Memoria

Practica MaD 2020/21

Autores:

mad17:

-Jacobo Jorge Hermida (jacobo.jorgeh@udc.es)
-Diego Villanueva Fariña (diego.villanueva@udc.es)
-Beltrán José Aceves Gil (beltran.aceves@udc.es)

1. Arquitectura global

2. Modelo
   - 2.1. Clases Persistentes
   - 2.2. Interfaces de los servicios ofrecidos por el modelo
   - 2.3. Diseño de un DAO
   - 2.4. Diseño de un servicio del modelo
   - 2.5. Otros aspectos

3. Interfaz gráfica

4. Partes opcionales
   - 4.1. Comentario de productos
   - 4.2. Etiquetado de comentarios
   - 4.3. Cacheado de búsquedas

5. Compilación e instalación de la aplicación

6. Problemas conocidos

1. Arquitectura global

La aplicación está dividida en 3 proyectos (Model ,Test y Web), cada uno de estos proyectos tiene una función específica.

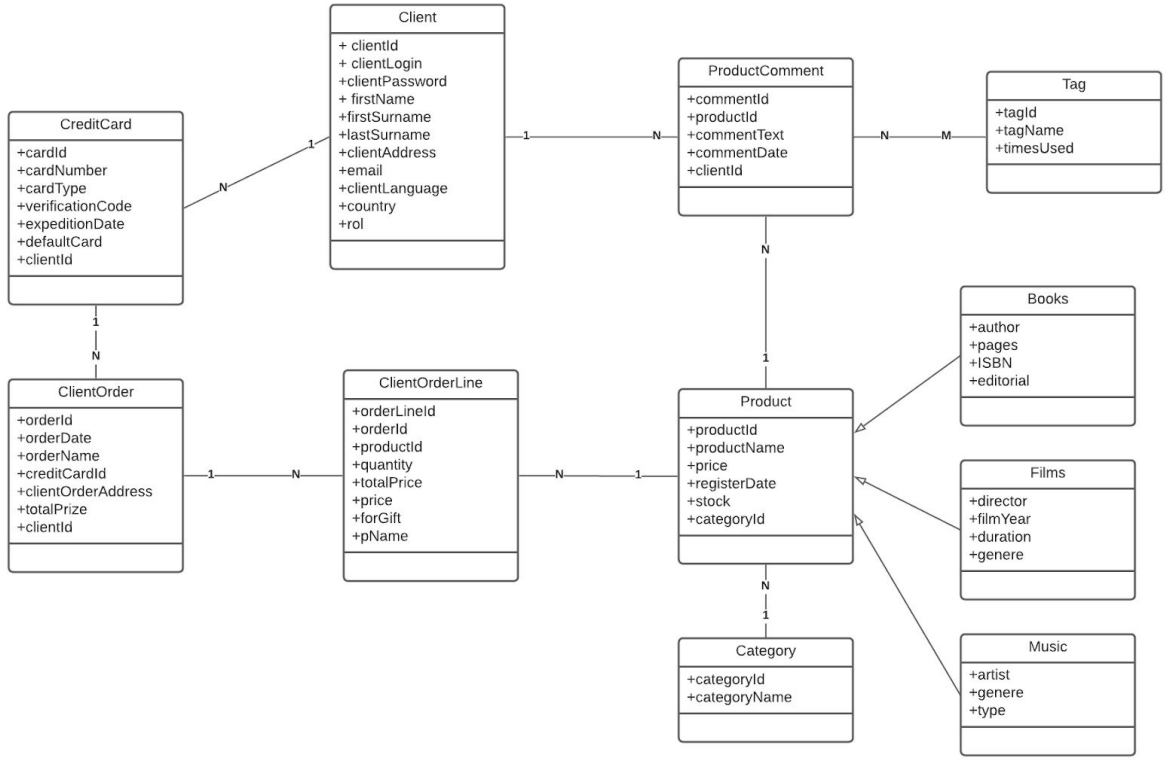
En el caso de **Model** es el proyecto encargado de interactuar con la BD mediante el Entity Framework. el cual crea los Daos (Model.DAOs) y los distintos servicios (Model.Services) a los accederán Web y Test, además tenemos los DTOs (ProductDetails,ClientDetails,etc).

