

UNIVERSIDAD PERUANA LOS ANDES
FACULTAD DE INGENIERIA
ESCUELA PROFECIONAL DE INGENIERIA DE
SISTEMAS Y COMPUTACIÓN



TEMA

**Diseño e Implementación de un sistema de login para una
Biblioteca**

CURSO: Desarrollo de Aplicaciones

DOCENTE: Ing. Fernandez Bejarano Raul Enrique

INTEGRANTES:

- Ramón Puente Eduardo Fredy
- Untiveros Moreno Diego Alonso

CICLO: IV SECCION:

A1

HUANCAYO - PERÚ

2025

Índice

Índice	2
Introducción	3
Objetivos específicos.....	3
Justificación	3
Descripción del proyecto	3
Funcionalidades principales	4
Requisitos funcionales	4
Beneficios del proyecto.....	4
Metodología	4
Resultados Esperados.....	5
Conclusiones.....	5
Evidencias del Proyecto	5
Script del Login.....	12
Script de la Base de Datos.....	43
Estructura de la Base de Datos	44
Codigo Base de Datos.....	44
Diagrama del Proyecto	48
Estructura del Proyecto iLib (Diagrama Textual)	48
Dependencias (Librerías externas)	50

Introducción

El sistema LibConnect, una aplicación informática creada con el propósito de optimizar la gestión de acceso y administración de usuarios en una biblioteca pública. Este sistema permite que tanto lectores como administradores puedan ingresar de manera segura ofreciendo a su vez herramientas para el registro, control de préstamos, devoluciones, administración de libros y generación de reportes.

Objetivos específicos

- Implementar un sistema de inicio de sesión y registro para usuarios de la biblioteca.
- Facilitar el control de préstamos, devoluciones, usuarios y libros desde una misma plataforma.
- Brindar una interfaz intuitiva para el personal encargado de la administración.
- Permitir la creación, edición y eliminación de registros dentro del sistema.
- Generar reportes de las actividades realizadas en la biblioteca.
- Mantener la seguridad de los datos con contraseñas

Justificación

Las bibliotecas públicas y privadas manejan diariamente gran cantidad de información sobre usuarios y libros. Muchos de estos procesos aún se realizan de manera manual, lo que genera errores, pérdida de tiempo y falta de control sobre los préstamos y devoluciones.

El sistema LibConnect surge como una alternativa tecnológica para modernizar la gestión bibliotecaria, ofreciendo una herramienta segura, práctica y adaptable. A través de su sistema de login, se garantiza que solo personal autorizado tenga acceso a las funciones internas, reduciendo riesgos de manipulación indebida de datos. Además, centraliza toda la información en una sola aplicación, mejorando la eficiencia del trabajo administrativo y ofreciendo una experiencia más fluida a los usuarios.

Descripción del proyecto

LibConnect es un sistema de gestión y autenticación diseñado especialmente para bibliotecas. Su propósito principal es administrar el flujo de préstamos y devoluciones de libros, el registro y control de usuarios, la edición de información y la generación de reportes de acciones realizadas dentro del sistema.

La interfaz del sistema cuenta con un panel lateral con los siguientes módulos:

- **Principal:** Muestra el mensaje de bienvenida, la fecha actual y una descripción general del sistema.
- **Préstamos:** Permite registrar nuevos préstamos de libros a los usuarios.
- **Devoluciones:** Gestiona la devolución de libros y actualiza su disponibilidad.
- **Usuarios:** Permite registrarse y iniciar sesión.
- **Libros:** Administra el catálogo de libros (registro, edición y eliminación).

- **Reportes:** Genera reportes de acciones, préstamos y devoluciones para control interno.

El sistema fue diseñado para que sea fácil de usar, adaptable a diferentes tipos de bibliotecas y que permita mantener un control detallado de todos los movimientos relacionados con los libros.

Funcionalidades principales

- Registro de nuevos usuarios mediante un formulario conectado a la base de datos.
- Inicio de sesión con verificación de credenciales seguras.
- Panel administrativo con módulos funcionales para la gestión bibliotecaria.
- Control de préstamos y devoluciones de libros en tiempo real.
- Edición y eliminación de registros existentes.
- Generación de reportes y consultas rápidas sobre la información almacenada.
- Visualización del día y fecha actual en la pantalla principal.
- Seguridad en la conexión con la base de datos y manejo cifrado de contraseñas.

Requisitos funcionales

1. Permite iniciar sesión solo a usuarios registrados.
2. Valida que los datos de acceso sean correctos antes de ingresar.
3. Permitir registrar, editar y eliminar usuarios y libros.
4. Registra los préstamos y devoluciones de manera precisa.
5. Generar reportes según los movimientos realizados.
6. Muestra mensajes informativos ante errores o acciones completadas.

Beneficios del proyecto

- Optimiza la administración de usuarios y libros dentro de la biblioteca.
- Facilita el control de préstamos y devoluciones mediante una interfaz clara.
- Reduce errores manuales y mejora la eficiencia del personal.
- Incrementa la seguridad y confidencialidad de la información.
- Permite llevar un registro histórico y automatizado de todas las acciones realizadas.
- Brinda una base sólida para futuras ampliaciones, como reservas de libros o consultas en línea.

Metodología

El proyecto se desarrolló aplicando la metodología incremental, la cual permite ir construyendo el sistema por módulos e ir probando cada parte antes de integrar nuevas funciones.

