
Projecte Final M6

- **Llibreria**

En aquest apartat explicarem, quines classes té la nostra llibreria amb les seves variables i constructors, totes les funcions de la nostre llibreria i finalment com ha quedat la interfície de la llibreria.

- ❖ **Classes i constructors**

- Clients:

```
public class Clients {  
  
    private String dni, nom, cognom, direccio;  
  
    public Clients() {  
        super();  
    }  
  
    public Clients(String _dni, String nom, String cognom, String direccio) {  
        super();  
        dni = _dni;  
        this.nom = nom;  
        this.cognom = cognom;  
        this.direccio = direccio;  
    }  
}
```

En aquesta captura podem veure les variables i els constructors declarats a la classe "Clients".

- Llibres:

```
public class Libro {  
  
    private int isbn, stock, any ;  
    private String titol, autor, editorial, tematica, ubicacio, dataAlta;  
    private double preu;  
  
    public Libro() {  
        super();  
    }  
  
    public Libro(int _isbn, int stock, int any, String dataAlta, String titol, String autor, String editorial,  
        String tematica, String ubicacio, double preu) {  
        super();  
        this.isbn = _isbn;  
        this.stock = stock;  
        this.any = any;  
        this.dataAlta = dataAlta;  
        this.titol = titol;  
        this.autor = autor;  
        this.editorial = editorial;  
        this.tematica = tematica;  
        this.ubicacio = ubicacio;  
        this.preu = preu;  
    }  
}
```

En aquesta captura podem veure les variables i els constructors declarats a la classe “Libro”.

- Linea:

```
public class Linea {  
  
    private Libro libro;  
    private int cantidad;  
    private double total;  
  
    public Linea() {  
        super();  
    }  
  
    public Linea(Libro libro, int cantidad, double total) {  
        super();  
        this.libro = libro;  
        this.cantidad = cantidad;  
        this.total = total;  
    }  
}
```

En aquesta captura podem veure les variables i els constructors declarats a la classe “Linea”.

- Vendes:

```
public class Vendes {  
  
    private int id;  
    private String fecha;  
    private String dni;  
    private ArrayList<Linea> linea;  
    private double total;  
  
    public Vendes() {  
        super();  
    }  
  
    public Vendes(int id, String fecha, String dni, ArrayList<Linea> linea, double total) {  
        super();  
        this.id = id;  
        this.fecha = fecha;  
        this.dni = dni;  
        this.linea = linea;  
        this.total = total;  
    }  
  
    public Vendes(String fecha, String dni, ArrayList<Linea> linea, double total) {  
        super();  
        this.fecha = fecha;  
        this.dni = dni;  
        this.linea = linea;  
        this.total = total;  
    }  
}
```

En aquesta captura podem veure les variables i els constructors declarats a la classe “Vendes”.

❖ Funcions de la llibreria

- Afegir un Llibre

```
// Afegir un llibre a la BBDD
@Override
public void afegirLlibre(Libro libro) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    IQuery query = new CriteriaQuery(Libro.class, Where.equal("isbn", libro.getIsbn()));
    Objects<Libro> llibres = odb.getObjects(query);

    if (llibres.isEmpty()) {
        odb.store(libro);
    } else {
        throw new Exception("Ja existeix un llibre amb aquest ISBN.");
    }

    odb.close();
}
```

- Retornar un llibre

```
// Retorna un llibre de la BBDD
@Override
public Objects<Libro> retornarLlibre(int isbn) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    IQuery query = new CriteriaQuery(Libro.class, Where.equal("isbn", isbn));
    Objects<Libro> llibre = odb.getObjects(query);

    odb.close();

    return llibre;
}
```

- Retornar tots els llibres

```
// Retorna tots els llibres de la BBDD
@Override
public Objects<Libro> retornarLlibres() throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);

    IQuery query = new CriteriaQuery(Libro.class);
    Objects<Libro> llibres = odb.getObjects(query);

    odb.close();

    if (llibres.isEmpty()) {
        throw new Exception("No s'han trobat llibres");
    } else {
        return llibres;
    }
}
```

