Rails tutorial Ch. 9 -Kevin P.

Stuff that is important:

- Rails can maintain state from one page to the next using persistent cookies (**cookies** method)
- We associate to each user a **remember token** and a corresponding **remember digest** for use in persistent sessions.
- Using the **cookies** method, we create a **persistent session** by placing a permanent remember token cookie on the browser.
- Login status is determined by the presence of a current user based on the temporary session's user id or the permanent session's unique remember token.
- The application signs users out by deleting the session's user id and removing the permanent cookie from the browser.
- The **ternary operator** is a compact way to write simple if-then statements.
- 19-year-old cookies are still perfectly good.

Cookies can store data when the browser is closed. This is good.
Cookies are not inherently secure. This is bad, but for now it just means we need a way to verify cookies like we would a login.

1. Create a random string of digits for use as a remember token.
2. Place the token in the browser cookies with an expiration date far in the future.
3. Save the hash digest of the token to the database.
4. Place an encrypted version of the user's id in the browser cookies.
5. When presented with a cookie containing a persistent user id, find the user in the database using the given id, and verify that the remember token cookie matches the associated hash digest from the database. (note this step is much like logging in)

users	
id	integer
name	string
email	string
created_at	datetime
updated_at	datetime
password_digest	string
remember_digest	string

New Users Model - get with:

```
$ rails generate migration add_remember_digest_to_users remember_digest:string
$ rails db:migrate
```

Get random string with `urlsafe_base64` from `SecureRandom` module:

```
>> SecureRandom.urlsafe_base64
=> "q5lt38hQDc_959PVoo6b7A"
```

Add new_token method to app/models/user.rb (body consists of above line of code)

Token digest works basically like a password digest (remember_token:password::remember_digest:password_digest) from chapter 6, but it's not done automatically for us.

In App/models/user.rb:

```
attr_accessor :remember_token #creates attribute

def remember
    self.remember_token = User.new_token #self avoids creating local var
    update_attribute(:remember_digest, User.digest(remember_token))
  end
```

Cookies

Create with **cookies[:remember_token]** (works like a hash - returns values for keys)

Special code to expire in 20 years:

```
cookies.permanent[:remember_token] = remember_token
```

Signed cookies avoid storing data in plaintext (this code also uses the above permanent method)

```
cookies.permanent.signed[:user_id] = user.id
```

Identify user w/ code like:

```
User.find_by(id: cookies.signed[:user_id])
```

Remember_token keeps a single compromised cookie from being used forever.

In users.rb:

```
def authenticated?(remember_token) #true if token matches digest
    BCrypt::Password.new(remember_digest).is_password?(remember_token)
  end
```

Forget - delete user_id and remember_token, set remember_digest to nil - logout calls this.

Problems happen if a user is logged in on two browsers at once, then logs out.

If digest is nil in `authenticated`, return immediately

Ternary operator

Replaces if{} else{} blocks. Syntax:

Code execution: `boolean? ? do_one_thing : do_something_else`

Assignment: var = `boolean? ? foo : bar`