# Book Share

**MINI PROJECT LAB REPORT**

*Submitted by*

**Ashwini Bisanalli, Anjali k, Deepak Denny**
**USN: 24MCAR0121,24MCAR0186,24MCAR0187**

*in partial fulfillment for the award of the degree of*

**MASTER OF COMPUTER APPLICATIONS**

**DEPARTMENT OF COMPUTER SCIENCE AND IT**

**JAIN KNOWLEDGE CAMPUS**
**JAYANAGAR 9TH BLOCK**
**BANGALORE- 560069**

**November - 2025**

JAIN DEEMED-TO-BE UNIVERSITY School Of Computer Science and IT

**DEPARTMENT OF COMPUTER SCIENCE AND IT**

**Jain Knowledge Campus**
**Jayanagar 9th Block, Bangalore-560069**

This is to certify that the project entitled

**Book Share**

***is the bonafide record of project work done by***

**Ashwini Bisanalli, Anjali k, Deepak Denny**
**USN: 24MCAR0121,24MCAR0186,24MCAR0187**

**MCA**

During the year

**2025-2026**

**Dr R Kamalraj**
Guide/Mentor
Department of Computer Science and IT
JAIN (Deemed-to-be University)

**Dr. Murugan R**
Programme Head- MCA,
Department of Computer Science and IT
JAIN (Deemed-to-be University)

JGi JAIN DEEMED-TO-BE UNIVERSITY | School Of Computer Science and IT

# CERTIFICATE

This is to certify that Ashwini Bisanalli, Anjali k, Deepak Denny, USN: 24MCAR0121, 24MCAR0186, 24MCAR0187 of MCA programme in the Department of Computer Science and IT has fulfilled the Mini Project Lab (23MCAIS303MPL) requirements prescribed for the MCA Programme in JAIN (Deemed-to-be University).

The Mini Project entitled, “Book Share” was carried out under my direct supervision. No part of the dissertation was submitted for the award of any degree or diploma prior to this date.

**Dr R Kamalraj**
Guide / Mentor
JAIN (Deemed-to-be University)

**Mini Project Lab Viva-voce:**

| **Name of the Examiner** | **Signature with Date** |
|---|---|
| 1........................................... | ........................................... |
| 2........................................... | ........................................... |

JAIN DEEMED-TO-BE UNIVERSITY | School Of Computer Science and IT

## DECLARATION

We affirm that the project work titled “Book Share”, being submitted in partial fulfillment for the award of MASTER OF COMPUTER APPLICATIONS is the original work carried out by us. It has not formed the part of any other project work submitted for award of any degree or diploma, either in this or any other University.

Ashwini Bisanalli-24MCAR0121

Anjali k-24MCAR0186

Deepak Deny-24MCAR0187

JGi JAIN DEEMED-TO-BE UNIVERSITY | School Of Computer Science and IT

## ACKNOWLEDGEMENT

We would like to acknowledge the following people, who have encouraged, guided and helped to accomplish our report to award our degree at the JAIN (Deemed to be University), Department of Computer Science and IT, School of Computer Science and IT:

1. Mini Project mentor Dr R Kamalraj for guiding us through pivotal moments of our study and professional career and for always being there to make sure that our progress was reviewed, documented and acknowledged. His/Her encouragement has been the greatest source of inspiration and confidence for carrying out our project work.
2. Faculty and staff members of **Department of Computer Science and IT** for sharing their expertise and always show their interests in our work.
3. Finally, we would like to thank our family, to whom this work is dedicated, for their support and encouragement during these years.

**Special Thanks to:**

❖ Dr. Sagar Gulati, Director, School of Computer Science and IT, JAIN (Deemed-to-be University)

❖ Dr. Ananta Charan Ojha, Deputy Director, School of Computer Science and IT, JAIN (Deemed-to-be University)

❖ Dr. K Suneetha, Head, Department of Computer Science and IT, JAIN (Deemed-to-be University)

❖ Dr. MURUGAN R, Programme Head, MCA Programme, Department of Computer Science and IT, JAIN (Deemed-to-be University)

❖ Dr. Pushpa J, Assistant Professor, Mini Project Coordinator (ISMS), Department of Computer Science and IT, JAIN (Deemed-to-be University)

# ABSTRACT

Access to physical books is often restricted due to geographical limitations, cost barriers, and limited availability. Traditional book-sharing models depend heavily on manual coordination and lack scalability. To address these challenges, BookShare is developed as an interactive, community driven digital platform that enables users to publish books they own, browse available titles, request to borrow books, and return them through a structured online environment.

BookShare integrates modern full-stack technologies React + TypeScript + Tailwind CSS for the frontend, and Node.js + Express.js + TypeScript with MongoDB Atlas for the backend. The platform supports user registration, book listing, borrowing workflows, approval mechanisms, and system-maintained book statuses. It provides seamless navigation, dynamic routing, a clean user interface, and efficient backend API integration.

The system follows a modular architecture featuring reusable UI components, RESTful APIs, secure authentication, and cloud-hosted database connectivity. BookShare improves accessibility, encourages sharing habits, and fosters a collaborative reading ecosystem.

This report documents the system analysis, design, architecture, implementation details, database models, testing methodologies, and outcomes of the BookShare project. It demonstrates how modern web technologies can be integrated to solve real-world problems in resource sharing and digital learning environments.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

## 1. BookShare

### 1.1 Background of the Project

Books and learning resources have traditionally been accessed through physical libraries, personal book collections, or direct borrowing among peers. While these methods have served generations of learners, they come with several limitations such as geographical dependency, limited availability, manual coordination, and delayed access. In an era driven by digital transformation and online information exchange, there is a growing need for systems that facilitate seamless resource sharing without physical constraints.

