|  |
| --- |
| **CONSIDERACIONES DEL EXAMEN:**   * La duración de este examen es de 3:50 * El alumno se asegurará de subir todos los archivos necesarios a la tarea correspondiente del aula virtual * **Cada ejercicio tiene una puntuación máxima**, que se conseguirá si cumple todas las funcionalidades pedidas. Por cada fallo o funcionalidad no realizada se restará a la puntuación máxima la parte correspondiente. * **Se valorará con 0,5 puntos que el código sea legible** (nombres de variables lógicas, utilización de funciones, no repetir código, organización de los archivos, MVC,..). |

Se desea desarrollar una aplicación windows para que una academia de baile pueda gestionar las clases, los clientes y la facturación.

Los datos que debe gestionar la aplicación son:

* **Descuentos**: 0,10,15,20,25,30,50,60
* El resto de datos esta en una base de datos MYSQL (hay un contenedor lanzado con mysql y se puede administrar desde [http://localhost:8080](http://localhost:8080/) con usuario root y clave root). **El puerto de mysql es 3606**

Las tablas ya están creadas y son las siguientes:

* + **Clientes**: contiene los datos de los clientes, con clave idCliente y campos, nombreCompleto, email y saldo
  + **Cursos**: contiene los datos de las clases, con clave idCurso y campos nombreCurso, nombreProfesor, tipoBaile, costeMinuto
  + **Inscripciones**: contiene los datos de las inscripciones de los clientes a las clases: idCurso, idCliente, fechaAlta, minutosCurso, estado, descuentoPorMinuto

Para gestionar la aplicación se deben realizar las siguientes tareas:

**TIENES UN ARCHIVO COMPRIMIDO CON PARTE DE LA SOLUCION YA CREADA EN EL ESCRITORIO. INCLUYE EL PROYECTO VISUAL STUDIO CON LOS FORMULARIOS YA CREADOS, EL ARCHIVO docker-compose.yaml. PUEDES COLOCAR LOS ARCHIVOS COMO QUIERAS PERO CUANDO EL PROFESOR VERIFIQUE LA APLICACIÓN NO VA A CAMBIAR NINGUN ARCHIVO**

1(**1,5 ptos**) Formulario **Clientes**

* Se debe mostrar al pulsar en la opción de menu Clientes: No puede abrirse varias veces pero si otros formularios.
* En el datagrid se deben mostrar todos los clientes existentes. SE VALORARA EL USO DE UN CONNECTION STRING UNICO PARA TODA LA APLICACIÓN.
* En el datagrid se debe permitir añadir y editar (no eliminar)
* Si se pulsa el botón “**Actualizar**” hay que guardar todos los cambios en la base de datos
* Si se pulsa el botón “**Nueva Inscripción**” y hay un cliente seleccionado se debe ir al formulario NuevaInscripcion

2(**1,5 ptos**) Formulario **NuevaInscripción**

* Se debe mostrar al pulsar en la opción de menu Inscripción y no puede pulsarse otra opción de menu mientras no se cierre este formulario
* Se debe mostrar en el desplegable **cmbClientes** el nombre de todos los clientes (NO ES VALIDO ESCRIBIRLOS A MANO). Si se ha llegado desde el formulario Clientes debe aparecer el cliente seleccionado en el mismo
* En la fecha **dtpFecha** no debe permitirse seleccionar una fecha anterior a la fecha actual del sistema.
* En el campo **nupMinutosCurso** sólo debe permitirse un valor entero entre 10 y 100.
* En el desplegable **lstCursos** deben aparecer el nombre de todos los cursos disponibles (NO ES VALIDO ESCRIBIRLOS A MANO)
* En el campo **cmbDescuento** deben aparecer todos los posibles descuentos (NO ES VALIDO ESCRIBIRLOS A MANO)
* Si se pulsa el botón “**Cerrar**” se debe cerrar el formulario
* Si se pulsa el botón “**Guardar**” y los datos son validos, se debe guardar la inscripción (el curso, el cliente, la fecha alta, minutos\_curso, el descuento y el estado debe ser “Abierta”) y mostrar un mensaje si se ha guardado bien o no.

3(**5 ptos**) Formulario **Facturacion**

* Se debe mostrar al pulsar en la opción de menu Facturación
* En el elemento **cmbCliente** deben aparecer todos los clientes
* Al seleccionar un cliente se deben mostrar todos los cursos en los que está inscrito (recuerda que el estado debe ser ABIERTA) en el datagrid que hay en el formulario.
* Si se pulsa el botón “**Ordenar por profesor**” debe mostrar las inscripciones ordenadas por nombre del profesor que imparte de la clase.
* Si se pulsa el botón “**Mostrar SALSA**” debe mostrar sólo las inscripciones de SALSA (sino hay ninguna debe indicarse).
* Si se pulsa el botón “**Exporta**r” se deben guardar las inscripciones mostradas en un archivo json. Debe solicitar al usuario mediante una ventana de dialogo el nombre y el directorio donde guardar el archivo.
* Crea una función de nombre **CalcularCoste** que reciba un descuento, unos minutos y un coste por minuto y devuelva el coste a cobrar. Ejemplos:

CalcularCoste(0,10, 10) es 100 porque 10 minutos\*10 euros por minuto y 0 el % de descuento

CalcularCoste(15,10,10) es 85 por 10 minutos \* 10 euros por minuto y 15 es el % de descuento

* Si se pulsa el botón “**Facturar**” y hay una inscripción seleccionada y el cliente tiene saldo se debe actualizar en la BBDD el saldo del cliente, restándole el coste de esa clase (utiliza la función CalcularCoste).
* Modifica el formulario y reemplaza la etiqueta **lblClientes** y el combo **cmbClientes** por un control personalizado que englobe los dos.

4(**1,5 ptos**) Realiza las siguientes tareas:

* Añade en el formulario **NuevaInscripcion** la funcionalidad para que al pulsar F1 en el desplegable **lstClases** se muestre la ayuda del archivo “**ayuda.chm**” que tienes en el directorio de la solución.
* Realiza los cambios para que todos los comentarios del código se guarden en un archivo xml (pon alguno para probar).
* Añade un proyecto de prueba unitaria y realiza tres pruebas con la función **CalcularCoste**. En una prueba comprueba valores correctos, en otra prueba un valor erróneo y en otra prueba valores muy grandes.
* Genera un instalador de la aplicación con las siguientes características:
  + Versión 2.0.0.1
  + Requisitos previos NET Framework 3.5 SP1 (se deben descargar del sitio web del proveedor de los componentes).
  + La aplicación debe buscar actualizaciones después de que se inicie la aplicación.
  + En el nombre del editor pon tu nombre e idioma español
  + El instalador se debe crear en el mismo directorio donde está la solución