TABLE OF CONTENTS

ABSTRACT 4
CHAPTER 1 INTRODUCTION 5
1.1 INTRODUCTION 5
1.2 PROBLEM STATEMENT 6
1.3 SCOPE AND RELEVANCE OF THE PROJECT 6
1.4 OBJECTIVES 7
CHAPTER 2 SYSTEM ANALYSIS 9
2.1 INTRODUCTION 9
2.2 EXISTING SYSTEM 9
2.2.1 LIMITATIONS OF EXISTING SYSTEM 10
2.3 PROPOSED SYSTEM 11
2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM 11
2.4 FEASIBILITY STUDY 12
2.4.1 TECHNICAL FEASIBILITY 12
2.4.2 OPERATIONAL FEASIBILITY 13
2.4.3 ECONOMIC FEASIBILITY 13
2.5 SOFTWARE ENGINEERING PARADIGM APPLIED 14
CHAPTER 3 SYSTEM DESIGN 15
3.1 INTRODUCTION 15
3.2 DATABASE DESIGN 16
3.2.1 TABLE NAME: USER TABLE 16
3.2.2 TABLE NAME: AUTHENTICATION TABLE 16
3.1.3 TABLE NAME: BOOK TABLE 16
3.2.4 TABLE NAME: USER BOOK TABLE 16
3.3 OBJECT ORIENTED DESIGN 16
3.3.1 UML DIAGRAMS 16
3.3.1.1 USE CASE DIAGRAM 16
3.3.1.2 ACTIVITY DIAGRAM 16
3.3.1.3 CLASS DIAGRAM 16
3.3.1.4 SEQUENCE DIAGRAM 16
3.4 MODULAR DESIGN 17

3.5 FORM DESIGN 18
CHAPTER 4 SYSTEM ENVIRONMENT 19
4.1 SOFTWARE REQUIREMENTS SPECIFICATION 19
4.2 TOOLS, PLATFORMS 19
CHAPTER 5 SYSTEM IMPLEMENTATION 21
5.1 CODING 21
CHAPTER 6 SYSTEM TESTING 22
6.1 INTRODUCTION 22
6.2 UNIT TESTING 22
6.2.1 TEST PLAN 23
6.2.2 TEST CASES 25
6.3 INTEGRATION TESTING 26
6.4 SYSTEM TESTING 28
6.4.1 TEST PLAN 29
6.4.2 TEST CASES 30
CHAPTER 7 SYSTEM MAINTENANCE 32
7.1 INTRODUCTION 32
7.2 MAINTENANCE 32
CHAPTER 8 SYSTEM SECURITY MEASURES 34
8.1 INTRODUCTION 34
8.2 OPERATING SYSTEM-LEVEL SECURITY 34
8.3 DATABASE LEVEL SECURITY 35
8.4 SYSTEM-LEVEL SECURITY 36
CHAPTER 9 SYSTEM PLANNING AND SCHEDULING 37
9.1 INTRODUCTION 37
9.2 GANNT CHART 38
CHAPTER 10 SYSTEM COST ESTIMATION 39
10.1 INTRODUCTION 39
10.2 LOC BASED ESTIMATION / FUNCTION POINT BASED ESTIMATION 39
CHAPTER 11 CONCLUSION 41
CHAPTER 12 FUTURE ENHANCEMENT AND SCOPE OF FURTHER DEVELOPMENT 42
12.1 INTRODUCTION 42

12.2 MERITS OF THE SYSTEM 42
12.3 LIMITATIONS OF THE SYSTEM 43
12.4 FUTURE ENHANCEMENT OF THE SYSTEM 44
CHAPTER 13 ANNEXURE 46
13.1 ORGANIZATION PROFILE 46
13.2 DOCUMENT GLOSSARY, FIGURES, TABLES 46
13.2.1 ANNEXURE 1: GLOSSARY 46
13.2.2 ANNEXURE 2: TABLES 48
13.3.3 ANNEXURE 3: DIAGRAMS 51
13.2.4 ANNEXURE 4: FIGURES 61
13.2.5 ANNEXURE 5: CODING 66
13.3 REFERENCES 100

ABSTRACT

The Flutter-based E-Book App presents a comprehensive solution for users seeking a seamless digital reading experience across Android devices. Leveraging Flutter with Dart on the client side, the app ensures a consistent and intuitive user interface. By integrating Firebase on the server side database the application gains real-time database functionality, user authentication, and storage capabilities for user-uploaded books. This fusion of technologies enables users to effortlessly manage their digital libraries, access a plethora of features such as bookmarking, downloads, and text-to-speech, and enjoy a native-like experience on Android devices.

At its core, the E-Book App boasts a robust set of features designed to enhance the reading experience. Users can conveniently add, delete, and update books within their library, ensuring flexibility and customization. The inclusion of a PDF reader, rating and search book in system further enriches the user experience, providing seamless navigation and exploration of digital content. Additionally, the app's incorporation of text-to-speech functionality offers accessibility and convenience, allowing users to engage with their favorite books hands-free.

The synergy between Flutter and Firebase underscores the app's modernity and efficiency in e-book management. Flutter's cross-platform capabilities ensure broad accessibility, while Firebase's real-time database and authentication features facilitate seamless interaction and secure data management. This combination empowers users to not only manage their digital book collections effortlessly but also fosters an engaging environment for reading and exploration. Ultimately, the E-Book App serves as a testament to the potential of Flutter and Firebase in delivering a compelling, feature-rich solution for digital book enthusiasts.

Frontend: Flutter

Backend :Dart

Database: Firebase

CHAPTER 1 INTRODUCTION

1.1 INTRODUCTION

In the rapidly evolving landscape of digital reading, the E-Book App emerges as a beacon of innovation, poised to redefine the way we engage with literature. With the increasing prevalence of digital content consumption, the demand for a seamless and personalized reading experience has never been greater. Leveraging the dynamic capabilities of Flutter for frontend development and the robust infrastructure of Firebase for backend support, the E-Book App is positioned as a versatile platform that transcends traditional boundaries, offering users a diverse array of literary experiences tailored to their individual preferences.

Flutter, renowned for its versatility and efficiency, serves as the cornerstone of our app's development. As a cross-platform framework, Flutter enables us to seamlessly deploy the E-Book App across a myriad of devices and operating systems, ensuring a consistent user experience regardless of the platform. This inherent flexibility not only streamlines the development process but also enhances accessibility, empowering users to access their digital libraries anytime, anywhere. Central to our endeavor is the utilization of Firebase as the backend infrastructure for the E-Book App. With its comprehensive suite of services, Firebase provides a robust foundation for user authentication, real-time database functionality, and seamless data storage and retrieval. By harnessing the power of Firebase, we aim to create a dynamic ecosystem wherein users can effortlessly manage their digital book collections, interact with fellow bibliophiles, and explore new literary realms with unparalleled ease.

The E-Book App project extends far beyond mere digitization; it encompasses a holistic approach to revolutionizing the digital reading experience. From intuitive UI design to immersive reading features such as text-to-speech functionality and bookmarking, every aspect of the app is meticulously crafted to enhance user engagement and satisfaction. Moreover, by fostering collaboration with key stakeholders, including users, developers, and publishers, we ensure that the app remains agile and responsive to evolving needs and expectations. Through a combination of innovative technology, user-centric design, and strategic collaboration, we aspire to set a new standard for digital reading, ushering in a new era of literary exploration and discovery.

1.2 PROBLEM STATEMENT

The digital reading landscape is evolving rapidly, driven by shifting consumer preferences towards convenient and accessible solutions. However, existing e-book platforms often fall short in providing a seamless and personalized experience that caters to the diverse needs of modern readers. In response to this challenge, the E-Book App project aims to revolutionize the way users discover and engage with digital books. Traditional e-book apps often lack the versatility and functionality required to meet the evolving needs of readers, resulting in frustration and dissatisfaction among users. Moreover, the proliferation of digital content has exacerbated the need for efficient management and organization of digital libraries.

Existing solutions often lack robust user management features, making it challenging for users to effectively manage their collections and personalize their reading experiences. Additionally, the absence of integrated features such as text-to-speech functionality and seamless synchronization across devices further hinders the adoption of digital reading solutions. These limitations underscore the pressing need for a comprehensive and user-centric e-book platform that addresses the evolving needs and expectations of modern readers.

The E-Book App project seeks to address these challenges by leveraging the power of Flutter for cross-platform development and Firebase for robust backend services. By focusing on user-centric design and functionality, the project aims to create a seamless and intuitive reading experience that caters to users of all ages and backgrounds. The project aims to deliver a high-quality e-book app that meets the needs of modern readers and achieves success in the competitive digital market landscape.

1.3 SCOPE AND RELEVANCE OF THE PROJECT

The E-Book App project encompasses a broad scope aimed at transforming the digital reading experience by providing users with a comprehensive platform for discovering, purchasing, and engaging with digital books. Key functionalities include Pdf Reading View, User Management, Text to Speech, and Book Management, ensuring a multifaceted approach to addressing the diverse needs of modern readers. By leveraging Flutter for cross-platform development, the app will be accessible across a wide range of devices and operating systems, ensuring maximum reach and usability. Additionally, Firebase integration will facilitate robust backend services, including real-

time database functionality and seamless data storage and retrieval, further enhancing the app's functionality and user experience.

The relevance of the E-Book App project lies in its ability to address the evolving demands and preferences of modern readers in the digital age. With the proliferation of digital content and the increasing adoption of mobile devices, there is a growing need for convenient and accessible solutions for accessing and managing digital books. By offering a seamless and intuitive reading experience, the app aims to cater to users of all ages and backgrounds, fostering a love for reading and promoting literacy in the digital era.

Furthermore, the project's relevance extends beyond individual users to encompass stakeholders such as developers, publishers, and educators. Collaboration with these stakeholders will ensure that the app aligns with their respective goals and expectations, thereby maximizing its impact and value. By providing a platform for authors and publishers to showcase their work and interact with readers, the app will contribute to the growth and diversification of the digital publishing industry, fostering innovation and creativity in literary content creation. Overall, the scope and relevance of the E-Book App project extend far beyond mere digitization; it represents a transformative initiative aimed at revolutionizing the way we discover, consume, and engage with digital books in the modern age.

1.4 OBJECTIVES

The primary objective of the E-Book App project is to create a versatile and user-centric digital platform that revolutionizes the way users discover, purchase, and engage with digital books. Through the utilization of Flutter for cross-platform development and Firebase for robust backend services, the app aims to provide a seamless and personalized reading experience tailored to the diverse preferences of modern readers. Key objectives include implementing essential functionalities such as Pdf Reading View, User Management, Text to Speech, and Book Management, ensuring a comprehensive solution that caters to the evolving needs of users.

Moreover, the project aims to foster collaboration with stakeholders such as users, developers, publishers, and educators to ensure that the app aligns with their respective goals and expectations. By soliciting feedback and input from these stakeholders throughout the development process, the project aims to create an app that not only meets but exceeds user expectations, thereby

maximizing its impact and relevance in the competitive digital market landscape. Additionally, the project seeks to contribute to the growth and diversification of the digital publishing industry by providing a platform for authors and publishers to showcase their work and interact with readers, fostering innovation and creativity in literary content creation. Overall, the objectives of the E-Book App project are rooted in creating a transformative digital platform that empowers users to explore new literary realms, connect with fellow book lovers, and cultivate a lifelong love for reading in the digital age.

CHAPTER 2 SYSTEM ANALYSIS

2.1 INTRODUCTION

The E-Book App project represents a paradigm shift in the digital reading landscape, addressing the shortcomings of existing e-book platforms and ushering in a new era of convenience and accessibility. By leveraging Flutter for cross-platform development and Firebase for robust backend services, the project aims to provide users with a cohesive and feature-rich experience that transcends platform limitations. Through the integration of Flutter, the app ensures a seamless and personalized reading experience across diverse devices and operating systems, empowering users to engage with digital books effortlessly. Additionally, Firebase integration facilitates real-time database functionality and secure user authentication, laying the foundation for a dynamic and collaborative reading environment.

In response to the fragmented nature of the current e-book app landscape, the proposed system seeks to provide a unified and intuitive platform for discovering, purchasing, and engaging with digital books. By addressing key challenges such as platform-specific limitations and disjointed user experiences, the project aims to revolutionize the way users interact with digital content. Key functionalities including PDF Reader, Text-to-Speech, and User Management are seamlessly integrated into the app, ensuring a comprehensive and user-centric reading experience. Through the convergence of innovative technologies and user-centric design principles, the E-Book App project sets a new standard for e-book management solutions, offering users a modern, efficient, and engaging platform for accessing and enjoying digital books.

2.2 EXISTING SYSTEM

The current landscape of e-book applications presents several challenges that hinder the seamless reading experience for users. Across various platforms, users encounter fragmented and inconsistent interfaces, leading to confusion and frustration. Platform-specific constraints often limit the functionality of e-book apps, resulting in disparities in features and usability between different devices. Additionally, server-side integration for real-time updates and user management functionalities is often disjointed, leading to a lack of synchronization between devices and a suboptimal reading experience.

Moreover, the existing system fails to adequately address the diverse needs of modern readers. While some e-book apps offer basic functionalities such as reading and bookmarking, they lack advanced features that enhance the reading experience, such as text-to-speech functionality or collaborative reading capabilities. This limitation restricts users' ability to customize their reading experience and engage with digital content in innovative ways. Furthermore, the fragmented nature of the existing system makes it challenging for users to manage their digital libraries effectively, leading to cluttered and disorganized collections that detract from the overall reading experience.

The existing system falls short in providing a cohesive and feature-rich platform for digital reading. With inconsistencies in user interface design, limitations in functionality, and fragmented server-side integration, users are left with a disjointed and subpar reading experience. As the demand for digital reading solutions continues to grow, there is a pressing need for a comprehensive and user-centric e-book app that addresses these shortcomings and offers a seamless and engaging reading experience across platforms.

2.2.1 LIMITATIONS OF EXISTING SYSTEM

The current landscape of e-book apps suffers from a myriad of limitations that impede the seamless digital reading experience. Firstly, platform-specific constraints hinder user accessibility and consistency across different devices and operating systems. Users often encounter disparate interfaces and functionalities depending on whether they are using iOS or Android devices, leading to a fragmented user experience. This lack of uniformity not only complicates the user's interaction with the app but also undermines the overall reading experience, as users struggle to adapt to different interfaces and features.

The existing e-book apps often exhibit inconsistencies in user interface design, further exacerbating the challenges faced by users. These inconsistencies range from varying navigation patterns to inconsistent placement of essential features, making it difficult for users to navigate the app intuitively. Consequently, users may find it cumbersome to perform basic tasks such as browsing for books, managing their library, or customizing their reading preferences. Overall, these limitations of the existing system underscore the pressing need for a cohesive and feature-rich e-book platform that transcends platform-specific constraints and delivers a seamless reading experience across devices.

2.3 PROPOSED SYSTEM

The proposed E-Book App seeks to address the limitations of the existing digital reading landscape by providing users with a cohesive and feature-rich platform for discovering and engaging with digital books. Leveraging Flutter for cross-platform development and Firebase for robust backend services, the app aims to revolutionize the digital reading experience. Key to the proposed system is the seamless integration of Flutter and Firebase technologies, ensuring a unified user experience across Android platforms while harnessing the power of Firebase for real-time updates and user management functionalities.

Central to the proposed system is the implementation of essential features designed to enhance the reading experience for users. These features include Pdf Reader, Text-to-Speech functionality, and robust User Management capabilities, enabling users to seamlessly manage their digital libraries and customize their reading experience to suit their preferences. By offering a comprehensive set of features, the proposed system aims to cater to users of all ages and backgrounds, fostering a love for reading in the digital age.

Moreover, the proposed system prioritizes user-centric design and functionality, ensuring that the app is intuitive and easy to navigate for users. By soliciting feedback from stakeholders throughout the development process, the project aims to create an app that not only meets but exceeds user expectations. Through continuous iteration and improvement, the proposed E-Book App aims to set a new standard for digital reading solutions, providing users with a modern, efficient, and engaging platform for organizing, reading, and enjoying digital books.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

The proposed system presents a multitude of advantages that significantly elevate the user experience. Firstly, it ensures a unified experience across platforms, addressing the current fragmentation in user interfaces and functionalities. Leveraging Flutter for cross-platform development, the app guarantees consistency in design and functionality, enabling users to seamlessly transition between devices without encountering disparities in their reading experience. This unification not only enhances user satisfaction but also simplifies development and maintenance processes, streamlining operations for the development team.

