THE UNIVERSITY OF ZAMBIA

SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTING AND INFORMATICS

FINAL YEAR PROJECT REPORT

ACADEMIC YEAR 2026

PERSONALIZED STUDY ASSISTANT AND SMART LEARNING RECOMMENDATIONS

An Individual Subsystem of the Intelligent Digital Library with Personalized Study Assistance

A report submitted to the Department of Computing and Informatics, The University of Zambia, in partial fulfilment of the requirements of the degree of Bachelor of Science in Computer Science (Systems and Electronics Engineering)

By

Name: LUNDA SIMUWELU
Computer Number: 2021399125

Supervisor:

Dr Lighton Phiri

October 23, 2026

Declaration

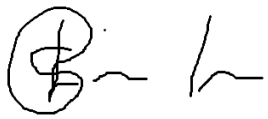
I declare that this final year project report is my own work and has been prepared in accordance with the academic requirements of the University of Zambia. All sources of information, ideas, data, software, images, and other material that are not my own have been appropriately acknowledged and referenced.

Where this project was undertaken as part of a group, I have clearly identified my individual contributions and have not presented the work of other group members as my own.

Student Name: **:** LUNDA SIMUWELU

Computer Number: 2021399125

Signature:



Date: October 23, 2026

Approval

This final year project report has been submitted for examination with the approval of the project supervisor.

Supervisor Name: Dr Lighton Phiri

Signature: _____

Date: October 23, 2026

Departmental/Examiner Approval (if applicable): _____

Abstract

Digital libraries give students access to books, lecture notes and research material, but they are designed mainly for storage and retrieval. They offer little help in understanding concepts, seeing how concepts depend on one another, or deciding what to study next. This project designed, implemented and tested a prototype Intelligent Digital Library with Personalized Study Assistant and Smart Learning Recommendations to address this gap.

The system represents each topic as a PCSMR record: the Problem a concept addresses, the Challenge that made the problem difficult, the Solution proposed, the Mechanism by which it works, and the Result it achieves. A language model drafts these records from uploaded PDF lecture material, and the learner reviews and edits each draft before relying on it. Concepts are linked by prerequisite relationships, with cycle prevention and a prerequisite-first learning path. Each concept can be viewed as notes, a flow diagram and an interactive concept map, and generated multiple-choice quizzes update a per-learner mastery record. The prototype was built with React, FastAPI and PostgreSQL. All AI access is isolated behind a replaceable provider interface, so the system keeps working when the AI provider changes.

Four automated backend tests covering nine test cases were recorded as passing, and a live extraction check on a sample passage produced a coherent draft in which only some fields could be traced to the source text. These tests show that the workflows behave as designed. They do not measure extraction accuracy, quiz quality, usability or learning gain, so the report defines an evaluation method for each. Feedback on the prototype identified the need for adaptive, multi-mode assessment with remediation, fuller study notes and an improved interface, and these are specified as recommendations. The study shows that structured, reviewable concept records can serve as a single foundation for several learning presentations and for prerequisite-based study guidance.

Keywords: digital library; personalized learning; knowledge representation; prerequisite graph; concept map; large language model; mastery tracking

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Lighton Phiri, for his guidance, constructive criticism and patience throughout this project. The feedback I received after the rehearsal presentation sharpened the direction of the work, particularly on adaptive assessment, question types and the structure of study notes.

I thank the lecturers and staff of the Department of Computing and Informatics, School of Natural and Applied Sciences, University of Zambia, for the knowledge and support that prepared me for this work.

I am grateful to my group members, A. F. Chipeta and J. M. Tambwisha, with whom the original proposal was prepared, and to my classmates for their discussions, shared experience and feedback.

I also acknowledge the developers of the open-source tools on which this prototype is built, including Python, FastAPI, PostgreSQL, React and React Flow.

Finally, I thank my family and friends for their encouragement, patience and support.

Intelligent Digital Library with Personalized Study Assistant and Smart Learning Recommendations

Final Year Project Report

Table of Contents

CHAPTER ONE 5

INTRODUCTION 5

1.1 Background 5

1.2 Problem Statement 6

1.3 Aim of the Study 6

1.4 Specific Objectives 7

1.5 Significance of the Study 7

1.6 Scope and Limitations 8

1.7 Organisation of the Report 9

CHAPTER TWO 9

LITERATURE REVIEW 9

2.1 Introduction 9

2.2 Digital Libraries and Digital Learning Environments 10

2.2.1 Digital Libraries 10

2.2.2 Digital Learning Environments 11

2.3 Challenges in Digital Learning 11

2.3.1 Fragmented Learning Resources 11

2.3.2 Information Overload 11
2.3.3 Insufficient Study Management 12
2.3.4 Lack of Personalized Academic Support 12
2.4 Personalized and Adaptive Learning 12
2.4.1 Personalized Learning 12
2.4.2 Adaptive Learning 12
2.4.3 Learner Modelling 13
2.5 Educational Recommender Systems 13
2.5.1 Content-Based Recommendation 13
2.5.2 Collaborative Recommendation 13
2.5.3 Hybrid Recommendation 13
2.5.4 Recommendation as Learning Guidance 14
2.6 Intelligent Tutoring Systems and Artificial Intelligence in Education 14
2.7 Multimedia Learning and Human-Computer Interaction 14
2.7.1 Cognitive Theory of Multimedia Learning 14
2.7.2 Multimedia Design Principles 15
2.7.3 Human-Computer Interaction and Usability 15
2.8 Concept Mapping and Knowledge Visualization 15
2.8.1 Concept Maps 15
2.8.2 Knowledge Graphs 16
2.8.3 Relationship Between PCSMR and Knowledge Graphs 16
2.9 Audio Learning and Text-to-Speech 16
2.10 AI-Assisted Knowledge Extraction 16
2.10.1 Document Processing 16
2.10.2 Concept Identification 17

2.10.3 PCSMR Extraction 17
2.10.4 Validation and Reliability 17
2.10.5 Source Grounding 17
2.11 Proposed PCSMR Framework 17
2.11.1 Problem 18
2.11.2 Challenge 18
2.11.3 Solution 18
2.11.4 Mechanism 18
2.11.5 Result 18
2.11.6 PCSMR as a Common Representation 18
2.12 Integration of Existing Approaches 19
2.13 Comparative Analysis 20
2.14 Research Gap 20
2.15 Conceptual Framework 21
2.16 Summary 22
2.17 References 22
CHAPTER THREE 23
METHODOLOGY, REQUIREMENTS AND SYSTEM DESIGN 23
3.1 Introduction 23
3.2 System Development Methodology 23
3.3 Requirements Analysis 24
3.3.1 Functional requirements 24
3.3.2 Non-functional requirements and current design responses 25
3.4 System Architecture 26
3.5 PCSMR Knowledge Representation 27

3.6 AI-Assisted Extraction and Document Processing 28
3.7 Prerequisite Graph and Learning Path 29
3.8 Notes, Concept Maps, Quizzes and Mastery 29
3.9 Database Design..... 30
3.10 Security, Privacy and Data Integrity 31
3.11 Testing and Evaluation Plan 31
3.12 Chapter Summary 32
CHAPTER FOUR 33
SYSTEM IMPLEMENTATION, TESTING AND RESULTS 33
4.1 Introduction 33
4.2 Implementation Environment and Technology Stack 33
4.3 Overview of the Implemented System 35
4.4 Frontend Implementation 35
4.4.1 Authentication interface 36
4.4.2 Library and concept review 36
4.4.3 PDF upload workflow 36
4.4.4 Concept map and quiz interface 36
4.5 Backend and Database Implementation 37
4.6 PCSMR Extraction from Uploaded Learning Material 37
4.7 Prerequisite Graph and Study Ordering 39
4.8 Quiz Generation and Mastery Tracking 39
4.9 Testing Strategy 40
4.10 Test Cases and Recorded Results 40
4.11 Interpretation of Results 41
4.12 Implementation Limitations and Outstanding Improvements 42

4.13 Chapter Summary 43
CHAPTER FIVE 43
CONCLUSION AND RECOMMENDATIONS 43
5.1 Introduction 43
5.2 Summary of the Work 44
5.3 Conclusions 45
5.4 Limitations 46
5.5 Recommendations 47
5.6 Closing Remarks 48

CHAPTER ONE

INTRODUCTION

1.1 Background

Digital libraries and online learning platforms have significantly expanded access to educational resources. Students can now retrieve textbooks, lecture notes, research papers, and other materials without the constraints of physical location or opening hours. However, access alone does not guarantee effective learning. Learners frequently struggle to identify the most relevant material for their current knowledge state, to organise concepts into coherent mental models, to recognise prerequisite relationships, and to decide what to study next.

