ChatGPT

AI-Augmented Architectural Knowledge Systems: Designing Institutional Memory and Retrieval Frameworks for Contemporary AEC Practice

Abstract

Architectural practice suffers from fragmented knowledge and fading institutional memory, as project-based work disrupts continuity and traditional Building Information Modeling (BIM) repositories often overlook contextual design wisdom. This paper proposes a conceptual framework for an **AI-augmented knowledge system** tailored to AEC firms: a graph-based "knowledge vault" with integrated semantic search and language-model retrieval. Inspired by recent practitioner concepts, our model uses a local markdown vault (e.g. Obsidian) for authoring, a private static-publishing layer for controlled dissemination, a Postgres/pgvector backend (via Supabase) for data and embeddings storage, and a serverless edge AI layer (Cloudflare Workers with vector search) to power natural-language queries. We ground this design in literature on knowledge management, BIM limitations, and information retrieval systems. Case scenarios illustrate retrieving past design decisions, standards excerpts, and project lessons to support architects' decisions. We discuss feasibility (scalability, BIM interoperability, latency, security), and identify research gaps at the intersection of PKM, BIM and AI retrieval. By theorising this architecture as an institutional memory infrastructure, we chart pathways toward AI-native, knowledge-driven design environments. This system conceptually bridges tacit and explicit knowledge, aiming to enable smarter design intelligence and sustain firm knowledge across generations.

Keywords: Knowledge management; architectural practice; institutional memory; knowledge graphs; BIM; AI retrieval; AEC; information systems.

Introduction

The Architecture, Engineering, and Construction (AEC) industry is undergoing digital transformation, yet it remains inherently fragmented and project-driven. Recent reviews note that while digitisation promises productivity and sustainability gains, the industry's business-to-business logic and dispersed structure impede seamless technology adoption [1]. Construction projects involve numerous firms in temporary consortia, so knowledge tends to remain siloed. As a result, **institutional memory** is hard to maintain: insights from past projects are often "re-learned" by each new team or lost altogether. This constant turnover wastes time and impedes innovation. For example, Vaz-Serra and Edwards report that AEC firms repeatedly re-discover solutions to similar problems because knowledge capture is weak [2]. In the absence of a "single source of truth," designers rely on personal memory or quick Internet searches, which can be error-prone and inefficient [2] [3].

Traditional knowledge repositories in AEC are largely built around Building Information Models. BIM excels at storing geometric and parametric data about building components, but it has well-known limitations as an institutional memory system [4]. State-of-art BIM tools typically exchange only geometry and property data, largely **excluding design intent and project context**. In other words,

BIM captures *what* was built but not *why* or *how* design decisions were made. This makes it difficult to retrieve general lessons learned or precedents when starting a new project. Furthermore, interoperability between discipline-specific BIM models remains challenging [4], so information is scattered across unlinked 3D models. In practice, firms often supplement BIM with document archives, emails, and personal notes – but these are fragmented and not easily searchable. As Deng *et al.* note, the AEC industry's long projects, dynamic teams, and abundance of unstructured data create a perfect storm that makes knowledge capture and sharing very difficult [5].

Meanwhile, advances in artificial intelligence and large language models (LLMs) are promising new ways to retrieve and apply knowledge. Retrieval-augmented generation (RAG) techniques combine contextual embeddings with generative AI to answer queries using curated data [6] [7]. Initial studies show that AI-powered assistants and conversational agents could benefit AEC professionals, though such applications remain underexplored [8] [9]. For example, a recent review finds Conversational AI offers "immense benefits" to AEC design and management, yet current adoption is low [8]. As powerful as LLMs are, they still need domain-specific data: systems like RAG use vector databases to ground AI replies in a firm's own knowledge base [6] [10].

Research motivation: We propose that combining AI with modern personal knowledge management (PKM) tools can address AEC's institutional memory gap. In particular, a graph-based knowledge vault (inspired by tools like Obsidian) can capture and link architectural knowledge, while AI-driven search and reasoning provides rapid access. This paper develops a conceptual architecture for an "AI-augmented Architectural Knowledge System" that draws on these technologies. We analyse design rationale, integration with BIM, and effects on architectural workflow. Our contributions are (1) a theoretical framework defining key concepts (AI-augmented systems, knowledge vaults, institutional memory infrastructure), (2) a layered system model combining Obsidian, Supabase, and Cloudflare components, (3) illustrative AEC use-cases, and (4) identification of research gaps. By treating the "Expert AEC Terminal" prototype as inspiration rather than advertisement, we position this work academically: as a thought-piece on what a knowledge-savvy AEC practice might look like.

