Supuesto práctico integrador
Desarrollo de una aplicación Web interactiva de tienda online

Objetivos

- Integrar y aplicar los conocimientos adquiridos en el módulo de Desarrollo de Aplicaciones Web en el Entorno del Servidor del ciclo formativo de Desarrollo de Aplicaciones Web.
- Abordar y solucionar los problemas tecnológicos que comporta completar el desarrollo de una aplicación Web interactiva con acceso a datos, empleando la tecnología ASP.NET Core MVC.
- Comprender el proceso del desarrollo de una aplicación Web con un alto nivel de acabado.

Enunciado

Se desea construir una **aplicación Web interactiva para la venta online de productos**. Esta aplicación Web se construirá utilizando la tecnología de desarrollo en entorno del servidor ASP.NET Core MVC y el servidor de base de datos SQL Server para resolver el almacenamiento de datos. La construcción de la aplicación Web se realizará de modo que se obtenga **el mayor nivel de acabado** posible.

La aplicación web interactiva de tienda online que se va a construir, deberá quedar adaptada a la venta de un tipo de productos concreto, por ejemplo: artículos de ferretería, pendas de vestir, videojuegos, vinos, material deportivo, etc. Por este motivo, cada alumno deberá elegir el tipo de productos que se venderán en la tienda online que vaya a desarrollar. Convendrá previamente consultar la idea a desarrollar con el profesor para verificar la idoneidad de los objetivos a alcanzar.

La aplicación Web resultante deberá permitir el acceso a los procesamientos de gestión a realizar por los usuarios comunes de la aplicación Web, así como a los procesamientos de administración o gestión avanzada. De modo que **se definirán, al menos, dos perfiles o casos de uso**: el perfil de Usuario y el perfil de Administrador. Según el perfil del usuario autentificado, se accederá a una parte u otra de la aplicación Web. Los usuarios comunes accederán al menú de usuario, dónde estarán disponibles las opciones de acceso a los procesamientos de compra y gestión de pedidos del usuario. Los administradores de la aplicación Web accederán al menú de administración que incluirá las opciones para el mantenimiento de la información y los procesamientos de gestión avanzada *(Back-office)*.

