AWAKELAB
#programmingbootcamp

[SIMULACRO 2] PRUEBA DE SALIDA FULL STACK JAVA

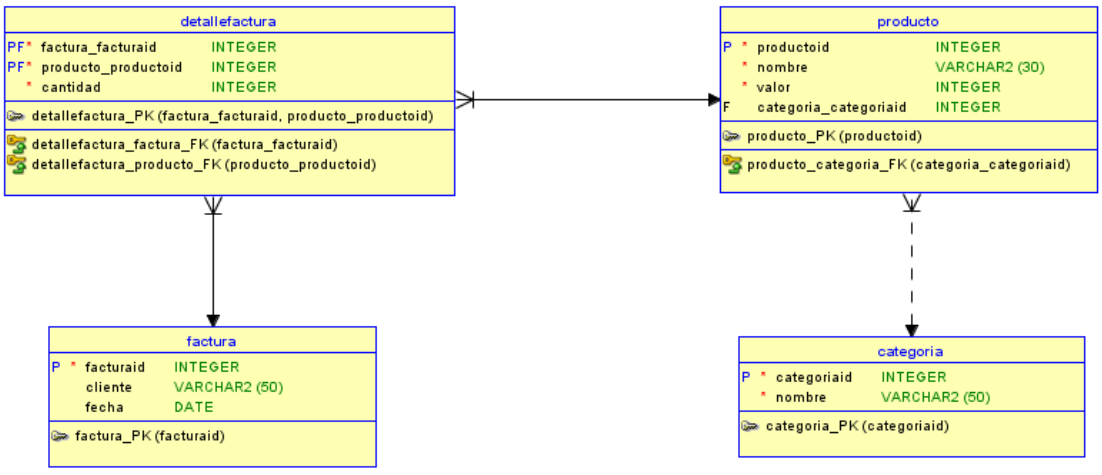
Plan Formativo	**Nivel de Dificultad:**
Full Stack Java	Medio
Nombre del proyecto: Simulacro 2 para la prueba de salida del curso Full Stack Java	**Tema:** Generar una solución de negocio basada en una plataforma web construida bajo arquitectura JEE
Objetivos del proyecto:	- Desarrollo de consultas a la base de datos - Crear algoritmos de acuerdo con requerimientos - Construir unidades de prueba - Crear una aplicación JEE - Implementar un servicio REST
Ejecución: Individual	

Descripción del ejercicio

1.- Problema

Un negocio de ventas mayoristas posee una base de datos que almacena cada una de las facturas que se emiten, la fecha en que esto se hace, y los productos que se asocian a cada una.

El modelo de datos utilizado es el siguiente:



Las tablas anteriores y sus campos son los siguientes:

- **factura:** permite registrar las facturas que se generan por cada venta. Considera un identificador, el nombre del cliente y la fecha en que se realiza.
- **categoria:** listado de categorías de productos; se compone de un identificador y el nombre.

- **producto:** en esta tabla se almacenan los productos a adquirir pertenecientes a las diferentes categorías. Tiene un identificador, el nombre del producto, el valor y el identificador de la categoría.
- **detallefactura:** cada registro de esta tabla representa un ítem de factura. En ella se puede encontrar un identificador de la factura, el identificador del producto y la cantidad comprada.

2.- Comandos iniciales de carga de datos

Para crear el modelo de datos en una base de datos Oracle 11g, ejecute el script que se adjunta al problema.

3.- Requerimientos a desarrollar

Desde la dirección de la clínica se solicita un sistema que permita tener control sobre las ventas realizadas; por lo tanto, le ha solicitado a usted que realice las siguientes tareas:

1. Realizar consultas a la base de datos.
2. Construir un algoritmo para el cálculo de monto de ventas.
3. Construir una unidad de pruebas para verificar el algoritmo de cálculo de monto de ventas.
4. Crear un sitio que despliegue el detalle de una factura.
5. Crear un servicio REST que despliegue el listado de ítems de una factura en formato Json.

