

Iron Brush Tattoo *Case Jewelry*

AnderSon

Tue May 3 10:45:24 CDT 2016

Contents

1	Config	2
1.1	Gemfile	2
1.2	Gems	3
1.3	Environments	3
1.3.1	Development	3
1.3.2	Production	4
2	First steps	6
3	Project	7
3.1	User Story	7
3.2	File Upload	7
3.2.1	Dragonfly	8
3.3	File Conversion	15
3.3.1	xlsx processing	15
3.3.2	pdf processing	17
3.4	Authentication	20
3.4.1	AuthO	20
3.4.2	nb	23
3.5	Views	23
3.5.1	Routes	23
3.5.2	Static Pages	24
3.6	Controllors	24
3.6.1	Pages	24
3.6.2	Dashboard	24
3.7	Models	25
3.7.1	Page	25
3.7.2	Spreadsheet	25

3.7.3	Label	25
3.8	Classes	25
3.8.1	SpreadsheetPdf	25
3.8.2	Label	30
3.9	TODO	31
<i>Rails 4.2.6</i>		
https://github.com/IronBrushTattoo/cj_rails.git ../../Retail_		
Jewelry		

1 Config

1.1 Gemfile

./Gemfile

```
source 'https://rubygems.org'

gem 'rails', '4.2.6'
gem 'pg', '~> 0.15'
gem 'sass-rails', '~> 5.0'
gem 'uglifier', '>= 1.3.0'
gem 'coffee-rails', '~> 4.1.0'
gem 'jquery-rails'
gem 'turbolinks'
gem 'jbuilder', '~> 2.0'
gem 'sdoc', '~> 0.4.0', group: :doc
gem 'dragonfly', '~> 1.0.12'
gem 'rack-cache', :require => 'rack/cache'
gem 'prawn'
gem 'prawn-table', '~> 0.2.2'
gem 'roo', '~> 2.3.2'
gem 'chronic', '~> 0.10.2'
gem 'omniauth', '~> 1.3'
gem 'omniauth-auth0', '~> 1.4'

group :development, :test do
  gem 'byebug'
end

group :development do
```

```
gem 'web-console', '~> 2.0'
gem 'spring'
end
```

1.2 Gems

3.2.1 1

1.3 Environments

1.3.1 Development

`./config/environments/development.rb`

```
Rails.application.configure do
  # Settings specified here will take precedence over those in config/application.rb.

  # In the development environment your application's code is reloaded on
  # every request. This slows down response time but is perfect for development
  # since you don't have to restart the web server when you make code changes.
  config.cache_classes = false

  # Do not eager load code on boot.
  config.eager_load = false

  # Show full error reports and disable caching.
  config.consider_all_requests_local = true
  config.action_controller.perform_caching = false

  # Don't care if the mailer can't send.
  config.action_mailer.raise_delivery_errors = false

  # Print deprecation notices to the Rails logger.
  config.active_support.deprecation = :log

  # Raise an error on page load if there are pending migrations.
  config.active_record.migration_error = :page_load

  # Debug mode disables concatenation and preprocessing of assets.
  # This option may cause significant delays in view rendering with a large
  # number of complex assets.
```

```

config.assets.debug = true

# Asset digests allow you to set far-future HTTP expiration dates on all assets,
# yet still be able to expire them through the digest params.
config.assets.digest = true

# Adds additional error checking when serving assets at runtime.
# Checks for improperly declared sprockets dependencies.
# Raises helpful error messages.
config.assets.raise_runtime_errors = true

# Raises error for missing translations
# config.action_view.raise_on_missing_translations = true

OmniAuth.config.on_failure = Proc.new { |env|
  message_key = env['omniauth.error.type']
  error_description = Rack::Utils.escape(env['omniauth.error'].error_reason)
  new_path = "#{env['SCRIPT_NAME']}#{OmniAuth.config.path_prefix}/failure?message=#{
  Rack::Response.new(['302 Moved'], 302, 'Location' => new_path).finish
}
end

