Phase 4: Frontend Development & JAVA Integration

Student#: 220274536
Sur (Family) Name: Markham
Given Name: Sienna

Student#: 219447192
Sur (Family) Name: Ryan
Given Name: Eamon

Student#: 220416038
Sur (Family) Name: Mollah
Given Name: Mahjabin

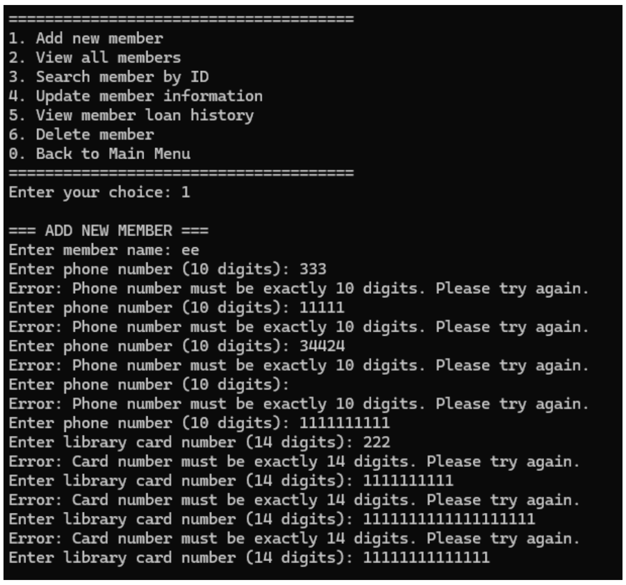
Section: D/Phase 4
Github Repository: https://github.com/eamontryan/DIGT_3107_team_3

Objective:

Demonstrate a working application that connects a **Java console frontend** with a **MySQL backend** using **JDBC**. Showcase functionality and database interaction through a **5-minute video recording**.

Tasks:

Tasks	Details
1. Java Console Application (Frontend)	- Text-based menu interface - Options like insert, view, update records
2. Database Integration	- Connected using JDBC - Executes SQL queries (SELECT, INSERT, etc.)

3. Final Video Demo (5 Minutes)	- Recording that shows: → App launch and menu → Performing operations → Database results/output → No slides needed phase4finaldemo.mp4 ***Note:** In the video, when setting the Phone Number and Library Card number, you can see that an example ‘911’ and ‘311’ were entered and accepted. This will not be the case in the current state of our GitHub repo, as we have updated to enforce strict minimum digit requirements for Phone Number (10) and Library Card Number (14). See below: 

```
======================================
1. Add new member
2. View all members
3. Search member by ID
4. Update member information
5. View member loan history
6. Delete member
0. Back to Main Menu
======================================
Enter your choice: 1

=== ADD NEW MEMBER ===
Enter member name: ee
Enter phone number (10 digits): 333
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits): 11111
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits): 34424
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits):
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits): 1111111111
Enter library card number (14 digits): 222
Error: Card number must be exactly 14 digits. Please try again.
Enter library card number (14 digits): 1111111111
Error: Card number must be exactly 14 digits. Please try again.
Enter library card number (14 digits): 1111111111111111111
Error: Card number must be exactly 14 digits. Please try again.
Enter library card number (14 digits): 11111111111111
```

4. Codebase Submission	- Source code for Java console app - SQL scripts for DB schema and test data
5. Final Report (PDF)	- Overview of system - ER diagram + schema - Functional description - Explanation of how Java connects to database - Screenshots (optional backup for video)

6. README / Setup Instructions	- How to run the app - Additional .jar files (if needed) - Database login info (test user)

Final Report

1. Overview of system:

The Library Management System (LMS) digitizes book inventory, loan management, overdue tracking, member data, circulation history, and fines for Westbridge University Library.

Phase 4 builds on the database designed in Phases 1–3 by integrating a Java console application that interacts with MySQL through JDBC.

This phase demonstrates backend + frontend integration, allowing users to perform live transactions via console menus.

The Java system supports:

- Viewing books, members, loan records and circulation details
- Registering new members and new books
- Borrowing and returning books with automated stock updates
- Viewing overdue items and fines
- Running advanced library queries (Phase 3 SQL logic)

Execution is menu-driven through the Main class, which routes user input to service classes:

- BookService – inventory access, adding books, availability lookup
- LoanService – borrowing, returning, transactions, fine calculation
- MemberService – CRUD operations and member loan history
- QueryService – analytical/statistical reports (top books, overdue, never borrowed, etc.)

JDBC is used to execute SQL queries and return formatted console output.