Moreover, the proposed system offers seamless server-side integration, facilitating real-time updates and robust user management functionalities. By harnessing the capabilities of Firebase for backend services, the app ensures smooth synchronization of data across devices, enabling users to access their digital libraries and preferences effortlessly. This integration empowers users to enjoy a dynamic and personalized reading experience, with the flexibility to manage their collections and preferences in real-time. Overall, the proposed system's comprehensive set of features, coupled with its seamless server-side integration, positions it as a transformative solution that enhances the reading experience for users of all demographics.

2.4 FEASIBILITY STUDY

The feasibility study for the E-Book App project evaluates the practicality and viability of developing a digital platform that revolutionizes the way users engage with digital books. In terms of technical feasibility, the project benefits from the compatibility and capabilities of chosen technologies, such as Flutter for front-end development and Firebase for backend database support. These technologies align with the project requirements and possess the necessary functionalities to deliver a seamless and intuitive reading experience. Operational feasibility analysis ensures that the proposed system fits within the existing business environment and objectives. It examines integration with user workflows and administrative processes, ensuring smooth operation and alignment with user needs.

The economic feasibility evaluates the potential revenue generation and cost-effectiveness of the project. With potential revenue streams from app purchases and in-app purchases, the project demonstrates economic viability, ensuring a positive return on investment. By conducting a comprehensive feasibility study, the project team gains insights into potential challenges, risks, and opportunities, enabling informed decision-making and ensuring the project's viability in meeting its objectives effectively.

2.4.1 TECHNICAL FEASIBILITY

The technical feasibility of the E-Book App project is rooted in the compatibility and capabilities of the chosen technologies, which include Flutter for front-end development and Firebase database with Dart for backend support. Flutter, renowned for its cross-platform capabilities, enables developers to create a unified user interface and experience across Android platforms. Its extensive

widget library and hot reload feature streamline the development process, allowing for rapid iteration and refinement of the app's features and functionalities.

The integration of Firebase database with Dart ensures robust backend support for the E-Book App. Firebase provides a scalable and secure infrastructure for real-time data storage and retrieval, facilitating seamless synchronization of user-uploaded books and collaborative reading experiences. With Firebase's powerful authentication system and real-time database functionality, the app can deliver the desired functionalities, including Pdf Reading View, User Management, Text-to-Speech, and Book Management, while ensuring a smooth and responsive user experience. Overall, the technical feasibility of the project is underpinned by the synergy between Flutter and Firebase, enabling the development of a versatile and feature-rich e-book management solution.

2.4.2 OPERATIONAL FEASIBILITY

Operational feasibility analysis is crucial to ascertain the compatibility of the proposed system with the existing business environment and objectives. This assessment involves a comprehensive evaluation of how well the proposed E-Book App aligns with user workflows and administrative processes. By understanding the operational dynamics of the target environment, the project team can identify potential challenges and opportunities for seamless integration. Furthermore, operational feasibility analysis enables the project team to anticipate any necessary adjustments or adaptations required to ensure the smooth operation of the E-Book App within the existing business framework.

Through the operational feasibility analysis, the project aims to mitigate risks associated with potential disruptions to user workflows and administrative processes. By proactively addressing integration issues and streamlining operational procedures, the project team can ensure that the E-Book App seamlessly integrates into the existing business environment. This proactive approach not only minimizes the likelihood of operational disruptions but also enhances user acceptance and adoption of the new system, ultimately contributing to the overall success of the project.

2.4.3 ECONOMIC FEASIBILITY

The economic feasibility of the E-Book App project is underpinned by its potential to generate revenue through various channels. Primarily, revenue streams are anticipated from app purchases and in-app purchases, which are common monetization strategies in the digital marketplace. By

offering a compelling and feature-rich reading experience, the app is poised to attract a wide user base, thereby increasing the likelihood of generating significant revenue. Furthermore, the scalability of the app allows for future expansion and diversification of revenue streams, ensuring long-term sustainability and profitability. The project's cost-effectiveness is also a crucial aspect of its economic feasibility. By leveraging open-source technologies like Flutter for frontend development and Firebase for backend support, the project minimizes development costs while maximizing functionality and performance. Additionally, careful planning and resource allocation throughout the project lifecycle ensure efficient use of resources, further enhancing the project's cost-effectiveness. The economic feasibility of the E-Book App project is evident in its potential to generate revenue while maintaining a prudent approach to cost management, ensuring a positive return on investment for stakeholders.

2.5 SOFTWARE ENGINEERING PARADIGM APPLIED

The software engineering paradigm applied in this project is an iterative development model. This approach involves breaking down the project into smaller, manageable iterations, with each iteration focusing on implementing specific features or functionalities of the E-Book App. By adopting an iterative approach, the project team can deliver incremental updates and improvements to the app, allowing for continuous feedback and refinement throughout the development process.

The iterative development model enables the project team to effectively manage risks and uncertainties associated with software development. By continuously monitoring progress and adjusting timelines as needed, the team can identify and address potential issues early on, reducing the likelihood of delays or setbacks. This iterative approach also fosters flexibility and adaptability, allowing the project team to respond to changing requirements or user feedback in a timely manner, ultimately ensuring the successful implementation of the E-Book App.

CHAPTER 3 SYSTEM DESIGN

3.1 INTRODUCTION

In today's digital age, navigating the vast landscape of digital reading presents a myriad of challenges for modern readers. With a multitude of platforms available, selecting the right one that aligns with individual preferences and efficiently manages reading habits can be daunting. Traditional methods of organizing and accessing books often fall short, unable to keep pace with the evolving needs and expectations of readers. Additionally, the absence of personalized recommendations and convenient features further complicates the reading experience, leaving users searching for a solution that seamlessly integrates into their digital lifestyle.

Enter the e-Book App, poised to revolutionize the digital reading experience. At its core, this project aims to provide readers with a comprehensive digital platform that offers a seamless and immersive reading experience. By harnessing the capabilities of Flutter and Firebase technologies, the e-Book App seeks to transform the way users discover, access, and manage their digital book collections. With Flutter's cross-platform development framework and Firebase's robust backend services, the app promises to bridge the gap between readers and their digital libraries, offering a cohesive and intuitive platform for literary exploration.

Central to the e-Book App's mission is the incorporation of advanced functionalities tailored to meet the diverse needs of modern readers. From PDF Reader to User Management and Text-to-Speech capabilities, the app strives to offer a feature-rich environment that enhances the overall reading experience. By seamlessly integrating these functionalities, users can enjoy a personalized and convenient reading journey, empowering them to delve into new literary realms and connect with their favorite books like never before. The e-Book App project recognizes the importance of user-centric design and accessibility. Through intuitive interfaces and streamlined navigation, the app aims to provide users of all ages and backgrounds with an inclusive and enjoyable reading experience. Whether you're a seasoned bookworm or a curious newcomer, the e-Book App promises to cater to your unique reading preferences and elevate your digital reading experience to new heights. It represents a commitment to innovation and excellence in digital reading. By the app seeks to redefine the way readers interact with digital books, offering a seamless and immersive platform that inspires a lifelong love for reading in the digital era.

3.2 DATABASE DESIGN

3.2.1 TABLE NAME: USER TABLE

Refer Annexure 2.1

3.2.2 TABLE NAME: AUTHENTICATION TABLE

Refer Annexure 2.2

3.1.3 TABLE NAME: BOOK TABLE

Refer Annexure 2.3

3.2.4 TABLE NAME: USER BOOK TABLE

Refer Annexure 2.4

3.3 OBJECT ORIENTED DESIGN

3.3.1 UML DIAGRAMS

Refer Annexure 3

3.3.1.1 USE CASE DIAGRAM

Refer Annexure 3.1

3.3.1.2 ACTIVITY DIAGRAM

Refer Annexure 3.2

3.3.1.3 CLASS DIAGRAM

Refer Annexure 3.3

3.3.1.4 SEQUENCE DIAGRAM

Refer Annexure 3.4

3.4 MODULAR DESIGN

Module 1: Login

Introduction: The Login module with Google enables users to log in to the e-book app using their Google account, providing a convenient and secure authentication method.

Input: User selects the Google login option.

Processing: Initiate the Google authentication process and verify user credentials.

Output: User gains access to the app upon successful authentication.

Module 2: Search Book

Introduction: The Search Book module focuses on enabling users to efficiently find book within the app by implementing robust search functionality.

Input: User enters search book name on search bar.

Processing: Query Firebase to retrieve books matching the search criteria or filters.

Output: Display the filtered list of books based on user preferences.

Module 3: View Book

Introduction: The View Book module facilitates users in accessing comprehensive information about selected books from their digital library.

Input: User selects a book from the library.

Processing: Retrieve and display the details of the selected book from Firebase.

Output: User can view the title, author, and other details of the selected book.

Module 4: Pdf Reader

Introduction: The PDF Reader module is designed to provide users with a seamless reading experience for PDF documents within their digital library.

Input: User selects a PDF button to read.

Processing: Retrieve the selected PDF document from Firebase storage.

Output: Display the contents in the PDF document.

Module 5: Bookmark

Introduction: The Bookmark Book module allows users to bookmark their book page for easy access and reference within the app.

Input: User click bookmark in pdf reader.

Processing: The selected book's information bookmark process and retrieve in the page.

Output: Display the bookmark in the pdf reader.

Module 6: Text to Speech

Introduction: The Text to Speech module enhances the accessibility of the e-book app by converting text-based content into speech, allowing users to listen to e-books instead of reading them.

Input: Text content in pdf reader.

Processing: Utilize the device's text-to-speech engine to convert text into audible speech.

Output: User can listen to the e-book content through audio playback.

Module 7: Upload book

Introduction: The Upload Book module offers users to upload user books into the system. This module enhances user allowing them to contribute their own materials.

Input: Users enter the book detail and click upload button.

Processing: The module processes the uploaded digital book file into firebase storage.

Output: The uploaded book is added to the user's digital library.

Module 8: Download

Introduction: The Download module empowers users to retrieve digital books from their library for offline access or sharing purposes.

Input: Users select the download option for the desired book within their digital library.

Processing: The module retrieves the requested book from the system's database, ensuring its availability and integrity for download.

Output: Successful retrieval, the selected book is downloaded to the user's device

3.5 FORM DESIGN

Refer Annexure 4

CHAPTER 4 SYSTEM ENVIRONMENT

4.1 SOFTWARE REQUIREMENTS SPECIFICATION

- Language : Dart
- Operating System : Windows 7 or later version
- Front End : Flutter
- Back End : Dart
- Database : Fire Base
- Tool : Android emulator, Visual Studio Code

4.2 TOOLS, PLATFORMS

FRONT END TOOL: Flutter

The frontend of the E-Book App is developed using the Flutter framework, which serves as the primary tool for building the user interface. Flutter, an open-source UI software development kit (SDK) provided by Google, enables the creation of natively compiled applications for multiple platforms from a single codebase.

Flutter offers a rich set of pre-built widgets and components, facilitating the creation of visually appealing and interactive user interfaces. Its hot reload feature allows for instant feedback, making the development process efficient and enabling rapid prototyping.

Utilizing Flutter ensures a consistent and responsive user experience across platforms, including Android and iOS. The framework enables the E-Book App to have a modern and intuitive user interface, enhancing user engagement and satisfaction.

BACK END TOOL: Firebase

The backend of the E-Book App is built using Dart programming language with Firebase integration. Dart, a programming language developed by Google, is used for backend logic and data management.

Firebase, a comprehensive mobile and web development platform provided by Google, is integrated into the backend for robust backend services. Firebase offers features such as real-time database functionality, user authentication, and cloud storage, facilitating seamless synchronization and secure storage of user-uploaded books.

By leveraging Dart with Firebase, the E-Book App ensures efficient data management and seamless user experience, enabling users to discover, access, and manage their digital book collections with ease.

PLATFORMS: Android Emulator

The Android Emulator serves as a virtual device that simulates the functionality and behavior of Android devices on a computer. It enables developers to test their applications, including e-book apps, without needing physical devices. With the Android Emulator, developers can simulate various screen sizes, resolutions, hardware configurations, and Android versions, providing a comprehensive testing environment for app development. By running the e-book app on the emulator, developers can ensure that it functions correctly across different Android devices and versions, identifying and addressing any compatibility issues or bugs before releasing the app to users. The emulator also supports features like multi-touch input, accelerometer, and screen rotation, allowing developers to replicate real-world usage scenarios and ensure a smooth user experience.

To use the Android Emulator for an e-book app, developers typically set up the emulator environment using tools like Android Studio or Visual Studio Code, where they can configure virtual devices with specific characteristics such as screen size, RAM, and Android version. Once the emulator is set up, developers can deploy and run their e-book app directly from their development environment, enabling them to interact with the app as if it were running on a physical Android device. This allows developers to test various aspects of the e-book app's functionality, such as navigation, page rendering, and text formatting, ensuring that it meets the desired performance standards and user expectations before being released to the Google Play Store or other distribution channels.

CHAPTER 5 SYSTEM IMPLEMENTATION

System Implementation involves the actual realization of the meticulously planned project. It marks the transition from theoretical design to practical execution, where the envisioned solution is brought to life. This phase encompasses a series of strategic steps, encompassing the installation of necessary software components, coding and programming, integration of modules, and rigorous testing to ensureseamless functionality and alignment with project goals. System Implementation stands as a pivotal phase, where the culmination of efforts and meticulous planning materializes into a functional system, ready to address the targeted challenges and deliver impactful solutions

5.1 CODING

Refer Annexure 5

CHAPTER 6 SYSTEM TESTING

6.1 INTRODUCTION

System testing stands as a pivotal phase within the software development lifecycle, ensuring the integrity, functionality, and overall quality of the E-Book App. It serves as a meticulous and systematic approach to evaluating the app's performance, reliability, and security. The primary aim of system testing is to ascertain that all integrated components of the software work seamlessly, meeting predefined requirements and delivering an impeccable user experience in real-world scenarios.

In alignment with this principle, the E-Book App undergoes thorough system testing to validate its functionality and performance. This critical testing phase involves the execution of meticulously designed test scenarios to assess the app's behavior, identify potential defects, and ensure the cohesive operation of all features and functionalities. Through rigorous system testing, our objective is to furnish users with a seamless and dependable platform, ensuring that the app effectively fulfills its core purpose of managing digital book collections efficiently.

The system testing process encompasses a diverse range of methodologies, including functional testing, performance testing, security testing, and usability testing. Each methodology serves to comprehensively evaluate the app's performance across various dimensions, ensuring that it meets the highest standards of quality and reliability. By subjecting the E-Book App to rigorous testing, we can proactively detect and rectify any issues early in the development cycle, thus delivering a refined, robust, and user-friendly product that exceeds user expectations.

6.2 UNIT TESTING

Unit testing ensures the correctness and functionality of individual modules within the eBook App. For instance, testing the Google login authentication module involves verifying the authentication process's accuracy using mock user credentials. Similarly, unit testing for user profile management checks if user data is stored and retrieved correctly from Firebase Firestore. An example of unit testing for the text-to-speech module involves validating the conversion of text to speech using sample text inputs.

6.2.1 TEST PLAN

1. INTRODUCTION

The primary goal of the unit testing plan is to ensure the reliability, functionality, and performance of individual modules within the e-book app under various scenarios. This document outlines the testing process, objectives, scope, phases, deliverables, environment, and risk mitigation strategies.

2. TEST OBJECTIVES

The main objectives of unit testing for the e-book app are as follows:

a) Verify the functionality and accuracy of each module, including Login, Search Book, View Book, PDF Reader, Bookmark, Text to Speech, Upload Book, and Download.

b) Ensure that each module integrates seamlessly with other components and maintains compatibility across different platforms, specifically iOS and Android.

c) Assess the performance of individual modules under various conditions to determine their responsiveness and scalability.

d) Validate the security measures implemented in each module to safeguard user data and transactions.

3. TEST SCOPE

The unit testing scope encompasses testing of the following modules within the e-book app:

a) Login: Verify Google authentication process and user credential validation.

b) Search Book: Test search functionality and data retrieval from Firebase.

c) View Book: Validate the retrieval and display of book details from Firebase.

d) PDF Reader: Ensure seamless display of PDF content retrieved from Firebase storage.

e) Bookmark: Test bookmarking feature functionality within the PDF reader.

f) Text to Speech: Validate the conversion of text to audible speech.

g) Upload Book: Test the process of uploading digital books to Firebase storage.

h) Download: Ensure successful retrieval and download of digital books from the system's database.

4. TEST PHASES

The unit testing process will be divided into the following phases:

a) Unit Testing: Testing individual modules such as Login, Search Book, View Book, etc., to ensure their correctness and functionality.

b) Integration Testing: Verifying the interaction and integration of different modules to ensure seamless operation.

c) System Testing: Evaluating the performance of the entire e-book app under various conditions.