BookShare is designed as a digital book-sharing and exchange system that enables users to publish the books they own, browse books shared by others, request to borrow them, and return them after use. It combines the benefits of an online catalog, peer-to-peer sharing, and community-driven participation into a single platform. By leveraging modern web technologies, BookShare transforms traditional book exchange practices into a structured, scalable, and user-friendly digital environment.

The system focuses on accessibility, simplicity, and collaboration ensuring that students, readers, and educational communities can easily exchange knowledge without barriers.

### 1.2 Problem Statement

Traditional book-sharing processes rely heavily on physical proximity, manual requests, and availability of printed copies. These outdated methods result in:

- Limited access due to geographical restrictions
- Dependency on physical presence
- Poor visibility of available resources
- Difficulty in locating specific books quickly
- Inefficient coordination between lenders and borrowers
- No centralized platform for book-sharing
- No automated system for borrow approval, return tracking, or book status updates

To overcome these limitations, there is a need for a **digital, scalable, and user-friendly book exchange system** that organizes resources systematically and automates the sharing workflow.

### 1.3 Objectives of the Project

The primary objectives of BookShare are:

1. To provide a centralized digital platform for publishing, browsing, and sharing books.
2. To enable seamless book exchange workflows, including borrow requests, approvals, and returns.

3. To design an intuitive and responsive user interface using React, TypeScript, and Tailwind CSS.
4. To implement secure backend APIs using Express.js and TypeScript to manage books, users, and transactions.
5. To store and retrieve data efficiently using MongoDB database.
6. To maintain accurate book status (Available, Borrowed).
7. To encourage collaborative reading habits through community-driven sharing.
8. To ensure scalability and easy extension for future enhancements like reviews, recommendations, and notifications.

## 1.4 Scope of the System

The BookShare system includes:

User Functionalities

- User Registration and Login
- Profile and dashboard access
- Browse all available books
- View detailed book information
- Request to borrow books
- Return borrowed books

Book Management

- Add new books
- Upload book cover images
- Modify book details
- Delete books
- Maintain accurate availability status

System-Controlled Features

- Approval or rejection of borrow requests
- Update book status dynamically
- Track borrower history
- Cloud database integration
- REST API-based communication

The scope excludes advanced features such as payment gateways, AI recommendations, or e-book hosting, which may be added in future enhancements.

## 1.5 Methodology

The  The development of BookShare follows a modular and iterative approach based on Software Development Life Cycle (SDLC):

1. Requirement Analysis – Identify functional and non-functional needs.
2. System Analysis – Study problems in existing systems and justify the proposed solution.
3. Design Phase – Architectural design, database modeling, UML diagrams.
4. Implementation – Develop frontend (React + TS + Tailwind CSS) and backend (Express + TS +

MongoDB).
5. Testing – Validate workflows, UI, APIs, and integrations.
6. Deployment – Configure cloud database and prepare environment for hosting.
7. Review & Documentation – Finalize project report and demonstration.

## 1.6 Organization of the Report

The report is structured into eight detailed chapters:

- Chapter 1 – Introduction
- Chapter 2 – Literature Survey
- Chapter 3 – System Analysis
- Chapter 4 – System Design & Architecture
- Chapter 5 – System Requirements
- Chapter 6 – Implementation
- Chapter 7 – Results & Screenshots
- Chapter 8 – Conclusion & Future Enhancements

# CHAPTER 2
# LITERATURE REVIEW

## 1.1 Evolution of Digital Book Systems

The concept of digital libraries and online book-sharing platforms has rapidly evolved over the last two decades. Early digital systems focused on offering static catalogs and basic information retrieval. Over time, advancements in web technologies enabled interactive, real-time, and community-driven systems. Key limitations of earlier models include:

- Outdated user interfaces
- Poor search capabilities
- Lack of sharing or exchange mechanisms
- Difficulty in categorizing and organizing books
- Limited user engagement features

Modern systems adopt dynamic frontend frameworks, responsive design principles, and cloud-based data management to improve accessibility and scalability.

## 1.2 Online Book Sharing Models

Several online book-sharing platforms exist but often lack one or more essential features:

| Platform Type | Limitations |
|---|---|
| Library Portals | Restricted to members, limited hours |
| Online Bookstores | Paid model; no sharing |
| Reading Communities (Goodreads, etc.) | Reviews only; no borrowing |
| Local Sharing Apps | Limited reach; manual coordination |

BookShare aims to combine the strengths of these systems while addressing their limitations.

## 1.3 Role of Web Technologies

### Frontend Technologies

React + TypeScript enables component-based UI development, ensuring:

- Fast rendering
- Modular design
- Reusability of components
- Strong typing (error-free code)

Tailwind CSS ensures:

- Responsive and mobile-friendly layouts
- Utility-first styling
- Consistent UI experience

**Backend Technologies**

Node.js + Express.js provides:

- Scalable server-side architecture
- RESTful APIs
- High-speed asynchronous operations

TypeScript improves backend reliability through static type checking.

**Database Technology**

MongoDB Database offers:

- Local hosted database
- Flexible document-based storage
- High availability and scalability
- Easy integration with Node/Express via Mongoose

## 1.4 Summary of Literature Survey

Research shows that accessible and collaborative learning ecosystems significantly improve knowledge distribution among students and communities. Key findings include:

- Digital libraries increase access to learning materials.
- Peer-to-peer sharing encourages community participation.
- Cloud-hosted systems offer better reliability and scalability.
- Component-driven UIs improve user engagement and learning efficiency.
- RESTful APIs enhance interoperability between client and server.