En el proyecto **Test** será donde comprobemos que los distintos métodos implementados en los servicios de **Model** se ejecutan de manera correcta mediante la implementación de distinto tipo de pruebas.

En el proyecto **Web** tenemos las distintas páginas que vamos a mostrar al usuario, para ello tenemos que manejar la interfaz mediante los .aspx y en los aspx.cs introducimos el código asociado a cada pagina, tambien tenemos un IoCManagerNinject en el cual manejamos las inyecciones necesarias de Ninject, además existe un SessionManager en el cual manejamos las distintas Cookies que tenemos en relación a los métodos, por ejemplo, al iniciar sesión guardamos información del cliente en una cookie.

2. Modelo

2.1. Clases Persistentes

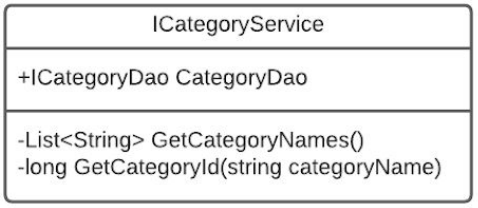


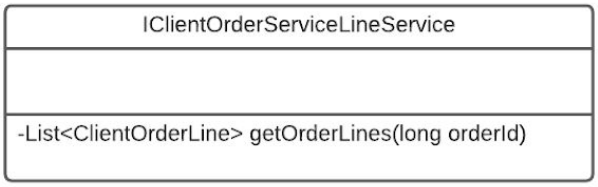
Algo a mencionar en particular de las clases persistentes es que para la gestión de categorías lo hemos implementado de la siguiente manera, tratamos ciertas categorías (En este caso Books, Films y Music) como entidades las cuales heredan de la clase Product en vez de como un simple atributo de la clase Category. Para el resto de categorías utilizamos la clase general de Category.

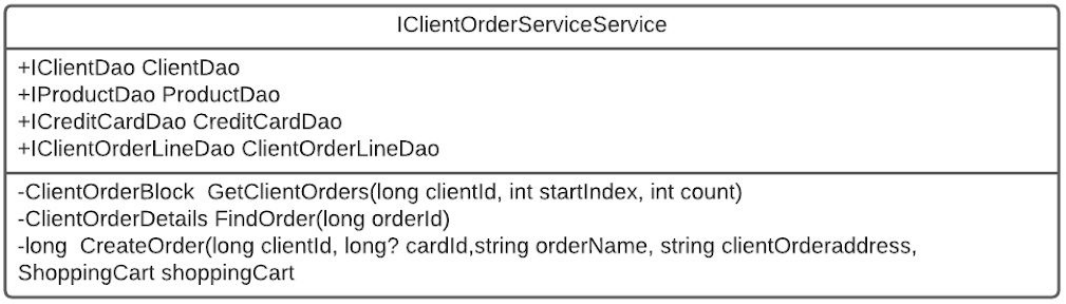
Para la gestión de las Tags en los Comments, tenemos una tabla intermedia en BBDD la cual gestiona esa relación N-M entre Tag y ProductCommentTag a pesar de que no se muestre en el diagrama

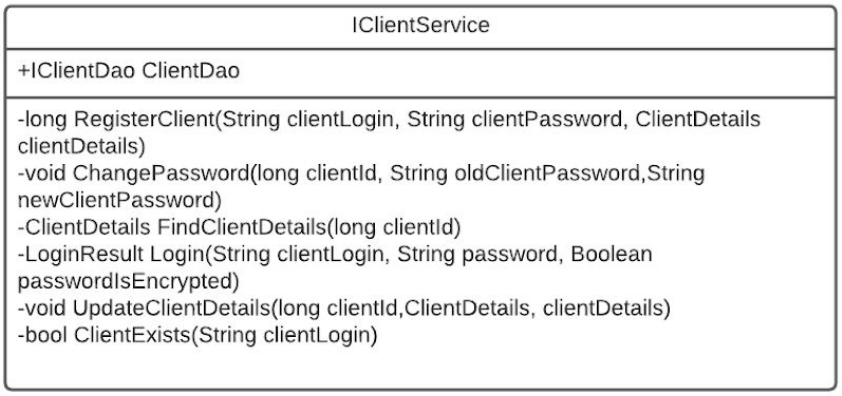
Pese a que en la aplicación se emplea un objeto ShoppingCart éste no está presente en este diagrama puesto que no es una entidad no persistente.