Traditional digital libraries are primarily designed for storage, indexing, and retrieval. They return documents or search results but leave the learner to reconstruct the underlying logic of the ideas contained in those documents. Learning management systems add course organisation, assessment, and progress monitoring, yet the explanation and sequencing of knowledge still depend largely on pre-defined course structures and human instructors.

Recent advances in artificial intelligence, knowledge graphs, educational recommender systems, and intelligent tutoring systems offer new possibilities. These technologies can support concept extraction, relationship modelling, personalised sequencing, and adaptive feedback. The present project integrates several of these capabilities into a single web-based system organised around a project-specific knowledge representation called the Problem–Challenge–Solution–Mechanism–Result (PCSMR) framework.

1.2 Problem Statement

Students using conventional digital libraries and online learning environments often face the following interrelated difficulties:

- Educational resources remain fragmented across multiple files and formats, making it hard to see conceptual connections.
- Search and retrieval systems typically return the same results for every user, ignoring individual progress, preferences, and knowledge gaps.
- Learners receive little structured support for identifying prerequisites or constructing an ordered study path.
- Progress monitoring, where it exists, is frequently limited to coarse completion indicators rather than concept-level mastery.
- AI-generated summaries or explanations, when available, are often ungrounded in the source material and lack a consistent explanatory structure that can drive further learning activities.

The technical problem addressed by this project is therefore to design, implement, and evaluate an intelligent digital library that converts uploaded learning material into structured, sourcegrounded PCSMR concept records, models prerequisite relationships, tracks learner mastery, and uses these representations to deliver adaptive study support and explainable recommendations.

1.3 Aim of the Study

The aim of this project is to design, implement, and evaluate an Intelligent Digital Library with a

Personalized Study Assistant that uses the PCSMR knowledge model to support conceptual understanding through AI-assisted extraction, knowledge-graph exploration, quiz-based mastery tracking, and adaptive learning recommendations.

1.4 Specific Objectives

The specific objectives of the project are:

1. To design and implement a secure digital repository that allows learners to upload, store, and manage educational PDF resources.
2. To develop an AI-assisted extraction pipeline that converts uploaded document text into structured PCSMR concept records (Problem, Challenge, Solution, Mechanism, and Result) with best-effort source traceability.
3. To represent concepts and their prerequisite relationships in a directed graph and to generate prerequisite-first learning paths.
4. To implement a personalized study assistant that generates multiple-choice quizzes from PCSMR records, scores learner attempts, and maintains per-concept mastery status.
5. To provide adaptive study support by combining prerequisite ordering, mastery signals, and concept presentation (structured notes and concept maps) so that learners receive ordered, progress-aware guidance.
6. To evaluate the functional correctness of the implemented system through automated and manual testing and to document its limitations with respect to extraction quality, authorization, and learning effectiveness.

1.5 Significance of the Study

The project contributes to the field of educational technology in three main ways. First, it proposes and operationalises the PCSMR framework as an intermediate knowledge representation that can drive multiple learning outputs (notes, maps, quizzes, and pathways) from the same structured record. Second, it demonstrates a practical integration of document processing, large-language-model extraction, graph-based ordering, and mastery tracking within a single modular web

application. Third, it provides a concrete prototype that can be examined, tested, and extended by subsequent students or researchers interested in knowledge-centred digital libraries and adaptive learning support.

For learners, the system offers a more structured way of engaging with uploaded material than simple document storage. For educators and developers, it illustrates both the opportunities and the engineering challenges of building source-grounded, AI-assisted study assistants.

1.6 Scope and Limitations

The scope of the implemented prototype is limited to the following:

- Text-based PDF documents (scanned or image-only PDFs are outside the current extraction capability).
- Single-user concept ownership and basic authentication (JWT-based sessions).
- AI-assisted drafting of PCSMR fields and quiz questions through a configurable provider interface, with human review and editing supported.
- Prerequisite graph construction with cycle detection and simple learning-path generation.
- Threshold-based mastery tracking derived from quiz scores (score ≥ 0.8 marked as mastered).
- Web-based React frontend and FastAPI backend with PostgreSQL persistence.

The following are explicitly outside the current scope or only partially addressed:

- Optical character recognition for scanned documents.
- Advanced collaborative or multi-user recommendation algorithms.
- Large-scale learner studies measuring learning gains.
- Production-grade record-level authorization across all endpoints.
- Persistent job queues for multi-process deployment.
- Audio narration, video generation, or complex multimedia output.

These boundaries are acknowledged so that claims made in later chapters remain consistent with the implemented system.

1.7 Organisation of the Report

The remainder of this report is organised as follows:

- Chapter 2 reviews the literature on digital libraries, personalized and adaptive learning, educational recommender systems, intelligent tutoring systems, multimedia learning, concept mapping, and AI-assisted knowledge extraction. It identifies the research gap that motivates the PCSMR-centred approach.
- Chapter 3 presents the methodology, requirements, system architecture, PCSMR knowledge model, database design, security considerations, and evaluation plan.
- Chapter 4 describes the implementation of the frontend, backend, extraction pipeline, prerequisite graph, quiz and mastery modules, together with the testing strategy and recorded results.
- Chapter 5 summarises the findings, draws conclusions, discusses limitations, and offers recommendations for future work.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

Digital libraries and online learning environments have expanded access to educational information by making electronic books, lecture notes, research papers, videos, and other resources available beyond the physical library. However, access to information does not automatically result in effective learning. Learners may still experience difficulty determining which resources are relevant, organizing materials, understanding relationships among concepts, identifying prerequisite knowledge, planning study activities, and obtaining individualized assistance when they encounter difficulties.

The proposed Intelligent Digital Library with Personalized Study Assistant and Smart Learning Recommendations addresses this broader problem by combining digital resource management with personalized learning support, recommendation, study management, visualization, and artificial-intelligence-assisted knowledge processing. The project is particularly concerned with moving beyond information presentation toward conceptual understanding.

A central project-specific framework, called Problem–Challenge–Solution–Mechanism–Result (PCSMR), is proposed to structure knowledge. PCSMR is not presented as an established theory in the existing literature. Instead, it is a conceptual model developed for this project. It organizes a concept around the problem that motivated it, the challenges involved, the solution proposed, the mechanism through which the solution operates, and the resulting outcome. The representation is intended to provide a common knowledge structure from which text, audio, visualization, concept maps, and adaptive recommendations can be generated.

This chapter reviews digital libraries, digital learning environments, educational recommender systems, personalized and adaptive learning, intelligent tutoring systems, artificial intelligence in education, multimedia learning, concept mapping, knowledge visualization, human-computer interaction, audio learning, and AI-assisted information extraction. It concludes by identifying the gap that motivates the proposed system.

2.2 Digital Libraries and Digital Learning Environments

2.2.1 Digital Libraries

Digital libraries provide organized access to collections of digital information and generally support functions such as storage, indexing, discovery, retrieval, metadata management, and access. For students, digital libraries can provide textbooks, scholarly publications, lecture materials, examination resources, and multimedia learning content.

The primary strength of a conventional digital library is information accessibility. However, finding a resource and understanding a resource are different activities. A student may successfully retrieve an article about a topic and still be unable to identify its central problem, understand why a particular solution was developed, or connect the topic with prerequisite knowledge. This

distinction motivates the proposed shift from resource-centred access toward knowledge-centred learning support.

2.2.2 Digital Learning Environments

Learning management systems and other digital learning environments extend resource access through functions such as course organization, assessment, communication, assignment management, and progress monitoring. These environments demonstrate the value of integrating educational resources with learner activity. Nevertheless, the organization and explanation of knowledge often remain dependent on predefined course structures and human instructors.

The proposed system extends this environment by treating uploaded learning materials as inputs to a knowledge-processing pipeline. Rather than leaving a PDF or lecture slide as a static file, the system aims to identify concepts and represent their relationships in a form that can support personalized learning.

2.3 Challenges in Digital Learning

2.3.1 Fragmented Learning Resources

A single topic may be distributed across lecture notes, textbooks, research papers, websites, videos, and other learning resources. Learners must often determine which source to use and manually establish connections between them. Fragmentation therefore creates both organizational and cognitive demands.

The proposed system addresses this issue by providing a centralized repository while also organizing extracted knowledge at the concept level. This creates the possibility of connecting information from different resources around common concepts and relationships.

2.3.2 Information Overload

Digital environments can expose learners to large volumes of potentially relevant information. The challenge is therefore not only finding information but filtering it according to the learner's current

objective, knowledge state, and context. Educational recommender systems are relevant because they can help prioritize resources and learning activities.

