

Design and Deterministic Validation of a Teacher Supervised Multi Agent Lesson Planning Pipeline with Source Traceability, Timing Constraints, Accessibility Checks, and Approval Gates

Nayyar Ahmed Khan^{1*}, Mohammed Ahmad² & Rim Hamdaoui²

¹Department of Computer Engineering, College of Computing and Information Technology, Shaqra University, Saudi Arabia.

²Department of Computer Science, College of Science and Humanities, Dawadmi, Shaqra University, Saudi Arabia.

Corresponding Author (Nayyar Ahmed Khan) Email: nayyar@su.edu.sa*



DOI: <https://doi.org/10.46431/mejast.2026.9310>

Copyright © 2026 Nayyar Ahmed Khan et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Article Received: 23 July 2026

Article Accepted: 25 September 2026

Article Published: 30 September 2026

ABSTRACT

This paper specifies a teacher supervised multi agent pipeline for lesson planning. Planner, source, access, assessment, and timing checks produce a versioned packet requiring teacher release. A deterministic validator was run on 12 constructed packets with 18 seeded defects and detected all 18 without false flags in that suite. A worked 45-minute lesson shows how a revision must be rechecked after timing and assessment changes. The unit test verifies packet rules, not large language model (LLM) lesson quality, classroom feasibility, or teacher time savings.

Keywords: Large Language Model Agents; Lesson Planning; Teacher Supervision; Source Traceability; Activity Timing; Accessibility Review; Assessment Alignment; Versioned Lesson Packets; Deterministic Validation; Multi Agent Workflow; Educational Quality Assurance.

1.0. Introduction

Lesson planning requires factual correctness, alignment with a learning objective, feasible timing, usable materials, and accessible activities. An LLM can produce a coherent plan quickly, but coherent prose does not establish that the plan can be taught. A multi agent system can split drafting and review into roles with visible outputs. The research question is whether a teacher can inspect and revise such a plan more effectively when checks are explicit and source linked [1 – 5].

This paper specifies a read-only planning pipeline. The agents do not contact students or publish to a learning platform. The teacher provides a brief, reviews flagged issues and approves a versioned lesson packet. The design treats teacher time as a measurable resource: a fast first draft has little value if verification and correction take longer than ordinary planning. The contribution is a role and permission model, a structured release packet, a deterministic timing check, a worked case, and an evaluation protocol [6–12]. The paper is a technical design study. It does not present measured lesson quality, workload savings, or student achievement.

1.1. Study Objectives

- Define role-specific inputs, outputs, and tool permissions.
- Construct a versioned lesson packet with a teacher release gate.
- Check source support, activity timing, access, and objective alignment.
- Validate deterministic checks on constructed lesson packets.
- Trace corrections across successive packet versions.
- Specify a comparative study of plan quality and teacher workload.

2.0. Literature Review

Recent work since 2020 has applied large language models to curriculum design, feedback, and educational agents. Separating agents by task can improve the visibility of intermediate decisions, but multiple roles do not guarantee independent reasoning when they share a model or source collection. A verifier can agree with a planner for the same mistaken reason. Source citation, deterministic checks, and teacher review therefore remain necessary [13-17]. Education guidance emphasizes teacher judgment, professional capability, privacy, and equitable access. A lesson that is scientifically accurate may still be inaccessible, unrealistic for the period, or misaligned with the intended assessment. This design treats those as separate review dimensions. The article proposes a testable workflow rather than claiming that agent teams outperform an individual teacher. Each agent receives only the fields needed for its role [18-22]. The planner and reviewers may read approved course sources and write into a staging packet. They cannot edit the live course, access identifiable student records, or send messages. A deterministic validator sums activity and transition times and flags any schedule exceeding T value [23-24]. A resource checker must mark unsupported claims explicitly rather than replacing them with fluent guesses.

The pipeline is a directed sequence with a limited revision loop: brief validation; plan generation; parallel source, inclusion, assessment, and timing checks; a consolidated issue report; one revision; teacher inspection; release or rejection [25-28]. A disagreement is preserved as an issue for the teacher. For example, an inclusion reviewer may propose simpler wording while a subject checker warns that the replacement loses a technical distinction. The last agent cannot silently overwrite the first. Figure 1 summarizes the sequence and the authority boundary described above.

