VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590014

2023-2024

A Mini Project Report On
"Bookspree – A Digital Bookshelf"

Submitted in partial fulfillment of the requirement in 6th Semester
React with Mini Project work (BCSL657B)

Department of Computer Science & Engineering

By

SHARON A DOBBIN 1AT22CS120

Under the Guidance of
Prof. Harshitha S

Assistant Professor, Dept. of CSE, AIT

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERINGATRIA INSTITUTE OF TECHNOLOGY
BANGALORE- 560024
2024-2025

ATRIA INSTITUTE OF TECHNOLOGY DEPARTMENT OF COMPUTE SCIENCE & ENGINEERING BANGALORE-560024

CERTIFICATE

Certified that the project work entitled "**Bookspree – A Digital Bookshelf**", carried out by **SHARON A DOBBIN (1AT22CS120)** is a bonafide student of Atria Institute of Technology, Bangalore, in partial fulfilment for the award of Bachelor of Engineering in Computer Science & Engineering of Visvesvaraya Technological University, Belgaum during the academic year 2024-2025. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies requirement in respect of project work prescribed for the said degree.

Signature of Principal **Signature of Guide** **Signature of HOD**

DECLARATION

I, SHARON A DOBBIN (1AT22CS120), student of VI semester B.E in Computer Science & Engineering at Atria Institute of Technology, hereby declare that the project work entitled "Bookspree – A Digital Bookshelf" has been carried out under the supervision of Prof. Harshitha S, Assistant Professor, Dept. of CS&E, Atria Institute of Technology and submitted in partial fulfilment of the course requirements for the award of degree in B.E in Computer Science & Engineering of Visvesvaraya Technological University, Belagavi during the year 2024-2025. We further declare that the report has not been submitted to any other University for the award of any other degree.

Place: Bangalore **Signature of the student**

Date: 21 MAY, 2025

NAME: SHARON A DOBBIN
(USN: 1AT22CS120)

ABSTRACT

This project, *Bookspree*, is a web-based book tracking and organization system inspired by platforms like *Goodreads* and *StoryGraph*. It allows users to explore, categorize, and manage books across various genres such as *Fantasy*, *Horror*, *Mystery & Crime*, *Non-Fiction & Biographies*, and *Children's Literature*. Users can view curated book listings, add new books to their shelves, and edit or delete existing entries with ease.

The organisation system has been optimised by use of Bookshelves such as Read, Currently Reading, Plan to Read, and Did not Finish.

The primary goal of Bookspree is to simplify the process of discovering and organizing books based on user interests and reading status. By providing genre-specific pages and interactive book cards, it enhances user engagement and encourages reading habits. Furthermore, an external link is embedded into the book cards, to allow users to visit an external website (in this case, Goodreads), and view information such as Book Reviews and Ratings, Author Information, more books like this, etc.

The application has been implemented using React for the frontend, with a little bit of Tailwind CSS for styling and responsiveness, as well as embedded CSS for interactive design. Book information is dynamically rendered using components, and user-added data is stored persistently in the browser using localStorage, allowing functionality without a backend. The modular component design ensures scalability and maintainability of the codebase.

Overall, Bookspree serves as a lightweight, user-friendly platform for book lovers to track and personalize their reading experience.

ACKNOWLEDGEMENT

I express gratitude to our institution and management for providing me with good infrastructure, laboratory, facilities and inspiring staff, and whose gratitude was of immense help in completion of this project successfully.

I express my sincere gratitude to **Dr. Rajesha S**, Principal, Atria Institute of Technology, for providing me with the required environment and for his valuable suggestion.

My sincere thanks to **Dr. Devi Kannan**, Head of the Dept. Computer Science and Engineering, Atria Institute of Technology, for her valuable support and for rendering me resources for this project work.

I express my gratitude to **Prof. Harshitha S**, Associate Professor, Dept. of Computer Science and Engineering, Atria Institute of Technology, who guided me with valuable suggestions in completing this project at every stage.

Last but not the least, the project would not have been a success without the support of my **parents** and **friends**. My sincere thanks should be rendered to everyone who helped me in all possible ways.

NAME: SHARON A DOBBIN

USN: 1AT22CS120

TABLE OF CONTENTS

CHAPTER – 1

INTRODUCTION

Reading has always been a powerful means of gaining knowledge, entertainment, and personal growth. In the digital age, managing and exploring books has become more interactive and convenient through online platforms. Inspired by the concept of *Goodreads*, this project—**Bookspree**—aims to offer a simplified and intuitive book exploration and management system that caters to a wide audience, especially casual readers and students.

