

Universidad Tecnológica Nacional

FRRO

TP JAVA

AÑO: 2021 - Comisión 308

Alumnos:

<u>Nombre y Apellido</u>	<u>Legajo</u>
1- Franco Giangiordano	46802
2- Gonzalo Turconi	46730

Docentes:

- Adrián Meca
- Mario Bressano

ÍNDICE

Capturas de pantalla:	8
Inicio	8
Contacto	9
Login	9
Registrarse	9
Menú administrador	10
Modificar cuenta	10
Eliminar cuenta	11
Cerrar sesión	11
Listado clientes	12
Listado localidades	12
Listado autores	13
Listado categorías	13
Listado editoriales	14
Listado Pedidos	14
Listado Libros	15
Detalle libro	16
Búsqueda de localidad	16
Búsqueda de cliente	17
Búsqueda de autor	17
Búsqueda de categoría	18
Búsqueda de editorial	18
Búsqueda de libro	19
Añadir localidad	19
Añadir Autor	20
Añadir categoría	20
Añadir editorial	21
Añadir libro	21
Modificar localidad	22
Menu cliente	24
Modificar cuenta	24
Eliminar cuenta	25
Cerrar sesión	25
Listado autores	26
Listado categorías	26
Listado editoriales	27
Detalle libro	29
Comentarios + calificación	29
Comentarios + calificación	30
Diseño adaptable a varias resoluciones de pantalla:	31
Inicio (ventana pequeña)	31
Contacto (ventana pequeña)	33
Menú (ventana pequeña)	34

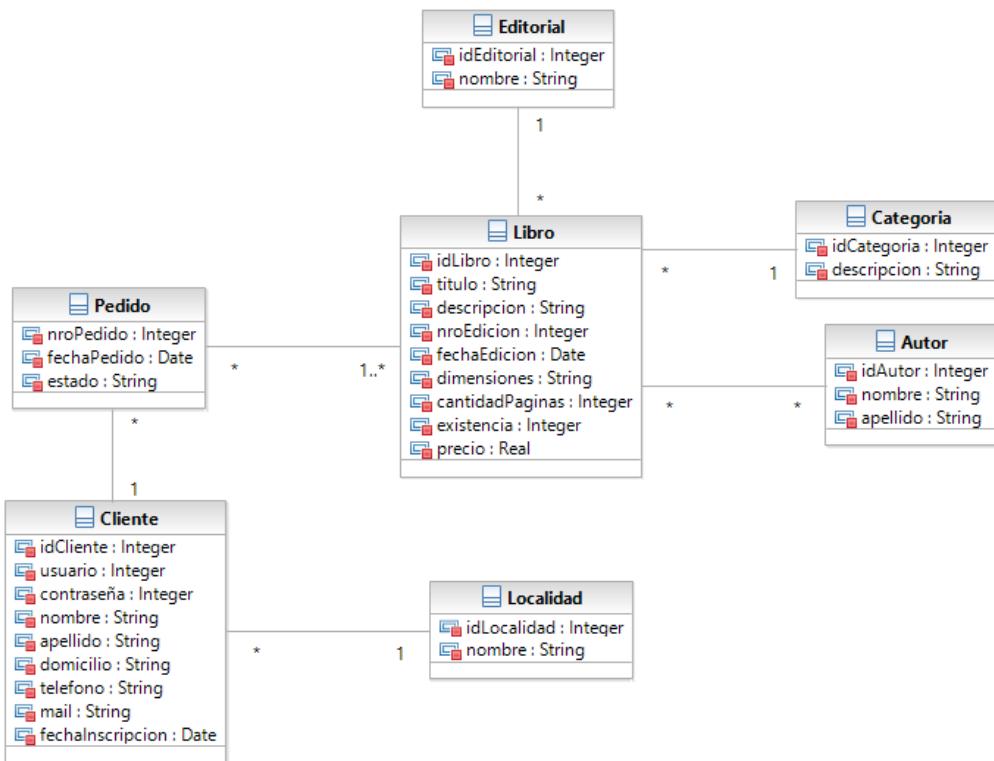
Listado Clientes (ventana pequeña)	35
Listado localidades (ventana pequeña)	37
Listado autores (ventana pequeña)	39
Listado categorías (ventana pequeña)	41
Listado editoriales (ventana pequeña)	43
Listado libros (ventana pequeña)	45
Listado pedidos (ventana pequeña)	46
Detalle libro (ventana pequeña)	47

Enunciado general:

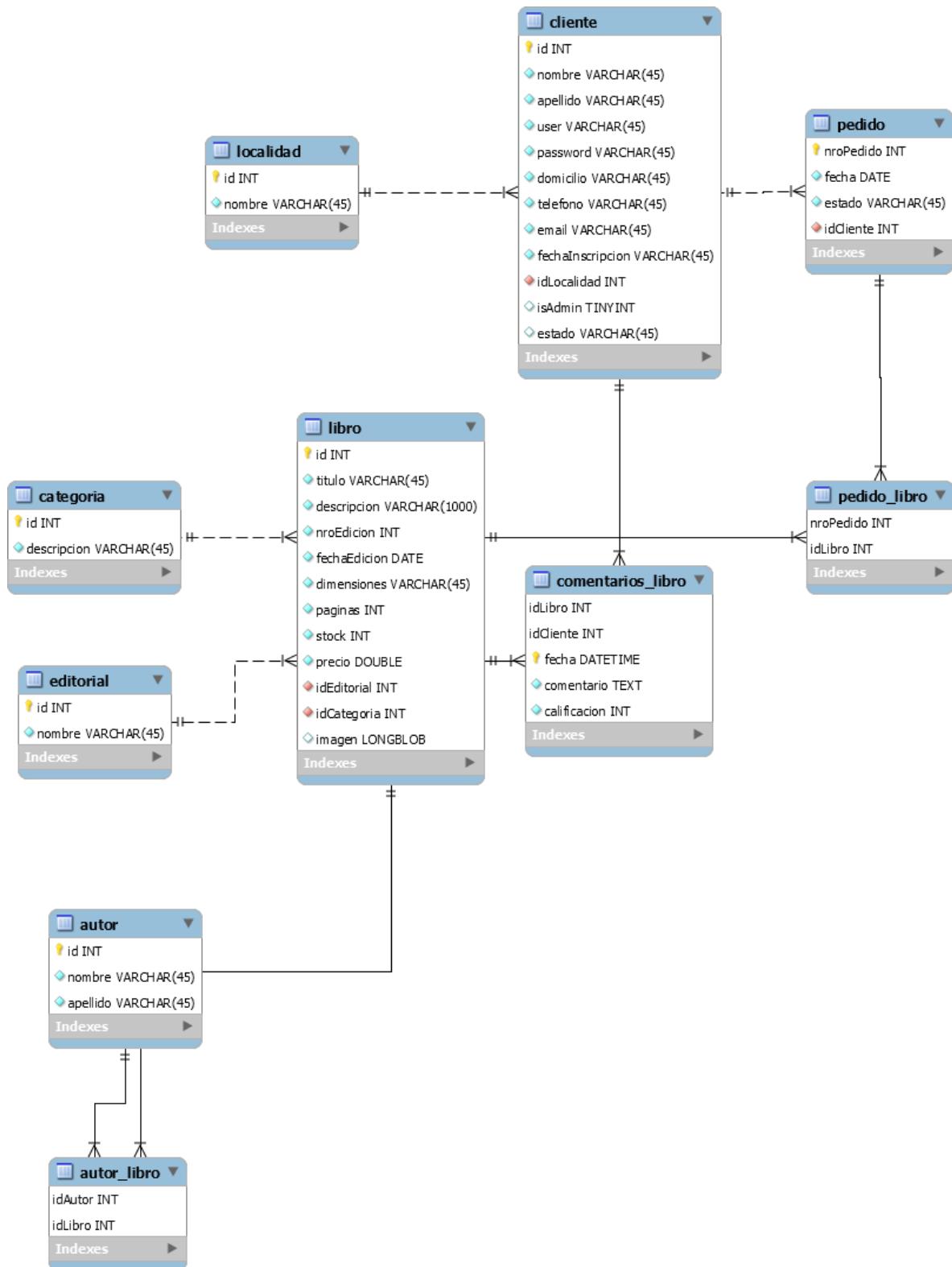
El sistema consiste en un aplicación web para gestión de pedidos de libros y de clientes.

Existen dos usuarios: administrador y cliente, ambos registrados

Diagrama de Clases:



Modelo de Datos:



Regularidad

Requerimiento	Cant. máx. 3 o 4 integ	Detalle/Listado de casos incluidos
ABMC Simple	1 x integrante	ABMC de Categoría, Autor, Editorial
ABMC Dependiente	2	ABMC de Libro y de Cliente
CU NO-ABMC	2	Préstamo de Libro, Devolución de Libro
Listado Simple	3	Listado de libros, Listado de Clientes, Listado de Reservas
Listado Complejo	-----	-----

Aprobación Directa

Requerimiento	Cant. máx. 3 o 4 integ	Detalle/Listado de casos incluidos
ABMC Todos	1 x integrante	Categoría, Autor, Editorial, Localidad, Libro, Pedido, Cliente
CU Complejos	2	Reserva, préstamo y devolución de Libro, Calificación/Reseña de libros
Listado Complejo	3	Listado de libros por estado de clientes por estado
Nivel de acceso	-----	Administrador-Cliente
Requerimiento extra		Manejo de archivos, Envío de emails

Casos de Uso

Casos de Uso Resumen re-estructurado

Código y Nombre del CASO DE USO: CUR01 Reservar Libro

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Resumen	reestructurado	sistema	negra	semántica	real

Meta del CASO DE USO: Reservar libro

Actor Primario: Cliente

Otros: <vacío>

PRECONDICIONES (de negocio):

- Existen clientes que deseen reservar un libro.
- Existen libros para reservar

PRECONDICIONES (de sistema):

- Existen libros cargados
- Cliente está logueado

DISPARADOR: Cliente desea reservar un libro

CAMINO BÁSICO:

1. Cliente selecciona la opción listado de libros. Sistema informa **invocando al CUU 1.1 Listado de libros**
2. Cliente elige un libro y solicita la reserva del mismo. Sistema registra reserva invocando al **CUU 1.2 Registrar reserva**

CAMINOS ALTERNATIVOS:

1.a<posterior>: Cliente consulta detalles del libro

1.a.1 Sistema muestra detalles del libro

2.a<reemplaza>: El usuario está inhabilitado

2.a.1 Sistema informa situación. FIN C.U

2.b<reemplaza>: Cliente ya agotó el cupo de reservas

2.b.1 Sistema informa situación. FIN C.U

2.c<reemplaza>: El libro no tiene stock

2.c.1 Sistema informa situación. FIN C.U

2.d<posterior>: Cliente cancela la reserva

2.d.1 Sistema registra la cancelación de la reserva **invocando al CUU 1.3 Cancelar Reserva**

POSTCONDICIONES (de sistema):

Universidad Tecnológica Nacional

Éxito: Se registró la reserva

Fracaso: No se registró la reserva porque el libro no tenía stock y/o el cliente está inhabilitado y/o el cliente agotó el cupo

Éxito alternativo: Se registró la reserva consultando el detalle del libro

Reglas de Negocio relacionadas con el casos de uso:

Paso(s)	Regla de negocio
RN1	El cliente tiene un cupo máximo de 5 libros para reservar
RN2	Solo pueden realizar reservas aquellos usuarios que estén habilitados
RN3	Desde el momento de la reserva, el cliente tiene un plazo máximo de 3 días para retirarlo
RN4	En caso de que el cliente no retire su libro o bien no lo devuelva a término quedará temporalmente inhabilitado

Código y Nombre del CASO DE USO: CUR 02 Confirmar préstamo

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Resumen	reestructurado	sistema	negra	semántica	real

Meta del CASO DE USO: Confirmar pedido

Actor Primario: Administrador

Otros: <vacío>

PRECONDICIONES (de negocio):

- Existen clientes que se presentan a retirar el libro/pedido

PRECONDICIONES (de sistema):

- Existen pedidos registrado en estado “Reservado”
- El administrador está logueado

DISPARADOR: Cliente se presenta a retirar el libro

CAMINO BÁSICO:

1 - Cliente se presenta a retirar el libro e informe nombre, apellido y la fecha del pedido. Administrador solicita listado de pedidos. Sistema muestra pedidos **invocando al CUU 2.1 Listar pedidos**

2 - Administrador solicita pedidos reservados. Sistema filtra los pedidos **invocando al CUU 2.2 Filtrar pedidos.**

3- Administrador confirma el pedido, Sistema registra la confirmación del pedido **invocando al CUU 2.3 Confirmar pedido**

Universidad Tecnológica Nacional

CAMINOS ALTERNATIVOS:

1.a<reemplaza> Cliente se presenta fuera de término

1.a.1 Administrador rechaza el pedido. Sistema actualiza el estado del pedido a “Cancelado”
.FIN C.U

2.a<posterior>: No existe el pedido del cliente

2.a.1 Administrador informa situación al cliente. FIN C.U

POSTCONDICIONES (de sistema):

Éxito: Se confirmó el préstamo del libro

Fracaso: Se registró el rechazo del pedido por presentarse fuera de término (RN1) o no se encontró el pedido del cliente.

Éxito alternativo:

Reglas de Negocio relacionadas con el casos de uso:

Paso(s)	Regla de negocio
RN1	Desde el momento de la reserva, el cliente tiene un plazo máximo de 3 días para retirarlo
RN2	En caso de que el cliente no retire su libro o bien no lo devuelva a término quedará temporalmente inhabilitado

Código y Nombre del CASO DE USO: CUR 03 Registrar devolución

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Resumen	reestructurado	sistema	negra	semántica	real

Meta del CASO DE USO: Registrar devolución

Actor Primario: Administrador

Otros: <vacío>

PRECONDICIONES (de negocio):

- Existen clientes que se presentan a devolver el libro

PRECONDICIONES (de sistema):

- Existen pedidos registrado en estado “En curso”
- El administrador está logueado

DISPARADOR: Cliente se presenta a devolver el libro

CAMINO BÁSICO:

- 1 - Cliente se presenta a devolver el libro e informe nombre, apellido y la fecha del pedido.
Administrador solicita listado de pedidos. Sistema muestra pedidos **invocando al CUU 2.1 Listar pedidos**
- 2 - Administrador solicita pedidos en curso. Sistema filtra los pedidos **invocando al CUU 2.2 Filtrar pedidos.**
- 3- Administrador confirma la finalización el pedido, Sistema registra la finalización del pedido **invocando al CUU 3.1 Finalizar pedido**

CAMINOS ALTERNATIVOS:

3.a<reemplaza> Cliente se presentó fuera de término

1.a.1 Administrador confirma la finalización del pedido e inhabilita al usuario. Sistema actualiza el estado del pedido a “Finalizado” y se actualiza el estado del usuario a “Inhabilitado” FIN C.U

POSTCONDICIONES (de sistema):

Éxito: Se actualizó el estado del pedido a “Finalizado”

Fracaso: <vacío>

Éxito alternativo: Se actualizó el estado del pedido a “Finalizado” y se inhabilitó al cliente.