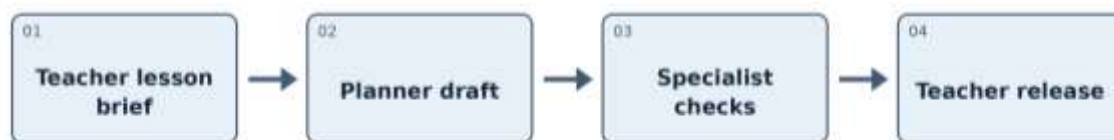


Figure 1. Teacher supervised planning pipelines.

A structured lesson brief enters a planner; specialist checks inspect sources, timing, access, and assessment; only a teacher may approve the versioned packet. Source: Original schematic prepared for this manuscript. [11] documented a teacher-centered workflow for differentiated planning, emphasizing iterative human decisions. [13] presented Lesson Agent, which generates lesson plans and related materials with teacher interaction and retrieval-based correction. Those studies inform the role split, while the present design makes schedule arithmetic, source support, access checks, and teacher release explicit packet-level conditions rather than inferring readiness from fluent output.

3.0. Methodology

The input brief B contains subject, learner level, duration T, prerequisite concepts, one observable learning objective, approved source identifiers, materials, and constraints. The planner produces a sequence of activities A with estimated minutes, student actions, and an exit task. The source checker returns a claim ledger C with passage identifiers and support status. The inclusion reviewer returns accessibility flags I and alternatives. The assessment checker returns an alignment judgment E. A teacher produces the approved lesson version V.

The release packet records the objective, timed activities, materials, exit task, source ledger, open issues, changes since the prior draft, model configuration, and teacher decision. The teacher checks each high-severity item before approving. A version identifier prevents a later model run from replacing the approved plan. A plan with an unresolved factual or safety issue cannot be released. These are operational gates, not a statement that all possible errors will be detected.

Equation (1) states the acceptance or scoring rule used by the proposed method; its variables are defined in the surrounding text and Table 1.

$$T_{\text{plan}} = \sum_i \text{minutes}(A_i) + \text{transition_minutes};$$

$$\text{Flag} = [T_{\text{plan}} > T] \vee \text{Unsupported} \vee \text{NoAlt} \vee [\text{Verb}(\text{exit}) \neq \text{Verb}(\text{objective})]. \quad (1)$$

In Equation (1), A_i is activity i , $\text{minutes}(A_i)$ is its planned duration, $\text{transition_minutes}$ is the sum of planned transitions, T_{plan} is total planned time, and T is the lesson period. Flag is true if time exceeds T , any factual claim is Unsupported, an accessibility alternative is absent (NoAlt), or the action verb in the exit task differs from that in the objective. The bracketed comparisons evaluate to true or false.

Algorithm 1. Offline decision procedure

INPUT brief B and planner packet $V0$

Validate objective, duration, source IDs, and permissions

Run time, source-ID, access, and objective-verb checks

Create issue ledger with evidence and severity

Revise $V0$ to $V1$; rerun every affected check

Teacher approves, overrides with reason, or rejects $V1$

The claim ledger uses one record per checkable statement: claim text, source passage, support decision, severity, and proposed edit. Unsupported decorative examples can be removed; an unsupported central concept blocks release. Severity is assigned before the planner sees the final review so that the agent cannot reduce a flag merely by rephrasing. The teacher can override a flag with a reason and approved source [29]. This produces an audit trail without making the automated reviewer the final authority.

Timing validation is deterministic. For activities $A1$ to A_n , total planned time equals the sum of activity minutes plus transitions and setup. The validator compares that total with T and flags the difference. It should also check that the exit task has a nonzero allocation and that group work has time for instructions and reporting. A schedule can meet the arithmetic total yet remain implausible if every learner must finish simultaneously; the teacher reviews that qualitative aspect [30-33]. The system records both the arithmetic result and the teacher's feasibility judgment.

Assessment alignment is evaluated with a mapping from objective verb to observable learner action. If the objective asks learners to compare evidence, an exit item requiring only a definition is flagged. The reviewer proposes a question that demands the comparison and states what a satisfactory answer would contain. The teacher

checks whether students practiced that action during the lesson. A mismatch can be repaired by changing the activity, the exit task, or the objective, but the agent should not silently change the teacher’s original goal.

