

A dark blue vertical bar on the left side of the page. A blue arrow points to the right from the bar, containing the date.

Julio de 2023

SISTEMA DE VENTAS WEB ASP.NET - MVC

Programación III
Trabajo Obligatorio

Several thin, curved lines in shades of blue and grey originate from the bottom left and sweep upwards and to the right.

Santiago Eguzquiza
Lautaro De León



Resumen

El proyecto en cuestión se ha enfocado en la creación de un sistema de ventas web altamente funcional y versátil. Su principal objetivo es proporcionar a los usuarios una herramienta integral que les permita gestionar de manera eficiente y efectiva sus productos, así como llevar a cabo transacciones comerciales. Este sistema abarca diversas funcionalidades clave, como la gestión de inventario, la realización de ventas y compras, y la generación automática de facturas precisas y detalladas.

Una de las características destacadas del sistema es su capacidad para brindar a los usuarios un control total sobre su historial de clientes. A través de una interfaz intuitiva y fácil de usar, los usuarios pueden mantener un registro completo de todos sus clientes, incluyendo información relevante como datos de contacto, historial de compras y preferencias individuales. Esto permite un mejor conocimiento del cliente y facilita la personalización de la experiencia de compra, lo que a su vez puede ayudar a aumentar la fidelidad del cliente y mejorar la satisfacción general.

Además, el sistema proporciona una función de consulta de cotizaciones diarias, lo que permite a los usuarios estar al tanto de los precios actualizados y las ofertas especiales en tiempo real. Esto no solo brinda una visión clara de los precios y las promociones vigentes, sino que también facilita la toma de decisiones informadas por parte de los usuarios al momento de realizar compras o establecer precios para sus propios productos.

Otra característica importante del sistema es su capacidad para gestionar perfiles de usuarios. A través de un sistema de roles cuidadosamente diseñado, el programa permite al administrador del sistema tener acceso total a todas las funcionalidades y opciones disponibles, garantizando así un control y supervisión efectivos. Al mismo tiempo, se pueden crear perfiles de usuarios con permisos restringidos para limitar su acceso a ciertas áreas o funciones del sistema, brindando así una mayor seguridad y protección de la información sensible.

Índice

Contenido

Resumen	1
Índice.....	2
Introducción	3
Modelo Vista Controlador	3
- Modelo	3
- Vista	3
- Controlador	3
- ¿Por qué MVC?	4
Interfaz de programación de aplicaciones (API).....	5
Entity Framework	6
Nuestro sistema de ventas.....	7
Base de datos	7
Identity	9
Diseño de Entidades	10
Acceso de los usuarios	12
Navegación	13
Validación de cédula.....	15
Facturación	16
Cotizaciones.....	18
Conclusión.....	20
Referencias.....	21

Introducción

Modelo Vista Controlador

El patrón de diseño Modelo Vista Controlador (MVC) es un enfoque arquitectónico utilizado en el desarrollo de aplicaciones de software. Se utiliza para separar y organizar la lógica de la aplicación en tres componentes principales: el modelo, la vista y el controlador.

- Modelo

El modelo representa los datos y la lógica de negocio de la aplicación. Es el responsable de la gestión de los datos, su validación y su interacción con la base de datos. El modelo no sabe nada acerca de la interfaz de usuario o de cómo se muestran los datos.

Los datos los tendremos habitualmente en una base de datos, por lo que en los modelos tendremos todas las funciones que accederán a las tablas y harán los correspondientes selects, updates, inserts, etc.

- Vista

La vista es la capa de presentación de la aplicación. Se encarga de mostrar los datos al usuario y de interactuar con él. La vista se preocupa principalmente de la apariencia y la forma en que los datos son presentados. No contiene lógica de negocio compleja, su función principal es mostrar la información al usuario y recibir sus interacciones.

Aunque la vista use los datos, no se realiza un acceso directo a éstos. Las vistas solicitarán los datos a los modelos y ella generará la salida, tal como nuestra aplicación requiera.

- Controlador

El controlador actúa como intermediario entre el modelo y la vista. Recibe las interacciones del usuario desde la vista y las maneja, actualizando el modelo y/o la vista según sea necesario. También puede realizar tareas como la validación de datos o la gestión de eventos. El controlador es el encargado de coordinar la interacción entre el modelo y la vista.¹

¹ ¿Qué es MVC? Desarrollo web - <https://desarrolloweb.com/articulos/que-es-mvc.html>

MVC se utiliza en muchos frameworks y tecnologías de desarrollo, como Ruby on Rails, ASP.NET, Spring y Laravel, entre otros. Cada uno de estos frameworks tiene su propia implementación de MVC, pero siguen los mismos principios básicos de separación de preocupaciones. A su vez este diseño de arquitectura surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado.

- ¿Por qué MVC?

Por 3 simples palabras antes mencionadas, separación de responsabilidades, la cual facilita el mantenimiento, la reutilización de código y la escalabilidad del sistema, entre otras cosas.

