SMART SKILL E-BOOK MAKER SYSTEM

Submitted by

Ayesha Shabbir (FA21-BSCS-316-E)

Aleesha Amjad (FA21-BSCS-288-E)

Session 2021-2025

Supervised by

Ms. Ayesha Nasir

Department of Computer Science

Lahore Garrison University

Lahore

SMART SKILL E-BOOK MAKER SYSTEM

A project submitted to the

Department of Computer Science

In

Partial Fulfillment of the Requirements for the

Bachelor's Degree in Computer Science

By

Ayesha Shabbir

Aleesha Amjad

Internal Supervisor

Ms. Ayesha Nasir _____
Lecturer
Department of Software Engineering

External Examiner _____

Chairperson

Dr. Arfan Ali Nagra _____
Associate Professor
Department of Computer Science

COPYRIGHTS

This is to certify that the project titled "Smart Skill E-Book Maker System" is the genuine work carried out by Ayesha Shabbir and Aleesha Amjad, student of BSCS of Computer Science Department, Lahore Garrison University, Lahore, during the academic year 2021-25, in partial fulfilment of the requirements for the award of the degree of Bachelor of Computer Science and that the project has not formed the basis for the award of previously of any other degree, diploma, fellowship or any other similar title.

Ayesha Shabbir _____

Aleesha Amjad _____

DECLARATION

This is to declare that the project entitled “Smart Skill E-Book Maker System” is an original work done by undersigned, in partial fulfilment of the requirements for the degree “Bachelor of Science in Computer Science” at Computer Science Department, Lahore Garrison University, Lahore.

All the analysis, design and system development have been accomplished by the undersigned. Moreover, this project has not been submitted to any other college or university.

Group Members

Ayesha Shabbir _____

Aleesha Amjad _____

Supervisor

Ms. Ayesha Nasir _____

Date: _____

DEDICATION

This project is wholeheartedly dedicated to all learners, educators, and self-improvement enthusiasts who believe in the power of knowledge and self-development. It is also a tribute to the creators, writers, and dreamers who have inspired others through written word and digital creations. We offer our deepest respect to everyone who inspires change by fostering skill development and lifelong learning.

A dedicated note of gratitude to our families, friends, and mentors for unwavering support, encouragement, and belief in our work, and unlimited patience and tolerance for our working methods. Their presence in our work was the fuel to our perseverance. Finally, this work is intended for future generations of learners and creators it is our hope that it empowers them to imagine bigger than they dream, learn infinitely, and create tools that foster access and equity in knowledge for all.

ACKNOWLEDGEMENTS

On behalf of our development team, we would like to extend our sincere appreciation to all of you who helped us realize our goal to develop the Smart Skill Building E-book Maker System App.

We like to sincerely thank to Ms. Ayesha Nasir, our supervisor, we owe a debt of gratitude for all her valuable direction, support, and expert recommendations from start of the project through to completion. Your encouragement and suggestions helped make this system the success it has become.

To our team, thank you so much for working tirelessly through each step of the development project. We could not have done this without your commitment and team mentality. Everyone played a piece of the puzzle, as each person's unique skills, creativity, and determination helped us overcome challenges and turn our thoughts into a completed and functional system.

To our early testers and users, we would like to thank you for participating in our early testing process and for your constructive feedback and suggestions. You assisted us in improving and developing our system by providing your experience and thus we were able to make it more effective, user friendly and reliable for everyone who accesses the app.

Finally, to the open-source community that provided so many tools, libraries, and documentation to assist in the development of our GEMS App, thank you so much for your many contributions. The thinking and culture around open-source and the sharing of knowledge makes the resolution of problems easier, and we learned best practices to make our application run more efficiently and increase our productivity.

Table of Content

List of Tables

List of Figures

List of Abbreviations

AI: Artificial Intelligence

API: Application Programming Interface

CRUD: Create, Read, Update, Delete

DB: Database

HTML: Hypertext Markup Language

IDE: Integrated Development Environment

JS: JavaScript

JSON: JavaScript Object Notation

MVC: Model-View-Controller

NoSQL: Not Only Structured Query Language

NPM: Node Package Manager

NPX: Node Package Execute

RN: React Native

SDK: Software Development Kit

UI: User Interface

UX: User Experience

URL: Uniform Resource Locator

VS Code: Visual Studio Code

JWT: JSON Web Token

Expo CLI: Expo Command Line Interface

ABSTRACT

With the rapid advancement of online learning, the need for tools that promote interactive, skills-based, content creation rather than passive reading is growing. Traditional e-book platforms lack flexibility, AI capability, and user-driven customization, and this limits their effectiveness for modern learners and educators. In order to resolve this gap, the Smart Skill E-Book Maker System is proposed: an amazing mobile-based application that empowers students, teachers, and professionals to create and edit as well as interact with structured digital e-books. All frontend development is done in React Native, back-end development is in Node.js environment, and data is stored in a MongoDB server, ensuring a responsive and dynamic development environment. The smart skill e-speaker system utilizes the Gemini API to offer AI-generated content suggestions and intelligent templates for any individual use. Significant features include reading and writing e-books, saving content, creating smart recommendations, and easy sharing of the content, all designed to promote engagement and personalized learning. The system is built around a modular and API-system based architecture, which results in scalability, flexibility, and easier maintainability, while enhancing the user experience, for both the admin and clients' use. This solution was created for addressing the void between static digital books and immersive and skill-based learning.

Chapter 1

INTRODUCTION

1.1 Background

Skill development and self-learning are essential to personal and professional development in the digital world we are living in. As more learning is being facilitated online, people witness huge advances in their skill development abilities using digital platforms and digital tools. Even with the many e-learning platforms that exist, many lack customization from the user's perspective, collaboration and an AI-supported approach to learning at their pace.

Skill-building through an e-book enables an innovation experience. On the one hand, the user has the traditional learning structure, and on the other hand, it has the digital customization option. A traditional e-book offers static learning content without interaction or creativity on the part of the user. The opportunity for a platform that allows users to build, modify, organize, share or developing their skill development content, using AI and mobile technologies is both timely and valuable.

In this Final Year Project document, we present to you the Smart Skill Building E-book Maker System App, an AI-supported mobile application that will assist users to design and manage customized skill-based e-books. The application will quickly assist users with the creative content discussed, with some smart functionality such as: AI feedback, templates, time to customization, voice to text, user sharing and collaborative building. The system will promote an organized learning environment but more importantly there are dedicated user’s roles for admins and clients.

1.2 Over View

The Smart Skill Building E-book Maker System is a mobile application built using react native, with a Node.js backend on the server side, and a MongoDB database that utilizes Firebase to enable real-time storage. There are two types of users on the platform: Admin and Client. Admin manage content templates, content categories, and content interactions among and between users; and Clients (Users) only create and refine their books. Future capabilities will add offline access, gamification features, and lastly smart notifications for user engagement.

As a mobile application that incorporates both reading and writing, this application allows the user to:

- Make/read books by customizing text, images, and templates.
- Store and organize books by folders.
- Utilize AI for grammar and layout, and structural suggestions.
- Use voice to text features, and language transformation for multi-languages.
- Share books with peers or mentors for feedback and rating.

1.3 Aim and Objective

The overall aim of this project is to create an intelligent mobile application that will allow users to create, drive and share personalized skill-building e-books using AI features and cloud storage.

Objectives

- To create the React Native mobile app using Expo CLI to ensure best usability and access across platforms.
- To use Node.js and MongoDB for backend and database management.
- To provide admin and client roles so that there is a defined scope of roles and capabilities.
- To allow users to create, edit, delete and store their e-books in categorized folders.
- To allow AI suggestions in writing content, layout, and grammar.
- To integrate with Firebase for storing user data, images, and e-books files.
- To include smart templates, voice to text, and language translation.
- To provide for sharing of e-books, a feedback system and collaborative interaction.
- To provide for cloud synchronization, while keeping offline functionality open for future enhancement.

1.4 Project Scope

This system will be directed towards individuals like students, trainers, and professionals who wish to build their own structured learning content or want to publish knowledge based on experience digitally. The project will not be focused on more advanced publishing capabilities (e.g. print distribution) or full offline editing at launch but it is built with the idea of expandability. The app will provide:

- Ability to create e-books with rich formatting capabilities.
- Different template categories for skills such as programming, business, design, etc.
- User-created e-books will be saved and retrieved through cloud storage.
- Access from mobile using React Native on Android and iOS.
- Possibility for expansion for later use of gamification and tracking progress.

1.5 Gantt Chart

The project development was completed with a structured timeline through phases which included requirement gathering, design, implementation of front end and back end, integration of the AI feature, testing, and writing up a report. The Gantt chart below shows the timeline of tasks that were completed and project milestones reached.

TASK	NOV (7th - 15th)	DEC (9th - 20th)	JAN (1st - 25th)	FEB (5th - 15th)	MARCH (1st - 25th)	APRIL (10th - 20th)	MAY (1st - 10th /15th - 25th)	JUNE (5th - 25th)	JULY (1st - 30th)
	2024		2025						
1.Plannig									
i. Topic selection									
ii. Material gathering									
iii. Proposal defense									
2.Designing									
i. ERD									
ii. DB schema									
iii. UI/UX									
iv. Flow chart									
v. Model diagram									
3.Documentation									
SRS									
4.Development									
i. Front-end									
ii. Backend									
iii. AI intergration									
5.Testing									
6.Deployment									
i. Report									
ii. Presentation									
iii. Final submission									

Figure 1: Gantt Chart of Project

1.6 Report Organization

The report consists of nine chapters:

Chapter 2: Literature Review current systems, technologies and research in the area of e-book creation, AI-assisted writing and educational skill development platforms.

Chapter 3: Problem Definition Provides details of the areas of problems users encounter in existing systems and where there is a need for a smart user-centric e-book creation solution.

Chapter 4: Software Requirement Specification Covers full procedures in defining the functional and non-functional requirements, as well as outlining the user types, use case diagrams and system usability.

Chapter 5: Methodology Clarifies the software development lifecycle were followed, tools and technologies used and the rationale behind each decision.

Chapter 6: Detailed Design and Architecture Covers system architecture, data flows between components, database schema and user interface design layouts.

Chapter 7: Implementation and Testing Outlines development processes undertaken, documented testing strategies and test cases completed, and user validation.

Chapter 8: Results and Discussion Analyzed system performance and feedback when using the system by participants in many varied experiences. This also examined how the system met the goals set.

Chapter 9: Conclusion and Future Work Summarized findings from the project and discussion on subsequently incorporating enhancements, including offline access to an e-book, gamified e-learning, and intelligent notifications.

Chapter 2

LITERATURE REVIEW

Advancements in AI and mobile technologies have enabled the production of personalized and smart learning systems. These systems are designed to help learners learn more efficiently by providing content creation support, intelligent guidance, and flexible access. The author of the paper [1], proposed a mobile learning framework that provides learners access to content whenever and wherever they would like. The framework provides more flexible learning opportunities, which helps with learning retention. The mobile learning framework was implemented with university students and provided them an improved experience with improved engagement levels as a result of a more user-friendly, mobile-based interface. The goals of the proposed framework are well aligned with the objectives of the Smart Skill E-Book Maker System as it provides the means to create content anywhere while also being built using a mobile-friendly application via Expo CLI.

The author of the paper [2], proposed an AI based educational assistant that would provide customized learning material based on learning performance and topic interests. The implementation of the assistant utilized NLP models and was incorporated into an e-learning platform. The platform was evaluated with 500 users which measured both learning speed and assessment scores, resulting in a 23% faster learning speed and 18% increase in assessment scores. Our Smart skill E-Book Maker System also utilizes AI functionality through the differential features within the Gemini API. Features such as template suggestions and real-time writing suggestions for learners' writing projects. Author of the paper [3] created a content generation tool for learners of language that allowed learners to develop their own grammar e-books. The e-book development system, which works on the user's browser, was implemented in Node.js and Firebase and had translation functionality powered by Google Translation API. A study conducted on high school students found they had better retention and developed more advanced grammar skills after using the system. Our system builds on this model, with MongoDB as the back-end and also adds multilingual capabilities to facilitate use by a broader audience.

Similarly, the author of the paper [4], specified a modular back-end architecture using RESTful APIs with MongoDB and Node.js. They used this technology to facilitate scalability, fault tolerance, and rapid access to data in their e-learning system. Their system was able to support over 300 simultaneous users without any lag or slowdown by adapting to a different server

provisioning. Thus, it could be used in a multi-user context for live educational purposes. Our project will implement a similar technology, in order to assist in the management of many users between the application and users on their respective devices. Author of the paper [5], emphasized the importance of security in e-learning platforms, and proposed a system based on role-based access control (RBAC) and JWT authentication. This was tested in penetration testing scenarios and successfully blocked 95% of unauthorized access attempts. Our Smart Skill E-Book Maker incorporates similar secure access controls for Admin and Client users in order to maintain the security and privacy of data. Author of the paper [6], assessed the usability of AI-based platforms, using SUS (System Usability Scale), which said too much automation could overwhelm new users. The author recommended a minimalistic UI and displaying only the core AI based features. Our system mirrors this with the incorporation of only the essential AI features: smart suggestions and voice to text, and inner application cleanliness through Expo CLI and React Native.

Chapter 3

PROBLEM DEFINITION

Introducing yet another major set of problems is the Smart Skill E-Book Maker System, which, in itself, aims to provide a dynamic and user-oriented platform for designing and consuming skill increasing content. Offline functionality must be put in place to allow users to continue accessing and editing their e-books without relying on the internet. Core features such as saving and deleting content must work accurately and efficiently to safeguard against loss of data. The technical challenges in implementing real-time AI feedback and intelligent content customization must be addressed while also keeping in mind performance levels on low-resource devices. Moreover, the system is supposed to maintain an integrated multilingual capability for a diverse user base without having to put the users through the hassle of switching between tools for a seamless experience. Role-based access control is required to differentiate between users, educators, and administrators, all of whom have different capabilities and workflows. Moreover, collaborative features such as concurrent content editing and peer review further complicate the content management and synchronization process. Tackling these challenges forms the basis for building a scalable interactive intelligent platform that works to support modern skill development and personalized learning.

Chapter 4

SOFTWARE REQUIREMENT SPECIFICATION

4.1 Introduction

A mobile app called the Smart Skill E-book System was developed to help individuals develop their skills with a reach on creating, editing and sharing their interactive e-books. This Software Requirement Specification (SRS) describes the system requirements and limitations, overall system design, user roles and system objectives. This document will provide a base for development, and help ensure all parties; including developers, testers, and end users, have a common understanding of the intended purpose and functionality of the system.

4.1.1 Purpose

The purpose of this project is to develop an easy-to-use e-book creation platform enriched with intelligent tools that support learners and educators in building and sharing skill-based content. The system reduces the complexity of traditional e-book tools by integrating smart templates, content customization, and collaborative features.

4.1.2 Document Conventions

The SRS font style is Times New Roman, in accordance with the document norms. For the headings, the main heading uses the heading 1 style with a font size of 16, the subheadings use the heading 2 style with a font size of 14, and the heading 2 style with a font size of 13. The font size used in the paragraph is 12.

4.1.3 Intended Audience and Reading Suggestions

The Software Requirement Specification (SRS) document was designed for a spectrum of stakeholders that play roles in planning, developing, testing, assessing, and improving the Smart Skill E-Book Maker System. Each group of reader will be directed to sections that relate to their own context and purpose. The purpose of this section is to help readers connect with the sub-sections of the document that relate the most to their goals.

A. Project Supervisors and Academic Evaluators

- **Purpose of Reading**: To check that the project meets FYP criteria, has a defined scope, and is technically feasible.
- **Suggested Sections**
  - Section 1 (Introduction) –to understand the problem being solved.
  - Section 2 (Overall Description) –to understand the context of the system,
  - Section 5 (Non-Functional Requirements) –check the quality attributes of the system.

B. Developers, technical Team Members

- **Purpose of Reading**: To be able to implement the system as specified in functionality, interfaces, and performance criteria.
- **Suggested Sections**
  - Section 3 (External Interface Requirements) –to understand the APIs, UI expectations, and software dependencies.
  - Section 4 (System Features) – necessary details on core functionality.
  - Section 2.5–2.6 (Constraints & Dependencies) –to realistically plan development efforts and integrations.

C. Testers and Quality Assurance Engineers

- **Purpose of Reading**: To create a test cases, validate that the system meets all requirements.
- **Suggested Sections**
  - Section 4 (System Features) – to develop the test scenarios.
  - Section 5 (Non-Functional Requirements) – to validate usability, performance, and security.
  - Section 2.2 (Product Functions) –to confirm what the system should do.

D. Future Developers and Maintenance Teams

- **Purpose of Reading**: To learn about the system structure, the reasons for the design and the potential for future development.
- **Suggested Sections**: Entire Document, but especially:
  - Section 4.1.4 (Product Scope)

- Section 4.2.1 (Product Perspective)
- Section 4.4–4.5.4 (System Features and Quality Attributes)

E. Non-Technical Stakeholders

- **Purpose of Reading**: To be clear on how the system will support their role and how they can utilize it for creating and sharing content.
- **Suggested Sections**
  - Section 1 (Introduction) – to understand the user-centric goals.
  - Section 2.3 (User Classes) – to clarify how their role fits into the system.
  - Section 4.1–4.2 (Features Overview) – to outline what they can and cannot do in the app.