- Eliminar un llibre

```
// Elimina un llibre de la BBDD
@Override
public void eliminarLlibre(int isbn) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    CriteriaQuery query = new CriteriaQuery(Libro.class, Where.equal("isbn", isbn));
    Objects<Libro> llibres = odb.getObjects(query);
    if (llibres.isEmpty()) {
        throw new Exception("No s'ha trobat el llibre indicat.");
    }
    Libro llibre = llibres.getFirst();
    odb.delete(libre);
    odb.close();
}
```

- Actualitzar el stock

```
// Actualitza el stock antic
@Override
public void eliminarLlibreDespresVenda(int isbn) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    CriteriaQuery query = new CriteriaQuery(Libro.class, Where.equal("isbn", isbn));
    Objects<Libro> llibres = odb.getObjects(query);
    if (llibres.isEmpty()) {
        throw new Exception("No s'ha trobat el llibre indicat.");
    }
    Libro llibre = llibres.getFirst();
    odb.delete(libre);
    odb.close();
}
```

- Actualitzar un llibre

```
// Actualitza els camps d'un llibre
@Override
public void actualitzarLlibre(int isbn_ant, int isbn_nou, int stock, int any, String dataAlta, String titol,
    String autor, String editorial, String tematica, String ubicacio, double preu) throws Exception {

    ODB odb = ODBFactory.open(DB_NAME);
    CriteriaQuery query = new CriteriaQuery(Libro.class, Where.equal("isbn", isbn_ant));
    Objects<Libro> libroAnterior = odb.getObjects(query);

    if (libroAnterior.isEmpty()) {
        throw new Exception("No s'ha trobat el llibre indicat.");
    } else {
        Libro libro = (Libro) odb.getObjects(query).getFirst();

        libro.setIsbn(isbn_nou);
        libro.setTitol(titol);
        libro.setAutor(autor);
        libro.setEditorial(editorial);
        libro.setPreu(preu);
        libro.setStock(stock);
        libro.setAny(any);
        libro.setDataAlta(dataAlta);
        libro.setTematica(tematica);
        libro.setUbicacio(ubicacio);
        odb.store(libro);
        odb.close();
    }
}
```

- Afegir un client

```
// Afegeix un client a la BBDD
@Override
public void afegirClient(Clients client) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    IQuery query = new CriteriaQuery(Clients.class, Where.equal("dni", client.getDni()));
    Objects<Clients> clients = odb.getObjects(query);

    if (clients.isEmpty()) {
        odb.store(client);
    } else {
        throw new Exception("Ja existeix un client amb aquest DNI.");
    }

    odb.close();
}
```

- Retornar un client

```
// Retorna un client de la BBDD
@Override
public Objects<Clients> retornarClient(String dni) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    IQuery query = new CriteriaQuery(Clients.class, Where.equal("dni", dni));
    Objects<Clients> client = odb.getObjects(query);

    odb.close();

    return client;
}
```

- Retornar tots els clients

```
// Retorna tots els clients de la BBDD
@Override
public Objects<Clients> retornarClients() throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);

    IQuery query = new CriteriaQuery(Clients.class);
    Objects<Clients> clients = odb.getObjects(query);

    odb.close();

    if (clients.isEmpty()) {
        throw new Exception("No s'han trobat llibres");
    } else {
        return clients;
    }
}
```

- Eliminar un client

```
// Elimina un client de la BBDD
@Override
public void eliminarClient(String dni) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    CriteriaQuery query = new CriteriaQuery(Clients.class, Where.equal("dni", dni));
    Objects<Clients> clients = odb.getObjects(query);
    if (clients.isEmpty()) {
        throw new Exception("No s'ha trobat el llibre indicat.");
    }
    Clients client = clients.getFirst();
    odb.delete(client);
    odb.close();
}
```

- Actualitzar un client

```
// Actualitza els camps d'un client
@Override
public void actualitzarClient(String dni_antig, String dni_nou, String nom, String cognom, String direccio)
    throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);
    CriteriaQuery query = new CriteriaQuery(Clients.class, Where.equal("dni", dni_antig));
    Objects<Clients> clientAnterior = odb.getObjects(query);

    if (clientAnterior.isEmpty()) {
        throw new Exception("No s'ha trobat el client indicat.");
    } else {
        Clients client = (Clients) odb.getObjects(query).getFirst();
        client.setDni(dni_nou);
        client.setNom(nom);
        client.setCognom(cognom);
        client.setDireccio(direccio);
        odb.store(client);
    }
    odb.close();
}
```

