Evaluación Diagnostica

Plan Formativo	Nivel de Dificultad:
Full Stack Java Trainee	**Medio**
Tema:	Desarrollo de Proyecto Web Java EE
Objetivo del proyecto:	Se debe generar una aplicación web basándose en el en el patrón de diseño MVC y el desarrollo de aplicaciones web JEE.-
Ejecución: Individual	
Descripción del ejercicio	

Caso "BikeShop Sales-Force"

"BikeShop" es una empresa dedicada a la comercialización bicicletas, accesorios y repuestos. Es una tienda de nicho que busca satisfacer las necesidades de un exclusivo segmento de deportistas y entusiastas de alto rendimiento, para lo cual ofrece una gran variedad de productos con un excelente nivel de calidad.

La empresa se creó a inicio de los noventas como un emprendimiento de un grupo de amigos entusiastas de los deportes ciclísticos. La empresa ha tenido un crecimiento paulatino y hoy en día cuenta con 3 locales en Estados Unidos y una plataforma web que recibe pedidos de los clientes pero que no ha tenido mayores cambios durante los últimos años.

Dado que hoy en día las personas están optando por medios de transporte ecológicos, la empresa apuesta por un aumento de las ventas para los próximos 5 años razón por la cual se ha propuesto renovar sus sistemas de inventario, órdenes y catálogo web.
Para esto, el CTO (Chief Tecnological Officer) ha formado un equipo de proyectos de primera línea del cual usted forma parte como desarrollador full-stack Java. El equipo de proyectos también lo conforma un Jefe de Proyectos, un Diseñador UX/UI, un Diseñador Web, un Analista Funcional, un Desarrollador Mobile, y un Arquitecto de Software.

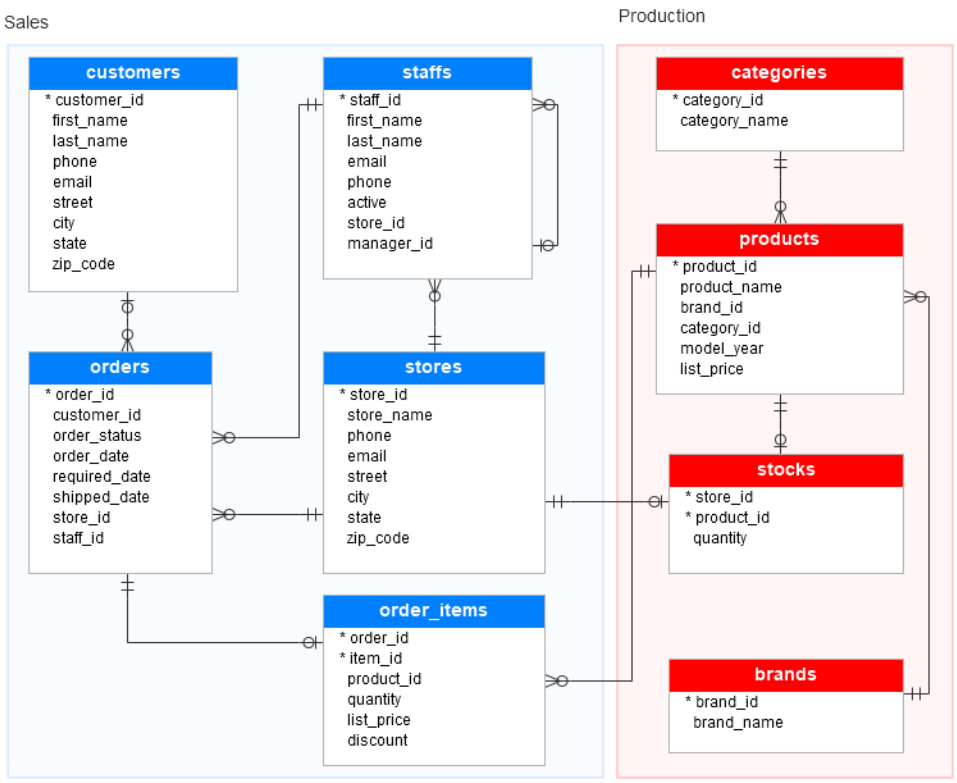
El proyecto busca mejorar la gestión de la venta, para lo cual se empezará a desarrollar un módulo que permite la visualización de las órdenes, en sus distintos estados, en cada tienda. Asimismo, se mejorará el sistema de incentivos para vendedores, realizando el cálculo de beneficios adicionales dependiendo del monto de venta que hayan tenido durante el mes.

