buildings

MDPI

Article

An Intelligent Natural Language Processing (NLP) Workflow for Automated Smart Building Design

Ebere Donatus Okonta [1], Francis Ogochukwu Okeke [2,3,*], Emeka Ebuz Mgbemena [4], Rosemary Chidimma Nnaemeka-Okeke [5], Shuang Guo [6], Foluso Charles Awe [7] and Chinedu Eke [8]

1 School of Computing, Engineering and Digital Technologies, Teesside University, Middlesbrough TS1 3BX, UK
2 School of Engineering, Technology and Design, Canterbury Christ Church University, Canterbury CT1 1QU, UK
3 Design, Surveying and Planning, East Kent College, Canterbury CT1 3AJ, UK
4 Department of Architecture, Obafemi Awolowo University, Ile-Ife 220005, Nigeria
5 Department of Architecture, University of Nigeria, Enugu 400241, Nigeria
6 Christ Church Business School, Canterbury Christ Church University, Canterbury CT1 1QU, UK
7 Department of Architecture, Federal University Oye-Ekiti, Oye-Ekiti 371104, Nigeria
8 Department of Accounting, Economics and Finance, Canterbury Christ Church University, Canterbury CT1 1QU, UK
* Correspondence: f.okeke142@canterbury.ac.uk

Abstract

The automation of smart building design processes remains a significant challenge, particularly in translating complex natural language requirements into structured design parameters within Computer-Aided Design (CAD) environments. Traditional design workflows rely heavily on manual input, which can be inefficient, error-prone, and time-consuming, limiting the integration of adaptive, real-time inputs. To address this issue, this study proposes an intelligent Natural Language Processing (NLP)-based workflow for automating the conversion of design briefs into CAD-readable parameters. This study proposes a five-step integration framework that utilizes NLP to extract key design requirements from unstructured inputs such as emails and textual descriptions. The framework then identifies optimal integration points—such as APIs, direct database connections, or plugin-based solutions—to ensure seamless adaptability across various CAD systems. The implementation of this workflow has the potential to enable the automation of routine design tasks, reducing the reliance on manual data entry and enhancing efficiency. The key findings demonstrate that the proposed NLP-based approach may significantly streamline the design process, minimize human intervention while maintaining accuracy and adaptability. By integrating NLP with CAD environments, this study contributes to advancing intelligent design automation, ultimately supporting more efficient, cost-effective, and scalable smart building development. These findings highlight the potential of NLP to bridge the gap between human input and machine-readable data, providing a transformative solution for the architectural and construction industries.

Keywords: Natural Language Processing (NLP); smart building; automation; CAD integration; design workflows; Building Information Modeling (BIM)

check for updates

Academic Editors: Li Yang and Pejman Sharafi

Received: 11 May 2025
Revised: 18 June 2025
Accepted: 4 July 2025
Published: 9 July 2025

Citation: Okonta, E.D.; Okeke, F.O.; Mgbemena, E.E.; Nnaemeka-Okeke, R.C.; Guo, S.; Awe, F.C.; Eke, C. An Intelligent Natural Language Processing (NLP) Workflow for Automated Smart Building Design. *Buildings* **2025**, *15*, 2413. https://doi.org/10.3390/buildings15142413

1. Introduction

With growing concerns over climate change, urbanization, and housing shortages, the quest for sustainable building solutions is more pressing than ever [1]. Therefore, the

Architecture, Engineering, and Construction (AEC) industry faces mounting pressure to deliver buildings that not only meet stringent environmental standards but also adapt intelligently to changing conditions and user needs [2]. This necessitates advanced tools capable of integrating diverse inputs from energy modeling to occupant behavior patterns and optimizing designs accordingly. This is making the traditional linear approach to building design increasingly inadequate for addressing these multifaceted challenges, though the adoption of BIM technology presents promising potential [3].

As urban environments grow to be complex and data-driven, the demand for smart building solutions that enhance energy efficiency, occupant comfort, and real-time adaptability has risen significantly [4]. Smart building relies on intricate designs that incorporate dynamic data inputs and digital technologies, but current design processes in Computer-Aided Design (CAD) systems are often inefficient, requiring significant manual input and repetitive adjustments [5]. This creates a bottleneck in translating complex, natural language requirements into structured design outputs that meet the technical specifications of Building Information Modeling (BIM) and CAD environments. Consequently, there is a critical need for automation that can bridge the gap between human input and machine-readable data in the context of smart building design. Thus, there is a need to automate design inputs into CAD environments essential for the development of smart buildings.

Automating the interpretation of client design briefs enabling seamless processing and direct transmission into CAD softwares has the potential to significantly enhance productivity, accelerate project timelines, and improve design accuracy. In a time when companies, researchers, and professionals are prioritizing efficiency, speed, and profitability, such automation is not just advantageous but essential. It streamlines the design process, reduces manual workload, minimizes errors, and facilitates the faster delivery of smart building projects. Natural Language Processing (NLP), a field of artificial intelligence dedicated to understanding and generating human language, offers promising capabilities for automating this gap. By processing design requirements expressed in natural language, NLP can enable CAD systems to interpret high-level design goals and update models accordingly, thus eliminating manual translation and entry. Although NLP has been explored in various fields for its automation potential, its integration into smart building workflows remains underdeveloped, particularly in frameworks that support adaptive, real-time design inputs and flexible CAD integration.

This automation revolution represents a fundamental shift in architectural design methodology. Traditional approaches rely heavily on designers manually interpreting client requirements and translating them into technical specifications—a process fraught with misinterpretations, inconsistencies, and time-consuming revisions [6]. By contrast, NLP-powered automation creates a direct pipeline from client vision to technical execution, preserving the integrity of design intent throughout the process. Its potential applications extend beyond the initial design phases. Throughout a project's lifecycle, requirements frequently evolve in response to changing client needs, regulatory updates, budget constraints, and site conditions [7]. An NLP-enabled system could continuously process these modifications, updating the design parameters in real-time and maintaining alignment between stakeholder expectations and design outputs. Furthermore, such automation democratizes the design process by enabling users with limited technical knowledge to communicate effectively with sophisticated design systems [8]. This accessibility broadens participation in the design conversation, potentially leading to more innovative and user-centered buildings. Non-technical stakeholders could articulate their needs in familiar language, with the system handling the complex translation into design specifications.

NLP has been used in construction to extract information from text, such as improving BIM queries and automating regulation compliance checks. However, these efforts focus on data retrieval and rule checking, not on translating design briefs into CAD parameters. Therefore, the current research addresses this gap by proposing a five-step integration framework that connects NLP tools with CAD systems to interpret and implement design requirements directly from natural language inputs. By analyzing integration points such as APIs, database connections, and plugins, the framework provides a flexible solution for incorporating NLP into CAD environments, allowing for greater adaptability across different software architectures. The central question the study seeks to address is as follows: How can NLP be effectively integrated with CAD systems to automate the translation of unstructured design briefs into structured smart building design parameters? The findings of this research contribute to the field of smart building design by introducing a structured, automated workflow that enhances the efficiency and accuracy of design updates. This framework not only addresses the current limitations of manual CAD workflows but also advances intelligent design automation, offering a scalable approach that can be adopted by practitioners in the smart building industry to reduce the time and cost associated with design modification.

