



International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

AI-Based UGC NET Quiz Generator Using Generative AI

A. Asiyathul Sajitha , Dr. J. Evangelin Deva Sheela

Department of Computer Applications, Sarah Tucker College, Tirunelveli-7

ABSTRACT

Preparing for competitive examinations such as the University Grants Commission National Eligibility Test (UGC NET) requires continuous practice with subject-specific multiple-choice questions. Conventional online quiz systems generally depend on predefined question banks, which may provide limited question variety and require manual effort to create and maintain questions. This paper presents an AI-Based UGC NET Quiz Generator that uses Generative Artificial Intelligence to automatically create multiple-choice questions for UGC NET Computer Science and Applications preparation. The proposed system is developed using Python and Flask for the web application, MySQL for data management, and Gemini AI for question generation. The system provides Computer Science topics such as Programming in C, Computer Networks, Operating Systems, Database Management Systems, and Artificial Intelligence. Based on the selected topic, Gemini AI generates multiple-choice questions with four options, the correct answer, and an explanation. Students can attempt the generated quiz through an interactive interface, while the system automatically evaluates submitted answers and calculates the score and percentage. Quiz attempts are stored in the database so that students can review their previous performance through quiz history. A topic-wise leaderboard is also provided to display student rankings based on quiz performance. The proposed system provides a flexible approach to AI-assisted examination preparation by reducing manual question creation and providing topic-based practice.

Keywords: UGC NET, Generative AI, Gemini AI, Multiple Choice Questions, Quiz Generator, Computer Science, Flask, MySQL, Educational Technology.

1. Introduction

The University Grants Commission National Eligibility Test (UGC NET) is a competitive examination conducted for determining eligibility for Assistant Professor and Junior Research Fellowship opportunities in India. The examination requires candidates to prepare systematically across different subjects and topics. Computer Science and Applications is one of the subjects included in the examination and comprises topics such as programming in C, computer networks, operating systems, databases, and artificial intelligence.

Regular practice with multiple-choice questions is an important part of examination preparation. Students commonly use textbooks, previous-year question papers, online quiz platforms, and manually prepared question banks for practice. Although these resources are useful, traditional quiz systems generally depend on a fixed collection of questions. Once the available questions have been attempted, students may repeatedly receive the same questions.

Artificial Intelligence (AI) and Generative AI provide an opportunity to improve automated educational systems. Large language models can generate text-based questions according to a given topic and instruction. This capability can be used to create different questions for practice without requiring every question to be manually entered into a database.

The proposed work develops an AI-Based UGC NET Quiz Generator specifically for Computer Science and Applications preparation. The system uses Gemini AI to generate multiple-choice questions based on the topic selected by the student. Each generated question contains four options, a correct answer, and an explanation.

The web application is developed using Python and Flask. MySQL is used to store student information, subjects, topics, quizzes, questions, and answers. The generated quiz is presented through an interactive web interface. After the student submits the quiz, the Flask backend compares the submitted answers with the stored correct answers and calculates the score and percentage.

The system also maintains quiz history so that students can review their previous attempts. A topic-wise leaderboard is provided to display student rankings based on their performance. The overall system therefore combines Generative AI, web technologies, and database management to provide an automated platform for UGC NET Computer Science practice.

2. Related Work

Recent research in educational artificial intelligence has explored automatic question generation, multiple-choice question generation, large language models, and the evaluation of AI-generated assessment items [1]-[5]. These studies demonstrate the potential of Generative AI to reduce the manual effort involved in creating educational questions while also highlighting the need for question-quality evaluation. Table 1 summarises selected studies relevant to the proposed AI-Based UGC NET Quiz Generator.

