IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

Generative AI for Question Generation, Answer Scoring and Faculty-Supervised Assessment

Prof. Dr. Aniruddha S. Rumale[1], Dr. Sushma Zade[2], Ganesh More[3], Sushant Gholap[4], Atmanand Mashal[5], Kalynai Patil[6]
Department of Information Technology[1-6]
Sandip Institute of Technology and Research Centre (SITRC), Nashik, Maharashtra, India
arumale@gmail.com , ganeshmore7125@gmail.com , sushantgholap699@gmail.com , atmanandmashal6@gmail.com Kalyanipatil6251@gmail.com

Abstract: *Writing question papers and marking descriptive answers remain the slowest, most subjective parts of examination. This review organises recent work on automating them into three themes: generating questions, scoring answers, and deciding how far automated scores can be trusted. Eight core studies on Bloom's-level question generation, semantic and multi-signal evaluation and LLM-based scoring are examined, together with two 2025 studies. The synthesis shows that quality falls at higher cognitive levels, that combined scoring signals outperform similarity alone, that prompt design strongly affects reliability, and that no surveyed work integrates the full examination workflow. We therefore outline a faculty-in-the-loop reference design in which every generated question and every low-confidence score is checked by a human before use, and set out a plan for evaluating it. A web prototype built with Flask, MySQL and a hosted LLM already implements question generation, timed delivery and objective scoring for multiple-choice examinations, while descriptive scoring and confidence routing remain future work; no evaluation results are reported yet..*

Keywords: Generative AI, Large Language Models, Automated Question Generation, Descriptive Answer Evaluation, Bloom's Taxonomy, Semantic Similarity, Human-in-the-Loop

I. INTRODUCTION

Consider the work behind a single end-semester paper. A faculty member must decide what the syllabus demands, write questions that cover it in proportion, balance easy and hard items, and later read every script, often late in a long stack, before marks are totalled and transcribed. Computers have eased the clerical end of this chain, but the two intellectually heavy steps, authoring questions and judging descriptive answers, are still done by hand. The consequences are familiar: slow results, uneven marking between examiners, and feedback that tells a student a score but not the topics behind it.

Large Language Models (LLMs) [9], [10], [14] and sentence-embedding models [12] now make both steps technically approachable. An LLM can be prompted with a syllabus segment to draft questions, and an embedding model can turn a student answer and a model answer into vectors whose cosine similarity gives a graded score. Published work has begun to test these ideas, yet each study tackles one narrow task and few ask how the pieces should fit into a working examination system with accountable human oversight.

This paper therefore asks three questions. (RQ1) What does current evidence say about generating exam questions with LLMs? (RQ2) What does it say about scoring descriptive answers automatically? (RQ3) What design follows if both are to be used in a real institution? The main contributions are:

A theme-based review of eight core studies [1]–[8] and two 2025 additions [16], [17], organised as generate, score and trust.

A cross-study synthesis and a list of recurring lessons and open gaps.

Copyright to IJARSCT
www.ijarsct.co.in
DOI: 10.48175/IJARSCT-40148

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

A faculty-in-the-loop reference design with an architecture, scenario walkthroughs and an evaluation plan, and a prototype that implements its multiple-choice generation, delivery and scoring stages.

II. Background and Theoretical Foundation

A. From Paper Exams to Language Models

Examinations began as oral questioning and became standardised written papers. Optical mark recognition automated objective scoring decades ago, and learning management systems later digitised scheduling and records. Content delivery moved online readily, but assessment moved more slowly: where it went digital it usually meant auto-scored multiple-choice quizzes, while descriptive assessment stayed on paper or on screen with manual marking.

B. Enabling Technologies

Three building blocks recur in the studies reviewed. Transformer LLMs [9]–[11], [14] generate text conditioned on a prompt. Sentence embeddings [12] map text to vectors so that meaning can be compared numerically. Bloom's Taxonomy [13] supplies six cognitive levels (Remember, Understand, Apply, Analyze, Evaluate, Create) that let a question be tagged and its difficulty spread controlled. Their combination became practical only recently, because LLM APIs are affordable for student and institutional projects, embedding libraries are mature, and institutions are looking for ways to cut grading load without hurting fairness.

III. Related Work

We group the core studies by the question they help answer. Table I summarises each study.

