FASE 0

Tecnologías a utilizar

Bases de datos

- Oracle 18c XE: Es la opción que ofrece Oracle como una base de datos gratis para aquellos interesados en probar el producto o realizar operaciones pequeñas, además de que se puede conectar con otros lenguajes y tecnologías como Java, .NET, Python, Node.js, Go, PHP, c/c++ y se pueden habilitar servicios REST. Oracle es una base de datos relacional y utiliza como storage engine MyISAM. Esta base de datos se hará uso para los sistemas de ventas donde se hará la conexión hacia el servidor de backend Java.
- Mongo DB: Es una base de datos no relacional de documentos open source donde se almacenan los datos en documentos similares a los JSON. Esta base de datos se hará uso en los sistemas de fábrica los cuales se conectarán al servidor BE

Servidores BE

- Java: Es un lenguaje de programación orientado a objetos, el cual está centrado principalmente en la creación de servicios web que son consumidos por cualquier tipo de frontal, por medio de un API REST por lo que permite ser utilizado como BE para ofrecer servicio al FE, por lo que se utilizara como servidor BE del sistema de ventas
    - Spring Boot: Es un Spring Framework el cual facilita la creación de aplicaciones Java, el cual hace inyección de dependencias y ofrece diversos módulos, donde se encuentra el web el cual permite crear controladores de vistas MVC y aplicaciones REST.
- Express: Es una infraestructura de aplicaciones web Node.js mínima que proporciona un conjunto solido de características para aplicaciones web, el cual ofrece mecanismo para el manejo de peticiones HTTP hacia diferentes URL. Este se utilizará como servidor BE de la base de datos de mongoDB y establecerá interacciones con el servidor BE del sistema de ventas.
- Node.js: Es un entorno de ejecución para JavaScript orientado a eventos asíncronos el cual permite el manejo y administración de paquetes y creación de aplicaciones networks.

Angular: Angular es un framework que se utiliza para el desarrollo de aplicaciones web por medio del uso de Typescript el cual soporta el MVC y MVVM, este se utilizará para desarrollar la parte de FE conformando la interfaz web del sistema de fábricas.

React: React es un framework que se utiliza para el desarrollo de aplicaciones web utilizando una combinación de HTML y Javascript. Este se utilizará para desarrollar la parte de FE conformando la interfaz web del sistema de ventas.

GitHub: Es un controlador de versiones el cual permite el desarrollo colaborativo, siendo una plataforma que permite organizar las tareas y el código que se está elaborando donde también se puede recuperar alguna versión en específico y se pueden crear bifurcaciones que permiten trabajar de forma más segura en ambientes aislados. Este se utilizará como medio principal para almacenar el código de la aplicación, los servidores BE y los exports de las bases de datos de ambos sistemas y dejará registro de las contribuciones de cada desarrollador por medio de los commits realizados

Azure DevOps: DevOps proviene de los términos Development y Operation, el cual hace una unión de los procesos, personas y tecnologías de forma constante en la cual Microsoft ofrece en la nube esta plataforma en la cual se puede poner en práctica SCRUM para el mejor manejo de tareas y registro y solución de bugs por medio del testing que ofrece la misma plataforma, por lo que esta será utilizada para la distribución de trabajo, control de flujo y aceptación de los módulos

IDE's: Visual Studio Code, SQL Developer, Spring Tool Suite 4, Mongodb Compass.

Objetos a implementar:

Sistema de ventas:

- Procedimiento Almacenado: El procedimiento se encargará de hacer el insert en la tabla de inventario donde dependiendo del tipo de dispositivo se realizara el insert de la información en la tabla correspondiente por lo que el procedimiento tomara todos los parámetros in de la tabla de inventario y timara parámetros extra ordenados según los 3 tipos de tablas en los cuales se almacenara la información para el insert, el número de parámetros extras no debe ser menor al número de atributos mayor de las tres tablas.

- Función: Obtener el monto total de crédito a pagar por cada cliente dentro del lapso de un mes, esta función tomara como parámetros el id del cliente, la fecha en que se realizó la compra y el monto acreditado y posteriormente con la fecha se obtendrá el mes y el año y se devolverá como string el monto total.
- Trigger: Al momento de ingresar un dato en la tabla de inventario se hará un before insert ingresando el valor del costo y este dentro del trigger se aplicarán las operaciones necesarias para aplicar el precio de venta.
- Vista #1: Join de las tablas Inventario, tipo_dispositivo y marca para el fácil acceso a la información del elemento la cual se utilizará como búsqueda inicial para luego operar con las tablas de teléfonos, televisiones o videojuegos.
- Vista #2: En esta vista se contendrá la información del cliente únicamente mostrando el id del cliente, nombre, nit, tipo de suscripción, la patente de comercio y la fecha de vencimiento de la suscripción para filtrar aquellos datos que no se van a utilizar a la hora de realizar consultas para realizar transacciones.
- Vista #3: Vista que contendrá el listado de todos los dispositivos vendidos en los últimos dos años donde se tendrá el número de serie del dispositivo, el código de modelo, la fecha de adquisición y el propietario actual.

