HIGH LEVEL DESIGN

Library Management System Web Application

Version<1.0>

Contents

Overview:

Architecture:
The system will be a web-based application with a three-tier architecture: presentation, business, and data access layers.
Spring Boot will be used as the foundation for the application, providing the web server, dependency injection, and other features.
Hibernate will be used as the Object-Relational Mapping (ORM) framework for interacting with the MySQL database.

Data Structures:
The system will use a set of entities that will be used to model the various elements of the system, such as books, authors, users, and library staff.
The entities will be represented as Java classes, and Hibernate will be used to map them to database tables.
The system will use a set of repositories that will be used to interact with the database and perform CRUD operations on the library entities.

Algorithms:
The system will use several algorithms for searching and filtering books, authors, and other library-related information.
The system will implement the algorithm to calculate fine if a user returns a book late.
The system will implement the algorithm to check the availability of a book before lending it.

Interfaces:
The system will use a set of controllers that will handle HTTP requests and route them to the appropriate services.
The system will use a set of services that will handle the business logic of the application, such as adding, updating, and deleting books, authors, and other library-related information.
The system will use JSP for the front-end, which will provide the user interface for the application

1. Introduction

1.1. Why this High Level Design Document?

The purpose of this High Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

1.2. Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

1.3. Definitions

- Firewall - Functionality that can allow or block certain ports and addresses.
- Microsoft Defender - A firewall built into the Windows Operating System.
- MySql Server - A database management system.
- Java - A possible programming language to interface between Microsoft Defender and MySql.
- JDBC - A possible Java-based interface between Microsoft Defender and the Database.
- JSP - The language that will be used for displaying user history and administrative functionality.
- Tomcat - a freopen-source implementation of Java Servlet and JavaServer Pages technologies developed under the Jakarta project at the Apache Software Foundation.
- ER - Entity Relation Diagram
- This is a protocol that lets network administrators centrally manage and automate the assignment of IP Addresses on the corporate network.
- Gateway - Bridges the gap between the internet and a local network.

1.4. Overview

The HLD will:

- a present all of the design aspects and define them in detail
- describe the user interface being implemented
- describe the hardware and software interfaces
- describe the performance requirements
- include design features and the architecture of the project

list and describe the non-functional attributes like:

- security
- reliability
- maintainability
- portability
- reusability
- application compatibility
- resource utilization
- serviceability

2. General Description

2.1 Product Perspective

The Library Management System will be composed of several different components. Some of these components will be programmed, while others will be implementations of open-source programs. The language implemented will be dictated by its purpose. The administrative and user interfaces will be using JSP to display the pages, and SQL to retrieve, insert, delete, and update the database. Either Java or JDBC will be used to submit SQL commands for the automated part of the project such as updating the user history and DHCP data. This setup will allow for multiple users to login and interact with the program at the same time. It will also be set up using two user levels. First is the basic user, which can only view their current average and their history. This page is automatically displayed based on their IP address. The second type of user is the Administrator. They have the ability to change information in the database such as settings and user history. This user level can only be attained by logging into the system.

2.2 Tools Used

- Spring Boot is used to generate all of the diagrams used in analysis and design phases of the project.
- The project will have a relational database backend that is SQL based. The actual software used is MySql.
- Interfacing with the database to display information on the user's web browser will be done using JSP. It can connect to the database and parse it into viewable

HTML code.

- Tomcat compiled JSP pages into servlets to be displayed through Apache.
- Automated interfacing with the database behind the scenes will be either Java or JDBC.
- Eclipse is the development platform.

2.3 General Constraints

The Library Management System must be user friendly and as automated as possible. Administrators should not be required to do anything besides the initial setup, and users should not be required to know any of the workings. Without logging in, the user will only have the ability to view that IP's current average and history. After logging in, that user then has the ability to change settings and user historie

2.4 Assumptions

- It supports all the 'back-end' operations of a library – acquiring stock/information, cataloging stock, loaning stock/disseminating information and reporting on these functions to enable effective service management.
- It enables users to find out what items or information a library has, and then borrow/access as appropriate their required items/information.

