AI for Early-Stage Zoning and Massing Compliance: A Lightweight Prototype for the Bangladesh National Building Code (BNBC 2020)

Gautam Adhikary[1] and Erfan ZamaniGoldeh[2]
[1,2] *The Welsh School of Architecture, Cardiff University*
[1] *arc.gautam92@gmail.com, 0009-0001-6970-3877*
[2] *zamanigoldehe@cardiff.ac.uk, 0000-0003-2111-8457*

Abstract. This research presents an AI-assisted framework for early-stage building code compliance using the Bangladesh National Building Code (BNBC 2020) as a case study. The system integrates Large Language Models with a deterministic rule engine inside Rhino–Grasshopper to provide real-time feedback on zoning and massing. It bridges unstructured code text and geometric modelling through a transparent, lightweight workflow. Validation showed high reliability, with semantic queries achieving around 90% F1 accuracy and deterministic checks reaching full precision. The approach demonstrates a scalable, human-centred method to improve design efficiency and regulatory literacy in low-BIM, resource-limited contexts. The study specifically addresses the research question of how to reconcile textual building codes with fluid geometric exploration before fundamental design decisions are permanently locked in. By providing a platform-agnostic and modular architecture, the framework enables architects to navigate complex regulatory landscapes without the technical overhead of high-fidelity BIM environments, significantly reducing late-stage rework.

Keywords. AI-assisted code compliance, Early-stage design, Large Language Models, Deterministic rule engine, Rhino–Grasshopper Code Check, BNBC 2020

1. Introduction

Automated code compliance (ACC) has become an essential part of the digital transformation of architecture, aiming to reduce design errors, improve efficiency, and ensure that buildings meet regulatory standards. Yet, most existing compliance systems, such as CORENET ePlanCheck and Solibri Model Checker, operate only after a detailed Building Information Model (BIM) has been created (Dimyadi & Amor, 2013; Peng & Liu, 2023). By this stage, fundamental design decisions about form, zoning, and spatial organisation are already locked in, meaning that any detected non-compliance often leads to costly redesigns or compromises (Beach et al., 2020). This reactive approach delays regulatory validation and limits the potential of automation to meaningfully support creativity during the early stages of design.

HUMANISTIC COMPUTATION AND INTELLIGENCE, Proceedings of the 31st International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2026, Volume 1, 51-60.

In conceptual and schematic phases, architects explore massing and zoning options rapidly using flexible tools such as Rhino or SketchUp. These platforms prioritise geometric freedom over structured data, which makes them unsuitable for BIM-based rule engines that depend on object classification and metadata. Consequently, compliance checking at these stages remains largely manual. Designers estimate setbacks or floor-area ratios through approximations, and regulatory constraints are often revisited only when statutory approval is required. The absence of integrated early-stage validation tools creates inefficiencies and increases the risk of regulatory conflicts being discovered too late in the process.

The challenge is even more pronounced in developing contexts such as Bangladesh, where BIM adoption remains low and digital design standards are still emerging (Hossain & Ahmed, 2022). The Bangladesh National Building Code (BNBC 2020) is comprehensive in scope but fully text-based, with no structured data schema, open database, or API to support computational access (Hossain & Ahmed, 2022). Compliance assessments therefore rely on manual interpretation, which can vary between practitioners and across agencies. Limited computational capacity, inconsistent data exchange, and the absence of training frameworks compound the difficulty.

This research therefore proposes a human-centred, AI-assisted compliance framework that integrates LLM-based reasoning with lightweight geometric modelling. Using BNBC 2020 as a case study, the system connects a LangGraph pipeline for rule extraction with a deterministic compliance engine and a Rhino–Grasshopper interface for real-time feedback. Through this setup, zoning and massing decisions can be evaluated dynamically, allowing architects to test alternatives and understand the regulatory implications of their designs before full documentation or BIM modelling begins. Unlike Revit-based ACC tools that require high-fidelity IFC data and fixed object hierarchies, this prototype offers a platform-agnostic, lightweight alternative suited for the fluid geometry of early-stage conceptualization.