The inclusion check uses a concrete rubric: readability of directions, text alternatives for visuals, availability of materials, options for participation, and the preservation of the same learning goal. A suggested simplification is flagged if it changes technical meaning [34-35]. A video link is tested in the learner environment and checked for captions. The review is not a diagnosis of individual students. It operates on the class-level brief and on the designed activity, leaving individualized support decisions to authorized staff.

Table 1 defines the main variables and control checks used in this method.

Table 1. Agent contracts and authority limits for lesson drafting, source checking, accessibility review, and final teacher release.

Role	Required output	Release authority
Planner	Timed activities and exit task	None
Source checker	Claim ledger with passage IDs	None
Inclusion reviewer	Access flags and alternatives	None
Teacher	Approved version and edits	Final release

Ablation testing can determine whether each agent role adds value. Run the same development briefs with planner only, planner plus source checker, and the full pipeline; hold model, sources, and teacher reviewers constant. Compare severe errors, teacher correction time, and useful flags. The purpose is to identify a role that reduces errors at an acceptable cost, not to assume that more agents are always better. A held-out set is needed before any comparative claim. The design also allows a simple checklist baseline, which may outperform a costly agent on routine omissions.

The brief schema should reject an empty objective, a nonpositive duration, or a source that the teacher cannot access. Optional fields can include group size, language of instruction, equipment, and homework constraints [36-40]. The system returns missing fields as questions instead of filling them with defaults. This input validation is important because an agent can otherwise create a credible plan around invented equipment or a guessed age range. A brief identifier and version allow the approved lesson to be traced to the teacher’s actual request.

The source checker should inspect the granularity of claims. “Plants need light for photosynthesis” and “all plant mass comes from soil” are separate statements even if they occur in one generated sentence. It records the source span and whether the claim is directly supported, contradicted, or outside the supplied material. A claim outside the material may be correct in the wider world but is still flagged for teacher verification. This avoids turning the source checker into an unrestricted web search whose output changes without course approval.

Issue consolidation should preserve provenance. Every finding has an issue ID, generating role, severity, evidence, proposed change, and status. A planner revision must link to the issue it addresses. An unresolved issue remains in the packet even if the revised prose no longer uses the original wording, until the relevant checker confirms resolution or the teacher records an override. This prevents a model from clearing a warning by rewriting it superficially. A released packet contains the final state of every severe issue.

A quality rubric can assign anchored ratings from 0 to 4 for each dimension. Source accuracy at level 4 requires all central claims to match approved material; level 0 contains a consequential error. Timing at level 4 includes activity, transition, and exit-task minutes within T; level 0 cannot be taught in the period. Accessibility at level 4 includes usable alternatives for required media and participation. The anchor descriptions should be agreed before reviewers see agent outputs. Severe errors are reported separately from mean ratings.

The offline validator test contains 12 constructed lesson packets. Each packet specifies an allowed duration, activity minutes, transition time, a source-support flag, an alternative-format flag, and objective and exit verbs. Eighteen defects are seeded across four categories. The validator sums minutes and compares the explicitly recorded flags and verbs; its expected outputs are known before execution. This test measures consistency of deterministic packet checks. It does not evaluate the quality of LLM-generated activities, the truth of a scientific claim, or whether a teacher would consider a lesson teachable.

The proposed evaluation uses a balanced sample of lessons across subjects, difficulty levels, and resource constraints. One condition uses the agent pipeline; the comparison uses the institution's ordinary planning process. Teachers are trained on both and record brief preparation, drafting, source checking, revision, and release time separately. The study design must account for repeated use and teacher experience. A pilot of several lessons is needed before choosing a sample size for a formal comparison.

Two independent reviewers score each final plan on source accuracy, objective alignment, feasible timing, accessibility, and assessment fit using anchored criteria. Count severe errors separately from average rubric scores. Time cost is total minutes from brief to approved lesson, not generation time [41-45]. Also record the fraction of flags accepted by teachers, the fraction of serious issues missed by agents, and teacher edits after release. Reviewer disagreement should be resolved with written reasons, not hidden by a composite score. Privacy is managed by excluding individual student profiles from B and limiting draft access to the course team. Class-level accommodations can be stated without names. A school should check provider retention terms before sending course materials. Re-run the evaluation set after changing the model, prompt, connected tools, or source corpus. The protocol is a prospective method; it is not a report of a conducted trial.