5. TEST DELIVERABLES

At the end of the unit testing process, the following deliverables will be provided:

a) Test Cases: Detailed test cases covering all functionalities and scenarios for each module.

b) Test Results: Comprehensive reports on test execution, including pass/fail status and identified defects.

6. TEST ENVIRONMENT

Unit testing will be conducted in a controlled environment with the following specifications:

a) Devices: Android smartphones for platform-specific testing.

b) Operating Systems: Android for compatibility testing.

c) Database: Firebase for data storage and retrieval.

d) Testing Tools: Appropriate testing tools for unit testing, such as JUnit for Android.

7. RISKS AND MITIGATIONS

Potential risks associated with unit testing, such as integration issues and performance bottlenecks, will be identified and mitigated throughout the testing process. Collaborative efforts will be made to address any issues promptly, ensuring a successful and reliable unit testing phase.

6.2.2 TEST CASES

Test Case ID	Test Objective	Precondition	Steps/ Cases	Test Data	Expected Result	Post Condition
TC_Login_01	Verify login functionality using Google authentication.	User is not logged in.	1.Initiate Google login authentication process. 2. Enter valid Google credentials. 3.Verify successful authentication.	Valid Google account credentials	User should be successfully logged in and granted access to the eBook app	User is logged in with their Google account.
TC_Book_Addition_02	Validate book addition functionality.	User is logged in and accessing the book management section.	1.Navigate to the book management section. 2. Add a new book to the app.	New book file for upload.	New book should be successfully added to the app	Book is successfully added and available for user access.
TC_Book_Search_04	Test book search functionality.	User is logged in and accessing the search feature.	1. Enter a search query in the search bar. 2. Initiate the search.	Search query	Relevant books matching the search query should be displayed.	Books matching the search query are listed for user access.
TC_Book_Details_Display_05	Verify book details display accurately.	User is logged in and selecting a book to view.	1. Select a book from the library. 2. View book details.	Selected book.	Detailed information about the selected book, including author, genre,	User has access to detailed book information.

					and synopsis, should be displayed	
TC_Bookmarking_06	Test bookmarking functionality within a book.	User is logged in and viewing a book.	1. Navigate to a specific page within the book. 2. Bookmark the page.	Selected page within the book.	The selected page should be successfully bookmarked for future reference.	Bookmark is saved for the selected page.
TC_Text_To_Speech_07	Ensure text-to-speech feature functions correctly	User is logged in and viewing a book.	1. Select a portion of text within the book. 2. Initiate the text-to-speech feature	Selected text within the book.	The selected text should be accurately converted to speech and played audibly.	User can listen to the selected text via the text-to-speech feature.

6.3 INTEGRATION TESTING

Integration testing for the e-book app verifies the interaction and communication between various modules to ensure they function seamlessly together, delivering a cohesive user experience. The testing approach includes both top-down and bottom-up strategies, focusing on data flow, API integrations, and module interactions.

TESTING PROCEDURE:

a. Identify Integration Points: Identify where different modules interact, such as APIs, database connections, and function calls.

b. Prepare Test Data: Prepare test data to simulate real-world scenarios and ensure effective handling by modules.

c. Develop Test Cases: Create integration test cases covering positive/negative scenarios, boundary conditions, and error handling.

d. Execute Test Cases: Execute test cases to verify module interactions and compare expected versus actual results.

e. Check Data Integrity: Verify data transfer accuracy between modules to ensure integrity.

f. Test Error Handling: Assess error handling mechanisms to ensure appropriate responses to exceptions.

g. Test Integration with External Systems: Verify integration with external systems or APIs for smooth communication.

h. Test Concurrent Usage: Evaluate system behavior under concurrent access to identify and address any concurrency issues.

i. Monitor Performance: Monitor system performance to identify and rectify any performance bottlenecks.

j. Regression Testing: Conduct regression testing to ensure existing functionalities are unaffected by integration changes.

k. Review Logs and Debug: Review logs to debug and resolve any issues identified during testing.

l. Document Results: Document testing outcomes, including identified issues and their resolutions.

m. Retest Fixes: Retest fixes to ensure successful resolution of identified issues.

n. Approval and Sign-off: Obtain stakeholder and project management approval for successful integration testing.

o. Continuous Monitoring: Continuously monitor the integrated system to maintain stability and reliability.

TEST CASES AND PURPOSE:

a) Test Case ID: IT_TC_001

- Test Objective: Verify user registration and login functionality integration.
- Purpose: Ensure users can register with valid details and log in successfully.

b) Test Case ID: IT_TC_002

- Test Objective: Test integration of installer management.

- Purpose: Verify listing and selection of approved installers by service providers.

c) Test Case ID: IT_TC_003

- Test Objective: Validate grid connection integration for completed solar plants.
- Purpose: Ensure the system can establish grid connections for installed solar plants.

d) Test Case ID: IT_TC_004

- Test Objective: Test integration of subsidy approval after successful grid connection.
- Purpose: Verify subsidies can be approved for solar plants with successful grid connections.

e) Test Case ID: IT_TC_005

- Test Objective: Validate user application submission integration.
- Purpose: Ensure users can submit installation requests and service providers can process them.

f) Test Case ID: IT_TC_006

- Test Objective: Test integration of user tracking of installation progress.
- Purpose: Verify users can track the status and progress of their installation requests.

g) Test Case ID: IT_TC_007

- Test Objective: Validate installer completion of installations integration.
- Purpose: Ensure installers can mark installations as completed.

h) Test Case ID: IT_TC_008

- Test Objective: Test integration of user submission of ratings and feedback for installations.
- Purpose: Verify users can provide ratings and feedback for completed installations.

These test cases ensure smooth interaction between different e-book app modules, delivering an integrated system that meets user expectations.

6.4 SYSTEM TESTING

System testing is a crucial phase in the software development process that aims to validate the entire integrated system's functionality. It ensures that all components of the e-book application

work together seamlessly and meet the specified requirements. The system testing plan and test cases for the e-book app are outlined below:

6.4.1 TEST PLAN

Test Plan for E-Book Application:

- Objectives and Scope: Define the objectives and scope of testing, including evaluating the app's performance, reliability, and adherence to user expectations.
- Testing Environment: Set up a dedicated testing environment with appropriate hardware and software configurations to simulate real-world usage scenarios.
- Test Cases: Create detailed test cases for each module and functionality of the e-book app, covering aspects such as login, book search, view book, PDF reader, bookmarking, text-to-speech, upload book, and download.
- Test Data: Prepare test data to support the test cases, including sample books, user credentials, and other necessary information.
- Test Execution Schedule: Develop a test execution schedule considering available resources and team members' availability to ensure efficient testing.
- Execution and Defect Tracking: Execute the test cases, record defects, and use a defect tracking system to manage and monitor the resolution progress.
- Test Results Analysis: Analyze the test results to identify any defects or discrepancies that require attention and ensure they are addressed promptly.
- Risk Assessment: Conduct a risk assessment to outline potential risks associated with the testing process and develop mitigation strategies to address them effectively.
- Test Report: Generate a comprehensive test report documenting the entire testing process, test results, and identified defects.
- Test Sign-off Criteria: Use test sign-off criteria to determine when the e-book application is ready for deployment based on the successful completion of testing objectives.

By following this test plan, we aim to ensure that the e-book application undergoes a thorough and rigorous evaluation, delivering a high-quality, reliable, and user-friendly platform for digital reading.

6.4.2 TEST CASES

Test Cases for E-Book Application:

a) Login Module:

Test Case 1: Verify that users can successfully log in using their Google account credentials. Test Case 2: Verify that users are redirected to the home screen upon successful authentication.

b) Search Book Module:

Test Case 1: Verify that users can search for books by entering keywords in the search bar. Test Case 2: Verify that the app displays relevant search results based on the user's input.

c) View Book Module:

Test Case 1: Verify that users can view detailed information about selected books from their digital library.

Test Case 2: Verify that the app retrieves and displays book details accurately from the Firebase database.

d) PDF Reader Module:

Test Case 1: Verify that users can open and read PDF documents within the app's PDF reader feature. Test Case 2: Verify that the PDF reader displays the contents of selected books accurately.

e) Bookmark Module:

Test Case 1: Verify that users can bookmark pages within PDF documents for easy access and reference. Test Case 2: Verify that bookmarked pages are saved and displayed correctly within the PDF reader.

f) Text to Speech Module:

Test Case 1: Verify that users can listen to e-book content through audio playback using the text-to-speech feature. Test Case 2: Verify that the text-to-speech engine accurately converts text-based content into audible speech.

g) Upload Book Module:

Test Case 1: Verify that users can upload their digital books into the app's digital library. Test Case 2: Verify that uploaded books are processed and added to the user's library correctly.

h) Download Module:

Test Case 1: Verify that users can download digital books from their library for offline access. Test Case 2: Verify that downloaded books are saved and accessible on the user's device.

These test cases cover various aspects of the e-book application's functionality and ensure that it meets user requirements and expectations effectively.

CHAPTER 7 SYSTEM MAINTENANCE

7.1 INTRODUCTION

System maintenance is a critical phase in the software development lifecycle that ensures the continued optimal performance, reliability, and usability of the e-book application. It involves proactively managing and enhancing the software to address any issues that may arise after deployment and to accommodate future changes or updates.

During the system maintenance phase, the development team is responsible for monitoring the application, identifying and resolving any defects or bugs, and implementing necessary updates or enhancements based on user feedback and changing business requirements. The primary goal of system maintenance is to sustain the application's functionality, security, and performance while minimizing any potential downtime or disruptions. This phase also includes regular backups of critical data, database optimization, security patches, and updates to ensure the protection of sensitive information and maintain data integrity. Additionally, the team may need to address compatibility issues with new operating system versions or hardware updates to ensure the application remains compatible with evolving technologies.

System maintenance is an ongoing process that requires a systematic approach, effective communication between development teams and stakeholders, and a commitment to continuous improvement. By prioritizing system maintenance, the e-book application can evolve and adapt to meet the ever-changing needs of users and technology, providing a sustainable and reliable platform for digital reading.

7.2 MAINTENANCE

System maintenance for the e-book application involves a series of activities aimed at preserving the system's functionality, reliability, and security to ensure its optimal performance over time.

One of the key components of maintenance is bug fixing and issue resolution. The development team actively identifies and addresses any defects or problems that users may encounter while using the application. This includes investigating and resolving software errors, performance issues, and other technical glitches.

Regular updates and enhancements are also essential in system maintenance. As technology and user requirements evolve, the system must adapt accordingly. The team works on implementing updates to improve existing features, add new functionalities, and enhance the overall user experience.

Furthermore, data integrity and security are paramount in the e-book application. Regular data backups are performed to safeguard critical information, and security measures are continuously monitored and strengthened to protect against potential threats.

System maintenance is an ongoing process, and it involves collaboration between the development team and stakeholders to ensure the system remains efficient and reliable. By prioritizing maintenance activities, the e-book application can deliver consistent and high-quality performance, meeting the needs of users while keeping up with advancements in technology.

CHAPTER 8 SYSTEM SECURITY MEASURES

8.1 INTRODUCTION

In the digital realm of the e-Book App, security stands as a paramount concern to safeguard sensitive user data, prevent unauthorized access, and uphold the integrity of information. The implementation of robust security measures is imperative to foster user trust and ensure the reliability of the system.

Given the ever-evolving landscape of cyber threats, the e-Book App is fortified with a comprehensive array of security protocols to counter potential vulnerabilities and breaches effectively. This introduction delineates the various security measures integrated into the system to fortify its defenses. Emphasizing a multi-layered approach, the security framework commences with stringent user authentication and access controls. Through meticulously designed login procedures, only authenticated users are granted entry, while granular permission settings restrict access to pertinent functionalities.

Throughout the development lifecycle, the e-Book App adheres to secure coding practices, ensuring robust defense mechanisms against potential exploits. Prompt application of software patches and updates is a standard practice to address emergent security vulnerabilities. Moreover, continuous monitoring and logging of system activities facilitate the early detection of anomalous behavior, enabling swift responses to potential security incidents.

By instilling confidence through robust security measures, the e-Book App endeavors to provide users with a safe and trustworthy platform for their digital reading endeavors.

8.2 OPERATING SYSTEM-LEVEL SECURITY

Operating system-level security is paramount in ensuring the robustness and integrity of the e-book app, safeguarding user data and system functionality. Similar to the "SUN CONNECT" Solar Energy Management System, the e-book app relies on the operating system as a foundational layer for managing resources, processes, and data. Implementing stringent security measures at this level is imperative to thwart potential threats and maintain the app's reliability. Regular Updates and Patches: Continuous updates and patches are applied to the operating system to address known

vulnerabilities and enhance security. By staying up-to-date with the latest security fixes and improvements, the e-book app mitigates the risk of exploitation by cyber threats.

User Authentication and Access Controls: User authentication mechanisms are implemented to verify the identity of users accessing the e-book app. Access controls are enforced to define each user's privileges, ensuring that only authorized individuals can interact with sensitive data and critical system components. This safeguards against unauthorized access and mitigates the risk of data breaches.

Backup and Recovery: Regular backups of critical data and system configurations are conducted to ensure data integrity and facilitate quick recovery in the event of a security incident or hardware failure. By maintaining reliable backup mechanisms, the e-book app can minimize downtime and mitigate the impact of security breaches.

Overall, operating system-level security measures are instrumental in fortifying the e-book app against potential threats, enhancing its resilience, and safeguarding user information and system functionality.

8.3 DATABASE LEVEL SECURITY

Database level security is paramount in ensuring the confidentiality, integrity, and availability of data within the e-book app. As the repository of critical information, the database holds sensitive user details and valuable content that must be safeguarded from unauthorized access and malicious activities.

To enforce robust security at the database level, several measures are implemented. First and foremost, access controls are put in place to restrict user access to specific database tables and functionalities based on their roles and privileges. This ensures that only authorized users can view, modify, or delete specific data, preventing unauthorized access or manipulation of sensitive information.

Database backups are regularly performed to ensure data integrity and availability in the face of unforeseen events such as system failures or data corruption. These backups are securely stored to facilitate quick recovery in case of data loss or system compromise. To defend against SQL injection attacks and other vulnerabilities, input validation is employed to sanitize user inputs

before processing them in the database. This prevents malicious code injection and helps maintain the database's stability and security.

8.4 SYSTEM-LEVEL SECURITY

System-level security is crucial for the e-book app, aimed at safeguarding the entire application and its underlying infrastructure from potential threats and unauthorized access. This comprehensive approach encompasses various measures and practices to ensure the confidentiality, integrity, and availability of the entire system.

One of the primary components of system-level security is access control. Strict access controls are implemented to limit user access to the application's functionalities based on their roles and permissions. This prevents unauthorized users from accessing sensitive information or performing critical actions within the system. Authentication are vital aspects of system-level security. Users are required to authenticate themselves using strong and secure credentials before gaining access to the system. Additionally, authorization ensures that users can only perform actions that are appropriate for their roles and responsibilities.

Data encryption is employed at various levels to protect data in transit and at rest. Secure communication protocols, such as SSL/TLS, are used to encrypt data transmitted between the application and the users' devices, ensuring that sensitive information remains confidential during transmission. Regular system updates and patches are applied to keep the system up-to-date with the latest security fixes and enhancements. Vulnerability assessments and penetration testing are performed to identify potential weaknesses in the system and address them proactively.

System-level security measures are paramount in ensuring the integrity and confidentiality of user data within the e-book app. Adopting secure coding practices during the development phase, such as rigorous code reviews and vulnerability scanning, is essential to mitigate potential threats and vulnerabilities. Additionally, adherence to industry security standards further fortifies the app's defenses against cyber threats and unauthorized access attempts. By integrating robust system-level security measures, including comprehensive code reviews and adherence to industry standards, the e-book app can provide users with a secure and trustworthy environment for managing their digital library. These measures ensure the protection of critical user information and the seamless operation of the app's functionalities, enhancing user confidence and satisfaction.

CHAPTER 9 SYSTEM PLANNING AND SCHEDULING

9.1 INTRODUCTION

System planning and scheduling form the backbone of any project, providing the roadmap for its successful execution. In the context of our e-book application, meticulous planning and scheduling are paramount to ensure a seamless development process and timely delivery of a high-quality product. This phase encompasses various essential steps aimed at understanding project requirements, defining system architecture, and orchestrating the development timeline.

1.System Study

At the outset, our project team embarks on an extensive system study to grasp the nuances of the e-book application project fully. We engage in stakeholder consultations to capture their expectations and ascertain project scope and feasibility. This phase involves evaluating existing processes and systems, if any, to pinpoint areas for enhancement or automation. Ultimately, the insights gathered culminate in a comprehensive system study report, laying the groundwork for our project.