Al dividir la aplicación en componentes distintos y bien definidos, es más fácil desarrollar y mantener el código. Cada componente tiene su propia responsabilidad y puede ser modificado o mejorado sin afectar a los demás.

Permite mucha reutilización de código. Por ejemplo, el modelo puede ser reutilizado en diferentes vistas o el controlador puede ser utilizado con diferentes modelos y vistas.

Es más fácil realizar pruebas unitarias y de integración en cada componente por separado. Por ejemplo, se pueden probar el modelo y el controlador sin necesidad de interactuar con la vista, lo que simplifica las pruebas y permite identificar y solucionar errores de forma más eficiente.²

² ¿Qué es MVC? Desarrollo web - <https://desarrolloweb.com/articulos/que-es-mvc.html>

Interfaz de programación de aplicaciones (API)

Una API, o Interfaz de Programación de Aplicaciones, es un conjunto de definiciones y protocolos que se utilizan para diseñar y facilitar la integración del software de distintas aplicaciones. En términos más simples, una API actúa como un intermediario que permite la comunicación entre diferentes sistemas o componentes de software.

Las APIs definen un conjunto de reglas y especificaciones que determinan cómo las aplicaciones deben interactuar entre sí. Proporcionan un conjunto de funciones, métodos y protocolos estándar que permiten a los desarrolladores acceder y utilizar las funcionalidades de una aplicación o servicio específico de manera controlada y segura.

Al utilizar una, los desarrolladores pueden aprovechar la funcionalidad existente de una aplicación o servicio sin necesidad de comprender los detalles internos de cómo funciona internamente. Esto simplifica el proceso de desarrollo de software al permitir que diferentes aplicaciones se comuniquen y compartan datos de manera eficiente.

Además, las APIs fomentan la reutilización de código y la colaboración entre desarrolladores y organizaciones. Al exponer ciertas funcionalidades a través de una API, los proveedores de servicios permiten que otros desarrolladores construyan sobre esa base, creando nuevas aplicaciones o servicios que se integran de forma transparente con la funcionalidad existente.

Las API son un medio simplificado para conectar su propia infraestructura a través del desarrollo de aplicaciones nativas de la nube, pero también le permiten compartir sus datos con clientes y otros usuarios externos. Las API públicas aportan un valor comercial único porque simplifican y amplían sus conexiones con los partners y, además, pueden rentabilizar sus datos (un ejemplo conocido es la API de Google Maps)³.

³ ¿Qué es una API y cómo funciona? - <https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>

Entity Framework

Entity Framework es una herramienta ORM (Mapeo Objeto-Relacional) que simplifica el manejo de bases de datos relacionales en el desarrollo de aplicaciones. Permite trabajar a un nivel alto de abstracción, evitando la necesidad de lidiar directamente con consultas SQL y las complejidades de las relaciones de tablas.

Con Entity Framework, las conexiones con la base de datos se realizan a través de métodos de acceso a entidades, en lugar de escribir consultas SQL. El framework se encarga de realizar las consultas de manera transparente según el modelo de datos definido en el código fuente de la aplicación.

Las principales capacidades de Entity Framework son:

- Abstracción del sistema gestor de base de datos: Es independiente del sistema de base de datos subyacente, como PostgreSQL, SQL Server o MariaDB. Esto permite cambiar de sistema de base de datos sin afectar el código de la aplicación.
- Abstracción del sistema relacional de datos: Entity Framework elimina la necesidad de trabajar directamente con consultas SQL y permite acceder a los datos relacionados mediante colecciones de objetos en código C#. Facilita el manejo de datos a un nivel alto de abstracción.

Entity Framework funciona principalmente por convenciones, lo que significa que no requiere configuraciones adicionales si se sigue el enfoque esperado por el framework. Sin embargo, se puede añadir código en las clases C# para personalizar el comportamiento de Entity Framework en caso de no cumplir con las convenciones predeterminadas.⁴

⁴ Entity Framework - <https://desarrolloweb.com/home/entity-framework>

Nuestro sistema de ventas

Base de datos

Para iniciar el desarrollo del sistema de ventas web, fue necesario determinar el enfoque a utilizar para la generación de la base de datos. Dado que estábamos trabajando con ASP.NET en el modelo MVC y utilizando Entity Framework, conocíamos que las opciones eran limitadas, por lo cual decidimos adoptar el enfoque "Database First". Esta elección se basó en la preferencia del equipo de trabajo de generar inicialmente la base de datos y posteriormente crear las clases de objetos correspondientes.