2.3.3 Insufficient Study Management

Independent learning requires learners to plan study sessions, prioritize topics, monitor progress, and adjust their activities. A personalized study assistant can support these activities through goals, schedules, reminders, progress indicators, and recommendations.

2.3.4 Lack of Personalized Academic Support

A resource repository does not necessarily explain difficult concepts or provide individualized feedback. Intelligent tutoring systems attempt to address this limitation through learner modelling, feedback, guidance, and personalized explanations. A 2025 systematic review of AIdriven intelligent tutoring systems examined 28 studies involving 4,597 students and found generally positive effects on learning and performance, while also identifying the need for stronger research designs and attention to ethical issues [1].

2.4 Personalized and Adaptive Learning

2.4.1 Personalized Learning

Personalized learning is a broad approach that accounts for differences among learners, including goals, prior knowledge, preferences, pace, performance, and learning context. Personalization may influence content selection, sequencing, difficulty, feedback, or presentation.

2.4.2 Adaptive Learning

Adaptive learning introduces computational mechanisms that modify aspects of the learning experience in response to learner information. Adaptation may involve content selection, learning sequence, difficulty, additional explanations, or recommendations.

For the proposed system, personalization should not depend on the assumption that learners possess fixed visual, auditory, or reading learning styles. Instead, learner preferences can be treated as one signal among several, together with interaction behaviour, performance, progress, resource

selection, and demonstrated difficulty. This provides a more defensible basis for adaptive presentation.

2.4.3 Learner Modelling

Adaptive systems require a representation of the learner. A learner model can contain completed topics, assessment performance, preferences, interaction history, inferred knowledge states, and areas of difficulty. In the proposed system, this model can influence which PCSMR explanations, prerequisites, resources, and presentation modes are prioritized.

2.5 Educational Recommender Systems

Educational recommender systems help learners select resources or activities from potentially large collections. Recommendation can use content characteristics, learner profiles, previous interactions, collaborative patterns, explicit preferences, knowledge state, or hybrid combinations.

2.5.1 Content-Based Recommendation

Content-based approaches recommend items whose characteristics are similar to resources previously preferred by a learner. In the proposed system, topic similarity and conceptual relationships can contribute to determining relevant resources.

2.5.2 Collaborative Recommendation

Collaborative approaches infer recommendations from patterns among users. A limitation is the cold-start problem, where insufficient interaction data exist for new learners. This motivates combining behavioural information with explicit content and knowledge relationships.

2.5.3 Hybrid Recommendation

Hybrid systems combine multiple recommendation strategies. For this project, a hybrid approach is appropriate because recommendations can combine learner behaviour, topic relevance, prerequisite relationships, progress, and PCSMR-derived knowledge.

2.5.4 Recommendation as Learning Guidance

The proposed system aims to recommend more than documents. It can recommend learning actions, such as reviewing a prerequisite concept, viewing a visual explanation, listening to an audio narrative, attempting a quiz, or progressing to a related topic. This shifts recommendation from simple resource ranking toward learning-path guidance.

2.6 Intelligent Tutoring Systems and Artificial Intelligence in Education

Intelligent Tutoring Systems (ITS) are computer-based learning environments designed to provide individualized instruction, guidance, feedback, or assessment. Artificial intelligence has expanded ITS capabilities through natural-language interaction, learner modelling, automated feedback, knowledge tracing, and content generation.

Recent research indicates that AI-driven ITS can produce positive learning outcomes, but AI alone does not guarantee educational effectiveness. Létourneau et al. reported generally positive effects while noting that advantages can be smaller when compared with non-intelligent tutoring systems and calling for stronger evaluation and attention to ethical implications [1].

This supports an important design principle for the proposed project: AI should operate within a pedagogically structured architecture. Rather than allowing an AI model to produce arbitrary explanations, the proposed system uses PCSMR as an intermediate knowledge representation. This can provide a consistent structure for generating multiple learning outputs.

2.7 Multimedia Learning and Human-Computer Interaction

2.7.1 Cognitive Theory of Multimedia Learning

Mayer's Cognitive Theory of Multimedia Learning provides a major theoretical foundation for learning from words and graphics. The theory describes dual channels for visual/pictorial and auditory/verbal processing, limited processing capacity, and active processing in which learners select, organize, and integrate information with prior knowledge [2].

This theory is relevant because the proposed application combines text, narration, diagrams, animations, and interactive representations. It also provides an important warning: adding more media does not automatically improve learning. Instructional material must be designed to support meaningful processing while minimizing unnecessary cognitive load.

2.7.2 Multimedia Design Principles

Research associated with multimedia learning identifies principles such as coherence, signalling, redundancy, spatial and temporal contiguity, segmenting, pre-training, modality, personalization, voice, and generative activity [2], [3]. These principles support the idea that the proposed visualizations should reveal relevant information progressively and emphasize relationships rather than fill the interface with decorative media.

For example, a PCSMR-based animation could begin by presenting a problem, introduce the challenge that prevents a straightforward solution, present the solution, demonstrate its mechanism step-by-step, and finally show the result. This sequence provides a causal narrative instead of presenting disconnected facts.

2.7.3 Human-Computer Interaction and Usability

Human-computer interaction concerns the design and evaluation of interactions between people and computational systems. The proposed application must allow learners to move between resources, PCSMR explanations, concept maps, audio, and visualizations without losing context. Usability should therefore be evaluated through effectiveness, efficiency, satisfaction, learnability, and the learner's ability to recover from errors or confusion.

2.8 Concept Mapping and Knowledge Visualization

2.8.1 Concept Maps

Concept maps represent concepts and their relationships through nodes and connecting relationships. They can make hierarchical and associative structures visible, allowing learners to see relationships that may be difficult to infer from linear text. A 2024 meta-analysis examining 78 studies reported a positive effect of concept maps on academic achievement [4]. A separate

meta-analysis investigated their effectiveness for science achievement across learners and contexts [5].

2.8.2 Knowledge Graphs

Knowledge graphs represent entities and relationships in graph structures. In an educational context, relationships may include prerequisite-of, part-of, depends-on, causes, related-to, or leads-to. This representation is useful for determining which concepts should be introduced before others.

2.8.3 Relationship Between PCSMR and Knowledge Graphs

PCSMR and knowledge graphs can serve complementary functions. PCSMR describes the internal explanatory structure of an individual concept, while a knowledge graph represents relationships between concepts. For example, an electric motor concept may have a PCSMR explanation describing electromagnetic force and continuous rotation, while the knowledge graph connects the motor to prerequisite concepts such as current, magnetic fields, force, and electromagnetic interaction.

2.9 Audio Learning and Text-to-Speech

Audio can provide an alternative channel for accessing explanations and can support accessibility and flexible learning. However, reading an entire document aloud is not equivalent to producing an effective educational explanation.

The proposed system therefore treats audio as an instructional narrative generated from the PCSMR representation. Instead of reading source material verbatim, the system can explain the problem, challenge, solution, mechanism, and result in a coherent sequence. This also helps ensure consistency between the audio explanation and other generated representations.

2.10 AI-Assisted Knowledge Extraction

2.10.1 Document Processing

The proposed application may receive PDFs, lecture notes, and other educational resources. The processing pipeline must extract usable text, segment the material, identify meaningful concepts, and prepare information for semantic analysis.

2.10.2 Concept Identification

Concept identification is the first major reasoning stage. The system should identify important concepts rather than treating an entire document as one undifferentiated body of text. A networking document, for example, may contain concepts such as packet switching, routing, transport protocols, addressing, and application protocols.

2.10.3 PCSMR Extraction

The central AI task is to convert unstructured educational content into structured Problem, Challenge, Solution, Mechanism, and Result fields. The objective is therefore more demanding than generic summarization. The system must determine which information explains motivation, constraints, proposed solutions, operation, and outcomes.

2.10.4 Validation and Reliability

Because PCSMR data feed several downstream components, extraction errors can propagate. An incorrect mechanism can create a misleading animation; an incorrect prerequisite can produce a poor recommendation; and an incorrect result can create a misleading explanation. The proposed system should therefore include schema validation, confidence information, source references, contradiction checks, and evaluation against manually reviewed examples.

2.10.5 Source Grounding

Generative AI can produce plausible statements that are not supported by source material. In an educational application this presents a significant reliability risk. The proposed system should therefore favour source-grounded extraction and, where possible, retain references to the source

passages from which each PCSMR element was derived. The system should distinguish between information explicitly supported by the material and relationships inferred by the AI.

