VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY

(An Autonomous Institute Affiliated to University of Mumbai
Department of Computer Engineering)

Department of Computer Engineering

Project Report on

LearnEase: Adaptive Learning Hub

Submitted in partial fulfillment of the requirements of Third Year (Semester–VI), Bachelor of Engineering Degree in Computer Engineering at the University of Mumbai Academic Year 2024-25

By

Jiten Purswani D12B/43

Srimathi Srinivasan D12B/55

Laveena Mirani D12B/30

Kareena Lachhani D12B/26

Project Mentor
Mrs. Pallavi Saindane

University of Mumbai
(AY 2024-25)

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY

(An Autonomous Institute Affiliated to University of Mumbai
Department of Computer Engineering)

Department of Computer Engineering

CERTIFICATE

This is to certify that _____of Third Year Computer Engineering studying under the University of Mumbai has satisfactorily presented the project on "----------" as a part of the coursework of Mini Project 2B for Semester-VI under the guidance of ----------. in the year 2024-25.

_____

Date

_____ _____

Internal Examiner External Examiner

_____ _____ _____

Project Mentor Head of the Department Principal

Dr. Mrs. Nupur Giri Dr. J. M. Nair

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)
Jiten Purswani D12B/43

(Signature)
Srimathi Srinivasan D12B/55

(Signature)
Laveena Mirani D12B/30

(Signature)
Kareena Lachhani D12B/26

Date:

ACKNOWLEDGEMENT

We are thankful to our college Vivekanand Education Society's Institute of Technology for considering our project and extending help at all stages needed during our work of collecting information regarding the project.

It gives us immense pleasure to express our deep and sincere gratitude to Assistant Professor **Mrs. Pallavi Saindane** (Project Guide) for her kind help and valuable advice during the development of project synopsis and for her guidance and suggestions.

We are deeply indebted to Head of the Computer Department **Dr.(Mrs.) Nupur Giri** and our Principal **Dr. (Mrs.) J.M. Nair ,** for giving us this valuable opportunity to do this project.

We express our hearty thanks to them for their assistance without which it would have been difficult in finishing this project synopsis and project review successfully.

We convey our deep sense of gratitude to all teaching and non-teaching staff for their constant encouragement, support and selfless help throughout the project work. It is a great pleasure to acknowledge the help and suggestion, which we received from the Department of Computer Engineering.

We wish to express our profound thanks to all those who helped us in gathering information about the project. Our families too have provided moral support and encouragement several times.

Computer Engineering Department

COURSE OUTCOMES FOR T.E MINI PROJECT 2B

Learners will be to:-

CO No.	COURSE OUTCOME
CO1	Identify problems based on societal /research needs.
CO2	Apply Knowledge and skill to solve societal problems in a group.
CO3	Develop interpersonal skills to work as a member of a group or leader.
CO4	Draw the proper inferences from available results through theoretical/ experimental/simulations.
CO5	Analyze the impact of solutions in societal and environmental context for sustainable development.
CO6	Use standard norms of engineering practices
CO7	Excel in written and oral communication.
CO8	Demonstrate capabilities of self-learning in a group, which leads to lifelong learning.
CO9	Demonstrate project management principles during project work.

ABSTRACT

The current learning systems are more generalised and teacher-centric. The teachers disseminate information equally to all studies using the same methods. But, each student has unique learning styles and preferences. This is a major concern that most students experience. To address this gap in their learning process, we are working on a Adaptive Learning system: LearnEase. This system has an interactive summarizer that can summarise the content that students provide. It also features a quiz module that provides dynamic quizzes based on the summarised content the student has studied. The proposed system also can help students plan by recommending schedules based on their requirements, strengths and weaknesses.

Index

Appendix

a. List of Figures

b. List of tables

c. Paper Publications :-

1. **Draft of the paper published.**
2. **Plagiarism report of the paper published /draft**
3. **Certificate of the paper publication**
4. **Xerox of project review sheets**

Chapter 1: Introduction

1.1 Introduction

The scope of tech-based platforms being integrated into the current learning space has been on a steady growth over the past few years. The proposed system is one such platform that will help students to route their learning journey according to their needs and expectations. The proposed system aims to provide a tailored learning experience for each learner.

The proposed system's objective is to use machine learning (ML) algorithms to deliver material that is appropriately structured and tailored to users, increasing their engagement and retention. The system will be responsive, with regular tests and evaluations to improve the recommendations and content sent to users by measuring their assimilation of the information.

Recognizing the diverse needs of students, the proposed system aims to leverage algorithms to provide tailored content summaries to students. This interactive system will allow students to learn at their own pace and repeat topics till they are left with a deeper understanding of the material.

In addition, the proposed system will provide frequent tests and evaluations to gauge the students' retention and assimilation from the above content.[1] This assessment will also aid in understanding the student's strengths and weaknesses that can be further used to recommend a course of action, a schedule according to which they can focus on improving the weaker sections and keep in touch with the other topics.[1] This schedule recommended will be able to aid students to approach their learning in a structured format.

By creating a dynamic work environment which emphasises the student's personality and adapts accordingly, the system seeks to enrich their learning experience. The system will be further improved through frequent feedback and assessment which can help refine the models.

1.2 Motivation

The traditional education system, though lays a foundation, fails to address the personal learning styles of students.[8] We ourselves have observed this gap and thought to develop a technological model to bridge it by integrating ML models.

The recent rise of online learning has boosted the demand for technology powered solutions in the education space. Many platforms are able to offer vast amounts of content but few have the ability to dynamically adapt to user's personal needs and real-time performance. We plan to leverage student data and ML algorithms to provide customized learning paths to fill this gap.

The motivation for developing this system is to ensure that education should be made more accessible and adaptable. It will help students to obtain only necessary materials and information and improve their understanding amidst the information overload available today. This system will help students to set priorities, learn accordingly and then test their knowledge assimilation.

1.3 Problem Definition

Because of the one-size-fits-all nature of the existing educational system, individual student needs are not met, which results in disengagement and subpar academic performance. Conventional approaches do not take into consideration different learning speeds, tailored content relevancy, or unique learning styles. Real-time, tailored feedback is absent from standardised testing, and time and resource restrictions frequently prevent teachers from giving students the targeted attention they need. Furthermore, there is still underutilization of educational data's potential to improve tailored learning. To increase student engagement and academic achievement, a system that adjusts to each student's particular learning style, pace, and preferences is crucial. Although there are many educational platforms available that offer a plethora of educational content, they are not curated specifically to match learner’s specific needs.

1.4 Lacuna of the existing systems

1. Overfitting risk: Many of the existing systems seem to be prone to overfitting. This limits the generalizability of the models to new, unseen data, which is crucial for designing an effective system that is adaptive.

2. Lack of Contextual understanding: There is a lack of specific, context-aware datasets that capture the nuances of the students’ requirements and preferences of individual students. Using generic datasets fail to be applicable for deriving actual learning patterns.

3. Limited Real-world testing: Most studies show limited testing in real-world situations, raising concerns about reliability, scalability and performance of systems in diverse situations. While algorithms may perform well in controlled environments, they often struggle when applied to real-world educational settings as student learning paths can be unpredictable.

4. Explainability Issues: Many systems lack transparency in their decision-making processes. The users are unaware of how or why content is recommended, leading to a lack of trust in the system.

5. Cold start problem: The existing systems face the challenge of cold start problem. This means that, when the system has very little information regarding new users, the recommendations are hampered. This occurs especially in the early stages of system release.

6. Potential Bias: Many algorithms that are used in personalised learning systems are susceptible to biases. This could be because of biased training data or model design. This is a very serious issue as it could lead to recommendation of incorrect or unfair learning paths to students.

1.6 Relevance of the Project

The LearnEase project is highly relevant in today's educational context, where traditional teaching methods often fail to address the diverse learning needs of students. With the increasing shift towards digital and personalized education, this system provides a much-needed solution that bridges the gap between generalized content delivery and individualized learning support. By leveraging machine learning algorithms, LearnEase facilitates content summarization, adaptive quizzes, and personalized study planning—features that directly respond to the academic challenges faced by students. The system is particularly beneficial for secondary school students, such as those studying under the Maharashtra State Board, who often lack structured digital support. Through real-time assessments, performance-based recommendations, and a responsive user interface, LearnEase enhances learning efficiency and promotes self-paced, goal-oriented study habits. This makes it a timely and impactful contribution to the evolving edtech ecosystem.

Chapter 2: Literature Survey

A. Overview of Literature Survey

2.1 Comparison with the existing system

Author	Publication	Year	Pros	Cons	Inferences
A. B. F. Mansur, N. Yusof, and A. H. Basori,	"Personalized Learning Model Based on Deep Learning Algorithm for Student Behaviour Analytic," *Procedia Computer Science*, vol. 163, pp [2]	2019	- Large set of attributes. - Automation of Feedback	- Potential risk of overfitting. - No specific context is present.	Need more specific dataset rather than a non contextual for a better understanding of the students.
Y. Ma, L. Wang, and Q. Zhang,	"A Personalized Learning Path Recommendation Method Incorporating Multi-Algorithm," *Applied Sciences*, vol. 13, no. 10, Article 5946 [3]	2023	- Personalized Learning paths - Swarm Intelligence	-Limited real-world testing - Dependencies	The PSO algorithm can be very helpful in creating personalized paths. Have to avoid dependencies.
Muñoz, Juan & Jan, Zohaib & Saavedra, Angelo	"Machine learning for learning personalization to enhance student academic performance." [4]	2021	- Ensemble Learning Approach - Cluster based Classifier	-Potential Overfitting - Dependencies	Cluster based classification is very popular nowadays and its accuracy rate is also very good

Author	Publication	Year	Pros	Cons	Inferences
T. B. Lalitha and P. S. Sreeja,	"Personalised Self-Directed Learning Recommendation System," *Procedia Computer Science*, vol. 171, pp. 583–592, 2020. [5]	2020	- Recommendation module by using collaborative filtering and content based filtering	- Lacks explainability to users how recommendation is generated	We can create a better recommendation model which will allow users as well to have a better understanding.
J. Alanya-Beltran	"Personalized Learning Recommendation System in E-learning Platforms Using Collaborative Filtering and Machine Learning," in *Proc. of the IEEE Conference on Artificial Intelligence (ACCAI)* [6]	2024	- Neural Networks - Decision Trees	- Cold start Problem - Potential Bias	Can implement neural networks which can be used to latent features from user- item interaction data.

Table 2.1.1 Literature Survey

Chapter 3: Introduction

3.1 Introduction to requirement gathering

Requirement gathering is the foundational step in the software development lifecycle. For the LearnEase project, this phase involved engaging with stakeholders such as students (target users), educators, and technical mentors to understand the educational challenges, especially with respect to personalized learning for students of Classes 9 and 10 (Maharashtra Board).

The team identified key pain points, such as:

- Lack of personalized learning experiences.
- Overwhelming volumes of study content.
- Absence of structured feedback mechanisms.
- Limited adaptability in conventional learning platforms.

Through brainstorming sessions, user surveys, and research into existing systems, the requirement gathering phase led to clearly defined project goals such as building a summarizer, dynamic quiz module, and a personalized schedule recommender.

3.2 Functional Requirements

These are the key functions the system must perform:

1. **Content Summarizer**
    - Allow students to upload study material (e.g., PDFs).
    - Use NLP models (BART) to generate concise summaries.
    - Display output in an interactive and readable format.
2. **Quiz Generator**
    - Generate customized quizzes based on the summarized content.
    - Adapt question difficulty based on user performance.
    - Provide real-time feedback and scores.
3. **Schedule Recommender**
    - Analyze user's performance, strengths, and weaknesses.
    - Generate a personalized study schedule using collaborative/content-based filtering.
    - Send reminders/notifications for upcoming sessions.

4. **User Management**
    - User signup/login.
    - Profile management.
    - Data tracking (history of performance, activity logs).
5. **Admin Panel (Optional/Advanced Phase)**
    - Monitor user activity.
    - Evaluate system performance.
    - Upload or approve learning materials.

3.3 Non-functional Requirements

Scalability

- System should handle multiple users simultaneously and accommodate growing usage as the user base expands.

Performance

- Fast processing of uploaded documents and summarization.
- Responsive interface across devices.

Usability

- Intuitive UI/UX (prototyped in Figma).
- Minimal user training required.

Reliability

- Accurate summarization and quiz generation.
- Minimal system crashes or downtime.

Maintainability

- Modular codebase for easier updates and debugging.
- Version control using GitHub.

Security

- Secure user data with authentication (login/password).
- Data encryption during upload and storage.

Portability

- Should work on all modern browsers and Android/iOS mobile devices (via React Native).

3.4 Hardware, Software, Technology and tools utilized

1. Frameworks and Libraries:

- React Native: For building interactive mobile applications.
- React: For creating dynamic and responsive web interfaces.
- Python: For backend development and implementing machine learning models.

2. Big Data and Machine Learning Tools:

- TensorFlow/Keras: For developing and training machine learning models.
- Pandas/Numpy: For data manipulation and analysis.

3. Development and Deployment Tools:

- Visual Studio Code: For code editing.
- Jupyter Notebook: For developing and testing machine learning models.
- Docker: For containerizing applications to ensure consistency across different environments.
- MongoDB: For cloud storage, computing resources, and deployment.
- Git/GitHub: For version control and collaboration.

Hardware Requirements:

- User Device: A modern laptop or desktop with at least 8GB RAM, Intel i5 or equivalent processor, and SSD storage for smooth operation of applications.
- Network Connectivity: Reliable high-speed internet connection to access cloud resources and interact with the application seamlessly.

Software Requirements:

- Web Browser: Latest versions of Chrome, Firefox, or Edge for accessing the web application.
- Operating System: Windows 10/11, macOS, or a recent Linux distribution to ensure compatibility with development and deployment tools.

3.5 Constraints

Time Constraints

- The complete development and integration had to be executed within two semesters, limiting feature expansion.

Resource Constraints

- Limited access to high-end GPUs for model training.
- Free-tier limitations of cloud services affected model inference speed.

Data Constraints

- Unavailability of large, subject-specific datasets for the Maharashtra board.
- Manual effort required for model fine-tuning due to lack of labeled data.

Cold Start Problem

- Difficulty in generating effective recommendations for new users due to insufficient usage data in early stages.

Model Limitations

- Summarization models (T5, LaMini) were not optimal for long documents.
- BART model required chunking for performance but lacked perfect cohesion across chunks.

Team Expertise

- The team had to self-learn many tools and frameworks during development, impacting early development speed.

