Universidad Pública de Navarra
Curso 2021/2022

Gestión de Sistemas de Información

Guión de la Práctica 01

Marilin Vega Leon, Dpto. de Automática y Computación

1. Contenidos del documento

Este documento recoge la Práctica 01 de la asignatura Gestión de Sistemas de Información correspondiente al curso 2021/2021.

2. Desarrollo de la práctica

Cada grupo de alumnos debe completar los ejercicios de la Sección 4, tal y como se describe en la Sección 5. Nótese que la siguiente práctica no puede realizarse si la presente no está finalizada, por lo que la entrega es relativamente temprana.

Los alumnos deben tener los siguientes conocimientos antes de comenzar la práctica:

- Herramientas básicas de diseño orientado a objetos;
- Programación de Java, incluyendo listas, herencias, *interfaces*/clases abstractas y gestión de paquetes. También es recomendable tener conocimientos en el uso de listas indexadas y generación/uso de *hash codes*. Antes de comenzar la práctica, se recomienda a los alumnos que repasen los siguientes conceptos básicos:
  - Privacidad de métodos y miembros en clases Java;
  - Definición de constructores;
  - Clonado, comparación y asignación de objetos;
  - Tipos de colecciones en el paquete `java.util`;
  - Generación de documentación en estilo javadoc.
- Comprensión básica de diagramas de clases UML.

Los alumnos deben adquirir los siguientes conocimientos en el transcurso de la práctica:

- Conocimientos teóricos:
  - Análisis básico de requisitos y comprensión de texto;
  - Captura de un modelo de negocio teórico en una librería de clases;
  - Comprensión del rol de la lógica de negocio en un sistema;
  - Descomposición básica de un sistema de información en lógica de negocio y sistema de gestión de datos.
- Conocimientos técnicos:
  - Uso básico de IDEs (Netbeans o Eclipse);

- Generación de librerías para modelos de negocio;
- Generación de prototipos sencillos de SIs.

La asistencia al laboratorio, si bien recomendable, no es obligatoria.

3. Fechas de interés

Las fechas de interés para el desarrollo de la práctica son las que siguen:

- 15 y 22 de Septiembre: Dias asignados para las sesiones de laboratorio.
- 28 de Septiembre (14h): Fecha límite de entrega.
- 2 de Octubre (14h): Fecha límite del plazo extraordinario entrega (con penalización).

4. Lista de ejercicios

En esta práctica se debe implementar un sistema de información (SI) para un portal de recomendaciones culinarias y de ocio. Eso sí, el SI se limitará a las capacidades más básicas, como son almacenamiento (no persistente) de la información y acceso a los eventos más generales del flujo de información en la empresa. Estos eventos se reducen a altas/bajas/modificaciones de diferentes tipos de información, así como la consulta y comprobación de la consistencia de ciertos datos.

☞ *Este sistema será mejorado y expandido en las siguientes prácticas, por lo que es importante alcanzar todos los objetivos que se proponen. Y es necesario reducir la cantidad de apaños de baja calidad, inconsistencias y fallos, de cara a mantener la paz mental en las siguientes prácticas. En caso peor, estas prácticas servirán para entender el impacto de las soluciones chapuceras en el mantenimiento y expansión de un sistema.*

La política de la empresa se resume en estos hechos:

C01. El sistema almacena información de locales de ocio en España. Un local tienen un nombre y está en una localización concreta. (Localidad, Provincia, Calle y Número). No puede haber dos locales en la misma dirección. Los locales pueden almacenar una breve descripción de no más de 300 caracteres.

C02. Existen tres tipos de locales: Restaurantes, Bares y Pubs. Los restaurantes tienen un precio estimado de menú, y una capacidad máxima tanto en comensales totales como en comensales por mesa. Los Bares tienen una serie de especialidades que deben ser almacenadas como tags. Finalmente, el sistema guarda datos acerca de las horas de clausura y apertura de los Pubs.