2.5. Special Design aspects

System analysis and design deal with planning the development of information systems through understanding and specifying in detail what a system should do and how the components of the system should be implemented and work together. System analysis and design solve business problems through analyzing the requirements of information systems and designing such systems by applying analysis and design techniques.
System analysis and design is the most essential phase in the development of a system since the logical system design arrived at as a result of systems analysis which is in turn converted into physical system design.

3.Design Details

3.1 Main Design Features

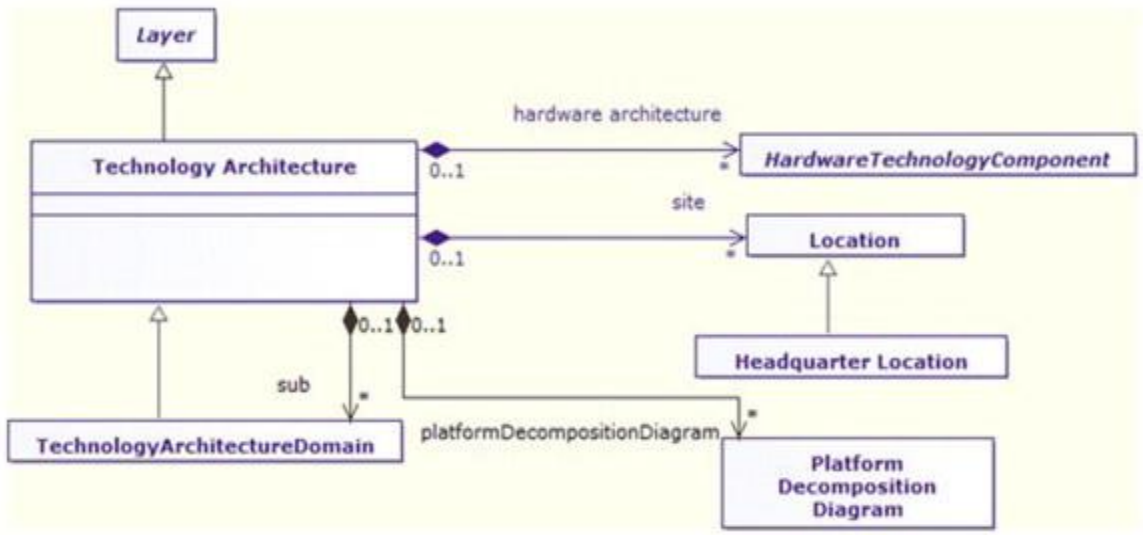
System analysis and design deal with planning the development of information systems through understanding and specifying in detail what a system should do and how the components of the system should be implemented and work together. System analysis and design solve business problems through analyzing the requirements of information systems and designing such systems by applying analysis and design techniques.
System analysis and design is the most essential phase in the development of a system since the logical system design arrived at as a result of systems analysis which is in turn converted into physical system design.

3.2. Application Architecture

The architecture of applications is usually broken into logical chunks called "tiers", where every tier is assigned a role. A“tier” can also be referred to as a “layer”. There are three layers involved in the application namely Presentation Layer, Business Layer and Data Layer. Each layer is explained in detail below.

3.3 Technology Architecture

Technology architecture deals with the deployment of application components on technology components. A standard set of predefined technology components is provided in order to represent servers, network, workstations, and so on .



The human side of architecture.

Historically, the discipline of Enterprise Architecture concentrated on shaping complex IT and technology architectures in alignment with business requirements. Today's practitioners expand their scope to address all structures that constitute an enterprise and make it work. Organizational reporting lines, information systems delivering data, processes driven by automated systems and human decisions—all these structures are just different aspects of the same system. The resulting architecture is the foundation of every single step the enterprise takes, from sending a receipt to a customer to a merger with another organization. Just as buildings are structures made for people to live in and look at, enterprises are structures made by people for people as a space for interactions and transactions. In consequence, people themselves cannot be seen as assets to be incorporated in an architectural description, only the enterprise's relationship to them. Any person in touch with an enterprise is both a user of its architecture and a contributor

3.3.1 Web Application Architecture

The web application market is an environment that is continuously evolving, incorporating new technologies and increasing safety standards.In this regard, the proper attention to the foundational design of a web app contributes to maintaining the required robustness, responsiveness, and security of this software. In a web app, this is what solid web app architecture is responsible for.The definition of web app architecture is broad and depends on the focus of building web applications.A web application architecture is a model of interaction between web application components. The kind of architecture for web application strictly depends on the way application logic will be allocated among client and server sides.Technically, it’s the skeleton of a web app, including its elements, databases, systems, servers, interfaces, and all the communication happening between them. In more abstract terms, it indicates the logic behind responses to client and server requests.

