BEWD Authentication

Matt Heath
Tech Lead, Platform
Hailo

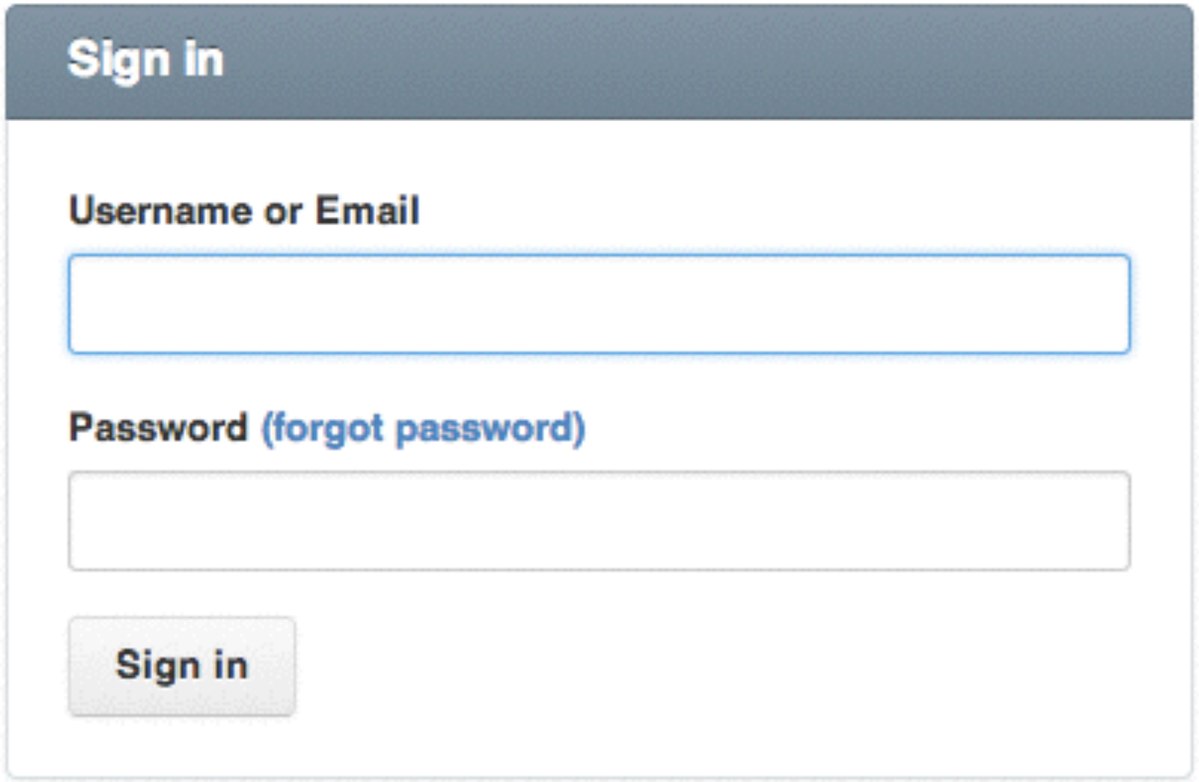
AGENDA

- Review
  - Ritly Solution
- Authentication
  - Authentication explained
  - Devise Gem
- Lab
  - Authenticated Ritly

REVIEW

» Ritly Solution

AUTHENTICATION



AUTHENTICATION

- Use of a combination of username and password to validate user identity. (Obvious, sorry…)
- Tracking a user's identity on our app through the *session*.


Security Cat sayz
This is not a safe site.

SECURITY

» Can I view users' passwords in my app?

HELL NO!

SECURITY

STORING PASSWORDS

» Bad practice to keep passwords in “clear text”

» Passwords can't be stored in plain text in your database.
» If your database is compromised then passwords are compromised as well.
» Don't use the same password for all sites.

SECURITY

SALTY HASHES

» Lots of sites/tutorials will tell you how to store passwords
» **IGNORE THESE!**

» Common advice is to use a one-way hash

```
Digest::SHA2.hexdigest("secret")
# => "e5e9fa1ba31ecd1ae84f75caaa474f3a663f05f4"
```

» Salt is random data that is used as an additional input to a one-way function that hashes a password.

```
salt = "a761ce3a45d97e41840a788495e85a70d1bb35"
password = "secret"
Digest::SHA2.hexdigest(salt+password)
# =>"7963ca00e2e48ea80c615d037494de00a0964682"
```

SECURITY

BCRYPT

- Why Not {MD5, SHA1, SHA256, SHA512, SHA-3, etc}?
- General purposes hashes are designed to be *fast*
- Computing power increases exponentially, and gets cheaper
- For about $300/hour, you could crack around 500,000,000,000 candidate passwords a second, *in 2011*.

