PECL

PLAYSTORM-Aplicación dedicada a la compra de videojuegos

Grupo 5.
Jorge Fernández Ráez – DNI: 70425739S – jorge.fernandezr@edu.uah.es
Iván Cáceres Gutiérrez – DNI: 54445116E – ivan.caceres@edu.uah.es
Adrián Márquez Mínguez – DNI: 47319572Q – adrian.marquezm@edu.uah.es

Índice

Tabla de contenido

Enunciado

Nuestra idea para esta PECL es la creación de una tienda online de videojuegos.

Esta aplicación se va a encargar de la compra y venta de videojuegos simulando aplicaciones que ya existen en la realidad, como puede ser *STEAM,* simulando una plataforma de distribución digital de videojuegos. En ella los clientes comprarán diferentes videojuegos que luego podrán jugar. Además, se pretende crear una comunidad en la que los clientes puedan valorar positiva o negativamente su satisfacción con el producto en cuestión. Las principales funciones que van a llevar a cabo nuestra aplicación son las siguientes:

- **Gestión de clientes:** Nuestro sistema se va a encargar de almacenar los clientes que se vayan creando en la aplicación. Los clientes que se van a crear son: Administrador, Cliente, Proveedores. Este archivo log contendrá los datos más importantes de estos clientes.
- **Gestión de compras/ventas:** El sistema se encargará de almacenar de los registros de las compras y ventas realizadas en la aplicación. Este archivo almacenara el precio, el producto, la fecha y los clientes implicados en la compra/venta.
- **Gestión de videojuegos:** Cabe destacar que esta aplicación almacenará en un archivo todos los productos, en este caso los videojuegos, que están en venta en la aplicación, además del precio para la compra, o características importantes de cada uno de ellos.

Por lo tanto, esta aplicación deberá almacenar estos datos. Para implementar esto debemos serializar el conjunto de datos explicados en el apartado anterior. Esto mantendrá actualizados toda la aplicación ante cualquier cambio. En principio será una aplicación local con una proyección a futuro de aspecto online.

Tipos de cliente:

Esta aplicación contará con los siguientes clientes:

- **Clientes:** Es el cliente principal que tendrá que registrarse para poder comprar un videojuego. Es el cliente habitual de la aplicación y debe estar registrado para poder acceder a ella. Los principales datos que se podrán almacenar para los clientes: ○ Nombre
  - Apellidos
  - Fecha de
  - nacimiento
  - Localización
  - Tarjeta de crédito
  - Cliente
  - Correo
  - Contraseña
  - Teléfono
  - Monedero

Los clientes tendrán que rellenar obligatoriamente los campos de nombre, fecha de nacimiento, tarjeta de crédito, cliente, correo y contraseña. Por otro lado, el campo monedero es un crédito que deberá abonar el cliente para poder comprar en la aplicación. Este dinero podrá ser recargado siempre que lo necesite, será recargado por el mismo con la finalidad de poder testear la aplicación. El correo deberá tener el formato siguiente: "*cliente@playstorm.es*". La contraseña deberá tener un mínimo de 6, donde tiene que haber mínimo un número, una letra mayúscula y un carácter especial.

- **Administrador:** Serán cuentas proporcionadas por los gestores de la aplicación, en este caso los desarrolladores. Estos administradores podrán eliminar juegos o los clientes que no cumplan la política de la Play Storm. Los principales datos del administrador serán:
    - Correo
    - Contraseña

  Para este cliente no haría falta ningún dato más, ya que no es una cara visible de la aplicación.
- **Proveedor:** Estas cuentas serán utilizadas por las empresas publicadoras de videojuegos. Estas ganaran dinero en la compra de sus productos. Las principales características de estos proveedores son:
    - Nombre compañía
    - Localización
    - CIF
    - Correo
    - Contraseña
    - Productos
    - Monedero
    - Suscrito
    - Código promocional
    - Estudios de Videojuegos

  El atributo de estudios de videojuegos va a ser los nombres de los estudios contratados por los proveedores para desarrollar su producto.

Productos:

Los productos que van a existir en nuestra aplicación van a ser videojuegos. La información, al igual que con los clientes, va a ser almacenada de tal manera que dispondremos de ella en cualquier momento. Por otro lado, los videojuegos contaran con estas características:

- Título
- Empresa
- Precio
- Categoría
- Descripción
- Cantidad

Estas principales características deberán ser incluidas por los proveedores. Todos los productos, una vez incluidos en la aplicación, se pondrán en venta.

Requisitos

Id	Nombre	Descripción	Fuente
RF01	Registrar Cliente	Un cliente podrá registrarse como cliente.	Cliente
RF02	Iniciar Sesión	Un usuario podrá iniciar sesión tras registrarse.	Cliente-ProveedorAdm
RF03	Cerrar Sesión	Un usuario podrá cerrar sesión después de haber iniciado sesión.	Cliente-ProveedorAdm
RF04	Añadir Dinero	Un cliente podrá añadir dinero a su monedero a través de su cuenta.	Cliente
RF05	Ver Perfil	Un cliente podrá ver los datos de su cuenta	Cliente
RF06	Comprar producto	Un cliente podrá comprar un producto seleccionado.	Cliente
RF07	Buscar producto	Un cliente podrá buscar un producto por sus características.	Cliente
RF08	Ver Carrito	Un cliente podrá ver los juegos que tiene en el carrito de compra	
RD01	Factura de Impresión	Un cliente podrá generar una factura de compra.	Cliente
RIU01	Biblioteca	Un cliente podrá ver su biblioteca de productos.	Cliente
RF09	Registrar Proveedor	Un proveedor podrá registrarse.	Proveedor
RF010	Añadir Producto	Un proveedor podrá publicar su producto.	Proveedor
RF011	Eliminar Producto	Un proveedor podrá eliminar un producto.	Proveedor
RIU02	Stock	Un proveedor podrá ver la disponibilidad de sus productos.	Proveedor
RF12	Suscribirse	Un proveedor podrá obtener ventajas por suscribirse.	Proveedor
RF13	Registrar un Estudio	Un proveedor podrá registrar uno o mas estudios de videojuegos en sus productos.	Proveedor
RIU03	Ver Ventas	Un proveedor podrá ver sus ventas.	Proveedor
RD02	Generar Fichero de Ventas	Un proveedor podrá generar un informe para ver sus ventas.	Proveedor
RF15	Modificación de Datos	Un usuario podrá modificar sus datos personales.	Cliente-Provee
RF16	Modificación de Producto	Un proveedor podrá modificar un producto.	Proveedor
Id	**Nombre**	**Descripción**	**Fuente**
RF18	Eliminar Usuario	Un administrador podrá eliminar un usuario.	Administrador
RF19	Eliminar Producto	Un administrador podrá eliminar un producto.	Administrador
RF20	Ver productos	Un administrador podrá ver todos los productos de su aplicación.	Administrador
RT01	Archivos Locales	No tendrá una base de datos online, sino que se guardará la información en ficheros locales.	Diseñador
RT02	Java	El lenguaje empleado para realizar la aplicación es Java.	Diseñador

Manual de Usuario

LogIn.java

Vista principal de la aplicación, en esta vista podremos Iniciar Sesión, ya sea como Administrador, Cliente o como Proveedor.