2. ER diagram + schema:

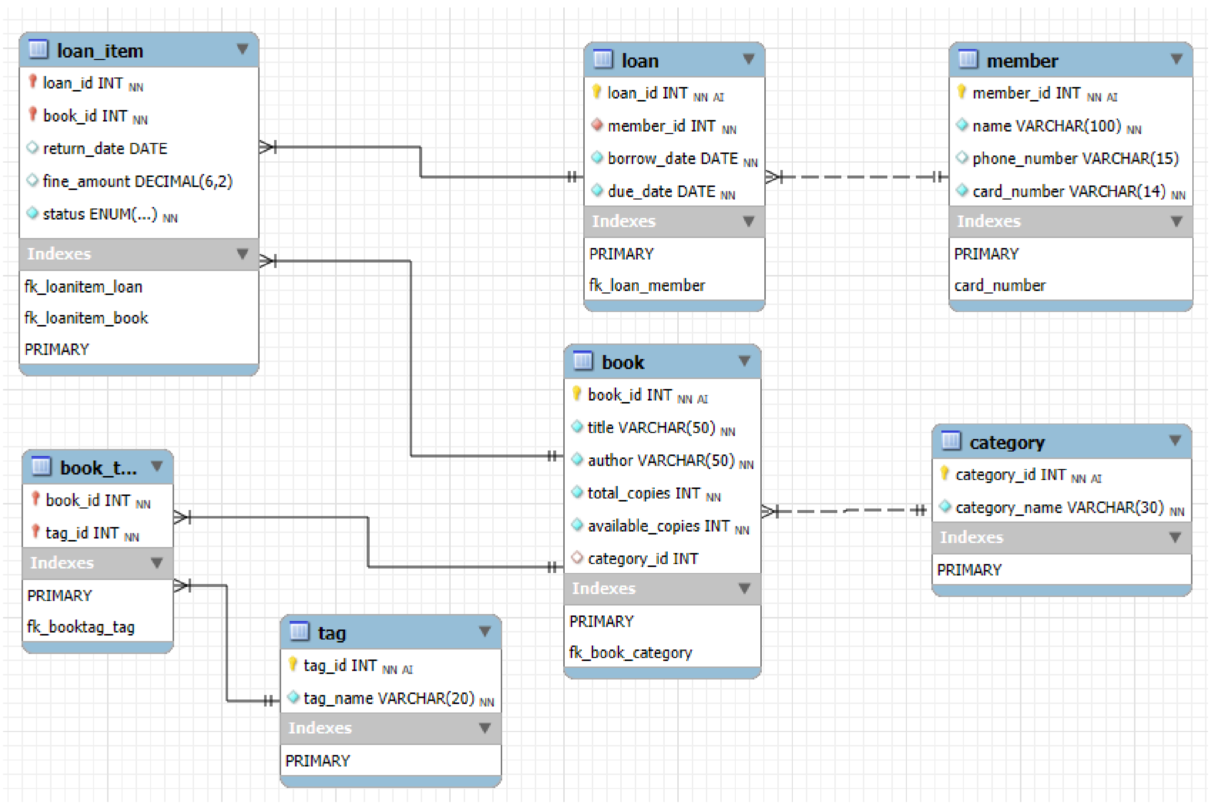
Entities:

- **book**(book_id, title, author, total_copies, available_copies, category_id)
- **member**(member_id, name, phone_number, card_number)
- **loan**(loan_id, member_id, borrow_date, due_date)
- **loan_item**(loan_id, book_id, return_date, fine_amount, status)
- **category**(category_id, category_name)
- **tag**(tag_id, tag_name)

- **book_tag**(book_id, tag_id)

Key business rules (Phase 1) in Phase 4 Java logic:

- Loan due date = borrow date + 15 days (LoanService.borrowBook)
- Fine = $10/day late (LoanService.returnBook & QueryService.overdue calculations)
- Cannot borrow if available_copies = 0 (checked in LoanService.borrowBook)
- A loan supports multiple books (loan_item table representing 1:N relationship)



3. Functional description:

The Java console program provides a structured multi-layer menu system implemented in Main.java:

Main Menu:

1. Queries & Reports
2. Member Management
3. Book Management
4. Loan Management
0. Exit

Queries & Reports Menu:
(implemented inside QueryService)

- List overdue books and fines
- Find most borrowed books
- List currently borrowed books
- Retrieve a member’s borrowing history
- Show books due in next 3 days
- Identify books never borrowed

Each option prints formatted query results from MySQL.

Member Management Menu:
(implemented inside MemberService)

- Add new member
- View all members
- Search member by ID
- Update member information
- View member loan history
- Delete member

Duplicate entry validation is handled for card numbers.
Deletion is prevented when outstanding loans exist (FK constraint).

