

# ***Estrategia***

## ***Trabajo Práctico***

### ***Gestión De Datos***

**FRBA OFERTAS**

**Número de Grupo:** 8

**Nombre de Grupo:** ZTS\_2019

#### **Integrantes**

- Saturni, Agustín
- Rodríguez Takahashi, Tomás
- Zeppa, Agustín

# Índice

---

Manejo Lógico General del Proyecto .....	3
Base de Datos: .....	6
Diagrama de Entidad Relación: .....	6
Descripción de las tablas y Migración: .....	6
Carpeta Interfaces .....	9
Carpeta Menú Administrador .....	11
Carpeta ABM de Usuarios.....	12
Carpeta ABM Clientes.....	13
Carpeta ABM Proveedores.....	16
Carpeta ABM de Rol .....	20
Carpeta Carga de Dirección.....	22
Carpeta Crear Oferta .....	23
Carpeta Comprar Oferta .....	24
Carpeta Entrega y Consumo de Ofertas .....	26
Carpeta Carga de Crédito .....	28
Carpeta Facturar al Proveedor .....	29
Carpeta Listado Estadístico .....	30
Descripción de los Triggers:.....	30
Descripción de los Funciones: .....	31
Descripción de los Procedures:.....	31

# Manejo Lógico General del Proyecto

---

El trabajo práctico "FRBA OFERTAS" implementa diferentes estrategias generales para el funcionamiento genérico del mismo. Se han utilizado patrones de diseño y buenas prácticas de programación, para poder lograr un correcto funcionamiento.

Dicho proyecto cuenta con una carpeta llamada "Manejo Lógico", la cual contiene cuatro clases que dan soporte al mismo para la realización de cada función.

A continuación, se explica detalladamente cada una:

- **Singleton Usuario**: Clase basada en el patrón de diseño singleton, el cual se utiliza para crear un objeto una sola vez y evitar la creación de varios objetos iguales. El mismo, hace referencia a la sesión del usuario al iniciar en el programa. Una vez que el usuario ingresa, dicha clase genera un objeto en donde se cargan las funciones que tiene disponible, los roles y su nombre de usuario. Dicha decisión fue tomada, ya que consideramos que estos datos se necesitan de manera global para realizar verificaciones. La misma se actualiza cuando es necesario, y posee un método para cerrar la sesión. Dicho método, libera el objeto y vacía las listas que poseen los roles y funciones. De esta manera, se evita que al utilizar el programa, se creen muchas sesiones que posteriormente quedan sin utilizar. De más esta decir, que de esta forma, conseguimos llevar a cabo muchas validaciones sin necesidad de consultar a la base constantemente.
- **Conexión BD**: Clase basada en el patrón de diseño singleton, la cual posee el string de conexión de la base de datos. El mismo se carga automáticamente desde el archivo de configuración del sistema. De esta forma, al igual que en el caso anterior, logramos pedir dicho dato siempre que sea necesario, sin necesidad de crear objetos repetidos.
- **HelperControls**: Clase con un único método que devuelve una lista con la totalidad de un cierto tipo de control de un determinado formulario. Se ha implementado, para realizar validaciones exhaustivas, como por ejemplo, verificar que muchos controles de tipo TextBox no estén vacíos.
- **CommonQueries**: Clase basada en el patrón singleton, la cual simplemente se implementó para no repetir estructuras que se utilizan a lo largo del proyecto. Por ejemplo, en las ABMs, se tuvo la necesidad de consultar a la base reiteradas veces, el estado de un rol. Ya que el mismo puede cambiar en cualquier momento si un administrador lo deshabilita.

## **Listados**

En cuanto a los listados del proyecto (proveedores, clientes, ofertas, etc), se utilizó siempre el control Data Grid View. En todos los casos, los atributos de modificación y eliminación que viene por defecto en el control están desactivados. Para todos los casos, según la necesidad y cuando corresponda, se utilizan botones dentro del control para efectuar una acción sobre un registro (Por ejemplo, para dar de baja a un cliente o proveedor, hay un botón de eliminar).

Una característica importante de estos listados es que todos se llenan utilizando una Query dinámica. Es decir, como los filtros de búsqueda de los formularios varían y pueden combinarse según como el usuario desee, se arma el SELECT en Visual Studio para poder filtrar como deseamos.

## **Navegación entre Formularios**

Dado que, en algunos casos, como el registro de un usuario, el formulario que se utiliza es el mismo tanto para las ABM de Clientes y Proveedores, y también para el registro de usuario que viene como opción al inicio del programa, se han utilizado BITS (0, 1 y 2) que se envían por parámetro a los constructores de cada formulario, para determinar de donde venimos. Es decir, para que cuando volvamos para atrás, el sistema no se confunda y por ejemplo vuelva al inicio de sesión en caso de que un administrador este dando un alta.

## **Conexión a la Base por Formulario**

Para cada formulario, se declara el objeto de manera global de la Clase ConexionBD, para poder obtener el string de conexión de la base de datos y una variable "Conexión" de tipo SqlConnection. En todos los formularios, se encuentra el método FORM\_LOAD, en el cual se inicializa la variable "conexión" mencionada.

De esta manera, tenemos 1 sola conexión declara de forma global por formulario, y la cual se abre y cierra cada vez que se necesita consultar algo a la base, o ejecutar algún procedimiento. (conexión.Open() y conexión.Close()).

## **Estrategia para Administrador**

Dado que por enunciado el administrador del sistema puede ejecutar todas las funciones del sistema, se tomó la decisión de que para aquellas funciones propias de clientes o proveedores, se tenga que seleccionar el username sobre el quien impactara dicha función. Es decir, si el administrador desea cargar crédito, deberá elegir el username del cliente al que le desea cargar. Si desea confeccionar una oferta, deberá elegir el username del proveedor que tendrá asignada la oferta.

### **Estrategia de Índices lógicos**

La mayoría de las tablas poseen índices lógicos, autogenerados como una columna más. Se basan en la constraint IDENTITY(1,1).

Esto se implementó para poder facilitar la búsqueda de datos, ordenar las queries por antigüedad (el índice más grande es el registro más nuevo), autogenerar IDs unívocos y facilitar el JOIN sobre tablas.