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

3.1.- Realizar consultas a la base de datos

El gerente de la tienda le ha solicitado algunos reportes con información de las tablas antes indicadas, 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, mientras que termina el desarrollo de la aplicación. Para esto, cree un package en su proyecto Java con nombre “consultas”, y agregue un archivo por cada una de ellas, identificando claramente de qué consulta se trata (ejm: Consulta-A.sql, Consulta-B.sql, …etc).

a) Listado de los productos que no están asociados a una categoría. Se debe indicar el identificador del producto y el nombre de la categoría, y debe estar ordena descendentemente por el nombre de la categoría.
b) Listado de las facturas que no tienen asignado un cliente, o bien que no tienen una fecha establecida. Se debe considerar un campo de nombre CANTIDAD, que indique la cantidad

de productos asociados a la factura. El reporte se debe ordenar por el identificador de la factura de manera ascendente.

c) Listado de los productos que han sido vendidos al menos una vez pertenecientes a la categoría “Electrodomésticos”.
d) Listado de productos adquiridos, precio y cantidades respectivas asociadas a las facturas con identificador entre 1 y 5; de la factura se debe indicar el identificador y la fecha de emisión. El listado debe estar ordenado por el identificador de factura de forma ascendente, y por el nombre del producto de manera descendente.
e) Listado de las facturas existentes en sistema, el subtotal (cantidad de productos por el valor del ítem) respectivo, el impuesto asociado a la venta y el total (subtotal más impuesto). Se sabe que el impuesto a aplicar es del 19% del subtotal de la venta. Del reporte se debe indicar el identificador de la factura y los valores solicitados, y debe estar ordenado ascendentemente por el identificador de la factura.

3.2.- Construir un algoritmo para el cálculo de monto de ventas.

Como parte del sistema de información, se necesita tener registro de las ventas que se realizan, en específico del monto de venta de cada una. Cada venta de manera inicial tiene un comprador, una fecha y un monto inicial. A toda venta se le debe adicionar el impuesto fijo que asciende al 19%, y el monto inicial más el monto de impuesto da el monto total de la venta.

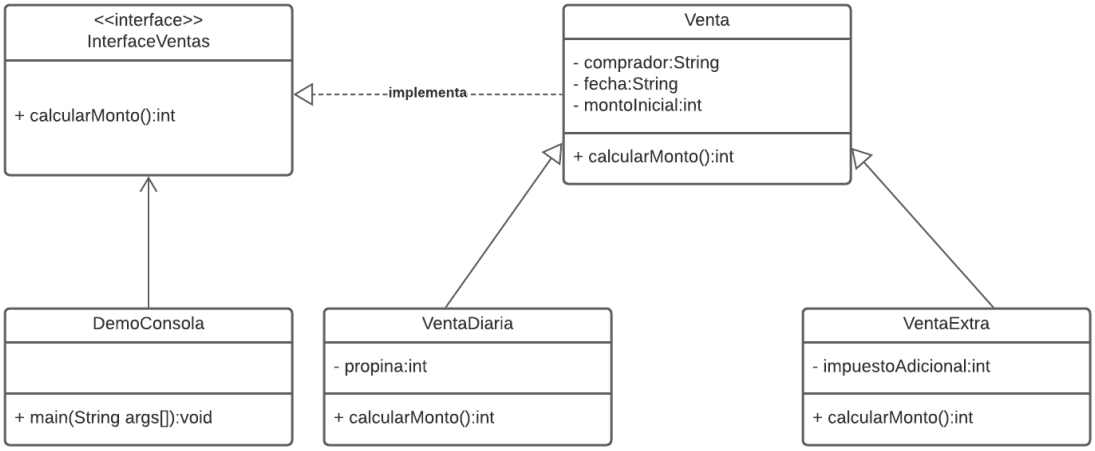
Además, existen dos tipos de ventas:

- **Venta diaria:** son las ventas que se realizan diariamente. Adicional al cálculo regular que se hace sobre el monto, se debe adicionar el monto de propina asignado al crear este tipo de venta. A la propina no se le aplica impuesto.
- **Venta extra:** son ventas especiales que se generan de manera irregular. Sobre estas ventas aplica un impuesto adicional, expresado en términos de porcentaje (número mayor o igual a cero y menor o igual a 100). Es importante considerar que este impuesto adicional se aplica sobre el impuesto regular.

