PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100-ft Ring Road, Bengaluru – 560 085, Karnataka, India

Project Report
on

Read Smart

Submitted by

RISHAV KUMAR - (PES1PG22CA168)

Feb. 2024 – June 2024
under the guidance of

Guide Details

Mr. Santosh Katti

Assistant Professor

Department of Computer Applications,
PESU, Bengaluru – 560085

FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER APPLICATIONS
PROGRAM – MASTER OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project entitled

Title of the project

is a bonafide work carried out by

RISHAV KUMAR-PES1PG22CA168

in partial fulfillment for the completion of Capstone Project Phase-2 work in the Program of Study MCA under rules and regulations of PES University, Bengaluru during the period Feb. 2024 – June 2024. The project report has been approved as it satisfies the academic requirements of 4th semester MCA.

Internal Guide	Chairperson	Dean- Faculty of Engineering & Technology
Santosh katti, Asst. Professor, Department of Computer Applications PES University Bengaluru - 560085	Dr. Veena S, Professor Department of Computer Applications PES University Bengaluru - 560085	Dr. B K Keshavan professor PES University Bengaluru - 560085

DECLARATION

I, **RISHAV KUMAR,** bearing **PES1PG22CA168** hereby declare that the Capstone project Phase-2 entitled, ***Read Smart,*** is an original work done by me under the guidance of **Mr. Santosh Katti,** *Designation*, PES University, and is being submitted in partial fulfillment of the requirements for completion of 4th Semester course in the Program of Study **MCA**. All corrections/suggestions indicated for internal assessment have been incorporated in the report.

PLACE:

DATE:

RISHAV KUMAR

Acknowledgment

I take great pleasure in expressing my sincere gratitude to all those who have guided me and supported me to successfully complete this project.

I would like to express my sincere gratitude to the Vice Chancellor of PES University, Dr. J Suryaprasad and Chairperson Dr. Veena S, who gave me an opportunity to go ahead with this project.\

I am grateful to my guide, Mr. P. Sreenivas,Asst. Professor, Department of Computer Applications, who has been my source of inspiration and provided me with guidance, encouragement and support, during the course of the project.

TABLE OF CONTENTS

ABSTRACT

“Read Smart revolutionizes learning by immersive and interactive learning through document reading .

The result it will give are as follows: - increased student engagement , enhanced learning comprehension , efficient assessment process, real time progress tracking

INTRODUCTION

Read Smart is an advanced educational platform built using the MERN stack, combining MongoDB, Express.js, React, and Node.js to deliver a seamless learning experience. The front-end is developed with React, ensuring a responsive and interactive user interface. Tailwind CSS is employed for styling, providing a modern and customizable design with utility-first classes.

MongoDB serves as the database, efficiently managing user data, PDFs, quizzes, and progress tracking. Express.js handles server-side logic and routes, while Node.js powers the back-end, ensuring robust performance and scalability. HTML and JavaScript are used throughout for structuring content and adding dynamic features.

Together, these technologies create a powerful and user-friendly platform where teachers can upload educational materials and create quizzes, and students can access content, complete quizzes, and track their progress effectively.

1.1 PROJECT DESCRIPTION

Read Smart is an innovative educational platform designed to enhance the learning process through interactive content and assessments. Built with the MERN stack, the platform leverages MongoDB for efficient data management, Express.js for streamlined server operations, React for a dynamic user interface, and Node.js for robust back-end functionality.

Teachers can upload PDFs and create tailored quizzes to evaluate student understanding. Students access these materials and complete the quizzes, with a unique feature that requires them to score over 50% to unlock the next set of content. This approach ensures that learners grasp each topic thoroughly before progressing.

Styled with Tailwind CSS, the platform offers a modern and intuitive design. HTML and JavaScript underpin the interactive elements, making the user experience both engaging and effective. *Read Smart* seamlessly integrates these technologies to create a comprehensive and user-friendly educational tool that supports both teaching and learning.

1.2 PROBLEM DEFINITION

Traditional PDF documents often result in passive learning experiences.
Traditional assessments may not fully capture a student's understanding.
Maintaining student engagement and motivation in the learning process can be challenging.
Creating and grading assessments can be time-consuming for educators.

1.3 PROPOSED SOLUTION

Read Smart offers a solution to several key challenges in traditional education. To tackle the issue of passive learning, the platform integrates interactive quizzes directly with PDF content, transforming static documents into engaging, active learning experiences. This approach ensures that students are not just reading but actively applying their knowledge. Traditional assessments often fall short in capturing the depth of a student's understanding, but *Read Smart* employs a variety of quiz formats to provide a more comprehensive evaluation. The platform also keeps students motivated and focused by requiring them to score above 50% before accessing new content, thereby sustaining their engagement throughout the learning process. For educators, *Read Smart* reduces the time and effort involved in creating and grading assessments with automated tools, allowing them to concentrate on teaching and supporting their students. In essence, *Read Smart* streamlines and enriches the educational experience for both learners and instructors.

1.4 PURPOSE

The purpose of the project is to create an interactive educational platform where teachers can upload Word documents, and students can access and engage with the content. The system generates questions based on the document, assesses student answers, and provides reference material for further learning, fostering a dynamic and personalized learning experience

1.5 SCOPE

The project scope involves creating a platform for teachers to upload educational Word documents. Students can access these documents, triggering interactive questions based on the content. The system evaluates student answers, offers reference material for further learning, and ensures scalability, performance, security, and user feedback

2. LITERATURE SURVEY

2.1 DOMAIN SURVEY

In the evolving landscape of educational technology, there is a growing need to enhance how students interact with learning materials and assessments. Traditional methods often result in passive learning experiences where students merely read documents without actively engaging with the content. Additionally, conventional assessments may not fully capture the depth of a student's understanding or provide timely feedback.

A common challenge in education is maintaining student engagement and motivation. Traditional learning tools can sometimes fail to keep students interested, especially if they lack interactive elements. Furthermore, educators face significant hurdles with the time-consuming nature of creating and grading assessments, which can detract from their primary focus of teaching.

