RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

PREPMASTER: AI-POWERED PERSONALIZED ACADEMIC ASSISTANT

Submitted by
SARUMATHY J (221501126)
SIVARANJANI K (221501139)

AI19643 FOUNDATIONS OF NATURAL LANGUAGE PROCESSING

Department of Artificial Intelligence and Machine Learning

Rajalakshmi Engineering College, Thandalam

RAJALAKSHMI
ENGINEERING COLLEGE

BONAFIDE CERTIFICATE

NAME …………………………………………………………………………………..……..…

ACADEMIC YEAR………………..………SEMESTER…………BRANCH………………

UNIVERSITY REGISTER No.

Certified that this is the Bonafide record of work done by the above students in the Mini Project titled **"PREPMASTER: AI-POWERED PERSONALIZED ACADEMIC ASSISTANT"** in the subject **AI19643 FOUNDATIONS OF NATURAL LANGUAGE PROCESSING** during the year **2024 - 2025.**

Signature of Faculty – in – Charge

Submitted for the Practical Examination held on ______________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

In today’s fast-paced academic environment, students face increasing pressure to excel while managing time and vast syllabi efficiently. Traditional learning methods often fall short in offering the personalized guidance and adaptive tools needed for effective academic preparation. PrepMaster is an AI-driven learning platform developed to address these challenges by providing individualized academic support, intelligent study planning, and adaptive assessment tools in one integrated solution. It comprises three core modules that enhance learning outcomes. The AI Tutor, powered by the LLaMA model, serves as a real-time conversational assistant, enabling students to ask questions and receive human-like explanations instantly. This fosters a deeper understanding of academic content through interactive dialogue and achieved a ROUGE-L F1 score of 0.59, indicating strong fluency and relevance. The Exam Strategy Planner, built using the BART model, generates personalized study plans based on syllabus, available time, and target exam dates, helping students reduce last-minute stress through strategic preparation. The AI Quiz Generator, also based on BART, creates dynamic, topic-specific multiple-choice questions that adapt to the learner’s progress, supporting continuous self-assessment and targeted revision. These BART-powered modules achieved ROUGE-1 and ROUGE-2 F1 scores of 0.66 and 0.40 respectively, reflecting high-quality content generation. PrepMaster’s frontend is developed using HTML, CSS, and JavaScript, delivering a responsive, user-friendly interface that enhances accessibility. By combining conversational AI, strategic planning, and intelligent content generation, PrepMaster fosters consistent study habits, improves comprehension, and boosts exam confidence—empowering students to reach their academic goals more efficiently.

Keywords: AI Tutor, Exam Strategy Planner, Quiz Generator, Llama Model, BART, ROUGE

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

Preparing for examinations has always been a challenging task for students, requiring consistent effort, proper planning, and effective practice. However, traditional methods often fail to adapt to the unique needs of individual learners. With the growth of artificial intelligence technologies, it is now possible to create personalized learning experiences that actively assist students in every stage of their preparation. PrepMaster is designed with this objective — to provide a smarter and more efficient way for students to study, plan, and assess their knowledge. PrepMaster is divided into three key modules: the AI Tutor, Exam Strategy Planner, and AI Quiz Generator. The AI Tutor leverages the Llama model to deliver an interactive learning experience where students can ask questions freely and receive detailed, easy-to-understand responses. The tutor adapts to the student's queries and learning pace, fostering a personalized educational environment that traditional methods often lack. The Exam Strategy Planner, powered by the BART model, focuses on guiding students in planning their studies. Students often struggle with managing large syllabi within limited timeframes. PrepMaster addresses this by generating customized study schedules based on the syllabus topics and available preparation time. This helps students systematically cover all subjects without feeling overwhelmed, promoting consistent and confident preparation. The AI Quiz Generator also utilizes the BART model to create dynamic quizzes based on the study material provided by the user. By generating relevant multiple-choice questions, it allows students to self-assess their understanding and strengthens knowledge retention. Regular quiz practice helps identify weak areas and improves overall exam readiness. The frontend design of PrepMaster, developed using HTML, CSS, and JavaScript, ensures smooth navigation, visually appealing layouts, and real-time interactions. With its intuitive interface, students can easily access all three modules without technical complexity.

PrepMaster aims to redefine exam preparation by integrating intelligent tutoring, strategic study planning, and automated assessments into one cohesive platform. It supports active learning, improves time management, and empowers students to take charge of their educational journey more effectively.

CHAPTER 2
LITERATURE REVIEW

[1] Title: NLP-Based Sentiment Analysis for Evaluating Student Feedback in English Language Education

Authors: Mayormente, M. D., & Gumpal, B. R. (2025)

The research demonstrates the effectiveness of using LSTM networks for automated sentiment analysis of student feedback, achieving an accuracy of 98.5%, precision of 96%, recall of 90%, and an F1-score of 91%. It identified positive sentiments for teaching faculty (60%) and library facilities (75%), while course content received only 45% positive feedback. However, the study's limitation lies in its reliance on sentiment classification, which may overlook nuanced feedback and deeper context that could further enrich educators' understanding of student needs.

[2] Title: A Generative Artificial Intelligence Based Tutor for Personalized Learning

Author: L. Bonde

This study develops a Generative AI tutor for personalized learning in higher education by implementing a Custom GPT model. Testing on an Applied Cryptography course showed promising results in adapting content to individual learner needs. Limitations of this model are only the content generation module is completed; full learner profiling, LMS integration, and ethical assessments remain as a future work.

[3] Title: Intelligent Deep-Learning Tutoring System to Assist Instructors in Programming Courses

Authors: Roldán-Álvarez, D., & Mesa, F. J (2024)

The author presents an AI-based intelligent tutor designed to support programming students by providing answers, examples, and personalized feedback during the coding process. It helps reduce the need for constant instructor supervision while improving independent learning. Early testers found it effective in enhancing programming skills and clarifying doubts in real time. The tool shows potential for use in other courses beyond programming. However, it currently struggles with handling complex code queries and adapting to varied learning styles. Further development is needed to improve its flexibility and accuracy.

[4] Title: Enhancing Academic Performance with Generative AI-Based Quiz Platform

Authors: Chang, C.-K., & Chien, L.-C.-T (2024)

The study demonstrates that the QuizGPT platform, using generative AI to create adaptive quizzes, enhances Python learning and engagement, with a positive correlation between platform activities and test scores. It offers personalized practice and supports self-paced learning, contributing to improved concept retention. The study's limitations include its focus on a single subject (Python) and the lack of exploration into the scalability and long-term effectiveness of AI-generated quizzes in broader educational contexts. Future research should investigate its applicability across diverse subjects and learner profiles to assess its generalizability.

[5] Title: Elevating Education through AI Tutor: Utilizing GPT-4 for Personalized Learning

Authors: Shahri, H., Emad, M., Ibrahim, N., Rais, R. N. B., & Al-Fayoumi, Y (2024)

The AI Tutor utilizes GPT-4 and Invideo AI to provide personalized learning through dynamic content such as videos, quizzes, and interactive lessons, fostering deep concept comprehension, creativity, and critical thinking. It adapts to individual learning styles, offering a tailored educational experience. However, the work does not address the scalability of this solution across various subjects or educational systems, nor the challenges in implementing it on a larger scale in diverse environments. Further exploration is needed to assess the feasibility and effectiveness of such a platform in broader educational settings.

[6] Title: AI-Tutoring in Software Engineering Education: Experiences with Large Language Models in Programming Assessments

Authors: Frankford, E., Sauerwein, C., Bassner, P., Krusche, S., & Breu, R (2024)

In this paper, it explores the integration of the GPT-3.5-Turbo model as an AI-Tutor within the APAS Artemis, identifying various user types and highlighting benefits like timely feedback and scalability. The AI-Tutor enhances personalized learning by providing immediate assistance and adapting to individual needs. However, challenges such as generic responses and concerns about inhibited learning progress when using the AI-Tutor were also noted. Further refinement is required to improve response specificity and ensure that the AI-Tutor complements rather than replaces traditional learning methods.

[7] Title: AI-Based Smart Education System to Enhanced the Learning of Students

Authors: Barde, A., Thakur, R., Patel, S., Sinah, N., & Barde, S (2024)

This paper examines the transformative role of AI in education, focusing on AI-driven tools like adaptive assessments, intelligent tutoring, and personalized learning systems. It highlights how AI chatbots and virtual assistants can offer tailored support, enhancing student engagement and enabling dynamic, gamified learning environments. Additionally, the paper discusses the potential for AI to bridge learning gaps by providing real-time feedback and fostering individualized learning paths for students at various skill levels.

[8] Title: AI-Driven Educational Transformation in Secondary Schools: Leveraging Data Insights for Inclusive Learning Environments

Authors: Duraes, D., Bezerra, R., & Novais, P (2024)

This work explores the potential of AI in secondary education, focusing on the use of student assessment data and socioeconomic context to create personalized learning pathways and targeted interventions. It aims to enhance academic outcomes, promote inclusivity, and bridge educational gaps by leveraging AI for adaptive support and equitable learning. It addresses the ethical challenges, including equity, transparency, and data privacy, that arise with AI deployment in educational settings. However, the limitations include the difficulty of generalizing the approach across diverse educational systems and the potential for biases in AI algorithms, which may affect the fairness and effectiveness of interventions. Further research is needed to refine AI models and ensure they are ethically aligned with diverse student needs.