TABLE I: Summary of the Core Literature

Study	Theme	What was done	Main caveat
Scaria et al. 2024 [1]	Generate	Five LLMs prompted for questions at each Bloom's level; expert and LLM evaluation	Create level weak; one domain
Duong-Trung et al. 2024 [2]	Generate	GPT-3.5-turbo fine-tuned on 1,026 questions across 29 topics (BloomLLM); beat plain GPT-4	Needs curated data; two courses
Bulathwela et al. 2023 [3]	Generate	Pre-trained models produce questions and distractors at scale	Weak distractors
ESWA 2023 [4]	Score	Keyword matching, similarity matrix and NER combined in a neural network	Needs keyword set per question
Aynetdinov & Akbik 2024 [5]	Score	SemScore: embedding cosine similarity, tested on 12 LLMs against 8 other metrics	Not tested on exam answers
ACM 2024 [6]	Trust	GPT-4, GPT-3.5, PaLM, LLaMA2 as essay scorers; QWK 0.68	Below human-rater agreement
PMC 2024 [7]	Trust	GPT-4, GPT-3.5, Claude 2 under varied prompts; 112–114% gain from rubric-referenced prompt	Prompt-sensitive
ScienceDirect 2024/2026 [8]	Trust	181 essays, 20 human raters versus GPT-4	L2 essays; stable is not correct

A. Generating Questions

Scaria et al. [1] found that LLMs produce relevant, high-quality questions when given adequate context, but quality differs across models and levels: Remember and Understand items are reliable, Create items are not. Duong-Trung et al. [2] fine-tuned a model on 1,026 curated questions and reduced hallucinated or impractical items compared with prompting alone, though the dataset is a cost most projects cannot meet. Bulathwela et al. [3] showed that pre-trained models can draft usable candidates across large content sets without per-topic tuning, but their multiple-choice distractors were sometimes implausible or easy to eliminate.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

B. Scoring Descriptive Answers

The ESWA study [4] combined keyword or concept matching, a semantic similarity matrix and named-entity recognition, and obtained better agreement with human marks than similarity alone, notably fewer errors from entity mismatches. SemScore [5] showed that a plain embedding cosine similarity correlates with human judgement more strongly than BLEU, ROUGE-L and the other metrics tested. Its limits are that it was validated on general instruction-following output and that one number cannot say which part of an answer is wrong.

C. Trusting Automated Scores

Prompted LLM scoring reached a QWK of 0.68 against human markers [6]: useful, yet below the agreement expected between two trained raters. Prompt design dominated reliability in [7], where a criteria- and sample-referenced justification prompt raised GPT-4 accuracy by 112–114% over weaker prompts. Study [8] adds a nuance: GPT-4 was steadier across repeated runs than individual humans were against each other, but consistency does not prove correctness. Two 2025 studies extend this line. One compares GPT-4 with prompt engineering against traditional short-answer-grading baselines [16]; the other uses retrieval-augmented generation to bring domain material into grading [17].

IV. Cross-Study Synthesis and Research Gap

Reading the studies together yields five recurring lessons.

Level matters. Reliability falls as cognitive demand rises, so Create-level questions need the most scrutiny [1], [2].

Options are a separate problem. Distractor quality lags behind question stems [3].

One signal is rarely enough. Combined keyword, entity and similarity signals beat similarity alone [4], while similarity alone is a strong, simple baseline [5].

Prompts are part of the model. Output quality shifts sharply with prompt design [7], so prompts must be tested, not assumed.

Scores need a safety net. LLM-only scoring is substantial but imperfect [6], and stability is not accuracy [8].

A. Research Gaps

First, there is no integrated system: the studies handle generation, delivery, scoring and analytics separately. Second, evidence for prompt-only generation without fine-tuning is thin [2]. Third, embedding metrics are not validated on short exam answers [5]. Fourth, little work combines semantic similarity with keyword or entity checks and routes low-confidence results to faculty. Fifth, STEM and technical descriptive answers are under-studied compared with essays [8]. The gap we address is the absence of a practical, integrated and faculty-supervised examination system.

V. Proposed Faculty-in-the-Loop Reference Design

Fig. 1 shows the design implied by the synthesis. The guiding rule is that every AI-produced artefact, whether a question or a score, passes a faculty check or a confidence-threshold check before it counts.

A. Pipeline

Faculty upload a syllabus, which is cleaned and split into topic chunks. A structured prompt per chunk states the question type, Bloom's level and marks, and the LLM returns candidate questions stored as pending. Faculty edit, approve or reject each item, including MCQ options. Approved items are assembled into a timed exam. Objective answers are matched against the key; descriptive answers receive a hybrid score. Borderline scores wait for faculty confirmation, and finalised scores feed topic- and outcome-level analytics and student feedback.

