HBRP PUBLICATION

Journal of Advancement in Software Engineering and Testing
Volume 8 Issue 3 Sep-Dec 2025
e-ISSN: 2584-2226

AI Based Automated MCQ Generator

***Dr. Anitha S*[1]*, Balasurya G*[2]*, Dinu Vishruth N V*[3*]*, Jenish C*[4]*, Kathiresh A*[5]**
[1]*Associate Professor, Student (B.E Cse, Cyber Security), Depaetment of Computer Science & Engineering, Sri Shakthi Institute of Engineering and Technology, Coimbatore, India*

[*]***Corresponding Author***
E-Mail Id: - *dinuvishruth123@gmail.com*

ABSTRACT
This project aims to develop an AI-based Automated MCQ Generator that can dynamically generate multiple-choice questions from user-provided input such as text topics or uploaded study materials. The development of an AI-powered Automated MCQ Generator capable of dynamically creating multiple-choice questions based on user inputs, including uploaded study materials (PDF, DOCX, TXT files) or manually entered topics. Unlike traditional systems that depend on a pre-existing question database, our solution leverages the Gemini API to generate fresh, contextually relevant questions in real time. The system is built with a lightweight and interactive frontend (HTML, CSS, JavaScript) and a Flask-based backend that handles file processing, topic input, and API integration. All MCQs are generated on demand, eliminating the need for static storage. In-memory session management ensures efficient handling of quiz sessions, without reliance on external databases.By automating the MCQ creation process, the project significantly reduces the manual effort required by educators and offers a scalable solution for personalized and adaptive assessments. This tool is especially suited for online education platforms, teachers, and learners seeking quick, intelligent quiz generation tailored to specific learning materials or topics.

Keywords*:-Gemini API Key, Dynamic Quiz Creation, Automatic Question Generation, PDF/DOCX/TXT Text Extraction, User-Customizable Difficulty, Smart Learning Tools .*

1. INTRODUCTION

In the rapidly evolving landscape of education and e-learning, automation and artificial intelligence (AI) play a pivotal role in enhancing learning efficiency and accessibility. This project presents an AI-based MCQ (Multiple Choice Questions) Generator that dynamically creates quiz questions based on user-inputted topics or uploaded study materials in formats such as PDF, DOCX, or TXT. Leveraging Google's Gemini API and Flask as the web framework, the system utilizes advanced natural language processing to extract content, generate contextually relevant questions, and present them in an interactive quiz interface. Users can customize the number of questions, difficulty level, and set a timer, making the tool versatile for both educators and learners. The traditional method of creating MCQs is often a labor-intensive process, requiring subject matter expertise, careful curation, and significant time investment. Static databases of questions, while useful, may not adapt well to the dynamic needs of different learners or specific content areas.

The system architecture is designed for simplicity, usability, and performance. The frontend, built with HTML, CSS, and JavaScript, offers an intuitive interface where users can either upload study material files or directly enter a topic. On the backend, Flask handles file processing, text extraction, API communication with Gemini, session management, and rendering of quiz interfaces. The use of in-memory session storage allows for quick deployment without the overhead of database management, making the system lightweight yet powerful enough to serve both small-scale classroom environments and larger online learning platforms.

At the core of the system lies the integration with Google's Gemini API, which provides the natural language understanding capabilities necessary to generate coherent and relevant MCQs. Once the text is extracted from the user input, the system formulates prompts that guide the AI model to produce multiple-choice questions tailored to the specified difficulty level. The generated quizzes maintain a standardized format, ensuring consistency in the assessment process and providing immediate usability for educators and learners alike.

Ultimately, the project demonstrates how AI and web technologies can be combined to automate and personalize the quiz creation process, fostering a more adaptive and engaging educational experience. By empowering teachers to generate quizzes quickly and enabling learners to test their knowledge on customized content, this tool represents a step forward in making education more flexible, accessible, and aligned with modern digital learning needs.

In addition to its core functionalities, the system emphasizes user flexibility and accessibility. Users are not restricted to a single mode of input; they can choose either to upload comprehensive study materials or simply input a topic of interest, catering to a wide range of educational scenarios. The ability to set parameters such as the number of questions, desired difficulty level, and quiz timer further enhances the customization, allowing for tailored assessments based on the learner's needs.

Furthermore, the choice of lightweight technologies such as Flask ensures minimal server overhead, making the application easily deployable on local servers, educational institutions' intranets, or cloud platforms. By prioritizing simplicity in both user interaction and backend processing, the system lowers the technical barrier for adoption, enabling even those with limited technological expertise to benefit from AI-driven quiz generation. This combination of customization, efficiency, and ease of use distinguishes the project from more rigid, database-dependent quiz systems.

2. CONTENT

Journal of this model ensures that the AI with web development to build a dynamic and intelligent MCQ generator that simplifies the process of quiz creation for both students and educators. Below is a detailed breakdown of the system components, functionality, and workflows:

Table 1: *Table Caption.*

Focus	**S**
Api key generation	Generating mcq intelligently
Api key generation	Generating mcq intelligently
File , text inputs	Understanding the concepts
Dynamic user input	Adds flexibility and customization

1. **User Interface:** Developed using HTML, CSS, and a bit of JavaScript

to create a user-friendly and responsive web form.

2. **Backend Logic:** The backend is built using the Flask web framework, which handles routing, session management, and server-side logic.
3. **Quiz Interface and Timer:** Once MCQs are generated, the user is redirected to the quiz page.
4. **Result Evaluation:** The backend compares the user's selected answers with the correct answers. Each question is marked as correct or incorrect.
5. **Technologies used:** On the frontend, HTML, CSS, and JavaScript are used to create a responsive and user-friendly interface where users can upload files or input a topic, select the difficulty level, set the number of questions, and configure a timer.
6. **Key generation:** The execution will be in two forms. One is student login and another is staff login. After giving the input the session key will be generated and that key is used to generate mcqs.

The proposed system introduces an innovative solution to the common challenges faced by educators and learners in creating customized assessments. By integrating Google's Gemini API, the project leverages advanced natural language understanding to automatically generate MCQs that are not only relevant to the provided content but also adaptable based on difficulty and quantity preferences. This dynamic approach ensures that quizzes are tailored to individual or classroom needs, promoting a more engaging and effective learning environment. The seamless combination of file uploads and topic-based input makes the system flexible and inclusive, accommodating a variety of educational contexts.

The technological stack for this project was selected with a focus on simplicity, efficiency, and scalability. Flask, a lightweight web framework, handles all backend operations including file processing, API communication, and session management. The frontend, built using HTML, CSS, and JavaScript, ensures a clean and intuitive user experience. Users can easily navigate through the interface to upload materials or input topics, set parameters for quiz generation, and access the quizzes immediately. The architecture is designed to minimize latency and maximize user responsiveness, making it suitable for both academic institutions and independent learners.

3. PROBLEM STATEMENT

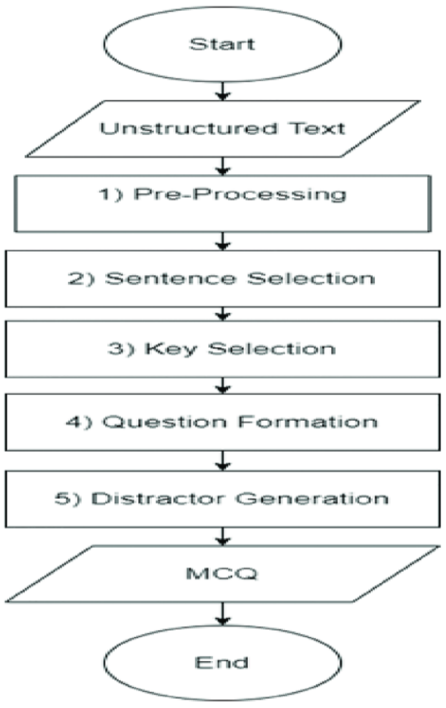
Traditional methods of creating assessments rely heavily on manually crafting multiple-choice questions (MCQs), which can be time-consuming, repetitive, and prone to bias. Static question banks often lack adaptability to different learning materials or real-time updates, making it difficult to cater to diverse learner needs. Furthermore, educators in online or large-scale learning environments face increasing pressure to generate assessments quickly without compromising quality. There is a clear need for an intelligent system that can dynamically create contextually relevant MCQs, saving time while maintaining assessment standards.

In the current educational landscape, the manual creation of assessments, particularly multiple-choice questions (MCQs), remains a labor-intensive and time-consuming process. Traditional systems often rely on static databases that require constant updating and fail to adapt dynamically to new learning materials or specific learner needs. As education increasingly moves toward digital and remote platforms, there is a growing demand for automated, intelligent solutions that can generate assessments quickly, accurately, and contextually. Educators struggle to create diverse, high-quality questions that align with different topics, difficulty levels, and learning outcomes. Moreover, existing systems typically lack flexibility, offering limited customization and often producing repetitive or outdated content. This project addresses these challenges by developing an AI-driven MCQ generator that utilizes the Gemini API to automatically create fresh, relevant, and customizable quiz questions from user-provided text or study materials, thereby reducing manual effort, improving assessment quality, and supporting a more personalized learning experience.

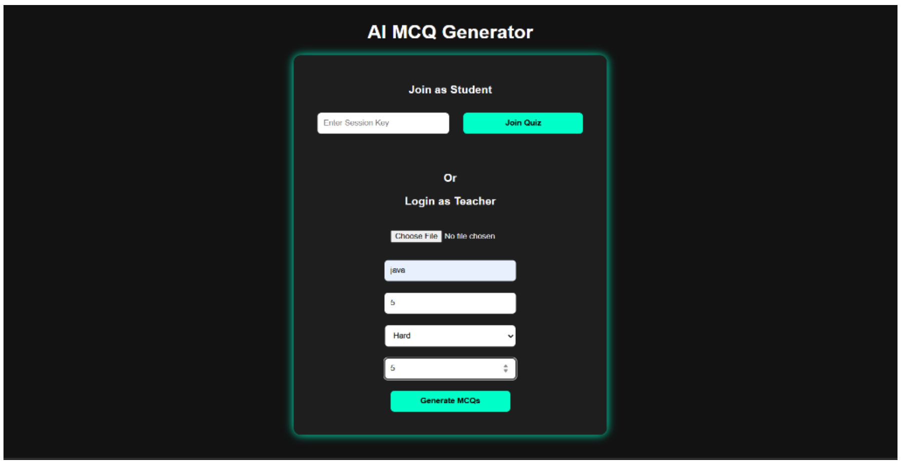
4. OBJECTIVE

In the current educational landscape, the manual creation of assessments, particularly multiple-choice questions (MCQs), remains a labor-intensive and time-consuming process. Traditional systems often rely on static databases that require constant updating and fail to adapt dynamically to new learning materials or specific learner needs. As education increasingly moves toward digital and remote platforms, there is a growing demand for automated, intelligent solutions that can generate assessments quickly, accurately, and contextually. Educators struggle to create diverse, high-quality questions that align with different topics, difficulty levels, and learning outcomes. Moreover, existing systems typically lack flexibility, offering limited customization and often producing repetitive or outdated content. This project addresses these challenges by developing an AI-driven MCQ generator that utilizes the Gemini API to automatically create fresh, relevant, and customizable quiz questions from user-provided text or study materials, thereby reducing manual effort, improving assessment quality, and supporting a more personalized learning experience.

FLOW DIAGRAM

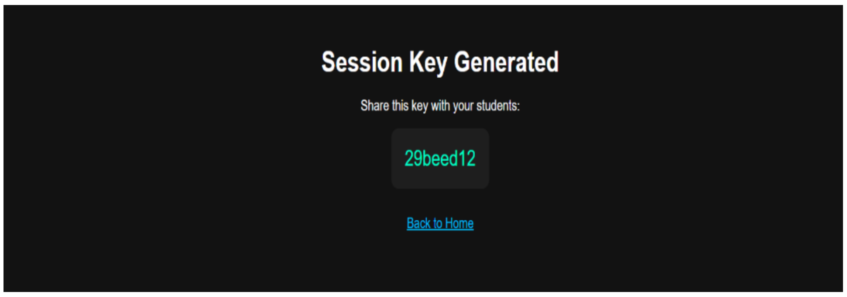


OUTPUT AND RESULT



java
5
Hard
5

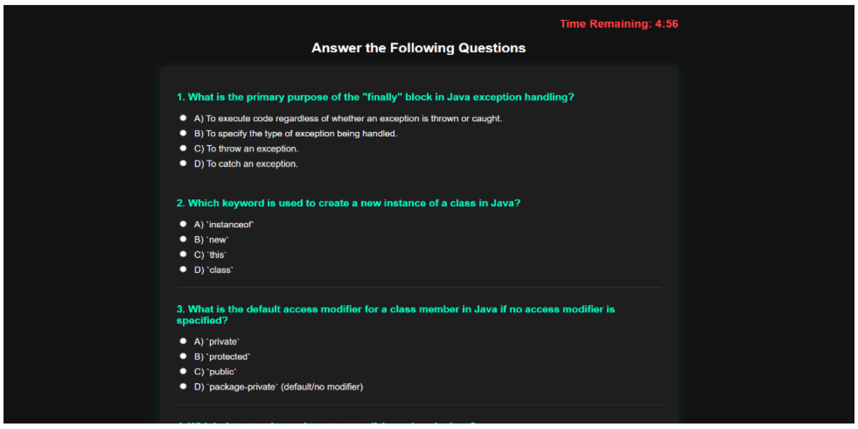
Fig.1:-The login page


Session Key Generated

Share this key with your students:

29beed12

Fig.2:-Session Key


Time Remaining: 4:56

Answer the Following Questions

1. What is the primary purpose of the "finally" block in Java exception handling?
   - A) To execute code regardless of whether an exception is thrown or caught.
   - B) To specify the type of exception being handled.
   - C) To throw an exception.
   - D) To catch an exception.

2. Which keyword is used to create a new instance of a class in Java?
   - A) `instanceof`
   - B) `new`
   - C) `this`
   - D) `class`

3. What is the default access modifier for a class member in Java if no access modifier is specified?
   - A) `private`
   - B) `protected`
   - C) `public`
   - D) `package-private` (default/no modifier)

Fig.3: Generated MCQs

5. REVIEW

Our project delivers an Automated MCQ Generator that streamlines quiz creation using AI. When a teacher uploads a file or enters a topic and clicks the "Generate" button, the system interacts with the Gemini API to create MCQs and instantly provides a unique session key. This key ensures secure access and separates each quiz session. Students then enter the session key into the provided box to retrieve and attempt the generated questions. The backend is built with Flask for fast, real-time communication, while Java handles the database management. The system smoothly manages file uploads

(PDF, DOCX, TXT), processes content, and dynamically generates quality MCQs without relying on a pre-existing database. It ensures a smooth experience for both teachers and students with minimal manual effort. Overall, the project showcases an innovative approach by blending AI-driven content generation with a session-based secure access model for interactive assessments.

The final stage of the MCQ Generator system where the user interacts with a dynamic quiz interface. This interface is generated after the user either uploads a document or inputs a topic, along with selecting difficulty, number of questions, and timer duration. The backend, powered by Google's Gemini AI model, processes the content and generates multiple-choice questions.

6. FUTURE WORKS

- **Enhancing Question Diversity** Improve the AI generation to create different types of MCQs like assertion-reason, match-the-following, true/false questions.
- **Difficulty Level Calibration** Implement better algorithms to accurately measure and adjust question difficulty based on student performance data.
- **Real-time Performance Analytics** Provide teachers with dashboards showing student scores, accuracy rates, and topic-wise strengths/weaknesses.
- **AI-based Distractor Generation** Improve MCQ quality by automatically generating more challenging wrong options (distractors) using advanced AI prompts.
- **Mobile App Development** Build a mobile version of the platform for Android/iOS to make quiz participation even more accessible.

7. CONCLUSION

This project aligns with the objectives of the Journal of Recent Trends in Computer Graphics and Multimedia Technology by showcasing a modern web-based system that utilizes interactive multimedia elements like dynamic forms, timers, and responsive quiz interfaces. The inclusion of AI-generated questions introduces an innovative layer to multimedia learning, enhancing user engagement and content delivery. Furthermore, the project fits within the scope of the Advanced Innovations in Computer Programming Languages journal through its use of Flask for backend development, API integration for AI-based question generation, and structured handling of dynamic content. The combination of programming logic, real-time processing, and educational technology reflects a blend of modern software development practices with intelligent automation, making it a strong candidate for research and future innovation in both multimedia and programming domains.

The proposed project demonstrates a practical application of modern programming concepts and multimedia technologies, in line with the vision of both the Journal of Recent Trends in Computer Graphics and Multimedia Technology and the Advanced Innovations in Computer Programming Languages. By integrating artificial intelligence for automatic question generation, the system enhances traditional learning platforms with smart interactivity and adaptive content. The use of Flask for backend logic, combined with Gemini API and HTML/CSS for a responsive user interface, bridges the gap between intelligent content generation and intuitive user experience. This blend of automation, educational support, and seamless UI design not only encourages dynamic learning environments but also paves the way for innovative approaches in both programming and multimedia-driven technologies.

In conclusion, this project represents a significant step toward the intelligent automation of educational content using cutting-edge technologies. By leveraging artificial intelligence through the Gemini API, along with a user-friendly interface developed using Flask and frontend technologies, the system successfully generates customized multiple-choice questions based on user-provided topics or content. It bridges the gap between modern programming techniques and practical learning tools, aligning with current trends in computer science and multimedia innovation. This solution not only enhances personalized learning but also sets the foundation for future advancements in AI-driven education platforms.

REFERENCES

1. Mannem Prashanth, Rashmi Prasad, and Aravind Joshi. (2010). Question generation from paragraphs at UPenn: QGSTEC system description. In Proceedings of QG2010: The Third Workshop on Question Generation:84–91.
2. Colleen E Crangle and Joyce Brothers Kart. 2015). A questions-based investigation of consumer mental Health information. PeerJ, 3:e867.
3. Gall MD. (1970). The use of questions in teaching. *Review of Educational Research, 40*: 707 720.
4. Swart (2010). Evaluation of Final Examination Papers in Engineering: A Case Study Using Bloom's Taxonomy. *IEEE Transactions on Education.*53(2):257-264.
5. Aldabe, I., De Lacalle, M. L., Maritxalar, M., Martinez, E., and Uria, L. (2006). Arikiturri: an Automatic question generator based on corpora and nlp techniques. *In International Conference on Intelligent Tutoring Systems*. Springer, Berlin, Heidelberg. International Journal on Natural Language Computing (IJNLC)10(2): 584-594
6. Folajimi, Y. O., and Omojola, O. E. (2013). Natural language processing techniques for automatic test Questions generation using discourse connectives. *Journal of Computer Science and Its Application, 20*(2), 6076.
7. Aditya S. (2018). Using Natural Language Processing for Smart Question Generation. An Intel Developer Program Documentation. https://software.intel.com/en-us/articles/using- natural language processing forsmart-question-generation.
8. Susanti, Y., Tokunaga, T., Nishikawa, H., and Obari, H. 2018. Automatic distractor generation for multiple Choice English vocabulary questions. *Research and Practice in Technology Enhanced Learning, 13*(1):15.
9. Das, B., and Majumder, M. (2017). Factual open cloze question generation for assessment of learne's knowledge. *International Journal of Educational Technology in Higher Education, 14*(1), 24.