[9] Title: Artificial Intelligence in Education and Learning (AI in Research)

Authors: Besimi, N., Besimi, A., & Çiço, B (2022)

This paper explores the growing role of AI in education, focusing on applications like AI tutoring, digital lessons, and research improvement models. It emphasizes AI's potential to enhance educational processes but highlights challenges related to resources, infrastructure, and readiness for AI integration. Limitations include the lack of detailed solutions for overcoming these challenges and the need for better infrastructure and educator training.

[10] Title: Generation of Quizzes and Solutions Based on Ontologies -- A Case for a Music Problem Generator

Authors: Mavalankar, A., Kelkar, T., & Choppella, V (2015)

This system effectively generates diverse and complex quizzes by utilizing ontologies, propositional logic, and similarity finding, making it adaptable across various domains like music, animals, and more. By analyzing concepts, relationships, and attributes, the system ensures that quizzes are both challenging and relevant. Additionally, manual effort in maintaining these knowledge bases can be time-consuming, and the system's accuracy depends on the quality of the underlying data. The limitations include the potential for incomplete or outdated knowledge bases, which could affect the quality of generated quizzes, and the challenge of scaling the system to support more domains without requiring substantial resources for data management and continuous updates.

CHAPTER 3
SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

- CPU: Intel Core i5 or better
- GPU: Integrated Graphics
- Storage: At least 500 GB SSD
- RAM - 16GB

3.2 SOFTWARE REQUIRED:

- Visual Studio Code (Version 1.79.0)
- HTML
- CSS
- Javascript
- Scikit-learn (Version1.3.2)
- PyTorch (Version 2.0.0)
- Transformers (Version 4.30.0)
- KeyBERT (Version 0.7.0)

CHAPTER 4
SYSTEM OVERVIEW

4.1 EXISTING SYSTEM

The reviewed systems demonstrate the transformative potential of AI in education, focusing on personalized learning, adaptive assessments, and real-time feedback. AI-driven tools like GPT-4 and GPT-3.5-Turbo generate dynamic content, quizzes, and videos tailored to individual learning needs. AI-based quiz platforms enhance engagement and performance in subjects like Python and programming, while sentiment analysis using LSTM models evaluates student feedback to identify areas for improvement. Despite their promising results, challenges such as scalability, ethical concerns, and maintaining instructor involvement persist. Studies also emphasize the need for better infrastructure, educator training, and integration across various educational systems. Future work aims to overcome these limitations and explore AI's broader application in diverse educational settings. These efforts collectively highlight AI's potential to improve both teaching and learning outcomes.

4.1.1 DRAWBACKS OF EXISTING SYSTEM

The existing systems, while showcasing the power of AI in education, have several drawbacks. A major limitation is the lack of an integrated platform that offers comprehensive educational support for students, including exam preparation, personalized study plans, and continuous progress tracking. While AI tools provide personalized content and feedback, they often operate in isolation without proper synchronization across different learning modules. Additionally, many systems focus on a single subject area, limiting their scalability and applicability to a broader range of disciplines. Ethical issues, such as data privacy and transparency, remain largely unaddressed, particularly in systems involving sensitive student information. Furthermore, these platforms often fail to consider the long-term effectiveness of AI-driven learning solutions in maintaining consistent engagement and improving academic performance across diverse educational environments. Lastly, challenges like resource constraints, lack of educator training, and infrastructure issues prevent seamless adoption of these AI-based systems in traditional classrooms.

4.2 PROPOSED SYSTEM

PrepMaster is an AI-driven platform designed to enhance the learning and exam preparation journey for students across different educational stages. The platform offers AI-powered tutor, exam strategy planner, and quiz generator that adapt to a learner's progress, ensuring they are always challenged appropriately. AI Tutor built on the LLaMA model, which provides personalized explanations and assistance, an Exam Strategy Planner that helps students plan their study schedule around exams, and a Quiz Generator powered by the BART model, which creates dynamic quizzes that assess and reinforce learning. PrepMaster provides real-time feedback and tracks student progress, helping them identify areas of improvement and fine-tune their study approach. The system promotes effective time management by suggesting study breaks and helping students create balanced study routines.This integrated approach not only supports exam preparation but also cultivates essential skills like problem-solving and critical thinking**.**

4.2.1 ADVANTAGES OF PROPOSED SYSTEM

PrepMaster is an integrated and adaptive learning platform that aims to revolutionize exam preparation for students. The system uses the LLaMA model as the AI tutor, providing personalized guidance and interactive lessons tailored to the individual needs of each student. This AI-driven tutor ensures students receive accurate, clear explanations and support throughout their learning journey. The Exam Strategy Planner within PrepMaster helps students organize their study schedule, prioritizing subjects and topics based on their previous year question papers. This intelligent planner adapts to the student's progress, ensuring that study time is utilized effectively and efficiently. The BART model powers the Quiz Generator, creating adaptive quizzes based on the topics the student needs to focus on, with increasing difficulty based on their performance. This adaptive approach ensures that students are continuously challenged without being overwhelmed. By combining these models and features, PrepMaster offers a dynamic approach to exam preparation, making studying more organized, personalized, and effective.

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 SYSTEM ARCHITECTURE

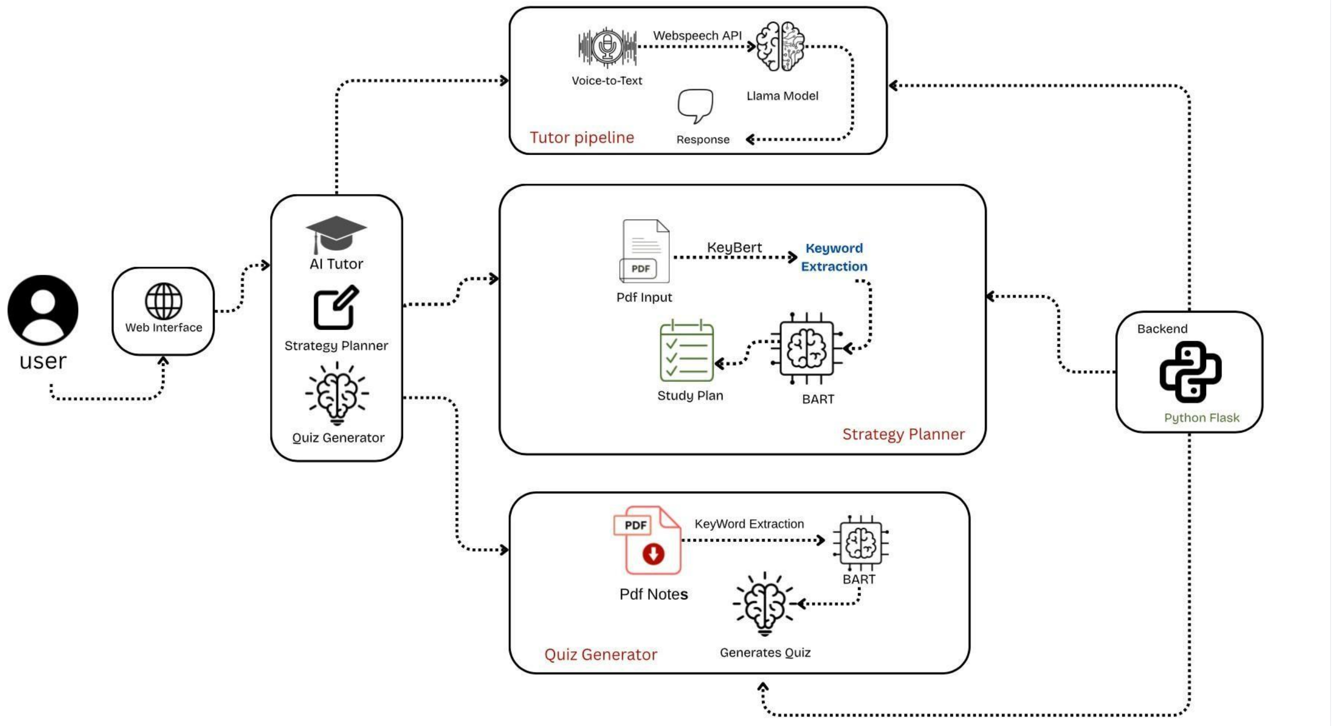


Fig 5.1: *System Architecture of PrepMaster:AI-Powered Personalized Academic Assistant*

PrepMaster is an AI-powered learning platform composed of three main modules: Tutor, Exam Planner, and Quiz Generator. Users interact through a web interface, accessing personalized educational support. The Tutor module manages real-time conversations by converting user speech to text, processing it through a Llama-based language model, and delivering intelligent, contextual responses. The Exam Planner module processes uploaded PDF study materials by performing keyword extraction generating detailed study plans using the BART model. Simultaneously, the Quiz Generator module extracts content from PDF notes and uses the BART model to automatically generate customized quizzes for active recall practice. The modules are integrated with a Python Flask backend. Together, these components empower PrepMaster to deliver a highly personalized, dynamic, and efficient learning experience.