2. Related Works

NLP workflow can improve specific aspects of smart building design, for example, applications like chatbots, virtual assistants, voice commands, and data analytics all make extensive use of NLP [9]. However, NLP techniques are essential for improving building automation management, user comfort, and occupant comfort when integrated into smart building designs [10]. This is very achievable through data collection, processing, and analysis, and response production, which are all part of the NLP workflow [11]. For example, voice-controlled automation in smart buildings uses voice recognition systems driven by NLP workflow to control devices like Alexa and Google Assistant that control features like lighting, security settings, heating, ventilation, and air conditioning [12,13]. This might mean that by evaluating speech patterns, identifying intent, and carrying out the necessary steps, an NLP workflow guarantees that voice commands are correctly processed, resulting in a flawless user experience. Furthermore, NLP-powered systems assess occupant feedback using sensor data and usage patterns to optimize energy consumption in smart buildings [14]. Google Nest Thermostat and the IBM Watson IoT for smart building management exemplify this. As a result, NLP-powered systems provide smart climate control systems within indoor settings of smart buildings, as well as sustainability insights via IoT sensors and smart meters, allowing for automated energy management while maintaining occupant comfort and sustainability [15]. Furthermore, improved maintenance management enables predictive maintenance in smart buildings by examining prior maintenance logs to predict when equipment, such as lifts or HVAC systems, will need to be serviced, decreasing downtime and repair costs [16].

The adoption of advanced technologies in the construction and design engineering fields has attracted considerable attention, with many studies aiming to streamline operations and boost productivity through automation and data-centric strategies [17–19]. NLP has gained prominence as a valuable tool for automating the extraction of information from unstructured text and technical documents [20,21]. Research has shown that NLP has the potential to significantly enhance construction management and design workflows. For example, Wu et al. [22] and Di Giuda et al. [23] showcased how NLP can automate project documentation and improve the efficiency of information retrieval in construction tasks. Similarly, Nabavi et al. [24] investigated the use of NLP for automated information queries, emphasizing its role in supporting better decision-making through timely and precise insights. Other studies, such as those by Gupta et al. [25] and Mahlawi and Sasi [26], examined methods for extracting data from emails to optimize communication and documentation processes within projects. Their findings highlight the challenges posed by varying data formats and the critical need for effective pre-processing steps—like tokenization and feature engineering—to ensure reliable information extraction. There is also an expanding body of literature comparing the performance of various NLP tools and approaches. For instance, Schmitt et al. [27] carried out a comparative analysis of several NLP libraries—such as SpaCy, NLTK, Stanford NLP, and Gate—to assess their effectiveness across different text analysis tasks. Their results indicated that the selection of specific tools and methodologies plays a critical role in influencing the accuracy and efficiency of data extraction. Although notable progress has been made in applying NLP within construction and design engineering, significant research gaps still exist. Many current studies are limited to small datasets or controlled settings [28–30], and there is a notable shortage of research focused on real-world scalability and implementation. In particular, automating data extraction from textual inputs—such as design briefs found in client emails—and integrating that data into CAD systems to generate technical drawings remains underexplored. This paper seeks to address these shortcomings by proposing an NLP-based framework designed to automatically extract design information from client email repositories and seamlessly incorporate it into design software.

3. Methodology

This section outlines the methodological approach taken to develop a novel framework for automating smart building design workflows. The core of this methodology lies in integrating NLP tools with CAD systems to bridge the gap between unstructured client requirements and structured design parameters. Through a narrative literature review, the study explores a five-step framework designed to streamline the interpretation and application of design briefs within CAD environments, aiming to enhance efficiency and reduce manual translation efforts. While this study primarily focuses on establishing a conceptual and architectural blueprint for this integration, it is grounded in technically feasible technologies and lays the foundation for future empirical validation and software development.

3.1. Proposed Framework Development

The proposed framework for automating smart building design workflows was developed through a comprehensive review of the existing literature, industry practices, and technological capabilities in the fields of NLP, BIM, and CAD. The framework was conceptualized to address the current gaps in automating the interpretation and application of unstructured design requirements within CAD environments. The development process involved analyzing the workflows of architects and building designers, identifying points where the manual translation of client briefs into design parameters creates inefficiencies, and exploring how NLP tools could streamline this process. This study proposes a five-step framework for automating smart building design workflows through the integration of NLP tools with CAD systems. The methodology focuses on developing an intelligent workflow that can process natural language inputs and seamlessly translate them into actionable design parameters within CAD software. The framework consists of the steps in Table 1, which outlines a five-step framework for integrating NLP with CAD systems to automate design workflows. Each step includes specific objectives, methods, tools, data sources, and rationales. The process begins with data extraction from unstructured inputs (Step 1) and proceeds to develop an integration framework linking NLP outputs with CAD systems (Step 2); automated design updates are then implemented (Step 3), followed by testing and validation to ensure accuracy and usability (Step 4). Finally, a feedback loop is implemented to facilitate iterative refinements for continuous improvement (Step 5).

Table 1. Proposed methods for the five-step framework.

Step	Objective	Methods/Procedures	Tools/Techniques	Data Collection	Rationale
1. Data Extraction and Processing	Extract relevant design requirements from natural language inputs.	- Collect unstructured text inputs (e.g., client briefs, emails). - Use NLP tools to parse text and identify key design parameters. - Structure extracted data in machine-readable formats (JSON, XML).	NLP libraries: spaCy, NLTK data structuring: JSON, XML	Sample project briefs, case studies, simulated client inputs	Provides foundational data required for all downstream processing.
2. Integration Framework Development	Develop seamless communication between NLP tools and CAD systems.	- Analyze CAD systems (e.g., Revit, AutoCAD) for integration points. - Develop APIs or plugins to connect NLP output with CAD input. - Implement middleware for data translation and command execution.	APIs (RESTful services), Autodesk Forge, Revit API, Python	CAD system documentation, API testing logs	Establishes the infrastructure to connect data extraction with design tools.
3. Automated Design Integration	Automate real-time updates in CAD models based on structured design inputs.	- Transmit structured data from NLP tools to CAD software. - Automate model updates based on received parameters. - Perform validation checks to ensure compliance with design standards and codes.	JSON data exchange Automated CAD scripts Compliance check routines	CAD model audit reports, compliance verification data	Translates integrated data into automated design actions in the CAD environment.
4. Testing and Validation	Test functionality and validate accuracy of system components and integration.	- Conduct unit tests on NLP modules. - Perform integration testing between NLP and CAD systems. - Facilitate user testing sessions with architects and designers. - Collect feedback on usability and effectiveness.	Unit and integration testing User testing Surveys/interviews	System performance metrics, user feedback surveys, testing logs	Verifies technical accuracy and collects feedback on system usability and performance.
5. Feedback Loop and Iteration	Refine system through user feedback and iterative development.	- Collect feedback through surveys and interviews. - Analyze feedback to identify areas for improvement. - Iteratively adjust NLP and CAD integration components. - Conduct multiple testing cycles for refinement.	Thematic analysis Iterative development Agile methodology	User feedback forms, performance improvement logs, iteration reports	Ensures continuous improvement and adaptation to evolving user needs and design scenarios.

This study is positioned as a conceptual and methodological proposal that introduces a structured workflow for integrating NLP techniques into CAD environments within the smart building design domain. The primary aim is to outline the architectural and technical steps required for such integration, identify potential automation pathways, and highlight the challenges involved—particularly around data translation, format standardization, and automation feasibility. While the framework is grounded in technically feasible technologies—such as spaCy-based NLP and Revit API-based model manipulation—it does not present a complete implementation, prototype, or benchmark system. Instead, the study focuses on establishing a generalizable, modular blueprint that can guide future empirical studies or software development efforts. Acknowledging this limitation, future work will involve the implementation of a repeatable proof-of-concept prototype using real-world client briefs, coupled with performance evaluation (e.g., accuracy, speed, error detection). This approach ensures that the current work remains focused on its contribution to theory, methodology, and workflow structure, laying the foundation for future empirical and technical validation.