Casos de Uso de Usuario

Código y Nombre del CASO DE USO: CUU 1.1 Listado de libros

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Mostrar libros disponibles

Actor Primario: Cliente

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados

El cliente está logueado

DISPARADOR: Cliente elige opción listar libro

CAMINO BÁSICO:

- 1- Cuando el cliente seleccionó la opción listar libros. Sistema muestra listado de libros disponibles.

CAMINOS ALTERNATIVOS:

1.a<posterior>: Cliente consulta detalles del libro

1.a.1 Sistema muestra detalles del libro

POSTCONDICIONES (de sistema):

Éxito: Cliente obtuvo el listado de libros

Fracaso: <vacío>

Éxito alternativo: <vacío>

Código y Nombre del CASO DE USO: CUU 1.2 Registrar reserva**Dimensiones de clasificación:**

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Reservar un libro

Actor Primario: Cliente

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados

El cliente está logueado

DISPARADOR: Cliente elige opción reservar libro

CAMINO BÁSICO:

1- Cliente selecciona reservar libro. Sistema valida que el cliente se encuentre habilitado, valida cupo disponible de libros reservados y que el libro tenga stock.

2- Sistema registra un nuevo pedido del cliente y añade la reserva del libro.

CAMINOS ALTERNATIVOS:**1.a<reemplaza>: El usuario está inhabilitado**

1.a.1 Sistema informa situación. FIN C.U

1.b<reemplaza>: Cliente ya agotó el cupo de reservas

1.b.1 Sistema informa situación. FIN C.U

1.c<reemplaza>: El libro no tiene stock

1.c.1 Sistema informa situación. FIN C.U

1.d<posterior>: Cliente cancela la reserva

1.d.1 Sistema registra la cancelación de la reserva.

2.a<reemplaza>: El cliente ya había hecho un pedido en ese día

2.a.1 Sistema añade la reserva del libro al pedido encontrado

POSTCONDICIONES (de sistema):

Éxito: Se registró la reserva del libro

Fracaso: No se registró la reserva del libro porque el usuario estaba inhabilitado o porque no tiene cupo disponible de reserva o porque el libro no tiene stock

Éxito alternativo: <vacío>

Código y Nombre del CASO DE USO: CUU 1.3 Cancelar reserva

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Cancelar Reserva

Actor Primario: Cliente

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados

El cliente está logueado

Existen pedidos registrados en estado “Reservado”

DISPARADOR: Cliente elige opción cancelar libro

CAMINO BÁSICO:

1- Cliente selecciona cancelar libro. Sistema actualiza el estado del pedido a “Cancelado”

CAMINOS ALTERNATIVOS:

1.a<reemplaza>: El pedido tiene más de un libro reservado

1.a.1 Sistema elimina el libro del pedido. FIN C.U

POSTCONDICIONES (de sistema):

Éxito: Se registró la cancelación del libro pedido y se actualizó el estado del mismo

Fracaso:

Éxito alternativo: Se eliminó el libro del pedido

Código y Nombre del CASO DE USO: CUU 2.1 Listar Pedidos

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Listar pedidos

Actor Primario: Administrador

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados
El administrador está logueado
Existen pedidos registrados

DISPARADOR: Administrador elige opción listar pedidos

CAMINO BÁSICO:

1- Cuando el administrador selecciona listar pedidos. Sistema consulta y muestra pedidos disponibles.

CAMINOS ALTERNATIVOS:

<vacío>

POSTCONDICIONES (de sistema):

Éxito: Se listaron los pedidos disponibles

Fracaso: <vacío>

Éxito alternativo: <vacío>

Código y Nombre del CASO DE USO: CUU 2.2 Filtrar Pedidos

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Filtrar pedidos

Actor Primario: Administrador

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados
El administrador está logueado

DISPARADOR: Administrador elige opción Pedidos Reservados

Universidad Tecnológica Nacional

CAMINO BÁSICO:

- 1- Cuando el administrador selecciona pedidos reservados. Sistema consulta y muestra pedidos en estado reservado.

CAMINOS ALTERNATIVOS:

1.a <reemplaza> Filtrar pedidos en curso

- 1.a.1 Cuando el administrador selecciona pedidos en curso. Sistema consulta y muestra pedidos en estado en curso.

1.b <reemplaza> Filtrar pedidos cancelados

- 1.b.1 Cuando el administrador selecciona pedidos cancelados. Sistema consulta y muestra pedidos en estado en cancelado.

1.c <reemplaza> Filtrar pedidos finalizados

- 1.c.1 Cuando el administrador selecciona pedidos finalizados. Sistema consulta y muestra pedidos en estado finalizado.

POSTCONDICIONES (de sistema):

Éxito: Se listaron los pedidos disponibles en estado reservado

Fracaso: <vacío>

Éxito alternativo: Se listaron los pedidos disponibles en estado en curso y/o cancelados y/o finalizados

Código y Nombre del CASO DE USO: CUU 2.3 Confirmar Pedido

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Confirmar pedido

Actor Primario: Administrador

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados

El administrador está logueado

Existen pedidos en estado reservado

DISPARADOR: Administrador elige opción Pedidos Reservados

CAMINO BÁSICO:

- 1- Cuando el administrador selecciona confirmar pedido. Sistema actualiza el estado del pedido a “En Curso”

CAMINOS ALTERNATIVOS:

1.a<reemplaza> Administrador selecciona rechazar pedido

- 1.a.1 Administrador rechaza el pedido. Sistema actualiza el estado del pedido a “Cancelado”
.FIN C.U

POSTCONDICIONES (de sistema):

Éxito: Se registró la confirmación del préstamo

Fracaso: Se registró el rechazo del pedido

Éxito alternativo: <vacío>

Código y Nombre del CASO DE USO: CUU 3.1 Finalizar Pedido

Dimensiones de clasificación:

Nivel	Estructura	Alcance	Caja	Instanciación	Interacción
Usuario	Sin-estructurar	Sistema	Negra	Real	Semántica

Meta del CASO DE USO: Finalizar pedido

Actor Primario: Administrador

Otros:

PRECONDICIONES (de sistema):

Existen libros registrados

El administrador está logueado

Existen pedidos en estado en curso

DISPARADOR: Administrador elige opción Pedidos en curso

CAMINO BÁSICO:

- 1- Cuando el administrador selecciona finalizar pedido. Sistema actualiza el estado del pedido a “Finalizado”

CAMINOS ALTERNATIVOS:

<vacío>

POSTCONDICIONES (de sistema):

Éxito: Se registró la finalización del pedido

Fracaso: <vacío>

Éxito alternativo: <vacío>

Capturas de pantalla:

Inicio

Biblioteca Entre hojas

Iniciar sesión

La mejor **colección de libros** para lectores apasionados

Le brindamos la mejor manera de vivir la vida a través de las palabras



Nuestros Servicios

Elijanos y te sentirás satisfecho. ¿Cerrarias los ojos a tantas historias?


Bibliotecarios Expertos

Excelente atención al cliente

Responsables de la catalogación, clasificación, indexación y exposición de los materiales, capaces de ayudarte y guarte en lo que necesites


Buenos libros, buenos ratos

Numerosa Variedad

La diversidad de libros es tan grande que puede resultar difícil contarlos y clasificarlos, sin embargo, contamos con categorías específicas para encontrar lo que buscas

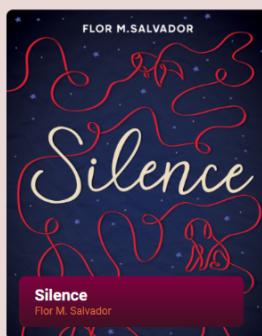

Sólo hay un bien: el conocimiento

Pedidos las 24 horas

Pedidos de documentos, interbibliotecarios, especiales para entidades y además contamos con un buzón 24 horas de devolución de documentos

Libros Reconocidos

Un buen libro es más que una piedra preciosa



Contacto

Biblioteca Entre hojas

¡ Contactate con nosotros !

¡Gracias por su interés! Complete los datos para contactarnos.
Estamos aquí para lo que necesite.

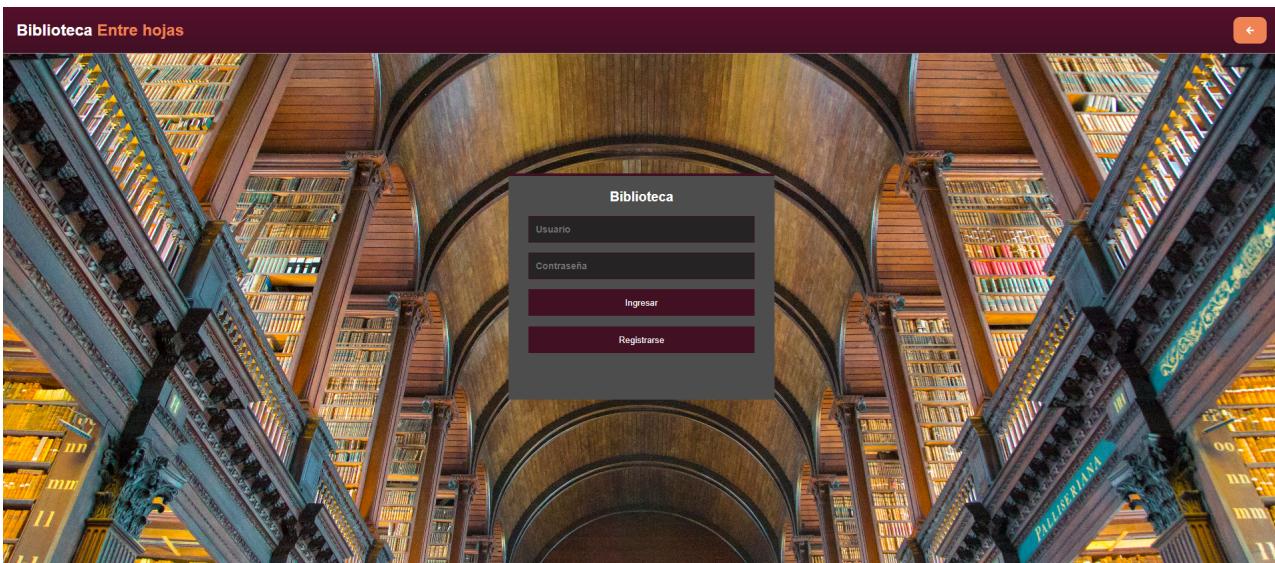
Nombre:

Apellido:

Correo electrónico:

Mensaje:

Login



Registrarse

Biblioteca Entre hojas

Registrate

Nombre:

Apellido:

Domicilio:

Teléfono:

Email:

Localidad:

Usuario:

Contraseña:

Menú administrador

Biblioteca Entre hojas

| Bienvenido Admin Roberto Gutierrez |

Cuenta Listados

VIVÍ LA MAGIA DE LOS LIBROS ILUSTRADOS

Biblioteca Entre Hojas bibliotecaentrehojas@gmail.com Galería Stigliano Derechos Reservados

Modificar cuenta

Biblioteca Entre hojas

Ingrese los nuevos datos de su cuenta

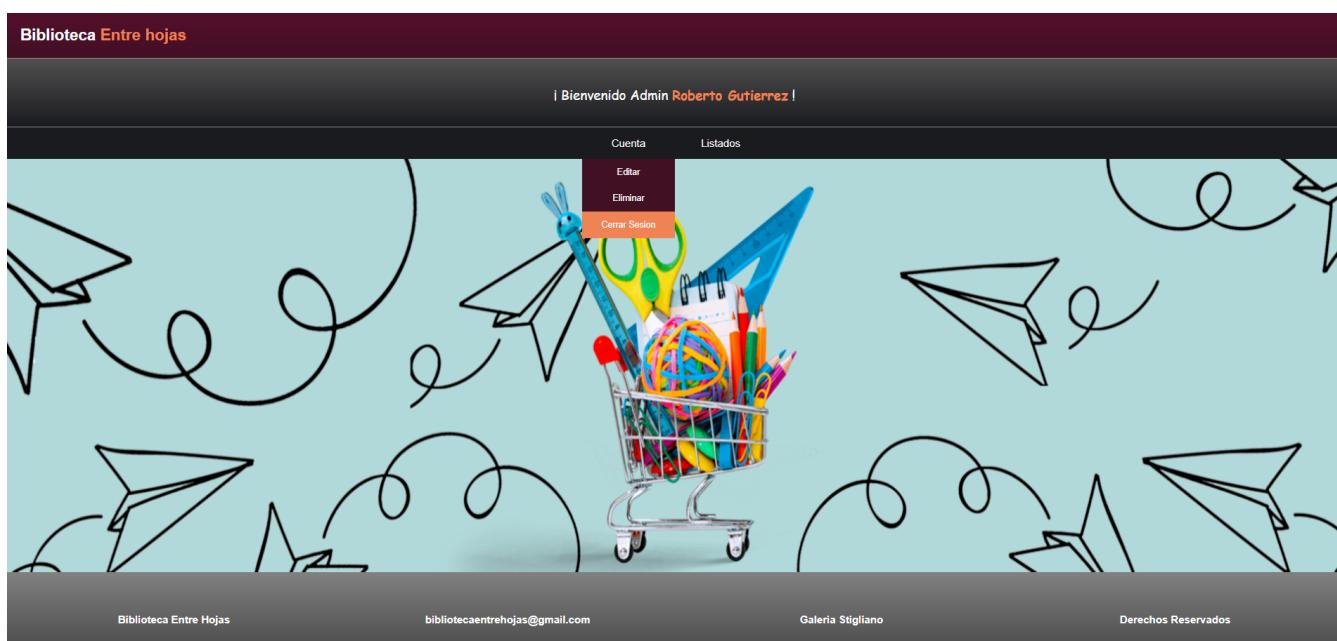
Roberto
Gutierrez
Juan B Justo 2560
3464 562147
elRobert@gmail.com
Pujato
robertito
.....