```

1.3.2 Production

```
./config/environments/production.rb
```

```

Rails.application.configure do
  # Settings specified here will take precedence over those in config/application.rb.

  # Code is not reloaded between requests.
  config.cache_classes = true

  # Eager load code on boot. This eager loads most of Rails and
  # your application in memory, allowing both threaded web servers
  # and those relying on copy on write to perform better.
  # Rake tasks automatically ignore this option for performance.
  config.eager_load = true

  # Full error reports are disabled and caching is turned on.
  config.consider_all_requests_local       = false

```

```

config.action_controller.perform_caching = true

# Enable Rack::Cache to put a simple HTTP cache in front of your application
# Add 'rack-cache' to your Gemfile before enabling this.
# For large-scale production use, consider using a caching reverse proxy like
# NGINX, varnish or squid.
config.action_dispatch.rack_cache = true

# Disable serving static files from the '/public' folder by default since
# Apache or NGINX already handles this.
config.serve_static_files = ENV['RAILS_SERVE_STATIC_FILES'].present?

# Compress JavaScripts and CSS.
config.assets.js_compressor = :uglifier
# config.assets.css_compressor = :sass

# Do not fallback to assets pipeline if a precompiled asset is missed.
config.assets.compile = false

# Asset digests allow you to set far-future HTTP expiration dates on all assets,
# yet still be able to expire them through the digest params.
config.assets.digest = true

# 'config.assets.precompile' and 'config.assets.version' have moved to config/initializers/assets.rb

# Specifies the header that your server uses for sending files.
# config.action_dispatch.x_sendfile_header = 'X-Sendfile' # for Apache
# config.action_dispatch.x_sendfile_header = 'X-Accel-Redirect' # for NGINX

# Force all access to the app over SSL, use Strict-Transport-Security, and use secure cookies.
# config.force_ssl = true

# Use the lowest log level to ensure availability of diagnostic information
# when problems arise.
config.log_level = :debug

# Prepend all log lines with the following tags.
# config.log_tags = [ :subdomain, :uuid ]

# Use a different logger for distributed setups.

```

```

# config.logger = ActiveSupport::TaggedLogging.new(SyslogLogger.new)

# Use a different cache store in production.
# config.cache_store = :mem_cache_store

# Enable serving of images, stylesheets, and JavaScripts from an asset server.
# config.action_controller.asset_host = 'http://assets.example.com'

# Ignore bad email addresses and do not raise email delivery errors.
# Set this to true and configure the email server for immediate delivery to raise de
# config.action_mailer.raise_delivery_errors = false

# Enable locale fallbacks for I18n (makes lookups for any locale fall back to
# the I18n.default_locale when a translation cannot be found).
config.i18n.fallbacks = true

# Send deprecation notices to registered listeners.
config.active_support.deprecation = :notify

# Use default logging formatter so that PID and timestamp are not suppressed.
config.log_formatter = ::Logger::Formatter.new

# Do not dump schema after migrations.
config.active_record.dump_schema_after_migration = false

OmniAuth.config.on_failure = Proc.new { |env|
  message_key = env['omniauth.error.type']
  error_description = Rack::Utils.escape(env['omniauth.error'].error_reason)
  new_path = "#{env['SCRIPT_NAME']}#{OmniAuth.config.path_prefix}/failure?message=#{
  Rack::Response.new(['302 Moved'], 302, 'Location' => new_path).finish
}

end

```

3.4.1

2 First steps

```

rake db:migrate
rake db:setup

```

3 Project

The purpose of this application is to produce several pdf files from an xlsx file, as a re-implementation of <https://github.com/IronBrushTattoo/cj> as a web application.