2.System Analysis

Following the system study, we delve into system analysis, where we meticulously dissect the requirements identified during the initial study phase. Here, we delineate functional aspects and processes, such as user authentication and content search functionalities. Additionally, we delineate non-functional requirements like security and performance, alongside conducting risk analysis to proactively address potential pitfalls. The outcome of this phase is a detailed system analysis report, guiding subsequent development stages.

3.Design

The design phase serves as the blueprint for our e-book application, where we meticulously craft the system's architecture and user interface mock-ups. We delineate database schemas, select appropriate technologies, and outline the development roadmap. By visualizing the system's structure and user interface, we lay the groundwork for a user-friendly and efficient application.

4.Coding and Testing

Subsequently, we enter the coding and testing phase, where the development team translates design specifications into functional code. We rigorously test the application at various levels, including unit testing, integration testing, and system testing, to ensure robustness and reliability. Bugs and issues are identified and rectified promptly, ensuring the application meets quality standards.

5. Implementation

The implementation phase marks the culmination of our efforts, where the fully developed e-book application is deployed in a production environment. User training sessions are conducted to facilitate seamless adoption, accompanied by comprehensive documentation. Performance tests are conducted, and necessary configurations are made to ensure optimal system performance. With thorough validation, the e-book application is officially launched for operational use, marking the successful completion of the project.

9.2 GANNT CHART

9.2 Gantt Chart

A Gantt chart is a vital tool for project management, providing a visual representation of tasks, durations, and timelines. It aids in planning, tracking progress, and managing resources effectively. Below is the Gantt chart for the development of the e-book app, outlining the tasks, durations, and schedule for completion.

Tasks	Duration	February				March				April			
Days		1	2	3	4	1	2	3	4	1	2	3	4
System study	2 weeks												
System analysis	1 weeks												
Design	2 weeks												
Coding and testing	6 weeks												
Implementation	2 weeks												

CHAPTER 10 SYSTEM COST ESTIMATION

10.1 INTRODUCTION

System cost estimation is paramount in project planning for the development of the e-book application, providing a comprehensive assessment of the financial resources required for its successful execution. This phase involves evaluating anticipated expenses associated with various project elements, including development, infrastructure, human resources, and ongoing maintenance. Cost estimation serves as a guiding framework for stakeholders, offering valuable insights into the project's financial feasibility and facilitating informed budget allocation and resource management.

A meticulous cost estimation process entails a systematic analysis of project requirements, scope, and objectives, considering factors such as technology choices, software and hardware components, labour expenses, and potential risks. By thoroughly evaluating these variables, organizations can generate accurate cost projections and allocate resources effectively, thereby minimizing the risk of cost overruns.

The primary goal of system cost estimation for the e-book application is to establish a realistic budget that encompasses all foreseeable expenses while mitigating the risk of financial challenges. A comprehensive and accurate estimation process lays the groundwork for successful project execution, ensuring that financial resources are allocated appropriately to support the development, maintenance, and enhancement of the e-book application.

10.2 LOC BASED ESTIMATION / FUNCTION POINT BASED ESTIMATION

LOC-based estimation involves counting the lines of code in a software project to estimate its size and potential complexity. However, it's essential to understand that LOC-based estimation has its limitations and may not always reflect the actual effort or complexity of a project accurately. The number of lines of code can vary depending on the programming language, coding style, and implementation approach, and it doesn't directly correlate with the overall project effort or quality. This estimation involves counting the lines of code in a software project to estimate its size and potential complexity. To understand that LOC-based estimation has its limitations and it doesn't directly correlate with the overall project effort or quality.

Function Point Based Estimation – Functionalities

a) Login: The e-book app will allow users to register and create accounts using their personal information securely. Users will be authenticated through various methods, ensuring secure access to the application.

b) Search: Users will have the ability to efficiently search for and discover e-books within the app. The search functionality will be robust, allowing users to find books based on various criteria and preferences.

c) View Book: Users can view comprehensive information about selected books from their digital library, including title, author, description, and cover image.

d) PDF Reader: The app will provide a seamless reading experience for PDF documents within the digital library, allowing users to access and read e-books effortlessly.

e) Bookmark: Users can bookmark their progress within a book for easy access and reference, enhancing the reading experience.

f) Text-to-Speech Conversion: The app will offer text-to-speech functionality, allowing users to listen to e-books instead of reading them, thereby enhancing accessibility.

g) Upload Book: Users will have the option to upload their digital books into the system, enriching their digital library with personalized content.

h) Download Book: Users can download e-books from their digital library for offline access, facilitating reading on the go.

These functionalities collectively contribute to the comprehensive nature of the e-book app, enhancing user satisfaction and providing a seamless reading experience.

CHAPTER 11 CONCLUSION

In conclusion, the development and implementation of the e-Book App have marked a significant milestone in the realm of digital reading, offering users a revolutionary platform to explore, access, and manage their digital book collections with unprecedented ease and efficiency. Leveraging modern technologies and adhering to sound software engineering principles, the project has successfully addressed the challenges faced by contemporary readers, providing them with a seamless and enriching reading experience.

Throughout this journey, we have demonstrated the feasibility and effectiveness of a comprehensive e-book management system that caters to the diverse needs and preferences of modern readers. The app's intuitive interface, advanced features such as PDF reading view, text-to-speech capabilities, and seamless integration with Firebase, have collectively contributed to enhancing user engagement and satisfaction.

The success of this project is attributed to the dedication, collaboration, and technical prowess of the project team, whose efforts have propelled the e-Book App from concept to reality. We firmly believe that the e-Book App has the potential to redefine the digital reading landscape, empowering users to embrace digital literature in a more personalized and immersive manner.

As we conclude this project, we recognize the opportunities for future enhancements and iterations. We remain committed to ongoing innovation, embracing emerging technologies, and continuously refining the app to better serve the evolving needs of our users.

In essence, the e-Book App project encapsulates our commitment to innovation, user-centric design, and the relentless pursuit of excellence in delivering solutions that enrich lives and inspire meaningful experiences in the digital era.

CHAPTER 12 FUTURE ENHANCEMENT AND SCOPE OF FURTHER DEVELOPMENT

12.1 INTRODUCTION

The horizon of technological possibilities is ever-expanding, and the e-book application has been engineered with flexibility and continuous improvement in mind. While the current iteration of the system effectively facilitates e-book management, there exists a wide spectrum of opportunities for augmenting its capabilities and enhancing its functionalities. This section delves into potential pathways for future development, highlighting the application's potential to adapt to emerging trends and fulfill evolving user requirements.

By embracing innovation and adopting an iterative approach, we can ensure that the application remains a pertinent and indispensable tool for users, authors, publishers, and administrators. The subsequent sections outline potential directions for enhancement, showcasing the application's capacity to evolve into a comprehensive platform that caters to diverse aspects of digital reading and publishing practices.

This journey of growth offers an exciting exploration of how the e-book application can continue to advance and make a meaningful contribution to the world of digital literature.

12.2 MERITS OF THE SYSTEM

The e-book management system brings forth a multitude of advantages that contribute to its significance and effectiveness in the digital literature domain. These merits underscore the value it offers to users, authors, publishers, and administrators alike, shaping a more streamlined and efficient e-book management process. Here are some of the notable merits of the system:

a) Efficiency Enhancement - The system streamlines the entire e-book management process, reducing manual efforts and administrative complexities. This efficiency enhancement translates into quicker access to desired e-books and optimized resource allocation.

b) User-Friendly Interface - The intuitive user interface ensures a user-friendly experience for all parties involved. Whether its users searching for e-books or authors uploading their works, the system's interface simplifies interactions and reduces learning curves.

c) Enhanced Accessibility - The platform's digital accessibility features ensure that e-books are accessible to users with diverse needs, including those with disabilities. This inclusivity fosters a more diverse and inclusive reading community.

d) Adaptability and Scalability - Designed with adaptability in mind, the system can evolve to accommodate future advancements and changing requirements. Its scalable architecture ensures that it can cater to a growing number of users and e-books.

e) Reduced Environmental Impact - By digitizing various processes, the system significantly reduces the environmental impact associated with traditional print publishing. This reduction in paper usage contributes to environmental sustainability and conservation efforts.

f) Enhanced User Experience - Users benefit from a seamless experience, from searching for e-books to reading them on various devices. This enhanced experience fosters customer satisfaction and loyalty.

g) Contribution to Digital Literacy - Ultimately, the system's facilitation of digital reading contributes to the broader goal of promoting digital literacy and lifelong learning. By providing access to a wide range of e-books, the system empowers users to explore new topics and expand their knowledge.

12.3 LIMITATIONS OF THE SYSTEM

While the e-book management system presents a range of benefits, it is important to acknowledge its limitations and potential challenges. These limitations offer insights into areas that may require further attention or improvement to ensure the system's optimal functionality and effectiveness. Some of the limitations include:

a) Dependency on Technology - The system heavily relies on technology infrastructure, including internet connectivity, server availability, and device compatibility. Any disruptions or technical issues in these components could impact the system's performance and accessibility.

b) Initial Setup Complexity - Implementing the system requires an initial setup phase, including database configuration, server deployment, and user training. This complexity may pose challenges for organizations with limited technical expertise or resources.

c) User Adoption - User acceptance and adoption of the system may vary. Users, particularly authors and publishers, might require training and time to adapt to the new digital publishing workflow. Ensuring smooth user onboarding is crucial for system success.

d) Data Security Concerns - As the system involves the exchange of sensitive information, data security is paramount. Robust measures need to be in place to safeguard user data, prevent unauthorized access, and ensure compliance with data protection regulations.

e) Content Quality and Diversity - The quality and diversity of e-books available in the system may vary, depending on the contributions of authors and publishers. Ensuring a diverse selection of high-quality e-books is essential for maintaining user interest and engagement.

f) Network Reliability - The system's effectiveness relies on stable internet connectivity, particularly during e-book downloads and updates. Network outages or slow connections could hinder seamless access to e-books and system updates.

g) Maintenance and Support - Ongoing maintenance, updates, and technical support are essential for the system's longevity and smooth operation. Adequate resources must be allocated to address potential issues and ensure timely assistance to users.

h) Integration Challenges - Integrating the system with existing digital publishing platforms or e-book distribution channels may require additional development and customization efforts. Ensuring seamless integration is crucial for maximizing the system's utility and effectiveness.

12.4 FUTURE ENHANCEMENT OF THE SYSTEM

The e-book management system has been designed to provide a robust and efficient solution for digital reading and publishing. As the digital literature landscape continues to evolve and technology advances, there are several avenues for future enhancements and the expansion of the system's capabilities. Some potential areas for improvement and further development include:

a) Enhanced Recommendation Engine - Implementing an advanced recommendation engine can personalize e-book recommendations based on user preferences, reading history, and genre interests. This can enhance the user's reading experience and promote e-book discovery.

b) Collaborative Reading Features - Introducing collaborative reading features such as virtual book clubs, discussion forums, and shared annotations can foster community engagement and

interaction among users. This can create a vibrant reading community and encourage social sharing of e-books.

c) Interactive E-books - Enhancing e-books with interactive multimedia elements such as videos, audio clips, and interactive quizzes can create immersive reading experiences for users. This can appeal to a wider audience, including children, students, and lifelong learners.

d) Blockchain Integration - Exploring blockchain technology for e-book distribution and copyright management can enhance transparency, security, and royalty tracking. Blockchain-based solutions can streamline royalty payments, prevent piracy, and protect authors' intellectual property rights.

e) Augmented Reality (AR) Integration - Integrating augmented reality technology into e-books can provide interactive and immersive reading experiences. Users can explore 3D models, visualize story elements, and interact with characters in real-time, enhancing engagement and comprehension.

f) Voice Assistant Integration - Integrating voice assistant technology such as Amazon Alexa or Google Assistant can enable hands-free e-book navigation and control. Users can listen to e-books, control playback, and interact with the system using voice commands, enhancing accessibility and convenience.

These potential enhancements reflect the dynamic nature of the digital literature landscape and the ongoing advancements in technology. By strategically planning and implementing these improvements, the e-book management system can continue to empower users, authors, publishers, and administrators, fostering a vibrant digital reading ecosystem and contributing to a more literate and connected society.

CHAPTER 13 ANNEXURE

13.1 ORGANIZATION PROFILE

Devopte IT Solutions, is a dynamic and innovative educational institution conveniently located near Kuzhivelippady, Kochi a bustling hub in the heart of the city. Established with a mission to empower learners of all ages and backgrounds, our center offers a diverse range of educational programs and services to cater to the diverse learning needs of our community.

At Devopte IT Solutions, we are committed to providing high-quality education that fosters holistic development, critical thinking, and lifelong learning. Our experienced and dedicated team of educators is passionate about creating a nurturing and stimulating environment where students can excel academically and personally.

13.2 DOCUMENT GLOSSARY, FIGURES, TABLES

13.2.1 ANNEXURE 1: GLOSSARY

Glossary:

User Authentication: The process of verifying the identity of a user attempting to access the e-book app, typically done through secure login credentials.

Firebase: A mobile and web application development platform developed by Google, used for building and managing backend services for the e-book app.

Flutter: An open-source UI software development kit (SDK) developed by Google, used for building natively compiled applications for mobile, web, and desktop from a single codebase.

Dart: The programming language used in conjunction with Flutter for developing the e-book app's frontend components.

Frontend: The client-side components of the e-book app that users interact with, including the user interface and user experience design.

Backend: The server-side components and logic of the e-book app responsible for processing data, managing databases, and performing system operations.

PDF Reader: A module within the e-book app that allows users to view and read PDF documents from their digital library.

Text-to-Speech: A module within the e-book app that converts text-based content into audible speech, enhancing accessibility for users.

Firebase Authentication: The Firebase service used for authenticating users and securing access to the e-book app.

Firebase Firestore: The cloud-based NoSQL database service provided by Firebase, used for storing and syncing data for the e-book app.

Bookmark: A feature within the e-book app that allows users to mark and save specific pages or sections of books for future reference.

Search Functionality: The capability within the e-book app to search for specific books or content based on user input.

Download: The functionality within the e-book app that enables users to retrieve and save books from their digital library for offline access.

Upload: The feature within the e-book app that allows users to add their own digital books to their library.

System Maintenance: Ongoing activities aimed at ensuring the optimal performance and reliability of the e-book app, including updates and bug fixes.

System Security: Measures implemented to protect user data, ensure system integrity, and prevent unauthorized access to the e-book app.

Usability Testing: Evaluation of the e-book app's user-friendliness and ease of use through user interactions and feedback.

Cloud Hosting: Storing and managing e-book app data and resources on remote servers accessed via the internet.

Functionality: The range of tasks, features, and operations that the e-book app can perform to fulfill user needs and requirements.

13.2.2 ANNEXURE 2: TABLES

2.1 TABLE NAME: USER TABLE

Purpose: The User Table is designed to list users along with their respective credentials.

FIELD	TYPE	SIZE	CONSTRAINT	DESCRIPTION
uid	varchar	20	Primary key	To store user id
email	varchar	20	Not null	To store user email
name	varchar	20	Not null	To store name of user
photoUrl	varchar	100	Not null	To store user photo

2.2 TABLE NAME: AUTHENTICATION TABLE

Purpose: The Authentication Table maintains records of user authentication activities.

FIELD	TYPE	SIZE	CONSTRAINT	DESCRIPTION
uid	varchar	20	Primary key	To store user id
email	varchar	20	Not Null	To store user email
createdate	date	20	Not null	To account create data
signindate	varchar	100	Not null	To sign in app date

2.3 TABLE NAME: BOOK TABLE

Purpose: The Book Table serves to store essential credentials related to books.