Las etapas seguidas fueron:

1. **Análisis de requerimientos:** Se definieron las necesidades del sistema y las funciones que debía cubrir.

2. **Diseño:** Se planificó la interfaz visual, la estructura de la base de datos y los diagramas de funcionamiento.
3. **Implementación:** Se desarrolló el código fuente en Java (NetBeans), conectando la aplicación con la base de datos MySQL.
4. **Pruebas:** Se verificó el correcto funcionamiento de los módulos (login, registro, préstamos, etc.).
5. **Documentación:** Se elaboraron los informes técnicos y manuales de uso del sistema.

Resultados Esperados

- Funcionamiento correcto del login y del registro de usuarios.
- Conexión estable con la base de datos MySQL.
- Control eficiente de préstamos y devoluciones.
- Interfaz moderna y fácil de usar.
- Cumplimiento de los objetivos específicos establecidos.

Conclusiones

El sistema LibConnect logró cumplir con el propósito de ofrecer una herramienta práctica para la gestión de bibliotecas, facilitando el control de préstamos, devoluciones, usuarios y libros.

Su diseño centrado en la simplicidad y seguridad permite que el personal de la biblioteca pueda realizar sus tareas con mayor rapidez y precisión.

Además, la implementación de un sistema de login contribuye a la protección de los datos y al acceso restringido solo a personal autorizado.

En conjunto, LibConnect representa un avance hacia la digitalización de procesos bibliotecarios, abriendo la posibilidad de futuras mejoras, como la incorporación de reservas en línea, notificaciones por correo o integración con catálogos virtuales.

Evidencias del Proyecto

Bienvenido a la Biblioteca Virtual LibConnect

Tipo de acceso:

Usuario



Email:

Contraseña:

Iniciar Sesión

[¿No tienes cuenta? Regístrate](#)

Crear Nueva Cuenta

Nombre:

Apellido Paterno:

Apellido Materno:

Domicilio:

Teléfono:

Email:

Contraseña:

Confirmar Contraseña:

Crear Cuenta

[¿Ya tienes cuenta? Inicia Sesión](#)

Bienvenido a la Biblioteca Virtual LibConnect

Tipo de acceso:

Email:

Contraseña:

Iniciar Sesión

Donde cada libro encuentra a su lector ❤

LibConnect

Principal
Préstamos
Devoluciones
Usuarios
Libros
Reportes

Administración/Control/Biblioteca

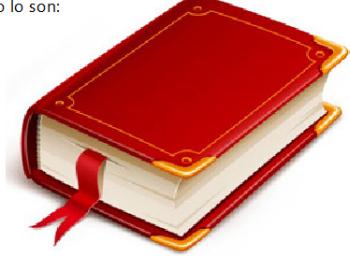
Hoy es viernes 24 de octubre de 2025

Bienvenido

Sistema de Gestión para Biblioteca Pública. Controle y administre de forma óptima y fácil el flujo de préstamos y devoluciones de Libros.

Esta herramienta le permitirá llevar un control completo y detallado de su Biblioteca, tendrá acceso a herramientas especiales para tareas específicas, como lo son:

- Préstamos
- Devoluciones
- Registro de Usuarios y Libros Nuevos
- Edición de Usuarios y Libros existentes
- Eliminar todo tipo de Registros
- Sección de Reportes de acciones en el sistema



Donde cada libro encuentra a su lector ❤

LibConnect

Principal
Préstamos
Devoluciones
Usuarios
Libros
Reportes

Administración/Control/Biblioteca

Hoy es viernes 24 de octubre de 2025



Nombre Usuario

Título Libro

Prestar

LibConnect

Donde cada libro encuentra a su lector ❤

Administración/Control/Biblioteca

Hoy es viernes 24 de octubre de 2025

Devolución de Libro

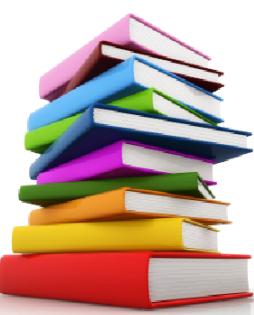
Nombre Usuario

Ingrese el nombre del usuario.

Título Libro

Ingrese el título del Libro a devolver.

Devolver



LibConnect

Donde cada libro encuentra a su lector ❤

Administración/Control/Biblioteca

Hoy es viernes 24 de octubre de 2025

Usuarios

Ingrese el nombre de usuario a buscar.

Buscar

ID	Nombre	Apellido P.	Apellido M.	Domicilio	Teléfono	Correo
1	Admin	Sistema	LibConnect	Oficina Central	999999999	admin@libconnect...
2	Juan	Pérez	García	Av. Principal 123	987654321	juan.perez@email...
3	Maria	López	Martínez	Calle Secundaria 4...	912345678	maria.lopez@email...
4	diego	univeros	moreno	chica	955177259	diego@gmail.com
5	Admin	Sistema	Principal	Oficina Central	992255889	admin@biblioteca...
7	Alejandro	Ramos	Soto	Av. Huancavelica	901-234-567	Alejo@gmail.com
8	Beatriz	Núñez	Vargas	Calle San Martín	998-303-404	bea@gmail.com
9	Camilo	Paredes	López	Jr. Ancash 345,	923-456-789	cam@gamil.com
10	David	Huamán	Torres	Urb. La Merced 10	934-567-890	david@gmail.com
11	Elena	Inocente	Ugarte	Av. Giraldéz 678,	945-678-901	elenaiu@gmail.co...
12	eduardo	quispe	yupanqui	chica	963852147	quispe@gmail.com

Nuevo **Editar** **Borrar**

Donde cada libro encuentra a su lector ❤

LibConnect

Administración/Control/Biblioteca

Hoy es viernes 24 de octubre de 2025

Libros

Ingrese el título del libro a buscar.