These findings strongly support the need for a system like BookShare.

## 1.5 Summary of Literature Survey

From the literature examined:

- Existing systems are often limited, outdated, or lack exchange features.
- Advanced web technologies enable better user experience and scalability.
- Community-driven sharing is crucial for effective resource distribution.
- BookShare fills a gap by providing an organized, interactive, cloud-integrated book exchange platform.

# CHAPTER 3
# SYSTEM ANALYSIS

## 3.1 Existing System

The existing book-sharing methods rely heavily on personal communication, making the process slow and unpredictable. Users usually have no idea which books are available or who currently holds them, resulting in wasted time and repeated follow-ups. This lack of transparency often leads to books being misplaced or forgotten, and borrowing cycles become difficult to track. These shortcomings clearly show the need for a digital system that centralizes all book-sharing activities. The traditional book-sharing model includes:

- Manual exchange between friends/classmates
- Physical library borrowing
- Sharing through WhatsApp groups or personal networks

**Limitations**

- Time-consuming coordination
- Limited awareness of available books
- No easy way to track borrowed/returned books
- No digital record of transactions
- Physical availability and location constraints

## 3.2 Proposed System

The proposed BookShare system overcomes these limitations by introducing a structured, automated workflow. Users can publish their books, browse available ones, and request or return them without manual coordination. The system creates a transparent environment where availability, status, and borrow history are clearly visible to all authorized users. With an intuitive interface and real-time updates, BookShare simplifies the entire process and makes book-sharing efficient and reliable. BookShare introduces a digital, structured, and automated way of sharing books:

**Key Features**

- Users publish books they own
- Others can browse and request them

- Owners approve or deny borrow requests
- System updates statuses automatically
- Borrower returns book → status becomes "Available"

This ensures transparency, accessibility, and ease of use.

## 3.3 Feasibility Study

### 3.3.1 Comparison of Existing vs. Proposed System

| **Aspect** | **Existing System** | **BookShare (Proposed)** |
|---|---|---|
| Accessibility | Limited, physical | Available 24/7 |
| Book Availability | Not visible | Full catalog |
| Borrow Request | Manual | Automated |
| Tracking | No tracking | System logs all |
| User Experience | Low | Modern UI |
| Scalability | Very low | High (Cloud DB) |

Comparing both systems highlights how the proposed digital solution significantly enhances accessibility and efficiency. While traditional methods depend on physical proximity and manual communication, BookShare provides a standardized platform accessible from anywhere. Automation reduces human error, while cloud storage ensures data consistency. This comparison demonstrates how BookShare modernizes book-sharing to meet current technological expectations.

### 3.3.2 Functional Requirements

The functional requirements of BookShare focus on delivering features that support both lenders and borrowers seamlessly. Each function is designed to reduce manual tasks and ensure smooth interaction between components. By clearly defining features such as browsing, requesting, approving, and returning books, the system ensures that all user actions follow a predictable and organized flow. This clarity enhances usability and improves the overall user experience.

**User Requirements**

- Register/Login

- Browse books
- View details
- Request/Return books
- Manage profile

**System Requirements**

- Store user/book data
- Maintain borrow status
- Secure authentication
- API communication between frontend & backend

**Admin/Owner Requirements**

- Approve/Reject borrow requests
- Add/Remove books

### 3.3.3 Non-Functional Requirements

Non-functional requirements ensure the system meets quality expectations beyond basic functionality. Performance, reliability, security, and usability are essential for maintaining user trust. BookShare must be fast, stable, and accessible across devices, ensuring consistent performance even as the number of users grows. These requirements guide the technical decisions that shape the system's architecture and behavior. The system is designed for users with basic computer and internet skills. The interface follows standard web application patterns:

- Simple login and signup forms
- Cards and lists in dashboard
- Familiar word-processor-like editor
- Easy links to analytics and settings

Because of this familiarity, the system is operationally feasible and can be used by students without heavy training.

## 3.4 Problem Definition and Requirements Overview

- **Performance:** Quick page loads, responsive UI
- **Reliability:** Cloud database availability
- **Security:** JWT authentication, protected endpoints
- **Scalability:** Modular code, reusable components
- **Usability:** Clean UI, simple navigation

## 3.5 Feasibility Study

The feasibility analysis confirms that BookShare is both technically and operationally practical. With widely supported tools like React, Express.js, and MongoDB, the system can be developed efficiently without high costs. The operational workflow is simple enough for users with minimal digital experience, and the economic feasibility is high since all major technologies are open-source. This ensures that the system can be maintained and expanded easily.

### Technical Feasibility

- React, Express, TypeScript, MongoDB are industry-standard.
- Easy integration ensures smooth development.

### Economic Feasibility

- All technologies used are free and open-source.
- MongoDB free tier is sufficient for project.

### Operational Feasibility

- Clean UI ensures ease of use even for non-technical users.

# CHAPTER 4

# SYSTEM DESIGN AND ARCHITECTURE

System design defines the structure, components, data flow, and working of BookShare. It ensures that the platform is scalable, modular, secure, and easy to maintain. This chapter presents the overall architecture, UML diagrams, database models, and logical workflows.