B. Two Scoring Paths

Objective items (MCQ, fill-in-the-blank) have a closed answer space, so a normalised exact match against the stored key is enough; embeddings would add cost without benefit, and no partial credit is given. Descriptive items have an open answer space, so a keyword-overlap score and an embedding cosine similarity are combined as a weighted sum before marks are awarded, following [4] and [5]. A starting split such as 0.8 similarity to 0.2 keyword is only a

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

hyperparameter to be calibrated against faculty-graded data; no weighting has been validated. Because the two paths have different per-answer cost, response time and generation latency are tracked separately.

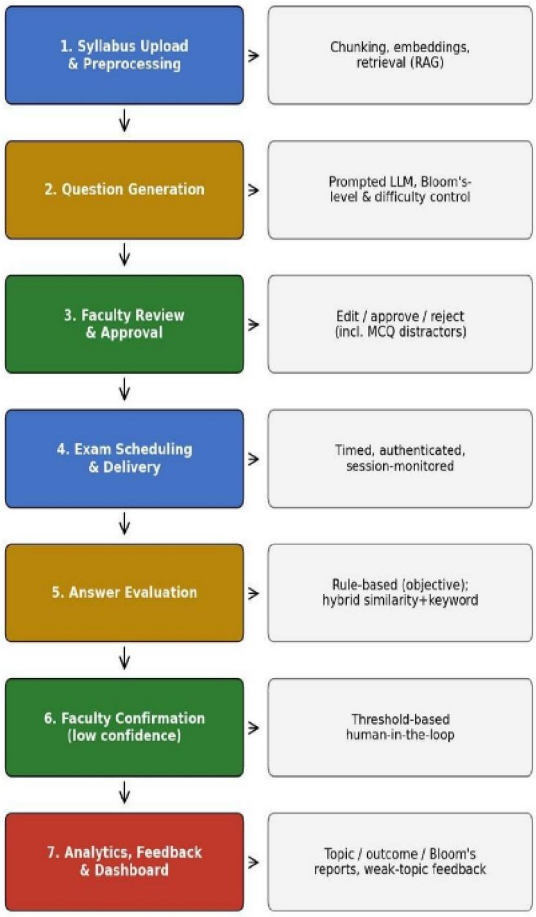


Fig. 1. Reference pipeline. Blue: workflow step; gold: AI-driven step; green: faculty checkpoint; red: output and reporting

C. Architecture

A web presentation layer talks by REST to a Flask backend, which coordinates the LLM service, the evaluation engine (Sentence-Transformers [12]) and the analytics engine. MySQL is the system of record for users, questions, exams, responses, results and analytics, so the three layers stay independently replaceable. Security rests on TLS, salted slow password hashing, server-side role checks on every endpoint, signed time-limited tokens, parameterised queries, and access to exam content gated by the exam window. Section VI describes the part of this architecture that has been built.

TABLE II: WHERE HUMAN CHECKS SIT IN THE DESIGN

Artefact	Risk (source)	Check	Owner
Generated question	Weak at higher Bloom's levels [1], [2]	Review before publishing	Faculty
MCQ options	Implausible distractors [3]	Review of all options	Faculty
Objective score	Closed answer space	Exact match	System
Descriptive score	Missed entities [4]; moderate agreement [6]	Confidence flag	Faculty
Analytics	Built on unconfirmed marks	Confirmed data only	System

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

D. Illustrative Scenarios

Question set. A faculty member uploads a unit and asks for a Bloom's mix; pending questions are reviewed and approved. *Borderline answer.* A student's short answer scores near the threshold, is flagged, and the faculty member confirms or corrects it before release. *Feedback.* After finalisation, the student sees weakest topics, and faculty see cohort weaknesses without cross-referencing a syllabus map.

VI. Prototype Implementation

To show that the multiple-choice part of the design can be built with commodity tools, we implemented a working web prototype named GenExam AI. It realises the generation, delivery and objective-scoring stages of Fig. 1. Table III states the status of every stage of the design.

A. Technology Stack

The prototype is a three-tier web application. The presentation layer is written in plain HTML5, CSS3 and JavaScript, without a front-end framework, as linked pages that are served separately from the API. The application layer is a Python Flask service, and persistent data is held in a MySQL database accessed through the mysql-connector-python driver. Questions are generated by calling a hosted large language model through its HTTP API, and PyPDF2 is used to read uploaded PDF documents.