2.11 Proposed PCSMR Framework

The Problem–Challenge–Solution–Mechanism–Result framework is proposed as the central knowledge representation for this project. It is intended to transform a concept from a static definition into a structured explanation of why it exists, what difficulty it addresses, how it works, and what it achieves.

2.11.1 Problem

The Problem identifies the need, deficiency, question, or situation that motivates a concept. It answers: What needed to be solved?

2.11.2 Challenge

The Challenge describes constraints, obstacles, complications, trade-offs, or conditions that make the problem difficult to solve. It answers: Why was the problem difficult?

2.11.3 Solution

The Solution identifies the approach, method, design, principle, or invention proposed to address the problem and challenge. It answers: What was proposed?

2.11.4 Mechanism

The Mechanism explains how the solution operates, including processes, components, sequences, relationships, and causal steps. It answers: How does it work?

2.11.5 Result

The Result describes the outcome achieved by applying the solution. It answers: What changed or what was accomplished?

2.11.6 PCSMR as a Common Representation

The proposed architecture treats PCSMR as an intermediate knowledge layer. Uploaded material is processed into concepts and PCSMR structures, which can then support multiple outputs. Structured notes can present the five elements directly; audio can narrate them; animations can reveal them sequentially; concept maps can visualize their relationships; and adaptive recommendation can use the resulting knowledge dependencies.

- Learning material → Concept Identification → AI-Assisted PCSMR Extraction → Validation → PCSMR Knowledge Model
- PCSMR Knowledge Model → Structured Notes / Audio / Visualization / Concept Maps
- PCSMR Knowledge Model + Learner Model → Adaptive Recommendations and Learning Pathways

2.12 Integration of Existing Approaches

The reviewed literature provides established foundations for most components of the proposed system. Digital libraries provide resource organization and access. Recommender systems provide mechanisms for filtering and selecting resources. Adaptive learning provides methods for tailoring learning experiences. Intelligent tutoring research provides individualized support. Multimedia learning provides evidence-based principles for combining words and graphics. Concept mapping provides mechanisms for exposing relationships. AI provides techniques for language understanding and structured information extraction.

The proposed contribution is therefore primarily an integration and evaluation contribution. The system aims to connect these capabilities through a shared PCSMR representation rather than implementing them as unrelated features.

- Learning materials are uploaded or imported.
- Concepts are identified from the material.
- AI extracts Problem, Challenge, Solution, Mechanism, and Result.

- The extraction is validated and stored.
- Concept relationships and prerequisites are represented.
- The learner model records progress, interaction, preferences, and performance.
- The adaptive engine selects appropriate resources and presentation modes.
- The system generates text, audio, visualization, concept maps, and learning recommendations.
- Learner interaction and performance feed back into the adaptive engine.

2.13 Comparative Analysis

Capability	Digital Library	Recommender	ITS	Multimedia	Concept Mapping	Proposed System
Resource access	High	Medium	Medium	Low	Low	High
Personalization	Limited	High	High	Medium	Medium	High
Recommendation	Limited	High	High	Low	Low	High
Intelligent explanation	Low	Medium	High	Medium	Low	High
Visualization	Limited	Limited	Medium	High	High	High
Audio explanation	Limited	Low	Medium	High	Low	High
Concept relationships	Limited	Medium	Medium	High	High	High
PCSMR representation	No	No	No	No	No	Proposed

Table 2.1 provides a high-level conceptual comparison. It should not be interpreted as a claim that every existing implementation lacks the listed capabilities. Its purpose is to show the principal focus of each approach and the proposed integration around PCSMR.

2.14 Research Gap

The literature demonstrates substantial progress in digital libraries, educational recommender systems, adaptive learning, intelligent tutoring, multimedia learning, concept mapping, and AIassisted education. However, these capabilities are frequently implemented as separate functions or evaluated around individual educational interventions.

First, conventional digital libraries primarily address access and retrieval, leaving learners to construct many conceptual relationships themselves. Second, educational recommenders can reduce information overload but may focus mainly on resource selection rather than explaining why a learner needs a particular resource at a particular point in a conceptual pathway. Third, adaptive systems depend on the quality of learner models and the pedagogical structure used to determine adaptation. Fourth, AI tutoring introduces powerful individualized interaction but also raises reliability, transparency, privacy, and hallucination concerns. Fifth, multimedia and concept mapping research provides evidence for particular representations, but these representations still require appropriate instructional sequencing and cognitive-load management.

The resulting research opportunity is to investigate whether a structured Problem–Challenge–Solution–Mechanism–Result representation can serve as a common intermediate layer connecting AI-assisted document understanding with adaptive presentation, concept visualization, audio explanation, and educational recommendation.

The project should not claim that PCSMR is inherently superior. Its value should be evaluated empirically. Appropriate evaluation measures may include PCSMR extraction accuracy, learner comprehension, recall, task completion, perceived usability, engagement, recommendation relevance, and learner satisfaction. This gives the proposed project a testable contribution rather than relying on the novelty of the framework alone.

2.15 Conceptual Framework

The conceptual framework derived from the literature is:

Learning Materials → Concept Identification → AI-Assisted PCSMR Extraction → Validation → PCSMR Knowledge Model → Knowledge Graph / Prerequisite Graph → Adaptive Learning

Engine → Text / Audio / Visualization / Concept Map / Recommendation → Learner Interaction → Performance and Behaviour Data → Adaptive Learning Engine.

The framework reflects the project's central proposition that the quality of learner-facing outputs depends substantially on the quality of the underlying structured knowledge representation. Consequently, AI extraction accuracy becomes a major technical and research concern.

2.16 Summary

This chapter reviewed the major literature areas supporting the proposed intelligent digital library. Digital libraries provide access to resources, recommender systems help learners select information, adaptive learning provides personalization, intelligent tutoring systems provide individualized assistance, multimedia learning provides principles for combining verbal and visual information, and concept mapping provides mechanisms for representing relationships among concepts. AI provides additional capabilities for document understanding and structured knowledge extraction.

The literature also highlights important challenges: access does not guarantee understanding; excessive information can increase selection difficulty; learners require support for planning and progress monitoring; and AI-generated educational content must be reliable and appropriately grounded. The PCSMR framework is proposed as a project-specific intermediate representation that captures Problem, Challenge, Solution, Mechanism, and Result. The framework is intended to connect AI-assisted knowledge extraction with adaptive learning, visualization, audio explanation, concept mapping, and recommendations.

The next chapter should translate this conceptual foundation into a concrete methodology covering system requirements, architecture, data collection, AI extraction design, learner modelling, recommendation methods, implementation technologies, testing, and evaluation.

2.17 References

[1] A. Létourneau, M. Deslandes Martineau, P. Charland, J. A. Karran, J. Boasen, and P. M. Léger, “A systematic review of AI-driven intelligent tutoring systems (ITS) in K-12 education,” npj Science of Learning, vol. 10, art. no. 29, May 2025, doi: 10.1038/s41539-025-00320-7.

[2] R. E. Mayer, “The past, present, and future of the Cognitive Theory of Multimedia Learning,” Educational Psychology Review, vol. 36, art. no. 8, Jan. 2024, doi: 10.1007/s10648023-09842-1.

[3] R. E. Mayer, “Using multimedia for e-learning,” Journal of Computer Assisted Learning, vol. 33, no. 5, pp. 403–423, 2017, doi: 10.1111/jcal.12197.

[4] “The impact of concept maps on academic achievement: A meta-analysis,” Heliyon, vol. 10, no. 1, art. no. e23290, Jan. 2024, doi: 10.1016/j.heliyon.2023.e23290.

[5] D. Anastasiou, C. N. Wirngo, and P. Bagos, “The effectiveness of concept maps on students’ achievement in science: A meta-analysis,” Educational Psychology Review, vol. 36, art. no. 39, Mar. 2024.

CHAPTER THREE

METHODOLOGY, REQUIREMENTS AND SYSTEM DESIGN

3.1 Introduction

This chapter describes the methodology, requirements, architecture, data model, and principal design decisions for the Intelligent Digital Library with Personalized Study Assistant and Smart Learning Recommendations. The system is organized around the PCSMR framework: Problem, Challenge, Solution, Mechanism, and Result. Instead of treating a learning resource only as a file to store or retrieve, the system represents a topic as a structured explanation of the need it addresses, the difficulty surrounding that need, the proposed solution, how the solution operates, and the outcome.