Desarrollo

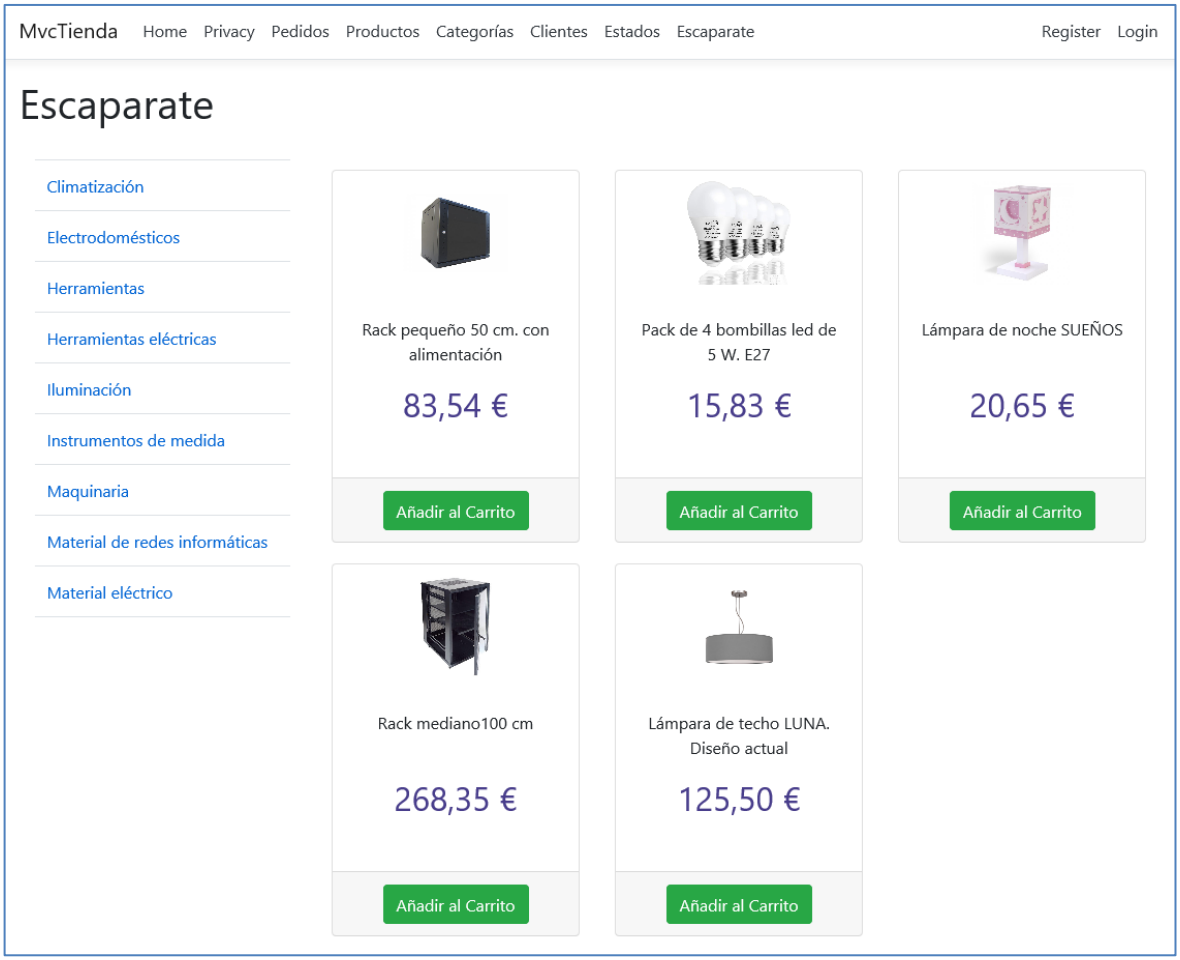
Para abordar la construcción de la aplicación Web de tienda online, que deberá completarse con un elevado nivel de acabado, se propone la realización de las siguientes tareas:

- Seleccionar el tipo de productos que se venderán a través de la aplicación Web de tienda online y consultarlo con el profesor. Además, elegir un nombre para la aplicación Web.
- Crear una aplicación Web de ASP.NET Core MVC con autentificación de Cuentas individuales.
- Crear el Modelo, teniendo en cuenta los requerimientos de la tienda online concreta a desarrollar en cada caso. Para ello, se puede considerar el ejemplo genérico para tienda online de las clases de datos y de la clase del contexto de datos que se incluye en el anexo.

- Registrar el contexto de datos en el archivo *Program.cs* para asignarle la cadena de conexión **DefaultConnection**. Además, asignar la instancia de base de datos *(localdb)\\MSSQL15* en la cadena de conexión **DefaulConnection** que está especificada en el archivo *appsettings.json*.
- Realizar una migración inicial para crear la base de datos y comprobar que se ha creado.
- Crear y adecuar los controladores, y sus vistas asociadas, para generar los procesamientos de tipo CRUD de cada una de las tablas de la base de datos que maneja la aplicación Web.
- Crear las opciones de menú correspondientes para cada uno de los controladores creados.
- Introducir y almacenar información suficiente en cada una de las tablas para poder realizar las pruebas correspondientes. Se recomienda introducir 10 filas en cada tabla como mínimo.
- En el controlador ***Productos*** crear la acción *CambiarImagen(int? id)* y la vista correspondiente, para modificar la imagen de un producto. Añadir la opción "Cambiar Imagen" en la lista de Productos. Ver el documento sobre el tratamiento de imágenes de productos.

