Critique on Char Broiled

by Team Friends!
Ari Vogel, Jane Cotler, Kristin Au, Preston Thompson
Code Flaw #1: (Ari)

```ruby
respond_to do [format]
  format.html { redirect_to({'controller' => 'users', 'action' => 'show', :id => current_user.id}) }
  format.json { head :no_content }
end
```

In the `manage_permissions` method in the StaticPagesController, you are using `format.html` and `format.json` but neither `manage_permissions.html.erb` nor `manage_permissions.json.erb` exist, so a 500 error is thrown.
Code Flaw #2: (Jane)

- Security concern in StaticPagesController: a Content Creator can be added without being logged in. One solution is to add a require_admin or a require_login before filter.
- In manage_permissions, make your code consistently indented (lines 10 and 19). Make sure this is consistent in the rest of your code as well.

```ruby
def manage_permissions
    @user = User.find_by_username(params[:user])
    if @user != nil
        if @user.is_content_creator? || @user.is_admin?
            flash[:danger] = @user.first_name + " already has content creation permission"
        else
            @user.make_content_creator()
            flash[:success] = @user.first_name + " is now a content creator"
        end
    else
        flash[:danger] = params[:user] + " is not a valid username"
    end
end
```

```ruby
def require_admin
    redirect_to :back, notice: "Is not an admin" unless @current_user.is_"admin?"
end
```

```ruby
def require_login
    redirect_to root_url, notice: "Please sign in." unless signed_in? # update to new_session_url
end
```
Code Flaw #3: SQL Injection (Kristin)

- User-inputted code is entered and processed in problem.js
- Processing user-inputted code as a string makes it easy for third party to inject malicious code and cause a security breach
Code Flaw #4: Pathname Resolution

While lines 141-147 work to produce the correct URL, it's not a safe way to retrieve the test pathname. Specifically, line 141 simply grabs the URL from the window, which will only work if the URL is `/problems/(id)/`. This works for the current manifestation of the project, but it is not safe for extension, and one could imagine a future iteration of the project where a problem might be shown on a page without the URL `/problems/(id)/`. This would cause this code to break. A possible solution to this problem is to add the pathname in the HTML for the problem in a hidden input field.

```javascript
var path = window.location.pathname;

// If the address has a trailing slash, don't include it in the ajax call
if (path.slice(-1) === '/') {
    path = path.substring(0, path.length - 1);
}

$.get(path + '/tests', function(data) {
    $.parseJSON(data).forEach(function(testCase) {
        currentTestId = testCase.id;
        currentInput = testCase.input;
        currentExpectedOutput = testCase.correct_output;
        run();
    });
});
```