Para dar solución a lo anterior, se decidió crear un algoritmo que realice el cálculo del monto asociado a cada venta, y que permita establecer la cantidad de ventas diarias en las que la propina es igual o superior al 10% del monto inicial de la venta, y además saber qué cantidad de ventas extras en las que el porcentaje de impuesto fijo (19%) es superior al impuesto adicional de ventas. Finalmente, se debe calcular la suma total de montos de ventas, considerando ambos tipos.

Esta es una idea que surgió recientemente pero que aún el jefe de proyectos no está del todo convencido de que dará el resultado esperado. Es por este motivo que el arquitecto ha decidido que lo mejor es construir una pieza de software extensible, con comportamiento polimórfico, para que a futuro se puedan incorporar nuevas lógicas de chequeo disminuyendo el impacto en el software.

Esta situación se puede representar a través del siguiente diagrama de clases:



Se le solicita que desarrolle un algoritmo de chequeo del monto de las ventas con comportamiento polimórfico. Se pide entonces que implemente los siguientes elementos:

- Una interface de nombre InterfaceVentas, en la que se defina un método de nombre calcularMonto().
- Una clase de nombre Venta, que implemente la interface InterfaceVenta.
- Las clases VentaDiaria y VentaExtra, las cuales deben heredar desde la clase Venta. En ellas se deben implementar los métodos calcularMonto() respectivos, usando las reglas indicadas en los párrafos anteriores.
- El programa DemoConsola, el cual debe contener un método main() en el cual se cree manualmente (no es necesario solicitar por pantalla los datos) cuatro ventas de tipo diario y cuatro ventas de tipo extra. Una vez creados estos objetos se debe obtener su monto final invocando al método calcularMonto() respectivo, y sumar dichos montos. Finalmente se debe indicar por medio de un mensaje por consola las cantidades de ventas que cumplen las condiciones antes indicadas.

Considere el siguiente ejemplo de ejecución del programa a modo de referencia:

```
-------------------------------------
Demostración algoritmo cálculo montos
-------------------------------------

Venta diaria 1: Inicial 3000, propina 100, total 3670

Venta diaria 2: Inicial 2500, propina 500, total 3474

Venta diaria 3: Inicial 10000, propina 1000, total 12900

Venta diaria 4: Inicial 7640, propina 400, total 13092
```

```
Venta extra 1: Inicial 5000, impuesto adicional 1190, total 7140

Venta extra 2: Inicial 15000, impuesto adicional 1785, total 19635

Venta extra 3: Inicial 3480, impuesto adicional 1242, total 5384

Venta extra 4: Inicial 20000, impuesto adicional 9520, total 33320

Monto total ventas diarias: 33137

Monto total ventas diarias: 65478

Monto total de ventas: 98615

Ventas diarias con propina igual o superior al 10% de monto inicial: 3

Ventas extras con porcentaje impuesto fijo (19%) mayor a impuesto
adicional de ventas: 1
```

Genere dentro de su proyecto en eclipse un package con nombre representativo que tenga las clases mencionadas.