A continuación, se listan los requisitos funcionales de alto nivel del sistema:

- El sistema debe permitir visualizar las órdenes de las distintas tiendas y su estado de ejecución.
- El sistema debe permitir el cálculo de beneficios para los vendedores
- A la fecha, ha transcurrido gran parte del proyecto y se tiene el siguiente avance:
- Ya se cuenta con un prototipo funcional del aplicativo
- Existe un modelo de datos diseñado
- Existe una base de datos con datos de prueba
- Existe una aplicación web que desarrolla algunas de las funcionalidades requeridas

Modelo de Datos

A continuación, se presenta el modelo de datos diseñado por el arquitecto en conjunto con el analista funcional.



Como se puede observar, el modelo tiene un área de trabajo con entidades relacionadas con el modelo de producción (catálogo e inventario) y otro grupo de entidades relacionadas con las ventas y órdenes.

La tabla *products* corresponde al maestro de productos de la compañía, en donde cada registro de producto tiene asociado una categoría de la tabla *categories* y una marca de la tabla *brands*. La tabla *stocks* contiene la cantidad del producto en existencia en cada una de las tiendas.

La tabla *orders* contiene la información de las órdenes que han sido cursadas, en donde el campo order_status señala el estado de la orden y los campos order_date, required_date y shipped_date almacenan las fechas de las órdenes en sus distintas etapas.

Por otra parte, las órdenes están asociadas a la tabla *order_items* que almacena la información de los productos, cantidades, precios y descuentos que se consignaron al momento de cursar la orden.

Cada orden, además, está asociada a un cliente (tabla customers) en donde se almacena la información general de cada cliente tal como nombre, dirección, teléfono, etc. Asimismo, cada orden está asociada a una tienda (tabla *stores*) y a un vendedor (tabla staff).
El maestro de tiendas (*stores*) contiene información de cada sucursal. Cada sucursal, a su vez, maneja un grupo de vendedores y managers los cuales se encuentran almacenados en la tabla *staffs*. El maestro de empleados (tabla *staff*) almacena una relación jerárquica entre los empleados a través del campo *manager*_id.

Requerimientos a Desarrollar

El jefe de proyectos, quien lleva un control meticuloso de las actividades del proyecto, le ha solicitado a Usted que realice las siguientes tareas:

- Realizar consultas a la base de datos
- Construir un algoritmo de cálculo de beneficios para los vendedores
- Construir una unidad de pruebas para verificar los algoritmos de cálculo de beneficios para los vendedores
- Crear un Reporte de Ordenes por Tienda
- Crear una API REST que disponibilice la información del Reporte de Ordenes por Tienda

A continuación, se especifica con mayor detalle cada uno de los requerimientos:

1. Realizar consultas a la base de datos

El Chief Sales Manager ha solicitado algunos reportes de información, razón por la cual el jefe del proyecto le ha encargado a usted que realice algunas consultas en la base de datos, para la extracción de cierta información necesaria para el negocio, por mientras se termina el desarrollo de la aplicación. **Para esto, cree un package en su proyecto Java con nombre "consultas", cree un archivo por cada una de ellas, identificando claramente de qué consulta se trata** (ejm: Consulta-A.sql, Consulta-B.sql, …etc).