Habiendo generado las entidades de datos en el servidor de SQL Server con sus respectivas relaciones entre cada entidad logramos obtener el siguiente diagrama (Imagen 1):

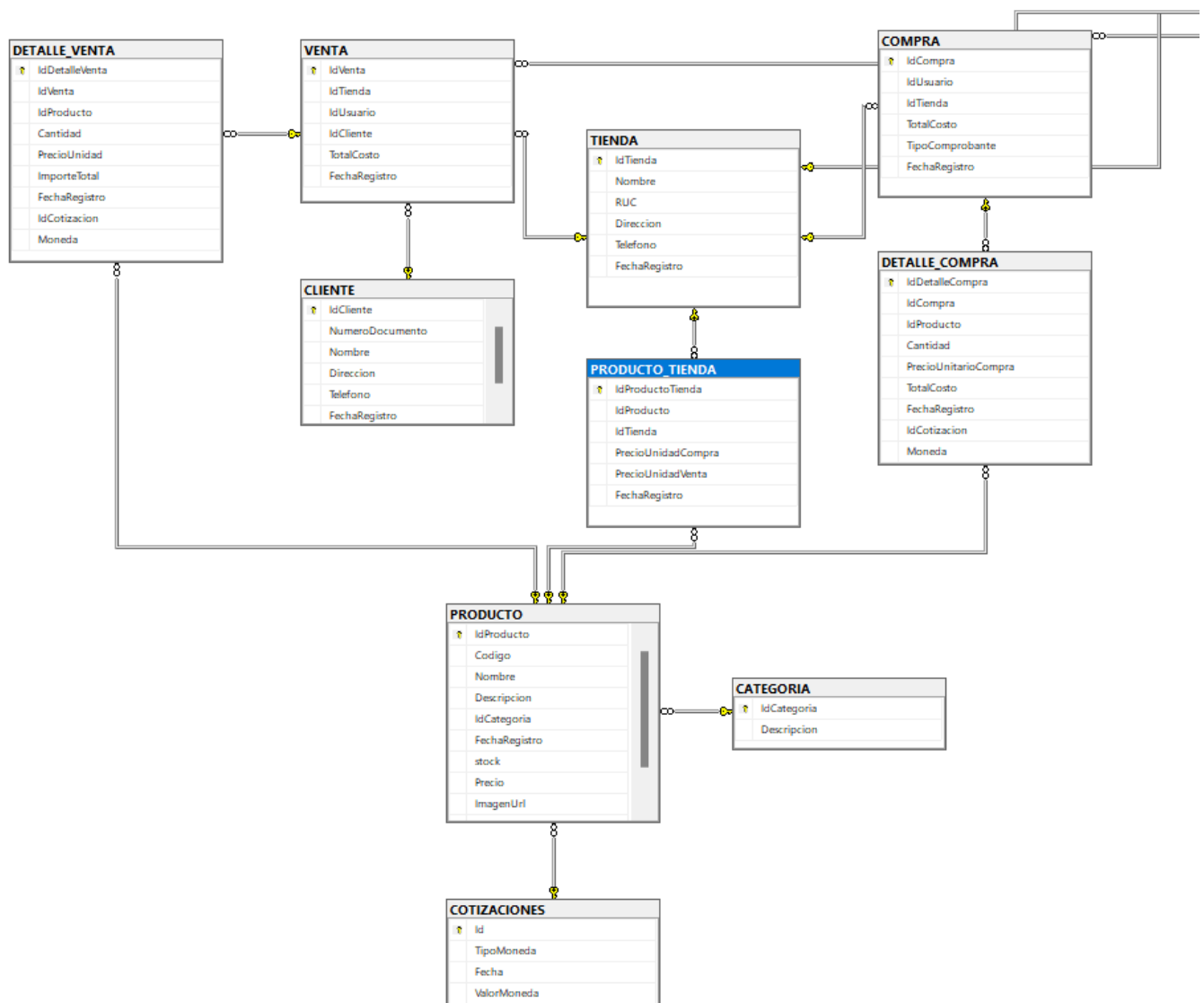
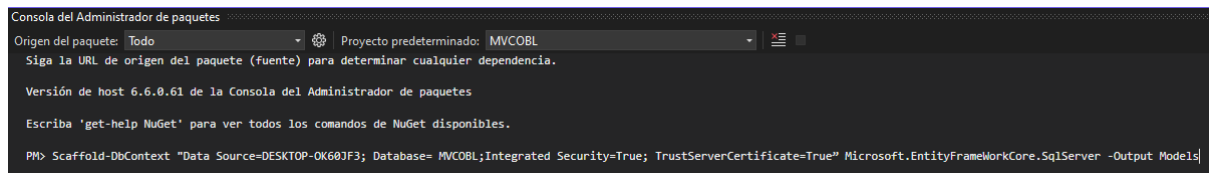


Imagen 1 - Diagrama de Entidades

A partir de esta base, se procedió a realizar un *scaffold* (una técnica para generar el código a partir de la base de datos) necesario para que la aplicación pueda realizar operaciones de creación, lectura, actualización y eliminación en la base de datos.