Book Management Menu
(implemented inside BookService)

- View all books
- Add a new book
- Check book availability / circulation metrics

BookService shows:

- current availability
- total loans
- warning if availability + borrowed != total (data integrity check)

Loan Management Menu
(implemented inside LoanService)

- Borrow a book
- Return a book

Borrow Workflow:

- Prompt member + book ID
- Check availability
- Insert loan + loan_item
- Update book stock
- Transaction control (commit / rollback)

Return Workflow:

- Prompt loan + book ID
- Check due date
- Compute fine at $10/day late
- Update loan_item with return_date and fine
- Increase available stock

Both workflows use:

- transaction handling

- multiple SQL updates
- formatted console outputs

4. Explanation of how Java connects to database:

The system uses the following JDBC components:

- Connection
- PreparedStatement
- ResultSet
- DriverManager

Database Connection Layer (Database.java)

- Reads credentials from .env
- Loads com.mysql.cj.jdbc.Driver
- Provides reusable getConnection()
- Displays config error messages if .env missing

This promotes:

- secure password handling
- environment-based configuration
- centralized DB access

SQL Execution Pattern:
Example SELECT (BookService / QueryService):

```
PreparedStatement stmt = conn.prepareStatement(sql);
ResultSet rs = stmt.executeQuery();
```

Example INSERT / UPDATE (MemberService, LoanService):

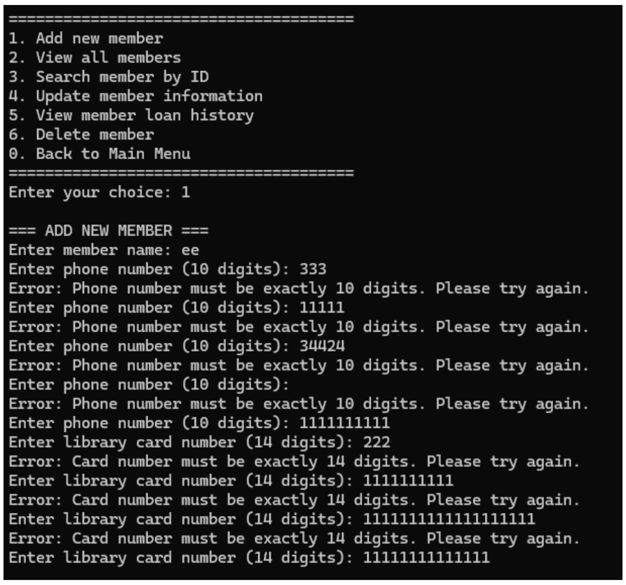
```
PreparedStatement pstmt = conn.prepareStatement(sql,
Statement.RETURN_GENERATED_KEYS);
pstmt.executeUpdate();
```

Exception + Transaction Handling

- try/catch SQLException
- rollback on transaction failure (LoanService)
- formatted user messages
- input validation wrappers

5. Screenshots (optional backup for video)

*__Note:__ In the video, when setting the Phone Number and Library Card number, you can see that an example ‘911’ and ‘311’ were entered and accepted. This will not be the case in the current state of our GitHub repo, as we have updated to enforce strict minimum digit requirements for Phone Number (10) and Library Card Number (14). See below:



```
======================================
1. Add new member
2. View all members
3. Search member by ID
4. Update member information
5. View member loan history
6. Delete member
0. Back to Main Menu
======================================
Enter your choice: 1

=== ADD NEW MEMBER ===
Enter member name: ee
Enter phone number (10 digits): 333
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits): 11111
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits): 34424
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits):
Error: Phone number must be exactly 10 digits. Please try again.
Enter phone number (10 digits): 1111111111
Enter library card number (14 digits): 222
Error: Card number must be exactly 14 digits. Please try again.
Enter library card number (14 digits): 1111111111
Error: Card number must be exactly 14 digits. Please try again.
Enter library card number (14 digits): 1111111111111111111
Error: Card number must be exactly 14 digits. Please try again.
Enter library card number (14 digits): 11111111111111
```

Deliverables for Phase 4:

A. Java application with sql queries
B. Readme file for setup instructions
C. **5-minutes video** recording (as described above)
D. Create **team#_phase4.pdf** containing your project final report

Submission Instructions:

Upload **team#_phase4.pdf and other .sql/Java files** to your **GitHub repository** and then **post** your GitHub repository link on **eClass.**

Video Submission Options:

· Upload the video to Google Drive and share the link
· OR upload it to your GitHub repository
· OR upload directly to eClass

Must Have:

Please include a cover page when submitting your work, with the student number, name, and section filled out for all team members, formatted as follows.

Student#:

Sur (Family) Name:

Given Name:

Section: