

**Programación Laboratorio III:**

**Comisión 3**

**Universidad Tecnológica Nacional de Mar del Plata**

**Docente:** Guillermo Giménez

**Informe TP Software Biblioteca:**

**Integrantes del grupo:**

Durante M.Florencia

Franco Agustín

Mango Eduardo

Padilla Lucas

**Informe sobre el Desarrollo**

**Software para Gestión de Biblioteca**

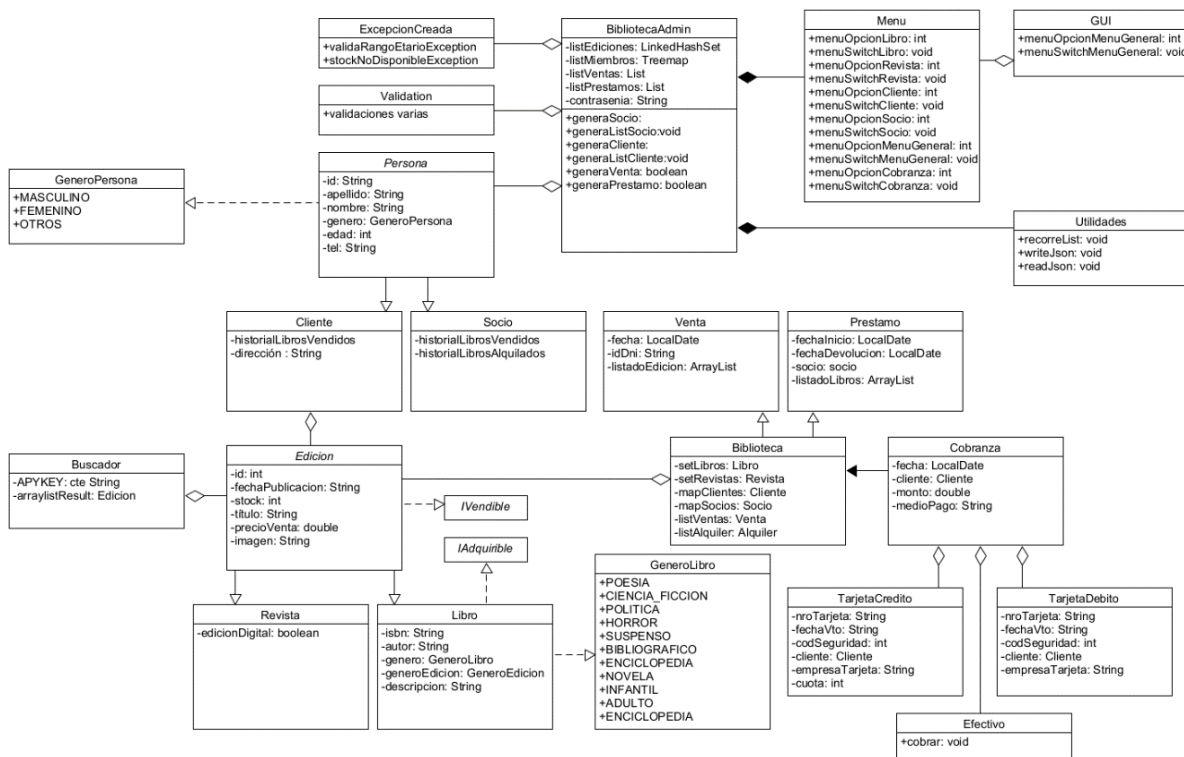


## Introducción

### Propósito:

Se desarrolló una aplicación para poder ser implementada en distintas bibliotecas. Se consensó con los integrantes del grupo la idea a plantear, los aspectos más relevantes, los distintos componentes. Para lo cual primero tuvo que ser implementado un diagrama UML para poder desglosar el software en sus partes principales. Obviamente teniendo en cuenta que este fue un mero bosquejo de lo que va a ser el proyecto en sí, pero que con el tiempo surgieron cambios y hay implementaciones, datos y métodos que se agregaron o quitaron para que todo funcionara de acuerdo a nuestro objetivo.