The study is structured around three main objectives. The first is to critically review the evolution of automated code compliance and identify the limitations of current BIM-dependent tools (Guo et al., 2021; J. Zhang & El-Gohary, 2014). The second is to develop a modular methodology that bridges unstructured text, AI reasoning, and 3D geometric evaluation. The third is to evaluate the resulting prototype through quantitative accuracy tests and qualitative assessments of its usability and transparency.

2. Related Work / Background

The literature review for this study was done adopting a mixed strategy, combining systematic searches in databases such as Scopus with AI-supported review tools like Elicit.ai and Research Rabbit. Over 500 papers were screened, leading to the identification of 35 high- and medium-relevance works grouped into three key clusters: (1) NLP and LLM-based rule interpretation, (2) semantic and ontology reasoning, and (3) early-stage computational design methods. High-relevance studies directly informed the methodology by clarifying how hybrid AI pipelines can balance flexibility and accuracy. Medium-relevance papers provided insight into domain-specific implementations and performance metrics.

Early Automated Code Compliance (ACC) systems have been built around the idea of translating building regulations into predefined rule sets that could be checked against structured BIM data. These early "black-box" tools, such as CORENET ePlanCheck in Singapore and Solibri Model Checker, were pioneering in demonstrating that parts of the regulatory review process could be automated (Dimyadi & Amor, 2013). They worked effectively in environments where detailed BIM models already existed, allowing rules to be applied directly to well-defined geometry and attributes. However, their logic was closed to users, their outputs were often non-transparent, and their dependency on fully modelled BIM data meant that they could only operate late in the design process (Beach et al., 2020).

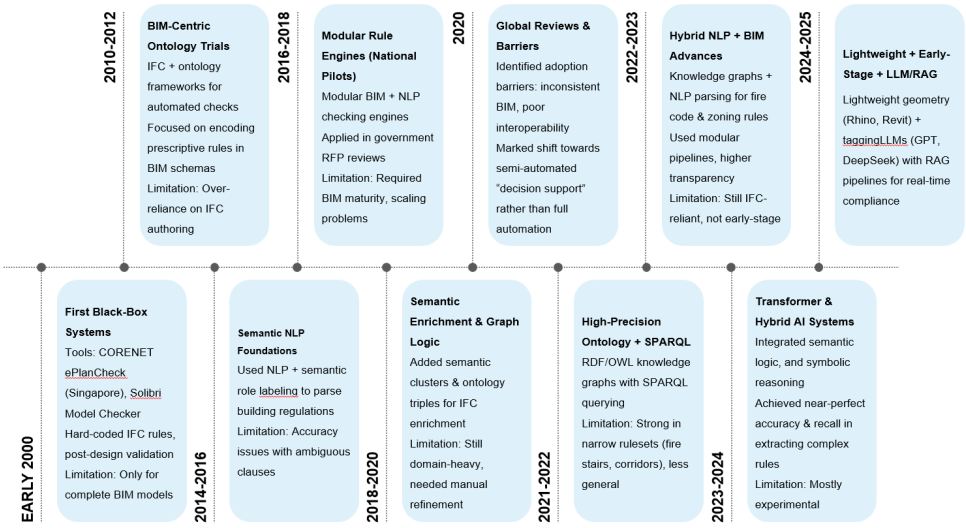

Figure 1. Timeline of Automated Code Compliance

The limitations of black-box systems led researchers to explore more transparent "white-box" approaches, allowing rules to be editable, traceable, and understandable by both designers and regulators. This new generation of systems relied heavily on Natural Language Processing (NLP) and semantic reasoning, translating unstructured building codes into structured, machine-interpretable formats. Studies such as Zhang and El-Gohary (2014, 2016) demonstrated how linguistic analysis, part-of-speech tagging, and syntactic parsing could be used to extract conditions, thresholds, and exceptions from legal texts. Unlike earlier rule-based engines, these white-box models allowed human oversight, enabling designers to understand the reasoning behind compliance outcomes and adjust rules for specific contexts.

