**College Library Management System**

**Objective:** The objective of this project is to develop a **desktop-based Library Management System** using **Java Swing and Oracle Database**. The system will provide efficient management of books, users (students and faculty), and administrative functionalities, ensuring smooth library operations.

**Introduction:** The Library Management System is designed to automate library operations such as book borrowing, returning, fine calculation, and user management. It provides an easy-to-use graphical interface for both users and administrators, improving efficiency and reducing manual work.

**Classes and their Functionalities**

1. **LoginPage**

* Provides the login interface for students, faculty, and admins.
* Authenticates users based on their credentials.
* Redirects to either the **User Dashboard** or **Admin Dashboard** after successful login.

2. **UserDashboard**

* Displays user details (Name, ID, Branch).
* Shows available books with an option to borrow.
* Allows users to check their borrowed books, return books, and view pending fines.

3. **AdminDashboard**

* Provides admin functionalities such as adding/removing books.
* Displays user details and borrowing history.
* Manages fine collection and approval of book requests.

4. **BookTransaction**

* Handles book borrowing, returning, and updating availability in the database.
* Manages fine calculation for overdue books.

5. **DBConnection**

* Establishes and manages a connection with the Oracle Database using JDBC.

**Methods used**

**1. Login.java (Handles user authentication)**

* validateUser(String userID, String password): Verifies login credentials for students, faculty, and admins.
* redirectToDashboard(String userType): Redirects users to their respective dashboards after successful login.

**2. UserDashboard.java (Handles student and faculty functionalities)**

* displayUserDetails(String userID): Fetches and displays user details.
* viewAvailableBooks(): Displays the list of available books.
* borrowBook(String bookID): Initiates the process for borrowing a book.
* viewActivity(): Shows borrowed history, pending returns, and fines.
* logout(): Logs out the user and returns to the login screen.

**3. AdminDashboard.java (Manages library operations)**

* viewBooks(): Displays all books in the library.
* addBook(String title, String author, int copies): Adds a new book to the database.
* removeBook(String bookID): Removes a book from the system.
* viewStudentDetails(): Shows details of all students.
* viewBorrowedHistory(): Displays books borrowed based on a date range.
* manageFines(): Allows admins to view and update fine details.

**4. BookTransaction.java (Handles book borrowing, returning, and fines)**

* borrowBook(String userID, String bookID): Records book borrowing details and updates available copies.
* returnBook(String userID, String bookID): Processes book return and updates the database.
* calculateFine(String userID, String bookID): Computes fines for late returns.
* updateFineStatus(String userID, double amount): Updates the fine payment status.

5. **DatabaseConnection.java (Handles database connectivity)**

* connect(): Establishes a connection with the Oracle database.
* executeQuery(String sql): Executes SELECT queries and returns results.
* executeUpdate(String sql): Executes INSERT, UPDATE, or DELETE queries.
* closeConnection(): Closes the database connection.

**Tables created in DataBase**

**1. students**

* **Columns:** student\_id, name, branch, email, password
* **Functionality:** Stores student details, including their login credentials.

**2. faculty**

* **Columns:** faculty\_id, name, department, email, password
* **Functionality:** Stores faculty details, allowing them to use the system as regular users.

**3. admins**

* **Columns:** admin\_id, name, email, password
* **Functionality:** Stores admin details for managing library operations.

**4. books**

* **Columns:** book\_id, title, author, publisher, publication\_year, isbn, total\_copies, available\_copies
* **Functionality:** Stores details of books available in the library.

**5. borrowed\_books**

* **Columns:** transaction\_id, user\_id, book\_id, borrow\_date, due\_date
* **Functionality:** Stores records of books borrowed by students and faculty, along with due dates.

**6. returned\_books**

* **Columns:** transaction\_id, user\_id, book\_id, return\_date
* **Functionality:** Stores records of books returned by users.

**7. fines**

* **Columns:** fine\_id, user\_id, book\_id, amount, status
* **Functionality:** Tracks fines for overdue books and their payment status.

**8. book\_requests**

* **Columns:** request\_id, user\_id, book\_id, request\_status, approval\_status
* **Functionality:** Stores book borrowing requests from users and tracks approval by admins.

**Flow of the Project**

**1. User Login & Authentication**

* The user is redirected to the **Login Page**, where they enter their **User ID** and **Password**.
* The system verifies the credentials using the **DatabaseConnection** class.
* Based on the **user type** (Student, Faculty, or Admin), the user is redirected to the appropriate **Dashboard**.

**2. User Dashboard (For Students & Faculty)**

* After logging in, users are greeted with a **User Dashboard** that displays:
  + User details (Name, ID, Branch).
  + Available books in the library.
  + Options to **borrow books**, **view borrowed history**, and **check pending fines**.
* **Borrowing Books:**
  + The user selects a book and clicks the **Borrow** button.
  + The system checks **availability** in the books table.
  + If available, a record is added to the borrowed\_books table, and the AVAILABLE\_COPIES count is reduced.
  + The user gets a **confirmation message**.
* **Returning Books:**
  + The user can view of list of books which should be returned.
* **Checking Activity & Fine Details:**
  + Users can view:
    - **Borrowed books history**
    - **Books that need to be returned**
    - **Pending fines**

**3. Admin Dashboard**

* After login, the **Admin Dashboard** is displayed with options:
  + **View Books**
  + **Add Books**
  + **Remove Books**
  + **View Borrowed/Returned History**
  + **View Student Details**
  + **Manage Fines**
* **Managing Books:**
  + The admin can add new books to the library, update book details, or remove books.
  + The books table is updated accordingly.
* **Managing Users:**
  + The admin can view all students, faculty members, and their details.
  + The admin can check their borrowed books and fines.

**Managing Fines:**

* + The admin can view users with pending fines.
  + Once a user pays a fine, the admin updates the **fine status**.

**4. Database Interaction**

* All user and book details are stored in the **Oracle Database**.
* The system interacts with tables like:
  + students, faculty, admins → Stores user credentials and details.
  + books → Stores book details.
  + borrowed\_books, returned\_books → Tracks book transactions.
  + fines → Manages fine records.

**5. Logging Out**

* Users/Admins can click the **Logout** button.
* The system redirects them to the **Login Page**, closing their session.