Construcción del escaparate

- Crear un nuevo controlador denominado ***Escaparate***. A continuación, añadir la acción *Index()*, y la vista correspondiente, para implementar la funcionalidad de mostrar la lista de productos del escaparate de la tienda, de modo que se obtenga un resultado similar al siguiente.



La acción *Index()* del controlador *Escaparate* recibe como parámetro el valor del Id de la categoría de productos que se desea mostrar. Si el valor del Id recibido como parámetro es *null*, entonces se mostrarán los productos del escaparate, es decir, aquellos que tienen el valor *true* en la propiedad *Producto.Escaparate*. Para pasar la lista de las categorías a la vista desde el modelo se empleará el objeto *ViewData*, una vez obtenida la lista de categorías ordenada en el modelo. En la vista, se presentarán los enlaces correspondientes a cada categoría en una tabla, de modo que se pueda seleccionar una categoría concreta para ver los productos que correspondan a esa categoría. Añadir la opción "Escaparate" en el menú de la aplicación Web.

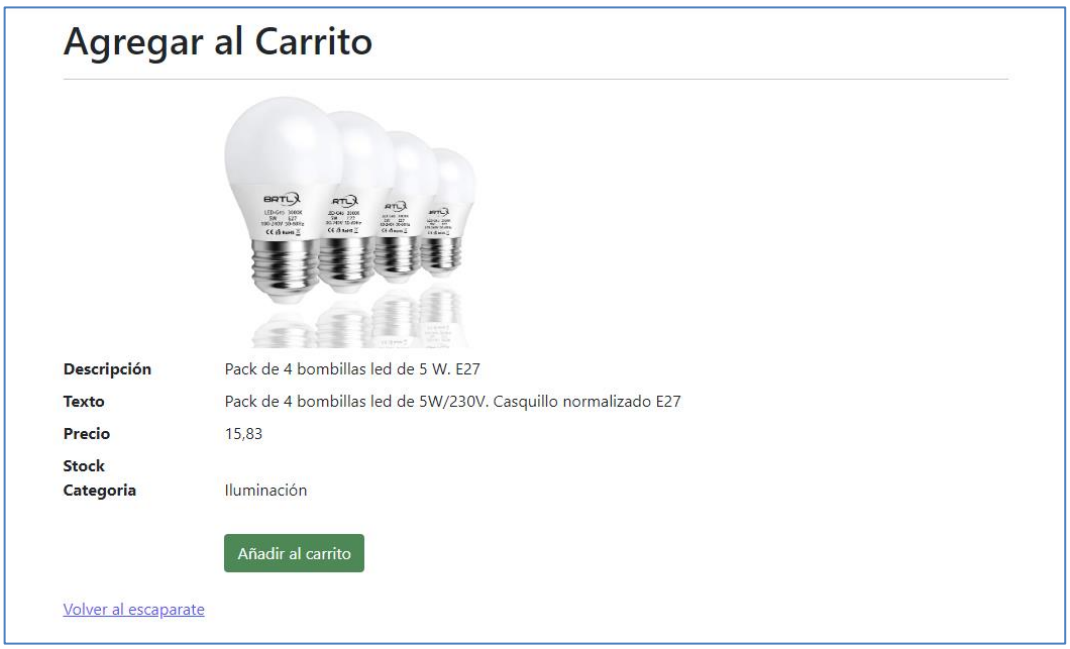
- Incorporar el sistema de autentificación y autorización de usuarios basado en *ASP.NET Core Identity* y enlazar la información de los usuarios registrados con la información almacenada en la base de datos que maneja la aplicación Web, a través del correo electrónico del usuario y del cliente. Para ello, desarrollar el controlador *MisDatos* y las acciones GET *Create()* y POST *Create()* para almacenar los datos del cliente correspondiente al usuario actual, en caso de que estos datos no estén ya almacenados. A continuación, desarrollar el procesamiento para modificar los datos del cliente correspondiente al usuario actual, añadiendo para ello los métodos GET *Edit()* y POST *Edit()* correspondientes en el controlador *MisDatos*.
- Establecer el menú de la aplicación Web según el rol del usuario que accede a la aplicación Web, de acuerdo con la siguiente tabla.