4.1.4 Product Scope

There are many different job options available in the gaming industry. It also accounts for a sizable portion of the employment gap. As many as 1.7 million people are employed in the gaming sector, according to data released by the American Gaming Association, and employment is increasing by 62,000 positions annually on average.

Simulation & Gaming has been a preeminent global platform for the study and exchange of ideas about simulation/gaming approaches in research, education, training, healthcare, and consulting for over 50 years. This excellent quarterly journal investigates the approaches' applicability to actual issues and circumstances in addition to scrutinizing them.

Shooting games are a highly particular genre of the video game industry, and there aren't many good 3D shooting simulation games that provide you a large amount of variation, including survival and learning, which is the major reason we made this game.

4.2 Overall Description

4.2.1 Product Perspective

The Smart Skill E-Book Maker System is designed as an independent mobile application, and not related to any current platform. It is intended to be a self-sufficient system that helps users create structured e-books for skills development through AI-assisted systems. The System is designed with a layered architecture that separates user interface, server logic, AI integration, and database storage. This architectural structure provides scalability, maintainability, and a modular approach enabling evolution over time without changing the main structure.

The application has two types of users: Admin and Client. The admin user is responsible for creating and providing smart templates for the Client's usage in creating e-books. These smart templates are structured formats to guide the Client through content creation. The Client user accesses the application from a responsive, cross platform mobile interface designed in React Native, selects a smart template to utilize, and creates their content through the AI-assisted Gemini API which provides context prompts to help the user better structure their writing in the context of the chosen template.

Each operation happens within the application itself. When content is created in the app, it is able to be displayed during the session, but it is not retained. The system purposely restricts the ability to save or export, or delete the content the creator makes. This is in line with the project's deliverable a lightweight session-based experience to create content.

A. Dependencies on Third-Party Software

The Smart Skill E-Book Maker System overreliance on the following third-party software and services to work properly:

- **React Native**: Used for building the cross-platform mobile application interface that works on Android & iOS devices.
- **Node.js and Express.js**: The backend frameworks used to manage the business logic, Api routing, user roles, and manage session.
- **MongoDB Atlas**: A cloud-based NoSQL database used to manage the smart templates and session information.
- **Gemini API**: An AI, external service that was used in our system that enables contextual suggestions for content structures and writing assistance.
- **HTTPS Protocol and JSON**: Used to safely communication between the frontend, backend and all third-party services, all using structured data with JSON standard.

B. Limitations of the Proposed System

The following limitations describe the scope and edges of the Smart Skill E-Book Maker System:

- **Dependent on the Internet**: The application will not work without an internet connection because it depends on online databases and services that make use of AI.
- **No Multilingual or Translation Capabilities**: There is no content translation or ability to input any content in multiple languages. The user is expected to know the default language.

- **No Grammar tools or Voice inputs**: Grammar correction, feedback, or voice to text input are features that are not utilized in this version.
- **Mobile-Only Platform**: The system is limited to mobile devices (i.e., Android/iOS). There is no web or desktop access.
- **Single-User Mode**: This app will have to be for individuals. There will not be any content that allows user to collaborate or for multi-user.

4.2.2. Product Functions

The Smart Skill E-Book Maker System has a number of basic functions that perform intelligent and guided mobile e-book creation. The main system functions consist of:

- **User Authentication**
  - Users can securely register and log into the application.
  - It supports role-based access (Admin and Client).
- **Access to the Dashboard based on User Role**
  - Admins will be directed to a Template Management interface.
  - Clients will be directed to the e-book creation area.
- **Template Management**
  - Admins can create, update and delete smart e-book templates.
  - A template defines the type of structure (i.e. Introduction, Chapter, Summary) to be used by Clients.
  - Admin can ensure that a template meets an educational or professional content requirement.
- **Template Selection**
  - Clients can view a list of templates created by Admin.
  - A template can be selected by topic or personal preferences to begin content generation.
- **AI Content Assistance**
  - The Gemini API will provide smart suggestions for headings, section ideas, and suggestions for layout.
  - The suggestions are based on the context of the user entry and template type selected.

- **Real-Time Content Input**
    - Clients can enter and modify content through the app during their session.
    - AI prompts are displayed inline, with the option to accept or ignore them.
- **Final View Mode**
    - Once content is complete, it is presented in a structured format for final review.
    - The user can scroll through every section before closing their session.

4.2.3 User Classes and Characteristics

The design of the Smart Skill E-Book Maker System provides for two different user classes, which are the Client users and Admin users. Each user class has very distinct roles, responsibilities and access to the system. These user classes are significant in defining how the system will be used and which parts of the system will be available to the various types of users based on their level of engagement.

A. Primary Users

The primary users of the system are the Client users which will use the system's core functionality via the application and will ultimately be the target audience of the platform. These are the users who will use the system with the main purpose of generating skill-enhancing content using the smart templates available in the application.

- **Role and Activities**

  Clients choose a template from a portfolio provided by their Admins and then are given access to build content using AI-simulated suggestions. They will enter text directly into the app, stick to the template's template structure and be provided support through the use of the Gemini API while building. Clients do not have control over templates or the ability to customize systems settings.

- **User Profile**

  Potential primary users could be students working on structured reports, or portfolios, professionals writing documents based on topics, or casual learners summing up new skills. These users should be able to demonstrate basic mobile application literacies, but should not be required to have technical understandings of backend services or AI systems.

- **System Interaction**

  Clients will only use a simplified user interface and will be restricted to a session-based method for workflows. A user's interaction ends when their content is complete within the app environment.

B. Secondary Users

Secondary users of the system are Admin users who oversee the system's structure and operational foundation. Unlike Clients, they don't create content but provide important input for the experience of the primary users.

- **Role and Activities**

  The admin creates, update, and delete templates that underpin all e-book content. Admins build templates that are organized, readable, and suitable for a learning or skill-building objective. Admins do not see any of the content created by Clients, nor do they have the AI assistance features available to them.

- **User Profile**

  The secondary users can be people like instructors, content designers, training coordinators or managers of the organization that are identifying, managing the template formats available. These users will need to understand the system at a moderate level to build effective templates but are not "coding" or managing any infrastructure.

- **System Interaction**

  Admins utilize a separate dashboard to manage the life-cycle of templates. Their work is continuous in nature, ensuring that Clients always have a set of relevant and well-structured options to choose from when developing content.

4.2.4 Operating Environment

The Smart Skill E-Book Maker System is developed in a lightweight, mobile-first inception that maximizes the usability of the front-end user, as well as the backend efficiency for the server. On the server-side, the Smart Skill E-Book Maker System is dependent on a cloud-based Linux server with the Node.js runtime environment (v.14 or higher). It needs a minimum of 4GB RAM, multi-core CPU capacity, and 50GB storage. The MongoDB Atlas cloud database provides the structure and user sessions.

The front-end utilizes React Native, providing a responsive User Interface (UI), and runs on Android (8.0+) and iOS (11+) platforms. Real-time AI suggestions are possible via the Gemini API. APIs via HTTPS protocols for secured communication.

Clients can use the application from any recent mobile phone with a stable internet connection. The application is a native app and does not run in a browser, it only works if it is installed directly on the mobile app. The application requires, at a minimum, 2GB RAM and dual core processors to run smoothly.

AI outputs are provided in real-time, and average response times range from 2-3 seconds - depending on overall network speed. There will be no offline mode, ability to save files, or export files as everything operates in one session.

Security is provided through role-based access control and secured API calls and backend level validations. Routine testing is supported by tools like Postman to review requests. The system accommodates multiple users concurrently, without degradation in functionality, fitting the role. Overall, the environment provides Admins and Clients a seamless, dependable, and context-focused platform for content creation on mobile devices.

4.2.5 Design and Implementation Constraints

The Smart Skill E-Book Maker System is designed specifically for mobile platforms using React Native technology and does not provide access on desktop or web. Its session-based design is deliberately set up to restrict the ability to save, delete or edit anything after it has been created. Users are required to finish their work in one session. The system is completely reliant on a network connection to the Gemini API for AI generated suggestions and MongoDB Atlas for the storing of templates. The system does not support offline capabilities. Both core capabilities would cease if either the network connection was lost or if the API services were unavailable. The specifics of the backend occur with Node.js and Express.js, both require secure communications access through HTTPS to the roles and access control capabilities to restrict what features are available for different user's access in the app. The app is designed for low resource mobile devices and does not accept features like collaborative editing, media uploads or multilingual support. These decisions were made in order to keep the system lightweight, task focused, clean and efficient for mobile based, real-time, AI-produced e-booking using a single session as best practice.

4.2.6 User Documentation

The Smart Skill E-Book Maker System will be accompanied by clear and easy to use documentation to facilitate the use of the system by the Admin and Client. Admins will be instructed on maintaining templates and quality of content, while Clients will receive instructions on selecting and using templates, using AI suggestions, and working the e-book creation in a single session. The documentation may be available in the app, and it may be available to download as a guide. Documentation will include guided instructions for performing each task, visuals when required, and best practices so that the end user can work independently and with limited barriers to use the system smoothly as possible.

4.2.7 Assumptions and Dependencies

The Smart Skill E-Book Maker System assumes users have access to modern smartphones and/or tablets, and reliable internet. The application depends on presumption that third-party services are available to gather AI suggestions through the Gemini API and to store content to MongoDB Atlas. The application likewise relies on the admin users to consistently maintain quality of templates to support high quality education content. It is assumed that all communications are through secure HTTPS, and it is assumed that the application functions in a Linux server environment that supports Node.js and MongoDB. If any if these fundamental aspects of the technology fail, it may impact the application or accessibility.

4.3 External Interface Requirements

4.3.1 User Interfaces (UI)

The Smart Skill E-Book Maker System will provide an easy-to-navigate user interface and user experience for various users (learners, educators, administrators). The system will be designed as a responsive UI that is functional and effective on smartphones, tablets, and desktops. The UI will be developed through React Native and Expo CLI, to ensure seamless UI design across multiple OS platforms. Key highlights of the UI are as follows:

- **Home Screen**: Displays content based on the current user and provides users with quick access to create, read, or edit e-books.
- **Content Creation Interface**: Enables users to create, edit, and save e-book content utilizing AI powered templates and smart content suggestions.

- **User Dashboard**: Provides access to contacts from the roles a user has, i.e., educator can manage all their content; guide students in learning progress, etc.
- **Login/ Authentication Screen**: Allows user to securely log in using username and password.
- **Admin Panel**: Provides access to control and improve user management, content moderation and with modifying the settings for the system.
- **Interactive Components**: Including all the text editor components, AI smart content suggestions, and tools for real-time content collaboration.

4.3.2 Hardware Interfaces

Created as a standardized smart phone and tablet application, the Smart Skill E-Book Maker System does not demand expensive specialized hardware components. Using only the everyday hardware options in the phone and tablet device, the application can be slim and simple, using only touch-based interaction for usability on many screen sizes and standard mobile device specifications. There is very little dependence on hardware that can limit applications from the start and assure higher usability and experiences.

4.3.3 Software Interfaces

The Smart Skill E-Book Maker System is composed of numerous software interfaces that work together to provide seamless functionality and a smart user experience. The application features a front-end built in React Native with Expo CLI, which allows for development on Android and iOS platforms. The back-end built with Node.js to handle the API requests and send and receive information to and from the database. The information was stored in MongoDB, a NoSQL database, which provides flexibility and scalability to store structured and semi-structured content such as e-books. Along with the eBook Maker System, there is the Gemini API incorporated, which is interfaced by artificial intelligence to allow for intelligent suggestions on content and intelligent customization of templates. As software interfaces, the components within this e-book maker system can provide seamless operation, modular development of systems, and interactions. The different software interfaces will the software perform effectively together, while presenting a modular architecture, delivering optimized performance and real-time feedback, to ensure the experience for content development and content utilization is as simple as possible for e-book visitors.

4.3.4 Communications Gap

Communication gaps imply potential issues or constraints in how different components of the system may communicate or interact with one another. It is essential to close communication gaps to maintain a more seamless and consistent user experience throughout platforms and devices. The gaps may arise from:

- **Real-Time Communication**: Making sure concurrent features (like content editing and AI-assisted suggestions) work across devices and platforms in real time without a glitch.
- **Data Synchronization**: The system is capable of intelligently synchronizing data between client (user's device) and backend (server), especially in saving and loading e-book content.
- **Error Handling**: When errors occur, especially concerning connectivity or fetching data, a clear and informative error message should be displayed to the user.
- **Cross-Platform Compatibility**: As this system is developed with React Native, it occasionally experiences behavior without gaps on different operating systems (iOS vs. Android) and on different devices (phone vs. tablet).

4.4 System Features

4.4.1 AI-Based Content Suggestion

- **Description and Priority**
  - **Description**: This feature enables the user to get AI-generated suggestions based on the context of the text being written. It enhances the productivity of the author and ensures relevance in his/her creation of educational material. The feature uses the Gemini API to generate intelligent and creative writing prompts or improvements.
  - **Priority**: High. It is a core feature to support smart content creation.
- **Stimulus/Response Sequences**
  - **Stimulus**: The user starts writing or editing a content.
  - **Response**: The system sends the content to the Gemini API, where it receives suggestions tailored to the context that are subsequently displayed in the UI for the user to accept or reject.
- **Functional Requirements**
  - **REQ-1**: The system should accept any real-time user input.
  - **REQ-2**: The contents must be sent securely to the Gemini API.

- **REQ-3**: The AI-generated suggestions will be shown in contrast to the text.
- **REQ-4**: Users will have the choice to accept or reject suggestions.
- **REQ-5**: The suggestion system should not store any user data after the end of a session.

4.4.2 Intelligent Template Selection

- **Description and Priority**
  - **Description**: This feature offers a variety of smart, pre-designed templates based on the selected categories (e.g., science, business, language arts). These templates help the user to commence e-book creation, which has structured layout parameters.
  - **Priority**: Medium to High. It accelerates the process and enhances the quality of the content.
- **Stimulus/Response Sequences**
  - **Stimulus**: Users select a category and click Choose Template.
  - **Response**: The system filters and displays intelligent templates fitting the selected category.
- **Functional Requirements**
  - **REQ-1**: Enabling users to select a topic or category.
  - **REQ-2**: The system should return a list of relevant templates.
  - **REQ-3**: The selected template should be reflected in the e-book editor.
  - **REQ-4**: Responsive and customize-able template should be available with the system.

4.4.3 Role-Based Access Control

- **Description and Priority**
  - **Description**: The specific privileges by category of users-admins and clients (both educators and students) with functionalities specific to each role. Admin manages the templates, creating user accounts for the clients, while clients create, edit, and view content.
  - **Priority**: High. Controlled accessibility ensures integrity of system.
- **Stimulus/Response Sequences**
  - **Stimulus**: A user logs on to the system.
  - **Response**: The system captures the user's credentials into a suitable dashboard according to the role played.

- **Functional Requirements**
    - **REQ-1**: The Admin Dashboard and Client-specific dashboard should be available to the user.
    - **REQ-2**: Templates and user account maintenance is a prerogative of an Administrator.
    - **REQ-3**: Clients should be able to create and manage their own e-books.
    - **REQ-4**: Unauthorized acts should be restricted by the system.

4.4.4 Content Saving Feature

- **Description and Priority**
    - **Description**: Users can save their progress in creating or editing e-books, so they never lose the content and can continue whenever they want.
    - **Priority**: High; it prevents loss of data and improves usability.
- **Stimulus/Response Sequences**
    - **Stimulus**: A user clicks on the "Save" command while working on a document.
    - **Response**: The system will save the current content to MongoDB associated with the user ID.
- **Functional Requirements**
    - **REQ-1**: The content must be saved with timestamps.
    - **REQ-2**: The system must save content under the legitimate user account.
    - **REQ-3**: A message should pop up confirming the save.

4.5 Other Nonfunctional Requirements

4.5.1 Performance requirements

The Smart Skill E-Book Maker will be entirely compatible with at least 200 users concurrently, without experiencing any even slight delays. The average response time would be kept below three seconds, even during peak hours. The AI-based suggestions (through Gemini API) will give quick intelligent content recommendations, while highly efficient backend handling through Node.js and MongoDB will bring about the least delay while loading or saving the user-generated content.

4.5.2 Safety Requirements

Loss of data in case of an unexpected shutdown or failure is prevented by the system. Regular backups of the database will also be scheduled on the server to secure all the user data, including

AI suggestions and e-book content. Also, the mobile app and the backend services do it that way to have less crash incidence and improve their reliability.

4.5.3 Security Requirements

Security is vital when dealing with user content and personal information. The system employs RBAC to restrict user access depending on the role they play (Admin or Client). All credentials were securely hashed and authenticated by JWT (JSON Web Tokens). The conversation between the app and the app server is over HTTPS, ensuring that no data are leaked during transit. Sensitive information such as user profiles and AI-generated content is stored safely in MongoDB, with limited access to it.