3.3.2. Presentation Layer

It is also known as Client layer. Top most layer of an application. This is the layer we see when we use a software.By using this layer, we can access the web pages. The main functionality of this layer is to communicate with the Application layer. This layer passes the information which is given by the user in terms of keyboard actions, mouse clicks to the Application Layer.

For example, the login page of Gmail where an end user could see text boxes and buttons to enter user id,password and to click on sign-in. In simple words, it is to view the application.

3.3.3. Data Access Layer

The Data-Access Layer (DAL) is a component of a software architecture that is responsible for managing the data storage and retrieval of an application. It sits between the business-logic layer and the data storage system and provides an abstraction layer that allows the business-logic layer to interact with the data storage system without being aware of its specific implementation.

The DAL is responsible for performing tasks such as:

- Connecting to the data storage system and managing the connection.
- Generating and executing SQL queries or other data access commands to retrieve and store data.
- Mapping the data from the data storage system to the application's data objects and vice versa.
- Handling errors and exceptions related to data access.
- Providing support for transactions and other data access features.
- The DAL is designed to be reusable and independent of the business logic and data storage implementation. This allows the application to be easily modified or extended without affecting the underlying data access code.

3.4 Standards

When designing a library management system using Java and Spring Boot, there are several standards and best practices that should be followed to ensure that the system is reliable, efficient, and maintainable. Some of these include:

- Adhering to the principles of object-oriented programming: This includes using encapsulation, inheritance, and polymorphism to create reusable and modular code.
- Following best practices for Spring Boot: Spring Boot is a popular framework for building Java-based applications, and there are several best practices that should be followed when using it. This includes using proper naming conventions, following the recommended project structure, and configuring the application's properties correctly.

- Following the MVC pattern: Spring Boot implements the MVC pattern for web applications. This pattern separates the concerns of the application into three layers: the model, the view, and the controller.
- Using appropriate design patterns: Design patterns are reusable solutions to common programming problems. Some of the patterns that can be used in a library management system include the Singleton pattern, the Factory pattern, and the Observer pattern.
- Additionally, it is also important to ensure that the system is secure and adheres to security standards. It should be tested and have proper validation and error handling.

By following these standards and best practices, a library management system designed using Java and Spring Boot can be more reliable, efficient, and maintainable, which can make it easier to support and improve over time.

3.5. Database Design

Database Design in Library Management System The above shows one possible database schema for the library management system. The users table stores the user details such as a unique user id, name, contact, and login information this page contains information of Library Management System database design. The database in turn is further described in detail giving all the fields used with the data types, constraints available, primary key and foreign key. Database design is used to manage large amounts of information.

3.6. Files

LM systems tend to be simpler and focus on efficiently storing data (files) in a library and managing versions or levels of the individual files. An example of a library management The system is CA-Panvalet. An example of a source code management system is SCLM. you can import a library ZIP file into Oracle Integration. The ZIP file must contain only the library JAR and metadata XML file.
In the left navigation pane, click Home > Integrations > Libraries.

1. Click Import in the banner.
2. In the Import Library dialog box, click Browse to select the ZIP file.
3. The Javascript Library page is displayed.
4. Make updates, as necessary, then click Save.

