

Refactor Vocational-Test:

- Refactor app.rb :

1. Se eliminan los paréntesis en las ocurrencias donde se llamaba a 'x' metodo y no precisaba de parámetros, tanto en la declaración del método y cuando es invocado.

<pre>42 - def getPoints() 43 outcomes = Outcome.all 44 responses = Response.all 45 end</pre>	<pre>46 + def getpoints 47 outcomes = Outcome.all 48 responses = Response.all 49 end</pre>
<pre>21 - points = getPoints()</pre>	<pre>22 + points = getpoints</pre>

2. Se quitan las extensiones de los archivos requeridos, a través del require.

<pre>1 - require './models/init.rb' 2</pre>	<pre>1 + require './models/init' 2</pre>
---	--

3. Se cambió la forma de declarar las variables,funciones,etc ,aplicandoles la técnica snake_case.

<pre>17 - clearCache() 18 19 - createResponsesFromChoices(data['choicesSelected'])</pre>	<pre>18 + clear_cache 19 20 + create_responses_from_choices(data['choicesSelected'])</pre>
--	--

4. Se determinó el limite max de una línea como 140, donde tuvimos que hacer varios ajustes en distintas instancias.

<pre>66 - if key == career.id 67 - results.push({career_id: career.id, name: career.name, points: 68 - points}) 69 - end</pre>	<pre>68 + next unless key == career.id 69 + 70 + results.push(71 + { career_id: career.id, name: career.name, points: points } 72 +) 73 + end 74 + end</pre>
--	--

5. También se aplicaron shorthand , donde un ejemplo concreto:

a count = count + 1, paso a la declaración siguiente count += 1

<pre>123 - if career.career_id == careerSelect['careerId'].to_i 124 - count = count + 1 125 - end 126 - end</pre>	<pre>127 + count += 1 if career.career_id == career_select['careerId'].to_i 128 - end</pre>
---	---

6. Se re-acomodaron estructuras condicionales ,donde era redundante la forma de declaración que estábamos llevando a cabo, cuando la condición bastaba con declarar una sola línea.

<pre>123 - if career.career_id == careerSelect['careerId'].to_i 124 - count = count + 1 125 - end 126 - end</pre>	<pre>127 + count += 1 if career.career_id == career_select['careerId'].to_i 128 - end</pre>
---	---

7. Se intercambiaron los tipos de comillas a la hora de instanciar un string, se utilizaban comillas dobles, y el cambio propuesto fue el de utilizar simples para los casos solicitados y que no necesitaban interpolación de cadenas.

Al igual a la hora de declarar la ruta del endpoint.

<pre>14 - post "/finish_survey" do 15 data = JSON.parse request.body.read 16</pre>	<pre>15 + post '/finish_survey' do 16 data = JSON.parse request.body.read 17</pre>
<pre>154 elsif user.password != params[:password] 155 @error = "Password incorrecta" 156</pre>	<pre>156 elsif user.password != params[:password] 157 @error = "Password incorrecta" 158</pre>

8. En un caso en particular se intercambió un if que era condición de continuación, se utilizo el next unless.(línea 68).

<pre>64 - careersPoints.each do key, points 65 careers.each do career 66 if key == career.id</pre>	<pre>66 + careersPoints.each do key, points 67 careers.each do career 68 next unless key == career.id</pre>
--	---

9. Se removieron Return redundantes, dejando así la variable o lo que fuese que retorna la función ,solo, al final.(línea 76 ejemplo)

59 - def getCareersWithPoints(careersPoints)	61 + def getCareersWithPoints(careersPoints)
60 careers = Career.all	62 careers = Career.all
61	63
62 results = []	64 results = []
63	65
64 - careersPoints.each do key, points	66 + careersPoints.each do key, points
65 careers.each do career	67 careers.each do career
66 - if key == career.id	68 + next unless key == career.id
67 - results.push({career_id: career.id, name: career.name, points: points})	69 +
68 - end	70 + results.push(
	71 + { career_id: career.id, name: career.name, points: points }
	72 +)
69 end	73 end
70 end	74 end
71	75
72 - return results	76 + results
73 end	77 end