FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Development of an Intelligent Knowledge Mangement System for Engineering Education

Sofia Merino Ferreira Marimba da Costa

Mestrado em Engenharia Informática e Computação

Supervisor: Prof. Rui Pinto

June 30, 2025

Development of an Intelligent Knowledge Mangement System for Engineering Education

Sofia Merino Ferreira Marimba da Costa

Mestrado em Engenharia Informática e Computação

June 30, 2025

Resumo

A crescente complexidade do ensino de engenharia e a rápida expansão dos recursos digitais criaram novos desafios na gestão, recuperação e personalização do acesso ao conhecimento. Esta dissertação apresenta o desenho, implementação e avaliação de um sistema inteligente de gestão do conhecimento desenvolvido para responder às exigências específicas do ensino de engenharia. O sistema proposto integra pesquisa semântica, recomendações baseadas em grafos e percursos de aprendizagem modulares, de forma a apoiar experiências de aprendizagem estruturadas e centradas no utilizador. A arquitetura da plataforma combina uma base de dados PostgreSQL, Elasticsearch para recuperação semântica e Neo4j para recomendações personalizadas com base nas interações e nos relacionamentos entre conteúdos. Foi realizada uma avaliação com métodos mistos para analisar o desempenho técnico, a usabilidade e a relevância educacional da plataforma. Os resultados indicaram que a funcionalidade de pesquisa semântica permitiu uma recuperação de informação mais contextualizada do que a simples correspondência de palavras-chave (MAP = 0.627, Recall@5 = 0.714). O sistema de recomendações apresentou sugestões relevantes em vários contextos da plataforma, com uma precisão elevada na correspondência entre os conteúdos recomendados e os interesses e objetivos de aprendizagem dos utilizadores. O feedback qualitativo destacou também a usabilidade, a clareza da interface e o valor educativo alcançado. No entanto, a avaliação apresentou algumas limitações, nomeadamente a escassez e reduzida diversidade dos recursos e conteúdos de aprendizagem disponíveis no sistema, bem como a impossibilidade de realizar uma avaliação de longo prazo que permitisse aferir os impactos reais nos resultados de aprendizagem. Apesar destas restrições, os resultados obtidos demonstram o potencial das tecnologias inteligentes para apoiar ambientes educativos inclusivos, adaptativos e centrados nos estudantes, em conformidade com os princípios da Educação 5.0.

Palavras-chave: Gestão de Conhecimento, Sistemas de Gestão de Conhecimento, Educação em Engenharia, Pesquisa Semântica, Sistema de Recomendação Baseado em Grafos, Percursos de Aprendizagem Modulares, Educação 5.0

Abstract

The growing complexity of engineering education and the rapid expansion of digital resources have created new challenges in managing, retrieving, and personalizing access to knowledge. This thesis presents the design, implementation, and evaluation of an intelligent knowledge management system developed to address the specific needs of engineering education. The proposed system integrates semantic search, graph-based recommendations, and modular learning paths to support structured, user-centered learning experiences. The platform architecture combines a PostgreSQL database, Elasticsearch for semantic retrieval, and Neo4j for personalized recommendations based on user interactions and content relationships. A mixed-methods evaluation was conducted to assess technical performance, usability, and educational relevance. Results indicated that the semantic search feature enabled more context-aware information retrieval beyond basic keyword matching (MAP = 0.627, Recall@5 = 0.714). The recommendation system provided relevant suggestions across multiple contexts within the platform, with consistently high precision in aligning recommendations to user interests and learning goals. Qualitative feedback also highlighted the usability of the platform, the clarity of the interface and the perceived educational value. However, the evaluation was subject to certain limitations, including the scarcity and limited diversity of available learning materials within the system, as well as the lack of a long-term assessment to measure learning outcomes over time. Despite these constraints, the findings illustrate the potential of intelligent technologies to support inclusive, adaptive and learner-centered educational environments aligned with the principles of Education 5.0.

Keywords: Knowledge Management, Knowledge Management Systems, Engineering Education, Semantic Search, Graph-Based Recommendation System, Modular Learning Paths, Education 5.0

Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs) provide a global framework to achieve a better and more sustainable future for all. It includes 17 goals to address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, peace, and justice.

The specific Sustainable Development Goals mentioned have the following names:

SDG 4 Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all

SDG 9 Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation

SDG	Target	Contribution	Performance Indicators and Metrics
4	4.3	Facilitates access to affordable and flexible higher education resources through open learning paths and intelligent content discovery.	Number of learning paths completed; User engagement with recommended resources
	4.4	Promotes development of technical and digital skills in engineering students by improving access to structured, personalized content.	Increase in interactions per session; Qualitative feedback on learning support
	4.7	Encourages learner autonomy and personalized pathways, fostering competencies aligned with sustainable development, digital literacy, and global citizenship.	User satisfaction with learning experience; Number of personalized paths completed
9	9.5	Supports research and innovation by proposing a replicable architecture using modern technologies (Node.js, React, Neo4j, Elasticsearch, semantic ranking).	System modularity; Scalability of platform across disciplines
	9.c	Enhances educational infrastructure through intelligent, adaptable tools that support more efficient and inclusive digital learning environments.	System responsiveness and availability; Number of distinct users; Recommendation coverage

Acknowledgements

Completing this thesis has been one of the most challenging yet deeply rewarding journeys I've ever experienced. I am sincerely grateful to everyone who supported and encouraged me through this stressful, exhausting, and demanding process.

First and foremost, I would like to thank my supervisor, Rui Pinto, for his passion for research and for the topic that so dearly drew me to this thesis. His guidance, constructive feedback, and continuous support throughout every stage of this work proved to be invaluable. Although his expectations were often ambitiously high, I'm grateful for the challenge, which ultimately pushed me to grow as both a developer and researcher.

To my friends and colleagues, from both high school and university, thank you for the countless moments of support, shared struggles, and much-needed distractions. A special thanks goes to my fellow members at NIAEFEUP, whose motivation, collaboration, and encouragement made a real difference in this journey. I especially appreciate your involvement during the evaluation phase of this thesis; without your help, motivation, and input, this work would not have been possible.

To my Erasmus friends, thank you for reminding me of the joy of discovery, for all the spontaneous conversations, cultural exchanges, and unforgettable moments that helped me grow not only academically but also personally. A special thank you to the *Bolonhesas* friend group; your friendship extended far beyond Erasmus, and our continued connection means the world to me. The memories we created still inspire me and gave me the energy and perspective I needed when returning to this thesis.

To my family, especially my dear mother, thank you for always believing in me. Your unconditional love, unwavering support, and emotional strength gave me the stability I needed to keep going. I owe this accomplishment to your love and confidence in me.

My heartfelt thanks to all of you.

Sofia Merino Costa

"I have no special talent.
I am only passionately curious."

Albert Einstein

Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
API	Application Programming Interface
BM25	Best Matching 25
CRUD	Create, Read, Update, Delete
DB	Database
HCD	Human-Centered Design
HEI	Higher Education Institution
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IR	Information Retrieval
JSON	JavaScript Object Notation
KM	Knowledge Management
KMS	Knowledge Management System
LMS	Learning Management System
ML	Machine Learning
NLP	Natural Language Processing
OWL	Web Ontology Language
PDF	Portable Document Format
RQ	Research Question
SQL	Structured Query Language
SUS	System Usability Scale
UI	User Interface
US	User Story
URL	Uniform Resource Locator

Chapter 1

Introduction

1.1 Context

Engineering education involves complex and ever-evolving fields of knowledge that require students to engage with a wide range of technical content. The continuous advancement of technology and engineering practices leads to a growing volume of educational resources, including lecture materials, technical documentation, scientific articles, and online tutorials. Navigating this abundance of information can be challenging, particularly when students must integrate concepts across multiple domains such as mathematics, computer science, and applied engineering.

In this context, efficient access to relevant knowledge plays a critical role in supporting students' academic success and engagement. However, traditional systems for managing educational content often lack the structure, adaptability, and intelligence required to assist learners in finding and organizing the most pertinent resources. These systems typically rely on static repositories, limited keyword-based search, and generic content listings that do not account for individual learning needs or content relevance. As a result, students may encounter information overload, redundant resources, or difficulty locating the materials best suited to their current learning goals.

Knowledge management systems (KMS) have been proposed as a technological solution to capture, organize, and distribute knowledge within both organizational and educational contexts. In theory, these systems should enhance knowledge discovery, support resource sharing, and enable users to interact with content in meaningful ways. In practice, however, many KMS implementations in academic settings fall short. They often prioritize document storage over usability and learning effectiveness, offering limited support for personalization, dynamic exploration, or progress tracking. Moreover, such systems are frequently designed with administrative goals in mind rather than student-centered learning processes.

In engineering education specifically, where understanding is built through the accumulation and contextualization of interrelated knowledge, the shortcomings of traditional KMS become more pronounced. Students must not only retrieve accurate information but also connect it to existing knowledge, apply it in problem-solving scenarios, and adapt it to evolving academic or

professional needs. Without intelligent tools that can facilitate this process, learners are left to navigate complex resource ecosystems with minimal guidance or support.

1.2 Motivation

The challenges outlined above highlight a growing need for more effective and intelligent systems that support learning in engineering education. As digital content continues to expand, students require not just access to resources but structured guidance in navigating and engaging with that content. Simply providing a collection of documents is no longer sufficient. At this time and age, learners benefit from tools that adapt to their needs, anticipate their learning goals, and streamline the discovery of relevant materials.

Intelligent technologies such as semantic search and recommendation systems offer promising solutions. These tools can analyze user behavior, content relationships, and contextual factors to deliver tailored learning experiences. In educational settings, this means students could receive recommendations for resources based on their progress, preferred content types, or learning objectives. Such capabilities are commonly used in commercial environments like e-commerce and social media applications but remain underutilized in academic platforms.

In the context of engineering education, integrating intelligent features into a KMS could bridge the gap between resource availability and educational value. A system that understands user context can prioritize content that aligns with both short-term goals (e.g., completing an assignment) and long-term development (e.g., mastering a topic). Additionally, features such as progress tracking and interactive assessments can reinforce learning and provide meaningful feedback loops.

This thesis is motivated by the opportunity to combine these capabilities into a cohesive platform designed specifically for students and educators in engineering disciplines. By exploring how semantic retrieval, personalization, and modular content organization can be embedded within a user-centered KMS, the work aims to contribute a practical and scalable solution to ongoing challenges in educational resource management.

1.3 Problem

Despite the availability of digital platforms and content repositories, students and educators often face difficulties in managing and retrieving relevant knowledge. Existing systems tend to treat content as static information, presenting it through generic lists or unstructured databases. These approaches do not support the evolving nature of learning or accommodate individual differences in background knowledge, learning pace, or educational goals.

Traditional keyword-based search mechanisms remain a dominant feature in many platforms, but they offer limited semantic understanding and contextual relevance. As a result, students may receive long, unordered lists of results that do not consider their intent or academic progression.

This can lead to frustration, cognitive overload, and disengagement, particularly when learners are forced to manually sift through irrelevant or redundant materials.

Additionally, existing platforms often lack built-in mechanisms for structured learning. Features such as module-based progression, milestone tracking, or adaptive content sequencing are rarely implemented. Without such elements, students are left with a fragmented experience that fails to guide them through coherent learning paths. Similarly, most systems do not utilize interaction data to improve content recommendations or reflect learner progress.

These challenges are particularly critical in engineering education, where building on what you've learned and putting it in context is often necessary for understanding. When systems cannot adapt to the learner's needs or guide them effectively, opportunities for deeper engagement and skill development are lost. Addressing these gaps requires a platform that combines intelligent content retrieval with personalized learning support. This platform should be able to respond to how students behave, give useful suggestions, and organize content in ways that meet real educational needs.

1.4 General Goals

The primary goal of this dissertation is to develop an intelligent knowledge management system that improves how engineering students and educators access, organize, and engage with educational resources. This thesis seeks to demonstrate how personalized information retrieval, semantic technologies, and user-aware design principles can be integrated into a single platform that enhances the learning experience.

The following general goals guided the development of the system:

- Investigate the use of semantic web and recommendation techniques to improve the relevance and precision of retrieved educational content.
- Develop a modular and scalable web-based platform that allows students and educators to organize and explore knowledge through structured learning modules.
- Incorporate features that enable personalized recommendations based on student preferences and interaction history.
- Implement mechanisms for tracking progress within learning paths to support continuous and goal-oriented learning.
- Ensure that the system remains accessible and adaptable by using open-source tools and lightweight, cloud-compatible technologies.
- Align the platform's development with current challenges in engineering education, ensuring its usability and relevance for both students and educators.

These goals were shaped by the following research questions, which defined the overall direction of the platform and guided the approach taken during the design and implementation:

- **RQ1:** What design strategies can be applied to implement a knowledge management system that accommodates the varying needs of students and educators in engineering education?
- **RQ2:** How can intelligent technologies enhance information retrieval, personalization, and resource accessibility in knowledge management systems for engineering education?
- **RQ3:** What are the existing gaps and limitations in knowledge management systems for engineering education, and how can they be addressed through innovative approaches?
- **RQ4:** How can the effectiveness of an intelligent KMS be evaluated in terms of user satisfaction, learning outcomes, and system performance in engineering education?

By aligning the design and development of the system with these research objectives, the implemented platform serves as a proof of concept of how intelligent features and learner-oriented tools can improve knowledge accessibility, organization, and discovery in educational environments.

1.5 Dissertation Structure

In addition to the introduction, the chapters of this dissertation are structured as follows:

- **Chapter 2 – Background:** Introduces the conceptual and technological foundations of the project, covering knowledge management systems, engineering education, Education 5.0, and intelligent technologies such as semantic search and recommendation systems.
- **Chapter 3 – Literature Review:** Describes the methodology used for literature collection and screening, and synthesizes related research that informed the system's design and innovation focus.
- **Chapter 4 – Implementation:** Details the architecture and core components of the developed platform, including semantic search, graph-based recommendations, learning paths, and interface features.
- **Chapter 5 – Evaluation:** Presents the methods used to assess the system's performance, usability, and educational impact, supported by quantitative and qualitative metrics.
- **Chapter 6 – Discussion:** Interprets the key findings in relation to the research questions, discusses limitations and methodological reflections, and explores broader implications.
- **Chapter 7 – Conclusion:** Summarizes the main contributions of the project and outlines directions for future research and development.

Chapter 2

Background

This chapter establishes the conceptual and technological foundations that support the design and implementation of the intelligent KMS proposed in this thesis. It introduces the core principles of knowledge management, its relevance in engineering education, and the intelligent features that enhance knowledge discovery and personalization.

2.1 Knowledge Management and Knowledge Management Systems

Knowledge exists in two primary forms: explicit knowledge, which is formal, codified, and easily shared (e.g., documents, manuals), and tacit knowledge, which is personal, experience-based, and harder to articulate [44]. Managing both forms of knowledge is essential for organizational learning and innovation.

According to Davenport and Prusak [9], knowledge management (KM) is the process of capturing, distributing, and using knowledge effectively within an organization. It involves a systematic approach to identifying valuable knowledge assets and ensuring they are accessible and applicable across various organizational contexts. Alavi and Leidner [2] further emphasize that KM encompasses activities related to knowledge creation, storage, transfer, and application, often supported by digital technologies.

A knowledge management system refers to a technology-based system designed to support these KM processes. In the context of education, a KMS serves as a digital infrastructure that facilitates the creation, organization, and dissemination of institutional knowledge, including teaching materials, curriculum designs, best practices, and research outputs [8, 55]. By making this collective knowledge accessible to educators and students, KMSs promote collaboration, innovation, and institutional memory. They play a critical role in preserving pedagogical expertise and improving decision-making in academic planning and delivery.

2.1.1 Knowledge Management Life Cycle

The knowledge management life cycle refers to the systematic stages through which knowledge is created, organized, shared, and applied within an organization. As the field of KM has evolved,

various models have been proposed to guide effective knowledge practices that support innovation, informed decision-making, and continuous learning. These models have become increasingly sophisticated, reflecting both the complexity of modern organizations and the integration of KM into digital platforms like Knowledge Management Systems.

One of the earliest and most influential models is the Knowledge Management Process Framework by Bukowitz and Williams [4], which is illustrated in Figure 2.1.

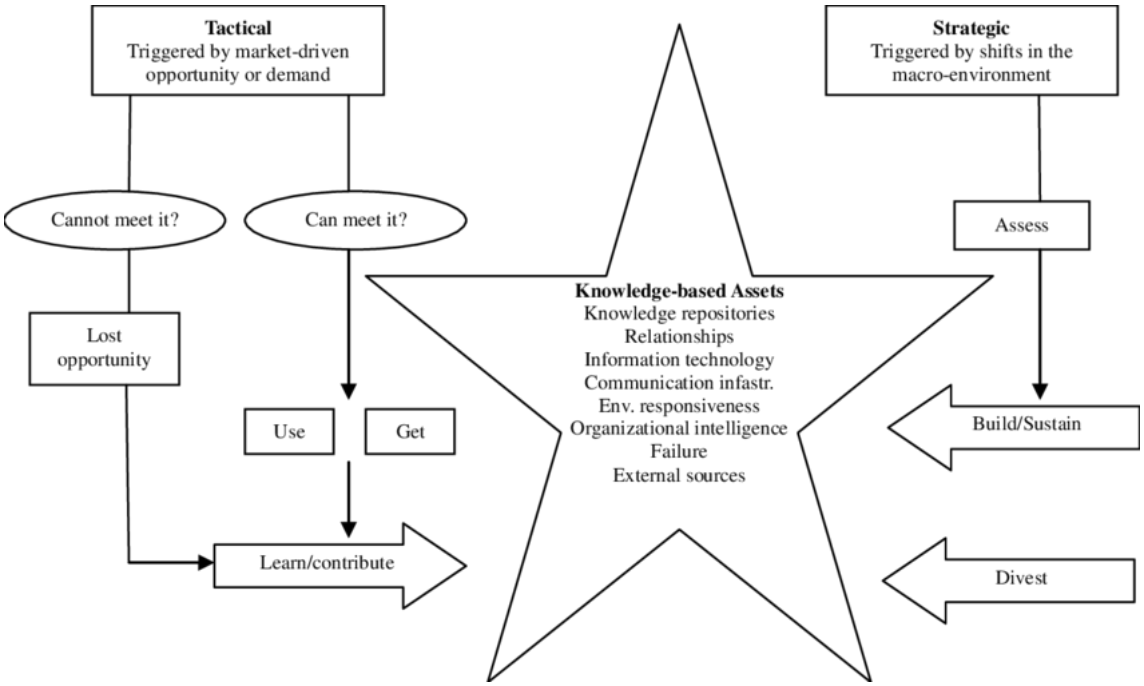

Figure 2.1: Bukowitz and Williams knowledge management process framework. [4]

This framework distinguishes between tactical knowledge processes that address immediate operational needs and strategic processes that build long-term capabilities. It includes interconnected phases such as use, get, learn/contribute, assess, build/sustain, and divest that align knowledge activities with business value creation and strategic goals.

Building upon this foundation, McElroy proposed a more innovation-centered view in his Knowledge Life Cycle (KLC) model [38], shown in Figure 2.2.

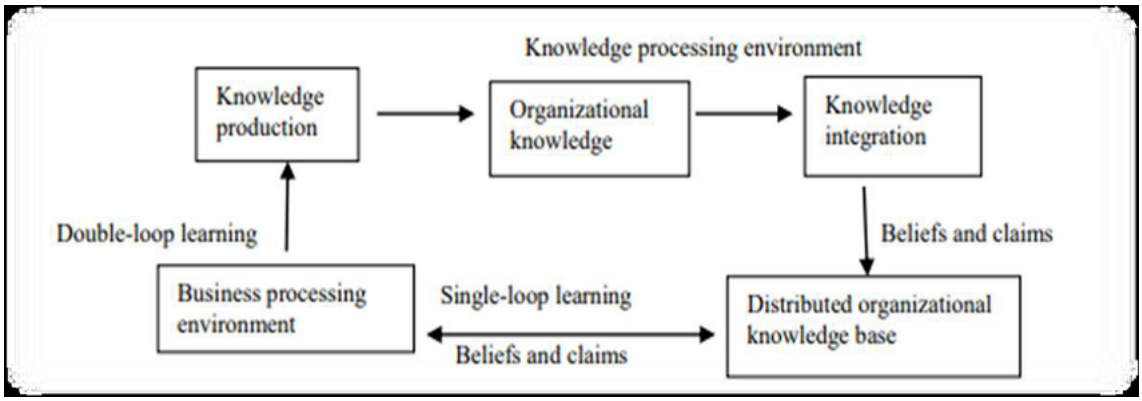

Figure 2.2: McElroy's knowledge life cycle [38]

The model distinguishes between a knowledge processing environment and a business processing environment, connected through both single-loop and double-loop learning. It illustrates how organizational knowledge is created, integrated, and stored in a distributed knowledge base. Knowledge production responds to gaps identified in business processes, while beliefs and claims feed both learning loops. This cyclical interaction enables continuous innovation, learning, and refinement of organizational knowledge.

A more recent contribution is the Knowledge Management Cycle (KMC) by Evans *et al.* [18], depicted in Figure 2.3.

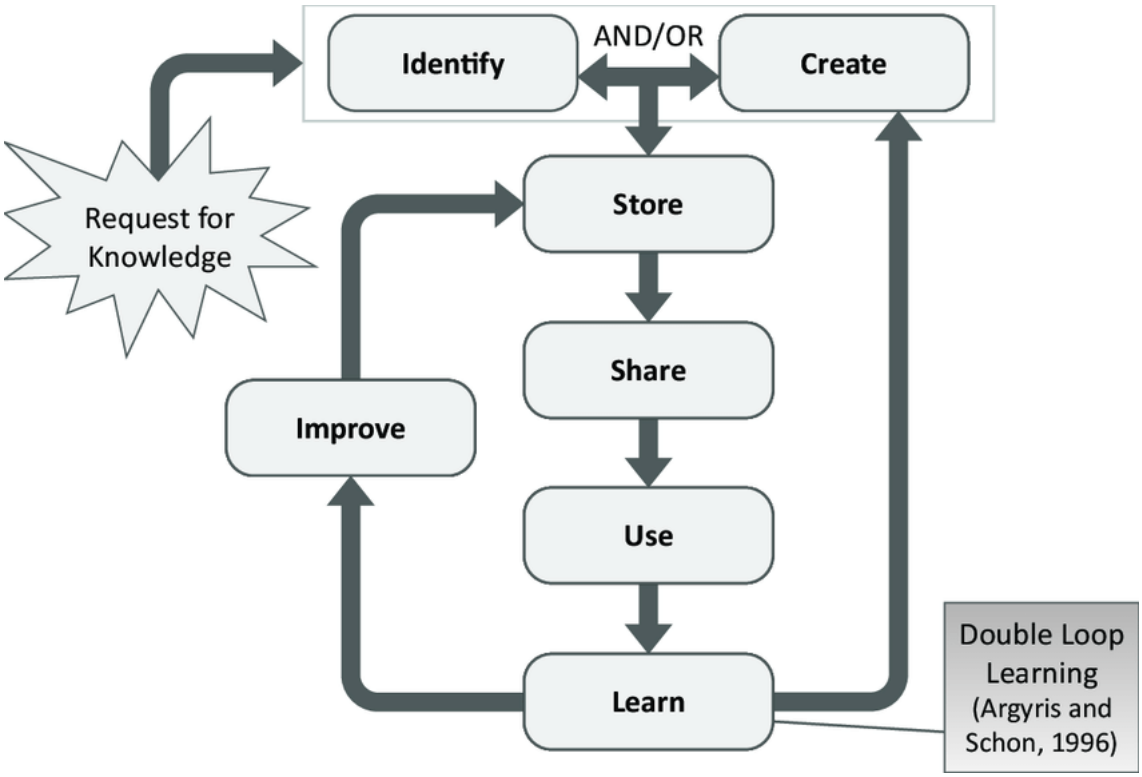

Figure 2.3: The knowledge management cycle model [18]

This model outlines seven iterative phases: identify, store, share, use, learn, improve, and create, with an emphasis on feedback loops and continuous refinement. It is particularly suited for environments like education, where knowledge is collaboratively generated, frequently updated, and context-dependent.

Despite their structural differences, all three models share core phases such as knowledge identification, creation or acquisition, storage, sharing, and application. What differentiates them is their emphasis: Bukowitz and Williams [4] focus on strategic alignment, McElroy [38] emphasizes innovation and knowledge production, while Evans *et al.* [18] promote iteration and learning within a continuous cycle.

In dynamic contexts like higher education, these models are best interpreted as non-linear and adaptable. For instance, a teacher might gain insights through classroom interactions, apply them in future lessons, and contribute new learning materials, which demonstrates the fluid and recursive nature of the KM cycle.

Designing a KMS for education, therefore, requires a flexible interpretation of the KM life cycle. Since educational knowledge is often tacit, collaborative, and evolving, the system should not only enable structured storage and retrieval but also foster co-creation, reflection, and continuous improvement.

By drawing on the strengths of these foundational models, educational institutions can develop KMS platforms that support both operational knowledge tasks and broader strategic goals such as teaching innovation, curriculum development, and institutional learning.

2.1.2 Knowledge Management Key Components

Effective knowledge management depends not only on life cycle stages but also on the enabling components that support each stage. These components form the foundation of any KMS, ensuring that knowledge is created, organized, and applied in a strategic and sustainable manner.

Based on a comprehensive literature review by Usman *et al.* [53], six key components have been identified as critical for implementing KM across various organizational contexts, including higher education institutions: people, processes, technology, information, governance, and strategy. These elements are interdependent and must work in concert to ensure successful KM practices.

- **People** are the core actors in any KM initiative. They serve as knowledge creators, consumers, and collaborators. In educational contexts, this includes faculty members, students, and administrative staff. Educators generate and curate pedagogical knowledge, while students contribute through projects, feedback, and peer learning [15, 25].

- **Processes** define the activities and workflows that enable knowledge to be captured, stored, disseminated, and applied. Well-designed KM processes ensure transparency, consistency, and adaptability, which are key for academic environments where knowledge evolves rapidly [24].

- **Technology** provides the infrastructure for KM. This includes platforms for content storage, collaborative tools, search engines, and analytics dashboards. In higher education, a KMS often integrates with existing learning platforms to support efficient content access and sharing [1, 25].
- **Information** refers to the structured content that is made available to users. This includes course syllabi, lecture recordings, academic articles, and institutional documentation. KM ensures that such content is contextually organized and readily accessible [15].
- **Governance** involves the policies and frameworks that oversee KM activities. This includes access control, data privacy, knowledge validation, and intellectual property rights. Effective governance ensures ethical and strategic use of knowledge assets [27, 71].
- **Strategy** ensures that KM efforts align with institutional goals. In universities, this might involve promoting interdisciplinary research, enhancing student learning outcomes, or fostering innovation in teaching practices. KM strategy must be agile and continuously adapted to evolving educational priorities [65].

Together, these components form the structural backbone of a knowledge management system. In higher education, they not only enable the management of explicit knowledge but also support the collaborative creation and sharing of tacit knowledge, such as teaching techniques and experiential learning. A component-based understanding of KM allows institutions to design systems that are both technically robust and pedagogically meaningful, contributing to improved educational outcomes, resource accessibility, and innovation.

2.1.3 Knowledge Management in Education

In the context of higher education institutions (HEIs), the needs and opportunities for knowledge management are closely aligned with those of business organizations [6]. It is increasingly recognized as a vital strategy for enhancing educational quality, organizational efficiency, and institutional learning. While the foundational concepts of KM originate in business environments, their relevance in academic contexts has grown due to the complexity of knowledge-intensive activities such as teaching, research, and administration.

Educators continuously generate diverse knowledge forms: instructional materials, curriculum designs, assessment strategies, and reflective practices. However, these assets often remain siloed within departments or individual staff, leading to missed opportunities for reuse, collaboration, and innovation [31]. KM initiatives in academia aim to systematize the capture, organization, and dissemination of this knowledge to support continuous improvement and strategic alignment. It connects educators, students, and administrators within a seamless knowledge-sharing ecosystem designed to facilitate the collection, sharing, and application of knowledge [20].

Kidwell, Vander Linde, and Johnson [31] describe how corporate KM practices can be adapted to HEIs, especially to enhance cross-departmental collaboration, institutional memory, and curriculum development. They argue that a well-implemented KMS can help universities avoid redundant efforts, support knowledge continuity during faculty turnover, and increase responsiveness to academic and societal needs.

Similarly, Dalkir [8] highlights that educational institutions must address both explicit and tacit knowledge. Explicit knowledge (e.g., syllabi, policy documents, recordings) can be easily stored and retrieved, while tacit knowledge (e.g., teaching experience, informal techniques) requires more dynamic, community-based approaches to capture and share. This is particularly relevant for pedagogical improvement and faculty development initiatives.

Beyond institutional knowledge sharing, KM plays a strategic role in engineering education, where students are expected to master both technical knowledge and applied problem-solving. Engineering programs face persistent challenges: gaps in foundational skills (e.g., mathematics, physics), varied learning styles, and uneven exposure to modern technologies [54]. These challenges can be mitigated by KM systems that provide personalized access to learning resources, collaborative spaces for students and faculty, and mechanisms to reuse and improve teaching content over time.

Research by Singh Sidhu *et al.* [54] examines engineering students' difficulties in transitioning from theoretical instruction to practical application. They found that knowledge gaps are often compounded by traditional lecture-based delivery, which assumes uniform prior knowledge among students. To address this, the study suggests that additional support mechanisms, such as modular resources, feedback loops, and peer learning, can help tailor instruction to individual needs.

In this context, a well-structured KMS enables the redistribution of successful practices, facilitates interdisciplinary teaching, and preserves institutional expertise. KM in education is not simply about archiving documents but about fostering a culture of knowledge sharing, aligning with the collaborative and lifelong learning ethos of academia.

2.2 Digital Transformation in Education

Engineering education faces ongoing challenges related to content relevance, student diversity, and the rapid evolution of technological fields. Digital transformation offers a strategic approach to address these issues by enhancing how knowledge is created, delivered, and managed across learning environments.

Knowledge management systems play a pivotal role in this transformation by facilitating the organization and dissemination of educational content. When combined with collaborative tools and intelligent recommendation engines, these systems support personalized learning pathways that adapt to individual student needs and learning histories [8, 12]. Furthermore, their dynamic architecture enables continuous updates to instructional materials, helping institutions align curricula with emerging industry demands and technological trends.

2.2.1 The Evolution of Information Systems to Knowledge Management Systems

Educational institutions have long relied on information systems to manage their operations and resources more effectively. One such system is the Education Management Information System (EMIS), defined by UNESCO as a comprehensive platform for collecting, integrating, processing, and disseminating data to support decision-making, planning, and management at all levels of the education system [61]. Shah [50] highlights the value of Management Information Systems (MIS) in improving school administration, noting enhancements in information accessibility, resource efficiency, and reporting quality.

As the digital transformation of education progressed, Learning Management Systems (LMS) emerged to support the instructional side of education. Platforms like Moodle, Blackboard, and Canvas have become essential tools for organizing course materials, managing assessments, and enabling communication between educators and learners, especially in hybrid and remote learning contexts.

However, while LMS platforms are effective for content delivery, they often fall short in addressing the needs of personalized and adaptive learning. Most are based on fixed course structures and lack the flexibility to support diverse learning styles, cross-disciplinary engagement, or dynamic content adaptation. These limitations are especially evident in the context of modern educational goals that emphasize learner-centered approaches, collaboration, and continuous improvement.

To address these limitations, knowledge management systems offer a more holistic and adaptive solution. Unlike traditional LMS, KMS are designed not only for delivering content but also for managing the creation, sharing, and reuse of knowledge across educational environments. They support features such as intelligent content recommendations, dynamic learning paths, semantic search, and collaborative tools tailored to user profiles and learning progress [8, 20]. These capabilities enable KMS to better support interdisciplinary learning, knowledge co-construction, and context-aware retrieval.

This evolution aligns with the goals of Education 5.0, which emphasizes personalized, technology-enhanced, and skill-focused learning experiences [14, 62]. KMS, when well-designed, can bridge the gap between static content delivery and the dynamic, learner-centered ecosystems required for future-ready education.

2.2.2 Foundations of Education 5.0 and Future Learning Needs

Education 5.0 reflects the shift towards a more progressive vision of learning that goes beyond knowledge acquisition to emphasize human-centric, technology-integrated, and values-driven education. It builds on the principles of previous industrial and educational revolutions, particularly Industry 4.0 and Education 4.0, placing greater emphasis on sustainability, emotional intelligence, ethics, inclusion, and digital fluency [14, 62]. The goal is to prepare learners not only for the workforce but also to become socially responsible and innovative contributors to a rapidly changing world.

In this paradigm, students are viewed as co-creators of knowledge, and education systems are expected to be adaptive, flexible, and responsive to diverse learner needs. Knowledge management systems, when integrated with intelligent technologies, can support this transformation by creating dynamic, personalized, and collaborative learning environments. These systems facilitate content customization, self-paced progression, and real-time feedback, which are essential to foster lifelong learning and support diverse cognitive and socioemotional profiles [8, 12].

Advanced tools such as intelligent recommendation engines, learning analytics, semantic search, and knowledge graphs enable KMS to deliver relevant learning materials based on a student's profile, preferences, and learning history. These systems can also help educators monitor engagement, identify struggling learners, and adapt instruction accordingly. Moreover, collaborative platforms and digital repositories allow for interdisciplinary teamwork and the reuse of institutional knowledge, aligning with Education 5.0's emphasis on problem-solving, critical thinking, and creativity.