3.1 User Story

- user logs in (3.4)
- chooses xlsx file for upload 3.2
- selects number of days back to make labels from
- submits
 - BACKGROUND
 - * cj-parser.rb does what it does...
 - ☐ rewrite in rails?
- downloads sheets(pdf files)

3.2 File Upload

- possible gems https://www.ruby-toolbox.com/categories/rails_file_uploads
 - paperclip
 - * nb
 - used paperclip before
 - seemed to be designed specifically for image files
 - always worked well
 - carrierwave
 - * nb
 - used before, but not thoroughly
 - i kind of remember having issues with it
 - dragonfly <https://github.com/markevans/dragonfly> <http://markevans.github.io/dragonfly/> <http://markevans.github.io/dragonfly/rails/>

Dragonfly is a highly customizable ruby gem for handling images and other attachments and is already in use on thousands of websites

3.2.1

- * nb

- used briefly before
- i remember it being an easy configuration

- attachment fu https://github.com/technoweenie/attachment_fu

Treat an ActiveRecord model as a file attachment, storing its patch, size, content type, etc. <http://weblog.technoweenie.net>

- * nb

- has not been maintained since Apr 25, 2009

- refile

- * nb

- was my next choice when previously working with file uploads

- jquery.fileupload-rails

- imagery

- attached

- papermill

- fileuploader-rails

- filecip

- simple-image-uploader

3.2.1 Dragonfly

<http://markevans.github.io/dragonfly/rails/>

1. Setup

- `⊠ gem 'dragonfly', '~> 1.0.12'`
 - `⊠ modify ??`
 - `⊠ bundle install`
- `⊠ rails g dragonfly`
generates `config/initializers/dragonfly.rb`
`./config/initializers/dragonfly.rb`

```
require 'dragonfly'

# Configure
Dragonfly.app.configure do
  plugin :imagemagick

  secret "72245c7371d66ccca6f9356779fa16e3104e6676c1e57af987e9e3f92130dca5"

  url_format "/media/:job/:name"

  datastore :file,
    root_path: Rails.root.join('public/system/dragonfly', Rails.env),
    server_root: Rails.root.join('public')
end

# Logger
Dragonfly.logger = Rails.logger

# Mount as middleware
Rails.application.middleware.use Dragonfly::Middleware

# Add model functionality
if defined?(ActiveRecord::Base)
  ActiveRecord::Base.extend Dragonfly::Model
  ActiveRecord::Base.extend Dragonfly::Model::Validations
end
```

2. Handling attachments

- example (replace Photo model with Spreadsheet)

Model: *Photo*

- add *image* attribute to Photo

```
class Photo < ActiveRecord::Base
  dragonfly_accessor :image # defines a reader/writer for image
  # ...
end
```

- needs *image_uid* column, create migration with

```
rails g migration

add_column :photos, :image_uid, :string
add_column :photos, :image_name, :string # Optional - if you want
                                         # urls to end with the
                                         # original filename
```

- view for uploading

```
app/views/photos/...

<% form_for @photo do |f| %>
  ...
  <%= f.file_field :image %>
  ...
<% end %>
```

- allow parameter *image* to be accepted by the controller

```
app/controllers/photos_controller.rb

params.require(:photo).permit(:image)
```

- view for displaying

```
<%= image_tag @photo.image.thumb('400x200#').url if @photo.image_stored? %>
```

– more can be done with models

- Spreadsheet model sketch based on above example

Model: *Spreadsheet*

3.7.2

– ☒ add *file* attribute to Spreadsheet

```
class Spreadsheet < ActiveRecord::Base
  dragonfly_accessor :file # defines a reader/writer for file
  # ...
end
```

– ☒ needs *file_{uid}* column, create migration with

```
rails g migration AddFileUidToSpreadsheets file_uid:string
rails g migration AddFileNameToSpreadsheets file_name:string
```

```
./db/migrate/20160504011342_add_file_uid_to_spreadsheets.
rb ./db/migrate/20160504011542_add_file_name_to_spreadsheets.
rb
```

```
add_column :spreadsheets, :file_uid, :string
add_column :spreadsheets, :file_name, :string # Optional - if you want
                                              # urls to end with the
                                              # original filename
```

```
rake db:migrate
```

