**Library Management System**

**-**

**System Documentation**

**Presented By: Ellen Dalton**

**Date: February 28, 2024**

## Table of Contents

1. User Documentation…………………………………………………………… 3
   1. About
   2. Classes and Functionality
   3. UML Diagrams
   4. How to Start/Access the Application
2. Development Documentation………………………………………………….. 6
   1. Javadocs
   2. Source Code Structure
   3. Build Process
   4. Compiler Time Dependencies
   5. Development Standards
   6. Setting up a Database for Development
   7. Getting the Source Code from the Repository
3. Deployment Documentation……...………………………………………….... 8
   1. Installation Manuel

## 

## 1.0 User Documentation

## 1.1 About

The Library Management System is a Java-based application that serves as a tool for managing library resources. Its core functionalities encompass the creation, modification, and deletion of books, authors, and patrons. The system ensures accurate tracking of borrowed books by specific patrons, maintains real-time counts of available books, and provides comprehensive information about books authored by specific authors. Additionally, librarians benefit from a user-friendly search feature, enabling them to locate books effortlessly by ISBN, title, or author name.

## 1.2 Classes and Functionality

To facilitate the management of books, authors, and patrons, the application is structured around six classes and one interface:

1. The Library Class: Manages the central repository of library resources. It includes lists for books, authors, and patrons and enables operations such as adding and deleting entries. It allows users to search for books by entering the title, author, or ISBN, and by accessing this class, librarians can lend books to patrons or return them to the library. This automatically updates the patron’s borrowed book list as well as the number of copies of a book that is available at any given time. The library class is an overarching class that utilizes methods in other classes in order to update the system as a whole.

2. The Book Class: Represents individual books within the library. This class stores essential details such as title, ISBN, author, publisher, and available copies. It implements functionality for accessing and updating book information and for managing borrowings and returns.

3. The Author Class: Handles information related to authors, including their names, their dates of birth, and associated books. This class allows for automatically updating the list of authored books when new book entries are created.

4. The Patron Class: Manages patron information, including names, addresses, and borrowed book lists. Allows for efficient tracking of borrowed books and facilitates updates when patrons borrow or return items.

5. The Address Class: Stores address information, including street, city, province, and postal code, facilitating the creation of patron records with accurate location details.

6. The Demo Class: A testing class designed to verify the functionality of the library system. Used to create, modify, and delete test data within the application.

\*\*The Borrowable Interface:\*\* Implemented by the Book class to ensure consistent behavior in updating available copies when books are borrowed or returned.

By leveraging these classes and interface, the Library Management System offers a comprehensive solution for libraries to effectively organize, track, and manage their collections while enhancing the overall user experience.

## 1.3 UML Diagrams

A screenshot of a computer

Description automatically generated

## 1.4 How to Start/Access the Application

* Ensure you have Visual Studio Code installed on your computer.
* Ensure you have the Java Development Kit (JDK) installed (version 17.0.9 or later).
* Ensure you have the Java Extension Pack installed in your VS Code.
* Once all required downloads are confirmed, download the source code from GitHub (this can be found at <https://github.com/ejd500/Term3-Sprint1-Java.git>) and open the folder using your VS Code application.
* From the navigation pane found on the left side of VS Code, click on the Demo class.
* From here, you will be able to run the application by using the built-in “Run” feature of VS Code. The “Run Java” button is located at the top right corner of VS Code, represented by a triangle pointing to the right. Click here to run the code.

## 2.0 Development Documentation

### 2.1 Javadocs

* Javadoc comments are provided for all classes/methods/interface to document their functionality and usage.
* To compile the Javadoc documentation in a HTML format, you can use the VS Code integrated terminal.
* Open the terminal within VS Code by selecting “Terminal” > “New Terminal” from the menu.
* In the terminal, navigate to the directory containing the downloaded Java files and use the following Javadoc command: javadoc -d docs \*.java
* This will generate documentation for all Java files in the current directory, and its subdirectories, and will output the HTML files to a directory named “docs”.
* Ensure you have the Live Server extension installed in your VS Code. This will enable you to run the Javadoc documentation from a browser.
* Right click on the help-doc.html file and select “Open with Live Server”. This should open the Javadoc Help menu in a browser environment.
* From here, you can navigate to all Javadoc documentation within the browser environment (the help menu is listed as a starting point so that it can help guide you through the usage of the Javadoc documentation).

## 2.2 Source Code Structure

All Java source files are organized in a single directory. Each class is defined in its own .java file, as is the Borrowable interface.

**2.3 Build Process**

Compile the project using a Java compiler (e.g., javac \*.java).

**2.4 Compiler Time Dependencies**

There are no external dependencies required at compile time.

**2.5 Development Standards**

Followed standard Java coding conventions and best practices for naming, formatting, and structuring code.

**2.6 Setting up a Database for Development**

This application does not utilize a database for development.

**2.7 Getting the Source Code from the Repository**

Visit <https://github.com/ejd500/Term3-Sprint1-Java.git> to download or clone source code from the repository.

## 3.0 Deployment Documentation

**3.1 Installation Manual**

* Ensure a Java Runtime Environment (JRE) is installed on the target system.
* Copy the compiled .class files to the deployment directory.
* Run the Demo class to start the application.
* See section 1.4 of this document for further instruction.