Visualization and simulation technologies further enhance comprehension by transforming abstract engineering concepts into interactive, visual experiences. These tools are particularly valuable in STEM education, where students often struggle with conceptual understanding [54]. Adaptive learning environments, in turn, personalize the pace and depth of content delivery, ensuring that students can advance according to their individual progress and mastery level.

By embedding the principles of Education 5.0 into intelligent KMS architectures, educational institutions can foster more inclusive, engaging, and skills-oriented learning ecosystems. This approach not only addresses the limitations of traditional one-size-fits-all models but also empowers students to become agile thinkers capable of addressing complex societal and industrial challenges in an increasingly interconnected and digital world.

The shift from traditional educational systems to knowledge-based, intelligent, and adaptive environments marks a profound transformation in engineering education. By integrating the principles of Education 5.0 with advanced KMS architectures, educational institutions can offer more inclusive, flexible, and future-ready learning experiences.

2.3 Intelligent Features in KMS

The integration of intelligent systems into knowledge management has significantly transformed KMS from static repositories into dynamic, adaptive platforms capable of supporting complex educational needs. The application of AI-driven techniques allows KMS to manage not just the storage of knowledge but also its intelligent organization, retrieval, and utilization [42]. Intelligent systems enable decision-making, pattern recognition, and contextual awareness in knowledge-driven environments [43].

Modern intelligent KMS leverage machine learning (ML), natural language processing (NLP), and knowledge graphs to offer personalized, context-aware learning experiences. These technologies enable systems to adapt to individual users, streamline information discovery, and deliver tailored recommendations, making them especially suitable for supporting the goals of Education 5.0 and other learner-centered educational paradigms.

2.3.1 Adaptive and Personalized Learning Paths

Adaptive and personalized learning paths represent a significant shift in modern education, using artificial intelligence (AI) to tailor the learning experience to each student's individual needs. These systems adjust content delivery, pacing, and assessment methods in real time, which has been linked to improvements in academic performance, engagement, and knowledge retention [58]. Integrating cognitive psychology principles with AI algorithms plays a key role in this process, enabling learning environments to respond to learners' preferences, cognitive styles, and prior knowledge.

Recent studies highlight the importance of learner modeling and technological adaptability in supporting personalization. Systems that take into account student preferences, learning histories, and performance data are more effective in guiding learners through tailored educational paths [67]. The widespread use of smart devices, along with advances in AI and cloud computing, has further enhanced the ability to deliver personalized learning experiences on demand.

To keep up with these advancements, educational technologies are increasingly combining both technology-driven and human-centered approaches. This balance is particularly relevant in the context of Education 5.0, where personalization, collaboration, and social impact are key goals. In this context, AI is not just a backend engine, it becomes an active part of the learning environment, helping educators and students make more informed and adaptive decisions [33].

Bringing these capabilities into knowledge management systems greatly improves their potential to offer adaptive, personalized learning experiences. When systems are designed to align with learners' needs and behaviors, they become more effective at supporting diverse educational goals and preparing students for an evolving, technology-driven world.

2.3.2 Graph-Based Recommendation System

Recommendation systems are a core component of modern digital platforms, designed to assist users in discovering relevant content by predicting preferences based on past behavior and item characteristics. Generally, there are two primary types of recommendation systems: collaborative filtering and content-based filtering.

Collaborative filtering relies on user behavior, such as past ratings or interactions, to suggest new items. It assumes that if users A and B liked similar items in the past, A may like items B has rated highly but A hasn't seen yet [49]. This approach requires substantial interaction data and can suffer from cold-start problems when dealing with new users or items.

Content-based filtering, on the other hand, recommends items that are similar to those the user has previously engaged with, based on features or metadata (e.g., categories, tags). This method works well for new users with little behavioral data but requires detailed item descriptions and may limit diversity in recommendations [35].

To overcome the limitations of these individual approaches, hybrid recommendation systems combine both collaborative and content-based filtering. They integrate multiple data sources and models to improve accuracy and robustness, particularly in sparse or dynamic environments [5].

Graph-based recommendation systems have emerged as a powerful paradigm that generalizes and extends traditional recommendation techniques. These systems represent users, items, and their interactions as nodes and edges in a graph structure. This allows them to model higher-order relationships, such as user-item-tag triads, co-purchase networks, and shared attributes across domains. Graph representations support richer semantic connections and enable *multi-hop reasoning* for more explainable and personalized recommendations [37].

Neo4j, a popular graph database, is frequently used in educational or knowledge management platforms to power recommendation engines. It enables efficient querying of relationship paths between users and resources using its Cypher query language. In intelligent knowledge management systems, such a graph-based approach supports features like personalized learning resource suggestions, expert identification, or collaboration opportunities by leveraging the inherent structure of domain knowledge and user activity.

Additionally, graph-based recommenders offer explainability, a growing concern in AI systems. By visualizing the path that leads to a recommendation (e.g., user $\rightarrow$ resource $\rightarrow$ topic $\rightarrow$ related resource), the system can provide interpretable justifications that enhance user trust and system transparency [37].

In sum, graph-based recommender systems bridge the gap between collaborative and content-based methods by leveraging both user interaction data and rich contextual metadata. Their integration into knowledge management platforms, particularly in education, enhances personalization, relevance, and explainability, which are key goals of intelligent and learner-centered systems aligned with Education 5.0.

2.3.3 Semantic Search for Intelligent Retrieval

Traditional keyword-based search systems often struggle to capture the deeper intent behind user queries, leading to irrelevant or superficial results. Semantic search addresses this limitation by interpreting not just the words in a query but also the meaning and context of search terms, leading to more accurate and meaningful information retrieval.

Semantic search systems build upon techniques from natural language processing, machine learning, and knowledge representation to better understand user intent. Rather than relying solely on exact keyword matches, these systems can identify synonyms, infer relationships between concepts, and account for the broader linguistic context of both queries and documents.

One foundational method for enabling semantic search involves the use of ontologies, which are formal structures that define relationships among concepts within a domain. By linking related terms, ontologies allow search systems to retrieve relevant results regardless of terminological variation [70]. These structured representations enhance contextual understanding and facilitate reasoning across domain-specific knowledge.

Another key advancement in semantic search is the application of vector-based representations. Techniques such as Word2Vec [40], GloVe [47], and BERT [10] generate high-dimensional embeddings that capture semantic similarities between words, phrases, or entire documents. These embeddings enable the comparison of content based on meaning rather than surface form [48].

A prominent implementation of semantic search in practice is semantic Elasticsearch, which extends the capabilities of traditional Elasticsearch [16] by incorporating vector-based retrieval alongside keyword search. By combining dense vector search (semantic) and sparse vector search (keyword-based) in a hybrid search strategy, Elasticsearch enhances the accuracy and relevance of search results, particularly in domains where users express queries in diverse and often imprecise language. Using sentence transformer-based encoders and approximate nearest neighbor algorithms, Elasticsearch computes and retrieves documents that are semantically aligned with the user's intent, even when terminology diverges from indexed content [17, 48].

In the context of educational knowledge management systems, the integration of semantic search offers substantial benefits. By interpreting learner queries with greater contextual awareness, it ensures more relevant and timely access to educational resources, even when users lack precise domain knowledge or use informal phrasing. This fosters improved usability, supports personalized learning, and ultimately enhances the overall learning experience.

The concepts and technologies discussed throughout this chapter establish the theoretical and technical foundations for the intelligent knowledge management system proposed in this thesis. The next chapter builds upon this groundwork by analyzing related research and current technological solutions in the field.

Chapter 3

Literature Review

This chapter presents a literature review conducted to examine existing research related to the design, implementation, and evaluation of intelligent knowledge management systems in the context of engineering education. Particular attention is given to studies addressing personalized recommendation approaches, semantic search techniques, and system design strategies that support diverse educational workflows and user roles. The review also considers the integration of intelligent features and semantic technologies to improve information accessibility and usability in higher education environments.

Section 3.1 presents the overall methodology adopted for the review, combining systematic and exploratory strategies to guide the search and selection of literature. Section 3.2 defines the criteria used to include or exclude studies from the review. Section 3.3 explains the screening and selection process used to identify the most relevant studies. Finally, Section 3.4 synthesizes the key trends, technological approaches, and innovations found across the reviewed works.

3.1 Methodology

This literature review adopted a flexible, hybrid methodology that combined systematic search techniques with exploratory discovery to create a varied and thematically relevant body of literature. Given the interdisciplinary nature of the topic and the diversity of existing research, this approach allowed for both structured coverage of key themes and the inclusion of contextually relevant works that might not have been found using keyword-based search alone.

The review process followed three main stages:

- **Systematic Review:** Based on predefined research questions, targeted search queries were executed across selected academic databases. This phase focused on retrieving peer-reviewed studies in knowledge management, semantic technologies, and intelligent educational systems.

- **Exploratory Expansion:** Relevant citations from the systematically retrieved papers were examined using backward and forward snowballing techniques. Additionally, thematic exploration was carried out to include conceptual or theoretical works that contributed to the understanding of emerging technologies and design strategies.
- **Iterative Filtering and Thematic Grouping:** As the review progressed, sources were iteratively screened and categorized by topic, allowing for dynamic refinement of inclusion based on thesis scope, thematic saturation, and methodological variety.

Together, these phases allowed for the collection of a comprehensive and thematically rich dataset, highlighting related work, methodologies, tools, and research gaps in the development and evaluation of KMS for higher education. While the review primarily targeted literature addressing system design and intelligent features (aligned with RQ1–RQ3), it also incorporated works that discuss evaluation practices, contributing insights relevant to RQ4.

3.1.1 Collection Process

The literature collection process began with a systematic search of academic databases, structured around the primary research questions. Search queries were iteratively refined to increase specificity and relevance. The following subsections describe the databases consulted, the search queries applied, and the exploratory strategies used to expand the dataset.

Databases

Three academic databases were used as primary sources: IEEE Xplore, ACM Digital Library, and ScienceDirect. These platforms were selected due to their focus on engineering, computing, and educational technologies, providing access to high-quality, peer-reviewed publications relevant to the thesis scope.

Search Queries

To align with the research questions, targeted search strings were formulated using relevant keywords and Boolean operators. Table 3.1 summarizes the main queries used during the systematic phase of the review.

Table 3.1: Search queries for literature collection

Research Question	Search Query
RQ1	`"knowledge management system" AND education AND ("methodology" OR "approach" OR "framework")` `("knowledge management" AND "higher education") OR ("knowledge management" AND "engineering education")`

RQ2	`"knowledge management" AND education`
	`"knowledge management" AND "recommendation system"`
	`"intelligent" AND "knowledge management system" AND education`
	`"semantic search" AND ("information retrieval" OR "knowledge management") AND education`
	`"knowledge management systems" AND "education" AND "personalization"`

Exploratory Collection Process

After completing the systematic search, more studies were found through exploratory methods such as checking the references of selected papers and following related topics. This allowed the inclusion of contextually significant works that might not have been captured in the initial search. These articles were also reviewed to ensure thematic relevance and basic methodological quality before inclusion.

3.1.2 Management Tools and Collections

Zotero[1] was used as the primary reference management tool for organizing and storing the literature. Collected studies were grouped into thematic collections, such as current solutions, design methodologies, intelligent features, and semantic web technologies, to streamline the review and support structured analysis.

3.2 Eligibility Criteria

To ensure the relevance and quality of the selected literature, a flexible set of inclusion and exclusion criteria was applied during the review process. The primary goal was to identify studies that

[1] https://www.zotero.org/

explored intelligent features within knowledge management systems, with a particular emphasis on those designed for educational contexts.

Inclusion Criteria

Studies were included if they met the following conditions:

- **Publication Date:** Published in the last ten years (2014-2025), reflecting the most recent decade of research on semantic web, intelligent systems, and knowledge management systems in education.
- **Educational Context:** Focused on knowledge management systems implemented in or designed for educational settings, particularly higher education or engineering education.
- **Thematic Relevance:** Addressed intelligent system functionalities such as semantic web techniques, semantic search, recommendation engines, personalization, or user modeling, either in design or evaluation.
- **Language:** Available in English.
- **Conceptual Contribution:** Provided theoretical or architectural frameworks, evaluation strategies, or design approaches that inform intelligent, user-centered system design.

Exclusion Criteria

Studies were excluded if they met any of the following conditions:

- **Non-Educational Focus:** Focused exclusively on business or industrial KMS, or other domains without clear applicability to education.
- **Lack of Intelligent Features:** Lacked any technical, intelligent, or system-based dimension (e.g., purely pedagogical or conceptual papers with no connection to KM tools or platforms).
- **Low Technological Relevance:** Provided insufficient methodological detail or empirical grounding to support their inclusion in thematic analysis.

This criteria framework allowed for the inclusion of diverse sources, ranging from theoretical proposals and system implementations to empirical evaluations, while maintaining a clear focus on the intersection of knowledge management, intelligent features, and educational relevance.

3.3 Screening and Selection Process

A structured, two-stage screening process was followed to select the most relevant studies:

- **Initial Screening:** Titles and abstracts were reviewed to assess basic relevance to the research questions, specifically the inclusion of intelligent features in KMS for educational use.
- **Secondary Screening:** For articles passing the initial screen, introductions and conclusions were reviewed in detail to evaluate thematic alignment and methodological clarity.

Selected studies were prioritized based on their contribution to topics such as user-centered design in KMS, intelligent search and recommendation components, and system evaluation methods in educational platforms.

3.3.1 Results

The literature review resulted in a diverse and thematically focused collection of studies, gathered through a combination of structured database searches and exploratory citation tracking. The final selection includes a mix of conceptual works, system design studies, empirical evaluations, and methodological proposals.

Key topics addressed across the selected literature include semantic information retrieval, personalized recommendation systems, user modeling, intelligent support features, and collaborative or co-design practices in educational settings. These works vary in scope and depth but collectively reflect current directions and persistent challenges in the development of intelligent knowledge management systems for education.

The reviewed studies provide the analytical foundation for Section 3.4, which synthesizes key findings and highlights design approaches, technological innovations, and evaluation practices relevant to this thesis.

3.4 Literature Findings

This section synthesizes literature relevant to the research questions presented in Chapter 1, with particular emphasis on system design (RQ1), intelligent functionalities (RQ2), current limitations (RQ3), and evaluation practices (RQ4).

3.4.1 Knowledge Management in Education

In the last decade, numerous educational institutions have implemented knowledge management systems to address the growing complexity of managing learning resources, fostering collaboration, and improving institutional knowledge flows. These practical applications span a range of technological solutions, from semantic portals to intelligent learning systems, and reflect an evolving commitment to digital transformation in higher education.

Dneprovskaya and Shevtsova [12] developed a KMS tailored for smart education environments, designed to support the alignment of educational content with the competencies required by the digital economy. Their system integrates intelligent databases, co-authorship tools, and

knowledge networks to enhance the quality and adaptability of instructional materials. Using a methodology grounded in content modeling and intelligent metadata management, the system enables dynamic updates and collaborative content refinement by educators and subject-matter experts.

Yulistia *et al.* [69] proposed a knowledge transfer model adapted specifically for private higher education institutions. Expanding on the classic SECI model, they introduced two additional components, People and Policy, to better reflect institutional culture and operational constraints. Their implementation approach included conceptual modeling based on literature synthesis and empirical analysis of knowledge-sharing practices in academic settings. The resulting model emphasizes socialization and externalization processes within administrative and pedagogical contexts.

Gulnaz and Balova [23] implemented a semantic portal for managing scientific knowledge at the university level. The system uses an ontological information model to organize academic materials thematically and to personalize user access to content. Methodologically, this work involved ontology engineering and semantic web technologies, including RDF and OWL, to create a structured, machine-readable representation of institutional knowledge assets.

Fernandez-Nieto *et al.* [20] reported on the co-design of a KMS to support Communities of Practice among higher education teaching teams. Their methodology involved participatory design workshops where educators collaboratively defined system requirements and prototyped tool features aligned with their day-to-day teaching practices. This human-centered approach resulted in a system that better reflected pedagogical workflows, institutional values, and user expectations.

More recently, Skillify [11] was developed as an enhanced learning management platform integrating generative AI capabilities. The system introduced text-to-speech features to improve accessibility for visually impaired learners and included a course recommendation engine to guide users in skill development. Its architecture distinguished user, professor, and administrator roles and demonstrated how modular KMS design can accommodate diverse educational actors.

Imamah *et al.* [28] proposed a KMS that dynamically generates personalized learning paths based on student preferences and content difficulty, using ant colony optimization algorithms. Their system classifies learning objects by difficulty and continually adapts to user input, offering a scalable solution for individualized instruction in digital learning environments.

Together, these studies provide concrete evidence of how KMS are being applied to real educational challenges. Whether through semantic modeling, co-design, intelligent recommendation, or adaptive learning path generation, these systems demonstrate the practical relevance of KMS when grounded in pedagogical goals, institutional context, and user needs. They also highlight the diversity of methodologies used, from formal modeling and intelligent system design to participatory workshops and accessibility-driven engineering.

3.4.2 Human-Centered and Co-Design Approaches

While many educational KMS initiatives have struggled with adoption due to misalignment with user needs, recent implementations increasingly emphasize human-centered and co-design method-

ologies to ensure usability, contextual fit, and pedagogical relevance. These approaches involve active participation from educators, students, and institutional stakeholders throughout the design and development process, resulting in systems that better reflect real-world workflows and values.

One of the most comprehensive examples is provided by Fernandez-Nieto *et al.* [20], who applied participatory design workshops with teaching teams in higher education to co-create a knowledge management tool. Their methodology involved multiple sessions where participants explored current knowledge-sharing challenges, mapped pedagogical routines, and collaboratively defined tool functionalities. The resulting prototype incorporated features directly linked to educators' daily tasks, such as lesson planning, collaborative content curation, and reflective practice. The study demonstrated how co-design not only improves system usability but also fosters greater ownership and alignment with institutional teaching cultures.

Yulistia *et al.* [69] also emphasized user-centered principles in their adaptation of the SECI model for private higher education. Their model addressed institutional challenges such as weak knowledge-sharing culture and unclear governance structures by emphasizing People and Policy as explicit components of the system design. The inclusion of these social dimensions was based on case study analysis of real organizational practices, demonstrating that effective knowledge transfer depends as much on institutional context as on system features.

In the Skillify platform by Dhairya *et al.* [11], accessibility and learner inclusivity were central design goals. The system incorporated a text-to-speech module to address the needs of visually impaired users and offered adaptive recommendation tools to guide skill development. While the project focused on technical innovation, its modular design (differentiating user, professor, and administrator roles) was shaped by usability goals and feedback from pilot testing with diverse learner profiles.

These studies show that incorporating design methods rooted in empathy, participation, and iterative refinement leads to more effective and widely adopted KMS. Human-centered and co-design approaches help ensure that systems address not only technical requirements but also the social, cultural, and cognitive dimensions of educational knowledge work. When educators and learners are involved as co-creators rather than end-users, KMS become more sustainable, usable, and meaningful in the academic context.

3.4.3 Intelligent Technologies in KMS

While participatory design improves system alignment with user needs, technological advancements have also introduced new capabilities that can enhance system intelligence and adaptability.

The integration of intelligent technologies, such as machine learning, natural language processing, and knowledge graphs, into educational knowledge management systems has enabled more dynamic, personalized, and scalable solutions. These systems aim to address long-standing challenges in knowledge retrieval, learner engagement, and content adaptability, often by automating processes that were previously manual and limited in scope.

Galgotia and Lakshmi [21] explored the use of machine learning algorithms to support decision-making and interdisciplinary collaboration within higher education. Their implementation applied

predictive analytics, specifically decision trees, to model academic performance and guide faculty in aligning resources with institutional goals. Despite demonstrating promising results, the authors acknowledged limitations in scalability and the uneven integration of intelligent systems across academic departments.

Wen *et al*. [64] developed AI-KM, an intelligent desktop-based KMS designed to enhance personal knowledge management through semantic search, document summarization, and multi-view knowledge visualization. The system supported Markdown-to-Mind Map conversion, distributed deployment, and large language model–based interactions. It enabled users to conduct complex semantic queries and visualize knowledge clusters. However, its reliance on GPU-intensive operations and the absence of robust version control presented challenges for long-term usability in resource-constrained educational settings.

Liu *et al*. [34] proposed a personalized academic communication recommender system that leverages knowledge graphs and NLP to match users with high-quality scholarly resources. The system architecture combined *Neo4j* and *MySQL* to manage both semantic relationships and user interactions. Functional modules included user profiling, resource classification, and recommendation logic. Although the approach improved search relevance and recommendation precision, it required continuous metadata curation and graph maintenance, which may be burdensome for institutions lacking technical support.

Imamah *et al*. [28] designed a dynamic KMS that generates personalized learning paths using an ant colony optimization algorithm. Their system adapts to learners' knowledge preferences and classifies learning objects by difficulty, enabling an automated path construction mechanism. Evaluations showed a significant increase in learner performance compared to traditional e-learning, demonstrating the pedagogical impact of integrating intelligent adaptive logic into KMS design.

Sugumaran [56] presented a conceptual framework for incorporating semantic technologies, such as ontologies, RDF, and intelligent agents, into KMS. This architecture provides the foundation for intelligent classification, automated content reasoning, and context-aware retrieval. While theoretical, the framework remains relevant to modern implementations and has influenced subsequent system designs that emphasize machine-readable knowledge structures and interoperability.

Sharma *et al*. [51] also addressed adaptive knowledge delivery through an ontology-based e-learning platform. Their system dynamically adjusted lesson plans based on user context and performance, creating a flexible environment for individualized learning. The methodology involved concept-context graphs and semantic tracking of user behavior, providing an early model of context-sensitive content adaptation in educational settings.

Across these implementations, intelligent technologies have proven capable of enhancing KMS in multiple dimensions, from personalization and adaptive learning to semantic enrichment and automated recommendation. However, challenges remain in terms of infrastructure demands, metadata dependency, and technical complexity. These studies highlight the need for balancing innovation with usability and sustainability, particularly in educational institutions with limited resources or varying levels of digital maturity.

3.4.4 Semantic Search and Knowledge Retrieval

In parallel with personalization and intelligent features, the ability to locate and retrieve relevant information efficiently is essential. Semantic search technologies have become increasingly important in educational KMS as institutions face the challenge of retrieving relevant information from growing volumes of unstructured and heterogeneous content. Unlike traditional keyword-based approaches, semantic search systems use ontologies, knowledge graphs, and natural language processing to interpret user intent and uncover deeper relationships between concepts, thereby improving the precision and contextual relevance of search results.

Gulnaz and Balova [23] developed a semantic portal for scientific knowledge management in universities, leveraging ontological information models to structure academic content thematically. Their system supported personalized access to knowledge based on user profiles and enabled thematic classification of documents using semantic metadata. The model incorporated RDF-based structures to allow machine-readable content organization and demonstrated improved retrieval accuracy in academic environments.

Gammack *et al.* [22] introduced a semantic knowledge management system designed to retrieve content from complex technical documents containing heterogeneous data formats, such as text, figures, and tables. The system employed NLP models trained to associate structured and unstructured content, enabling users to perform context-aware semantic queries. Tested on real-world data collections, such as the IPCC Special Report, the system outperformed traditional search techniques in surfacing relevant content across different media formats.

Sheeba and Krishnan [52] developed a fuzzy ontology-based semantic search system tailored to educational KMS. Their system dynamically updated student profiles using fuzzy logic and semantic inference to personalize search results based on learning preferences. The integration of ontologies enabled more accurate content matching across domains, although the system required extensive profiling and complex rule sets to function effectively.

Kravchenko *et al.* [32] proposed a framework for semantic search, classification, and structuring in knowledge-intensive environments. Their model used ontologies and logical reasoning mechanisms to support automated knowledge integration across heterogeneous sources. Applied in the context of educational institutions, this system enabled improved knowledge discovery and reuse, with a focus on transforming data into actionable insights through semantic enrichment.

Sharma *et al.* [51] applied a concept-context graph methodology to enable adaptive knowledge retrieval in an ontology-based e-learning platform. Their approach allowed semantic interpretation of learner input and adjusted lesson paths dynamically based on contextual understanding. The system demonstrated how semantic modeling could enhance both content delivery and search, supporting more personalized and relevant educational experiences.

Sugumaran [56] provided a foundational framework for integrating semantic web technologies into KMS, including RDF schemas, OWL ontologies, and intelligent agents. While conceptual, this work has informed several later implementations by defining architectural components for automated knowledge classification, context-aware reasoning, and dynamic search.

Together, these implementations show that semantic search can significantly improve knowledge accessibility and relevance in educational KMS, particularly in interdisciplinary and data-rich environments. By combining NLP, fuzzy logic, and ontological reasoning, these systems enable users to retrieve content based on meaning rather than exact terminology. However, these benefits depend on the availability of high-quality metadata, domain-specific ontologies, and system architectures that support real-time semantic inference.

3.4.5 Recommendation Systems for Learning

Recommendation systems have become increasingly important within educational knowledge management systems, particularly in addressing issues of information overload and guiding learners toward relevant resources. These systems aim to personalize learning experiences, support adaptive content delivery, and enhance user engagement by leveraging learner profiles, behavior data, semantic structures, or graph-based relationships.

Lu and Feng [36] proposed a hybrid recommendation algorithm for AI-assisted learning platforms that combines graph convolutional networks (GCNs) with user behavior modeling. Their system captured real-time preferences and dynamically adapted recommendations based on user interaction history. The experimental results showed strong performance in accuracy, recall, and F1-score, validating the system's effectiveness in surfacing relevant resources. However, the reliance on real-time user tracking raised concerns about data privacy and computational overhead.

Liu *et al.* [34] developed a personalized academic recommendation system based on a knowledge graph architecture. Their system used *Neo4j* to model relationships between users, topics, and academic resources, while integrating natural language processing to improve the semantic interpretation of user queries. The platform included modules for user management, knowledge description, and recommendation logic. Although effective at increasing the contextual relevance of recommended content, the system required regular maintenance of graph structures and high-quality metadata to ensure long-term performance.

Hu *et al.* [26] addressed information overload in traditional KMS by integrating an intelligent recommendation module that filtered knowledge based on user context and task relevance. Their system used a rule-based matching mechanism combined with user profiles to enhance knowledge discovery. While their model showed improved relevance over generic search, it struggled to adapt to changes in user needs due to static profiling.

Imamah *et al.* [28] also embedded recommendation capabilities in their dynamic learning path system by classifying learning objects based on difficulty and aligning them with student preferences. The recommendation component, based on ant colony optimization, played a crucial role in constructing personalized paths. Their system outperformed conventional e-learning platforms in terms of learner performance and satisfaction, highlighting the pedagogical value of adaptive path-based recommendation strategies.

Skillify [11] further demonstrated practical integration of course recommendation with generative AI features. The platform provided tailored learning suggestions based on user skill gaps and learning goals. While not fully evaluated through large-scale deployment, the modular design

of Skillify showed the feasibility of embedding intelligent recommendation logic into LMS-style environments targeting diverse user roles.

These implementations demonstrate the practical value of recommendation systems in educational KMS, particularly when they incorporate graph-based reasoning, semantic analysis, or adaptive modeling. However, challenges persist in maintaining high-quality user models, managing sparse data in early system use (cold-start problem), and addressing privacy concerns related to behavior tracking. The literature underscores the need for balanced approaches that combine intelligent techniques with transparent, user-aligned design practices.

3.4.6 Limitations and Challenges in Current KMS

Despite the growing interest in knowledge management systems for education, many initiatives face persistent challenges that hinder adoption, usability, and long-term impact. These limitations are often interrelated, spanning technical, organizational, cultural, and methodological domains.

1. Misalignment with User Needs and Workflows. A recurring issue in educational KMS is the lack of alignment with the actual practices and routines of educators and learners. Fernandez-Nieto *et al*. [20] found that many institutional platforms fail to support how teaching teams naturally collaborate or share knowledge. Their co-design study identified overlooked aspects of pedagogical culture and routine that, when not accounted for, reduced the perceived usefulness and engagement with the system. Similarly, Touré *et al*. [59] observed that redesigning a platform without meaningful user involvement led to limited adoption despite added functionality.

2. Usability and Adoption Barriers. Educational KMS frequently suffer from poor usability and unclear user value, leading to abandonment even when technically functional. Early-generation systems were often built for administrative efficiency rather than user-centered knowledge practices. Even recent systems, like AI-KM [64], can be hampered by complexity, requiring high levels of technical literacy or training. Without intuitive interfaces and seamless integration into existing educational platforms, both students and educators may resist adoption.

3. Data Silos and Lack of Interoperability. Knowledge assets in educational institutions are often distributed across disconnected systems (e.g., LMS, libraries, repositories), making integrated knowledge management difficult. Many KMS struggle to interconnect with external platforms or institutional databases, limiting their ability to provide holistic views of available content. Karna *et al*. [30] addressed this issue by proposing a protocol-based recommendation model for interoperability, but real-world examples remain scarce.

4. Inadequate Personalization and Adaptability. Traditional KMS generally provide static, one-size-fits-all access to resources, failing to accommodate individual learner needs, preferences, or progression. Although newer systems incorporate recommendation engines or adaptive learning

paths (e.g., [28, 36]), many still rely on rigid user profiles or oversimplified behavior models. Cold-start problems, metadata sparsity, and lack of dynamic user feedback mechanisms continue to limit effective personalization.

5. Social and Cultural Barriers. In addition to technical gaps, organizational and cultural dynamics pose significant obstacles. Fernandez-Nieto *et al.* [20] found that educators are sometimes reluctant to adopt new platforms due to lack of trust, inadequate support, or unclear benefits. Yigzaw *et al.* [68] further highlighted how these challenges are exacerbated in developing regions, where infrastructure limitations, low digital literacy, and resistance to change hinder adoption.

5. Metadata Quality and Knowledge Representation Challenges. Effective semantic search and recommendation depend on rich, accurate, and structured metadata which are often lacking in educational content repositories. Ontology-based systems such as those by Gulnaz and Balova [23] or Liu *et al.* [34] demonstrate potential but also highlight the significant overhead required for initial ontology creation and ongoing maintenance. This limits scalability and long-term system viability, especially in institutions without dedicated semantic infrastructure or staff.

6. Infrastructure and Resource Constraints. Advanced KMS incorporating AI, NLP, or semantic reasoning often require computational resources and expertise that are unavailable in many educational contexts, particularly in under-resourced or developing regions. Wen *et al.* [64] and Galgotia and Lakshmi [21] noted that system demands, such as GPU processing or continuous data integration, can pose a barrier to broader adoption, even when the system's design is pedagogically sound.

7. Evaluation Gaps and Lack of Evidence. Many KMS implementations are not rigorously or systematically evaluated, particularly in terms of long-term learning impact, user experience, or institutional outcomes. Studies often focus on system architecture or prototype functionality rather than real-world usage, resulting in a gap between technical innovation and educational validation. As Vyas [63] and Yigzaw *et al.* [68] suggest, more longitudinal and mixed-method evaluations are needed to demonstrate value and refine system design.

8. Organizational and Cultural Resistance. Successful KMS deployment is not only a technical task but also a cultural one. Resistance to change, lack of institutional incentives, and distrust in digital tools can undermine even well-designed systems. Fernandez-Nieto *et al.* [20] emphasized that without strong stakeholder involvement and visible benefits, educators may not adopt systems consistently. In some regions, as Yigzaw *et al.* [68] noted, low digital literacy and inadequate support structures further intensify these barriers.

The limitations of educational KMS reflect deeper tensions between what systems are designed to do and what users need them to do. As educational environments grow more complex and data-driven, future systems must move beyond functional completeness to support pedagogical

alignment, cultural integration, and sustainable maintenance. Addressing these systemic issues is crucial to realizing the full potential of intelligent knowledge management in education.

3.4.7 Evaluation of KMS

Evaluating educational knowledge management systems requires a multidimensional approach that accounts for technical performance, user experience, and pedagogical impact. However, existing studies often focus on limited aspects, such as algorithmic accuracy or system architecture, while overlooking broader indicators of educational value, adoption, and institutional integration. The diversity of KMS objectives and implementation contexts has also led to varied and inconsistent evaluation practices across the literature.

In systems enhanced by intelligent features, such as personalized recommendations or semantic search, evaluation often focuses on quantitative metrics like accuracy, precision, recall, and F1-score. For instance, Lu and Feng [36] reported strong evaluation results for their hybrid recommendation algorithm using graph convolutional networks, achieving over 99% accuracy in recommending relevant learning resources. Similarly, Imamah *et al.* [28] measured learning gains and algorithm performance to demonstrate the effectiveness of their adaptive learning path system. These metrics are useful for comparing algorithm variants, but they may not capture educational relevance, long-term usability, or learner satisfaction.

Semantic retrieval systems are often evaluated using task-based or scenario-based testing to assess contextual relevance and user satisfaction. Gammack *et al.* [22] conducted case-based evaluations with domain-specific queries to validate the ability of their semantic model to retrieve information across mixed media formats. Their results demonstrated higher accuracy than traditional keyword search, particularly in retrieving technical content such as figures, tables, and contextual passages. However, these evaluations rarely include long-term studies of user learning outcomes or integration with curriculum goals.

Systems developed through participatory or human-centered design typically include qualitative assessments such as interviews, surveys, task completion analysis, or observational studies. Fernandez-Nieto *et al.* [20] evaluated their co-designed platform through user workshops and prototype testing, focusing on the tool's alignment with educators' workflows and its perceived usefulness. Touré *et al.* [59] conducted structured interviews and iterative usability testing during the redesign of ALEX, measuring improvements in system engagement and satisfaction. These studies highlight that usability is often as critical as functionality for long-term adoption.

