
**PREPWISE**

ON

Submitted in partial fulfillment of the requirements
of the degree of

**Bachelor of Engineering**

**(Information Technology)**

By

**Mr. Aryan Patankar - Roll No (33)**

Under the guidance of

**Prof. Dipti Karani**


**Department of Information Technology**

**VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY, Chembur, Mumbai 400074**

**(An Autonomous Institute, Affiliated to University of Mumbai)**


# Vivekanand Education Society's
## Institute of Technology

(Autonomous Institute Affiliated to University of Mumbai, Approved by AICTE & Recognised by Govt. of Maharashtra)
*NAAC accredited with 'A' grade*

**April 2025**

## *Certificate*

This is to certify that project entitled

**"PrepWise"**

**Group Members Names**

Mr. Aryan Patankar - Roll No (33)

In fulfillment of degree of BE. (Sem.VI) in Information Technology for Project is approved.

**Prof. Dipti Karani**
**Project Mentor**

**External Examiner**

**Dr.(Mrs.)J.M.Nair**
**Principal**

**Dr.(Mrs.)Shalu Chopra**
**H.O.D**

Date:08 /04 /2025
Place: VESIT, Chembur

# *Declaration*

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I 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)**

Aryan Patankar - Roll No (33)

# Abstract

Abstracts contain most of the following kinds of information in brief form. The body of your paper will, of course, develop and explain these ideas much more fully. As you will see in the samples below, the proportion of your abstract that you devote to each kind of information—and the sequence of that information—will vary, depending on the nature and genre of the paper that you are summarizing in your abstract. And in some cases, some of this information is implied, rather than stated explicitly. The Publication Manual of the American Psychological Association, which is widely used in the social sciences, gives specific guidelines for what to include in the abstract for different kinds of papers—for empirical studies, literature reviews or meta-analyses, theoretical papers, methodological papers, and case studies.

**Keywords-***literature, theoretical, methodological, include, Publication*

# Contents

# List of Figures

# ACKNOWLEDGEMENT

The project report on ” GoGreen - A Comprehensive Sustainability Platform” is the outcome of the guidance, moral support and devotion bestowed on our group throughout our work. For this we acknowledge and express our profound sense of gratitude to everybody who has been the source of inspiration throughout project preparation. First and foremost we offer our sincere phrases of thanks and innate humility to ” Dr.(Mrs.)Shalu Chopra HOD”, ” Dr.(Mr.)Manoj Sabnis Deputy HOD”, ” Prof Bharati Raut” for providing the valuable inputs and the consistent guidance and support provided by them.We can say in words that we must at outset tender our intimacy for receipt of affectionate care to Vivekanand Education Society’s Institute of Technology for providing such a stimulating atmosphere and conducive work environment.

# CHAPTER: 1 INTRODUCTION

# Chapter 1

# Introductio

# n

## 1.1. Introduction

In today's fast-paced academic environment, personalized and adaptive learning solutions are critical to help students prepare effectively for exams. Traditional study methods often lack interactivity, customization, and scalability. This project, **PrepWise**, is an AI-powered exam preparation platform that leverages Google's Gemini LLM to generate topic-based and document-based multiple-choice questions (MCQs) in real-time. It empowers students to engage in focused and adaptive learning using either predefined topics or by uploading their own study materials (PDFs). Built using modern web technologies like React, Vite, and TypeScript, this tool offers an intuitive, fast, and responsive interface for learners.

## 1.2. Objectives

The main objectives of this project are:

1. **Automated MCQ Generation**: Generate high-quality MCQs dynamically using Gemini LLM based on selected topics or uploaded PDFs.

2. **Interactive Learning Experience**: Enable users to attempt questions one at a time with immediate feedback and the ability to progress sequentially.

3. **Custom Content Support**: Allow users to upload personal notes, textbooks, or any study material in PDF format to generate tailored questions.

4. **Frontend in Modern Stack**: Build the entire interface in React + Vite + TypeScript to ensure performance, scalability, and clean code structure.

5. **AI Integration**: Use Google Gemini for natural language understanding and question generation, ensuring relevance and difficulty matching

## 1.3. Motivation

Many students struggle to find efficient and engaging ways to prepare for exams. Static question banks often fail to adapt to individual learning needs. Furthermore, generating quality questions manually is time-consuming. With the emergence of powerful LLMs like Gemini, there is a tremendous opportunity to build intelligent systems that adapt to a student's context in real-time. This project is motivated by the vision to:

- Make quality exam preparation accessible and personalized.
- Empower students to make the most of their own study resources.
- Introduce AI into everyday academic practice in a useful, not gimmicky, way.
- Build a proof-of-concept that could be extended to support various subjects, languages, and learning levels.

## 1.4. Scope of the Work

The scope of this project includes:

- **Frontend Development**:
    - Build a responsive UI using React + Vite + TypeScript.
    - Components for topic selection, file upload, MCQ display, and navigation.
    - Dark/light theme compatibility and mobile responsiveness.
- **AI Integration**:
    - Connect with Gemini LLM API for topic-based and context-aware question generation.
    - Prompt engineering for generating valid MCQs and evaluating user responses.
- **PDF Processing**:
    - Enable PDF upload and extract readable text using backend tools (like PyMuPDF or PDF.js).
    - Chunk long documents into manageable inputs for Gemini.
- **MCQ Session Handling**:
    - Store user responses temporarily and show the next question on button click.

## 1.5. Feasibility Study

1. **Technical Feasibility**
   - The project was developed using Flask, React, TypeScript, and MongoDB—all open-source and well-supported technologies.

2. **Economic Feasibility**
   - As an academic project, cost was minimized by using freely available tools like GitHub, VS Code, and MongoDB Atlas.

3. **Operational Feasibility**
   - The application was successfully deployed and tested in a development environment. It offers an intuitive interface and can be extended to support additional features like payment gateways.

## 1.6. Organization of the report

- **Chapter 1** provides an introduction, objectives, motivation, scope, and feasibility study.
- **Chapter 2** covers the literature survey and background research.
- **Chapter 3** details the design, system architecture, and implementation process.
- **Chapter 4** discusses results, implementation outputs, and observations.
- **Chapter 5** concludes the project and outlines future enhancements.

# CHAPTER: 2: LITERATURE

# SURVEY

# Chapter 2

# Literature

# Survey

## 2.1. Introduction

The landscape of education is evolving rapidly with the integration of artificial intelligence. Traditional exam preparation methods such as textbooks, static question banks, and repetitive test series often fail to adapt to individual learning styles and paces. In contrast, modern learners require personalized, engaging, and dynamic platforms to prepare effectively.