Durante el proceso de scaffold, se solicitó que todas las tablas generadas se almacenaran en una carpeta denominada "Models".



```
Consola del Administrador de paquetes
Origen del paquete: Todo Proyecto predeterminado: MVC0BL
Siga la URL de origen del paquete (fuente) para determinar cualquier dependencia.
Versión de host 6.6.0.61 de la Consola del Administrador de paquetes
Escriba 'get-help NuGet' para ver todos los comandos de NuGet disponibles.
PM> Scaffold-DbContext "Data Source=DESKTOP-OK60JF3; Database= MVC0BL;Integrated Security=True; TrustServerCertificate=True" Microsoft.EntityFrameworkCore.SqlServer -Output Models
```

Imagen 2 - Línea de Scaffold

Obteniendo como resultado una carpeta llamada “Models” dentro de nuestro proyecto conteniendo todos los modelos de las entidades de la base de datos.

Como se puede apreciar en la carpeta “Models”, existen 6 entidades que no fueron ilustradas en el diagrama anterior (Imagen 1), esto se debe a que al desarrollar nuestro programa en el modelo MVC en el entorno ASP.NET para generar la entidad de Usuario, la cual es utilizada para generar los perfiles de aquellas personas encargadas de manipular el sistema, se decidió utilizar IDENTITY para esta clase.

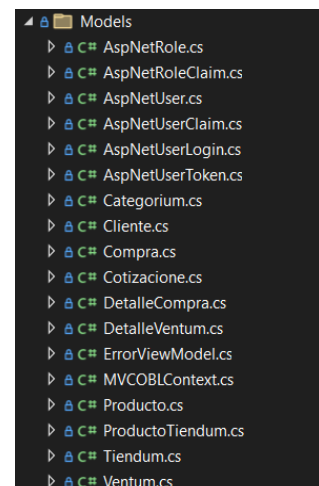


Imagen 3 - Carpeta "Models"

Identity

ASP.NET Identity es un sistema de membresía diseñado específicamente para aplicaciones .NET modernas. Proporciona funcionalidades completas de autenticación y autorización para autenticar y autorizar a los usuarios dentro del sistema.

La autenticación en ASP.NET Identity se refiere al proceso mediante el cual un usuario verifica su identidad. Esto se logra al proporcionar credenciales, como un correo electrónico y una contraseña, para iniciar sesión en el sistema. ASP.NET Identity maneja el proceso de autenticación de forma segura y permite verificar la validez de las credenciales proporcionadas por el usuario.

Una vez que el usuario ha sido autenticado con éxito, el sistema (o servidor) utiliza ASP.NET Identity para determinar si el cliente tiene los permisos necesarios para acceder a un recurso específico o realizar una acción particular. Esto se conoce como autorización. ASP.NET Identity proporciona una forma flexible de definir roles y políticas de autorización, lo que permite controlar de manera precisa el acceso a diferentes áreas o funcionalidades de una aplicación.⁵

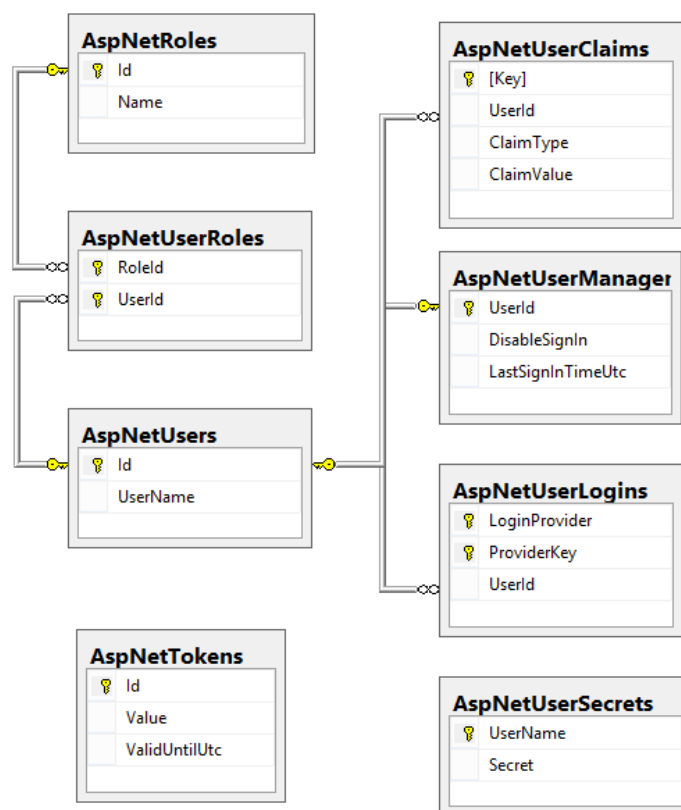


Imagen 4 - Entidades predefinidas de Identity

⁵ Introduction to ASP.NET Identity – <https://www.codeguru.com/dotnet/asp-net-identity-library/>

Continuando con el desarrollo del software de ventas, al aplicar las migraciones (Imagen 5), actualizar la base de datos y agregar las relaciones entre las tablas anteriormente existentes y las recién creadas (para el manejo de autorización y autenticación de usuarios) de la siguiente forma resulto nuestra base de datos:



A partir de cada entidad generada en la base de datos y con la utilización de Database First en nuestro proyecto logramos obtener un modelo de cada clase.