In studies where systems are still at the design or prototyping stage, evaluation is often theoretical or based on expert review. Karna *et al.* [30] proposed an interoperable recommendation model using the OAI-PMH protocol but did not conduct real-world testing. Similarly, Sharma *et al.* [51] validated their ontology-based platform through a conceptual walkthrough, illustrating potential use cases rather than measuring system efficacy. These early-stage validations are valuable but underscore the need for follow-up empirical testing.

A recurring limitation in the literature is the lack of longitudinal studies that assess the impact of KMS on teaching practices, institutional knowledge flows, or student learning trajectories over

time. As noted by Vyas [63] and Yigzaw *et al.* [68], many studies remain narrowly focused on system design or algorithmic performance, with limited attention to educational impact, contextual relevance, or adoption sustainability. This gap is particularly evident in AI-enhanced systems, where technological novelty often overshadows classroom utility.

In summary, while a variety of evaluation strategies are used in educational KMS research, they often remain fragmented and context-specific. More comprehensive frameworks are needed to integrate technical validation with pedagogical effectiveness, user experience, and institutional alignment. Such frameworks should also consider longitudinal outcomes and the socio-cultural environments in which these systems operate, ensuring that evaluation reflects both system performance and its real-world educational value.

3.4.8 Summary of Key Insights

This chapter has reviewed the most recent and relevant research on knowledge management systems in education, with a particular emphasis on real-world implementations from the past decade. The literature reveals a steady evolution from static, content-centered platforms toward more dynamic, intelligent, and user-centered systems that attempt to align with pedagogical needs and institutional goals.

One of the clearest insights is that successful KMS in education must move beyond traditional models of information storage and retrieval. Systems designed without integration into teaching practices or collaboration cultures often struggle with adoption and sustainability. Studies that involved educators in the design process, such as those using co-design or human-centered approaches, showed better alignment with pedagogical workflows and more meaningful user engagement.

Technological advancements, especially in artificial intelligence and semantic web technologies, are expanding what KMS can do. From adaptive learning paths powered by optimization algorithms to knowledge graphs and NLP-enhanced recommendation engines, intelligent technologies offer significant potential for improving personalization, resource discovery, and learner support. However, these systems also introduce complexity—requiring high-quality metadata, consistent user profiling, and scalable infrastructure to be effective.

Semantic search emerged as a key enabler of meaningful information retrieval in educational KMS, helping users locate resources across disciplines and formats by understanding the conceptual intent behind queries. Ontology-based models and linked data approaches enhance contextual relevance, but they demand significant initial investment in knowledge modeling and ongoing curation.

Recommendation systems are increasingly central to the user experience in KMS, with graph-based, hybrid, and behavior-driven techniques enabling more targeted and adaptive content delivery. Still, many face issues such as the cold-start problem, privacy concerns, and metadata sparsity, which can limit their effectiveness in diverse and evolving educational settings.

Across the literature, recurring challenges include usability limitations, poor integration with institutional systems, fragmented data environments, and a lack of robust long-term evaluations. While some systems report promising pilot results, few have demonstrated sustained impact on teaching practices, learning outcomes, or institutional collaboration. These limitations underscore the importance of not only designing technically sophisticated systems but also ensuring pedagogical alignment, cultural fit, and continuous feedback loops.

The research conducted indicates a clear shift toward intelligent, user-responsive KMS in education. However, realizing their full potential depends on addressing ongoing technical, organizational, and evaluative challenges. These insights provide a critical foundation for this thesis, which aims to build on the strengths of existing systems while addressing gaps in personalization, semantic search, and contextual adaptability through the design of an intelligent KMS tailored to the needs of engineering education.

Chapter 4

Implementation

This chapter provides a detailed description of the implementation of the intelligent knowledge management system developed as part of this thesis. It is organized into several key sections: an initial overview of the system's requirements, components, and technologies in Section 4.1; the architecture and deployment model of the platform in Section 4.2; the underlying data model, including relational, graph, and search-based representations in Section 4.3; and the core features and functionalities from the user's perspective in Section 4.4.

Together, these sections offer both technical depth and explanations for the user-centric design decisions. The platform aims to improve engineering education by integrating intelligent capabilities such as semantic search, personalized recommendations, and modular learning paths. This chapter also discusses the main implementation challenges and the strategies used to address them, highlighting the iterative development process and the trade-offs between intended functionality and practical constraints.

4.1 Overview

The platform developed for this thesis serves as a prototype for an intelligent KMS designed to enhance engineering education.

The system is composed of four main components: a frontend interface for user interaction, a backend server that handles application logic, data management, and API communication, a data layer including a relational database, a graph-based recommendation system, and a semantic search engine, and external services such as cloud storage. This modular architecture ensures a seamless user experience and provides scalability for future growth.

Although the system is not fully adaptive or self-learning, it integrates foundational intelligent features such as semantic information retrieval and graph-based personalization, which represent an important step toward creating more advanced, intelligent systems. The solution demonstrates the value of integrating AI-driven techniques to foster autonomous learning, enhance contextual resource discovery, and improve the overall educational experience in engineering domains.

4.1.1 Technologies and Tools

The implementation of the platform involved a diverse set of technologies across the frontend, backend, data storage, and supporting services.

The frontend was developed using *React* [39] in combination with *JSX*, a syntax extension for JavaScript that facilitates the creation of component-based user interfaces. *Tailwind CSS* was employed for utility-first styling, enabling rapid and consistent layout design across the application.

On the backend, the platform was built using *Node.js* [46] along with the *Express.js* [45] framework to manage routing and application logic. Authentication was handled using *Passport.js* [29], and a RESTful API was implemented to facilitate communication between the frontend and backend layers.

Relational data management was initially supported through *PostgreSQL* [57], which was chosen for its familiarity, robustness and wide support for structured queries. To simplify access during development and ensure availability across environments, the database was hosted on *Aiven*[1], a managed cloud service. As the design of the recommendation system evolved, *Neo4j* [41] was integrated as a graph database to support recommendation logic, further detailed in Section 4.3.

For semantic search capabilities, the platform adopted a hybrid approach using *Elasticsearch* [16] combined with transformer-based semantic embeddings generated using the Hugging Face transformers library[2]. This setup allowed the system to go beyond traditional keyword matching and retrieve results based on contextual meaning, thereby improving the accuracy and relevance of search results.

Regarding external services, *Cloudflare R2* [7] was used to manage cloud storage of file-based resources, such as uploaded documents and images. Additionally, *Docker* [13] was used to containerize and manage certain components of the system, such as *Neo4j* and *Elasticsearch*, facilitating easier local development and consistent environment setup.

4.1.2 Requirements and Use Cases

The design of the platform was guided by a set of functional requirements derived from the needs of its target users, primarily students and educators in engineering education. These needs were identified during the initial literature review, which highlighted recurring challenges such as limited access to relevant learning materials, lack of personalized content, and insufficient support for self-directed learning.

The platform supports the following primary actors:

- **Student** – the main user who explores learning content, completes assessments, and receives recommendations.
- **Educator** – a contributor responsible for uploading resources, designing modules, and building learning paths and assessments.

[1] https://aiven.io/
[2] https://huggingface.co/models?library=transformers&sort=downloads

- **System** – the backend intelligent layer that provides personalized recommendations, semantic search, and progress tracking features.

To address the identified user needs, a set of functional requirements was expressed through user stories. These requirements capture expected system behavior from the perspective of its users and guided the design of platform features and interactions.

Table 4.1 presents the user stories defined for the platform. These stories are categorized based on user roles (students and educators) and served as the foundation for designing the platform's core features and interactions.

Table 4.1: Functional requirements of the platform expressed as user stories.

ID	User Story
User Requirements	
US1	As a new user, I want to sign up for the platform and select my role (e.g., student or educator) so that I can access the platform and perform role-specific interactions.
US2	As a returning user, I want to log in securely so that I can access my personal data and continue my learning journey.
Student-Oriented Requirements	
US3	As a student, I want to update my profile so that my recommendations and content can be personalized.
US4	As a student, I want to search for resources using keywords and phrases so that I can quickly find relevant learning materials.
US5	As a student, I want to bookmark resources to my personal library so that I can easily access them later.
US6	As a student, I want to see related or recommended resources, modules, and learning paths based on what I'm currently viewing so that I can discover additional relevant materials.
US7	As a student, I want to browse available learning materials so that I can find structured learning experiences on specific topics.
US8	As a student, I want to be able to access modules individually so that I can study specific topics without enrolling in a full learning path.
US9	As a student, I want to enroll in a learning path so that I can track my progress.
US10	As a student, I want to receive personalized recommendations based on my past interactions and preferences so that I can discover relevant materials more efficiently.
US11	As a student, I want to complete assessments to track my understanding and progress.

ID	User Story
US12	As a student, I want to view detailed descriptions of resources, including type, category, and tags, to ensure their relevance before accessing them.
US13	As a student, I want to view my learning progress, including completed modules and learning paths, so that I can track my achievements.
US14	As a student, I want to filter learning content by different criteria to find content that matches my current needs.
US15	As a student, I want to create my own study paths by grouping modules of interest so that I can personalize and organize my learning journey.
Educator-Oriented Requirements	
US16	As an educator, I want to upload new learning resources to the platform so that I can share my materials with students.
US17	As an educator, I want to create new modules by selecting and organizing resources from the library.
US18	As an educator, I want to create new learning paths by arranging existing modules into structured sequences.
US19	As an educator, I want to view and manage all the learning materials I have created in a dedicated space.
US20	As an educator, I want to create assessments as part of module creation to evaluate student knowledge.

These user stories provided a direct connection between real user expectations and the resulting system design. Each one informed and inspired the implementation of core functionalities and also contributed to the development of additional features that enriched the overall learning experience.

4.2 System Architecture

The proposed KMS is implemented as a modular, web-based platform designed to support key features for educational content management, recommendation, and personalized learning. Its architecture follows a layered structure that separates responsibilities across four main components: the frontend, backend, data storage, and external services. This organization promotes scalability, maintainability, and a clear division of concerns between user interaction, application logic, and data processing.

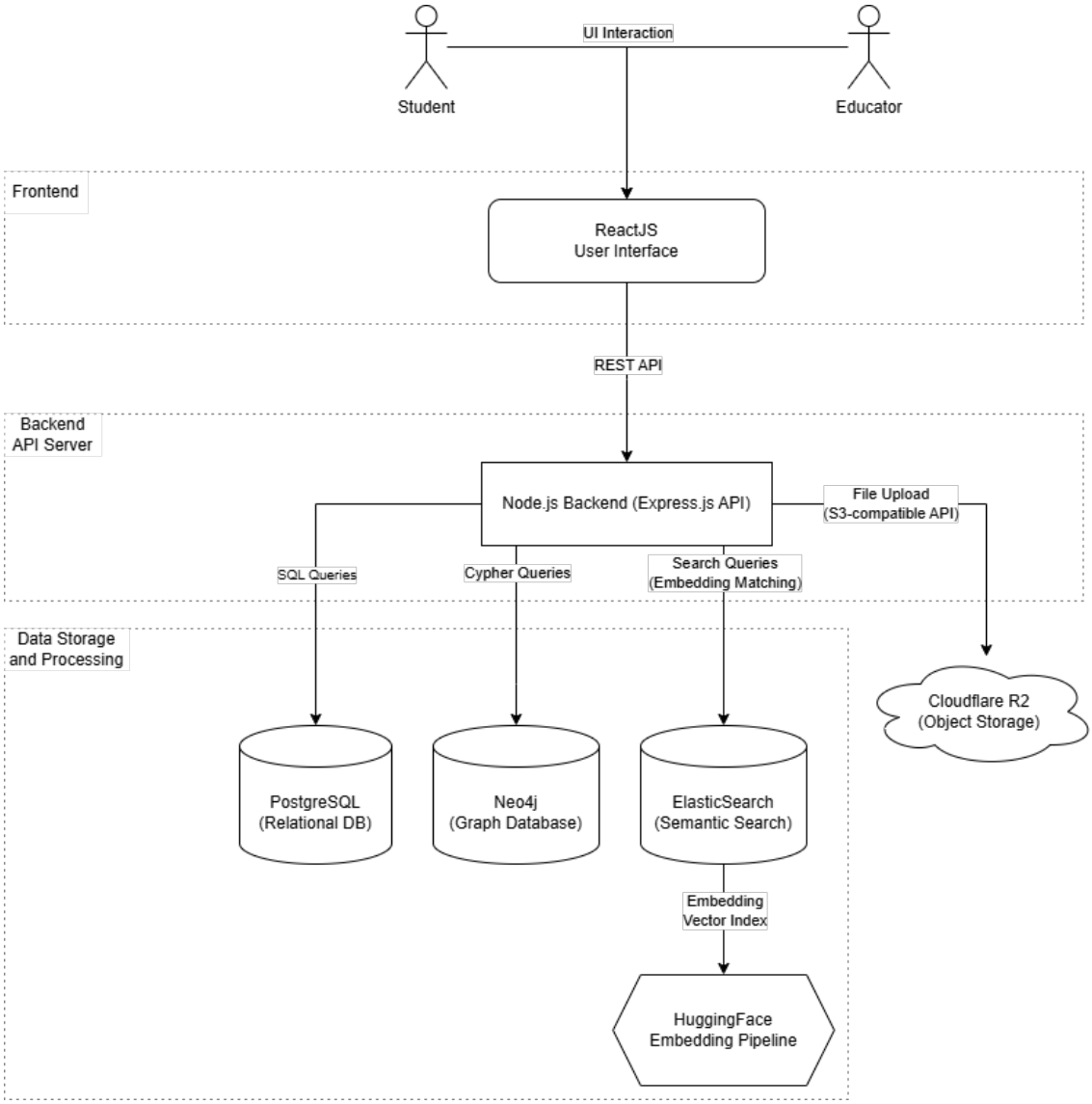

Figure 4.1: Component and data flow diagram.

Figure 4.1 illustrates the interaction between the system's components and their respective responsibilities across the frontend interface, backend server, data management layers, and supporting services such as cloud storage. Each component was selected based on its suitability for modularity, scalability, and development efficiency, with technology choices grounded in both practical implementation needs and evidence from related work.

The frontend, developed using *React*, serves as the primary interface for students and educators, enabling them to explore content, take assessments, and receive recommendations. *React* was chosen for its component-based architecture, responsiveness, and active ecosystem, which support rapid prototyping and reusable interface logic. User actions are routed to the backend, developed in *Node.js* with the *Express.js* framework. This stack was selected due to its non-blocking I/O model, lightweight nature, and compatibility with the frontend JavaScript environment. While alternatives such as Django offer robust capabilities, the familiarity with *Node.js* allowed for faster

development cycles and simplified integration across the stack. Authentication is handled via *Passport.js*, and the backend coordinates services such as progress tracking, semantic search, and personalized recommendations.

Data is managed across specialized components selected for their alignment with the system's functional requirements. *PostgreSQL* serves as the main relational database due to its ACID compliance, support for complex joins, and indexing capabilities, which are essential for tracking structured user progress, assessments, and content relationships. *Neo4j* functions as a graph database, modeling dynamic relationships between users, resources, and modules to support traversal-based recommendation queries. Its inclusion was motivated by literature on graph-based recommendation systems and prior academic experience with *Cypher*, which provided a more expressive and efficient querying mechanism than relational alternatives. For semantic information retrieval, *Elasticsearch* was chosen over ontology-based approaches like *OWL* and *SPARQL*, which proved too abstract and integration-heavy during early development. Combined with transformer-based embeddings, *Elasticsearch* enables hybrid queries that support both lexical relevance and contextual meaning, improving the retrieval of relevant learning materials.

External services enhance the platform's functionality while supporting modular deployment. *Cloudflare R2* was selected for file storage due to its S3 compatibility and generous free-tier access, which made it a cost-effective choice during prototyping without sacrificing performance. Resources such as documents and images are uploaded and accessed securely via signed URLs generated by the backend. To streamline development and maintain environment consistency, key services like *Elasticsearch* and *Neo4j* were containerized using *Docker*, enabling future integration of orchestration and deployment automation.

4.2.1 Deployment Overview

Building on the modular architecture, the deployment approach prioritized service independence and environmental consistency during development. Although a complete *Docker Compose* setup was not fully implemented, the initial containerization allowed for consistent local testing, isolated configurations, and simplified dependency management. This approach balanced the needs of lightweight prototyping with a scalable architecture, facilitating future migration to cloud environments with minimal structural changes.

Figure 4.2 presents the local development and deployment setup designed to support modularity, service independence, and environment consistency throughout development.

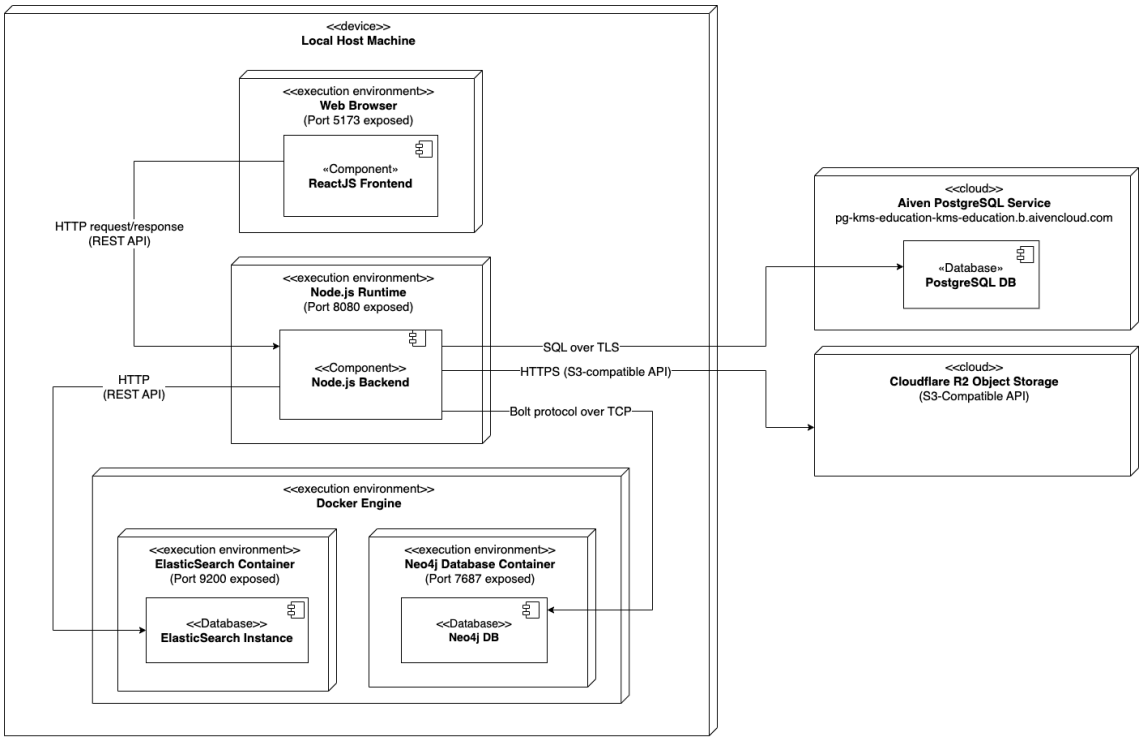

Figure 4.2: Local development and deployment setup diagram.

As illustrated in the previous diagram, the *React* frontend is served locally via Vite on port 5173, while the *Node.js* backend exposes a RESTful API on port 8080. The backend coordinates multiple services: a *PostgreSQL* database hosted on *Aiven Cloud* for structured data; Dockerized containers for *Elasticsearch* (port 9200) and *Neo4j* (port 7687); a local *Hugging Face* model (`all-MiniLM-L6-v2`) for semantic indexing; and *Cloudflare R2* for managing uploaded learning materials.

Although the system has not been deployed to a production environment, its modular and loosely coupled architecture ensures portability and scalability. The current configuration supports future integration with orchestration tools such as *Docker Compose*, positioning the system for cloud deployment with minimal adaptation.

4.2.2 Security and Access Control

Authentication is managed using *Passport.js*. Users are assigned roles (e.g., student, educator) that define access to features such as content creation, assessment, or progress tracking. File access from *Cloudflare R2* is secured using signed URLs, ensuring temporary and restricted access to uploaded resources.

Overall, this modular architecture ensures flexibility, maintainability, and scalability, supporting future enhancements such as collaborative features, advanced analytics, or additional intelligent services.

4.3 Data Model and Database Design

The platform's ability to support intelligent features such as personalized recommendations, semantic search, and modular learning paths relies heavily on a robust and well-structured data model. This section presents the design and rationale behind the three primary data representations used in the system: the relational model for structured data storage, the graph model for recommendation logic, and the search model for semantic information retrieval.

Each model serves a specific purpose:

- The relational model, implemented with *PostgreSQL*, supports core platform functionalities such as user management, resource organization, assessment tracking, and learning path progression.
- The graph model, implemented with *Neo4j*, enables advanced personalization by modeling rich, interconnected relationships between users, resources, and educational structures.
- The semantic search model, powered by *Elasticsearch* and transformer-based embeddings, allows users to retrieve learning materials using natural language queries and concept-based search.

The following subsections describe each model in detail, including schema structures, implementation decisions, and how they contribute to the overall system functionality.

4.3.1 Relational Model (*PostgreSQL*)

The platform's core structured data (e.g., users, resources, assessments, modules, and learning paths) is managed using a relational model implemented in *PostgreSQL*. This schema was designed to ensure data integrity, consistency, and scalability, while enabling key features like progress tracking, personalized content delivery, and learning assessment.

The entity-relationship diagram (ERD) showing the main entities and relationships in the system is presented in Appendix A.1.

The database includes the following tables:

- `users` - stores user account information, authentication data (hashed passwords), preferences (e.g., interests, education level), and role type (student, educator).
- `resources` - stores metadata about uploaded learning resources, including title, description, tags, categories, content type, and storage URL. Each resource can be associated with modules or tagged for recommendation.
- `modules` - represent standalone learning units. Each module can be part of one or more learning paths, has a title, summary, objectives, and estimated duration.
- `learning_paths` - structured sequences of modules. Each learning path can include metadata such as a title, summary, objectives, difficulty, and estimated duration.

- `learning_path_modules` - junction table linking modules to learning paths with a defined order.
- `learning_path_progress` - tracks each user's progress through a learning path, including completed modules, time spent, and current module status.
- `user_module_progress` - tracks user progress at the module level, including completion status, assessment status, and time spent.
- `assessments` and `assessment_results` - store self-assessable quizzes associated with modules, including questions, answers, solutions, and user performance data.
- `user_interactions` - logs user interactions with resources, modules, and learning paths. This data feeds into the recommendation engine.
- `bookmarks` — tracks resources users have saved for later.
- `module_resources` - maps resources to the modules they support and controls their display order.
- `user_searches`, `user_search_results`, `user_learning_content_search_results` - store semantic search interaction logs, used for evaluation and analysis.

Foreign key constraints were defined across all relationship tables to maintain referential integrity. For example, `assessment_results` enforces foreign keys linking to `users`, `modules`, and `assessments`, while `learning_path_modules` uses a composite primary key to manage ordered modules within a learning path.

The schema also incorporates constraints and checks for data quality, such as uniqueness on user-resource bookmarks and status enumerations for progress tracking (e.g., `not_started`, `in_progress`, `completed`). Indexes were defined implicitly by primary keys and selectively for high-frequency fields like `user_id`, `resource_id`, and `module_id`.

Overall, the relational schema was designed to balance data normalization and performance, while providing a robust structure for tracking educational engagement, personalization, and learning outcomes.

4.3.2 Graph Model (*Neo4j*)

To enable personalized recommendations based on user interactions and content relationships, the platform integrates a graph-based model using *Neo4j*. This graph structure allows for efficient traversal of interconnected entities such as users, resources, modules, and learning paths, capturing complex patterns that are difficult to express using purely relational data.

Data is periodically synchronized from *PostgreSQL* via a custom *Node.js* script, presented in Appendix B, which transforms key entities and interactions into graph nodes and relationships.

Node types in the graph model include:

- `User` – represents an individual interacting with the system. Stores profile data such as education level, field of study, topic interests, and content preferences.
- `Resource` – learning materials uploaded to the system, containing title, description, tags, type, and category.
- `Module` – standalone learning units that aggregate resources.
- `LearningPath` – structured sequences of modules.
- `Tag`, `Category`, `ResourceType` – represent classification attributes for resources.
- `Interaction` – logs an event between a user and a content item (e.g., viewed, started, bookmarked).

Relationships between nodes capture meaningful interactions and associations:

- `(:User)-[:PERFORMED]->(:Interaction)` – links a user to an action they initiated.
- `(:Interaction)-[:TARGET]->(:Resource / :Module / :LearningPath)` – identifies the target of the action.
- `(:User)-[:BOOKMARKED]->(:Resource)` – marks a saved resource.
- `(:Resource)-[:HAS_TAG]->(:Tag)`, `(:Resource)-[:HAS_CATEGORY]->(:Category)` – classify content.
- `(:Module)-[:HAS_RESOURCE]->(:Resource)` – links learning resources to modules.
- `(:LearningPath)-[:HAS_MODULE]->(:Module)` – links modules to learning path.
- `(:Resource)-[:OF_TYPE]->(:ResourceType)` – stores the content format (e.g., video, article).

Each `Interaction` node contains properties such as interaction type, timestamp, and a computed weight. There is a `calculateInteractionWeight()` function (line 215 of Appendix B) that assigns weights based on the type and recency of an interaction. This enables prioritizing recent and meaningful interactions (e.g., completing a learning path has a higher weight in comparison to viewing a learning path) in the recommendation logic.

Figure 4.3 presents an abstract schema of the graph model used in *Neo4j*.

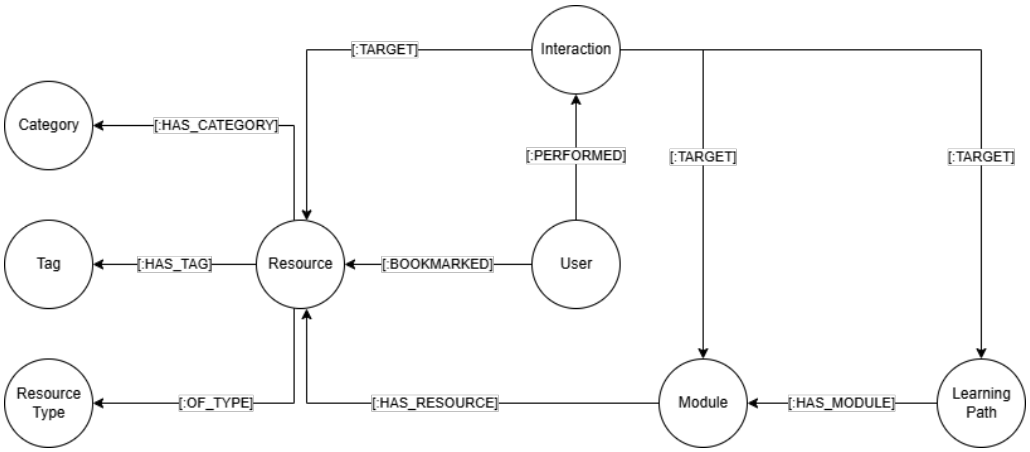

Figure 4.3: *Neo4j* database graph schema.

The graph structure allows the system to perform personalized content recommendations based on traversals of relationships such as shared tags, co-interacted resources, or similar learning paths. *Cypher* queries are used to retrieve relevant content for each user based on both direct and indirect connections, enabling a more contextualized and intelligent experience.

4.3.3 Search Model (*Elasticsearch*)

For semantic information retrieval, the platform leverages *Elasticsearch* in combination with transformer-based language models. This hybrid search approach combines lexical matching with dense semantic embeddings to support natural language queries and context-aware search over learning content.

Dedicated indexes were created for resources, modules, and learning paths, incorporating both textual metadata and vector representations to enable flexible and meaningful search results.

Three indexes were defined for `resources`, `modules`, and `learning paths`. However, `modules` and `learning paths` are unified under a single `learning_content` index, as they are retrieved and displayed together within the same search interface. This consolidation simplifies query processing and allows joint semantic ranking across different types of learning content.

Table 4.2 summarizes the structure, field types, and semantic indexing used in both the `resources` and `learning_content` indexes.

Table 4.2: Field structure and types used in the `resources` and `learning_content` indexes.

Field	Type	Description
resources		
`title`	text	Title of the resource.
`description`	text	Full-text description of the resource.
`type`	keyword	Type of the resource (e.g., video, pdf).
`tags`	keyword	Tags used for filtering and classification.
`category`	keyword	Subject category of the resource.
`format`	keyword	Resource format or delivery method.
`embedding`	dense_vector	Embedding vector for semantic similarity search.
learning_content (modules)		
`title`	text	Title of the module.
`summary`	text	Summary or abstract of the module.
`objectives`	text	Learning objectives and outcomes.
`embedding`	dense_vector	Semantic vector for recommendation and search.
learning_content (learning paths)		
`title`	text	Title of the learning path.
`summary`	text	Overview of the learning path.
`objectives`	text	Learning objectives and outcomes.
`difficulty_level`	keyword	Level of difficulty (e.g., beginner, intermediate).
`embedding`	dense_vector	Embedding vector for semantic similarity.

The indexed fields can be grouped into three main categories:

- **Textual fields:** `title`, `description`, `summary`, and `objectives` are indexed as *text* types. These fields are analyzed using *Elasticsearch*'s standard analyzer and tokenized for full-text search using the BM25 [19] ranking algorithm.
- **Categorical fields:** Fields such as `type`, `tags`, `category`, `format`, and `difficulty_level` are defined as *keyword* types. These fields are not tokenized and are used primarily for filtering, faceting, and aggregation.
- **Embedding field:** The `embedding` property is defined as a *dense_vector* of 384 dimensions. These vectors are generated using the `all-MiniLM-L6-v2` transformer model [66] and are indexed with cosine similarity enabled to support semantic search functionality.

To represent content semantically, the vector embeddings were generated using a transformer-based language model from *Hugging Face*. The implementation uses the *feature-extraction* pipeline with the `Xenova/all-MiniLM-L6-v2` model [66], a lightweight and efficient sentence encoder based on the MiniLM architecture. This model converts input text into fixed-size dense vectors that capture semantic meaning.

The embedding generation process consists of the following steps:

1. **Preprocessing:** HTML tags are stripped from content to ensure clean input.
2. **Text fusion:** Fields such as title, summary, tags, and objectives are concatenated into a single input string.
3. **Embedding inference:** The transformer model generates an embedding vector for the input.
4. **Post-processing:** The embedding is trimmed or zero-padded to ensure a consistent dimensionality of 384.

The resulting embeddings are stored in *Elasticsearch* and indexed with cosine similarity enabled. This allows the system to measure how similar two pieces of text are in semantic space, rather than relying on exact word overlap.

When a user submits a search query, the system generates an embedding vector for the input using the same transformer model used during indexing. This vector is then used in a hybrid query that combines semantic and lexical matching. The *Elasticsearch* query uses a `bool.should` clause that evaluates:

- A `script_score` component using cosine similarity to rank documents based on vector closeness to the query.
- A `multi_match` query on fields such as `title`, `description`, `tags`, and `category`, with term frequency-based ranking via BM25.

The fields are weighted to prioritize more informative text (e.g., `title` is given higher weight than `category`), and fuzzy matching is enabled to allow tolerance for minor typos or variations. This setup ensures that users receive results that are both semantically and textually relevant, even in the presence of ambiguous or imprecise queries.

To improve relevance and recall, the platform adopts a hybrid search strategy that combines:

- **BM25 keyword search:** *Elasticsearch*'s default scoring algorithm, which ranks results based on term frequency and inverse document frequency [19]. This is effective for direct text matches.
- **Vector similarity search:** Based on cosine similarity between query embeddings and stored embeddings, this method enables semantically relevant results even when there is no exact term match.

By combining both approaches, the system improves its ability to retrieve results that are both contextually relevant and lexically appropriate, particularly useful for students who may not always use precise keywords in their queries.

Additionally, the semantic search pipeline is tightly integrated with the backend logic. When a new resource, module, or learning path is added to the platform, it is indexed in *Elasticsearch*

immediately. This includes generating an embedding and associating it with the respective content item. This process ensures that the search system remains synchronized with the primary *PostgreSQL* database.

Overall, the use of semantic search significantly enhances the retrieval and discovery of educational content. It allows students to find resources that align with their learning intent, even when their queries are vague, incomplete, or phrased differently than the content stored, thereby contributing to the intelligent behavior of the knowledge management system.

4.4 Core Features

This section presents the platform's main functionalities from the user's perspective. Each feature is described in terms of its purpose, interaction flow, and integration with the system's architecture.

The design of these features was directly informed by the functional requirements collected through user stories, presented in Table 4.1. These user stories captured the needs and expectations of both students and educators and served as a foundation for translating user needs into practical system components. By linking each user story to the corresponding implemented feature, the table provides clear traceability between the requirements gathering phase and the system's design.

The user interface design was adapted and extended from an open-source *React* template, the *Material Tailwind Dashboard*[3] by Creative Tim, which provided a foundation for the overall layout, component styling, and navigation structure.

The following subsections present each core feature in detail, showing how these implementations respond to the mapped user needs.

4.4.1 User Registration

This feature directly addresses user stories US1, US2, and US3, which express the need for secure account creation, login, and profile customization to enable personalized learning experiences.

The platform uses a two-step registration process designed to personalize the learning experience from the beginning. As shown in Figure 4.4, the first step captures basic user details and allows the selection of a role (student or educator).