5.2 SYSTEM WORKFLOW

The system workflow for PrepMaster initiates with the user inputting study materials via PDF uploads or voice or text. Following this, the system undertakes text extraction and preprocessing to ready the content for the intelligent core models. The AI tutor then processes this prepared text to generate tailored learning tools. The exam planner model encompasses a structured study plan for effective learning, automatically created quiz questions for self-evaluation, and AI-driven responses to user inquiries, serving as a personalized tutor. Ultimately, this suite of resources is delivered through a user-friendly interface, an engaging and customized educational journey.

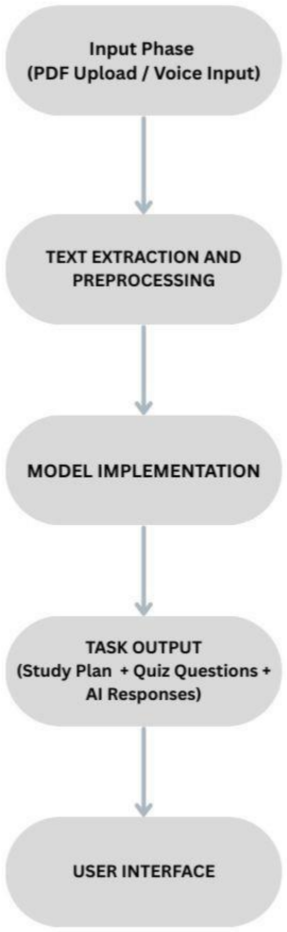

Input Phase
(PDF Upload / Voice Input)
TEXT EXTRACTION AND PREPROCESSING
MODEL IMPLEMENTATION
TASK OUTPUT
(Study Plan + Quiz Questions + AI Responses)
USER INTERFACE

Fig 5.2: *System Workflow of PrepMaster: AI-Powered Personalized Academic Assistant*

5.3 LIST OF MODULES

- User Interaction and Voice Processing Module
- AI Tutor Module
- Exam Study Plan Generator Module
- Quiz Generator Module
- Backend Integration and Deployment Module

5.4 MODULE DESCRIPTION

5.4.1 USER INTERACTION AND VOICE PROCESSING MODULE

The User Interaction and Voice Processing Module serves as the entry point for users engaging with the PrepMaster platform. In this module, the user’s voice input is first captured through a web interface. The captured voice data is processed using a speech-to-text engine that transcribes spoken language into written text with high accuracy. After transcription, the text undergoes tokenization, breaking it into manageable subword units suitable for language model processing. This step is critical to ensuring that even complex or lengthy user inputs are parsed efficiently without losing meaning. Additionally, the module is designed for real-time operation, minimizing latency between user input and system response.

5.4.2 AI TUTOR MODULE

The Tutor Module acts as the core intelligent conversational agent within PrepMaster, responsible for addressing and resolving user queries. After receiving tokenized text from the voice processing pipeline, the Tutor module utilizes a fine-tuned Llama language model to interpret and respond to the input. Internally, the model processes the tokens through multiple layers of self-attention, calculating attention scores that help the system understand the contextual relationships between words, even across long inputs. These relationships enable the model to retain conversation history, grasp nuances in questions, and generate appropriate, context-aware responses. The module predicts the next sequence of words using a probability distribution over the vocabulary, ensuring fluid and coherent communication. By combining natural language understanding with dynamic generation capabilities, the Tutor Module provides users with an intelligent, personalized tutoring experience.

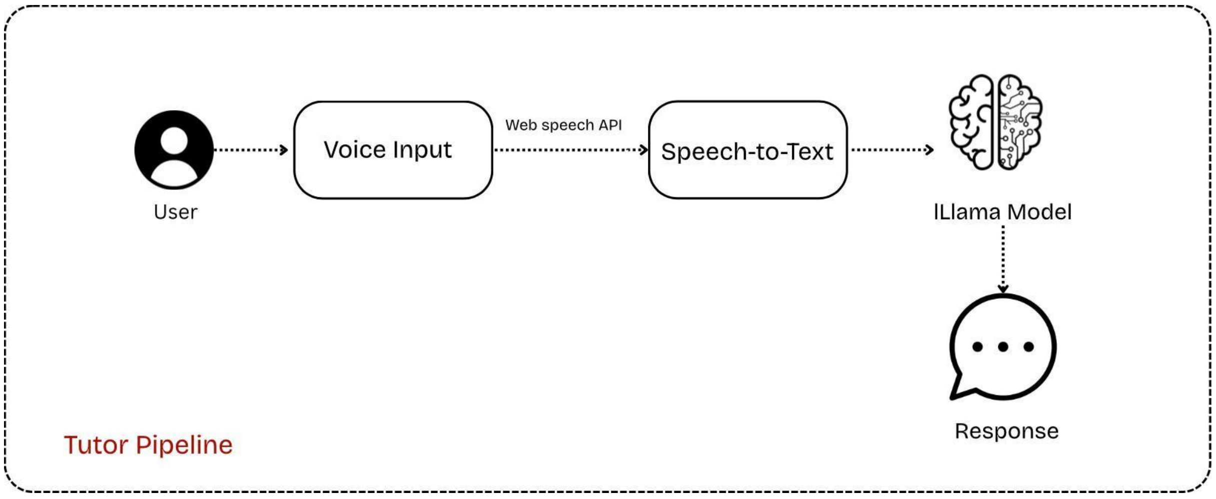


Fig 5.3.1 *Workflow of AI tutor pipeline*

5.4.3 EXAM STUDY PLAN GENERATOR MODULE

The Exam Planner Module plays a crucial role in helping students manage and structure their study time effectively. When users upload PDF notes, this module first extracts the text content and applies Llama Indexing techniques to identify critical topics, concepts, and keywords within the material. By focusing on important segments of the text, the module ensures that the subsequent study plan emphasizes areas most relevant for exams. The extracted information is then processed through a BART-based sequence-to-sequence model, where the encoder captures the input context and the decoder generates a coherent study roadmap. Through the use of cross-attention layers, the model maintains consistency between topics and organizes content into logical study sessions. The final study plan is structured to balance depth and breadth, ensuring thorough coverage of subjects without overwhelming the learner. This module empowers users to approach their academic preparation strategically, focusing their efforts where it matters most and saving valuable time.

5.4.4 QUIZ GENERATOR MODULE

The Quiz Generator Module is designed to enhance learning by promoting active recall, one of the most effective study techniques. After extracting and analyzing content from uploaded study materials, this module employs a BART-based text generation model to create customized quizzes tailored to the user’s learning content. During quiz generation, the model carefully formulates question-answer pairs, ensuring that each question is relevant, factually accurate, and directly linked to the uploaded materials. The system dynamically generates multiple-choice questions providing users with diverse practice opportunities. By simulating exam-like questioning, the Quiz Generator helps reinforce memory retention, strengthen understanding of core concepts, and prepare users more effectively for real assessment conditions.

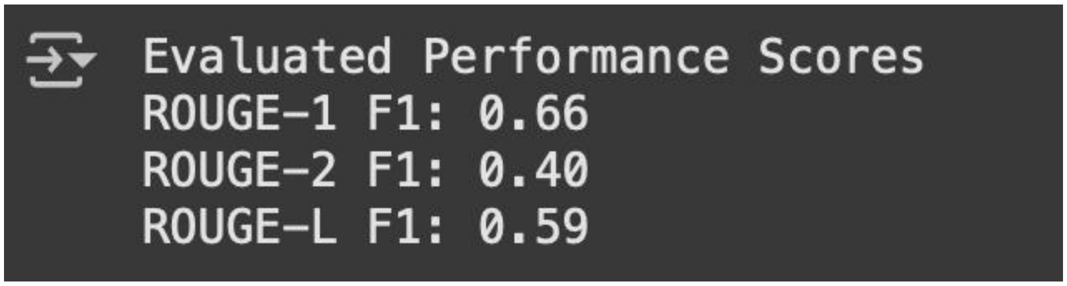
5.4.5 BACKEND INTEGRATION AND DEPLOYMENT MODULE

The Backend Integration and Deployment Module keeps everything running smoothly by coordinating all the different parts of the PrepMaster system, ensuring they work seamlessly together. Built on a robust Python Flask backend, this module is responsible for managing API endpoints, coordinating requests between the frontend user interface and the different AI-driven modules, and ensuring secure session handling. It handles critical tasks such as authentication, routing of user queries to the appropriate processing units, and aggregation of model outputs before presenting them back to the user. The backend is designed for scalability and reliability, supporting multiple simultaneous users while maintaining fast response times. It acts as the backbone of the PrepMaster system, enabling seamless, real-time communication between users and the advanced AI models that drive tutoring, exam planning, and quiz generation functionalities.