Por otro lado, podremos Registrarnos.

Registrarse-Option

Cuando pinchemos en el botón de registrase podremos seleccionar como deseamos registrarnos, ya sea como Cliente o como Proveedor.

Registrarse-Datos-Cliente

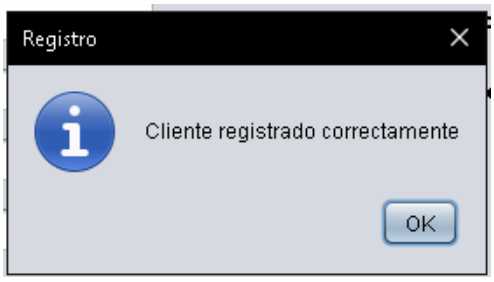
Para llevar a cabo el registro de Clientes, será necesario rellenar todos los campos, en nuestro caso no será necesario rellenar los datos de manera correcta, es decir no tenemos en cuenta la gestión de tipos.

Correo existente

Si introducimos un correo ya existente, nos saltara el siguiente mensaje:

Correo nuevo

Si introducimos un correo nuevo nos saltara el siguiente mensaje:



Registrarse-Datos-Empresa

Con el fin de registrar una nueva empresa será necesario rellenar todos los campos, una vez hemos hecho eso, será necesario darle al botón de Aceptar.

De igual modo será necesario, introducir un correo nuevo o surgirán errores o paneles como hemos visto anteriormente.

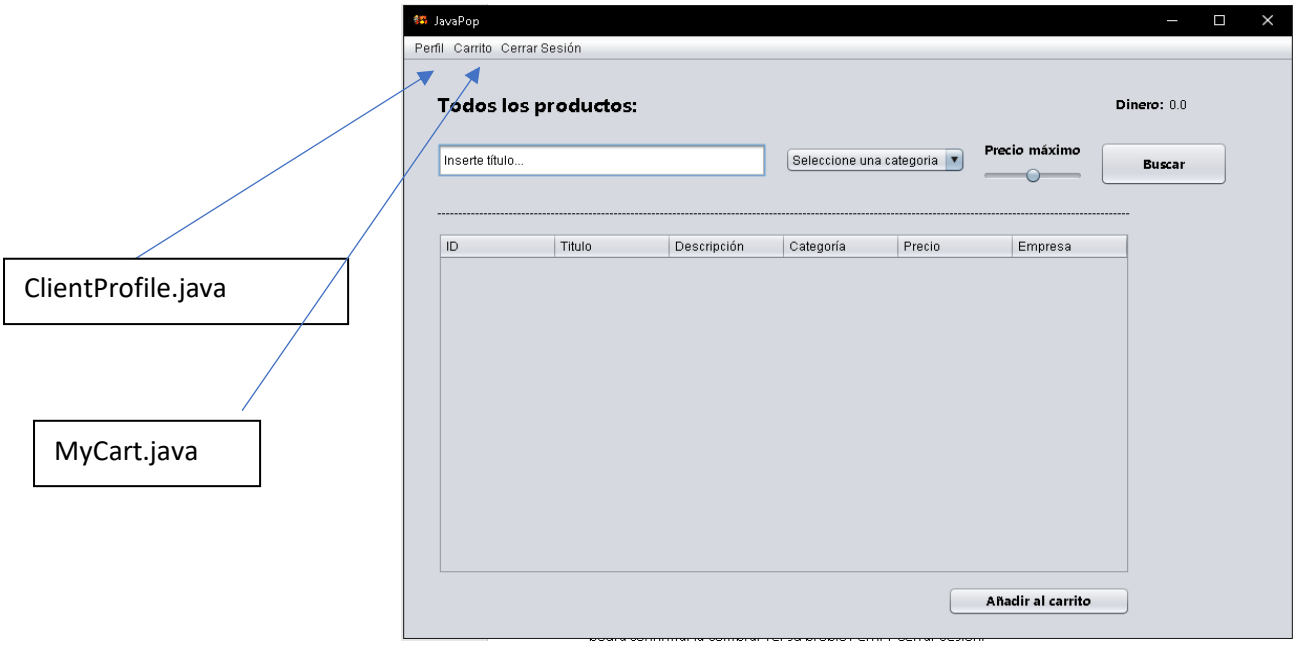
Cliente-Vistas

GameSearch.java

Una vez un Usuario se ha registrado y ha iniciado sesión, este podrá comenzar a usar la aplicación, en la interfaz que mostraremos a continuación podrá: Buscar un juego y añadirlo al carrito, en el cual podrá confirmar la compra, ver su propio Perfil y Cerrar Sesión.

Opciones de Interfaz:

1. Ver mi perfil
2. Ver carrito
3. Cerrar Sesion



En esta interfaz el cliente podrá seleccionar características del producto que busca, siendo estas: Titulo, Categoría, Precio máximo.

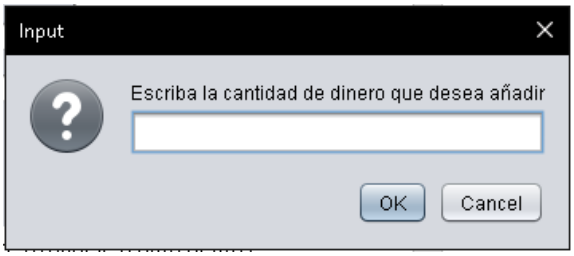
Anadir al Carrito

Si no seleccionamos ningún producto que añadir nos saltara la siguiente advertencia:

Anadir dinero

En la parte superior de la ventana GameSearch.java podremos desplegar un menú en donde podremos añadir dinero de manera manual o ver el perfil.

En este caso nos centraremos en la interfaz para añadir dinero de manera manual, siendo esta:



En donde introduciremos la cantidad que deseamos y se modificara automáticamente en la vista anterior.