- Afegir una venda

```
// Afegir una venda a la BBDD
@Override
public void afegirVenta(Vendes venda) throws Exception {
    ODB odb = ODBFactory.open(DB_NAME);

    for (Linea linea : venda.getLinea()) {

        int isbn = linea.getLibro().getIsbn();
        IQuery query = new CriteriaQuery(Libro.class, Where.equal("isbn", isbn));
        Objects<Libro> libros = odb.getObjects(query);

        if (libros.isEmpty()) {
            throw new Exception("No s'ha trobat el llibre amb el ISBN: " + isbn);
        } else {
            Libro libro = libros.getFirst();

            if (libro.getStock() > 0) {
                if (linea.getCantidad() < libro.getStock()) {
                    libro.setStock(libro.getStock() - linea.getCantidad());
                    odb.close();
                    eliminarLibreDespresVenda(libro.getIsbn());
                    odb = ODBFactory.open(DB_NAME);
                    odb.store(libro);
                    odb.commit();
                    System.out.println("libro.getIbn() --> "+libro.getIsbn());
                    System.out.println("libro.getStock() --> "+libro.getStock());
                    System.out.println("Despres de eliminarLibreDespresVenda(libro.getIsbn(), odb);");
                } else {
                    throw new Exception("No es pot comprar el llibre amb el ISBN: " + isbn
                        + ", porque no hi ha suficient stock");
                }
            } else {
                throw new Exception("No es pot comprar el llibre amb el ISBN: " + isbn + ", porque no hi ha stock");
            }
        }
    }

    Vendes vendaa = new Vendes();
    vendaa.setLinea(venda.getLinea());
    vendaa.setDni(venda.getDni());
    vendaa.setFecha(venda.getFecha());
    vendaa.setId(venda.getId());
    vendaa.setTotal(venda.getTotal());

    odb.store(vendaa);
    odb.commit();
    odb.close();
}
```

- Retornar totes les vendes

```
// Retorna totes les vendes de la BBDD
@Override
public Objects<Vendes> retornarVendes() throws Exception {

    ODB odb = ODBFactory.open(DB_NAME);

    IQuery query = new CriteriaQuery(Vendes.class);
    Objects<Vendes> vendes = odb.getObjects(query);

    odb.close();

    if (vendes.isEmpty()) {
        throw new Exception("No s'han trobat vendes");
    } else {
        return vendes;
    }
}
```


❖ **Interficie**

- Interficie de la llibreria.

```
public interface Libreria {

    //llibre
    public void afegirLlibre(Libro libro) throws Exception;

    public void eliminarLlibre(int isbn) throws Exception;

    public void eliminarLlibreDespresVenda(int isbn) throws Exception;

    public Objects<Libro> retornarLlibre(int isbn) throws Exception;

    public Objects<Libro> retornarLlibres() throws Exception;

    public void actualitzarLlibre(int isbn_antic, int isbn_nou, int stock, int any, String dataAlta, String titol,
        String autor, String editorial, String tematica, String ubicacio, double preu) throws Exception;

    //Client
    public void afegirClient(Clients client) throws Exception;

    public Objects<Clients> retornarClient(String dni) throws Exception;

    public Objects<Clients> retornarClients() throws Exception;

    public void eliminarClient(String dni) throws Exception;

    public void actualitzarClient(String dni_antic, String dni_nou, String nom, String cognom, String direccio) throws Exception;

    //Venda
    public void afegirVenda(Vendes venda) throws Exception;

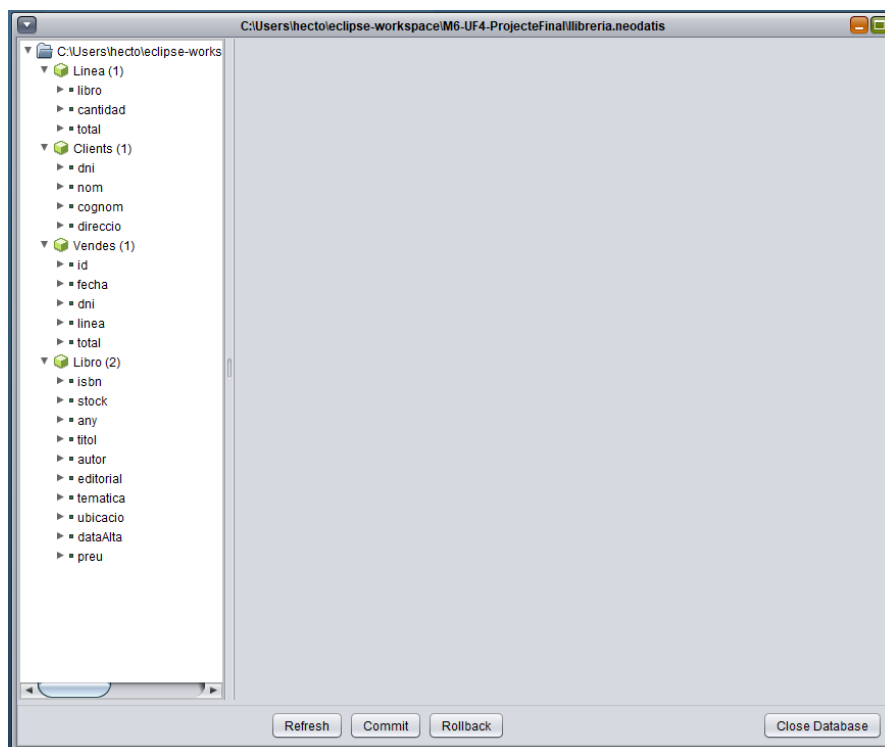
    public Objects<Vendes> retornarVendes() throws Exception;

}
```