Bookspree is a genre-based book tracking and organization application designed to help users discover, catalog, and track books based on their reading preferences. The platform features dedicated pages for popular genres such as *Fantasy*, *Horror*, *Mystery & Crime*, *Non-Fiction & Biographies*, and *Children's Literature*. Users can view curated book collections as well as contribute by adding new titles, editing existing ones, or removing books they no longer wish to keep on their virtual shelf.

The project is built using **React.js** for a responsive and dynamic frontend interface, styled with **Tailwind CSS** for a clean, modern appearance as well as embedded CSS for a more attractive design. It does not rely on a complex backend; instead, it uses **localStorage** for persisting user-added data, ensuring a lightweight and efficient experience, and uses a JSON file for storing the login credentials created during the registration of an account in the Register Page. The modular component structure, including reusable elements like BookCard, promotes maintainability and scalability.

This project serves as a foundation for building more advanced book-related applications and demonstrates key frontend development skills such as component-based design, state management, conditional rendering, and user interaction handling.

CHAPTER 2

PROBLEM STATEMENT

Existing book platforms like Goodreads can be overwhelming for casual users due to complex features and account requirements. There is a need for a simple, user-friendly system that allows readers to explore, add, edit, and manage books by genre without requiring signups or internet access. This project addresses that need by providing a lightweight, browser-based application with local data storage and an intuitive interface.

PROBLEM OVERVIEW:

With the increasing popularity of digital reading platforms, there is a growing need for simple, user-friendly systems that allow users to explore and organize their reading interests without the complexity of full-scale applications like Goodreads. Many such platforms require user registration, constant internet access, and expose users to cluttered interfaces filled with social and recommendation features that might overwhelm readers who prefer a more minimal experience.

This project addresses the gap by providing a lightweight, genre-based book management web application that allows users to browse pre-listed books and manage their own entries easily. Users can add, edit, or delete books under specific genres like Romance & Fantasy, Children's Literature, Horror, Mystery & Crime, and more. The system relies solely on client-side technologies, using React for the frontend interface and browser localStorage for persistent data handling—eliminating the need for complex backend systems or databases.

The goal is to offer a straightforward interface that caters to readers, students, and book lovers who want a quick and customizable way to maintain their virtual bookshelf organized by genre. By focusing on usability and local data management, the application ensures accessibility, ease of use, and a distraction-free experience tailored for personal book tracking.

CHAPTER 3
OBJECTIVE

The primary objective of this project is to design and implement a user-friendly, web-based book management system that allows users to explore, categorize, and manage books across various genres.

1. To Create a Genre-Based Book Browsing Experience

The application is designed to categorize books into distinct genres such as Romance & Fantasy, Horror, and more. This classification allows users to explore books based on their interests and encourages focused reading habits by presenting relevant titles under each genre.

2. To Enable Book Management without Backend Dependencies

A key objective is to develop a fully functional system without relying on external servers or databases. Instead, the project utilizes localStorage to persist user data directly in the browser. This ensures offline accessibility and removes complexity from the implementation, making it lightweight and ideal for personal or educational use.

3. To Allow Book Addition, Editing, and Deletion

Users should be able to not only view pre-listed books but also add their own titles with relevant information such as book name, author, cover image, and an optional Goodreads link. They can edit or delete these entries as needed. This gives users full control over their reading collection in a flexible and intuitive way.

4. To Build a Responsive and Visually Appealing Interface

The project aims to offer a clean, responsive UI using Tailwind CSS and React. The interface should be easy to navigate, visually consistent across devices, and engaging for users. Special attention is given to color schemes, typography, and layout for each genre page to enhance user engagement and readability.

5. To Promote Independent Reading and Organization

By offering a platform where users can manage their books by genre, the project encourages independent reading habits. It helps users keep track of books they're interested in, have read, or want to explore—thereby promoting better reading organization and motivation.

6. To Ensure Simplicity and Accessibility

Another goal is to maintain simplicity in both the codebase and the user interface. The system is built using widely understood technologies like React and JavaScript, making it easy to maintain and extend. The user interface is minimal and free of distractions, making it accessible even to users who are not tech-savvy.

