#recoderschallenge

Ensayo prueba de certificación Talento Digital –

Organizado por ReCoders

Perfil Full Stack Java

Contexto

"Amores Caninos" es una iniciativa creada con el objetivo de generar un espacio en donde nuestros amigos de raza canina puedan encontrar a esa pareja que les haga sentir como perro con dos colas. Sus comienzos fueron muy humildes; los miembros fundadores iban de casa en casa en sus barrios, preguntando a cada persona si estaban interesados en que sus compañeros caninos encontraran pareja, y recopilando sus datos. Su idea fue ganando apoyo con el tiempo, y los miembros recurrieron a distintas metodologías para manejar la gran cantidad de solicitudes que recibían y dar una atención oportuna a las personas interesadas. Grupos en redes sociales, hojas de cálculo y entrevistas de satisfacción a domicilio estuvieron en su set de herramientas y técnicas por mucho tiempo.

Hace muy poco, se dieron cuenta de que su idea había escalado mucho, pero sus metodologías ya no estaban a la altura. Los grupos en redes sociales eran un caos, el emparejamiento con datos en hojas de cálculo era engorroso, y no había tiempo ni personal que diera abasto para hacer entrevistas a domicilio. Necesitaban algo más eficiente para poder seguir con su idea. Es así como, después de analizarlo mucho, llegaron a la conclusión de que era necesario implementar un sistema que pudiera centralizar los emparejamientos y hacer el proceso más rápido y sencillo, tanto para ellos como para sus usuarios.

Y ahí es donde entras tú. Debes diseñar y construir un sitio web que haga realidad la funcionalidad más básica de este sistema: el emparejamiento. Este es tu momento para demostrar lo que has aprendido y crear algo que puede ayudar a otros a ser más felices. ¡Todos los perros del mundo (y sus compañeros humanos) te están apoyando!

Requerimientos

Debes crear un **sitio web** que permita que un amigo canino encuentre a su pareja soñada, tomando en cuenta distintas características y preferencias. Para mantener las cosas lo más simple posible, el "emparejamiento" consistirá en un sistema de búsqueda, que mostrará los perfiles que se ajustan a ciertas preferencias tales como la edad, la raza, el género y otras características, para que el usuario escoja el perfil que

más le llame la atención. El sitio web debe cargar los datos de los perfiles y demás desde una base de datos.

El sitio web debe estar compuesto por tres pantallas (o páginas): una pantalla de **inicio**, una de **selección de preferencias**, y una de **resultados de búsqueda**. No es necesario implementar las funcionalidades de inicio de sesión o de registro. Todas las pantallas deben tener un encabezado (header) y un pie de página (footer) a gusto. ¡Sé creativ@! Además, el sitio web debe adaptarse a los tamaños de los diferentes dispositivos, ya sean celulares, tablets, o equipos de escritorio.

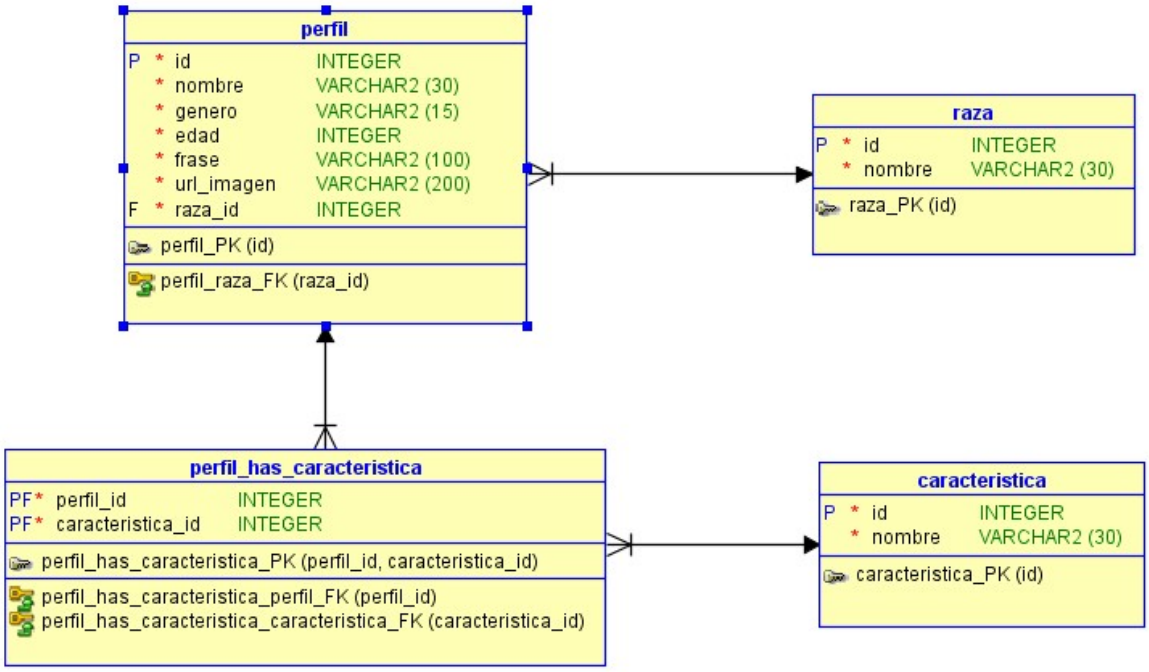
La **pantalla de inicio** debe tener un botón que lleve al usuario hasta la pantalla de selección de preferencias. Puedes agregar más descripción y/o imágenes para que la página se vea más llamativa, pero es mejor que te concentres en la funcionalidad. Con que el botón funcione, está bien.