Alongside NLP, ontology-based and graph-driven reasoning became another major research trajectory. Ontologies encode domain knowledge in structured relationships linking objects, properties, and constraints so that computers can infer meaning rather than just match keywords. Systems built on frameworks such as OWL, RDF, and

SHACL allowed rules to be represented as interconnected triples supporting flexible and explainable logic. Research by Guo et al. (2021) and Peng and Liu (2023) highlighted that these semantic models enabled greater interoperability between regulatory knowledge and BIM environments. Designers could query rules dynamically, and compliance engines could reason across multiple clauses simultaneously. Ontology-driven systems also proved scalable: new regulations could be added simply by extending the graph structure rather than reprogramming the rule engine.

The rapid progress of machine learning and transformer-based NLP models over the past five years has significantly expanded the potential of these methods. Large Language Models (LLMs) such as GPT and BERT can understand context, syntax, and semantics in complex regulatory text, allowing rules to be extracted with less manual annotation. Zhang and El-Gohary (2023) and Wu et al. (2023) showed that when these models are combined with symbolic reasoning or ontology layers, they can achieve over 90% accuracy in interpreting rule-based conditions. Moreover, frameworks such as Retrieval-Augmented Generation (RAG) make it possible to ground LLM outputs in verified sources, reducing the risk of hallucinated or incorrect interpretations.

A growing body of research focuses on applying these techniques in design environments. While most academic prototypes have been developed for structured BIM platforms like Revit or ArchiCAD, there is increasing recognition that early-stage design requires a flexible approach. Unlike BIM-based tools that require detailed object hierarchies, Rhino-Grasshopper lightweight environments prioritise flexibility, rapid iteration, and visual feedback. Studies by Rampini and Cecconi (2022) and Seong and Shin (2024) have argued that early integration of compliance intelligence into these flexible environments can help designers explore regulatory constraints creatively rather than treating them as limitations imposed later.

Most ACC systems have been developed in high-BIM economies, where data standards, regulatory databases, and interoperability protocols are well established. In contrast, developing regions often face fragmented documentation, inconsistent code interpretation, and limited computational infrastructure. Within this context, the Bangladesh National Building Code (BNBC 2020) presents a unique case. It is extensive and detailed, covering aspects from zoning and fire safety to structural and environmental requirements, yet it remains entirely text-based with no machine-readable structure. Researchers such as Hossain and Ahmed (2022) have identified this lack of digitalisation as one of the main barriers to advancing automation in the country's AEC industry.

Addressing BNBC through automation poses two intertwined challenges: interpreting the text accurately and delivering feedback through tools that local practitioners already use. A fully BIM-based approach would be impractical, given the low level of BIM penetration in Bangladesh's professional ecosystem (Shanto et al., 2024). Instead, a lightweight AI-assisted workflow that can interpret text, reason over extracted rules, and interact with accessible modelling tools represents a more realistic pathway.

This layered review confirmed that while research on automated compliance has matured considerably, there remains a consistent gap at the intersection of AI

reasoning, human-centred design, and low-resource application contexts. In summary, the evolution of automated compliance research shows a clear progression from closed, BIM-dependent black-box systems to transparent, modular, and increasingly intelligent frameworks capable of handling unstructured data. Despite of these advances, there is still limited exploration of how such technologies can be embedded into the creative, ambiguous early stages of design. The present research addresses this gap by proposing an LLM-integrated, human-centred system that operates directly within early design tools. By doing so, it contributes not only to the field of computational compliance but also to a broader agenda of making automation more inclusive, explainable, and globally relevant.

3. Methodology

This study developed a prototype that connects textual building code, AI-based reasoning, and geometric modelling to provide real-time compliance feedback during early-stage design. The goal was to demonstrate how language models and lightweight computational tools can bridge the gap between written regulations and spatial design exploration. The workflow was organised around three connected layers: first, the preparation of data from the Bangladesh National Building Code (BNBC 2020); second, AI reasoning and validation using a LangGraph pipeline; and third, visual feedback within the Rhino and Grasshopper environment.

