System Requirements Specification Index

For

Library Management System

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

TABLE OF CONTENTS

Library Management System
System Requirements Specification

1 Project Abstract

The City Public Library System requires a streamlined library management application to digitize their operations. The system will track books, library members, and handle borrowing operations. It will enable librarians to efficiently manage the library's collection by categorizing books, processing member registrations, and handling book checkouts and returns. This system provides an organized way for library staff to manage resources and serve community members effectively.

2 Business Requirements:

Screen Name	Console input screen
Problem Statement	1. System needs to store and manage different types of library data (books and members) 2. System must support operations such as book checkout, return, and member registration 3. Console should implement object oriented concepts like inheritance and method overriding to achieve desired outcome

3 Constraints

3.1 Class Requirements

1. `Book` Class:
    - Attributes: book_id, title, author, genre, publication_year, is_available
    - Methods: display_info(), checkout(), return_to_library()

- Example: `Book("B001", "To Kill a Mockingbird", "Harper Lee", "Fiction", 1960, True)`

2. `Member` Class:
   - Attributes: member_id, name, email, books_borrowed
   - Methods: display_info(), borrow_book(), return_book()
   - Example: `Member("M001", "John Smith", "john@example.com", [])`
3. `FictionBook` Class (inherits from `Book`):
   - Additional attributes: fiction_type
   - Override methods: display_info()
   - Example: `FictionBook("B001", "To Kill a Mockingbird", "Harper Lee", "Fiction", 1960, "Novel")`
4. ‘NonFictionBook` Class (inherits from `Book`):
   - Additional attributes: subject
   - Override methods: display_info()
   - Example: `NonFictionBook("B002", "A Brief History of Time", "Stephen Hawking", "Non-Fiction", 1988, "Physics")`
5. `Library` Class:
   - Attributes: name, address, books, members
   - Methods: add_book(), add_member(), checkout_book(), return_book(), get_available_books(), search_book_by_title(), search_book_by_author()
   - Static methods: get_book_count(), get_member_count()
   - Example: `Library("City Public Library", "123 Main St, Anytown")`

3.2 Operation Constraints

1. Book Checkout:
   - Member must exist in the system
   - Book must be available
   - Member cannot exceed maximum borrowing limit (3 books)
   - Must update book availability status

2. Book Return:
   - Book and member must exist in the system
   - Must update book availability status

- Member must have borrowed the book

3. Member Registration:
   - Member ID must be unique
   - Email must be valid format (must contain @ and a domain)

4. Book Addition:
   - Book ID must be unique
   - Publication year must be a valid year (not in the future)
   - Book must be assigned to the correct subclass (Fiction/NonFiction)

5. Exception Handling:
   - Must handle BookNotFoundException
   - Must handle MemberNotFoundException
   - Must handle BookNotAvailableException
   - Must handle MaxBooksExceededException
   - Must handle InvalidInputException

6. Object-Oriented Requirements:
   - Must use proper encapsulation (private attributes with getters/setters)
   - Must implement inheritance for book types
   - Must use polymorphism with method overriding
   - Must implement static methods and class variables

3.3 Output Constraints

1. Display Format:
   - Book information: display ID, title, author, genre, publication year, availability status
   - Member information: display ID, name, email, books borrowed
   - Each item must be displayed on a new line with proper formatting

2. Output Format:
   - Must show in this order:
     - Show "== LIBRARY MANAGEMENT SYSTEM =="
     - Show "Library Name: {name}"
     - Show "Address: {address}"

- Show "Total Books: {count}"
- Show "Total Members: {count}"
- Show "Current Book Collection:"
- Show books with format: "{id} | {title} by {author} | {genre} | {year} | Available: {status}"
- Show "Search Results:" when displaying search results

4. TEMPLATE CODE STRUCTURE:

1. Book Classes:

- `Book` (base class)
- `FictionBook` (derived class)
- `NonFictionBook` (derived class)

2. Member Class:

- `Member`

3. Library Class:

- `Library`

4. Exception Functions:

- `BookNotFoundException`
- `MemberNotFoundException`
- `BookNotAvailableException`
- `MaxBooksExceededException`
- `InvalidInputException`

5. Program Control:

- `main()` - main program function

5. EXECUTION STEPS TO FOLLOW:

1. Run the program
2. View the main menu
3. Select operations:
   - Option 1: Add New Book
   - Option 2: Add New Member

- Option 3: Checkout Book
- Option 4: Return Book
- Option 5: Display All Books
- Option 6: Display All Members
- Option 7: Search for Books
- Option 0: Exit

4. Perform operations on the library system
5. View results after each operation
6. Exit program when finished