

Proyecto Individual

Memoria del Plan del Proyecto de Modelado Software

Autor: Luis Jaramillo Pulido

ÍNDICE

1. Introducción
2. Objetivos
3. Software utilizado en el desarrollo de la app.
4. Diseño: Capas de Presentación y Negocio.
5. Requisitos
6. Patrones
7. Vistas

PRESENTACIÓN DEL PROYECTO

1.INTRODUCCIÓN

La idea principal es desarrollar una herramienta que permita la gestión y administración de clientes, productos y ventas, entre sus principales funcionalidades están las funciones de alta, baja y modificación de los clientes, productos, y la creación de un pedido mediante crear venta.

2.OBJETIVOS:

El objetivo de la aplicación es facilitar y simplificar la administración de clientes, productos y ventas a través de una interfaz gráfica amigable, que permita el acceso a las funciones de gestión de los clientes y productos y ventas. Adicionalmente, otros usuarios podrán realizar acciones de gestión como consultas en clientes y productos, y también se podrá realizar una query para mostrar los productos de una venta mayores a un precio dado.

3.SOFTWARE UTILIZADO EN EL DESARROLLO DE LA APP:

Los servicios y aplicaciones utilizadas para el desarrollo de la app son los siguientes:

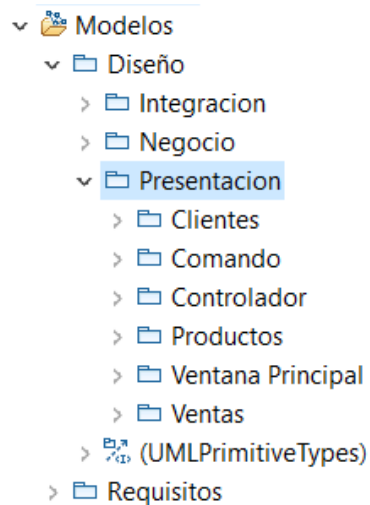
- IBM RATIONAL SOFTWARE ARCHITECT FOR WEBSPHERE SOFTWARE v.9.0.
- NETBEANS
- SQLITE
- WORD
- Eclipse

4.DISEÑO: CAPAS DE PRESENTACIÓN, NEGOCIO E INTEGRACIÓN

En el siguiente apartado se mostrará los diagramas de secuencia y los diagramas de clases implementados en el proyecto en las capas de Presentación, Negocio e Integración.

4.1 DIAGRAMAS DE PRESENTACIÓN

En el modelo UML en el apartado paquetes diseño, está ubicado el diagrama de presentación.



4.1.1 Diagramas de Secuencia de Presentación de Cliente y Diagrama de clases de Presentación de Cliente.

- ▼ Presentacion
 - ▼ Clientes
 - > Asociaciones
 - > Sucesos
 - ClientePresentacion
 - > ClienteGUI
 - > PanelCliente
 - > AltaCliente
 - > BajaCliente
 - > ConsultarCliente
 - > ModificarCliente
 - > Comando
 - > Controlador
 - > Productos
 - > Ventana Principal
 - > Ventas

4.1.2 Diagramas de Secuencia de Presentación de Producto y Diagrama de clases de Presentación de Producto.

- ▼ Modelos
 - ▼ Diseño
 - > Integracion
 - > Negocio
 - ▼ Presentacion
 - > Clientes
 - > Comando
 - > Controlador
 - ▼ Productos
 - > Asociaciones
 - > Sucesos
 - ProductoPresentacion
 - > PanelProducto
 - > ProductoGUI
 - > AltaProducto
 - > BajaProducto
 - > CargarProducto
 - > ModificarProducto
 - > Ventana Principal
 - > Ventas

4.1.3 Diagramas de Secuencia de Presentación de Venta y Diagrama de clases de Presentación de Venta.

- ▼ Modelos
 - ▼ Diseño
 - > Integracion
 - > Negocio
 - ▼ Presentacion
 - > Clientes
 - > Comando
 - > Controlador
 - > Productos
 - > Ventana Principal
 - ▼ Ventas
 - > Asociaciones
 - > Sucesos
 - VentasPresentacion
 - HashMapLineaVentas
 - > PanelVentas
 - > VentasGUI
 - > AgregarProductoVenta
 - > CrearVenta
 - > QuitarProductoVenta

4.1.4 Diagramas de Secuencia de Presentación del controlador y Dispatcher y Diagrama de clases de Presentación de Controlador y Dispatcher.