4.5.4 Software Quality Attributes

The system offers a clean, intuitive, and responsive interface for both admin and client users. It was developed in React Native and Expo CLI for cross-platform capability to be scalable for future updates. The architecture is modular in nature, promoting ease of maintenance, debugging, and extending new functionalities such as more templates or AI capabilities. The documentation and coding standard of the system make it easier for others to manage the codebase.

4.5.5 Business Rules

- E-books can only be created, edited, or saved by registered users.
- AI features like content suggestions and template suggestions are open to users with valid login credentials.
- Admin can add or remove templates, manage users, and monitor the activity of the system.
- The system will sign out idle users out of the system automatically after thirty minutes of inactivity for safety and security measures.

METHODOLOGY

The methodology follows an iterative and agile approach that prioritizes the concepts of flexibility, rapid iteration and improvement, and user involvement. The project is intended to not only fulfil the functional and non-functional requirements of the specification but to also take advantage of cutting-edge technologies such as AI, so that the user experience can be enhanced and expanded. Each phase of the project is planned and carried out in a way that allows the product to be, scale, change, iterate, and most importantly-it matches the needs of the learners and educators. Following the methodology, are the main phases or steps which are related to the research and development processes that the project will begin to embrace. Agile Methodology for Smart Skill E-Book Maker System:

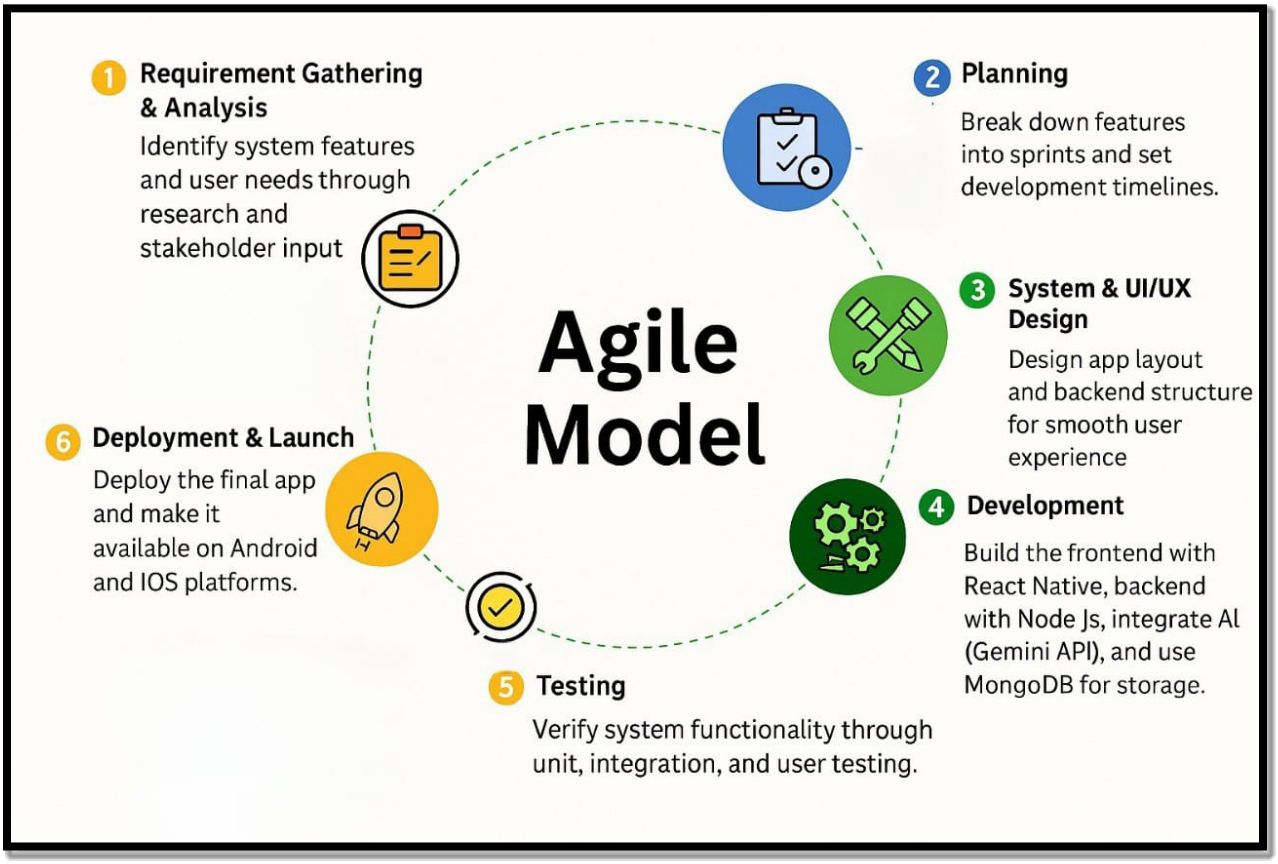

Figure 2: Agile Methodology of Smart Skill Building eBook Maker System

5.1 Preliminary Research and Problem Identification

The first stage is to research the current space of e-book platforms and digital learning technology. The goal is to find out about the shortcomings of current systems in supporting guides that provide interactive, skill-based learning. The information gathered from market analysis, competitive analysis, and consultation with educators and learners prohibits mis-defining and/or scoping the limits of the Smart Skill E-Book Maker System. It proves the problem is well understood and scoped before starting development.

5.2 Stakeholder and User Requirement gathering

During this phase, we gathered the functionality and non-functionality requirements in detail from the important stakeholders such as the teachers, students, and administrators. We use a variety of techniques such as interview, survey questionnaire, and use case analysis. We were mainly concerned about their expectations for the personalizing content creation, editing, ai-generate content suggestions, intelligent templates, and role-user access. This phase ensures that whatever product we deliver will meet the expectations of real users, and address real learning problems.

5.3 Designing the Development Environment

The development environment is created with Expo CLI for the front end. It simplifies the development and testing of mobile development with React Native. Node.js and Express.js were selected as the backend choice because of their performance and scalability. MongoDB is used as a NoSQL database to provide flexibility, speed, and efficiency for data storage. The combination described above allows for rapid API integration, development cycle speed, and cross-platform development. The project will use Git and GitHub for version control.

5.4 System Architecture and UI/UX Design

Overall architecture is planned using a modular, API-driven architecture. The frontend, backend, and database are logically separated for improved long-term scalability and maintainability. UI/UX design will take place in software systems such as using Figma, where wireframes and prototypes are created based on principles of usability. The designs create a clean interface with easy navigation and accessibility for all users, but focus on educators and learners.

5.5 Agile Model Planning and Sprint Division

The Agile methodology allows you to make progress by breaking the work into iterations called sprints, each sprint can focus on specific modules such as user authentication, content editing, or AI. There are daily standups, sprint reviews and retrospectives to ensure that you can monitor how everything is coming together, adaptability, and proper risk management and engagement. The Agile model advances project development in a flexible manner, encouraging adaptability and opportunities for collaboration as development progresses.

5.6 Frontend Development with React Native and Expo

Utilizing React Native and Expo, the user interface of the mobile application is developed. The individual components developed are login/signup, e-book reading, AI suggestions, and content creation. With Expo's live reloading and development tools, rapid iteration for testing and deploying functionality is achievable. During development, the interface is tested in various screen sizes, as well as devices, to assist in maintaining a consistent and responsive user experience.

5.7 Backend Development using Node.js and MongoDB

The backend was developed using Node.js, with help from Express.js to develop RESTful APIs to complete user operations, manage content, handle authentication, and provide AI requests. MongoDB is used with Mongoose library for flexible and scalable schemas for users, content, and templates. Security practices like authentication tokens (JWT) and data validation are used to assist with completing operations securely.

5.8 AI Integration using Gemini API for Smart Features

One of the key functions in the system is to connect to Gemini API so that we can have AI-based content suggestions, personalized content recommendations and intelligent templates. The backend of the system uses API calls to send user-generated user response inputs to Gemini, which, in turn provides us with optimized content suggestions. This process enables learning by making content generation smarter, quicker and relevant to individualized user's needs.

5.9 Functional Testing and Debugging

Upon developing the system there is an extensive process of functional tests that is carried out to ensure that all components are working as they should work. This includes unit testing individual modules, integration testing between the frontend and backend, and user acceptance testing in front

of a small group of users. API tests would be conducted using Postman, whereas mobile debugging would be done using some of the testing features that Expo provides. In this phase, bugs and usability issues are solved through multiple refinement cycles.

5.10 Final Deployment and User Acceptance Testing

After being internally tested, the application is now ready to be deployed. The backend is deployed to cloud services such as Render or Heroku, while the mobile application is packaged up using Expo's build service for Android and iOS devices. Once the application is ready to be pushed out to users, the final phase of user acceptance testing (UAT) is done to confirm that the system is ready for use. Feedback is then collected and minor changes are made as required.

5.11 Documentation, Reporting, and Maintenance Plan

Complete documentation is completed, covering the system architecture and design, API references, database schema, installation/use instructions, and a final project report that outlines each stage of development, methodologies used, test results, and deployment strategy. A basic maintenance plan is established describing how updates, bug fixes, and enhancements will be managed in the future.

5.12 Tooling, Technologies and Development Stack

These tools and technologies were selected based on their contemporary technology, development community, and integration ensuring development is performed quickly and can be scaled.

- **Frontend**: React Native (via Expo CLI) for cross-platform mobile UI development.
- **Backend**: Node.js + Express.js to implement server logic and RESTful APIs.
- **Database**: MongoDB to perform flexible and scalable NoSQL storage.
- **AI Integration**: Gemini API for intelligent content recommendations and personalization.
- **UI Design**: Figma for prototyping and visual design.
- **Testing Software**: Postman (API testing), Expo Go (mobile preview).
- **Version Control**: Git and GitHub - to manage code and work collaboratively.

5.13 Hardware and Software Requirements

5.13.1 Hardware Requirements

- **Developer Machine**: At least 8GB of RAM, an i5 Processor and above, and an SSD.
- **Mobile Device**: Android (v8.0+) or iOS (v13.0+) to test the Expo app.

5.13.2 Software Requirements

- **Code Editor**: Visual Studio Code
- **Runtime Environment**: Node.js (v18+)
- **Package Manager**: npm
- **Expo CLI**: To run and build the React Native app
- **MongoDB**: Cloud-based or local
- **Postman**: For API testing
- **Git**: For version control (including GitHub)

Chapter 6

DETAILED DESIGN AND ARCHITECTURE

This chapter introduces the overall high-level system structure and design of the Smart Skill Building eBook Maker System, which will enable users to generate customized skill-based digital eBooks using a mobile application. An architecture with well-defined definitions is required to ensure scalability, flexibility, and system reliability. This chapter provides the high-level system architecture, main components, and their interactions, along with database design and flow diagrams. Each module has been designed to satisfy the functional and non-functional requirements defined in the previous phases of development. Implementation using advanced technologies like React Native, Node.js, and MongoDB has enabled modular and efficient system implementation.

6.1 System Architecture

The architecture of the Smart Skill Building eBook Maker System has been created with modularity, scalability, and maintainability with seamless interconnectivity among functionalities. The system will be capable of user registration, login, selecting or creating skill category, and creation of personalized eBooks with enriched learning content. For the effective completion of these functional specifications, the system has been segmented into three main subsystems: Frontend (Mobile App), Backend (Server API), and Database (MongoDB).

Major Functions of the System

The system is tasked with the following high-level tasks

- Handling user and role authentication (Admin, Client).
- Facilitating users to create, modify, and organize e-books using templates.
- Safe storage and retrieval of e-book content and user data.
- Providing AI-based recommendations for content improvement.
- Facilitating sharing, critique, and collaborative work among consumers Offering persistence and data synchronization.

Top-Level System Decomposition

The system was comprised of the following major parts:

1. Frontend

- Handles user interaction and presentation logic.
- Offers the user interface for creating, editing, and reading e-books.
- Backend API communication to send and receive data.

2. Backend

- Used as the central focus of application logic.
- Performs CRUD operations API endpoints.
- Sanitizes user input and provides role-based access control.
- Manages user sessions and authentication tokens (JWT).

3. Database

- Stores user, book, template, and feedback information in structured as well as unstructured forms.
- Enables flexible document-based storage for dynamic e-book structuring.

4. Authentication Module

- Supports secure user sign up, login, logout, and session management using JSON Web Tokens (JWT)

5. Cloud Storage

- Scheduled integration with Firebase or other file storage solutions to store big files like images, audio files, or optimized templates.

Interaction Between Components

- The frontend talks to the backend using RESTful API calls.
- The logic of backend processes and MongoDB database interaction for data storage.
- JWT authentication will only display or edit content to authenticated users.
- AI modules (under development) will provide intelligent content suggestions, potentially via third-party API.

Rationale For Architectural Decomposition

This tiered structure was adopted based on the following reasoning:

- **Separation of concerns**: Each layer maintains a specific set of responsibilities, but they are independent to execute and verify.
- **Scalability**: The system can accommodate features like AI tools, offline functionality, and third-party plugins without altering core logic.
- **Cross-platform compatibility**: React Native supports development on both Android and iOS from a shared codebase.
- **Schema flexibility**: MongoDB's NoSQL structure accommodates dynamic content like user-defined templates and e-books that can be customized.
- **Maintainability**: Modular design enables better debugging, upgrading, and future expansion.

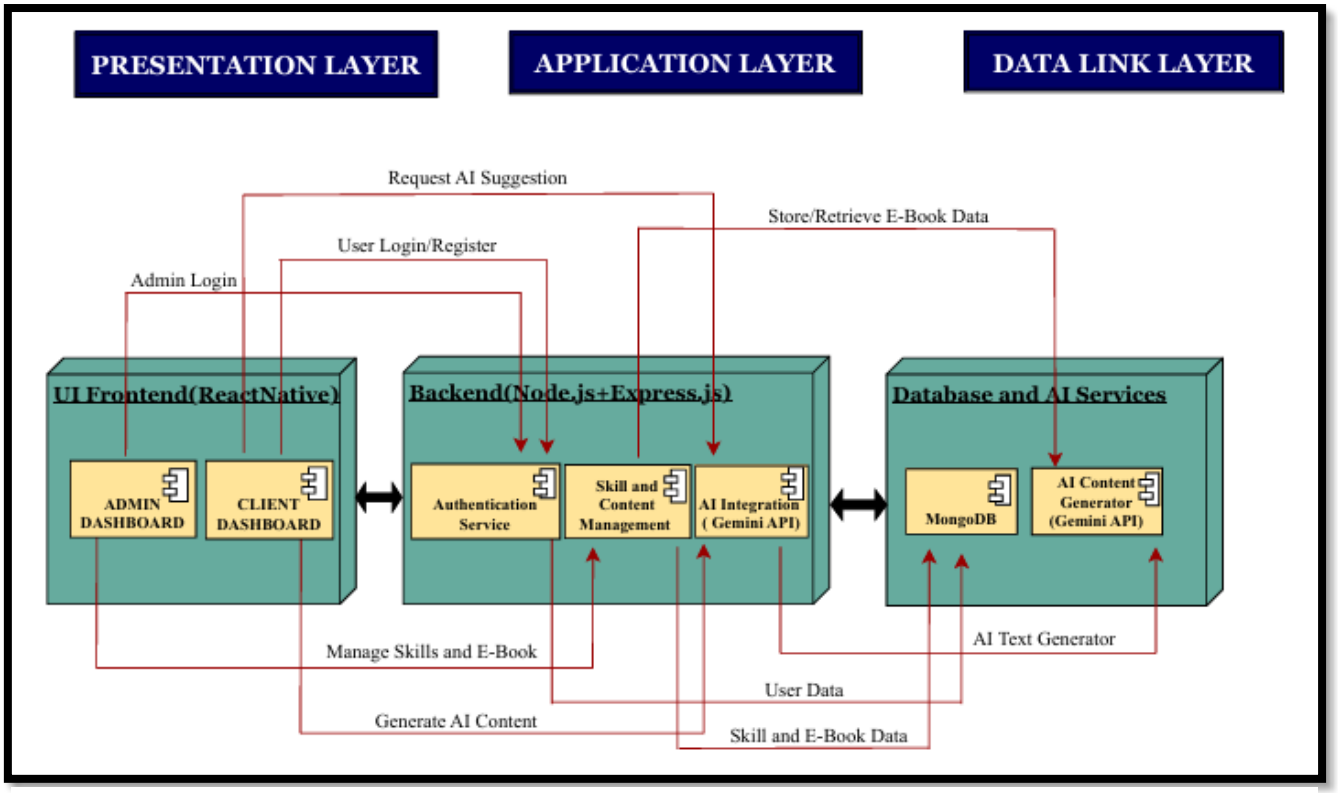

Figure 3:Architecture Diagram

6.1.1 Architecture Design Approach

The Smart Skill Building eBook Maker System is implemented with a three-tiered modular structure based on the Three-Tier Model. This type of structured support system understandability, extensibility and maintainability by distributing work to three different layers:

- **Presentation Layer** (Frontend): Implemented in React Native, it manages the user interface and interactions, communicating with backend services through RESTful APIs.

- **Business Logic Layer** (Backend): Developed with Node.js and Express.js, this layer handles request processing, application logic, authentication, session control, and integration with AI features such as content assistance.
- **Data Layer**: Utilizes MongoDB for storing user data and eBook content, as well as supporting real-time synchronization where needed.