3.2. CAD Environment and Tooling

For the conceptual design and prototype simulation of the NLP-CAD integration workflow, this study primarily utilized Autodesk Revit 2023 and AutoCAD 2023 (with .NET API). These platforms were chosen due to their widespread use in architectural design, extensive API ecosystems, and robust support for parameter-driven modeling, making them ideal for integrating structured design data extracted via NLP. Both versions offer full support for custom parameter definition and scripting through the Revit API, seamless integration with Python-based middleware (e.g., via Dynamo or Autodesk Forge), and the ability to receive model inputs via external APIs or plugin scripts. While the developed workflow is platform-agnostic in principle and could be adapted for other suitable platforms like Graphisoft ArchiCAD 26, Bentley OpenBuildings Designer, or BricsCAD BIM, it was optimized around Revit 2023 as the reference environment for API-based automation and visual demonstrations

4. Results and Discussion

This section presents the findings from the development of the proposed framework for automating smart building design workflows, focusing on the integration of NLP with CAD systems. This study elaborates on the framework's key components, discusses the identified integration points, and addresses critical aspects such as data format standardization and error handling. Furthermore, an illustrative workflow example is provided to demonstrate the practical application of the framework and its inherent limitations and the significant implications it holds for the future of intelligent building design.

4.1. Integration Framework Development

The NLP model, after it has been trained, needs to be integrated into the CAD software. This study identified and proposed five (5) steps towards the development of a framework that automates the integration process, as illustrated in Figure 1. The integration of NLP tools with CAD software begins with identifying integration points [31], determining where the NLP tool will fit into the existing CAD framework. This can be achieved through APIs, through direct database access, or as a standalone plugin. Understanding the BIM software's technical requirements, including its programming languages, frameworks, and data formats, is crucial to ensuring compatibility [32]. Users can input natural language descriptions of their design requirements, feedback, or modifications if the process is manual [33], and this, in turn, is communicated to the CAD software. The other option is the automated process, where the proposed NLP tool processes relevant design parameters and requirements from emails and communicates this process to the CAD software for automated design [34,35].

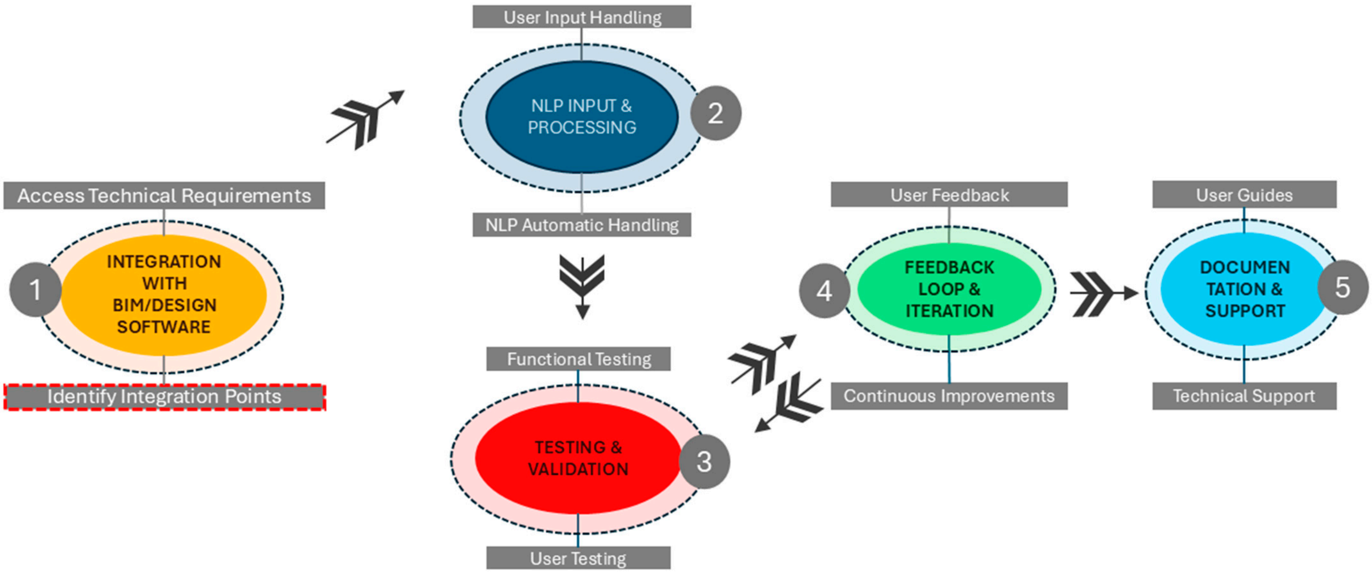

Figure 1. Proposed framework for automating (source: authors).

The third aspect of the framework is testing and validation, which cuts across functional testing to ensure that the NLP integration performs as expected across various scenarios and user inputs [36], and conducting user testing sessions to gather feedback on the usability and effectiveness of the NLP-enhanced CAD features, helping to identify areas for improvement and ensuring a seamless user experience [37].

Furthermore, a feedback loop and iteration steps are established, which allow users to review and provide feedback on the design recommendations or modifications suggested by the NLP tool [38]. This feedback is used for continuous improvement, refining the NLP models and integration processes to enhance accuracy and user satisfaction, consistent with the study by Mustansir et al. [39]. Lastly, the proposed framework captures comprehensive user guides developed to help users understand how to use the NLP-enhanced features within the CAD software. This includes step-by-step instructions for inputting design requirements, interacting with the system, and troubleshooting common issues. Additionally, technical support is provided to address any issues users may encounter, ensuring they can effectively utilize the NLP tool in their design processes [40].

Managing uncertainty and inconsistencies in real-world design inputs is a central challenge for intelligent automation in architecture, particularly due to the unpredictable and informal nature of client communication. The proposed NLP-CAD integration framework addresses this by handling ambiguity, conflicting requirements, and incomplete data through a combination of controlled vocabularies, conflict detection modules, and default parameter banks. Ambiguous terms are flagged and mapped to industry standards, while contradictory inputs are cross-checked with metadata to prevent faulty updates. Incomplete entries are marked as tentative, with confidence scores guiding manual reviews. The system incorporates human-in-the-loop verification, allowing designers to oversee and refine automated suggestions via a dedicated CAD interface. A feedback mechanism captures user interventions to improve the system's contextual understanding over time. These features collectively ensure the framework is not just a theoretical model but a scalable, adaptive tool suitable for live architectural environments where client needs are often fluid and imprecise.

4.2. Identifying the Integration Points

In a bid to identify the integration points, the study established three basic methods: the application programming interface (API), the direct database, and the use of a plugin. An API is a set of rules or protocols that enables software applications to communicate with each other to exchange data, features, and functionality [41,42]. Direct database access involves a direct connection between the application and the database, offering simplicity and control over queries but requiring more manual security management [43]. Plugins are extensions integrated with software or websites to increase the efficacy of an existing program, website, or software, providing additional tools, features, and functionalities to enhance the core capabilities [44,45]. In a bid to identify an integration point, Table 2 highlights the advantages and disadvantages of each method.

Table 2. Exploring integration points.

