HW 4: BDD+TDD Cycle

In this homework you will use a combination of Behavior-Driven Design (BDD) and Test-Driven Development (TDD) with the Cucumber and RSpec tools to add a new feature to a SaaS app, and deploy the resulting app on Heroku.

Adding director info to RottenPotatoes

In this assignment you'll use BDD & TDD to add a "find movies with same director" feature to RottenPotatoes.

You can start from your HW 3 solution, or from the repo saasbook/hw4_rottenpotatoes on GitHub. (If you choose to do the latter, you should fork the repo and then clone from your own fork, so that you can push your own changes to tyour copy of the repo on GitHub. Remember to make your repository private.)

1. Setup of Cucumber and RSpec: make sure your Gemfile contains these changes from Section 4.4 of ELLS, as well as the line gem 'rspec-rails' , within the group :test,:development block. Run bundle install --without production to make sure the gems are properly installed. Finally, run these commands to set up the Cucumber directories (under features/) and RSpec directories (under spec/) if they don't already exist, allowing overwrite of any existing files:
   rails generate cucumber:install capybara
   rails generate cucumber_rails_training_wheels:install
   rails generate rspec:install
   You can double-check if everything was installed by running the tasks rake spec and rake cucumber. Since presumably you have no features or specs yet, both tasks should execute correctly reporting that there are zero tests to run. Depending on your version of rspec, it may also display a message stating that it was not able to find any _spec.rb files.
2. Create and apply a migration that adds the Director field to the movies table. The director field should be a string containing the name of the movie's director. HINT: use the add_column method of ActiveRecord::Migration to do this. *(Remember that once the migration is applied, you also have to do* `rake db:test:prepare` *to load the new post-migration schema into the test database!)*
3. We've provided three Cucumber scenarios to drive creation of the happy path of Search for Movies by Director:
   - The first lets you add director info to an existing movie, and doesn't require creating any new views or controller actions (but does require modifying existing views, and will require creating a new step definition and possibly adding a line or two to features/support/paths.rb).
   - The second lets you click a new link on a movie details page "Find Movies With Same Director", and shows all movies that share the same director as the displayed movie. For this you'll have to modify the existing Show Movie view,

and you'll have to add a route, view and controller method for Find With Same Director.
    - The third handles the sad path, when the current movie has no director info but we try to do "Find with same director" anyway.

Going one Cucumber step at a time, use RSpec to create the appropriate controller and model specs to drive the creation of the new controller and model methods. At the least, you will need to write tests to drive the creation of:

- a RESTful route for Find Similar Movies (HINT: use the 'match' syntax for routes as suggested in "Non-Resource-Based Routes" at the end of section 3.9 of ELLS)
- a controller method to receive the click on "Find With Same Director", and grab the id (for example) of the movie that is the subject of the match (i.e. the one we're trying to find movies similar to)
- a model method in the Movie model to find movies whose director matches that of the current movie

It's up to you to decide whether you want to handle the sad path of "no director" in the controller method or in the model method, but you must provide a test for whichever one you do. Remember to include the line require 'spec_helper' at the top of your _spec.rb files.

We want you to report your code coverage as well. Make sure that the line gem 'simplecov' exists in your Gemfile. Add the following lines to the **TOP** of spec/spec_helper.rb and features/support/env.rb: http://pastebin.com/vcq4MuBB
Now when you run rake spec or rake cucumber, SimpleCov will generate a report in a directory named coverage/. Since both RSpec and Cucumber are so widely used, SimpleCov can intelligently merge the results, so running the tests for Rspec does not overwrite the coverage results from SimpleCov and vice versa. See the screencast for step-by-step instructions on setting up SimpleCov.

TURN-IN:

- Cucumber feature file (if different from the one provided) and step definitions (i.e., contents of your features/ directory)
- RSpec tests (i.e., contents of spec/ directory)
- SimpleCov report files showing 90% or greater coverage for your models and controllers, as described in ELLS section 5.8 and its accompanying screencast.
- The URI of your deployed app on Heroku that passes all the scenarios (later in the semester, we'll show how to run Cucumber scenarios directly against your deployed app, not just your local copy)
- Any files you modified (i.e. app/, config/routes.rb, db/migrate/, etc.)