– ☒ view for uploading

```
./app/views/spreadsheets/
```

```
<% form_for @spreadsheet do |f| %>
  ...
  <%= f.file_field :file %>
  ...
<% end %>
```

- ☒ allow parameter *file* to be accepted by the controller

```
./app/controllers/spreadsheets_controller.rb
```

```
params.require(:spreadsheet).permit(:file)
```

```
??
```

```
class SpreadsheetsController < ApplicationController
  before_action :set_spreadsheet, only: [:show, :edit, :update, :destroy]

  def index
    @spreadsheets = Spreadsheet.all
  end

  def show
    @spreadsheet = Spreadsheet.find(params[:id])
    respond_to do |format|
      format.html
      format.pdf do
        pdf = SpreadsheetPdf.new(@spreadsheet, view_context)
        send_data pdf.render,
                  filename: "spreadsheet_#{@spreadsheet.created_at.strftime('%Y%m%d%H%M%S')}.pdf",
                  type: "application/pdf",
                  disposition: "inline"
      end
    end
  end

  def new
    @spreadsheet = Spreadsheet.new
  end

  def edit
  end

  def create
    @spreadsheet = Spreadsheet.new(spreadsheet_params)

    respond_to do |format|
```

```

        if @spreadsheet.save
          format.html { redirect_to @spreadsheet, notice: 'Spreadsheet was saved' }
          format.json { render :show, status: :created, location: @spreadsheet }
        else
          format.html { render :new }
          format.json { render json: @spreadsheet.errors, status: :unprocessable_entity }
        end
      end
    end
  end

  def update
    respond_to do |format|
      if @spreadsheet.update(spreadsheet_params)
        format.html { redirect_to @spreadsheet, notice: 'Spreadsheet was updated' }
        format.json { render :show, status: :ok, location: @spreadsheet }
      else
        format.html { render :edit }
        format.json { render json: @spreadsheet.errors, status: :unprocessable_entity }
      end
    end
  end

  def destroy
    @spreadsheet.destroy
    respond_to do |format|
      format.html { redirect_to spreadsheets_url, notice: 'Spreadsheet was destroyed' }
      format.json { head :no_content }
    end
  end

  private

  def set_spreadsheet
    @spreadsheet = Spreadsheet.find(params[:id])
  end

  def spreadsheet_params
    params.require(:spreadsheet).permit(:index, :file, :days)
  end
end

```

- * nb
- 1 1(b)i
- ☒ view for displaying


```
./app/views/spreadsheets/show.html.erb ./app/views/
spreadsheets/index.html.erb
```

```
<%= @spreadsheet.file_name if @spreadsheet.file_stored? %>
```
- more can be done with models

3. Caching

- ☒ ??


```
gem 'rack-cache', :require => 'rack/cache'
```

 - ☒ bundle install
- ☒ uncomment in 1.3.2


```
config.action_dispatch.rack_cache = true
```

4. Custom Endpoints

3.5.1

- ☐ text generation example


```
get "text/:text" => Dragonfly.app.endpoint { |params, app|
  app.generate(:text, params[:text], 'font-size' => 42)
}
```
- ☐ endpoint callable from javascript (e.g. /image?file=egg.png&size=30x30)


```
get "image" => Dragonfly.app.endpoint { |params, app|
  app.fetch_file("some/dir/#{params[:file]}").thumb(params[:size])
}
```

5. Using Another Data Store

<http://www.sitepoint.com/file-uploads-dragonfly/>

- ☐ add `./public/system/dragonfly` to `./.gitignore`

```
# See https://help.github.com/articles/ignoring-files for more about ignoring
#
# If you find yourself ignoring temporary files generated by your text editor
# or operating system, you probably want to add a global ignore instead:
#   git config --global core.excludesfile '~/.gitignore_global'

# Ignore bundler config.
/.bundle

# Ignore all logfiles and tempfiles.
/log/*
!/log/.keep
/tmp

/public/system/dragonfly
```