Study	Approach	Main contribution / limitation
Hadifar et al. (2023) [1]	EduQG dataset and educational question generation	Developed a dataset for multiple-choice question generation in education. It mainly focuses on dataset creation.
Ling and Afzaal (2024) [2]	Pre-trained large language models	Generated question-answer pairs for higher education using language models. The generated questions require evaluation.
Grévisse et al. (2024) [3]	LLM-generated MCQs	Studied AI-generated MCQs in Computer Science and Medicine. Some questions may have quality issues and need checking.
Hang et al. (2024) [4]	LLM-based MCQ generation	Proposed MCQGen for automated MCQ generation for personalized learning. It demonstrates the use of LLMs for educational assessment.
Maity et al. (2025) [5]	LLM-based question generation	Evaluated LLMs for generating educational questions. The study highlights the importance of prompting and question evaluation.

Table 1. Summary of related work on AI-based question generation.

However, other studies have also explored large language models for generating multiple-choice questions in computing courses [6], educational question classification and generation [7], and the evaluation of ChatGPT-generated questions [8]. Research has also examined educator perspectives on MCQ generation using large language models [9] and the use of prompting techniques for automatic question generation [10]. The proposed system focuses specifically on UGC NET Computer Science and Applications preparation and integrates AI-based question generation with a complete quiz workflow.

3. Problem Statement and Objectives

Problem Statement

UGC NET Computer Science students need regular practice with multiple-choice questions. Traditional quiz systems mainly depend on fixed question banks, which provide limited question variety and require manual question creation. Therefore, there is a need for a system that can automatically generate topic-based questions and provide immediate performance evaluation.

Objectives

The main objectives of the proposed system are:

- To generate UGC NET Computer Science MCQs automatically using Gemini AI.
- To provide topic-based quiz practice for students.
- To automatically evaluate answers and calculate scores.
- To store quiz history for reviewing previous attempts.
- To provide a topic-wise leaderboard for comparing student performance.
- To reduce the manual effort required for creating quiz questions.

4. Proposed System

The proposed system is an AI-Based UGC NET Quiz Generator developed using Python and Flask to support students preparing for the UGC NET Computer Science and Applications examination. The system allows students to select a Computer Science topic and generate multiple-choice questions (MCQs) using Gemini AI. Each generated question contains four options, the correct answer, and an explanation to help students understand the concept. Students can attempt the quiz through the web interface, and the system evaluates their selected answers by comparing them with the stored correct answers. Based on the evaluation, the application calculates the score and displays the quiz results. MySQL is used to store user details, topics, questions, and quiz attempt information. The system also maintains quiz history and provides a topic-wise leaderboard to compare student performance. The

application integrates AI-based question generation with quiz evaluation and performance tracking in a single platform. This reduces the manual effort required to prepare practice questions and provides students with an interactive way to practise Computer Science topics. The overall architecture of the proposed system is shown in Figure 1.

