BEWD - Authentication

Week 6 / Lesson 2

Agenda

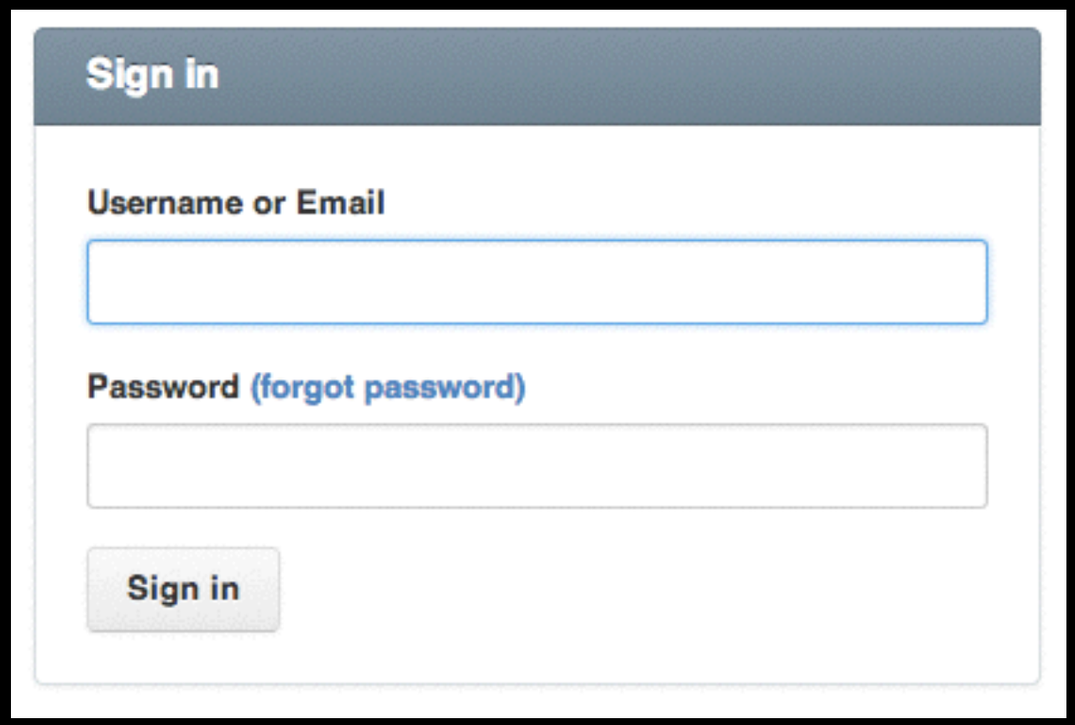
- Review
  - Rewsly Solution
  - Rity Solution (Briefly)
- Authentication
  - Authentication Explained
  - Devise Gem
- Lab
- Authenticated Ritly

Review

Rewsly & Ritly: more projects named by Scooby Doo

- Let's review Rewsly solution from last class.
- Since we're going to work on Ritly, let's review that (briefly) as well.

Authentication



Authentication

- Use of a combination of username and password to validate user identity. (Obvious I know...)
- Tracking a user's identity on our app through the session.
- Auth vs. Auth (authentication/authorization)

Authentication

Security

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

Hashing

Use one way hash.

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

Security

Adding Salt

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

```
salt = "a761ce3a45d97e41840a788495e85a70d1bb3815"
password = "secret"
Digest::SHA2.hexdigest(salt+password)
# =>"7963ca00e2e48ea80c615d037494de00a0964682"
```

Authentication

Managing Users

- When the user is authenticated we store the user_id in the session.

Managing Users

Session

- 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.

Managing Users

Cookie

- 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.

Authentication

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

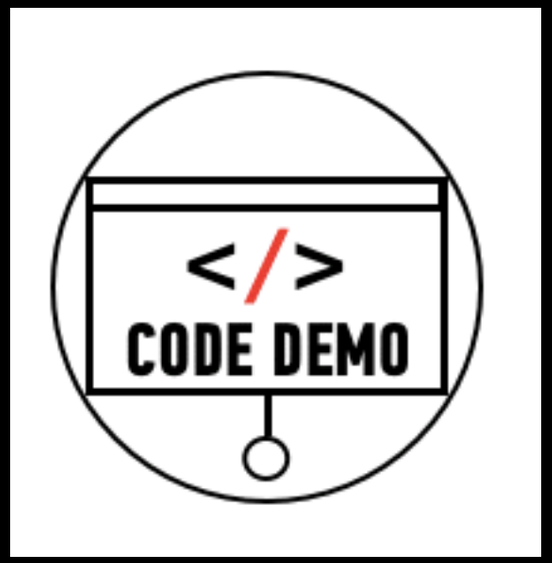
Authentication

Devise Gems (NOT USING THIS)

https://github.com/plataformatec/devise

- Straight-forward to implement, integrate and customize. (no)
- Handles complex security, so you don't have to. (yes)
- Provides controller filters and view helpers (yes)
- Recently updated (v3.0.0) with Rails 4 support! (whee!)

Ritly - Adding Authentication from Scratch



u stop it now ok

~~Devise~~ Authentication

Recap

- Adding Devise Gem to the Gemfile
- Same process for BCrypt and other gems

- gem 'devise', '~> 3.0.0'

Before Filters

Recap

- Blocking access

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

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

Lab Time - Authenticated Rewsly

Homework

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

Resources

Cheat Sheet

- No cheat sheet this class. Remember, Google is your friend!

Tips, Tricks & Advanced Reading

- If you want to expand your knowledge about Rails authentication gems visit Ruby Toolbox for a few more authentication gem options.
- Great article explaining passwords, hashing, and salt.
- Advanced article about authorization and users management in rails.
- Tutorial on how to create an advanced admin panel.
- Authentication From Scratch Rails Cast.

Still Feel Lost?

Catch Up With These Resources

- Devise Rails Cast