3.3 File Conversion

3.3.1 xlsx processing

1. Roo

<https://github.com/roo-rb/roo>

- ☒ ??

```
gem 'roo', '~> 2.3.2'
```

– ☒ bundle install

- ☐ examine code from native application

```
file:///home/son/IBT/jewelry/Retail_Jewelry/cj-parser.
rb
```

```

: require 'roo'
:
: def get_labels(file)
:   puts "getting labels"
:
:   labels = []
:
:   xls_file = Roo::Spreadsheet.open(file)
:
:   xls_file.sheets.each do |sheet|
:
:     sheet = xls_file.sheet(sheet)
:
:     sheet.parse[4..-1].each do |row|
:
:       zero,one,two,four,five,ten = nil_convert(row[0]),
:       nil_convert(row[1]),
:       nil_convert(row[2]),
:       nil_convert(row[4]),
:       nil_convert(row[5]),
:       nil_convert(row[10])
:
:       sizes = strip(five.to_s)
:       gauge = "#{sizes[0]}g"
:       size = "#{sizes[1]}\\""
:       desc = two.gsub("&", "and")
:       id = one.to_s.split(/-/)[0]
:       price = "$#{four.to_s.split(".")[0]}"
:       supply = five
:       updated = Chronic.parse(ten).to_f
:
:       label = Label.new(gauge,
:
:                           size,
:                           desc,
:                           id,
:                           price,
:                           supply,
:                           updated
:
:     )
:
:   end
:
: end

```



```

:       seconds = 60*60*24*$days
:
:       if (Time.now.to_f - updated.to_f) < seconds
:         puts label.id
:         $labelID = label.id
:         labels.push label
:       end
:
:     end
:   end
:
:   return labels
:
: end

```

3.3.2 pdf processing

1. Prawn <https://github.com/prawnpdf/prawn>

(a) Setup

- ☒ ??

```

#gem 'prawn', '~> 2.1'
gem 'prawn'

```

- ☒ bundle install

i. Manual

<http://prawnpdf.org/manual.pdf>

- ☒ clone repository

```
git clone https://github.com/prawnpdf/prawn
```

- ☐ switch to the stable branch

```
git branch stable
```

- ☒ bundle install

- ☐ bundle exec rake manual generates *manual.pdf* in the project root

(b) Tutorials

i. Creating PDF Using Prawn in Ruby on Rails

<http://www.idyllic-software.com/blog/creating-pdf-using-prawn-in-ruby-on-rails>

- ☒ ??

```
gem 'prawn'
bundle install
```

- ☒ ./config/initializers/mime_types.rb
create a PDF Mime::Type inside mime_types.rb

```
# Be sure to restart your server when you modify this file.
```

```
# Add new mime types for use in respond_to blocks:
# Mime::Type.register "text/richtext", :rtf
```

```
Mime::Type.register "application/pdf", :pdf
```

