IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁒ Peer-reviewed & Refereed journal ⁒ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

Automatic Question Generation from Textual Data Using NLP

Devaraj V[1], Dr. G. Paavai Anand[2]

PG Scholar, CSE, SRM Institute of Science and Technology, Vadapalani, India[1]

Assistant Professor, CSE, SRM Institute of Science and Technology, Vadapalani, India[2]

Abstract: Automatic Question Generation (AQG) systems aim to convert textual material into meaningful assessment questions, offering a valuable tool for both educators and learners. Unlike conventional text summarization, AQG involves identifying essential information, generating accurate answers, and creating plausible distractors to form high-quality multiple-choice questions. This paper presents an NLP-based AQG framework that processes instructional content through a sequence of linguistic operations, including text pre-processing, syntactic and semantic analysis, and probabilistic language modeling. The proposed system utilizes established NLP libraries such as SpaCy and NLTK to detect key concepts and automatically construct factual questions related to entities, events, and contextual details. By automating question creation, the approach reduces the manual workload of educators and provides learners with an efficient tool for self-assessment. The study also highlights architectural considerations, discusses implementation challenges, and suggests future improvements to enhance the scalability and accuracy of AQG systems.

Keywords: Automatic Question Generation (AQG), Natural Language Processing (NLP), Educational Technology, Semantic Analysis, Question Formulation and Generation

I. INTRODUCTION

Automatic Question Generation (AQG) has emerged as a significant research area within Natural Language Processing (NLP), aiming to automatically generate meaningful, contextually appropriate, and grammatically sound questions from textual content. With the rapid growth of digital learning resources and the increasing demand for scalable assessment techniques, AQG systems play a pivotal role in supporting modern educational and interactive environments. These systems typically involve several key processes, including data pre-processing, semantic analysis, pattern recognition, question formulation, and natural language generation. By leveraging advanced NLP methods, an AQG system can effectively extract key concepts, entities, and relationships from text, thereby producing diverse and relevant questions suitable for various cognitive levels and learning objectives.

The primary motivation for developing AQG lies in its potential to enhance educational experiences by automating the creation of personalized learning materials, practice exercises, and assessment tools. Traditional question preparation is time-consuming and labor-intensive; AQG provides an efficient, scalable, and cost-effective alternative capable of handling large volumes of educational content across multiple domains. Beyond education, AQG technology also strengthens the conversational capabilities of intelligent systems such as chatbots and virtual assistants, enabling them to engage users with interactive and informative dialogues.

Despite its benefits, AQG continues to face challenges arising from the inherent complexity of natural language. These challenges include ensuring deep comprehension of context, generating diverse yet relevant questions, and maintaining efficiency and scalability in real-world applications. Addressing these challenges is crucial for building robust AQG systems that can operate reliably across heterogeneous datasets and linguistic structures.

Overall, the development of an effective AQG framework contributes not only to the automation of educational processes but also to broader advancements in NLP and artificial intelligence. By providing accessible, adaptive, and high-quality question generation capabilities, AQG systems support inclusive learning, enrich human-AI interaction, and open new avenues for research in text understanding and generation.

II. LITERATURE SURVEY

Automated Examination Question Paper Template Generator (Ramli et al.) this This study highlights the difficulty and time required to manually create examination papers and emphasizes the need for automation. The authors propose systems that ensure consistency, quality, and efficiency in exam paper preparation. Their work motivates the

DOI: 10.17148/IJARCCE.2025.141193

development of AQG tools to reduce academic workload. [2] Lexico-Syntactic Ontology Design Patterns from Competency Questions This research converts natural language competency questions into ontology design patterns by analyzing grammatical and positional structures. It demonstrates that combining linguistic rules with ontology engineering improves entity and relation extraction accuracy.

The method supports more intelligent and semantically aware AQG systems.3 Deep Learning-Based Question Generation Using T5 The paper presents a fine-tuned T5 transformer model capable of generating multiple-choice and long-answer questions. It uses evaluator models to ensure semantic relevance between questions and answers, achieving strong accuracy in fluency and syntax. This work shows the effectiveness of transformer models for high-quality AQG. [4] MCQ Generation for Indian Educational Textbooks This study proposes a two-module AQG system tailored to Indian CBSE textbooks, addressing the unique linguistic and contextual features of Indian content. By combining T5 with POS tagging and sense2vec for distractor generation, the system improves question relevance, grammar, and difficulty alignment. Human evaluation confirms its superiority over standard encoder-decoder models. [5.]