Opción de menú	Rol del usuario	Controlador	Método de acción
Inicio	Cualquiera	*Home*	*Index()*
Privacidad	Cualquiera	*Home*	*Privacy()*
Estados	Administrador	*Estados*	*Index()*
Categorías	Administrador	*Categorias*	*Index()*
Productos	Administrador	*Productos*	*Index()*
Clientes	Administrador	*Clientes*	*Index()*
Pedidos	Administrador	*Pedidos*	*Index()*
Escaparate	Usuario	*Escaparate*	*Index()*
Mis Datos	Usuario	*MisDatos*	*Edit()*

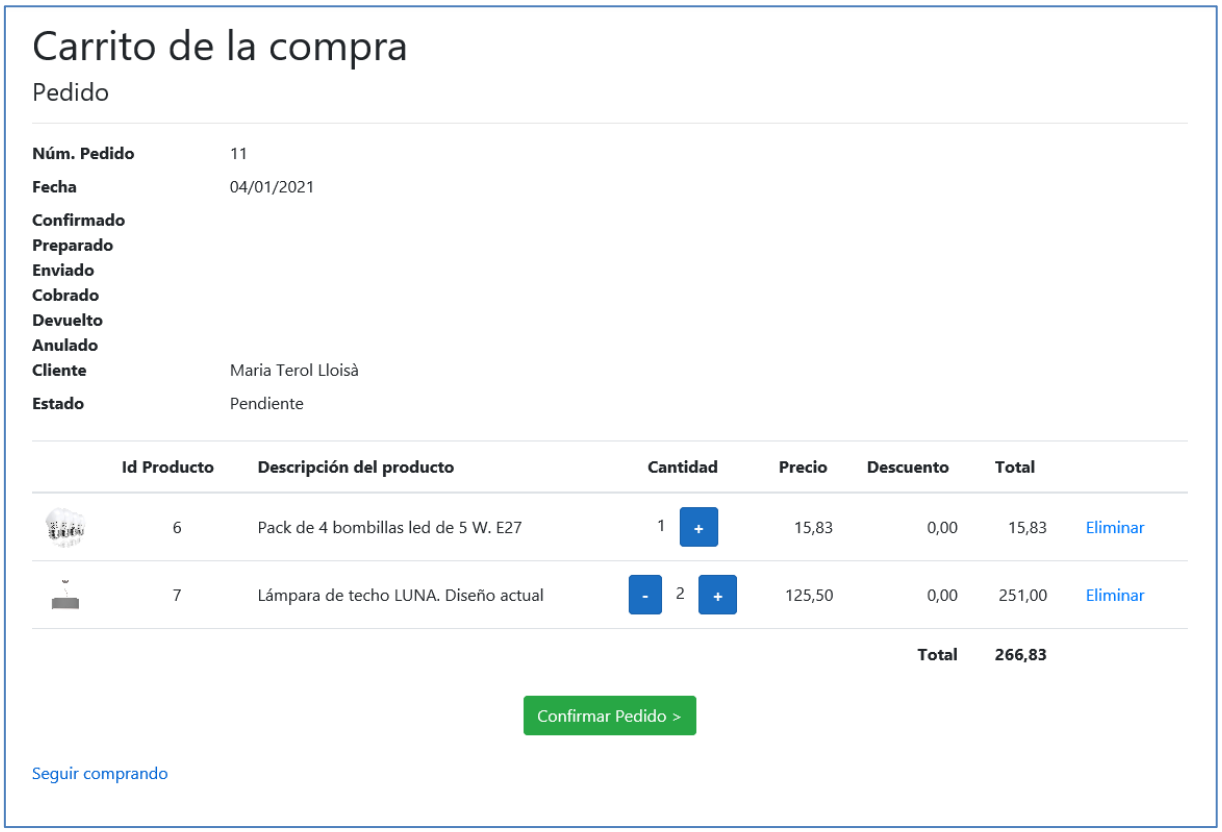
- Teniendo en cuenta la tabla anterior, especificar el filtro *Autorize* en los controladores y/o en las acciones que corresponda, de manera que se puedan evitar los accesos indebidos.

