# Sistem de Management al unei Biblioteci

Proiect realizat pentru

**Universitatea Transilvania din Brașov**

**Facultatea de Matematică și Informatică**


**Autor:**

**Vicaș Antonio-Samir**

**Anul de studiu:** II

**Specializarea:** Informatică aplicată

**Grupa:** 10LF333

**Email:** antoniox2004.av@gmail.com


**Data:** Decembrie 2024

**Rezumat**

Proiectul Library Management System este o aplicație dezvoltată în Java care permite gestionarea articolelor unei biblioteci, cum ar fi cărți, reviste și benzi desenate. Aplicația oferă funcționalități pentru adăugarea, editarea, listarea și ștergerea articolelor, utilizând o interfață simplă bazată pe linia de comandă (CLI). Datele sunt stocate și gestionate în fișiere JSON, asigurând persistența între sesiunile de utilizare.

Acest proiect ilustrează utilizarea conceptelor de programare orientată pe obiecte, precum moștenirea și polimorfismul, împreună cu manipularea datelor în format JSON. Proiectul este structurat modular, facilitând extinderea și întreținerea ulterioară.

Cuvinte-cheie: Java, JSON, Programare Orientată pe Obiecte, Sistem de Gestionare a Bibliotecilor, CLI.

# Cuprins

**1 Introducere** **3**
1.1 Scopul proiectului . . . . . 3

**2 Cerințe** **4**
2.1 Hardware . . . . . 4
2.2 Software . . . . . 4
2.3 Dependete . . . . . 4

**3 Sistem de operare** **4**
3.1 Environment Configuration . . . . . 4

**4 Design** **5**
4.1 Interfaces . . . . . 5
4.2 Modules . . . . . 5
4.3 Main Application . . . . . 5

**5 Implementare. Descrierea detaliată a claselor** **7**
5.1 Clasa Item . . . . . 7
5.2 Clasa Book . . . . . 7
5.3 Clasa Magazine . . . . . 7
5.4 Clasa Comic . . . . . 8
5.5 Clasa LibraryManager . . . . . 8
5.6 Clasa LibraryApp . . . . . 9

**6 Testare** **9**
6.1 Testare funcționalități principale . . . . . 9

**7 Argumentarea Laboratoarelor** **10**
7.1 Tipuri primitive de date . . . . . 10
7.2 Input, Output, While, If, For, Switch . . . . . 10
7.3 Clase și Moștenire . . . . . 10
7.4 Interfețe . . . . . 11
7.5 Aplicație . . . . . 11

7.6 Maven . . . . . 11
7.7 Concluzie . . . . . 11

**8 Concluzie** **12**

**9 Erori și Limitări** **12**

# Listă de figuri

1 Diagrama UML a proiectului . . . . . 6

# 1 Introducere

Proiectul Library Management System este o aplicație dezvoltată în Java care oferă funcționalități pentru gestionarea articolelor dintr-o bibliotecă. Aplicația permite utilizatorilor să creeze, să editeze, să listeze și să gestioneze articole precum cărți, reviste și benzi desenate. Sistemul utilizează fișiere JSON pentru stocarea și încărcarea datelor, asigurând o interfață simplă și eficientă în linia de comandă (CLI).

## 1.1 Scopul proiectului

Acest proiect este conceput pentru:

- A oferi o soluție de gestionare a datelor pentru o bibliotecă simplă.
- A demonstra utilizarea conceptelor de programare orientată pe obiecte, precum moștenirea, polimorfismul și utilizarea interfețelor.
- A explora stocarea și manipularea datelor în format JSON.

Funcționalități principale:

1. **Adăugare de articole:** Utilizatorii pot adăuga cărți, reviste sau benzi desenate cu detalii specifice.
2. **Editarea articolelor:** Permite modificarea detaliilor articolelor existente.
3. **Listarea articolelor:** Afișează toate articolele din bibliotecă în format text.
4. **Salvarea și încărcarea datelor:** Suport pentru stocarea datelor în fișiere JSON și încărcarea lor ulterioară.
5. **Gestionarea fișierelor:** Posibilitatea de a crea și șterge fișiere JSON.

Beneficii:

- **Simplicitate:** Aplicația oferă o interfață intuitivă, bazată pe linia de comandă.
- **Extensibilitate:** Structura modulară permite extinderea funcționalităților prin adăugarea de noi tipuri de articole.
- **Persistență a datelor:** Stocarea articolelor în fișiere JSON asigură disponibilitatea datelor între sesiunile aplicației.

