Deccan Education Society's
Fergusson College (Autonomous), Pune
Department of Computer Science

A

Project Report
on

"BookAura"

In partial fulfillment of Post Graduate course

in

M.Sc. Computer Science – I

(Semester -II)

CSC-560 Project

SUBMITTED BY

Dhimar Arin Avinash (246203)
Dighe Atharva Ajey (246205)
Pare Prajakta Prabhakar(246249)

Deccan Education Society's
Fergusson College (Autonomous), Pune
Department of Computer Science

CERTIFICATE

This is to certify that the project entitled BookAura submitted by

1. Dhimar Arin Avinash
2. Pare Prajakta Prabhakar
3. Dighe Atharva Ajey

in partial fulfillment of the requirement of the completion of M.Sc. (C.S)-I [Semester-II], has been carried out by them under our guidance satisfactorily during the academic year 2024-2025.

Place: Pune
Date: / /2025

Dr. Kavita Khobragade
Head,
Department of Computer Science
Fergusson College (Autonomous), Pune

Project Guide:

1.

Examiners Name **Sign**

1. _____ _____

2. _____ _____

Index

1. Introduction

In the current digital landscape, the demand for a scalable, efficient, and feature-rich book management system is paramount. This web application is designed to address key technical limitations in existing platforms, offering robust functionalities for users, authors, publishers, and administrators.

1.1 Existing System

The existing systems exhibit several technical drawbacks:

- **Restricted Accessibility**: Content access is often subscription-based with limited free-tier options.
- **Lack of Data Analytics**: Authors and publishers lack real-time insights into book engagement metrics such as views, bookmarks, and feedback.
- **Limited NLP-Based Audiobook Support**: Existing platforms primarily support audiobooks in limited languages, without real-time text-to-speech integration.
- **Inefficient Moderation Pipelines**: Manual content approval processes without structured workflow automation.
- **Rigid Publishing Mechanism**: Minimal control for authors and publishers regarding private and public book management.

1.2 Need for the System

To overcome these challenges, our system incorporates:

- **Full-Stack User Engagement**: Users can read, listen, bookmark, and provide structured feedback using advanced UI/UX components.
- **Comprehensive Author & Publisher Management**: CRUD operations for book publishing, detailed analytics, and structured publishing workflows.
- **Multi-Language Audiobook Generation**: Real-time text-to-speech conversion using Python libraries and APIs supporting English, Hindi, and Marathi.
- **Automated Moderation System**: Role-based content approval pipeline with structured feedback mechanisms.
- **Admin-Driven Governance**: Centralized role-based access control (RBAC) ensuring structured management of platform entities.

1.3 Overview of the Project

This project is a full-stack web application integrating frontend, backend, database management, and AI-driven functionalities. The system follows a microservices-based architecture, ensuring modularity and scalability. Key modules include:

- **User Module**: Interactive book browsing, reading, audio processing, and bookmark management.
- **Publisher Module**: Requires administrative approval before content submission, ensuring compliance with platform policies.
- **Moderator Module**: Content review workflows, approval mechanisms, and structured moderation of user-generated content.
- **Administrator Module**: High-level control over moderation, agreements, and content governance.

The system leverages technologies such as React.js for the frontend, Flask for backend services, MySQL for database management, and Python-based text-to-speech APIs for audiobook generation, ensuring a robust, scalable, and feature-rich platform.

2. Analysis

2.1 Feasibility Study

2.1.1 Technical Feasibility

The system is developed using a microservices architecture with modular components to ensure scalability and maintainability. The frontend is built using React.js for a dynamic UI, while the backend utilizes Flask for efficient API handling. Database management involves MySQL and Firebase to store structured and unstructured data. Audiobook generation relies on Python-based text-to-speech (TTS) libraries, ensuring multi-language support.

2.1.2 Economical Feasibility

The system minimizes operational costs by leveraging open-source technologies. Hosting is optimized through cloud-based services, ensuring scalability with minimal infrastructure costs. The use of API-driven functionalities eliminates the need for extensive hardware investments, making the solution financially viable.

2.1.3 Operational Feasibility

The platform is designed with an intuitive UI/UX to enhance user engagement. Role-based access control (RBAC) ensures structured content management, reducing manual intervention. Moderators and administrators have automated workflows for content approval, ensuring streamlined operations. The system also integrates analytics dashboards for publishers to track book performance, improving operational efficiency.

2.2 Hardware and Software Requirements

Hardware Requirements:

- **Server**: Cloud-based or dedicated server with at least 8GB RAM and multi-core CPU
- **Database Storage**: Scalable cloud-based storage (e.g.,MySQL)
- **Client Devices**: Web browsers supporting React-based applications (Chrome, Firefox, Edge, Safari)

Software Requirements:

- **Frontend**: React.js, Tailwind CSS,ShadCn

- **Backend**: Flask, Python
- **Database**: MySQL
- **Audio Processing**: Text-to-speech APIs (Google TTS, PyTTSX3, gTTS , Bhashini)
- **Version Control**: Git, GitHub

3. Design

3.1 Database Table Design

Roles Table

Column Name	Data Type	Constraints
role_id	INT	PRIMARY KEY, AUTO_INCREMENT
role_name	VARCHAR(50)	NOT NULL, UNIQUE

Categories Table

Column Name	**Data Type**	**Constraints**
category_id	INT	PRIMARY KEY, AUTO_INCREMENT
category_name	VARCHAR(255)	NOT NULL

Users Table

Column Name	Data Type	Constraints
user_id	INT	PRIMARY KEY, AUTO_INCREMENT
username	VARCHAR(50)	NOT NULL, UNIQUE
email	VARCHAR(100)	NOT NULL, UNIQUE
password_hash	VARCHAR(255)	NOT NULL
role_id	INT	NOT NULL, FOREIGN KEY REFERENCES roles(role_id)
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP
updated_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP

Books Table

Column Name	Data Type	Constraints
book_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	NOT NULL, FOREIGN KEY REFERENCES users(user_id)
title	VARCHAR(255)	NOT NULL
description	TEXT	
coverUrl	TEXT	NOT NULL
fileUrl	TEXT	NOT NULL
audioUrl	TEXT	NOT NULL
is_public	BOOLEAN	DEFAULT FALSE
is_approved	BOOLEAN	DEFAULT FALSE
uploaded_by_role	ENUM('Author', 'Publisher')	NOT NULL
uploaded_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

Platform Administrators Table

Column Name	**Data Type**	**Constraints**
admin_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	NOT NULL, FOREIGN KEY REFERENCES users(user_id)

Authors/Publishers Table

Column Name	**Data Type**	**Constraints**
author_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	NOT NULL, FOREIGN KEY REFERENCES users(user_id)
bio	TEXT	
is_approved	BOOLEAN	DEFAULT FALSE

Normal Users Table

Column Name	Data Type	Constraints
normal_user_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	NOT NULL, FOREIGN KEY REFERENCES users(user_id)
additional_info	TEXT	
is_flagged	BOOLEAN	DEFAULT FALSE

Book Category Table

Column Name	Data Type	Constraints
book_category_id	INT	PRIMARY KEY, AUTO_INCREMENT
category_id	INT	FOREIGN KEY REFERENCES categories(category_id)
book_id	INT	FOREIGN KEY REFERENCES books(book_id)

Views Table

Column Name	Data Type	Constraints
book_view_id	INT	PRIMARY KEY, AUTO_INCREMENT
book_id	INT	NOT NULL, FOREIGN KEY REFERENCES books(book_id)
book_view	INT	NOT NULL