Integration Points	Advantages	Disadvantages
APIs (Application Programming Interfaces)	- Enhanced user interaction via an understanding of and responses to natural language queries. This is achieved as APIs create new connections between plugins and software not only through communication but also through database linkages towards enhanced project data extraction [46,47]. - The automation and efficiency of systems in data entry, workflow automation and report generation, establishing a connection between the software and database [46]. - Provide analytics and a contextual understanding and real-time monitoring.	- APIs can be discontinued or specifications changed, allowing for new corresponding adjustments in BIM software, potentially disrupting workflow [48]. - The possibility for APIs to experience latency, as well as rate limits, implies that real-time data synchronization or batch processing capabilities may be hampered [49,50]. - The complexity of integration also exists as the data mapping and transformation process requires an alignment between existing data structures and formats specific to each API variant [51,52].
Direct Database Access	- Provides real-time access and integration via immediate updates as well as compatibility [53]. - Provides avenues for data integrity and consistent supply [54]. - Allows for customization and extensibility via tailored workflows, plugin integration and scalability [55,56].	- The rapid growth of building components in the BIM object database increases the difficulty of the efficient querying of components that users require [57]. - The risk of data complexity is of paramount importance, as a high dependency on the database structure may lead to a need for corresponding BIM updates [58].
Plugins	- Enhanced Functionalities: Core CAD software now allows for the accurate translation of NLP techniques, offering specialized access to capabilities and tools that are not available out of the box [22]. - Workflow Optimization: Eliminating the requirement for manual data entry reduces the possibility of errors [46,59]. - Specialized tools for disciplines: catering to specialized information in terms of CAD through custom-built plugins is proven to drive productivity and efficiency towards cost and time savings [46]. - Improved collaboration among numerous project stakeholders to ensure seamless data interchange [60]. - Cost and time savings: lowering the time required for project rework or the project delivery timetable [46,61].	- Internally, some plugins utilize APIs to communicate with different services for development purposes [62]. - Compatibility issues may arise regarding model data size and unsuitability with early-stage BIM models. Also, compatibility discourse with regard to the lack of sustainability of CAD models exists [61]. - Due to a lack of regulatory enforcement and sectoral motivation, plugins, in most cases, are currently limited to supporting building certification rather than as a part of a design process [61]. - Poor and outdated plugins pose multiple risks, especially due to the fact that they are case-dependent. Hence, they possess an inability to develop 2D data in a specific case into more visualized data that is enhanced by a more practical viewing and searching criterion [63].

As illustrated in Figure 2, the framework allows for multiple integration strategies between NLP-generated design data and CAD environments. These approaches are grounded in real-world practices supported by leading CAD platforms such as Autodesk Revit, AutoCAD, and ArchiCAD. APIs offer the most direct route for automation, while plugin development provides customizable UI-level extensions. Database access supports batch operations and advanced querying in BIM workflows. Each pathway is practically implementable using well-documented SDKs, middleware tools, and scripting libraries, affirming the technical feasibility of the proposed integration model.

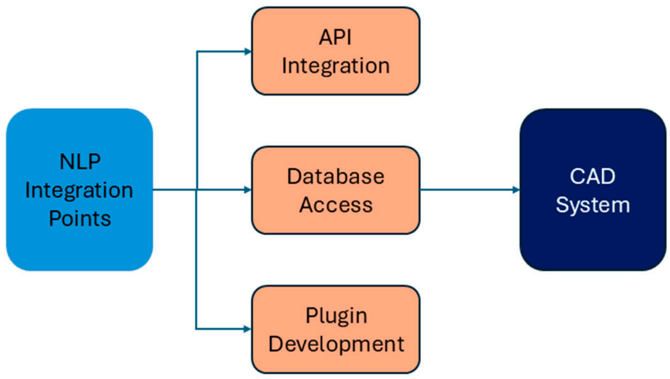

Figure 2. Integration feasibility of API, database, and plugin approaches (source: authors).

The best integration point largely depends on the specific project requirements. For automation and real-time monitoring, APIs are highly beneficial due to their ability to automate processes and provide real-time data analytics. For customization and real-time access, direct database access is optimal, offering immediate updates and extensive customization options. For enhanced functionalities and specialized tools, plugins excel by adding specialized capabilities and improving workflow efficiency. Given their pros and cons, APIs appear to be the most versatile option, offering significant benefits in terms of automation, real-time monitoring, and interaction enhancement. However, if real-time data access and extensive customization are paramount, direct database access might be the best choice. For projects requiring specialized tools and improved collaboration, plugins would be the preferred integration point.

4.3. Data Format Standardization in NLP-CAD Integration

One of the most critical challenges in automating the integration of NLP outputs into CAD environments lies in the standardization of data formats. While the proposed framework outlines a high-level integration strategy, its practical implementation requires the careful handling of format conversion, semantic alignment, and interoperability between systems.

Concerning semantic alignment in natural language, natural language inputs such as client emails or design briefs often contain subjective or vague descriptors (e.g., "spacious lobby", "sufficient lighting") that are not directly translatable into CAD parameters. These ambiguities complicate the creation of structured design data required by CAD systems. To address this, the framework proposes the use of controlled vocabularies and domain-specific ontologies that map ambiguous phrases to predefined design specifications. For example, "spacious" may be mapped to a minimum area threshold (e.g., $\geq$30 m^2), while "sufficient lighting" could reference specific lux levels or window-to-floor area ratios. These mappings help standardize terminology and improve the reliability of data extraction.

Also, regarding format incompatibility between NLP output and CAD input, NLP tools typically output structured data in formats such as JSON or XML, which must be further transformed into formats compatible with CAD software. However, different CAD platforms (e.g., Autodesk Revit, AutoCAD) have unique data schemas and parameter-handling protocols, making direct integration challenging. To solve this, this study's framework proposes that a middleware layer is implemented to bridge this gap. This layer acts as a translator between the NLP output and the CAD system input, converting JSON or XML data into native CAD scripts or API-compatible commands. Tools such as Autodesk Forge and Dynamo scripts can facilitate this transformation, ensuring consistent parameter interpretation across platforms.

Furthermore, with regard to the absence of a unified schema for design requirements, there is no universally accepted data schema for expressing design requirements in architectural workflows. This makes it difficult to generalize NLP outputs across different CAD environments. This can be mitigated by introducing a standardized intermediate data schema referred to as the *Design Parameter JSON (DPJ)* format. This schema can contain well-defined fields such as "space_type", "area", "insulation_type", "U_value", and "material", which can be dynamically populated by the NLP engine and consumed by the CAD integration module. The schema is extensible and adaptable to different project requirements and design stages.

Lastly, as concerns the issues of versioning and software interoperability, frequent updates in NLP libraries (e.g., spaCy, NLTK) and CAD APIs can disrupt interoperability. Changes in data structures, deprecations, or inconsistencies between library versions may affect data exchange processes. This study proposes schema versioning and backward-compatible API design. Each data payload includes a version identifier, allowing both NLP and CAD systems to validate compatibility before processing. In cases of incompatibility, the system generates warnings and logs discrepancies for manual review, ensuring system stability and traceability.

4.4. Integration Workflow Example

As illustrated in Figure 3, the goal is to automate CAD software based on design requirements received via email using email handling libraries and NLP. First, the system fetches emails from the server using specified Python libraries requiring authentication to access the inbox. For instance, a client might send an email with requirements such as "We need a flat roof with 0.60 U-Value. . ." Next, the email content is parsed using libraries like email, decoding the body and extracting relevant text for further processing. The extracted text is then analyzed using NLP tools like spaCy or NLTK to identify key design requirements, such as material type, U value, roof deck type, etc.