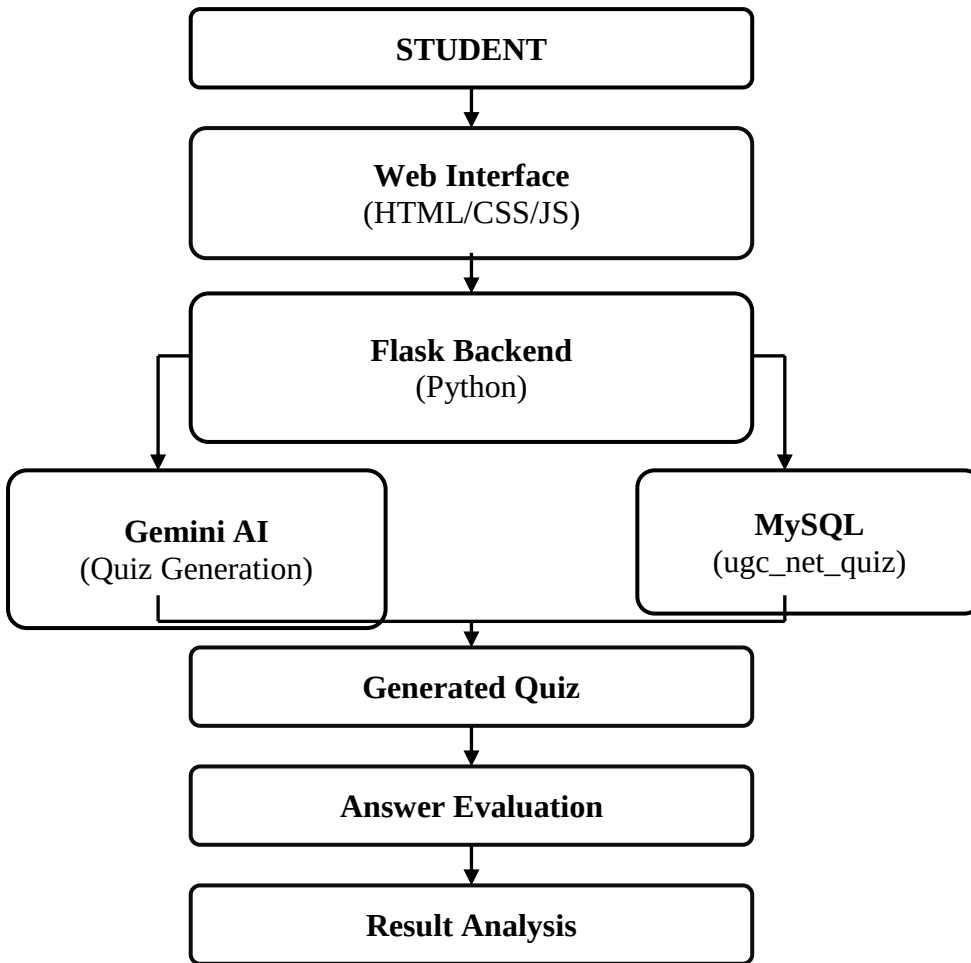


Figure 1. Architecture of the proposed AI-Based UGC NET Quiz Generator.

4.1 System Modules

Module	Function
User Management	Handles registration, login and logout.
Computer Science Subject and Topic Management	Displays the Computer Science subject and available topics.
AI-Based Quiz Generation	Generates MCQs using Gemini AI
Quiz Management	Displays questions and accepts student answers.
Answer Evaluation	Compares submitted answers with correct answers.
Result Analysis	Calculates score and percentage.
Quiz History	Stores and displays previous quiz attempts
Leaderboard	Displays topic-wise student rankings

Table 2. Functional modules of the proposed system.

5. Methodology

5.1 User Registration and Authentication

The student first creates an account by providing the required registration information. The system validates the submitted details and stores the user information in the MySQL database.

During login, the entered credentials are checked against the stored user information. If the credentials are valid, the student is redirected to the dashboard. Authentication prevents unauthorized access to student-specific quiz information.

5.2 Computer Science Subject and Topic Selection

The current implementation provides Computer Science as the fixed subject. The student does not select from multiple subjects.

The system provides different Computer Science topics for practice. Examples include:

- Programming in C
- Computer Networks
- Operating Systems
- Database Management Systems
- Artificial Intelligence

The student selects the required topic before generating a quiz.

5.3 AI-Based Question Generation

After the student selects a topic, the Flask backend sends a structured request to Gemini AI.

The request contains information about the selected topic and the required question format. Gemini AI generates multiple-choice questions containing:

- Question
- Option A
- Option B
- Option C
- Option D
- Correct answer
- Explanation

The generated response is processed by the Flask application and converted into quiz data that can be stored and displayed by the system.

The main advantage of this approach is that questions do not have to be manually entered into the database before every quiz.

5.4 Quiz Management

The generated questions are presented to the student through an interactive web interface.

For every question, four options are displayed. The student selects one option and proceeds through the quiz. After answering the available questions, the student submits the quiz.