Bookmarks Table

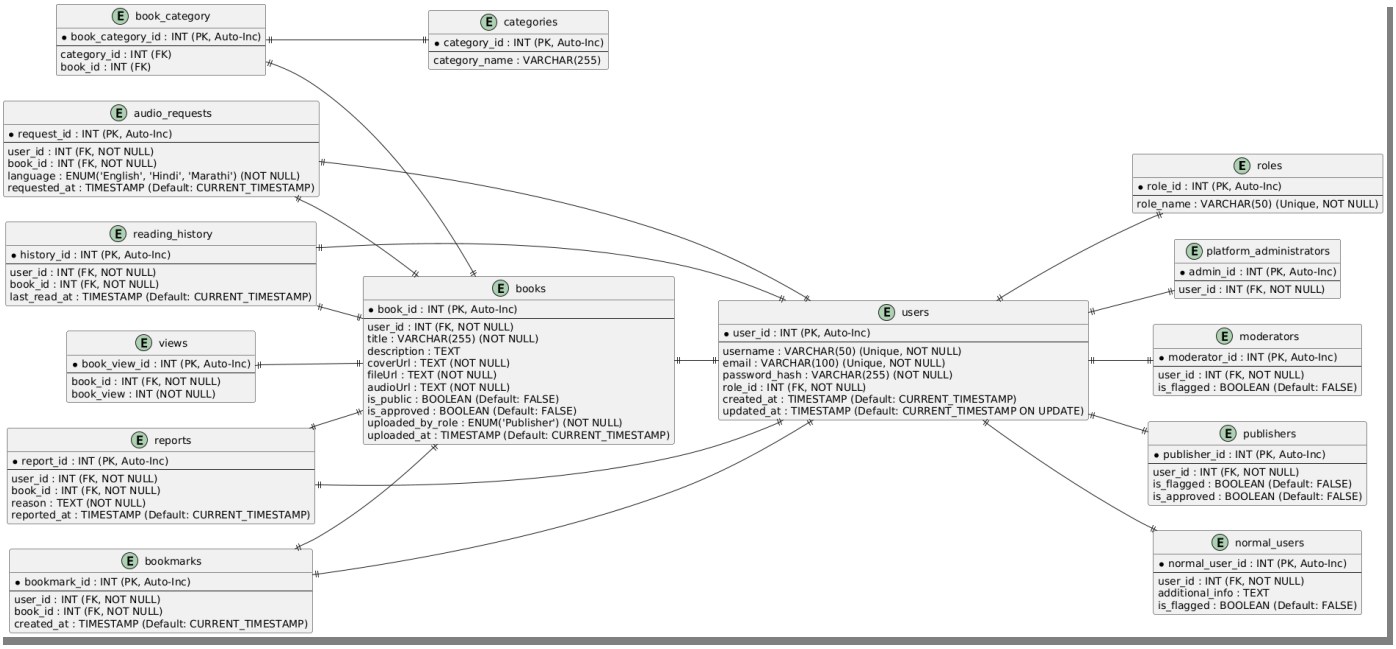
Column Name	Data Type	Constraints
bookmark_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	NOT NULL, FOREIGN KEY REFERENCES users(user_id)
book_id	INT	NOT NULL, FOREIGN KEY REFERENCES books(book_id)
created_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

Reading History Table

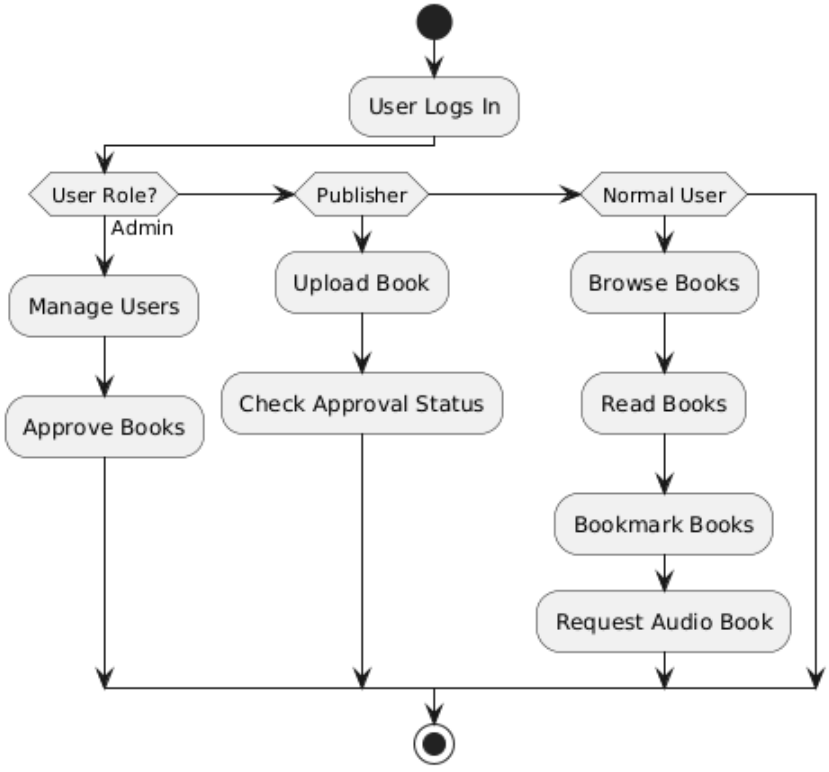
Column Name	Data Type	Constraints
history_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	NOT NULL, FOREIGN KEY REFERENCES users(user_id)
book_id	INT	NOT NULL, FOREIGN KEY REFERENCES books(book_id)
last_read_at	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP

3.2 Software Engineering Diagram

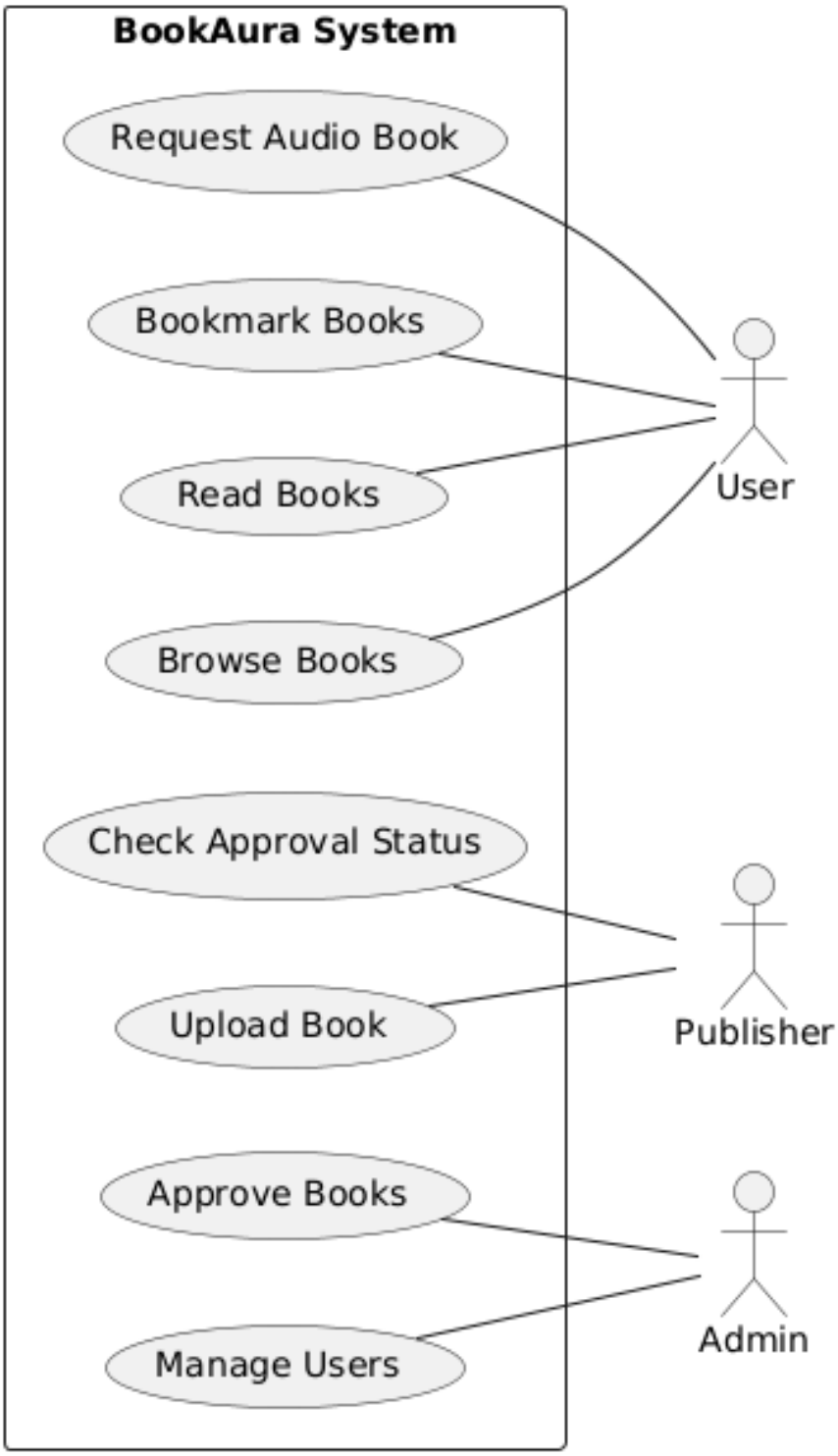
1. ER Diagram



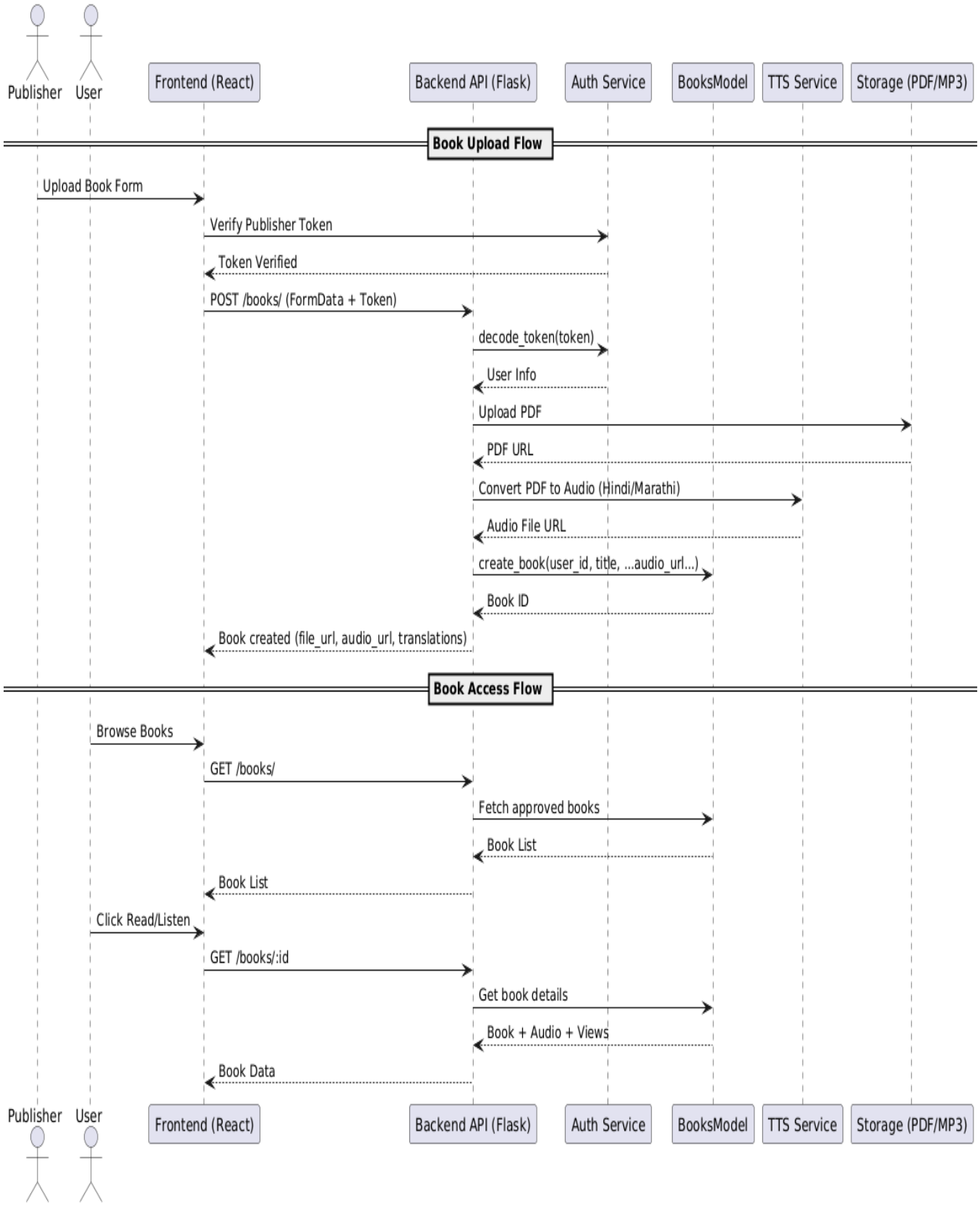
2. Activity Diagram



2. Use Case Diagram

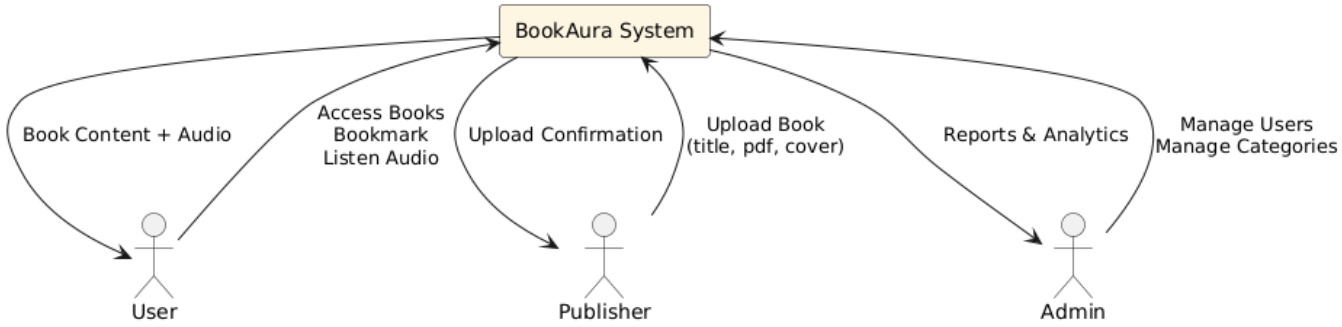


3.Sequence Diagram

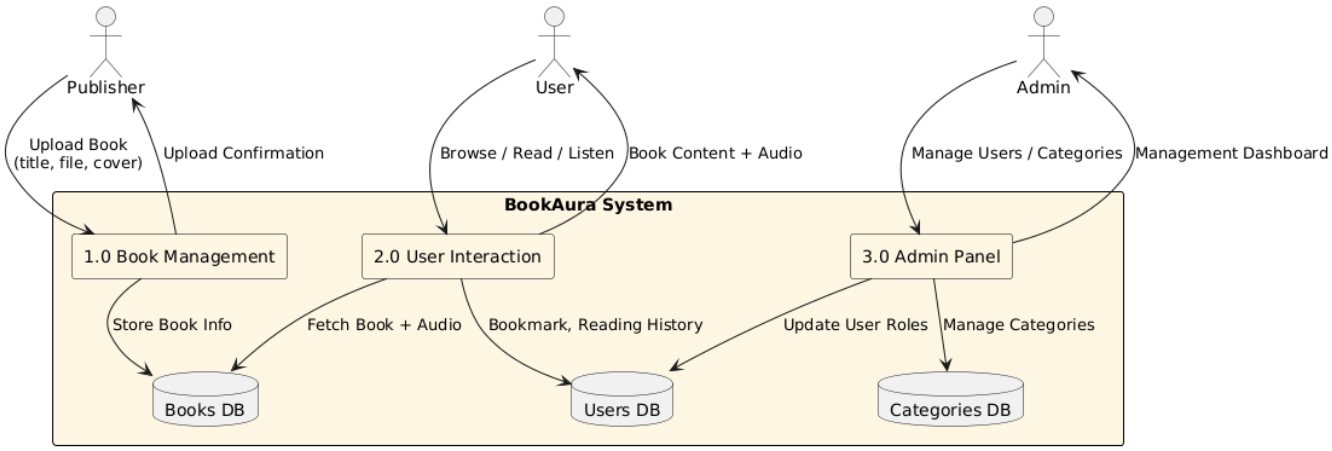


4. DFD

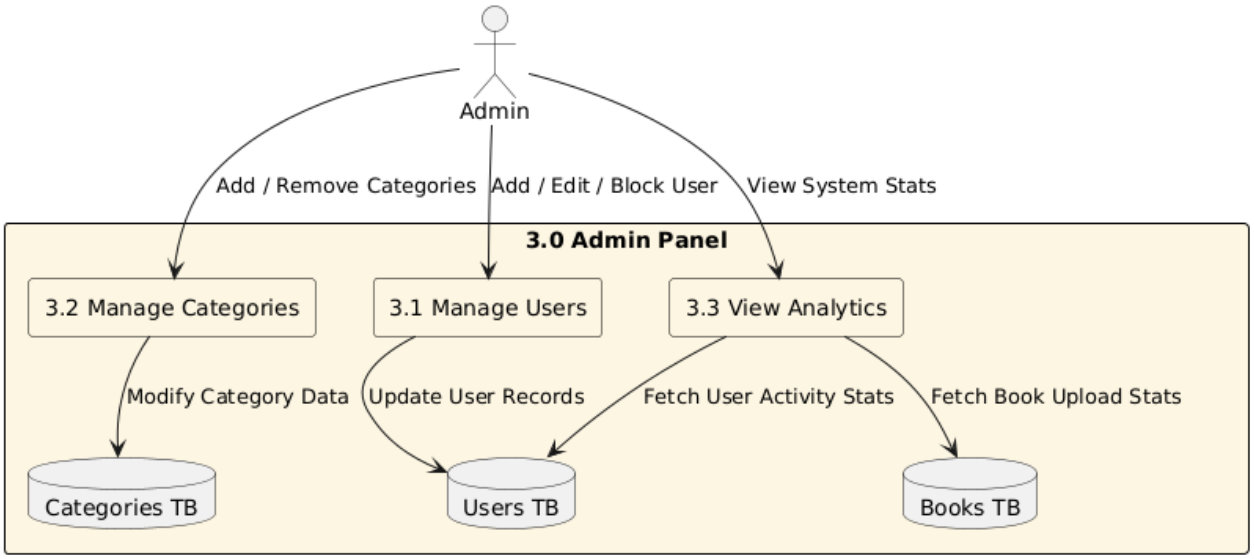
Level 0 DFD



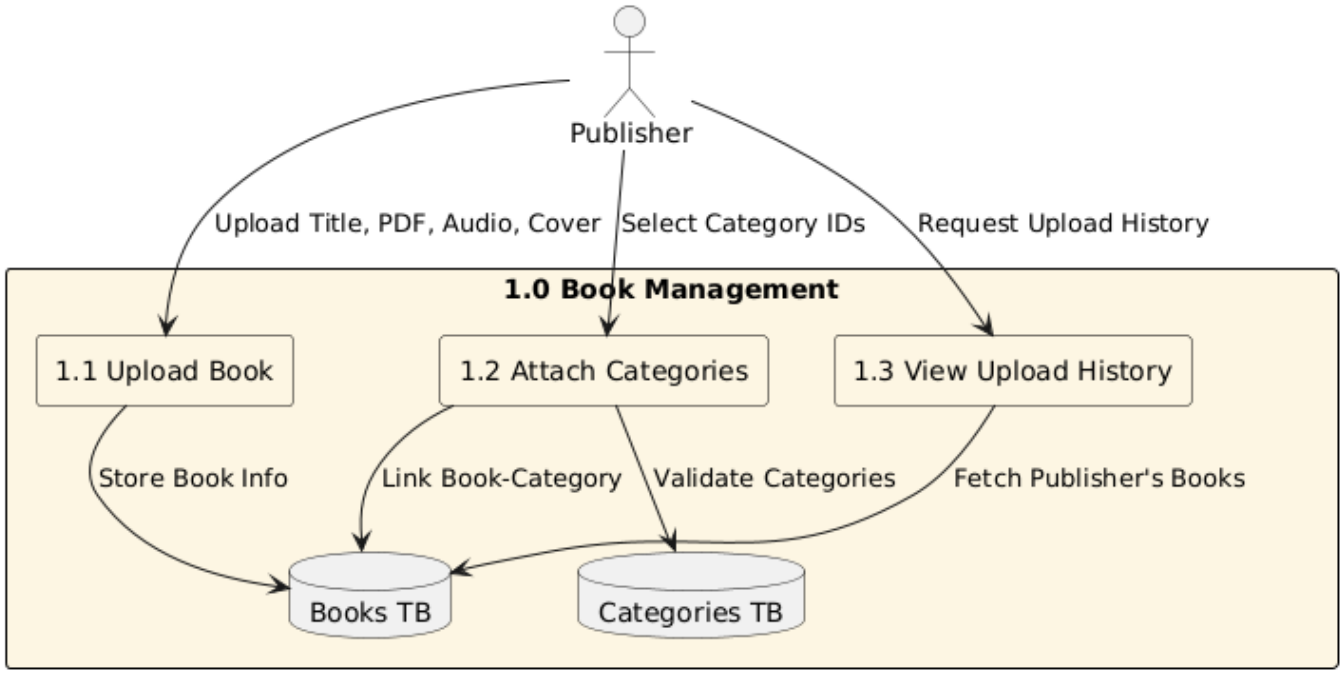
Level 1 DFD



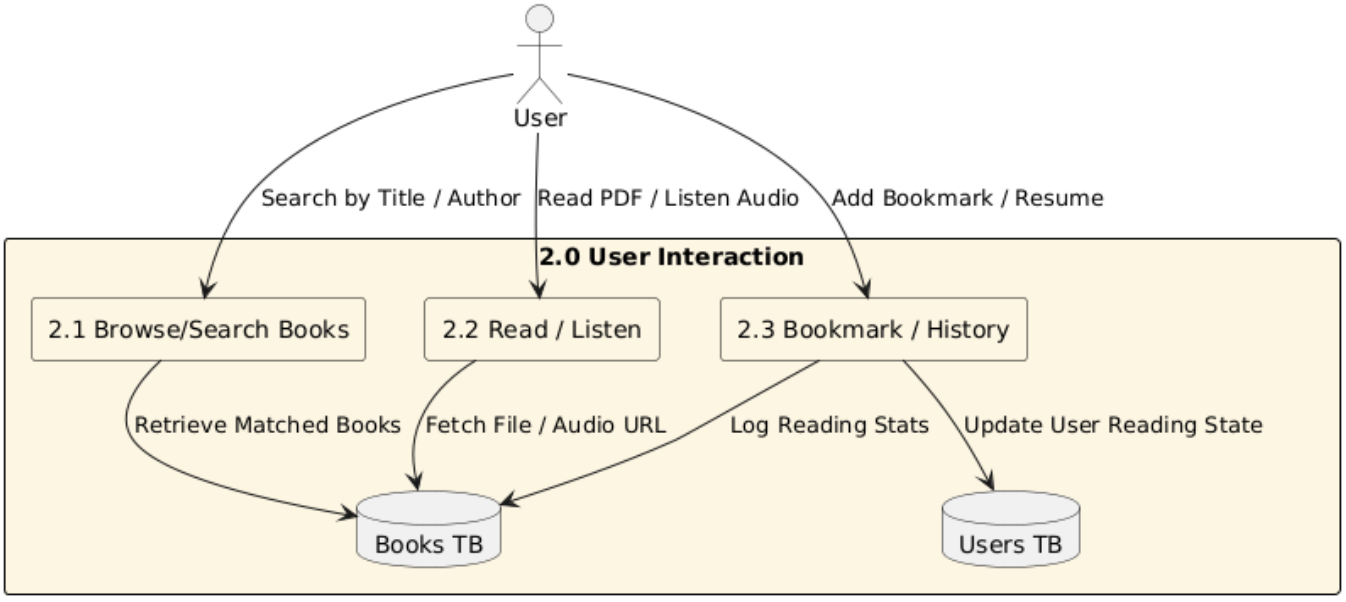
Level 2 Public Administrator



Level 2 Publisher



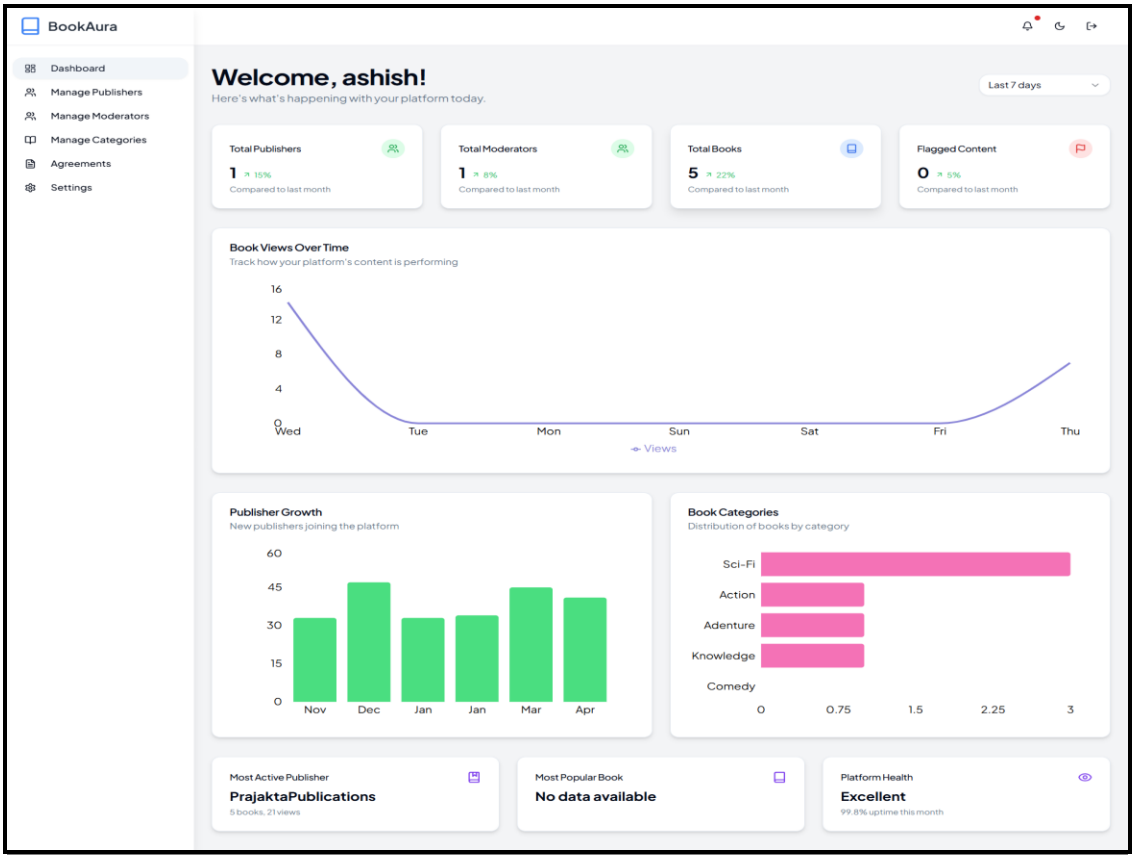
Level 2 Normal User



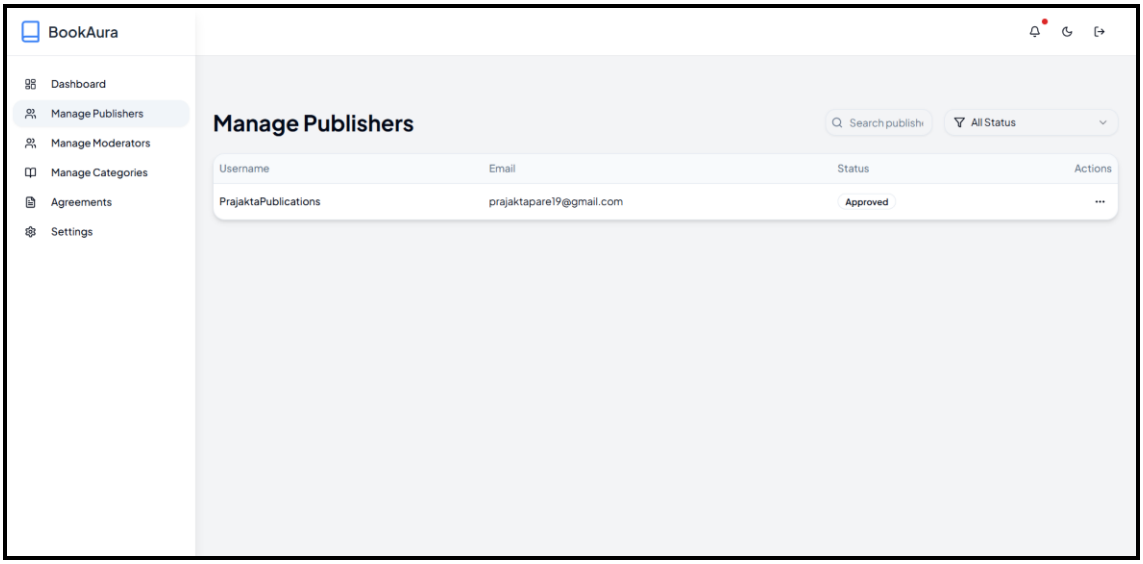
3.3. UI Screenshots

a) Platform Administrator

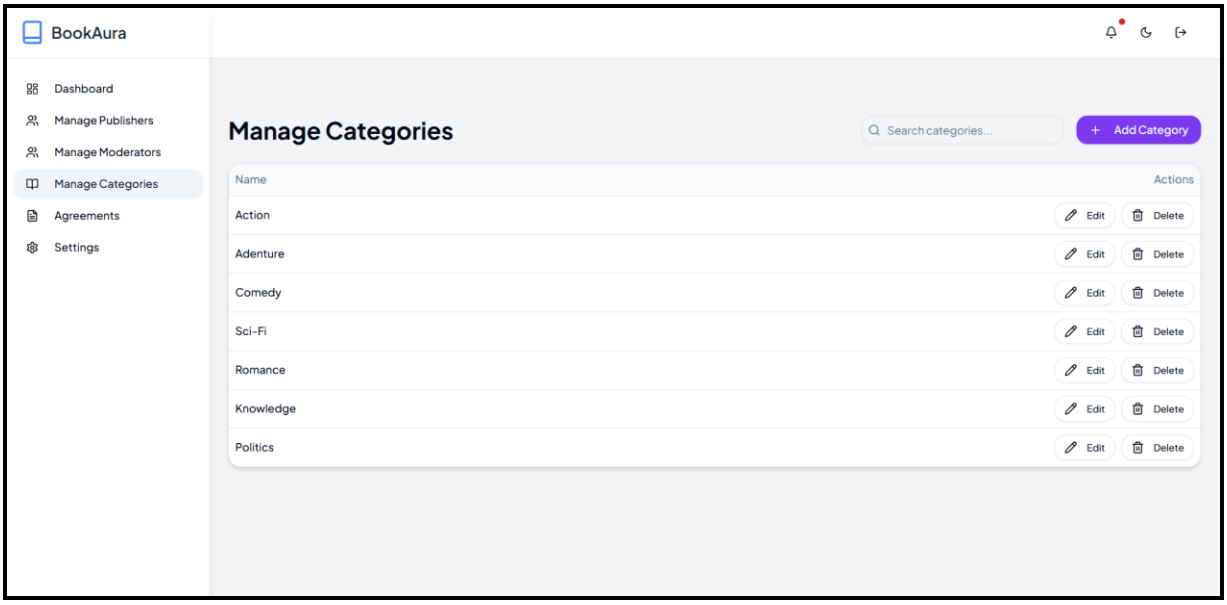
i) Dashboard



ii) Manage Publishers

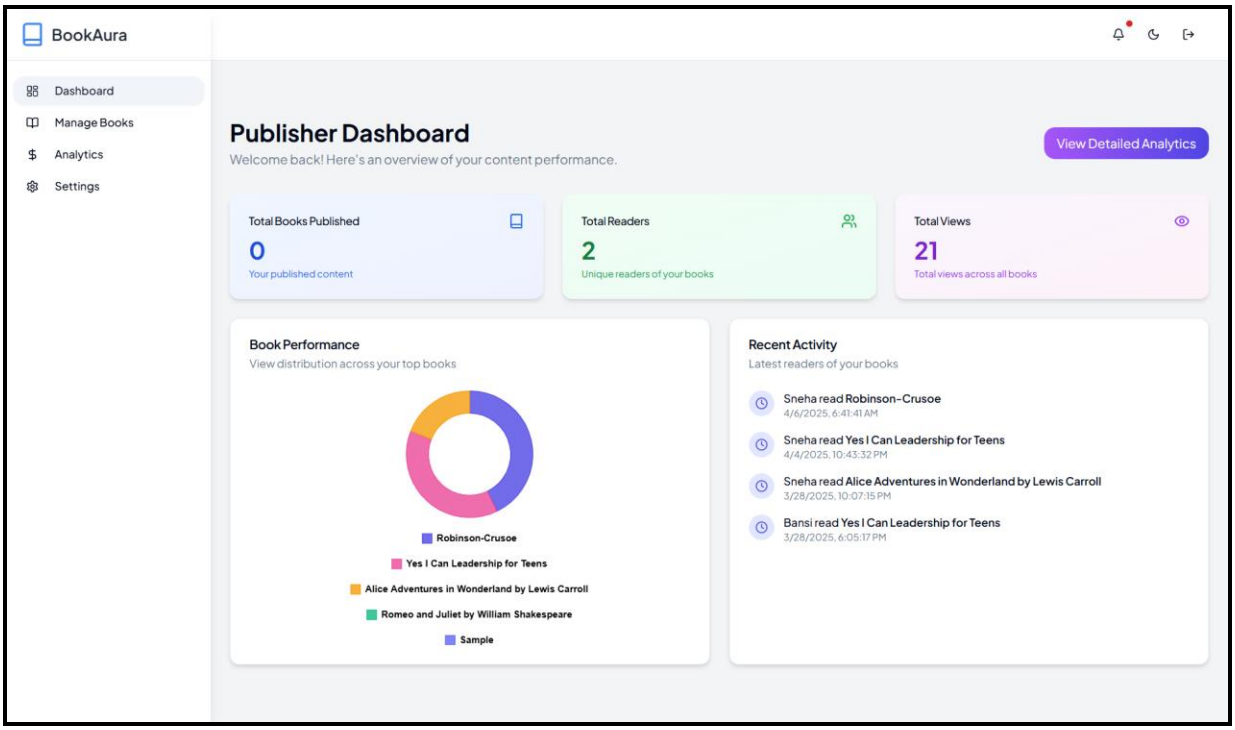


iii) Manage Categories

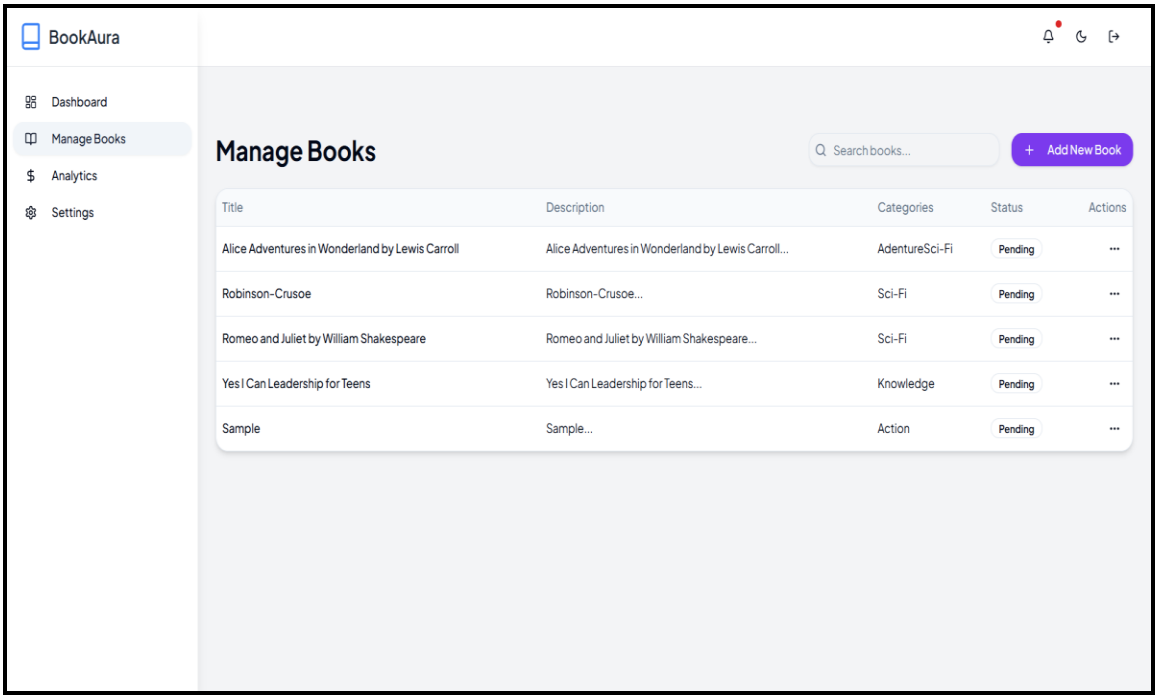


b) Publisher

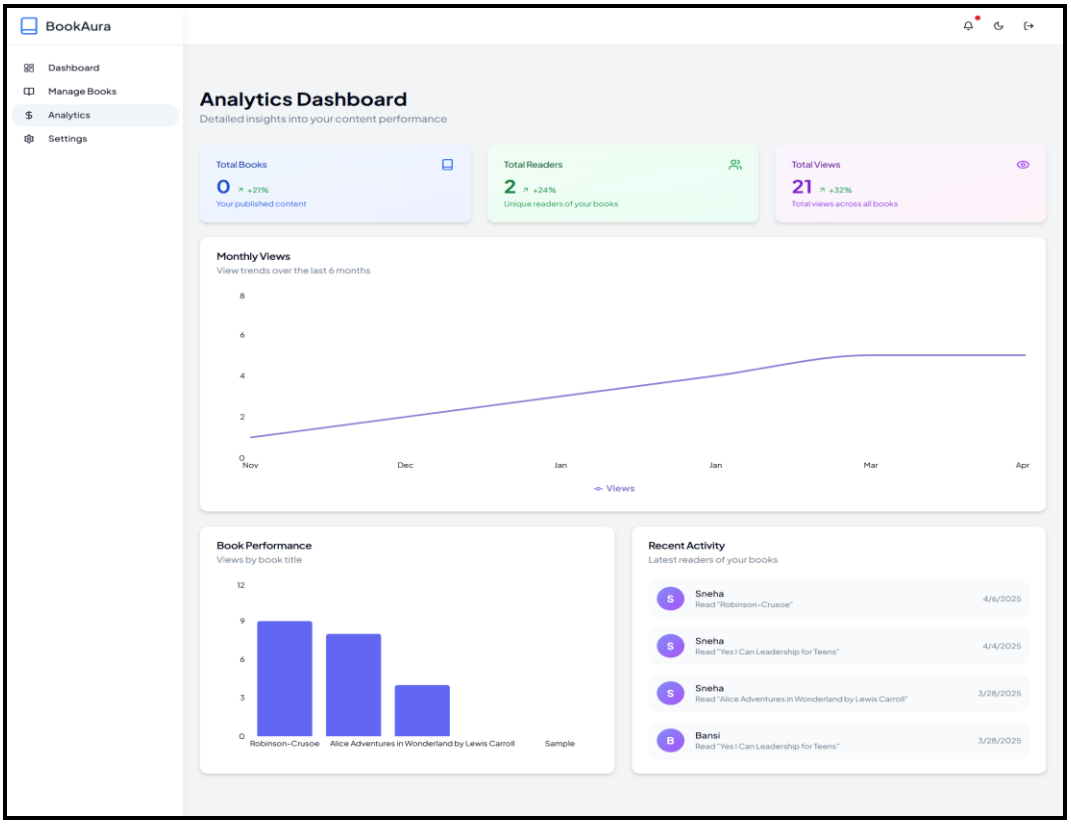
i) Dashboard



ii) Manage Book

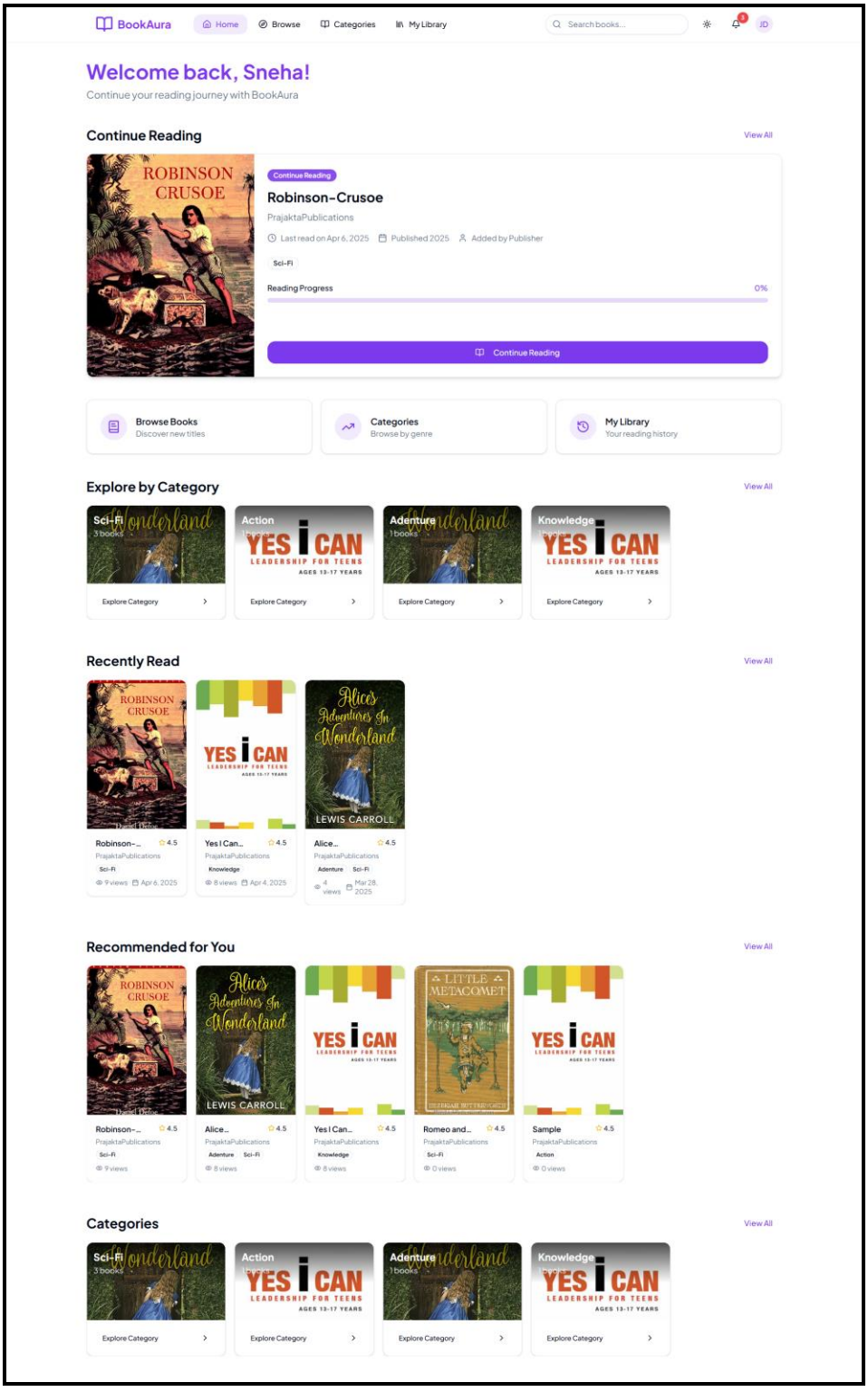


iii) Analytics

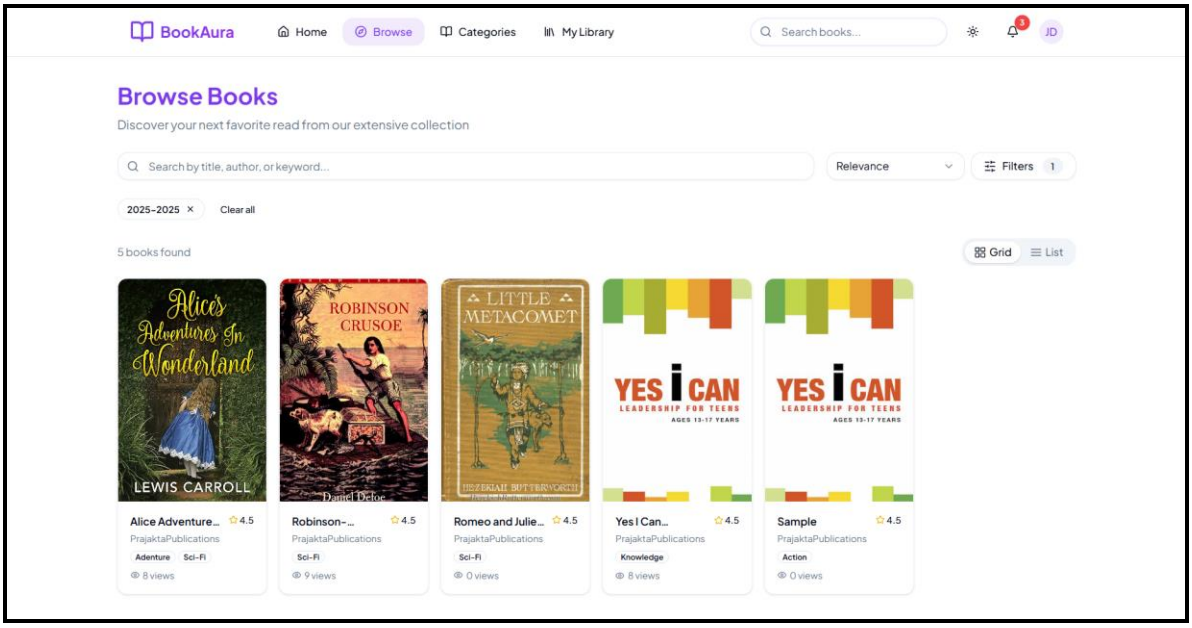


c) User

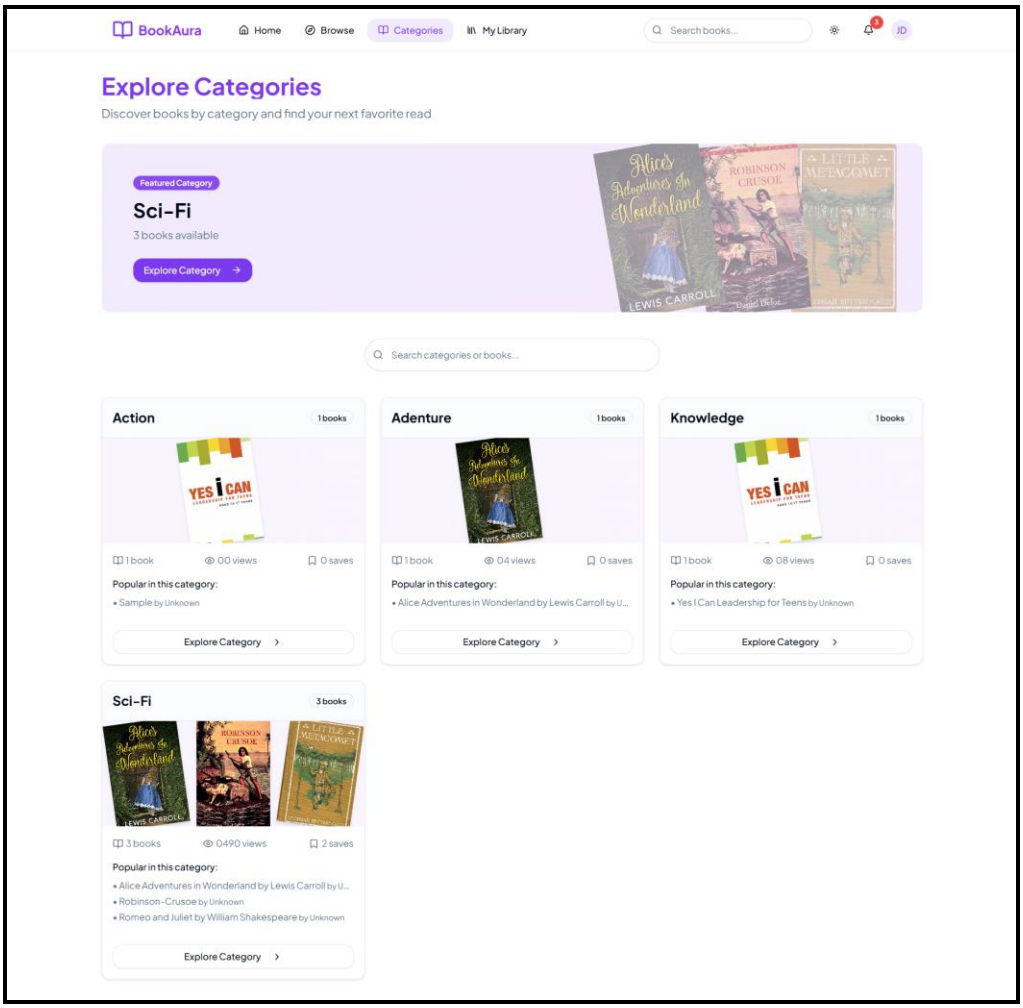
i) Home


BookAura
Home
Browse