Literature Review

Knowledge Management in AEC

The AEC sector has long been recognized as a **knowledge-intensive but fragmented industry** [3] [11]. Temporary project teams, diverse stakeholders, and distributed responsibilities mean that valuable experience is often isolated. Rezgui *et al.* (2010) observe that AEC organizations struggle to communicate effectively across partners, leading to information inconsistencies [3]. Vaz-Serra and Edwards (2021) describe AEC knowledge management as a "nightmare" because every project may recreate solutions and learning from scratch [2]. Deng *et al.* (2023) reinforce this view: they identify team instability, lengthy projects, complexity, and excessive unstructured data as core barriers to capturing and reusing knowledge [5]. In summary, AEC practitioners repeatedly face the problem that **knowledge from past work is not easily accessible for future tasks** [12] [2].

In response, AEC research has proposed various IT-based knowledge management (KM) approaches. Early efforts included constructing taxonomies, ontologies, or databases of lessons learned. Some studies emphasize the need for "organizational memory" systems – repositories where documents, design rationales, and project data are stored for reuse [13]. For instance, Ozorhon *et al.* (2013) describe a web-based KM tool that aggregates project and corporate data into an organizational memory, aiding future decision-making [13]. In Portugal, the ConstruKnowledge KMS was field-tested in construction firms: it aggregated guidelines and example solutions, and showed that using it raised managers'

confidence in decisions and helped retain corporate knowledge [14]. However, such systems often require significant manual input and maintenance.

Another line of AEC-KM work leverages Building Information Modeling (BIM) as a knowledge repository. Since BIM models encode rich data about project components, they naturally appeal as knowledge stores. Deng *et al.* (2023) note that **BIM can organize multi-modal knowledge** (text, geometry, attributes) in an integrated way [15]. A well-implemented BIM could serve as a "single source of truth" for project information, theoretically improving coordination. For example, BIM standards like IFC try to ensure interoperability. However, many authors point out that BIM excels at project-specific data but falls short for overarching firm knowledge. Wang and Sacks (2025) argue that BIM's discipline-specific silos and limited data structure hinder machine learning: BIM tools struggle to capture interdisciplinary design intent [4]. Similarly, Deng *et al.* find that although BIM supports multi-disciplinary data, other knowledge (e.g. reasoning, lessons) is still managed ad-hoc outside the model [15]. In practice, firms often keep design notes, tutorials, or internal wikis that are disconnected from BIM.

The limits of both generic KM tools and BIM have prompted exploration of **knowledge graphs and semantic models** in AEC. A knowledge graph (KG) explicitly links entities (design concepts, materials, processes) and is well-suited to capture relationships. For example, Yang *et al.* (2022) built a large "Built Environment Knowledge Graph" from 80,000 published abstracts, highlighting that consolidated project knowledge can assist risk assessment and decision-making [12]. By embedding such graphs, LLMs can be provided with structured context to improve domain answers. Similarly, Li *et al.* (2026) demonstrate an **LLM retrieval framework** for early design: they create an architectural knowledge graph for design logic and show that retrieval-augmented generation yields more accurate precedent recommendations [6]. These studies suggest that **semantic linking and KG-based retrieval** can greatly enrich design assistance. However, most KG work in AEC still focuses on published research or formal ontologies, not on a firm's own tacit knowledge.

Personal Knowledge Management (PKM) Systems

Outside of AEC, the PKM community has developed tools for individual knowledge curation. Systems like Evernote, OneNote, and more recently Obsidian and Roam, allow professionals to capture notes, link ideas, and tag content. Although formal AEC literature on PKM is sparse, practitioners often note that linkable note-taking (e.g. Zettelkasten method) mirrors how design thinking works. In essence, **an Architectural Knowledge Vault** can be seen as a firm-scale PKM: a space where architects create, connect, and store insights. Such vaults rely on human curation (authors link and tag documents), as opposed to purely AI-driven search. The value of PKM is that the knowledge base grows with user intent: architects embed metadata anticipating future queries. However, without AI augmentation, retrieval is still manual (keyword search or navigation). This duality—**human-curated vs AI-retrieved knowledge**—underscores our framework: we assume domain experts initially author the content (ensuring accuracy), while AI tools provide flexible, semantic retrieval and reasoning on top.

AI and LLM-Based Retrieval Systems

The rise of large language models has sparked interest in embedding AI into professional tools. "Retrieval-Augmented Generation" (RAG) combines vector similarity search over a knowledge base with LLM completion, which has been shown to improve answer correctness [6]. In AEC, some pilot applications are emerging: for instance, NVIDIA's developer blog outlines how RAG can power workplace assistants to find design documents or guidelines. More academically, Liang *et al.* (2025) introduce **AECBench**, a benchmark showing that generic LLMs still struggle with domain-specific tasks (e.g. looking up building code tables) [16]. This highlights both opportunity and challenge: AI retrieval

could significantly speed knowledge access, but the models must be carefully guided by vetted data to avoid hallucination and to ensure safety.