- ▼ 📁 Modelos
 - ▼ 📁 Diseño
 - > 📁 Integracion
 - > 📁 Negocio
 - ▼ 📁 Presentacion
 - > 📁 Clientes
 - > 📁 Comando
 - ▼ 📁 Controlador
 - > 📁 Asociaciones
 - > 📁 Sucesos
 - ▼ 📁 Dispatcher
 - 📄 Dispatcher
 - > 📄 DispatcherImp
 - > 📄 «Singleton» Dispatcher
 - > 📄 Dispatcher
 - 📄 Controlador
 - > 📄 ActionListenerImp
 - > 📄 «Singleton» ApplicationController
 - > 📄 ApplicationControllerImp
 - > 📄 RespuestaComando
 - > 📄 ApplicationController

4.1.5 Diagramas de Secuencia de Presentación de Comando (Cliente,Producto y Venta) y Diagrama de clases de Presentación de Comando(Cliente,Producto y Venta)

- ▼ 📁 Comando
 - ▼ 📁 Cliente
 - > 📁 Sucesos
 - 📄 ComandoClientImp
 - > 📄 AltaCliente
 - > 📄 BajaCliente
 - > 📄 ConsultarCliente
 - > 📄 ListarClientes
 - > 📄 ModificarCliente
 - > 📄 AltaCliente
 - > 📄 BajaCliente
 - > 📄 ConsultarCliente
 - > 📄 ListarCliente
 - > 📄 ModificarCliente
 - ▼ 📁 Comando
 - 📄 Comando
 - > 📄 IDEventos
 - > 📄 Comando
 - > 📁 FactoriaComando
 - ▼ 📁 Productos
 - > 📁 Sucesos
 - 📄 ComandoProductoImp
 - > 📄 BajaProducto
 - > 📄 ConsultarProducto
 - > 📄 ListarProductos
 - > 📄 ModificarProducto
 - > 📄 NuevoProducto
 - > 📄 ComandoProducto
 - ▼ 📁 Ventas
 - 📄 ComandoVentasImp

4.2 DIAGRAMAS DE NEGOCIO

En el modelo UML en el apartado paquetes diseño, está ubicado la capa de negocio

- ▼ Modelos
 - ▼ Diseño
 - > Integracion
 - ▼ Negocio
 - > Clientes
 - > FactoriaSA
 - > Productos
 - > Ventas
 - > Presentacion
 - > (UMLPrimitiveTypes)
 - > Requisitos

4.2.1 Diagramas de Secuencia de Negocio de Cliente y Diagrama de clases de Negocio de Cliente.

- ▼ Modelos
 - ▼ Diseño
 - > Integracion
 - ▼ Negocio
 - ▼ Clientes
 - > Sucesos
 - ClienteNegocio
 - > SAClienteImp
 - > TransferCliente
 - > SACliente
 - > altaClienteNegocio
 - > bajaClienteNegocio
 - > consultarClienteNegocio
 - > listarClienteNegocio
 - > modificarClienteNegocio

4.2.2 Diagramas de Secuencia de Negocio de Producto y Diagrama de clases de Negocio de Producto.

- ▼ Modelos
 - ▼ Diseño
 - > Integracion
 - ▼ Negocio
 - > Clientes
 - > FactoriaSA
 - ▼ Productos
 - > Sucesos
 - ProductoNegocio
 - > SAProductoImp
 - > TransferProducto
 - > SAProducto
 - > altaProductoNegocio
 - > bajaProductoNegocio
 - > consultarProductoNegocio
 - > listarProductoNegocio
 - > modificarProductoNegocio