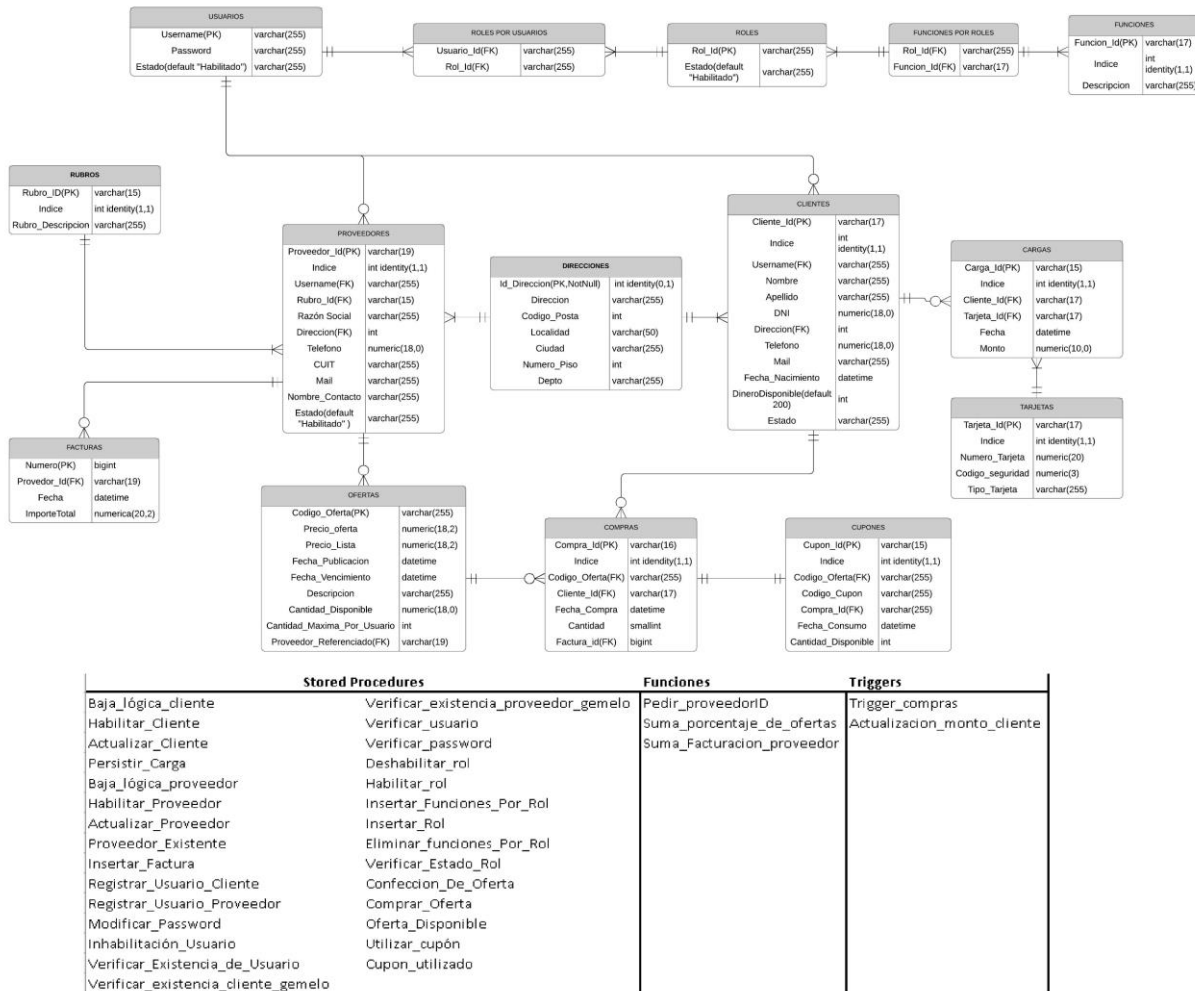
Ejemplo de un ID Autogenerado: ClientelD120. Se concatena el string ClientelD con el Índice.

### **Archivo Configuración**

El mismo se llama app.config. Si se desea cambiar la fecha, se debe respetar el formato (020-02-06 14:10:00.000).

## Base de Datos:

### Diagrama de Entidad Relación:



### Descripción de las tablas y Migración:

- **Usuarios:** se compone de una clave primaria "Username", un campo "Password" y por último un "Estado" ya que se bloquea a un usuario, impidiendo que ingrese al sistema. Los usernames se arman desde los datos que se cargan en las tablas de clientes y proveedores.
- **Roles por usuarios:** se compone de dos claves foráneas "Usuario\_Id" y "Rol\_Id".

- **Roles:** compuesta por una clave primaria, "Rol\_Id" y un campo "Estado" dado que un rol puede ser dado de baja. Esta permitido dar de alta a nuevos roles.
- **Funciones por roles:** compuesta de dos claves foráneas "Rol\_Id" y "Funcion\_Id". Hay determinadas funciones que son únicas de proveedor o de cliente. Por lo tanto, podrán ser asignadas a otro rol pero no se podrán utilizar. Se tomo la decisión de dejar que se asignen al rol como base para una mejora en el futuro, pero para dicha entrega no se permite que por ejemplo un proveedor "cargue crédito".
- **Funciones:** su clave primaria es "Funcion\_Id", además se compone de un "Índice" y una "Descripción". Son fijas en el sistema.
- **Proveedores:** se compone de una clave primaria "Proveedor\_Id", un "Índice", una "Razón Social", un "Teléfono", un "CUIT", un "Mail", un "Nombre de contacto" y un campo "Estado" ya que puede darse de baja el rol o al proveedor en si. Además tiene tres claves foráneas: "Username", "Rubro\_id" y "Dirección". Todos sus campos son obligatorios menos el nombre de contacto. Se tomo dicha decisión ya que se considero que es un campo que no tiene relevancia sobre la funcionalidad del trabajo practico. Los username son la concatenación de las siglas "pr" mas 4 caracteres del cuit. Ejemplo: PR1224-2. Además, todos los proveedores migrados tienen como email su username@gmail.com.
- **Clientes:** su clave primaria es "Cliente\_id", se compone de un campo "Índice", un "Nombre", un "Apellido", un "DNI", un "Teléfono", un "Mail", una "Fecha de nacimiento", una "Cantidad de dinero Disponible" y un campo de "Estado". Sus claves foráneas son "Username" y "Direccion". Los nombres de usuario de dicha tabla para los clientes que se migraron de la tabla maestra fueron autogenerados por la concatenación del nombre y 3 caracteres del DNI.
- **Direcciones:** su clave primaria es "Id\_Dirección", además se compone de una "Dirección", un "Código Postal", una "Localidad", una "Ciudad", un "Número de piso" y un "Departamento". Los únicos campos obligatorios de esta tabla son la dirección y la ciudad. Dicha decisión fue tomada dado que la tabla maestra no posee ninguno de los otros campos, y consideramos que quedaría inconsistente si todos los otros son NULL, siendo estos campos obligatorios. Esta tabla no permite duplicados exactamente iguales. En caso de que haya dos usuarios con la misma dirección, se les asigna el mismo ID.

Para entender mejor todos los casos posibles de modificación de dirección, ver procedure "actualizar cliente".

- **Rubros:** su clave primaria es "Rubro\_Id" y tiene los campos "Índice" y "Rubro\_Descripción". Los rubros se pueden dar de alta solo en el momento de registrar un nuevo proveedor.
- **Facturas:** compuesta de una clave primaria "Número", una "Fecha" (día en la que se confecciona la misma), un "Importe total" y una clave foránea "Proveedor\_Id".
- **Oferta:** la clave primaria de esta tabla es "Código de Oferta". Se compone de los campos: "Precio\_oferta", "Precio\_Lista", "Fecha\_Publicación", "Fecha\_Vencimiento", "Descripción", "Cantidad\_Disponible", una "Cantidad\_Máxima\_por\_Usuario". Como claves foráneas la tabla OFERTAS se compone por un "Proveedor\_referenciado". Dado que la tabla maestra tenía muchos registros exactamente iguales, variando solo el código de oferta, se eliminó esos duplicados, tomando uno de los códigos de oferta al azar como identificador. Dicha decisión fue tomada ya que consideramos ilógico que existan 2 o mas códigos de oferta con exactamente los mismos datos; al menos debe variar la fecha de publicación. Pero no puede haber 2 ofertas exactamente iguales, que se publiquen y venzan el mismo día.
- **Carga:** su clave primaria es "Carga\_id", sus campos son: "Índice", "Fecha" (en la que se realizó la carga), "Monto" (como el monto de la carga realizada). Sus claves foráneas son: "Cliente\_Id" y "Tarjeta\_Id". Dicha tabla se utiliza a modo de "historial de cargas". Para la migración de los datos, se dejó la posibilidad de que haya registros repetidos, ya que consideramos que en un sistema puede haber muchas cargas en un mismo momento por una persona. Por lo cual, existen registros repetidos.
- **Tarjetas:** su clave primaria es "Tarjeta\_Id", además se compone de un "Índice", un "Numero\_Tarjeta", un "Código de seguridad" y un "Tipo de tarjeta" (débito, crédito). Dicha tabla sirve para validar si hay tarjetas repetidas o no. Es decir, al momento de hacer una carga registramos la tarjeta que usamos. Si nuevamente deseo cargar con la misma tarjeta, los datos deben coincidir. Esta fue la única validación que se tomó, ya que es coherente que no exista 1 tarjeta con datos distintos. No se tomó en cuenta que una tarjeta pertenezca a un usuario, ya que en el ingreso de los datos podríamos usar la tarjeta de cualquier persona. Consideramos que para hacer validaciones a ese nivel, deberíamos tener información de un banco.