Chapter 4: Proposed Design

4.1 Block Diagram of the system

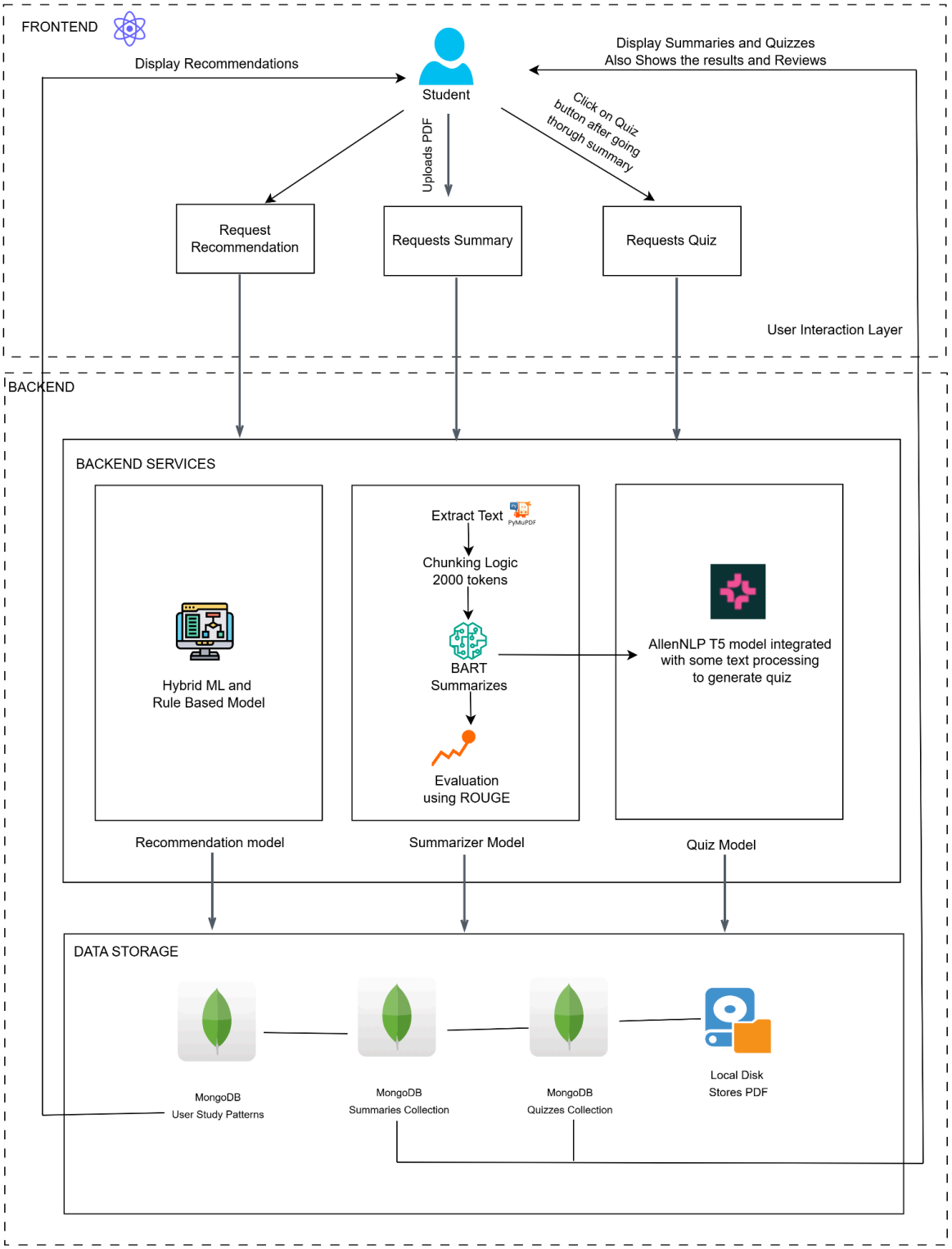

Fig 4.1.1 Block Diagram architecture of the system

4.2 Modular design of the system

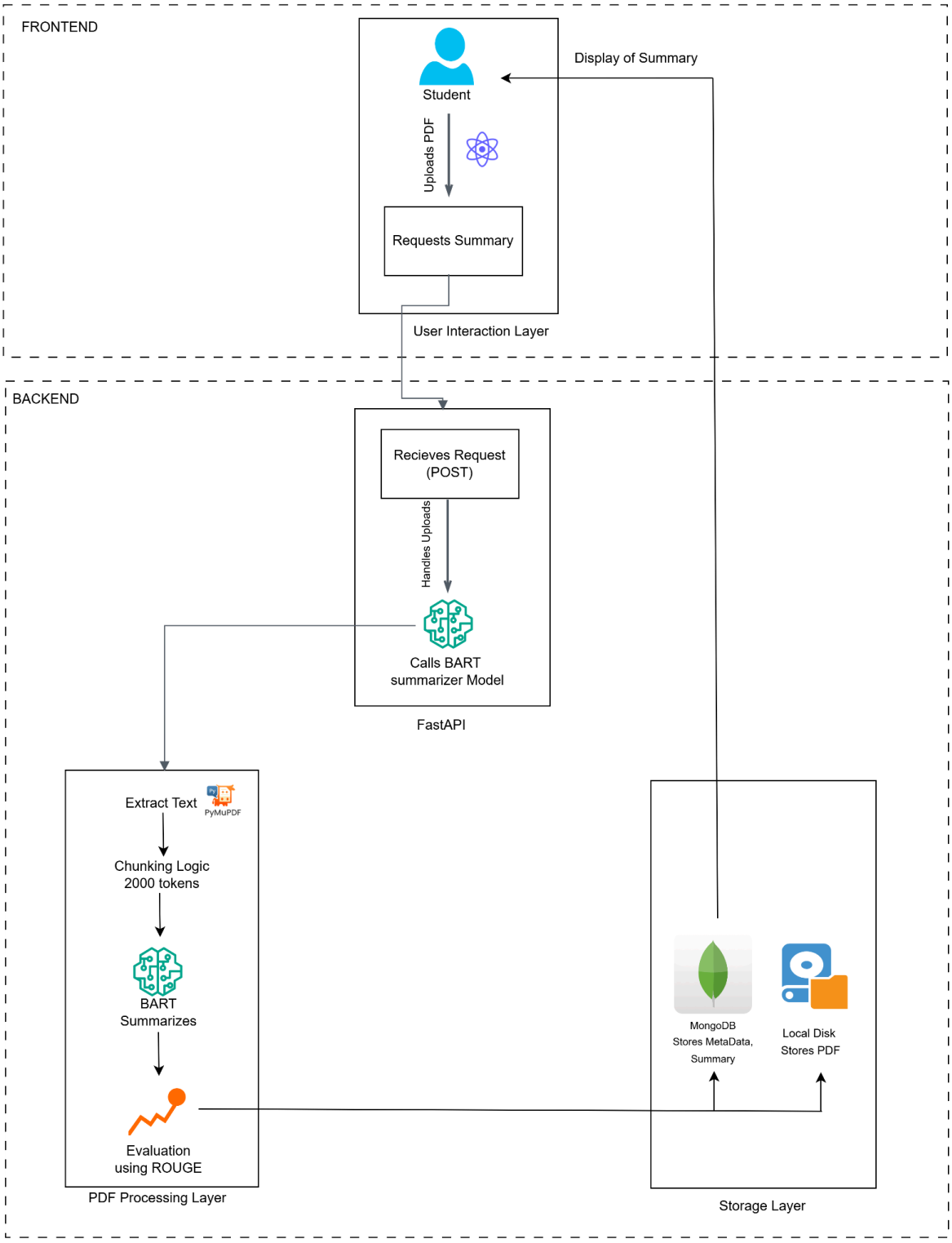

Fig 4.2.1 Summarizer Modular Design

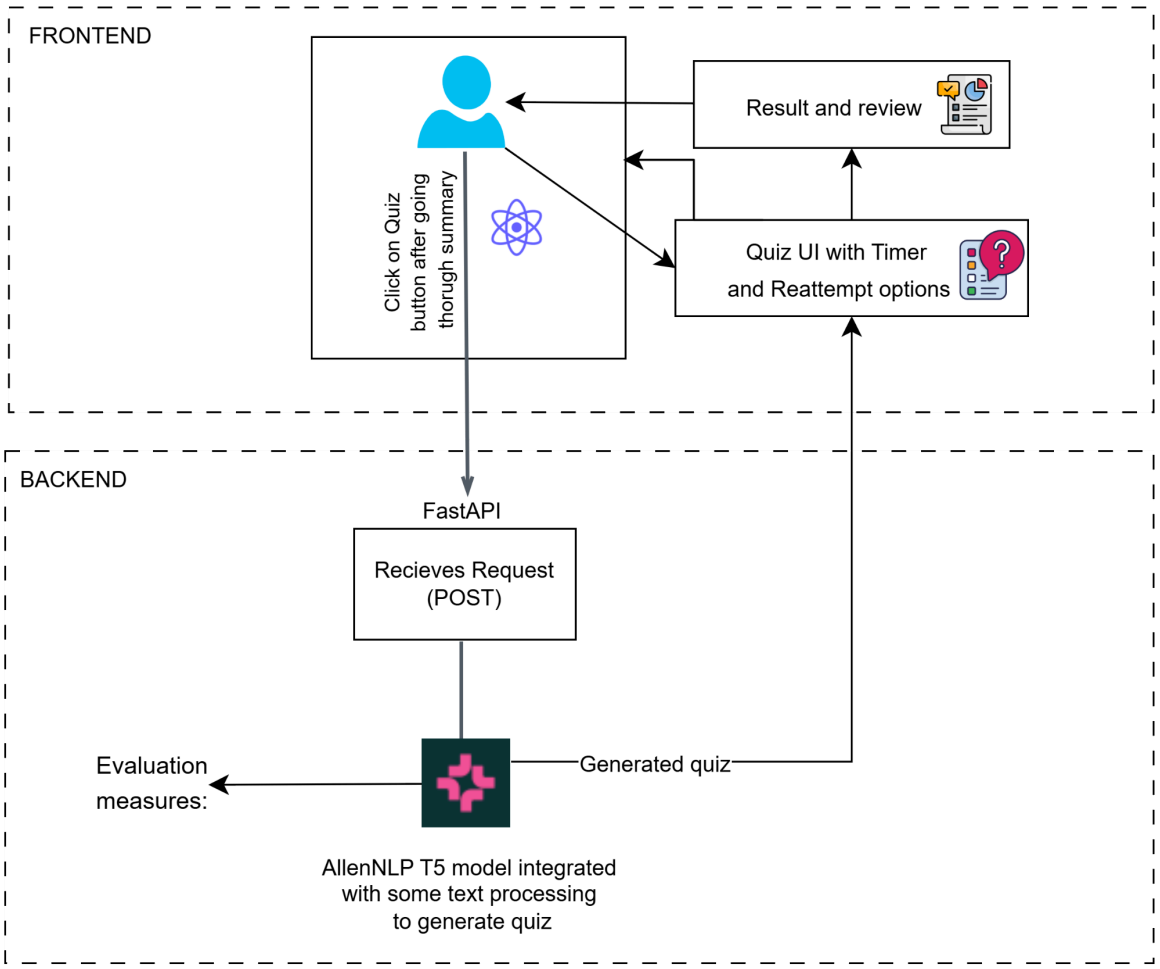

Fig 4.2.2 Quiz Modular Diagram

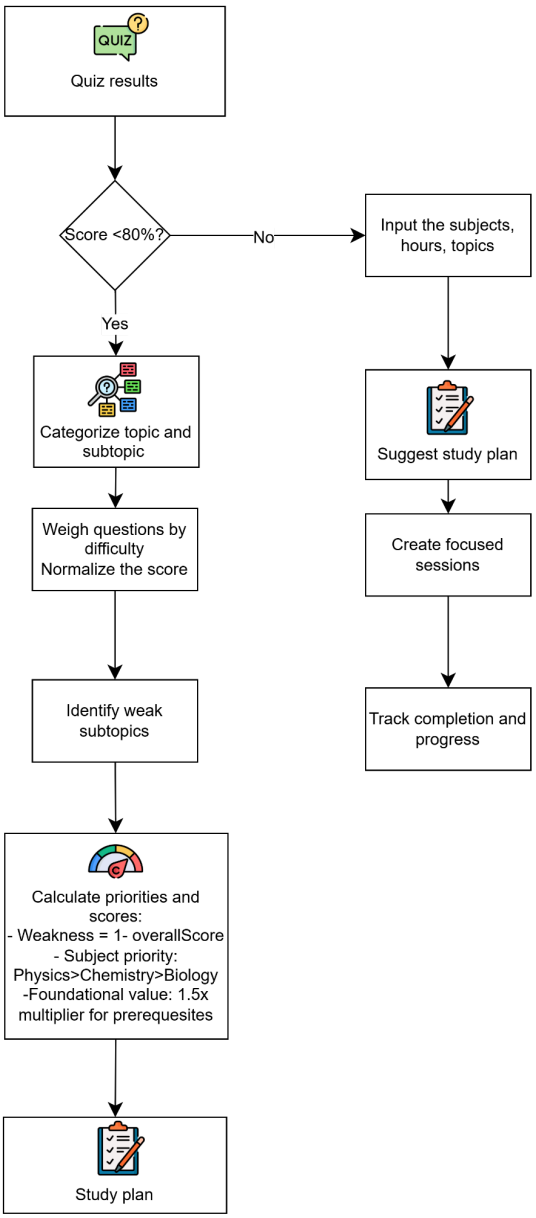

Fig 4.2.3 Schedule Recommendation Modular Diagram

4.3 Detailed design

LearnEase application system consists of a React Native based frontend. Backend uses RESTFUL APIs for processing, MongoDB database for storage and user authentication.

It consists of three main ML modules:

- Summarizer
- Quiz Generator
- Study Plan Recommender

Summarizer Model

The summarization model is based on the BART model for processing PDF documents uploaded by the user and condenses the content into concise summaries. BART is a transformer based architecture used for generating effective abstractive summaries.

When a user uploads a PDF, its text is extracted and preprocessed to remove noise. The content is then split into manageable chunks as the BART model has some limitations on the number of input tokens.

Each chunk is fed into the summarization model which has been trained on the CNN daily dataset. Summaries for each of the chunks are generated and combined to give the final output summary to the user.

Quiz Generator Model

Quiz Generator model is implemented using Allen-AI's T5 small model pre-trained on Squad2 dataset and fine tuned using SciQ dataset. It is used to generator multiple choice questions by extracting the key concepts from the summary and using a predefined set of distractors as the incorrect options.

Schedule Recommendation Model

There are two routes for schedule recommendation. If the user scores low in quizzes of concepts (below 80%) then they are directed to RecommendationSchedule. High scorers are directed to GenerateSchedule. Firstly, quiz data is classified into difficulty levels after standardizing them and detecting their metadata.

Raw Questions → Standardize Format → Detect Metadata → Structured Output

↓

(Subject → Topic → Subtopic → Difficulty)

↓

Keyword Matching + Counting

The RecommendationSchedule works as follows:

Initially the topic and subtopic structure of each subject is created. Questions are processed to apply difficulty weighting. Update the topics and subtopics based on correctness of the student response.

Normalize scores across the topics and identify weak topics. Calculate overall subject scores and priorities. Sort subjects by priority. Priority is calculated based on three factors:

Weakness : (1-overallScore)

Subject Importance: physics > chemistry > biology

Foundational value: 1.5x multiplier for prerequisite topics

The study plan is generated by processing the number of available hours/days.