ID	Título	Fecha de ...	Autor	Categoría	Edición	Idioma	Páginas	Descripción	Ejemplares...	Stock	Disponibl...
1	Cien Año...	1967	Gabriel ...	Ficción	Editorial ...	Español	471	Obra ma...	5	5	5
2	1984	1949	George ...	Distopía	Secker a...	Inglés	328	Novela d...	4	4	3
3	El Principi...	1943	Antoine ...	Infantil	Reynal &...	Francés	96	Clásico in...	6	6	6
4	Don Quij...	1605	Miguel d...	Novela	Edición C...	Español	863	Las aven...	12	12	14
5	El Secret...	2023-05...	Laura Ríos	Fantasía	Ediciones...	Español	450	Una ave...	50	50	45
6	Program...	2022-11...	Ricardo S...	Tecnolog...	Tech Press	Español	780	Guía co...	120	120	115
7	Matar a ...	1960-07...	Harper L...	Novela C...	Edición E...	Francés	96	Un cuent...	450	450	440
8	El Código...	2003-03...	Dan Bro...	Thriller/...	Robert L...	Inglés	454	Un simb...	260	260	255
9	Harry Po...	1997-06...	J.K. Rowlin...	Fantasía ...	Saga 1	Inglés	223	El inicio d...	500	500	480
10	El Alquim...	1988-01...	Paulo Co...	Ficción Fi...	Edición B...	Portugués	208	Un joven...	350	350	340
11	La Odisea	700 a.C. (...)	Homero	Épica Clá...	Edición G...	Griego A...	544	El viaje d...	70	70	65
12	Los Jueg...	2009-09...	Suzanne	Juvenil/D...	Trilogía 1	Inglés	274	Los Jueg...	400	400	380

Buscar

Nuevo **Editar** **Borrar**

Donde cada libro encuentra a su lector ❤

LibConnect

Administración/Control/Biblioteca

Hoy es viernes 24 de octubre de 2025

Reportes

User ID	Book ID	Fecha de Salida	Fecha de Entrega
diego untiveros moreno	1984	24-10-2025	
diego untiveros moreno	Cien Años de Soledad	24-10-2025	24-10-2025
diego untiveros moreno	Cien Años de Soledad	23-10-2025	23-10-2025
diego untiveros moreno	El Principito	23-10-2025	23-10-2025