Institutional Memory and Organizational Learning

Organizational learning theory emphasizes that firms must convert individual and project knowledge into collective institutional memory. AEC firms have long experience to preserve – design heuristics, typical build strategies, regulatory interpretations. Studies note that such memory can improve future project outcomes [12]. For example, Yang *et al.* cite research showing that knowledge from past projects "can be helpful for managers in assessing project risks" [12]. However, few studies explicitly call for AI in this domain. Deng *et al.* (2023) argue future KM research should address cognitive and evaluation aspects [17]. In practice, E-learning or L&D teams and knowledge managers are beginning to pilot AI chatbots on internal data. Yet empirical evidence is scarce on how AI retrieval truly affects design cognition or firm strategy. This gap motivates our analysis of "AI-Augmented Knowledge Systems" as an infrastructure for continuous learning.

Conceptual Framework

AI-Augmented Knowledge Systems: We define these as integrated platforms where human-curated knowledge (documents, notes, design models) is augmented by AI-powered retrieval and reasoning. Such systems go beyond static databases: they allow a user to query the firm's collective knowledge in natural language, receive synthesized answers, and discover relevant resources. In this context, *AI-augmented* implies that machine learning (e.g. language models, embeddings) assists in filtering, ranking, and generating responses, but the underlying knowledge is still authored or vetted by people. This approach leverages the strengths of both: humans ensure quality and context, AI accelerates search and insight.

Architectural Knowledge Vault: We borrow the term "vault" to convey a secure, long-term repository of architectural knowledge. Analogous to "second brain" PKM systems, our vault is based on a **graph database of notes**. Here, "graph" refers to interconnected notes (markdown files) that link by topic, project, or spatial element. Each note may represent a concept (e.g. "curtain wall detail", "Passive House principle", or a meeting summary). Links and tags form a graph structure that embodies semantic relationships. The vault thus acts as a machine-readable knowledge store of firm intellectual assets. In practice, we envision using a tool like Obsidian: it stores markdown files locally (ensuring durability and control) and automatically builds a bidirectional link graph from references. This infrastructure supports both **explicit knowledge** (documented solutions, checklists, code citations) and more tacit insights (through manual linking).

Institutional Memory Infrastructure: At the organizational level, we conceive an infrastructure that continually accrues knowledge across projects. Institutional memory here includes design rationales, lessons-learned logs, meeting minutes, precedent libraries, and even informal notes from senior staff. The infrastructure must enable both *capture* (putting new knowledge in) and *retrieval* (getting relevant knowledge out). In our model, the Obsidian vault plus publishing layer serves as the capture/editing interface – architects add or update notes as projects proceed. The AI retrieval layer then serves as the querying interface – any team member can query the vault's contents. This infrastructure differs from point solutions (like a one-off code checker) because it supports a continuous, firm-wide knowledge cycle.

Human-Curated vs AI-Retrieved Knowledge: It is important to distinguish between knowledge **creation** and knowledge **access**. In our framework, the actual writing of notes and linking them is done

by people. These are the curators: they decide what belongs in the vault, how to structure it, and what connections matter. AI steps in on the retrieval side: when a user poses a question, the system uses embeddings to find relevant notes, then may use an LLM to generate a summary or answer. The AI is thus a service to sift and synthesize the curated content. This mirrors the duality noted in knowledge literature: explicit knowledge (documents, rules) vs tacit knowledge (insights, intuitions). Our system treats notes as the explicit artifacts, while the AI effectively makes tacit links explicit on-the-fly.

By framing these concepts, we ground our proposal in theory: KM literature identifies the challenges (fragmentation, storage, retrieval) [2] [5], and AI research offers tools (KGs, RAG) to help. Our contribution is to bring these threads together into a cohesive architecture for AEC practice.

System Architecture & Design Model

The proposed framework is a **multi-layer architecture** (Figure 1) that separates knowledge authoring, publication, storage, AI retrieval, and user interface. This modular design allows each component to be implemented with best-of-breed tools while maintaining clear data flows. The layers are:

- **Knowledge Authoring (Obsidian vault):** At the base is a folder of markdown notes stored locally (the "vault"). Each note contains text (plain markdown) describing a concept, decision, technical detail, or lesson-learned. Authors (architects, engineers) create and edit notes using a dedicated editor (Obsidian). Crucially, notes use **bidirectional links** (e.g. `[[thermal mass]]`) and **tags** (e.g. `#code`, `#lighting`) to form semantic connections. Templates and frontmatter metadata can enforce consistency (e.g. project name, date, discipline). Because Obsidian is *local-first*, it ensures data ownership and offline access. The result is a growing **personal knowledge graph** where nodes are notes and edges are links or shared tags.
- **Publishing & Access Control (Obsidian Publish / static site):** To share knowledge across the firm, the vault is published as a controlled-access website. This can be done via Obsidian Publish or other static-site generators. Published content remains markdown-rendered HTML, preserving links. Access control (e.g. behind VPN or password) ensures only authorised staff see the archive. Importantly, publishing is one-way: it exposes content but users interact with it only through the AI interface (below). The published site can serve as a read-only institutional wiki.
- **Data Persistence & Indexing (Supabase):** The vault's contents are ingested into a database for backend use. We propose using Supabase (an open-source Postgres backend with pgvector). Each note is stored as a record: fields include the note's text, title, tags, creation date, etc. We also compute an embedding vector for each note (e.g. via OpenAI or HuggingFace models) and store it in a pgvector column [10]. This enables **semantic search**: a query vector can be compared against all note embeddings to find relevant notes efficiently. Supabase provides APIs to query these vectors, as well as indexing on keywords or tags. The database also holds secondary metadata (author, project, attachments). By centralizing in Postgres, we ensure durability (with backups) and flexibility for additional indexing. The use of an existing toolkit ("the vector DB is the one you already have" concept [10]) simplifies engineering.
- **AI Retrieval & Reasoning (Cloudflare Workers + Models):** On top of the index is a serverless AI engine. We envision Cloudflare Workers (with GPU-backed Workers AI) running a small application that handles user queries. When a user asks a question (free text), the system first sends it to a language model to parse intent or extract keywords. It then creates an embedding of the query and performs a semantic search on Supabase to retrieve, say, the top N relevant notes. These notes' text are then provided as context to an LLM (via OpenAI or Cloudflare's own models). The LLM generates an answer or summary grounded in the retrieved content. This pipeline – query→vector search→context→answer – is inspired by retrieval-augmented architectures [6]. Cloudflare Workers' global edge deployment means inference and DB calls