Actualizar datos

Eliminar cuenta



Cerrar sesión



Listado clientes

Biblioteca Entre hojas

Filtrar por :

Solo Suspendidos | Solo Habilitados | Listar Todos |

Listado de Clientes

NUMERO	NOMBRE	APELLIDO	USUARIO	DOMICILIO	TELEFONO	EMAIL	FECHA INSCRIPCION	LOCALIDAD	ESTADO	ACCION
2	Julieta	Carracedo	julicarrace2	Belgrano 526	341 694712	juliCarracedo@gmail.com	2019-05-08	Cañada de Gomez	habilitado	<button>Suspender Cliente</button>
3	Ernesto	Valverde	elErnest	Av.Centenario 433	341156799	ernestito@gmail.com	2021-12-05	Rosario	habilitado	<button>Suspender Cliente</button>
4	Franco	Giangiordano	fran	Belgrano 123	3464516658	frangiordano@gmail.com	2021-12-10	Casilda	habilitado	<button>Suspender Cliente</button>



Listado localidades

Biblioteca Entre hojas

Listado de Localidades

NUMERO	NOMBRE	ELIMINAR	MODIFICAR
1	Casilda		
2	Pujato		
3	Cañada de Gomez		
4	Rosario		



Listado autores

Biblioteca Entre hojas

Listado de autores

NUMERO	NOMBRE	APELLIDO	ELIMINAR	MODIFICAR
15	Viktor	Franki		
16	Anita	Brookner		
17	Agota	Kristof		
18	Graham	Greene		
19	Claudia	Piñeiro		
20	Stephen	King		
21	Pierre	Lemaitre		
22	René	Goscinny		



BUSCAR

Ingrese id

Buscar Autor



AÑADIR

Añadir Autor

Listado categorías

Biblioteca Entre hojas

Listado de Categorías

NUMERO	DESCRIPCION	ELIMINAR	MODIFICAR
13	Psicología		
14	Ficción		
15	Policial		
16	Terror		
17	Infantil		



BUSCAR

Ingrese id

Buscar Categoría



AÑADIR

Añadir Categoría

Listado editoriales

Biblioteca Entre hojas

Listado de Editoriales

NUMERO	NOMBRE	ELIMINAR	MODIFICAR
5	Herder	☒	✍
6	Libros del asteroide	☒	✍
7	Edhasa	☒	✍
8	Debolsillo	☒	✍
9	Alfaguara	☒	✍
10	Libros del Zorzal	☒	✍



BUSCAR

Ingrese id

Buscar Editorial



AÑADIR

Añadir Editorial

Listado Pedidos

Biblioteca Entre hojas

Filtrar por :

Solo Reservados Solo Cancelados Solo En Curso Solo Finalizados

Listado de pedidos

NUMERO	FECHA	ESTADO	ANULAR PEDIDO	IDCLIENTE	IDLIBRO	TITULO
23	2022-01-22	cancelado	2		47	Vidas breves
					48	Ayer

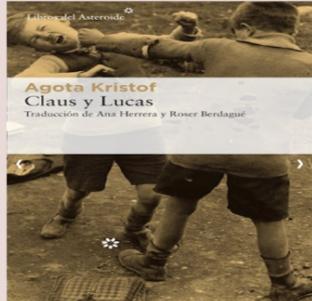
Listado Libros

Biblioteca Entre hojas

Filtrar por :

Solo Reservados | Solo Cancelados | Solo En Cuso | Solo Finalizados | Listar Todos

Listado de libros



Título: Claus y Lucas

PORADA	ID #	TITULO	DESCRIPCION	PRECIO	DETALLES	ELIMINAR	MODIFICAR
	46	El hombre en busca del sentido	El doctor Frankl, psiquiatra y escritor, suele preguntar a sus pacientes aquejados de múltiples padecimientos: «¿Por qué no se suicida usted?» Y muchas veces, de las respuestas extrae una orientación para la psicoterapia a aplicar: a éste, lo que le ata a la vida son los hijos; al otro, un talento, una habilidad sin explotar; a un tercero, quizás, sólo unos cuantos recuerdos que merece la pena rescatar del olvido.	1700.0	Ver detalles		
	47	Vidas breves	Vidas breves cuenta la historia de Fay, de sus discretas alegrías e ilusiones desde que, en los años cuarenta, abandonó su modesta carrera de cantante por un matrimonio muy alejado del romanticismo que predican las canciones y películas de la época. Una vida en busca de amor y de verdaderos afectos en la que una extravagante mujer, la glamurosa y egocéntrica Julia, acaba convirtiéndose en una influencia sutil pero constante.	1955.0	Ver detalles		
	48	Ayer	Una inolvidable novela de Agota Kristof, la autora de la trilogía Claus y Lucas. Sandor Lester, exiliado en una fría ciudad europea, lleva una vida solitaria y monótona. Inmerso en una rutina alienante en la fábrica de relojes donde trabaja, pasa sus ratos libres escribiendo, frecuentando a gente en su misma situación o en compañía de Yolande, una mujer a la que no ama.	1300.0	Ver detalles		
	49	Claus y Lucas	En un país en guerra ocupado por un ejército extranjero; dos hermanos; Claus y Lucas; han sido abandonados por su familia y puestos al cuidado de su abuela; a la que sus vecinos llaman la Bruja. La barbarie del convulso mundo en el que viven les lleva a emular la残酷 que ven en él. De una inteligencia superior; serán capaces de utilizar cualquier recurso para sobrevivir; pero una vez asegurada su supervivencia intentan poner remedio a muchas de las dramáticas situaciones que les rodean	1660.0	Ver detalles		
	50	El revés de la trama	El comandante de policía Henry Scobie y su mujer Louise viven desde hace años junto a otros funcionarios británicos en una remota colonia de África Occidental. Un entorno asfixiante que todos están deseando abandonar, especialmente Louise. Henry, por su parte, es un hombre integro que acepta estoicamente su situación y su matrimonio con una mujer por la que siente compasión más que cariño y a la que, por encima de todo, procura hacer feliz.	2300.0	Ver detalles		
	51	El capitán y el enemigo	Un adolescente que empieza a descubrir zonas inexploradas de la personalidad de los adultos, un hombre maduro que lleva una existencia lúcida y extraña al mismo tiempo, unas vidas que transean de la tranquilidad londinense a las intrigas políticas de Panamá. Con estos elementos, El capitán y el enemigo, nos introduce magistralmente en el universo siempre vario, rico y contradictorio del ser humano.	1100.0	Ver detalles		
	52	El americano tranquilo	En El americano tranquilo se revela su declarado antiamericанизmo, atacando claramente la política intervencionista de los Estados Unidos en Indochina. El hechizo del ambiente da pie a una novela que refleja, además, un complejo entramado político, aunque como dice el autor en la dedicatoria "este es un relato y no un fragmento de la historia".	2500.0	Ver detalles		
	53	Una suerte pequeña	Después de veinte años una mujer vuelve a la Argentina, de donde partió escapando de una desgracia. Pero la que regresa es otra: no luce igual, su voz es diferente. Ni siquiera lleva el mismo nombre. ¿La reconocerán quienes la conocieron entonces? ¿La reconocerá él? Mary Lohan, Mané Lauría o María Elena Pujol -la que es, la que fue, la que había sido alguna vez- vuelve al suburbio de Buenos Aires donde formó una familia y vivió hasta que decidió huir.	1800.0	Ver detalles		
	54	Las viudas de los jueves	En Altos de la Cascada viven familias que llevan un mismo estilo de vida y que quieren mantenerlo cueste lo que cueste. Allí, en el country, un grupo de amigos se reúne semanalmente lejos de las miradas de sus hijos, sus empleadas domésticas y sus esposas, quienes, excluidas del encuentro varónil, se autodenominan, bromeando, "las viudas de los Jueves"	2700.0	Ver detalles		
	55	Las maldiciones	La permanente tensión entre la necesidad de trabajo de un joven de provincia y las ocultas intenciones del político que lo ha elegido como secretario privado es lo que mueve los hilos de esta novela: dos hombres en conflicto en una historia en la que hasta la paternidad está en juego.	2500.0	Ver detalles		

Show 10 entries

Search: []

Next

Showing 1 to 10 of 25 entries

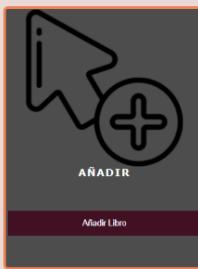
Previous 1 2 3



Ingrese Id

BUSCAR

Buscar Libro



AÑADIR

Añadir Libro

Detalle libro

Biblioteca Entre hojas +/-

Título: El hombre en busca del sentido

Id: 46

Descripción: El doctor Frankl, psiquiatra y escritor, suele preguntar a sus pacientes aquejados de múltiples padecimientos: «¿Por qué no se suicida usted?» Y muchas veces, de las respuestas extrae una orientación para la psicoterapia a aplicar: a éste, lo que le atañe a la vida son los hijos; al otro, un talento, una habilidad sin explotar; a un tercero, quizás, sólo unos cuantos recuerdos que merece la pena rescatar del olvido.

Nro Edición: 2

Fecha Edición: 2016-07-22

Dimensiones: 20x15

Nro páginas: 50

Stock: 160

Precio: 1700.0

Editorial: Herder

Categoría: Psicología

Nombre Autor: Viktor

Apellido Autor: Frankl



VIKTOR EL HOMBRE
FRANKL EN BUSCA
DE SENTIDO

¡ Califica el libro !

Obtendremos su puntuación.

★★★★★

¡ Realiza un comentario !

Escríbenos tu opinión acerca de este libro.

Enviar

USUARIO	FECHA Y HORA	CALIFICACIÓN	COMENTARIO

Búsqueda de localidad

Biblioteca Entre hojas +/-

Busqueda de Localidad



BUSCAR

Ingrese id

Buscar Localidad

ID	LOCALIDAD	ELIMINAR	MODIFICAR
1	Casilda	X	Pencil

Búsqueda de cliente

Biblioteca Entre hojas



Busqueda de Cliente



NUMERO	NOMBRE	APELLIDO	USUARIO	DOMICILIO	TELEFONO	EMAIL	FECHA INSCRIPCION	LOCALIDAD	ADMINISTRADOR	ESTADO	ELIMINAR	MODIFICAR
1	Roberto	Gutierrez	robertito	Juan B Justo 2560	3464 562147	elRobert@gmail.com	2020-12-02	Pujato	1	habilitado		

Búsqueda de autor

Biblioteca Entre hojas



Busqueda de Autor

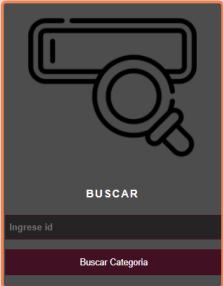


NUMERO	NOMBRE	APELLIDO	ELIMINAR	MODIFICAR
15	Viktor	Frankl		

Búsqueda de categoría

Biblioteca Entre hojas ←

Busqueda de Categoría



The search interface for categories features a magnifying glass icon over a keyboard key. Below it is a button labeled "BUSCAR". At the bottom, there are two input fields: "Ingresar id" and "Buscar Categoría", followed by a "MODIFICAR" button.

ID	CATEGORIA	ELIMINAR	MODIFICAR
13	Psicología	X	Pencil

Búsqueda de editorial

Biblioteca Entre hojas ←

Busqueda de Editorial



The search interface for publishers features a magnifying glass icon over a keyboard key. Below it is a button labeled "BUSCAR". At the bottom, there are two input fields: "Ingresar id" and "Buscar Editorial", followed by a "MODIFICAR" button.

ID	EDITORIAL	ELIMINAR	MODIFICAR
5	Herder	X	Pencil

Búsqueda de libro

Biblioteca Entre hojas

Busqueda de Libro



Ingrese id
Buscar Libro

PORTRADA	ID	TITULO	DESCRIPCION	NROEDICION	FECHAEDICION	DIMENSIONES	PAGINAS	STOCK	PRECIO	PAGINAS	EDITORIAL	CATEGORIA	ELIMINAR	MODIFICAR	NOMBRE AUTOR	APELLIDO AUTOR
	46	El hombre en busca del sentido	El doctor Frankl, psiquiatra y escritor, suele preguntar a sus pacientes aquejados de múltiples padecimientos: «¿Por qué no se suicida usted?» Y muchas veces, de las respuestas extrae una tristeza que el psicólogo intenta aplicar a él: lo que le aporta la vida son los hijos; al otro, un talento, una habilidad sin explotar; a un tercero, quizás, sólo unos cuantos recuerdos que merece la pena rescatar del olvido.	2	2016-07-22	20x15	160	50	1700.0	160	Herder	Psicología			Viktor	Frankl

Añadir localidad

Biblioteca Entre hojas

Añadir Localidad



Ingrese el nombre
Añadir Localidad

Añadir Autor

Biblioteca Entre hojas

Añadir Autor

Ingrese el nombre

Ingrese el apellido

Añadir Autor

Añadir categoría

Biblioteca Entre hojas

Añadir Categoría

Ingresar la descripción

Añadir Categoría

Añadir editorial

Biblioteca Entre hojas



Añadir Editorial



Añadir libro

Biblioteca Entre hojas



Añadir Libro

Modificar localidad

Biblioteca Entre hojas



Modificacion de Localidad



Modificar autor

Biblioteca Entre hojas



Modificacion de Autor



Modificar categoría

Biblioteca Entre hojas



Modificacion de Categoría

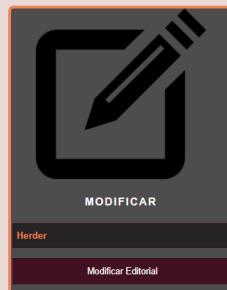


Modificar editorial

Biblioteca Entre hojas



Modificacion de Editorial



MODIFICAR
Herder
Modificar Editorial

Modificar libro

Biblioteca Entre hojas



Modificacion de Libro



El capitán y el enemigo
Un adolescente que empieza a descubrir zonas insospechadas de la personalidad de los a...
6
Fecha de edición:
05/11/2000
Dimensiones
15
X
10
208
44
1100,0
Edhasa
Ficción
Seleccione el/los autores
 Viktor Frankl
 Anita Brookner
 Agota Kristof
 Graham Greene
 Claudia Piñeiro
 Stephen King
 Pierre Lemaitre
 René Goscinny
Subir portada (opcional)
Seleccionar archivo No se eligió archivo
Modificar Libro

Menu cliente

Biblioteca Entre hojas

I Bienvenido Cliente **Juliet Carracedo** de Cañada de Gomez !