CHAPTER 4

REQUIREMENT ANALYSIS

1. Functional Requirements

These are the core capabilities that the system must support to fulfil user needs:

- **User Authentication:**
  - Users can sign up with a username and password.
  - Login is validated against stored credentials in a JSON file.
  - Only authenticated users can access personalized dashboards and features.
- **Book Management:**
  - Users can add books by specifying the title, author, cover image URL, and external Goodreads link.
  - Books can be categorized into shelves: "Read," "Currently Reading," "Want to Read," and "Did Not Finish."
  - Users can edit or delete books they added.
- **Genre Navigation:**
  - Users can browse books by different genres (e.g., Fantasy, Romance & Fantasy, Children's Literature, etc.).
  - Each genre page displays a grid of relevant book cards.
- **Persistent Storage:**
  - Books added by users are stored using the browser's local storage.
  - Login credentials are stored in a JSON file managed by the backend.

2. Non-Functional Requirements

- **Usability:**
  - The system uses a simple, intuitive UI styled with Tailwind CSS for ease of use and responsiveness.
  - Forms for book input and user actions are user-friendly and easy to understand.
- **Performance:**
  - Pages should load quickly with minimal latency due to the lightweight frontend and use of local storage.
- **Maintainability:**
  - Code is modular and follows best practices, making it easy to modify and extend.
- **Security:**
  - While the project uses a basic JSON-based backend for learning purposes, data handling assumes secure access (e.g., no plain password exposure in frontend).
- **Portability:**
  - As the frontend is built using React and the backend is a simple Node.js server, it can run on any platform supporting these technologies.

3. Technical Requirements

- **Frontend:**
  - React.js with Tailwind CSS for styling.
  - React Router for navigation between pages.
- **Backend:**
  - Node.js and Express server.
  - JSON file used to store user credentials for signup and login functionality.
- **Development Tools:**
  - VSCode as the primary IDE.
  - Local development environment with Node.js installed.
- **Data Storage:**
  - LocalStorage for book shelf data.
  - JSON file for user account management.

4. Constraints

- **No Database Integration:**
  - The project does not use a traditional database like MySQL or MongoDB; instead, it relies on a JSON file and browser storage.
- **Single User Environment (Prototype Level):**
  - The system is not designed for concurrent multi-user support at scale, as JSON file handling lacks concurrency control.
- **Limited Security Implementation:**
  - As a prototype, password encryption or advanced security measures like token-based authentication are not included.
- **Local Deployment:**
  - The current setup is designed for local use and does not include cloud hosting or deployment pipelines.

CHAPTER 5
DESIGN

Dashboard Component Design

The **Dashboard** is the central user interface that appears after successful login. It is designed to offer a personalized space for users to manage their bookshelf and access key functionalities of the Bookspree application.

Purpose and Functionality

The Dashboard serves as the main hub where users can:

- View their categorized book shelves: *Read*, *Currently Reading*, *Want to Read*, and *Did Not Finish*.
- Add new books to these shelves using a structured form.
- Instantly see the books displayed under each corresponding shelf based on their chosen category.

Component Structure

The Dashboard consists of three main parts:

1. **Welcome Message** – Displays a friendly greeting at the top of the page to enhance user experience.
2. **Book Entry Form** – Allows users to enter book details such as title, author, Goodreads link, cover image URL, and select a shelf.
3. **Book Display Section** – Dynamically renders the books grouped by their shelf category using simple filtering and mapping of book data.

Data Handling

- All books entered by the user are stored in the browser's localStorage, ensuring data persists between sessions.
- When a book is submitted, it is saved and immediately reflected on the UI under the relevant shelf.
- The component re-renders automatically as the list of books updates, ensuring a real-time interactive experience.

User Interaction

- The user can repeatedly add books using the form, which validates required fields.
- Clicking the Goodreads link on any displayed book redirects users to its Goodreads page in a new tab.

This modular and user-centric approach provides a seamless reading log experience, reflecting the core goal of Bookspree — to organize and explore personal reading journeys efficiently.