The chapter distinguishes implemented behaviour from intended or future behaviour. The reviewed code contains account registration and login, PDF upload and extraction, editable PCSMR concept records, prerequisite relationships, notes and concept-map rendering, multiplechoice quizzes, and quiz-based mastery records. Broader features such as advanced recommendation ranking, document search, pagination, and a document-management library should only be described as implemented if they have subsequently been added and verified.

3.2 System Development Methodology

The implementation is structured as a modular web application, with the user interface, API, database, and AI extraction provider separated into distinct responsibilities. This separation makes it possible to develop and test the core learning model without tying it to one AI vendor or one presentation format.

An iterative prototyping approach is a suitable description if it reflects the actual development process: implement a small workflow, exercise it, inspect failures, and refine the design. The code supports this style of incremental development through a replaceable extraction-provider interface, individual API routers, database migrations, and automated tests. The author must confirm that iterative prototyping accurately describes the work actually performed; if the project followed a different approved methodology, this section should be adjusted accordingly.

The main implementation sequence represented in the code is: define the PCSMR concept record; implement authentication and persistence; implement manual concept creation and editing; add PDF processing and AI-generated drafts; implement prerequisite edges and learning paths; add rendering and quizzes; then connect quiz scores to learner mastery. This sequence is a description of the architecture and code organization, not proof of the chronological order in which the work was done.

3.3 Requirements Analysis

3.3.1 Functional requirements

ID	Requirement	Description
FR1	Account management	A learner can register, sign in, retrieve their current account details, and access protected API routes using a bearer token.
FR2	Manual concept entry	A learner can create a PCSMR concept without relying on an AI service.
FR3	PDF ingestion	A learner can upload a PDF within configured size and contenttype restrictions and receive a processing status.

FR4	AI-assisted PCSMR drafting	The extraction provider attempts to produce a topic, five PCSMR fields, and a normalized difficulty value from extracted document text.

ID	Requirement	Description
FR5	Human review	A learner can edit an AI-generated concept and mark it as human-reviewed.
FR6	Concept presentation	The same PCSMR record can be returned as structured notes or as a five-stage flow diagram.
FR7	Prerequisite graph	A learner can connect concepts with prerequisite edges; duplicate links and edges that would create a cycle are rejected.
FR8	Learning path	The API can return a prerequisite-first sequence ending with a requested concept.
FR9	Quiz generation and assessment	The system can request generated multiple-choice questions, return options without exposing the answer in the question response, and score submitted answers.
FR10	Mastery tracking	Quiz performance updates a per-user mastery status and score for the relevant concept.
FR11	Source traceability	The data model can store source character offsets associated with a PCSMR field when the extraction process identifies a matching span.

3.3.2 Non-functional requirements and current design responses

Quality	Design response and qualification attribute
Maintainability	The extraction provider interface isolates AI-specific SDK calls and prompt parsing from routers and core graph logic.

Usability	The React interface provides separate views for the library, upload workflow, PCSMR understanding, flow diagram, notes, study path, quiz, and concept map.
Quality attribute	Design response and qualification
Responsiveness	PDF processing and quiz generation use FastAPI background tasks and a polling status pattern rather than keeping the initial request open for the full AI call.
Data integrity	Relational constraints include unique user email, unique prerequisite edge pairs, and one mastery record per user/concept pair. Graph cycle prevention is implemented in service logic.
Security	Password hashes and expiring JWT access tokens are used. However, the reviewed API does not consistently enforce record ownership on ID-based endpoints. Authorization hardening is therefore a required outstanding task, not a satisfied requirement.
Explainability	Source-span records can associate extracted fields with source offsets. The current implementation uses a simple substring match, so traceability is partial and should not be presented as guaranteed evidence for every AIgenerated field.

3.4 System Architecture

The application uses a client-server architecture. The frontend is implemented with React and Vite. The backend exposes HTTP API endpoints through FastAPI. SQLAlchemy's asynchronous session layer provides database access, while PostgreSQL is the intended relational database as reflected by the PostgreSQL UUID types and project configuration. PDF text extraction is handled by pypdf. AI extraction and quiz drafting are delegated to a provider interface that currently includes Gemini, Anthropic, and a manual fallback provider.

The main request and processing flow is summarized below:

Learner

↓

React + Vite interface

↓ HTTP / JSON + bearer token

FastAPI routers and authentication dependencies

↓

Application services: PCSMR, graph, rendering, quiz and job handling

├── pypdf: PDF bytes → extracted text

├── ExtractionProvider: Gemini / Anthropic / manual fallback

└── SQLAlchemy asynchronous sessions

↓

PostgreSQL tables

The frontend's API module centralizes requests and token handling. Backend routers separate account, document, concept, graph, rendering, overview, and mastery responsibilities. The architecture is modular, although the background-job registry is currently an in-process Python dictionary. This is adequate only for a single-process demonstration environment: job state can be lost on restart and is not automatically shared between multiple server workers.

3.5 PCSMR Knowledge Representation

The central Concept record stores a topic, optional document association, creator, five PCSMR text fields, difficulty, extraction status, and creation/update timestamps. The PCSMR fields are stored as text rather than as a rigid hierarchy, which keeps each concept self-contained and allows fields to be edited after extraction. The current model intentionally does not implement a parent-child concept hierarchy.

PCSMR field	Intended meaning
Problem	The need, gap, or problem that makes the concept relevant.
Challenge	The difficulty or constraint that makes the problem non-trivial.
Solution	The proposed approach to address the problem.
Mechanism	How the solution operates, ideally in a clear sequence.

Result	The outcome or effect attributed to the solution.

The extraction-status field distinguishes manually entered records, AI-generated drafts, and records marked as human-reviewed. This supports a human-in-the-loop workflow: AI output is treated as a draft that can be checked and corrected, rather than automatically assumed to be accurate.

3.6 AI-Assisted Extraction and Document Processing

When a learner uploads a PDF, the API checks the declared content type and configured maximum file size. The file bytes are read, a Document record is created with a pending status, and processing is scheduled as a FastAPI background task. The worker updates the status to processing, extracts page text with pypdf, and passes the resulting text to the selected extraction provider. On success, it stores a Concept in draft status, saves any source spans returned by the provider, and marks the document as done. If an exception occurs during processing, the document is marked failed.

The extraction prompt defines each PCSMR field and instructs the model to use only information supported by the source. If a field is not supported, the prompt requests the phrase 'Not stated in the source.' The response is expected as JSON. The parser strips common Markdown code fences, converts returned field values to strings, and normalizes difficulty to beginner, intermediate, or advanced. The provider boundary means the main application does not need to know which AI vendor implements the extraction call.

A key limitation is that source-span detection currently searches the extracted source text for the first 40 characters of a generated field. Since the model may paraphrase the source, a valid field often will not match literally. Therefore, source spans should be described as best-effort traceability, not guaranteed citation or semantic evidence. PDF text extraction can also be incomplete for scanned documents or complex layouts; optical character recognition is not evidenced in the reviewed code.

The worker runs the provider call in a background thread to avoid blocking the event loop during the synchronous network operation. The status polling approach keeps the initial upload response short. For deployment with multiple processes or persistent job requirements, the in-memory job registry should be replaced with a persistent job store or queue.

3.7 Prerequisite Graph and Learning Path

Prerequisite relationships are represented by the ConceptEdge table. Each edge contains a source concept, a target concept, an edge type, and a timestamp. An edge from A to B means that A is a prerequisite of B, so A should be learned before B. A unique constraint prevents duplicate ordered pairs.

Before an edge is inserted, the graph service checks whether the proposed edge would create a cycle. It begins traversing from the proposed target along existing outgoing edges. If it can reach the proposed source, adding the edge would close a cycle and is rejected. The service also rejects a self-edge. Direct prerequisites are retrieved from incoming edges; transitive prerequisites are found by repeatedly following prerequisite links. A learning path uses depth-first traversal and appends each prerequisite before the concept that depends on it.

The graph algorithm enforces ordering constraints within the stored edge set. However, the reviewed API currently needs stronger ownership checks before allowing a learner to create or inspect edges, because authentication alone does not establish that both concepts belong to the current user.

3.8 Notes, Concept Maps, Quizzes and Mastery

The rendering endpoint supports multiple representations of one Concept record. Notes rendering deterministically formats the topic and PCSMR fields as Markdown. Map rendering produces nodes for Problem, Challenge, Solution, Mechanism, and Result, connected in that sequence. This PCSMR flow map is distinct from the prerequisite graph, which links separate concepts.

Quiz generation sends a PCSMR draft to the configured provider and requests multiple-choice questions that assess understanding of why the concept exists and how it works. The current provider interface defaults to three questions. Generated quiz items store the question, correct answer, distractors, and associated PCSMR field. The learner-facing quiz endpoint shuffles the correct answer among the options and does not include the answer field in its response.