## 4.1 High-Level System Architecture

The BookShare system follows a three-tier architecture:

**1. Presentation Layer – Frontend (React + TypeScript + Tailwind CSS)**

- Renders UI components
- Handles routing (Home, Categories, Book List, Book Detail, Dashboard)
- Sends API requests to backend
- Displays dynamic responses and real-time book statuses

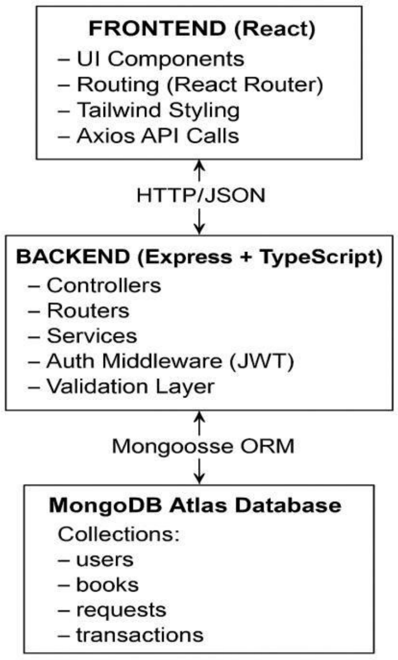
**2. Application Layer – Backend (Node.js + Express.js + TypeScript)**

- Exposes RESTful APIs for CRUD operations
- Manages user authentication, books, requests, and transactions
- Handles book status and borrow/return workflows
- Validates data and ensures secure API calls (JWT)

**3. Data Layer – MongoDB Database**

- Stores all user, book, and transaction data
- Ensures high availability and scalability
- Uses Mongoose schemas to structure data

### 4.2 Architecture Diagram (Text-Based UML)



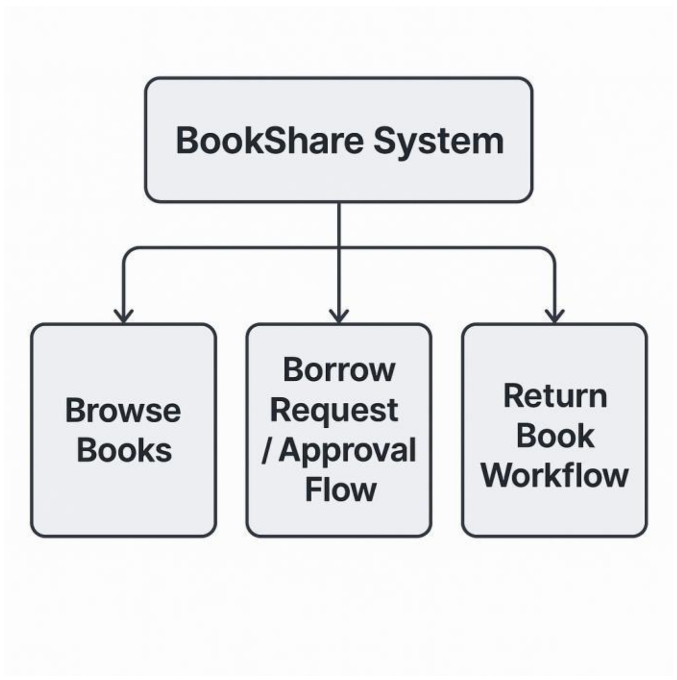
### 4.3 Use Case Diagram (Text UML)

**Actors**

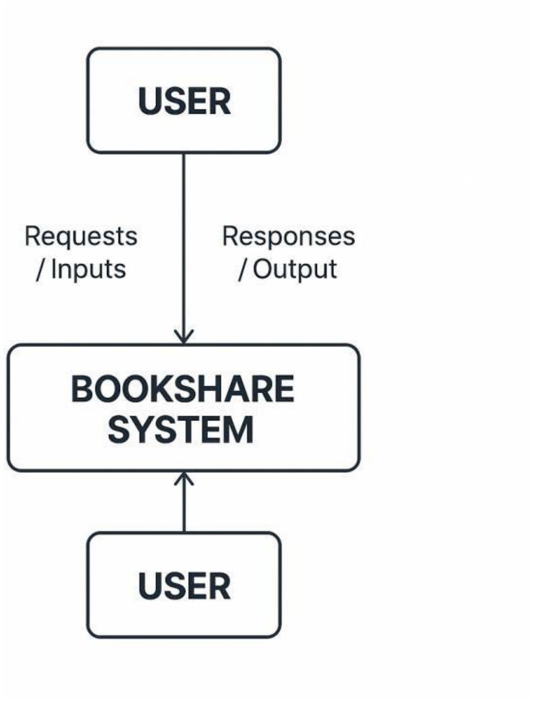
- **User (Borrower)**
- **Owner (Lender)**
- **System (Automated)**

**Use Cases**

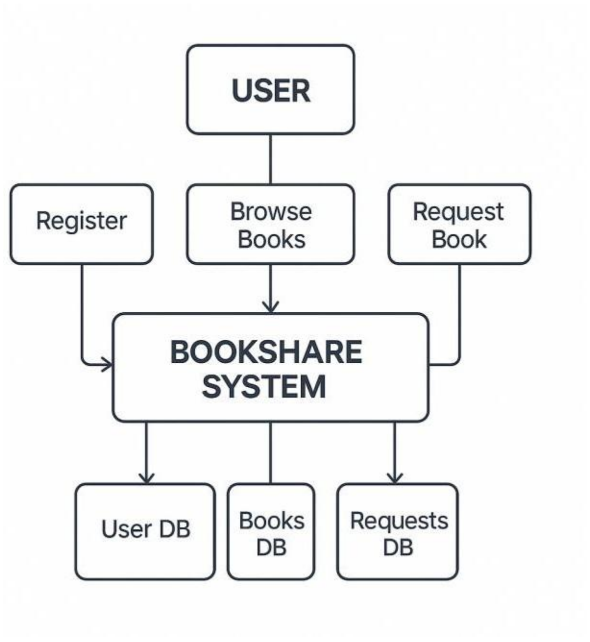
- Register/Login
- Browse books
- View book details
- Add book
- Request book
- Approve/Reject request
- Return book
- Update status