Mi Cuenta Listados

Biblioteca Entre Hojas bibliotecaentrehojas@gmail.com Galeria Stigliano Derechos Reservados

Modificar cuenta

Biblioteca Entre hojas

Ingrese los nuevos datos de su cuenta

Juliet
Carracedo
Belgrano 526
341 694712
juliCarracedo@gmail.com
Cañada de Gomez
julicarrace2
.....
Actualizar datos

Eliminar cuenta



Cerrar sesión

Listado autores

Biblioteca Entre hojas

Listado de autores

NUMERO	NOMBRE	APELLIDO
15	Viktor	Frankl
16	Anita	Brookner
17	Agota	Kristof
18	Graham	Greene
19	Claudia	Piñeiro
20	Stephen	King
21	Pierre	Lemaitre
22	René	Goscinnny



The search interface for authors features a magnifying glass icon over a dark background. Below the icon is the word 'BUSCAR'. At the bottom, there are two input fields: 'Ingresar id' and 'Buscar Autor', separated by a horizontal line.

Listado categorías

Biblioteca Entre hojas

Listado de Categorias

NUMERO	DESCRIPCION
13	Psicología
14	Ficción
15	Policial
16	Terror
17	Infantil



The search interface for categories features a magnifying glass icon over a dark background. Below the icon is the word 'BUSCAR'. At the bottom, there are two input fields: 'Ingresar id' and 'Buscar Categoria', separated by a horizontal line.

Listado editoriales

Biblioteca Entre hojas

Listado de Editoriales

NUMERO	NOMBRE
5	Herder
6	Libros del asteroide
7	Edhasa
8	Debolsillo
9	Alfaguara
10	Libros del Zorzal



Listado de libros

Listado de libros



Título: El revés de la trama

Show: 10 entries

Search:

PORTADA	ID	TÍTULO	DESCRIPCIÓN	PRECIO	DETALLES	SOLICITAR
	46	El hombre en busca del sentido	El doctor Frankl, psiquiatra y escritor, suele preguntar a sus pacientes aquejados de múltiples padecimientos: «¿Por qué no se suicida usted?» Y muchas veces, de las respuestas extrae una orientación para la psicoterapia a aplicar: a éste, lo que le ata a la vida son los hijos; al otro, un talento, una habilidad sin explotar; a un tercero, quizás, sólo unos cuantos recuerdos que merece la pena rescatar del olvido.	1700.0	Ver detalles	Reservar Libro
	47	Vidas breves	Vidas breves cuenta la historia de Fay, de sus discretas alegrías e ilusiones desde que, en los años cuarenta, abandonó su modesta carrera de cantante por un matrimonio muy alejado del romanticismo que predicaban las canciones y películas de la época. Una vida en busca de amor y de verdaderos afectos en la que una extravagante mujer, la glamurosa y egocéntrica Julia, acaba convirtiéndose en una influencia sutil pero constante.	1955.0	Ver detalles	Reservar Libro
	48	Ayer	Una inolvidable novela de Agota Kristof, la autora de la trilogía Claus y Lucas. Sándor Lester, exiliado en una fría ciudad europea, lleva una vida solitaria y monótona. Inmerso en una rutina alienante en la fábrica de relojes donde trabaja, pasa sus ratos libres escribiendo, frecuentando a gente en su misma situación o en compañía de Yolande, una mujer a la que no ama.	1300.0	Ver detalles	Reservar Libro
	49	Claus y Lucas	En un país en guerra ocupado por un ejército extranjero, dos hermanos: Claus y Lucas, han sido abandonados por su familia y puestos al cuidado de su abuela; a la que sus vecinos llaman la Bruja. La barbarie del convulso mundo en el que viven les lleva a emular la残酷 que ven en él. De una inteligencia superior, serán capaces de utilizar cualquier recurso para sobrevivir; pero una vez asegurada su supervivencia intentan poner remedio a muchas de las dramáticas situaciones que les rodean	1660.0	Ver detalles	Reservar Libro
	50	El revés de la trama	El comandante de policía Henry Scobie y su mujer Louise viven desde hace años junto a otros funcionarios británicos en una remota colonia de África Occidental. Un entorno asfixiante que todos están deseando abandonar, especialmente Louise. Henry, por su parte, es un hombre íntegro que acepta estoicamente su situación y su matrimonio con una mujer por la que siente compasión más que cariño y a la que, por encima de todo, procura hacer feliz.	2300.0	Ver detalles	Reservar Libro
	51	El capitán y el enemigo	Un adolescente que empieza a descubrir zonas insospechadas de la personalidad de los adultos, un hombre maduro que lleva una existencia lúcida y extraña al mismo tiempo, unas vidas que transitan de la tranquilidad londinense a las intrigas políticas de Panamá. Con estos elementos, El capitán y el enemigo, nos introduce magistralmente en el universo siempre vario, rico y contradictorio del ser humano.	1100.0	Ver detalles	Reservar Libro
	52	El americano tranquilo	En El americano tranquilo se revela su declarado antiamericanismo, atacando claramente la política intervencionista de los Estados Unidos en Indochina. El hechizo del ambiente da pie a una novela que refleja, además, un complejo entramado político, aunque como dice el autor en la dedicatoria "este es un relato y no un fragmento de la historia."	2500.0	Ver detalles	Reservar Libro
	53	Una suerte pequeña	Después de veinte años una mujer vuelve a la Argentina, de donde partió escapando de una desgracia. Pero la que regresa es otra: no luce igual, su voz es diferente. Ni siquiera lleva el mismo nombre. ¿La reconocerán quienes la conocieron entonces? ¿La reconocerá él? Mary Lohan, Mariel Lauria o María Elena Pujol -la que es, la que fue, la que había sido alguna vez- vuelve al suburbio de Buenos Aires donde formó una familia y vivió hasta que decidió huir.	1800.0	Ver detalles	Reservar Libro
	54	Las viudas de los jueves	En Altos de la Cascada viven familias que llevan un mismo estilo de vida y que quieren mantenerlo cueste lo que cueste. Allí, en el country, un grupo de amigos se reúne semanalmente lejos de las miradas de sus hijos, sus empleadas domésticas y sus esposas, quienes, excluidas del encuentro varonil, se autodenominan, bromeando, "las viudas de los jueves"	2700.0	Ver detalles	Reservar Libro
	55	Las maldiciones	La permanente tensión entre la necesidad de trabajo de un joven de provincia y las ocultas intenciones del político que lo ha elegido como secretario privado es lo que mueve los hilos de esta novela: dos hombres en conflicto en una historia en la que hasta la paternidad está en juego.	2500.0	Ver detalles	Reservar Libro

Showing 1 to 10 of 25 entries

Previous [1](#) [2](#) [3](#)

Detalle libro

Biblioteca Entre hojas

Titulo: **El hombre en busca del sentido**

Id: 46

Descripción: El doctor Frankl, psiquiatra y escritor, suele preguntar a sus pacientes aquejados de múltiples padecimientos: «¿Por qué no se suicida usted?» Y muchas veces, de las respuestas extrae una orientación para la psicoterapia a aplicar: a éste, lo que le ata a la vida son los hijos; al otro, un talento, una habilidad sin explotar; a un tercero, quizás, sólo unos cuantos recuerdos que merece la pena rescatar del olvido.

Nro Edición: 2

Fecha Edición: 2016-07-22

Dimensiones: 20x15

Nro páginas: 50

Stock: 160

Precio: 1700.0

Editorial: Herder

Categoría: Psicología

Nombre Autor: Viktor

Apellido Autor: Frankl



¡ Califica el libro !

Obtendremos su puntuación.

★ ★ ★ ★ ★

¡ Realiza un comentario !

Escríbenos tu opinión acerca de este libro.

Enviar

USUARIO	FECHA Y HORA	CALIFICACION	COMENTARIO
julicarrace2	2022-01-22 19:51:44	★★★★★	Me encanto la trama y su final es definitivamente sorprendente, lo recomiendo!

Comentarios + calificación

USUARIO	FECHA Y HORA	CALIFICACION	COMENTARIO
julicarrace2	2022-01-22 19:51:44	★★★★★	Me encanto la trama y su final es definitivamente sorprendente, lo recomiendo!

Comentarios + calificación

Biblioteca Entre hojas

Título: **Vidas breves**

Id: 47

Descripción: Vidas breves cuenta la historia de Fay, de sus discretas alegrías e ilusiones desde que, en los años cuarenta, abandonó su modesta carrera de cantante por un matrimonio muy alejado del romanticismo que predicaban las canciones y películas de la época. Una vida en busca de amor y de verdaderos afectos en la que una extravagante mujer, la glamurosa y egocéntrica Julia, acaba convirtiéndose en una influencia sutil pero constante.

Nro Edición: 4

Fecha Edición: 2021-04-12

Dimensiones: 18x15

Nro páginas: 37

Stock: 304

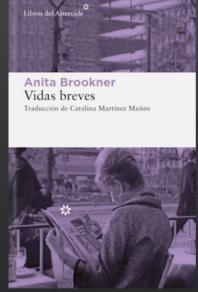
Precio: 1955.0

Editorial: Libros del asteroide

Categoría: Ficción

Nombre Autor: Anita

Apellido Autor: Brookner



¡ Califica el libro !

Obtendremos su puntuación:



¡ Realiza un comentario !

Escríbenos tu opinión acerca de este libro.

Enviar

USUARIO	FECHA Y HORA	CALIFICACION	COMENTARIO
julicarrace2	2022-01-22 19:51:44	★★★★★	Me encanto la trama y su final es definitivamente sorprendente, lo recomiendo!

Diseño adaptable a varias resoluciones de pantalla:

Inicio (ventana pequeña)

Biblioteca Entre hojas

Iniciar sesión

La mejor colección de libros para lectores apasionados

Le brindamos la mejor manera de vivir la vida a través de las palabras

Nuestros Servicios

Elijanos y te sentirás satisfecho. ¿Cerrarías los ojos a tantas historias?



Bibliotecarios Expertos

Exelente atención al cliente

Responsables de la catalogación, clasificación, indexación y exposición de los materiales, capaces de ayudarte y guiarte en lo que necesites



Buenos libros, buenos ratos

Numerosa Variedad

La diversidad de libros es tan grande que puede resultar difícil contarlos y clasificarlos, sin embargo, contamos con categorías específicas para encontrar lo que buscas



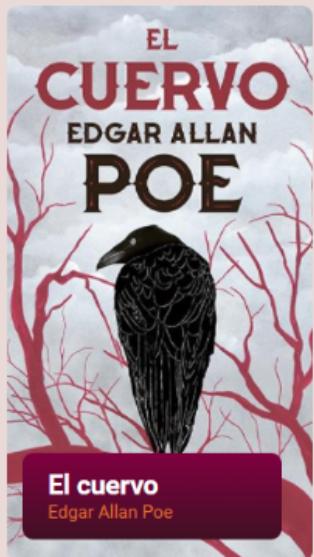
Sólo hay un bien: el conocimiento

Pedidos las 24 horas

Pedidos de documentos, interbibliotecarios, especiales para entidades y además contamos con un buzón 24 horas de devolución de documentos

Libros Reconocidos

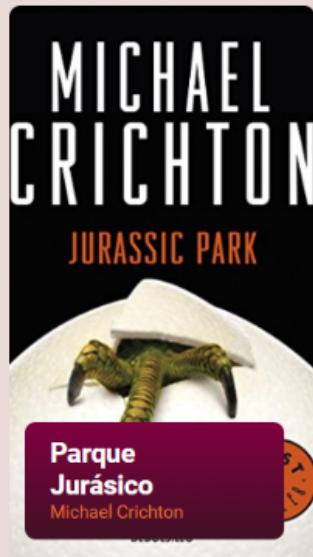
Un buen libro es más que una piedra preciosa



El cuervo
Edgar Allan Poe



Sinsajo
Suzanne Collins



Parque Jurásico
Michael Crichton

Biblioteca Entre Hojas

bibliotecaentrehojas@gmail.com

Galeria Stigliano

[Contacto](#)

Biblioteca Entre hojas



¡ Contactate con nosotros !

¡Gracias por su interés! Complete los datos para contactarnos.
Estamos aquí para lo que necesite.

Nombre:

Apellido:

Correo electrónico:

Mensaje:

Contactar

Biblioteca Entre hojas

¡ Bienvenido Admin Roberto Gutierrez !