Actualizar

Script del Login

```
package com.mycompany.ilib;

import
com.formdev.flatlaf.intellijthemes.m
aterialthemeuilite.FlatMaterialLighte
rIJTheme;

import
com.mycompany.ilib.DAOUsersImpl;

import
com.mycompany.interfaces.DAOUse
rs;

import
com.mycompany.models.Users;

import
com.mycompany.ilib.Dashboard;

import javax.swing.*;
import
javax.swing.border.EmptyBorder;
import java.awt.*;
```

```
import  
java.awt.event.MouseAdapter;  
import java.awt.event.MouseEvent;  
  
/**  
 * CLASE PRINCIPAL - Pantalla de  
Login y Registro  
* Esta es la primera ventana que se  
muestra al iniciar la aplicación  
*/  
  
public class MainApp extends JFrame  
{  
  
    // Componentes de UI  
    private JTextField emailTxt;  
    private JPasswordField  
passwordField;  
    private JLabel errorLabel;  
    private JButton primaryButton;  
    private JButton secondaryButton;  
    private JLabel titleLabel;  
  
    // Campos para registro
```

```
private JLabel nameLabel,  
lastNamePLabel, lastNameMLabel,  
domicilioLabel, phoneLabel;  
  
private JTextField nameTxt,  
lastNamePTxt, lastNameMTxt,  
domicilioTxt, phoneTxt;  
  
private JPasswordField  
confirmPasswordField;  
  
private JLabel  
confirmPasswordLabel;  
  
// Paneles  
  
private JPanel cardPanel;  
  
private JPanel loginPanel;  
  
private JPanel registerPanel;  
  
private CardLayout cardLayout;  
  
// ComboBox para seleccionar rol  
  
private JComboBox<String>  
roleComboBox;  
  
private final String LOGIN_PANEL =  
"Login";
```

```
private final String  
REGISTER_PANEL = "Register";  
  
public MainApp() {  
    initComponentsManual();  
    configureFrame();  
    cardLayout.show(cardPanel,  
LOGIN_PANEL);  
  
    assignButtonReferences(loginPanel);  
}  
  
private void configureFrame() {  
    setTitle("LibConnect - Sistema  
de Gestión Bibliotecaria");  
  
    setDefaultCloseOperation(JFrame.EXIT_ON_  
IT_ON_CLOSE);  
    setResizable(false);  
    pack();  
    setLocationRelativeTo(null);  
}  
}
```

```
private void  
  
initComponentsManual() {  
  
    setLayout(new BorderLayout(0,  
0));  
  
    // Panel Superior  
  
    JPanel topPanel = new  
JPanel(new  
FlowLayout(FlowLayout.CENTER));  
  
    topPanel.setBackground(new  
Color(18, 90, 173)); // ★ Azul de la  
  
aplicación  
  
    titleLabel = new  
JLabel("Bienvenido a la Biblioteca  
Virtual LibConnect");  
  
    titleLabel.setFont(new  
Font("Segoe UI", Font.BOLD, 28));  
  
    titleLabel.setForeground(Color.WHIT  
E); // ★ Texto blanco para  
  
contraste
```

```
titleLabel.setBorder(new  
EmptyBorder(25, 0, 15, 0));  
  
topPanel.add(titleLabel);  
  
add(topPanel,  
  
BorderLayout.NORTH);  
  
  
  
// Panel Central  
  
cardLayout = new CardLayout();  
  
cardPanel = new  
JPanel(cardLayout);  
  
  
  
cardPanel.setBackground(Color.WHI  
TE);  
  
cardPanel.setBorder(new  
EmptyBorder(20, 60, 20, 60));  
  
  
  
loginPanel = createLoginPanel();  
  
cardPanel.add(loginPanel,  
LOGIN_PANEL);  
  
  
  
registerPanel =  
createRegisterPanel();
```

```
        cardPanel.add(registerPanel,  
REGISTER_PANEL);  
  
        add(cardPanel,  
BorderLayout.CENTER);  
  
    // Panel Inferior  
  
    JPanel bottomPanel = new  
JPanel(new  
FlowLayout(FlowLayout.CENTER));  
  
bottomPanel.setBackground(Color.W  
HITE);  
  
errorLabel = new JLabel(" ");  
  
errorLabel.setForeground(Color.RED)  
;  
errorLabel.setFont(new  
Font("Segoe UI", Font.PLAIN, 12));  
  
errorLabel.setBorder(new  
EmptyBorder(0, 0, 10, 0));  
  
bottomPanel.add(errorLabel);
```

```
        add(bottomPanel,  
BorderLayout.SOUTH);  
  
    }  
  
    //-----  
    private JPanel createLoginPanel() {  
  
        JPanel panel = new JPanel(new  
        GridBagLayout());  
  
        panel.setOpaque(false);  
  
        GridBagConstraints gbc =  
        createGBC();  
  
        //-----  
        // ComboBox Rol  
  
        JLabel lblRole = new  
        JLabel("Tipo de acceso:");  
  
        lblRole.setFont(new  
        Font("Segoe UI", Font.PLAIN, 14));  
  
        gbc.gridy++;  
        gbc.gridx = 0;  
        gbc.gridwidth = 1;  
        gbc.weightx = 0.3;  
        gbc.fill =  
        GridBagConstraints.NONE;
```

```
gbc.anchor =
GridBagConstraints.WEST;
panel.add(lblRole, gbc);

roleComboBox = new
JComboBox<>(new String[]{""
Usuario", " Administrador"});
roleComboBox.setFont(new
Font("Segoe UI", Font.PLAIN, 14));

roleComboBox.setPreferredSize(new
Dimension(250, 38));
gbc.gridx = 1;
gbc.weightx = 0.7;
gbc.fill =
GridBagConstraints.HORIZONTAL;
panel.add(roleComboBox, gbc);

// Email
addField(panel, gbc, "Email:",
emailTxt = new JTextField(25));

// Contraseña
```

```
addField(panel, gbc,  
"Contraseña:", passwordField = new  
JPasswordField());  
  
// Botón Login  
JButton btnLogin =  
createPrimaryButton("Iniciar  
Sesión");  
gbc.gridx++;  
gbc.gridwidth = 2;  
gbc.insets = new Insets(25, 5, 5,  
5);  
panel.add(btnLogin, gbc);
```

```
// Botón Registro  
JButton btnGoToRegister =  
createSecondaryButton("¿No tienes  
cuenta? Regístrate");  
gbc.gridx++;  
gbc.insets = new Insets(10, 5, 5,  
5);  
panel.add(btnGoToRegister,  
gbc);
```

```
// Mostrar/ocultar registro  
según rol  
  
roleComboBox.addActionListener(e -> {  
    String selected = roleComboBox.getSelectedItem().toString();  
  
    btnGoToRegister.setVisible(selected.contains("Usuario"));  
