

Sessions, Cookies, & Authentication

Ruby on Rails

Any idea on the
concept of
Cookies?

Cookies

Cookies

Stored in the Browser

Logged in as
pkennedytx1@gmail.com.
[Log Out](#)
Logged in!

Log In

Email

Password

Elements

Application

Storage

http://localhost:3000

Filter

| Name | Value |
|--------------------|----------------------------|
| _blog_flash_se... | aFdpUkNwZVFMN1FjQU... |
| _ga | GA1.1.374117029.157109... |
| _ga_3QMKD5H... | GS1.1.1571171218.3.0.15... |
| immersive-rail... | Vjc2VIBmbDILbnBLMGRn... |
| _sessions_lectu... | SbtKDkaZMbSSmxHGh... |

- Key-value pairs.
- Will expire after a defined amount of time.
- Special Cookies hash in rails.
- “Bookmarks” a page.

Special Rails Cookies Hash

1. Using cookies allows you to store certain information in the browser.
2. Notice the cookies hash in the example to the right.
3. Expires is also important because you need a time frame to let the data disappear.

```
class ApplicationController
  before_filter :do_something_with_cookies

  def do_something_with_cookies
    puts "My cookie is: #{cookies[:foo]}"
    cookies[:foo] = {
      value: "some value!",
      expires: 30.minutes.from_now,
    }
  end
end
```

Sessions



From what you know.
How might a website keep
track of how a user is
logged in?

Specific Sessions Hash (a type of cookie)

To identify a session Rails stores a **SECURE** and **TAMPER PROOF** on the browser that contains the entire hash.

The session hash is used similarly to the cookie hash. And can be used the same way.

```
def create
  user = User.find_by_email(params[:email])
  if user && user.authenticate(params[:password])
    session[:user_id] = user.id
    redirect_to login_path, notice: "Logged in!"
  else
    flash.now.alert = "Email or password is invalid"
    render "new"
  end
end
```

| | | | | |
|-------------------------------------|---|-----------|---|-----------|
| _blog_flash_session | aFdpUkNwZVFMN1FjQUpvYlhrbU1jZE9KUnVTREFEcTdhb0ZoUTFVZkJIRUNaK2VpbC9BSENX... | localhost | / | Session |
| _ga | GA1.1.374117029.1571095078 | localhost | / | 2021-10-1 |
| _ga_3QMKD5H30Z | GS1.1.1571171218.3.0.1571171218.0 | localhost | / | 2021-10-1 |
| _immersive-rails-assessment_session | Vjc2ViBmbDlbnBLMGRnVVJTWWtSVTdwWUhrdWRZR3B4V2t4bWxvYTVBa0EwbU8ycGpUM0... | localhost | / | Session |
| _sessions_lecture_session | 6hp7VqPL%2BW6WNd9Kx5dlz8d%2BuLZ2wRm72uWW8t7vXetu2dqskeiZiHSToTtj48wKXeOn... | localhost | / | Session |

Conclusion

Why would you need both **Sessions** and **Cookies**?

They are similar but, **not** the same.

Session expiration is the session itself. The cookie may actually have a defined time length that it will be available.

Sessions and cookies are like temporary free database tables that are unique to a given user and will last until you give it a date to end or the session is closed.

NOTE: they are not really hashes, rails pretends they are. But just know they act very similarly.

Authentication

—

What is the point of Authentication?

The whole point is to make sure the user using your application is who they say they are.

Basic HTTP Authentication.

Digest Authentication.

Encrypted Digest Authentication. (best to use... for now.)

Let's get cracking.

The Do's of Authentication

Don't store passwords in plain text. What does this mean?

Encrypted Digest

More special rails syntax.

Password_digest in models

Has_secure_password

```
class User < ApplicationRecord
  has_secure_password
  validates_uniqueness_of :email
end
```

```
def change
  create_table :users do |t|
    t.string :email
    t.string :password_digest

    t.timestamps
  end
end
```

Creating a new user session

Continuation of Encrypted Digest

To initialize a new user session you now need a new controller. A `sessions_controller.rb` file.

And corresponding routes for the user `:new` `:create` etc...

Check via `authenticate` method where you will find the sessions variable that stores the ID.

```
class SessionsController < ApplicationController
  def new
  end

  def create
    user = User.find_by_email(params[:email])
    if user && user.authenticate(params[:password])
      session[:user_id] = user.id
      redirect_to login_path, notice: "Logged in!"
    else
      flash.now.alert = "Email or password is invalid"
      render "new"
    end
  end

  def destroy
    session[:user_id] = nil
    redirect_to login_path, notice: "Logged out!"
  end
end
```

Remember Me?

```
def create_remember_token  
  self.remember_token = User.encrypt(User.new_remember_token)  
end
```

Anddddddd done.

—