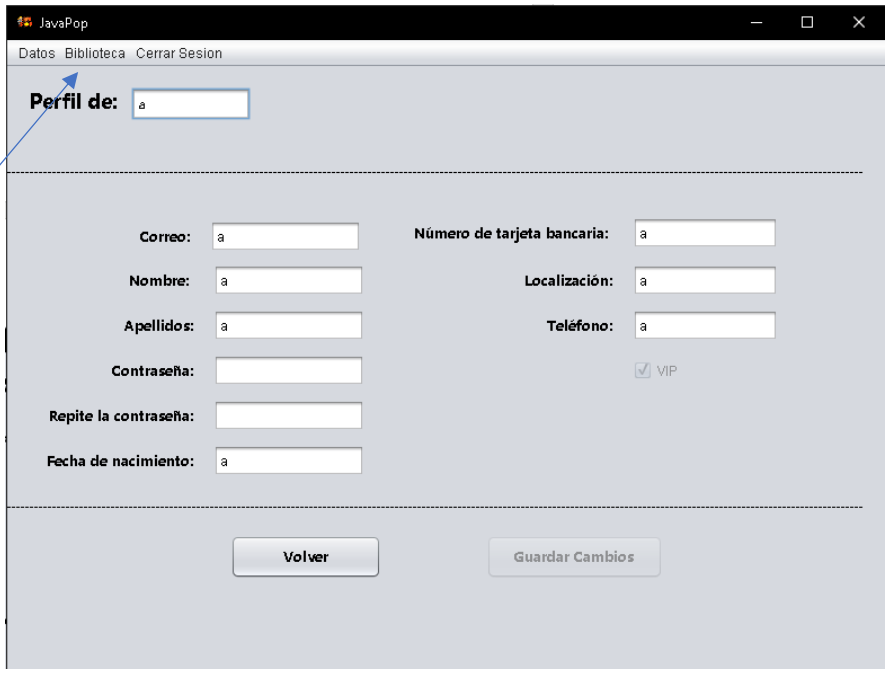
ClientProfile.java

La siguiente interfaz mostrará los datos del cliente en cuestión, en la esquina inferior derecha vemos el botón de Guardar Cambios, este no estará disponible hasta que le demos en la parte superior izquierda a Editar Datos, en donde se permitirá editar los datos del Cliente en cuestión.

Opciones de esta interfaz:

1. Editar Datos
2. Ver mi biblioteca
3. Cerrar Sesión

MyLibrary.java



MyLibrary.java

La segunda opción de ClientProfile.java será ver la Biblioteca personal del cliente activo en este momento, es decir podremos ver los juegos que hemos comprado.

MyCart.java

En esta interfaz podremos ver los juegos que hemos decidido comprar, con el finde confirmar dicha compra o cancelarla, pudiendo hacer esto con los botones **Devolver** y **Comprar.**

Empresa-Vistas

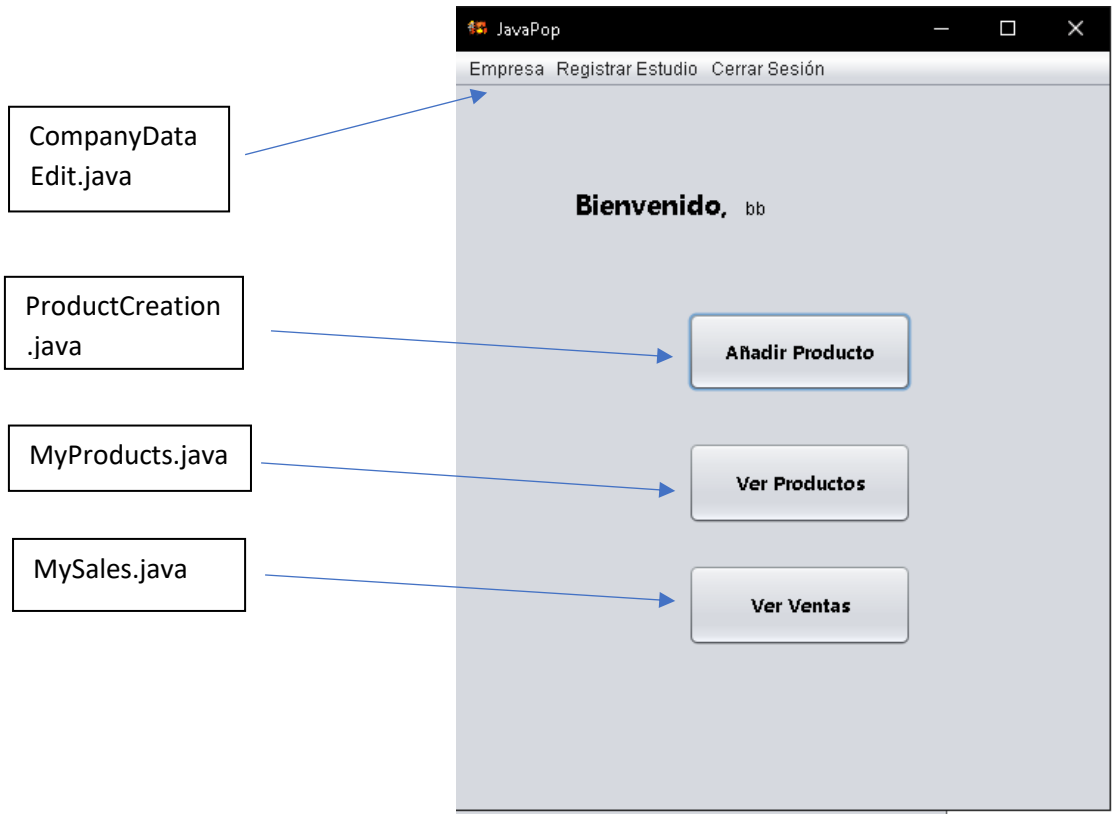
CompanyMenu.java

Una vez iniciamos sesión con una empresa, nos aparecerá la siguiente interfaz:

En esta podremos:

1. Añadir un nuevo producto
2. Ver los productos pertenecientes a la empresa
3. Ver ventas

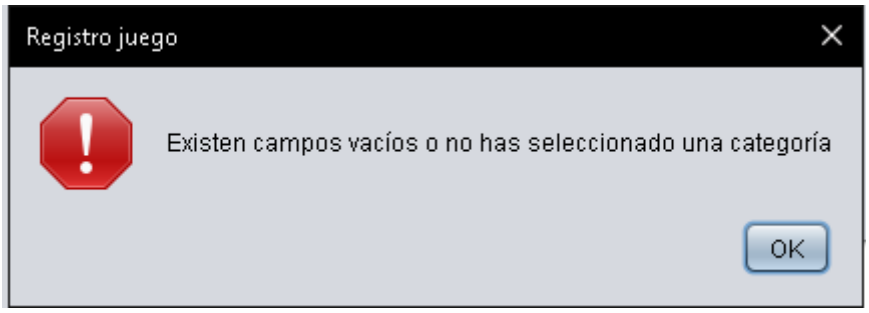
Además de las opciones descritas con anterioridad, podremos editar los datos de una empresa



ProductCreation.java

En esta interfaz podremos crear nuevos juegos, en donde debemos rellenar **todos** los campos, si esto no sucede saltara un error mediante un panel por pantalla.

Crear-Error



MyProducts.java

En esta interfaz podremos ver todos los productos que pertenecen a la empresa, así mismo podremos eliminar los productos que deseemos.