Es entonces que a partir de ese modelo podemos generar un controlador para cada uno, donde se seleccionó la opción de que fuera un controlador con acciones y vistas para crear, editar, leer y eliminar datos de sus respectivas entidades.

Dentro de ellos, está toda la lógica de acceso a datos, es decir, los métodos para poder realizar un CRUD (Create Read Update Delete) generándote un modelo básico de interacción con la entidad, como lo es ilustrado en la imagen 8.

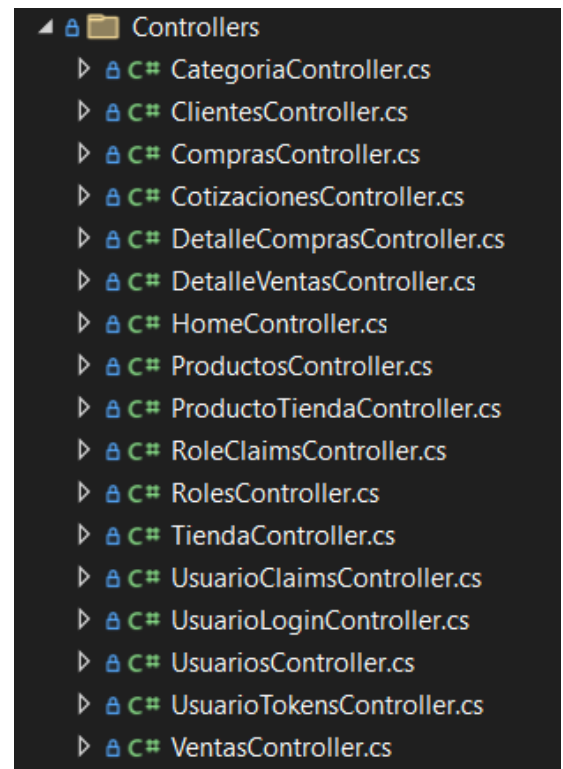


Imagen 7 - Controladores

Clientes

Crear

NumeroDocumento	Nombre	Direccion	Telefono	FechaRegistro			
51879468	Lautaro	Av. Agua 492	091837931	28/6/2023 19:49:03	Editar	Detalles	Borrar

Imagen 8 - Ilustración del CRUD

Acceso de los usuarios

ASP.NET Identity desempeña un papel fundamental en el siguiente aspecto del sistema. Gracias a este framework, se garantiza que todas las acciones de crear, leer, actualizar y eliminar (CRUD) solo sean realizadas por usuarios autenticados. Sin autenticación previa, no es posible llevar a cabo estas acciones. Por otro lado, se permite visualizar el catálogo de productos sin necesidad de iniciar sesión en el sistema.

Una vez que los usuarios están autenticados, entra en juego el componente de autorización de ASP.NET Identity. En la tabla de Roles mencionada anteriormente, se definen roles como "Administrador" y "Empleado". Los permisos y privilegios varían según el rol del usuario que accede al sistema.

El rol de "Administrador" posee los permisos más amplios y tiene acceso total a todas las funcionalidades y características del sistema, incluyendo la capacidad de realizar acciones CRUD en todos los elementos del sistema.

Por otro lado, el rol de "Empleado" tiene un conjunto más limitado de permisos y su acceso se restringe a ciertas funcionalidades y acciones específicas. Dependiendo de su rol, un usuario tendrá acceso y autorización solo para realizar acciones permitidas según su función en el sistema.