- ☒ ./app/controllers/spreadsheets_controller.rb
Spreadsheets

<http://www.idyllic-software.com/blog/creating-pdf-using-prawn-in-ruby-on-rails>

```
class InvoicesController < ApplicationController
```

```
  before_filter :authenticate_customer!, :only => [:index, :show]
```

```
  def index
```

```
    @invoices = Invoice.all_invoices(current_customer)
```

```
  end
```

```
  def show
```

```
    @invoice = Invoice.find(params[:id])
```

```
    respond_to do |format|
```

```
      format.html
```

```
      format.pdf do
```

```
        pdf = InvoicePdf.new(@invoice, view_context)
```

```

        send_data pdf.render, filename:
          "invoice_#{@invoice.created_at.strftime("%d/%m/%Y")}.pdf",
          type: "application/pdf"
      end
    end
  end

end

  - ☒ open pdf in browser instead of download
    add /disposition: "inline"/ after type
  • ☒ create a new class app/pdfs/spreadsheetpdf
    ??
    Spreadsheets
  • ☒ restart server
  • ☒ ??
    set the @spreadsheet and viewcontext

    def initialize(spreadsheet, view)
      super()
      @spreadsheet = spreadsheet
      @view = view
      text "Spreadsheet #{@spreadsheet.id}"
    end

  • ☐ create different methods inside ?? as per what you
    want to show on the pdf
    - example
      def logo
        logopath = "#{Rails.root}/app/assets/images/logo.png"
        image logopath, :width => 197, :height => 91
      end

```

A. Issues

- ☐ NameError (uninitialized constant SpreadsheetsController::SpreadsheetPdf)

2. nb <https://rubygems.org/search?utf8=%E2%9C%93&query=latex> <http://www.sitepoint.com/hackable-pdf-typesetting-in-ruby-with-prawn/>

<https://github.com/prawnpdf/prawn>

1 is active and looks rad!

- outdated but possibly useful

<https://github.com/baierjan/rails-latex> <https://github.com/bruce/rtex>

3.4 Authentication

3.4.1 AuthO

<https://manage.auth0.com/#/> <https://manage.auth0.com/#/applications/eD2I5SEBmvgx3tEzeFd35mJfvW5HkasK/quickstart>

- ☒ 1. Add dependencies

??

```
gem 'omniauth', '~> 1.3'
gem 'omniauth-auth0', '~> 1.4'
```

– ☒ bundle install

- ☐ 2. Initialize Omniauth AuthO

`./config/initializers/auth0.rb`

```
Rails.application.config.middleware.use OmniAuth::Builder do
  provider(
    :auth0,
    'eD2I5SEBmvgx3tEzeFd35mJfvW5HkasK',
    'CbeRu17uzZttvi8qLafpZJLWr2vTcEf33VLAztuT16FmeujTgrHLtWMSxSe-Plwf',
    'sonarch.auth0.com',
    callback_path: "/auth/auth0/callback"
  )
end
```

– ☐ add to gitignore?

- ☒ 3. Add the AuthO callback handler

```
rails g controller auth0 callback failure --skip-template-engine --skip-assets
```

```
./app/controllers/auth0_controller.rb
```

```
class Auth0Controller < ApplicationController
  def callback
    # This stores all the user information that came from Auth0 and the IdP
    session[:userinfo] = request.env['omniauth.auth']

    # Redirect to the URL you want after successful auth
    redirect_to '/dashboard'
  end

  def failure
    # show a failure page or redirect to an error page
    @error_msg = request.params['message']
  end
end
```

– ☒ replace the generated routes in ./config/routes.rb

3.5.1

```
get "/auth/auth0/callback" => "auth0#callback"
get "/auth/failure" => "auth0#failure"
```

- ☒ 4. Specify the callback on AuthO Dashboard

For security purposes, register callback URL of the application at
<https://manage.auth0.com/#/applications>

<http://yourUrl/auth/auth0/callback>

- ☒ 5. Triggering login manually or integrating the AuthOLock

– ☒ passwordless - Email Code

```
./app/views/pages/login.html.erb
```

```

<script src="https://cdn.auth0.com/js/lock-passwordless-1.0.min.js"></script>
<script type="text/javascript">
  var clientID = 'eD2I5SEBmvGx3tEzeFd35mJfvW5HkasK';
  var domain = 'sonarch.auth0.com';

  var lock = new Auth0LockPasswordless(clientID, domain);

  function open() {
    lock.emailcode({
      callbackURL: 'http://a1e4c04e.ngrok.io/auth/auth0/callback',
      authParams: {
        scope: 'openid profile'
      }
    });
  }
</script>
<button onclick="window.open();">Email Code</button>