});  
  
btnLogin.addActionListener(e -> performLogin());  
  
btnGoToRegister.addActionListener(e -> switchToRegister());  
  
return panel;  
}
```

```
private JPanel  
  
createRegisterPanel() {  
  
    JPanel panel = new JPanel(new  
        GridBagLayout());  
  
    panel.setOpaque(false);  
  
    GridBagConstraints gbc =  
        createGBC();  
  
   addField(panel, gbc, nameLabel  
        = new JLabel("Nombre:"), nameTxt =  
        new JTextField(25));  
  
   addField(panel, gbc,  
  
lastNamePLabel = new  
    JLabel("Apellido Paterno:"),  
  
lastNamePTxt = new JTextField());  
  
   addField(panel, gbc,  
  
lastNameMLabel = new  
    JLabel("Apellido Materno:"),  
  
lastNameMTxt = new JTextField());  
  
   addField(panel, gbc,  
  
domicilioLabel = new  
    JLabel("Domicilio:"), domicilioTxt =  
    new JTextField()));
```

```
addField(panel, gbc, phoneLabel  
= new JLabel("Teléfono:"), phoneTxt  
= new JTextField());
```

```
JTextField registerEmailTxt =  
new JTextField();  
addField(panel, gbc, new  
JLabel("Email:"), registerEmailTxt);
```

```
JPasswordField  
registerPasswordField = new  
JPasswordField();  
addField(panel, gbc, new  
JLabel("Contraseña:"),  
registerPasswordField);
```

```
addField(panel, gbc,  
confirmPasswordField = new  
JLabel("Confirmar Contraseña:"),  
confirmPasswordField = new  
JPasswordField());
```

```
JButton btnRegister =  
  
createPrimaryButton("Crear  
Cuenta");  
  
gbc.gridx++;  
  
gbc.gridwidth = 2;  
  
gbc.insets = new Insets(25, 5, 5,  
5);  
  
panel.add(btnRegister, gbc);
```

```
JButton btnGoToLogin =  
  
createSecondaryButton("¿Ya tienes  
cuenta? Inicia Sesión");  
  
gbc.gridx++;  
  
gbc.insets = new Insets(10, 5, 5,  
5);  
  
panel.add(btnGoToLogin, gbc);
```

```
btnRegister.addActionListener(e  
-> performRegister(  
    nameTxt.getText(),  
    lastNamePTxt.getText(),  
    lastNameMTxt.getText(),
```

```
        domicilioTxt.getText(),  
  
        phoneTxt.getText(),  
  
        registerEmailTxt.getText(),  
  
        new  
  
        String(registerPasswordField.getPass  
  
word()),  
  
        new  
  
        String(confirmPasswordField.getPass  
  
word())  
    );
```

```
btnGoToLogin.addActionListener(e -  
> switchToLogin());
```

```
    return panel;
```

```
}
```

```
private GridBagConstraints  
createGBC() {  
  
    GridBagConstraints gbc = new  
    GridBagConstraints();  
  
    gbc.gridx = 0;  
  
    gbc.gridy = -1;
```

```
    gbc.anchor =
    GridBagConstraints.WEST;
    gbc.fill =
    GridBagConstraints.HORIZONTAL;
    gbc.insets = new Insets(8, 5, 8,
    5);
    gbc.weightx = 1.0;
    return gbc;
}
```

```
private void addField(JPanel panel,
GridBagConstraints gbc, String
labelText, JComponent field) {
    JLabel label = new
JLabel(labelText);
    addField(panel, gbc, label, field);
}
```

```
private void addField(JPanel panel,
GridBagConstraints gbc, JLabel label,
JComponent field) {
    label.setFont(new Font("Segoe
UI", Font.PLAIN, 14));
```

```
gbc.gridx++;
gbc.gridy = 0;
gbc.gridwidth = 1;
gbc.weightx = 0.3;
gbc.fill =
GridBagConstraints.NONE;
gbc.anchor =
GridBagConstraints.WEST;
panel.add(label, gbc);

field.setFont(new Font("Segoe
UI", Font.PLAIN, 14));
if (field instanceof JTextField ||

field instanceof JPasswordField) {
    field.setBorder(BorderFactory.create
CompoundBorder(
    BorderFactory.createLineBorder(new
Color(180, 180, 180)),
    BorderFactory.createEmptyBorder(8,
10, 8, 10)
)
```

```
        });

        field.setPreferredSize(new Dimension(field.getPreferredSize().width, 38));

    }

    gbc.gridx = 1;
    gbc.weightx = 0.7;
    gbc.fill =
        GridBagConstraints.HORIZONTAL;

    panel.add(field, gbc);

}
```

```
private JButton

createPrimaryButton(String text) {

    JButton button = new
        JButton(text);

    button.setFont(new
        Font("Segoe UI", Font.BOLD, 14));

    button.setBackground(new
        Color(18, 90, 173));

    button.setForeground(Color.WHITE);
```

```
button.setCursor(new  
Cursor(Cursor.HAND_CURSOR));  
  
button.setFocusPainted(false);  
  
button.setBorder(new  
EmptyBorder(12, 20, 12, 20));  
  
button.setPreferredSize(new  
Dimension(button.getPreferredSize()  
.width, 42));
```

```
Color originalColor =  
  
button.getBackground();  
  
Color hoverColor =  
  
originalColor.darker();  
  
button.addMouseListener(new  
MouseAdapter() {
```

```
    @Override  
  
    public void  
  
    mouseEntered(MouseEvent e) {
```

```
        button.setBackground(hoverColor);  
  
    }  
  
    @Override
```

```
public void  
mouseExited(MouseEvent e) {  
  
button.setBackground(originalColor)  
;  
}  
});  
  
return button;  
}  
  
}
```

```
private JButton  
createSecondaryButton(String text) {  
  
JButton button = new  
JButton(text);  
  
button.setFont(new  
Font("Segoe UI", Font.PLAIN, 12));  
  
button.setForeground(new  
Color(18, 90, 173));  
  
button.setCursor(new  
Cursor(Cursor.HAND_CURSOR));  
  
button.setFocusPainted(false);  
  
button.setBorderPainted(false);
```

```
button.setContentAreaFilled(false);

button.addMouseListener(new
MouseAdapter() {

@Override
public void
mouseEntered(MouseEvent e) {

button.setText("<html><u>" +
+ text + "</u></html>");

}
@Override
public void
mouseExited(MouseEvent e) {

button.setText(text);

}
});
```

