Instituto Tecnológico Colonia
CTC

Obligatorio 1: Aplicación para Distribuidora de vinos

Entregado como requisito para la obtención del título de Analista Programador

Leandro Chelentano
Nicolas Gutierrez

Tutor: Alen López

Año: 2021

Declaración de auditoría.

Nosotros, Leandro Chelentano y Nicolas Gutierrez, declaramos que el trabajo que se presenta en esta obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras cursábamos la materia Programación 1;
- Hemos tenido en cuenta las clases dictadas por el Prof. Alén López quién además nos proporcionó material;
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

_____	_____
Leandro Chelentano	Nicolas Gutierrez
06/06/2021	06/06/2021

Lectura.

Planteamiento.

Se planteó la creación de un software codificado en HTML, CSS y JavaScript destinado a una hipotética distribuidora de vinos, la cual contaría con varias bodegas y cepas a su vez.

Debido a lo ya mencionado el software requiere de la implementación de un sistema llamado ABM (por sus siglas; Altas, Bajas y Modificaciones.) el cual sería destinado al apartado de vinos, requiriendo que se contengan propiedades como un número de identificación, el tipo de cepa, la bodega, el stock y el precio.

A su vez se pedía un sistema de estadísticas, el cual sea capaz de al consultarse ofrecer datos tales como; El nombre de la bodega que contiene el vino más barato, cuál es la cepa más costosa y una breve consulta de stock a partir de una bodega y una cepa.

Como respuesta a este planteamiento el equipo decide implementar una serie de funciones que sobrepasan los requerimientos iniciales, pero que hacen que el software contenga algunas propiedades más avanzadas y útiles, como si realmente fuese a ser empleado.

En lo referente al apartado ABM.

Se opta por suplir todos los requerimientos que el programa base plantea, siendo estos la implementación de un ABM para los vinos, con algunas características como la reutilización de un número identificador eliminado, o que ante un ingreso duplicado, simplemente se produzca un aumento en el stock pertinente.

En lo referente a las cepas y las bodegas no se deja en claro su naturaleza, por lo que el equipo implementa otros dos sistemas AMB diferentes para estas propiedades, cabe destacar que todas las bases de datos, por más que sean por naturaleza volátiles, están relacionadas entre ellas, muy necesario para el funcionamiento del programa en sí, además de las estadísticas en tiempo real de las que se dispone.

Se dispone también de un apartado específico para el control de stock, para no ser de esta manera necesaria la edición de todas las características del vino y permitiendo una línea de trabajo más simple y eficiente.

Concerniente a posibles “bugs”.

Se implementan diferentes sistemas para evitar que se produzcan errores a la hora de que un posible usuario haga un mal uso del programa, previniendo múltiples desperfectos.

Sobre el producto final.

El presente obligatorio contiene diversos materiales, tales como; pseudocódigo, explicación, código fuente, datos de testing y manual de uso del programa (los cuales pasan a manos del instituto CTC al recibirse junto al presente documento). Siendo creado bajo la premisa de ser empleado por una distribuidora de vinos.

El software permite el ingreso, en cuanto al vino, de cuatro diferentes valores, siendo estos el nombre, el precio, la cepa, y el stock. El dato del número identificador se otorga automáticamente en base a los números ya otorgados, disponiéndose asimismo de una función desde la cual puede elegirse un identificador previamente eliminado, el cual puede ser reutilizado, en lugar del automaticamente asignado. Referente al dato del nombre, es un dato único, que prácticamente podría ser utilizado como identificador, ya que si este nombre es el mismo a uno definido con anterioridad, no se realizaría el ingreso de forma normal, sino que se notaría una inflación en el stock del vino ya existente, el cual contendría su mismo nombre.

Por otro lado, el dato referente a la cepa es escogido en base a la base de datos de las cepas, el cual debe ser ingresado antes de comenzar con el ingreso del vino, debido a que ninguno de los campos puede no contener caracteres.

En cuanto a las cepas, el único dato que se ingresa es el nombre, ya que el identificador se asigna automáticamente, además de esto, aunque oculto para el usuario, se dispone de una tercera propiedad denominada “enUso”, la cual contiene un valor numérico que crece o decrece cada vez que la cepa se da un nuevo uso o se deja sin él, respectivamente.

Para con las bodegas; se permite el ingreso de tres datos, el nombre, teléfono y localidad, funcionando de manera similar a las cepas, pero sin la propiedad mencionada de “enUso”.