The rise of interactive learning platforms addresses these issues by integrating engaging content and assessments. These solutions aim to make learning more dynamic and responsive, offering tools that help both students and educators overcome traditional limitations. By incorporating interactive quizzes and progress tracking, such platforms not only foster deeper understanding but also streamline the assessment process, making it more efficient for educators and more motivating for students.

Read Smart is positioned within this domain to address these specific needs by combining interactive content with automated assessment tools, enhancing both the learning experience and educational efficiency.

2.2 RELATED WORK

In recent years, the field of educational technology has seen significant advancements aimed at transforming traditional learning experiences. Platforms like *Khan Academy* and *Coursera* have pioneered online learning by offering a vast array of courses with integrated assessments, providing students with flexible learning options. These platforms typically offer video lectures and quizzes, but often lack the interactive engagement needed for deep learning.

Similarly, systems such as *Quizlet* and *Duolingo* leverage interactive quizzes and gamification to boost student engagement. *Quizlet* offers flashcards and practice tests, while *Duolingo* integrates language learning with gamified challenges. While these tools enhance engagement, they sometimes fall short in providing comprehensive learning materials or detailed feedback on performance.

Another notable example is *Edmodo*, which blends classroom management with educational resources, allowing teachers to share content and assessments. It supports student-teacher interaction but can be limited in terms of content integration and automated grading.

Emerging platforms like *Socrative* and *Nearpod* aim to bridge these gaps by offering real-time assessments and interactive content. *Socrative* focuses on formative assessments and instant feedback, while *Nearpod* provides interactive lessons and live assessments. Both platforms enhance interactivity but may still require additional tools for comprehensive content delivery and student progress tracking.

Read Smart builds on these advancements by integrating interactive PDFs with customized quizzes and automated progress tracking. This approach addresses the limitations of existing tools by combining engaging content with effective assessment methods, thus creating a more cohesive and interactive learning experience.

2.3 EXISTING SYSTEMS

Current educational technology systems offer diverse approaches to learning and assessment, each with its strengths and limitations. Traditional learning management systems (LMS) like *Moodle* and *Blackboard* provide robust frameworks for course management, content delivery, and assessment. These platforms enable educators to upload materials, create assignments, and track student progress. However, their design often centers on static content delivery and may lack interactive elements that engage students actively with the material.

Google Classroom has streamlined the process of assigning and collecting work, facilitating communication between teachers and students. Its integration with other Google tools supports collaborative work and resource sharing. Yet, it often lacks advanced interactive features for quizzes and assessments, which can limit its effectiveness in gauging student understanding in real-time.

Edpuzzle offers an innovative approach by allowing teachers to embed quizzes and interactive questions within video content. This method enhances engagement and ensures that students are actively processing the material. However, it is primarily focused on video-based content and may not fully address the need for comprehensive, text-based resources.

Quizlet and *Kahoot!* are popular tools for creating interactive quizzes and flashcards, often used to reinforce learning in a more dynamic and engaging way. While these tools excel in creating interactive assessments, they are often standalone applications that do not integrate seamlessly with other learning materials, such as PDFs or textbooks.

These existing systems highlight a growing trend towards interactive and engaging learning, yet they often lack the integration of interactive content with comprehensive assessment tools. *Read Smart* aims to address these gaps by merging interactive PDFs with customized quizzes and progress tracking, offering a more cohesive and integrated educational experience.

2.4 TECHNOLOGY SURVEY

The educational technology landscape is marked by a range of tools designed to enhance learning and assessment. Learning Management Systems (LMS) like *Moodle* and *Blackboard* offer comprehensive solutions for managing courses, including content delivery, assignment tracking, and student analytics. While these platforms provide structure, they often lack interactive features that engage students more actively. Tools such as *Edpuzzle* address this by embedding quizzes within video content, promoting engagement through multimedia, but they are primarily focused on video and do not cater to text-based resources like PDFs.

Assessment-focused platforms such as *Quizlet* and *Kahoot!* excel in creating interactive quizzes and flashcards, which reinforce learning through gamification. However, these tools operate separately from other educational materials, limiting their integration with diverse content. *Google Classroom* streamlines assignment distribution and collaboration, integrating with Google's suite of tools to facilitate content management, yet it falls short in providing advanced interactive assessment features.

Platforms like *Socrative* provide real-time assessments and immediate feedback, aiding in the tracking of student performance. Despite their strengths in assessment, they often lack comprehensive content delivery and integration. *Read Smart* builds on the strengths of these technologies by merging interactive PDFs with customized quizzes and automated progress tracking, creating a more cohesive and

engaging educational experience that addresses the limitations of existing systems.

3. HARDWARE AND SOFTWARE REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

Processor	Intel i3 and above
Ram	Minimum 4 GB
Hard Disk	Minimum 10 GB

3.2 SOFTWARE REQUIREMENTS

Operating System	Windows or Linux
Frontend Tool	ReactJS (v18.2.0 or higher)
Backend Tool	NodeJS(v18.16.0 or higher)
Database	MongoDB (v5.0.2 or higher)
Web Browser	Google Chrome, Microsoft Edge, Brave
Development Tool	Visual Studio Code
API Testing Tool	Postman

4.SOFTWARE REQUIREMENTS SPECIFICATION

4.1 USERS

Teacher

Student

4.2 FUNCTIONAL REQUIREMENTS

The *Read Smart* platform must facilitate user authentication and management, providing distinct access for both teachers and students. Teachers should have the ability to register, log in, and manage their accounts, including creating, updating, and deleting their profiles. Similarly, students need to register, log in, and manage their profiles, with access to content governed by their progress and performance.

Teachers must be able to upload PDF documents to the platform, with support for various file sizes and types. The system should allow for the organization of these PDFs into courses or modules, making it easier for teachers to manage and categorize content.

For quiz creation and management, teachers should be able to design quizzes with different question types such as multiple-choice, true/false, and short answer. They need to set correct answers, assign points, and schedule quizzes with start and end dates, as well as determine if they are timed.

Students should interact with PDFs by viewing and engaging with the content, and their access to new materials should be contingent on passing the associated quizzes. The platform must allow students to take quizzes, view questions, and submit answers, providing immediate feedback on their performance.