MySales.java

Como hemos descrito anteriormente, en los botones de CompanyMenu.java podemos ver las ventas de nuestra empresa, esto lo implementaremos mediante la siguiente interfaz:

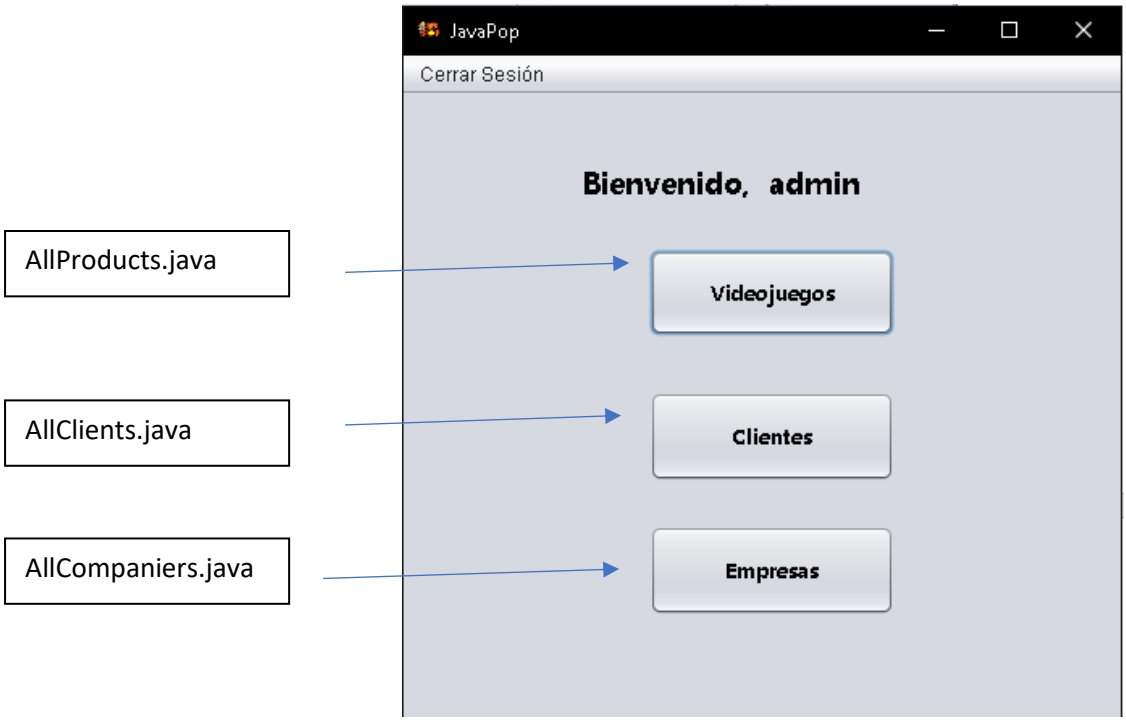
CompanyDataEdit.java

En este caso podremos modificar los datos pertenecientes a tu empresa, una vez modificamos los datos y le damos a aceptar, estos se verán reflejados.

Admin-Vistas

AdminMenu.java

En esta interfaz podremos Cerrar Sesión y realizar 3 acciones , las cuales se describirán a continuación pero las cuales se ejecutaran apretando los botones de la interfaz.



AllProducts.java

En la siguiente interfaz el administrador podrá eliminar un juego en concrete independientemente del proveedor que sea.

Además, en la tabla podremos ver las características de los productos que hay en la web.

AllClients.java

En esta interfaz podremos ver los clientes que están registrados en la aplicación pudiendo eliminarlos, siempre y cuando seleccionemos uno de ellos , si eso no pasa saltara un panel avisándonos de que debemos seleccionar un producto.

Eliminar-Error

AllCompanies.java

En la siguiente interfaz el administrador podrá eliminar una empresa registrada siempre y cuando se haya seleccionado una.

Estudio Patrones

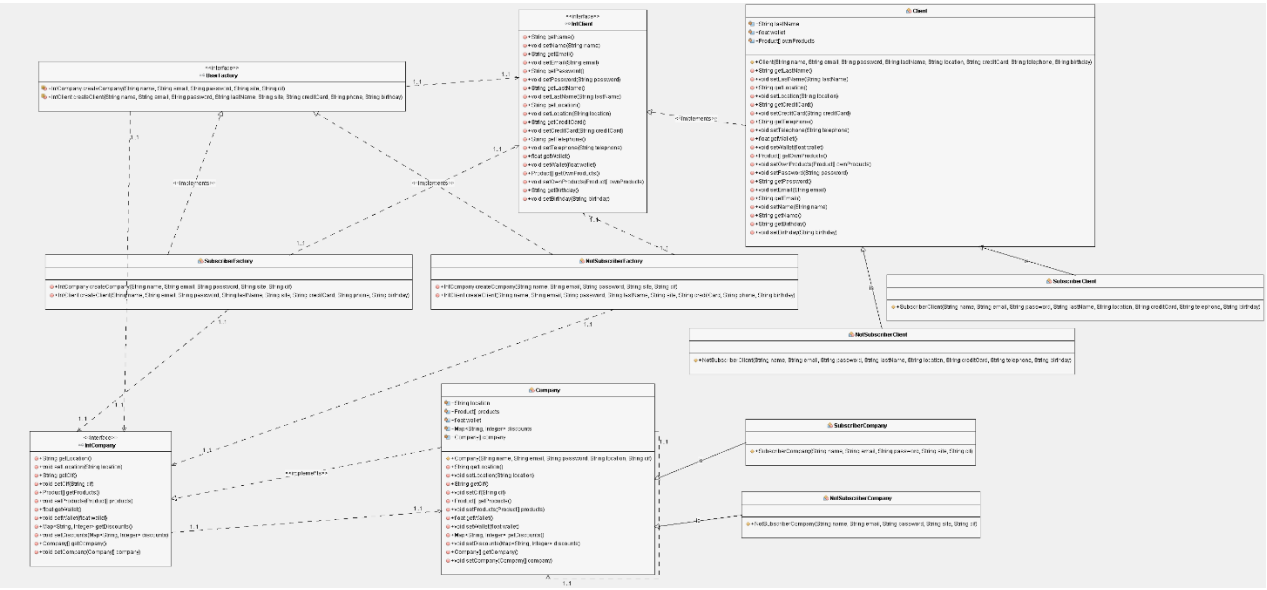
Se muestran a continuación los patrones software empelados en la realización del sistema.

Hemos empleado 2 patrones Creacionales, 2 patrones Estructurales y 6 patrones de Comportamiento.

Creación- Abstract Factory

Utilizaremos el patrón Abstract Factory el cual será utilizado para crear nuevos Usuarios, ya sean Clientes, Proveedores y el Administrador, este patrón permite producir familias de objetos relacionados sin especificar sus clases concretas.