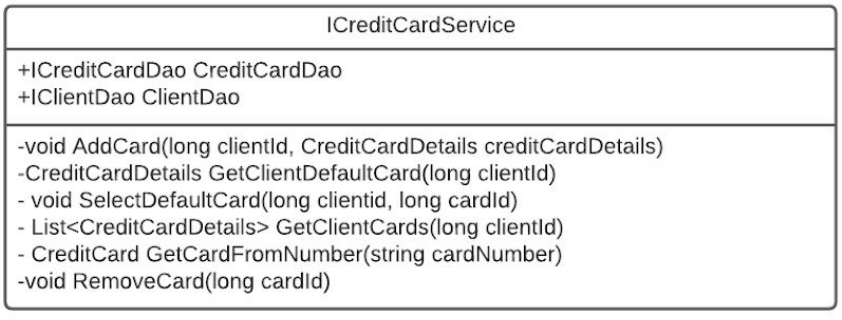
2.2. Interfaces de los servicios ofrecidos por el modelo

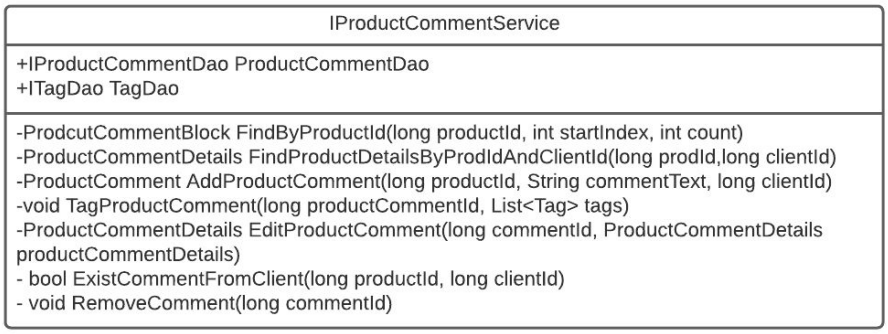


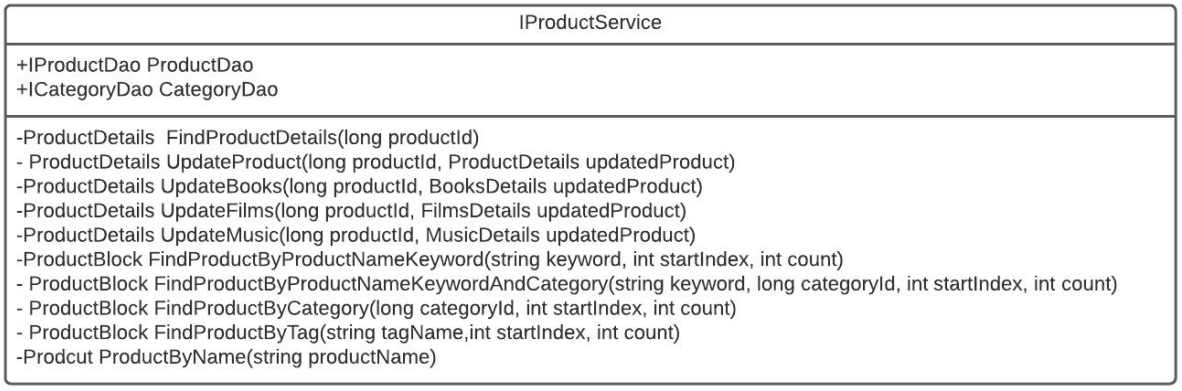


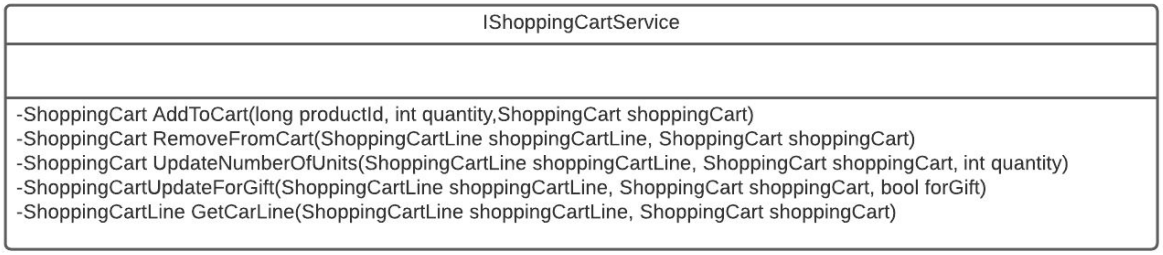


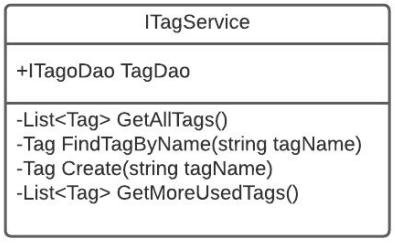