En aquesta captura podem veure com queda la interfície de la nostre llibreria.

- **Base de Dades**

BBDD Neodatis



Aquí es pot veure l'estructura del neodatis

- **Interfície del usuari**

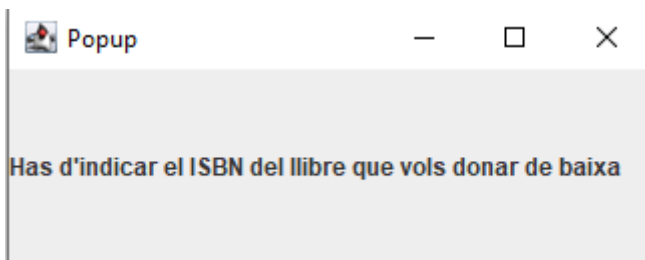
Control d'errors

```
1 import javax.swing.JFrame;
2
3
4 public class PopupOmplirCamps extends JFrame {
5
6     public PopupOmplirCamps(String mensaje) {
7         setTitle("Popup");
8         JLabel label = new JLabel(mensaje);
9         add(label);
10        setSize(300, 100);
11        setLocationRelativeTo(null);
12    }
13 }
14
```

Per controlar els diferents errors que puguin sortir durant l'execució, em creat una classe que obrira una finestra amb un missatge indicant el error.

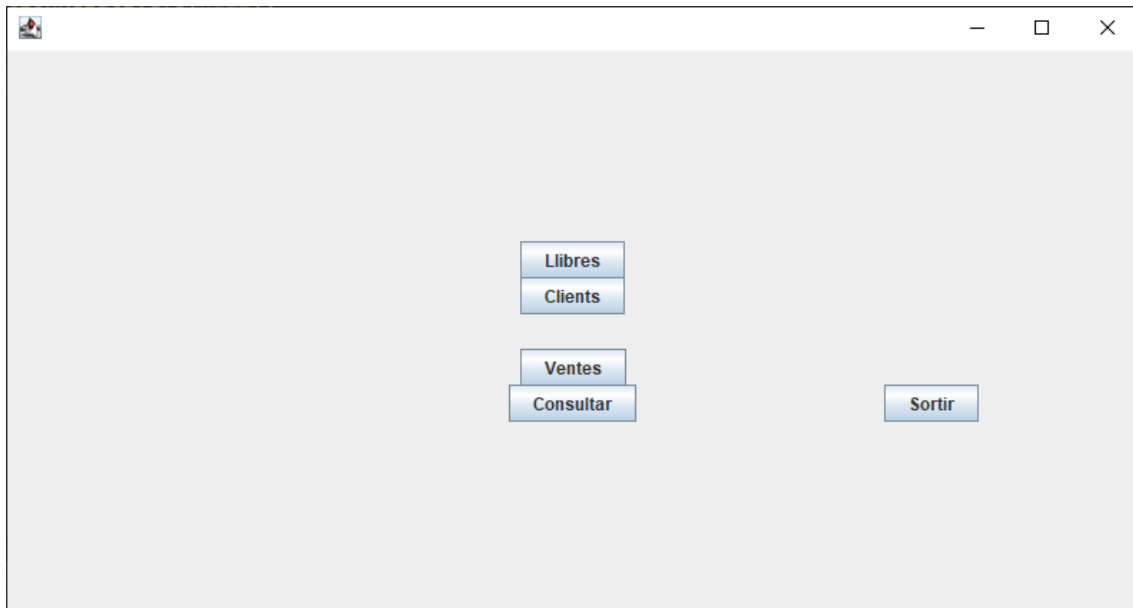
```
}else {
    PopupOmplirCamps popupOmplirCamps = new PopupOmplirCamps(
        "Has d'indicar el DNI del client que vols donar de baixa");
    popupOmplirCamps.setVisible(true);
}
```