- **Fabrica Abstracta**: AbstractFactory.
- **Fabrica Concreta**: SubscriberFactory, NotSubscriberFactory.
- **Producto** Abstracto: IntClient, IntCompany.
- **ProductoConcreto**: SubscriberClient,SubscriberCompany,NotSubscriberClient, NotSubscriberCompany.
- **Cliente**: ClientSingUp, CompanySignUp.

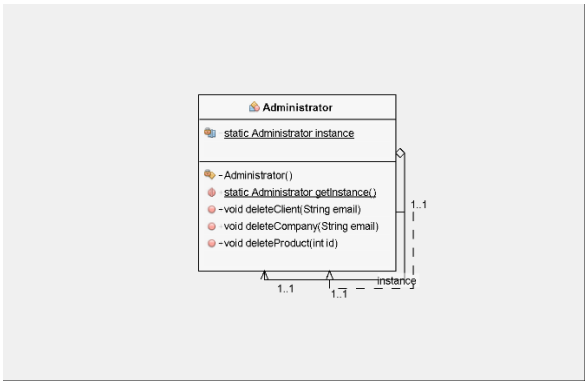


Creación-Singleton

Utilizaremos el patrón Singleton el cual es un patrón de diseño creacional que nos permite asegurarnos de que una clase tenga una única instancia, hemos empleado este patrón para crear el usuario Admin, este será único en la aplicación.

La utilización de este patrón nos permite crear una instancia única para el usuario mas importante de la aplicación.

- **Administrator**: Clase con este patrón.

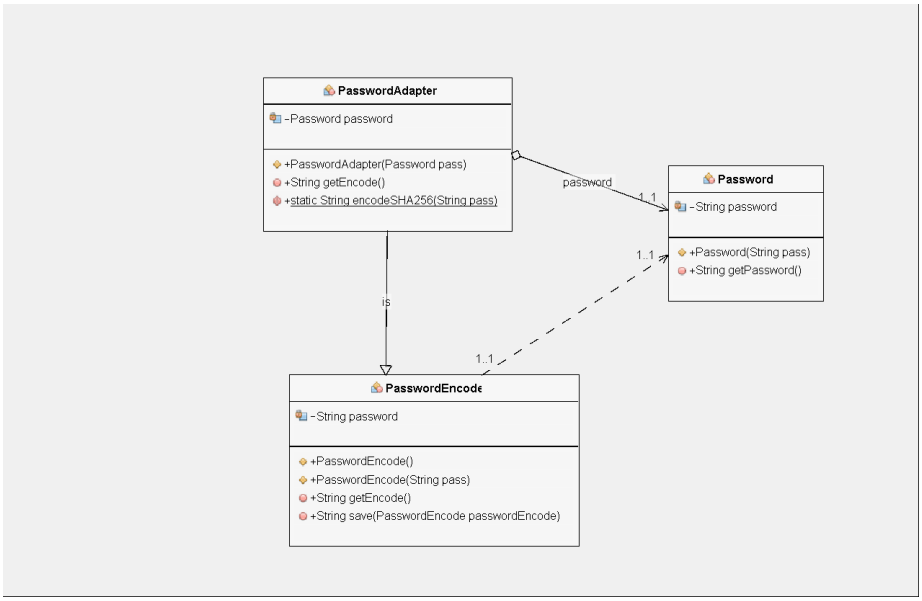


Estructurales-Adapter

El patrón Adapter lo hemos utilizado para el cifrado y descifrado de contraseñas en la aplicación, están serán utilizadas para aportar seguridad a la misma, el funcionamiento es el siguiente:

Un nuevo cliente desea crearse un usuario en nuestra aplicación, para ello introduce todos los datos que le pedimos y crea el usuario. Una vez el usuario esta creado el patrón Adapter se encargará de cifrar dicha contraseña, la cual se mantendrá almacenada a lo largo de la misma, si este usuario desea iniciar sesión la aplicación cifrara la contraseña que ha introducido y la comparara con la contraseña que le corresponde.

- **Objetivo**: PasswordEncode.
- **Adaptable**: Password.
- **Adapatador**: PasswordAdapter.



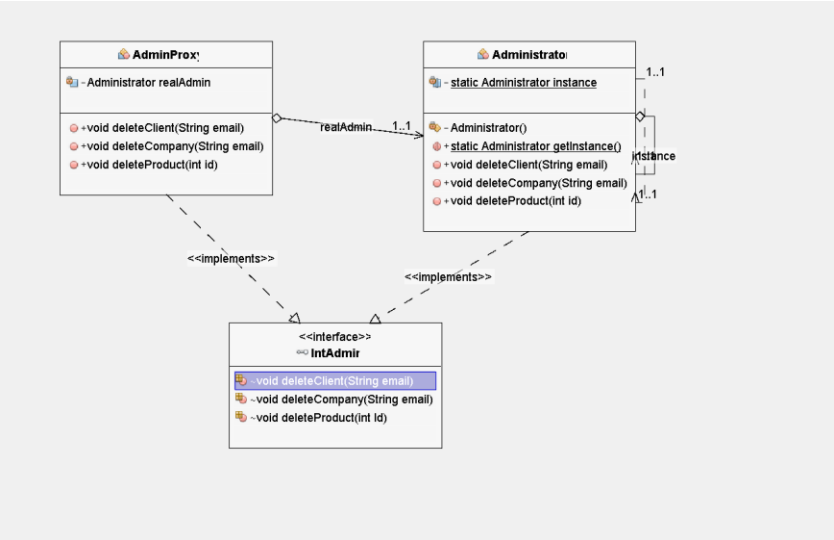
Estructurales Proxy

El patrón Proxy controla el acceso al objeto original, permitiéndote hacer algo antes o después de que la solicitud llegue al objeto original. En nuestro caso será empleado para asegurarnos de que un Administrador es quien realiza la solicitud de eliminar un cliente.

La aplicación de este patrón nos permite aportar seguridad a nuestra aplicación, una falla de estas características daría lugar a la perdida de confianza de nuestros clientes en la aplicación.