Cuenta

Editar

Eliminar

Cerrar Sesión

Listados



Biblioteca Entre Hojas

bibliotecaentrehojas@gmail.com

Galeria Stigliano

Derechos Reservados

Listado Clientes (ventana pequeña)

Biblioteca Entre hojas

Filtrar por :

Solo Suspendidos

Solo Habilitados

Listar Todos

Listado de Clientes

ID	NOMBRE	APELLIDO	USUARIO	DOMICILIO	TELEFONO	EMAIL	FECHAINSCRIPCION	LOCALIDAD	ESTADO	ACCION
2	Julieta	Carracedo	julicarrace2	Belgrano 526	341 694712	juliCarracedo@gmail.com	2019-05-08	Cañada de Gomez	habilitado	Suspender Cliente
3	Ernesto	Valverde	elErnest	Av.Centenario 433	341156799	ernestito@gmail.com	2021-12-05			

LOCALIDAD	Rosario
ESTADO	habilitado
ACCION	
Suspender Cliente	
ID	4
NOMBRE	Franco
APELLIDO	Giangiordano
USUARIO	fran
DOMICILIO	Belgrano 123
TELEFONO	3464516658
EMAIL	frangiordano@gmail.com
FECHAINSCRIPCION	2021-12-10
LOCALIDAD	Casilda
ESTADO	habilitado
ACCION	
Suspender Cliente	

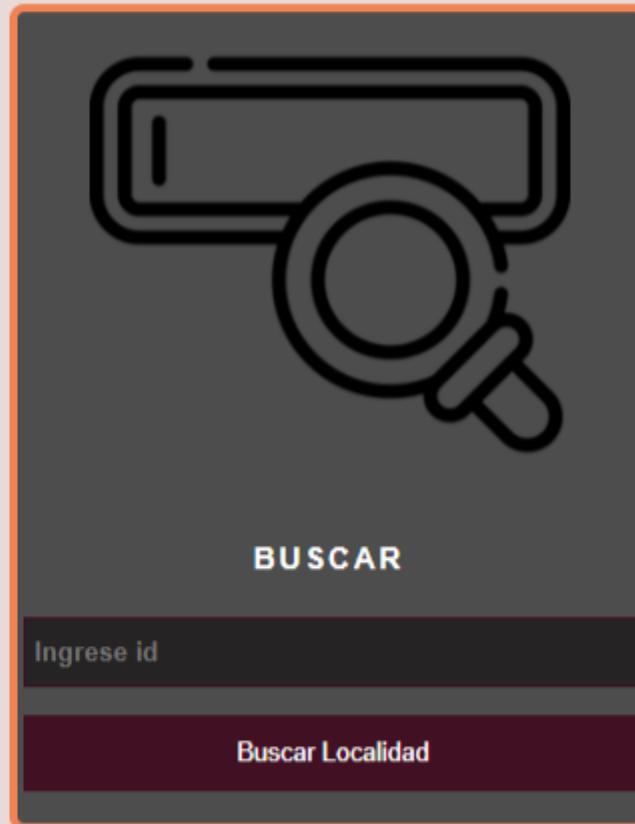


BUSCAR

Ingrese id

Buscar Cliente

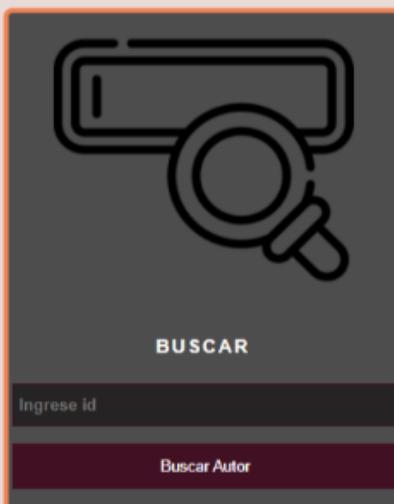
Biblioteca Entre hojas		
<h2>Listado de Localidades</h2>		
ID	1	
NOMBRE	Casilda	
ELIMINAR		
MODIFICAR		
ID	2	
NOMBRE	Pujato	
ELIMINAR		
MODIFICAR		
ID	3	
NOMBRE	Cañada de Gomez	
ELIMINAR		
MODIFICAR		
ID	4	
NOMBRE	Rosario	
ELIMINAR		
MODIFICAR		



Listado autores (ventana pequeña)

Biblioteca Entre hojas		
Listado de autores		
ID	15	
NOMBRE	Viktor	
APELLIDO	Frankl	
ELIMINAR		
MODIFICAR		
ID	16	
NOMBRE	Anita	
APELLIDO	Brookner	
ELIMINAR		
MODIFICAR		
ID	17	
NOMBRE	Agota	
APELLIDO	Kristof	
ELIMINAR		
MODIFICAR		
ID	18	
NOMBRE	Graham	
APELLIDO	Greene	
ELIMINAR		
MODIFICAR		
ID	19	
NOMBRE	Claudia	
APELLIDO	Piñeiro	
ELIMINAR		
MODIFICAR		

ID	20
NOMBRE	Stephen
APELLIDO	King
ELIMINAR	
MODIFICAR	
ID	21
NOMBRE	Pierre
APELLIDO	Lemaitre
ELIMINAR	
MODIFICAR	
ID	22
NOMBRE	René
APELLIDO	Goscinny
ELIMINAR	
MODIFICAR	



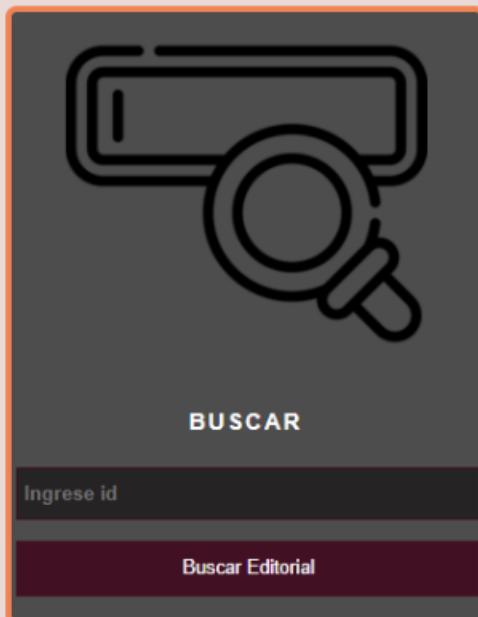
Biblioteca Entre hojas		
<h2>Listado de Categorias</h2>		
ID	13	
DESCRIPCION	Psicología	
ELIMINAR		
MODIFICAR		
ID	14	
DESCRIPCION	Ficción	
ELIMINAR		
MODIFICAR		
ID	15	
DESCRIPCION	Policial	
ELIMINAR		
MODIFICAR		
ID	16	
DESCRIPCION	Terror	
ELIMINAR		
MODIFICAR		

ID	17
DESCRIPCION	Infantil
ELIMINAR	
MODIFICAR	



Biblioteca Entre hojas		
<h2>Listado de Editoriales</h2>		
ID	5	
NOMBRE	Herder	
ELIMINAR		
MODIFICAR		
ID	6	
NOMBRE	Libros del asteroide	
ELIMINAR		
MODIFICAR		
ID	7	
NOMBRE	Edhasa	
ELIMINAR		
MODIFICAR		
ID	8	
NOMBRE	Debolsillo	
ELIMINAR		
MODIFICAR		

ID	9
NOMBRE	Alfaguara
ELIMINAR	
MODIFICAR	
ID	10
NOMBRE	Libros del Zorzal
ELIMINAR	
MODIFICAR	



Biblioteca Entre hojas ←

Filtrar por :

Solo Reservados
Solo Cancelados
Solo En Curso
Solo Finalizados
Listar Todos

Listado de libros

The image shows a book cover for 'Una suerte pequeña' by Claudia Piñeiro. The cover features a close-up photograph of a person's hands holding a red book. The author's name is at the top, and the title is in the center. A circular badge in the bottom right corner says 'BEST SELLER'. Below the badge, the word 'PEROLIBRO' is visible.

Título: Una suerte pequeña

Nota: Imagen recortada en cuanto al listado de libros, pues sigue la misma dinámica que los anteriores listados, solo que es mucho más extenso para colocar en foto.

Listado pedidos (ventana pequeña)

Biblioteca Entre hojas

Filtrar por :

Solo Reservados

Solo Cancelados

Solo En Curso

Solo Finalizados

Listado de pedidos

ESTADO	
ANULAR PEDIDO	
ID CLIENTE	2
ID LIBRO	46
TITULO	El hombre en busca del sentido
ID LIBRO	53
TITULO	Una suerte pequeña
ID LIBRO	55
TITULO	Las maldiciones

Biblioteca Entre hojas



Anita Brookner
Vidas breves
Traducción de Catalina Martínez Muñoz

Título: Vidas breves

Id: 47

Descripción: Vidas breves cuenta la historia de Fay, de sus discretas alegrías e ilusiones desde que, en los años cuarenta, abandonó su modesta carrera de cantante por un matrimonio muy alejado del romanticismo que predicaban las canciones y películas de la época. Una vida en busca de amor y de verdaderos afectos en la que una extravagante mujer, la glamurosa y egocéntrica Julia, acaba convirtiéndose en una influencia sutil pero constante.

Nro Edicion: 4

Fecha Edicion: 2021-04-12

Dimensiones: 18x15

Nro paginas: 37

Stock: 304

Precio: 1955.0

Editorial: Libros del asteroide

Categoría: Ficción

Nombre Autor: Anita

Apellido Autor: Brookner

¡ Califica el libro !

Obtendremos su puntuacion.



¡ Realiza un comentario !

Escribenos tu opinion acerca de este libro.

Enviar

USUARIO

julicarrace2

FECHA Y HORA

2022-01-22 19:51:44

CALIFICACION

★★★★★

COMENTARIO

Me encanto la trama y su final es definitivamente sorprendente, lo recomiendo!

Fragmentos de código C.U. Elegido: Reservar Libro

listaLibros.jsp

```
<%@page import="java.util.LinkedList" %>
<%@page import="entities.Autor" %>
<%@page import="entities.Libro" %>
<%@page import="entities.Cliente" %>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<%
    LinkedList<Libro> libros = (LinkedList<Libro>)request.getAttribute("Libros");
    LinkedList<Libro> librosPedidos = (LinkedList<Libro>)request.getAttribute("LibrosPedidos");
    LinkedList<Libro> librosRetirados = (LinkedList<Libro>)request.getAttribute("LibrosRetirados");
```

```

LinkedList<Libro> librosEstado = (LinkedList<Libro>)request.getAttribute("librosEstado");
Cliente cl = (Cliente) (request.getSession().getAttribute("Cliente"));
int admin = cl.getisAdmin();

%>
<meta charset="ISO-8859-1" name="description" content="Bootstrap.">
<link rel="icon" href="icons/libros.ico">
<link rel="stylesheet" href="estilos/header.css">
<link rel="stylesheet" href="estilos/tabla.css">
<link rel="stylesheet" href="estilos/busqueda.css">
<link rel="stylesheet" href="estilos/bootstrap.min.css">
<link rel="stylesheet" href="estilos/bookCard.css">
<link rel="stylesheet" href="estilos/menuFiltrar2.css">
<link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.13.0/css/all.min.css" rel="stylesheet">
<link rel="stylesheet" href="http://cdn.datatables.net/1.10.2/css/jquery.dataTables.min.css"></style>
<link rel="stylesheet" href="estilos/slideShow.css">
<link rel="stylesheet" href="estilos/jquery.dataTables.min.css">

<title>Listado Libros</title>

<script src="https://kit.fontawesome.com/cbd6eda0d3.js" crossorigin="anonymous"></script>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
<script type="text/javascript" src="http://cdn.datatables.net/1.10.2/js/jquery.dataTables.min.js"></script>
<script type="text/javascript"
src="http://maxcdn.bootstrapcdn.com/bootstrap/3.2.0/js/bootstrap.min.js"></script>
</head>

<body>

<header>
<nav class="nav__hero">
<div class="contieneLogo">
<div class="logo">
<h2 class="logo__name">Biblioteca<span class="point"> Entre hojas </span></h2>
</div>
</div>
<div>
<% if(admin != 1){ %>
<form action="ServletMenu?accion=irMenuCliente" method="post">
<button class="botonAtras" type="submit"><i class="fas fa-arrow-left"></i></button>
</form>
<% }%>
<% if(admin == 1){ %>
<form action="ServletMenu?accion=irMenuAdmin" method="post">
<button class="botonAtrasLibro" type="submit"><i class="fas fa-arrow-left"></i></button>
</form>
<% }%>
</div>
</nav>

```

```

</header>

<% if(admin == 1){ %>

<section class="filtro">
    <div class="hero__textos">
        <h2 class="title2"> Filtrar por :</h2>
    </div>
</section>

<nav>
    <input type="checkbox" id="check">
    <label for="check" class="checkbtn">
        <i class="fas fa-bars"></i>
    </label>
    <ul>
        <li>
            <form action="ServletLibro?accion=listarPorEstado" method="post">
                <input type="hidden" value="reservado" name="estado"> </input>
                <button id="button" type="submit">Solo Reservados</button>
            </form>
        </li>
        <li>
            <form action="ServletLibro?accion=listarPorEstado" method="post">
                <input type="hidden" value="cancelado" name="estado"> </input>
                <button id="button" type="submit">Solo Cancelados</button>
            </form>
        </li>
        <li>
            <form action="ServletLibro?accion=listarPorEstado" method="post">
                <input type="hidden" value="en curso" name="estado"> </input>
                <button id="button" type="submit">Solo En Curso </button>
            </form>
        </li>
        <li>
            <form action="ServletLibro?accion=listarPorEstado" method="post">
                <input type="hidden" value="finalizado" name="estado"> </input>
                <button id="button" type="submit">Solo Finalizados</button>
            </form>
        </li>
        <li>
            <form action="ServletLibro?accion=listar" method="post">
                <button id="button" type="submit">Listar Todos </button>
            </form>
        </li>
    </ul>
</nav>

```

```

<h1>
    Listado de libros
</h1>

<% LinkedList<Libro> lista = new LinkedList<>();
lista = (librosEstado == null)?(lista=libros):(lista=librosEstado);%>

<div class="slideshow-container">

<%for(Libro l : lista){%>

<div class="mySlides fade">
    
    <h4> Titulo: <%=l.getTitulo()%></h4>
</div>

<%}>

<a class="prev" onclick="plusSlides(-1)">#10094;</a>
<a class="next" onclick="plusSlides(1)">#10095;</a>
</div>

<br>
<script src="js/slideShow.js"></script>

<div class="row">
    <div class="col-sm-11 mb-3 container" style="max-width: 1000px;">
        <input type="text" id="myFilter" class="form-control" onkeyup="myFunction()" placeholder="Buscar por
        titulo...">
    </div>
</div>

<div class="row" id="myItems">
    <div class="box">
        <%for(Libro l : lista){ %>
            <div class="card col-sm-11 mb-3" style="max-width: 1000px;">
                <div class="row g-0">
                    <div class="col-md-4">
                        
                    </div>
                    <div class="col-md-8">
                        <div class="card-body">
                            <% if(admin == 1){ %>
                                <div class="actions">
                                    <form class="formularioEliminar" action="ServletLibro?accion=borrar" method="post">

```

```

        <input type="hidden" value=<%=String.valueOf(l.getId())%> name="id">
        <input type="image" id="botonEliminar" src="icons/trash-fill.png"/>
    </form>
<form action="ServletMenu?accion=modificarLibro" method="post">
    <input type="hidden" value=<%=String.valueOf(l.getId())%> name="id">
    <input type="image" id="button" src="icons/pencil.png"/>
</form>
</div>
<%}>
<label data-label="id"><b>Id:</b> <%=l.getId()%></label>
<h5 class="card-title"><b>Titulo:</b> <%=l.getTitulo()%></h5>
<p class="card-text"><b>Descripcion:</b> <%=l.getDescripcion()%></p>
<p class="card-text"><b>Precio:</b> $<%=l.getPrecio()%></p>
<div class="buttons">
    <form action="ServletLibro?accion=detalleLibro" method="post">
        <input type="hidden" value=<%=String.valueOf(l.getId())%> name="id"> </input>
        <button type="submit" class="btn btn-secondary">Ver detalles</button>
    </form>

<% if(admin != 1){ %>
    <td>
        <% if(!librosPedidos.contains(l) && !librosRetirados.contains(l)){ %>
            <form action="ServletPedido?accion=reservaLibro" method="post">
                <input type="hidden" value=<%=String.valueOf(l.getId())%> name="id"> </input>
                <button type="submit" class="btn btn-success">Reservar Libro</button>
            </form>
        <%}else if (librosPedidos.contains(l)){%>
            <form action="ServletPedido?accion=cancelarReserva" method="post">
                <input type="hidden" value=<%=String.valueOf(l.getId())%> name="id"> </input>
                <button type="submit" class="btn btn-danger">Cancelar Reserva</button>
            </form>
        <%}else{%
            <label>Pendiente de devolucion</label>
        <%}>
    </td>
<%}>
</div>
</div>
</div>
</div>
<%}>
</div>
</div>
</div>
</div>
</div>

