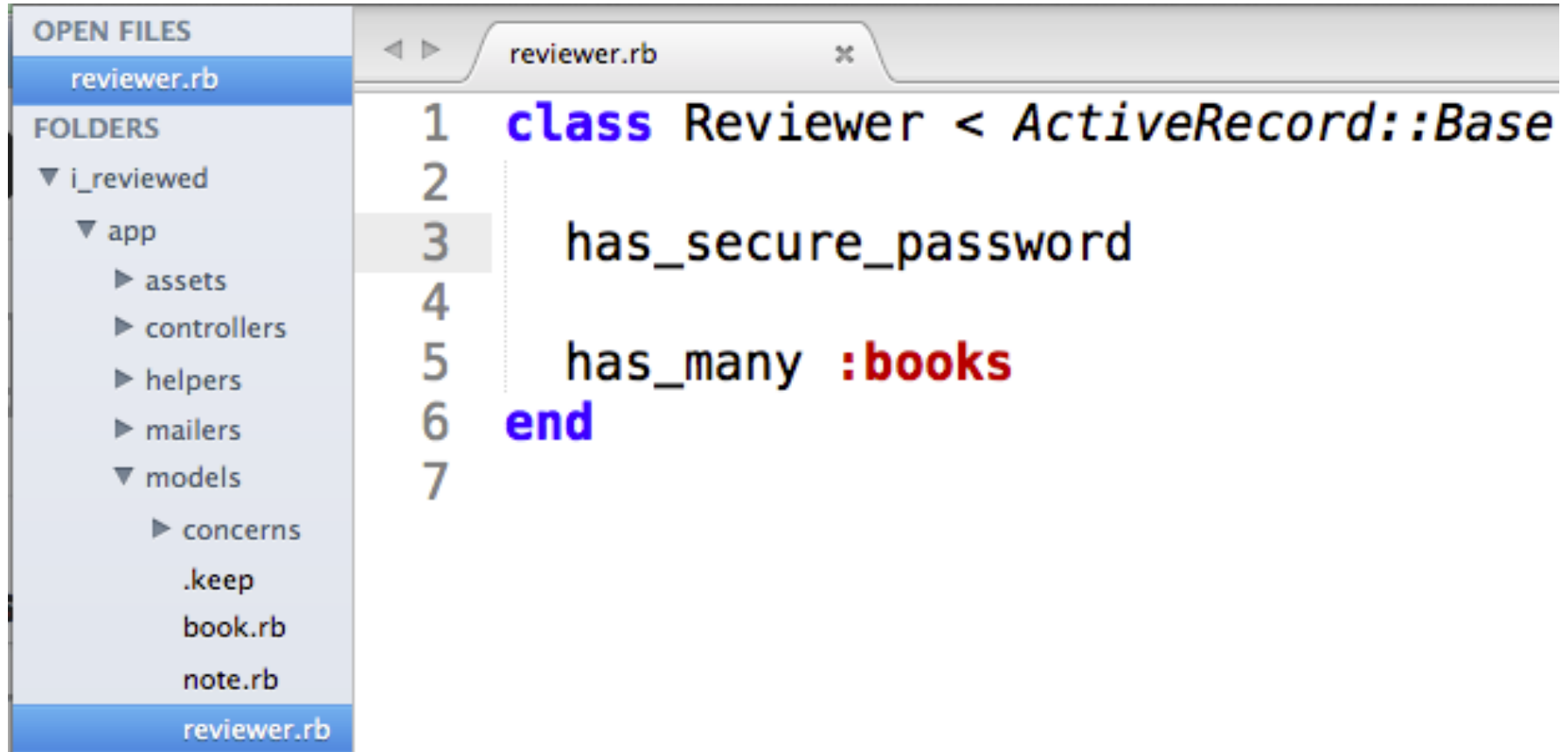
```
OPEN FILES
Gemfile

FOLDERS
▼ i_reviewed
  ► app
  ► bin
  ► config
  ► db
  ► lib
  ► log
  ► public
  ► test
  ► tmp
  ► vendor
.gitignore
config.ru
Gemfile
Gemfile.lock
Rakefile
README.rdoc

1 source 'https://rubygems.org'
2
3 gem 'rails', '4.1.1'
4 gem 'sqlite3'
5 gem 'sass-rails', '~> 4.0.3'
6 gem 'uglifier', '>= 1.3.0'
7 gem 'coffee-rails', '~> 4.0.0'
8 gem 'jquery-rails'
9 gem 'turbolinks'
10 gem 'jbuilder', '~> 2.0'
11 gem 'sdoc', '~> 0.4.0', group: :doc
12 gem 'spring', group: :development
13
14 # Use ActiveRecord has_secure_password
15 gem 'bcrypt', '~> 3.1.7'
```

Remember to restart your server!!!