Sessions are allocated proportionally to priority scores. Enforces minimum (30 minutes) and maximum session durations. Creates focused sessions targeting weakest subtopics.

Remaining sessions are allocated for review sessions to revise strong topics

The GenerateSchedule working is:

Allow user to input weak subjects or subjects they want to focus on, the hours per day they can allocate, and select topics from each subject. The model suggests a schedule study plan by providing a permutation of these inputs along with some study tips. It allows student to toggle topics as completed and track their progress.

4.4 Project Scheduling & Tracking: Gantt Chart

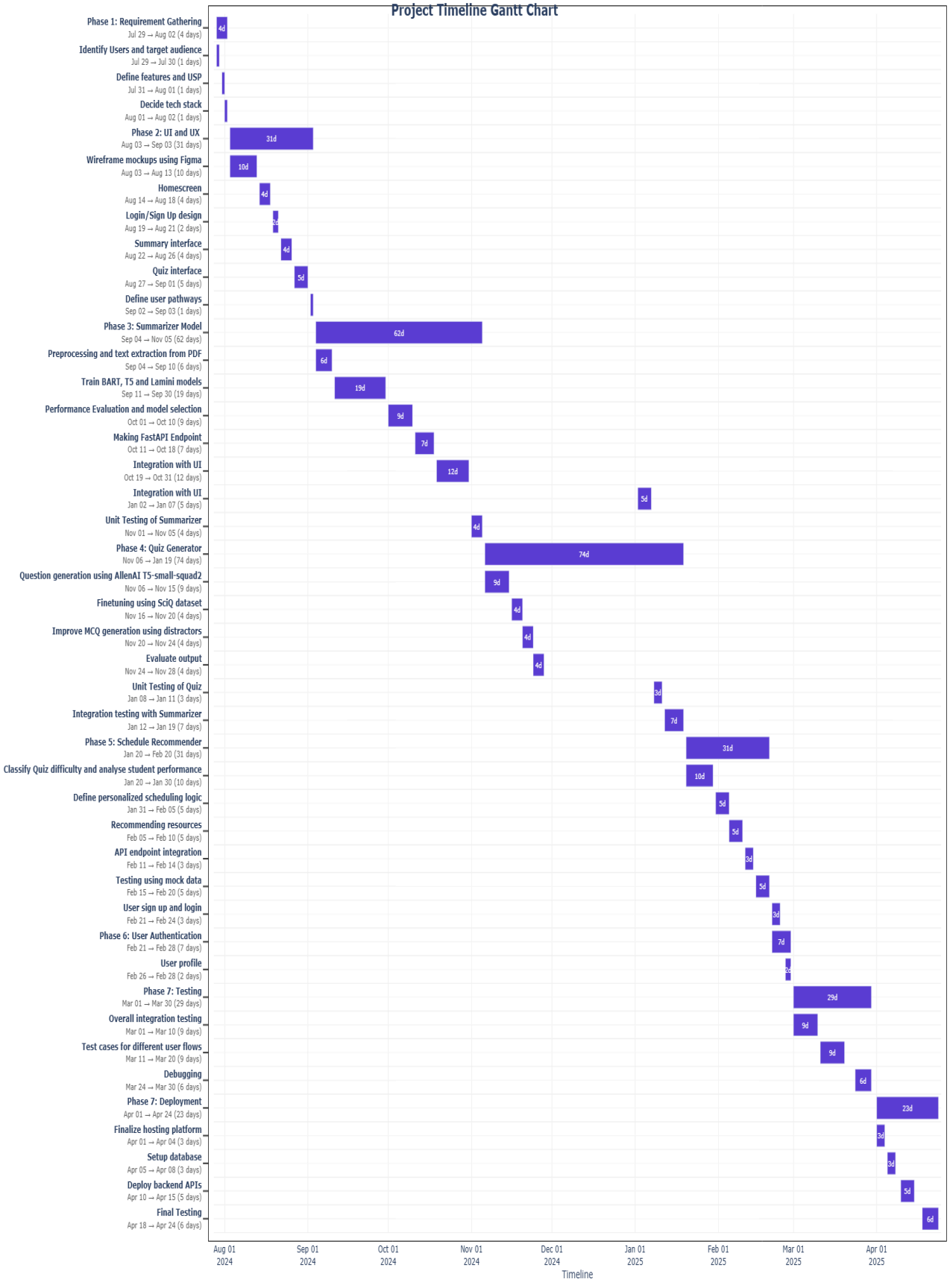

Fig 4.4.1 Gantt Chart

Chapter 5: Implementation of the Proposed System

5.1 Methodology Employed

The proposed system follows a modular approach, integrating summarization and quiz generation functionalities. The process begins with data ingestion from domain-specific academic sources, followed by preprocessing steps like tokenization, stopword removal, and text normalization. For summarization, the BART model was fine-tuned on the XSum dataset, while the quiz generator leveraged SciQ and additional domain-augmented datasets. The system pipeline is designed to ensure coherence, scalability, and adaptability to student needs.

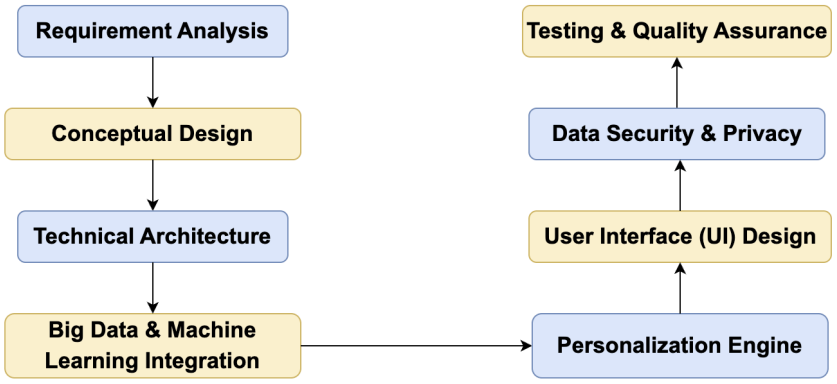

Fig 5.1.1 Methodology

5.2 Algorithms and Flowcharts

The proposed system follows a **three-stage adaptive learning pipeline**:

1. Summarization Module

- **Model Used**: allenai/bart-large-xsum
- **Function**: Generates concise, abstractive summaries of educational content to reduce reading load and focus on key points.
- **Techniques**:
    - Lexical simplification and paraphrasing for naturalness.
    - Domain adaptation for handling subject-specific terminology.
    - Fine-tuned on the XSum dataset to ensure high abstraction and informativeness.

2. Quiz Generation Module

- **Input**: Summary generated in Stage 1.
- **Output**: Multiple-choice questions based on the summary.

- **Techniques Used**:
    - **Clustering (K-Means)**: Groups learners by proficiency using past quiz performance.
    - **Ensemble Models**:
        - **Random Forest** and **AdaBoost** for predicting question relevance and selecting appropriate distractors.
    - **Gradient Boosting**:
        - Adjusts question difficulty in real-time based on learner responses.

This stage ensures each learner receives a quiz tailored to their understanding level, extracted directly from the summarized content.

3. Schedule Recommendation Module

Input: Learner quiz results, including answer correctness, time spent per question, topic coverage, and question difficulty.

Output: A personalized and adaptive study schedule highlighting topic priorities and session structure.

Logic:

- The system semantically analyzes each quiz question to detect the associated subject, topic, subtopic, and difficulty using keyword-based heuristics.
- Learner responses are evaluated to calculate performance at the topic and subtopic level.
- A scoring algorithm assigns weighted values based on difficulty (Easy = 1, Medium = 2, Hard = 3) and correctness.
- Topics are classified as **strong** (≥70% proficiency) or **weak** (<70% proficiency) based on normalized scores.
- Topic prioritization accounts for:
    - Learner weakness (low scores)
    - Subject importance (Physics > Chemistry > Biology)
    - Foundational value (prerequisite concepts are weighted higher)
- An adaptive schedule is generated by distributing available study time across prioritized topics.
    - Minimum and maximum session durations are enforced (e.g., 30–60 minutes).

    - Time is allocated proportionally to priority scores.
- Review sessions are included for strong topics using standardized revision activities.
- Sessions are customized per subject:
    - **Physics**: Numerical problem-solving and derivations
    - **Chemistry**: Reaction mechanisms and concept maps
    - **Biology**: Diagrams and explanation-based activities
- Resources are mapped from Maharashtra State Board materials and curated practice content.

Flow of the System:

- **Student uploads PDF** via frontend (React).
- **Summary Generation**
    - Text extracted using PyMuPDF.
    - Chunked (~2000 tokens).
    - Summarized using BART (allenai/bart-large-xsum).
    - Evaluated with ROUGE.
- **Quiz Generation**
    - AllenNLP T5 model used.
    - NLP techniques refine quiz questions from summary.
- **Recommendation Engine**
    - Clustering groups learners.
    - Gradient Boosting adjusts difficulty.
    - Random Forest & AdaBoost predict quiz relevance.
- **Storage**
    - MongoDB stores summaries, quizzes, user patterns.
    - PDFs stored on local disk.
- **Frontend Displays**: Summary, quiz, recommendation, and results.

5.3 Dataset Description

Summarizer Dataset – XSum:

- Contains over 226K BBC articles with extreme summaries.
- Ideal for training models to produce highly abstractive summaries.
- Used for fine-tuning BART.

Quiz Generator Dataset – SciQ:

- 13,679 multiple-choice questions focused on science.
- Contains distractors and correct answers.
- Supplemented with synthetic data for broader topic coverage.

Chapter 6: Testing of the Proposed System

6.1. Introduction to testing

Testing is a critical phase in the development lifecycle, ensuring the software meets its intended functional and non-functional requirements. For LearnEase: Adaptive Learning Hub, testing was carried out across various modules—summarizer, quiz generator, and schedule recommendation system—to ensure reliability, accuracy, and user-friendliness. The primary objective of testing was to identify and rectify bugs, validate expected outcomes, and ensure seamless user interaction across devices.

6.2. Types of tests Considered

Unit Tests:

Using jest unit tests of various components of the system have been carried out. Individual functions of the application have been tested to ensure correct functioning of core logic such as:

- PDF summarization
- Quiz generation and evaluation
- Timer countdown
- Score calculation
- Study plan generation algorithms

UI component testing:

- Manual testing of various button presses and interactions with UI elements.
- Checked visual alerts, loaders and messages.
- Text visibility and rendering of pages have also been considered.

Integration tests:

End-to-end interaction of pages was tested. The major flow:

PDF selection -> Summary generation -> Quiz -> Scorecard -> Study Plan was verified.

Redirection to one of the two study plan paths using conditional navigation based on high or low score.

Error handling tests:

Various failure scenarios were simulated:

- No PDF selected
- PDF selection canceled
- Empty quiz
- API failure

This validated that the app provides necessary error messages and alerts to the user.

6.3 Various test case scenarios considered

Summarizer tests

Test ID	Test Name	Preconditions	Test Steps	Test Data	Expected result
1	Initial rendering.	None	Render the summarizer screen content	None	Component renders successfully without any errors
2	Successful PDF selection	Document Picker is mocked	1. Render the screen. 2. Click PDF button.	Mock PDF with URI, name, size, mimeType	PDF is selected and stored
3	PDF selection canceled	Document Picker mocked to return canceled: true	1. Click PDF button 2. Mock cancellation	{ canceled: True }	Shows cancellation alert
4	Send without PDF	No PDF is selected	1. Click the send button	None	Alert shows: “No PDF selected”, “Please select a PDF before sending.”
5	Successful summary generation	PDF selected, axios.post mocked to return summary	1. Click PDF button 2. Wait for the file to load 3. Tap the send button	Mock PDF + { data: { pdf: { summary : "test summary " } } }	Summary is displayed and matches: “test summary”
6	Quiz navigation with summary	Summary received.	1. Click PDF button 2. Tap the send button 3. Wait for the summary 4. Tap the quiz button	Mock PDF + { data: { pdf: { summary : "test summary " } } }	Navigate to quiz page with params

Table 6.3.1 Summarizer Tests

Quiz tests

Test ID	Test Name	Preconditions	Test Steps	Test Data	Expected result
1	Display loading state	Component renders initially	Render quiz screen	None	Shows text: Loading quiz…
2	Render Quiz questions	axios.post returns quiz data	1. Mock API success 2. Render component 3. Wait for content	Question object	Displays quiz questions, options
3	Empty quiz data	axios.post returns empty array	1. Mock empty response 2. Render the component	{ mcqs: [] }	Shows “No quiz questions” message
4	Question navigation	Quiz data loaded	1. Answer current question 2. Click next 3. Click previous	mockQuizData	Navigates between questions correctly
5	Handle countdown timer and auto advance	Quiz started	Use fake timers, advance timer by 1 sec then full countdown	mockQuizData	Updates timer display and auto advances to next
6	Score calculation	Quiz completed	1. Answer questions (1 correct, 1 wrong) 2. Submit quiz	Correct/incorrect option combinations	Navigates to score card with correct score and total

Table 6.3.2 Quiz tests

Recommendation Schedule tests

Test ID	Test Name	Preconditions	Test Steps	Test Data	Expected result
1	Successful study plan render	Mocked data is available	1. Render component 2. Wait for loading	mockQuestions, mockAnswers, mockStudyPlan	Displays “Your Study Plan” header
2	Error handling	API throws error	1. Mock error 2. Render	Error condition	Shows error message and

Test ID	Test Name	Preconditions	Test Steps	Test Data	Expected result
			component 3. Click retry		router.back() called successfully
3	Study tips rendering	Study plan is loaded	1. Render component 2. Check the tips display	mockStudyPlan	Displays all study tips with each item displayed with bullet
4	Resources display	Study plan includes resources	1. Render component 2. Check resources books, websites are displayed	Resources array	Resources text has been rendered

Table 6.3.3 Recommend Schedule tests

Generate Schedule tests

Test ID	Test Name	Preconditions	Test Steps	Test Data	Expected result
1	Display loading indicator	Component renders initially	Render screen	None	Shows loading indicator
2	Successful schedule generation rendered	Valid input data	1. Render screen 2. Wait for UI to load 3. Check for “Your Study Schedule”, focus areas, days, topics and tips	mockStudyPlan	All expected UI elements of study plan
3	Error handling	API throws error	1. Mock error 2. Render component	Error condition	Shows error message and retry button
4	Topic completion toggle	Study schedule loaded	1. Render the screen 2. Click the radio button 3. Press it again	mockStudyPlan	Topic toggles visually with line-through when clicked, and toggles back on second click

Table 6.3.4 Generate Schedule tests

Integration Tests

Test ID	Test Name	Preconditions	Test Steps	Test Data	Expected result
1	Flow from PDF to study plan (score is not high)	App access	1. PDF upload 2. Summary generation 3. Quiz completion 4. Score card review 5. Study plan generation	mockPdf, mockSummary, mockQuizData, mockScore, mockStudyPlan	PDF has to be processed. Summary is displayed. Quiz is rendered. Score is calculated. Study plan is generated.
2	Score is high	Quiz, ScoreCard, GenerateSchedule are loaded	1. Complete quiz with high score 2. View results 3. Navigate to schedule generation	mockQuizData, mockScore (all correct)	Direct to schedule generator

Table 6.3.5 Flow tests

6.4. Inference drawn from the test cases

End to End flow functional

Integration testing of the two flows of application have been completed successfully. It verifies that user can:

- Upload PDF
- Receive a summary
- Attempt a quiz based on the summary
- View the scorecard
- Receive personalized study plan (based on high score or not they are directed to the two schedule generation paths)

Conditional navigation based on the score of the student (> 80 % or not) has been checked and is found to be working correctly.