- **Compras:** su clave primaria es "Compra\_Id", además se compone de un "Índice", una "Fecha\_Compra" y una "Cantidad". Sus claves foráneas son: "Código\_Oferta", "Cliente\_id" y "Factura\_id".
- **Cupones:** su clave primaria es "Cupon\_id", además se compone de un "Índice", un "Código\_Cupón", una "Fecha\_Consumo" y una "Cantidad\_Disponible". Sus claves foráneas son "Código\_Oferta" y "Compra\_Id". Pueden existir varios cupones relacionados a una misma oferta.

A continuación se explicara detalladamente cada función del sistema, con todas las consideraciones que fueron tomadas en cuenta.

## Carpeta Interfaces

---

Para la navegación a través del proyecto, se han implementado diferentes formularios, los cuales se encuentran en la carpeta "Interfaces".

- **Formulario Inicio de Sesión**

Componentes:

- Botón para Iniciar Sesión: se valida que el usuario y contraseña sean correctos. De ser así, se ingresa al menú principal.  
En caso de que el usuario exista, pero la contraseña no sea la correcta, se informara a través de un error la situación. En caso de que dicho evento suceda 3 veces, se bloquea al usuario, no permitiendo de que ingrese al sistema.  
En caso de que el usuario con el que se desea ingresar no exista, no se tomara el caso de bloqueo; simplemente se informa de que no existe el mismo.
- Botón Registrarse: Permite registrar un usuario. Dicho botón nos lleva a un formulario de registro, en donde se elige entre rol de Cliente o de Proveedor. Para la solución planteada, no se permite registrar administrador.
- Botón Cambio de Contraseña: Permite cambiar la Contraseña de un usuario existente.

Login



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

**Usuario**

**Contraseña**

**Iniciar Sesión**

**Registrarse**

**Cambiar Contraseña**

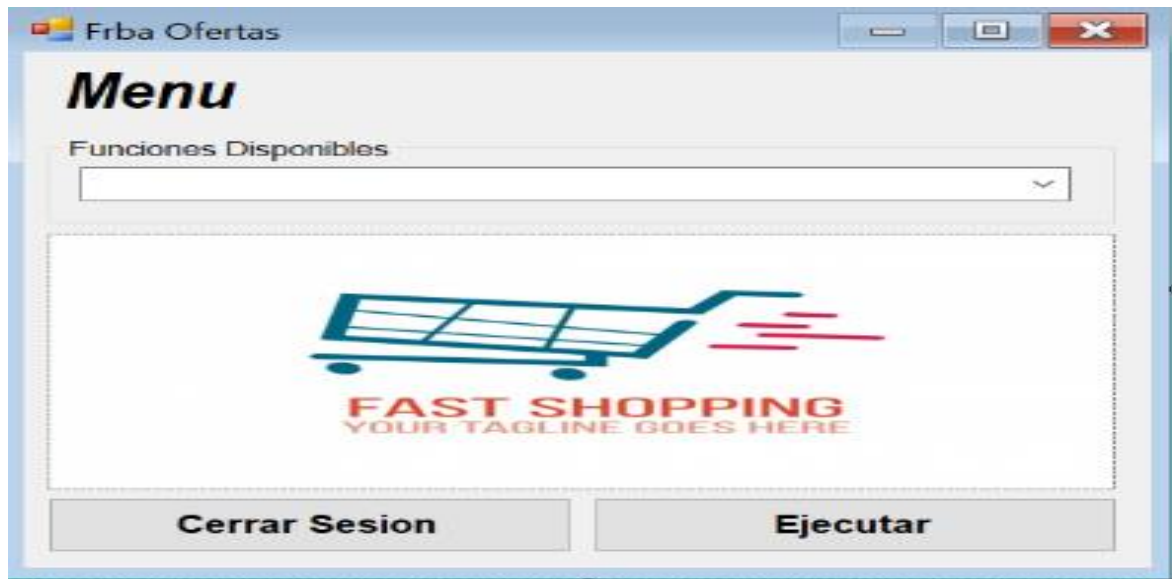
- **Formulario Menu\_Principal**

El siguiente formulario, es el menú principal del sistema, a través del cual se accede a las funciones disponibles para cada usuario.

Componentes

- Combo Box de Funciones Disponibles: dicho combo nos permite visualizar las funciones que tenemos disponibles según nuestros roles. Así por ejemplo, de ser un Administrador, se nos desplegaran todas las funciones del sistema. Una vez seleccionada una, se toca el botón de ejecutar y nos permite navegar hacia la misma. Solo se mostraran las funciones que se encuentren en la lista "funciones" del singleton de usuario. (Recordar que en el mismo se encuentran las funciones, los roles y el username)
- Botón ejecutar: Inicia la función seleccionada.

- Botón Cerrar Sesión: Nos regresa al formulario de inicio de sesión. Limpia todos los datos del singleton usuario, como mencionamos anteriormente.



## Carpeta Menú Administrador

Dicha carpeta, contiene tres formularios relacionados a las interfaces de navegación que utiliza el administrador para poder acceder a las ABMs de clientes, proveedores y roles. Estos formularios, no poseen demasiada complejidad, simplemente se utilizan a para organizar el acceso a las diferentes funcionalidades.

- **Formularios de Menú de Cliente y Menú de Proveedor**

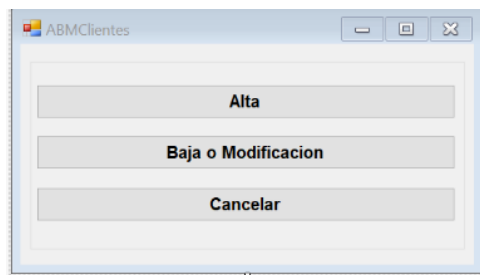
Poseen los mismos componentes, variando muy poco su funcionalidad:

- Botón de Altas: nos conduce al formulario de Registro de Usuarios mencionado anteriormente. En caso de que se acceda al formulario de registro de usuarios desde el menú de ABM de Clientes, el Combo Box que especifica el rol que se asigna al nuevo usuario será autoseleccionado con la opción de cliente y se bloqueara no permitiendo la modificación del mismo. Dicho esto, de manera similar funciona si accedemos desde el menú de ABM Proveedores, bloqueando el combo box como corresponda para dicho caso.

IMPORTANTE: Para ambos casos, se valida que el rol al que se

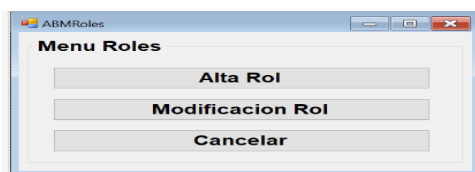
hace referencia (cliente o proveedor) este HABILITADO. Si por alguna razón, el administrador del sistema decidió inhabilitar un rol, no se permiten realizar altas sobre el mismo hasta que lo habilite nuevamente.

- Botón de Bajas o Modificaciones: Simplemente, nos conduce al formulario de listado de clientes o proveedores. La especificación de los mismos será detallada en la siguiente sección.
- Botón Cancelar: Nos conduce al formulario de menú principal.



- **Formulario de Menú de Roles**

Posee los mismos botones mencionados anteriormente para Menú Cliente y Menú Proveedor. En este caso, no hay validaciones para acceder a los mismos. Simplemente hay que ser administrador para que dicho menú este disponible.



## Carpeta ABM de Usuarios

Dicha carpeta contiene el formulario genérico de registro de usuarios y el de cambio de contraseña.

Nota Importante: no se permite registrar usuarios en caso de que un rol este inhabilitado.

- **Formulario Registro de Usuario**

Aquí, se encuentra la lógica para cargar el nombre de usuario y contraseña que deseamos, así como también seleccionar si deseamos ser cliente o proveedor. Se ha utilizado el algoritmo SHA2\_256 para encriptar la contraseña.

Todos los datos son obligatorios, y se verificara en este momento que no exista un usuario previamente registrado con dicho nombre, independientemente de si es proveedor o cliente.

Componentes

- Botón Siguiente: si todas las validaciones son correctas, se procede al formulario de alta de cliente o alta de proveedor, para registrar los datos correspondientes para cada caso.
- Botón volver: Se regresa al formulario de inicio de sesión. (Ver nota)

**Nota:** Si el alta del usuario viene desde de las ABM de Clientes o proveedores, el botón cancelar nos regresa al menú del administrador que da las altas. En cambio, si estamos registrándonos como un usuario estándar al inicio del programa, volveremos al inicio de sesión.

