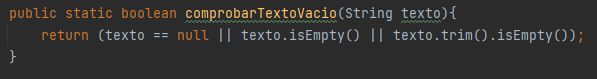
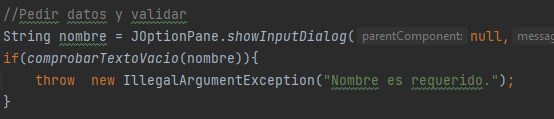
**Alancay, Abel Matías. Actividad 2:**

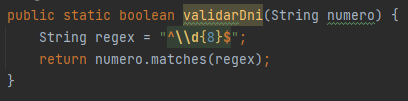
**Validación de datos:**

Una vez ingresado los datos mediante un jOptionPane utilizo dos métodos para validar los datos.

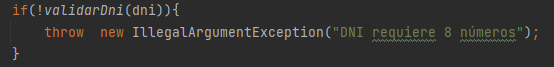
-El método **comprobarTextoVacio**, recibe un String y retorna un boolean que indica si es una cadena vacía o no. Este método lo utilizo en todos los atributos String de las distintas clases.  
  
  
  
En caso que este vacía, en el código arrojo una excepción **illegalArgumentException**, indicando el motivo de error. Dicha excepción es tratada al final de la aplicación.



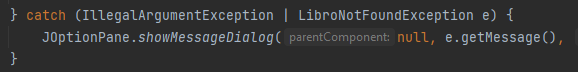
-El método **validarDni** también recibe un String y retorna true, si el String solo contiene 8 dígitos enteros. Para esto utilice regex para comprobar si el texto coincide con una expresión.



En caso que el método retorne falso, en el código arrojo una excepción de **illegalArgumentException**, indicando el motivo de error, que es tratada al final de la aplicación.

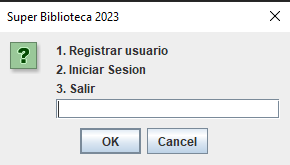


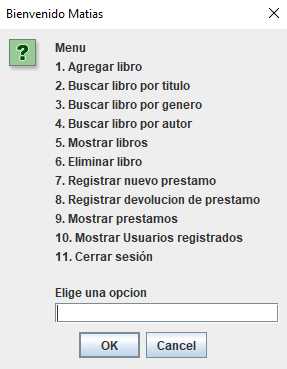
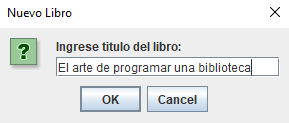
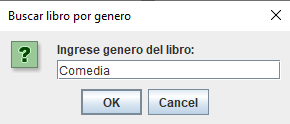
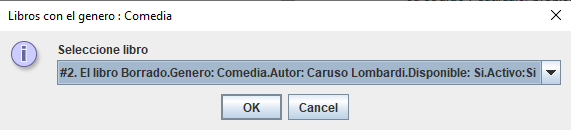
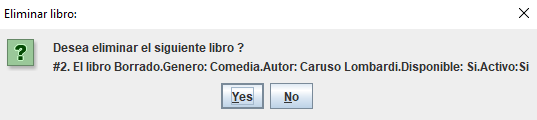
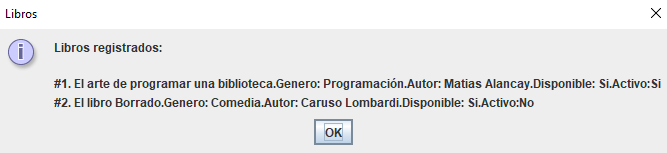
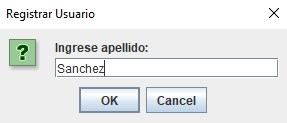
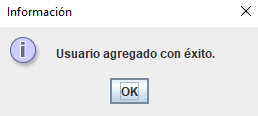
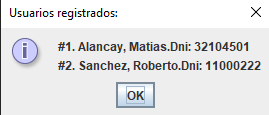
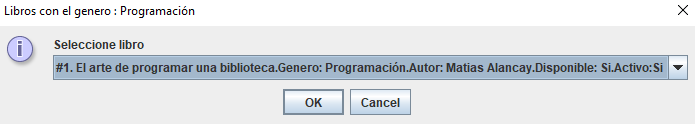
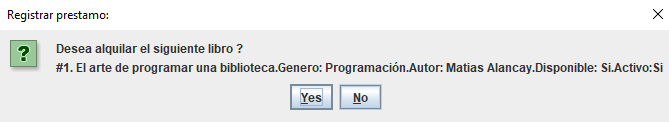
Acá vemos como es capturada la excepción de en caso que los textos estén vacíos o si el dni no cumple con los 8 dígitos.

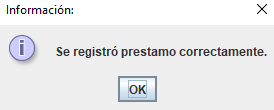
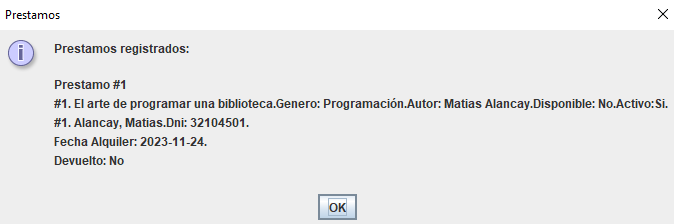


**Descripción del sistema:**

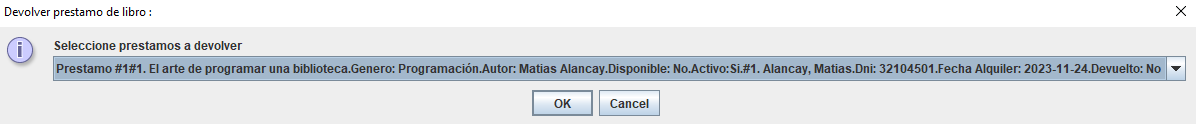
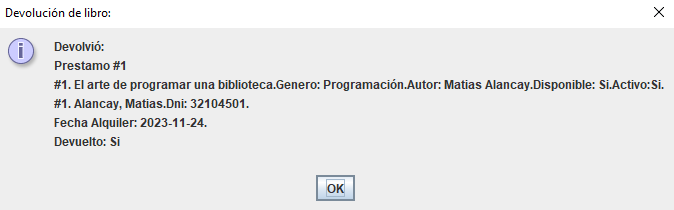
Contamos con las siguientes clases para modelar el sistema pedido:  
Libro, Usuario, préstamo, LibroDAO, UsuarioDAO, PrestamoDAO y Biblioteca.  
  