4.0. Results and Discussion

Across the 12 constructed packets, the validator identified all 18 seeded defects: six timing excesses, four unsupported-source flags, four missing access alternatives, and four objective-exit verb mismatches. There were no false flags within this tightly specified input set [46-49].

Table 2. Constructed issue ledger for a science lesson, listing the timing, source, access, and assessment defects that a teacher must resolve before release.

Check	Flag in first draft	Proposed resolution
Timing	55 minutes for a 45-minute period	Remove or shorten activity
Source	Unsupported plant-mass claim	Delete or verify in approved text
Access	Color-only diagram	Add labels and text description
Assessment	Definition-only exit question	Ask for evidence interpretation

This 18/18 result is a unit-test outcome, not an estimate of sensitivity on authentic lessons. A semantic mismatch can pass when the verbs match but the content differs; the teacher must review that case. Table 2 records the constructed trace or issue case used to illustrate the decision path.

The worked photosynthesis packet illustrates how a detected issue changes another requirement. Its first version totals 55 minutes in a 45-minute period and contains an unsupported claim, a color-only diagram, and a definition-only exit task. A revised schedule of 5, 10, 15, 8, and 5 minutes plus 2 minutes of transitions totals 45 minutes. Removing an activity to fit the time could nevertheless remove the practice needed for the new exit task, so objective alignment is rerun after revision. Table 3 reports on the offline component-test outputs and their denominators.

Table 3. Deterministic validator outcomes on 12 constructed lesson packets containing 18 prespecified defects; these values are unit-test results, not classroom performance.

Defect category	Seeded	Detected	False flags
Time excess	6	6	0
Unsupported source flag	4	4	0
Missing access alternative	4	4	0
Objective-exit verb mismatch	4	4	0

The result establishes that the packet schema and arithmetic checks behave as specified on constructed records. It does not show that multiple LLM agents improve lesson quality or save teacher time. Those claims require paired plans, independent subject review, and complete time accounting in a future classroom study [50-52].

Role separation improves inspect ability only when outputs have specific contracts. A generic instruction to “review quality” can produce agreement without evidence. Source locations, deterministic arithmetic, and a teacher-visible disagreement are stronger controls. Even with these controls, a model may miss a subtle misconception, and an agent cannot know all classroom constraints. The teacher’s ability and time to review remain central.

The method may be particularly useful for generating alternatives and checking routine omissions. It can also introduce new work: cleaning sources, checking generated claims, and reviewing unnecessary flags. A future evaluation must compare complete preparation time and the quality of approved plans, stratified by subject and teacher experience [53]. Averages should not conceal a severe factual mistake or an activity that cannot be performed safely.

Equity requires checking the actual learning environment. A draft may assume a paid device, a video without captions, or a language background that not all learners share. Inclusion suggestions also require subject review so that simpler wording does not erase the concept [54]. The system is a planning aid with a bounded release gate, not an autonomous instructor.

The agent roles are correlated if they use the same model and prompt context. Agreement among them should not be interpreted as independent evidence. A deterministic time sum and a human reading of the cited source provide different kinds of checks [55]. The design should favor complementary methods rather than adding agents for the

appearance of consensus. A future evaluation can report errors that every automated role missed and analyze whether the source set or task contract caused the shared failure.

The comparison condition must be fair. Teachers using ordinary planning should have access to the same curriculum and usual peer support. Training time for the agent pipeline and the burden of preparing a brief should count toward its cost. If teachers reuse packets, the evaluation should distinguish initial creation from later adaptation. The unit of analysis may be a lesson nested within a teacher; simple averages can overstate precision when a few teachers create many lessons [56].

An agent pipeline can be simplified if a role fails to add value. If a deterministic timing check catches the same errors as a timing agent at lower cost, retain the deterministic check. If the inclusion reviewer produces many changes that a teacher rejects because they alter the objective, revise its contract and examples. The ablation study can reveal these tradeoffs [57]. More roles increase latency, token use, and the number of disagreements a teacher must resolve; the goal is the smallest workflow that catches material issues.

Teacher autonomy is a design constraint, not merely a preference. The system should offer editable source-linked suggestions, not lock a teacher into a generated sequence. A teacher override should be possible with a documented reason, except where a school has a separate safety rule. The evaluated artifact is the final approved packet and its preparation process. Student learning outcomes would require a further study of actual teaching, not an inference from document quality alone.

