



Universidad de Alcalá

Escuela Politécnica Superior

Universidad de Alcalá

Práctica Final

Gestión de una biblioteca

Patrones Software

Grado en Ingeniería Informática - Curso 2019/2020

Marcos Barranquero Fernández – 51129104N

Eduardo Graván Serrano – 03212337L

ÍNDICE

Introducción	3
Análisis de requisitos	5
Requisitos funcionales	5
Requisitos no funcionales	8
Modelo de datos	11
Implementación de la base de datos	11
Arquitectura e implementación de la aplicación	13
Interacción con la BBDD	13
Interacción con el usuario	14
Interacción con el administrador	14
Sistema de préstamos y devoluciones	14
Comunicación GUI – Funcionalidad	14
Sistema de logueo	15
Estudio de patrones	16
Creación – Singleton	16
Creación – Builder	17
Creación - Factory method	18
Creación - Abstract factory	19
Estructural – Fachada	20
Estructural – Bridge	21
Estructural – Patrón adapter	22
Comportamiento – Iterator	22
Comportamiento – Command	24
Comportamiento – Strategy	25
Diagramas de clase	26
Diagrama de casos de uso	29
Diagrama de casos de uso – Administrador	29
Diagrama de casos de uso – Usuario	29
Manual de instalación	30
Manual de usuario	32
Clientes de la biblioteca	32
Administradores de la biblioteca	37

INTRODUCCIÓN

El sistema desarrollado está enfocado a dar solución a la necesidad de una biblioteca de gestionar los diferentes artículos que ofrece a sus clientes, así como de ser capaces de hacer un seguimiento de los clientes de la biblioteca.

A la hora de desarrollar la aplicación, se ha tenido en cuenta el horario de la biblioteca física para la gestión de préstamos. Este horario va de 8:00 a 20:30.

La aplicación está pensada para ser instalada en ordenadores que se encuentren físicamente en la biblioteca. Ya que la aplicación cuenta con la lógica de usuario/administrador, se puede instalar en ordenadores privados de administradores, así como en ordenadores públicos de la biblioteca.

Desde el punto de vista de los administradores, estos serán capaces de dar de alta/baja a clientes de la biblioteca, gestionar el préstamo/devolución de libros y portátiles, y se les da la posibilidad de registrar nuevos libros/portátiles en la base de datos.

Los clientes serán capaces de consultar el catálogo de libros de la biblioteca, siendo posible la búsqueda por título/autor/isbn... para facilitar la exploración del catálogo. Podrán también consultar la disponibilidad de los portátiles que presta la biblioteca desde la aplicación. A parte de esto, serán capaces de devolver libros de forma automática a la biblioteca.

Para otro tipo de gestiones más complejas, como el préstamo de libros o el préstamo/devolución de portátiles, deberán acercarse al mostrador de la biblioteca para que lo registre el administrador desde su aplicación.

En el caso de que la devolución de alguno de los artículos se haga fuera del plazo marcado por la biblioteca, se le aplicará un castigo al usuario de forma automática. Este castigo será de 1 día por cada día que de más que haya tardado en devolver un libro, y de 1 día por cada hora de más que haya tardado en devolver un portátil. Si el usuario está castigado, no podrá hacer uso de los servicios de préstamos de la biblioteca.

En cuanto a las interfaces gráficas del sistema, estas deberán ser fáciles de comprender y utilizar para todos los usuarios, teniendo especial importancia que los clientes no tengan ningún tipo de problema ni necesiten ayuda externa para el correcto uso de la aplicación.

El sistema desarrollado cuenta con un sistema de log, mediante el cual todas las acciones que haga el usuario y modifiquen de alguna manera el estado de la base de datos, así como otras acciones como login, logout, inicialización del log, conexión con la base de datos... serán mostradas por consola, así como escritas en un fichero de texto indicando el nivel de log que indica esta acción (información, warning, error), y el día y hora exactos en el que se registró esta acción.

El lenguaje de programación utilizado es Java, lo cual nos permite una gran portabilidad de forma nativa gracias a la tecnología de la JVM.

De cara al desarrollo de las interfaces, se ha hecho uso del framework Swing de Java.

Para mantener la coherencia con los datos que gestiona la aplicación, se ha hecho uso del sistema gestor de base de datos Derby. En esta base de datos se almacenan todos los datos con los que trabaja la aplicación (lista de usuarios, datos de estos usuarios, catálogo de libros, información del estado de estos libros, etcétera).

Debido a que la instalación de este sistema gestor de base de datos no es precisamente intuitiva para que funcione con aplicaciones sin servidor, se presenta un manual de instalación para la aplicación que recoge una posible forma de hacer funcionar esta base de datos.

Se presentan también manuales de usuario, tanto para administradores, como para clientes de la biblioteca. Este manual recoge el funcionamiento de todas las interfaces con las que se encontrarán los usuarios de la aplicación, explicando su funcionamiento con detalle para intentar solventar cualquier tipo de confusión a la hora de utilizar la aplicación.

En cuanto a la documentación del código, se presentan los comentarios que entendemos son necesarios para la comprensión del funcionamiento interno de los métodos, así como Javadoc para todas las clases y métodos que se han desarrollado para el sistema.

ANÁLISIS DE REQUISITOS

Esta sección de la memoria está enfocada al análisis de requisitos final que se ha tenido en cuenta a la hora de desarrollar el sistema software.

REQUISITOS FUNCIONALES

Los requisitos funcionales se pueden resumir con la siguiente lista:

RF01

Identificador	RF-01
Nombre	Validación de usuario
Funcionalidad	El sistema comprueba que tanto la contraseña como el usuario coinciden con alguno de los registrados en la base de datos de los usuarios o administradores del Sistema.

RF02

Identificador	RF-02
Nombre	Gestión de libros
Funcionalidad	El Sistema permite dar de alta y ver los libros guardados.

RF03

Identificador	RF-03
Nombre	Validación de entradas
Funcionalidad	El Sistema debe validar las entradas del usuario para impedir registro de datos erróneos.

RF04

Identificador	RF-04
Nombre	Devolución de préstamos
Funcionalidad	El sistema permite gestionar devoluciones de libros para los usuarios de la administración.

RF05

Identificador	RF-05
Nombre	Gestión de castigo de usuarios
Funcionalidad	El sistema permite penalizar a los usuarios cuyos préstamos hayan vencido.

RF06

Identificador	RF-06
Nombre	Logeo de actividad
Funcionalidad	El sistema permite monitorizar la actividad de los usuarios y administradores.

RF07

Identificador	RF-07
Nombre	Clasificación de usuarios
Funcionalidad	El sistema distingue entre usuarios de la administración de la biblioteca y clientes.

RF08

Identificador	RF-08
Nombre	Almacenamiento de los datos
Funcionalidad	El sistema almacena los datos utilizando una base de datos Derby Apache.

RF09

Identificador	RF-09
Nombre	Límite de prestamos
Funcionalidad	Cada usuario tiene un límite de libros y artículos asignados máximos que puede tener prestados a la vez.

RF10

Identificador	RF-10
Nombre	Atomicidad de datos
Funcionalidad	Las operaciones con datos y asignaciones deben de ser atómicas respecto al almacenamiento de datos.

RF11

Identificador	RF-11
Nombre	Congruencia de datos
Funcionalidad	Las operaciones con datos y asignaciones deben de ser congruentes en la aplicación respecto al almacenamiento de datos.

RF12

Identificador	RF-12
Nombre	Congruencia con el horario de la biblioteca
Funcionalidad	La aplicación no debe permitir realizar ni devolver préstamos fuera del horario de la biblioteca.

RF13

Identificador	RF-13
Nombre	Alta de datos
Funcionalidad	Debido a que la biblioteca recibe varias unidades de los mismos libros y portátiles, la aplicación debe permitir dar de alta un número de las mismas unidades a la vez.

RF14

Identificador	RF-14
Nombre	Visualización de libros
Funcionalidad	La aplicación debe permitir consular los libros ordenando por ID, autor, título e ISBN.

RF15

Identificador	RF-15
Nombre	Visualización de portátiles
Funcionalidad	La aplicación debe permitir consular los portátiles disponibles.

RF16

Identificador	RF-16
Nombre	Limitación de los administradores.
Funcionalidad	La aplicación no debe permitir a los administradores tomar prestados libros o portátiles, pues tienen su propio ordenador.

REQUISITOS NO FUNCIONALES

Los requisitos no funcionales se pueden resumir con la siguiente lista:

RNF01

Identificador	RNF-01
Nombre	Rendimiento
Descripción	La aplicación debe ser ligera para hacer un buen uso de los recursos hardware del terminal en el que se ejecute. Requisitos mínimos: Pentium II, 512MB de RAM, 500MB de almacenamiento en disco disponible.

RNF02

Identificador	RNF-02
Nombre	Disponibilidad
Descripción	El sistema debe ser accesible a cualquier hora del día, todos los días de la semana.

RNF03

Identificador	RNF-03
Nombre	Accesibilidad
Descripción	La interfaz gráfica de usuario debe ser clara y limpia para facilitar la navegabilidad de los usuarios. La aplicación no debería necesitar ningún tipo de documentación complementaria para su uso.

RNF04

Identificador	RNF-04
Nombre	Seguridad
Descripción	El sistema realizará copias de seguridad frecuentemente. Se debe seguir el RGPD en el tratamiento de los datos de los usuarios de la aplicación. El administrador del sistema es el encargado de realizar dichas copias de seguridad.

RNF05

Identificador	RNF-05
Nombre	Disponibilidad multiplataforma
Descripción	El sistema se podrá ejecutar sobre los principales sistemas operativos (Windows 7 en adelante, Linux, macOS).

RNF06

Identificador	RNF-06
Nombre	Lenguaje de programación
Descripción	El lenguaje de programación a utilizar para el desarrollo de la aplicación será Java en su versión 8.

RNF07

Identificador	RNF-07
Nombre	Interfaz gráfica
Descripción	El sistema tendrá una interfaz gráfica desarrollada utilizando la librería Swing.

RNF08

Identificador	RNF-08
Nombre	Manual de usuario
Descripción	Se entregará un manual de usuario que detalle el uso de la aplicación y los aspectos básicos del funcionamiento del sistema que los usuarios deban conocer.

RNF09

Identificador	RNF-09
Nombre	Empleo de patrones de diseño
Descripción	La aplicación debe desarrollarse haciendo uso de patrones software.

RNF10

Identificador	RNF-10
Nombre	Escalabilidad y modularidad de la aplicación
Descripción	El sistema debe ser escalable y modular, para poder permitir mantenimiento e implementaciones de funcionalidad futuras.

RNF11

Identificador	RNF-11
Nombre	Uso de métodos adecuados en relación a seguridad y estabilidad
Descripción	El sistema no debe utilizar métodos “deprecated” que puedan comprometer el rendimiento, la seguridad o estabilidad de la aplicación.

RNF12

Identificador	RNF-12
Nombre	Estándares de buenas practices
Descripción	El sistema debe emplear buenas prácticas relacionadas con las tecnologías empleadsa en su fase de desarrollo, y debe seguir el estándar CamelCase.

RNF13

Identificador	RNF-13
Nombre	Interacción del texto
Descripción	El sistema permite las opciones de selección, copiado y pegado de texto.

MODELO DE DATOS

En cuanto al modelo de datos, se ha hecho uso del sistema gestor de base de datos Derby. Este gestor es propiedad de Apache y viene integrado con la versión Apache del IDE Netbeans.

En el manual de instalación se explican los pasos necesarios para hacer funcionar este sistema gestor de base de datos. Se hace entrega también de los ficheros SQL necesarios para crear las tablas y poblarlas con algunos datos de prueba.

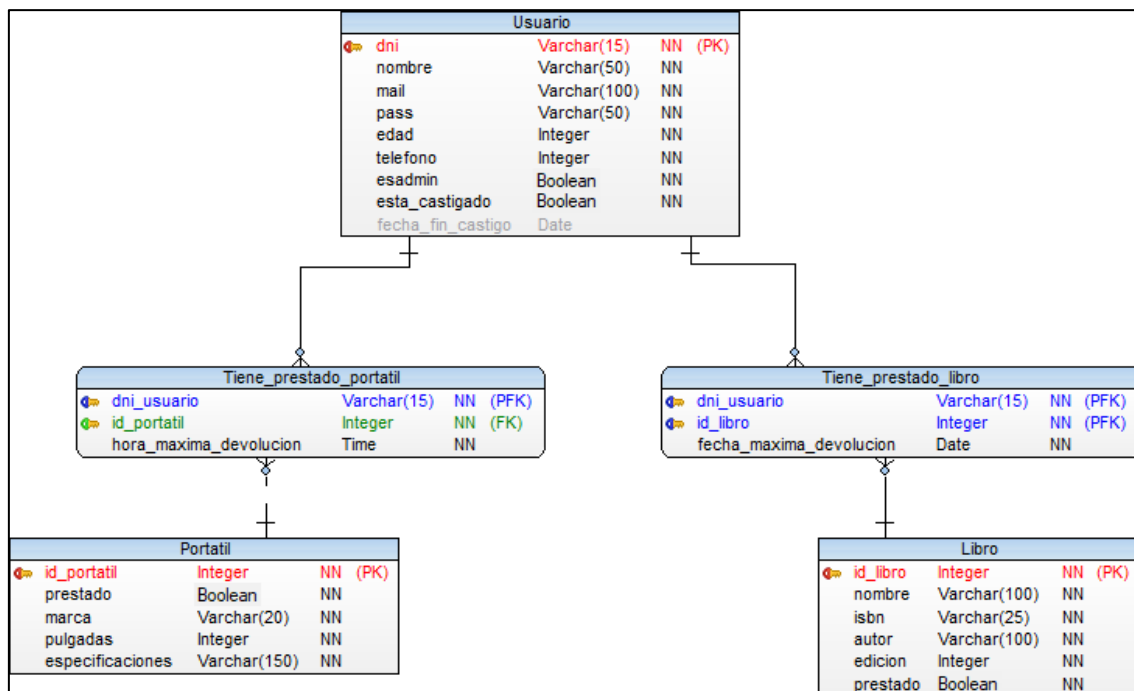
IMPLEMENTACIÓN DE LA BASE DE DATOS

Haciendo un análisis de los datos que maneja la aplicación, se ve la necesidad de tener un esquema de persistencia de datos sobre los siguientes puntos:

- Usuarios (Clientes/Administradores)
- Libros
- Portátiles
- Estado de los artículos anteriores
- Relaciones entre los clientes y los artículos que ofrece la biblioteca:
 - Préstamos de libros
 - Préstamos de portátiles

Con todo esto, se ha desarrollado un esquema de base de datos bastante sencillo.