FIELD	TYPE	SIZE	CONSTRAINT	DESCRIPTION
Did	varchar	10	Primary key	To store book document id
title	varchar	20	Not null	To store book title
description	varchar	100	Not null	To store book description
pages	int	10	Not null	To store book page number
language	varchar	20	Not null	To store book language
author	varchar	20	Not null	To store author of book
aboutAuthor	varchar	100	Not null	To store about the author
bookurl	varchar	50	Not null	To store book url for pdf
Coverurl	varchar	100	Not null	To store url for cover image
Category	varchar	20	Not null	To store category of book

2.4 TABLE NAME: USER BOOK TABLE

Purpose: The User Book Table provides detailed insights into users' interactions with books**.**

FIELD	TYPE	SIZE	CONSTRAINT	DESCRIPTION
Did	varchar	10	Primary key,	To store book id
uid	varchar	20	Foreign key	To store id
title	varchar	20	Not null	To store book title
description	varchar	100	Not null	To store book description
pages	int	10	Not null	To store book page number
language	varchar	20	Not null	To store book language
author	varchar	20	Not null	To store author of book
aboutAuthor	varchar	100	Not null	To store about the author
bookurl	varchar	50	Not null	To store book url for pdf
Coverurl	varchar	100	Not null	To store url for cover image
Category	varchar	20	Not null	To store category of book

