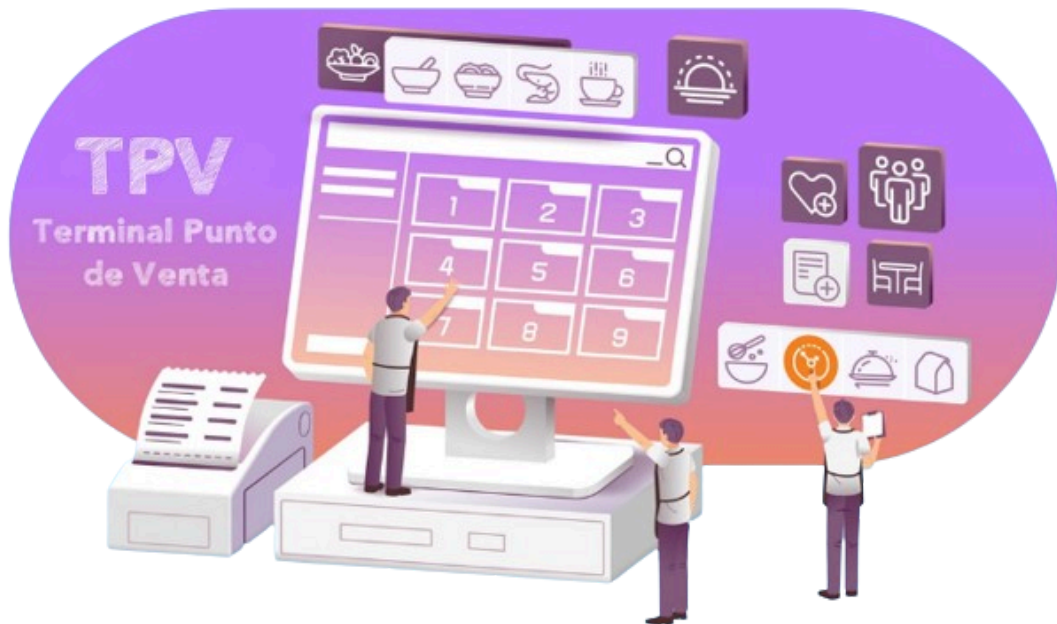




TPV - Kiosco Whimsy



Proyecto final del Ciclo Superior de Desarrollo de Aplicaciones Multiplataforma

IES Henri Matisse, 06/2024

Alumno: Marcos Palomero Pirvanescu

Tutor: Jerónimo Valero Romero

ÍNDICE

A) DESCRIPCIÓN DEL PROBLEMA

B) SOLUCIÓN PROPORCIONADA.

1. Introducción
2. Tecnologías y herramientas utilizadas
3. Base de datos
4. Estructura del proyecto
 - 4.1 Organización de la aplicación
 - 4.2 Explicación de las clases
 - 4.3 Librerías
 - 4.4 Interfaz gráfica
5. Manual de usuario

C) CONCLUSIONES

1. Dificultades encontradas
2. Ampliaciones
3. Conclusiones personales

A. DESCRIPCIÓN DEL PROBLEMA

El kiosco *Whimsy* ha solicitado desarrollar el software de un terminal de punto de venta (TPV), o *point of sale* en inglés (POS), para la gestión de su negocio.



Cuando hablamos de software de TPV, nos referimos al programa de gestión que utiliza una empresa para gestionar todo el proceso de venta, desde los pedidos hasta el inventario. Ayuda a agilizar las ventas, los pagos, los registros, el inventario, y cualquier otro aspecto del negocio.

En cuanto al hardware, se cuenta con dispositivos que permiten el cobro con tarjeta de crédito/débito y que normalmente se usan con pantalla táctil, aunque también pueden manejarse con teclado y ratón.

Las principales diferencias entre un TPV y un datáfono vendrían a ser las siguientes:



TPV	Datáfono
Sistema de pago físico y virtual	Sistema de pago físico
Almacena todos los datos de la venta	Almacena los datos del pago con tarjeta
Genera facturas e imprime tickets	Solo imprime tickets

En este caso, el programa le permitirá gestionar tareas relacionadas con la venta y el stock principalmente, aunque también incluirá campañas de publicidad o la visualización de los datos de los usuarios/empleados del negocio.

B. SOLUCIÓN PROPORCIONADA.

1. Introducción

A continuación se explican las tecnologías empleadas en el desarrollo, el esquema de la base de datos y la explicación de sus tablas, varios aspectos sobre la estructura del proyecto y el manual de usuario de la aplicación.

2. Tecnologías y herramientas utilizadas



Este proyecto se ha realizado con .NET WPF, *Windows Presentation Foundation*, un marco de interfaz de usuario que es independiente de la resolución y usa un motor de representación basado en vectores, creado para aprovechar las ventajas del hardware de gráficos moderno. WPF ofrece un conjunto completo de características de desarrollo de aplicaciones que incluyen controles, enlace de datos o *binding*, diseño, gráficos en 2D y 3D, animación, estilos, plantillas, documentos, elementos multimedia, texto y tipografía.

<https://learn.microsoft.com/es-es/dotnet/desktop/wpf/overview/?view=netdesktop-8.0>

El IDE en el que se ha desarrollado es Visual Studio 2022 y se emplean el lenguaje de marcado XAML (*Extensible Application Markup Language*) y C# (*csharp*). Para la gestión de la base de datos se ha utilizado MySQL (ver siguiente apartado)



Recursos

- Iconos: <https://yesicon.app/>
- Imágenes: <https://www.flaticon.es/>
- Logo: <https://looka.com/onboarding> , Adobe Photoshop
- Fake data para rellenar la base de datos: <https://chat.openai.com/chat>
- Proyecto en GitHub: <https://github.com/mck21/Kiosco-Whimsy/tree/master>
- Pantalla de bienvenida: <https://www.youtube.com/watch?v=m9RJls7gHYI>
- Diagramas: <https://app.diagrams.net/>

3. Base de datos

Los datos que maneja el programa son: productos, categorías, ofertas, ventas, detalles de las ventas, clientes, usuarios, datos personales, roles y permisos.

Atributos de cada una de las tablas:

- Productos :
 - ID
 - Descripción
 - Precio
 - Ubicación
 - Stock
 - Iva
 - Imagen
- Categorías/tipos de producto:
 - ID
 - Categoría
 - Imagen
- Ofertas:
 - ID
 - Descripción
 - Fecha de inicio
 - Fecha de finalización
 - Fichero
- Ventas:
 - ID
 - Fecha
 - Total
 - Mensaje
- Detalle/Línea de venta:
 - ID producto
 - ID venta

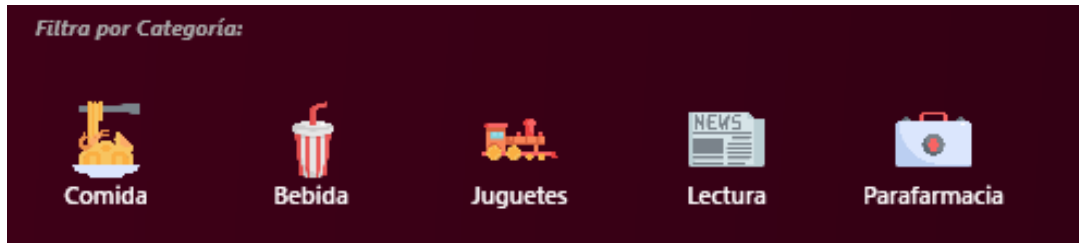
- Cantidad
 - Precio unitario
- Cliente:
 - ID
 - CIF
- Usuario:
 - ID
 - Nombre de usuario
 - Contraseña
- Persona (datos personales de usuarios y clientes):
 - ID
 - Nombre
 - Apellidos
 - Email
 - Dirección
 - Población
 - Teléfono
 - Cod postal
- Rol:
 - ID
 - Nombre del rol
- Permiso:
 - ID
 - Descripción

Cardinalidad de las entidades:

- Un producto puede tener o no oferta (0..1) y tiene una categoría (1..1).
- Un producto puede estar en 1 o muchas ventas y en una venta pueden haber sido vendidos 1 o muchos productos, la tabla de unión detalle de venta almacena los productos vendidos en una venta (N..N).
- Una venta es realizada por un usuario (trabajador) y un usuario puede realizar muchas ventas (1..N).
- Un cliente puede tener asignadas 1 o muchas ventas (1..N)
- Un cliente puede tener o no oferta/descuento (0..1) y tiene datos personales (1..1).
- Un usuario tiene un rol y datos personales (1..1).
- Un rol puede tener de 1 a muchos permisos y un permiso puede pertenecer a 1 o muchos roles (N..N).

De las tablas a la aplicación

La relación entre productos y tipo de productos se utiliza en la pantalla de la venta para poder filtrar la lista de productos seleccionables, ya que si la lista de productos es muy grande, el usuario tendría que hacer un scroll demasiado tedioso para intentar encontrar el producto deseado.



