ENUNCIADO

Crear una interfaz como la que se adjunta en el fichero .EXE. Sigue los pasos que se describen:

1. Descarga el .EXE y ejecútalo. Fíjate en las pantallas, los menús, botones y mensajes que se muestran para implementarlo del mismo modo todo. Es decir, **la interfaz a desarrollar debe ser una copia de la entregada. El código de acceso es "1"**
2. Crea un **workspace nuevo** con el nombre "TALADRO"
3. Descarga el .ZIP. Es un proyecto MAVEN. Modifica el artifactId del fichero pom.xml para que sea "ExamenJavaFX-01-userName", donde userName es tu usuario CEU. Por ejemplo "**ExamenJavaFX-01- amorillo4532**".
4. Importa el proyecto en Eclipse. En este proyecto verás unas clases que podrás usar (**Estas clases no se modifican):**
    - ExamenService → Servicio para todo lo que tengas que consultar en BBDD. Devuelve datos ficticios. Lee los comentarios para saber cómo funciona
    - Modelo
    - AppController → Clase que "te puede venir bien"
5. Implementa sobre ese proyecto toda la capa de interfaz gráfica de usuario **en el paquete gui**. Empieza por lo principal (navegación y pantallas). Y luego ve ajustando los detalles.

DESCRIPCIÓN DE LA APLICACIÓN

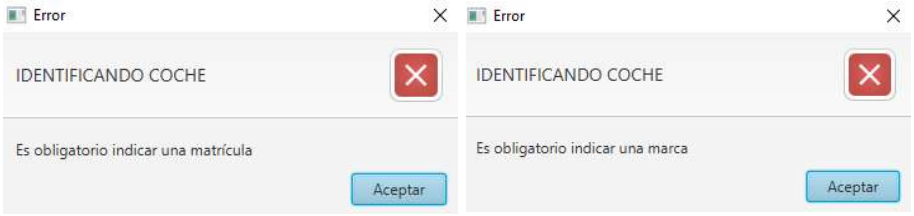
Puedes ver la aplicación en el .EXE. De todos modos, en las siguientes páginas de este documento, se hace una descripción de los principales detalles de cada pantalla. Ten en cuenta que la aplicación debe ejecutarse en una **ventana centrada y no redimensionable**. Esta aplicación sirve para gestionar las reparaciones de los coches de un taller

RÚBRICA

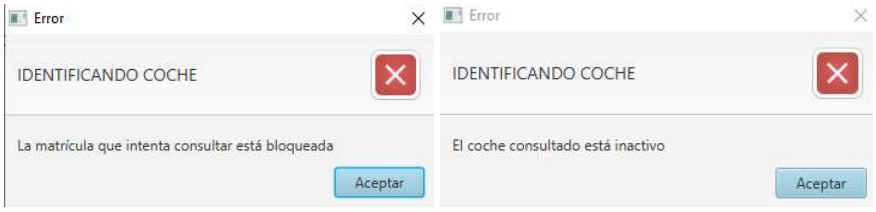
General	Diseño de pantallas y estilos	1
	Navegación, menú, opción salir	1
	Mostrar información de login en pantallas	1,5
Acceso	Control de campos vacíos	1
	Llamada al servicio en hilo paralelo y popup error	1,75
	Carga de datos en combobox	1
Consulta	Control de checkbox y campo filtro inactivo	1
	Carga de datos en la tabla y llamada al servicio	1,75
PENALIZACIONES: Nomenclatura incorrecta de métodos, clases, variables, etc. Clase sin formatear, Incorrecto uso de camelCase, Nombres sin sentido en clases, variables, métodos, etc. Comentarios autogenerados sin borrar. Código sobrante. SerialversionID faltante. Cualquier otro tipo de característica que afecte a la limpieza del código		-3

PANTALLA DE IDENTIFICACIÓN DE COCHE

- La pantalla permite identifica buscando en BBDD el coche con el que vamos a trabajar.
- El campo “matrícula” y el combobox “marca” son **obligatorios**. Si no se selecciona algo, debe aparecer un **popup** de error indicándolo:

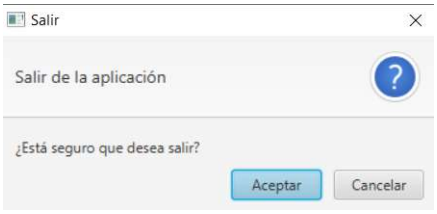


- El **combo de marcas** debe cargarse con las marcas que tengamos en BBDD (Usa el servicio proporcionado método **consultarMarcas**()).
- Para **identificar el coche**, debes llamar al método **identificarCoche**() del servicio proporcionado:
  - Este método debe llamarse en un **Task** independiente ya que su ejecución es lenta.
  - El método devuelve un objeto Coche si no hay errores. Si hay errores, se lanza la excepción **NotFoundException**
  - La única matrícula con la que se puede entrar es la **número 1**
  - Si todo va bien, debes pasar a la siguiente pantalla
  - Si lanza excepción, debes mostrarla el mensaje de dicha excepción en un **popup de error** y no cambiar de pantalla:



PANTALLA DE INICIO

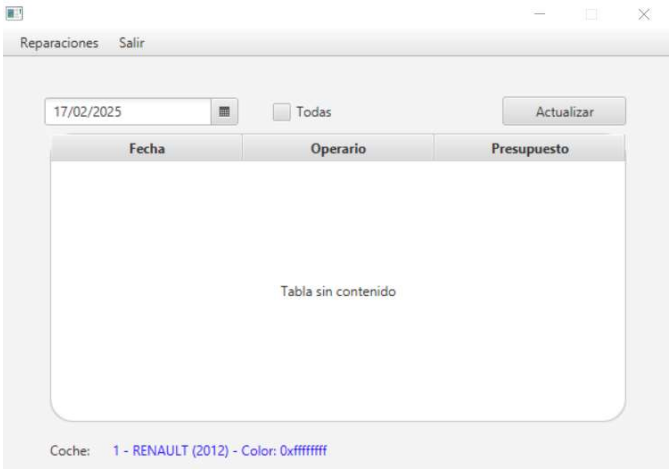
- Debe mostrar en el centro los **datos del coche** con el que hemos entrado. Puedes usar el **toString**() de la clase Coche.
- Mostrará arriba un **menú** con estas opciones:
  - **Reparaciones**:
    - **Nueva**: Mostrará la pantalla de Nueva Reparación (manteniendo el menú arriba)
    - **Buscar**: Mostrará la pantalla de Buscar Reparaciones (manteniendo el menú arriba)
  - **Salir**:
    - **Cerrar coche**: Debe volver a la pantalla anterior
    - **Cerrar aplicación**: Debe preguntar si estamos seguros de salir. En caso afirmativo, cerramos la aplicación.



PANTALLA DE NUEVA REPARACIÓN

- Debe mostrar un **mensaje estático** indicando que no tiene permisos para crear reparaciones.
- En la parte de abajo se mostrarán los **datos del coche** con el que hemos entrado. Puedes usar el toString() de la clase Coche

PANTALLA DE BUSCAR REPARACIONES



- En la parte de abajo se mostrarán los **datos del coche** con el que hemos entrado. Puedes usar el toString() de la clase Coche
- La fecha se cargará con la **fecha actual** y el checkbox estará desmarcado
- Cuando selecciono el checkbox, el campo para la fecha debe **desactivarse**. Si vuelvo a deseleccionar el checkbox, el campo fecha vuelve a **activarse**.
- Cuando le doy al botón "**Actualizar**" se deben **cargar en la tabla** todos los registros de reparaciones de la BBDD. Para consultarlos usa el servicio proporcionado. Tienes dos métodos:
  - **consultarReparaciones**(matricula) → devuelve Lista de todas las reparaciones para la matrícula del coche indicado.
  - **consultarReparaciones**(fecha, matricula) → devuelve Lista de todas las reparaciones para la matrícula del coche indicado en la **fecha** que se le pasa. Nota: sólo hay registros para fechas de hoy, mañana y pasado.
  - En ambos casos, si no hay registros, se devuelve una **lista vacía**. Si no hay registros, debes **limpiar la tabla**.
- No es necesario lanzar la llamada al servicio en un Task independiente.
- No es necesario formatear fecha ni importes en la tabla.
- Debes de aplicar a la tabla estos **estilos**. Haz lo que consideres para que así sea:

```
-fx-background-radius: 10;
-fx-border-radius: 10;
```