The Flask backend receives the submitted answers and processes them for evaluation.

5.5 Answer Evaluation

The system automatically evaluates the submitted answers.

For each question, the selected answer is compared with the correct answer stored with the generated question.

The basic evaluation process is:

Student Answer → Compare with Correct Answer → Correct/Wrong → Calculate Score

Correct answers contribute to the student's score, while incorrect answers do not receive marks according to the implemented evaluation logic.

The evaluation is performed by the application backend rather than by Gemini AI.

5.6 Result Analysis

After the quiz is submitted, the system calculates the student's total score and percentage.

The result page displays the student's performance and provides information about the quiz attempt. The application also uses Chart.js for visual representation of result information.

The result analysis helps students understand their performance after completing a quiz.

5.7 Quiz History

The system stores completed quiz attempts in the MySQL database.

The quiz history allows students to review their previous attempts and performance. This provides a record of the student's practice activity and allows progress to be monitored over multiple quizzes.

5.8 Topic-Wise Leaderboard

The proposed system includes a topic-wise leaderboard.

The leaderboard ranks students according to their performance in quizzes associated with a particular Computer Science topic. Students can therefore view their position compared with other students who have attempted quizzes for the same topic.

6. Implementation

Component	Technology Used
Programming Language	Python
Web Framework	Flask
Frontend	HTML5, CSS3, JavaScript
Template Engine	Jinja2
AI Service	Gemini AI
Database	MySQL
ORM	SQLAlchemy
Charts	Chart.js
Development Environment	Visual Studio Code

Table 3. Software and technology components.

The system follows a client-server architecture where the frontend connects with the Flask backend. Flask manages the application logic, Gemini AI generates quiz questions, and MySQL stores the application data. Authentication, quiz management, answer evaluation, results, history, and leaderboard are handled by the Flask backend.

7. Experimental Evaluation and Results

The proposed system was evaluated to verify its functionality and the quality of the AI-generated questions. The major system functions, including topic selection, quiz generation, quiz submission, answer evaluation, result calculation, quiz history, and leaderboard were tested successfully. In addition, 20 AI-generated questions from five Computer Science topics were evaluated for correctness, topic relevance, clarity, option quality, and explanation quality.

7.1 Test Conditions

The system was tested using the Flask and MySQL environment. Five Computer Science topics were selected for AI-based question generation. The topics were Database Management Systems, Computer Networks, Programming in C, Operating Systems, and Artificial Intelligence. The first four AI-generated questions from each topic were selected, giving a total of 20 questions. The system functions and generated questions were evaluated for correctness, relevance, clarity, option quality, and explanation quality. The questions, answers, options, and explanations were checked for quality.

7.2 Evaluation Procedure:

The 20 AI-generated questions were manually reviewed by the author. Four questions from each of the five selected topics were used for evaluation. The questions were checked for correctness, topic relevance, clarity, option quality, and explanation quality. Each question was assessed separately for every criterion and marked as Satisfactory or Not Satisfactory.

The major application functions, including quiz submission, automatic answer evaluation, score calculation, quiz history, and topic-wise leaderboard, were also verified using the implemented system.

Evaluation Criterion	Questions Tested	Satisfactory Questions	Percentage
Question Correctness	20	19	95%
Topic Relevance	20	20	100%
Clarity	20	19	95%
Option Quality	20	20	100%
Explanation Quality	20	19	95%
Overall	100	97	97%

Table 4. Evaluation of AI-Generated Questions.