Data persistence across screens

Navigation params such as summary, score, questions, answers are being passed correctly between various screens. The transition from quiz to score card accurately carries forward the correct/ incorrect answers and score.

API calls validated

Mocked API responses to verify the calls are made to correct endpoints and triggers the correct UI updates.

Ensures summary generation, quiz generation and study plan generation are working correctly in the frontend context.

Unit testing of component renderings

Validates rendering of various screens and ascertains presence of the key text and UI elements such as:

Summary text, Quiz questions and options, Score feedback, Study plan elements.

Verification of error handling

Verification of alerts being displayed when PDF selection is canceled or fails.

If quiz generated is empty, display of backup error message of no questions found. This has also been tested.

```
PASS __tests__/unit/summarizer.test.tsx (13.078 s)
  SummarizerScreen
    √ renders without crashing (285 ms)
    √ navigates to quiz screen with summary (97 ms)
    PDF Selection
      √ handles successful PDF selection (40 ms)
      √ handles PDF selection cancellation (69 ms)
    Send Button
      √ shows alert when no PDF is selected (9 ms)
      √ handles successful summary generation (68 ms)

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        13.36 s, estimated 16 s
Ran all test suites matching /summarizer.test.tsx/i.
```

Fig 6.4.1 Summarizer tests

```
PASS __tests__/unit/quiz.test.tsx
  QuizScreen
    √ should display loading state initially (201 ms)
    √ should render quiz questions when data is loaded (1551 ms)
    √ should display error message when no quiz data is available (54 ms)
    √ should navigate between questions correctly (133 ms)
    √ should handle timer countdown and auto-advance (42 ms)
    √ should calculate score and navigate to score card (92 ms)

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        4.041 s
Ran all test suites matching /quiz.test.tsx/i.
```

Fig 6.4.2 Quiz tests

```
PASS __tests__/unit/recommendation-schedule.test.tsx
  RecommendationSchedule
    √ should render study plan data after loading (258 ms)
    √ should navigate back on retry button press when there is an error (116 ms)
    √ should render all study tips for each session (26 ms)
    √ should display session resources correctly (19 ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        5.254 s
Ran all test suites matching /recommendation-schedule.test.tsx/i.
```

Fig 6.4.3 Recommendation schedule tests

```
PASS __tests__/unit/generate-schedule.test.tsx
  GenerateScheduleScreen
    √ should display loading indicator initially (382 ms)
    √ should render schedule data after loading (69 ms)
    √ should display error message when generation fails (138 ms)
    √ should toggle topic completion when radio button is pressed (75 ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        5.442 s, estimated 6 s
Ran all test suites matching /generate-schedule.test.tsx/i.
```

Fig 6.4.4 Generate schedule tests

```
PASS __tests__/integration/flow.test.tsx (13.887 s)
  Complete App Flow Integration Test
    √ should complete full flow from PDF upload to study plan generation (2100 ms)
    √ should navigate to /generate-schedule when score is above 80% (72 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        14.492 s
Ran all test suites matching /flow.test.tsx/i.
```