C03. El sistema almacena información de usarios. Cada usuario tiene un nick único de al menos tres caracteres y una contraseña. Además, se debe almacenar la información de la fecha de nacimiento del mismo. No se permiten usuario de menos de 14 años. Cada usuario debe tener un perfil determinado, a escoger entre Propietario y Cliente.

C04. Un Cliente es un usuario de la página que puede acceder a la información de los locales de ocio. Además, puede hacer reviews de los mismos.

C05. Una Review es una una opinión dejada en el sistema por un Cliente acerca de un Local. Una Review incluye una valoración (entre 0 y 5 estrellas), un breve comentario (500 caracteres) y la fecha en que se produjo la visita. Cada Cliente puede hacer tantas Rewiews como quiera, incluso varias del mismo local, aunque no se permite valorar dos veces la misma visita.

C06. Un Propietario es un Usuario que puede poseer un Local. Un local debe tener entre 1 y 3 Propietarios, y un Propietario puede tener cualquier número de Locales.

C07. Un usuario puede realizar una Contestación a una Review, siempre y cuando el local al que se refiera sea suyo. No puede haber más de una contestación a una Review. Una review consiste simplemente en un breve comentario (500 caracteres).

C08. El sistema debe almacenar la fecha en que se generan tanto las Reviews como las Contestaciones.

C09. Tanto los restaurantes como los Bares son Reservables. El sistema debe poder almacenar Reservas de Clientes para Bares o Restaurantes. Cada Reserva incluye únicamente la fecha y hora en que se efectuará, así como un posible porcentaje de descuento.

En las 72 horas siguientes al comienzo de la primera práctica se abrirá una ronda de preguntas al respecto de los requisitos. Esto se debe a que, por más que se refinen, pueden resultar ambiguos u oscuros. Todas estas preguntas formularán vía el foro dedicado en MiAulario, se contestarán de manera individual, y serán recopiladas en un documento anexo que se colocará en los recursos de MiAulario.

Dados los requisitos anteriores, complete los ejercicios listados a continuación:

Ejercicio 1. Abra un IDE para trabajar en lenguaje Java (preferiblemente NetBeans). Cree un nuevo proyecto de Java con el nombre GSIGRXX, donde XX hace referencia al número de grupo asignado en las prácticas. Genere un paquete `GSILabs.BModel`, que de momento quedará vacío.

Ejercicio 2. Añada al paquete `GSILabs.BModel` las clases e *interfaces* que necesite para incluir la lógica de negocio incluída en los puntos anteriores. Se recomienda que implemente las clases que se icluyen en la Figura 1, ya que están de acuerdo a las políticas recogidas en la Sección 3. Todas las clases deben de estar adecuadamente comentadas y producir un *javadoc* completo.

De cara a aligerar la carga de trabajo, todos los `interface` de Java en el diagrama de la Figura 1 pueden descargarse de la sección de Recursos de MiAulario.

En el proceso tenga en cuenta los siguientes comentarios:

- Algunas clases no son identificables de manera unívoca a partir de sus miembros, pero es posible generar un identificador único de manera transparente. Clases como `AtomicInteger` pueden serle de ayuda en el proceso.
- Se recomienda que las clases instanciables sobreescriban los métodos `toString()` e `equals()`.

Para una comprobación inicial del contenido de este ejercicio se recomienda software de visualización de código fuente o *bytecode*, como puede ser ClassVisualizer. Este software permite navegar estructuras de clases a partir de las relaciones entre ellas, y generalmente también habilita la comprobación de los métodos y miembros de cada clase.

En este momento la lógica de negocio debe de estar completa. Sin embargo, no existe funcionalidad alguna al respecto de sistema de información. De alguna manera, se ha creado el diccionario del sistema, la información con la cual éste trabajará, pero no existe un soporte en el que almacenarla o del que extraerla. Esto se alcanza al completar el siguiente paquete: `GSILabs.BSystem`.