To support progress tracking and reporting, the system should monitor student achievements, including quiz scores and completion of materials. Students must have access to their own progress and scores, while teachers should receive detailed reports on student performance, encompassing individual quiz results and overall progress.

The platform needs to include a notification system to alert students about quiz deadlines, new content, and performance updates. Teachers should also be notified about student progress and quiz submissions. An internal messaging system is required for communication between teachers and students regarding course content and performance.

Data security and privacy are critical; the system must ensure the protection of user data, including personal information and academic records. Access control measures should be implemented to restrict access to authorized users only.

Finally, the platform should support integration with other educational tools or systems if necessary and be designed to scale efficiently as the number of users and documents grows, ensuring consistent performance and reliability

4.3 NON-FUNCTIONAL REQUIREMENTS

The non-functional requirements for the *Read Smart* platform focus on the overall quality and performance aspects essential for delivering a robust and reliable educational tool.

Performance and Scalability: The platform must handle a growing number of users, documents, and concurrent interactions without experiencing performance degradation. It should efficiently manage large PDF files and support high volumes of quiz submissions while maintaining fast response times and smooth user interactions.

Reliability and Availability: *Read Smart* should ensure high availability, with minimal downtime and quick recovery from any system failures. The platform must provide reliable access to educational materials and assessments, maintaining operational continuity and supporting uninterrupted learning.

Security and Data Privacy: The system must safeguard user data through strong security measures, including encryption for sensitive information and secure authentication protocols. Compliance with data protection regulations is essential to protect personal and academic data from unauthorized access and breaches.

Usability and Accessibility: The platform should offer a user-friendly interface that is intuitive for both teachers and students. It must adhere to accessibility standards to accommodate users with disabilities, ensuring that all features are usable by individuals with various needs and preferences.

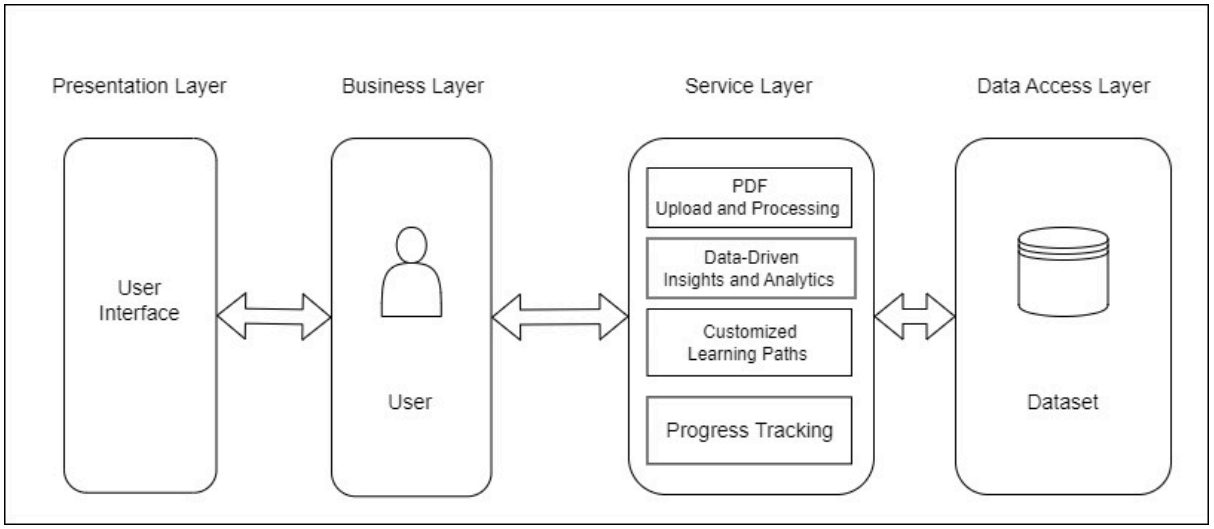
Maintainability and Support: The system should be designed for ease of maintenance, with clear documentation and a modular architecture that allows for efficient updates and bug fixes. Support mechanisms should be in place to assist users with technical issues and provide timely resolutions.

Compatibility and Integration: *Read Smart* needs to be compatible with various web browsers and devices, ensuring a consistent user experience across different platforms. It should also support integration with other educational tools and systems as required, facilitating a seamless workflow for users.

Efficiency and Resource Management: The platform should optimize resource usage, such as server load and storage, to ensure cost-effectiveness and environmental sustainability. Efficient resource management will contribute to overall system performance and reduce operational costs