Este esquema ha sido modelado con Toad Data Modeler en su versión 7.1, haciendo uso del modelo universal (no hay soporte de Derby). El modelo es el siguiente:



Se adjunta también el archivo .txp que almacena el modelo para Toad Data Modeler. Se debe decir, que la imagen de arriba ha sido modificada para añadir el tipo Boolean, ya que no están presentes en el modelo universal de Toad Data Modeler, por lo que, si se abre, se podrá ver que los datos marcados como Boolean están configurados como Integer.

La base de datos se ha construido para ser lo más cercana posible a la lógica de la aplicación que se ha desarrollado. Por ello, se tienen características como las siguientes:

- En la composición de la clave primaria de la tabla “Tiene_prestado_portatil”. Debido a que un cliente sólo puede tener prestado un portátil en un determinado momento, se ha implementado esta limitación a nivel de base de datos para asegurarnos que, en cualquier caso, el usuario no podría tomar prestados más que un portátil.
- En el caso de la tabla “Tiene_prestado_libro”, como el cliente puede tomar prestado más de un libro a la vez, se usa una clave primaria compuesta entre las claves primarias de las tablas que relaciona (“Usuario” y “Libro”).
- Ningún campo de la base de datos puede almacenar valores nulos, excepto “fecha_fin_castigo” de la tabla “Usuario”. Esto se debe a que esta fecha de fin de castigo solo puede existir si el usuario está castigado.

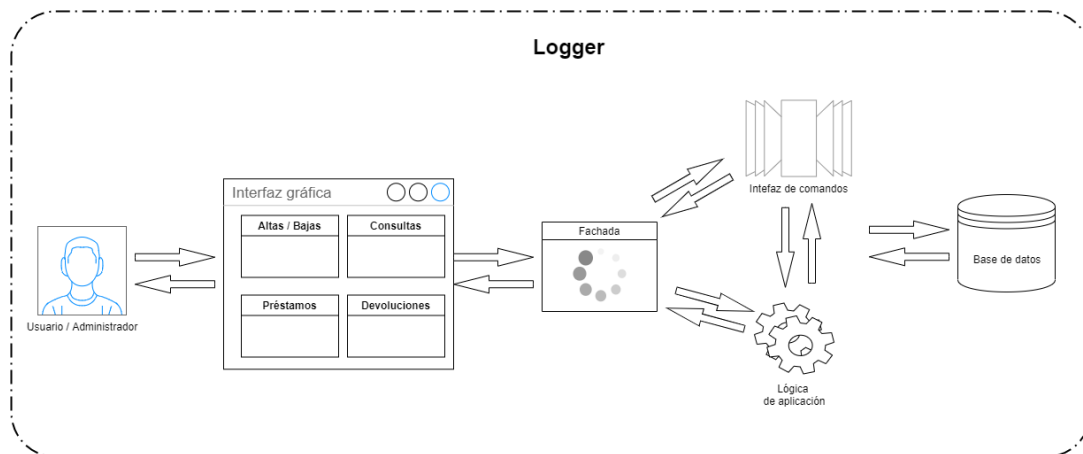
ARQUITECTURA E IMPLEMENTACIÓN DE LA APLICACIÓN

En este apartado se presentará una vista general de la arquitectura de la aplicación.

Como una visión general, la aplicación se puede dividir en seis módulos:

1. **Interacción con la base de datos**, que comprende todas las operaciones de alta, baja, modificaciones sobre la BBDD.
2. **Interacción con el usuario**, que engloba las operaciones que puede realizar el usuario desde la aplicación.
3. **Interacción con el administrador**, similar a lo descrito para el usuario, pero esta vez para el administrador.
4. **Sistema de préstamos y devoluciones**, que estudia las operaciones propias que se pueden realizar en la biblioteca.
5. **Comunicación del sistema con la GUI**, que describe la forma que tiene de interactuar la interfaz con la funcionalidad de la aplicación.
6. **Sistema de logueo**, que permite loguear toda la actividad de la aplicación.

A modo de referencia rápida, podemos representar la arquitectura con el siguiente diagrama:



INTERACCIÓN CON LA BBDD

Para interactuar con la base de datos, en primer lugar, se tiene una clase que permite lanzar directamente consultas sobre la base de datos, y devolver los resultados de esta.

Para controlar dichas consultas y no poder ejecutar ahí cualquier cosa, se utilizan clases de comandos relacionadas con los distintos componentes de la base de datos: comandos para libros, para usuarios, etc., que tienen descrita toda la funcionalidad necesaria para dar de alta, baja, recuperar elementos concretos, así como cosas más complejas como llevar a cabo las operaciones necesarias para los préstamos y devoluciones.

INTERACCIÓN CON EL USUARIO

El usuario puede iniciar y cerrar sesión, consultar y devolver libros, y consultar la disponibilidad de portátiles.

Para el inicio y cierre de sesión, se recupera el usuario de la base de datos mediante comandos y se pasa como argumento por las distintas interfaces gráficas.

Para la consulta y devolución de libros, se vuelve a interactuar con la base de datos leyendo los libros prestados y disponibles. En la consulta, puede ordenar los libros en función de su autor, isbn, título, etc.

Para la consulta de portátiles se lee una vez más de la BBDD.

Todas estas operaciones pasan por la Fachada, que es una interfaz de comunicación entre la interfaz gráfica y las operaciones reales del programa.

INTERACCIÓN CON EL ADMINISTRADOR

El administrador puede dar de alta y baja usuarios, y crear, prestar y devolver libros y portátiles.

Todas estas operaciones se pueden realizar desde el panel del administrador. Al dar de alta o baja elementos, la GUI verifica que los campos rellenados son correctos y no contienen inconsistencias antes de lanzar el comando que interactúa con la base de datos.

Una vez más, todas estas operaciones pasan por la fachada de comunicación.

SISTEMA DE PRÉSTAMOS Y DEVOLUCIONES

El administrador puede prestar y devolver libros y portátiles. Para ello, debe tener físicamente el libro o portátil a prestar y el DNI del usuario, e introducirlos en la interfaz. Puede cargar los datos del libro o portátil asociado metiendo su identificador.

Al realizar el préstamo, se carga directamente la fecha de devolución. Al realizar la devolución, se castiga automáticamente al usuario si se ha pasado del plazo de préstamo. La aplicación muestra mensajes denotando esta información.

Se dota al usuario con la capacidad de devolver libros. Se cargan los libros que posee en una lista, y podrá seleccionar aquel que devuelve.

Una vez más, todas estas operaciones pasan por la fachada de comunicación y lanza los comandos necesarios sobre la base de datos.

COMUNICACIÓN GUI – FUNCIONALIDAD

Toda operación que se pueda realizar desde la GUI o interfaz gráfica pasa por la fachada para realizar las operaciones necesarias en el sistema y en la BBDD.

Esta arquitectura permite desacoplar la interfaz de las operaciones, dando facilidad a que la aplicación sea mantenible y actualizable.

La GUI es la encargada de validar todos los campos rellenados para las operaciones, y una vez se han verificado, llama a la fachada pasándole los argumentos sin posibilidad de que contengan errores. La fachada realizará la operación necesaria, y, si es necesario, devolverá los resultados de dicha operación para mostrarlos por la interfaz.

SISTEMA DE LOGUEO

La aplicación permite loguear toda la actividad del usuario o administrador en la aplicación. Esto incluye altas, bajas, devoluciones, etc. Para llevar acabo esto, se tiene una clase `Logger` que se llama transversalmente desde los métodos de las operaciones, detallando la operación realizada.

Este log escribe tanto por consola como en un documento las operaciones realizadas. Tiene 3 niveles de logueo: de información, de warning o precaución, y de error. Según el resultado de las distintas operaciones, se llamará con un nivel u otro.

ESTUDIO DE PATRONES

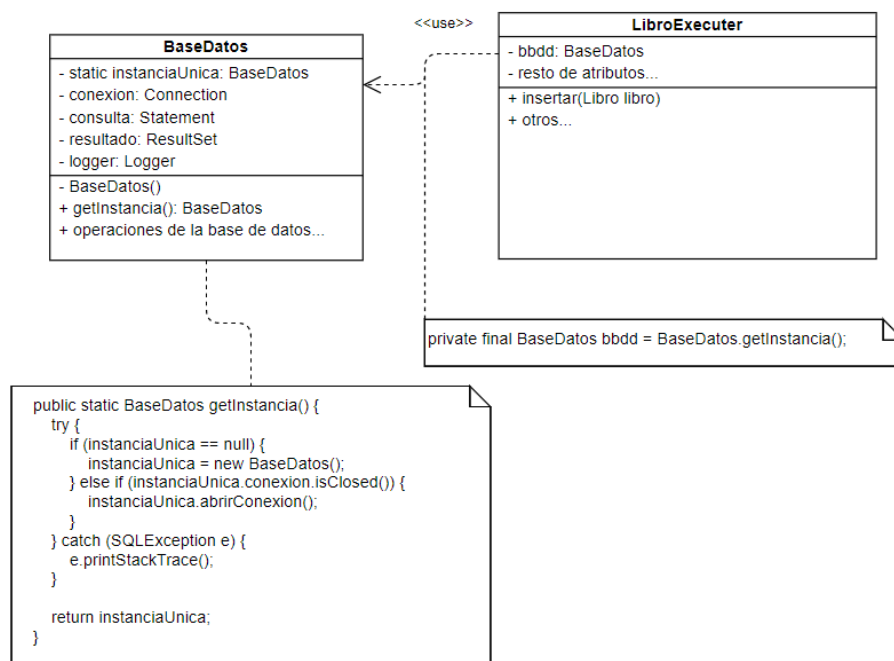
A la hora de desarrollar la aplicación, se han utilizado los siguientes patrones software:

CREACIÓN – SINGLETON

Se ha empleado el patrón singleton para la clase *BaseDatos*, de forma que no se esté creando una nueva instancia y conexión con la base de datos de derby cada vez que se usa en una clase distinta.

A su vez, se ha combinado este patrón con el Factory en las factorías, es decir, las factorías son objetos Singleton.

También se ha combinado con el patrón Fachada, convirtiendo la fachada en un objeto Singleton con instancia única para todas las clases de interfaz gráfica.

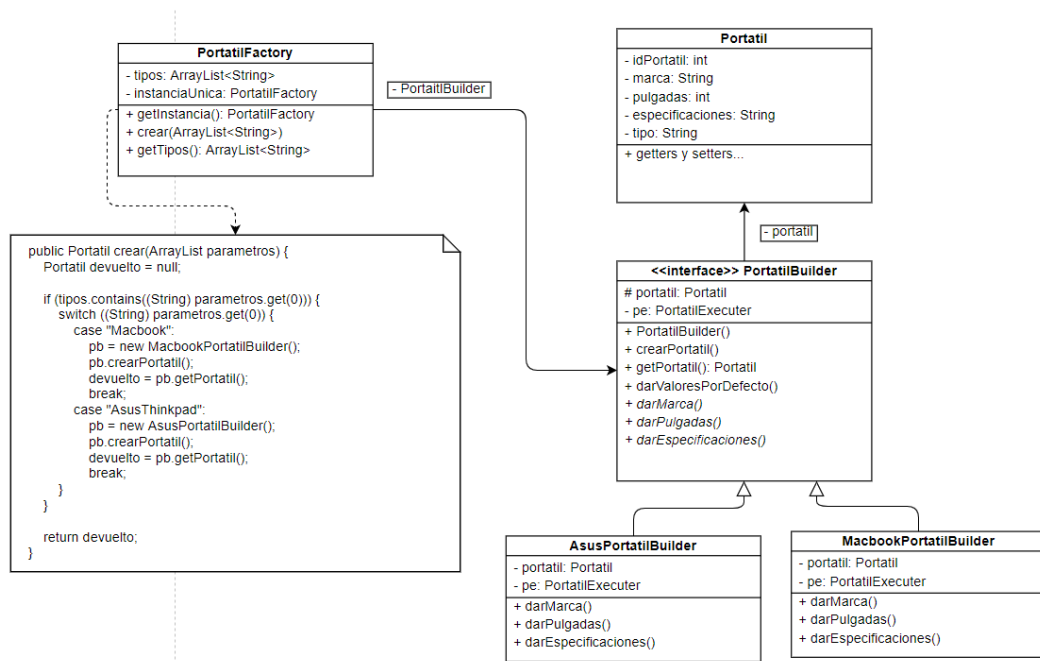


CREACIÓN – BUILDER

Se ha utilizado el patrón builder para la creación de portátiles. Se han creado dos tipos distintos de portátil:

- Macbook: portátil de Apple
- Asus Thinkpad: portátil de asus.