3.1. SCOPE AND CORPUS PREPARATION

BNBC 2020 served as the primary dataset for this research. The full document exceeds 1,600 pages and covers structural, fire, and environmental regulations. To maintain focus and technical feasibility, the study targeted a 70-page subset containing early design parameters such as floor area ratio (FAR), maximum ground coverage (MGC), minimum setbacks, room size requirements, and fire-stair provisions.

The selected text was manually cleaned and converted from PDF to editable DOCX format. Redundant formatting, cross-references, and non-text elements were removed to ensure accurate parsing. Rule tables and numeric values were manually verified against the printed BNBC to preserve accuracy. To facilitate AI retrieval, the text corpus was divided into logical units based on semantic boundaries like sections, tables, and definitions. This structure ensured that each chunk could be meaningfully embedded without fragmenting context.

3.2. SYSTEM ARCHITECTURE OVERVIEW

The prototype architecture integrates three interconnected modules: (a) a text and embedding layer, (b) an AI reasoning and compliance engine and (c) a 3D modelling and feedback interface. Together, these modules allow regulatory clauses to be read, interpreted, and applied directly to geometric models in real time.

Figure 2 illustrates this architecture. On the text side, the pre-processed BNBC corpus was embedded into a Chroma vector store using OpenAI embedding models. This enabled contextual retrieval of code segments when a user query or system check was initiated. The AI reasoning layer, built in Python, operated through a LangGraph pipeline, which organised LLM calls into discrete, reusable nodes. It determined the

query type, retrieved relevant rule segments, built a retrieval-augmented (RAG) prompt, and then produced an interpretable output through OpenAI's GPT model.

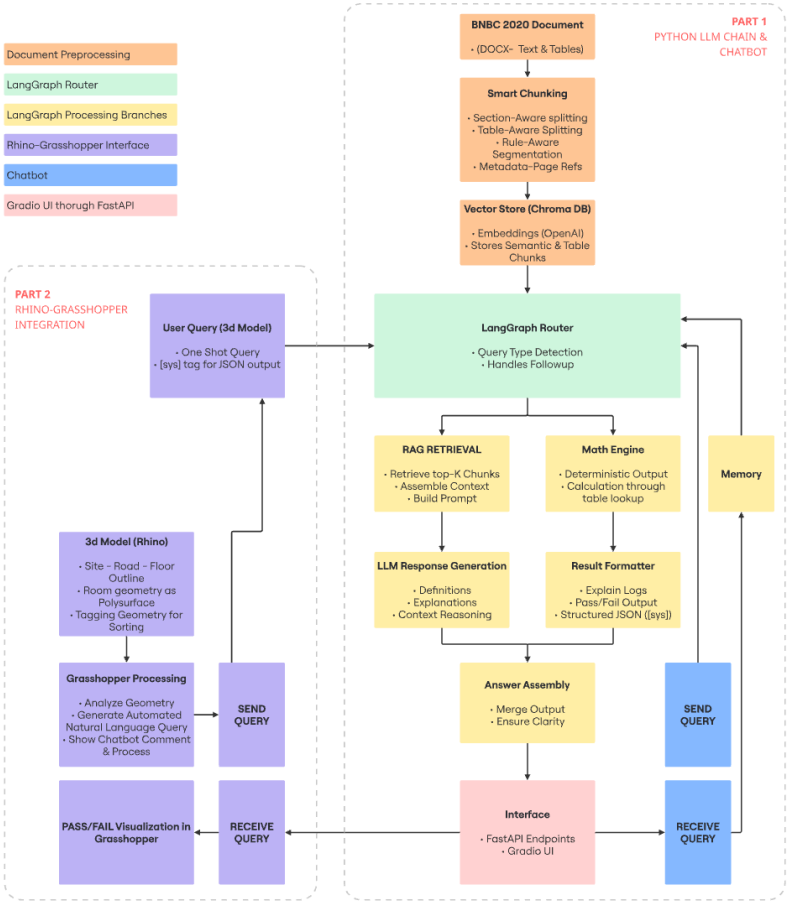

Figure 2. System architecture of AI reasoning, deterministic rule, and Rhino integration