SISTEMA DE FÁBRICAS (FASE 1):

Propuesta de colecciones y JSON:

- **Colección:** catalogo_dispositivos. En esta colección se guardarán los dispositivos electrónicos disponibles de la fábrica. **JSONS:** los jsons a guardar tendrán atributos para tipo, marca, modelo, categoría y los específicos de tipos de dispositivos para televisiones, smartwatch y videojuegos.
- **Colección:** ventas. En esta colección se guardarán los datos de las ventas realizadas en cada tienda de dispositivos.
- **Colección:** last_seen. En esta colección se guardan documentos vistos de las ventas en la página de reportes. **JSONS**: tendrán atributos de la fecha de consulta de la pagina de reporte y referencia a los reportes las ventas que se vieron en esa fecha.
- **Colección:** dispositivos_individuales. En esta colección se llevan los documentos correspondientes a los dispositivos individuales pedidos por las ventas. **JSONS:** se

tendrán referencia a la tienda que realizó la venta, el dispositivo (con sus características) que se vendió y el precio, con la fecha de la venta.

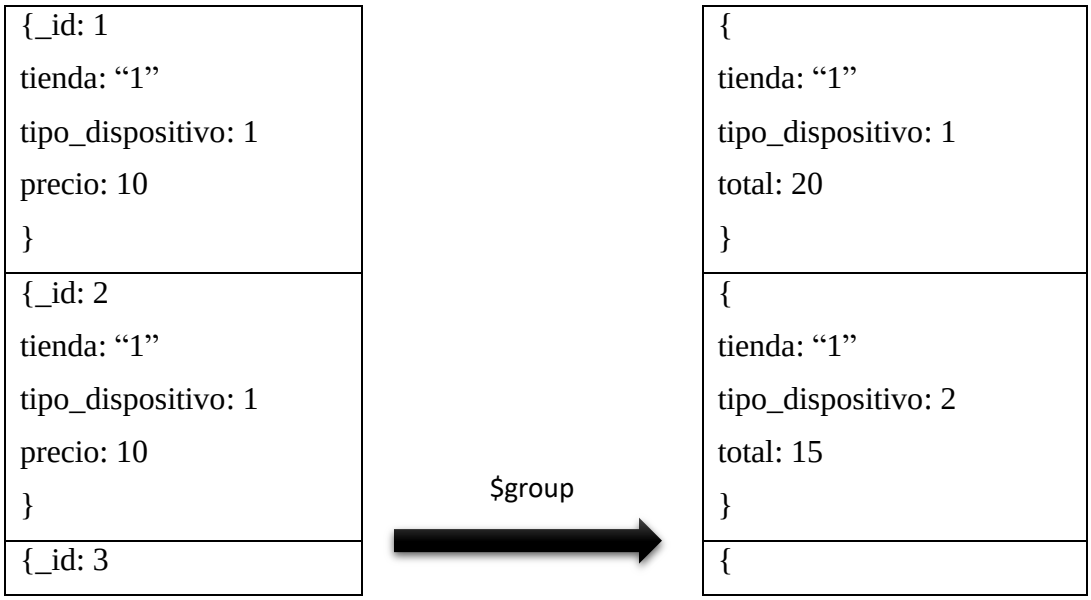
- **Colección:** clientes. Es una colección que funcionará como catálogo de los clientes que podrán comprar. **JSONS:** Los documentos tendrá atributos de id, usuario y contraseña para cada cliente.
- **Colección:** pedidos. Es una colección donde se guardarán los dispositivos que han sido pedidos por las ventas. **JSONS:** cada json corresponderá a un dispositivo pedido, y contendrá los atributos con las especificaciones del producto y la tienda que lo encargó, y se podrá especificar el tiempo en que se encargarán con otro atributo.

Propuesta de Aggregations:

- Para la parte de reportes, con esta aggregation se busca procesar documentos de la colección de ventas, para agruparlos por tiendas de electrónicos y por tipos de dispositivos y sumar el total vendido en cada tienda por cada tipo de electrónico vendido.

  Explicación gráfica:

  db.ventas.aggregate({$group: {tienda: "$tienda", tipo_dispositivo: "$tipo_dispositivo", total: {sum: "$precio" }}})



tienda: “1” tipo_dispositivo: 2 precio: 15 }	tienda: “2” tipo_dispositivo: 1 total: 10 }
{_id: 4 tienda: “2” tipo_dispositivo: 1 precio: 10 }	

- Pensando en el registro de dispositivos individuales en el sistema, esta aggregation se utilizará para procesar los dispositivos que aún no han sido enviados a las ventas de dispositivos. Es decir, que están pendientes de envío y por tanto, siguen en inventario de fábrica.
  Explicación gráfica:
  db.dispositivos_individuales.aggregate({$match: {entregado: 0}])

{_id: 1 serie: “123” idInventario: 1 entregado: 0 }	
{_id: 2 serie: “124”	{_id: 1 serie: “123”

idInventario: 2 entregado: 1 }
{_id: 3 serie: “125” idInventario: 2 entregado: 0 }
{_id: 4 serie: “126” idInventario: 1 entregado: 1 }

$match

idInventario: 1 entregado: 0 }
{_id: 3 serie: “125” idInventario: 2 entregado: 0 }