[3] https://www.creative-tim.com/product/material-tailwind-dashboard-react?ref=readme-mtdr

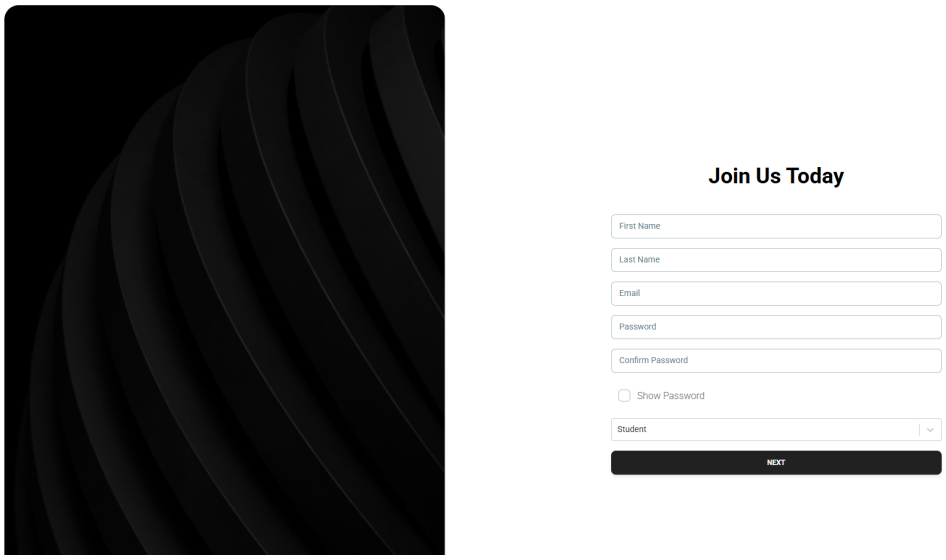

Figure 4.4: Sign-Up Process – Step 1: Personal details and role selection.

This step is essential for establishing a user profile and assigning the appropriate interface logic. While the platform maintains a unified experience, some features are unique based on the role to better suit the user's context. For example, educators may access content creation options, while students engage more with learning paths and assessments.

Figure 4.5 presents the second (optional) step of the registration process, where users are invited to provide information about their education level, field of study, topic interests, preferred content types, and language preference.

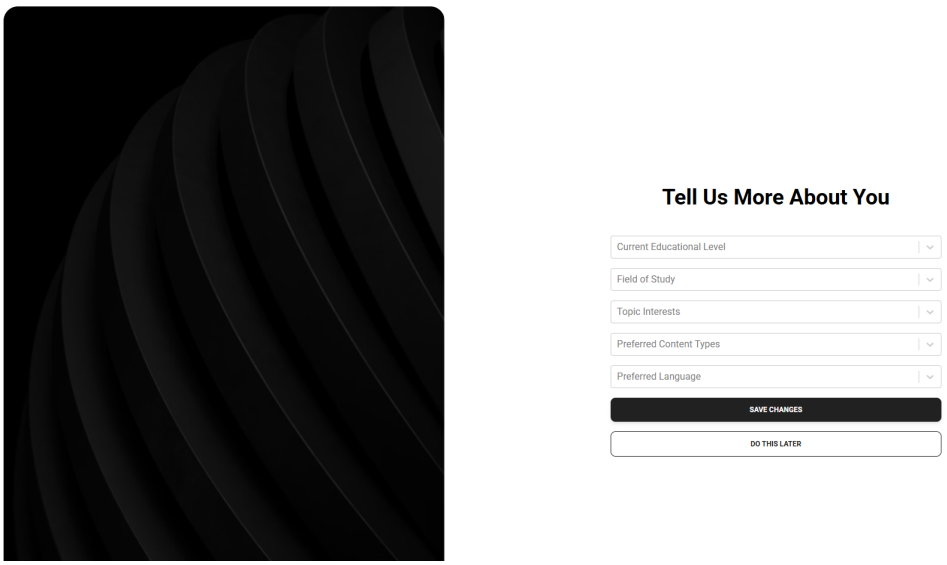

Figure 4.5: Sign-Up Process – Step 2: Profile preferences.

This data feeds into the recommendation engine, allowing the system to deliver more relevant suggestions from the start. However, users who prefer to explore before specifying preferences

can skip this step. In that case, fallback strategies such as popularity-based recommendations are applied until the system collects enough interaction data to personalize suggestions. This approach helps balance personalization with usability, minimizing onboarding friction while still enabling adaptive learning over time.

The authentication mechanism is implemented using the *Passport.js* middleware, which manages sessions securely and supports credential-based login.

Upon registration, the backend performs the following actions:

- Validates user input and hashes the password using the `bcrypt.js` library.
- Stores the user in the *PostgreSQL* `users` table with all relevant profile and preference fields.
- Initializes a session using *Passport*'s `req.logIn` method to authenticate the user immediately.

The sign-in flow follows a similar structure, verifying credentials and creating a secure session upon successful login. This ensures that users can access personalized features immediately after registering.

4.4.2 Resource Upload and Management

This functionality fulfills the needs outlined in user stories US16 and US19, by allowing educators to upload and manage their own learning resources through an intuitive interface.

The platform includes a dedicated resource upload feature that is only available to educator accounts. By restricting this functionality, the system ensures that submitted learning materials satisfy a standard of academic quality and pedagogical relevance. Educators play a key role in expanding the platform's repository by contributing structured, diverse, and pedagogically valuable resources.

As shown in Figure 4.6, the upload interface is designed for clarity and ease of use, supporting both metadata entry and file attachment.

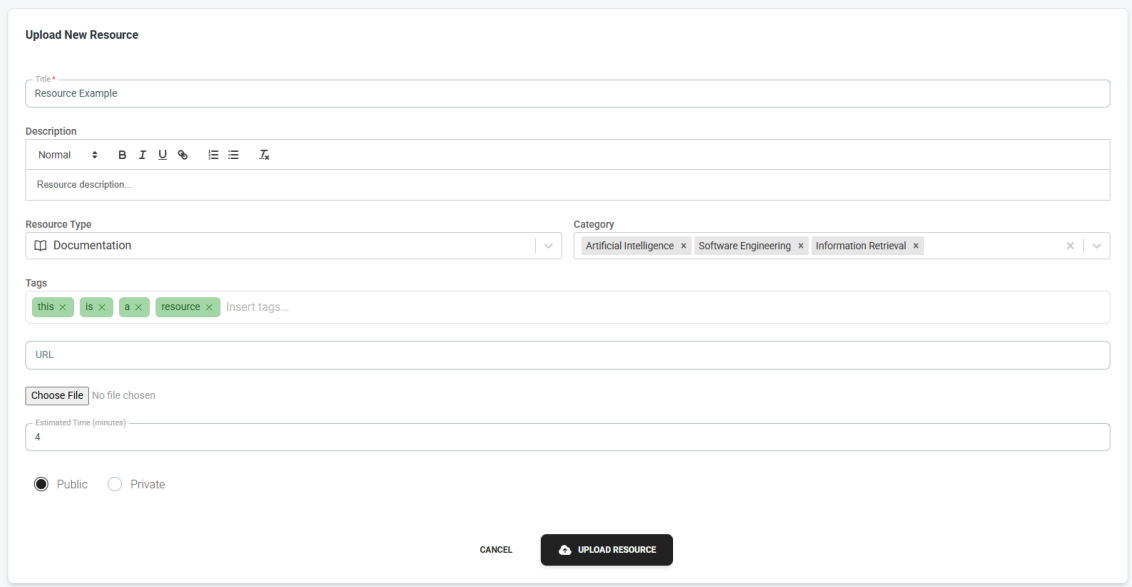

Figure 4.6: Resource upload interface (educator view).

To make the submission process easier, the form is separated into user-friendly sections:

- **Title and Description:** A rich-text editor (*Quill*) allows educators to provide a formatted description of the resource, enhancing readability and structure.
- **Categorization:** Users select the resource type (e.g., Article, Video, Document) and assign one or more subject categories (e.g., Artificial Intelligence, Mathematics) for classification.
- **Tagging:** Tags can be freely added to improve searchability and semantic indexing.
- **Estimated Time:** Indicates the expected duration to consume the resource, supporting better time management for learners.
- **Format and Visibility:** Files in various formats (PDF, DOCX, PPTX, PNG, JPG, etc.) can be uploaded. Educators may choose to make resources public or restrict them to private use.

This feature supports pedagogical consistency by encouraging educators to provide metadata that aligns with instructional design principles, such as setting clear learning expectations and categorizing materials by topic and duration. The resource submission workflow promotes intentional content curation, ensuring that materials are discoverable and suited to different learners' goals, pacing, and learning preferences.

Upon submission, the platform processes the input data by storing the resource metadata in the *PostgreSQL* database, optionally uploading the file to *Cloudflare R2*, and generating semantic embeddings for indexing in *Elasticsearch*. This ensures that each uploaded resource is immediately searchable and retrievable through both keyword and semantic queries, reinforcing the system's intelligent capabilities and content consistency across services.

An example of a submitted resource as seen by users is shown in Figure 4.7, which displays an embedded YouTube video, resource metadata, and two recommendation sections for both resources and modules.

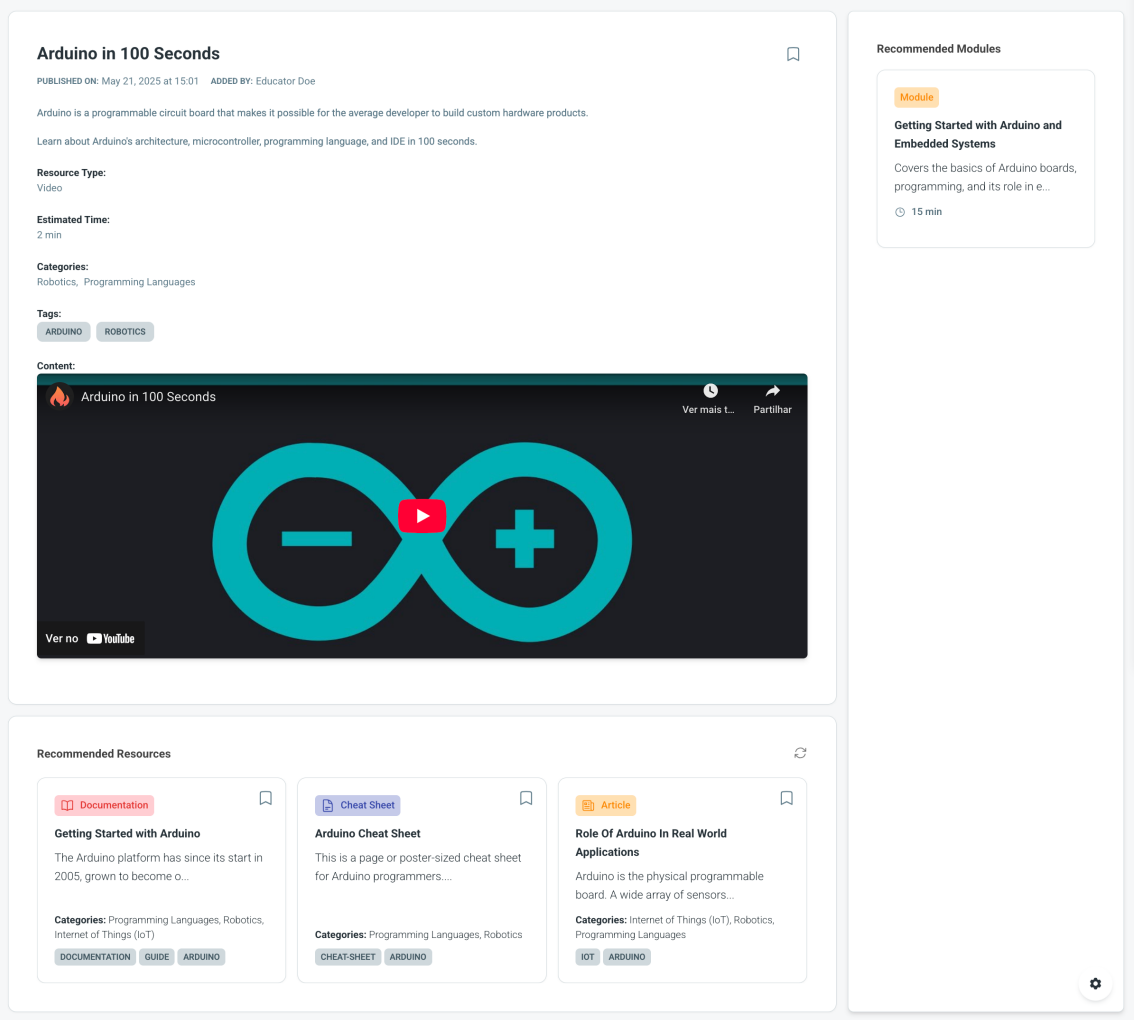

Figure 4.7: Resource details page.

4.4.3 Modules and Learning Path Progress

The design of modules and learning paths aligns with several user stories (US8, US9, US15, US17, US18, and US20), which highlight student expectations for modular learning, progress tracking, and educator-driven content creation and assessment.

To structure the learning experience and support personalized progression, the platform organizes resources and learning content into modules and learning paths. This design reflects common educational strategies such as scaffolded learning and mastery-based progression, ensuring that learners advance in a coherent and purposeful way. Tracking progress at both the module and path level reinforces learner engagement and provides clear feedback on achievements, which are both important aspects of self-regulated learning.

Modules

The implementation of modules addresses user stories US8, US17, and US20, which highlight the need for structured, standalone content delivery and educator-driven module creation with embedded assessments.

Modules serve as structured learning units that integrate educational resources with a formative assessment. This combination aligns with blended learning principles and supports active learning by encouraging students to engage with content and apply knowledge through practice.

The module creation interface is designed as a guided three-step process for educators.

In the first step, shown in Figure 4.8, educators input a title, summary, and objectives and select relevant categories. By prompting educators to define objectives, the system encourages instructional alignment and outcome-driven design.

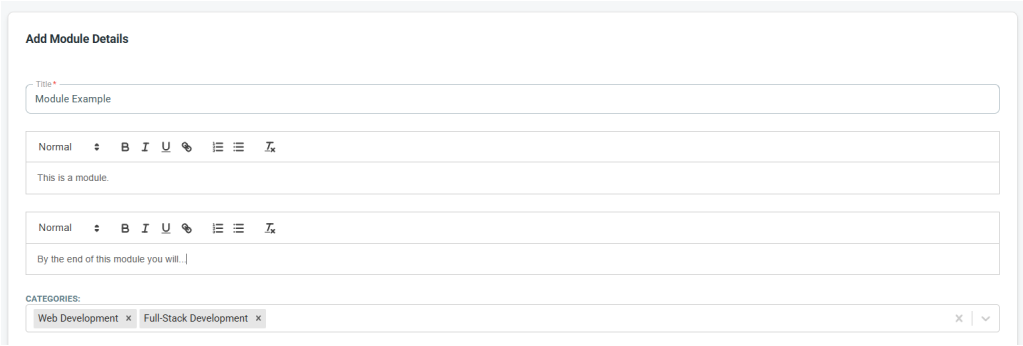

Figure 4.8: Module Creation Process—Step 1: Module metadata form.

Next, educators manually select resources or choose from system-generated suggestions. This approach supports both educator autonomy and assisted planning. Drag-and-drop ordering provides flexibility in structuring the learning flow, as illustrated in Figure 4.9.

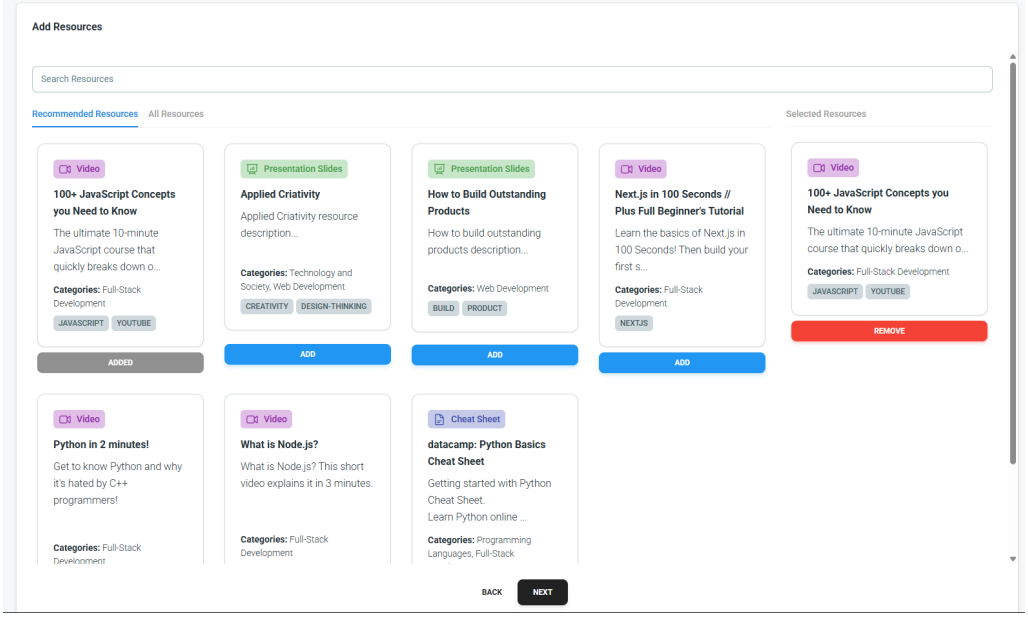

Figure 4.9: Module Creation Process—Step 2: Resource selection.

Finally, educators create a short multiple-choice quiz to assess comprehension. A minimum of three questions is required. Questions can be added manually or imported/exported in standard formats (e.g., JSON or XML) for reusability and efficiency (Figure 4.10).

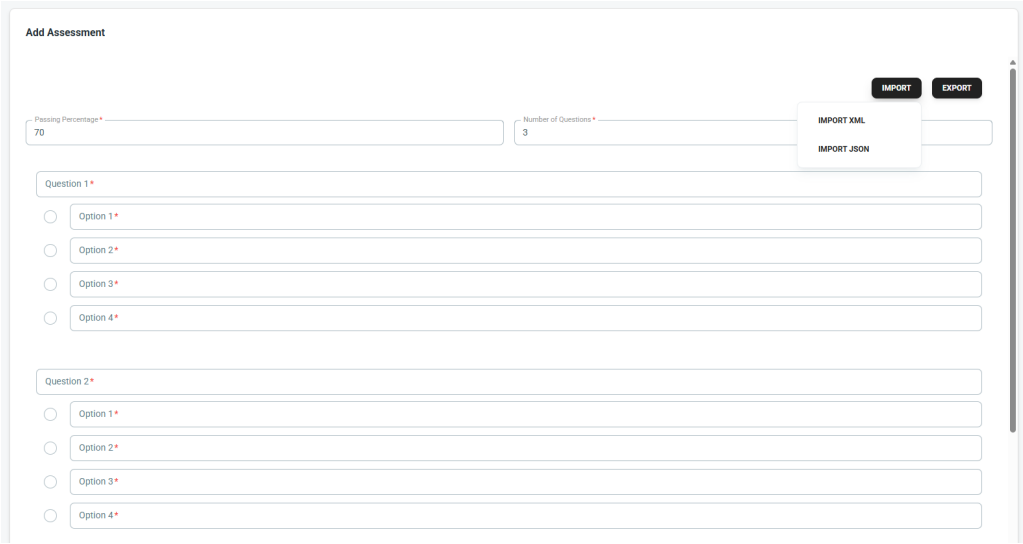

Figure 4.10: Module Creation Process—Step 3: Assessment creation.

An example of a completed module from the student perspective is shown in Figure 4.11.

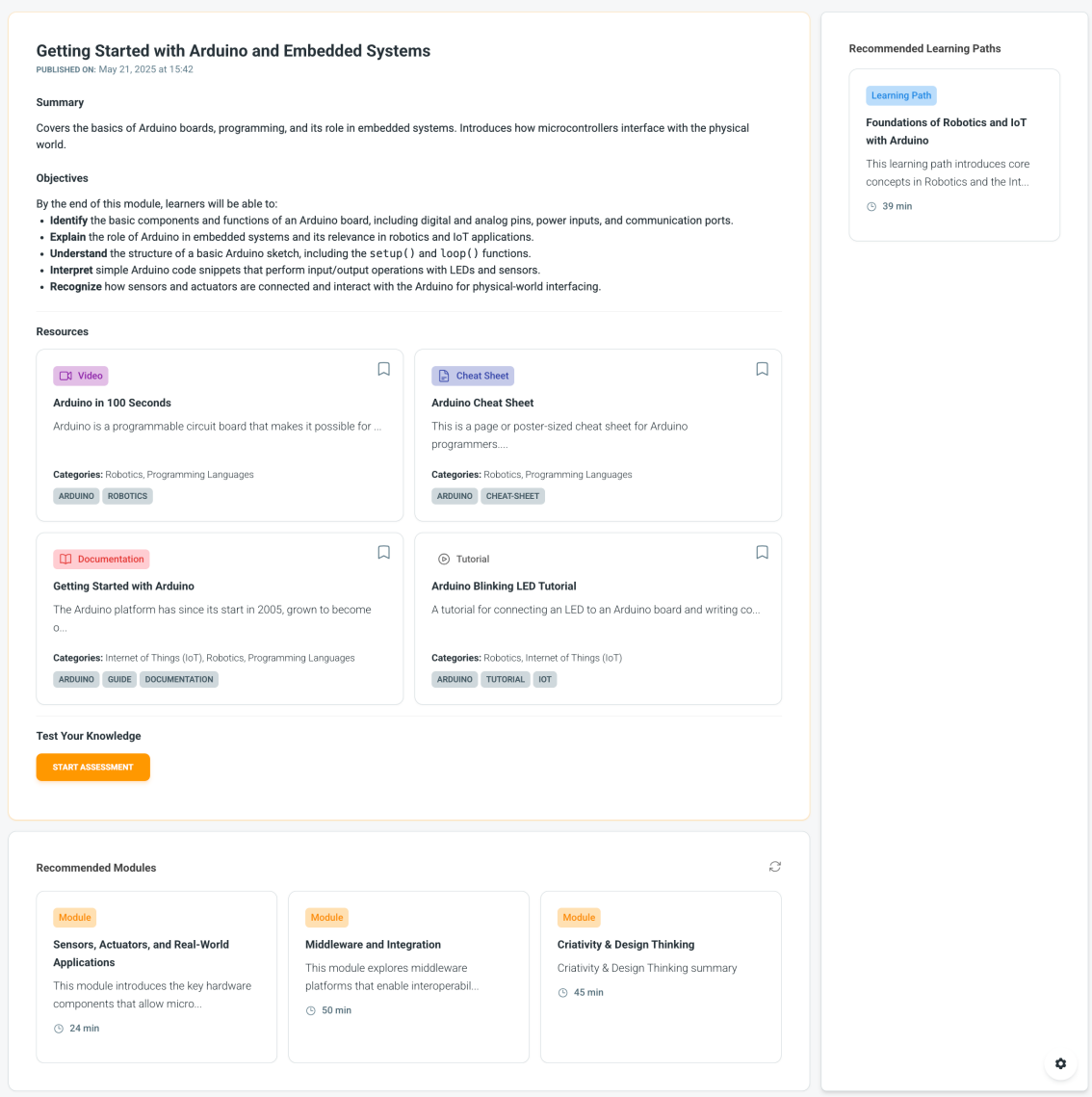

Figure 4.11: Module details page.

When viewing a module, students can access all related resources and are encouraged to complete the corresponding assessment to test their understanding. Personalized module and learning path recommendations are also shown to promote continued and interest-driven learning.

Learning Paths

The design of learning paths responds to user stories US9 and US18, which emphasize the importance of progress tracking and the ability for educators to build sequenced, goal-oriented learning journeys.

Learning paths are designed to offer a sequenced and goal-oriented learning experience. By chaining together multiple modules, educators can construct structured journeys that foster skill

acquisition over time. This approach aligns with instructional scaffolding and supports learning progression by building upon previously acquired knowledge.

The sequential module unlocking mechanism reinforces the pedagogical principle of mastery learning: learners can only advance after demonstrating sufficient understanding. This guides students through a logical flow of topics and helps prevent gaps in foundational knowledge.

Learning path creation is done through a two-step form.

In the first step, shown in Figure 4.12, educators enter key details including the title, summary, learning objectives, difficulty level, categories, and visibility settings. This encourages educators to define the intended outcomes of the path clearly.

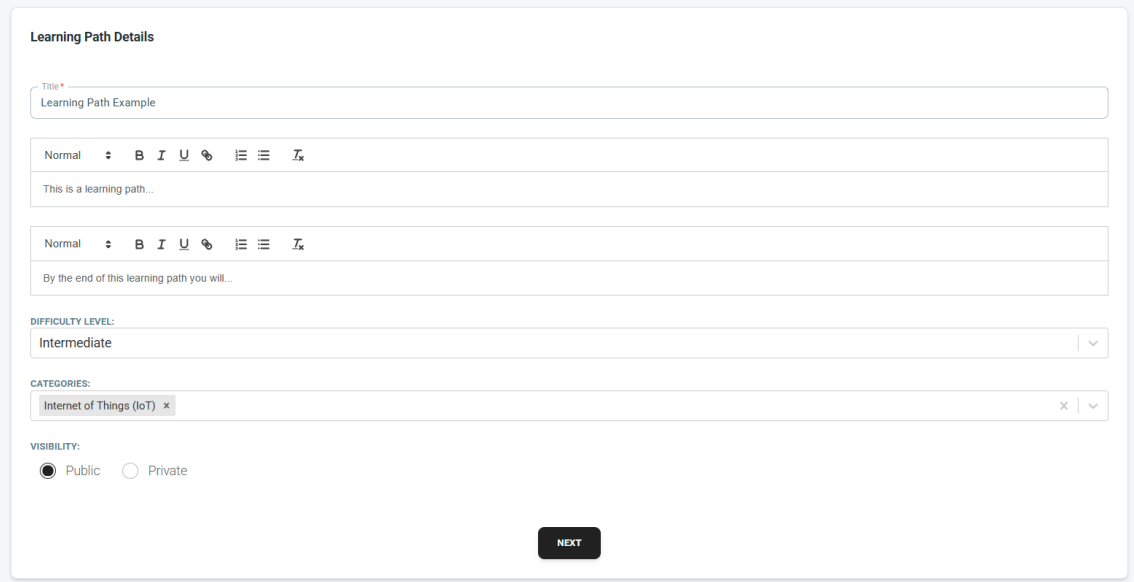

Figure 4.12: Learning Path Creation Process—Step 1: Learning path metadata form.

Next, educators select which modules to include in the path. This can be done manually or based on system recommendations, as seen in Figure 4.13.

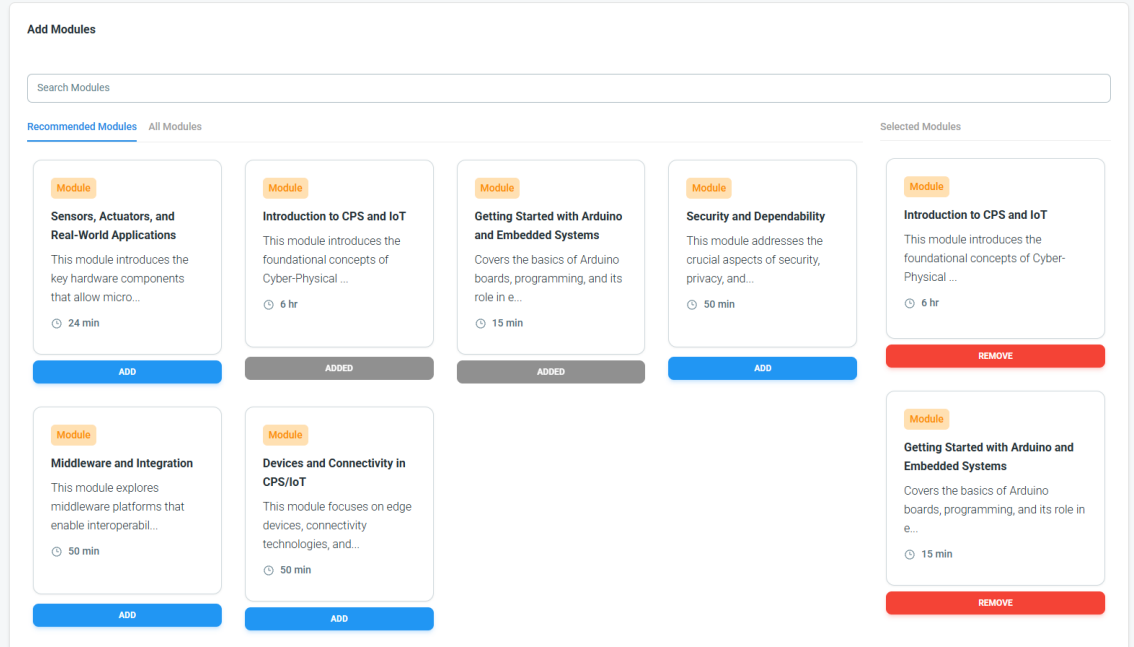

Figure 4.13: Learning Path Creation Process—Step 2: Module selection.

When students view a learning path, such as the one shown in Figure 4.14, they are presented with a clear visual representation of their current progress. The interface is designed to promote focus, motivation, and continuity which are key factors in effective self-paced learning.

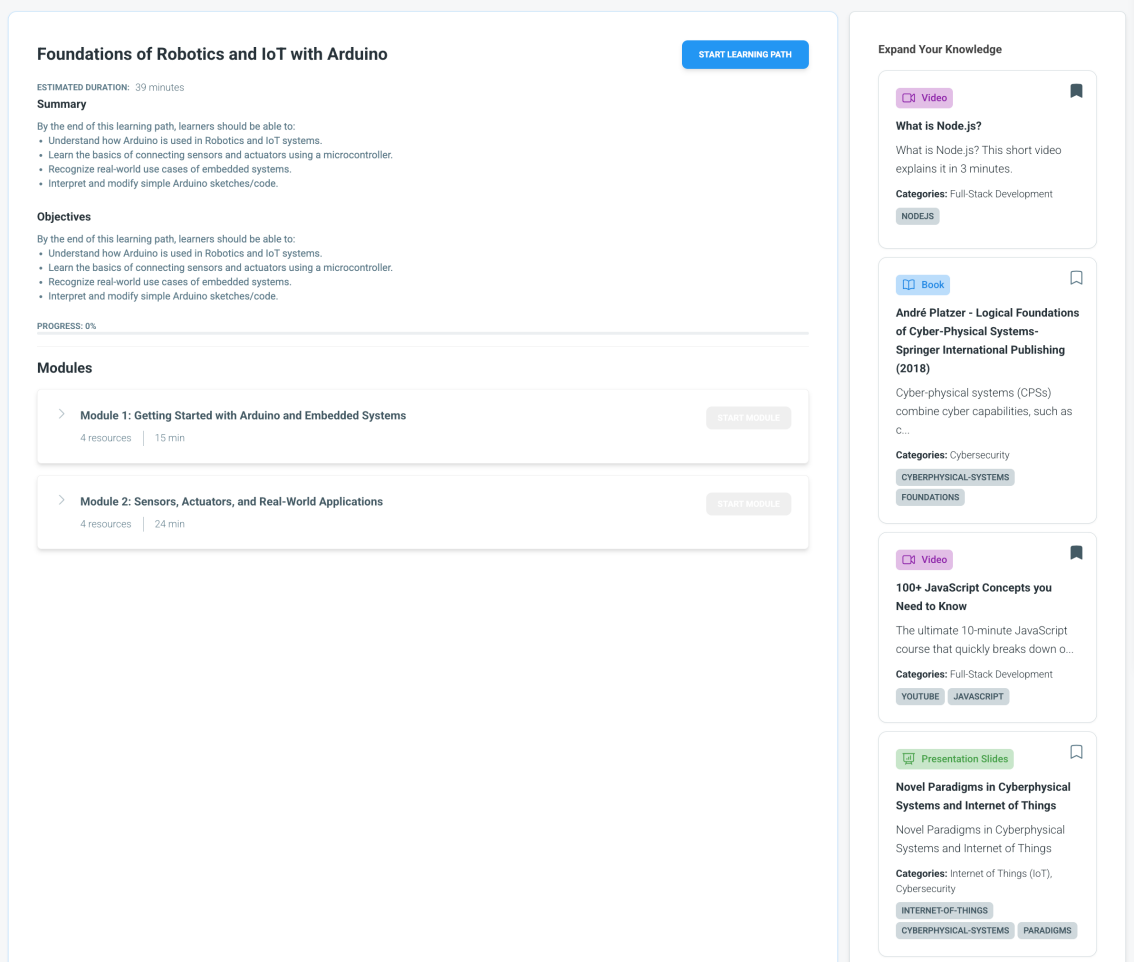

Figure 4.14: Learning path details page.

Modules are displayed sequentially, with access controlled by assessment completion. This ensures structured progression and reinforces competency before moving forward.

All interactions, such as starting a module, passing an assessment, or completing a learning path, trigger backend updates that are reflected in both the `user_module_progress` and `learning_path_progress` tables. This enables:

- Accurate real-time progress tracking.
- Unlocking of the next module, based on logic managed by both the frontend and backend.
- Display of visual cues for progress and performance (e.g., green “Completed” button states).

Once all modules are completed and assessments passed, the learning path is marked as complete.

Study Paths

This feature directly implements user story US15, which expresses the need for students to create and personalize their own study paths. Study paths enable students to create their own learning sequences, promoting self-directed and interest-driven learning. Although learning paths are created by educators, study paths give students autonomy to assemble content according to their personal goals, learning styles, or revision needs.

This feature supports learner agency, a key principle in personalized education. By empowering students to take ownership of their learning journey, the platform fosters motivation, reflection, and lifelong learning habits. The ability to arrange modules around individual preferences mirrors flexible learning models often used in higher education.