- **Sujeto**: Administrator

- **Proxy**: AdminProxy
- **Sujeto Real**: AllProducts, AllClients, AllCompanies
- **Cliente**: IntAdmin



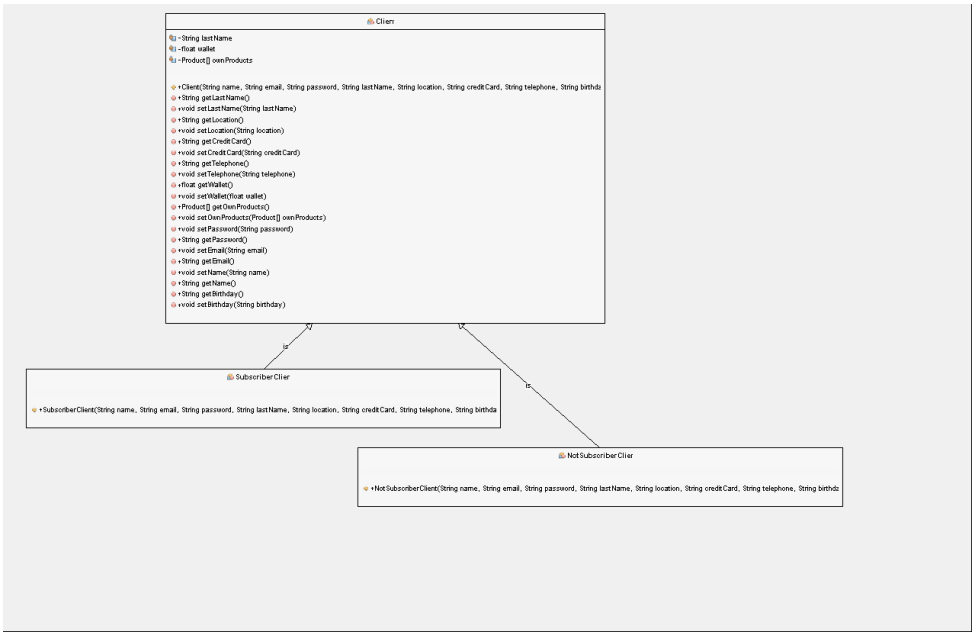
Comportamiento-Template Method

El patrón Template Method es un patrón de diseño de comportamiento que define el esqueleto de un algoritmo en la superclase, pero permite que las subclases sobrescriban pasos del algoritmo sin cambiar su estructura.

En nuestro caso lo utilizamos cuando hacemos una herencia de Client con SubscriberClient y NotSubscriberClient.

En este caso tendremos dos tipos distintos de clientes, por un lado, suscrito y por otro lado no suscrito.

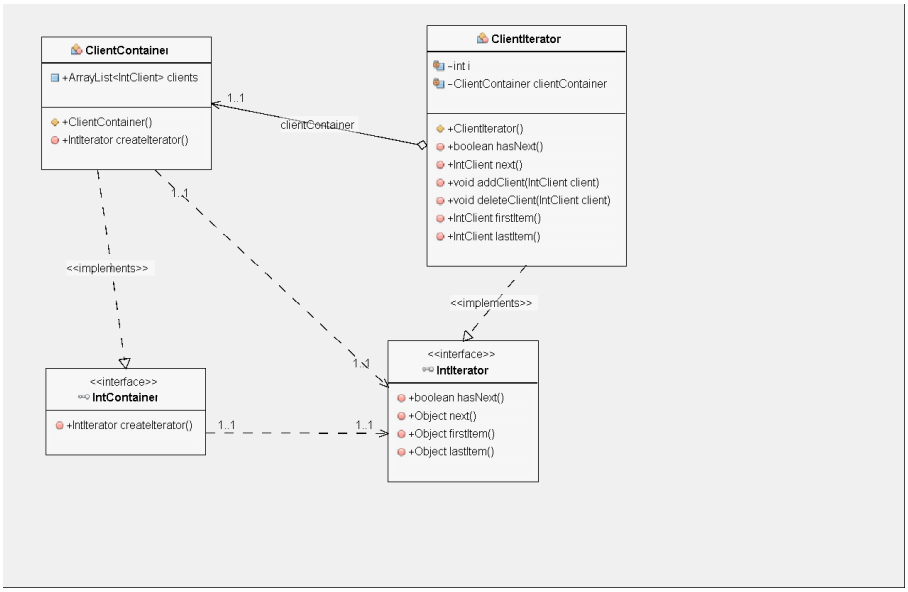
- **Clase Abstracta**: Client.
- **Clase Concreta**: SubscriberClient, NotSubscriberClient.



Comportamiento-Iterator

El patrón Iterator es un patrón de diseño de comportamiento que te permite recorrer elementos de una colección sin exponer su representación subyacente, en nuestra aplicación lo hemos empleado para recorrer listas de objetos.

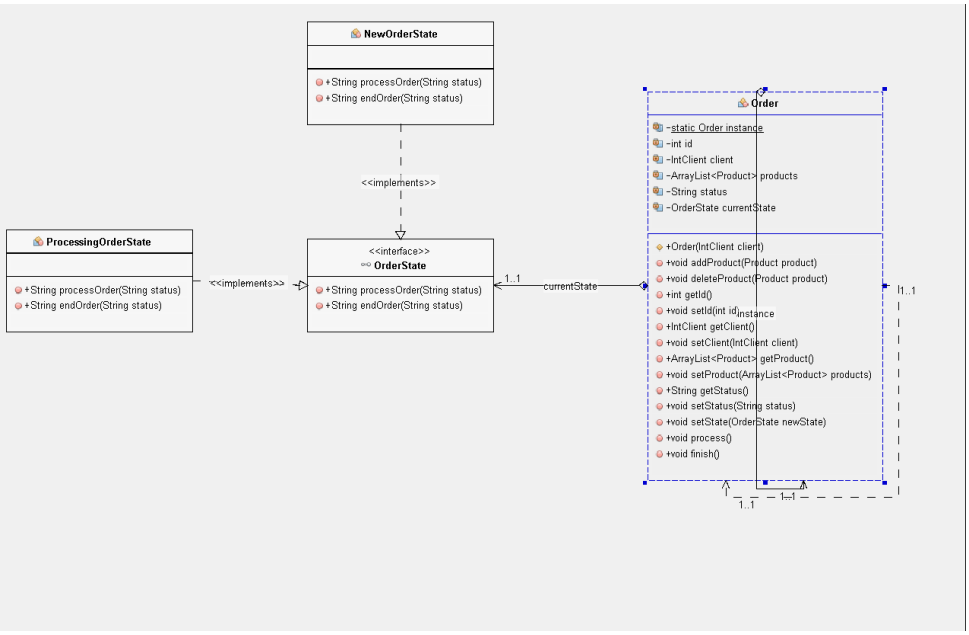
- **Agregado**: ClientContainer, CompanyContainer, ProductContainer
- **Iterador**: ClientIterator, CompanyIterator, ProductIterator
- **Interfaz** Iterador: IntContainer
- **Interfaz Agregado**: IntIterator



Comportamiento-State

El patrón State lo hemos empleado para ver el estado de un pedido en concreto, estos pueden ser finalizado en el caso de que un producto haya sido comprado y en proceso en el caso de que un pedido se haya añadido al carrito.