Ambos tienen especificaciones distintas, pero en atributos compartidos, de forma que se adapta perfectamente al uso de este patrón. Para instanciarlos, se ha combinado con el patrón Factory Method, teniendo así una factoría de portátiles que en función del String que reciban crearán un portátil de tipo Asus o Apple. El siguiente diagrama de clases muestra su uso:

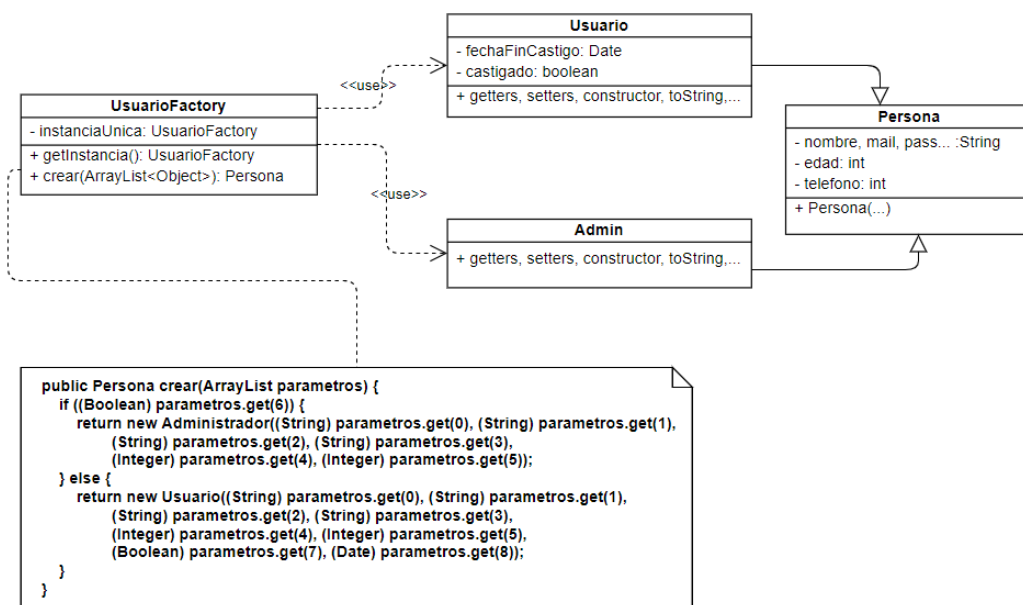


CREACIÓN - FACTORY METHOD

Se emplean varias factorías de métodos: para portátiles y para usuarios.

La idea es emplear una factoría de métodos que, en función de los argumentos pasados en el ArrayList, cree un elemento de un tipo u otro o con unas características concretas. Los argumentos vienen en un ArrayList para poder emplear un Abstract Factory Method para todas las factorías.

- Para el caso de portátiles, en función del argumento, que es una String, se llamará al builder creando un portátil de un tipo u otro.
- Para el caso de usuarios, se emplea un booleano que determina si es administrador o no, creando así un usuario o un administrador. Los argumentos vienen en un ArrayList para poder emplear un Abstract Factory Method para todas las factorías.



CREACIÓN - ABSTRACT FACTORY

Debido a que se iban a tener tres fábricas distintas para crear las tres unidades de información básicas de la aplicación (Usuarios, Libros, Portátiles), se decidió implementar el patrón Abstract Factory para unificar la creación de estas fábricas bajo una interfaz.

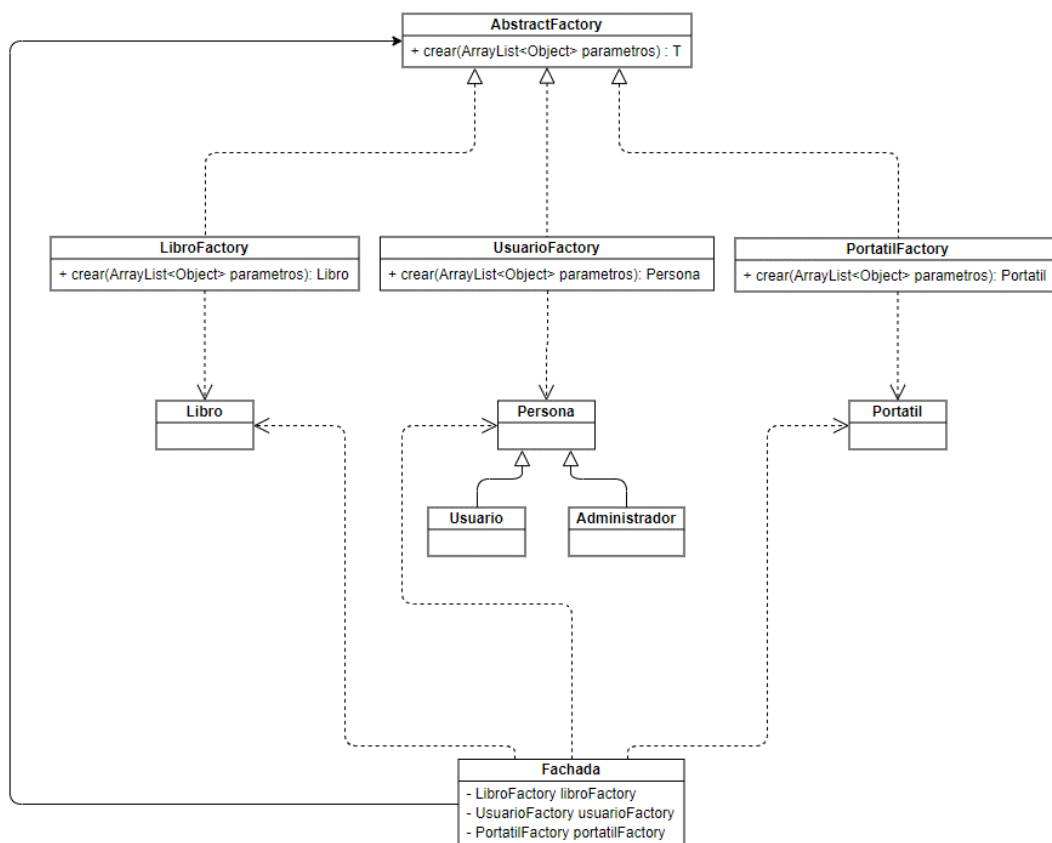
Las fábricas concretas reciben como parámetro un ArrayList de objetos e instancian las clases concretas en base usando este ArrayList como forma de recibir los atributos.

En el interior de estas fábricas se aplican otros patrones como el Singleton, el Factory Method o el Builder.

Las factorías son usadas desde:

- Clase Fachada
- Clases ejecuter (intermediarias con la base de datos)

En el siguiente diagrama, se muestra la interacción de este patrón con la Fachada.



Las llamadas a la creación de este tipo de objetos se hacen siempre preparando el ArrayList de Objetos de antemano. El comportamiento interno de la creación es la recuperación de estos objetos del ArrayList, casteándolos al tipo que sea necesario para satisfacer los constructores de cada clase.

Por ejemplo, en el caso de la instanciación de un libro:

```
ArrayList<Object> atributosLibro = new ArrayList<>();
atributosLibro.add(idLibro);
atributosLibro.add(edicion);
atributosLibro.add(nombre);
atributosLibro.add(isbn);
atributosLibro.add(autor);
atributosLibro.add(false);
Libro aInsertar = libroFactory.crear(atributosLibro);
```

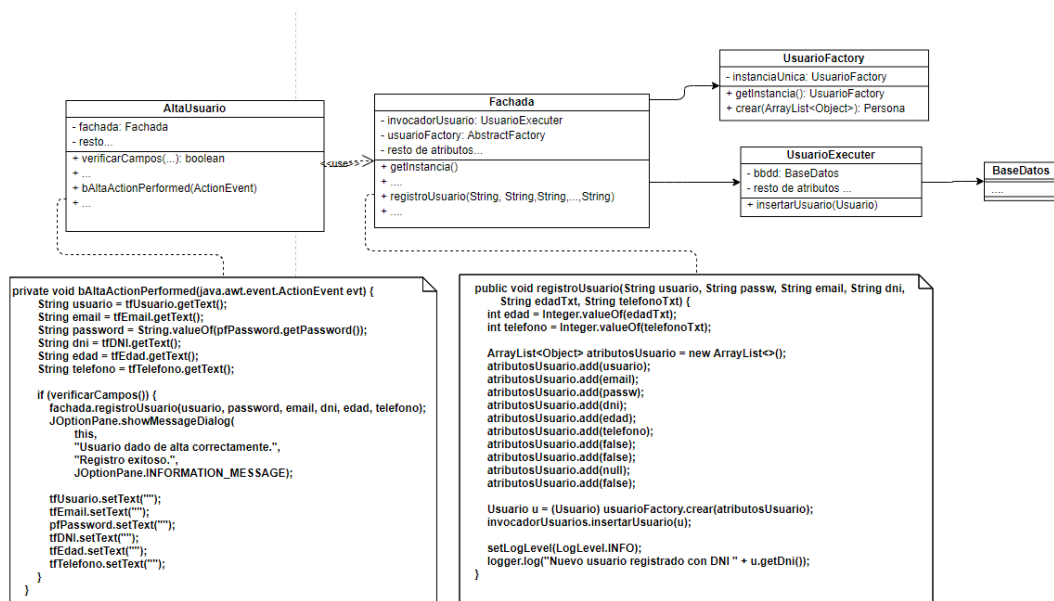
Y el método de creación:

```
public Libro crear(ArrayList parametros) {
    return new Libro((Integer)parametros.get(0),
        (Integer)parametros.get(1),
        (String)parametros.get(2),
        (String)parametros.get(3),
        (String)parametros.get(4),
        (Boolean)parametros.get(5));
}
```

ESTRUCTURAL – FACHADA

Se ha empleado el patrón fachada para comunicar todas las GUIs con el resto de la aplicación. De esta forma, el código de la GUI está destinado a verificar los campos, y si todos ellos son correctos y no crea inconsistencias en la BBDD (dni erróneos, préstamo de libros que no existan, etc), y la fachada se ocupa de interactuar con la base de datos, crear y devolver usuarios, gestionar préstamos, etc. Esto permite modularidad en la aplicación, que facilita su mantenibilidad y escalabilidad.

Como ejemplo, podemos ver en el siguiente diagrama las llamadas a la creación de usuario. Podemos observar que la GUI verifica los diferentes campos y delega el resto de proceso de creación de usuario en la fachada. A su vez, la fachada llama al executor de usuarios, creando el usuario en la base de datos.



ESTRUCTURAL – BRIDGE

Se ha utilizado al patrón Bridge a la hora de desacoplar las opciones de logeo que tiene la aplicación del resto de la aplicación.

De esta forma, se ha conseguido que la aplicación pueda crecer de forma totalmente independiente a las clases de log, y viceversa.

El log se encarga de recibir un mensaje, añadirle una fecha y una hora concretos y sacarlo por la salida de consola del programa. Además, lo escribe al fichero de texto del log del día en que está registrando los datos.