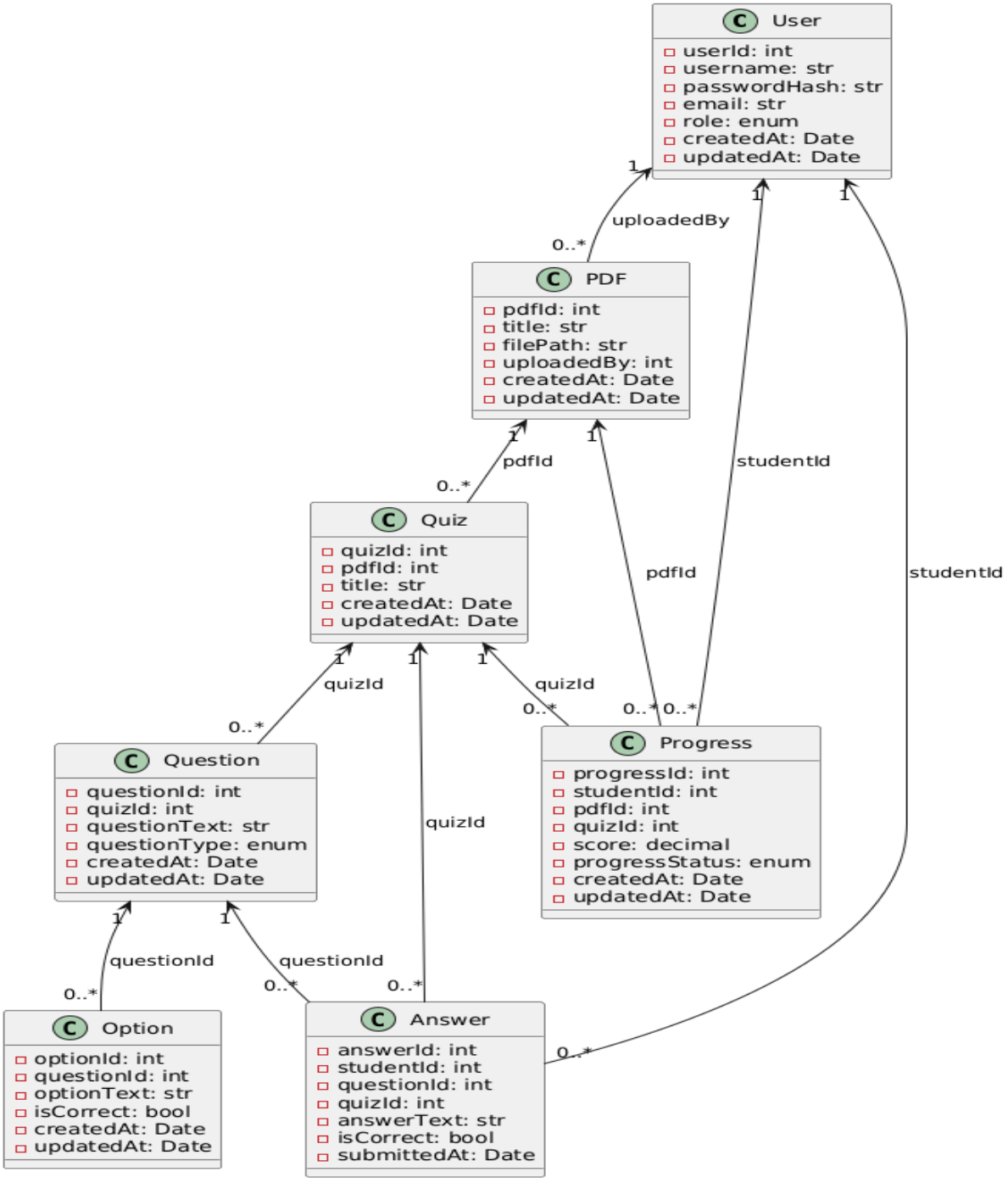
5. SYSTEM DESIGN

5.1 ARCHITECTURE DIAGRAM



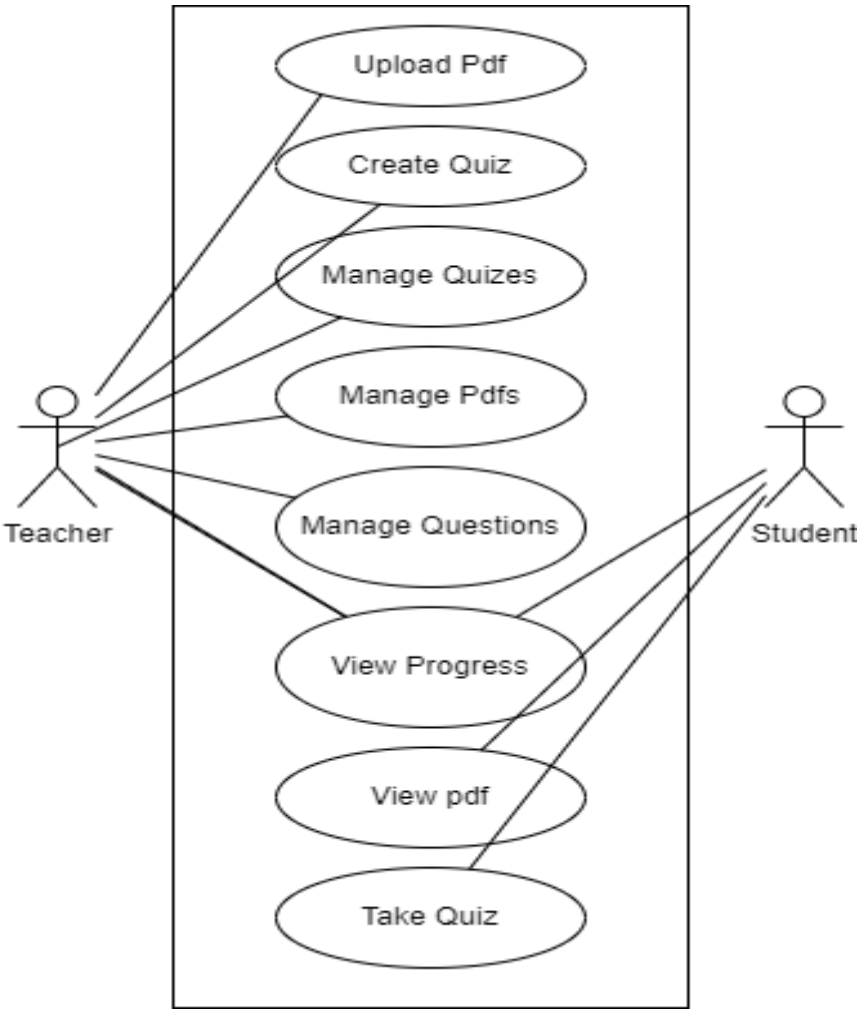
6. DETAILED DESIGN

6.1 CLASS DIAGRAM



User: Represents both teachers and students. Contains attributes like `userId`, `username`, `passwordHash`, `email`, `role`, and timestamps for creation and updates.
PDF: Represents the PDF documents. Contains attributes like `pdfId`, `title`, `filePath`, `uploadedBy` (which links to `User`), and timestamps.
Quiz: Represents quizzes associated with PDFs. Contains attributes like `quizId`, `pdfId` (which links to `PDF`), `title`, and timestamps.
Question: Represents questions within a quiz. Contains attributes like `questionId`, `quizId` (which links to `Quiz`), `questionText`, `questionType`, and timestamps.
Option: Represents options for multiple-choice questions. Contains attributes like `optionId`, `questionId` (which links to `Question`), `optionText`, `isCorrect`, and timestamps.
Answer: Represents answers submitted by students. Contains attributes like `answerId`, `studentId` (which links to `User`), `questionId` (which links to `Question`), `quizId` (which links to `Quiz`), `answerText`, `isCorrect`, and `submittedAt`.
Progress: Tracks student progress through quizzes and PDFs. Contains attributes like `progressId`, `studentId` (which links to `User`), `pdfId` (which links to `PDF`), `quizId` (which links to `Quiz`), `score`, `progressStatus`, and timestamps.