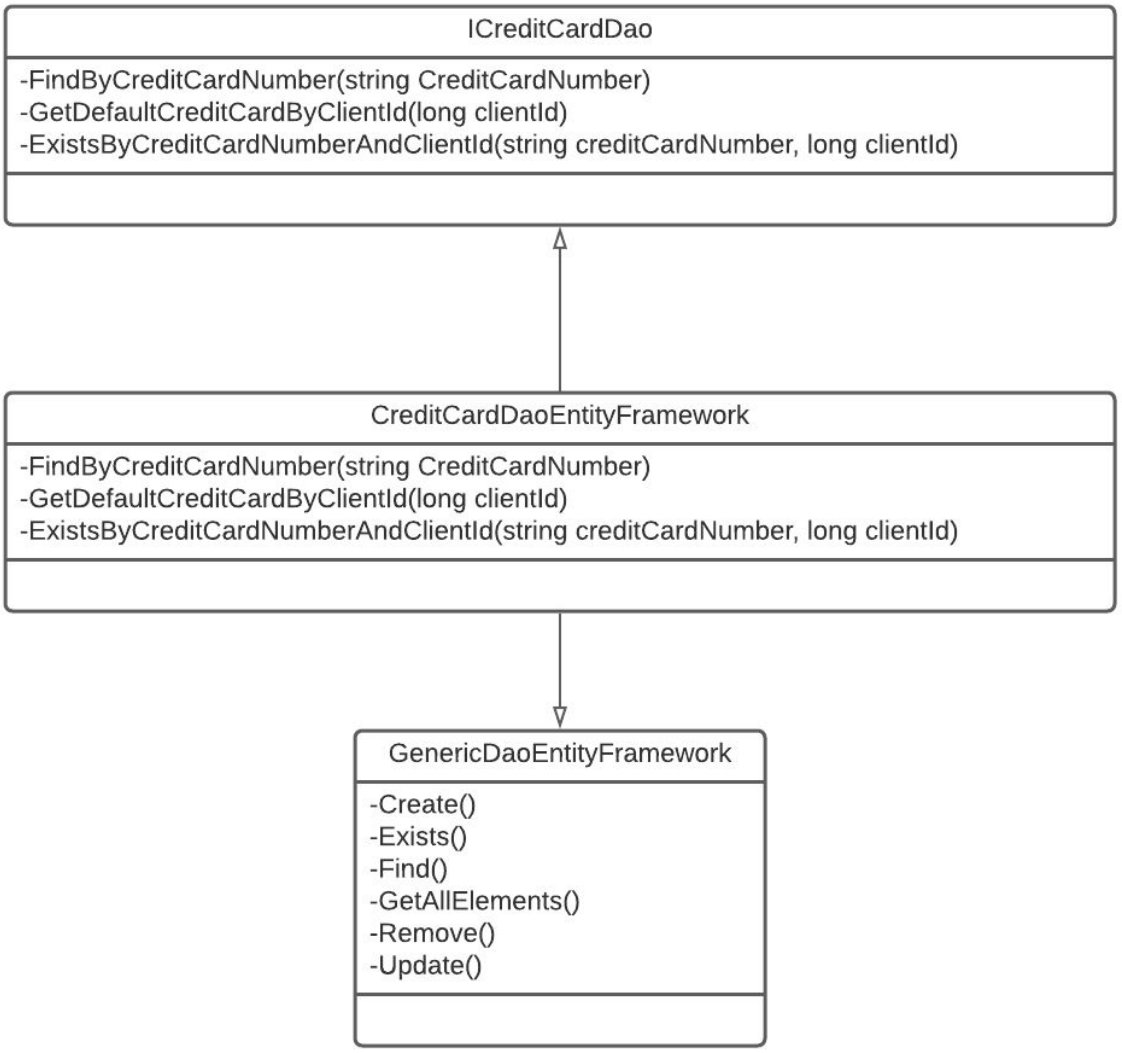






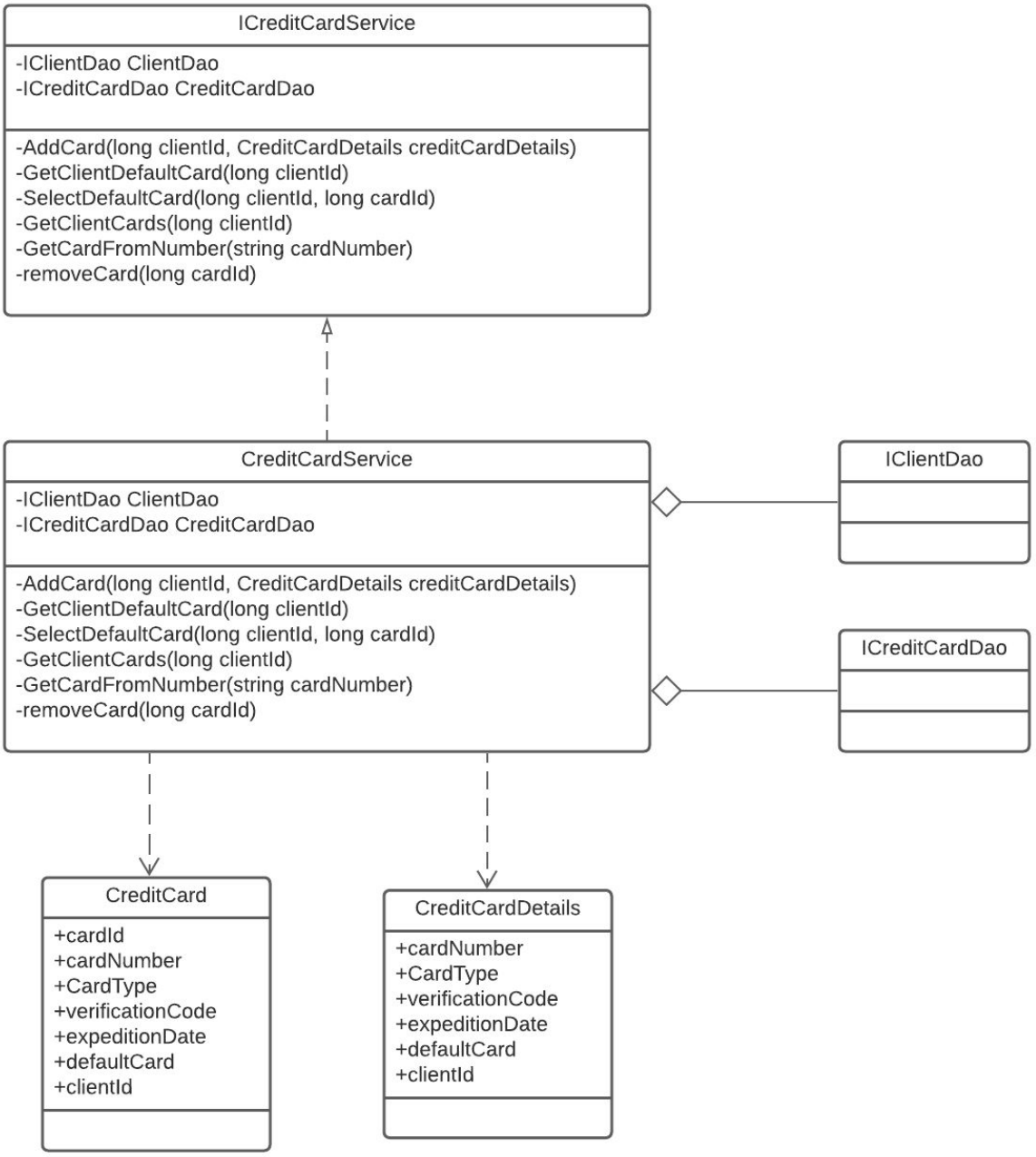
Para cada servicio hemos implementado las operaciones necesarias para cumplir los requisitos de nuestra aplicación web, para ello tenemos una interfaz de servicio para cada clase tanto persistente como no persistente ya que asi tenemos un código que es facil de mantener ya que si agrupamos más los casos de uso en menos servicios podríamos llegar a generar alguna clase Dios que complicaría el mantenimiento del código.