El sistema cuenta con una clase Biblioteca que contiene como atributos clases que funcionan como si fuesen un “DAO” que manejan una lista para los distintos requisitos funcionales. En Biblioteca encontramos los “DAO” para usuarios, libros y préstamos.  
  
Cada DAO, tiene una lista específica según su funcionalidad.   
Ejemplo LibrosDAO contiene una lista de Libro, en la cual se realizan las distintas funcionalidades requeridas. Como agregar, eliminar, mostrar o consultar un libro por título, género o autor. Y así cada DAO, realiza las distintas acciones que le corresponden. Además, para simplificar algunas acciones sobre las colecciones también se utilizó stream para las búsquedas por ejemplo.  
  
También se utilizó excepciones en el caso de libros y usuarios cuando no se encontraban.  
  
**Implementacion de distintas funcionalidades:**

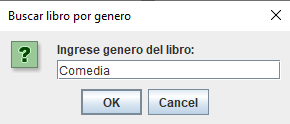
Para poder acceder a las distintas funcionalidades sobre libros y prestamos el usuario debe registrarse o iniciar sesión. Los distintos menús se utilizó un switch para controlar las opciones.  
  
 

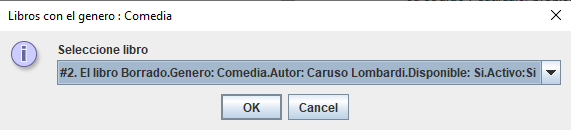
Una vez logueado, puede acceder a las distintas opciones:  
   
  
**Requisitos Funcionales:**  
**1. Gestión de Libros:**   - Los usuarios pueden agregar nuevos libros a la biblioteca.  
En este caso no hay restricciones sobre el libro, ya que puede agregar varios libros del mismo para poder realizar más préstamos.  
La aplicación ira pidiendo los datos requeridos para agregar un libro y validándolo.  
   
  
   - Los usuarios pueden eliminar libros de la biblioteca.  
En el caso de eliminar libro, se implementó baja lógica a través de un campo booleano activo(false, eliminado) en vez de removerlo de la lista. Para mantener los demás datos relacionados y no haya incongruencia de datos.  
  
   
Para eliminar un libro debo antes buscar un libro usando las opciones de búsqueda por autor, titulo o género.   
  
De esta forma podre seleccionar el libro y luego eliminarlo con previa confirmación.  
   
  
Al eliminar un libro, ya no aparecerá en las distintas búsquedas pero si en el mostrar libros que es para poder ver el estado de todos los libros de la biblioteca pero figurara como **Activo:No**  
  
**2. Gestión de Usuarios:**   - Los usuarios pueden registrarse en el sistema.  
Al elegir la opción de registrar usuario, me pedirá los datos requeridos para crear un nuevo usario.  
   
  
Al finalizar correctamente la carga del usuario, veré el siguiente mensaje de confirmación.  
   
  
   
  
Si elijo la opción Mostrar usuarios registrados veré los usuarios creados.  
   
  
   - Los usuarios pueden alquilar libros.  
Para ello primero debemos seleccionar un libro mediante las distintas opciones de búsqueda.  
  
  
Al seleccionar el libro podré elegir luego la opción de registrar nuevo préstamo y confirmar el préstamo. Para poder acceder a todas las funcionalidades se debe iniciar sesión con lo cual el usuario activo es quien está realizando el alquiler del libro.  


- Los usuarios deben devolver los libros  
Mediante la opción registrar devolución de préstamo puedo devolver un libro.  
Al seleccionar esta opción solo me figuraran los préstamos sin devolver para facilitar la elección.

  
Se muestra los distintos atributos propios del préstamo, incluido el atributo Devuelto:NO  
  
Una vez confirmada la devolución del libro, cambiara el estado del atributo Devuelto a SI.  
  


**4. Consultas y Búsquedas:**   - Los usuarios pueden buscar libros en la biblioteca por título, autor o género.  
Las búsquedas solo incluyen los libros que se encuentran disponibles, es decir sin alquilar o que ya fueron devueltos.  
  
 

Las distintas búsquedas son igual solo cambia el criterio de búsqueda.  


**Problemas afrontados durante el desarrollo de la aplicación y resolución:**  
- Uso de **expresiones regulares** en java (Regex) para comprobar que se ingrese 8 dígitos para el dni.  
  
**Resolución**: búsqueda en la web de distintos ejemplos y puesta en práctica por separado para que su prueba sea sencilla.

- Uso adicional de **jOptionPane**, no solo para mostrar mensajes sino para poder confirmar una acción o para mostrar una lista de valores de la cual puede elegir uno. El uso de una lista de valores facilitó el uso de objetos y disminuyo validaciones, ya que estaba obligado a elegir de una lista.  
  
**Resolución**: consultar documentación oficial de Java. Allí se encuentran los distintos usos del jOptionPane.  
  
- Uso de **stream**, para simplificar acciones de búsqueda en colecciones.