3.3. LANGGRAPH REASONING PIPELINE

At the core of the AI reasoning process lies the LangGraph orchestration system, which structures LLM interactions as modular state transitions. When a query was received, the router node first identified its intent. The system is designed to recognise three categories: definition, explanation, and math. Definition queries requested textual meanings from the code (for example, "What is a setback?"). Explanation queries sought reasoning behind a rule or contextual understanding (e.g., "Why is minimum open space required?"). Math queries triggered quantitative validation (e.g., "Check if proposed FAR meets regulation").

For non-numeric tasks, the RAG retrieval node located the top relevant text chunks from the BNBC vector store and assembled them into a contextual prompt. The

call_llm node generated responses using GPT, while the clarifier node refined the wording for clarity. For numeric tasks, the math_orchestrator triggered the deterministic rule engine and compared design inputs against BNBC thresholds. Both paths converged in the compliance_checker, which merged outputs and formatted them into structured JSON ready for display.

3.4. RHINO–GRASSHOPPER INTEGRATION AND FEEDBACK

The system's interactive component was implemented in Rhino–Grasshopper, where designers could generate, modify, and test conceptual massing models. Each building element such as site boundary, floor footprint, or building height was tagged with metadata required for compliance checking. A Python script embedded in Grasshopper analysed this geometry and automatically constructed a natural-language query to send through FastAPI.

The communication layer ensured that both manual and automated queries were possible. For example, a user could either type a question in the Gradio chatbot interface (e.g., "Check if this building exceeds maximum FAR") or allow the script to trigger a predefined check marked by the "[sys]" tag for JSON output. Once processed, the AI pipeline returned a structured response including quantitative values, compliance status, and explanatory notes.

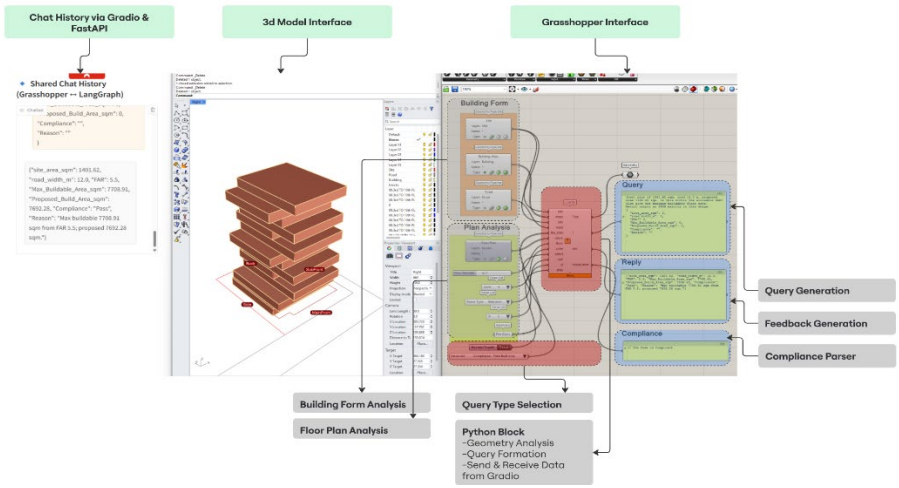

Figure 3. Integrated Gradio–Rhino–Grasshopper interface showing real-time compliance feedback

Visual feedback appeared immediately in the Grasshopper interface. Each building form or surface was colour-coded based on compliance status green for compliant, red for violation. Simultaneously, the chat interface displayed a text explanation outlining the reasoning behind the result, ensuring transparency and interpretability. As shown in Figure 3, the prototype integrates multiple views: the Gradio chatbot, the Rhino 3D model, and the Grasshopper logic blocks. Together they form a closed feedback loop between textual reasoning and visual modelling.

