

Facultad de Informática y Ciencias Aplicadas

Escuela de Informática / Desarrollo de Sistemas Informáticos Web 1 (DSWI-1)



ASIGNATURA:

Desarrollo de Sistemas Informáticos Web 1 (DSWI-1)

SECCION 01, CICLO 01-2024

CATEDRATICO:

Ing. Kirio Marvin Ventura Fuentes

TEMA:

**Desarrollar una aplicación web para la venta de
suministros y la elaboración de café.**

INTEGRANTES DE GRUPO:

Escamilla Salinas Raúl Arturo	27-1589-1999
González Espinal Kimberly Ali	27-3337-2023
González Fuentes Silvia Marisol	27-1932-2023
Hernández Montes Miguel Ángel	27-1098-2023

Fecha de entrega: miércoles 21 de mayo de 2024

Introducción

En la era actual, las tiendas en línea se erigen como pilares fundamentales para las empresas que ansían alcanzar a un público más amplio y distribuir sus productos de manera eficiente. En esta dinámica, surge el presente documento que detalla el proceso de desarrollo de una aplicación web, destinada a ofrecer una en línea de suministros para la elaboración de café. Entre los artículos que abarcará se encuentran el Chemex, V60, filtros Aeropress, y otros elementos esenciales para los aficionados al café de alta calidad.

La aplicación será concebida dentro del entorno .NET Framework, empleando ASP.NET y C# para el backend, y HTML, Css con Bootstrap y Javascript para el frontend. Estas tecnologías han sido seleccionadas por su robustez, seguridad y facilidad de uso, lo que permitirá forjar una aplicación web sólida y atractiva para los usuarios.

A lo largo de este documento, se expondrán en detalle los requerimientos del sistema, la arquitectura de la aplicación, el diseño de la interfaz de usuario, la implementación técnica y las pruebas realizadas para garantizar su correcto funcionamiento. Cabe destacar que este documento será actualizado de manera continua durante el ciclo 01/2024 de la materia de Desarrollo de Sistemas Informáticos Web 1, en consonancia con las diferentes evaluaciones programadas, totalizando 5 entregas en total.

Con esta aplicación, se aspira a brindar a los usuarios una experiencia de compra en línea fluida y satisfactoria, que les conceda el placer de disfrutar de los mejores suministros para la elaboración de café desde la comodidad de su hogar. Además, se explorarán aspectos adicionales como la estrategia de marketing digital, la seguridad de la plataforma y la escalabilidad del sistema para adaptarse a las demandas del mercado en constante evolución.

Es importante destacar que la aplicación de tienda en línea mencionada será desarrollada específicamente para MiCafesito, una empresa ficticia especializada en la venta de equipos y suministros para la elaboración de café de alta calidad. Con el fin de modernizar sus operaciones y expandir su alcance comercial, MiCafesito ha decidido incursionar en el comercio electrónico mediante el desarrollo de esta plataforma web. A través de esta iniciativa, MiCafesito busca ofrecer a sus clientes una experiencia de compra en línea fluida y satisfactoria, facilitando el acceso a sus productos desde cualquier ubicación con conexión a internet.

índice

Fecha de entrega: miércoles 21 de abril de 2024	1
Introducción	2
Objetivo General:	4
Objetivos Específicos:	4
Marco de Referencia:.....	5
Descripción detallada del análisis de la aplicación:	5
Alcance	8
Cronograma de actividades para el desarrollo de la aplicación web MiCafecito:	8
Delimitación Espacial:.....	9
Delimitaciones Temporales:.....	10
Análisis de Requerimientos del Sistema:	10
Módulo 1: Autenticación	11
Módulo 2: Usuarios	13
Módulo 3: Catálogo de Productos	15
Módulo 4: Pedidos	18
Base de Datos - Esquema Relacional	23
Diccionario de Datos	23
Script Base de datos	26
Scripts Procedimientos Almacenados	27
Capa de Servicios.....	35
DataContracts	35
ServiceContracts	40
Implementación de servicios	43
Utilitarios.....	67
Capa de Presentación Asp.net.....	68
Services auxiliares	85
Secciones tienda en línea	89
Capturas de pantalla	107
Página principal (Vista completa).....	107
Imagen principal y menú.....	108
Área de productos.....	108
Página Login	109
Carrito de compras.....	110
Proceso de orden	110
Menu de opciones.....	111
Detalle del producto	111
Pedidos realizados	112
Conclusiones:	112
Algunas recomendaciones sobre el proyecto:.....	113
Bibliografía:	113

Objetivo General:

Desarrollar una aplicación web que brinde a los usuarios una experiencia de compra en línea fluida y satisfactoria, permitiéndoles obtener información de como adquirir de los mejores suministros para la elaboración de café como equipos para elaboración con métodos (V60, Chemex, Prensas francesas, Aeropress, etc) y máquinas de expreso (Tampers, filtros, dosificadores, etc.), así como suministros como filtros, desde la comodidad de su hogar proporcionando una experiencia satisfactoria para los usuarios.

Objetivos Específicos:

1. Implementar un sistema de registro y almacenamiento de suministros para la venta en línea.
2. Diseñar una interfaz de usuario intuitiva que facilite la navegación y la selección de productos.
3. Integrar medidas de seguridad sólidas para proteger la información de los usuarios durante el proceso de compra.
4. Implementar un sistema de pago seguro que permita a los usuarios realizar transacciones en línea de manera confiable, utilizando métodos de pago populares y garantizando la protección de la información financiera del cliente.
5. Ofrecer contenido informativo y educativo sobre la elaboración de café, incluyendo guías paso a paso, recomendaciones de productos y técnicas de preparación, con el objetivo de fomentar el interés y la participación de los usuarios en el arte del café.
6. Proporcionar un servicio de atención al cliente efectivo, mediante la integración de un sistema de soporte que permita a los usuarios realizar consultas, recibir asistencia personalizada y resolver cualquier problema o duda relacionada con la aplicación o los productos ofrecidos.

Marco de Referencia:

Contexto Empresarial:

La empresa MiCafesito se dedica a la venta de equipos y suministros para la elaboración de café de alta calidad. Con el objetivo de modernizar sus operaciones y expandir su alcance comercial, MiCafesito ha decidido incursionar en el comercio electrónico mediante el desarrollo de una tienda en línea.

La incursión en el comercio electrónico a través de una tienda en línea es crucial en la actualidad debido al crecimiento exponencial del comercio electrónico en todo el mundo. Según datos de Statista, las ventas minoristas en línea a nivel mundial alcanzaron los 4.8 billones de dólares en 2021, representando un aumento significativo respecto a años anteriores. Esta tendencia se ve impulsada por la creciente preferencia de los consumidores por realizar compras en línea debido a su conveniencia, variedad de productos y facilidad de acceso.

Además, la pandemia de COVID-19 ha acelerado la adopción del comercio electrónico, con un incremento notable en el número de consumidores que prefieren realizar compras en línea para evitar el contacto físico y cumplir con las medidas de distanciamiento social. Según un informe de Adobe, las ventas en línea aumentaron un 32% en el primer trimestre de 2021 en comparación con el mismo período del año anterior, evidenciando el impacto significativo de la pandemia en el comportamiento de compra de los consumidores.

Por lo tanto, incursionar en una tienda en línea no solo permite a MiCafesito adaptarse a las nuevas tendencias de compra de los consumidores, sino que también le brinda la oportunidad de alcanzar a un público más amplio, expandir sus operaciones a nivel nacional e internacional, y mejorar la accesibilidad y conveniencia para sus clientes.

Descripción detallada del análisis de la aplicación:

La aplicación de tienda en línea de MiCafesito está diseñada para resolver el problema principal de la empresa: la necesidad de modernizar sus operaciones comerciales y expandir su alcance comercial. Esta aplicación ofrece una solución integral al permitir que MiCafesito traslade su modelo de negocio al ámbito digital, brindando a los usuarios una experiencia de compra en línea fluida y satisfactoria.

En primer lugar, la aplicación presentará una lista de suministros en línea completo y actualizado de equipos para la elaboración de café de alta calidad. Los usuarios pueden explorar una amplia variedad de productos, desde Chemex y V60 hasta filtros Aeropress y otros elementos esenciales, con descripciones detalladas, imágenes y precios claros.

Además, la aplicación ofrece una interfaz de usuario intuitiva y fácil de navegar, lo que facilita a los usuarios encontrar y seleccionar los productos deseados con facilidad. Un diseño limpio y moderno garantiza una experiencia de usuario atractiva y satisfactoria, lo que mejora la usabilidad y la accesibilidad de la aplicación.

La aplicación también incluye funcionalidades clave para simplificar el proceso de compra en línea. Los usuarios pueden agregar productos a su carrito de compras, revisar su selección y proceder al pago de forma segura y conveniente. Se ofrecen múltiples opciones de pago, incluidas tarjetas de crédito, transferencias bancarias y plataformas de pago en línea, para adaptarse a las preferencias de los usuarios.

Además, la aplicación prioriza la seguridad de la información del usuario. Se implementan medidas de seguridad robustas, como encriptación de datos, autenticación de usuarios y certificados SSL, para proteger la información personal y financiera de los usuarios durante el proceso de compra en línea.

Los usuarios podrán acceder a la aplicación web a través de un formulario de inicio de sesión. También se les proporcionará la opción de registrarse si son nuevos usuarios. Para el inicio de sesión, se solicitarán credenciales como correo electrónico y contraseña.

La aplicación contará con roles y perfiles de usuario para gestionar los diferentes niveles de acceso y funcionalidades. Entre los roles posibles se incluyen:

- Administrador: Tendrá acceso completo a todas las funciones de la aplicación, incluida la gestión de usuarios, productos, pedidos y contenido.
- Usuario Regular: Podrá navegar por la tienda, realizar compras, ver su historial de pedidos y acceder a contenido educativo sobre café.
- Usuario No Registrado: Podrá navegar por la tienda y ver los productos, pero no podrá realizar compras ni acceder a funcionalidades exclusivas para usuarios registrados.

Contará con seguridad y privacidad que se implementará las medidas de seguridad como cifrado de datos, protección contra ataques de seguridad y políticas de privacidad claras para garantizar la seguridad y confidencialidad de la información de los usuarios.

Se requiere establecer una base sólida para el desarrollo de la aplicación web MiCafesito, asegurando una experiencia de usuario fluida y funcionalidades eficientes para la gestión de productos y pedidos relacionados con el café.

Tecnologías Utilizadas:

El desarrollo de la aplicación se llevará a cabo utilizando tecnologías sólidas y seguras. Se utilizará el entorno .NET Framework, con ASP.NET y C# para el backend, y Bootstrap para el frontend. Estas tecnologías fueron seleccionadas por su fiabilidad, seguridad y facilidad de uso, además de cumplir con los objetivos de aprendizaje de la materia de Desarrollo de Sistemas Informáticos Web 1.

Seguridad de la Información:

Se implementarán medidas de seguridad sólidas para proteger la información confidencial de los usuarios, como datos personales y detalles de pago. Se utilizarán prácticas de seguridad estándar de la industria, como encriptación de datos, autenticación de usuarios y certificados SSL, para garantizar la confidencialidad e integridad de la información.

Metodología de Desarrollo:

Se seguirá una metodología híbrida de desarrollo ágil adaptada a equipos pequeños de trabajo que permita iteraciones rápidas y flexibles para adaptarse a los cambios y requerimientos del proyecto y las diferentes entregas durante el ciclo 01/2024. Se establecerán entregas periódicas y pruebas continuas para garantizar la calidad y funcionalidad de la aplicación en cada etapa del desarrollo.

Alcance

El alcance de la aplicación de tienda en línea de MiCafesito abarcará el desarrollo e implementación de una plataforma web que permita a los usuarios explorar un catálogo completo de equipos y suministros para la elaboración de café de alta calidad. La aplicación estará disponible en línea durante el período del ciclo 01/2024 de la materia de Desarrollo de Sistemas Informáticos Web 1.

La aplicación contará con las siguientes características principales:

1. Catálogo en línea: Se mostrará un catálogo completo y actualizado de productos, incluyendo descripciones detalladas, imágenes y precios.
2. Interfaz de usuario intuitiva: Se diseñará una interfaz de usuario fácil de navegar, con un diseño limpio y moderno para mejorar la experiencia del usuario.
3. Formulario de contacto: Se proporcionará un formulario de contacto para que los usuarios interesados puedan realizar consultas y solicitar información sobre los productos mostrados en el catálogo.
4. Información adicional: Se incluirá información sobre MiCafesito, como su historia, misión y visión, así como detalles de contacto adicionales para consultas fuera de línea.

Cronograma de actividades para el desarrollo de la aplicación web MiCafesito:

Definición de Requerimientos y Diseño Inicial

- Reunión inicial con el equipo para discutir objetivos y requerimientos.
- Investigación de mercado y análisis de la competencia.
- Definición de la arquitectura general del sistema.
- Diseño de la interfaz de usuario y experiencia de usuario.

Desarrollo de la Funcionalidad Básica

- Configuración del entorno de desarrollo.
- Implementación del sistema de registro e inicio de sesión de usuarios.
- Desarrollo de la estructura de la base de datos para usuarios, productos y pedidos.
- Creación de las páginas principales de la aplicación: inicio, lista de suplementos y perfil de usuario.

Desarrollo de Funcionalidades Avanzadas

- Implementación del sistema de gestión de productos y pedidos.
- Desarrollo del sistema de pago en línea y métodos de envío.
- Integración de herramientas de análisis y seguimiento de usuarios.

- Incorporación de contenido educativo sobre café y métodos de preparación.

Pruebas y Optimización

- Pruebas exhaustivas de todas las funcionalidades de la aplicación.
- Corrección de errores y optimización del rendimiento.
- Ajustes finales en el diseño y la usabilidad de la interfaz.
- Preparación para el lanzamiento y configuración de servidores y dominios.

Mantenimiento y**Actualizaciones**

- Monitoreo continuo del rendimiento y la seguridad de la aplicación.
- Implementación de actualizaciones periódicas para agregar nuevas funcionalidades y corregir errores.
- Atención al cliente y soporte técnico para resolver consultas y problemas de los usuarios.
- Evaluación de métricas clave y ajustes estratégicos según el desempeño de la aplicación

Delimitación Espacial:

La delimitación espacial de la aplicación de tienda en línea de MiCafesito estará definida por su disponibilidad en línea durante el período del ciclo 01/2024 de la materia de Desarrollo de Sistemas Informáticos Web 1. La aplicación estará accesible a través de internet y estará dirigida a usuarios ubicados en cualquier parte del mundo.

Debido a las limitaciones de tiempo, la implementación de la sección de compra de productos se pospondrá para futuras actualizaciones fuera del alcance de este proyecto académico. En su lugar, la aplicación mostrará únicamente el catálogo en línea de suplementos y proporcionará un formulario de contacto para realizar compras offline.

Estas delimitaciones permitirán enfocar los esfuerzos del desarrollo en la creación de una plataforma web funcional y atractiva que cumpla con los objetivos establecidos dentro de los límites de tiempo y recursos disponibles.

Delimitaciones Temporales:

Fecha de Lanzamiento: Establecer una fecha específica para el lanzamiento oficial de la aplicación web MiCafecito. Esta fecha marcará el final del período de desarrollo y el inicio de la fase de implementación.

Período de Desarrollo: Definir un período de tiempo específico para el desarrollo de la aplicación, que puede estar determinado por factores como recursos disponibles, complejidad del proyecto y objetivos. Este período incluirá todas las etapas de diseño, desarrollo, pruebas y optimización de la aplicación.

Actualizaciones y Mantenimiento: Establecer una frecuencia para las actualizaciones y el mantenimiento continuo.

Ciclo de Vida del Producto: Reconocer que la aplicación web MiCafecito tendrá un ciclo de vida limitado, con un período de tiempo durante el cual será relevante y competitiva en el mercado. Esto implica la necesidad de estar preparado para adaptarse a los cambios en las demandas de los usuarios y las tendencias de la industria a lo largo del tiempo.

Análisis de Requerimientos del Sistema:

El objetivo de esta sección es proporcionar una comprensión clara y completa de las necesidades del sistema a los stakeholders y al equipo de desarrollo.

El Sistema se divide en 4 módulos según se detalla a continuación:

1. Módulo de Autenticación.
2. Módulo de Administración de Usuarios.
3. Módulo Catálogo de Productos.
4. Módulo de Pedidos.

Módulo 1: Autenticación

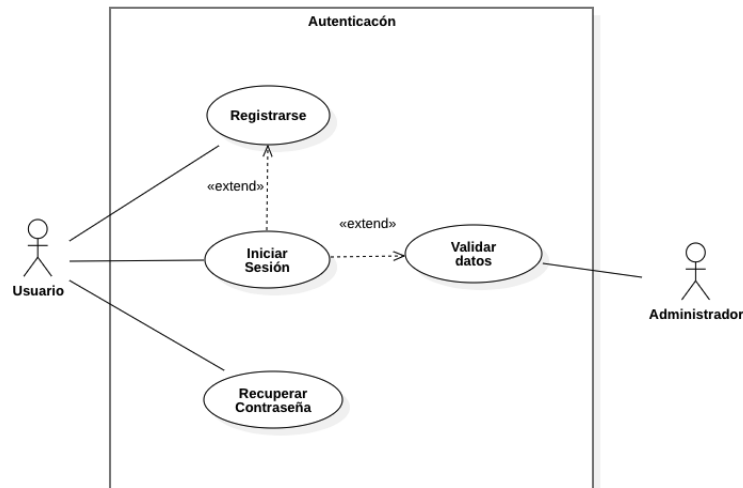


Diagrama de casos de uso módulo Autenticación

Caso de Uso 1: Registrarse

Descripción: Este caso de uso permite a un usuario nuevo registrarse en el sistema proporcionando la información requerida.

Actores: Usuario

Precondiciones: El usuario no debe tener una cuenta registrada previamente.

Flujo Principal:

1. El usuario accede a la página de registro.
2. El usuario completa el formulario de registro con su información personal.
3. El sistema valida la información ingresada por el usuario.
4. El sistema crea una cuenta para el usuario.

Postcondiciones: El usuario queda registrado en el sistema y puede iniciar sesión.

Caso de Uso 2: Iniciar Sesión

Descripción: Este caso de uso permite a un usuario iniciar sesión en el sistema utilizando sus credenciales.

Actores: Usuario y Sistema

Precondiciones: El usuario debe estar registrado en el sistema.

Flujo Principal:

1. El usuario accede a la página de inicio de sesión.
2. El usuario ingresa su nombre de usuario y contraseña.
3. El sistema valida las credenciales del usuario.

4. El sistema inicia sesión y redirige al usuario a su perfil.

Postcondiciones: El usuario inicia sesión en el sistema y puede acceder a las funcionalidades disponibles.

Caso de Uso 3: Validar Credenciales

Descripción: Este caso de uso permite al sistema validar las credenciales proporcionadas por un usuario durante el proceso de inicio de sesión.

Actores: Sistema

Precondiciones: El usuario intenta iniciar sesión en el sistema.

Flujo Principal:

1. El sistema recibe las credenciales ingresadas por el usuario.
2. El sistema verifica si las credenciales son válidas comparándolas con la información almacenada en la base de datos.

Postcondiciones: El sistema determina si las credenciales son válidas y permite el acceso del usuario si es el caso.

Caso de Uso 4: Recuperar Contraseña

Descripción: Este caso de uso permite a un usuario recuperar su contraseña en caso de olvido.

Actores: Usuario

Precondiciones: El usuario debe haber olvidado su contraseña y desear recuperarla.

Flujo Principal:

1. El usuario accede a la opción de recuperar contraseña.
2. El usuario proporciona su correo electrónico asociado a la cuenta.
3. El sistema envía un correo electrónico al usuario con un enlace o instrucciones para restablecer la contraseña.
4. El usuario sigue las instrucciones recibidas en el correo para restablecer su contraseña.

Postcondiciones: El usuario puede restablecer su contraseña y volver a acceder al sistema.

Módulo 2: Usuarios

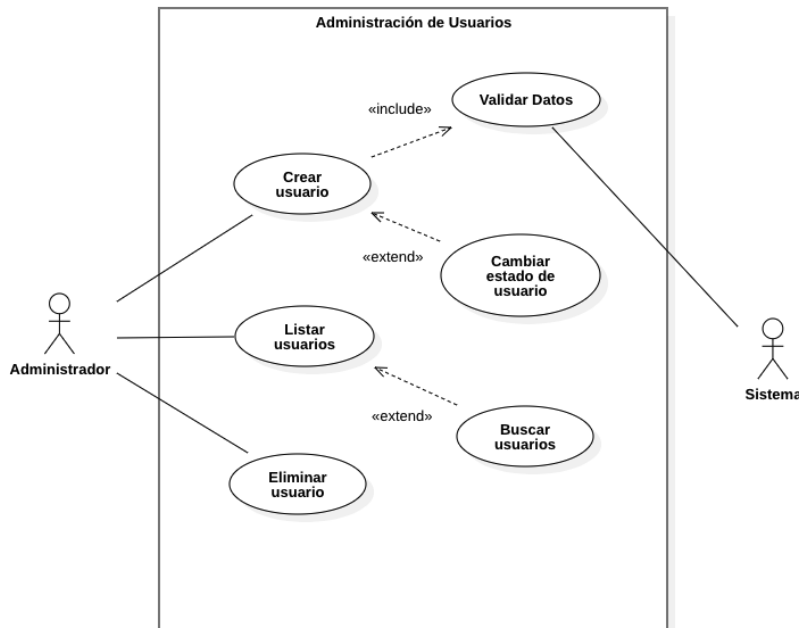


Diagrama de casos de uso módulo Administración de Usuarios

Caso de Uso 5: Listar Usuarios

Descripción: Este caso de uso permite al administrador listar todos los usuarios registrados en el sistema.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema.