- **Formulario Cambio de Contraseña**

En este caso, se verifica que el username y la contraseña actual son correctos, y en ese caso, se toma la contraseña nueva para hacer un update sobre la tabla de usuarios. Al finalizar, se vuelve al inicio de sesión.

## Carpeta ABM Clientes

---

Dicha carpeta, contiene los formularios relacionados a las altas, bajas y modificaciones que se realizan sobre los usuarios que tenga asignado el rol de cliente. Los mismos son: alta de clientes, listado de clientes y modificaciones de clientes. Los últimos dos funcionan conjuntamente.

Recordamos que, para el caso de las altas de clientes que se accede desde el menú de ABM de Clientes, primero se navega hacia el formulario de registro de usuario, para poder cargar un username, password y bloquear el combo box de roles (no permitiendo seleccionar proveedores). Además, una vez

llegada a esta instancia, recordar que ya se validó si el username que se desea dar de alta se encuentra registrado o no.

- **Formulario Alta de Clientes**

Con este formulario, se permite cargar los datos específicos que se registran en la tabla de clientes. En esta instancia se valida que no haya clientes gemelos; ya que puede haber 2 usernames distintos pero que se carguen los mismos datos de cliente; EL DNI ES UN VALOR UNICO E IRREPETIBLE. Dicho esto, el resto de los datos se pueden repetir pero el DNI no. Además, los datos ingresados son sometidos a múltiples validaciones, donde se informa de forma específica cual es el error en caso de no cumplir alguno de los siguientes requisitos:

- Todos los campos son obligatorios.
- El DNI es una cadena únicamente numérica, de 8 caracteres. (VALOR UNICO)
- Nombre y Apellido son cadenas únicamente no numéricas.
- El teléfono es una cadena únicamente numérica.
- El mail y el nacimiento no posee validaciones.

#### Componentes

- Botón Siguiente: nos conduce a un formulario para cargar la dirección asociada al cliente. Este formulario es genérico, y se accede tanto desde el alta de cliente como proveedor. Al clickear este botón, se valida si existen clientes gemelos o no, como mencionamos antes.
- Botón Volver: nos conduce nuevamente al formulario de registro de usuarios.



- **Formulario Listado de Clientes**

Como dice el nombre, el mismo consiste en un listado con todos los clientes del sistema, permitiendo filtrar por diferentes campos: nombre, apellido, dni o email. Cada filtro se hace con la función LIKE de SQL, salvo el DNI, que debe ser un texto exacto.

A través de este listado, podemos dar una baja lógica a un cliente, o modificar sus datos. Como buena práctica, todo cliente posee un campo de estado, que determina si está habilitado o no. Por lo tanto, la baja consiste en cambiar este campo, nunca se lo elimina de la tabla.

Los filtros no poseen validaciones, ya que en caso de ingresar basura, no se encontrara nada en la base. Las únicas restricciones, son que no se puede hacer click sobre los botones de eliminar y modificar si el listado esta vacío.

Nota Importante: luego de una modificación, el formulario de listado se carga nuevamente, para poder actualizar los datos de la base. Decidimos esto, ya que el data grid view se carga inicialmente, y ante una modificación hay que refrescarlo.

## Componentes

- Contenedor de Clientes (Data Grid View): control que se utiliza para listar los clientes.
- Botón Seleccionar: ejecuta una query para llenar el contenedor. Dado que los campos para filtrar dicho contenedor son dinámicos, y puede utilizarse cualquier combinación deseada, se utiliza una query dinámica. Es decir, se arma un string dinámicamente para poder asignar los filtros que deseamos al SELECT de SQL.
- Botón Eliminar (Dentro del Data Grid View): invoca al procedimiento de SQL correspondiente para dar de baja a un cliente. Si el cliente ya está dado de baja, se informa que ya está inhabilitado, abortando la ejecución del procedimiento.
- Botón Modificar (Dentro del Data Grid View): nos conduce al formulario de modificaciones. El mismo será explicado en el siguiente apartado.
- Botón Limpiar: limpia los text boxes de filtros, dejándolos todos vacíos.
- Botón Cancelar: nos lleva al menú de ABMs de Clientes.

### • Formulario Bajas y Modificaciones de Clientes

Aquí, se realizan los cambios sobre un cliente que se selecciona en el listado, así como también la habilitación del mismo en caso de que este inhabilitado. La lógica de este formulario es simple: mostramos todos los datos del contenedor, y además, damos la opción de modificar aquellos que estén permitidos.

En un primer instante, todos los text box están bloqueados, con la propiedad ReadOnly.

Existen múltiples validaciones sobre los datos que se ingresan:

- Todos los campos son obligatorios menos algunos de la dirección (ver apartado de dirección).
- DNI Cadena únicamente numérica de 8 caracteres
- Nombre y Apellido solo STRINGS
- Email no poseen validaciones

Componentes:

- Botón Modificar Clientes: Me habilita los campos que pueden ser modificados. Desactiva la propiedad ReadOnly.
- Botón Guardar Modificación: Ejecuta el procedimiento de SQL actualizar cliente. Guarda cualquier modificación que se haya efectuado.
- Botón Habilitar Cliente: Habilita un cliente en caso que este inhabilitado. Si no, nos informa de la existencia de un error.
- Botón Cancelar: Nos regresa al formulario de menú de clientes.

The screenshot shows a Windows application window titled "Modificaciones\_Clientes". The form is divided into two main sections: "Información del Cliente" on the left and "Ubicación del Cliente" on the right. The "Información del Cliente" section contains fields for ID, Username, Nombre, Apellido, DNI, Fecha de Nacimiento (with a date picker showing "jueves , 12 de dicierr"), Telefono, and Email. The "Ubicación del Cliente" section contains fields for Estado, Direccion, Ciudad, Localidad,Codigo Postal, N° Piso, and Depto. At the bottom of the form, there are four buttons: "Modificar Cliente", "Guardar Modificacion", "Habilitar Cliente", and "Cancelar".

## Carpeta ABM Proveedores

Dicha carpeta, contiene los formularios relacionados a las altas, bajas y modificaciones que se realizan sobre los usuarios que tenga asignado el rol de proveedor. Los mismos son: alta de proveedor, listado de proveedor y modificaciones de proveedor. Los últimos dos funcionan conjuntamente.

Recordamos que, para el caso de las altas de proveedor que se accede desde el menú de ABM de proveedores, primero se navega hacia el formulario



de registro de usuario, para poder cargar un username, password y bloquear el combo box de roles (no permitiendo seleccionar clientes). Además, una vez llegada a esta instancia, recordar que ya se validó si el username que se desea dar de alta se encuentra registrado o no.

- **Formulario de Alta de Proveedores**

Con este formulario, se permite cargar los datos específicos que se registran en la tabla de proveedores. En esta instancia se valida que no haya proveedores gemelos; ya que puede haber 2 usernames distintos pero que se carguen los mismos datos de proveedor; LA RAZON SOCIAL Y EL CUIT SON VALORES UNICOS E IRREPETIBLES. Además, los datos ingresados son sometidos a múltiples validaciones, donde se informa de forma específica cual es el error en caso de no cumplir alguno de los siguientes requisitos:

- Todos los campos son obligatorios menos el nombre de contacto.
- CUIT debe ser una cadena de 13 caracteres con el siguiente formato: 12-34567423-1
- Teléfono cadena numérica únicamente.

Componentes:

- Botón Siguiente: nos conduce a un formulario para cargar la dirección asociada al cliente. Este formulario es genérico, y se accede tanto desde el alta de cliente como proveedor. Al clickear este botón, se valida si existen proveedores gemelos o no, como mencionamos antes.
- Botón Volver: nos conduce nuevamente al formulario de registro de usuarios.



- **Formulario Listado de Proveedores**

Como dice el nombre, el mismo consiste en un listado con todos los proveedores del sistema, permitiendo filtrar por diferentes campos: razón social, CUIT, rubro o email. Cada filtro se hace con la función LIKE de SQL, salvo el CUIT, que debe ser un texto exacto.