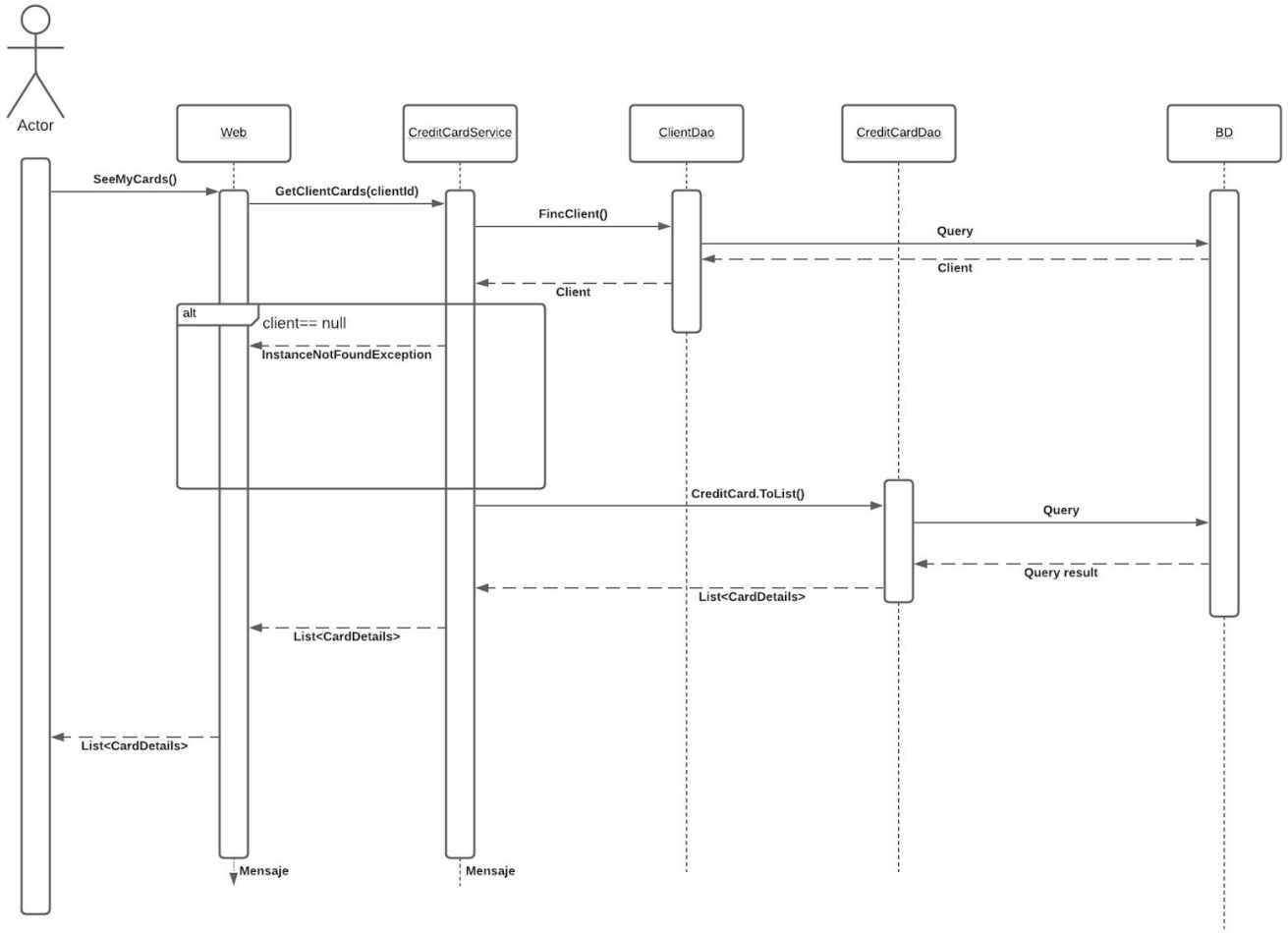
2.3. Diseño de un DAO



Cada DAO cuenta con un interfaz, al cual llamaremos cuando queramos realizar alguna consulta, además cada DAO hereda de GenericDaOEntityFramework de ModelUtil en el cual están declaradas varias operaciones básicas. En la clase CreditCardDaoEntityFramework es donde implementaremos los métodos de la interfaz, en nuestro caso optamos por usar Linq para comunicarnos con el Entity Framework. En este caso no usamos paginación pero en otros DAOs si que se usa, para ello usamos Skip(int