Flujo Principal:

1. El administrador accede a la sección de administración de usuarios.
2. El sistema muestra una lista de todos los usuarios registrados en el sistema.

Postcondiciones: El administrador puede ver la lista de usuarios registrados.

Caso de Uso 6: Buscar Usuarios

Descripción: Este caso de uso permite al administrador buscar usuarios por nombre de usuario, correo electrónico u otro criterio de búsqueda.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema.

Flujo Principal:

1. El administrador accede a la función de búsqueda de usuarios.
2. El administrador ingresa el criterio de búsqueda.
3. El sistema realiza la búsqueda y muestra los resultados coincidentes.

Postcondiciones: El administrador puede ver los resultados de la búsqueda de usuarios.

Caso de Uso 7: Crear Usuario

Descripción: Este caso de uso permite al administrador crear un nuevo usuario en el sistema.

Actores: Administrador y sistema

Precondiciones: El administrador ha iniciado sesión en el sistema.

Flujo Principal:

1. El administrador accede a la función de creación de usuarios.
2. El administrador completa el formulario con la información del nuevo usuario.
3. El sistema valida la información ingresada.
4. El sistema crea una cuenta para el nuevo usuario.

Postcondiciones: Se crea una cuenta para el nuevo usuario en el sistema.

Caso de Uso 8: Cambiar Estado de Usuario

Descripción: Este caso de uso permite al administrador cambiar el estado (activo, inactivo, suspendido, etc.) de un usuario en el sistema.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema.

Flujo Principal:

1. El administrador accede a la función de gestión de usuarios.
2. El administrador selecciona al usuario cuyo estado desea cambiar.
3. El administrador elige el nuevo estado para el usuario.
4. El sistema actualiza el estado del usuario en la base de datos.

Postcondiciones: El estado del usuario seleccionado se actualiza en el sistema.

Caso de Uso 9: Eliminar Usuario

Descripción: Este caso de uso permite al administrador eliminar un usuario del sistema.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema.

Flujo Principal:

1. El administrador accede a la función de gestión de usuarios.
2. El administrador selecciona al usuario que desea eliminar.
3. El administrador confirma la eliminación del usuario.
4. El sistema elimina al usuario de la base de datos.

Postcondiciones: El usuario seleccionado es eliminado del sistema y ya no tiene acceso.

Módulo 3: Catálogo de Productos

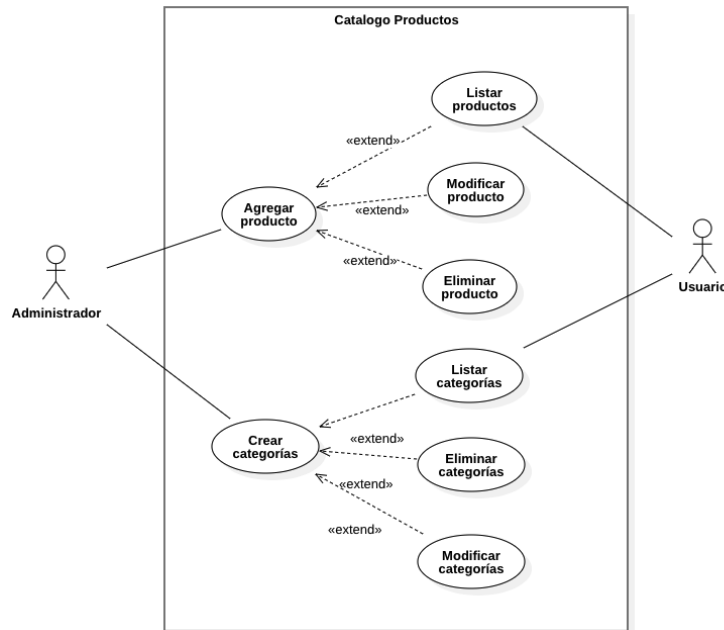


Diagrama de casos de uso módulo Catálogo de Productos

Caso de Uso 10: Agregar Producto

Descripción: Este caso de uso permite al administrador agregar un nuevo producto al catálogo.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para agregar productos.

Flujo Principal:

1. El administrador accede a la función de agregar producto.
2. El administrador completa el formulario con la información del nuevo producto, incluyendo nombre, descripción, categoría, precio, etc.
3. El sistema valida la información ingresada.
4. El sistema agrega el nuevo producto al catálogo.

Postcondiciones: El nuevo producto se agrega al catálogo y está disponible para su compra.

Caso de Uso 11: Modificar Producto

Descripción: Este caso de uso permite al administrador modificar la información de un producto existente en el catálogo.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para modificar productos.

Flujo Principal:

1. El administrador accede a la función de modificar producto.
2. El administrador selecciona el producto que desea modificar.
3. El administrador edita la información del producto según sea necesario.
4. El sistema valida los cambios realizados.
5. El sistema actualiza la información del producto en el catálogo.

Postcondiciones: Los cambios en la información del producto se guardan correctamente en el sistema.

Caso de Uso 12: Eliminar Producto

Descripción: Este caso de uso permite al administrador eliminar un producto del catálogo.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para eliminar productos.

Flujo Principal:

1. El administrador accede a la función de eliminar producto.
2. El administrador selecciona el producto que desea eliminar.
3. El administrador confirma la eliminación del producto.
4. El sistema elimina el producto seleccionado del catálogo.

Postcondiciones: El producto seleccionado es eliminado del catálogo y ya no está disponible para su compra.

Caso de Uso 13: Listar Productos

Descripción: Este caso de uso permite a los usuarios ver una lista de todos los productos disponibles en el catálogo.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema.

Flujo Principal:

1. El usuario accede a la sección de catálogo de productos.
2. El sistema muestra una lista de todos los productos disponibles.

Postcondiciones: El usuario puede ver la lista de productos disponibles para su compra.

Caso de Uso 14: Listar Categorías

Descripción: Este caso de uso permite a los usuarios ver una lista de todas las categorías disponibles en el catálogo.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema.

Flujo Principal:

1. El usuario accede a la sección de categorías de productos.
2. El sistema muestra una lista de todas las categorías disponibles.

Postcondiciones: El usuario puede ver la lista de categorías disponibles para explorar productos.

Caso de Uso 15: Crear Categoría

Descripción: Este caso de uso permite al administrador crear una nueva categoría para clasificar productos en el catálogo.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para crear categorías.

Flujo Principal:

1. El administrador accede a la función de crear categoría.
2. El administrador completa el formulario con la información de la nueva categoría, incluyendo nombre, descripción, etc.
3. El sistema valida la información ingresada.
4. El sistema crea la nueva categoría en el catálogo.

Postcondiciones: La nueva categoría se agrega al catálogo y está disponible para clasificar productos.

Caso de Uso 16: Modificar Categoría

Descripción: Este caso de uso permite al administrador modificar la información de una categoría existente en el catálogo.

Actores: Administrador

Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para modificar categorías.

Flujo Principal:

1. El administrador accede a la función de modificar categoría.
2. El administrador selecciona la categoría que desea modificar.
3. El administrador edita la información de la categoría según sea necesario.

4. El sistema valida los cambios realizados.
5. El sistema actualiza la información de la categoría en el catálogo.

Postcondiciones: Los cambios en la información de la categoría se guardan correctamente en el sistema.

Caso de Uso 17: Eliminar Categoría

Descripción: Este caso de uso permite al administrador eliminar una categoría del catálogo, junto con todos los productos asociados a ella.

Actores: Administrador

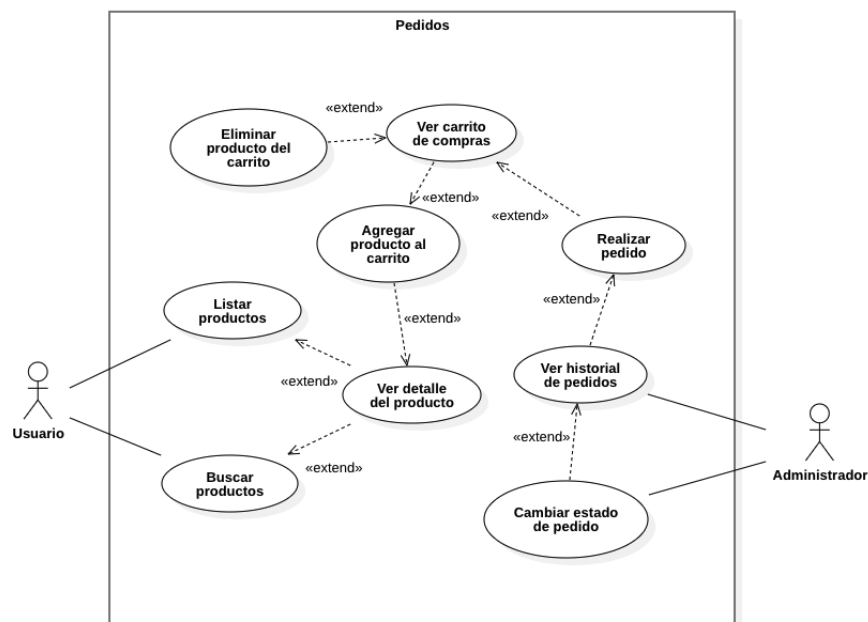
Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para eliminar categorías.

Flujo Principal:

1. El administrador accede a la función de eliminar categoría.
2. El administrador selecciona la categoría que desea eliminar.
3. El administrador confirma la eliminación de la categoría.
4. El sistema elimina la categoría seleccionada del catálogo, así como todos los productos asociados a ella.

Postcondiciones: La categoría seleccionada y todos los productos asociados a ella son eliminados del catálogo.

Módulo 4: Pedidos



Caso de Uso 18: Listar Productos

Descripción: Este caso de uso permite al usuario ver una lista de todos los productos disponibles para su compra.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema.

Flujo Principal:

1. El usuario accede a la sección de catálogo de productos.
2. El sistema muestra una lista de todos los productos disponibles junto con su información relevante.

Postcondiciones: El usuario puede ver la lista de productos disponibles para su compra.

Caso de Uso 19: Buscar Productos

Descripción: Este caso de uso permite al usuario buscar productos por nombre, categoría u otro criterio de búsqueda.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema.

Flujo Principal:

1. El usuario accede a la función de búsqueda de productos.
2. El usuario ingresa el criterio de búsqueda.
3. El sistema realiza la búsqueda y muestra los resultados coincidentes.

Postcondiciones: El usuario puede ver los resultados de la búsqueda de productos.

Caso de Uso 20: Ver Detalle del Producto

Descripción: Este caso de uso permite al usuario ver los detalles completos de un producto específico, incluyendo descripción, precio, etc.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema y está viendo la lista de productos.

Flujo Principal:

1. El usuario selecciona un producto de la lista.
2. El sistema muestra los detalles completos del producto seleccionado.

Postcondiciones: El usuario puede ver los detalles completos del producto seleccionado.

Caso de Uso 21: Agregar Producto al Carrito

Descripción: Este caso de uso permite al usuario agregar un producto al carrito de compras para posteriormente realizar un pedido.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema y está viendo los detalles de un producto.

Flujo Principal:

1. El usuario selecciona la opción de agregar el producto al carrito.
2. El sistema añade el producto al carrito de compras del usuario.

Postcondiciones: El producto seleccionado se agrega al carrito de compras del usuario.

Caso de Uso 22: Ver Carrito de Compras

Descripción: Este caso de uso permite al usuario ver los productos que ha agregado al carrito de compras.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema y ha agregado al menos un producto al carrito de compras.

Flujo Principal:

1. El usuario accede a la sección de carrito de compras.
2. El sistema muestra una lista de los productos agregados al carrito junto con su información relevante.

Postcondiciones: El usuario puede ver los productos que ha agregado al carrito de compras.

Caso de Uso 23: Eliminar Producto del Carrito

Descripción: Este caso de uso permite al usuario eliminar un producto del carrito de compras antes de realizar el pedido.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema y ha agregado al menos un producto al carrito de compras.

Flujo Principal:

1. El usuario accede al carrito de compras.
2. El usuario selecciona el producto que desea eliminar.
3. El usuario elige la opción de eliminar el producto del carrito.
4. El sistema elimina el producto seleccionado del carrito de compras.

Postcondiciones: El producto seleccionado es eliminado del carrito de compras del usuario.

Caso de Uso 24: Realizar Pedido

Descripción: Este caso de uso permite al usuario realizar un pedido de los productos agregados al carrito de compras.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema y ha agregado al menos un producto al carrito de compras.

Flujo Principal:

1. El usuario accede al carrito de compras.
2. El usuario selecciona la opción de realizar pedido.
3. El sistema procesa el pedido y lo registra en el historial de pedidos del usuario.
4. El sistema muestra una confirmación del pedido realizado.

Postcondiciones: Se registra el pedido en el historial del usuario y se procede al proceso de pago offline.

Caso de Uso 25: Ver Historial de Pedidos

Descripción: Este caso de uso permite al usuario ver un historial de los pedidos realizados anteriormente.

Actores: Usuario

Precondiciones: El usuario ha iniciado sesión en el sistema y ha realizado al menos un pedido.

Flujo Principal:

1. El usuario accede a la sección de historial de pedidos.
2. El sistema muestra una lista de todos los pedidos realizados por el usuario, incluyendo detalles como fecha, productos, estado del pedido, etc.

Postcondiciones: El usuario puede ver su historial de pedidos anteriores.

Caso de Uso 26: Cambiar Estado de Pedido

Descripción: Este caso de uso permite al administrador cambiar el estado de un pedido en el sistema.

Actores: Administrador

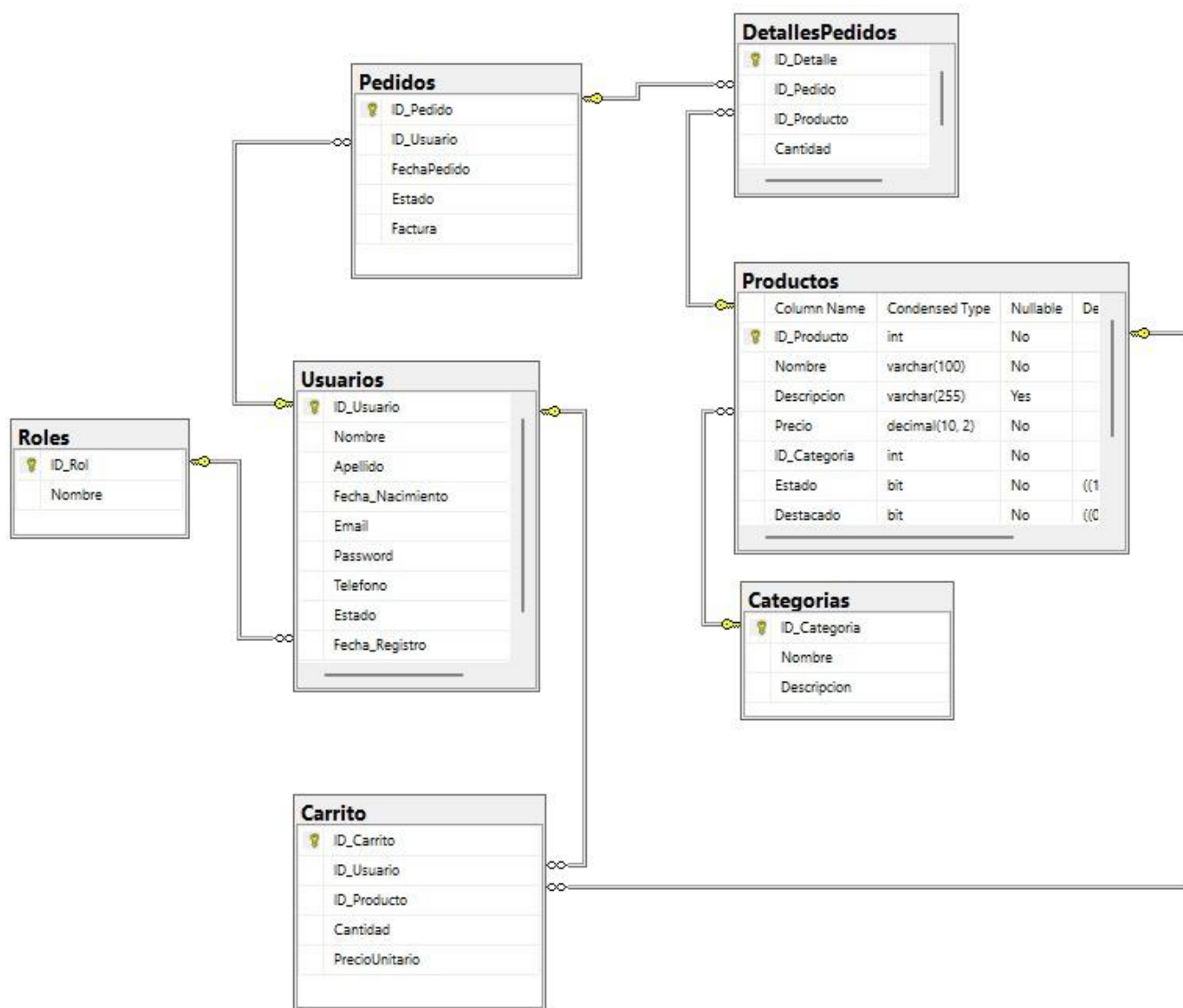
Precondiciones: El administrador ha iniciado sesión en el sistema y tiene permisos para gestionar pedidos.

Flujo Principal:

1. El administrador accede a la función de gestión de pedidos.
2. El administrador selecciona el pedido cuyo estado desea cambiar.
3. El administrador elige el nuevo estado para el pedido.
4. El sistema actualiza el estado del pedido en la base de datos.

Postcondiciones: El estado del pedido seleccionado se actualiza en el sistema.

Base de Datos - Esquema Relacional



Diccionario de Datos

1. Tabla: roles

Características de la tabla			
Nombre de la tabla	Roles		
Descripción de la tabla	Proporciona información sobre los roles o funciones dentro de un sistema		
Campos de Tabla			
Nombre de Campo	Tipo de Campo	Longitud de campo	Breve descripción sobre el campo
ID_Role	int	1,1	Identificador único del Rol
Nombre:	varchar	50	Nombre del rol.
Propósito de la tabla			

Almacenar los diferentes roles que pueden tener los usuarios en el sistema.

2. Tabla: usuarios

Características de la tabla			
Nombre de la tabla	Usuarios		
Descripción de la tabla	Es una parte fundamental de los sistemas de gestión de usuarios y datos personales en bases de datos.		
Campos de Tabla			
Nombre de Campo	Tipo de Campo	Longitud de campo	Breve descripción sobre el campo
ID_Usuario	int	1,1	Identificador único del usuario.
Nombre:	varchar	50	Nombre del usuario.
Apellido	Varchar	50	Apellido del usuario
Fecha de nacimiento	Date	100	Fecha de nacimiento del usuario
Email	Varchar	100	Correo electrónico del usuario
Contraseña	Varbinary	500	Contraseña del usuario (almacenada de forma cifrada.)
Teléfono	Varbinary	500	Número de teléfono del usuario (almacenado de forma cifrada).
Estado	Char	1	Estado del usuario (activo/inactivo).
Fecha de registro	Datetime	30	Fecha y hora en la que se registró el usuario.
ID_Role	Int	20	Identificador del rol del usuario (relacionado con la tabla Roles).
Propósito de la tabla			
Almacenar la información de los usuarios registrados en el sistema, incluyendo su información personal y rol asignado.			

3. Tabla: categorías

Características de la tabla			
Nombre de la tabla	Categorías		
Descripción de la tabla	Proporciona un medio para categorizar y etiquetar datos, lo que facilita la búsqueda, navegación y organización de la información		
Campos de Tabla			
Nombre de Campo	Tipo de Campo	Longitud de campo	Breve descripción sobre el campo
ID_Categoria	int	1,1	Identificador único de la categoría.
Nombre	Varchar	50	Nombre de la categoría
Descripción	varchar	100	Descripción de la categoría
Propósito de la tabla			
Almacenar las categorías disponibles para los productos en el sistema.			

4. Tabla: productos

Características de la tabla			
Nombre de la tabla	Productos		
Descripción de la tabla	es una parte esencial de muchas bases de datos, especialmente en sistemas de gestión de inventario, comercio electrónico y sistemas de gestión de ventas.		
Campos de Tabla			
Nombre de Campo	Tipo de Campo	Longitud de campo	Breve descripción sobre el campo
ID_Producto	int	1,1	Identificador único del producto.
Nombre:	varchar	50	Nombre del producto.
Descripción	Varchar	225	Descripción del producto
Precio	Decimal		Precio del producto
ID_Categoria	In	1,1	Identificador de la categoría del producto (relacionado con la tabla Categorías).
Estado	Varchar	20	Estado del producto (Disponible/No disponible).
Propósito de la tabla			
Almacenar la información de los productos disponibles para su venta en el sistema.			

5. Tabla: pedidos

Características de la tabla			
Nombre de la tabla	Pedidos		
Descripción de la tabla	Es una estructura fundamental en sistemas de gestión de ventas y comercio electrónico, ya que almacena información sobre las transacciones realizadas por los clientes		
Campos de Tabla			
Nombre de Campo	Tipo de Campo	Longitud de campo	Breve descripción sobre el campo
ID_Pedido	int	1,1	Identificador único del pedido.
ID_Usuario:	varchar		Identificador del usuario que realizó el pedido (relacionado con la tabla Usuarios).
Fecha de pedido	Datetime		Fecha y hora en la que se realizo el pedido
Estado	Varchar	20	Estado del pedido (Pendiente/Procesado/Entregado, etc.).
Propósito de la tabla			
Almacenar la información de los pedidos realizados por los usuarios en el sistema.			