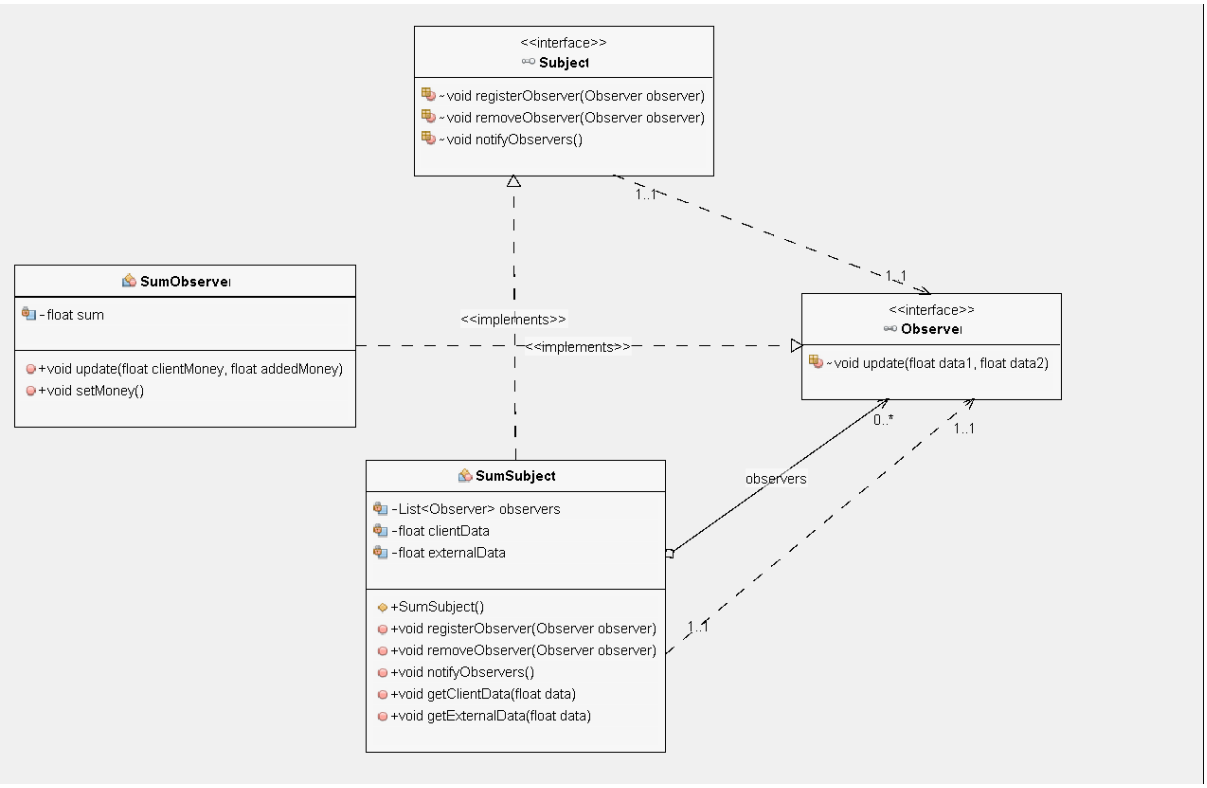
- **Estado**: OrderState
- **Estado Concreto**: NewOrderState, ProcessingOrderState.
- **Contexto**: Order



Comportamiento-Observer

El patrón Observer lo hemos empleado para que un cliente puede añadirse dinero a si mismo para comprar productos.

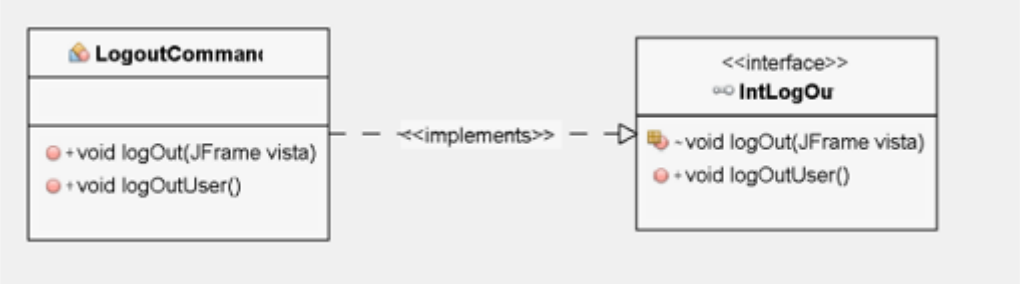
- **Observador Concreto**: SumObserver
- **Sujeto Concreto**: SumSubject
- **Sujeto**: Subject
- **Observador**: Observer



Comportamiento-Command

Patrón utilizado para cerrar sesión, cuando seleccionamos el botón de cerrar sesión el código es implementado mediante el patrón Command.

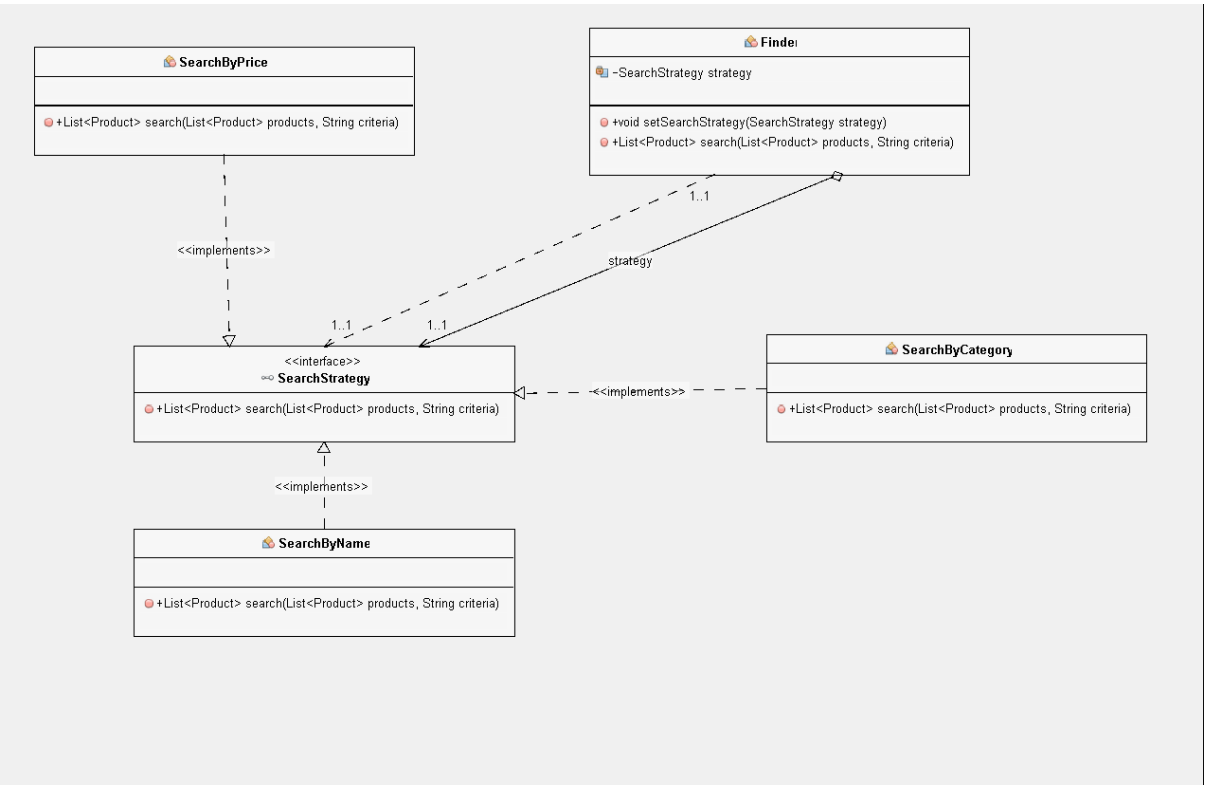
- **Comando**: IntCommand
- **Invocado**: LogOutCommand
- **Invocador**: AdminMenu, CompanyMenu, GameSearch