4.2.3 Diagramas de Secuencia de Negocio de Venta y Diagrama de clases de Negocio de Venta

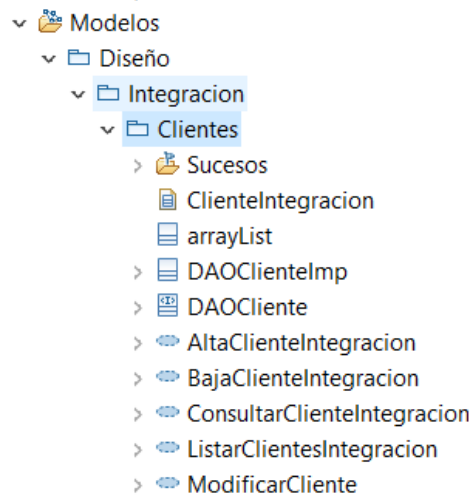
- ▼ Modelos
 - ▼ Diseño
 - > Integracion
 - ▼ Negocio
 - > Clientes
 - > FactoriaSA
 - > Productos
 - ▼ Ventas
 - > Asociaciones
 - > Sucesos
 - VentasNegocio
 - > collection<LineaVenta>
 - > DAOCliente
 - > FactoriaQuery
 - > Iterator<LineaVenta>
 - > LineaVenta
 - > Query
 - > SAVentasImp
 - > TransferTOA
 - > TransferVenta
 - > SAVenta
 - > CrearVentaNegocio
 - > EnsambladorObjetosTransferencia
 - > listarVentasNegocio
 - > mostrarProductosEnVentaMayoresPrecio

4.3 DIAGRAMAS DE INTEGRACIÓN

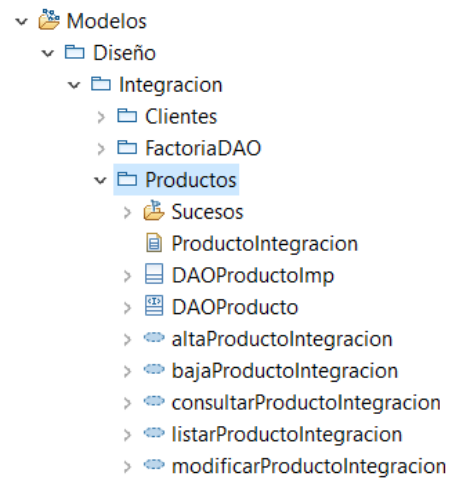
En el modelo UML en el apartado paquetes diseño, está ubicado la capa de integración.

- > Diagramas
- ▼ Modelos
 - ▼ Diseño
 - ▼ Integracion
 - > Clientes
 - > FactoriaDAO
 - > Productos
 - > Query
 - > Transaccion
 - > Ventas
 - > Negocio
 - > Presentacion
 - > (UMLPrimitiveTypes)
 - > Requisitos

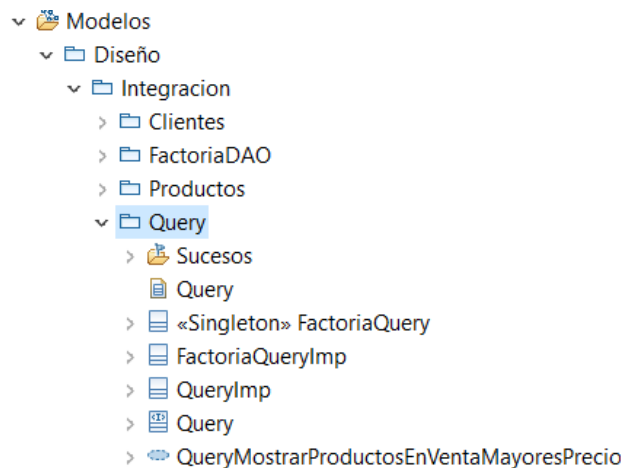
4.3.1 Diagramas de Secuencia de Integración Clientes y Diagrama de clases de Integración de Clientes



4.3.2 Diagramas de Secuencia de Integración Productos y Diagrama de clases de Integración de Productos



4.3.3 Diagramas de Secuencia de Integración Query



4.3.4 Diagramas de Secuencia de Integración Ventas y diagrama de clases de Integración Ventas.

- ▼ 📁 Modelos
 - ▼ 📁 Diseño
 - ▼ 📁 Integracion
 - > 📁 Clientes
 - > 📁 FactoriaDAO
 - > 📁 Productos
 - > 📁 Query
 - > 📁 Transaccion
 - ▼ 📁 Ventas
 - > 📁 Sucesos
 - 📄 VentasIntegracion
 - > 📄 DAOVentasImp
 - > 📄 DAOVentas
 - > 📄 consultarVentaIntegracion
 - > 📄 crearVentaIntegracion
 - > 📄 ListarVentasIntegración

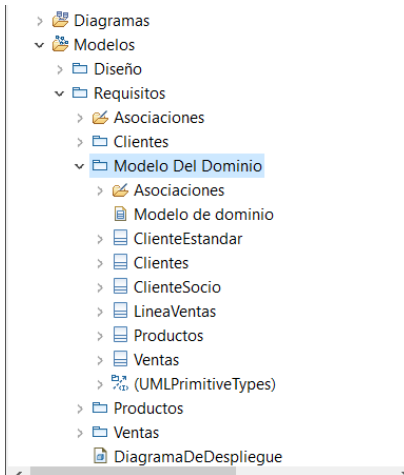
4.3.5 Diagramas de Secuencia TManager y Transaccion y diagramas de clases de Integracion de de TManager y transaccion.