7.3 Sample Output

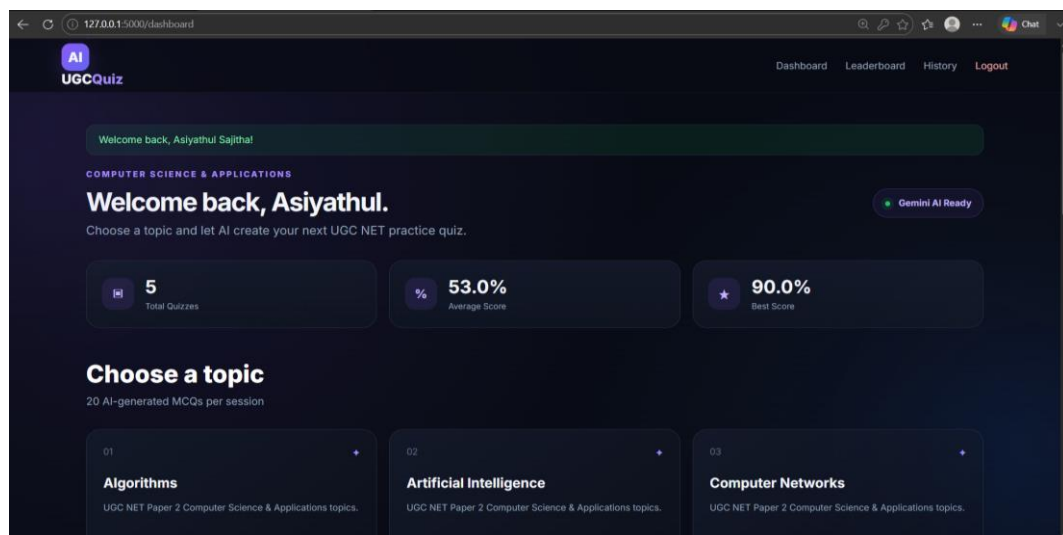


Figure 2. Computer Science Topic Selection and AI Quiz Generation.

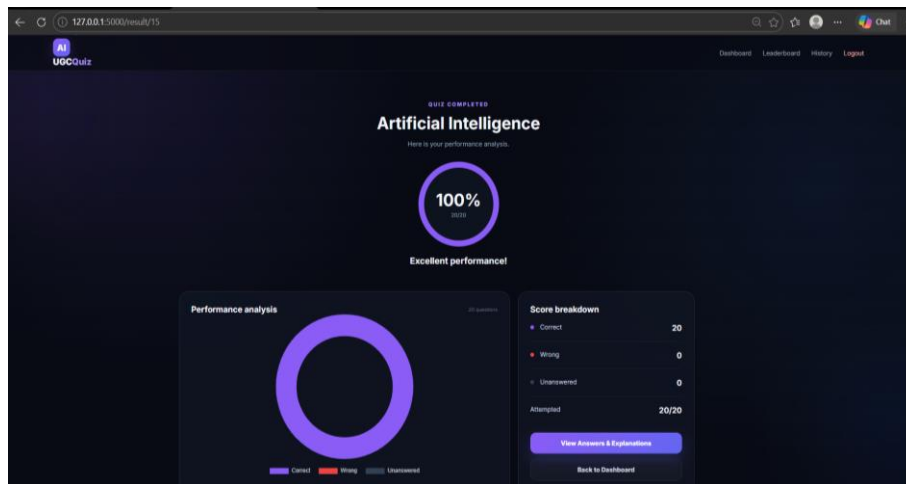


Figure 3. Quiz Result and Performance Analysis.

7.4 Evaluation Metrics

The quality of the AI-generated questions was evaluated using five criteria: correctness, topic relevance, clarity, option quality, and explanation quality.

The percentage for each criterion was calculated using:

$$\text{Percentage} = (\text{Satisfactory ratings} / \text{Total ratings}) \times 100$$

Across 20 questions evaluated against five criteria, the aggregate evaluation result was:

$$\text{Overall Percentage} = (97 / 100) \times 100 = 97\%$$

The functional performance of the system was also verified by testing the major application features. The tests confirmed that topic selection, AI quiz generation, quiz participation, automatic answer evaluation, result calculation, quiz history, and leaderboard functions operated successfully.