**PrepWise** is a next-gen exam preparation tool that harnesses the power of **Google's Gemini LLM** to generate intelligent multiple-choice questions (MCQs) based on user-selected topics or uploaded study material (PDFs). The platform is built using modern web technologies like **React, Vite, and TypeScript**, ensuring a smooth, scalable, and responsive user experience.

This solution provides adaptive learning paths, real-time feedback, and the flexibility for users to test themselves using either standard topics or their own customized study content — enhancing learning outcomes and retention.

## 2.2. Problem Definition

Despite the growing availability of online learning platforms, students still face several challenges in personalized exam preparation:

1. **Static Content**: Predefined question banks lack adaptability to a student's individual learning context or materials.

2. **Time-Consuming Preparation**: Manually creating or sourcing practice questions from personal study materials is tedious and impractical.

3. **Lack of Personalization**: Existing solutions don’t allow integration of personal notes or PDFs for custom MCQ generation.

4. **Low Engagement**: Generic question sets do not adapt to the difficulty level or prior performance of the learner.

5. **Inaccessible AI-powered Tools**: Most AI-based educational tools are either overly complex or not tailored for real-time, topic-specific learning.

The solution focuses on developing an AI-powered web platform that dynamically generates multiple-choice questions from either pre-defined academic topics or user-uploaded PDF study materials, thereby enabling personalized, interactive, and efficient exam preparation.

## 2.3. Review of Literature Survey

1. **AI in Education**
   Recent studies have shown that artificial intelligence has the potential to revolutionize personalized learning. Tools like GPT and Gemini are capable of contextual understanding, which can be leveraged to generate domain-specific learning materials, including MCQs, summaries, and explanations [1].

2. **MCQ Generation Using NLP**
   Various research works have explored the use of NLP models to generate educational content. For example, "Automatic Question Generation using NLP Techniques" (Sharma et al., 2021) discusses how transformers and attention-based models can generate grammatically and contextually valid MCQs from input text [2].

3. **PDF Text Extraction for Educational Insights**
   Tools like PyMuPDF, Tika, and PDF.js have been widely used in research to extract educational content from PDF files. This allows students to use their lecture notes or reference books directly for automated content generation [3].

4. **LLMs for Contextual Understanding**
   Gemini, like other LLMs (e.g., GPT-4), is trained on diverse datasets, making it capable of interpreting and extracting insights from unstructured data such as user-uploaded PDFs. Studies have shown their effectiveness in summarization, question-answering, and semantic comprehension tasks [4].

5. **React + Vite in Educational Tools**
   Modern frontend frameworks like React and build tools like Vite have been adopted in scalable ed-tech applications for their fast load times, modular component structure, and TypeScript support, which enhances maintainability and type safety [5].

## References:


1. Baker, R. S. (2020). *Artificial Intelligence in Education: Promises and Implications for Teaching and Learning*.

2. Sharma, A., Gupta, V., & Jain, S. (2021). *Automatic Question Generation using NLP Techniques*. IJCA.

3. Python Software Foundation. (2022). *PyMuPDF Documentation*.

4. OpenAI. (2023). *LLMs and Multimodal Models for Education*.

5. GitHub Repositories: *React in EdTech Projects*, *Vite + TS Starter Templates*.

# CHAPTER: 3 DESIGN AND

# IMPLEMENTATION

# Chapter 3

# Design and Implementation

## 3.1. Introduction

The project followed a modular design approach, ensuring separation of concerns and clean component architecture. Agile principles were used with weekly sprints.

## 3.2. Requirement Gathering

Key functional requirements included:

- User authentication
- User can select a predefined topic from a list (e.g., DBMS, OOP, CN, AI, etc.)
- User can upload a PDF document (notes, textbook, etc.)
- The system generates MCQs dynamically based on the selected topic or uploaded content
- Users can view one question at a time with options

Tools and technologies used:

- Frontend: **React, Vite, TypeScript**
- Backend: **Flask**(for handling file uploads and Gemini API calls)
- API Integration: **Gemini API** (or Google Vertex AI if hosted)
- File handling: **PDF** parsing using libraries like `pdf-parse` (Node) or `PyMuPDF` / `pdfminer.six` (Python)

## 3.3. Proposed Design

The platform consists of the following pages and features:

### 1. Home Page

- User selects a subject/topic (e.g., DBMS, OOP, AI) from a dropdown.

- Optionally, the user can upload a PDF document (lecture notes, syllabus, or textbook).
- A start button initiates the MCQ generation flow.

### 2. MCQ Interaction Page

- Displays one question at a time with four options.
- The user selects an answer and clicks **Next**.
- Immediate feedback (optional) is shown (e.g., correct/incorrect).
- Tracks user accuracy and progress.

### 3. PDF Upload & Insight Page

- After uploading a PDF, insights are shown:
    - Summary of PDF content
    - Important topics and concepts
    - Suggested MCQs for practice
- Option to generate questions from the PDF.

## 3.4. Proposed Algorithm

Step 1: Start

Step 2: User lands on the Home Page

Step 3: User logs in using email

Step 4: User is redirected to the dashboard (Post-login Home Page)

Step 5: User chooses one of the following:
A) Search for any topic
B) Uploads a PDF document

Step 6: If predefined topic selected:
↳ Send prompt to Gemini: "Generate 1 MCQ with 4 options on <topic>"
↳ Receive MCQ and display it to the user

Step 7: If PDF is uploaded:
↳ Extract text from PDF
↳ Send prompt to Gemini: "Generate 1 MCQ with 4 options based on this content:\n<text>"

↳ Receive MCQ and display it

Step 8: User selects an answer
↳ Store response
↳ Show feedback (optional)
↳ Allow navigation to the next question

Step 9: Track and display accuracy statistics in the session

Step 10: On session completion:
↳ Show summary of performance
↳ Allow retry or new topic upload