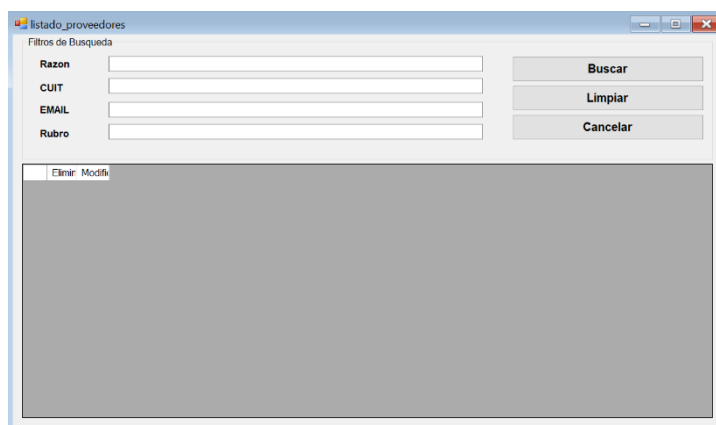
A través de este listado, podemos dar una baja lógica a un proveedor, o modificar sus datos. Como buena práctica, todo proveedor posee un campo de estado, que determina si está habilitado o no. Por lo tanto, la baja consiste en cambiar este campo, nunca se lo elimina de la tabla.

Los filtros no poseen validaciones, ya que en caso de ingresar basura, no se encontrara nada en la base. Las únicas restricciones, son que no se puede hacer click sobre los botones de eliminar y modificar si el listado esta vacío.

Nota Importante: luego de una modificación, el formulario de listado se carga nuevamente, para poder actualizar los datos de la base. Decidimos esto, ya que el data grid view se carga inicialmente, y ante una modificación hay que refrescarlo.

### Componentes:

- Contenedor de Proveedores (Data Grid View): control que se utiliza para listar los proveedores.
- Botón Seleccionar: ejecuta una query para llenar el contenedor. Dado que los campos para filtrar dicho contenedor son dinámicos, y puede utilizarse cualquier combinación deseada, se utiliza una query dinámica. Es decir, se arma un string dinámicamente para poder asignar los filtros que deseamos al SELECT de SQL.
- Botón Eliminar (Dentro del Data Grid View): invoca al procedimiento de SQL correspondiente para dar de baja a un proveedor. Si el cliente ya está dado de baja, se informa que ya está inhabilitado, abortando la ejecución del procedimiento.
- Botón Modificar (Dentro del Data Grid View): nos conduce al formulario de modificaciones. El mismo será explicado en el siguiente apartado.
- Botón Limpiar: limpia los text boxes de filtros, dejándolos todos vacíos.
- Botón Cancelar: nos lleva al menú de ABMs de Clientes.



- **Formulario Bajas y Modificaciones de Proveedores**

Aquí, se realizan los cambios sobre un proveedor que se selecciona en el listado, así como también la habilitación del mismo en caso de que este

inhabilitado. La lógica de este formulario es simple: mostramos todos los datos del contenedor, y además, damos la opción de modificar aquellos que estén permitidos.

En un primer instante, todos los text box están bloqueados, con la propiedad ReadOnly.

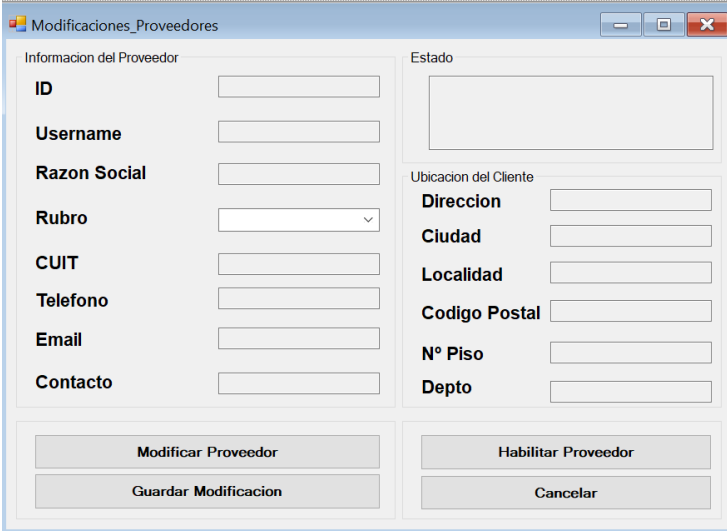
Para el caso de los rubros, se utiliza un combo box para mostrar los rubros disponibles que hay en el sistema. No se puede dar de alta a un rubro en esta sección. Existirá un nuevo rubro solo en el caso de que se registre un nuevo proveedor e ingrese uno que no existe.

Existen múltiples validaciones sobre los datos que se ingresan:

- Todos los campos son obligatorios menos algunos de la dirección (ver apartado de dirección) y contacto.
- CUIT Cadena de 13 caracteres con formato "24-34529853-2"
- Teléfono cadena numérica únicamente.
- Email no poseen validaciones.

#### Componentes:

- Botón Modificar Proveedor: Me habilita los campos que pueden ser modificados. Desactiva la propiedad ReadOnly.
- Botón Guardar Modificación: Ejecuta el procedimiento de SQL actualizar proveedor. Guarda cualquier modificación que se haya efectuado.
- Botón Habilitar Proveedor: Habilita un proveedor en caso que este inhabilitado. Si no, nos informa un error.
- Botón Cancelar: Nos regresa al formulario de menú de proveedores.



## Carpeta ABM de Rol

En la presente carpeta, se encuentran los formularios de alta de roles y modificación o bajas de roles.

Los roles son cliente, proveedor y administrador, como especifica el enunciado. Para el presente proyecto se consideró que no habrá asignación de varios roles a un único usuario. Es decir, no se encuentra configurada la función que permita esto. Sin embargo, el proyecto está configurado para que esto sea posible a futuro al tener una tabla de roles por usuario.

La inhabilitación de un rol implica la baja automática de todos los usuarios asignados a dicho rol. En caso de volverlo a activar, no se actualizan automáticamente. Se deberá dar de alta uno por uno a los que se deseen. Además, no se pueden ejecutar ningún tipo de alta o registro de usuario con un rol inhabilitado.

- **Formulario Alta de Roles**

Con esta funcionalidad lo que se logra es la creación de un nuevo rol y la asignación de funciones a ese rol. Se verificará que el rol a crear no exista.

Componentes:

- Con el botón ">" se pueden asignar las funciones.
- Con el botón "<" podemos desasignarlas.
- El botón finalizar, ejecuta los procedimientos "insertar rol" y "insertar\_funciones\_por\_Rol".
- El botón cancelar retorna al formulario de "Menu Principal".

Alta de Roles

Introduzca el nombre del nuevo ROL

Seleccione las Funciones Deseadas

Funciones Disponibles del Sistema

list\_totales

Funciones a Asignar

list\_rol

>

<

Cancelar

Finalizar

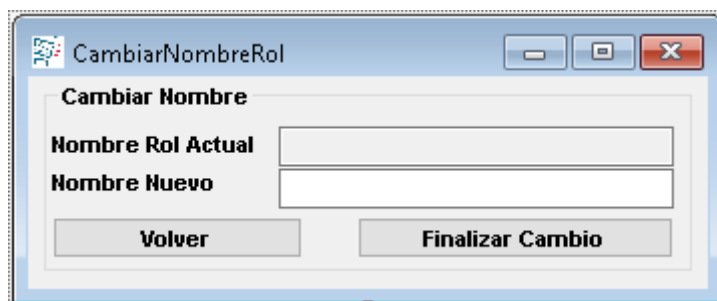
- **Cambiar nombre Rol:**

Con esta funcionalidad podemos cambiar el nombre de un Rol. A la hora de realizar el cambio se verifica que no exista otro rol con ese nombre nuevo ni que el nombre nuevo sea igual al nombre actual del rol.