```

```

        <form class="formulario" action="ServletLibro?accion=buscar" method="post">
            <input id="campoTexto" type="text" placeholder="Ingrese id " maxlength="10"
name="id" required>
                <button id= "boton" type="submit">Buscar Libro</button>
            </form>
        </div>

        <% if(admin == 1){ %>
<div class="card">
    
    <h4> Añadir </h4>
    <form class="formulario" action="ServletMenu?accion=anadirLibro" method="post">
        <button id="boton_AñadirLibro" type="submit">Añadir Libro</button>
    </form>
</div>
<%}%>
</div>

</body>
<script>
$(document).ready(function(){
    $('#myTable').dataTable();
});
</script>
<script>
function myFunction() {
    var input, filter, cards, cardContainer, h5, title, i;
    input = document.getElementById("myFilter");
    filter = input.value.toUpperCase();
    cardContainer = document.getElementById("myItems");
    cards = cardContainer.getElementsByClassName("card");
    for (i = 0; i < cards.length; i++) {
        title = cards[i].querySelector(".card-body h5.card-title");
        if (title.innerText.toUpperCase().indexOf(filter) > -1) {
            cards[i].style.display = "";
        } else {
            cards[i].style.display = "none";
        }
    }
}
</script>

</html>

```

ServletPedido.java

```

package Servlet;

import java.io.IOException;
import java.time.LocalDate;
import java.util.LinkedList;

```

Universidad Tecnológica Nacional

```

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import entities.Cliente;
import entities.Libro;
import entities.Pedido;
import entities.PedidoLibro;
import logic.LogicCliente;
import logic.LogicLibro;
import logic.LogicPedido;

/**
 * Servlet implementation class ServletPedido
 */
@WebServlet("/ServletPedido")
public class ServletPedido extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public ServletPedido() {
        super();
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

        String accion = request.getParameter("accion");
        if(accion.equalsIgnoreCase("listar")) {
            buscar(request,response);
        }else if(accion.equalsIgnoreCase("reservaLibro")) {
            reservarLibro(request,response);
        }else if(accion.equalsIgnoreCase("cancelarReserva")) {
            cancelarReserva(request,response);
        }else if(accion.equalsIgnoreCase("confirmarPedido")) {
            confirmarPedido(request,response);
        }else if(accion.equalsIgnoreCase("finalizarPedido")) {
            finalizarPedido(request,response);
        }else if(accion.equalsIgnoreCase("anularPedido")) {
            anularPedido(request,response);
        }
    }
}

```

```

private void anularPedido(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    LogicPedido ctrlped = new LogicPedido();
    int nroPed = Integer.parseInt(request.getParameter("nro"));
    String idCl = request.getParameter("idCl");
    ctrlped.anularPedido(nroPed,idCl);
    request.setAttribute("mensaje", "Pedido anulado");
    request.getRequestDispatcher("WEB-INF/menuAdmin.jsp").forward(request, response);
}

private void finalizarPedido(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException{
    LogicPedido ctrlped = new LogicPedido();
    int nroPed = Integer.parseInt(request.getParameter("nro"));
    ctrlped.finalizarPedido(nroPed);
    request.setAttribute("mensaje", "Pedido finalizado");
    request.getRequestDispatcher("WEB-INF/menuAdmin.jsp").forward(request, response);
}

private void confirmarPedido(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    LogicPedido ctrlped = new LogicPedido();
    int nroPed = Integer.parseInt(request.getParameter("nro"));
    ctrlped.confirmarPedido(nroPed);
    request.setAttribute("mensaje", "Pedido confirmado");
    request.getRequestDispatcher("WEB-INF/menuAdmin.jsp").forward(request, response);
}

private void cancelarReserva(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    LogicPedido ctrlped = new LogicPedido();
    int idLibro = Integer.parseInt(request.getParameter("id"));
    Cliente cliente = (Cliente) request.getSession().getAttribute("Cliente");
    int idCliente = cliente.getId();

    ctrlped.cancelarReserva(idLibro, idCliente);

    request.setAttribute("mensaje", "Reserva cancelada");
    request.getRequestDispatcher("WEB-INF/menuCliente.jsp").forward(request, response);
}

private void reservarLibro(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

    int idLibro = Integer.parseInt(request.getParameter("id"));
    Cliente cliente = (Cliente) request.getSession().getAttribute("Cliente");
    int idCliente = cliente.getId();

    LogicPedido ctrlped = new LogicPedido();

```

```

LogicLibro ctrllib = new LogicLibro();
LogicCliente ctrlcli = new LogicCliente();
Cliente cl = ctrlcli.buscarClientePorId(idCliente);
Libro lib= ctrllib.buscarLib(idLibro);
LinkedList<Libro> librosPedidos = ctrlped.listadoLibCliente(idCliente, "reservado");

String mensaje = ctrlped.validarReserva(cl,lib,librosPedidos);

if(!mensaje.equalsIgnoreCase(" ")) {
    request.setAttribute("mensaje", mensaje);
    request.getRequestDispatcher("WEB-INF/menuCliente.jsp").forward(request, response);
}

else {
    Pedido ped = ctrlped.buscarReserva(idCliente);
    if(ped!=null) {
        ctrlped.agregarLibroAPedido(ped, lib);
        request.setAttribute("mensaje", "Libro reservado");
        request.getRequestDispatcher("WEB-INF/menuCliente.jsp").forward(request,
response);
    }
}
}
}

```

```

private void buscar(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

```

```

    LogicPedido ctrlped = new LogicPedido();
    String estado = request.getParameter("estado");

    LinkedList<Pedido> pedidos = ctrlped.listadoPed(estado);
    LinkedList<PedidoLibro> pl = new LinkedList<>();

```

```

    for(Pedido p: pedidos) {
        PedidoLibro pedlib = new PedidoLibro();
        pedlib.setPed(p);

```

```

        pedlib.setLibros(ctrlped.listadoLibPed(p.getNroPedido()));
        pl.add(pedlib);
    }

    request.setAttribute("Pedidos", pl);

    request.getRequestDispatcher("WEB-INF/listaPedido.jsp").forward(request, response);

}

}

```

LogicCliente.java

```

package logic;

import java.util.LinkedList;

import data.DataCliente;
import entities.Cliente;

public class LogicCliente {

    private DataCliente dc;

    public LogicCliente() {
        dc = new DataCliente();
    }

    public LinkedList<Cliente> listadoCliente() {
        return dc.listado();
    }

    public LinkedList<Cliente> listadoPorEstado(String estado) {
        return dc.listadoPorEstado(estado);
    }

    public Cliente buscarCliente(String user, String pass) {
        return dc.buscarCliente(user, pass);
    }

    public Cliente buscarClientePorId(int id) {
        return dc.buscarClientePorId(id);
    }

    public int validarCliente(String user) {
        return dc.validaCliente(user);
    }
}

```

```

public int agregarCliente(Cliente c) {
    return dc.agregarCliente(c);
}

public void borrarCliente(int id) {
    dc.borrar(id);
}

public void modificarCliente(Cliente c) {
    dc.modificar(c);
}

public void actualizarEstadoCliente(int id, String string) {
    dc.actualizarEstadoCliente(id,string);
}

}
}

```

DataCliente.java

```

package data;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.time.LocalDate;
import java.util.LinkedList;

import entities.Cliente;

public class DataCliente {

    public Cliente buscarCliente(String user, String pass) {

        DataLocalidad dl = new DataLocalidad();
        Cliente c=null;
        PreparedStatement stmt=null;
        ResultSet rs=null;
        try {
            stmt=DbHandler.getInstancia().getConn().prepareStatement(
                "select id,nombre, apellido, user, password, domicilio, telefono, email,
fechalincripcion, idLocalidad, isAdmin, estado from cliente where user=? and password=?");
            stmt.setString(1, user);
            stmt.setString(2, pass);
            rs=stmt.executeQuery();

            if(rs!=null && rs.next()) {
                c=new Cliente();
                c.setId(rs.getInt("id"));

```

```

        c.setNombre(rs.getString("nombre"));
        c.setApellido(rs.getString("apellido"));
        c.setUser(rs.getString("user"));
        c.setPassword(rs.getString("password"));
        c.setDomicilio(rs.getString("domicilio"));
        c.setTelefono(rs.getString("telefono"));
        c.setEmail(rs.getString("email"));
        c.setFechaInscripcion(rs.getObject("fechalincripcion",LocalDate.class));
        c.setLocalidad(dl.buscarLocalidad(rs.getInt("idLocalidad")));
        c.setisAdmin(rs.getInt("isAdmin"));
        c.setEstado(rs.getString("estado"));
    }
} catch (SQLException e) {
    e.printStackTrace();
}finally {
    try {
        if(rs!=null) {rs.close();}
        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return c;
}

```

```
public Cliente buscarClientePorId(int id) {
```

```

    DataLocalidad dl = new DataLocalidad();
    Cliente c=null;
    PreparedStatement stmt=null;
    ResultSet rs=null;
    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(
            "select id,nombre, apellido, user, domicilio, telefono, email,
fechalincripcion, idLocalidad, isAdmin, estado from cliente where id=?");
        stmt.setInt(1, id);
        rs=stmt.executeQuery();

        if(rs!=null && rs.next()) {
            c=new Cliente();
            c.setId(rs.getInt("id"));
            c.setNombre(rs.getString("nombre"));
            c.setApellido(rs.getString("apellido"));
            c.setUser(rs.getString("user"));
            c.setDomicilio(rs.getString("domicilio"));
            c.setTelefono(rs.getString("telefono"));
            c.setEmail(rs.getString("email"));
            c.setFechaInscripcion(rs.getObject("fechalincripcion",LocalDate.class));
            c.setLocalidad(dl.buscarLocalidad(rs.getInt("idLocalidad")));
            c.setisAdmin(rs.getInt("isAdmin"));
        }
    }
}
```

```

        c.setEstado(rs.getString("estado"));
    }
} catch (SQLException e) {
    e.printStackTrace();
}finally {
    try {
        if(rs!=null) {rs.close();}
        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return c;
}

public int agregarCliente(Cliente c) {
    PreparedStatement stmt= null;
    ResultSet keyResultSet=null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "insert into cliente(nombre, apellido, user, password,
domicilio, telefono, email, fechalincripcion, idLocalidad, isAdmin, estado) values(?,?,?,?,?,?,?,?,?,?)",
                PreparedStatement.RETURN_GENERATED_KEYS
            );
        stmt.setString(1, c.getNombre());
        stmt.setString(2, c.getApellido());
        stmt.setString(3, c.getUser());
        stmt.setString(4, c.getPassword());
        stmt.setString(5, c.getDomicilio());
        stmt.setString(6, c.getTelefono());
        stmt.setString(7, c.getEmail());
        stmt.setObject(8, c.getFechalincripcion());
        stmt.setInt(9, c.getLocalidad().getId());
        stmt.setInt(10, c.getisAdmin());
        stmt.setString(11, c.getEstado());

        stmt.executeUpdate();

        keyResultSet=stmt.getGeneratedKeys();
        if(keyResultSet!=null && keyResultSet.next()){
            c.setId(keyResultSet.getInt(1));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
}

```

```

try {
    if(keyResultSet!=null)keyResultSet.close();
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
    return 1;
}
}
return 0;
}

public LinkedList<Cliente> listado(){
    DataLocalidad dl = new DataLocalidad();
    PreparedStatement stmt=null;
    ResultSet rs=null;
    LinkedList<Cliente> list= new LinkedList<>();

    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(
            "select id,nombre, apellido, user, password, domicilio, telefono, email,
fechalincripcion, idLocalidad, isAdmin, estado from cliente where isAdmin=?");
        stmt.setInt(1, 0);
        rs=stmt.executeQuery();
        if(rs!=null) {
            while(rs.next()) {
                Cliente c=new Cliente();
                c.setId(rs.getInt("id"));
                c.setNombre(rs.getString("nombre"));
                c.setApellido(rs.getString("apellido"));
                c.setUser(rs.getString("user"));
                c.setPassword(rs.getString("password"));
                c.setDomicilio(rs.getString("domicilio"));
                c.setTelefono(rs.getString("telefono"));
                c.setEmail(rs.getString("email"));

                c.setFechalincripcion(rs.getObject("fechalincripcion",LocalDate.class));
                c.setLocalidad(dl.buscarLocalidad(rs.getInt("idLocalidad")));
                c.setisAdmin(rs.getInt("isAdmin"));
                c.setEstado(rs.getString("estado"));
                list.add(c);
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    finally {
        try {
            if(rs!=null) {rs.close();}
        }
    }
}

```

```

        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

return list;
}

public void borrar(int id) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement("delete from cliente where id=?");
        stmt.setInt(1, id);
        stmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null)stmt.close();
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

public void modificar(Cliente c) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "update cliente set nombre=?, apellido=?, user=?,
password=?, domicilio=?, telefono=?, email=?, idLocalidad=?, estado=? where id=?");
        stmt.setString(1, c.getNombre());
        stmt.setString(2, c.getApellido());
        stmt.setString(3, c.getUser());
        stmt.setString(4, c.getPassword());
        stmt.setString(5, c.getDomicilio());
        stmt.setString(6, c.getTelefono());
        stmt.setString(7, c.getEmail());
        stmt.setInt(8, c.getLocalidad().getId());
        stmt.setString(9, c.getEstado());
        stmt.setInt(10, c.getId());

        stmt.executeUpdate();

    } catch (SQLException e) {
}