The system architecture follows a service-oriented approach, where features like user authentication, eBook operations, and AI functionalities are modular services interacting via APIs. This promotes reusability, maintainability, and scalability. This approach allows efficient development, better integration of external services, and easier future upgrades. A monolithic structure was avoided to ensure better separation of concerns, support for parallel development, and readiness for future system expansion. Key technologies were chosen based on their flexibility and performance:

- MongoDB for its adaptable schema and real-time synchronization features.
- AI APIs (Gemini) for intelligent suggestions and grammar correction.

6.1.2 Architecture Design

The Smart Skill Building eBook Maker System is divided into three main layers UI Frontend, Backend Server, and Database & AI Services each containing a set of highly cohesive services. Such a service-based, modular architecture allows separation of concerns, ease of maintenance, and makes every layer independent to develop.

- **UI Frontend** (React Native)
    - **Admin Interface**: Manages user accounts, monitors eBook content, and monitors system usage.
    - **User Interface**: where users are allowed to register, login, create and view eBooks, and request AI-generated for content suggestion.
- **Backend Server** (Node.js + Express)
    - **Authentication Service**: Secures user login, JWT management, and access control.
    - **E-Book Management**: Responsible for all create/read/update/delete eBook and user progress operations.

- **AI Integration Module**: Talks to Gemini API to produce and provide AI-powered text to the eBook process.

- **Database & AI Services**
  - **MongoDB**: Serves as the unified datastore for user profiles, eBook documents, and progress tracking in a schema-agile form.
  - **Gemini API**: Offers content-generation features, which produce AI-generated text based on user input.

6.1.3 Subsystem Architecture

In order to support design principles of modularity, scalability, and separation of concerns, the Smart Skill Building eBook Maker System is partitioned into separate subsystems. These subsystems interact with one another in order to share basic application functions such as AI recommendations, content creation, storage, and authentication.

- **Authentication Subsystem**: Manages user login, registration, and secure session management with JWT. It validates the credentials and manages the access according to user roles.
- **eBook Management Subsystem**: Enables users to add, edit, read, and delete eBooks. Supports text, image, and template input, with all of them being saved in MongoDB.
- **AI Integration Subsystem**: Facilitates intelligent content suggestions by forwarding prompts to the Gemini API and then to the user interface with AI-driven text.
- **Admin Subsystem:** Provides admin-level access for user management, system activity reporting, and eBook category management through one-of-a-kind dashboard.
- **Data Persistence Subsystem** (MongoDB): Serves as the hub for all the data, performing CRUD on all the users, eBooks, templates, and feedback via AI.

6.2 Detailed System Design

The Smart Skill Building eBook Maker System is designed with modularity so that each module of software can be tested, upgraded, and maintained independently. This chapter deconstructs the system into individual software modules and submodules and describes their in-depth design and their position within the overall framework. Each of the following components is mapped to the above-layered structure (Frontend, Backend, Database, and External Services).

6.2.1 Classification

The Smart Skill Building eBook Maker System is composed of multiple software components, each classified based on its function and role within the system architecture. Below is the classification of the key components.

Table 1:Classification and Description

Component	Classification	Description
Frontend (React Native App)	Subsystem	Responsible for managing user interactions and the presentation layer of mobile platforms.
Authentication Module	Module	Provides login, registration, and JWT-based session management.
eBook Editor Component	Component (Frontend)	UI element for users to write, read, and revise eBook material.
AI Content Generator Module	Module (Backend Service)	Integrates with Gemini API to enrich or generate content using AI.
User-Management Module	Module (Backend)	Allows admin to manage user access rights, roles, and profiles.
Node.js Server	Subsystem	Core backend for business logic and API routing.
MongoDB Database	Subsystem	Saves user data, eBooks, templates, and user history of progress in document-based mode.

6.2.2 Definition

Each software component in the Smart Skill Building eBook Maker System has a specific purpose and contributes to meeting the system's functional and non-functional requirements.

- **Frontend** (React Native): Handles presentation and user interaction. Provides screens and UI components for user registration, login, skill selection, eBook creation, and editing content. Communicates with the backend via RESTful API calls.

- **Authentication Module**: Manages secure user login, registration, logout, and role-based access control. Manages session management through JSON Web Tokens (JWT) so that only Admins or Clients can access specific functionalities.
- **E-Book Editor Component**: Enables users to create, design, and style their own eBooks. Supports templates, inserting media/text, and features AI-suggested content to enhance learning outcomes.
- **AI Content Generation Module**: Incorporates third-party APIs (e.g., Gemini API) to generate smart content. Generates appropriate AI-created content to be inserted in eBooks based on users' requests (e.g., "Write an introduction about Python").
- **User Management Module**: Used by Admins to track, moderate, and manage users. Allows viewing of user information, access control, and eBook activity monitoring across the site.
- **Backend Server** (Node.js + Express): Performs application logic, processes API calls, processes business rules, interacts with the database, and interacts with the AI content generation module.
- **Database** (MongoDB): Stores all dynamic information like user profiles, eBooks, templates, and comments. A NoSQL document-oriented database that offers flexibility in storing structured data and unstructured data.
- **API Route Files**: Implement RESTful API endpoints for client-server interaction. Manage activities such as login, eBook creation, template retrieval, and submission of AI prompts through properly defined routing.
- **Mongoose Models**: Create MongoDB schemas such as (User, Book, Template). Support uniform structure and validation on stored data to enable structured and uniform processing of data.
- **Middleware Components**: Functions (AUTH Middleware) that authenticate JWT tokens, verify user roles, and protect restricted routes. Only authorized and authenticated users should be permitted to use system resources.

6.2.3 Responsibilities

- **Frontend** (React Native)
    - Demonstrates the user interface for registration, login, and eBook creation.
    - Collects user input and sends requests to the backend via API.
    - Displays results that are offered by the backend (i.e., saved e-Books, AI suggestions).

- Supports interactive operations such as choosing skills, applying templates, and text editing.

- **Authentication Module**
  - Authenticates users on login and registration.
  - Manages and assigns user roles (Admin or Client).
  - Has secure sessions via JWT.
  - Provides access control by limiting unauthorized users from sensitive actions.

- **E-Book Editor Component**
  - Facilitates eBooks creation, editing, and formatting.
  - Supports multimedia insertion and employs design templates.
  - Accepts user input for content generation with AI.
  - Saves changes in real time or upon user action.

- **AI Content Generation Module**
  - Processes user inputs and requests smart suggestions or content via Gemini API.
  - Returns context-sensitive text (e.g., introductions, summaries, definitions).
  - Enhances eBooks by providing intelligent writing advice.

- **User Management Module**
  - Enables Admins to monitor system usage and user activity.
  - Enables moderation of users, i.e., banning or admitting them.
  - Tracks user eBook draft and edit history for approval by administrator.

- **Backend Server** (Node.js + Express)
  - Responsible for routing APIs and business logic execution.
  - Manages CRUD operations for eBooks, users, and templates.
  - Binds AI services and manages user request flow.
  - Acts as the intermediate layer between frontend and database.

- **Database** (MongoDB)
  - Stores permanent data like user accounts, eBooks, and templates.
  - It accommodates flexible schema for supporting custom eBook layouts.
  - Manages backend requests for data storage and retrieval.

- **API Route Files**
    - Describe frontend requests being routed to backend controllers.
    - Logic separation by purpose (for example, user routes, book routes).
    - Standardize code structure and maintain RESTful principles.
- **Mongoose Models**
    - Define data structure for each document type (User, Book).
    - Validate data before saving to MongoDB.
    - Guarantee the accuracy and consistency of data stored.
- **Middleware Components**
    - Intercept HTTP requests to validate user sessions.
    - Validate required roles before granting access to secure routes.
    - Deal with errors and provide security throughout the API lifecycle.

6.2.4 Constraints

The Smart Skill Building E-book Maker System comprises different potential limitations and assumptions that facilitate smooth operational performance and user experience. For frontend, the user must be logged into the system in order to access features. The frontend relies on the backend and the backend relies on API correctly returning and formatting data (JSON is the primary format). The way authentication in the system, utilizes JWT tokens, must be valid, unexpired for the user to access secured sections of the system. If the user does not have a valid, existing JWT token they will not be able to access the sections of the system secured by JWT authorization. As it relates to the PDF Editor eBook creation piece, limitations on the eBook creator exist (file size, formatting) and a strong internet connection is vital, especially in regards to real-time saving, but also when utilizing AI. The use of AI-powered content utilizing the Gemini API, expect no restrictions except for the limits based on the requested size, and service limits, and must have an internet connection. There is admin-only features that are limited (user/account management) to help mitigate performance based on storage usage and number of database queries.

As it relates to the backend, since the system runs on a Node.js based server the server must be able to support multiple users at once, provide timely responses, and check all endpoint requests included but not limited to checking security restrictions. The Document database provider (MongoDB) places a 16MB document restriction on each document and uses strict scripted data

formats using Mongoose schemas. Querying data efficiently is also necessary for the system not to run slushy. Finally, as it relates to the API and left off from above the API routes.

6.2.5 Composition

The Smart Skill Building E-book Maker System includes some key subcomponents that work together to deliver its unique functionality. In the front end, authentication module, eBook editor module, and AI content module are integrated together with the React Native interface. All of the subcomponents work harmoniously to ensure that every request pathway has to pass through validation, logic processing and storage areas in order for each request to be handled properly. Each of the subcomponents has a function and role:

- The authentication module has the job of handling user login and permissions.
- The eBook editor provides tools for creating content and creating contracts.
- The AI module connects the user to information from the external services, such as the Gemini API, to provide smart content options.
- On the back end, the server built on Node.js is organized into express routes that connect to and from API route files, middleware, and controller functions.
- The API route files establish the communication between the front and back end; middleware provide security and validation (i.e. validates the user JWT).
- At the data level, the mongoose models for information storage (i.e. user, eBooks, templates) and MongoDB stores all of the users' content, data from feedback, and templates.

6.2.6 Uses/Interactions

The Smart Skill Building E-book Maker System provides a platform that connects users to a system of eBooks. Users first access the system by signing up or logging in. Once authenticated, the user is afforded the choice to either read existing eBooks or write one based on that choice the application can tailor the screen and features being deployed to the user.

The Smart Skill Building E-book Maker System keeps it simple in the Writer user option providing users two methods for getting started either selecting preformulated templates and organized by skill topics or creating new content via the embedded AI assistant. The AI assistant is helpful in suggesting titles, topics or content ideas, and even generating entire topics based on

prompts. This feature alleviates any apprehension around writing a structured and skill-based eBook, if a user has not written anything previously. Users selecting the reader option can engage with many users generated or AI-generated eBooks organized by categories. Users can further build their knowledge using content based on skill topics they specify in this option.

The Smart Skill Building E-book Maker System relies on the user interaction with all the frontend components of (React Native UI, navigation, and input screens) to access services from backend components of (Node.js APIs and MongoDB database). Other users of the system include Administrators or the Developers themselves managed through tools also found in the System to view and user engagement with the system, manage their content and maintain system rules. The system is designed to motivate both learning and creativity.

6.2.7 Resources

Smart Skill Building E-book Maker System is dependent on multiple software tools, libraries, platforms, and development sources, for design to function smoothly, intelligent features and user functionality to exist and create workable formats. These resources will also allow for client-side and server-side operation, as well as allow for testing, development and deployment workflows.

- **Frontend Tools**: The frontend of the app is all done in React Native with Expo CLI, which allows for the cross-platform mobile capabilities to be integrated. The app is deployed and simulated using Expo Go, which allows for a smooth capability of being able to preview the app through Expo Go by its intended features.
- **Design Resources**: Figma is used to design the user interface for the app and create a consistent visual layout for all screens, but also allows for collaborative design and rapid prototypes.
- **Back-end Development**: The back-end uses Node.js and Express.js, where all business logic, RESTful API routes, authentication and database interactions take place.
- **Database**: MongoDB is a NoSQL database to help facilitate storing all dynamic data types (user accounts, templates, feedback, eBook content, etc.). Mongoose is used to establish schema types and enforce schema structures to help organized data.
- **Authentication**: The system creates a session using JSON Web Tokens (JWT) to allow for sessions to be stored safely and securely. Each time access is granted to protected routes, it is determined by role-based access to authenticated users only.

- **AI Integration**: The application is connected to the Gemini API for AI content generation. This connection allows users to generate eBooks as well as to receive suggestions to improve the user's content using AI.
- **Testing and API tools**: Postman is used to test the RESTful API endpoints and to validate the data is reliably returning and that the response from the backend is not getting jumbled during development in regards to the in-range and out-of-range responses.
- **Software Libraries**
    - **React Navigation**: It's used for managing multiple screen navigation in the application, which is to manage routing to different screens in a react native app.
    - **Axios or Fetch API**: It's used for holding HTTP requests and HTTP responses that are pushed to the backend to send data from the frontend and get data back to the frontend in real time.
    - **Express Middleware**: Authentication checks, requests being parsed and error handling on the server side of things, all fall under express middleware.
    - **Mongoose**: Schemas and validation fall under collections in MongoDB.
    - **Jsonwebtoken**: Used to create (or sign) and then verify JWT's for the user during their authentication and authorization.

6.2.8 Processing

The collaborative E-book authoring system is indeed called Smart Skill Building E-book Maker System. Through its interactivities and system functions, it uses front-end input combined with back-end logic, database function, and AI. When users log in or register for accounts, the system uses JWT tokens for authenticating users, leading to a secure session-based access. The application hence, changes its functionalities and features based on the user roles (Reader or Writer).

When the writers are on their eBook writing experience, they have a clear option to start with an empty document or even a template or, as the case may be, use AI-generated content through the Gemini API. Herein, user input (with some extra information) is created and sent to the back-end. The back-end is interlinked with Gemini, and the organized contents will be reverted to our app. The response is achieved asynchronously and entirely dependent upon Gemini API response time with respect to the user. While performing actions like saving, editing, deleting, etc., the system will fire the appropriate route of the RESTful API that will perform the CRUD operations via MongoDB and Mongoose functionality (like data validation, schema enforcement, etc.).

The state transitions of the system are managed in such a way that it can navigate and switch states (edit/view modes, in-content submission, etc.) through state management in React Native. Activities performed in real time (such as automatically saving the content) are subject to the availability of the Internet connection and to the actions taken simultaneously by the concurrent users. Writing is made limited to a reasonable readable size to maximize spatial complexity and is also distributed onto the cloud or put under conditions to be accessed publicly.

6.2.9 Interface/Exports

The back-end of the Smart Skill Building E-book Maker System exposes services and resources for use. These services and resources are exposed under clearly defined exports that include RESTful API end points, data models (schemas), middleware functions, and configuration constants. Combined, these exports are used as core functions, specifically, user registration and log in, e-book creation, organization of folders and templates, and access control.

The User Authentication Service is an exported service that has significant function in the service resources. This service allows users to register securely and to login as authenticated users. The User Authentication Service defines a collection of functions that verifies user input, hashes passwords using bcrypt.js, issues authentication token using Jsonwebtoken, and issues responses. The User Authentication Service is utilized via the /Api/user/register and /Api/user/login endpoints. This component acts as a subroutine/API and includes distinct functions performing many important functions to verify users as registered users and perform access control on a user's actions in the system. This User Authentication Service is limited in that it only allows a single email address at registration and secure password storage. Another exported service is the Book Management Service that allows users a full set of CRUDS (Create, Read, Update, Delete) abilities.

6.2.10 Detailed Subsystem Design

The Smart Skill Building eBook Maker System comprises several connected subsystems that work together to create a seamless user experience for producing AI-powered eBooks. Each subsystem, from user authentication and eBook information management to AI-driven content generation, was built to be modular, maintainable, and efficient. Our backend utilizes Node.js to access a MongoDB database for persistent data management of user profile data, eBooks, feedback on content, etc. The system was developed with RESTful APIs and every endpoint is associated with a service operation referenced in the architectural diagram.

The frontend was developed with a React Native shell and invokes these APIs to deliver real-time features such as login, skill selection, eBook edit actions, and AI input and suggestions. Each middleware fulfills authorization, input validation, and error handling requirements. AI does interface with the Gemini API to assist in providing intelligent suggestions based on user prompts.

Every component and module have the separation of responsibilities. For example, the authentication module only invokes actions for logging in and registering users, and the AI module only manages external communication between the eBook Maker and the Gemini API. Some activities in the system have considerable unique behaviors. We handle these by fully defining meaningful states in the lifecycle of an eBook whereby we represent its state with state machine logic in the application. There eBook states represent behaviors (or transitions) such as: draft to final, or everyone was given topic: i.e. AI could represent the provide AI quote for an indirect eBook parameter called: skill.

