Tarea 2
Bases de Datos
Rubíes en los rieles

Cecilia Reyes C.
reyes@inf.utfsm.cl

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

Diego Plaza S.
dplaza@alumnos.inf.utfsm.cl

Alejandro Díaz O.
ajdiaz@alumnos.inf.utfsm.cl

Sebastián González M.
sebastian.gonzalezm@alumnos.usm.cl

22 de mayo de 2013

1. Introducción

El AngelHack llegó a Chile finalmente, busca ser una plataforma para emprendedores tecnológicos. Se está invitando a participar en una jornada de 2 días donde se puede completar algo que generalmente demora semanas de trabajo y dedicación. Los participantes se reúnen en equipos de experiencia interdisciplinaria: desarrolladores, un diseñador y un participante no-técnico. A veces se presentan participantes sin equipo y buscan gente que esté *guachita*, lo que causa un problema para la organización, pues debe llevarse un registro de los proyectos que participan, con todos los datos de sus integrantes. Convenientemente, los organizadores del AngelHack les pidieron a los ayudantes de Bases de Datos que los apoyaran. Ellos asintieron de inmediato, argumentando que tienen muchos motivados estudiantes, dispuestos a realizar el trabajo a cambio de una nota, como la evaluación de la segunda tarea.

La aplicación a construir debe permitir el flujo de información para dos tipos de usuarios diferentes: los organizadores y los participantes. Los últimos deben poder inscribirse y ser asignados a equipos con menos de 4 integrantes. Los administradores deben poder evaluar a los equipos, ver su información y asignar un ganador.

2. Modelo de datos

Una de las filosofías de AngelHack es ser creativo. Es por esto, que ellos no quieren entregar un modelo de datos en particular, y lo dejan a libre elección de ustedes, los desarrolladores. Se espera que el modelo se cree a partir de los requerimientos funcionales indicados en el punto 4. Recuerde que un mal modelo de datos, puede traer grandes consecuencias en el producto final.

3. Requerimientos técnicos

Los organizadores de AngelHack ponen algunos requisitos, ya que éstos, ya tienen servidores configurados, y por tiempo y ganas, no quieren y no pueden instalar más entornos:

- Ruby [1]: En el servidor de producción de AngelHack está instalada la última versión estable de Ruby, la cual corresponde a la versión 2.0.0-p195, por lo tanto, el sistema DEBE desarrollarse bajo esta versión.

[1] http://www.ruby-lang.org/es/

- Ruby on Rails [2]: Por razones de seguridad, y debido a que las últimas versiones de RoR tienen mejoras y menos vulnerabilidades, se pide usar la versión 3.2.13.
- SQLite: RoR puede complementar variadas bases de datos, pero en este caso, a pedido de los organizadores, debe ocupar SQLite, el cual es la base de datos por defecto de este framework.
- Uso de gemas [3][4]: se pide encarecidamente no rehacer la rueda. El uso de gemas ahorra considerablemente el tiempo de desarrollo. Busque, lea, infórmese, configure y ahorre tiempo.

4. Requerimientos Funcionales

Los administradores requieren mantener el control de la información de los grupos de participantes. Cada grupo está compuesto por un máximo de 4 personas: un diseñador, un participante no-técnico y dos desarrolladores. Uno de ellos actuará como líder de grupo, el cual tiene el papel de inscribir a todo su equipo.

Como se mencionó anteriormente, algunos participantes llegan con un grupo o parte de éste, el cual comparte una idea de proyecto, pero también existe la posibilidad de participantes *guachitos*, sin ideas y sin grupo.

AngelHack se realizará en dos fases: la de inscripción y la de competencia. La fase de inscripción consiste en el registro tanto de los grupos ya formados, como el registro de los participantes *guachitos*. Una vez inscritos ambos tipos de participantes, los líderes de equipo buscarán tantos *guachitos* como sea necesario, para agregarlos a su equipo, siendo el máximo 4 participantes (total de integrantes de un equipo). La fase de competencia consiste en la evaluación del trabajo de cada equipo por parte de los administradores. Cada día, éstos le asignarán una nota de evaluación a cada equipo, el cual refleja su trabajo, dedicación e innovación, entre otros.
Finalmente, al cabo de 2 días, y tomando de base las notas, se seleccionará a un ganador.