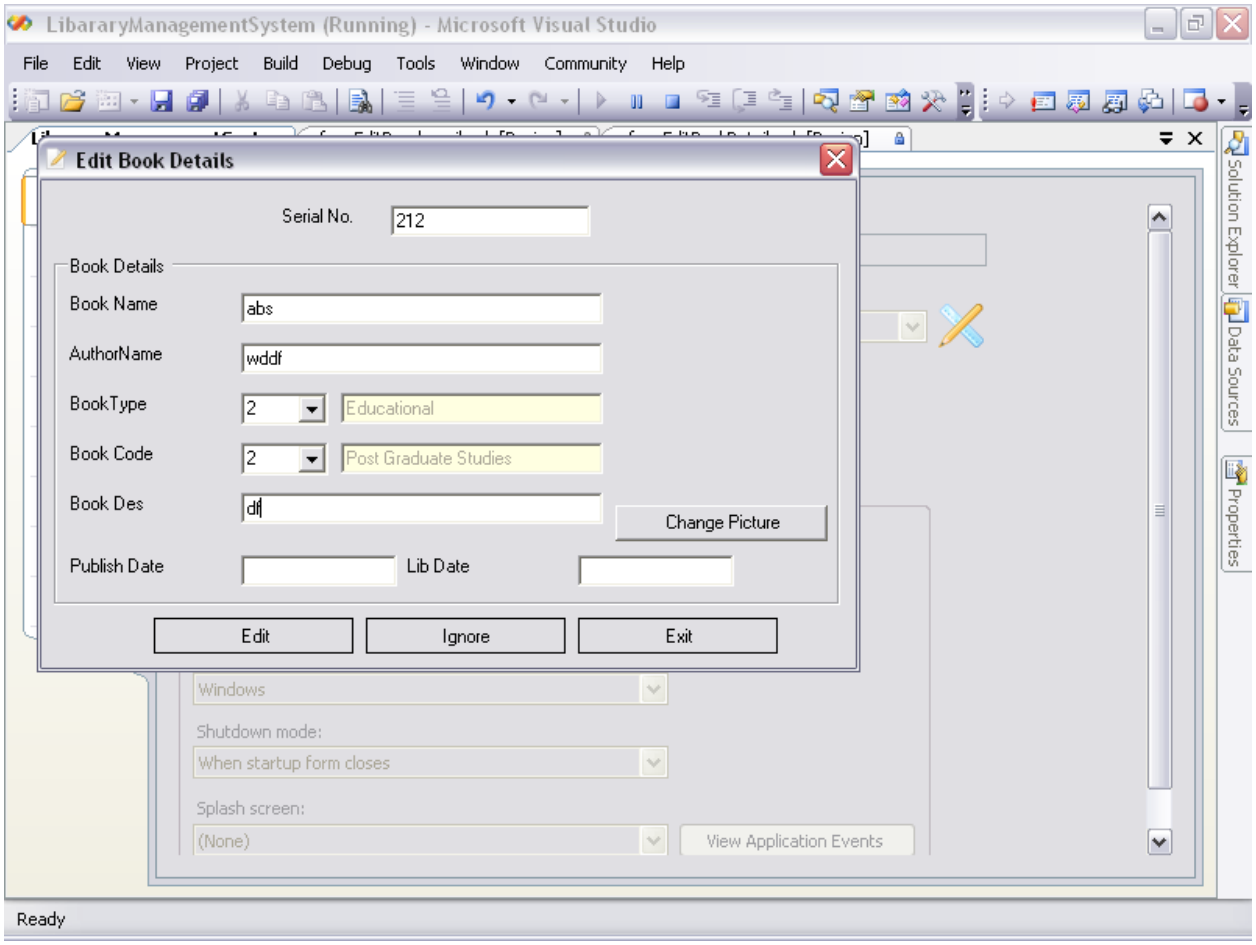
- Previous
- Next

3.7. User Interface

The user interface (UI) is the point at which human users interact with a computer, website or application. The goal of effective UI is to make the user's experience easy and intuitive, requiring minimum effort on the user's part to receive the maximum desired outcome. User Interface (UI) Design focuses on anticipating what users might need to do and ensuring that the interface has elements that are easy to access, understand, and use to facilitate those actions. UI brings together concepts from interaction design, visual design, and information architecture.
Choosing Interface Elements

Users have become familiar with interface elements acting in a certain way, so try to be consistent and predictable in your choices and their layout. Doing so will help with task completion, efficiency, and satisfaction.
Interface elements include but are not limited to:
Input Controls: buttons, text fields, checkboxes, radio buttons, dropdown lists, list boxes, toggles, date field
Navigational Components: breadcrumb, slider, search field, pagination, slider, tags, icons
Informational Components: tooltips, icons, progress bar, notifications, message boxes, modal windows

There are times when multiple element User Interface (UI) Design focuses on anticipating what users might need to do and ensure that the interface has elements that are easy to access, understand, and use to facilitate those actions. UI brings together concepts from interaction design, visual design, and information architecture.