- ▼ 📁 TManager
 - > 📄 Asociaciones
 - > 📄 Sucesos
 - 📄 TManager
 - > 📄 «Singleton» TManager
 - > 📄 TManagerImp
 - > 📄 TManager
 - 📄 Transaccion
 - > 📄 TManagerImp
 - > 📄 TransaccionImp
 - > 📄 Transaccion
- ▼ 📄 TManager
 - 📄 concurrentHashMap
 - 📄 factoriaTransaccion
 - 📄 Property
 - 📄 tManagerImp
 - > 📄 TManagerEliminaTransaccion
 - > 📄 TManagerGetTransaccion
 - > 📄 TManagerNuevaTransaccion
- ▼ 📄 Transaccion
 - 📄 connection
 - 📄 driverManager
 - 📄 factoriaTransaccion
 - 📄 factoriaTransaccionImp
 - 📄 preparedStatement
 - 📄 Property
 - 📄 tManagerImp
 - 📄 transaccionImp
 - > 📄 TransaccionCommit
 - > 📄 TransaccionRollback
 - > 📄 TransaccionStart
















5. REQUISITOS:

5.1 MODELO DE DOMINIO

















En el modelo UML en el apartado paquetes requisitos.



5.2 Diagramas de Actividad y Casos de Uso de Cliente, Producto y Venta

- >  Diagramas
- ▼  Modelos
 - >  Diseño
 - ▼  Requisitos
 - >  Asociaciones
 - ▼  Clientes
 - >  Diagrama de Casos de Uso
 - >  Diagramas de Actividad
 - >  Modelo Del Dominio
 - ▼  Productos
 - >  Diagrama de Casos de Uso
 - >  Diagramas de Actividad
 - ▼  Ventas
 - >  Diagrama de Casos de Uso
 - >  Diagramas de Actividad

5.3 Diagrama de Despliegue de Cliente, Producto y Venta

- >  Diagramas
- ▼  Modelos
 - >  Diseño
 - ▼  Requisitos
 - >  Asociaciones
 - >  Clientes
 - >  Modelo Del Dominio
 - >  Productos
 - >  Ventas
 -  DiagramaDeDespliegue
 -  BB.DD
 - >  jdk 8u 121
 - >  musicalia.jar
 -  sqlite-jdbc
 - >  Node1
 - >  Sqlite 3.16.2

6. PATRONES

Se ha usado una arquitectura multicapa, dentro de la arquitectura multicapa los principales patrones aplicados son:

Patrones Generales usados tanto en presentación como en negocio

Patrón Factoría Abstracta: Utilizada para la generación de los comandos, los servicios de aplicación Singleton y DAO.

PATRON DE DISEÑO SINGLETON Garantiza que una clase sólo tenga una instancia y proporciona un punto de acceso global a ella. Es la encargada de que una clase solo pueda instanciar un objeto, su constructor es privado, y se instancia a través de un método publico estático.

CAPA DE PRESENTACION

PATRON COMANDO: Patrón que indica las acciones posibles en el sistema.

DISPATCHER Es el responsable del control de la vista y la navegación, controlando la elección de la siguiente vista a mostrar y proporcionando el mecanismo para dirigir el control a este recurso.

APPLICATION CONTROLLER Centraliza las peticiones de comandos y vistas.

PATRON SERVICE TO WORKER Es una mezcla de patrones, tiene el controlador de la aplicación, tiene comandos para acceder a los servicios de la aplicación, y tiene un despachador para acceder a la vista

CAPA DE NEGOCIO

SERVICIO DE APLICACIÓN: Para encapsular la lógica de negocio, es un servicio que permite centralizar y agregar comportamiento para proporcionar una capa de servicio uniforme, es la encargada de eliminar una transacción, la capa de negocio solo se preocupa por lógica de negocio y utiliza el DAO para interactuar con la fuente de datos.

TRANSFER OBJECT: se usa para encapsular los datos. Se usa para enviar y recuperar el Transfer Object. Construye un nuevo Transfer Object basado en los requerimientos de la aplicación cuando el cliente solicita un Transfer Object.