6.2 USE CASE DIAGRAM



1. **Teacher**

   Upload PDF: Teachers upload PDF documents to the platform.

   Create Quiz: Teachers create quizzes associated with uploaded PDFs.

   Manage PDFs: Teachers can update or delete their uploaded PDFs.

   Manage Questions: Teachers add, update, or delete questions within quizzes.

2. **Student**

View PDF: Students access and view PDF documents.

Take Quiz: Students take quizzes that are associated with the PDFs they have access to.

View Progress: Students can view their progress, including quiz scores and completed materials.

Relationships

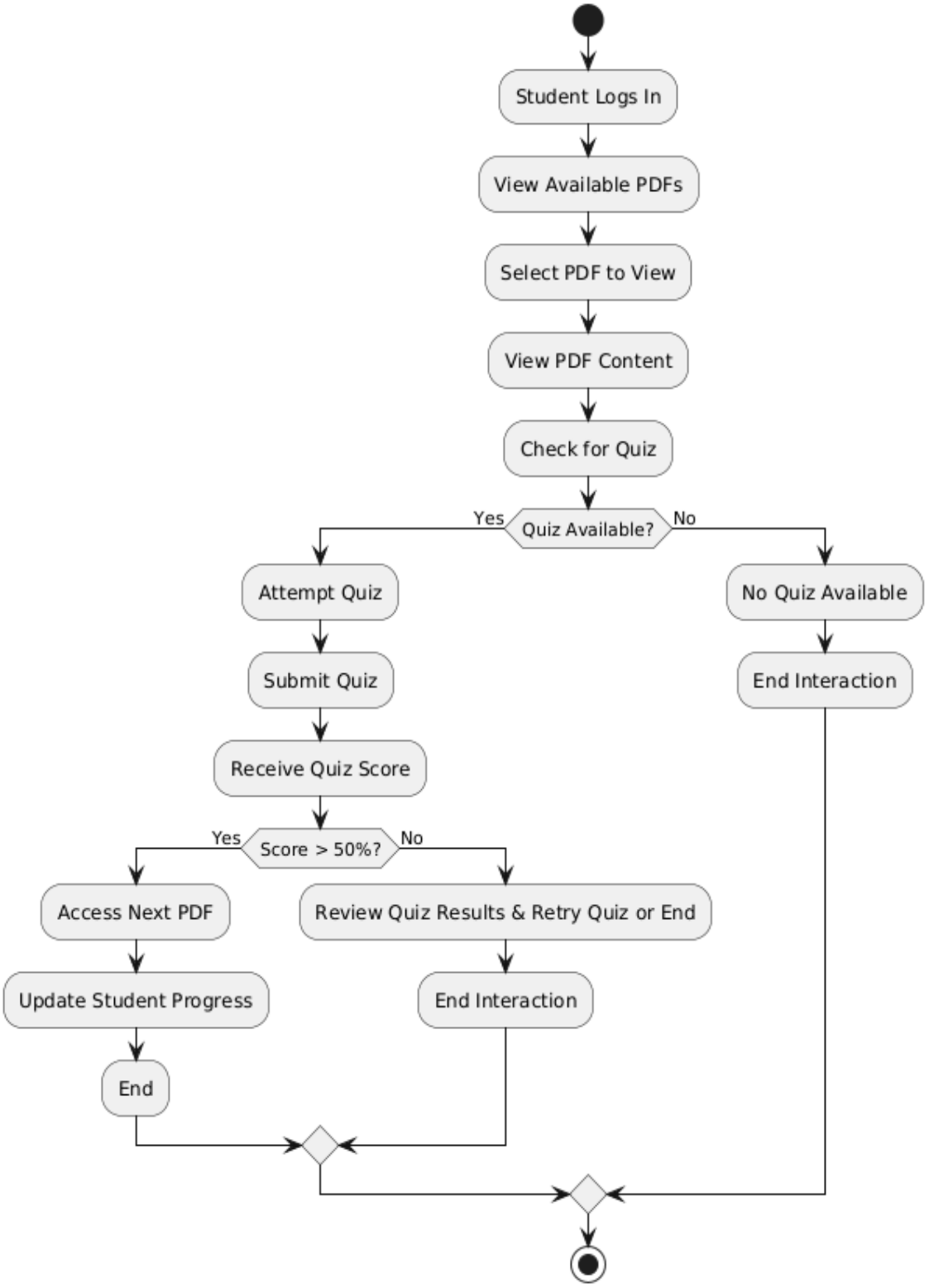
Teacher interacts with **Upload PDF**, **Create Quiz**, **Manage PDFs**, and **Manage Questions** use cases.

Student interacts with **View PDF**, **Take Quiz**, and **View Progress** use cases.

Take Quiz is associated with **View Quiz** to provide students with quiz content.

View Progress allows students to track their performance, which is influenced by the quizzes they take and the PDFs they view.

6.3 ACTIVITY DIAGRAM



Start: The process begins when the student starts using the *Read Smart* platform.
Student Logs In: The student logs into their account to access personalized features.
View Available PDFs: The student sees a list of available PDFs they can choose from.
Select PDF to View: The student selects a PDF to read.
View PDF Content: The student reads the content of the selected PDF.
Check for Quiz: The system checks if there is a quiz associated with the PDF.

- **Quiz Available?**
    - **Yes**: Proceed to attempt the quiz.
    - **No**: Proceed to the end of the interaction or other available actions.

Attempt Quiz: The student starts taking the quiz associated with the PDF.
Submit Quiz: The student submits their answers for the quiz.
Receive Quiz Score: The system calculates and provides feedback on the student's quiz score.
Score > 50%?

- **Yes**: The student is given access to the next PDF.
- **No**: The student reviews their quiz results and may choose to retry the quiz or end their interaction.