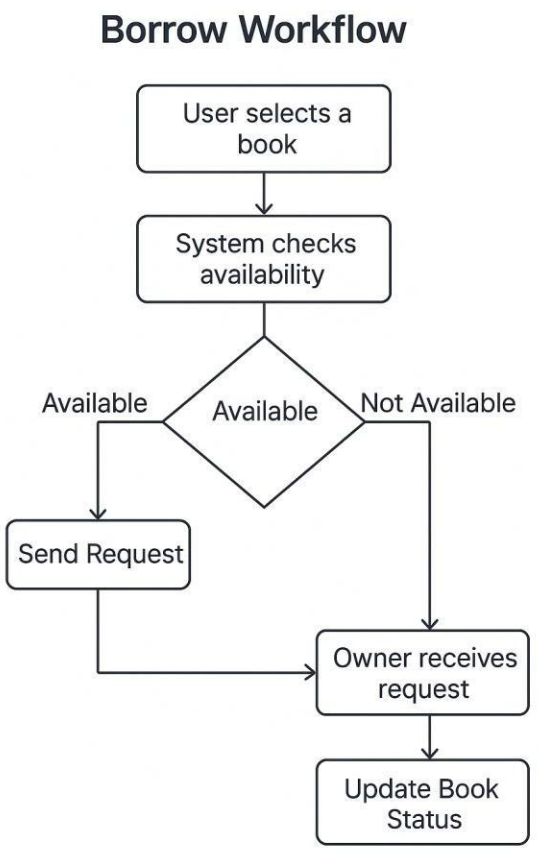
### 4.3.1 DFD Level 0 – Context Diagram



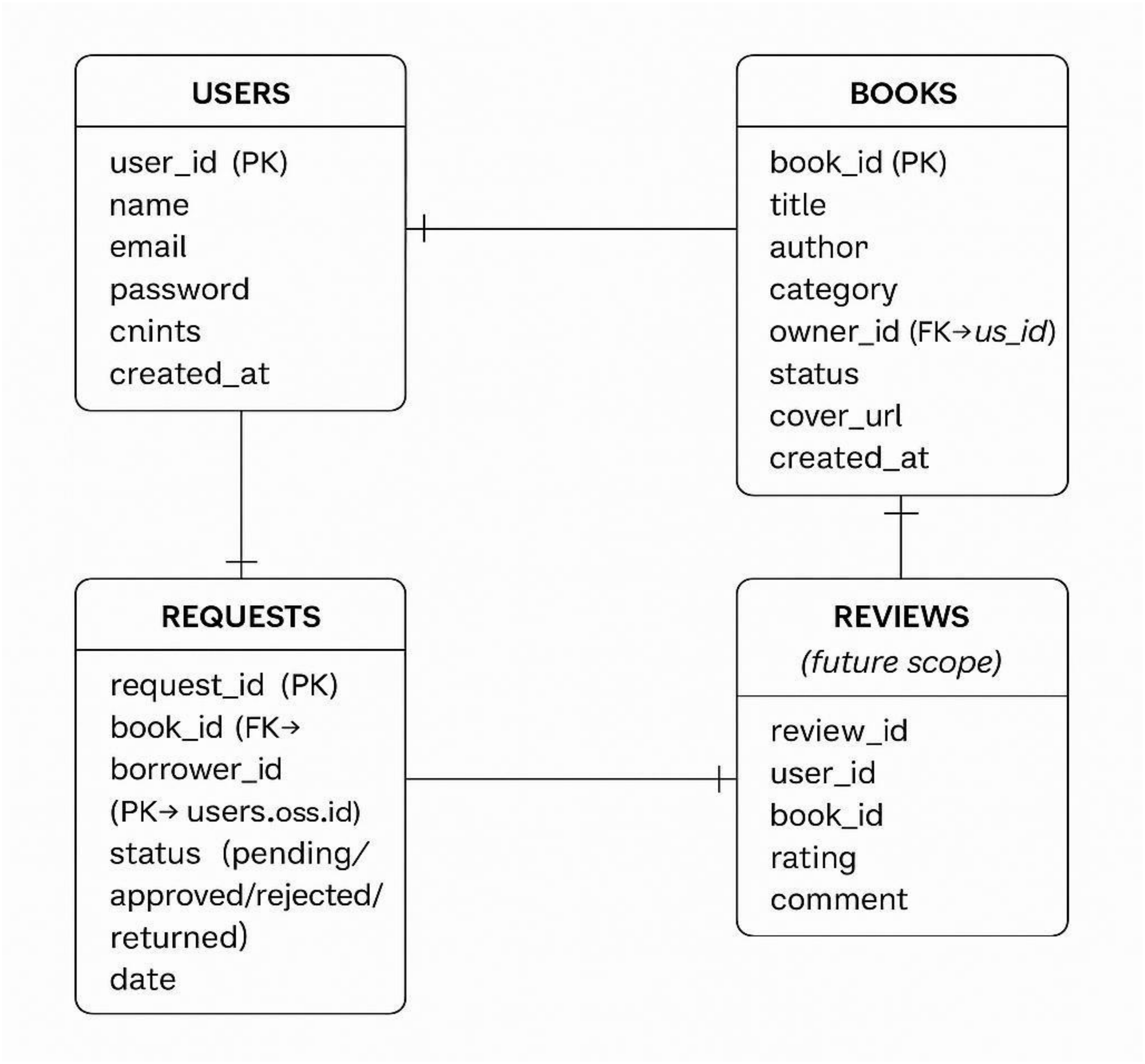
### 4.3.2 Level 1 DFD – Detailed View



## 4.4 Activity Diagram – Borrow Workflow



**4.5 ER Diagram (Text-Based)**



**Relationships:**

- One User → Many Books
- One User → Many Requests
- One Book → Many Requests

# CHAPTER 5
# SYSTEM REQUIREMENTS

## 5.1 Hardware Requirements

**Minimum**

- Processor: Dual Core 2.0 GHz
- RAM: 4 GB
- Storage: 5 GB
- Display: 1024×768 resolution
- Internet: Basic connectivity