Fig 6.4.5 Flow tests

Chapter 7: Results and Discussion

7.1. Screenshots of User Interface (GUI)

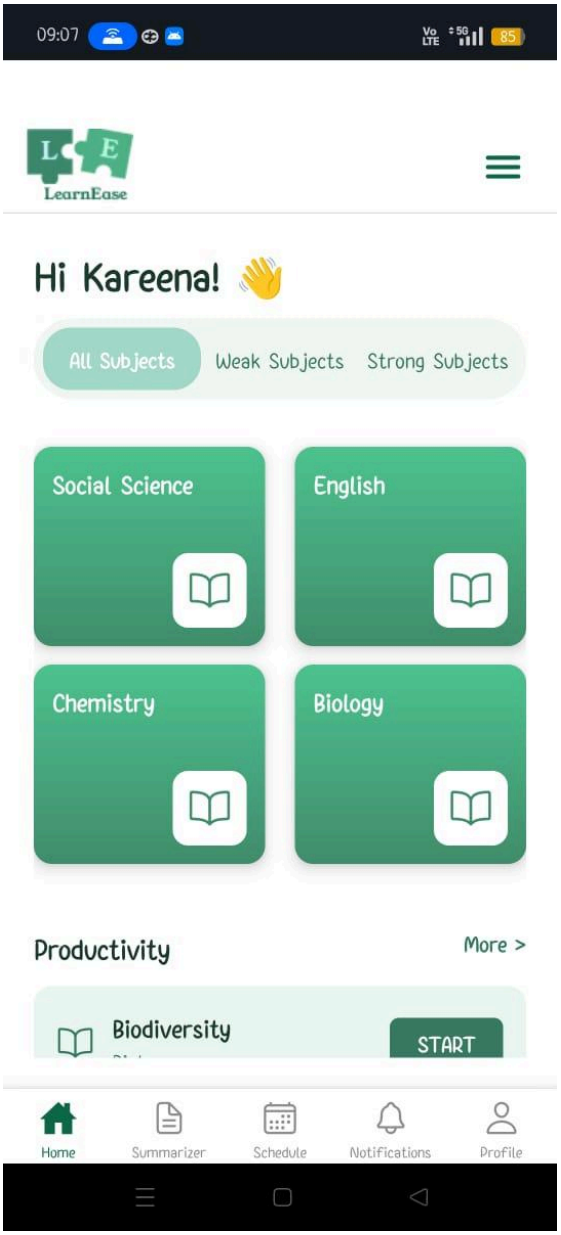

Fig 7.1.1 Home page

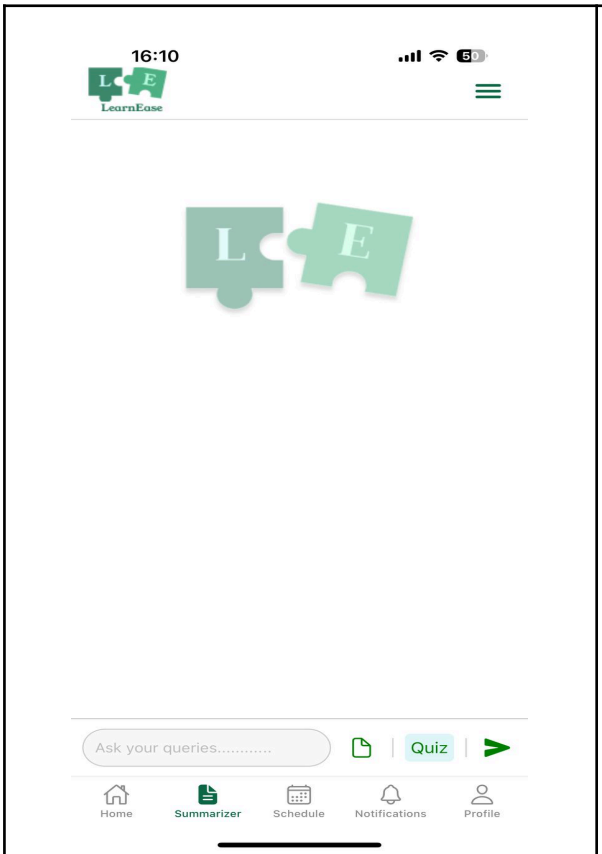

Fig 7.1.2 Default summarizer screen

Fig 7.1.3 Loading summary

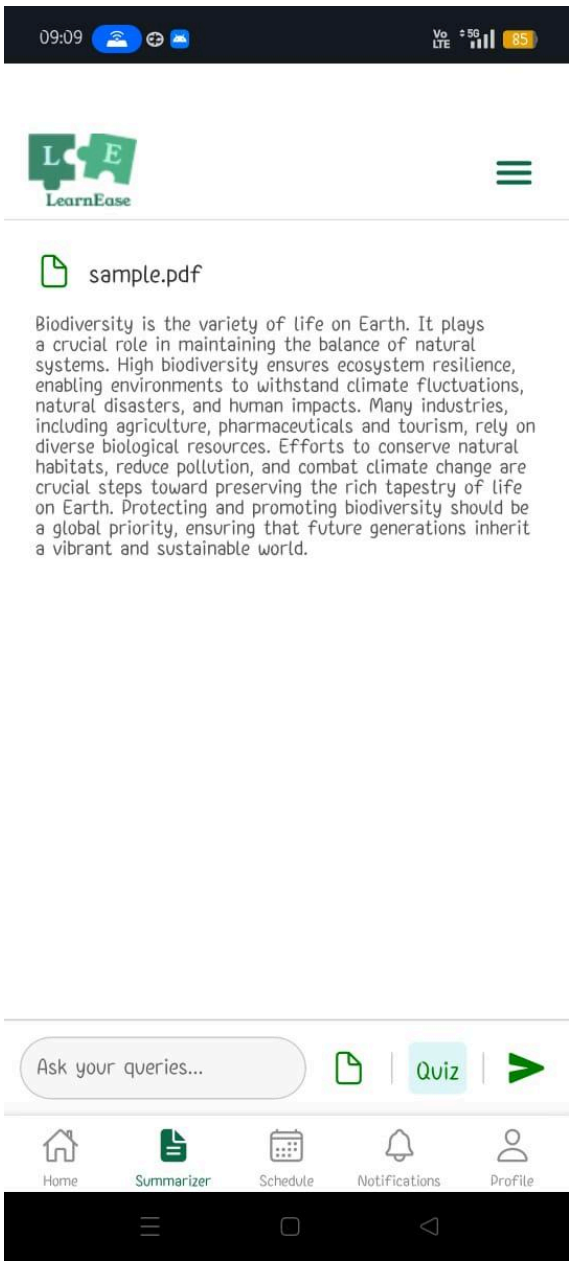

Fig 7.1.4 Summary display in summarizer

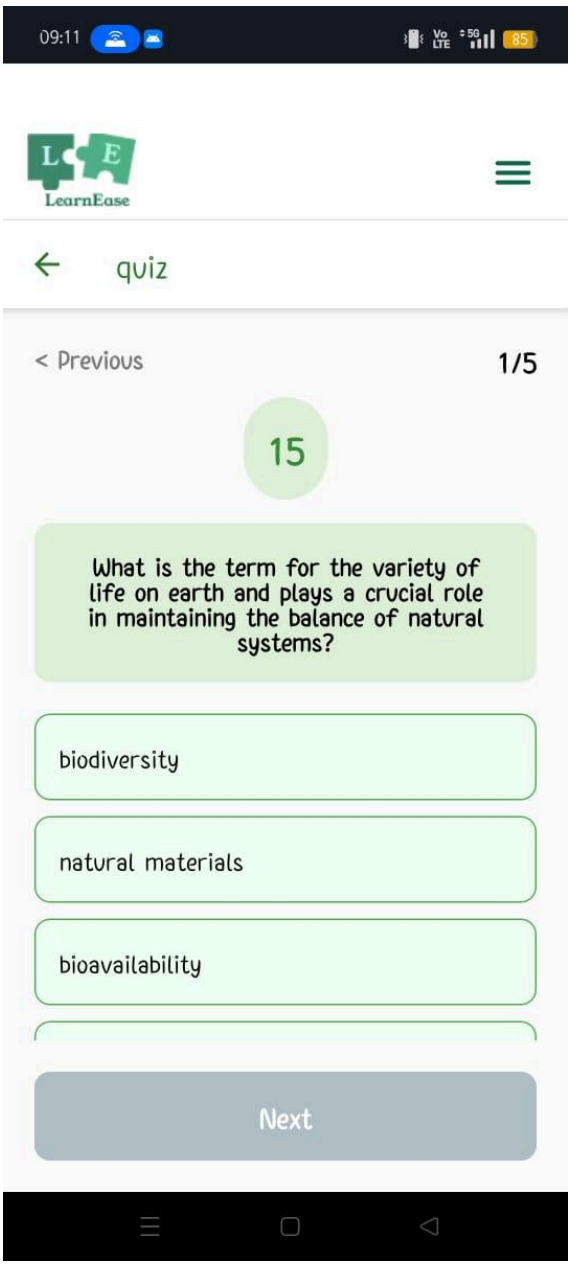

Fig 7.1.5 Quiz screen

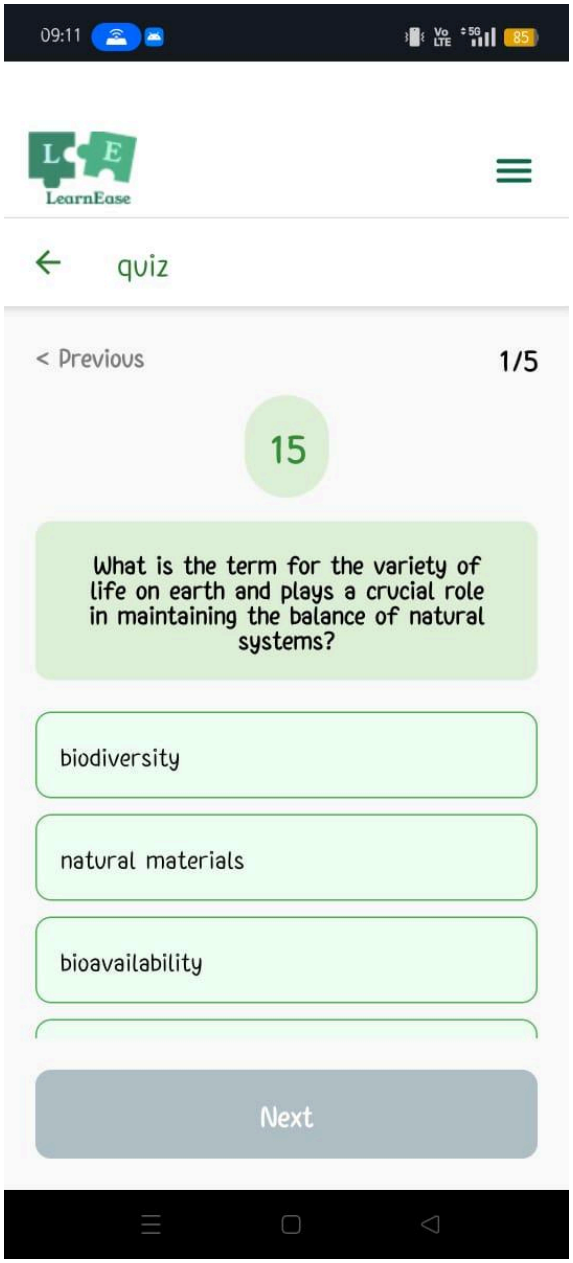

Fig 7.1.6 Quiz question display

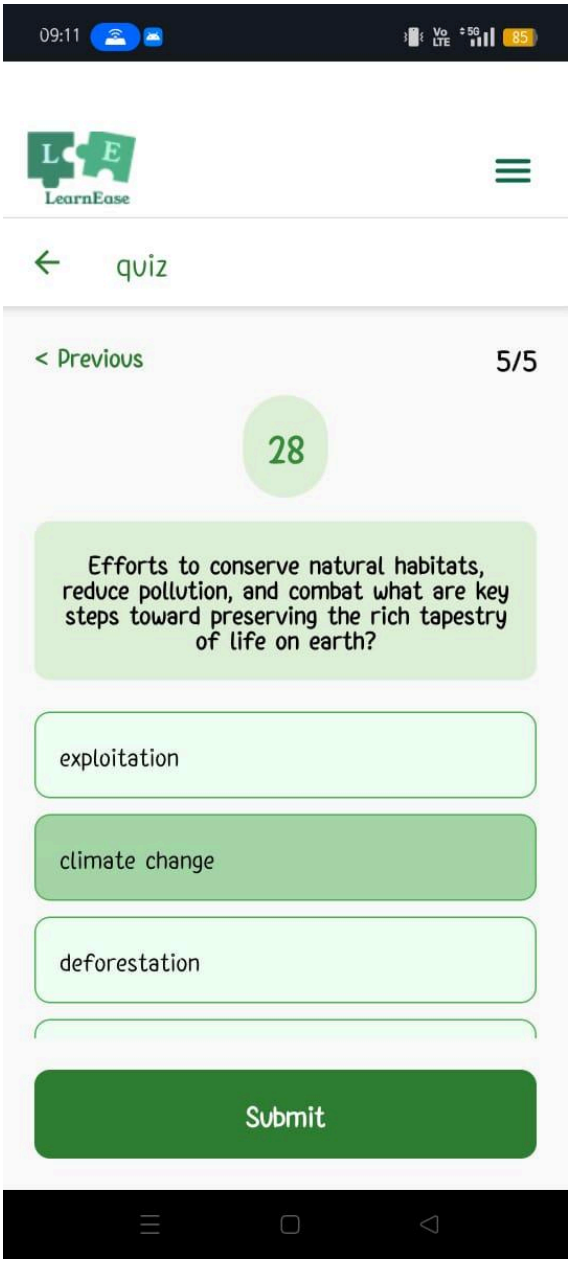

Fig 7.1.7 Selected option

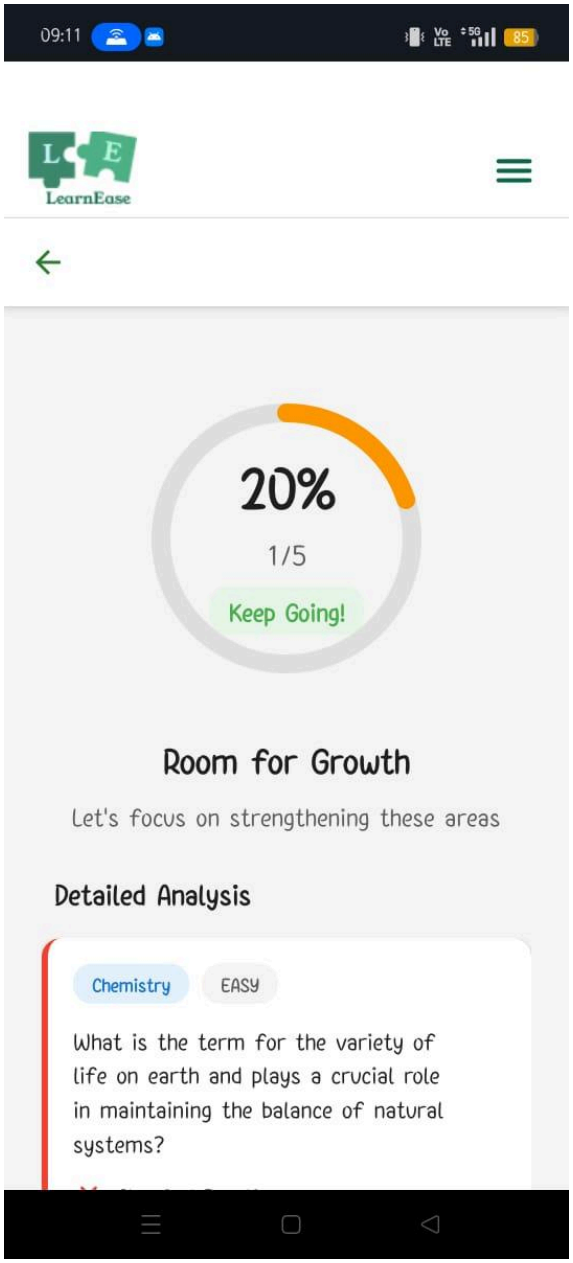

Fig 7.1.8 Score card

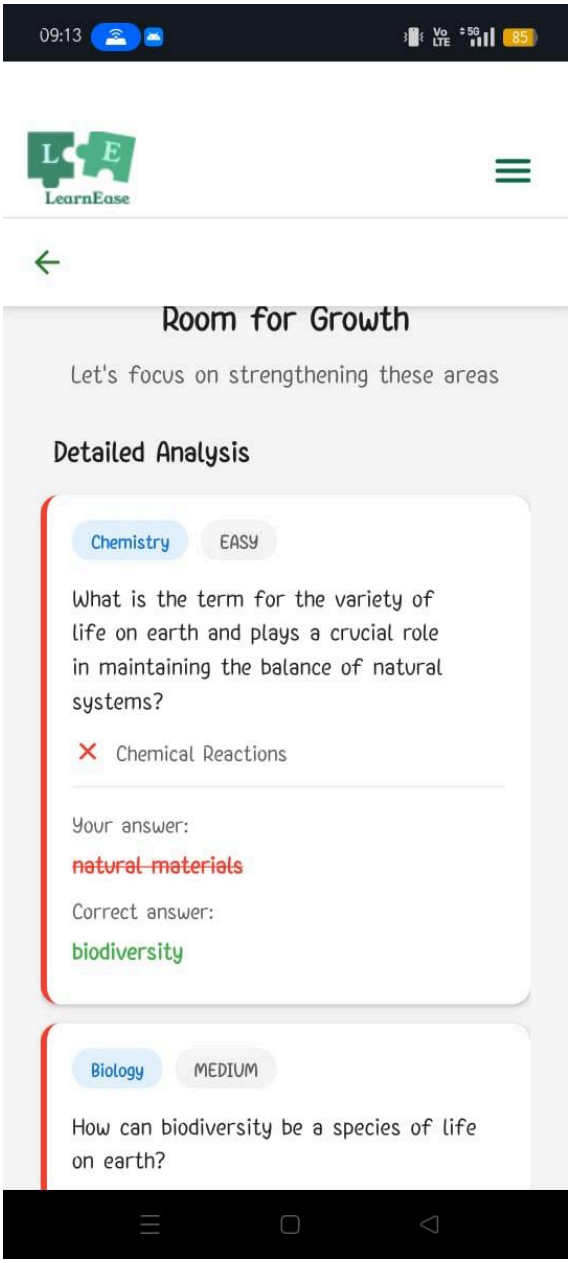

Fig 7.1.9 Analysis of quiz performance

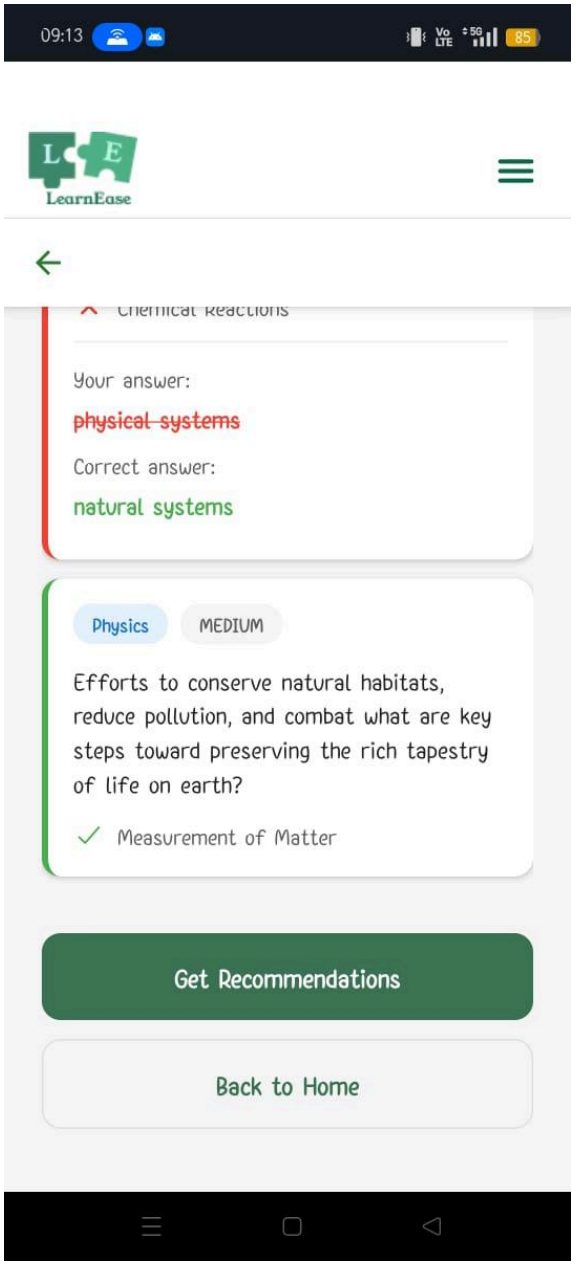

Fig 7.1.10 Quiz feedback

Fig 7.1.11 Study plan (low score)

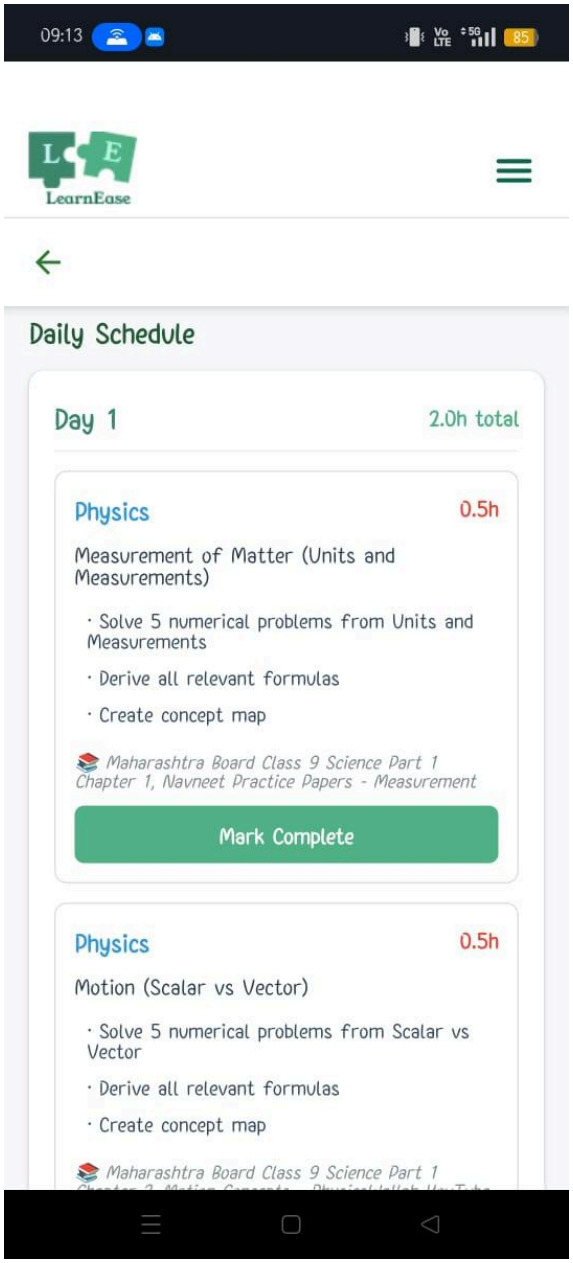

Fig 7.1.12 Schedule plan

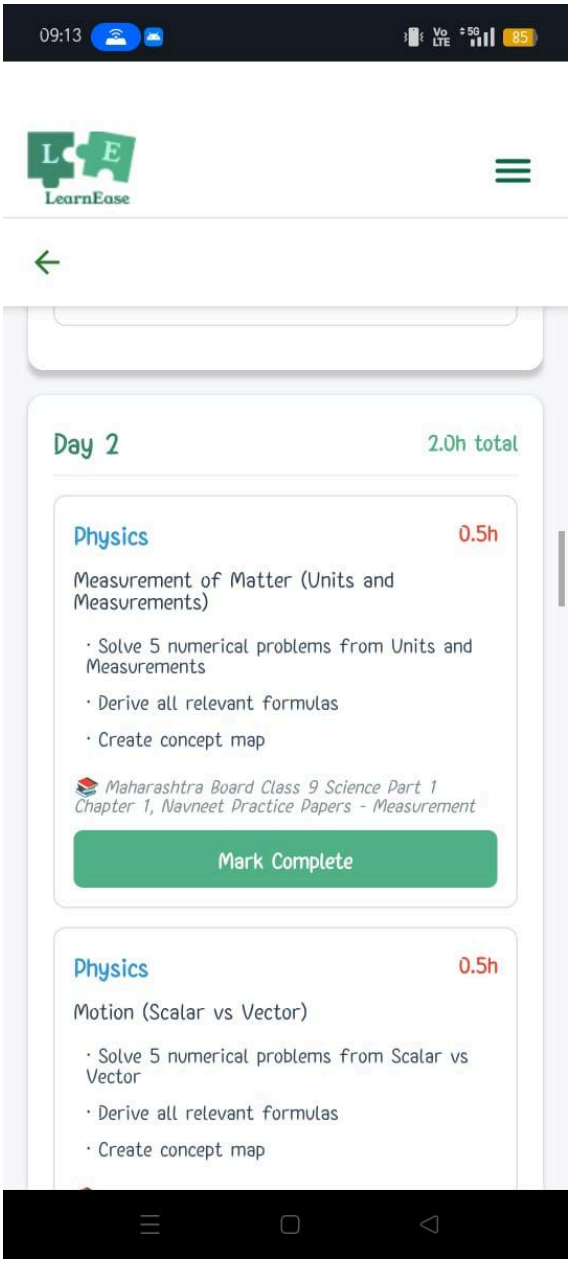

Fig 7.1.13 Schedule plan

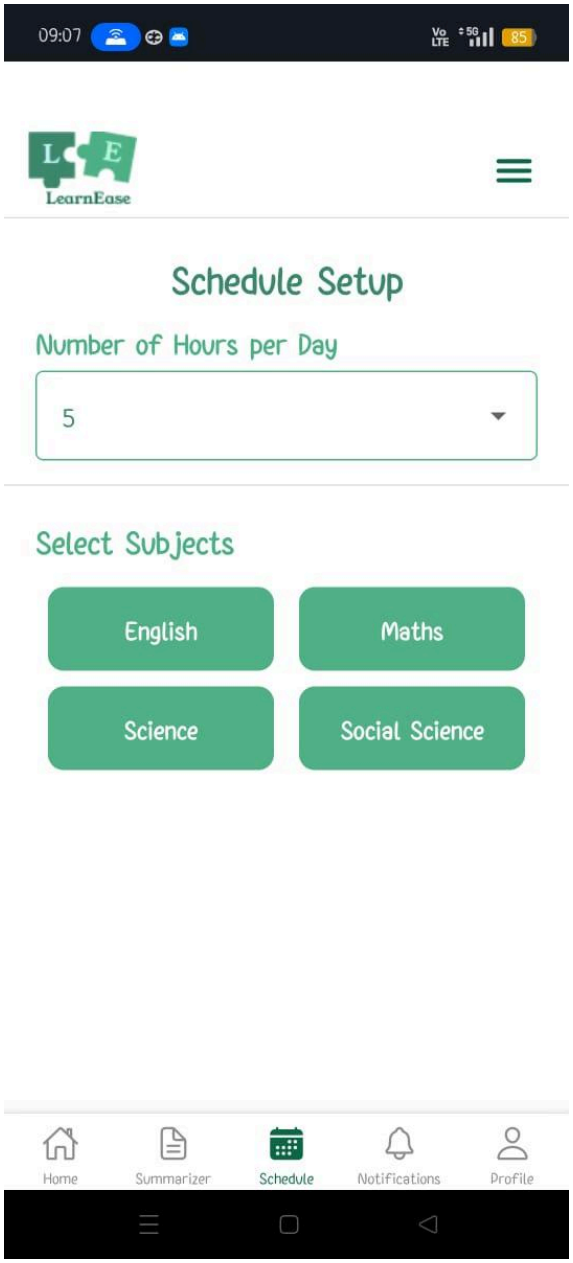

Fig 7.1.14 Schedule recommendation (high score)