happen close to the user, improving speed [18]. Workers AI supports the LLM hosting and vector operations, and their pay-as-you-use model keeps costs aligned with usage [19]. Optionally, Cloudflare's "Vectorize" feature or Redis could cache popular retrievals. Crucially, this layer isolates AI tasks: notes remain in Supabase or the vault, and only embeddings/IDs flow to the LLM.

- **Frontend Interaction (Cloudflare Pages / Web UI):** The user interacts through a web interface served via Cloudflare Pages (a global static hosting) [20]. The UI can be a simple search bar or chat widget. Users enter natural-language queries (e.g. "How did we solve HVAC zoning in Project X?") and receive a summary answer plus links to relevant notes. The interface can highlight source snippets and allow users to click into the published vault if needed. Because the UI is static and lightweight, it enjoys low latency and high availability. Pages ensures the interface is globally distributed. This client layer also handles user authentication (e.g. single-sign-on) so that only authorized staff access the system.

Design Logic Highlights: The separation of layers provides flexibility. Architects need not change their note-taking habits: they continue using Obsidian for authoring. All metadata flows into Supabase behind the scenes (via an Obsidian plugin or webhook). The core design choice is to treat all project knowledge as text: even CAD/BIM outputs can be summarized or linked from notes, although full BIM integration is future work. By indexing on vectors rather than strict keyword search, the system can find conceptually related notes (e.g. synonyms, paraphrases) even if keywords differ. We also plan for hybrid search: a user could choose a tag-based filter or boolean search if desired, but the default is semantic. Security is enforced at multiple levels: Obsidian vault files can be encrypted locally, the Supabase DB can use row-level security, and Cloudflare Workers supports cryptographic secrets. We avoid storing any private data in the LLM or third-party servers beyond embeddings.

In sum, the architecture draws inspiration from contemporary "headless" knowledge systems: editing in markdown, static site publishing, and serverless function calling. It mimics many personal knowledge management practices, scaled to an organization with AI acceleration. (Figure 1 illustrates the layer interactions, with notes ingested into the DB, embeddings driving search, and responses delivered via the web UI.)

Figure 1 (textual): The system has five layers. 1) **Vault:** Users author notes in Obsidian (markdown files). 2) **Publish:** Notes are published as a private website. 3) **Database:** Supabase stores note text, metadata, and vector embeddings. 4) **AI Engine:** Cloudflare Workers handles queries, performing vector search and invoking LLMs. 5) **Frontend:** A web interface (Cloudflare Pages) allows user queries and displays answers.

Use Cases in AEC Practice

Design Decision Retrieval: Architects often need to recall how a past problem was solved. For instance, imagine an architect wonders about a structural detail from a similar project five years ago. Instead of digging through folders or emails, they could ask the system a question like *"What insulation strategy did we use at Project Acero for the thermal wall?"*. The AI layer would semantically match this query to relevant notes (perhaps a memo or a design meeting note describing "Project Acero wall insulation"), retrieve that content, and the LLM would summarize: *"We specified a triple-glazed envelope with 150mm mineral wool behind the cladding, chosen for cost-efficiency. Refer to notes [link] for calculations."* Here the system surfaces the exact institutional knowledge needed, linking design rationale with code references. This leverages precedent-based design support: studies show that retrieving precedent knowledge via RAG improves recommendations [6]. It also eliminates "re-learning loops" where each team re-solves the same issue.

Code and Standards Referencing: Another common task is looking up building codes or standards. Instead of opening a 500-page PDF, a user can query *"According to our standards vault, what is the latest wind load code citation for Gujarat region?"* The system could search notes tagged #code or containing "wind loading" and retrieve a standardized summary: for example, quoting from IS1893 and any firm-specific commentary. Because LLMs struggle with tabular code data by themselves [16], our system uses the firm's curated excerpts. When Liang *et al.* tested LLMs on building code tasks, the models performed well on recall-based queries but poorly on detailed table lookups [16]. Our approach remedies this by ensuring the exact code text (or expert summary) is in the vault, so the AI need only regurgitate it.