Step 11: User logs out

Step 12: End

## 3.5. Architectural Diagrams

### 3.5.1. UML Diagram

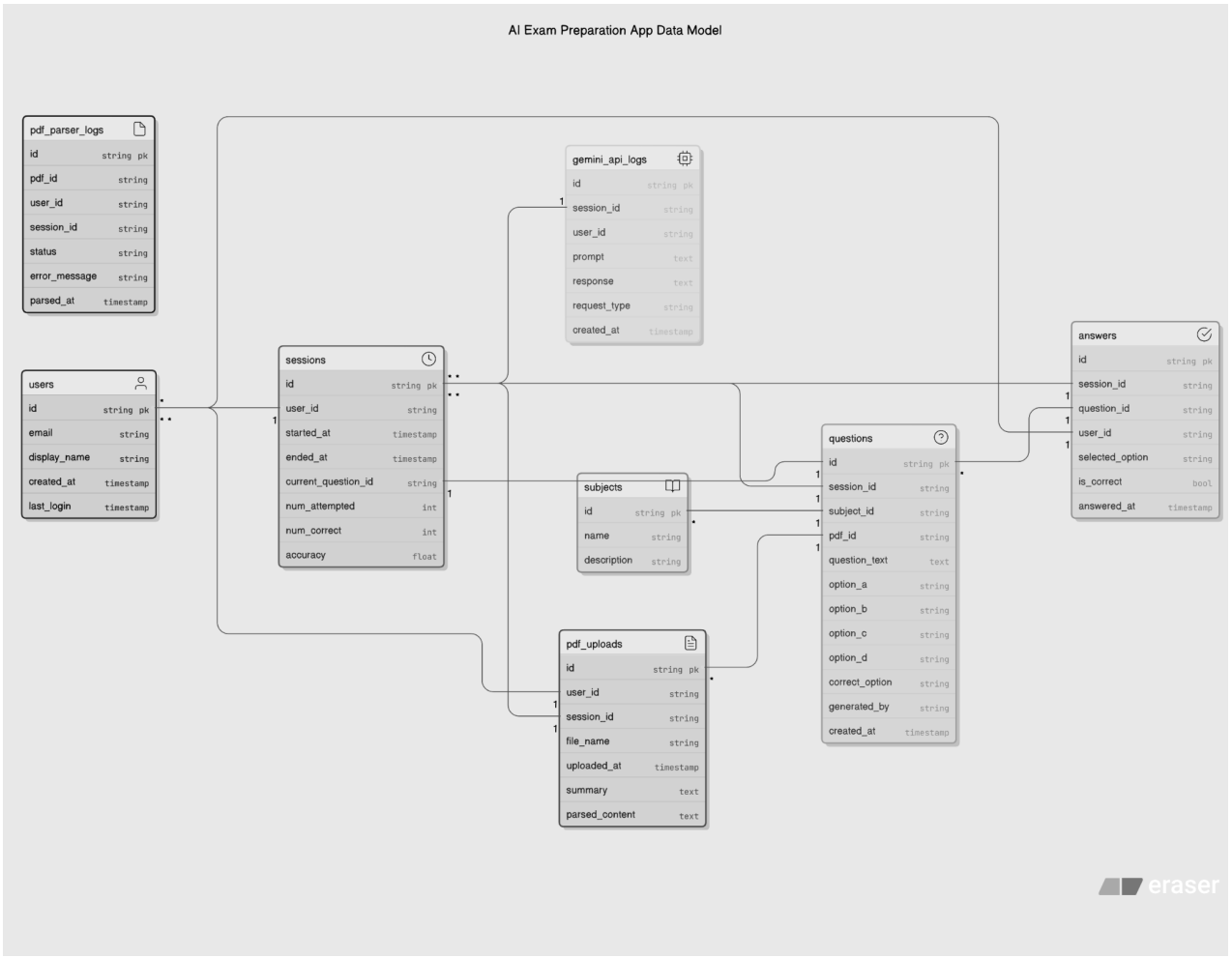



Figure 3.1: UML Diagrams

### 3.5.2. Data Flow Diagram

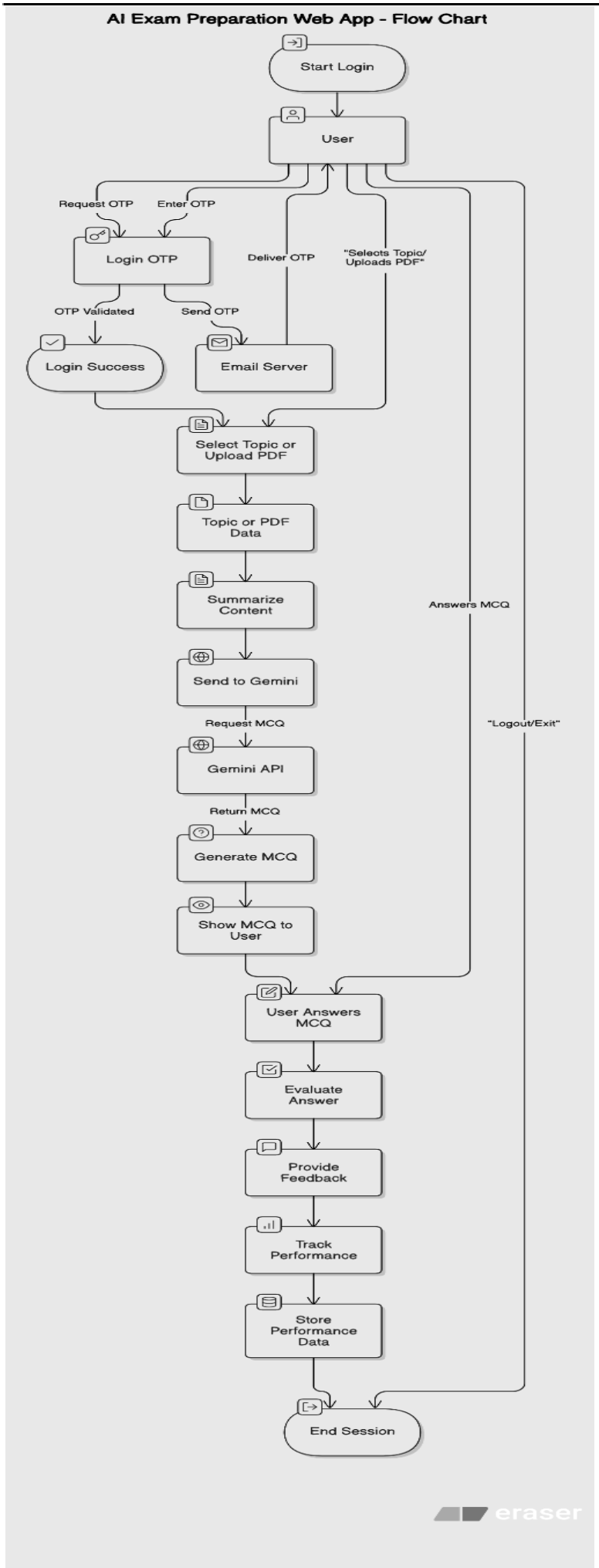

AI Exam Preparation Web App - Flow Chart
Start Login
User
Request OTP
Enter OTP
Login OTP
Deliver OTP
"Selects Topic/ Uploads PDF"
OTP Validated
Send OTP
Login Success
Email Server
Select Topic or Upload PDF
Topic or PDF Data
Summarize Content
Answers MCQ
Send to Gemini
Request MCQ
"Logout/Exit"
Gemini API
Return MCQ
Generate MCQ
Show MCQ to User
User Answers MCQ
Evaluate Answer
Provide Feedback
Track Performance
Store Performance Data
End Session
eraser