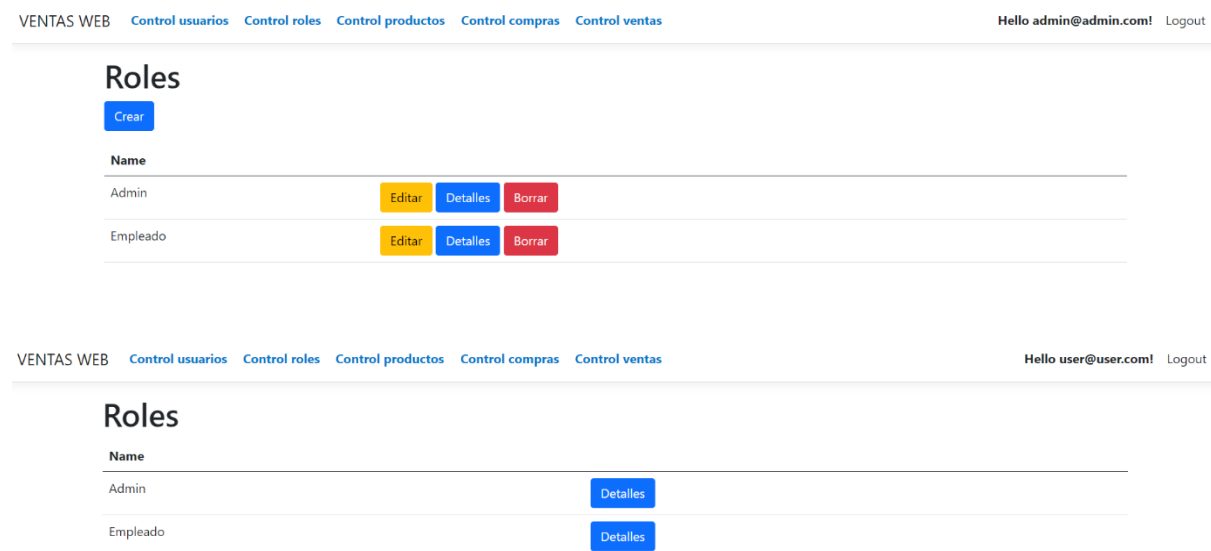


Imagen 9 - Vista y acceso de cada usuario, en la parte superior la del administrador y en la inferior la del empleado

Navegación

Para el diseño del encabezado de cada pestaña, se optó por utilizar un estilo sencillo y agradable a la vista. El encabezado se dividió en cinco partes principales: Control de usuarios, Productos, Control de compras, Control de ventas y Moneda.

Cada uno de estos módulos se presenta como un enlace en el encabezado, y al pasar el cursor sobre ellos, se despliega una lista que muestra el contenido específico relacionado con ese módulo en particular. El contenido de la lista varía según el módulo en el que estemos navegando.



Imagen 10 - Header de Navegación

Al hacer clic en el enlace deseado en el encabezado de navegación, somos redirigidos a la vista principal (Index) del controlador correspondiente. Desde allí, podemos realizar diversas acciones como crear, actualizar, eliminar y ver detalles de cada elemento de la entidad en cuestión. Estas acciones se llevan a cabo mediante botones presentes en la interfaz.

Tanto el encabezado de navegación como el diseño general se implementaron utilizando CSS (Cascading Style Sheets). El diseño visual y la apariencia del encabezado se definieron en un archivo .css ubicado en la carpeta "wwwroot" del proyecto el cual se vincula al sistema mediante una etiqueta (Imagen 11). Este archivo CSS, junto con otros recursos estáticos, se almacena en dicha carpeta (Imagen 12) para ser accesible y utilizado por el sistema.

```
<link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
<link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
<link rel="stylesheet" href="~/MVC0BL.styles.css" asp-append-version="true" />
```

Imagen 11 - Vinculación del CSS

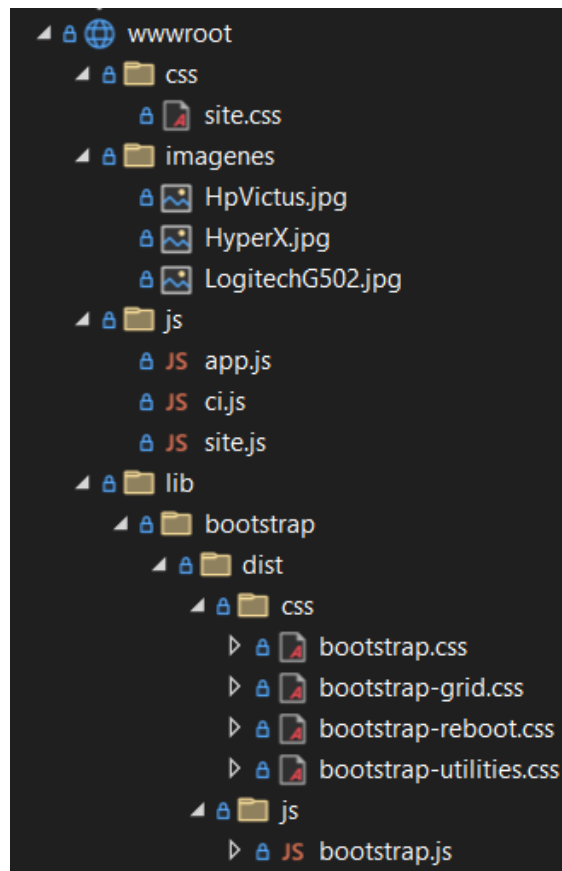


Imagen 12 - " Carpeta wwwroot"

Validación de cédula

En la imagen 11 se puede observar que dentro de la carpeta "wwwroot" hay una subcarpeta que contiene tres archivos de JavaScript. Dos de estos archivos, app.js y ci.js, se utilizaron para realizar la validación del número de identificación civil (cédula) de Uruguay, ya que esta validación era fundamental para el funcionamiento correcto del sistema.