Study paths are private by default and support:

- Custom grouping of modules by topic or interest.
- Individual progress tracking.
- Revisiting and organizing content for review or deeper exploration.

Their flexibility also lays the foundation for future adaptive learning features, where personalized paths could dynamically evolve based on student performance and preferences.

4.4.4 Assessments and Learning Evaluation

As described in user story US11, students expressed the need to evaluate their progress. This feature provides a formative mechanism to reinforce learning and unlock sequential content. The assessment component is a core mechanism in the system for validating students' understanding and enabling structured module progression. Each module includes a multiple-choice quiz defined by the educator, which must be successfully completed for the module to be marked as complete.

Assessments serve two primary pedagogical purposes:

- **Formative evaluation:** Enabling students to test their understanding and receive immediate feedback, supporting self-assessment and reflection.
- **Progression control:** Ensuring learners achieve a minimum level of mastery before advancing to more complex content, reinforcing structured learning progression.

This dual function aligns with widely accepted educational frameworks such as mastery learning and formative assessment theory. By embedding quizzes at the end of each module, the platform fosters active recall and metacognitive engagement which are both critical for long-term knowledge retention.

Each assessment consists of:

- A set of educator-defined multiple-choice questions.
- A solution array specifying correct answers.

- A configurable passing threshold (default 70%) that determines whether the learner has achieved sufficient understanding.

Figure 4.15 shows the assessment interface where students interact with quiz content.

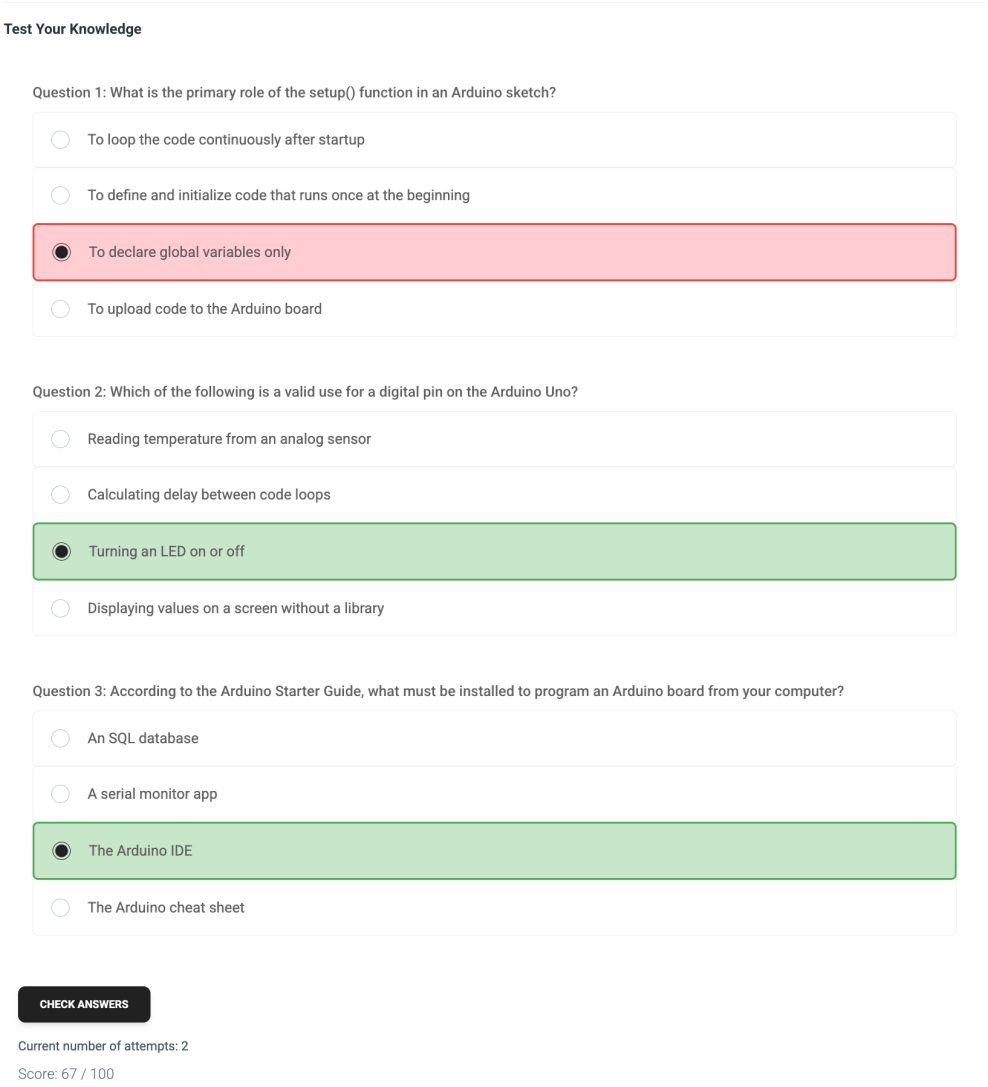

Figure 4.15: Module assessment interface.

The interface is designed for clarity and cognitive ease, using radio buttons for answer selection and providing immediate visual feedback upon submission. This setup supports fast feedback loops that help learners identify misunderstandings and adjust their thinking in real time.

Students are allowed multiple attempts, promoting a growth mindset and reinforcing the idea that learning is an iterative process. While past results are not yet stored or visualized, this is planned as a future enhancement to support deeper reflection, progress monitoring, and self-regulation.

The assessment logic operates in two distinct modes:

- **Independently**, for standalone modules, enabling flexible and modular content usage.
- **Sequentially**, within learning paths, where passing an assessment unlocks the next module to maintain a coherent learning flow.

By requiring students to actively engage with quizzes to complete modules, the platform reinforces principles of active learning, discourages passive browsing, and gives educators indirect feedback on content effectiveness and learner comprehension. This strengthens the learning experience while promoting accountability and goal-oriented progression.

4.4.5 User Dashboard

The dashboard supports key elements from user stories US5 and US13, providing access to bookmarks and tracking progress in a centralized, personalized view. The user dashboard page (see Appendix C) serves as the central hub of the platform, offering students a personalized overview of their learning journey. It was designed to promote autonomy and motivation by making progress, achievements, and next steps clearly visible and easy to navigate.

From a pedagogical standpoint, the dashboard supports key principles of self-regulated learning, helping users monitor their activities, set goals, and reflect on progress. By consolidating critical information such as ongoing and completed content, personalized recommendations, and study paths, it reduces cognitive load and fosters purposeful engagement.

The dashboard consists of several distinct sections, each supporting a different phase of the learning process:

- **Recently Viewed Resources:** Enables quick return to previously accessed content, reinforcing learning through repetition and supporting continuity.
- **Recommended Resources:** Displays personalized suggestions based on user preferences and interactions. This supports adaptive learning by aligning content with learner interests and context.
- **My Study Paths:** Lists custom paths created by the user, reinforcing ownership, planning, and self-direction in the learning process.
- **In Progress:** Highlights active learning paths and modules with visual progress indicators. This provides immediate feedback and sustains momentum by acknowledging ongoing effort.
- **Completed:** Presents a record of completed modules and paths, supporting motivation through visible achievement and progress tracking.
- **Bookmarked Resources:** Offers a space for strategic content saving, supporting time management and goal-oriented study planning.

These sections are displayed in a horizontally scrollable layout to present a variety of elements without visual clutter. This design follows usability principles suited to educational platforms: maintaining clear visual hierarchy, minimizing distractions, and offering intuitive access to task-relevant content.

By combining real-time progress data and context-aware recommendations, the dashboard becomes more than a navigation tool, it serves as a learning companion that encourages reflection, agency, and intentional exploration.

4.4.6 Recommendation System

In response to user stories US6 and US10, the system provides personalized recommendations based on profile data and usage history. Building upon the graph model detailed in Section 4.3, this section describes how the recommendation system leverages those relationships to deliver personalized content suggestions across the platform.

The recommendation system is a core pedagogical component designed to enhance the user experience by offering personalized learning content. By analyzing user interests, past interactions, and profile data, it helps learners discover relevant resources, modules, and learning paths aligned with their individual goals and preferences.

From an educational perspective, personalized recommendations reduce the cognitive burden of content discovery, increase relevance, and promote learner engagement. By aligning content with a learner's needs and context, the system supports differentiated instruction, which is an essential principle in modern educational technology. Furthermore, it encourages exploratory learning by surfacing content that extends or complements what the learner is currently engaging with.

Recommendation Queries and Their Locations

The recommendation system operates through six distinct queries, strategically located across the platform to provide meaningful suggestions at the right moment in the learning process. These queries ensure that users receive meaningful suggestions wherever they are, enhancing their experience by providing relevant learning materials based on their activity and preferences.

For example, on the user dashboard (Appendix C), users are immediately greeted with personalized content upon logging in. At first, recommendations are generated based on the user's specified preferences and topic interests. If no preferences are provided during sign-up or profile setup, the system defaults to suggesting the most popular resources across the platform. As the user begins interacting with content, the recommendation engine dynamically incorporates both their stated preferences and behavioral data to deliver increasingly personalized suggestions. This approach ensures that users quickly re-engage with the platform and discover new content that aligns with their ongoing learning journey.

When users view a resource details page (Figure 4.7), the system analyzes the specific resource they are engaging with and recommends additional resources or related modules. By assessing

content similarity and category matching, this query suggests complementary materials, expanding the user's understanding of the topic.

Similarly, when viewing a module details page (Figure 4.11), the system recommends related modules and learning paths that align with the current module being explored. The query used in this location considers both the user's learning history and the content of the current module to suggest additional modules or paths that will help continue the learning journey. By recommending content that complements the current module, the system ensures that users are exposed to a logical progression in their studies.

Within the learning path details page (Figure 4.14), the system suggests resources that align with the user's learning path, allowing users to deepen their knowledge in the subjects they are currently studying. The recommendation query used here is designed to expand the user's learning journey by offering resources that directly support the completion of their current path, ensuring a continuous and enriched learning experience.

Throughout the platform, the system tracks users' interactions with resources, modules, and learning paths, such as which items they have viewed, bookmarked, or completed. The interaction-based query uses this data to recommend new content based on similarities to previously engaged materials. This query is crucial across all sections of the platform, as it provides personalized recommendations that evolve with the user's journey.

Each of these placements is designed to serve a pedagogical function: bridging gaps, reinforcing knowledge, or extending learning. The system supports both discovery-driven and goal-oriented learners by responding to diverse learning styles.

Recommendation Logic and Scoring Model

The recommendation engine applies a consistent graph-based logic across all queries, regardless of whether it is suggesting resources, modules, or learning paths.

Each query evaluates candidate items based on a combination of semantic similarity, structural proximity within the graph, and user-specific interaction patterns. While the specific *Cypher* queries differ depending on the recommendation context, they generally share the following logic components:

- **Content-Based Similarity:** Compares shared categories and tags and uses string similarity metrics (e.g., Sørensen–Dice) to measure how closely titles and descriptions match the target content. These comparisons are implemented using the *APOC* plugin in *Neo4j*, which provides advanced string functions such as `apoc.text.sorensenDiceSimilarity`. This supports both topical alignment and semantic association within the graph structure.

- **Interaction-Based Weighting:** Enhances recommendations by factoring in user-specific interactions (e.g., viewed, bookmarked, completed). More meaningful actions are weighted more heavily, and recent activity is prioritized using a recency function.

- **Graph Structure Awareness:** Uses relationships between nodes, such as `HAS_CATEGORY`, `HAS_TAG`, and `PERFORMED`/`TARGET`, to inform recommendation paths. This supports both semantic association and collaborative relevance based on user behavior.
- **Score Calculation:** Each candidate receives a total score computed as a weighted sum of the above factors. Items are then ordered by this score to produce the final ranked recommendations.

This logic is applied uniformly but tailored per context. For instance:

- On a **resource details page**, the system recommends related resources or modules by comparing metadata and usage patterns.
- On a **module details page**, it suggests modules or learning paths that share topics or are frequently explored by similar users.
- On a **learning path details page**, it identifies supporting resources relevant to the learning objectives of that path.

A domain-specific formula is applied to interaction nodes to reflect the pedagogical significance of user actions, as shown in Equation 4.1:

$$\text{Weight} = \text{BaseWeight} \times \max(1, 10 - \text{DaysSinceInteraction}) \tag{4.1}$$

This formula is applied during the graph synchronization process and directly affects recommendation ranking, ensuring that the most recent and meaningful behaviors influence recommendations. More details on the weighting logic and interaction synchronization can be seen in Appendix B.

The recommendation system integrates semantic, structural, and behavioral dimensions into a flexible and adaptive scoring model. This enables context-aware recommendations that reflect both the learner's profile and their evolving engagement with the platform, reinforcing personalization, learner progression, and exploratory learning.

Fallback Strategies

To ensure continuity of support, especially during cold-start scenarios, the system incorporates multiple fallback strategies that maintain recommendation quality when interaction data is sparse or unavailable. These strategies are essential for maintaining engagement and providing valuable content to users who may have limited interaction history or are new to the platform. In cases where personalized recommendations cannot be generated, fallback queries are used to deliver recommendations based on aggregate data, such as the popularity of resources or a random selection. This ensures that users are always presented with content to explore, regardless of their interaction history or profile completeness.

To ensure continuity of support for all users, the system incorporates multiple fallback strategies. These maintain a personalized experience even when user-specific data is unavailable, which is a common challenge in educational platforms during the cold start phase.

- **Profile-Based Fallback:** When the system is unable to generate recommendations based on user interactions, it defaults to recommending content based on the user's profile data, including topic interests and preferred content types. This guarantees that users still receive content aligned with their general preferences.
- **Popularity-Based Fallback:** For users with limited interaction data or when no profile-based recommendations can be generated, the system offers content that is popular across the platform. These recommendations are drawn from the most frequently accessed or highly rated resources, ensuring that users are exposed to well-regarded and commonly used materials.
- **Randomized Fallback:** In cases where there is no data available (e.g., very new users or those with minimal activity), the system resorts to recommending a random selection of resources or modules. This approach ensures that users are not left without content to explore, allowing them to begin engaging with the platform and discover new materials.

These strategies collectively ensure that learners are never left without guidance, supporting uninterrupted engagement and fostering a sense of progress. As user data accumulates, the system gradually shifts from generalized to personalized recommendations, enabling a smooth transition toward adaptive learning.

4.4.7 Semantic Search Engine

The semantic search functionality was designed to meet the needs expressed in user stories US4, US7, US12, and US14, providing an efficient and intuitive way to discover relevant content. The semantic search engine enables users to quickly find relevant resources, modules, and learning paths that match their interests or specific educational needs.

From an educational perspective, search supports self-directed and inquiry-based learning. It allows students to take initiative, ask their own questions, and retrieve content on demand—principles that align with learner autonomy and lifelong learning. By encouraging exploration outside of structured learning paths, the platform fosters curiosity and deeper engagement.

Search Pages and Dynamic Filtering

The platform includes two dedicated search pages: one for resources and another for learning content (modules and learning paths). Figure 4.16 shows the resource search page, where users can enter natural language queries and apply dynamic filters to refine the results.

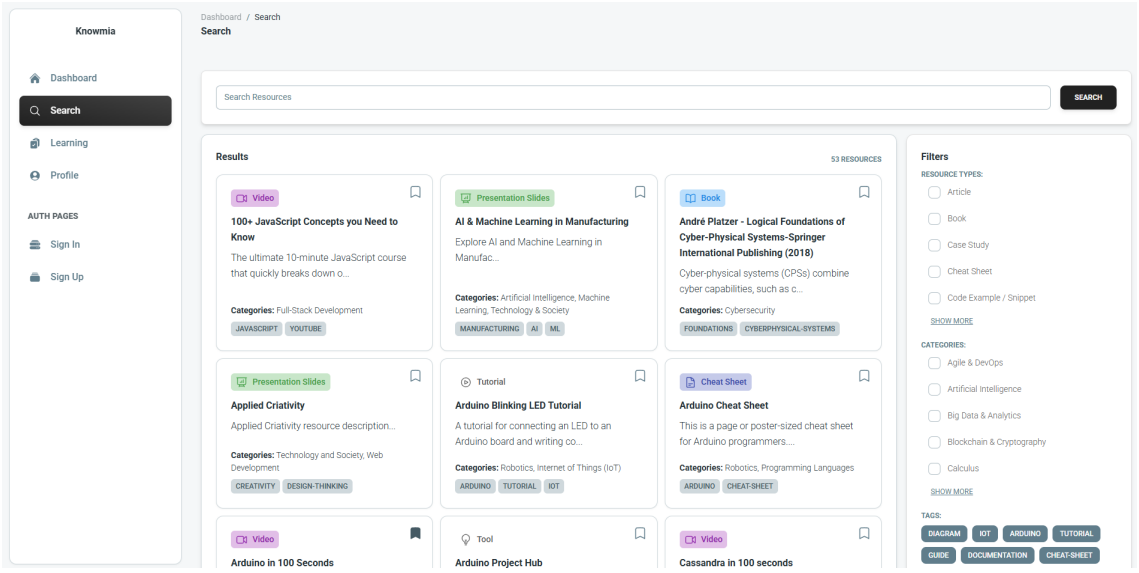

Figure 4.16: Search page for resources with filtering options.

Filters on this page help narrow down results by:

- **Resource Type:** Filter by type, such as articles, videos, tutorials, and more.
- **Category:** Filter by academic subject or course area (e.g., Computer Science, Mathematics).
- **Tags:** Filter by thematic tags such as “AI,” “Data Science,” or “Sustainability.”

Figure 4.17 presents the learning content search page, which displays both modules and learning paths and includes additional filters designed to support instructional alignment and user choice.

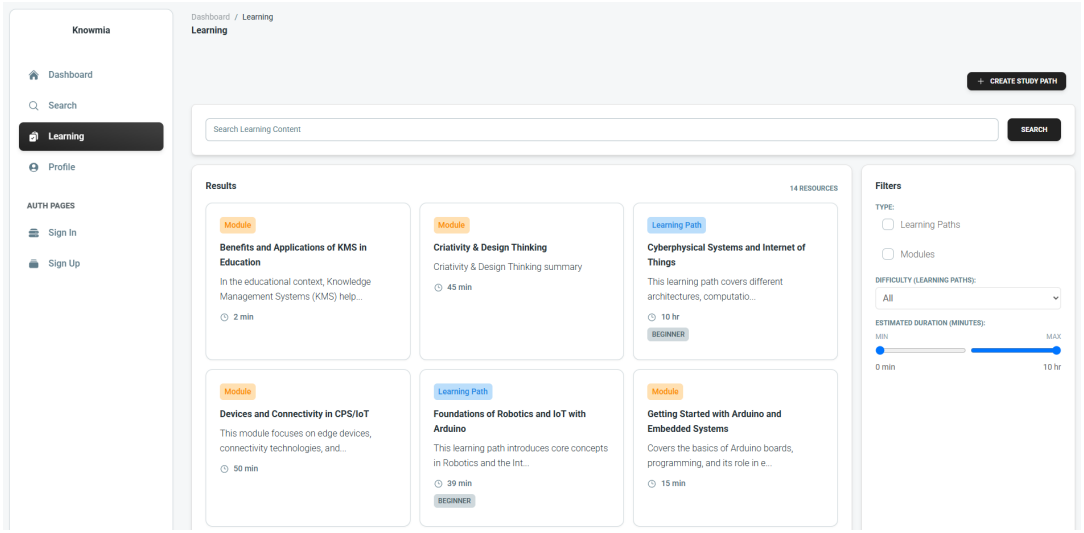

Figure 4.17: Learning search page displaying modules and learning paths.

Filters available here include:

- **Learning Type:** Distinguishes between standalone modules and sequential paths.
- **Difficulty:** Enables learners to choose content that matches their current proficiency.
- **Duration:** Helps users find content that fits their available study time, supporting better planning.

These filters operate together with the main search query, ensuring results are not only relevant but also tailored to individual learning preferences. Filters can be toggled on or off, making the search process flexible, intuitive, and responsive to diverse learning needs.

Search Results

Currently, the system limits the number of results displayed to the top 10 per query. This decision was due to the scarcity of available resources and to simplify validation during the evaluation phase. However, in a real-world scenario, promoting focused browsing and limiting cognitive overload, especially for new users, can enhance decision-making and engagement within the platform.

4.4.8 Alignment with Functional Requirements

To reinforce the alignment between the system's design and the functional needs expressed by its users, Table 4.3 provides a detailed mapping of each core feature to the corresponding user stories. These user stories capture both student and educator perspectives and served as the foundation for translating user expectations into practical, usable features.

Table 4.3: Detailed mapping of core features to user stories and functional requirements.

Core Feature	**User Story ID**	**Functional Requirement**
User Registration	US1	Sign up and select role (student or educator)
	US2	Secure login and resume learning
	US3	Set preferences to enable personalized experience
Resource Upload and Management	US16	Upload new learning resources
Modules and Learning Path Progress	US8	Access modules individually
	US9	Enroll in learning paths
	US13	Track progress across modules and paths
	US17	Create modules as an educator
	US18	Create learning paths as an educator
	US20	Add assessments to modules during creation
Assessments and Learning Evaluation	US11	Complete assessments to monitor understanding
User Dashboard	US5	Bookmark and revisit learning resources
	US13	Display user progress and completion status
	US19	Manage all created content in one place
Recommendation System	US6	Discover content related to current context
	US10	Receive personalized suggestions based on preferences and history
Semantic Search Engine	US4	Search using natural language or keywords
	US7	Browse available learning materials
	US12	Inspect detailed metadata before accessing resources
	US14	Apply filters by category, content type, and more
Study Paths	US15	Create custom study paths by grouping modules of interest

4.5 Implementation Challenges and Solutions

Several implementation challenges emerged throughout the development of the platform, requiring iterative adjustments and technical workarounds.

One of the most challenging difficulties involved the integration of an ontology into the backend. Although *WebProtégé* [60] provided a user-friendly interface for designing and exporting ontologies, integrating the generated *OWL* file into the backend logic was a struggle. Much of the

available documentation was either too outdated or abstract, and it was unclear how to operationalize the concepts defined in the ontology within the functional architecture of a *Node.js* application. After experimenting with various approaches and attempting to extract and map relevant concepts, the effort ultimately proved too complex and time-consuming for the available timeframe. While the ontology remained useful as a design reference to define entities and relationships, its direct integration was discarded. Instead, semantic capabilities were implemented using a more pragmatic and educationally effective solution: semantic *Elasticsearch* following a hybrid approach, enabling concept-aware retrieval without formal reasoning. This shift preserved the system's ability to offer contextually rich and intelligent search results, aligning with the pedagogical goals of supporting meaningful knowledge discovery.

This strategic realignment directly informed the subsequent challenge of embedding semantic capabilities into the search engine itself. The initial approach used a transformer-based model via the *HuggingFace* Inference API, but this method proved unreliable, as repeated authentication token failures interrupted the embedding process. To maintain control and performance, the model pipeline was installed and run locally using the *HuggingFace* transformers library. This ensured stable and fast embedding generation for resources and learning content. The resulting vectors were indexed in *Elasticsearch* alongside traditional fields, enabling hybrid semantic search with improved performance and full autonomy over the inference environment.

Containerizing services using *Docker Compose* introduced additional challenges, especially in managing the backend. While *Elasticsearch* and *Neo4j* ran reliably in containers, the backend encountered repeated build issues due to *Node.js* module resolution errors and inconsistent environment variables. These issues, combined with slow rebuild cycles and high system resource consumption, ultimately led to a compromise: running backend and frontend services in local development mode while keeping the data services containerized. This hybrid setup balanced development agility with reproducibility and allowed prioritization of implementing user-facing features without delays.

Another key challenge was synchronizing data between *PostgreSQL* and *Neo4j* to support the recommendation system, which relied on relationship data derived from user interactions. A custom *Node.js* script (Appendix B) was developed to extract users, resources, modules, and interactions from the relational database and transform them into graph structures suitable for *Neo4j*. This script was designed to be modular, efficient, and rerunnable, enabling the graph database to reflect the most recent learning behaviors and thereby generate accurate and relevant recommendations.

These technical decisions were always guided by the need to deliver a functional, pedagogically valuable system. Each workaround or adjustment preserved the integrity of the platform's educational objectives, ensuring that features like personalized search, adaptive recommendations, and structured learning paths remained central to the user experience.

The challenges encountered throughout the implementation phase shaped a system architecture that prioritized flexibility, reliability, and educational value. These experiences also highlight

the trade-offs often faced in real-world educational software development, where technical feasibility must be balanced with pedagogical intent. The resulting system, while not without its compromises, was able to support the core learning features envisioned during the design stage and serve as a solid foundation for both evaluation and future refinement.

Chapter 5

Evaluation

This chapter presents an evaluation of the intelligent knowledge management system developed in this thesis, with a focus on assessing its performance in three core areas: personalized recommendations, semantic information retrieval, and structured content progression. The evaluation was designed to reflect realistic usage within the context of engineering education and is directly aligned with the fourth research question:

> *RQ4: How can the effectiveness of an intelligent KMS be evaluated in terms of user satisfaction, learning outcomes, and system performance in engineering education?*

To answer this question, a mixed-methods approach was adopted, combining system-level performance metrics with user-centered feedback. Ten students from the Faculty of Engineering of the University of Porto (FEUP) participated in guided testing sessions lasting 30-40 minutes. These sessions simulated the end-to-end experience of using the platform, from signing in and configuring preferences to exploring recommendations, performing topic-based searches, completing structured learning paths, and submitting feedback.

The evaluation was conducted along three dimensions:

1. **Semantic Search and Recommendation Performance** – measured through quantitative metrics such as Precision@k (P@k), Recall@k (R@k), average precision (AP), and mean average precision (MAP).

2. **Usability and User Satisfaction** – assessed via post-session surveys, Likert-scale responses, and the System Usability Scale (SUS).

3. **Educational Value and Content Progression** – analyzed through assessment scores and user reflections on the learning paths.

This chapter outlines the evaluation methodology, tasks and data collection methods in Section 5.1. Next, in Section 5.2, the quantitative analysis of the system performance across different features is presented. The qualitative user feedback and usability outcomes is summarized in Section 5.3. Finally, Section 5.4 discusses the key findings and triangulates results from both evaluation perspectives.

5.1 Methodology

This section outlines the evaluation methodology adopted to assess the system's technical performance, usability, and educational value. The study used a task-based user testing protocol combined with interaction logging and post-session surveys. Although sessions were conducted locally on a single computer, the evaluation methodology was designed to simulate realistic platform usage.

5.1.1 Participants

A total of ten participants were recruited for the evaluation. All were enrolled in the Informatics and Computing Engineering program at FEUP. The group included both bachelor's and master's students, ensuring a diverse set of user profiles across academic levels and topic familiarity.

Prior to the test, participants self-reported the following information to contextualize results:

- Current education level
- Year of study
- Prior knowledge of the topics covered during the testing session

This demographic data helped contextualize their feedback and interpret results such as search success and learning outcomes based on their academic experience and topic familiarity.

5.1.2 Evaluation Tasks

Participants completed a structured sequence of five tasks designed to simulate the system's intended use:

1. **User Onboarding**
    - **Task:** Participants registered into the system, selected their preferences, and configured their profiles.
    - **Purpose:** To test the cold-start logic of the recommendation engine and capture profile-based personalization.
2. **Resource Exploration and Recommendation Validation**
    - **Task:** Users interacted with their personalized dashboard. They were encouraged to bookmark useful resources, interact with recommendations, and rate their relevance based on the submitted preferences at sign-up.
    - **Purpose:** To test the performance of the Neo4j-based graph recommendation system and assess perceived personalization quality.
3. **Semantic Search Task**

- **Task:** Participants searched for predefined topics using natural language queries. After reviewing the results, they rated the relevance of the retrieved resources.
- **Purpose:** To evaluate the semantic search engine's ability to return contextually accurate results using Elasticsearch and embedding-based matching.

4. **Learning Path Interaction**

   - **Task:** Participants selected one of two learning paths and completed its modules. Each module included resources and an assessment to verify knowledge gained.
   - **Purpose:** To assess the structured delivery of educational content, the usability of learning progression logic, and the effectiveness of assessments.

5. **Feedback Submission**

   - **Task:** At the end of the session, participants completed a feedback form to evaluate usability, visual design, task clarity, recommendation quality, search performance, and learning outcomes.
   - **Purpose:** To collect qualitative data about user experience and satisfaction and identify areas for improvement.

Each session lasted approximately 30–40 minutes.

5.1.3 Data Collection

To ensure a comprehensive assessment, multiple types of data were collected throughout the user testing sessions, including interaction logs, performance metrics, and participant feedback.

Interaction Logs. User activities were logged during each session, including search queries, relevance ratings, resource views and bookmarks, and assessment completions. These logs enabled reconstruction of user behavior and interaction patterns across different stages of engagement.

Quantitative Metrics. Standard information retrieval metrics were applied to evaluate the semantic search and recommendation performance. These include Precision@k, which measures the proportion of relevant items among the top-k retrieved results (Equation 5.1); Recall@k, which captures the fraction of all relevant items that were retrieved (Equation 5.2); Average precision, which averages the precision at ranks where relevant documents appear (Equation 5.3); and mean average precision, which aggregates AP scores across all queries (Equation 5.4). Relevance judgments were manually assigned to the top-k results for each query based on content-topic alignment.

- **Precision@k:** Proportion of relevant items in the top-k results.

$$\text{Precision@}k = \frac{\text{Number of relevant items in top } k}{k} \tag{5.1}$$

- **Recall@k:** Proportion of relevant items retrieved among all relevant items available.

$$\text{Recall@}k = \frac{\text{Number of relevant items in top } k}{\text{Total number of relevant items}} \tag{5.2}$$

- **Average Precision:** Average of precision values at each rank where a relevant item is retrieved.

$$\text{AP} = \frac{1}{R} \sum_{k=1}^{n} P(k) \cdot rel(k) \tag{5.3}$$

  where R is the number of relevant items, $P(k)$ is the precision at rank k, and $rel(k)$ is a binary indicator (1 if the item at rank k is relevant, 0 otherwise).

- **Mean Average Precision:** Mean of AP values across all queries.

$$\text{MAP} = \frac{1}{Q} \sum_{q=1}^{Q} \text{AP}(q) \tag{5.4}$$

  where Q is the total number of queries.

Controlled Scenarios. Predefined test cases were used to simulate three recommendation states: cold start (no preferences), profile-based (preferences only), and personalized (after user interactions). These controlled conditions allowed for consistent comparison of fallback and personalized recommendation strategies.

Qualitative Feedback. Post-session surveys combined Likert-scale questions with open-ended prompts, capturing participants' perceptions of usability, content relevance, system clarity, and educational value.

Assessment Outcomes. Assessment results from completed modules were used to analyze short-term learning progression. Each assessment included auto-graded questions based on the module's objectives. While not measuring advanced mastery, these results offered insight into comprehension and content accessibility.

5.2 Quantitative Evaluation

This section presents the results of the quantitative evaluation of the semantic search and recommendation features of the developed system. The goal is to assess how effectively the system retrieves and recommends relevant content based on user context and queries.

The evaluation focuses on two core functionalities:

- **Semantic Search** — evaluated using a set of predefined natural language queries.
- **Recommendation System** — evaluated across different user states and platform contexts.

All metrics used in this section, including Precision@k, Recall@k, Average precision, and mean average precision, were described in Section 5.1 and applied using manually assigned relevance labels.

5.2.1 Semantic Search

The semantic search engine was tested using seven representative queries designed to reflect realistic information needs within an educational context. These queries were entered into the system, and the top 5 results were evaluated for relevance. Table 5.1 presents the queries used in this test.

Table 5.1: Evaluated semantic queries.

Query ID	Query Text
Q1	What is knowledge management?
Q2	Arduino for beginners
Q3	How to use AI in Manufacturing?
Q4	Industry 4.0 vs Industry 5.0
Q5	Differences between NoSQL databases
Q6	Introduction to Cyber-Physical Systems
Q7	Design Thinking and Product Development

Each query returned a ranked list of up to 10 results, of which the top 5 were evaluated. Relevance was judged based on alignment between the resource content and the query intent. Tables 5.2 and 5.3 summarize Precision@5, average precision, and Recall@5 for each query. Figure 5.1 visualizes comparative performance.

Table 5.2: Precision@5 and average precision per query.

Query ID	Relevance	P@5	Total Relevant	AP Calculation	AP
Q1	10111	0.80	5	1.0, 0.67, 0.75, 0.80	0.644
Q2	11101	0.80	6	1.0, 1.0, 1.0, 1.0	0.667
Q3	10101	0.60	4	1.0, 0.67, 0.60	0.568
Q4	11101	0.80	6	1.0, 1.0, 1.0, 0.80	0.633
Q5	11111	1.00	6	1.0 (×5)	0.833
Q6	01100	0.40	4	0.5, 0.67	0.292
Q7	11100	0.60	4	1.0, 1.0, 1.0	0.750
				MAP	**0.627**

Table 5.3: Recall@5 per Query

Query ID	Relevant Retrieved	Total Relevant	R@5
Q1	4	5	0.80
Q2	4	6	0.67
Q3	3	4	0.75
Q4	4	6	0.67
Q5	5	6	0.83
Q6	2	4	0.50
Q7	3	4	0.75
		Average Recall@5	**0.714**

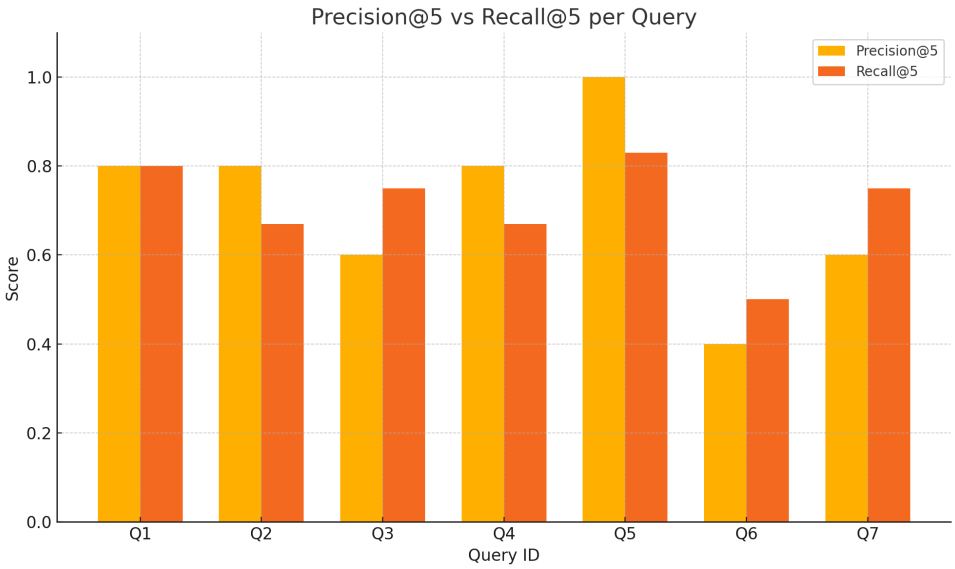

Figure 5.1: Precision@5 and Recall@5 scores across evaluated queries.