En este caso, se ha hecho uso de una interfaz funcional para dar cuerpo a las diferentes técnicas de log. Esto se debe a que la interfaz log tiene un único método y se veía más conveniente el uso de lambdas para crear las clases que implementan esta interfaz. Por ejemplo, para logear un mensaje de tipo info, tenemos el siguiente método con clase anónima:

```
public static Logger Info() {  
    return mensaje -> {  
        // Saco el mensaje por pantalla  
        System.out.println("INFO [" + DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss").format(LocalDate.now()) + "]: " + mensaje);  
        // Si no existe el directorio de logs, lo creo  
        new File("./logs").mkdir();  
        // Escribo el mensaje al archivo de log correspondiente  
        try (FileWriter fw = new FileWriter("./logs/" + DateTimeFormatter.ofPattern("dd-MM-yyyy").format(LocalDate.now()) + ".log", true)) {  
            fw.write("INFO [" + DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss").format(LocalDate.now()) + "]: " + mensaje + "\r\n");  
        } catch (IOException ex) {  
            System.out.println("Error al escribir en el archivo de log: " + ex.getMessage());  
        }  
    };  
}
```

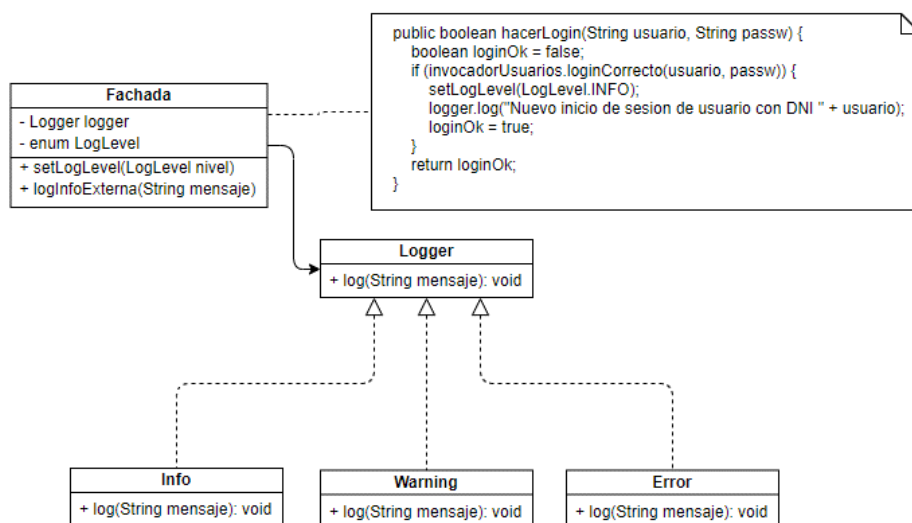
Esto nos permite la instanciación de las clases de log de la siguiente forma:

```
this.logger = Logger.Info();
```

```
this.logger = Logger.Warning();
```

```
this.logger = Logger.Error();
```

Aun así, el diagrama de clases se presenta haciendo la distinción entre clases:



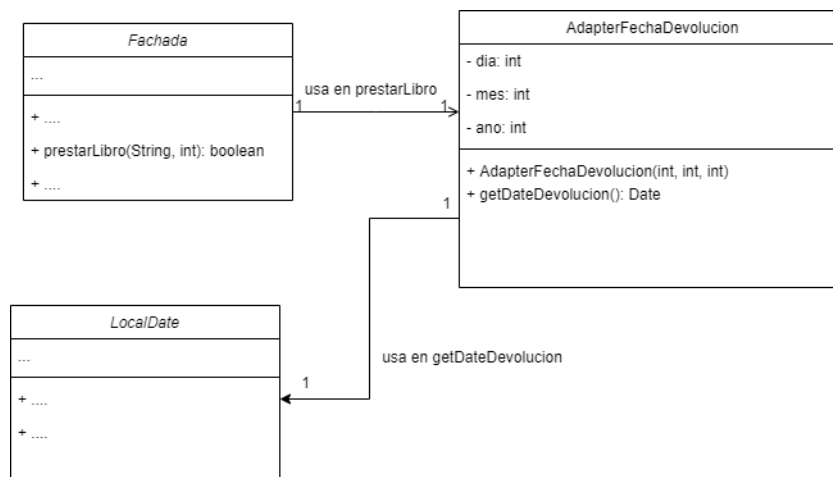
Esto es simplemente un ejemplo, las llamadas al log se hacen por toda la aplicación.

ESTRUCTURAL – PATRÓN ADAPTER

Se han utilizado Adapters o adaptadores para traducir las fechas y horas para calcular la fecha u hora de devolución. Son adaptadores no basados en interfaz.

- La clase *AdapterFechaDevolucion* recibe un día, mes y año del préstamo, y actúa calculando y permitiendo obtener la fecha máxima de devolución de un libro.
- La clase *AdapterHoraDevolucion* recibe minutos y horas del préstamo, y actúa calculando y permitiendo obtener la hora máxima de devolución de un portátil.

El siguiente diagrama de clases muestra la estructura del patrón con *AdapterFechaDevolucion*. Para el adaptador de hora, el diagrama es similar:



COMPORTAMIENTO – ITERATOR

Se ha creado una colección propia que hace de envoltorio de un *ArrayList* con el objetivo de poder construir un *Iterator* propio.

El iterador cumple las funciones básicas de cualquier iterador, que son:

- Comprobar si hay un elemento siguiente y moverse a él si es así
- Recuperar los datos para un elemento en concreto en función de la posición del iterador
- Reiniciar el iterador a la posición inicial

El iterador es usado siempre que se desea iterar sobre la colección de libros. Un ejemplo sería al rellenar en la interfaz gráfica la tabla de libros en la biblioteca:

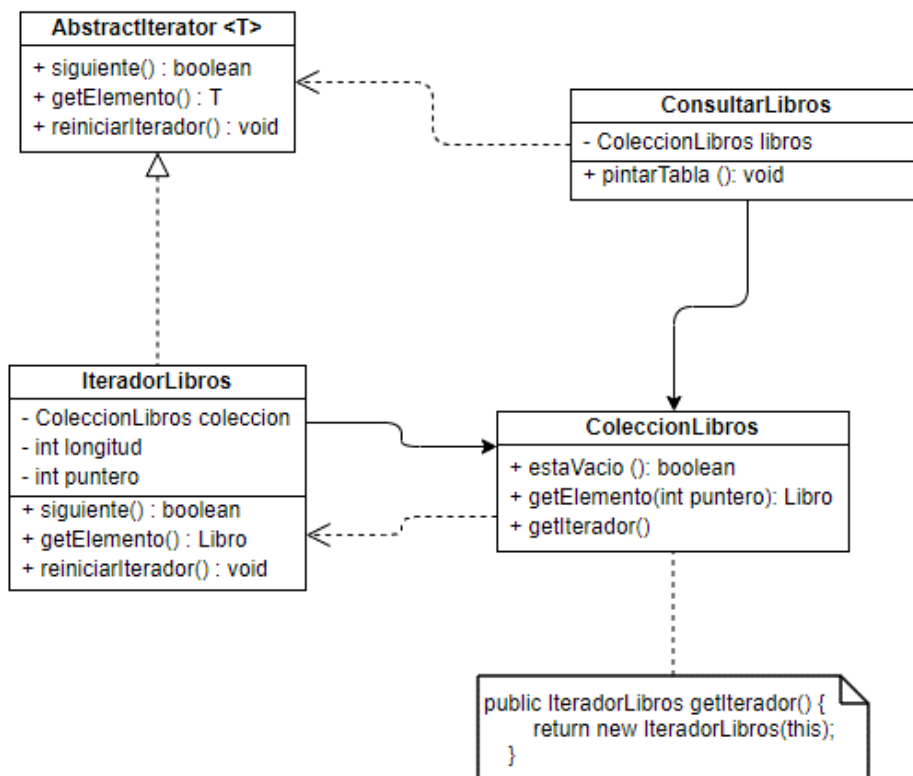
```

private void pintarTabla() {
    IteradorLibros it = libros.getIterador();
    int i = 0;

    Libro libro;
    while (it.siguiente()) {
        libro = it.getElemento();
        this.tbLibros.setValueAt(libro.getIdLibro(), i, 0);
        this.tbLibros.setValueAt(libro.getNombre(), i, 1);
        this.tbLibros.setValueAt(libro.getIsbn(), i, 2);
        this.tbLibros.setValueAt(libro.getAutor(), i, 3);
        this.tbLibros.setValueAt(libro.getEdicion(), i, 4);
        this.tbLibros.setValueAt(libro.estaPrestado() ? "Prestado" : "Disponible", i, 5);
        i++;
    }
}

```

El patrón Iterator desarrollado para nuestro sistema queda representado por el siguiente diagrama de clases:

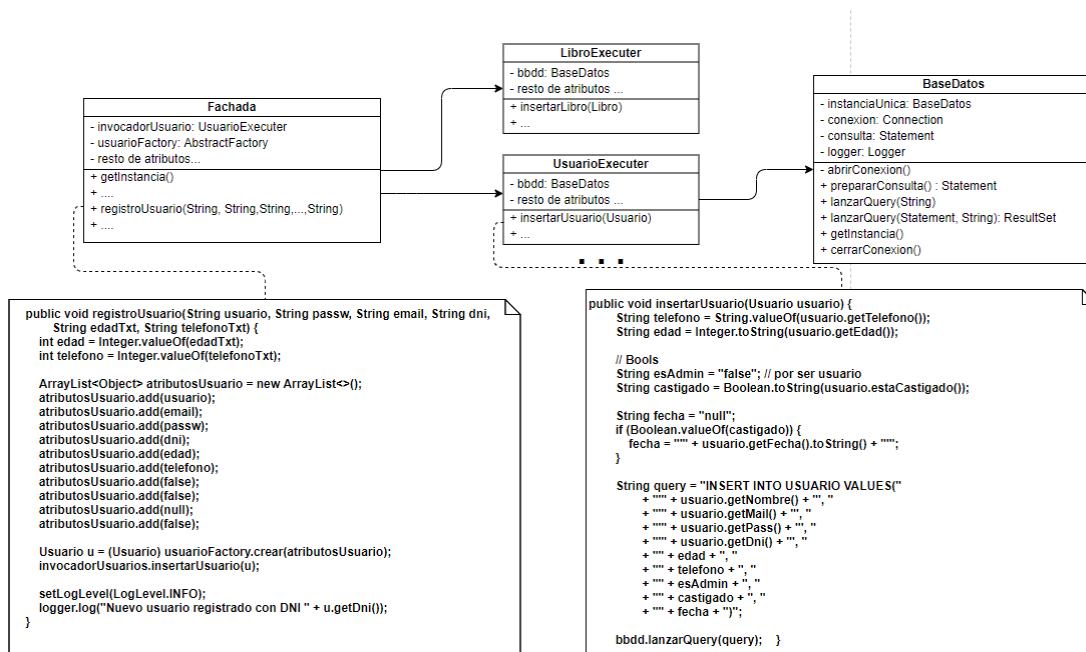


COMPORTAMIENTO – COMMAND

Se ha implementado el patrón de comando con objetivo de desacoplar las consultas de la base de datos de la clase dedicada a ejecutar las consultas sobre la base de datos. De esta forma, en lugar de tener todos los métodos que interactúan con la base de datos en una misma clase enorme, tenemos los comandos que separan las consultas en función de los elementos con los que interactúan:

- **LibroExecuter** contiene todas las consultas relacionadas con los libros.
- **UsuarioExecuter** contiene todas las consultas relacionadas con los usuarios y administradores.
- **PortatilExecuter** contiene las consultas relacionadas con los portátiles.
- **PrestamoLibrosExecuter y PrestamoPortatilExecuter** tienen las consultas relacionadas con los préstamos.

En el siguiente diagrama podemos ver el caso concreto de dar de alta a un usuario. Desde la GUI, una vez validados los datos, se llama a la fachada con los argumentos necesarios para crear el usuario. Desde la fachada, se llama al UsuarioExecuter pasándole el usuario que queremos crear. El usuarioExecuter carga la query y la ejecuta contra la base de datos.



COMPORTAMIENTO – STRATEGY

Se ha implementado el patrón Strategy para la ordenación de colecciones de libros en base a los diferentes atributos posibles que puede tener la clase Libro.

Para ello tenemos las siguientes clases:

- Interfaz OrdenarLibros
- Clase OrdenarLibrosAutor
- Clase OrdenarLibrosEdicion
- Clase OrdenarLibrosEstado
- Clase OrdenarLibrosID
- Clase OrdenarLibrosISBN
- Clase OrdenarLibrosTitulo

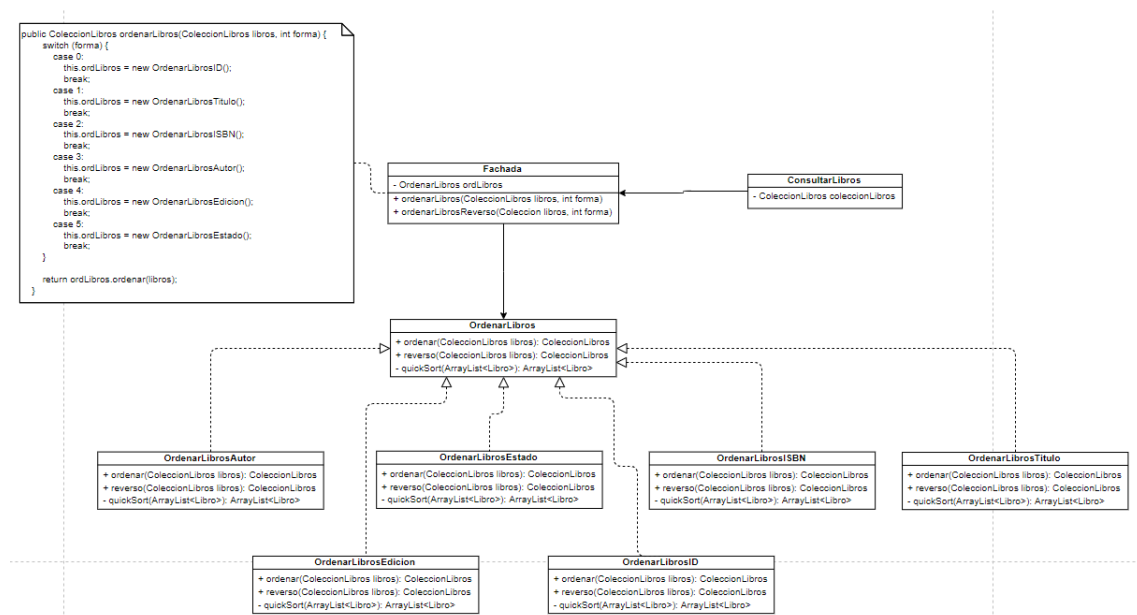
Todas ellas llaman a un método privado interno que se encarga de ordenar la colección haciendo uso del algoritmo de ordenación Quick Sort. Esta ordenación se hará comparando Strings, ints, e incluso booleans, dependiendo del atributo de la clase por el que estemos ordenando.