Cuando se ingresa un nuevo cliente en el sistema, se solicitan los datos más importantes, y uno de ellos es el número de identificación civil. En Uruguay, este número cuenta con una validación propia que se determina mediante una fórmula matemática que genera un último valor. Esta validación implícita se utiliza dentro del programa para realizar una validación adicional y asegurarse de que el número de cédula ingresado sea válido.

Los archivos de JavaScript, en este caso app.js y ci.js, contienen la lógica necesaria para realizar la validación del número de cédula uruguaya dentro del programa. Estos scripts se ejecutan en el cliente (navegador web) y permiten verificar que el número de identificación ingresado cumpla con los criterios de validación establecidos por el sistema.

El código de la siguiente validación fue obtenido de otro colega desarrollador que lo posee en un repositorio de GitHub.⁶

The screenshot shows a web application interface. At the top, there is a navigation bar with links: 'EB', 'Control usuarios', 'Control roles', and 'Control productos'. Below this, the main heading is 'Crear cliente'. To the right of the heading, a modal message box is displayed with the text 'localhost:7258 dice' and 'Cedula Invalida', and a blue 'Aceptar' button. The form below contains the following fields: 'NumeroDocumento' with the value '45453', 'Nombre' with the value 'prueba', 'Direccion' with the value 'prueba', and 'Telefono' with the value '444444'. At the bottom of the form are two buttons: a blue 'Crear' button and a red 'Volver' button.

Imagen 13 - Mensaje emergente que alerta sobre la introducción errónea de la cédula

⁶ Validación cédula de identidad uruguaya - https://picandocodigo.github.io/ci_js/
https://github.com/picandocodigo/ci_js

Facturación

Dentro del programa, existen dos tipos de facturas: facturas de venta y facturas de compra. Estos tipos de facturas se dividen en dos partes principales.

En primer lugar, se crea un vínculo de Compra/Venta donde se le solicita al usuario ingresar los detalles que relacionan esta compra o venta con otras entidades. Por ejemplo, se puede solicitar el ID de usuario responsable en ese momento, la sucursal o el cliente asociado a la transacción.

Una vez que se selecciona "crear", el sistema redirige al usuario a los detalles específicos de esa factura. En esta sección, todo lo que se ingresa queda automáticamente asociado al ID de la factura correspondiente. A esto se le conoce como "línea de factura".

Conforme se agregan líneas de factura con los datos ingresados, como producto y cantidad, se muestran simultáneamente en una interfaz. Esto permite visualizar de manera clara y organizada cada una de las líneas de factura y sus respectivos detalles.

ID Compra	ID Producto	Cantidad	Precio unitario	Total
25	12	2	170,00	340,00
25	19	2	5677,00	11354,00

Finalizar

Imagen 14 - Línea de factura

En este caso específico, se agregaron dos unidades del producto con ID 12 y dos unidades del producto con ID 19 a la factura. El precio unitario no se ingresa manualmente, sino que se carga automáticamente utilizando el precio por unidad del producto seleccionado. Esto evita posibles manipulaciones de los datos al momento de crear la factura, lo cual es importante para garantizar la integridad de la información de la factura.

Del mismo modo, el total se calcula automáticamente multiplicando el precio unitario por la cantidad, obteniendo así el total de esa línea de factura en particular. Este cálculo asegura que el total refleje correctamente el importe correspondiente a la cantidad de productos y su precio unitario.

Cuando el usuario ha completado la factura y está listo para finalizar, debe hacer clic en el botón "Finalizar". Esto redirigirá al usuario nuevamente a la pestaña de compras, permitiendo continuar con otras acciones o crear nuevas facturas.

Dentro de la pestaña de compras, se muestra un historial de todas las compras realizadas en la empresa. Cada compra se presenta de forma clara y organizada, proporcionando información relevante sobre la transacción.

Además del historial de compras, se encuentran dos botones adyacentes. El primer botón permite borrar la compra seleccionada, lo que brinda la posibilidad de eliminar una compra específica del registro. Esto puede ser útil en situaciones donde se requiere corregir un error o eliminar una transacción incorrecta.

El segundo botón permite visualizar la factura asociada a la compra seleccionada. Al hacer clic en este botón, se abre la factura correspondiente en una vista detallada. Esto permite revisar la información específica de la factura, incluyendo detalles de los productos, precios, cantidades y cualquier otra información relevante.

