PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2513 – Tecnologías y Aplicaciones Web (II/2016)

Profesor: Raúl Montes

Interrogación 2

7 de octubre de 2016

Ejercicio 1 (30 %): Preguntas conceptuales

Responde en forma precisa y concisa las siguientes preguntas:

1. (1.5 ptos) Suponiendo que dos reglas CSS estén especificadas en el mismo archivo CSS y ambas seleccionen el mismo elemento HTML pero aplicando diferentes valores para una misma propiedad. Explica cómo el *browser* decide el valor final de esta propiedad "en conflicto" para el elemento en cuestión, suponiendo que quien leerá la explicación sabe cómo interpretar una regla, pero no tiene idea de cómo se determinan los valores con múltiples reglas en conflicto.

2. (1.5 ptos) Imagina una aplicación que cuenta con un lista de categorías de productos y que permite a sus usuarios **autenticados** suscribirse a cada categoría por separado, recibiendo avisos por e-mail por cada nuevo producto agregado a la categoría. Esta funcionalidad se ofrece mediante un pequeño formulario al lado de cada categoría, que cuenta con un *select*/*combobox* para seleccionar la frecuencia de los avisos (diario, semanal, mensual) y un botón "Suscrbirse". Al presionar el botón, quedas suscrito.
   Explica, en términos de rutas, cómo verías esta funcionalidad de una manera *resource-oriented*, y escribe lo que necesitarías en el archivo `routes.rb` para recibir el request del formulario mencionado.

3. (1.5 ptos) Un amigo te cuenta que cambió la configuración para mantener información de sesión (por ejemplo, `user_id` del usuario actualmente "logueado") en una aplicación Rails. En lugar de utilizar la configuración *default* que almacena estos datos en *cookies*, lo configuró para almacenar la información mediante ActiveRecord (y base de datos). Y luego te dice: "así que ahora ya no estoy utilizando *cookies* en mi aplicación". ¿Es correcta esta última afirmación? Fundamenta explicando cómo funciona el manejo de sesión en este caso.

4. (1.5 ptos) En el contexto del desarrollo de una aplicación Web habrán muchos momentos en que tendrás que decidir si repetir/copiar-pegar código o si, de alguna manera, mantenerte DRY (y ya sabemos cuál será tu elección, ¿no?). Menciona y explica dos elementos del framework Rails, **relacionados a las vistas**, que te ayudan en aquéllos momentos.

Introducción para ejercicios prácticos

El proyecto "Datos Comparados de Candidatos" (DCC) busca adelantarse a los resultados de las próximas elecciones municipales mediante diferentes técnicas de análisis de datos (patente pendiente). Una de sus fuentes de datos es una encuesta web que realizan por comuna. Ya tienen el modelo implementado con las siguientes clases `ActiveRecord`:

```ruby
class Municipality < ActiveRecord::Base
  # name: string
  has_many :answers
end

class Candidate
  # name: string
  belongs_to :municipality
end

class Answer < ActiveRecord::Base
  # will_vote: boolean
  # voted_past_election: boolean
  # politics_satisfaction: integer
  belongs_to :municipality
  belongs_to :preferred_candidate, class_name: 'Candidate'
end
```

En los comentarios de estas clases puedes encontrar los atributos relevantes (además de los usuales `id` y *timestamps*).

Ejercicio 2 (35 %): Recopilando los datos

El objetivo del DCC es que con ciertos links que difundirán mediante elaboradas técnicas de *spam*, los encuestados en potencia lleguen a una URL de la forma `/municipalities/42/answers/new`, en donde `42` corresponde a un `id` de una comuna específica (similar URL existe para todas las comunas).

Tu misión es escribir el código de *controllers* y *templates* (sin considerar *layout*) que genera y procesa el formulario de la encuesta. Debes suponer las prácticas recomendadas de Rails respecto a qué implementar y cómo hacerlo.

La encuesta considera las siguientes preguntas y el tipo de respuesta que se solicita en cada una:

- ¿Piensa votar en las próximas elecciones?: respuesta sí o no (boolean)
- ¿Votó en las elecciones pasadas?: respuesta sí o no (boolean)
- De 1 a 100, ¿qué tan satisfecho está con los políticos actuales?: respuesta número de 1 a 100
- ¿Por cuál de estos candidatos de su comuna votaría Ud.?: selección única desde una lista de candidatos de la comuna indicada.