# has\_secure\_password



The screenshot shows a code editor with a sidebar on the left and a main editing area on the right. The sidebar, titled 'OPEN FILES', lists 'reviewer.rb' under 'FOLDERS'. The main area shows the code for 'reviewer.rb' with line numbers 1 through 7. The code defines a 'Reviewer' class inheriting from 'ActiveRecord::Base'. It includes the 'has\_secure\_password' method on line 3 and the 'has\_many :books' association on line 5. The 'end' keyword is on line 6.

```
1 class Reviewer < ActiveRecord::Base
2
3   has_secure_password
4
5   has_many :books
6 end
7
```

# How it all works

```
hazink1:~/i_reviewed$ rails c
Loading development environment (Rails 4.1.1)
irb(main):001:0> ActiveRecord::Base.logger = nil
=> nil
irb(main):002:0> Reviewer.column_names
=> ["id", "name", "password_digest", "created_at", "updated_at"]
irb(main):003:0> Reviewer.create! name: "Joe", password: "abc123"
=> #<Reviewer id: 1, name: "Joe", password_digest: "$2a$10$l0vzu5tjoP3t29DHNit/JutymYUP4GhwhiheUvS9GnZ...",
  created_at: "2014-07-27 14:28:08", updated_at: "2014-07-27 14:28:08">
irb(main):004:0> joe = Reviewer.find_by name: "Joe"
=> #<Reviewer id: 1, name: "Joe", password_digest: "$2a$10$l0vzu5tjoP3t29DHNit/JutymYUP4GhwhiheUvS9GnZ...",
  created_at: "2014-07-27 14:28:08", updated_at: "2014-07-27 14:28:08">
irb(main):005:0> joe.authenticate('password')
=> false
irb(main):006:0> joe.authenticate('abc123')
=> #<Reviewer id: 1, name: "Joe", password_digest: "$2a$10$l0vzu5tjoP3t29DHNit/JutymYUP4GhwhiheUvS9GnZ...",
  created_at: "2014-07-27 14:28:08", updated_at: "2014-07-27 14:28:08">
```

# Seed users

```
seeds.rb
1  Reviewer.destroy_all
2  Book.destroy_all
3
4  Book.create! [
5    { name: "My way or highway", author: "Me" },
6    { name: "The Rails Way", author: "The Other Guy" },
7    { name: "The Starving Students' Cookbook", author: "Dede Hall" },
8    { name: "The Vegetarian Family Cookbook", author: "Nava Atlas" },
9    { name: "Ember.js in Action", author: "Joachim Haagen Skeie" }
10 ]
11
12 mine = Book.find_by author: "Me"
13 mine.notes.create! [
14   { title: "Fascinating", note: "This book is simply amazing!" },
15   { title: "Wow", note: "This guy is so full of himself... :)" }
16 ]
17
18 reviewers = Reviewer.create! [
19   { name: "Joe", password: "abc123" },
20   { name: "Jim", password: "123abc" }
21 ]
22
23 Book.all.each do |book|
24   book.reviewer = reviewers.sample
25   book.save!
26 end
```