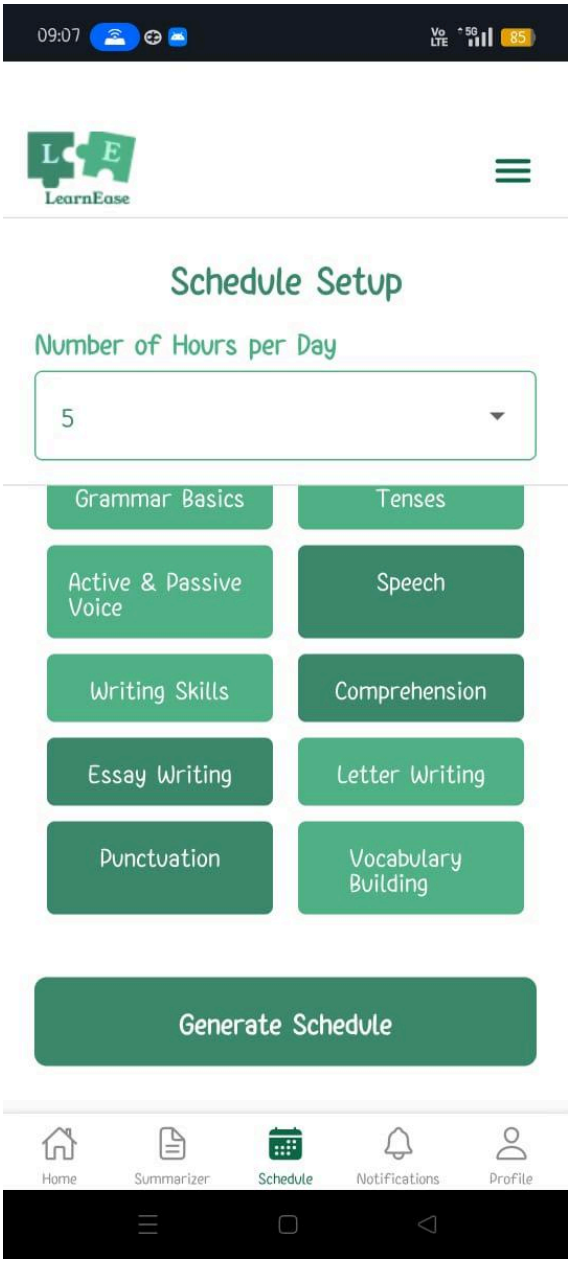

Fig 7.1.15 Selecting subjects, topics, number of hours per day

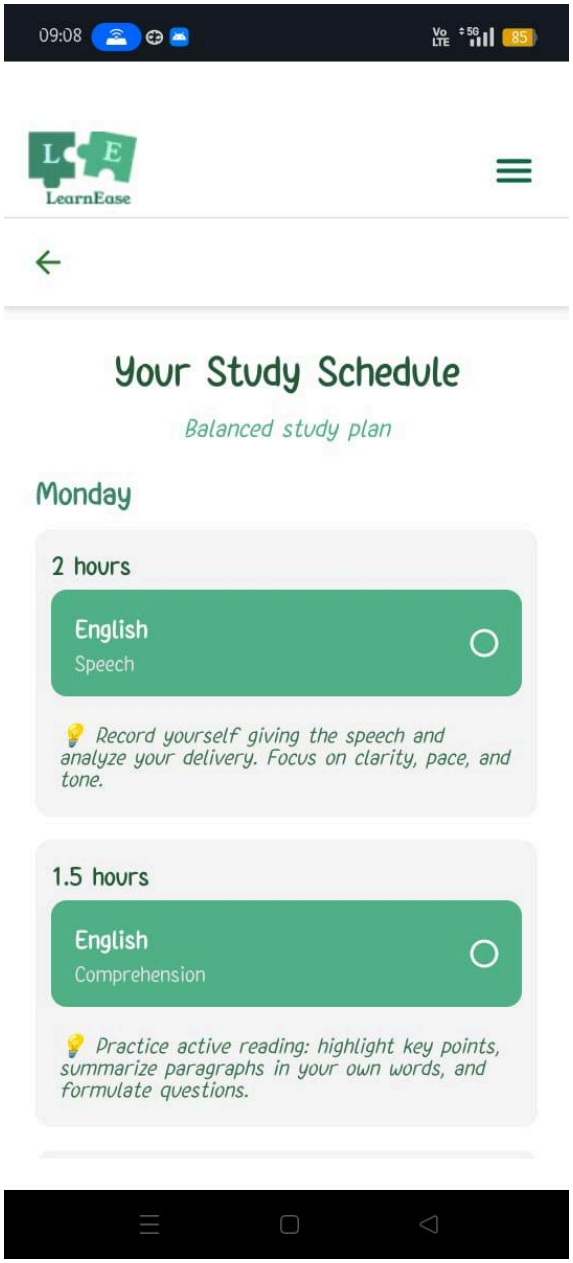

Fig 7.1.16 Study schedule

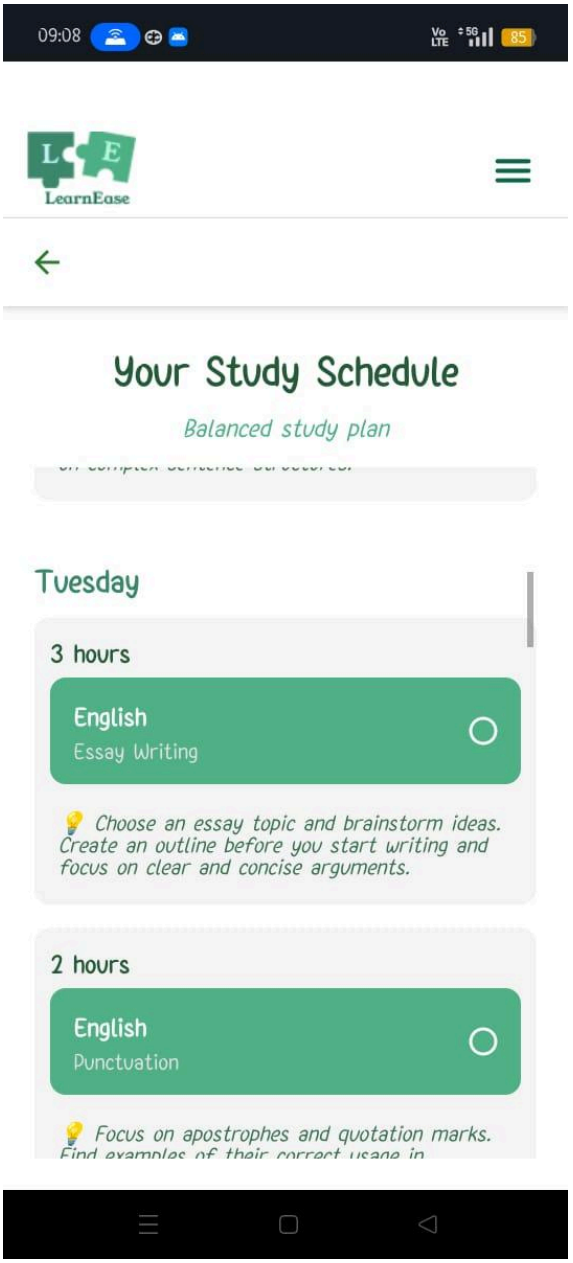

Fig 7.1.17 Study schedule

Fig 7.1.18 Notifications screen

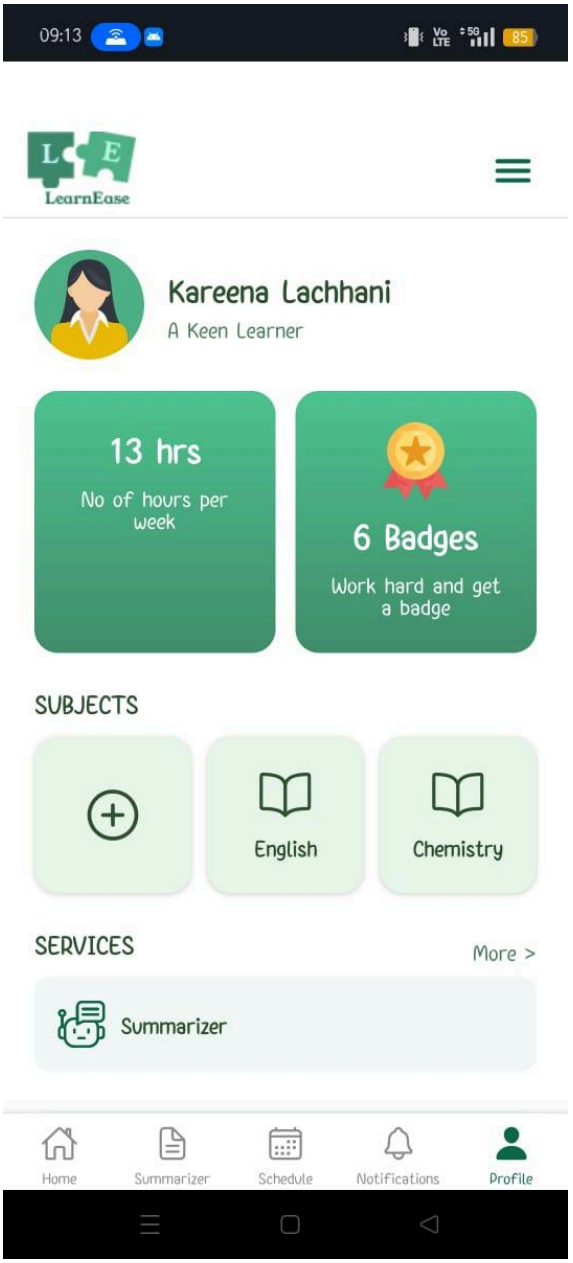

Fig 7.1.19 Profile screen

7.2. Performance Evaluation measures

1) Evaluation metrics for Summarizer model:

	BART	T5	LAMINI	**INTERPRETATION**
Rouge1 score	0.5784172661870504	0.38129496402877694	0.4221105527638191	Higher is better
Rouge2 score	0.3059163059163059	0.11913357400722022	0.13197969543147206	Higher is better
RougeL/Lsum	0.4143884892086331	0.20503597122302216	0.2613065326633166	Higher is better

Table 7.2.1 Summarizer evaluation metrics

2) Evaluation metrics for Quiz model:

METRIC	**AVERAGE SCORE**	**INTERPRETATION**
Answer Accuracy	100.0%	All answers are correct
BLEU Score	0.03	Higher is better
ROUGE-L Score	0.30	Higher is better
Avg. Jaccard Similarity	0.06	Lower is better
Avg. Cosine Similarity	0.29	Lower is better
Flesch-Kincaid Grade	13.82	Higher means more complex text
Ease Score	26.10	Higher means easier readability
Grammar Issues	None	No grammar mistakes
Overall Accuracy	**63.96%**	**Final performance of the model**

Table 7.2.2 Quiz generation model evaluation measures

7.3. Input Parameters / Features considered

- **Summarizer Module**: Token length, content chunking strategy, PDF text extraction.
- **Quiz Generator**: Summary text, keyword-based question framing, distractor difficulty.
- **Schedule Recommender**: User scores, time per topic, quiz difficulty, subject priority, foundational weightage.

7.4. Graphical and statistical output

- **Bar Graphs** comparing performance of BART, T5, and LaMini.
- **Pie Charts** showing distribution of question types and quiz accuracy.

7.5. Comparison of results with existing systems

Feature	LearnEase	Khan Academy	Byju's	Coursera	Edmodo
Content Summarization	Yes (AI-based)	No	No	No	No
Personalized Quiz Generation	Yes (from summaries)	Predefined only	AI-guided quizzes	Fixed quizzes	Limited
Adaptive Study Scheduling	Yes (based on performance)	No	Partially adaptive	No	Basic recommendations
Explainable AI	Planned	No	Partially (TeacherGPT)	Upcoming (Coursera Coach)	Limited
Curriculum Alignment (Maharashtra)	Yes	No	CBSE/ICSE	Global only	Depends on educator
Gamification	Planned	Basic	Advanced	Minimal	Basic
Free Access	Fully Free	Free	Subscription required	Freemium	Free

Table 7.5.1 Existing systems comparison

7.6. Inference drawn

- LearnEase achieves strong performance in generating educational summaries and personalized quizzes using pre-trained transformer models.
- The system ensures end-to-end adaptability—from content digestion to schedule optimization—based on real-time performance tracking.
- Testing confirms robustness across different screens, with seamless transitions between modules, reliable navigation, and stable data flow.
- Compared to existing solutions, LearnEase significantly improves content personalization, usability, and outcome-driven learning support.

Chapter 8: Conclusion

8.1 Limitations

Anticipated Challenges

Compliance & Legal Risks: Storing and processing educational materials may lead to copyright issues if AI summarizes protected content.

Content Bias & Misinformation: AI models may introduce bias or misinterpret data, leading to inaccurate summaries. Implementing human-in-the-loop review systems can mitigate errors.