TOA: Ensamblador de objetos de transferencia, se encarga de ensamblar varios transfer, y su uso es para obtener datos que de un solo transfer no se puede obtener. En mi proyecto, uso este patrón para obtener el nombre del cliente de una venta, obtengo el transferVenta a partir de un idVenta, y a partir del transfer Venta obtengo el id del cliente, y con ello accedo al dao para obtener los datos del cliente que en este caso es su nombre.

CAPA DE INTEGRACIÓN

DAO el patrón DAO propone separar por completo la lógica de negocio de la lógica para acceder a los datos, de esta forma, el DAO proporcionará los métodos necesarios para insertar, actualizar, borrar y consultar los datos.

7.VISTAS Y FUNCIONAMIENTO

7.1 CLIENTE

Proyecto Individual

Clientes Productos Ventas

ID

DNI

Nombre

Apellidos

Direccion

Socio

Activo

☐ es Socio

Lista de Clientes

```
1: Luis Vizan DNI: 1234 Socio: true activo: false
2: Carlota Ramirez DNI: 55555 Socio: true activo: true
3: Luis Alberto Jaramillo DNI: 9012 Socio: false activo: true
27: Miguel Santos DNI: 1111 Socio: false activo: true
28: karina jaramillo DNI: 89741 Socio: false activo: false
34: Pedro Martinez DNI: 4565 Socio: true activo: true
35: Celia Loayza DNI: 50763622 Socio: true activo: true
```

7.1.1 Alta cliente

Se introduce los datos del cliente excepto el id que se quiere dar de alta, se marca checkbox si el cliente va ser socio y se pulsa el botón “Alta cliente”

Proyecto Individual

Cientes

Productos

Ventas

ID

DNI

89987

Nombre

Rosa

Apellidos

Flores

Direccion

calle rosa

Socio

No

Activo

Si

Alta cliente

Modificar Cliente

Baja cliente

Cargar Cliente

☐ es Socio

Limpiar

Lista de Clientes

1: Luis Vizan DNI: 1234 Socio: true activo: false

2: Carlota Ramirez DNI: 55555 Socio: true activo: true

3: Luis Alberto Jaramillo DNI: 9012 Socio: false activo: true

27: Miguel Santos DNI: 1111 Socio: false activo: true

28: karina jaramillo DNI: 89741 Socio: false activo: false

34: Pedro Martinez DNI: 4565 Socio: true activo: true

35: Celia Loayza DNI: 50763622 Socio: true activo: true

7.1.1 Baja cliente

Se introduce el id de un cliente que exista y este activo, y se pulsa en “Baja cliente”

The screenshot shows the 'Proyecto Individual' application window with the 'Clientes' tab selected. The form contains the following fields and buttons:

- Form fields: ID (34), DNI, Nombre, Apellidos, Direccion, Socio, and Activo.
- Buttons: Alta cliente, Modificar Cliente, Baja cliente (highlighted with a red box), Cargar Cliente, and Limpiar.
- Checkbox: ☐ es Socio.
- Lista de Clientes: A list of 5 clients with their DNI, Socio status, and Activo status.

Lista de Clientes:

- 1: Luis Vizan DNI: 1234 Socio: true activo: false
- 2: Carlosa Ramirez DNI: 55555 Socio: true activo: true
- 3: Luis Alberto Jaramillo DNI: 9012 Socio: false activo: true
- 27: Miguel Santos DNI: 1111 Socio: false activo: true
- 28: karina jaramillo DNI: 89741 Socio: false activo: false
- 34: Pedro Martinez DNI: 4565 Socio: true activo: true
- 35: Celia Loayza DNI: 50763622 Socio: true activo: false

7.1.3 Modificar Cliente

Se introduce el id de un cliente que exista y se introduce los datos a modificar y luego se pulsa “modificar cliente”

The screenshot shows the 'Proyecto Individual' application window with the 'Clientes' tab selected. The form contains the following fields and buttons:

- Form fields: ID (34), DNI (4565), Nombre (Pedro), Apellidos (Martinez Sanchez), Direccion (calle vista alegre), Socio (Si), and Activo (Si).
- Buttons: Alta cliente, Modificar Cliente (highlighted with a red box), Baja cliente, Cargar Cliente, and Limpiar.
- Checkbox: ☒ es Socio.
- Lista de Clientes: A list of 5 clients with their DNI, Socio status, and Activo status.