6. Tabla: detalles de pedidos

Características de la tabla			
Nombre de la tabla	DetallePedidos		
Descripción de la tabla	Almacena información detallada sobre los productos incluidos en cada pedido.		
Campos de Tabla			
Nombre de Campo	Tipo de Campo	Longitud de campo	Breve descripción sobre el campo
ID_Detalle	int	1,1	Identificador único del detalle del pedido.

ID_Pedido	Int		Identificador del pedido al que pertenece este detalle (relacionado con la tabla Pedidos).
ID_Producto	Int		Identificador del producto incluido en este detalle (relacionado con la tabla Productos).
Cantidad	Int		Cantidad del producto incluido en el detalle del pedido.
PrecioUnitario	Decimal		Precio unitario del producto incluido en el detalle del pedido.
Propósito de la tabla			
Almacenar los detalles de los productos incluidos en cada pedido realizado por los usuarios en el sistema.			

Script Base de datos

```
CREATE DATABASE MiCafesitoEnLinea
Go

USE MiCafesitoEnLinea;
Go

-- Crear tabla para Roles
CREATE TABLE Roles (
    ID_Rol INT PRIMARY KEY IDENTITY(1,1),
    Nombre VARCHAR(50) NOT NULL
);

CREATE TABLE Usuarios (
    ID_Usuario INT IDENTITY(1,1) PRIMARY KEY,
    Nombre VARCHAR(50) NULL,
    Apellido VARCHAR(50) NULL,
    Fecha_Nacimiento DATE,
    Email VARCHAR(100) NOT NULL UNIQUE,
    Password varbinary(500) NULL,
    Telefono varbinary(500) NULL,
    Estado char(1) NOT NULL,
    Fecha_Registro DATETIME DEFAULT GETDATE() NOT NULL,
    ID_Rol INT,
    FOREIGN KEY (ID_Rol) REFERENCES Roles(ID_Rol)
);

-- Crear tabla para Categorías de Productos
CREATE TABLE Categorías (
    ID_Categoría INT PRIMARY KEY IDENTITY(1,1),
    Nombre VARCHAR(50) NOT NULL,
    Descripción VARCHAR(100)
);

-- Crear tabla para Productos
CREATE TABLE Productos (
    ID_Producto INT PRIMARY KEY IDENTITY(1,1),
    Nombre VARCHAR(100) NOT NULL,
    Descripción VARCHAR(255),
    Precio DECIMAL(10, 2) NOT NULL,
    ID_Categoría INT,
    Estado VARCHAR(20) DEFAULT 'Disponible',
    FOREIGN KEY (ID_Categoría) REFERENCES Categorías(ID_Categoría)
```

```
);

-- Crear tabla para Pedidos
CREATE TABLE Pedidos (
    ID_Pedido INT PRIMARY KEY IDENTITY(1,1),
    ID_Usuario INT,
    FechaPedido DATETIME DEFAULT GETDATE(),
    Estado VARCHAR(20) DEFAULT 'Pendiente',
    Factura VARCHAR(50),
    FOREIGN KEY (ID_Usuario) REFERENCES Usuarios(ID_Usuario)
);

-- Crear tabla para Detalles de Pedidos
CREATE TABLE DetallesPedidos (
    ID_Detalle INT PRIMARY KEY IDENTITY(1,1),
    ID_Pedido INT,
    ID_Producto INT,
    Cantidad INT,
    PrecioUnitario DECIMAL(10, 2),
    FOREIGN KEY (ID_Pedido) REFERENCES Pedidos(ID_Pedido),
    FOREIGN KEY (ID_Producto) REFERENCES Productos(ID_Producto)
);
```

Scripts Procedimientos Almacenados

Módulo de Autenticación

```
-- SP para validar usuario
CREATE PROCEDURE dbo.SP_ValidarUsuario
    @Email VARCHAR(100),
    @Password VARCHAR(25),
    @Patron VARCHAR(100)
AS
BEGIN
    DECLARE @ID_Usuario INT;

    SELECT @ID_Usuario = ID_Usuario
    FROM Usuarios
    WHERE Email = @Email AND CONVERT(VARCHAR(25), DECRYPTBYPASSPHRASE(@Patron, Password)) =
@Password;

    IF @ID_Usuario IS NOT NULL
    BEGIN
        SELECT @ID_Usuario AS ID_Usuario;
    END
    ELSE
    BEGIN
        SELECT 0 AS ID_Usuario;
    END;
END;
GO

-- SP para resetear contrasea
CREATE PROCEDURE dbo.SP_ResetearContrasena
    @Email VARCHAR(100),
    @PasswordActual VARCHAR(25),
    @NuevaContrasena VARCHAR(25),
```

```
@VerificarContrasena VARCHAR(25),
@Patron VARCHAR(100)
AS
BEGIN
    DECLARE @ID_Usuario INT;
    IF @NuevaContrasena = @VerificarContrasena
        BEGIN
            SELECT @ID_Usuario = ID_Usuario
            FROM Usuarios
            WHERE Email = @Email AND CONVERT(VARCHAR(100), DECRYPTBYPASSPHRASE(@Patron, Password)) =
@PasswordActual;

            IF @ID_Usuario IS NOT NULL
                BEGIN
                    DECLARE @NuevaContrasenaEncriptada VARBINARY(500);
                    SET @NuevaContrasenaEncriptada = ENCRYPTBYPASSPHRASE(@Patron, @NuevaContrasena);

                    UPDATE Usuarios
                    SET Password = @NuevaContrasenaEncriptada
                    WHERE ID_Usuario = @ID_Usuario;

                    SELECT 'Contraseña actualizada exitosamente.' AS message, 'exito' as tipo;
                END
            ELSE
                BEGIN
                    SELECT 'Las contraseñas no coinciden.' AS message, 'error' as tipo;
                END;
        END
    ELSE
        BEGIN
            SELECT 'Contraseña actual inválida.' AS message, 'error' as tipo;
        END;
END;
GO
```

Modulo de Usuarios

```
USE MiCafesitoEnlinea;
GO

-- SP para insertar usuario
CREATE PROCEDURE dbo.SP_CrearUsuario
    @Nombre VARCHAR(50),
    @Apellido VARCHAR(50),
    @FechaNacimiento DATE,
    @Email VARCHAR(100),
    @Password VARCHAR(25),
    @Telefono VARCHAR(25),
    @Estado CHAR(1),
    @ID_Rol INT,
    @Patron VARCHAR(100)
AS
BEGIN
    DECLARE @PasswordEncriptado VARBINARY(500);
    DECLARE @TelefonoEncriptado VARBINARY(500);
    DECLARE @new_id INT;

    SET @PasswordEncriptado = ENCRYPTBYPASSPHRASE(@Patron, @Password);
    SET @TelefonoEncriptado = ENCRYPTBYPASSPHRASE(@Patron, @Telefono);
```

```
INSERT INTO Usuarios (Nombre, Apellido, Fecha_Nacimiento, Email, Password, Telefono, ID_Rol,
Estado)
VALUES (
    @Nombre,
    @Apellido,
    @FechaNacimiento,
    @Email,
    @PasswordEncriptado,
    @TelefonoEncriptado,
    @ID_Rol,
    @Estado
);

SET @new_id = SCOPE_IDENTITY();

IF @new_id IS NOT NULL
    SELECT @new_id AS ID_Usuario;
ELSE
    SELECT 0 AS ID_Usuario;
END;
GO

-- SP para leer usuario
CREATE PROCEDURE dbo.SP_LeerUsuario
    @ID_Usuario INT,
    @Patron VARCHAR(100)
AS
BEGIN
    SELECT
        u.ID_Usuario,
        u.Nombre,
        u.Apellido,
        u.Fecha_Nacimiento,
        u.Email,
        CONVERT(VARCHAR(25), DECRYPTBYPASSPHRASE(@Patron, u.Telefono)) AS Telefono,
        u.ID_Rol,
        r.Nombre,
        u.Estado,
        u.Fecha_Registro
    FROM
        Usuarios as u
        Inner join dbo.Roles as r on u.ID_Rol = r.ID_rol
    WHERE
        ID_Usuario = @ID_Usuario;
END;
GO

CREATE PROCEDURE dbo.SP_ActualizarUsuario
    @ID_Usuario INT,
    @Nombre VARCHAR(50),
    @Apellido VARCHAR(50),
    @FechaNacimiento DATE,
    @Telefono VARCHAR(25),
    @Estado CHAR(1),
    @ID_Rol INT,
    @Patron VARCHAR(100)
AS
BEGIN
    DECLARE @TelefonoEncriptado VARBINARY(500);

    SET @TelefonoEncriptado = ENCRYPTBYPASSPHRASE(@Patron, @Telefono);
```

```
UPDATE Usuarios
SET Nombre = @Nombre,
    Apellido = @Apellido,
    Fecha_Nacimiento = @FechaNacimiento,
    Telefono = @TelefonoEncriptado,
    ID_Rol = @ID_Rol,
    Estado = @Estado
WHERE ID_Usuario = @ID_Usuario;

IF @@ROWCOUNT > 0
    SELECT 1 AS Modificado;      -- Indicar que se realizó la modificación
ELSE
    SELECT 0 AS Modificado;      -- Indicar que no se encontró el usuario para modificar
END;
GO

-- SP para insertar un nuevo rol
CREATE PROCEDURE SP_InsertarRol
    @Nombre VARCHAR(50)
AS
BEGIN
    INSERT INTO Roles (Nombre)
    VALUES (@Nombre);
END;
GO

-- SP para obtener los detalles de un rol por su ID
CREATE PROCEDURE SP_ObtenerRol
    @ID_Rol INT
AS
BEGIN
    SELECT *
    FROM Roles
    WHERE ID_Rol = @ID_Rol;
END;
GO

-- SP para actualizar los detalles de un rol existente
CREATE PROCEDURE SP_ActualizarRol
    @ID_Rol INT,
    @Nombre VARCHAR(50)
AS
BEGIN
    UPDATE Roles
    SET Nombre = @Nombre
    WHERE ID_Rol = @ID_Rol;
END;
GO

-- SP para eliminar un rol por su ID
CREATE PROCEDURE SP_EliminarRol
    @ID_Rol INT
AS
BEGIN
    DELETE FROM Roles
    WHERE ID_Rol = @ID_Rol;
END;
GO

-- SP para listar todos los roles
CREATE PROCEDURE SP_ListarRoles
```

```
AS
BEGIN
    SELECT *
    FROM Roles;
END;
GO
```

Módulo de Catálogo de Productos

```
-- CATEGORIAS
-- SP para insertar una nueva categoría
CREATE PROCEDURE SP_InsertarCategoria
    @Nombre VARCHAR(50),
    @Descripcion VARCHAR(100)
AS
BEGIN
    INSERT INTO Categorias (Nombre, Descripcion)
    VALUES (@Nombre, @Descripcion);
END;
GO

-- SP para obtener los detalles de una categoría por su ID
CREATE PROCEDURE SP_ObtenerCategoria
    @ID_Categoria INT
AS
BEGIN
    SELECT *
    FROM Categorias
    WHERE ID_Categoria = @ID_Categoria;
END;
GO

-- SP para actualizar los detalles de una categoría existente
CREATE PROCEDURE SP_ActualizarCategoria
    @ID_Categoria INT,
    @Nombre VARCHAR(50),
    @Descripcion VARCHAR(100)
AS
BEGIN
    UPDATE Categorias
    SET Nombre = @Nombre,
        Descripcion = @Descripcion
    WHERE ID_Categoria = @ID_Categoria;
END;
GO

-- SP para eliminar una categoría por su ID
CREATE PROCEDURE SP_EliminarCategoria
    @ID_Categoria INT
AS
BEGIN
    DELETE FROM Categorias
    WHERE ID_Categoria = @ID_Categoria;
END;
GO

-- SP para listar todas las categorías
```

```
CREATE PROCEDURE SP_ListarCategorias
AS
BEGIN
    SELECT *
    FROM Categorias;
END;
GO

-- PRODUCTOS
-- SP para insertar un nuevo producto
CREATE PROCEDURE SP_InsertarProducto
    @Nombre VARCHAR(100),
    @Descripcion VARCHAR(255),
    @Precio DECIMAL(10, 2),
    @ID_Categoria INT
AS
BEGIN
    INSERT INTO Productos (Nombre, Descripcion, Precio, ID_Categoria)
    VALUES (@Nombre, @Descripcion, @Precio, @ID_Categoria);
END;
GO

-- SP para obtener los detalles de un producto por su ID
CREATE PROCEDURE SP_ObtenerProducto
    @ID_Producto INT
AS
BEGIN
    SELECT *
    FROM Productos
    WHERE ID_Producto = @ID_Producto;
END;
GO

-- Búsqueda de productos por nombre, descripcion, precio y/o categoría
CREATE PROCEDURE SP_BuscarProductoPorAtributo
    @CriterioBusqueda VARCHAR(100),
    @TipoBusqueda VARCHAR(20)
AS
BEGIN
    IF @TipoBusqueda = 'Nombre' -- Si se busca por nombre
    BEGIN
        SELECT * FROM Productos WHERE Nombre LIKE '%' + @CriterioBusqueda + '%';
    END
    ELSE IF @TipoBusqueda = 'Descripcion' -- Si se busca por descripción
    BEGIN
        SELECT * FROM Productos WHERE Descripcion LIKE '%' + @CriterioBusqueda + '%';
    END
    ELSE IF @TipoBusqueda = 'Precio' -- Si se busca por precio
    BEGIN
        SELECT * FROM Productos WHERE Precio = CAST(@CriterioBusqueda AS DECIMAL(10, 2));
    END
    ELSE IF @TipoBusqueda = 'ID_Categoria' -- Si se busca por ID de categoría
    BEGIN
        SELECT * FROM Productos WHERE ID_Categoria = CAST(@CriterioBusqueda AS INT);
    END
    ELSE
    BEGIN
        RAISERROR ('Tipo de búsqueda no válido.', 16, 1);
    END
END;
GO
```



```
-- SP para actualizar los detalles de un producto existente
CREATE PROCEDURE SP_ActualizarProducto
    @ID_Producto INT,
    @Nombre VARCHAR(100),
    @Descripcion VARCHAR(255),
    @Precio DECIMAL(10, 2),
    @ID_Categoria INT
AS
BEGIN
    UPDATE Productos
    SET Nombre = @Nombre,
        Descripcion = @Descripcion,
        Precio = @Precio,
        ID_Categoria = @ID_Categoria
    WHERE ID_Producto = @ID_Producto;
END;
GO

-- SP para eliminar un producto por su ID
CREATE PROCEDURE SP_EliminarProducto
    @ID_Producto INT
AS
BEGIN
    DELETE FROM Productos
    WHERE ID_Producto = @ID_Producto;
END;
GO

-- SP para listar todos los productos
CREATE PROCEDURE SP_ListarProductos
AS
BEGIN
    SELECT *
    FROM Productos;
END;
GO
```

Módulo de Pedidos

```
-- PEDIDOS
-- SP para insertar un nuevo pedido
CREATE PROCEDURE SP_InsertarPedido
    @ID_Usuario INT,
    @Factura VARCHAR(50)
AS
BEGIN
    INSERT INTO Pedidos (ID_Usuario, Factura)
    VALUES (@ID_Usuario, @Factura);
END;
GO

-- SP para obtener los detalles de un pedido por su ID
CREATE PROCEDURE SP_ObtenerPedido
    @ID_Pedido INT
AS
BEGIN
```

```
SELECT *
FROM Pedidos
WHERE ID_Pedido = @ID_Pedido;
END;
GO

-- SP para actualizar los detalles de un pedido existente
CREATE PROCEDURE SP_ActualizarPedido
    @ID_Pedido INT,
    @ID_Usuario INT,
    @Factura VARCHAR(50)
AS
BEGIN
    UPDATE Pedidos
    SET ID_Usuario = @ID_Usuario,
        Factura = @Factura
    WHERE ID_Pedido = @ID_Pedido;
END;
GO

-- SP para eliminar un pedido y sus detalles asociados por su ID
CREATE PROCEDURE SP_EliminarPedido
    @ID_Pedido INT
AS
BEGIN
    BEGIN TRANSACTION; -- Inicia una transacción

    -- Eliminar detalles de pedido asociados al pedido
    DELETE FROM DetallesPedidos
    WHERE ID_Pedido = @ID_Pedido;

    -- Eliminar el pedido
    DELETE FROM Pedidos
    WHERE ID_Pedido = @ID_Pedido;

    COMMIT; -- Confirma la transacción
END;
GO

-- SP para listar todos los detalles de un pedido
CREATE PROCEDURE SP_ListarDetallesPedido
    @ID_Pedido INT
AS
BEGIN
    SELECT *
    FROM DetallesPedidos
    WHERE ID_Pedido = @ID_Pedido;
END;
GO

-- SP para insertar un nuevo detalle de pedido
CREATE PROCEDURE SP_InsertarDetallePedido
    @ID_Pedido INT,
    @ID_Producto INT,
    @Cantidad INT,
    @PrecioUnitario DECIMAL(10, 2)
AS
BEGIN
    INSERT INTO DetallesPedidos (ID_Pedido, ID_Producto, Cantidad, PrecioUnitario)
    VALUES (@ID_Pedido, @ID_Producto, @Cantidad, @PrecioUnitario);
END;
GO
```

```
-- SP para actualizar los detalles de un detalle de pedido existente
CREATE PROCEDURE SP_ActualizarDetallePedido
    @ID_Detalle INT,
    @ID_Pedido INT,
    @ID_Producto INT,
    @Cantidad INT,
    @PrecioUnitario DECIMAL(10, 2)
AS
BEGIN
    UPDATE DetallesPedidos
    SET ID_Pedido = @ID_Pedido,
        ID_Producto = @ID_Producto,
        Cantidad = @Cantidad,
        PrecioUnitario = @PrecioUnitario
    WHERE ID_Detalle = @ID_Detalle;
END;
GO

-- SP para eliminar un detalle de pedido por su ID
CREATE PROCEDURE SP_EliminarDetallePedido
    @ID_Detalle INT
AS
BEGIN
    DELETE FROM DetallesPedidos
    WHERE ID_Detalle = @ID_Detalle;
END;
GO
```

Capa de Servicios

DataContracts

WCFMiCafesitoServices/App_Code/DataContracts/Cart.cs

```
using System;
using System.Runtime.Serialization;

[DataContract]
public class Cart
{
    [DataMember]
    public int ID_Carrito { get; set; }

    [DataMember]
    public int ID_Usuario { get; set; }

    [DataMember]
    public int ID_Producto { get; set; }

    [DataMember]
    public double PrecioUnitario { get; set; }

    [DataMember]
    public int Cantidad { get; set; }
}
```

```
    public DateTime Fecha { get; set; }  
}
```

WCFMiCafesitoServices/App_Code/DataContracts/Category.cs

```
using System.Runtime.Serialization;  
  
[DataContract]  
public class Category  
{  
  
    [DataMember]  
    public int ID_Categoria { get; set; }  
  
    [DataMember]  
    public string Nombre { get; set; }  
  
    [DataMember]  
    public string Descripcion { get; set; }  
}
```

WCFMiCafesitoServices/App_Code/DataContracts/Order.cs

```
using System;  
using System.Runtime.Serialization;  
  
[DataContract]  
public class Order  
{  
  
    [DataMember]  
    public int ID_Pedido { get; set; }  
  
    [DataMember]  
    public int ID_Usuario { get; set; }  
  
    [DataMember]  
    public DateTime FechaPedido { get; set; }  
  
    [DataMember]  
    public string Estado { get; set; }  
  
    [DataMember]  
    public string Factura { get; set; }  
  
    [DataMember]  
    public double SubTotal { get; set; }  
}
```

WCFMiCafesitoServices/App_Code/DataContracts/OrderDetail.cs

```
using System.Runtime.Serialization;  
  
[DataContract]  
public class OrderDetail  
{
```

```
[DataMember]
public int ID_Detalle { get; set; }

[DataMember]
public int ID_Pedido { get; set; }

[DataMember]
public int ID_Producto { get; set; }

[DataMember]
public int Cantidad { get; set; }

[DataMember]
public double PrecioUnitario { get; set; }

[DataMember]
public double Total { get; set; }

}
```

WCFMiCafesitoServices/App_Code/DataContracts/Product.cs

```
using System.Runtime.Serialization;

[DataContract]
public class Product
{
    [DataMember]
    public int ID_Producto { get; set; }

    [DataMember]
    public string Nombre { get; set; }

    [DataMember]
    public string Descripcion { get; set; }

    [DataMember]
    public double Precio { get; set; }

    [DataMember]
    public int ID_Categoria { get; set; }

    [DataMember]
    public string Categoria { get; set; }

    [DataMember]
    public string ImageUrl { get; set; }

}
```

WCFMiCafesitoServices/App_Code/DataContracts/Role.cs

```
using System.Runtime.Serialization;

[DataContract]
```

```
public class Role
{
    [DataMember]
    public int ID_Rol { get; set; }

    [DataMember]
    public string Nombre { get; set; }
}
```

WCFMiCafesitoServices/App_Code/DataContracts/SearchCriteria.cs

```
using System.Runtime.Serialization;

[DataContract]
public class SearchCriteria
{
    [DataMember]
    public CriteriaType Criteria { get; set; }

    [DataMember]
    public string Value { get; set; }
}

public enum CriteriaType
{
    Id,
    Email,
    Name,
    Estatus,
    Role
}
```