Following this, the extracted parameters are transmitted to the CAD software via an API call, typically in the form of a JSON payload. The CAD software processes the API request, updating the building model to reflect the new design requirements, which may involve recalculating structural loads and modifying design elements accordingly. Automated validation checks ensure that the updated design complies with structural standards, building codes, and the intended design objectives.

While this study proposes a structured pipeline to enable such automation, it is important to recognize that prior research has also explored the application of NLP in the construction domain, albeit with different objectives. For instance, Nabavi et al. [24] developed an AI-based framework that utilizes NLP techniques to facilitate information retrieval from BIM. Their approach employs a support vector machine (SVM) to classify user queries and leverages ontologies for semantic understanding, thereby enabling efficient access to project data. However, this research primarily focuses on querying existing BIM data, rather than transforming design briefs into CAD-executable parameters.

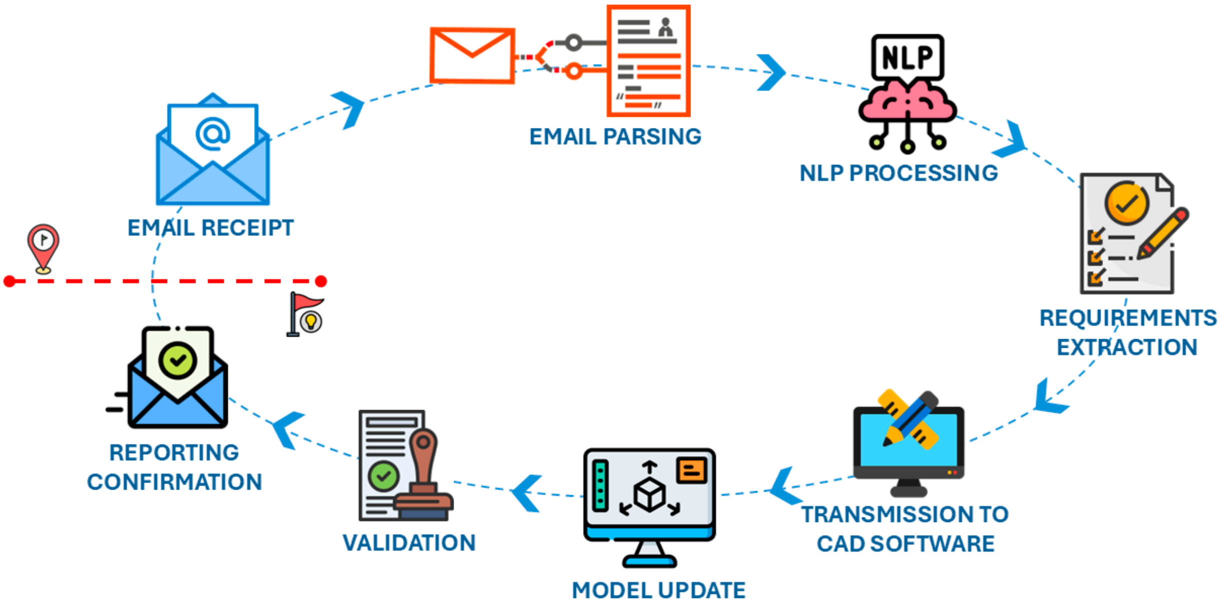

Figure 3. Automated integration workflow example (source: authors).

Similarly, Song et al. [64] applied NLP and deep learning techniques to analyze building regulations in support of automated rule-checking systems. Their method performs semantic analysis of regulatory texts to streamline compliance verification. While this work advances automation in compliance checking, it does not directly address the translation of informal design intent into actionable building elements within CAD environments.

4.5. Error Handling, Edge Cases, and Validation in the Integration Workflow

While Figure 3 in Section 4.4 illustrates a streamlined example of automating CAD updates using NLP-extracted design inputs, real-world implementation necessitates robust mechanisms to handle input variability, system errors, and output validation. For example, Error handling is critical across various stages of the NLP-CAD pipeline, including input parsing, entity recognition, data transmission, and CAD command execution, as seen in Figure 4. Strategies for managing errors include input validation filters that screen incoming text for unsupported file formats, missing content, or malformed data before NLP parsing; fallback rules that trigger human review or adjusted parsing logic when the system fails to extract sufficient design parameters; exception logging that records errors with timestamps, types, and affected components to support debugging and audits; and retries with timeouts, where API or database failures activate retry logic with exponential backoff to maintain reliability without overloading the system.

Edge case scenarios, such as contradictory requirements or unconventional design requests, can significantly affect automation and can be addressed through several mechanisms. Conflicting inputs can be managed by a conflict detection module that identifies incompatible parameters (e.g., a “flat roof” paired with a steep “pitch angle”) and prompts for clarification. Ambiguous descriptions, like “medium-sized hall” without quantitative context, trigger the use of predefined default values while flagging the instance for review. In cases of incomplete data, such as missing U-values or insulation types, the system can insert placeholders and mark the design entry as incomplete, prompting user intervention.

Validation ensures that the final output is technically sound, complies with building standards, and aligns with client intent. Implemented validation layers include semantic validation, which ensures that extracted parameters logically align (e.g., wall material matches building use); compliance checks, where automated scripts compare design parameters against local building codes or energy efficiency benchmarks; simulation or model verification, using BIM-capable CAD systems like Revit to simulate design updates in a sandbox environment before committing changes to the live model; and user-in-the-loop verification, which introduces a human validation stage for critical design changes, allowing designers to approve or reject automated updates.

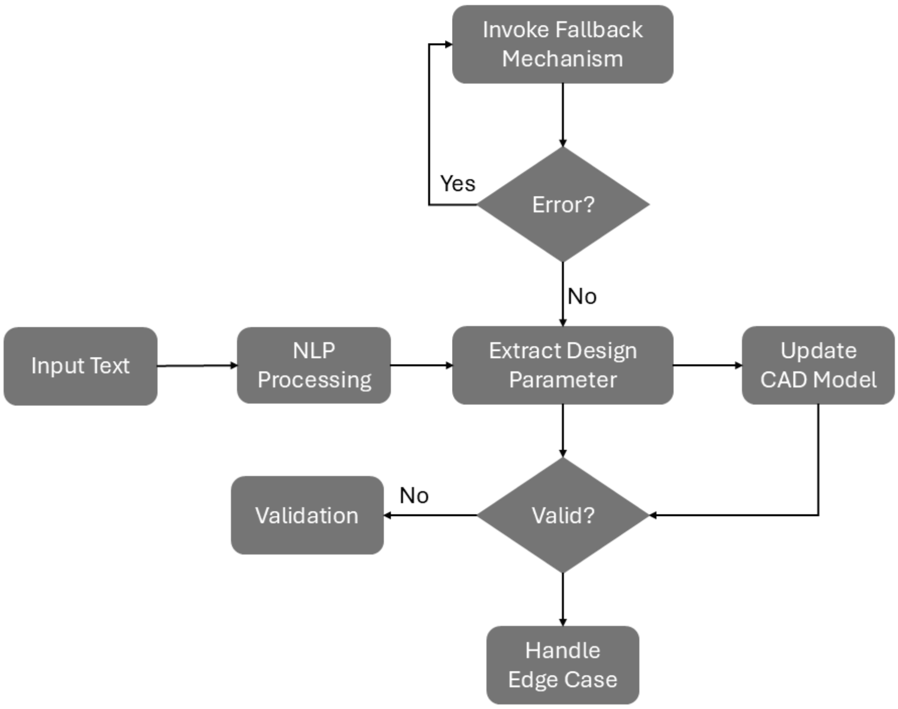

Figure 4. Error handling, edge cases, and validation in the integration workflow (source: authors).