Use case Diagram

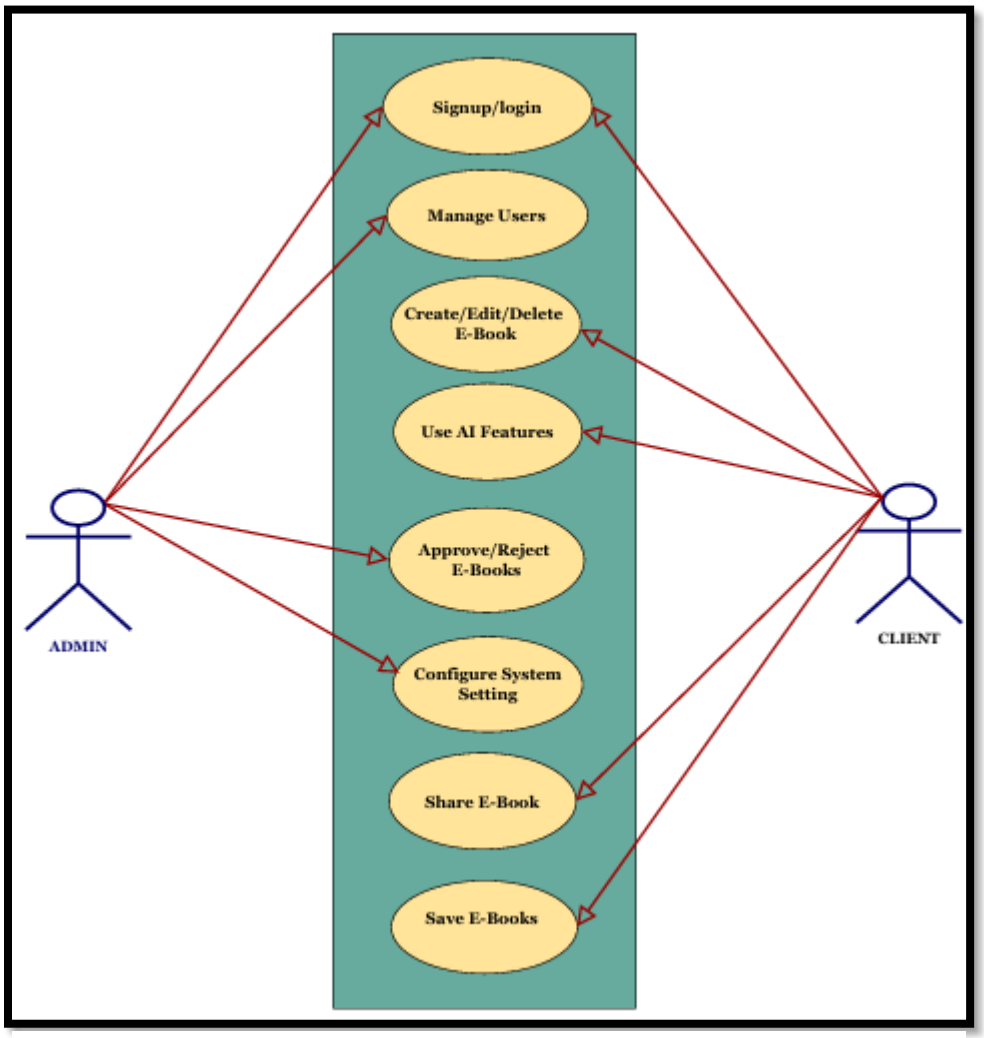

Figure 4:Use Case Diagram of Smart eBook

The Smart Skill E-book Maker System accommodates two actors which mostly include Client and Admin who take specific parts in the system. A client can register or login via the platform to create and edit e-books through a user-friendly interface. While creating their contents, clients can get AI suggestion to provide extra quality and structure in their writing. They are also able to save their e-books for some time for further editing or reading. On the other way, the administrator manages the users and reviews the submitted e-books, either approving or rejecting the e-books based on their standards. Besides that, Admins configure the system settings and oversee the use of templates and AI tools. Such a structure allows users to concentrate on the content and be assured that administrations maintain control over quality and other platform workings.

Activity Diagram

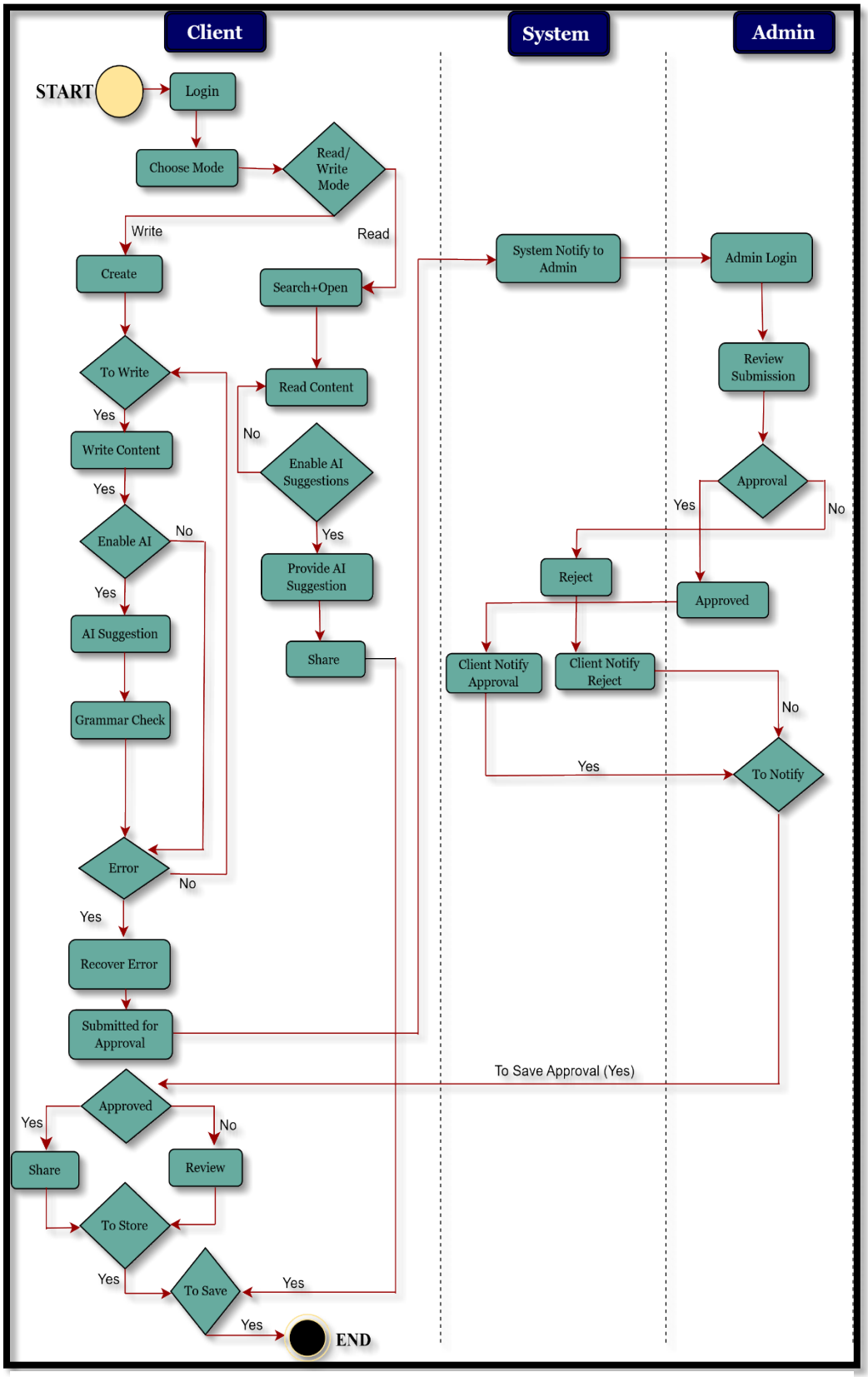

Figure 5:Activity Diagram of Smart eBook

The activity diagram presents the user logging into the application and providing the choice between Read Mode and Write Mode. If Write Mode is selected, the user can start creating new e-book contents. While writing, the users can be provided with the choice of enabling AI suggestions for grammar, structure, or content flow improvement. They can have the functionality of grammar checks at any time, and the error recoveries can be done for any unexpected problems that may appear. At the time of completion of the e-book, the user submits the content for approval. This triggers a notification from the system to the admin for reviewing the contents submitted. The admin approves the e-book after a successful review and notifies the user of the approval. The e-book, therefore, can be saved or shared according to the functionality of the particular application. In Read Mode, a user can read by browsing, searching for, and opening different previously established e-books. All this creates an excellent experience for the user by intelligent content creation, validation by admin, and real-time interaction while keeping a clear demarcation between reading and writing workflows.

Sequence Diagram

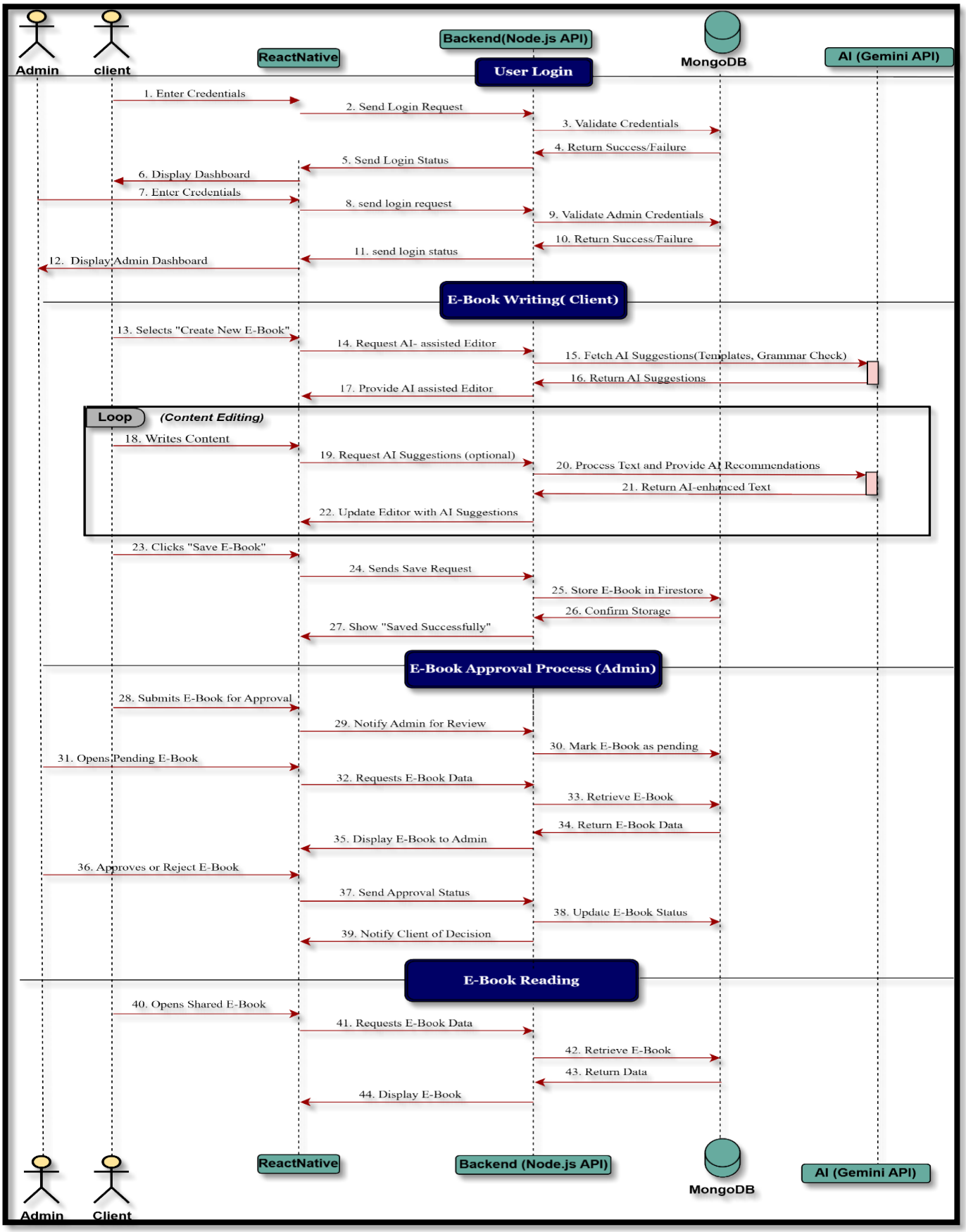

Figure 6:Sequence Diagram of Smart eBook

The following displays a sequence diagram that shows the end-to-end interaction flow between all involved parties, namely the client, admin, frontend (React Native), backend (Node.js), and MongoDB, with AI (Gemini API). The echoing process starts from user login sequence, through which the client or admin enters credentials. The backend validates these credentials and stores them in MongoDB, after which the appropriate dashboard is displayed (client or admin). Once logged in, if a client clicks on "Create New E-Book", it sends a request to access the AI-assisted editor. The system then fetches suggestions such as grammar enhancements or template recommendations from the Gemini API and displays them in the writing interface. The client writes the content and can optionally seek AI support during editing. When the user has finished, he or she clicks "Save E-Book" and the content is saved in MongoDB, along with a success message.

For admin approval, the e-book submitted by the client is marked pending by back-end processing and notification to the admin. The administrator retrieves and reviews the content. After the review, the content is approved by the admin (in this case, there is no rejection in this flow), and the backend updates the status of the e-book's approval and notifies the client.

In read mode, users can open shared e-books by sending a request to retrieve stored content from the database, then display it in the app. This whole sequence is constructed to assure all kinds of user roles smooth interaction, real-time AI assistance, and safety while handling the content on the platform.

Component Diagram

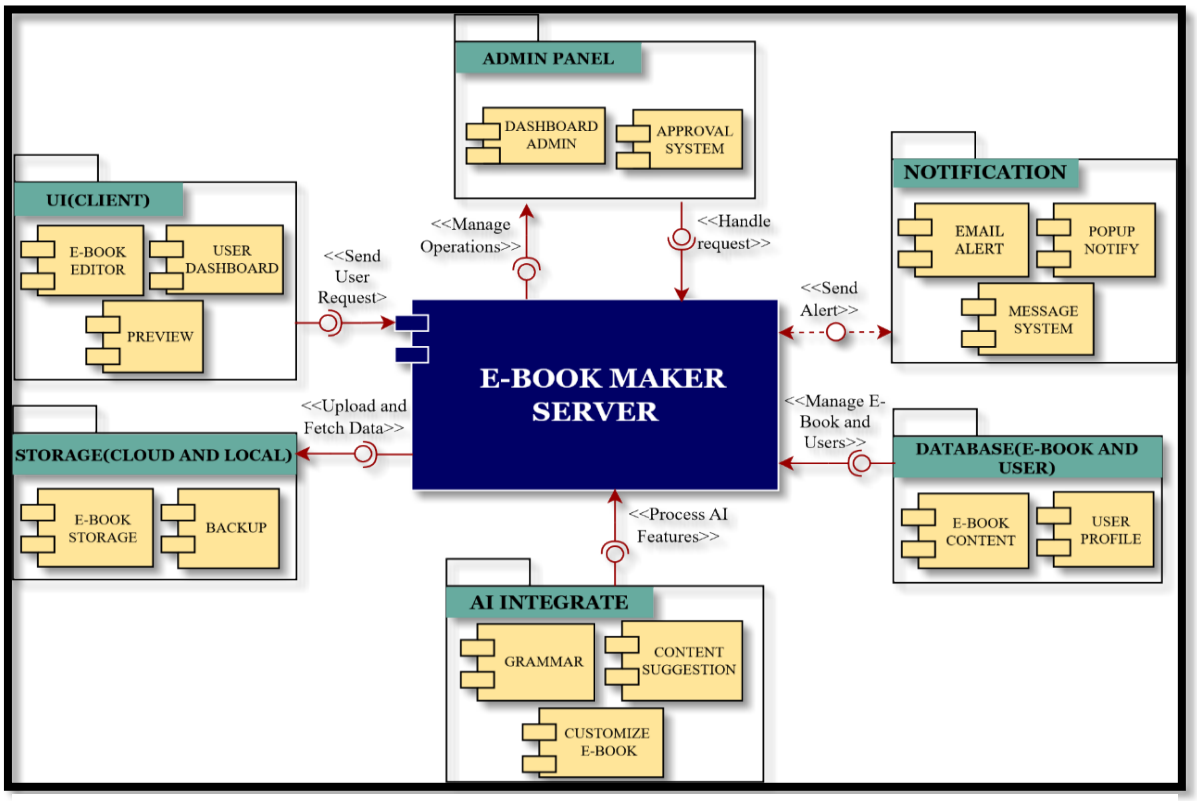

Figure 7:Component Diagram of Smart eBook

The Component diagram for the Smart Skill Building E-book Maker Systems gives the modular design of the software application-the way in which their parts cooperate in delivering the user and admin experience. At the end-user side, the core houses the user dashboard which acts as the nerve center linking up to other essential features in addition to the E-book Editor, Customize E-book, Preview, and User Profile. The E-book Editor is mostly used in writing, typing and formatting their E-books and could design it through AI-supported modules in grammar correction or smart suggestions. Users may modify their E-books using templates and formats, and preview the final design before saving it. Notifications are conveyed through email alerts, pop-up notifications in the app, and client UI notifications. The system will update users when their submissions are approved, revised, or provide feedback notifications. The backend serves as a great storage segment for cloud and local back up, so storage both in terms of e-books and users will be all housed under one central database. Admin functionalities are a completely different process using the Admin Dashboard, which would oversee user activities and also maintain the approval system and governance of communications. The heart and soul of the system are marked as the E-book Maker Server. This server is responsible for coordinating all requests from users to system

modules, data fetch/upload, and working with AI functionality through APIs like Gemini. Scalable data security with assistance from intelligent users makes the system efficient.