B. Frontend

The interface has three groups of pages. Public pages (landing, registration and login) introduce the system and authenticate users. Student pages comprise a dashboard that lists the available exams with the number of exams attempted, exams passed and the average score; the examination page; a results page that lists every attempt and opens a detailed view; and a profile page where the student can update their name. The administrator page is organised into five tabs: dashboard, exams, questions, students and results. Pages share a common script for communication with the REST API. After login, the token and basic user details are kept in the browser's local storage and sent as a bearer token with each request, a page opened without a token returns the user to the login page, and the navigation and portal label change with the user's role. The layout is responsive, with a collapsible sidebar on small screens.

The examination page opens with an introduction screen that shows the number of questions, the duration, the total marks and the passing marks, and states the rules: one attempt, automatic submission when the timer expires, and results shown immediately. During the exam one question is displayed at a time with Previous and Next buttons, a live countdown, and a question palette that marks each question as answered, current or not answered. Submission is confirmed through a dialog that summarises progress. The result view shows pass or fail status, the percentage, marks obtained, total marks and the numbers of correct and wrong answers, followed by a review of every question that can be filtered by all, correct, wrong or skipped, and that shows the stored explanation where one exists. The result can be printed.

Administrators create an exam by entering its title, subject, duration, number of questions, total and passing marks, difficulty (easy, medium or hard), status and description. For a selected exam, questions can be produced in three ways: from a topic, a count and a difficulty level sent to the LLM; from an uploaded PDF, whose text is extracted and used to generate questions; or by typing them manually. Every question has four options, one correct answer, marks and an optional explanation, and any question can be edited or deleted afterwards. The administrator also sees the list of students and the results of all attempts.

C. Backend

The backend is a Flask application organised around three Blueprints for authentication, administration and examination, mounted under /api/auth, /api/admin and /api/exam. Routes, models, services and utility modules are kept in separate packages, and a small database module supplies connections. Settings, including the database credentials, the application secret and the LLM API key, are read from environment variables with python-dotenv instead of being written into source code. Flask-CORS allows cross-origin calls to the API so that the interface can be hosted separately, and the service provides a health-check route and returns JSON bodies for unknown routes and server errors. The

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

interface uses calls such as the current-user and profile-update endpoints under /api/auth and the student results endpoint under /api/exam.

Passwords are stored as salted hashes produced with the Werkzeug security utilities, sessions use signed JSON Web Tokens (PyJWT), and SQL statements are parameterised. Each account carries a role, student or administrator, and the first administrator account is created by a one-time setup script. For generation, the service sends the topic, count and difficulty, or the text extracted from a PDF, to the LLM and asks for the questions as structured JSON containing the question text, four options, the correct answer and an explanation.

D. Status Against the Reference Design

Table III maps the stages of the reference design to the prototype. The prototype follows the design's main rule only in part: an administrator can add, edit or delete any question, but an explicit pending and approval state, descriptive scoring, confidence flagging, Bloom's-level tagging and topic-level analytics are not yet implemented.

TABLE III: DESIGN STAGES AND THEIR STATUS IN THE PROTOTYPE

Design stage	Status	What the prototype does
Accounts and roles	Implemented	Registration, login, signed tokens, student and administrator roles
Exam set-up	Implemented	Administrator form for title, subject, duration, question count, marks, difficulty and status
Question generation	Implemented	MCQs from a topic prompt or an uploaded PDF, plus manual entry
Faculty review	Partial	Add, edit and delete of any question; no separate approval state yet
Timed delivery	Implemented	Countdown timer, question palette, confirmation before submission
Objective scoring	Implemented	MCQ answers marked against the stored key; marks, percentage and pass or fail
Feedback	Implemented	Per-question review with correct answer and explanation
Analytics	Partial	Student totals and average; administrator lists of students and results; no topic-level analytics
Descriptive scoring	Not yet	Hybrid keyword and embedding score of Section V-B is planned
Confidence flagging	Not yet	Routing of borderline scores to faculty is planned
Bloom's-level tagging	Not yet	Questions carry a difficulty level, not a Bloom's level

VII. EVALUATION PLAN

Neither the design nor the prototype has been evaluated, so we specify how they will be tested. The prototype will be compared with a faculty-prepared control set of the same exam. The scoring measures apply once descriptive scoring is added; until then, objective scoring is verified against the stored key. Table IV lists the measures; agreement between automated and faculty marks will be reported with a kappa-style statistic, as in [6].