# HTTP is a stateless protocol

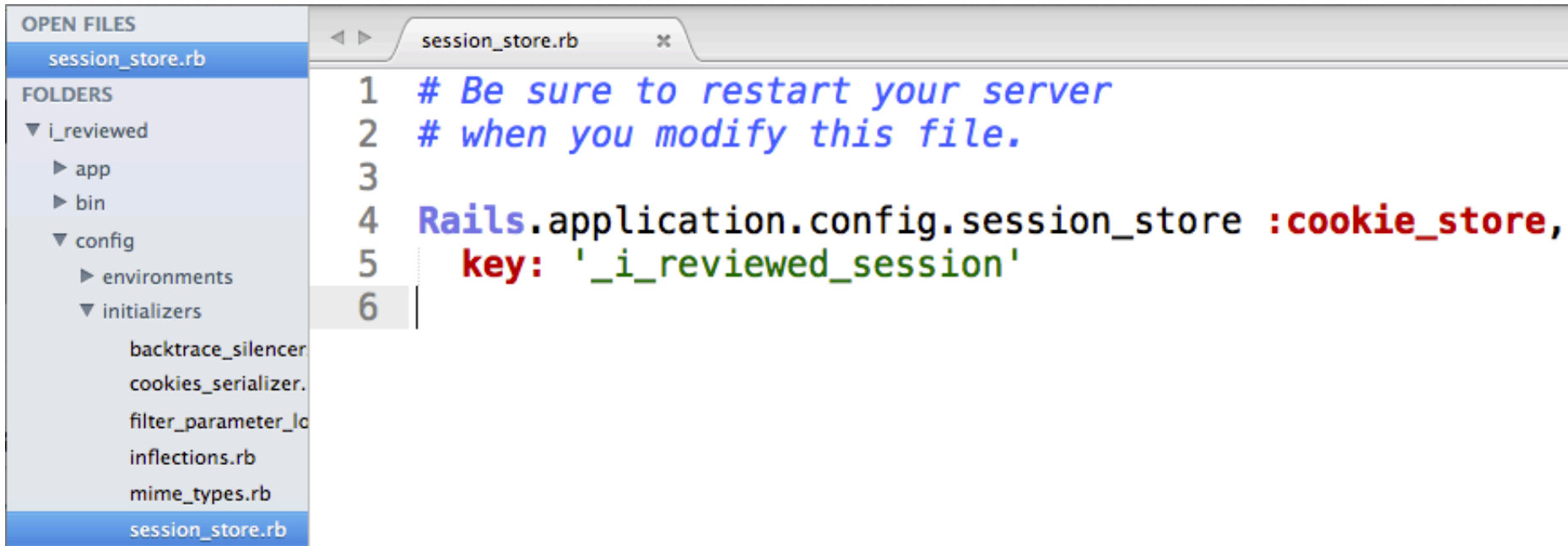
- Http is a stateless protocol
  - Each new request even from the same browser knows nothing about a previous request that was made
  - This means that even if a user logs in on request one – he will be treated as unknown on all the subsequent requests
  - Cookies and Sessions to the rescue



# Sessions in Rails

- Session is created and made available automatically through a `session` hash
- By default, the session-associated information is stored on the client's browser
- So... The server sends the browser a cookie, which the browser stores and sends back to the server on all subsequent requests (until the session ends)

# session\_store.rb initializer



```
1  # Be sure to restart your server
2  # when you modify this file.
3
4  Rails.application.config.session_store :cookie_store,
5    key: '_i_reviewed_session'
6
```

Cookie Store (default) stores session information in the browser

# Restful Sessions controller

- Session can be thought of as a resource – let's go ahead and create a RESTful sessions controller

```
hazink1:~/i_reviewed$ rails g controller sessions new create destroy -q  
hazink1:~/i_reviewed$ _
```

# config/routes.rb

routes.rb

✕

```
Rails.application.routes.draw do

  root to: "books#index"
  resources :books do
    resources :notes, only: [:create, :destroy]
  end

  resources :sessions, only: [:create, :destroy, :new]
end
```

# Sessions Controller actions

- We can think of `new` action as login page and `destroy` as a logout page
- Thus, we'll need `new` (and `create`) actions to create a session and `destroy` to destroy a session
- Let's map login/logout routes to make this more clear

# config/routes.rb

routes.rb

✕

```
Rails.application.routes.draw do
```

```
  root to: "books#index"
```

```
  resources :books do
```

```
    resources :notes, only: [:create, :destroy]
```

```
  end
```

```
  resources :sessions, only: [:create, :destroy, :new]
```

```
  get '/login' => "sessions#new", as: "login"
```

```
  delete '/logout' => "sessions#destroy", as: "logout"
```

```
end
```

This lets us refer to these routes in our code as login\_path/logout\_path or login\_url/logout\_url...

# All routes so far

```
hazink1:~/i_reviewed$ rake routes
```

	Prefix	Verb	URI Pattern	Controller#Action
	root	GET	/	books#index
book_notes	POST	/books/:book_id/notes(.:format)	notes#create	
book_note	DELETE	/books/:book_id/notes/:id(.:format)	notes#destroy	
books	GET	/books(.:format)	books#index	
	POST	/books(.:format)	books#create	
new_book	GET	/books/new(.:format)	books#new	
edit_book	GET	/books/:id/edit(.:format)	books#edit	
book	GET	/books/:id(.:format)	books#show	
	PATCH	/books/:id(.:format)	books#update	
	PUT	/books/:id(.:format)	books#update	
	DELETE	/books/:id(.:format)	books#destroy	
sessions	POST	/sessions(.:format)	sessions#create	
new_session	GET	/sessions/new(.:format)	sessions#new	
session	DELETE	/sessions/:id(.:format)	sessions#destroy	
login	GET	/login(.:format)	sessions#new	
logout	DELETE	/logout(.:format)	sessions#destroy	