return button;

}

```
private void switchToRegister() {

titleLabel.setText("Crear Nueva
Cuenta");

errorLabel.setText(" ");
```

```
        cardLayout.show(cardPanel,  
REGISTER_PANEL);  
  
assignButtonReferences(registerPan  
el);  
pack();  
setLocationRelativeTo(null);  
}
```

```
private void switchToLogin() {  
    titleLabel.setText("Bienvenido a  
LibConnect");  
    errorLabel.setText(" ");  
    cardLayout.show(cardPanel,  
LOGIN_PANEL);
```

```
assignButtonReferences(loginPanel);  
pack();  
setLocationRelativeTo(null);  
}
```

```
private void assignButtonReferences(JPanel activePanel) {  
    for (Component comp : activePanel.getComponents()) {  
        if (comp instanceof JButton) {  
            JButton btn = (JButton) comp;  
            if ("Iniciar Sesión".equals(btn.getText()) ||  
                "Crear Cuenta".equals(btn.getText())) {  
                primaryButton = btn;  
            } else if (btn.getText().contains("¿")) {  
                secondaryButton = btn;  
            }  
        }  
    }  
  
    private void performLogin() {
```

```
String email =
emailTxt.getText().trim();

String password = new
String(passwordField.getPassword())

;

String selectedRole =
roleComboBox.getSelectedItem().toS
tring();

errorLabel.setText(" ");

if (email.isEmpty() ||

password.isEmpty()) {
    errorLabel.setText("Debe
ingresar email y contraseña.");
    return;
}

try {
    DAOUsers daoUsers = new
    DAOUsersImpl();
    Users user =
    daoUsers.loginUser(email,
password);
```

```
if (user != null) {  
    // Verificar que el rol  
    // coincida  
  
    boolean isAdmin =  
        selectedRole.contains("Administrado  
r");  
  
    boolean userIsAdmin =  
        "admin".equals(user.getRole());  
  
    if (isAdmin &&  
        !userIsAdmin) {  
        errorLabel.setText("No  
tienes permisos de administrador.");  
        return;  
    } else if (!isAdmin &&  
        userIsAdmin) {  
  
        errorLabel.setText("Debes iniciar  
sesión como administrador.");  
        return;  
    }  
}
```

```
//  LOGIN EXITOSO -
```

Abrir Dashboard con el rol

```
this.dispose(); // Cierra la
```

ventana de login

```
Dashboard dashboard =
```

```
new Dashboard(user.getRole());
```

```
dashboard.setVisible(true);
```

```
} else {
```

```
errorLabel.setText("Email o  
contraseña incorrectos.");
```

```
}
```

```
} catch (Exception e) {
```

```
errorLabel.setText("Error al  
conectar. Intente más tarde.");
```

```
System.err.println("Error en
```

```
Login: " + e.getMessage());
```

```
e.printStackTrace();
```

```
}
```

```
}
```

```
private void  
performRegister(String name, String  
lastNameP, String lastNameM, String  
domicilio,  
String phone,  
String email, String password, String  
confirmPassword) {  
  
    errorLabel.setText(" ");  
  
    if (name.isEmpty() ||  
lastNameP.isEmpty() ||  
lastNameM.isEmpty() ||  
domicilio.isEmpty() ||  
phone.isEmpty() ||  
email.isEmpty() ||  
password.isEmpty() ||  
confirmPassword.isEmpty()) {  
  
        errorLabel.setText("Debe  
llenar todos los campos de  
registro.");  
  
        return;  
    }  
}
```

```
    if  
  
        (!password.equals(confirmPassword)  
  
    ) {  
  
        errorLabel.setText("Las  
contraseñas no coinciden.");  
  
        return;  
  
    }  
  
}
```

```
if (!email.matches("^[\w-  
\\\\.]+@[\\w-]+\\.[\\w-]{2,4}$")) {  
  
    errorLabel.setText("Formato  
de correo electrónico inválido.");  
  
    return;  
  
}  
  
}
```

```
Users newUser = new Users();  
  
newUser.setName(name);  
  
newUser.setLast_name_p(lastName  
P);
```

```
newUser.setLast_name_m(lastName  
M);
```

```
newUser.setDomicilio(domicilio);

newUser.setTel(phone);

newUser.setEmail(email);

newUser.setPassword(password);

newUser.setRole("user"); // Por
```

defecto usuario normal

```
try {

    DAOUsers daoUsers = new

    DAOUsersImpl();

    daoUsers.registrar(newUser);

    JOptionPane.showMessageDialog(th

    is,

    "¡Cuenta creada

    exitosamente!\nAhora puedes

    iniciar sesión con:\nEmail: " + email,

    "Registro Exitoso",
```

```
JOptionPane.INFORMATION_MESSAGE  
GE);  
switchToLogin();
```

```
} catch (Exception e) {  
    if (e.getMessage() != null &&  
        e.getMessage().contains("El correo  
ya está registrado")) {  
        errorLabel.setText("Este  
correo electrónico ya está  
registrado.");  
    } else {  
        errorLabel.setText("Error al  
registrar el usuario. Intente  
nuevamente.");  
        System.err.println("Error en  
Registro: " + e.getMessage());  
        e.printStackTrace();  
    }  
}
```