3.5. RESULTS

The performance of the proposed prototype was evaluated through a series of tests designed to measure both its quantitative accuracy and qualitative usability. The assessment focused on how effectively the system could interpret and validate building code clauses from the BNBC 2020. Testing covered three categories of chatbot interaction: definition, explanation, and math reasoning and one deterministic system mode that executed direct numerical checks. Each category was assessed for precision, recall, and F1 score. In total, 100 test cases were analysed using different context.

The first was the chatbot mode, which allowed users to query the system using natural language through the Gradio interface. The second was the system mode, which processed geometry-based inputs through Grasshopper using structured JSON queries. This dual setup enabled both qualitative and quantitative analysis capturing the model's semantic accuracy and its behaviour within an interactive design workflow.

Table 1. Precision, Recall, and F1 Scores Across Chatbot and System Mode Categories

Category	Pass (TP)	Fail (FN)	Precision	Recall	F1 Score
Chatbot - Definition	16	4	1	0.8	0.889
Chatbot - Explanation	12	3	1	0.8	0.889
Chatbot - Math	34	3	1	0.919	0.958
System Mode - Math	28	0	1	1	1
Total Result	90	10	1	0.9	0.947

For semantic and interpretive tasks, the system was tested on its ability to define and explain BNBC clauses using the 'Chatbot' interface. As shown in Table 1, the 'Chatbot - Definition' queries (16/20 correct) and 'Chatbot - Explanation' queries (12/15 correct) both achieved high F1 scores of 0.889. The perfect 1.0 precision score is critical, as it confirms the system did not "hallucinate" or provide incorrect information. The minor recall gap was not due to reasoning errors, but retrieval failures on ambiguous clauses or router misclassification known challenge in RAG pipelines.

The system's numeric validation performance was near-perfect. The deterministic 'System Mode - Math' bypasses the LLM for geometric checks and achieved a perfect 1.0 F1 score across all 28 trials. This confirms its reliability for automated validation. This accuracy was also reflected in the 'Chatbot - Math' queries (0.958 F1), where the LLM successfully interpreted textual inputs (e.g., "Check FAR for a 12m road") and routed them to the correct deterministic checks. While focused on residential standards, the 100-test validation serves as a transferable baseline; the system's tool-agnostic nature allows scaling across diverse occupancies (e.g., healthcare or assembly) by swapping underlying CSV rule tables.

Finally, the "human-centred" workflow was validated within the Rhino-Grasshopper interface. The system provided colour-coded feedback on the 3D model, with an average query time of less than two seconds, representing a significant efficiency gain over traditional manual code lookup and area-approximation methods. This visual loop, supported by textual explanations in Gradio, allowed designers to understand why a design failed and iterate accordingly. This dual-layer feedback was intuitive and made compliance a learnable, interactive part of the design process.

4. Discussion

The results show that combining language model reasoning with deterministic validation effectively supports early-stage building code compliance. The AI layer accurately interpreted BNBC 2020 text, while the deterministic engine produced precise checks for FAR, setbacks, and room size, creating a balanced system.

Overall performance was strong, with an average F1 score of 0.9 and perfect deterministic accuracy. Chatbot mode was useful for design exploration, helping architects understand the reasoning behind rules. Limitations were identified, primarily retrieval errors on ambiguous clauses and geometric mismatches with irregular footprints. To mitigate retrieval errors in ambiguous clauses, future iterations will implement semantic reranking and knowledge-graph grounding, replacing keyword-based heuristics with triplet-based logic to ensure consistent grounding.

5. Conclusion and Future Work

This study prototyped an AI system for early-stage building code compliance, linking unstructured BNBC 2020 text with Rhino–Grasshopper modelling. Combining LLM reasoning with deterministic validation, it provided real-time feedback on massing decisions without requiring full BIM models.

The system showed strong accuracy. The LLM effectively extracted and explained code clauses, while the deterministic engine delivered precise numeric checks. An average F1 score of 0.9 and perfect deterministic accuracy confirmed its reliability. Visual feedback in Grasshopper made compliance an interactive process, helping designers understand rule logic.