The semantic search system demonstrated reliable performance, achieving a MAP of 0.627 and an average R@5 of 0.714 across the set of queries. As shown in Figure 5.1, P@5 and R@5 values were closely aligned for most queries, indicating that the top-ranked results were generally relevant and comprehensive. Q1 and Q5, for example, achieved both high precision and recall, suggesting effective retrieval and ranking for well-defined topics.

While Precision@5 and Recall@5 were generally aligned, some divergence was observed; for example, in Q3 and Q7, where recall was slightly higher than precision. This implies that relevant documents were present among the retrieved results, but not always concentrated at the top of the ranking. The highest-performing query, Q5, reflected strong alignment between query intent and available resources. In contrast, lower-scoring queries like Q6 suggest potential areas for improvement, such as expanding topic coverage or refining indexing strategies.

5.2.2 Recommendation System

The recommendation system was evaluated through a series of controlled user scenarios to test how well it adapts to different recommendation contexts and personalization states. As described in Section 5.1, these scenarios simulated different stages of user interaction, including cold-start behavior and progressively personalized responses. Table 5.4 outlines the main locations within the platform where recommendations appear.

Table 5.4: Recommendation locations within the platform.

Location	Description
Dashboard	Personalized recommendations based on interactions and preferences.
Resource Page	Recommended resources and modules related to a specific resource.
Module Page	Recommended modules and learning paths related to the current one.
Learning Path Page	Recommended resources to expand knowledge beyond the current learning path.

Dashboard Recommendations

The dashboard was tested under three conditions: cold start (no preferences), profile-based (using selected preferences), and personalized (after multiple interactions). These scenarios were designed to evaluate the fallback strategies and personalization logic used in the system.

1. **Cold Start:** A new user without preferences triggered a fallback query that returned the most popular resources on the platform.

2. **Profile-Based:** After setting preferences for preferred topics (e.g., Fullstack Development, Artificial Intelligence, and Internet of Things) and content types (e.g., Articles, Videos, and Presentation Slides), recommendations were filtered according to the first fallback query, based on user preferences.

3. **Personalized:** After interacting with several resources, recommendations were tailored using both preferences and behavior history.

Table 5.5 shows the number of relevant items returned and the resulting Precision@6 for each scenario.

Table 5.5: Recommendation quality on the dashboard across user states.

User State	Fallback Used	Items Shown (k)	Relevant	P@k
Cold Start	Yes (popular content)	6	–	–
Preferences Only	Yes (preference match)	6	6	1.00
After Interactions	No	6	6	1.00

In the cold-start scenario, the system returned the most interacted-with resources overall (e.g., "*100+ JavaScript Concepts You Need to Know*" and "*AI & Machine Learning in Manufacturing*"). These results are ranked based on total interaction counts across all users of the platform. Since these recommendations are not specific to any user profile or behavior, relevance was not formally evaluated in this scenario.

In the profile-based scenario, all recommended resources were aligned with the selected topics and content types. For example, resources such as "*Next.js in 100 Seconds // Plus Full Beginner's Tutorial*" (Video, Full-Stack Development) and "*Concepts and Architectures for CPS and IoT*" (Presentation Slides, Internet of Things) clearly matched the specified interests. This consistency resulted in a perfect Precision@6 of 1.00.

In the personalized scenario, the recommendation engine effectively filtered out previously viewed and bookmarked content while still adhering to the user's interests. Although two resources were repeated from previous stages, these had not yet been interacted with. This demonstrates that the system correctly avoids recommending content already explored, while still prioritizing relevance, once again achieving Precision@6 of 1.00.

Resource Page Module Recommendations

This test focused on how well the system recommends modules when viewing a resource. Two cases were considered: when the resource is already part of a module and when it is standalone.

Table 5.6, displays the results obtained for the evaluation of module recommendation relevance on the resource page.

Table 5.6: Recommendation relevance based on resource context.

Resource Type	Items Shown	Relevant	P@k	Fallback Used
Part of Module	1	1	1.00	No
Standalone Resource	2	2	1.00	Yes

In the first scenario, a resource that is part of an existing module, for example, "*What is a Knowledge Management System,*" included in the "*What is Knowledge Management*" module, was selected. The system recommended only one module (its own), which aligns with expectations, given that the resource is not reused in other existing modules. This reflects a correct and focused recommendation, though limited in scope due to current platform content.

In contrast, for standalone resources not currently associated with any module, such as "*Database System Concepts*," the system returned two modules: "*What is Knowledge Mangement*" and "*Introduction to Cassandra Wide-Column Store NoSQL Database*." While still below the expected maximum of five, this outcome is attributed to the limited quantity of module content available in the system. Importantly, the recommendations were contextually appropriate: both modules include resources categorized under topics like "Database Systems," "Software Engineering," and "Knowledge Management Systems," which match the standalone resource's categories.

These results demonstrate that the recommendation engine is capable of establishing thematic similarity through shared metadata such as categories. However, the number of recommended items remains limited due to the current size of the content database. As more modules are added to the system, the recommendation space is expected to expand and yield more varied and precise results.

Module Page Learning Path Recommendations

Next, the system was tested on the module page, where it recommends learning paths. The test compared results for modules that already belong to learning paths and those that do not.

Table 5.7, displays the results obtained for the evaluation of learning path recommendation relevance on the module page.

Table 5.7: Recommendation relevance in module contexts.

Module Type	Items Shown	Relevant	P@k	Fallback Used
With Learning Path	1	1	1.00	No
Without Learning Path	3	2	0.67	Yes

For example, when viewing the module "*What is Knowledge Management*," which is part of the "*Introduction to Knowledge Management System*" learning path, the recommendation system correctly returned that specific learning path. While the module could theoretically belong to multiple learning paths, the system's response is accurate given the current amount of content.

In contrast, when accessing a standalone module such as "*Creativity & Design Thinking*," the system recommended all three available learning paths. However, only two of them were judged to be topically relevant. Since the module had no predefined learning path associations, the fallback strategy relied on global learning path popularity to populate the recommendation list.

These results indicate that while the recommendation logic successfully prioritizes actual module-to-path relationships when available, its fallback strategy could benefit from additional filtering based on content similarity or category alignment. As the number of modules and learning paths increases, the precision of these recommendations is also expected to improve.

Learning Path Page Resource Recommendations

The final test assessed resource recommendations shown on learning path pages. Since each learning path covers different topics, the goal was to determine if recommended resources were distinct and relevant.

- LP1 - Introduction to Knowledge Management System
- LP2 - Getting Started with Arduino
- LP3 - Cyberphysical Systems and Internet of Things

Table 5.8 shows the number of items shown, how many were unique, and how many were considered relevant.

Table 5.8: Resource recommendations by learning path.

Learning Path ID	Items Shown	Unique Recommendations	Relevant	P@6
LP1	6	0	3	0.50
LP2	6	0	2	0.33
LP3	6	2	4	0.67

The same six resources appeared for both LP1 and LP2, despite covering different topics. This overlap led to low uniqueness and only moderate relevance. LP3, however, included two distinct items aligned with the learning path theme, which resulted in a higher Precision@6 of 0.67.

These results highlight a limitation in the current scoring strategy, which relies mostly on textual similarity and popularity. Integrating additional filters such as tags or semantic embeddings could reduce overlap and improve relevance across learning paths.

5.3 Qualitative Feedback

To better understand how users experienced the platform, qualitative feedback was collected at the end of each evaluation session. Participants completed a feedback form (Appendix D) that included Likert-scale statements and open-ended questions. These responses covered ease of use, clarity of the interface, usefulness of recommendations, effectiveness of learning content, and general satisfaction with the system. This section presents an overview of the responses provided by the 10 participants who completed the evaluation.

5.3.1 Participant Information

All 10 participants were enrolled in the Informatics and Computing Engineering program and represented a balanced distribution between bachelor's and master's students (5 each). They had different levels of academic experience and familiarity with the topics covered during the session.

Figure 5.2 shows the breakdown by academic level, while Figure 5.3 shows their year of study.

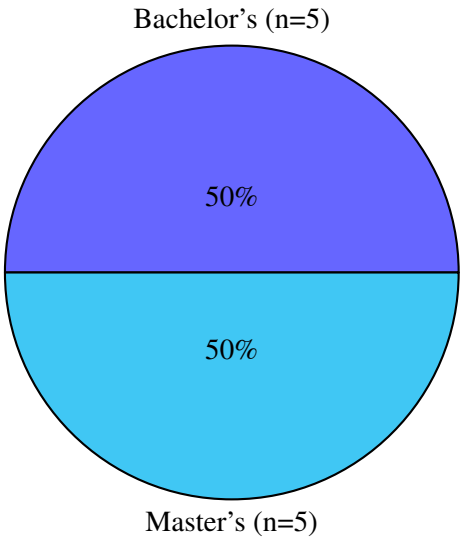

Figure 5.2: Distribution of participants by academic level.

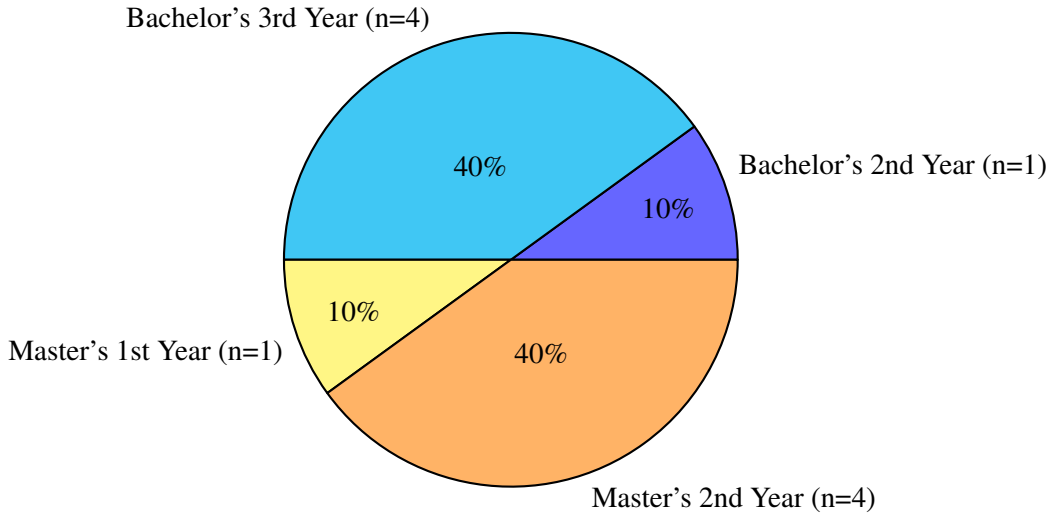

Figure 5.3: Distribution of participants by year of study.

The academic year distribution shows participants primarily from advanced stages of their studies, with the majority in their third year of bachelor's studies (40%) or second year of master's studies (40%). This distribution ensures representation from both undergraduate and graduate perspectives, with participants having sufficient foundational knowledge to engage meaningfully with the evaluation materials.

Additionally, participants were asked to self-assess their prior knowledge of the topics explored during the evaluation, as illustrated in Figure 5.4.

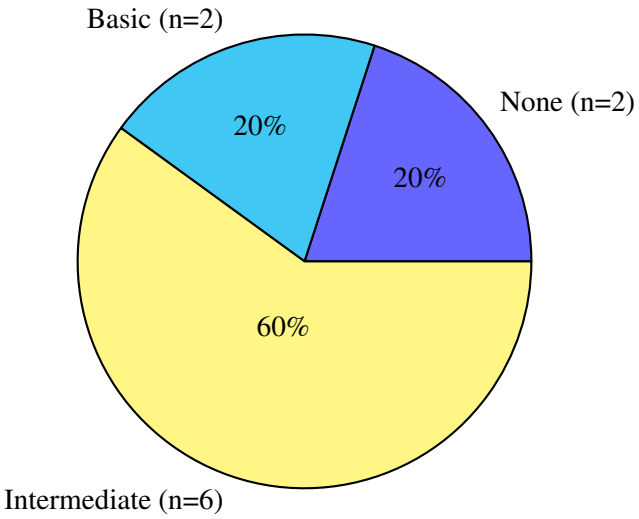

Figure 5.4: Self-assessed prior knowledge distribution.

Figure 5.4 shows that the majority of participants considered themselves to have intermediate knowledge, while fewer identified as having only basic or no prior familiarity. This distribution helps contextualize the evaluation results, as differences in prior knowledge may have influenced how participants understood the questions and how they performed in the assessments.

A more detailed view of the participants' profiles is provided in Table 5.9.

Table 5.9: Detailed participant profiles.

Participant ID	**Education Level**	**Year of Study**	**Prior Knowledge Level**
P1	Bachelor's	3rd	Basic
P2	Bachelor's	3rd	Intermediate
P3	Bachelor's	2nd	Intermediate
P4	Master's	2nd	Intermediate
P5	Master's	2nd	Basic
P6	Bachelor's	3rd	None
P7	Bachelor's	3rd	Intermediate
P8	Master's	1st	Intermediate
P9	Master's	2nd	Intermediate
P10	Master's	2nd	None

5.3.2 Usability and Visual Design

Participants were asked to rate their agreement with several usability-related statements on a 5-point Likert scale (from Strongly Disagree to Strongly Agree). The statements focused on how easy the platform was to use, how intuitive the interface felt, and how well users understood what actions to take.

1. I found the system easy to use.

2. The visual design and layout of the platform were clear and intuitive.
3. I understood what to do at each step.
4. The learning path structure was clear.
5. The assessments were relevant.
6. I felt that I learned new or useful information from the learning paths/modules.
7. The search system seemed to understand the meaning of what I typed, not just the keywords.
8. I would use a system like this in my studies.

Figure 5.5 presents the distribution of responses.

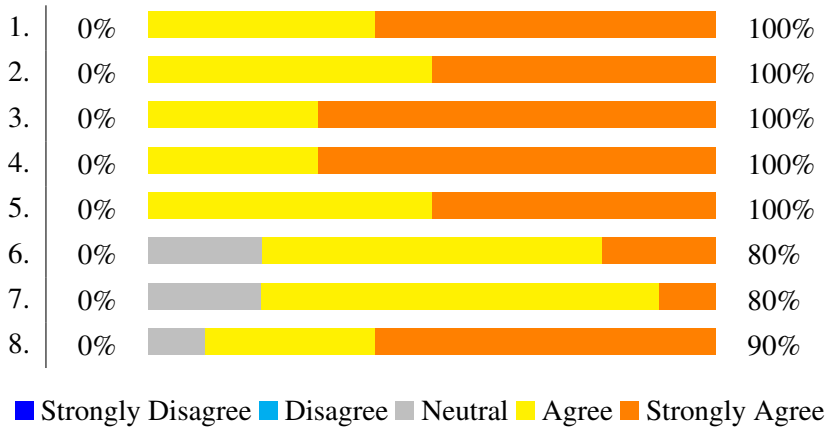

Figure 5.5: Participant agreement on usability and visual design.

Most participants responded positively, especially in relation to ease of use, task clarity, and layout.

5.3.3 Recommendation Perception and Semantic Search

Participants were asked about their impressions of the recommendation system and semantic search. All users observed that recommendations changed throughout the session, which confirmed that the platform successfully transitioned from fallback suggestions to personalized results. This was seen as a positive feature.

Table 5.10 presents some example queries and their corresponding relevance ratings, along with participant comments.

Table 5.10: Example queries and search result ratings.

Query	**User Rating (1–5)**	**Comments**
"actuator flow"	4	Returned relevant Arduino-related resources.
"intro to javascript"	4	Included resources related to javascript and programming languages.
"industry systems"	4	Resources contained relevant mentions of industry topics.
"arduino sensors"	3.5	Mostly relevant Arduino resources; some mismatches.
"difference between nosql databases"	4	Result #3 should be ranked higher.
"introduction to cyber-physical systems"	4	Only 4 relevant resources in the top 5
"basics arduino"	4	First result was off-topic (Python); others were relevant.

Most users felt that the system returned results that matched what they meant, not just the exact words they typed. However, a few responses pointed out occasional mismatches or limitations, which can be due to the scarce number of resources and variety.

5.3.4 Learning Path Outcomes

Participants completed one of two possible learning paths during the session:

- LP1 - Introduction to Knowledge Management Systems
- LP2 - Getting Started with Arduino

Table 5.11 shows which learning path each participant selected and their assessment scores (out of 100). Most participants achieved full scores in both modules, regardless of their prior knowledge.

Table 5.11: Learning path selection and assessment scores.

Participant	Learning Path	Assessment Score Module #1	Assessment Score Module#2
P1	LP1	100	100
P2	LP2	100	100
P3	LP2	100	100
P4	LP2	100	100
P5	LP1	100	100
P6	LP2	67; 100	100
P7	LP1	100	100
P8	LP1	100	100
P9	LP1	100	100
P10	LP2	67; 100	100

While all users completed the paths successfully, these assessments were focused on basic understanding. More advanced or varied question types would be needed to evaluate deeper learning in future iterations.

5.3.5 Frustrations and Suggestions

Although the overall feedback was positive, participants shared a number of valuable suggestions and minor frustrations they experienced that can inform future improvements:

- **Assessment Review:** Users wanted the ability to revisit the answers of submitted assessments.
- **Recommendation Placement:** When completing a learning path, seeing unrelated recommendations inside modules was seen as confusing and distracting, disrupting the expected flow.
- **Search Persistence:** Search queries were lost after clicking a result and navigating back to the search page.
- **Performance:** Dashboard loading was slow, especially on first access.
- **UI Suggestions:**
  - Locked module cards showed a pointer cursor, which was misleading.
  - Sidebar cards should remain fixed while scrolling to improve readability of page content.
  - The dashboard had too many sections and felt visually cluttered.

Some feedback also pointed to future opportunities:

> "*If integrated with my course units, the system could recommend materials based on my weaknesses, which would be incredible.*"

This comment supports the idea of using assessment results to guide recommendations, which is a feature considered but not implemented in the current prototype.

Another participant noted:

> "*Yes, I would use it regularly. Particularly in engineering, there is a lack of centralized systems where we can find useful information on different topics.*"

This highlights the current gaps in KMS for education, where resources are not easily discoverable to support the educational trajectory of a student.

5.3.6 System Usability Scale

To complement qualitative feedback, the 10-item system usability scale was used to assess perceived usability. SUS is a widely recognized and reliable tool for measuring system usability, originally developed by John Brooke in 1996 [3]. It gives a score from 0 to 100, with higher values indicating better perceived usability. Each participant completed the questionnaire after finishing the session.

The 10 statements included in the SUS questionnaire are as follows:

1. I think that I would like to use this product frequently.
2. I found the product unnecessarily complex.
3. I thought the product was easy to use.
4. I think that I would need the support of a technical person to be able to use this product.
5. I found the various functions in this product were well integrated.
6. I thought there was too much inconsistency in this product.
7. I imagine that most people would learn to use this product very quickly.
8. I found the product very awkward to use.
9. I felt very confident using the product.
10. I needed to learn a lot of things before I could get going with this product.

The SUS score is calculated using a standard formula: for each of the 10 items, odd-numbered statements (positive) are scored as the response value minus 1, while even-numbered statements (negative) are scored as 5 minus the response value, as defined in Equation 5.5:

$$s_i = \begin{cases} \text{Response}_i - 1 & \text{if } i \text{ is odd} \\ 5 - \text{Response}_i & \text{if } i \text{ is even} \end{cases} \tag{5.5}$$

The adjusted scores for all ten questions are then summed and multiplied by 2.5 to produce a final SUS score on a scale from 0 to 100, as shown in Equation 5.6.

$$\text{SUS Score} = \left(\sum_{i=1}^{10} s_i \right) \times 2.5 \tag{5.6}$$

Figure 5.6 summarizes the responses to each statement.

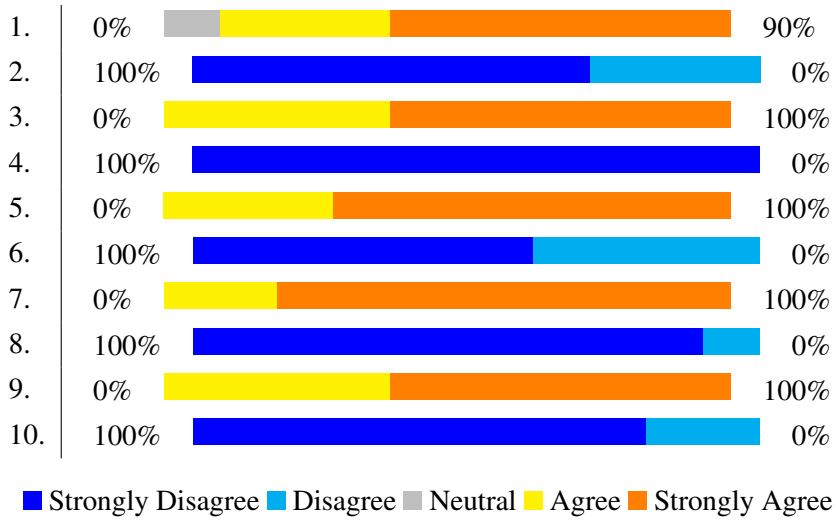

Figure 5.6: System usability scale question responses.

The average SUS score was 93.0, which is considered excellent. This confirms the positive experience reported throughout the session and suggests that the system is easy to learn and use.

5.4 Discussion of Results

Although the user testing sessions were limited in duration and sample size, the evaluation phase revealed several key insights regarding the system's technical performance, usability, and educational value.

The semantic search engine showed consistent performance across a range of queries, achieving a mean average precision of 0.627 and an average Recall@5 of 0.714. Queries with specific scopes generally produced highly relevant results, while broader or interdisciplinary queries had lower relevance due to limited resource coverage. These results suggest that the semantic search approach is effective when sufficient and topically aligned content is available.

The recommendation engine, particularly in personalized contexts that combined profile preferences and interaction history, achieved strong results with a Precision@6 of 1.00. This indicates that the system was able to match users with relevant content based on their interests and actions. However, limitations emerged in content-specific recommendation scenarios, especially on the learning path details page. In some cases, the same resources were recommended across different learning paths, even when those paths addressed different topics. This reflects a limitation in the

current scoring strategy, which relies mainly on textual similarity and interaction frequency. Without deeper metadata, for example, topic tags, semantic categories, or graph-based embeddings, the system tends to favor popular or generic resources. The performance of LP3, which demonstrated better recommendation diversity, reinforces the idea that more refined content filtering and similarity modeling could improve precision in future iterations.

All participants were able to complete their selected learning paths and successfully pass the associated module assessments. This suggests that the learning content was generally clear and accessible, even for users with limited prior knowledge of the topics. However, the high success rate and the intuitive nature of the assessments point to a limitation in their ability to evaluate deeper levels of understanding. One likely contributing factor is the absence of input from professional educators during the design of the assessments. As a result, the current evaluation materials focus primarily on basic recall and conceptual understanding. Future iterations of the system would benefit from collaboration with educators to design more diverse and cognitively demanding assessment types that better align with educational goals and learning outcomes.

User perception was highly positive across all evaluated dimensions. The platform received a SUS score of 93.0, which is considered excellent in terms of usability. Likert-scale responses and open-ended feedback further supported these results. Participants highlighted learning paths, search filters, and various personalized recommendations as features they enjoyed the most. Reported issues, such as the inability to review assessment answers or the disappearance of search queries upon navigation, were relatively minor and can be addressed in future iterations.

A common theme across all evaluation data is that the system's performance is highly dependent on the availability, diversity, and structure of content. Both the semantic search and recommendation components perform well under the right conditions, but their effectiveness is limited by gaps in topical coverage, lack of deeper metadata (e.g., semantic relationships), and limited learning materials. This points to the importance of continuous content expansion and metadata enrichment as the platform evolves.

The combination of quantitative metrics and qualitative feedback provides a comprehensive perspective on the system's performance and user experience. In general, the two data sources reinforce each other: the high Precision@k and MAP scores observed in the semantic search and recommendation tests align with the positive user perceptions reported in post-session surveys. Participants noted that the system "*understood what [they] meant, not just the keywords*," which supports the improved relevance observed in semantic search metrics. Similarly, the high precision across different recommendation contexts is reflected in users' agreement that the recommended materials matched their interests and preferences.

At the same time, a few contradictions were also informative. For example, while the Precision@k values for learning path recommendations were only moderate (ranging from 0.33 to 0.67), participants still expressed appreciation for the overall structure and flow of the learning content. This highlights how users may value content structure and flow even when specific recommendations are not perfectly contextually perfect.

In summary, this mixed-methods evaluation confirmed the potential of the platform while

also identifying concrete areas for improvement. The results validate core design decisions, such as using modular learning paths, semantic retrieval, and graph-based personalization, but also highlight the importance of content quality, recommendation diversity, and ongoing refinement of both algorithms and user-facing features.

Chapter 6

Discussion

This chapter reflects on the results obtained during the development and evaluation of the intelligent knowledge management system for engineering education. It interprets the key findings, connects them to the four guiding research questions, and discusses challenges and limitations.

6.1 Key Findings

The development of the platform aimed to address key limitations in knowledge management for engineering education by combining intelligent technologies with a user-centered design. The final solution focused on three main areas: improving information retrieval through semantic search, offering personalized content recommendations through a graph-based approach, and guiding students through structured learning experiences via modular learning paths. These components were designed in response to the project's research questions and were evaluated using a mix of qualitative and quantitative methods.

Figure 6.1 presents a conceptual model of the solution.

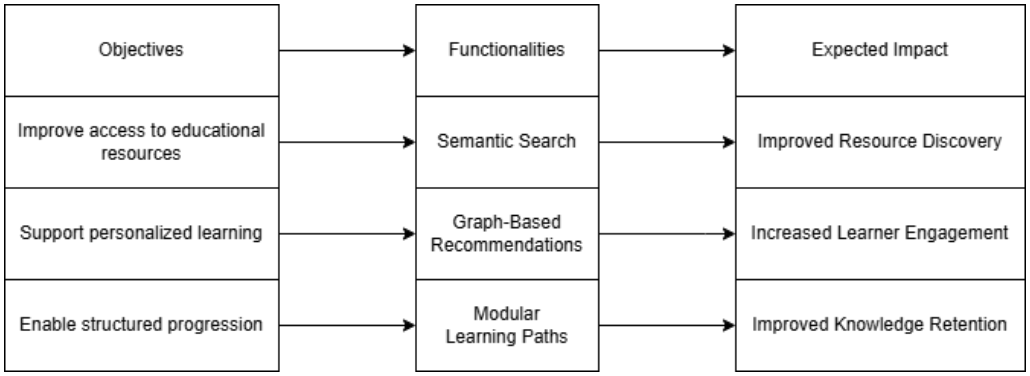

Figure 6.1: Conceptual model linking system objectives, functionalities, and impact.

This model helps clarify how the developed features respond directly to core educational limitations such as resource discoverability, relevance of content, and structured progression. These functionalities were not developed in isolation but as part of an integrated platform designed to

improve the learning experience through personalization, usability, and intelligent content delivery. Together, they demonstrate how even a relatively lightweight solution can address meaningful gaps in current knowledge management systems.

Semantic Search

The semantic search system helped improve how users find relevant content, going beyond basic keyword matching. By combining Elasticsearch with a transformer-based language model, the system could interpret natural-language queries and return more meaningful and context-aware results. This addressed a common issue in existing educational platforms where search tools often return shallow or irrelevant matches. Testing with a small but carefully curated set of queries, along with participant feedback, showed that users generally found the results more relevant and helpful.

Recommendation System

The recommendation engine was built using Neo4j and focused on generating personalized suggestions by analyzing relationships between users, resources, and learning materials. The graph structure made it possible to capture connections based on user interactions and content similarity. Although relatively simple in scope, the recommendation logic adapted over time as users engaged with the system. Evaluation through predefined user scenarios and user feedback suggested that the recommendations became more relevant after initial interactions. This confirmed the system's potential for personalization, even if some areas, such as recommendation diversity, still have room for improvement.

Modular Learning Paths

To support structured and progressive learning, the platform included modular learning paths with integrated assessments. Each module included a defined set of resources with a short quiz that had to be completed before progressing. User progress was tracked and stored in the database, updating automatically based on assessment results. While the assessments were basic and not created with educational experts, they still provided a way to verify concept understanding and maintain learning flow. The modular structure also helped break down topics into manageable steps, offering a clear and trackable learning experience. This structure lays the groundwork for future enhancements such as adaptive learning or feedback based on user performance.

6.1.1 Research Questions

The findings discussed above relate directly to the research questions defined at the beginning of this thesis:

RQ1. What design strategies can be applied to implement a knowledge management system that accommodates the varying needs of students and educators in engineering education? To support both student and educator needs, the system followed a human-centered design approach. Different user roles were considered early in the design phase by creating personas and mapping their typical workflows. This helped identify essential features such as resource creation, search, progress tracking, and personalized recommendations. The concept of creating low-fidelity prototypes made it easier to test and refine interface decisions, improving the system's overall usability.

RQ2. How can intelligent technologies enhance information retrieval, personalization, and resource accessibility in knowledge management systems for engineering education? To enhance information retrieval and resource accessibility, the system integrated two intelligent technologies: semantic search and graph-based recommendations. Semantic search was implemented by generating dense vector embeddings from resource titles and descriptions using a transformer-based language model (`all-MiniLM-L6-v2`). These embeddings were indexed and queried using Elasticsearch, enabling the system to match semantically similar content even when users' queries did not contain exact keywords. This approach addressed common limitations of traditional keyword-based search by allowing more natural language interaction and improving the relevance of retrieved resources.

For personalization, a graph-based recommendation engine was developed using Neo4j. The graph modeled relationships between users, learning resources, modules, and topics, incorporating interaction data such as resource views and completed modules. Based on these connections, the system was able to suggest context-aware learning materials aligned with each user's activity and preferences. A fallback strategy based on popular resources was also implemented to support new users with no prior interactions.

Together, these technologies made the platform more adaptive to individual learning needs and enhanced the discovery of relevant content. By supporting semantic understanding and personalization, the system advanced beyond static or manually curated approaches and demonstrated how intelligent components can improve both accessibility and engagement in educational KMS environments.

RQ3. What are the existing gaps and limitations in knowledge management systems for engineering education, and how can they be addressed through innovative approaches? Many existing educational platforms still struggle with outdated interfaces, rigid workflows, ineffective search tools, and limited support for personalization. This project aimed to address these issues by developing a modular system that supports structured learning progression and personalized content discovery. The use of semantic search helped address the limitations of traditional keyword-based retrieval by enabling the system to interpret the meaning behind user queries rather

than relying on exact word matches. In addition, the recommendation engine introduced a foundational approach to personalized learning by suggesting relevant content based on user behavior and individual preferences.

The project also focused on maintaining a clean and consistent user experience. Design decisions were influenced not only by academic research but also by real student needs and experiences. This helped avoid common usability issues found in other platforms, such as confusing navigation or disconnected learning features.

RQ4. How can the effectiveness of an intelligent KMS be evaluated in terms of user satisfaction, learning outcomes, and system performance in engineering education? The effectiveness of the system was evaluated through a combination of quantitative and qualitative methods, focusing on the relevance of semantic search results, the quality of personalized recommendations, and the overall user experience. As detailed in Chapter 5, the semantic search system was evaluated using a manually constructed set of user queries and expected results, enabling precision and recall analysis. In addition, qualitative feedback was collected through participant user testing sessions, where users interacted with the search feature and commented on the relevance and usefulness of the results. Together, these methods helped verify the system's ability to return contextually appropriate resources, addressing the information retrieval aspect of system performance.

Personalized recommendations were evaluated using two approaches. Gathering qualitative feedback through user testing sessions, in which participants interacted with the system and shared their impressions regarding the usefulness and relevance of the suggested resources. Additionally, predefined user scenarios were used to simulate a consistent interaction path, allowing for a more detailed examination of the system's recommendation behavior in different contexts.

Learning outcomes were not evaluated through formal empirical studies due to time and resource constraints. However, partial insight into learning effectiveness was obtained by tracking the completion of a learning path, which required users to complete associated assessments to progress through the modules. These assessments provided a basic measure of understanding, although their design was simple and lacked input from educational professionals, limiting their validity as formal evaluation instruments. Nonetheless, the system's architecture, built around modular content delivery and assessment-based progression, lays a foundation for future studies that could explore educational impact more rigorously. Overall, the applied evaluation methods offer an initial indication of system quality and effectiveness, particularly in terms of semantic retrieval, recommendation relevance, and alignment with the needs of learners in engineering education.