// ★ MÉTODO MAIN - PUNTO DE
ENTRADA DE LA APLICACIÓN

```
public static void main(String  
args[]) {  
  
    try {  
  
        FlatMaterialLighterIJTheme.setup();  
  
    } catch(Exception ex) {  
  
        System.err.println("Fallo al  
inicializar LaF");  
  
        try {  
  
            UIManager.setLookAndFeel(UIMana  
ger.getSystemLookAndFeelNam  
e());  
  
        } catch (Exception e) {  
  
            System.err.println("No se  
pudo establecer LookAndFeel por  
defecto.");  
  
        }  
  
    }  
}
```

```
SwingUtilities.invokeLater(() -> {  
    new  
    MainApp().setVisible(true);  
});  
}  
}
```

Script de la Base de Datos

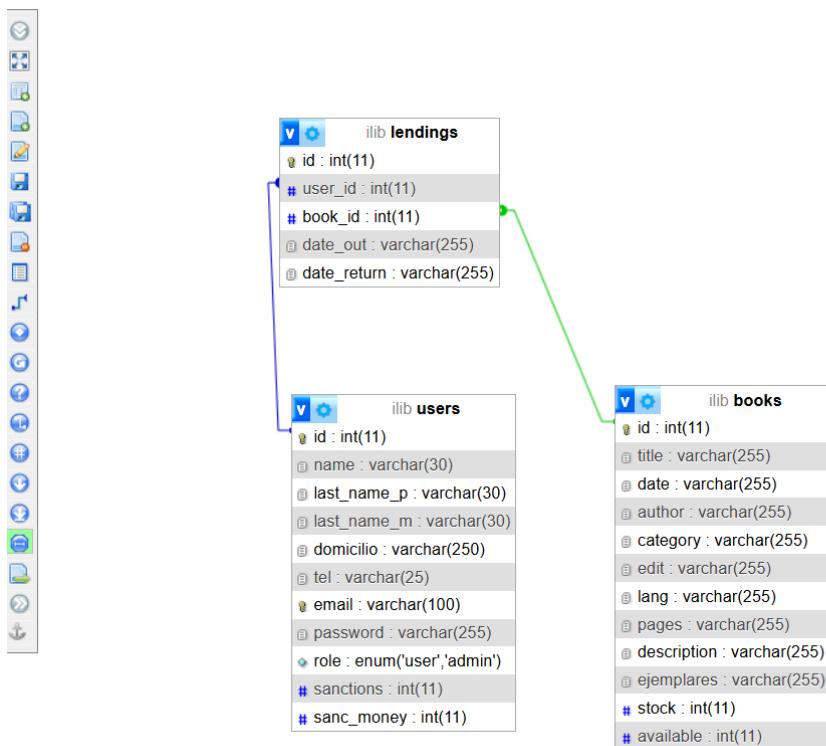
```
-- Crear base de datos  
CREATE DATABASE iLib; USE  
iLib;  
  
-- Tabla de usuarios CREATE TABLE users (  
id INT AUTO_INCREMENT PRIMARY KEY,  
username VARCHAR(50) NOT NULL,  
password VARCHAR(100) NOT NULL,  
fullname VARCHAR(100), email  
VARCHAR(100), role VARCHAR(20)  
DEFAULT 'user'  
);  
  
-- Tabla de libros  
CREATE TABLE books ( id INT  
AUTO_INCREMENT PRIMARY KEY, title  
VARCHAR(150) NOT NULL, author  
VARCHAR(100), category VARCHAR(50),  
year INT,  
available BOOLEAN DEFAULT TRUE  
);  
  
-- Tabla de préstamos  
CREATE TABLE lendings ( id INT AUTO_INCREMENT PRIMARY KEY,  
user_id INT NOT NULL, book_id INT NOT NULL, lending_date DATE  
NOT NULL, return_date DATE,  
status VARCHAR(20) DEFAULT 'Prestado',
```

```

FOREIGN KEY (user_id) REFERENCES users(id),
FOREIGN KEY (book_id) REFERENCES books(id) );

```

Estructura de la Base de Datos



Código Base de Datos

```

-- =====
-- INSTRUCCIONES:
-- 1. Abre phpMyAdmin
-- 2. Crea una base de datos llamada "ilib"
-- 3. Selecciona la base de datos "ilib"
-- 4. Ve a la pestaña "SQL"
-- 5. Pega TODO este código
-- 6. Click en "Continuar"
-- =====

```

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";

-- -----
-- Tabla: books

-- -----
CREATE TABLE IF NOT EXISTS `books` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `title` varchar(255) NOT NULL,
    `date` varchar(255) NOT NULL,
    `author` varchar(255) NOT NULL,
    `category` varchar(255) NOT NULL,
    `edit` varchar(255) NOT NULL,
    `lang` varchar(255) NOT NULL,
    `pages` varchar(255) NOT NULL,
    `description` varchar(255) NOT NULL,
    `ejemplares` varchar(255) NOT NULL,
    `stock` int(11) NOT NULL,
    `available` int(11) NOT NULL,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- -----
-- Tabla: users

-- -----
```

```
CREATE TABLE IF NOT EXISTS `users` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `name` varchar(30) NOT NULL,
    `last_name_p` varchar(30) NOT NULL,
    `last_name_m` varchar(30) NOT NULL,
    `domicilio` varchar(250) DEFAULT NULL,
    `tel` varchar(25) DEFAULT NULL,
    `email` varchar(100) NOT NULL,
    `password` varchar(255) NOT NULL,
    `role` ENUM('user', 'admin') NOT NULL DEFAULT 'user',
    `sanctions` int(11) DEFAULT 0,
    `sanc_money` int(11) DEFAULT 0,
    PRIMARY KEY (`id`),
    UNIQUE KEY `email` (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