**Recommended**

- Processor: Quad-Core 2.5+ GHz
- RAM: 8 GB or above
- Storage: 20 GB
- Stable broadband internet

## 5.2 Software Requirements

**Operating System**

- Windows 10 / 11
- Ubuntu 20.04+
- macOS Monterey or above

**Frontend Tools**

- Node.js (v16+)
- React + TypeScript
- Tailwind CSS
- Axios
- Vite

**Backend Tools**

- Node.js + Express.js
- TypeScript
- Mongoose (MongoDB ODM)
- JWT Authentication
- Dotenv

**Database**

- MongoDB Database

**Development Tools**

- Visual Studio Code
- Git
- Postman (API Testing)

### 5.3 Software Component Features

Frontend Components

- Navbar – global navigation links
- BookCard – compact book preview widget
- BookList – grid display with filters
- BookDetail – complete book info page
- AddBook Form – upload & submit books
- Dashboard – borrow history & account info

# CHAPTER 6
# IMPLEMENTATION AND TESTING

## 6.1 Module Description

This chapter explains how each module was built using React, TypeScript, Tailwind CSS, Express.js, and MongoDB.

### 6.1.1 Frontend Implementation

The frontend is implemented using React + TypeScript, ensuring strong typing, fewer runtime errors, and better scalability.

**Key Modules**

**1. App Component**

- Root of the application
- Wraps all routes inside BrowserRouter
- Applies global styles

**2. Navbar Component**

- Contains navigation links
- Responsive using Tailwind utility classes
- Displays Home, Categories, Add Book, Dashboard

**3. Home Page**

- Showcases book categories
- Hero section and quick links
- Fetches featured books from API

**4. Categories Page**

- Displays list of preset categories
- Clicking category filters book list

**5. BookList Page**

- Fetches list of books via BookService

- Renders BookCard components
- Supports search, sort, and category filters

**6. BookDetail Page**

- Displays full description
- Shows status: Available / Borrowed
- Button to request book

**7. AddBook Page**

- Form to create new book entry
- Fields: title, author, category, cover URL
- Sends POST request to backend

### 6.1.2 Backend Implementation

The backend follows a **controller–service–model architecture**.

**1. User Module**

- Registration
- Login
- Password encryption (bcrypt)
- JWT token generation

**2. Book Module**

APIs include:

- POST /books
- GET /books
- GET /books/:id
- PUT /books/:id
- DELETE /books/:id

**Book Fields**

- title
- author
- category
- owner_id
- cover_url
- status

**3. Borrow Request Module**

Handles full borrow workflow:

- Create request
- Approve or reject
- Update status
- Track borrow history

**4. Middleware – Authentication (JWT)**

- Protects private routes
- Ensures only logged-in users can borrow/add books

### 6.1.3 API Endpoints

**User APIs**

- POST /api/auth/register
- POST /api/auth/login

**Book APIs**

- GET /api/books
- GET /api/books/:id

- POST /api/books
- PUT /api/books/:id
- DELETE /api/books/:id

**Request APIs**

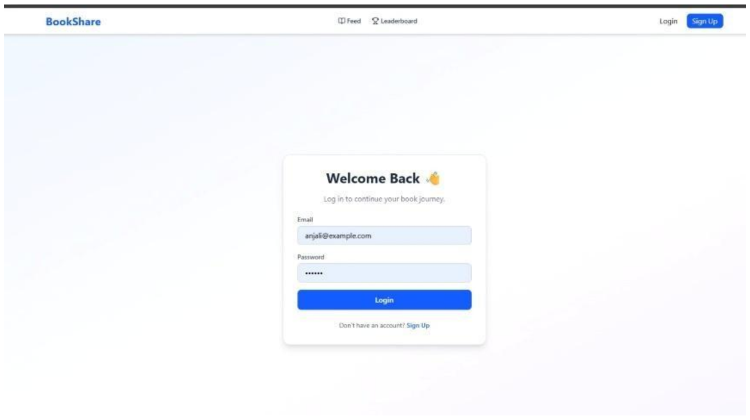
- POST /api/requests
- PUT /api/requests/:id/approve
- PUT /api/requests/:id/reject
- PUT /api/requests/:id/return

### 6.1.4 Testing

**Types of Testing Conducted**

- Unit testing (backend controllers)
- UI testing (button clicks, navigation)
- Functional testing (full borrow flow)
- API testing using Postman
- Integration testing (frontend + backend)

# CHAPTER 7
# RESULTS AND SCREENSHOTS


BookShare

Welcome Back 👋

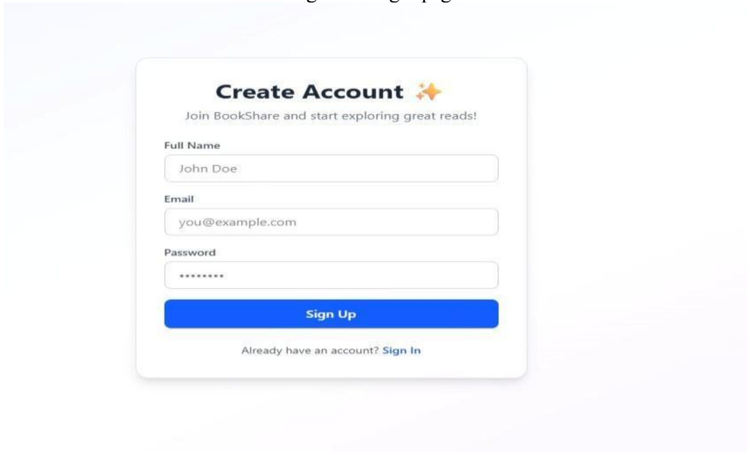
Log in to continue your book journey.