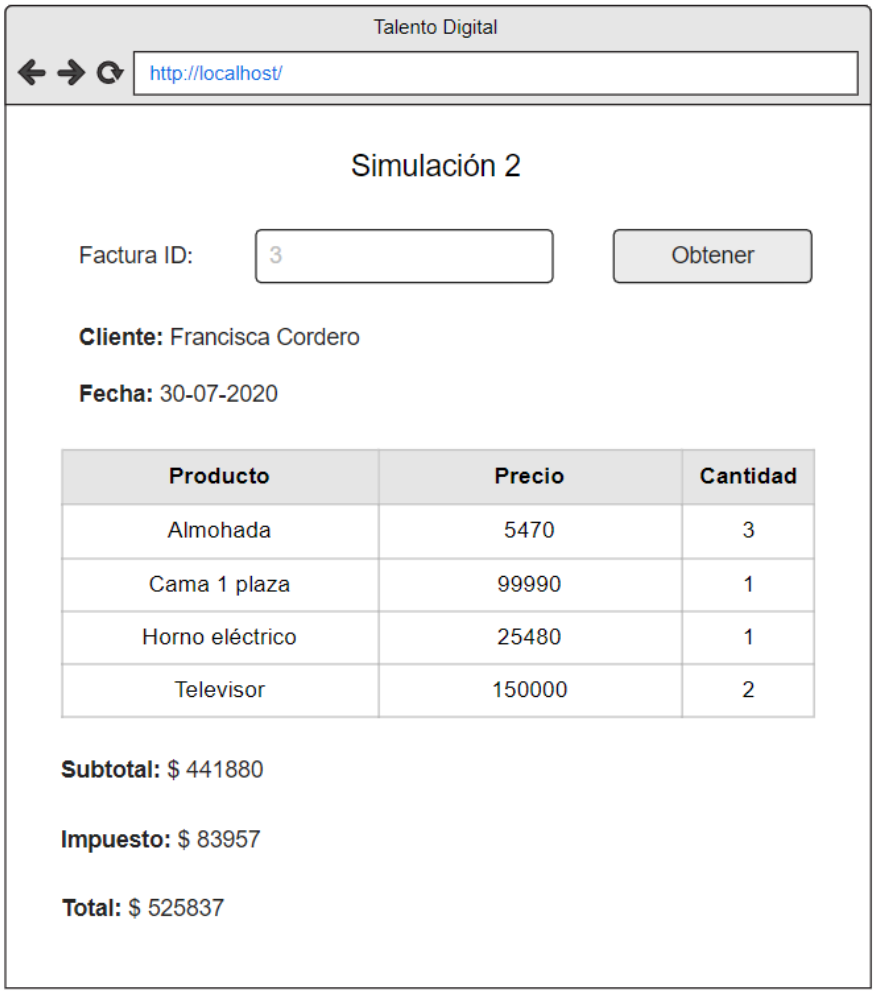
3.3.- Construir unidades de pruebas

Construya una clase de pruebas en Java que permita verificar el correcto funcionamiento del algoritmo de cálculo de monto de ventas, considerando al menos los siguientes tests:

- Tests que consideren ventas diarias (2 casos).
- Tests que consideren ventas extras (2 casos).

3.4.- Crear un sitio que despliegue el detalle de una factura

Para visualizar los datos de una factura, se requiere crear una página web dinámica que despliegue el reporte de facturas, tal como se detalla en la siguiente imagen mock-up.



Se pide:

a) Inicialmente, en la página debe aparecer el cuadro para insertar un ID de factura y un botón de procesamiento.
b) Al presionar el botón, se debe desplegar el detalle de la factura, tal como se muestra en la imagen.
c) Si se ingresa otro identificador de factura y se presiona el botón de procesamiento, se deben actualizar los datos.

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

3.5.- Crear un servicio REST que despliegue la información del reporte de horas agendadas

Disponibilice un servicio REST que permita obtener el detalle de una factura (nombre de producto, precio del producto y cantidad vendida). Recuerde que el servicio tiene que recibir como parámetro el identificador de la factura.

Requerimientos

- El sistema debe ser construido utilizando el framework Spring MVC, conectándose a una base de datos Oracle 11g express.
- En las tablas no existen campos autoincrementales.
- La revisión del problema se realizará en base al modelo antes planteado; no se permite agregar, modificar o quitar campos del modelo.

Contribuciones		
Requerimientos de los participantes		
Conocimientos previos • HTML • CSS • Javascript • Responsividad • Java Enterprise Edition • Spring Framework • Oracle 11g express edition • Servicios Rest	**Actitudes para el trabajo** • Proactividad • Aplicar casos anteriores en contextos similares • Uso del tiempo • Efectividad en la solución planteada	**Valores** Tiempo de resolución. Iniciativa
Objetivo General de Aprendizaje	Desarrollar una plataforma que permita mostrar los registros provenientes de una búsqueda, junto con un servicio REST de obtención de datos.	
Objetivos particulares	- Creación de sitios web responsivos - Conectar un sitio a una base de datos - Desarrollar una solución en base a un framework	
Duración del proyecto	Seis horas	
Tips o listado de Preguntas Guía		

Productos a obtener durante la realización del proyecto
- Un sitio web compuesto por una vista - Debe obtener datos en formato JSON desde un servicio Rest, y cargarlos en una base de datos
Especificaciones de desempeño
Deberá realizar la actividad según requerimientos técnicos en un plazo máximo de 6 horas
Cronograma de actividades
Sugerencias bibliográficas para la investigación