Tipo producto tiene dos atributos que aquí se distinguen a la perfección, imagen y categoría:



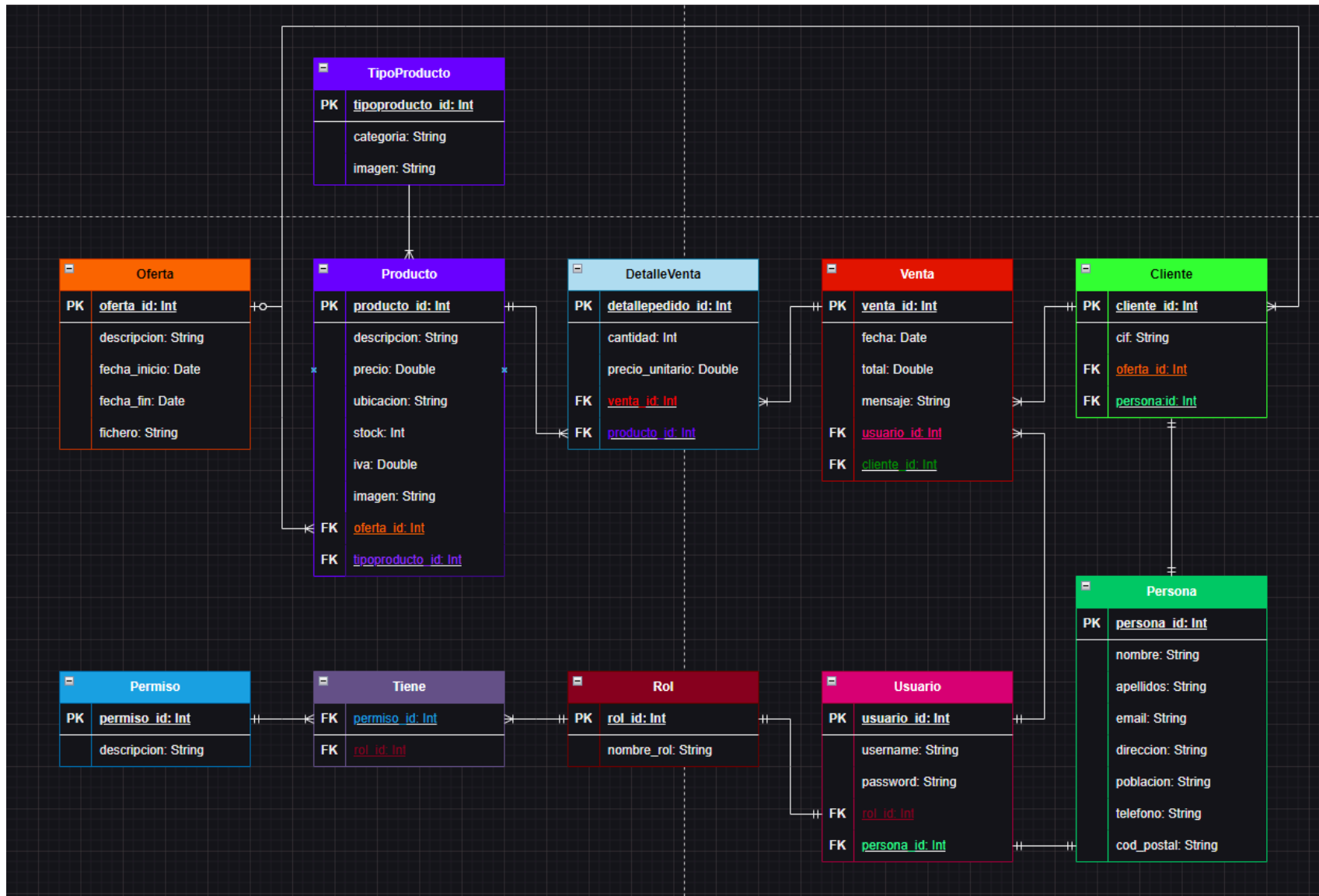
Otra tabla crucial en el programa es la de detalle de venta (cada una de las líneas del ticket), que contiene el producto seleccionado, la venta a la que pertenece y la cantidad y el precio de producto para obtener el total que costará la venta:

RESUMEN VENTA		
Producto	Precio	Cantidad
Rompecabezas	15	2 + -
Muñeca de Peluche	10	1 + -
Agua Mineral	0,8	1 + -
Bolsa de Chuches	0,5	1 + -
Vino	1,2	1 + -
Coca-Cola	1,2	3 + -

En ticket de venta:

Producto	Precio	Cantidad
Rompecabezas	15	2
Muñeca de Peluche	10	1
Agua Mineral	0,8	1
Bolsa de Chuches	0,5	1
Vino	1,2	1
Coca-Cola	1,2	3

Diagrama de la base de datos

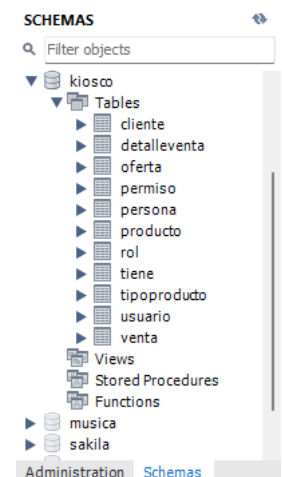
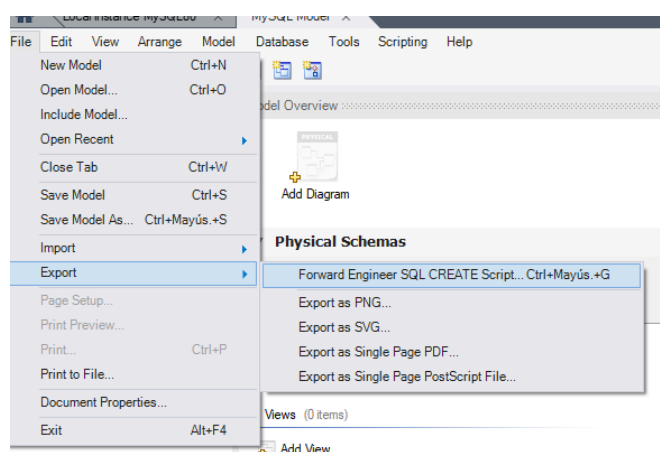
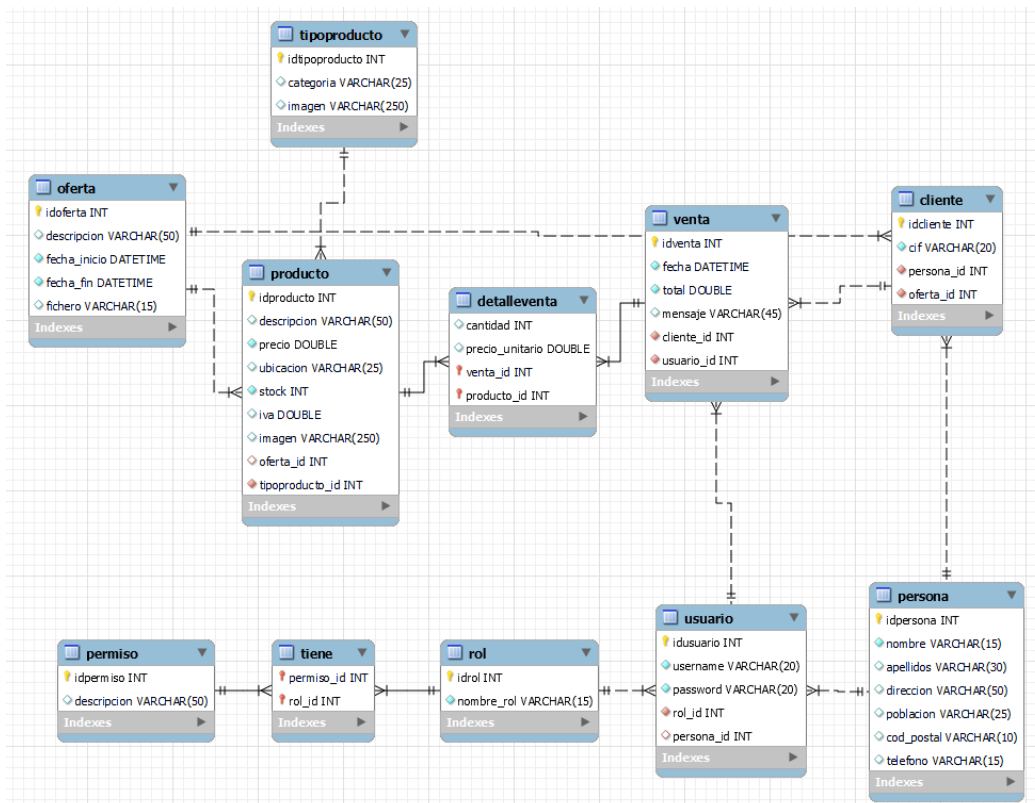


La base de datos kiosco se hospeda en el sistema gestor de bases de datos relacionales MySQL Workbench.



<https://keepcoding.io/blog/que-es-mysql-workbench/>

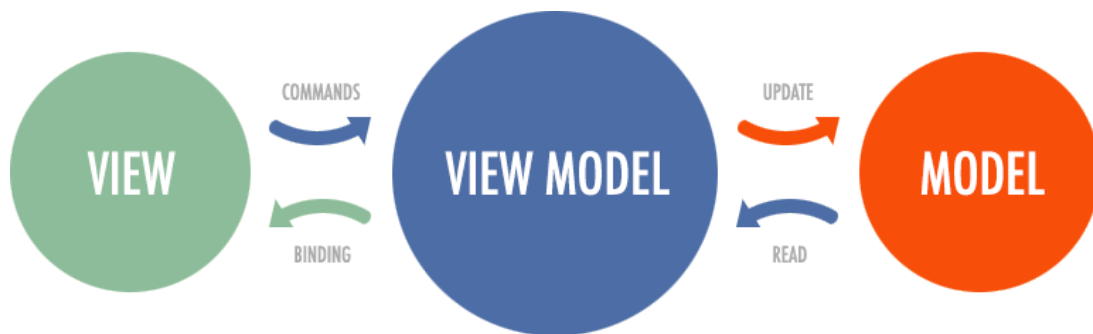
Este nos permite hacer el **paso a tablas** desde un diagrama y viceversa.



4. Estructura del proyecto

4.1 Organización de la aplicación

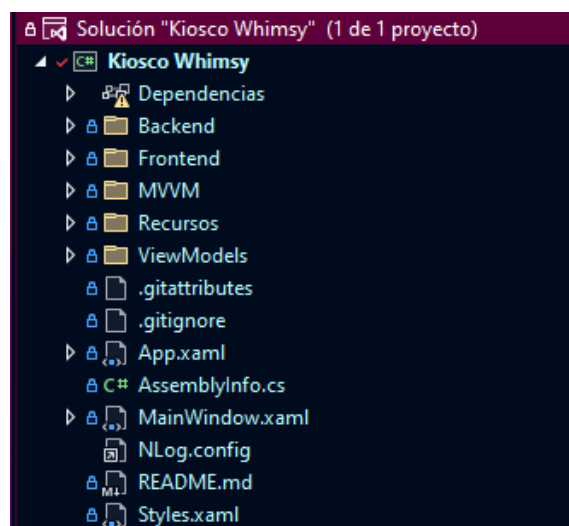
El proyecto sigue el patrón de diseño MVVM. Hay tres componentes principales en este patrón: el modelo, la vista y el modelo de vista. Cada uno de ellos tiene un propósito diferente. En el diagrama siguiente se muestran las relaciones entre los tres componentes:



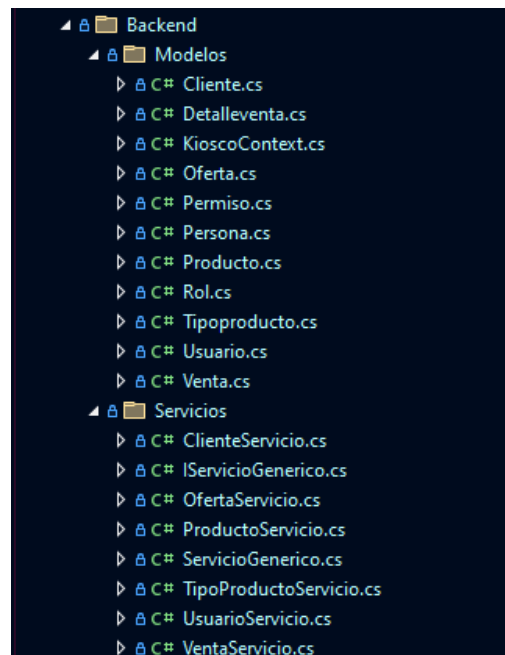
La vista conoce el modelo de vista y el modelo de vista conoce el modelo, pero el modelo no conoce el modelo de vista y el modelo de vista no conoce la vista. Por tanto, el modelo de vista aísla la vista del modelo y permite que el modelo se desarrolle independientemente de la vista.