6.2 Main Challenges

The main challenge during the implementation of the solution was integrating and evolving several complex subsystems (a relational database, a semantic search engine, a graph database, modular

learning paths, and assessment logic) within a limited development period. The balance between the breadth of these components and the depth of their implementation required continual trade-offs between functionality and refinement.

Although a pre-existing UI template was used to speed up the design of the frontend, most of the platform's logic still had to be developed from the ground up. This included implementing user registration and authentication flows, API endpoints, state management, modular learning logic, and real-time progress tracking. The template mainly helped with layout and styling, but it didn't provide the logic needed to support the platform's functionality. Building and connecting everything added more work than expected.

Unexpected bugs appeared on a near-daily basis during development, particularly as new components were added or existing ones became interconnected. Debugging and fixing these issues often consumed considerable time and energy, diverting focus from more research-oriented components such as the recommendation engine and semantic search configuration. Addressing compatibility issues between services, fixing state inconsistencies, and debugging asynchronous behavior in the frontend were frequent and time-consuming tasks.

The design and implementation of the modular learning path structure also required careful thought. The logic for unlocking modules, tracking progress, and validating assessment completion had to be clearly defined and consistently maintained across the frontend, backend, and database layers. Designing this system in a way that was both flexible and reliable required careful coordination of user interactions, backend logic, and relational data integrity. This complexity increased the risk of introducing logical inconsistencies, especially when modifying the flow of learning path progression.

Despite these challenges, the project successfully delivered a working prototype that integrated several intelligent features and served as a proof of concept for a more comprehensive platform. The process also revealed valuable areas for further optimization, such as tighter integration between learning analytics and recommendations, improved debugging workflows, and more robust user modeling strategies.

6.3 Limitations and Threats to Validity

6.3.1 Limitations of the Evaluation Process

This subsection outlines the main limitations of the evaluation phase, especially regarding content availability, the number of participants, and the testing conditions. These factors influenced how far the results could be interpreted or generalized.

One of the biggest limitations was the small number and variety of resources available on the platform. Since all materials were added manually and came from a limited range of sources, the system lacked broader topic coverage. This affected the quality of semantic search results and reduced the range of possible recommendations. In some cases, especially when fallback logic

was used, the system couldn't return the expected number of recommended items due to a lack of diverse content.

The creation of learning paths, modules, and assessments was also affected by the lack of input from educational experts. As a result, the assessments remained relatively simple, which limited their ability to measure deeper levels of understanding. However, their inclusion still provided a useful way to track user progress and gather preliminary insights into how learners interacted with the learning content. The scarcity of available learning paths further limited the ability to evaluate how well the recommendation engine could adapt across different learning scenarios.

The evaluation itself was also limited by the short testing sessions and the small number of participants (n=10). Since all sessions were done locally on a single device, it wasn't possible to simulate a real-world learning environment or observe how users might interact with the system over time. This made it difficult to assess things like evolving user preferences, longer-term learning progress, or system performance in more realistic conditions.

Additionally, although the recommendation system offered basic personalization based on user interactions and graph relationships in Neo4j, more advanced techniques such as machine learning or collaborative filtering were not implemented due to time constraints and the complexity of integrating them into the existing tech stack. As a result, the system had limited ability to adapt to user behavior or improve its recommendations over time.

Finally, the platform was only tested in a prototype environment. No formal testing was done on performance aspects such as speed, scalability, or behavior under heavier usage. So, while the system worked during testing, its technical reliability in real-world conditions remains unknown.

Even with these limitations, the evaluation helped confirm that the system's main features (semantic search, personalized recommendations, and modular learning) functioned as intended. It also provided useful feedback and ideas for improving the platform in future iterations, especially in terms of expanding the content and refining recommendation logic.

6.3.2 System-Level Limitations

Beyond the evaluation itself, the knowledge management system developed faced some broader limitations that affected its overall scope and potential impact. These included technical constraints, design choices, and the lack of real-world deployment.

One major limitation was that the system was never tested in an actual classroom or used by students and educators over an extended period. This meant it wasn't possible to track how users might engage with the platform in the long term, how learning might progress over time, or how useful the recommendations would be in real educational settings.

From a technical point of view, although the platform includes semantic search and a graph-based recommendation engine, some features remain isolated. For example, assessment results were stored in the relational database but weren't connected to the graph. Because of that, the recommendation system couldn't take into account user performance or adjust suggestions based on assessment outcomes.

An ontology was also created early in the project using WebProtégé to model relationships between concepts. However, integrating it into the system turned out to be too complex due to limited documentation and lack of support in the Node.js environment. As a result, the idea was dropped, and a more practical solution using vector-based semantic search was implemented instead.

Finally, given the system's overall complexity and the limited development timeframe, the project focused on three core components: modular learning, semantic search, and graph-based recommendations. It was necessary to prioritize features that were both technically feasible and central to the system's objectives. This focus ensured that key functionalities could be properly designed, implemented, and evaluated. However, it also meant that other potential features, such as collaborative tools, adaptive feedback, and real-time learning analytics, were not explored in depth and are proposed as directions for future work.

6.3.3 Threats to Validity

When interpreting the results of this thesis, a few potential threats to validity should be kept in mind:

- **Internal Validity:** Because there were no pre- or post-tests and the assessments were simple, it's hard to say whether users actually learned something or were just able to guess correct answers.
- **External Validity:** All participants were students from the same degree program, and the system wasn't used in real courses, so it's unclear how well the results apply to other contexts or users.
- **Construct Validity:** Key ideas like "learning outcomes" or "recommendation quality" were mostly measured using user impressions and task completion, not standardized or validated instruments.
- **Conclusion Validity:** Since the project was exploratory and the sample size was small, the results should be seen as early indications rather than solid proof of the system's effectiveness.

These limitations don't take away from the contributions made, but they do define the boundaries of what can be concluded from this work.

Ultimately, the work developed in this thesis resulted in a functional prototype that combines intelligent retrieval, structured learning, and personalized recommendations. While several limitations remain, the system provides a practical foundation for future research and more advanced implementations in engineering education.

Chapter 7

Conclusions

This chapter summarizes the main contributions of the thesis and outlines directions for future research and development. Building on the findings and evaluation results discussed previously, it highlights what was achieved through the implemented prototype and what remains open for further exploration.

7.1 Main Contributions

The main contribution of this thesis is the development of a functional prototype of an intelligent knowledge management system tailored to the needs of engineering students and educators. The platform integrates intelligent capabilities, such as semantic search and personalized recommendations, alongside modular learning paths to support personalized and structured knowledge exploration. In doing so, it addresses core challenges identified in the literature and aligns with the four research questions guiding the study.

From a scientific perspective, the project demonstrates how combining transformer-based semantic retrieval with graph-based recommendations can improve the relevance and adaptability of educational content delivery. Unlike traditional platforms that rely on keyword-based queries, the implemented semantic search engine interprets user intent and contextual meaning, addressing the well-documented limitations of surface-level retrieval mechanisms.

The recommendation engine further responds to the identified lack of personalization and adaptability in many educational KMS by dynamically suggesting resources based on user interactions and preferences. The use of a Neo4j graph database enabled the modeling of meaningful relationships between users, resources, and learning paths, contributing to more personalized and context-aware suggestions.

Another important contribution lies in the practical implementation of modular learning paths with integrated progress tracking and assessment checkpoints. This design addresses the absence of structured, goal-oriented learning progression in many reviewed systems, providing users with clearer learning trajectories and opportunities for self-assessment.

In terms of usability, the platform was designed based on human-centered principles and informed by real student needs. This helped mitigate common barriers to adoption, such as rigid workflows, unintuitive navigation, and fragmented interfaces. The simplified user experience contributes to addressing the misalignment between KMS design and actual educational workflows noted in several studies.

This thesis also demonstrates that intelligent, personalized KMS solutions can be developed using open-source technologies and deployed without requiring extensive institutional infrastructure. This positions the system as a viable option for smaller educational institutions, independent learning contexts, or initiatives exploring the integration of intelligent tools in education.

7.2 Future Work

Several limitations identified throughout the development and evaluation process suggest opportunities for future research and platform improvement.

A key priority is to expand the content base, both in quantity and diversity. A larger and more thematically varied set of resources would improve the performance of semantic search and recommendation features. In parallel, integrating data-driven personalization techniques, such as collaborative filtering, user profiling, or reinforcement learning, could significantly enhance the adaptability and accuracy of recommendations over time.

From a pedagogical standpoint, future iterations should involve educational experts to improve the design and validity of assessments. Aligning learning modules with formal curriculum standards and incorporating more cognitively demanding tasks would strengthen the system's capacity to support deeper learning. Adaptive learning features, such as feedback based on quiz performance or progress-based content recommendations, remain essential future enhancements.

To further validate educational impact, a more rigorous evaluation design is proposed. This could include a controlled experimental setup with two groups: one using the developed platform and the other using more traditional methods or standard LMS platforms. Pre- and post-tests could be administered to both groups to measure differences in knowledge acquisition and engagement. While this was not feasible within the scope of the current study, such an approach would provide more robust evidence of the platform's effectiveness in real-world educational settings.

In terms of integration with existing workflows, future versions of the platform could explore interoperability with established LMSs such as Moodle. This could involve exporting learning paths, importing user progress, or embedding intelligent components (e.g., semantic search and recommendation engines) within traditional LMS environments. By doing so, the platform could enhance (not replace) existing educational infrastructures, offering intelligent features as modular, plug-and-play extensions to conventional systems.

Other suggested improvements include:

- Integrating assessment data into the graph structure to support performance-based recommendations.

- Exploring ontology-based reasoning as a complement to vector-based semantic search.
- Conducting long-term user studies in real educational settings to better understand engagement, retention, and system impact.
- Extending collaborative features to support content sharing, peer feedback, or educator-student interaction.

Although this prototype has a heavy focus on students, future work could further develop the educator interface to support learning analytics and classroom management. These additions would contribute to a more comprehensive and collaborative knowledge management environment.

This thesis sets the groundwork for intelligent educational systems that not only manage content but also actively support learning. With further development, the platform has the potential to evolve into a more robust, scalable, and impactful solution for engineering education and beyond.

References

[1] S. S. Abu Naser, M. J. Shobaki, and Y. M. Abu Amuna. “Promoting knowledge management components in the Palestinian Higher Education Institutions - A comparative study”. In: *International Letters of Social and Humanistic Sciences* (2016), pp. 42–53.

[2] Maryam Alavi and Dorothy E. Leidner. “Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues”. In: *MIS Quarterly* 25.1 (2001), pp. 107–136.

[3] John Brooke. “SUS - A quick and dirty usability scale”. In: *Usability Evaluation in Industry*. Ed. by Patrick W. Jordan et al. London, UK: Taylor Francis, 1996, pp. 189–194.

[4] Wendi R. Bukowitz and Ruth L. Williams. *The Knowledge Management Fieldbook*. London: Financial Times/Prentice Hall, 1999.

[5] Robin Burke. “Hybrid recommender systems: Survey and experiments”. In: *User modeling and user-adapted interaction* 12.4 (2002), pp. 331–370.

[6] John M. Carroll et al. “Knowledge management support for teachers”. In: *Educational Technology Research and Development* 51.4 (2003), pp. 42–64. ISSN: 1556-6501. DOI: 10.1007/BF02504543. URL: https://doi.org/10.1007/BF02504543.

[7] Cloudflare, Inc. *Cloudflare R2 | Zero Egress Fee Object Storage | Cloudflare*. Accessed: 2025-03. 2024. URL: https://www.cloudflare.com/en-gb/developer-platform/products/r2/.

[8] Kimiz Dalkir. *Knowledge Management in Theory and Practice*. 3rd. MIT Press, 2017.

[9] Thomas H. Davenport and Laurence Prusak. *Working Knowledge: How Organizations Manage What They Know*. Harvard Business Press, 1998.

[10] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[11] Dhairya et al. “Skillify: Enhanced Learning Management System Using Generative AI”. In: *2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT)*. Vol. 1. Aug. 2024, pp. 1527–1532. DOI: 10.1109/ICCPCT61902.2024.10672966. URL: https://ieeexplore.ieee.org/document/10672966 (visited on 11/01/2024).

[12] Natalia V. Dneprovskaya and Inessa V. Shevtsova. “The Knowledge Management System Development for Smart Education”. In: *2018 IEEE International Conference "Quality Management, Transport and Information Security, Information Technologies" (IT&QM&IS)*. Sept. 2018, pp. 602–606. DOI: 10.1109/ITMQIS.2018.8525129. URL: https://ieeexplore.ieee.org/document/8525129 (visited on 10/31/2024).

[13] Docker, Inc. *Docker – Empowering App Development for Developers*. Accessed: 2025-02. 2024. URL: https://www.docker.com.

[14] Mufaro Dzingirai, Tatenda Leeroy Murombedzi, and Anesu Ndoro. "What is Education 5.0". In: *Innovation Ecosystem in Higher Education Towards Sustainable Development*. Ed. by Mufaro Dzingirai, Tatenda Leeroy Murombedzi, and Anesu Ndoro. Innovation and Resource Management Strategies for Startups Development, 2024. Chap. 3, p. 22.

[15] EESemi. *Knowledge Management (KM): Knowing what you know and profiting from it*. Retrieved from https://www.eesemi.com/knowledge-management.html. 2005.

[16] Elastic. *Elasticsearch – Distributed, RESTful Search and Analytics Engine*. Accessed: 2025-04. 2024. URL: https://www.elastic.co/elasticsearch.

[17] Elastic.co. *Hybrid Search*. Available at: https://www.elastic.co/docs/solutions/search/hybrid-search. 2023.

[18] M. Max Evans, Kimiz Dalkir, and Catalin Bidian. "A Holistic View of the Knowledge Life Cycle: The Knowledge Management Cycle (KMC) Model". In: 2014. URL: https://api.semanticscholar.org/CorpusID:198943605.

[19] Jake Feinberg and John Kamau. *Practical BM25 – Part 2: The BM25 Algorithm and its Variables*. Elastic Blog. 2020. URL: https://www.elastic.co/blog/practical-bm25-part-2-the-bm25-algorithm-and-its-variables (visited on 06/30/2025).

[20] Gloria Milena Fernandez-Nieto et al. "Co-designing a knowledge management tool for educator communities of practice". In: *Proceedings of the 2024 ACM Designing Interactive Systems Conference*. DIS '24. New York, NY, USA: Association for Computing Machinery, July 2024, pp. 1970–1990. ISBN: 9798400705830. DOI: 10.1145/3643834.3660682. URL: https://dl.acm.org/doi/10.1145/3643834.3660682 (visited on 10/30/2024).

[21] Dhruv Galgotia and Nirupa Lakshmi. "Implementation of Knowledge Management with Artificial Intelligence in Higher Education". In: *2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA)*. ISSN: 2642-7354. Dec. 2021, pp. 832–836. DOI: 10.1109/ICCCA52192.2021.9666300. URL: https://ieeexplore.ieee.org/document/9666300 (visited on 11/13/2024).

[22] Jack Gammack et al. "Semantic knowledge management system for design documentation with heterogeneous data using machine learning". In: *Procedia CIRP*. 32nd CIRP Design Conference (CIRP Design 2022) - Design in a changing world 109 (Jan. 2022), pp. 95–100. ISSN: 2212-8271. DOI: 10.1016/j.procir.2022.05.220. URL: https://www.sciencedirect.com/science/article/pii/S2212827122006680 (visited on 11/13/2024).

[23] Zhomartkyzy Gulnaz and Balova Tatiana. "The implementation of a knowledge management system model". In: *2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT)*. Oct. 2014, pp. 1–5. DOI: 10.1109/ICAICT.2014.7035945. URL: https://ieeexplore.ieee.org/document/7035945 (visited on 11/01/2024).

[24] M. Harper. *What are the best four components of knowledge management?* Retrieved from https://www.apqc.org/blog/whatare-best-four-components-knowledge-management. Houston, TX, 2019.

[25] J. Hietala. *Knowledge management*. Retrieved from Valamis: https://www.valamis.com/hub/knowledge-management. 2019.

[26] Hanqing Hu et al. "Research on Intelligent Recommendation System Based on Knowledge Management". In: *2021 International Conference on Intelligent Computing, Automation and Applications (ICAA)*. June 2021, pp. 195–198. DOI: 10.1109/ICAA53760.2021.00041. URL: https://ieeexplore.ieee.org/document/9653492 (visited on 06/04/2025).

[27] J. Hudson et al. *Model for effective governance of knowledge management: A Case Study at the US Nuclear Regulatory Commission*. No. IAEA-CN–220. 2015.

[28] Imamah et al. "Development of Dynamic Personalized Learning Paths Based on Knowledge Preferences and the Ant Colony Algorithm". In: *IEEE Access* 12 (2024). Conference Name: IEEE Access, pp. 144193–144207. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3442312. URL: https://ieeexplore.ieee.org/document/10634179 (visited on 11/03/2024).

[29] Jared Hanson. *Passport.js – Simple, unobtrusive authentication for Node.js*. Accessed: 2025-02. 2024. URL: https://www.passportjs.org.

[30] Nyoman Karna, Iping Supriana, and Nur Maulidevi. "Recommendation system on knowledge management system via OAI-PMH". In: *2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*. Sept. 2017, pp. 1–5. DOI: 10.1109/EECSI.2017.8239154. URL: https://ieeexplore.ieee.org/document/8239154 (visited on 05/28/2025).

[31] Jill J. Kidwell, Karen M. Vander Linde, and Sandra L. Johnson. "Applying Corporate Knowledge Management Practices in Higher Education". In: *Educause Quarterly* 23.4 (2000), pp. 28–33.

[32] Yury Kravchenko, Elmar Kuliev, and Ilona Kursitys. "Information's semantic search, classification, structuring and integration objectives in the knowledge management context problems". In: *2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT)*. ISSN: 2472-8586. Oct. 2016, pp. 1–5. DOI: 10.1109/ICAICT.2016.7991671. URL: https://ieeexplore.ieee.org/document/7991671 (visited on 06/09/2025).

[33] Montserrat Leon and et al. "Artificial Intelligence and Personalized Learning Paths: Integrating Technology-Driven and Human-Driven Approaches in Education 5.0". In: *International Journal of Smart Innovation and Education* 14.1 (2024), pp. 13–19. URL: https://ideas.repec.org/a/vrs/ijsiel/v14y2024i1p13-19n1001.html.

[34] Jiajia Liu, Meilin Luan, and Ning Jiang. "Personalized Academic Communication Recommendation System Based on Knowledge Graph". In: *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*. May 2024, pp. 610–614. DOI: 10.1109/ICECAI62591.2024.10674837. URL: https://ieeexplore.ieee.org/document/10674837 (visited on 05/28/2025).

[35] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. "Content-based recommender systems: State of the art and trends". In: *Recommender Systems Handbook*. Springer, 2011, pp. 73–105.

[36] Pingping Lu and Xinyi Feng. "Personalized Recommendation Algorithm in AI-Assisted Learning System". In: *2024 6th International Conference on Communications, Information System and Computer Engineering (CISCE)*. ISSN: 2833-2423. May 2024, pp. 1289–1294. DOI: 10.1109/CISCE62493.2024.10653402. URL: https://ieeexplore.ieee.org/document/10653402 (visited on 11/01/2024).

[37] Thanet Markchom, Huizhi Liang, and James Ferryman. "Review of Explainable Graph-Based Recommender Systems". In: *ACM Computing Surveys* (2023).

[38] Mark W. McElroy. *The New Knowledge Management: Complexity, Learning, and Sustainable Innovation*. Boston: Butterworth-Heinemann, 2003.

[39] Meta. *React – A JavaScript library for building user interfaces*. Accessed: 2025-02. 2024. URL: https://reactjs.org.

[40] Tomás Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. 2013. URL: http://arxiv.org/abs/1301.3781.

[41] Neo4j, Inc. *Neo4j – The World's Leading Graph Database*. Accessed: 2025-03. 2024. URL: https://neo4j.com.

[42] Ngoc Thanh Nguyen and Edward Szczerbicki. *Intelligent Systems for Knowledge Management*. Springer, 2009. ISBN: 978-3-642-04169-3.

[43] Ngoc Thanh Nguyen and Edward Szczerbicki. *Smart Information and Knowledge Management: Advances, Challenges, and Critical Issues*. Springer, 2010. ISBN: 978-3642045837.

[44] Ikujiro Nonaka and Hirotaka Takeuchi. *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, 1995.

[45] OpenJS Foundation. *Express.js – Fast, Unopinionated, Minimalist Web Framework for Node.js*. Accessed: 2025-02. 2024. URL: https://expressjs.com.

[46] OpenJS Foundation. *Node.js – JavaScript Runtime Built on Chrome's V8 Engine*. Accessed: 2025-02. 2024. URL: https://nodejs.org.

[47] Jeffrey Pennington, Richard Socher, and Christopher Manning. "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162/.

[48] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics. 2019, pp. 3982–3992. URL: https://aclanthology.org/D19-1410/.

[49] Francesco Ricci, Lior Rokach, and Bracha Shapira. "Introduction to recommender systems handbook". In: *Recommender Systems Handbook*. Springer, 2011, pp. 1–35.

[50] Madiha Shah. "Impact of Management Information Systems (MIS) on School Administration: What the Literature Says". In: *Procedia - Social and Behavioral Sciences* 116 (2014), pp. 2799–2804. DOI: 10.1016/j.sbspro.2014.01.659.

[51] Monika Sharma et al. “Methodology for the Development of an Ontology based E-Learning Platform”. In: *2020 International Conference on Computation, Automation and Knowledge Management (ICCAKM)*. Jan. 2020, pp. 101–106. DOI: 10.1109/ICCAKM46823.2020.9051540. URL: https://ieeexplore.ieee.org/document/9051540 (visited on 11/04/2024).

[52] T. Sheeba and Reshmy Krishnan. “Semantic Predictive Model of Student Dynamic Profile Using Fuzzy Concept”. In: *Procedia Computer Science*. International Conference on Computational Intelligence and Data Science 132 (Jan. 2018), pp. 1592–1601. ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.05.124. URL: https://www.sciencedirect.com/science/article/pii/S1877050918308561 (visited on 11/04/2024).

[53] Hassan Shuaibu, Jigish Zaveri, and Aminu Hamza. “An Integrated View of Knowledge Management Enablers, Components, and Benefits: Comprehensive Literature Review”. In: *Journal of International Technology and Information Management* 30 (Jan. 2021), pp. 1–23. DOI: 10.58729/1941-6679.1520.

[54] Manjit Singh Sidhu and Lee Chen Kang. “Emerging Trends and Technologies for Enhancing Engineering Education: An Overview”. In: *International Journal of Information and Communication Technology Education (IJICTE)* 6.4 (2010), pp. 1–11. DOI: 10.4018/jicte.2010100104.

[55] John (example) Smith. “Educational Knowledge Management: A Case Study of a University KMS Implementation”. In: *Computers & Education* XX.X (2013), pp. XXX–XXX.

[56] V. Sugumaran. “Chapter 14 - Semantic technologies for enhancing knowledge management systems”. In: *Successes and Failures of Knowledge Management*. Ed. by Jay Liebowitz. Boston: Morgan Kaufmann, Jan. 2016, pp. 203–213. ISBN: 978-0-12-805187-0. DOI: 10.1016/B978-0-12-805187-0.00014-0. URL: https://www.sciencedirect.com/science/article/pii/B9780128051870000140 (visited on 11/07/2024).

[57] The PostgreSQL Global Development Group. *PostgreSQL – The World’s Most Advanced Open Source Relational Database*. Accessed: 2025-02. 2024. URL: https://www.postgresql.org.

[58] Naveen Thimmanna, Sandhya Patil, and Sanjay Kulkarni. “Personalized Learning Paths: Adapting Education with AI-Driven Curriculum”. In: *European Economics Letters* 14.1 (2024). ISSN: 2323-5233, pp. 13–19. DOI: 10.52783/eel.v14i1.993. URL: https://www.eelet.org.uk/index.php/journal/article/view/993.

[59] Carine Edith Touré, Christine Michel, and Jean-Charles Marty. “Re-Designing Knowledge Management Systems - Towards User-Centred Design Methods Integrating Information Architecture”. In: *ArXiv* abs/1601.08032 (2014). URL: https://api.semanticscholar.org/CorpusID:215827042.

[60] Tania Tudorache et al. “WebProtégé: A Collaborative Ontology Editor and Knowledge Acquisition Tool for the Web”. In: *Semantic Web* 11.1 (2011), pp. 1–13. DOI: 10.5555/2595053.2595061.

[61] UNESCO Institute for Educational Planning. *Educational Management Information System (EMIS)*. URL: https://learningportal.iiep.unesco.org/en/glossary/educational-management-information-system-emis.

[62] Rita Vieira et al. “Society 5.0 and Education 5.0 : A Critical Reflection”. In: *2023 18th Iberian Conference on Information Systems and Technologies (CISTI)*. ISSN: 2166-0727. June 2023, pp. 1–6. DOI: 10.23919/CISTI58278.2023.10211386. URL: https://ieeexplore.ieee.org/document/10211386 (visited on 12/12/2024).

[63] Piyush Vyas. “Knowledge management and higher education institute: Review & topic analysis”. In: *Journal of Open Innovation: Technology, Market, and Complexity* 10.3 (Sept. 2024), p. 100349. ISSN: 2199-8531. DOI: 10.1016/j.joitmc.2024.100349. URL: https://www.sciencedirect.com/science/article/pii/S2199853124001434 (visited on 10/30/2024).

[64] Haolin Wen et al. “AI-KM: A distributed multi-view and intelligent knowledge management software”. In: *SoftwareX* 27 (Sept. 2024), p. 101840. ISSN: 2352-7110. DOI: 10.1016/j.softx.2024.101840. URL: https://www.sciencedirect.com/science/article/pii/S2352711024002115 (visited on 10/30/2024).

[65] D. Williams. *Components of a knowledge management system*. Retrieved from RealKM Cooperative Limited: https://realkm.com/2016/02/03/components-of-aknowledge-management-system/. 2016.

[66] Xenova. *all-MiniLM-L6-v2: Sentence-Transformer Model on Hugging Face*. https://huggingface.co/Xenova/all-MiniLM-L6-v2. Accessed: 2025-06-30. 2024.

[67] Haoran Xie et al. “Trends and development in technology-enhanced adaptive/personalized learning: A systematic review of journal publications from 2007 to 2017”. In: *Computers Education* 140 (2019), p. 103599. DOI: 10.1016/j.compedu.2019.103599. URL: https://www.sciencedirect.com/science/article/pii/S0360131519301526.

[68] Samuel T. Yigzaw, Ilkka Jormanainen, and Markku Tukiainen. “Trends in the Role of ICT in Higher Education Knowledge Management Systems: A Systematic Literature Review”. In: *Proceedings of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality*. TEEM’19. New York, NY, USA: Association for Computing Machinery, Oct. 2019, pp. 473–480. ISBN: 978-1-4503-7191-9. DOI: 10.1145/3362789.3362805. URL: https://doi.org/10.1145/3362789.3362805 (visited on 10/30/2024).

[69] Yulistia, Ermatita, and Reza Firsandaya Malik. “Knowledge Transfer Model for Private Higher Education Knowledge Management System”. In: *2019 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS)*. Oct. 2019, pp. 191–194. DOI: 10.1109/ICIMCIS48181.2019.8985229. URL: https://ieeexplore.ieee.org/document/8985229 (visited on 10/30/2024).

[70] Meng Zhang, Chuan Wu, and Guofu Ye. “Networked Intelligent Retrieval System Based on Semantic”. In: *Information Computing and Applications*. Springer, 2013, pp. 194–203. DOI: 10.1007/978-3-642-53756-6_23.

[71] S. Zyngier. “Knowledge management governance”. In: *Knowledge management: concepts, methodologies, tools, and applications*. IGI Global, 2008, pp. 2276–2284.

Appendix A

Entity Relationship Diagram

This appendix displays the entity relationship diagram that represents the structure of the PostgreSQL database of the platform. It illustrates the main entities, their attributes, and the relationships between them, serving as the foundation for managing users, resources, modules, learning paths, assessments, and user interactions. The diagram supports the understanding of the system's data model and its alignment with the implemented functionalities.

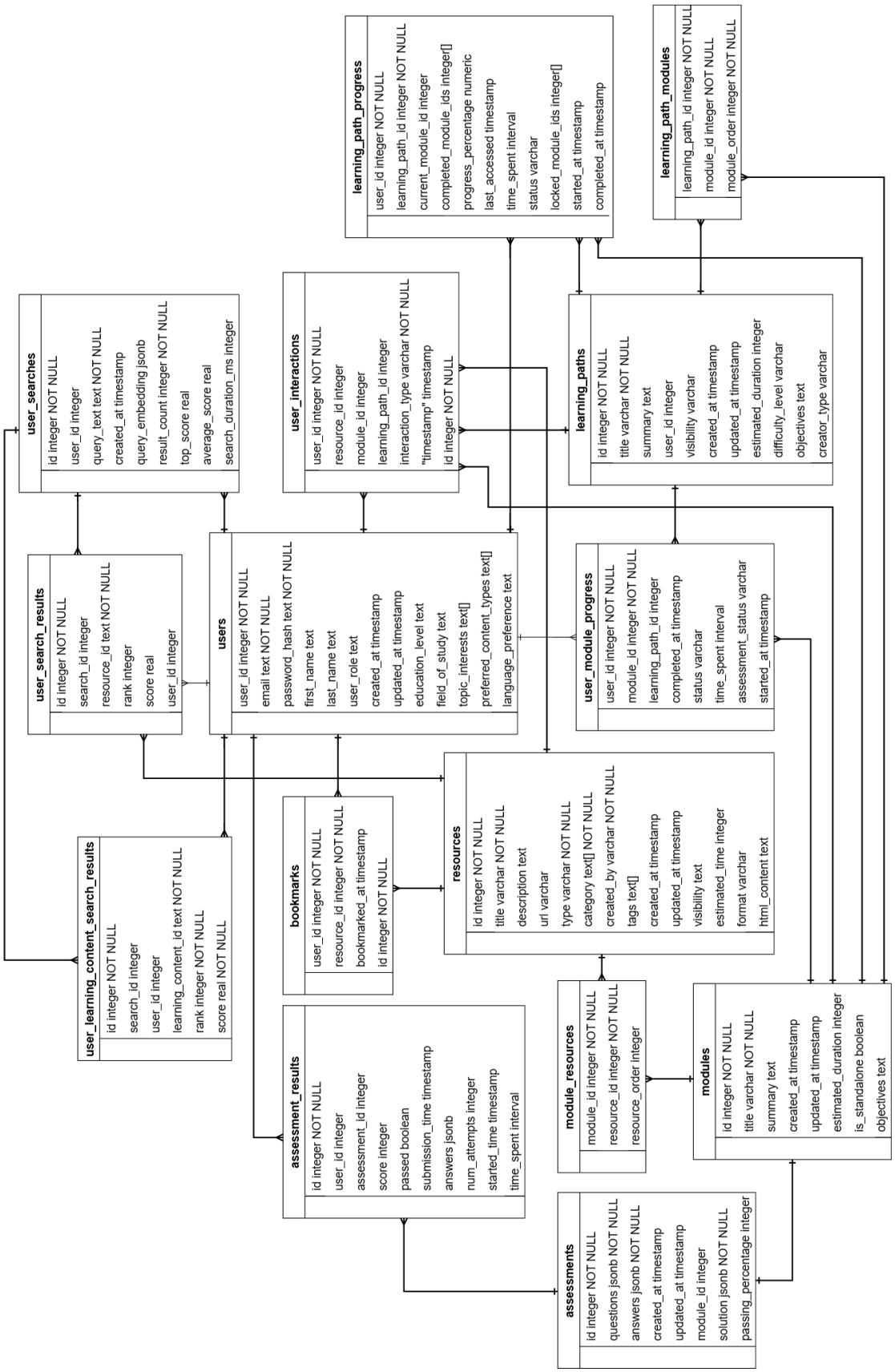

Figure A.1: Entity-relationship diagram of the PostgreSQL database.

Appendix B

PostgreSQL to Neo4j Synchronization Script

This appendix includes the Node.js script used to synchronize structured data between the relational PostgreSQL database and the Neo4j graph database. The script handles users, resources, modules, learning paths, and interactions and generates weighted relationships for the recommendation engine.