Access Next PDF: If the quiz score is above 50%, the student gains access to the next PDF.
Update Student Progress: The system updates the student's progress record, reflecting the latest achievements and access permissions.
End Interaction: The student's session ends, and they can choose to log out or continue with other tasks.

6.4 DATABASE DESIGN

6.4.1 DOCUMENT STRUCTURE(USER)

User_id	**Int**
Username	**String**
Password	**String**
Email	**String**
Role	**ENUM('teacher', 'student')**

Fig 6.4.1 Document Structure(User)

6.4.2 DOCUMENT STRUCTURE(PDF)

pdf_id	**Int**
Title	**String**
File_Path	**String**
Created_at	**Timestamp**
Updated_at	**Timestamp**

fig 6.4.2 Document Structure(Pdf)

6.4.3 DOCUMENT STRUCTURE(Progress Table)

Progress_Id	**Int**
Student_Id	**Int**
Score	**Int**
Progress_Status	**ENUM('completed', 'incomplete')**

fig 6.4.3 Document Structure(Progress table)

7. IMPLEMENTATION

Step 1: Configure the MongoDB Database Schema

- **Define Schemas:** Establish MongoDB schemas for users, PDFs, quizzes, questions, options, answers, and progress. Use Mongoose to create models that represent these entities and their relationships.
- **Design Collections:** Create collections in MongoDB to store data corresponding to each schema. This will facilitate efficient querying and data management.

Step 2: Backend (Express.js and Node.js)

- **Set Up Server:** Initialize an Express.js server. Configure it to handle various HTTP requests.
- **Define Routes:** Create RESTful routes for CRUD operations (GET, POST, PUT, DELETE) for managing PDFs, quizzes, questions, and student progress. Examples include routes for fetching quizzes, creating new questions, and updating student progress.
- **Implement Middleware:** Install middleware for authentication and authorization (e.g., JWT) to secure routes. Add middleware for handling errors and logging.
- **Connect to MongoDB:** Use Mongoose to establish a connection to the MongoDB database. Implement data models based on the previously defined schemas.
- **Create Controllers:** Develop controllers to interact with the database and handle business logic. Controllers will manage data retrieval, updating, and deletion.
- **Error Handling:** Implement middleware to handle errors gracefully, providing meaningful responses to the client.

Step 3: React Frontend

- **Set Up React App:** Initialize a React application using tools like Create React App or Vite.
- **Build Components:** Develop components for user authentication, PDF viewing, quiz taking, and progress tracking. Components might include forms for login and registration, PDF viewers, and quiz interfaces.
- **Implement Routing:** Use React Router to manage navigation between different views (e.g., home, PDF content, quizzes, and progress reports).
- **Connect to Backend:** Use Axios or Fetch to make HTTP requests to the backend API endpoints for CRUD operations. Ensure data is fetched and displayed appropriately in the UI.
- **Handle Authentication:** Implement JWT-based authentication to manage user sessions. Ensure that routes are protected based on user roles (teacher or student).
- **State Management:** Utilize React hooks or Redux for managing global state, such as user authentication status, quiz results, and progress tracking.

Step 4: Workflow for Learning

- **Display PDFs:** On the main page or dashboard, list available PDFs. Allow users to access content based on their progress and quiz performance.
- **Quiz Integration:** Present quizzes associated with each PDF. Ensure that students can only access new PDFs if they achieve a score of over 50% on the previous quizzes.
- **Progress Tracking:** Implement a system to track and display student progress. Provide visual feedback on completed quizzes and available content.