Este patrón se usa exclusivamente para ordenar los elementos de la tabla que nos muestra los libros que tiene la biblioteca desde la interfaz de usuario.

Al hacer click en una de las cabeceras de esta tabla, estamos eligiendo una de estas estrategias y haciendo que se ejecute el algoritmo correspondiente.

Desde la clase de la interfaz se determina si se quiere ordenar de forma descendente o ascendente, desde la fachada se determina el algoritmo de ordenación que queremos usar, se instancia ese algoritmo de ordenación y se hace la llamada a ordenar.

El diagrama de clases para este patrón es el siguiente:



DIAGRAMAS DE CLASE

Los diagramas de clase se envían adjuntos en una carpeta aparte para su mejor visualización. Aún así, hemos decidido incluirlos en el documento. Se encuentran distribuidos en función de lo relacionadas que están las clases para su mejor visualización.

DIAGRAMA DE CLASES – GUI Y FACHADA

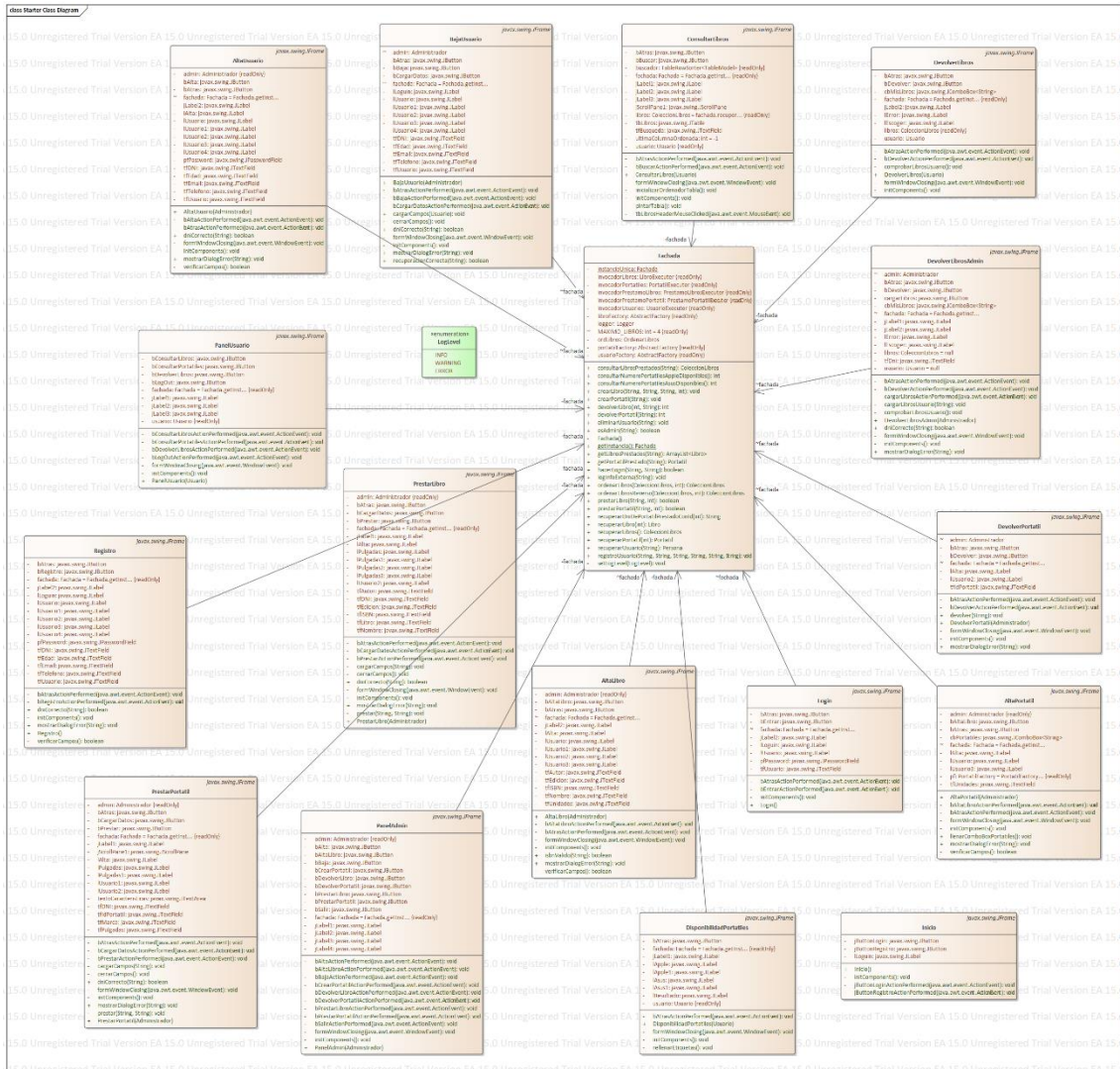


DIAGRAMA DE CLASES – FACTORY Y EXECUTERS

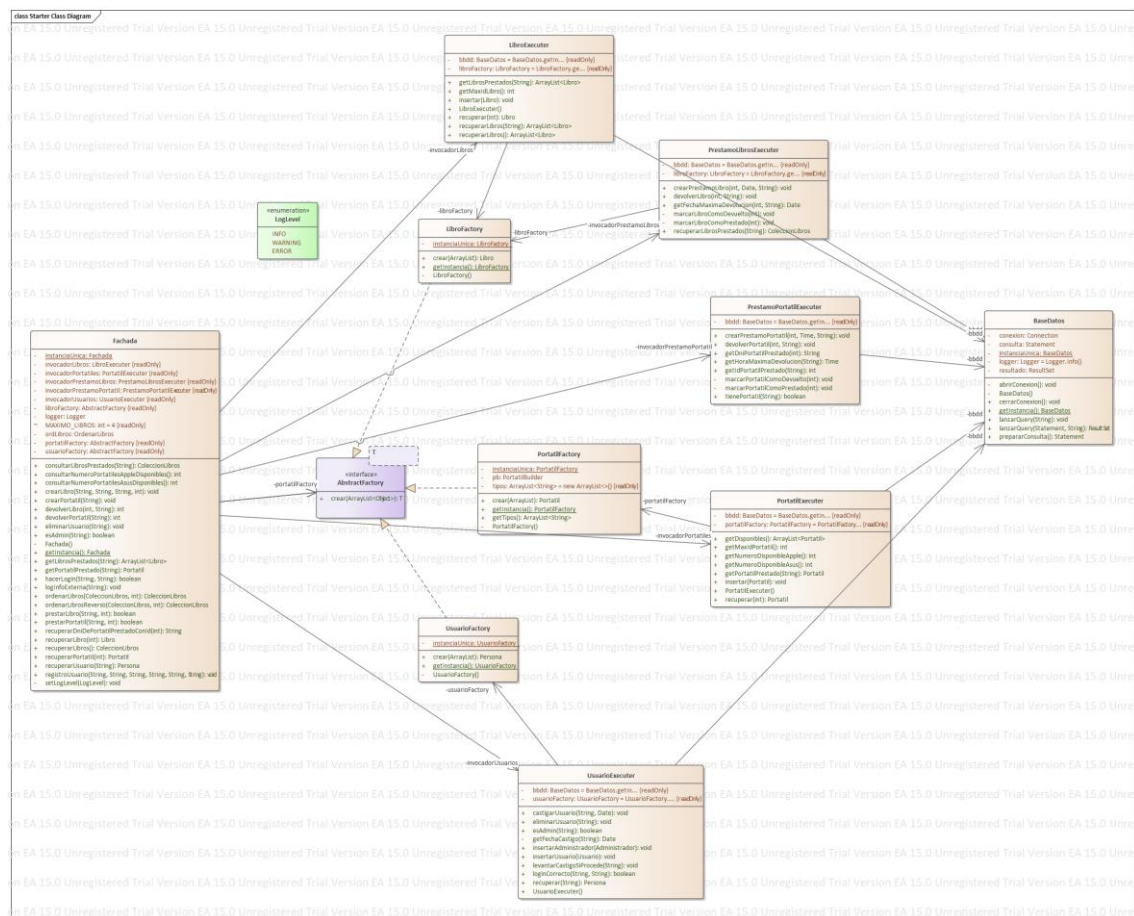


DIAGRAMA DE CLASES – STRATEGY

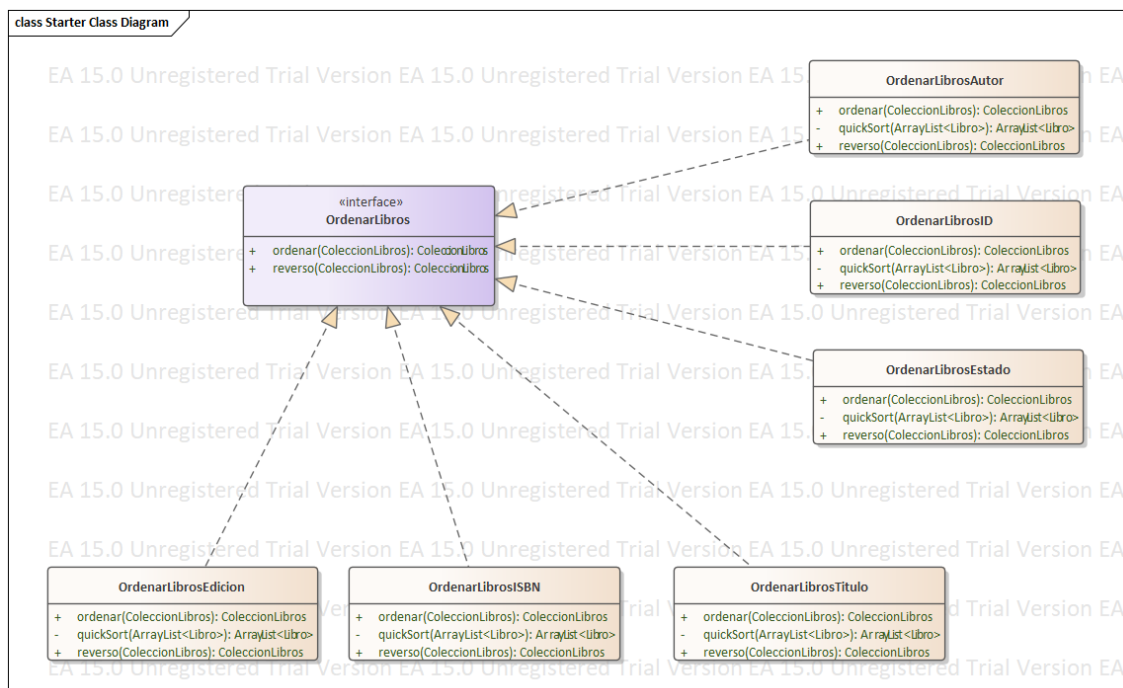


DIAGRAMA DE CLASES – CLASES BASE E ITERATOR

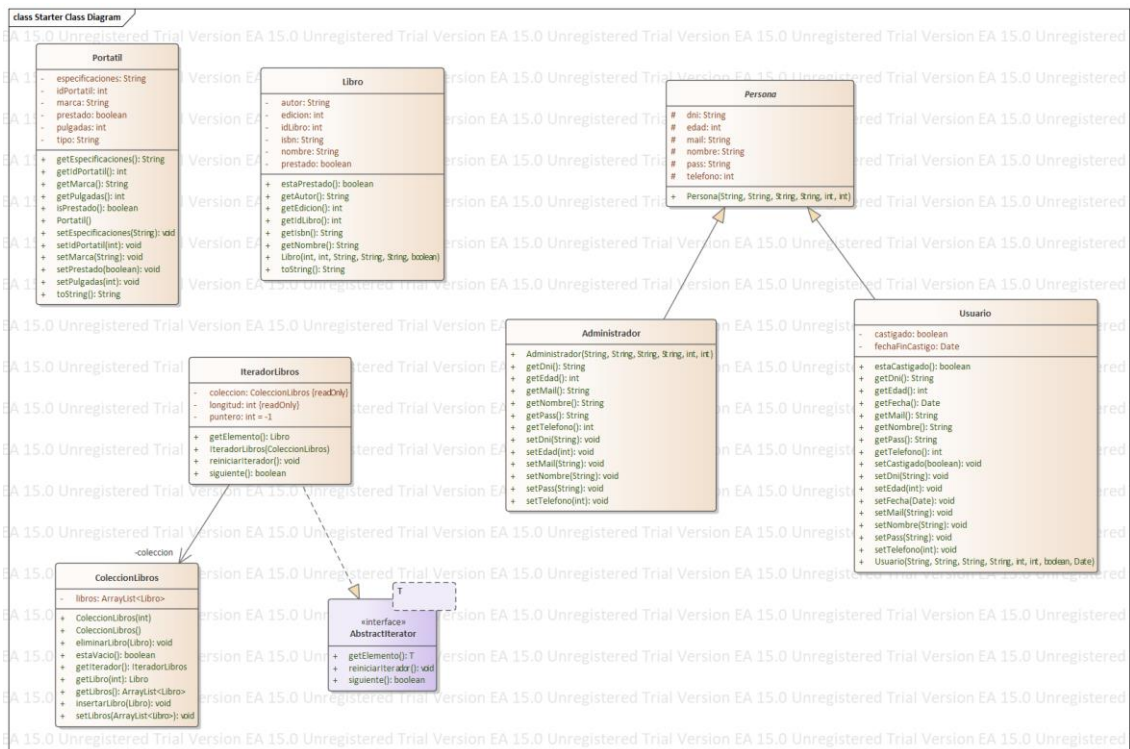


DIAGRAMA DE CLASES – BUILDER

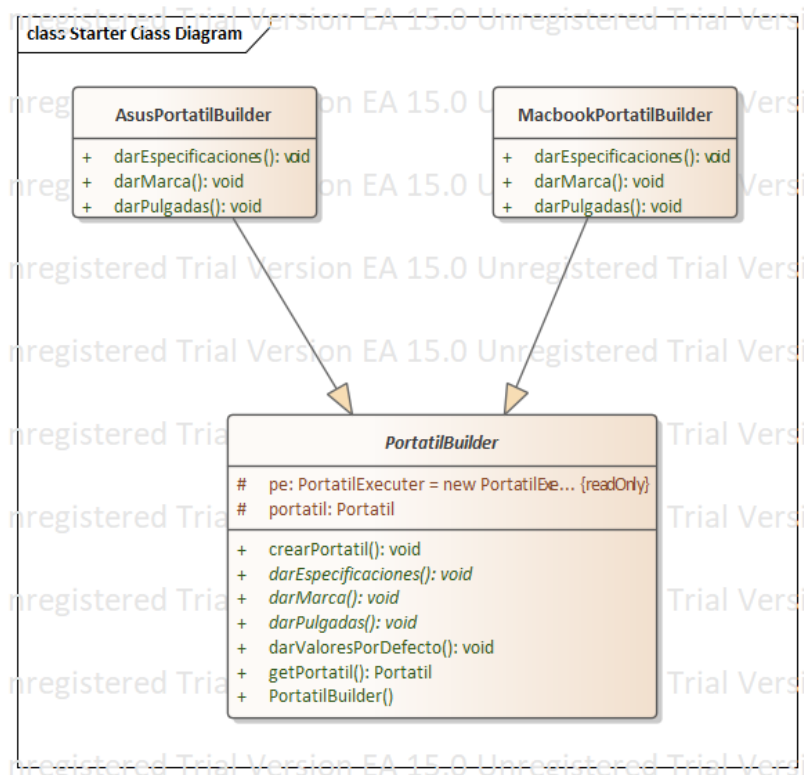


DIAGRAMA DE CASOS DE USO

Se adjuntan los siguientes diagramas de casos de uso, para el administrador y para el usuario.

DIAGRAMA DE CASOS DE USO – ADMINISTRADOR