TABLE IV: PLANNED EVALUATION MEASURES

Area	Measures
Question quality	BLEU-4, ROUGE-L, METEOR, ChrF, BERTScore; human ratings (1–5) of grammaticality, appropriateness, relevance, complexity, answerability
Scoring quality	Cosine similarity / SemScore, keyword and entity match, QWK, flagging precision and recall, evaluation consistency
System	Response time, generation latency, scalability, throughput, reliability, user satisfaction

VIII. LIMITATIONS AND ETHICAL CONSIDERATIONS

The reference design is only partly implemented: the prototype covers multiple-choice generation, delivery and objective scoring, and has been checked only for database and API connectivity, so its output quality is unmeasured.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

Because it handles multiple-choice items only, it does not exercise the hybrid scoring of Section V-B. Before real examinations, the prototype also needs hardening: the development server runs over plain HTTP, and keeping the access token in browser local storage exposes it to script injection if a page is ever compromised. Generated questions can hallucinate or stray from the syllabus, and generation depends on an external API with its outages, rate limits and per-call cost; both generation and delivery need reliable internet. Quality may vary with subject domain, and technical vocabulary may need prompt tuning or richer model answers. The reviewed evidence leans on essays and general text [8], so transfer to technical answers is unproven.

Ethically, consistent scoring is not necessarily fair scoring, so marks must be auditable and contestable. Student responses and model answers are sensitive and need protection in transit and at rest. Teams should watch for automation bias, where reviewers accept AI output uncritically; the confidence threshold should direct attention, not replace it. Tab-switch detection and session locking reduce opportunistic malpractice but do not stop a determined student, for example using a second device.

IX. Conclusion

The evidence reviewed shows that LLMs and embeddings are useful for drafting questions and scoring answers, particularly at lower cognitive levels, but not yet dependable enough for unsupervised high-stakes use. Prompt design, combined scoring signals and human checkpoints emerge as the practical levers. The reference design places faculty at the two riskiest points, higher-order question quality and low-confidence scores, while automating the repetitive work around them. The prototype of Section VI already covers multiple-choice generation, timed delivery and objective scoring. The next steps are to add descriptive scoring with confidence routing and an explicit approval state, and to test the system against the plan in Section VII.

References

[1]. N. Scaria, S. Dharani Chenna, and D. Subramani, "Automated educational question generation at different Bloom's skill levels using large language models: Strategies and evaluation," in Proc. AIED, Springer, 2024, pp. 165–179.
[2]. N. Duong-Trung et al., "BloomLLM: Large language models based question generation combining supervised fine-tuning and Bloom's Taxonomy," in Proc. EC-TEL, Springer LNCS vol. 15160, 2024, pp. 93–98.
[3]. S. Bulathwela, H. Muse, and E. Yilmaz, "Scalable educational question generation with pre-trained language models," in Proc. AIED, Springer, 2023, pp. 327–339.
[4]. "Revolutionizing subjective assessments: A three-pronged comprehensive approach with NLP and deep learning," Expert Systems with Applications, Elsevier, 2023.
[5]. A. Aynetdinov and A. Akbik, "SemScore: Automated evaluation of instruction-tuned LLMs based on semantic textual similarity," arXiv:2401.17072, 2024.
[6]. "On automated essay grading using large language models," in Proc. 8th Int. Conf. Computer Science and Artificial Intelligence, ACM, 2024.
[7]. "Harnessing LLMs for multi-dimensional writing assessment: Reliability and alignment with human judgments," PMC, 2024.
[8]. "Comparing GPT and human raters in essay assessment: Variability, bias, and the potential of LLM-based scoring," ScienceDirect, 2024/2026.
[9]. A. Vaswani et al., "Attention is all you need," in Proc. NeurIPS, 2017.
[10]. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL, 2019.
[11]. OpenAI, "GPT-4 technical report," 2023.
[12]. N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in Proc. EMNLP, 2019.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 5, October 2026
Impact Factor: 8.2

[13]. L. W. Anderson and D. R. Krathwohl, Eds., A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Addison Wesley Longman, 2001.
[14]. T. Brown et al., "Language models are few-shot learners," in Proc. NeurIPS, 2020.
[15]. K. P. Yancey, G. Laflair, A. Verardi, and J. Burstein, "Rating short L2 essays on the CEFR scale with GPT-4," in Proc. BEA Workshop, 2023, pp. 576–584.
[16]. R. F. Mello et al., "Automatic short answer grading in the LLM era: Does GPT-4 with prompt engineering beat traditional models?," in Proc. LAK '25, ACM, 2025, pp. 93–103.
[17]. Y. Chu et al., "Enhancing LLM-based short answer grading with retrieval-augmented generation," arXiv:2504.05276, 2025