Future work will focus on automating the knowledge extraction pipeline. We aim to convert unstructured textual data directly into a knowledge graph. This will enable the faster and more accurate expansion of deterministic reasoning rules, improving both scalability and maintainability. Also, in collaboration with other researchers, there is a plan to extend the deterministic engine to structural and MEP components, leveraging the framework's modular architecture to accommodate broader BNBC chapters.

Acknowledgements

The authors would like to thank Professor Wassim Jabi, Dr Simon Lannon and the Welsh School of Architecture, Cardiff University, for their continued guidance. The authors also express their gratitude to family members, colleagues and especially Deepali for their encouragement and feedback throughout the process.

Attribution Statement

AI tools such as ChatGPT and QuillBot were used responsibly for paraphrasing, structural refinement, and language clarity in accordance with CAADRIA ethical guidelines. No generative system produced or altered research data, results, or interpretations. All coding, analysis, and evaluations were conducted and verified by the authors. All uses of AI have been clearly disclosed to ensure integrity and accountability in the research process.

References

Beach, T. H., Hippolyte, J.-L., & Rezgui, Y. (2020). Towards the adoption of automated regulatory compliance checking in the built environment. *Automation in Construction*, 118, 103285. https://doi.org/10.1016/j.autcon.2020.103285

Dimyadi, J., & Amor, R. (2013). Automated Building Code Compliance Checking – Where is it at? https://www.semanticscholar.org/paper/Automated-Building-Code-Compliance-Checking-%E2%80%93-Where-Dimyadi-Amor/3e6e4b32759797aca585b1e8e6034e8f399ab654

Guo, D., Onstein, E., & Rosa, A. D. L. (2021). A Semantic Approach for Automated Rule Compliance Checking in Construction Industry. *IEEE Access*, 9, 129648–129660. https://doi.org/10.1109/access.2021.3108226

Hossain, M. M., & Ahmed, S. (2022). Developing an automated safety checking system using BIM: A case study in the Bangladeshi construction industry. *International Journal of Construction Management*, 22(7), 1206–1224. https://doi.org/10.1080/15623599.2019.1686833

Jeongmin, S., & Sangyun, S. (2024). Implementation of an Automated Code Checking Algorithm Based on Site Analysis. Buildings, 14(6), 1654. https://doi.org/10.3390/buildings14061654

Peng, J., & Liu, X. (2023). Automated code compliance checking research based on BIM and knowledge graph. *Scientific Reports*, 13(1). https://doi.org/10.1038/s41598-023-34342-1

Rampini, L., & Cecconi, F. R. (2022). Artificial intelligence in construction asset management: A review of present status, challenges and future opportunities. *Journal of Information Technology in Construction (ITcon)*, 27(43), 884–913. https://doi.org/10.36680/j.itcon.2022.043

Shanto, A. A., Ananta, A. F., Rahman, M., & Manjur, K. A. (2024). Bim in Bangladesh's Education System and Construction Industry: Adaptability And Benefits in A Developing Country Context. 314–332. https://doi.org/10.2991/978-94-6463-478-5_24

Wu, J., Xue, X., & Zhang, J. (2023). Invariant Signature, Logic Reasoning, and Semantic Natural Language Processing (NLP)-Based Automated Building Code Compliance Checking (I-SNACC) Framework. *Journal of Information Technology in Construction*, 28, 1–18. https://doi.org/10.36680/j.itcon.2023.001

Zhang, J., & El-Gohary, N. M. (2014). Extending Building Information Models Semi-Automatically Using Semantic Natural Language Processing Techniques. *Computing in Civil and Building Engineering (2014),* 2246–2253. https://doi.org/10.1061/9780784413616.279

Zhang, J., & El-Gohary, N. M. (2016). Semantic NLP-Based Information Extraction from Construction Regulatory Documents for Automated Compliance Checking. *Journal of Computing in Civil Engineering,* 30(2), 04015014. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000346

Zhang, R., & El-Gohary, N. (2023). Transformer-based approach for automated context-aware IFC-regulation semantic information alignment. *Automation in Construction*, 145, 104540. https://doi.org/10.1016/j.autcon.2022.104540