# 2 Cerințe

## 2.1 Hardware

- Procesor: Min. 2 GHz Dual-Core.
- Memorie RAM: Min. 4 GB.
- Spațiu de stocare: Min. x MB.

## 2.2 Software

- Java Development Kit (JDK): Versiunea 17 sau mai recentă
- IDE recomandat: IntelliJ IDEA / Eclipse / VS Code (cu plugin Java)
- Maven: Pentru gestionarea dependențelor și build-ul proiectului

## 2.3 Dependete

- Biblioteca JSON: org.json:json (versiunea 20240303)

# 3 Sistem de operare

Sistem de operare compatibil:

- Windows 10 sau mai recent
- macOS 11 sau mai recent
- Linux (distribuții recente, cu suport Java)

## 3.1 Environment Configuration

- Asigură-te că variabila de mediu `JAVA_HOME` este configurată corect.
- Adaugă directorul `bin` al JDK în `PATH`.

# 4 Design

## 4.1 Interfaces

- Storable: Interfață pentru obiectele care pot fi salvate și încărcate din fișiere JSON.

## 4.2 Modules

- Item: Clasă abstractă pentru articolele din bibliotecă.
- Book: Clasă pentru cărți.
- Magazine: Clasă pentru reviste.
- Comic: Clasă pentru benzi desenate.
- LibraryManager: Gestioneaza operatiile asupra bibliotecii.

## 4.3 Main Application

- LibraryApp: Clasa principală a aplicației.