WCFMiCafesitoServices/App_Code/DataContracts/User.cs

```
using System;
using System.Runtime.Serialization;

[DataContract]
public class User
{
    [DataMember]
    public int ID_Usuario { get; set; }

    [DataMember]
    public string Nombre { get; set; }

    [DataMember]
    public string Apellido { get; set; }

    [DataMember]
    public DateTime Fecha_Nacimiento { get; set; }

    [DataMember]
    public string Email { get; set; }

    [DataMember]
    public string Password { get; set; }
}
```

```
[DataMember]
public string Telefono { get; set; }

[DataMember]
public string Estado { get; set; }

[DataMember]
public DateTime Fecha_Registro { get; set; }

[DataMember]
public int ID_Rol { get; set; }

[DataMember]
public string NombreRol { get; set; }
}
using System;
using System.Runtime.Serialization;

[DataContract]
public class User
{
    [DataMember]
    public int ID_Usuario { get; set; }

    [DataMember]
    public string Nombre { get; set; }

    [DataMember]
    public string Apellido { get; set; }

    [DataMember]
    public DateTime Fecha_Nacimiento { get; set; }

    [DataMember]
    public string Email { get; set; }

    [DataMember]
    public string Password { get; set; }

    [DataMember]
    public string Telefono { get; set; }

    [DataMember]
    public string Estado { get; set; }

    [DataMember]
    public DateTime Fecha_Registro { get; set; }

    [DataMember]
    public int ID_Rol { get; set; }

    [DataMember]
    public string NombreRol { get; set; }
}
```

ServiceContracts

WCFMiCafesitoServices/App_Code/ServiceContracts/ICartService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.ServiceModel;
using System.Web;

/// <summary>
/// Summary description for ICartService
/// </summary>
///
namespace MiCafesito
{
    [ServiceContract]
    public interface ICartService
    {

        [OperationContract]
        void AddCartItem(Cart cart);

        [OperationContract]
        List<Cart> GetCartItemsByUserId(int id);

        [OperationContract]
        void UpdateCartItemById(int id, int quantity, double unitprice);

        [OperationContract]
        void DeleteCartItem(int id);

        [OperationContract]
        void DeleteCartItemsByUserId(int id);

        [OperationContract]
        void UpdateCartItemQuantityById(int id, int quantity);

    }
}
```

WCFMiCafesitoServices/App_Code/ServiceContracts/ICategoryService.cs

```
using System.Collections.Generic;
using System.ServiceModel;

namespace MiCafesito
{
    [ServiceContract]
    public interface ICategoryService
    {

        [OperationContract]
        List<Category> GetAllCategories();

        [OperationContract]
        Category GetCategoryById(int id);

    }
}
```



```
[OperationContract]
void UpdateCategory(Category category);

[OperationContract]
void DeleteCategory(int id);

[OperationContract]
void AddCategory(Category category);
}
}
```

WCFMiCafesitoServices/App_Code/ServiceContracts/IOrderDetailsService.cs

```
using System.Collections.Generic;
using System.ServiceModel;

namespace MiCafesito
{
    [ServiceContract]
    public interface IOrderDetailsService
    {
        [OperationContract]
        List<OrderDetail> GetAllOrderDetailById(int id);

        [OperationContract]
        void UpdateOrderDetail(OrderDetail orderDetails);

        [OperationContract]
        void DeleteOrderDetail(int id);

        [OperationContract]
        void AddOrderDetail(OrderDetail orderDetails);

        [OperationContract]
        void DeleteOrderDetailById(int id);
    }
}
```

WCFMiCafesitoServices/App_Code/ServiceContracts/IOrderService.cs

```
using System.Collections.Generic;
using System.ServiceModel;

namespace MiCafesito
{
    [ServiceContract]
    public interface IOrderService
    {
        [OperationContract]
        List<Order> GetAllOrders();

        [OperationContract]
        Order GetOrderById(int id);

        [OperationContract]
        void UpdateOrder(Order order);

        [OperationContract]
    }
```

```
void DeleteOrder(int id);

[OperationContract]
int AddOrder(Order order);

[OperationContract]
List<Order> GetAllOrdersByUserId(int id);
}
}
```

WCFMiCafesitoServices/App_Code/ServiceContracts/IProductService.cs

```
using System.Collections.Generic;
using System.ServiceModel;

namespace MiCafesito
{
    [ServiceContract]
    public interface IProductService
    {
        [OperationContract]
        List<Product> GetAllProducts();

        [OperationContract]
        Product GetProductById(int id);

        [OperationContract]
        void UpdateProduct(Product productos);

        [OperationContract]
        void DeleteProduct(int id);

        [OperationContract]
        void AddProduct(Product productos);

        [OperationContract]
        List<Product> GetAllProductsByCategoryId(int id);

        [OperationContract]
        List<Product> GetProductsFeatured();

        [OperationContract]
        List<Product> GetProductsByName(string criteria);
    }
}
```

WCFMiCafesitoServices/App_Code/ServiceContracts/IRolesService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.ServiceModel;
using System.Web;

namespace MiCafesito
{
    [ServiceContract]
```

```
public interface IRolesService
{
    [OperationContract]
    List<Role> GetAllRoles();
}
```

WCFMiCafesitoServices/App_Code/ServiceContracts/IUserService.cs

```
using System.Collections.Generic;
using System.ServiceModel;

namespace MiCafesito
{
    [ServiceContract]
    public interface IUserService
    {
        [OperationContract]
        List<User> GetAllUsers();

        [OperationContract]
        User GetUserById(int id);

        [OperationContract]
        int Login(string email, string password);

        [OperationContract]
        void UpdateUser(User user);

        [OperationContract]
        void DeleteUser(int id);

        [OperationContract]
        void AddUser(User user);

        [OperationContract]
        void ResetPassword(string email, string password, string newPassword,
string confirmPassword);
    }
}
```

Implementación de servicios

WCFMiCafesitoServices/App_Code/Services/CartService.cs

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;

namespace MiCafesito
{
    public class CartService : ICartService
    {
        private CustomConfigurationManager _config;
        private SqlConnection connection;
        public CartService()
        {

```

```
{
    _config = new CustomConfigurationManager();
    connection = new SqlConnection(_config.ConnectionString);
}

public void AddCartItem(Cart cart)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_InsertarCarrito",
connection))
        {
            command.CommandType = System.Data.CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Usuario",
cart.ID_Usuario);
            command.Parameters.AddWithValue("@ID_Producto",
cart.ID_Producto);
            command.Parameters.AddWithValue("@Cantidad", cart.Cantidad);
            command.Parameters.AddWithValue("@PrecioUnitario",
cart.PrecioUnitario);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo agregar
el item al carrito"); }
    finally { connection.Close(); }
}

public void DeleteCartItem(int id)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_EliminarCarrito",
connection))
        {
            command.CommandType = System.Data.CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Carrito", id);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar el item del carrito"); }
    finally { connection.Close(); }
}

public void DeleteCartItemsByUserId(int id)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_EliminarCarritoByUserID", connection))
        {
            command.CommandType = System.Data.CommandType.StoredProcedure;
```

```
        command.Parameters.AddWithValue("@ID_Usuario", id);

        connection.Open();
        command.ExecuteNonQuery();
    }
}
catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar los items del carrito"); }
finally
{
    connection.Close();
}
}

public List<Cart> GetCartItemsByUserId(int id)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ListarCarritoByUserID", connection))
        {
            command.CommandType = System.Data.CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Usuario", id);

            connection.Open();
            using (SqlDataReader reader = command.ExecuteReader())
            {
                List<Cart> cartItems = new List<Cart>();

                while (reader.Read())
                {
                    Cart cart = new Cart();
                    cart.ID_Carrito =
Convert.ToInt32(reader["ID_Carrito"]);
                    cart.ID_Usuario =
Convert.ToInt32(reader["ID_Usuario"]);
                    cart.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
                    cart.Cantidad = Convert.ToInt32(reader["Cantidad"]);
                    cart.PrecioUnitario =
Convert.ToDouble(reader["PrecioUnitario"]);

                    cartItems.Add(cart);
                }

                return cartItems;
            }
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo obtener
los items del carrito"); }
    finally { connection.Close(); }
}

public void UpdateCartItemById(int id, int quantity, double unitprice)
```

```
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_ActualizarCarrito",
connection))
        {
            command.CommandType = System.Data.CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Carrito", id);
            command.Parameters.AddWithValue("@Cantidad", quantity);
            command.Parameters.AddWithValue("@PrecioUnitario", unitprice);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
actualizar el item del carrito"); }
    finally { connection.Close(); }
}

public void UpdateCartItemQuantityById(int id, int quantity)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ActualizarItemCarritoCantidad", connection))
        {
            command.CommandType = System.Data.CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Carrito", id);
            command.Parameters.AddWithValue("@Cantidad", quantity);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
actualizar la cantidad del item del carrito"); }
    finally { connection.Close(); }
}
}
```

WCFMiCafesitoServices/App_Code/Services/CategoryService.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;

namespace MiCafesito
{
    public class CategoryService : ICategoryService
    {
        private CustomConfigurationManager _config;
        private SqlConnection connection;
```

```
public CategoryService()
{
    _config = new CustomConfigurationManager();
    connection = new SqlConnection(_config.ConnectionString);
}

public void AddCategory(Category category)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_InsertarCategoria",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@Nombre", category.Nombre);
            command.Parameters.AddWithValue("@Descripcion",
category.Descripcion);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo agregar
la categoria"); }
    finally { connection.Close(); }
}

public void DeleteCategory(int id)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_EliminarCategoria",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Categoria", id);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar la categoria"); }
    finally { connection.Close(); }
}

public List<Category> GetAllCategories()
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_ListarCategorias",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;
```

```
        connection.Open();
        using (SqlDataReader reader = command.ExecuteReader())
        {
            List<Category> categories = new List<Category>();

            while (reader.Read())
            {
                Category category = new Category();
                category.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);
                category.Nombre = reader["Nombre"].ToString();
                category.Descripcion =
reader["Descripcion"].ToString();

                categories.Add(category);
            }

            return categories;
        }
    }
}

catch (Exception ex) { throw new ArgumentException("No se pudieron
obtener las categorias"); }
finally { connection.Close(); }
}

public Category GetCategoryById(int id)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_ObtenerCategoria",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Categoria", id);

            connection.Open();
            using (SqlDataReader reader = command.ExecuteReader())
            {
                Category category = new Category();

                while (reader.Read())
                {
                    category.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);
                    category.Nombre = reader["Nombre"].ToString();
                    category.Descripcion =
reader["Descripcion"].ToString();
                }

                return category;
            }
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la categoria"); }
    finally { connection.Close(); }
```



```
        return null;
    }

    public void UpdateCategory(Category category)
    {
        try
        {
            using (SqlCommand command = new
SqlCommand("SP_ActualizarCategoria", connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@ID_Categoria",
category.ID_Categoria);
                command.Parameters.AddWithValue("@Nombre", category.Nombre);
                command.Parameters.AddWithValue("@Descripcion",
category.Descripcion);

                connection.Open();
                command.ExecuteNonQuery();
            }
        }
        catch (Exception ex) { throw new ArgumentException("No se pudo
actualizar la categoria"); }
        finally { connection.Close(); }
    }
}
```

WCFMiCafesitoServices/App_Code/Services/OrderDetailService.cs

```
using System;
using System.Activities.Expressions;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;

namespace MiCafesito
{
    public class ProductDetailService : IOrderDetailsService
    {
        private CustomConfigurationManager _config;
        private SqlConnection connection;

        public ProductDetailService()
        {
            _config = new CustomConfigurationManager();
            connection = new SqlConnection(_config.ConnectionString);
        }

        public void AddOrderDetail(OrderDetail orderDetails)
        {
            try
            {
                using (SqlCommand command = new
SqlCommand("SP_InsertarDetallePedido", connection))
                {
                    command.CommandType = CommandType.StoredProcedure;
                }
            }
        }
    }
}
```

```
        command.Parameters.AddWithValue("@ID_Pedido",
orderDetails.ID_Pedido);
        command.Parameters.AddWithValue("@ID_Producto",
orderDetails.ID_Producto);
        command.Parameters.AddWithValue("@Cantidad",
orderDetails.Cantidad);
        command.Parameters.AddWithValue("@PrecioUnitario",
orderDetails.PrecioUnitario);

        connection.Open();
        command.ExecuteNonQuery();
    }
}
catch (Exception ex) { throw new ArgumentException("No se pudo agregar
el detalle del pedido"); }
finally { connection.Close(); }
}

public void DeleteOrderDetail(int id)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_EliminarDetallePedido", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Detalle", id);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar el detalle del pedido"); }
    finally { connection.Close(); }
}

public void DeleteOrderDetailById(int id)
{
    // SP_EliminarDetallesPedido
    // ID_Pedido INT

    try
    {
        using (SqlCommand command = new
SqlCommand("SP_EliminarDetallesPedido", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Pedido", id);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar los detalles del pedido"); }
```

```
        finally { connection.Close(); }
    }

    public List<OrderDetail> GetAllOrderDetailByOrderId(int id)
    {
        try
        {
            using (SqlCommand command = new
SqlCommand("SP_ListarDetallePedido", connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@ID_Pedido", id);

                connection.Open();
                using (SqlDataReader reader = command.ExecuteReader())
                {
                    List<OrderDetail> detallePedidos = new
List<OrderDetail>();

                    while (reader.Read())
                    {
                        OrderDetail orderDetail = new OrderDetail();
                        orderDetail.ID_Detalle =
Convert.ToInt32(reader["ID_Detalle"]);
                        orderDetail.ID_Pedido =
Convert.ToInt32(reader["ID_Pedido"]);
                        orderDetail.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
                        orderDetail.Cantidad =
Convert.ToInt32(reader["Cantidad"]);
                        orderDetail.PrecioUnitario =
Convert.ToDouble(reader["PrecioUnitario"]);
                        orderDetail.Total = Convert.ToDouble(reader["Total"]);

                        detallePedidos.Add(orderDetail);
                    }

                    return detallePedidos;
                }
            }
        }
        catch (Exception ex) { throw new ArgumentException("No se pudieron
obtener el detalle del pedido"); }
        finally { connection.Close(); }
    }

    public void UpdateOrderDetail(OrderDetail orderDetails)
    {
        try
        {
            using (SqlCommand command = new
SqlCommand("SP_ActualizarDetallePedido", connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@ID_Detalle",
orderDetails.ID_Detalle);
                command.Parameters.AddWithValue("@ID_Pedido",
```

```
orderDetails.ID_Pedido);
            command.Parameters.AddWithValue("@ID_Producto",
orderDetails.ID_Producto);
            command.Parameters.AddWithValue("@Cantidad",
orderDetails.Cantidad);
            command.Parameters.AddWithValue("@PrecioUnitario",
orderDetails.PrecioUnitario);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
actualizar el detalle del pedido"); }
    finally { connection.Close(); }
}
}
```

WCFMiCafesitoServices/App_Code/Services/OrderService.cs

```
using System;
using System.Data.SqlClient;
using System.Data;
using System.Collections.Generic;

namespace MiCafesito
{
    public class OrderService : IOrderService
    {
        private CustomConfigurationManager _config;
        private SqlConnection connection;

        public OrderService()
        {
            _config = new CustomConfigurationManager();
            connection = new SqlConnection(_config.ConnectionString);
        }

        public int AddOrder(Order order)
        {
            try
            {
                using (SqlCommand command = new SqlCommand("SP_InsertarPedido",
connection))
                {
                    command.CommandType = CommandType.StoredProcedure;

                    command.Parameters.AddWithValue("@ID_Usuario",
order.ID_Usuario);
                    command.Parameters.AddWithValue("@FechaPedido",
order.FechaPedido);
                    command.Parameters.AddWithValue("@SubTotal", order.SubTotal);
                    command.Parameters.AddWithValue("@Estado", order.Estado);

                    // Add output parameter for inserted ID
                    SqlParameter outputParameter = new SqlParameter("@PedidoID",
SqlDbType.Int);
```

```
        outputParameter.Direction = ParameterDirection.Output;
        command.Parameters.Add(outputParameter);

        connection.Open();
        command.ExecuteNonQuery(); // Use ExecuteNonQuery for INSERT

statements

        // Retrieve inserted ID from output parameter
        int insertedOrderID = Convert.ToInt32(outputParameter.Value);

        return insertedOrderID;
    }
}
catch (Exception ex) { throw new ArgumentException("No se pudo agregar
el pedido"); }
finally { connection.Close(); }
}

public void DeleteOrder(int id)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_EliminarPedido",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Pedido", id);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar el pedido"); }
    finally { connection.Close(); }
}

public List<Order> GetAllOrders()
{
    List<Order> orders = new List<Order>();

    try
    {
        using (SqlCommand command = new SqlCommand("SP_ListarPedidos",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            connection.Open();

            using (SqlDataReader reader = command.ExecuteReader())
            {
                while (reader.Read())
                {
                    Order order = new Order();
```

```
                order.ID_Pedido =
Convert.ToInt32(reader["ID_Pedido"]);
                order.ID_Usuario =
Convert.ToInt32(reader["ID_Usuario"]);
                order.FechaPedido =
Convert.ToDateTime(reader["FechaPedido"]);
                order.Estado = reader["Estado"].ToString();
                order.Factura = reader["Factura"] == DBNull.Value ?
"N/A" : reader["Factura"].ToString();
                order.SubTotal = reader["SubTotal"] == DBNull.Value ?
0 : Convert.ToDouble(reader["SubTotal"]);

                orders.Add(order);
            }
        }
    }
}
catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de pedidos"); }
finally { connection.Close(); }

return orders;
}

public List<Order> GetAllOrdersByUserId(int id)
{
    List<Order> orders = new List<Order>();

    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ListarPedidosByUserID", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Usuario", id);

            connection.Open();

            using (SqlDataReader reader = command.ExecuteReader())
            {
                while (reader.Read())
                {
                    Order order = new Order();
                    order.ID_Pedido =
Convert.ToInt32(reader["ID_Pedido"]);
                    order.ID_Usuario =
Convert.ToInt32(reader["ID_Usuario"]);
                    order.FechaPedido =
Convert.ToDateTime(reader["Fecha"]);
                    order.Estado = reader["Estado"].ToString();
                    order.Factura = reader["Factura"].ToString();

                    orders.Add(order);
                }
            }
        }
    }
}
```

```
        catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de pedidos"); }
        finally { connection.Close(); }

        return orders;
    }

    public Order GetOrderById(int id)
    {
        Order order = new Order();

        try
        {
            using (SqlCommand command = new SqlCommand("SP_ObtenerPedido",
connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@ID_Pedido", id);

                connection.Open();

                using (SqlDataReader reader = command.ExecuteReader())
                {
                    while (reader.Read())
                    {
                        order.ID_Pedido =
Convert.ToInt32(reader["ID_Pedido"]);
                        order.ID_Usuario =
Convert.ToInt32(reader["ID_Usuario"]);
                        order.FechaPedido =
Convert.ToDateTime(reader["Fecha"]);
                        order.Estado = reader["Estado"].ToString();
                        order.Factura = reader["Factura"].ToString();
                    }
                }
            }
        }
        catch (Exception ex) { throw new ArgumentException("No se pudo obtener
el pedido"); }
        finally { connection.Close(); }

        return order;
    }

    public void UpdateOrder(Order order)
    {
        try
        {
            using (SqlCommand command = new SqlCommand("SP_ActualizarPedido",
connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@ID_Pedido",
order.ID_Pedido);
                command.Parameters.AddWithValue("@ID_Usuario",
order.ID_Usuario);
```

```
        command.Parameters.AddWithValue("@Factura", order.Factura);

        connection.Open();
        command.ExecuteNonQuery();
    }
}
catch (Exception ex) { throw new ArgumentException("No se pudo
actualizar el pedido"); }
finally { connection.Close(); }
}
}
```

WCFMiCafesitoServices/App_Code/Services/ProductsService.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;

namespace MiCafesito
{
    public class ProductsService : IProductService
    {
        private CustomConfigurationManager _config;
        private SqlConnection connection;
        List<Category> categories;

        public ProductsService()
        {
            _config = new CustomConfigurationManager();
            connection = new SqlConnection(_config.ConnectionString);

            // Load Categories calling CategoryService
            CategoryService categoryService = new CategoryService();
            categories = categoryService.GetAllCategories();
        }

        public void AddProduct(Product product)
        {
            try
            {
                using (SqlCommand command = new SqlCommand("SP_InsertarProducto",
connection))
                {
                    command.CommandType = CommandType.StoredProcedure;

                    command.Parameters.AddWithValue("@Nombre", product.Nombre);
                    command.Parameters.AddWithValue("@Descripcion",
product.Descripcion);
                    command.Parameters.AddWithValue("@Precio", product.Precio);
                    command.Parameters.AddWithValue("@ID_Categoria",
product.ID_Categoria);
                    command.Parameters.AddWithValue("@ImageUrl",
product.ImageUrl);

                    connection.Open();
                    command.ExecuteNonQuery();
                }
            }
            catch (Exception ex) { throw new ArgumentException("No se pudo
agregar el producto"); }
            finally { connection.Close(); }
        }
    }
}
```



```
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo agregar
el producto"); }
    finally { connection.Close(); }