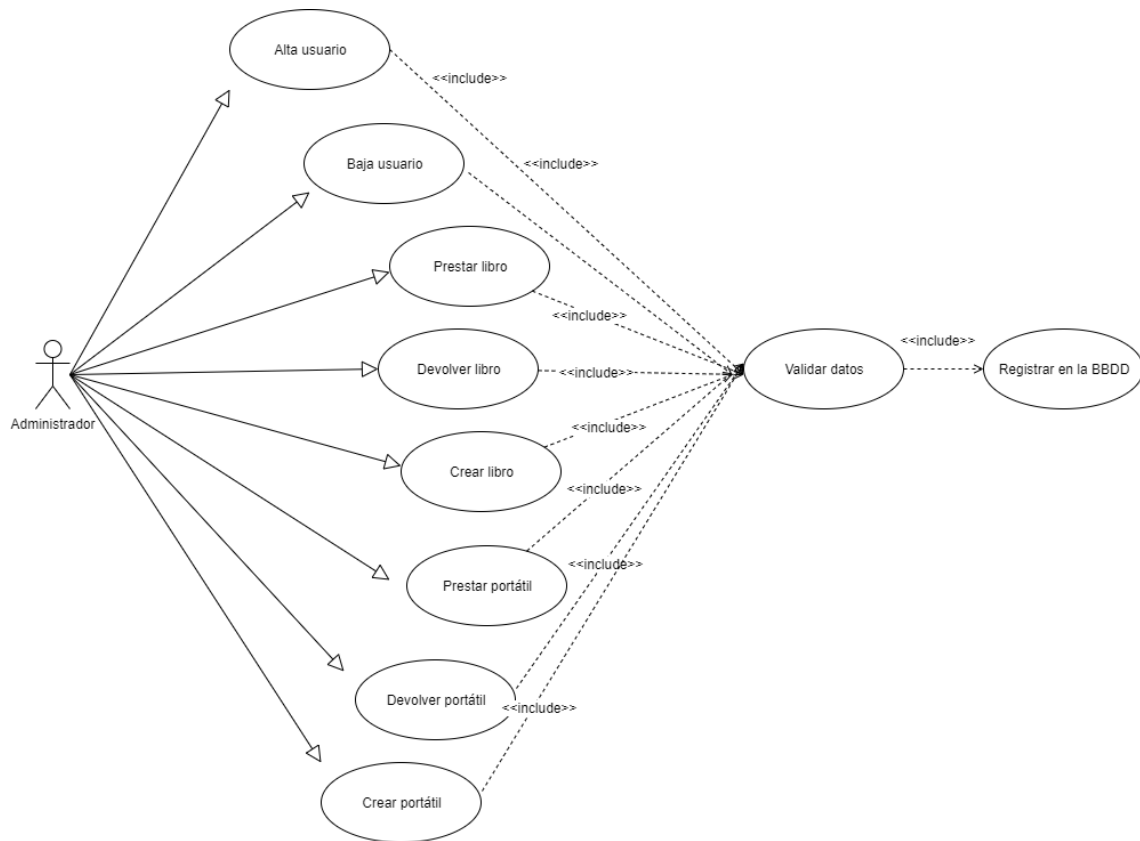
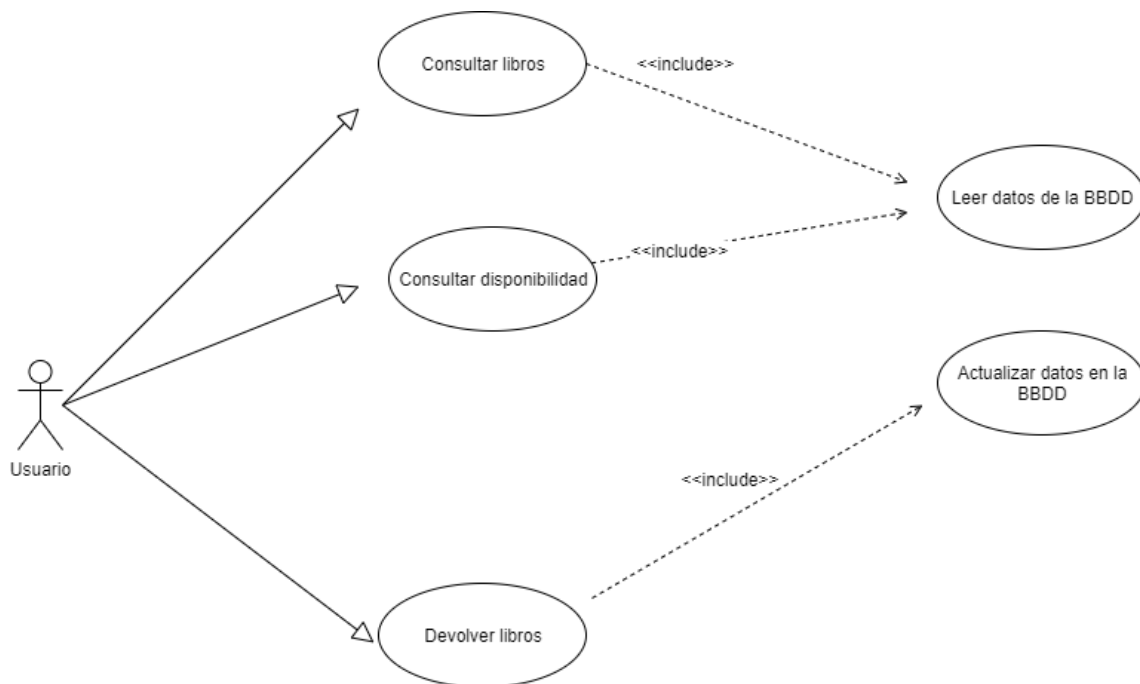


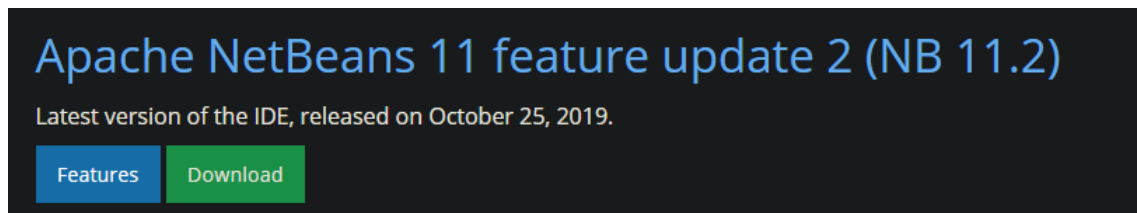
DIAGRAMA DE CASOS DE USO – USUARIO



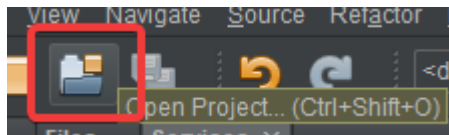
MANUAL DE INSTALACIÓN

Para el correcto funcionamiento del sistema se necesita tener instalado el sistema gestor de base de datos Derby en su modo “integrado” o “serverless”. Para ello, la opción más sencilla y de la que se va a hablar en este manual, es la instalación de Apache Netbeans.

Puesto que Derby es propiedad de Apache, viene incluido con la versión Apache de Netbeans. La descarga e instalación de Apache Netbeans se puede hacer desde la siguiente página web <https://netbeans.apache.org/download/>. Para el desarrollo de la práctica, se ha utilizado la versión 11.2, que es la última a 19/01/2020:

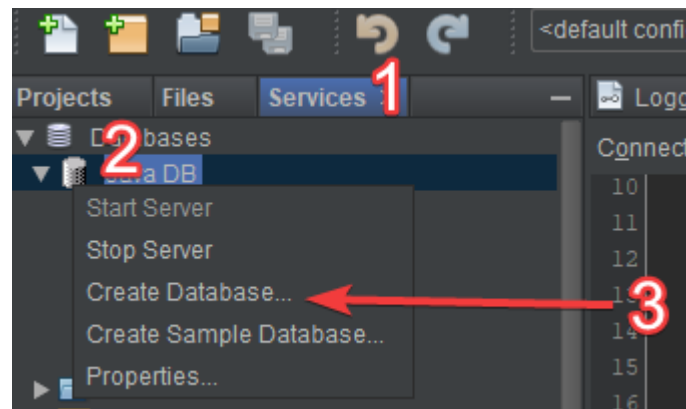


Una vez tenemos Apache Netbeans instalado, cargamos el proyecto Netbeans tal y como se entrega en el fichero. Para ello hacemos click en el botón de Open Project:

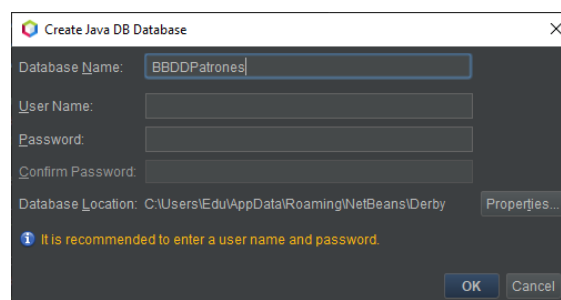


Navegamos hasta el proyecto y lo abrimos.

El siguiente paso es crear la base de datos. Para ello abrimos la pestaña de servicios de Apache Netbeans, abrimos Databases y, haciendo click derecho en Java DB, creamos una nueva base de datos:

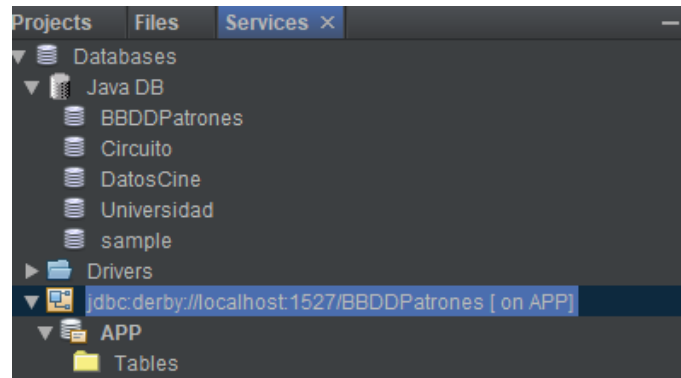


La base de datos se debe crear con nombre BBDDPatrones, sin especificar usuario ni contraseña.