--
-- Tabla: lendings
--

```
CREATE TABLE IF NOT EXISTS `lendings` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `user_id` int(11) NOT NULL,
    `book_id` int(11) NOT NULL,
    `date_out` varchar(255) NOT NULL,
    `date_return` varchar(255) DEFAULT NULL,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
-- 
-- Claves Foráneas
-- 

ALTER TABLE `lendings` 
    ADD CONSTRAINT `fk_lendings_user` FOREIGN KEY (`user_id`)
    REFERENCES `users` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
    ADD CONSTRAINT `fk_lendings_book` FOREIGN KEY (`book_id`)
    REFERENCES `books` (`id`) ON DELETE CASCADE ON UPDATE CASCADE;

-- 
-- Datos iniciales
-- 

-- Administrador
INSERT INTO `users` (`name`, `last_name_p`, `last_name_m`,
`domicilio`, `tel`, `email`, `password`, `role`)
VALUES ('Admin', 'Sistema', 'LibConnect', 'Oficina Central',
'999999999', 'admin@libconnect.com', 'admin123', 'admin');

-- Usuarios de prueba
INSERT INTO `users` (`name`, `last_name_p`, `last_name_m`,
`domicilio`, `tel`, `email`, `password`, `role`)
VALUES
('Juan', 'Pérez', 'García', 'Av. Principal 123', '987654321',
'juan.perez@email.com', 'juan123', 'user'),
('María', 'López', 'Martínez', 'Calle Secundaria 456', '912345678',
'maria.lopez@email.com', 'maria123', 'user');

-- Libros de prueba
INSERT INTO `books` (`title`, `date`, `author`, `category`, `edit`,
`lang`, `pages`, `description`, `ejemplares`, `stock`, `available`)
```

VALUES

```
('Cien Años de Soledad', '1967', 'Gabriel García Márquez',
'Ficción', 'Editorial Sudamericana', 'Español', '471', 'Obra maestra
del realismo mágico', '5', 5, 5),
('1984', '1949', 'George Orwell', 'Distopía', 'Secker and Warburg',
'Inglés', '328', 'Novela distópica', '4', 4, 4),
('El Principito', '1943', 'Antoine de Saint-Exupéry', 'Infantil',
'Reynal & Hitchcock', 'Francés', '96', 'Clásico infantil', '6', 6,
6);
```

COMMIT;

-- =====

-- CREDENCIALES DE ACCESO:

--

-- ADMIN:

-- Email: admin@libconnect.com

-- Password: admin123

--

-- USUARIOS:

-- juan.perez@email.com / juan123

-- maria.lopez@email.com / maria123

-- =====

Diagrama del Proyecto

Estructura del Proyecto iLib (Diagrama Textual)

```
iLib/
|
└── └── Source Packages/
    |
    └── └── com.mycompany.db/
```

```
|   |   └ Database.java
# Configuración de conexión a BD
|
|
|   └ com.mycompany.ilib/
# CAPA DE IMPLEMENTACIÓN
|   |   └ DAOBooksImpl.java
# Implementación CRUD de Libros
|   |   └ DAOLendingsImpl.java
# Implementación CRUD de Préstamos
|   |   └ DAOUsersImpl.java
# Implementación CRUD de Usuarios
|   |   └ Dashboard.java
# Ventana principal del sistema
|   |   └ MainApp.java
# ★ PUNTO DE ENTRADA (Login/Registro)
|
|
|   └ com.mycompany.interfaces/
# CAPA DE INTERFACES (DAO)
|   |   └ DAOBooks.java
# Interfaz para operaciones de Libros
|   |   └ DAOLendings.java
# Interfaz para operaciones de
Préstamos
|   |   └ DAOUsers.java
# Interfaz para operaciones de
Usuarios
|
|
|   └ com.mycompany.models/
# CAPA DE MODELOS (Entidades)
|   |   └ Books.java
# Modelo de datos: Libro
|   |   └ Lendings.java
# Modelo de datos: Préstamo
|   |   └ Users.java
# Modelo de datos: Usuario
|
|
|   └ com.mycompany.utils/
# UTILIDADES
```

```

|   |   └ (Vacío actualmente)
|
|   └── com.mycompany.views/
# CAPA DE VISTAS (UI)
|       ├── Books.java
# Panel de gestión de Libros
|       ├── Lendings.java
# Panel de gestión de Préstamos
|       ├── Principal.java
# Vista principal (legacy?)
|       ├── Reports.java
# Panel de reportes
|       ├── Returns.java
# Panel de devoluciones
|       ├── UpBooks.java
# Panel de actualización de Libros
|       ├── UpUsers.java
# Panel de actualización de Usuarios
|       └── Users.java
# Panel de gestión de Usuarios

```

Dependencias (Librerías externas)

```

libs/
└── Dependencies/
# LIBRERÍAS EXTERNAS
    ├── AbsoluteLayout-SNAPSHOT.jar
# Layout manager
    ├── flatlaf-2.4.jar
# Look and Feel moderno
    ├── flatlaf-intellij-themes-
2.4.jar      # Temas adicionales
FlatLaf
    ├── mysql-connector-j-
8.0.33.jar      # Driver JDBC para
MySQL
    └── protobuf-java-3.21.9.jar

```

Arquitectura del Sistema