When an attempt is submitted, the API compares submitted answers with stored answers using case-insensitive, whitespace-trimmed string comparison. The score is the number of correct

submitted answers divided by the number of submitted answers; an empty answer list produces zero. A score of 0.8 or greater sets mastery to mastered, while a lower score sets it to in_progress. Mastery is stored per user and concept. This is a simple threshold-based mastery rule, not a full adaptive recommendation algorithm. The current evidence supports describing performance-based mastery tracking and prerequisite ordering; it does not yet support claiming sophisticated personalized recommendation ranking.

3.9 Database Design

The reviewed initial migration defines the following seven relational tables:

Table	Purpose
users	Account identity, unique email, password hash, learning preference, creation timestamp.
documents	Uploader, original filename, processing status, creation timestamp.
concepts	PCSMR content, difficulty, extraction status, creator, optional document link, timestamps.
concept_edges	Directed prerequisite links between concepts; duplicate ordered pairs are prohibited.
source_spans	PCSMR field name and start/end character offsets for a concept's source trace.
quiz_items	Question, correct answer, distractor options, PCSMR field, and associated concept.
mastery	Per-user/per-concept mastery status, optional score, and update timestamp.

UUID primary keys are used for the principal entities. Foreign keys connect concepts to their creator and optional source document, prerequisite edges to their two concepts, quiz items and source spans to concepts, and mastery records to users and concepts. The migration includes uniqueness constraints for user email, edge pairs, and the user/concept mastery pair. The reviewed schema does not define a separate table for quiz-attempt history, so the current design stores the latest mastery status and score rather than a full historical record of every attempt.

3.10 Security, Privacy and Data Integrity

Passwords are hashed with bcrypt and login uses signed JSON Web Tokens with an expiration time. Protected routes use a dependency that resolves the current user from the bearer token. Upload validation checks the declared content type and maximum file size before parsing the uploaded bytes. These are useful baseline controls, but they do not by themselves guarantee record-level authorization.

The source review identified an authorization gap: several ID-based routes fetch records by identifier without consistently filtering by the current user's ID. Examples include retrieving and editing a concept, retrieving a document and its concepts, rendering a concept, requesting quiz data or submitting an attempt, and manipulating prerequisite links. The job-status endpoint is also not protected by the current-user dependency, and job ownership is not represented in the inmemory registry. Before a public demonstration or deployment, these routes should verify ownership and return a consistent 404 or 403 response for unauthorized access. Edges should only connect concepts authorized for that user.

Additional hardening should include testing upload content beyond the client-provided MIME type, avoiding detailed internal exception messages in user-visible job results, setting production CORS origins explicitly, protecting secrets in environment variables, and evaluating safer token storage for a deployed browser application. These are recommendations from the code review and should not be described as already completed.

3.11 Testing and Evaluation Plan

The repository contains four pytest test functions across the inspected test files: an end-to-end flow test, provider-switching test, parser robustness test, and health-check test. The end-to-end test is designed to exercise registration and login, protected routes, concept creation and editing, notes/map rendering, prerequisite graph constraints, PDF processing, generated quiz handling, and mastery updates. The flow test substitutes a stand-in model for external AI calls, which makes the test more repeatable but does not measure the accuracy of live Gemini or Anthropic output.

The source archive review did not independently execute the test suite or verify a live database, so this chapter must not claim that the tests passed during this review. Before final submission, record the actual command used, environment and database details, pass/fail count, and any known skipped tests. Add authorization tests with two separate users: user A must not be able to read, edit, render, quiz, or create graph links involving user B's records. Also test malformed PDFs, scanned PDFs, empty extracted text, AI-provider failure, invalid model output, and boundary scores around the 0.8 mastery threshold.

A project evaluation should separate functional correctness from AI quality. Functional evaluation can measure whether workflows return the expected status codes and database state. AI extraction evaluation should use a manually reviewed sample of source documents and score each PCSMR field for factual support, coverage, and correctness. Source-span evaluation should measure whether offsets point to relevant source text. Quiz evaluation should assess answer correctness, distractor plausibility, and whether questions test conceptual understanding rather than simple recall. Report numerical results only after these evaluations have actually been carried out.

3.12 Chapter Summary

This chapter has described the implementation-oriented design of the Intelligent Digital Library. The system stores PCSMR concept records, extracts text from uploaded PDFs, uses a replaceable AI provider to draft structured explanations and quizzes, models prerequisites as a directed graph, and tracks mastery from quiz performance. The design supports several presentations of the same concept record and retains a human-review step for AI-generated drafts. The main outstanding engineering issue identified by the source review is consistent record-level authorization. Limitations in source-span matching, in-memory job tracking, and the simple mastery rule should be acknowledged when interpreting the system's current capabilities.

CHAPTER FOUR

SYSTEM IMPLEMENTATION, TESTING AND RESULTS

4.1 Introduction

This chapter describes the implementation of the Intelligent Digital Library with Personalized Study Assistant and Smart Learning Recommendations. It presents the technologies used, the major system modules, the workflow for transforming uploaded learning material into structured PCSMR concepts, and the implementation of prerequisite ordering, concept presentation, quizzes, and mastery tracking. It then summarizes the test cases and results recorded in the project's test materials, discusses limitations, and identifies additional evaluation needed before making claims about learning effectiveness.

The implementation follows a client-server design. The browser-based user interface is implemented with React and Vite, while the backend is implemented using FastAPI and Python. PostgreSQL stores the application records through SQLAlchemy, and Alembic manages schema migration. AI-assisted extraction and quiz drafting are isolated behind a provider interface so that the application can select a configured provider or use a manual fallback.

4.2 Implementation Environment and Technology Stack

Component	Technology / library	Role in the system
Frontend	React 18, Vite	Renders the application interface and coordinates user workflows.
Concept-map visualization	@xyflow/react	Displays nodes and relationships in an interactive graph-style view.
Backend API	Python, FastAPI	Provides authenticated HTTP endpoints for accounts, concepts, documents, graphs, rendering, and
Component	Technology / library	Role in the system
		mastery.

Database access	SQLAlchemy asynchronous sessions	Maps application entities to relational tables and executes database operations.
Database	PostgreSQL	Persists user accounts, documents, concepts, prerequisite edges, quiz items, source spans, and mastery.
Schema migration	Alembic	Defines and applies database schema changes.
PDF text extraction	pypdf	Extracts selectable text from uploaded PDF files.
AI extraction	Provider interface, including Gemini and other provider/fallback options	Drafts PCSMR fields and quiz content from supplied text.
Authentication	bcrypt password hashing and JWT access tokens	Protects credentials and authenticates requests to protected endpoints.
Testing	pytest and FastAPI test client/ASGI transport	Exercises backend workflows and error conditions.

The exact operating-system version, Python version used for the final test run, PostgreSQL version, Node.js version, AI model identifier, and deployment configuration should be entered from the actual development/test environment. The presence of a virtual environment in the submitted archive is not, by itself, evidence that every dependency version is appropriate for deployment.

4.3 Overview of the Implemented System

The system is organized into related modules rather than one monolithic application. The React interface uses a central API client to send requests to the FastAPI backend. The backend routes delegate work to models, database sessions, and service functions. Uploaded PDFs are processed in a background task, while the browser checks processing status and requests the resulting concept records.

Module	Implemented responsibility

Authentication	Registration, login, current-user lookup, password hashing, and token-based access.
Concept management	Create, list, retrieve, and edit structured PCSMR concept records.
Document processing	Accept PDF uploads, store document metadata/status, extract text, and trigger AI-assisted drafting.
AI extraction service	Build prompts, call the configured provider, parse model output, normalize fields, and return concept data.
Prerequisite graph	Create and query directed prerequisite relationships, reject duplicate/self/cyclic links, and compute learning order.
Rendering	Represent a PCSMR concept as formatted notes or as a five-stage map.
Quiz and mastery	Generate multiple-choice questions, accept answers, calculate a score, and update per-user mastery.
Overview and learner interface	Show library items, concepts, mastery state, study-path information, upload progress, and concept/quiz views.

4.4 Frontend Implementation

The frontend is a React single-page application. The main application component coordinates authentication state and navigation between the library, PDF upload, concept map, and individual concept views. The API module centralizes requests, attaches the saved bearer token, and handles unauthorized responses by clearing the token and notifying the application that the session has ended.