State Machine

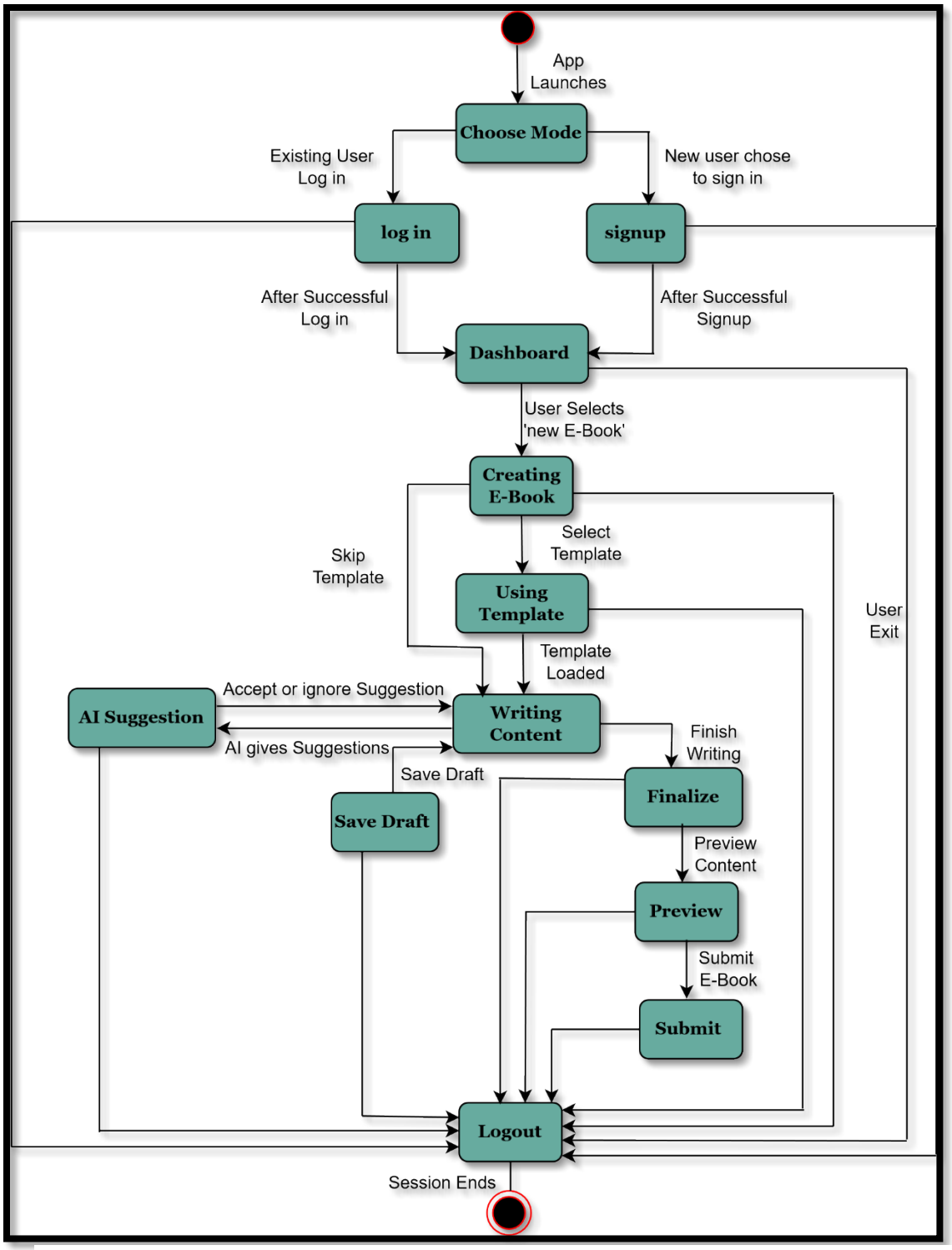

Figure 8:State Machine Diagram of Smart eBook

The State Machine Diagram for Smart Skills E-Book Maker System shows the key states that a user will traverse in interaction with the application. It starts from the Choose Mode where the user is either able to Sign Up or Log in. If the user logs in, he gets dropped on the Dashboard. The next option is to create an e-book. Under the Creating E-Book state, it allows users to use a predefined template or jump straight to Writing Content. Users can write and then go to the suggestion state for AI suggestions and then head back to write. Users can save drafts, write them out, and still return to work on them. After writing, users would pass through the Finalize state, then the Preview and Submit states, to end e-book production. They can also log out at any point after they log in. That's why the e-book-making process can be session and stage flexible-end user centric in managing their session for all stages. The guide helps automated content creation with optional AI assistance and session management.

Class Diagram

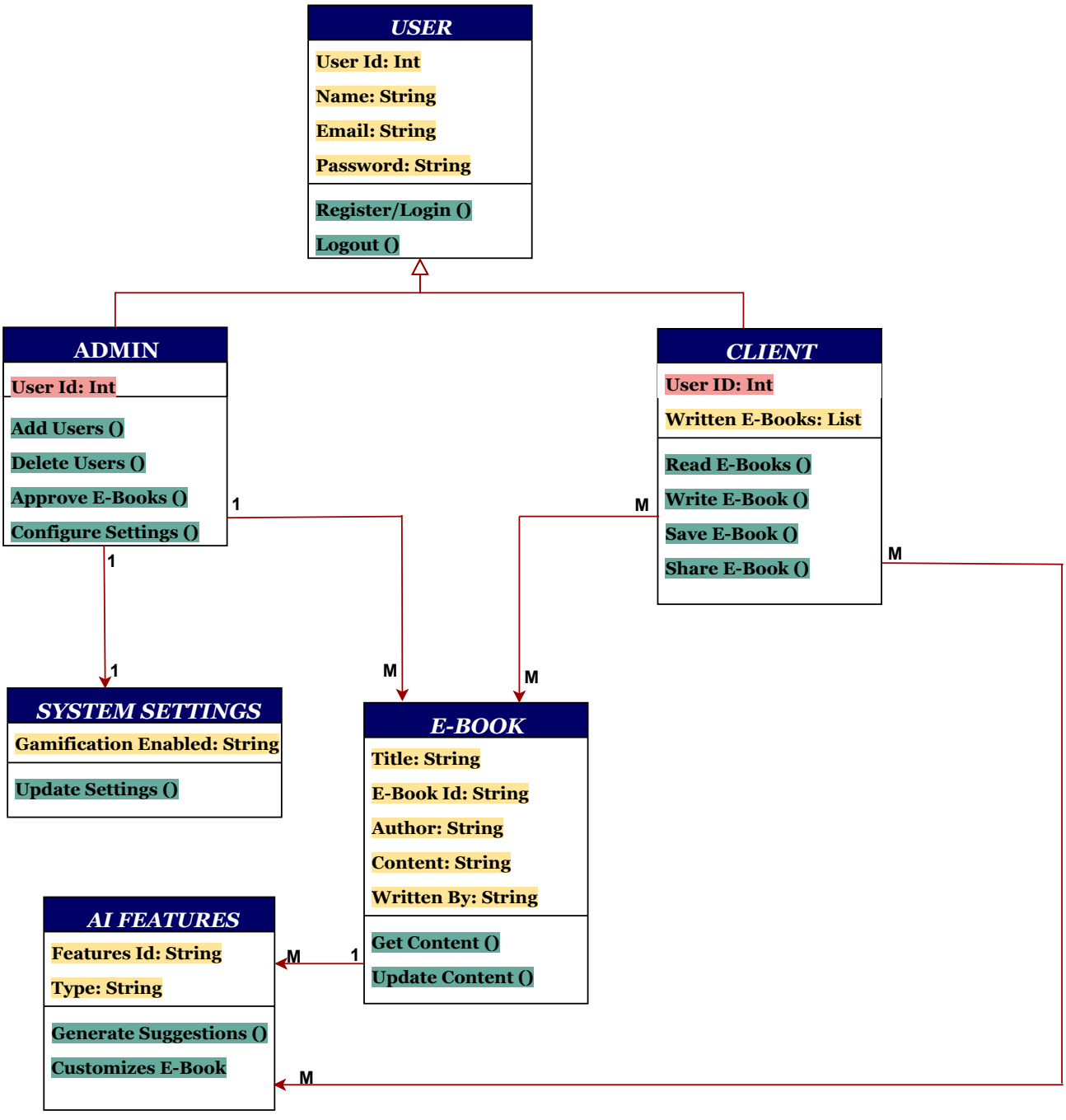

Figure 9:Class Diagram of Smart eBook

The class diagram above represents the high-level structure of the Smart Skill E-book Maker System, showing the major classes involved in the project and their relationships. The User class represents the User aspect of the system and contains the common attributes User Id, Name, Email, and Password, as well as common functions such as Register/Login () and Logout (). This class, which we are calling User, can be extended to create the two role-based classes Admin and Client.

The admin class has all the required functions for managing the system, such as adding deleting users, approving e-books, and setting up the system's settings for General information. The Client class represents the normal user who has all the primary functions to work with e-books. Some functions for Clients could include reading, writing, saving, and sharing e-books. Each Client could also relate to an array of E-books that were written (one to many association relationships) based on the E-Book class.

The E-Book class has important attributes such as Title, E-Book Id, Author, Content, and Written By. There are several operations such as Get Content () and Update Content () to manage the content of e-books. The system also has a class associated to AI Features that provide suggestions and increases the level of customization based on content, so users can use an AI-driven approach to create an E-book. The system has system settings.

Dataflow Diagram

Figure 10:Dataflow Diagram of Smart eBook

The Data Flow Diagram (DFD) for the Smart Skill Building E-Book Maker System shows the interaction of the system major components (the Client/User and the Admin), and how data flows between a client to their work and the various subsystems.

Clients (users) start by registering for the system and logging in as clients. Once logged in, Clients can perform critical tasks, like creating, editing and deleting their e-books. Clients also use the system's AI-enabled functions, which assist users by providing suggestions and improving content creation. The data obtained from this process, including user credentials, e-book content, and AI suggestions etc., is retained and maintained by the system.

On the administrative side of the system, the admin has functions including registering new users, maintaining and managing user data and in essence supervises the entire publication workflow of the content. The admin's role to evaluate, approve or reject any e-books submitted by Clients in order for them to publish them. Admin also manages and is responsible for updating user information, and has access to data that has been processed by the AIs cognitive functions with a view to determining Quality Assessment (QA) or compliance of user generated content.

The DFD also demonstrates how the data flows seamlessly across the system components: the clients interacted with the e-book tools and AI elements of the system and data was updated and stored and processed, with the oversight and other controls managed by the admin.

Chapter 7

IMPLEMENTATION AND TESTING

7.1 Development Methodology

To maximize flexibility, quick iterations, and collaboration in development, the Agile Scrum methodology was adopted. The project was divided into two-week sprints for ongoing planning, developing, testing, and fixing. Some Agile practices we used for this project were:

- **Sprint Planning**: Prioritized the features to work on, such as e-book creation, and AI suggestions about your content, and document upload.
- **Daily Stand Ups**: Recognizing progress and roadblocks.
- **Sprint Reviews and Retrospectives**: For continuous improvement post each cycle.

7.2 Tools, Technologies and Techniques

- **Frontend**: React Native with Expo CLI was used, facilitating quick mobile app development across platforms.
- **Backend**: To create a scalable and efficient API layer, Node.js was chosen along with its Express.js framework.
- **Database**: MongoDB stores user data, e-books, and templates with flexible schema support.
- **Storage**: For storing e-books, images, and documents, Firebase was chosen.
- **Authentication**: Role-based authentication was conducted with JWT (JSON Web Tokens).
- **Version Control**: Git hosted on GitHub was the source control tool used by the team for collaboration and code reviews.
- **UI/UX Design**: Utilizing Figma, the UI was created for modern and responsive user-friendly interfaces.

7.3 Core Functionalities

- **Read & Write Mode**: Users can leverage existing templates, or build their own personalized e-books from scratch.
- **AI Suggestion Feature**: Users are provided with suggestions based on the content structure, grammar, and generative ideas leveraging AI Backed APIs.
- **Template Based System**: Off-the-shelf templates enables users to begin skill-based content creation quickly.
- **Secure Login/Registration**: Users can sign up and access functional capabilities based on their role (student, supervisor or admin).
- **Firebase File Uploads**: Users can upload their e-books or documents of any standard template for next level creating/backing up.
- **Categorized folder trail**: Users can categorize their e-books into folders based on topic, skill, or anything based on their interest.

7.4 Testing Strategy

7.4.1 Functional Testing

Functional testing was performed manually to test each feature, including:

- Account creation, login/logout.
- Creating, editing, saving and deleting e-books.
- AI suggestions for grammar and skill content.
- File uploads and downloads (utilizing Firebase).
- Admin (to view shared content across the site).

7.4.2 Unit Testing

- Used Postman and Jest (backend) to test the core functions like e-book creation, folder management and API endpoints.
- Used React Native Testing Library on the frontend for component level testing.

7.4.3 End-to-End Testing

Simulated actual user interactions with the application to test that all components in the frontend and backend could work together seamlessly.

7.4.4 Security Testing

- JWT token authentication for user session tracking.
- Access control for user-specific documents.
- Implementation of input validation to test for XSS/injection vulnerabilities.

7.5 Evaluation and Runtime Comparison

When testing and deployment were completed, the app was assessed compared to the original project specifications and core modules. The system was evaluated based on:

- **Accuracy**: testing both relevance and grammar accuracy of AI-generated content.
- **Performance**: the app maintained reasonably fast API response times (< 500ms average) for usage performance purposes even when simulated concurrent usage was performed (load testing).
- **Scalability**: system architecture built using Node.js and MongoDB with future implementation opportunities (multi-user, offline mode, gamification).
- **Responsiveness**: responsive UI based on React Native framework for both Android and iOS with smooth navigation.

7.6 Controlled Libraries and Templates

- **React Native Components**: reusable UI components across different and similar screens.
- **AI Suggestion Module**: external AI service (Gemini API or OpenAI API) for curation of generated content and correction of content.
- **Code Walkthroughs**: code was clean and maintainable as individual code directly reviewed in on-going peer reviews and code walkthrough processes.

7.7 Testing Strategy and Use Case Coverage

7.7.1 Authentication

Table 2:TS01 Verify user login with valid credentials

Test Case ID	**TC01**
Title	Verify user login with valid credentials
Description	Ensure a user can log in using correct credentials.
Pre-Conditions	User account exists in the system.
Test Steps	1. Open the Smart AI Building E-Book app. 2. Enter valid username and password. 3. Click "Login".
Expected Result	User is successfully logged in and redirected to the home screen.

Table 3:TS02 Verify error message when logging in with incorrect credentials

Test Case ID	**TC02**
Title	Verify error message when logging in with incorrect credentials.
Description	Ensure that an appropriate error message is displayed when incorrect credentials are used.
Pre-Conditions	None
Test Steps	1. Open the Smart AI Building E-Book app. 2. 2. Enter an invalid username/password. 3. 3. Click "Login".
Expected Result	Error message "Wrong credentials. Please try again." appears.

Table 4:TS03 Verify success message when logging in with correct credentials

Test Case ID	TC03
Title	Verify success message when logging in with correct credentials
Description	Ensure that a success message is displayed upon successful login.
Pre-Conditions	User has entered correct credentials
Test Steps	1. Open the Smart AI Building E-Book app. 2. 2.Enter a valid username/password. 3. Click "Login".
Expected Result	Success message "Login successful!" is displayed.

7.7.2 Forgot Password

Table 5:TS04 Verify Forgot Password Functionality

Test Case ID	TC04
Title	Verify Forgot Password Functionality.
Description	Ensure that a user can successfully initiate the password recovery process.
Pre-Conditions	A user account must exist.
Test Steps	1. Go to the login page. 2. 2. Click on the "Forget Password" link. 3. Enter the registered email address. 4. Click on the "Send OTP" button.
Expected Result	The user is redirected to the Email Verification Screen with a message to check their email for an OTP.

Table 6:TS05 Verify OTP Email Verification

Test Case ID	**TC05**
Title	Verify OTP Email Verification.
Description	Ensure that a user can verify their email using the OTP received.
Pre-Conditions	User has initiated the forgot password process and received an OTP.
Test Steps	1. On the Email Verification Screen, enter the OTP received via email. 2. Click on the "Verify OTP" button. 3. If the user has not received the OTP, click on the "Resend OTP" button.
Expected Result	1. If the OTP is correct, the user is redirected to the Reset Password Screen. 2. If the OTP is incorrect, an error message is displayed. 3. If the "Resend OTP" button is clicked, a new OTP is sent to the user's email.