El programa también dispone de una característica capaz de relacionar y desrelacionar los vinos con una o más bodegas, esta funcionalidad es posible gracias a que dentro de cada vino o bodega hay una lista de vinos contenidos para las bodegas y una lista de bodegas para los vinos, esta operatoria es sumamente útil para, entre otras cosas, el correcto funcionamiento de las estadísticas.

Indice

1. Pseudocodigo de las estadísticas.

1.1: Bodega con el vino más barato

```
Inicio
Leer VinoLowCostPrice = numero.maximo_Valor
Leer VinoLowCostBodega = ‘’

Para cada  i = 0 hasta vinos.largo hacer
        Si vinos[i].precio < VinoLowCostPrice hacer
                Si vinos[i].bodegas.largo > 0 hacer
                        VinoLowCostPrice = vinos[i].precio
                        VinoLowCostBodega = vinos[i].bodegas[0]
                Fin si
        Fin si
Fin para
Si VinoLowCostBodega es Indefinido hacer
        VinoLowCostBodega = ‘’
        Hacer valor elementoporId ‘wineLowCost’ = VinoLowCostBodega
Sino
        Hacer valor elementoporId ‘wineLowCost’ = VinoLowCostBodega
Fin Si
Fin
```

1.2: Cepa mas cara

```
Inicio
Leer priceWineMoreExpensive = 0
Leer cepaMoreExpensive

Para cada  i=0 hasta vinos.largo hacer
        Si vinos[i].precio menor que priceWineMoreExpensive
        Hacer priceWineMoreExpensive = vinos[i].precio
        Hacer cepaMoreExpensive = vinos[i].cepa
Fin Para


Si cepaMoreExpensive es Indefinido
        Hacer cepaMoreExpensive = “”
Sino
        Hacer valor elementoporId ‘whineHighCost’ = cepaMoreExpensive
Fin Si
Fin
```

1.3: Consulta stock

Variables empleadas: Nombre de la bodega seleccionada, Nombre de la cepa seleccionada, base de datos de vinos, base de datos de cepas, base de datos de bodegas.

INICIO

Variables: winne, storee (definidas por el usuario), cepaa, existenceOnStore = false
Leer variables

var bodegaId;
Dar tantas vueltas como bodegas hay
    Si nombre de bodega[vueltas] == storee
        bodegaId = vueltas;
    Fin del condicional
Fin del bucle

var vinosAVerificar = bodegas[bodegaSeleccionada].vinosContenidos;
var stockk = [];
Limpio array stockk

Dar tantas vueltas como vinos hay (i)
    Dar tantas vueltas como vinosAVerificar hay (o)
        Si nombre de vinos[i] == vinosAVerificar[o]
            Si cepa de vinos[i] == cepaa
                Se agrega array stockk el stock de vinos[i];
            Fin del condicional
        Fin del condicional
    Fin del bucle
Fin del bucle

Var totalStock;
Dar tantas vueltas como indices en stockk hay (i)
    totalStock = totalStock + stockk[i];
Fin del bucle

Mostrar totalStock;

FIN

2. Funciones utilizadas.

2.1. Funcionamiento general.

El funcionamiento general del programa incluye recibir una cierta cantidad de datos, procesarlos, relacionarlos y posteriormente almacenarlos, para mostrarlos cuando sea requerido así como su modificacion o eliminacion.

Si hay algún dato que no es de correcta naturaleza, se le hace saber del error al usuario mediante ventanas emergentes.

2.2. Función para el ingreso de datos.

Esta función es similar para todos los AMB del proyecto, consta de captar los datos que se muestran en los distintos formularios, y estos son almacenados en arrays asociativos. Determinados campos de texto solo acepte números, mientras que es indiferente para otros, y otros sólo aceptan valores mostrados en selects.

2.3. Función para mostrar datos ingresados.

Esta función es llamada cada vez que hay alguna clase de cambio en la base de datos, de cualquier naturaleza, como el ingreso de datos, la forma de mostrarlos es muy similar para ambos tres ABM, con la diferencia de que los vinos y las bodegas contienen una función extra para espaciar los datos en una suerte de columnas, que aunque imperfectas facilitan el ser leídas.

2.4. Función para eliminar datos.

Esta función es empleada de forma similar en todos los ABM, ocurre que al estar la posibilidad de que el elemento se encuentre relacionado con otros podría ocasionar un “bug”, por ejemplo, la función de eliminar una bodega que contiene un vino se encuentra bloqueada, pues desembocará en un malfuncionamiento del programa, ya que el vino estaría enlazado en una bodega inexistente.