CHAPTER 6

RESULT AND DISCUSSION

The PrepMaster platform has shown strong potential in enhancing personalized learning through its AI tutor and adaptive quiz generation features. The AI tutor provides real-time, tailored support, helping students grasp concepts more effectively. Adaptive quizzes reinforce learning by adjusting difficulty based on individual performance. Student feedback indicates noticeable improvements in academic outcomes, driven by personalized study plans that address specific strengths and weaknesses. The platform's intuitive interface also contributes to sustained engagement and ease of use. The quality of the system's generated content was evaluated using ROUGE metrics, achieving ROUGE-1 F1: 0.66, ROUGE-2 F1: 0.40, and ROUGE-L F1: 0.5 demonstrating its effectiveness in producing relevant and coherent summaries. Future developments will focus on expanding subject coverage, adding collaborative learning features like peer study groups, introducing voice-based tutoring, and implementing real-time performance analytics to support continuous improvement.



```
Evaluated Performance Scores
ROUGE-1 F1: 0.66
ROUGE-2 F1: 0.40
ROUGE-L F1: 0.59
```

Fig 6.1 *Evaluated Performance Scores for PrepMaster*

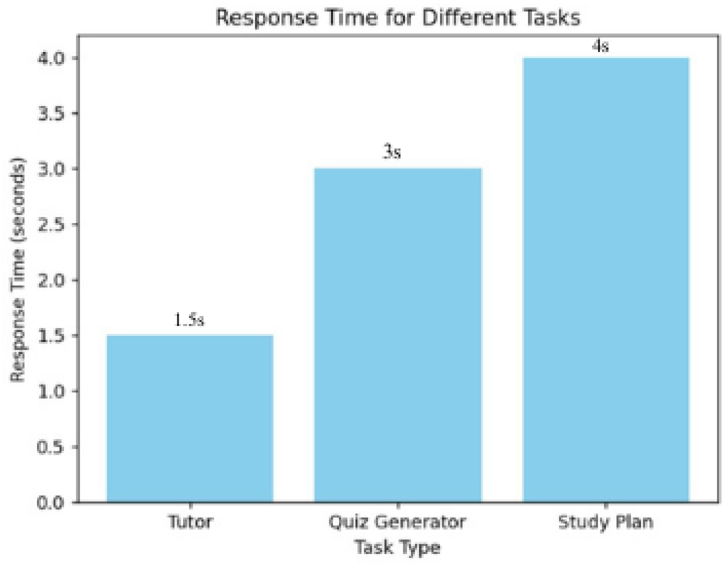
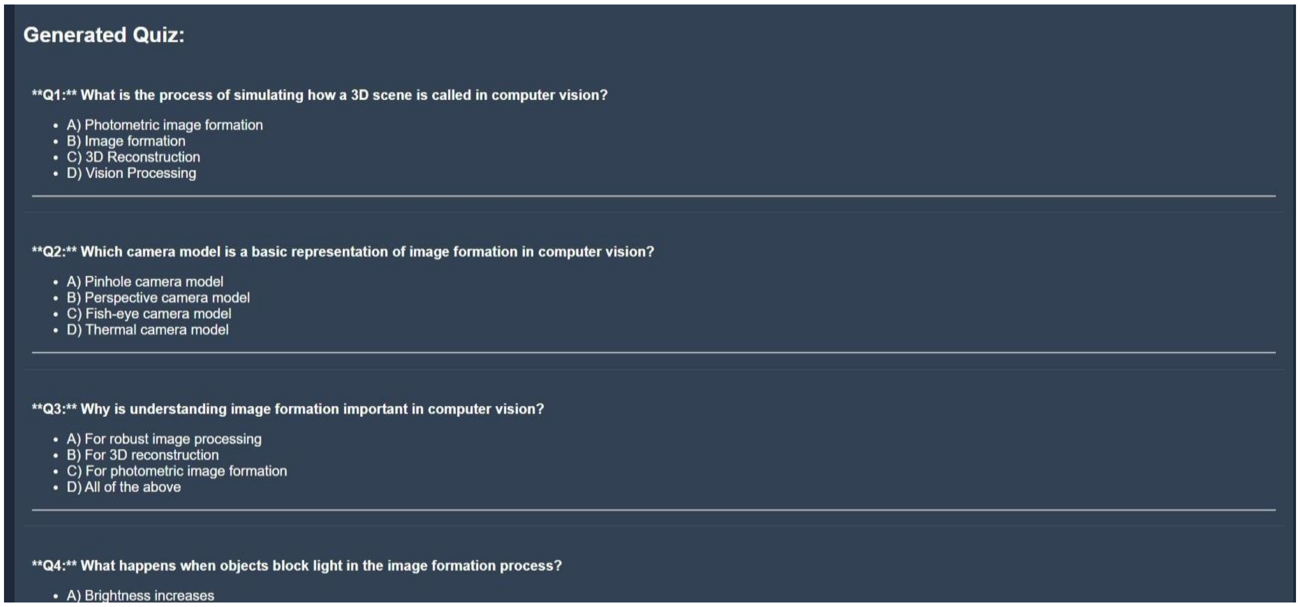


Fig 6.2 *Response time for different tasks*

The bar graph in Fig 6.2 compares the response time for different tasks: Tutor, Quiz Generator, and Study Plan. Among the three, the Study Plan task takes the longest time to respond, at approximately 4 seconds. The Quiz Generator task follows with around 3 seconds of response time. In contrast, the Tutor task has the shortest response time, about 1.5 seconds. This indicates that tasks requiring more planning or data processing, like Study Plan generation, tend to take longer. The system appears to be most efficient in handling simple, direct tasks like tutoring.



Generated Quiz:

Q1: What is the process of simulating how a 3D scene is called in computer vision?

- A) Photometric image formation
- B) Image formation
- C) 3D Reconstruction
- D) Vision Processing

Q2: Which camera model is a basic representation of image formation in computer vision?

- A) Pinhole camera model
- B) Perspective camera model
- C) Fish-eye camera model
- D) Thermal camera model

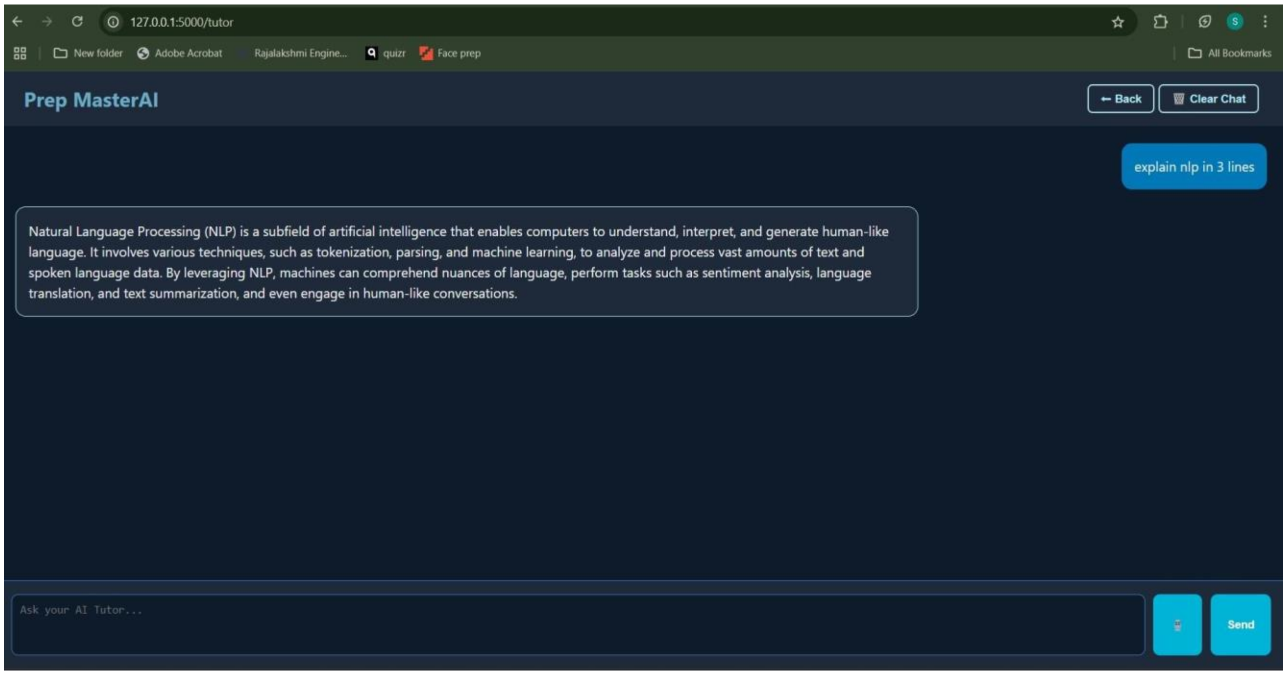
Q3: Why is understanding image formation important in computer vision?

- A) For robust image processing
- B) For 3D reconstruction
- C) For photometric image formation
- D) All of the above

Q4: What happens when objects block light in the image formation process?