El objetivo principal es desarrollar una solución tecnológica que permita una gestión eficiente de préstamos, devoluciones, alquileres y venta, catalogación y acceso a la información bibliográfica, para acceder a las ediciones de una biblioteca. Además de los miembros que acceden a esta, como son los Clientes y Socios.



### ***Alcance del Proyecto:***

El proyecto en sí incluye la gestión de libros, revistas, socios, clientes, alquileres y ventas de las distintas ediciones, y su respectivo sistema de cobranza, pagos (efectivo, tarjeta de crédito y de débito).

El objetivo es que sea aplicado por usuarios personal que trabaja en la biblioteca, para el manejo de las personas que acceden a las distintas ediciones, ya sea para comprarlas o alquilarlas, y el sistema de manejo de los cobros. Además, que puedan modificar los datos de los usuarios, listarlos, generar las bajas de ser necesarias. Todo mediante el uso de una interfaz que le facilite la tarea, accediendo de manera simple pero eficaz al manejo de la información.

### ***Requisitos del Sistema***

#### ***Funcionalidades Necesarias:***

- Gestión e implementación de los distintos menús a ser aplicados.
- Registro y gestión de personas, socios y clientes.
- Sistema de alquiler y posible venta para socios.
- Integración con sistemas de cobranza para los casos en los que se realizaran ventas (tarjeta de crédito, débito, efectivo).
- Sistema de venta para clientes.
- Gestión de libros y revistas.
- Utilización de API para obtener información relacionada con libros y revistas.
- Utilización de API Gráfica para obtener visualización de la implementación del programa.
- Sistemas de ABML de las clases implementadas: clientes, socios, libros, revistas.
- Implementación y uso de enumeraciones.
- Implementación y manejo de excepciones para el tratamiento de errores.

- Implementación de interfaces, para el manejo de los alquileres y de las ventas.

## ***Funcionalidades Principales***

### ***Gestión de Libros y Revistas:***

- Catálogo de libros y revistas disponibles.
- Registro de información detallada (título, autor, ISBN, categoría, etc.).
- Búsqueda y filtrado avanzado.

Para instanciar libros y revistas, primero desarrollamos la clase abstracta Edición, de la cual heredan las dos anteriores. En un principio teníamos decidido ingresar cada dato de estos de forma manual. Eso fue así hasta que encontramos la API de Google Libros, la cual ofrece flexibilidad a la hora de hacer búsquedas, ya que necesita de palabras claves en lugar de términos exactos.

De esta forma, el usuario solo tiene que ingresar muy pocos datos, como stock y precio, dado que el resto se cargan automáticamente con la información que devuelve la API (en formato JSON).

La secuencia para agregar una edición es la siguiente:

El usuario accede al menú de libros o al de revistas, según lo que quiera agregar. Realiza una búsqueda, la cual devuelve (o no) una lista de resultados. Todo esto sucede dentro de la clase Buscador, la cual tiene una lista que se borra y carga constantemente con cada búsqueda nueva.

Por último, el usuario decide si agregar alguna de las ediciones devueltas por el buscador, agregando luego unos pocos datos más (genero, stock y precio).

En conclusión, trabajar con la API permite trabajar con rapidez, evitándole al usuario tener que agregar manualmente varios de los atributos de las ediciones como son, título, autor, editorial, fecha, etc.

### ***Gestión de Personas:***

Se implementó una clase Persona abstracta y de la cual van a heredar otras dos clases que son socio y cliente. Esta clase tiene los atributos principales que además heredan sus hijas. Posee un menú, con las opciones correspondientes de generar el alta de algún socio o cliente nuevo, ver el listado de estos y de sus ventas o alquileres, también podemos modificar los datos que poseen.

En ambos casos se utilizó el caso de la baja lógica para no perder la información en la base de nuestra biblioteca, para el caso que estos miembros pudieran ser reactivados dado el caso.

Para los miembros se emplearon treemap, que es una colección de datos ordenada por un dato que en este caso es el idDni, lo que va a permitir que en caso de búsqueda se los encuentre mediante este atributo.