13.3.3 ANNEXURE 3: DIAGRAMS

3.2 USE CASE DIAGRAM

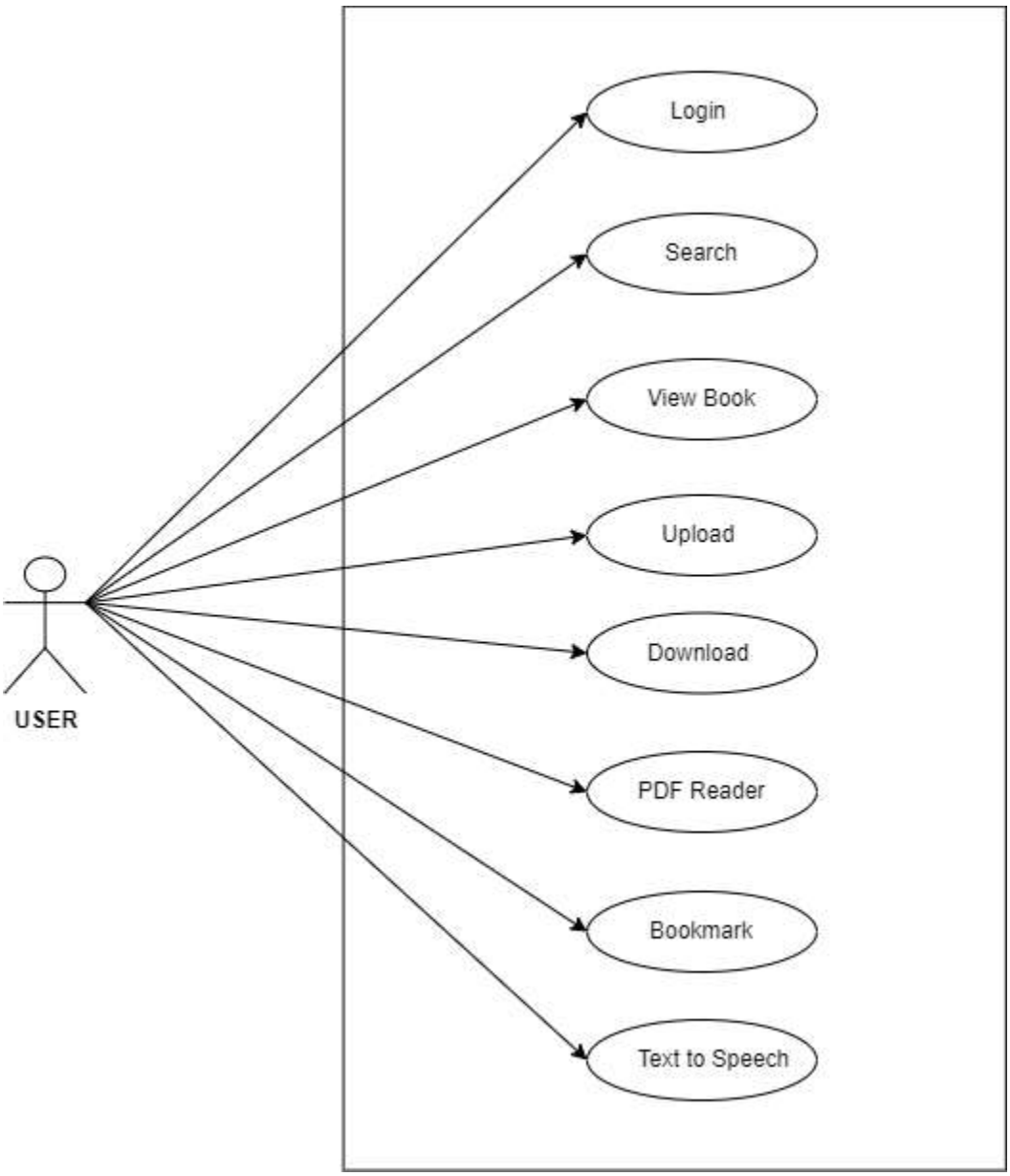


Figure 3.1.1

3.2 ACTIVITY DIAGRAM

SEARCH ACTIVITY

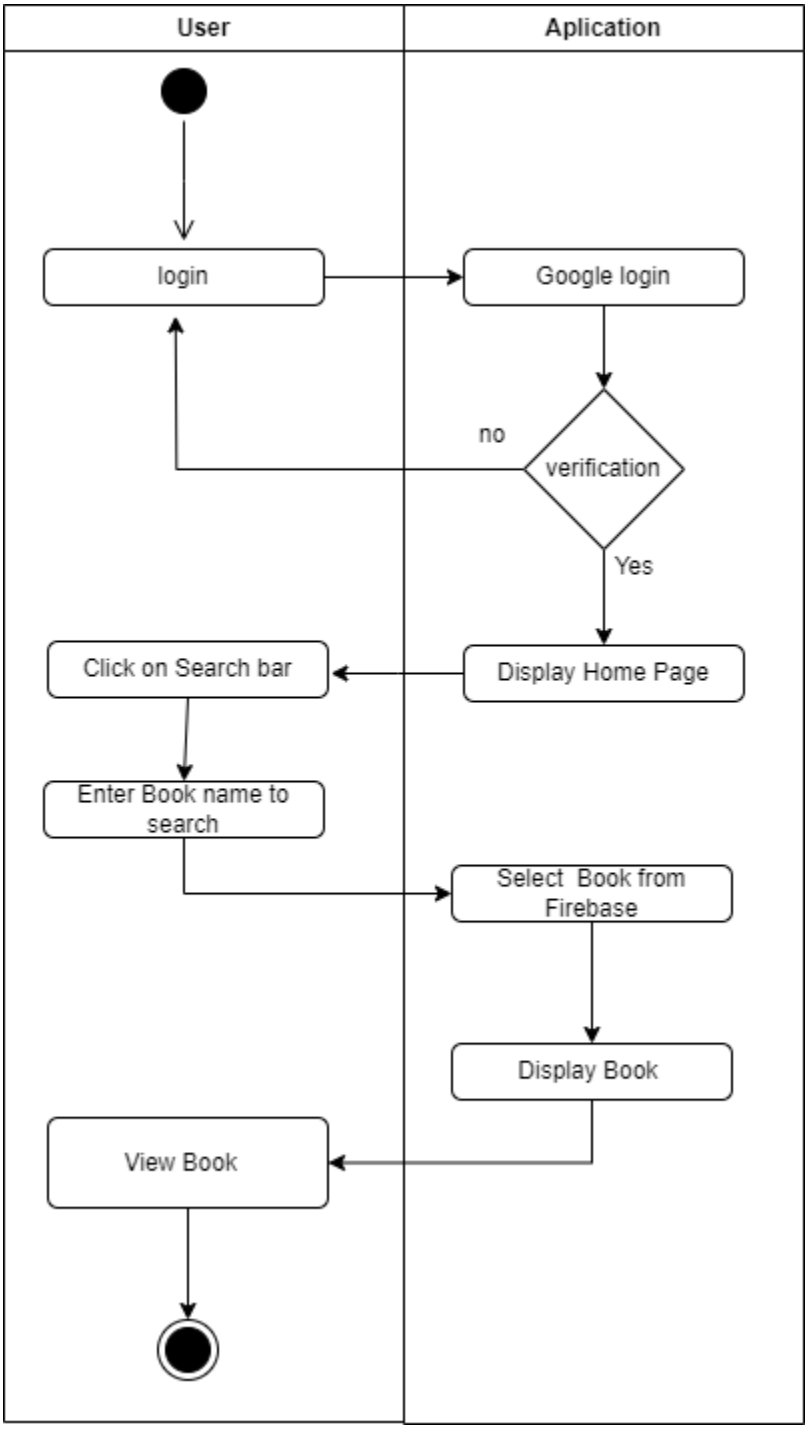


Figure 3.2.1

VIEW BOOK ACTIVITY

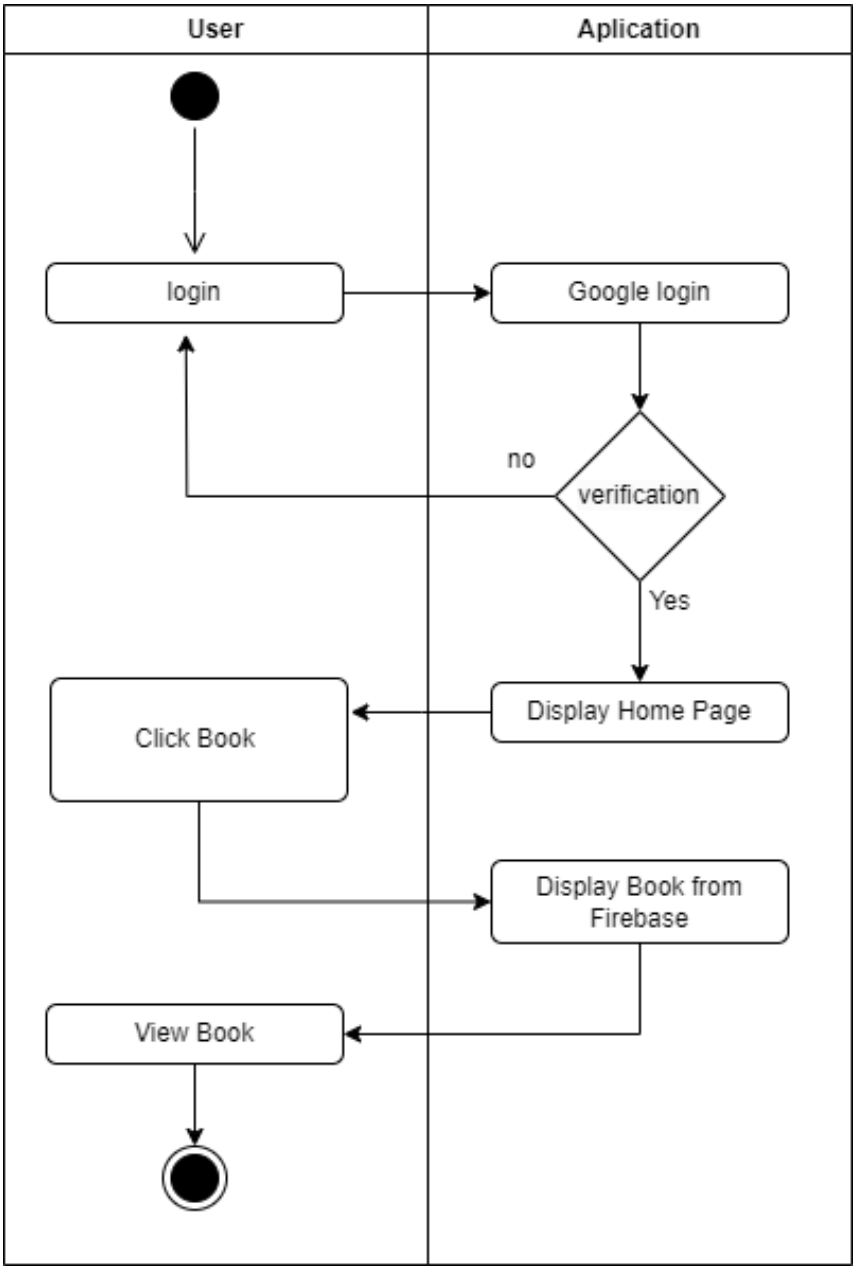


Figure 3.2.2

UPLOAD ACTIVITY

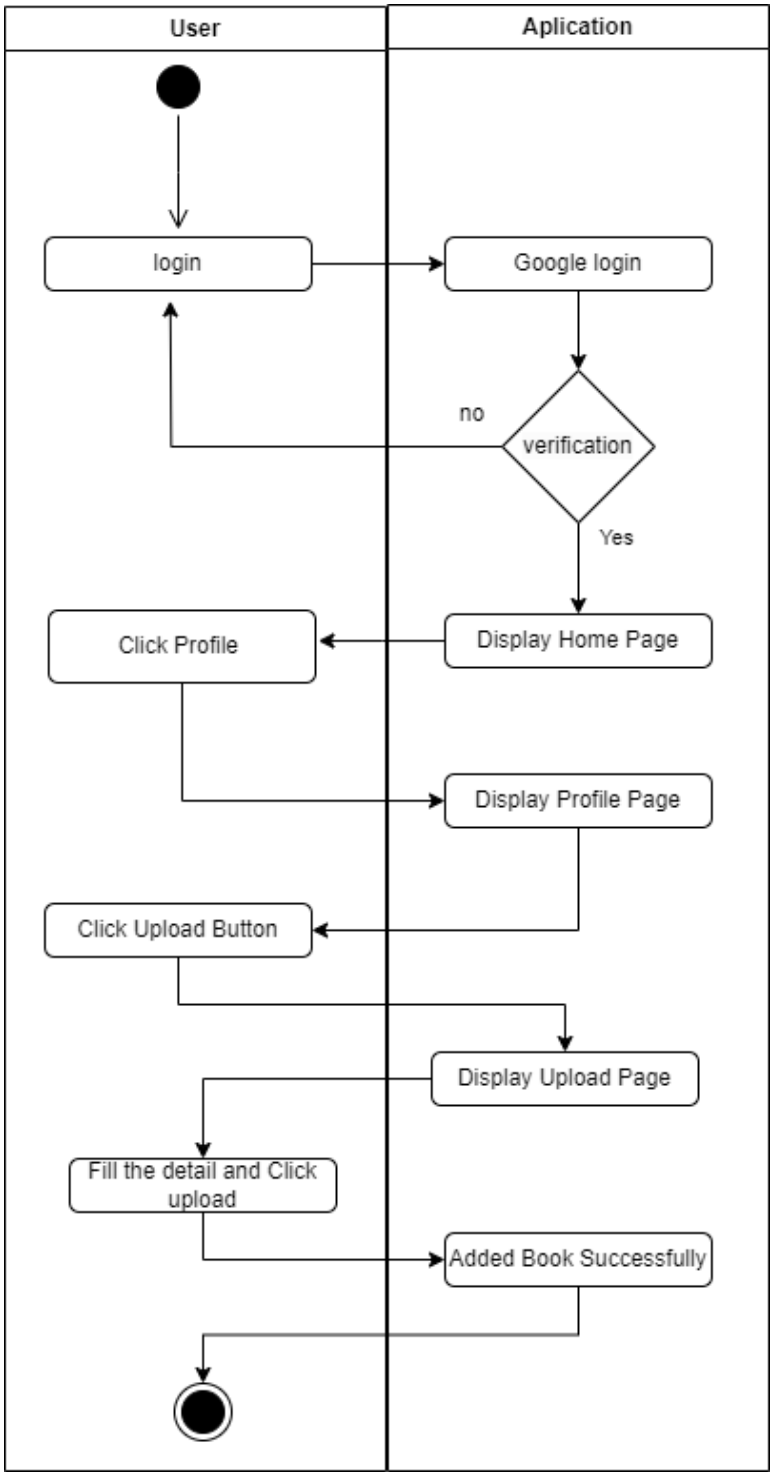


Figure 3.2.3

DOWNLOAD ACTIVITY

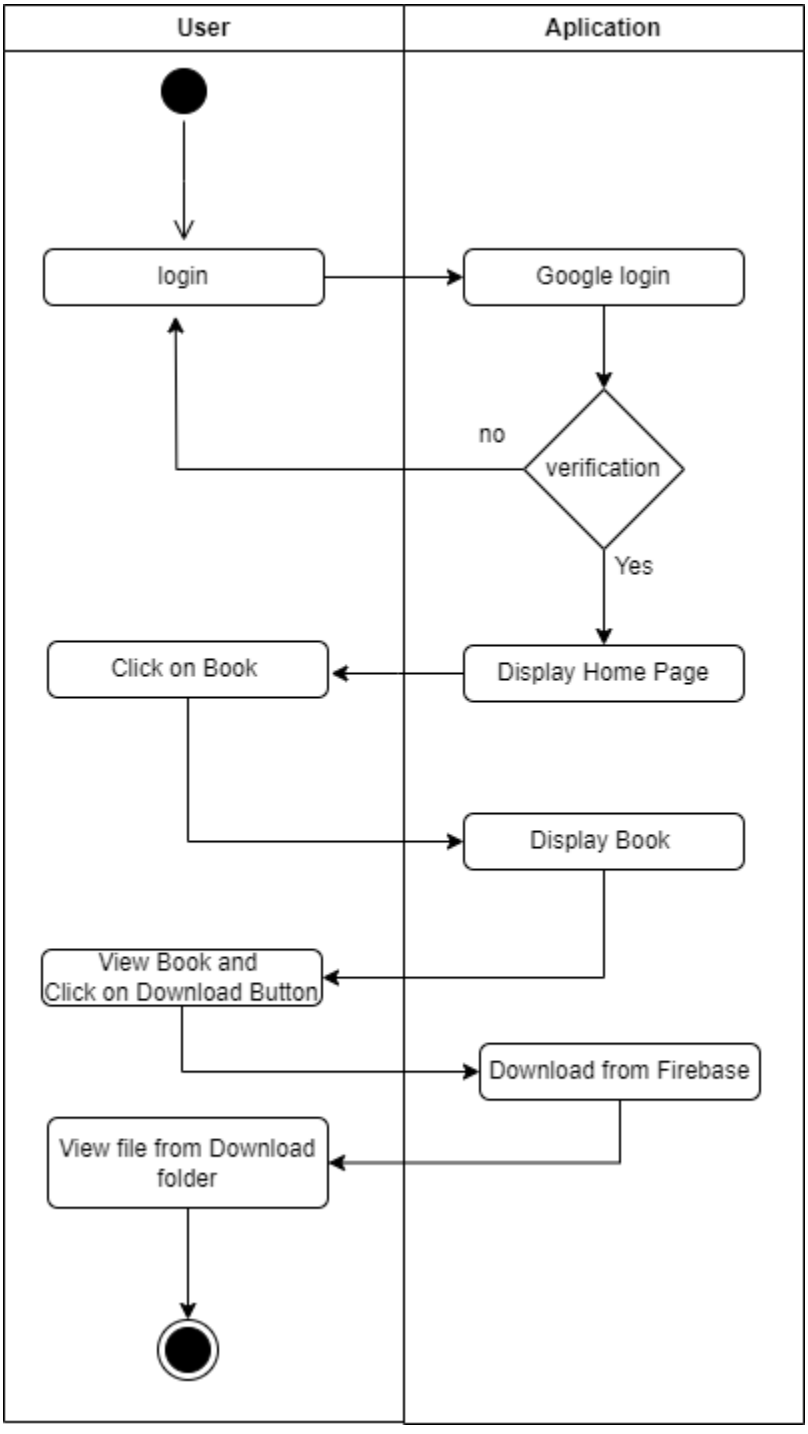


Figure 3.2.4

PDF READER ACTIVITY

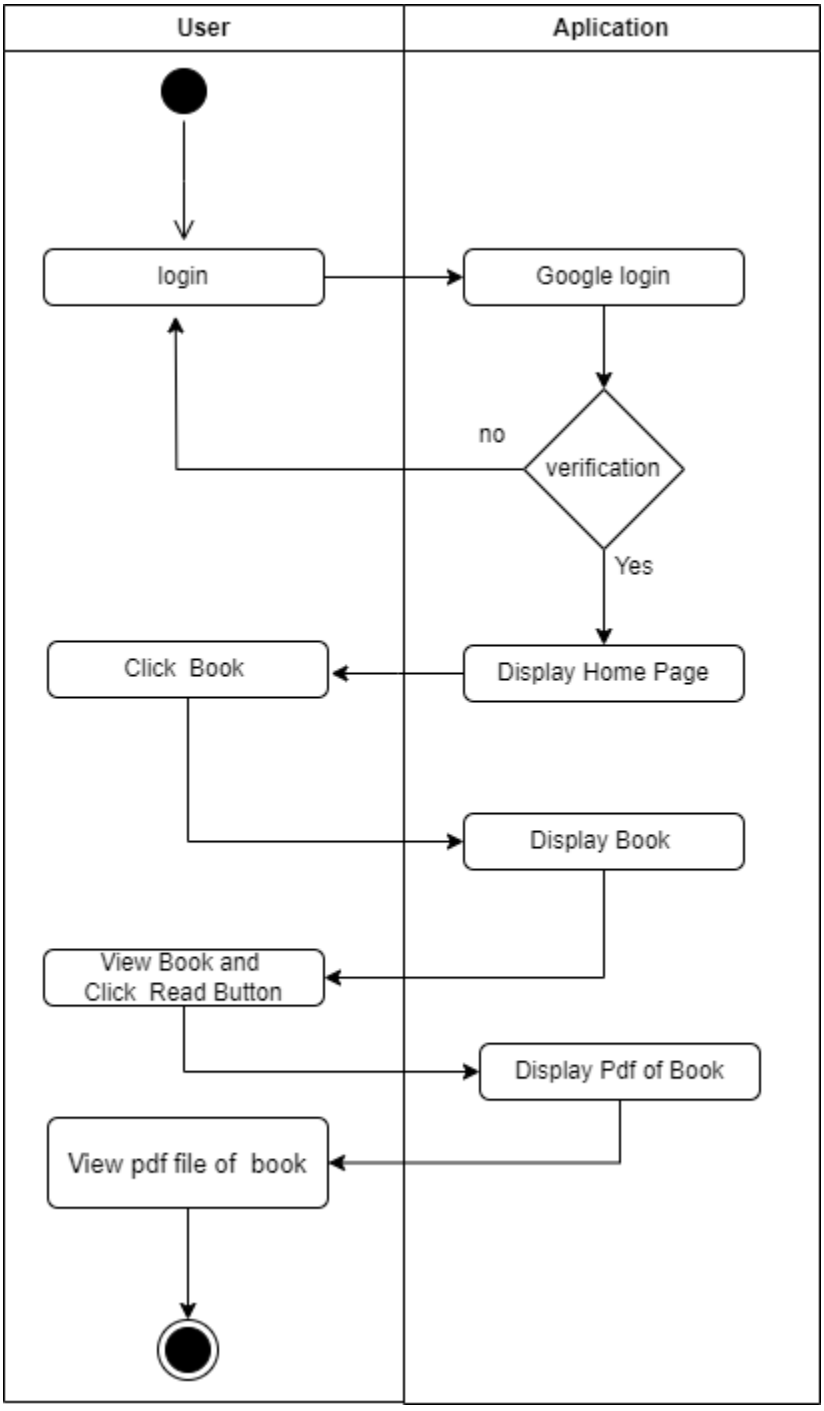


Figure 3.2.5

BOOKMARK ACTIVITY

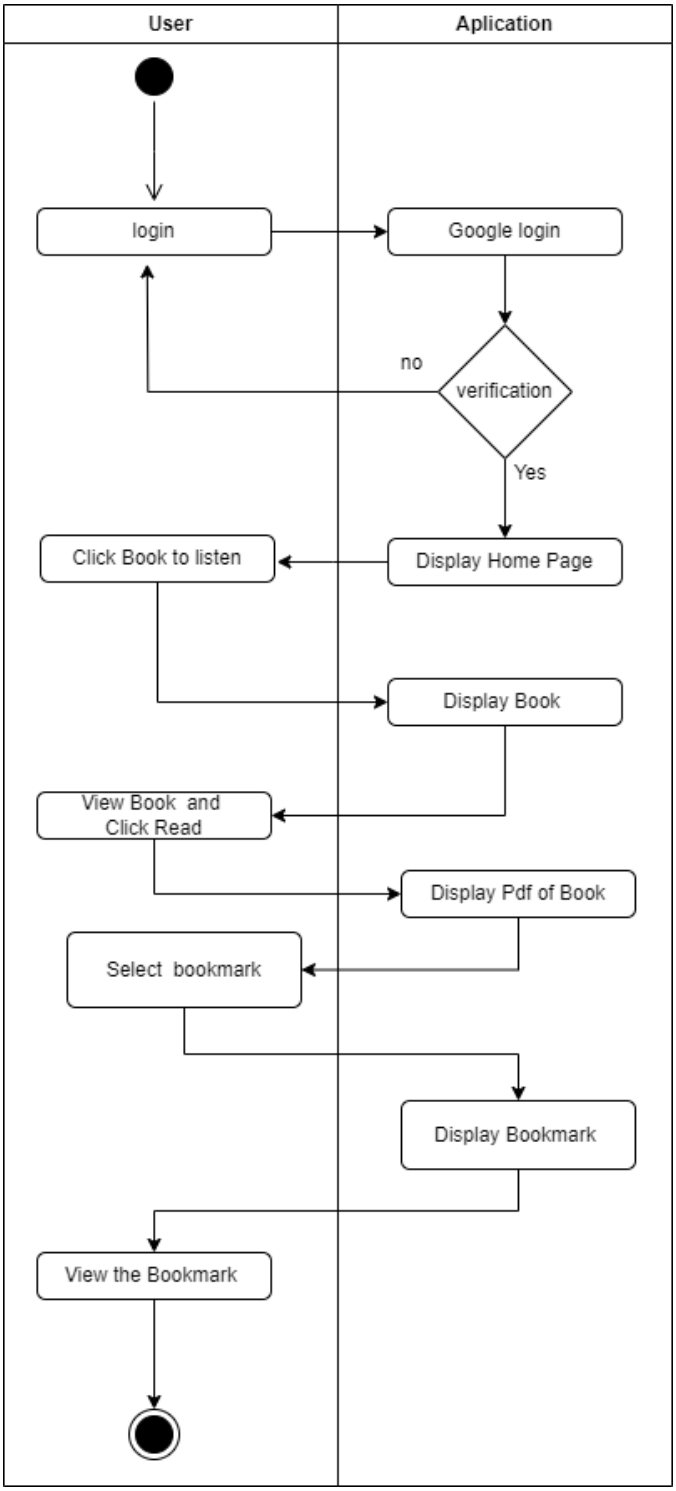


Figure 3.2.6

TEXT TO SPEECH ACTIVITY

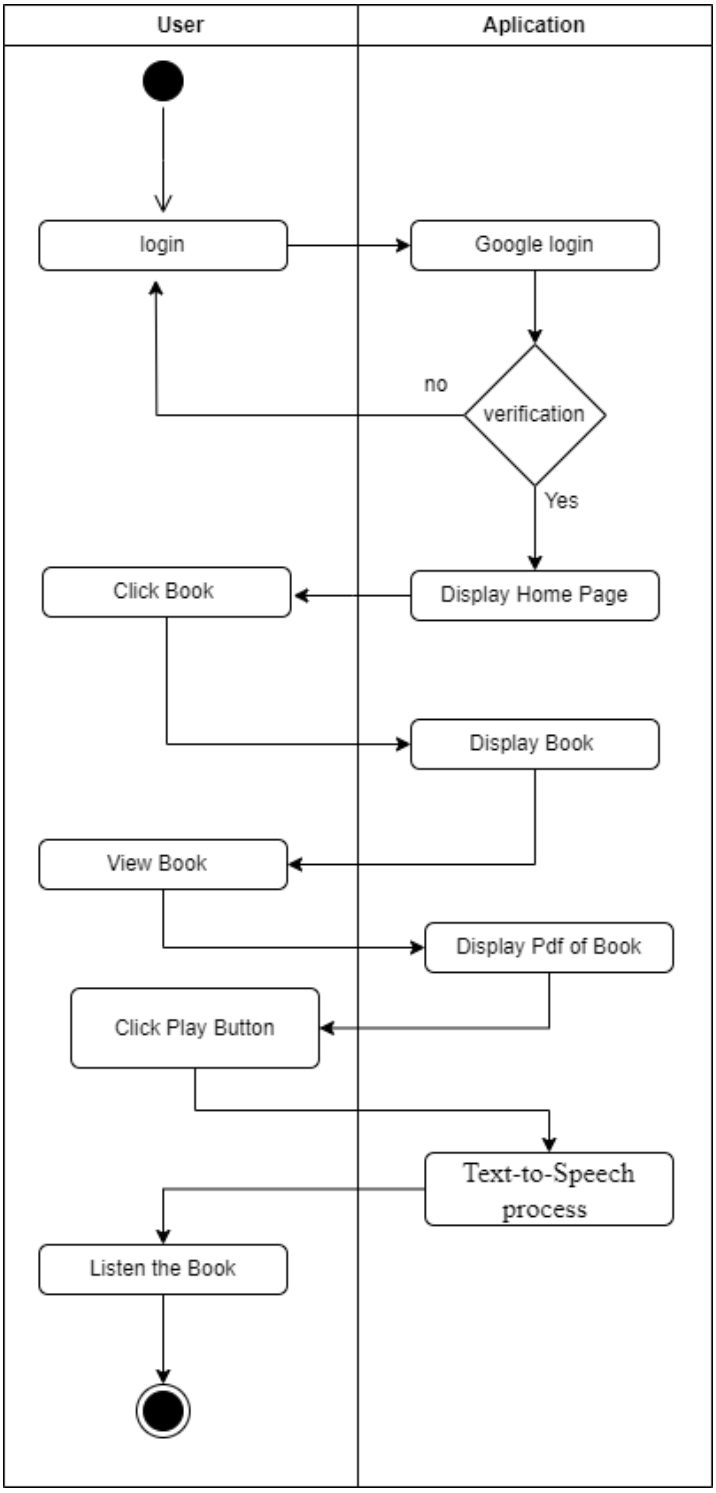


Figure 3.2.7

3.3 CLASS DIAGRAM

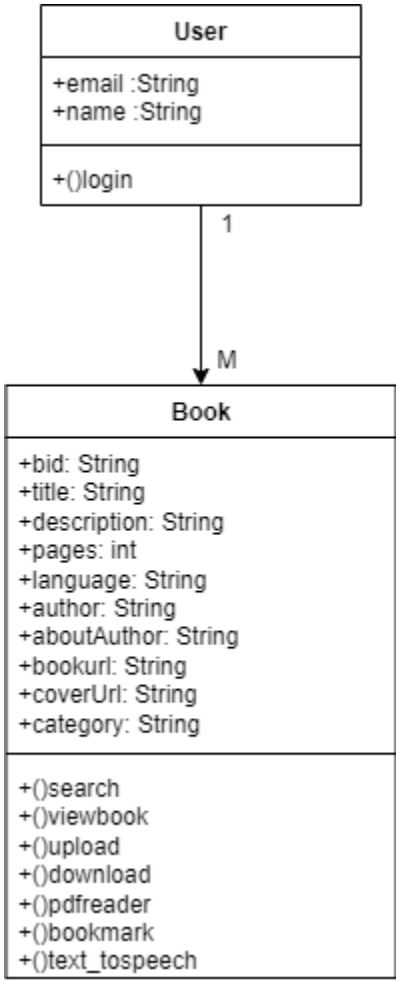


Figure 3.3.1

3.4 SEQUENCE DIAGRAM

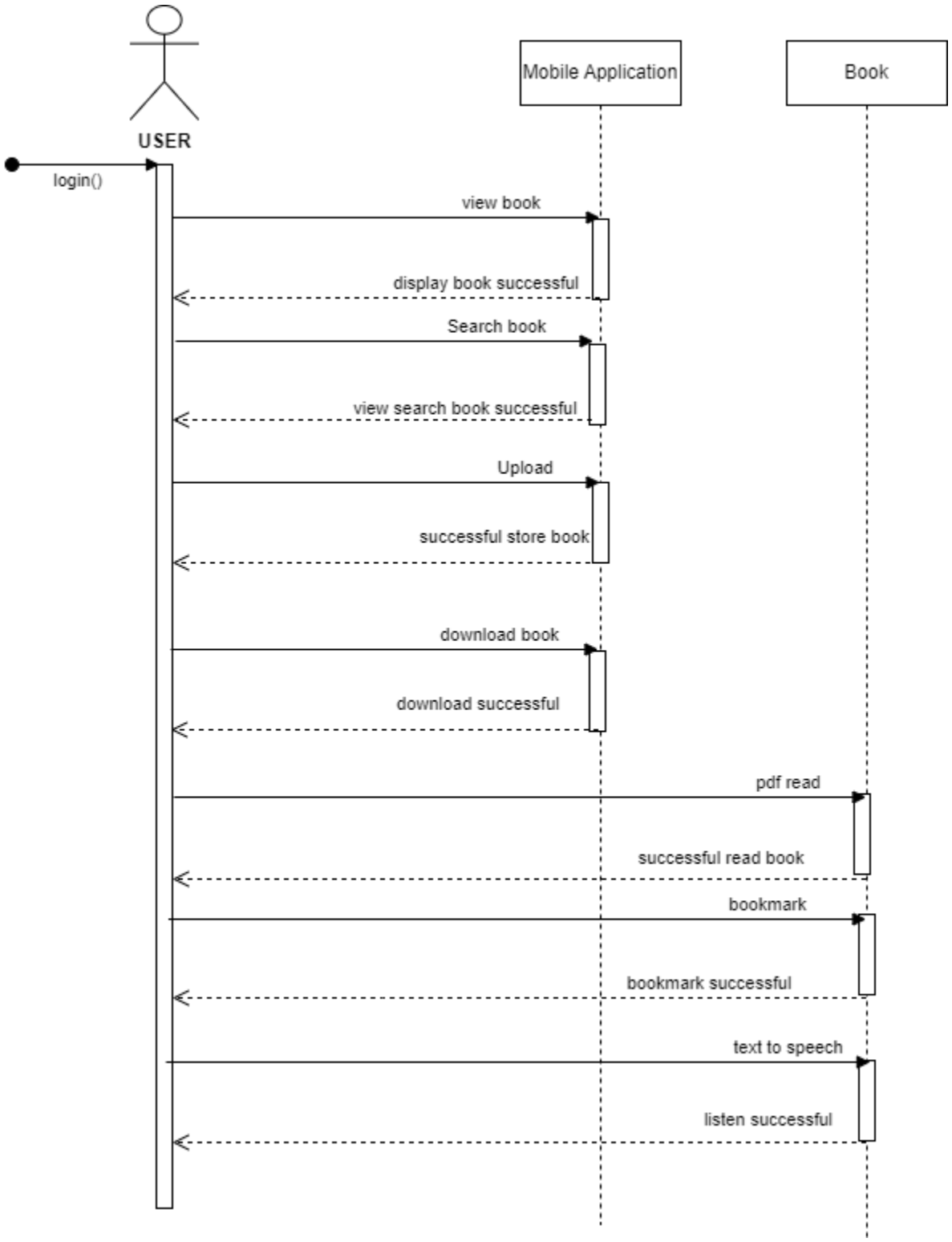


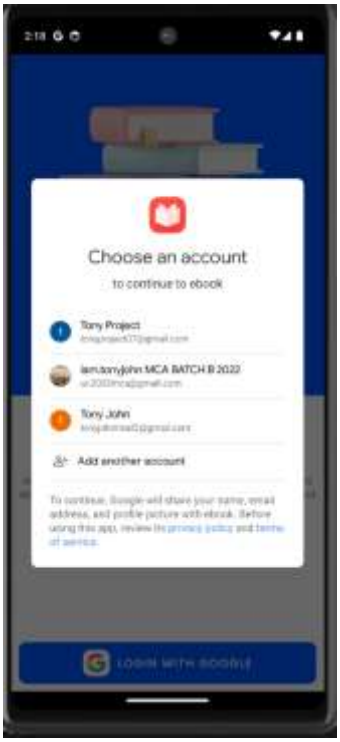
Figure 3.4.1

13.2.4 ANNEXURE 4: FIGURES

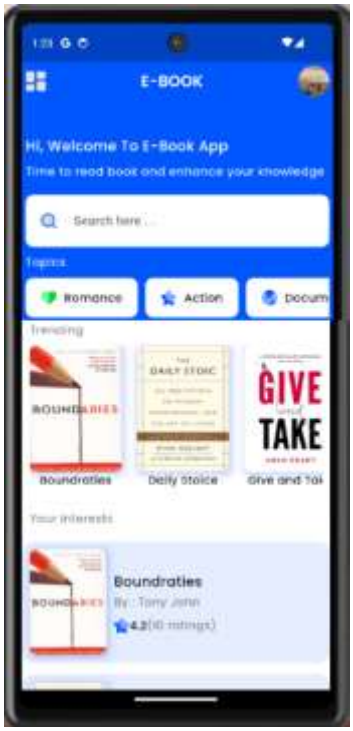
SPLACE PAGE:

WELCOME PAGE:

LOGIN PAGE:



HOME PAGE:



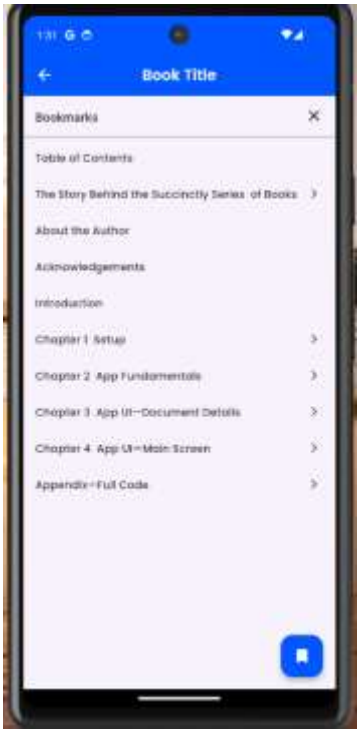
PROFILE PAGE:

BOOK PAGE:

UPLOAD PAGE:

PDF PAGE:

BOOKMARK PAGE:



DOWNLOAD PAGE:

13.2.5 ANNEXURE 5: CODING

main.dart

```
import 'package:ebook/Config/Themes.dart';
import 'package:ebook/Pages/SplacePage/SplacePage1.dart';
import 'package:ebook/Pages/WelcomePage/WelcomePage.dart';
import 'package:ebook/firebase_options.dart';
import 'package:firebase_core/firebase_core.dart' ;
import 'package:flutter/material.dart';
import 'package:get/get_navigation/src/root/get_material_app.dart';
void main()
async {
WidgetsFlutterBinding.ensureInitialized();
await Firebase.initializeApp(
 options: DefaultFirebaseOptions.currentPlatform,
);
  runApp(const MyApp());
}
class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'E Book',
      theme: lightTheme,
    // home: const WelcomePage(),
     home:const SplacePage1(),
```

```
  );
  }
}
```

Splace.dart

```
import 'package:lottie/lottie.dart';
class SplacePage1 extends StatelessWidget {
  const SplacePage1({super.key});
  @override
  Widget build(BuildContext context) {
    SplaceController splaceController = Get.put(SplaceController());
    return Scaffold(
      backgroundColor: Theme.of(context).colorScheme.primary,
      body: Center(
        child:  Lottie.asset("Assets/Animation/Anim1.json")),
    );
  }
}
```

Welcome.dart

```
import 'package:ebook/Components/PrimaryButton.dart';
import 'package:ebook/Controller/AuthController.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:get/get.dart';
class WelcomePage extends StatelessWidget {
 const WelcomePage({super.key});
  @override
```

```
Widget build(BuildContext context) {
  AuthController authController = Get.put(AuthController());
  return Scaffold(
    body: Column(
      children: [
        Container(
          height: 500,
          padding: const EdgeInsets.all(20),
          color: Theme.of(context).colorScheme.primary,
          child: Row(mainAxisAlignment: MainAxisAlignment.center, children: [
            Expanded(
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  const SizedBox(height: 50),
                  Image.asset("Assets/Images/book.png",width: 380,  ),
                  const SizedBox(height: 60),
                  Text("E - Book App",
                    style: Theme.of(context).textTheme.headlineLarge?.copyWith(
                      color: Theme.of(context).colorScheme.background,),
                  ),
                  const SizedBox(height: 10),
                  Flexible(
                    child: Text(
                    "Here you can find best book for you and you can also read book and listen book ",
                      textAlign: TextAlign.center,
                      style: Theme.of(context).textTheme.labelSmall?.copyWith(
```

```
                color: Theme.of(context).colorScheme.background,),
              ),),],),)]),),
        Padding(
          padding: const EdgeInsets.all(10),
          child: Column(
            children: [
              const SizedBox(height: 30),
              Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Text("Disclaimer",style: Theme.of(context).textTheme.labelMedium,),
                ],),
              const SizedBox(height: 10),
              Row(
                children: [
                  Flexible(
                    child: Text(
                      "In the realm of e-book apps, placeholder text serves to illustrate the layout and design of a digital publication, devoid of substantive content. The text provided within this application is for demonstrative purposes only and does not reflect the actual content of any books or materials available.",
                      textAlign: TextAlign.center,
                      style: Theme.of(context).textTheme.labelSmall,
                    ),),],),],),),
        const Spacer(),
        Padding(
          padding: const EdgeInsets.all(10),
          child: PrimaryButton(btnName: "LOGIN WITH GOOGLE",
```

```
                        ontap: () {
                        authController.loginWithEmail();
                              },),),],),
   ); }}
```

HomePage.dart

```
import 'package:ebook/Components/BookCard.dart';
import 'package:ebook/Components/BookTitle.dart';
// ignore: unused_import
import 'package:ebook/Config/Colors.dart';
import 'package:ebook/Controller/BookController.dart';
import 'package:ebook/Models/Data.dart';
import 'package:ebook/Pages/BookDetails/BookDetails.dart';
import 'package:ebook/Pages/Homepage/Widgets/AppBar.dart';
import 'package:ebook/Pages/Homepage/Widgets/CategoryWidget.dart';
import 'package:ebook/Pages/Homepage/Widgets/MyInputeTextField.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:flutter_svg/flutter_svg.dart';
import 'package:get/get.dart';
class HomePage extends StatelessWidget {
const HomePage({super.key});
  @override
  Widget build(BuildContext context) {
    BookController bookController=Get.put(BookController());
   bookController.getUserBook();
    return Padding(padding:const EdgeInsets.only(left: 5,right: 5),
     child:Scaffold(
      body: SingleChildScrollView(
```

```
      child: Column(
       children:
      [
       Container(
        color: Theme.of(context).colorScheme.primary,
        height: 390,
        child: Expanded(
         child: Row(
          crossAxisAlignment: CrossAxisAlignment.center,
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
         children: [
          Expanded(
           child: Column(
            children: [
           const   SizedBox(height:50),
       const       HomeAppBar(),
          const     SizedBox(height:50),
              Padding(padding: const  EdgeInsets.only(left: 5,right: 5),
           child:
              Row(
               children: [Text(
                 "Hi, Welcome To E-Book App",style:
Theme.of(context).textTheme.bodyLarge?.copyWith(color:
Theme.of(context).colorScheme.background),
               )],
              ),),
          const     SizedBox(height:10),
              Padding(padding: const EdgeInsets.only(left: 5,right: 5),
         child:
```

```
                Row(
                 children: [
                   Flexible(child:Text(
                   "Time to read book and enhance your knowledge",style:
Theme.of(context).textTheme.labelMedium?.copyWith(color:
Theme.of(context).colorScheme.background),
                 ),),],