A lesson-packet evaluation should examine error severity as well as frequency. A missing material item may be inconvenient, while a false central explanation can undermine the lesson [58-60]. Reviewers can assign a prespecified severity label and report how many errors remain in the final approved version. The teacher is part of the intervention, so the evaluation should also examine whether flags are presented in a form that helps teachers notice and correct them. Counting automated flags without examining the final plan would miss the actual purpose of the workflow.

Latency may affect use even if total preparation time is favorable. A teacher revising a lesson the evening before class needs a response within a practical window. Measure the time to first draft, time to completed checks, and time waiting for a teacher decision separately. The architecture may run independent specialist checks in parallel, but that changes neither their authority nor the need to reconcile conflicts. If a checker times out, the packet must show that its review is incomplete rather than displaying a false approval.

Source governance is another limitation. A school may have official curriculum files, teacher notes, and material from external publishers with different permissions. A checker should cite only material that can be used in the course and seen by the reviewer. If a new example requires an external source, the agent can mark it for approval rather than silently importing it. The release packet should record which documents supported central claims so that a later curriculum update can identify affected lessons.

The proposed design can fail through automation bias. A teacher may spend less time inspecting a polished packet than a rough one. A release interface should foreground severe flags, unsupported claims, and assumptions, with links to the original sources [61]. A future study can compare the rate at which teachers catch planted errors in a

plain draft and in a packet with checks. Such a test would measure the human-automation interaction rather than assuming that a visible checklist guarantees attentive review.

The illustrative issue ledger shows a dependency between corrections. Shortening a 55-minute sequence can remove the very activity that practices interpretation, making the revised exit question unfair. The planner must therefore rerun objective and assessment checks after the timing edit. Similarly, replacing a color-only diagram with a text alternative may change the explanation of the phenomenon; the source checker should review the revised description. The pipeline treats a correction as a new version, not a patch exempt from validation. A release decision can be summarized as approved, approved with documented minor edits, or rejected. A severe factual or safety issue precludes either approval state [62]. The packet records which flags the teacher accepted or overrode, the final minutes, and the material actually available. These fields form proposed data for a later study of teacher workload and issue detection. The worked example contains no measured teacher ratings.

The case can be represented as a version chain. Version 0 contains the 55-minute schedule, unsupported claim, color-only diagram, and definition-only exit item. Version 1 shortens an activity, deletes the unsupported statement, labels the diagram, and asks for evidence interpretation. The time check then passes arithmetically. A teacher may still reject Version 1 if the shortened activity no longer gives students enough practice. The version chain makes the distinction between automated resolution and pedagogical approval explicit. For the timing correction, consider a 5-minute opener, 10-minute source reading, 15-minute paired interpretation, 8-minute discussion, 5-minute exit response, and 2-minute transition allowance. These figures sum to 45 minutes. They are an illustrative schedule, not a tested estimate of how long students will need. A future study should compare planned and observed timings, recording whether the teacher changed the sequence during delivery.

5.0. Conclusion

The proposed agent pipeline turns a lesson draft into a source-linked, timed, and teacher-approved packet. Its technical value is traceable checks and explicit authority boundaries. A comparative pilot should establish whether those features improve plan quality at an acceptable total workload before deployment expands.

6.0. Future Recommendations

- 1) Test the pipeline on authentic lessons across several subjects.
- 2) Compare role ablations against a simple checklist baseline.
- 3) Measure planned versus observed classroom activity durations.
- 4) Evaluate how teachers detect deliberately planted content errors.
- 5) Assess accessibility suggestions with learners using varied formats.
- 6) Quantify total preparation and review time over repeated use.

Declarations

Source of Funding

The authors received no funding for this contribution.

Competing Interests Statement

There are no conflicts of interest by the authors in any way.

Consent for publication

The authors give consent for the publication.

Authors' contributions

All the authors contributed equally to the article.

Informed Consent

Not Applicable. The reported offline checks used constructed records and no human participants.

Availability of data and material

Constructed test inputs and deterministic procedures are described in the manuscript.

Institutional Review Board Statement

Not Applicable. No human participants or identifiable student records were included in the offline checks.

Ethical Approval

Not Applicable to the constructed offline unit tests.