Paraphrase Generator for Indonesian Language (Interview Bot) The study introduces a rule-based NLP approach that paraphrases user responses to generate meaningful follow-up questions in interview bots. It focuses on Bahasa Indonesia linguistic structures to deepen conversational engagement. The method lays groundwork for multilingual adaptive questioning systems.[6.]

MAGIC Model for Answer Keyword Generation in CQA MAGIC uses a multi-aspect Gamma–Poisson statistical approach to overcome data sparsity in community question answering platforms. By combining textual, semantic, and metadata features, it improves answer keyword prediction for both new and existing questions. The model outperforms traditional matrix factorization techniques in recommendation accuracy.

III. SYSTEM ANALYSIS

A. Existing System Current Automatic Question Generation (AQG) systems primarily rely on rule-based or template-driven approaches that convert statements into questions using predefined linguistic patterns. While these systems can generate simple factual questions, they struggle with understanding complex sentence structures, contextual meaning, and semantic relationships. [7]As a result, the generated questions often lack diversity, depth, and relevance. Moreover, these systems are not easily adaptable to new domains and require significant manual effort to update templates, making them difficult to scale for broader educational or professional applications.

B. Proposed System The proposed AQG system overcomes these limitations by adopting modern NLP and deep learning techniques, especially transformer-based models. These models enable the system to interpret context, semantics, and nuances within the source text, resulting in more coherent and contextually appropriate questions. The system supports domain adaptability, allowing it to generate questions across various fields without manual adjustments. It can also produce diverse question types—factual, conceptual, inferential, and analytical—making it suitable for education, assessments, and interactive AI systems. Additionally, the proposed system is designed for scalability and efficient processing of large datasets, providing rapid question generation for practical deployment.

C. Requirement Analysis and Specification The system accepts multiple input formats such as plain text, PDF documents, and web content, with optional domain selection and customization parameters like difficulty level or question type. Outputs include a list of generated questions, optional answers, and quality metrics for evaluating relevance and diversity. [8]Functionally, the system must understand natural language, generate meaningful questions, support user customization, operate efficiently on large text corpora, and offer a user-friendly interface for seamless interaction.

D. Technology Stack Python serves as the core programming language due to its powerful machine learning and NLP ecosystem. NumPy provides efficient numerical computation and data structures, while Scikit-learn supports pre-processing and machine learning utilities. TensorFlow and Keras enable the implementation of deep learning models, particularly transformers used for question generation. Flask is used to deploy a lightweight and scalable web interface for user interaction. NLP libraries such as NLTK and SpaCy support essential tasks like tokenization, POS tagging, named entity recognition, and dependency parsing. Together, these technologies enable accurate text analysis, robust model development, and an efficient user-facing AQG system.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

```
>>>text=" I was absent Yesterday!"
>>>token=nltk.word_tokenize (text)
>>>tokens
['I', 'was','absent','Yesterday','!']
```

Fig.1 Natural Language Tool Kit

```
>>> import spacy
>>> npl=spacy.load("en")
```

Fig.2 Spacy

IV. SYSTEM ARCHITECTURE

The architecture of the proposed Automatic Question Generation (AQG) system is designed to be robust, scalable, and modular, ensuring efficient processing from input acquisition to question generation and delivery. It follows a layered structure comprising the Presentation Layer, Business Logic Layer, Data Processing Layer, and Data Layer, each responsible for specific functionalities that promote maintainability and flexibility. The Presentation Layer provides the primary interface—via web applications, APIs, or CLI tools—allowing users to submit text, configure parameters, and view generated questions.