Es importante mencionar, que aunque se permite el cambio de nombre de rol, la función no está completamente implementada, sino que se la deja como base para futuras mejoras. Debido a que un cambio el nombre de un rol, implicaría que muchas validaciones que se hacen sobre "Clientes" y "Proveedores" dejen de funcionar. Se necesitaría que dichas validaciones sean dinámicas con el nombre que esta persistido en la base, y no se consideró lo suficientemente importante para validar en este proyecto.

COMPONENTES:

- **Finalizar:** se confirma el cambio de nombre, se dispara el procedure "actualizar cliente". Luego te retorna al formulario modificaciones de rol.
- **Cancelar:** te retorna al formulario de modificaciones de rol



- **Modificación de Roles:**

Con esta funcionalidad podemos asignarle o sacarle funciones a un rol. Podemos habilitar y deshabilitar roles e incluso modificarles el nombre. Se tomo en cuenta que es posible modificar las funciones de un rol, pero existen funciones que son propias de clientes o proveedores. Es decir yo podre asignar "confección de ofertas" a un cliente, pero este no podrá usarla al momento de seleccionarla.

COMPONENTES:

- **Cambiar nombre rol:** inicia el formulario cambiar nombre rol.

- **Finalizar modificación:** guarda los cambios que hayamos hecho en cuanto a la asignación de funciones.
- **Habilitar:** habilita el rol. Dispara el procedure "Habilitar\_Rol".
- **Deshabilitar:** deshabilita el rol. Ejecuta el procedure "Deshabilitar\_Rol".
- **>:** asigna funciones.
- **<:** desasigna funciones.
- **Volver:** retorna al formulario de menú principal.
- **Combo box ROL:** muestra los roles actuales.

## Carpeta Carga de Dirección

En esta sección se cargan los datos de la dirección que está asociada a un cliente o proveedor. Dicho formulario es genérico y se accede desde cualquiera de las altas.

NOTA IMPORTANTE: Los procedimientos de SQL de registro de clientes y proveedores se efectúan en este formulario al tocar el botón de finalizar. Se tomó esta decisión ya que era conveniente navegar entre varios formularios enviando por parámetros los datos de registro para no crear varios formularios idénticos.

Es importante tomar en cuenta que la tabla maestra no tenía ningún dato registrado sobre los códigos postales, localidades, números de piso y departamentos. Además, consideramos que era muy complicado actualizar

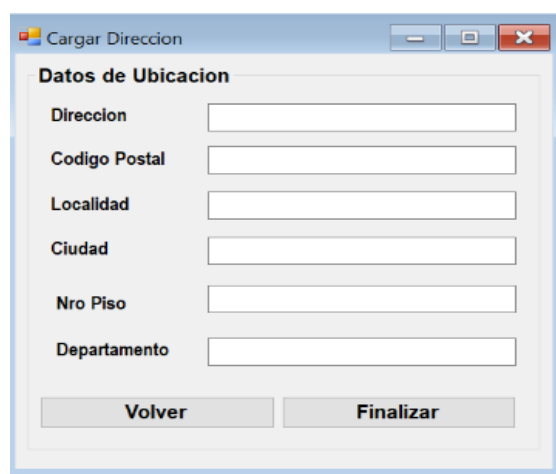
dichos campos para las direcciones que ya se encuentra registradas por la migración de datos efectuada al principio del proyecto. Es por estas razones, que se permite no registrar esos 4 valores

Validaciones que se realizan sobre los datos ingresados:

- Código postal numérico y de 4 dígitos.
- Numero de Piso numérico únicamente
- Departamento debe ser solo Letras. Ejemplo "A".
- Ciudad y Localidad no permiten números

Componentes

- Botón Finalizar: Finaliza la creación de un cliente o proveedor, asignándole por último los datos de ubicación. Al finalizar el procedimiento de SQL correspondiente dependiendo que tipo de alta se estaba realizando, se vuelve al menú de cliente, o menú de proveedor, o inicio de sesión.
- Botón volver: regresa a alta de cliente o alta de proveedor, dependiendo cual se esté realizando.



## Carpeta Crear Oferta

---

- **Crear Oferta**

En este formulario el proveedor y el administrador, pueden confeccionar ofertas para que sean publicadas y compradas por clientes. Si la funcionalidad fuera accedida por un administrador, el mismo deberá ingresar el proveedor al cual referencia la oferta. Se realizan las verificaciones necesarias sobre todos los campos, inclusive las fechas en los procedimientos explicados más adelante.

Carga de Oferta

**Proveedor Username**

**Descripcion**

**Fecha de Publicacion**

**Fecha de Vencimiento**

**Precio Oferta**

**Precio Lista**

**Cantidad Disponible**

**Cantidad máxima por Usuario**

Publicar Oferta

Volver

## Carpeta Comprar Oferta

En esta carpeta se encuentran los formularios de: ofertas, Cantidad a comprar y Elegir clientes.

**Ofertas** Compuesto de un "data grid view" en el cual se cargarán, para la vista del cliente, las ofertas publicadas siempre y cuando no estén vencidas y ya hayan sido publicadas. Se puede realizar una búsqueda por rango de precios

Ofertas

Filtros de Búsqueda

**Descripcion**


Buscar

Limpiar

Volver

☐ Rango de Precios

**Desde**  **Hasta**

	Comprar	Descripcion	precio_oferta	Precio_lista	fecha_publicacion	fecha_vencimiento	Cantidad_disponibl	can
		Una experiencia i...	129,00	387,00	6/2/2020	7/2/2020	11	11
		Cena para 2 pers...	138,00	276,00	5/2/2020	7/2/2020	10	10
		Cena para 2 pers...	143,00	429,00	6/2/2020	7/2/2020	13	13
		Una experiencia i...	74,00	148,00	5/2/2020	7/2/2020	5	5
		Carnes Argentina...	83,00	166,00	6/2/2020	7/2/2020	10	10
		2 noches para 2 ...	31,00	31,00	5/2/2020	7/2/2020	10	10
		4 noches para 2 ...	119,00	119,00	6/2/2020	8/2/2020	9	9
		3 noches para 2 ...	69,00	207,00	6/2/2020	7/2/2020	9	9
		3 noches por per...	28,00	56,00	5/2/2020	7/2/2020	8	8
		una plancha a va...	82,00	82,00	5/2/2020	7/2/2020	8	8
		Una experiencia i...	136,00	272,00	6/2/2020	7/2/2020	13	13

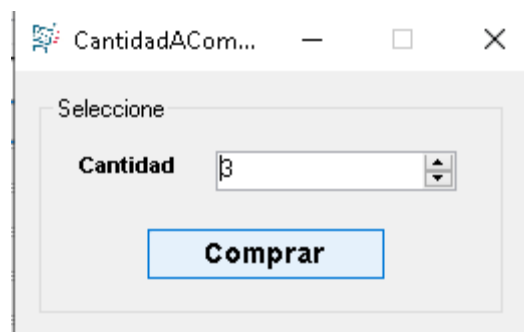


o no, eso es a elección del cliente e incluso completando una descripción de la oferta que desea comprar o no.

**Nota:** El botón “limpiar” refresca el formulario. El botón comprar, nos lleva al siguiente formulario explicado a continuación. En este listado solo se cargarán las ofertas que se encuentren publicadas antes de la fecha actual y con una fecha de vencimiento posterior a la actual. Por ende, el usuario nunca podrá comprar una oferta que no esté publicada o vencida.

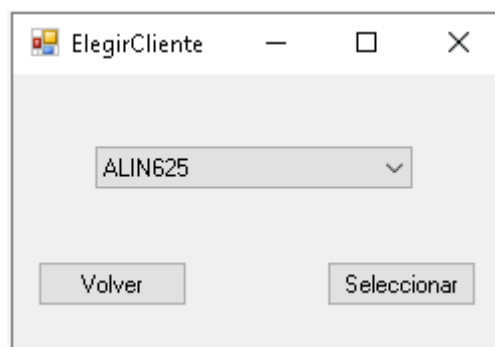
- ***Cantidad a Comprar***

Formulario hecho para que el cliente pueda determinar qué cantidad de una misma oferta desea comprar. Y finalice la compra.