2.5. Función para modificar datos.

Se trata de una función importante dentro del programa, pero que podría ocasionar graves daños si en algún caso se modifica un valor importante, esto se ve, por ejemplo, en que debido a que los vinos y bodegas se encuentran relacionados entre sí por el nombre, si se modificara este, terminariamos con una situacion igual que si se hubiese eliminado.

Este funcionamiento lo comparten todos los AMB.

2.6. Función para relacionar vinos (relateWineStock()).

Esta función es empleada para relacionar un vino con una bodega y viceversa, básicamente se coloca el nombre del vino dentro del array de vinos contenidos, en el array de bodegas, y el nombre de la bodega en el array bodegas del array vinos.

2.7. Función para desrelacionar vinos (desrelateWineStock()).

Esta función es la contraparte del punto 2.6, elimina el nombre del vino y la bodega del array pertinente dentro de cada array asociativo.

2.8. Función de estadísticas (StoreWithMoreLowCostWine()).

Esta función recorre la lista de vinos buscando el menor precio ingresado, lo guarda en variable y utiliza esa misma variable para buscar la bodega asociada, si tiene alguna, y mostrarla en el cuadro de texto.

2.9. Función de estadísticas (moreExpensiveCepa()).

Esta función es similar a la del punto 2.8 pero en vez de buscar el menor precio, se busca el mayor ingresado en la lista de vinos, de allí mismo se busca la cepa asociada al vino que hemos encontrado como “el más barato” y se imprime en el cuadro de texto.

2.10. Función de estadísticas (stockSearch()).

Esta función es llamada cada vez que se selecciona un cepa en el select de la barra lateral en la sección de estadísticas. Se encarga de sacar los vinos que la bodega seleccionada contiene, y los recorre en un dulce para ver si tiene la cepa buscada, y si es así agrega el valor a una variable contador, que es la que finalmente se muestra la suma de todos el stock en el lugar correspondiente.

3. Testing

3.1. Ejecución de funciones con variables en blanco.

Al intentar agregar o modificar en cualquiera de los AMB un índice con un dato en blanco en algunas de las variables, es detectado por el programa, el cual cancela la operación y da un mensaje de alerta, como se aprecia en la figura 1.

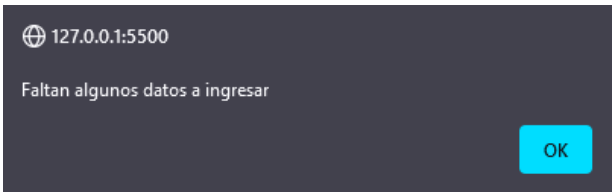

Figura 1.

3.2. Ingreso de variables numéricas como letras.

Cuando el programa espera el ingreso de un número, pero en cambio recibe un valor con el que no se puede operar, lanza un mensaje de error, tal como sucede con el precio y el stock de un nuevo vino, ver en figura 2.

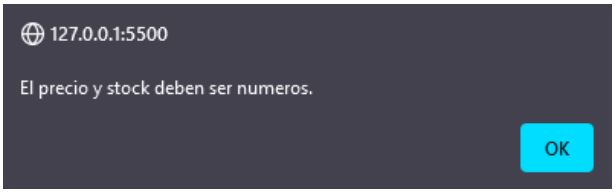

Figura 2.

3.3. Modificar un elemento relacionado a otro.

Al momento de editar, por ejemplo, una bodega que contiene vinos, el programa lanza una alerta, cancelando la operación, previniendo que los datos dentro del los array resulten corruptos, al posiblemente perder la relación entre ellos, puede observarse a continuación en la figura 3. Al estar un vino relacionado el programa dará una alerta del problema, y dirá que debes des-relacionar el vino a la bodega antes de editarlo.

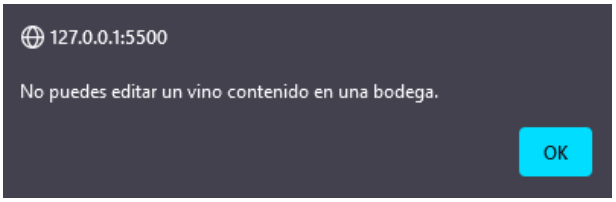

Figura 3.

3.4. Al relacionar un vino y una bodega.

Al relacionar un vino y una bodega estamos guardando datos en ambas listas, tanto en bodegas como en vinos. En la bodega se guardará el nombre del vino, y en la lista de vinos se guardará el nombre de la bodega, para así cuando se seleccione un vino en el Select del workspace, todas las bodegas que están asociadas al mismo vino aparecerán en el select Bodegas. Al estar asociados entre si, ninguno de estos podrá modificarse ya que se relacionan con su mismo nombre. Al cumplirse esta relación, el programa alertara de que ha sido completada con éxito como en la siguiente imagen.

