Comentarios Correcciones 16/9/2017

A Java hay que decirle el conjunto de directorios para encontrar clases o archivos con el Class Path.

Para Testing se utiliza otro Class Path.

Eclipse sólo maneja un Class Path, entonces a mano hay que decirle el class path correcto.

Hay que importar de vuelta como eclipse Project y va a andar.

Manejando transacciones a mano, idea de los Test:

Base de memoria se descarta al cerrar la máquina virtual pero entre test y test persiste. Entonces los test son dependientes unos de los otros.

A los test hay que agregarle “extends abstractPersistenceTest”.

Cuando arranca otro test empieza otra sección, rollbackea automáticamente al final de cada test. Así los test van a ser realmente independientes.

Mirar annotation @beforeClass, JPA SCHEMAGENERATOR, SCHEMA EXPORT

Cambiar los nombres de las tablas a plural y cambiar los nombres de las columnas para usar guión bajo en lugar de escribir ejemplo “tipoDeDato”.

(no es necesario, solo informativo) Para realizar S = A + 1 \* 2, de la forma implementada son necesarias 5 consultas y un join bastante complejo. Lo mejor sería parsear y desparsear para evitar tantas consultas. Un converter puede serializar como String. Pero esto no es compatible con otra cosa que no sea java, en cambio la forma implementada sí. En la práctica no se persiste el árbol, sólo el string. Esto no hace falta cambiarlo, es sólo para tenerlo en cuenta como experiencia.

JPA EXTRA también tiene converter a Local Date.

En repositorios hay cosas para reutilizar seguro.

Clase singleton, una transacción para toda la aplicación.

Nunca una transacción dura la vida de un objeto, mucho menos de un singleton.

Transacción es un objeto transitorio.

Si hay un begin y un commit, debe haber un rollback (en un try catch).

Boiler plate -> código que es necesario pero puede generar problemas.

Mirar “implements transactionalOps” y usar en código withTransaction (en los test de contexto se usa), se le pone el código adentro y es equivalente pero mucho mejor a utilizar créate transaction, commit y rollback.

Lazy initialization sólo tiene sentido para objetos pesados y complejos.

Sincronizado es para acceso concurrente.

A createQuery(… , Empresa.class) se le pasa el tipo de lo esperado. Tenemos código muy parecido pero que cambia por los tipos. Esto se soluciona con generics (ver código de ejemplo recibido por email).

Queries más complejas where metodología.nombre =:nombre

Set parameter(“nombre”, nombre)

(se había realizado este comentario porque en el código en una query sql se utilizaba un alias para referenciar a la única variable que se utilizaba en la consulta).

@column es sólo para escalares, sino hay que usar joinedColumn.

Nunca utilizar arrays (no se puede cambiar el tamaño y tiene pocas operaciones), utilizar listas o collections.

No usar For, ni arrays en el código. En lugar de For utilizar streams.

No utilizar calendar, es un gran dolor de cabeza y encima es mutable. Es mejor local date, le pedís year y te devuelve tipo year, no entero.

Arrays.asList(empresa.longevidad())

El constructor público se hereda por defecto.

Super está implícito.

Minimizar los setters. Desde arena no se modifica las cuentas, entonces usar sólo get.

En aplicar condiciones hay un Fold de libro, hay que cambiarlo y buscar el equivalente en Stream.

Equals hashMap

Generate HashCode

Otra opción Apache commons.

Java objects.equals

Objects.hashcode

Nunca más hacer un hashcode a mano.

Importante borrar el código que no se utiliza (hay mucho código comentado, dijeron que lo iban a controlar en la próxima corrección), y realizar las abstracciones de repositorio.

Comentarios Correcciones 20/10/2017

Cuando se entra a parte que no debe entrar tiene que redireccionar al login

Bootstrp themes -> se puede descargar, viene página con todos los ítems y podés copiar el código. Tenés CSS de ese tema. Buscás páginas de temas que se adaptan a lo que necesitás.

Páginas web renderizan de arriba para abajo entonces primero va el css y código java crítico. Luego va la página y al final java script que no es crítico así carga la página primero sin tener que esperarlo.

Hacer botón que diga ver cuentas, ver indicadores porque no se nota que es un botón.

Metodología -> agregar título y cómo lo ordena. Cuál es el mejor y peor. El problema es que no se distingue de la pantalla de empresa. Se debe implementar una tabla estilo reporte en lugar de botones.

Después de que haga login se debe quitar el botón de hacer login porque ya no sirve. Para mostrar que hizo login se puede mostrar el nombre de usuario loggueado en lugar del botón de login.

Importante poner cosas responsive en el menú. Ejemplo al achicar la pantalla al tamaño de un celular se puede poner los botones del menú en una barra desplegable para que se vea mejor.

Spark te permite personalizar tus propias páginas 404.

Cuentas -> hacer que se vea lindo -> se puede usar md como si fuera div dentro del td.

Home -> en lugar de bienvenido que te lleve a lo que consideres más importante como por ejemplo a la lista de empresas.

Problema aleatorio de error 500 -> lo solucionó quitando observable a indicador auxiliar y a metodología.

No manejar mensajes de error con cookies. Opciones:

Get … ?error=mal escrito -> es lo más prolijo, se usa en internet.

Otra opción Post a indicadores, en el redirect hacer con post, Post a nueva diciendo tenés un error.

Si cookie no se borra se puede pisar. También se puede poner que expire con número negativo.

Usuario puede usar link para acceder a las cosas de otro usuario. Cambiarlo porque debería mostrar sólo lo que el usuario puede ver.

Problema con el footer -> mirar fixed to bottom o darle a body una altura mínima para que no suba el footer.

Corrección documentación:

Base de datos por un lado y servidor que lo procese por el otro. Internet se conecta con el servidor y el servidor con la BD.

Rectángulo con lo que es del sistema y usuario afuera. Es importante en el diagrama de arquitectura separar en partes todos los objetos que tengas. Ej. Servidor se puede separar en MVC.

Servidor corriendo spark.

Protocolos usuados HTTP

(Existen otros protocolos:

HTTPS -> conexión cifrada con el servidor, más procesamiento desde el lado del cliente y del servidor

FTP -> para subir archivos

SCH, Telnet, etc.)

Abajo descripción más formal de lo que usás.

Principales componentes lógicos de alto nivel -> lo mismo que el diagrama pero más formal.

Resumen correcciones:

Meter manejo de errores, error no podés ver esta página porque no tenés permisos. Error crear algo con datos errones. Formatear error 404.

Para muchas empresas usar paginación. Por ejemplo mostrar 5 empresas por página.

Metodología mostrarlo más a modo reporte. Se le puede poner primeras empresas en verde, las del medio en naranja y las de abajo en rojo, o usar columna con 1,2,3,4, etc. Para distinguir cuál es mejor.

Hacer la página más responsive.