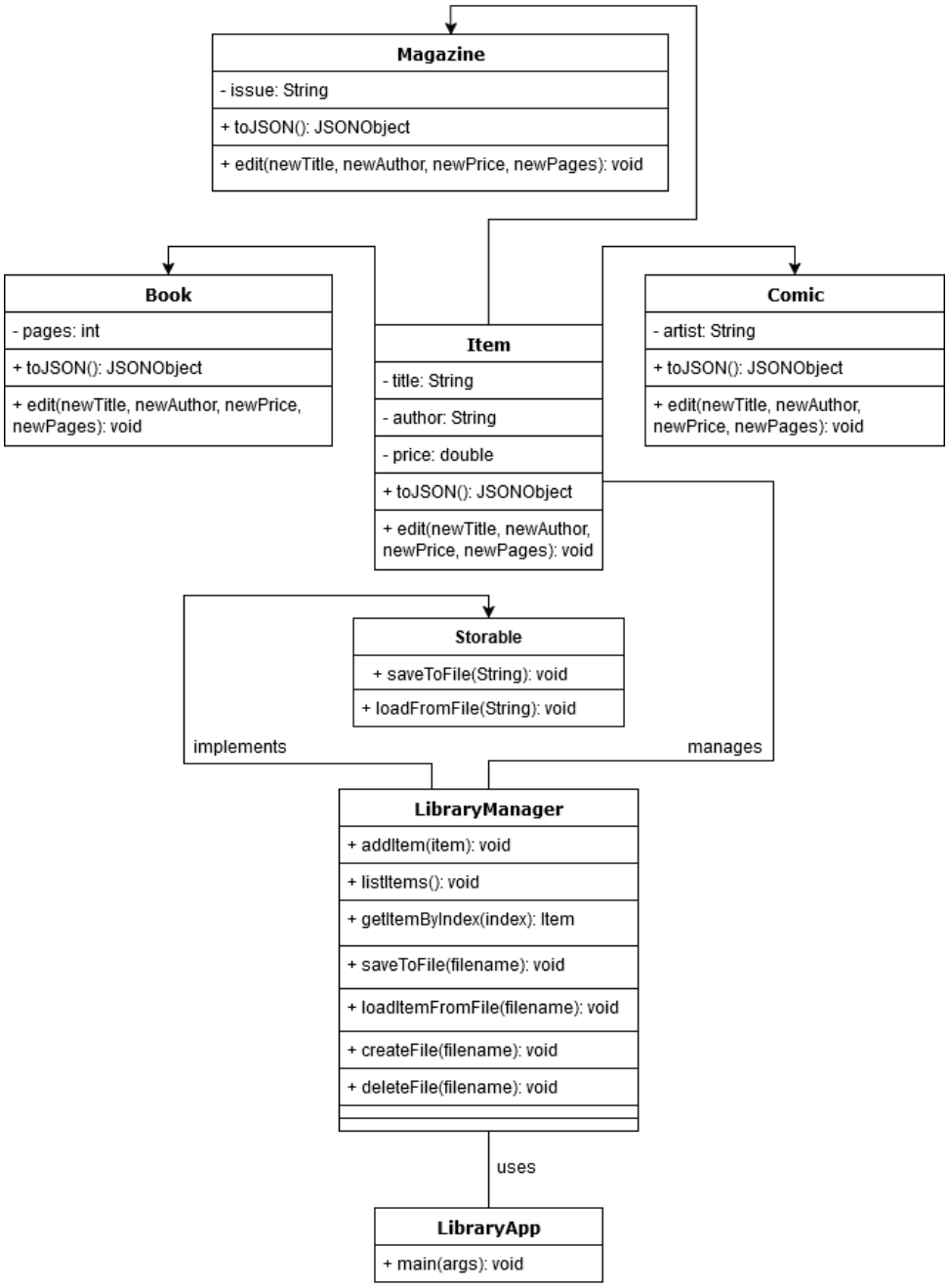


Figura 1: Diagrama UML a proiectului

# 5 Implementare. Descrierea detaliată a claselor

## 5.1 Clasa Item

- Este o clasă abstractă care definește structura de bază pentru articolele din bibliotecă.
- Atribute:
  - `title` (String): Titlul articolului.
  - `author` (String): Autorul articolului.
  - `price` (double): Prețul articolului.
- Metode:
  - `Item(String title, String author, double price)`: Constructor pentru inițializarea atributelor de bază.
  - `abstract JSONObject toJSON()`: Metodă abstractă pentru convertirea obiectului într-un format JSON.
  - `void edit(String newTitle, String newAuthor, double newPrice)`: Permite modificarea detaliilor comune.
  - `String toString()`: Returnează o reprezentare textuală a articolului.

## 5.2 Clasa Book

- Extinde clasa `Item`.
- Atribute suplimentare:
  - `pages` (int): Numărul de pagini al cărții.
- Metode:
  - `Book(String title, String author, double price, int pages)`: Constructor.
  - `JSONObject toJSON()`: Convertește obiectul într-un format JSON.
  - `void edit(String newTitle, String newAuthor, double newPrice, int newPages)`: Permite modificarea detaliilor cărții.

## 5.3 Clasa Magazine

- Extinde clasa `Item`.
- Atribute suplimentare:
  - `issue` (String): Ediția revistei.
- Metode:
  - `Magazine(String title, String author, double price, String issue)`: Constructor.

- `JSONObject toJSON()`: Convertește obiectul într-un format JSON.
- `void edit(String newTitle, String newAuthor, double newPrice, String newIssue)`: Permite modificarea detaliilor revistei.

## 5.4 Clasa Comic

- Extinde clasa `Item`.
- Atribute suplimentare:
  - `artist` (String): Artistul care a realizat banda desenată.
- Metode:
  - `Comic(String title, String author, double price, String artist)`: Constructor.
  - `JSONObject toJSON()`: Convertește obiectul într-un format JSON.
  - `void edit(String newTitle, String newAuthor, double newPrice, String newArtist)`: Permite modificarea detaliilor benzii desenate.

## 5.5 Clasa LibraryManager

- Implementarea interfeței `Storable`.
- Responsabilă cu gestionarea articolelor din bibliotecă și interacțiunea cu fișierele JSON.
- Atribute:
  - `items` (List¡Item¿): Lista articolelor gestionate.
- Metode:
  - `void addItem(Item item)`: Adaugă un articol în bibliotecă.
  - `void listItems()`: Listează toate articolele.
  - `Item getItemByIndex(int index)`: Returnează un articol pe baza indexului.
  - `void saveToFile(String filename)`: Salvează articolele în format JSON.
  - `void loadFromFile(String filename)`: Încarcă articolele dintr-un fișier JSON.
  - `void createFile(String filename)`: Creează un fișier JSON gol.
  - `void deleteFile(String filename)`: Șterge un fișier JSON.

### 5.6 Clasa LibraryApp

- Este clasa principală a aplicației, care implementează interfața în linia de comandă (CLI).
- Metodă principală:
  - `main(String[] args)`: Oferă utilizatorului funcționalități pentru gestionarea bibliotecii printr-un meniu.
- Funcționalități:
  - Adăugare articole: Permite utilizatorului să adauge cărți, reviste și benzi desenate.
  - Editare articole: Permite editarea atributelor unui articol selectat.
  - Salvare și încărcare: Oferă opțiuni pentru salvarea și încărcarea datelor în/din fișiere JSON.
  - Gestionare fișiere: Creează și șterge fișiere JSON.
  - Ieșire: Închide aplicația.