3.8. Reports

A library management system is software that is used to automate the processes of a library, such as cataloging, circulation, and the acquisition of new materials. Reports in a library management system typically include information on the library's inventory, circulation statistics, and user information. Examples of reports that may be generated by a library management system include:

- Catalog reports, which provide information on the library's inventory of materials, such as title, author, and call number.
- Circulation reports, which provide information on the circulation of materials, such as the number of items checked out, overdue items, and the most popular items.
- Users reports, which provide information on the library's user, such as contact information, reading history, and fines.
- Acquisition reports, which provide information on the library's budget, purchase orders, and vendors.
- Statistical reports, which provide data on the usage of the library and its resources, such as number of visitors, number of checkouts, number of reference questions, etc

These reports can be used to make data-driven decisions to improve the operations of the library and better serve the needs of users.

3.9 Error Handling

Error handling is an important aspect of any library management system, as it ensures that the system can gracefully handle unexpected situations and provide meaningful feedback to users. Here are a few best practices for error handling in a library management system:

- Use exceptions: Exceptions are a built-in mechanism in Java for handling errors. They allow you to separate the error handling code from the main logic of the system, making the code more readable and maintainable.
- Log errors: Logging errors is an important part of error handling as it allows developers to understand what went wrong and fix the problem. Spring Boot has a built-in logging system, and you can also use third-party libraries such as Log4j or Logback.
- Provide meaningful error messages: Error messages should be clear and informative, so that users can understand what went wrong and take appropriate action.
- Consider security: Error handling should also consider security issues like input validation, authentication, and authorization.
- Additionally, it's important to test the system to ensure that it can handle different types of errors and provide appropriate feedback to users. This can include testing the system with invalid inputs, testing how the system behaves when it can't connect to a database or external service, and testing how the system behaves when it runs out of memory.

By implementing proper error handling, a library management system can ensure that it can continue to function even in the presence of errors, and provide meaningful feedback to users, which can improve the overall user experience.

3.10. Interfaces

In a library management system, there are typically several interfaces that are used to interact with the system. These interfaces can be divided into two main categories: user interfaces and application interfaces.

User interfaces: These interfaces provide a way for users and librarians to interact with the system. They can include graphical user interfaces (GUIs) that are accessed through a web browser or a desktop application, as well as command-line interfaces (CLIs) that can be used by librarians for more advanced tasks. User interfaces allow users to search for books, check out and return books, and view their account information. Librarians would use the user interface to manage the library's inventory, process transactions, and maintain user records.

Application interfaces: These interfaces provide a way for other applications to interact with the library management system. They can include APIs (Application Programming Interfaces) that allow other systems to access the library's catalog, circulation, and user management functions. These interfaces allow other systems to search the library's catalog, place holds on books, and check the availability of books. Additionally, many modern library management systems are cloud-based, which allows access to the system from anywhere, and also allows for easy data backup and recovery.

Database Interfaces: This interface allows the system to interact with the database where all the information of the library is stored. It allows the system to store, retrieve and update data as required by other components of the system.

Overall, the interfaces in a library management system are designed to provide easy access to the system's functions for both users and librarians, and to allow other systems to interact with the library management system and access its data

3.11. Performance

Performance in a library management system refers to how well the system is able to handle tasks and respond to user requests in a timely and efficient manner. Factors that can affect performance include:

- Hardware resources such as CPU, memory, and storage
- Software design and architecture
- Number of concurrent users and transactions
- Network infrastructure and connectivity
- Size and complexity of the data being managed

To improve performance, several techniques can be used such as:

- Caching: storing frequently accessed data in memory
- Load balancing: distributing workloads across multiple servers
- Monitoring and tuning: tracking system resource usage and adjusting settings to optimize performance

3.12. Security

Security in a library management system is important to protect sensitive information such as user's personal data and library's inventory data. Some common security measures include:

- Authentication: ensuring that only authorized users have access to the system
- Encryption: protecting data as it is transmitted and stored.
- Access control: limiting what actions users can perform and which data they can access.
- Auditing: tracking and logging system activity to detect and respond to security breaches.
- Regular software updates and vulnerability assessments.
- It is also important to have a disaster recovery plan in place in case of data loss or system failure.