}

public void DeleteProduct(int id)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_EliminarProducto",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Producto", id);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
eliminar el producto"); }
    finally { connection.Close(); }
}

public List<Product> GetAllProducts()
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_ListarProductos",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            connection.Open();
            using (SqlDataReader reader = command.ExecuteReader())
            {
                List<Product> productos = new List<Product>();

                while (reader.Read())
                {
                    Product product = new Product();

                    product.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
                    product.Nombre = reader["Nombre"].ToString();
                    product.Descripcion =
reader["Descripcion"].ToString();
                    product.Precio =
Convert.ToDouble(reader["Precio"].ToString());
                    product.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);
                    product.ImageUrl = reader["ImageUrl"].ToString();

                    productos.Add(product);
                }
            }
        }
    }
}
```

```
        }

        return productos;
    }
}

catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de productos"); }
finally { connection.Close(); }
}

public List<Product> GetAllProductsByCategoryId(int id)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ListarProductosByCategoryID", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Categoria", id);

            connection.Open();
            using (SqlDataReader reader = command.ExecuteReader())
            {
                List<Product> products = new List<Product>();

                while (reader.Read())
                {
                    Product product = new Product();

                    product.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
                    product.Nombre = reader["Nombre"].ToString();
                    product.Descripcion =
reader["Descripcion"].ToString();
                    product.Precio =
Convert.ToDouble(reader["Precio"].ToString());
                    product.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);
                    product.ImageUrl = reader["ImageUrl"].ToString();

                    products.Add(product);
                }

                return products;
            }
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de productos por categoria"); }
    finally { connection.Close(); }
}

public Product GetProductById(int id)
{
    try
    {
```

```
using (SqlCommand command = new SqlCommand("SP_ObtenerProducto",
connection))
{
    command.CommandType = CommandType.StoredProcedure;

    command.Parameters.AddWithValue("@ID_Producto", id);

    connection.Open();
    using (SqlDataReader reader = command.ExecuteReader())
    {
        Product product = new Product();

        while (reader.Read())
        {
            product.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
            product.Nombre = reader["Nombre"].ToString();
            product.Descripcion =
reader["Descripcion"].ToString();
            product.Precio =
Convert.ToDouble(reader["Precio"].ToString());
            product.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);

            // Get Category Name
            foreach (Category category in categories)
            {
                if (category.ID_Categoria == product.ID_Categoria)
                {
                    product.Categoria = category.Nombre;
                    break;
                }
            }
            product.ImageUrl = reader["ImageUrl"].ToString();
        }

        return product;
    }
}

catch (Exception ex) { throw new ArgumentException("No se pudo obtener
el producto"); }
finally { connection.Close(); }

return null;
}

public List<Product> GetProductsByName(string criteria)
{
    // SP_ObtenerProductoByName
    // @Nombre nvarchar(100)

    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ObtenerProductoByName", connection))
        {
            command.CommandType = CommandType.StoredProcedure;
```

```
        command.Parameters.AddWithValue("@Criterio", criteria);

        connection.Open();
        using (SqlDataReader reader = command.ExecuteReader())
        {
            List<Product> products = new List<Product>();

            while (reader.Read())
            {
                Product product = new Product();

                product.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
                product.Nombre = reader["Nombre"].ToString();
                product.Descripcion =
reader["Descripcion"].ToString();
                product.Precio =
Convert.ToDouble(reader["Precio"].ToString());
                product.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);
                product.ImageUrl = reader["ImageUrl"].ToString();

                products.Add(product);
            }

            return products;
        }
    }
}

catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de productos por nombre"); }
finally { connection.Close(); }

}

public List<Product> GetProductsFeatured()
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ListarProductosDestacados", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            connection.Open();
            using (SqlDataReader reader = command.ExecuteReader())
            {
                List<Product> products = new List<Product>();

                while (reader.Read())
                {
                    Product product = new Product();

                    product.ID_Producto =
Convert.ToInt32(reader["ID_Producto"]);
                    product.Nombre = reader["Nombre"].ToString();
                    product.Descripcion =
reader["Descripcion"].ToString();
                    product.Precio =
```

```
Convert.ToDouble(reader["Precio"].ToString());
                product.ID_Categoria =
Convert.ToInt32(reader["ID_Categoria"]);
                product.ImageUrl = reader["ImageUrl"].ToString();

                products.Add(product);
            }

            return products;
        }
    }
}

catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de productos destacados"); }
finally { connection.Close(); }
}

public void UpdateProduct(Product productos)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ActualizarProducto", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Producto",
productos.ID_Producto);
            command.Parameters.AddWithValue("@Nombre", productos.Nombre);
            command.Parameters.AddWithValue("@Descripcion",
productos.Descripcion);
            command.Parameters.AddWithValue("@Precio", productos.Precio);
            command.Parameters.AddWithValue("@ID_Categoria",
productos.Precio);
            command.Parameters.AddWithValue("@ImageUrl",
productos.ImageUrl);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex) { throw new ArgumentException("No se pudo
actualizar el producto"); }
    finally { connection.Close(); }
}
}
}
```

WCFMiCafesitoServices/App_Code/Services/RolesService.cs

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Runtime.Remoting.Messaging;
using System.Web;

namespace MiCafesito
```

```
{
    public class RolesService: IRolesService
    {
        private CustomConfigurationManager _config;
        private SqlConnection connection;

        public RolesService()
        {
            _config = new CustomConfigurationManager();
            connection = new SqlConnection(_config.ConnectionString);
        }

        public List<Role> GetAllRoles()
        {
            List<Role> roles = new List<Role>();

            try
            {
                using (SqlCommand command = new SqlCommand("SP_ListarRoles",
connection))
                {
                    command.CommandType = System.Data.CommandType.StoredProcedure;
                    connection.Open();

                    using (SqlDataReader reader = command.ExecuteReader())
                    {
                        while (reader.Read())
                        {
                            Role role = new Role();
                            role.ID_Rol = Convert.ToInt32(reader["ID_Rol"]);
                            role.Nombre = reader["Nombre"].ToString();

                            roles.Add(role);
                        }
                    }
                }
            }
            catch (Exception ex) { throw new ArgumentException("No se pudo obtener
la lista de roles"); }
            finally { connection.Close(); }

            return roles;
        }
    }
}
```

WCFMiCafesitoServices/App_Code/Services/UserService.cs

```
using System;
using System.Data.SqlClient;
using System.Data;
using System.Collections.Generic;

namespace MiCafesito
{
    public class UserService : IUserService
```

```
{
    private CustomConfigurationManager _config;
    private SqlConnection connection;

    public UserService()
    {
        _config = new CustomConfigurationManager();
        connection = new SqlConnection(_config.ConnectionString);
    }

    public void AddUser(User user)
    {
        int newUserId = 0;

        // Create connection and command objects
        try
        {
            using (SqlCommand command = new SqlCommand("SP_CrearUsuario",
connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                // Add parameters
                command.Parameters.AddWithValue("@Nombre", user.Nombre);
                command.Parameters.AddWithValue("@Apellido", user.Apellido);
                command.Parameters.AddWithValue("@FechaNacimiento",
user.Fecha_Nacimiento);
                command.Parameters.AddWithValue("@Email", user.Email);
                command.Parameters.AddWithValue("@Password", user.Password);
                command.Parameters.AddWithValue("@Telefono", user.Telefono);
                command.Parameters.AddWithValue("@Estado", user.Estado);
                command.Parameters.AddWithValue("@ID_Rol", user.ID_Rol);
                command.Parameters.AddWithValue("@Patron", _config.Patron);

                // Open connection and execute the stored procedure
                connection.Open();
                object result = command.ExecuteScalar();

                // Check if the result is not null and convert it to int
                if (result != null)
                {
                    newUserId = Convert.ToInt32(result);
                }
            }
        }
        catch (Exception ex) { throw new ArgumentException("No se pudo agregar
el usuario"); }
        finally { connection.Close(); }
    }

    public void DeleteUser(int id)
    {
        try
        {
            using (SqlCommand command = new SqlCommand("SP_EliminarUsuario",
connection))
            {
                command.CommandType = CommandType.StoredProcedure;
```

```
        command.Parameters.AddWithValue("@ID_Usuario", id);

        connection.Open();
        command.ExecuteNonQuery();
    }
}
catch (Exception ex)
{
    throw new ArgumentException("No se pudo eliminar el usuario");
}
finally { connection.Close(); }
}

public List<User> GetAllUsers()
{
    List<User> userList = new List<User>();

    try
    {
        using (SqlCommand command = new SqlCommand("SP_ListarUsuarios",
connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@Patron", _config.Patron);

            connection.Open();
            using (SqlDataReader reader = command.ExecuteReader())
            {
                while (reader.Read())
                {
                    User user = new User
                    {
                        ID_Usuario = reader.GetInt32(0),
                        Nombre = reader.GetString(1),
                        Apellido = reader.GetString(2),
                        Fecha_Nacimiento = reader.GetDateTime(3),
                        Email = reader.GetString(4),
                        Telefono = reader.IsDBNull(5) ? null :
reader.GetString(5),
                        ID_Rol = reader.GetInt32(6),
                        NombreRol = reader.GetString(7),
                        Estado = reader.GetString(8),
                        Fecha_Registro = reader.GetDateTime(9)
                    };

                    userList.Add(user);
                }
            }
        }
    }
    catch (Exception ex)
    {
        throw new ArgumentException("Error al obtener la lista de
usuarios");
    }
    finally { connection.Close(); }
```



```
        return userList;
    }

    public User GetUserById(int id)
    {
        User user;

        try
        {
            using (SqlCommand command = new SqlCommand("SP_LeerUsuario",
connection))
            {
                command.CommandType = CommandType.StoredProcedure;

                command.Parameters.AddWithValue("@ID_Usuario", id);
                command.Parameters.AddWithValue("@Patron", _config.Patron);

                connection.Open();
                using (SqlDataReader reader = command.ExecuteReader())
                {
                    if (reader.Read())
                    {
                        user = new User
                        {
                            ID_Usuario = reader.GetInt32(0),
                            Nombre = reader.GetString(1),
                            Apellido = reader.GetString(2),
                            Fecha_Nacimiento = reader.GetDateTime(3),
                            Email = reader.GetString(4),
                            Telefono = reader.GetString(5),
                            ID_Rol = reader.GetInt32(6),
                            NombreRol = reader.GetString(7),
                            Estado = reader.GetString(8),
                            Fecha_Registro = reader.GetDateTime(9)
                        };
                    }

                    return user;
                }
            }
        }
        catch (Exception ex)
        {
            throw new ArgumentException("Error al obtener el usuario");
        }
        finally { connection.Close(); }

        return null;
    }

    public int Login(string user, string password)
    {
        int newUserId = 0;
        try
        {
            using (SqlCommand command = new SqlCommand("SP_ValidarUsuario",
connection))
            {
```

```
        command.CommandType = CommandType.StoredProcedure;

        command.Parameters.AddWithValue("@Email", user);
        command.Parameters.AddWithValue("@Password", password);
        command.Parameters.AddWithValue("@Patron", _config.Patron);

        connection.Open();
        object result = command.ExecuteScalar();

        // Check if the result is not null and convert it to int
        if (result != null)
        {
            newUserId = Convert.ToInt32(result);
        }
    }
}
catch (Exception ex)
{
    throw new ArgumentException("Usuario y/o contrasena invalidas");
}
finally { connection.Close(); }

return newUserId;
}

public void ResetPassword(string email, string password, string
newPassword, string confirmPassword)
{
    try
    {
        using (SqlCommand command = new
SqlCommand("SP_ResetearContrasena", connection))
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@Email", email);
            command.Parameters.AddWithValue("@PasswordActual", password);
            command.Parameters.AddWithValue("@NuevaContrasena",
newPassword);
            command.Parameters.AddWithValue("@VerificarContrasena",
confirmPassword);
            command.Parameters.AddWithValue("@Patron", _config.Patron);

            connection.Open();
            command.ExecuteNonQuery();
        }
    }
    catch (Exception ex)
    {
        throw new ArgumentException("No se pudo resetear la contrasena");
    }
    finally { connection.Close(); }
}

public void SearchUser(SearchCriteria criteria)
{
    // Implementation
}
```

```
public void UpdateUser(User user)
{
    try
    {
        using (SqlCommand command = new SqlCommand("SP_ActualizarUsuario",
connection)
        {
            command.CommandType = CommandType.StoredProcedure;

            command.Parameters.AddWithValue("@ID_Usuario",
user.ID_Usuario);
            command.Parameters.AddWithValue("@Nombre", user.Nombre);
            command.Parameters.AddWithValue("@Apellido", user.Apellido);
            command.Parameters.AddWithValue("@FechaNacimiento",
user.Fecha_Nacimiento);
            command.Parameters.AddWithValue("@Telefono", user.Telefono);
            command.Parameters.AddWithValue("@Estado", user.Estado);
            command.Parameters.AddWithValue("@ID_Rol", user.ID_Rol);
            command.Parameters.AddWithValue("@Patron", _config.Patron);

            connection.Open();
            command.ExecuteNonQuery();

        }
    }
    catch (Exception ex)
    {
        throw new ArgumentException("No se pudo actualizar el usuario");
    }
    finally { connection.Close(); }
}

}
```

Utilitarios

WCFMiCafesitoServices/App_Code/Utils/CustomConfigurationManager.cs

```
using Newtonsoft.Json;
using System;
using System.IO;

public class CustomConfigurationManager
{
    public string ConnectionString { get; private set; }
    public string Patron { get; private set; }

    public CustomConfigurationManager()
    {
        string jsonFilePath = Path.Combine(
            AppDomain.CurrentDomain.BaseDirectory,
            "keys.json"
        );
    }
}
```

```
try
{
    using (StreamReader reader = File.OpenText(jsonPath))
    {
        string json = reader.ReadToEnd();
        dynamic configData = JsonConvert.DeserializeObject(json);
        ConnectionString = configData.connectionString;
        Patron = configData.patron;
    }
}
catch (Exception ex)
{
    Console.WriteLine("Error al cargar la conexión a la base de datos: " +
ex.Message);
}

public static implicit operator string(CustomConfigurationManager v)
{
    throw new NotImplementedException();
}
}
```

Capa de Presentación Asp.net

WAMiCafesitoApp/Default.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Public.Master"
AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="WAMiCafesitoApp.Default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>

<asp:Content ID="Content3" ContentPlaceHolderID="header" runat="server">
    <div class="container-full hero py-5">
        <div class="gradient-overlay">
            <div class="container">
                <div class="row">
                    <div class="col-md-6 mt-5 pt-3">
                        <h1 class="display-4">¡Descubre el
                        <br />
                        V60 Hario!</h1>
                        <p class="lead">
                            Disfruta descubriendo tus granos favoritos,
                        <br />
                            método de preparación e inventando tu receta..
                        </p>
                        <div class="mt-4 d-flex gap-3">
                            <a href="/Store/ProductDetail.aspx?id=7" class="btn
btn-informacion btn-lg rounded-pill text-white px-4 ml-4">Más Información</a>
                        </div>
                    </div>
                    <div class="col-md-6 d-flex justify-content-center">
                        
    </div>
    </div>
    </div>
    </div>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="MainContent" runat="server">
    <section>
        <div class="container my-5">
            <header class="mb-4">
                <h3>PRODUCTOS DESTACADOS</h3>
            </header>

            <asp:HiddenField ID="hdnToastMessage" runat="server" />
            <asp:HiddenField ID="hdnToastType" runat="server" />
            <div class="row">
                <asp:Repeater ID="featuredProductsRepeater" runat="server"
OnItemDataBound="featuredProductsRepeater_ItemDataBound">
                    <ItemTemplate>
                        <div class="col-lg-3 col-md-6 col-sm-6 d-flex mb-5">
                            <div class="card w-100 my-2 shadow-2-strong">
                                <img src='/Assets/Images/<%# Eval("ID_Producto")
%>.png'

                                class="card-img-top"
                                style="aspect-ratio: 1 / 1"
                                onerror="this.onerror=null;
this.src='/Assets/Images/Default.png';" />
                                <div class="card-body d-flex flex-column">
                                    <h5 class="card-title"><%# Eval("Nombre")
%></h5>

                                    <span class="a-price d-flex align-items-start"
aria-hidden="true">

                                        <span class="a-price-symbol">$</span>
                                        <span class="a-price-whole"><%#
Convert.ToInt32(Math.Floor(Convert.ToDouble(Eval("Precio")))) %></span>
                                        <span class="a-price-fraction"><%#
String.Format("{0:00}", GetFractionalPart(Eval("Precio"))) %></span>
                                        </span>
                                    <p class="card-text"><%# Eval("Descripcion",
"{0:C}") %></p>

                                    <div class="d-flex align-items-end py-2 px-0
mt-auto gap-2">

                                        <asp:HyperLink ID="btnViewDetail"
runat="server"
                                        CssClass="btn btn-light shadow-none

                                        <i class="fa-solid fa-magnifying-
glass"></i> Detalle

                                        </asp:HyperLink>
                                        <asp:LinkButton
                                        CssClass="btn btn-light shadow-none

                                        ID="btnAddToCart"
                                        runat="server"
                                        CommandName="AddToCart"
                                        CommandArgument='<%#
Eval("ID_Producto") %>'

                                        OnClick="btnAddToCart_Click"><i

```

```

class="fa-solid fa-cart-shopping"></i> Ordenar</asp:LinkButton>
</div>

</div>
</div>
</div>
</ItemTemplate>
</asp:Repeater>
</div>
</div>
</section>
</asp:Content>
<asp:Content ID="Content4" ContentPlaceHolderID="FooterScripts" runat="server">
    <script src="./Assets/Scripts/ErrMesasges.js" type="text/javascript"></script>
    <script>
        var toastMessage = document.getElementById('<%= hdnToastMessage.ClientID
%>');
        var toastType = document.getElementById('<%= hdnToastType.ClientID %>');
        if (toastMessage) {
            showAnimatedToast(toastMessage, toastType);
        }
    </script>
</asp:Content>

```

WAMiCafesitoApp/Default.aspx.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WAMiCafesitoApp.ServiceApi;
using WAMiCafesitoApp.Services;

namespace WAMiCafesitoApp
{
    public partial class Default : System.Web.UI.Page
    {
        IProductService productService = new ProductServiceClient();
        private CartService _cartService = new CartService();
        protected void Page_Load(object sender, EventArgs e)
        {
            LoadFeaturedProducts();
        }

        protected void LoadFeaturedProducts()
        {
            List<Product> products =
productService.GetProductsFeatured().ToList();
            if (products.Count > 0)
            {

```

```
        featuredProductsRepeater.DataSource = products;
        featuredProductsRepeater.DataBind();

    }

protected string GetFractionalPart(object price)
{
    if (price != null && price != DBNull.Value)
    {
        decimal priceValue = Convert.ToDecimal(price);
        int wholePart = (int)priceValue;
        decimal fractionalPart = priceValue - wholePart;
        int cents = (int)(fractionalPart * 100);
        return cents.ToString("00");
    }
    return "00";
}

protected void featuredProductsRepeater_ItemDataBound(object sender,
RepeaterItemEventArgs e)
{
    if (e.Item.ItemType == ListItemType.Item || e.Item.ItemType ==
ListItemType.AlternatingItem)
    {
        HyperLink verDetalleLink =
(HyperLink)e.Item.FindControl("btnViewDetail");

        if (verDetalleLink != null)
        {

            Product product = (Product)e.Item.DataItem;

            if (product != null)
            {

                verDetalleLink.NavigateUrl =
$"Store/ProductDetail.aspx?id={product.ID_Producto}";
            }
        }

        LinkButton addToCartButton =
(LinkButton)e.Item.FindControl("btnAddToCart");

        if (addToCartButton != null)
        {
            Product product = (Product)e.Item.DataItem;

            if (product != null)
            {
                addToCartButton.CommandArgument =
product.ID_Producto.ToString();
            }
        }
    }
}

protected void btnAddToCart_Click(object sender, EventArgs e)
```

```
{
    LinkButton btn = (LinkButton)sender;
    int productId = Convert.ToInt32(btn.CommandArgument);

    // Retrieve the product using the productId
    ServiceApi.Product product = GetProductById(productId);

    if (product != null)
    {
        // Assuming you have a method to get the quantity, e.g., from a
        TextBox within the Repeater item
        RepeaterItem item = (RepeaterItem)btn.NamingContainer;
        TextBox txtQuantity = (TextBox)item.FindControl("txtQuantity");

        List<ServiceApi.Cart> cartItems =
        _cartService.AddOrUpdateCartItem(product, 1);

        // Update the session with the updated cart items
        Session["CartItems"] = cartItems;

        ShowErrorMessage($"Producto agregado al carrito.");
    }
}

private Product GetProductById(int productId)
{
    return productService.GetProductById(productId);
}

private void ShowErrorMessage(string message)
{
    hdnToastMessage.Value = message;
    hdnToastType.Value = "info";
}
}
```

WAMiCafesitoApp/Login.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs"
Inherits="WAMiCafesitoApp.Login" %>

<!DOCTYPE html>
<html lang="en">
<head runat="server">
    <meta charset="UTF-8">
    <title>Acceso a usuarios MiCafesito.com</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMHjY6hW+ALEwIH"
crossorigin="anonymous">
```



```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.5.2/css/all.min.css" integrity="sha512-
SnH5WK+bZxgPHs44uWIX+LLJAJ9/2PkPKZ5QiAj6Ta86w+fsb2TkcmfRyVX3pBnMFCv7oQPJkl9QevSCWr
3W6A==" crossorigin="anonymous" referrerpolicy="no-referrer" />
<link href="/Assets/Styles/style.css" rel="stylesheet" />
<link href="/Assets/Styles/login.css" rel="stylesheet" />
</head>
<body>
  <section class="h-100 gradient-form" style="background-color: #eee;">
    <div class="container py-5 h-100">
      <div class="row d-flex justify-content-center align-items-center h-
100">
        <div class="col-xl-10">
          <div class="card rounded-3 text-black">
            <div class="row g-0">
              <div class="col-lg-6">
                <div class="card-body p-md-5 mx-md-4">
                  <div class="pb-1 pt-5 d-flex">
                    <a href="/" target="_blank" class="d-flex
gap-2 logo-link">
                      