                ),),
             const     SizedBox(height:20),
              const     MyInputTextField(),
               const    SizedBox(height:10),
                 Padding(padding: EdgeInsets.only(left: 5,right: 5),
            child:
                 Row(
                  children:
[Text("Topics",style:Theme.of(context).textTheme.labelMedium?.copyWith(
                    color: Theme.of(context).colorScheme.background,
                  ),),],),),
                const  SizedBox(height: 10),
                 Padding(
                padding: const EdgeInsets.only(left: 5,right: 5), // Adjust the value as needed
                child: SingleChildScrollView(
                      scrollDirection: Axis.horizontal,
                      child: Row(
          children: categoryData.map((e) => CategoryWidget(iconPath: e["icon"]!, btnName:
e["lebel"]!,)).toList(),
                        ),),),],),
              ),],),), ),
        const SizedBox(height: 10,),
            Padding
```

```
    (
padding: const EdgeInsets.all(10),
child: Column(children:
 [ Row (children: [Text("Trending",
 style: Theme.of(context).textTheme.labelMedium,),
 ],
 ),
    const SizedBox(height: 10,),
 SingleChildScrollView(
  scrollDirection: Axis.horizontal,
   child: Obx(() =>  Row(
     children: bookController.bookData.
    map((e) => BookCard(
      title: e.title!,
      coverUrl: e.coverUrl!,
      ontap :(){
        Get.to(BookDetails(book: e,));
      },
     ),
     ).toList(),),
  ),),],),),
const SizedBox(height: 10,),
    Padding
    (padding: const EdgeInsets.all(10),
child: Column(children:
 [ Row (children: [Text("Your Interests",
 style: Theme.of(context).textTheme.labelMedium,),
 ],),],),),
```

```
      const SizedBox(height: 10,),
      Obx(() => Column(
    children: bookController.bookData.
    map((e) => BookTitle(
      title: e.title!,
      coverUrl: e.coverUrl!,
      author: e.auther!,
      rating: e.rating!,
      totalRating: e.numberofRating!,
      ontap :(){
            Get.to(BookDetails(book: e,));
          },),).toList()))
],),),
),); }}
```

BookDetail.dart

```
import 'package:ebook/Components/BackButton.dart';
import 'package:ebook/Config/Colors.dart';
import 'package:ebook/Models/BookModel.dart';
import 'package:ebook/Pages/BookDetails/BookActionBtn.dart';
import 'package:ebook/Pages/BookDetails/HeaderWidget.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/rendering.dart';
import 'package:flutter/widgets.dart';
import 'package:flutter_svg/flutter_svg.dart';
class BookDetails extends StatelessWidget {
  final BookModel book;
  const BookDetails({super.key,required this.book});
```

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: SingleChildScrollView(
      child: Column(
        children: [
          SingleChildScrollView(
            child: Container(
              padding: const EdgeInsets.all(15),


              color: primaryColor,
              child:Row(
                children: [
                  Expanded(child: HeaderWidget(
                    coverUrl: book.coverUrl!,
                    title: book.title!,
                    author: book.auther!,
                    description: book.descrption!,
                    rating: book.rating!,
                    pages: book.pages.toString(),
                    language: book.language.toString(),
                  ),
                  ),],


              )
            ),
          ),
      const SizedBox(height: 20,),
```

```
        Padding(
      padding: const EdgeInsets.all(10),
      child:Column(crossAxisAlignment: CrossAxisAlignment.start  ,children: [Row(
        children: [
          Text("About Book", style: Theme.of(context).textTheme.bodyMedium),
        ],
      ),
       const SizedBox(height: 8,),
      Row(
        children: [
          Flexible(child: Text(book.descrption!,textAlign: TextAlign.justify,  style:
Theme.of(context).textTheme.labelSmall)),
        ],
      ),
      const SizedBox(height: 12,),
      Row(
        children: [
          Text("About Author", style: Theme.of(context).textTheme.bodyMedium),
        ],
      ),
       const SizedBox(height: 8,),
      Row(
        children: [
          Flexible(child: Text(book.aboutAuthor!,textAlign: TextAlign.justify , style:
Theme.of(context).textTheme.labelSmall)),
        ],
      ),
       const SizedBox(height: 15,),
        BookActionBtn(bookUrl:book.bookurl!),
```

```
      ],)),],),
    ),);
 }}
```

UploadBook.dart

```
import 'package:ebook/Components/BackButton.dart';
import 'package:ebook/Components/MutiLineTextFormFeild.dart';
import 'package:ebook/Components/MyTextFormField.dart';
import 'package:ebook/Config/Colors.dart';
import 'package:ebook/Controller/BookController.dart';
import 'package:ebook/Controller/PdfController.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
class AddNewBookPage extends StatelessWidget {
 const AddNewBookPage({super.key});

 @override
 Widget build(BuildContext context) {
  TextEditingController controller = TextEditingController();
  PdfController pdfController = Get.put(PdfController());
  BookController bookController = Get.put(BookController());
  return Scaffold(
   body: SingleChildScrollView(
    child: Column(
     children: [
      Container(
       // height: 500,
       padding: const EdgeInsets.symmetric(vertical: 40, horizontal: 20),
       color: Theme.of(context).colorScheme.primary,
```

```
child:
  Row(mainAxisAlignment: MainAxisAlignment.center, children: [
 Expanded(
  child: Column(
   mainAxisAlignment: MainAxisAlignment.center,
   children: [
    const SizedBox(height: 20),
    Row(
     crossAxisAlignment: CrossAxisAlignment.center,
     mainAxisAlignment: MainAxisAlignment.spaceBetween,
     children: [
      const MyBackButton(),
      Text(
       "ADD NEW BOOK",
       style: Theme.of(context)
          .textTheme
          .bodyLarge
          ?.copyWith(
           color:
              Theme.of(context).colorScheme.background,
          ),
      ),
      const SizedBox(width: 70)
     ],
    ),
    const SizedBox(height: 60),
    InkWell(
      onTap: () {
```

```
                bookController.pickImage();
              },
              child: Obx(
                () => Container(
                  height: 190,
                  width: 150,
                  decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(20),
                    color: Theme.of(context).colorScheme.background,
                  ),
                  child: Center(
                    child: bookController.isImageUploading.value
                        ? const CircularProgressIndicator(
                            color: primaryColor,
                          )
                        : bookController.imageUrl.value == ""
                            ? Image.asset(
                                "Assets/Icons/addImage.png")
                            : ClipRRect(
                                borderRadius:
                                    BorderRadius.circular(10),
                                child: Image.network(
                                  bookController.imageUrl.value,
                                  fit: BoxFit.cover,
                                ),),),),),),
            const SizedBox(height: 20),
          ],),)]),),
      Padding(
```