- A) Brightness increases

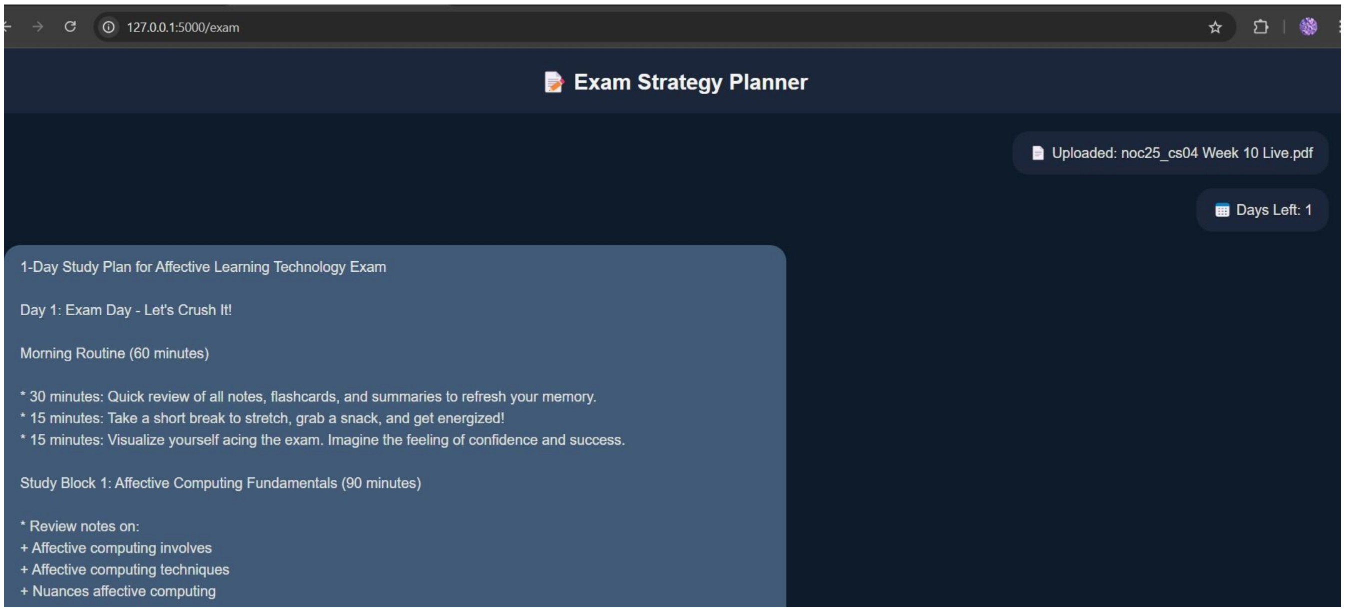
Fig 6.3 *AI Quiz generator*



explain nlp in 3 lines

Natural Language Processing (NLP) is a subfield of artificial intelligence that enables computers to understand, interpret, and generate human-like language. It involves various techniques, such as tokenization, parsing, and machine learning, to analyze and process vast amounts of text and spoken language data. By leveraging NLP, machines can comprehend nuances of language, perform tasks such as sentiment analysis, language translation, and text summarization, and even engage in human-like conversations.

Fig 6.4 *AI Tutor page*



1-Day Study Plan for Affective Learning Technology Exam

Day 1: Exam Day - Let's Crush It!

Morning Routine (60 minutes)

* 30 minutes: Quick review of all notes, flashcards, and summaries to refresh your memory.
* 15 minutes: Take a short break to stretch, grab a snack, and get energized!
* 15 minutes: Visualize yourself acing the exam. Imagine the feeling of confidence and success.

Study Block 1: Affective Computing Fundamentals (90 minutes)

* Review notes on:
+ Affective computing involves
+ Affective computing techniques
+ Nuances affective computing

Fig 6.5 *Exam Strategy planner*

MATHEMATICAL CALCULATION

ROUGE Score Calculation:

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) measures the similarity between a generated text and a reference text, commonly used in language generation tasks.

- ROUGE-1: Unigram (single word) overlap.
- ROUGE-2: Bigram (two-word sequence) overlap.
- ROUGE-L: Longest Common Subsequence (measures fluency and sequence similarity).

$$\text{ROUGE-N} = \frac{\text{Number of overlapping N-grams}}{\text{Total number of N-grams in the reference}}$$

$$\text{Precision} = \frac{\text{Number of matching n-grams in candidate and reference}}{\text{Total number of n-grams in candidate}}$$

$$\text{Recall} = \frac{\text{Number of matching n-grams in candidate and reference}}{\text{Total number of n-grams in reference}}$$

$$\text{F1} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

Example:

Reference text:
“Image formation is the process of capturing a 3D scene onto a 2D plane using a camera lens.”

Generated text:
“Image formation captures a 3D scene on a 2D surface using the lens.”

ROUGE-1 (Single words):

- Matching words = 9
- Words in reference = 14
- Words in generated = 13

Precision = 9 ÷ 13 = 0.69
Recall = 9 ÷ 14 = 0.64
F1 Score = 0.66

ROUGE-2 (Two-word phrases):

- Matching bigrams = 5
- Total bigrams in reference = 13
- Total in generated = 12

Precision = 5 ÷ 12 = 0.42
Recall = 5 ÷ 13 = 0.38
F1 Score = 0.40

ROUGE-L (Longest common word sequence):

- Longest match = 8 words
- Generated length = 13, Reference length = 14

Precision = 8 ÷ 13 = 0.62
Recall = 8 ÷ 14 = 0.57
F1 Score = 0.59

Final Scores:

- **ROUGE-1 F1**: 0.66
- **ROUGE-2 F1**: 0.40
- **ROUGE-L F1**: 0.59

These scores show how well the generated text matches the original explanation.

APPENDIX

SAMPLE CODE

```
from flask import Flask, request, jsonify, render_template
from flask_cors import CORS
import os
import PyPDF2
from keybert import KeyBERT
from transformers import AutoModelForCausalLM, AutoTokenizer
import torch
import logging

app = Flask(__name__, static_folder="static", template_folder="templates")
CORS(app)

logging.basicConfig(level=logging.DEBUG)

llama_model_tutor=AutoModelForCausalLM.from_pretrained("meta-llama/Llama-3b-hf)
llama_tokenizer_tutor = AutoTokenizer.from_pretrained("meta-llama/Llama-3b-hf")

# Load BART Model for Quiz Generator and Exam Planner
bart_model_quiz_exam =
AutoModelForCausalLM.from_pretrained("facebook/bart-large")
bart_tokenizer_quiz_exam  =  AutoTokenizer.from_pretrained("facebook/bart-large")


# Serve Homepage
@app.route('/')
def home():
    return render_template('index.html')

# Serve AI Tutor Page
@app.route('/tutor')
def tutor():
    return render_template('tutor.html')

# Serve Exam Planner Page
@app.route('/exam')
def exam():
    return render_template('exam.html')

#  Serve AI Quiz Generator Page
@app.route('/quiz')
def quiz():
    return render_template('quiz.html')


# AI Tutor API (LLaMA Model )

@app.route('/ask-tutor', methods=['POST'])
def ask_tutor():
    data = request.get_json()
    question = data.get("question", "")

    if not question:
        return jsonify({"answer": "Please ask a valid question."})

    try:
        answer = ask_llama(question)
    except Exception as e:
        return jsonify({"answer": "Sorry, something went wrong with the AI response."})

    return jsonify({"answer": answer})

def ask_llama(question):
    """Fetch response from LLaMA model for tutor."""
    inputs = llama_tokenizer_tutor(question, return_tensors="pt")
    outputs = llama_model_tutor.generate(inputs['input_ids'], max_length=150,
num_return_sequences=1)
    answer = llama_tokenizer_tutor.decode(outputs[0], skip_special_tokens=True)
    return answer


# Quiz Generator

def extract_pdf_text(file):
    """Extract text content from the uploaded PDF file."""
    try:
        pdf_reader = PyPDF2.PdfReader(file)
        text = ""
        for page in pdf_reader.pages:
            page_text = page.extract_text()
            if page_text:
                text += page_text + "\n"
        return text
    except Exception as e:
        print(f"Error extracting text from PDF: {e}")
        return ""

def extract_topic_content(pdf_text, topic):
    """Find relevant text for the topic."""
    lines = pdf_text.split("\n")
    filtered_content = [line for line in lines if topic.lower() in line.lower()]
    return " ".join(filtered_content).strip()

def extract_keywords(content, num_keywords=20):
    """Extract keywords from content using KeyBERT."""
    kw_model = KeyBERT()
    keywords = kw_model.extract_keywords(content, keyphrase_ngram_range=(1, 3),
stop_words='english', top_n=num_keywords)
    return [kw[0] for kw in keywords]

def generate_mcqs_bart(topic, keywords, content, num_questions=5,
difficulty="medium"):
    prompt = f"""
You are an expert quiz generator. Generate {num_questions} {difficulty} level MCQs
based on the study content below.

--- Study Content ---
{content}

--- Keywords ---
{', '.join(keywords)}

--- Format ---
Q: <question>
A. <option1>
B. <option2>
C. <option3>
D. <option4>
"""
```

```
    inputs = bart_tokenizer_quiz_exam(prompt, return_tensors="pt", truncation=True,
padding=True)
    outputs = bart_model_quiz_exam.generate(inputs['input_ids'], max_length=250,
num_return_sequences=1)
    quiz_output = bart_tokenizer_quiz_exam.decode(outputs[0],
skip_special_tokens=True)
    return quiz_output

@app.route('/generate-quiz', methods=['POST'])
def generate_quiz():
    if 'file' not in request.files:
        return jsonify({"error": " No file part"}), 400
    file = request.files['file']
    topic = request.form.get('topic', "").strip()
    difficulty = request.form.get('difficulty', "medium")
    num_questions = int(request.form.get('num_questions', 5))

    if not file or not topic:
        return jsonify({"error": " Please provide a valid PDF and topic."}), 400

    try:
        pdf_text = extract_pdf_text(file)
        content = extract_topic_content(pdf_text, topic)

        if not content:
            return jsonify({"error": " No relevant content found for the topic!"}), 400

        keywords = extract_keywords(content)

        quiz_output = generate_mcqs_bart(topic, keywords, content, num_questions,
difficulty)

        quiz_questions = []
        for q in quiz_output.strip().split("\n\n"):
            parts = q.split("\n")
            if len(parts) >= 4:
                question = parts[0].replace("Q: ", "") if len(parts) > 0 else ""
                options = [parts[i][3:] for i in range(1, 5) if i < len(parts)]
                if question and len(options) == 4:
                    quiz_questions.append({
                        "question": question,
                    "options": options
                })

        return jsonify({"quiz": quiz_questions})

    except Exception as e:
        return jsonify({"error": "Error fetching quiz. Try again!"}), 500

# Exam Planner (BART)

def generate_study_plan_with_bart(keywords, days_left):
    """Generate study plan using BART model."""
    prompt = f"""
You are an expert study planner.
Based on the following important topics extracted from my notes:

{', '.join(keywords)}

I have {days_left} days left before my exam.
Generate a very detailed day-by-day study plan.

- Prioritize important topics first.
- Mention exactly what to study each day.
- Include study tips and revision strategies.
- Be very structured and motivational.

Format it nicely day by day.
"""
    inputs = bart_tokenizer_quiz_exam(prompt, return_tensors="pt", truncation=True, 
padding=True)
    outputs = bart_model_quiz_exam.generate(inputs['input_ids'], max_length=250, 
num_return_sequences=1)
    study_plan = bart_tokenizer_quiz_exam.decode(outputs[0], skip_special_tokens=True)
    return study_plan

@app.route('/generate-strategy', methods=['POST'])
def generate_strategy():
    """Receive PDF, extract content, generate study plan."""
    if 'file' not in request.files:
        return jsonify({"error": " No file part"}), 400
    file = request.files['file']
    days_left = int(request.form.get('days_left', 0))
    try:
        # Extract PDF text
        pdf_text = extract_pdf_text(file)

        # Extract Keywords
        keywords = extract_keywords(pdf_text)

        # Generate Study Plan
        study_plan = generate_study_plan_with_bart(keywords, days_left)

        if study_plan:
            return jsonify({"strategy": study_plan})
        else:
            return jsonify({"error": " Failed to generate study plan."}), 500

    except Exception as e:
        print(f"Error generating strategy: {str(e)}")
        return jsonify({"error": " Error generating strategy. Try again!"}), 500


#  Run Flask App
if __name___== '__main__':
    app.run(debug=True, port=5000)
  return jsonify({"quiz": quiz_questions})

# Run Flask App
if __name___== '__main__':
    app.run(debug=True, port=5000)
```