La **pantalla de selección de preferencias** deberá contener un formulario donde se puedan elegir las distintas preferencias: raza, género, edad y características (o rasgos de personalidad). Las razas y características disponibles deberían cargarse dinámicamente, desde la base de datos. La edad debe estar expresada en años. El formulario debería permitir seleccionar solo una característica, ya que seleccionar más de una puede complicar demasiado las cosas (si aceptas el desafío de filtrar perfiles usando más de una característica, eres bienvenido a intentarlo). Sería bueno si agregaras un botón para volver a la página de inicio, por si los usuarios se arrepienten. También debes agregar un botón separado del formulario, que muestre todos los perfiles en la pantalla de resultados de búsqueda, sin aplicar filtros. Por último, el formulario debe tener un botón que envíe las preferencias seleccionadas hacia el servidor y muestre la pantalla de resultados.

La **pantalla de resultados de búsqueda** debe mostrar un listado con los perfiles que cumplan con las preferencias seleccionadas en la pantalla anterior. Si no hay resultados que coincidan con las preferencias seleccionadas, se debe indicar esa situación a través de un mensaje. Como mínimo debería mostrar el nombre, la raza, el género, la frase y la edad de cada perfil. Podrías mostrar las características que posee cada perfil también. Además, en cada perfil hay una imagen que podrías mostrar junto con la información, pero *cuidado*, ya que los tamaños de las imágenes no están estandarizados (si aceptas el desafío de mostrar las imágenes, deberás ingeniártelas para que todas las imágenes se muestren bien, independiente de su tamaño). Por último, también agrega un enlace para volver a la pantalla de inicio.

Datos a utilizar

La estructura de la base de datos que debes utilizar es la siguiente:



Las columnas de identificación en cada tabla deben tener un mecanismo de autoincremento. Hay que destacar que en la tabla de *perfil*, el campo *edad* viene expresado en años, y que el campo *url_imagen* contiene una url con la ruta hacia la imagen de perfil. El campo *frase* solo es una frase para dar impresión, nada más.

El script para cargar los datos iniciales vendrá adjunto a este documento, en el archivo amorescaninos_datos.sql. Deberías echarle un vistazo al menos una vez para que entiendas la estructura de la base de datos y los datos a los que corresponde cada campo.

Consideraciones

Este ensayo está destinado a ser desarrollado usando el lenguaje Java y el framework Spring, pero si quieres desarrollar la idea usando otros lenguajes o tecnologías, eres libre de hacerlo. Si vas a desarrollar usando Java, ten en cuenta las siguientes consideraciones:

- Puedes usar Spring Framework o Spring Boot a tu elección, usa el que más te acomode. Asimismo, puedes usar Eclipse o Spring Tool Suite para desarrollar el proyecto, dependiento de tu elección de framework.
- Puedes usar JdbcTemplate o JPA para el tratamiento de datos. Sugerimos que uses JPA, si te sientes con confianza de usarlo.
- La base de datos que debes usar es Oracle. Puedes usar otra si quieres, pero es mejor que uses la recomendada. Sí, sabemos que no quieres, pero es mejor que lo hagas, confía en nosotros :D.
- El motor de plantillas web *debe* ser JSP. Para este ensayo, no deberías utilizar motores como Thymeleaf o similares. Sabemos que son mucho mejores, pero usar JSP es un requisito para la prueba. Lo sentimos mucho.
- Para dar responsividad a tus páginas web, debes usar Bootstrap. Es un requisito para la prueba.
- Para generar dinamismo en las páginas web, puedes usar Javascript puro (vanilla) o jQuery, o ambos combinados. Recomendamos usar jQuery.