```
padding: const EdgeInsets.all(10),
child: Column(children: [
  Row(
    children: [
      Expanded(
        child: Obx(
          () => Container(
            padding: const EdgeInsets.all(16),
            decoration: BoxDecoration(
              color: bookController.pdfUrl.value == ""
                  ? Theme.of(context).colorScheme.primary
                  : Theme.of(context).colorScheme.background,
              borderRadius: BorderRadius.circular(10),
            ),
            child: bookController.isPdfUploading.value
                ? Center(
                    child: CircularProgressIndicator(
                      color: Theme.of(context).colorScheme.background,
                    ),)
                : bookController.pdfUrl.value == ""
                    ? InkWell(
                        onTap: () {
                          bookController.pickPDF();
                        },
                        child: Row(
                          mainAxisAlignment:
                              MainAxisAlignment.center,
                          children: [
```

```
                Image.asset(
                  "Assets/Icons/upload.png"),
                const SizedBox(width: 8),
                Text(
                 "Book PDF",
                 style: Theme.of(context)
                   .textTheme
                   .bodyLarge
                   ?.copyWith(
                     color: Theme.of(context)
                       .colorScheme
                       .background,
                   ),),],
            ),)
          : InkWell(
            onTap: () {
             bookController.pdfUrl.value = "";
            },
            child: Row(
             mainAxisAlignment:
               MainAxisAlignment.center,
             children: [
              Image.asset(
               "Assets/Icons/delete.png",
               width: 20,
              ),
              const SizedBox(width: 8),
              Text(
```

```
                        "Delete Pdf",
                        style: Theme.of(context)
                            .textTheme
                            .bodyLarge
                            ?.copyWith(
                              color: Theme.of(context)
                                  .colorScheme
                                  .primary,
                            ),),],),
                ),),),),
  ],),
const SizedBox(height: 10),
MyTextFormField(
  hintText: "Book title",
  icon: Icons.book,
  controller: bookController.title,
),
const SizedBox(height: 10),
MutiLineTextFormField(
    hintText: "Book Description",icon: Icons.description_outlined,
    controller: bookController.des),
const SizedBox(height: 10),
MyTextFormField(
  hintText: "Author Name",
  icon: Icons.person_2,
  controller: bookController.auth,
),
const SizedBox(height: 10),
```

```
MyTextFormField(
  hintText: "About Author",
  icon: Icons.person_2_outlined,
  controller: bookController.aboutAuth,
),
const SizedBox(height: 10),
Row(
  children: [
    Expanded(
      child: MyTextFormField(
        hintText: "Pages",
       isNumber: true,
        icon: Icons.book,
        controller: bookController.pages,
      ),
    ),
const SizedBox(width: 10),
    Expanded(
      child: MyTextFormField(
        hintText: "Language",
        icon: Icons.language,
        controller: bookController.language,
      ),),],),
const SizedBox(height: 20),
Row(
  children: [
    Expanded(
      child: Container(
```

```
                    padding: const EdgeInsets.all(16),
                    decoration: BoxDecoration(
                      color: Colors.red,
                       borderRadius: BorderRadius.circular(10),
                       border: Border.all(
                         width: 2,
                         color: Colors.red,
                       )),
                    child: Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                       const Icon(Icons.close,color: Colors.white, ),
                       const SizedBox(width: 8),
                       Text( "CANCLE",style:
Theme.of(context).textTheme.bodyLarge?.copyWith(color: Colors.white,),),
                            ],
                    ),
                  ),
                ),
                const SizedBox(width: 10),
                Expanded(
                    child: Obx(
                  () => Container(
                    padding: const EdgeInsets.all(16),
                    decoration: BoxDecoration(
                      color: Theme.of(context).colorScheme.primary,
                      borderRadius: BorderRadius.circular(10),
                    ),
                    child: bookController.isPdfUploading.value
```

```
                ? const Center(
                    child: CircularProgressIndicator(),
                  )
                : InkWell(
                    onTap: () {
                      bookController.createBook();
                    },
                    child: Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: [
                        Icon(
                          Icons.upload_sharp,
                          color: Theme.of(context)
                              .colorScheme
                              .background,
                        ),
                        const SizedBox(width: 8),
                        Text(
                          "POST",
                          style: Theme.of(context)
                              .textTheme
                              .bodyLarge
                              ?.copyWith(
                                color: Theme.of(context)
                                    .colorScheme
                                    .background,
                              ),),],),),
            ),)), ],),],]),),
```

```
        ],),  ),
    ); }}
```

Bookpage.dart

```
// ignore: file_names, unused_import
import 'package:ebook/Components/BackButton.dart';
import 'package:ebook/Controller/PdfController.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:syncfusion_flutter_pdfviewer/pdfviewer.dart';

class BookPage extends StatelessWidget {
  final String bookUrl;
  const BookPage({super.key, required this.bookUrl});

 @override
  Widget build(BuildContext context) {
    PdfController pdfController=Get.put(PdfController());
    return Scaffold(
      appBar: AppBar(
        leading: BackButton(color: Theme.of(context).colorScheme.background,),
        backgroundColor: Theme.of(context).colorScheme.primary,
        title: Text("Book Title",style:
Theme.of(context).textTheme.headlineMedium?.copyWith(color:
Theme.of(context).colorScheme.background),),
        centerTitle: true,
      ),
      //bookmark
      floatingActionButton: FloatingActionButton(onPressed: () {
        pdfController.pdfViewerKey.currentState?.openBookmarkView();
```

```
    },
    child: Icon(Icons.bookmark,color: Theme.of(context).colorScheme.background,),),
    //pdfview
    body: SfPdfViewer.network(
      bookUrl,
      key: pdfController.pdfViewerKey,
    ),
   );
 }}
```

pdfController.dart

```
import 'dart:io';
import 'package:file_picker/file_picker.dart';
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:get/get.dart';
import 'package:syncfusion_flutter_pdfviewer/pdfviewer.dart';

class PdfController extends GetxController {
  final GlobalKey<SfPdfViewerState> pdfViewerKey = GlobalKey();
}
```

imageController.dart

```
import 'package:firebase_storage/firebase_storage.dart';
import 'package:get/get.dart';
import 'package:image_picker/image_picker.dart';
import 'package:uuid/uuid.dart';
class ImageController extends GetxController {
  ImagePicker imagePicker = ImagePicker();
```

```
  void pickImage() async {
    final XFile? image =
        await imagePicker.pickImage(source: ImageSource.gallery);
    if (image != null) {
      var uuid = const Uuid();
      var fileName = uuid.v1();
      var response = FirebaseStorage.instance.ref().child("Image/$fileName");
      print(response.storage.ref().getDownloadURL());
    }
  }
}
```

bookController.dart

```
import 'dart:convert';
import 'dart:io';
import 'dart:typed_data';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:ebook/Config/Messages.dart';
import 'package:ebook/Models/BookModel.dart';
import 'package:file_picker/file_picker.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter/widgets.dart';
import 'package:get/get.dart';
import 'package:image_picker/image_picker.dart';
import 'package:uuid/uuid.dart';

class BookController extends GetxController {
  TextEditingController title = TextEditingController();
```

```
TextEditingController des = TextEditingController();
TextEditingController auth = TextEditingController();
TextEditingController aboutAuth = TextEditingController();
TextEditingController pages = TextEditingController();
TextEditingController language = TextEditingController();
ImagePicker imagePicker = ImagePicker();
final storage = FirebaseStorage.instance;
final db = FirebaseFirestore.instance;
final fAuth = FirebaseAuth.instance;
RxString imageUrl = "".obs;
RxString pdfUrl = "".obs;
int index = 0;
RxBool isImageUploading = false.obs;
RxBool isPdfUploading = false.obs;
RxBool isPostUploading = true.obs;
var bookData = RxList<BookModel>();
var currentUserBooks = RxList<BookModel>();

@override
void onInit() {
  // TODO: implement onInit
  super.onInit();
  getAllBooks();
}
void getAllBooks() async {
  bookData.clear();
 successMessage("Book Get Fun");
  var books = await db.collection("Books").get();
```

```
  for (var book in books.docs) {
    bookData.add(BookModel.fromJson(book.data()));
  }
}
void getUserBook() async {
  currentUserBooks.clear();
  var books = await db
      .collection("userBook")
      .doc(fAuth.currentUser!.uid)
      .collection("Books")
      .get();
  for (var book in books.docs) {
    currentUserBooks.add(BookModel.fromJson(book.data()));
  }
}
void pickImage() async {
  isImageUploading.value = true;
  final XFile? image =
      await imagePicker.pickImage(source: ImageSource.gallery);
  if (image != null) {
    print(image.path);
    uploadImageToFirebase(File(image.path));
  }
  isImageUploading.value = false;
}
void uploadImageToFirebase(File image) async {
  var uuid = const Uuid();
  var filename = uuid.v1();
```

```
  var storageRef = storage.ref().child("Images/$filename");
  var response = await storageRef.putFile(image);
  String downloadURL = await storageRef.getDownloadURL();
  imageUrl.value = downloadURL;
  print("Download URL: $downloadURL");
  isImageUploading.value = false;
 }
 void createBook() async {
  isPostUploading.value = true;
  var newBook = BookModel(
   id: "$index",
   title: title.text,
   descrption: des.text,
   coverUrl: imageUrl.value,
   bookurl: pdfUrl.value,
   auther: auth.text,
   aboutAuthor: aboutAuth.text,
   pages: int.parse(pages.text),
   language: language.text,
   rating: "",
  );

  await db.collection("Books").add(newBook.toJson());
  addBookInUserDb(newBook);
  isPostUploading.value = false;
  title.clear();
  des.clear();
  aboutAuth.clear();
```

```
  pages.clear();
  language.clear();
  auth.clear();
  imageUrl.value = "";
  pdfUrl.value = "";
  successMessage("Book added to the db");
  getAllBooks();
  getUserBook();
 }
 void pickPDF() async {
  isPdfUploading.value = true;
  FilePickerResult? result = await FilePicker.platform.pickFiles(
   type: FileType.custom,
   allowedExtensions: ['pdf'],
  );
  if (result != null) {
   File file = File(result.files.first.path!);
   if (file.existsSync()) {
    Uint8List fileBytes = await file.readAsBytes();
    String fileName = result.files.first.name;
    print("File Bytes: $fileBytes");
    final response =
        await storage.ref().child("Pdf/$fileName").putData(fileBytes);
    final downloadURL = await response.ref.getDownloadURL();
    pdfUrl.value = downloadURL;
    print(downloadURL);
   } else {
    print("File does not exist");
```

```
    }
   } else {
    print("No file selected");
   }
   isPdfUploading.value = false;
  }
  void addBookInUserDb(BookModel book) async {
   await db
      .collection("userBook")
      .doc(fAuth.currentUser!.uid)
      .collection("Books")
      .add(book.toJson());
  }
}
```

authController.dart

```
import 'package:ebook/Config/Messages.dart';
import 'package:ebook/Pages/Homepage/HomePage.dart';
import 'package:ebook/Pages/WelcomePage/WelcomePage.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:get/get.dart';
import 'package:google_sign_in/google_sign_in.dart';
class AuthController extends GetxController {
  RxBool isLoading = false.obs;
  final auth = FirebaseAuth.instance;
  void loginWithEmail() async {
   isLoading.value = true;
   try {
    final GoogleSignInAccount? googleUser = await GoogleSignIn().signIn();
```

```
      final GoogleSignInAuthentication? googleAuth =
          await googleUser?.authentication;
      final credential = GoogleAuthProvider.credential(
        accessToken: googleAuth?.accessToken,
        idToken: googleAuth?.idToken,
      );
      await auth.signInWithCredential(credential);
     successMessage('Login Success');
      Get.offAll(HomePage());
    } catch (ex) {
      print(ex);
   errorMessage('Error! Something went wrong try again');
  }
    isLoading.value = false;
  }
  void signout() async {
    await auth.signOut();
    successMessage('Logout');
    Get.offAll(WelcomePage());
  }
}
```

Bookcard.dart

```
import 'package:flutter/material.dart';
class BookCard extends StatelessWidget {
  final String coverUrl;
  final String title;
  final VoidCallback ontap;
  const BookCard({super.key, required this.coverUrl, required this.title, required this.ontap});
```

```
@override
Widget build(BuildContext context) {
 return  Padding(
   padding: const EdgeInsets.only(right: 20),
   child: InkWell(
     onTap: ontap,
     child: SizedBox(
       width: 120,
       child: Column(
     children: [
      Container(
        decoration: BoxDecoration(boxShadow:[
        BoxShadow(color:  Theme.of(context).colorScheme.primary.withOpacity(0.3)
        ,spreadRadius: 1,blurRadius: 8,offset:const Offset(2,2),
        ),
        ],
        ),
        child: ClipRRect(
          borderRadius: BorderRadius.circular(10),
          child: Image.network(coverUrl,width: 120,),
        ),
      ),
      const SizedBox(width: 10,),
      Text(title,textAlign: TextAlign.center,maxLines: 1,
      style: Theme.of(context).textTheme.bodyMedium,)
     ],
     ),
```

```
        ),
      ),
    )
 ;
  }
}
```

Booktitle.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_svg/svg.dart';

class BookTitle extends StatelessWidget {
  final String title;
  final String coverUrl;
  final String author;
  final String rating;
  final int totalRating;
  final VoidCallback ontap;

  const BookTitle({super.key,
  required this.title,
  required this.coverUrl,
  required this.author,
  required this.rating,
  required this.totalRating, required this.ontap,
  });

  @override
```

```
 Widget build(BuildContext context) {
  return   Padding(
    padding: const EdgeInsets.only(bottom: 15),
    child: InkWell(
      onTap: ontap,
      child: Container(
            padding: const EdgeInsets.all(10),
           decoration: BoxDecoration(  color:
Theme.of(context).colorScheme.primary.withOpacity(.1),borderRadius:
BorderRadius.circular(15),),
            child:Row(
              children: [

            Container(
              decoration: BoxDecoration(boxShadow:[
              BoxShadow(color:  Theme.of(context).colorScheme.primary.withOpacity(0.3)
              ,spreadRadius: 1,blurRadius: 8,offset:const Offset(2,2),
              ),
              ],
              ),
              child: ClipRRect(
                borderRadius: BorderRadius.circular(10),
                child: Image.network(coverUrl,width: 100,),

              ),
            ),
            const SizedBox(width: 10),
            Expanded(
              child: Column(
```

```
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
                Text(title,maxLines: 2,style: Theme.of(context).textTheme.bodyLarge,),
                const SizedBox(height: 4,),
                Text("By : $author",style: Theme.of(context).textTheme.labelMedium),
               const SizedBox(height: 10),
                Row(children: [
                 SvgPicture.asset("Assets/Icons/star.svg"),
                 Text(rating,style: Theme.of(context).textTheme.bodyMedium,),
                  Text("($totalRating ratings)",style: Theme.of(context).textTheme.labelMedium,),
                ],)
          ],))
            ],
          ) ,
        ),
    ),
  );
 }
}
```

Primrybutton.dart

```
import 'package:flutter/material.dart';
class PrimaryButton extends StatelessWidget {
 final String btnName;
 final VoidCallback ontap;
 const PrimaryButton({super.key, required this.btnName,required this.ontap});
@override
 Widget build(BuildContext context) {
  return InkWell(
```

```
    onTap: ontap,
    child: Container(
     height: 55,
     padding: EdgeInsets.all(10),
     decoration: BoxDecoration(
      color: Theme.of(context).colorScheme.primary,
      borderRadius: BorderRadius.circular(10),
     ),
     child: Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
       Container(
        padding: EdgeInsets.all(5),
        decoration: BoxDecoration(
         color: Theme.of(context).colorScheme.background,
         borderRadius: BorderRadius.circular(10),
        ),
        child: Image.asset("Assets/Icons/google.png"),
       ),
       SizedBox(width: 10),
       Text(
        btnName,
        style: Theme.of(context).textTheme.bodyMedium?.copyWith(
            color: Theme.of(context).colorScheme.background,
            letterSpacing: 1.5,
           ),  ),
      ], ),),);
}}
```

13.3 REFERENCES

BOOKS:

1. "Flutter for Beginners: An introductory guide to building cross-platform mobile applications with Flutter" by Albert Johanson, 1st Edition, Publisher: Packt Publishing.
2. "Firebase Essentials - Android Edition: Learn to Build Real-Time, Scalable Web and Mobile Apps with Firebase" by Neil Smyth, 1st Edition, Publisher: Techtopia Press.
3. "Dart Programming Language: A Comprehensive Beginner's Guide to Learn Dart Programming from A-Z" by Ryan Turner, 1st Edition, Publisher: Independently published.

WEBSITES:

1. Flutter Documentation: https://flutter.dev/docs, Accessed on [Current Date]
2. Firebase Documentation: https://firebase.google.com/docs, Accessed on [Current Date]
3. Dart Documentation: https://dart.dev/guides, Accessed on [Current Date]