3.13. Reliability

Reliability of a library management system refers to its ability to perform its intended functions consistently and without failure. A reliable system is one that is dependable, and can be trusted to work correctly under normal and expected conditions. It should also be able to handle and recover from unexpected situations or errors.

Some key factors that contribute to the reliability of a library management system include:

- robust error handling and recovery mechanisms

- regular backups and disaster recovery plans
- regular testing and quality assurance to identify and fix bugs and errors
- use of redundancy and failover mechanisms to ensure continuity of service in case of hardware or software failure
- regular maintenance and updates to keep the system running at optimal performance
- A reliable library management system can help ensure the availability of library resources and minimize disruptions to the library services. It can also increase user satisfaction and trust in the system.

3.14. Maintainability

Maintainability of a library management system refers to its ability to be easily maintained and updated over time. A maintainable system is one that is well-organized, well-documented, and has a clear and modular design. This makes it easier to troubleshoot and fix problems, add new features, and make changes to the system without disrupting the existing functionality.

Some key factors that contribute to the maintainability of a library management system include:

- a clear and consistent coding style
- well-documented code and clear comments
- use of industry standard best practices for software development
- use of modular design, with clear separation of concerns
- regular testing to catch and fix bugs early
- use of version control systems to keep track of changes to the codebase

A maintainable library management system can reduce the total cost of ownership, minimize downtime, and minimize the time and effort required to keep the system running smoothly over time.

3.15. Portability

A library management system's portability refers to its ability to run on different platforms or devices without the need for major changes or modifications. A portable library management system can be easily adapted to different operating systems, hardware configurations, and environments, and it can be used by different libraries or organizations. This can provide a cost-effective and flexible solution for managing library resources, as it eliminates the need to purchase and maintain multiple systems for different platforms.

3.16. Reusability

There are several components of a library management system that can be reused, including:

- Cataloging and metadata: This information can be reused to create a unified catalog of all library resources, which can be accessed by users through a single search interface.
- Circulation and tracking: The system's ability to track the book status of resources, including the due date and user information, can be reused to manage the circulation of resources and ensure that they are returned on time.
- Authentication and user management: The system's ability to authenticate users and manage their access to resources can be reused to control access to digital resources and provide personalized services to users.
- It's also important to consider that when a library management system is replaced by a new one, the data should be migrated, so it can be reused.

3.17. Application Compatibility

Library management systems are software applications that are designed to help manage the operations of a library. These systems can be compatible with a variety of platforms, including:

- Windows, Mac, and Linux operating systems
- Web browsers such as Chrome, Firefox, and Safari
- Mobile devices such as smartphones and tablets

3.18. Resource utilization

A library management system can utilize several resources, including:

- Physical resources such as books, journals, and other print materials.
- Digital resources such as e-books, online journals, and databases.
- Human resources such as librarians and library staff to manage and assist users.
- Technical resources such as computers and software for cataloging, searching, and tracking resources.

3.19. Major Classes

In a library management system, there are several major classes of objects that are used to represent and manage the different aspects of a library's operations. Some common classes include:

- Books: This class represents the physical books in the library's collection. It typically includes information such as the book's title, author, ISBN, publication date, and location in the library.
- Users: This class represents the library's users (i.e., users) and includes information such as their name, contact information, and borrowing history.
- Transactions: This class represents the borrowing and returning of books by users. It includes information such as the book, the user, the date borrowed and returned, and any fines incurred.
- Catalog: This class represents the catalog of books in the library and includes information such as the title, author, subject, and keywords.
- Administrator: An administrator can view or modify a user’s traffic history An administrator can modify system settings and user.

These classes can be further divided into subclasses, depending on the specific requirements of the library management system. It's also important to note that the classes in library management systems are closely interrelated, and the relationships between them are often modeled using object-oriented programming techniques.