Així és com se crida a la classe

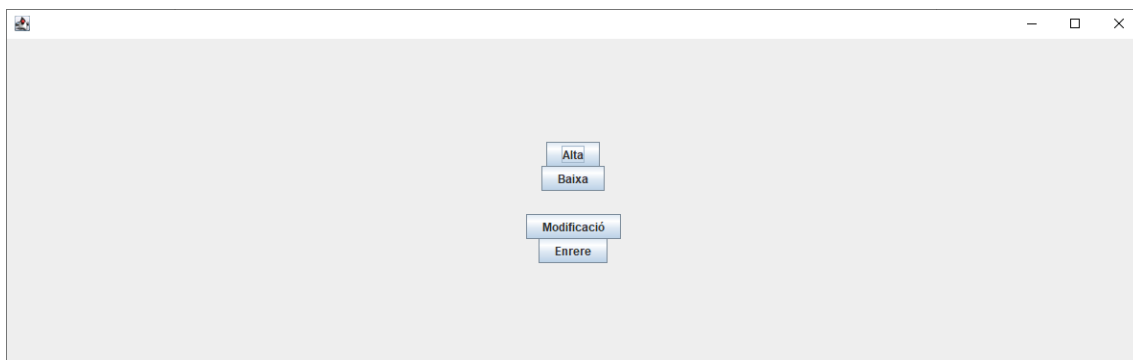


Disseny interfície

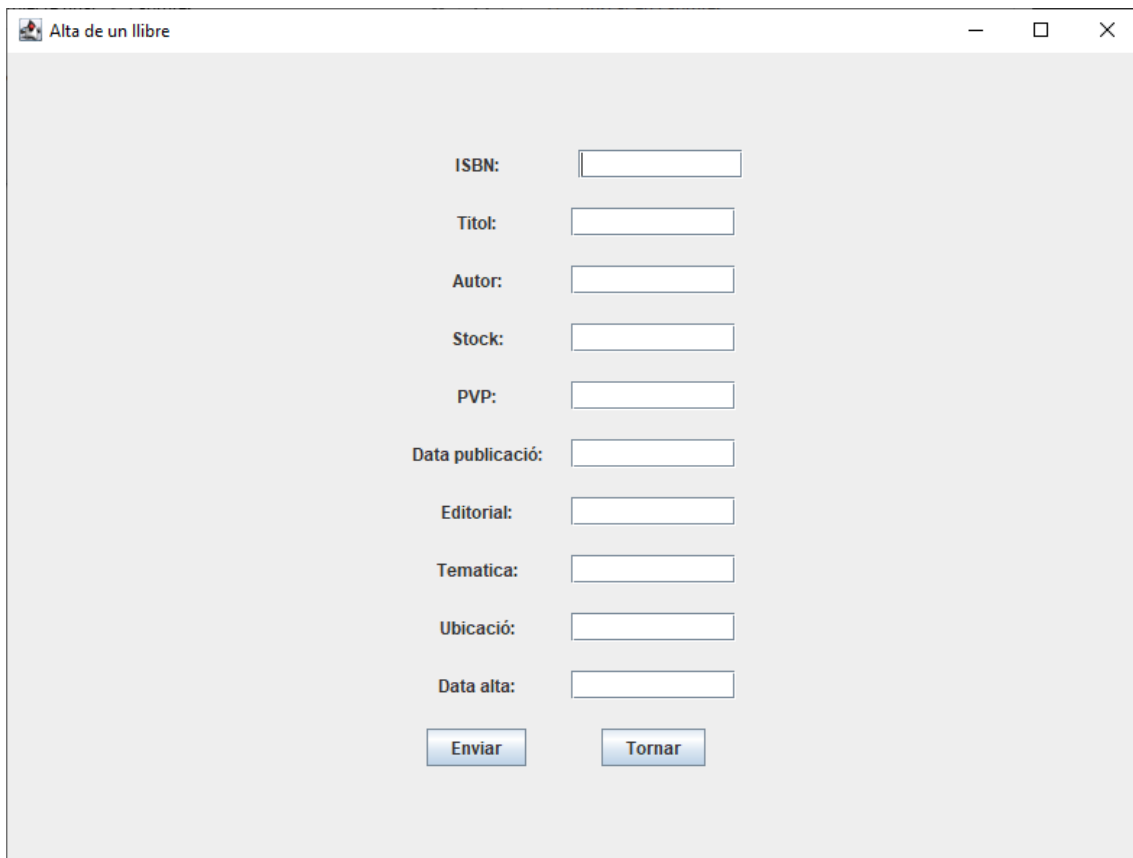
Menú inicial



Menú llibres



Alta llibres



A screenshot of a web form titled "Alta de un llibre" (Add a book). The form is displayed in a window with standard OS controls (minimize, maximize, close). It contains several input fields for book details, each preceded by a label. At the bottom, there are two buttons: "Enviar" (Send) and "Tornar" (Return).

ISBN:	<input type="text"/>
Títol:	<input type="text"/>
Autor:	<input type="text"/>
Stock:	<input type="text"/>
PVP:	<input type="text"/>
Data publicació:	<input type="text"/>
Editorial:	<input type="text"/>
Temàtica:	<input type="text"/>
Ubicació:	<input type="text"/>
Data alta:	<input type="text"/>
<input type="button" value="Enviar"/> <input type="button" value="Tornar"/>	