startIndex) para saltar los elementos que queremos y Take(int count) para obtener *count* elements.

2.4. Diseño de un servicio del modelo





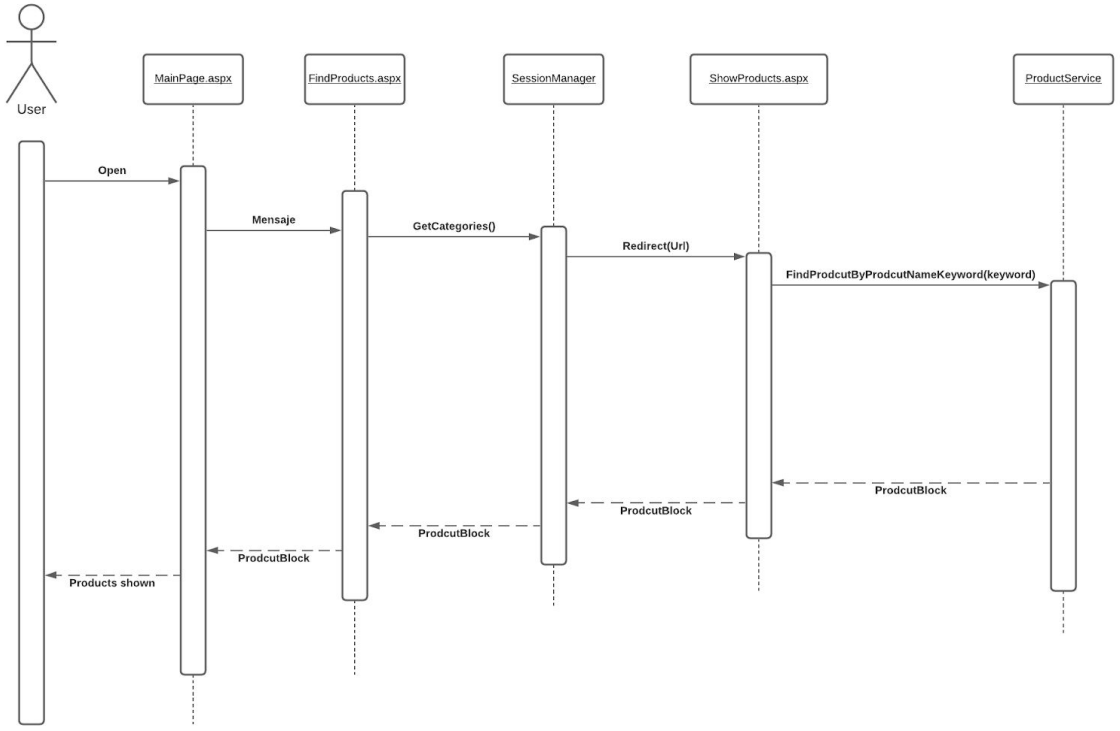
En este ejemplo tenemos un método en el cual podemos tener una excepción de tipo InstanceNotFoundException cuado el clientId que se proporciona no pertenece a ningún cliente, tras comprobar que el cliente existe se procederá a obtener todas las tarjetas de crédito del usuario pidiendo al DAO que ejecute esta búsqueda en la , en caso de ser 0 se devolverá una lista vacía.