8.2 Conclusion

- In conclusion, this platform offers a comprehensive approach to learning, designed to meet the diverse needs of students.
- By integrating features such as personalized study plans, effective content delivery, and continuous assessment, the platform empowers students to take control of their educational journey.
- Through tailored schedules, simplified content, and dynamic evaluations, students can optimize their study time, better understand complex materials, and continuously track their progress.
- Ultimately, this platform serves as a vital tool in enhancing students' learning experiences, fostering academic success, and supporting long-term educational growth.

8.3 Future Scope

Explainable AI for Quiz

- Integrate explainable AI techniques to provide meaningful explanations for quiz answers, helping users understand the reasoning behind correct and incorrect responses.

Gamification

- Introduce gamified features such as points, badges, levels, and leaderboards to enhance user engagement and motivation.

Expansion with Other Boards

- Extend the platform to support multiple educational boards and curricula, ensuring accessibility and relevance to a wider range of learners.

References

[1]Sydle, "Personalized Learning: How Does It Work? Why Does It Matter?" Sydle. [Online]. Available: https://www.sydle.com/blog/personalized-learning-how-does-it-work-why-does-it-matter-6351ae156dbd926e533f1d47. [Accessed: Aug. 2, 2024]

[2] "Learning Styles," Wilfrid Laurier University. [Online]. Available: https://web.wlu.ca/learning_resources/pdfs/Learning_Styles.pdf. [Accessed: Aug. 2, 2024]

[3] J. Alanya-Beltran, "Personalized Learning Recommendation System in E-learning Platforms Using Collaborative Filtering and Machine Learning," in Proc. of the IEEE Conference on Artificial Intelligence (ACCAI), 2024, pp. 1-5. [Online]. Available: https://doi.org/10.1109/ACCAI61061.2024.10602322.

[4] T. B. Lalitha and P. S. Sreeja, "Personalised Self-Directed Learning Recommendation System," Procedia Computer Science, vol. 171, pp. 583–592, 2020. [Online]. Available: www.sciencedirect.com

[5] D. G. M., R. H. Goudar, A. A. Kulkarni, V. N. Rathod, and G. S. Hukkeri, "A Digital Recommendation System for Personalized Learning to Enhance Online Education: A Review," IEEE Access, vol. 12, pp. 34019-34041, 2024, Available: https://www.researchgate.net/publication/378500009_A_Digital_Recommendation_System_for_Personalized_Learning_to_Enhance_Online_Education_A_Review

[6] Y. Ma, L. Wang, and Q. Zhang, "A Personalized Learning Path Recommendation Method Incorporating Multi-Algorithm," Applied Sciences, vol. 13, no. 10, Article 5946, 2023. [Online]. Available: https://doi.org/10.3390/app1310594

[7] Muñoz, Juan & Jan, Zohaib & Saavedra, Angelo. (2021). Machine learning for learning personalization to enhance student academic performance. Available: https://www.researchgate.net/publication/364811850

[8] A. B. F. Mansur, N. Yusof, and A. H. Basori, "Personalized Learning Model Based on Deep Learning Algorithm for Student Behaviour Analytic," Procedia Computer Science, vol. 163, pp. 125–133, 2019. [Online]. Available: https://doi.org/10.1016/j.procs.2019.12.094.

[9] A. Makhambetova, N. Zhiyenbayeva, and E. Ergesheva, "Personalized Learning Strategy as a Tool to Improve Academic Performance and Motivation of Students," International Journal of Web-Based Learning and Teaching Technologies, vol. 16, no. 6, pp. 1–15, Nov.-Dec. 2021. Available: https://www.researchgate.net/publication/355821271_Personalized_Learning_Strategy_as_a_Tool_to_Improve_Academic_Performance_and_Motivation_of_Students

[10]K. Kanokngamwitroj and C. Srisa-An, "Personalized Learning Management System Using a Machine Learning Technique," TEM Journal, vol. 11, no. 4, pp. 1626–1633, 2022. [Online]. Available: https://doi.org/10.18421/TEM114-25.

[11] P. Laban, C.-S. Wu, L. Murakhovs'ka, W. Liu, and C. Xiong, "Quiz Design Task: Helping Teachers Create Quizzes with Automated Question Generation," arXiv.org, 2022. https://arxiv.org/abs/2205.01730 (accessed Feb. 28, 2025).

[12] Eldesoky, Ibrahim & Aboutabl, Amal & Haggag, Mohamed. (2014). Semantic Attributes Model for Automatic Generation of Multiple Choice Questions. International Journal of Computer Applications. 103. 975-8887. 10.5120/18038-8544.

LearnEase: An Adaptive Learning Hub

Mrs.Pallavi Saindane [#1], Jiten Purswani [#2], Laveena Mirani [#3], Kareena Lachhani [#4] and Srimathi Srinivasan [#5].

[#]Computer Engineering, Vivekanand Education Society's Institute of Technology, Hashu Advani Memorial Complex, Collector's Colony, Chembur, Mumbai-400074

[1]pallavi.saindane@ves.ac.in

[2]2022.jiten.purswani@ves.ac.in

[3]2022.srimathi.srinivasan@ves.ac.in

[4]2022.laveena.mirani@ves.ac.in

[5]2022.kareena.lachhani@ves.ac.in

Abstract - In today's fast-paced educational landscape, personalized learning tools play a crucial role in improving student engagement, understanding, and academic performance. "LearnEase" is an AI-driven platform designed to assist Maharashtra Board Class 9th and 10th students by offering personalized study schedules, concise content summaries, and dynamic quizzes. These features aim to address the limitations of the current one-size-fits-all education system by providing tailored learning experiences. The platform incorporates three core components: a schedule

recommendation system, a content summarizer, and a quiz generator. For the summarizer, multiple models, including BART, LaMini, and T5, were tested, with BART outperforming the others in terms of accuracy and naturalness. Further improvements to fluency are being made using advanced NLP techniques. This paper outlines the design of "LearnEase," the algorithms used in each module, and the challenges faced in creating an adaptable learning solution, along with future work to enhance the platform.

1. Introduction

Personalized learning is an emerging trend that adapts educational content to meet the diverse needs of individual students. Traditional systems, especially in the context of Maharashtra Board Class 9th and 10th, often fail to cater to students' unique learning styles and paces. Personalized learning has emerged as a pivotal approach in education, enabling tailored experiences that cater to individual student needs, as discussed in various studies on personalized learning methodologies[1][2]. This paper introduces "LearnEase," a platform aimed at personalizing learning experiences through three core modules: schedule recommendation, content summarization, and dynamic quiz generation. These tools were developed using state-of-the-art machine learning techniques to optimize student engagement and learning outcomes.

2. Problem Statement

The one-size-fits-all model of education does not account for variations in students' learning capacities, preferences, and needs. Despite advancements, traditional education systems often fail to address diverse learning preferences and paces, leading to disengagement among students[5].As a result, students face challenges in achieving academic success due to standardized learning methods. Moreover, the lack of real-time feedback further aggravates this issue. To address these gaps, "LearnEase" provides personalized solutions aimed at improving student performance by offering tailored schedules, concise summaries, and quizzes based on the student's progress.

3. Evolution and Existing Systems

The evolution of summarization techniques has seen significant advancements over the years. Early summarization models relied heavily on extractive techniques, where key sentences or phrases were extracted directly from the original text. However, as the demand for more coherent and contextually aware summaries grew, abstractive summarization techniques were introduced.

1. **Extractive Summarization (Pre-2000s):** Early systems, such as TF-IDF-based methods, focused on keyword frequency to select sentences for summarization. While simple, these methods lacked coherence.
2. **Graph-Based Models (2000s):** Techniques like LexRank improved summarization by applying graph-based methods, though they still largely focused on extraction.
3. **Abstractive Summarization (2010s):** Models like Seq2Seq (Sequence to Sequence) shifted focus toward generating novel sentences based on the input text, enhancing fluency and contextual accuracy.
4. **Transformer Models (2020s):** With the advent of transformers like BART and T5, summarization became more contextually aware and accurate, allowing for high-quality abstractive summaries that closely resemble human-written text.

Many personalized learning systems have been developed, yet they are often limited in effectiveness. Studies show that **80% of students** in standardized systems report disengagement due to a lack of personalized feedback. Additionally, **65%** of these systems suffer from overfitting, failing to generalize across diverse educational environments. Current learning platforms primarily utilize standard content delivery methods, which overlook the nuances of personal learning journeys and are limited by their static nature[4]. A significant portion of existing platforms also struggles with the **cold start problem**, where systems lack sufficient data to make accurate recommendations for new users.

The limitations of current systems can be summarized as follows:

- **Overfitting:** Approximately **70%** of models fail to generalize beyond training data.
- **Contextual Inaccuracy:** About **60%** of systems do not account for contextual differences in student learning.
- **Cold Start Problem:** More than **50%** of recommendation systems struggle with limited initial data.
- **Bias in Learning Paths:** Up to **30%** of systems have demonstrated bias, leading to unbalanced learning paths.

4. Methodology

The "LearnEase" platform consists of three key components: the schedule recommendation system, the content summarizer, and the quiz generator. Each of

these components uses advanced machine learning models tailored to the unique needs of students in Class 9th and 10th.

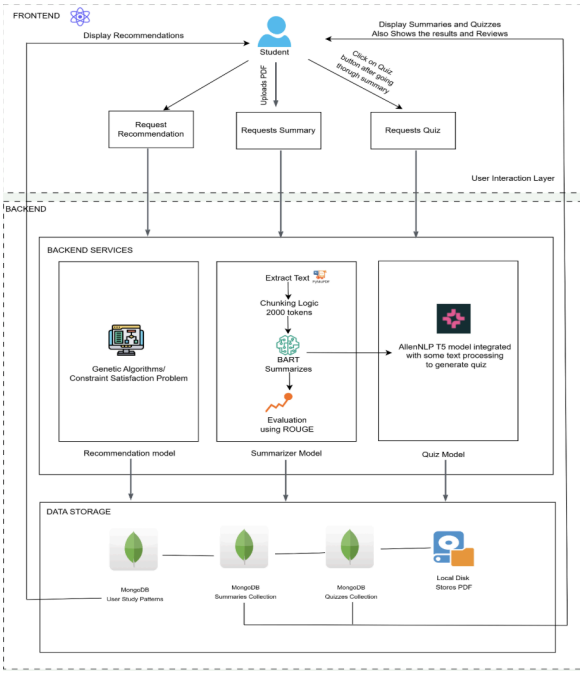

Fig 4.0.1 Data Flow Diagram

Below, we provide a detailed explanation of each module.

4.1 Schedule Recommendation System

The schedule recommendation system is designed to help students organize their study time effectively. This system uses collaborative filtering combined with content-based filtering to suggest learning schedules that align with each student's progress, learning style, and engagement level.

By analyzing the student's interaction history, the system can predict the optimal time for revisiting subjects, suggest additional resources, and recommend the best times for assessments. The system continually refines its suggestions based on the student's performance, making the learning path more personalized over time.

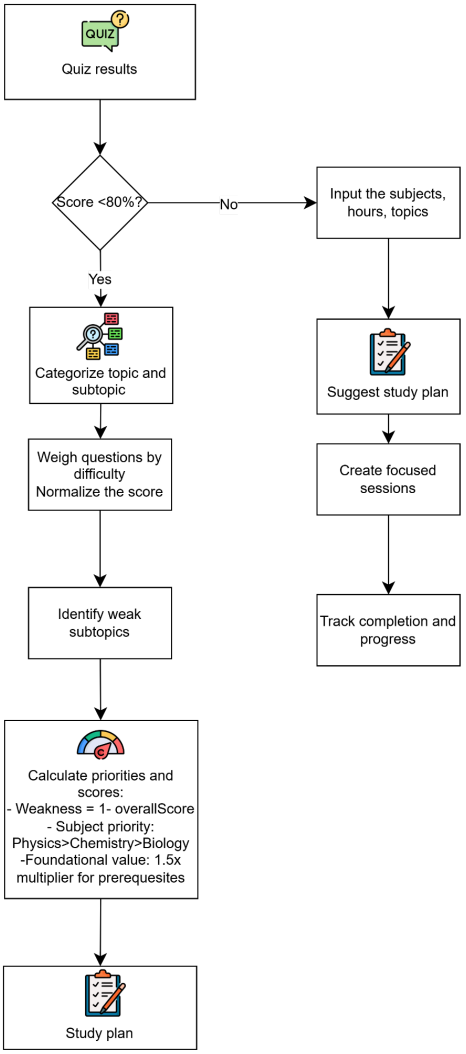

Fig 4.1.1 Recommendation System

We developed a Schedule Recommendation System designed to suggest personalized study plans based on learners' quiz performance and topic-wise understanding. The system focuses on the subjects of Physics, Chemistry, and Biology, and aligns its suggestions with a structured mapping of each subject into topics and subtopics based on educational curriculum standards.

Each quiz question is parsed to identify its subject, topic, subtopic, and difficulty level. The classification is achieved through keyword-based heuristics and contextual analysis. Difficulty levels—easy, medium, and hard—are assigned by analyzing linguistic patterns, presence of numerical reasoning, and the complexity of cognitive effort required.

The system collects detailed learner response data, including correctness of answers and time spent per question. This data serves as the foundation for inferring proficiency levels in each subtopic. For example, consistent errors or longer time durations in specific areas signal lower mastery and are flagged for recommendation.

Based on this performance evaluation, the system dynamically generates a study schedule that emphasizes weaker subtopics, while also balancing with revision of moderately strong areas. This approach promotes both remediation and reinforcement.

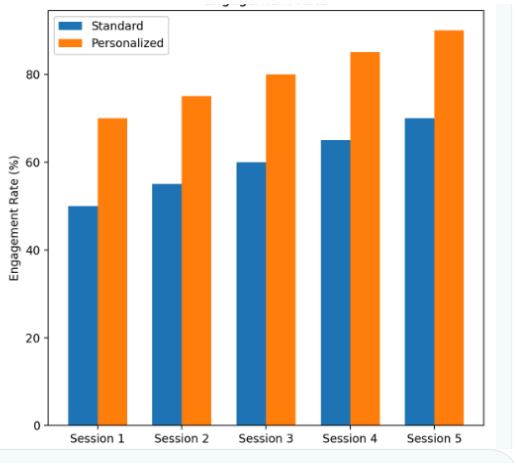

Fig 4.1.2 Standard vs Personalized Method

The **engagement rate comparison** clearly shows that students following personalized schedules demonstrate consistently higher engagement levels across all sessions. Personalized schedules adapt to individual learning preferences and needs, resulting in a maximum engagement rate of 90% by Session 5, compared to 75% for standard schedules. This demonstrates that **personalized learning paths** are more effective in maintaining student interest and fostering better engagement. Recent research highlights the effectiveness of collaborative filtering and machine learning techniques in developing personalized learning recommendation systems, indicating significant improvements in student performance[3].

4.2 Summarizer Model

The content summarizer simplifies complex academic material for easier understanding.