```

update 3.5.1

- ☐ 6. Accessing user information

via *userinfo* stored in the session on step 3

```

class DashboardController < SecuredController
  def show
    @user = session[:userinfo]
  end
end

<div>
  
  <h2>Welcom <%= @user[:info][:name] %></h2>
</div>

./app/views/dashboard/show.html.erb

<div>
  
  <h2>Welcom <%= @user[:info][:name] %></h2>
</div>

```

3.6.2 3.6.1

- ☐ Troubleshooting

- ☐ "We're sorry, we can't send you the email..."
auth0 in development
<https://auth0.com/docs/lifecycle>

- ☐ get error description on failure

`./config/environments/production.rb`

```
OmniAuth.config.on_failure = Proc.new { |env|  
  message_key = env['omniauth.error.type']  
  error_description = Rack::Utils.escape(env['omniauth.error'].error_reason)  
  new_path = "#{env['SCRIPT_NAME']}#{OmniAuth.config.path_prefix}/failure?message=  
  Rack::Response.new(['302 Moved'], 302, 'Location' => new_path).finish  
}
```

1 1.3 1.3.2

3.4.2 nb

- oauth?
 - login with familiar accounts
 - * google, fb, twitter, etc

3.5 Views

3.5.1 Routes

`./config/routes.rb`

```
Rails.application.routes.draw do  
  root 'spreadsheets#new'
```

```
  resources :spreadsheets
```

```
  get "spreadsheets" => "spreadsheets#new"
```

```

    get "login" => "pages#login"

    get "/auth/auth0/callback" => "auth0#callback"
    get "/auth/failure" => "auth0#failure"
  end
end

```

3.6.1 3.4.1 4

3.5.2 Static Pages

```
root 'pages#home'
```

3.6.1

1. Home

```
./app/views/pages/home.html.erb
```

3.6 Controllers

3.6.1 Pages

Static pages controller

```
rails g controller pages --skip-assets
```

```
./app/controllers/pages_controller.rb
```

```
class PagesController < ApplicationController
end
```

- nb

3.4.1

3.6.2 Dashboard

```
rails g controller Dashboard show
```

```
./app/controllers/dashboard_controller.rb
```

```
class DashboardController < SecuredController
  def show
    @user = session[:userinfo]
  end
end
```


3.7 Models

3.7.1 Page

3.7.2 Spreadsheet

```
./app/models/spreadsheet.rb
```

```
class Spreadsheet < ActiveRecord::Base
  dragonfly_accessor :file # defines a reader/writer for file
end
```

3.7.3 Label

3.8 Classes

3.8.1 SpreadsheetPdf

```
mkdir ./app/pdfs
```

```
./app/pdfs/spreadsheet_pdf.rb
```

```
class SpreadsheetPdf < Prawn::Document

  def initialize(spreadsheet, view)
    super()
    @spreadsheet = spreadsheet
    file_path = "public/system/dragonfly/development"
    xls_file = get_labels("#{file_path}/#{@spreadsheet.file_uid}")
    @view = view

    make_labels(xls_file)
  end

  def desc_box(desc)
    formatted_text_box [
      {
        text: "#{desc}",
        color: "ffffff",
        styles: [:italic],
        size: 11
      }
    ]
  end
end
```

```

    ], at: [$padding,$top], width: $text_width
end

def size_box(gauge,size)
  transparent(0.1) do
    stroke_rectangle [0, $box_height+$padding], $box_width, $box_height*0.75
  end

  transparent(0.6) do
    formatted_text_box [
      {
        text: "#{gauge} #{size}",
        color: "ffffff",
        size: 15,
        styles: [:normal],
        character_spacing: 0.5
      }
    ], at: [($box_width/2)-$padding,$mid], width: $text_width
  end
end

def id_box(id)
  formatted_text_box [
    {
      text: "#{id}",
      color: "ffffff",
      align: :left
    }
  ], at: [$padding,$low], width: $text_width
end

def price_box(price)
  formatted_text_box [
    {
      text: "#{price}",
      color: "ffffff",
      align: :right,
      styles: [:italic]
    }
  ]
end