                      <h1 class="logo-text">MiCafesito <span
class="logo-text-small d-block">Enlínea</span></h1>
                    </a>
                  </div>
                  <div>
                    <form id="form1" runat="server"
class="signin-form">
                      <asp:HiddenField ID="hdnToastMessage"
runat="server" />
                      <h1 class="login-text">Ingresa a tu
cuenta</h1>
                      <div class="form-outline mb-4">
                        <asp:TextBox ID="txtUser"
runat="server" CssClass="form-control" />
                        <asp:Label ID="lblUser"
runat="server" AssociatedControlID="txtUser" CssClass="form-label">Correo
electrónico</asp:Label>
                      </div>
                      <div class="form-outline mb-4">
                        <asp:TextBox ID="txtPassword"
runat="server" TextMode="Password" CssClass="form-control" />
                        <asp:Label ID="lblPassword"
runat="server" AssociatedControlID="txtPassword" CssClass="form-
label">Contraseña</asp:Label>
                      </div>
                      <div class="pt-1 mb-5 pb-1">
                        <asp:Button ID="btnLogin"
runat="server" Text="Ingresar" CssClass="btn btn-comprar btn-md rounded-pill mr-3
text-white px-4" OnClick="btnLogin_Click" />
                        <a class="text-muted"
href="#">¿Olvidaste tu contraseña?</a>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </section>
```

```
<div class="d-flex align-items-center justify-content-center pb-4">  
    <p class="mb-0 me-2">¿No tienes  
una cuenta?</p>  
    <button type="button" class="btn btn-link">Regístrate</button>  
</div>  
</form>  
</div>  
</div>  
</div>  
<div class="col-lg-6 d-flex align-items-center gradient-custom-2 img-logo">  
</div>  
</div>  
</div>  
</div>  
</div>  
</section>  
  
<div class="toast-container position-fixed top-0 end-0 p-3">  
    <div id="liveToast" class="toast bg-danger text-white" role="alert" aria-live="assertive" aria-atomic="true">  
        <div class="toast-body d-flex align-items-center">  
            <i class="fas fa-exclamation-triangle fa-sm me-2"></i>  
            <span id="toastMessage"></span>  
        </div>  
    </div>  
</div>  
  
<!-- Bootstrap JS from CDN (Optional, for certain Bootstrap components) -->  
<script  
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"  
integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"  
crossorigin="anonymous"></script>  
  
<script>  
    // Function to show the Bootstrap toast  
    function showAnimatedToast(message) {  
        var toast = new bootstrap.Toast(document.getElementById('liveToast'), {  
            animation: true,  
            autohide: true,  
            delay: 3500000  
        });  
  
        var toastMessage = document.getElementById('toastMessage');  
        toastMessage.innerText = message;  
  
        // Show the toast with animation  
        toast.show();  
    }  
  
    var toastMessage = document.getElementById('<%= hdnToastMessage.ClientID
```

```
%>').value;  
    if (toastMessage) {  
        showAnimatedToast(toastMessage);  
    }  
</script>
```

```
</body>  
</html>
```

WAMiCafesitoApp/Login.aspx.cs

```
using System;  
using WAMiCafesitoApp.ServiceApi;  
  
namespace WAMiCafesitoApp  
{  
    public partial class Login : System.Web.UI.Page  
    {  
        readonly UserServiceClient userService = new UserServiceClient();  
        protected void Page_Load(object sender, EventArgs e)  
        {  
            if (!IsPostBack)  
            {  
                // Comprobar si el usuario ya ha iniciado sesión y redirigirlo si  
es necesario  
                if (Session["UserId"] != null)  
                {  
                    RedirectBasedOnRole();  
                }  
            }  
  
            protected void btnLogin_Click(object sender, EventArgs e)  
            {  
                string username = txtUser.Text.Trim();  
                string password = txtPassword.Text.Trim();  
  
                try  
                {  
                    int userId = userService.Login(username, password);  
  
                    if (userId > 0)  
                    {  
                        // El usuario se autenticó correctamente  
                        // Obtener la información del usuario  
                        User user = userService.GetUserById(userId);  
  
                        if (user != null)  
                        {  
                            // Almacenar el objeto de usuario en la sesión  
                            Session["UserId"] = user.ID_Usuario;  
                            Session["RoleId"] = user.ID_Rol;  
  
                            // Redirigir al usuario según su rol  
                            RedirectBasedOnRole();  
                        }  
                    }  
                }  
            }  
        }  
    }  
}
```

```
        }
        else
        {
            // Error al obtener la información del usuario
            // Manejar el error o mostrar un mensaje
            ShowErrorMessage("Error al obtener la información del
usuario.");
        }
    }
    else
    {
        // La autenticación falló
        // Manejar credenciales inválidas o mostrar un mensaje
        ShowErrorMessage("Credenciales inválidas. Inténtalo de nuevo.");
    }
}
catch (Exception ex)
{
    // Manejar la excepción o mostrar un mensaje de error
    ShowErrorMessage("Se produjo un error durante el inicio de sesión.
Por favor, inténtelo de nuevo más tarde.");
}

private void RedirectBasedOnRole()
{
    // Verificar si tanto UserId como RoleId existen en la sesión
    if (Session["UserId"] != null && Session["RoleId"] != null)
    {
        int roleId = Convert.ToInt32(Session["RoleId"]);

        // Redirigir según el rol
        if (roleId == 1)
        {
            // Suponiendo que el rol 1 es para administradores
            // Redirigir a la página de administrador
            Response.Redirect("~/Admin/Default.aspx");
        }
        else
        {
            // Redirigir a la página de cliente
            Response.Redirect("~/Default.aspx");
        }
    }
    else
    {
        ShowErrorMessage("Error al redirigir. Por favor, inicie sesión
primero.");
    }
}

private void ShowErrorMessage(string message)
{
    hdnToastMessage.Value = message;
}
}
```

WAMiCafesitoApp/Public.Master

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Public.Master.cs"
Inherits="WAMiCafesitoApp.Site1" %>

<!DOCTYPE html>
<html lang="en">
<head runat="server">
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH"
crossorigin="anonymous">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/font-
awesome/6.5.2/css/all.min.css" integrity="sha512-
SnH5WK+bZxgPHs44uWIX+LLJAJ9/2PkPKZ5QiAj6Ta86w+fsb2TkcmfRyVX3pBnMFCv7oQPJkl9QevSCWr
3W6A==" crossorigin="anonymous" referrerpolicy="no-referrer" />
    <link href="/Assets/Styles/style.css" rel="stylesheet" />
    <asp:ContentPlaceHolder ID="head" runat="server">
    </asp:ContentPlaceHolder>
</head>

<body>
    <form id="form2" runat="server">

        <!--Main Navigation-->
        <header>
            <!-- Jumbotron -->
            <div class="p-3 text-center bg-white border-bottom">
                <div class="container">
                    <div class="row gy-3">
                        <div class="col-lg-2 col-sm-4 col-4">
                            <a href="/" target="_self" class="float-start d-flex
align-items-center gap-2 logo-link">
                                
                                <h1 class="logo-text">MiCafesito <span
class="logo-text-small d-block">Enlínea</span></h1>
                            </a>
                        </div>

                        <div class="order-lg-last col-lg-5 col-sm-8 col-8">
                            <div class="d-flex float-end">
                                <asp:Placeholder ID="UnauthenticatedLnks"
runat="server" Visible="true">
                                    <asp:HyperLink ID="loginLnk" runat="server"
NavigateUrl="~/Login.aspx" CssClass="me-1 py-1 px-3 nav-link d-flex align-items-
center">
                                        <i class="fas fa-user-alt m-1 me-md-
2"></i>
                                        <p class="d-none d-md-block mb-
0">Ingresar</p>
                                    </asp:HyperLink>
                                    <asp:HyperLink ID="registreLnk" runat="server"
NavigateUrl="~/Signup.aspx" CssClass="me-1 py-1 px-3 nav-link d-flex align-items-
```

```

center">
                                <i class="fas fa-heart m-1 me-md-2"></i>
                                <p class="d-none d-md-block mb-
0">Registrarse</p>
                                </asp:HyperLink>
                                </asp:Placeholder>
                                <asp:Placeholder ID="AuthenticatedLnks"
runat="server" Visible="false">
                                <asp:HyperLink ID="cartLnk" runat="server"
NavigateUrl="~/Store/Cart.aspx" CssClass="me-1 py-1 px-3 nav-link d-flex align-
items-center">
                                <i class="fas fa-shopping-cart m-1 me-md-
2"></i>
                                <p class="d-none d-md-block mb-
0">Carrito</p>
                                <asp:Label ID="cartCountLbl"
Visible="false" runat="server" CssClass="badge bg-primary rounded-pill ms-
1"></asp:Label>
                                </asp:HyperLink>
                                <asp:Button ID="cartBtn" runat="server"
OnClick="closeSessionLnk_Click" Text="Salir" CssClass="btn btn-outline-primary me-
1 py-1 px-3 d-flex align-items-center"></asp:Button>
                                </asp:Placeholder>
                                </div>
                                </div>
                                <div class="col-lg-5 col-md-12 col-12">
                                    <div class="input-group search-box">
                                        <asp:TextBox ID="txtSearchBox" runat="server"
CssClass="form-control"></asp:TextBox>
                                        <asp:LinkButton ID="searchBtn" runat="server"
CssClass="btn btn-outline-secondary" OnClick="searchBtn_Click">
                                            <i class="fas fa-search"></i>
                                        </asp:LinkButton>
                                    </div>
                                </div>
                                </div>
                                </div>
                                </div>
                                </div>
                                <!-- Navbar -->
                                <nav class="navbar navbar-expand-lg navbar-light bg-white">
                                    <div class="container justify-content-center justify-content-md-
between">
                                        <button
                                            class="navbar-toggler border py-2 text-dark"
                                            type="button"
                                            data-mdb-toggle="collapse"
                                            data-mdb-target="#navbarLeftAlignExample"
                                            aria-controls="navbarLeftAlignExample"
                                            aria-expanded="false"
                                            aria-label="Toggle navigation">
                                            <i class="fas fa-bars"></i>
                                        </button>
                                        <div class="collapse navbar-collapse"
id="navbarLeftAlignExample">

```

```

        <ul class="navbar-nav me-auto mb-2 mb-lg-0">
            <asp:Repeater ID="categoryRepeater" runat="server">
                <ItemTemplate>
                    <asp:HyperLink ID="categoryLink"
runat="server" CssClass="nav-link text-dark cat-link"
                    NavigateUrl='<%#
"/Store/Category.aspx?cat=" + Eval("ID_Categoria") %>' Text='<%# Eval("Nombre")
%>' />
                </ItemTemplate>
            </asp:Repeater>
        </ul>
    </div>
</div>
</nav>
<asp:ContentPlaceHolder ID="header" runat="server">
</asp:ContentPlaceHolder>

</header>

<!-- main -->
<main>
    <asp:ContentPlaceHolder ID="MainContent" runat="server">
    </asp:ContentPlaceHolder>
</main>
<!-- main -->

<!-- Feature -->
<section class="mt-5" style="background-color: #f5f5f5;">
    <div class="container text-dark pt-3">
        <header class="pt-4 pb-3">
            <h3>Por qué comprar con nosotros</h3>
        </header>

        <div class="row mb-4">
            <div class="col-lg-4 col-md-6">
                <figure class="d-flex align-items-center mb-4">
                    <span class="rounded-circle bg-white p-3 d-flex me-2
mb-2">
                        <i class="fas fa-camera-retro fa-2x fa-fw text-
primary floating"></i>
                    </span>
                    <figcaption class="info">
                        <h6 class="title">Precios razonables</h6>
                        <p>Disfruta de productos asequibles para una
experiencia de café excepcional y deliciosa.</p>
                    </figcaption>
                </figure>
                <!-- itemside // -->
            </div>
            <!-- col // -->
            <div class="col-lg-4 col-md-6">
                <figure class="d-flex align-items-center mb-4">
                    <span class="rounded-circle bg-white p-3 d-flex me-2
mb-2">
                        <i class="fas fa-star fa-2x fa-fw text-primary
floating"></i>
                    </span>
                    <figcaption class="info">

```

```

<h6 class="title">Mejor calidad</h6>
<p>Garantizamos la excelencia en cada taza con
productos de calidad superior y deliciosa.</p>
</figcaption>
</figure>
<!-- itemside // -->
</div>
<!-- col // -->
<div class="col-lg-4 col-md-6">
<figure class="d-flex align-items-center mb-4">
<span class="rounded-circle bg-white p-3 d-flex me-2
mb-2">
<i class="fas fa-plane fa-2x fa-fw text-primary
floating"></i>
</span>
<figcaption class="info">
<h6 class="title">Envíos a todo El Salvador</h6>
<p>Enviamos nuestros productos a cualquier lugar,
para que puedas disfrutar del mejor café en casa, ¡sin importar dónde estés!</p>
</figcaption>
</figure>
<!-- itemside // -->
</div>
<!-- col // -->
<div class="col-lg-4 col-md-6">
<figure class="d-flex align-items-center mb-4">
<span class="rounded-circle bg-white p-3 d-flex me-2
mb-2">
<i class="fas fa-users fa-2x fa-fw text-primary
floating"></i>
</span>
<figcaption class="info">
<h6 class="title">Satisfacción del cliente</h6>
<p>Tu felicidad es nuestra prioridad, te
aseguramos una experiencia de compra satisfactoria.</p>
</figcaption>
</figure>
<!-- itemside // -->
</div>
<!-- col // -->
<div class="col-lg-4 col-md-6">
<figure class="d-flex align-items-center mb-4">
<span class="rounded-circle bg-white p-3 d-flex me-2
mb-2">
<i class="fas fa-thumbs-up fa-2x fa-fw text-
primary floating"></i>
</span>
<figcaption class="info">
<h6 class="title">Clientes satisfechos</h6>
<p>Nuestros clientes hablan por nosotros,
proporcionamos atención excepcional y productos de alta calidad para que disfrutes
al máximo de tu café.</p>
</figcaption>
</figure>
<!-- itemside // -->
</div>
<!-- col // -->
<div class="col-lg-4 col-md-6">
<figure class="d-flex align-items-center mb-4">

```



```

        <span class="rounded-circle bg-white p-3 d-flex me-2
mb-2">
        <i class="fas fa-box fa-2x fa-fw text-primary
floating"></i>
        </span>
        <figcaption class="info">
            <h6 class="title">Todo lo que necesitas</h6>
            <p>
                Explora nuestra amplia selección de productos,
desde cafeteras hasta accesorios, ¡todo lo que necesitas para tu café perfecto!
            </p>
        </figcaption>
    </figure>
    <!-- itemside // -->
</div>
<!-- col // -->
</div>
</div>
<!-- container end.// -->
</section>
<!-- Feature -->

<!-- Footer -->
<footer class="text-center text-lg-start text-muted" style="background-
color: #f5f5f5;">
    <!-- Section: Links -->
    <section class="">
        <div class="container text-center text-md-start pt-4 pb-4">
            <!-- Grid row -->
            <div class="row mt-3">
                <!-- Grid column -->
                <div class="col-12 col-lg-3 col-sm-12 mb-2">
                    <div class="d-flex flex-column">

                        <p class="mt-2 text-dark">
                            © 2023 MiCafesito.com
                        </p>
                        <div>
                            <i class="fab fa-lg fa-cc-visa text-dark"></i>
                            <i class="fab fa-lg fa-cc-amex text-dark"></i>
                            <i class="fab fa-lg fa-cc-mastercard text-
dark"></i>
                            <i class="fab fa-lg fa-cc-paypal text-
dark"></i>

                        </div>
                    </div>
                </div>
            </div>
            <!-- Grid column -->

        </div>
    </section>

    </footer>
    <!-- Footer -->
</form>

```

```
<div class="toast-container position-fixed top-0 end-0 p-3">
  <div id="liveToast" class="toast text-white" role="alert" aria-
live="assertive" aria-atomic="true">
    <div class="toast-body d-flex align-items-center">
      <i class="fas fa-exclamation-triangle fa-sm me-2"></i>
      <span id="toastMessage"></span>
    </div>
  </div>
</div>

<!-- Bootstrap JS from CDN (Optional, for certain Bootstrap components) -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"
crossorigin="anonymous"></script>
  <asp:ContentPlaceHolder ID="FooterScripts" runat="server">
  </asp:ContentPlaceHolder>
</body>
</html>
```

WAMiCafesitoApp/Public.Master.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WAMiCafesitoApp.ServiceApi;

namespace WAMiCafesitoApp
{
    public partial class Site1 : System.Web.UI.MasterPage
    {
        ServiceApi.ICategoryService categoryService = new
ServiceApi.CategoryServiceClient();
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                if (Session["UserId"] != null)
                {
                    HideLoginLink();
                }

                if (Session["CartItems"] != null)
                {
                    HideLoginLink();
                }

                LoadCategories();
            }
        }
    }
}
```

```
protected void LoadCategories()
{
    List<ServiceApi.Category> categories =
categoryService.GetAllCategories().ToList();

    if (categories.Count > 0)
    {
        categoryRepeater.DataSource = categories;
        categoryRepeater.DataBind();
    }
}

protected void RedirectLogin()
{
    Response.Redirect("~/Login.aspx");
}

protected void HideLoginLink()
{
    UnauthenticatedLnks.Visible = false;
    AuthenticatedLnks.Visible = true;
}

protected void closeSessionLnk_Click(object sender, EventArgs e)
{
    // Almacenar el objeto de usuario en la sesión
    Session["UserId"] = null;
    Session["RoleId"] = null;
    Response.Redirect("/Default.aspx");
}

protected void searchBtn_Click(object sender, EventArgs e)
{
    Response.Redirect("/Store/ResultsPage.aspx?criteria=" +
txtSearchBox.Text);
}
}
```

WAMiCafesitoApp/Web.config

```
<?xml version="1.0" encoding="utf-8"?>
<!--
    For more information on how to configure your ASP.NET application, please visit
    https://go.microsoft.com/fwlink/?LinkId=169433
-->
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.7.2" />
    <httpRuntime targetFramework="4.7.2" />
  </system.web>
  <system.codedom>
    <compilers>
      <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
```

```
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:1659;1699;1701" />
    <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:41008
/define:_MYTYPE=\&quot;Web\&quot;; /optionInfer+" />
</compilers>
</system.codedom>
<system.serviceModel>
    <bindings>
        <basicHttpBinding>
            <binding name="BasicHttpBinding_IUserService" />
            <binding name="BasicHttpBinding_ICategoryService" />
            <binding name="BasicHttpBinding_IProductService" />
            <binding name="BasicHttpBinding_IOrderService" />
            <binding name="BasicHttpBinding_IOrderDetailsService" />
            <binding name="BasicHttpBinding_ICartService" />
        </basicHttpBinding>
    </bindings>
    <client>
        <endpoint address="http://localhost:61183/Service.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_IUserService"
contract="ServiceApi.IUserService"
        name="BasicHttpBinding_IUserService" />
        <endpoint address="http://localhost:61183/Service.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_ICategoryService"
contract="ServiceApi.ICategoryService"
        name="BasicHttpBinding_ICategoryService" />
        <endpoint address="http://localhost:61183/Service.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_IProductService"
contract="ServiceApi.IProductService"
        name="BasicHttpBinding_IProductService" />
        <endpoint address="http://localhost:61183/Service.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_IOrderService"
contract="ServiceApi.IOrderService"
        name="BasicHttpBinding_IOrderService" />
        <endpoint address="http://localhost:61183/Service.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_IOrderDetailsService"
contract="ServiceApi.IOrderDetailsService"
        name="BasicHttpBinding_IOrderDetailsService" />
        <endpoint address="http://localhost:61183/Service.svc"
binding="basicHttpBinding"
        bindingConfiguration="BasicHttpBinding_ICartService"
contract="ServiceApi.ICartService"
        name="BasicHttpBinding_ICartService" />
    </client>
</system.serviceModel>
</configuration>
```

Services auxiliaries

Auth.cs

```
using System;
using System.Web;
using static WAMiCafesitoApp.Helpers.Auth;

namespace WAMiCafesitoApp.Helpers
{
    public class Auth
    {
        public Auth() { }

        public delegate void CallBack(string message);
        public int isAuthenticated()
        {
            int userId = 0;

            if (
                HttpContext.Current.Session["UserId"] != null &&
                int.TryParse(HttpContext.Current.Session["UserId"].ToString(), out
userId))
            {
                return userId;
            }

            return 0;
        }
        public int isAuthenticatedOrRedirect()
        {
            int userId = isAuthenticated();

            if (userId.Equals(0))
                HttpContext.Current.Response.Redirect("/login.aspx");
            return userId;
        }

        public int isAdminAuthorized(CallBack callback)
        {
            int userId = isAuthenticated();

            if (userId.Equals(0) || !(HttpContext.Current.Session["RoleId"] !=
null &&
                HttpContext.Current.Session["RoleId"].ToString().Equals("1")))
            {
                HttpContext.Current.Response.Redirect("/login.aspx");
            }

            return userId;
        }
    }
}
```

CartService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using WAMiCafesitoApp.Helpers;
using WAMiCafesitoApp.ServiceApi;

namespace WAMiCafesitoApp.Services
{
    public class CartService
    {
        private ICartService _cartService = new CartServiceClient();
        private readonly Auth _auth = new Auth();

        public List<Cart> AddOrUpdateCartItem(Product product, int quantity)
        {
            int userId = _auth.isAuthenticatedOrRedirect();
            if (!userId.Equals(0))
            {
                List<Cart> cartItems =
                    _cartService.GetCartItemsByUserId(userId).ToList();

                Cart cartItem = cartItems.FirstOrDefault(item => item.ID_Producto
                    == product.ID_Producto);

                if (cartItem == null)
                {
                    Cart cart = new Cart
                    {
                        ID_Usuario = userId,
                        ID_Producto = product.ID_Producto,
                        Cantidad = quantity,
                        PrecioUnitario = product.Precio,
                        Fecha = DateTime.Now
                    };

                    _cartService.AddCartItem(cart);
                }
                else
                {
                    _cartService.UpdateCartItemById(cartItem.ID_Carrito,
                        cartItem.Cantidad + quantity, cartItem.PrecioUnitario);
                }

                return GetCartItems();
            }

            public void UpdateCartItemQuantity(int id, int quantity)
            {
                int userId = _auth.isAuthenticatedOrRedirect();
                if (!userId.Equals(0))
                {
                    _cartService.UpdateCartItemQuantityById(id, quantity);
                }
            }
        }
    }
}
```

```
public List<Cart> GetCartItems()
{
    List<Cart> cartItems = new List<Cart>();
    int userId = _auth.isAuthenticated();
    if (!userId.Equals(0))
    {
        cartItems = _cartService.GetCartItemsByUserId(userId).ToList();
        return cartItems;
    }

    return cartItems;
}

public void DeleteCartItem(int id)
{
    int userId = _auth.isAuthenticated();
    if (!userId.Equals(0))
    {
        _cartService.DeleteCartItem(id);
    }
}

public void ClearCart()
{
    int userId = _auth.isAuthenticated();
    if (!userId.Equals(0))
    {
        _cartService.DeleteCartItemsByUserId(userId);
    }
}
}
```