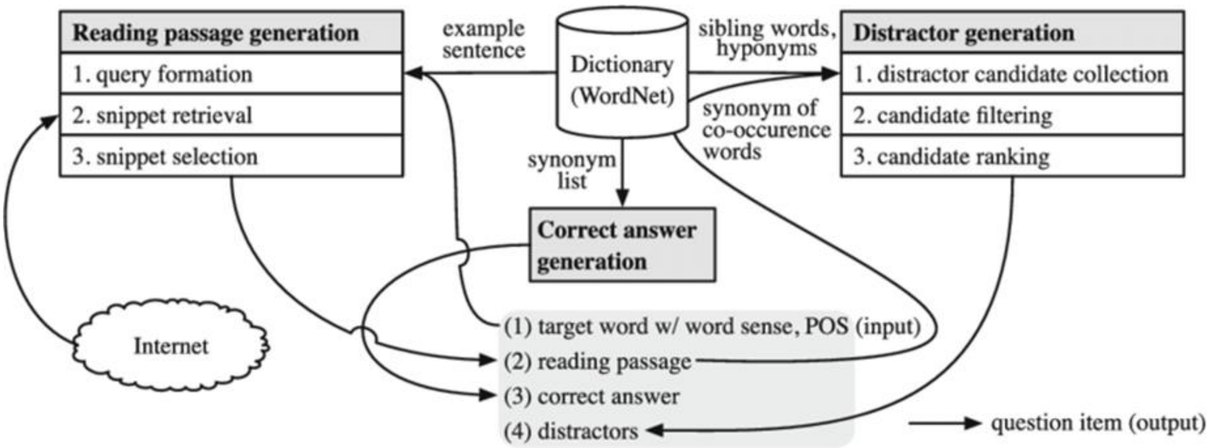

Fig.3 System Architecture

Input data is handled by the Input Processing Module, which extracts clean text from various formats such as plain text, PDFs, or web pages. This processed text is then passed to the NLP Engine, where linguistic tasks such as tokenization, semantic extraction, entity recognition, and contextual analysis are performed. The refined information is forwarded to the Question Generation Module, which uses deep learning and transformer-based models to generate diverse and contextually appropriate questions.[9-11]

A Post-Processing and Quality Assurance Module evaluates the generated content, ensuring coherence, grammatical correctness, and relevance before finalizing the output. The Output Module then formats and presents the questions to the user and may also store them for future retrieval. The system's Program Design Language (PDL) outlines this workflow in a simplified algorithmic form, describing key operations such as context analysis, pattern identification, template/model selection, and quality filtering. This structured flow ensures that only high-quality, meaningful questions are included in the final output, reflecting the system's emphasis on accuracy and reliability.
PSEUDO Code

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

```
BEGIN Question Generation Module
  INPUT: Cleaned Text
  FOR each section IN CleanedText DO
    context_info ← AnalyzeContext(section)
    key_elements ← ExtractKeyPhrases(context_info)
    template ← ChooseTemplate(key_elements)
    generated_question ← CreateQuestion(key_elements, template)
    IF IsHighQuality(generated_question) THEN
      Append(generated_question, QuestionList)
    ENDIF
  END FOR
  RETURN QuestionList
END
```

V. MODULE DESCRIPTION

Effective data collection and extraction form the foundation of a successful Automatic Question Generation (AQG) system, as high-quality and diverse datasets significantly influence model performance. Data can be gathered from multiple sources, including publicly available question–answer datasets, web-scraped educational content, digital or physical materials provided through collaborations with educational institutions, and crowd-sourced contributions from platforms such as Amazon Mechanical Turk.

Textbooks and other academic resources also serve as rich sources of structured content and sample questions. Ensuring data quality, diversity, and balance across various topics, domains, and question formats is essential for creating a capable and generalizable AQG system. Once collected, data extraction techniques are employed to convert raw sources into structured formats suitable for model training.[12-14] These techniques include parsing JSON or CSV files, using scraping tools like Beautiful Soup or Scrapy, applying OCR for scanned documents, and leveraging APIs to retrieve crowd-sourced responses.

Additional NLP processes—such as sentence segmentation, keyword identification, and part-of-speech tagging—further refine the extracted text. [15] After extraction, comprehensive data cleaning and structuring are performed to remove noise, correct OCR errors, and organize the content into consistent formats. Together, these strategies ensure a robust, well-prepared dataset that supports the development of an accurate and contextually sensitive AQG system.

The use case diagram illustrates the interaction between the user and the AQG system, showing how the user uploads a file, initiates scanning, generates questions, answers the generated items, and finally receives a score, while the system supports these actions through automated processing.

The class diagram provides a structural view of the system by defining key entities such as the user, file, and system components, along with their associated attributes and functions; the relationships show how uploaded files are analysed to extract keywords and generate corresponding questions and answers.