OUTPUT SCREENSHOTS

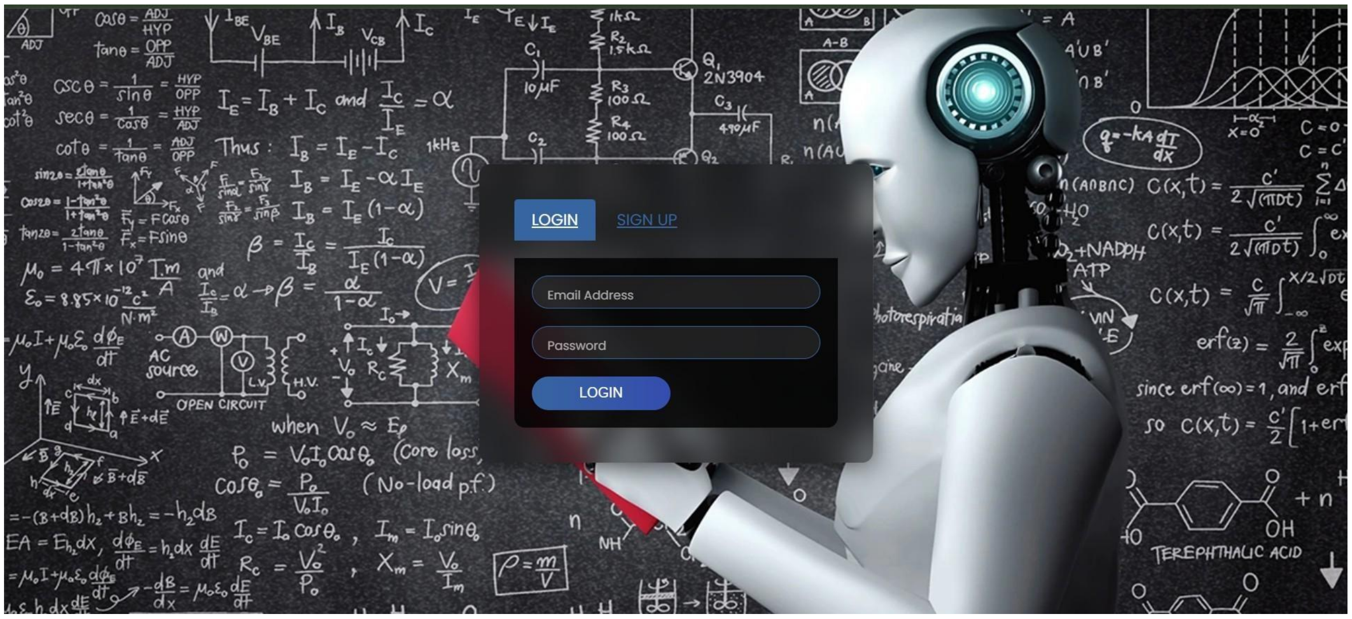


Fig A.1: *Login Page*

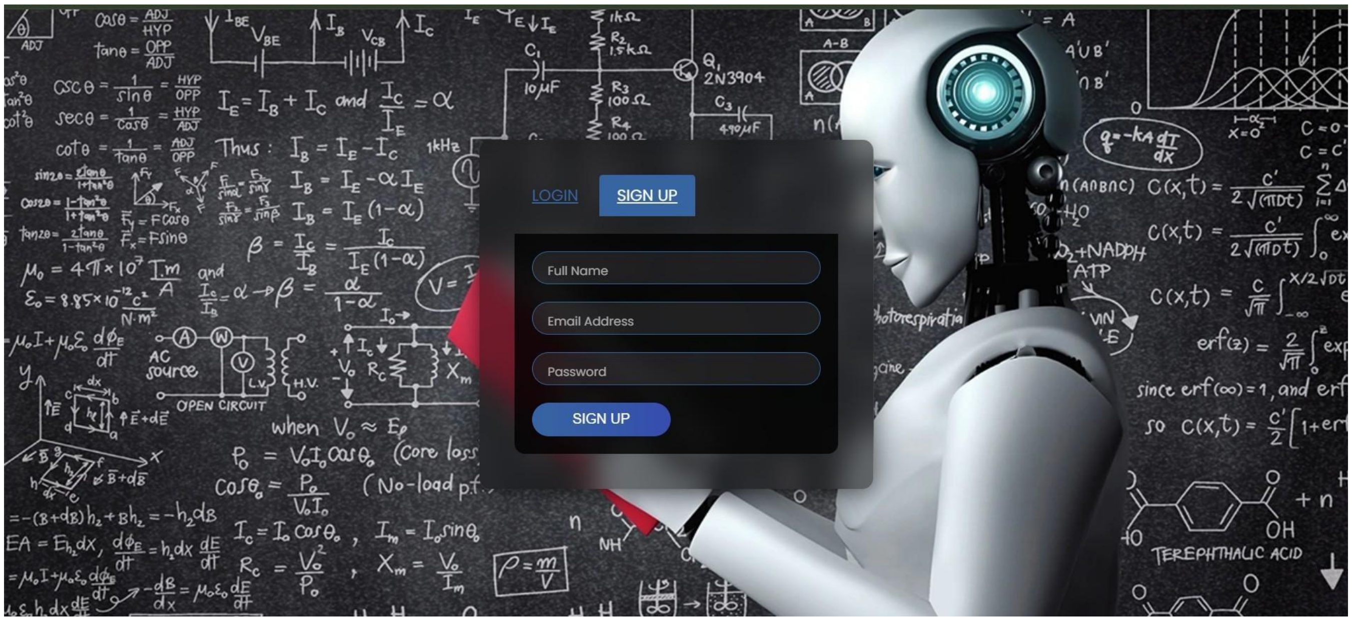
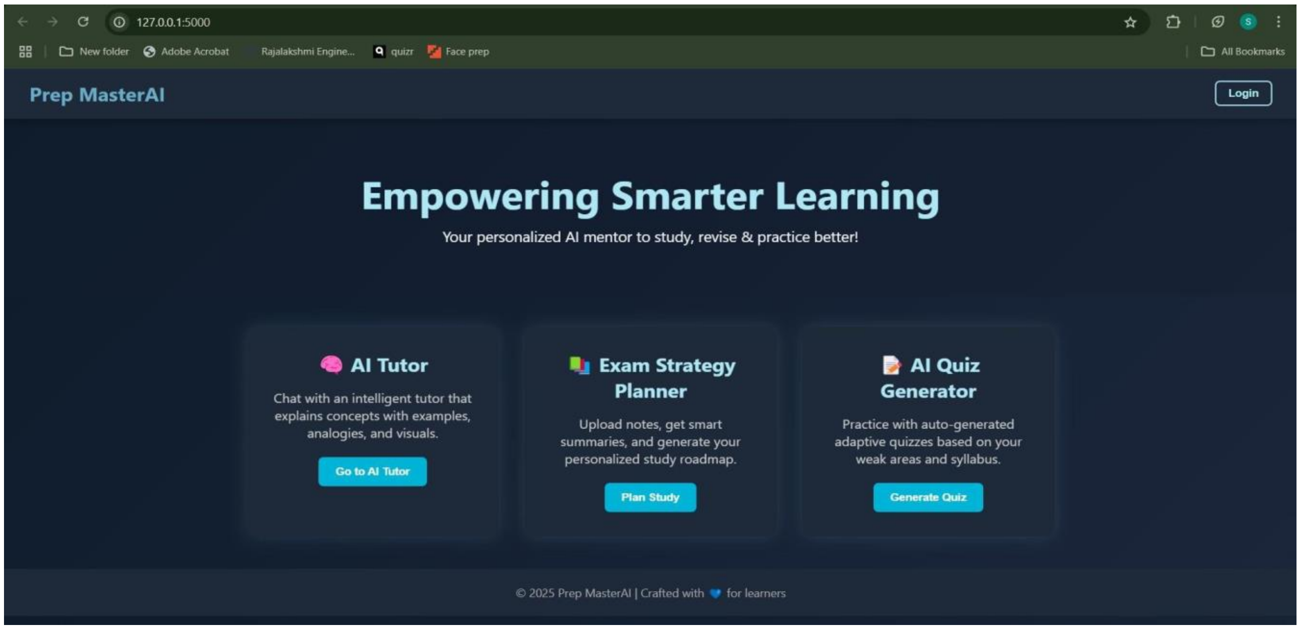