Table 7:TS06 Verify Password Reset Functionality

Test Case ID	**TC06**
Title	Verify Password Reset Functionality
Description	Ensure that a user can successfully reset their password.
Pre-Conditions	User has verified their email using OTP.
Test Steps	1. On the Reset Password Screen, enter a new password in the "New Password" field. 2. 2. Confirm the new password in the "Confirm Password" field. 3. 3. Click on the "Reset Password" button. 4. 4. Enter different passwords in the "New Password" and the "Confirm Password" field.
Expected Result	1. If the passwords match, the password is reset, and the user is notified of the successful reset. 2. If the passwords do not match, an error message is displayed: "Alert - Passwords do not match!".

7.7.3 User Signup

Table 8:TS07 Verify User Signup with Valid Credentials

Test Case ID	**TC07**
Title	Verify User Signup with Valid Credentials.
Description	Ensure that a new user can register an account with valid credentials.
Pre-Conditions	None
Test Steps	1. Open the Smart AI Building E-Book app. 2. Navigate to the signup page. 3. Enter valid information in all required fields (Full Name, Username, Password, and Confirm Password). 4. Click on the "Signup" or "Register" button.
Expected Result	The user is successfully registered and redirected to the home screen or a login success message is displayed.

Table 9:TS08 Verify User Signup with Invalid Credentials

Test Case ID	**TC08**
Title	Verify User Signup with Invalid Credentials.
Description	Ensure that an error message is displayed when a user tries to sign up with invalid credentials.
Pre-Conditions	None
Test Steps	1. Open the Smart AI Building E-Book app. 2. Navigate to the signup page. 3. Enter invalid information in one or more required fields (e.g., missing fields, mismatched passwords). 4. Click on the "Signup" or "Register" button.
Expected Result	An appropriate error message is displayed, indicating the reason for the failed signup attempt (e.g., "Incorrect credentials").

7.7.4 Home Screen

Table 10:TS09 Verify Home Screen Display

Test Case ID	TC09
Title	Verify Home Screen Display.
Description	Ensure the home screen is displayed correctly after successful login.
Pre-Conditions	User is logged in.
Test Steps	1. Perform login steps. 2. 2. Observe the screen after login
Expected Result	The home screen with a welcome message and user's name is displayed.

7.7.5 E-Book Creation

Table 11:TS10 Validate E-Book Creation with valid Input

Test Case ID	TC10
Title	Validate E-Book Creation with valid Input
Description	This test checks if users can successfully create new e-books with valid data.
Pre-Conditions	User must be logged in.
Test Steps	1. Navigate to the e-book creation page. 2. Enter a valid title, description, and content. 3. Click the save button.
Expected Result	book is successfully created and saved.

Table 12:TS11 Validate E-Book Creation with Invalid Input

Test Case ID	TC11
Title	Validate E-Book Creation with Invalid Input
Description	This test checks if the system handles invalid input during e-book creation.
Pre-Conditions	User must be logged in.
Test Steps	1. Navigate to the e-book creation page. 2. Enter an invalid title, description, and/or content. 3. Click the save button.
Expected Result	System displays an error message for missing or invalid input.

7.7.6 E-Book Reading

Table 13:TS12 Validate E-Book Reading with Valid Input

Test Case ID	TC12
Title	Validate E-Book Reading with valid Input
Description	This test checks if users can successfully read e-books with valid data.
Pre-Conditions	User must be logged in and have access to an e-book.
Test Steps	1. Navigate to the e-book reading page. 2. Open a valid e-book. 3. Navigate through the content
Expected Result	E-book content is displayed correctly. User can navigate through all pages.

Table 14:TS13 Validate E-Book Reading with Invalid Input

Test Case ID	**TC13**
Title	Validate E-Book Reading with Invalid Input
Description	This test checks if the system handles invalid input or missing content during e-book reading.
Pre-Conditions	User must be logged in.
Test Steps	1. Navigate to the e-book reading page. 2. Attempt to open an invalid or missing e-book.
Expected Result	Error message displayed for invalid or missing content.

Chapter 8

RESULTS AND DISCUSSION

8.1 Overview

The Smart Skill Building E-book Maker System App has been designed and tested in order to verify that it met the problem statement and functional requirements as defined initially. The overall intent of the project was to provide users with an intelligent, engaging and flexible platform to read and then create their own skill-based e-books with assistance from AI. The application has been put through rigorous testing procedures to confirm reliable performance, accurate AI recommendations, effective usability, and acceptable data integrity related to such features as template use, AI recommendations, cloud data saving, folder management, and user feedback. Extensive test cases were developed and executed to validate the system with many different end-user testing scenarios. Evaluation metrics included functionality, usability, performance, responsiveness, and scalability. Screen shots were taken from the working app in order to validate user interface performance and feature functionality.

8.2 Application Screens & Functional Overview

This section provides a detailed summary and assessment of each of the screens from the “Smart Skill Building E-book Maker System” app I implemented. Each of the screens are assessed according to functionality (what it does), the purpose (why it does it), interaction (how the users can contribute), as well as their performance or interaction based on testing experiences.

8.2.1 Splash Screen

- **Description**: Splash screen is the initial screen during application startup. Smart A1 Building can be recognized easily on the splash screen, together with the identification that it is an e-book application. Splash screen is employed to make a good impression and is typically displayed for a few seconds when the application is loading background resources.
- **Functionality**: Splash screen is a branding opportunity and is utilized for a seamless transition to the main app interface. It informs users of the app's purpose right from the start.

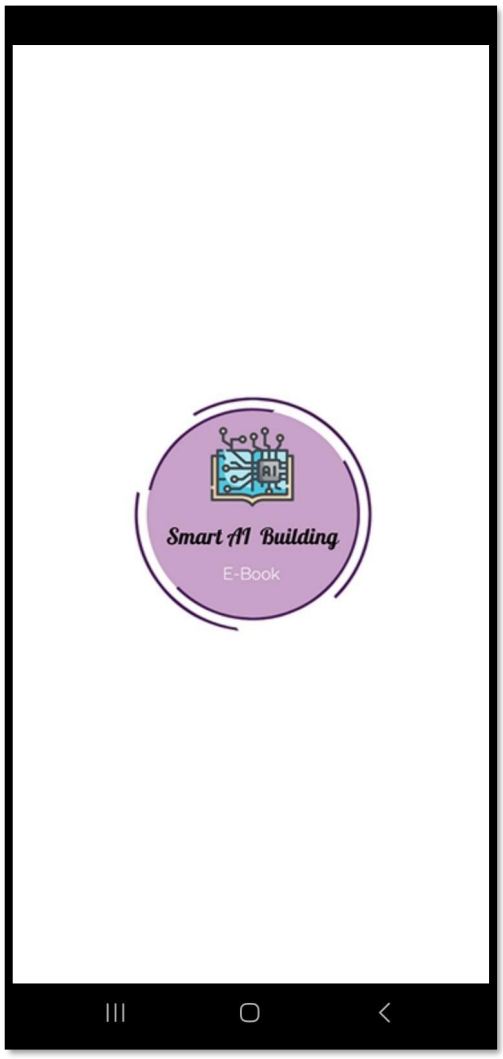

Figure 11:Splash Screen of Smart Skill E-book Maker App

8.2.2. Sign-Up & Login Screens

- **Description**: This screen facilitates user authentication. Users can either sign in with existing credentials or create a new account through sign-up.
- **Functionality**: Integrated with Firebase Authentication, it ensures secure login. Form validation prevents invalid entries, and navigation logic redirects users to the dashboard upon successful login.

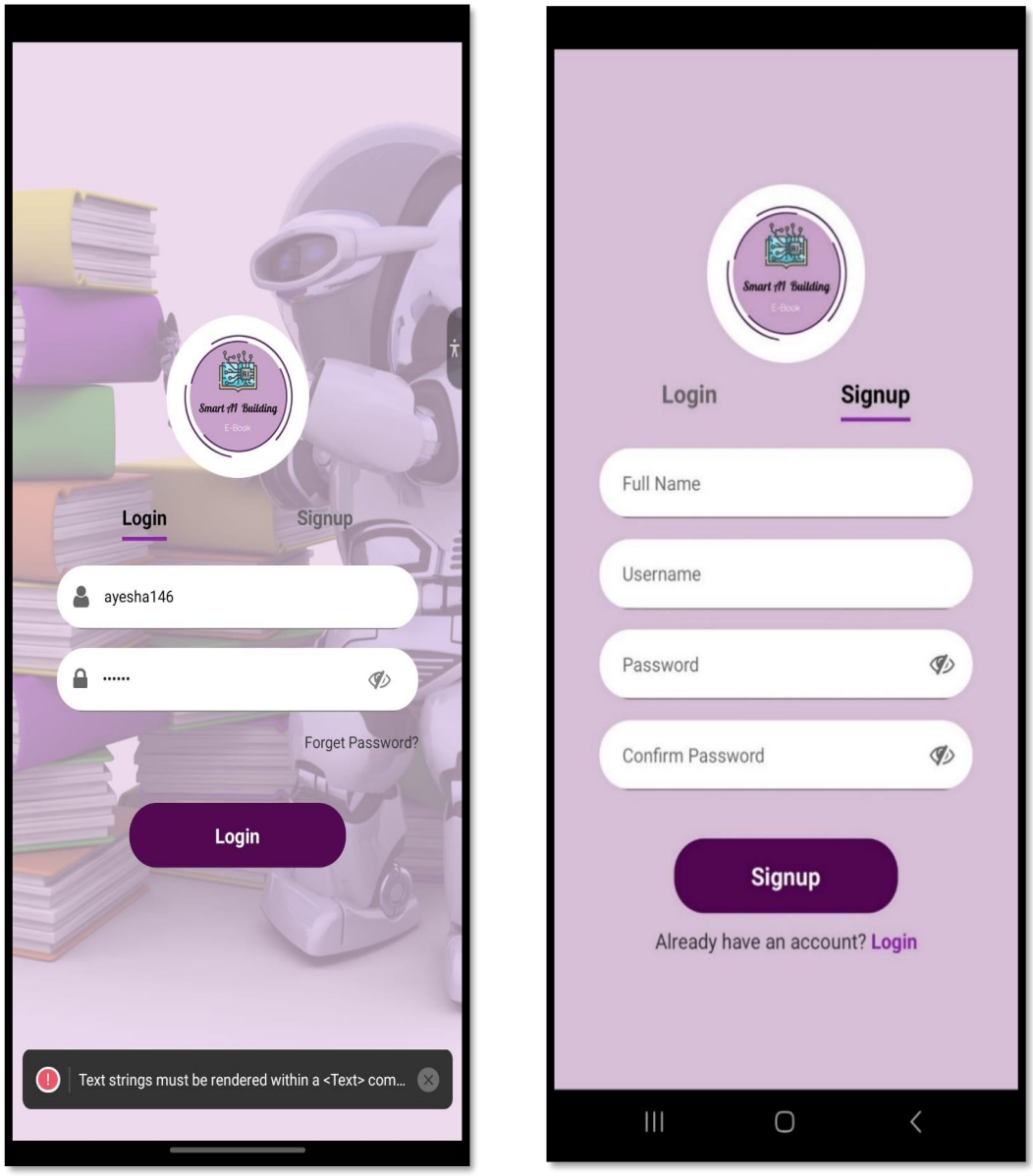

Figure 12:Login/Signup Screen for Registered Users

8.2.3 Forgot Password & Verification Screen

- **Description**: If users forget their password, they are redirected to this screen to recover their account.
- **Functionality**: Users input their registered email, and a verification code or reset link is sent via Firebase. After verification, users are allowed to set a new password and regain access.

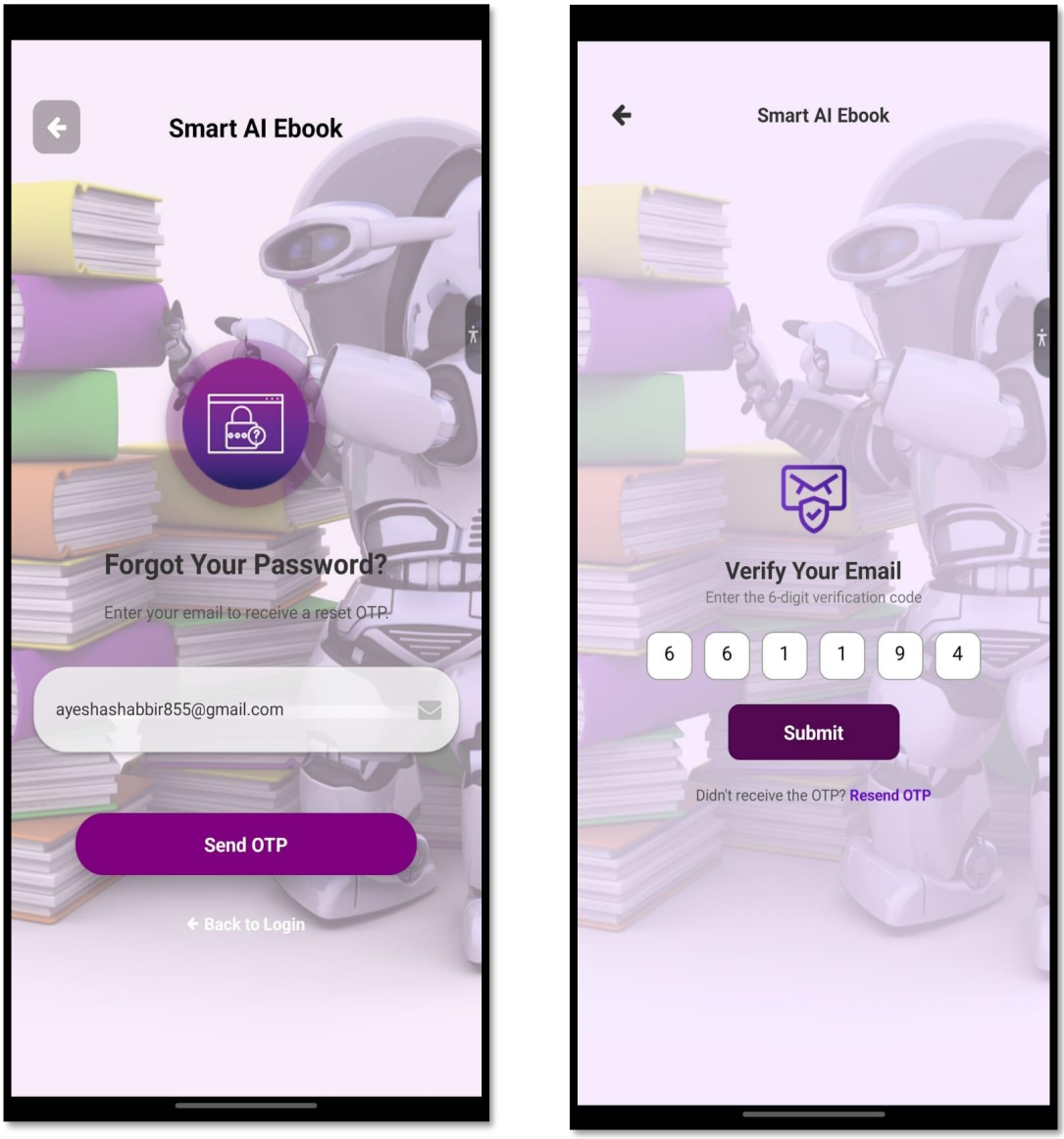

Figure 13:Forgot Password/ OPT Verification Screen

8.2.4 Welcome Home Screen

- **Description**: Once logged in, the user is greeted with a welcome screen offering two major options: Read E books or Write E-books.
- **Functionality**: It acts as a dashboard and navigation hub. Users choose the action they wish to perform. The screen also displays a summary of their recent activities and available features.

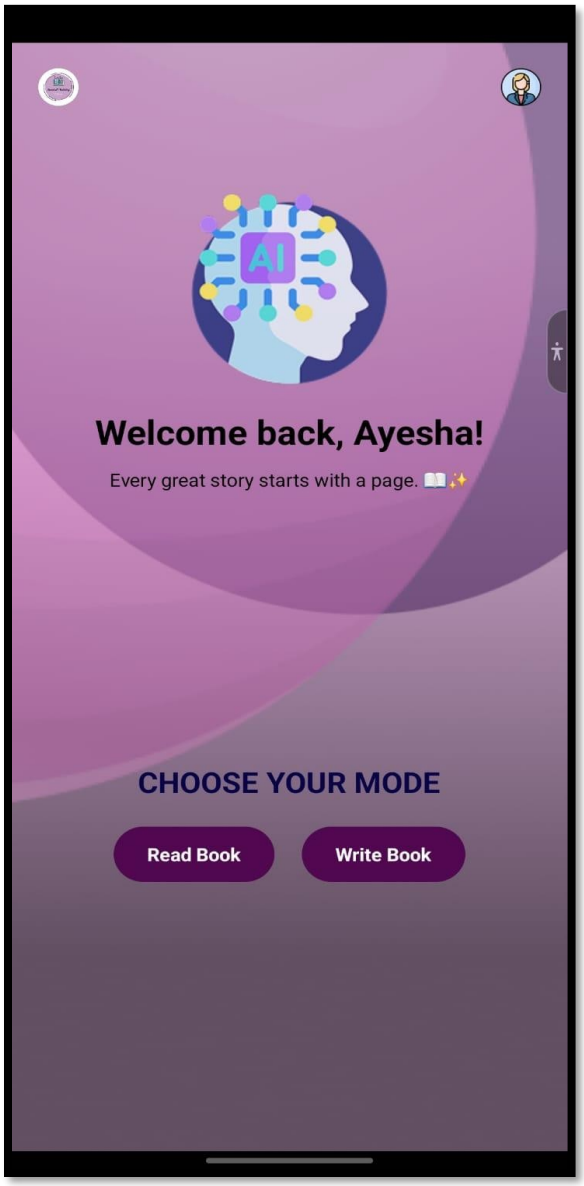

Figure 14:Dashboard Home Screen

8.2.5 Read Section Screen

- **Description**: If the user selects "Read," they are taken to this screen to browse and view saved or shared e books.
- **Functionality**: E-books are fetched from the Firebase database based on the user's history or categories. Users can open, like, or comment on a book. Navigation drawer provides access to features like Save, Share, Like, or return to Home.