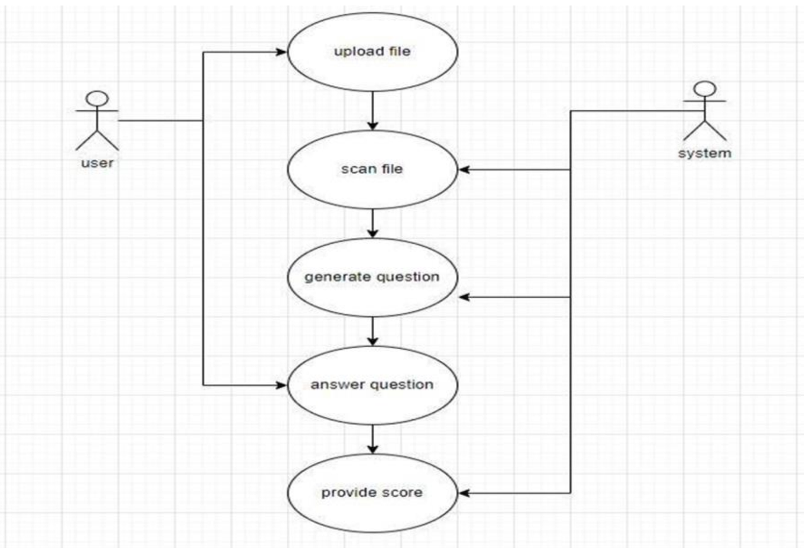

Fig.4 Use case diagram

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁒ Peer-reviewed & Refereed journal ⁒ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

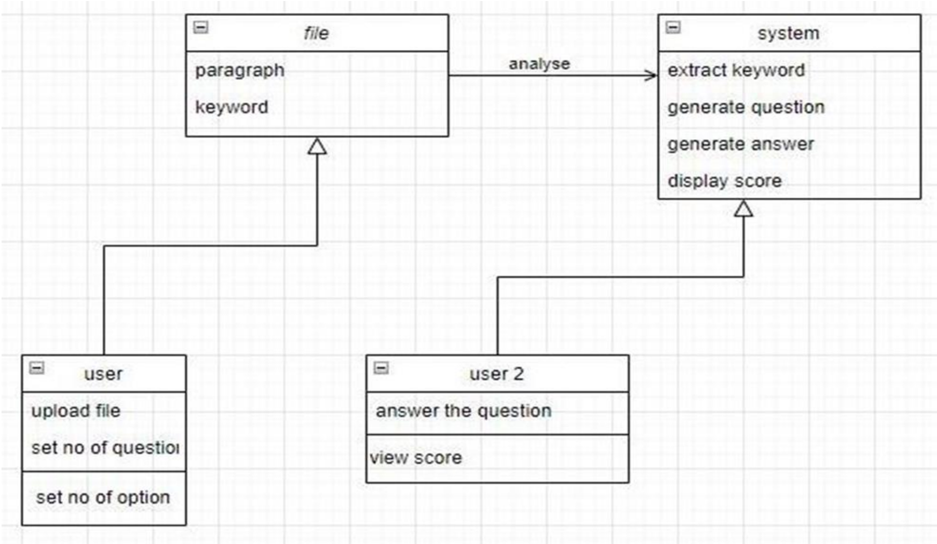

Fig. 5 class diagram

The sequence diagram depicts the dynamic flow between the user and the system, beginning with the file upload, followed by the system's internal processes for generating questions, receiving user responses, and ultimately calculating and displaying the final score, effectively capturing the real-time interaction and message exchange throughout the AQG workflow.