Validator.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text.RegularExpressions;
using System.Web;

namespace WAMiCafesitoApp.Services
{
    public static class Validator
    {
        // Validar dirección de correo electrónico
        public static bool ValidateEmail(string email)
        {
            if (string.IsNullOrEmpty(email))
                return false;

            // Regular expression for validating email address
            string emailPattern = @"^[^@\s]+@[^@\s]+\.[^@\s]+$";
            return Regex.IsMatch(email, emailPattern);
        }
    }
}
```

```
// Validar cadena genérica (nombre, apellido, etc.)
public static bool ValidateString(string input, int minLength = 1, int
maxLength = 100)
{
    if (string.IsNullOrEmpty(input))
        return false;

    if (input.Length < minLength || input.Length > maxLength)
        return false;

    // Expresión regular para validar nombres con caracteres Unicode
    string namePattern = @"^[\p{L}\s'-]+$";
    return Regex.IsMatch(input, namePattern);
}

// Validar contraseña y confirmación de contraseña
public static bool ValidatePassword(string password, string
confirmPassword, int minLength = 8)
{
    if (string.IsNullOrEmpty(password) ||
string.IsNullOrEmpty(confirmPassword))
        return false;

    if (password.Length < minLength)
        return false;

    if (password != confirmPassword)
        return false;

    // Expresión regular para la validación de contraseñas (al menos un
número, una letra minúscula y una letra mayúscula)
    string passwordPattern = @"^(?=.*[a-z])(?=.*[A-Z])(?=.*\d).+$";
    return Regex.IsMatch(password, passwordPattern);
}

// Validar número (entero o de punto flotante)
public static bool ValidateNumber(string number)
{
    if (string.IsNullOrEmpty(number))
        return false;

    string numberPattern = @"^-?\d+(\.\d+)?$";
    return Regex.IsMatch(number, numberPattern);
}

// Validar fecha
public static bool ValidateDate(string date)
{
    if (string.IsNullOrEmpty(date))
        return false;

    return DateTime.TryParse(date, out _);
}

// Validar número de teléfono con formato de El Salvador
public static bool ValidatePhoneNumber(string phoneNumber)
{
    if (string.IsNullOrEmpty(phoneNumber))
```



```

        return false;

        // Expresión regular para validar número de teléfono de El Salvador
        string phoneNumberPattern = @"^(\\+?503)?\\d{8}$"; // Formato: (opcional
"+503")xxxxxxxx
        return Regex.IsMatch(phoneNumber, phoneNumberPattern);
    }

    // Validar cadena que no contenga inyección SQL, XSS, comandos de shell,
LDAP o XPath
    public static bool ValidateSafeString(string input)
    {
        if (string.IsNullOrEmpty(input))
            return false;

        // Patrones comunes de inyección SQL
        string[] sqlInjectionPatterns =
        {
            "--", ";--", ":", "/*", "*/", "@@"
        };

        // Patrón común de Cross-Site Scripting (XSS)
        string xssPattern = @"<.*?>|&#.*?;|&..*?;";

        // Verificar inyección SQL
        foreach (var pattern in sqlInjectionPatterns)
        {
            if (input.IndexOf(pattern, StringComparison.OrdinalIgnoreCase) >=
0)

                return true;
        }

        // Verificar XSS
        if (Regex.IsMatch(input, xssPattern, RegexOptions.IgnoreCase))
            return true;

        return false;
    }
}

```

Secciones tienda en línea

WAMiCafesitoApp/Store/Cart.aspx

```

<%@ Page Title="" Language="C#" MasterPageFile="~/Public.Master"
AutoEventWireup="true" CodeBehind="Cart.aspx.cs"
Inherits="WAMiCafesitoApp.Store.ShoppingCart" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="header" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <div class="container my-5">

```

```
<div class="d-flex align-items-start">
  <div class="col-md-9">
    <h2 class="mb-3">Carrito de Compras</h2>
    <asp:HiddenField ID="hdnToastMessage" runat="server" />
    <asp:HiddenField ID="hdnToastType" runat="server" />

    <asp:GridView ID="gvCart" runat="server"
      AutoGenerateColumns="false" CssClass="table cart-table">
      <Columns>
        <asp:TemplateField HeaderText="Product Name">
          <ItemTemplate>
            <div class="d-flex">
              <div class="product-img">
                <img src='<%# Eval("Imagen") %>' alt='<%#
Eval("Nombre") %>' onerror="this.onerror=null;
this.src='/Assets/Images/Default.png';" />
              </div>
              <div>
                <div class="product-name"><%#
Eval("Nombre") %></div>
                <div class="product-desc">
                  <%# Eval("Descripcion") %>
                </div>
                <div class="product-cat mt-3"><%#
Eval("Categoria") %></div>
              </div>
            </div>

            </ItemTemplate>
          </asp:TemplateField>

          <asp:BoundField
            DataField="PrecioUnitario"
            HeaderText="Precio"
            DataFormatString="{0:C}" />

          <asp:TemplateField HeaderText="Cantidad">
            <ItemTemplate>
              <div class="d-flex align-items-center px-3">
                <asp:TextBox ID="txtQuantity" runat="server"
Text='<%# Eval("Cantidad") %>' Width="38px" CssClass="form-control me-
2"></asp:TextBox>
                <asp:LinkButton ID="lnkUpdate" runat="server"
OnCommand="btnUpdate_Command" CommandArgument='<%# Eval("ID_Carrito") %>'>
                  <i class="fa-solid fa-arrows-rotate"></i>
                </asp:LinkButton>
              </div>
            </ItemTemplate>
          </asp:TemplateField>

          <asp:BoundField DataField="TotalPrice" HeaderText="Total"
DataFormatString="{0:C}" />
          <asp:TemplateField>
            <ItemTemplate>
              <asp:LinkButton ID="lnkDelete" runat="server"
OnCommand="btnDelete_Command" CommandArgument='<%# Eval("ID_Carrito") %>'
CssClass="text-danger">
```

```

                <i class="fa-regular fa-trash-can"></i>
            </asp:LinkButton>
        </ItemTemplate>
    </asp:TemplateField>
</Columns>
</asp:GridView>
<asp:Button ID="btnClearCart" CssClass="btn btn-danger btn-clear-
cart" OnClick="btnClearCart_Click" runat="server" Text="Limpiar Carrito" />
</div>
<div class="col-md-3 ms-4 cart-summary">
    <h6>Resumen de Orden</h6>
    <div class="d-flex justify-content-between cart-row">
        <span class="cart-label">Subtotal</span>
        <span class="cart-value">
            <asp:Label ID="lblSubTotal" runat="server"
Text="Label"></asp:Label>
        </span>
    </div>
    <div class="d-flex justify-content-between cart-row">
        <span class="cart-label">IVA 13%</span>
        <span class="cart-value">
            <asp:Label ID="lblTaxes" runat="server"
Text="Label"></asp:Label>
        </span>
    </div>
    <div class="d-flex justify-content-between cart-total">
        <span>Total</span>
        <span>
            <asp:Label ID="lblTotal" runat="server"
Text="Label"></asp:Label>
        </span>
    </div>
    <div class="cart-actions">
        <asp:Button ID="btnCheckout" CssClass="btn btn-light shadow-
none me-1 btn-add-to-cart px-4" OnClick="btnCheckout_Click" runat="server"
Text="Procesar Orden" />
    </div>
    <div class="cart-actions">
        <asp:LinkButton ID="lnkContinueShopping" runat="server"
OnClick="lnkContinueShopping_Click" CssClass="lnk-continue-shopping">Continuar
Comprando</asp:LinkButton>
    </div>
</div>
</div>
</asp:Content>
<asp:Content ID="Content4" ContentPlaceHolderID="FooterScripts" runat="server">
    <script src="../../Assets/Scripts/ErrMesasges.js"
type="text/javascript"></script>
    <script>
        var toastMessage = document.getElementById('<%= hdnToastMessage.ClientID
%>');
        var toastType = document.getElementById('<%= hdnToastType.ClientID %>');
        if (toastMessage) {
            showAnimatedToast(toastMessage, toastType);
        }
    </script>
</asp:Content>

```

WAMiCafesitoApp/Store/Cart.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WAMiCafesitoApp.ServiceApi;
using WAMiCafesitoApp.Services;

namespace WAMiCafesitoApp.Store
{
    public partial class ShoppingCart : System.Web.UI.Page
    {
        private CartService _cartService = new CartService();
        private IProductService productService = new ProductServiceClient();
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                LoadCartItems();
            }
        }

        private void LoadCartItems()
        {
            List<ServiceApi.Cart> cartItems = _cartService.GetCartItems();
            Double subTotal = cartItems.Sum(c => c.Cantidad * c.PrecioUnitario);
            Double taxes = subTotal * 0.13;
            Double total = subTotal + taxes;

            lblSubTotal.Text = $"{subTotal:C2}";
            lblTaxes.Text = $"{taxes:C2}";
            lblTotal.Text = $"{total:C2}";

            gvCart.DataSource = cartItems.Select(c =>
            {
                Product product = GetProduct(c.ID_Producto);
                return new
                {
                    c.ID_Carrito,
                    Imagen = $"/Assets/Images/{product.ID_Producto}.png",
                    product.Nombre,
                    product.Descripcion,
                    product.Categoria,
                    c.Cantidad,
                    c.PrecioUnitario,
                    TotalPrice = c.Cantidad * c.PrecioUnitario
                };
            }).ToList();
            gvCart.DataBind();
        }

        private Product GetProduct(int productId)
        {

```

```
        return productService.GetProductById(productId);
    }

    protected void DeleteCartItem(object sender, GridViewDeleteEventArgs e)
    {
        int cartId = (int)gvCart.DataKeys[e.RowIndex].Value;
        _cartService.DeleteCartItem(cartId);
        LoadCartItems();
        ShowErrorMessage("Producto eliminado del carrito de compras");
    }

    private void ShowErrorMessage(string message)
    {
        hdnToastMessage.Value = message;
        hdnToastType.Value = "info";
    }

    protected void btnClearCart_Click(object sender, EventArgs e)
    {
        _cartService.ClearCart();
        LoadCartItems();
        ShowErrorMessage("Se ha vaciado el carrito de compras");
    }

    protected void btnDelete_Command(object sender, CommandEventArgs e)
    {
        int cartId = Convert.ToInt32(e.CommandArgument);

        // Confirm deletion if confirmation dialog was displayed
        if (Page.ClientScript.IsClientScriptBlockRegistered("confirmDelete"))
        {
            bool confirmed = ScriptManager.GetCurrent(this).IsInAsyncPostBack;
            if (!confirmed)
                return; // User canceled deletion through confirmation dialog
        }

        _cartService.DeleteCartItem(cartId);

        LoadCartItems();
        ShowErrorMessage("Producto eliminado del carrito de compras");
    }

    protected void btnUpdate_Command(object sender, CommandEventArgs e)
    {
        int cartId = Convert.ToInt32(e.CommandArgument);
        int rowIndex =
            ((GridViewRow)((Control)sender).NamingContainer).RowIndex;
        GridViewRow row = gvCart.Rows[rowIndex];
        int quantity = int.Parse((row.FindControl("txtQuantity") as
            TextBox).Text);

        ServiceApi.Cart cartItem =
            _cartService.GetCartItems().FirstOrDefault(c => c.ID_Carrito == cartId);
        if (cartItem != null)
        {
            _cartService.UpdateCartItemQuantity(cartId, quantity);
        }
    }
}
```

```

    }
    gvCart.EditIndex = -1;

    LoadCartItems();
    ShowErrorMessage("Producto actualizado en el carrito de compras");
}

protected void btnCheckout_Click(object sender, EventArgs e)
{

}

protected void lnkContinueShopping_Click(object sender, EventArgs e)
{
    Response.Redirect("~/Default.aspx");
}
}
}

```

Utils

WAMiCafesitoApp/Store/Category.aspx

```

<%@ Page Title="" Language="C#" MasterPageFile="~/Public.Master"
AutoEventWireup="true" CodeBehind="Category.aspx.cs"
Inherits="WAMiCafesitoApp.Store.CategoryPage" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="header" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <section>
        <div class="container my-5">
            <header class="mb-4">
                <h3>
                    <asp:Label ID="lblCategoryName" runat="server"></asp:Label>
                </h3>
            </header>

            <asp:HiddenField ID="hdnToastMessage" runat="server" />
            <asp:HiddenField ID="hdnToastType" runat="server" />
            <div class="row">
                <asp:Repeater ID="featuredProductsRepeater" runat="server"
OnItemDataBound="featuredProductsRepeater_ItemDataBound">
                    <itemtemplate>
                        <div class="col-lg-3 col-md-6 col-sm-6 d-flex mb-5">
                            <div class="card w-100 my-2 shadow-2-strong">
                                <img src='/Assets/Images/<%# Eval("ID_Producto")
%>.png'
                                class="card-img-top"
                                style="aspect-ratio: 1 / 1"
                                onerror="this.onerror=null;
this.src='/Assets/Images/Default.png';"
                                />
                                <div class="card-body d-flex flex-column">
                                    <h5 class="card-title"><%# Eval("Nombre")
%></h5>

```

```

aria-hidden="true">
        <span class="a-price d-flex align-items-start"
            <span class="a-price-symbol">${</span>
            <span class="a-price-whole"><%#
Convert.ToInt32(Math.Floor(Convert.ToDouble(Eval("Precio")))) %></span>
            <span class="a-price-fraction"><%#
String.Format("{0:00}", GetFractionalPart(Eval("Precio"))) %></span>
            </span>
            <p class="card-text"><%# Eval("Descripcion",
"{0:C}") %></p>
        <div class="d-flex align-items-end py-2 px-0
            mt-auto gap-2">
                <asp:HyperLink ID="btnViewDetail"
                    runat="server"
                    CssClass="btn btn-light shadow-none
                <i class="fa-solid fa-magnifying-
                    glass"></i>Detalle
                </asp:HyperLink>
                <asp:LinkButton
                    CssClass="btn btn-light shadow-none
                    ID="btnAddToCart"
                    runat="server"
                    CommandName="AddToCart"
                    CommandArgument='<%#
Eval("ID_Producto") %>'
                    OnClick="btnAddToCart_Click">
                    <i class="fa-solid fa-cart-
shopping"></i>Ordenar</asp:LinkButton>
            </div>
        </div>
    </div>
</div>
</itemtemplate>
</asp:Repeater>
</div>
</div>
</section>
</asp:Content>
<asp:Content ID="Content4" ContentPlaceHolderID="FooterScripts" runat="server">
    <script src="../../Assets/Scripts/ErrMesasges.js"
type="text/javascript"></script>
    <script>
        var toastMessage = document.getElementById('<%= hdnToastMessage.ClientID
%>');
        var toastType = document.getElementById('<%= hdnToastType.ClientID %>');
        if (toastMessage) {
            showAnimatedToast(toastMessage, toastType);
        }
    </script>
</asp:Content>

```

WAMiCafesitoApp/Store/Category.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WAMiCafesitoApp.ServiceApi;
using WAMiCafesitoApp.Services;

namespace WAMiCafesitoApp.Store
{
    public partial class CategoryPage : System.Web.UI.Page
    {
        IProductService productService = new ProductServiceClient();
        ICategoryService categoryService = new CategoryServiceClient();
        private CartService _cartService = new CartService();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                if (!string.IsNullOrEmpty(Request.QueryString["cat"]))
                {
                    int categoryId;

                    if (int.TryParse(Request.QueryString["cat"], out categoryId))
                    {
                        Category category =
categoryService.GetCategoryById(categoryId);

                        if (category.Nombre != null)
                        {
                            lblCategoryName.Text = category.Nombre;
                        }
                        LoadProductsByCategoryId(categoryId);
                    }
                }
                else
                {
                    Response.Redirect("~/Default.aspx");
                }
            }
        }

        protected void LoadProductsByCategoryId(int catId)
        {
            List<Product> products =
productService.GetAllProductsByCategoryId(catId).ToList();
            if (products.Count > 0)
            {
                featuredProductsRepeater.DataSource = products;
                featuredProductsRepeater.DataBind();
            }
        }
    }
}
```



```
    }  
}  
  
protected string GetFractionalPart(object price)  
{  
    if (price != null && price != DBNull.Value)  
    {  
        decimal priceValue = Convert.ToDecimal(price);  
        int wholePart = (int)priceValue;  
        decimal fractionalPart = priceValue - wholePart;  
        int cents = (int)(fractionalPart * 100);  
        return cents.ToString("00");  
    }  
    return "00";  
}  
  
protected void featuredProductsRepeater_ItemDataBound(object sender,  
RepeaterItemEventArgs e)  
{  
    if (e.Item.ItemType == ListItemType.Item || e.Item.ItemType ==  
ListItemType.AlternatingItem)  
    {  
        HyperLink verDetalleLink =  
(HyperLink)e.Item.FindControl("btnViewDetail");  
  
        if (verDetalleLink != null)  
        {  
            Product product = (Product)e.Item.DataItem;  
  
            if (product != null)  
            {  
                verDetalleLink.NavigateUrl =  
$"/Store/ProductDetail.aspx?id={product.ID_Producto}";  
            }  
        }  
  
        LinkButton addToCartButton =  
(LinkButton)e.Item.FindControl("btnAddToCart");  
  
        if (addToCartButton != null)  
        {  
            Product product = (Product)e.Item.DataItem;  
  
            if (product != null)  
            {  
                addToCartButton.CommandArgument =  
product.ID_Producto.ToString();  
            }  
        }  
    }  
}  
  
protected void btnAddToCart_Click(object sender, EventArgs e)  
{  
    LinkButton btn = (LinkButton)sender;
```

```
int productId = Convert.ToInt32(btn.CommandArgument);

// Retrieve the product using the productId
ServiceApi.Product product = GetProductById(productId);

if (product != null)
{
    // Assuming you have a method to get the quantity, e.g., from a
    TextBox within the Repeater item
    RepeaterItem item = (RepeaterItem)btn.NamingContainer;
    TextBox txtQuantity = (TextBox)item.FindControl("txtQuantity");

    List<ServiceApi.Cart> cartItems =
    _cartService.AddOrUpdateCartItem(product, 1);

    // Update the session with the updated cart items
    Session["CartItems"] = cartItems;

    ShowErrorMessage($"Producto agregado al carrito.");
}
}

private Product GetProductById(int productId)
{
    return productService.GetProductById(productId);
}

private void ShowErrorMessage(string message)
{
    hdnToastMessage.Value = message;
    hdnToastType.Value = "info";
}
}
```

WAMiCafesitoApp/Store/ProductDetail.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Public.Master"
AutoEventWireup="true" CodeBehind="ProductDetail.aspx.cs"
Inherits="WAMiCafesitoApp.Store.ProductDetail" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="header" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <asp:HiddenField ID="hdnToastMessage" runat="server" />
    <asp:HiddenField ID="hdnToastType" runat="server" />
    <section>
        <div class="container my-5">
            <div class="row">
                <div class="col-lg-4">
                    <!-- Product Image with Lightbox -->
                    <a href="#" data-bs-toggle="modal" data-bs-
target="#imageModal">
                        <asp:Image ID="imgProduct" runat="server" CssClass="img-
fluid rounded" />
                    </a>
                </div>
            </div>
        </div>
    </section>
</asp:Content>
```

```

        </a>

        <!-- Lightbox Modal -->
        <div class="modal fade" id="imageModal" tabindex="-1" aria-
labelledby="imageModalLabel" aria-hidden="true">
            <div class="modal-dialog modal-lg p-0">

                <div class="modal-content p-0">
                    <button type="button" class="modal-close-icon"
data-bs-dismiss="modal">

                        <i class="fa-regular fa-circle-xmark"></i>
                    </button>
                    <div class="modal-body p-0">
                        <img id="modalImage" class="img-fluid"
alt="Product Image">

                    </div>
                </div>
            </div>
        </div>
        <div class="col-lg-7 ps-4 pt-4">
            <h2 class="mb-0">
                <asp:Label ID="lblProductName" runat="server"
Text=""></asp:Label>
            </h2>
            <div class="d-flex align-items-center gap-2 mb-4">
                <strong>Categoria:</strong>
                <asp:Label ID="lblCategory" runat="server"
Text=""></asp:Label>
            </div>
            <p class="lead mb-4">
                <asp:Label ID="lblProductDescription" runat="server"
Text=""></asp:Label>
            </p>
            <p class="lead">
                <strong>Precio:</strong>
                <span class="a-price d-flex align-items-start" aria-
hidden="true">

                    <span class="a-price-symbol">$</span>
                    <asp:Label ID="lblPriceInteger" runat="server"
CssClass="a-price-whole" Text=""></asp:Label>
                    <asp:Label ID="lblPriceDecimal" runat="server"
CssClass="a-price-fraction" Text=""></asp:Label>
                </span>
            </p>

            <div class="mb-4">
                <!-- Quantity selection -->
                <asp:Label ID="lblQuantity" runat="server"
AssociatedControlID="txtQuantity" Text="Cantidad:" CssClass="form-
label"></asp:Label>

                <div class="d-flex align-items-center gap-3">
                    <asp:TextBox ID="txtQuantity" runat="server"
CssClass="form-control max-80" Text="1" Type="Number" Min="1"></asp:TextBox>
                    <asp:LinkButton ID="btnAddToCart" runat="server"
CssClass="btn btn-light shadow-none me-1 btn-add-to-cart px-4"
OnClick="btnAddToCart_Click"><i class="fa-solid fa-cart-shopping"></i>
Ordenar</asp:LinkButton>
                </div>
            </div>