#Code Implementation of the Dashboard

```
import React, { useState, useEffect } from 'react';
import { Link } from 'react-router-dom';

const Dashboard = () => {
  const [books, setBooks] = useState(() => {
    const stored = localStorage.getItem('books');
    return stored ? JSON.parse(stored) : [];
  });

  const [showForm, setShowForm] = useState(false);
  const [editBookId, setEditBookId] = useState(null);

  useEffect(() => {
    localStorage.setItem('books', JSON.stringify(books));
  }, [books]);

  const handleSubmit = (e) => {
    e.preventDefault();
    const formData = new FormData(e.target);
    const newBook = {
      id: editBookId || Date.now(),
      title: formData.get('title'),
      author: formData.get('author'),
      link: formData.get('link'),
      cover: formData.get('cover') ||
'https://via.placeholder.com/150x220?text=No+Cover',
      shelf: formData.get('shelf'),
    };
```

```
    if (editBookId) {
      setBooks((prev) =>
        prev.map((book) => (book.id === editBookId ? newBook : book))
      );
    } else {
      setBooks((prev) => [newBook, ...prev]);
    }

    e.target.reset();
    setShowForm(false);
    setEditBookId(null);
  };

  const startEdit = (book) => {
    setEditBookId(book.id);
    setShowForm(true);
    setTimeout(() => {
      document.getElementById('title').value = book.title;
      document.getElementById('author').value = book.author;
      document.getElementById('link').value = book.link;
      document.getElementById('cover').value = book.cover;
      document.querySelector(`input[name="shelf"][value="${book.shelf}"]`).che
cked = true;
    }, 0);
  };

  const deleteBook = (id) => {
    if (window.confirm('Delete this book?')) {
      setBooks((prev) => prev.filter((book) => book.id !== id));
    }
  };

  const shelves = ['read', 'currently reading', 'want to read', 'did not finish'];

  return (
```

```
   <div className="max-w-5xl mx-auto px-4 py-8">
     <h1 className="text-3xl font-bold text-center text-red-800 mb-6">Book
Dashboard</h1>

     <div className="text-center mb-6">
       <button
         onClick={() => {
           setShowForm(!showForm);
           setEditBookId(null);
         }}
         className="bg-red-800 text-white px-6 py-2 rounded-full font-semibold"
       >
         {showForm ? 'Close Form' : editBookId ? 'Edit Book' : 'Add New Book'}
       </button>
     </div>

     <div className="text-center mb-6">
       <Link
         to="/genre-search"
         className="bg-red-800 text-white px-6 py-2 rounded-full font-semibold
inline-block"
       >
         Search by Genre
       </Link>
     </div>

     {showForm && (
       <form
         onSubmit={handleSubmit}
         className="bg-white shadow-md rounded px-8 pt-6 pb-8 mb-8 space-y-
4"
       >
         <input id="title" name="title" placeholder="Book Title" required
className="w-full border p-2 rounded" />
         <input id="author" name="author" placeholder="Author" required
className="w-full border p-2 rounded" />
```

```
        <input
          id="link"
          name="link"
          type="url"
          placeholder="Goodreads Link"
          required
          className="w-full border p-2 rounded"
        />
        <input
          id="cover"
          name="cover"
          type="url"
          placeholder="Cover Image URL (optional)"
          className="w-full border p-2 rounded"
        />
        <div className="space-x-4">
          {shelves.map((shelf) => (
            <label key={shelf} className="capitalize">
              <input type="radio" name="shelf" value={shelf} required
```
```
className="mr-1" />
              {shelf}
            </label>
          ))}
        </div>
        <button type="submit" className="bg-red-800 text-white px-6 py-2
```
```
rounded-full font-semibold">
          {editBookId ? 'Update Book' : 'Add Book'}
        </button>
      </form>
    )}

    {shelves.map((shelf) => (
      <div key={shelf} className="mb-12">
        <h2 className="text-xl font-semibold text-red-800 capitalize border-b-2
```
```
border-red-800 mb-4">
```

```
          {shelf}
        </h2>
        <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 gap-6">
          {books
            .filter((book) => book.shelf === shelf)
            .map((book) => (
              <div key={book.id} className="bg-white p-4 shadow rounded
relative flex flex-col items-center">
                <a
                  href={book.link}
                  target="_blank"
                  rel="noopener noreferrer"
                  className="text-center hover:opacity-90"
                >
                  <img
                    src={book.cover}
                    alt={book.title}
                    className="h-48 w-auto object-cover mb-2 rounded"
                  />
                  <h3 className="text-red-800 font-bold">{book.title}</h3>
                  <p className="text-gray-600">{book.author}</p>
                </a>
                <div className="mt-2 flex space-x-2">
                  <button
                    onClick={() => startEdit(book)}
                    className="bg-yellow-400 text-black px-3 py-1 rounded text-sm"
                  >
                    Edit
                  </button>
                  <button
                    onClick={() => deleteBook(book.id)}
                    className="bg-red-600 text-white px-3 py-1 rounded text-sm"
                  >
                    Delete
                  </button>
```

```
              </div>
            </div>
          ))}
        </div>
      </div>
    ))}
  </div>
 );
};

export default Dashboard;
```

CHAPTER 6

IMPLEMENTATION

The implementation phase of the **Bookspree** project involved translating the planned features and UI components into functional, interactive code using modern web development tools. The primary goal was to build a lightweight, responsive book review system that enables users to log in, manage their personal bookshelf, and explore genre-based book recommendations.