Construcción del carrito de la compra

- Implementar las acciones GET *AgregarCarrito()* y POST *AgregarCarrito()* en el controlador *Escaparate* para añadir un producto al carrito de la compra desde el escaparate, a través del botón "Añadir al Carrito" de cada producto. En ambas acciones, se recibe como parámetro el valor del Id del producto a añadir. La acción GET, y la vista correspondiente, mostrará los datos del producto a añadir y un botón de confirmación que permitirá iniciar la acción POST, tal como se muestra en la siguiente ilustración. Si se trata del primer producto que se añade al carrito, entonces debe crearse un nuevo pedido. Al crear un nuevo pedido, se le asignará el cliente correspondiente al usuario actual y se establecerá el valor "Pendiente" al estado del pedido. Además, se guardará el número del pedido en la variable de sesión denominada *NumPedido*.



- Crear un nuevo controlador denominado ***Carrito***. A continuación, añadir la acción *Index()*, y la vista correspondiente, para visualizar el carrito de la compra del pedido actual, a través de un procesamiento de tipo maestro-detalle, tal como se muestra en la siguiente ilustración.



En la ilustración anterior, se observa que el carrito de la compra muestra los datos del pedido que corresponde con la compra que está realizando el usuario durante la sesión actual, a través un procesamiento maestro-detalle. La tabla maestra corresponde con los datos del pedido actual y el detalle incluye los datos de cada uno de los productos que el usuario está comprando. El enlace "Seguir comprando" redirige a la acción *Index()* del controlador *Escaparate*. Para facilitar el desarrollo, se debe utilizar una variable de sesión, denominada *NumPedido*, que almacenará el valor del número del pedido que se encuentra actualmente en el carrito de la compra. Esta variable de sesión permite acceder al número de pedido que se encuentra en el carrito de la compra, desde cualquier página de la aplicación Web. Si el valor de esta variable de sesión es *null*, entonces el carrito de la compra está vacío. Añadir la opción "Carrito" en el menú de usuario de la aplicación Web.

- En el controlador *Carrito*, implementar las siguientes acciones para resolver las diversas opciones disponibles en el carrito y que pueden observarse sobre la ilustración anterior.
  - Acción *ConfirmarPedido()*. Implementa la funcionalidad para confirmar la compra en firme del pedido actual, que se encuentra en el carrito de la compra. Se recibe como parámetro el valor del Id del pedido a confirmar. Si el carrito está vacío, entonces no puede confirmarse el pedido. Al confirmar el pedido actual, se modifica el valor del estado del pedido al valor "Confirmado", se modifica la fecha en la que el pedido fue confirmado con la fecha actual y, se elimina la variable de sesión *NumPedido* utilizando el método *Remove()*. Al finalizar, se redirige la ejecución hacia ver el escaparate. Si el pedido que se encuentra en el carrito de la compra no se confirma, entonces ese pedido quedará almacenado en el estado "Pendiente".
  - Acción *CarritoVacio()*. Implementa la funcionalidad para mostrar una página con un texto que indica que el carrito está vacío. Esto ocurre cuando el valor de la variable de sesión *NumPedido* es *null* y, por tanto, no existe ningún producto añadido al carrito.
  - Acción *EliminarLinea()*. Implementa la funcionalidad para eliminar una línea de detalle del carrito. Se recibe como parámetro el valor del Id del detalle de la línea que se desea eliminar. Al finalizar, se redirige la ejecución hacia ver el carrito.
  - Acción *MasCantidad()*. Implementa la funcionalidad de modificar el detalle del producto actual para incrementar la cantidad en una unidad de compra. Se recibe como parámetro el valor del Id del Detalle de la línea que se desea modificar para aumentar la cantidad a comprar. Al finalizar, redirige la ejecución hacia ver el carrito.
  - Acción *MenosCantidad()*. Implementa la funcionalidad de modificar el detalle del producto actual para decrementar la cantidad en una unidad, si el valor de la cantidad es mayor que 1. Se recibe como parámetro el valor del Id del Detalle de la línea que se desea modificar. Al finalizar, redirige la ejecución hacia ver el carrito.
  - Establecer adecuadamente el filtro *Autorize* en los controladores *Escaparate* y *Carrito*.