Lessons-Learned Archives: Project post-mortems often reveal costly mistakes or ingenious solutions. We envision a "Lessons Learned" section of the vault where such insights are logged as notes. For example, a structural engineer might note "In Project Terra, brickwork cracking occurred due to insufficient reinforcement clearance; ensure 30mm cover next time." Later, someone can ask *"Lessons from brick veneer failures?"* and retrieve that entry. By preserving anecdotal wisdom, the system functions like a collective memory. Prior research confirms that project knowledge is **reusable** across projects [12], and capturing it formally can improve risk management.

Firm Knowledge Continuity: When senior team members retire or move on, their tacit knowledge leaves too. With this platform, crucial know-how is not locked in people's heads. New hires can query the vault to learn "How do we detail [XYZ]?" and get firm-specific answers. Over time, the vault accumulates an organizational intelligence that remains even as staff change. This continuity contrasts with conventional archiving (where old files sit unread) by actively surfacing relevant insights on demand. Vaz-Serra & Edwards found that a firm KMS "serves to add value to the company" by retaining knowledge [21]. In our case-study scenario, a newly graduated architect using the system would effectively tap into decades of accumulated design intelligence with natural language.

Impact on Workflow and Cognition: Integrating this system into daily workflows could significantly alter AEC practice. Routine queries (material properties, past decisions, code checks) become conversational and immediate, reducing time spent rummaging through disparate sources. Cognitively, designers can offload recall to the system: instead of memorizing code numbers or precedent names, they focus on asking the right question. The iterative loop of writing and retrieval reinforces knowledge: as users link notes anticipating questions, the vault self-organizes around demand (as observed in PKM studies). Over time, design reasoning may shift from "Let me remember if we did this before" to "Let's ask our knowledge system." This could democratize access to expertise within a firm, flattening the learning curve for juniors. Of course, human oversight remains crucial; AI outputs must be checked. But in all cases, the aim is to make hidden knowledge visible at critical decision points, fundamentally augmenting designers' intelligence with institutional memory.

Feasibility & Implementation Considerations

Scalability: Serverless technologies underpin scalability. Cloudflare Workers runs on demand and scales to hundreds of concurrent queries without manual provisioning. Supabase (Postgres) can handle large document stores and vector indexes as data grows; automatic indexing and pagination keep search efficient. In practice, performance will depend on vault size and query volume. Supabase recommends embedding sizes and indexing strategies for scale [10]. Precomputing embeddings and using approximate nearest-neighbor search can ensure sub-second responses even for thousands of notes. Cloudflare's global distribution also cuts round-trip latency. Cost scales with usage: Cloudflare's Workers AI uses a pay-per-token pricing [19], and Supabase may incur minimal fees at scale. These are trade-offs; heavy generative queries (like long answer summaries) cost more tokens. Nevertheless, for

moderate query traffic, this architecture avoids the fixed costs of dedicated servers and GPUs, making proof-of-concept trials affordable.

Data Structuring Challenges: Architecting the note schema is non-trivial. Markdown is unstructured text by default, so we must standardize how information is captured. The team should adopt consistent templates (e.g. “Decision Log”, “Technical FAQ”, “Standard Excerpt”). Metadata (project name, discipline, date) should use frontmatter fields. Inconsistent entries could reduce retrieval accuracy. Ensuring notes link to actual BIM elements is another challenge: while the vault exists outside the BIM model, we may want interoperability. For example, a note about a wall type might link to the BIM object ID. This requires manual cross-referencing or use of unique IDs. Future work could involve automated extraction: e.g. a script that parses IFC or CAD for names and populates related notes. However, our current model treats BIM as a separate source: the vault may contain high-level references (e.g. “See wall type W12 in model”), but deep integration is left to future research.

Interoperability with BIM/CAD: Interfacing this knowledge system with BIM tools raises both opportunities and hurdles. On one hand, federated models via IFC could expose geometry and properties to the knowledge vault: e.g. embedding architectural element attributes into notes. On the other hand, the semantic gap is large. BIM standards like IFC are not designed to contain design reasoning or case studies. Translating a BIM object into a meaningful note (or vice versa) may require custom mappings. For instance, linking a note titled “Rain screen assembly” to a BIM facade type might use a shared identifier or naming convention. Standards bodies (ISO 19650, buildingSMART) focus on data exchange and do not currently specify knowledge-linkage protocols. Thus, initial deployment of our system would likely be *BIM-agnostic*, focusing on textual knowledge. In practice, firms could maintain pointer links (URLs or GUIDs) between BIM elements and vault notes. Long-term, we anticipate extensions where ontologies connect BIM classes with vault content (akin to the BIGs concept), but that remains experimental.