7.1 SCREENSHOTS

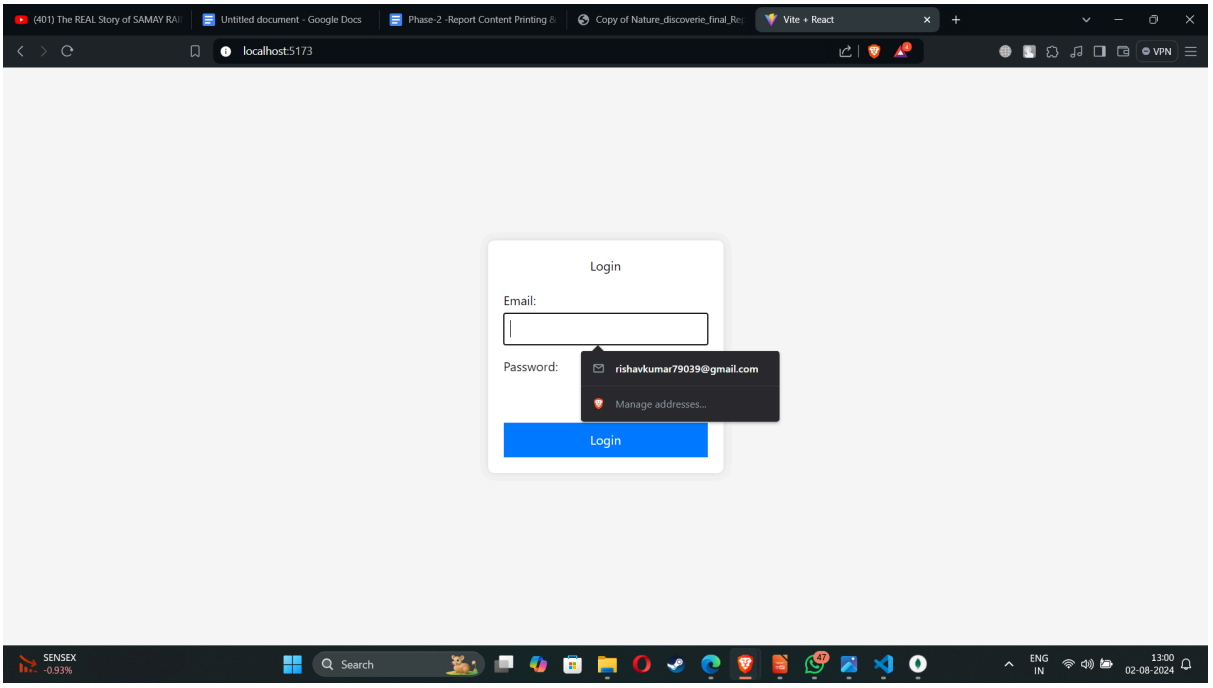


fig-7.1.1 - login page

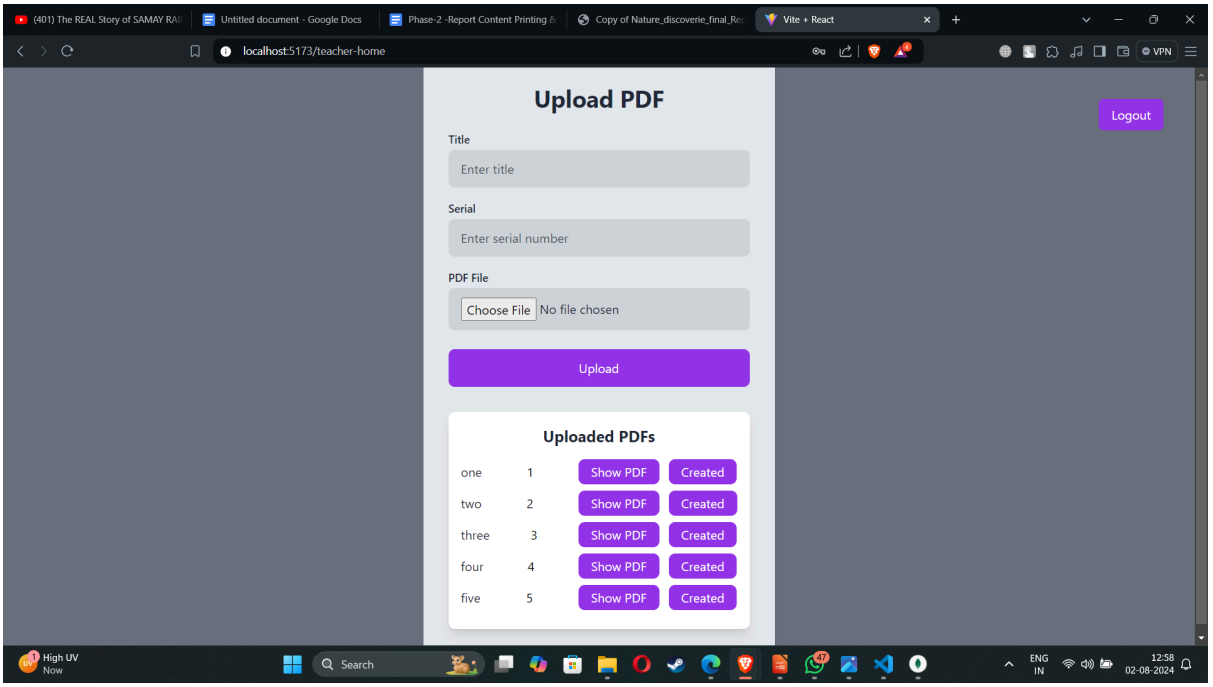


fig-7.1.2 teacher-home page

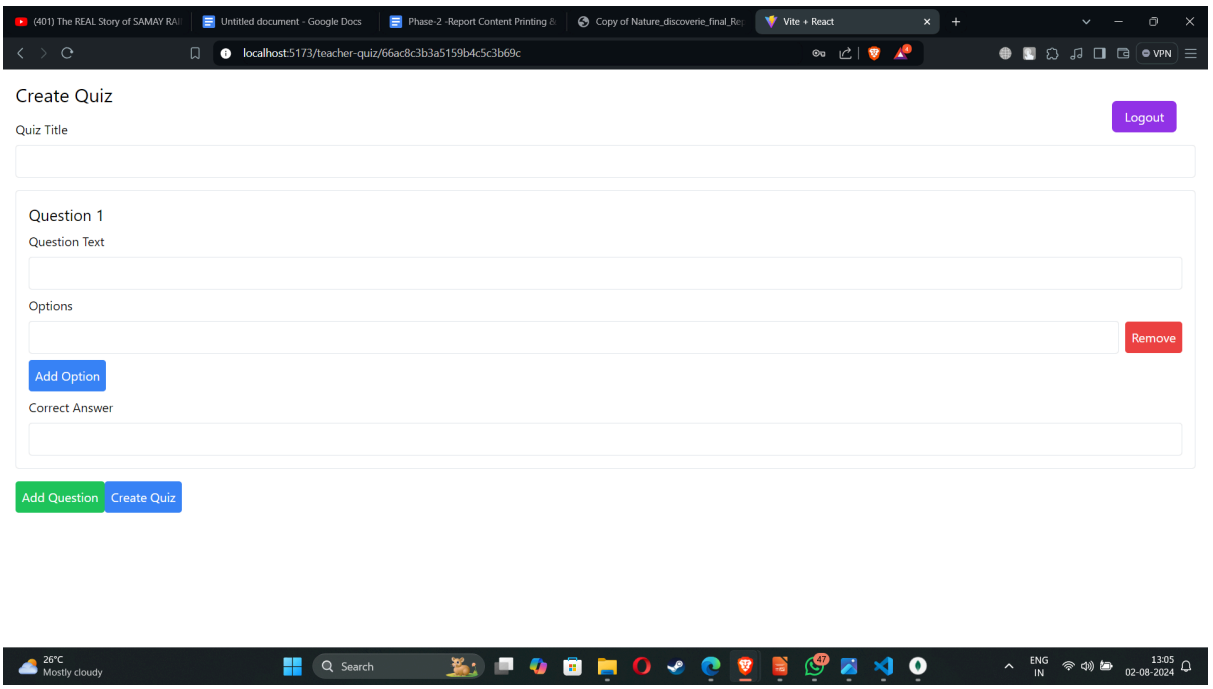


fig-7.1.3 - Teacher-quiz-create page

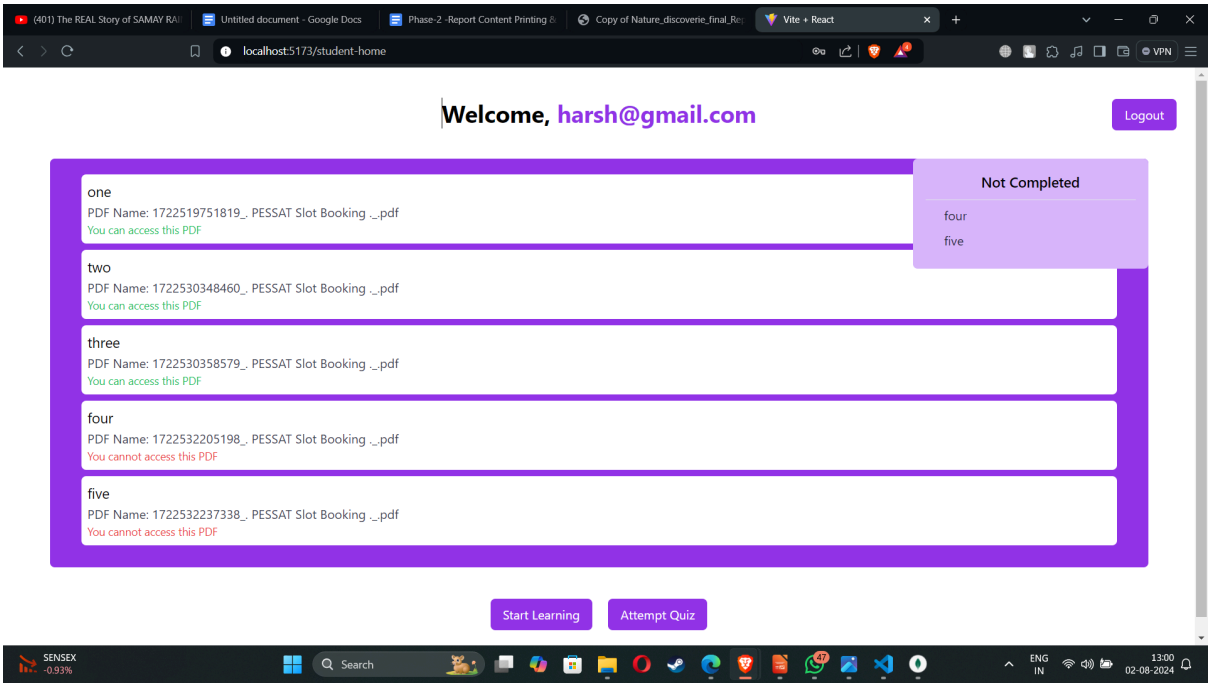


fig-7.1.4 student page

8. SOFTWARE TESTING

Login Testing

Test Case Id	Test Case Description	Test Steps	Expected Result	Pass/Fail
1.1	Verify successful login with valid credentials.	. Enter valid username and password. 2. Click "Login".	User is redirected to the dashboard.	
1.2	Verify login with invalid credential	1. Enter invalid username or password. 2. Click "Login".	Error message "Invalid username or password" is displayed.	
1.3	Check login with	1. Leave username and/or password fields empty. 2. Click "Login".	Error message "Username and password are required" is displayed.	

1.4	Verify password visibility toggle functionality.	1. Enter password. 2. Click the visibility toggle button	Password field shows/hides the password correctly.	

2. PDF Upload Testing

Test Case Id	Test Case Description	Test Steps	Expected Result	Pass/Fail
2.1	Verify successful PDF upload.	1. Click "Upload PDF". 2. Select a valid PDF file. 3. Click "Submit"..	PDF is uploaded successfully and appears in the list.	
2.2	Check PDF upload with invalid file format.	1. Click "Upload PDF". 2. Select a non-PDF file. 3. Click "Submit"..	Error message "Invalid file format" is displayed..	
2.3	Verify upload with large PDF file	1. Click "Upload PDF". 2. Select a large PDF file. 3. Click "Submit".	PDF uploads without errors or an appropriate size limit message is shown.	

	.			
2.4	Check PDF upload with missing file.	1. Click "Upload PDF". 2. Do not select a file. 3. Click "Submit".	Error message "Please select a file" is displayed.	

3. Quiz Creation Testing

Test Case Id	Test Case Description	Test Steps	Expected Result	Pass/Fail
3.1	Verify successful quiz creation.	1. Click "Create Quiz". 2. Fill in quiz details. 3. Click "Save".	Quiz is created and saved successfully.	