Email: anjali@example.com

Password: ••••••

Don't have an account? Sign Up

Fig 7.1 - Login page


Create Account ✨

Join BookShare and start exploring great reads!

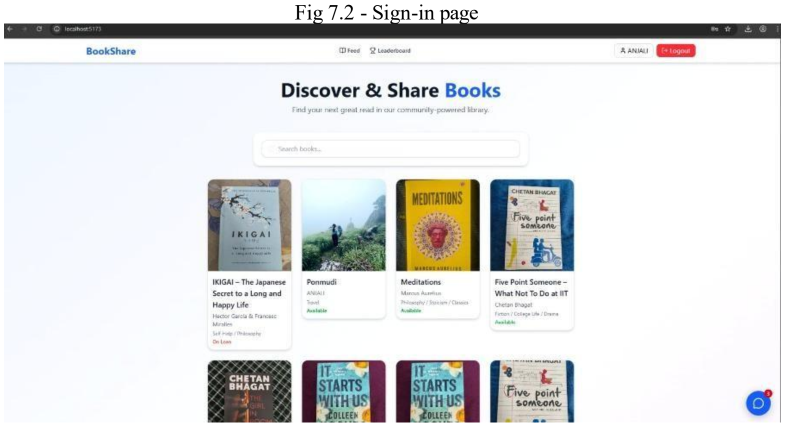
Full Name: John Doe

Email: you@example.com

Password: ••••••••

Already have an account? Sign In

Fig 7.2 - Sign-in page


localhost:5173

BookShare

Discover & Share Books

Find your next great read in our community-powered library.

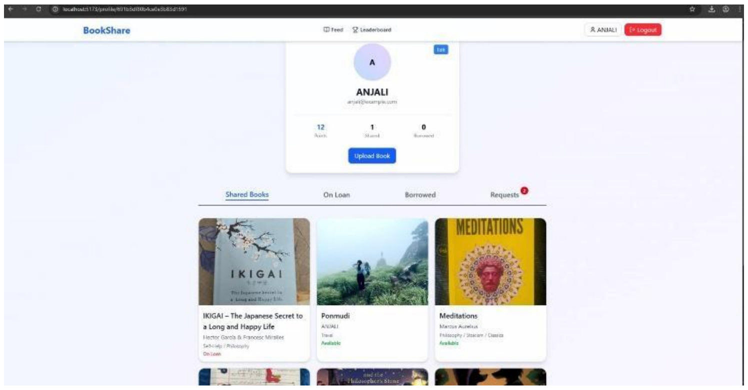
IKIGAI – The Japanese Secret to a Long and Happy Life
Hector Garcia & Francesc Miralles
Self Help / Philosophy
On Loan

Ponmudi
ANJALI
Travel
Available

Meditations
Marcus Aurelius
Philosophy / Stoicism / Classics
Available

Five Point Someone – What Not To Do at IIT
Chetan Bhagat
Fiction / College Life / Drama
Available

Fig 7.3- Home page



ANJALI

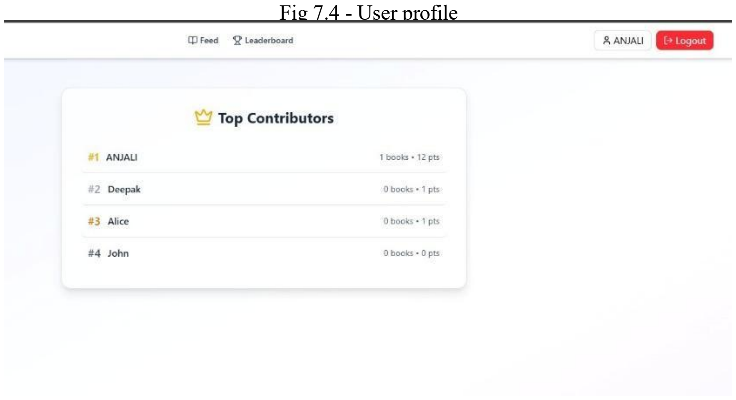
12 Points · 1 Shared · 0 Borrowed

IKIGAI – The Japanese Secret to a Long and Happy Life
Hector Garcia & Francesc Miralles
Self-Help / Philosophy
On Loan

Ponmudi
ANJALI
Travel
Available

Meditations
Marcus Aurelius
Philosophy / Stoicism / Classics
Available

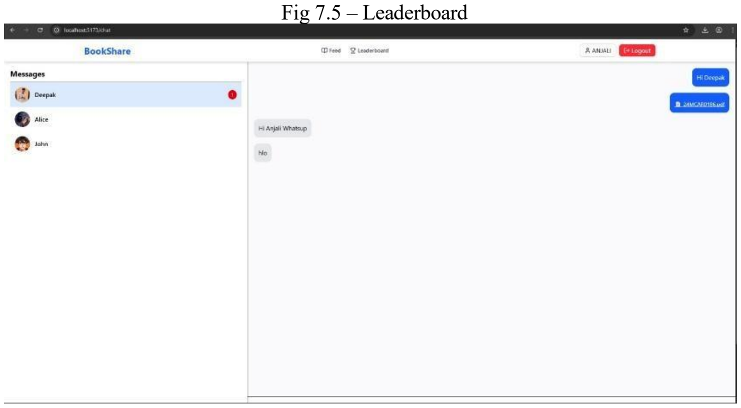
Fig 7.4 - User profile



**Top Contributors**

| | | |
|---|---|---|
| #1 | ANJALI | 1 books • 12 pts |
| #2 | Deepak | 0 books • 1 pts |
| #3 | Alice | 0 books • 1 pts |
| #4 | John | 0 books • 0 pts |