8. Discussion

The proposed system combines Generative AI with a web-based quiz application to provide topic-based UGC NET Computer Science practice. Gemini AI generates questions, while Flask and MySQL manage quiz processing, answer evaluation, results, history, and leaderboard functions. The system reduces manual question creation, but AI-generated questions may require verification for accuracy and quality.

9. Conclusion

This paper presented an AI-Based UGC NET Quiz Generator for Computer Science preparation. The system uses Gemini AI to generate topic-based MCQs and provides automatic answer evaluation, result analysis, quiz history, and a topic-wise leaderboard. The successful functional testing shows that the implemented system performs its intended operations effectively.

10. Future Work

The proposed system can be enhanced by adding more UGC NET subjects and topics and by introducing different difficulty levels such as easy, medium, and difficult. Future versions can include AI-based question validation, adaptive quizzes based on student performance, and previous-year UGC NET questions. The system can also provide personalized topic recommendations based on student performance. A mobile application and multilingual support can further improve accessibility and usability for students.

11. References

- [1] A. Hadifar, S. K. Bitew, J. Deleu, C. Develder, and T. Demeester, "EduQG: A multi-format multiple-choice dataset for the educational domain," *IEEE Access*, vol. 11, pp. 20885–20896, 2023, doi: 10.1109/ACCESS.2023.3248790.
- [2] J. Ling and M. Afzaal, "Automatic question-answer pairs generation using pre-trained large language models in higher education," *Computers and Education: Artificial Intelligence*, vol. 6, Art. no. 100252, 2024, doi: 10.1016/j.caeai.2024.100252.
- [3] C. Grévisse, M. A. S. Pavlou, and J. G. Schneider, "Docimological quality analysis of LLM-generated multiple choice questions in computer science and medicine," *SN Computer Science*, vol. 5, Art. no. 636, 2024, doi: 10.1007/s42979-024-02963-6.

-
- [4] C. N. Hang, C. W. Tan, and P.-D. Yu, "MCQGen: A large language model-driven MCQ generator for personalized learning," *IEEE Access*, vol. 12, pp. 102261–102273, 2024, doi: 10.1109/ACCESS.2024.3420709.
- [5] S. Maity, A. Deroy, and S. Sarkar, "Can large language models meet the challenge of generating school-level questions?" *Computers and Education: Artificial Intelligence*, vol. 8, Art. no. 100370, 2025, doi: 10.1016/j.caeai.2025.100370.
- [6] A. Tran, K. Angelikas, E. Rama, C. Okechukwu, D. H. Smith IV, and S. MacNeil, "Generating multiple choice questions for computing courses using large language models," in *2023 IEEE Frontiers in Education Conference (FIE)*, College Station, TX, USA, 2023, pp. 1–8, doi: 10.1109/FIE58773.2023.10342898.
- [7] S. Al Faraby, A. Romadhony, and Adiwijaya, "Analysis of LLMs for educational question classification and generation," *Computers and Education: Artificial Intelligence*, vol. 7, Art. no. 100298, 2024, doi: 10.1016/j.caeai.2024.100298.
- [8] S. Bhandari, Y. Liu, Y. Kwak, and Z. A. Pardos, "Evaluating the psychometric properties of ChatGPT-generated questions," *Computers and Education: Artificial Intelligence*, vol. 7, Art. no. 100284, 2024, doi: 10.1016/j.caeai.2024.100284.
- [9] G. Biancini, A. Ferrato, and C. Limongelli, "Multiple-choice question generation using large language models: Methodology and educator insights," in *Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization (UMAP '24)*, 2024, pp. 584–590, doi: 10.1145/3631700.3665233.
- [10] L. Wang, R. Song, W. Guo, and H. Yang, "Exploring prompt pattern for generative artificial intelligence in automatic question generation," *Interactive Learning Environments*, vol. 33, no. 3, pp. 2559–2584, 2025, doi: 10.1080/10494820.2024.2412082.