LibVault

Database Requirements

Version 1

3/2/2025

1. Introduction

Project Overview: LibVault is a relational database system designed to execute library operations efficiently and in a user-friendly manner. It allows for the organization of loanable items, enforces borrowing policies, tracks user memberships, and generates reports in a manner that is useful for both staff and the clientele.

Scope: LibVault's Library Management System will cover library operations such as Book & Digital Media Management, Membership Management, Borrowing & Returns, and Reservations & Notifications. It will automate tracking of loans, returns, and overdue items, while also generating reports to support data-driven decision-making. The program will include the creation of an ER model, a relational schema, and an SQL-based database with realistic records. However, the project does not include the development of a front-end user interface or integration with external library systems at this stage.

Glossary:

- **Database Management System (DBMS):** Software used to create and manage databases, ensuring structured data storage and retrieval.
- **ER Model (Entity-Relationship Model):** A diagram representation of entities and their relationships in a database.
- **SQL (Structured Query Language):** A programming language used for managing and querying relational databases.
- **Primary Key:** A unique identifier for a record in a database table.
- **ISBN (International Standard Book Number):** A unique identifier assigned to books and other publications.
- **Role-Based Access Control (RBAC):** A security model that restricts access based on user roles.
- **Overdue Tracking:** The process of monitoring items that have not been returned by their due date.
- **Reservations:** A system that allows patrons to place holds on unavailable items.
- **SSL/TLS (Secure Sockets Layer/Transport Layer Security):** Cryptographic protocols that provide secure communication over a network.
- **SSH (Secure Shell):** A network protocol that allows secure access to remote computers.

2. Stakeholders

- **Library Clients:** End-users who will interact with the system to search for books, manage reservations, and check their borrowing history.
- **Librarians:** The users responsible for managing book inventory, processing checkouts and returns, and enforcing borrowing policies.
- **Library Administrators:** Those that oversee system-wide operations, manage user roles, configure policies, and generate reports.
- **Developers:** Technical users that are responsible for updating and maintaining the database system according to stakeholder needs.
- **Contributors:** users that can upload their own content to the library database for use by others.
- **Security Assessors:** users that check the database for vulnerabilities and possible exploits and then report them to developers and suggest ways to fix them.

3. **Requirements**

3.1 Functional Requirements

The database must support:

- **User Management:** Register members, manage profiles, enforce borrowing limits.
- **Item Cataloging:** Store and organize books, magazines, and digital resources.
- **Loan Transactions:** Handle book checkouts, returns, renewals, and overdue tracking.
- **Reservations & Notifications:** Allow patrons to reserve books and receive reminders.
- **Reporting & Analytics:** Generate reports on book popularity, overdue loans, and user activity.

3.2 Data Entities

Media:

- Attributes:
  - Title - String
  - Author - String
  - ISBN - String
  - Publication Year - Number
  - Genre - String
  - Availability Status - String
- Constraints:
  - ISBN Must be Unique - Primary Key
  - Publication Year Must be a Valid Year
  - Genre Must be from a Select Set Genres
    - Fiction, Non-fiction, Drama, etc…
  - Availability Status Must be from Predefined Statuses

        - Available, On Hold, Reserved

Clients:

- Attributes:
    - Member ID - Number
    - Name - String
    - Phone Number - String
    - Membership Type - String
    - Account Status - String
- Constraints:
    - Member ID Must be Unique - Primary Key
    - Phone Number Must be a Valid Phone Number
    - Membership Type Must be from a Predefined Set of Memberships
        - Active, Suspended
    - Availability Status Must be from Predefined Statuses
        - Regular, Student, Senior Citizen

Reports:

- Attributes:
    - Report ID - Number
    - Overdue Fee - Number
    - ISBN of Media Checked Out - String
    - Name of Media Checked Out - String
    - Membership Type - String
    - Account Status - String
- Constraints:
    - Report ID Must be Unique - Primary Key
    - ISBN and Name Must Exist in Media
        - Max of 10 Checked out Media
    - Membership Type Must be from a Predefined Set of Memberships
        - Active, Suspended
    - Availability Status Must be from Predefined Statuses
        - Regular, Student, Senior Citizen

Reservations:

- Attributes:
    - Reservation ID - Number
    - ISBN of Media Checked Out - String
    - Name of Media Checked Out - String
    - Member ID - Number
    - Reservation Date - String
    - Expiration Date - String
    - Media Availability Status - String
- Constraints:
    - Reservation ID Must be Unique - Primary Key
    - ISBN and Name Must Exist in Media
        - Max of 3 Reserved Media

        - Member ID Must exist in Client
        - Reservation Date and Expiration Date Must be on the Calendar
            - The Expiration Date will be 2 weeks after the book becomes available

3.3 Optional: Non-Functional Requirements

- **Performance:** Queries should execute within **2 seconds** for 95% of operations.
- **Security:** Member data must be **encrypted** and accessed via **role-based permissions**.
- **Scalability:** System should handle **1,000+ books and 100+ members** without degradation.

4. Hardware and Software Requirements

4.1 Hardware Requirements

1. Local Deployment (MySQL on Laptop/Personal Machine)

- **Processor:** Intel Core i5 or AMD Ryzen 5 (or higher)
- **Memory (RAM):** Minimum **8GB RAM** (16GB recommended for better performance)
- **Storage: At least 50GB SSD** (ensuring smooth database operations and backups)
- **Network:** Stable **internet connection** for software updates and remote access (if needed)
- **Graphics:** Not necessary (headless operation is fine)

2. EECS Server Deployment (MariaDB on EECS Servers)

- The server infrastructure is managed by the university, so no specific hardware details are needed.
- Ensure a stable SSH connection for remote access.
- Sufficient processing power and memory should be allocated based on database load.

3. Cloud Deployment (AWS RDS, Google Cloud SQL, or Azure Database for MySQL)

- **Instance Type:** Minimum **t3.medium** (2 vCPUs, 4GB RAM) for small-scale operations
- **Storage: 100GB SSD** (adjustable based on data growth)
- **Network Requirements:**
    - **Firewall configurations** to restrict unauthorized access
    - **SSL encryption** for secure connections

4.2 Software Requirements

1. Database Management System (DBMS)

- **MariaDB 10.x** (if hosted on EECS servers)
- **MySQL 8.x** (if using a local or cloud-based instance)
- **PostgreSQL 13.x** (optional alternative for relational database handling)

2. Database Client Tools

- **MySQL Workbench** (GUI for database visualization and management)
- **DBeaver** (cross-platform SQL management tool)
- **phpMyAdmin** (optional web-based interface for MySQL)
- Command Line Interface (CLI): mysql command-line tool for direct SQL execution

3. Programming Languages (For Interfacing & Scripts)

- **SQL** – For data manipulation and retrieval
- **Python** – For scripting, data processing, and report automation
- **PHP/JavaScript** – For building front-end interfaces (if required)

4. Version Control & Collaboration Tools

- **Git** – For source code and database schema management
- **GitHub** – For project repository and collaboration

5. Security & Backup Tools

- **SSL/TLS Encryption** – Ensuring secure communication with the database
- **Automated Backup Mechanism** – Periodic data backups (cron jobs for scheduled backups)
- **User Authentication and Role-Based Access Control (RBAC)** – Restricting access based on user roles