## 6 Testare

### 6.1 Testare funcționalități principale

- **Adăugare articole:**
  - Test: Adăugarea unei cărți cu titlu, autor, preț și pagini valide.
  - Rezultat: Cartea a fost adăugată cu succes în bibliotecă.
- **Editare articole:**
  - Test: Modificarea titlului și prețului unei reviste existente.
  - Rezultat: Revista a fost actualizată corect.
- **Salvare și încărcare:**
  - Test: Salvarea datelor într-un fișier JSON și reîncărcarea acestora.
  - Rezultat: Datele au fost salvate și încărcate corect.
- **Ștergere fișiere:**
  - Test: Ștergerea unui fișier JSON existent.
  - Rezultat: Fișierul a fost șters cu succes.

# 7 Argumentarea Laboratoarelor

Proiectul *Sistem de Management al unei Biblioteci* a fost realizat utilizând conceptele și tehnologiile studiate pe parcursul laboratoarelor. Fiecare laborator a contribuit la înțelegerea și implementarea unor funcționalități esențiale.

## 7.1 Tipuri primitive de date

Acest laborator a fost fundamental pentru înțelegerea variabilelor de bază utilizate în proiect.

- Tipuri de date precum `String`, `double`, `int` au fost utilizate pentru a stoca atributele fiecărui articol (titlu, autor, preț, etc.).
- Cunoașterea tipurilor primitive a fost esențială pentru a implementa metodele de calcul și procesare a datelor.

## 7.2 Input, Output, While, If, For, Switch

Acest laborator a fost esențial pentru implementarea interacțiunii utilizatorului cu aplicația:

- **Input și Output**:
  - Funcționalitățile de introducere și afișare a datelor au fost implementate folosind `Scanner` pentru input și metode precum `System.out.print()` pentru output.
- **While și Switch**:
  - Meniul principal al aplicației este implementat folosind un iclu `while` combinat cu o structură `switch` pentru gestionarea opțiunilor utilizatorului.
- **If și For**:
  - Condițiile și buclele `for` au fost utilizate pentru listarea articolelor și validarea datelor introduse de utilizator.

## 7.3 Clase și Moștenire

Laboratorul despre clase și moștenire a stat la baza organizării proiectului.

- Clasa abstractă `Item` definește atributele și metodele comune articolelor.
- Clasele `Book`, `Magazine` și `Comic` extind clasa `Item`, adăugând atribute și funcționalități specifice.
- Moștenirea a permis reducerea redundanței și reutilizarea logicii comune între clase.

## 7.4 Interfețe

Interfața `Storable` a fost implementată pentru a standardiza funcționalitățile de salvare și încărcare a datelor.

- Metodele abstracte `saveToFile` și `loadFromFile` au fost implementate în clasa `LibraryManager`.
- Utilizarea interfețelor a permis o organizare clară și extensibilitate pentru viitoare funcționalități de persistență.

## 7.5 Aplicație

Laboratorul despre aplicații a fost aplicat în implementarea clasei principale `LibraryApp`.

- Aplicația interacționează cu utilizatorul printr-un meniu în linia de comandă.
- Toate funcționalitățile (adăugare, editare, salvare, încărcare) sunt accesibile prin această interfață.

## 7.6 Maven

Laboratorul despre Maven a facilitat gestionarea dependențelor și structurarea proiectului.

- Biblioteca JSON (`org.json:json`) a fost integrată prin Maven.
- Structura proiectului (`src/main/java`, `src/test/java`) a fost generată automat.
- Utilizarea `pom.xml` a permis construirea și gestionarea proiectului într-un mod simplificat.

## 7.7 Concluzie

Fiecare laborator a contribuit la dezvoltarea abilităților necesare pentru finalizarea acestui proiect. Îmbinarea conceptelor fundamentale cu utilizarea unor unelte moderne precum Maven a permis realizarea unui sistem bine structurat, modular și scalabil.

# 8 Concluzie

Proiectul oferă o soluție simplă pentru gestionarea articolelor de bibliotecă. În viitor, se pot adăuga:

- Interfață grafică.
- Funcționalitati de căutare avansată.
- Integrare cu o bază de date.

# 9 Erori și Limitări

- **Erori posibile:**
  - Datele incorecte introduse de utilizator pot duce la comportamente neașteptate.
  - Ștergerea fișierelor fără verificare suplimentară poate cauza pierderea datelor.
- **Limitări:**
  - Aplicația nu suportă căutarea avansată sau filtrarea articolelor.
  - Sistemul de persistență este limitat la fișiere JSON; nu există integrare cu baze de date.
  - Nu există suport pentru gestionarea utilizatorilor sau a drepturilor de acces.