4.4.1 Authentication interface

The authentication component provides registration and login workflows. After successful authentication, the application can request the current-user endpoint to restore the session. Protected actions depend on the access token being sent with API requests. In the reviewed implementation, the token is stored in browser local storage. This is convenient for a prototype,

but token storage should be reviewed as part of production security hardening because browseraccessible storage can be read by injected JavaScript.

4.4.2 Library and concept review

The library component presents concept records retrieved through the API. A learner can open a concept to inspect its PCSMR content and use the concept view for further study. The concept workflow supports AI-drafted records and manual edits, allowing generated content to be checked rather than accepted without review.

4.4.3 PDF upload workflow

The upload component sends a PDF to the backend and receives a document/job status. Since extraction can take longer than a normal API request, the interface polls the status endpoint and can update the user when processing finishes or fails. This separates the initial file-transfer response from the time needed for text extraction and model inference.

4.4.4 Concept map and quiz interface

The concept-map component uses @xyflow/react to display nodes and connecting edges. The PCSMR map represents the sequence Problem → Challenge → Solution → Mechanism → Result. This is different from the prerequisite graph, which represents dependencies between separate concepts. The quiz panel presents multiple-choice questions and submits the learner's answers for scoring and mastery updates.

Insert screenshots captured from the running application in the final report. Each screenshot should have a numbered caption and should show the actual interface rather than a design mockup.

4.5 Backend and Database Implementation

The FastAPI backend separates routes by responsibility. Authentication routes manage accounts; concept routes manage PCSMR records; document routes accept uploads and expose processing status; graph routes manage prerequisites and learning paths; rendering routes provide notes, maps,

and quiz content; mastery routes process quiz attempts and return learner state; and overview routes aggregate information for the interface.

The relational schema includes users, documents, concepts, concept_edges, source_spans, quiz_items, and mastery. UUID identifiers are used for principal entities. Foreign keys link concepts to creators and optional source documents, graph edges to concepts, quiz items and source spans to concepts, and mastery records to users and concepts. Uniqueness constraints prevent duplicate user email addresses, duplicate ordered prerequisite pairs, and duplicate mastery records for a given user/concept pair.

Alembic provides the initial schema migration, making the database structure reproducible in a configured environment. The database is the persistent source of truth for concepts, documents, quiz items, and mastery. The current background job-status registry, however, is held in process memory rather than in a persistent queue. Job information can therefore be lost when the server restarts, and multiple server processes would not automatically share the same registry.

4.6 PCSMR Extraction from Uploaded Learning Material

The PDF processing pipeline begins when the learner uploads a PDF. The backend validates the declared content type and maximum upload size, creates a document record with a pending status, and schedules processing as a background task. The worker updates the status, extracts text from the PDF using pypdf, submits the extracted text to the configured provider, parses the result, saves the PCSMR concept as a draft, and updates the document status. Failures are reflected in the document's processing status.

Stage	Operation	Output
1. Upload	Receive PDF bytes and validate basic upload restrictions.	Document record and initial status.
2. Text extraction	Use pypdf to extract selectable text from the document.	Plain-text content for the model.
3. Prompt construction	Explain the PCSMR fields and request sourcegrounded structured output.	Provider request.

4. Model inference	Call the selected extraction provider.	Model-generated response.
5. Parsing and normalization	Remove common code fences, parse JSON, flatten supported list values, and normalize difficulty.	Structured PCSMR fields.
6. Persistence	Save the concept as a draft and save source-span data where found.	Concept and traceability records.
7. Completion	Update document status and expose the result through the API.	Status available to the frontend.

The extraction prompt requests the model to use information supported by the supplied source and to identify fields not stated in the source. The extraction service is separated behind a provider interface, so the application can select a provider through configuration rather than embedding vendor-specific logic throughout the API.

The source-span mechanism attempts to associate PCSMR fields with source character offsets by searching the extracted text for a short portion of the generated field. This is a best-effort technique. When the model paraphrases the original text, a literal match may not exist. The current implementation should therefore not be described as guaranteeing exact evidence for every generated statement. Similarly, pypdf extracts text from PDFs with an accessible text layer; the reviewed code does not demonstrate an OCR pipeline for image-only scanned documents.

4.7 Prerequisite Graph and Study Ordering

The prerequisite graph models relationships between concepts. If concept A must be understood before concept B, the directed edge represents A as a prerequisite of B. When a new edge is requested, the graph service checks whether the proposed connection would create a cycle. It rejects self-links and duplicate links and traverses existing edges to identify whether the target can already reach the proposed source.

The learning-path function follows prerequisite relationships and places prerequisite concepts before the requested target. This gives the learner an explicit order in which to approach connected

material. The mechanism is graph-based ordering, not a claim that the system has learned an optimal sequence from large-scale learner data.

4.8 Quiz Generation and Mastery Tracking

The quiz workflow asks the configured provider to generate multiple-choice questions from a PCSMR concept. Quiz records store the question, answer, distractors, and associated PCSMR field. The API response used by the learner interface omits the stored correct answer and presents the available options in shuffled order.

On submission, the backend compares each submitted answer with the stored answer after trimming whitespace and ignoring letter case. The score is computed as the number of correct submitted answers divided by the number of submitted answers. An empty submission receives a score of zero. A score of 0.8 or higher marks the concept as mastered; a lower score marks it as in progress. Mastery is recorded per learner and concept, and is then available to other views such as the library, study path, or graph.

This is a transparent threshold-based mastery rule. It is not, by itself, evidence of a sophisticated adaptive recommender. A future evaluation should determine whether the threshold is educationally appropriate and whether generated questions reliably measure understanding of PCSMR concepts.

4.9 Testing Strategy

The backend repository includes pytest tests intended to exercise integrated API workflows and focused behaviour. The end-to-end test uses an ASGI transport and a configured PostgreSQL database. External model calls in the principal end-to-end flow are replaced with a stand-in model so that expected behaviour can be tested without depending on network availability, provider quotas, or changing model responses. Separate tests exercise provider selection and parser robustness, and a health-check test checks basic service availability.

The project presentation records four automated backend tests passing, covering eight grouped test areas, and reports a separate live Gemini extraction check on a sample passage. These are project-recorded results, not independently reproduced results from this document preparation. Before

submission, rerun the latest tests and confirm the outcomes, because the code archive and the presentation may not represent the exact same revision.

4.10 Test Cases and Recorded Results

Test ID	Test case	Recorded status	Evidence / qualification
TC-01	Register/login, duplicate email, incorrect password, and protectedroute behaviour.	PASS recorded	End-to-end automated test.
TC-02	Create/edit concepts, review status, missing identifiers, and malformed identifiers.	PASS recorded	End-to-end automated test.
TC-03	Reject duplicate, reverse, indirectcycle, and self prerequisite links.	PASS recorded	End-to-end automated test.
TC-04	Return prerequisite-first study order for a three-concept chain.	PASS recorded	End-to-end automated test.
TC-Test ID	Upload a PDF, process it in the Test case	PASS Recorded status	Stand-in model used for the Evidence / qualification
05	background, and create a draft principal integration flow. concept with source-span records.	recorded	
TC-06	Parse model output containing code fences, list values, and an invalid recorded difficulty label.	PASS	Provider/parser test.
TC-07	Generate quiz questions, avoid exposing answers, score attempts, update mastery, and avoid duplicate quiz records on regeneration.	PASS recorded	End-to-end automated test.
TC-	Switch extraction providers and	PASS	Provider configuration test.

08	report a clear error when a required recorded key is missing.		
TC-09	Perform a live Gemini extraction request on a sample passage.	PASS recorded; independently reproduced and retained its output before final submission.	Project presentation reports a separate live check;

The table uses 'PASS recorded' rather than presenting these outcomes as independently verified here. For the final report, update the status column after running the tests on the current code revision. Attach the relevant console output or test report in an appendix if permitted by the department.

4.11 Interpretation of Results

The recorded test cases indicate that the principal backend workflows have been designed and exercised: authentication, concept management, prerequisite validation, learning-path generation, PDF processing, AI-response parsing, quiz handling, and mastery updates. The use of a stand-in model makes the core application tests more deterministic. A separate live-provider check is useful because it exercises the external integration, but one sample passage cannot establish general extraction accuracy.

The current evidence is strongest for functional workflow behaviour. It does not yet quantify PCSMR factual accuracy, field completeness, source-span precision, quiz quality, system performance under load, or improvement in student learning. Those questions require a documented evaluation dataset, explicit metrics, and recorded observations. No numerical accuracy, response-time, usability, or learning-gain claims should be added without measurements.

