Evaluación Individual

24 de junio de 2013

Se cuenta con una aplicación web para administrar alumnos y cursos de una universidad. Permite listar cursos, listar alumnos, y agregar o quitar alumnos de un curso.

La aplicación está desarrollada en el lenguaje **Java**, y posee una arquitectura según **Domain Driven Design**, aplicando el patrón de inyección de dependencias a través de **Spring**. La capa web está implementada utilizando los patrones **MVC** y **Front Controller**, a través de **Spring MVC**. La vista está compuesta por archivos **JSP** con **JSTL** (no contienen código Java). Para la persistencia de los objetos de dominio se utiliza **Hibernate**, almacenando los datos en una base de datos **PostgreSQL**. Por último, la administración y construcción del proyecto se realiza mediante **Maven**.

Ninguna de las características mencionadas debe ser alterada.

Se pide:

1) Actualmente existe un servlet en la aplicación que se ocupa de redirigir a /bin/home/home cuando el usuario entra a /. Eliminar este servlet e implementar el mismo comportamiento sin utilizar servlets ni archivos .jsp. (1 punto)
2) Analizar la implementación de la matriculación/desmatriculación de un alumno en un curso y corregir los errores de diseño. (1.5 puntos)
3) Analizar la implementación del login. En caso de encontrar algún error de diseño, explicar el problema y corregirlo. En caso de no encontrar ningún error, agregar un comentario indicando que está implementado correctamente. (1 punto)
4) Cuando un alumno ya se encuentra matriculado en todos los cursos, se obtiene un error al hacer click en el botón “Matricular” en su perfil. Corregir este problema para que se muestre un mensaje de error acorde. (1 punto)
5) Implementar cupos en los cursos. Un curso debe tener una cantidad máxima de alumnos permitida. Sólo se debe permitir matricular un alumno a un curso si hay cupos disponibles. En caso de no haber cupos, se debe mostrar al usuario un mensaje de error indicando el problema. (2.5 puntos)
6) Se quiere tener un registro de las altas y bajas de alumnos producidas en cada curso a lo largo del tiempo. Para esto, cada vez que se matricula o se desmatricula un alumno de un curso, se debe registrar de alguna forma la fecha, el usuario que realizó el cambio, el alumno, el curso, y de qué tipo de operación se trata (alta o baja). Modificar la pantalla de detalle de un curso para mostrar esta información (ordenada por fecha descendentemente). (3 puntos)

Nota: En todos los ejercicios en los que se pide analizar y corregir un error se espera que se explique **detalladamente** el motivo del mismo (colocando la explicación como comentario en el código fuente).

Material a entregar

El código fuente se debe enviar en un archivo comprimido a la dirección de la cátedra, indicando en el título del mail “EVALUACION: <apellido del alumno>”.

Dentro del directorio **src/test/resources** existen los archivos **create.sql** e **insert.sql**, que contienen scripts para la creación de la base de datos y la carga de datos iniciales. Se debe agregar el archivo **update.sql**, que debe contener los scripts de alteración de la estructura de la base de datos.

Para que la entrega se considere válida, se debe poder ejecutar *mvn clean package* y los scripts de alteración de estructura de la base de datos.

Evaluación

En la evaluación se considerará el correcto funcionamiento de la aplicación (incluyendo scripts de migración) y la calidad del código entregado (esto incluye diseño de objetos, estilo de programación, adecuado ajuste a la arquitectura y patrones involucrados).

NO SE RETIRE DEL LABORATORIO SIN QUE SE HAYA CONFIRMADO
LA RECEPCION DEL MAIL DE ENTREGA.