- ***Gestión de Clientes:***

Posee los mismos atributos que su clase padre. Estos además tienen el listado de las compras realizada de las distintas ediciones, libros o revistas.

- ***Gestión de Socios:***

Posee además de los atributos de su clase padre, un campo que es la dirección. Los socios además del listado de compras también poseen un listado de alquileres de libros.

### ***Sistema de Alquiler y Venta:***

El proceso de venta de las distintas ediciones, los libros y las revistas, generado por los distintos clientes de la institución y de alquiler de libros para socios, ha sido implementado, generando un proceso de venta o alquiler según sea el caso, y brindando luego que esta acción pueda ser vista por el personal administrador de la biblioteca.

### ***Sistema de Cobranza:***

El sistema de venta de las ediciones, fue implementado con un sistema de cobranzas, con métodos de pago realizados con tarjeta de crédito, débito, efectivo, para los distintos casos en los que se realizó venta de libros y revistas con los socios/ clientes correspondientes.

### ***Implementación y Desarrollo***

- ***Metodología de Desarrollo:***

El desarrollo del TP fue realizado con Java, y además se empleó una API de la cual se obtuvo la información acerca de los libros y de las revistas. También se empleó GitHub para poder ir trabajando a la par con nuestros compañeros de equipo en distintos temas propuestos para luego poder ir ensamblándolos. De esta manera es como se suele trabajar en las distintas empresas tecnológicas, en grupos interdisciplinarios, fomentando la colaboración y apoyándonos de las fortalezas individuales.

La Programación Orientada a Objetos (POO), paradigma basado en el uso de objetos, fue sumamente útil para poder representar elementos de la realidad como es el objeto en cuestión, con sus métodos y atributos, las clases necesarias para organizarlo y estructurar todo el código necesario para implementar nuestro sistema. En el contexto del desarrollo del software de una biblioteca, la POO fue necesaria para representar de manera clara, los diferentes elementos, aspectos y funcionalidades para que se pueda implementar.

## ***Pruebas y Validaciones***

- ***Estrategia de Pruebas:***

Se agregó un sistema general de validaciones, para que los datos ingresados por el socio fueran los requeridos y no otros, lo que podría ocasionar un comportamiento inesperado del sistema en sí.

Las pruebas se fueron implementando individualmente, para luego poder realizar un testeó final. Observando que todo el ensamble funcionara de acuerdo al objetivo final. Se buscó que no hubiera lugares en el código donde existieran errores que pusieran fin a la ejecución normal del código implementado. Para esto se puso en práctica el uso de excepciones, que lanzaran distintos mensajes informando los errores que podía haber en el caso que se hiciera algo que no fuera lo que se pretendía: Ej. Al esperar que el usuario ingresara un dato numérico y este por error ingresara un carácter.

También se implementó el uso de enumeradores por ejemplo en los casos donde se pretendía que el usuario ingresara las opciones que los desarrolladores querían y no otras, como fue en el caso del atributo de género de las personas (masculino, femenino y otro) y en el caso de los distintos géneros literarios y de las opciones implementadas en el sistema de cobranzas, tarjeta de crédito, débito y efectivo. Esto le brinda al usuario una serie de posibilidades constantes a elegir y los desarrolladores se garantizan de esta manera que estos van a ser los datos que van a obtener de los usuarios, evitando el ingreso de otros, lo que cambiaría el sentido de las respuestas y evitan inconvenientes en el normal funcionamiento del proyecto.

## ***Desafíos y Soluciones:***

Uno de los errores más comunes con el que nos encontramos fue con la herramienta Scanner en el uso de los menús. Que provocaba bucles infinitos o que el código diera la

sensación de no continuar su ejecución, sin lanzar ningún error. Esto surgió prácticamente en todo el TP al tener que ingresar datos. Fue dificultosa la tarea de hallar en qué lugares se debía realizar la limpieza del buffer para que estos problemas se resolvieran satisfactoriamente.