4.12 Implementation Limitations and Outstanding Improvements

Limitation	Effect and recommended improvement

Record-level authorization	Some ID-based endpoints do not consistently verify that a concept, document, quiz, job, or both ends of a graph edge belong to the authenticated user. Add ownership checks and multi-user authorization tests before deployment.
In-memory job state	The job registry is process-local and may be lost on restart or diverge across server workers. Consider a persistent job table or queue for deployment.
PDF scope	Text extraction depends on text being available in the PDF. Scanned pages and complex layouts may require OCR or additional document processing.
Source traceability	Literal substring matching is not reliable for paraphrased generated fields. A stronger approach would store quoted source passages or evidence spans selected and validated against the original text.
Recommendation sophistication	The current behaviour supports prerequisite-first learning order and rule-based mastery indicators. A broader hybrid or behaviour-driven recommendation engine is not evidenced by the reviewed code.

Limitation	Effect and recommended improvement
Mastery model	A single threshold and latest score provide a basic signal, but do not capture attempt history, confidence, time, or uncertainty.
AI quality evidence	Automated parser/provider tests do not establish factual accuracy of generated PCSMR fields or educational quality of generated quizzes.
Security hardening	Review token storage, production CORS settings, exception disclosure, upload validation, secrets management, and per-user access boundaries.

4.13 Chapter Summary

This chapter has described the implementation of the Intelligent Digital Library, including the React interface, FastAPI backend, PostgreSQL data model, PDF processing pipeline, AI-assisted PCSMR extraction, prerequisite graph, concept renderers, quiz workflow, and mastery tracking. The project materials record automated tests covering the principal backend workflows and a separate live-provider extraction check. The results support a claim that the prototype implements and exercises its main functional flows, subject to verification against the latest code revision. They do not yet establish general AI accuracy or improved learning outcomes. The most important next steps are to enforce record-level authorization consistently, re-run and document tests, and conduct a small, reproducible evaluation of extraction and quiz quality.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This chapter concludes the report on the Intelligent Digital Library with Personalized Study Assistant and Smart Learning Recommendations. It restates the aim and objectives, summarises the principal findings from the design and implementation work, draws overall conclusions, acknowledges limitations, and presents recommendations for further development and research.

5.2 Summary of the Work

The project set out to create a web-based system that moves beyond document storage and retrieval toward structured conceptual support for learners. The central design decision was the adoption of the Problem–Challenge–Solution–Mechanism–Result (PCSMR) framework as a common intermediate knowledge representation. Uploaded PDF material is processed into PCSMR concept records; these records then support structured notes, five-stage concept maps, multiple-choice quizzes, prerequisite-aware learning paths, and per-learner mastery tracking.

The implemented system comprises:

- A React and Vite single-page application providing authentication, library views, PDF upload with status polling, concept review and editing, interactive concept maps, and quiz interfaces.
- A FastAPI backend organised into modular routers and services for authentication, document processing, concept management, prerequisite graph operations, rendering, quiz generation, and mastery updates.
- A PostgreSQL relational schema storing users, documents, concepts, prerequisite edges, source spans, quiz items, and mastery records, managed through SQLAlchemy and Alembic.
- An AI extraction service isolated behind a provider interface, enabling the use of external models (including Gemini) or a manual fallback while keeping the rest of the application independent of any single vendor.
- Automated backend tests covering authentication, concept management, graph constraints, PDF processing (with a stand-in model), quiz scoring, and mastery updates, together with a reported live-provider extraction check.

The functional tests recorded in the project materials indicate that the principal workflows operate as designed. The system successfully demonstrates an end-to-end path from PDF upload through PCSMR drafting, human review, graph construction, quiz assessment, and mastery update.

5.3 Conclusions

The following conclusions are drawn from the design, implementation, and testing work:

1. PCSMR as a unifying representation is feasible. A single structured concept record can drive multiple pedagogical outputs—notes, maps, quizzes, and ordered study paths—without requiring separate content pipelines for each modality. This supports the claim that a carefully designed intermediate knowledge model can improve coherence across learning features.
2. AI-assisted extraction can accelerate concept drafting but cannot yet be treated as fully reliable. The provider interface and parsing layer successfully produce draft PCSMR fields

and quiz questions. However, source-span matching remains best-effort, and no quantitative evaluation of factual accuracy or educational quality was conducted within the project. AI output must therefore remain subject to human review.

3. Prerequisite graph ordering provides transparent adaptive guidance. Cycle detection and prerequisite-first path generation give learners an explicit, explainable sequence. This is a rule-based form of adaptation grounded in the knowledge structure rather than opaque ranking.

4. Threshold-based mastery tracking supplies a usable progress signal. Linking quiz performance to a per-concept mastery status allows the interface to reflect learner progress and to influence study-path presentation. The 0.8 threshold is simple and transparent, though its educational optimality has not been validated.

5. Modular architecture supports iterative development and future extension. Separation of the frontend, API layer, domain services, database, and AI provider interface made it possible to develop and test individual capabilities while keeping the overall system coherent.

6. Engineering gaps remain before production deployment. Incomplete record-level authorization, in-memory job state, limited PDF robustness, and the absence of a formal extraction-quality evaluation mean that the current prototype is suitable for demonstration and further research rather than unrestricted multi-user deployment.

Overall, the project has successfully realised a working prototype of a personalized study assistant with adaptive capability inside an intelligent digital library. The contribution lies primarily in the integration of PCSMR extraction, graph-based ordering, and mastery-aware presentation rather than in any single algorithmic novelty.

5.4 Limitations

The principal limitations of the work are:

- Extraction quality evidence is limited. Automated tests verify parsing and workflow behaviour; they do not measure factual correctness, completeness, or educational usefulness of the generated PCSMR fields and quiz questions.
- Authorization is incomplete. Several identifier-based endpoints do not consistently enforce ownership, creating a security gap for multi-user scenarios.
- Job state is not durable. The background processing registry resides in process memory and is lost on server restart.
- PDF support is restricted to text-layer documents. Scanned or image-only PDFs are not handled.
- Mastery model is elementary. Only the latest quiz score and a fixed threshold are used; attempt history, confidence, time-on-task, and uncertainty are not modelled.
- No learner-outcome study was conducted. Claims about improved learning efficiency or academic performance cannot be supported by the present evidence.
- Recommendation sophistication is modest. Adaptation currently consists of prerequisite ordering and mastery indicators rather than a hybrid behavioural recommender.

These limitations are reported so that readers can interpret the results appropriately and so that future work can target the most important gaps.

5.5 Recommendations

5.5.1 Immediate Engineering Improvements

1. Implement consistent record-level authorization on every identifier-based endpoint and add multi-user authorization tests.
2. Replace the in-memory job registry with a persistent job table or queue so that processing state survives restarts and can be shared across workers.
3. Strengthen upload validation (file signatures, page limits) and surface clearer user-facing messages for empty or unreadable PDFs.

4. Improve source-span handling by storing validated evidence quotes rather than relying solely on literal substring matching.
5. Conduct and document a small, manually annotated evaluation of PCSMR extraction accuracy and quiz quality on a defined sample of educational passages.

5.5.2 Extensions to the Personalized Study Assistant

1. Enrich the learner model with attempt history, recency, and optional confidence indicators so that mastery estimates become more nuanced.
2. Develop explainable recommendation rules that combine prerequisite eligibility, mastery gap, and recent activity, and expose the reason for each suggestion to the learner.
3. Add support for additional presentation formats (for example, audio narration generated from PCSMR fields) while preserving the same underlying knowledge record.
4. Introduce a lightweight review workflow that allows a lecturer or peer to mark concepts as approved, thereby improving the reliability of shared resources.

5.5.3 Future Research Directions

1. Conduct a controlled usability and learning study comparing the PCSMR-based assistant with a conventional digital library interface, measuring task completion, comprehension, and perceived usefulness.
2. Investigate the suitability of the PCSMR schema across a wider range of subject domains and concept types, refining field definitions and status values where necessary.
3. Explore hybrid recommendation approaches that incorporate collaborative signals once a larger user base is available, while retaining transparency and prerequisite constraints.
4. Examine the privacy and ethical implications of sending educational content to external AI providers and develop clearer consent and data-minimisation practices.

5.6 Closing Remarks

The Intelligent Digital Library with Personalized Study Assistant demonstrates that a structured knowledge representation such as PCSMR can serve as a practical bridge between document ingestion and adaptive learning support. The prototype successfully implements the core technical pipeline—from PDF upload through concept extraction, graph construction, quiz assessment, and mastery tracking—and thereby provides a concrete foundation for further engineering and educational research. Future work that addresses authorization, extraction quality evaluation, and learner-outcome measurement will determine how far the approach can be extended beyond the current prototype.