Reglas

Y aquí llegamos a las reglas. Esto no es un concurso, competencia o algo parecido, pero ya que se trata de una actividad orientada a reforzar tus conocimientos y ponerte a prueba, seguir estas reglas ayudarán al objetivo de este ensayo:

1. La creación del código fuente de este ensayo debe hacerse de forma **individual**. Se permite formar grupos, el intercambio de ideas y opiniones con otros compañeros, conectarse a través de canales de audio u otros, y compartir material y recursos de estudio, pero **NO compartir código.** El objetivo de esto es que te acostumbres a escribir código, a usar las herramientas que te brindan los IDE y a solucionar los errores más típicos por tu cuenta, haciendo tú mism@ los cambios en el código fuente o las configuraciones.
2. Este ensayo **no tiene hora o fecha de entrega**, y puedes hacerlo a tu propio ritmo, cuando tengas tiempo. Sin embargo, el **Jueves 19 de noviembre de 2020 a las 21:00 hrs.** se revisará una solución a este ensayo (preparada de antemano por Jaime Reyes), así que deberías tenerlo listo antes de esa fecha y hora. Para medir tu progreso y rapidez como forma de autoevaluación, considera que debes tener listo lo que exige este ensayo **antes de 4 horas** de haber iniciado, efectivamente dedicadas al proyecto (si tuviste interrupciones mientras estabas desarrollando el proyecto, descuéntalas).

3. Si decidiste desarrollar el proyecto acompañado de un grupo, para compartir ideas, considera que el **grupo no debe ser superior a 5 personas**. Un número superior invocaría demasiado al caos. Lo ideal es que sean grupos entre 2 y 3 personas.
4. Si tienes ganas de hacer el ensayo en grupo pero no tienes compañeros, **puedes contactar a personas por la comunidad ReCoders** en Discord, a través del canal *#comunidad*.
5. Todos los grupos y miembros son **libres de usar los canales de ReCoders** en Discord para comunicarse y coordinarse, o intercambiar info entre grupos con respecto al ensayo. Los canales para esto serán los canales de audio y *#java.*
6. Si quieres **compartir tu proyecto** una vez que lo termines, ¡bienvenid@! Te sugerimos subirlo a Github o algún repositorio remoto, y que compartas el enlace por el canal #compartir_código. Deberías compartirlo solo **después de que se revise la solución**, para evitar que posibles "espías de código" se vean tentados a romper la regla #1 de este ensayo.

Bonus track (para después del ensayo)

¿Lograste terminar el ensayo y te pareció demasiado fácil? ¿Quedaste con hambre de más? Si es así, esta sección te puede interesar. He aquí algunas funcionalidades que puedes agregar a tu proyecto una vez que termines los requerimientos del ensayo, para que siga creciendo y se convierta en el sitio de amores caninos definitivo:

1. Completar los desafíos que se incluyeron dentro de los requerimientos base, si es que no lo has hecho ya.
2. Generar una API que ofrezca la misma funcionalidad del sitio web, recibiendo las preferencias en formato JSON, y devolviendo la lista con los perfiles en ese mismo formato.
3. Agregar las validaciones que correspondan al formulario de búsqueda, en cuanto a edad y cantidad de características seleccionadas.
4. Implementar los mecanismos de registro e inicio/fin de sesión. Esto requiere modificar la base de datos, ingresar los datos de usuario necesarios, y asociar cada usuario a algún perfil existente. SI estás usando Java, puedes apoyarte en Spring Security.
5. Habilitar la funcionalidad de crear, editar y borrar perfiles. No olvides que cada perfil debe estar asociado a un usuario de la aplicación.

6. Agregar metadatos a los perfiles, tales como fecha de creación y actualización. Para esto, debes editar la estructura de la base de datos, y agregar los datos que falten. También estos metadatos deberían generarse de forma automática a la hora de registrar nuevos perfiles y editarlos.
7. Crear una pantalla que se parezca más a los sitios y apps de emparejamiento en la vida real. Es decir, en vez de que muestre una página de resultados con todos los perfiles que coinciden, muestre el nombre y la imagen de un solo perfil cada vez, junto con un botón para pasar al siguiente perfil y otro para "conocer más", que muestre más detalles del perfil.
8. Agregar a cada perfil un campo de descripción u otros campos que te parezcan interesantes de agregar y de mostrar. Sí, hay que modificar y rellenar la base de datos :P.
9. Aplicar un algoritmo de emparejamiento más robusto, que considere las preferencias de ambos perfiles. Es decir, que cada perfil en el sitio debe tener sus propias preferencias (características que buscan de otro perfil, como edad, raza, etc.), y cuando haya una correspondencia mutua entre ambos perfiles y las preferencias que buscan, se habrá encontrado una pareja compatible. Para lograr esto, hay que modificar la estructura de la base de datos, para que se registren las preferencias de cada perfil también, y agregar esos datos.
10. Implementar un sistema de chat para cuando dos perfiles conecten, y que así puedan programar encuentros, citas y conocerse en la vida real. Esto lleva mucho trabajo, pero si lo logras, habrás construido una aplicación web totalmente funcional.

Si lograste hacer todo hasta aquí, *¡felicidades, eres máster en desarrollo web y estás más que preparad@ para la prueba!*

¡¡MUCHO ÉXITO!!

Preparado con mucho cariño por ReCoders, para todos los miembros de la comunidad