Eliminar las variables de sesión al cerrar la sesión de usuario

- El valor de la variable de sesión *NumPedido* se debe eliminar, o poner a null, al Cerrar la sesión del usuario. Ello evita que pueda existir un valor de pedido actual en el carrito de la compra, cuando acceda un usuario a la aplicación Web. Para ello, se pueden utilizar los métodos *Remove()* o *Clear()* en el método *onPost* de la página *Logout.cshtml.cs*, que está situada en la carpeta */Areas/Identity/Account/Pages* del Proyecto. El uso del método Clear() elimina todas las variables de sesión, lo que es siempre recomendable al cerrar la sesión del usuario.

Desarrollo opcional

Opcionalmente, se propone la construcción de aspectos avanzados de la aplicación Web, que deberán completarse, igualmente, con un alto nivel de acabado.

- Construir el controlador ***MisPedidos***, y las vistas asociadas, para poder gestionar los pedidos correspondientes al usuario actual.
- Implementar la opción "Detalles" en el controlador *MisPedidos* para que los usuarios pertenecientes al rol de Usuario puedan visualizar los datos de cada pedido, así como las líneas de cada pedido a través de un procesamiento de tipo maestro-detalle. En el procesamiento "Detalles" del controlador Mis Pedidos de un usuario, si se desea, se puede también agregar una opción para poder confirmar un pedido que se encuentre en estado "Pendiente".
- Añadir la opción "Mis Pedidos" en el menú de usuario.
- Establecer el filtro *Autorize* sobre el controlador *MisPedidos* de la manera más adecuada.
- Implementar la opción "Detalles" en el controlador *Pedidos* para que los usuarios pertenecientes al rol de Administrador puedan visualizar los datos de cada pedido, así como las líneas de cada pedido a través de un procesamiento de tipo maestro-detalle.
- Se recomienda utilizar Github como herramienta de control de versiones durante el desarrollo de la aplicación Web de tienda online.

Entrega

La entrega del supuesto práctico integrador se realizará en la sesión de clase asignada mediante presentación y demostración pública de la aplicación Web desarrollada antes del **21/02/2025**.

Calificación

Como síntesis del aprovechamiento docente, la superación del supuesto práctico integrador se basa en alcanzar un alto nivel de acabado de la funcionalidad de la aplicación Web planteada. Los criterios de calificación a aplicar se basan en la valoración de los siguientes aspectos:

Criterio de calificación	Valoración
Nivel de acabado y perfección alcanzado de la implementación y presentación visual	35 %
Grado de dificultad de la aplicación Web en relación con los objetivos planteados	35 %
Adecuación y coherencia de la exposición oral en la presentación y demostración	30 %

Las máximas calificaciones se obtendrán si se completan, con un alto nivel de acabado, también las tareas de desarrollo opcional. El supuesto práctico integrador es una actividad de enseñanza y aprendizaje de carácter individual. El trabajo desarrollado deberá ser original por parte del alumno/a que lo realiza. En caso de que no fuera original, la calificación obtenida será la menor posible.