References

- [1] Faith, L., Zaugg, T., Stoly, N., Szabo, M., Haghi, F., Badlis, C., & Olmedo, S.L. (2025). Persona, Break Glass, Name Plan, Jam (PBNJ): A new AI workflow for planning and problem solving. *AI*, 6(12): 310. <https://doi.org/10.3390/ai6120310>.
- [2] Quan, J., Ouyang, Y., Wang, H., & Wang, Y. (2026). LessonAgent: A multimodal pipeline for automated generation of lesson plans, presentations, and podcasts. *Information*, 17(2): 197. <https://doi.org/10.3390/info17020197>.
- [3] Ahmad, M., et al. (2025). Learning three-dimensional face recognition from sparse views for robust identity verification. Available at SSRN 5428214.
- [4] Akram, F., et al. (2024). Integrating artificial bee colony algorithms for deep learning model optimization: A comprehensive review. *Solving with Bees: Transformative Applications of Artificial Bee Colony Algorithm*, Pages 73–102.
- [5] Al Omari, O.M.A., Khan, N.A., & Mahafdah, R. (2017). Ranking and reputation based resource allocation in P2P system. *Mediterranean Journal of Basic and Applied Sciences*, 1(1): 293–301.
- [6] Alanezi, R., Alanezi, M.A., & Khan, N.A. (2018). Development of web based E-cooperative training system. In *2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE)*, IEEE.
- [7] Alangari, S., et al. (2022). Developing a blockchain-based digitally secured model for the educational sector in Saudi Arabia toward digital transformation. *PeerJ Computer Science*, 8: e1120.

- [8] Alangari, S., & Khan, N.A. (2021). Artificially intelligent warehouse management system. *Asian Journal of Basic Science & Research*, 3(3): 16–24.
- [9] Alotaibi, R., Khan, N.A., & Akhtar, M.M. (2025). Statistical probability prediction model for E-learning and realtime proctoring using IoT devices. *Journal of King Saud University–Science*, Page 37.
- [10] Alshalaan, M., & Khan, N.A. (2025). Complexities and challenges for securing digital assets and infrastructure in academia: A review on digital asset security. *Complexities and Challenges for Securing Digital Assets and Infrastructure*, Pages 225–244.
- [11] Alshalaan, M., & Khan, N.A. (2025). Blockchain-enabled federated learning framework for secure and collaborative drug discovery: Integrating AI, molecular docking, and distributed ledger technology.
- [12] Alshreef, B.S., & Khan, N.A. (2025). LSTM neural networks in healthcare disruption forecasting: A comprehensive review and implementation framework. In *International Conference on Innovative Computing and Cutting-edge Technologies*, Springer Nature Switzerland Cham.
- [13] Alshreef, B.S., & Khan, N.A. (2026). Disruption forecasting: A comprehensive review and implementation framework. In *Advances in Smart Computing and Emerging Technologies: Proceedings of the Third International Conference on Innovative Computing and Cutting-edge Technologies (ICICCT 2025)*, Springer Nature.
- [14] Alsulami, M.H., Alotaibi, S., & Khan, N. (2021). Smart university model for Saudi Arabian universities. *Design Engineering*, Pages 162–181.
- [15] Alsulami, M.H., et al. (2021). Zigbee technology to provide elderly people with well-being at home. *International Journal of Sensors Wireless Communications and Control*, 11(9): 921–927.
- [16] Ayadi, W., Khalid, M.N., & Khan, N.A. (2026). Learning and fuzzy control. In *Advances in Smart Computing and Emerging Technologies: Proceedings of the Third International Conference on Innovative Computing and Cutting-edge Technologies (ICICCT 2025)*, Springer Nature.
- [17] Ayadi, W., et al. (2025). Enhancing anomaly detection in autonomous systems using deep learning and fuzzy control. In *International Conference on Innovative Computing and Cutting-edge Technologies*, Springer Nature Switzerland Cham.
- [18] Gangwar, S., et al. (2026). AI-driven signal processing for single-cell data analysis: A case study in cancer research. In *Sensor Signal Processing for Intelligent Systems*, Pages 171–194, Springer Nature Switzerland Cham.
- [19] Haidar, S.W., et al. (2026). A secure data collaboration framework using differential partial AI and federated learning: Architecture, simulation, and performance evaluation. *Procedia Computer Science*, 282: 1237–1242.
- [20] Haidar, S.W., et al. (2025). Deep learning driven automated system for speech emotion recognition. In *2025 IEEE 7th International Conference on Computing, Communication and Automation (ICCCA)*, IEEE.
- [21] Haidar, S.W., et al. (2025). Enhancing problem-solving skills through modelling-based AI and generative flow networks (GFN). In *2025 2nd Global AI Summit-International Conference on Artificial Intelligence and Emerging Technology (AI Summit)*, IEEE.