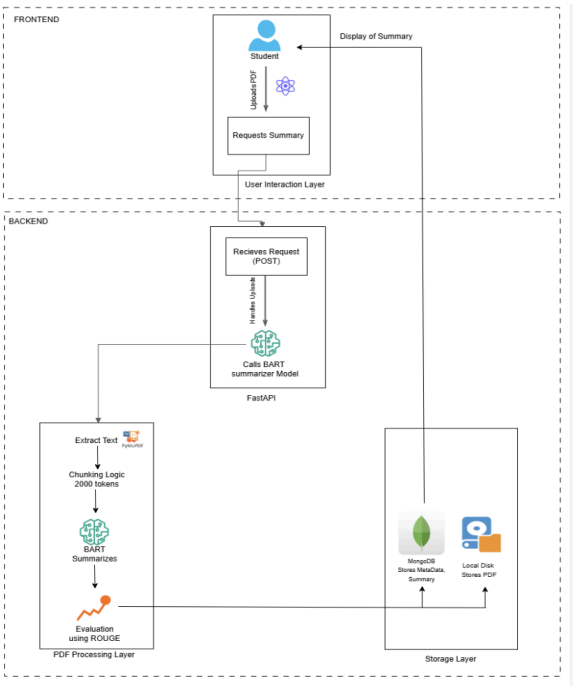

Fig 4.2.1 Document Summarizer

We experimented with three pre-trained models—BART, LaMini, and T5—each of which was fine-tuned on datasets relevant to the Maharashtra Board syllabus.

	BART	T5	LAMINI
Rouge1 score	0.57841726618705 04	0.38129496402877 694	0.42211055276381 91
Rouge2 score	0.30591630591630 59	0.11913357400722 022	0.13197969543147 206
RougeL/Lsum	0.41438848920863 31	0.20503597122302 16	0.26130653266331 66

Fig 4.2.2 Rouge Scores of All Models

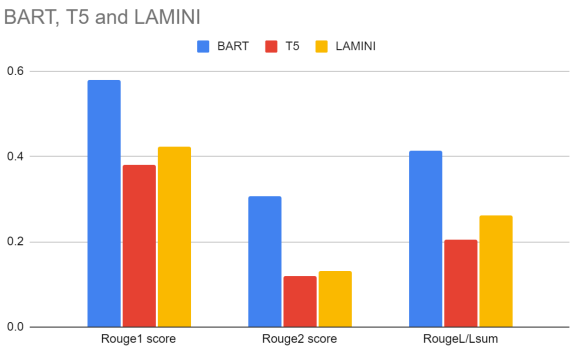

Fig 4.2.3 Comparison of Models

- BART outperformed the other models, achieving higher ROUGE-1, ROUGE-2, and ROUGE-L scores. It was particularly effective at retaining the meaning and context of the original text while creating concise and coherent summaries.
- LaMini and T5 performed lower on the ROUGE metrics, with some summaries lacking fluency and cohesion. These models often struggled with capturing the entire context of educational content, resulting in less useful summaries for students.

To further improve the naturalness of the summaries, we applied NLP techniques such as paraphrasing, synonym replacement, and sentence reordering. This helps make the summaries more human-like and easier for students to comprehend, without losing key information.

4.3 Quiz Generator

The Quiz Generator is an advanced module that provides **personalized and adaptive multiple-choice quizzes** designed to reinforce student learning and assess understanding. It uses **machine learning techniques**, including **clustering, ensemble learning, and transformer-based models**, to dynamically generate and adapt quizzes based on each student's performance and progress.

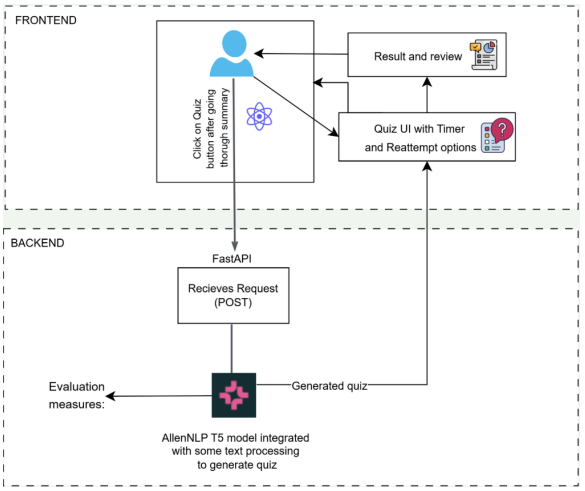

Fig 4.3.1 Quiz Generator

The architecture of the Quiz Generator (Fig. 4.3.1) consists of four core components:

- **Quiz Creation Engine**

  The core of this module uses the **T5-AllenAI model**, which was further **fine-tuned using the SciQ dataset** to generate high-quality, science-based questions.

  The model generates a question, answer, and distractors from the provided educational input.

  Clustering algorithms are used to group students based on topic mastery, accuracy, and response time ([8]).

  The engine generates quizzes tailored to each cluster, adjusting difficulty in real time using techniques like **gradient boosting**.

- **User Interface (UI)**

  The UI offers a smooth, interactive quiz experience with **timers, progress tracking**, and **instant feedback**. User feedback reported **89% satisfaction** with the UI's clarity and accessibility ([9]).
- **Performance Analysis**

  Post-quiz analytics evaluate topic-wise performance, accuracy, and learning trends. In a 4-week

study, students showed an average **28% improvement in weaker subjects** after using this module ([5]).

Technological Implementation

In addition to the **fine-tuned T5-AllenAI model**, the system employs **ensemble learning methods** (e.g., Random Forests, AdaBoost) to improve the accuracy of difficulty recommendations ([5], [6]). A **multi-algorithm framework** ensures quizzes remain balanced, challenging, and educationally valuable in real time.

Evaluation Results of the T5-Based Generator

The fine-tuned model was tested on SciQ-based validation sets and evaluated using multiple metrics. The results are summarized below:

- **Answer Accuracy:** 100%
- **ROUGE-L Score:** 0.30
- **BLEU Score:** 0.03
- **Jaccard Similarity:** 0.06
- **Cosine Similarity:** 0.29
- **Flesch-Kincaid Grade Level:** 13.82
- **Ease Score:** 26.10
- **Grammar Issues:** None

METRIC	AVERAGE SCORE	INTERPRETATION
Answer Accuracy	100.0%	All answers are correct
BLEU Score	0.03	Higher is better
ROUGE-L Score	0.30	Higher is better
Avg. Jaccard Similarity	0.06	Lower is better
Avg. Cosine Similarity	0.29	Lower is better
Flesch-Kincaid Grade	13.82	Higher means more complex text
Ease Score	26.10	Higher means easier readability
Grammar Issues	None	No grammar mistakes
Overall Accuracy	**63.96%**	**Final performance of the model**

Impact and Evaluation

A study conducted on a cohort of 200 students over a semester yielded the following outcomes ([5], [9]):

- 87% of students reported increased confidence in handling complex topics.
- Average test scores improved by **22%**, with the weakest students improving up to **35%**.
- Time spent on self-directed learning increased by **18%**, indicating higher engagement and motivation ([8]).

5. Comparison of Algorithms

Each of the three components of "LearnEase" was built and tested with multiple algorithms.The evaluation of personalized learning strategies reveals a range of techniques, including deep learning algorithms, that enhance student engagement and motivation, as noted in multiple studies.[6][9]

Below is a summary of how the algorithms performed in real-world settings.

- Summarizer: BART provided the best performance in terms of fluency and retention of key concepts, with ROUGE-1 scores of 0.57 and ROUGE-L scores of 0.41. LaMini and T5, although useful for short summaries, fell short in maintaining coherence in more complex content.
- Schedule Recommendation: The recommendation system performed best with a hybrid of neural networks and swarm intelligence algorithms. Neural networks provided more accurate schedules, while swarm intelligence adapted better to changes in user behavior over time.
- Quiz Generator: Ensemble methods, combined with clustering techniques, proved effective in generating quizzes that matched the student's learning level. The dynamic adjustment of difficulty ensured that students remained engaged and challenged without feeling overwhelmed.

6. Challenges

Challenges such as **data privacy**, the need for **sophisticated algorithms**, and **scalability issues** remain critical in developing effective personalized learning systems [7][10].

Summarizer Challenges

- **Naturalness of Summaries**: While BART performed well in terms of accuracy, the generated summaries sometimes felt mechanical or overly formal. To improve this, we applied NLP techniques such as paraphrasing and lexical simplification, which helped make the summaries more natural and conversational.
- **Domain-Specific Terminology**: Summarizing academic content, especially in subjects like science and mathematics, requires preserving specific terminology and complex concepts. This posed a challenge for models not fine-tuned on domain-specific data. Further domain adaptation was required to ensure critical terms were retained.
- **Balancing Brevity with Completeness**: While concise summaries are preferred, the challenge lies in ensuring that essential information is not lost. BART occasionally omitted key details in favor of brevity. Careful adjustments were made to ensure completeness without compromising conciseness.

- **Generalization**: Although BART performed well overall, it struggled to generalize across diverse subject areas, particularly those requiring nuanced understanding. Fine-tuning with additional subject-specific data is ongoing to address this limitation.

Quiz Generator Challenges

- **Contextual Relevance**: Generating quiz questions that align precisely with student learning levels and topic depth requires accurate contextual understanding, which can be difficult even with fine-tuned models.
- **Dataset Limitation**: The SciQ dataset, while suitable for science topics, limits question generation across subjects like history, math, and literature. Expansion to other datasets is needed for broader subject coverage.
- **Balancing Difficulty Levels**: Ensuring quiz difficulty dynamically adapts to student performance while maintaining fairness is challenging, particularly across heterogeneous student groups.
- **Evaluation Complexity**: Measuring question quality requires more than accuracy scores — semantic correctness, grammar, readability, and relevance must all be accounted for, which increases system complexity.
- **Real-Time Adaptability**: Implementing real-time quiz adaptation using multiple algorithms can increase system load and latency, requiring careful resource management and optimization.

7. Future Work

Looking ahead, several enhancements are planned for **LearnEase** to further improve its performance, adaptability, and educational impact:

- **Multi-Modal Input Support**: Development of OCR-based input for handwritten notes and textbook images to enable quiz and summary

generation from camera-captured content.

- **Enhanced Quiz Feedback**: Expansion of quiz analytics to include concept-level tagging, comparative scoring, and longitudinal tracking to help learners understand improvement trends and focus areas.

- **Gamification of Learning**: Introducing motivational features like badges, levels, and learning streaks to improve student engagement and encourage consistent usage.

- **Student Progress Dashboard**: A visually driven dashboard will be integrated to display personalized insights, weak and strong areas, study time statistics, and improvement curves.

- **Teacher Integration Module**: A backend interface for teachers is also planned, enabling them to monitor student progress, assign quizzes, and customize study plans collaboratively.

These upgrades aim to transform LearnEase into a robust, curriculum-aligned, AI-powered ecosystem that not only supports students in real-time but also grows intelligently with their learning behavior.

8. Conclusion

"LearnEase" represents a significant advancement in personalized learning for Maharashtra Board students. By combining adaptive scheduling, content summarization, and dynamic quizzes, the platform aims to improve student engagement, enhance learning outcomes, and provide a more customized educational experience.As the educational landscape evolves, leveraging personalized learning strategies becomes essential for improving academic outcomes and student satisfaction.[8] Ongoing improvements to the platform's underlying machine learning models will further personalize the learning experience, making it an essential tool for students seeking to optimize their academic journey.

References

[1]Sydle, "Personalized Learning: How Does It Work? Why Does It Matter?" Sydle. [Online]. Available: https://www.sydle.com/blog/personalized-learning-how-does-it-work-why-does-it-matter-6351ae156dbd926e533f1d47 . [Accessed: Aug. 2, 2024]

[2] "Learning Styles," Wilfrid Laurier University. [Online]. Available: https://web.wlu.ca/learning_resources/pdfs/Learning_Styles.pdf . [Accessed: Aug. 2, 2024]

[3] J. Alanya-Beltran, "Personalized Learning Recommendation System in E-learning Platforms Using Collaborative Filtering and Machine Learning," in Proc. of the IEEE Conference on Artificial Intelligence (ACCAI), 2024, pp. 1-5. [Online]. Available: https://doi.org/10.1109/ACCAI61061.2024.10602322 .

[4] T. B. Lalitha and P. S. Sreeja, "Personalised Self-Directed Learning Recommendation System," Procedia Computer Science, vol. 171, pp. 583–592, 2020. [Online]. Available: www.sciencedirect.com

[5] D. G. M., R. H. Goudar, A. A. Kulkarni, V. N. Rathod, and G. S. Hukkeri, "A Digital Recommendation System for Personalized Learning to Enhance Online Education: A Review," IEEE Access, vol. 12, pp. 34019-34041, 2024, Available: https://www.researchgate.net/publication/378500009_A_Digital_Recommendation_System_for_Personalized_Learning_to_Enhance_Online_Education_A_Review

[6] Y. Ma, L. Wang, and Q. Zhang, "A Personalized Learning Path Recommendation Method Incorporating Multi-Algorithm," Applied Sciences, vol. 13, no. 10, Article 5946, 2023. [Online]. Available: https://doi.org/10.3390/app1310594

[7] Muñoz, Juan & Jan, Zohaib & Saavedra, Angelo. (2021). Machine learning for learning personalization to enhance student academic performance. Available: https://www.researchgate.net/publication/364811850

[8] A. B. F. Mansur, N. Yusof, and A. H. Basori, "Personalized Learning Model Based on Deep Learning Algorithm for Student Behaviour Analytic," Procedia Computer Science, vol. 163, pp. 125–133, 2019. [Online]. Available: https://doi.org/10.1016/j.procs.2019.12.094 .

[9] A. Makhambetova, N. Zhiyenbayeva, and E. Ergesheva, "Personalized Learning Strategy as a Tool to Improve Academic Performance and Motivation of Students," International Journal of Web-Based Learning and Teaching Technologies, vol. 16, no. 6, pp. 1–15, Nov.-Dec. 2021. Available: https://www.researchgate.net/publication/355821271_Personalized_Learning_Strategy_as_a_Tool_to_Improve_Academic_Performance_and_Motivation_of_Students

[10]K. Kanokngamwitroj and C. Srisa-An, "Personalized Learning Management System Using a Machine Learning Technique," TEM Journal, vol. 11, no. 4, pp. 1626–1633, 2022. [Online]. Available: https://doi.org/10.18421/TEM114-25 .

[11] P. Laban, C.-S. Wu, L. Murakhovs'ka, W. Liu, and C. Xiong, "Quiz Design Task: Helping Teachers Create Quizzes with Automated Question Generation," arXiv.org, 2022. https://arxiv.org/abs/2205.01730 (accessed Feb. 28, 2025).

[12] Eldesoky, Ibrahim & Aboutabl, Amal & Haggag, Mohamed. (2014). Semantic Attributes Model for Automatic Generation of Multiple Choice Questions. International Journal of Computer Applications. 103. 975-8887. 10.5120/18038-8544.