**1. Introduction**

This documentation provides a comprehensive guide to the development of the Library Management System, including the source code structure, build process, compiler dependencies, development standards, theoretical database design, and instructions for accessing the source code from GitHub.

**2. Javadocs**

Each class and method in the Library Management System has been documented with Javadoc comments. These comments provide descriptions of the purpose and functionality of each class, method, and attribute.

**Example:**

java

Copy code

/\*\*

\* Represents a library item.

\*/

abstract class LibraryItem {

/\*\*

\* The unique identifier of the item.

\*/

private String id;

/\*\*

\* The title of the item.

\*/

private String title;

/\*\*

\* The author of the item.

\*/

private Author author;

// Additional attributes and methods...

}

**3. Source Code Directory Structure**

LibraryManagementSystem/

│

├── src/

│ ├── LibraryManagementSystem.java

│ ├── LibraryItem.java

│ ├── Book.java

│ ├── Periodical.java

│ ├── Author.java

│ ├── Patron.java

│ ├── Student.java

│ ├── Employee.java

│ ├── LibraryItemBorrowing.java

│ ├── LibraryItemReturning.java

│ └── Main.java

│

├── docs/

│ └── Javadocs (Generated Javadocs files)

│

└── README.md

**4. Build Process**

1. **Compile the Project:**
   * Open a terminal or command prompt.
   * Navigate to the src directory.
   * Run the following command to compile the Java files:

javac \*.java

1. **Run the Project:**
   * Run the Main class using the following command:

java Main

**5. Compiler Time Dependencies**

Ensure that you have JDK installed on your system. The project does not have any additional dependencies beyond the standard Java libraries.

**6. Development Standards**

* **Code Conventions:** Follow standard Java coding conventions.
* **Naming Conventions:** Use camelCase for variables and methods, Pascal Case for class names.
* **Comments:** Use Javadoc comments for classes and methods. Use inline comments for complex logic.
* **Testing:** Ensure each method is tested for various input scenarios.
* **Version Control:** Use Git for version control.

**7. Theoretical Database Design**

While the current implementation does not include a database, here is a theoretical design:

**Tables and Relationships**

* **LibraryItem Table:**
  + id: Primary Key
  + title
  + authorId: Foreign Key referencing Author Table
  + ISBN
  + publisher
  + numberOfCopies
  + type: Indicates whether the item is a Book or Periodical
  + format (for Book)
  + periodicalType (for Periodical)
* **Author Table:**
  + id: Primary Key
  + name
  + dateOfBirth
* **Patron Table:**
  + id: Primary Key
  + name
  + address
  + phoneNumber
  + type: Indicates whether the patron is a Student or Employee
* **Borrowing Table:**
  + id: Primary Key
  + patronId: Foreign Key referencing Patron Table
  + itemId: Foreign Key referencing LibraryItem Table
  + borrowDate
  + returnDate

**8. Accessing the Source Code from GitHub**

1. **Clone the Repository:**
   * Open a terminal or command prompt.
   * Run the following command to clone the repository:

sh

Copy code

git clone https://github.com/your-username/library-management-system.git

1. **Navigate to the Project Directory:**
   * Change directory to the cloned repository:

sh

Copy code

cd library-management-system

1. **Compile and Run the Project:**
   * Follow the build process steps mentioned above to compile and run the project.