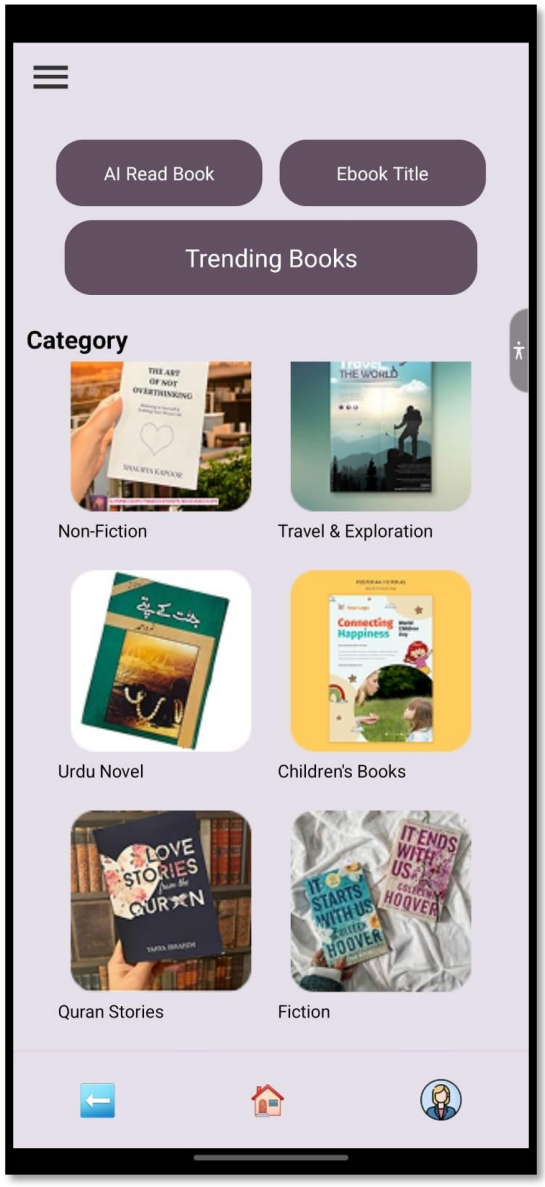

Figure 15:Read Mode Screen for book Skill Category Selection

8.2.6 Write Section Screen

- **Description**: If the user selects "Write," they can begin creating their own e-book from scratch or from a premade template.
- **Functionality**: A rich text editor enables adding text, formatting, and multimedia. Users can also use AI suggestions, templates, and voice-to-text. The side navigation includes Save, Share, and Exit options.

Figure 16:Write Book using Template Screen

8.2.7 AI Suggestion and Template Screen

- **Description**: This screen offers content ideas and smart templates powered by the Gemini API to assist users in writing structured and meaningful content.
- **Functionality**: AI provides suggestions for topics, summaries, and grammar improvements. Templates give a structured starting point for quick e-book creation.

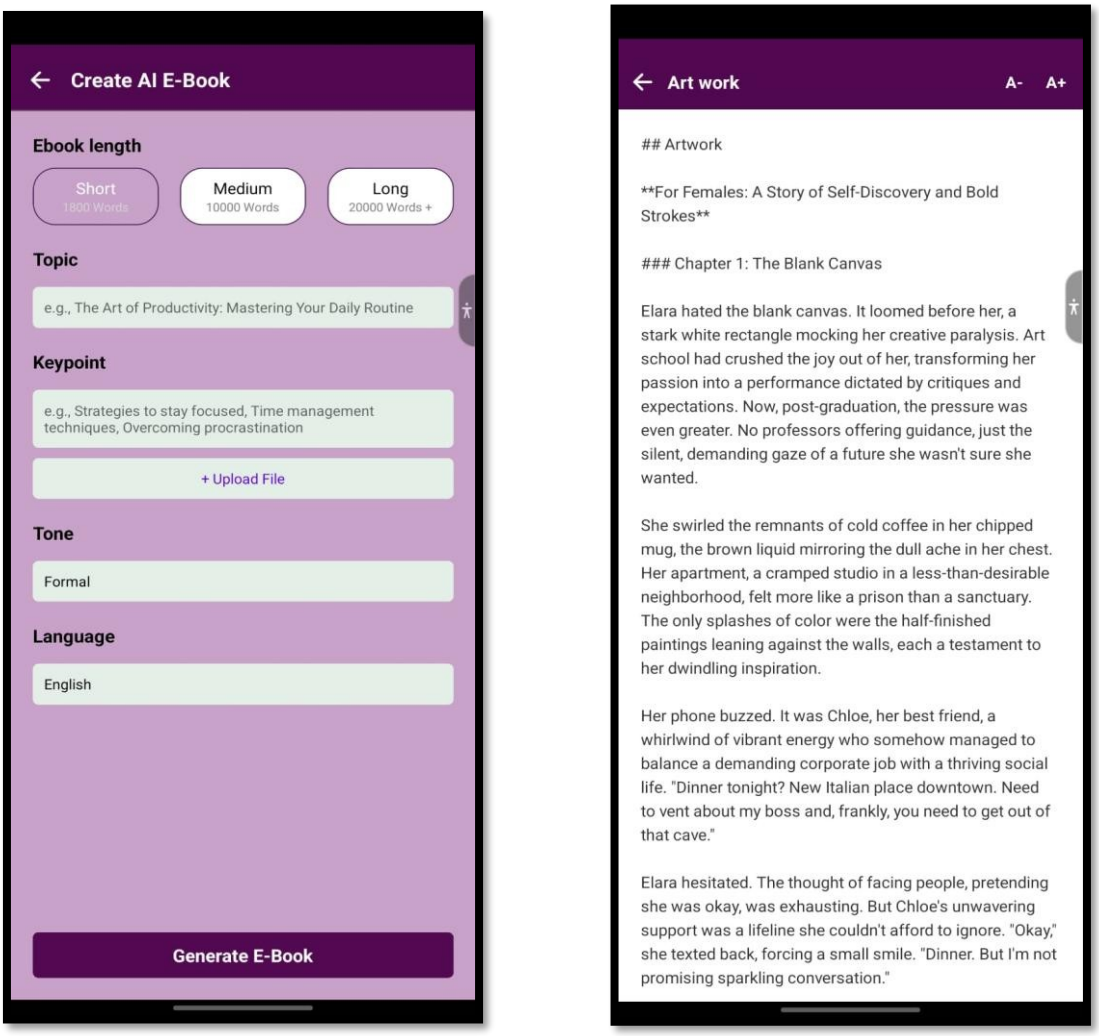

Figure 17:Write Book with AI Suggestions Screen

8.2.8 View/Read E-book Screen

- **Description**: Displays the selected e-book for full reading.
- **Functionality**: Users can scroll through the book, zoom, or navigate chapters. Options to Like, Share, or Save are also available in the navigation menu.

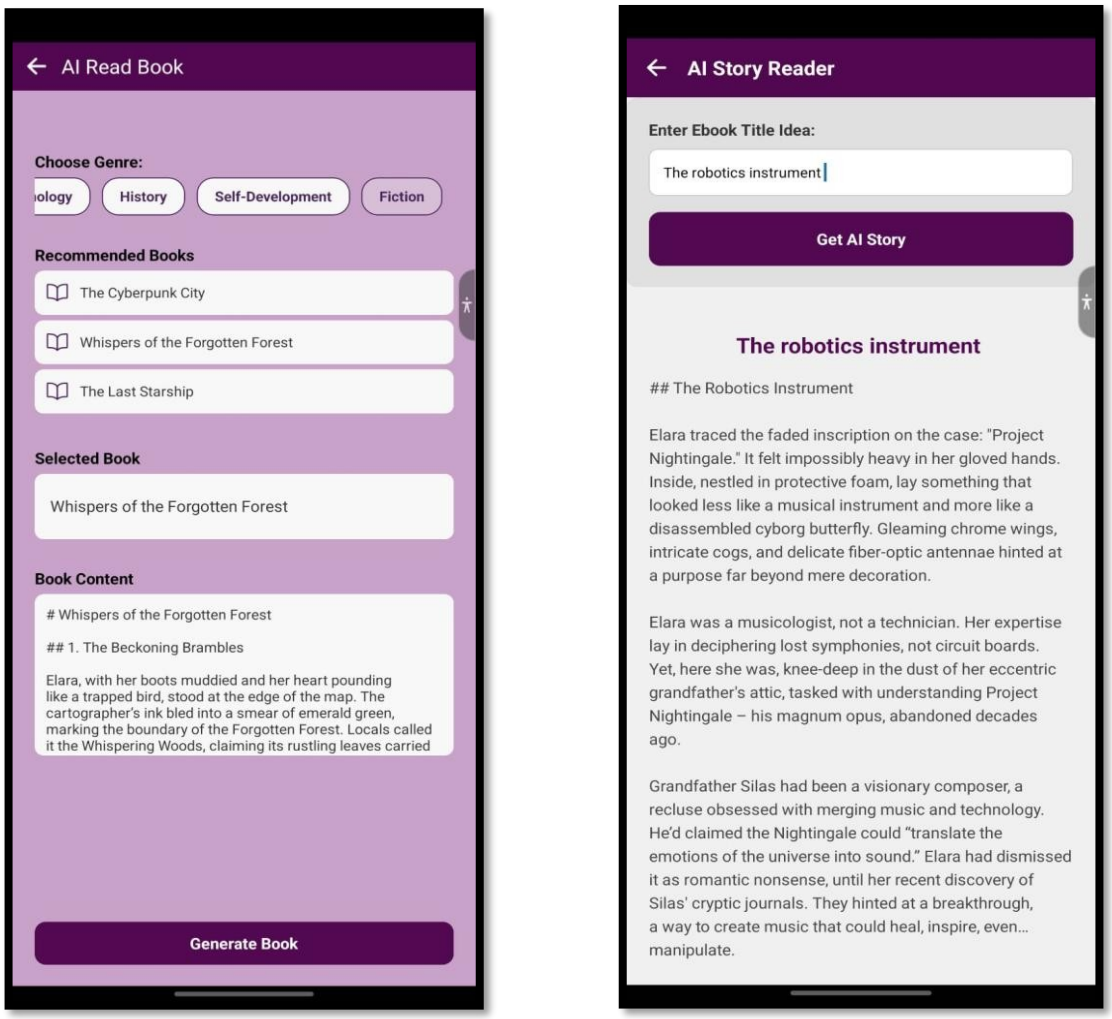

Figure 18: AI-Generated Book Detail View Read Mode Screen

8.2.9 Profile and Logout Screen

- **Description**: This screen provides access to the user's profile information and logout functionality.
- **Functionality**: Users can update their information, change passwords, and securely log out. Logging out clears session data and returns to the login screen.

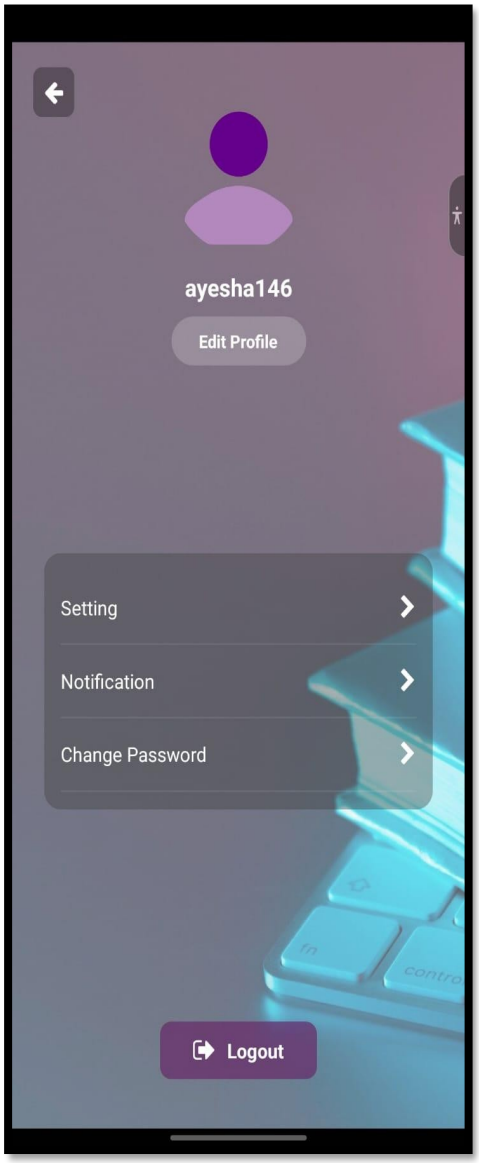

Figure 19:User Profile Screen

Chapter 9

CONCLUSION AND FUTURE WORK

9.1 Conclusion

The Smart Skill Building E-book Maker System App is an educational technology that is an exciting new learning technology because it included two important elements of learning for skill building: allowing users to create (write) e-books and consume (read) e-books. The Smart Skill Building E-book Maker System App is more than a static platform, it allows users - students, mentors, and learners - to produce original content, benefit from AI-supported writing apexes, manage their works in organized folders, use template-based e-book designs, and even offer templates for digital on-demand books, all in a single mobile app that simply was developed using React Native, Node.js, MongoDB, and Firebase.

The idea for the proposed solution was developed in response to the emerging trend for self-paced, AI-supported learning environments, particularly with respect to skill building and digital content production. The development arose from the concern that there appeared to be scant opportunities to engage in interactive, intelligent, rapidly self-customized educational platforms that allow content consumption (read) and content production (create). Therefore, content consumption (read) and content production (create) ought to be aligned with the learner's perceived learning journey.

As can be expected, the implementation of the system followed a design process from requirement gathering, and design in Figma to develop the system using React Native (frontend), Node.js (backend), and Firebase (installation), with application testing completed rigorously using functional, unit, and performance evaluation methods. The assessments confirmed that the Smart Skill Building E-book Maker System App performed as well as adhered to the functional requirements and also achieved the performance measures of responsiveness, AI accuracy and responsiveness.

9.2 Future Work

Several additional future directions and enhancements are needed to further extend the capabilities of the Smart Skill Building E-book Maker System App.

A. Gamification and Tracking Progress

Gamification features like badges, levels, and point-based achievements can encourage continued learning and e-book writing. Additionally, a progress-tracking module would allow users to view how they are progressing and accomplishing their learning objectives.

B. Offline Capability and Syncing

Another highly useful enhancement would be to implement offline capabilities so users could create or read content without being connected to the internet. Automatic sync could be implemented to upload the users' new or edited content upon re-establishing a connection. This would be especially useful for users in areas with limited connectivity.

C. Grammar and Spelling Checker

With AI We will continue to enhance AI in our Smart Skill Building E-book Maker System App, specifically with grammar and spelling suggestions in real-time leveraging natural language processing models. The grammars and spelling feature will significantly improve the quality of the content produced, as well as the writing skill of the users.

D. Voice-to-Text and Multilingual Features

Provide a voice input (speech to text) capability so users can dictate their content. Add multilingual and multi-language support and translation feature to generate e-books are available in a range of languages so that the e-book has a more significant global impact.

E. Admin Dashboard and Usage Analytics

Admin dashboard and analytics on user activity such as the templates used the most, the frequency of AI usage, and performance analytics to facilitate opportunities for enhancing the system based on users' behavior.

F. API Integrations with Clou AI Service Providers

Expand cloud-based NLP-based AI capability.

REFERENCES

[1] M. Ally, Mobile Learning: Transforming the Delivery of Education and Training, Athabasca University Press, 2009.

[2] L. Chen, P. Chen, and Z. Lin, “Artificial Intelligence in Education: A Review,” IEEE Access, vol. 8, pp. 75264–75278, 2020.

[3] S. Lee and J. Kim, “Content Creation and E-Book Development for Language Learning,” Educational Technology & Society, vol. 22, no. 3, pp. 78–88, 2019.

[4] R. Singh and G. Kaur, “A scalable architecture for e-learning systems using REST and MongoDB,” International Journal of Computer Applications, vol. 182, no. 47, pp. 10–15, 2019.

[5] T. Jones, L. Smith, and Y. Zhao, “Security and Privacy in Educational Technologies: Current Practices and Future Challenges,” Educational Technology Research and Development, vol. 66, no. 6, pp. 1367–1385, 2018.

[6] M. Rahman and T. Khatun, “Usability Challenges in AI-Powered Educational Software,” Journal of Interactive Learning Research, vol. 32, no. 1, pp. 21–38, 2021.