Latency & Performance: The response time depends on several factors. Query embedding generation is fast (milliseconds) given modern GPUs, and vector search in a well-indexed Postgres can also be near-instant. The main delay is the LLM completion (which might take 1–2 seconds for a moderate answer). Cloudflare Workers are located at edge nodes, so network latency is minimal if both user and data center are regionally close. To optimize performance, the system can use smaller language models for quick answers (deployed on Cloudflare Workers) and only call larger models for in-depth queries. Caching is possible: frequent queries (e.g. “Show all standards for steel design”) could cache answers. Overall, a well-tuned deployment should keep end-user wait times under a few seconds.

Cost Implications: As noted, serverless and managed services minimize upfront capital. Supabase is open-source and inexpensive to scale. Cloudflare Workers has a free tier and then metered pricing. The largest cost driver will be the LLM API or model hosting. If using a provider like OpenAI, token fees per query must be budgeted. For example, answering a detailed question might consume a few cents worth of tokens, so a busy office could run $1000s per year easily. To mitigate this, organizations could host open-source LLMs on Cloudflare Workers AI (limit 14GB models currently) to avoid per-token fees, trading off some accuracy. Another cost factor is embedding generation: if using an external API, the cost of generating vectors for each note must be considered (though this is one-time per note unless re-indexed). Additionally, premium features like Cloudflare AI Gateway (for access control, retries) might incur fees. Firms will need to weigh these against productivity gains; however, even partial automation of retrieval could yield substantial time savings (office surveys often cite knowledge search as a major time sink).

Security & Access Control: Protecting proprietary information is critical. Obsidian vault files can be encrypted at rest on local machines. The published site must enforce authentication (e.g. via company

SSO). Supabase offers Role-Based Access Control and Row-Level Security; only the AI layer (a secured service account) should have full DB access. User credentials for the AI system should not expose internal data – for example, each query could embed a user token so results can be logged by user. Cloudflare Workers allow JWT verification and rate limiting (through AI Gateway) to ensure only valid users query the model. Moreover, the system can strip PII or sensitive project data from notes if needed; any content transformation should occur before indexing. The LLM itself must be used carefully: queries and retrieved data should not be sent to a public model if it poses a security risk. One solution is to run models entirely on private instances (e.g. Workers AI's private functions) or to ensure via the API that "do not train on this" flags are used. Finally, given LLMs' tendency to "hallucinate," answers must be traceable. The system's UI should always display the source notes or the exact passages it used to generate an answer, so that users can verify. This transparency is crucial for trust in an enterprise setting.

Risks and Mitigations:
- *Hallucination/Bias:* If an LLM hallucinates an answer not supported by the vault, it can mislead designers. We mitigate this by (a) using strict RAG pipelines that require context snippets, and (b) clearly marking answer confidence or source quotes. For high-stakes queries (e.g. code compliance), the system might present retrieved text *only* and eschew generative summarization.
- *Incomplete Archives:* The system is only as good as its content. Important knowledge not in the vault simply won't be found. This risk means initial results may have gaps. To address it, firms should encourage regular knowledge capture (perhaps via mandatory note-taking templates) and consider integrating external resources (e.g. buildingSMART datasets or standards libraries) where possible. The system can also fall back on the web if needed, but with caution.
- *Over-reliance:* Users might over-trust the AI and skip critical thinking. Training and clear UI design can encourage users to double-check. For instance, the system could ask clarifying questions or provide multiple sources when uncertainty is detected.
- *Data Quality:* The performance depends on note quality. Redundant or contradictory entries could confuse retrieval. Periodic housekeeping (review and prune notes) is advisable.
- *Scale Limitations:* A very large firm with millions of notes would challenge the vector DB. In that scenario, sharding or knowledge graph indexing might be needed. Early trials should monitor scale.

Overall, while there are non-trivial considerations, none are insurmountable. The core technologies (markdown storage, Postgres, Workers AI) are robust and industry-proven. The architecture relies on open standards and known components, making it technically feasible today.

Research Gap Identification

Despite these emerging ideas, the literature still has critical gaps at the intersection of AI, PKM, and AEC. First, **integrating personal knowledge management with enterprise AEC data** remains under-explored. Much AEC KM research focuses on BIM or project databases, not on leveraging individual note-taking tools in an organizational context. We found no studies on how tools like Obsidian could be scaled from personal to firm-wide use. Second, research on **non-BIM knowledge infrastructures** is sparse. BuildingSMART and IFC research addresses model data, but not the broad range of engineering documents, calculations, and qualitative insights that this paper considers. Our framework invites investigation into standards or schemas that could link free-form notes with formal building data.