- [22] Hassan, M.A.A., Khan, N.A., & Nasim, M.A.M. (2017). Managing data replication in mobile ad-hoc network databases using content based energy optimization. *Mediterranean Journal of Basic and Applied Sciences*, 1(1): 142–154.
- [23] Khan, N.A. (2018). Cloud applications development and deployment: The future of cost-effective programming and a step ahead. *Middle East Journal of Applied Science & Technology*, 1(1): 30–36.
- [24] Khan, N.A. (2019). Wireless requirements and benefits in the academics domain. *Middle East Journal of Applied Science & Technology*, 2(3): 45–49.
- [25] Khan, N.A. (2019). Basics of ethical hacking and computer security paperback–1 January 2019.
- [26] Khan, N.A. (2019). Security management protocols in cloud computation. *Middle East Journal of Applied Science & Technology*, 2(1): 16–23.
- [27] Khan, N.A. (2021). Measuring academics intentions to use a project management system (PMS): A case study of the College of Computing and Information Technology, Shaqra University. *Trends in Future Informatics and Emerging Technologies*, Pages 58–69.
- [28] Khan, N.A. (2022). Development of an artificially intelligent advising system for Saudi medical transcription. *Development*, 6(3): 94796.
- [29] Khan, N.A. (2024). Development of Intelligent Pick and Drop Service Manager for Small Cities. *Asian Journal of Basic Science & Research*, 6(3): 20–27.
- [30] Khan, N.A. (2024). Development of Intelligent Help System for Small Cities. *Asian Journal of Applied Science and Technology*, 8(3): 112–119.
- [31] Khan, N.A. (2025). Development of Intelligent Student Information System. *Arabian Journal of Basic Science and Research*, 7(1).
- [32] Khan, N.A. (2025). An application of machine learning, big data, and IoT of enterprise architecture: Challenges. *Quality Control-Artificial Intelligence, Big Data, and New Trends: Artificial Intelligence, Big Data, and New Trends*, Page 111.
- [33] Khan, N.A., & Ahamad, D. (2025). Living smart: AI-based urban assistance systems for sustainable wellbeing in small cities. *Middle East Journal of Applied Science & Technology*, 3(2): 89–99.
- [34] Khan, N.A., et al. (2021). Development of mubadarah system-an intelligent system for proposals at a university. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, IEEE.
- [35] Khan, N.A., et al. (2025). Transformative impact of artificial intelligence on higher education: A comprehensive analysis of pedagogical innovation, institutional transformation, and future learning ecosystems. *Asian Journal of Applied Science and Technology*, 9(4): 57–76.
- [36] Khan, N.A., et al. (2021). Development of Medidrone: A drone based emergency service system for Saudi Arabian Healthcare. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, IEEE.

