# **Identificación de Clases, Atributos y Métodos**

**Clases Necesarias:**

BaseUser:

* Atributos:
  + username (String)
  + password (String)
  + isPremium (boolean)
* Métodos:
  + BaseUser(String username, String password): Constructor

Book:

* Atributos:
  + name (String)
  + ISBN (String)
  + type (String)
  + price (double)
* Métodos:
  + Book(String name, String ISBN, String type, double price): Constructor
  + getName(): String: Obtiene el nombre del libro
  + getISBN(): String: Obtiene el número ISBN del libro
  + getPrice(): double: Obtiene el precio del libro
  + toString(): String: Representación en cadena del objeto Book

Loan:

* Atributos:
  + user (User)
  + book (Book)
  + magazine (Magazine)
  + startDate (Date)
  + dueDate (Date)
* Métodos:
  + Loan(User user, Book book, Date startDate, Date dueDate): Constructor para préstamo de libro
  + Loan(User user, Magazine magazine, Date startDate, Date dueDate): Constructor para préstamo de revista
  + getUser(): User: Obtiene el usuario asociado al préstamo
  + getBook(): Book: Obtiene el libro asociado al préstamo
  + getMagazine(): Magazine: Obtiene la revista asociada al préstamo
  + getStartDate(): Date: Obtiene la fecha de inicio del préstamo
  + getDueDate(): Date: Obtiene la fecha de vencimiento del préstamo

Magazine:

* Atributos:
  + name (String)
  + ISSN (String)
  + type (String)
  + price (double)
* Métodos:
  + Magazine(String name, String ISSN, String type, double price): Constructor
  + getName(): String: Obtiene el nombre de la revista
  + getISSN(): String: Obtiene el número ISSN de la revista
  + getPrice(): double: Obtiene el precio de la revista
  + toString(): String: Representación en cadena del objeto Magazine

User:

* Atributos:
  + username (String)
  + password (String)
  + isPremium (boolean)
  + selectedItems (ArrayList<Item>)
* Métodos:
  + User(String username, String password, boolean isPremium): Constructor
  + getUsername(): String: Obtiene el nombre de usuario
  + getPassword(): String: Obtiene la contraseña
  + isPremium(): boolean: Verifica si el usuario es premium
  + setPremium(boolean premium): Cambia el estado de un usuario base a premium
  + toString(): String: Representación en cadena del objeto User
  + getSelectedItems(): ArrayList\<Item>: Obtiene la lista de artículos seleccionados
  + clearSelection(): Vacía la lista de selección

LibraryManager:

* Atributos:
  + availableBooks (ArrayList<Book>)
  + availableMagazines (ArrayList<Magazine>)
* Métodos:
  + LibraryManager(): Constructor
  + getAvailableBooks(): ArrayList\<Book>: Obtiene la lista de libros disponibles
  + getAvailableMagazines(): ArrayList\<Magazine>: Obtiene la lista de revistas disponibles
  + initializeLibrary(): Inicializa la biblioteca con algunos libros y revistas
  + displayLibrary(): Muestra los recursos de la biblioteca
  + findBook(ArrayList\<Book> books, String name): Book: Encuentra un libro por nombre
  + findMagazine(ArrayList\<Magazine> magazines, String name): Magazine: Encuentra una revista por nombre

LoanManager:

* Atributos:
  + loans (ArrayList<Loan>)
  + libraryManager (LibraryManager)
* Métodos:
  + LoanManager(LibraryManager libraryManager): Constructor
  + processLoan(User user, ArrayList\<Book> availableBooks, ArrayList\<Magazine> availableMagazines, Scanner scanner): Procesa el préstamo
  + borrow(User user, Book book, Scanner scanner): Realiza el préstamo de un libro
  + borrow(User user, Magazine magazine, Scanner scanner): Realiza el préstamo de una revista
  + printLoanList(User user): Imprime la lista de préstamos de un usuario
  + countLoans(User user, Class<?> resourceType): int: Cuenta los préstamos de un usuario por tipo de recurso
  + saveLoansToCSV(): Guarda la información de los préstamos en un archivo CSV
  + addDays(Date date, int days): Date: Añade días a una fecha
  + formatDate(Date date): String: Formatea una fecha como cadena
  + clearSelection(User user): Vacía la lista de selección de un usuario
  + readIntegerInput(Scanner scanner): int: Lee un número entero desde la entrada estándar

PremiumUser:

* Atributos:
  + Hereda de User
* Métodos:
  + PremiumUser(String username, String password): Constructor de PremiumUser

BanerInterface:

* Métodos:
  + showOptions(): Muestra las opciones

ExportArchiveInterface:

* Métodos:
  + export(): Exporta datos

LoadInterface:

* Métodos:
  + showListResources(): Muestra la lista de recursos

ReadArchiveInterface:

* Métodos:
  + read(): Lee datos

SelectionInterface:

* Métodos:
  + addResource(User user, Item item): Añade un recurso a la selección del usuario
  + clearResource(User user): Vacía la selección del usuario
  + loanMode(User user, Scanner scanner): Cambia el modo de préstamo

**Diagrama UML:**

**Imagen que contiene Diagrama

Descripción generada automáticamente**

**Reflexión (asumo que de trabajado ahorita:**

Trabajar en un equipo grande fue desafiante, pero me dió una experiencia valiosa. Al rediseñar las interfaces del código, enfrentamos desafíos importantes debido a la mala comunicación y la falta de tiempo. La coordinación es difícil porque cada miembro del equipo tenía su propia interpretación de la solución. Tuvimos que superar malentendidos y aclarar nuestras ideas, lo que hizo que todo el proceso tardara más de lo esperado. Los problemas encontrados incluyen diferencias de implementación y problemas de integración debido a la falta de estandarización en nuestras prácticas de codificación. A través de estos desafíos, aprendí la importancia de una comunicación clara y un enfoque unificado en todos los proyectos grupales, incluso para los cambios de código más pequeños. Aunque este desafío fue difícil, mejoró nuestra colaboración y resiliencia en el entorno de desarrollo de software.