a) La tienda Baldwin, que atiende la ciudad de New York, realizará una campaña de mailing. Para esto, se requiere un listado de los clientes del estado de New York (NY), ordenado alfabéticamente por apellido y después nombre. El reporte debe tener la siguiente forma:

customer_id	last_name ▲ 1	first_name ▲ 2	phone	email	street	city	state	zip_code
539	Acevedo	Jamika	*NULL*	jamika.acevedo@yahoo.com	8027 NW. Poplar St.	Ozone Park	NY	11417
1100	Acevedo	Penny	*NULL*	penny.acevedo@yahoo.com	318 Mulberry Drive	Ballston Spa	NY	12020
897	Acosta	Bettyann	(717) 746-6658	bettyann.acosta@gmail.com	7949 Chapel St.	Lancaster	NY	14086
616	Acosta	Shery	*NULL*	shery.acosta@yahoo.com	17 Canal Ave.	Saratoga Springs	NY	12866
836	Adams	Corinna	*NULL*	corinna.adams@msn.com	38 Trenton Court	Rosedale	NY	11422
854	Adkins	Phylis	(212) 325-9145	phylis.adkins@msn.com	7781 James Ave.	New York	NY	10002

b) La tienda Baldwin desea hacer una campaña de advertising con la empresa Yahoo. Para esto, se requiere un listado de los clientes que tienen correo Yahoo y que no registran número de teléfono, ordenado por apellido y nombre. El reporte debe tener la siguiente forma:

customer_id	last_name ▲ 1	first_name ▲ 2	phone	email	street	city	state	zip_code
539	Acevedo	Jamika	*NULL*	jamika.acevedo@yahoo.com	8027 NW. Poplar St.	Ozone Park	NY	11417
1100	Acevedo	Penny	*NULL*	penny.acevedo@yahoo.com	318 Mulberry Drive	Ballston Spa	NY	12020
616	Acosta	Shery	*NULL*	shery.acosta@yahoo.com	17 Canal Ave.	Saratoga Springs	NY	12866
1289	Anderson	Shanelle	*NULL*	shanelle.anderson@yahoo.com	646 Surrey Street	Bethpage	NY	11714
732	Austin	Lavona	*NULL*	lavona.austin@yahoo.com	926 South Euclid St.	Utica	NY	13501
1063	Austin	Tammy	*NULL*	tammy.austin@yahoo.com	182 Stillwater Ave.	Lake Jackson	TX	77566

c) Se requiere un reporte con la cantidad de órdenes totales de cada tienda, ordenado de mayor a menor cantidad. El reporte debe tener la siguiente forma:

store_name	count(*) 1
Baldwin Bikes	1093
Santa Cruz Bikes	348
Rowlett Bikes	174

d) Como se puede observar, la tienda Baldwin es la que tiene mayor cantidad de órdenes. Se requiere un reporte con los vendedores de la tienda Baldwin Bikes y la cantidad de órdenes asociada a cada uno, ordenado descendentemente por cantidad. El reporte debe tener la siguiente forma:

last_name	first_name	count(*) 1
Boyer	Marcelene	553
Daniel	Venita	540

e) Por último, se requiere un reporte que muestre el monto total vendido por cada vendedor de la tienda Baldwin Bikes. Tenga presente que para calcular el monto de una orden debe multiplicar la cantidad del ítem solicitado por el precio (no considere los descuentos). El reporte debe tener la siguiente forma:

last_name	first_name	MONTO
Boyer	Marcelene	2938888.73
Daniel	Venita	2887353.48

2. Construir un algoritmo para el cálculo de beneficios para vendedores

Un objetivo importante de este proyecto es crear un sistema que incentive a los vendedores a realizar un mejor trabajo. De esta forma, los vendedores tendrán mayor incentivo para atender más pedidos y realizar más ventas.

Para darle solución al problema anterior, se define que se construirá un módulo especial que permitirá a fin de mes calcular el beneficio adicional que obtiene cada vendedor y que se suman a su renta base.