Específicamente se pide:

1. **Registro de participantes**: Los participantes *guachitos* deben registrarse en el sistema mediante Facebook, el cual debe entregar los datos tales como: nombre, email, fotografía y fecha de nacimiento. Luego del registro, el participante debe elegir su competencia. Los integrantes de un equipo formado previamente deben registrarse individualmente, entregando los mismos datos que los participantes *guachitos*.
2. **Login de Administradores**: Un administrador debe tener email y contraseña para su ingreso al dashboard. Éstos pueden ser registrados via seed.rb, manualmente desde la base de datos, o desde la consola del proyecto.
3. **Login de Participantes**: Tal como el registro, el ingreso debe ser mediante Facebook. Todos los participantes pueden ingresar a ver su ficha con sus datos e información del equipo.
4. **Registro de Equipos**: Los líderes de cada equipo (puede ser cualquier participante, pero sólo uno por equipo), deben poder inscribir a su equipo. Esto consiste en nombre del grupo, nombre de proyecto, y una corta descripción de éste. También debe poder agregar a todos sus integrantes al equipo. Para que el equipo pueda ser registrado, debe necesariamente tener 4 participantes inscritos. Un número menor a éste, imposibilita el registro del equipo.
5. **Asignación de notas**: Los administradores pueden evaluar el desempeño de cada equipo colocando notas de 0 a 100, siendo ésta, una por día. Es decir, por cada equipo, debe haber un total de dos notas.

[2] http://rubyonrails.org/
[3] http://rubygems.org/
[4] https://www.ruby-toolbox.com/

6. **Elegir ganadores**: En el dashboard de administradores debe haber un botón para el proceso de generar el ganador. Éste, debe resultar del mayor promedio de notas entre todos los equipos participantes.

7. **Admin Dashboard**: El administrador debe contar con un panel de administración donde se puedan asignar notas a cada equipo, y generar el equipo ganador. Además, éste puede ver la información de todos los equipos, y sus integrantes.

8. **Dashboard de participantes**: Los participantes ingresan a su panel donde revisan su información personal, información del equipo y sus notas por día.

5. Consideraciones

5.1. Consideraciones Generales

- Dada la dinámica forma de trabajar en esta tarea, no se encasillará el trabajo con ningún modelo en particular. La idea principal de esta tarea es proveer al alumno la capacidad de trabajar con plataformas que permitan desarrollar aplicaciones de manera flexible.
- 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.

5.2. Consideraciones de Entrega

- La fecha de entrega es el **Lunes 24 de Junio**, hasta las 23:59 horas, inamovible.
- Sobre el equipo de trabajo: La entrega debe ser realizada en equipos de 2 a 3 personas. Manteniendo los equipos de las tareas anteriores, salvo excepciones conversadas, las cuales deben ser enviadas al correo de Andrea Figueroa (máximo una semana tras la publicación de la tarea).
- Sobre el formato de entrega: Su equipo poseerá un repositorio GIT (http://forge.inf.utfsm.cl/bdxx-2013-1) el cual le permitirá gestionar un desarrollo coordinado y en general sin conflictos de versiones. Los ayudantes revisarán la última versión de lo que esté en su repositorio, sea cuidadoso de dejar una versión funcional en dicho repositorio. No se responderá por problemas de mala utilización del repositorio (en este sentido, las consultas se deben dirigir directamente al Laboratorio de Computación ubicado en el piso subterráneo del Departamento de Informática).
- Sobre el motor de bases de datos: Como se dijo anteriormente, para esta tarea utilizará el motor de bases de datos SQLite (sqlite3 gem). Verifique que su tarea sea capaz de generar el schema utilizando *rake*.
- Sobre actitudes indebidas: Se sancionará con la nota mínima 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.

5.3. Modo de Trabajo

- Se realizarán ayudantías introductorias a las tecnologías de Ruby y Ruby on Rails, pero estas cubrirán solamente lo básico del tema. Se busca que desarrollen las habilidades de buscar y leer documentación, recursos y demáses. Buenas fuentes de documentación abundan en Internet.[5][6][7]
- También se recomienda aprovechar las horas de laboratorios para realizar las consultas pertinentes. No dude en realizar consultas a sus ayudantes asignados o en la plataforma Moodle del curso (de esa manera todos los ayudantes podremos ver su duda).
- Un buen recurso es la guía *Getting Started* en el sitio *Guides Ruby on Rails* (en referencias), indica cómo configurar la base de datos inicialmente y también sobre Migraciones. Se recomienda completar el proyecto de prueba que se presenta en dicha guía.

5.4. 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.
- Para el registro y login mediante Facebook, se recomienda leer acerca de las gemas Koala[8] y/o Omniauth[9], las cuales pueden ser una gran ayuda para cumplir estos requisitos.
- Interfaz simple e intuitiva. Se recomienda uso de frameworks para front-end, tales como: Twitter Bootstrap[10], Flat UI[11] o Foundation[12]. Se descontarán 20 puntos por incluir 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 20 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.

[5] http://guides.rubyonrails.org/
[6] http://railscasts.com/
[7] http://railsforzombies.org/
[8] https://github.com/arsduo/koala
[9] https://github.com/mkdynamic/omniauth-facebook
[10] http://twitter.github.io/bootstrap/
[11] http://designmodo.github.io/Flat-UI/
[12] http://foundation.zurb.com/