- Use BCrypt (an adaptive hash)

- Seriously, BCrypt

- Bcrypt.

AUTH

MANAGING USERS

» When the user is authenticated we store the ‘`user_id`’ in the *session*.

AUTH

SESSIONS

- Session data commonly includes the browser user's identity (name, login, shopping cart, etc.).
- To work, the web server must uniquely identify each browser's particular HTTP requests while the session lasts.
- Commonly, web servers identify browsers by asking them to store a *cookie*.

AUTH

COOKIES

- Used to store small bits of information (maximum size about 4k).

- Cookies allow web servers to provide a temporary unique ID to a browser, to enable session management.
  - Browser storage is not secure.
  - Sensitive data (credit card numbers, etc.) should never be set in a cookie

AUTH

GEMS

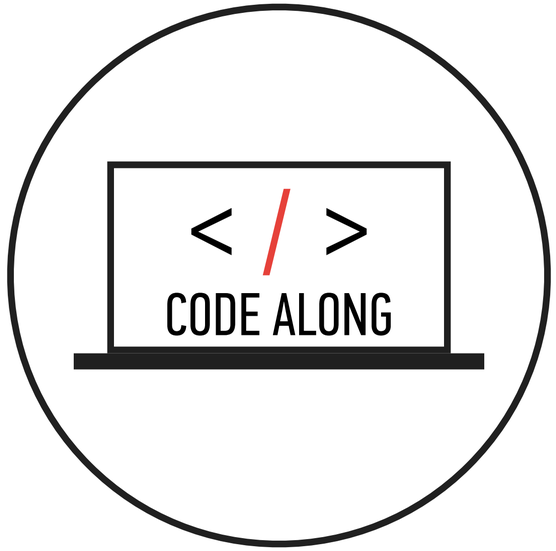
Creating authentication from scratch is a complex process (see resources for more info). However Developers have created Gems to make authentication "easy".

- *Devise*
- CanCan
- Clearance
- OmniAuth
- DoorKeeper

AUTH

DEVISE GEM

- Straight-forward to implement, integrate and customize.
- Handles complex security, so you don't have to.
- Provides controller filters and view helpers (more on that in the code along).
- Recently updated (v3.0.0) with Rails 4 support
- Uses BCrypt!



Ritly - Adding Devise

DEVISE

RECAP

» Adding Devise Gem to the `Gemfile`

```
gem 'devise', '~> 3.0.0'
```

DEVISE

RECAP

Using Devise:

```
rails g devise:install # creates all the devise
Controllers, views and initialisers

rails g devise user    # creates User model (or
modifies it if it exists)

rake db:migrate        # Let's Go!
```

DEVISE

RECAP

View Helpers:

```
<%= user_is_logged_in? %>
<%= current_user %>
```

DEVISE

RECAP

Blocking access:

```
class ApplicationController ...
  before_action :authenticate_user!
end

class HomeController < ApplicationController
  skip_before_action :authenticate_user!
end
```

DEVISE

RECAP

Changing the default Route names:

```
devise_for :users, :path_names => { sign_in:
  'login', sign_out: 'logout' }
```

DEVISE

RECAP

Changing the default Route names:

```
devise_for :users, :path_names => { sign_in:
  'login', sign_out: 'logout' }
```

Lab Time - Authenticated Rewsly

HOMEWORK

Write a list of information/data you want to store about your user.

RESOURCES

Tips, Tricks & Advanced Reading

If you want to expand your knowledge about Rails authentication gems visit Ruby Toolbox (https://www.ruby-toolbox.com/categories/rails_authentication) for a few more authentication gem options.

Great article explaining passwords, hashing, and salt
http://scientopia.org/blogs/goodmath/2013/03/02/passwords-hashing-and-salt/

Advanced article about authorization and users management in rails:
http://edapx.com/2012/04/18/authorization-and-user-management-in-rails/

Tutorial on how to create an advanced admin panel.
http://everydayrails.com/2012/07/31/rails-admin-panel-from-scratch.html

Authentication From Scratch Rails Cast:
http://railscasts.com/episodes/250-authentication-from-scratch)

Still Feel Lost?

Devise Rails Cast: http://railscasts.com/episodes/209-introducing-devise