A screenshot of a Windows application window titled 'CantidadACom...'. The window has a standard Windows title bar with minimize, maximize, and close buttons. Inside the window, there is a label 'Seleccione' above a text box labeled 'Cantidad' which contains the number '3'. Below the text box is a blue button with the text 'Comprar'.

- ***Elegir Cliente:***

Esta funcionalidad está hecha para que un administrador pueda usar la función de comprar oferta a través del uso de la cuenta de un cliente. Al presionar el botón “Seleccionar” lo lleva al formulario de compra de oferta con los datos del cliente elegido (“Usuario”, “Dinero disponible”, etc).

A screenshot of a Windows application window titled 'ElegirCliente'. The window has a standard Windows title bar with minimize, maximize, and close buttons. Inside the window, there is a dropdown menu showing 'ALIN625'. Below the dropdown menu are two buttons: 'Volver' and 'Seleccionar'.

# Carpeta Entrega y Consumo de Ofertas

---

En la presente carpeta encontramos los formularios "Entrega Consumo Oferta" y "Listado Cupones".

**Nota:** solo se listan los cupones cuya fecha de publicación sea igual o menor a la fecha del archivo de configuración ya que facilita las verificaciones. Si no existiera esta validación habría muy pocos casos en los que las entregas del cupón estén efectivas.

En caso de que acceda a esta funcionalidad un administrador, este puede canjear cualquier oferta de cualquier proveedor y en el caso de que la acceda un proveedor solo se listan las ofertas que tiene asociadas como proveedor. Además, dado el gran volumen de ofertas disponibles con periodos de disponibilidad muy cortos (publicación a vencimiento), se muestran solo aquellos cupones cuya fecha de oferta se comprende entre la fecha de publicación y fecha de vencimiento.

Consideramos que no se puede canjear en el día de publicación y vencimiento.

- **Listado Cupones**

En esta pantalla podremos buscar los cupones, el cual el proveedor le canjee a un cliente. Dicho cupón, además de estar relacionado con un cliente, estará relacionado con una oferta, a la cual pertenece, por lo que también será posible ingresar el id de la oferta asociada.

El formulario nos permite ingresar los campos que nosotros elijamos, creando una query dinámica, que se completará a medida que llenemos los campos seleccionados, teniendo la posibilidad también de no completar ningún campo, así obteniendo la totalidad de los cupones.

La visualización de los cupones esta limitada por un checkbox con la leyenda "Incluir Cupones ya consumidos", dejando la posibilidad de visualizar los cupones sin uso y usados (si este componente esta seleccionado), o solo los que están sin uso (si el checkbox no está seleccionado), es decir los cupones que son canjeables al momento de la visualización.

Una vez buscados los cupones, con el componente DataGridView cargado, tendremos la posibilidad de entregar este cupón al cliente, siempre y cuando no haya sido utilizado, caso en el cual se mostrara el mensaje de error.

Si no nos encontramos con un impedimento para canjear el cupón, pasaremos al siguiente formulario denominado *Entrega Consumo Oferta*.



Entrega de Ofertas

## Detalles

Código de cupón

84670EF2-0533-4A49-80DF-4B99C234B718

Fecha de consumo

jueves , 6 de febrero de 2020

Cliente Username

SINA1545

Volver

Entregar Oferta

## Carpeta Carga de Crédito

---

- **Carga de Crédito**

En este formulario se le da la posibilidad al cliente de realizar la carga de crédito del dinero que utiliza el sistema para poder realizar las compras de las ofertas. Las cargas se realizan únicamente con tarjetas, y pueden elegir entre tarjetas de crédito y débito. Esta función puede ser accedida por un administrador, en ese caso, el combobox se completa con todos los clientes del sistema para que el administrador decida a quien quiere cargarle saldo.

Carga de Credito

Carga

Usuario

agustin

Tipo de Pago

Monto

Número de la Tarjeta

Código de Seguridad

Saldo Actual

\$3354

Cargar

Cancelar

**Nota:** una vez realizada la carga se refresca el monto del saldo actual del cliente.

Cuando esta funcionalidad es accedida por el Administrador, el campo de Usuario debe ser ingresado por el mismo para realizarle la carga a un cliente.

## Carpeta Facturar al Proveedor

- Facturar al proveedor**

Función que tiene el administrador en la cual a través de la elección de dos fechas puede visualizar las ofertas facturadas y no facturadas o ambas, de un proveedor determinado. Las ofertas no facturadas pueden ser facturadas creando sus facturas correspondientes.

Facturación a Proveedor

Proveedor

pr111-2

Visualizar

Desde

martes , 1 de octubre de 2019

Facturar

Hasta

viernes , 1 de mayo de 2020

Volver

Visualizacion

Ambos

Total Facturado: 0

	Codigo_Oferta	Descripcion	Compra_Id	Facturado	Fecha_compra
▶	EELVBCXUCR7	Pistola Pulverizador...	CompralD10714	143,00	9/1/2020
	EFVRQZEJND4	Auriculares Sony. ...	CompralD10841	36,00	11/1/2020
	EGHXROKTRF12	Teléfono Inalámbr...	CompralD10886	139,00	20/2/2020
	AACANBNYWE13	una plancha a vap...	CompralD11	41,00	21/2/2020
	EJAMBWLRSU11	Pistola Pulverizador...	CompralD11164	120,00	8/1/2020
	EJUDMWDFRZ10	Cámara Nikon S25...	CompralD11230	25,00	29/4/2020
	EJUDMWDFRZ10	Cámara Nikon S25...	CompralD11231	25,00	28/4/2020
	EKCJOYLFCD11	Slender Shaper. La...	CompralD11273	26,00	28/4/2020
	EKDPXFOFZJ10	Slender Shaper. La...	CompralD11278	62,00	16/1/2020

29

## Carpeta Listado Estadístico

- **Listado Estadístico**

El listado estadístico es una funcionalidad del administrador que le va a permitir generar dos consultas a nivel semestral sobre ciertos aspectos de los proveedores:

- **Mayor porcentaje de Descuento:** nos muestra el TOP 5 de los proveedores que más porcentaje de descuentos ofrecieron en sus ofertas vendidas para el semestre elegido. La consulta nos muestra: El Id de proveedor, la suma total de los porcentajes de descuento de ese proveedor, la cantidad de ofertas publicadas en el semestre y la cantidad de ofertas vendidas.
- **Mayor Facturación:** nos muestra el TOP 5 de los proveedores que mayor facturación alcanzaron en el semestre elegido. La consulta nos muestra: El proveedor Id, la facturación total para ese proveedor en el semestre y la cantidad de facturas realizadas.

proveedor_id	Porcentaje_de_Descuento	Cantidad Ofertas Publicadas	Cantidad de Ofertas Vendidas
ProveedorID9	30216,67	964	964
ProveedorID4	29766,67	968	968
ProveedorID7	29216,67	1004	1004
ProveedorID17	28866,67	957	957
ProveedorID31	28850,00	952	952

## Descripción de los Triggers:

- **Trigger\_compras:** este es un trigger puesto en la tabla compras que se dispara ante un insert. La función de este trigger es la de insertar un nuevo cupón en la tabla de cupones y actualizar las cantidades de stock en la tabla OFERTAS. Además actualiza el saldo del cliente en la

tabla *CLIENTES*. Se verifica que el cliente no haya alcanzado la cantidad máxima de compras por clientes de la oferta, y que tenga saldo suficiente.

- **Actualizacion\_monto\_cliente:** trigger puesto sobre la tabla de cargas, realiza la actualización del saldo del cliente en la tabla *CLIENTES* luego de realizar una carga.

## Descripción de los Funciones:

---

- **Pedir\_proveedorID:** función que devuelve un proveedor id luego de recibir un username.
- **Suma\_porcentaje\_de\_ofertas:** esta función nos calcula la suma en porcentaje de descuentos en ofertas que hizo un proveedor entre dos fechas.
- **Suma\_Facturacion\_proveedor:** esta función nos calcula la suma de los montos de las facturas que realizó un proveedor entre dos fechas determinadas.