Categories
My Library
Search books...
Welcome back, Sneha!
Continue your reading journey with BookAura
Continue Reading
View All
ROBINSON CRUSOE
Continue Reading
Robinson-Crusoe
PrajaktaPublications
Last read on Apr 6, 2025
Published 2025
Added by Publisher
Sci-Fi
Reading Progress
0%
Continue Reading
Browse Books
Discover new titles
Categories
Browse by genre
My Library
Your reading history
Explore by Category
View All
Sci-Fi
3 books
Action
Adenture
Knowledge
YES I CAN
LEADERSHIP FOR TEENS
AGES 13-17 YEARS
Explore Category
Recently Read
View All
Robinson-...
Yes I Can...
Alice...
LEWIS CARROLL
Recommended for You
View All
Romeo and...
Sample
Categories
View All

ii) Browse



iii) Categories



4. Testing

4.1 Importance of Testing

Testing is critical to ensure the reliability and security of the **book management and audiobook generation platform**. Given the **role-based access control (RBAC)** and **multiple user roles (authors, publishers, moderators, and administrators)**, testing ensures:

- **Seamless User Access**: Prevents unauthorized users from accessing restricted areas.
- **Accurate Book Management**: Validates book uploads, approvals, and accessibility.
- **Efficient Audiobook Processing**: Ensures text-to-speech (TTS) functionalities work correctly.
- **Data Integrity**: Confirms that **views, bookmarks, and reading history** are recorded properly.
- **Content Moderation Workflow**: Ensures that flagged content is reviewed and processed correctly.

4.2 Types of Testing

1. Unit Testing (Component-Level Verification)

- **Scope:** Focuses on individual components such as **user authentication, book upload, and TTS processing**.
- **Tools:** Jest (React frontend), PyTest (Flask backend).
- **Example:**
  - Testing /login API to verify that **only valid users can authenticate**.
  - Testing the **audio generation function** to ensure it produces a valid output file for given book text.

2. Integration Testing (Module Interactions Validation)

- **Scope:** Verifies interactions between modules like **user roles, book management, and audio requests**.
- **Tools:** Postman (API Testing), Selenium (UI testing).
- **Example:**