Lista de Clientes:

- 1: Luis Vizan DNI: 1234 Socio: true activo: false
- 2: Carlosa Ramirez DNI: 55555 Socio: true activo: true
- 3: Luis Alberto Jaramillo DNI: 9012 Socio: false activo: true
- 27: Miguel Santos DNI: 1111 Socio: false activo: true
- 28: karina jaramillo DNI: 89741 Socio: false activo: false
- 34: Pedro Martinez DNI: 4565 Socio: true activo: true
- 35: Celia Loayza DNI: 50763622 Socio: true activo: false

7.1.4 Cargar Cliente

Se introduce el id de un cliente que exista y luego se pulsa en “cargar cliente”.

Proyecto Individual

Cientes Productos Ventas

ID: 3

DNI:

Nombre:

Apellidos:

Dirección:

Socio: ☐ es Socio

Activo: ☐

Alta cliente

Modificar Cliente

Baja cliente

Cargar Cliente

Limpiar

Lista de Clientes

1: Luis Vizán DNI: 1234 Socio: true activo: false
 2: Carlota Ramirez DNI: 55555 Socio: true activo: true
 3: Luis Alberto Jaramillo DNI: 9012 Socio: false activo: true
 27: Miguel Santos DNI: 1111 Socio: false activo: true
 28: Karina Jaramillo DNI: 89741 Socio: false activo: false
 34: Pedro Martínez DNI: 4565 Socio: true activo: true
 35: Celia Loayza DNI: 50763622 Socio: true activo: false

7.2 PRODUCTO

Proyecto Individual

Cientes Productos Ventas

ID: 21

Nombre: ukele

Precio: 121.0

Categoría: cuerda

Stock: 45

Alta Producto

Modificar Producto

Baja Producto

Cargar Producto

Lista de Productos

18: guitarra clasica, cuerda: 250.0 € stock: 42
 19: piano, percusion: 1000.0 € stock: 48
 20: trompeta, viento: 110.0 € stock: 200
 21: ukele, cuerda: 121.0 € stock: 45
 22: violin, cuerda: 310.0 € stock: 20

En producto el funcionamiento es similar en los casos de alta, baja, modificar y cargar de la vista del Cliente.

7.3 VENTA

Proyecto Individual

Cientes Productos Ventas

IDProducto:

Nombre:

Precio:

IDCliente:

IDVenta:

Cargar Producto Agregar Producto

Quitar Producto Cerrar Venta

mostrar ClienteVenta query

Carrito de Compra

Lista de Ventas

idVenta: 8 idCliente: 2 Total: 410.0
idVenta: 9 idCliente: 3 Total: 1500.0

7.3.1 Crear Venta

Se introduce el id de un producto que exista, se pulsa en “cargar producto”, se pulsa en “agregar Producto” para añadirlo al carrito, o se pulsa varias veces para aumentar la cantidad del producto, y si ya no se desea agregar mas productos se pulsa en “cerrar venta”

Proyecto Individual

Cientes Productos Ventas

IDProducto: 19

Nombre: piano

Precio: 1000.0

IDCliente:

IDVenta:

Cargar Producto Agregar Producto

Quitar Producto Cerrar Venta

mostrar ClienteVenta query

Carrito de Compra

Lista de Ventas

idVenta: 8 idCliente: 2 Total: 410.0
idVenta: 9 idCliente: 3 Total: 1500.0

18: x2 = 500.0€
19: x1 = 1000.0€

7.3.2 Mostrar Nombre del Cliente de una Venta

Se introduce el id de una venta que exista, y se pulsa “mostrar nombre cliente de una Venta”

Proyecto Individual

Cientes Productos Ventas

IDProducto

Nombre

Precio

IDCliente

IDVenta

Cargar Producto Agregar Producto

Quitar Producto Cerrar Venta

mostrar ClienteVenta query

Carrito de Compra

Lista de Ventas

idVenta 8 idCliente Carlota Ramirez

7.3.3 Mostrar los productos de una venta mayores a un precio dado

Se introduce el id de una venta que exista, y un valor en precio, entonces se mostrara los productos de una venta superiores a ese precio dado.

Proyecto Individual

Cientes Productos Ventas

IDProducto

Nombre

Precio

IDCliente

IDVenta

Cargar Producto Agregar Producto

Quitar Producto Cerrar Venta

mostrar ClienteVenta query

Carrito de Compra

Lista de Ventas

18: guitarra clasica, cuerda: 250.0 € stock: 42
19: piano, percusion: 1000.0 € stock: 48