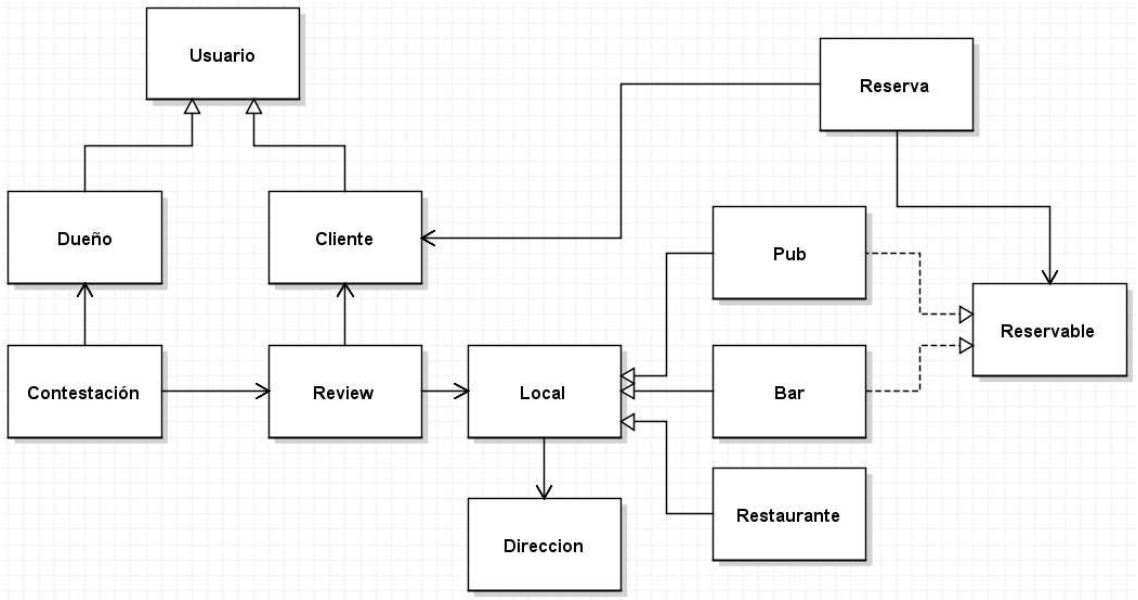

Figura 1: Diagrama general de clases del paquete `GSILabs.BModel`.

☞ *Es importante entender la diferencia entre la lógica de negocio y su* almacenamiento. *En el transcurso de la asignatura, ilustraremos como la información toma diferentes formas en cada capa o subsistema, aunque siempre representa el mismo hecho o elemento del modelo de negocio físico.*

Ejercicio 3. Descargue el `interface GSILabs.BSystem.LeisureOffice` de MiAulario. Cree una clase instanciable `GSILabs.BSystem.BusinessSystem` que implemente dicho `interface`. Para ello debe usar las clases del paquete generado en el Ejercicio 2, pero también puede valerse de tantas otras clases como necesite.

Ejercicio 4. Cree una clase ejecutable de nombre `GSILabs.BTesting.P01Tester`. En la ejecución se debe crear una instancia de la clase `GSILabs.BSystem.BussinessSystem`, en la que deben introducirse una cierta cantidad de locales, usuarios, reviews, reservas, etc. Use dicha clase para comprobar (y mostrar por linea de comandos) los siguientes sucesos:

S1) Si introduce a un usuario, este puede ser luego localizado a partir de su ID;

S2) Si busca a un usuario que no existe con `findClient`, el resultado es `null`;

S3) No se pueden introducir dos locales en la misma dirección;

S4) Si se añade un local, y se elimina posteriormente, se puede introducir un bar en la misma dirección;

S5) No se puede introducir un usuario menor de edad;

S6) No se pueden hacer reservas para un local inexistente;

S7) No se pueden hacer reservas para un local inexistente, aunque esté en la misma dirección que otro existente;

S8) No se pueden añadir comentarios para Reviews que no existen;

S9) No se pueden añadir cuatro dueños a un bar;

S10) No se pueden añadir dos reviews del mismo usuario, el mismo día para el mismo local.