Compras

Crear

Usuario	Sucursal	Fecha		
user@user.com	MALDONADO	12/7/2023 3:31:04	Borrar	Factura
user@user.com	MALDONADO	12/7/2023 20:51:03	Borrar	Factura
user@user.com	MALDONADO	12/7/2023 20:51:39	Borrar	Factura

Imagen 15 - Índice de Compras

Factura

Cotizacion USD38,05

Usuario	Tipo	Fecha	Id Tienda
6d6f667f-9d69-4ebb-bf63-2c77fee51f6d	Compra	12/7/2023 10:27:11	1

ID Producto	Cantidad	Precio unitario	Moneda	Total
12	1	170,00	UYU	170,00
19	1	1500,00	UYU	1500,00
20	2	500,00	USD	38050,00

TOTAL
39720,00

Imagen 16 - Modelo de factura

Cotizaciones

En relación a la API, el sistema se encuentra conectado a una API que proporciona cotizaciones diarias de diferentes partes del mundo. Esto es necesario debido a que, como se mencionó anteriormente, los precios de los productos pueden estar en dólares o en pesos uruguayos, y se solicita especificar la moneda al dar de alta un producto en el sistema.⁷

En el sistema, se obtiene y almacena la cotización actual del dólar utilizando la API. La actualización de la cotización se realiza automáticamente una vez al día, ya que la API limita la cantidad de consultas que se pueden realizar. No obstante, se ha incluido un botón específico que permite actualizar la cotización en cualquier momento según la preferencia del usuario.

Todas las cotizaciones del dólar obtenidas se guardan en la base de datos, registrando el valor de la moneda en esa fecha específica. Esto permite tener un historial detallado de las cotizaciones a lo largo del tiempo, lo cual puede ser útil para análisis, seguimiento y referencias posteriores.

Cotizaciones

Actualizar Cotización

Moneda	Valor	Fecha	
USD	38,20	11/7/2023 0:00:00	Borrar
USD	38,09	12/7/2023 0:00:00	Borrar
USD	38,05	12/7/2023 0:00:00	Borrar

Imagen 17 - Índice del apartado de Cotizaciones

⁷ Api de Cotizaciones - <https://currencylayer.com/documentation>

Adicionalmente, la cotización del dólar se muestra en la factura final (imagen16), tanto para las facturas de venta como para las facturas de compra. Como se mencionó previamente, todas las transacciones de facturación se manejan exclusivamente en pesos uruguayos.

En la factura, para cada línea de factura, se realiza una conversión de moneda según corresponda. Si el precio de un producto está ingresado en dólares, se multiplica la cantidad seleccionada de ese producto por la cotización del dólar vigente en el momento de realizar la compra o venta.

De esta manera, al tener el total de cada línea de factura convertido a pesos uruguayos, simplemente se suman todos estos totales para obtener el total final de la factura en pesos uruguayos. Esto garantiza que el total refleje de manera precisa el valor en la moneda local.

La conversión de moneda en la factura asegura una presentación clara y coherente de los montos en la moneda utilizada en el sistema y simplifica el proceso de seguimiento y contabilidad para la empresa.

ID Producto	Cantidad	Precio unitario	Moneda	Total en Pesos
12	1	170,00	UYU	170,00
19	1	1500,00	UYU	1500,00
20	1	500,00	USD	19025,00

Imagen 18 - Líneas de una factura donde se puede apreciar el precio en dolares pero convertido en pesos en el total

Conclusión

Durante este proyecto, el objetivo principal fue desarrollar un sistema de ventas web utilizando ASP.NET MVC. Este sistema fue diseñado para facilitar y mejorar el proceso de ventas de una empresa, brindando una herramienta útil y eficiente para administrar el inventario de productos, gestionar tiendas, clientes, realizar transacciones de compra y venta, además de generar informes.

A lo largo del proyecto, se logró desarrollar un sistema de ventas web funcional y completo. Se implementaron características clave que, a nuestro parecer, son las más vitales del sistema, como la gestión de productos, el registro de clientes, manejar una moneda extranjera y lo más importante, generar facturas de compra y de venta.

Este sistema de ventas web desarrollado en ASP.NET MVC tiene una gran importancia y relevancia en la actualidad. Proporciona una solución eficiente para agilizar y optimizar los procesos de compra y venta, lo que puede resultar en una gran mejora en el manejo de las empresas.

Durante el desarrollo de este proyecto, logramos un gran aprendizaje. Pudimos mejorar nuestras habilidades en el desarrollo web utilizando ASP.NET MVC. Además, hemos aprendido a trabajar en equipo, gestionar nuestros tiempos de una mejor manera y dividir tareas con más facilidad.

Aunque se logró desarrollar un sistema de ventas web completo, existen áreas que podrían mejorarse y expandirse en el futuro. Por ejemplo, se podrían implementar funciones adicionales, como la integración con sistemas de pago en línea, o la incorporación de características de análisis de datos.

Referencias

[1,2] ¿Qué es MVC? Desarrollo web

<https://desarrolloweb.com/articulos/que-es-mvc.html>

[3] ¿Qué es una API y cómo funciona?

<https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>

[4] Entity Framework

<https://desarrolloweb.com/home/entity-framework>

[5] Introduction to ASP.NET Identity –

<https://www.codeguru.com/dotnet/asp-net-identity-library/>

[6] Validación cédula de identidad uruguaya

https://picandocodigo.github.io/ci_js/

https://github.com/picandocodigo/ci_js

[7] API de Cotizaciones

<https://currencylayer.com/documentation>