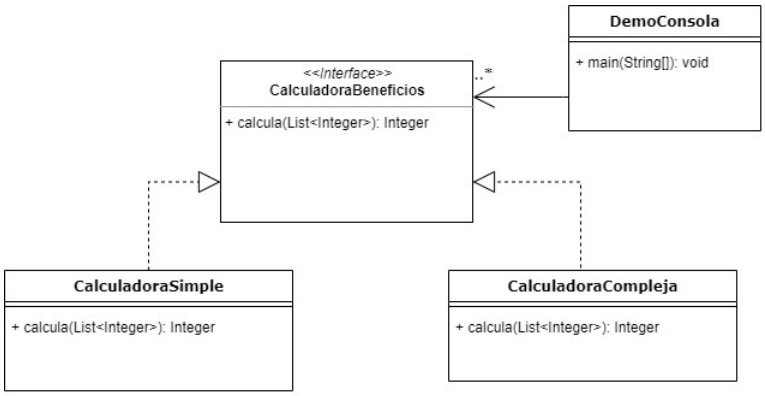
Se le solicita que desarrolle un algoritmo de cálculo de beneficio a vendedores con distintas lógicas de cálculo. Posteriormente, en la medida que se aprecien buenos resultados, se irán extendiendo e incorporando nuevas lógicas y reglas de negocio para el cálculo.
Construya dos algoritmos de cálculo de beneficios para vendedores, uno que realice un **cálculo simple** y otro que realice un **cálculo complejo**.

Las reglas del algoritmo de **cálculo simple** son las siguientes:

- El beneficio para los vendedores corresponde a un 3% del monto total de sus ventas durante el mes.
- Las reglas del algoritmo de **cálculo complejo** son las siguientes:
- El beneficio para los vendedores corresponde siempre a un 1% del monto total de sus ventas durante el mes, no obstante,
- Si se produce una venta por un monto superior a 1.000 US$, entonces se debe agregar al beneficio total, un monto correspondiente al 5% de esa venta.
- Si se produce una venta por un monto superior a 500 US$ pero inferior a 1.000 US$, entonces se debe agregar al beneficio total, un monto correspondiente al 2% de esa venta.
- Si la suma total de las ventas supera los 5.000 US$, debe agregar un monto adicional de beneficio de 100 US$

Ambos algoritmos deben recibir un listado de valores enteros correspondiente los montos de las ventas realizadas durante el mes (montos en dólares), y retornar un valor entero con el monto del beneficio calculado para ese vendedor (valor entero redondeado).

El arquitecto del proyecto le sugiere que realice un diseño de clases similar al del siguiente diagrama:



Para hacer una demostración de los algoritmos a la gerencia, se le solicita que cree una aplicación de consola que genere 10 valores aleatorios de monto de venta entre 100 y 2.000, y que posteriormente realice el cálculo de los beneficios para el vendedor utilizando las dos implementaciones creadas anteriormente (algoritmo simple y complejo).

A continuación, se presenta un ejemplo de la ejecución:

```
---------------------------------------------
Demostración Calculo Beneficios para Vendedores
---------------------------------------------

Tomando 10 valores de montos de venta aleatorios...
136 460 395 122 441 256 1690 699 525 654

Beneficio con Algoritmo Simple: 161 US$
Beneficio con Algoritmo Complejo: 276 US$
```

Considere hacer un diseño de clases polimórfico, mediante interfaces, en donde deberá crear una interfaz y dos clases concretas que implementen dicha interfaz. Una de ellas que implemente el algoritmo de cálculo simple y otra que implemente el algoritmo de cálculo complejo. Asimismo, una clase que sea la que ejecuta la aplicación de consola y genera el listado de stock con valores aleatorios para entregárselo a cada uno de los algoritmos.
Genere dentro de su proyecto en eclipse un package con nombre representativo que tenga las clases mencionadas.