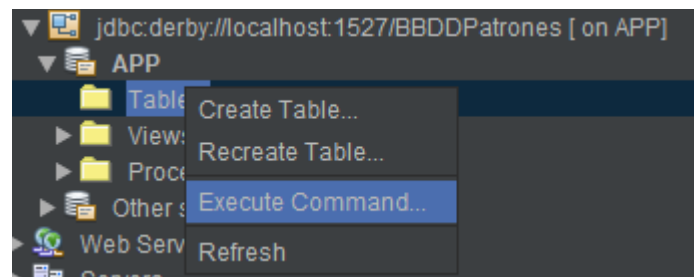


Una vez está creada la base de datos, hacemos click derecho sobre ella y nos conectamos. Puesto que no hemos especificado ni usuario ni contraseña, no hace falta rellenar nada del prompt que saldrá, simplemente hay que darle a siguiente.

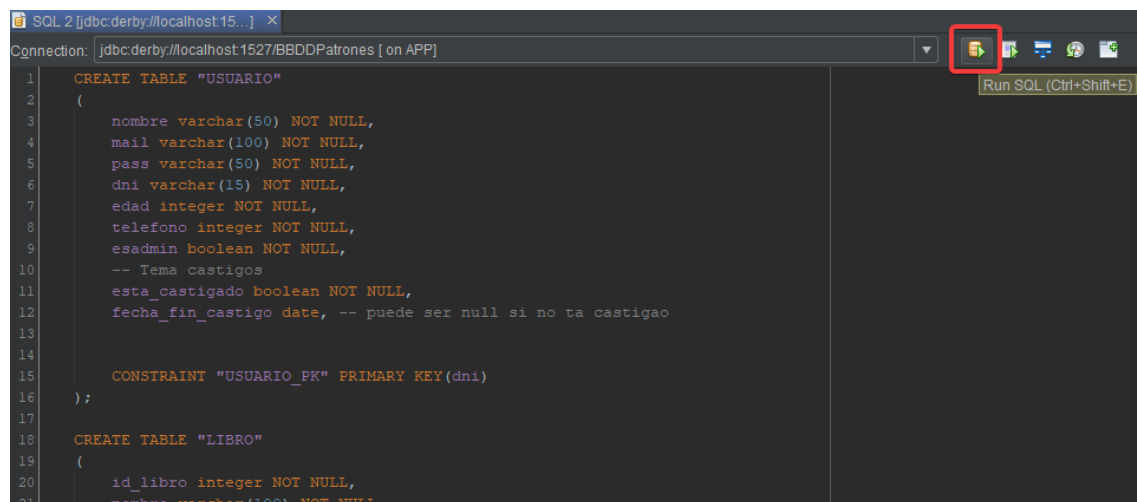
Una vez nos hayamos conectado, podremos ver la conexión aquí:



Hacemos click derecho sobre Tables y elegimos "Execute command...":



Se nos abrirá una ventana para lanzar consultas SQL. Copiamos los contenidos del archivo SQL de creación de tablas que se envía adjunto y le damos a Run SQL:



La consola de abajo debería devolver el siguiente mensaje:

```
Execution finished after 0.136 s, no errors occurred.
```

Una vez hemos hecho esto, repetimos con el archivo de datos de prueba. Los copiamos y pegamos en la ventana para lanzar consultas y la lanzamos. Obtendremos un mensaje similar.

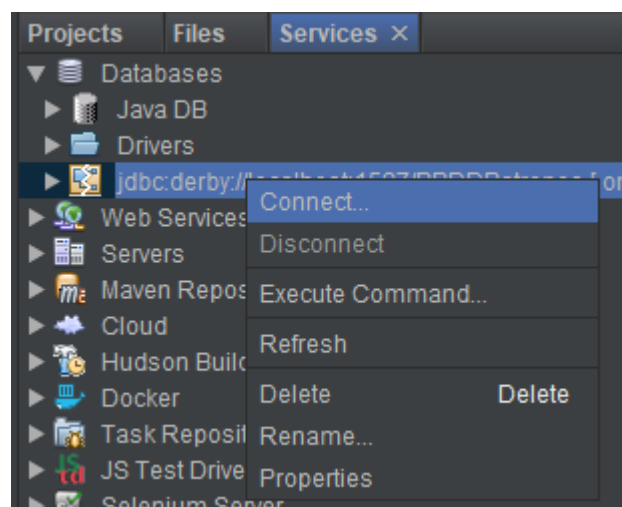
Si todo ha ido correctamente, el sistema está preparado para conectarse a la base de datos y funcionar con normalidad.

Para ejecutar la aplicación, simplemente ejecutamos el archivo Main que se encuentra dentro del paquete Main.

Si en algún momento nos saliese el siguiente error al intentar lanzar la aplicación:

```
ERROR [19/01/2020 20:12:22]: Error al abrir la conexión con la base de datos
java.sql.SQLException: java.net.ConnectException : Error connecting to server localhost on port 1,527 with message Connection refused: connect.
    at org.apache.derby.client.am.SQLExceptionFactory.getSQLException(Unknown Source)
    at org.apache.derby.client.am.SQLException.getSQLException(Unknown Source)
    at org.apache.derby.jdbc.ClientDriver.connect(Unknown Source)
    at java.sql.DriverManager.getConnection(DriverManager.java:664)
    at java.sql.DriverManager.getConnection(DriverManager.java:270)
```

Esto significa que Netbeans no ha lanzado el proceso con el servidor de la base de datos. Para solucionarlo simplemente volvemos al panel de servicios y reabrimos la conexión con nuestra base de datos:



Una vez hecho esto, si el servidor de base de datos ha conseguido lanzarse adecuadamente, podremos volver a lanzar la aplicación y conectarnos adecuadamente.

MANUAL DE USUARIO

Se detallan los siguientes manuales de usuario a modo de guías rápidas de uso.

CLIENTES DE LA BIBLIOTECA

Las tareas que pueden realizar los clientes de la aplicación son:

- **Acceso al catálogo de libros:** puede consultar los libros que tiene la biblioteca, ordenarlos, buscar en la base de datos, y comprobar su estado (disponible/prestado).
- **Devolución de libros:** los clientes de la biblioteca serán capaces de devolver libros desde la aplicación, simplemente eligiendo el libro que quieran devolver y dejándolo sobre la mesa de libros devueltos.
- **Comprobar la disponibilidad de portátiles:** el usuario será capaz de ver cuantos portátiles hay disponibles de cada tipo para hacer más fácil la gestión a la hora de pedirlos en el mostrador.

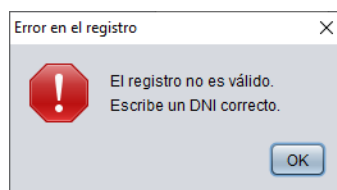
Nada más acceder a la aplicación, se nos presenta con el siguiente menú:



Desde aquí, podremos acceder a las ventanas de Login (si ya tenemos una cuenta en la biblioteca) y de Registro (si todavía no tenemos una cuenta en la biblioteca).

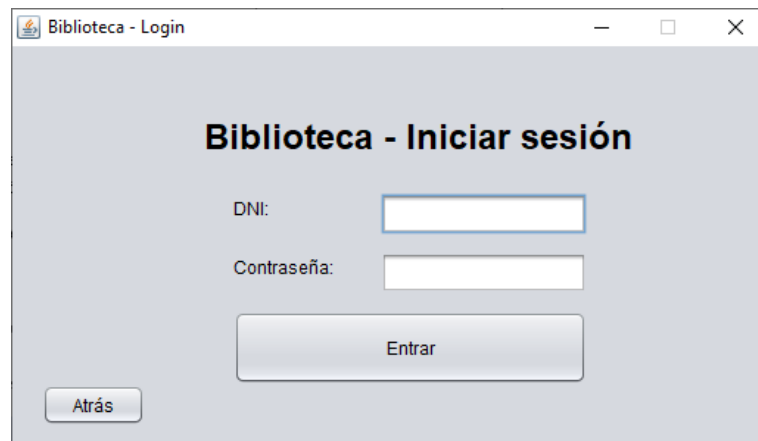
En caso de que queramos registrarnos, simplemente hacemos click en el botón de registro y se nos llevará al panel de recogida de datos. Este panel tiene la siguiente interfaz:

Simplemente tenemos que cumplimentar el formulario y hacer click en el botón de “Registrarse”. Si alguno de los campos que hayamos introducido no son válidos, la aplicación nos lo hará saber para que podamos corregirlo:



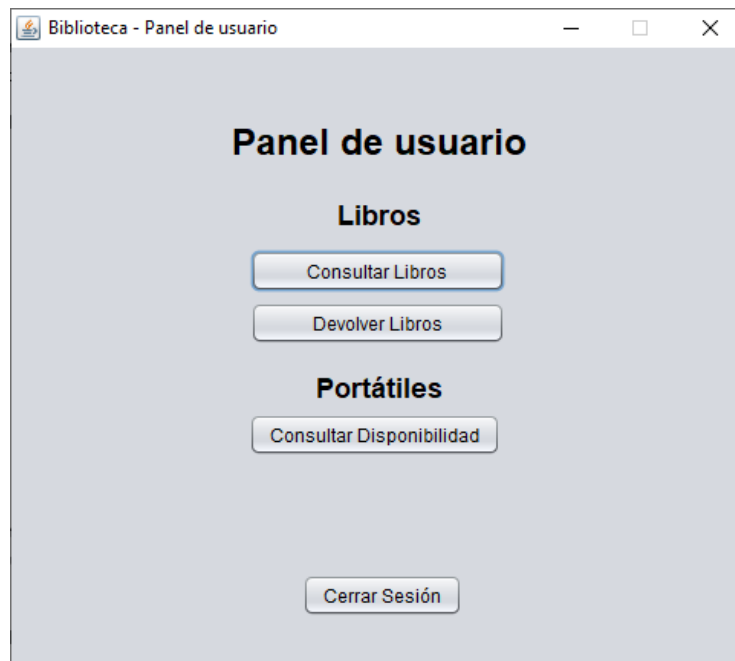
Una vez hayamos completado el registro, se nos redirige al menú inicial, donde podremos hacer click en el botón de Login para identificarnos.

El panel de Login tiene la siguiente interfaz:



Introducimos nuestros datos y le damos a entrar. Si todo está correcto, se nos redirigirá al panel de usuario. Si hay algún tipo de error durante la identificación, se nos notificará para corregirlo. (Para hacer pruebas se puede utilizar el usuario “51129104N” y la contraseña “contrasena123”).

El panel de usuario tiene la siguiente interfaz:



Dependiendo de lo que queramos hacer, pulsaremos un botón u otro. En caso de que ya hayamos terminado nuestra gestión con la biblioteca, simplemente pulsamos el botón de cerrar sesión y se nos devolverá al menú de inicio.

En caso de que queramos consultar libros, hacemos click sobre este botón. Se nos recibirá con la siguiente interfaz:

Biblioteca - Consulta de libros

Consultar libros en biblioteca

Introduzca el título / ISBN / autor del libro que busca

Buscar

Libros en la biblioteca

ID	Título	ISBN	Autor	Edición	Estado
1	Juego de tronos	9788408221937	George R. Martin	3	Prestado
2	Una España mejor	9788420438498	Mariano Rajoy	1	Disponible
3	Don Quijote	9788433980571	Belén Estéban	2	Disponible
4	Patrones Software	9788498389852	Salvador Otón	5	Disponible
5	Patrones Software	9788498389852	Salvador Otón	5	Disponible
6	Quien se ha lleva...	9788467058147	Francisco Roig B...	5	Disponible
7	Como ser bueno ...	9788491816614	Mario Conde	3	Disponible
8	El principito	9788408221937	Su autor	8	Disponible
9	Spiderman	9788408221937	Stan Lee	2	Disponible

Atrás

Podemos hacer click sobre los nombres de las columnas para ordenar los libros. Si hacemos click 1 vez se ordenarán en orden descendente, si hacemos click por segunda vez, se ordenará en orden ascendente.

Podemos también hacer búsquedas en esta tabla introduciendo texto en la barra de arriba y pulsando el botón de buscar:

Consultar libros en biblioteca

Introduzca el título / ISBN / autor del libro que busca

Buscar

Libros en la biblioteca

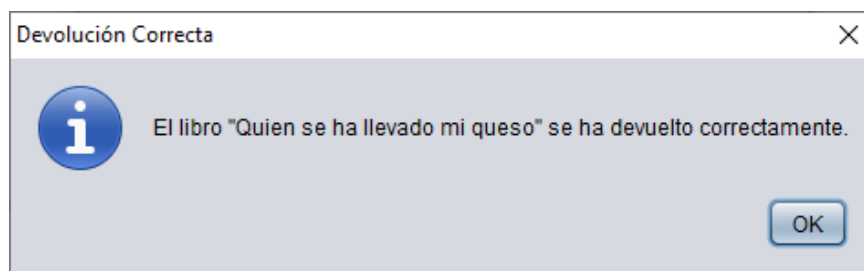
ID	Título	ISBN	Autor	Edición	Estado
3	Don Quijote	9788433980571	Belén Estéban	2	Disponible

En caso de que queramos volver a ver todos los libros, simplemente tenemos que limpiar la barra y volver a darle al botón de buscar.