Figure 3.2: Data Flow Diagram

## 3.6. Hardware Requirements

- **Device Used:** Laptop
- **Processor:** Intel Core i5 (Quad-Core)
- **RAM:** 8 GB
- **Usage:** Suitable for initial development and testing

## 3.7. Software Requirements

- **Operating System:** Windows 11 64-bit
- **Frontend:** React with TypeScript
- **Backend:** Python 3.11+ with Flask
- **Package Manager:** Node.js v18.16.1 (with npm)
- **Database:** MongoDB Atlas (Cloud-based NoSQL)
- **Code Editor:** Visual Studio Code (VS Code)
- **Version Control:** Git & GitHub for collaboration and code management

## 3.8. Code

**GITHUB LINK**

-https://github.com/AryanPatankar27/PrepWise-WebX-CA_Updated

# CHAPTER: 4 RESULTS AND

# DISCUSSION

# Chapter 4

# Results and Discussion

## 4.1. Introduction

This chapter documents the major outputs and screens of the DMart Clone project.

## 4.2. Results of Implementation

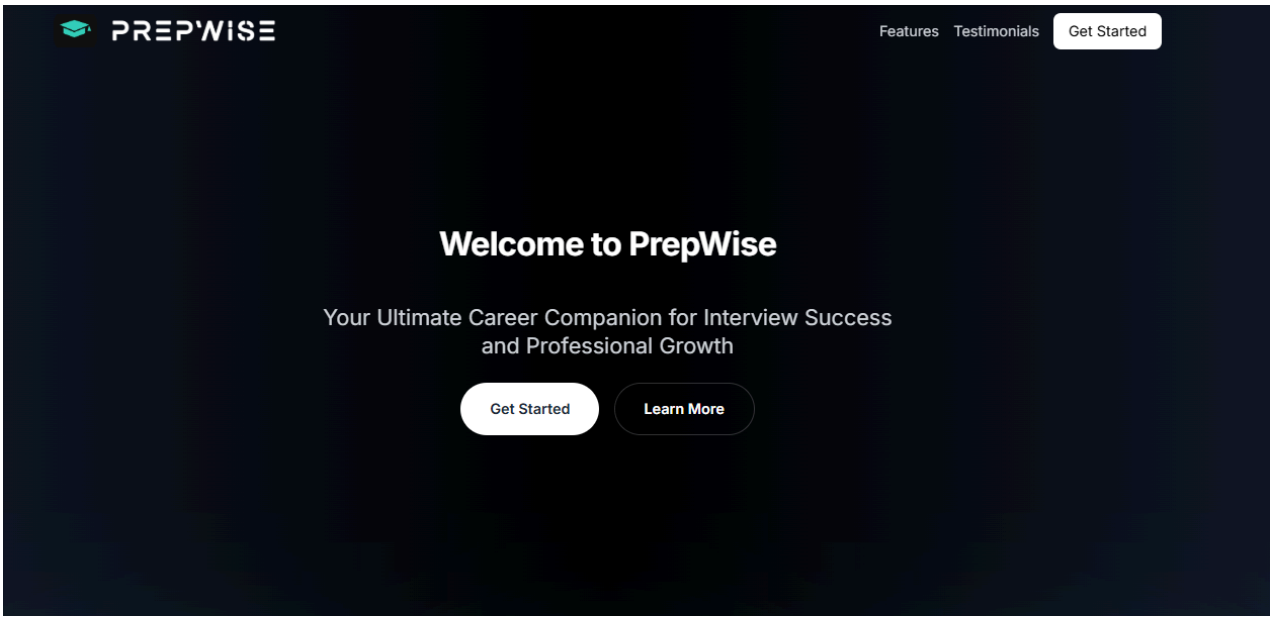



Figure 4.1: Home Page

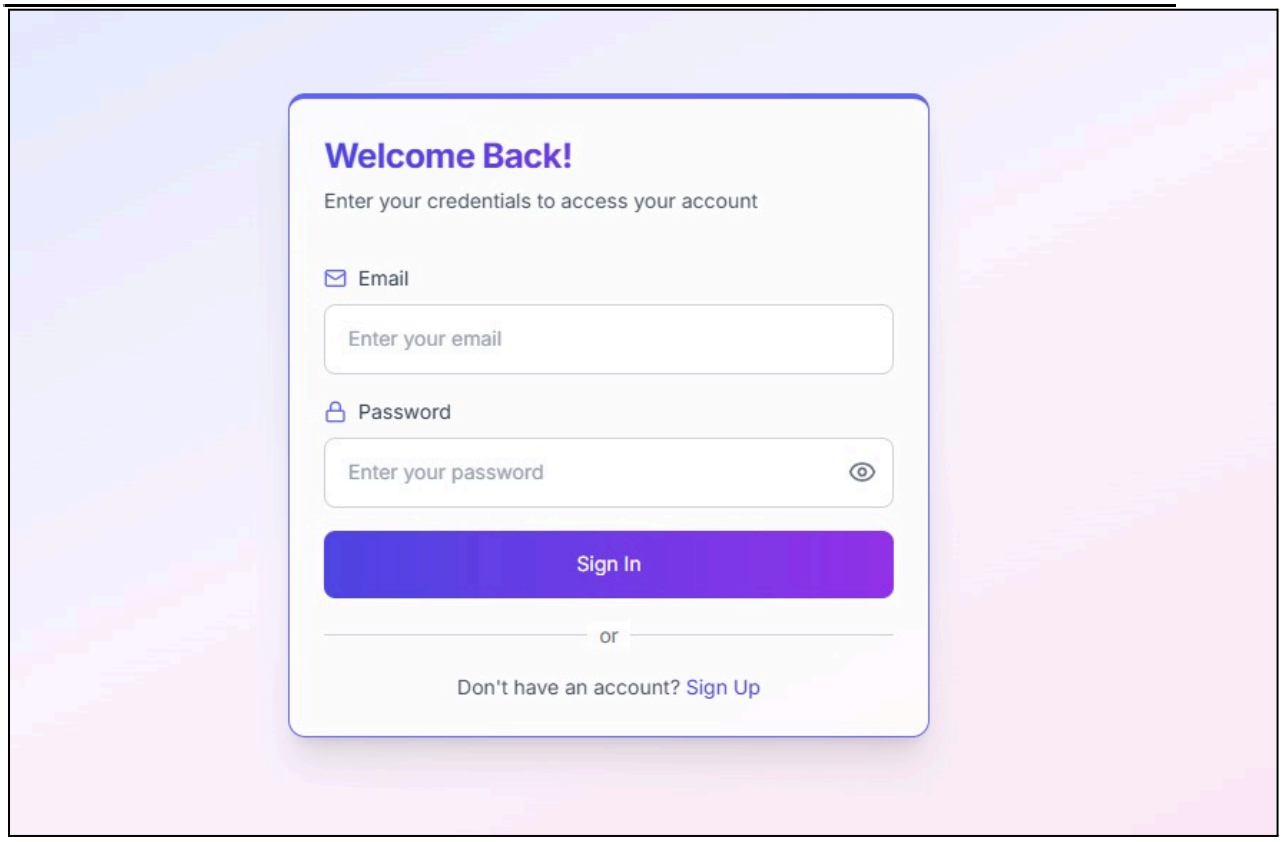



Figure 4.2: Login Page

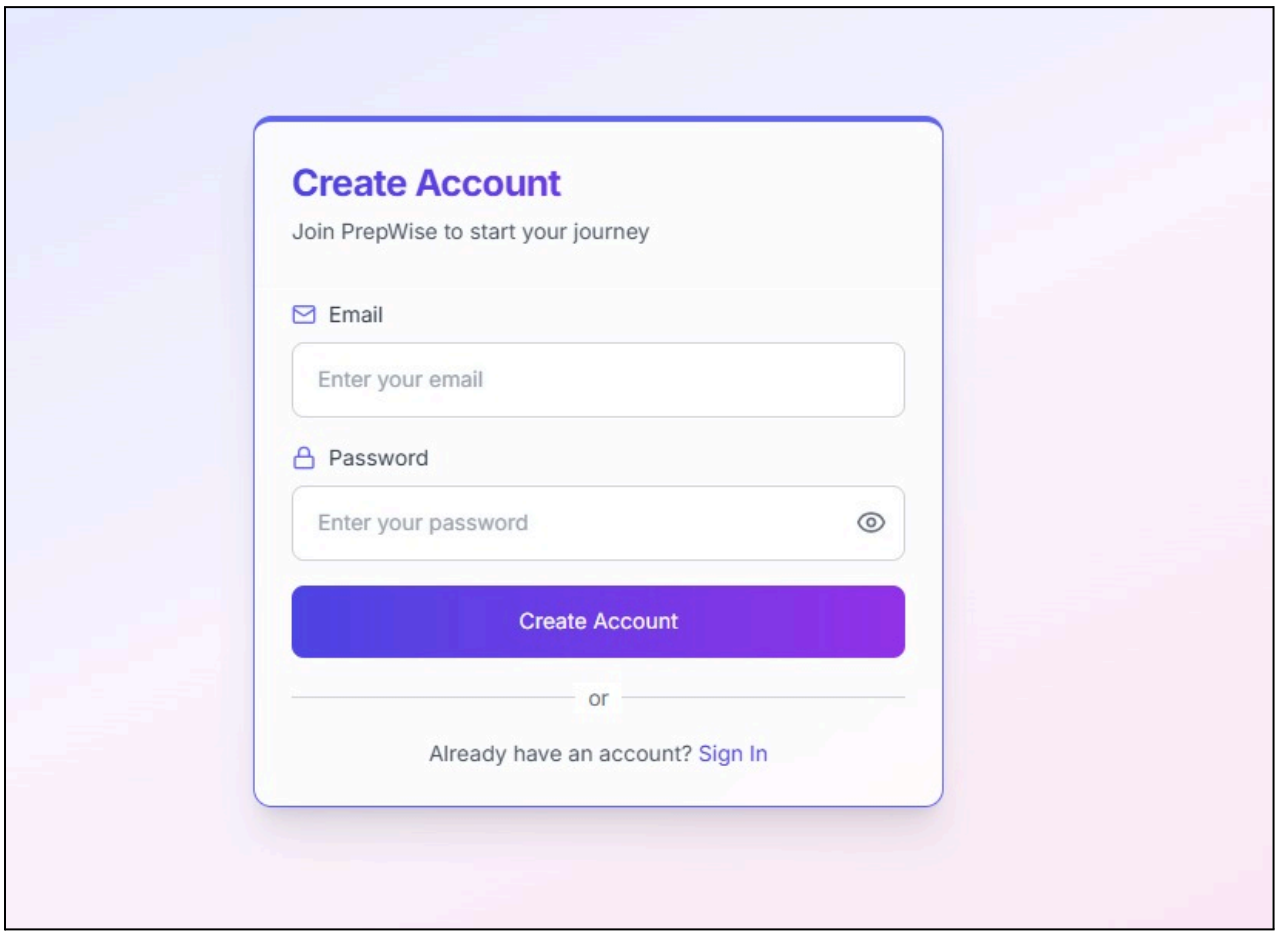



Figure 4.3: Sign Up Page

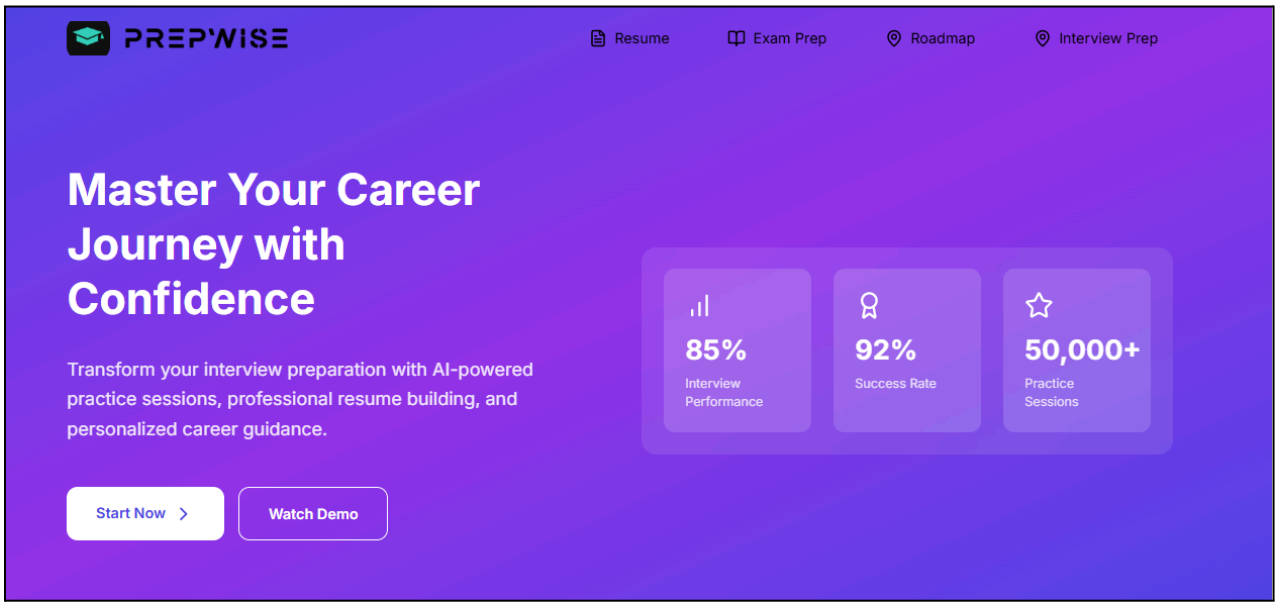



Figure 4.4: Dashboard

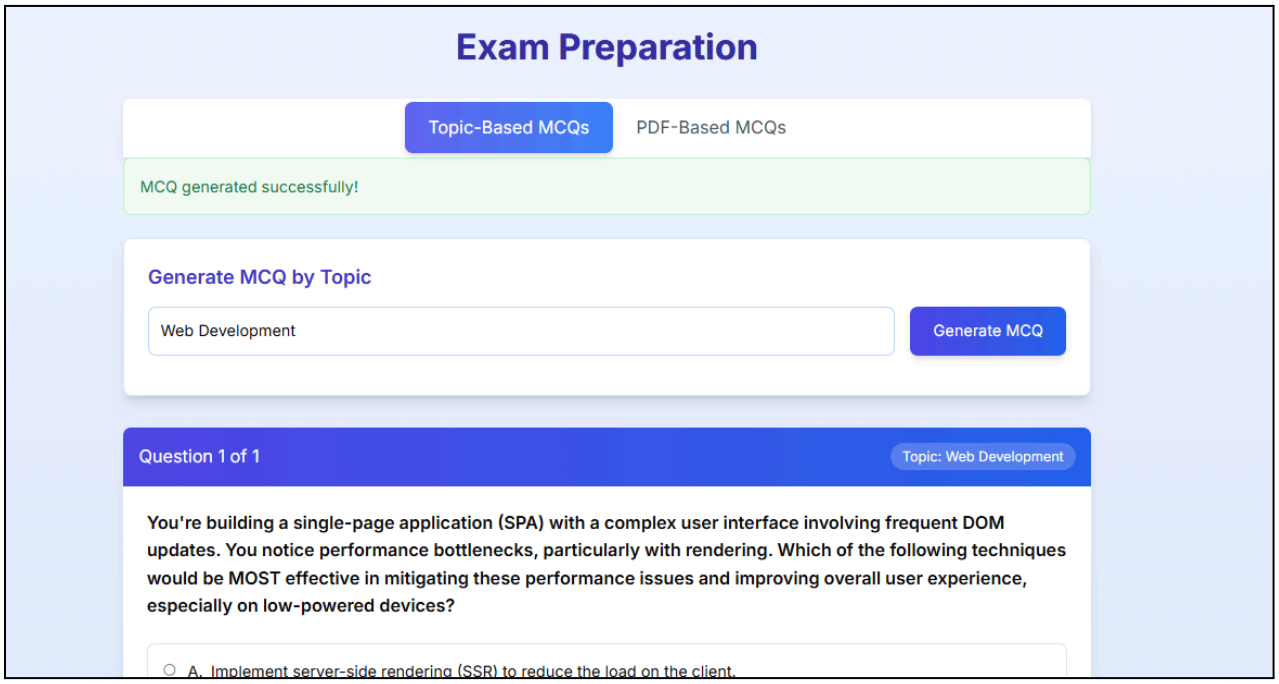
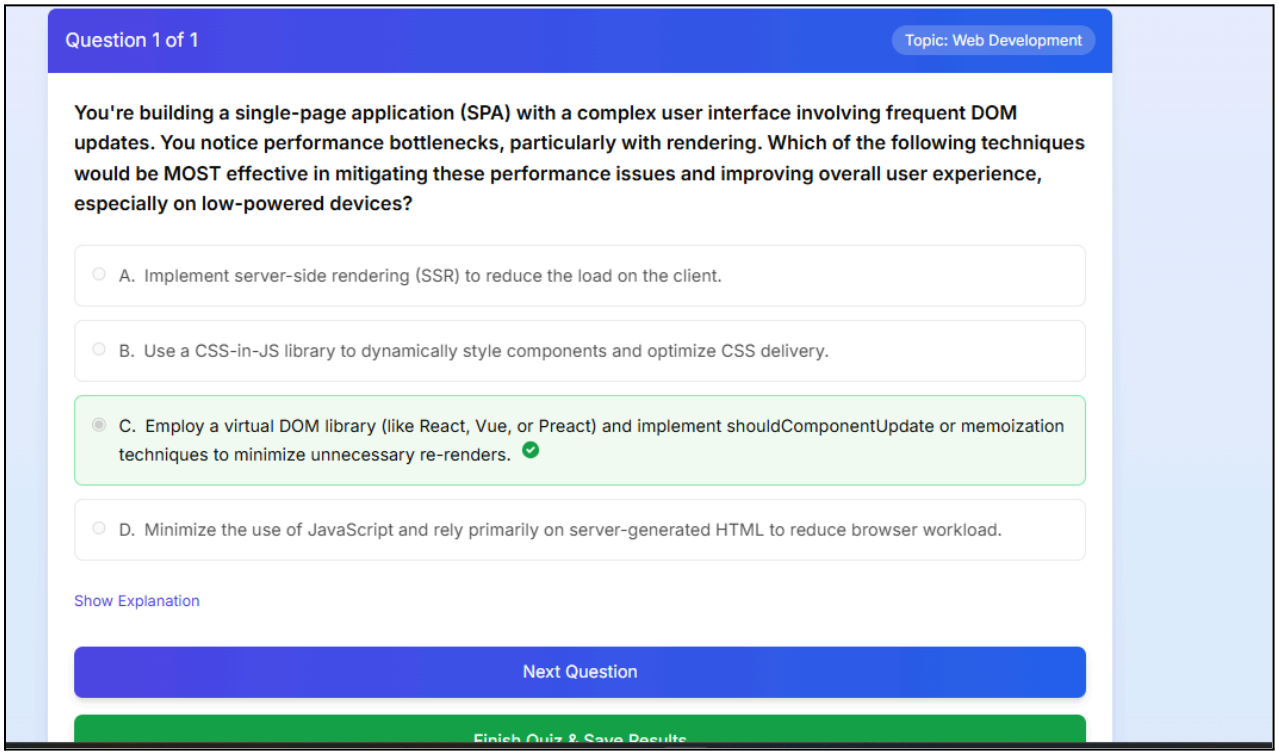