Dentro del código de la clase, comente adecuadamente cómo comprueba cada uno de los sucesos.

5. Metodología de entrega y evaluación

Cada grupo de alumnos debe entregar en una carpeta comprimida el proyecto, bien sea de Eclipse o NetBeans (sólo se admitirán formatos *.zip o *.7z). La entrega de la misma se hará a través de una Tarea dedicada en MiAulario.

La nota se asignará, sobre un total de 12 puntos, en función de los siguientes criterios:

e1.1 El nombrado de proyecto y paquetes es adecuado (1 punto);

e1.2 Todas las clases del paquete `GSILabs.BModel` se han implementado de manera completa (3 puntos);

e1.3 El paquete `GSILabs.BSystem` es funcional de acuerdo a los requisitos expuestos en este documento y en el *interface* `GSILabs.BSystem.LeisureOffice` (3 puntos);

e1.4 El proyecto es capaz de producir un *javadoc* completo y significativo (2 puntos);

e1.5 La clase de testeo comprueba los *items* solicitados en el Ejercicio 4, mostrando los resultados en la linea de comandos (3 puntos).

Cada uno de los criterios podrá evaluarse como *fallido* (0 %), *incompleto* (25 %, 50 % o 75 %) o *completo* (100 %).

En caso de que el proyecto no sea entregado adecuadamente, se abrirá un *plazo de entrega extraordinario*. Por cada día de retraso en la entrega la penalización será de 3 puntos. De esta manera, un trabajo entregado 5 minutos tarde será penalizado con -3 puntos, y uno entregado 24h y 5 minutos tarde tendrá una penalización de -6 puntos.

Los alumnos podrán obtener puntos extra si alcanzan estos objetivos:

e2.1 Se ha implementado un control de errores por medio de excepciones documentadas y controladas (+2 punto);

e2.2 Se han sobreescrito los métodos genéricos (`toString()`, `equals()`,...) en todas las clases (+1 punto);

e2.3 Se han implementado medidas configurables de bloqueo de instancias o eliminación en cascada para mantener la coherencia de la información (+2 punto);

e2.4 El código esta comentado de manera apropiada (+1 punto);

e2.5 El grupo ha descargado el interfaz `LookupService`, disponible en el aulario, y ha hecho que `BusinessSystem` lo implemente (+3 puntos).

En cualquiera de estos casos los alumnos son responsables de documentar adecuadamente qué han tratado de hacer y cómo lo han conseguido.

Finalmente, cada grupo puede realizar opcionalmente un pequeño informe contestando a las preguntas que siguen:

e3.1 ¿Sería posible mantener la clase `BusinessSystem` y reemplazar todo el almacenamiento de información por una base de datos? ¿Qué efectos tendría esto en el funcionamiento de los programas que hacen uso de `BusinessSystem`?

e3.2 ¿Puede predecir problemas en relación a la consistencia de la información? ¿Se solucionarían estos empleando una base de datos?

Cada pregunta se valora en un máximo de 2 puntos, y debe contestarse en menos de 300 palabras. El formato de envío será un documento de nombre GSIP01GRXX.pdf, siguiendo las pautas básicas de sangrado, espaciado, justificación, etc. Este documento se adjuntará a la carpeta comprimida de la entrega. Cualquier respuesta que incluya faltas de ortografía será automáticamente valorada con 0 puntos. Dado el carácter no-obligatorio de este informe, se invita a no enviar informes que no estén bien meditados y escritos[1].

Control de versiones

Este documento no ha sufrido modificaciones desde su publicación (08/IX/21).

1. Esto quiere decir que la respuestas a las preguntas no deben ser *Iría más rápido* y *Sí, por supuesto*, respectivamente.