4.6. Limitations and Significance of the Framework

While the proposed framework offers a novel approach to automating smart building design workflows, it has certain limitations. First, the accuracy of the NLP is highly dependent on the quality and consistency of the input data. Variations in client language, ambiguous descriptions, and industry-specific terminology may reduce the system's ability to correctly interpret design requirements without human intervention. Second, integration with CAD systems is constrained by the openness and flexibility of their APIs. Not all CAD platforms offer extensive API access, which may limit the framework's interoperability across different software environments. Additionally, the real-time automation of design modifications in complex building models may introduce challenges related to processing speed, system scalability, and compliance with building codes and standards.

While the proposed framework demonstrates clear conceptual advantages in reducing manual workload and minimizing human error, the author acknowledges the current limitation in providing quantitative validation of accuracy and error reduction. As this study focuses on the architectural design of the integration framework, the results presented remain primarily qualitative. Future research will implement a working prototype using real-world client briefs to evaluate performance metrics such as parameter extraction accuracy (e.g., precision, recall, F1-score), the update fidelity of CAD models, and the error frequency reduction in comparison to manual workflows. This will allow for empirical benchmarking and provide concrete evidence of system efficacy.

Despite its limitations, the proposed framework represents a significant advancement in the digital transformation of smart building design processes. By automating the translation of natural language design requirements into actionable CAD model updates, the framework reduces the reliance on manual data entry and interpretation, which are often time-consuming and prone to errors. This automation can enhance productivity, improve design accuracy, and facilitate more efficient collaboration between clients and design teams. In the context of smart buildings, this framework lays the groundwork for more intelligent design systems that are responsive to user needs, adaptive to evolving requirements, and capable of supporting sustainable and efficient building practices.

5. Conclusions

This study proposes an intelligent, NLP-based workflow designed to automate smart building design processes by integrating NLP tools with CAD systems. The study posits that the proposed five-step integration framework can successfully extract design parameters from unstructured inputs, such as client emails and textual descriptions, and seamlessly translate them into machine-readable formats. The study findings demonstrate that by evaluating different integration methods—APIs, direct database connections, and plugin-based solutions—APIs represent the most versatile approach due to their automation capabilities and real-time data processing. Additionally, the study posits that automating design updates within CAD software can significantly reduce manual workload, accelerate project timelines, and improve design accuracy by minimizing human error. The implementation of NLP-driven automation enhances workflow efficiency, making design processes more adaptive and scalable for real-world applications.

The significance of this research lies in its contribution to automating and enhancing smart building design workflows, offering an innovative solution for addressing the challenges of integrating dynamic, real-time data with traditional CAD environments. By reducing manual input and enabling automated design updates, the proposed framework paves the way for more adaptive, efficient, and scalable smart building design processes. This not only saves time but also improves the accuracy of design decisions, ensuring that the building design aligns with specified requirements and standards.

The study acknowledges several limitations that may impact the full realization of the proposed NLP-based workflow for automating smart building design processes. One key limitation is the dependency on the quality and consistency of input data, as variations in client language, ambiguous descriptions, and industry-specific terminology may affect the accuracy of NLP-based interpretation. Incorporating machine learning techniques into the workflow could allow the system to adapt to user preferences and continuously improve its responses based on past interactions. This would enable the NLP tool to evolve alongside the changing demands of smart building design and enhance its ability to suggest design improvements. Also, the study acknowledges that this is a framework proposal and not a case study; thus, future studies could explore the implementation of the proposed framework in live smart building projects to assess its performance in real-world conditions. Further research could also focus on improving the NLP models' accuracy, expanding the range of design parameters that can be processed, and addressing the challenges of cross-platform compatibility. Additionally, integrating more advanced machine learning techniques for continuous learning and improvement of the system's capabilities could further enhance the workflow's adaptability and effectiveness in smart building design. Future research will explore expert interviews to validate the feasibility and effectiveness of the proposed framework. A sample of industry professionals will provide insights into the practical applications of and challenges and potential improvements in the integration process. The expert feedback will be analyzed to assess the overall impact of NLP-enhanced design automation in smart building workflows.

Author Contributions: Conceptualization, Data Curation, Formal Analysis, Investigation, Writing the Original Draft, and Review and Editing of the Manuscript—E.D.O.; Investigation, Project Administration, and Supervision—E.E.M.; Formal Analysis, Investigation, Project Administration, Visualization, and Review and Editing of the Manuscript—F.O.O.; Review and Editing of the Manuscript, and Validation—R.C.N.-O.; Review and Editing of the Manuscript, and Validation—S.G.; Review and Editing of the manuscript, and Funding Acquisition—F.C.A.; Review and Editing of the manuscript, and Visualization—C.E. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The authors confirm that the data supporting the findings of this study are available within the article.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Okeke, F.O.; Ahmed, A.; Imam, A.; Hassanin, H. A review of corncob-based building materials as a sustainable solution for the building and construction industry. *Hybrid Adv.* **2024**, *6*, 100269. [CrossRef]
2. Okeke, F.O.; Ezema, E.C.; Ibem, E.O.; Sam-amobi, C.; Ahmed, A. Comparative analysis of the features of major green building rating tools (GBRTs): A systematic review. *Lect. Notes Civ. Eng.* **2025**, *539*, 355–370.
3. Alizadehsalehi, S.; Hadavi, A.; Huang, J.C. From BIM to extended reality in AEC industry. *Autom. Constr.* **2020**, *116*, 103254. [CrossRef]
4. Okonta, D.E.; Vukovic, V. Smart cities software applications for sustainability and resilience. *Heliyon* **2024**, *10*, e32654. [CrossRef]
5. Chakrabarty, B.K. Integrated Computer-Aided Design by Optimization: An Overview. In *Integrated CAD by Optimization: Architecture, Engineering, Construction, Urban Development and Management*; Springer: Cham, Switzerland, 2022; pp. 1–49. [CrossRef]
6. Mba, E.J.; Okeke, F.O.; Igwe, A.E.; Ebohon, O.J.; Awe, F.C. Changing needs and demand of clients vs ability to pay in architectural industry. *J. Asian Archit. Build. Eng.* **2025**, *15*, 1–24. [CrossRef]
7. Okpalike, C.; Okeke, F.O.; Ezema, E.C.; Oforji, P.I.; Igwe, A.E. Effects of Renovation on Ventilation and Energy Saving in Residential Building. *Civ. Eng. J.* **2022**, *7*, 124–134. [CrossRef]
8. Holmström, J.; Carroll, N. How organizations can innovate with generative AI. *Bus. Horiz.* **2024**, *in press*. [CrossRef]
9. Inupakutika, D.; Nadim, M.; Gunnam, G.R.; Kaghyan, S.; Akopian, D.; Chalela, P.; Ramirez, A.G. Integration of NLP and speech-to-text applications with chatbots. *Electron. Imaging* **2021**, *33*, 1–6. [CrossRef]
10. Qiang, G.; Tang, S.; Hao, J.; Di Sarno, L.; Wu, G.; Ren, S. Building automation systems for energy and comfort management in green buildings: A critical review and future directions. *Renew. Sustain. Energy Rev.* **2023**, *179*, 113301. [CrossRef]
11. Sezgin, E.; Hussain, S.A.; Rust, S.; Huang, Y. Extracting medical information from free-text and unstructured patient-generated health data using natural language processing methods: Feasibility study with real-world data. *JMIR Form. Res.* **2023**, *7*, e43014. [CrossRef]
12. Dogan, S. Voice Control and Voice Searching. In *Future Tourism Trends Volume 2 (Building the Future of Tourism)*; Tanrisever, C., Pamukçu, H., Sharma, A., Eds.; Emerald Publishing Limited: Leeds, UK, 2024; pp. 159–172. [CrossRef]
13. Dweik, W.; Abdalla, M.; AlHroob, Y.; AlMajali, A.; Mustafa, S.A.; Abdel-Majeed, M. Skeleton of implementing voice control for building automation systems. *Sci. Program.* **2022**, *2022*, 6886086. [CrossRef]
14. Olawade, D.B.; Wada, O.Z.; David-Olawade, A.C.; Fapohunda, O.; Ige, A.O.; Ling, J. Artificial intelligence potential for net zero sustainability: Current evidence and prospects. *Next Sustain.* **2024**, *4*, 100041. [CrossRef]
15. Xu, H.; Omitaomu, F.; Sabri, S.; Zlatanova, S.; Li, X.; Song, Y. Leveraging generative AI for urban digital twins: A scoping review on the autonomous generation of urban data, scenarios, designs, and 3D city models for smart city advancement. *Urban Inform.* **2024**, *3*, 29. [CrossRef]
16. Jung, S.Y. Augmenting Deep Learning for Human-Centered Artificial Intelligence. Ph.D. Thesis, National Yang Ming Chiao Tung University, Hsinchu, Taiwan, 2022. Available online: https://www.proquest.com/openview/753dd7e6d315448d3f3bf1a8a2d3bc93/1?cbl=2026366&diss=y&pq-origsite=gscholar (accessed on 25 August 2024).
17. Farooqui, A.; Bengtsson, K.; Falkman, P.; Fabian, M. Towards data-driven approaches in manufacturing: An architecture to collect sequences of operations. *Int. J. Prod. Res.* **2020**, *58*, 4947–4963. [CrossRef]