## Descripción de los Procedures:

---

- **Baja\_lógica\_cliente:** Este procedure recibe el DNI del cliente y el username. Verifica que exista ese usuario en el sistema, lo borra de la tabla *Roles\_Por\_Usuario* y actualiza la tabla *CLIENTES* modificando el campo "Estado" de habilitado a inhabilitado. En el caso de que no se encuentre el Usuario en la tabla *Roles\_Por\_Usuario* significa que el cliente ya esta inhabilitado.
- **Habilitar\_Cliente:** recibe el DNI del cliente y el usuario, verifica que **no** exista ese usuario en *Roles\_Por\_Usuario* y si la condición da verdadera entonces hace un update a la tabla *CLIENTES* modificando el campo "Estado" de inhabilitado a habilitado. Si el usuario existe en la tabla *Roles\_Por\_Usuario* entonces significa que el Cliente ya fue habilitado.
- **Actualizar\_Cliente:** la función del procedure es la de modificar los campos de un cliente. La primera validación que se realiza es que aquellos campos que son únicos por cliente no se repitan cuando inserto nuevos valores en la modificación. Lo segundo que se valida es que no haya direcciones duplicadas y que al modificar una dirección la misma no impacte sobre varios usuarios.
- **Persistir\_Carga:** este procedure recibe el usuario del cliente al cual se le va a realizar la carga, el monto que se va a cargar y los datos de la tarjeta. Lo primero que se hace es verificar que la tarjeta exista y coincidan sus datos. Si existe, pero no coinciden el número con el código de seguridad se aborta la operación. En el caso de que no

- exista la tarjeta se la agrega a la tabla TARJETAS y se realiza el insert a la tabla CARGAS. (**NOTA: ver trigger actualización\_monto\_cliente**).
- **Baja\_lógica\_proveedor:** recibe el CUIT y el usuario del proveedor. Verifica que exista el proveedor en la tabla ROLES\_POR\_USUARIO, en el caso de que exista, borra el registro de la tabla y hace un update en la tabla PROVEEDORES actualizando el valor "Estado" de habilitado a inhabilitado. En el caso de que no exista el proveedor, significa que ya está inhabilitado y lanza un error.
  - **Habilitar\_Proveedor:** recibe el CUIT y el usuario del proveedor. Verifica que no exista el proveedor en la tabla ROLES\_POR\_USUARIO, en el caso de que no exista, inserta el registro en esta tabla y actualiza el valor "Estado" de inhabilitado a habilitado en la tabla de PROVEEDORES. Si el proveedor ya existe en la tabla de ROLES\_POR\_USUARIO significa que ya fue habilitado y lanza un error.
  - **Actualizar\_Proveedor:** igual al procedure actualizar\_cliente incluyendo la lógica de la tabla rubros.
  - **Proveedor\_Existente:** recibe un usuario y verifica que exista el proveedor, si no existe en la tabla PROVEEDORES lanza un error.
  - **Insertar\_Factura:** recibe un proveedor id, la fecha en la que se crea la factura (sacada del config), el número de la factura a crear y el importe.
  - **Registrar\_Usuario\_Cliente:** este procedure recibe un username y una contraseña, los datos personales del cliente incluyendo los datos de la dirección donde vive. Comienza insertando en la tabla USUARIOS, luego verifica que la dirección en la que vive el nuevo cliente no exista ya en la tabla DIRECCIONES, y por último ingresa los datos del nuevo cliente. En el caso de que ya exista esa dirección, le asigna la dirección id respectivo al cliente nuevo.
  - **Registrar\_Usuario\_Proveedor:** recibe un usuario y una contraseña y los datos del cliente nuevo incluyendo los datos de la Dirección. En el caso de que ya exista el rubro que se ingresa, funciona igual a la lógica de la dirección. Se le asigna al nuevo proveedor el rubro id correspondiente.
  - **Modificar\_Password:** recibe un usuario, la contraseña que se corresponde con ese usuario y la nueva contraseña a actualizar. Este procedure verifica que exista el usuario (en el caso de que no exista lanza un error), luego verifica que la contraseña actual se corresponda con ese usuario y por ultimo verifica que la nueva contraseña no sea igual a la actual. Si cumple con todos esos requisitos se actualiza la contraseña en la tabla USUARIOS encriptándola con el método SHA2\_256.
  - **Inhabilitación\_Usuario:** recibe un usuario y hace un update en la tabla USUARIOS actualizando el estado a inhabilitado.



- **Verificar\_Existencia\_de\_Usuario:** recibe un nombre de usuario y se fija que no exista en la tabla USUARIOS. Si existe lanza un error.
- **Verificar\_existencia\_cliente\_gemelo:** recibe los datos de un nuevo cliente y verifica contra la tabla CLIENTES que no exista uno igual. Si existe uno igual, lanza un error.
- **Verificar\_existencia\_proveedor\_gemelo:** funcionamiento igual al del procedure verificar\_existencia\_cliente\_gemelo.
- **Verificar\_usuario:** recibe un usuario y verifica que exista en la tabla USUARIOS. Si no existe lanza un error.
- **Verificar\_password:** recibe un usuario y una contraseña, verifica que se corresponda esa contraseña con ese usuario, si no se corresponden lanza un error. Luego verifica que el estado del usuario sea "Habilitado" caso contrario, lanza un error.
- **Deshabilitar\_rol:** recibe un rol id y verifica que el estado en la tabla de ROLES sea "Habilitado", en el caso de que pase la condición hace un Update en la tabla ROLES modificando el estado a inhabilitado, luego dependiendo si el rol es "Cliente" o "Proveedor" elimina el registro en la tabla ROLES\_POR\_USUARIO y hace un update en la tabla correspondiente al rol (CLIENTES O PROVEEDORES) y modifica el estado a inhabilitado.
- **Habilitar\_rol:** recibe un rol id verifica que el rol este deshabilitado en la tabla de ROLES y modifica el estado como habilitado. Si la condición falla lanza un error.
- **Insertar\_Funciones\_Por\_Rol:** recibe un rol id y una función id y luego verifica que no exista en la tabla FUNCIONES POR ROL la función asignada al rol. Si cumple la condición inserta el rol id y la función id en la tabla FUNCIONES POR ROL, si no cumple la condición lanza un error.
- **Insertar\_Rol:** este procedure recibe rol id, verifica que no exista en la tabla ROLES y lo inserta. Si ya existe lanza un error.
- **Eliminar\_funciones\_Por\_Rol:** recibe un rol id y borra todos los registros que contengan ese rol id en la tabla FUNCIONES POR ROL.
- **Verificar\_Estado\_Rol:** recibe un rol id y verifica que el estado se encuentre con el valor de habilitado en la tabla ROLES, caso contrario lanza un error.

- **Confeccion\_De\_Oferta:** comienza verificando que el proveedor que recibe exista. Luego verifica que no exista una oferta igual ya creada. Si cumple con esto inserta en la tabla ofertas la nueva oferta.
- **Comprar\_Oferta:** recibe el código de la oferta, el precio de la oferta, el cliente que realiza la compra, la cantidad de stock de la oferta, la cantidad de ofertas compradas por el usuario y la cantidad máxima de ofertas por usuario. El procedure comienza verificando que la cantidad a comprar sea mayor que 0, caso contrario lanza un error. Luego verifica que la cantidad de ofertas a comprar no supere el stock actual de la oferta. Una vez cumplida las dos condiciones realiza el insert en la tabla COMPRAS. **(NOTA: ver trigger compras.)**
- **Oferta\_Disponible:** recibe un código de cupón y la fecha actual del sistema, verifica que la fecha se encuentre entre las fechas de publicación y la de vencimiento de la oferta para poder realizar el canje del cupón. Caso contrario lanza un error.
- **Utilizar\_cupón:** recibe un id de cupón y la fecha actual en la que se lo consume. Se realiza un update en la tabla CUPONES actualizando la fecha de consumo del mismo.
- **Cupon\_utilizado:** recibe un id de cupón y verifica que no tenga una fecha de consumo. Caso contrario lanza un error ya que el cupón fue utilizado en algún momento.