Listing B.1: Node.js Script for Syncing PostgreSQL to Neo4j

```
require("dotenv").config();

const { pool } = require("./postgres");
const { driver } = require("./neo4j");

const syncData = async () => {
  const pgClient = await pool.connect();
  const neoSession = driver.session();

  try {
    console.log("Starting data synchronization...");
    await neoSession.run(`MATCH (n) DETACH DELETE n`);
    console.log("Cleared existing Neo4j data.");

    // Sync Users
    const users = await pgClient.query(
      "SELECT user_id, first_name, last_name, user_role, education_level,
          field_of_study, topic_interests, preferred_content_types,
          language_preference FROM users"
    );
    for (let user of users.rows) {
      await neoSession.run(
        `MERGE (u:User {id: $id})
```

```
22        SET u.first_name = $first_name, u.last_name = $last_name,
23        u.user_role = $user_role,
24        u.education_level = $education_level,
25        u.field_of_study = $field_of_study,
26        u.topic_interests = $topic_interests,
27        u.preferred_content_types = $preferred_content_types,
28        u.language_preference = $language_preference
29        `,
30        {
31          id: user.user_id.toString(),
32          first_name: user.first_name,
33          last_name: user.last_name,
34          user_role: user.user_role,
35          education_level: user.education_level,
36          field_of_study: user.field_of_study,
37          topic_interests: user.topic_interests,
38          preferred_content_types: user.preferred_content_types,
39          language_preference: user.language_preference,
40        }
41      );
42    }
43
44    // Sync Resources
45    const resources = await pgClient.query(
46      "SELECT id, title, description, category, tags, type FROM resources"
47    );
48
49    for (let resource of resources.rows) {
50      const category = Array.isArray(resource.category)
51        ? resource.category
52        : [];
53      const tags = Array.isArray(resource.tags) ? resource.tags : [];
54
55      // Merge Resource node
56      await neoSession.run(
57        `MERGE (r:Resource {id: $id})
58         SET r.title = $title, r.description = $description, r.type = $type, r.
               category = $category, r.tags = $tags`,
59        {
60          id: resource.id.toString(),
61          title: resource.title,
62          description: resource.description,
63          type: resource.type,
64          category: resource.category,
65          tags: resource.tags,
66        }
67      );
68
69      // Sync Categories
```

```
70      for (let cat of category) {
71        cat = cat.trim();
72        if (cat) {
73          await neoSession.run(
74            `
75            MATCH (r:Resource {id: $id})
76            MERGE (c:Category {name: $cat})
77            MERGE (r)-[:HAS_CATEGORY]->(c)
78            `,
79            { id: resource.id.toString(), cat }
80          );
81        }
82      }
83      for (let tag of tags) {
84        tag = tag.trim();
85        if (tag) {
86          await neoSession.run(
87            `
88            MATCH (r:Resource {id: $id})
89            MERGE (t:Tag {name: $tag})
90            MERGE (r)-[:HAS_TAG]->(t)
91            `,
92            { id: resource.id.toString(), tag }
93          );
94        }
95      }
96
97      // Sync Resource Type
98      if (resource.type) {
99        await neoSession.run(
100         `
101         MATCH (r:Resource {id: $id})
102         MERGE (rt:ResourceType {name: $type})
103         MERGE (r)-[:OF_TYPE]->(rt)
104          `,
105         { id: resource.id.toString(), type: resource.type }
106       );
107     }
108   }
109
110   (...)
111
112   // Sync User Interactions using Interaction nodes
113   const userInteractions = await pgClient.query(
114     "SELECT user_id, resource_id, module_id, learning_path_id, interaction_type,
          timestamp FROM user_interactions"
115   );
116
117   for (let interaction of userInteractions.rows) {
```

```
    if (interaction.user_id && interaction.interaction_type) {
      const timestamp = interaction.timestamp
        ? interaction.timestamp.toISOString()
        : new Date().toISOString();

      const weight = calculateInteractionWeight(
        interaction.interaction_type,
        interaction.timestamp
      );

      const interactionId = `${interaction.user_id}_${interaction.
          interaction_type}_${timestamp}`;

      // Create Interaction node
      await neoSession.run(
        `
    MERGE (i:Interaction {id: $interactionId})
    SET i.type = $type, i.timestamp = datetime($timestamp), i.weight = $weight
    `,
          {
            interactionId,
            type: interaction.interaction_type.toUpperCase(),
            timestamp,
            weight,
          }
      );

      // Create PERFORMED relationship
      await neoSession.run(
        `
    MATCH (u:User {id: $userId}), (i:Interaction {id: $interactionId})
    MERGE (u)-[:PERFORMED]->(i)
    `,
          {
            userId: interaction.user_id.toString(),
            interactionId,
          }
      );

      // Create TARGET relationship to Resource / Module / LearningPath
      if (interaction.resource_id) {
        await neoSession.run(
          `
      MATCH (i:Interaction {id: $interactionId}), (r:Resource {id: $targetId})
      MERGE (i)-[:TARGET]->(r)
      `,
            {
              interactionId,
              targetId: interaction.resource_id.toString(),
```

```
            }
          );
        } else if (interaction.module_id) {
          await neoSession.run(
            `
        MATCH (i:Interaction {id: $interactionId}), (m:Module {id: $targetId})
        MERGE (i)-[:TARGET]->(m)
        `,
            {
              interactionId,
              targetId: interaction.module_id.toString(),
            }
          );
        } else if (interaction.learning_path_id) {
          await neoSession.run(
            `
        MATCH (i:Interaction {id: $interactionId}), (lp:LearningPath {id: $targetId
            })
        MERGE (i)-[:TARGET]->(lp)
        `,
            {
              interactionId,
              targetId: interaction.learning_path_id.toString(),
            }
          );
        } else {
          console.log(
            `Skipping TARGET relationship due to missing target: ${JSON.stringify(
              interaction
            )}`
          );
        }
      } else {
        console.log(
          `Skipping interaction due to missing user_id or interaction_type: ${JSON.
              stringify(
            interaction
          )}`
        );
      }
    }

    console.log("Data synchronization completed successfully!");
  } catch (error) {
    console.error("Error syncing data:", error);
  } finally {
    pgClient.release();
    await neoSession.close();
  }
```

```
};

function calculateInteractionWeight(interactionType, timestamp) {
  let baseWeight = 0;

  switch (interactionType) {
    case "viewed_resource":
    case "viewed_module":
    case "viewed_learning_path":
      baseWeight = 2;
      break;
    case "started_module":
    case "started_learning_path":
      baseWeight = 3;
      break;
    case "bookmarked":
      baseWeight = 4;
      break;
    case "completed_module":
      baseWeight = 5;
      break;
    case "completed_learning_path":
      baseWeight = 6;
      break;
    default:
      baseWeight = 1;
  }

  const now = new Date();
  const interactionDate = new Date(timestamp);
  const timeDiff = now - interactionDate;
  const daysDiff = timeDiff / (1000 * 3600 * 24);

  const recencyFactor = Math.max(1, 10 - daysDiff);

  return baseWeight * recencyFactor;
}
module.exports = { syncData };
```

Listing B.1 shows the partial implementation, including the interaction weight formula described in Chapter 4.

Appendix C

User Dashboard Section Screenshots

This appendix presents the student user dashboard interface. The dashboard functions as a personalized homepage, offering quick access to key features such as recently viewed resources, personalized recommendations, learning progress, and bookmarked content. It is designed to support user engagement and self-regulated learning by highlighting current activity and suggesting relevant next steps.

The following figures follow the sequential flow of the user dashboard page, from top to bottom, as it is displayed to the user during typical interaction.

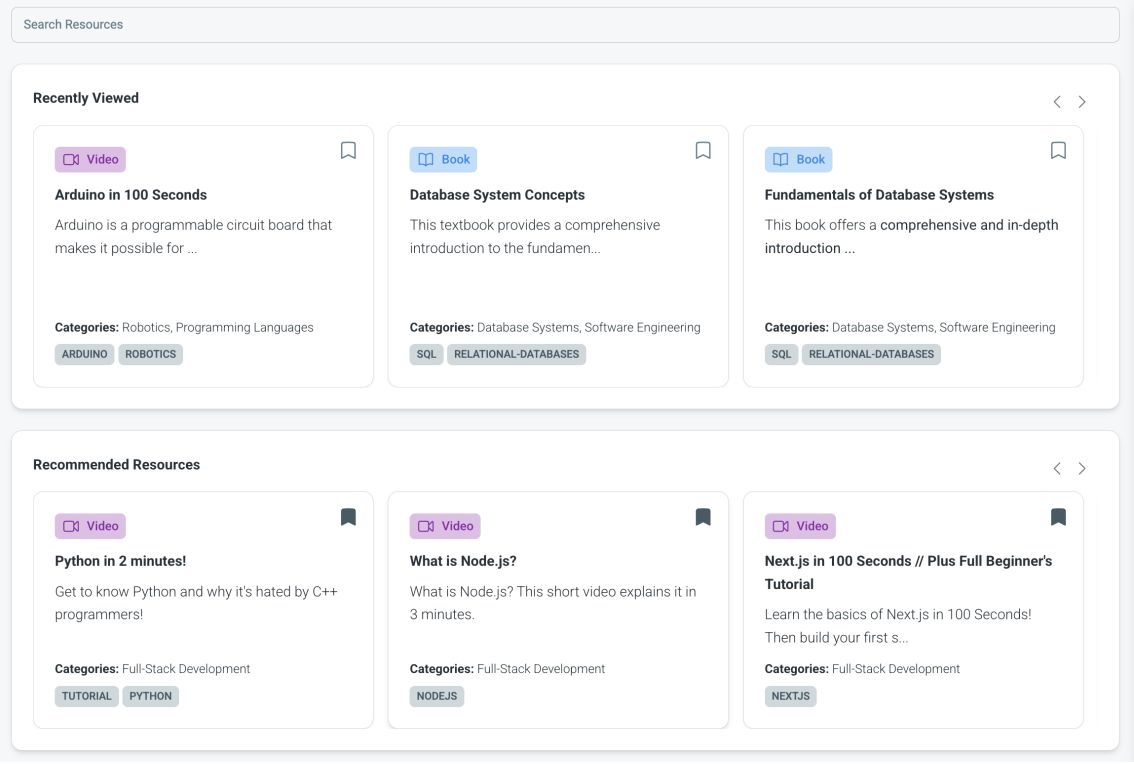

Figure C.1: User dashboard interface displaying the *Recently Viewed* and *Recommended Resources* sections.

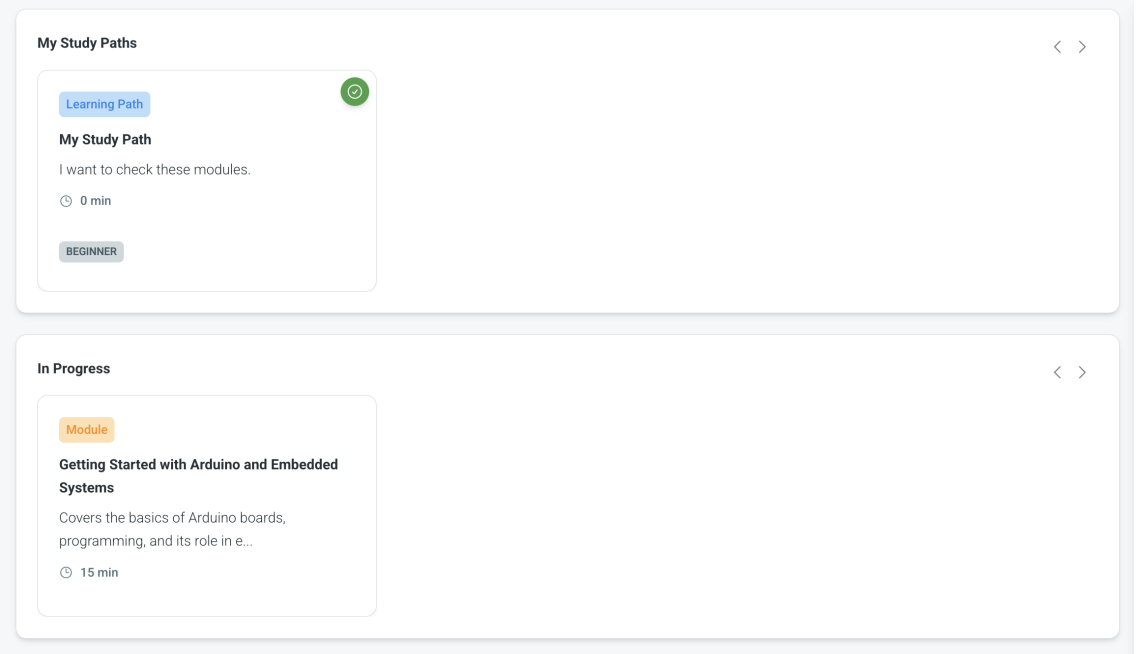

Figure C.2: User dashboard interface displaying the *My Study Paths* and *In Progress* sections.

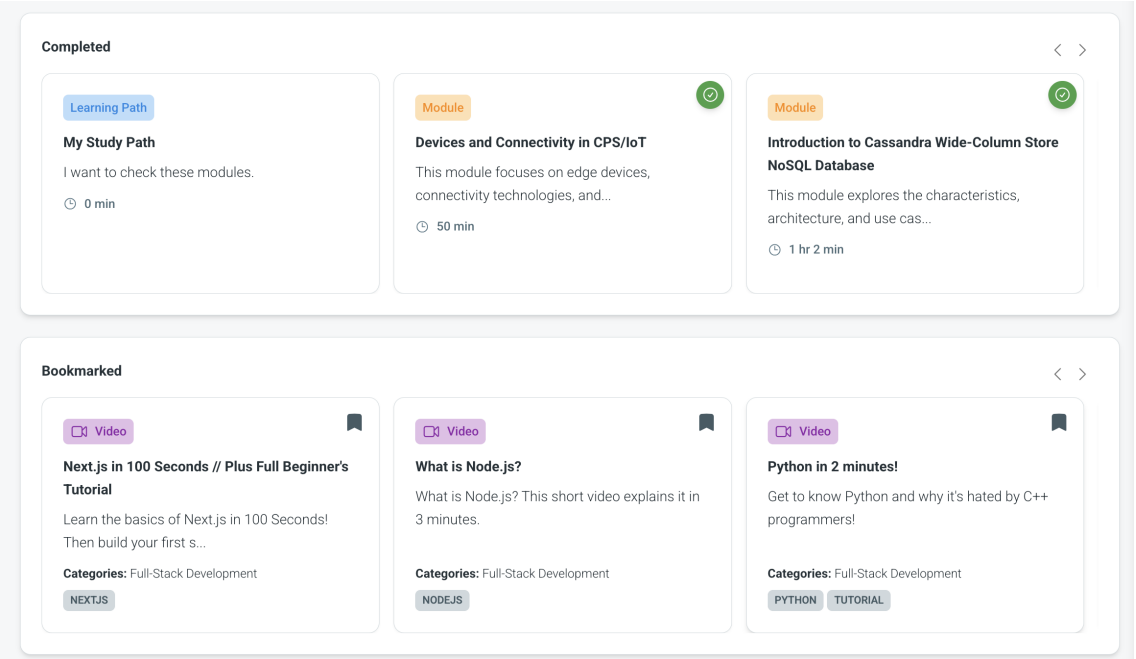

Figure C.3: User dashboard interface displaying the *Completed* and *Bookmarked* sections.

Appendix D

Feedback Form

This appendix presents the feedback form used during the evaluation phase of the platform. The form was designed to collect both quantitative and qualitative feedback from participants regarding their experience using the system. It includes sections for basic participant information, overall user experience, perceived usefulness, and suggestions for improvement. The figures follow the sequential layout of the form as presented to users.

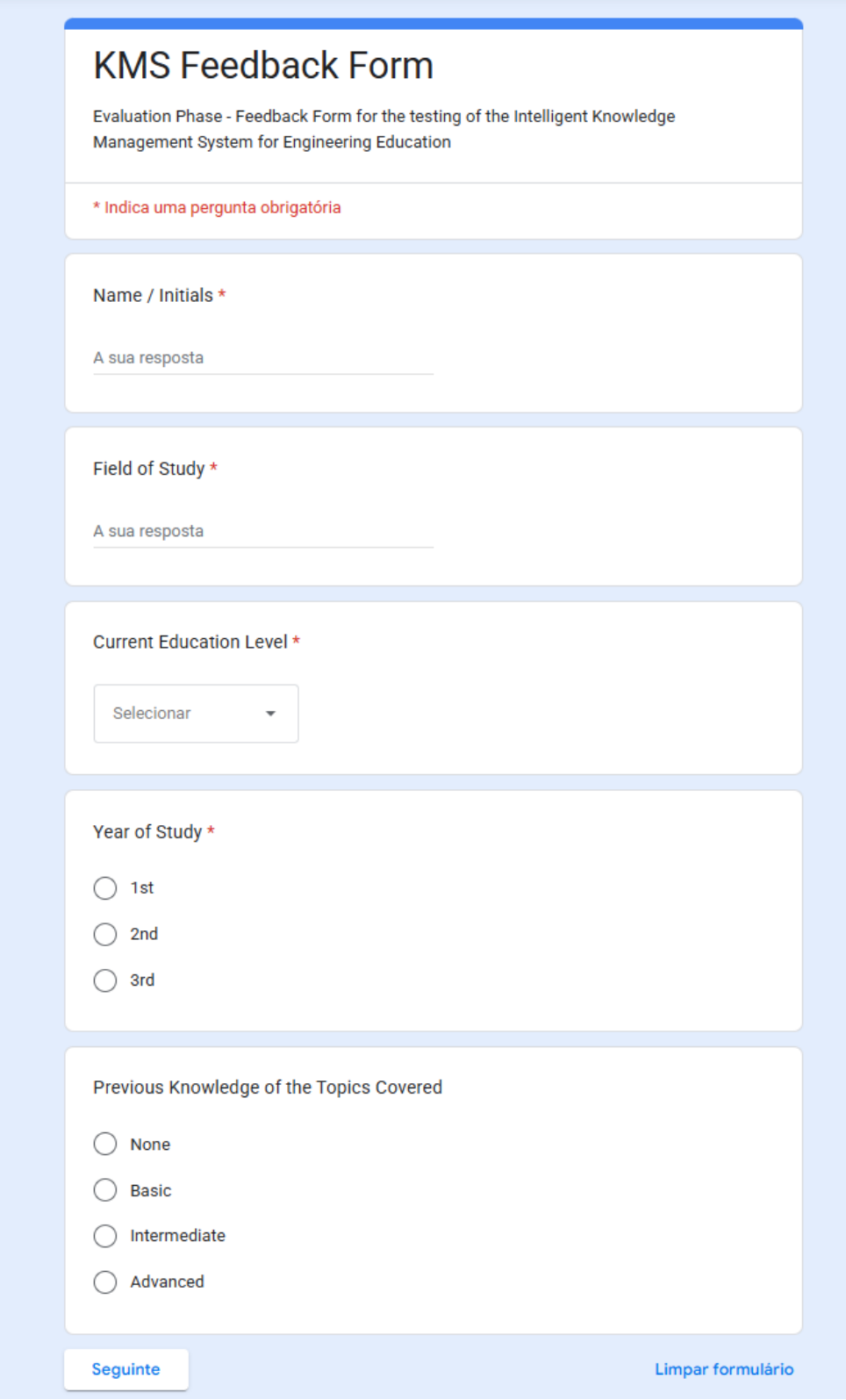

Figure D.1: Feedback form displaying inputs for the participant information.

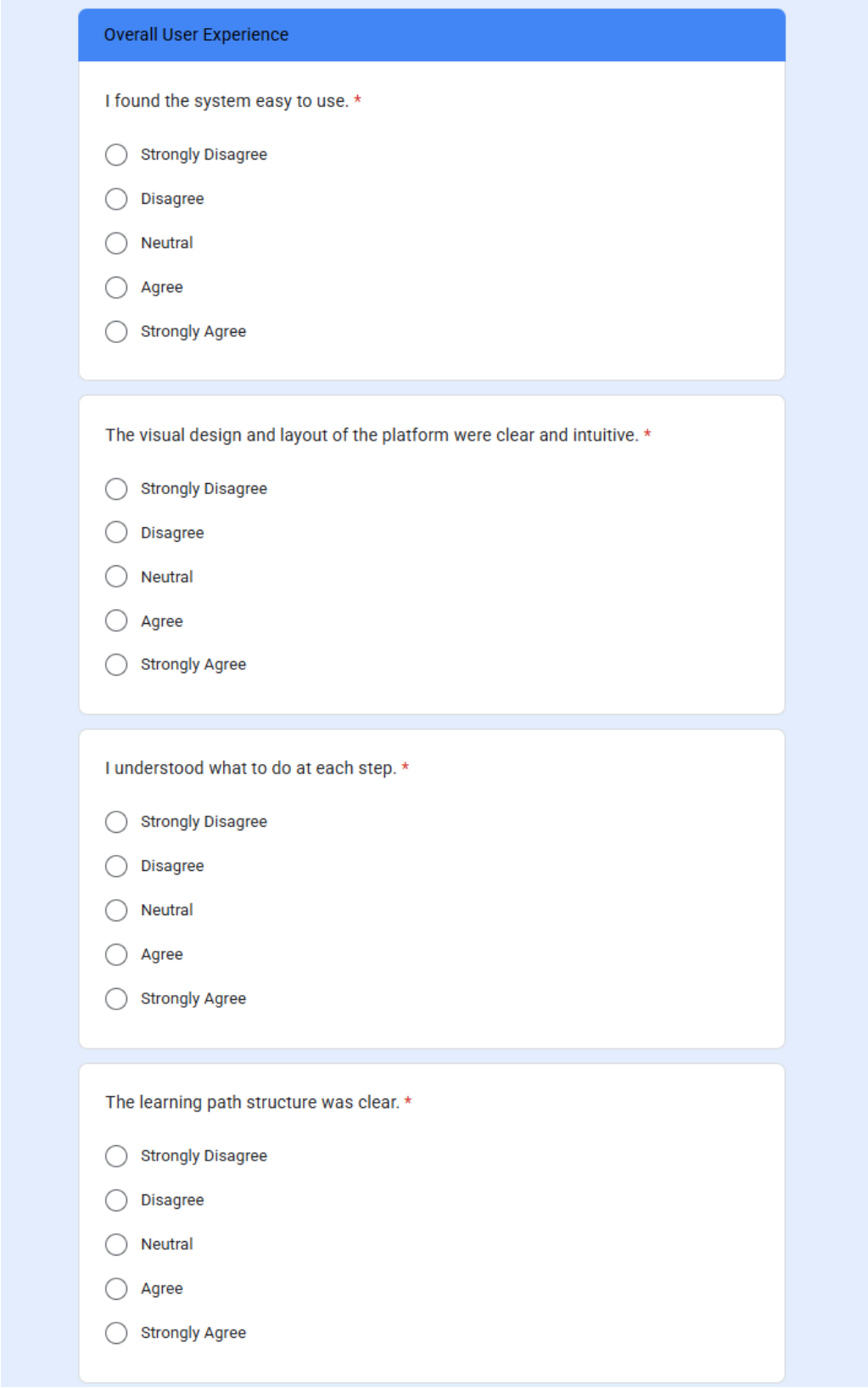

Figure D.2: Feedback form questions about the overall user experience of the platform (Part 1).

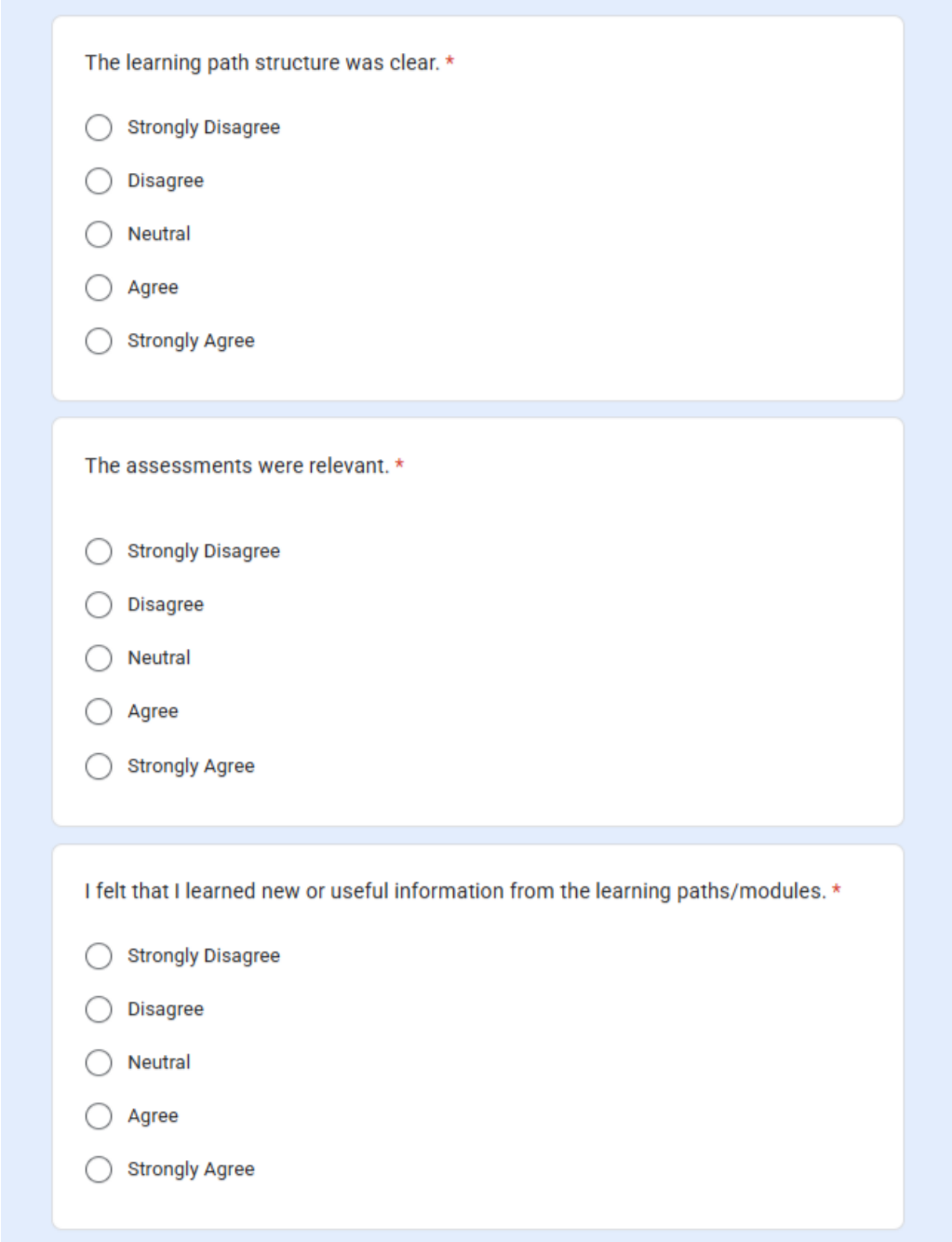

Figure D.3: Feedback form questions about the overall user experience of the platform (Part 2).

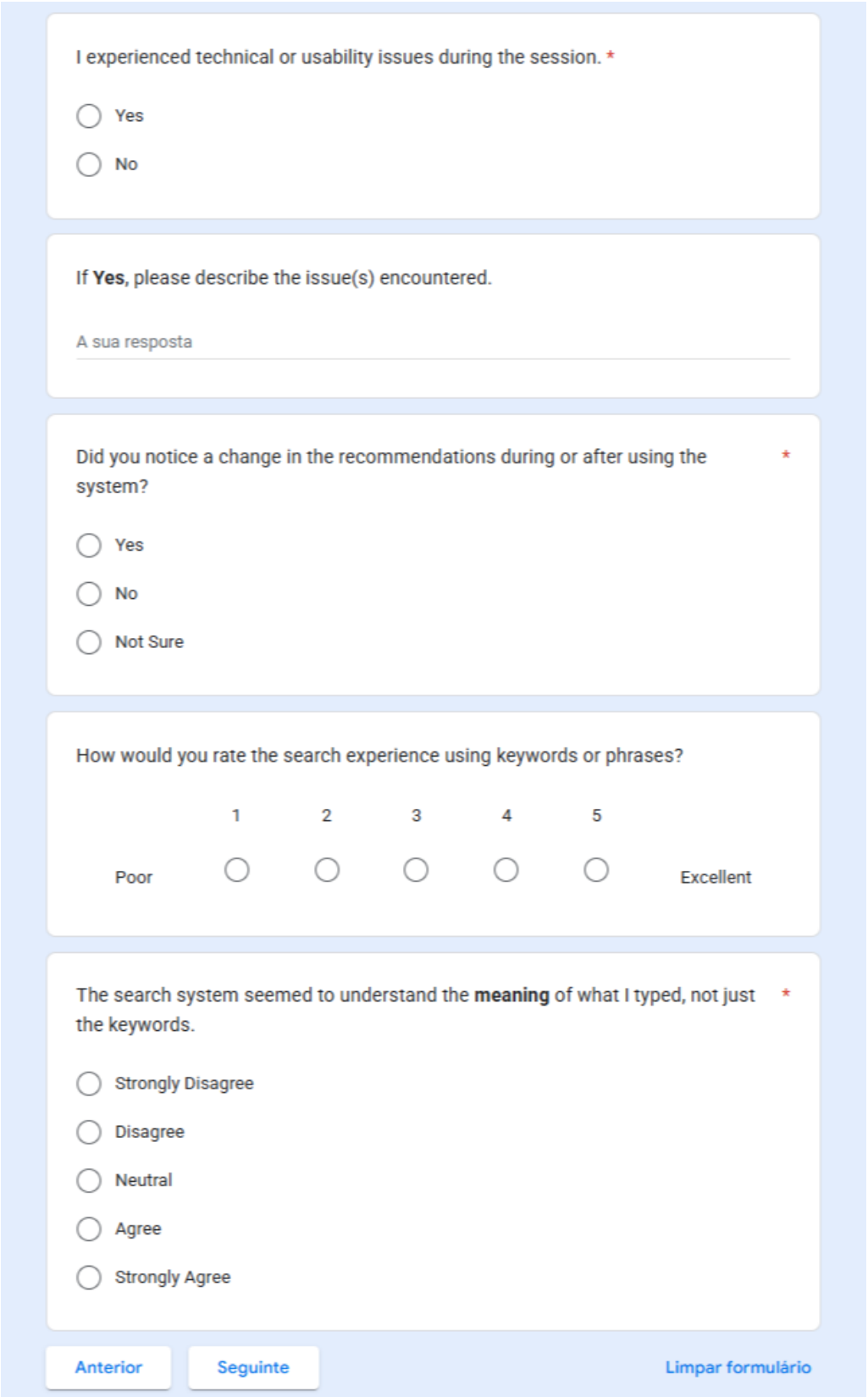

Figure D.4: Feedback form questions about the overall user experience of the platform (Part 3).

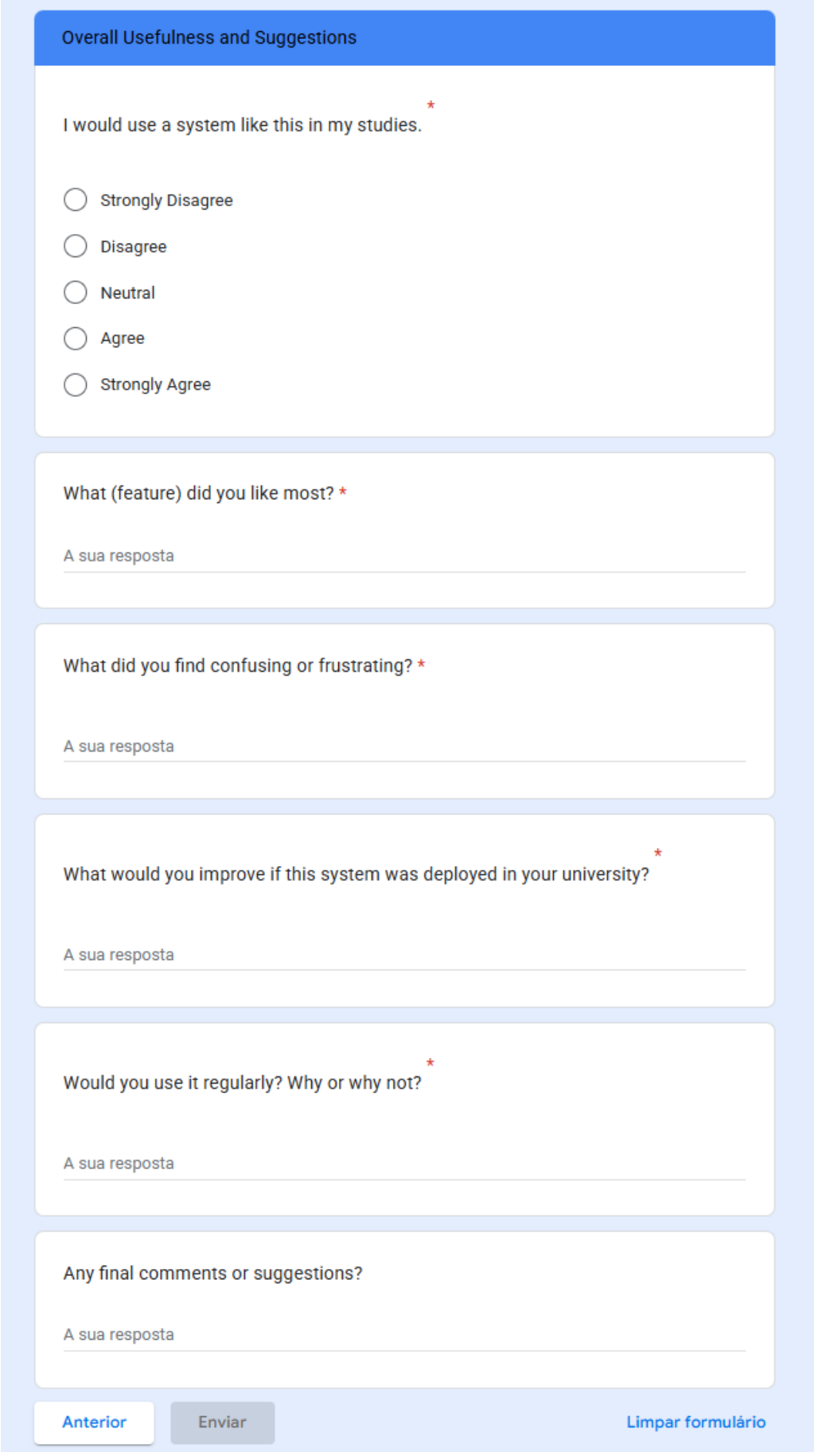

Figure D.5: Feedback form questions about the overall usefulness and suggestions of the platform.