18. Cerquitelli, T.; Pagliari, D.J.; Calimera, A.; Bottaccioli, L.; Patti, E.; Acquaviva, A.; Poncino, M. Manufacturing as a Data-Driven Practice: Methodologies, Technologies, and Tools. *Proc. IEEE* **2021**, *109*, 399–422. [CrossRef]
19. Czvetkó, T.; Kummer, A.; Ruppert, T.; Abonyi, J. Data-driven business process management-based development of Industry 4.0 solutions. *CIRP J. Manuf. Sci. Technol.* **2022**, *36*, 117–132. [CrossRef]
20. Olivetti, E.A.; Cole, J.M.; Kim, E.; Kononova, O.; Ceder, G.; Han, T.Y.-J.; Hiszpanski, A.M. Data-driven materials research enabled by natural language processing and information extraction. *Appl. Phys. Rev.* **2020**, *7*, 041317. [CrossRef]
21. Shah, V. Advancements in deep learning for natural language processing in software applications. *Int. J. Comput. Sci. Technol.* **2020**, *4*, 45–56.
22. Wu, C.; Li, X.; Guo, Y.; Wang, J.; Ren, Z.; Wang, M.; Yang, Z. Natural language processing for smart construction: Current status and future directions. *Autom. Constr.* **2022**, *134*, 104059. [CrossRef]
23. Di Giuda, G.M.; Locatelli, M.; Schievano, M.; Pellegrini, L.; Pattini, G.; Giana, P.E.; Seghezzi, E. Natural Language Processing for Information and Project Management. In *Digital Transformation of the Design, Construction and Management Processes of the Built Environment*; Springer: Cham, Switzerland, 2019; pp. 95–102. [CrossRef]
24. Nabavi, A.; Ramaji, I.J.; Sadeghi, N.; Anderson, A.J. Leveraging Natural Language Processing for Automated Information Inquiry from Building Information Models. *J. Inf. Technol. Constr.* **2023**, *28*, 266–285. [CrossRef]
25. Gupta, R.; Kondapally, R.; Guha, S. Large-Scale Information Extraction from Emails with Data Constraints. In *Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2019; pp. 124–139. [CrossRef]
26. Mahlawi, A.Q.; Sasi, S. Structured Data Extraction from Emails. In Proceedings of the 2017 International Conference on Networks & Advances in Computational Technologies (NetACT), Thiruvananthapuram, India, 20–22 June 2017; pp. 323–328. [CrossRef]
27. Schmitt, X.; Kubler, S.; Robert, J.; Papadakis, M.; Letraon, Y. A Replicable Comparison Study of NER Software: StanfordNLP, NLTK, Opennlp, Spacy, Gate. In Proceedings of the Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS), Granada, Spain, 22–25 October 2019; IEEE: Piscataway, NJ, USA, 2019. Available online: https://mpapad.github.io/publications/pdfs/SNAMS2019.pdf (accessed on 20 August 2024).
28. Rogers, A.; Gardner, M.; Augenstein, I. QA Dataset Explosion: A Taxonomy of NLP Resources for Question Answering and Reading Comprehension. *ACM Comput. Surv.* **2022**, *55*, 1–45. [CrossRef]
29. Zulkarnain; Putri, T.D. Intelligent transportation systems (ITS): A systematic review using a Natural Language Processing (NLP) approach. *Heliyon* **2021**, *7*, e08615. [CrossRef]
30. Huang, J.; Johanes, M.; Kim, F.C.; Doumpioti, C.; Holz, G.-C. On GANs, NLP and Architecture: Combining Human and Machine Intelligences for the Generation and Evaluation of Meaningful Designs. *Technol. Archit. Des.* **2021**, *5*, 207–224. [CrossRef]
31. Wang, N.; Issa, R.R.A.; Anumba, C.J. NLP-Based Query-Answering System for Information Extraction from Building Information Models. *J. Comput. Civ. Eng.* **2022**, *36*, 04022008. [CrossRef]
32. Zhou, Y.; She, J.; Huang, Y.; Li, L.; Zhang, L.; Zhang, J. A Design for Safety (DFS) Semantic Framework Development Based on Natural Language Processing (NLP) for Automated Compliance Checking Using BIM: The Case of China. *Buildings* **2022**, *12*, 780. [CrossRef]
33. Zhao, L.; Alhoshan, W.; Ferrari, A.; Letsholo, K.J.; Ajagbe, M.A.; Chioasca, E.-V.; Batista-Navarro, R.T. Natural Language Processing for Requirements Engineering. *ACM Comput. Surv.* **2021**, *54*, 1–41. [CrossRef]
34. Fischbach, J.; Frattini, J.; Vogelsang, A.; Mendez, D.; Unterkalmsteiner, M.; Wehrle, A.; Henao, P.R.; Yousefi, P.; Juricic, T.; Radduenz, J.; et al. Automatic creation of acceptance tests by extracting conditionals from requirements: NLP approach and case study. *J. Syst. Softw.* **2023**, *197*, 111549. [CrossRef]
35. Chowdhary, K.R. Natural Language Processing. In *Fundamentals of Artificial Intelligence*; Springer: New Delhi, India, 2020; pp. 603–649. [CrossRef]
36. Vajjala, S.; Majumder, B.; Gupta, A.; Surana, H. *Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2020. Available online: https://books.google.com.ng/books?hl=en&lr=&id=hvrrDwAAQBAJ&oi=fnd&pg=PP1&dq=Ensure+that+the+NLP+integration+performs+as+expected+across+various+scenarios+and+user+inputs&ots=gsew8n0_fe&sig=OAq16KP-Y-NhdboHZJzdVAVYsQE&redir_esc=y (accessed on 20 August 2024).
37. Cejas, O.A.; Azeem, M.I.; Abualhaija, S.; Briand, L.C. NLP-based automated compliance checking of data processing agreements against GDPR. *IEEE Trans. Softw. Eng.* **2023**, *49*, 4282–4303. [CrossRef]
38. Shaik, T.; Tao, X.; Li, Y.; Dann, C.; McDonald, J.; Redmond, P.; Galligan, L. A Review of the Trends and Challenges in Adopting Natural Language Processing Methods for Education Feedback Analysis. *IEEE Access* **2022**, *10*, 56720–56739. [CrossRef]
39. Mustansir, A.; Shahzad, K.; Malik, M.K. Towards automatic business process redesign: An NLP based approach to extract redesign suggestions. *Autom. Softw. Eng.* **2022**, *29*, 1–35. [CrossRef]