```

```

e.printStackTrace();
    } finally {
try {
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
}
}
}

public int validaCliente(String user) {
    PreparedStatement stmt=null;
    ResultSet rs=null;
    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(
                "select id,nombre, apellido, domicilio, telefono, email,
fechainscripcion, idLocalidad, isAdmin, estado from cliente where user=?");
        stmt.setString(1, user);
        rs=stmt.executeQuery();

        if(rs!=null && rs.next()) {
            return 1;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }finally {
        try {
            if(rs!=null) {rs.close();}
            if(stmt!=null) {stmt.close();}
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    return 0;
}

```

```

public LinkedList<Cliente> listadoPorEstado(String estado){
    DataLocalidad dl = new DataLocalidad();
    PreparedStatement stmt=null;
    ResultSet rs=null;
    LinkedList<Cliente> list= new LinkedList<>();

    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(
                "select id,nombre, apellido, user, password, domicilio, telefono, email,
fechainscripcion, idLocalidad, estado from cliente where isAdmin=? and estado=?");
        stmt.setInt(1, 0);
        stmt.setInt(2, 0);
    }
}

```

```

stmt.setString(2, estado);
rs=stmt.executeQuery();
if(rs!=null) {
    while(rs.next()) {
        Cliente c=new Cliente();
        c.setId(rs.getInt("id"));
        c.setNombre(rs.getString("nombre"));
        c.setApellido(rs.getString("apellido"));
        c.setUser(rs.getString("user"));
        c.setPassword(rs.getString("password"));
        c.setDomicilio(rs.getString("domicilio"));
        c.setTelefono(rs.getString("telefono"));
        c.setEmail(rs.getString("email"));

c.setFechaIncripcion(rs.getObject("fechalincripcion",LocalDate.class));
        c.setLocalidad(dl.buscarLocalidad(rs.getInt("idLocalidad")));

        c.setEstado(rs.getString("estado"));
        list.add(c);
    }
}

} catch (SQLException e) {
    e.printStackTrace();
}

} finally {
    try {
        if(rs!=null) {rs.close();}
        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

return list;
}

public void actualizarEstadoCliente(int id, String estado) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "update cliente set estado=? where id=?");
        stmt.setString(1, estado);
        stmt.setInt(2, id);
        stmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {

```

```

        try {
            if(stmt!=null)stmt.close();
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

LogicLibro.java

```

package logic;

import java.util.LinkedList;

import javax.servlet.http.HttpServletResponse;

import data.DataLibro;
import entities.Comentario;
import entities.Libro;

public class LogicLibro {

    private DataLibro dlib;

    public LogicLibro() {
        dlib = new DataLibro();
    }

    public void agregarLibro(Libro l) {
        dlib.agregar(l);
    }

    public LinkedList<Libro> listadoLib() {
        return dlib.listado();
    }

    public LinkedList<Libro> listarLibrosPorEstado(String estado) {
        return dlib.listadoPorEstado(estado);
    }

    public void listarImgLib(int id, HttpServletResponse response) {
        dlib.listarImg(id, response);
    }

    public Libro buscarLib(int id) {
        return dlib.buscar(id);
    }
}

```

```

    }

    public void borrarLib(int id) {
        dlib.borrar(id);
    }

    public void modificarLib(Libro l) {
        dlib.modificar(l);
    }

    public void cargarComentarioLib(Comentario c) {
        dlib.cargarComentario(c);
    }
}

```

DataLibro.java

```

package data;

import java.io.BufferedReader;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.sql.PreparedStatement;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.util.LinkedList;

import javax.servlet.http.HttpServletResponse;

import entities.Autor;
import entities.Comentario;
import entities.Libro;

public class DataLibro {

    public Libro buscar(int id) {

        DataEditorial de = new DataEditorial();
        DataCategoria dc = new DataCategoria();

        Libro l=null;
        PreparedStatement stmt=null;
        ResultSet rs=null;
        try {
            stmt=DbHandler.getInstancia().getConn().prepareStatement(

```

Universidad Tecnológica Nacional

```

        "select id,titulo, descripcion, nroEdicion, fechaEdicion, dimensiones,
paginas, stock, precio, idEditorial, idCategoria from libro where id=?");
stmt.setInt(1, id);
rs=stmt.executeQuery();

if(rs!=null && rs.next()) {
    l=new Libro();
    l.setId(rs.getInt("id"));
    l.setTitulo(rs.getString("titulo"));
    l.setDescripcion(rs.getString("descripcion"));
    l.setNroEdicion(rs.getInt("nroEdicion"));
    l.setFechaEdicion(rs.getObject("fechaEdicion",LocalDate.class));

    l.setDimensiones(rs.getString("dimensiones"));
    l.setNroPaginas(rs.getInt("paginas"));
    l.setExistencia(rs.getInt("stock"));
    l.setPrecio(rs.getDouble("precio"));
    l.setEditorial(de.buscar(rs.getInt("idEditorial")));
    l.setCategoria(dc.buscar(rs.getInt("idCategoria")));
    //l.setImagen(rs.getBlob("imagen"));
    l.setAutores(buscaAutores(id));

    l.setComentarios(buscarComentarios(l.getId()));

}

} catch (SQLException e) {
    e.printStackTrace();
}finally {
    try {
        if(rs!=null) {rs.close();}
        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return l;
}

```

```

public void cargarComentario(Comentario c) {
    PreparedStatement stmt= null;
    ResultSet keyResultSet=null;

    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "insert into comentarios_libro(idLibro,
idCliente,fecha,comentario,calificacion) values(?,?,?,?,?)");
        stmt.setInt(1, c.getLibro().getId());
        stmt.setInt(2,c.getUsuario().getId());
    }
}

```

```

stmt.setObject(3, c.getFecha());
stmt.setString(4,c.getReseña());
stmt.setInt(5, c.getCalificacion());
stmt.executeUpdate();

} catch (SQLException e) {
e.printStackTrace();
} finally {
try {
if(keyResultSet!=null)keyResultSet.close();
if(stmt!=null)stmt.close();
DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
e.printStackTrace();
}
}
}

private LinkedList<Comentario> buscarComentarios(int id) {

DataCliente dc = new DataCliente();
PreparedStatement stmt=null;
ResultSet rs=null;
LinkedList<Comentario> list= new LinkedList<>();

try {
stmt=DbHandler.getInstancia().getConn().prepareStatement(
"select idCliente, fecha, comentario, calificacion from comentarios_libro where
idLibro=?");
stmt.setInt(1, id);
rs=stmt.executeQuery();
if(rs!=null) {
while(rs.next()) {
Comentario c = new Comentario();
c.setUsuario(dc.buscarClientePorId(rs.getInt("idCliente")));
c.setFecha(rs.getObject("fecha",LocalDateTime.class));
c.setReseña(rs.getString("comentario"));
c.setCalificacion(rs.getInt("calificacion"));
list.add(c);
}
}
}

} catch (SQLException e) {
e.printStackTrace();
}

} finally {
try {
if(rs!=null) {rs.close();}
if(stmt!=null) {stmt.close();}
}

```

```

        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

return list;
}
}

private LinkedList<Autor> buscaAutores(int id) {
    PreparedStatement stmt=null;
    ResultSet rs=null;
    LinkedList<Autor> list= new LinkedList<>();
    DataAutor da = new DataAutor();

    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(
            "select idAutor from autor_libro where idLibro=?");
        stmt.setInt(1, id);
        rs=stmt.executeQuery();

        if(rs!=null) {
            while(rs.next()) {
                Autor l= da.buscar(rs.getInt("idAutor"));
                list.add(l);
            }
        }
    }

    } catch (SQLException e) {
        e.printStackTrace();
    }

    } finally {
        try {
            if(rs!=null) {rs.close();}
            if(stmt!=null) {stmt.close();}
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

return list;
}
}

public void agregar(Libro l) {
    PreparedStatement stmt= null;
    ResultSet keyResultSet=null;
    try {
        stmt=DbHandler.getInstancia().getConn().

```

```

        preparedStatement(
                "insert into libro(titulo, descripcion, nroEdicion, fechaEdicion,
dimensiones, paginas, stock, precio, idEditorial, idCategoria, imagen) values(?,?,?,?,?,?,?,?,?,?)",
                PreparedStatement.RETURN_GENERATED_KEYS
        );
        stmt.setString(1, l.getTitulo());
        stmt.setString(2, l.getDescripcion());
        stmt.setInt(3, l.getNroEdicion());
        stmt.setObject(4, l.getFechaEdicion());
        stmt.setString(5, l.getDimensiones());
        stmt.setInt(6, l.getNroPaginas());
        stmt.setInt(7, l.getExistencia());
        stmt.setDouble(8, l.getPrecio());
        stmt.setObject(9, l.getEditorial().getId());
        stmt.setObject(10, l.getCategoría().getId());
        stmt.setBlob(11,l.getImagen());

        stmt.executeUpdate();

        keyResultSet=stmt.getGeneratedKeys();
        if(keyResultSet!=null && keyResultSet.next()){
            l.setId(keyResultSet.getInt(1));
            agregarAutores(l.getAutores(), l.getId());
        }
    }

} catch (SQLException e) {
    e.printStackTrace();
} finally {
try {
    if(keyResultSet!=null)keyResultSet.close();
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
}
}
}

}

```

```

private void agregarAutores(LinkedList<Autor> autores, int idLibro) {

    for (Autor autor : autores) {
        PreparedStatement stmt=null;

        try {
            stmt= DbHandler.getInstancia().getConn().prepareStatement(
                    "insert into autor_libro(idAutor,idLibro) values(?,?)");

            stmt.setInt(1, autor.getId());
            stmt.setInt(2, idLibro);

```

```

        stmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
    try {
        if(stmt!=null)stmt.close();
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

}

public LinkedList<Libro> listado(){
    DataEditorial de = new DataEditorial();
    DataCategoria dc = new DataCategoria();
    Statement stmt=null;
    ResultSet rs=null;
    LinkedList<Libro> list= new LinkedList<>();

    try {
        stmt= DbHandler.getInstancia().getConn().createStatement();
        rs= stmt.executeQuery("select id,titulo, descripcion, nroEdicion, fechaEdicion,
dimensions, paginas, stock, precio, idEditorial, idCategoria, imagen from libro");
        if(rs!=null) {
            while(rs.next()) {
                Libro l=new Libro();
                l.setId(rs.getInt("id"));
                l.setTitulo(rs.getString("titulo"));
                l.setDescripcion(rs.getString("descripcion"));
                l.setNroEdicion(rs.getInt("nroEdicion"));
                l.setFechaEdicion(rs.getObject("fechaEdicion",LocalDate.class));
                l.setDimensiones(rs.getString("dimensiones"));
                l.setNroPaginas(rs.getInt("paginas"));
                l.setExistencia(rs.getInt("stock"));
                l.setPrecio(rs.getDouble("precio"));
                l.setEditorial(de.buscar(rs.getInt("idEditorial")));
                l.setCategoria(dc.buscar(rs.getInt("idCategoria")));
                l.setAutores(buscaAutores(l.getId()));
                l.setImagen(rs.getBinaryStream("imagen"));

                list.add(l);
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

        } finally {
            try {
                if(rs!=null) {rs.close();}
                if(stmt!=null) {stmt.close();}
                DbHandler.getInstancia().releaseConn();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }

    return list;
}

public void listarImg(int id, HttpServletResponse response) {

    InputStream inputStream=null;
    OutputStream outputStream=null;
    BufferedInputStream bufferedInputStream=null;
    BufferedOutputStream bufferedOutputStream=null;
    response.setContentType("image/*");

    try {
        outputStream=response.getOutputStream();
        PreparedStatement stmt = DbHandler.getInstancia().getConn().prepareStatement("select
id,titulo,descripcion,nroEdicion,fechaEdicion,dimensiones,paginas,stock,precio,idEditorial,idCategoria,
imagen from libro where id=?");
        stmt.setInt(1, id);
        ResultSet rs = stmt.executeQuery();

        if(rs.next()) {
            inputStream=rs.getBinaryStream("imagen");
        }
        bufferedInputStream= new BufferedInputStream(inputStream);
        bufferedOutputStream= new BufferedOutputStream(outputStream);
        int i=0;
        while((i=bufferedInputStream.read())!=-1) {
            bufferedOutputStream.write(i);
        }
    }catch(Exception e){

    }
}

public void borrar(int id) {

    //BORRAMOS LOS PEDIDOS DE ESE LIBRO
    borrarPedidosLibro(id);

    //BORRAMOS LOS AUTORES DE ESE LIBRO

```

```

borrarAutoresLibro(id);

//BORRAMOS EL LIBRO

PreparedStatement stmt= null;
try {
    stmt=DbHandler.getInstancia().getConn().
        prepareStatement("delete from libro where id=?");
    stmt.setInt(1, id);
    stmt.executeUpdate();

} catch (SQLException e) {
e.printStackTrace();
} finally {
try {
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
}
}
}

public void borrarPedidosLibro(int id) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement("delete from pedido_libro where idLibro=?");
        stmt.setInt(1, id);
        stmt.executeUpdate();

    } catch (SQLException e) {
e.printStackTrace();
} finally {
try {
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
}
}
}

public void borrarAutoresLibro(int id) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement("delete from autor_libro where idLibro=?");
        stmt.setInt(1, id);
        stmt.executeUpdate();
    }
}

```

```

        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
    try {
        if(stmt!=null)stmt.close();
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

public void modificar(Libro l) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
                prepareStatement(
                        "update libro set titulo=?, descripcion=?,
nroEdicion=?, fechaEdicion=?, dimensiones=?, paginas=?, stock=?, precio=?,
idEditorial=?, idCategoria=? where id=?");
        stmt.setString(1, l.getTitulo());
        stmt.setString(2, l.getDescripcion());
        stmt.setInt(3, l.getNroEdicion());
        stmt.setObject(4, l.getFechaEdicion());
        stmt.setString(5, l.getDimensiones());
        stmt.setInt(6, l.getNroPaginas());
        stmt.setInt(7, l.getExistencia());
        stmt.setDouble(8, l.getPrecio());
        stmt.setInt(9, l.getEditorial().getId());
        stmt.setInt(10, l.getCategoría().getId());
        stmt.setInt(11, l.getId());
        stmt.executeUpdate();

        actualizaAutores(l.getAutores(),l.getId());

        if(l.getImagen()!=null) {
            actualizarPortada(l);
        }
    }

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
    try {
        if(stmt!=null)stmt.close();
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

```

private void actualizarPortada(Libro l) {

    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "update libro set imagen=? where id=?");
        stmt.setBlob(1,l.getImagen());
        stmt.setInt(2, l.getId());
        stmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null)stmt.close();
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