<https://medium.com/@reyes.leomaris/aplicando-el-patr%C3%B3n-de-dise%C3%B1o-mvvm-d4156e51bbe5>

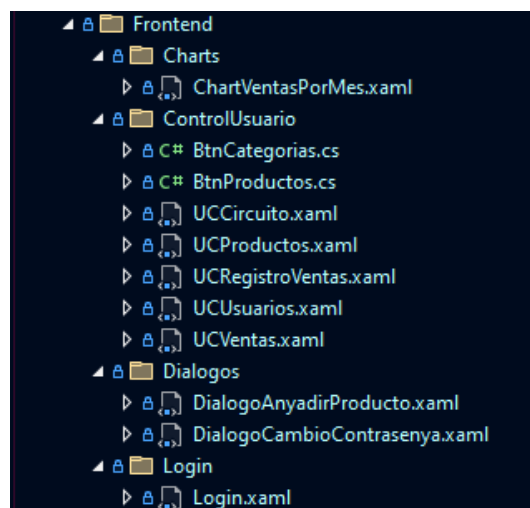
La solución del proyecto se divide en 4 carpetas principales: Backend, Frontend, Recursos y ViewModels.



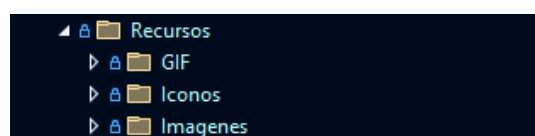
- Backend: sección donde se encuentra la lógica de negocio; los modelos de las entidades y los servicios. Es la parte que el usuario no ve.



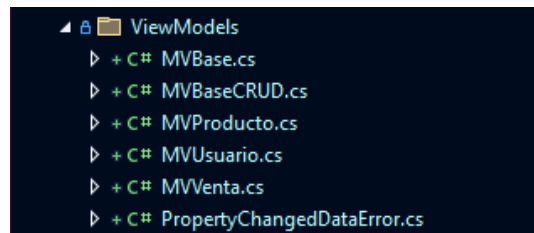
- Frontend: implica el diseño de la interfaz de usuario, aquí se incluyen los gráficos (charts), los controles de usuario (UC - User Control), diálogos y el login.



- Recursos: parte donde almacenamos los distintos iconos e imágenes.



- ViewModels: aquí se hospedan las clases modelo-vista, las cuales se encargan de realizar el enlace o *binding* entre los modelos y las vistas (UI) y manejar la lógica de presentación.



4.2 Explicación de las clases

En el directorio de los modelos encontramos los POCOs - *Plain Old Csharp Objects*, que se encuentran las clases autogenerados por *EntityFramework* (4.3).

A Producto.cs y a Venta.cs se le añaden **Annotations** para la validación y el insertado en la base de datos. Así como:

- [Required(ErrorMessage = "El campo Descripción es obligatorio")]
- [MaxLength(50)]
- [MinLength(1)]
- [Key]
- [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
- [Range(1, 200, ErrorMessage = "El valor debe estar entre 1 y 200")]

KioscoContext.cs contiene la conexión a la base de datos y el mapeo de las entidades a tablas, entre otras configuraciones.

Se añade *UseLazyLoadingProxies()* a la configuración, que es una característica de *EntityFrameworkCore* que permite que las entidades relacionadas se carguen automáticamente desde la base de datos cuando se acceden por primera vez. Con la carga perezosa se pueden cargar las entidades relacionadas cuando realmente sean necesarias, en vez de cargarlas todas de una vez.

<https://medium.com/@darshana-edirisinghe/entity-framework-performance-improvement-section-1-different-loading-mechanisms-in-entity-3e3ce2affee6#:~:text=Lazy%20loading%20is%20a%20feature,loading%20them%20all%20at%20once.>

En el directorio de los servicios todas las clases extienden del Servicio Genérico (implementa la interfaz *IServicioGenerico*) que contiene los métodos para interactuar con la base de datos:

- GetAll() → Obtiene todos los objetos de una determinada entidad.
- FindBy() → Busca elementos según la expresión o predicado pasado como parámetro.
- FindByID() → Busca un objeto por su identificador
- Add() → Inserta un objeto nuevo en la base de datos

- Delete() → Borra un objeto de la base de datos en función de su id
- Save() → Realiza un commit para que los cambios se hagan persistentes
- AddOrUpdate() → Inserta o actualiza si ya existe un objeto

En *UsuarioServicio* además, está el método que verifica que las credenciales introducidas en el login por el usuario, son correctas.

En la parte del frontend se encuentran:

Charts

- **ChartVentasPorMes:** Carga el gráfico de ventas por mes haciendo uso de una clase auxiliar VentasPorMes (representa el total de ventas en un mes), de una lista de meses y de la clase SeriesCollection para guardar la colección de datos representados. Extiende de MetroWindow (ajustes de ventana en la interfaz).

Controles de usuario

Todas las clases UC extienden de UserControl.

- **UCCircuito:** Contiene las animaciones y el canvas del circuito de la pantalla de bienvenida y hace uso del Styles.xaml
- **UCProductos:** Muestra el datagrid de productos y permite añadir, editar y eliminar productos.
- **UCRegistroVentas:** Muestra el datagrid de ventas y permite eliminar ventas y abrir el UCVentas para añadir una nueva venta. Se pueden filtrar las ventas por empleado y por fecha.
- **UCUsuarios:** Muestra el datagrid de usuarios/empleados.
- **UCVentas:** Permite filtrar la lista de productos, añadir productos a la venta, imprimir el ticket de venta y guardar la propia venta.
- **BtnProductos:** Carga las propiedades de cada uno de los botones de producto del UCVentas. Extiende de la clase Button.
- **BtnCategorias:** Carga las propiedades de cada uno de los botones de tipo de producto del UCVentas. Extiende de la clase Button.

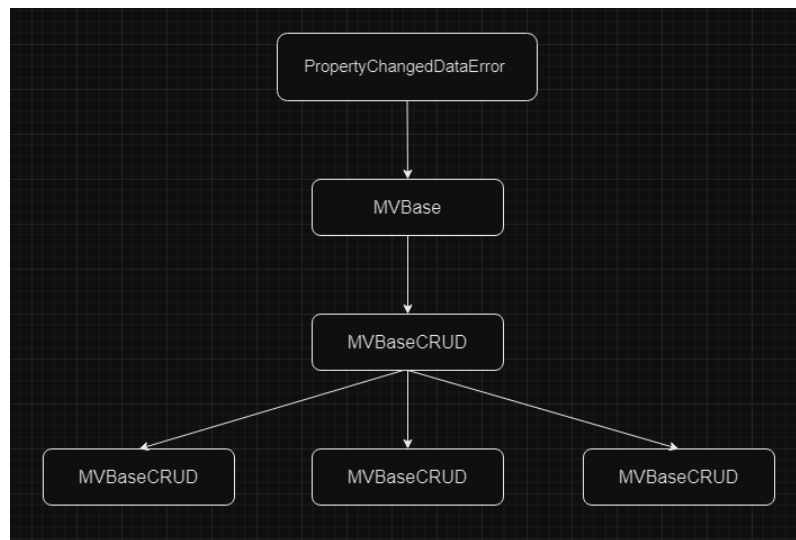
Diálogos

- **DialogoAnyadirProducto:** Valida que los campos obligatorios están rellenos por el usuario y guarda un producto en la base de datos. Este mismo diálogo se reutiliza para la edición de los productos en el UCProducto. Extiende de MetroWindow (ajustes de ventana en la interfaz).
- **DiálogoCambioContrasena:** Cambia la contraseña del usuario que ha iniciado sesión, siempre y cuando se introduzca correctamente la antigua contraseña. Extiende de MetroWindow (ajustes de ventana en la interfaz).

Login

- **Login:** Conexión con la base de datos y lógica de usuarios de la base de datos para hacer login en la aplicación .

En los ViewModels, las clases modelo-vista extienden de MVBaseCRUD > MVBase > PropertyChangedDataError.



- **PropertyChangedDataError** se encarga del manejo de errores y de la validación.
- **MVBase** maneja el evento de error y contiene métodos para la validación de errores en los formularios. Aquí se encuentra el btnGuardar que se activa o desactiva en función de si hay o no errores.
- **MVBaseCRUD** implementa los métodos para interactuar con la base de datos (Servicio Genérico), add, update (que hace un addOrUpdate) y delete. Además esta clase traza los logs de la aplicación con la librería NLog (4.3).

.....

- **MVProducto:** Enlaza la entidad Producto con la vista del diálogo y del UserControl. Contiene las listas para los comboBox de producto, la lista de ubicaciones, de imágenes, de categorías.. y de productos. Tiene un método *cargarListalmagenes()* que limpia la ruta relativa de las imágenes en caso de tener, antes de ser mostradas en el comboBox.
- **MVUsuario:** Enlaza la entidad Producto con la vista del UCUsuario. Contiene la lista de usuarios para el datagrid y guarda el usuario que ha iniciado sesión.
- **MVVenta:** Enlaza la entidad Venta con la vista del UCRegistroVentas y del UCVentas. Contiene las listas de ventas, categorías, detalle de ventas, clientes, usuarios.. además de las variables de fecha y total. Tiene los métodos *cargarRutaRelativaDelmagenes()*, que carga las rutas relativas donde se ubican las imagenes de los productos y categorías para mostrarlas en los respectivos botones de la ventana de ventas; y *calcularTotal()*, que se encarga del sumatorio del valor de los productos seleccionados para la venta. También controla la lógica de los filtros para la lista de ventas, para filtrar por empleado y por fecha.