Frontend Implementation

The frontend of Bookspree is developed using **React.js**, a popular JavaScript library for building user interfaces. Tailwind CSS is used for styling, allowing for rapid and consistent UI development. React's component-based architecture enabled clean separation of concerns, with reusable components such as BookCard, Dashboard, Login, Signup, and individual genre pages.
Key frontend features include:

- **User Authentication Forms**: Login and Signup pages were built using basic form components with validation. Upon account creation or successful login, users are redirected to the Dashboard.
- **Dynamic Book Entry and Display**: Users can add books with details such as title, author, cover image, and a Goodreads link. These books are displayed under categorized shelves like *Read*, *Currently Reading*, etc., using filters.
- **Genre Pages**: Static genre-based pages showcase curated books for each category, enhancing discovery and user engagement.

Backend & Data Storage

To keep the backend simple yet functional, the project uses **Node.js with Express** and **JSON file storage** as a lightweight alternative to a full database. This backend handles:

- **User Authentication**: During signup, user credentials are saved in a JSON file. During login, these credentials are validated against the stored data.
- **Data Persistence**: Books added from the Dashboard are stored in the browser's localStorage, allowing data to persist across sessions without requiring a full-fledged database.

This hybrid approach allows the project to function without heavy server infrastructure, making it ideal for learning environments or small-scale personal use.

Integration
Frontend and backend components are loosely coupled. Communication between them (e.g., during login/signup) is handled via HTTP requests using the fetch() API. LocalStorage is used on the client side to store and retrieve book data efficiently.

Testing and Debugging
Each component was tested manually to ensure smooth interaction. Console logs, conditional rendering, and localStorage behavior were monitored during development to identify and fix bugs quickly.

CHAPTER 7
RESULT

The development and implementation of the **Bookspree** book review system successfully fulfilled its intended objectives. The project resulted in a fully functional web application that enables users to:

- **Create an Account and Log In** using a secure system backed by a lightweight Node.js server with JSON-based data storage.
- **Access a Personalized Dashboard** upon login, where they can add books to different shelves such as *Read*, *Currently Reading*, *Want to Read*, and *Did Not Finish*.
- **Add Book Details Dynamically**, including title, author, cover image, and Goodreads link, with persistent local storage to retain user-added books across sessions.
- **Edit and Delete Book Entries**, giving users full control over their collection.
- **Explore Genre-Based Book Lists** with preloaded recommendations under genres such as Romance & Fantasy, Children's Literature, Horror, Mystery & Crime, Non-Fiction Biographies, and Fantasy.

The user interface is clean, responsive, and built with Tailwind CSS, ensuring usability across different devices. The use of React.js for the frontend allows for fast updates and smooth user interactions.

By combining a client-side experience with a simple backend for user authentication, the project delivers a balance of interactivity and functionality. The system demonstrates how modern web technologies can be used to build a real-world application with minimal resources.

CHAPTER 8

CONCLUSION

The development of **Bookspree: A Book Review and Management System** has successfully demonstrated the creation of a dynamic, user-friendly web application that enables users to organize, explore, and manage their personal reading journey. The project achieved its core objectives, including account creation and login functionality, genre-based book browsing, and shelf management with full CRUD (Create, Read, Update, Delete) support.

The use of **React.js** for the frontend provided a highly responsive and interactive user experience, while the lightweight **Node.js + JSON file backend** offered simple yet effective data storage and user authentication. LocalStorage was also utilized to persist book data added by users across sessions.

Throughout the project, careful attention was paid to usability, visual consistency, and modular design. The integration of components like BookCard, reusable form logic, and clean routing ensured that the codebase remained maintainable and scalable.

Overall, this project showcases how modern web technologies can be effectively utilized to build a complete and meaningful application. It also serves as a strong foundation for future enhancements, such as integrating a real-time database, adding user-specific book tracking, or implementing recommendations powered by AI.