Figure 4.5:Topic Wise MCQS



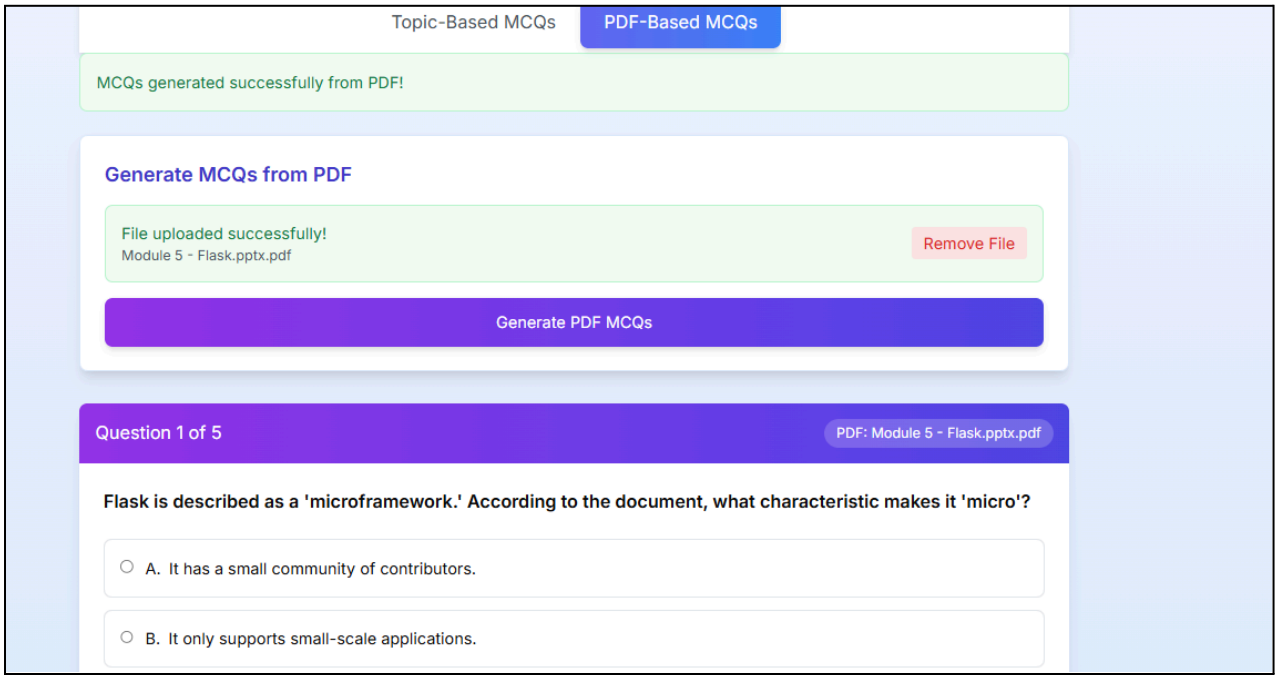




Figure 4.6:PDF Based MCQs

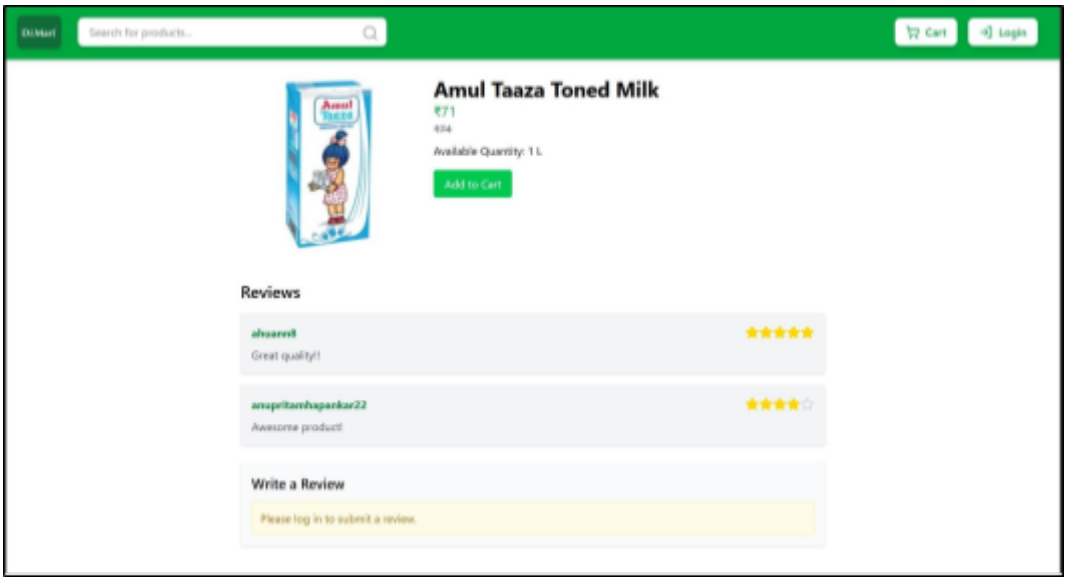



Figure 4.6: Product Detail Page

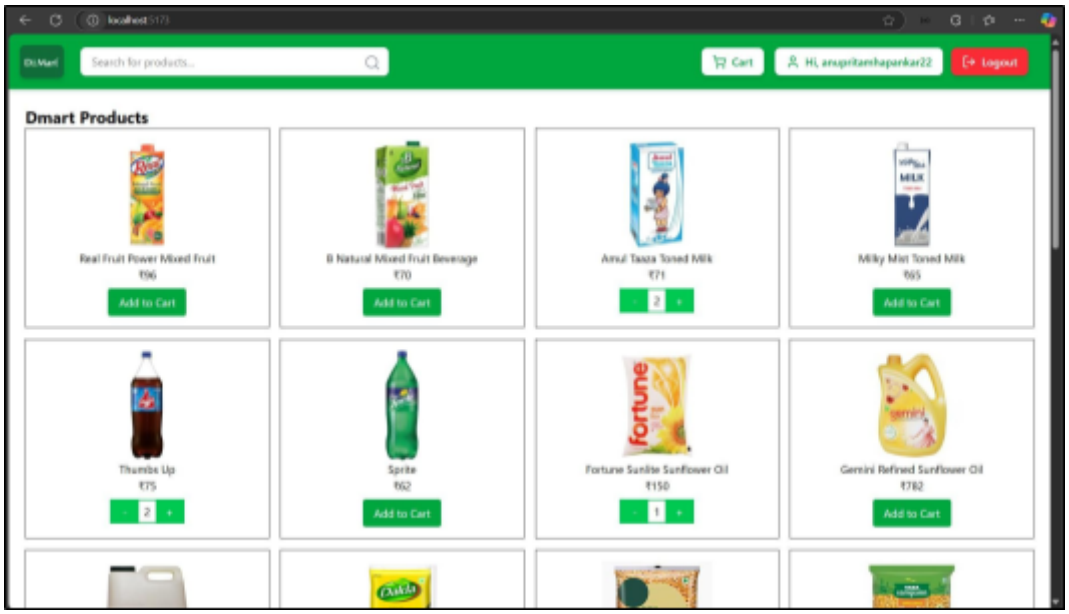



Figure 4.7: Home Page after Login

## 4.3. Observation/Remarks

During the development and preliminary testing of **PrepWise**, several critical observations were made:

1. **Effective Question Generation**: The Gemini LLM was able to generate high-quality, context-aware multiple-choice questions for a wide variety of topics with minimal latency.

2. **PDF-based Learning Enhancement**: Users found value in uploading custom study material (PDFs) and getting instant MCQs, enabling focused and personalized practice sessions.

3. **User Engagement**: The interactive format—allowing users to answer one question at a time and navigate with a "Next Question" button—promoted higher user engagement and better learning retention.

4. **Ease of Use**: The frontend developed using React + Vite + TypeScript offered a smooth, responsive, and intuitive user interface, even for non-tech-savvy users.

5. **Accuracy & Relevance**: While most questions generated were relevant, occasional inaccuracies or ambiguities in options highlighted the need for post-generation validation.

6. **Scalability**: The modular architecture allows seamless integration of additional features like scoring, analytics, user authentication, and gamification in the future.

# CHAPTER: 5 CONCLUSION

# Chapter 5

# Conclusio

# n

## 1.1. Conclusion

**PrepWise** successfully demonstrates how Generative AI, combined with modern web technologies, can revolutionize exam preparation. By allowing users to either select a topic or upload custom study materials, the platform generates contextually accurate and personalized MCQs in real-time.

This tool reduces the dependency on static study materials and gives users control over what they want to study and how they want to be tested. With the backing of Gemini's powerful language model, PrepWise bridges the gap between AI and education by providing adaptive and interactive learning experiences.

It proves to be especially useful for:

- Competitive exam aspirants
- University students
- Self-learners and educators looking for dynamic assessment tools

## 1.2. Future Scope

- **Scoring and Analytics**
  Introduce real-time score calculation, performance tracking, difficulty adjustment based on user responses, and detailed analytics reports for each topic.

- **Multilingual Support**