Fig 7.5 – Leaderboard



Messages

Deepak
Alice
John

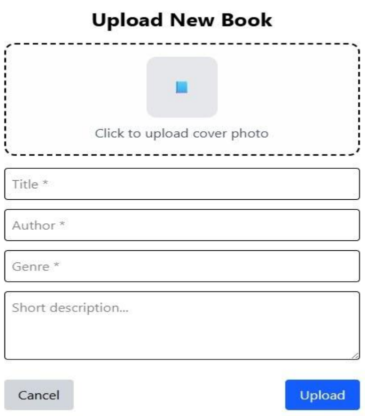
Hi Deepak

24MCAR0186.pdf

Hi Anjali Whatsup

Hlo

Fig 7.6 – Chats



**Upload New Book**

Click to upload cover photo

Title *

Author *

Genre *

Short description...

Fig 7.7 – Upload books

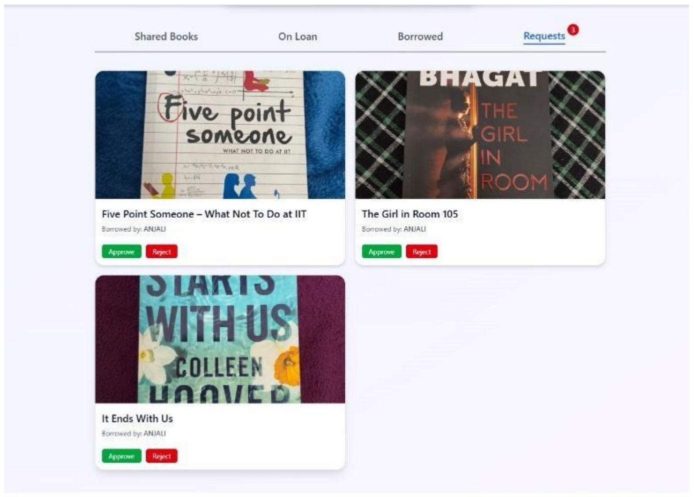


Fig 7.8 - Book request

## 7.1 Book Detail Page

| **Test Category** | **Description** | **Result** |
|---|---|---|
| Unit Testing | Module-level tests for controllers | Passed |
| API Testing | CRUD operations, authentication | Passed |
| UI Testing | Navigation, components, responsiveness | Passed |
| Integration Testing | Frontend ↔ Backend ↔ DB | Passed |
| Performance Testing | API load tests | Passed |

The system performed successfully across all functional and non-functional parameters.

# CHAPTER 8

# CONCLUSION AND FUTURE ENHANCEMENT

## 8.1 Conclusion

BookShare is developed as a modern digital solution addressing the limitations of traditional book-sharing methods. By integrating a full-stack architecture using React + TypeScript + Tailwind CSS for the frontend, Node.js + Express.js + TypeScript for the backend, and MongoDB Local Database for storage, the system ensures:

- Smooth navigation and clean UI design
- Scalable and modular backend services
- Secure user authentication
- Efficient book indexing and management
- Automated borrow request and approval workflows
- Real-time updates of book availability

The system successfully demonstrates how technology can enhance resource accessibility and community-driven learning. It offers a practical, user-friendly, and efficient platform for book exchange within educational environments and general communities.

BookShare fulfills its objectives by providing a streamlined, digital, and interactive alternative to manual book-sharing practices.

## 8.2 Future Enhancement

The system is fully functional but can be expanded in future versions with additional features:

**1. Recommendation System**

AI-based personalized book suggestions using user reading patterns.

**2. Review & Rating Module**

Allow users to post reviews, rate books, and provide comments.

**3. Push Notifications**

Notify users of approvals, due dates, new books, and recommendations.

**4. In-App Chat System**

Direct messaging between borrower and owner.

**5. Book Return Reminders**

Automated email/SMS reminders for overdue returns.

**6. Admin Panel**

A separate administrative dashboard for monitoring:

- System statistics
- User activity
- Book categories
- Abuse detection

**7. Integration with Cloud Storage**

Upload and store book cover images using Firebase Storage or AWS S3.

**8. Mobile App Version**

Develop native Android/iOS apps using React Native.

**9. Location-based Book Search**

Find nearest available books using geolocation.

**10. Gamification**

Points, badges, leaderboard for active contributors.

## REFERENCES

1) Clear, J. (2018). Atomic Habits. Avery Publishing.
2) Sharma, P. (2021). Digital Libraries and Community Knowledge Sharing. Journal of Information Technology.
3) Smith, A. (2022). Gamification Techniques in Educational Apps. International Review of Educational Technology.
4) React TypeScript Documentation – https://react.dev
5) Express.js Documentation – https://expressjs.com
6) Tailwind CSS Docs – https://tailwindcss.com
7) MDN Web Docs – https://developer.mozilla.org
8) Postman API Testing Guide – https://postman.com
9) Mongoose ODM Documentation – https://mongoosejs.com