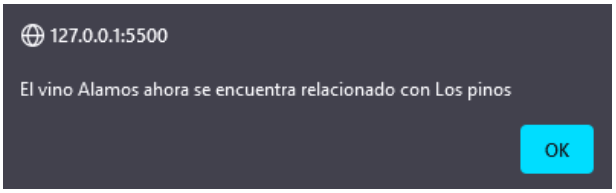

Figura 4.

3.5. Al des-relacionar un vino y una bodega.

Al des-relacionar un vino y una bodega estamos dando permisos al programa de poder modificar cualquier dato de cada uno de ellos, ya sea en bodegas (Nombre, teléfono, localidad), o en Vinos(Nombre, cepa, stock, etc). Después de realizar cualquier cambio pueden volver a relacionarse cualquiera de estos que se hayan seleccionado. El programa dará una alerta cuando esto se cumpla como se muestra a continuación.

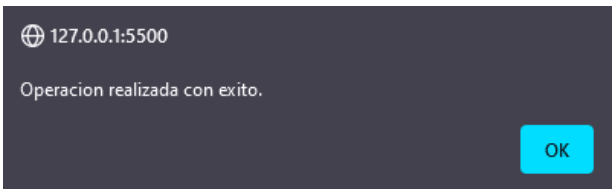

Figura 5.

3.6. Al relacionar un vino y una bodega ya relacionados.

Al intentar relacionar un vino con una bodega que ya estan relacionados entre sí, el programa compara una lista con otra y si ya estan asociadas este mismo lanza una alerta e impide que esta función vuelva a cumplirse, evitando errores en el programa, como se muestra en la siguiente imagen:

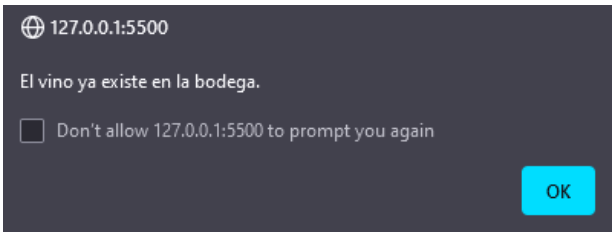

Figura 6.

4. MANUAL DE USUARIO

4.1. Funcionamiento de los sectores ingreso, baja y modificación.

En todas las secciones se encuentra las herramientas de altas bajas y modificaciones, las cual permite hacer cambios a las diferentes bases de datos, el funcionamiento es muy simple ya que cada zona tiene sus propios botones y sus propios inputs de texto.
Realizar un mal funcionamiento de esta zona es casi imposible debido a que hay alerts para prevenir cualquier error.

4.2. Funcionamiento del Sector de relación.

El sector de relación sirve para ligar vinos y bodegas así como también hacer la operación contraria. Cada operación puede realizarse una sola vez estando cada vino relacionado una sola vez a una bodega en específico, aunque un vino puede estar relacionado a más de una bodega.

4.3. Funcionamiento del sector de ventas.

El sector de ventas es una forma eficiente de controlar el stock de un vino pudiendo agregar o vender un vino sin necesidad de editar todo el elemento.

4.4. Funcionamiento del sector de estadísticas.

El funcionamiento del sector de estadísticas es muy simple, debido a que la mayor parte del trabajo se hace de manera automática, no obstante la parte del control de stock solicita el input del texto del usuario, por lo tanto es el único momento en el que el usuario debe realizar una operación para mostrar una estadística.

4.5. Funcionamiento del sector de vinos de Ids. eliminadas.

Si el usuario desea emplear nuevamente un id previamente eliminado puede hacerlo mediante el botón “Usar id. Eliminada”. Si la operación que se hace posteriormente a elegir o seleccionar una id. Eliminada no es dar de alta la operación se cancelará y se volverá al sistema de id.s automaticas.

4.6. Advertencias Generales.

Si el usuario pretende realizar cambios o modificaciones no admitidas por el programa, el mismo cancelará la operación para impedir que se realicen daños al software.

Se recomienda una vez que el ítem está ingresado, no realizar modificaciones ya que estoy podrías contar la base de datos. Si esta operación es completamente necesaria deberá desvincularse ese ítem del resto de la base de datos y luego volver a vincularse para no realizar ningún daño.