Tarea 2
Bases de Datos

Andrea Figueroa R.
abfiguer@alumnos.inf.utfsm.cl

Camilo Rivas F.
camilo.rivas@alumnos.usm.cl

Camilo Valenzuela C.
camilo.valenzuela@alumnos.usm.cl

Javier Jeria M.
javier.jeria.13@sansano.usm.cl

Cecilia Reyes C.
reyes@inf.utfsm.cl

23 de mayo de 2016

Introducción

En un viaje realizado por algunos de sus ayudantes a la meca del consumismo para realizar una inversión en su apariencia y que no los traten de vagabundos por la calle. Se dieron cuenta de que las grandes tiendas del retail no ofrecen gran variedad de ropa o que las tiendas vendian prendas a precios muy altos.

Defraudados de esta realidad, decidieron crear un sitio, para reunir a las pequeñas PYMEs y darles una posibilidad de mostrar sus productos. Con esto, además de ayudar a la economía chilena, sus queridos ayudantes podrán comprar prendas de calidad a bajos precios para parecer unos verdaderos futuros ingenieros.

Descripción del sistema

Se tiene un mall virtual, donde los administradores de cada tienda se registran en el sistema, ingresando sus datos. Luego de registrados, pueden crear tiendas virtuales especificando el tipo de tienda de una lista de posibles categorias predefinidas.

En cada tienda se tienen distintos productos. Para cada producto, además de la información básica (título, descripción, etc), se debe especificar una o más categorias que lo definen. Cada producto tiene un stock que puede ser aumentado por el administrador o disminuido mediante la compra.

Las personas que ingresan al portal pueden ver todas las tiendas que estan en el mall virtual, así como ver los productos de la tienda. Para realizar la compra de algún producto, deberán registrarse en el sistema primero.

Para registrarse en el portal, el usuario tiene que ingresar su datos básicos (nombre, apellido, email, etc), un telefono de contacto y una dirección de contacto.

Modelo de Datos Lógico

Para esta tarea, cada grupo de tarea estará a cargo de diseñar el modelo de datos que pueda resolver esta problemática en función de los requerimientos y vistas del caso que se den más adelante. Recuerde que es muy importante el modelo de datos, ya que un mal modelo, puede llevar a muy malos resultados.

Requerimientos Técnicos

Entre los requerimientos técnicos, encontramos que se debe ocupar:

- Ruby: La versión a utilizar es 2.1.2.
- Ruby on Rails: framework a utilizar. Se pide instalar al menos la versión 4.1.4. Si por error instalaron una versión distinta, por favor notificar para tenerlo en consideración, puesto que pueden haber problemas de compatibilidad con distintas versiones.
- SQLite: Es la base de datos por defecto de Ruby on Rails. Trabajaremos con esta para simplicidad del problema.
- Uso de Gemas: Existe una gran variedad de 'código listo' para su reutilización. Se pedirá el uso de al menos la gema Devise[1]. Cualquier otro uso de gemas que aporte a la solución, será bienvenido.

Trabajar con Ruby on Rails implica trabajar con el patrón de diseño Modelo-Vista-Controlador[2], por lo que es recomendable que estudien cómo funciona este patrón de diseño para saber como utilizar correctamente el framework a utilizar.

Modalidad de Trabajo

El trabajo de esta tarea será dividido en 6 sesiones, todas las semanas se subirá a moodle un documento con los requisitos funcionales que deben cumplir en cada sesión, la cual tendrá el siguiente formato.

1. En los primero minutos de la sesión se explicará que es lo que se debe hacer y de ser necesario se hará un pequeño demo.
2. Los integrantes del equipo dividirán el trabajo de la forma recomendada en el documento de la semana.
3. En los últimos 15 minutos de la sesión se evaluará si se cumplió el objetivo requerido y se pondrán los puntajes a cada grupo.

Cada sesión tendrá una nota de 0 a 100, la nota final de la tarea será el promedio aritmético de todas éstas, existirá una sesión recuperativa al final, la cual estará al nivel de la última sesión, ésta nota reemplazará la peor sesión.

Condiciones Generales

- Estimen cuál será la mejor forma de resolver los problemas que pueda notar. Su decisión adjúntela en un documento llamado "supuestos" (en el formato que como equipo estimen conveniente), en donde además podrá explicar cosas que su ayudante deberá tener en cuenta al momento de revisar su tarea. Dado el formato de esta tarea es EXTREMADAMENTE IMPORTANTE el desarrollo de este documento.
- Se aplicarán descuentos en el caso de existir links rotos (5 puntos por cada link roto).
- No deben aparecer mensajes de error por pantalla, provenientes de la lógica de negocio o del DBMS, trate los mensajes de manera que un usuario entienda.

[1] https://github.com/plataformatec/devise
[2] http://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador

Consideraciones de Entrega

- **Sobre el trabajo**: La entrega de cada sesión debe ser realizada en grupos de dos personas previamente inscritos, los mismos de la tarea 1.
- **Almacenamiento**: Su equipo deberá crear un repositorio GIT (**https://gitlab.labcomp.cl/** con el nombre **bdXX-2016-1**) en el cual se realizará la entrega de su trabajo. Cualquier problema relacionado con la utilización del repositorio debe ser hablado en el Laboratorio de Computación.
- **Presentación**: Se prohíbe usar Flash, tendrá nota 0 si incluye animaciones de este estilo. No existen restricciones en cuanto al uso de JavaScript, Ajax, etc.
- **Sobre actitudes indebidas**: Se sancionará con la nota **cero** y sin derecho a reclamo, cualquier situación que no corresponda durante el desarrollo de la tarea. Esto incluye la copia, o utilización indebida o no autorizada de cuentas que no correspondan a las asignadas por su ayudante.

Restricciones

- Implementación de lógica de autenticación, es decir, que los participantes no puedan ingresar a la zona de administradores, y viceversa. Se solicita ocupar la gema Devise para un fácil manejo de esta lógica.
- Interfaz simple e intuitiva. Se recomienda uso de frameworks para front-end, tales como: Twitter Bootstrap8 , Flat UI9 o Foundation10. Se evaluará con nota 0 el uso de contenido Flash en su sitio (En otras palabras, no pueden usar Flash).
- Ocupar scaffolding en Rails permite generar mucho código que facilita el desarrollo de aplicaciones, sin embargo, mucho de este código no lo van a utilizar, por lo que se descontarán 50 puntos por no borrar las vistas y lógica que no sean necesarias y hayan sido generadas utilizando scaffolding. En otras palabras, se pide que si utilizan scaffolding, lo hagan con criterio.