- [37] Khan, N.A., et al. (2026). A microservices-based optical mark recognition framework with advanced image processing and cloud-native architecture for scalable educational assessment. *Middle East Journal of Applied Science & Technology*, 9: 40–49.
- [38] Khan, N.A., et al. (2026). Design, development, and large-scale experimental evaluation of a multilingual, header-aware, course-centric intelligent attendance management and automated academic report generation framework for higher education institutions.
- [39] Khan, N.A., et al. (2026). User-centered development of an academic content website.
- [40] Khan, N.A., et al. (2025). Network intrusion management of web form spamming using blockchain. *Irish Interdisciplinary Journal of Science & Research*, 9(03).
- [41] Khan, N.A., et al. (2024). An IoMT enabled iterative artificial bee colony approach using federated learning for detection of heart disease. In *Solving with Bees: Transformative Applications of Artificial Bee Colony Algorithm*, Pages 103–116, Springer Nature Singapore.
- [42] Khan, N.A., et al. (2026). Revolutionizing drug design: The convergence of AI and quantum computing—A systematic review. *Artificial Intelligence in Precision Drug Design*, 2: 317–326.
- [43] Khan, N.A., & Al Ghamdi, A.R. (2015). Cyber forensics and proposed techniques to overcome cyber threats for cyber security. *International Journal of Engineering and Management Research*, 5(5): 187–191.
- [44] Khan, N.A., & Albatein, J. (2021). COVIBOT—An intelligent WhatsApp based advising bot for Covid-19. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, IEEE.
- [45] Khan, N.A., & Alshalaan, M. (2025). AI-driven blockchain framework for digital transformation of academic accreditation process: A Saudi Arabian perspective.
- [46] Khan, N.A., & Alshalaan, M. (2026). Systematic review on machine learning applications in hardware security for vehicles. *Software Engineering and Cybersecurity in Modern Electric Vehicles*, Pages 91–110.
- [47] Khan, N.A., et al. (2021). An empirical analysis on users' acceptance and usage of BYOD-technology for Saudi universities: A case study of Shaqra University. In *2021 International Conference on Technological Advancements and Innovations (ICTAI)*, IEEE.
- [48] Khan, N.A., Khan, A.R., & Rajeyyagari, S. (2025). Innovation in teaching and learning with the use of modern computational tools: A post Covid experience. *Middle East Journal of Applied Science & Technology*, 8(2): 74–82.
- [49] Khan, N.A., et al. (2026). Serverless absence management: A Google-native playbook for rapid, auditable university leaves workflows.
- [50] Khan, N.A., et al. (2026). A cloud-based multi-tier approval system for academic course registration: Design, implementation, and evaluations.
- [51] Khan, N.A., et al. (2020). Internet of Things (IoT) based educational data mining (EDM) system. *Journal of Mechanical and Construction & Mathematical Sciences*, 15(3): 271–284.

- [52] Khan, N.A., et al. (2019). Intrusion management to avoid web-form spamming in cloud based architectures. In 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), IEEE.
- [53] Khan, N.A., et al. (2025). A data-driven analysis of online learning expectations: Perspectives from Middle Eastern students and educators. In International Conference on Innovative Computing and Cutting-edge Technologies, Springer Nature Switzerland Cham.
- [54] Khan, N.A., et al. (2026). A data-driven analysis of online learning expectations: Perspectives. In Advances in Smart Computing and Emerging Technologies: Proceedings of the Third International Conference on Innovative Computing and Cutting-edge Technologies (ICICCT 2025), Springer Nature.
- [55] Khan, N.A., Rajeyyagari, S., & Khan, A.R. (2025). Development of intelligent library services for university students. Mediterranean Journal of Basic and Applied Sciences, 9(1): 142–147.
- [56] Khan, N.A., Siddiqi, A.M.U., & Ahmad, M. (2021). Development of intelligent alumni management system for universities. Asian Journal of Basic Science & Research, 3(2): 51–60.
- [57] Khan, N.A., et al. (2019). Prevention of web-form spamming for cloud based applications: A proposed model. In 2019 Amity International Conference on Artificial Intelligence (AICAI), IEEE.
- [58] Kumar, D., & Dhir, S. (2021). Trends in Future Informatics and Emerging Technologies: Advanced Computing Techniques: Implementation, Informatics and Emerging Technologies, Volume 1.
- [59] Mahafdah, R.F., Al-Omari, O.M., & Khan, N.A. (2018). Learning modal adaptability to improve reading and writing skills of students.
- [60] Ren, M., et al. (2026). Explainable AI for structure-based drug design: Concepts and case studies. In Artificial Intelligence in Precision Drug Design, Volume 2, Pages 433–455, Elsevier.
- [61] Siddiqi, A.M.U., et al. (2026). Design and development. In Advances in Smart Computing and Emerging Technologies: Proceedings of the Third International Conference on Innovative Computing and Cutting-edge Technologies (ICICCT 2025), Springer Nature.
- [62] Siddiqi, A.M.U., et al. (2025). Design and development of a cost-effective software maintenance management model. In International Conference on Innovative Computing and Cutting-edge Technologies, Springer Nature Switzerland Cham.
- [63] Zamani, A.S., Akhtar, M.M., & Khan, N.A. (2025). An application of machine learning, big data and IoT of enterprise architecture: Challenges, solutions and open issues. In Quality Control-Artificial Intelligence, Big Data, and New Trends, IntechOpen.
- [64] Zamani, A.S., et al. (2026). Challenges in docking accuracy and reproducibility. In Recent Advances in Molecular Docking, Pages 435–446.