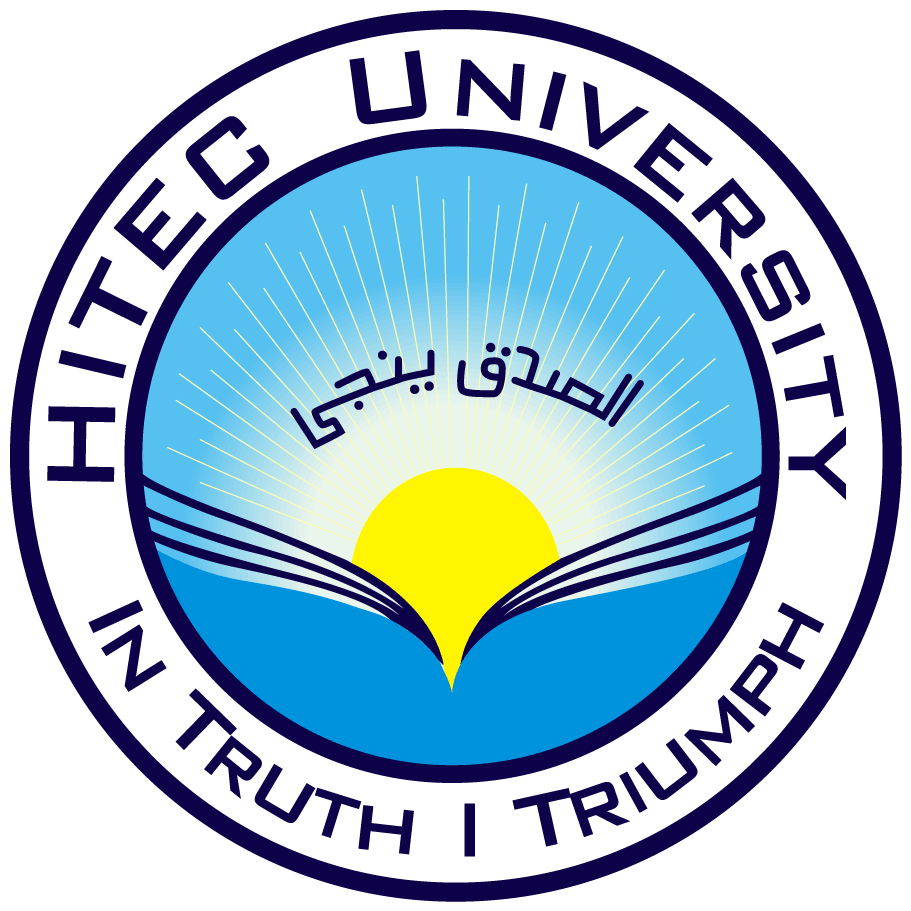
DATA BASE LAB



Semester Project Report

**Submitted to:**

Sir Khurram Shazad

**Submitted by:**

Muhammad Muneeb Iqbal 22-CS-026

Zaraq Ali Khan 22-CS-010

Taimoor Hassan Awan 22-CS-050

**Section: (B)**

**Department of Computer Science**

**HITEC University, Taxila**

Contents

[1. Introduction 3](#_Toc187416579)

[2. Problem Statement 3](#_Toc187416580)

[3. Objectives 3](#_Toc187416581)

[4. Database Design 3](#_Toc187416582)

[4.1 Entity-Relationship Diagram 3](#_Toc187416583)

[4.2 Normalization 4](#_Toc187416584)

[4.3 SQL Schema 4](#_Toc187416585)

[5. Application Development 4](#_Toc187416586)

[5.1 Technology Stack 4](#_Toc187416587)

[5.2 Application Architecture 4](#_Toc187416588)

[6. Features and Functionalities 4](#_Toc187416589)

[6.1 Books Management 4](#_Toc187416590)

[6.3 Transactions Management 5](#_Toc187416591)

[6.4 Error Handling 5](#_Toc187416592)

[7. Testing and Validation 5](#_Toc187416593)

[8. Challenges and Solutions 5](#_Toc187416594)

[8.1 Challenge: Handling database relationships 5](#_Toc187416595)

[8.2 Challenge: Input validation 5](#_Toc187416596)

[9. Conclusion 5](#_Toc187416597)

[10. Appendix 6](#_Toc187416598)

[10.1 Screenshots 6](#_Toc187416599)

[10.2 Code Repository 7](#_Toc187416600)

[10.3 EERD 7](#_Toc187416601)

# 

# 1. Introduction

The Library Management System (LMS) is a web-based application designed to manage and automate the daily operations of a library. It allows users to manage books, members, and transactions efficiently while ensuring data integrity and ease of access.

# 2. Problem Statement

Traditional library management methods often involve manual record-keeping, which can lead to errors, inefficiencies, and difficulty in tracking data. This project aims to address these issues by developing a robust database-driven application for managing library operations.

# 3. Objectives

* To design and implement a normalized database for a library management system.
* To develop a user-friendly web-based portal using Flask.
* To provide CRUD (Create, Read, Update, Delete) functionalities for managing books, members, and transactions.
* To ensure data validation and error handling.

# 4. Database Design

## 4.1 Entity-Relationship Diagram

The ER diagram includes the following entities:

* Books: BookID, Title, AuthorID, CategoryID, ISBN, Availability.
* Members: MemberID, Name, Contact, MembershipDate.
* Transactions: TransactionID, BookID, MemberID, IssueDate, ReturnDate.

## 

## 4.2 Normalization

The database was normalized to the Third Normal Form (3NF):

* Removed repeating groups and ensured atomic columns (1NF).
* Removed partial dependencies by splitting tables (2NF).
* Eliminated transitive dependencies (3NF).

## 4.3 SQL Schema

The database schema includes tables for:

* Books
* Members
* Transactions

# 5. Application Development

## 5.1 Technology Stack

* Backend: Flask (Python)
* Frontend: HTML, CSS, Bootstrap
* Database: SQLite

## 5.2 Application Architecture

* Model: SQLite database for data storage.
* View: HTML templates for user interface.
* Controller: Flask routes for handling user actions.

# 6. Features and Functionalities

## 6.1 Books Management

* Add, update, delete, and view books.
* ISBN validation.6.2 Members Management
* Add, update, delete, and view members.
* Contact validation.

## 6.3 Transactions Management

* Record book issues and returns.
* Update return dates.

## 6.4 Error Handling

* Input validation for forms.
* Handling database constraints.

# 7. Testing and Validation

The application was tested for:

* Functional correctness (CRUD operations).
* Data validation (e.g., ISBN format, contact number length).
* Error handling (e.g., duplicate entries, missing fields).

# 8. Challenges and Solutions

## 8.1 Challenge: Handling database relationships

Solution: Used foreign keys to enforce referential integrity.

## 8.2 Challenge: Input validation

Solution: Implemented client-side and server-side validation for all forms.

# 9. Conclusion

The Library Management System successfully streamlines library operations by automating book, member, and transaction management. The system ensures data integrity, user-friendly interaction, and robust error handling.

# 10. Appendix

## 10.1 Screenshots

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## 10.2 Code Repository

<https://github.com/immuneeb64/DBS-Project>

## 10.3 EERD

**A diagram of a diagram

Description automatically generated**