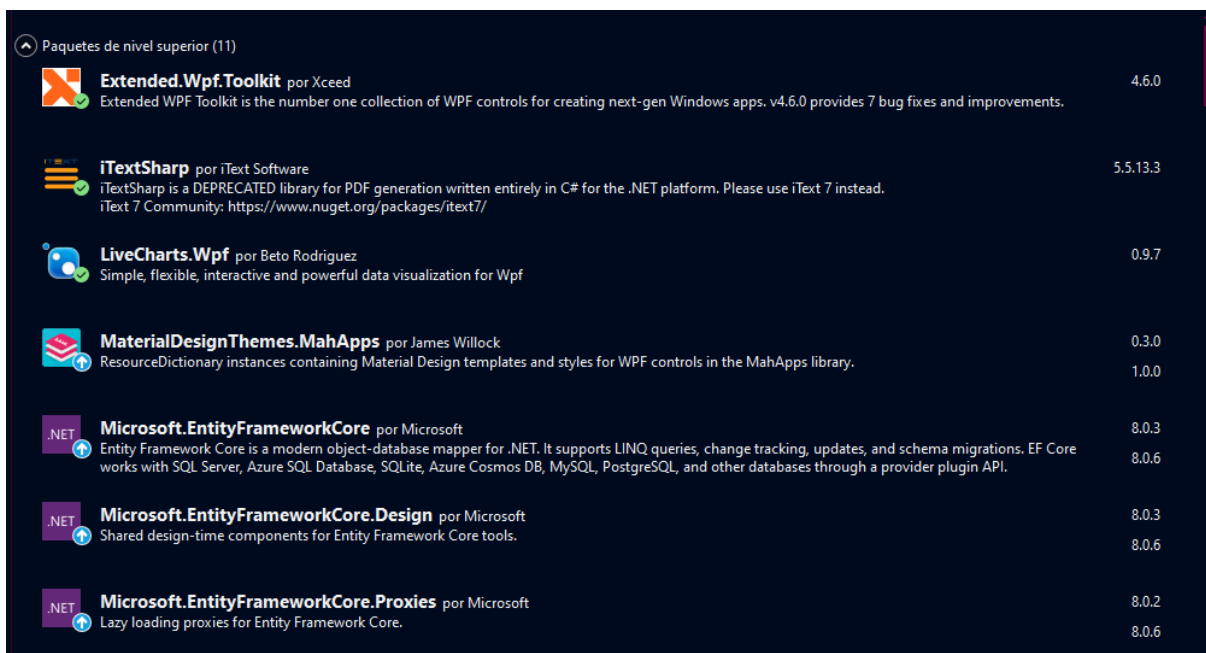
4.3 Librerías



El gestor de paquetes que usa Visual Studio es Nuget, diseñado para permitir a los desarrolladores compartir código reutilizable.

Las librerías utilizadas en este proyecto son la siguientes:

- Extended Toolkit: Controles extra para WPF. (NumericUpDown)
- ITextSharp: Para generar el ticket en PDF.
- LiveCharts: Para la visualización de gráficos en la aplicación.
- MaterialDesignThemes.Mahapps: Diseños de ventana y diferentes estilos para la interfaz gráfica.
- EntityFrameworkCore: Mapper para .NET, paso de entidades de MySQL a la solución del proyecto.
- NLog: Para registrar el historial de las operaciones en la consola.





4.4 Interfaz gráfica

La aplicación sigue un tema oscuro, con iconos e imágenes claros que resaltan con el fondo. Por lo general el color de la letra es blanco y se alterna entre las fuentes Trebuchet MS y Segoe UI (default).

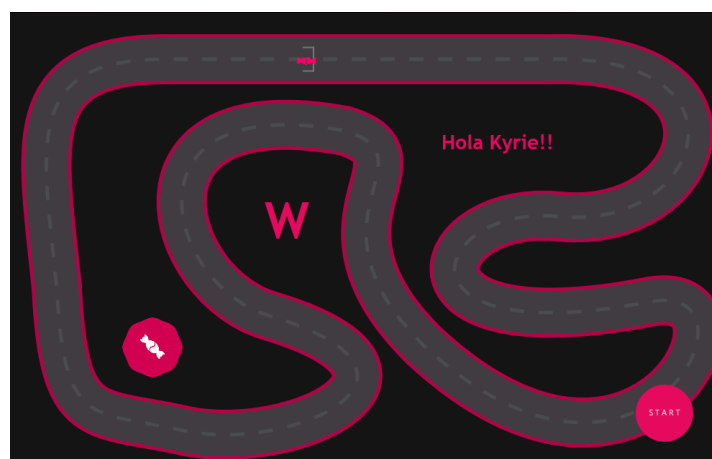
En la parte de la interfaz gráfica se han implementado las librerías Mahapps, Material design, LiveCharts o ExtendedToolkit. Las funcionalidades de cada librería se explican en el punto 4.3.

Grid, DockPanel, WrapPanel, Canvas o Border son algunas de las clases contenedoras de elementos predeterminadas de xaml de WPF utilizadas en la interfaz.

TextBlock, TextBox, PasswordBox, ComboBox, DatePicker o Button son algunos de los controles predeterminados para la introducción de datos y manejo de eventos que se han implementado.

El estilo utilizado en la mayoría de los botones es **MaterialDesignIconForegroundButton**, que le da un fondo transparente al botón y un hover sombreado circular a su alrededor.

Buscando recursos en GitHub y Youtube acordes a la tecnología utilizada para añadir y hacer más visual y atractiva la aplicación, encontré el recurso del **circuito** de la pantalla de bienvenida, de todas las funcionalidades que vi mientras investigaba, me llamó la atención añadir esta.



Las ventanas exteriores o principales, en modo minimizado tienen los bordes redondeados, mientras que los diálogos, como el diálogo que muestra el gráfico de ventas por mes o los de inserción de datos por parte del usuario, los tienen sin redondear y con un fino borde de color *Steel* para diferenciarlo del fondo.

Se utiliza en varios puntos de la interfaz un degradado sutil que le aporta un toque moderno y estilizado a la aplicación, por ejemplo en la sección de ventas:

```
<Grid.Background>
  <LinearGradientBrush StartPoint="0,0" EndPoint="1,1">
    <GradientStop Color="#3e0017" Offset="0"/>
    <GradientStop Color="#18000b" Offset="1"/>
  </LinearGradientBrush>
</Grid.Background>
```

Se ha modificado el estilo de los UC para que siguieran la línea de color, y no tuvieran el color gris predeterminado, de la siguiente manera:

```
<UserControl.Resources>
  <Style x:Key="DataGridRowStyle" TargetType="DataGridRow">
    <Setter Property="Background" Value="#690028"/>
    <Setter Property="Foreground" Value="White"/>
  </Style>
  <Style x:Key="DataGridColumnHeaderStyle" TargetType="DataGridColumnHeader">
    <Setter Property="HorizontalContentAlignment" Value="Center"/>
    <Setter Property="Background" Value="#5e0023"/>
    <Setter Property="Foreground" Value="White"/>
    <Setter Property="FontWeight" Value="Bold"/>
    <Setter Property="Padding" Value="6"/>
  </Style>
  <Style x:Key="DataGridTextElementStyle" TargetType="TextBlock">
    <Setter Property="HorizontalAlignment" Value="Center"/>
    <Setter Property="Background" Value="#690028"/>
    <Setter Property="Foreground" Value="White"/>
  </Style>
</UserControl.Resources>
```

Logo de la app (hecho por IA y retocado en Adobe Photoshop):

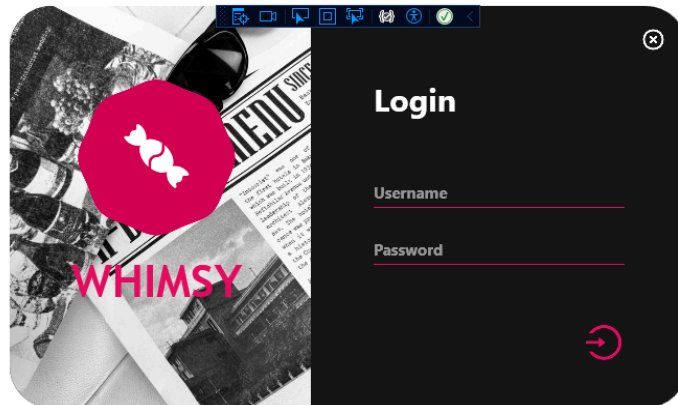


Paleta de colores empleados:

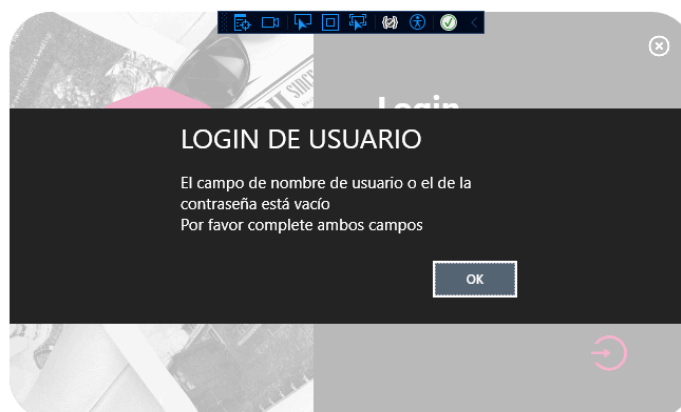
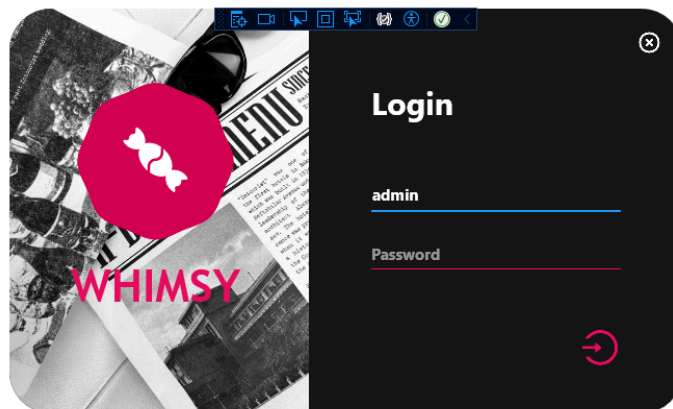


5. Manual de usuario

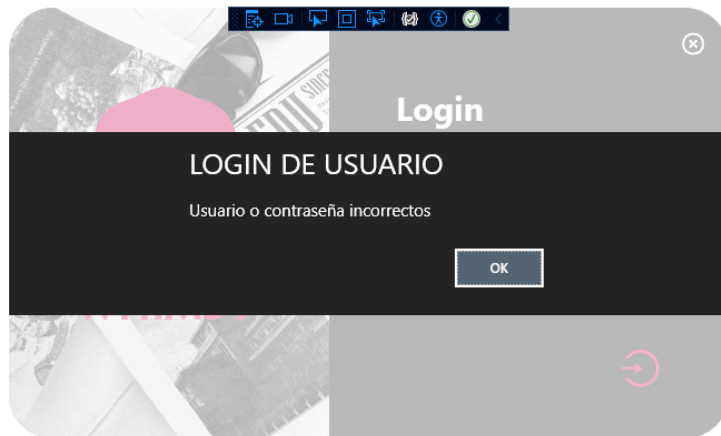
Lo primero que ve el usuario al ejecutar la aplicación es la ventana de login, donde el usuario debe iniciar sesión con su nombre de usuario y contraseña:



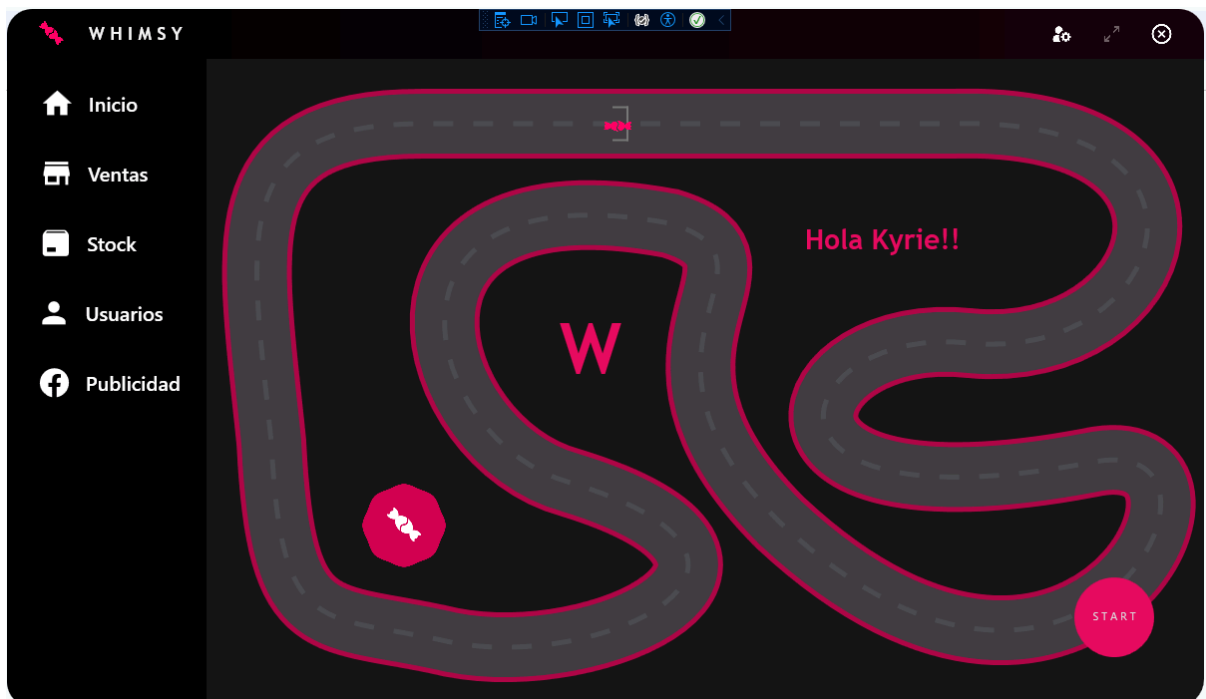
En caso de dejar vacío uno de los dos campos, salta el mensaje que pide al usuario rellenar todos los campos.



Si, por otra parte, el usuario o la contraseña no son correctos:



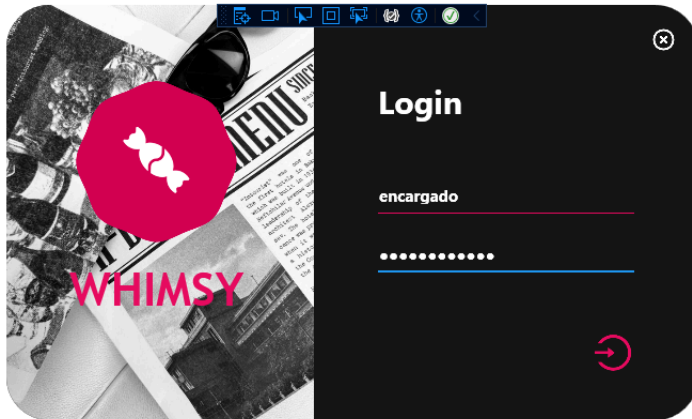
Una vez el usuario ha introducido las credenciales correctamente, aparece en la pantalla de bienvenida, donde sale un mensaje con el nombre del usuario y la animación de un caramelo en un circuito que comienza al darle al botón “START”.



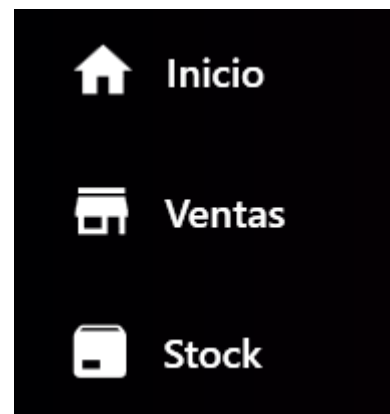
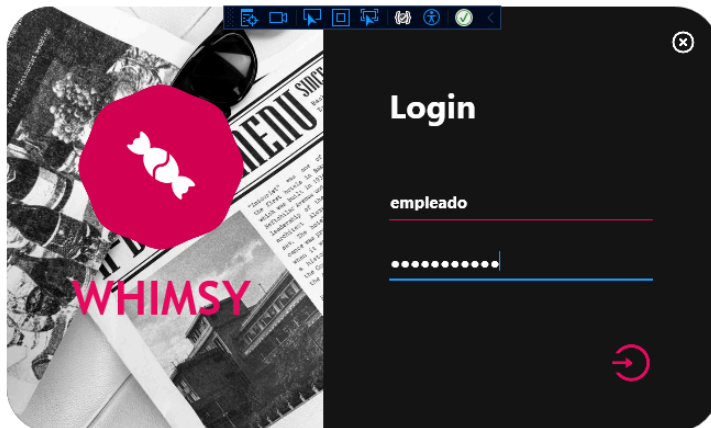
En la ventana principal se puede observar un menú de opciones en la parte lateral izquierda, con los botones para abrir: Inicio, Ventas, Stock, Usuarios y Publicidad. Y en la parte superior derecha encontramos los botones de maximizar y cerrar ventana, además del de modificación de contraseña del usuario.

Con la gestión de permisos que tiene cada usuario se maneja la visibilidad de las funcionalidades, es por eso que el usuario administrador puede verlas todas, mientras que el usuario encargado o empleado solo ve algunas:

El usuario de encargado tiene oculta la visualización de usuarios:



El usuario de empleado tiene oculta la visualización de usuarios y de hacer campañas publicitarias:



Al clicar sobre el botón de modificar contraseña se abre el diálogo para hacerlo.

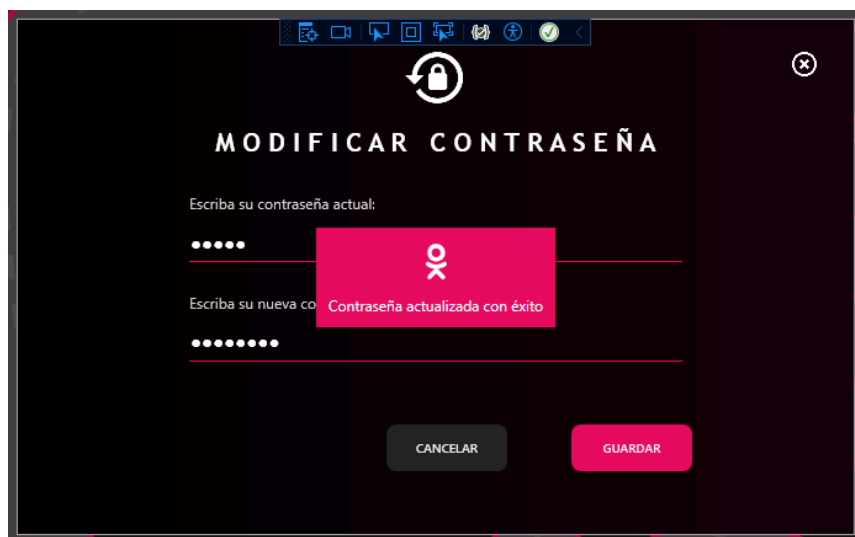


Si el usuario deja uno de los dos campos vacíos:

Si la contraseña antigua no coincide con la del usuario:



Si todo es correcto, el usuario ha modificado su contraseña:



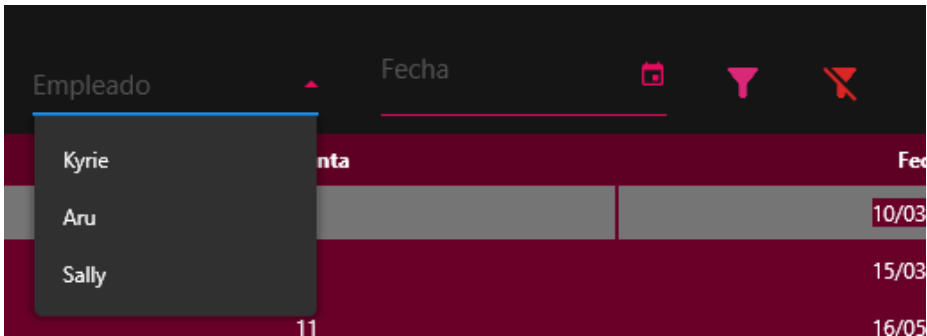
Al hacer doble clic en el botón “Ventas”, se abre la ventana del Registro de ventas. Como es común en los TPVs, se trabaja en pantalla completa cuando se procede a trabajar con las ventas, es por eso que se maximiza la ventana automáticamente.



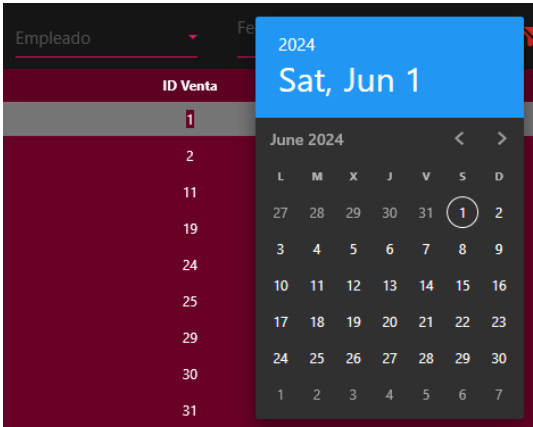
ID Venta	Fecha	Total	Empleado	CIF
1	10/03/2024	9.98 €	Kyrie	GHT789012
2	15/03/2024	29.97 €	Aru	ABC123456
11	16/05/2024	28.49 €	Kyrie	-
19	16/05/2024	58.49 €	Kyrie	-
24	21/05/2024	27.00 €	Kyrie	MNO901234
25	21/05/2024	48.69 €	Kyrie	JKL345678
29	23/05/2024	4.00 €	Kyrie	-
30	23/05/2024	12.50 €	Aru	-
31	01/06/2024	9.00 €	Sally	-
32	23/05/2024	22.39 €	Kyrie	-
33	27/05/2024	40.19 €	Kyrie	-
34	27/05/2024	66.50 €	Kyrie	-
35	27/05/2024	5.00 €	Kyrie	MNO901234
36	27/05/2024	50.48 €	Kyrie	JKL345678
37	01/06/2024	48.69 €	Kyrie	-

En esta ventana se muestran todas las ventas realizadas en una tabla. Hay una sección de controles que filtran la lista de ventas:

Se puede filtrar por el empleado que ha realizado la venta:



Por la fecha en la que se ha realizado la venta:



O bien por ambas.

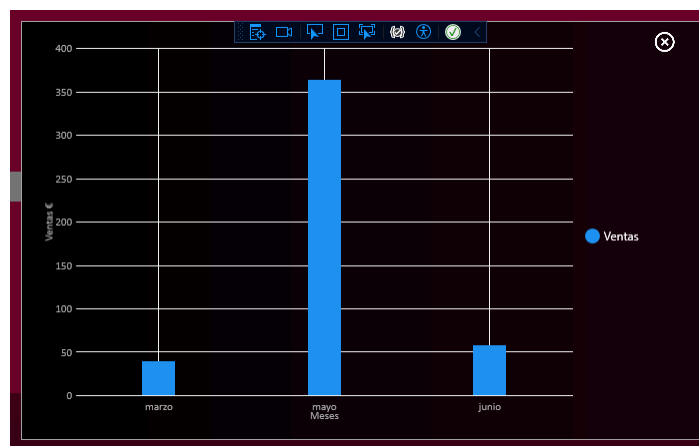
Para activar los filtros hay que pulsar sobre el icono marcado en azul una vez se haya seleccionado algún criterio por el cual filtrar. Y para desactivarlos, se debe clicar sobre el marcado en rojo.



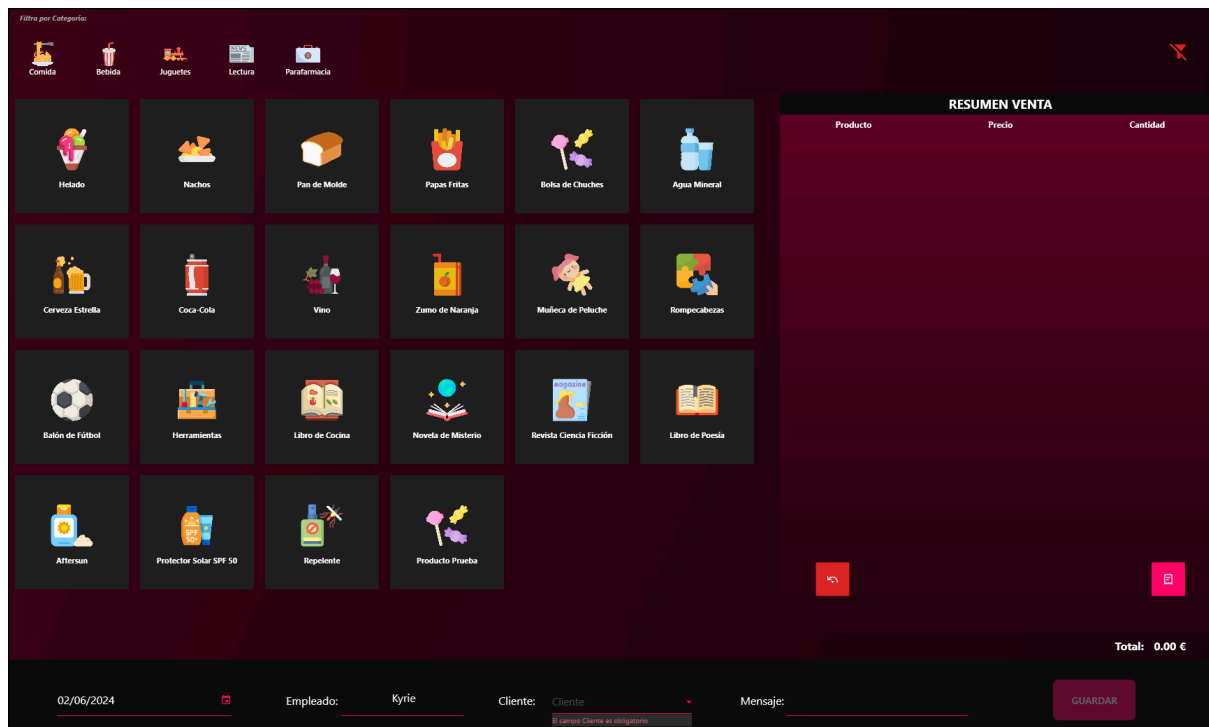
En la parte superior derecha de este panel de Registro de ventas, existen dos botones:



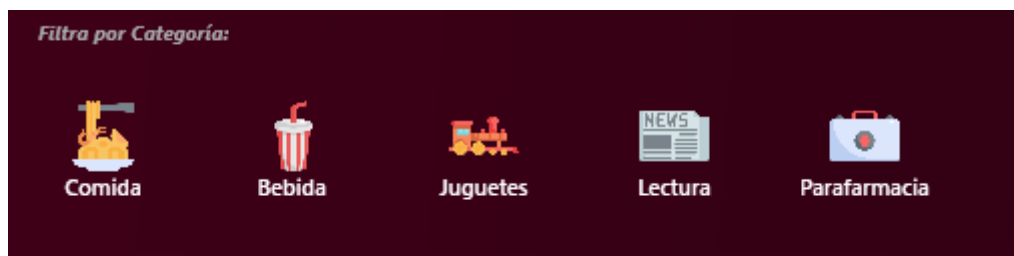
El de la derecha abre una ventana con el gráfico de ventas por mes:



Y el de la izquierda abre el panel para insertar una nueva venta. Esta es la pantalla más importante de la aplicación.



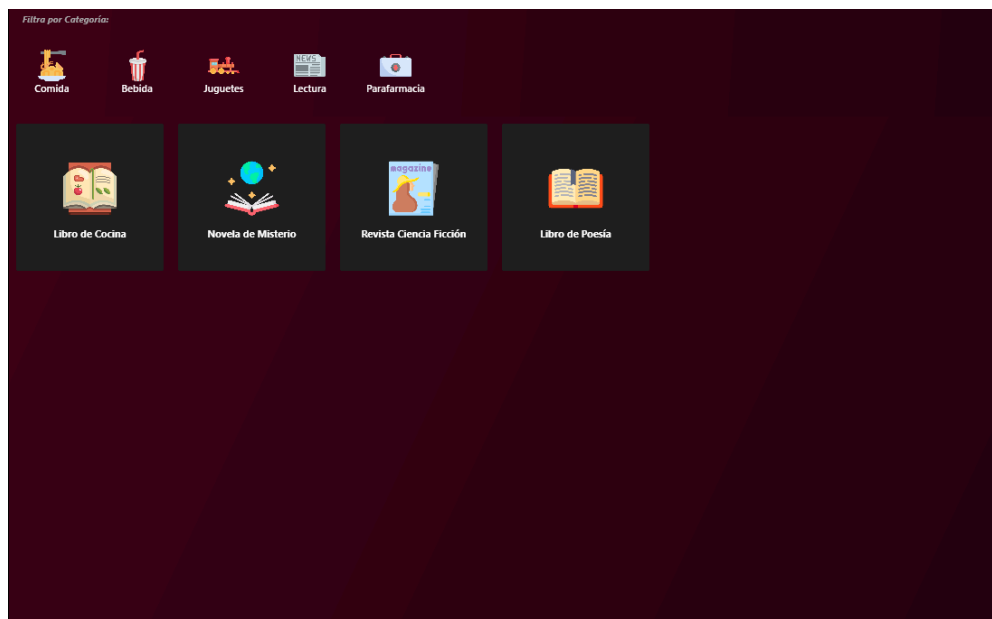
En la parte superior, encontramos los botones de categorías. Estos tienen la función de filtrar la lista de productos seleccionables de justo debajo, en función de su categoría.



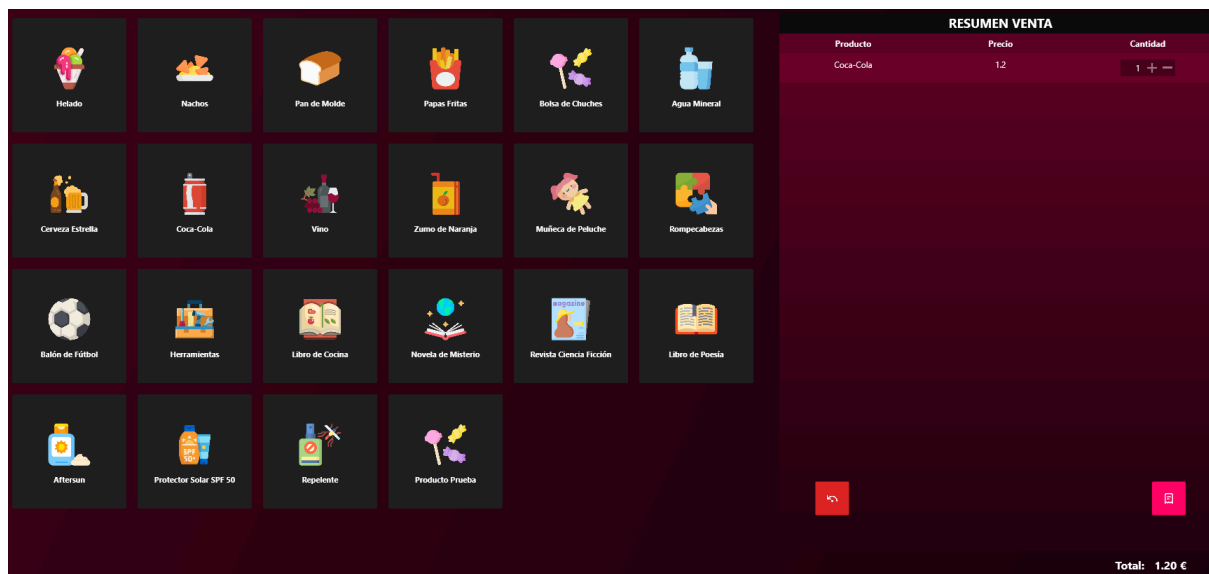
A la derecha de esta lista de categorías, se encuentra el botón para limpiar los filtros:



Al filtrar, por ejemplo, por lectura:



Al seleccionar un producto de la lista, si está disponible, se añadirá al resumen de venta.



El resumen de venta, contiene el precio unitario del producto seleccionado y la cantidad que desee el usuario introducir, manejado por un control + y -. Además si el usuario clicla otra vez sobre el mismo producto, la cantidad también aumentará.

Producto	Precio	Cantidad
Coca-Cola	1.2	3 + -

En la parte inferior derecha se encuentra el valor total de la venta, que se actualiza en función del precio unitario de los productos seleccionados y su cantidad.

Total: 3.60 €

Al sobrepasar el umbral de producto disponible (establecido en 5), salta una alerta para reponer el stock del producto seleccionado:



Si el producto no se encuentra disponible, no se podrá añadir a la venta y saltará el siguiente mensaje, además de deshabilitarse el control de + y -:



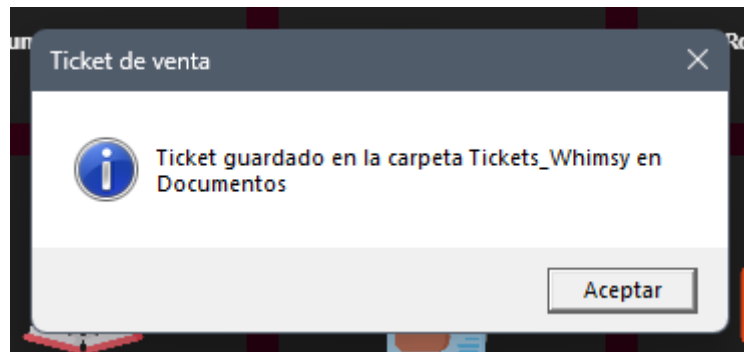
En el propio resumen de la venta se encuentra el botón “deshacer”, que elimina el último producto seleccionado de la lista y actualiza el total.



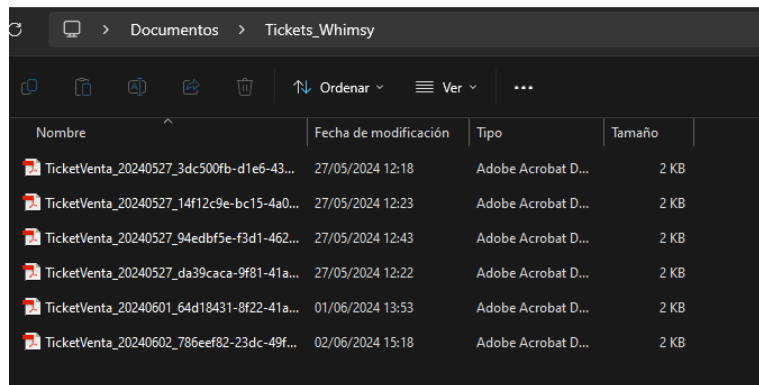
Y a su derecha el botón para guardar en PDF el ticket de la venta.



Se guarda en una carpeta llamada Tickets_Whimsy en Documentos.



Con la fecha en la que se ha realizado la venta y un identificador único para no sobrescribir tickets anteriores.



El ticket generado guarda la fecha, el total y el resumen de la venta, así como el título del negocio, un mensaje con el empleado que ha atendido la venta y otro de agradecimiento.

WHIMSY

Fecha:
2024-06-02

TOTAL:
22,59

Resumen de venta:

Producto	Precio	Cantidad
Coca-Cola	1,2	3
Libro de Poesía	8,99	1
Muñeca de Peluche	10	1

Le ha atendido Kyrie

¡GRACIAS POR COMPRAR EN WHIMSY!

Antes de guardar la venta, el usuario debe rellenar la fecha de la venta y el cliente al que se le ha realizado la misma. El campo empleado es fijo y es el usuario que ha iniciado la sesión. El campo mensaje es opcional.

02/06/2024  Empleado: Kyrie Cliente: Cliente  Mensaje: GUARDAR

El campo Cliente es obligatorio

Al guardar la venta, se recarga toda la pantalla y se puede volver a realizar una nueva venta.

Clicando en el botón de Stock:



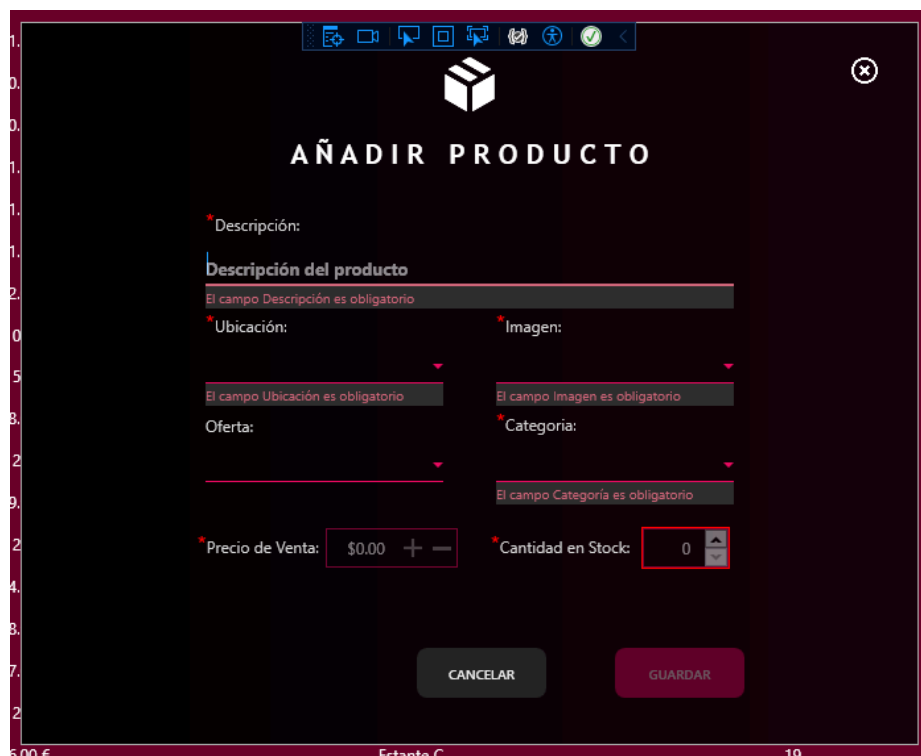
Se pueden visualizar los productos y todos sus datos (precio, ubicación, categoría...):

GESTIÓN DE STOCK 						
Producto	Precio	Ubicación	Stock	Oferta	Categoría	
Helado	1.50 €	Nevera	50	Sin Descuento	Comida	
Nachos	0.75 €	Estante A	99	Sin Descuento	Comida	
Pan de Molde	1.00 €	Estante A	29	Sin Descuento	Comida	
Papas Fritas	1.20 €	Estante A	39	Sin Descuento	Comida	
Bolsa de Chuches	0.50 €	Estante A	190	Sin Descuento	Comida	
Agua Mineral	0.80 €	Nevera	199	Sin Descuento	Bebida	
Cerveza Estrella	1.50 €	Nevera	79	Sin Descuento	Bebida	
Coca-Cola	1.20 €	Nevera	91	Sin Descuento	Bebida	
Vino	1.20 €	Nevera	16	Sin Descuento	Bebida	
Zumo de Naranja	2.00 €	Nevera	27	Sin Descuento	Bebida	
Mufeca de Peluche	10.00 €	Almacén	42	Descuento del 10%	Juguetes	
Rompecabezas	15.00 €	Almacén	49	Sin Descuento	Juguetes	
Balón de Fútbol	8.00 €	Almacén	20	Descuento del 15%	Juguetes	
Herramientas	12.00 €	Almacén	24	Sin Descuento	Juguetes	
Libro de Cocina	9.99 €	Estante B	27	Sin Descuento	Lectura	
Novela de Misterio	12.50 €	Estante B	10	Sin Descuento	Lectura	
Revista Ciencia Ficción	4.50 €	Estante B	31	Sin Descuento	Lectura	
Libro de Poesía	8.99 €	Estante B	5	Descuento del 20%	Lectura	
Altarsun	7.00 €	Estante C	39	Sin Descuento	Parafarmacia	
Protector Solar SPF 50	12.00 €	Estante C	21	Sin Descuento	Parafarmacia	
Repelente	6.00 €	Estante C	19	Sin Descuento	Parafarmacia	
Producto Prueba	4.00 €	Nevera	0	Sin Descuento	Bebida	