```

private void actualizaAutores(LinkedList<Autor> autores, int idLibro) {

    //ELIMINAMOS LOS AUTORES PARA ESE LIBRO
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement("delete from autor_libro where idLibro=?");
        stmt.setInt(1,idLibro);
        stmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null)stmt.close();
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    //INSERTAMOS LOS NUEVOS AUTORES PARA ESE LIBRO
    agregarAutores(autores,idLibro);

}

```

```

void actualizaExistencia(int id, int stock) {
    Universidad Tecnológica Nacional

```

```

PreparedStatement stmt= null;
try {
    stmt=DbHandler.getInstancia().getConn();
        prepareStatement("update libro set stock=? where id=?");
    stmt.setInt(1,stock);
    stmt.setInt(2,id);
    stmt.executeUpdate();

} catch (SQLException e) {
e.printStackTrace();
} finally {
try {
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
}
}
}

public LinkedList<Libro> listadoPorEstado(String estado){
DataEditorial de = new DataEditorial();
DataCategoria dc = new DataCategoria();
PreparedStatement stmt=null;
ResultSet rs=null;
LinkedList<Libro> list= new LinkedList<>();

try {
    stmt=DbHandler.getInstancia().getConn().prepareStatement("select distinct id,titulo,
descripcion, nroEdicion, fechaEdicion, dimensiones, paginas, stock, precio, idEditorial, idCategoria, imagen
from libro"
+ " inner join pedido_libro pl on id=pl.idLibro inner join pedido p on
p.nroPedido=pl.nroPedido where p.estado=?");
    stmt.setString(1, estado);
    rs=stmt.executeQuery();
    if(rs!=null) {
        while(rs.next()) {
            Libro l=new Libro();
            l.setId(rs.getInt("id"));
            l.setTitulo(rs.getString("titulo"));
            l.setDescripcion(rs.getString("descripcion"));
            l.setNroEdicion(rs.getInt("nroEdicion"));
            l.setFechaEdicion(rs.getObject("fechaEdicion",LocalDate.class));
            l.setDimensiones(rs.getString("dimensiones"));
            l.setNroPaginas(rs.getInt("paginas"));
            l.setExistencia(rs.getInt("stock"));
            l.setPrecio(rs.getDouble("precio"));
            l.setEditorial(de.buscar(rs.getInt("idEditorial")));
        }
    }
}
}

```

```

        l.setCategoria(dc.buscar(rs.getInt("idCategoria")));
        l.setAutores(buscaAutores(l.getId()));
        l.setImagen(rs.getBinaryStream("imagen"));

        list.add(l);
    }
}

} catch (SQLException e) {
    e.printStackTrace();

} finally {
    try {
        if(rs!=null) {rs.close();}
        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return list;
}

}

```

LogicPedido.java

```

package logic;

import java.util.LinkedList;

import data.DataPedido;
import entities.Cliente;
import entities.Libro;
import entities.Pedido;

public class LogicPedido {

    private DataPedido dp;

    public LogicPedido() {
        dp = new DataPedido();
    }

    public LinkedList<Pedido> listadoPed(String estado) {
        return dp.listado(estado);
    }

    public LinkedList<Libro> listadoLibPed(int id) {
        return dp.librosPedido(id);
    }
}

```

```

public void agregarPedido(Pedido p) {
    dp.agregarPedido(p);
}

public void agregarLibroAPedido(Pedido p, Libro l) {
    dp.agregarLibro(p, l);
}

public Pedido buscarReserva(int idCl) {
    return dp.buscaReserva(idCl);
}

public LinkedList<Libro> listadoLibCliente(int idCliente, String estado) {
    return dp.librosPorEstado(idCliente, estado);
}

public void cancelarReserva(int idLibro, int idCliente) {
    dp.cancelarReserva(idLibro, idCliente);
}

public void confirmarPedido(int nroPed) {
    dp.confirmarPedido(nroPed);
}

public void finalizarPedido(int nroPed) {
    dp.finalizarPedido(nroPed);
}

public String validarReserva(Cliente cl, Libro lib, LinkedList<Libro> libros) {
    if(cl.getEstado().equalsIgnoreCase("suspendido")) {
        return "El usuario se encuentra actualmente suspendido";
    }else if(lib.getExistencia()==0){
        return "El libro no tiene stock";
    }else if(libros.size() == 5){
        return "Ya ha alcanzado el limite de reservas";
    }else {
        return " ";
    }
}

public void anularPedido(int nroPed, String idCl) {
    LinkedList<Libro> libros = new LinkedList<Libro>();
    libros = dp.librosPedido(nroPed);
    dp.actualizarEstado(nroPed, libros, "cancelado");
}

```

DataPedido.java

```
package data;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.time.LocalDate;
import java.util.LinkedList;

import entities.Libro;
import entities.Pedido;

public class DataPedido {

    public LinkedList<Pedido> listado(String estado){
        DataCliente dc = new DataCliente();
        PreparedStatement stmt=null;
        ResultSet rs=null;
        LinkedList<Pedido> list= new LinkedList<>();

        try{ stmt=DbHandler.getInstancia().getConn().
                prepareStatement("select nroPedido,fecha,estado,idCliente from pedido where
estado=?");
            stmt.setString(1, estado);
            rs= stmt.executeQuery();
            if(rs!=null) {
                while(rs.next()) {
                    Pedido p=new Pedido();
                    p.setNroPedido(rs.getInt("nroPedido"));
                    p.setFecha(rs.getObject("fecha",LocalDate.class));
                    p.setEstado(rs.getString("estado"));
                    p.setCliente(dc.buscarClientePorId(rs.getInt("idCliente")));
                    list.add(p);
                }
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }

        } finally {
            try {
                if(rs!=null) {rs.close();}
                if(stmt!=null) {stmt.close();}
                DbHandler.getInstancia().releaseConn();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```

        return list;
    }

public LinkedList<Libro> librosPedido(int id) {
    PreparedStatement stmt=null;
    ResultSet rs=null;
    LinkedList<Libro> list= new LinkedList<>();
    DataLibro dlib = new DataLibro();

    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(
                "select idLibro from pedido_libro where nroPedido=?");
        stmt.setInt(1, id);
        rs=stmt.executeQuery();

        if(rs!=null) {
            while(rs.next()) {
                Libro l = dlib.buscar(rs.getInt("idLibro"));

                list.add(l);
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }

    } finally {
        try {
            if(rs!=null) {rs.close();}
            if(stmt!=null) {stmt.close();}
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

return list;
}

public Pedido buscaReserva(int idCI) {

    Pedido p = null;
    PreparedStatement stmt=null;
    ResultSet rs=null;
    DataCliente dc = new DataCliente();

    try {
        stmt=DbHandler.getInstancia().getConn().prepareStatement(

```

```

        "select nroPedido, fecha, estado, idCliente from pedido where idCliente=?  

and fecha=? and estado=?");  

        stmt.setInt(1, idCl);  

        stmt.setDate(2, java.sql.Date.valueOf(LocalDate.now())); //convertimos LocalDate en  

Date  

        stmt.setString(3, "reservado");  

        rs=stmt.executeQuery();  

        if(rs!=null && rs.next()) {  

            p=new Pedido();  

            p.setNroPedido(rs.getInt("nroPedido"));  

            p.setFecha(rs.getObject("fecha",LocalDate.class));  

            p.setEstado(rs.getString("estado"));  

            p.setCliente(dc.buscarClientePorId(idCl));  

        }  

    } catch (SQLException e) {  

        e.printStackTrace();  

    } finally {  

        try {  

            if(rs!=null) {rs.close();}  

            if(stmt!=null) {stmt.close();}  

            DbHandler.getInstancia().releaseConn();  

        } catch (SQLException e) {  

            e.printStackTrace();  

        }  

    }  

    return p;  

}  

public void agregarLibro(Pedido p, Libro l) {  

    DataLibro dlib = new DataLibro();  

    PreparedStatement stmt= null;  

    ResultSet keyResultSet=null;  

    try {  

        stmt=DbHandler.getInstancia().getConn().  

            prepareStatement(  

                "insert into pedido_libro(nroPedido, idLibro) values(?,?)");  

        stmt.setInt(1, p.getNroPedido());  

        stmt.setInt(2, l.getId());  

        stmt.executeUpdate();  

        dlib.actualizaExistencia(l.getId(), l.getExistencia()-1); //actualizamos la existencia del libro  

    } catch (SQLException e) {  

        e.printStackTrace();  

    } finally {

```

```

try {
    if(keyResultSet!=null)keyResultSet.close();
    if(stmt!=null)stmt.close();
    DbHandler.getInstancia().releaseConn();
} catch (SQLException e) {
    e.printStackTrace();
}
}

public void agregarPedido(Pedido p) {
    PreparedStatement stmt= null;
    ResultSet keyResultSet=null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "insert into pedido(fecha, estado, idCliente) values(?, ?, ?)",
                PreparedStatement.RETURN_GENERATED_KEYS
            );

        stmt.setDate(1, java.sql.Date.valueOf(p.getFecha()));
        stmt.setString(2,p.getEstado());
        stmt.setInt(3, p.getCliente().getId());
        stmt.executeUpdate();

        keyResultSet=stmt.getGeneratedKeys();
        if(keyResultSet!=null && keyResultSet.next()){
            p.setNroPedido(keyResultSet.getInt(1));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(keyResultSet!=null)keyResultSet.close();
            if(stmt!=null)stmt.close();
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

```

//DEVUELVE LOS LIBROS RESERVADOS PARA UN CLIENTE EN PARTICULAR
public LinkedList<Libro> librosPorEstado(int idCliente, String estado) {
    PreparedStatement stmt=null;
    ResultSet rs=null;
    LinkedList<Libro> list= new LinkedList<>();

```

```

try {
    stmt=DbHandler.getInstancia().getConn().prepareStatement(
        "select nroPedido from pedido where idCliente=? and estado=?");
    stmt.setInt(1, idCliente);
    stmt.setString(2, estado);
    rs=stmt.executeQuery();

    if(rs!=null) {
        while(rs.next()) {

            list.addAll(librosPedido(rs.getInt("nroPedido")));
        }
    }
}

} catch (SQLException e) {
    e.printStackTrace();

} finally {
    try {
        if(rs!=null) {rs.close();}
        if(stmt!=null) {stmt.close();}
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

return list;
}

public int buscarNroPedido(int idLibro, int idCliente) {
    PreparedStatement stmt= null;
    ResultSet rs=null;

    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement(
                "select p.nroPedido from pedido p inner join pedido_libro pl
on pl.nroPedido=p.nroPedido where idCliente=? and idLibro=? and p.estado=?");
        stmt.setInt(1, idCliente);
        stmt.setInt(2, idLibro);
        stmt.setString(3, "reservado");
        rs= stmt.executeQuery();
        if(rs!=null && rs.next()) {
            return rs.getInt("nroPedido");

        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

        } finally {
    try {
        if(stmt!=null)stmt.close();
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
return 0;
}

public void cancelarReserva(int idLibro, int idCliente) {

    int nroPedido = buscarNroPedido(idLibro,idCliente);
    LinkedList<Libro> libros = new LinkedList<Libro>();
    libros = librosPedido(nroPedido);

    if (libros.size() > 1) {
        quitarLibro(nroPedido,idLibro); //Solo quitamos ese libro pero no cancelamos el pedido
completo
    }else{
        actualizarEstado(nroPedido,libros,"cancelado");
    }
}

public void actualizarEstado(int nroPedido,LinkedList<Libro> libros ,String estado) {

    DataLibro dlib = new DataLibro();
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
                prepareStatement(
                    "update pedido set estado=? where nroPedido=?");
        stmt.setString(1, estado);
        stmt.setInt(2, nroPedido);
        stmt.executeUpdate();
    if( libros != null && (estado.equals("cancelado") || estado.equals("finalizado"))) {
        for(Libro l: libros) {
            dlib.actualizaExistencia(l.getId(), l.getExistencia()+1);
        }
    }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
    try {
        if(stmt!=null)stmt.close();
        DbHandler.getInstancia().releaseConn();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

```

    }

}

private void quitarLibro(int nroPedido, int idLibro) {
    PreparedStatement stmt= null;
    try {
        stmt=DbHandler.getInstancia().getConn().
            prepareStatement("delete from pedido_libro where nroPedido=? and
idLibro=?");
        stmt.setInt(1, nroPedido);
        stmt.setInt(2, idLibro);
        stmt.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if(stmt!=null)stmt.close();
            DbHandler.getInstancia().releaseConn();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

public void confirmarPedido(int nroPed) {
    actualizarEstado(nroPed,null , "en curso");
}

public void finalizarPedido(int nroPed) {
    LinkedList<Libro> libros = new LinkedList<Libro>();
    libros = librosPedido(nroPed);
    actualizarEstado(nroPed,libros,"finalizado");
}
}

```

Pedido.java

```

package entities;

import java.time.*;
import java.util.LinkedList;

public class Pedido {

    private int nroPedido;
    private LocalDate fecha;
    private String estado;
    private Cliente cliente;
    Universidad Tecnológica Nacional

```

```

private LinkedList<Libro> libros;

public int getNroPedido() {
    return nroPedido;
}
public void setNroPedido(int nroPedido) {
    this.nroPedido = nroPedido;
}
public LocalDate getFecha() {
    return fecha;
}
public void setFecha(LocalDate fecha) {
    this.fecha = fecha;
}
public String getEstado() {
    return estado;
}
public void setEstado(String estado) {
    this.estado = estado;
}
public Cliente getCliente() {
    return cliente;
}
public void setCliente(Cliente cliente) {
    this.cliente = cliente;
}
public LinkedList<Libro> getLibro() {
    return libros;
}
public void setLibro(LinkedList<Libro> libros) {
    this.libros = libros;
}

@Override
public String toString() {
    return "Pedido [nroPedido=" + nroPedido + ", fecha=" + fecha + ", estado=" + estado + ",
cliente=" + cliente
                + ", libro=" + " " + "]";
}

public Pedido(int nroPedido, LocalDate fecha, String estado, Cliente cliente, LinkedList<Libro> libro) {
    this.nroPedido = nroPedido;
    this.fecha = fecha;
    this.estado = estado;
    this.cliente = cliente;
    this.libros = libro;
}

public Pedido(){}

```

}

}