Fig A.2: *Sign Up Page*



Empowering Smarter Learning

Your personalized AI mentor to study, revise & practice better!

AI Tutor

Chat with an intelligent tutor that explains concepts with examples, analogies, and visuals.

Exam Strategy Planner

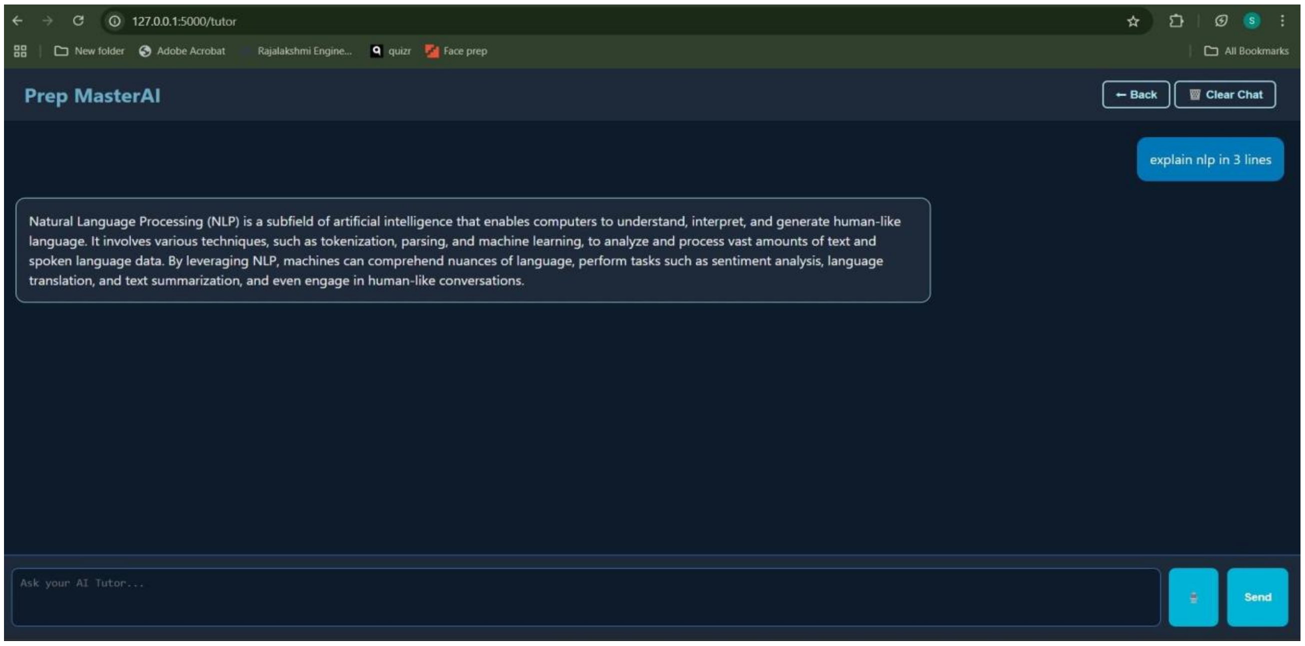
Upload notes, get smart summaries, and generate your personalized study roadmap.

AI Quiz Generator

Practice with auto-generated adaptive quizzes based on your weak areas and syllabus.

© 2025 Prep MasterAI | Crafted with 💙 for learners

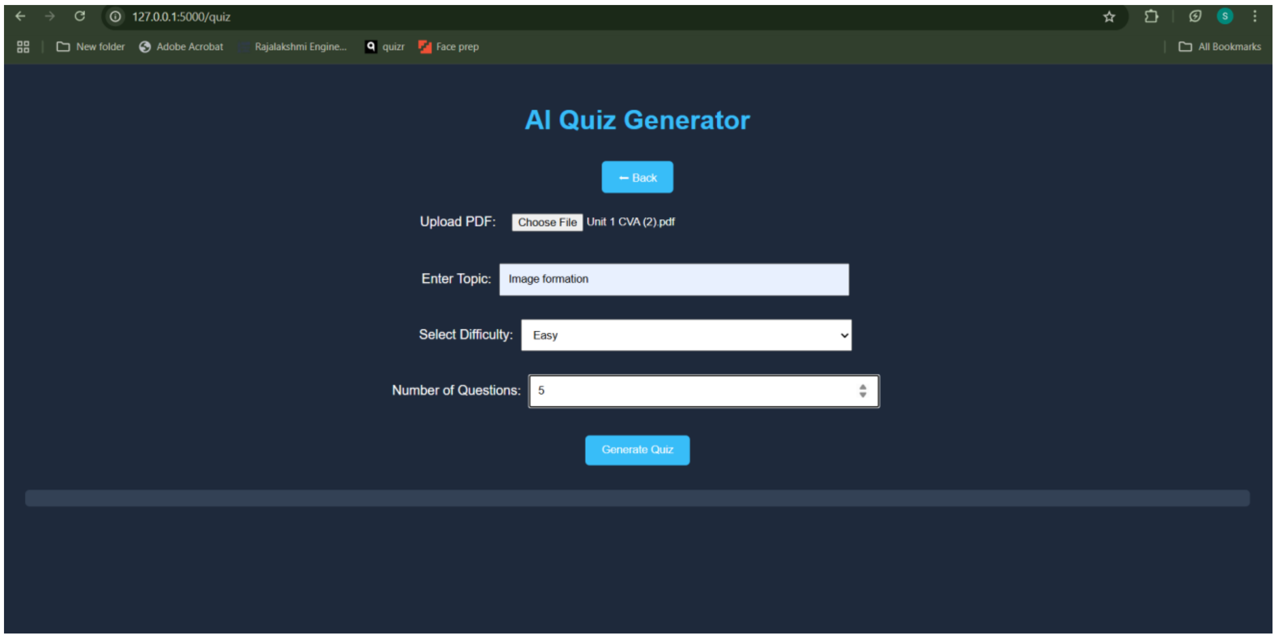
Fig A.3: *Home page*



explain nlp in 3 lines

Natural Language Processing (NLP) is a subfield of artificial intelligence that enables computers to understand, interpret, and generate human-like language. It involves various techniques, such as tokenization, parsing, and machine learning, to analyze and process vast amounts of text and spoken language data. By leveraging NLP, machines can comprehend nuances of language, perform tasks such as sentiment analysis, language translation, and text summarization, and even engage in human-like conversations.