Third, the idea of **serverless AI systems for AEC** is virtually absent in scholarly work. Cloud computing literature covers cloud-based BIM or IoT, but not on-demand inference networks supporting designers. This architecture uses cloud-native paradigms (FaaS, vector DB) that few AEC researchers have studied. Fourth, **cognitive implications of AI retrieval** need study: how does immediate access to institutional

knowledge change design thinking? Deng *et al.* highlight "cognitive barriers" in AEC KM [17], suggesting future work should examine how tools alter human workflows and mental models. Finally, there is a need for **empirical validation**: while the benefits seem obvious anecdotally, systematic trials (usability studies, productivity metrics, error rates) of AI-augmented systems in real firms have not been published. Our conceptual model reveals where these investigations could be focused, from query intent studies to ROI analysis.

In summary, key missing areas include (a) case studies of AI-PKM in firms, (b) frameworks for knowledge beyond BIM, (c) performance evaluations of edge-AI retrieval systems in architecture, and (d) sociotechnical research on how such systems fit into office culture.

Discussion

The implications of AI-augmented knowledge systems for AEC practice are broad. On the one hand, such systems promise to **democratize expertise** within firms. Junior designers could ask the knowledge system questions that previously only senior staff could answer, potentially flattening experience curves. This shift could accelerate decision-making and reduce design errors: by making collective wisdom queryable, the system functions like an "organizational memory" agent embedded in daily work. Over time, firms might evolve new roles: for example, Knowledge Curators who ensure the vault's quality, or Data Stewards who tune the AI.

For **design intelligence**, the move is from isolated intelligence (individual or team knowledge) to networked intelligence (firm-wide memory plus AI). Wang and Sacks argue that knowledge graphs can underpin "intelligent and generative design tools" [7]. Our framework realizes a slice of that vision: imagine future CAD or BIM plugins that query the vault mid-design (e.g. "suggest cladding assemblies used on similar buildings"), blending parametric modeling with AI-sourced precedents. By bridging explicit knowledge (documents, data) with generative AI, architects gain a hybrid cognition loop. Knowledge becomes an explicit design material; decisions can be annotated, indexed, and revisited, making even the creative process more data-driven.

At the **ecosystem level**, this approach hints at a new class of digital infrastructure for the built environment. Rather than monolithic, project-siloed data stores, we see a federated, graph-based knowledge mesh: personal vaults connected to corporate archives, all queryable via AI. This ecosystem could interoperate with other digital twins (e.g. linking sensor data or operation manuals into the knowledge graph). As "digital twins" of buildings become commonplace, so too might "digital twins of knowledge" – living models of a firm's expertise.

Looking ahead to **AI-native design environments**, one can envision voice-activated design assistants that have instant access to company knowledge. An architect might sketch a form and say, "Have we ever used this structural system before?" The system could respond with relevant case studies and performance data from past projects. In this scenario, knowledge is no longer passively stored but actively woven into the creative workflow. Of course, this raises important ethical and practical questions about accountability (who is responsible if the AI suggests a flawed past solution?) and human agency. These warrant further study.

In conclusion, AI-augmented knowledge systems represent a potential paradigm shift for AEC practice. They marry the **distributive cognitive approach** – using tools to extend human memory – with cutting-edge AI. By conceptualizing knowledge as an institutional asset to be actively managed and queried, we empower designers with collective intelligence. As Wang and Sacks note, moving beyond BIM's limits to

graph-based systems is a “paradigm shift” [7]. Our analysis suggests that incorporating AI into that paradigm shift could lead to more resilient, knowledgeable, and agile architecture firms.

Conclusion

This paper has presented a comprehensive vision for an AI-augmented architectural knowledge system that addresses critical challenges in contemporary AEC practice. By synthesizing literature on AEC knowledge management and leveraging modern AI/PKM tools, we developed a layered system architecture that unites human-curated content with machine-assisted retrieval. Key insights include: the persistent fragmentation of AEC knowledge despite BIM’s presence [3] [4], the potential of knowledge graphs and vector search to unlock design precedents [6] [12], and the need to treat institutional memory as strategic infrastructure. The proposed framework is not a commercial product pitch, but a conceptual model: it illustrates how Obsidian (for vault), Supabase (for indexing), and Cloudflare (for serverless AI) can interoperate in an enterprise setting. We showed use-case scenarios – from retrieving design decisions to referencing codes – that concretely demonstrate how such a system could save time and enrich decision-making. We also critically examined implementation issues (scalability, interoperability, security) and identified theoretical and empirical gaps (e.g. PKM integration, cognitive effects of AI retrieval) that warrant future research.

In sum, this work argues that AEC firms should think beyond static BIM and invest in **living knowledge ecosystems**. By treating knowledge as an evolving asset and equipping it with AI-enabled access, we can build smarter practices that learn and improve over time. The conceptual contributions of this paper provide a foundation for future prototyping and evaluation in real offices. Ultimately, the integration of AI with architectural knowledge promises to make practice not only more efficient but also more innovative and resilient.