Comportamiento-Strategy

El patrón Strategy se ha implementado para que cada vez que se realice una búsqueda, dependiendo de los parámetros dados, buscará los productos

- **Context**: Finder
- **Estrategia**: SearchStrategy
- **Estrategias Concretas**: SearchByCategory, SearchByName, SearchByPrice



Diagramas de Casos de Uso

Se adjuntan los siguientes diagramas de casos de uso, tanto para el Administrador como para tanto los clientes y proveedores.

Diagrama de Caso de Uso-Administrador

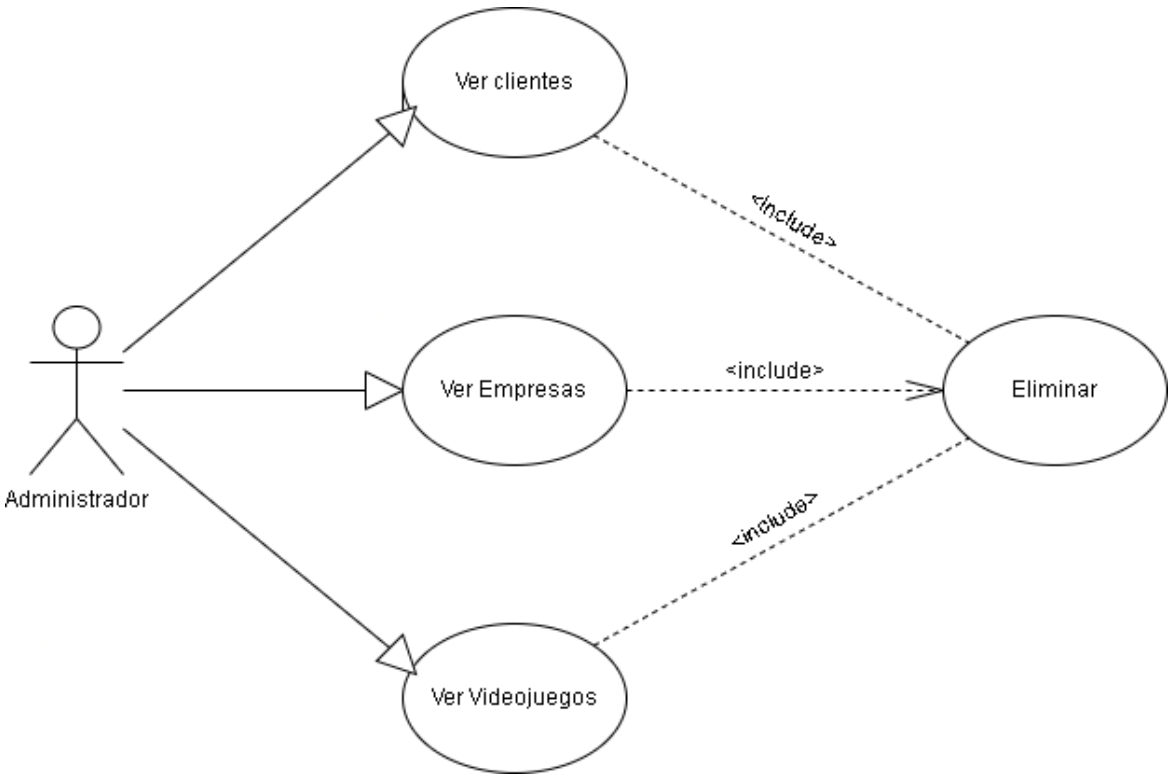


Diagrama de Caso de Uso-Cliente

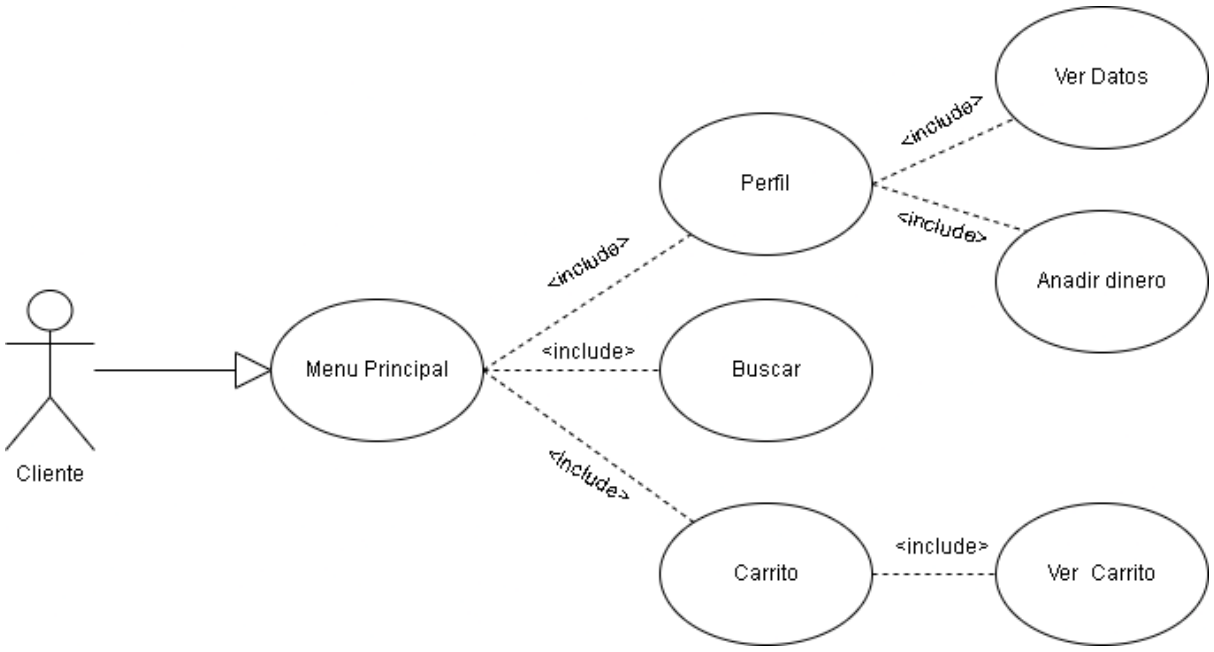


Diagrama de Caso de Uso-Proveedor