Volviendo al panel de usuario, si hacemos click en “Devolver Libro”, veremos la siguiente interfaz:

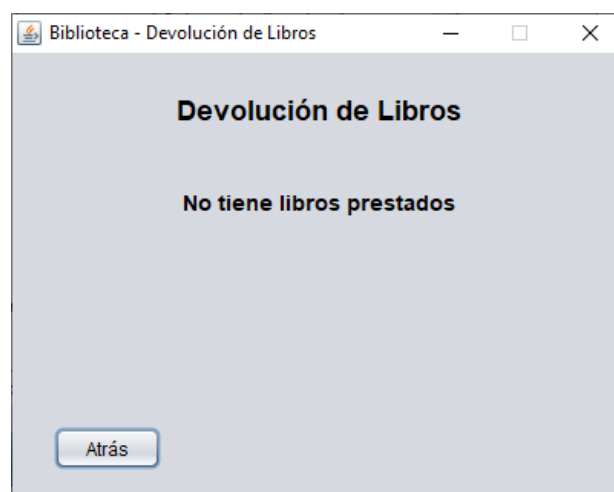


Simplemente tenemos que seleccionar el libro que queramos devolver de entre los libros que tengamos prestados y hacer click en “Devolver”.



Si vemos el mensaje anterior, la devolución se ha procesado adecuadamente y el cliente podrá dejar el libro en una de las mesas habilitadas para la devolución de libros.

Si no tenemos libros que devolver la aplicación nos lo hará saber:



Una vez hayamos terminado, hacemos click en atrás y volvemos al panel de usuario.

Si hacemos click en “Consultar Disponibilidad”, se nos llevará a una interfaz en la que podremos ver cuantos portátiles de cada tipo hay disponibles en la biblioteca:



En el caso de que no hubiese portátiles disponibles, la aplicación nos lo haría saber. En caso de que el usuario esté interesado en tomar prestado un portátil, tendría que ir al mostrador a pedirlo.

Una vez hayamos terminado con la aplicación de la biblioteca, es importante cerrar la sesión desde el panel de usuario.

ADMINISTRADORES DE LA BIBLIOTECA

El administrador puede llevar a cabo:

- **Gestión de usuarios:** alta y bajas de usuarios.
- **Gestión de libros:** préstamo, devolución y alta de libros.
- **Gestión de portátiles:** préstamo, devolución y alta de portátiles.

En primera instancia, el administrador debe escribir sus credenciales en la ventana de inicio de sesión. En los datos de prueba proporcionados, estas credenciales son “admin” para el usuario y la contraseña.



Una vez iniciada la sesión, se llegará al siguiente menú, llamado panel de administrador:

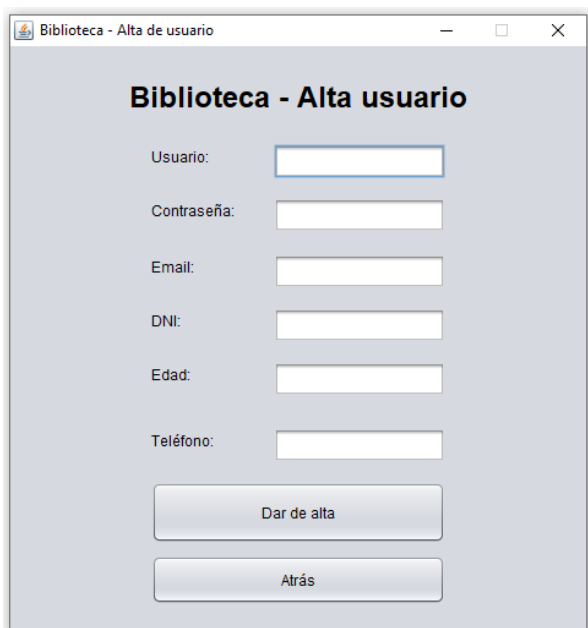


Desde este panel, se pueden llevar a cabo las operaciones previamente descritas.

GESTIÓN DE USUARIOS

ALTA USUARIOS

Para dar de alta a un usuario, desde el panel de administrador pulse sobre Alta usuarios, bajo Gestión Usuarios. Se abrirá la siguiente ventana, donde debe rellenar todos los campos para dar de alta a un usuario. Cabe reseñar que un usuario puede registrarse sin iniciar sesión él solito.



Biblioteca - Alta de usuario

Usuario:

Contraseña:

Email:

DNI:

Edad:

Teléfono:

Si alguno de los campos es incorrecto – DNI incorrecto, correo sin @, menor de edad, etc – saldrá un mensaje indicando que se corrija dicho campo:



Biblioteca - Alta usuario

Usuario:

Contraseña:

Email:

Error en el registro

El registro no es válido.
Escribe un email correcto.

BAJA USUARIOS

Para dar de baja a un usuario, desde el panel de administrador pulse sobre Baja usuario, bajo Gestión Usuarios. Se abrirá la siguiente ventana y se pedirá el DNI del usuario a dar de baja. Pulsando sobre cargar datos se pueden consultar los datos de dicho usuario sin darle de baja. Pulsando sobre dar de baja, se eliminará al usuario de la aplicación definitivamente. (¡Cuidado!)



Biblioteca - Baja de usuario

Usuario:

Email:

Edad:

Teléfono:

DNI:

GESTIÓN DE LIBROS

PRÉSTAMO DE LIBRO

Para ejecutar el préstamo de un libro, desde el panel de administrador pulse sobre Prestar Libro, bajo Gestión Libros. Coja el libro que le traiga el usuario que quiere tomarlo prestado. La ID del libro está escrita en el reverso, bajo el código de barras. Pídale el DNI al usuario también. Rellenando el campo de la ID del libro y pulsando sobre cargar datos, se pueden ver los datos del libro. Para llevar a cabo el préstamo, rellene el campo del DNI y pulse sobre el botón prestar. Si el usuario se encuentra castigado, saldrá una ventana emergente denotándolo y no dejará completar el préstamo. También detecta otros errores, como introducir DNI incorrecto, o que no esté en la base de datos, o que la id tampoco exista en la base de datos, etc.

Biblioteca - Préstamo libro

ID Libro:

DNI:

Nombre:

ISBN:

Autor:

Edición:

DEVOLUCIÓN DE UN LIBRO

Para llevar a cabo la devolución de un libro, desde el panel de administrador pulse sobre Devolver libro, bajo Gestión Libros. Pídale al usuario que le indique su DNI. Escribalo en el campo asociado del DNI y pulse sobre cargar datos. Se mostrarán todos los libros que tiene ese usuario. Seleccione el libro a devolver y pulse sobre el botón devolver. Puede devolver varios haciendo esto varias veces.

Si el usuario no tiene libros prestados, al pulsar sobre cargar se indicará con un mensaje.

Biblioteca - Devolución de libros

Devolución de Libros

DNI del usuario:

Escoja el libro que desea devolver

Biblioteca - Devolución de libros

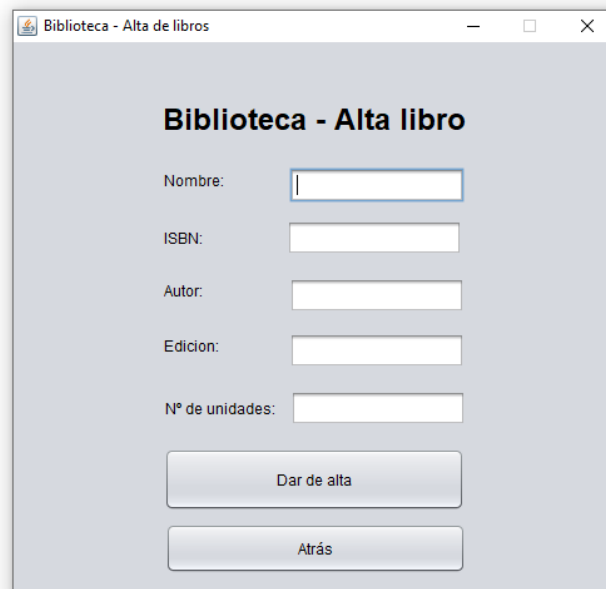
Devolución de Libros

DNI del usuario:

No tiene libros prestados

ALTA DE LIBRO

Para llevar a cabo el alta de uno o varios libros, desde el panel de administrador pulse sobre Crear libro, bajo Gestión Libros. Rellene las características del libro en los campos asociados. Introduzca el nº de unidades a dar de alta de ese libro, y pulse sobre el botón de Dar de alta. Si introduce algún dato incorrecto, se mostrará un mensaje indicándolo.

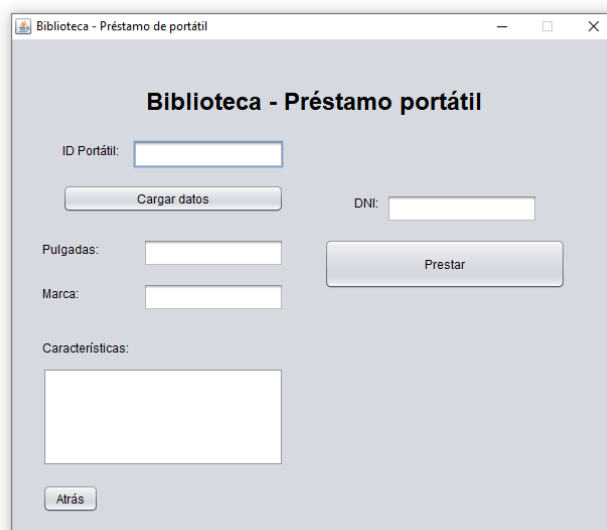


The screenshot shows a web application window titled "Biblioteca - Alta de libros". The main heading is "Biblioteca - Alta libro". Below the heading, there are five input fields with labels: "Nombre:", "ISBN:", "Autor:", "Edición:", and "Nº de unidades:". Each field is a simple text box. At the bottom of the form, there are two buttons: "Dar de alta" and "Atrás".

GESTIÓN DE PORTÁTILES

PRÉSTAMO DE PORTÁTILES

Para llevar a cabo el préstamo de un portátil, desde el panel de administrador pulse sobre Prestar portátil, bajo Gestión portátiles. Tome el portátil del tipo del que el usuario quiera, y mire su id. La id se encuentra en una etiqueta pegada en la tapa. Pídale al usuario su DNI. Introduzca el id y el DNI en los campos asociados. Si pulsa sobre cargar datos, se pueden ver las características de ese portátil en concreto. Para efectuar el préstamo, pulse sobre el botón de prestar. Si hay algún error o problema, como que el usuario esté castigado, se mostrará un mensaje advirtiéndolo.



The screenshot shows a web application window titled "Biblioteca - Préstamo de portátil". The main heading is "Biblioteca - Préstamo portátil". Below the heading, there are two input fields: "ID Portátil:" and "DNI:". Between these fields is a button labeled "Cargar datos". Below the "ID Portátil:" field, there are two more input fields: "Pulgadas:" and "Marca:". To the right of these fields is a button labeled "Prestar". At the bottom left, there is a large text area labeled "Características:". At the very bottom left, there is a button labeled "Atrás".

DEVOLUCIÓN DE PORTÁTILES

Para llevar a cabo la devolución de un portátil, desde el panel de administrador pulse sobre Devolver portátil, bajo Gestión portátiles. Tome el portátil que el usuario le devuelva, y mire su id. Se encuentra en una pegatina sobre la tapa. Introduzca la ID del portátil en el campo asociado.


Si el usuario se ha pasado de plazo, se mostrará un mensaje advirtiéndolo y diciendo cuantos días estará castigado sin poder tomar prestados libros ni portátiles.



The screenshot shows a web application window titled "Biblioteca - Devolución portátil". The main heading is "Biblioteca - Devolución portátil". Below the heading, there is a label "ID Portátil:" followed by a text input field. Underneath the input field is a button labeled "Devolver". In the bottom left corner, there is a button labeled "Atrás".

ALTA PORTÁTILES

Para llevar a cabo el alta de uno o varios portátiles, desde el panel de administrador pulse sobre Devolver portátil, bajo Gestión portátiles. Se mostrará la siguiente ventana:



The screenshot shows a web application window titled "Biblioteca - Alta de portátil". The main heading is "Biblioteca - Registrar portátil". Below the heading, there is a label "Tipo:" followed by a dropdown menu currently showing "Macbook". Underneath the dropdown is a label "Nº de unidades:" followed by a text input field. Below the input field is a button labeled "Dar de alta". In the bottom left corner, there is a button labeled "Atrás".

Elija, desde el desplegable, el tipo de portátil que va a dar de alta. Introduzca el nº de unidades que va a registrar de ese tipo, y pulse sobre dar de alta.

SALIR

Todas las interfaces tienen un botón atrás que le llevará al panel anterior. Si pulsa sobre salir desde el panel de administrador, cerrará sesión y volverá a la ventana de inicio de sesión.