Fig A.4: *AI Tutor Page*



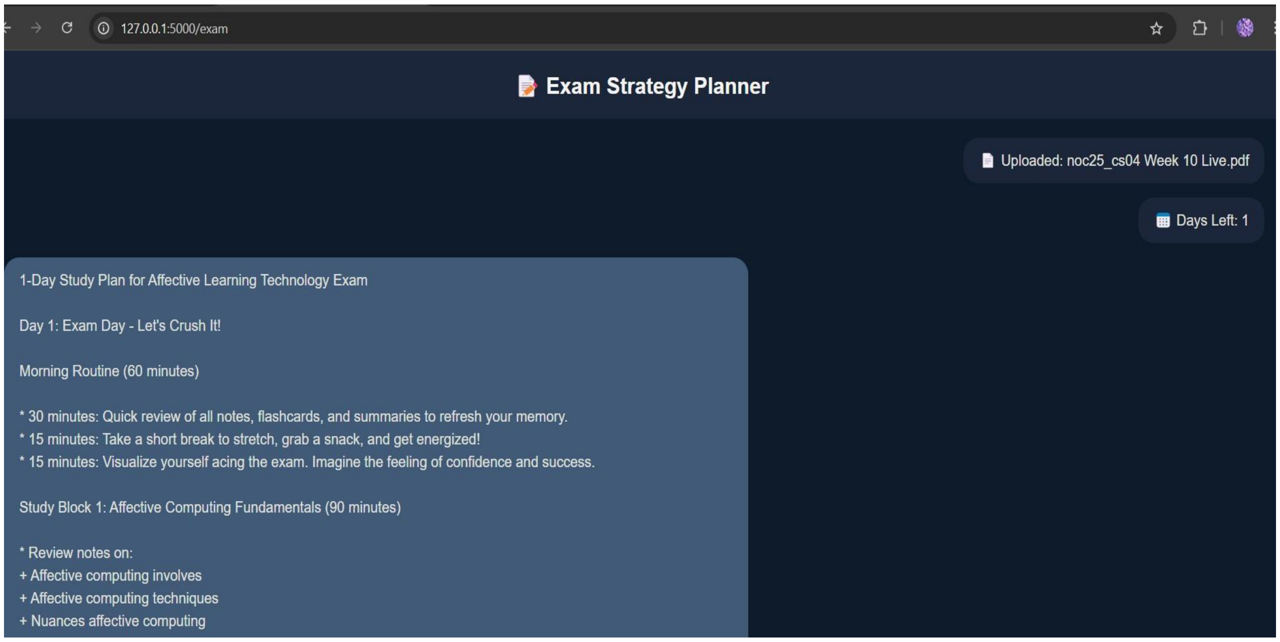
Upload PDF: Unit 1 CVA (2).pdf

Enter Topic: Image formation

Select Difficulty: Easy

Number of Questions: 5

Fig A.5: *AI Quiz Generator Page*



Uploaded: noc25_cs04 Week 10 Live.pdf

Days Left: 1

1-Day Study Plan for Affective Learning Technology Exam

Day 1: Exam Day - Let's Crush It!

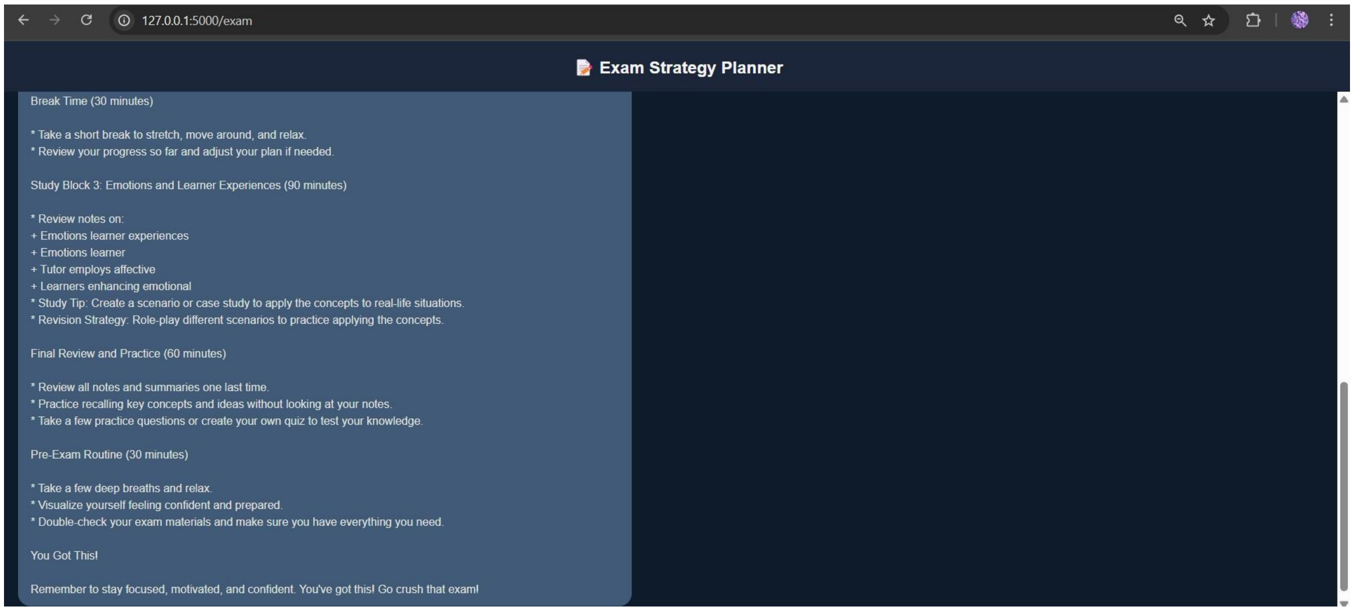
Morning Routine (60 minutes)

* 30 minutes: Quick review of all notes, flashcards, and summaries to refresh your memory.
* 15 minutes: Take a short break to stretch, grab a snack, and get energized!
* 15 minutes: Visualize yourself acing the exam. Imagine the feeling of confidence and success.

Study Block 1: Affective Computing Fundamentals (90 minutes)

* Review notes on:
+ Affective computing involves
+ Affective computing techniques
+ Nuances affective computing

Fig A.6: *Uploading the days left and pdf file*



```
Break Time (30 minutes)

* Take a short break to stretch, move around, and relax.
* Review your progress so far and adjust your plan if needed.

Study Block 3: Emotions and Learner Experiences (90 minutes)

* Review notes on:
+ Emotions learner experiences
+ Emotions learner
+ Tutor employs affective
+ Learners enhancing emotional
* Study Tip: Create a scenario or case study to apply the concepts to real-life situations.
* Revision Strategy: Role-play different scenarios to practice applying the concepts.

Final Review and Practice (60 minutes)

* Review all notes and summaries one last time.
* Practice recalling key concepts and ideas without looking at your notes.
* Take a few practice questions or create your own quiz to test your knowledge.

Pre-Exam Routine (30 minutes)

* Take a few deep breaths and relax.
* Visualize yourself feeling confident and prepared.
* Double-check your exam materials and make sure you have everything you need.

You Got This!

Remember to stay focused, motivated, and confident. You've got this! Go crush that exam!
```

Fig A.7: *Output from the exam planner model*

REFERENCES

[1] Mayormente, M. D., & Gumpal, B. R. (2025). NLP-based sentiment analysis for evaluating student feedback in English language education. In 2025 International Conference on Multi-Agent Systems for Collaborative Intelligence (ICMSCI) (pp. 112–118).

[2] Bonde, L. (2024). A generative artificial intelligence based tutor for personalized learning. In 2024 IEEE SmartBlock4Africa (pp. 1–10). IEEE.

[3] Roldán-Álvarez, D., & Mesa, F. J. (2024). Intelligent deep-learning tutoring system to assist instructors in programming courses. IEEE Transactions on Education, 67(1), 153–161.

[4] Chang, C.-K., & Chien, L.-C.-T. (2024). Enhancing academic performance with generative AI-based quiz platform. In 2024 IEEE International Conference on Advanced Learning Technologies (ICALT) (pp. 193–195). IEEE.

[5] Shahri, H., Emad, M., Ibrahim, N., Rais, R. N. B., & Al-Fayoumi, Y. (2024). Elevating education through AI tutor: Utilizing GPT-4 for personalized learning. In 2024 15th Annual Undergraduate Research Conference on Applied Computing (URC) (pp. 1–5).

[6] Frankford, E., Sauerwein, C., Bassner, P., Krusche, S., & Breu, R. (2024). AI-tutoring in software engineering education: Experiences with large language models in programming assessments. In 2024 IEEE/ACM 46th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET) (pp. 309–319).

[7] Barde, A., Thakur, R., Patel, S., Sinah, N., & Barde, S. (2024). AI-based smart education system to enhance the learning of students. In 2024 International Conference on Advances in Computing Research on Science Engineering and Technology (ACROSET) (pp. 1–7).

[8] Duraes, D., Bezerra, R., & Novais, P. (2024). AI-driven educational transformation in secondary schools: Leveraging data insights for inclusive learning environments. In 2024

IEEE Global Engineering Education Conference (EDUCON) (pp. 1–9).

[9] Besimi, N., Besimi, A., & Çiço, B. (2022). Artificial intelligence in education and learning (AI in research). In 2022 11th Mediterranean Conference on Embedded Computing (MECO) (pp. 1–6). IEEE.

[10] Mavalankar, A., Kelkar, T., & Choppella, V. (2015). Generation of quizzes and solutions based on ontologies: A case for a music problem generator. In 2015 IEEE Seventh International Conference on Technology for Education (T4E) (pp. 73–76).