A la hora de realizar búsquedas de ediciones con la API Google, se presentó un problema cuando se usaban los getters de la librería JSON. El compilador arrojaba excepciones JSONException cuando se intentaba guardar los datos en variables. En base a esto descubrimos que no todas las ediciones contenían los mismos datos en el JSON que devuelve la API. Por lo cual se decidió dejar el catch del JSONException vacío, y que directamente la edición si no encontraba en ella, todos los datos necesarios.

Otro de los inconvenientes surgidos fue la implementación del Git y de sus Branch, pero fue muy necesaria esta herramienta para evitar sobrescribir código realizado, lo que fue muy necesario para el ensamble y la implementación del trabajo final.

Para la persistencia de datos se implementó la librería Jackson para poder guardar el proyecto con toda la información que contiene y la implementación en sus distintas clases. Para esto nos enfrentamos con el inconveniente del guardado de las distintas clases abstractas, y de las clases independientes que componen al sistema. Como por ejemplo la biblioteca en sí y además, los listados de socios, clientes, ventas, alquileres, etc.

### ***Interfaz Gráfica:***

En el proyecto, se ha implementado una interfaz gráfica de usuario (GUI) utilizando la biblioteca Swing para Java, complementada con la biblioteca FlatLaf. Swing es una biblioteca de Java ampliamente utilizada para la creación de interfaces gráficas, ofreciendo una gran flexibilidad y control sobre los componentes visuales y la interacción del usuario.

FlatLaf, por su parte, es una biblioteca moderna y ligera que proporciona un conjunto de estilos visuales planos y atractivos para las aplicaciones Swing. Al integrar FlatLaf, el software no solo obtiene un aspecto moderno y profesional, sino que también mejora la experiencia del usuario al ofrecer una interfaz limpia y consistente y nos permite abreviar y optimizar mucho el código utilizado.



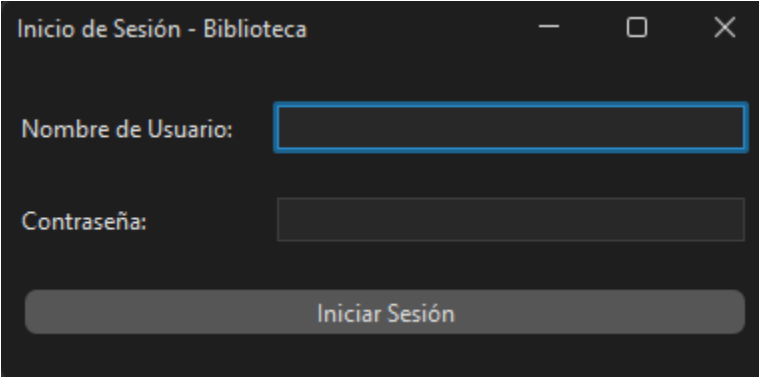
La combinación de Swing y FlatLaf ha permitido desarrollar una interfaz gráfica funcional, facilitando a los usuarios de la biblioteca el acceso y gestión de los recursos disponibles de manera eficiente y agradable. Esta elección tecnológica asegura que el software sea tanto visualmente atractivo como robusto en términos de funcionalidad y rendimiento.