3.2	Check quiz creation with missing required fields..	1. Click "Create Quiz". 2. Leave required fields empty. 3. Click "Save"...	Error message "Required fields cannot be empty" is displayed..	
3.3	Verify creating a quiz with multiple question types. .	1. Click "Create Quiz". 2. Add different types of questions. 3. Click "Save"..	All question types are saved and appear in the quiz.	

3.4	Check quiz creation with invalid data.	1. Click "Create Quiz". 2. Enter invalid data (e.g., special characters in numeric fields). 3. Click "Save".	Error message indicating invalid data is displayed.	

4. Student Can Give Quiz Testing

Test Case Id	Test Case Description	Test Steps	Expected Result	Pass/Fail
3.1	Verify that a student can attempt a quiz.	1. Log in as a student. 2. Access a quiz. 3. Answer questions and submit.	Quiz is successfully submitted and results are recorded.	
3.2	Check quiz attempt with incomplete answers.	1. Log in as a student. 2. Start a quiz but do not answer all questions. 3. Submit the quiz.	System handles incomplete submissions and provides appropriate feedback..	
3.3	Verify quiz attempt after reaching PDF score limit.	1. Ensure previous quizzes have been taken. 2. Check if the	Student can attempt quizzes only if previous scores meet the required	

	.	student can access new quizzes.	criteria.	
3.4	Check quiz creation with invalid data.	1. Ensure previous quizzes have been taken. 2. Check if the student can access new quizze	Student can attempt quizzes only if previous scores meet the required criteria..	

\

9. CONCLUSION

In conclusion, the *Read Smart* platform effectively enhances the learning experience by integrating interactive PDFs and quizzes into a seamless educational tool. By leveraging the MERN stack, the project ensures a robust and scalable solution for both educators and students. The iterative development and comprehensive testing phases have ensured that the platform is both functional and user-friendly. With successful deployment and ongoing support, *Read Smart* stands poised to facilitate engaged learning and provide valuable insights into student progress, addressing key challenges in traditional education methods.

APPENDIX A