Al final de las preguntas incluye un votón "Enviar respuesta" que enviará el formulario. Ante esa acción, la aplicación debe almacenar la información recopilada en el modelo descrito anteriormente y finalmente redirigir al usuario al *home* de la aplicación (cuyo *alias* de ruta es *root*).

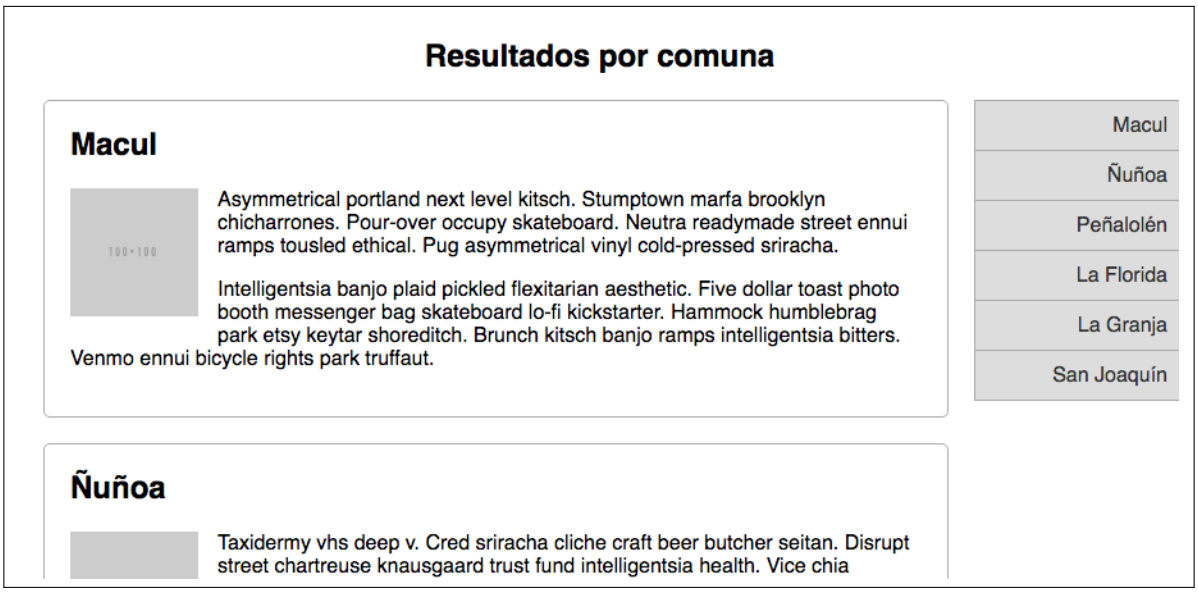
Notas:

- No es necesario que te preocupes de las validaciones asociadas; puedes suponer que no habrá errores al persistir modelos.
- Usa los elementos de formulario que parezcan adecuados dependiendo de la pregunta y el tipo de dato a almacenar.

Ejercicio 3 (35 %): Concluyendo con CSS

Luego de procesar los datos con sus algoritmos, DCC querrá publicar una página con las conclusiones por comuna. Es una página que contiene un recuadro por cada comuna, con un gráfico (puedes suponer una simple imagen) a la izquierda y una explicacion de las conclusiones. A la derecha de estos recuadros hay un menú con lengüetas por comuna que sirven como acceso directo a la comuna que quieras.

DCC ya tiene un diseño de la página, pero necesita tu ayuda para escribir el HTML y CSS que hará que se vea como en la imagen entregada por el diseñador (ver imagen incluida).



Las conclusiones por comuna usan el 80 % del ancho de la página mientras que las lengüetas el 20 % restante. El borde externo es de la imagen, representando el *viewport*, no es parte de lo que tienes que implementar.

Notas:

- Tú eliges si usas ERb o simple HTML, y si usas SCSS o CSS directamente.
- No necesitas escribir todo el contenido que se ve en la imagen en tu HTML/ERb; concéntrate en la estructura y usa *placeholders*.
- Aunque tengas sólo una muestra del contenido, debes considerar que tu implementación funcione como se desea suponiendo todo el contenido que se espera tener.
- Si usas ERb puedes suponer un modelo asignado desde el controlador que tenga sentido para generar esta página (con la URL de la imagen, el texto de análisis, nombre de comuna...).