Para gestionar el stock se permite al usuario hacer la inserción de nuevos productos, la edición de alguno ya existente o el borrado. Para añadir un producto el usuario debe clicar sobre este botón de arriba a la derecha:



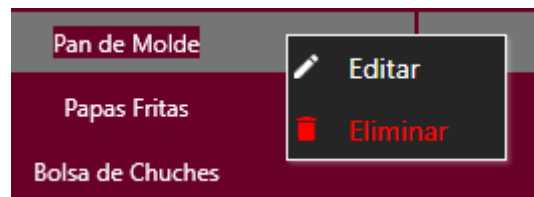
Y se abre el diálogo para introducir los datos del nuevo producto, hasta que el usuario no rellene todos los campos obligatorios, el botón de guardar producto estará deshabilitado:

Una ventana modal con el título 'AÑADIR PRODUCTO' y un icono de caja. Contiene varios campos de entrada con asteriscos que indican que son obligatorios: 'Descripción' (con un mensaje de error 'El campo Descripción es obligatorio'), 'Ubicación' (con un mensaje de error 'El campo Ubicación es obligatorio'), 'Imagen' (con un mensaje de error 'El campo Imagen es obligatorio'), 'Categoría' (con un mensaje de error 'El campo Categoría es obligatorio'), 'Precio de Venta' (con un valor de '\$0.00' y botones de incremento/decremento) y 'Cantidad en Stock' (con un valor de '0' y botones de incremento/decremento). Hay botones 'CANCELAR' y 'GUARDAR' al final. El botón 'GUARDAR' parece estar deshabilitado.

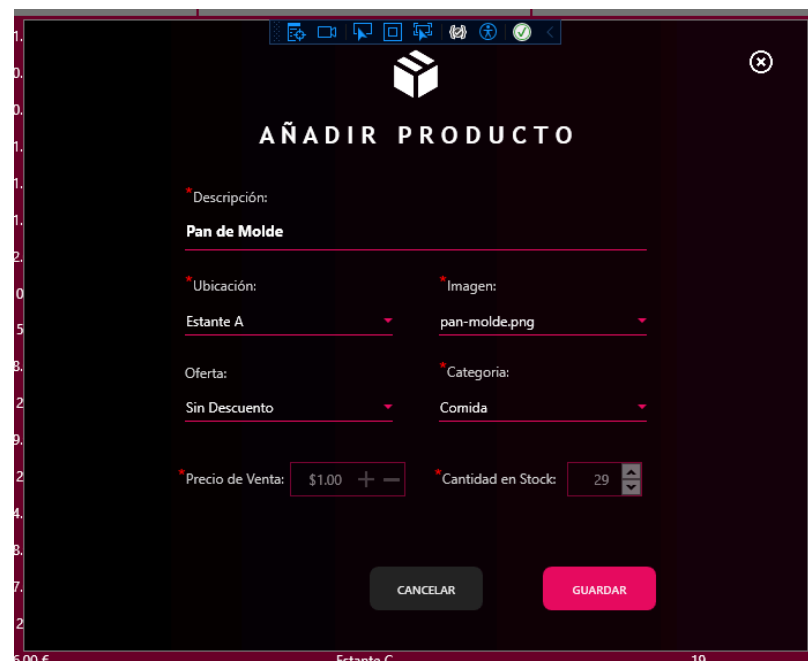
Al guardar un producto correctamente, sale un mensaje para informar al usuario.

Una ventana modal de éxito con el título 'prueba'. Muestra los datos guardados: 'Ubicación: Estante A', 'Imagen: [icono de caja]', 'Oferta: Descuento del 10%', y 'Categoría: Juguetes'. En el centro hay un mensaje 'Producto añadido a la base de datos' con un icono de una caja y una marca de verificación.

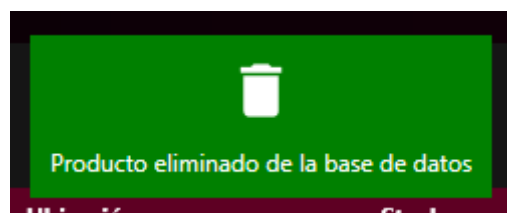
Para la edición y el borrado de un producto, se debe hacer clic derecho sobre el producto y saldrán ambas opciones:



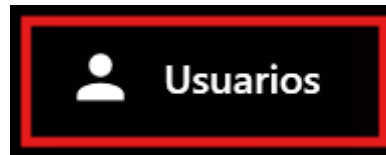
Al seleccionar Editar, el usuario puede modificar los datos del producto en el diálogo que se abre.

A dialog box titled 'AÑADIR PRODUCTO' (Add Product) is shown. It contains several input fields: 'Descripción:' with the value 'Pan de Molde'; 'Ubicación:' with a dropdown menu showing 'Estante A'; 'Imagen:' with a dropdown menu showing 'pan-molde.png'; 'Oferta:' with a dropdown menu showing 'Sin Descuento'; 'Categoría:' with a dropdown menu showing 'Comida'; 'Precio de Venta:' with a text input showing '\$1.00' and plus/minus buttons; and 'Cantidad en Stock:' with a numeric input showing '29' and up/down arrows. At the bottom, there are two buttons: 'CANCELAR' and 'GUARDAR'.

Al seleccionar eliminar, el producto se borra de la base de datos y sale el siguiente mensaje:



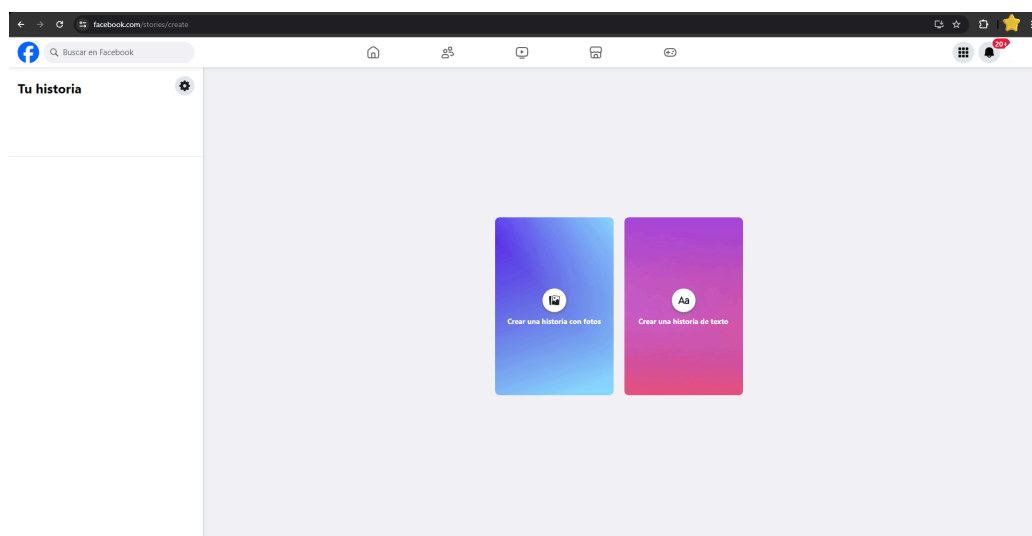
Para abrir la sección de “Usuarios”, se ha de hacer doble clic en el botón:



Se pueden visualizar los datos de los usuarios de los empleados: usuarios, contraseñas y datos personales de cada uno (esta sección solo es visible para el usuario administrador (gerente)).

GESTIÓN DE USUARIOS			
Nombre de usuario y contraseña	Nombre	Teléfono	Dirección
Gerente: admin - admin	Kyrie García	123456789	Calle Mayor 123 (Valencia)
Encargado: encargado - encargado123	Aru López	987654321	Avenida Libertad 456 (Valencia)
Empleado: empleado - empleado123	Sally Martínez	654987321	Plaza España 789 (Valencia)

La última sección es un enlace que abre en el navegador predeterminado del usuario la página oficial de Facebook. Más concretamente a publicar un *storie* para hacer cualquier tipo de campaña publicitaria.



C. CONCLUSIONES

1. Dificultades encontradas

Algunas de las dificultades que he encontrado al realizar este proyecto se encontraban en la pantalla de venta, la forma de cargar la lista de botones de producto ha sido cambiada, en lugar de usar un *ItemsControl*, se recorre una lista de botones de clase *BtnProducto.cs*; o la carga dinámica de las imágenes entraba en conflicto con otras partes de la aplicación.

Una de las partes que más me ha costado ha sido el manejo de la cantidad en el resumen de venta, ya que el control *NumericUpDown* me daba bastantes problemas.

Por otra parte he tenido que lidiar con problemas de versiones, he cambiado la librería de Reports a *ITextSharp*, que es un generador de documentos PDF.

2. Ampliaciones

Además de las propuestas en el enunciado, como la de añadir funcionalidades para el control de la caducidad de los productos o de si un producto es de temporada, algunas de las ampliaciones a realizar en este proyecto son:

- La inserción, edición y eliminación de usuarios.
- Control del usuario en la gestión de permisos.
- Visualización y gestión de clientes.
- Añadir una imagen que se sobrepone en el botón de producto si este tiene descuento/oferta.
- Implementación de filtros en todas las tablas y mejorar su visualización con alguna animación para plegarlos y desplegarlos.



3. Conclusiones personales

En lo personal, me ha gustado mucho realizar este proyecto, ya que es el más completo que hemos realizado en todo el ciclo y donde se ve la utilidad de todo lo aprendido estos dos años.

Al principio sí considero que estaba un poco agobiado, sobre todo porque contaba con que empezaba más tarde que el resto, por un viaje. Pero dedicando varias tardes a la semana durante estos dos/tres últimos meses, he podido hacerlo sin demasiados bloqueos.

Quiero agradecer a Jero, mi tutor de proyecto, por ayudarme a resolver todas las dudas que han surgido durante el desarrollo del mismo, y en general, a todos los profesores del ciclo de DAM por enseñarme el mundo del desarrollo de software como lo han hecho. Recomendaría este instituto y este ciclo a cualquiera que me preguntara por mi formación.