2.5. Otros aspectos

Con respecto al tema de las excepciones, nosotros empleamos las siguientes:

- **IncorrectPasswordException**: Se lanza cuando un usuario introduce una contraseña incorrecta, tanto a la hora de hacer login como a la hora de cambiar su contraseña
- **NotEnoughStockException**: Se lanza cuando se quieren añadir al carrito más unidades de las que hay disponibles de un producto

3. Interfaz gráfica



4. Partes opcionales

4.1. Comentario de productos

En esta parte, añadimos una tabla ProductComment en la que guardamos la información necesaria de los comentarios teniendo una relación 1-N con Product, un usuario podrá añadir un solo comentario a cada producto pudiendo editarlo siempre que quiera, para ello habilitamos un botón que en caso de no tener ningun comentario hecho sobre el producto te permite crearlo y en caso contrario editarlo.

En la ventana de editar el comentario se podrán cambiar los distintos campos de un comentario(Texto y Etiquetas), además en esta ventana se le ofrecerá al usuario la opción de eliminar este comentario, cuando un usuario entre en la ventana de información detallada de un producto mostraremos una lista de comentarios en los que mostraremos quien es el autor, el texto del comentario y la ultima fecha de edición.

4.2. Etiquetado de comentarios

Para etiquetar los comentarios creamos una tabla TAG que tiene relación N-M con ProductComment, para crear estas etiquetas cuando un usuario crea o edita un comentario será posible tanto crear una nueva etiqueta como seleccionar entre las ya existentes.

Las 5 etiquetas más usadas serán mostradas en todas las páginas, además estas 5 etiquetas que mostramos en la parte inferior de la pantalla si clickeamos en ellas realizaremos una búsqueda de todos los productos en los que algún comentario tenga asociada esa etiqueta. Además el tamaño de estas etiquetas que mostramos varía según las veces que están siendo usadas, con unos tamaños mínimos y máximos predeterminados para que visualmente no sean ni muy pequeñas ni muy grandes siendo el máximo 50px y el mínimo 10px.

4.3. Cacheado de búsquedas

Cuando se busca por palabra, por categoría o ambas se comprueba si tenemos esa búsqueda en caché, en caso de tenerla recuperamos el ítem asociado a esa búsqueda. Para estas comprobaciones tenemos una lista del objeto Search en el que guardamos el tipo de búsqueda(categoría,keyword,startIndex,count).

Cómo guardamos solo 5 búsquedas en el caso de no encontrar la búsqueda y la lista de búsquedas ya tenga 5, eliminamos el primer elemento ya que es la búsqueda que hace más tiempo que no se realiza, en el caso de que la búsqueda si que estuviese en la lista eliminamos esa entrada y la añadiremos al final ya que es la última búsqueda realizada.

En caso de que la búsqueda esté en caché recuperaremos el objeto, en caso de que no estuviese realizamos la búsqueda sobre la base de datos.

5. Compilación e instalación de la aplicación

Teniendo el proyecto importado dentro de VisualStudio, para compilarlo, bastaría con hacer click derecho sobre el proyecto y clickear en compilar

Para la instalación de la aplicación bastará con irse al proyecto Web, entrar en la carpeta Pages, y hacer click derecho sobre el archivo MainPage.aspx y después seleccionar “Ver en el explorador”. Haciendo eso se nos abrirá el navegador en la página principal de nuestra aplicación.

6. Problemas conocidos

No existe ningún error conocido.