```

```
</div>

</div>
</div>
</div>
</section>
</asp:Content>

<asp:Content ID="Content4" ContentPlaceHolderID="FooterScripts" runat="server">
  <script src="../../Assets/Scripts/ErrMesasges.js"
type="text/javascript"></script>
  <script>
    var toastMessage = document.getElementById('<%= hdnToastMessage.ClientID
%>');
    var toastType = document.getElementById('<%= hdnToastType.ClientID %>');
    if (toastMessage) {
      showAnimatedToast(toastMessage, toastType);
    }
  </script>
</asp:Content>
```

WAMiCafesitoApp/Store/ProductDetail.aspx.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WAMiCafesitoApp.Helpers;
using WAMiCafesitoApp.ServiceApi;
using WAMiCafesitoApp.Services;

namespace WAMiCafesitoApp.Store
{
    public partial class ProductDetail : System.Web.UI.Page
    {
        private CartService _cartService = new CartService();
        private IProductService productService = new ProductServiceClient();
        private Auth auth = new Auth();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                if (!string.IsNullOrEmpty(Request.QueryString["id"]))
                {
                    int productId;

                    if (int.TryParse(Request.QueryString["id"], out productId))
                    {
                        Product product =
productService.GetProductById(productId);
                        Session["SelectedProduct"] = product;
                    }
                }
            }
        }
    }
}
```

```
        if (product != null)
        {
            RenderProductDetail(product);
            Session["Product"] = product;
        }
        else
        {
            Response.Write("Product not found.");
        }
    }
}
else
{
    Response.Redirect("~/Default.aspx");
}
}

protected void RenderProductDetail(Product product)
{
    if (product != null)
    {
        lblProductName.Text = product.Nombre;
        lblProductDescription.Text = product.Descripcion;
        lblCategory.Text = product.Categoria;
        string imageUrl = $"/Assets/Images/{product.ID_Producto}.png";
        imgProduct.ImageUrl = imageUrl;
        imgProduct.Attributes.Add("onerror", "this.onerror=null;
this.src='/Assets/Images/default.png'");
        setPrice(product.Precio);
    }
    else
    {
        Response.Write("Product not found.");
    }
}

protected void setPrice(double precio)
{
    int priceInteger = (int)precio;
    int priceDecimal = (int)((precio - priceInteger) * 100);

    lblPriceInteger.Text = priceInteger.ToString();
    lblPriceDecimal.Text = priceDecimal.ToString("00");
}

protected void btnAddToCart_Click(object sender, EventArgs e)
{
    Product product = Session["SelectedProduct"] as Product;
    int quantity = Convert.ToInt32(txtQuantity.Text);

    if (product != null && quantity != 0)
    {
        List<ServiceApi.Cart> cartItems =
        _cartService.AddOrUpdateCartItem(product, quantity);
    }
}
```

```
// Update the session with the updated cart items
Session["CartItems"] = cartItems;

string plural = quantity > 1 ? "s" : "";

ShowErrorMessage(
    $"{quantity} Producto{plural}" +
    $" agregado{plural} al carrito."
);
}

}

private void ShowErrorMessage(string message)
{
    hdnToastMessage.Value = message;
    hdnToastType.Value = "info";
}
}
}
```

WAMiCafesitoApp/Store/ResultsPage.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Public.Master"
AutoEventWireup="true" CodeBehind="ResultsPage.aspx.cs"
Inherits="WAMiCafesitoApp.Store.ResultsPage" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="header" runat="server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="MainContent" runat="server">
    <section>
        <div class="container my-5">
            <header class="mb-4">
                <h3>
                    <asp:Label ID="lblCategoryName" runat="server"></asp:Label>
                </h3>
            </header>

            <asp:HiddenField ID="hdnToastMessage" runat="server" />
            <asp:HiddenField ID="hdnToastType" runat="server" />
            <div class="row">
                <asp:Repeater ID="featuredProductsRepeater" runat="server"
OnItemDataBound="featuredProductsRepeater_ItemDataBound">
                    <itemtemplate>
                        <div class="col-lg-3 col-md-6 col-sm-6 d-flex mb-5">
                            <div class="card w-100 my-2 shadow-2-strong">
                                <img src='/Assets/Images/<%# Eval("ID_Producto")
%>.png'
                                class="card-img-top"
                                style="aspect-ratio: 1 / 1"
                                onerror="this.onerror=null;
this.src='/Assets/Images/Default.png';"
                                />
                            <div class="card-body d-flex flex-column">
```

```

%></h5>
aria-hidden="true">
        <span class="a-price d-flex align-items-start">
            <span class="a-price-symbol">$</span>
            <span class="a-price-whole"><%#
Convert.ToInt32(Math.Floor(Convert.ToDouble(Eval("Precio")))) %></span>
            <span class="a-price-fraction"><%#
String.Format("{0:00}", GetFractionalPart(Eval("Precio"))) %></span>
            <p class="card-text"><%# Eval("Descripcion",
"{0:C}") %></p>
        <div class="d-flex align-items-end py-2 px-0
mt-auto gap-2">
            <asp:HyperLink ID="btnViewDetail"
                runat="server"
                CssClass="btn btn-light shadow-none
                <i class="fa-solid fa-magnifying-
glass"></i>Detalle
            </asp:HyperLink>
            <asp:LinkButton
                CssClass="btn btn-light shadow-none
                ID="btnAddToCart"
                runat="server"
                CommandName="AddToCart"
                CommandArgument='<%#
Eval("ID_Producto") %>'
                OnClick="btnAddToCart_Click">
                <i class="fa-solid fa-cart-
shopping"></i>Ordenar</asp:LinkButton>
        </div>
    </div>
</div>
</div>
</itemtemplate>
</asp:Repeater>
</div>
</div>
</section>
</asp:Content>
<asp:Content ID="Content4" ContentPlaceHolderID="FooterScripts" runat="server">
    <script src="../../../Assets/Scripts/ErrMesasges.js"
type="text/javascript"></script>
    <script>
        var toastMessage = document.getElementById('<%= hdnToastMessage.ClientID
%>');
        var toastType = document.getElementById('<%= hdnToastType.ClientID %>');
        if (toastMessage) {
            showAnimatedToast(toastMessage, toastType);
        }
    </script>
</asp:Content>

```

WAMiCafesitoApp/Store/ResultsPage.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WAMiCafesitoApp.ServiceApi;
using WAMiCafesitoApp.Services;

namespace WAMiCafesitoApp.Store
{
    public partial class ResultsPage : System.Web.UI.Page
    {
        IProductService productService = new ProductServiceClient();
        ICategoryService categoryService = new CategoryServiceClient();
        private CartService _cartService = new CartService();

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                if (!string.IsNullOrEmpty(Request.QueryString["criteria"]))
                {
                    string criteria = Request.QueryString["criteria"];
                    if (!Validator.ValidateSafeString(criteria)) {
                        LoadProductsBySearchCriteria(criteria);
                    } else
                    {
                        ShowErrorMessage(criteria + " no es un criterio de
búsqueda válido.");
                    }
                }
                else
                {
                    Response.Redirect("~/Default.aspx");
                }
            }
        }

        protected void LoadProductsBySearchCriteria(string criteria)
        {
            List<Product> products =
productService.GetProductsByName(criteria).ToList();
            if (products.Count > 0)
            {
                featuredProductsRepeater.DataSource = products;
                featuredProductsRepeater.DataBind();
            } else
            {
                ShowErrorMessage(criteria + " no es un criterio de búsqueda
válido.");
            }
        }
    }
}
```



```
    }
}

protected string GetFractionalPart(object price)
{
    if (price != null && price != DBNull.Value)
    {
        decimal priceValue = Convert.ToDecimal(price);
        int wholePart = (int)priceValue;
        decimal fractionalPart = priceValue - wholePart;
        int cents = (int)(fractionalPart * 100);
        return cents.ToString("00");
    }
    return "00";
}

protected void featuredProductsRepeater_ItemDataBound(object sender,
RepeaterItemEventArgs e)
{
    if (e.Item.ItemType == ListItemType.Item || e.Item.ItemType ==
ListItemType.AlternatingItem)
    {
        HyperLink verDetalleLink =
(HyperLink)e.Item.FindControl("btnViewDetail");

        if (verDetalleLink != null)
        {
            Product product = (Product)e.Item.DataItem;

            if (product != null)
            {
                verDetalleLink.NavigateUrl =
$"/Store/ProductDetail.aspx?id={product.ID_Producto}";
            }
        }

        LinkButton addToCartButton =
(LinkButton)e.Item.FindControl("btnAddToCart");

        if (addToCartButton != null)
        {
            Product product = (Product)e.Item.DataItem;

            if (product != null)
            {
                addToCartButton.CommandArgument =
product.ID_Producto.ToString();
            }
        }
    }
}

protected void btnAddToCart_Click(object sender, EventArgs e)
{
    LinkButton btn = (LinkButton)sender;
    int productId = Convert.ToInt32(btn.CommandArgument);
}
```

```
// Retrieve the product using the productId
ServiceApi.Product product = GetProductById(productId);

if (product != null)
{
    // Assuming you have a method to get the quantity, e.g., from a
    TextBox within the Repeater item
    RepeaterItem item = (RepeaterItem)btn.NamingContainer;
    TextBox txtQuantity = (TextBox)item.FindControl("txtQuantity");

    List<ServiceApi.Cart> cartItems =
    _cartService.AddOrUpdateCartItem(product, 1);

    // Update the session with the updated cart items
    Session["CartItems"] = cartItems;


    ShowErrorMessage($"Producto agregado al carrito.");
}

private Product GetProductById(int productId)
{
    return productService.GetProductById(productId);
}

private void ShowErrorMessage(string message)
{
    hdnToastMessage.Value = message;
    hdnToastType.Value = "info";
}
}
```

Capturas de pantalla

Página principal (Vista completa)




[Ingresar](#) [Registrarse](#) [Carrito](#)

[Filtros](#) | [Cafeteras de Espresso](#) | [Molinillos de Café](#) | [Métodos de Infusión Directa](#) | [Métodos de Filtrado](#) | [Otros Métodos Especiales](#) | [Accesorios](#) | [Insufumos](#)


¡Descubre el V60 Hario!

Disfruta descubriendo tus granos favoritos, método de preparación e inventando tu receta.


[Ordena Ahora](#) [Más Información](#)




PRODUCTOS DESTACADOS




Prensa Francesa
\$25⁰⁰
Cafetera de émbolo para preparar café de infusión directa con filtrado de prensa.
[Agregar al carrito](#)




Aeropress
\$30⁰⁰
Dispositivo de preparación de café de infusión directa que utiliza presión de aire.
[Agregar al carrito](#)




Moka Pot
\$35⁰⁰
Cafetera de aluminio para preparar café espresso en la estufa.
[Agregar al carrito](#)




Chemex
\$46⁰⁰
Cafetera de vidrio con filtro de papel grueso para preparar café de sabor limpio.
[Agregar al carrito](#)




V60
\$20⁰⁰
Cono de goteo de cerámica con espirales internas para preparar café de filtro.
[Agregar al carrito](#)



Kalita Wave
\$30⁰⁰
Dripper de café de acero inoxidable con fondo plano y canales de extracción.
[Agregar al carrito](#)




Clever Dripper
\$25⁰⁰
Dripper de café con válvula de drenaje que permite el control del tiempo de extracción.
[Agregar al carrito](#)




Báscula de precisión
\$30⁰⁰
Báscula digital precisa para medir la cantidad de café y agua.
[Agregar al carrito](#)


Por qué comprar con nosotros




Precios razonables
Disfruta de productos asequibles para una experiencia de café excepcional y deliciosa.




Mejor calidad
Garantizamos la excelencia en cada taza con productos de calidad superior y deliciosa.




Envíos a todo El Salvador
Enviamos nuestros productos a cualquier lugar, para que puedas disfrutar del mejor café en casa, sin importar dónde estés!




Satisfacción del cliente
Tu felicidad es nuestra prioridad, te aseguramos una experiencia de compra satisfactoria.



Clientes satisfechos
Nuestros clientes hablan por nosotros, proporcionamos atención excepcional y productos de alta calidad para que disfrutes al máximo de tu café.



Todo lo que necesitas
Explora nuestra amplia selección de productos, desde cafeteras hasta accesorios, todo lo que necesitas para tu café perfecto!



© 2023 Copyright: MDBootstrap.com

STORE
[Sobre Nosotros](#)
[Encuentranos](#)
[Categorías](#)

INFORMATION
[Ayuda](#)
[Información de envío](#)
[Refund](#)







Imagen principal y menú

 **MiCafesito**
En línea



 Carrito

[Salir](#)

[Cafeteras de Espresso](#) | [Molinillos de Café](#) | [Métodos de Infusión Directa](#) | [Métodos de Filtrado](#) | [Otros Métodos Especiales](#) | [Accesorios](#) | [Insumos](#)

¡Descubre el V60 Hario!

Disfruta descubriendo tus granos favoritos, método de preparación e inventando tu receta..

[Más Información](#)

PRODUCTOS DESTACADOS

Área de productos


PRODUCTOS DESTACADOS



Prensa Francesa

\$25⁹⁹


Cafetera de émbolo para preparar café de infusión directa con filtrado de prensa.

 Agregar al carrito

Aeropress

\$30⁹⁹


Dispositivo de preparación de café de infusión directa que utiliza presión de aire.

 Agregar al carrito

Moka Pot

\$35⁹⁹

Cafetera de aluminio para preparar café espresso en la estufa.

 Agregar al carrito

Chemex

\$46⁹⁹

Cafetera de vidrio con filtro de papel grueso para preparar café de sabor limpio.

 Agregar al carrito

Página principal

Por qué comprar con nosotros



Precios razonables

Disfruta de productos asequibles para una experiencia de café excepcional y deliciosa.



Mejor calidad

Garantizamos la excelencia en cada taza con productos de calidad superior y deliciosa.



Envíos a todo El Salvador

Enviamos nuestros productos a cualquier lugar, para que puedas disfrutar del mejor café en casa, ¡sin importar dónde estés!



Satisfacción del cliente

Tu felicidad es nuestra prioridad, te aseguramos una experiencia de compra satisfactoria.



Clientes satisfechos

Nuestros clientes hablan por nosotros, proporcionamos atención excepcional y productos de alta calidad para que disfrutes al máximo de tu café.



Todo lo que necesitas

Explora nuestra amplia selección de productos, desde cafeteras hasta accesorios, ¡todo lo que necesitas para tu café perfecto!

**MiCafesito**
En línea

© 2023 Copyright: MDBootstrap.com

STORE

[Sobre Nosotros](#)
[Encuétranos](#)
[Categorías](#)

INFORMATION

[Ayuda](#)
[Información de envío](#)
[Refunds](#)

Página Login

**MiCafesito**
En línea

Ingresa a tu cuenta

Correo electrónico

Contraseña

Ingresar







[¿Olvidaste tu contraseña?](#)[¿No tienes una cuenta? Regístrate](#)

Carrito de compras

MiCafesito
Enlinea Carrito[Salir](#)Cafeteras de
Espresso |Molinillos de
Café |Métodos de Infusión
Directa |Métodos de
Filtrado |Otros Métodos
Especiales |

Accesorios | Insumos

Carrito de Compras

Product Name	Precio	Cantidad	Total
 Cafeteras de espresso Máquina de café para hacer espresso con sistema de bombeo y vaporizador de leche integrado. Cafeteras de Espresso	\$299.99	1 	\$299.99 
 Aeropress Dispositivo de preparación de café de infusión directa que utiliza presión de aire. Métodos de Infusión Directa	\$29.99	2 	\$59.98 

[Limpiar Carrito](#)

Resumen de Orden

Subtotal \$359.97

IVA 13% \$46.80

Total \$406.77[Procesar Orden](#)[Continuar Comprando](#)

Proceso de orden

MiCafesito
Enlinea Carrito[Salir](#)Cafeteras de
Espresso |Molinillos de
Café |Métodos de Infusión
Directa |Métodos de
Filtrado |Otros Métodos
Especiales |

Accesorios | Insumos

Pedido realizado con éxito

Gracias por tu compra, tu pedido ha sido enviado con éxito.
En breve uno de nuestros agentes comerciales se comunicará contigo.

[Ver historial de pedidos](#)

Menu de opciones

MiCafesito
En línea Carrito

Salir

Cafeteras de
Espresso |Molinillos de
Café |Métodos de Infusión
Directa |Métodos de
Filtrado |Otros Métodos
Especiales |

Accesorios | Insumos

Métodos de Infusión Directa



Prensa Francesa

\$24⁹⁹

Aeropress

\$29⁹⁹

Moka Pot

\$34⁹⁹

Detalle del producto

MiCafesito
En línea Carrito

Salir

Cafeteras de
Espresso |Molinillos de
Café |Métodos de Infusión
Directa |Métodos de
Filtrado |Otros Métodos
Especiales |

Accesorios | Insumos



Prensa Francesa

Categoría: Métodos de Infusión Directa

Cafetera de émbolo para preparar café de infusión directa con filtrado de prensa.

Precio:

\$24⁹⁹

Cantidad:

1

 Ordenar

Pedidos realizados

Pedidos | Productos | Categorías | Usuarios

Orders

ID	Producto	Fecha	Estado	SubTotal	Factura	
5	Raúl Escamilla	mayo 21, 2024 - 00:00 a. m.	Recibido	\$0.00	N/A	Ver Detalle 
6	Raúl Escamilla	mayo 21, 2024 - 13:02 p. m.	Recibido	\$0.00	N/A	Ver Detalle 
7	Raúl Escamilla	mayo 21, 2024 - 13:18 p. m.	Recibido	\$94.97	N/A	Ver Detalle 
8	Raúl Escamilla	mayo 21, 2024 - 15:11 p. m.	Recibido	\$49.98	N/A	Ver Detalle 
9	Raúl Escamilla	mayo 21, 2024 - 15:24 p. m.	Recibido	\$359.97	N/A	Ver Detalle 
10	Raúl Escamilla	mayo 21, 2024 - 15:35 p. m.	Recibido	\$24.99	N/A	Ver Detalle 

Conclusiones:

Satisfacción de una Necesidad Específica: La creación de MiCafecito surge de la identificación de una necesidad en el mercado de suministros y productos relacionados con el café. Al desarrollar esta aplicación, se ofrece una solución digital que facilita a los amantes del café la compra de suministros y la obtención de información útil sobre la elaboración de esta bebida.

Enfoque en la Experiencia del Usuario: La aplicación se diseñó con un enfoque centrado en el usuario, priorizando la facilidad de navegación, la claridad en la presentación de productos y la incorporación de contenido educativo relevante. Esto contribuye a una experiencia positiva y satisfactoria para los usuarios, lo que puede fomentar la fidelización y el boca a boca positivo.

Innovación y Adaptación a las Tendencias: MiCafecito incorpora características innovadoras, como la integración de contenido educativo y la atención a la cliente personalizada. Además, se diseñó con la flexibilidad necesaria para adaptarse a las tendencias cambiantes del mercado y las demandas de los consumidores, lo que le permite mantenerse relevante a lo largo del tiempo.

Compromiso con la Calidad y la Seguridad: Se ha prestado especial atención a la calidad y seguridad de la aplicación, desde el proceso de desarrollo hasta la implementación de medidas de protección de datos y transacciones seguras. Esto contribuye a generar confianza entre los usuarios y a garantizar una experiencia de compra segura y satisfactoria.

Planificación Estratégica y Gestión Eficiente: La definición de objetivos claros, la elaboración de un cronograma detallado y la identificación de limitaciones temporales y espaciales han sido

fundamentales para el éxito del proyecto. La planificación estratégica y la gestión eficiente de recursos han permitido cumplir con los plazos establecidos y alcanzar los objetivos propuestos.

Algunas recomendaciones sobre el proyecto:

Investigación de Mercado: realizar una investigación exhaustiva del mercado para comprender las necesidades y preferencias de la audiencia objetivo. Analizar a la competencia para identificar oportunidades y áreas de diferenciación.

Definición de Objetivos Claros: Establecer objetivos claros y medibles para el proyecto MiCafecito. Esto ayudará a mantener un enfoque claro y a evaluar el éxito del proyecto una vez que esté en funcionamiento.

Identificación de Funcionalidades Clave: Definir las funcionalidades principales que se requiere que tenga la aplicación, como la navegación de productos, el proceso de compra, el sistema de registro y login, y el contenido educativo sobre café.

Diseño Centrado en el Usuario: Priorizar la experiencia del usuario al diseñar la interfaz de usuario y la experiencia de usuario. Asegurar de que la aplicación sea fácil de usar, intuitiva y estéticamente atractiva para los usuarios.

Seguridad y Privacidad: Asegurarse de implementar medidas sólidas de seguridad y privacidad para proteger la información sensible de los usuarios, como datos personales y financieros.

Pruebas Rigurosas: Realizar pruebas exhaustivas de la aplicación en todas las etapas del desarrollo para identificar y corregir errores y problemas de usabilidad.

Bibliografía:

Statista. (2022). Global retail e-commerce sales from 2014 to 2025 (in trillion U.S. dollars).

<https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/>

Adobe. (2021). Digital Economy Index: Q1 2021 [Informe]. Adobe.