Future Research Directions

Building on this foundation, several research avenues emerge. First, **prototyping and user studies** are needed: deploying a minimal version of this system in a small firm to observe impacts on workflow, accuracy, and satisfaction. Quantitative metrics (time saved, errors avoided) and qualitative feedback would guide refinement. Second, **AI model development** for AEC: fine-tuning LLMs on firm-specific knowledge or developing domain-tuned embeddings could boost performance. Third, **interoperability with BIM**: exploring methods to automatically link model elements with knowledge nodes (e.g. via common ontologies or tagging). Fourth, **knowledge engineering**: designing taxonomies or semantic models for architecture content could enhance retrieval precision (for example, mapping note tags to buildingSMART concepts). Fifth, **cognitive and organizational studies**: investigating how AI retrieval affects design cognition, teamwork, and learning culture; for example, does it encourage knowledge sharing or foster dependency on the system? Sixth, **scalability studies**: evaluating system performance on large datasets and optimizing vector search, possibly via specialized graph databases or distributed indices. Finally, **policy and ethics**: as firms grant AI access to internal knowledge, questions of data governance, privacy, and professional responsibility arise, requiring guidelines.

These directions will require interdisciplinary collaboration between architects, computer scientists, and organizational experts. By pursuing them, we can move from conceptual models to evidence-based tools that reshape the future of AEC knowledge work.

References

Cloudflare. (2023). *Cloudflare Workers AI Overview*. Cloudflare Developer Documentation. Retrieved from https://developers.cloudflare.com/workers-ai/

Deng, H., Xu, Y., Deng, Y., & Lin, J. (2023). Transforming knowledge management in the construction industry through ICT: A 15-year review. *Automation in Construction*, 144, 104530.

Li, D., Shi, Y., Schwartz, M., & Kapadia, M. (2026). Early-stage architecture design assistance by LLMs and knowledge graphs. *Automation in Construction*, 182, 106756.

Liang, C., Huang, Z., Wang, H., Chai, F., Zhao, X., Li, Y., *et al.* (2025). AECBench: A hierarchical benchmark for knowledge evaluation of large language models in the AEC field. *arXiv preprint* arXiv:2509.18776.

Rezgui, Y., Hopfe, C., & Vorakulpipat, C. (2010). Generations of knowledge management in the AEC industry: An evolutionary perspective. *Advanced Engineering Informatics*, 24(2), 219–228.

Samuelson, O., & Stehn, L. (2023). Digital transformation in construction – a review. *Journal of Information Technology in Construction (ITcon)*, 28, 385–404.

Saka, A. B., Oyedele, L. O., Akanbi, L. A., Ganiyu, S. A., Chan, D. W. M., & Bello, S. A. (2023). Conversational artificial intelligence in the AEC industry: A review of present status, challenges and opportunities. *Advanced Engineering Informatics*, 55, 101869.

Supabase. (2023). *AI & Vectors – Supabase Documentation*. Retrieved from https://supabase.com/docs/guides/ai

Vaz-Serra, P., & Edwards, P. (2021). Addressing the knowledge management “nightmare” for construction companies. *Construction Innovation*, 21(2), 300–320.

Wang, Z., & Sacks, R. (2025). Building Information Graphs (BIGs): Remodeling building information for learning and applications. *Data-Centric Engineering*, 6, e44.

Yang, X., Yang, Z., Zheng, L., & Li, Y. (2022). BEKG: Building Environment Knowledge Graph – A large-scale knowledge graph for the built environment. *arXiv preprint* arXiv:2211.02864.

---

1 itcon.org
https://www.itcon.org/papers/2023-20-ITcon-Samuelson.pdf

2 11 14 21 Addressing the knowledge management “nightmare” for construction companies | Emerald Insight
https://www.emerald.com/insight/content/doi/10.1108/CI-02-2019-0013/full/html

3 Generations of knowledge management in the architecture, engineering and construction industry: An evolutionary perspective - ScienceDirect
https://www.sciencedirect.com/science/article/abs/pii/S1474034609000810

4 7 Building Information Graphs (BIGs): remodeling building information for learning and applications | Data-Centric Engineering | Cambridge Core
https://www.cambridge.org/core/journals/data-centric-engineering/article/building-information-graphs-bigs-remodeling-building-information-for-learning-and-applications/ED0C0B75DA5D10EA87B206FCAF5FAA0C

5 15 17 Transforming knowledge management in the construction industry through information and communications technology: A 15-year review - ScienceDirect
https://www.sciencedirect.com/science/article/abs/pii/S0926580522004022

6 Early-stage architecture design assistance by LLMs and knowledge graphs - ScienceDirect
https://www.sciencedirect.com/science/article/pii/S0926580525007964

8 Conversational artificial intelligence in the AEC industry: A review of present status, challenges and opportunities - ScienceDirect
https://www.sciencedirect.com/science/article/pii/S1474034622003275

9 16 AECBench: A Hierarchical Benchmark for Knowledge Evaluation of Large Language Models in the AEC Field
https://arxiv.org/html/2509.18776v3

10 AI & Vectors | Supabase Docs
https://supabase.com/docs/guides/ai

12 13 arxiv.org
https://arxiv.org/pdf/2211.02864

18 19 20 Overview · Cloudflare Workers AI docs
https://developers.cloudflare.com/workers-ai/