3. Construir una unidad de pruebas para verificar el correcto cálculo de los beneficios para vendedores

Construya una clase de pruebas en Java que permita verificar el correcto funcionamiento del algoritmo de "cálculo complejo" de beneficios para vendedores, considerando al menos los siguientes tests:

- Tests que considere casos normales, con distintos montos y cantidad de ventas.
- Tests que considere condiciones de borde, por ejemplo, qué pasa cuando viene un stock cero, u otra condición de excepción.
-

4. Crear reporte web de Ordenes por Tienda

Se requiere crear una página web dinámica que despliegue el listado de órdenes que ha tenido una tienda en un rango de tiempo, tal como se detalla en la siguiente imagen mock-up.

Se pide:

a) En el combobox de tienda, desplegar los registros a partir de la información de la base de datos.
b) En el combobox de estados, poblarlos con la siguiente información: 1 = Pendiente; 2 = En Proceso; 3 = Rechazado; 4 = Completado
c) Implementar la acción de búsqueda a partir de los filtros seleccionados.
d) Desplegar el listado de órdenes, ordenado desde la fecha más reciente de forma descendente, tal como se muestra en el mock. En el caso del estado de la orden, los códigos de los estados son los siguientes: 1 = Pendiente; 2 = En Proceso; 3 = Rechazado; 4 = Completado

Para realizar el requerimiento, el arquitecto le señala lo siguiente:

- Utilizar jsp y taglibs jstl para el despliegue de la vista (u otra tecnología de vista)
- Utilizar bootstrap para los elementos
- Que los elementos se ajusten a distintos tamaños de pantalla

5. Crear una API REST que disponibilice el reporte de órdenes por tienda (opcional)

Disponibilice un servicio REST que permita obtener la misma información del reporte de órdenes por tienda creada anteriormente. Recuerde que el servicio puede recibir parámetros de tienda, estado y fechas.

Requerimientos de los participantes		
Conocimientos previos • HTML, CSS, JS • Java SE • MySql, DDL, DML • JEE • JSP • Servlets • JDBC Template • Rest API •	**Actitudes para el trabajo** • Cumplimiento de plazos • Diseño y Estructura • Trabajo en equipo	**Valores** • Tiempo de resolución. • Enfoque al requerimiento. • Estructura de Solución.
Objetivo General de Aprendizaje	**El participante al finalizar el proyecto será capaz de:** Desarrollar un sitio web dinámico bajo el patrón de diseño MVC usando tecnología JEE, conectándose a una base de datos relacional (MySQL, Oracle, PostgreSQL)	
Duración del proyecto	Cierre de Proyecto	

Productos para obtener durante la realización del proyecto

Un proyecto Java Web Dinámico. Debe incluir clases java, interfaces, archivos JSP, hojas de estilo, scripts y todos aquellos elementos que sean necesarios para su despliegue enun servidor de aplicaciones.

Un archivo de extensión SQL con el script de creación de la base de datos. En el scriptdebe incluir al menos tres registros en cada tabla.

Un archivo Readme.txt con todo lo que se necesite saber del proyecto: nombre de los integrantes, ruta de los repositorios en GitHub de cada alumno, un resumen del proyecto, y aquellos aspectos que el equipo considere relevante indicar.

Un repositorio en Github de tipo público con el contenido del proyecto.

Especificaciones de desempeño

Deberá realizar la actividad según requerimientos técnicos; el resultado deberá ser entregado de acuerdo con lo indicado en el punto anterior, acompañado por un archivo de texto plano Readme.txt, contextualizando el problema y planteamiento de la solución. La solución deberá ser gestionada a repositorio Github.

Sugerencias bibliográficas para la investigación

Qué es Java Enterprise (J2ee, JEE)

https://www.fundesem.es/bt/publicacion-java-ee-y-el-desarrollo-web-un-enfoque-de-aprendizaje

Etiquetas JSTL

http://www.jtech.ua.es/ayto/ayto2008/jsp/sesion07-apuntes.html

MVC

https://www.ecodeup.com/patrones-de-diseno-en-java-mvc-dao-y-dto/