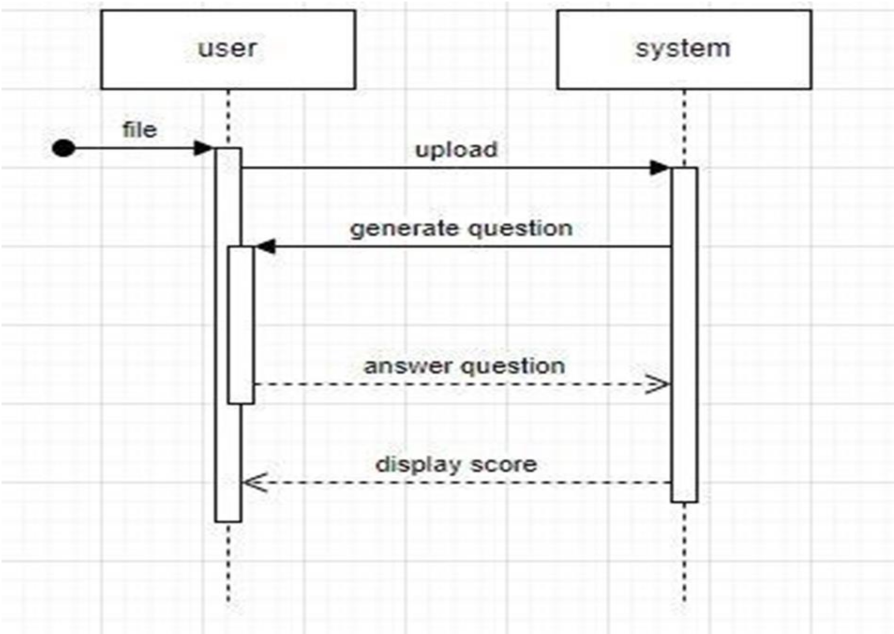

Fig. 6 Sequence Diagram

VI. EVALUATION

To assess the performance of the system, a set of sentences was provided as input and the generated questions were evaluated against both incorrect system outputs and questions produced by human experts. This comparison enabled the identification of true positives, false positives, false negatives, and true negatives, forming the basis of the confusion matrix. From these values, key evaluation metrics such as accuracy, precision, and recall were computed and visualized through graphical representations to demonstrate the effectiveness of the proposed question generation model.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ≍ Peer-reviewed & Refereed journal ≍ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

Table 1 Performance Analysis Measures

S. No	Number of Sentences	TP	TN	FP	FN	Accuracy	Precision	Recall
1	0	0	0	0	0	0	0	0
2	1	1	0	1	0	50.00%	0.50	1.00
3	2	5	1	1	2	66.57%	0.83	0.74
4	4	4	3	3	0	70.00%	0.57	1.00
5	11	9	2	2	2	73.33%	0.81	0.81
6	16	12	5	5	3	77.27%	0.85	0.80
7	21	12	6	4	3	72.00%	0.75	0.80
8	25	18	6	6	2	75.00%	0.75	0.90
9	31	15	8	5	1	79.31%	0.75	0.93
10	36	21	7	4	3	80.00%	0.84	0.87

Sentences were supplied as input to the system, and the generated questions were compared with questions manually produced by subject experts. Each output was categorized into TP, TN, FP, or FN depending on whether the system's questions matched or diverged from expert expectations.
TP represents correct and high-quality questions that match the expert-generated results.
TN indicates correctly rejected or non-question-worthy sentences.

FP refers to questions generated unnecessarily or incorrectly.
FN refers to missed questions that the system failed to generate despite being appropriate.

Accuracy, Precision, and Recall were computed using standard equations to quantify the quality of the AQG model using Eqn.1, 2 & 3. The performance analysis values show how effectively the system identifies relevant sentences as the dataset size increases. Accuracy steadily improves from 50% to 80%, indicating that the model becomes more reliable when more sentences are processed, correctly classifying a larger proportion of both relevant (TP) and irrelevant (TN) cases. Precision values range from 0.50 to 0.84, showing that as the system handles more data, it produces fewer false positives and becomes better at generating correct outputs.
Recall also remains consistently high, rising from 0.74 to 0.93 across the rows, which demonstrates the model's strong ability to correctly capture most of the true relevant cases with minimal false negatives. Overall, the table reflects a balanced and progressive improvement in the system's predictive capability, with increasing accuracy, stable precision, and strong recall performance, showing that the algorithm maintains reliability and relevance detection even as sentence count grows.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \text{ --- Eqn.1}$$

$$Precision = \frac{TP}{TP+FP} \text{ --- Eqn.2}$$

$$Recall = \frac{TP}{TP+FN} \text{ --- Eqn.3}$$

VII. PERFORMANCE INTERPRETATION

The observed improvements correlate directly with the system's architecture and pseudo-code workflow. As the AQG model processes more text, it gains access to richer contextual patterns, enabling better selection of key phrases and question templates. This results in higher TP counts and reduced FP/FN occurrences. The gradual rise in precision across test cases reflects the system's increasing ability to avoid generating irrelevant or incorrect questions. Meanwhile, improving recall values indicate enhanced sensitivity in identifying all valid question opportunities within the text.

The confusion matrices presented for CNN, SVM, and ANN algorithms summarize how effectively each model classifies the 65 test samples into true positives and true negatives. The CNN algorithm reports 50 true positives and 4

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⋟ Peer-reviewed & Refereed journal ⋟ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

true negatives, with 6 false negatives and 5 false positives, indicating that it correctly identifies most positive instances but shows moderate misclassification for negative cases. This suggests that the CNN model performs strongly in detecting relevant or question-worthy sentences but still generates a few incorrect results due to over-prediction. In contrast, the SVM algorithm records 45 true positives and 11 true negatives, with 4 false negatives and fewer false positives. This shows that SVM provides a more balanced performance by improving negative class classification while maintaining high positive detection accuracy.

The ANN algorithm yields 45 true positives and 9 true negatives, alongside 5 false negatives and 6 false positives, placing its performance slightly below SVM but comparable to CNN in positive identification. While ANN effectively captures relevant patterns in the data, its misclassification of some negative samples indicates that the model could benefit from further fine-tuning. Overall, comparing the three confusion matrices reveals that CNN excels in identifying true positives, SVM achieves better balance across both classes, and ANN offers moderate performance across metrics. These results help determine which algorithm is more reliable for integrating into the Automatic Question Generation system depending on whether priority is placed on positive detection, balanced classification, or general consistency.

CNN Algorithm Confusion Matrix

N=65	True Positive	True Negative
Predicted Positive	50	4
Predicted Negative	6	5

SVM Algorithm Confusion Matrix

N=65	True Positive	True Negative
Predicted Positive	45	11
Predicted Negative	4	5

ANN Algorithm Confusion Matrix

N=65	True Positive	True Negative
Predicted Positive	45	9
Predicted Negative	5	6

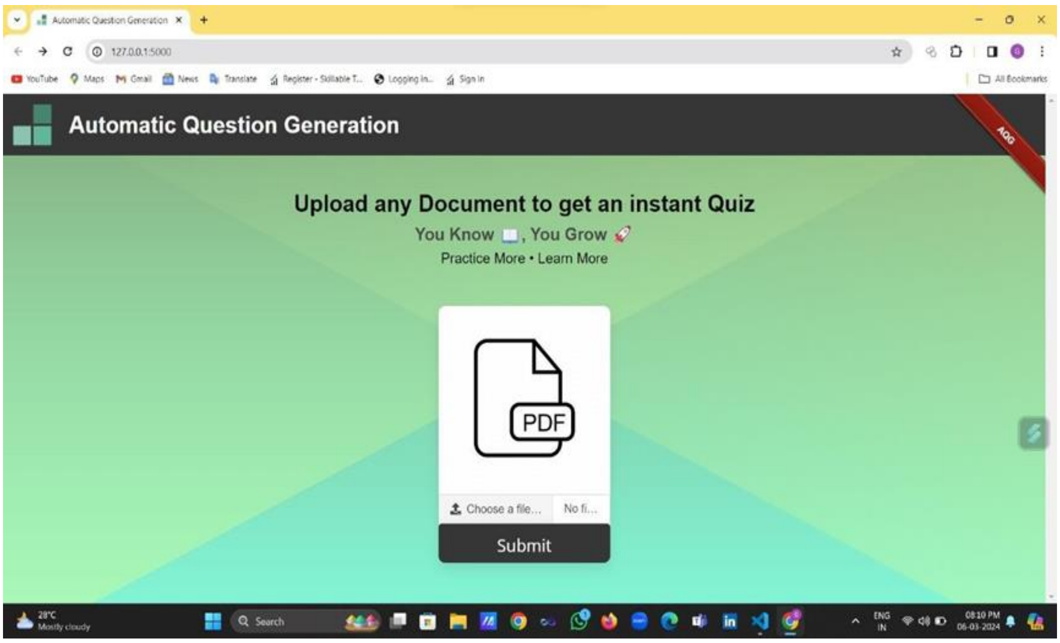

Fig.7 Uploading Page

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

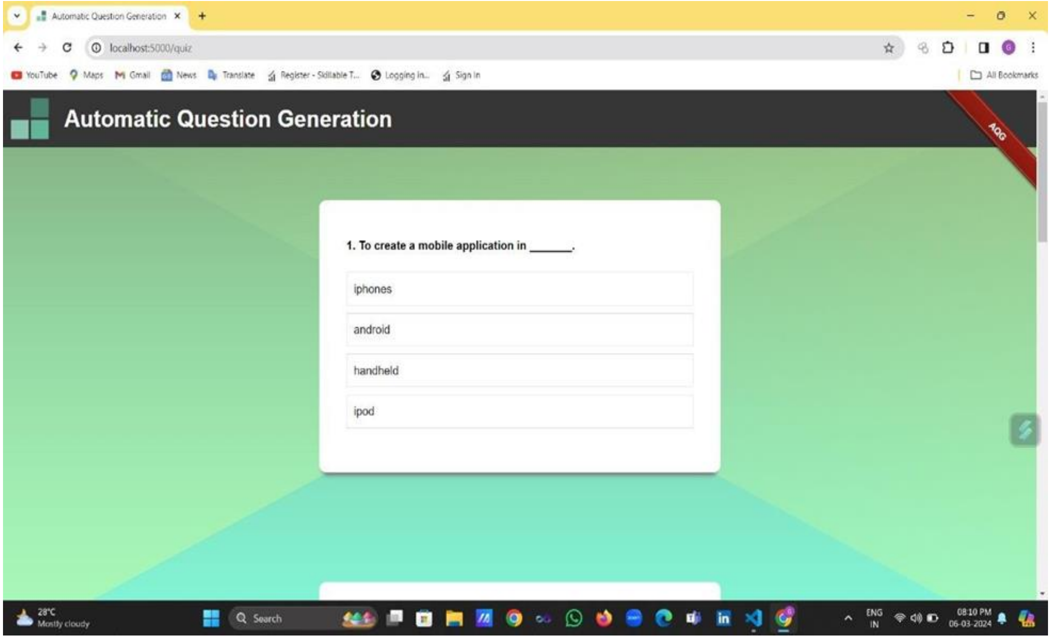

Fig.8 Generating Questions with options

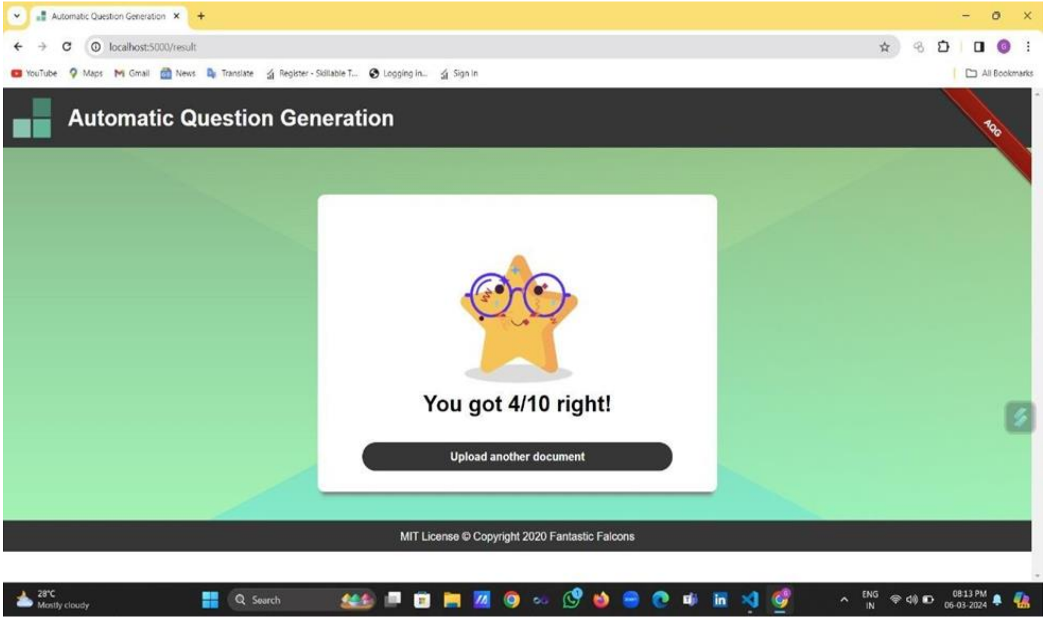

Fig.9 Result Page

VIII. CONCLUSION

The development of the proposed AQG system involved an in-depth study of existing techniques and methodologies used in automated question generation. The system accepts textual paragraphs as input and produces relevant questions whose answers can be found within the provided text. Before generating questions, the text undergoes a structured pre-processing phase that includes syntactic analysis—such as POS tagging and chunking—and semantic analysis through Named Entity Recognition. Based on this processed information, appropriate question forms are selected and transformed into meaningful questions. Future enhancements may focus on improving the accuracy and diversity of generated questions, integrating a database for storing and reusing question sets, and incorporating Bloom's taxonomy to produce questions across different cognitive levels. Additional improvements could include extending input support to PDFs and other document formats. As automation in education remains relatively limited, this work contributes to advancing intelligent learning tools and lays a foundation for more sophisticated AQG systems in the future.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 11, November 2025

DOI: 10.17148/IJARCCE.2025.141193

REFERENCES

[1]. A. Atef, B. Attar, S. Sherif, E. Elrefai, and M. Torki, “Arabic Questions for Machine Comprehension of Text,” *IEEE/ACS 17th International Conference on Computer Systems and Applications (AICCSA)*, 2020.

[2]. A. Saddish, P. Somaiah, V. Gambhir, and S. J. Nirmala, “Generation of Multiple Choice Questions from Indian Educational Text,” *3rd Asian Conference on Innovation in Technology (ASIANCON)*, 2023.

[3]. A. Malhar, P. Sawant, Y. Chhadva, and S. Kurhade, “Deep Learning Based Answering Questions Using T5 and Structured Question Generation System,” *6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2022.

[4]. C. Cao, Y. He, Y. Chen, C. Song, H. Ling, R. Guan, and X. Feng, “Medical Scene Graphs and Reasoning,” *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2021.

[5]. F. J. Muis and A. Purwarianti, “Sequence-to-Sequence Learning for Indonesian Automatic Question Generator,” *7th International Conference on Advance Informatics: Concepts, Theory and Applications (ICAICTA)*, 2020.

[6]. H.-G. Lee, Y. Jang, and H. Kim, “Machine Reading Comprehension Framework Based on Self-Training for Domain Adaptation,” *IEEE Access*, 2021.

[7]. J. Du and Y. Gao, “Query-focused Abstractive Summarization via Question-Answering Model,” *IEEE International Conference on Big Knowledge (ICBK)*, 2021.

[8]. K. Ovchinnikova and E. Sidorova, “Analysis of Competency Questions for Automatic Generation of Lexico-Syntactic Ontology Design Patterns,” *IEEE International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)*, 2022.

[9]. L. Zhang and J. Liu, “Intent-aware Prompt Learning for Medical Question Summarization,” *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2022.

[10]. N. Pol, R. Sharma, M. S., Y. Singh, R. Arunadevi, and S. Kayathri, “Outcome Based Predictive Analysis of Automatic Question Paper Generation Using Hybrid MLP-CNN Approach,” *International Conference on Sustainable Communication Networks and Applications (ICSCNA)*, 2023.

[11]. Q. Liu, T. D. T. Do, and L. Cao, “Answer Keyword Generation for Community Question Answering by Multiaspect Gamma–Poisson Matrix Completion,” *IEEE Intelligent Systems*.

[12]. R. Ramli, S. Sivan, and H. Razalli, “A Review on Automated Examination Question Paper Template Generator,” *16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*, 2020.

[13]. S. Suakanto, J. Siswanto, D. Angela, T. F. Kusumasari, and T. W. Sen, “Paraphrase Generator for Indonesian Language: Basic Method for Generating Question on Interviewbot Engine,” *International Conference on Advancement in Data Science, E-learning and Information Systems (ICADEIS)*, 2022.

[14]. S. Gangopadhyay and S. M. Ravikiran, “Focused Questions and Answer Generation by Key Content Selection,” *IEEE Sixth International Conference on Multimedia Big Data (BigMM)*, 2020.

[15]. X. Qiu, H. Xue, L. Liang, Z. Xie, S. Liao, and G. Shi, “Automatic Generation of Multiple-choice Cloze-test Questions for Lao Language Learning,” *International Conference on Asian Language Processing (IALP)*, 2021.