  Expand support for regional and international languages to make exam preparation inclusive and accessible globally.

- **Gamification**
  Add leaderboards, badges, streaks, and quizzes to enhance engagement and encourage consistent study habits.

- **Voice Interaction**
  Integrate speech-to-text capabilities to allow users to verbally request MCQs or answers, aiding accessibility.

- **Subjective Question Generation**
  Include short answers, long-form questions, and explainers, not just MCQs.

- **Integration with LMS and Google Classroom**
  Allow teachers to generate assessments and track students' progress in a structured environment.

- **Offline Support / PWA**
  Convert the web app into a Progressive Web App (PWA) so users can practice even without internet access.

# Bibliography


1. **Brown, T. et al.** (2020). *Language Models are Few-Shot Learners*. In *Advances in Neural Information Processing Systems*, 33. https://arxiv.org/abs/2005.14165

2. **Google AI**. (2023). *Introducing Gemini: Google DeepMind's Most Capable AI Model*. Retrieved from: https://deepmind.google/technologies/gemini

3. **Vaswani, A. et al.** (2017). *Attention is All You Need*. In *NeurIPS*. https://arxiv.org/abs/1706.03762

4. **OpenAI**. (2023). *OpenAI API Documentation*. Retrieved from: https://platform.openai.com/docs

5. **React Official Docs**. (2024). *React – A JavaScript Library for Building User Interfaces*. Retrieved from: https://react.dev

6. **TypeScript Documentation**. (2024). *TypeScript: JavaScript with Syntax for Types.* Retrieved from: https://www.typescriptlang.org/docs

7. **Vite.js**. (2024). *Vite – Next Generation Frontend Tooling*. Retrieved from: https://vitejs.dev

8. **Streamlit Docs**. (2023). *Streamlit — The fastest way to build and share data apps*. Retrieved from: https://docs.streamlit.io

9. **Scardamalia, M. & Bereiter, C.** (2006). *Knowledge Building: Theory, Pedagogy, and Technology*. In *The Cambridge Handbook of the Learning Sciences*.

10. **Koedinger, K.R. et al.** (2013). *Data Mining and Education*. *Wiley Interdisciplinary Reviews: Cognitive Science*, 4(1), 49–59.

11. **Pappano, L.** (2012). *The Year of the MOOC*. *The New York Times*. Retrieved from: https://www.nytimes.com

12. **Haleem, A. et al.** (2022). *Role of Artificial Intelligence in Higher Education: A Systematic Review*. *Materials Today: Proceedings*.


---

If you are submitting this as an academic report, you can format it in IEEE or APA style depending on your university/institution's guideline. Let me know if you'd like me to convert this list into either of those formats.