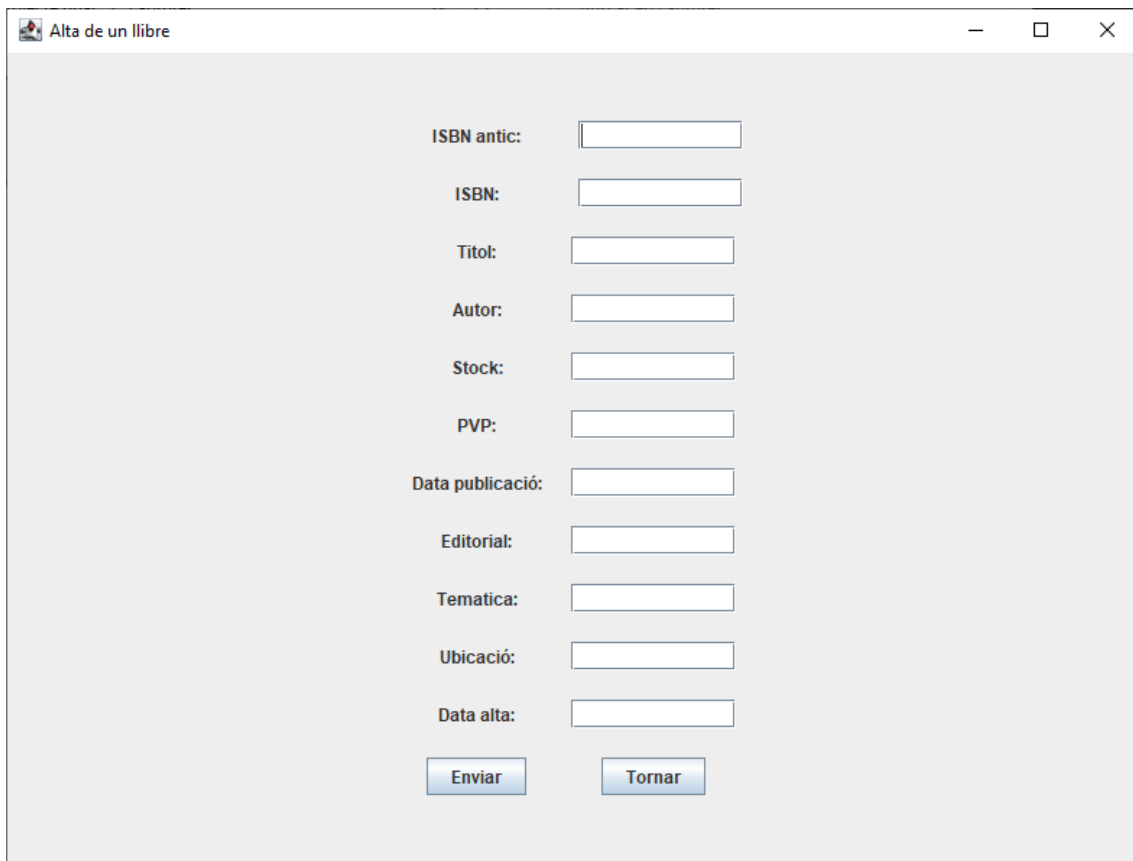
Baixa llibre



A screenshot of a web form titled "Baixa de un llibre" (Remove a book). The form is displayed in a window with standard OS controls. It contains a single input field for the ISBN, preceded by a label. Below the input field are two buttons: "Enviar" (Send) and "Tornar" (Return).

ISBN:	<input type="text"/>
<input type="button" value="Enviar"/> <input type="button" value="Tornar"/>	

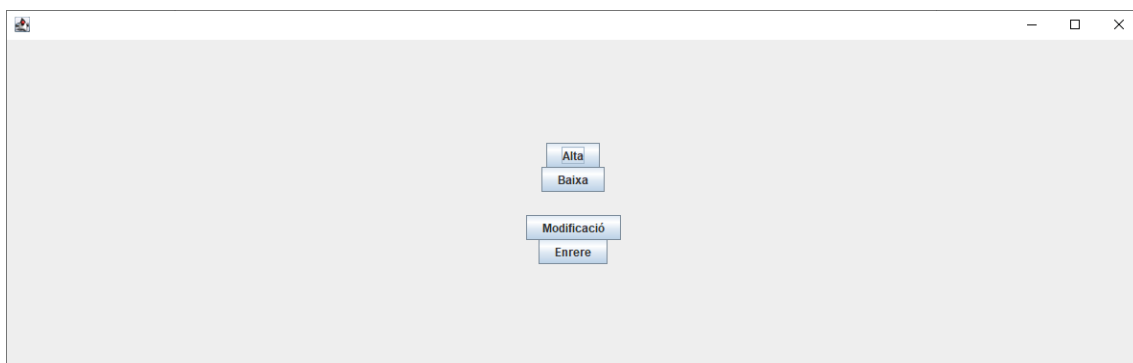
Modificació llibre



The screenshot shows a web browser window with the title "Alta de un llibre". The form contains the following fields and buttons:

- ISBN antic:
- ISBN:
- Títol:
- Autor:
- Stock:
- PVP:
- Data publicació:
- Editorial:
- Temàtica:
- Ubicació:
- Data alta:
- Enviar
- Tornar

Menú clients



The screenshot shows a web browser window with a menu containing the following buttons:

- Alta
- Baixa
- Modificació
- Enrere

Alta clients



Nou Menu

DNI:

Nom:

Cognom:

Direcció:

Baixa clients



Nou Menu

DNI:

Modificació clients



Nou Menu

DNI antic:

DNI:

Nom:

Cognom:

Direcció:

Finestra ventes

Menu de Ventas

Cliente: Fecha: 06/05/2023

Libro	Cantidad	Precio
-------	----------	--------

Afegir llibre Enviar Enrere

Total: 0€

Botó per afegir un llibre a la finestra venta

A...

Nom llibre:

Quantitat:

Enviar

Menú consulta

Llibres

Clients

Ventas

Enrere

Consulta clients/l·libres/ventes (les tres son la mateixa estructura)

Nom valors	Informació recuperada
ISBN:	111111987
Títol:	Harry Potter
Autor:	Hector vallve
Stock:	4
Any publicació:	1987
Data alta:	20/04/2023
Editorial:	Penguin
Preu:	5.3
Tematica:	Fantasia
Ubicació:	Alcover

Inici Següent Anterior Ultim Enrere