40. Raharjana, I.K.; Siahaan, D.; Fatichah, C. User stories and natural language processing: A systematic literature review. *IEEE Access* **2021**, *9*, 53811–53826. [CrossRef]
41. Pais, S.; Cordeiro, J.; Jamil, M.L. NLP-based platform as a service: A brief review. *J. Big Data* **2022**, *9*, 54. [CrossRef]
42. Locatelli, M.; Seghezzi, E.; Pellegrini, L.; Tagliabue, L.C.; Di Giuda, G.M. Exploring Natural Language Processing in Construction and Integration with Building Information Modeling: A Scientometric Analysis. *Buildings* **2021**, *11*, 583. [CrossRef]
43. Bennett, T. Direct Database Access vs. REST APIs: Compare Application Activity. DreamFactory Blog. Available online: https://blog.dreamfactory.com/direct-database-access-vs-rest-apis-pros-and-cons-for-application-connectivity/#:~:text=Direct%20database%20access%20and%20REST%20APIs%20differ (accessed on 18 August 2024).
44. Tuama, D.Ó. What Is a Plugin? Code Institute Global. Available online: https://codeinstitute.net/global/blog/what-is-a-plugin/ (accessed on 23 June 2024).
45. Denissov, N. Creating an Educational Plugin to Support Online Programming Learning A Case of IntelliJ IDEA Plugin for A+ Learning Management System. Master's Thesis, Aalto University, Espoo, Finland, 2021. Available online: https://aaltodoc.aalto.fi/items/7b9aa893-9614-4ac8-84a3-58f8a6c832eb (accessed on 22 June 2024).
46. Saad, A.; Ajayi, S.O.; Alaka, H.A. Trends in BIM-based plugins development for construction activities: A systematic review. *Int. J. Constr. Manag.* **2022**, *23*, 2756–2768. [CrossRef]
47. Oti, A.H.; Tizani, W.; Abanda, F.H.; Jaly-Zada, A.; Tah, J.H.M. Structural sustainability appraisal in BIM. *Autom. Constr.* **2016**, *69*, 44–58. [CrossRef]
48. Murali, A. What Is API Deprecation & Its Guidelines. Document360 Blog. Available online: https://document360.com/blog/api-deprecation/ (accessed on 1 May 2024).
49. Afsari, K.; Eastman, C.; Shelden, D. Building Information Modeling data interoperability for Cloud-based collaboration: Limitations and opportunities. *Int. J. Archit. Comput.* **2017**, *15*, 187–202. [CrossRef]
50. Moretti, N.; Xie, X.; Merino, J.; Brazauskas, J.; Parlikad, A.K. An openBIM Approach to IoT Integration with Incomplete As-Built Data. *Appl. Sci.* **2020**, *10*, 8287. [CrossRef]
51. Sobernig, S.; Gaubatz, P.; Strembeck, M.; Zdun, U. Comparing complexity of API designs. *ACM SIGPLAN Not.* **2012**, *47*, 157–166. [CrossRef]
52. Nolst Trenité, T. Simplifying the Complex: Strategies to Reduce Existing API Complexity Using CPS Techniques. Master's Thesis, Utrecht University, Utrecht, The Netherlands, 2023. Available online: https://studenttheses.uu.nl/handle/20.500.12932/44009 (accessed on 15 September 2024).
53. Du, J.; Zou, Z.; Shi, Y.; Zhao, D. Zero latency: Real-time synchronization of BIM data in virtual reality for collaborative decision-making. *Autom. Constr.* **2018**, *85*, 51–64. [CrossRef]
54. Antonopoulos, P.; Arasu, A.; Singh, K.D.; Eguro, K.; Gupta, N.; Jain, R.; Kaushik, R.; Kodavalla, H.; Kossmann, D.; Ogg, N.; et al. Azure SQL Database Always Encrypted. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020. [CrossRef]
55. Reimann, P.; Schwarz, H.; Mitschang, B. Design, implementation, and evaluation of a tight integration of database and workflow engines. *J. Inf. Data Manag.* **2011**, *2*, 353.
56. Nikolov, N.; Dessalk, Y.D.; Khan, A.Q.; Soylu, A.; Matskin, M.; Payberah, A.H.; Roman, D. Conceptualization and scalable execution of big data workflows using domain-specific languages and software containers. *Internet Things* **2021**, *16*, 100440. [CrossRef]
57. Wu, S.; Shen, Q.; Deng, Y.; Cheng, J. Natural-language-based intelligent retrieval engine for BIM object database. *Comput. Ind.* **2019**, *108*, 73–88. [CrossRef]
58. Ignatova, E.; Zotkin, S.; Zotkina, I. The extraction and processing of BIM data. *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *365*, 062033. [CrossRef]
59. Aydin, C. Best Revit Plugins: Enhancing Efficiency in BIM Workflows. Project Cubicle. Available online: https://www.projectcubicle.com/best-revit-plugins-enhancing-efficiency-in-bim-workflows/ (accessed on 18 August 2024).
60. Abbasnejad, B.; Moeinzadeh, S.; Ahankoob, A.; Wong, P.S.P. The Role of Collaboration in the Implementation of BIM-Enabled Projects. In *Handbook of Research on Driving Transformational Change in the Digital Built Environment*; IGI Global: Hershey, PA, USA, 2021; pp. 1–36. [CrossRef]
61. Atik, S.; Doménech, T.; Raslan, R. The Opportunities and Challenges of Using LCA-Based BIM Plugins in Early-Stage Building Design: An Industry Expert Perspective. In Proceedings of the APAC Conference on Sustainable Built Environment, Kandy, Sri Lanka, 16–18 December 2022; Springer: Singapore, 2022; pp. 401–408. [CrossRef]
62. Narkhede, S. What Is an SDK? (Definition, Benefits, vs. API, Uses). Builtin 2022. Available online: https://builtin.com/software-engineering-perspectives/sdk (accessed on 5 August 2024).

63. Kazado, D.; Kavgic, M.; Eskicioglu, R. Integrating Building Information Modeling (BIM) and sensor technology for Facility Management. *J. Inf. Technol. Constr.* **2019**, *24*, 440–458.
64. Song, J.; Kim, J.; Lee, J.K. NLP and deep learning-based analysis of building regulations to support automated rule checking system. In Proceedings of the 35th International Symposium on Automation and Robotics in Construction, Berlin, Germany, 20–25 July 2018; pp. 586–592. [CrossRef]