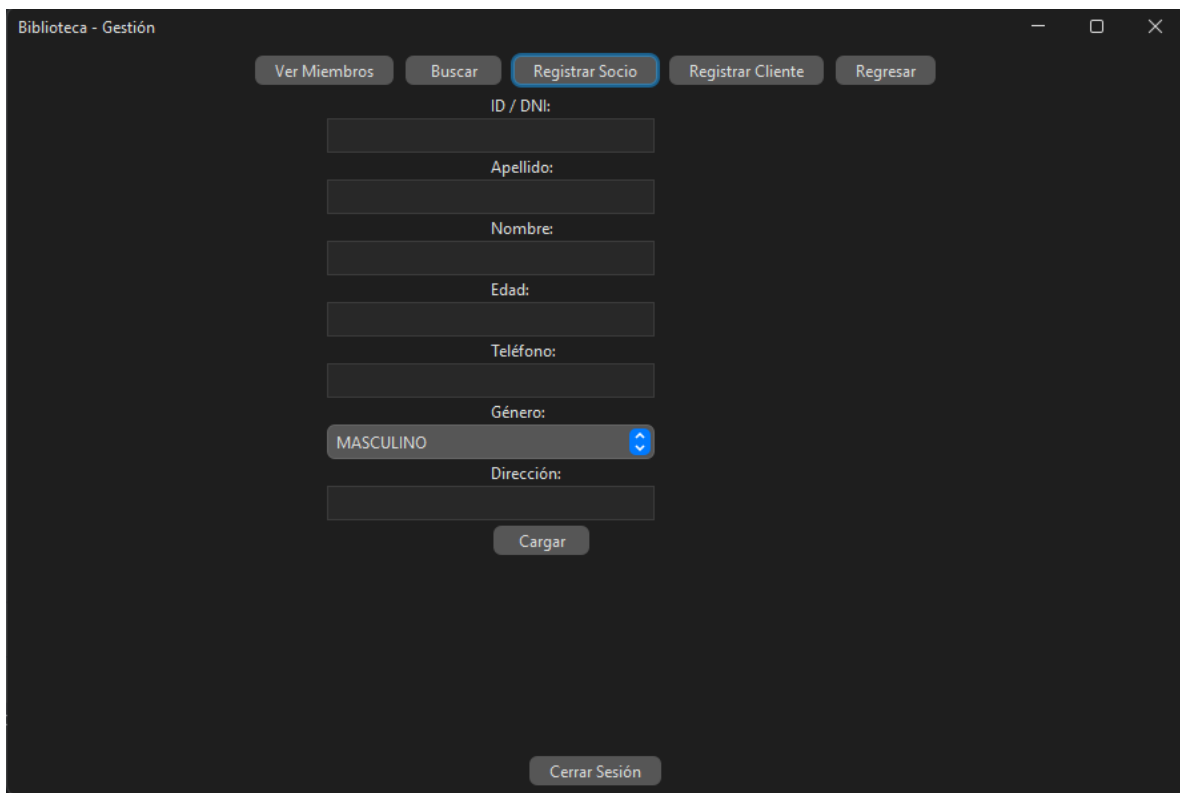
La interfaz gráfica se ha dividido en un menú inicial de inicio de sesión, con un menú principal de sesión iniciada, el cual se divide en submenús de las distintas Ediciones, Libros y Revistas, Miembros involucrados como son los Socios y los Clientes, y las Ventas y Prestamos respectivamente.

Nos permitió crear una experiencia simple e intuitiva, con el objeto de que cualquier persona pueda utilizar el software sin la necesidad de que se tuviera que informar acerca de su funcionamiento. Se optó por una apariencia sencilla pero moderna, con modo oscuro y sin sobrecarga de colores, botones o información, buscando que cada menú sea intuitivo sin que exista una sobrecarga de posibilidades. Se buscó además unificar la apariencia de todos los menús principales, para que ofreciera una experiencia uniforme y simplificada.

**Nombre de usuario:** admin

**Contraseña:** password





## *Consideraciones y conclusiones finales*

Las diversas acciones contenidas en este trabajo demuestran la complejidad de un proyecto de esta índole, como lo fue en este caso una biblioteca. Una vez trabajadas todas las partes individuales del proyecto fue necesario adaptarlo al proyecto general. De esa manera se pudo ver en qué partes surgían los distintos inconvenientes para que esto no sucediera en la presentación. Para esto fue primordial fue el trabajo en equipo para poder cumplir con los requerimientos de la consigna estipulada y adaptarnos a los tiempos estipulados de presentación, para que todo funcionara correctamente, y demuestra también porque las empresas que desarrollan aplicaciones escogen implementan esta metodología.

## *Referencias*

Se recurrió al uso de varias herramientas que fueron de gran ayuda para la realización del proyecto de referencia:

- IntelliJ IDEA
- GitHub (Desktop y terminal)
- Inteligencia artificial (ChatGPT)
- API para la implementación de libros y revistas
- API para la implementación gráfica