```

```

    ], at: [$box_width-($padding*3),$low], width: $text_width
end

def make_labels(file)

  define_grid(:columns => 4, :rows => 8, :row_gutter => 10, :column_gutter => 0)
  margin = 14

  count = 1
  row = 0
  column = 0

  start_new_page(:margin => margin)

  file.each do |label|

    grid(row, column).bounding_box do
      #stroke_axis

      fill_color "000000"
      stroke_color "ffffff"

      $box_width = 144
      $box_height = 81
      $padding = 12
      $text_width = 90
      $top = $box_height-$padding
      $mid = $top - 30
      $low = $mid - 20

      transparent(0.9) do
        fill_rectangle [0, $box_height], $box_width, $box_height

        font "Times-Roman", :size => 10

        desc_box(label.desc)
        size_box(label.gauge, label.size)
        id_box(label.id)
        price_box(label.price)
      end
    end
  end
end

```

```

end

if count%4 == 0
  row+=1
  column = 0
else
  column+=1
end

if count%32 == 0
  start_new_page(:margin => margin)
  row = 0
  column = 0
end

count+=1
end

end

def logo
  logopath = "#{Rails.root}/app/assets/images/logo.png"
  image logopath, :width => 197, :height => 197
end

def get_labels(file)

  labels = []

  xls_file = Roo::Spreadsheet.open(file, extension: :xlsx)

  xls_file.sheets.each do |sheet|

    sheet = xls_file.sheet(sheet)

    sheet.parse[0..-1].each do |row|

      zero,one,two,four,five,ten = nil_convert(row[0]),
        nil_convert(row[1]),

```

```

    nil_convert(row[2]),
    nil_convert(row[4]),
    nil_convert(row[5]),
    nil_convert(row[10])

    sizes = strip(five.to_s)
    gauge = "#{sizes[0]}g" unless sizes[0].nil?
    size = "#{sizes[1]}\\" unless sizes[1].nil?
    desc = two.gsub("&", "and")
    id = one.to_s.split(/-/)[0]
    price = "$#{four.to_s.split(".")[0]}"
    supply = five
    updated = Chronic.parse(ten).to_f

    label = Label.new(gauge,
                      size,
                      desc,
                      id,
                      price,
                      supply,
                      updated
                    )

    seconds = 60*60*24*@spreadsheet.days.to_f

    if (Time.now.to_f - updated.to_f) < seconds
      #puts label.id
      $labelID = label.id
      labels.push label
    end

  end
end

return labels

end

def nil_convert(c)
  if c.nil?

```

```

        ""
      else
        c
      end
    end

    def strip(s)
      s.gsub(/"/, '').
        gsub(/g/, '').
        gsub(/G/, '').
        gsub(/,/, '').
        split(' ')
    end
  end
end

```

- nb

1

Spreadsheets

3.8.2 Label

./app/pdfs/label.rb

```

class Label

  def initialize(gauge, size, desc, id, price, supply, updated)
    @gauge = gauge
    @size = size
    @desc = desc
    @id = id
    @price = price
    @supply = supply
    @updated = updated
  end

  attr_reader :gauge, :size, :desc, :id, :price, :supply, :updated
end

```

3.9 TODO

- ☐ Testing
- ☐ sidekiq
 - ☐ background processes for creating pdfs?
- ☐ requirements
 - ☒ roo
 - ☒ chronic
 - ☐ 1
- ☒ migrate code from cj-parser
- ☐ user authentication
- ☐ file upload
 - ☐ only xlsx file?
 - ☐ AWS
- ☐ file storage
 - ☐ archival api?
- ☐ production
 - ☐ heroku
 - * ☐ secrets