  - Ensuring that **a book uploaded by an author** is **not immediately available to the public** until a moderator approves it.
  - Testing that **users can request audio conversions**, and the **backend processes and returns the correct file**.

5. Drawbacks and Limitations

While the system offers a comprehensive **book management and audiobook generation platform**, there are certain **limitations**:

5.1 Technical Limitations

- **Limited Language Support for Audiobooks:** Currently supports only **English, Hindi, and Marathi**, making it less useful for users wanting other languages.
- **Processing Time for Audiobooks:** Large books may experience a **delay in text-to-speech conversion** due to processing constraints.

5.2 User Experience Limitations

- **No Offline Mode:** Users cannot download books or audiobooks for offline access.
- **Moderation Delays:** If **moderators are unavailable**, book approvals might take longer than expected.
- **Limited Personalization:** The platform does not yet offer **AI-based book recommendations**.

5.3 Security Limitations

- **No End-to-End Encryption for Audio Files:** Audio files are stored securely but are not encrypted in transit.
- **Potential Data Breach Risks:** If a user's credentials are leaked, their entire library and activity history can be accessed.

6. Future Enhancements and Conclusion

6.1 Future Enhancements

To improve the system, the following **enhancements** are planned:

A. AI & Personalization

AI-Based Book Recommendations: Implement a **machine learning model** to suggest books based on user preferences.

Smart Audiobook Summarization: Add an **AI-driven summary generator** to help users get quick insights into books before reading.

B. Feature Enhancements

Offline Mode: Allow users to download books and audiobooks for offline access.

Live Text-to-Speech Reader: Instead of pre-converting books into audiobooks, allow real-time **text-to-speech narration**.

Multi-User Collaboration: Enable multiple users (co-authors) to work on a book together.

C. Security & Performance

End-to-End Encryption for Audio Files: Enhance security by encrypting all stored audio files.

Load Balancing for High Traffic: Implement auto-scaling **server clusters** to handle peak loads efficiently.

7.2 Conclusion

The **book management and audiobook generation platform** successfully integrates multiple features, including:

Role-based user management (Publishers, Moderators, Administrators).

Secure book publishing and approval workflows.

Text-to-speech audiobook generation.

User engagement tracking with analytics dashboards.

7. Bibliography

This section provides references to the **technologies, frameworks, and research materials** used in the development of this project.

- **Frontend Development**
  - React.js: https://react.dev/
  - Tailwind CSS: https://tailwindcss.com/
- **Backend Development**
  - Flask: https://flask.palletsprojects.com/
  - Node.js (For additional services): https://nodejs.org/
- **Database Management**
  - MySQL: https://dev.mysql.com/doc/
- **Text-to-Speech Processing**
  - Google Text-to-Speech API: https://cloud.google.com/text-to-speech
  - pyTTSX3 Python Library: https://pypi.org/project/pyttsx3/
- **Hosting & Deployment**
  - AWS (S3, Lambda, RDS): https://aws.amazon.com/
  - Vercel for Frontend Deployment: https://vercel.com/